Autoencoders

Chapter 14

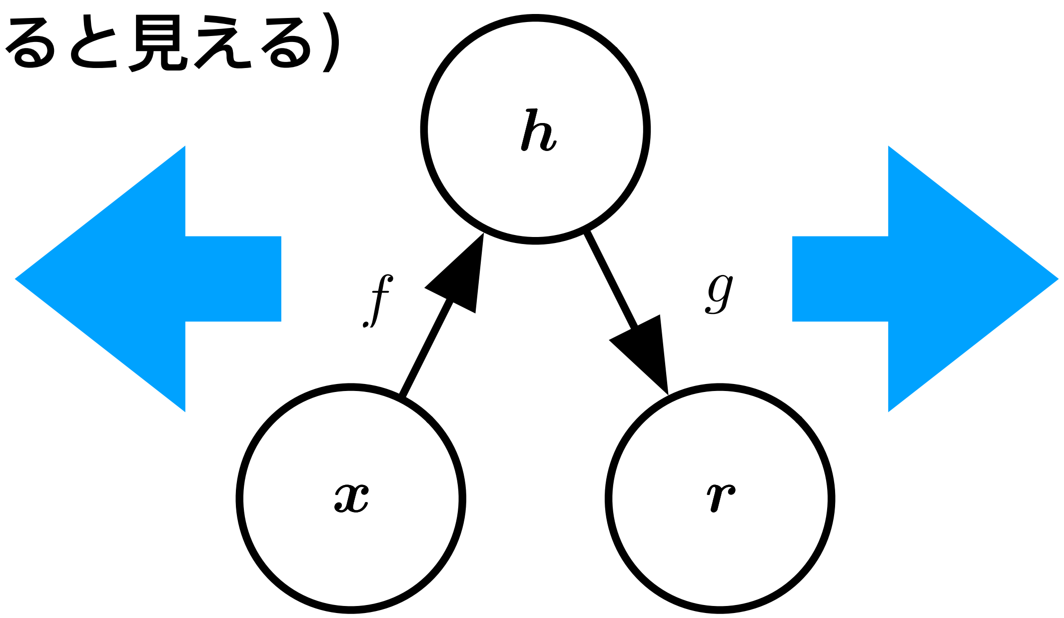
楊　森（Sen Yang）2019/07/28

Autoencoders

・Autoencoderは、inputを、outputにコピーするneural network。
・丸コピーは意味ないので、いろんな制限がある。
・これらの制限で、autoencoderはコピーできるように、データの役に立つプロパティを学習できる。

現代autoencoderはencoderとdecoderを、決定的関数から、確率的マッピングに一般化した：
（ノイズ注入を含めるので、outputが分布からサンプリングされると見える）

$p_{encoder}(h \mid x)$



$p_{decoder}(x \mid h)$

Autoencoderの応用：
・次元削減（dimensionality reduction）
・情報検索（information retrieval）

Undercomplete Autoencoders

・考え方①：h（code）dimensionを制限する（input dimensionと比べる）
・そうすると、autoencoderは一番顕著な（salient）特徴を学習しないといけない

Autoencoderのloss function: $$L(x, g(f(x)))$$

・線形decoder、LがMSEの場合、undercomplete autoencoderはPCAと同じ結果を出す。
・非線形encoder、非線形decoderのcapacityが大きすぎると、undercomplete autoencoderはデータ分布に関する情報を学習しないことがある。
　　・例えば、強すぎるnonlinear encoderがx(i)をcode iに表して、decoderがcode iからサンプルにマップできるけど、code iには何の役に立つ情報もない。

Regularized（正則化された）Autoencoders

・考え方②：モデルがコピー以外のプロパティを持つのを促す**loss function**を使う
・**encoder**や**decoder**を浅く、**code size**を小さくする制限はしない
・プロパティは以下の種類がある
　・**sparsity of the representation**（まばらな表現）
　・**smallness of the derivative of the representation**（小さい導関数の表現）
　・**robustness to noise**（ノイズへの頑健性）**or to missing inputs**（見つからないインプットへの頑健性）

Sparse Autoencoder:

$$L(x, g(f(x))) + \Omega(h) \qquad \Omega(h) = \lambda \sum_i |h_i|$$

Denoising Autoencoder:

$$L(x, g(f(\tilde{x})))$$

$\tilde{x}$ はノイズを入れた後の**x**のコピー

Contractive（収縮性の） Autoencoder:

$$L(x, g(f(x))) + \Omega(h, x) \qquad \Omega(h, x) = \lambda \sum_i \|\nabla_x h_i\|^2$$

Sparse Autoencodersへの考え方

Sparsity penaltyがコピータスクのregularizerであるのではなく、sparse autoencoderが、latent variableを持つgenerative modelの最尤推定であると考える。（Rather than thinking of the sparsity penalty as a regularizer for the copying task, we can think of the entire sparse autoencoder framework as approximating maximum likelihood training of a generative model that has latent variables. p496）

$$\log p_{model}(x) = \log \sum_h p_{model}(h, x)$$

一つの高い可能性のhの点推定で近似する

$$\log p_{model}(h, x) = \log p_{model}(h) + \log p_{model}(x|h)$$

sparsityを誘導するために、Laplace priorを使う

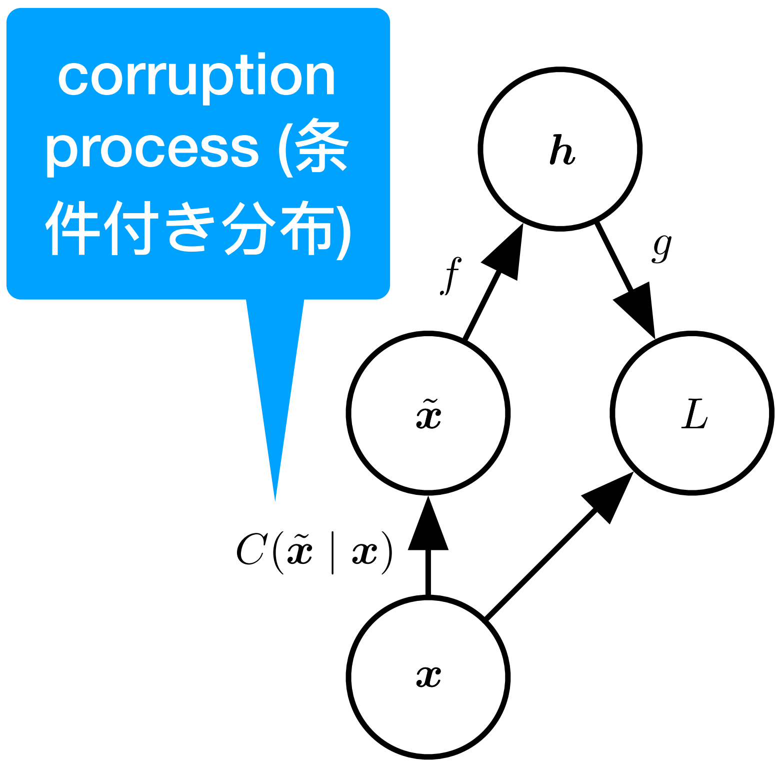
（sparsityが強い場合、点推定も近似できるから？）

$$p_{model}(h) = \frac{\lambda}{2} e^{-\lambda|h_i|}$$

$$-\log p_{model}(h) = \sum_i (\lambda|h_i| - \log\frac{\lambda}{2}) = \Omega(h) + const$$

autoencoderをトレーニングするのは、生成モデルを近似的にトレーニングする方法の一種

Denoising Autoencoders



期待（Expectation）：

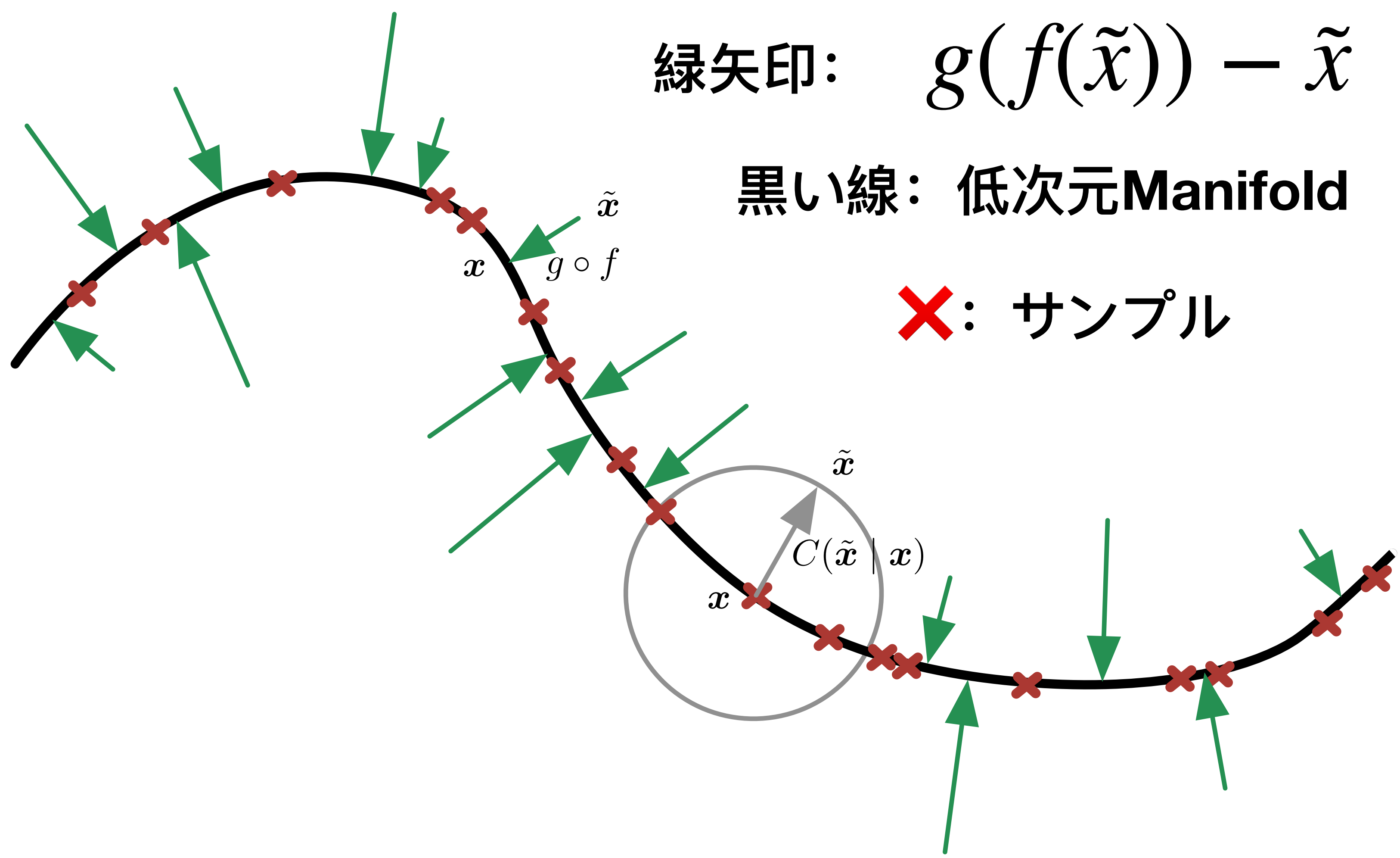
$$-\mathbb{E}_{x\sim\hat{p}_{data}(x)}\mathbb{E}_{\tilde{x}\sim C(\tilde{x}|x)} log p_{decoder}(x\,|\,h=f(\tilde{x}))$$

DAEの一番強いプロパティは、$g(f(x)) - x$ **ベクトル場（緑矢印）を学習できること。**

緑矢印： $g(f(\tilde{x})) - \tilde{x}$

黒い線：低次元Manifold

×：サンプル



Denoising Autoencoder and Score Matching

Denoising Autoencoderが学習した $g(f(x)) - x$ **ベクトル場（vector field）は、**$\nabla_x log p_{data}(x)$**スコア（gradient field）を予測する。**

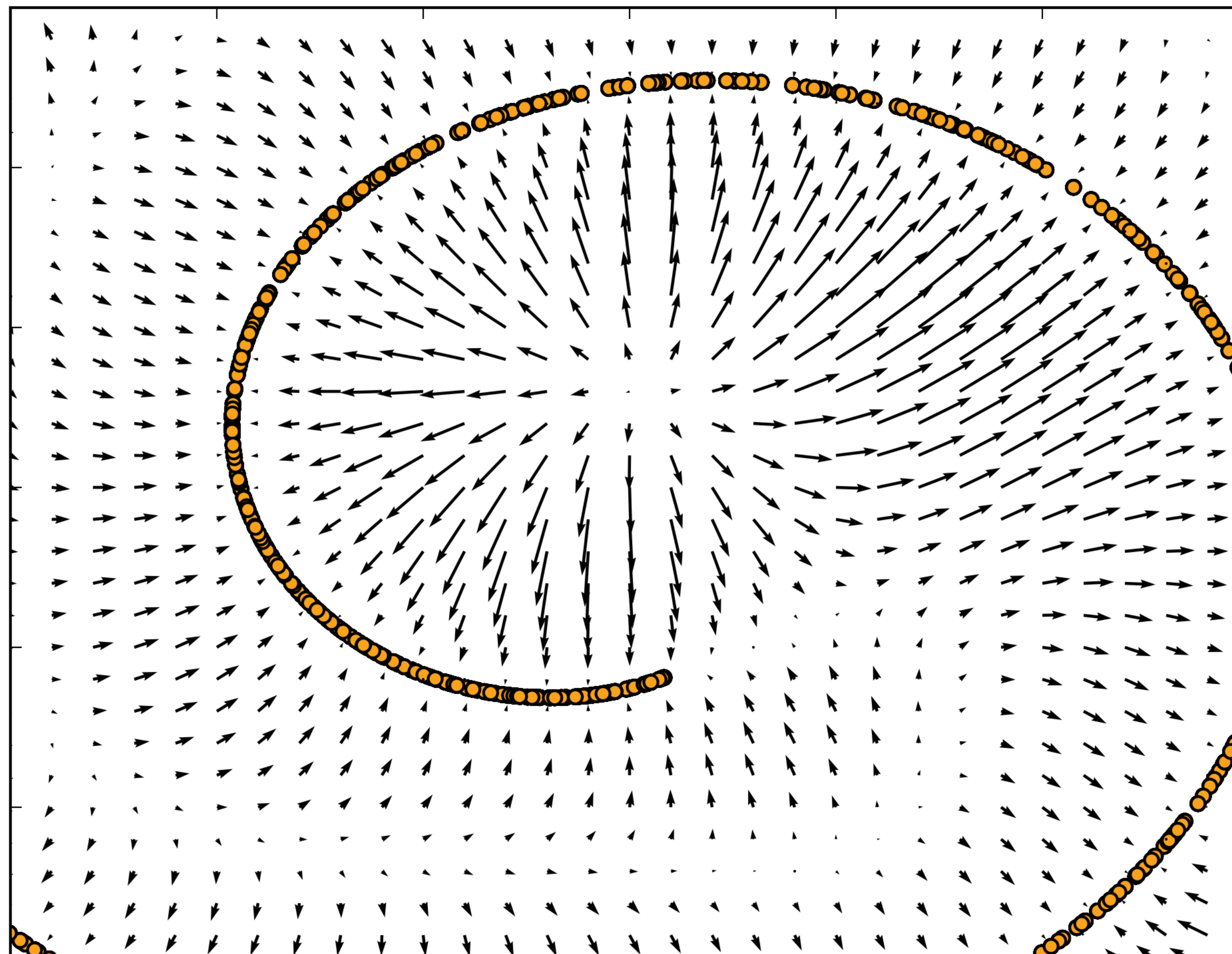
$log p_{data}$ **勾配場を学習するのは、** p_{data}**自体の構造を学習する方法だ。**

前提：Gaussian corruption $C(\tilde{x} = \tilde{x} \mid x) = \mathcal{N}(\tilde{x}; \mu = x, \sum = \sigma^2 I)$

ベクトル場とスコアの関係：σ->0

$$\frac{r_\sigma(x) - x}{\sigma^2} \rightarrow \frac{\partial log p(x)}{\partial x}$$

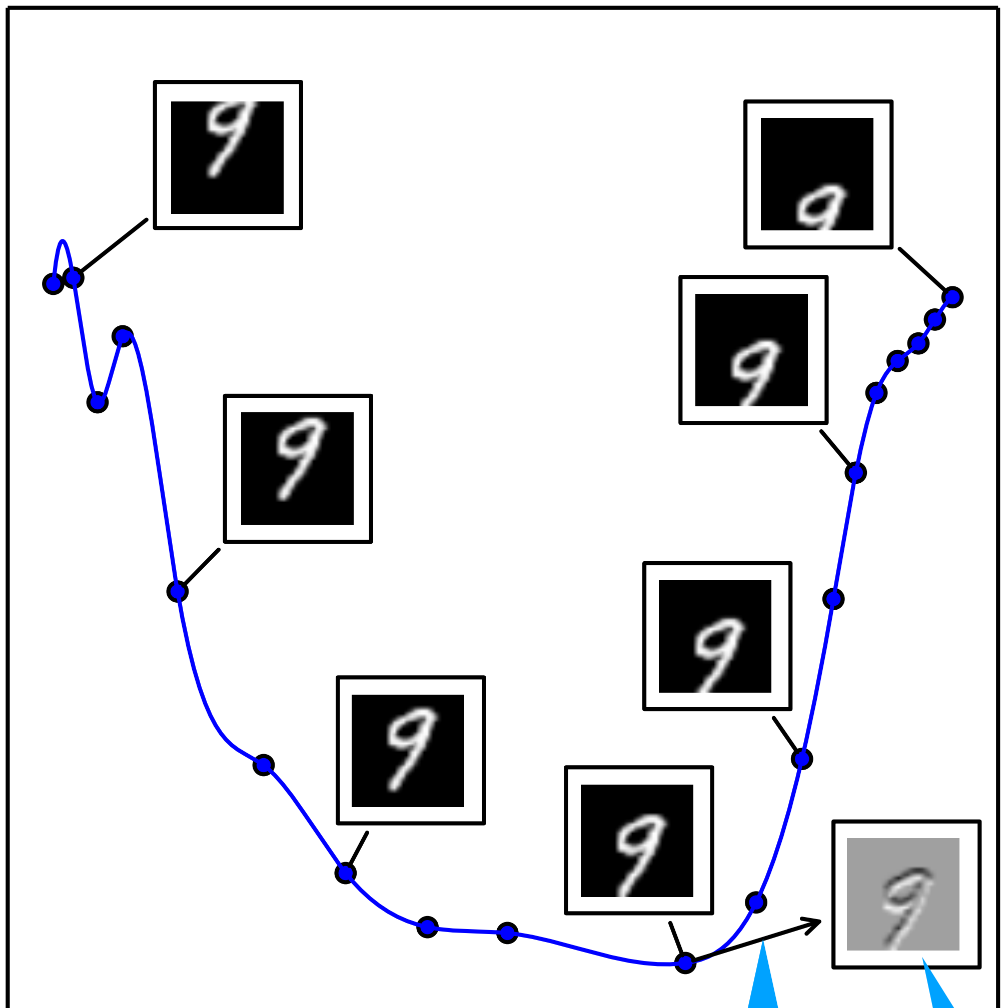
参考：Alain, G. and Bengio, Y. (2013). What regularized auto-encoders learn from the data generating distribution. In *ICLR'2013*



Denoising Autoencoder
が学習した $r(x) - x$
ベクトル場

Learning Manifolds with Autoencoders①

Manifold（多様体）の大切特性：接平面（tangent plane）の集合。d次元のmanifoldの接平面は、d基底から構成され、ローカル変化（variation）方向を示す。



Autoencoderトレーニングプロセスの2つ力：

① Decoderでxを再生できる表現hを学習する
→inputのvariation全部コピーしたい

② 制限や正則化を満たす
→inputのvariation全部無視したい

結果的に、Autoencoderはxの再建に必要な変化（variation）のみを表現する。（represent only the variations that are needed to reconstruct training examples）

つまり、最終的に、Autoencoderは、Manifoldの接平面方向での変化のみに敏感に反応する。

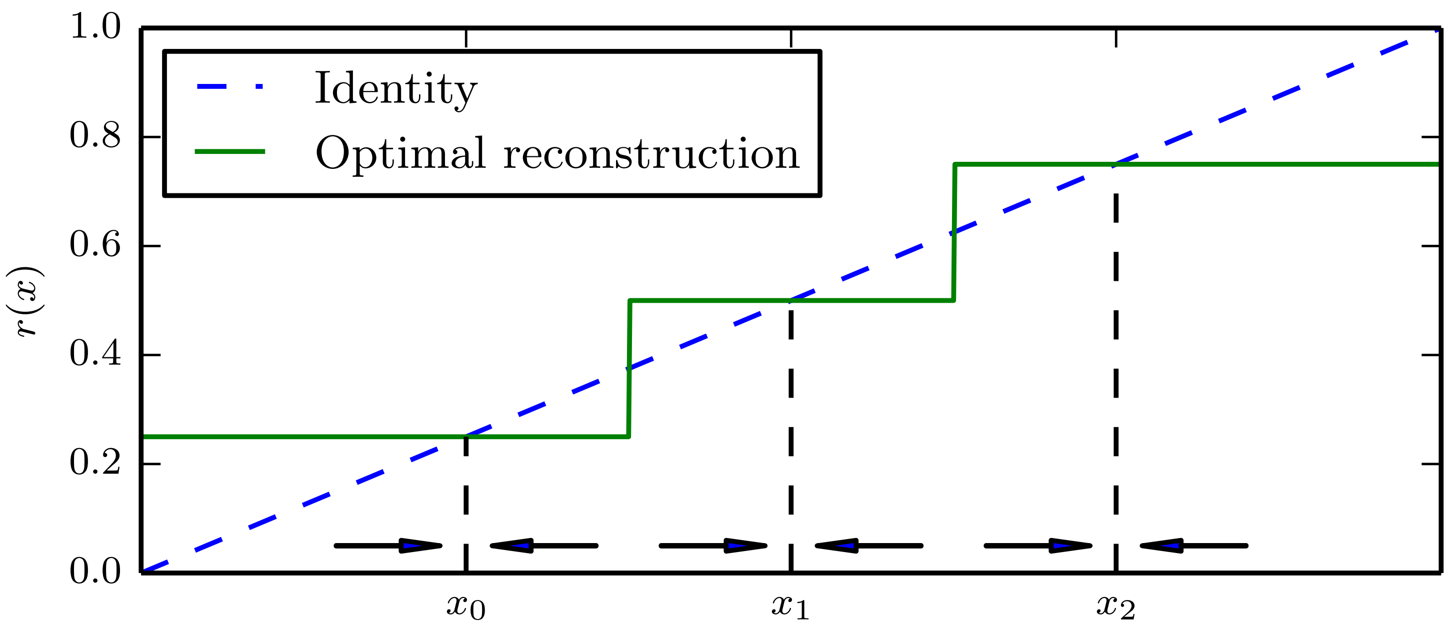
上下移動の1D manifold。
tangent lineになっている。

このtangent方向に移動すると、ピクセルがどう変わる：
灰色（変わらない）、黒（暗くなる）、白（明るくなる）

Learning Manifolds with Autoencoders②

再建（reconstruction）関数を、データポイント周囲の摂動（perturbation）に鈍感にさせて、Autoencoderはmanifold構造をキャプチャーできる

1Dデータの例：0D多様体（一つの点）の集合



青線：同一関数r(x)=x

黒矢印：r(x)-xベクトル

緑線：

・Autoencoderが学習した再建関数

・データポイント付近のr(x)の導関数を小さくする

・Manifold間の導関数は大きい

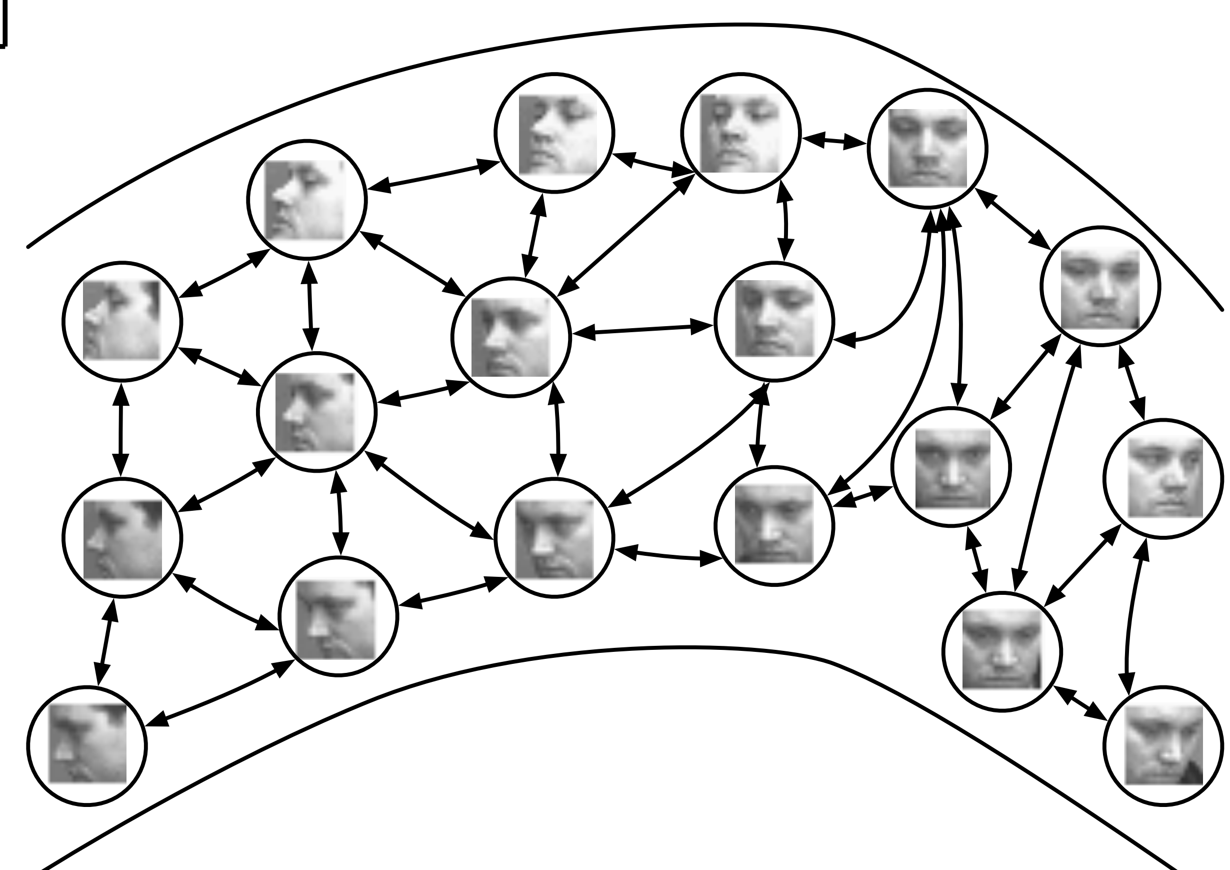
Autoencoder以前のManifold学習手段：最近傍グラフベースのノンパラメトリック方法

・nodeに接平面を付ける

・接平面の方向を、nodeとneighborsの差分の方向と、関連づける

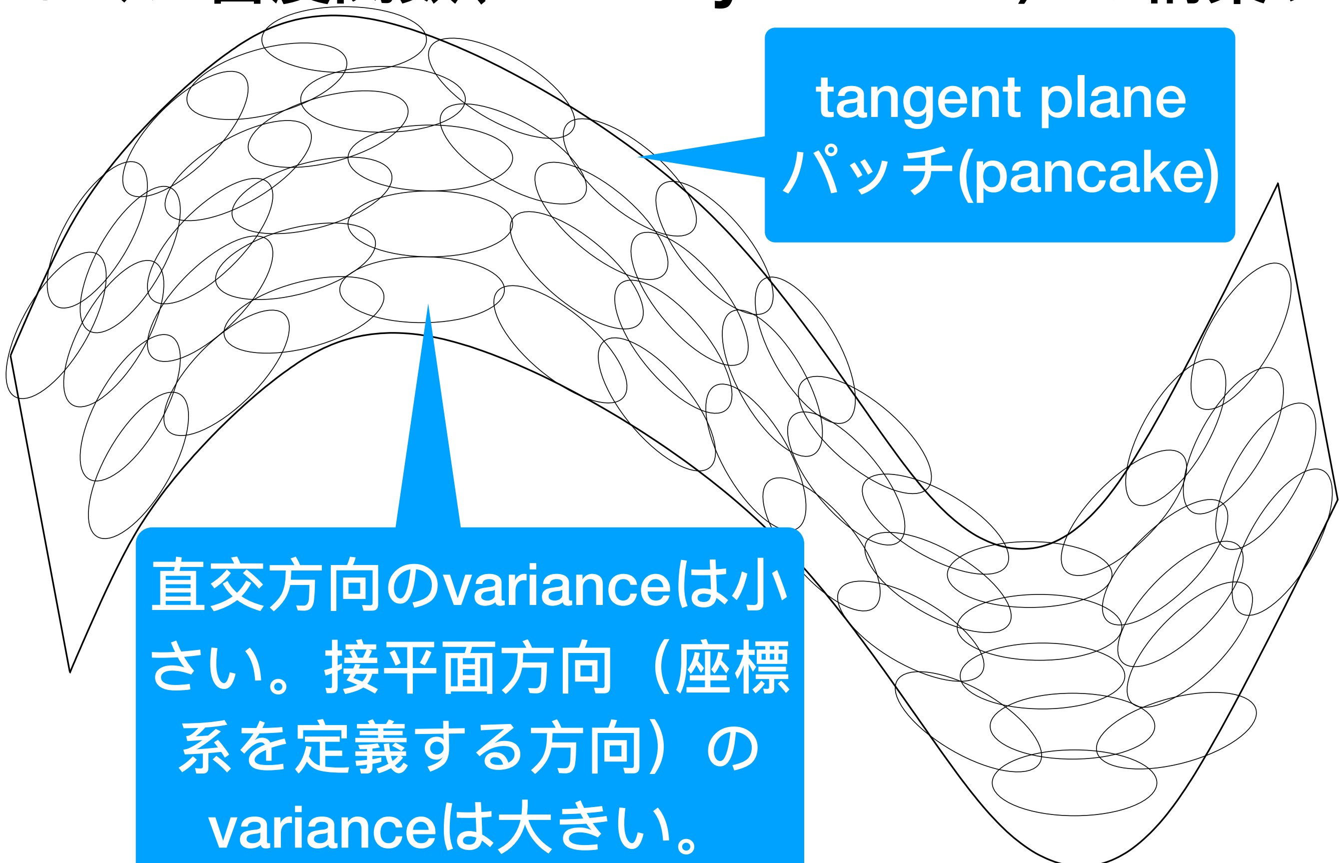
・nodeにグローバル座標系（Embedding）をつける（次ページの図）

・内挿（interpolation）で新しいサンプルに一般化する



Learning Manifolds with Autoencoders②続

（ノンパラメトリック方法）接平面（tangent plane）集合からグローバル座標系（もしくは密度関数、density function）を構築する



tangent plane
パッチ(pancake)

直交方向のvarianceは小さい。接平面方向（座標系を定義する方向）のvarianceは大きい。

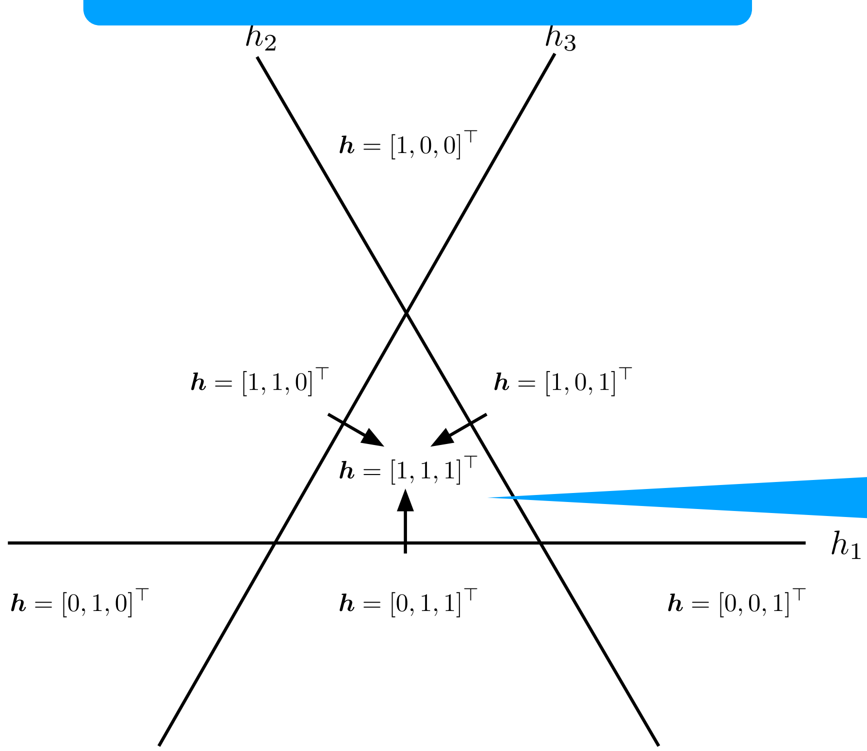
しかし、ノンパラメトリック方法の一番のディメリットは、複雑なManifoldを学習するために、たくさんサンプルが必要

理由：

・ノンパラメトリック方法は、新しいインプットに対して、内挿しかできない

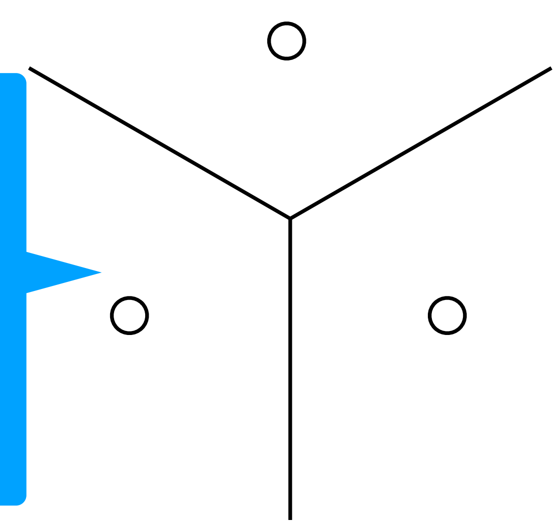
・Manifoldがスムーズじゃない場合、Manifoldの複雑なVariationをキャプチャーするために、たくさんサンプルが必要

この課題が存在するので、分散表現（distributed representation）や深層学習でManifoldをキャプチャーするのが進んでいる。



分散表現スペース：例えば、h=[1,1,1]のサンプルがなくても、[1,1,1] regionをキャプチャーできる

最近傍スペース：全てのregionにサンプルが必要



Contractive（収縮性の） Autoencoders

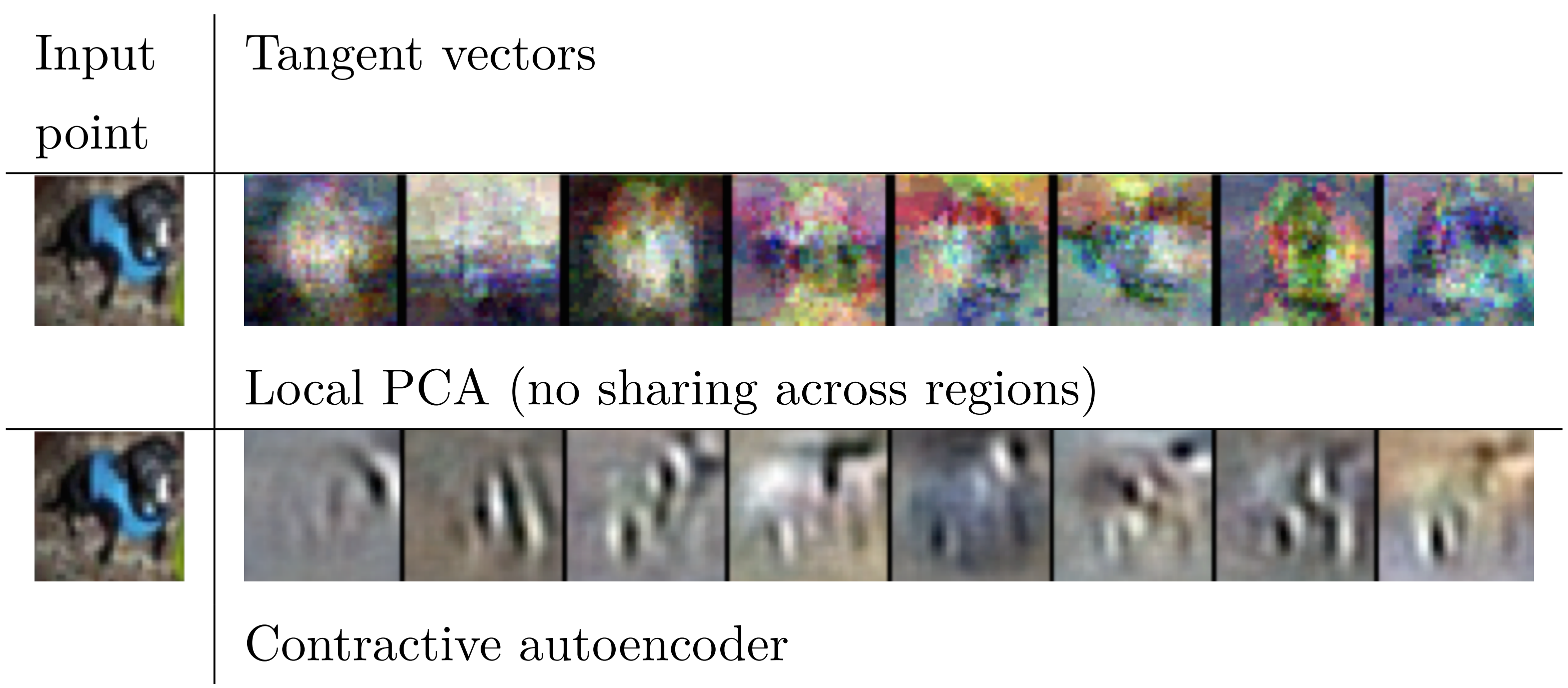
$$\Omega(h) = \lambda \| \frac{\partial f(x)}{\partial x} \|^2$$

Denoising autoencoderは再建関数（reconstruction function）を小さい摂動に抵抗させよう

Contractive autoencoderは特徴抽出関数（feature extraction function）を小さい摂動に抵抗させよう

①収縮というのは、Contractive Autoencoderのトレーニングで、J（ヤコビ行列）の特異値が1より小さくなる

②CAEの目標：Jの大きい特異値に対する方向は、Manifoldの接（tangent）方向になる



CAEが予測した接ベクトル（tangent vector）は、オブジェクト（犬の頭や脚）の意味がある移動や変化を示せる。