Deep Learning - Foundations and Concepts

Chapter 16. Continuous Latent Variables

nonlineark@github

April 17, 2025

Outline

Maximum variance formulation

Problem

Consider a data set of observations $\{x_n\}$ where $n = 1, \ldots, N$, and x_n is a Euclidean variable with dimensionality D. Our goal is to project the data onto a space having dimensionality $M < D$ while maximizing the variance of the projected data.

Maximum variance formulation

Let's calculate the variance of the projected data on a unit direction v:

$$y_n = x_n \cdot v$$

$$E(y_n) = \frac{1}{N}\sum_{n=1}^{N} y_n = E(x_n) \cdot v$$

$$\begin{aligned}\mathrm{var}(y_n) &= \frac{1}{N}\sum_{n=1}^{N}(y_n - E(y_n))^2 = \frac{1}{N}\sum_{n=1}^{N}((x_n - E(x_n)) \cdot v)^2 \\ &= \frac{1}{N}\sum_{n=1}^{N} v^T(x_n - E(x_n))(x_n - E(x_n))^T v = v^T S v\end{aligned}$$

where S is the data covariance matrix defined by:

$$S = \frac{1}{N}\sum_{n=1}^{N}(x_n - E(x_n))(x_n - E(x_n))^T$$

Maximum variance formulation

Let's find the unit direction v_1 for the largest variance. Suppose that $\lambda_1 \geq \cdots \geq \lambda_D$ are the D eigenvalues of S, and their corresponding orthonormal eigenvectors are $u_1, \ldots, u_D$ respectively. Let $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_D)$, $U = \begin{pmatrix} u_1 & \cdots & u_D \end{pmatrix}$. We have:

$$v_1 = U\alpha_1$$
$$v_1^T S v_1 = \alpha_1^T U^T S U \alpha_1 = \alpha_1^T \Lambda \alpha_1 \leq \lambda_1 ||\alpha_1||^2 = \lambda_1$$

The equality holds if and only if v_1 is an eigenvector corresponds to the largest eigenvalue λ_1. Without loss of generality, we could set $v_1 = u_1$.

Maximum variance formulation

Let's find the unit direction v_2 for the second largest variance. Because v_2 is orthogonal to v_1 thus u_1, in the coordinate system formed by the orthonormal basis $u_1, \ldots, u_D$, its first coordinate is 0:

$$v_2 = U\alpha_2$$
$$v_2^T S v_2 = \alpha_2^T \Lambda \alpha_2 \leq \lambda_2 ||\alpha_2||^2 = \lambda_2$$

Again, the equality holds if and only if v_2 is an eigenvector corresponds to the second largest eigenvalue λ_2. Without loss of generality, we could set $v_2 = u_2$.

Maximum variance formulation

If we consider the general case of an M-dimensional projection space, the optimal linear projection for which the variance of the projected data is maximized is now defined by the M eigenvectors $u_1, \ldots, u_M$ of the data covariance matrix S corresponding to the M largest eigenvalues $\lambda_1, \ldots, \lambda_M$.

Minimum-error formulation

We now discuss an alternative formulation of PCA based on projection error minimization:

- We want to find an orthonormal basis $u_1, \ldots, u_D$, where the M-dimensional linear subspace can be presented by the first M of the basis vectors.
- Each data point x_n is approximated by $\tilde{x}_n = \sum_{i=1}^{M} z_{ni} u_i + \sum_{i=M+1}^{D} b_i u_i$.

such that the squared distance between the original data point x_n and its approximation $\tilde{x}_n$, averaged over the data set:

$$J = \frac{1}{N} \sum_{n=1}^{N} ||x_n - \tilde{x}_n||^2$$

is minimized.

Minimum-error formulation

$$0 = \frac{\partial J}{\partial z_{ni}} = -\frac{2}{N}(x_n^T u_i - z_{ni}) \implies z_{ni} = x_n^T u_i$$

$$0 = \frac{\partial J}{\partial b_i} = -\frac{2}{N}\sum_{n=1}^{N}(x_n^T u_i - b_i) \implies b_i = (E(x_n))^T u_i$$

$$x_n - \tilde{x}_n = \sum_{i=M+1}^{D}((x_n - E(x_n)) \cdot u_i)u_i$$

$$J = \frac{1}{N}\sum_{n=1}^{N}||x_n - \tilde{x}_n||^2 = \sum_{i=M+1}^{D} u_i^T S u_i$$

Minimum-error formulation

We recognize that J is the total variance of the projected data on the unit directions $u_{M+1}, \ldots, u_D$. To minimize J, $u_{M+1}, \ldots, u_D$ should be the eigenvectors corresponding to the smallest $D - M$ eigenvalues of S, and hence the eigenvectors defining the principal subspace are those corresponding to the M largest eigenvalues.

Data compression

One application for PCA is data compression:

$$\tilde{x}_n = \sum_{i=1}^{M}(x_n \cdot u_i)u_i + \sum_{i=M+1}^{D}(E(x_n)\cdot u_i)u_i = E(x_n) + \sum_{i=1}^{M}((x_n - E(x_n))\cdot u_i)u_i$$

This represents a compression of the data set, because for each data point we have replaced the D-dimensional vector x_n with an M-dimensional vector.

Data whitening

Suppose we have a data set of observations $\{x^n\}$ where $n = 1, \ldots, N$, and x^n is a Euclidean variable with dimensionality D. We often want to transform the data set to standardize certain of its properties. For example, making a linear re-scaling of the individual variables such that each variable has zero mean and unit variance:

$$\bar{x}_d = \frac{1}{N} \sum_{n=1}^{N} x_d^n$$

$$\sigma_d^2 = \frac{1}{N} \sum_{n=1}^{N} (x_d^n - \bar{x}_d)^2$$

$$\tilde{x}_d^n = \frac{x_d^n - \bar{x}_d}{\sigma_d}$$

Data whitening

The covariance matrix for the standardized data has components:

$$\rho_{ij} = E(\tilde{x}_i^n \tilde{x}_j^n) - E(\tilde{x}_i^n)E(\tilde{x}_j^n) = \frac{1}{N}\sum_{n=1}^{N} \frac{x_i^n - \bar{x}_i}{\sigma_i} \frac{x_j^n - \bar{x}_j}{\sigma_j}$$

If two components x_i and x_j of the data are perfectly correlated, then $\rho_{ij} = 1$, and if they are uncorrelated, then $\rho_{ij} = 0$.

Data whitening

Using PCA we can make a more substantial normalization of the data to give it zero mean and unit covariance, so that different variables become decorrelated:

$$
\begin{aligned}
y^n &= \Lambda^{-\frac{1}{2}} U^T (x^n - \bar{x}) \\
E(y^n) &= 0 \\
E(y^n (y^n)^T) &= \frac{1}{N} \sum_{n=1}^{N} \Lambda^{-\frac{1}{2}} U^T (x^n - \bar{x})(x^n - \bar{x})^T U \Lambda^{-\frac{1}{2}} \\
&= \Lambda^{-\frac{1}{2}} U^T S U \Lambda^{-\frac{1}{2}} = \Lambda^{-\frac{1}{2}} \Lambda \Lambda^{-\frac{1}{2}} = I \\
\mathrm{cov}(y^n) &= E(y^n (y^n)^T) - E(y^n)(E(y^n))^T = I
\end{aligned}
$$

High-dimensional data

In some applications of PCA, the number of data points is smaller than the dimensionality of the data space. For such cases, we can calculate the eigenvalues and eigenvectors more efficiently this way:

- Let $X = \begin{pmatrix} x_1 - \bar{x} & \cdots & x_N - \bar{x} \end{pmatrix}^T$, then $S = \frac{1}{N} X^T X$.
- Calculate the eigenvalues and eigenvectors of $\frac{1}{N} X X^T$ instead, say $\frac{1}{N} X X^T v = \lambda v$.
- Then λ is an eigenvalue of S and $u = X^T v$ is an eigenvector of S.
  - $\frac{1}{\sqrt{N\lambda}} u$ is the corresponding unit eigenvector (suppose v is already a unit vector).

Probabilistic PCA

PCA can also be expressed as the maximum likelihood solution of a probabilistic latent variable model, known as probabilistic PCA:

- A probabilistic PCA model represents a constrained form of a Gaussian distribution.
- We can derive an EM algorithm for PCA that is computationally efficient.
- The combination of a probabilistic model and EM allows us to deal with missing values in the data set.
- Mixtures of probabilistic PCA models can be formulated in a principled way and trained using the EM algorithm.

Probabilistic PCA

- The existence of a likelihood function allows direct comparison with other probabilistic density models.
- Probabilistic PCA can be used to model class-conditional densities and hence be applied to classification problems.
- A probabilistic PCA model can be run generatively to provide samples from the distribution.
- Probabilistic PCA forms the basis for a Bayesian treatment of PCA.

Generative model

Probabilistic PCA is a simple example of the linear Gaussian framework:

- Introduce an explicit M-dimensional latent variable z corresponding to the principal component subspace:
  - $p(z) = \mathcal{N}(z; 0, I)$.
- The D-dimensional observed variable x is conditioned on the value of the latent variable:
  - $p(x|z) = \mathcal{N}(x; Wz + \mu, \sigma^2 I)$, where $W \in \mathbb{R}^{D \times M}$ and $\mu \in \mathbb{R}^D$.
- The probabilistic PCA model is an example of a naive Bayes model, as $p(x|z) = \prod_{d=1}^{D} \mathcal{N}(x_d; W_d z + \mu_d, \sigma^2)$, where W_d is the dth row of W.
- We can view the probabilistic PCA model from a generative viewpoint: $x = Wz + \mu + \epsilon$, where $z \sim \mathcal{N}(z; 0, I)$ is an M-dimensional Gaussian latent variable, and $\epsilon \sim \mathcal{N}(\epsilon; 0, \sigma^2 I)$ is a D-dimensional Gaussian noise.

Likelihood function

The marginal distribution is again Gaussian, and is given by:

$$p(x) = \mathcal{N}(x; \mu, WW^T + \sigma^2 I) = \mathcal{N}(x; \mu, C)$$

The predictive distribution $p(x)$ is governed by the parameters μ, W and σ^2. However, there is redundancy in this parameterization corresponding to rotations of the latent space coordinates.

Maximum likelihood

Given a data set $\{x_n\}$, the log likelihood function is given by:

$$L = \sum_{n=1}^{N} \log p(x_n)$$

$$= -\frac{ND}{2} \log 2\pi - \frac{N}{2} \log \det C - \frac{1}{2} \sum_{n=1}^{N} (x_n - \mu)^T C^{-1} (x_n - \mu)$$

It's easy to maximize L with respect to μ:

$$0 = \frac{\partial L}{\partial \mu} = \sum_{n=1}^{N} (x_n - \mu)^T C^{-1} \implies \mu_{ML} = \frac{1}{N} \sum_{n=1}^{N} x_n = \bar{x}$$

Maximum likelihood

Maximization with respect to W and σ^2 is more complex but nonetheless has an exact closed-form solution:

$$\sigma^2_{ML} = \frac{1}{D-M} \sum_{i=M+1}^{D} \lambda_i$$

$$W_{ML} = U_M (\Lambda_M - \sigma^2_{ML} I)^{\frac{1}{2}} R$$

where:

- $\lambda_1 \geq \cdots \geq \lambda_D$ are the eigenvalues of S.
- $u_1, \ldots, u_D$ are the corresponding eigenvectors of S.
- $\Lambda_M = \mathrm{diag}(\lambda_1, \ldots, \lambda_M)$.
- $U_M = \begin{pmatrix} u_1 & \cdots & u_M \end{pmatrix}$.
- R is an arbitrary $M \times M$ orthogonal matrix.

Maximum likelihood

It is worth taking a moment to study the form of the covariance matrix C:

$$C = W_{ML}W_{ML}^T + \sigma_{ML}^2 I = U_M(\Lambda_M - \sigma_{ML}^2 I)U_M^T + \sigma_{ML}^2 I$$

Since the variance of the predictive distribution along some direction specified by the unit vector v is given by $v^T C v$, the variance along the direction of the eigenvectors is given by:

$$u_i^T C u_i = \begin{cases} (\lambda_i - \sigma_{ML}^2) + \sigma_{ML}^2 = \lambda_i & \text{if } 1 \leq i \leq M \\ \sigma_{ML}^2 & \text{if } M+1 \leq i \leq D \end{cases}$$

In other words, this model correctly captures the variance of the data along the principal axes and approximates the variance in all remaining directions with a single average value σ_{ML}^2.

Maximum likelihood

The relation between probabilistic PCA and conventional PCA. The posterior distribution in latent space $p(z|x)$ is also a Gaussian:

$$p(z|x) = \mathcal{N}(z; M^{-1}W_{ML}^T(x - \mu_{ML}), \sigma_{ML}^2 M^{-1})$$
$$M = W_{ML}^T W_{ML} + \sigma_{ML}^2 I$$

If we take the limit $\sigma_{ML}^2 \to 0$, then the posterior mean reduces to:

$$(W_{ML}^T W_{ML})^{-1} W_{ML}^T (x - \bar{x}) = R^T \Lambda_M^{-\frac{1}{2}} U_M^T (x - \bar{x})$$

which represents an orthogonal projection of the data point onto the latent space, and so we recover the standard PCA model.

Maximum likelihood

The probabilistic PCA model defines a multivariate Gaussian distribution in which the number of independent parameters can be controlled while still allowing the model to capture the dominant correlations in the data:

- The number of parameters scales quadratically with D for the general Gaussian distribution.
- If we restrict the covariance matrix to be diagonal, then it can no longer express any correlations between them.
- For the probabilistic PCA model, the number of degrees of freedom in the covariance matrix C is given by: $DM + 1 - \frac{M(M-1)}{2}$:
  - The number of independent parameters in this model therefore only grows linearly with D, for fixed M.
  - If we take $M = D - 1$, then we recover the standard result for a full covariance Gaussian.
  - If $M = 0$, the model is equivalent to the isotropic covariance case.

Factor analysis

Factor analysis is a linear Gaussian latent variable model:

$$p(z) = \mathcal{N}(z; 0, I)$$
$$p(x|z) = \mathcal{N}(x; Wz + \mu, \Psi)$$

where $\Psi = \text{diag}(\psi_1, \ldots, \psi_D)$ is a $D \times D$ diagonal matrix.

Factor analysis

- Factor analysis is an example of a naive Bayes model, as $p(x|z) = \prod_{d=1}^{D} \mathcal{N}(x_d; W_d z + \mu_d, \psi_d)$, where W_d is the dth row of W.
- As with probabilistic PCA, factor analysis is invariant to rotations in the latent space.
- Another difference between probabilistic PCA and factor analysis is their behavior under transformations of the data set:
  - Probabilistic PCA is covariant under a rotation of the axes of the data space, since $R\sigma^2 I R^T = \sigma^2 I$.
  - Factory analysis is covariant under component-wise re-scaling of the data variables, since $\Phi\Psi\Phi^T$ is still diagonal when Φ is diagonal.

Independent component analysis

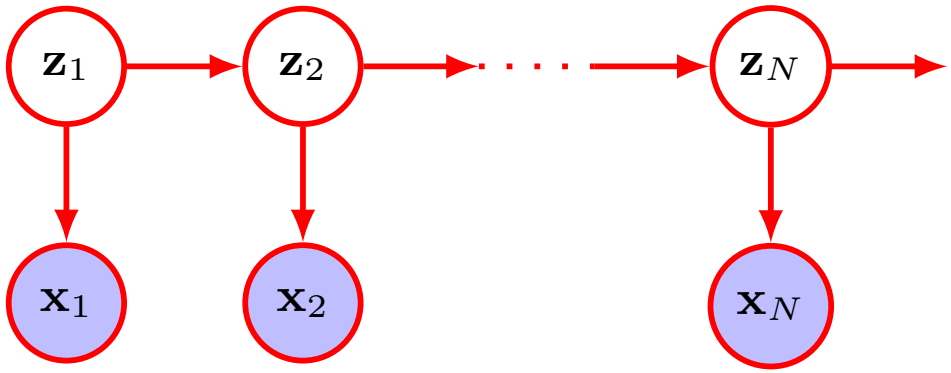
In independent component analysis, the data variables are assumed to be linear mixtures of the latent variables, and the latent variables are assumed non-Gaussian and mutually independent:

$$p(z) = \prod_{m=1}^{M} p(z_m)$$

$$x = Az$$

Kalman filters

Figure: A linear dynamical system, or Kalman filter



Evidence lower bound

Consider a model $p(x, z; w)$ with an observed variable x, a latent variable z, and a learnable parameter vector w. If we introduce an arbitrary distribution $q(z)$ over the latent variable then we can write the log likelihood function $\log p(x; w)$ as:

$$\begin{aligned}\log p(x; w) &= \int q(z) \log \frac{p(x, z; w)}{q(z)} \mathrm{d}z - \int q(z) \log \frac{p(z|x; w)}{q(z)} \mathrm{d}z \\ &= \mathcal{L}(q, w) + \mathrm{KL}(q(z)||p(z|x; w)) \\ &\geq \mathcal{L}(q, w)\end{aligned}$$

we see that $\mathcal{L}(q, w)$ forms a lower bound on the log likelihood, known as the evidence lower bound or ELBO.

Evidence lower bound

We can maximize the log likelihood function using a two-stage iterative procedure called the expectation maximization algorithm, or EM algorithm:

- We first initialize the parameters $w^{(\text{old})}$.
- In the E step we keep $w^{(\text{old})}$ fixed and we maximize the lower bound with respect to $q(z)$:
  - This is achieved when $q(z) = p(z|x; w^{(\text{old})})$ for which the Kullback-Leibler divergence is zero.
- In the M step we keep this choice of $q(z)$ fixed and maximize $\mathcal{L}(q, w)$ with respect to w:
  - This is equal to maximizing $\mathcal{Q}(w, w^{(\text{old})}) = \int p(z|x; w^{(\text{old})}) \log p(x, z; w) \mathrm{d}z$.

Evidence lower bound

For the particular case of i.i.d. data set, we have:

$$\mathcal{L}(q, w) = \sum_{n=1}^{N} \int q(z_n) \log \frac{p(x_n, z_n; w)}{q(z_n)} \mathrm{d}z_n$$

$$\mathcal{Q}(w, w^{(\mathrm{old})}) = \sum_{n=1}^{N} \int p(z_n | x_n; w^{(\mathrm{old})}) \log p(x_n, z_n; w) \mathrm{d}z_n$$

Expectation maximization

We can now use the EM algorithm to learn the parameters of the probabilitic PCA model:

$$
\begin{aligned}
p(z_n) &= \mathcal{N}(z_n; 0, I) = \frac{1}{\sqrt{2\pi}^K} \exp(-\frac{1}{2}||z_n||^2) \\
p(x_n|z_n; W, \mu, \sigma^2) &= \mathcal{N}(x_n; W z_n + \mu, \sigma^2 I) \\
&= \frac{1}{\sqrt{2\pi\sigma^2}^D} \exp(-\frac{1}{2\sigma^2}||x_n - W z_n - \mu||^2)
\end{aligned}
$$

Expectation maximization

$$
\begin{aligned}
\mathcal{Q} &= \sum_{n=1}^{N} E(\log p(z_n) \log p(x_n|z_n; W, \mu, \sigma^2)) \\
&= -\frac{NK}{2} \log 2\pi - \frac{1}{2} \sum_{n=1}^{N} E(||z_n||^2) - \frac{ND}{2} \log 2\pi\sigma^2 \\
&\quad - \frac{1}{2\sigma^2} \sum_{n=1}^{N} E(||x_n - W z_n - \mu||^2) \\
&= -\frac{NK}{2} \log 2\pi - \frac{1}{2} \sum_{n=1}^{N} \mathrm{tr}(E(z_n z_n^T)) - \frac{ND}{2} \log 2\pi\sigma^2 \\
&\quad - \frac{1}{2\sigma^2} \sum_{n=1}^{N} (||x_n - \mu||^2 - 2(x_n - \mu)^T W E(z_n) + \mathrm{tr}(W^T W E(z_n z_n^T)))
\end{aligned}
$$

Expectation maximization

We already know that the exact maximum likelihood solution for μ is given by the sample mean $\bar{x}$. For the E step:

$$
\begin{aligned}
p(z_n|x_n) &= \mathcal{N}(z_n; M^{-1}W_{\text{old}}^T(x_n - \bar{x}), \sigma_{\text{old}}^2 M^{-1}) \\
M &= W_{\text{old}}^T W_{\text{old}} + \sigma_{\text{old}}^2 I \\
E(z_n) &= M^{-1}W_{\text{old}}^T(x_n - \bar{x}) \\
E(z_n z_n^T) &= \text{cov}(z_n) + E(z_n)(E(z_n))^T = \sigma_{\text{old}}^2 M^{-1} + E(z_n)(E(z_n))^T
\end{aligned}
$$

Expectation maximization

For the M step:

$$0 = \frac{\partial \mathcal{Q}}{\partial W}(H) = \frac{1}{\sigma^2}\sum_{n=1}^{N} \mathrm{tr}((E(z_n)(x_n - \bar{x})^T - E(z_n z_n^T)W^T)H)$$

$$\implies W = (\sum_{n=1}^{N}(x_n - \bar{x})(E(z_n))^T)(\sum_{n=1}^{N} E(z_n z_n^T))^{-1}$$

$$0 = \frac{\partial \mathcal{Q}}{\partial \sigma} = -\frac{ND}{\sigma} + \frac{1}{\sigma^3}\sum_{n=1}^{N}(||x_n - \bar{x}||^2 - 2(x_n - \bar{x})^T W E(z_n) + \mathrm{tr}(W^T W E(z_n z_n^T)))$$

$$\implies \sigma^2 = \frac{1}{ND}\sum_{n=1}^{N}(||x_n - \bar{x}||^2 - 2(x_n - \bar{x})^T W E(z_n) + \mathrm{tr}(W^T W E(z_n z_n^T)))$$

EM for PCA

We can take the limit $\sigma^2 \to 0$, corresponding to standard PCA. Let $X = \begin{pmatrix} x_1 - \bar{x} & \cdots & x_N - \bar{x} \end{pmatrix}^T$ and $\Omega = \begin{pmatrix} E(z_1) & \cdots & E(z_N) \end{pmatrix}$, we have:

$$
\begin{aligned}
M &= W_{\text{old}}^T W_{\text{old}} \\
\Omega &= M^{-1} W_{\text{old}}^T X^T = (W_{\text{old}}^T W_{\text{old}})^{-1} W_{\text{old}}^T X^T \\
W &= X^T \Omega^T (\Omega \Omega^T)^{-1}
\end{aligned}
$$

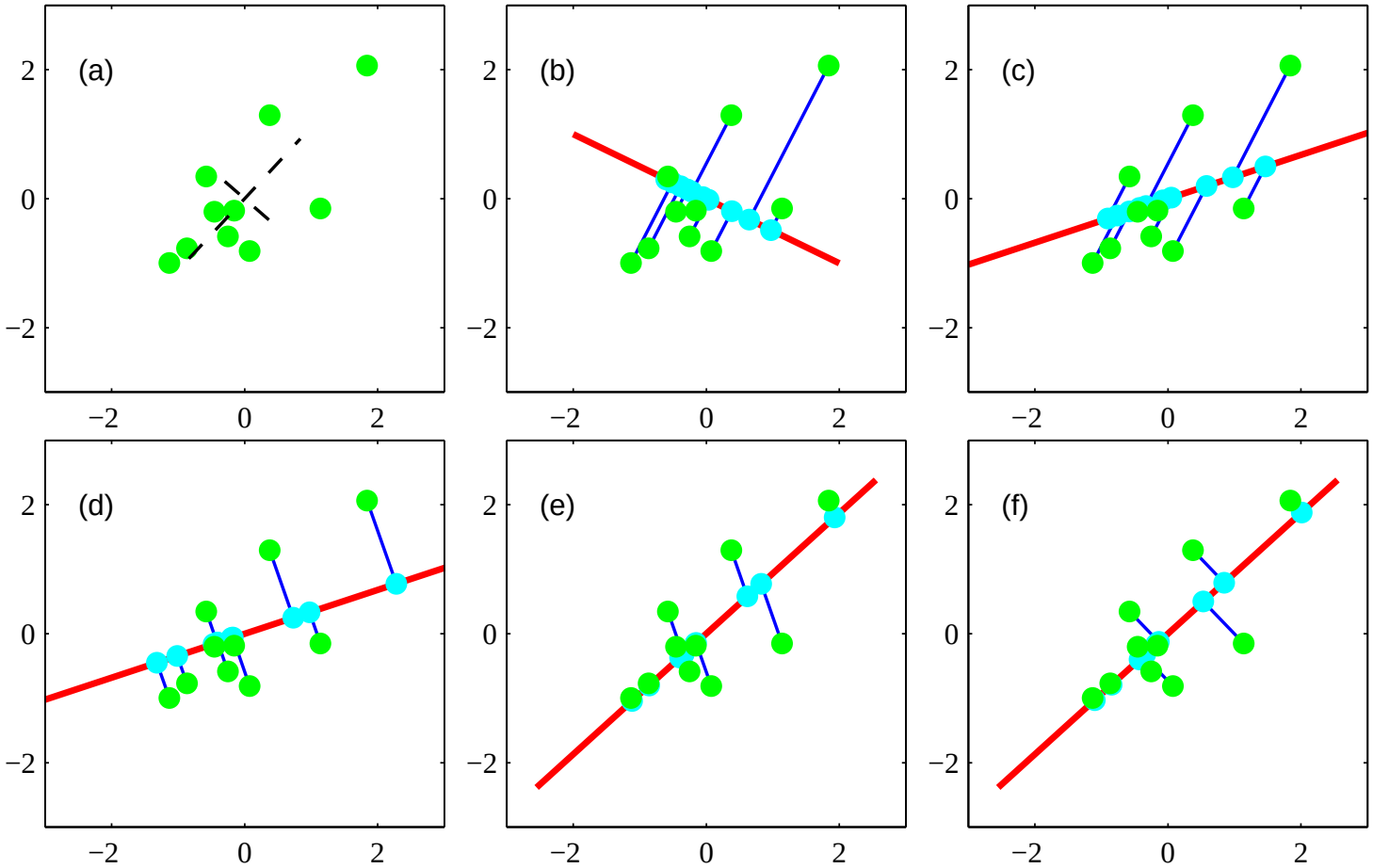
EM for PCA

We see that:

- The E step involves an orthogonal projection of the data points onto the current estimate for the principal subspace.
- The M step represents a re-estimation of the principal subspace to minimize the reconstruction error in which the projections are fixed.

EM for PCA

Figure: Synthetic data illustrating the EM algorithm for PCA



EM for factor analysis

We can also use the EM algorithm to learn the parameters of the factor analysis model:

$$p(z_n) = \mathcal{N}(z_n; 0, I) = \frac{1}{\sqrt{2\pi}^K} \exp(-\frac{1}{2}||z_n||^2)$$

$$p(x_n|z_n; W, \mu, \Psi) = \mathcal{N}(x_n; Wz_n + \mu, \Psi)$$

$$= \frac{1}{\sqrt{2\pi}^D} \frac{1}{\sqrt{\det \Psi}} \exp(-\frac{1}{2}(x_n - Wz_n - \mu)^T \Psi^{-1}(x_n - Wz_n - \mu))$$

EM for factor analysis

$$\mathcal{Q} = \sum_{n=1}^{N} E(\log p(z_n) \log p(x_n|z_n; W, \mu, \Psi))$$

$$= -\frac{N(K+D)}{2} \log 2\pi - \frac{1}{2} \sum_{n=1}^{N} \mathrm{tr}(E(z_n z_n^T)) - \frac{N}{2} \log \det \Psi$$

$$- \frac{1}{2} \sum_{n=1}^{N} (x_n - \mu)^T \Psi^{-1} (x_n - \mu) + \sum_{n=1}^{N} (x_n - \mu)^T \Psi^{-1} W E(z_n)$$

$$- \frac{1}{2} \sum_{n=1}^{N} \mathrm{tr}(W^T \Psi^{-1} W E(z_n z_n^T))$$

EM for factor analysis

We already know that the exact maximum likelihood solution for μ is give by the sample mean $\bar{x}$. For the E step:

$$
\begin{aligned}
p(z_n|x_n) &= \mathcal{N}(z_n; M^{-1}(W_{\text{old}}^T \Psi_{\text{old}}^{-1}(x_n - \bar{x})), M^{-1}) \\
M &= W_{\text{old}}^T \Psi_{\text{old}}^{-1} W_{\text{old}} + I \\
E(z_n) &= M^{-1} W_{\text{old}}^T \Psi_{\text{old}}^{-1}(x_n - \bar{x}) \\
E(z_n z_n^T) &= \text{cov}(z_n) + E(z_n)(E(z_n))^T = M^{-1} + E(z_n)(E(z_n))^T
\end{aligned}
$$

EM for factor analysis

For the M step:

$$0 = \frac{\partial \mathcal{Q}}{\partial W} H = \sum_{n=1}^{N} \mathrm{tr}((E(z_n)(x_n - \bar{x})^T - E(z_n z_n^T) W^T) \Psi^{-1} H)$$

$$\implies W = (\sum_{n=1}^{N} (x_n - \bar{x})(E(z_n))^T)(\sum_{n=1}^{N} E(z_n z_n^T))^{-1}$$

$$0 = \frac{\partial \mathcal{Q}}{\partial \Lambda} H = \frac{N}{2} \mathrm{tr}(\Psi H) - \frac{N}{2} \mathrm{tr}(S H) + \sum_{n=1}^{N} \mathrm{tr}(W E(z_n)(x_n - \bar{x})^T H)$$

$$- \frac{1}{2} \sum_{n=1}^{N} \mathrm{tr}(W E(z_n z_n^T) W^T H)$$

$$\implies \Psi = \mathrm{diag}(S - W \frac{1}{N} \sum_{n=1}^{N} E(z_n)(x_n - \bar{x})^T)$$

Nonlinear latent variable models

We can use the flexibility of deep neural networks to represent more complex transformations from the latent space to the data space:

$$p_z(z) = \mathcal{N}(z; 0, I)$$
$$x = g(z; w)$$

where w represents the weights and biases.

Nonlinear latent variable models

For learning:

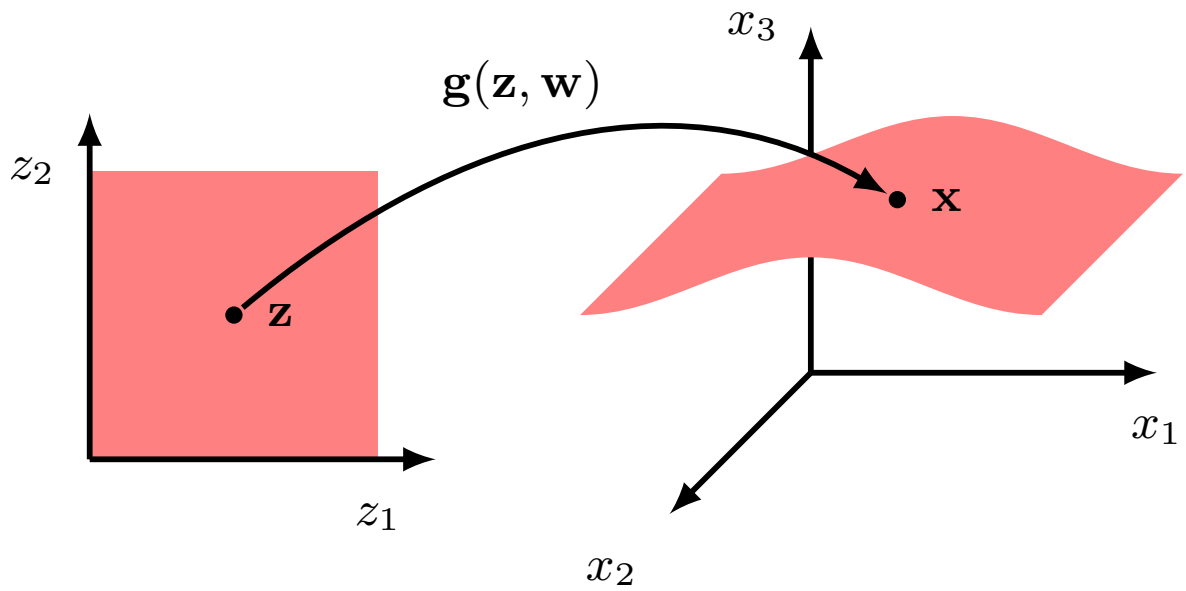
- The likelihood function is given by: $p_x(x) = p_z(z(x))|\det J(x)|$, where $J(x) = \frac{\partial(z_1,\ldots,z_D)}{\partial(x_1,\ldots,x_D)}$ is the Jacobian matrix.
- We need the inverse $z = g^{-1}(x; w)$ of the neural network function, but for most neural networks this inverse will not be well defined.

For sampling:

- Sampling from such a model is straightforward because we can generate samples from $p_z(z)$ and then transform each of them using the neural network function to give corresponding samples of x.

Nonlinear manifolds

Consider the situation in which z has dimensionality M and x has dimensionality D, where $M < D$. In this case the distribution over x is confined to a manifold.



Nonlinear manifolds

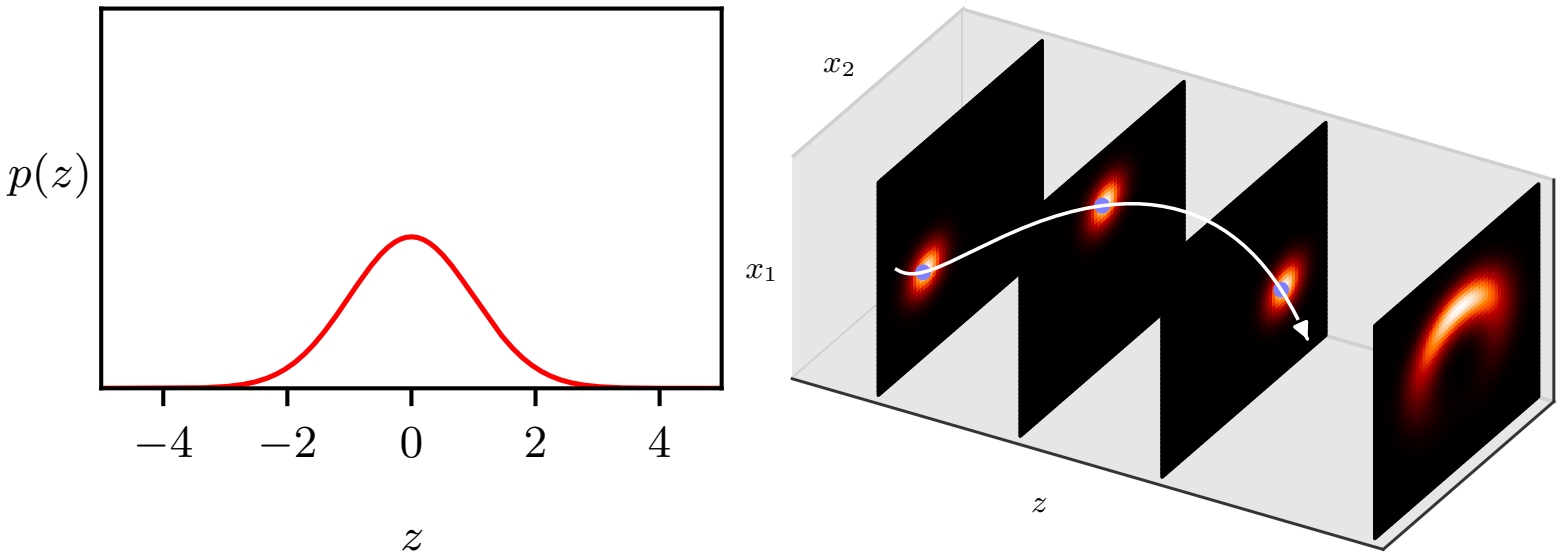
However, this framework assigns zero probability density to any data vector that does not lie exactly on the manifold, which is a problem for gradient-based learning. To address this, we define a conditional distribution across the entire data space, whose parameters are given by the output of the neural network:

$$p(z) = \mathcal{N}(z; 0, I)$$
$$p(x|z; w) = \mathcal{N}(x; g(z; w), \sigma^2 I)$$

This is assuming that x is a vector of continuous variables, and $g(z; w)$ has linear output-unit functions.

Nonlinear manifolds

Figure: Illustration of a nonlinear latent-variable model for a one-dimensional latent space and a two-dimensional data space



Likelihood function

Now suppose we wish to fit the model to an observed data set by maximizng the likelihood function. One approach for evaluating the likelihood function would be to draw samples from the latent space distribution:

$$p(x; w) = \int p(x|z; w)p(z)\mathrm{d}z = \frac{1}{K}\sum_{k=1}^{K} p(x|z_k; w)$$

where $z_k \sim p(z)$. However, the value of K needed for effective training will typically be far too high to be practical.

Discrete data

If the observed data set comprises independent binary variables then we can use a conditional distribution of the form:

$$p(x|z;w) = \prod_{d=1}^{D} g_d(z;w)^{x_d}(1 - g_d(z;w))^{1-x_d}$$

$$g_d(z;w) = \sigma(a_d(z;w))$$

Discrete data

For one-hot encoded categorical variables, we can use a multinomial distribution:

$$p(x|z;w) = \prod_{d=1}^{D} g_d(z;w)^{x_d}$$

$$g_d(z;w) = \frac{\exp(a_d(z;w))}{\sum_{d'=1}^{D} \exp(a_{d'}(z;w))}$$

Four approaches to generative modelling

- Nonlinear latent-variable models based on deep neural networks offer a highly flexible framework for building generative models.
- However, we have also identified some challenges associated with training such models that force us to develop more sophisticated techniques than those needed for linear models.

Four approaches to generative modelling

- Generative adversarial networks (GANs):
  - Relax the requirement for the network mapping to be invertible.
  - Abandon the concept of a likelihood function and instead introduce a second neural network whose function is to provide a training signal for the generative network.
- Variational autoencoders (VAEs):
  - Use a second neural network whose role is to approximate the posterior distribution over the latent variables.

Four approaches to generative modelling

- Normalizing flows:
  - The dimensionality of the latent space is set to be equal to that of the data space and the generative neural network is modified so that it becomes invertible.
  - The likelihood function can be evaluated without approximation.
- Diffusion models:
  - Use a network that learns to transform a sample from the prior distribution into a sample from the data distribution through a sequence of denoising steps.