Foundations and Trends® in
sample
Vol. xx, No xx (xxxx) 1–23
© xxxx xxxxxxxx
DOI: xxxxxx

Probabilistic Dimensionality Reduction

Neil D. Lawrence[1]

[1] *385a Glossop RoadSheffieldS10, U.K., N.Lawrence@dcs.shef.ac.uk*

Abstract

abstract

Contents

Notation and Symbols

General Comments

Any background on notational choice here.

0.0.1 Reading Notation

The use of the design matrix convention means that the sample covariance matrix is given as $\mathbf{S} = n^{-1}\hat{\mathbf{y}}^\top\hat{\mathbf{y}}$.

$$\text{cov}\left(\mathbf{Y}\right) = \frac{1}{n}\sum_{i=1}^{n}\hat{\mathbf{y}}_{i,:}\hat{\mathbf{y}}_{i,:}^\top = n^{-1}\hat{\mathbf{y}}^\top\hat{\mathbf{y}}$$

whilst the centered inner product matrix is given by $\mathbf{K} = \hat{\mathbf{y}}\hat{\mathbf{y}}^\top$

$$\mathbf{K} = \left(k_{i,j}\right)_{i,j}, \qquad k_{i,j} = \hat{\mathbf{y}}_{i,:}^\top\hat{\mathbf{y}}_{j,:}$$

General

q	dimension of latent/embedded space
p	dimension of data space
n	number of data points
$\mathbf{y} = [\mathbf{y}_{1,:},\ldots,\mathbf{y}_{n,:}]^\top = [\mathbf{y}_{:,1},\ldots,\mathbf{y}_{:,p}] \in \Re^{n\times p}$	data matrix
$\hat{\mathbf{y}} = [\hat{\mathbf{y}}_{1,:},\ldots,\hat{\mathbf{y}}_{n,:}]^\top = [\hat{\mathbf{y}}_{:,1},\ldots,\hat{\mathbf{y}}_{:,p}] \in \Re^{n\times p}$	*centered* data matrix
$\mathbf{X} = [\mathbf{x}_{1,:},\ldots,\mathbf{x}_{n,:}]^\top = [\mathbf{x}_{:,1},\ldots,\mathbf{x}_{:,q}] \in \Re^{n\times q}$	latent variables
$\mathbf{W} \in \Re^{p\times q}$	mapping matrix
$\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^\mathrm{T} \in \Re^{n\times n}$	centering matrix
σ	standard deviation of a data set
ℓ	length scale of a kernel matrix

Vectors, Matrices and Norms

$\mathbf{1}$	vector with all entries equal to one
$\mathbf{0}$	vector with all entries equal to zero
$\mathbf{I}$	identity matrix
$\mathbf{A}^\top$	transposed matrix (or vector)
$\mathbf{A}^{-1}$	inverse matrix (in some cases, pseudo-inverse)
$\mathrm{tr}\left(A\right)$	trace of a matrix
$\lvert A\rvert$	determinant of a matrix
$\lvert\cdot\rvert_2$	2-norm, $\lvert\mathbf{x}\rvert_2 := \sqrt{\mathbf{x}^\top\mathbf{x}}$
$\mathbf{a}_{i,:}$	a column vector from the ith row of a given matrix $\mathbf{A}$
$\mathbf{a}_{:,j}$	a column vector from the jth column of a given matrix $\mathbf{A}$

Probability

$P(C)$	probability of an event C
$p(x)$	density evaluated at x
$p(x\vert y)$	density evaluated at x conditioned on y
$q(x)$	approximating distribution (often variational)
$\langle\cdot\rangle$	expectation of a random variable
$\langle\cdot\rangle_{p(\cdot)}$	expectation of a random variable under the density $p(\cdot)$
$\mathcal{N}\left(\boldsymbol{\mu},\Sigma\right)$	multivariate normal distribution with mean vector $\boldsymbol{\mu}$ and

1

Nonlinear Probabilistic Dimensionality Reduction

We have shown how linear dimensionality reduction, with a probabilistic interpretation, can be achieved through marginalization of the latent space, $\mathbf{X}$. Nonlinear dimensionality reduction involves assuming a nonlinear relationship between the latent variables and each observed data dimension. So we have

$$y_{i,j} = f_j(\mathbf{x}_{i,:}) + \epsilon_{i,j}$$

where $\epsilon_{i,j}$ is some corrupting noise, typically independently and identically distributed. The linear special case of assuming

$$f_j(\mathbf{x}) = \mathbf{w}_{j,:}^{\top}\mathbf{x}_{i,:}$$

that we discussed in chapter **??** combined with a Gaussian assumption for the noise variables,

$$\epsilon_{i,j} \sim \mathcal{N}\left(0, \sigma^2\right)$$

implies a Gaussian likelihood of the form

$$p(\mathbf{y}_{i,:}|\mathbf{W}, \mathbf{x}_{i,:}, \sigma^2) = \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:}, \sigma^2\mathbf{I}\right)$$

that allows the nuisance variables, $\mathbf{X} = \{\mathbf{x}_{i,:}\}_{i=1}^{n}$ to be marginalized by allocating a Gaussian for them (Section **??**). However, this constrained our data to be related to our latent variables in a linear way. This constraint is very strong, and is clearly often violated in practice. In chapter **??** we discussed the use of mixtures of Gaussian densities for modelling data. The motivation was that the center of each Gaussian density represented a "prototype" data point that was then corrupted through various transformations. The corruption of the points was modeled by Gaussian densities. Here we would like to consider a more realistic corruption of a prototype.

Let's consider an image of the handwritten digit 6 (image br1561_6.3.pgm), taken from the ***USPS ??*** data. The image has 64 rows by 57 columns, giving a data point which is living in a $p = 3,648$ dimensional space. Let's consider a simple model for handwritten 6s. We model each pixel independently with a binomial distribution. Each pixel is the result of a single binomial trial.

$$p(\mathbf{y}) = \prod_{j=1}^{p} \pi^{y_j}(1-\pi)^{(1-y_j)},$$

where the probability across pixels are taken to be independent and are governed by the same probability of being on, π. The maximum likelihood solution for this parameter is given by the number of pixels that are on in the data point divided by the total number of pixels:

$$\pi = \frac{849}{3,648}.$$

The probability of recovering the original six in any given sample is then given by[1] ***Add number of ones***

$$p(\mathbf{y} = \text{given image}) = \pi^{849}(1-\pi)^{3,648-849} = 2.67 \times 10^{-860}.$$

This implies that even if we sampled from this model every nanosecond between now and the end of the universe[2] we would still be highly

[1] Note this is not the binomial distribution which gives probabilities for total number of successes in a series of binary trials, here we don't only need a precise number of successes, we also need the successes to occur at the right pixels.

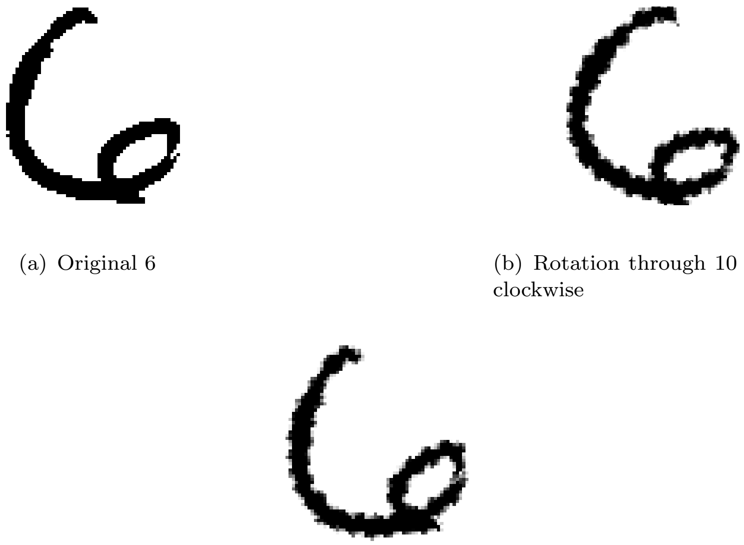
[2] We follow **?** in assuming that will be in approximately *number* years.

unlikely to see the original six. Our expected waiting time would be 10^{217} years and there is only a $fill$ probability of seeing it before the universe ends[3]. This is clearly a very poor generative model of the six. The independence assumption across features (just as in chapter **??**) is a very poor one. Let's consider an alternative model for handwritten sixes. We generate a data set in the following way: we take the prototype 6 and rotate the image 360 times, by one degree for reach rotation. This corruption of the prototype 6 is designed to reflect the sort of corruptions that real prototypes might undergo. A rotated 6 is still a 6, and although it is not realistic to consider such drastic rotations as we have applied, it will be helpful in visualizing the corruption of the data. In figure 1.1 we show the original digit and rotations of 10 degrees clockwise and anticlockwise.

In figure **??** we take all 360 examples of the rotated digits and visualize these data using principal component analysis. and linearly project them down to a two dimensional space (using the posterior given in Section **??**). Selected examples from the data are also placed on the plot near their corresponding projected position. What we see makes intuitive sense. The data, projected into the two dimensional space, proscribes a circle. The data set is inherently one dimensional. The dimension of the data is associated with the rotation transformation. A pure rotation would lead to a pure circle. In practice rotation of images requires some interpolation and this leads to small corruptions of the latent projections away from the circle.

The data are one dimensional in terms of latent representation: they were generated by a rotation of the original images which relied on a singled parameter, the angle of rotation, but because we represented them with a linear low dimensional space we required more dimensions to represent the data. This motivates the need for nonlinear low dimensional representations, before we consider this further though, we first introduce a standard approach to representing nonlinear functions through a *basis*.

[3] This can be computed with the geometric distribution.



(a) Original 6

(b) Rotation through 10 degrees clockwise

(c) Rotation through 10 degrees anticlockwise

Fig. 1.1 The prototype 6 taken from the ***USPS*** data along with two example rotations of the digit. The original image has been converted to 64×64 to match dimensionality of the rotated images.

1.1 Basis Function Representations

A common approach to regression is to specify that a function is given by a linear sum over a fixed basis set,

$$f\left(\mathbf{x}_{i,:};\mathbf{w}\right)=\sum_{k=1}^{M}w_k\phi_k\left(\mathbf{x}_{i,:}\right), \tag{1.1}$$

In this equation there are M basis functions, and the kth basis function is represented by $\phi_k\left(\cdot\right)$ and $\mathbf{w}=\left[w_1,\ldots,w_M\right]^\top$. Basis functions can take several forms, the idea of the basis function is to map the data into a feature space, from which a linear sum over the basis leads to a non linear function. A common *local basis* is the *radial basis function*

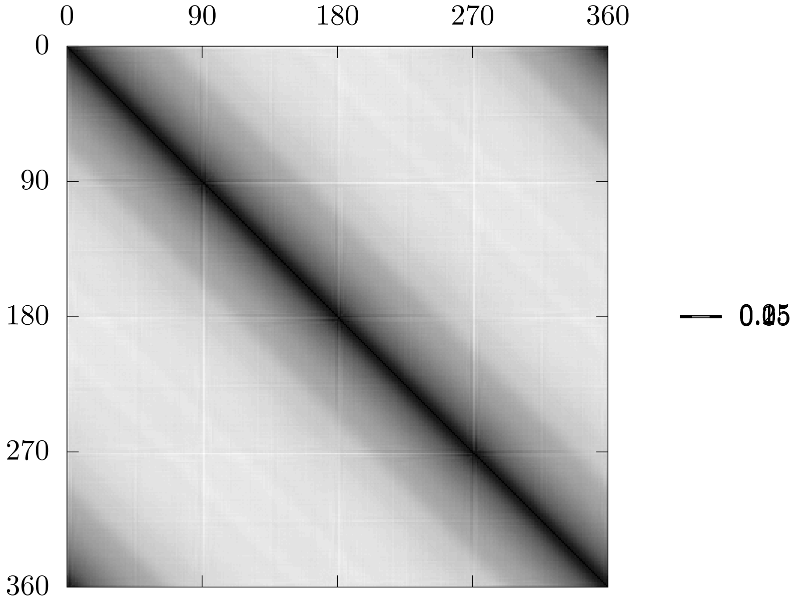


Fig. 1.2 Inter-point squared distances for the rotated digits data. Much of the data structure can be seen in the matrix. All points are ordered by angle of rotation. We can see that the distance between two points with similar angle of rotation is small (note in the upper right and lower left corners the low distances associated with 6s rotated by roughly 360 degrees and unrotated 6s.

where

$$\phi_k\left(\mathbf{x}_i\right) = \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right),$$

where each basis function is centered at $\boldsymbol{\mu}_k$, and has radius of ℓ. A set of such basis functions is visualized in figure 1.3 is the center associated with the kth basis function. Weighting these basis functions using a matrix $\mathbf{W}$,

$$f_j(\mathbf{x}_i) = \sum_{k=1}^{M} w_{j,k}\phi_k(\mathbf{x}_i)$$

provides us with a function. To demonstrate the type of basis functions shown in figure 1.3 we can sample the matrix $\mathbf{W}$ from a Gaussian

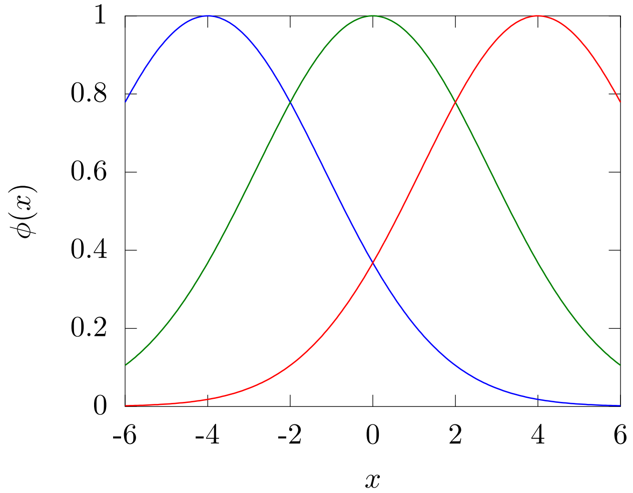


Fig. 1.3 A set of radial basis functions with width $\ell = 2$ and location parameters $\boldsymbol{\mu} = [-4 \;\; 0 \;\; 4]^\top$.

density,

$$w_{j,k} \sim \mathcal{N}\left(0, \alpha\right),$$

to give us the parameters. We can then visualize the resulting functions by plotting them as in figure 1.4

We can make use of these functions to map, for example, to map from a $q = 2$ to a $p = 3$ dimensional space.

By the probabilistic model

can then be specified by adding noise,

$$y\left(\mathbf{x}_i\right) = f\left(\mathbf{x}_i; \mathbf{w}\right) + \epsilon_i,$$

where ϵ_i is the noise associated with the ith data point. If the noise is taken to be Gaussian distributed with variance σ^2,

$$\epsilon_i \sim \mathcal{N}\left(0, \sigma^2\right),$$

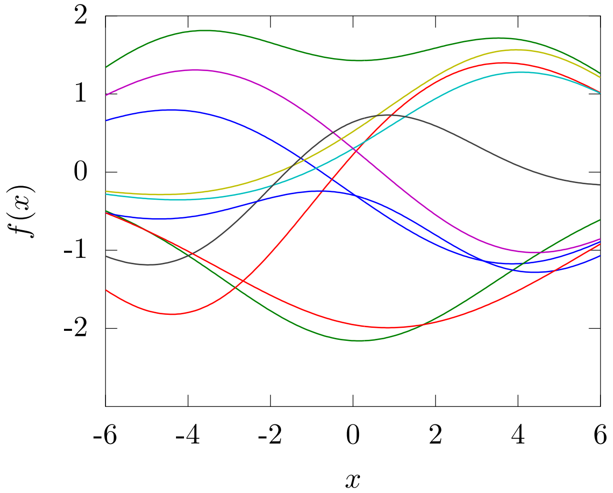


Fig. 1.4 Functions sampled using the basis set from figure 1.3. Each line is a separate sample, generated by a weighted sum of the basis set. The weights, $\mathbf{w}$ are sampled from a Gaussian density with variance $\alpha = 1$.

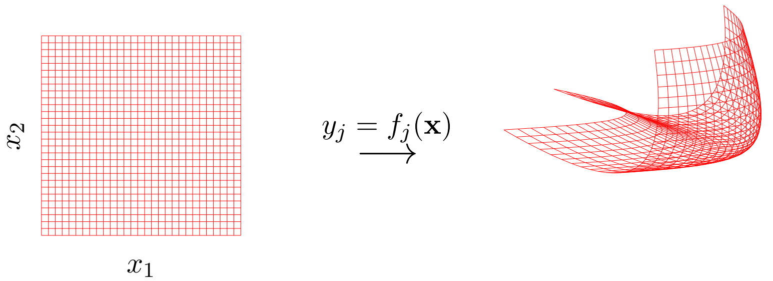


Fig. 1.5 A three dimensional manifold formed by mapping from a two dimensional space to a three dimensional space.

then we can write down the following probabilistic representation for our data,

$$p\left(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2\right) = \prod_{i=1}^{M} \mathcal{N}\left(y_i|w_i, \sigma^2\right),$$

where the mean of the Gaussian distributions in the product is given by the evaluations of our function at the training points, $w_i = f\left(\mathbf{x}_i; \mathbf{w}\right)$.

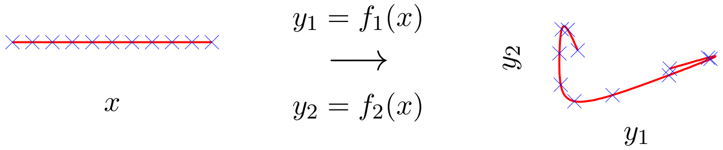


Fig. 1.6 A string in two dimensions, formed by mapping from one dimension, x, line to a two dimensional space, $[y_1,\ y_2]$ using nonlinear functions $f_1(\cdot)$ and $f_2(\cdot)$.

This looks very similar to the representation we used in Section **??**. However, there is one important difference. We have made use of a parameter vector in the representation above. To determine the parameters, we might want to use Bayesian inference, for computational convenience placing a zero mean Gaussian prior over $\mathbf{w}$ with covariance matrix $\gamma'\mathbf{I}$,

$$p\left(\mathbf{w}\right) = \mathcal{N}\left(\mathbf{w}|\mathbf{0}, \gamma'\mathbf{I}\right).$$

By constructing a design matrix from our basis functions, $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_m]$, where $\boldsymbol{\phi}_j = [\phi_j\left(\mathbf{x}_1\right), \ldots, \phi_j\left(\mathbf{x}_n\right)]^\top$ is the vector containing the values of the jth basis function at all input data points, we can rewrite (1.1) for the training data in matrix vector form,

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

1.2 Nonlinear Probabilistic Approaches and Normalization

The difficulty for probabilistic approaches to dimensionality reduction is that the generative model the proscribe is difficult to normalize. If we assume that the prior distribution is Gaussian,

$$\mathbf{x}_{i,:} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$$

and that the mapping for each data point is given by

$$y_{i,j} = f_j(\mathbf{x}_{i,:}) + \epsilon_{i,j}$$

with a standard Gaussian noise assumption,

$$\epsilon_{i,j} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2\right)$$

then the likelihood of the data given the latent variables is given by

$$p(\mathbf{y}_{i,:}|\mathbf{x}_{i,:}) = \prod_{j=1}^{p} \mathcal{N}\left(y_{i,j}|f_j(\mathbf{x}_{i,:}), \sigma^2\mathbf{I}\right).$$

The marginalized likelihood of the data then has the form

$$\begin{aligned} p(\mathbf{y}_{i,:}) = &\int p(\mathbf{y}_{i,:}, \mathbf{x}_{i,:})\mathrm{d}\mathbf{x}_{i,:} \\ &\int p(\mathbf{y}_{i,:}|\mathbf{x}_{i,:})p(\mathbf{x}_{i,:})\mathrm{d}\mathbf{x}_{i,:} \\ &\int \prod_{j=1}^{p} \mathcal{N}\left(y_{i,j}|f_j(\mathbf{x}_{i,:}), \sigma^2\right) \mathcal{N}\left(\mathbf{x}_{i,:}|\mathbf{0}, \mathbf{I}\right) \mathrm{d}\mathbf{x}_{i,:}. \end{aligned}$$

The key component of this integral is in the exponent of the joint density. Taking the logarithm we can see that the exponent has the form

$$\log p(\mathbf{y}_{i,:}, \mathbf{x}_{i,:}) = -\frac{1}{2}\sum_{j=1}^{p}(y_{i,j} - f_j(\mathbf{x}_{i,:}))^2 - \frac{1}{2}\mathbf{x}_{i,:}^\top\mathbf{x}_{i,:} + \text{const}$$

where we have introduced a constant term that does not depend on $\mathbf{x}_{i,:}$. For the linear case, where we have $f_j(\mathbf{x}) = \mathbf{w}_{j,:}^\top\mathbf{x}$, we can rewrite this exponent as

$$\log p(\mathbf{y}_{i,:}, \mathbf{x}_{i,:}) = -\frac{1}{2}\mathbf{y}_{i,:}^\top\mathbf{y}_{i,:} + \mathbf{y}_{i,:}^\top\mathbf{W}\mathbf{x}_{i,:} - \frac{1}{2}\mathbf{x}_{i,:}^\top\mathbf{W}^\top\mathbf{W}\mathbf{x}_{i,:} - \frac{1}{2}\mathbf{x}_{i,:}^\top\mathbf{x}_{i,:} + \text{const}$$

which is a quadratic form implying that integrating over $\mathbf{x}_{i,:}$ to find the marginal likelihood $p(\mathbf{y}_{i,:})$ *is* tractable. This is the integral that is performed in probabilistic PCA and factor analysis (see chapter **??**). However, for more general nonlinear functions,[4] $f(\cdot)$, this integral is *not* tractable. The motivation for using general nonlinear functions is that

[4] There are some specific nonlinear functions which keep the integral tractable. For example if $y = \exp(x)$ we can perform the integral and the resulting density over y is known as the *log normal*.

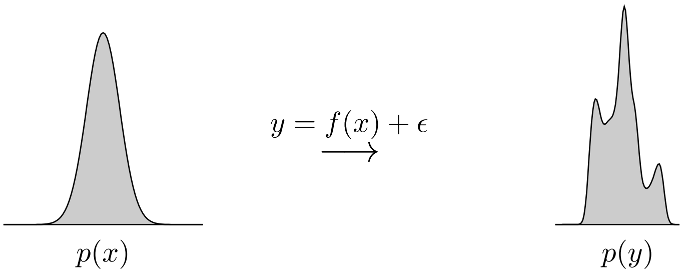


Fig. 1.7 A Gaussian distribution propagated through a non-linear mapping. We define a one dimensional Gaussian distribution in x (left). Then the relationship between y and x is defined as $y_i = f(x_i) + \epsilon_i$ where ϵ is Gaussian random noise with standard deviation $\sigma = 0.2$ and $f(\cdot)$ is computed using an RBF basis with 100 centres uniformly distributed between -4 and 4 and basis function widths $\ell = 0.1$. The new distribution over y (right) is multimodal and difficult to normalize. It is this normalization problem that provides problems for models based on such nonlinear functions.

they do lead to a much richer class of probability densities, for example, even in the special case where we don't perform any dimensionality reduction, for example $p = 1$ and $q = 1$, we can plot the distribution over x and make a numerical estimate of the corresponding density over y. This is done in figure 1.7. Placing the density through the nonlinear function leads to a more complex multimodal density. However, the price we pay for the greater representational power of the model is the inability to express the marginal likelihood for $p(\mathbf{y}_{i,:})$ in a closed form, to fit such models we need to look for approximations.

1.3 Density Networks

Density networks were introduced by **?** as an approximation for fitting such models. Density networks make use of a sample based approximation to the marginal likelihood to represent the data. Making explicit dependence of the mapping function, $f_j(\mathbf{x}_{i,:};\ \boldsymbol{\theta})$, on its parameters, $\boldsymbol{\theta}$, we have

$$p\left(\mathbf{y}|\mathbf{X}\right) = \prod_{i=1}^{n}\prod_{j=1}^{p}\mathcal{N}\left(y_{i,j}|f_j(\mathbf{x}_{i,:};\ \boldsymbol{\theta}), \sigma^2\right)$$

and typically we would chose a Gaussian prior for the latent space (although in principle we could choose any distribution here from which we can sample).

$$p\left(\mathbf{X}\right) = \mathcal{N}\left(\mathbf{x}_{i,:}|\mathbf{0},\mathbf{I}\right)$$

1.3.1 Sample Based Approximation

The sample based approximation involves replacing the continuous integral over the density with a discrete sum of samples from the density. Given a set of m samples, $\{\hat{\mathbf{x}}_{s,:}\}_{s=1}^{m}$ drawn from $p(\mathbf{x}_{i,:})$ we can write down the sample based approximation to the marginal likelihood for the ith data point as

$$\begin{aligned} p(\mathbf{y}_{i,:}) &= \int \prod_{j=1}^{p} p\left(y_{i,j}|\mathbf{x}_{i,:},\boldsymbol{\theta}\right) p(\mathbf{x}_{i,:}) \mathrm{d}\mathbf{x}_{i,:} \\ &\approx \frac{1}{m}\sum_{s=1}^{m}\prod_{j=1}^{p} p\left(y_{i,j}|\hat{\mathbf{x}}_{s,:},\boldsymbol{\theta}\right) \end{aligned}$$

where we have been explicit about including the parameter vector, $\boldsymbol{\theta}$, in the likelihood and the approximation becomes more accurate as our sample size, m, goes towards infinity. The joint likelihood of the entire data set is given by a product over the likelihoods for each data point,

$$p(\mathbf{y}|\boldsymbol{\theta}) = \prod_{i=1}^{n} p(\mathbf{y}_{i,:}|\boldsymbol{\theta})$$

reflecting an assumption that the data is independent and identically distributed *given* the parameters, $\boldsymbol{\theta}$. So to compute the likelihood of the entire data set, $\mathbf{y}$, we need to consider sample based approximations for every data point. The key innovation in density networks is to use the *same set* of samples for each data point. This turns out to significantly reduce the computational demands. Taking the logarithm and substituting with our approximation we have

$$\log p\left(\mathbf{y}|\boldsymbol{\theta}\right) = \sum_{i=1}^{n} \log \frac{1}{m} \sum_{s=1}^{m} p\left(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{s,:},\boldsymbol{\theta}\right).$$

1.3.2 Maximizing the Approximate Likelihood

We can directly maximize the approximate likelihood by taking its gradients with respect to the parameters, $\boldsymbol{\theta}$,

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}\right) = \sum_{s=1}^{m} \frac{p\left(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{s,:}, \boldsymbol{\theta}, \right)}{\sum_{s'=1}^{m} p\left(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{s',:}, \boldsymbol{\theta}\right)} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{s,:}, \boldsymbol{\theta}\right).$$

For convenience we define,

$$r_{i,s} = \frac{p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \hat{\mathbf{x}}_{s,:}\right)}{\sum_{s'=1}^{m} p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \hat{\mathbf{x}}_{s',:}\right)},$$

which, as we will see has multiple interpretations depending on the context. These include the posterior probability of mixture component membership (sometimes referred to as responsibility) and importance sampling weights. This allows us to write the gradient of the log likelihood as a weighted sum of gradients of the likelihood as follows

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}\right) = \sum_{s=1}^{m} r_{i,s} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \hat{\mathbf{x}}_{s,:}\right).$$

1.3.3 Density Network Mappings

The formalism for density networks allows any general $f(\cdot)$ but at the time of publication *multi-layer perceptron* models were particularly popular so it probably seemed natural to consider functions based on these models. These can be seen as basis function models where each basis functions is given by

$$\phi_k(\mathbf{x}_{i,:}; \mathbf{v}_{k,:}) = \frac{1}{1 + \exp(-\mathbf{v}_{k,:}^{\top} \mathbf{x}_{i,:})}.$$

The parameters of the basis functions, $\mathbf{V} = \{\mathbf{v}_{k,:}\}_{k=1}^{M}$, can be optimized along with the matrix $\mathbf{W}$.

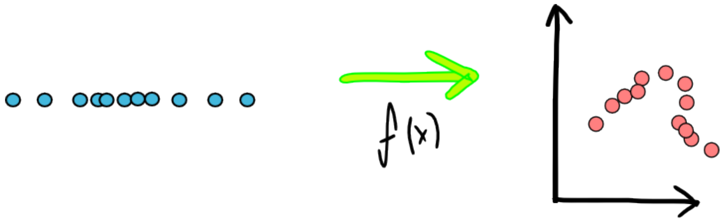


Fig. 1.8 One dimensional Gaussian mapped to two dimensions.

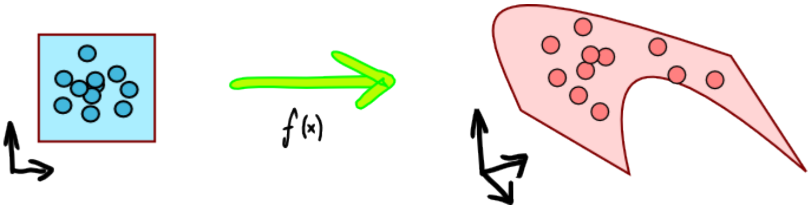
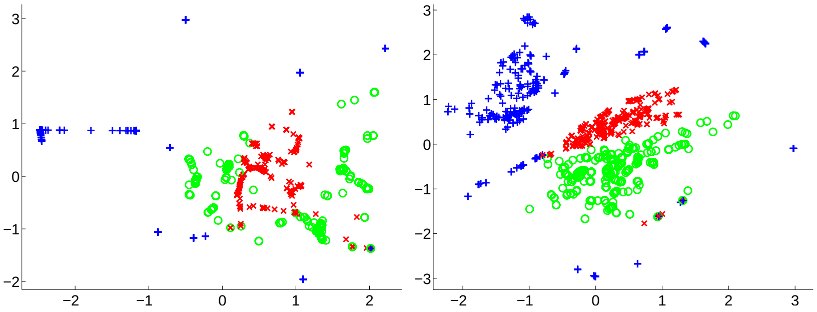


Fig. 1.9 Two dimensional Gaussian mapped to three dimensions.



(a) (b)

Fig. 1.10 Oil data visualised with a density network using an MLP network with (a) 100 (b) 400 points in the sample. Nearest neighbour errors: (a) 22 (b)16. Code can be run with (a) `demOilDnet4` (b) `demOilDnet5`

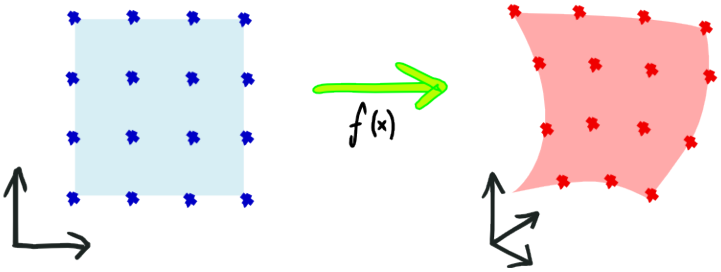


Fig. 1.11 One dimensional Gaussian mapped to two dimensions.

1.4 Likelihood Optimisation

1.4.1 Example: Oil Data

1.5 Generative Topographic Mapping

Generative Topographic Mapping (GTM) **?**

Key idea: Lay points out on a *grid.*

Constrained mixture of Gaussians.

1.5.1 GTM Prior

Prior distribution is a mixture model in a latent space.

$$p\left(\mathbf{X}\right)=\prod_{i=1}^{n}p\left(\mathbf{x}_{i,:}\right)$$

$$p\left(\mathbf{x}_{i,:}\right)=\frac{1}{m}\sum_{s=1}^{m}\delta\left(\mathbf{x}_{i,:}-\hat{\mathbf{x}}_{s,:}\right)$$

The $\hat{\mathbf{x}}_{s,:}$ are laid out on a regular grid.

1.5.2 Mapping and E-Step

Likelihood is a Gaussian with non-linear mapping from latent space to data space for the mean

$$p\left(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}\right)=\prod_{i=1}^{n}\prod_{j=1}^{p}\mathcal{N}\left(y_{i,j}|w_{j}\left(\mathbf{x}_{i,:};\mathbf{W},\ell\right),\sigma^{2}\right)$$

In the original paper **?** an RBF network was suggested,

In the E-step, posterior distribution over k is given by

$$r_{i,k}=\frac{\prod_{j=1}^{p}\mathcal{N}\left(y_{i,j}|f_{j}\left(\hat{\mathbf{x}}_{k};\mathbf{W},\ell\right),\sigma^{2}\right)}{\sum_{s=1}^{m}\prod_{j=1}^{p}\mathcal{N}\left(y_{i,j}|f_{j}\left(\hat{\mathbf{x}}_{s};\mathbf{W},\ell\right),\sigma^{2}\right)}$$

sometimes called the “responsibility of component k for data point i”.

1.5.3 Likelihood Optimisation

We then maximise the lower bound on the log likelihood,

$$\log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}\right)\geq\left\langle\log p\left(\mathbf{y}_{i,:},\hat{\mathbf{x}}_{s,:}|\boldsymbol{\theta}\right)\right\rangle_{q(s)}-\left\langle\log q\left(s\right)\right\rangle_{q(s)},$$

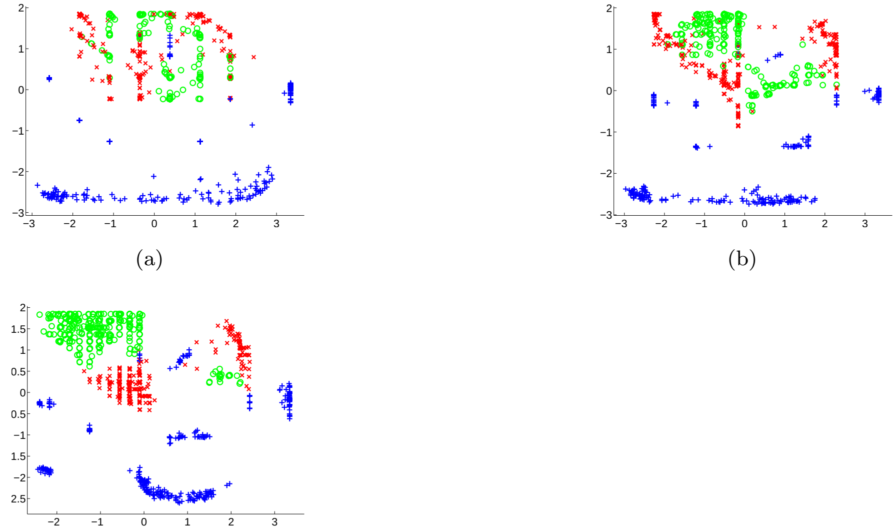
Free energy part of bound

$$\left\langle\log p\left(\mathbf{y}_{i,:},\hat{\mathbf{x}}_{s,:}|\boldsymbol{\theta}\right)\right\rangle=\sum_{s=1}^{m}r_{i,s}\log p\left(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{s,:},\boldsymbol{\theta}\right)+\text{const}$$

When optimising parameters in EM, we ignore dependence of $r_{i,k}$ on parameters. So we have

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}}\left\langle\log p\left(\mathbf{y}_{i,:},\hat{\mathbf{x}}_{s,:}|\boldsymbol{\theta}\right)\right\rangle=\sum_{s=1}^{m}r_{i,s}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}}\log p\left(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{s,:},\boldsymbol{\theta}\right)$$

which is very similar to density network result! Interpretation of posterior is slightly different.



(a)

(b)

(c)

Fig. 1.12 Oil data visualised with the GTM using an RBF network with (a) 10×10 (b) 20 × 20 and (c) 30 × 30 points in the grid. Nearest neighbour errors: (a) 74 (b) 44 (c) 11. These experiments can be recreated with (a) `demOilDnet1` (b) `demOilDnet2` (c) `demOilDnet3`.

1.5.4 Oil Data

1.5.5 Magnification Factors

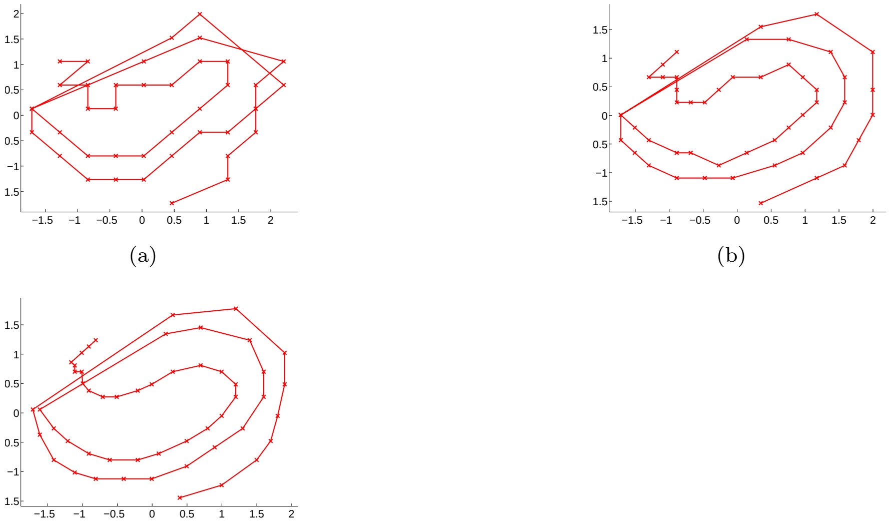
?

1.5.6 Stick Man Data

1.5.7 Separated Means: Bubblewrap Effect

1.5.8 Equivalence of GTM and Density Networks

GTM and Density Networks have the same origin. **??**.



(a)

(b)

(c)

Fig. 1.13 Oil data visualised with the GTM using an RBF network with (a) 10×10 (b) 20×20 (c) 30×30 points in the grid. Experiments can be recreated with (a) `demStickDnet1` (b) `demStickDnet2` (c) `demStickDnet3`.

In original Density Networks paper MacKay suggested Importance Sampling **?**.

Early work on GTM also used importance sampling.

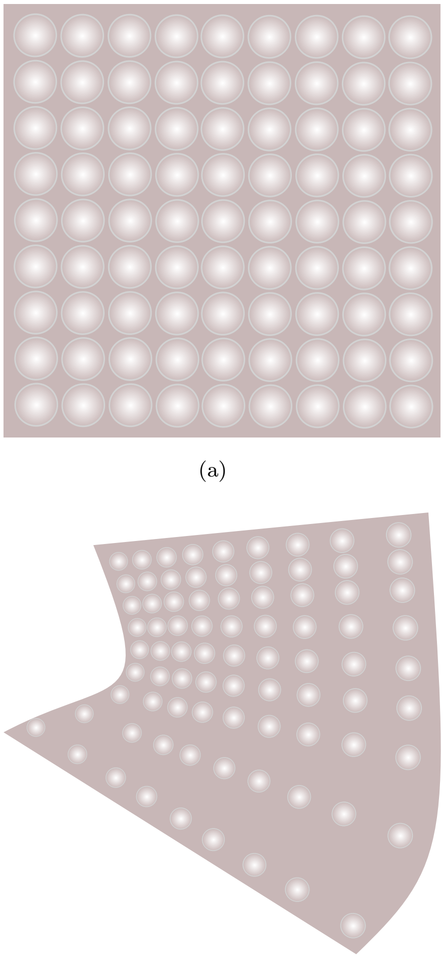
Main innovation in GTM was to lay points out on a grid (inspired by Self Organizing Maps **?**.

1.6 Non Linear Factor Analysis

Variational approach to dimensionality reduction.

Combine Gaussian prior over latent space with neural network **?**

Assume variational prior separates. Optimise with respect to variational distributions.



(a)

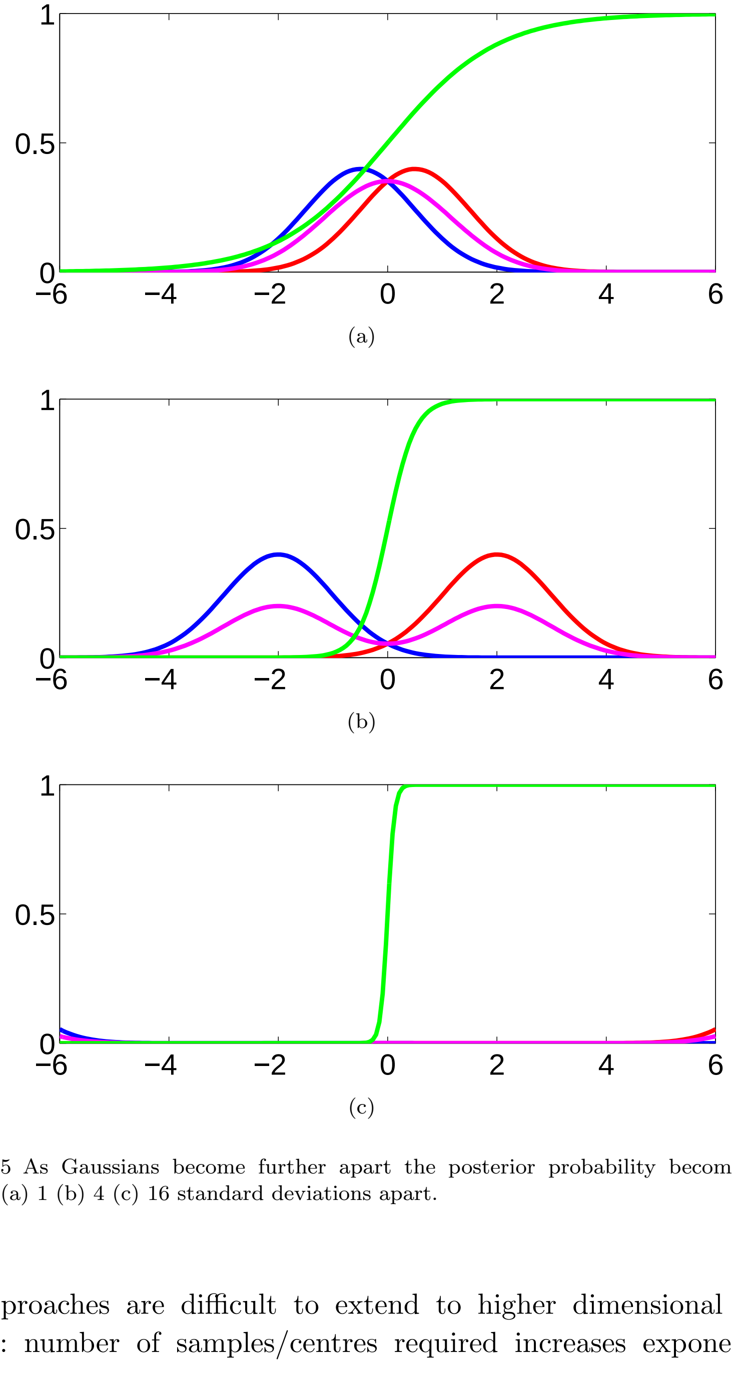
(b)

Fig. 1.14 The manifold is more like bubblewrap than a piece of paper.

1.6.1 Summary

Two point based approaches to dimensionality reduction.

Approaches seem to generalise well even when dimensions of data is greater than number of points.



(a)

(b)

(c)

Fig. 1.15 As Gaussians become further apart the posterior probability becomes more abrupt. (a) 1 (b) 4 (c) 16 standard deviations apart.

Approaches are difficult to extend to higher dimensional latent spaces: number of samples/centres required increases exponentially with dimension.

Next we will explore a different probabilistic interpretation of PCA and extend that to non-linear models.