Gaussian Models & Gaussian Processes

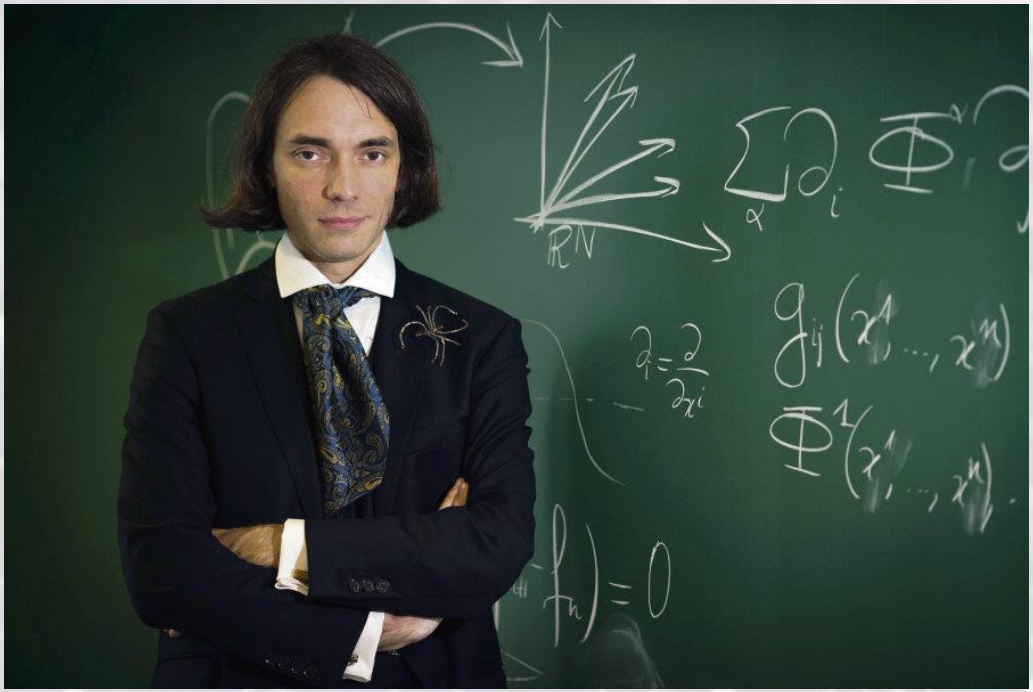
CS 446/546

Outline

- Overview
- MLE parameter estimates for the Gaussian Distribution
- LDA/QDA
- Marginal and Conditional Gaussians
- Maximum Entropy of the Gaussian
- Gaussian Processes

Overview

- Cedric Villani: "our supreme goddess of unreason – the Gauss curve."



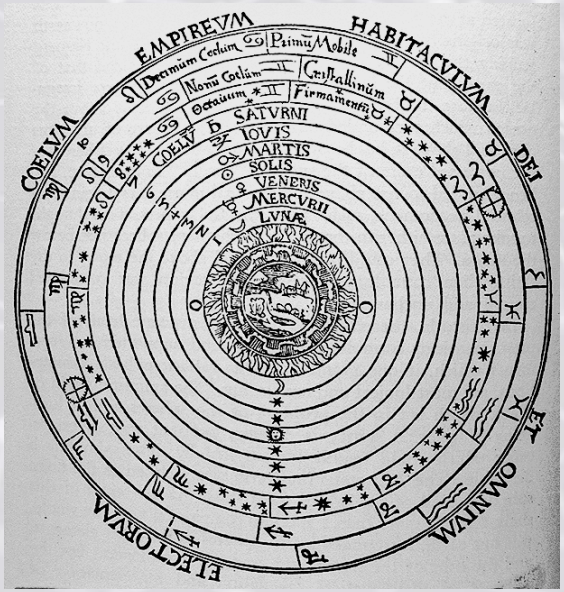
(*) Awarded Fermat Prize (2009), Fields medal (2010).

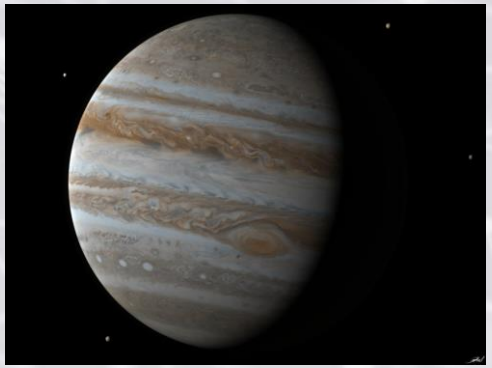
https://www.youtube.com/watch?v=Kc0Kthyo0hU

Overview: A brief history of the Normal Distribution

- The importance of the normal curve stems primarily from the fact that the distributions of copious natural phenomena are at least approximately normally distributed.

- One of the first applications of the normal distribution was to the **analysis of errors of measurement** made in astronomical observations, errors that were both due to instrument imprecision and human error. In the 17th century, Galileo observed that these errors were symmetric and that small errors occurred more frequently than large errors.

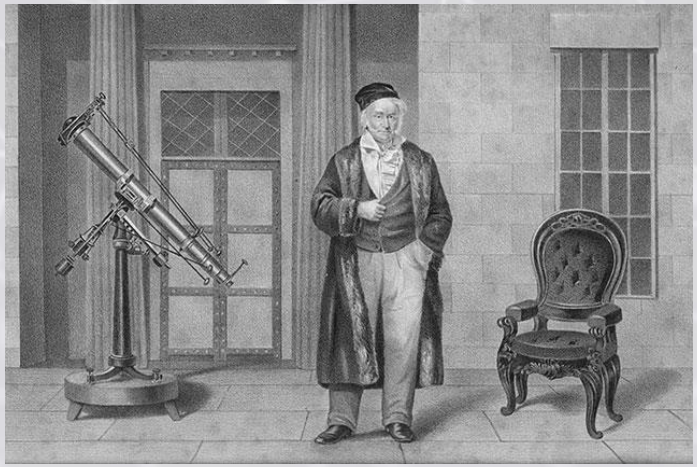




Overview: A brief history of the Normal Distribution

- This led to several hypothesized distributions of errors, but it was not until the early 19th century that it was discovered that these errors followed a normal distribution. Independently, the mathematicians Adrain in 1808 and Gauss in 1809 explicitly developed the formula for the normal distribution and showed that errors were fit well by this distribution.

- This same distribution had been discovered by Laplace in 1778 when he derived the central limit theorem. Laplace showed that even if a distribution is not normally distributed, the means of repeated samples from the distribution would be very nearly normally distributed, and that the larger the sample size, the closer the distribution of means would be to a normal distribution.



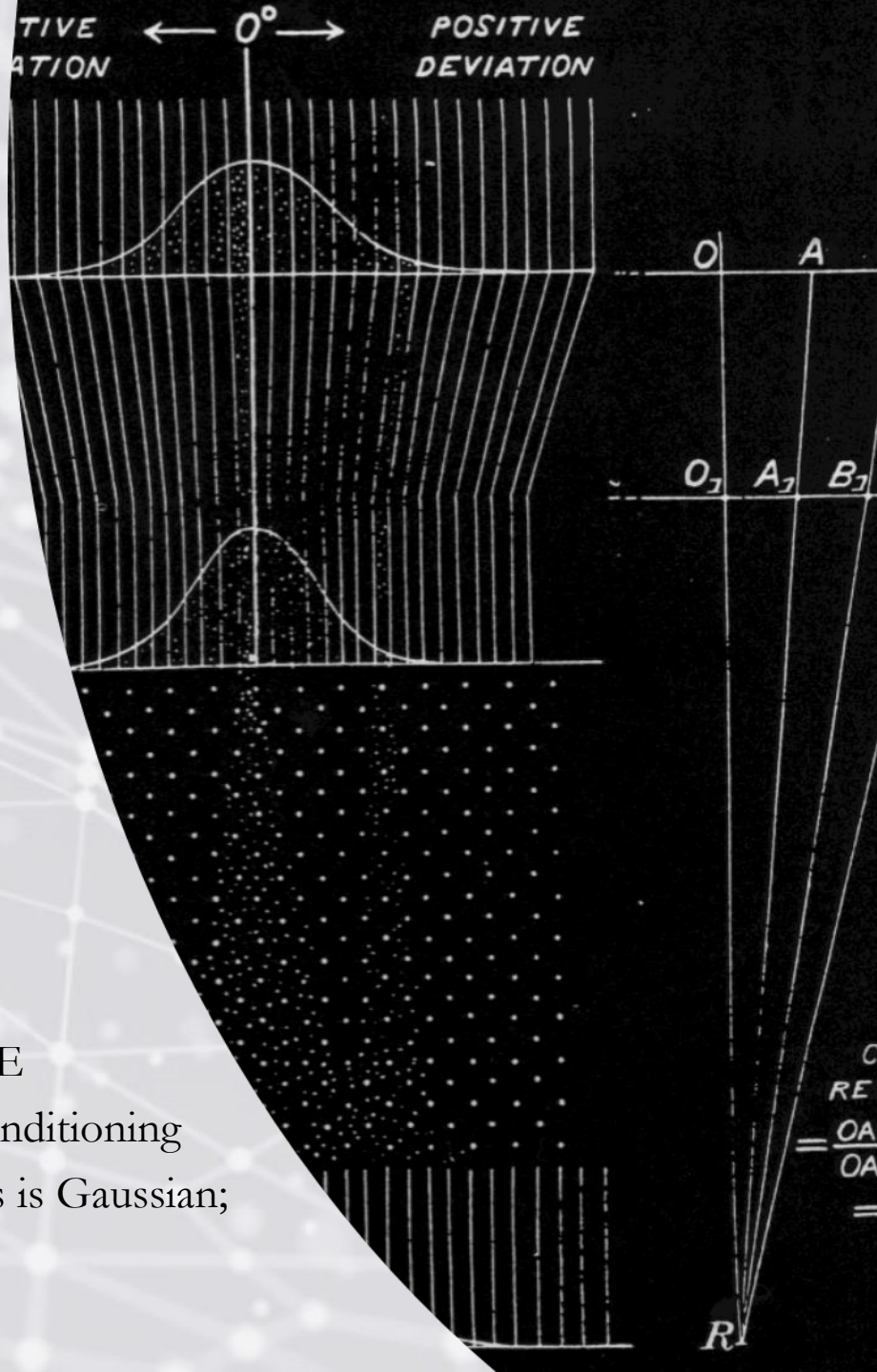
Overview: Why does everyone use Gaussians?

(*) There are at least (3) good reasons:

(1) The Central Limit Theorem (CLT).

(2) The Gaussian distribution has maximum entropy relative to all probability distributions with a fixed mean and standard deviations (i.e. up to *second moment* statistics).

(3) "Nice" computational properties: Gaussian MLE parameter estimates have closed-form formulae; conditioning a Gaussian yields a Gaussian; product of Gaussians is Gaussian; member of *exponential family* distributions.

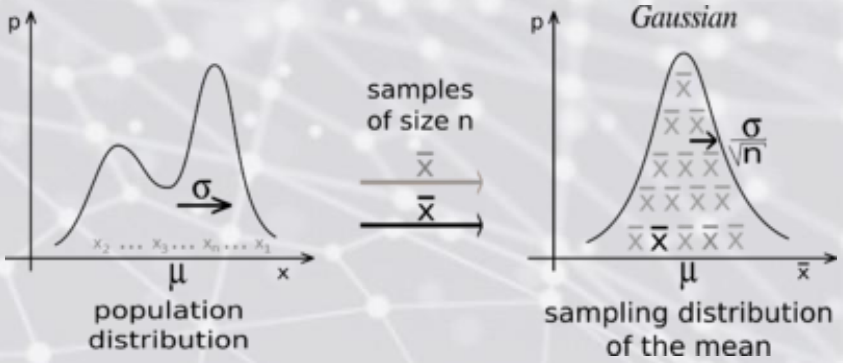


Overview: CLT

The *Central Limit Theorem* (a conceptual pillar of statistics)

In words: given a sufficiently large sample size from a population (with a finite level of variance), the mean of all samples from the same population will be approximately equal to the mean of the population. Furthermore, all of the samples will follow an approximate normal distribution pattern, with all variances being approximately equal to the variance of the population divided by each sample's size.

In a picture:



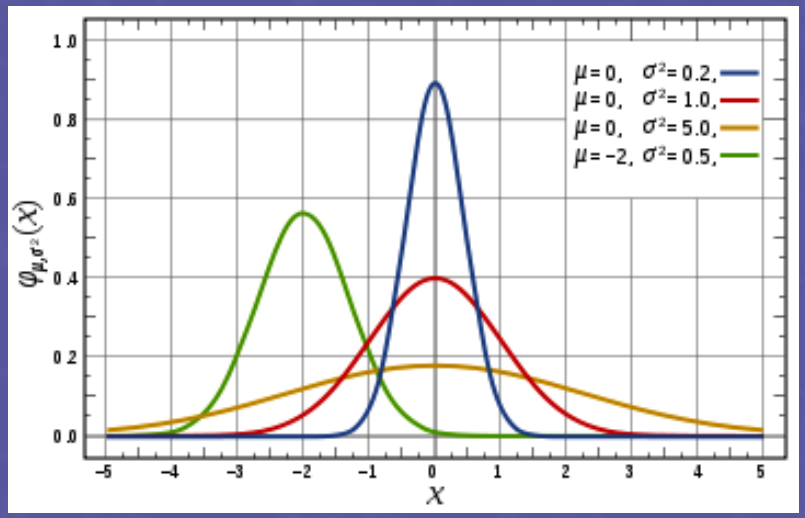
Whatever the form of the population distribution, the sampling distribution tends to a Gaussian, and its dispersion is given by the Central Limit Theorem.

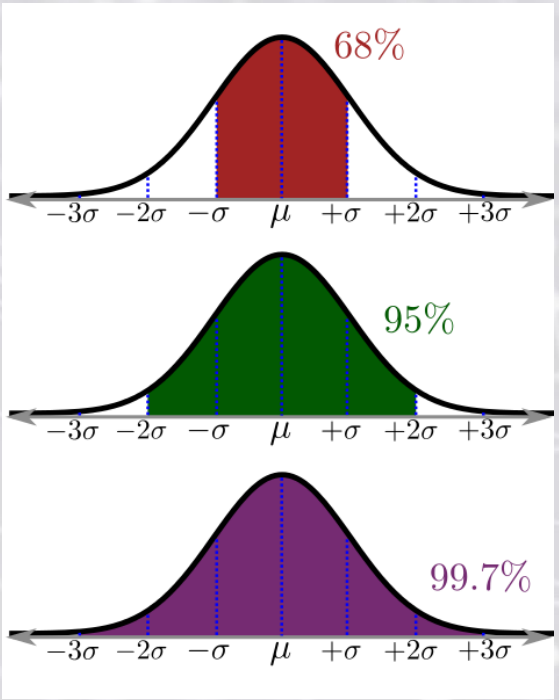
In a theorem: Suppose $\{X_1, X_2, \ldots,\}$ is a sequence of I.I.D. random variables with $E[X_i]=\mu$ and $Var[X_i]=\sigma^2<\infty$ Then as n approaches infinity, the random variable $(1/n)(X_1+\ldots+X_n)$ converges to a normal $N(0, \sigma^2/n)$:

$$\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right)-\mu\right)\xrightarrow{d} N\left(0,\frac{\sigma^2}{n}\right)$$

Overview

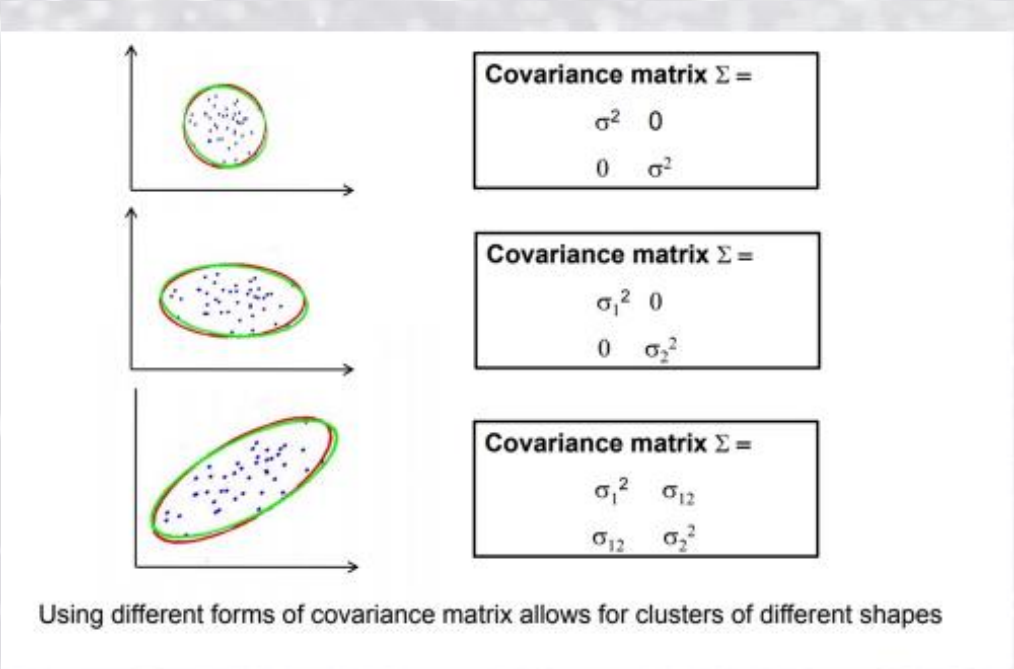
$$N(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



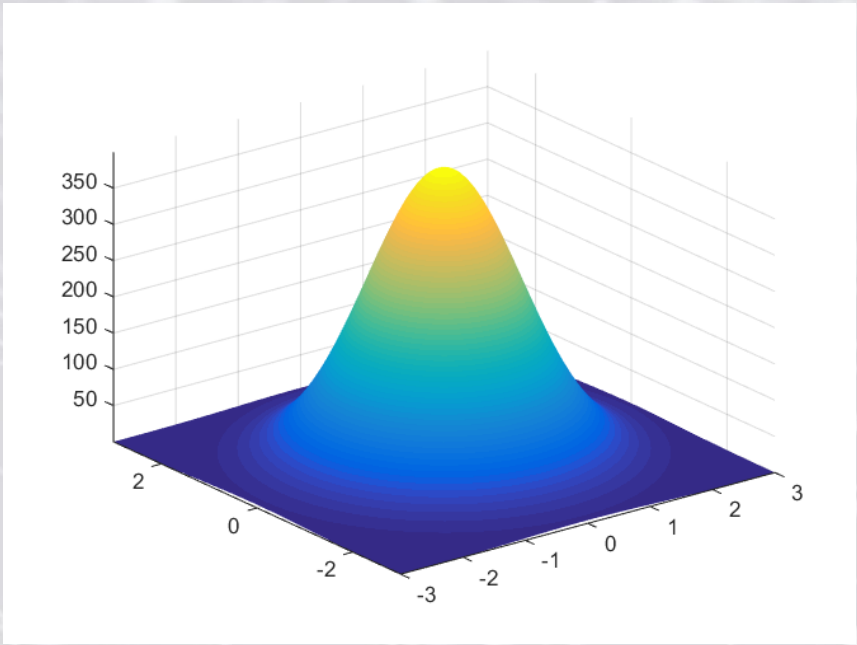


Overview

$$N(x;\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}$$



Using different forms of covariance matrix allows for clusters of different shapes



Information Form of the Gaussian

- The multi-variate Gaussian distribution is also frequently expressed in an alternative form called the **Information form**.

- Suppose $\mathbf{x} \sim N(\mu,\Sigma)$; the information form (also called canonical form) is defined as:

- Let $\Lambda = \Sigma^{-1}, \xi = \Sigma^{-1}\boldsymbol{\mu}$, then: $\boldsymbol{\mu} = \Lambda^{-1}\xi, \Sigma = \Lambda^{-1}$:

$$N\left(\mathbf{x} \mid \xi, \Lambda\right) = \frac{|\Lambda|}{\left(2\pi\right)^{D/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}^T \Lambda \mathbf{x} + \xi^T \Lambda^{-1} \xi - 2\mathbf{x}^T \xi\right)\right]$$

• Where Λ is called the **precision matrix**; each component of Λ quantifies the "partial correlation" between two random variables.

• **Partial correlation** is in the range [-1,1] and controls for confounding variables in measuring the association between two random variables; for instance, the (i,j) entry in the precision matrix yields the correlation between X_i and X_j, controlling for all other variables in the model.

MLE for the Gaussian

• In any beginning statistics class, students are inevitably shown the formulae for the empirical estimates of the parameters of a Gaussian distribution :

$$\bar{x} = \frac{1}{N}\sum_i x_i \quad \hat{\sigma}^2 = \frac{1}{N}\sum_i \left(x_i - \bar{x}\right)^2$$

Q: Where do these results come from? Are they the only way to estimate the parameters of a Gaussian distribution? Are they *good* estimates?

MLE for the Gaussian

• In any beginning statistics class, students are inevitably shown the formulae for the empirical estimates of the parameters of a Gaussian distribution :

$$\bar{x} = \frac{1}{N}\sum_i x_i \quad \hat{\sigma}^2 = \frac{1}{N}\sum_i (x_i - \bar{x})^2$$

Q: Where do these results come from? Are they the only way to estimate the parameters of a Gaussian distribution? Are they *good* estimates?

A: They are derived from the maximum likelihood estimation procedure (i.e., the MLE, which Fisher first popularized); in fact, there are a number of different methods for estimating distribution parameters – MLE just happens to be one of the most commonly employed methods.

• One could, for instance, follow a Bayesian procedure (i.e. MAP) to estimate the distribution parameters – the only catch is that we then need to define a *prior* (when in doubt, we adhere to the *principle of insufficient reason* and use an **uninformative prior** – what is this equivalent to?)

• Note that the MLE procedure tends to overfit as an estimate; also observe that it can produce **biased estimates** (however this is not necessarily a bad outcome *per se*). The MLE for variance for the Gaussian is biased, for example.

MLE for the Gaussian

$$\bar{x} = \frac{1}{N}\sum_i x_i \quad \hat{\sigma}^2 = \frac{1}{N}\sum_i (x_i - \bar{x})^2$$

- To get a better handle on the MLE procedure, we next derive the MLE formulae for the Gaussian.
- First, however, we need a few results from matrix algebra (these results are exceptionally useful when taking derivatives/optimizing matrix equations):

$$(1)\ \frac{\partial(\mathbf{b}^T\mathbf{a})}{\partial\mathbf{a}} = \mathbf{b}$$

$$(2)\ \frac{\partial(\mathbf{a}^T A\mathbf{a})}{\partial\mathbf{a}} = (A + A^T)\mathbf{a}$$

$$(3)\ \frac{\partial}{\partial A} trace(BA) = B$$

$$(4)\ \frac{\partial}{\partial A}\log|A| = A^{-T} = (A^{-1})^T$$

$$(5)\ trace(ABC) = trace(CAB) = trace(BCA)$$

(*) Note that the **trace(A)** of a square matrix is defined as the sum of its diagonal entries.

(*) Lastly, a useful formulation known as "**trace trick**" asserts, from (5):

$$\mathbf{x}^T A\mathbf{x} = \text{trace}(\mathbf{x}^T A\mathbf{x}) = \text{trace}(\mathbf{x}\mathbf{x}^T A) = \text{trace}(A\mathbf{x}\mathbf{x}^T)$$

MLE for the Gaussian

$$\bar{x} = \frac{1}{N}\sum_i x_i \quad \hat{\sigma}^2 = \frac{1}{N}\sum_i (x_i - \bar{x})^2$$

• Let's now prove the classic MLE formulae for estimating the parameters of a Gaussian distribution.

Pf.

(I) Estimate of the mean ($\hat{\mu}$):

First we begin with the log-likelihood expression (dropping additive constants):

$$l(\boldsymbol{\mu}, \Sigma) = \log p(D \mid \boldsymbol{\mu}, \Sigma) = \frac{N}{2}\log|\Lambda| - \frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})^T \Lambda (\mathbf{x}_i - \boldsymbol{\mu})$$

where $\Lambda = \Sigma^{-1}$ is the precision matrix.

Q: What next?

MLE for the Gaussian

Pf.

(I) Estimate of the mean ($\hat{\mu}$):

• First we begin with the log-likelihood expression (dropping additive constants):

$$l(\boldsymbol{\mu},\Sigma)=\log p(D\,|\,\boldsymbol{\mu},\Sigma)=\frac{N}{2}\log|\Lambda|-\frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})$$

• Naturally, we take the partial derivative of the log-likelihood with respect to $\boldsymbol{\mu}$ and set it equal to zero. For notational convenience, we use the substitution: $\mathbf{y}_i=\mathbf{x}_i-\boldsymbol{\mu}$.

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=\frac{\partial}{\partial\boldsymbol{\mu}}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})=\frac{\partial}{\partial\mathbf{y}_i}\mathbf{y}_i\Sigma^{-1}\mathbf{y}_i\frac{\partial\mathbf{y}_i}{\partial\boldsymbol{\mu}}$$

Why?

MLE for the Gaussian

Pf.

(I) Estimate of the mean ($\hat{\mu}$):

• First we begin with the log-likelihood expression (dropping additive constants):

$$l(\boldsymbol{\mu},\Sigma)=\log p(D\,|\,\boldsymbol{\mu},\Sigma)=\frac{N}{2}\log|\Lambda|-\frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})$$

• Naturally, we take the partial derivative of the log-likelihood with respect to $\boldsymbol{\mu}$ and set it equal to zero. For notational convenience, we use the substitution: $\mathbf{y}_i=\mathbf{x}_i-\boldsymbol{\mu}$.

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=\frac{\partial}{\partial\boldsymbol{\mu}}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})=\frac{\partial}{\partial\mathbf{y}_i}\mathbf{y}_i\Sigma^{-1}\mathbf{y}_i\frac{\partial\mathbf{y}_i}{\partial\boldsymbol{\mu}}$$

$$=-1\left(\Sigma^{-1}+\Sigma^{-T}\right)\mathbf{y}_i$$

Why?

MLE for the Gaussian

Pf.

(I) Estimate of the mean ($\hat{\mu}$):

• First we begin with the log-likelihood expression (dropping additive constants):

$$l(\boldsymbol{\mu},\Sigma)=\log p(D\,|\,\boldsymbol{\mu},\Sigma)=\frac{N}{2}\log|\Lambda|-\frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})$$

• Naturally, we take the partial derivative of the log-likelihood with respect to $\boldsymbol{\mu}$ and set it equal to zero. For notational convenience, we use the substitution: $\mathbf{y}_i=\mathbf{x}_i-\boldsymbol{\mu}$.

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=\frac{\partial}{\partial\boldsymbol{\mu}}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})=\frac{\partial}{\partial\mathbf{y}_i}\mathbf{y}_i\Sigma^{-1}\mathbf{y}_i\frac{\partial\mathbf{y}_i}{\partial\boldsymbol{\mu}}$$

$$=-1\left(\Sigma^{-1}+\Sigma^{-T}\right)\mathbf{y}_i$$

Why? (2) $\dfrac{\partial(\mathbf{a}^TA\mathbf{a})}{\partial\mathbf{a}}=(A+A^T)\mathbf{a}$

MLE for the Gaussian

Pf.

(I) Estimate of the mean ($\hat{\mu}$):

• First we begin with the log-likelihood expression (dropping additive constants):

$$l(\boldsymbol{\mu},\Sigma)=\log p(D\,|\,\boldsymbol{\mu},\Sigma)=\frac{N}{2}\log|\Lambda|-\frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})$$

• Naturally, we take the partial derivative of the log-likelihood with respect to $\boldsymbol{\mu}$ and set it equal to zero. For notational convenience, we use the substitution: $\mathbf{y}_i=\mathbf{x}_i-\boldsymbol{\mu}$.

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=\frac{\partial}{\partial\boldsymbol{\mu}}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})=\frac{\partial}{\partial\mathbf{y}_i}\mathbf{y}_i\Sigma^{-1}\mathbf{y}_i\frac{\partial\mathbf{y}_i}{\partial\boldsymbol{\mu}}$$

$$=-1\left(\Sigma^{-1}+\Sigma^{-T}\right)\mathbf{y}_i$$

Why?

This implies:

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=-\frac{1}{2}\sum_{i=1}^{N}-2\Sigma^{-1}(\mathbf{x}_i-\boldsymbol{\mu})=\Sigma^{-1}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})$$

MLE for the Gaussian

Pf.

(I) Estimate of the mean ($\hat{\mu}$):

• First we begin with the log-likelihood expression (dropping additive constants):

$$l(\boldsymbol{\mu},\Sigma)=\log p(D\,|\,\boldsymbol{\mu},\Sigma)=\frac{N}{2}\log|\Lambda|-\frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})$$

• Naturally, we take the partial derivative of the log-likelihood with respect to $\boldsymbol{\mu}$ and set it equal to zero. For notational convenience, we use the substitution: $\mathbf{y}_i=\mathbf{x}_i-\boldsymbol{\mu}$.

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=\frac{\partial}{\partial\boldsymbol{\mu}}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})=\frac{\partial}{\partial\mathbf{y}_i}\mathbf{y}_i\Sigma^{-1}\mathbf{y}_i\frac{\partial\mathbf{y}_i}{\partial\boldsymbol{\mu}}$$

$$=-1\left(\Sigma^{-1}+\Sigma^{-T}\right)\mathbf{y}_i$$

Why? $\left(\Sigma^T=\Sigma\right)$

This implies:

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=-\frac{1}{2}\sum_{i=1}^{N}-2\Sigma^{-1}(\mathbf{x}_i-\boldsymbol{\mu})=\Sigma^{-1}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})$$

MLE for the Gaussian

Pf.

(I) Estimate of the mean ($\hat{\mu}$):

$$l(\boldsymbol{\mu},\Sigma)=\log p(D\,|\,\boldsymbol{\mu},\Sigma)=\frac{N}{2}\log|\Lambda|-\frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})^T\Lambda(\mathbf{x}_i-\boldsymbol{\mu})$$

• Naturally, we take the partial derivative of the log-likelihood with respect to $\boldsymbol{\mu}$ and set it equal to zero. For notational convenience, we use the substitution: $\mathbf{y}_i=\mathbf{x}_i-\boldsymbol{\mu}$.

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=\frac{\partial}{\partial\boldsymbol{\mu}}(\mathbf{x}_i-\boldsymbol{\mu})^T\,\Lambda(\mathbf{x}_i-\boldsymbol{\mu})=\frac{\partial}{\partial\mathbf{y}_i}\mathbf{y}_i\Sigma^{-1}\mathbf{y}_i\,\frac{\partial\mathbf{y}_i}{\partial\boldsymbol{\mu}}$$

$$=-1\left(\Sigma^{-1}+\Sigma^{-T}\right)\mathbf{y}_i$$

This implies:

$$\frac{\partial}{\partial\boldsymbol{\mu}}l(\boldsymbol{\mu},\Sigma)=-\frac{1}{2}\sum_{i=1}^{N}-2\Sigma^{-1}(\mathbf{x}_i-\boldsymbol{\mu})=\Sigma^{-1}\sum_{i=1}^{N}(\mathbf{x}_i-\boldsymbol{\mu})$$

Finally, when we set the partial derivative equal to zero and solve it yields:

$$\hat{\boldsymbol{\mu}}=\frac{1}{N}\sum_{i=1}^{N}x_i=\bar{x}$$

, the standard empirical mean formula.

MLE for the Gaussian

Pf.

(II) Estimate of the covariance matrix ($\hat{\Sigma}$):

$$l(\boldsymbol{\mu}, \Sigma) = \log p(D \mid \boldsymbol{\mu}, \Sigma) = \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^T \Lambda (\mathbf{x}_i - \boldsymbol{\mu})$$

• This time we take the partial derivative of the log-likelihood with respect to Λ and set it equal to zero. For notational convenience, we use the substitution: $S_\mu = \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$

MLE for the Gaussian

Pf.

(II) Estimate of the covariance matrix ($\hat{\Sigma}$):

$$l(\boldsymbol{\mu}, \Sigma) = \log p(D \mid \boldsymbol{\mu}, \Sigma) = \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^T \Lambda (\mathbf{x}_i - \boldsymbol{\mu})$$

• This time we take the partial derivative of the log-likelihood with respect to Λ and set it equal to zero. For notational convenience, we use the substitution: $S_\mu = \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$

First, we use the "trace trick" to rewrite the log-likelihood as:

$$l(\Lambda) = \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} trace\left[(\mathbf{x}_i - \boldsymbol{\mu})^T \Lambda (\mathbf{x}_i - \boldsymbol{\mu}) \right]$$

MLE for the Gaussian

Pf.

(II) Estimate of the covariance matrix ($\hat{\Sigma}$):

$$l(\boldsymbol{\mu}, \Sigma) = \log p(D \mid \boldsymbol{\mu}, \Sigma) = \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^T \Lambda (\mathbf{x}_i - \boldsymbol{\mu})$$

• This time we take the partial derivative of the log-likelihood with respect to Λ and set it equal to zero. For notational convenience, we use the substitution: $S_\mu = \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$

First, we use the "trace trick" to rewrite the log-likelihood for as:

$$l(\Lambda) = \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} trace\left[(\mathbf{x}_i - \boldsymbol{\mu})^T \Lambda (\mathbf{x}_i - \boldsymbol{\mu}) \right]$$

$$= \frac{N}{2} \log |\Lambda| - \frac{1}{2} trace\left[S_\mu \Lambda \right]$$

MLE for the Gaussian

Pf.

(II) Estimate of the covariance matrix ($\hat{\Sigma}$):

$$l(\Lambda) = \frac{N}{2}\log|\Lambda| - \frac{1}{2}trace\left[S_\mu \Lambda\right]$$

• This time we take the partial derivative of the log-likelihood with respect to Λ and set it equal to zero. For notational convenience, we use the substitution: $S_\mu = \sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$

$$\frac{\partial l(\Lambda)}{\partial \Lambda} = \frac{N}{2}\Lambda^{-T} - \frac{1}{2}S_\mu^T = 0$$

Why?

MLE for the Gaussian

Pf.

(II) Estimate of the covariance matrix ($\hat{\Sigma}$):

$$l(\Lambda) = \frac{N}{2}\log|\Lambda| - \frac{1}{2}trace\left[S_\mu \Lambda\right]$$

• This time we take the partial derivative of the log-likelihood with respect to Λ and set it equal to zero. For notational convenience, we use the substitution: $S_\mu = \sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$

$$\frac{\partial l(\Lambda)}{\partial \Lambda} = \frac{N}{2}\Lambda^{-T} - \frac{1}{2}S_\mu^T = 0$$

Why?

(3) $\frac{\partial}{\partial A}trace(BA) = B$

(4) $\frac{\partial}{\partial A}\log|A| = A^{-T} = \left(A^{-1}\right)^T$

MLE for the Gaussian

Pf.

(II) Estimate of the covariance matrix ($\hat{\Sigma}$):

$$l(\Lambda) = \frac{N}{2}\log|\Lambda| - \frac{1}{2}trace\left[S_\mu \Lambda\right]$$

• This time we take the partial derivative of the log-likelihood with respect to Λ and set it equal to zero. For notational convenience, we use the substitution: $S_\mu = \sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$

$$\frac{\partial l(\Lambda)}{\partial \Lambda} = \frac{N}{2}\Lambda^{-T} - \frac{1}{2}S_\mu^T = 0$$

This implies: $\Lambda^{-T} = \Lambda^{-1} = \Sigma = \frac{1}{N}S_\mu$

Why?

MLE for the Gaussian

Pf.

(II) Estimate of the covariance matrix ($\hat{\Sigma}$):

$$l(\Lambda) = \frac{N}{2}\log|\Lambda| - \frac{1}{2}trace\left[S_\mu \Lambda\right]$$

• This time we take the partial derivative of the log-likelihood with respect to Λ and set it equal to zero. For notational convenience, we use the substitution: $S_\mu = \sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$

$$\frac{\partial l(\Lambda)}{\partial \Lambda} = \frac{N}{2}\Lambda^{-T} - \frac{1}{2}S_\mu^T = 0$$

This implies: $\Lambda^{-T} = \Lambda^{-1} = \Sigma = \frac{1}{N}S_\mu$

Finally, we have:

$$\hat{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$$

which is precisely the standard empirical covariance formula, as was to be shown.

LDA

• **Linear Discriminant Analysis** (LDA) and **Quadratic Discriminant Analysis** (QDA) are two common, related regimes in ML for modelling class posteriors, i.e., $P(Y=k|X=x)$.

• In this approach, we model the distribution of the predictors X separately in each of the response classes (i.e. given Y: $P(X=x|Y=k)$) and then use Bayes' theorem to flip these around into estimates for the class posteriors.

LDA

• **Linear Discriminant Analysis** (LDA) and **Quadratic Discriminant Analysis** (QDA) are two common, related regimes in ML for modelling class posteriors, i.e., $P(Y=k|X=x)$.

• In this approach, we model the distribution of the predictors X separately in each of the response classes (i.e. given Y: $P(X=x|Y=k)$) and then use Bayes' theorem to flip these around into estimates for the class posteriors.

• When these distributions are assumed to be normal we get the LDA/QDA method, which is very similar to logistic regression in the case of two classes – recall that thresholding a logistic function gives a linear decision boundary.

• However, unlike logistic regression, LDA/QDA tend to be more numerically stable (recall that logistic regression requires *gradient ascent* or some such approximation technique); in addition, LDA/QDA are easily adaptable to the case of classification for more than two classes.

LDA: Bayes' Rule for Classification

- Recall that Bayes' Theorem states:

$$P(Y=k \mid X=x)=\frac{\pi_k f_k(x)}{\sum_{l=1}^{K}\pi_l f_l(x)}$$

Where we let $f_k(x)=P(X=x \mid Y=k)$ denote the *density function* for X for an observation that comes from the *k*th class; π_k represents the *prior probability* that randomly chosen datum comes from the *k*th class.

- For LDA/QDA we assume that $f_k(x)$ is Gaussian; in the 1-D case:

$$f_k(x)=\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left[-\frac{1}{2\sigma_k^2}(x-\mu_k)^2\right]$$

In addition, with LDA in particular, we assume the variances across the classes are equal, namely: $\sigma_1^2=\sigma_2^2=\cdots\sigma_k^2$.

LDA

$$P(Y=k \mid X=x)=\frac{\pi_k f_k(x)}{\sum_{l=1}^{K}\pi_l f_l(x)} \qquad f_k(x)=\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left[-\frac{1}{2\sigma_k^2}(x-\mu_k)^2\right]$$

- By Bayes' Theorem we have:

$$p_k(x)=\frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma_k}\exp\left[-\frac{1}{2\sigma_k^2}(x-\mu_k)^2\right]}{\sum_{l=1}^{K}\pi_l \frac{1}{\sqrt{2\pi}\sigma_l}\exp\left[-\frac{1}{2\sigma_l^2}(x-\mu_l)^2\right]}$$

Taking the log and rearranging terms, one can show that this is equivalent to assigning the observation to the class for which:

$$\hat{\delta}_k(x)=x\frac{\hat{\mu}_k}{\hat{\sigma}^2}-\frac{\hat{\mu}_k^2}{2\hat{\sigma}^2}+\log(\hat{\pi}_k)$$

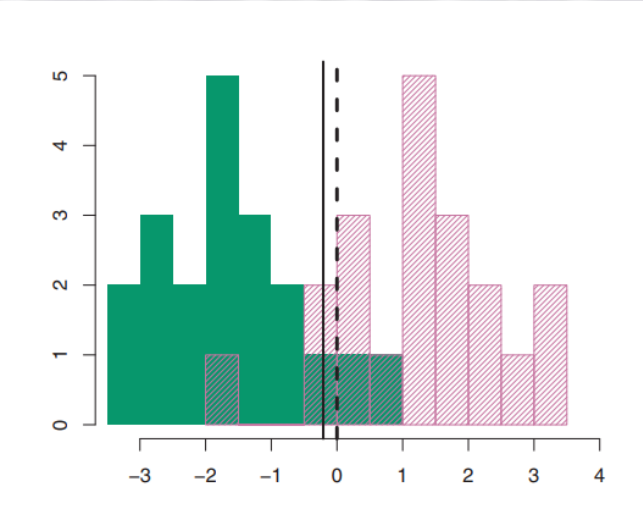
is largest. Here each of the parameter estimates $\hat{\mu}_k, \hat{\sigma}_k^2, \hat{\pi}_k = nk/n$ are the standard MLE estimates derived previously. The term *linear* (per LDA) applies because the discriminant function $\hat{\delta}_k$ is a linear function of the input x, and hence the decision boundary is a line.

LDA

(*) LDA for 2 classes (with 1-D data) is equivalent to assigning the observation to the class for which:

$$\hat{\delta}_k(x) = x\frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + \log(\hat{\pi}_k)$$

is largest.



• In the image, 20 observations were drawn from each of two classes, and are shown as histograms. The Bayes' decision boundary (optimal) is shown as a dashed line; the solid vertical line represents the LDA decision boundary estimated from training data.

LDA: Higher Dimensions & Multiple Classes

• LDA can be extended in a natural way to incorporate data in more than one dimension and more than two classes. Previously, in the 1-D case, we assumed the variance for each class was equal; now we assume that the covariance matrix for each class is identical, i.e. $\Sigma_c = \Sigma$ (for all classes); this is sometimes called **parameter sharing/tying**.

LDA: Higher Dimensions & Multiple Classes

• LDA can be extended in a natural way to incorporate data in more than one dimension and more than two classes. Previously, in the 1-D case, we assumed the variance for each class was equal; now we assume that the covariance matrix for each class is identical, i.e. $\Sigma_c = \Sigma$ (for all classes); this is sometimes called **parameter sharing/tying**.

• Using Bayes' Theorem again, this time with multi-variate Gaussian density functions yields:

$$p_k(x) = \frac{\pi_k \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\boldsymbol{\mu}_k)^T \Sigma (x-\boldsymbol{\mu}_k)\right]}{\sum_{l=1}^{K} \pi_l \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\boldsymbol{\mu}_l)^T \Sigma (x-\boldsymbol{\mu}_l)\right]}$$

(*) Taking logs and rearranging terms renders the following expression; LDA assigns a new datum x to the class for which:

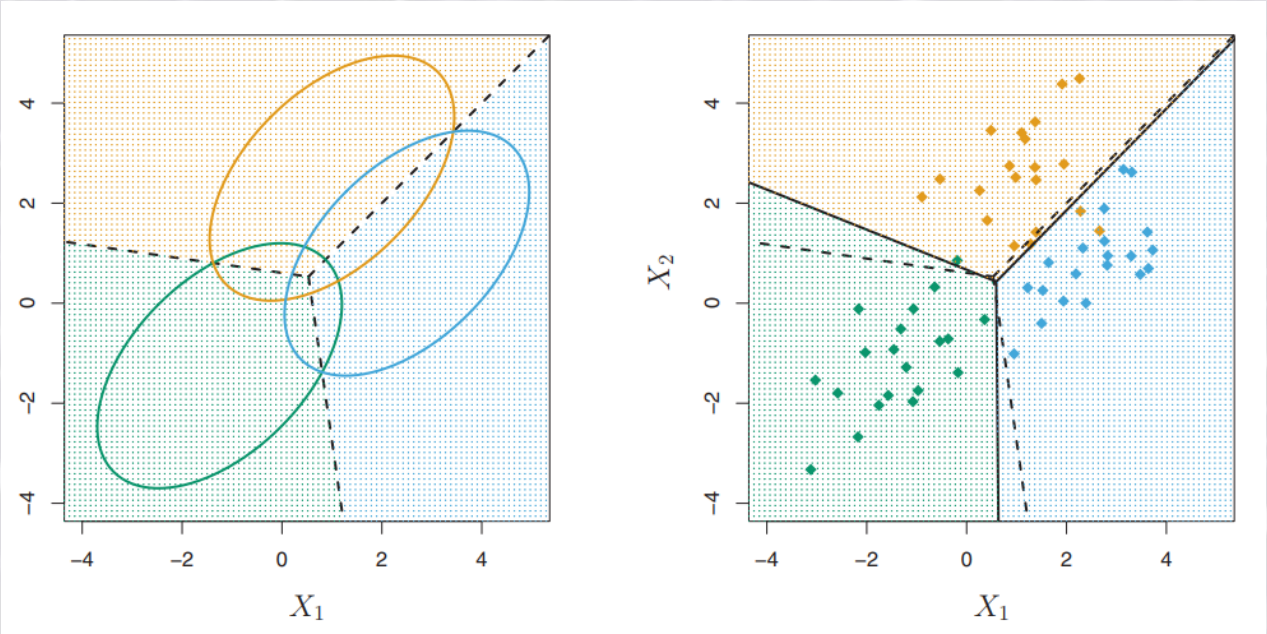
$$\hat{\delta}_k(x) = \mathbf{x}^T \Sigma^{-1} \hat{\boldsymbol{\mu}}_k - \frac{1}{2} \hat{\boldsymbol{\mu}}_k^T \Sigma^{-1} \hat{\boldsymbol{\mu}}_k + \log(\hat{\pi}_k)$$

is largest. Again, we use the MLE estimates for the parameters in the equation; the decision boundary is again linear, as the discriminant function is linear in x.

LDA: Higher Dimensions & Multiple Classes

$$\hat{\delta}_k(x) = \mathbf{x}^T \Sigma^{-1} \hat{\boldsymbol{\mu}}_k - \frac{1}{2} \hat{\boldsymbol{\mu}}_k^T \Sigma^{-1} \hat{\boldsymbol{\mu}}_k + \log(\hat{\pi}_k)$$

• The figure shows an LDA example with 3 classes (d=2); the observations from each class are drawn from a multi-variate Gaussian distribution. Left: ellipses that contain 95% of the probability for each of three classes are shown; the dashed lines are the optimal (Bayes') decision boundary. Right: 20 observations were generated from each class, and the corresponding LDA decision boundaries are indicated using solid black lines.



LDA: Case Study

• Now that we've seen the method to both fit an LDA model as well as perform posterior class inference, we next consider a case study using financial data with LDA to review some common model assessment procedures for classification in ML.

• We use the *default* data set from the ISLR database (linked below); the data consist of 10,000 training observation with 4 variables: *income* (income of customer), *balance* (average monthly credit card balance), *student* (yes/no), *default* (yes/no) – we want to predict whether a customer will default on their credit card payment.

LDA: Case Study

• Now that we've seen the method to both fit an LDA model as well as perform posterior class inference, we next consider a case study using financial data with LDA to review some common model assessment procedures for classification in ML.

• We use the *default* data set from the ISLR database (linked below); the data consist of 10,000 training observation with 4 variables: *income* (income of customer), *balance* (average monthly credit card balance), *student* (yes/no), *default* (yes/no) – we want to predict whether a customer will default on their credit card payment.

• Results were calculated using LDA on the default data: the *training error* was 2.75% -- this seems like a strong result at first blush, however, recall that training error is an optimistic estimate of test error.

In addition the data are highly **imbalanced** (only 3.33% of the customers defaulted). This means that a trivial **null classifier** (viz., a classifier that always predicts not default will only garner a 3.33% training error rate).

https://cran.r-project.org/web/packages/ISLR/ISLR.pdf

LDA: Case Study

• The confusion matrix for the *default* data using an LDA:

		True default status		
		No	Yes	Total
Predicted default status	No	9,644	252	9,896
	Yes	23	81	104
	Total	9,667	333	10,000

• From the confusion matrix, the **TN** (true negative rate) = 9,644/10,000 = 96.44%, the **FN** (false negative rate) = 252/10,000 = 2.52%, the **TP** (true positive rate) = 81/10,000 = 0.81% and the **FP** (false positive rate) = 23/10,000 = 0.23%.

• While these results appear strong (at least for the training set) because the overall error rate is low, the error rate among individuals who defaulted is actually very high (the unbalanced data masks this).

• Class-specific performance can be more important than overall accuracy of a classifier in many domains – particularly biology and medicine. In particular, *sensitivity* and *specificity* are often used to characterize the performance of a classifier (say for a screening test).

In this case: **sensitivity** (TP/P)=81/333=24.3% (very low!) ; **specificity** (TN/N)=9,644/9,667=99.8%

LDA: Case Study

• The confusion matrix for the *default* data using an LDA:

		True default status		
		No	Yes	Total
Predicted default status	No	9,644	252	9,896
	Yes	23	81	104
	Total	9,667	333	10,000

In this case: **sensitivity** (TP/P)=81/333=24.3% (very low!) ; **specificity** (TN/N)=9,644/9,667=99.8%

Can we improve the sensitivity score? Quite possibly.

If we are concerned about incorrectly predicting default status for individuals who default, then we can consider lowering the probability threshold for LDA. Originally, we predicted the default class whenever the posterior threshold exceeded 0.5: P(default=Yes|X=x)>0.5.

But, if we lower this threshold this might improve the sensitivity of the model; let's lower it to 0.2, so now we'll predict the default class whenever: P(default=Yes|X=x)>0.2.

LDA: Case Study

• The confusion matrix for the *default* data using an LDA with threshold at 0.5:

		True default status		
		No	Yes	Total
Predicted default status	No	9,644	252	9,896
	Yes	23	81	104
	Total	9,667	333	10,000

In this case: **sensitivity** (TP/P)=81/333=24.3% (very low!) ; **specificity** (TN/N)=9,644/9,667=99.8%

• The confusion matrix for the *default* data using an LDA with threshold at 0.2:

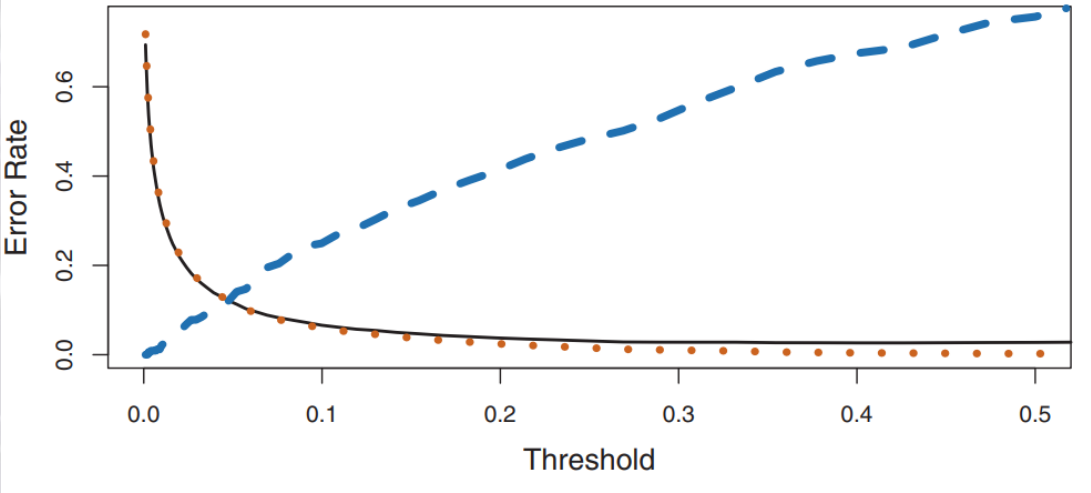
		True default status		
		No	Yes	Total
Predicted default status	No	9,432	138	9,570
	Yes	235	195	430
	Total	9,667	333	10,000

Overall accuracy: 96.27% ; **sensitivity** (TP/P)=58.5%; **specificity** (TN/N)=97.5%

A significant improvement!

LDA: Case Study

• As it turns out, as the threshold is reduced, the error rate among individuals who default decreases steadily, but the error rate among the individuals who do not default increases. In the figure below the black line shows the overall error rate, the blue line shows the fraction of defaulting customers that are incorrectly classified, and the orange line shows the fraction of errors among the non-defaulting customers.



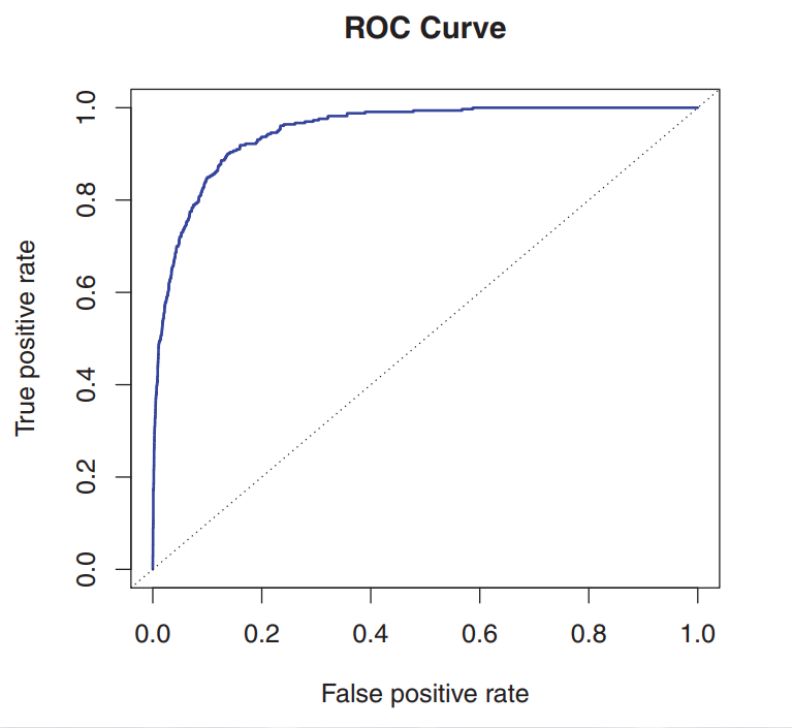
• How can we decide on a best threshold value? Use *domain knowledge*, such as detailed information about the costs associated with default.

LDA: Case Study

• Recall that we can identify the TPR and FPR and formulate the **ROC curve** for a classifier using different thresholding values.

• The **AUC** (*area under the curve*) for the ROC plot provides a metric to compare different classifiers (a large AUC is better, with 1 representing the theoretical maximum and ½ representing a "pure chance" classifier).

• The figure shows the ROC curve for the LDA classifier on the default data; AUC in this case 0.95, which is very strong.



QDA

• We previously mentioned that LDA assumes Gaussian likelihoods with a shared covariance matrix for each class ($\Sigma_c = \Sigma$). In contrast, QDA (quadratic discriminant analysis) lifts the assumption of a shared covariance matrix so that each class has its covariance matrix.

(*) The overall result is that QDA results in a more flexible discriminative model (at the cost of extra computational overhead and potentially larger model variance).

QDA

• We previously mentioned that LDA assumes Gaussian likelihoods with a shared covariance matrix for each class ($\Sigma_c = \Sigma$). In contrast, QDA (quadratic discriminant analysis) lifts the assumption of a shared covariance matrix so that each class has its covariance matrix.

(*) The overall result is that QDA results in a more flexible discriminative model (at the cost of extra computational overhead and potentially larger model variance).

$$p_k(x) = \frac{\pi_k \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left[-\frac{1}{2}(x-\boldsymbol{\mu}_k)^T \Sigma_k (x-\boldsymbol{\mu}_k)\right]}{\sum_{l=1}^{K} \pi_l \frac{1}{(2\pi)^{d/2}|\Sigma_l|^{1/2}} \exp\left[-\frac{1}{2}(x-\boldsymbol{\mu}_l)^T \Sigma_l (x-\boldsymbol{\mu}_l)\right]}$$

(*) As before, taking logs and rearranging terms renders the following expression; QDA assigns a new datum x to the class for which:

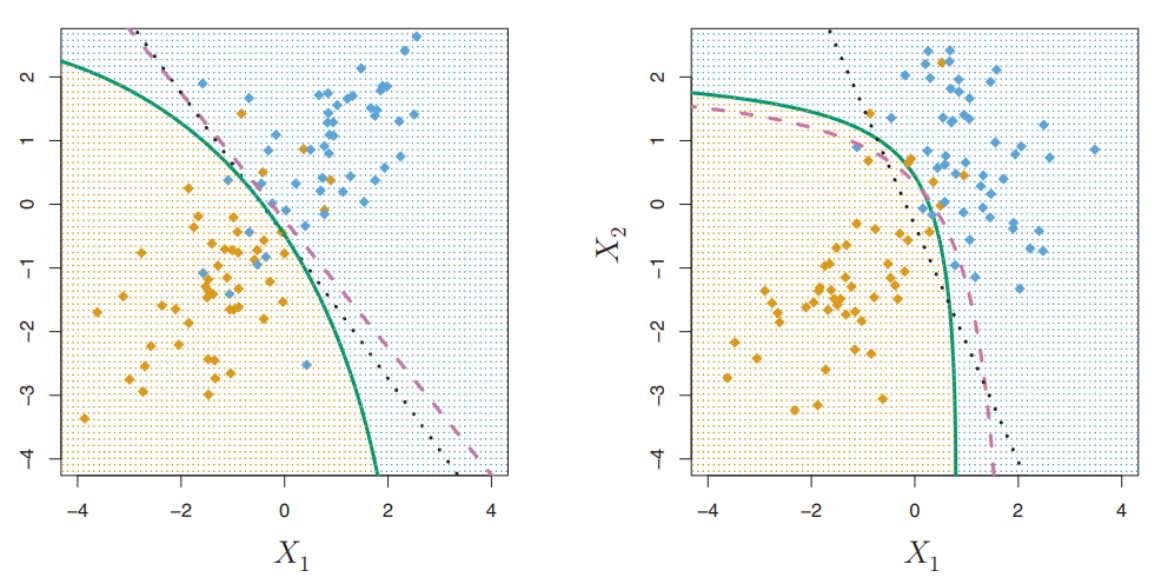
$$\hat{\delta}_k(x) = (\mathbf{x}-\hat{\boldsymbol{\mu}}_k)^T \hat{\Sigma}_k^{-1}(\mathbf{x}-\hat{\boldsymbol{\mu}}_k) - \frac{1}{2}\log|\Sigma_k| + \log(\hat{\pi}_k)$$

is largest. We use the MLE estimates for the parameters in the equation; the decision boundary is parabolic for QDA, as the discriminant function above is quadratic in x.

QDA

• As usual, the bias-variance tradeoff dictates in practice whether it is preferable to use LDA over QDA (or vice versa).

• Notice that with LDA we compute a single, shared covariance matrix that consists of d(d+1)/2 total parameters (why?); whereas for QDA we calculate a separate covariance matrix for each of the *K* classes, for a total of *K*p(p+1)/2 parameters.



• The figure illustrates the performances of LDA and QDA in two scenarios. Left: The Bayes' (ideal) model is in purple; LDA (black); QDA (green); for the 2-class problem shown, the true decision boundary is linear so LDA outperforms QDA; Right: the true decision boundary is non-linear, and the performance of QDA is superior.

Research Applications of LDA/QDA: Emotion Recognition

Kwon, et al., "Emotion Recognition By Speech Signals", EUROSPEECH, 2003.

• The researchers derived models for emotion recognition in audio signals. They selected pitch, log energy, MFCCs features (common audio signal features), etc. The extracted features were analyzed using QDA and SVMs. Experimental results demonstrated that pitch and energy were the most important factors for emotion classification.

• They used SVM, LDA, QDA and HMM models for classification. They were able to achieve 96.3% accuracy for stressed/neural style classification and 70.1% for 4-class speaking style classification for state-of-the-art performance.

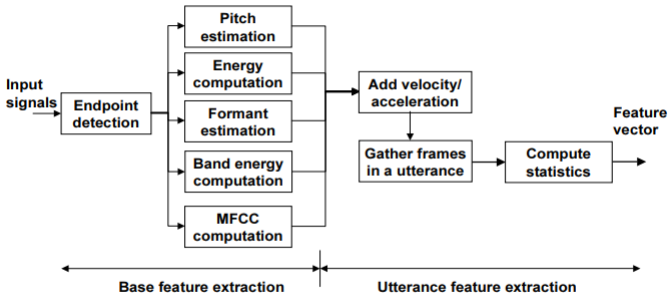


Figure 1: Block diagram of the feature extraction module.

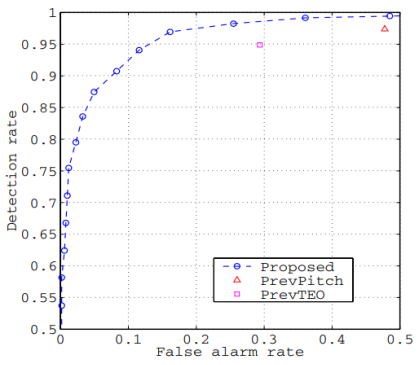


Figure 4: ROC curve of stressed/neutral style classification compared with the previous results.

Table 3: Pairwise classification accuracy (%) using LDA, QDA, GSVM and LSVM

	Angry		Bored		Happy		Neutral		Sad	
A	LD	GS	74	77	63	65	78	74	82	86
	QD	LS	69	76	34	62	73	76	77	82
B	82	81			73	75	65	66	57	55
	79	80			63	75	58	65	49	55
H	62	66	74	76			76	80	81	83
	86	63	76	75			86	80	86	80
N	68	66	56	61	58	56			69	69
	64	68	60	58	32	57			61	65
S	86	86	59	67	78	80	72	74		
	82	85	65	65	69	80	65	76		

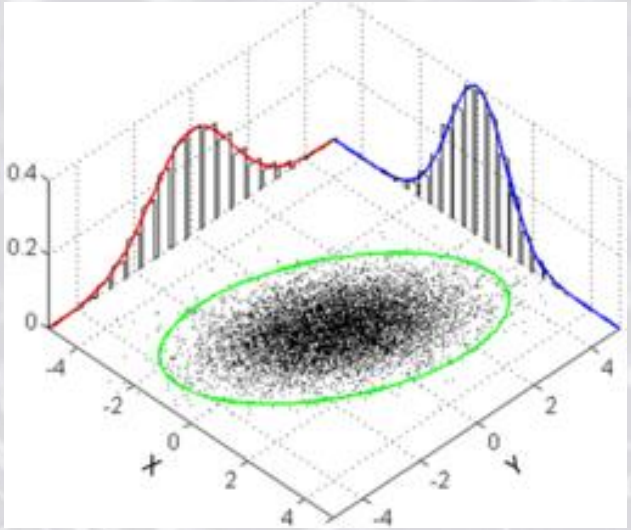
http://www.isca-speech.org/archive/archive_papers/eurospeech_2003/e03_0125.pdf

Marginal & Conditional Gaussians

- Given a join distribution $p(\mathbf{x}_1, \mathbf{x}_2)$ it useful to be able to compute **marginals** $p(\mathbf{x}_1)$ and **conditionals** $p(\mathbf{x}_1 | \mathbf{x}_2)$.

Theorem. Suppose $\mathbf{x}=(\mathbf{x}_1, \mathbf{x}_2)$ is jointly Gaussian with parameters:

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$



Then the marginals are given by:

$$p(\mathbf{x}_1) = N(\mathbf{x}_1 | \boldsymbol{\mu}_1, \Sigma_{11})$$
$$p(\mathbf{x}_2) = N(\mathbf{x}_2 | \boldsymbol{\mu}_2, \Sigma_{22})$$

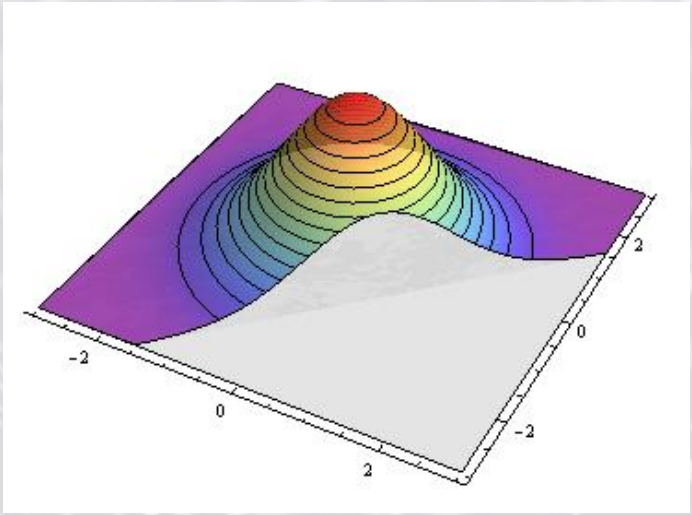
(*) The marginal equations should be intuitively clear – we simply extract the rows and columns corresponding with $\mathbf{x}_1$ and $\mathbf{x}_2$; when we marginalize a Gaussian the distribution remains Gaussian, as one should expect.

Marginal & Conditional Gaussians

- Given a join distribution $p(\mathbf{x}_1, \mathbf{x}_2)$ it useful to be able to compute **marginals** $p(\mathbf{x}_1)$ and **conditionals** $p(\mathbf{x}_1 | \mathbf{x}_2)$.

Theorem. Suppose $\mathbf{x}=(\mathbf{x}_1, \mathbf{x}_2)$ is jointly Gaussian with parameters:

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$



Then the "posterior conditionals" are given by:

$$p(\mathbf{x}_1 | \mathbf{x}_2) = N(\mathbf{x}_1 | \boldsymbol{\mu}_{1|2}, \Sigma_{1|2})$$

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2)$$

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

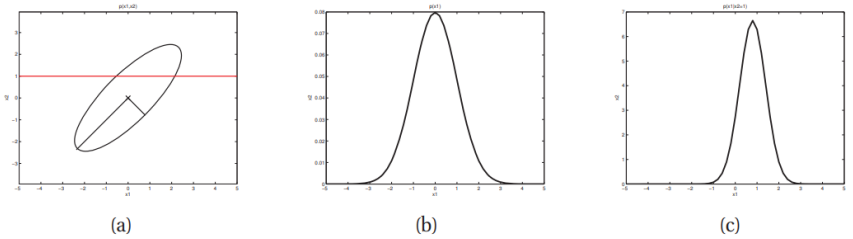
(*) The posterior conditionals show that the conditional mean is just a linear function of $\mathbf{x}2$, and the conditional covariance is just a constant matrix that is independent of x2; when we condition on a Gaussian the result is a Gaussian, as should be intuitive.

Marginal & Conditional Gaussians: Example

- Consider a 2-D example. The covariance matrix is given by:

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}$$

where ρ is a non-negative parameter.



(a) (b) (c)

Figure 4.9 (a) A joint Gaussian distribution $p(x_1, x_2)$ with a correlation coefficient of 0.8. We plot the 95% contour and the principal axes. (b) The unconditional marginal $p(x_1)$. (c) The conditional $p(x_1|x_2) = \mathcal{N}(x_1|0.8, 0.36)$, obtained by slicing (a) at height $x_2 = 1$. Figure generated by `gaussCondition2Ddemo2`.

- The marginal $p(x_1)$ is a 1-D Gaussian, obtained by projecting the joint distribution onto the x1 line:

$$p(x_1) = N(x_1 \mid \mu_1, \sigma_1)$$

Marginal & Conditional Gaussians: Example

- Consider a 2-D example. The covariance matrix is given by:

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}$$

where ρ is a non-negative parameter.

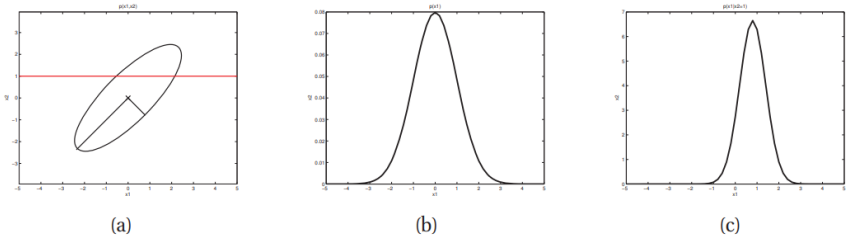


Figure 4.9 (a) A joint Gaussian distribution $p(x_1, x_2)$ with a correlation coefficient of 0.8. We plot the 95% contour and the principal axes. (b) The unconditional marginal $p(x_1)$. (c) The conditional $p(x_1|x_2) = \mathcal{N}(x_1|0.8, 0.36)$, obtained by slicing (a) at height $x_2 = 1$. Figure generated by `gaussCondition2Ddemo2`.

- The marginal $p(x_1)$ is a 1-D Gaussian, obtained by projecting the joint distribution onto the x1 line:

$$p(x_1) = N(x_1 \mid \mu_1, \sigma_1)$$

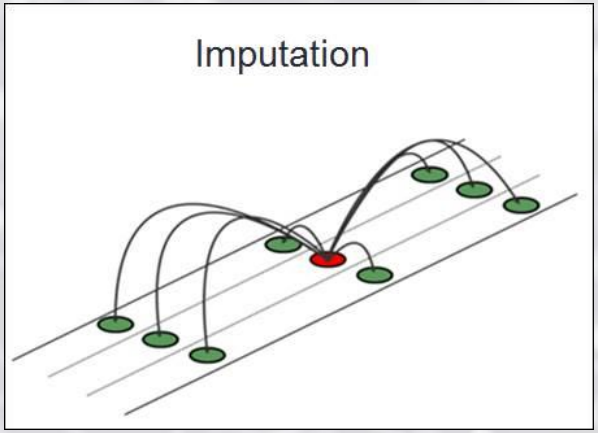
- Suppose we observe $X_2 = x_2$; the conditional $p(x_1 \mid x_2)$ is obtained by "slicing" the joint distribution through the $X_2 = x_2$ line:

$$p(x_1 \mid x_2) = N\left(x_1 \mid \mu_1 + \frac{\rho\sigma_1\sigma_2}{\sigma_2^2}(x_2 - \mu_2), \sigma_1^2 - \frac{(\rho\sigma_1\sigma_2)^2}{\sigma_2^2}\right)$$

$$p(\mathbf{x}_1 \mid \mathbf{x}_2) = N(\mathbf{x}_1 \mid \boldsymbol{\mu}_{1|2}, \Sigma_{1|2})$$
$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2)$$
$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Research Application Example: MVNI (Multi-variate Normal Imputation)

Shafer, et al., "Multiple imputation for multivariate missing-data problems: a data analyst's perspective", 1998 (1400+ citations).



(*) Missing/corrupted data are an inevitable part of data science – particularly in medicine/epidemiology. **Data Imputation** is the process of (cogently) replacing missing elements in a data set.

Shafer et al., devised the **MVNI** (*multi-variate normal imputation*) method which assumes all variables in the model have a joint MVN distribution (after applying a data transformation/pre-processing steps such as the *log transform*). Then by using the MVN conditioning equations, imputation is performed.

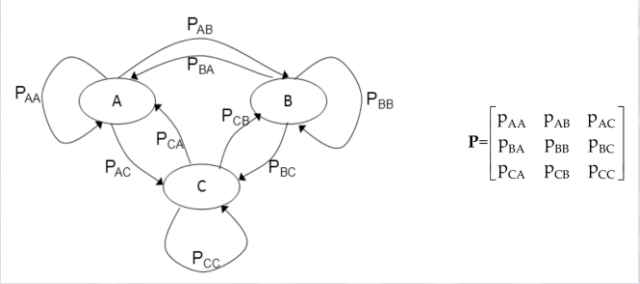
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.120.6337&rep=rep1&type=pdf

Research Application (Aside): Gibbs Sampling – how do I sample from a MVN?

• **MCMC** (*Markov Chain Monte Carlo*) is a sampling method that allows one – in principle – to approximate a complex integral (oftentimes the answer to an interesting ML/data science problem naturally involves complex integration, e.g. calculating the mean/expected value of a model distribution)

• Recall that a **Markov Chain** is a sequential model that transitions from one state to another in a probabilistic fashion, where the next state that the chain takes is conditioned on the previous state(s). Markov Chains are useful in that if they are constructed properly, and allowed to run for a sufficiently long time, the states that a chain will take correspond with samples from a specified target probability distribution.

• Thus we can construct Markov chains to sample from the distribution whose integral we would like to approximate.



Manhattan Project

Research Application (Aside): Gibbs Sampling – how do I sample from a MVN?

• The **Gibbs Sampler** is an MCMC method that generates samples from a target joint distribution $p(x_1,x_2,\ldots,x_D)$ by first initializing each variable in the model (e.g. randomly) and then repeatedly sampling from "full conditionals" for each variable in the model, i.e. $p(x_i \mid x_1,x_2,\ldots,x_{i-1},x_{i+1},\ldots x_D)$.

Example: Consider the task of sampling from a Bivariate Gaussian

$$p(\mathbf{x}) = N(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma) \quad \text{with parameters} \quad \boldsymbol{\mu} = [\mu_1, \mu_2] = [0,0], \; \Sigma = \begin{bmatrix} 1 & \rho_{12} \\ \rho_{21} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

In order to sample from this distribution using the Gibbs sampler, we need the conditional distribution formulas:

$$p\left(x_1 \mid x_2^{(t-1)}\right) = N\left(\mathbf{x} \mid \mu_1 + \rho_{21}\left(x_2^{(t-1)} - \mu_2\right), \sqrt{1-\rho_{21}^2}\right)$$

$$p\left(x_2 \mid x_1^{(t-1)}\right) = N\left(\mathbf{x} \mid \mu_2 + \rho_{11}\left(x_1^{(t-1)} - \mu_1\right), \sqrt{1-\rho_{12}^2}\right)$$

$$p(\mathbf{x}_1 \mid \mathbf{x}_2) = N(\mathbf{x}_1 \mid \boldsymbol{\mu}_{1|2}, \Sigma_{1|2})$$

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2)$$

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Research Application (Aside): Gibbs Sampling – how do I sample from a MVN?

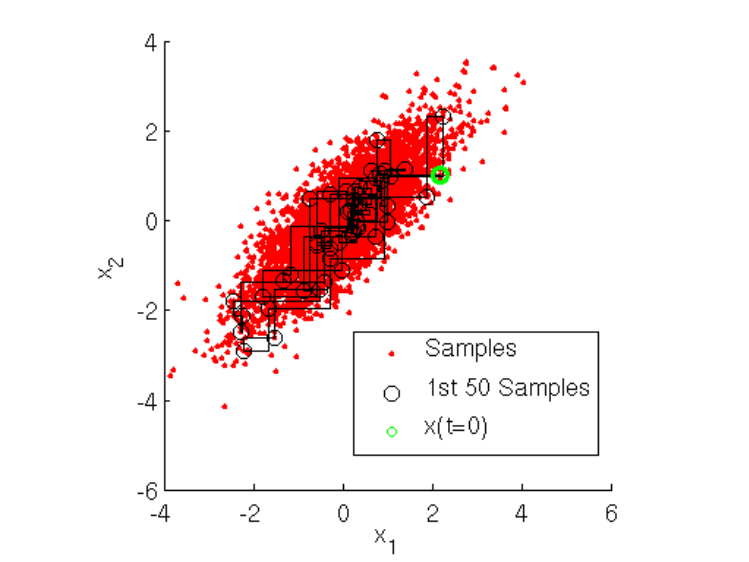
Example: Consider the task of sampling from a Bivariate Gaussian

$$p(\mathbf{x}) = N(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma) \text{ with parameters } \boldsymbol{\mu} = [\mu_1, \mu_2] = [0, 0], \ \Sigma = \begin{bmatrix} 1 & \rho_{12} \\ \rho_{21} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

$$p\left(x_1 \mid x_2^{(t-1)}\right) = N\left(\mathbf{x} \mid \mu_1 + \rho_{21}\left(x_2^{(t-1)} - \mu_2\right), \sqrt{1-\rho_{21}^2}\right)$$

$$p\left(x_2 \mid x_1^{(t-1)}\right) = N\left(\mathbf{x} \mid \mu_2 + \rho_{11}\left(x_1^{(t-1)} - \mu_1\right), \sqrt{1-\rho_{12}^2}\right)$$

The figure shows the results of applying Gibbs sampling; at each step the Gibbs sampler takes a step only in the x_1 direction, then only in the x_2 direction.



Conditional Gaussians: Derivation

• We've seen several examples of the utility of the Gaussian conditional equations. Now we consider their derivation.

Recall the main result, which states: Suppose $\mathbf{x}=(\mathbf{x}_1, \mathbf{x}_2)$ is jointly Gaussian with parameters:

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \; \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$

Then the "posterior conditionals" are given by:

$$p(\mathbf{x}_1 \mid \mathbf{x}_2) = N(\mathbf{x}_1 \mid \boldsymbol{\mu}_{1|2}, \Sigma_{1|2})$$

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2)$$

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

(*) Before discussing the main step in the derivation, we first need an important result from linear algebra known as the **Matrix Inversion Lemma**.

Matrix Inversion Lemma

- **The Matrix Inversion Lemma** states: For a general partitioned matrix

$$M = \begin{pmatrix} E & F \\ G & H \end{pmatrix}$$

The inverse is given by:

$$M^{-1} = \begin{pmatrix} E^{-1} + E^{-1}F\left(M/E\right)GE^{-1} & -E^{-1}F\left(M/E\right)^{-1} \\ -\left(M/E\right)^{-1}GE^{-1} & \left(M/E\right)^{-1} \end{pmatrix}$$

Where the "**Schur complements**" are defined:

$$M/H = E - FH^{-1}G$$

$$M/E = H - GE^{-1}F$$

(*) While this result looks quite complicated, the proof utilizes a single trick of block diagaonilizing the *M* matrix to make the inversion cleaner.

Conditional Gaussians: Derivation

- Using the matrix inversion lemma, we can now derive the conditional Gaussian equations.

We begin by factoring the joint $p(\mathbf{x}_1,\mathbf{x}_2)$ as $p(\mathbf{x}_2)p(\mathbf{x}_1|\mathbf{x}_2)$ (which holds by the multiplication rule of probability):

Let
$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

Conditional Gaussians: Derivation

- Using the matrix inversion lemma, we can now derive the conditional Gaussian equations.

We begin by factoring the joint $p(\mathbf{x}_1,\mathbf{x}_2)$ as $p(\mathbf{x}_2)p(\mathbf{x}_1|\mathbf{x}_2)$ (which holds by the multiplication rule of probability):

Let $$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}(\Sigma/\Sigma_{22})^{-1} & -\Sigma_{12}\Sigma_{22}^{-1}\\ 0 & \Sigma_{22}^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

Follows from matrix inversion lemma

$$M^{-1} = \begin{pmatrix}E^{-1}+E^{-1}F(M/E)GE^{-1} & -E^{-1}F(M/E)^{-1}\\ -(M/E)^{-1}GE^{-1} & (M/E)^{-1}\end{pmatrix}$$

Conditional Gaussians: Derivation

- Using the matrix inversion lemma, we can now derive the conditional Gaussian equations.

We begin by factoring the joint $p(\mathbf{x}_1,\mathbf{x}_2)$ as $p(\mathbf{x}_2)p(\mathbf{x}_1|\mathbf{x}_2)$ (which holds by the multiplication rule of probability):

Let $$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}(\Sigma/\Sigma_{22})^{-1} & -\Sigma_{12}\Sigma_{22}^{-1}\\ 0 & \Sigma_{22}^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

Follows from matrix inversion lemma

$$M^{-1} = \begin{pmatrix}E^{-1}+E^{-1}F(M/E)GE^{-1} & -E^{-1}F(M/E)^{-1}\\ -(M/E)^{-1}GE^{-1} & (M/E)^{-1}\end{pmatrix}$$

Conditional Gaussians: Derivation

- Using the matrix inversion lemma, we can now derive the conditional Gaussian equations.

We begin by factoring the joint $p(\mathbf{x}_1,\mathbf{x}_2)$ as $p(\mathbf{x}_2)p(\mathbf{x}_1|\mathbf{x}_2)$ (which holds by the multiplication rule of probability):

$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}(\Sigma/\Sigma_{22})^{-1} & -\Sigma_{12}\Sigma_{22}^{-1}\\ 0 & \Sigma_{22}^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)^T(\Sigma/\Sigma_{22})^{-1}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)\right\}\times\exp\left\{-\frac{1}{2}(\mathbf{x}_2-\boldsymbol{\mu}_2)^T\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right\}$$

(*) **<u>This is an expression of the form</u>**: exp(quadratic form in x_1, x_2) x exp(quadratic form in x_2).

Conditional Gaussians: Derivation

- Using the matrix inversion lemma, we can now derive the conditional Gaussian equations.

We begin by factoring the joint $p(\mathbf{x}_1,\mathbf{x}_2)$ as $p(\mathbf{x}_2)p(\mathbf{x}_1|\mathbf{x}_2)$ (which holds by the multiplication rule of probability):

$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}(\Sigma/\Sigma_{22})^{-1} & -\Sigma_{12}\Sigma_{22}^{-1}\\ 0 & \Sigma_{22}^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)^T(\Sigma/\Sigma_{22})^{-1}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)\right\}\times\exp\left\{-\frac{1}{2}(\mathbf{x}_2-\boldsymbol{\mu}_2)^T\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right\}$$

(*) **<u>This is an expression of the form</u>**: exp(quadratic form in x_1, x_2) x exp(quadratic form in x_2).

Thus we have factored the joint as:

$$p(\mathbf{x}_1,\mathbf{x}_2) = p(\mathbf{x}_1|\mathbf{x}_2)\,p(\mathbf{x}_2)$$
$$= N(\mathbf{x}_1|\boldsymbol{\mu}_{1|2},\Sigma_{1|2})\,N(\mathbf{x}_2|\boldsymbol{\mu}_2,\Sigma_{22})$$

Conditional Gaussians: Derivation

• Using the matrix inversion lemma, we can now derive the conditional Gaussian equations.

We begin by factoring the joint $p(\mathbf{x}_1,\mathbf{x}_2)$ as $p(\mathbf{x}_2)p(\mathbf{x}_1|\mathbf{x}_2)$ (which holds by the multiplication rule of probability):

$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}(\Sigma/\Sigma_{22})^{-1} & -\Sigma_{12}\Sigma_{22}^{-1}\\ 0 & \Sigma_{22}^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)^T(\Sigma/\Sigma_{22})^{-1}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)\right\}\times\exp\left\{-\frac{1}{2}(\mathbf{x}_2-\boldsymbol{\mu}_2)^T\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right\}$$

(*) **<u>This is an expression of the form</u>**: exp(quadratic form in x_1, x_2) x exp(quadratic form in x_2).

Thus we have factored the joint as:

$$p(\mathbf{x}_1,\mathbf{x}_2) = p(\mathbf{x}_1|\mathbf{x}_2)\,p(\mathbf{x}_2)$$
$$= N(\mathbf{x}_1|\boldsymbol{\mu}_{1|2},\Sigma_{1|2})\,N(\mathbf{x}_2|\boldsymbol{\mu}_2,\Sigma_{22})$$

So what are the relevant equations for $\mu_{1|2}$ and $\Sigma_{1|2}$, i.e., the Gaussian conditioning parameter formulae?

Conditional Gaussians: Derivation

$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}(\Sigma/\Sigma_{22})^{-1} & -\Sigma_{12}\Sigma_{22}^{-1}\\ 0 & \Sigma_{22}^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)^T(\Sigma/\Sigma_{22})^{-1}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)\right\}\times\exp\left\{-\frac{1}{2}(\mathbf{x}_2-\boldsymbol{\mu}_2)^T\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right\}$$

Thus we have factored the joint as:

$$p(\mathbf{x}_1,\mathbf{x}_2) = p(\mathbf{x}_1\mid\mathbf{x}_2)\,p(\mathbf{x}_2)$$
$$= N(\mathbf{x}_1\mid\boldsymbol{\mu}_{1|2},\Sigma_{1|2})\,N(\mathbf{x}_2\mid\boldsymbol{\mu}_2,\Sigma_{22})$$

So what are the relevant equations for $\mu_{1|2}$ and $\Sigma_{1|2}$, i.e., the Gaussian conditioning parameter formulae?

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)$$
$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

$$p(\mathbf{x}_1\mid\mathbf{x}_2) = N(\mathbf{x}_1\mid\boldsymbol{\mu}_{1|2},\Sigma_{1|2})$$
$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)$$
$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Conditional Gaussians: Derivation

$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}I & 0\\ -\Sigma_{22}^{-1}\Sigma_{21} & I\end{pmatrix}\begin{pmatrix}(\Sigma/\Sigma_{22})^{-1} & -\Sigma_{12}\Sigma_{22}^{-1}\\ 0 & \Sigma_{22}^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\}$$

$$= \exp\left\{-\frac{1}{2}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)^T(\Sigma/\Sigma_{22})^{-1}\left(\mathbf{x}_1-\boldsymbol{\mu}_1-\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right)\right\}\times\exp\left\{-\frac{1}{2}(\mathbf{x}_2-\boldsymbol{\mu}_2)^T\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right\}$$

Thus we have factored the joint as:

$$p(\mathbf{x}_1,\mathbf{x}_2) = p(\mathbf{x}_1\mid\mathbf{x}_2)\,p(\mathbf{x}_2)$$
$$= N(\mathbf{x}_1\mid\boldsymbol{\mu}_{1|2},\Sigma_{1|2})\,N(\mathbf{x}_2\mid\boldsymbol{\mu}_2,\Sigma_{22})$$

So what are the relevant equations for $\mu_{1|2}$ and $\Sigma_{1|2}$, i.e., the Gaussian conditioning parameter formulae?

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)$$
$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

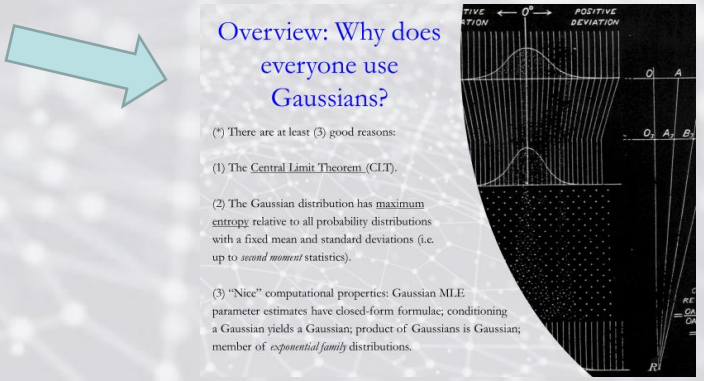
$\Leftrightarrow$

$$p(\mathbf{x}_1\mid\mathbf{x}_2) = N(\mathbf{x}_1\mid\boldsymbol{\mu}_{1|2},\Sigma_{1|2})$$
$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)$$
$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

(*) This is precisely what we wanted to show in order to derive the Gaussian conditioning equations! QED

Maximum Entropy Distributions

• We previously noted that there are (at least) three good reasons for the ubiquity of Gaussian distributions in practice:



(1) CLT

(2) "Nice" computational properties, including: Gaussian MLE parameter estimates have closed form solutions; conditioning equations are readily computable, etc.

(3) The Gaussian distribution has maximum entropy relative to all probability distributions with a fixed mean and standard deviation (i.e. up to second moment statistics).

(*) To date, we expounded on conditions (1) and (2); now we delve into (3).

Maximum Entropy Distributions

• From our prior information theory lecture, we introduced the definition and intuitive meaning of (Shannon) *entropy*:

The **entropy** of a random variable X with distribution *p*, denoted by *H*(X) or sometimes *H*(p) is a measure of surprise/uncertainty. In particular, for a discrete random variable with *K* states, it is defined:

$$H(X) = -\sum_{k=1}^{K} p(X=k)\log_2\left(p(X=k)\right)$$

Maximum Entropy Distributions

• From our prior information theory lecture, we introduced the definition and intuitive meaning of (Shannon) *entropy*:

(*) The **entropy** of a random variable X with distribution *p*, denoted by *H*(X) or sometimes *H*(p) is a measure of surprise/uncertainty. In particular, for a discrete random variable with *K* states, it is defined:

$$H(X) = -\sum_{k=1}^{K} p(X=k)\log_2\left(p(X=k)\right)$$

• For a *continuous random variable* X with density function *p*(x), the entropy is defined analogously (note that is it often referred to as **differential entropy** in this context):

$$H(X) = -\int_X p(x)\log p(x)\,dx$$

Maximum Entropy Distributions

• As a general, guiding scientific principle, Laplace offered the *principle of insufficient reason* (PRI) which asserts that (later codified by the economist Keynes) :

"equivalent states of knowledge should be assigned equivalent epistemic probabilities" – to paraphrase: where there is no reason to presume otherwise, our prior beliefs should be encoded as uninformative (i.e. maximally entropic).

(*) Remember that we previously proved (using the information inequality) that the uniform distribution is maximally entropic (this should be intuitive); next we demonstrate that the Gaussian is also maximally entropic under some basic conditions.

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** for all continuous distributions with fixed mean (finite) and variance.

• There are several ways to prove this result – we note that the *information inequality* can be used to directly demonstrate this property (as in vein of our earlier proof that the uniform distribution is maximally entropic). In the interest of using a diverse array of techniques, we'll opt instead to use the method of **Langrange multipliers**.

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** for all continuous distributions with fixed mean (finite) and variance.

• There are several ways to prove this result – we note that the *information inequality* can be used to directly demonstrate this property (as in vein of our earlier proof that the uniform distribution is maximally entropic). In the interest of using a diverse array of techniques, we'll opt instead to use the method of **Langrange multipliers**.

• We reiterate that the classical method of Lagrange multipliers (LM) is applicable for optimization problems with equality constraints. The essence of the LM method is that a sufficient condition for the solution to a constrained optimization problem is that, the gradient of the *Lagrangian* equals zero, i.e.:

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

(*) $f(\mathbf{x})$ is the **objective function** (i.e. the function we want to optimize) and the $g_i(\mathbf{x})$ functions are constraint equations.

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** (in 1-D) for all continuous distributions with fixed mean (finite) and variance.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

(*) Let's now organize the objective function and constraints for the maximum entropy problem.

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** (in 1-D) for all continuous distributions with fixed mean (finite) and variance.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

(*) Let's now organize the objective function and constraints for the maximum entropy problem.

Objective function: $f(p) = -\int_X p(x) \log p(x) dx$ (we express the objective wrt p, a probability distribution).

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** (in 1-D) for all continuous distributions with fixed mean (finite) and variance.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

(*) Let's now organize the objective function and constraints for the maximum entropy problem.

Objective function: $f(p) = -\int_X p(x) \log p(x) dx$ (we express the objective wrt *p*, a probability distribution).

Constraint equations: (1) $g_1(p): \int_X p(x) dx = 1$ (why?)

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** (in 1-D) for all continuous distributions with fixed mean: μ (finite) and variance: σ^2.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

(*) Let's now organize the objective function and constraints for the maximum entropy problem.

Objective function: $f(p) = -\int_X p(x) \log p(x) dx$ (we express the objective wrt p, a probability distribution).

Constraint equations: (1) $g_1(p): \int_X p(x) dx = 1$

(2) $g_2(p): \int_X x p(x) dx = \mu$ (why?)

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** (in 1-D) for all continuous distributions with fixed mean: μ (finite) and variance: σ^2.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

(*) Let's now organize the objective function and constraints for the maximum entropy problem.

Objective function: $f(p) = -\int_X p(x) \log p(x) dx$ (we express the objective wrt p, a probability distribution).

Constraint equations: (1) $g_1(p): \int_X p(x) dx = 1$

(2) $g_2(p): \int_X x p(x) dx = \mu$ (3) $g_3(p): \int_X x^2 p(x) dx = \sigma^2$ (why?)

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** (in 1-D) for all continuous distributions with fixed mean: μ (finite) and variance: σ^2.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

$$f(p) = -\int_X p(x) \log p(x) dx$$

$$g_1(p): \int_X p(x) dx = 1 \qquad g_2(p): \int_X x p(x) dx = \mu$$

$$g_3(p): \int_X x^2 p(x) dx = \sigma^2$$

- Now define the *Lagrangian*:

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** for all continuous distributions with fixed mean: μ (finite) and variance: σ^2.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

$$f(p) = -\int_X p(x)\log p(x)\,dx$$

$$g_1(p): \int_X p(x)dx = 1 \qquad g_2(p): \int_X xp(x)dx = \mu$$

$$g_3(p): \int_X x^2 p(x)dx = \sigma^2$$

- Now define the *Lagrangian*:

$$L(x,\lambda) = -\int_X p(x)\log p(x)\,dx + \lambda_0\left(\int_X p(x)dx - 1\right) + \lambda_1\left(\int_X xp(x)dx - \mu\right) + \lambda_2\left(\int_X x^2 p(x)dx - \sigma^2\right)$$

Maximum Entropy of the Gaussian Distribution

(*) Proof that the **Gaussian distribution has maximum entropy** (in 1-D) for all continuous distributions with fixed mean: μ (finite) and variance: σ^2.

$$\nabla_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = 0, \text{ where } L(\mathbf{x},\lambda) = f(\mathbf{x}) - \sum_i \lambda_i g_i(\mathbf{x})$$

$$f(p) = -\int_X p(x)\log p(x)\,dx \qquad g_1(p): \int_X p(x)dx = 1 \qquad g_2(p): \int_X xp(x)dx = \mu$$

$$g_3(p): \int_X x^2 p(x)dx = \sigma^2$$

- Now define the *Lagrangian*:

$$L(x,\lambda) = -\int_X p(x)\log p(x)\,dx + \lambda_0\left(\int_X p(x)dx - 1\right) + \lambda_1\left(\int_X xp(x)dx - \mu\right) + \lambda_2\left(\int_X x^2 p(x)dx - \sigma^2\right)$$

Set $\nabla_{x,\lambda,p} L(x,\lambda) = 0$ and solve…

This yields: $p(x) = e^{(\lambda_0 - 1) + \lambda_1 x + \lambda_2 x^2} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$, as was to be shown.

Gaussian Processes: Introduction

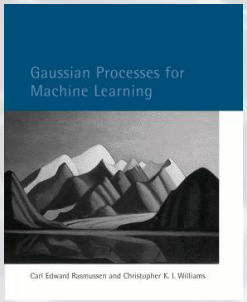
• Previously we have seen many useful variants of regression models: linear regression, regularized regression, linear basis function models (e.g. radial basis regression), and logistic regression.

(*) In each case, these models were both parametric (i.e. the number of parameters was fixed) and non-probabilistic (i.e. non-Bayesian) – at least in our earlier treatments.

• The task of regression is inherently an **inductive problem**: we wish to generate (continuous) predictors for new inputs $\mathbf{x}_*$ (over all possible input values), given a data set D of *n* observations: D=$\{(\boldsymbol{x}_i, yi)|i = 1, \ldots, n\}$.

• In order to devise such a function, we must make some basic assumptions about the characteristics of the underlying function, as otherwise any function which is consistent with the training data would be equally valid – and this would yield an **ill-posed problem**.

Recommended text on Gaussian Processes:
Rasmussen et al., *Gaussian Processes for Machine Learning*, MIT (2006).



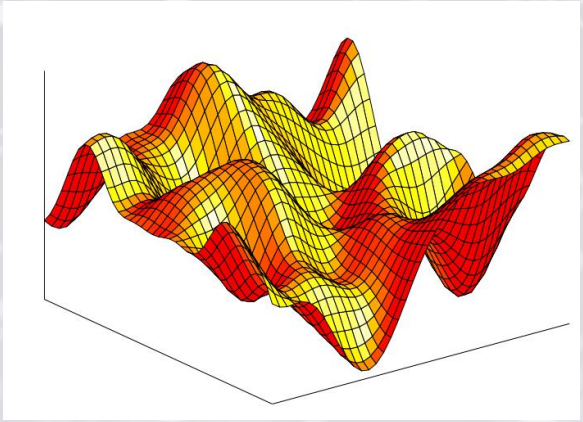
Gaussian Processes: Introduction

• In determining the characteristics of the underlying function there are (2) common approaches:

(1) Restrict the class of possible functions that we consider (e.g., linear functions, quadratic functions, etc.)

(2) Use an optimization process to search over different models as well as the parameters of the model. Give a **prior probability** (in some generic sense) to every possible function, where higher probabilities are given to functions that we consider to be more likely – such as *smooth functions*.

(*) The second approach admits of a Bayesian scheme and is generally more flexible, as we subsequently demonstrate.



Gaussian Processes: Introduction

• To optimize over different models and parameters in conjunction with a prior, we need to generalize the idea of a probability distribution to something over which we can optimize.

• A **stochastic process** is a collection of random variables put together: instead of having a set of parameters that specify a probability distribution (e.g. mean and covariance), we have asset of functions that a distribution over that set of functions.

•Dealing with a general stochastic process is often difficult because combining the random variables is generally hard.

•However, if we restrict the process in such a way that all of the random variables have a Gaussian distribution, and the joint distribution over any (finite) subset of the variables is also Gaussian, then this is a **Gaussian Process** (GP), and is much easier to deal with.

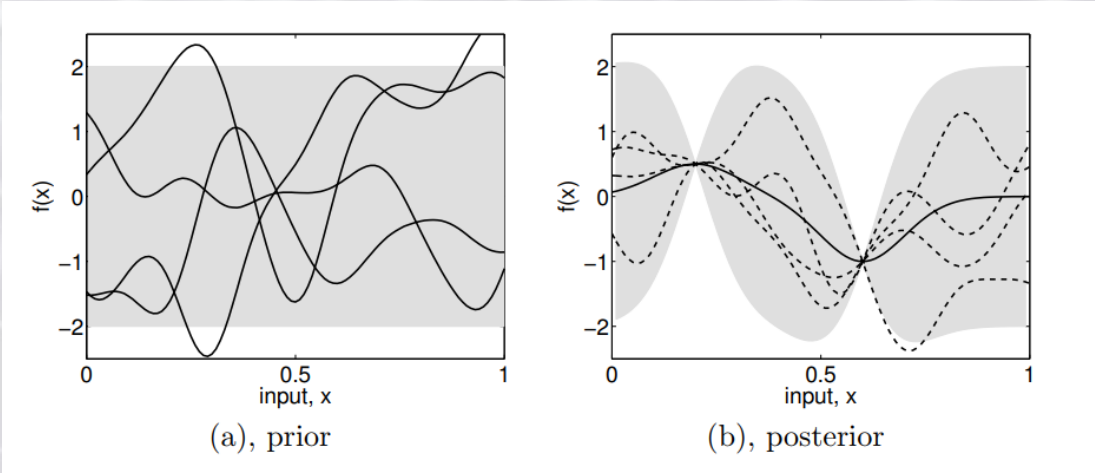
(*) Some consider GPR (Gaussian Process Regression) as regression model "for the 21st century."

https://arxiv.org/pdf/1309.7837.pdf

Gaussian Processes: A Pictorial Introduction to Bayesian Modelling

• Consider a simple 1-D regression problem, mapping from an input *x* to an output *f*(x).

• In the figure, on the left we show a number of sample functions drawn at random from the prior distribution over functions specified by a particular Gaussian Process which favors smooth functions.



(a), prior (b), posterior

• The prior is taken to represent our prior beliefs over the kinds of functions we expect to observe, before seeing any data. In the absence of knowledge to the contrary, we have assumed the average value over the sample functions at each *x* is zero. The shaded region denotes twice the pointwise standard deviation.

• Given a data set *D*, the dashed lines in the right image show sample functions which are consistent with *D*, and the solid line depicts the mean value of such functions. The combination of the prior and the data leads to the **posterior distribution over functions**.

Gaussian Processes

• The way to think about modelling with GPs is that we put a probability distribution over the space of functions (possibly an infinite dimensional space!) and sample from that.

• Ostensibly, if we wanted to specify $f(\mathbf{x})$ for any input vector $\mathbf{x}$, we could just list the value of f for every possible value of $\mathbf{x}$ – but this would require an infinite number of point-wise specifications.

• However, by the Gaussian assumption intrinsic to GPs (namely: any subset of the random variables is jointly Gaussian) we only need to specify a mean and covariance matrix to fully specify the GP (just like a Gaussian density is fully specified by its mean and covariance).

• The mean and covariance specification implies a distribution over functions. We can then perform *posterior inference* (i.e. generate $f(\mathbf{x}_*)$) for any test input vector $\mathbf{x}_*$, given a training set D.

(*) In summary, **GPs are just smoothers**, meaning that they fit a smooth curve to a set of data points (note that GPs can also be used in a similar way to perform classification tasks).

Gaussian Processes

- GPs are specified by *mean* and *covariance functions* (in practice the data are centered so that we work with a zero mean GP for convenience). In this case, the GP is completely described as a function G(k(**x**,**x**')) that models some underlying function *f*(**x**), where the covariance function k(**x**,**x**') gives us the expected covariance matrix between the values of *f* at **x** and **x**'.

(*) For a GP, the prior is encoded *via* the choice of covariance function.

Gaussian Processes

• GPs are specified by *mean* and *covariance functions* (in practice the data are centered so that we work with a zero mean GP for convenience). In this case, the GP is completely described as a function G(k(**x**,**x**')) that models some underlying function $f(\mathbf{x})$, where the covariance function k(**x**,**x**') gives us the expected covariance matrix between the values of f at **x** and **x**'.

(*) For a GP, the prior is encoded via the choice of covariance function.

• There are many legitimate choices for covariance functions to be used in conjunction with GP. We'll focus on the most common choice, the **square exponential** (SE) covariance function (also called a radial basis kernel (RBF):

$$k_{SE}\left(\mathbf{x},\mathbf{x}'\right)=\sigma_f^2\exp\left(-\frac{1}{2l^2}\left\|\mathbf{x}-\mathbf{x}'\right\|^2\right)$$

where the **length scale** (l) and **signal variance** (σ_f^2) are hyperparameters (we analyze their effects on the GP subsequently). For a set of input vectors the covariance function enables us to specify a matrix of covariances **K** where the element at place (i, j) in the matrix is K_{ij} = $k(\mathbf{x}^{(i)},\mathbf{x}^{(j)})$.

Gaussian Processes

- That's a lot to take in. Let's review the salient points:

A **Gaussian Process** is a collection of random variables, any finite number of which have a joint Gaussian distribution.

A GP is completely specified by its mean function and covariance function. In full generality, we define the mean function m(**x**) and the covariance function k(**x**, **x**') of a stochastic process *f*(**x**) as:

$$m(\mathbf{x}) = E\left[f(\mathbf{x}) \right]$$

$$k(\mathbf{x}, \mathbf{x}') = E\left[\left(f(\mathbf{x}) - m(\mathbf{x}) \right) \left(f(\mathbf{x}') - m(\mathbf{x}') \right) \right]$$

Gaussian Processes

- That's a lot to take in. Let's review the salient points:

A **Gaussian Process** is a collection of random variables, any finite number of which have a joint Gaussian distribution.

A GP is completely specified by its mean function and covariance function. In full generality, we define the mean function m(**x**) and the covariance function k(**x**, **x**') of a stochastic process *f*(**x**) as:

$$m(\mathbf{x}) = E\left[f(\mathbf{x}) \right]$$

$$k(\mathbf{x}, \mathbf{x}') = E\left[\left(f(\mathbf{x}) - m(\mathbf{x}) \right)\left(f(\mathbf{x}') - m(\mathbf{x}') \right) \right]$$

For simplicity, we'll set m(**x**)=0 (that is, we center the data) and use the square exponential kernel, k_{SE}. Finally, we write the Gaussian Process as:

$$f(\mathbf{x}) \sim GP\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}') \right)$$

This notation explicitly conveys the fact that **GPs describe a distribution over functions**.

Gaussian Processes: Prediction with Noise-free Observations

• Consider noise-free data $D=\{(\boldsymbol{x}_i, f_i)|i = 1, \ldots, n\}$. The joint distribution of the training outputs, $\mathbf{f}$, and the test outputs $\mathbf{f}_*$ is given by:

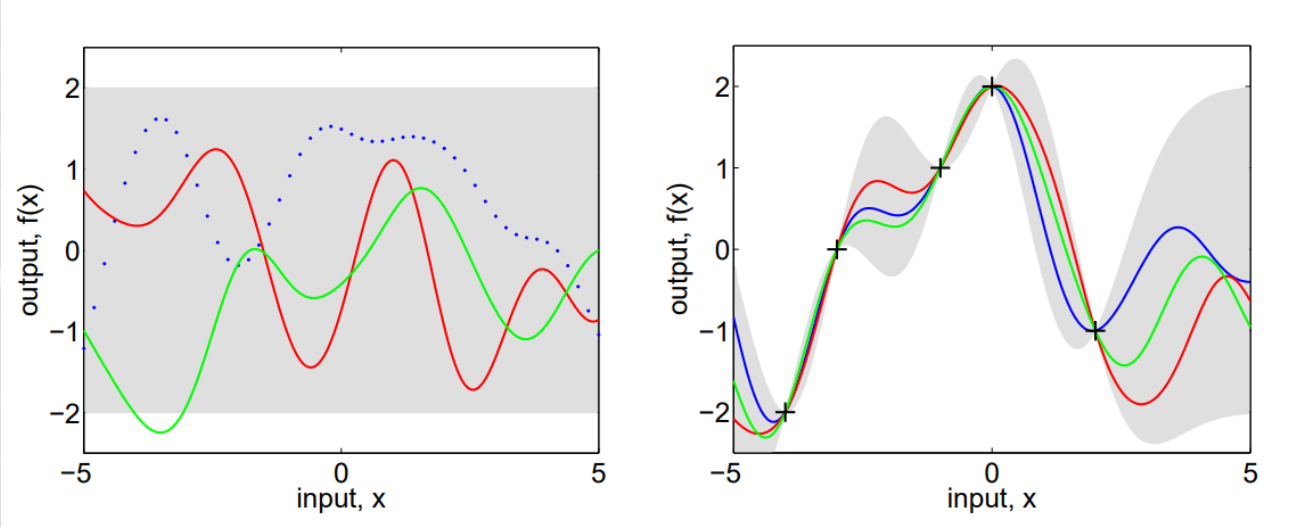
$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right)$$

• If there are n training points and n_* test points, then $K(X,X_*)$ denotes the n x n_* matrix of the covariances evaluated at all pairs of training and test points, and similarly for the other entries: $K(X, X)$, $K(X_*, X_*)$ and $K(X_*, X)$.

• To get the posterior distribution over functions, we need to restrict this joint prior distribution to contain only those functions which agree with the observed data points.

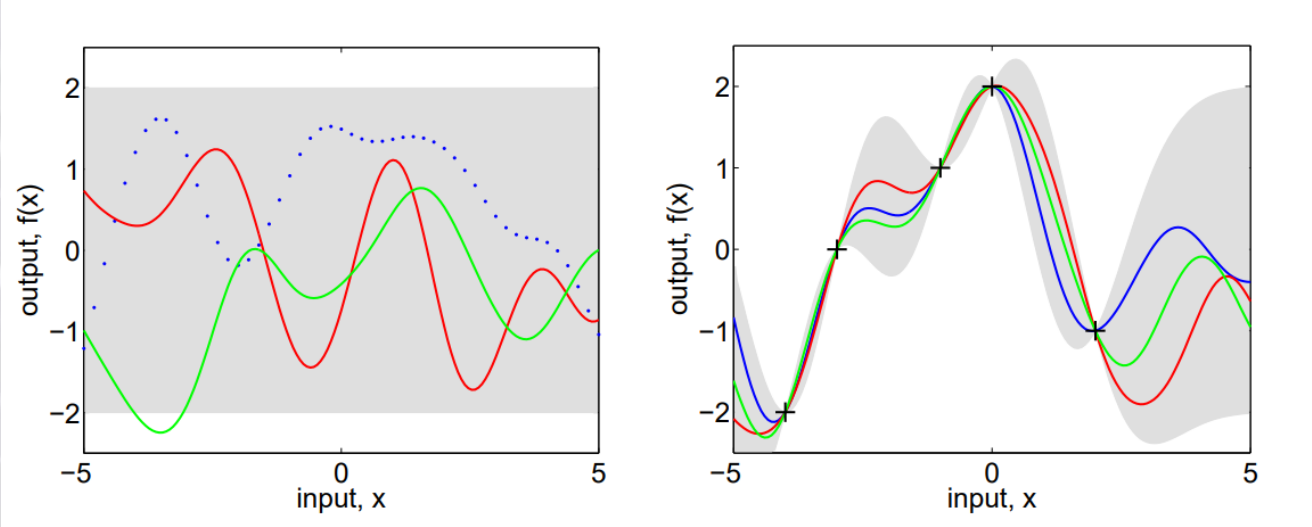
Gaussian Processes: Prediction with Noise-free Observations

• Graphically, you can think of generating functions from the prior, and rejecting the ones that disagree with the observations.



Gaussian Processes: Prediction with Noise-free Observations

• Graphically, you can think of generating functions from the prior, and rejecting the ones that disagree with the observations.

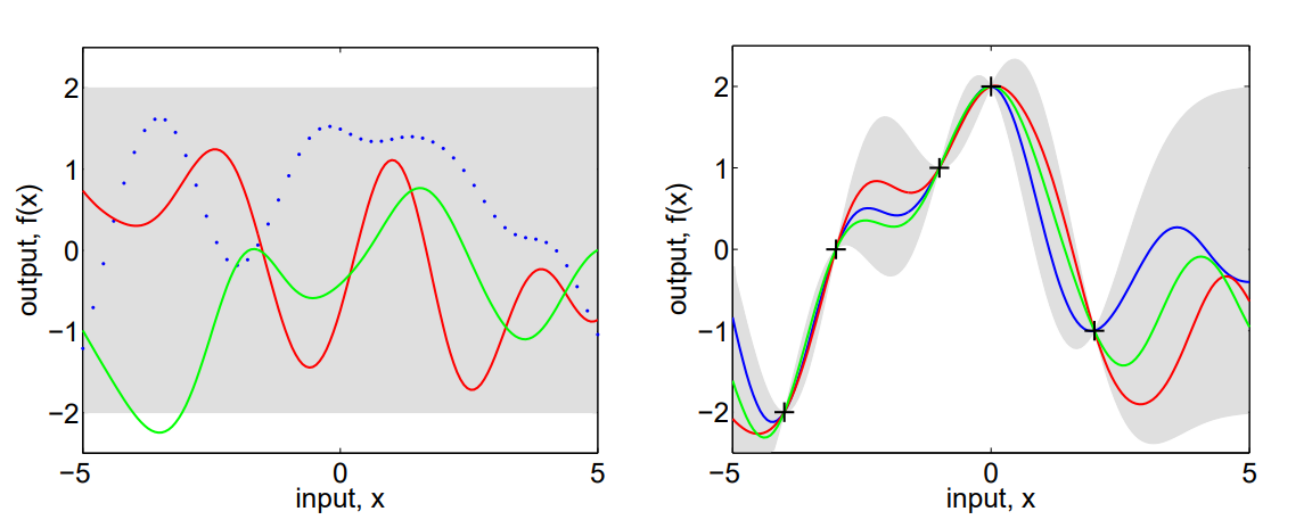


In practice, the posterior predictive function distribution is formed by <u>conditioning the joint</u>:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

on $\mathbf{f}$ (and X_* and X).

Gaussian Processes: Prediction with Noise-free Observations



The noise-free posterior predictive function distribution is formed by conditioning the joint:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

on $\mathbf{f}$ (and X_* and X).

This yields the following distribution, from which samples of $\mathbf{f}_*$ can be obtained:

$$\mathbf{f}_* \mid X_*, X, \mathbf{f} \sim N\left(K(X_*,X)K(X,X)^{-1}\mathbf{f}, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \right)$$

Gaussian Processes: Prediction with Noise-free Observations

The posterior predictive function distribution is formed by conditioning the joint:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

on **f** (and X_* and X).

$$\mathbf{f}_* \mid X_*, X, \mathbf{f} \sim N\left(K(X_*,X)K(X,X)^{-1}\mathbf{f}, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \right)$$

$\boldsymbol{\mu}$ $\qquad$ $\boldsymbol{\Sigma}$

Q: Where does this formula come from?

Gaussian Processes: Prediction with Noise-free Observations

The posterior predictive function distribution is formed by conditioning the joint:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

on **f** (and X_* and X).

$$\mathbf{f}_* \mid X_*, X, \mathbf{f} \sim N\left(K(X_*,X)K(X,X)^{-1}\mathbf{f}, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \right)$$

Q: Where does this formula come from?

A: The Gaussian conditioning formulas that we previously derived (check this)!

$$p(\mathbf{x}_1 \mid \mathbf{x}_2) = N(\mathbf{x}_1 \mid \boldsymbol{\mu}_{1|2}, \Sigma_{1|2})$$

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2)$$

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Gaussian Processes: Prediction with Noise-free Observations

The posterior predictive function distribution is formed by conditioning the joint:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

on $\mathbf{f}$ (and X_* and X).

$$\mathbf{f}_* \mid X_*, X, \mathbf{f} \sim N\left(K(X_*,X)K(X,X)^{-1}\mathbf{f}, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \right)$$

Q: What is the computational bottleneck in these expressions?

Gaussian Processes: Prediction with Noise-free Observations

The posterior predictive function distribution is formed by conditioning the joint:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

on **f** (and X_* and X).

$$\mathbf{f}_* \mid X_*, X, \mathbf{f} \sim N\left(K(X_*,X)K(X,X)^{-1}\mathbf{f}, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \right)$$

Q: What is the computational bottleneck in these expressions?

A: Matrix inversion $O(n^3)$ – can be reduced with numerical approximation or Strassen method; or using a small (yet useful) sample/training set.

Gaussian Processes: Prediction with Noise-free Observations

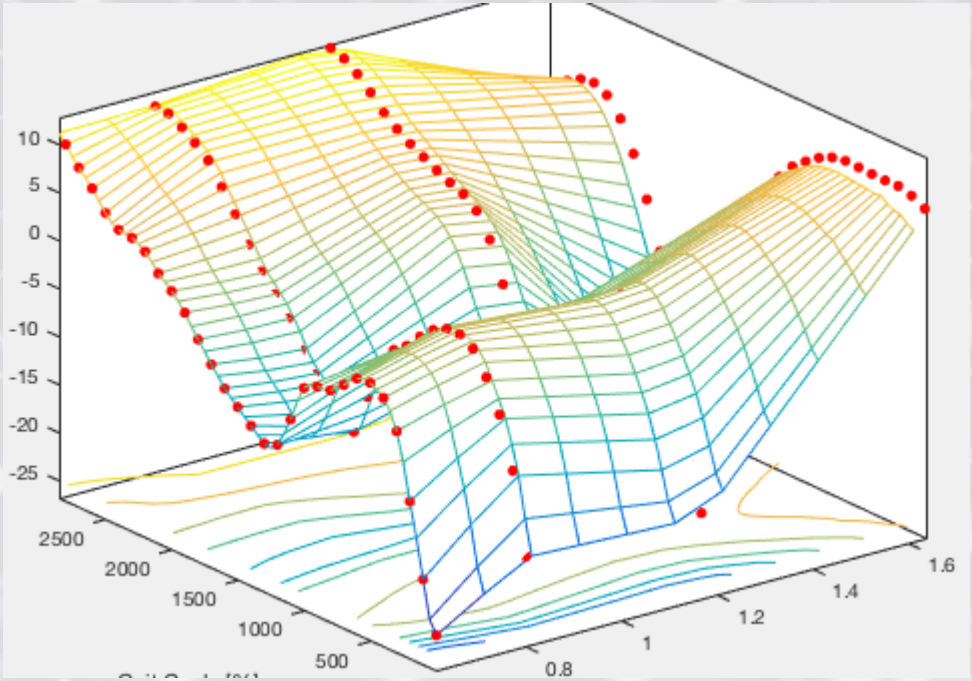
The posterior predictive function distribution is formed by conditioning the joint:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

on **f** (and X_* and X).

$$\mathbf{f}_* \mid X_*, X, \mathbf{f} \sim N\left(K(X_*,X)K(X,X)^{-1}\mathbf{f}, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \right)$$

Note that GPR (Gaussian Process Regression) can be instantiated in higher dimensions.



Gaussian Processes: Prediction with Noisy Observations

• It is typical for more realistic modelling situations that we do no have access to function values themselves, but only noisy versions thereof: y = f(x)+ε (e.g. imprecise/corrupted measurements).

• Assuming additive **independent, identically distributed** (IID) Gaussian noise ε, with variance σ_n^2, the prior on the noisy observations becomes:

$$\text{cov}(y_p, y_q) = k(\mathbf{x}_p, \mathbf{x}_q) + \sigma_n^2 \delta_{pq} \text{ or } \text{cov}(\mathbf{y}) = K(X, X) + \sigma_n^2 I$$

where is a **Knoecker delta function**, which is one *iff* p = q and zero otherwise (note the "noise term" above is a diagonal matrix due to independence).

Gaussian Processes: Prediction with Noisy Observations

• It is typical for more realistic modelling situations that we do no have access to function values themselves, but only noisy versions thereof: y = f(x)+ε (e.g. imprecise/corrupted measurements).

Assuming additive **independent, identically distributed** (IID) Gaussian noise ε, with variance σ_n^2, the prior on the noisy observations becomes:

$$\text{cov}(y_p, y_q) = k(\mathbf{x}_p, \mathbf{x}_q) + \sigma_n^2 \delta_{pq} \text{ or } \text{cov}(\mathbf{y}) = K(X, X) + \sigma_n^2 I$$

where is a **Knoecker delta function**, which is one *iff* p = q and zero otherwise (note the "noise term" above is a diagonal matrix due to independence).

• Introducing the noise term, we can we write the joint distribution of the observed target values and the function values at the test locations under the prior as:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right)$$

Gaussian Processes: Prediction with Noisy Observations

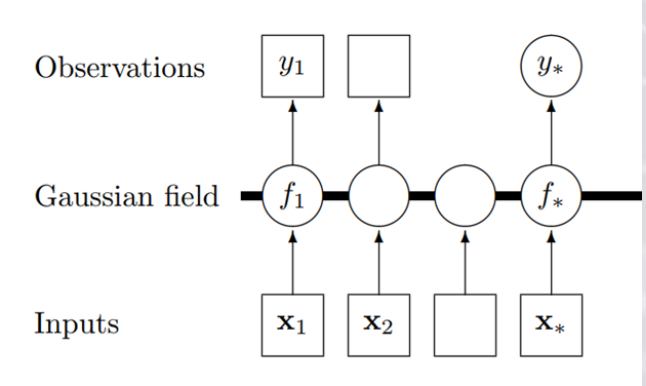
$$\begin{bmatrix}\mathbf{y}\\ \mathbf{f}_*\end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix}K(X,X)+\sigma_n^2 I & K(X,X_*)\\ K(X_*,X) & K(X_*,X_*)\end{bmatrix}\right)$$

• Once again, if we apply the conditioning formulas to the joint distribution, this yields the noisy posterior predictive function distribution:

$$\mathbf{f}_* \mid X_*, X, \mathbf{y} \sim N\left(K(X_*,X)\left[K(X,X)+\sigma_n^2 I\right]^{-1}\mathbf{y}, K(X_*,X_*)-K(X_*,X)\left[K(X,X)+\sigma_n^2 I\right]^{-1}K(X,X_*)\right)$$

$\boldsymbol{\mu}$ $\qquad$ $\boldsymbol{\Sigma}$

• Observe that our notation reflects the fact that y is generated from an underlying function perturbed by noise: $y = f(x)+\varepsilon$; in practice σ_n^2 is estimated from data or using prior knowledge.

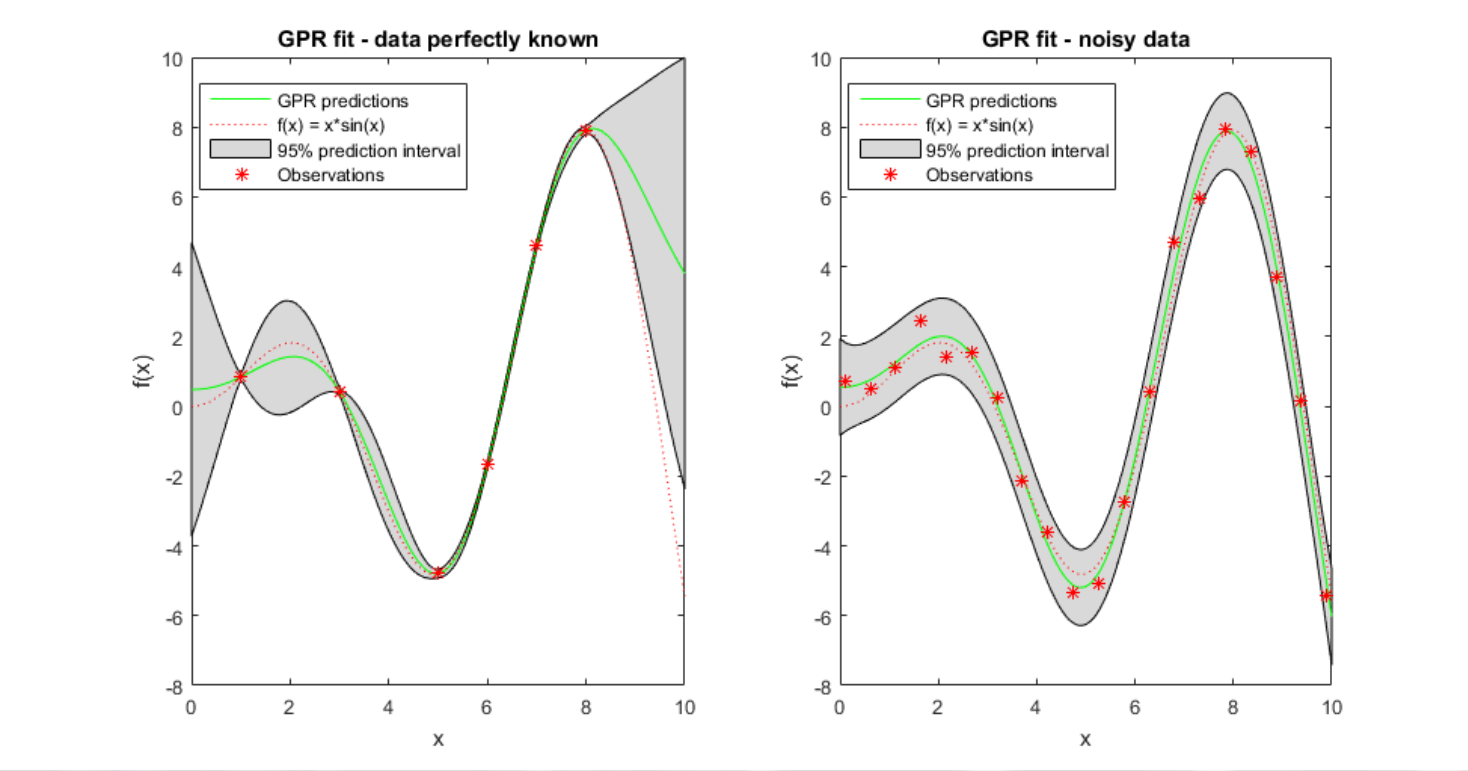


Gaussian Processes: Prediction with Noisy Observations

$$\begin{bmatrix}\mathbf{y}\\ \mathbf{f}_*\end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix}K(X,X)+\sigma_n^2 I & K(X,X_*)\\ K(X_*,X) & K(X_*,X_*)\end{bmatrix}\right)$$

- Once again, if we apply the conditioning formulas to the joint distribution, this yields the noisy posterior predictive function distribution:

$$\mathbf{f}_* \mid X_*, X, \mathbf{y} \sim N\left(K(X_*,X)\left[K(X,X)+\sigma_n^2 I\right]^{-1}\mathbf{y}, K(X_*,X_*)-K(X_*,X)\left[K(X,X)+\sigma_n^2 I\right]^{-1}K(X,X_*)\right)$$



Gaussian Processes: Varying the Hyperparameters

• The covariance hyperparameters can be varied (we discuss hyperparameter optimization next).

Recall that the square exponential covariance function (with noise) has the following form:

$$k_{SE}(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2l^2}\|\mathbf{x}-\mathbf{x}'\|^2\right) + \sigma_n^2\delta_{pq}$$

with **length scale** (l), **signal variance** (σ_f^2) and **noise variance** (σ_n^2) hyperparameters. Modifying the signal variance controls the overall variance of the function, while the length scale changes the degree of smoothing, trading it off against how well the curve matches the training data.

Gaussian Processes: Varying the Hyperparameters

Modifying the signal variance controls the overall variance of the function, while the length scale changes the degree of smoothing, trading it off against how well the curve matches the training data.

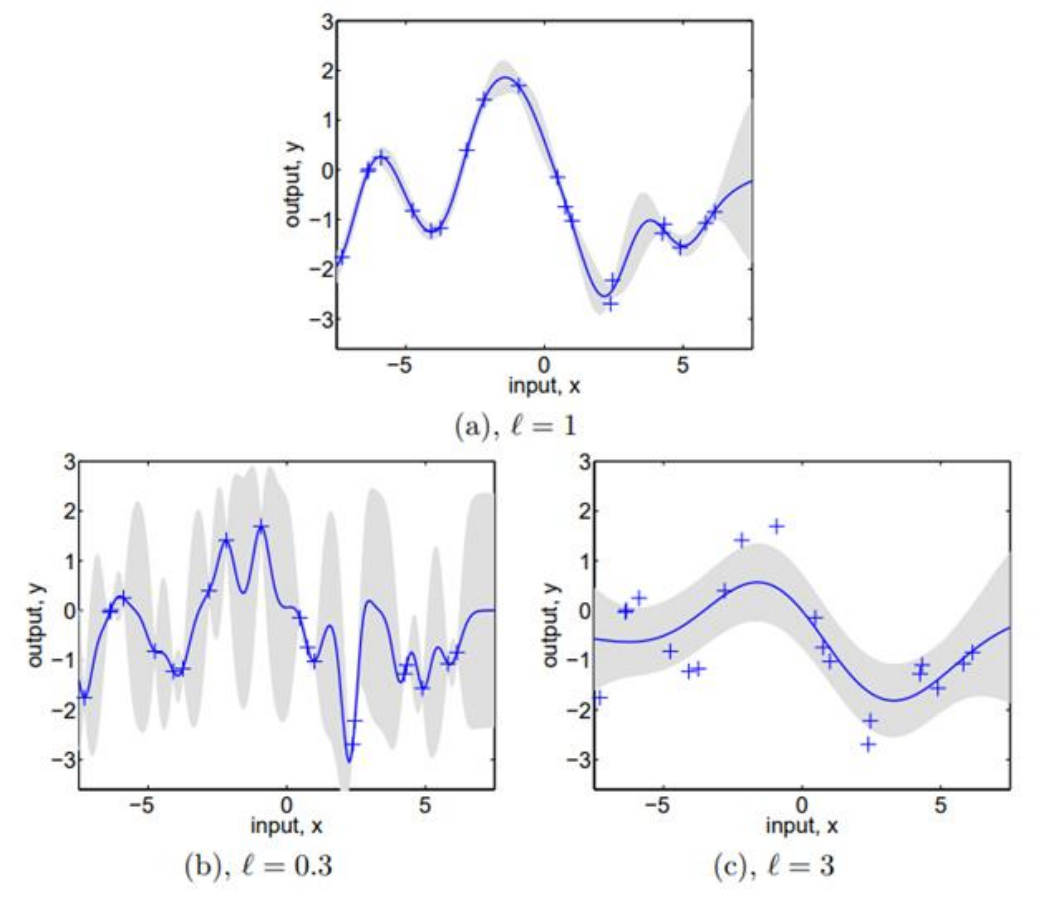


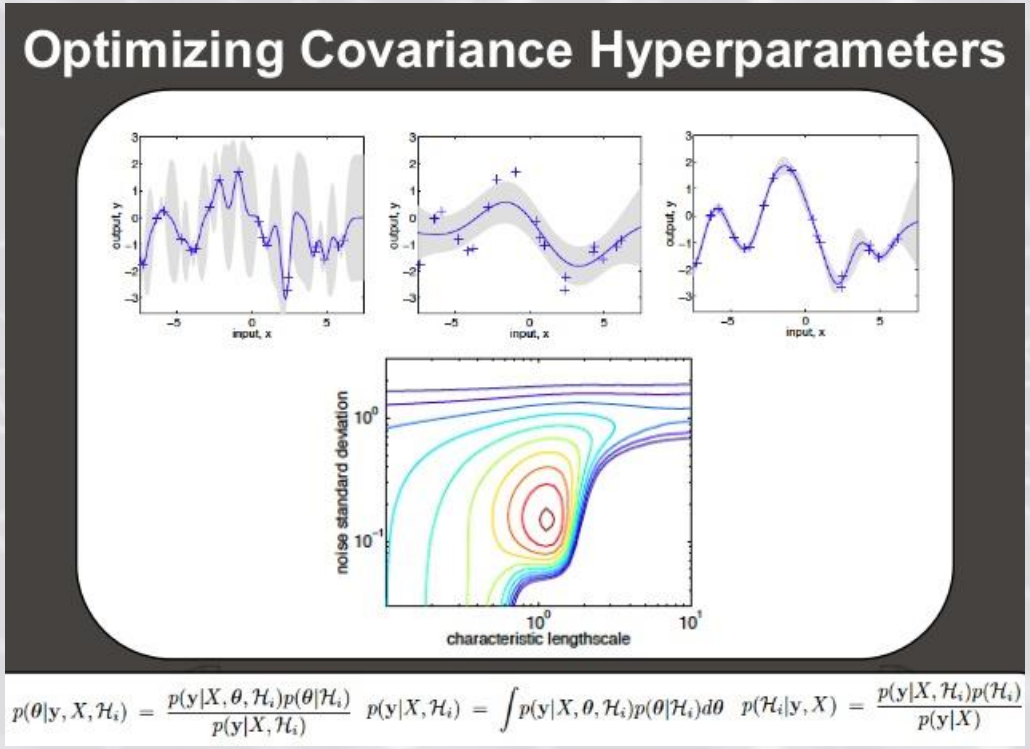
Figure 2.5: (a) Data is generated from a GP with hyperparameters $(\ell, \sigma_f, \sigma_n) = (1, 1, 0.1)$, as shown by the + symbols. Using Gaussian process prediction with these hyperparameters we obtain a 95% confidence region for the underlying function f (shown in grey). Panels (b) and (c) again show the 95% confidence region, but this time for hyperparameter values $(0.3, 1.08, 0.00005)$ and $(3.0, 1.16, 0.89)$ respectively.

Gaussian Processes: Learning the Hyperparameters

- The covariance hyperparameters can be tuned by minimizing the log likelihood:

$$\log(\mathbf{y} \mid X) = -\frac{1}{2}\mathbf{y}^T\left(K+\sigma_n^2 I\right)^{-1}\mathbf{y} - \frac{1}{2}\log\left|K+\sigma_n^2 I\right| - \frac{n}{2}\log 2\pi$$

The derivation of this formula relies once again on the conditional Gaussian equations. One can use *gradient descent* (or a variant such as conjugate gradients) to approximate the minimum value and thereby tune the hyperparameters.



Gaussian Process Regression Algorithm

GPR Algorithm

For a given training data set (X,**y**), test data $\mathbf{X}_*$, covariance function k() and hyperparameters: $\boldsymbol{\theta} = \sigma_f^2$, l, and σ_n^2:

- compute the covariance matrix $K=k(X, X) + \sigma_n I$ for hyper parameters $\boldsymbol{\theta}$
- compute the covariance matrix $\mathbf{k}^*=k(X, X_*)$.
- compute the covariance matrix $\mathbf{k}^{**}=k(X_*, X_*)$.
- the mean of the process is: $\mathbf{k}^{*T}K^{-1}\mathbf{y}$
- the covariance is: $\mathbf{k}^{**}-\mathbf{k}^{*T}K^{-1}\mathbf{k}^*$

(*) In practice, the inversion of the K matrix can be unstable. However, since K is *symmetric* and *positive definite* (why? Sum of positive definite matrices is PD – try showing this), it has a **Cholesky decomposition**, whereby: $K=LL^T$ (L represents a lower-triangular matrix); this allows us invert the desired matrix in a numerically stable way.

Research Application: GPs for Spatial Epidemiology

Vanhatalo, et al., "Sparse Log Gaussian Processes via MCMC for Spatial Epidemiology", JMLR, 2007.

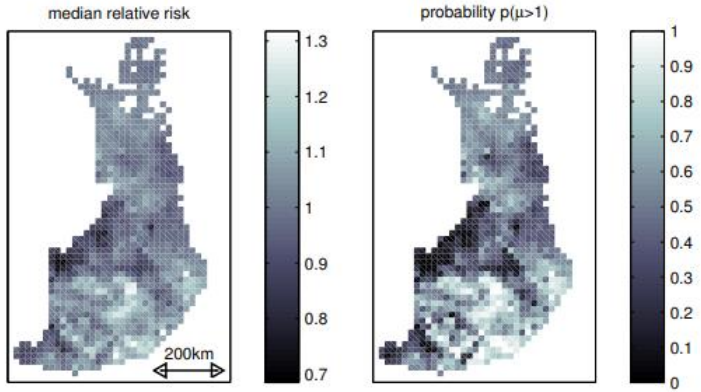
• Spatial epidemiology concerns both describing and understanding the spatial variation in the disease risk in geographically referenced health data. The authors use sparse log GPs for disease mapping, where the aim is to describe the overall disease distribution on a map and, for example, highlight areas of elevated or lowered mortality or morbidity risk.

• The expected number of deaths is evaluated using age, gender and scholarly degree standardization and the logarithm of the relative risk is given a Gaussian process prior; the authors use an MCMC method for hyperparameter approximation; the fully independent training conditional (FITC) sparse approximations are used to reduce the computational complexity of inverting the K matrix.

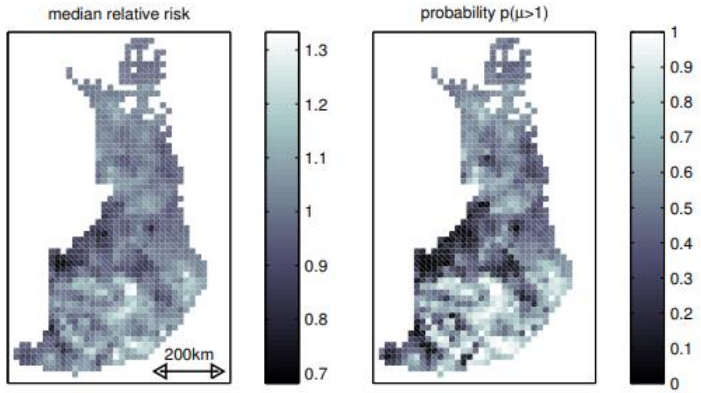
http://proceedings.mlr.press/v1/vanhatalo07a/vanhatalo07a.pdf

Research Application: GPs for Spatial Epidemiology

Vanhatalo, et al., "Sparse Log Gaussian Processes via MCMC for Spatial Epidemiology", JMLR, 2007.



(a) FITC sparse approximation



(b) Full Gaussian process

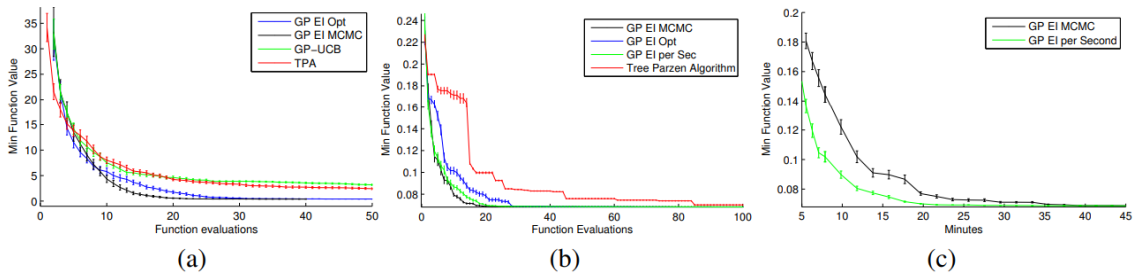
Figure 2: The relative risk of cerebral vascular diseases. The maps are results of models using the exponential covariance function trained with data aggregated in a 20km×20km lattice. In the FITC approximation there was 221 inducing inputs. The resolution in the maps is the same as in the training data. The posterior median and standard deviation of the length-scale of the covariance function were 33.0km and 9.8km in the FITC approximation and 27.0km and 7.9km in a full GP. In case of the FITC approximation and 10km×10km lattice data the median was 25.4km and the standard deviation 6.1km.

Research Application: GPs for Optimizing ML Algorithm Performance

Snoek, et al., "Practical Bayesian Optimization of Machine Learning Algorithms", NIPS, 2012.

• The authors attempt to automate parameter optimization for ML algorithms using GPs. of such parameters as a procedure to be automated. In particular, they use a Bayesian optimization framework; the posterior distribution reflects the results of running learning algorithm experiments with different hyperparameters.

• In order to pick new hyperparameter values to test, the authors apply two standard Bayesian optimization measures: EI (expected improvement) and UCB (upper confidence bound). The authors present novel algorithms to parallelize the hyperparameter search with an MCMC approach.



(a) (b) (c)

Figure 3: Comparisons on the Branin-Hoo function (3a) and training logistic regression on MNIST (3b). (3c) shows GP EI MCMC and GP EI per Second from (3b), but in terms of time elapsed.

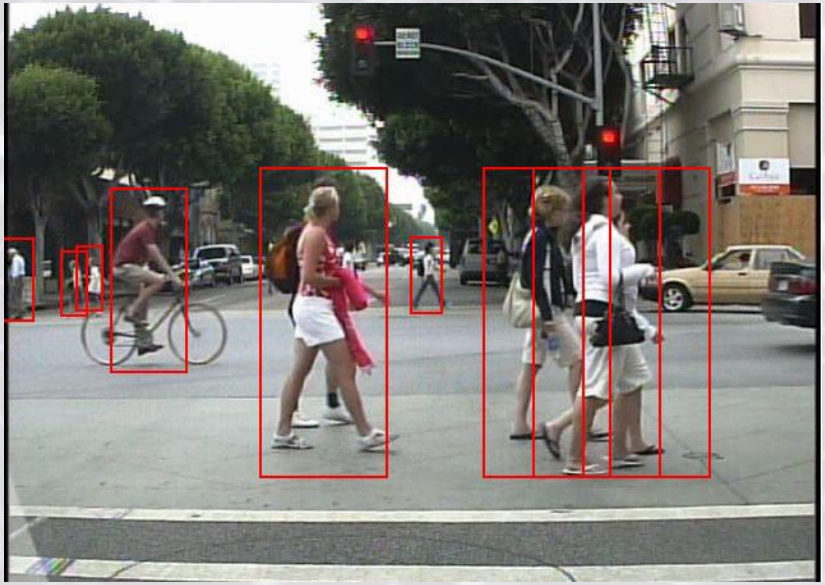
http://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf

Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Rhodes, A. D., Jordan Witte, Melanie Mitchell, and Bruno Jedynak. "Bayesian optimization for refining object proposals. ", IPTA, 2017.

Rhodes, A. D., Jordan Witte, Melanie Mitchell, and Bruno Jedynak. "Gaussian Processes with Context-Supported Priors for Active Object Localization," IJCNN, 2018.

- Accurate *object localization* is an enduring and critical challenge in computer vision.
- Precise and rapid localization of pedestrians in images remains an unsolved problem.

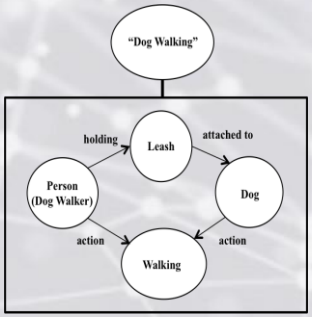


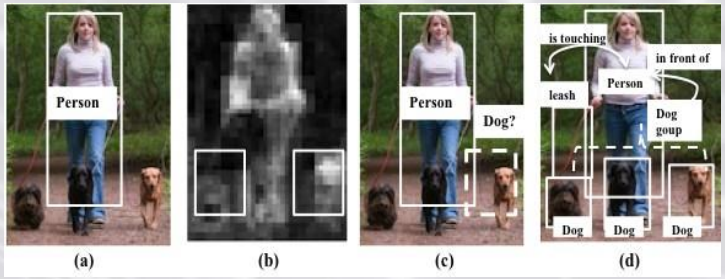
https://arxiv.org/ftp/arxiv/papers/1703/1703.08653.pdf

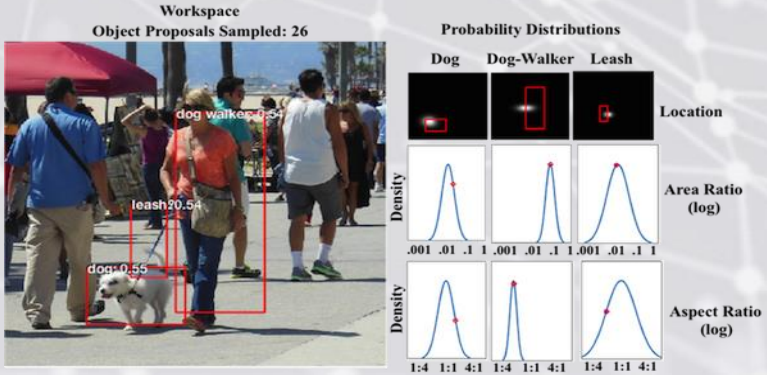
Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Background and Related Work

- *Situate* is a computer vision framework for *active* object localization in *visual situations*.
- Our system learns the expected structure of a "visual situation" from training images by inferring a set of joint probability distributions—a *context-situation model*—linking aspects of the relevant objects.



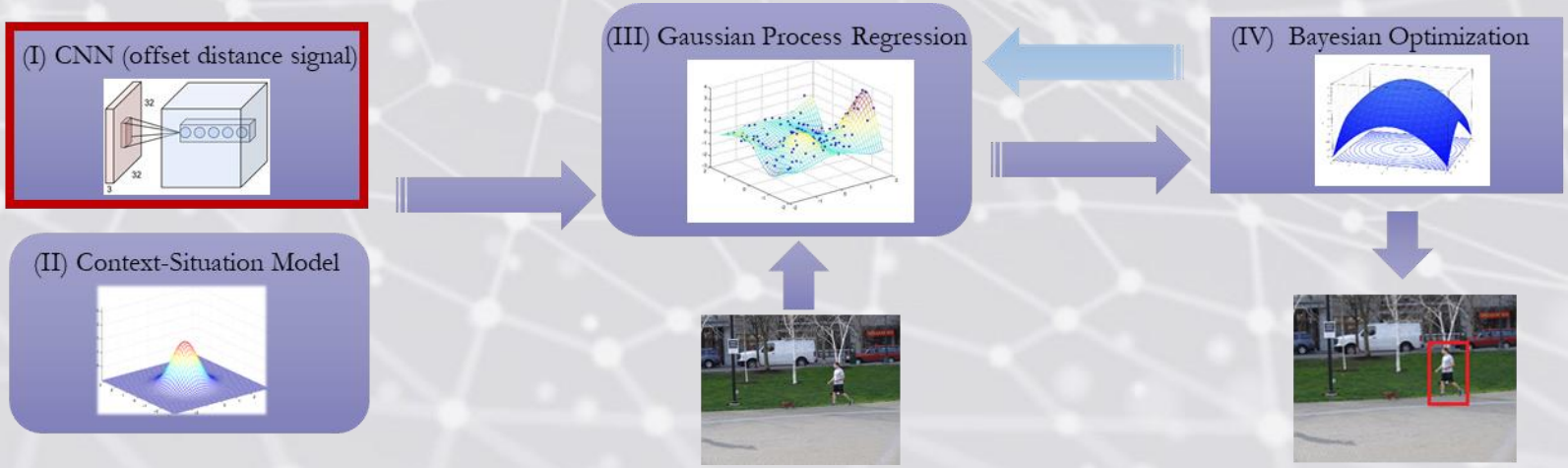




Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

General Algorithm Pipeline

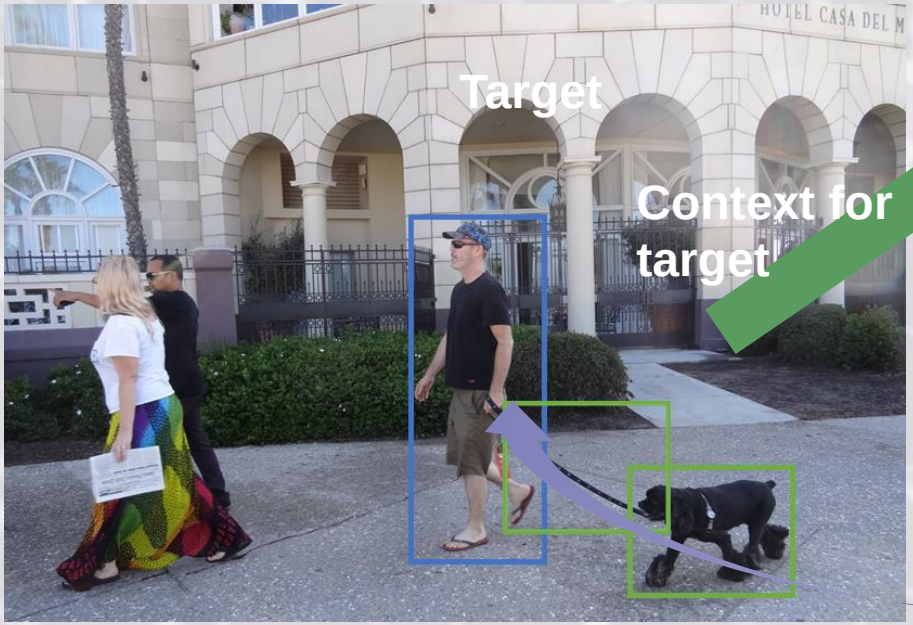
- (I) We train a convolutional neural network (CNN) to score bounding-box proposals to approximate an offset distance from the target object.
- (II) From training data, we develop context-situation model, given various location and size parameters for a particular visual situation.
- (III) We apply a Gaussian Process (GP) to approximate this offset response signal over the (large) search space of the target.
- (IV) A Bayesian active search is then used to achieve fine-grained localization of the target.



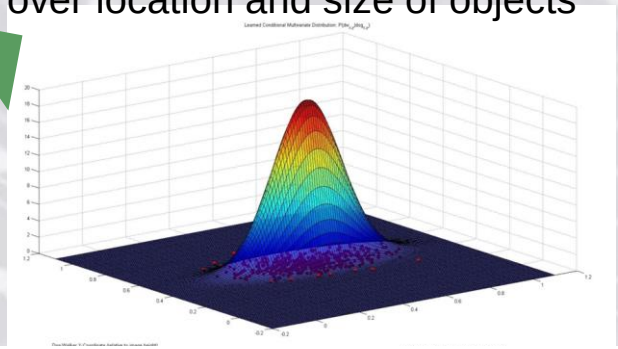
Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Context-Situation Models

- We define a *context-situation model* as a joint distribution of *location* and *size parameters* for a target object bounding-box, given various location and size parameters for a particular visual situation: $p\left(x_{target}, s_{target} \middle| \{x_{context}, s_{context}\}_{1:C}\right)$.
- We use the context-situation model to generate target object proposals used in an active search.



Context-Situation Model:
Learned joint probability distribution over location and size of objects

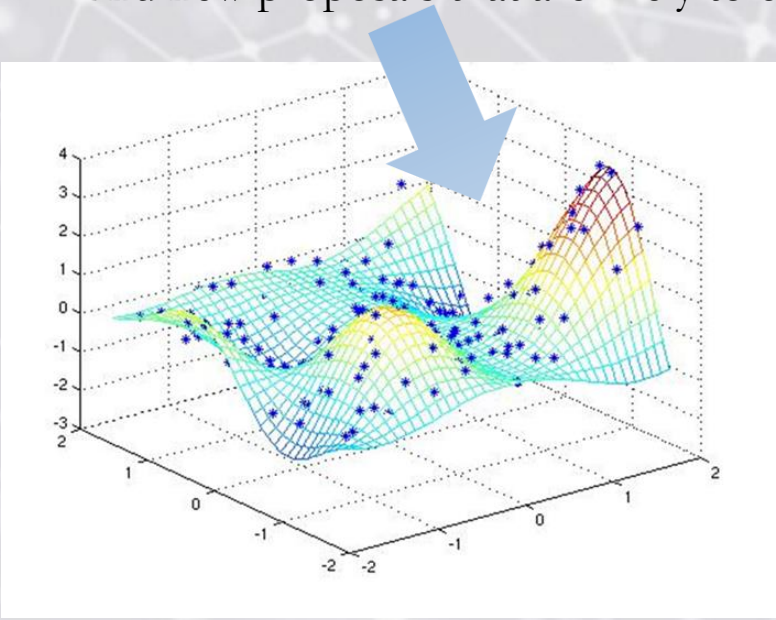


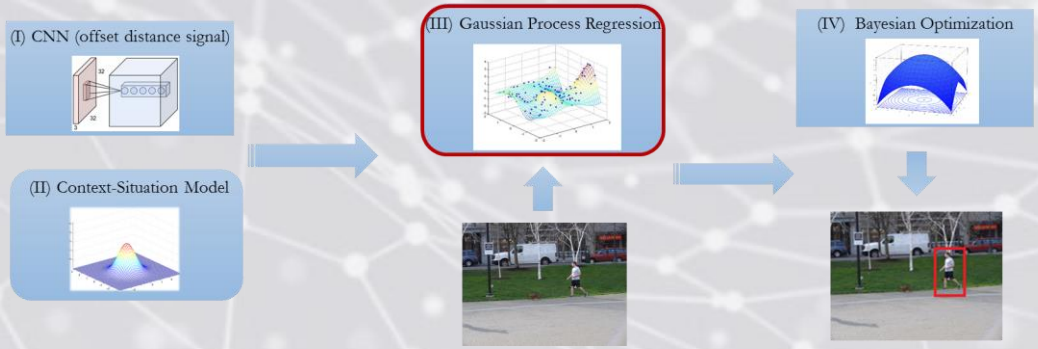
Sample from Context-Situation Model

Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Gaussian Process Regression

- Because it is computationally expensive to generate offset prediction values for a large number of bounding-box proposals (due to the CNN), we use GPR to approximate the offset prediction values over the *target search space* (*i.e.* a large grid of values).
- Next, we *actively search* this space according to a Bayesian optimization scheme (IV) to find new proposals that are likely to capture the target object.

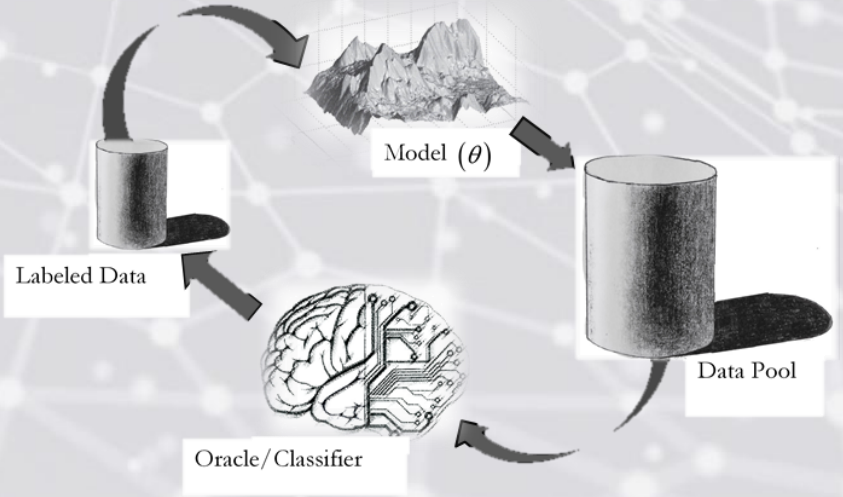




Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Active Learning Queries

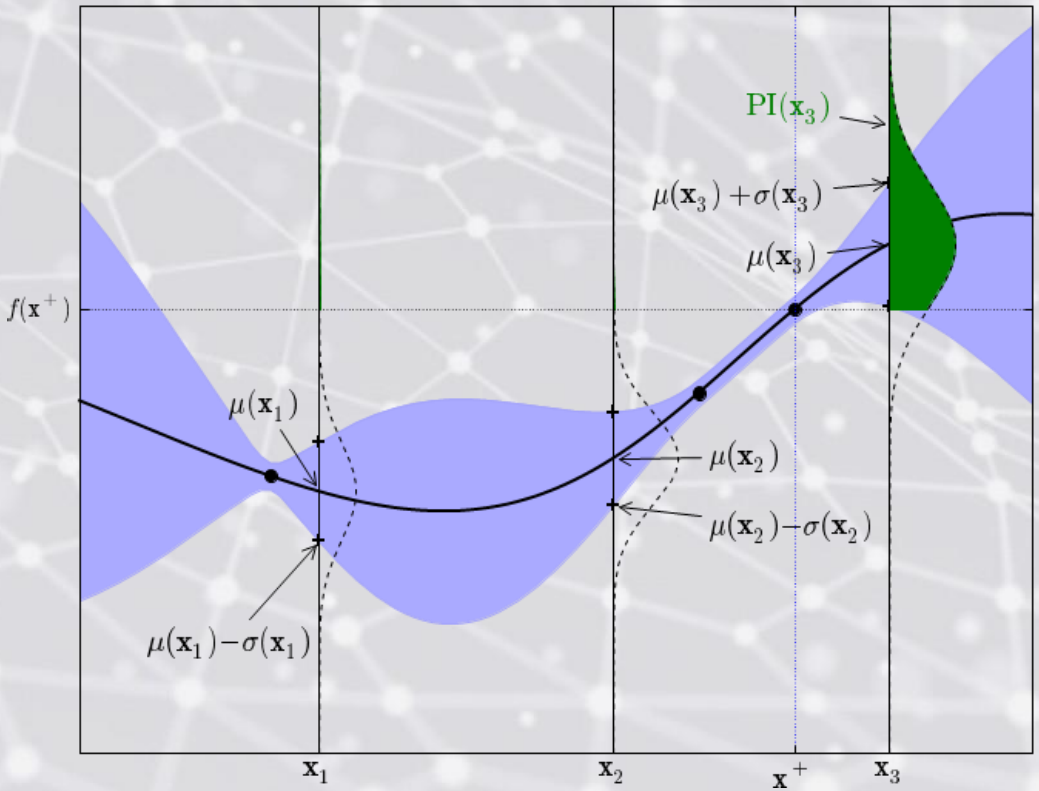
- Ideally, in addition to *exploring* regions of high uncertainty, we should also *exploit*, to some degree, "regions of promise", respecting our target object.
- *Acquisition functions* are used to guide the search for the optimum of the GPR approximation to the true objective function (whose maximum occurs, ideally, for a proposal that perfectly crops the pedestrian).
- *High acquisition* indicates greater likelihood of an objective function maximum.
- Commonly used acquisition functions (we omit the details for brevity) in this setting include: *probable improvement* (PI) and *expected improvement* (EI).



Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Bayesian Optimization

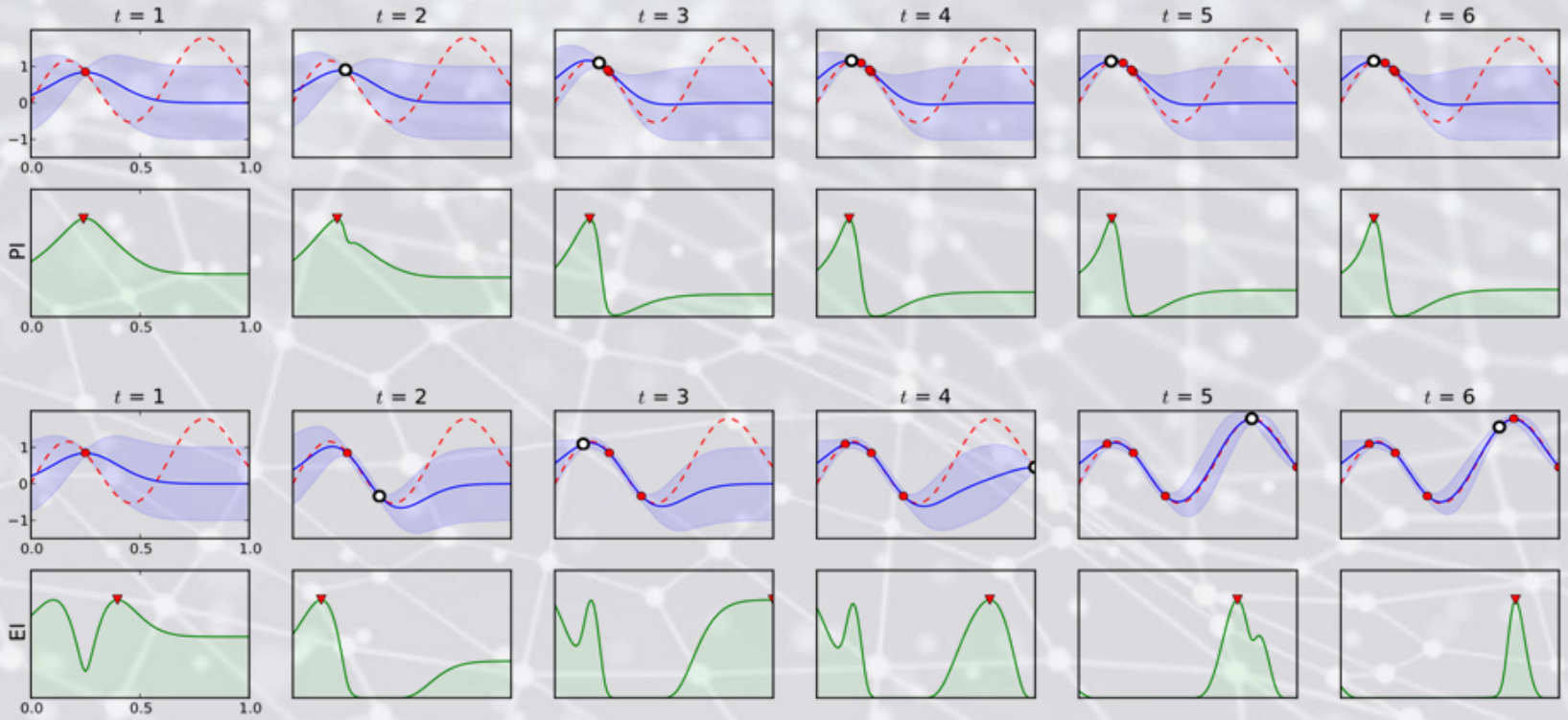
- In the figure we display a Gaussian process showing the region of *probable improvement*. The maximum observation is at $\mathbf{x}^+$.
- The darkly-shaded area in the superimposed Gaussian above the dashed line can be used as a measure of improvement. The model predicts almost no possibility of improvement by observing at $\mathbf{x}_1$ or $\mathbf{x}_2$, while sampling at $\mathbf{x}_3$ is more likely to improve on f($\mathbf{x}^+$).
-



Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Bayesian Optimization

- Below are example iterations of both *PI* and *EI*-based active queries with GPR.



- In the current work, we use a variant of *EI* that is fine-tuned to our problem parameters.

Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

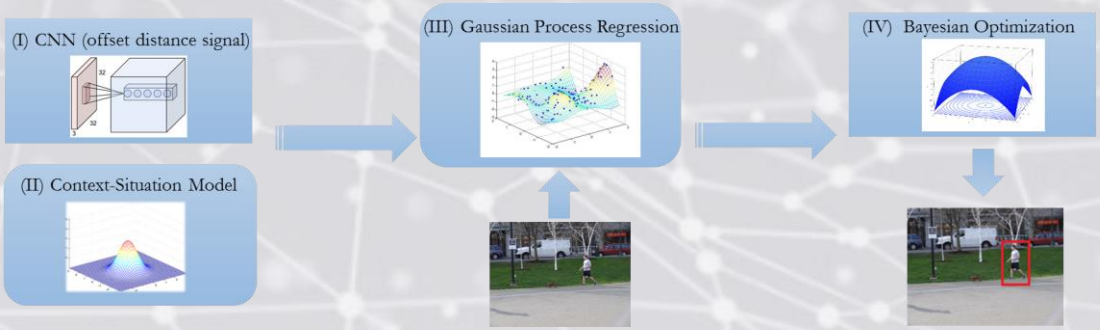
Algorithm: Gaussian Process Context Localization (GP-CL)

Input: Image I, a set of C context objects, trained model y giving response signals, learned context-situation model $p(x_{target}, s_{target} \mid \cdot)$, n_0 initial bounding-box proposals for target object generated by the context-situation model, and corresponding response signal values: $D_{n_0} = \{(x_i, s_i), y(x_i, s_i)\}_{i=1}^{n_0}$, GP hyperparameters θ, size of GP realization space M, dynamic design parameter for Bayesian active search ξ, size of GP memory GP_{mem} (as number of generations used), batch size n, number of iterations T, current set of bounding-box proposals and response signals $D_{proposal}^{(t)}$.

1:Compute n_0 initial bounding box proposals:
$\{(x_i, s_i)\}_{i=1}^{n_0} \sim p(x_{target}, s_{target} \mid \cdot)$
2: $D_{proposal}^{(0)} \leftarrow D_{n_0}$
3:**for** t = 1 to T **do**
4: Compute $\mu(x)^{(t)}$ and $\sigma(x)^{(t)}$ for the GP realization $f_M^{(t)}$ of $D_{proposal}^{(t-1)}$ over grid of M points (Equation 4)
5: **for** i = 1 to n **do**
6: $z_i = \underset{x}{\operatorname{argmax}}\, a_{CEI}\left(f_M^{(t)} \setminus \{z_j\}_{j=1}^{j=i-1}, \xi\right)$ (Equation 5)
7: $sample: s_i \sim p(\cdot)_s$
8: $p_i = (z_i, s_i)$
9: **end for**
10: $D^{(t)} \leftarrow \{(x_i, s_i), y(x_i, s_i)\}_{i=1}^{n}$
11: $D_{propos\ al}^{(t)} \leftarrow \bigcup_{j=t-GP_{mem}}^{t} D^{(j)}$
12: **end for**
13: **Return** $\underset{x}{\operatorname{argmax}}\, \mu(x)^{(T)}$

GP-CL Algorithm

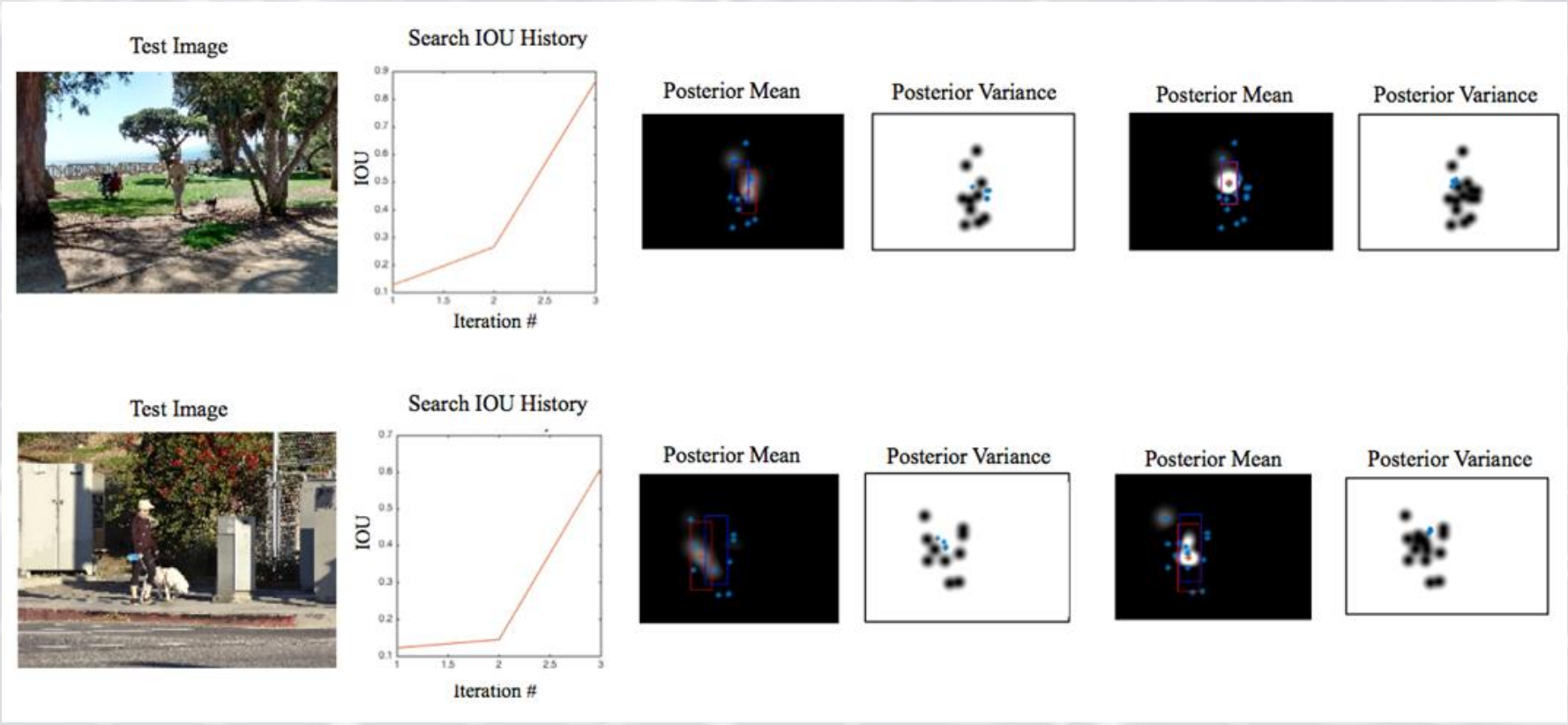
Step 1: Sample initial target proposals from context-situation model
Step 2: Score these proposals using the offset-prediction model (CNN)
Step 3: Compute GPR values over search space
Step 4: Using Bayesian optimization procedure, return proposals in search space with maximum acquisition
Step 5: **Return** to Step 3 (loop)



Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

GP-CL Example Runs

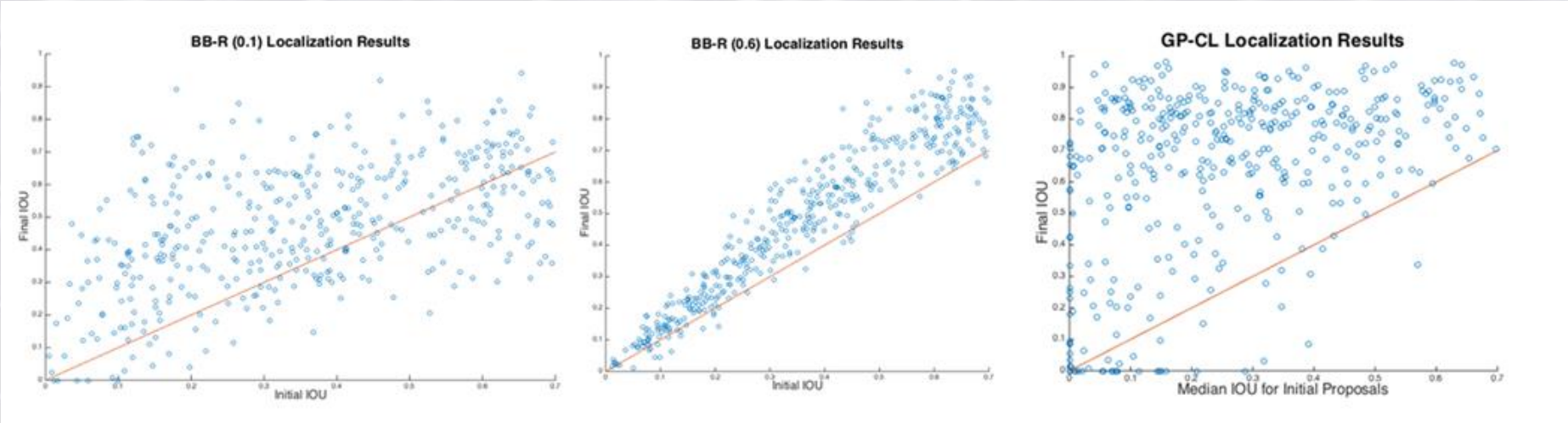
- Examples of runs on two test images with the GP-CL algorithm. In each row the test image is shown on the far-left; the "search IOU history" is displayed in the second column, with the algorithm iteration number on the horizontal axis and IOU with the ground-truth target bounding box on the vertical axis.



Research Application: GPs Efficient Object (e.g. Pedestrian) Detection with Context Models

Experimental Results

- Graph of BB-R (0.6), BB-R (0.1) and GP-CL localization results for test images. The horizontal axis indicates the median IOU for the initial proposal bounding boxes, while the vertical axis designates the final IOU with the target object ground truth. The line depicted indicates "break-even" results.



Method	IOU Difference Median (SE)	Median Relative IOU Improvement	% of Test Set with IOU Improvement	% of Test Set Localized
BB-R (0.6)	.0614 (.0035)	34.62%	**90.1%**	12.3%
BB-R (0.1)	.1866 (.0077)	92.91%	90.0%	33.2%
GP-CL	**.4742** (.012)	**194.02%**	89.3%	**75.2%**

Fin