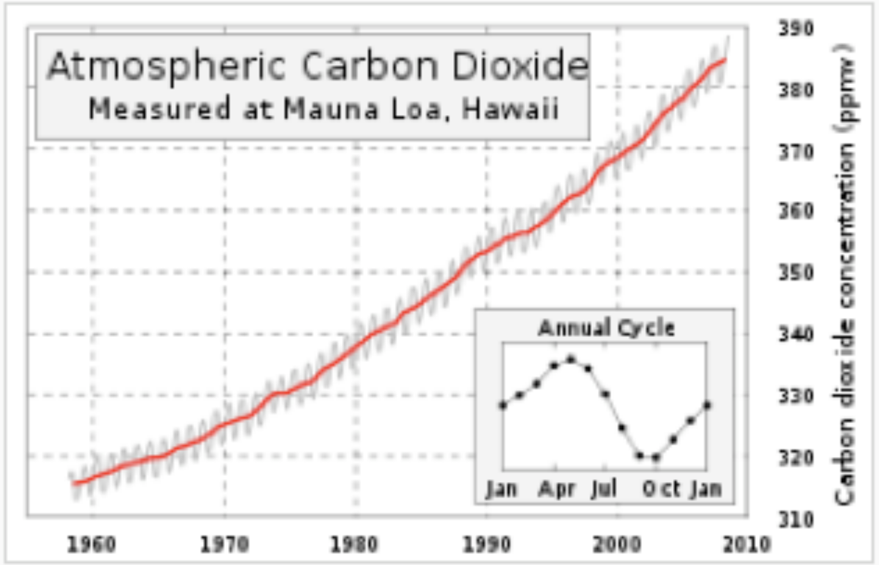
Gaussian Processes

Sargur Srihari

Topics in Gaussian Processes

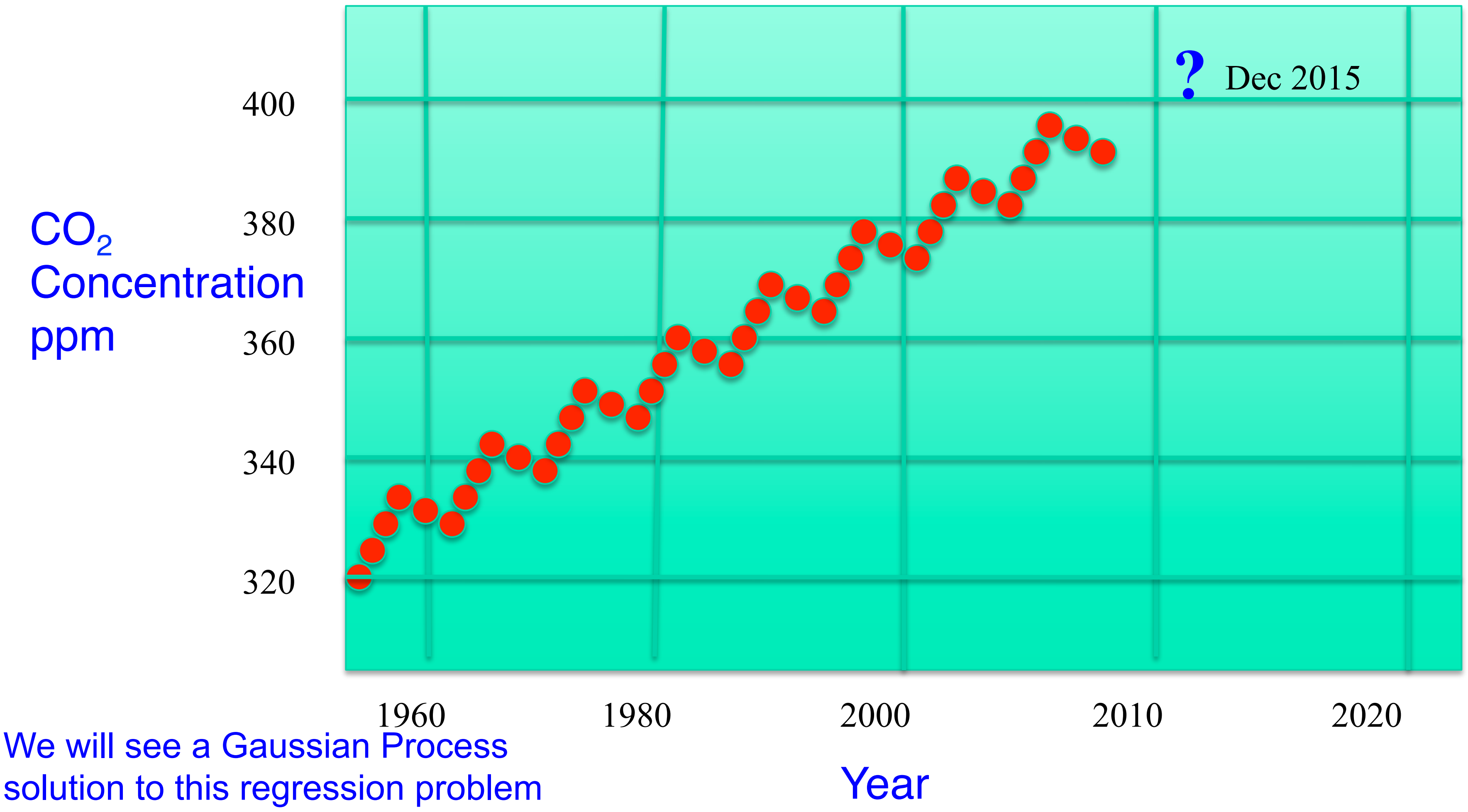
1. Examples of use of GP
2. Duality: From Basis Functions to Kernel Functions
3. GP Definition and Intuition
4. Linear regression revisited
5. Gaussian processes for regression
6. Learning the hyperparameters
   Automatic Relevance Determination
7. Gaussian processes for classification
   Laplace approximation
8. Connection to neural networks

National Academy of Sciences: Keck Center


Atmospheric Carbon Dioxide
Measured at Mauna Loa, Hawaii
Annual Cycle
Jan
Apr
Jul
Oct
Jan
1960
1970
1980
1990
2000
2010
390
380
370
360
350
340
330
320
310
Carbon dioxide concentration (ppmv)

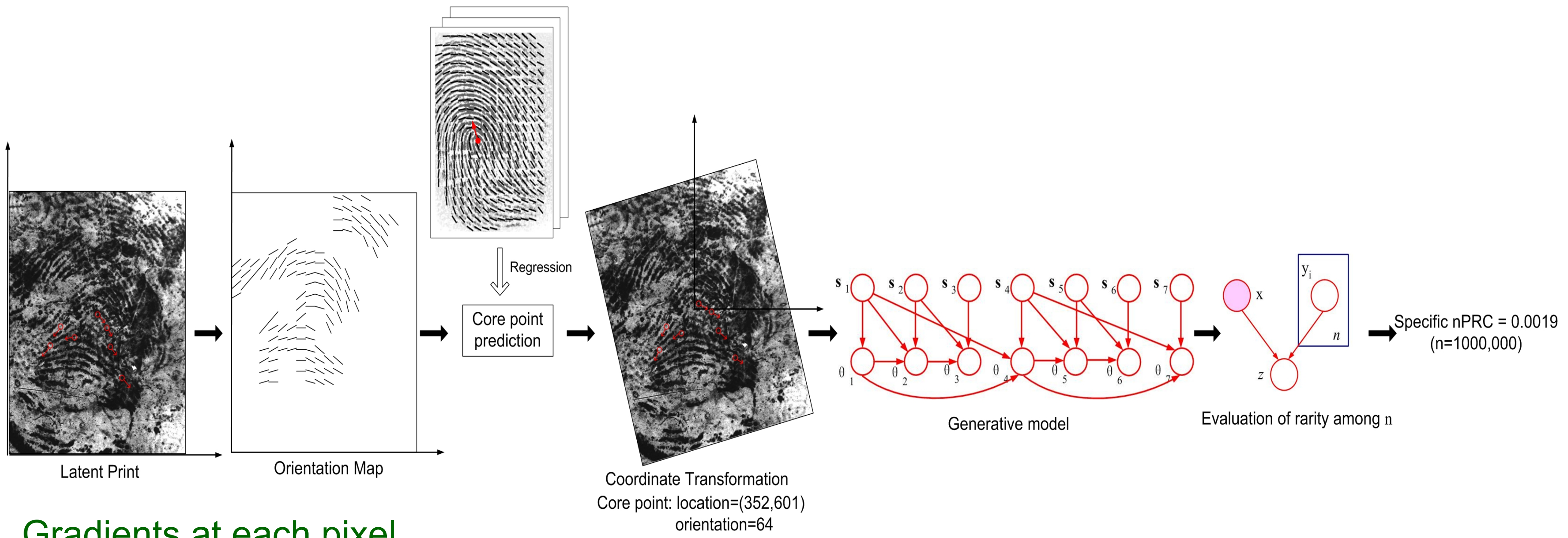
The Keeling Curve of atmospheric CO_2 concentrations measured at Mauna Loa Observatory.

Regression Problem: Carbon Dioxide in Atmosphere



We will see a Gaussian Process solution to this regression problem

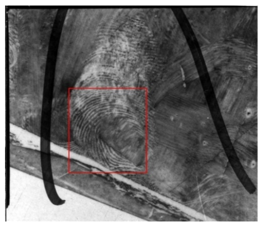
Fingerprint Core Point using GP Regression

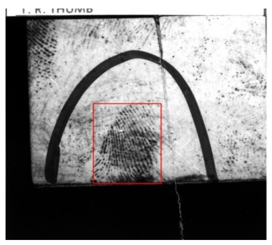


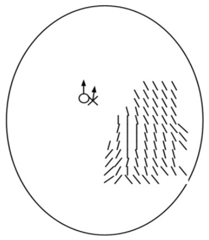
Gradients at each pixel

0 0 0 34 32 33 32 38 42 35 29 32 24 24 22 21 17 10 5 169 143 131 135 131 131 127 139
146 0 0 0 0 0 0 0 32 33 33 30 34 39 45 27 33 31 22 22 29 15 11 7 176 150 141 134 130
136 119 129 131 118 0 0 0 0 0 29 30 31 35 36 35 39 40 33 33 36 24 21 34 25 17 10 1 173
151 140 132 136 122 129 113 110 0 0 0 0 0 34 32 33 31 30 39 41 43 37 32 31 19 23 26
16 9 8 178 161 146 136 125 129 116 134 127 121 0 0 0 0 0 41 38 36 30 33 44 43 43 39
35 31 26 23 26 19 5 1 176 164 156 136 121 129 114 121 110 102 0 0 0 · · · · · ·

$\Rightarrow$ Core point (x, y, θ): $(253, 221, 85)$







NIPS 2010
Su and Srihari

Paper at NIPS 2010

NIPS 2010

Evaluation of Rarity of Fingerprints in Forensics

Chang Su and Sargur N. Srihari

University at Buffalo, The State University of New York

ABSTRACT

Since the earliest days of forensics, importance of considering the rarity of features used in the comparison of prints has been known. Yet methods to compute rarity of features has been elusive due to the large number of variables and the complexity of the distributions. When a latent print, typically found in a crime scene, is compared to a known (inked or live scan) in a database, the degree of uncertainty involves both rarity and similarity. It has become necessary to quantify this uncertainty in courtroom situations.

Proposed method for rarity, based on level 2 detail (minutiae), determines probability of random match among n knowns:

1. The coordinate system is transformed into standard position with core point as origin; Gaussian Process regression is used to determine the core point. Method better than standard topology-based method.
2. Generative model, that takes into account inter-minutia dependencies and minutia confidences determines evidence probability; model is validated using a goodness-of-fit test.
3. Specific probability of match among n is evaluated.

The rarity of several examples, including latent fingerprints and minutia configurations are given.

1. Core Point Prediction (Gaussian Process)

Given orientation map of latent print, core point is predicted by

$$\hat{y}^* = \mathbf{k}(\mathbf{g}^*_{MAX}, G)[K+\sigma^2 I]^{-1}\mathbf{y} \quad \text{where} \quad k(\mathbf{g},\mathbf{g}') = \theta_1 \exp(-\frac{\theta_2}{2}|\mathbf{g}-\mathbf{g}'|^2)$$

where $\mathbf{g}^*_{MAX}$ is orientation map with the maximum predictive probability

$$\mathbf{g}^*_{MAX} = \underset{\mathbf{g}^*}{\operatorname{argmax}}\, p(m(y^*)|\mathbf{g}^*, G, \mathbf{y})$$

Orientation map (32 x 32 vector of gradients):

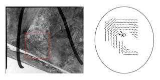
```
0 0 0 34 32 33 32 38 42 35 29 32 24 24 22 21 17 10 5 169 143 131 135  131 131 127 139
146 0 0 0 0 0 0 0 32 33 33 30 34 39 45 27 33 31 22 22 29 15 11 7 176 150 141 134  130
136 119 129 131 118 0 0 0 0 0 29 30 31 35 36 35 39 40 33 33 36 24 21 34 25 17 10 1 173
151 140  132 136 122 129 113 110 0 0 0 0 0 34 32 33 31 30 39 41 43 37 32 31 19 23 26
16 9 8 178 161 146 136  125 129 116 134 127 121 0 0 0 0 0 41 38 36 30 33 44 43 43 39
35 31 26 23 26 19 5 1 176 164 156 136 121 129 114 121 110 102 0 0 0 · · · · · ·
```

Core point

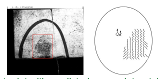
(s, θ): [(253, 221), 85]

Orientation map estimated in blocks of pixels using gradients

$$\theta_o = \frac{1}{2}\tan^{-1}\left(\frac{\sum_{i=1}^{W}\sum_{j=1}^{W} 2G_x(i,j)G_y(i,j)}{\sum_{i=1}^{W}\sum_{j=1}^{W}\left(G_x^2(i,j)-G_y^2(i,j)\right)}\right)$$



Latent print with predicted core point *within* print



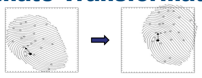
Latent print with predicted core point *outside* print

Performance Comparison with Standard Method (Poincare topology)

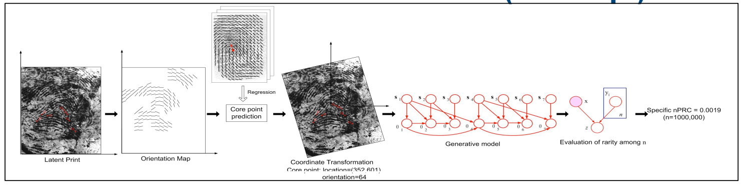
	Poincare Index	Gaussian Processes
Good	90.6%	93.1%
Bad	68.2%	87.1%
Ugly	46.6%	72.7%
Overall	68.6%	84.5%

Training data : NIST4(4000 fingerprints)
Testing data: NIST27
(258 latent prints: 88 good, 85 bad, 85 ugly)

2. Coordinate Transformation

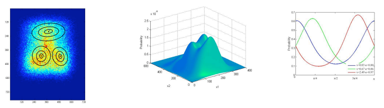


Process Flow for a Latent Print (Four Steps)

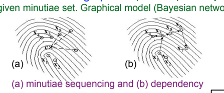


3. A Generative Model

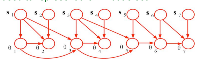
Distribution of minutiae: mixture model with bivariate Gaussian for location and von Mises for Orientation



Distribution of fingerprint captures dependency between minutiae: define a unique sequence for given minutiae set. Graphical model (Bayesian network) used to represent the minutiae set.



(a) minutiae sequencing and (b) dependency



Graphical model

For minutia set X joint distribution is

$$\boxed{p(\mathbf{X}) = p(\mathbf{s}_1)p(\theta_1|\mathbf{s}_1)\prod_{n=2}^{N} p(\mathbf{s}_n)p(\theta_n|\mathbf{s}_n, \mathbf{s}_{\psi(n)}, \theta_{\psi(n)})}$$

s =location, θ is orientation, ψ(n) are parent nodes

which involves three mixture models

$$f(\mathbf{s}) = \sum_{k_1=1}^{K_1}\pi_{k_1}\mathcal{N}(\mathbf{s}|\mu_{k_1},\Sigma_{k_1})$$

$$f(\mathbf{s},\theta) = \sum_{k_2=1}^{K_2}\pi_{k_2}\mathcal{N}(\mathbf{s}|\mu_{k_2},\Sigma_{k_2})\mathcal{V}(\theta|\nu_{k_2},\kappa_{k_2})$$

Within a mixture component vonMises models orientation given the location

$$f(\theta_n|\mathbf{s}_n,\mathbf{s}_{\psi(n)},\theta_{\psi(n)}) = \sum_{k_3=1}^{K_3}\pi_{k_3}\mathcal{V}(\theta_n|\nu_{k_3},\kappa_{k_3})$$

Goodness of Fit: Chi-square test for three generative models.

Generative models	Dataset sizes	Model accepted	Model rejected
$f(\mathbf{s})$	4000	3387	613
$f(\mathbf{s},\theta)$	4000	3216	784
$f(\theta_n\|\mathbf{s}_n,\mathbf{s}_{\psi(n)},\theta_{\psi(n)})$	4096	3558	538

4. Specific nPRC Calculation with Minutiae Confidence

Specifying minutia correspondence

Minutiae are within tolerance ε = [ϵ_s, ϵ_θ].

$$\| \mathbf{s}_a - \mathbf{s}_b \| \le \epsilon_s \wedge |\theta_a - \theta_b| \le \epsilon_\theta$$

Modeling minutia quality

Confidence of minutia x_n is $(d_{sn}, d_{\theta n})$, where d_{sn} is location confidence and $d_{\theta n}$ is direction confidence. Given minutiae $x_n = (s_n, \theta_n)$ distributions of location s' and direction θ' are modeled by Gaussian and von-Mises:

$$c(s'|s_n, d_{s_n}) = \mathcal{N}(s'|s_n, d_{s_n}^{-1}) \qquad c(\theta'|\theta_n, d_{\theta_n}) = \mathcal{V}(\theta'|\theta_n, d_{\theta_n})$$

Specific nPRC given by

$$p_\epsilon(\mathbf{X}, \hat{m}, n) = 1 - (1 - p_\epsilon(\mathbf{X}, \hat{m}))^{n-1}$$

where $p_e(X, \hat{m})$ is the probability that $\hat{m}$ minutiae match between latent print and a random one among n knowns

$$p_\epsilon(\mathbf{X}, \hat{m}) = \sum_{m' \in M} p(m')\binom{m'}{\hat{m}} \cdot \sum_{i=1}^{\binom{N}{\hat{m}}} p_\epsilon(\widetilde{\mathbf{X}}_i)$$

and probability that there is a one-to-one correspondence between ^X and ^X' is given by

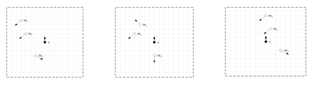
$$p_\epsilon(\widetilde{\mathbf{X}}) = p_\epsilon(\mathbf{s}_1, \theta_1)\prod_{n=2}^{\hat{m}} p_\epsilon(\mathbf{s}_n)p_\epsilon(\theta_n|\mathbf{s}_n, \mathbf{s}_{\psi(n)}, \theta_{\psi(n)})$$

$$p_\epsilon(\mathbf{s}_n, \theta_n) = \int_{s'}\int_{\theta'}\iint_{|\mathbf{x}-\mathbf{x}'|\le\epsilon} c(\mathbf{s}'|\mathbf{s}_n, d_{s_n})c(\theta'|\theta_n, d_{\theta_n})f(\mathbf{s},\theta)ds'd\theta' d\mathbf{s}d\theta \qquad p_\epsilon(\mathbf{s}_n) = \int_{s'}\int_{|\mathbf{s}-\mathbf{s}'|\le\epsilon_s} c(\mathbf{s}'|\mathbf{s}_n, d_{s_n})f(\mathbf{s})ds'd\mathbf{s}$$

$$p_\epsilon(\theta_n|\mathbf{s}_n, \mathbf{s}_{\psi(n)}, \theta_{\psi(n)}) = \int_{\theta'}\int_{|\theta-\theta'|\le\epsilon_\theta} c(\theta'|\theta_n, d_{\theta_n})f(\theta|\mathbf{s}_n, \mathbf{s}_{\psi(n)}, \theta_{\psi(n)})d\theta' d\theta$$

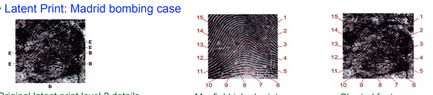
Examples of Rarity Evaluation

• Three Minutia Configurations, specific nPRC for n=1000



1.2 E-2,　　*7.9 E-4*　　*2.3 E-6*

• Latent Print: Madrid bombing case



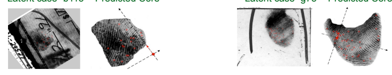
Original latent print level 2 details.　　Mayfield inked print　　Charted features on latent print

Probability of at least one print in FBI IAFIS database (470 million prints) that shares the same minutiae is 0.93.

Probability of falsely identifying source of latent print is 7.8 ×10^{-7}

• Latent Prints: NIST special dataset 27

Latent case "b115"　Predicted Core　　Latent case "g73"　Predicted Core



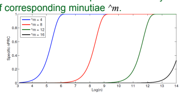
Minutiae confidence manually assigned by visual inspection. Tolerance set at 10 pixels and π/8.

Specific nPRCs for the latent prints, with n = *100, 000*.

Latent Print "b115"			Latent Print "g73"		
N	$\hat{m}$	$p_\epsilon(\hat{m}, \mathbf{X})$	N	$\hat{m}$	$p_\epsilon(\hat{m}, \mathbf{X})$
16	2	0.73	39	4	1
	4	9.04×10^{-6}		8	3.11×10^{-14}
	8	2.46×10^{-19}		12	2.56×10^{-25}
	12	6.13×10^{-31}		24	3.10×10^{-52}
	16	1.82×10^{-46}		39	7.51×10^{-79}

Values of specific nPRC are largely dependent on given latent print

Dependence of specific nPRC of latent print "g73" with size of data set n parameterized by number of corresponding minutiae ^*m*.



When $\hat{m}$ decreases and n increases, PRC increases

Dual Representation

- Linear regression model

$$y(\mathrm{x},\mathrm{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathrm{x}) = \mathrm{w}^T \phi(\mathrm{x})$$

ϕ_j are basis functions or features

  - Parameters obtained by minimizing regularized *sum-of-squares:*

$$J(\mathrm{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{\mathrm{w}^T\phi(\mathrm{x}_n) - t_n\right\}^2 + \frac{\lambda}{2}\mathrm{w}^T\mathrm{w}$$

where $\mathrm{w}=(\mathrm{w}_0,\ldots,\mathrm{w}_{M-1})^T$, $\phi=(\phi_0,\ldots\phi_{M-1})^T$, N samples $\{\mathrm{x}_1,\ldots\mathrm{x}_N\}$, λ = regularization coefficient

  - Solution obtained by setting gradient of $J(\mathrm{w})$ wrt w equal to zero

$$\mathrm{w} = -\frac{1}{\lambda}\sum_{n=1}^{N}\left\{\mathrm{w}^T\phi(\mathrm{x}_n) - t_n\right\}\phi(\mathrm{x}_n) = \boxed{\sum_{n=1}^{N} a_n\phi(\mathrm{x}_n) = \Phi^T\mathrm{a}}$$

where Φ is the design matrix whose n^{th} row is given by $\phi(\mathrm{x}_n)^T$

$$\Phi = \begin{bmatrix} \phi_0(x_1) & . & . & \phi_{M-1}(x_1) \\ . & & & . \\ \phi_0(x_n) & . & . & \phi_{M-1}(x_n) \\ . & & & . \\ \phi_0(x_N) & . & . & \phi_{M-1}(x_N) \end{bmatrix} \text{ is a } N \times M \text{ matrix}$$

vector $\mathrm{a}=(a_1,\ldots,a_N)^T$ with

$$a_n = -\frac{1}{\lambda}\left\{\mathrm{w}^T\phi(\mathrm{x}_n) - t_n\right\}$$

  - Solution is a linear combination of vectors $\phi(\mathrm{x}_n)$ with weights a_n
  - Instead of working with parameter vector w we work with vector a

Transformation from Basis to Kernel

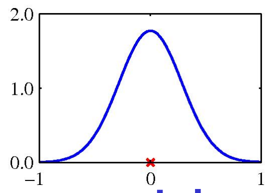
Gram Matrix Definition:
Given N vectors, it is an N x N matrix of inner products

- Substitute $\mathrm{w} = \Phi^T \mathrm{a}$ into $J(\mathrm{w})$
- with Gram matrix $\mathrm{K}=\Phi\Phi^{\mathrm{T}}$ elements

$$K_{nm} = \phi(\mathrm{x}_n)^{\mathrm{T}}\phi(\mathrm{x}_m) = k(\mathrm{x}_n,\mathrm{x}_m)$$

  - where we introduce the kernel function $k(\mathrm{x},\mathrm{x}') = \phi(\mathrm{x})^{\mathrm{T}}\phi(\mathrm{x}')$

    Example is Gaussian kernel $k(\mathrm{x},\mathrm{x}') = \exp(-\|\mathrm{x}-\mathrm{x}'\|^2/2\sigma^2)$



- Sum of squares error written in terms of Gram matrix as

$$J(\mathrm{a}) = \frac{1}{2}\mathrm{a}^T\mathrm{K}\mathrm{K}\mathrm{a} - \mathrm{a}^T K\mathrm{t} + \frac{1}{2}\mathrm{t}^T\mathrm{t} + \frac{\lambda}{2}\mathrm{a}^T K\mathrm{a}$$

whose solution is $\boxed{\mathrm{a} = (\mathrm{K} + \lambda \mathrm{I}_N)^{-1}\mathrm{t}}$

Solution for least squares problem expressed entirely in terms of kernel $k(\mathrm{x},\mathrm{x}')$

- Prediction for new input x

$$y(\mathrm{x}) = \mathrm{w}^T\phi(\mathrm{x}) = \mathrm{k}(\mathrm{x})^T(\mathrm{K} + \lambda I_N)^{-1}\mathrm{t}$$

where $\mathrm{k}(\mathrm{x})$ has elements $k_n(\mathrm{x}) = k(\mathrm{x}_n,\mathrm{x})$

  - Prediction is linear combination of kernel evaluated at training points
    - Need to invert N x N matrix
    - but avoid problems of very high (even infinite) dimensionality of x

Role of Gaussian Processes

1. As a kernel method
   - Duality leads to a non-probabilistic model for linear regression
   - Extending role of kernels to probabilistic discriminative models leads to Gaussian Processes (kernels arise naturally in a Bayesian setting)
2. As a Bayesian method
   - In Bayesian linear regression
     - Model has the form $y(\mathrm{x},\mathrm{w})=\mathrm{w}^T\phi(\mathrm{x})$ where a prior distribution over w induces a prior distribution over functions $y(\mathrm{x},\mathrm{w})$
     - Given training data set, we evaluate the posterior distribution over w which gives posterior distribution over regression functions
     - With noise it implies a predictive distribution $p(t|\mathrm{x})$ for a new input vector x
   - In Gaussian Processes
     - Dispense with parametric model and instead define a prior probability distribution over functions directly
     - For a finite training set we only need to consider values of the function at discrete set of input values x_n

Probabilistic Linear Regression Revisited

- Re-derive the predictive distribution by working in terms of distribution over functions $y(\mathrm{x},\mathrm{w})$
  - It will provide a specific example of a Gaussian Process
- Consider model with M fixed basis functions

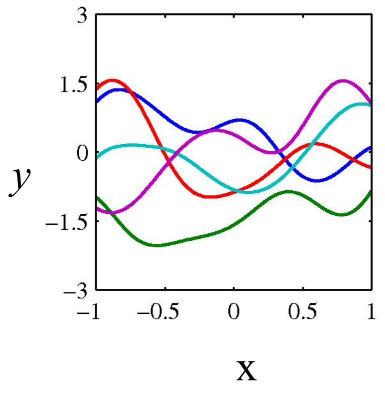
$$y(\mathrm{x})=\mathrm{w}^T\phi(\mathrm{x})$$

where x is the input vector and w is the M-dimensional weight vector

- Assume a Gaussian distribution of weight w

$$p(\mathrm{w}) = \mathcal{N}(\mathrm{w}|0,\alpha^{-1}\mathrm{I})$$

- Probability distribution over w induces a probability distribution over functions $y(\mathrm{x})$



Probabilistic Linear Regression is a Gaussian Process

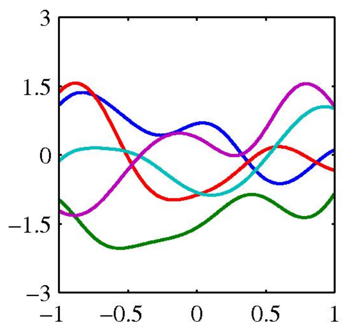
- We wish to evaluate y (x) at training points $x_1,.., x_N$
- Our interest is the joint distribution of values **y**= $[y(x_1),..y(x_N)]$

  Since $y(x)=w^T\phi(x)$ we can write

  $\mathbf{y} = \Phi w$ N x M times M x 1 yields a N x 1

  - where Φ is the design matrix with elements $\Phi_{nk}=\phi_k(x_n)$
- Since **y** is a linear combination of elements of w which are Gaussian distributed as $p(w) = \mathcal{N}(w|0,\alpha^{-1}I)$

  y is itself Gaussian with



mean: $E[\mathbf{y}] = \Phi E[w]=0$ and

variance: $\mathrm{Cov}[\mathbf{y}]=E[\mathbf{y}\mathbf{y}^T]=\Phi E[ww^T]\Phi^T=(1/\alpha)\Phi\Phi^T=K$

where K is the N x N Gram Matrix with elements

$$K_{nm}= k(x_n,x_m) = (1/\alpha)\phi(x_n)^T\phi(x_m)$$

and $k(x,x')$ is the kernel function

Covariance expressed by Kernel function

Thus Gaussian distributed weight vector induces a Gaussian joint distribution over training samples

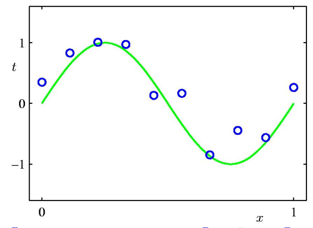
General definition of Gaussian Process

- We saw a particular example of a Gaussian process
  - Linear regression using the parametric model $y(\mathrm{x})=\mathrm{w}^T\phi(\mathrm{x})$
  - With samples **y**= $[y(\mathrm{x}_1),..y(\mathrm{x}_N)]$
  - Assume $p(\mathrm{w}) = \mathcal{N}(\mathrm{w}|0,\alpha^{-1}\mathrm{I})$
  - Then $\mathrm{E}[\mathbf{y}] = 0$ and $\mathrm{Cov}[\mathbf{y}]=\mathrm{K}$, the Gram matrix which is equivalent to pairwise kernel values
- More generally, a Gaussian process is a probability distribution over functions $y(\mathrm{x})$
  - Such that the set of values of $y(\mathrm{x})$ evaluated at arbitrary points $\mathrm{x}_1,..,\mathrm{x}_N$ jointly have a multivariate Gaussian distribution
    - For a single input x_1, output y_1 is univariate Gaussian. For two inputs $\mathrm{x}_1, \mathrm{x}_2$ the output y_1, y_2 is bivariate Gaussian, etc

Stochastic Process

- A *stochastic process* $y\,(\mathrm{x})$ is specified by giving the joint probability distribution for any finite set of values $y\,(\mathrm{x}_1),\ldots,y\,(\mathrm{x}_N)$ in a consistent manner
    - The random variables typically develop over time
- A Gaussian process is a stochastic process which is Gaussian
- When input x is 2-D it is also known as a *Gaussian Random Field*

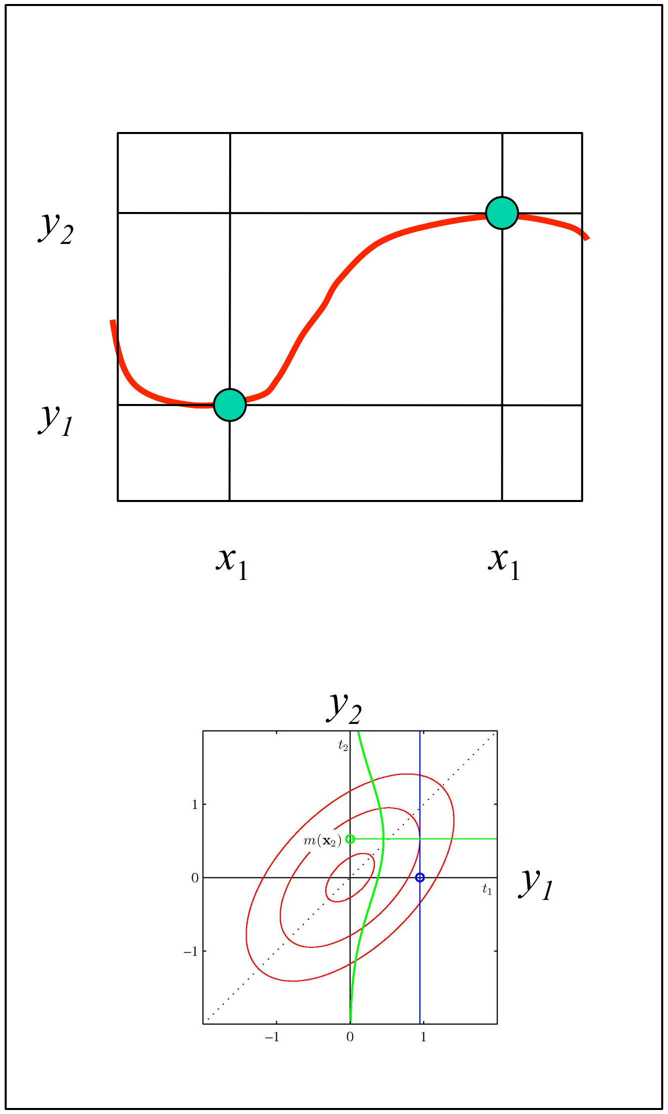
Parametric Regression vs GP Regression



- In parametric regression we have several samples from which we learn the parameters
- In Gaussian processes we we view the samples as one huge input that has a Gaussian distribution
  - with a mean and a covariance matrix
- A **Gaussian process** is a stochastic process X_t, $t \in T$, for which any finite linear combination of samples has a joint Gaussian Distribution
  - any linear functional applied to the sample function X_t will give a normally distributed result. Notation-wise, one can write $X \sim \mathrm{GP}(m,K)$, meaning the random function X is distributed as a GP with mean function m and covariance function K.

Gaussian Process with Two Samples

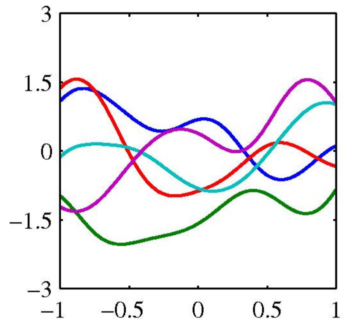
- Let y be a function (curve)
  - of a one-dimensional variable x
- We take two samples y_1 and y_2 corresponding to x_1 and x_2
- Assume they have a bivariate Gaussian distribution $p(y_1,y_2)$
- Each point from this distribution
  - has an associated probability
  - It also defines a function $y(x)$
    - Assuming that two points are enough to define a curve
- More than two points will be needed to define a curve
  - Which leads to a higher dimensional probability distribution



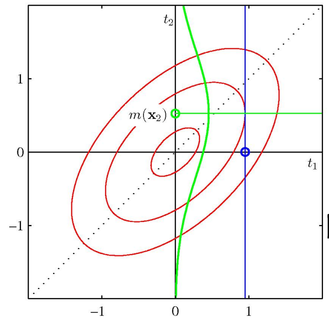
Gaussian Process with N samples

- A process that generates samples over time $\{y_1,..y_N\}$ is a GP iff every set of samples $Y=[y_1,..,y_N]$ is a vector-valued Gaussian random variable
- We define a distribution over all functions
  - Constrained by the samples
- The samples have a joint Gaussian distribution in N-dimensional space

Regression: Possible functions constrained by N samples



Case of two samples t_1 and t_2 that are bivariate Gaussian



Specifying a Gaussian Process

- Key point about Gaussian Stochastic Processes
  - Joint distribution over N variables $y_1,.., y_N$ is completely specified by the second-order statistics,
    - i.e., mean and covariance
- With mean zero, it is completely specified by covariance of $y(\mathrm{x})$ evaluated at any two values of x which is given by a kernel function

$$E[y(\mathrm{x}_n)\, y(\mathrm{x}_m)] = k(\mathrm{x}_n,\mathrm{x}_m)$$

- For the Gaussian Process defined by the linear regression model $y(\mathrm{x},\mathrm{w})=\mathrm{w}^T\phi(\mathrm{x})$ with prior $p(\mathrm{w}) = \mathcal{N}(\mathrm{w}|0,\alpha^{-1}\mathrm{I})$ the kernel function is

$$K_{nm}= k\,(\mathrm{x}_n,\mathrm{x}_m) = (1/\alpha)\, \phi(\mathrm{x}_n)^T\phi(\mathrm{x}_m)$$

Defining a Kernel Directly

- GP can also be specified directly by choice of kernel function (instead of indirectly by basis function)
- Samples of functions drawn from Gaussian processes for two different choices of kernel functions are shown next

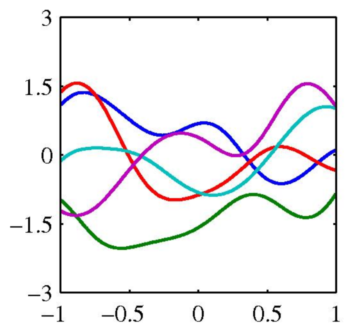
Samples from Gaussian Processes for two kernels

Functions are drawn from Gaussian processes

Note that each sample is a function

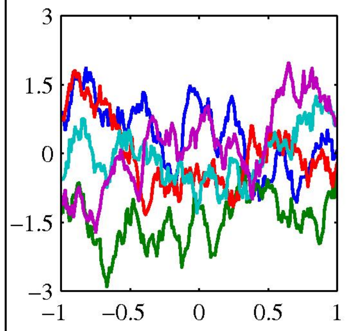
Gaussian Kernel

$$k(\mathrm{x},\mathrm{x}') = \exp\left(-\|\mathrm{x}-\mathrm{x}'\|^2/2\sigma^2\right)$$



Exponential Kernel

$$k(\mathrm{x},\mathrm{x}') = \exp\left(-\theta\,|\mathrm{x}-\mathrm{x}'|\right)$$



Ornstein-Uhlenbeck process for Brownian motion

Probability Distributions and Stochastic Processes

- A probability distribution describes the distributions of scalars (x) and vectors $(\mathbf{x})$
- A stochastic process describes distribution of functions $f(x)$
  - Can think of a function as a very long vector
    - Each entry in the vector is $f(x)$ which is the value of function at x
- A Gaussian process is a generalization of Gaussian distributions where it is describing the distribution of functions
- Since the vector is infinite-dimensional we constrain it to only those points x corresponding to training and test points

3. Gaussian Process for Regression (Direct Approach)

- We specify Gaussian Process directly over functions
  - Abandon approach of defining a distribution over weights $\mathbf{w}$

- Take into account noise on observed target values as

$$t_n = y_n + \varepsilon_n \quad \text{where } y_n = y(\mathbf{x}_n)$$

  - Noise process has a Gaussian distribution $p(t_n|y_n)=N(t_n|y_n,\beta^{-1})$

- Note that target t_n is output y_n corrupted by noise
- Defining $\mathbf{t} = (t_1,..,t_N)^T$ our goal is to determine a distribution $p(\mathbf{t})$
  - Which is a distribution over functions

3. Gaussian Process for Regression (Direct Approach)

- Assuming noise is independent for each data point
  - joint distribution *of* $\mathbf{t} = (t_1,..,t_N)^T$ on values $\mathbf{y} = (y_1,..,y_N)^T$ is

$$p(\mathbf{t}|\mathbf{y}) = N(\mathbf{t}|\mathbf{y}, \beta^{-1} I_N)$$

- From definition of GP, marginal distribution of $\mathbf{y}$ is given by
  - a Gaussian with zero mean, covariance matrix given by Gram matrix K

$$p(\mathbf{y}) = \mathrm{N}(\mathbf{y}|0,\mathrm{K})$$

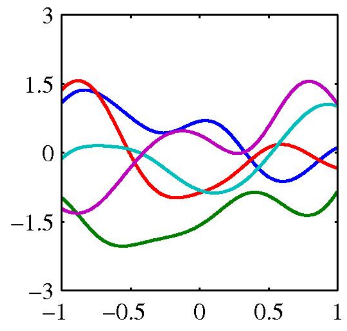
- Where kernel function that determines K is chosen to express:
  - property that for points x_n, x_m that are similar corresponding values $y(\mathrm{x}_n)$, $y(\mathrm{x}_m)$ will be more strongly correlated than for dissimilar points
  - K depends on application

Regression: Marginal Distribution

- From distributions $p(\mathrm{t}|\mathrm{y})$ and $p(\mathrm{y})$ we can get marginal $p(\mathrm{t})$ conditioned on input values $\mathrm{x}_1,..,\mathrm{x}_N$
- Applying Sum rule and product rule

$$p(\mathrm{t}) = \int p(\mathrm{t}\,|\,\mathrm{y})p(\mathrm{y})dy$$
$$= N(\mathrm{t}\,|\,0,C)$$

From the result that when $p(\mathrm{y})$ and $p(\mathrm{t}|\mathrm{y})$ are Gaussian $p(\mathrm{t})$ is also Gaussian



where covariance matrix C has the elements

$$C(\mathrm{x}_n,\mathrm{x}_m) = \underbrace{k(\mathrm{x}_n,\mathrm{x}_m)}_{\text{Due to } \mathrm{y}(\mathrm{x})} + \underbrace{\beta^{-1}\delta_{nm}}_{\text{Due to } \varepsilon}$$

- The two Gaussian sources of randomness, $y(\mathrm{x})$ and ε are independent, so their covariances simply add

δ specified by ε

Widely used kernel function for Gaussian Process

- Exponential of a quadratic form

  with addition of constant and linear terms

$$\boxed{k(\mathrm{x}_n,\mathrm{x}_m) = \theta_0 \exp\left\{-\frac{\theta_1}{2}\|\mathrm{x}_n - \mathrm{x}_m\|^2\right\} + \theta_2 + \theta_3 \underbrace{\mathrm{x}_n^T \mathrm{x}_m}}$$

Corresponds to a parametric model that is a linear function of input variables

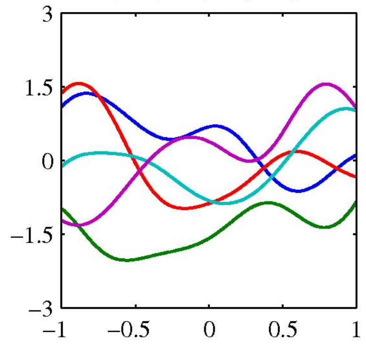
- Samples from this prior are plotted for various values of the parameters θ_0, θ_1, θ_2, and θ_3

Samples from a Gaussian Process Prior

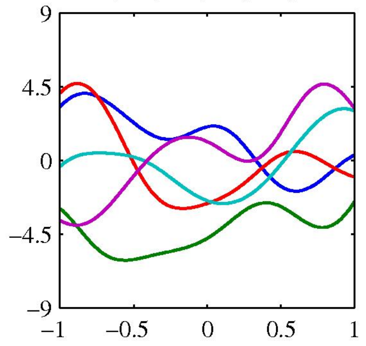
$$k(\mathrm{x}_n, \mathrm{x}_m) = \theta_0 \exp\left\{ -\frac{\theta_1}{2} \| \mathrm{x}_n - \mathrm{x}_m \|^2 \right\} + \theta_2 + \theta_3 \mathrm{x}_n^T \mathrm{x}_m$$

for different settings of $\theta_0, \theta_1, \theta_2, \theta_3$

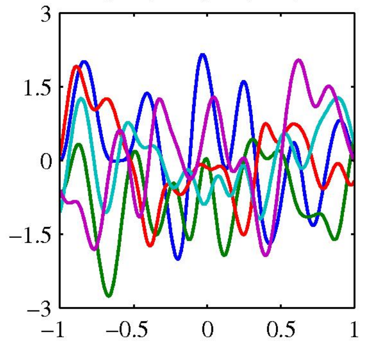
(1.00, 4.00, 0.00, 0.00)



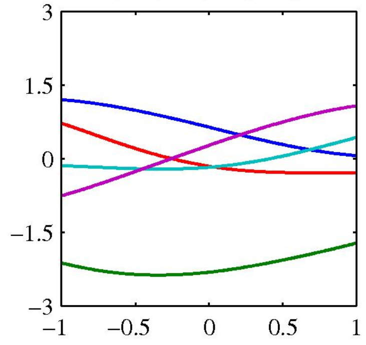
(9.00, 4.00, 0.00, 0.00)



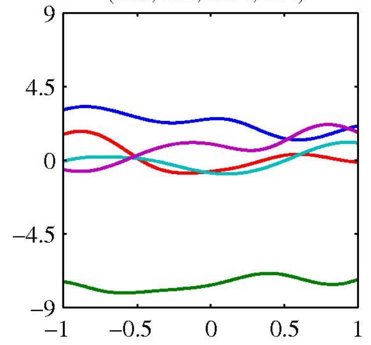
(1.00, 64.00, 0.00, 0.00)



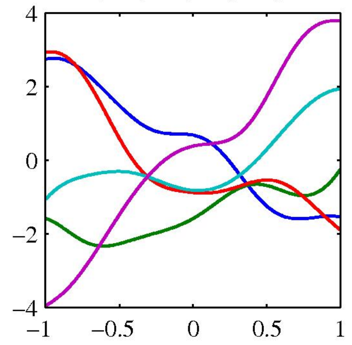
(1.00, 0.25, 0.00, 0.00)



(1.00, 4.00, 10.00, 0.00)



(1.00, 4.00, 0.00, 5.00)



Each function is a sample

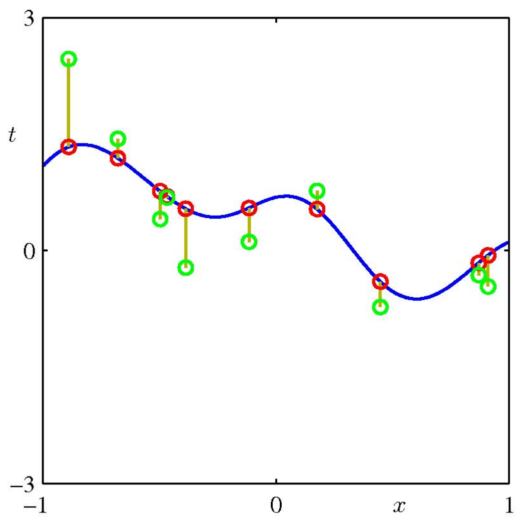
Illustration of sampling data points from a Gaussian Process

- A sample from joint distribution $p(\mathrm{y}) = \mathrm{N}(\mathrm{y}|0,\mathrm{K})$
  - where $\mathrm{y} = [y_1,.. y_N]^{\mathrm{T}}$
  - Each y_i is associated with a value of x since the elements of y are ordered
- Noise values are added as defined by $p(\mathrm{t}) = \mathrm{N}(\mathrm{t}|0,\mathrm{C})$

Blue curve:
a sample function from the Gaussian process prior over functions

Red points:
values of y_n obtained by evaluating function at set of input values $\{x_n\}$

Green points:
values of t_n obtained by adding independent Gaussian noise to each of y_n



Making predictions using Gaussian Processes

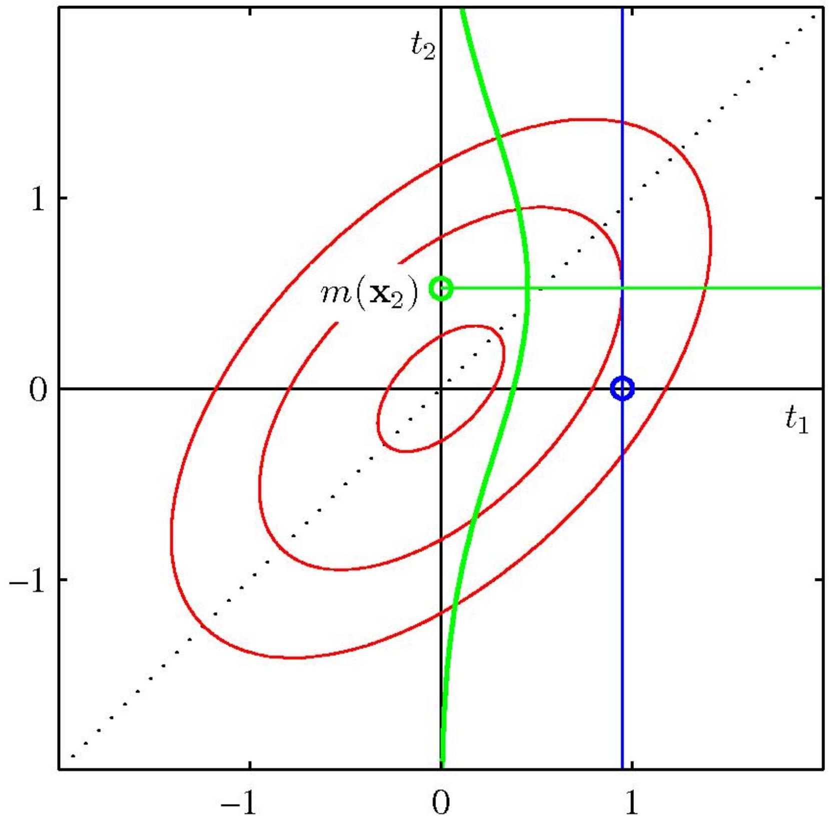
- Goal of regression: predict target for new inputs
  - given training data $\mathbf{t}_N = (t_1,..,t_N)^T$
  - with corresponding input values $(\mathrm{x}_1,..\mathrm{x}_N)$
- Goal is to predict target variable t_{N+1} given $\mathbf{x}_{N+1}$
- This requires we evaluate the predictive distribution

  $p(t_{N+1}|\mathbf{t}_N)$ which is also conditioned on $\mathrm{x}_1,..,\mathrm{x}_N$ and x_{N+1} which are not shown to keep notation simple
- To find conditional distribution $p(t_{N+1}|\mathbf{t}_N)$ we begin by writing down the joint distribution $\mathrm{p}(\mathbf{t}_{N+1})$
- We then obtain the required conditional distribution

Mechanism for Gaussian process regression

- One training point and one test point
- Red ellipses show contours of $p(t_1,t_2)$
- Green curve is $p(t_2|t_1)$
- t_1 is the training point
- Conditioning on t_1 corresponding to blue line
- Predictive distribution for t_{N+1} is Gaussian whose mean and variance both depend on $\mathbf{x}_{N+1}$

Joint distribution of $\mathbf{t}_2=(t_1,t_2)^T$



Finding Conditional Distribution $p(t_{N+1}|\mathbf{t})$

- Joint distribution is given by

$$p(\mathbf{t}_{N+1}) = \mathcal{N}(\mathbf{t}_{N+1}|\mathbf{0}, \boldsymbol{C}_{N+1})$$

  - where $\mathbf{C}_{N+1}$ is the $(N+1)$ *x* $(N+1)$ covariance matrix with elements given by $C(\mathrm{x}_n,\mathrm{x}_m) = k(\mathrm{x}_n,\mathrm{x}_m) + \beta^{-1} d_{nm}$
- Because the joint is Gaussian, conditional Gaussian distribution is given by partitioning the covariance matrix as

$$\mathbf{C}_{N+1} = \begin{pmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^T & c \end{pmatrix}$$

where $\boldsymbol{C}_N$ is the N *x* N covariance matrix

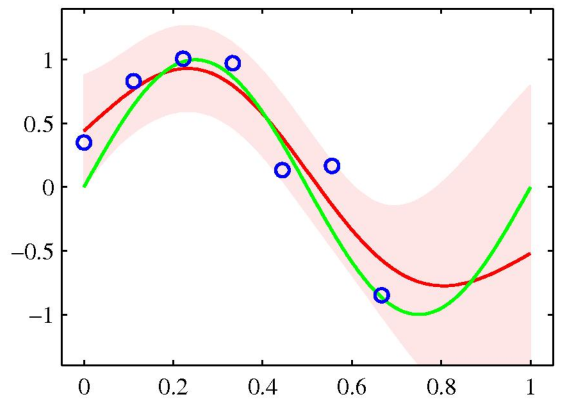
vector k has elements $k(\mathrm{x}_n,\mathrm{x}_{N+1})$ for $n=1,.., N$

and scalar $\mathrm{c} = k(\mathrm{x}_{N+1},\mathrm{x}_{N+1})+\beta^{-1}$

- Conditional distribution $p(t_{N+1}|\mathbf{t})$ is Gaussian with
  - Mean $m(\mathrm{x}_{N+1}) = \mathrm{k}^T\mathbf{C}_N^{-1}\mathbf{t}$
  - Variance $\sigma^2(\mathrm{x}_{N+1}) = \mathrm{c} - \mathrm{k}^T\mathbf{C}_N^{-1}\mathrm{k}$
- Key results that define Gaussian Regression

Gaussian process regression applied to sinusoidal data set

- Green: sinusoidal function from which data points are obtained
- Red: mean of the Gaussian process predictive distribution
- Shaded region: plus and minus two standard deviations
- Uncertainty increases to right of data points



Summary of Gaussian Process Regression

- A Gaussian Process is a generalization of a multivariate Gaussian distribution to infinitely many variables
- A Gaussian *distribution* is fully specified by a mean vector μ and covariance matrix Σ

  $\mathrm{f} = (f_1,..f_n)^{\mathrm{T}} \sim \mathcal{N}(\mu,\Sigma)$ indexes $i = 1,..n$

- A Gaussian *process* is fully specified by a mean function $m(\mathrm{x})$ and covariance function $k(\mathrm{x},\mathrm{x}')$

  $f(x) \sim \mathcal{GP}(m(\mathrm{x}), k(\mathrm{x},\mathrm{x}'))$ indexes x *(which are functions, or infinite vectors)*

- Note that the kernel function k appears in place of the covariance matrix
  - Both express similarity of two points in multidimensional space
  - Note: correlation is same as covariance when all variables are zero-mean, unit variance

The marginalization property

- Thinking of a GP as a Gaussian distribution with an infinitely long mean vector and an infinite by infinite covariance matrix may seem impractical
- Luckily we have the marginalization property

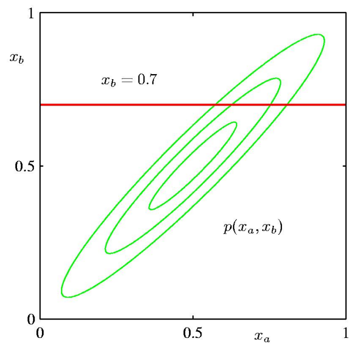
$$p(\mathrm{x}) = \int p(\mathrm{x},\mathrm{y})\,dy$$

$$\mathrm{x} = \begin{bmatrix} \mathrm{x}_a \\ \mathrm{x}_b \end{bmatrix}$$

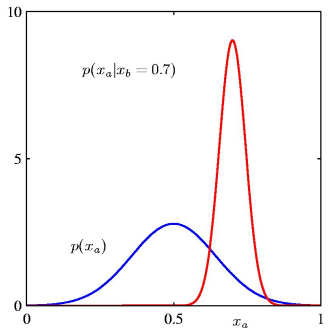
$$p(\mathrm{x}_a, \mathrm{x}_b) = N\left(\begin{bmatrix} \mu_a \\ \mu_b \end{bmatrix}, \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}\right)$$

$$\Rightarrow p(\mathrm{x}_a) = N(\mu_a, A)$$

Joint is Gaussian



Marginal and Conditional are Gaussian



- Marginalizing over unknown function values x_b
- To get distribution over known values x_a
- Similarly conditional distribution

$$\Rightarrow p(\mathrm{x}_a|\mathrm{x}_b) = N(\mu_a\text{-}\mathrm{A}^{-1}B(x_b - \mu_b), A^{-1})$$

Predictive distribution

- Mean $m(\mathrm{x}_{N+1}) = \mathrm{k}^T \mathbf{C}_N^{-1} \mathbf{t}$
- Variance $\sigma^2(\mathrm{x}_{N+1}) = \mathrm{c} - \mathrm{k}^T \mathbf{C}_N^{-1} \mathrm{k}$

  where $\boldsymbol{C}_N$ is the N ***x*** N covariance matrix with elements $k(\mathrm{x}_n,\mathrm{x}_m) + \beta^{-1}\delta_{nm}$

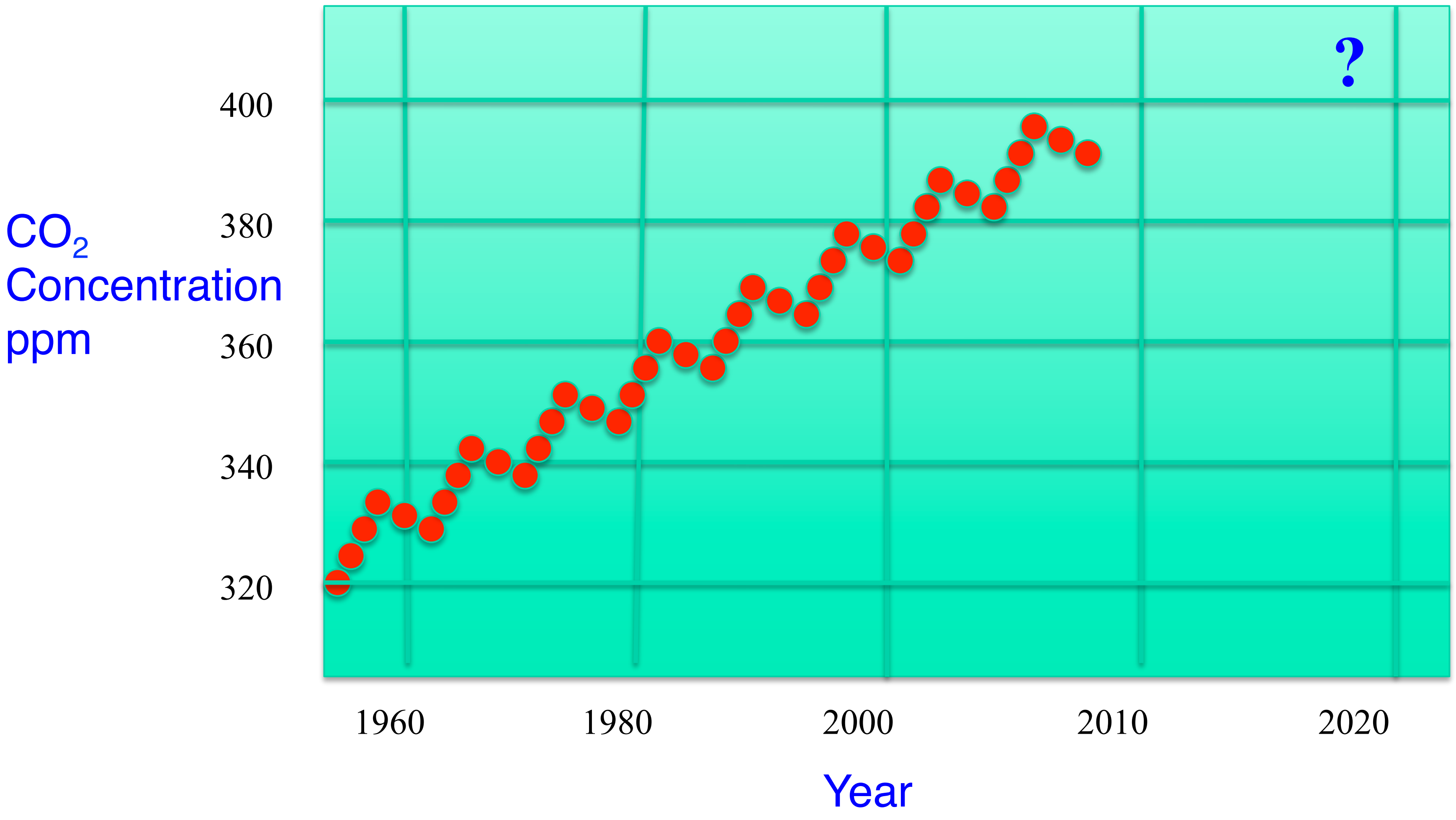
  vector k has elements $k(\mathrm{x}_n,\mathrm{x}_{N+1})$ for $n=1,..,N$

  and scalar $\mathrm{c} = k(\mathrm{x}_{N+1},\mathrm{x}_{N+1}) + \delta^{-1}$
- Same restrictions on kernel function as in general kernel methods (positive definite)
- Mean can also be written as

$$m(\mathrm{x}_{N+1}) = \sum_{n=1}^{N} a_n k(\mathrm{x}_n, \mathrm{x}_{N+1})$$

  - Where a_n is the nth component of $\mathrm{C}_N^{-1}\mathrm{t}$
  - If $k(\mathrm{x}_n,\mathrm{x}_m)$ depends only on distance $||\mathrm{x}_n - \mathrm{x}_m||$ then we obtain an expansion in radial basis functions

Regression Problem: Carbon Dioxide in Atmosphere

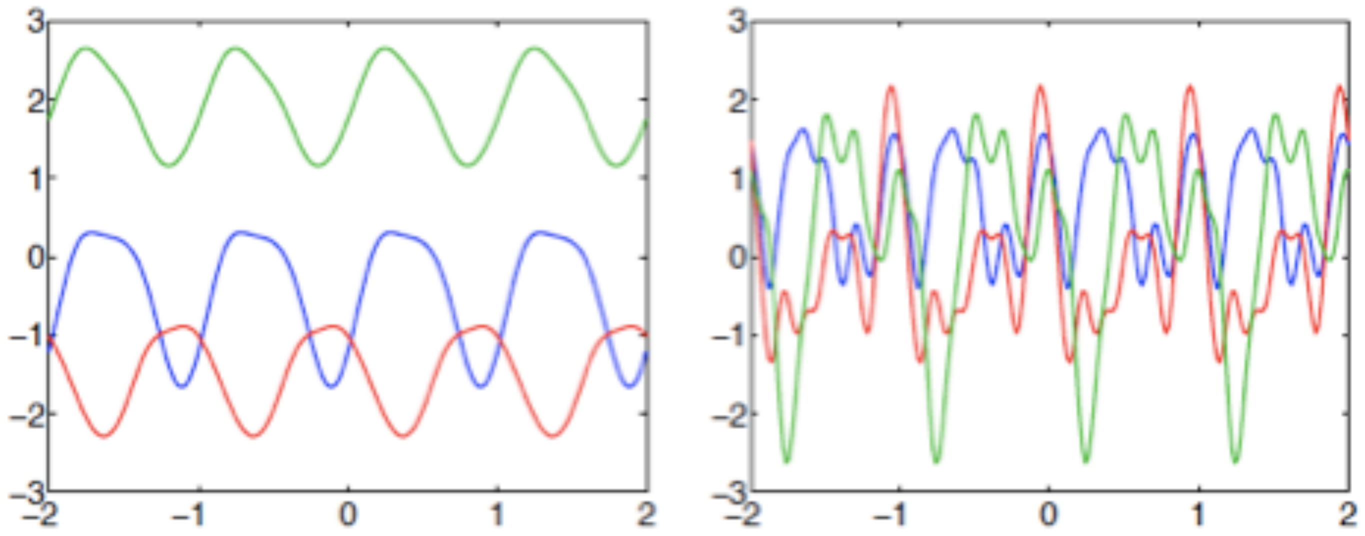


Periodic Smooth Functions

To create distribution over periodic functions of x:

- First map inputs to $u=(\sin x, \cos x)^T$
- Then measure distances in u-space
- Combined with squared exponential (SE) covariance function with characteristic length ale l we get

$$k_{periodic}(x,x') = \exp\left(\frac{-2\sin^2(\pi(x-x'))}{l^2}\right)$$



Three functions drawn at random: left $l > 1$, right $l < 1$

Covariance Function for Periodic Data

The covariance function consists of several terms, parameterized by 11 hyperparameters

$$k(x,x') = k_1(x,x') + k_2(x,x') + k_3(x,x') + k_4(x,x')$$

Where

Long-term smooth trend (squared exponential)

$$k_1(x,x')=\mathrm{q}_1{}^2\exp(-(x-x')^2/\mathrm{q}_2{}^2$$

Seasonal trend (quasi-periodic smooth)

$$k_2(x,x')=q_3{}^2 exp(-2sin^2[p(x-x')]/q_s{}^2 \ \mathbf{x}\ exp(-(1/2)(x-x')^2/q_4{}^2),$$

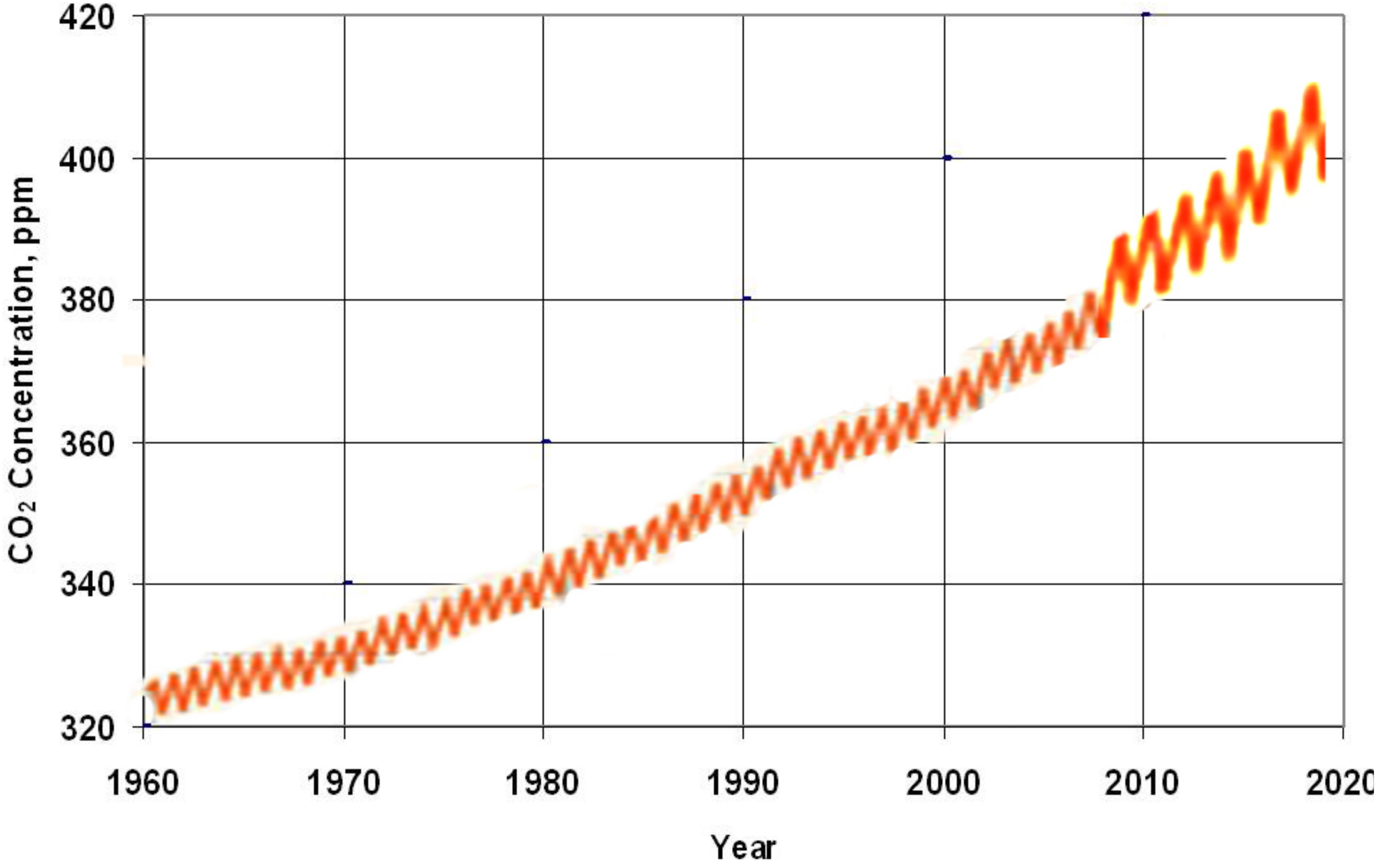
Short-term and medium-term anomaly (rational quadratic)

$$k_3(x,x') = \theta_6^2\left(1 + \frac{(x-x')^2}{2\theta_8\theta_7}\right) - \theta_8$$

Noise (independent Gaussian and dependent)

$$k_4(x,x') = \theta_9^2 \exp\left(-\frac{(x-x')2}{2\theta_{10}}\right) + \theta_{11}^2\delta_{xx}{}'$$

Gaussian Process Fit to CO_2 data



Computational Comparison

- Method 1: Parameter Space viewpoint using linear regression
- Method 2: Function space viewpoint using Gaussian process
  - Inversion of matrix of size N x N
    - Standard methods require $O(N^3)$ computations
  - In the basis function model we have to invert a matrix S_N of size M x M which has $O(M^3)$ complexity
- When $M < N$ basis function approach is more efficient
  - For large training sets many approximate methods for GP
- Gaussian process allows considering covariance functions that can only be expressed in terms of an infinite number of basis functions

4. Learning the hyperparameters

- Prediction of a Gaussian process model will depend on the choice of covariance function
- Rather than fixing the covariance function we can use a parametric family of functions and then infer parameter values from the data
- Parameters govern
  - Length-scale of the correlations
    - Region over which variables are correlated
  - Precision of the noise
- They correspond to the hyper-parameters in a standard parametric model

Techniques for Learning Hyperparameters

- Based on evaluation of likelihood function $p(\mathrm{t}|\theta)$
  - where θ denotes the hyperparameters of the Gaussian process model
- Point estimate of θ is obtained by maximizing log-likelihood

$$\ln p(\mathrm{t}\,|\,\theta) = -\frac{1}{2}\ln|C_N| - \frac{1}{2}\mathrm{t}^T C_N^{-1}\mathrm{t} - \frac{N}{2}\ln(2\pi)$$

- Gradient of log-likelihood

$$\frac{\partial}{\partial\theta_i}\ln p(\mathrm{t}\,|\,\theta) = -\frac{1}{2}Tr\left(C_N^{-1}\frac{\partial C_N}{\partial\theta_i}\right) + \frac{1}{2}\mathrm{t}^T C_N^{-1}\frac{\partial C_N}{\partial\theta_i}C_N^{-1}\mathrm{t}$$

  - Can have multiple maxima
- Can be treated in a Bayesian manner by introducing a prior over θ
- We have assumed a constant β. For heteroscedastic problems noise variance itself depends on x
  - Can be handled by introducing a second Gaussian process

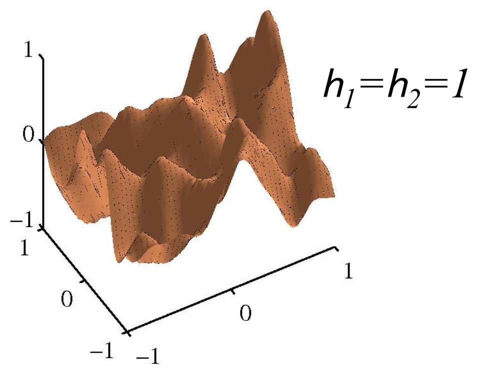
5. Automatic Relevance Determination

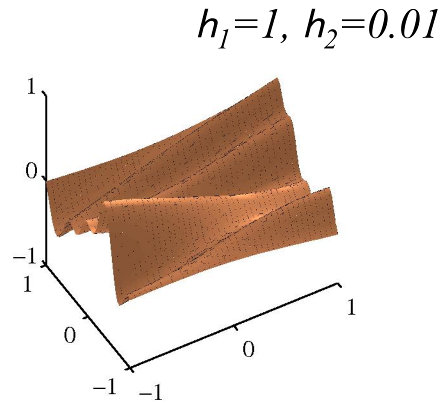
- Determining importance of variables
- Consider a Gaussian process with 2-D input space $\mathrm{x}=(x_1,x_2)$ having kernel function

$$k(x,x') = \theta_0 \exp\left\{-\frac{1}{2}\sum_{i=1}^{2}\eta_i(x_i - x_i')^2\right\}$$

- As particular parameter η_i becomes small
  - function becomes insensitive to corresponding variable x_i
- By adapting these parameters to a data set by mle it becomes possible to detect variables that have little effect on predictive distribution
  - Such variables can be discarded

Samples from the ARD prior for Gaussian processes



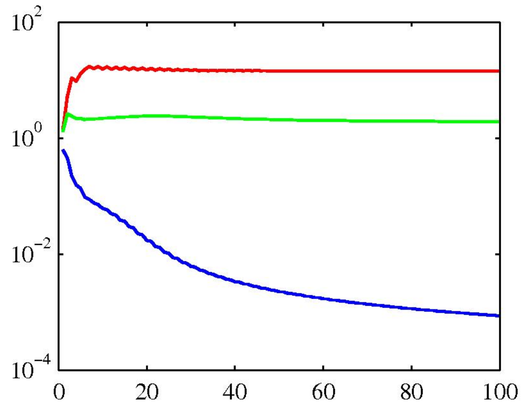


Automatic Relevance Determination in a Gaussian Process for Synthetic Problem

- Three inputs x_1, x_2 and x_3
- Target variable t generated by sampling 100 values of x_1 from a Gaussian, evaluating $sin\ (2\pi x_1)$ and adding Gaussian noise
- x_2 by adding noise to x_1
- x_3 are samples from an independent Gaussian
- Thus

  x_1 is a good predictor of t
  x_2 is a more noisy predictor
  x_3 has only chance correlations with t

Values for hyperparameters h_1, h_2, h_3 as a function of number of iterations (on log scale)



h_1 converges to a large value
h_2 converges to a much smaller value
h_3 becomes small indicating it is irrelevant for predicting t

6. Gaussian Processes for Classification

- Gaussian processes make predictions that lie on the entire real axis
- For two-class classification we need to model posterior probabilities of the target variable for a new input variable to lie in the interval $(0,1)$
- Can easily adapt Gaussian processes for classification
    - by transforming output of Gaussian process using appropriate nonlinear activation function

Transforming Gaussian process output for classification

- Two class problem with target variable $t \in \{0,1\}$
- We define a Gaussian process over a function $a(\mathrm{x})$
- Then transform the function using a logistic sigmoid

$$y = \sigma(a)$$

- Then we obtain a non-Gaussian stochastic process over functions $y(\mathrm{x})$ where $y \in \{0.1\}$

Two-class Classification

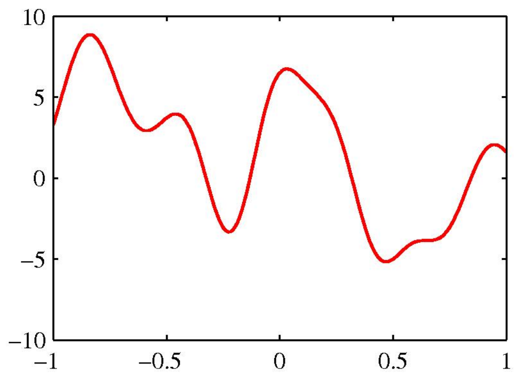
One-dimensional input space

Bernoulli distribution

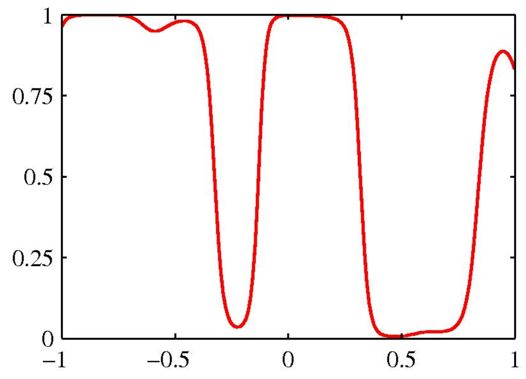
$$p(t \mid a) = \sigma(a)^t (1 - \sigma(a))^{1-t}$$

Transforming output on real line to $(0,1)$ interval

A sample from a Gaussian process prior over functions $a(\mathrm{x})$



Result of transforming this sample using a logistic sigmoid function



Next: How to define the Gaussian Process

Gaussian Process for Classification

- Training set samples $\mathrm{x}_1 \ldots \mathrm{x}_N$
- Corresponding target variables $\mathbf{t} = (t_1,..,t_N)^T$
- Test point x_{N+1} with target value t_{N+1}
- Goal is to determine the predictive distribution $p(t_{N+1}|\mathbf{t})$
  - where conditioning on input variable is left implicit
- Define a Gaussian process over function $a(\mathbf{x})$
- $\mathbf{a}_{N+1}$ has the components $a(\mathrm{x}_1),..,a(\mathrm{x}_{N+1})$
- Gaussian process prior takes the form

  $p(\mathbf{a}_{N+1})=N(\mathbf{a}_{N+1}|0,C_{N+1})$

  Unlike regression the covariance matrix no longer includes a noise term since all of the training data points are assumed to be correctly labeled

  Covariance matrix has elements $C(\mathrm{x}_n,\mathrm{x}_m)=k(\mathrm{x}_n,\mathrm{x}_m)+\nu\delta_{nm}$
- Predictive distribution is intractable

$$p(t_{N+1}=1|\mathbf{t}_N) = \int p(t_{N+1}=1|a_{N+1})p(a_{N+1}|\mathbf{t}_N)da_{N+1}$$

Three Approximation Methods for Gaussian Process Classification

1. Variational Inference
   - Uses local variational bound on logistic sigmoid
   - Allows product of sigmoids to be approximated by product of Gaussians thereby allowing marginalization over a_N to be performed analytically
2. Expectation Propagation
   - Because true posterior is unimodal, can get good results
3. Laplacian Approximation

7. Laplace Approximation

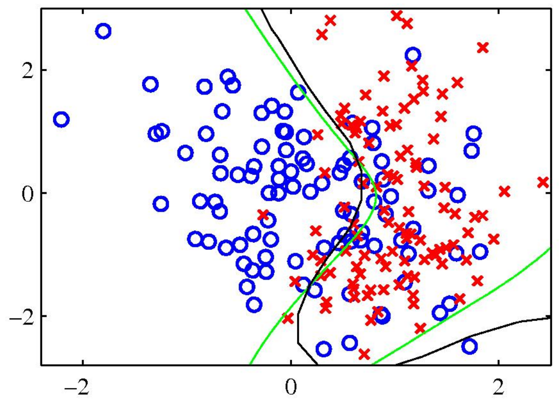
- We seek a Gaussian approximation to the posterior distribution over a_{N+1}

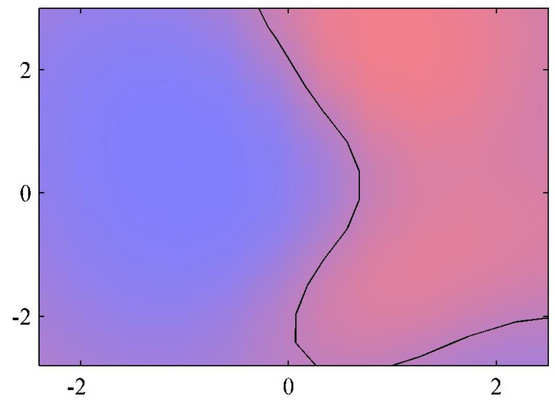
Illustration of Gaussian process for classification

Using Laplace Approximation

Green: Optimal boundary
Black: Gaussian Process Classifier

Predicted posterior probability together with Gaussian process decision boundary





8. Connection to Neural Networks

- Range of functions represented by a neural network is governed by M, number of hidden units
- In a Bayesian neural network
  - Prior distribution over paramter vector w produces a prior distributiion over functions from $y(\mathrm{x})$
  - As $M \rightarrow oo$ distribution of functions generated will tend to a Gaussian process
- Neural network outputs share the hidden units thus borrowing statistical strength from each other
  - i.e., weights associated with each hidden unit is influenced by all output variables, not just by one of them
  - This property is lost in the Gaussian process
- For non-stationary kernels
  - ie, $k(\mathrm{x},\mathrm{x}')$ cannot be expressed as a function of $\mathrm{x}-\mathrm{x}'$
  - breaks translation invariance in neural network weight space