ASSIGNMENT 1

J. Elder

Overview

- In the visual world, objects are often obscured or occluded by intervening objects, resulting in fragmented boundaries and a loss of shape information.
- One advantage of generative models is that they can fill in missing data based upon partial observation. In the context of our problem, this means that the missing portion of the boundary can be estimated. This is the problem of *shape completion.*
- We will evaluate our models by occluding a contiguous 10% portion of an object boundary, and then using our models to estimate these missing data.

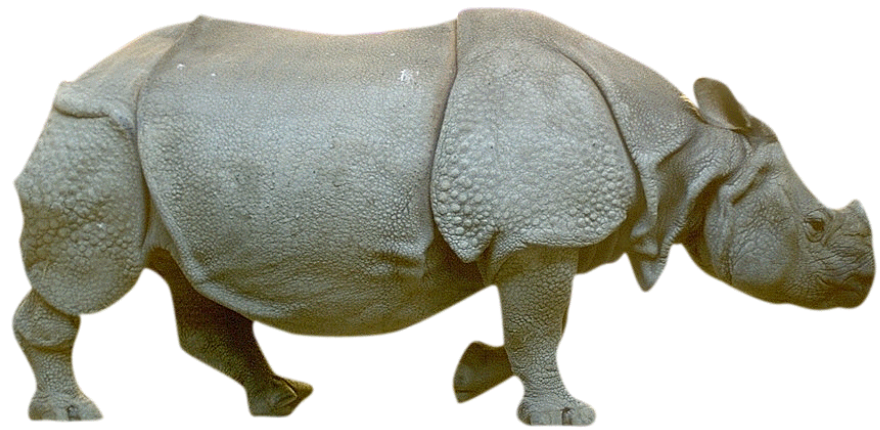
Dataset

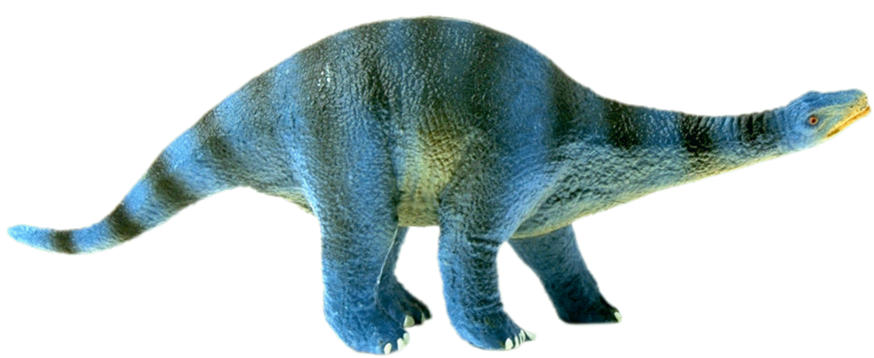
- The dataset is drawn from the Hemera database of 150,000 blue-screened photo-objects.
- From these I have selected 350 animal objects and randomly partitioned them into training and test datasets of 175 objects each.
- The boundary of each object has been down-sampled to a vector of $D = 128$ points. Each point of a shape is a 2D Euclidean coordinate. We represent this as a complex number $x + iy$. The data and code I provide uses this representation.
- Each shape has been normalized to a unit circle using a Procrustes transformation. This means:
  - There is a 1:1 correspondence between the 128-element vectors representing each shape, which facilitates analysis.
  - The expected position of a point on a shape is given by the corresponding point on the unit circle:

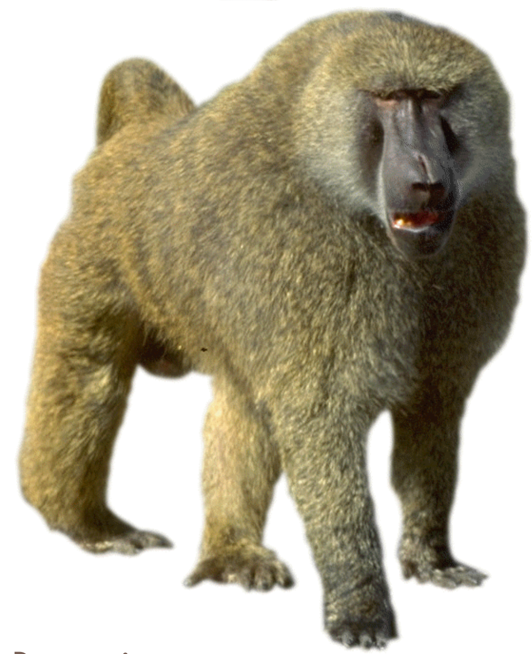
$$E\left[\left(x_i, y_i\right)\right] = \left(\cos\theta_i, \sin\theta_i\right), \text{ where } \theta_i = \frac{2\pi i}{D}$$

- You can access the training dataset now from the course website.

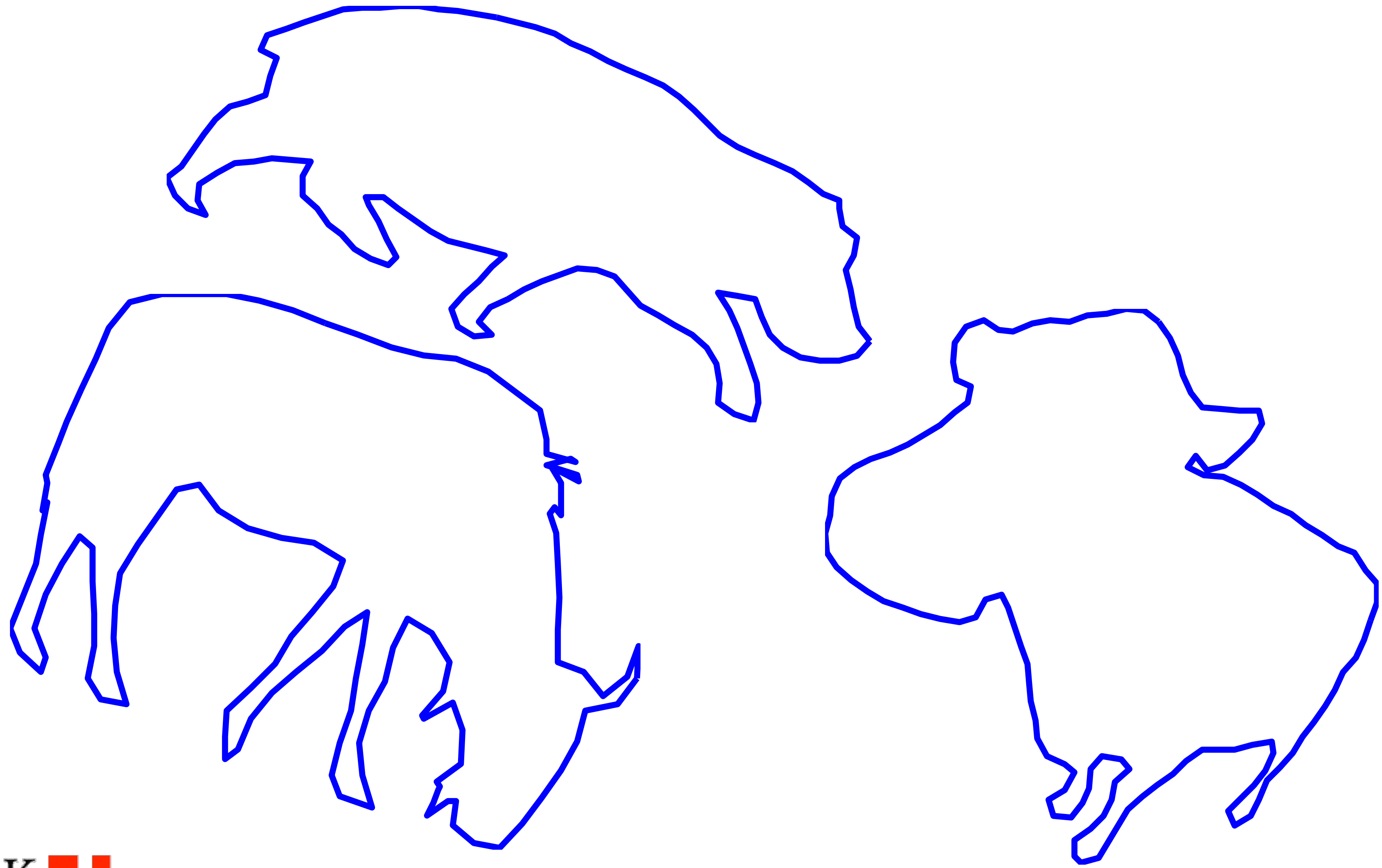
Animal Objects







Polygon Approximations



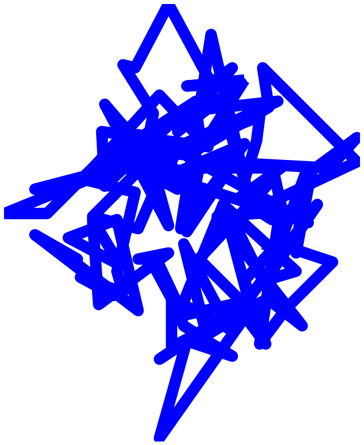
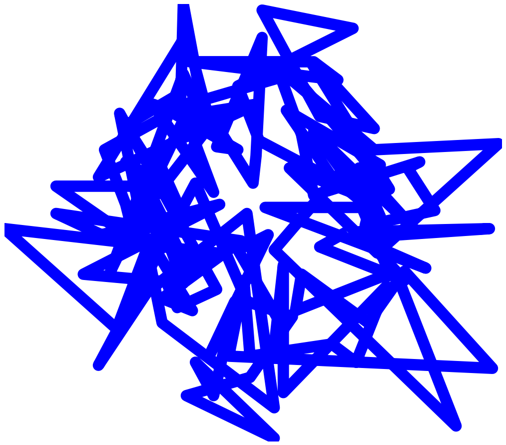
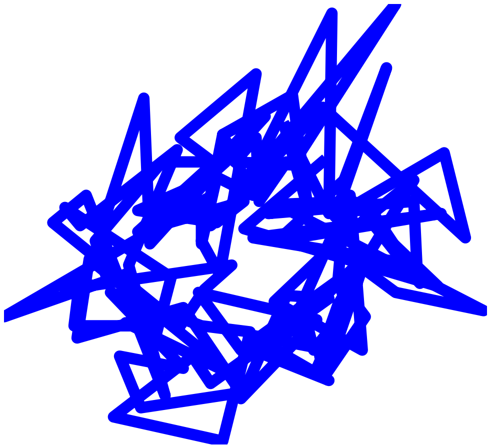
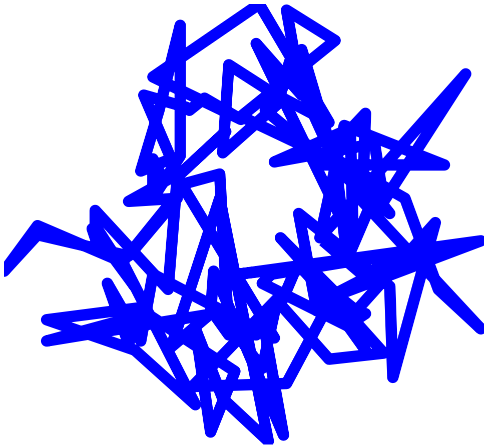
Shape Models

- I have provided code for 3 models. You will invent more.

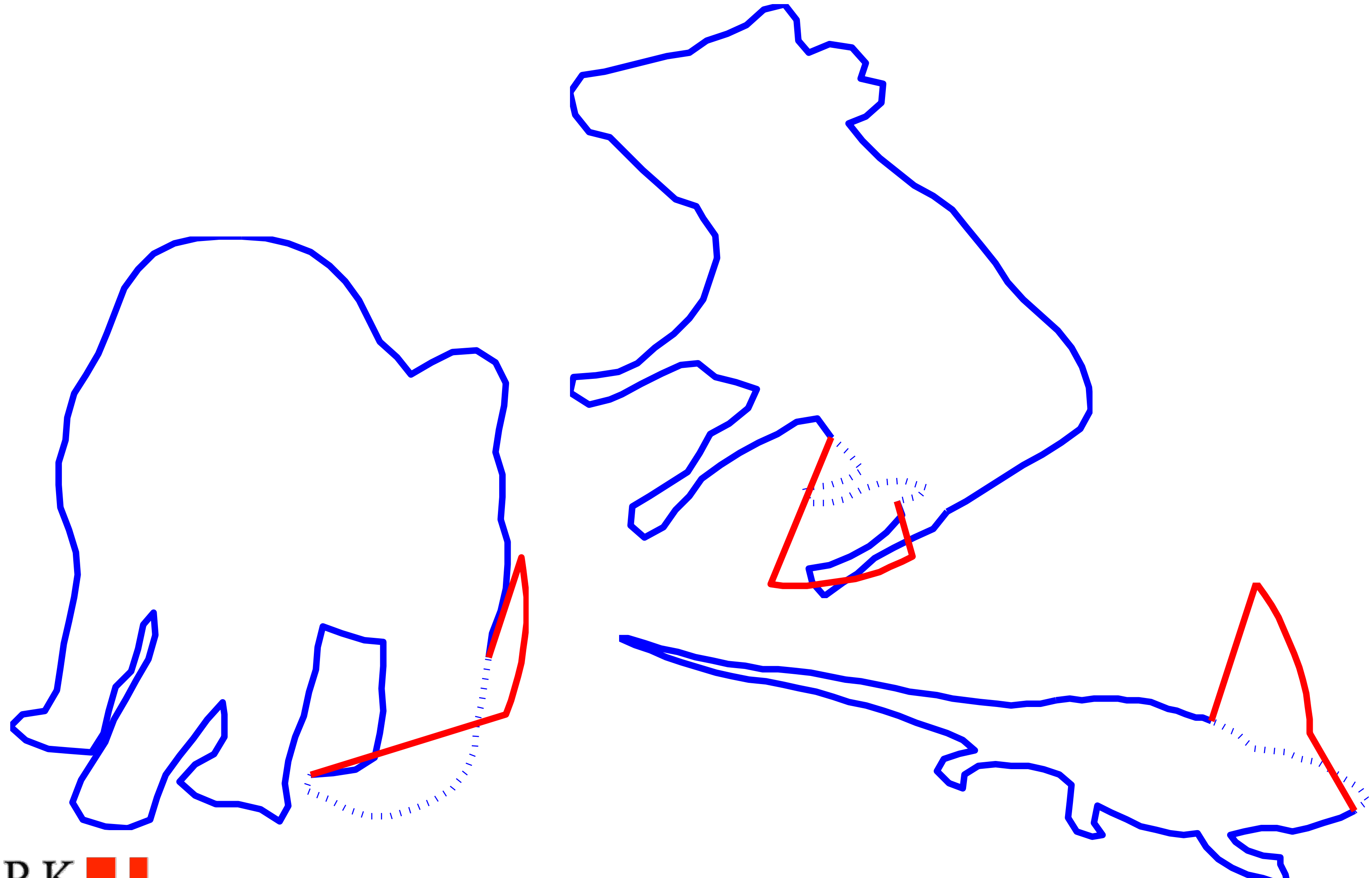
Shape Model 1

- This is a very simple generative model that assumes shape vectors are drawn from an isotropic multivariate normal distribution. (In other words the covariance matrix is a diagonal matrix with a constant diagonal.) There is a single scalar parameter: the variance.
- Functions:
  - ShapeModel1ML.m - computes maximum likelihood estimate of the parameter
  - ShapeModel1Sample.m - generates and displays random samples from the model
  - ShapeModel1Complete.m - estimates missing portion of a given shape

Shape Model 1 Samples

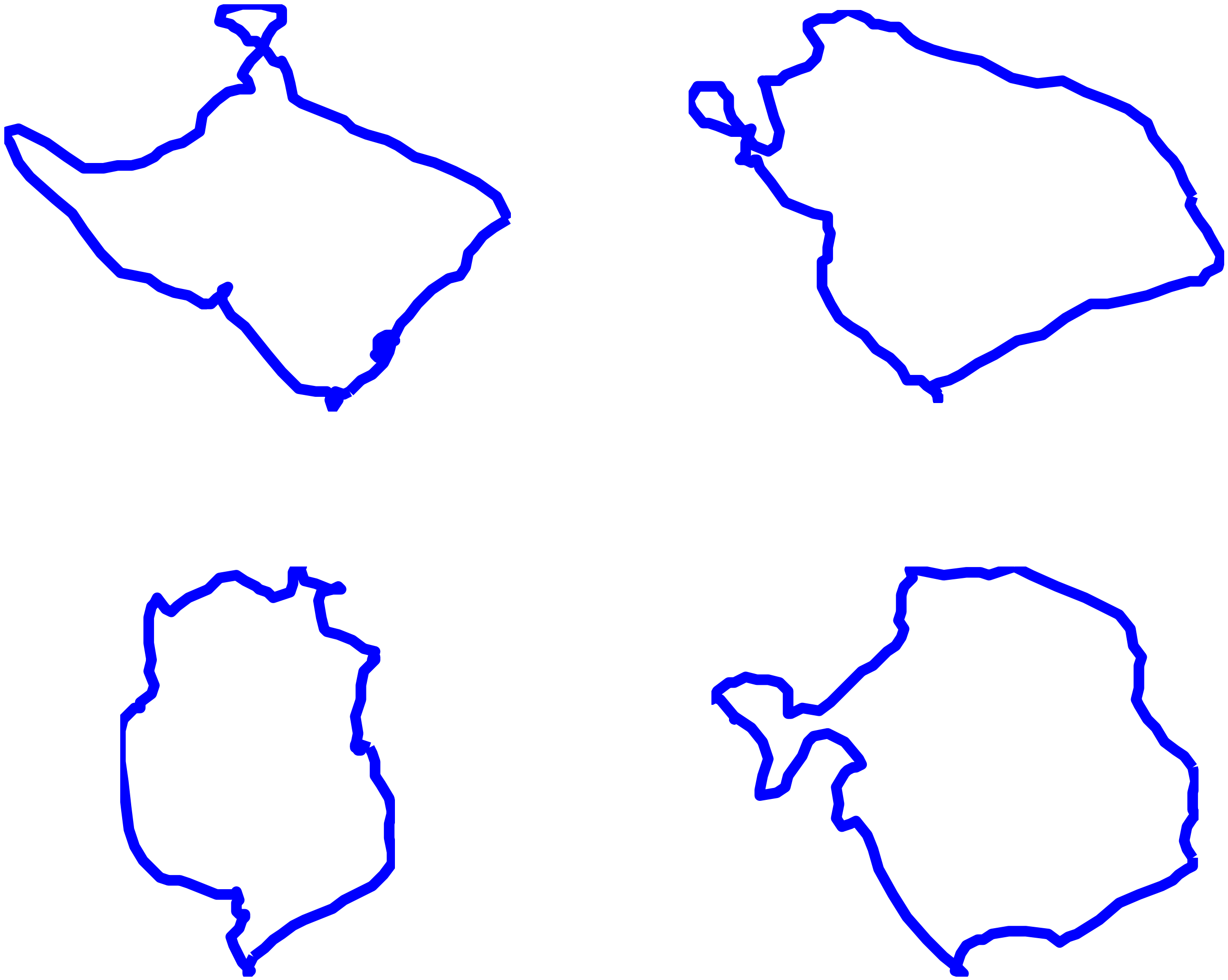
Shape Model 1 Shape Completions



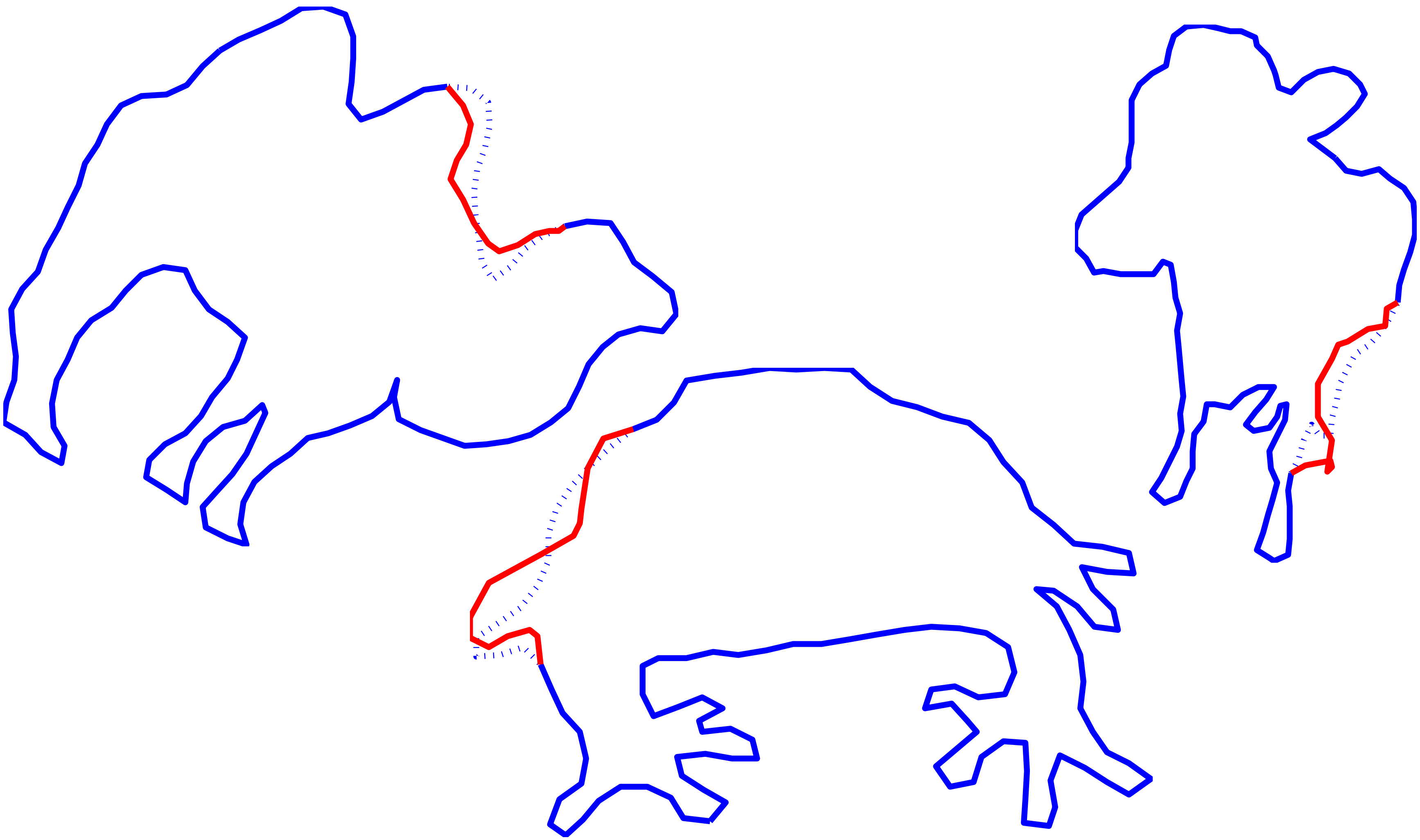
Shape Model 2

- In this generative model, shape vectors are assumed to be samples from a general multivariate normal distribution. There is only one parameter, the covariance matrix, but this represents $D(D+1)/2$ degrees of freedom (i.e., scalar unknowns).
- Functions:
  - ShapeModel2ML.m - computes maximum likelihood estimate of the parameters
  - ShapeModel2Sample.m - generates and displays random samples from the model
  - ShapeModel2Complete.m - estimates missing portion of a given shape

Shape Model 2 Samples



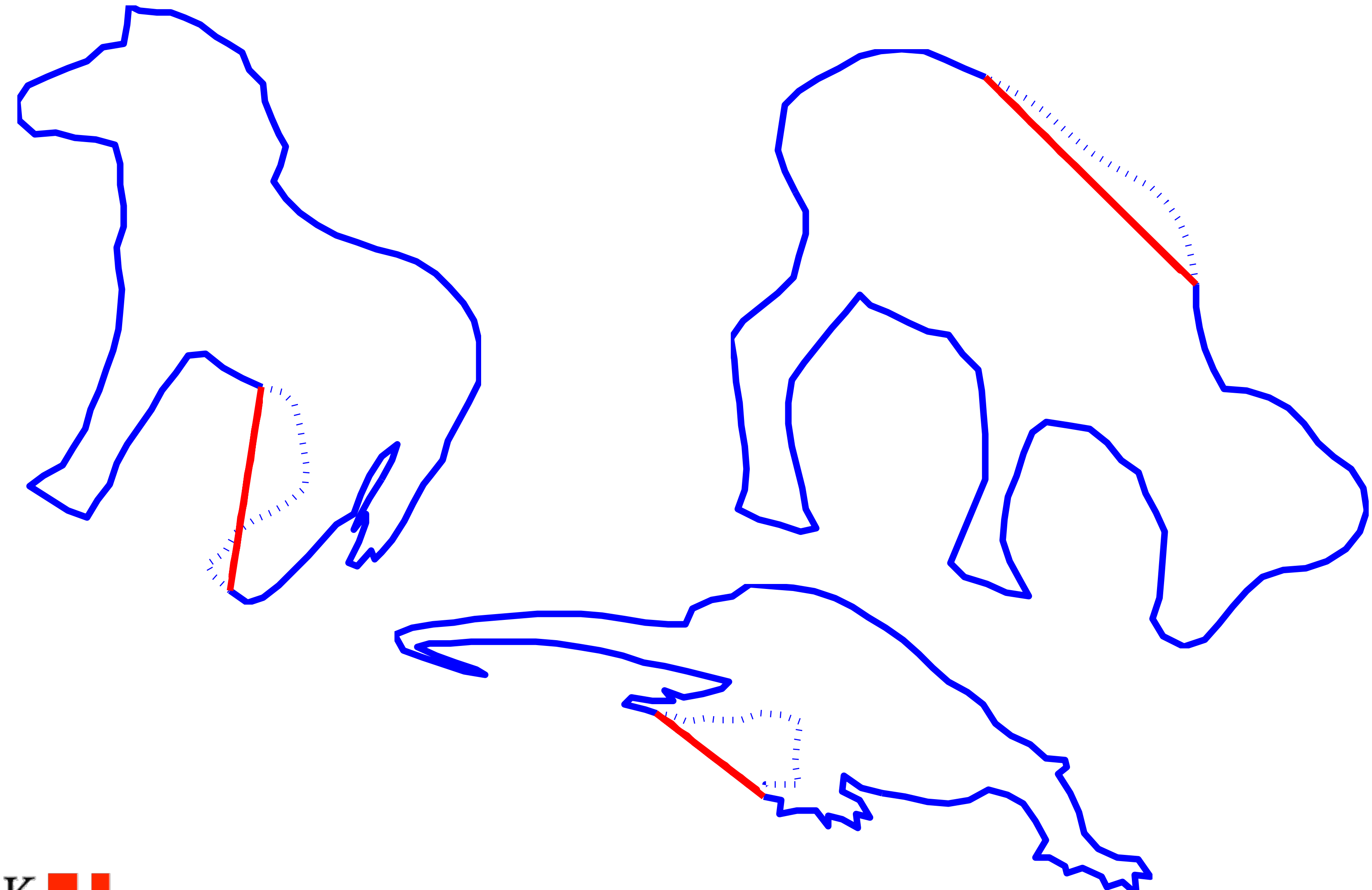
Shape Model 2 Completions



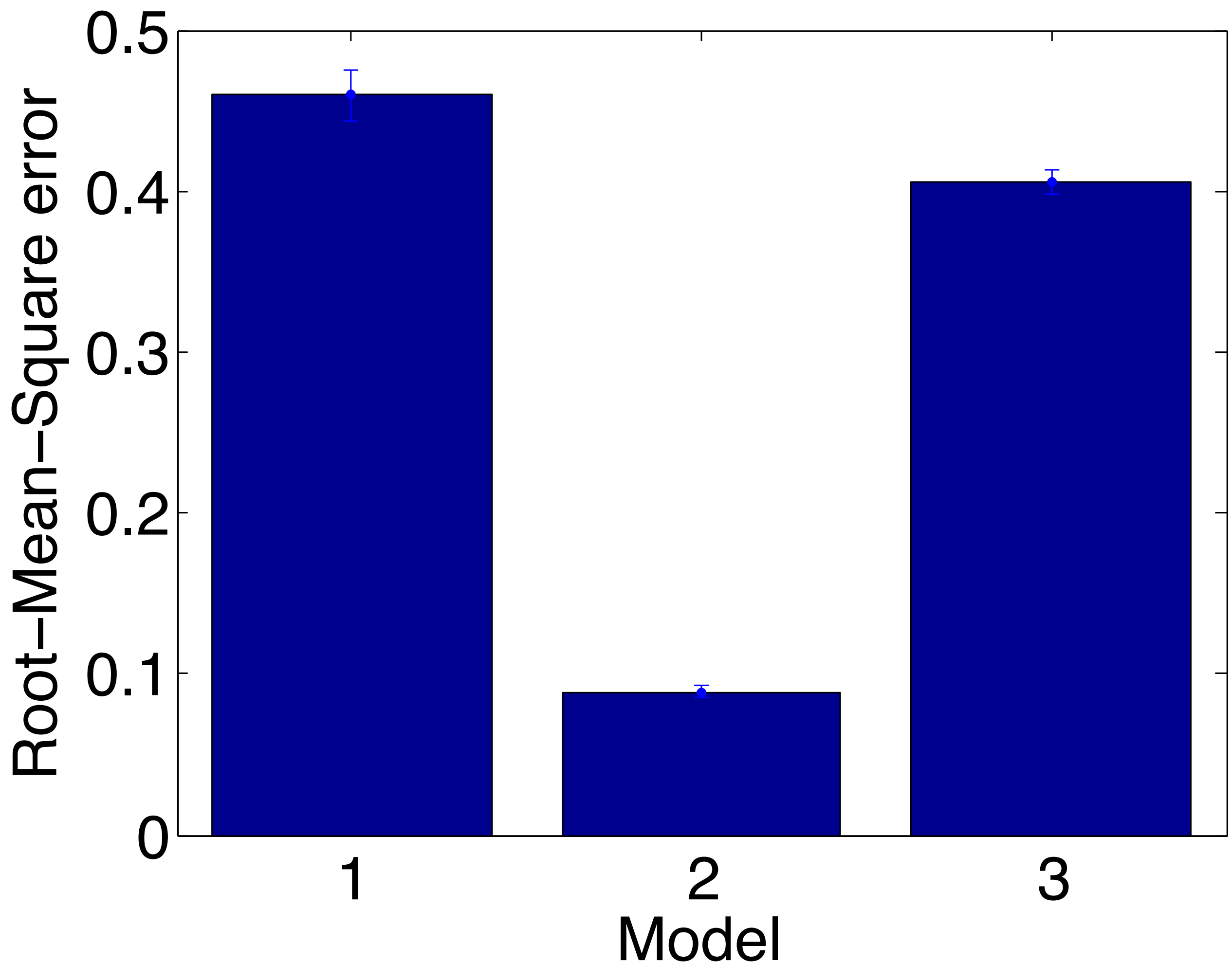
Shape Model 3

- This model is not generative: it simply uses linear interpolation to estimate the missing points.
- Functions:
  - ShapeModel3Complete.m - estimates missing portion of a given shape

Shape Model 3 Completions



Evaluation on Shape Completion



LINEAR REGRESSION

J. Elder

Credits

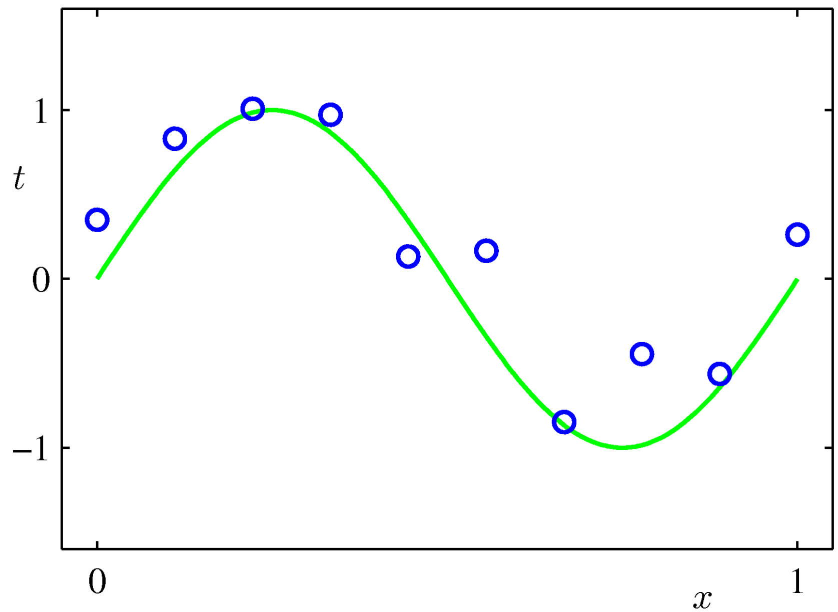
- Some of these slides were sourced and/or modified from Christopher Bishop, Microsoft UK

Outline

- Maximum Likelihood Regression
- Regularized Regression
- Bayesian Regression
- Prediction
- Kernel Regression
- Bayesian Model Comparison

Linear Basis Function Models (1)

- Example: Polynomial Curve Fitting



$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Linear Basis Function Models (2)

- Generally

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

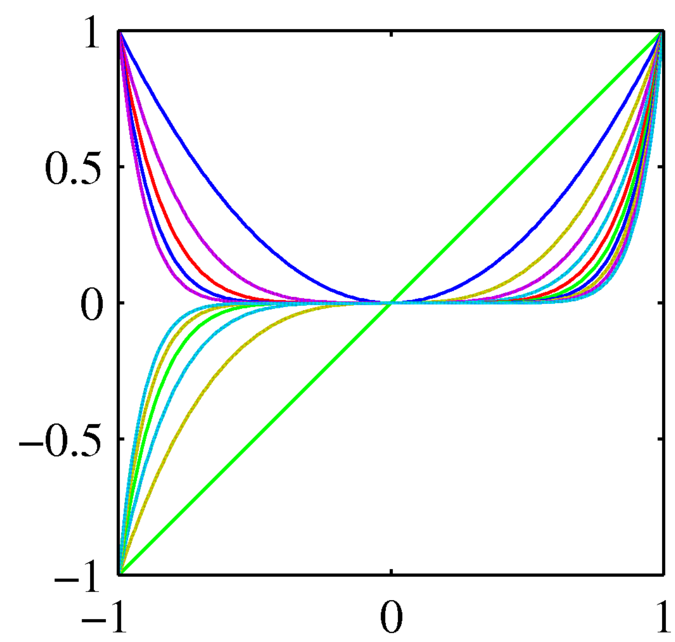
- where $\phi_j(x)$ are known as *basis functions.*
- Typically, $\Phi_0(x) = 1$, so that w_0 acts as a bias.
- In the simplest case, we use linear basis functions : $\Phi_d(x) = x_d$.

Linear Basis Function Models (3)

- Polynomial basis functions:

$$\phi_j(x) = x^j.$$

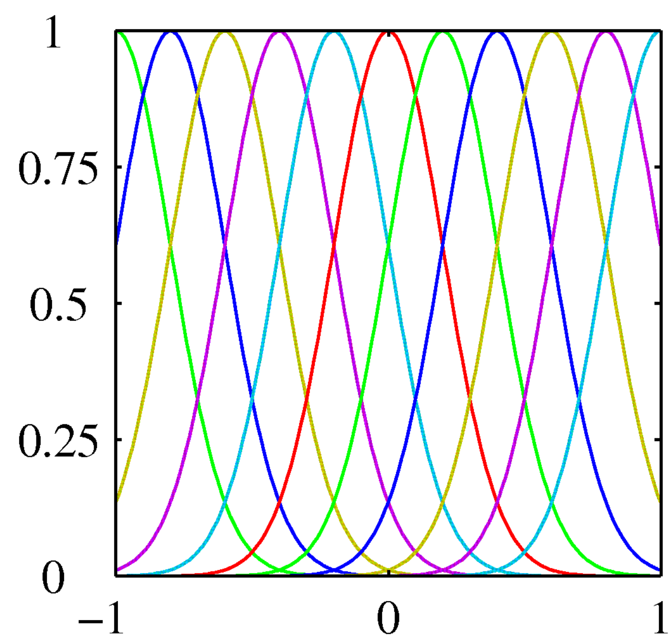
- These are global
  - a small change in x affects all basis functions.
  - A small change in a basis function affects y for all x.



Linear Basis Function Models (4)

- Gaussian basis functions:

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$



- These are local:
  - a small change in x affects only nearby basis functions.
  - a change in a basis function affects y only for nearby x.
  - μ_j and s control location and scale (width).

Linear Basis Function Models (5)

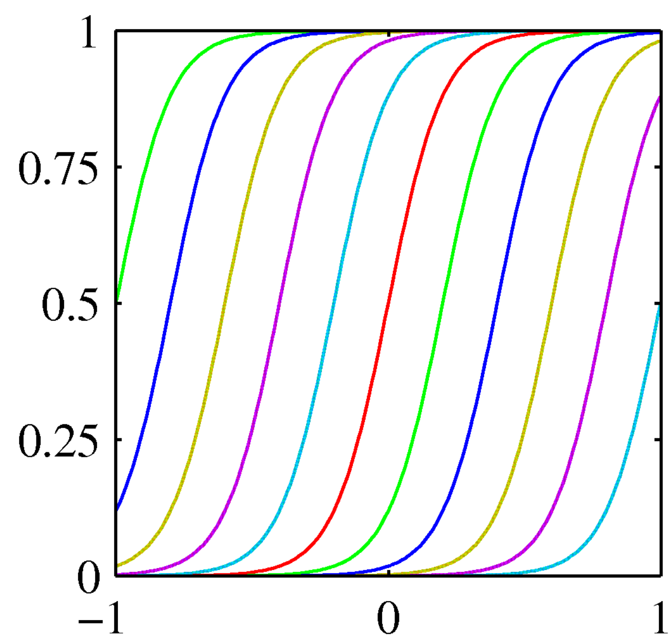
- Sigmoidal basis functions:

$$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right)$$

- where

$$\sigma(a) = \frac{1}{1 + \exp(-a)}.$$

- Also local:
  - a small change in x affects only nearby basis functions.
  - a change in a basis function affects y only for nearby x.
  - μ_j and s control location and scale (slope).



Maximum Likelihood and Least Squares

- Assume observations from a deterministic function with added Gaussian noise:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \qquad \text{where} \qquad p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$$

- which is the same as saying,

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}).$$

- Given observed inputs, $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and targets, $\mathbf{t} = [t_1, \ldots, t_N]^{\mathrm{T}}$ we obtain the likelihood function

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}).$$

Maximum Likelihood and Least Squares

- Taking the logarithm, we get

$$\begin{aligned}\ln p(\mathbf{t}|\mathbf{w},\beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n),\beta^{-1}) \\ &= \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) - \beta E_D(\mathbf{w})\end{aligned}$$

- where

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

- is the sum-of-squares error.

Maximum Likelihood and Least Squares

- Computing the gradient and setting it to zero yields

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t}|\mathbf{w}, \beta) = \beta \sum_{n=1}^{N} \left\{ t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} = \mathbf{0}.$$

- Solving for w, we get

$$\mathbf{w}_{\mathrm{ML}} = \overbrace{\left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}}}^{\text{The Moore-Penrose pseudo-inverse, } \boldsymbol{\Phi}^{\dagger}.} \mathbf{t}$$

- where

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}.$$

Maximum Likelihood and Least Squares

- Maximizing with respect to the bias, w_0, alone, we see that

$$
\begin{aligned}
w_0 &= \bar{t} - \sum_{j=1}^{M-1} w_j \overline{\phi_j} \\
&= \frac{1}{N}\sum_{n=1}^{N} t_n - \sum_{j=1}^{M-1} w_j \frac{1}{N}\sum_{n=1}^{N} \phi_j(\mathbf{x}_n).
\end{aligned}
$$

- We can also maximize with respect to β, giving

$$
\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{t_n - \mathbf{w}_{\mathrm{ML}}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2
$$

Geometry of Least Squares

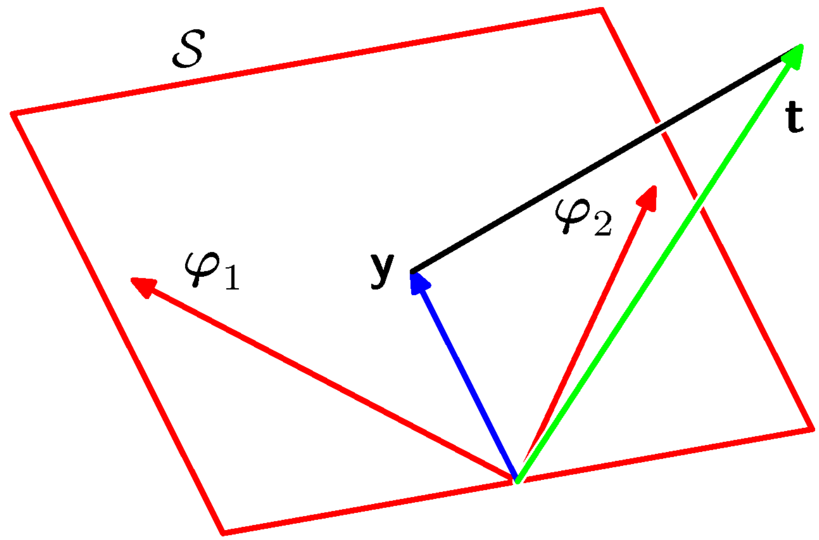
- Consider

$$\mathbf{y} = \boldsymbol{\Phi}\mathbf{w}_{\mathrm{ML}} = [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_M]\,\mathbf{w}_{\mathrm{ML}}.$$

$$\mathbf{y} \in \mathcal{S} \subseteq \mathcal{T} \qquad \mathbf{t} \in \mathcal{T}$$

N-dimensional

M-dimensional



- S is spanned by $\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_M$.
- w_{ML} minimizes the distance between t and y by making y the orthogonal projection of t onto S

Sequential Learning

- Data items considered one at a time (a.k.a. online learning); use stochastic (sequential) gradient descent:

$$
\begin{aligned}
\mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} - \eta \nabla E_n \\
&= \mathbf{w}^{(\tau)} + \eta (t_n - \mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n)) \boldsymbol{\phi}(\mathbf{x}_n).
\end{aligned}
$$

- This is known as the *least-mean-squares (LMS) algorithm*. Issue: how to choose η? (We will not cover this.)

END OF LECTURE
OCT 18, 2010

Assignment 1 Lab

- Wed Nov 3, 2:30-5:30 (pm!)
- Bring your laptops!

Regularized Least Squares (1)

- Consider the error function:

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Data term + Regularization term

- With the sum-of-squares error function and a quadratic regularizer, we get

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

- which is minimized by

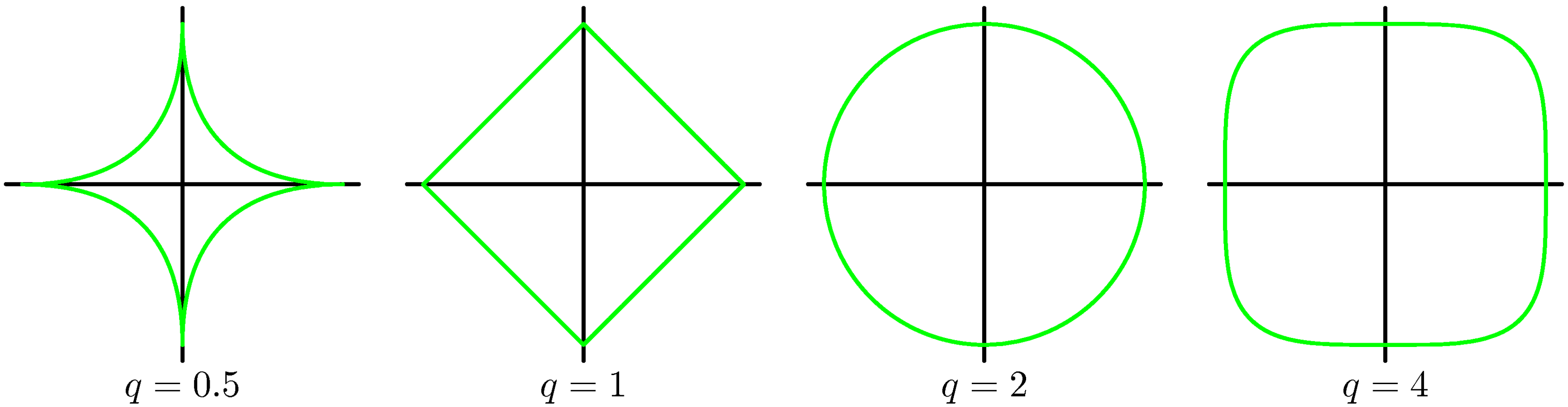
$$\mathbf{w} = \left(\lambda\mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}.$$

λ is called the regularization coefficient.

Regularized Least Squares (2)

- With a more general regularizer, we have

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$

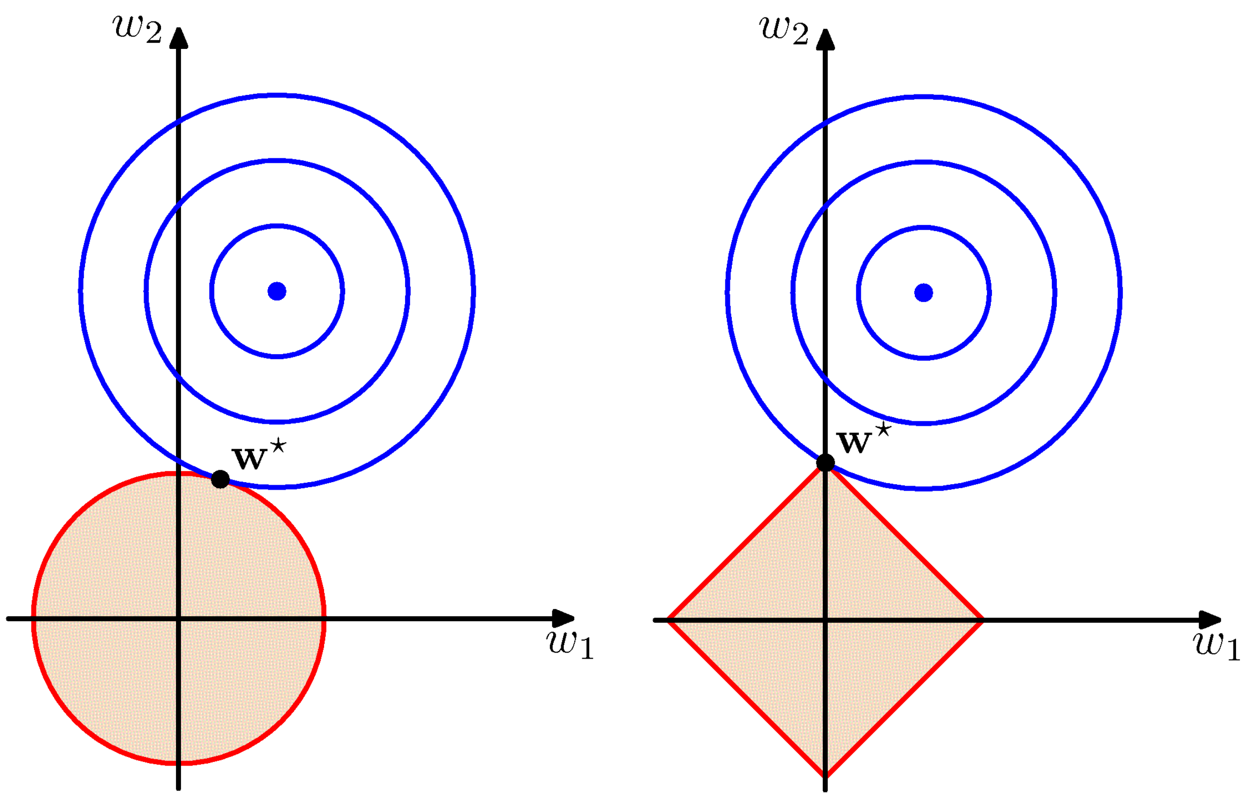


Lasso

Quadratic

Regularized Least Squares (3)

- Lasso generates sparse solutions.



Multiple Outputs (1)

- Analogous to the single output case we have:

$$\begin{aligned} p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) &= \mathcal{N}(\mathbf{t}|\mathbf{y}(\mathbf{W}, \mathbf{x}), \beta^{-1}\mathbf{I}) \\ &= \mathcal{N}(\mathbf{t}|\mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I}). \end{aligned}$$

- Given observed inputs $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and targets $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]^{\mathrm{T}}$

  we obtain the log likelihood function

$$\begin{aligned} \ln p(\mathbf{T}|\mathbf{X}, \mathbf{W}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n|\mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}\mathbf{I}) \\ &= \frac{NK}{2} \ln\left(\frac{\beta}{2\pi}\right) - \frac{\beta}{2}\sum_{n=1}^{N}\left\|\mathbf{t}_n - \mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right\|^2. \end{aligned}$$

Multiple Outputs (2)

- Maximizing with respect to W, we obtain

$$\mathbf{W}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{T}.$$

- If we consider a single target variable, t_k, we see that

$$\mathbf{w}_k = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}_k = \boldsymbol{\Phi}^{\dagger}\mathbf{t}_k$$

- where $\mathbf{t}_k = [t_{1k}, \ldots, t_{Nk}]^{\mathrm{T}}$, which is identical with the single output case.

Bayesian Linear Regression (1)

- Define a conjugate prior over w

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0).$$

- Combining this with the likelihood function and using results for marginal and conditional Gaussian distributions, gives the posterior

- where $$p(\mathbf{w}|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$$

$$\begin{aligned}
\mathbf{m}_N &= \mathbf{S}_N \left(\mathbf{S}_0^{-1}\mathbf{m}_0 + \beta \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t} \right) \\
\mathbf{S}_N^{-1} &= \mathbf{S}_0^{-1} + \beta \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}.
\end{aligned}$$

Bayesian Linear Regression (2)

- A common choice for the prior is

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

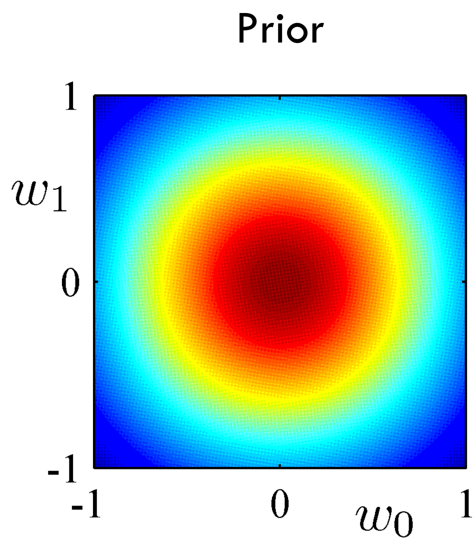
- for which

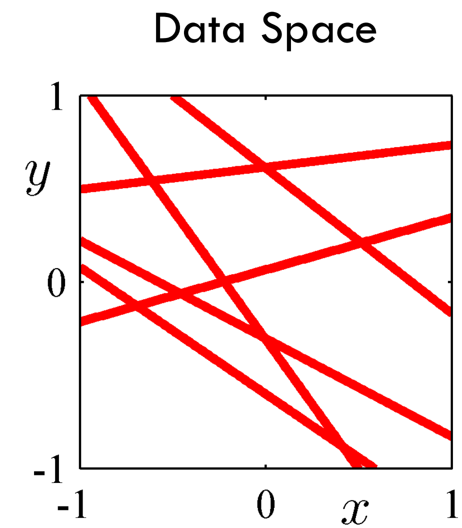
$$\begin{aligned} \mathbf{m}_N &= \beta \mathbf{S}_N \mathbf{\Phi}^{\mathrm{T}} \mathbf{t} \\ \mathbf{S}_N^{-1} &= \alpha \mathbf{I} + \beta \mathbf{\Phi}^{\mathrm{T}} \mathbf{\Phi}. \end{aligned}$$

- Next we consider an example …

Bayesian Linear Regression (3)

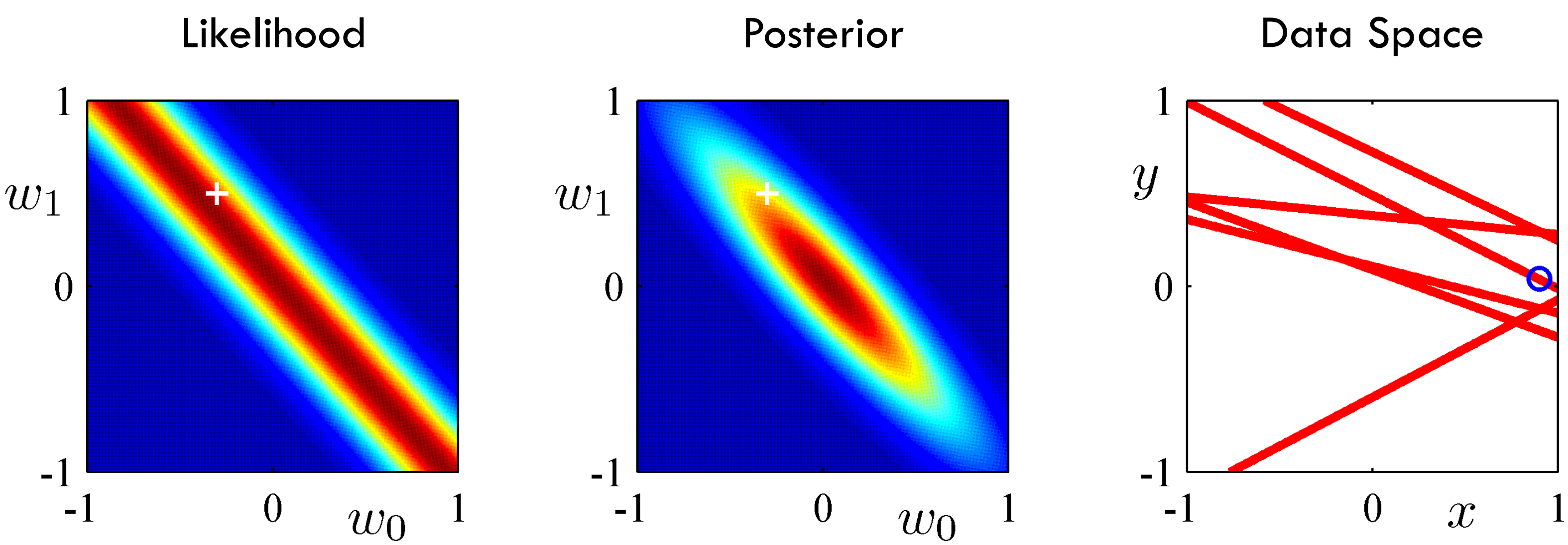
0 data points observed





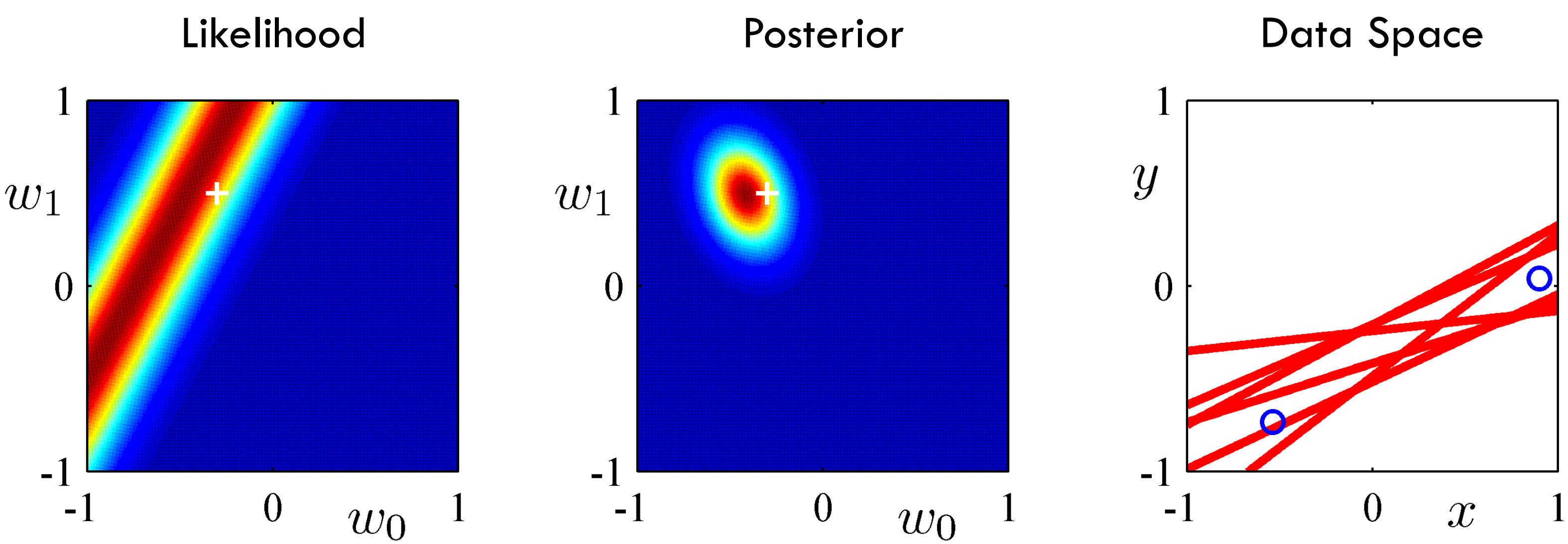
Bayesian Linear Regression (4)

1 data point observed



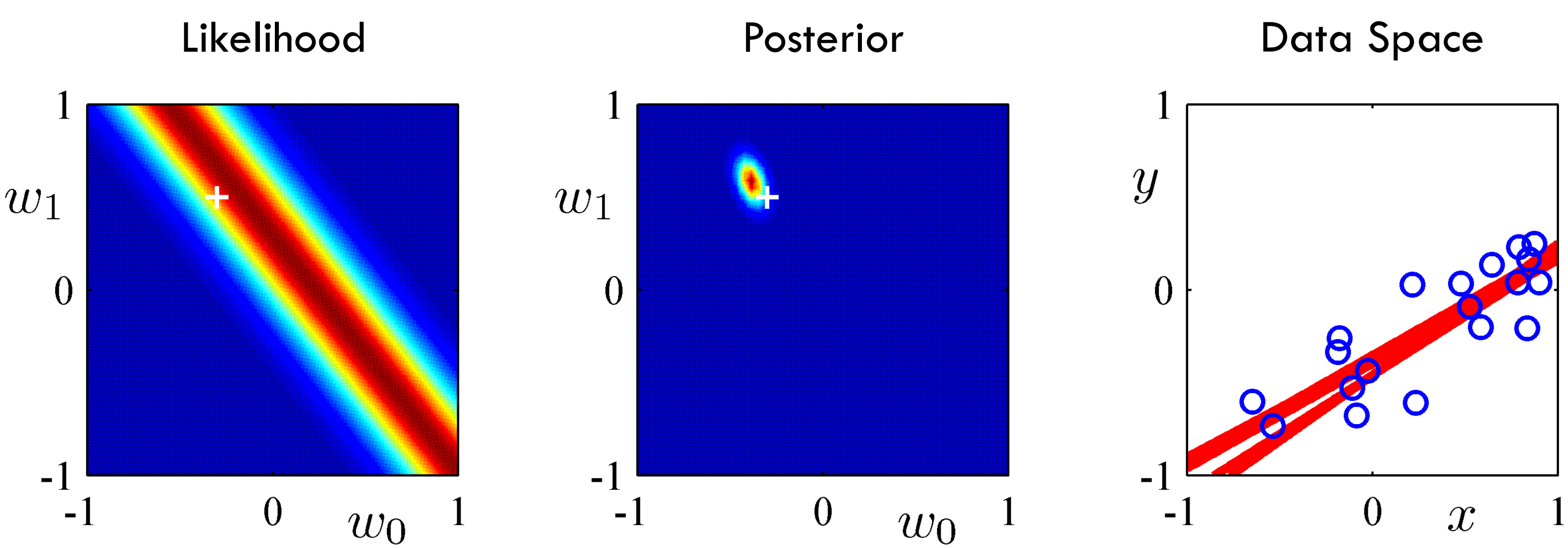
Bayesian Linear Regression (5)

2 data points observed



Bayesian Linear Regression (6)

20 data points observed



Predictive Distribution (1)

- Predict t for new values of x by integrating over w:

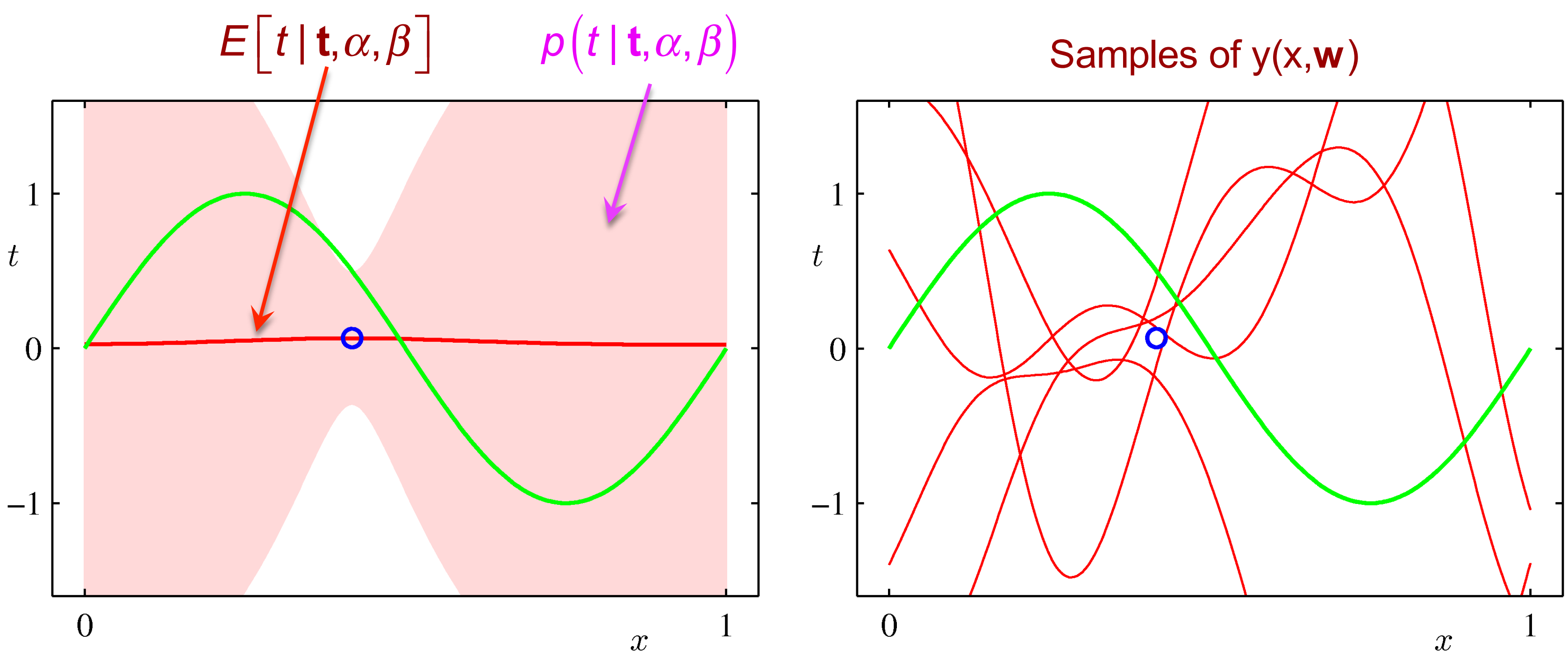
$$
\begin{aligned}
p(t|\mathbf{t}, \alpha, \beta) &= \int p(t|\mathbf{w}, \beta) p(\mathbf{w}|\mathbf{t}, \alpha, \beta) \, \mathrm{d}\mathbf{w} \\
&= \mathcal{N}(t|\mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}), \sigma_N^2(\mathbf{x}))
\end{aligned}
$$

- where

$$
\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}).
$$

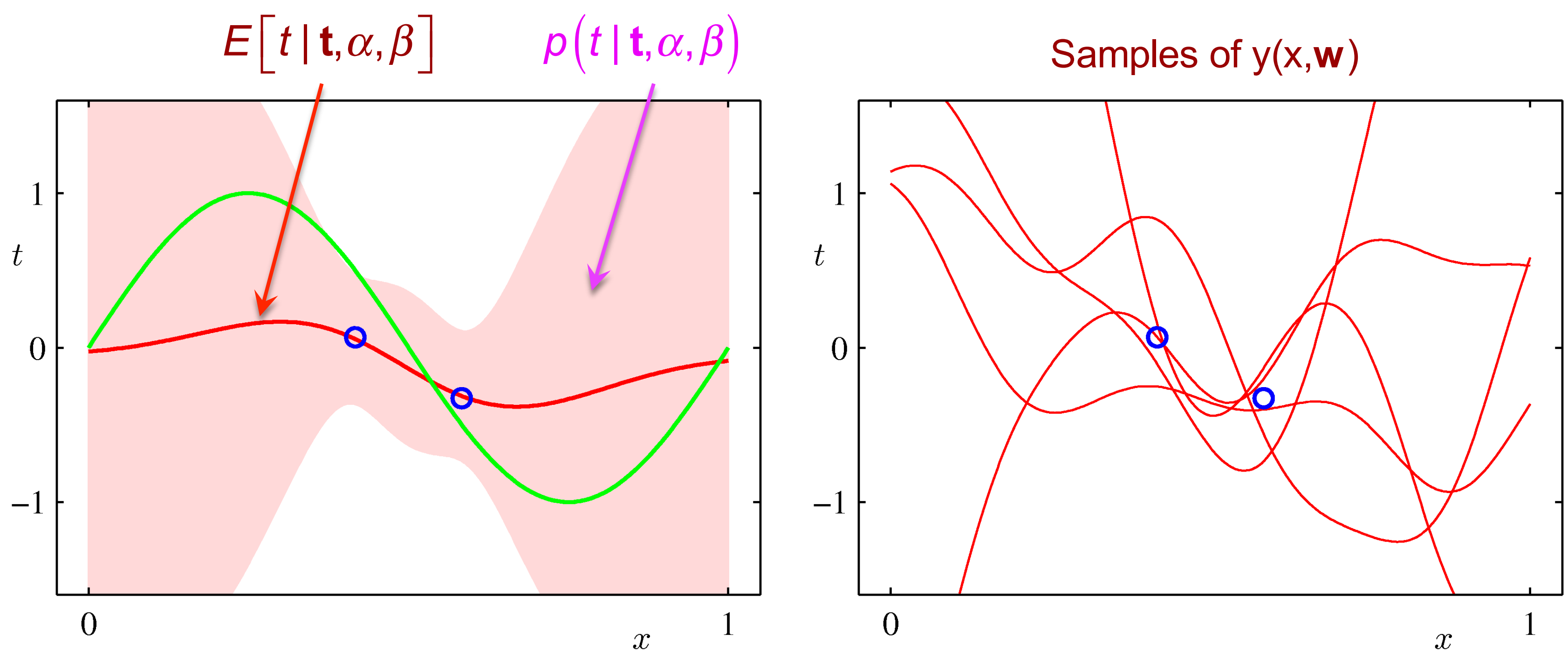
Predictive Distribution (2)

- Example: Sinusoidal data, 9 Gaussian basis functions, 1 data point



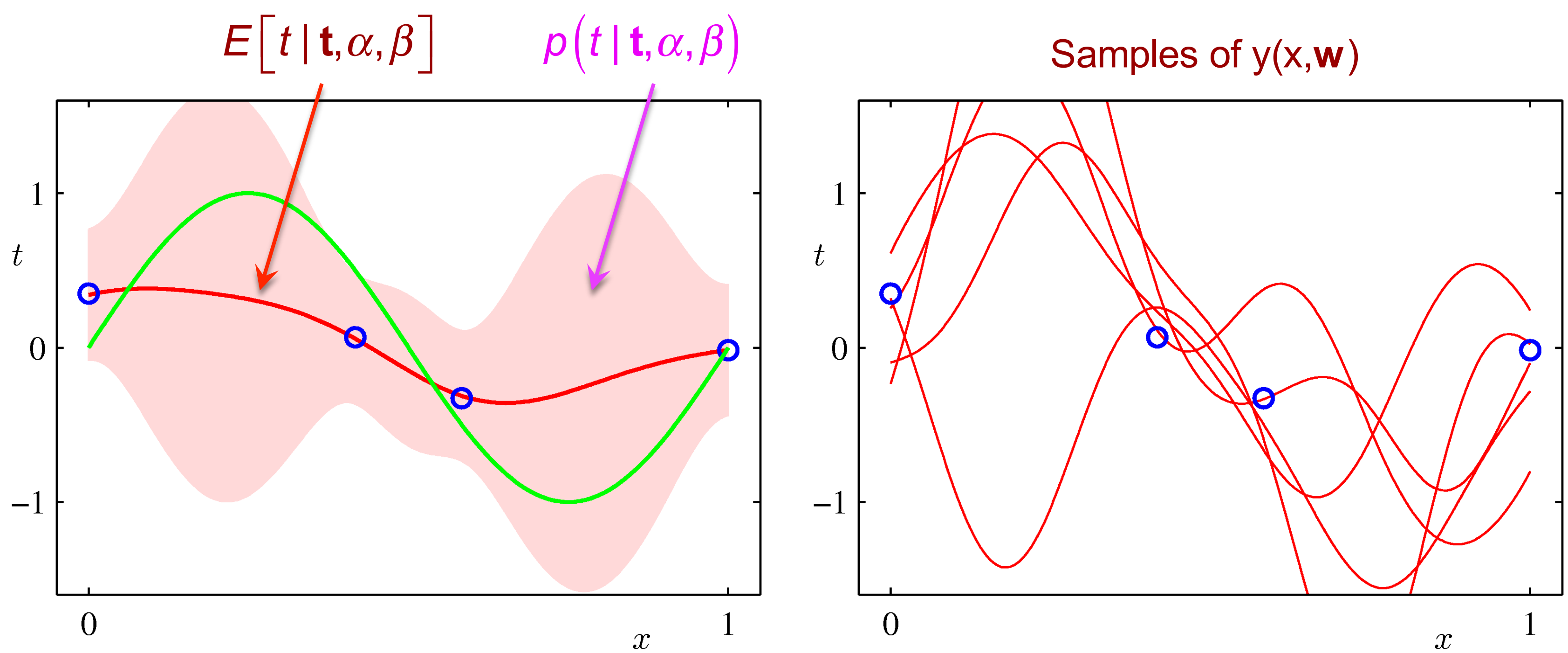
Predictive Distribution (3)

- Example: Sinusoidal data, 9 Gaussian basis functions, 2 data points



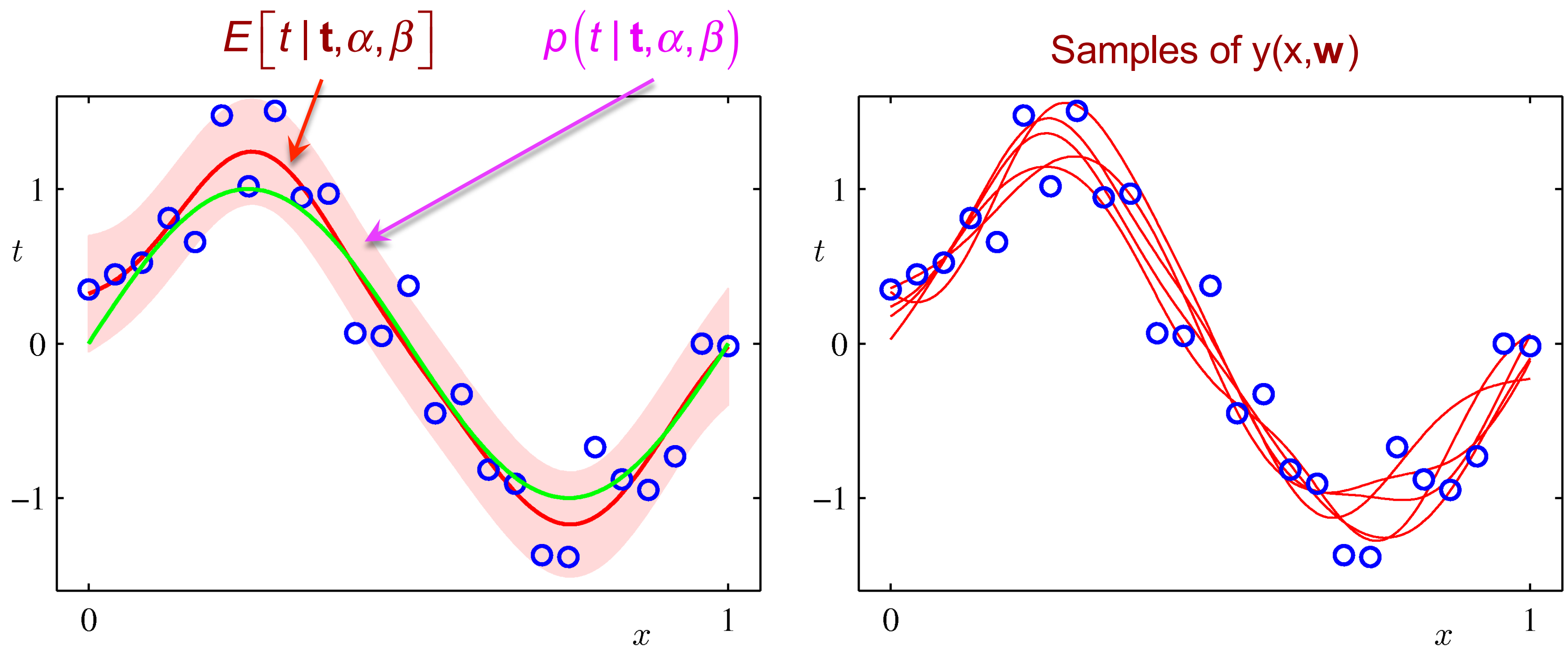
Predictive Distribution (4)

- Example: Sinusoidal data, 9 Gaussian basis functions, 4 data points



Predictive Distribution (5)

- Example: Sinusoidal data, 9 Gaussian basis functions, 25 data points



Equivalent Kernel (1)

- The predictive mean can be written

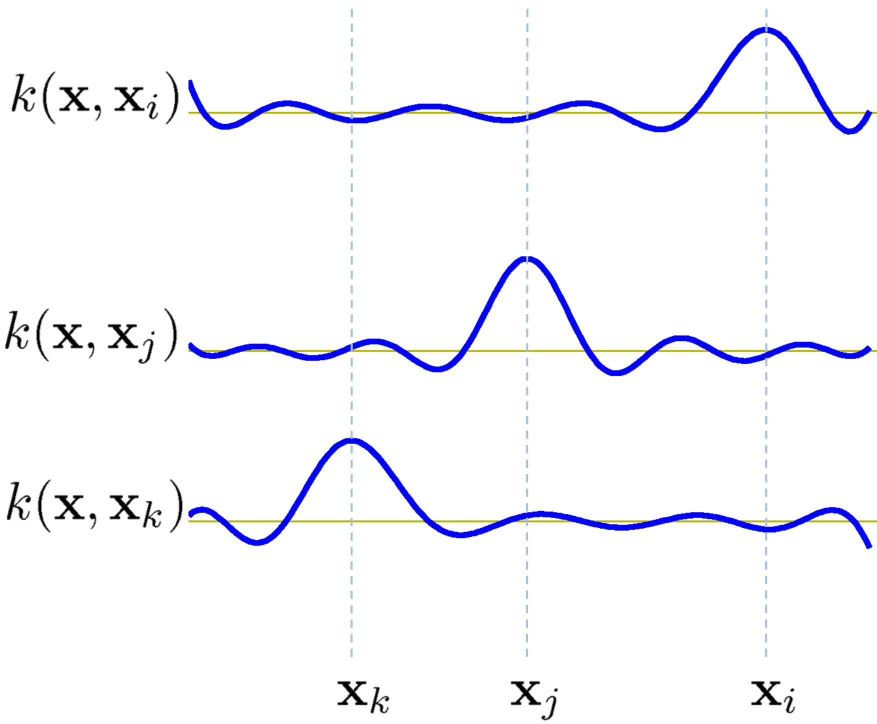
$$
\begin{aligned}
y(\mathbf{x}, \mathbf{m}_N) &= \mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}) = \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} \\
&= \sum_{n=1}^{N} \underbrace{\beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}_n)}_{} t_n \\
&= \sum_{n=1}^{N} \overbrace{k(\mathbf{x}, \mathbf{x}_n)}^{} t_n .
\end{aligned}
$$

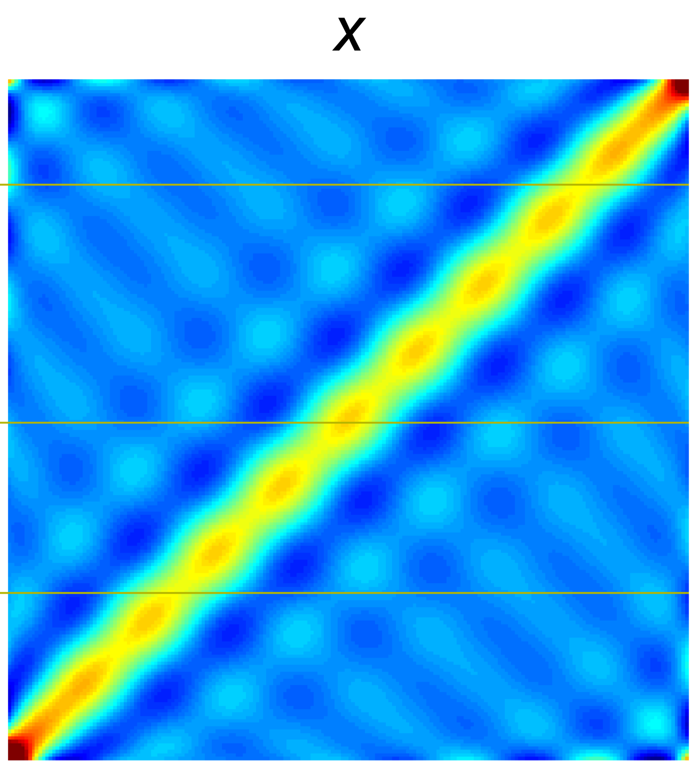
Equivalent kernel or smoother matrix.

- This is a weighted sum of the training data target values, t_n.

Equivalent Kernel (2)

For Gaussian basis

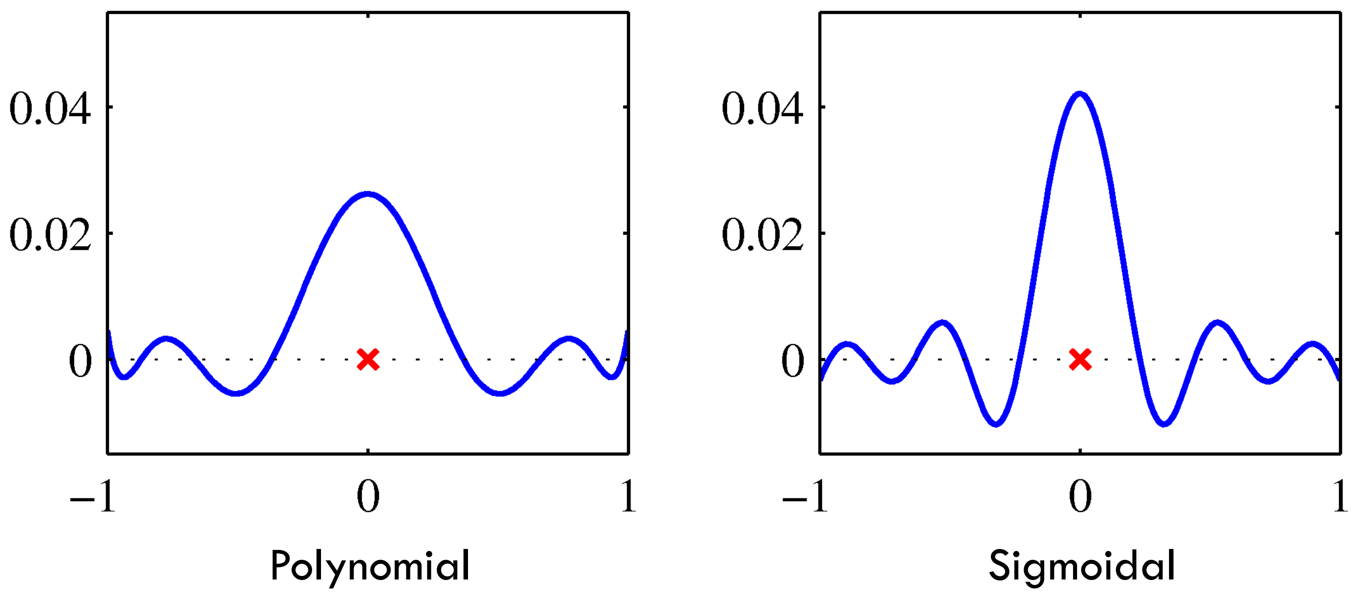




Weight of t_n depends on distance between x and x_n; nearby x_n carry more weight.

Equivalent Kernel (3)

- Non-local basis functions have local equivalent kernels:



Equivalent Kernel (4)

- The kernel as a covariance function: consider

$$\begin{aligned}\operatorname{cov}[y(\mathbf{x}), y(\mathbf{x}')] &= \operatorname{cov}[\boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{w}, \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}')] \\ &= \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}') = \beta^{-1}k(\mathbf{x}, \mathbf{x}').\end{aligned}$$

- We can avoid the use of basis functions and define the kernel function directly, leading to *Gaussian Processes* (Chapter 6).

Equivalent Kernel (5)

$$\sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) = 1$$

- for all values of x; however, the equivalent kernel may be negative for some values of x.

- Like all kernel functions, the equivalent kernel can be expressed as an inner product:

$$k(\mathbf{x}, \mathbf{z}) = \boldsymbol{\psi}(\mathbf{x})^{\mathrm{T}} \boldsymbol{\psi}(\mathbf{z})$$

- where .

$$\boldsymbol{\psi}(\mathbf{x}) = \beta^{1/2} \mathbf{S}_N^{1/2} \boldsymbol{\phi}(\mathbf{x})$$

Bayesian Model Comparison (1)

- How do we choose the 'right' model?
- Assume we want to compare models M_i, i=1, …,L, using data D; this requires computing

$$p(\mathcal{M}_i|\mathcal{D}) \propto p(\mathcal{M}_i)p(\mathcal{D}|\mathcal{M}_i).$$

Posterior   Prior   *Model evidence or marginal likelihood*

- *Bayes Factor*: ratio of evidence for two models

$$\frac{p(\mathcal{D}|\mathcal{M}_i)}{p(\mathcal{D}|\mathcal{M}_j)}$$

Bayesian Model Comparison (2)

- Having computed $p(M_i|D)$, we can compute the predictive (mixture) distribution

$$p(t|\mathbf{x}, \mathcal{D}) = \sum_{i=1}^{L} p(t|\mathbf{x}, \mathcal{M}_i, \mathcal{D}) p(\mathcal{M}_i|\mathcal{D}).$$

- A simpler approximation, known as *model selection*, is to use the model with the highest evidence.

Bayesian Model Comparison (3)

- For a model with parameters w, we get the model evidence by marginalizing over w

$$p(\mathcal{D}|\mathcal{M}_i) = \int p(\mathcal{D}|\mathbf{w}, \mathcal{M}_i) p(\mathbf{w}|\mathcal{M}_i)\, \mathrm{d}\mathbf{w}.$$

- Note that

$$p(\mathbf{w}|\mathcal{D}, \mathcal{M}_i) = \frac{p(\mathcal{D}|\mathbf{w}, \mathcal{M}_i) p(\mathbf{w}|\mathcal{M}_i)}{p(\mathcal{D}|\mathcal{M}_i)}$$

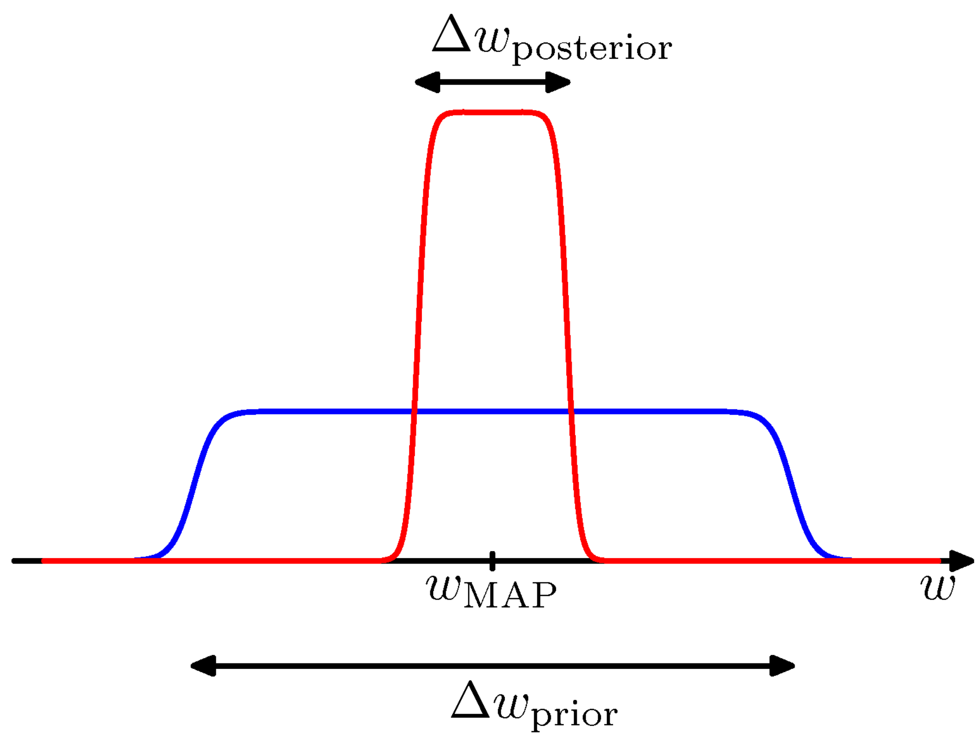
Bayesian Model Comparison (4)

- For a given model with a single parameter, w, consider the approximation

$$p(\mathcal{D}) = \int p(\mathcal{D}|w)p(w)\,\mathrm{d}w$$

$$\simeq \quad p(\mathcal{D}|w_{\mathrm{MAP}})\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}$$

- where the posterior is assumed to be sharply peaked.



Bayesian Model Comparison (5)

- Taking logarithms, we obtain

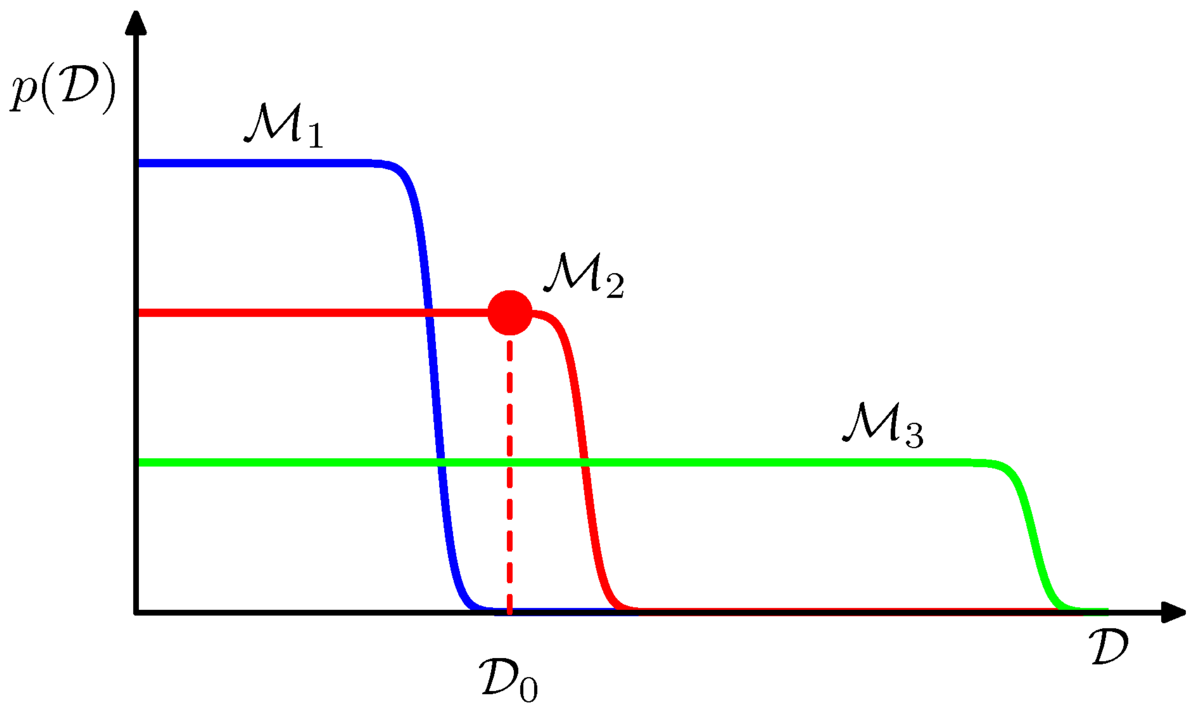
$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|w_{\mathrm{MAP}}) + \underbrace{\ln\left(\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}\right)}_{\text{Negative}}.$$

- With M parameters, all assumed to have the same ratio $\Delta w_{\mathrm{posterior}}/\Delta w_{\mathrm{prior}}$, we get

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\mathbf{w}_{\mathrm{MAP}}) + \underbrace{M \ln\left(\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}\right)}_{\text{Negative and linear in M.}}.$$

Bayesian Model Comparison (6)

- Matching data and model complexity



Limitations of Fixed Basis Functions

- M basis function along each dimension of a D-dimensional input space requires M^D basis functions: the curse of dimensionality.
- In later chapters, we shall see how we can get away with fewer basis functions, by choosing these using the training data.