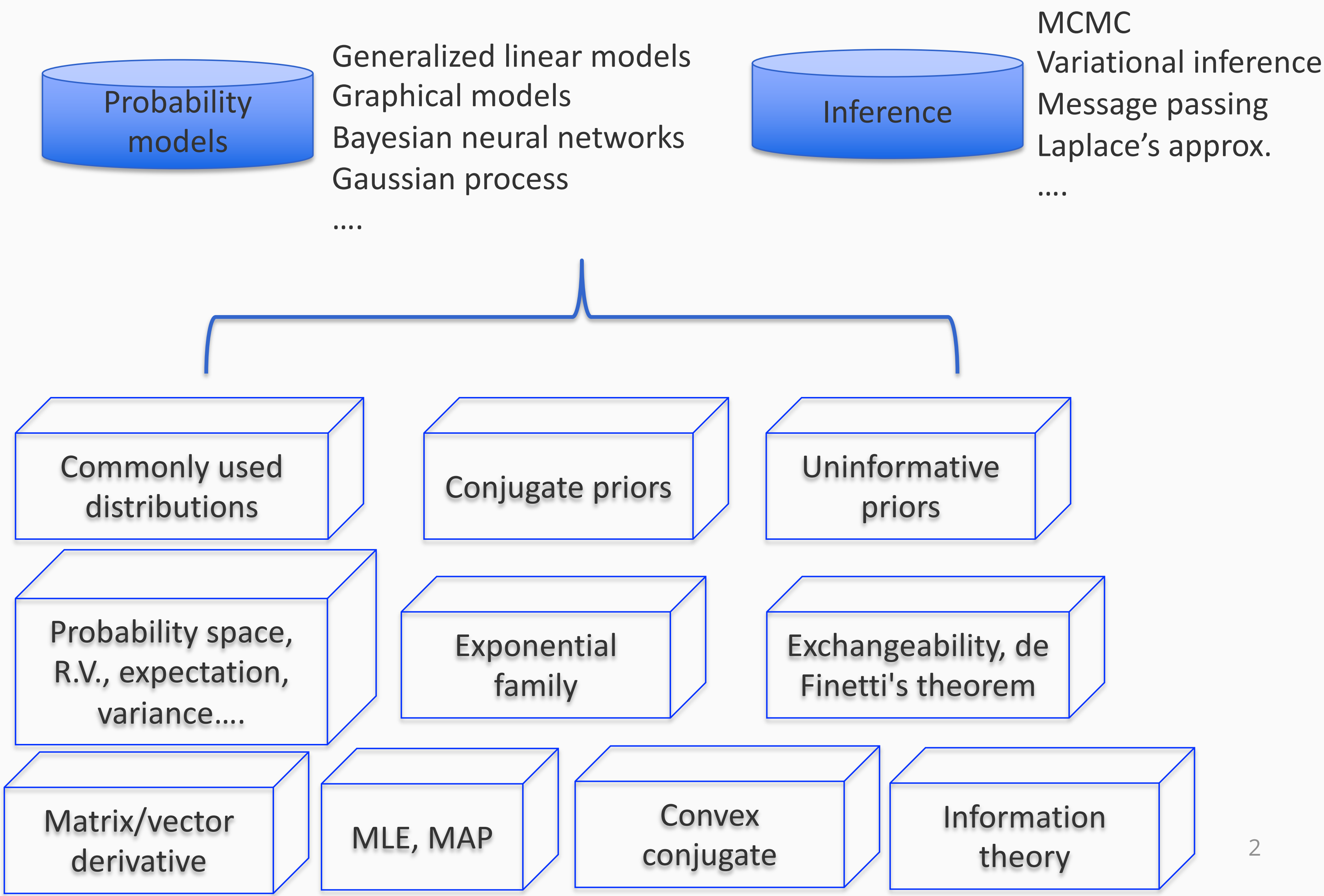
Generalized Linear Models

Spring 2024

Instructor: Shandian Zhe
zhe@cs.utah.edu
School of Computing
THE UNIVERSITY OF UTAH

So far, we have …



Our next stage

- Discuss several important and widely used probabilistic models (and framework)
- Discuss efficient posterior inference algorithm
- We will start with generalized linear models

Outline

- Linear models for regression
- Linear models for classification
- Generalized linear models

Linear models for regression

- Linear models with (nonlinear) basis functions
- Overfitting and regularization
- Bayesian linear regression
- Predictive distribution
- Empirical Bayes

Linear models for regression

- Simplest model: linear regression

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D$$

$$\mathbf{x} = (x_1, \ldots, x_D)^{\mathrm{T}}$$

Linear models for regression

- Simplest model: linear regression

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D$$

$$\mathbf{x} = (x_1, \ldots, x_D)^{\mathrm{T}}$$

Limitation: only model linear function of the input variables

Linear models for regression

- To allow nonlinear modeling, we in general introduce *nonlinear* M basis functions over the input variables

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

Linear models for regression

- To allow nonlinear modeling, we in general introduce *nonlinear* *M* basis functions over the input variables

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

$$\phi_j : \mathbb{R}^D \longrightarrow \mathbb{R}$$

Basis function: can be any (nonlinear) over the input variables

Examples of basis functions

- D = 1

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

- D > 1

$$\phi_j(\mathbf{x}) = x_j \qquad \phi_j(\mathbf{x}) = sin(x_j) \qquad \ldots$$

Examples of basis functions

- D = 1

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

- D > 1

$$\phi_j(\mathbf{x}) = x_j \qquad \phi_j(\mathbf{x}) = sin(x_j) \qquad \ldots$$

Through nonlinear basis functions, we can model nonlinear functions while maintaining a linear structure

Examples of basis functions

- D = 1

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

- D > 1

$$\phi_j(\mathbf{x}) = x_j \qquad \phi_j(\mathbf{x}) = sin(x_j) \qquad \ldots$$

Through nonlinear basis functions, we can model nonlinear functions while maintaining a linear structure

Neural Networks can be viewed as nonlinear bases as well

Maximum likelihood estimation (MLE)

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

- Assume the observation is the function corrupted by random Gaussian noise

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

Maximum likelihood estimation (MLE)

- Consider an observed dataset $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$

$$t_1, \dots, t_N$$

likelihood

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

$$\boldsymbol{\phi}(\mathbf{x}_n) = [\phi_0(\mathbf{x}_n), \dots, \phi_{M-1}(\mathbf{x}_n)]^\top$$

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

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

Maximum likelihood estimation (MLE)

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

$$0 = \sum_{n=1}^{N} t_n \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} - \mathbf{w}^{\mathrm{T}} \left(\sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}_n) \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} \right)$$

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

Design matrix

Maximum likelihood estimation (MLE)

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

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

$$\boldsymbol{\Phi}^{\dagger} \equiv \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}$$ Moore-Penrose pseudo-inverse

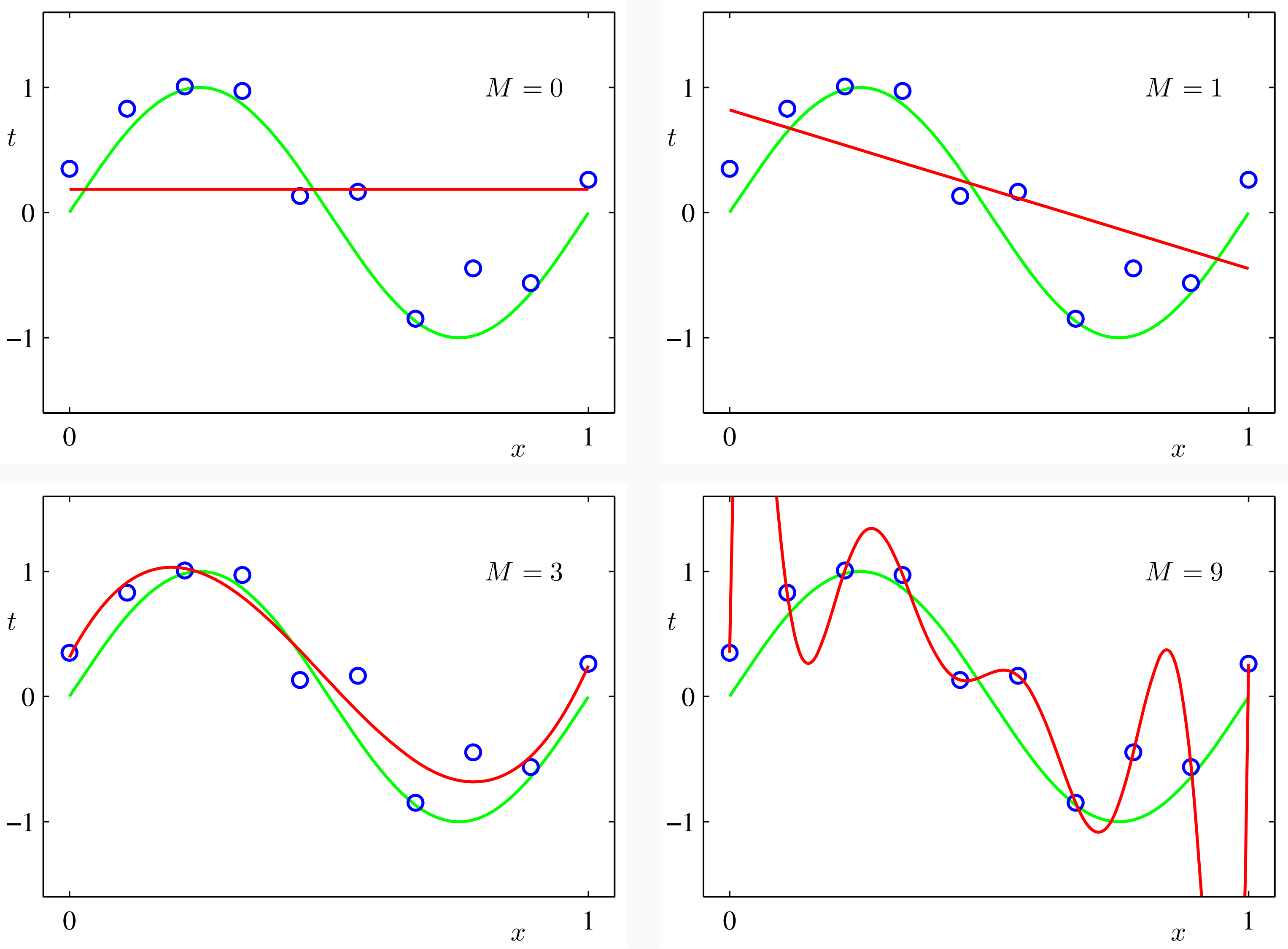
Overfitting and regularization

- Consider polynomial regression

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

Question: what is the highest order we can choose (M)?

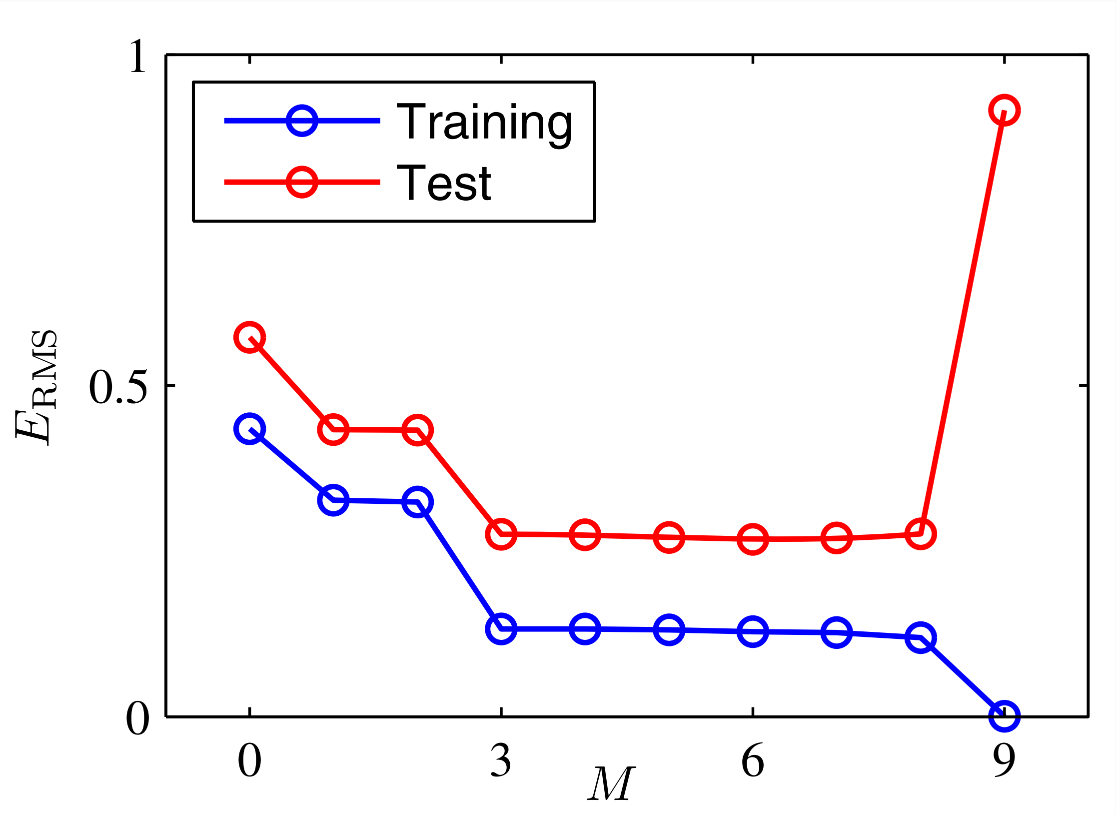
Overfitting and regularization



Overfitting and regularization

	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Overfitting and regularization



Overfitting: how to address it?

	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

We should constraint the weights from growing too big;

Weights are encouraged to decay toward 0, unless supported by data!

Regularized least square

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

Regularization strength

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

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

Regularized least square

- Set gradient to 0

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

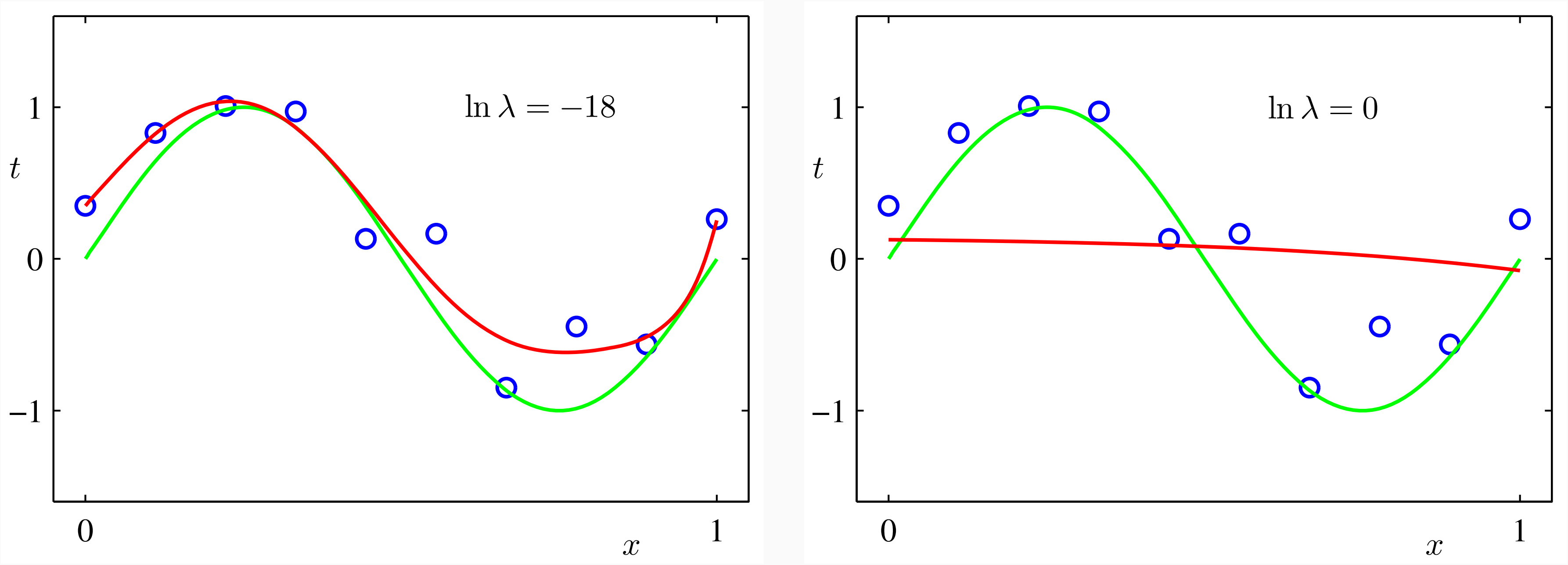
$$\boxed{\mathbf{w}_{\mathrm{ML}} = \left(\mathbf{\Phi}^{\mathrm{T}} \mathbf{\Phi}\right)^{-1} \mathbf{\Phi}^{\mathrm{T}} \mathbf{t}}$$

Go back to polynomial regression again

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

	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Go back to polynomial regression again



More general regularizer

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

When q = 2, we go back to our quadratic regularizer

When q = 1, it is known as *lasso:* a classical sparse regression approach; it turns out using lasso can lead many weights to 0

In general, the smaller q leads to sparser models

Bayesian linear regression

- We assign a prior over the weights, which corresponds to a regularizer

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

$$p(\mathbf{t}|\mathbf{w}, \mathbf{X}) = \mathcal{N}(\mathbf{t}|\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I})$$

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

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

Bayesian linear regression

- Take a simple choice

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

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

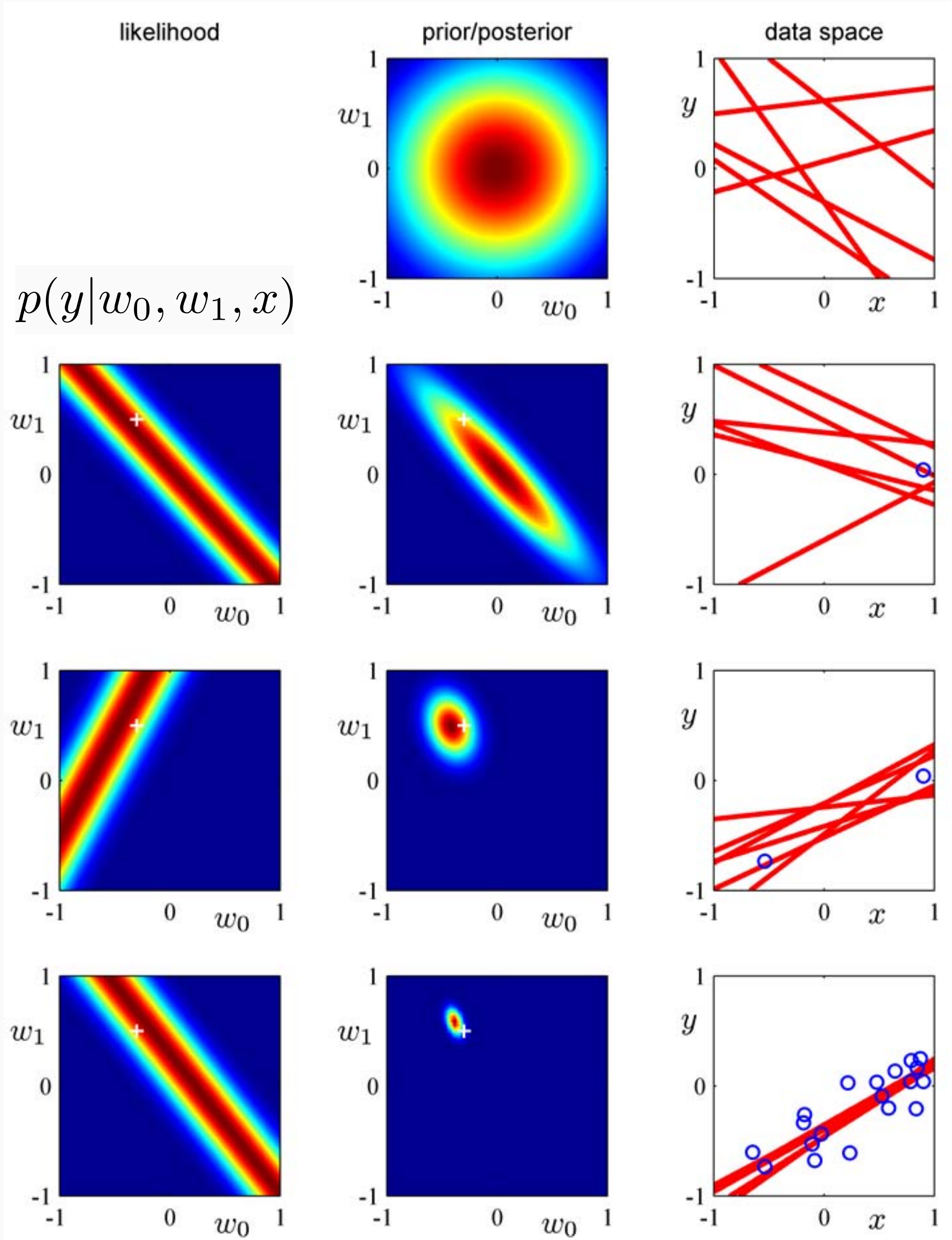
$$\boxed{\begin{aligned} \mathbf{m}_N &= \beta \mathbf{S}_N \mathbf{\Phi}^{\mathrm{T}} \mathbf{t} \\ \mathbf{S}_N^{-1} &= \alpha \mathbf{I} + \beta \mathbf{\Phi}^{\mathrm{T}} \mathbf{\Phi}. \end{aligned}}$$

See how the posterior changes

1st point

2nd point

20th point



Bayesian linear regression

- Gaussian prior corresponds to quadratic regularization; Laplace prior lasso
- In general

$$p(\mathbf{w}|\alpha) = \left[\frac{q}{2}\left(\frac{\alpha}{2}\right)^{1/q}\frac{1}{\Gamma(1/q)}\right]^{M}\exp\left(-\frac{\alpha}{2}\sum_{j=1}^{M}|w_j|^q\right)$$

q = 1, Laplace's prior
q = 2, Gaussian

Predictive distribution

- We want to integrate all values of ***w*** into prediction

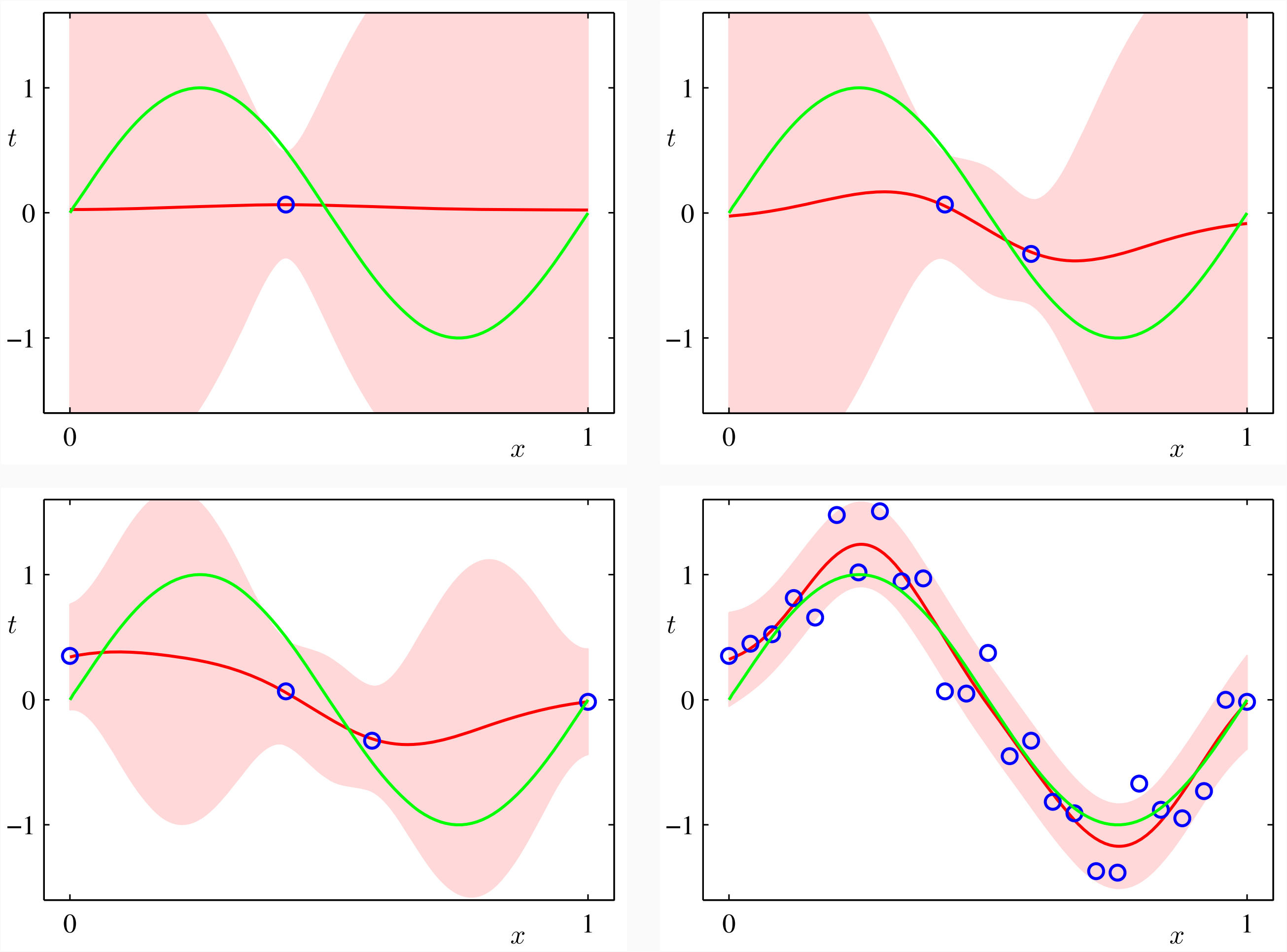
$$p(t|\mathbf{t}, \alpha, \beta) = \int p(t|\mathbf{w}, \beta)p(\mathbf{w}|\mathbf{t}, \alpha, \beta)\,\mathrm{d}\mathbf{w}$$

$$\mathcal{N}(t|\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}), \beta^{-1}) \qquad \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$$

$$p(t|\mathbf{x}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(t|\mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}), \sigma_N^2(\mathbf{x}))$$

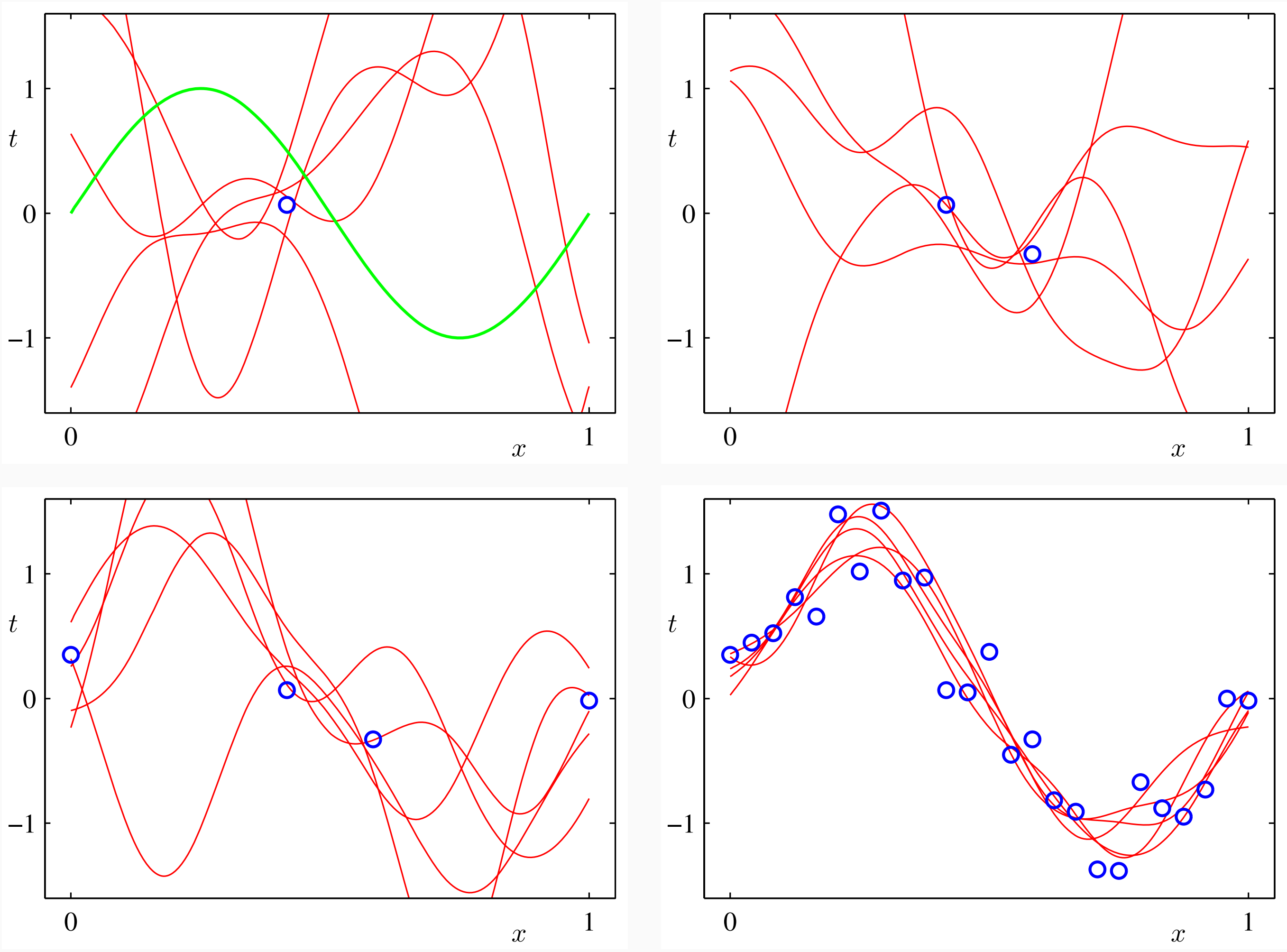
$$\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x})$$

Predictive distribution



Learn a sinusoidal function with 9 Gaussian basis functions

t(x,$\boldsymbol{w}$) using samples from the posterior $p(\mathbf{w}|\mathbf{t})$



Bayesian model comparison

- Suppose we want to compare a set of models $\{M_1, \ldots, M_L\}$.
- The data is generated by one model, which we are not sure. We express the prior by $p(M_i)$
- Given the training data D, we wish to evaluate

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

Model evidence

Bayesian model comparison

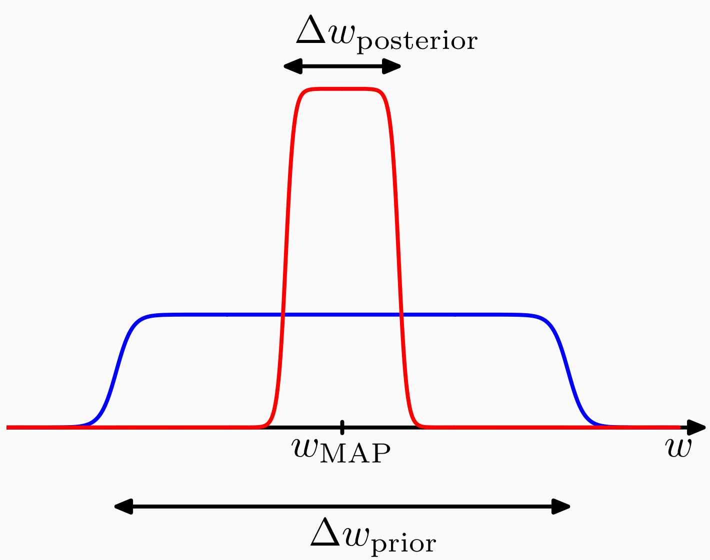
- Bayes factor $p(\mathcal{D}|\mathcal{M}_i)/p(\mathcal{D}|\mathcal{M}_j)$
- Model averaging: Bayesian version of ensemble

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

- Model selection: choose the most probable model *alone* to make prediction

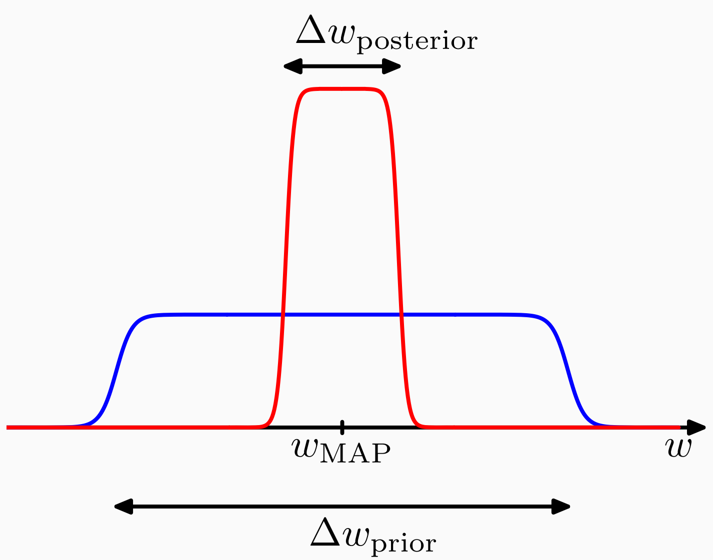
Crude evidence approximation

- Assume the posterior is centered around its mode and flat prior $p(w) = 1/\Delta w_{\text{prior}}$



Crude evidence approximation

- Assume the posterior is centered around its mode and flat prior $p(w) = 1/\Delta w_{\text{prior}}$



$$p(\mathcal{D}) = \int p(\mathcal{D}|w)p(w)\,\mathrm{d}w \simeq p(\mathcal{D}|w_{\text{MAP}})\frac{\Delta w_{\text{posterior}}}{\Delta w_{\text{prior}}}$$

Evidence penalizes over-complex models

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

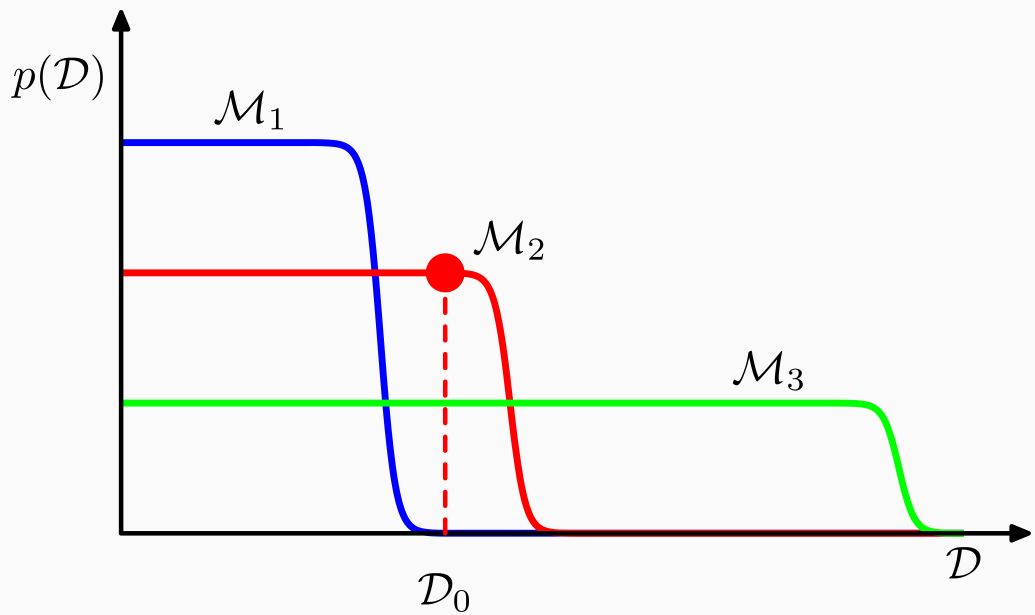
Given M parameters and assume the same ratio

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\mathbf{w}_{\mathrm{MAP}}) + M \underbrace{\ln\left(\overbrace{\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}}^{<1}\right)}_{<0}$$

The larger M, the more complex the model, the better fit of the data (1st term), the smaller the second term

Evidence penalizes over-complex models

- Maximizing evidence naturally leads to a trade-off between data fitting and model complexity



Evidence approximation & empirical Bayes

- Approximating the predictive distribution by maximizing the evidence

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

$$p(t|\mathbf{t}) = \iiint \boxed{p(t|\mathbf{w}, \beta)p(\mathbf{w}|\mathbf{t}, \alpha, \beta)} p(\alpha, \beta|\mathbf{t})\,\mathrm{d}\mathbf{w}\,\mathrm{d}\alpha\,\mathrm{d}\beta$$

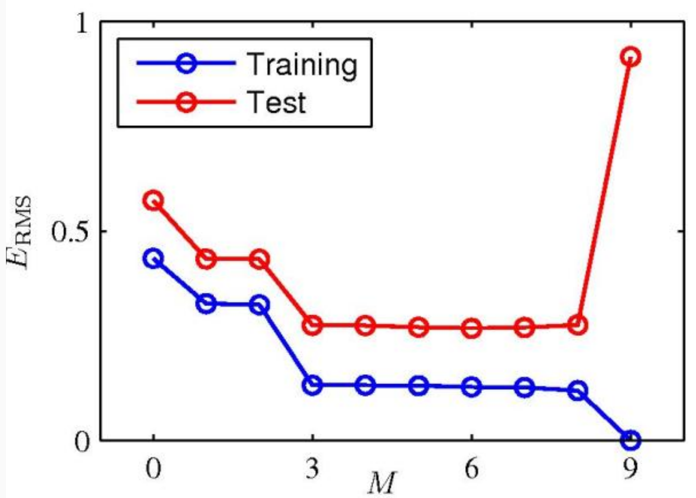
$$p(t|\mathbf{t}) \simeq p(t|\mathbf{t}, \widehat{\alpha}, \widehat{\beta}) = \int p(t|\mathbf{w}, \widehat{\beta})p(\mathbf{w}|\mathbf{t}, \widehat{\alpha}, \widehat{\beta})\,\mathrm{d}\mathbf{w}$$

where the hyperparameters $\widehat{\alpha}, \widehat{\beta}$ are obtained by maximizing the evidence $p(\mathbf{t}|\alpha, \beta)$.

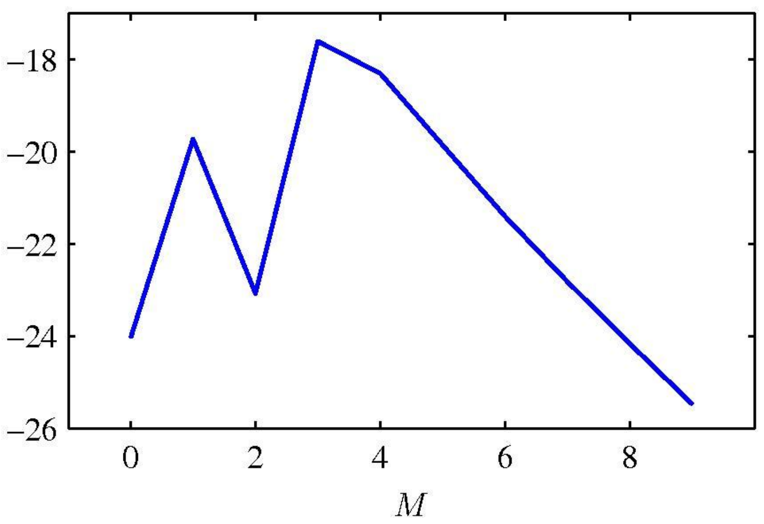
This is known as Empirical Bayes or type II maximum likelihood

Model evidence and cross-validation

- Consider the degree of polynomial regression



Root-mean-square error



Model evidence

Outline

- Linear models for regression
- Linear models for classification
  - Logistic regression
  - Probit regression
  - Multi-class regression
  - Ordinal regression
- General linear models

Logistic regression

- Let us first consider binary classification problem: C_1, C_2

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\right)$$

$$\sigma(a) = 1/\big(1 + \exp(-a)\big)$$

Logistic sigmoid function

$$p(\mathcal{C}_2|\boldsymbol{\phi}) = 1 - p(\mathcal{C}_1|\boldsymbol{\phi})$$

Logistic regression

- Interesting property of sigmoid function

$$\frac{d\sigma}{da} = \sigma(1 - \sigma).$$

Logistic regression

- Given a dataset $\{\phi_n, t_n\}$, where $t_n \in \{0,1\}$, $\phi_n = \phi(\mathbf{x}_n)$ and $n = 1, \ldots, N$, the likelihood function is given by

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n}$$

$$\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$$

$$y_n = p(\mathcal{C}_1|\phi_n) = \sigma(\mathbf{w}^\top \phi_n)$$

Logistic regression

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

$$\downarrow$$

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n) \boldsymbol{\phi}_n$$

Iterative reweighted least squares

- Newton-Raphson scheme

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \mathbf{H}^{-1}\nabla E(\mathbf{w})$$

Hessian matrix

Iterative reweighted least squares

- First consider linear model for regression

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^\top \boldsymbol{\phi}_n\}^2$$

$$\nabla E(\mathbf{w}) \;=\; \sum_{n=1}^{N}(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\mathbf{w} - \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}$$

Iterative reweighted least squares

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\mathbf{w} - \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N}\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$$

$$\begin{aligned}\mathbf{w}^{(\mathrm{new})} &= \mathbf{w}^{(\mathrm{old})} - (\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\left\{\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\mathbf{w}^{(\mathrm{old})} - \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}\right\} \\ &= (\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}\end{aligned}$$

The same as least square solution!

One step solves it! Why?

Iterative reweighted least squares

- Logistic regression

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

$$\begin{aligned}
\nabla E(\mathbf{w}) &= \sum_{n=1}^{N} (y_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^{\mathrm{T}}(\mathbf{y} - \mathbf{t}) \\
\mathbf{H} &= \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(1 - y_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}} = \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{R}\boldsymbol{\Phi}
\end{aligned}$$

N x N diagonal matrix $R_{nn} = y_n(1 - y_n) \quad y_n = \sigma(\mathbf{w}^\top \boldsymbol{\phi}_n)$

Iterative reweighted least squares

$$\begin{aligned}
\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - (\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{R}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}(\mathbf{y}-\mathbf{t}) \\
&= (\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{R}\boldsymbol{\Phi})^{-1}\left\{\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{R}\boldsymbol{\Phi}\mathbf{w}^{(\text{old})} - \boldsymbol{\Phi}^{\mathrm{T}}(\mathbf{y}-\mathbf{t})\right\} \\
&= (\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{R}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{z}
\end{aligned}$$

Iterative updates

$$\underbrace{\mathbf{z} = \boldsymbol{\Phi}\mathbf{w}^{(\text{old})} - \mathbf{R}^{-1}(\mathbf{y}-\mathbf{t})}_{\text{Updated responses}}$$

Weight matrix $\mathbf{R}$ depends on $\mathbf{w}$

Multiclass logistic regression

- Suppose we have *K* classes, C_1, …, C_K

$$p(\mathcal{C}_k|\boldsymbol{\phi}) = y_k(\boldsymbol{\phi}) = \frac{\exp(a_k)}{\sum_j \exp(a_j)} \qquad a_k = \mathbf{w}_k^{\mathrm{T}}\boldsymbol{\phi}$$

K groups of parameters $\{\mathbf{w}_k\}$

This is often referred to as softmax

$$\frac{\partial y_k}{\partial a_j} = y_k(I_{kj} - y_j)$$

Multiclass logistic regression

- likelihood

$$p(\mathbf{T}|\mathbf{w}_1, \ldots, \mathbf{w}_K) = \prod_{n=1}^{N} \prod_{k=1}^{K} p(\mathcal{C}_k|\boldsymbol{\phi}_n)^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

T: N x K observation matrix, each row is one-hot vector

Multiclass logistic regression

- We can use Newton-Raphson updates as well

$$\nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = \sum_{n=1}^{N} (y_{nj} - t_{nj}) \, \boldsymbol{\phi}_n$$

$$\nabla_{\mathbf{w}_k} \nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = -\sum_{n=1}^{N} y_{nk} (I_{kj} - y_{nj}) \boldsymbol{\phi}_n \boldsymbol{\phi}_n^{\mathrm{T}}.$$

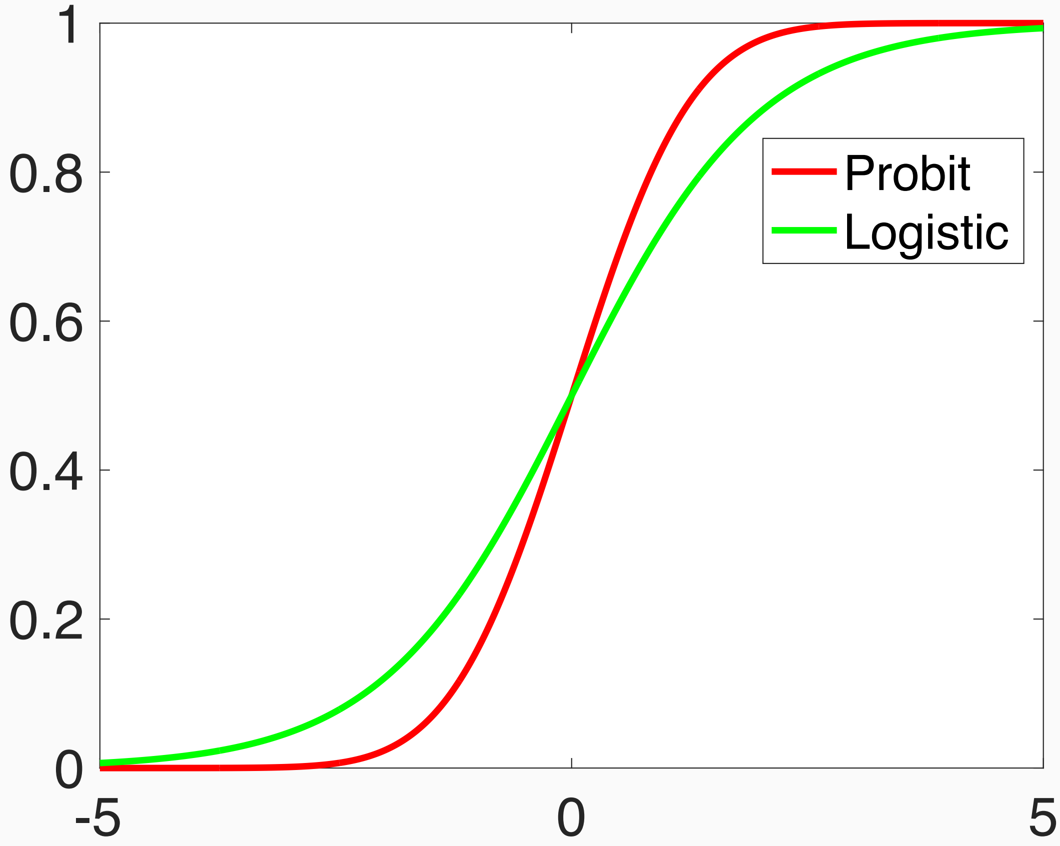
Probit regression

- An alternative model for binary classification

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \psi(\mathbf{w}^\top\boldsymbol{\phi})$$

$$\psi(a) = \int_{\infty}^{a} \mathcal{N}(x|0,1)\mathrm{d}x$$

Probit function vs. logistic function



Probit regression

- Equivalent latent variable model

Given $a = \mathbf{w}^\top \boldsymbol{\phi}$

sample the label *t* from $p(t|a) = \psi(a)^t \big(1 - \psi(a)\big)^{1-t}$

$\Updownarrow$

Sample a latent variable z from

$$z \sim \mathcal{N}(z|a, 1)$$

Sample the label t from a step distribution

$$p(t|z) = I(t=0)I(z \leq 0) + I(t=1)I(z \geq 0)$$

Ordinal regression

- Consider to predict K classes with *ordering* relationship, $C_1 < C_2 <\ldots< C_K$, e.g., rank, disease progression, …
- Using multi-class logistic regression is not appropriate

Ordinal regression

- Consider multi-class Probit regression

Partition real domain into ordered regions

$$(\infty, b_1], (b_1, b_2], \ldots, (b_{K-1}, b_K], (b_K, \infty)$$

Given $a = \mathbf{w}^\top \boldsymbol{\phi}$

Sample a latent variable z from $z \sim \mathcal{N}(z|a, 1)$

Check which region *z* falls in, e.g., $[b_k, b_{k+1})$

Output the corresponding label: *k*

Generalized linear models

- Let us consider the exponential family to model data

$$p(t|\eta) = \exp\left(\eta t - g(\eta)\right)$$

Consider the expectation of t

$$\mathbb{E}[t|\eta] = y = \frac{\mathrm{d}g(\eta)}{\mathrm{d}\eta}$$

This is a mapping $\eta = \psi(y)$

From expectation to natural parameters

Generalized linear models

- In linear model, we commonly model the expectation parameters as

$$y = f\left(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x})\right)$$

- If we choose $f = \psi^{-1}$ $\quad$ $\eta = \psi(y)$

$$\eta = \psi(\psi^{-1}(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}))) = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x})$$

f^{-1} is called link function (link expectation to natural paras)

Generalized linear models

- Given training data $(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)$

$$E(\mathbf{w}) = \sum_{n=1}^{N} \log p(t_n|\eta)$$
$$= \sum_{n=1}^{N} \eta_n t_n - g(\eta_n)$$

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = \sum_{n=1}^{N} \frac{\partial \eta_n}{\partial \mathbf{w}} t_n - \frac{\partial g}{\partial \eta_n} \frac{\partial \eta_n}{\partial \mathbf{w}}$$

Generalized linear models

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = \sum_{n=1}^{N} \frac{\partial \eta_n}{\partial \mathbf{w}} t_n - \frac{\partial g}{\partial \eta_n} \frac{\partial \eta_n}{\partial \mathbf{w}}$$

$$= \sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}_n)(t_n - y_n)$$

Feature vector

prediction error

$$\mathbb{E}[t_n|\eta_n] = y_n = \frac{\mathrm{d}g(\eta_n)}{\mathrm{d}\eta_n}$$

$$\eta_n = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_n)$$

This is consistent with linear regression and logistic regression

What you should know

- What is design matrix?
- How to obtain MLE for linear regression?
- How to obtain posterior and predictive distribution for linear regression?
- What is the empirical Bayes and type II MLE?
- Newton-Rapson method for logistic regression
- What is probit regression? What is the equivalent model? How to conduct multi-class classification?
- What is generalized linear model? What is link function? What is the general form of the gradient?