

線形回帰と最小二乗法

岡山大学 異分野基礎科学研究所

大槻純也



前回の復習

ベイズの定理

事後確率
posterior probability

事前確率
prior probability

$$p(\boldsymbol{w}|\mathcal{D}) \propto p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})$$

$\boldsymbol{w}$: モデルパラメータ
知りたいもの

$\mathcal{D}$: 観測データ
そのために測定するもの

測定により確率が更新される

ここまでの例では、 $p(\mathcal{D}|\boldsymbol{w})$ は統計データとして与えられていた
物理で応用する場合は、モデルを仮定して $p(\mathcal{D}|\boldsymbol{w})$ を計算する
(モデル=関数形、ハミルトニアン)

補足： $p(\mathcal{D}|\boldsymbol{w})$ の意味

$p(\mathcal{D}|\boldsymbol{w})$

本来の意味はprobability of $\mathcal{D}$ given $\boldsymbol{w}$

つまり、モデル $\boldsymbol{w}$ が与えられた下で、測定結果 $\mathcal{D}$ が得られる確率

これを $\boldsymbol{w}$ の関数とみる ($\mathcal{D}$ は固定)

このとき $p(\mathcal{D}|\boldsymbol{w})$ を「尤度関数 (likelihood function)」と呼ぶ

測定で得られた $\mathcal{D}$ に対して、
 $\boldsymbol{w}$ の値がどの程度尤もらしいか

注意：確率ではない
実際、規格化されていない

最尤推定 (Maximum likelihood)

Likelihood関数を最大化

$$\boldsymbol{w}_{\text{ML}} = \arg \max_{\boldsymbol{w}} p(\mathcal{D}|\boldsymbol{w})$$

最小二乗法に対応 (後で確認)

$$p(\boldsymbol{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})}{p(\mathcal{D})}$$

本来はこれを見るべき

問題点

1. prior probabilityを無視
ex. 「罹患率」の例題で致命的
2. $\boldsymbol{w}$ の分布を無視
実際は $\boldsymbol{w}$ は確率として与えられている

今回からやること

データのフィッティングをしたいとき

→ 最小二乗法を使う $t = f(x)$

ただし、関数形が分かっていることが前提

関数形が分からない場合は？

→ 人間がグラフを見て関数形を推測

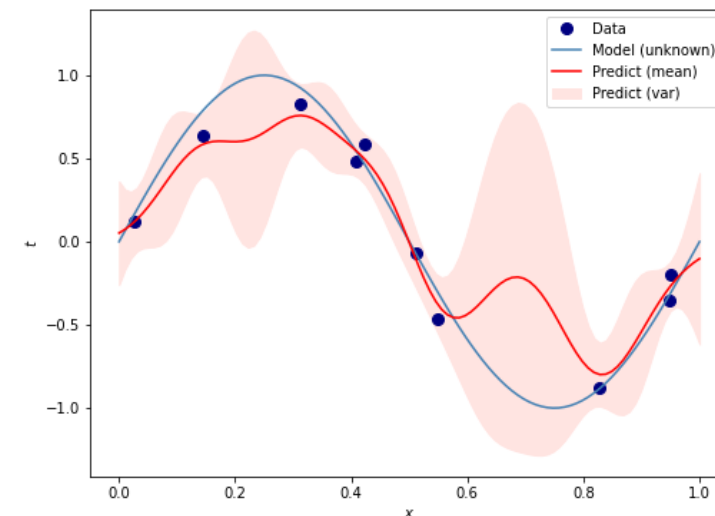
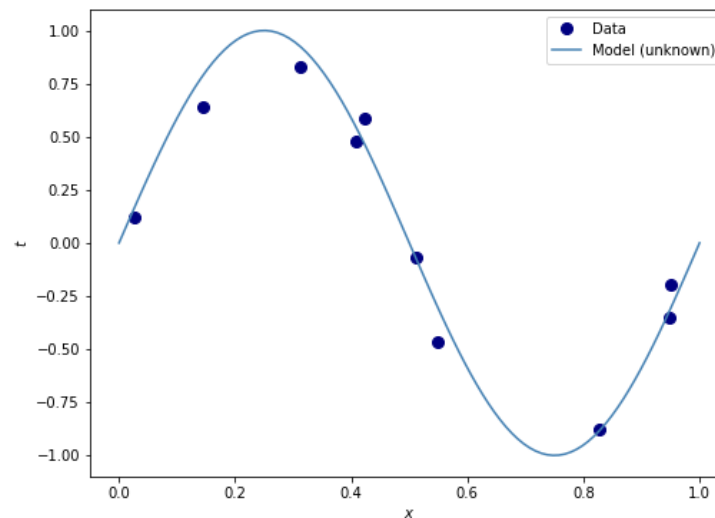
多次元の場合は？ $t = f(x_1, x_2, \dots, x_D)$

→ がんばってグラフを描く

or あきらめる

ベイズの定理を応用して、 $p(\mathbf{w}|\mathcal{D})$ から自動的に

- ・ 関数形を決める
- ・ 予測する、測定点を決める



用語説明：回帰 (Regression)

回帰 (Regression) とは：

データの組 $\{(x_n, t_n)\}$ が与えられたときに、
任意の x に対して t を予測する関数 $t = y(x)$ を見つけること

簡単に言えばフィッティング。ただし

- 関数形は未知 (関数形も含めて決定する)
- x は多次元 $\mathbf{x} = (x_1, x_2, \dots, x_D)$

機械学習の言葉を使うと

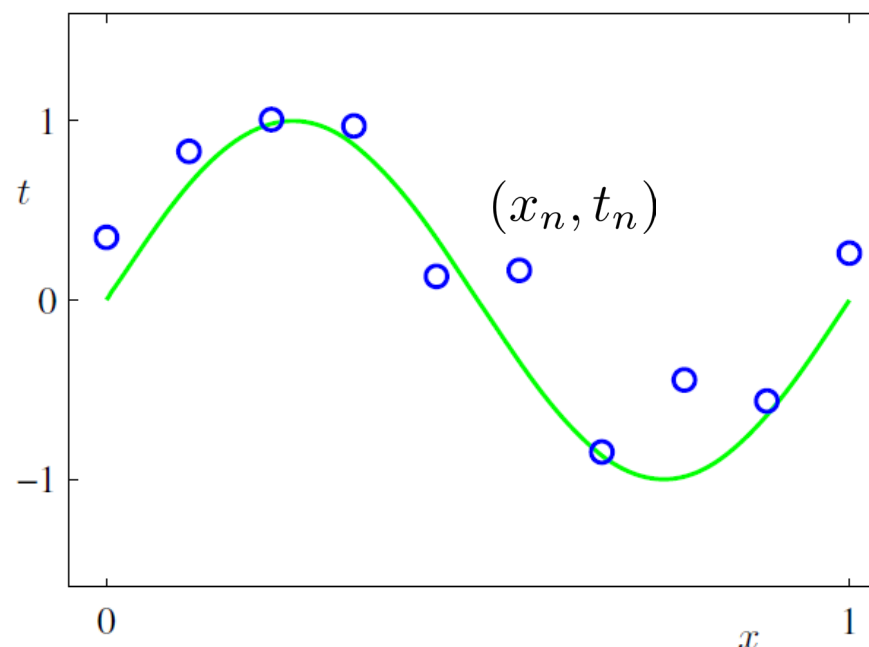
- ▶ 教師あり学習
 - t が連続 $\rightarrow$ 回帰 (Regression)
 - t が不連続 $\rightarrow$ 分類 (Classification)
- ▶ 教師なし学習

Training set

$$\mathbf{x} = \{x_1, x_2, \dots, x_N\}$$

$$\mathbf{t} = \{t_1, t_2, \dots, t_N\} \text{ target}$$

表記はPRMLに従う



PRML Fig. 1.2

例：多項式フィッティング

まずは関数 $y(x)$ として多項式 (polynomial) を考える

$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \cdots + w_Mx^M$$

$$= \sum_{j=0}^M w_j x^j$$

パラメーター $\{w_j\}$ に関して線形 (linear)

→ 線形回帰 (Linear regression)

入力 x に対しては非線形 (nonlinear)

一般的には

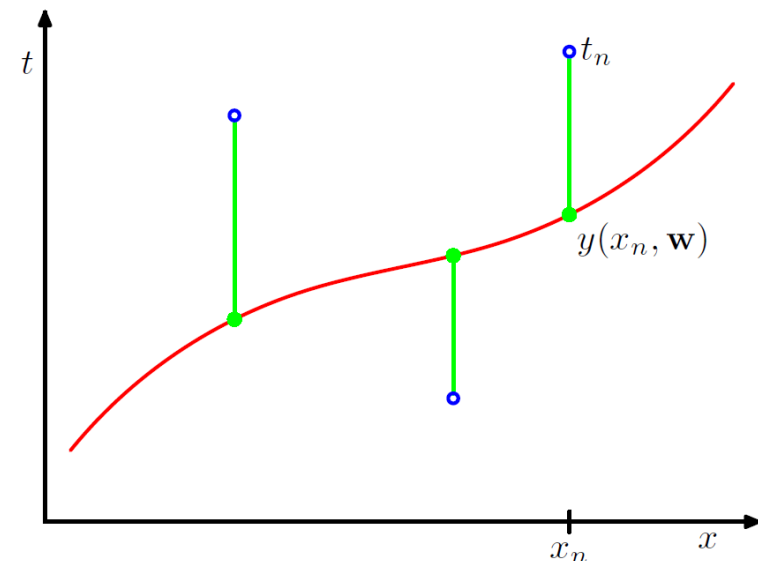
$$y(x, \mathbf{w}) = \sum_{j=0}^M w_j \phi_j(x)$$

基底関数 (basis function)
非線形

誤差関数 (Error function)

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N [y(x_n, \mathbf{w}) - t_n]^2$$

関数 $y(x, \mathbf{w})$ がデータ $\{(x_n, t_n)\}$ をどの程度表現できてるか



最小二乗法 (Least squares)

誤差関数 (Error function)

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N [y(x_n, \mathbf{w}) - t_n]^2$$

$$y(x, \mathbf{w}) = \sum_{j=0}^M w_j x^j$$

誤差関数を最小化

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} E(\mathbf{w})$$

関数を最小にする $\mathbf{w}$ を返す演算子
(argument of the minimum)

予測曲線

$$t = y(x, \mathbf{w}^*)$$

argminを計算すると

$$\mathbf{w}^* = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{t} \equiv \Phi^+ \mathbf{t}$$

$$\mathbf{t} = (t_1, t_2, \dots, t_N)^T \quad \text{Tは転置}$$

$$\Phi = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^M \\ 1 & x_2 & x_2^2 & \cdots & x_2^M \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^M \end{pmatrix}$$

$N \times M$ 行列

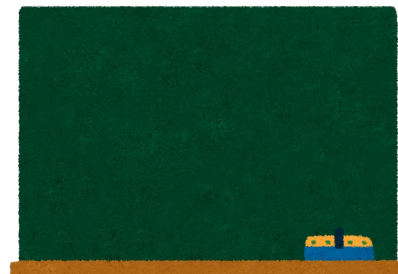
計画行列
design matrix

$$\Phi^+ = (\Phi^T \Phi)^{-1} \Phi^T$$

ムーア・ペンローズの疑似逆行列
Moore-Penrose pseudo inverse

$\mathbf{t} = \Phi \mathbf{w}$ を $\mathbf{w}$ について解く
ことに相当するため

ベクトルの微分を使うと便利 (次頁)



公式：ベクトル微分

成分ごとの微分ではなく、**ベクトルでの微分**を使うと計算が楽になる

定義

$$\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}) \equiv \begin{pmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_N} \end{pmatrix}$$

公式

$$\frac{\partial}{\partial \mathbf{x}} (\mathbf{x}^T \mathbf{y}) = \frac{\partial}{\partial \mathbf{x}} (\mathbf{y}^T \mathbf{x}) = \mathbf{y}$$

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

A は $(N \times N)$ 行列

線形回帰では2次形式しか現れないのでこれで十分

誤差関数を行列・ベクトル表記する

$$\begin{aligned} E(\mathbf{w}) &= \frac{1}{2} \sum_{n=1}^N [y(x_n, \mathbf{w}) - t_n]^2 \\ &= \frac{1}{2} (\Phi \mathbf{w} - \mathbf{t})^T (\Phi \mathbf{w} - \mathbf{t}) \end{aligned} \quad \text{Tは転置}$$

$$y(x, \mathbf{w}) = \sum_{j=0}^M w_j x^j = \underline{\mathbf{w}^T \phi(x)} = \underline{\phi^T \mathbf{w}(x)}$$

ベクトル同士の内積

ここで

$$\phi(x) = \begin{pmatrix} 1 \\ x \\ \vdots \\ x^M \end{pmatrix}$$

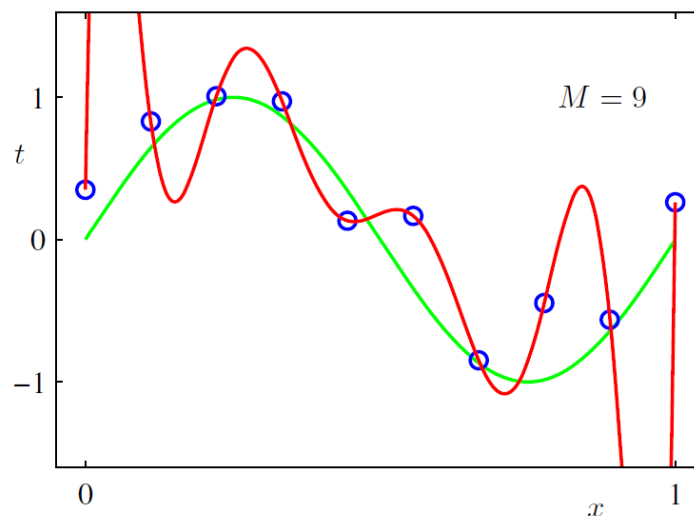
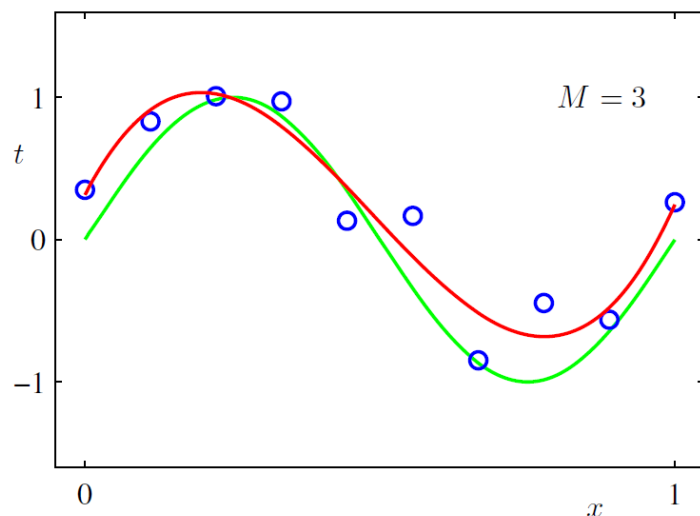
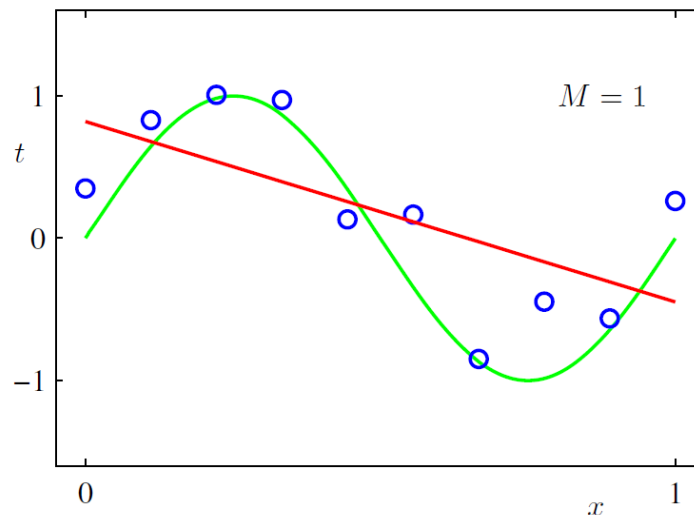
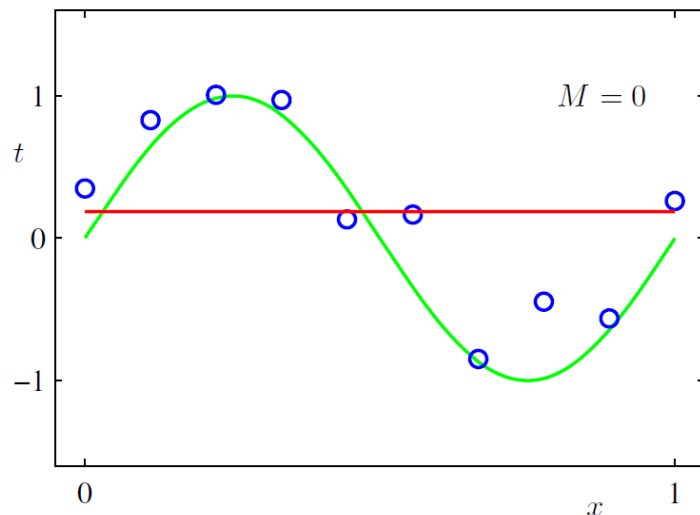
一般的には

$$\phi(x) = \begin{pmatrix} \phi_0(x) \\ \phi_1(x) \\ \vdots \\ \phi_M(x) \end{pmatrix}$$

ベクトルは全て**縦ベクトル**として表す

例：多項式フィッティング

$N=10$
 $M=[0:9]$



$M=3$ くらいが妥当

$M=9$ は明らかにおかしい
Overfittingと言う
(過学習、過剰適合)

→ $E(\mathbf{w}^*)$ が小さければ小さい
ほど良いわけではない

青丸：測定データ
緑：正解
赤：予測曲線

どうやってoverfittingを避けるか

テストデータを利用する

$$\begin{aligned} \text{Test set } \mathbf{x}' &= \{x'_1, x'_2, \dots, x'_{N'}\} \\ \mathbf{t}' &= \{t'_1, t'_2, \dots, t'_{N'}\} \end{aligned}$$

2つの **Root-Mean-Square (RMS) error** (平均二乗偏差) を定義

Training error

$$E_{\text{RMS}}^{\text{Training}} = \sqrt{\frac{1}{N} \sum_{n=1}^N [y(x_n, \mathbf{w}^*) - t_n]^2} = \sqrt{\frac{2}{N} E(\mathbf{w}^*)}$$

フィッティングの精度を表す

Test error

$$E_{\text{RMS}}^{\text{Test}} = \sqrt{\frac{1}{N'} \sum_{n=1}^{N'} [y(x'_n, \mathbf{w}^*) - t'_n]^2}$$

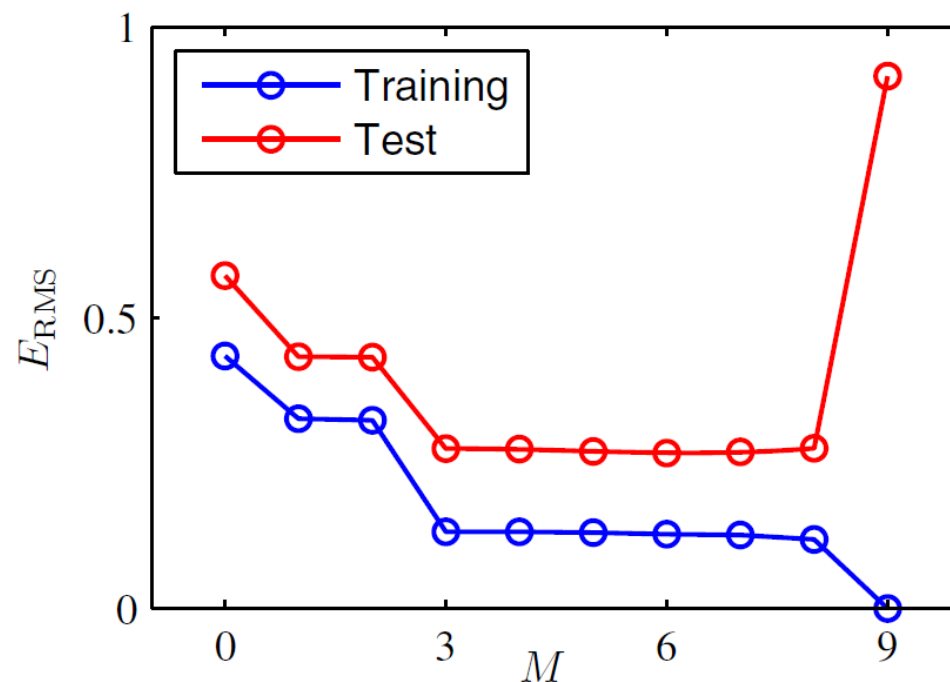
汎化性能を表す

問題点：testデータが必要

testデータがない場合には、trainingデータを減らしてtestに使う（trainingデータが減ってしまう）。これを系統的に行う方法を**交差検証法**と呼ぶ。

generalization (汎化) :

学習に使ったデータとは異なるデータも正しく予測できること。汎化性能。



PRML Fig. 1.5

もう一つの方法：正則化

正則化 (Regularization) :

誤差関数に項を付け加えて、解を制約すること

$$\begin{aligned}
 E(\mathbf{w}) &= \frac{1}{2} \sum_{n=1}^N [y(x_n, \mathbf{w}) - t_n]^2 + \frac{\lambda}{2} \sum_j |w_j|^2 \\
 &= \frac{1}{2} (\Phi \mathbf{w} - \mathbf{t})^T (\Phi \mathbf{w} - \mathbf{t}) + \frac{\lambda}{2} \mathbf{w}^T \mathbf{w}
 \end{aligned}$$

$\mathbf{w}$ のノルムにペナルティを課す

λ は **hyper parameter** と呼ばれる

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = 0 \quad \text{より}$$

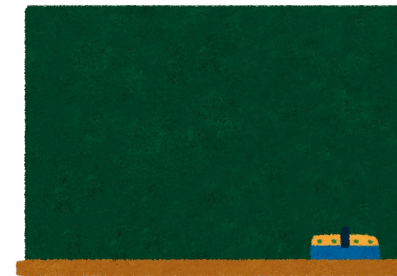
$$\mathbf{w}^* = (\Phi^T \Phi + \lambda \mathbf{I})^{-1} \Phi^T \mathbf{t}$$

リッジ回帰
Ridge regression

最小二乗解との違い ($\mathbf{I}$ は単位行列)
逆行列計算を安定化させる (ランク落ちを防ぐ)

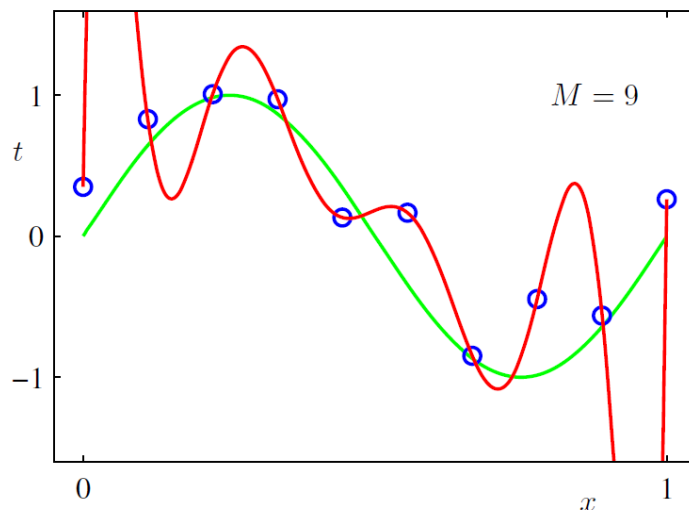
	$M = 0$	$M = 1$	$M = 6$	$M = 9$
w_0^*	0.19	0.82	0.31	0.35
w_1^*		-1.27	7.99	232.37
w_2^*			-25.43	-5321.83
w_3^*			17.37	48568.31
w_4^*				-231639.30
w_5^*				640042.26
w_6^*				-1061800.52
w_7^*				1042400.18
w_8^*				-557682.99
w_9^*				125201.43

PRML Table 1.1

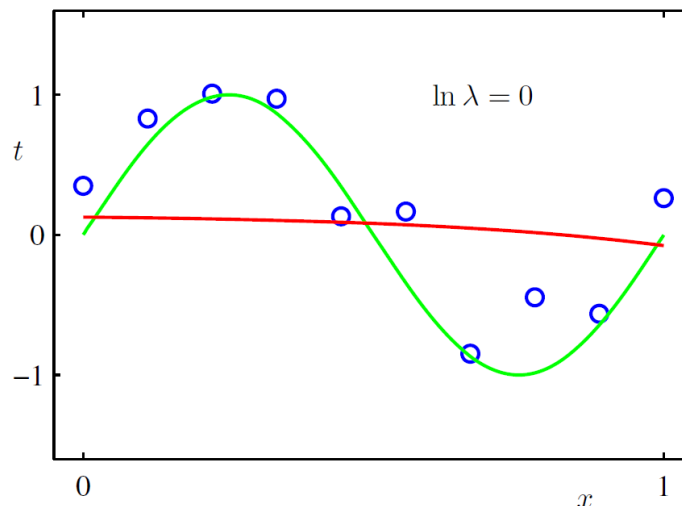
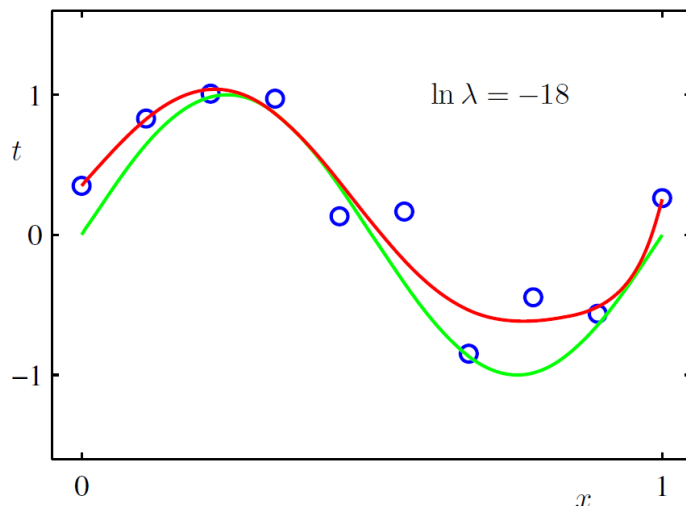


結果

正則化なし
($\ln \lambda = -\infty$)



正則化あり



PRML Fig. 1.7

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

PRML Table 1.2

overfittingが抑えられている

ただし、 λ が大きすぎるとデータの特徴が消えてしまう

適切な λ の値をどのように決めるかという問題が残る。とりあえずは、 $\lambda = 10^{-8}$ のように非常に小さくとる。