University of Amsterdam
Informatics Institute

AMLAB
Amsterdam Machine Learning Lab

Machine Learning 1

Lecture 4 - Model Selection - Bias Variance Decomposition - Gaussian Posteriors - Sequential Bayesian Learning - Bayesian Predictive Distributions

Erik Bekkers

Image credit: Kirillm | Getty Images

University of Amsterdam
Informatics Institute

AMLAB Amsterdam Machine Learning Lab

Machine Learning 1

Lecture 3.5 - Supervised Learning

Regularized Least Squares

Erik Bekkers

(Bishop 3.1.4)

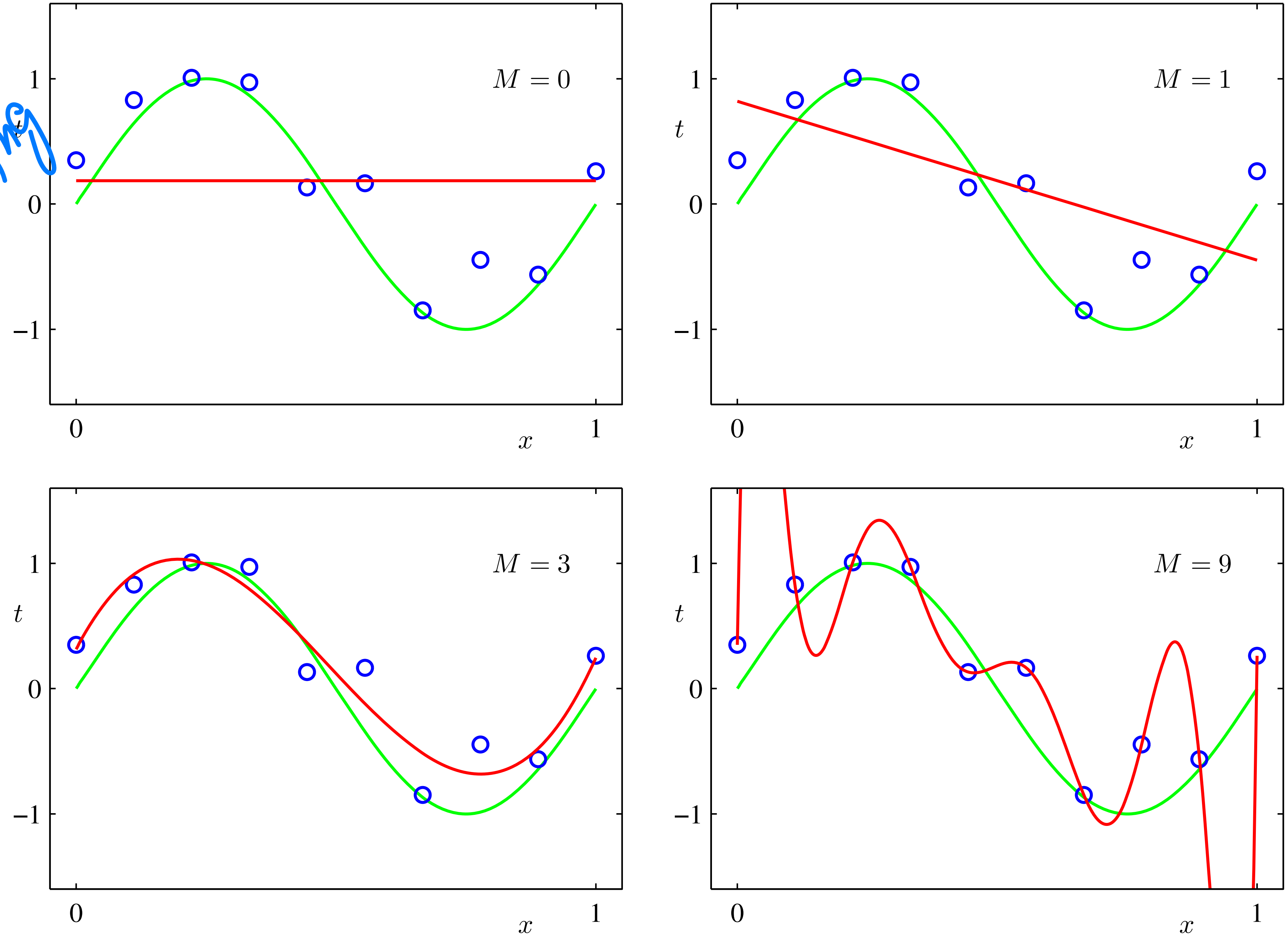
Image credit: Kirillm | Getty Images

Slide credits: Patrick Forré and Rianne van den Berg

Example: Overfitting and Underfitting

$$t = \sin(2\pi x) + \varepsilon \qquad \varepsilon \sim \mathcal{N}(0, \beta^{-1}) \qquad y(x, \mathbf{w}) = w_0 + \sum_{i=1}^{M} w_i x^i$$

underfitting



overfitting

Figure: Fits of different polynomials (Bishop 1.4)

Example: Overfitting (M=9)

	$M = 0$	$M = 1$	$M = 3$	$M = 9$
$w_0^\star$	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

Table: Polynomial coefficients (Bishop 1.1)

Regularized Least Squares

- Instead of manually constraining the number of parameters for small datasets, add penalty term for large parameter values:

$$\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{t_i - y(\mathbf{x}_i, \mathbf{w})\}^2 \quad + \frac{\lambda}{2}\sum_{i=1}^{M} w_i^2$$

- Ridge regression
- ℓ_2 regularization
- weight decay

- The bias term w_0 is often not included in regularization

Parameter estimates with Gaussians

- Given **Likelihood**/Data model:

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

- The ***ML parameter estimate*** is obtained via **least squares**:

$$\mathbf{w}_{ML} = \underset{\mathbf{w}}{\operatorname{argmin}} \; \frac{\beta}{2} \sum_{i=1}^{N} (y(x_i, \mathbf{w}) - t_i)^2$$

- Additionally, given Gaussian weight **Prior**:

When $\lambda = \frac{\alpha}{\beta}$

$$p(\mathbf{w} \mid \alpha) = \prod_{i=1}^{M} \mathcal{N}(w_i \mid 0, \alpha^{-1})$$

$\rightarrow$ Ridge

- The **MAP parameter estimate** is obtained via **regularized least squares**:

$$\mathbf{w}_{MAP} = \underset{\mathbf{w}}{\operatorname{argmin}} \; \frac{\beta}{2} \sum_{i=1}^{N} (y(x_i, \mathbf{w}) - t_i)^2 + \frac{\alpha}{2} \mathbf{w}^T \mathbf{w}$$

$\hookrightarrow \frac{\alpha}{2} \sum_i w_i^2$

Example: Regularized Polynomial Regression

no regularization

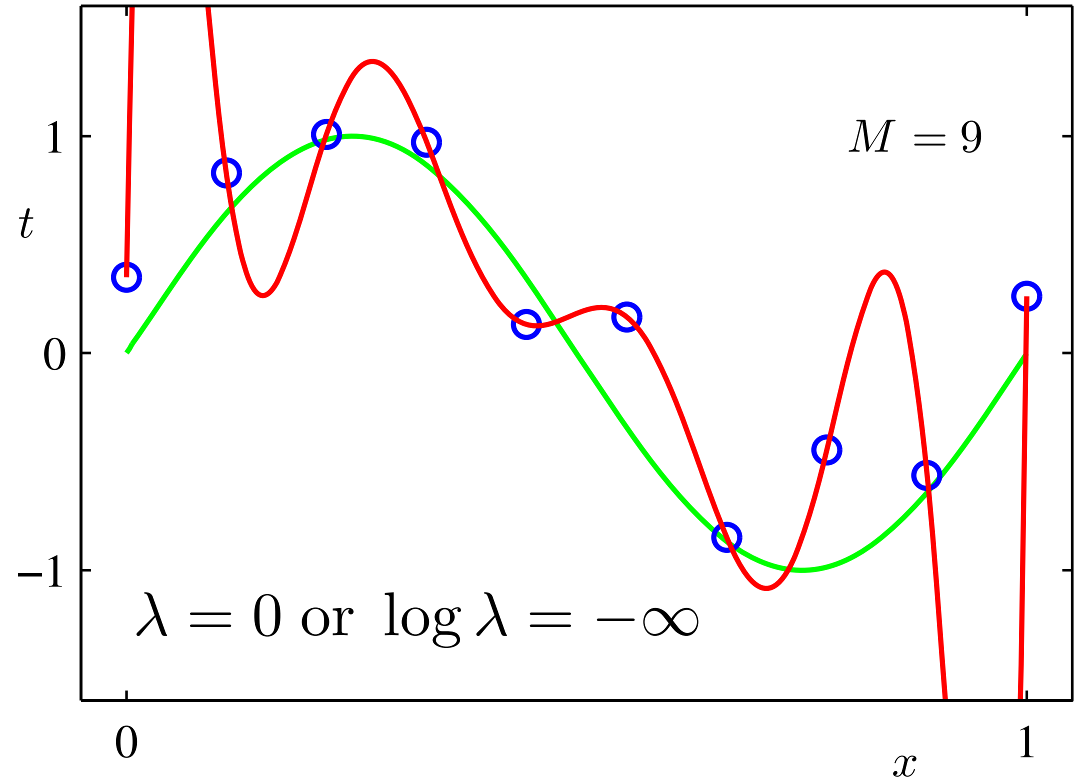


Figure: polynomial regression (Bishop 1.4)

good λ

large λ

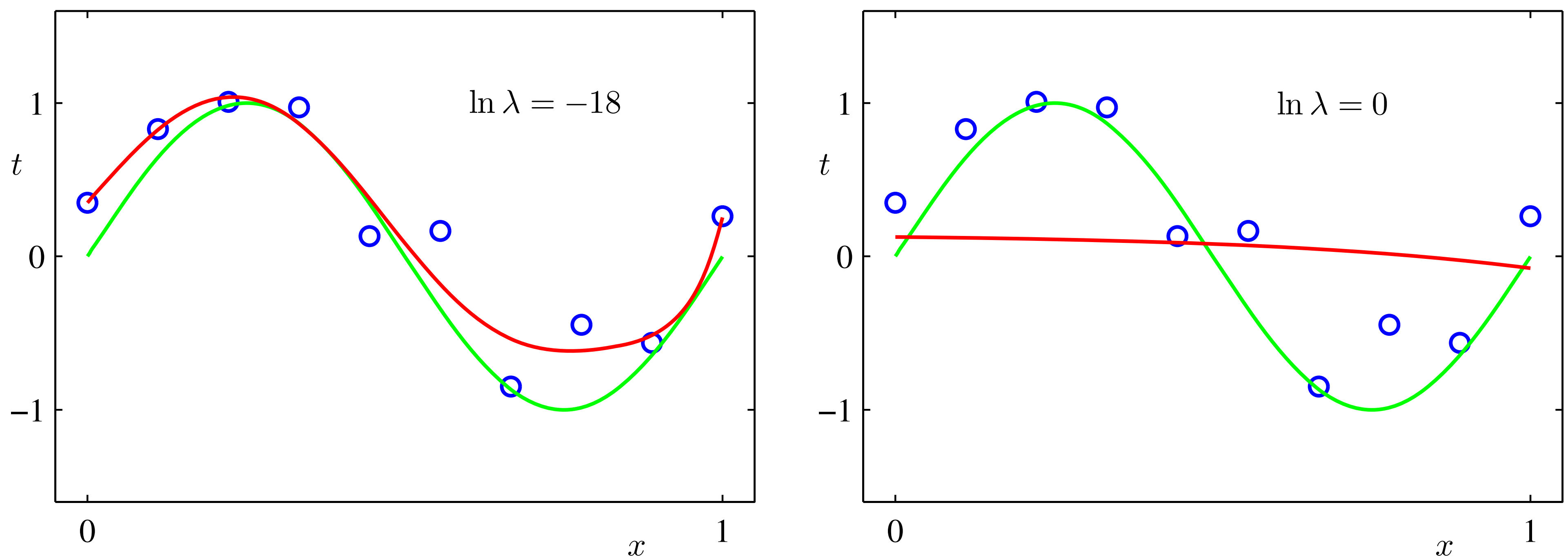


Figure: Regularized polynomial regression (Bishop 1.7)

Example: Regularized Polynomial Regression

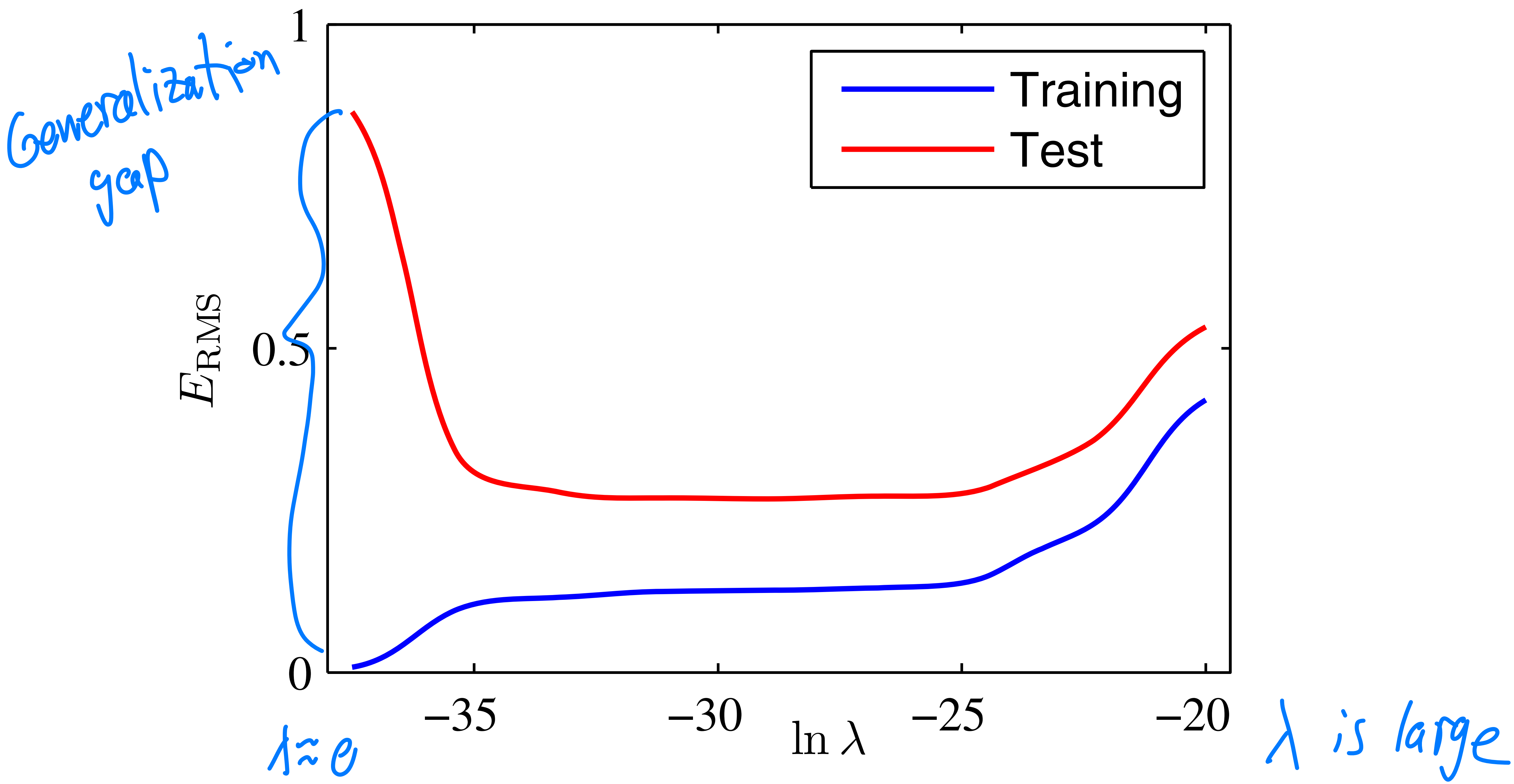


Figure: train and test errors for regularized M=9 polynomial regression (Bishop 1.8)

Regularized Least Squares (II)

- Weight decay: $\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{t_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)\}^2 + \frac{\lambda}{2}\sum_{i=1}^{M-1}|w_i|^2$

- General penalty:

$$\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{t_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)\}^2 + \frac{\lambda}{2}\sum_{i=1}^{M-1}|w_i|^q$$

- Case $q = 1$: Lasso

$+\frac{\lambda}{2}\sum_{i=1}|w_i|$

- sparsification

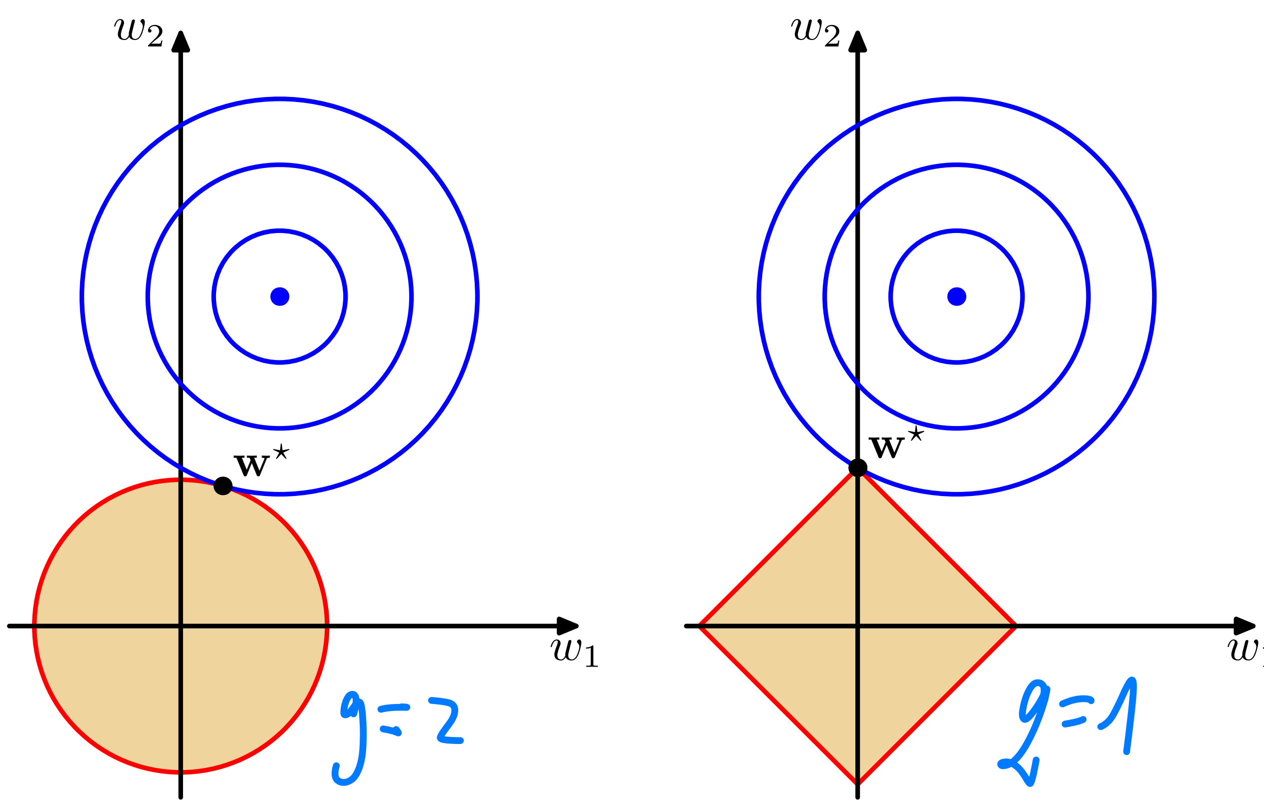


Figure: regularization as constrained optimization (Bishop 3.4)

- Equivalent to some constraint optimization problem (Bishop App. E)

minimize $\frac{1}{2}\sum_{i=1}^{N}\{t_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)\}^2$ subject to constraint $\sum_{i=1}^{M}|w_i|^q \leq \eta$

Regularized Least Squares: sparse weights

minimize $\frac{1}{2}\sum_{i=1}^{N}\{t_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)\}^2$ subject to constraint $\sum_{i=1}^{M}|w_i|^q \leq \eta$

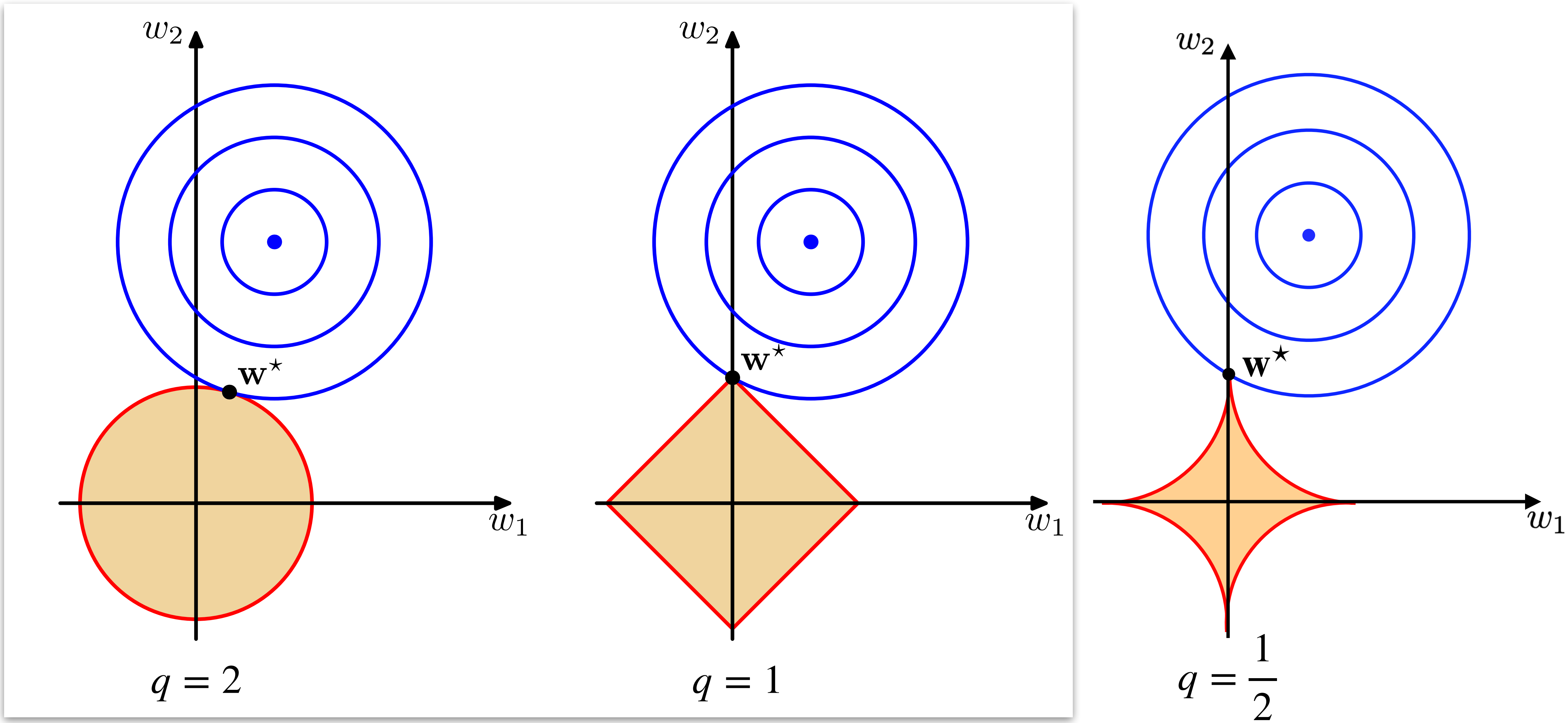


Figure: regularization as constrained optimization (Bishop 3.4) feature selection

Example: Prostate specific antigen predediction

q=2 (Ridge regression)

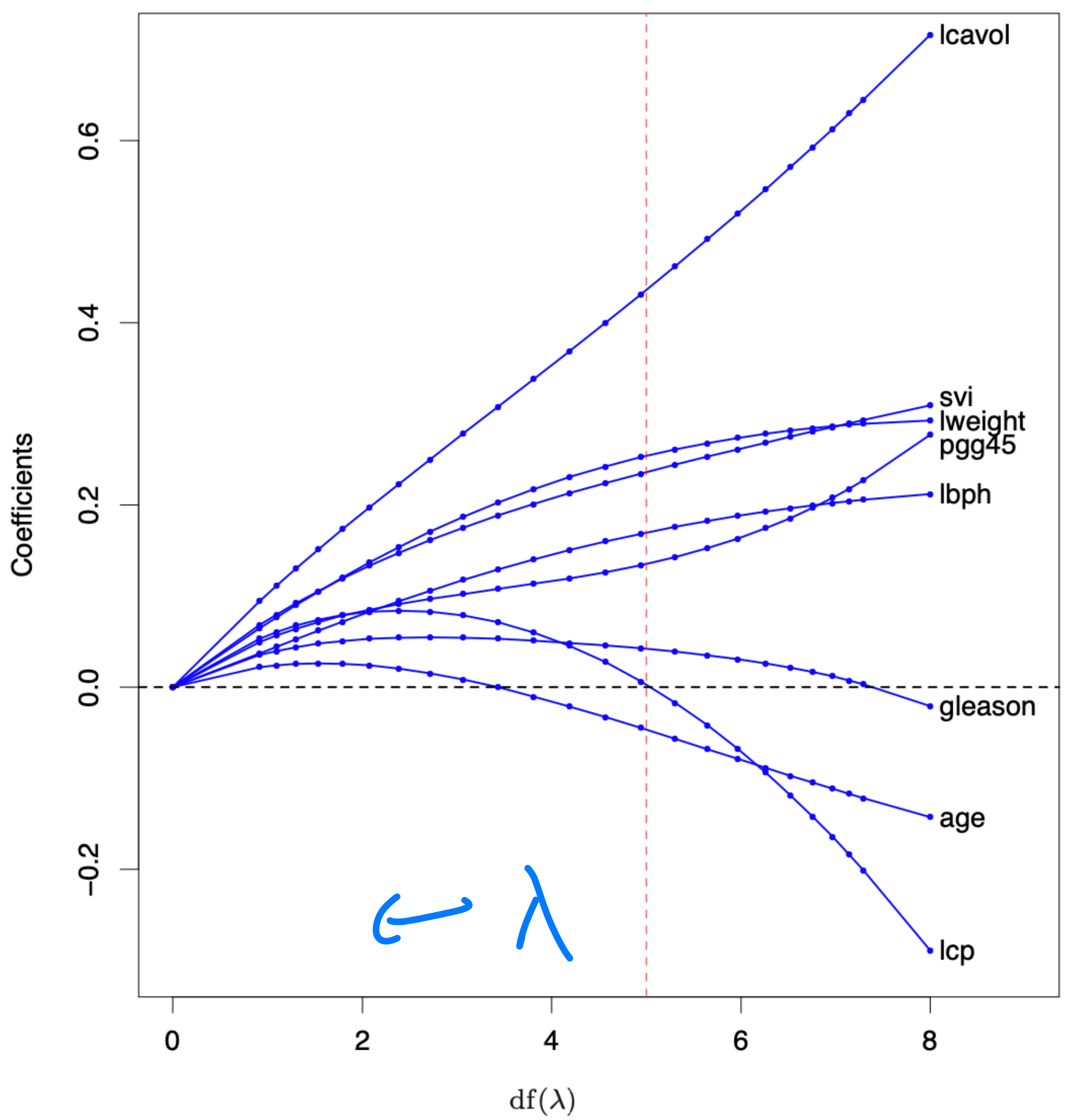


FIGURE 3.8. *Profiles of ridge coefficients for the prostate cancer example, as the tuning parameter λ is varied. Coefficients are plotted versus $\mathrm{df}(\lambda)$, the effective degrees of freedom. A vertical line is drawn at $\mathrm{df} = 5.0$, the value chosen by cross-validation.*

q=1 (Lasso regression)

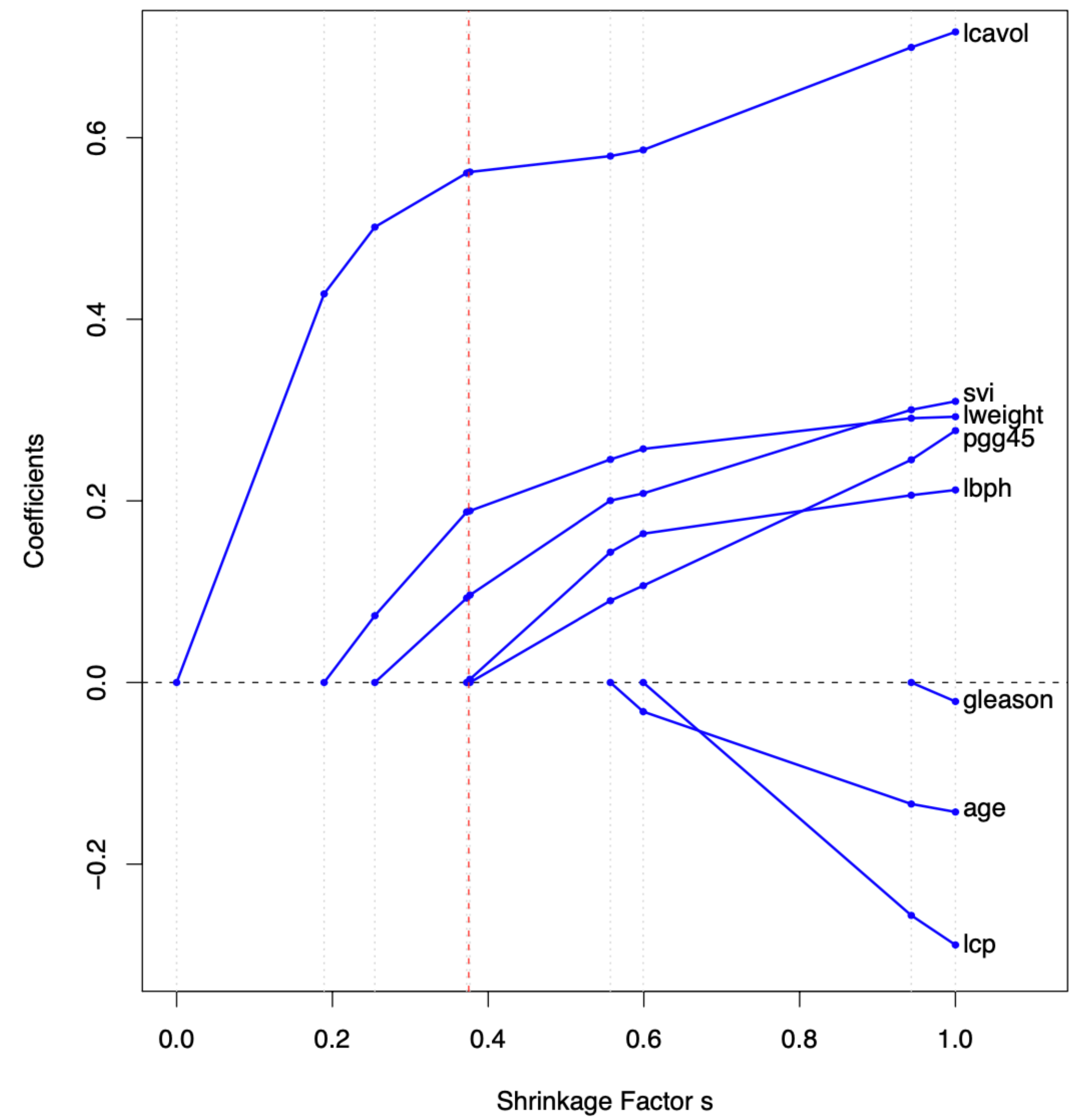


FIGURE 3.10. *Profiles of lasso coefficients, as the tuning parameter t is varied. Coefficients are plotted versus $s = t/\sum_1^p |\hat{\beta}_j|$. A vertical line is drawn at $s = 0.36$, the value chosen by cross-validation. Compare Figure 3.8 on page 65; the lasso profiles hit zero, while those for ridge do not. The profiles are piece-wise linear, and so are computed only at the points displayed; see Section 3.4.4 for details.*

Figures from the Elements of Statistical Learning (ESL - Hastie et al.)

University of Amsterdam
Informatics Institute

AMLAB Amsterdam Machine Learning Lab

Machine Learning 1

Lecture 4.1 - Supervised Learning

Model Selection

Erik Bekkers

(Bishop 1.3)

Slide credits: Patrick Forré and Rianne van den Berg

Image credit: Kirillm | Getty Images

Supervised Learning: Evaluating Errors

Q: How can we reliably estimate the model performance properly for unknown data?

Q: How can we choose the optimal hyperparameters?

Model selection | Dataset splits

Divide data $D = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)\}$ in 3 groups:

- Training set D_{train} (± 80 % of D):
  - Minimize the error $E(y(\mathbf{x}, \mathbf{w}), t)$ } obtain w^*
    for every $(\mathbf{x}, t) \in D_{\text{train}}$

model selection

- Validation set D_{val} (± 10 % of D):
  - Used to *estimate* generalization error $E(y(\mathbf{x}_{\text{val}}, \mathbf{w}^*), t_{\text{val}})$ } estimate performance
    for every $(\mathbf{x}_{\text{val}}, t_{\text{val}}) \in D_{\text{val}}$

- Test set D_{test} (± 10 % of D):
  - final test/generalization error estimate $E(y(\mathbf{x}_{\text{test}}, \mathbf{w}^*), t_{\text{test}})$
    for every $(\mathbf{x}_{\text{test}}, t_{\text{test}}) \in D_{\text{test}}$

cannot be used for model selection

Supervised Learning: Small Datasets

- Small dataset $\Rightarrow$ small validation and test set
- Noisy model selection and estimate of generalization error
- Cross-validation:
  - Split data $D = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)\}$ in to K folds
  - Train model y on $K-1$ folds (fold k left out) $\Rightarrow$ $\hat{y}^{-k}$

$D =$

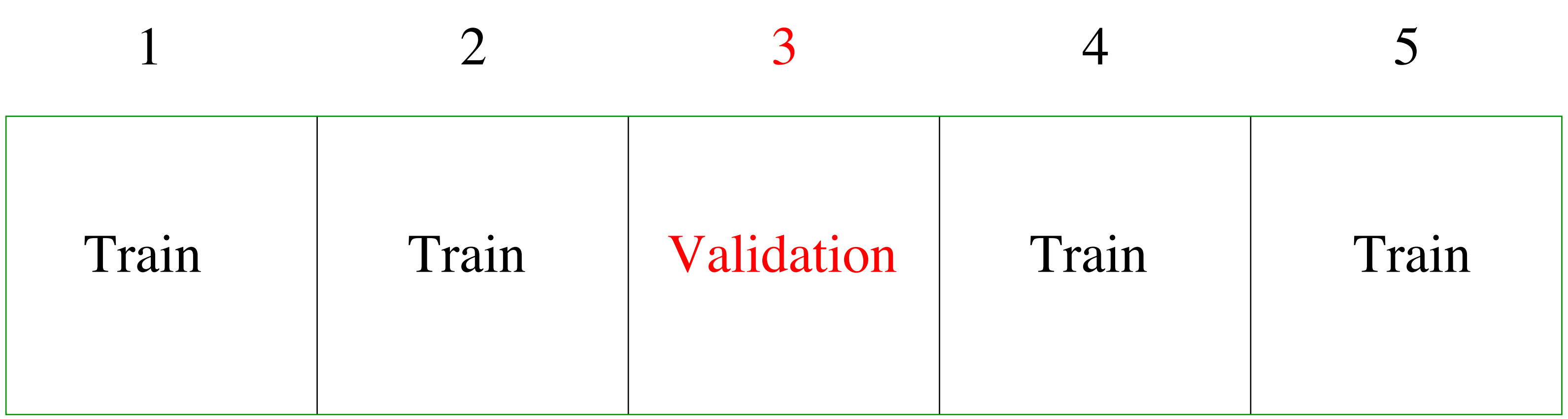


Figure: K-fold splitting of dataset (ESL 7.10)

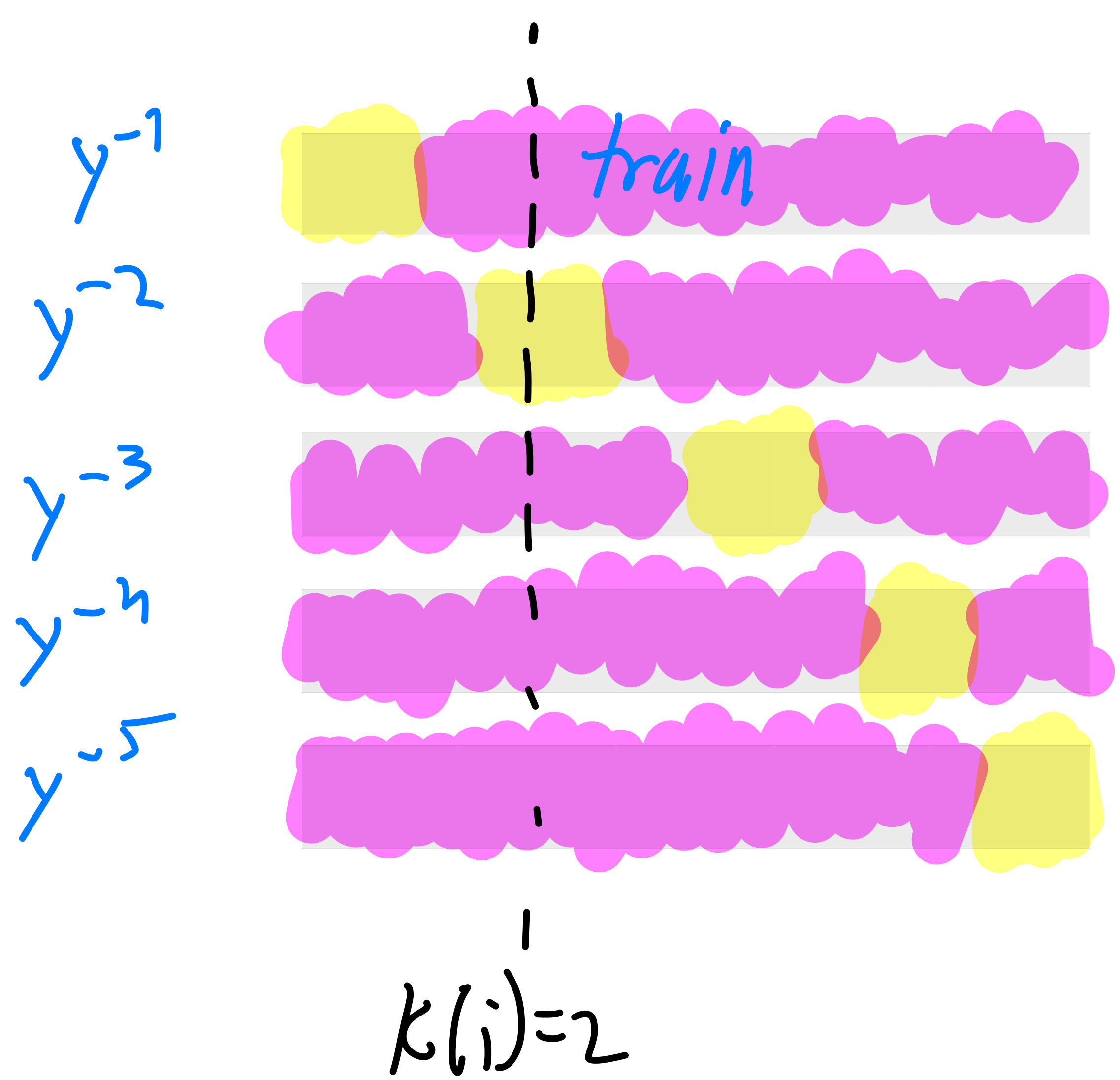
  - Leave-one-out cross validation: $K = N$

Cross-Validation

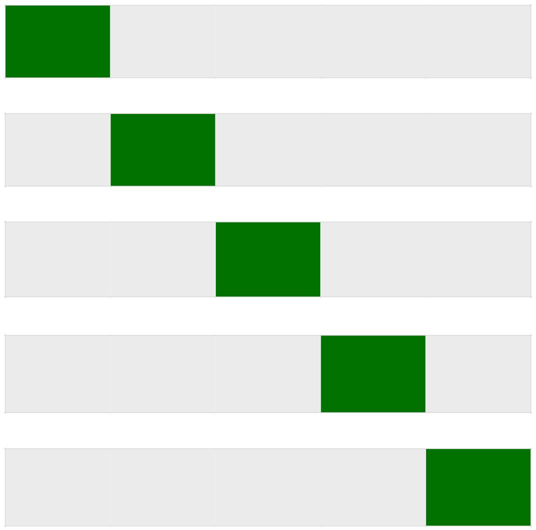
- K trained functions $\hat{y}^{-k}$
- Use indexing function $\kappa : \{1, \ldots, N\} \to \{1, \ldots, K\}$
- Cross validation error:

$$CV(\hat{y}) = \frac{1}{N} \sum_{i=1}^{N} E(\hat{y}^{-\kappa(i)}(\mathbf{x}_i), t)$$

model selection



Cross-Validation: Model Selection



- Hyperparameter selection:
  - $$CV(\hat{y}_\alpha) = \frac{1}{N}\sum_{i=1}^{N} E(\hat{y}_\alpha^{-\kappa(i)}(\mathbf{x}_i), t)$$
  - $\alpha^* = \underset{\alpha}{\arg\max}\ CV(\hat{y}_\alpha)$
- Multiple hyperparameters: $\beta \in \{\beta_1, \beta_2\}, \mu \in (\mu_1, \mu_2, \mu_3\}$
  - How many times should CV be performed?

    $2 \times 3 = 6$
  - Total number of training runs?

    $6 \times K$

Cross-Validation: Test Error Estimation

- After Model selection we obtain some

  α^*, β^*

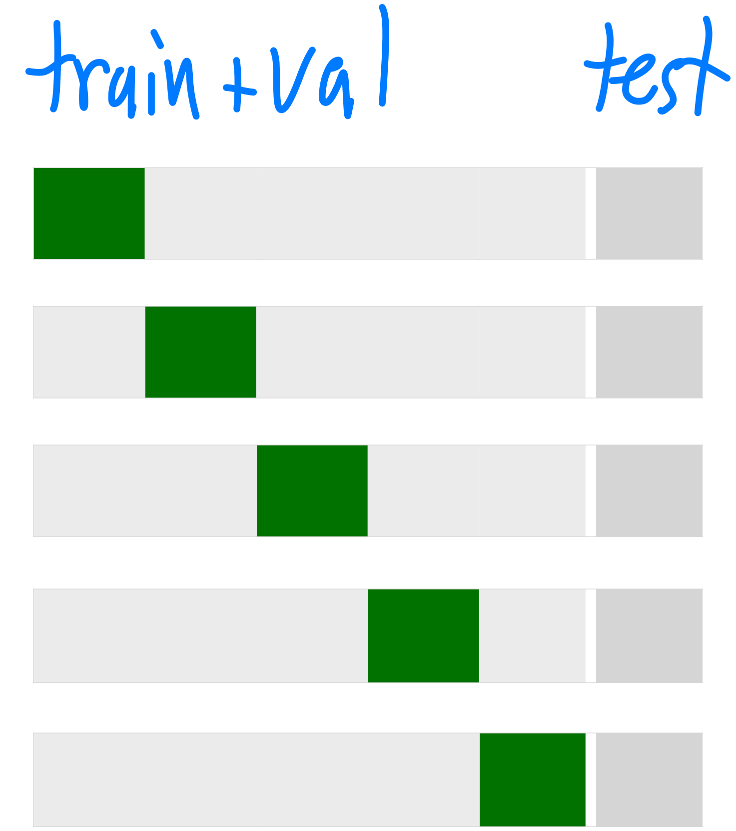
  use ensemble method

- Retrain y on all train data with α^*, β^*

- Evaluate model on held-out test set

  - Still noisy estimate of generalization error if test set is small

- Nested cross validation!



Nested Cross-Validation

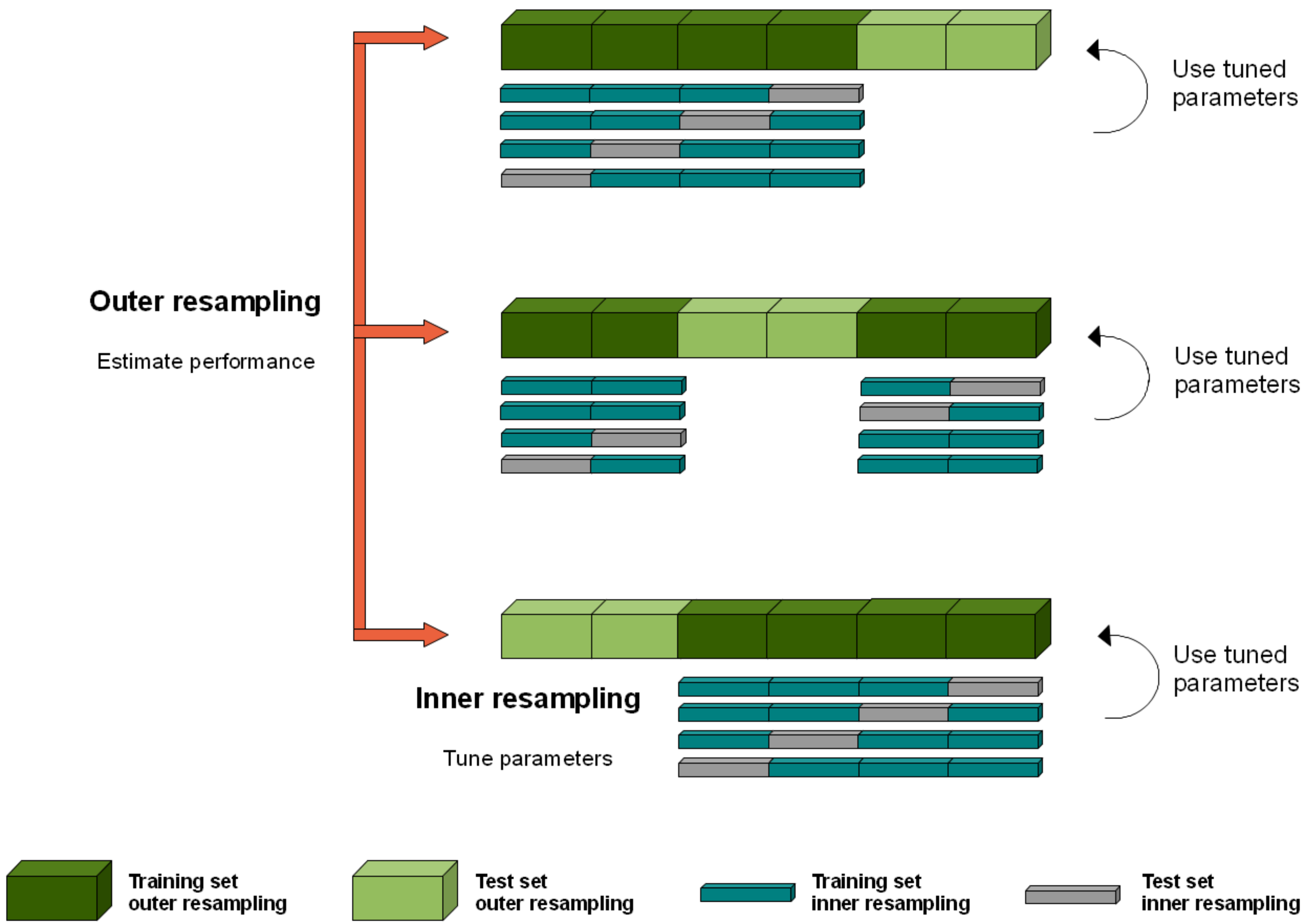


Figure: Nested cross-validation
https://mlr-org.github.io/mlr-tutorial/devel/html/nested_resampling/index.html
(site is offline unfortunately)

University of Amsterdam
Informatics Institute

AMLAB
Amsterdam
Machine Learning Lab

Machine Learning 1

Lecture 4.2 - Supervised Learning

Bias Variance Decomposition

Erik Bekkers

(Bishop 1.5.5, 3.2)

Slide credits: Patrick Forré and Rianne van den Berg

Image credit: Kirillm | Getty Images

Expected Loss for Regression $\mathbb{E}_{(\mathbf{x},t)\sim p(\mathbf{x},t)}[L(t, y(\mathbf{x})]$

Why do models make errors?
What kind of errors can we expect?

- Consider dataset of observations $(\mathbf{x}, t) \sim p(\mathbf{x}, t)$ and a model $y(\mathbf{x})$

- The model makes errors (regression loss function):

$$L(t, y(\mathbf{x})) = (t - y(\mathbf{x}))^2$$

- Every time we make an observation of random variables $(\mathbf{x}, t)$ we make a different error. We now consider the expected loss:

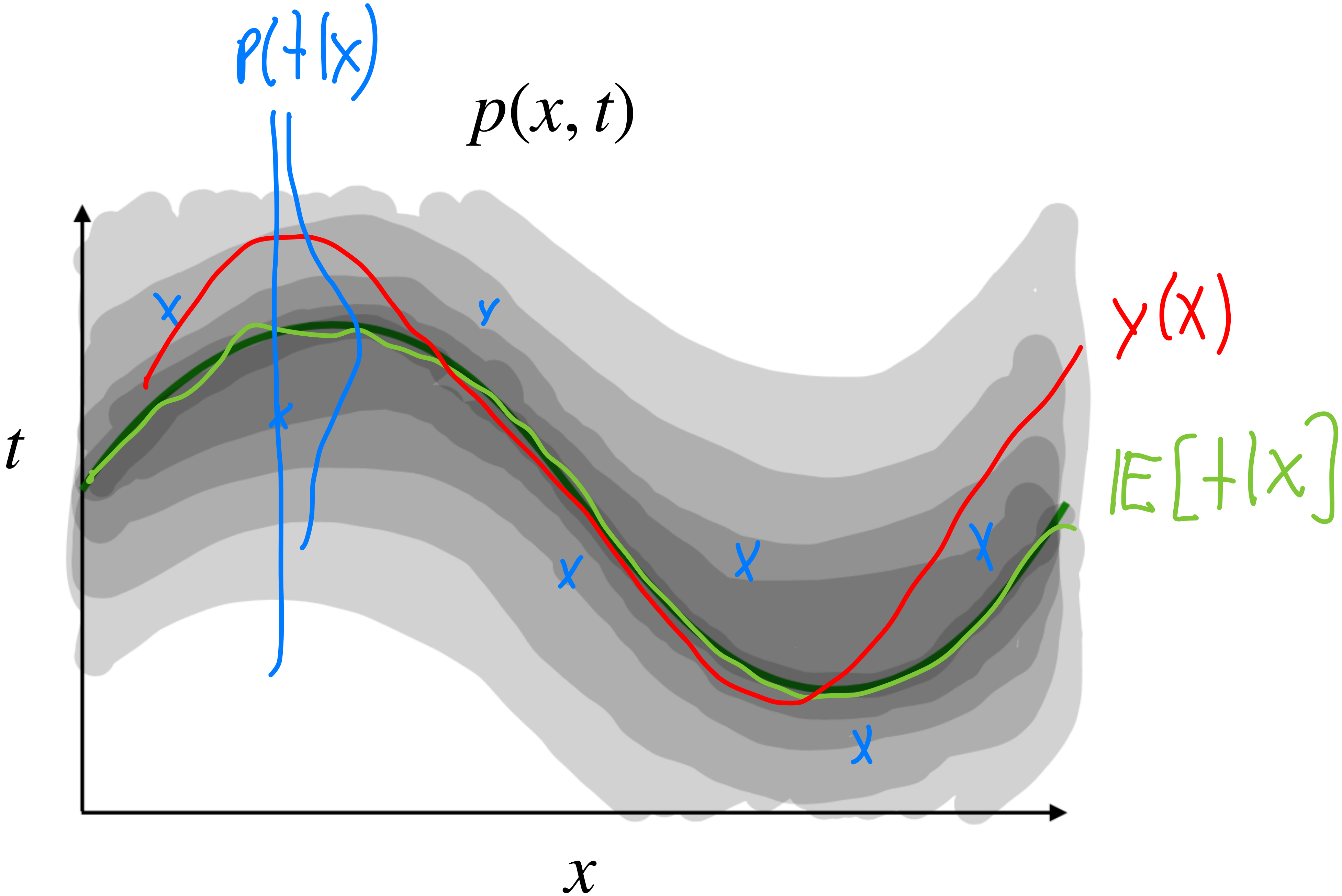
$$\mathbb{E}_{(\mathbf{x},t)\sim p(\mathbf{x},t)}[L(t, y(\mathbf{x})] = \iint (t - y(x))^2 p(x,t)\, dx\, dt$$

- The best model y we can possibly have (Bishop 1.5.5):

Regression function

$$y(\mathbf{x}) = \mathbb{E}[t|x]$$

Example: $t = \sin 2\pi x + \epsilon$, with $\epsilon \sim \mathcal{N}(0,\beta^{-1})$



Expected Loss for Regression $\mathbb{E}_{(\mathbf{x},t)\sim p(\mathbf{x},t)}[L(t, y(\mathbf{x}))]$

- Consider the expected loss:

$$\mathbb{E}[L] = \int\int (y(\mathbf{x}) - t)^2 p(\mathbf{x}, t)\mathrm{d}t\mathrm{d}\mathbf{x}$$

- Let's analyze it relative to the regression function $\mathbb{E}[t \,|\, \mathbf{x}] := \mathbb{E}_{t\sim p(t|\mathbf{x})}[t \,|\, \mathbf{x}]$:

$$\mathbb{E}[L] = \int\int (y(\mathbf{x}) \overbrace{- \mathbb{E}[t \,|\, \mathbf{x}] + \mathbb{E}[t \,|\, \mathbf{x}]}^{=0} - t)^2 p(\mathbf{x}, t)\mathrm{d}t\mathrm{d}\mathbf{x}$$

Expected Loss for Regression $\mathbb{E}_{(\mathbf{x},t)\sim p(\mathbf{x},t)}[L(t, y(\mathbf{x})]$

- Consider the expected loss:

$$\mathbb{E}[L] = \int\int (y(\mathbf{x}) - t)^2 p(\mathbf{x}, t)\mathrm{d}t\mathrm{d}\mathbf{x}$$

- Let's analyze it relative to the regression function $\mathbb{E}[t\,|\,\mathbf{x}] := \mathbb{E}_{t\sim p(t|\mathbf{x})}[t\,|\,\mathbf{x}]$:

$$\mathbb{E}[L] = \int\int (\overbrace{y(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}]}^{a} + \overbrace{\mathbb{E}[t\,|\,\mathbf{x}] - t}^{b})^2 p(\mathbf{x}, t)\mathrm{d}t\mathrm{d}\mathbf{x}$$

- Write out the square ($(a+b)^2 = a^2 + 2ab + b^2$) and use product rule ($p(\mathbf{x}, t) = p(t\,|\,\mathbf{x})p(\mathbf{x})$)

$$\mathbb{E}[L] = \int\int (y(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2 \; p(t\,|\,\mathbf{x})\mathrm{d}t \; p(\mathbf{x})\mathrm{d}\mathbf{x}$$

$$+2\int\int (y(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])(\mathbb{E}[t\,|\,x] - t) \; p(t\,|\,\mathbf{x})\mathrm{d}t \; p(\mathbf{x})\mathrm{d}\mathbf{x} = 0$$

$$\int t\, p(t|x)\, dt = \mathbb{E}[t|x]$$

$$+\int\int (\mathbb{E}[t\,|\,\mathbf{x}] - t)^2 \; p(t\,|\,\mathbf{x})\mathrm{d}t \; p(\mathbf{x})\mathrm{d}\mathbf{x}$$

Expected Loss for Regression $\mathbb{E}_{(\mathbf{x},t)\sim p(\mathbf{x},t)}[L(t, y(\mathbf{x}))]$

- Consider the expected loss:

$$\mathbb{E}[L] = \int\int (y(\mathbf{x}) - t)^2 p(\mathbf{x}, t)\mathrm{d}t\mathrm{d}\mathbf{x}$$

- Let's analyze it relative to the regression function $\mathbb{E}[t\,|\,\mathbf{x}] := \mathbb{E}_{t\sim p(t|\mathbf{x})}[t\,|\,\mathbf{x}]$:

$$\mathbb{E}[L] = \int\int (y(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}] + \mathbb{E}[t\,|\,\mathbf{x}] - t)^2 p(\mathbf{x}, t)\mathrm{d}t\mathrm{d}\mathbf{x}$$

- Write out the square ($(a+b)^2 = a^2 + 2ab + b^2$) and use product rule ($p(\mathbf{x}, t) = p(t\,|\,\mathbf{x})p(\mathbf{x})$)

$$\mathbb{E}[L] = \int\int (y(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2 \; p(t\,|\,\mathbf{x})\mathrm{d}t \; p(\mathbf{x})\mathrm{d}\mathbf{x} \qquad + \int\int (\mathbb{E}[t\,|\,\mathbf{x}] - t)^2 \; p(t\,|\,\mathbf{x})\mathrm{d}t \; p(\mathbf{x})\mathrm{d}\mathbf{x}$$

- Thus the expected loss can be decomposed in two terms

$$\mathbb{E}[L] = \int (y(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2 \; p(\mathbf{x})\mathrm{d}\mathbf{x} + \int \mathrm{var}[t\,|\,\mathbf{x}] \; p(\mathbf{x})\mathrm{d}\mathbf{x}$$

Summary so far…

- Any model y will make errors, this expected loss decomposes into

$$\mathbb{E}_{(\mathbf{x},t)\sim p(\mathbf{x},t)}[L(t,y(\mathbf{x})] = \underbrace{\int (y(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2\, p(\mathbf{x})\mathrm{d}\mathbf{x}}_{\text{sub-optimal model}} + \underbrace{\int \mathrm{var}[t\,|\,\mathbf{x}]\, p(\mathbf{x})\mathrm{d}\mathbf{x}}_{\text{intrinsic noise}}$$

- The best possible model is the regression function $y(\mathbf{x}) = \mathbb{E}[t\,|\,\mathbf{x}]$

$D_1 = \{(x_1,t_1), \ldots (x_n,t_n)\} \rightarrow y_{D_1}$

$D_2 = \{(x_1,t_1), \ldots \} \rightarrow y_{D_2}$

- In practice we approximate it with fits $y_D = \underset{y}{\mathrm{argmin}} \sum_{(\mathbf{x},t)\in D} L(t, y(\mathbf{x}))$

- What can we say about the expected loss of y_D?

The Average Expected Loss

- Let's analyze performance of a learning algorithm by averaging the expected loss over learned y_D for different datasets D

$$\mathbb{E}_D[\mathbb{E}[L]] = \int \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2]\ p(\mathbf{x})\mathrm{d}\mathbf{x} + \int \mathrm{var}[t\,|\,\mathbf{x}]\ p(\mathbf{x})\mathrm{d}\mathbf{x}$$

The Average Expected Loss

- Let's analyze performance of a learning algorithm by averaging the expected loss over learned y_D for different datasets D

$$\mathbb{E}_D[\mathbb{E}[L]] = \int \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2]\; p(\mathbf{x})\mathrm{d}\mathbf{x} + \int \mathrm{var}[t\,|\,\mathbf{x}]\; p(\mathbf{x})\mathrm{d}\mathbf{x}$$

- Analyze it relative to the average model $\mathbb{E}_D[y_D(\mathbf{x})]$

$$\mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2] = \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})] + \mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t\,|\,\mathbf{x}])^2]$$

The Average Expected Loss

- Let's analyze performance of a learning algorithm by averaging the expected loss over learned y_D for different datasets D

$$\mathbb{E}_D[\mathbb{E}[L]] = \int \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}[t \mid \mathbf{x}])^2]\, p(\mathbf{x}) \mathrm{d}\mathbf{x} + \int \mathrm{var}[t \mid \mathbf{x}]\, p(\mathbf{x}) \mathrm{d}\mathbf{x}$$

- Analyze it relative to the average model $\mathbb{E}_D[y_D(\mathbf{x})]$

$$\mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}[t \mid \mathbf{x}])^2] = \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})] + \mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t \mid \mathbf{x}])^2]$$

(Expand square $(a+b)^2 = a^2 + 2ab + b^2$)

$$= \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})])^2] + \mathbb{E}_D[(\mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t \mid \mathbf{x}])^2]$$

$$+2\, \mathbb{E}_D[(y_D(\mathbf{x}) - \cancel{\mathbb{E}_D[y_D(\mathbf{x})]})(\mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t \mid \mathbf{x}])]$$

0

The Average Expected Loss

- Let's analyze performance of a learning algorithm by averaging the expected loss over learned y_D for different datasets D

$$\mathbb{E}_D[\mathbb{E}[L]] = \int \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2]\; p(\mathbf{x})\mathrm{d}\mathbf{x} + \int \mathrm{var}[t\,|\,\mathbf{x}]\; p(\mathbf{x})\mathrm{d}\mathbf{x}$$

- Analyze it relative to the average model $\mathbb{E}_D[y_D(\mathbf{x})]$

$$\mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}[t\,|\,\mathbf{x}])^2] = \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})] + \mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t\,|\,\mathbf{x}])^2]$$

(Expand square $(a-b)^2 = a^2 - 2ab + b^2$)

$$= \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})])^2] + \mathbb{E}_D[(\mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t\,|\,\mathbf{x}])^2]$$

$$-2\;\mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})])(\mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t\,|\,\mathbf{x}])]$$

(Compute expectation, rewrite)

$$= \overbrace{\mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})])^2]}^{\text{variance}} + \overbrace{(\mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t\,|\,\mathbf{x}])^2}^{\text{bias}}$$

Bias-Variance Decomposition

- What can we say about the expected loss of y_D?
- On average (over datasets D) our model y_D will make three types of errors:

$$\mathbb{E}_D[\mathbb{E}[L]] = \int \mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})])^2]\, p(\mathbf{x})d\mathbf{x} \qquad \text{Variance}$$

$$+ \int (\mathbb{E}_D[y_D(\mathbf{x})] - \mathbb{E}[t\,|\,\mathbf{x}])^2\, p(\mathbf{x})d\mathbf{x} \qquad \text{Bias}^2$$

$$+ \int \mathrm{var}[t\,|\,\mathbf{x}]\, p(\mathbf{x})d\mathbf{x} \qquad \text{Noise}$$

Bias-Variance Decomposition: Example

- Generate $L = 100$ datasets of $N = 50$ points:
  - $x \sim U(0,1)$
  - $t = \sin(2\pi x) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \alpha^{-1})$
  - $\mathbb{E}[t \mid x] = \sin(2\pi x)$
- Parametrize y with 24 Gaussian basis functions
  The $L = 100$ datasets each gives a model
  - $y^{(l)}(x) = (\mathbf{w}^{(l)})^T \boldsymbol{\phi}(x)$
- That minimizes:
  - $$E_D = \frac{1}{2} \sum_{i=1}^{N} \{t_n - \mathbf{w}^T \boldsymbol{\phi}(x)\}^2 + \frac{\lambda}{2} \mathbf{w}^T \mathbf{w}$$
- Let $\bar{y}(x) = \mathbb{E}_D[y_D(x)]$ the average function

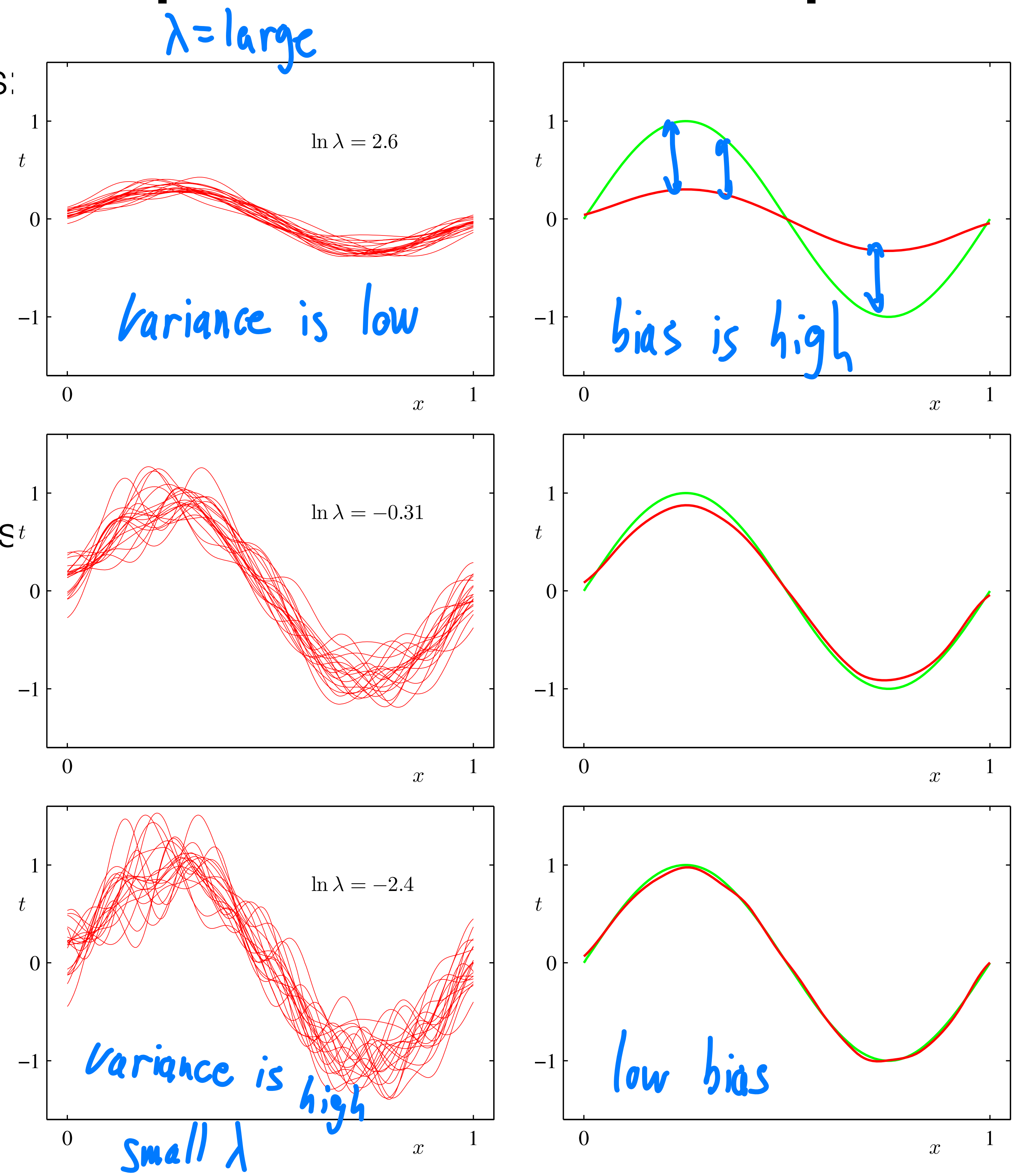


Figure: bias-variance decomposition (Bishop 3.5)

Bias-Variance Decomposition

- What can we say about the expected loss of y_D?
- On average (over datasets D) our model y_D will make three types of errors:

$$\mathbb{E}_D[\mathbb{E}[L]] = \int \mathbb{E}_D[(y_D(x) - \mathbb{E}_D[y_D(x)])^2]\, p(x)\mathrm{d}x \qquad \text{Variance}$$

$$+ \int (\mathbb{E}_D[y_D(x)] - \mathbb{E}[t \mid x])^2\, p(x)\mathrm{d}x \qquad \text{Bias}^2$$

$$+ \int \mathrm{var}[t \mid x]\, p(x)\mathrm{d}x \qquad \text{Noise}$$

Bias-Variance Decomposition

- What can we say about the expected loss of y_D?

- On average (over datasets D) our model y_D will make three types of errors:

$$\mathbb{E}_D[\mathbb{E}[L]] \approx \quad \frac{1}{N}\sum_{n=1}^{N}\frac{1}{L}\sum_{l=1}^{L}(y^{(l)}(x_n) - \bar{y}(\mathbf{x}_n))^2] \qquad \text{Variance}$$

$$+\frac{1}{N}\sum_{i=1}^{N}\bar{y}(x_n) - \sin 2\pi x_n)^2 \qquad \text{Bias}^{\ 2}$$

Bias-Variance Decomposition: Example

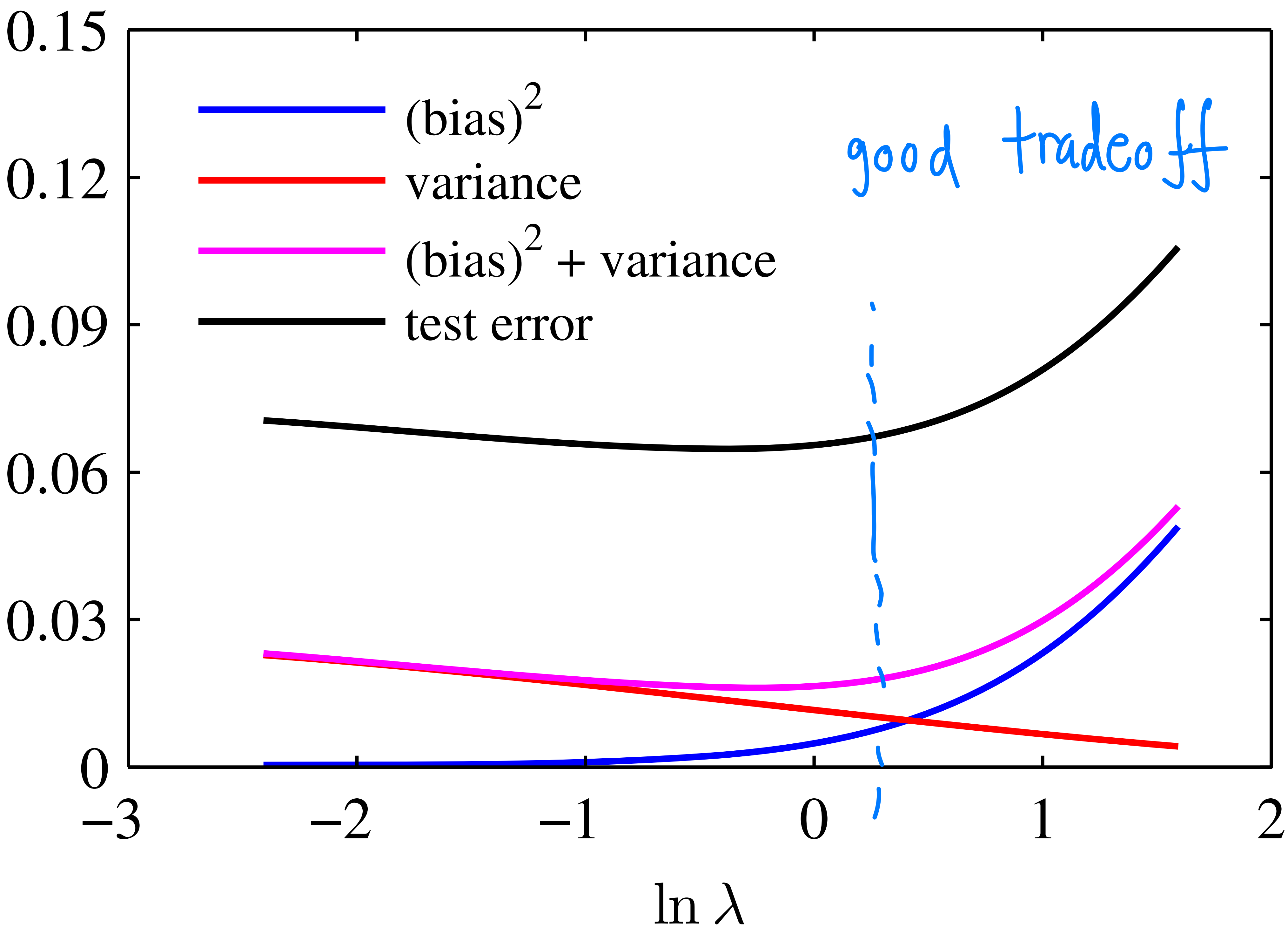


Figure: bias-variance decomposition (Bishop 3.6)

Bias-Variance Decomposition

- In practice we don't want to split our dataset into L datasets to determine the best model complexity (best value of λ)
- Better to keep large dataset
  - Less overfitting.
  - Different optimal model complexity!
- Model averaging? Bayesian regression!

University of Amsterdam
Informatics Institute

Machine Learning 1

Lecture 4.3 - Supervised Learning
Bayesian Linear Regression - Gaussian Posteriors

Erik Bekkers

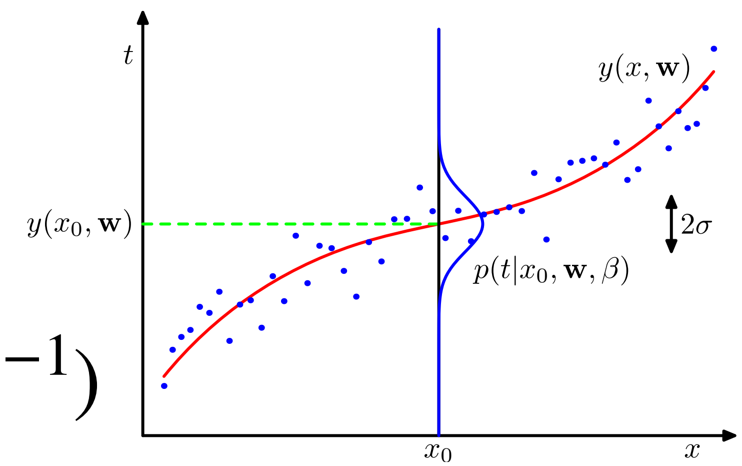
(Bishop 3.3.1 (and 2.3.3)

Slide credits: Patrick Forré and Rianne van den Berg

Image credit: Kirillm | Getty Images

Bayesian Linear Regression

- Regression problem with:



  - Data: $\mathbf{X} = (\mathbf{x}_1, \ldots \mathbf{x}_N)^T$, $\mathbf{t} = (t_1, \ldots, t_N)^T$
  - Predictive distribution $p(t' \,|\, \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t' \,|\, \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}'), \beta^{-1})$

- Probabilistic model with Gaussians:
  - Likelihood: $$p(\mathbf{t} \,|\, \mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n \,|\, \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) = \mathcal{N}(\underline{t} \,|\, \underline{\Phi} w, \beta^{-1} \mathbb{1})$$
  - Conjugate prior: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w} \,|\, \mathbf{m}_0, \mathbf{S}_0)$
  - Posterior: $$p(\mathbf{w} \,|\, \mathbf{t}, \mathbf{X}) = \frac{p(\mathbf{t} \,|\, \mathbf{X}, \mathbf{w}, \beta) p(\mathbf{w})}{p(\mathbf{t} \,|\, \mathbf{X}, \beta)} = \mathcal{N}(\underline{w} \,|\, \underline{m_N}, S_N)$$

- Maximum A Posteriori estimate:
  - $\mathbf{w}_{\text{MAP}} = m_N$

Bishop Ch 2.3, Eq. 2.116

$$\mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi}$$
$$\mathbf{m}_N = \mathbf{S}_N (\mathbf{S}_0^{-1} \mathbf{m}_0 + \beta \boldsymbol{\Phi}^T \mathbf{t})$$

Bayesian Linear Regression

- Special simple prior:
  - $p(\mathbf{w} \mid \alpha) = \mathcal{N}(\mathbf{w} \mid \mathbf{0}, \alpha^{-1}\mathbf{I})$ $\quad$ ($\mathbf{m}_0 = \mathbf{0}$ and $\mathbf{S}_0 = \alpha^{-1}\mathbf{I}$)
- Posterior
  - $p(\mathbf{w} \mid \mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}_N, \mathbf{S}_N)$
  - With $\mathbf{m}_N = \mathbf{S}_N(\cancel{\mathbf{S}_0^{-1}\mathbf{m}_0}^{0} + \beta\mathbf{\Phi}^T\mathbf{t}) = \beta S_N \Phi^T t$

    $\mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \beta\mathbf{\Phi}^T\mathbf{\Phi} = (\alpha I + \beta \Phi^T \Phi)$

Bayesian Linear Regression

- Limiting cases of the posterior
  - $p(\mathbf{w}\,|\,\mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(\mathbf{w}\,|\,\mathbf{m}_N, \mathbf{S}_N)$ with $\mathbf{m}_N = \beta\,\mathbf{S}_N\,\mathbf{\Phi}^T\,\mathbf{t}$

    $\mathbf{S}_N^{-1} = \alpha\,\mathbf{I} + \beta\mathbf{\Phi}^T\mathbf{\Phi}$

- Infinitely broad prior ($p(\mathbf{w}\,|\,\alpha) = \mathcal{N}(\mathbf{w}\,|\,\mathbf{0}, \alpha^{-1}\mathbf{I})$, $\alpha \to 0$):
  (No restriction/assumption on $\mathbf{w}$)
  - $\lim_{\alpha\to 0} \mathbf{m}_N = \lim_{\alpha\to 0} \beta\,(\cancel{\alpha}\mathbf{I} + \beta\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t} = (\underline{\Phi}^T\underline{\Phi})^{-1}\underline{\Phi}^T\underline{t} = w_{ML}$

- Infinitely narrow prior ($\alpha \to \infty$):
  - $\lim_{\alpha\to\infty} \mathbf{m}_N = \lim_{\alpha\to\infty} \beta\,(\alpha\mathbf{I} + \beta\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t} = \ldots = \underline{m_0}$
  - $\lim_{\alpha\to\infty} \mathbf{S}_N = \lim_{\alpha\to\infty} (\alpha\mathbf{I} + \beta\mathbf{\Phi}^T\mathbf{\Phi})^{-1} = 0$ (zero matrix)

UNIVERSITY OF AMSTERDAM
Informatics Institute

AMLAB Amsterdam Machine Learning Lab

Machine Learning 1

Lecture 4.4 - Supervised Learning
Bayesian Linear Regression - Sequential Bayesian Learning

Erik Bekkers

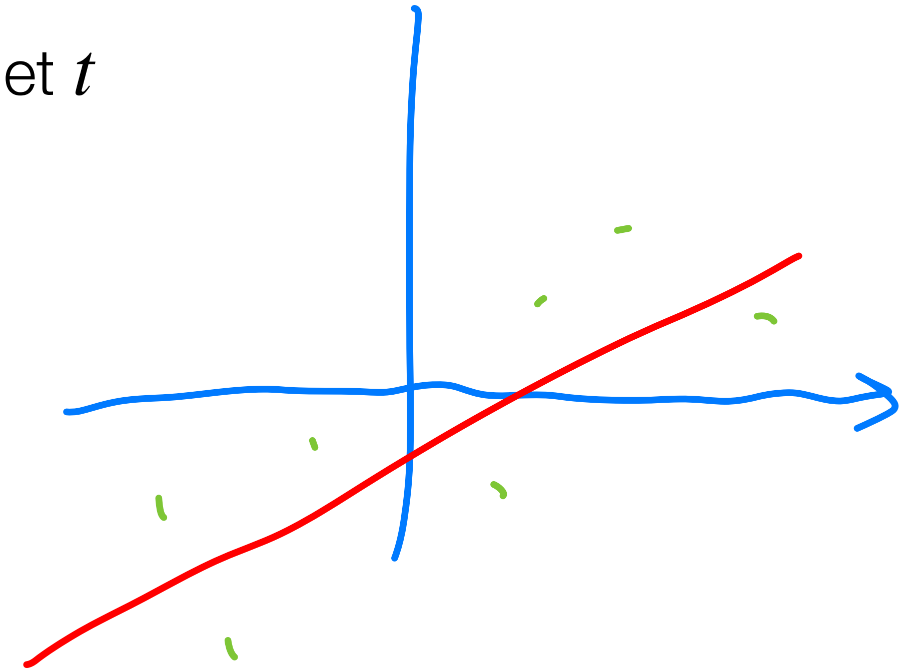
(Bishop 3.3.1)

Slide credits: Patrick Forré and Rianne van den Berg

Image credit: Kirillm | Getty Images

Example: Sequential Bayesian Learning

- Data come in as a sequences of observations of input x, target t



- **Synthetic data generated by**
  - $x \sim \mathcal{U}(x \,|\, -1,1)$
  - $t = f(x, \mathbf{a}) + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, 0.2^2)$
  - $f(x, \mathbf{a}) = a_0 + a_1 x, \quad a_0 = -0.3, \, a_1 = 0.5$
- **Modeling choices**
  - Target distribution: $p(t' \,|\, x', \mathbf{w}, \beta) = \mathcal{N}(t' \,|\, y(x', \mathbf{w}), \beta^{-1}),$ $\qquad \beta^{-1} = 0.2^2$
  - Linear model: $y(x, \mathbf{w}) = w_0 + w_1 x$
  - Prior: $p(\mathbf{w} \,|\, \alpha) = \mathcal{N}(\mathbf{w} \,|\, \mathbf{0}, \alpha^{-1}\mathbf{I}),$ $\qquad \alpha = 2$
- **Sequential Bayesian Learning**: Posterior after $N-1$ observations is prior for arrival of N^{th} datapoint!

Example: Sequential Bayesian Learning

- Data generated by

$$t = -0.3 + 0.5x + \epsilon$$

- Prior

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

- Sample 1 datapoint

(x_1, t_1)

- Likelihood

$$p(t_1 \mid x_1, \mathbf{w}, \beta) = \mathcal{N}(t_1 \mid w_0 + w_1 x, \beta)$$

- Posterior

$$p(\mathbf{w} \mid x_1, t_1, \alpha, \beta) \propto p(t \mid x, \underline{w}, \beta) \cdot p(\underline{w} \mid \alpha)$$

get w_0, w_1

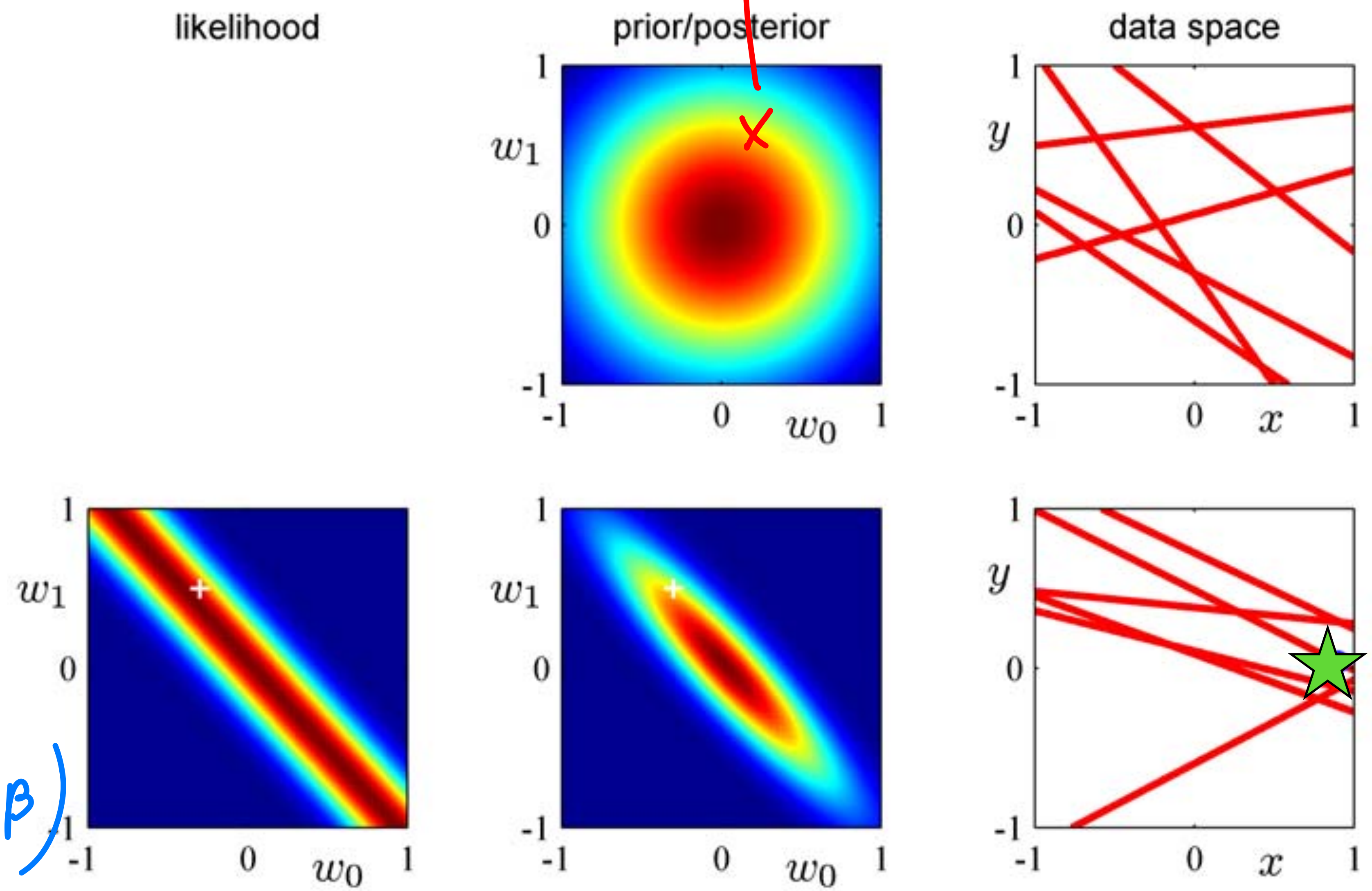


Figure: Sequential Bayesian learning (Bishop 3.7)

Example: Sequential Bayesian Learning

- Sample 2nd data point
- Posterior → Prior
- Likelihood

$p(t_2 \mid x_2, \mathbf{w}, \beta)$

$p(w \mid (x_1, t_1), (x_2, t_1), \beta) =$

$= \dfrac{p(t_1 \mid \ldots) \cdot p(t_2 \mid \ldots) \cdot p(w \mid \alpha)}{p(t_1 \mid x) \cdot p(t_2 \mid x)}$

- Posterior

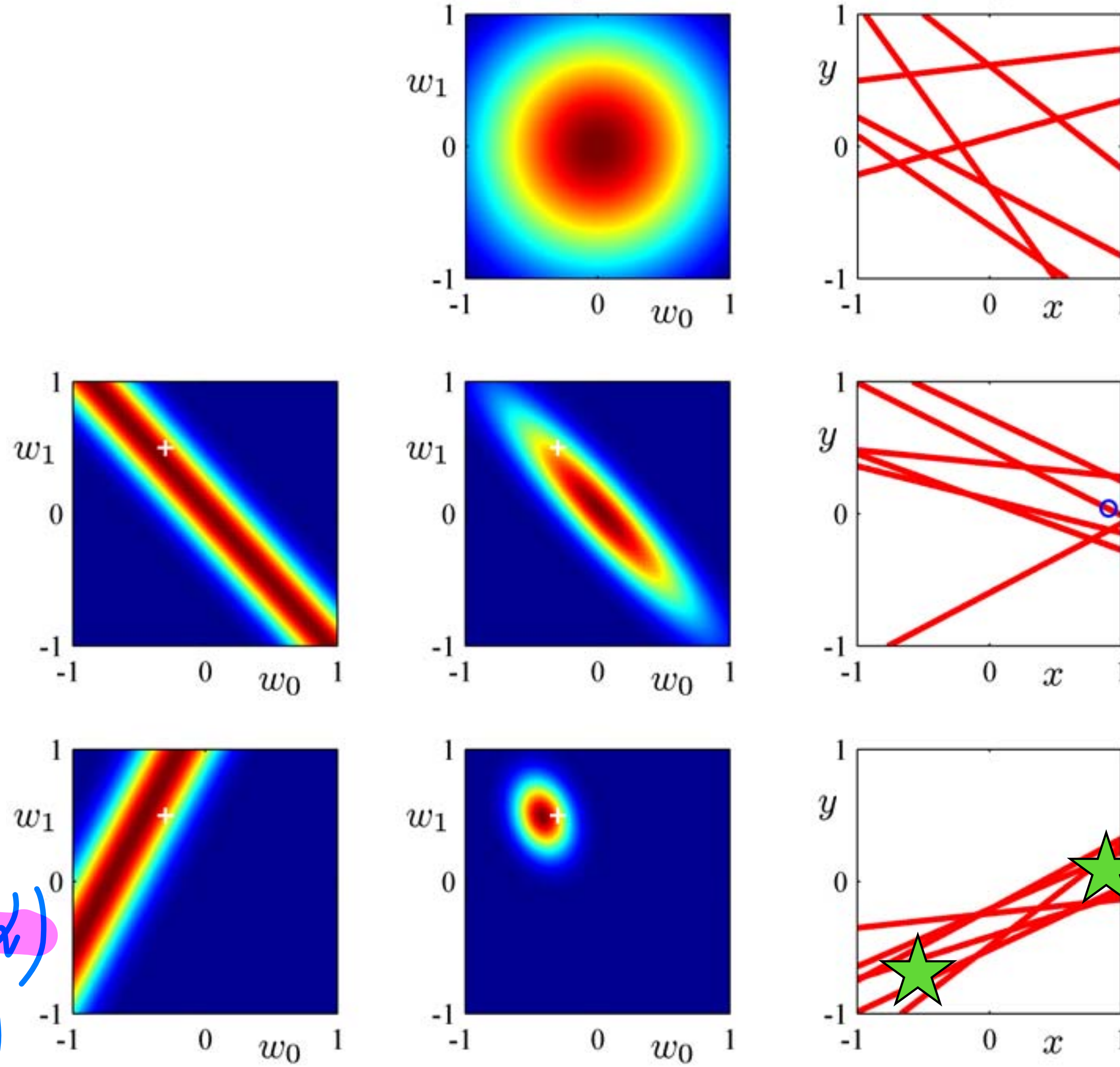


Figure: Sequential Bayesian learning (Bishop 3.7)

$$p(\mathbf{w} \mid (x_1, t_1), (x_2, t_2), \alpha, \beta) \propto p(t_2 \mid x_2, \mathbf{w}, \beta) p(\mathbf{w} \mid (x_1, t_1), \alpha, \beta)$$

Example: Sequential Bayesian Learning

- After 19 data ponts

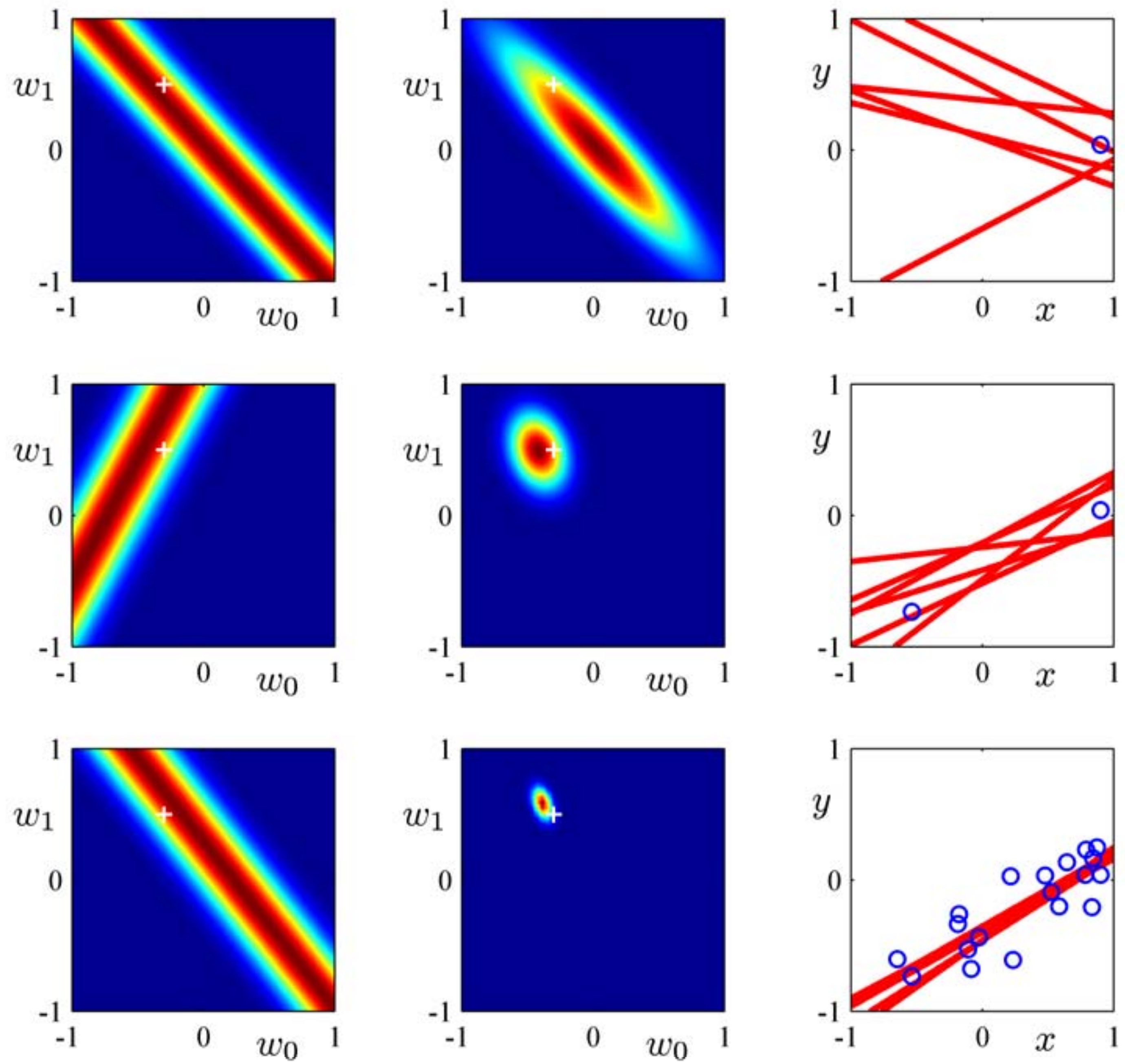
- Prior

$$p(\mathbf{w} \,|\, \{(x_n, t_n)\}_{n=1}^{19}, \alpha, \beta)$$

- Likelihood

$$p(t_{20} \,|\, x_{20}, \mathbf{w}, \beta)$$

- Posterior



$$p(\mathbf{w} \,|\, \{(x_n, t_n)\}_{n=1}^{20}, \alpha, \beta) \propto p(t_{20} \,|\, x_{20}, \mathbf{w}, \beta)p(\mathbf{w} \,|\, \{(x_n, t_n)\}_{n=1}^{19}, \alpha, \beta)$$

Figure: Sequential Bayesian learning (Bishop 3.7)

Bayesian Linear Regression

- Limiting cases of the posterior

  - $p(\mathbf{w} \,|\, \mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(\mathbf{w} \,|\, \mathbf{m}_N, \mathbf{S}_N)$ with $\mathbf{m}_N = \beta\, \mathbf{S}_N\, \mathbf{\Phi}^T\, \mathbf{t}$

    $\mathbf{S}_N^{-1} = \alpha\, \mathbf{I} + \beta \mathbf{\Phi}^T \mathbf{\Phi}$

- After infinite amount of data ($N \to \infty$):

  - $\lim\limits_{N\to\infty} [\mathbf{\Phi}^T\mathbf{\Phi}]_{ij} = \lim\limits_{N\to\infty} \sum_{n=1}^{N} \phi_i(\mathbf{x}_n)\phi_j(\mathbf{x}_n)$

  - $\lim\limits_{N\to\infty} \mathbf{S}_N =$

  - $\lim\limits_{N\to\infty} \mathbf{m}_N = \lim\limits_{N\to\infty} \beta\,(\alpha\mathbf{I} + \beta\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$

University of Amsterdam
Informatics Institute

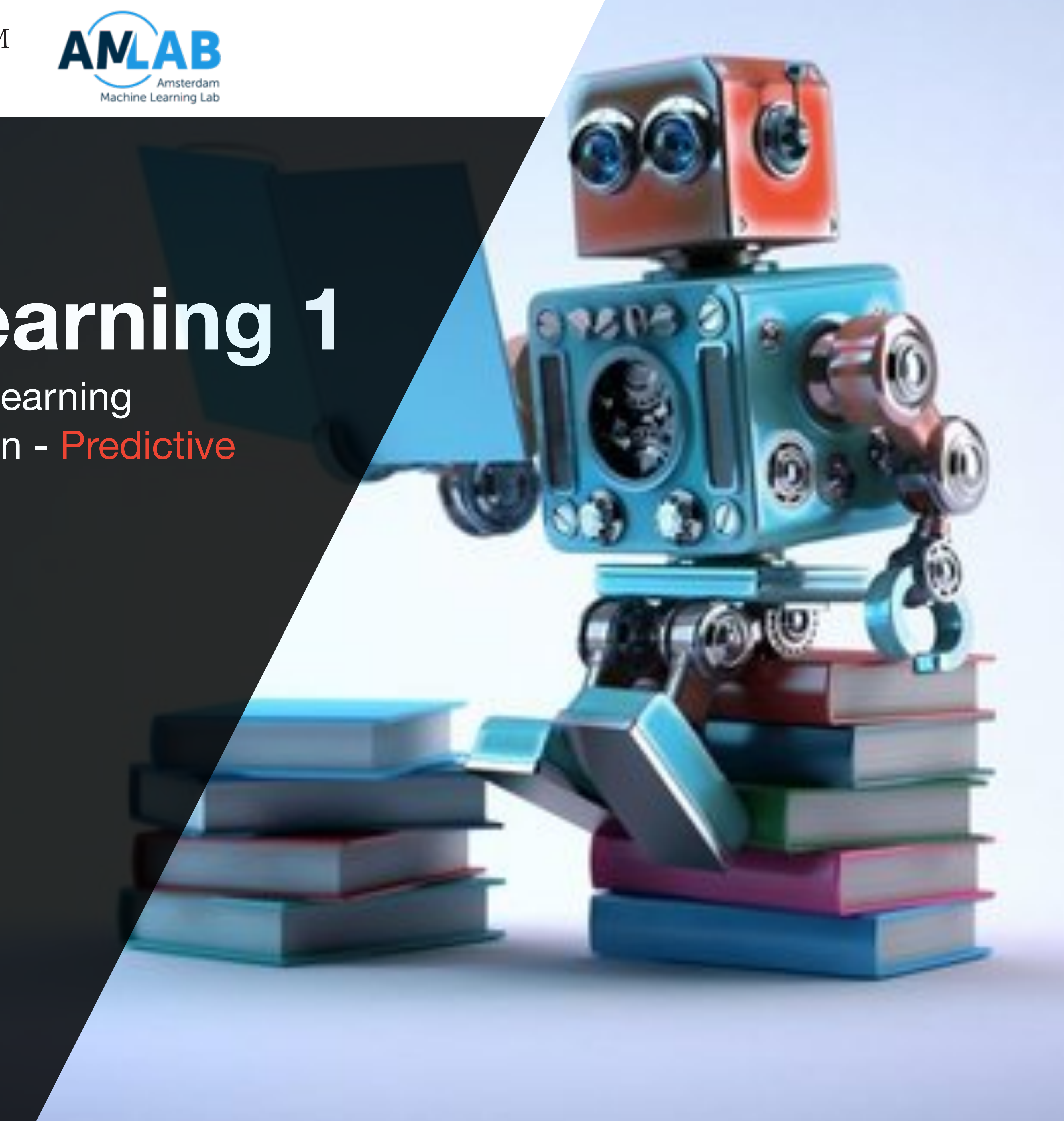
AMLAB Amsterdam Machine Learning Lab

Machine Learning 1

Lecture 4.5 - Supervised Learning
Bayesian Linear Regression - Predictive Distribution

Erik Bekkers

(Bishop 3.3.2)



Slide credits: Patrick Forré and Rianne van den Berg

Image credit: Kirillm | Getty Images

Predictive Distribution

- Observed dataset with inputs $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$ and targets $\mathbf{t} = (t_1, \ldots, t_N)^T$
- Gaussian Posterior distribution (from Gaussian prior and Gaussian likelihood)
  - $p(\mathbf{w} \mid \mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}_N, \mathbf{S}_N)$ with $\mathbf{m}_N = \beta\, \mathbf{S}_N\, \mathbf{\Phi}^T\, \mathbf{t}$

$$\mathbf{S}_N^{-1} = \alpha\, \mathbf{I} + \beta \mathbf{\Phi}^T \mathbf{\Phi}$$

- Parametrized Gaussian predictive distribution:
  - $p(t' \mid \mathbf{x}', \mathbf{w}, \beta) = \mathcal{N}(t \mid \phi(\mathbf{x}')^T \mathbf{w}, \beta^{-1})$
- Gaussian Bayesian predictive distribution for new input
  - $p(t' \mid x', \mathbf{X}, \mathbf{t}, \alpha, \beta) = \int \mathcal{N}(t' \mid \boldsymbol{\phi}(\mathbf{x}')^T \mathbf{w}, \beta^{-1}) \mathcal{N}(\mathbf{w} \mid \mathbf{m}_N, \mathbf{S}_N) \mathrm{d}\mathbf{w}$

Bishop Eq. 2.115

$$= \mathcal{N}(t' \mid \mathbf{x}', \boldsymbol{\phi}(\mathbf{x}')^T \mathbf{m}_N, \sigma_N^2(\mathbf{x}'))$$

- With $\sigma_N^2(\mathbf{x}') = \beta^{-1} + \boldsymbol{\phi}(\mathbf{x}')^T \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}')$

Predictive Distribution

- Datasets:
  - $t = \sin(2\pi x) + \epsilon$
  - $\epsilon \sim \mathcal{N}(0, \beta^{-1})$
- Dataset sizes:
  - $N = 1,2,4,25$
- Model:
  - $y(x, \mathbf{w}) = \boldsymbol{\phi}(x)^T \mathbf{w}$
  - $\phi_j(x)$: Gaussian basis functions

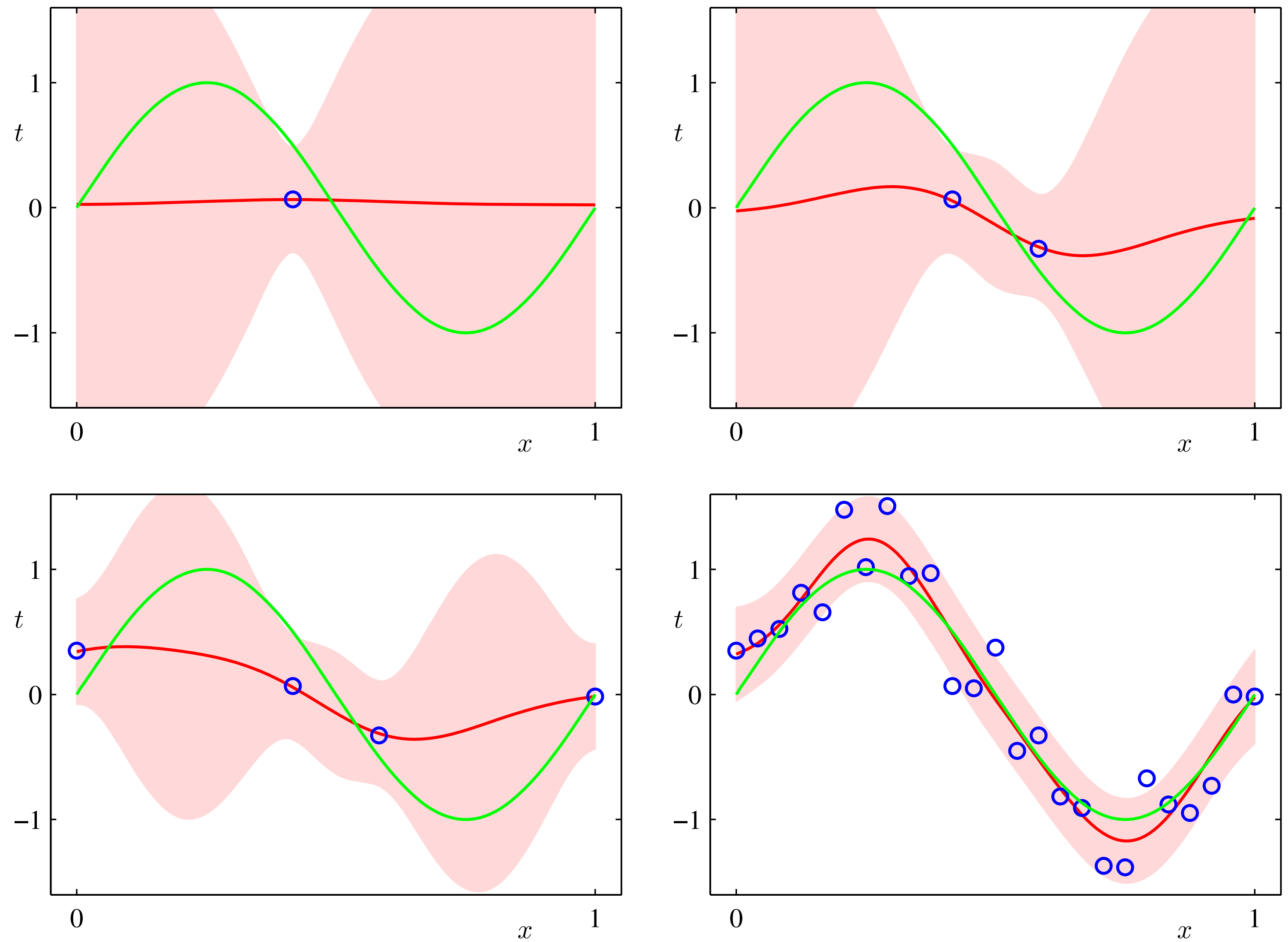


Figure: Predictive distribution (Bishop 3.8)

- Predictive distribution:
  - $p(t' \mid x', \mathbf{X}, \mathbf{t}, \alpha, \beta) = \; = \mathcal{N}(t' \mid \mathbf{x}', \boldsymbol{\phi}(\mathbf{x}')^T \mathbf{m}_N, \sigma_N^2(\mathbf{x}'))$
  - $\sigma_N^2(\mathbf{x}') = \beta^{-1} + \boldsymbol{\phi}(\mathbf{x}')^T \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}'), \quad \mathbf{m}_N = \beta\, \mathbf{S}_N\, \boldsymbol{\Phi}^T\, \mathbf{t}, \quad \mathbf{S}_N^{-1} = \alpha\, \mathbf{I} + \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi}$

Samples drawn from Bayesian Predictive Distribution

$$p(\mathbf{w} \mid \mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}_N, \mathbf{S}_N),$$

with $\mathbf{m}_N = \beta \, \mathbf{S}_N \, \boldsymbol{\Phi}^T \, \mathbf{t}$

$$\mathbf{S}_N^{-1} = \alpha \, \mathbf{I} + \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi}$$

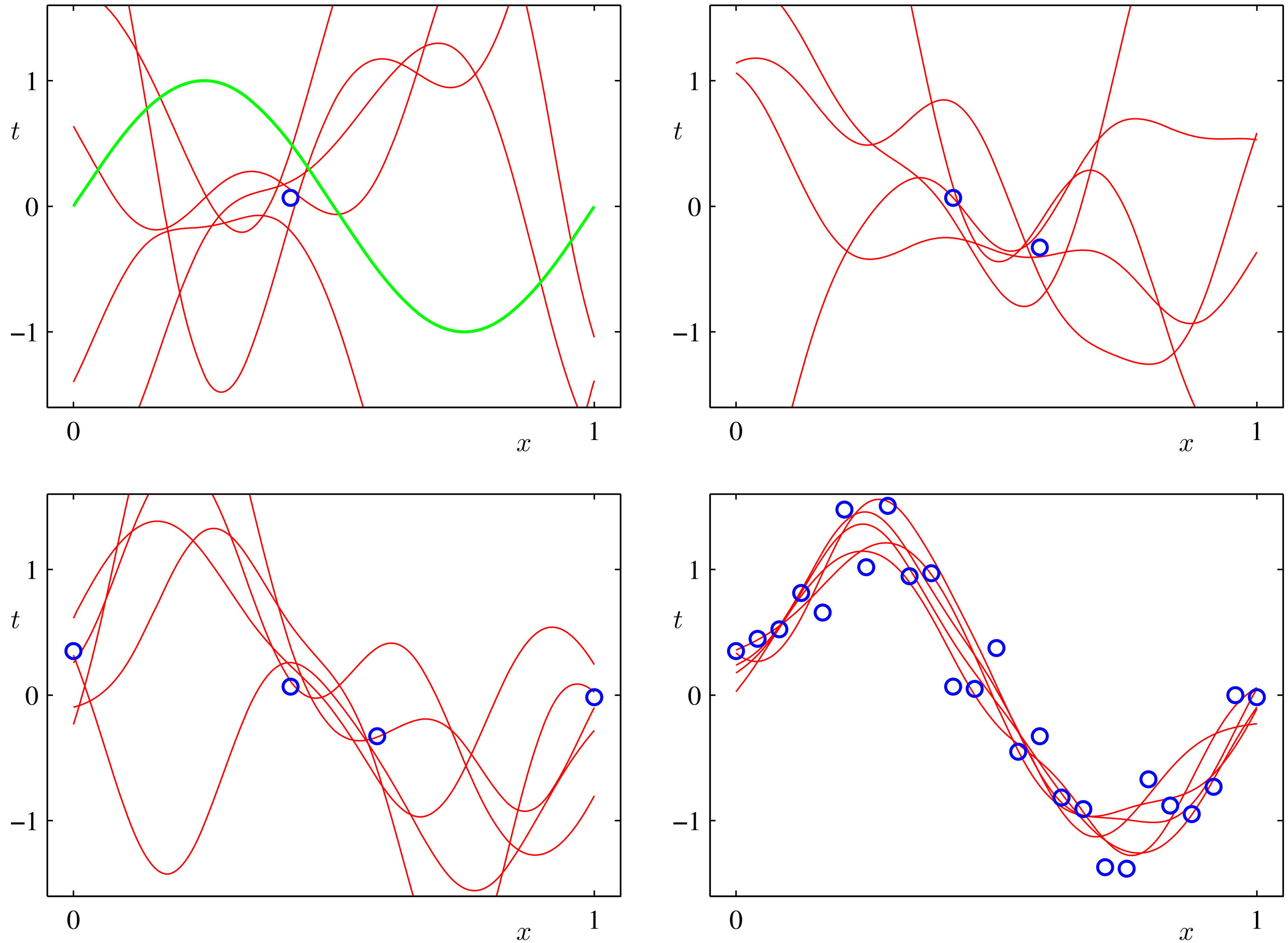


Figure: *Sample functions $y(x, \mathbf{w})$ with $\mathbf{w}$ sampled from posterior distribution (Bishop 3.9)*