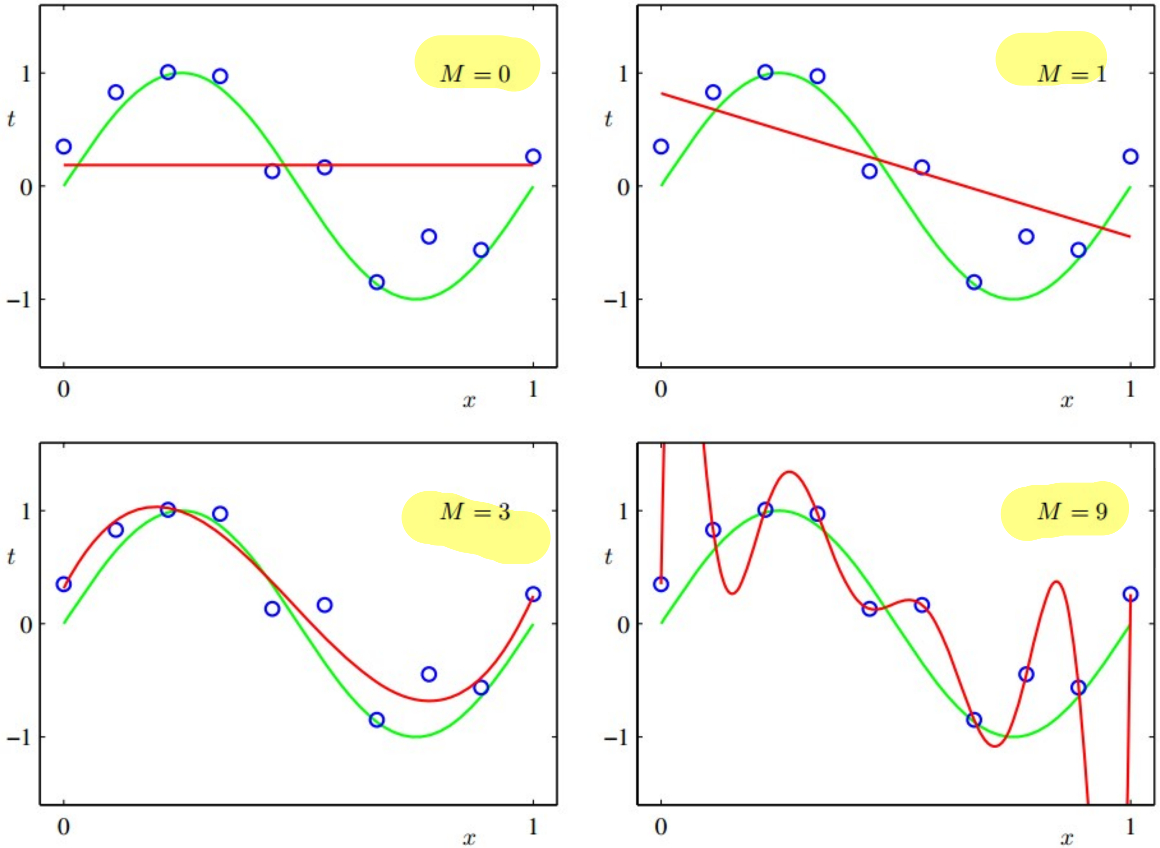
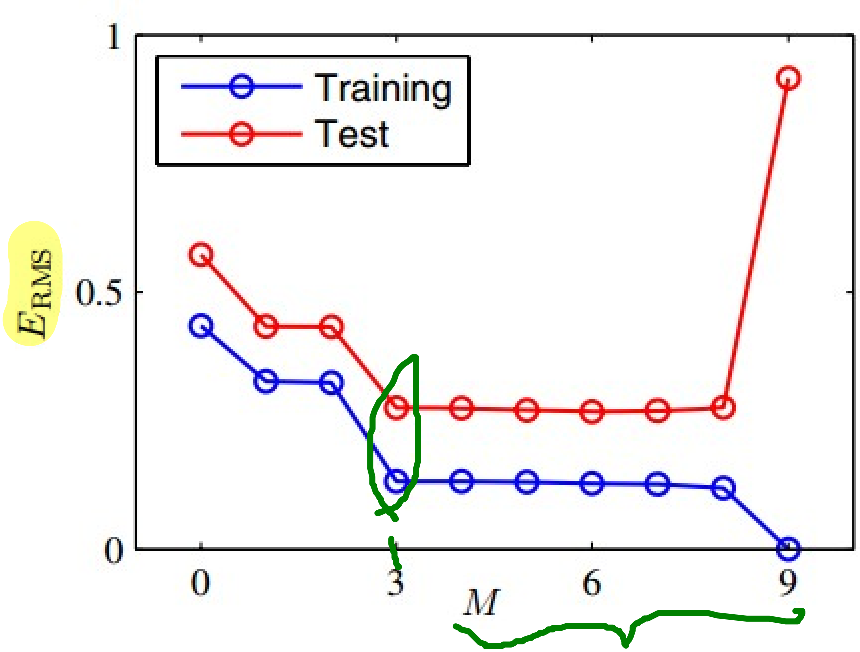
Lecture 4 – Part I
Regularization

- Motivation, definition
- Observation: Large weights and overfitting
- Regularization: closed form in linear regression + intuitions
- Does it work ? A few examples
- The Bayesian interpretation
- Regul during GD: Parameter shrinkage, weight decay
- Lasso

Complexity controlled **explicitly** (rare case)

M = polynomial order





Bishop, 2006

Regularization (general definition)

- A possible def: *“Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.”* From <u>Deep Learning</u>, by Ian Goodfellow and Yoshua Bengio and Aaron Courville

  https://www.deeplearningbook.org/

- Goal: Regularization allows to ***restrain*** **a model’s complexity, *quantitatively*,** without *explicitly* limiting the model (i.e. order of polynomial fitting, etc)

  Not explicitly modify H, but do restrain the actual visited part of H (H=hypothesis space)

- Examples:

  - Lasso, Ridge, Elastic-Net
  - Dropout (see DeepNetworks)
  - feature selection procedures
  - ensemble methods

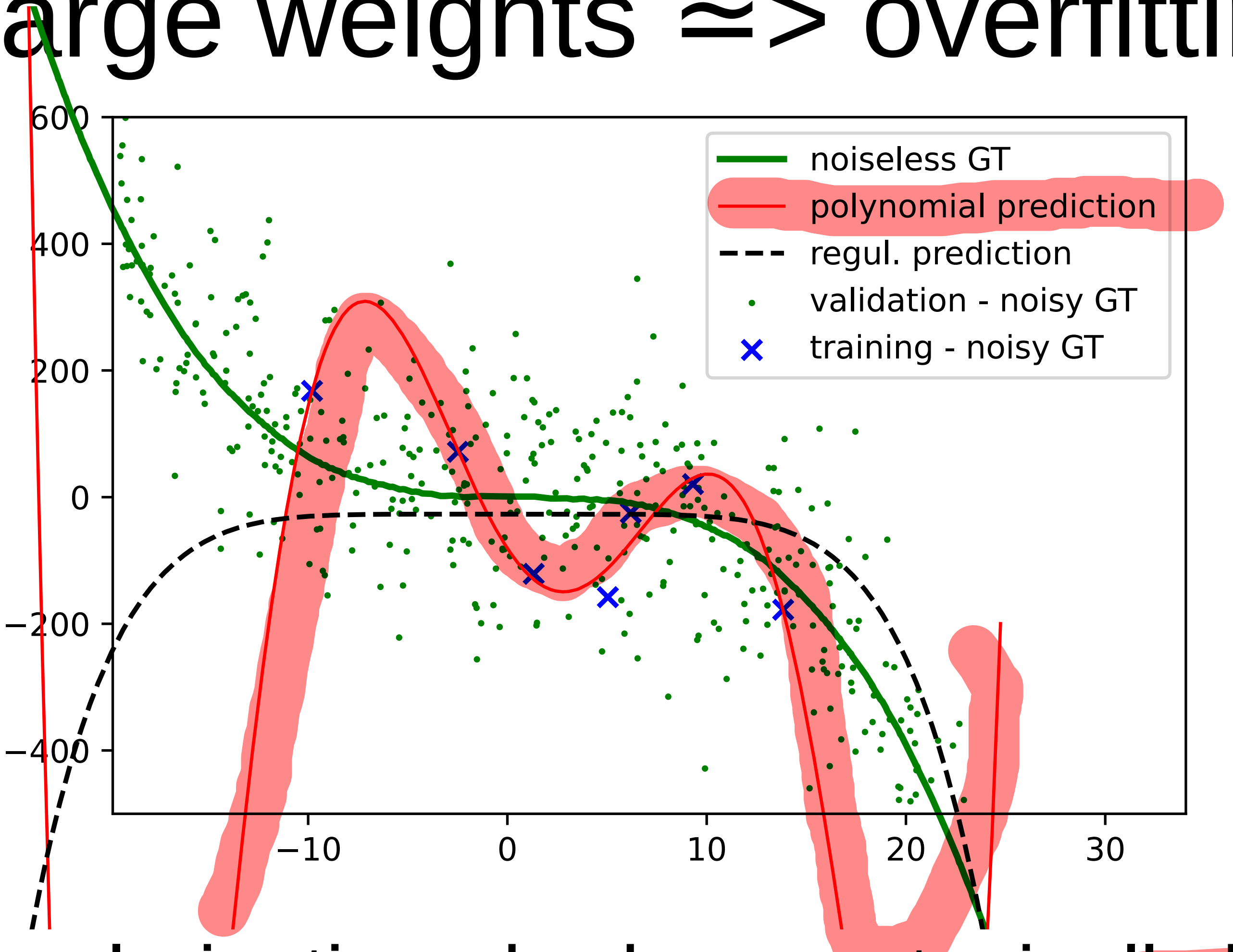
  PCA as pre-processing

  early stopping

- Here we focus on classics, i.e. **Ridge** and **Lasso**

Empirical Observation

Large weights ≃> overfitting



- No regularization : bad score, typically high weights (esp. coeffs of large order are too high)
- cf `lecture4-unregularized regression has large coefficients.ipynb` .

Intuition: Large weights≃>overfitting

(it’s actually more complicated)

$x \to x + \delta x$, $y \to y + W.\delta x$

$O(1)$ and $W \gg 1$

- **Large weights** : output $W.x$ is **very sensitive** to small changes in data x. So, small perturbation of training data → big changes in weights → big changes in output (→ overfitting)

- **Small weights** : output $W.x$ is less sensitive, i.e. is **more robust** w.r.t. change in data : not so different output for slightly different data → less overfitting (=better generaliztion)

$$y \xrightarrow[x \to x + \delta x]{} y + W.\delta x \simeq y \pm o(1)$$

- Remark: actually, the value of weights itself is meaningless. But, that’s the spirit.

$x \to a_0 + x a_1 + x^2 a_2 + \dots + x^p a_p \quad x \in \mathbb{R}$

Adding a Regularization term

There are two standard regularization terms. For a ML problem with a given Loss L :

- **Ridge** regul.: $+ \lambda \|\vec{w}\|_2^2$ (L2-Regul)

$$\mathcal{L}_{Ridge} = \mathcal{L} + \lambda \cdot \underbrace{\|\vec{w}\|^2}_{\hookrightarrow \sum_{d=1}^{D} |w_d|^2}$$

- **Lasso** regul.: (L1 regul)

$$\mathcal{L}_{Lasso} = \mathcal{L} + \lambda \|\vec{w}\|_1$$

$$\|\vec{w}\| = \sum_{d=1}^{D} |w_d|$$

- Elastic-Net: a mix of them both:

$$\mathcal{L}_{Elastic} = \mathcal{L} + \alpha_1 |\vec{w}|_1 + \alpha_2 \|w\|_2^2$$

Linear Regression: **One-shot solution**

$$\frac{\partial}{\partial w_1}\mathcal{L} = \frac{1}{N}\sum x_{n1}(\ldots)$$

- (without regularization)

$$\mathcal{L} = \frac{1}{2}\frac{1}{N}\sum_{n=1}^{N}\left(\vec{w}\cdot\vec{x}_n - y_n\right)^2 = \frac{1}{2}\frac{1}{N}\sum_{n=1}^{N}\left(w_1 x_{n1} + w_2 x_{n2} + \ldots + w_D x_{nD} - y_n\right)^2$$

$$\vec{\nabla}_{\vec{w}}\mathcal{L} = \vec{0} \iff \frac{1}{N}\frac{\cancel{2}}{\cancel{2}}\underbrace{\sum_{n=1}^{N}\underbrace{\vec{x}_n}_{\mathbb{R}^D}\big(\overbrace{\vec{w}\cdot\vec{x}_n - y_n}^{\mathbb{R}}\big)} = \vec{0}$$

$$\sum_{n=1}^{N}\left(\underbrace{\vec{x}_n\cdot\vec{w}} - y_n\right)\vec{x}_n = \vec{0}$$

$$\big(\underset{N,D}{X}\cdot\underset{D}{w} - \underset{N}{Y}\big)_{N}\cdot\underset{N,D}{X} = \underset{1,D}{(\ldots)} = \underset{D}{(\ldots)}$$

$$\Updownarrow$$

$$\big(\underset{N,D}{X}\cdot\underset{D}{w} - \underset{N}{Y}\big)\cdot\underset{N,D}{X} = \underset{D}{\vec{0}}$$

$$\underbrace{(X^T)}_{D,N}\underbrace{(X \cdot w - Y)}_{N,} = \vec{0}_D$$

$X : N, D$

$Y : N$

$w : D$

$$X^T \cdot X w - X^T \cdot Y = \vec{0}$$

$$\underbrace{(X^T X)}_{D,D} w = X^T Y$$

$$\underbrace{(X^T X)}_{dd'} = \sum_{n=1}^{N} x_{n,d} \, x_{n,d'}$$

$\propto$ covariance matrix of the data (of the features of the data)

$X^T X$ is almost surely semi-definite positive ($\lambda \geq 0$)

$\Rightarrow$ is invertible

$$w = \underbrace{(X^T X)^{-1}}_{D,D} \cdot \underbrace{X^T}_{D,N} \cdot \underbrace{Y}_{N}$$

$$\mathcal{L} = \frac{1}{N} \sum (\vec{w} \, \vec{x}_n - y_n)^2 + \lambda \|\vec{w}\|_2^2$$

$$\|w\|_2^2 = \sum_{d=1}^{D} w_d^2$$

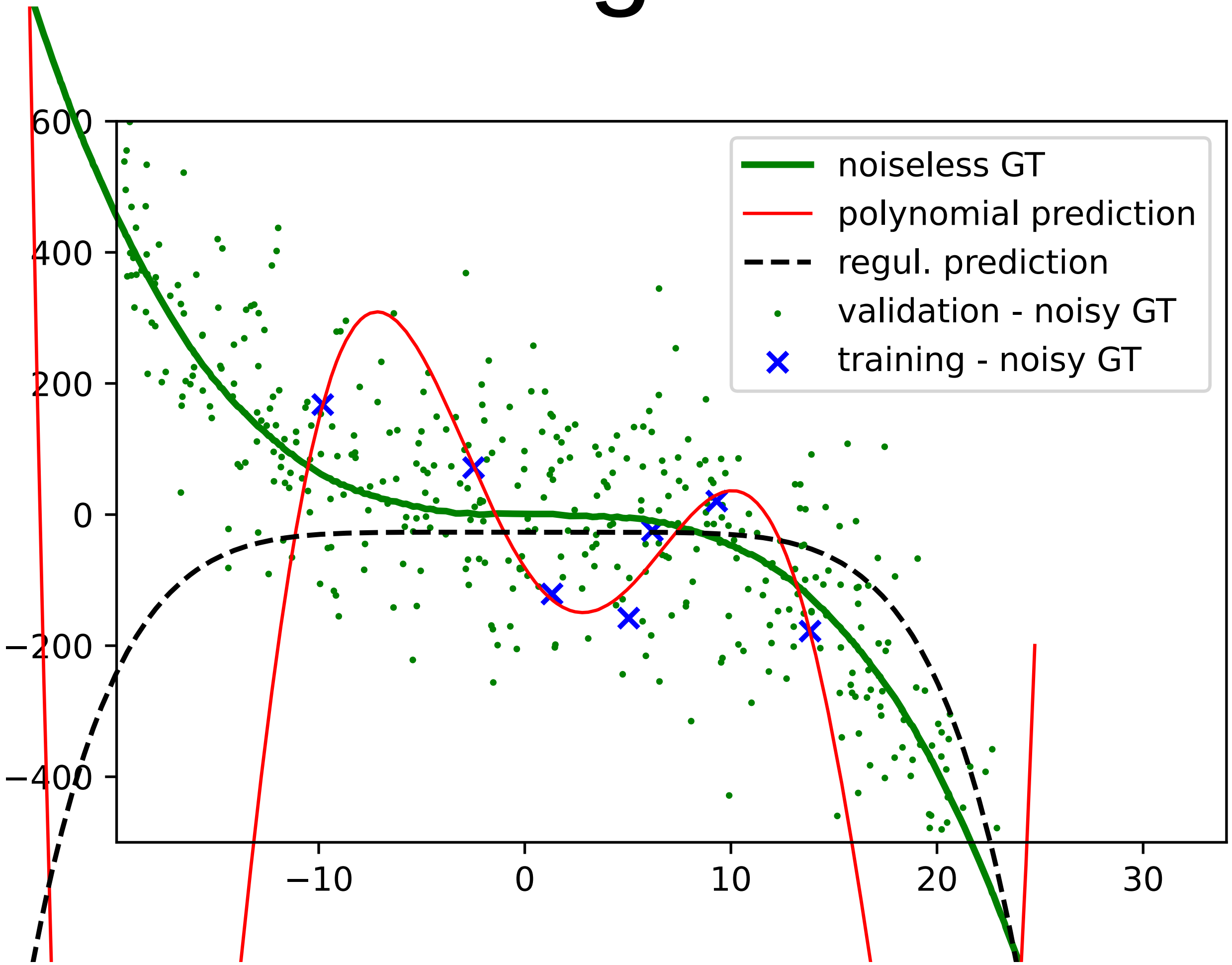
Linear Regression: **One-shot solution**

- (**with** regularization)

Linear Regression:
One-shot solution

- (**with** regularization, in D=1 – even more intuitive)

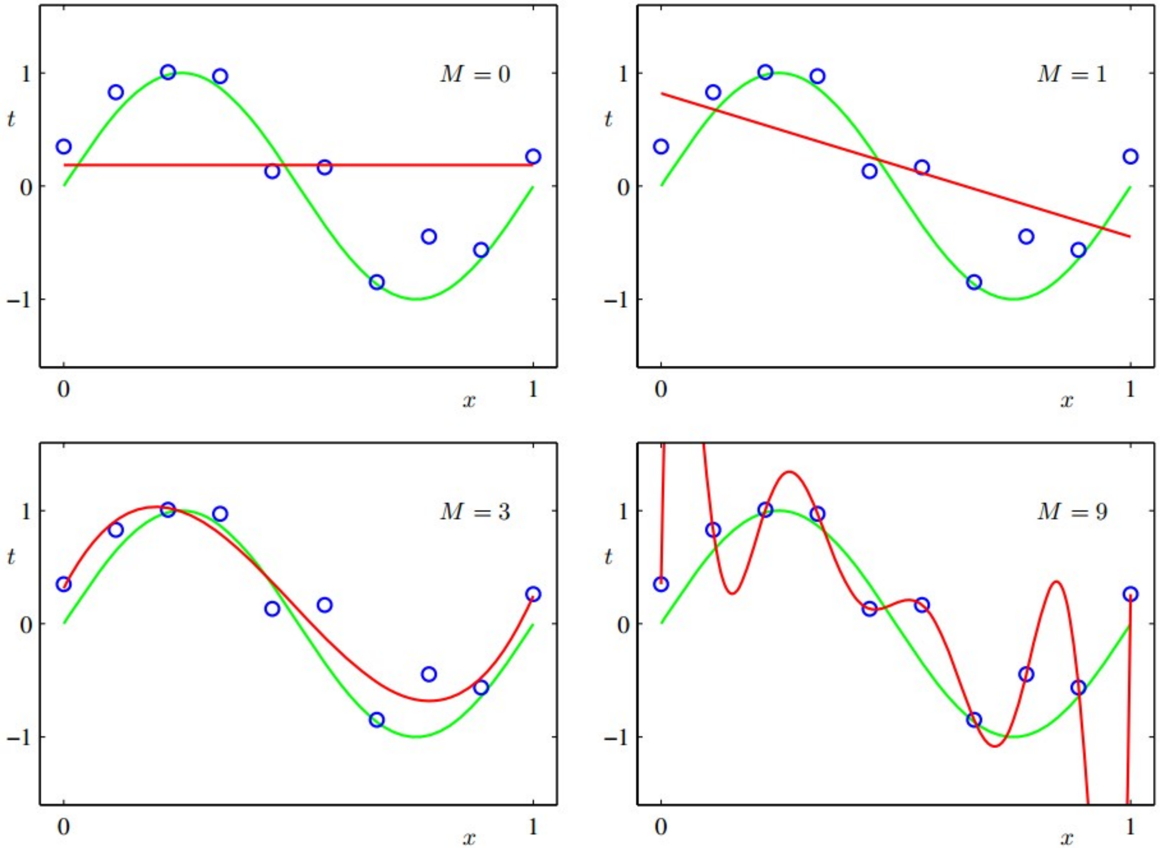
Does regul. work ?

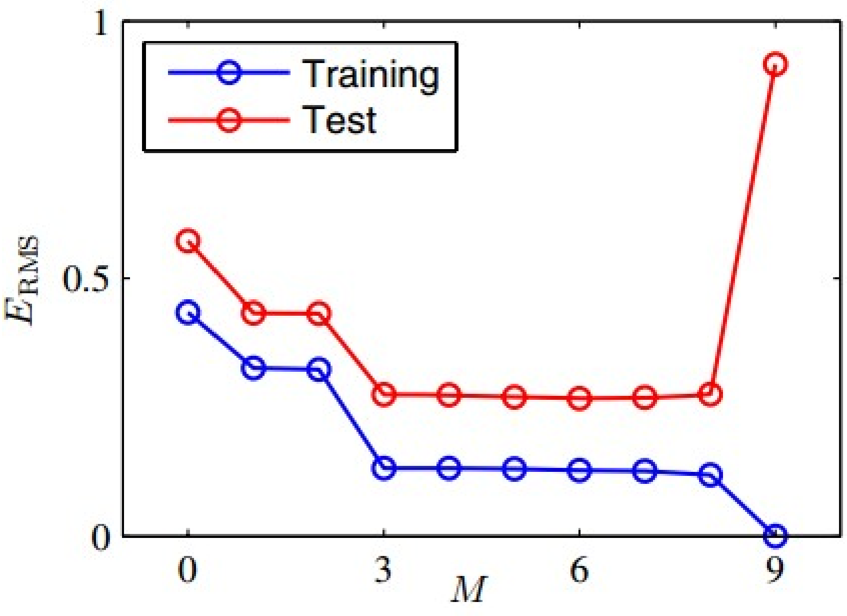


- No regularization : bad score, typically high weights (esp. coeffs of large order are too high)
- **With regularization: better score, all coeffs. shrink a lot (towards 0)**

Complexity controlled **explicitly** (rare case)

M = polynomial order



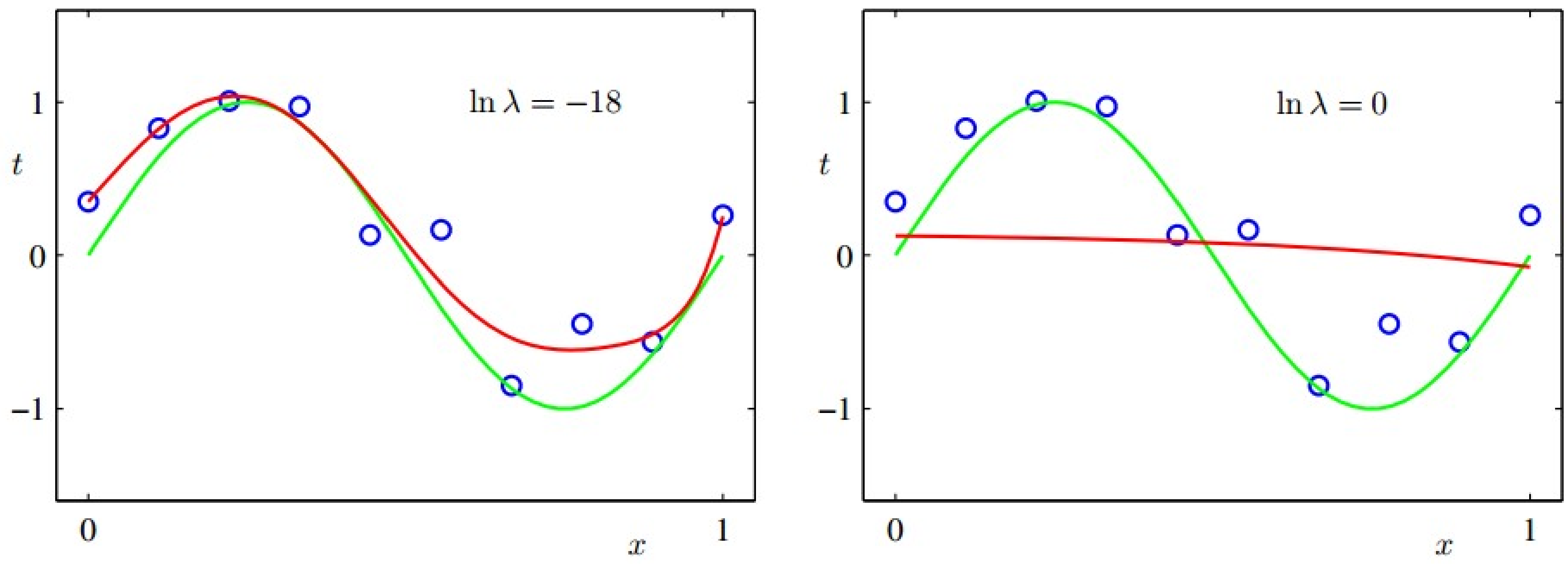


Bishop, 2006

Complexity controlled **indirectly : regularization**

M = polynomial order: still $M=9$

But, with **regularization parameter λ** changed



Important: practice with the **tutorial** to do more tests / play with:

`lecture4-regularization-dependence on lambda.ipynb`

Bayesian computations
The basics (prerequisite)

- MLE + an *a priori* opinion on what things should be = Maximum A Posteriori = MAP
- i.e. estimate a random variable, with the opinion (a priori) that its mean is of order τ :

Regularization
Bayesian **interpretation**

- MLE + an *a priori* opinion on what the model is = MAP → we can get the L2 regul from that !
- Assume model's weights follow a Gaussian distribution

Regularization during GD: Parameter shrinkage, weight decay

- What does regularization do **during a GD ?**

Lasso Regularization

- If we use the L1 norm: (or L0 norm)
- Effect: tends to set some weight to 0 exactly
  → it's already feature selection !

References

- **Algebra** reminder: *Bishop,* appendix C, p. 695-701 (only 6 pages !!)
- **Regularization**: *Bishop*, sec. 3.1.4, p. 144-146
  See also Sec. 5.5, p. 256-271, for much much more (Neural Nets).
- Another good **book**: (more recent, 2016): *Deep Learning*, by Ian Goodfellow and Yoshua Bengio and Aaron Courville https://www.deeplearningbook.org/ , in particular the chapter 5, https://www.deeplearningbook.org/contents/ml.html
-