l_2 Regularization

1. For l_2 regularization we have,

$$\tilde{\mathcal{L}}(\mathbf{w}) = \mathcal{L}(\mathbf{w}) + \frac{1}{2}\alpha\|\mathbf{w}\|^2$$

2. For SGD (or its variants), we are interested in

$$\nabla\tilde{\mathcal{L}}(\mathbf{w}) = \nabla\mathcal{L}(\mathbf{w}) + \alpha\mathbf{w}$$

3. Update rule:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \nabla\tilde{\mathcal{L}}(\mathbf{w}^t) = \mathbf{w}^t - \nabla\mathcal{L}(\mathbf{w}^t) - \alpha\mathbf{w}^t$$

4. Let us see the geometric interpretation of this

1. Assume $\mathbf{w}^* = \arg\min_{\mathbf{w}} \; \mathcal{L}(\mathbf{w})$
2. thus, $\nabla\mathcal{L}(\mathbf{w}^*) = \mathbf{0}$
3. Consider $\mathbf{u} = \mathbf{w} - \mathbf{w}^*$
4. Using Taylor series approximation of $\mathcal{L}(\mathbf{w})$ around $\mathbf{w}^*$ (upto 2 order)

$$\begin{aligned}
\mathcal{L}(\mathbf{w}) = \mathcal{L}(\mathbf{u}+\mathbf{w}^*) &= \mathcal{L}(\mathbf{w}^*) + \mathbf{u}^T\nabla\mathcal{L}(\mathbf{w}^*) + \frac{1}{2}\mathbf{u}^T H\mathbf{u} \\
&= \mathcal{L}(\mathbf{w}^*) + (\mathbf{w}-\mathbf{w}^*)^T\nabla\mathcal{L}(\mathbf{w}^*) + \frac{1}{2}(\mathbf{w}-\mathbf{w}^*)^T H(\mathbf{w}-\mathbf{w}^*) \\
&= \mathcal{L}(\mathbf{w}^*) + \frac{1}{2}(\mathbf{w}-\mathbf{w}^*)^T H(\mathbf{w}-\mathbf{w}^*), \qquad (\because \mathcal{L}(\mathbf{w}^*) = \mathbf{0})
\end{aligned}$$

where H is hessian of $\mathcal{L}(\mathbf{w})$ evaluated at $\mathbf{w}^*$.

1. $\nabla \mathcal{L}(\mathbf{w}) = H(\mathbf{w} - \mathbf{w}^*)$
2. $\nabla \tilde{\mathcal{L}}(\mathbf{w}) = \nabla \mathcal{L}(\mathbf{w}) + \alpha \mathbf{w} = H(\mathbf{w} - \mathbf{w}^*) + \alpha \mathbf{w}$
3. If H is symmetric and positive semidefinite, we have $H = Q \Lambda Q^T$, where $QQ^T = Q^T Q = I$ and Λ is a diagonal matrix having eigenvalues of H.

- Minimizer of $\tilde{\mathcal{L}}$ is as follows.

$$
\begin{aligned}
\tilde{\mathbf{w}} &= (H + \alpha I)^{-1} H \mathbf{w}^* \\
&= (Q \Lambda Q^T + \alpha I)^{-1} Q \Lambda Q^T \mathbf{w}^* \\
&= (Q \Lambda Q^T + \alpha Q I Q^T)^{-1} Q \Lambda Q^T \mathbf{w}^* \\
&= \left(Q(\Lambda + \alpha I) Q^T \right)^{-1} Q \Lambda Q^T \mathbf{w}^* \\
&= (Q^T)^{-1} (\Lambda + \alpha I)^{-1} Q^{-1} Q \Lambda Q^T \mathbf{w}^* \\
&= Q (\Lambda + \alpha I)^{-1} \Lambda Q^T \mathbf{w}^* \\
&= Q D Q^T \mathbf{w}^*
\end{aligned}
$$

where $D = (\Lambda + \alpha I)^{-1} \Lambda$.

- We see that

$$D = \begin{bmatrix} \frac{\lambda_1}{\lambda_1+\alpha} & 0 & 0 & \cdots & 0 \\ 0 & \frac{\lambda_2}{\lambda_2+\alpha} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{\lambda_d}{\lambda_d+\alpha} \end{bmatrix}$$

- $\mathbf{w}^*$ first gets rotated by Q^T to give $Q^T\mathbf{w}^*$
- However if $\alpha = 0$ then Q rotates $Q^T\mathbf{w}^*$ back to $\mathbf{w}^*$
- If $\alpha \neq 0$, then each element i of $Q^T\mathbf{w}^*$ gets scaled by λ_i before it is rotated back by Q.
- If $\lambda_i >> \alpha$, then $\frac{\lambda_i}{\lambda_i+\alpha} = 1$. If $\lambda_i << \alpha$, then $\frac{\lambda_i}{\lambda_i+\alpha} = 0$.
- Thus only significant directions (larger eigen values) will be retained.
- Effective parameters $= \sum_{i=1}^{n} \frac{\lambda_i}{\lambda_i+\alpha} < n$

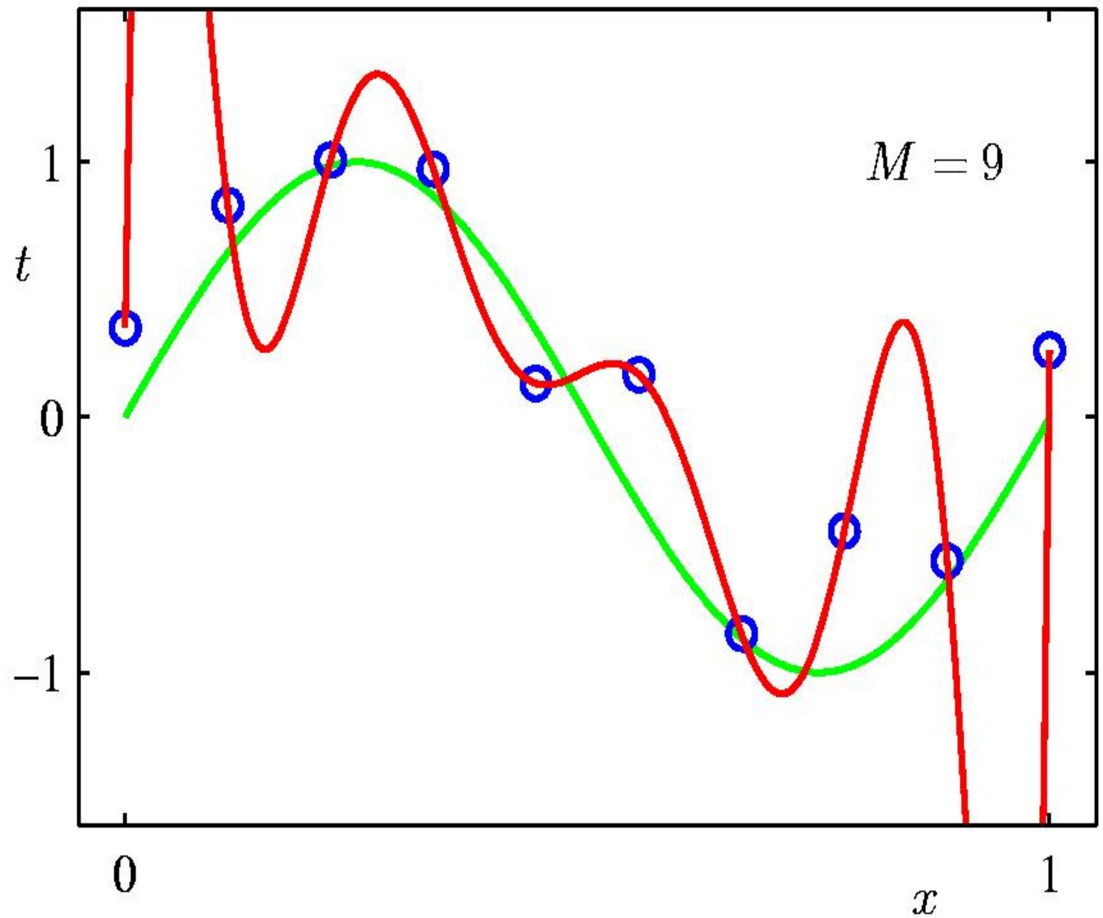
Regularization

- Use complex models, but penalize large coefficient values:

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

Regularization on 9th Order Polynomial

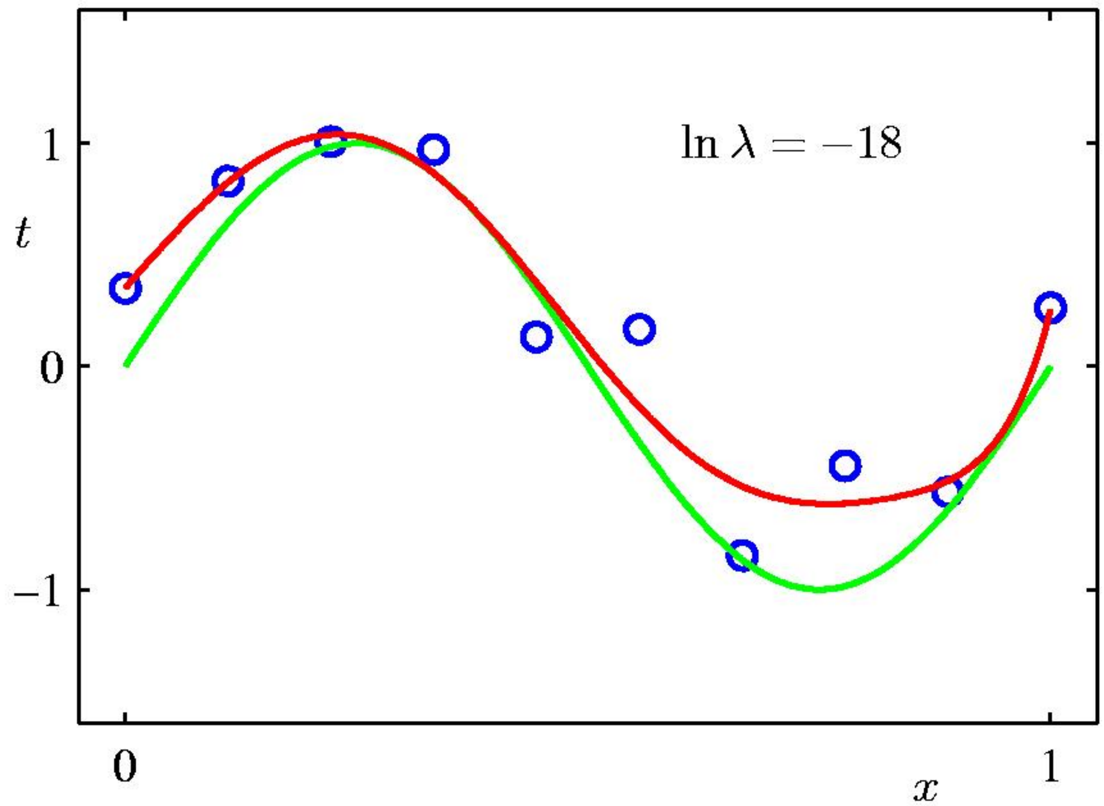
$\ln \lambda = -\text{inf}$



Too small λ – no regularization effect

Regularization on 9th degree polynomial:

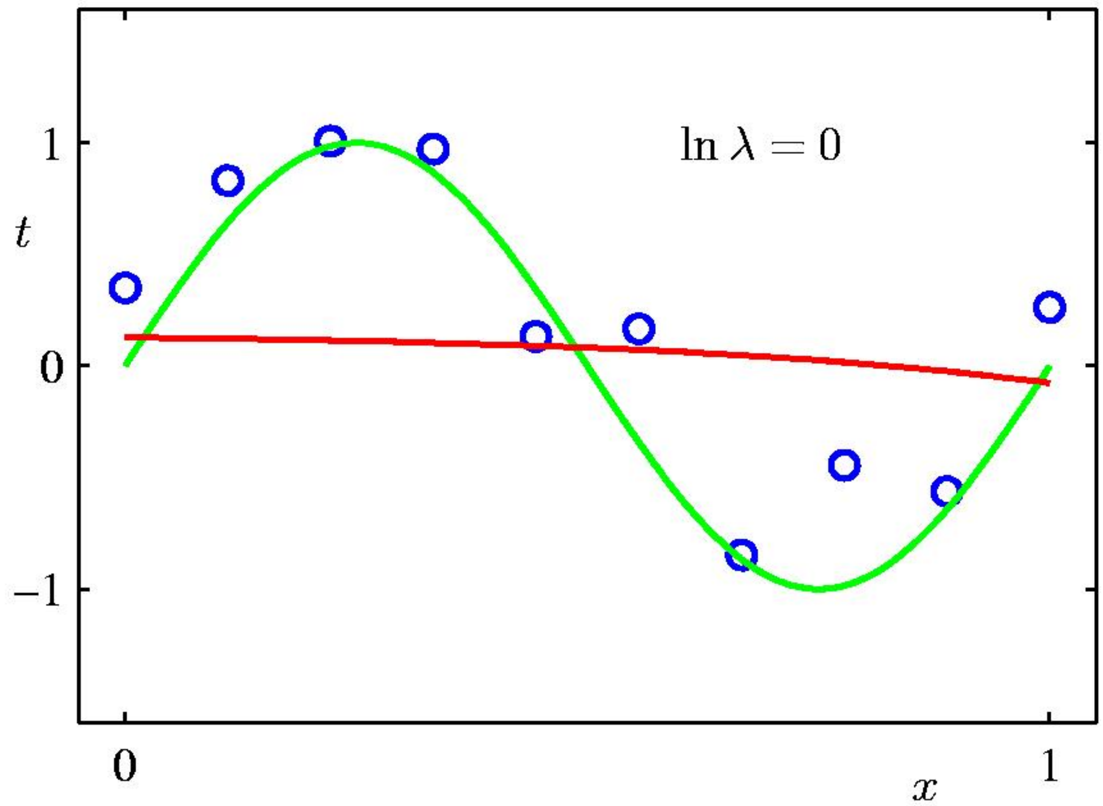
$$\ln \lambda = -18$$



Right λ –good fit

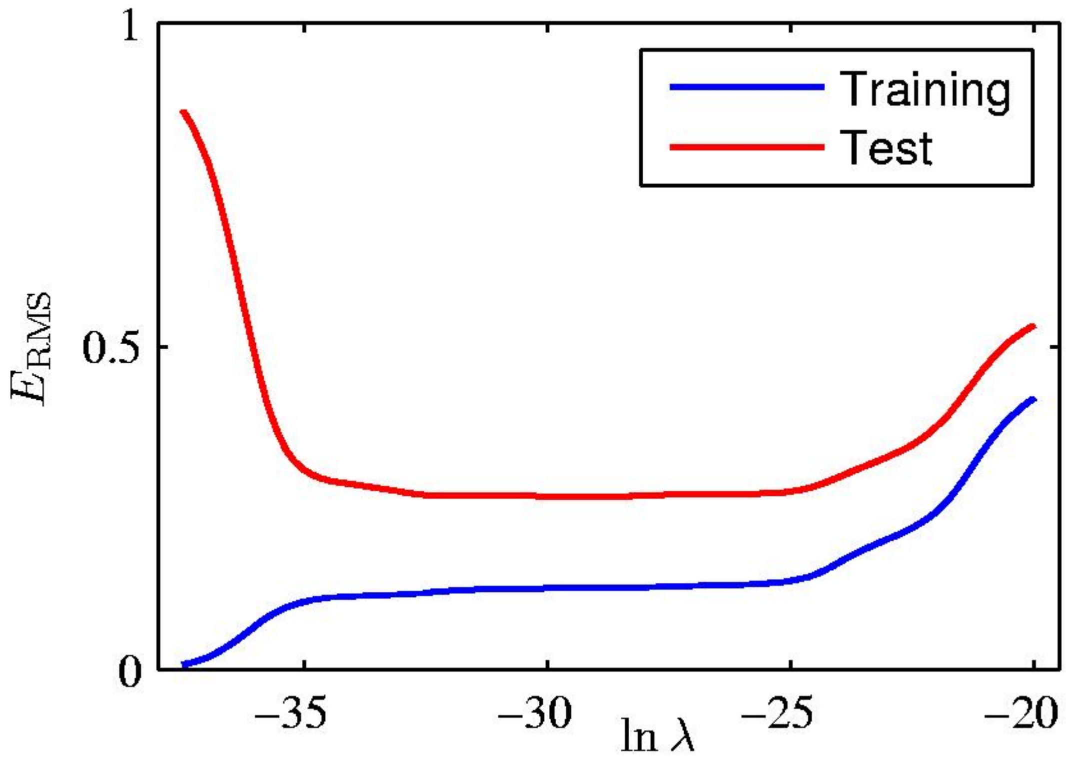
Regularization:

$$\ln \lambda = 0$$



Large λ –regularization dominates

Regularization: E_{RMS} *vs.* $\ln \lambda$

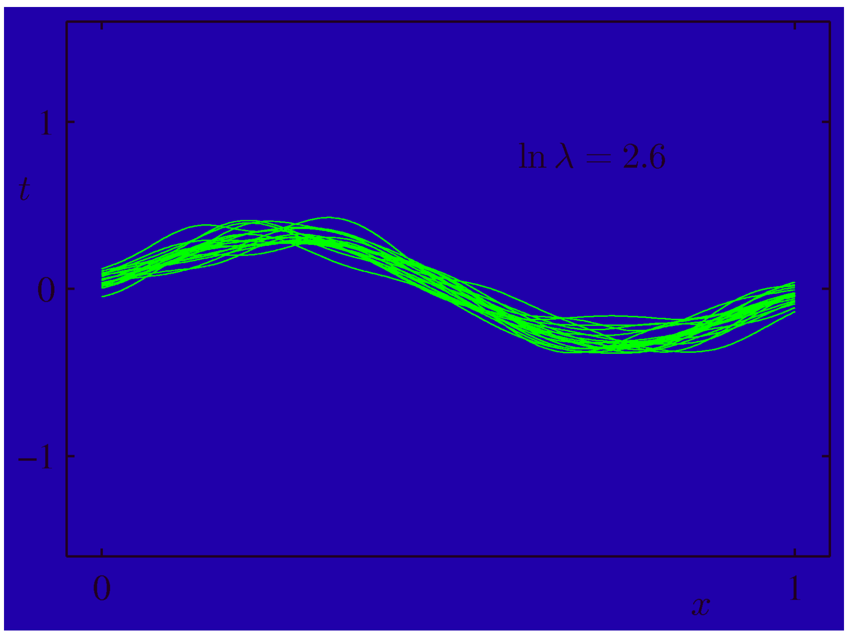


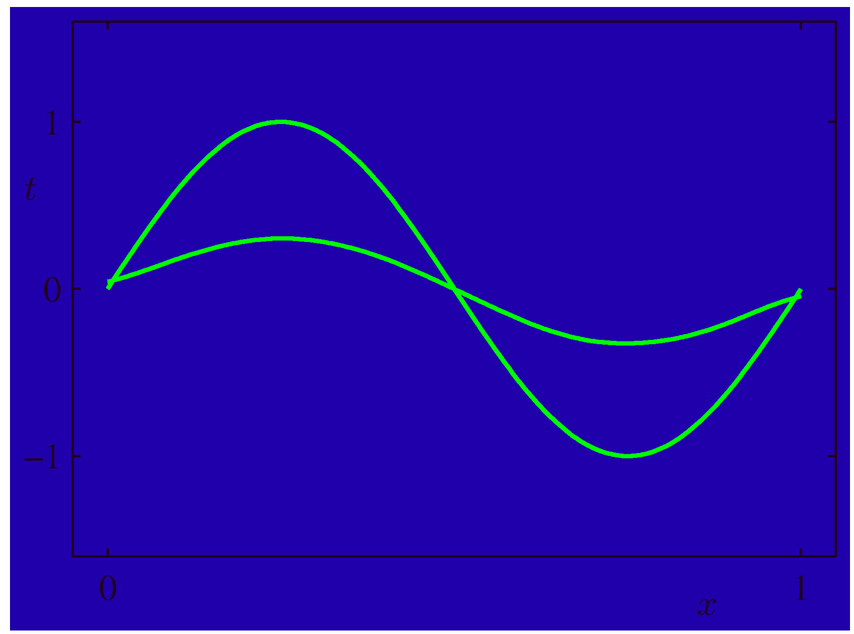
Polynomial Coefficients

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	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

The Bias-Variance Decomposition (5)

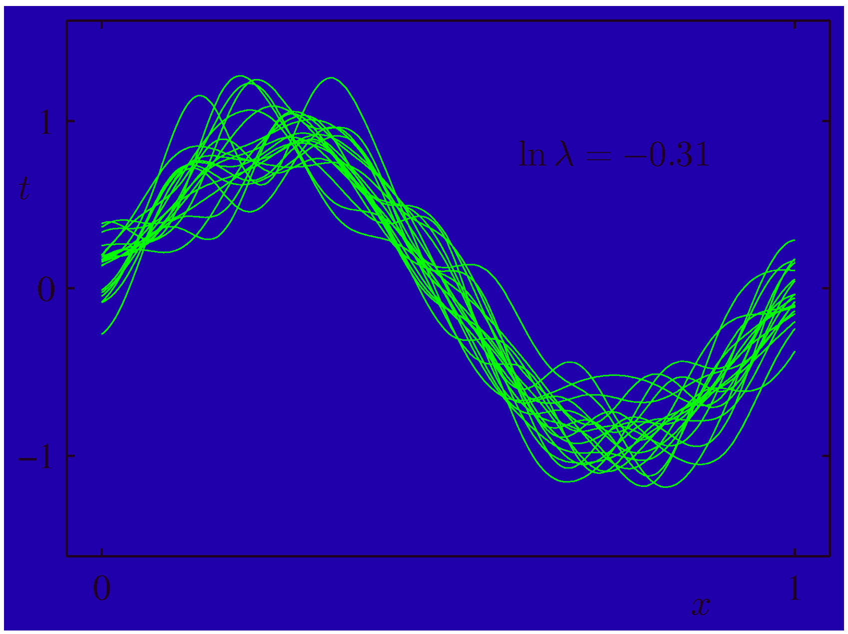
- Example: 100 data sets, each with 25 data points from the sinusoidal h(x) = sin(2px), varying the degree of regularization, λ.

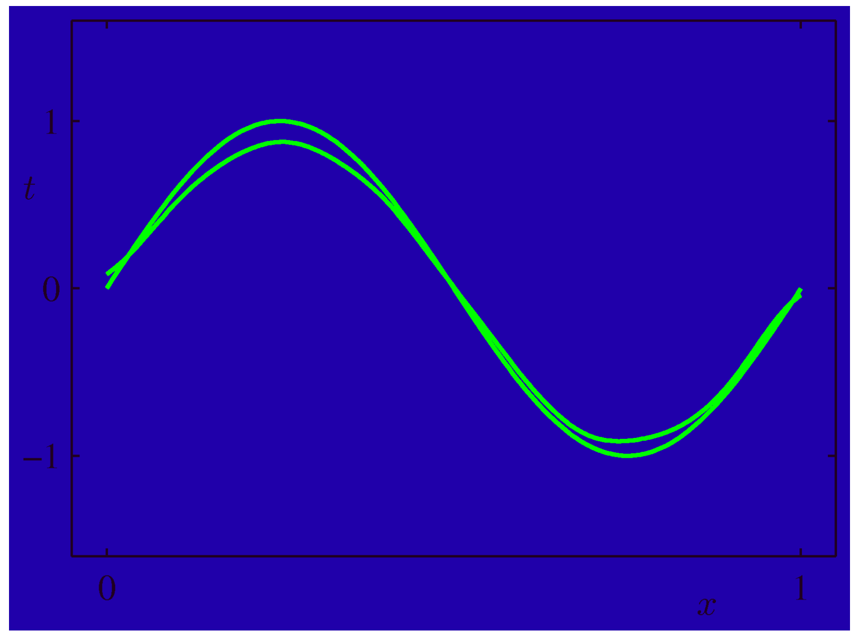




The Bias-Variance Decomposition (6)

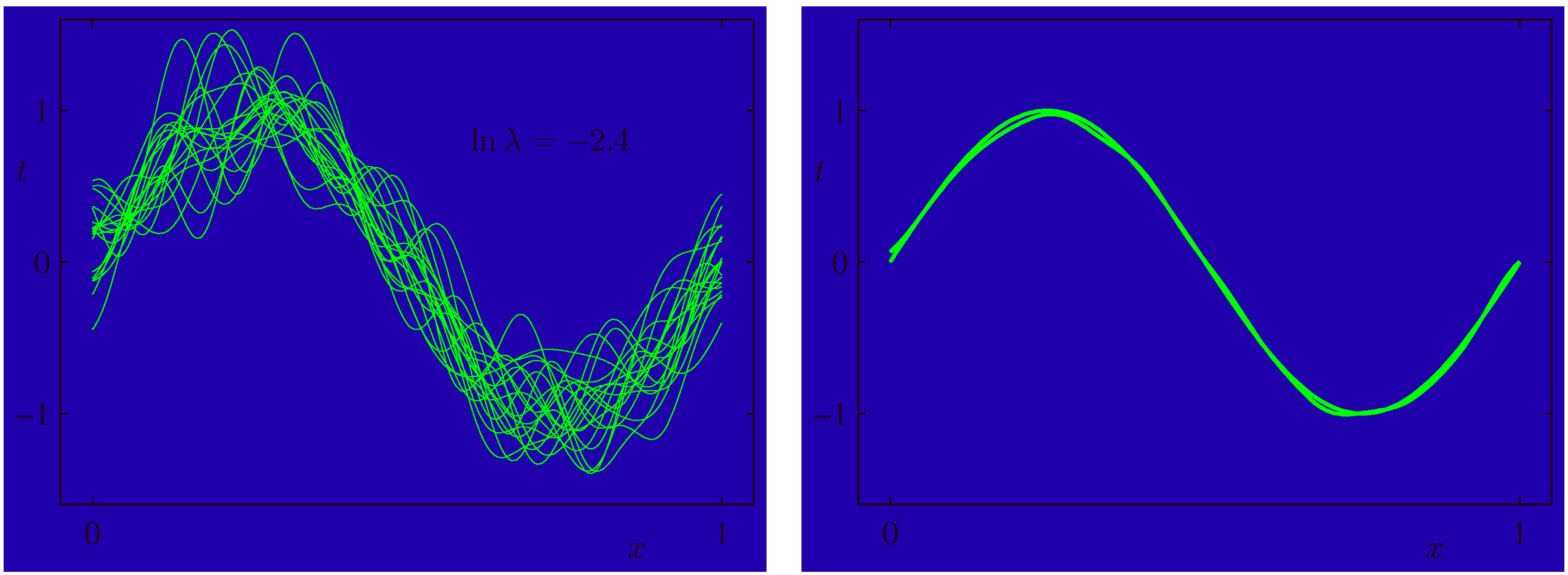
- Regularization constant $\lambda = \exp\{-0.31\}$.





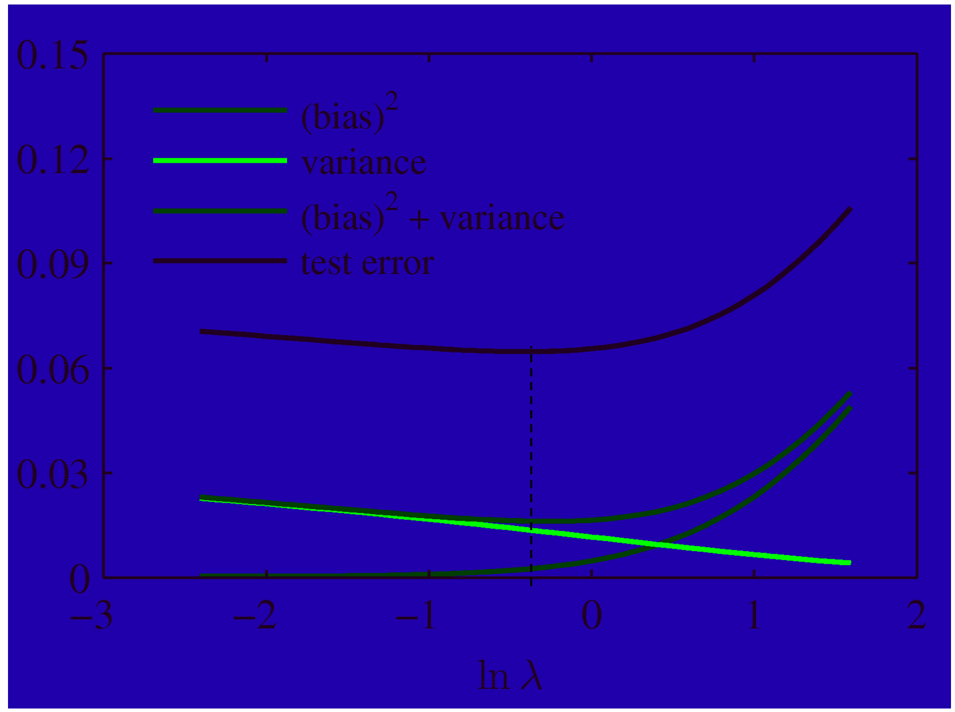
The Bias-Variance Decomposition (7)

- **Regularization constant λ = exp{-2.4}.**

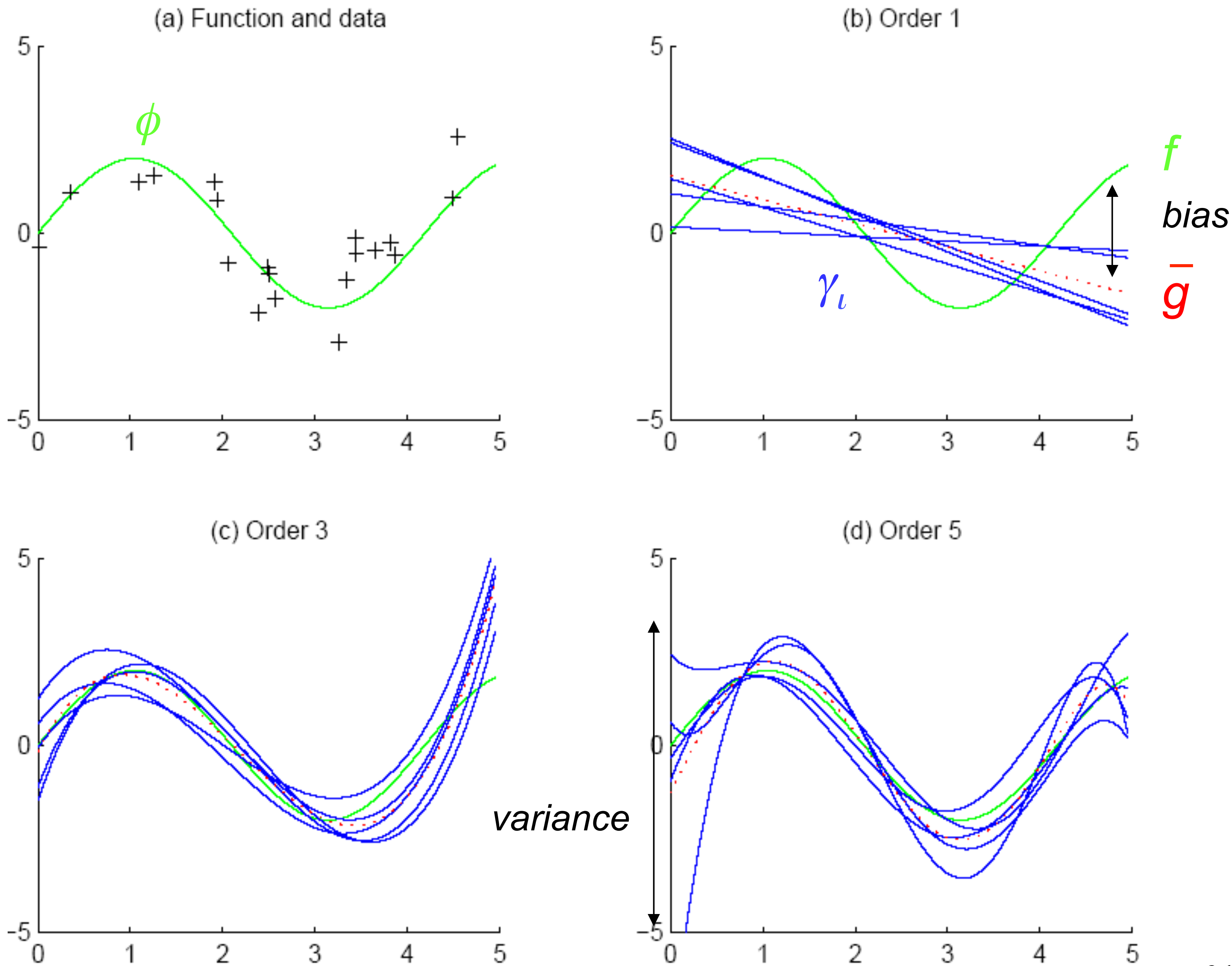


The Bias-Variance Trade-off

- From these plots, we note that;
  - **an over-regularized model (large λ) will have a high bias**
  - **while an under-regularized model (small λ) will have a high variance.**

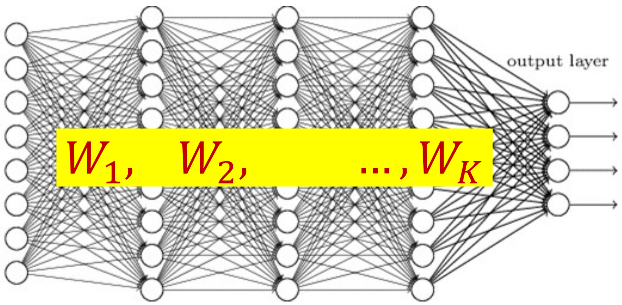


Minimum value of bias^2+variance is around λ=-0.31
This is close to the value that gives the minimum error on the test data.



(a) Function and data
ϕ
(b) Order 1
f
bias
$\bar{g}$
γ_i
(c) Order 3
(d) Order 5
variance

Objective function for neural networks



Y_t

Desired output of network: d_t

Error on i-th training input: $Div(Y_t, d_t; W_1, W_2, \ldots, W_K)$

Batch training error:

$$Err(W_1, W_2, \ldots, W_K) = \frac{1}{T} \sum_t Div(Y_t, d_t; W_1, W_2, \ldots, W_K)$$

- Conventional training: minimize the total error:

$$\widehat{W}_1, \widehat{W}_2, \ldots, \widehat{W}_K = \underset{W_1, W_2, \ldots, W_K}{\operatorname{argmin}} \; Err(W_1, W_2, \ldots, W_K)$$

Regularizing the weights

$$L(W_1, W_2, \dots, W_K) = \frac{1}{T}\sum_t Div(Y_t, d_t) + \frac{1}{2}\lambda \sum_k \|W_k\|_2^2$$

- Batch mode:

$$\Delta W_k = \frac{1}{T}\sum_t \nabla_{W_k} Div(Y_t, d_t)^T + \lambda W_k$$

- SGD:

$$\Delta W_k = \nabla_{W_k} Div(Y_t, d_t)^T + \lambda W_k$$

- Minibatch:

$$\Delta W_k = \frac{1}{b}\sum_{\tau=t}^{t+b-1} \nabla_{W_k} Div(Y_\tau, d_\tau)^T + \lambda W_k$$

- Update rule:

$$W_k \leftarrow W_k - \eta \Delta W_k$$

Incremental Update: Mini-batch update

- Given (X_1, d_1), (X_2, d_2),…, (X_T, d_T)
- Initialize all weights $W_1, W_2, \ldots, W_K$; $j = 0$
- Do:
  - Randomly permute (X_1, d_1), (X_2, d_2),…, (X_T, d_T)
  - For $t = 1: b: T$
    - $j = j + 1$
    - For every layer k:
      - $\Delta W_k = 0$
    - For t’ = t : t+b-1
      - For every layer k:
        - » Compute $\nabla_{W_k} Div(Y_t, d_t)$
        - » $\Delta W_k = \Delta W_k + \nabla_{W_k} Div(Y_t, d_t)$
    - Update
      - For every layer k:

$$W_k = W_k - \eta_j(\Delta W_k + \lambda W_k)$$

- Until Err has converged