

# Advanced Machine Learning

## Lecture 2: Linear models

Sandjai Bhulai  
Vrije Universiteit Amsterdam

s.bhulai@vu.nl  
8 September 2023



VRJE  
UNIVERSITEIT  
AMSTERDAM

Faculty of Science



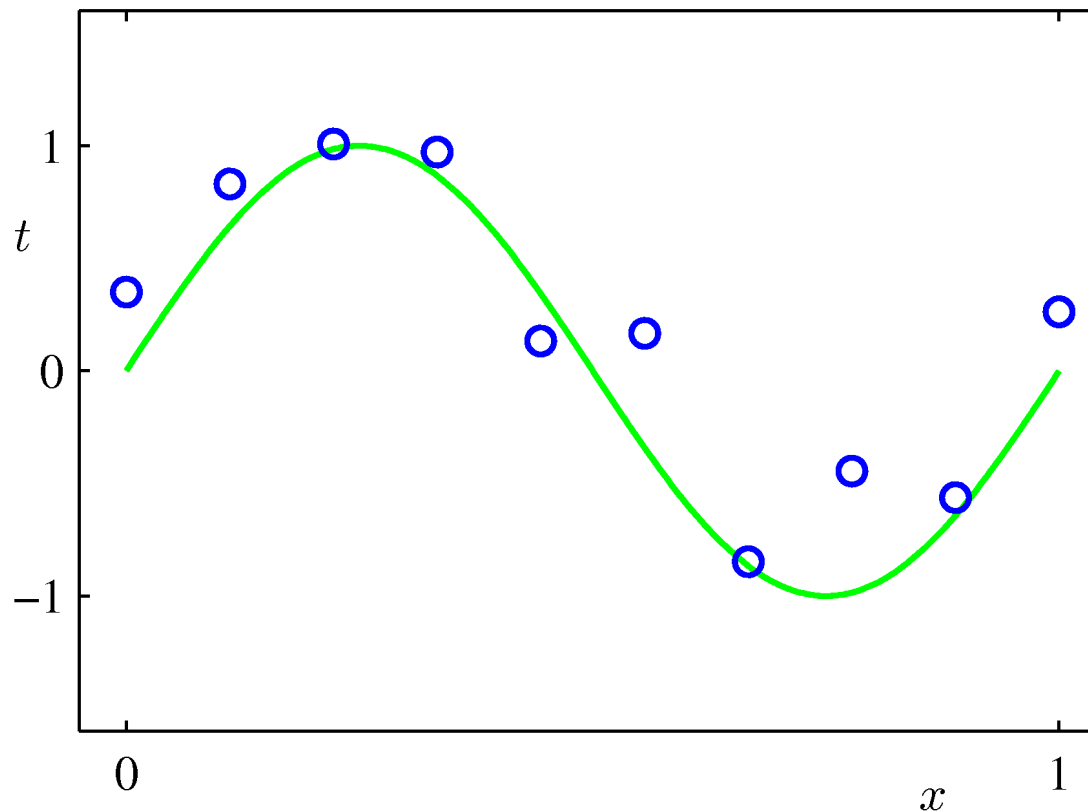
The background of the slide is a dark blue field filled with glowing white and light blue elements. These include streams of binary code (0s and 1s) that appear to be flowing or falling, similar to the 'Matrix' effect. There are also thin, intersecting lines and some larger, fainter data-like patterns scattered across the space, creating a sense of digital complexity and movement.

# Linear models

## Advanced Machine Learning

# Polynomial curve fitting

- 10 points sampled from  $\sin(2\pi x) + \text{disturbance}$



$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} + \dots$$

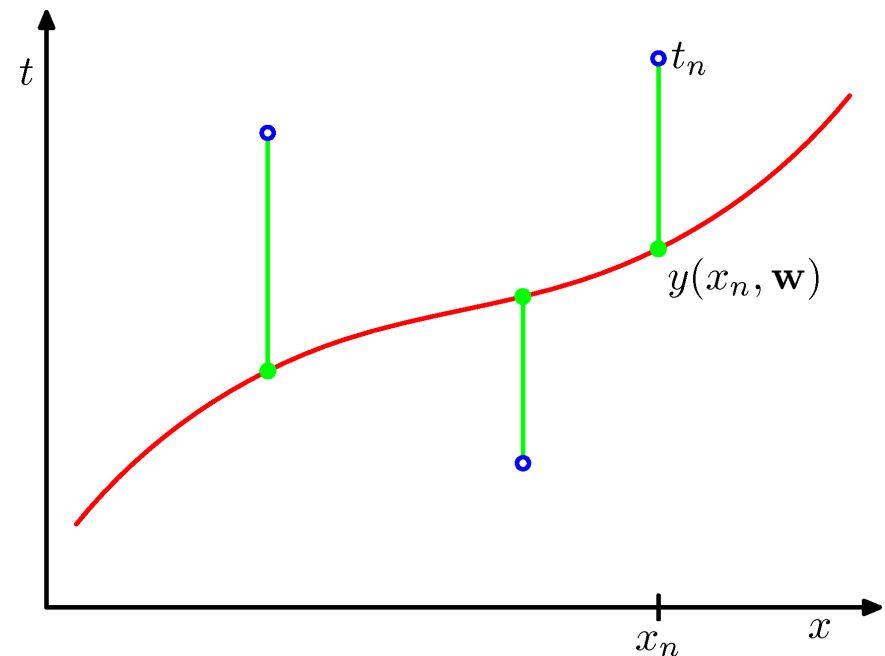
# Polynomial curve fitting

- Polynomial curve

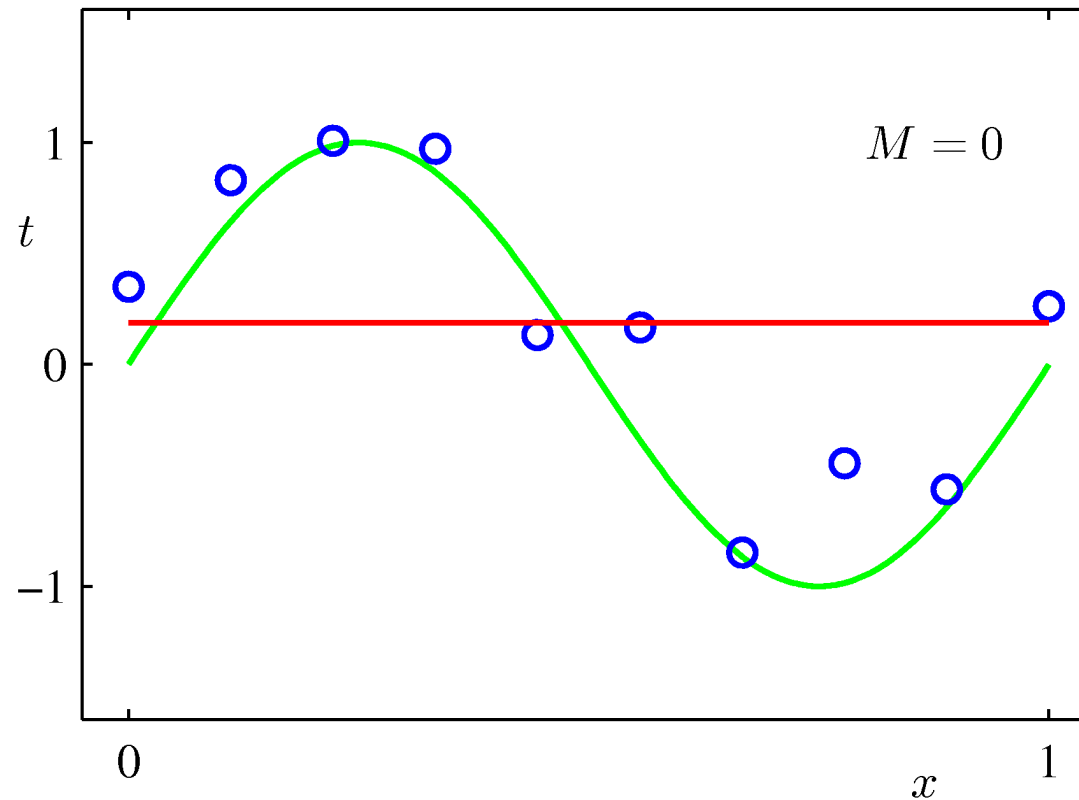
$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M = \sum_{j=0}^M w_jx^j$$

- Performance is measured by

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

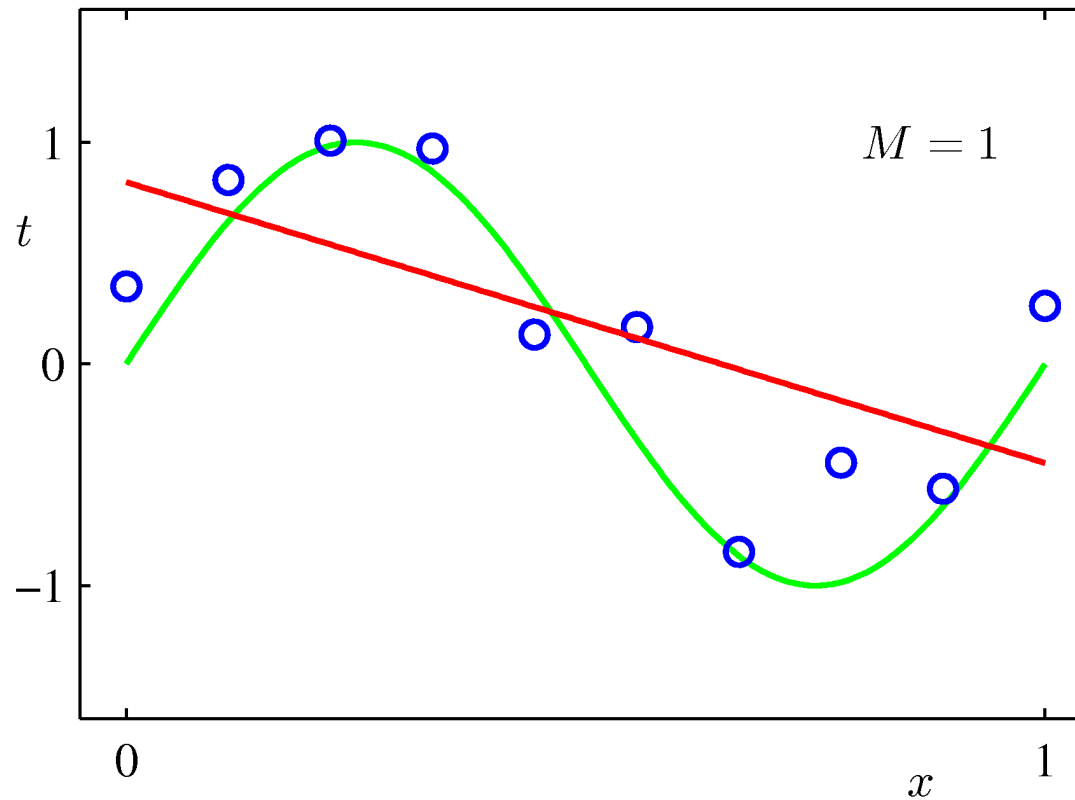


# Polynomial curve fitting: order 0



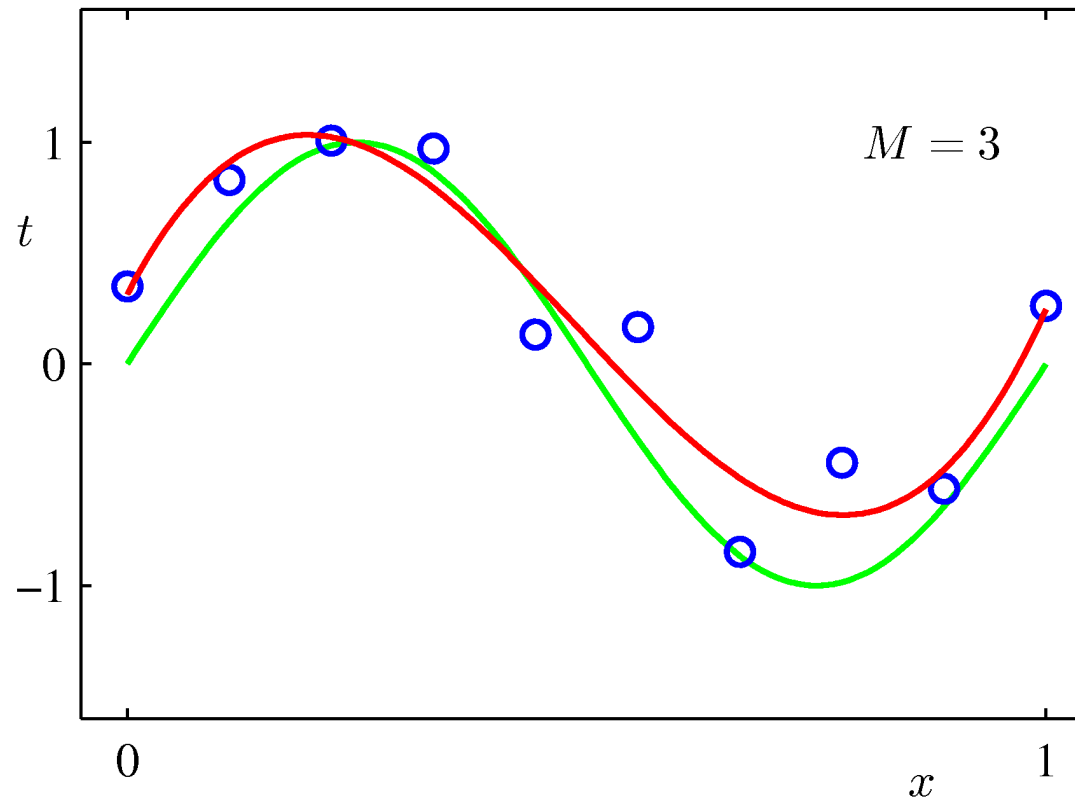
$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M = \sum_{j=0}^M w_jx^j$$

# Polynomial curve fitting: order 1



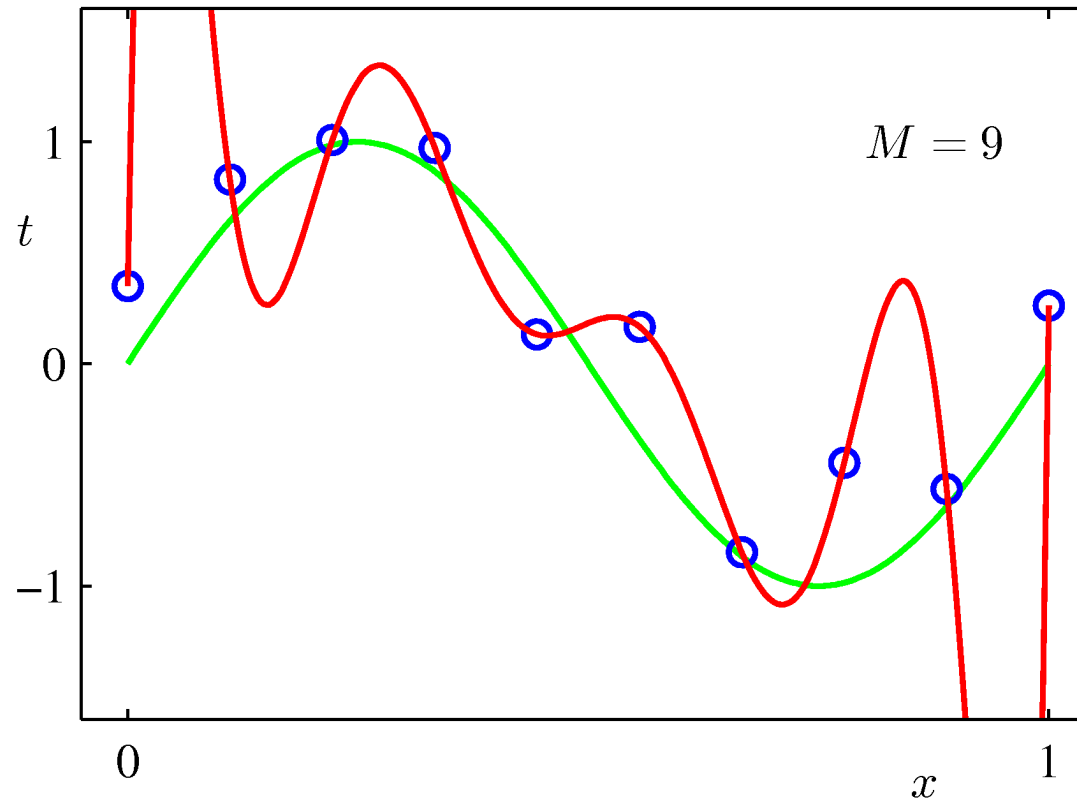
$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M = \sum_{j=0}^M w_jx^j$$

# Polynomial curve fitting: order 3



$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M = \sum_{j=0}^M w_jx^j$$

# Polynomial curve fitting: order 9

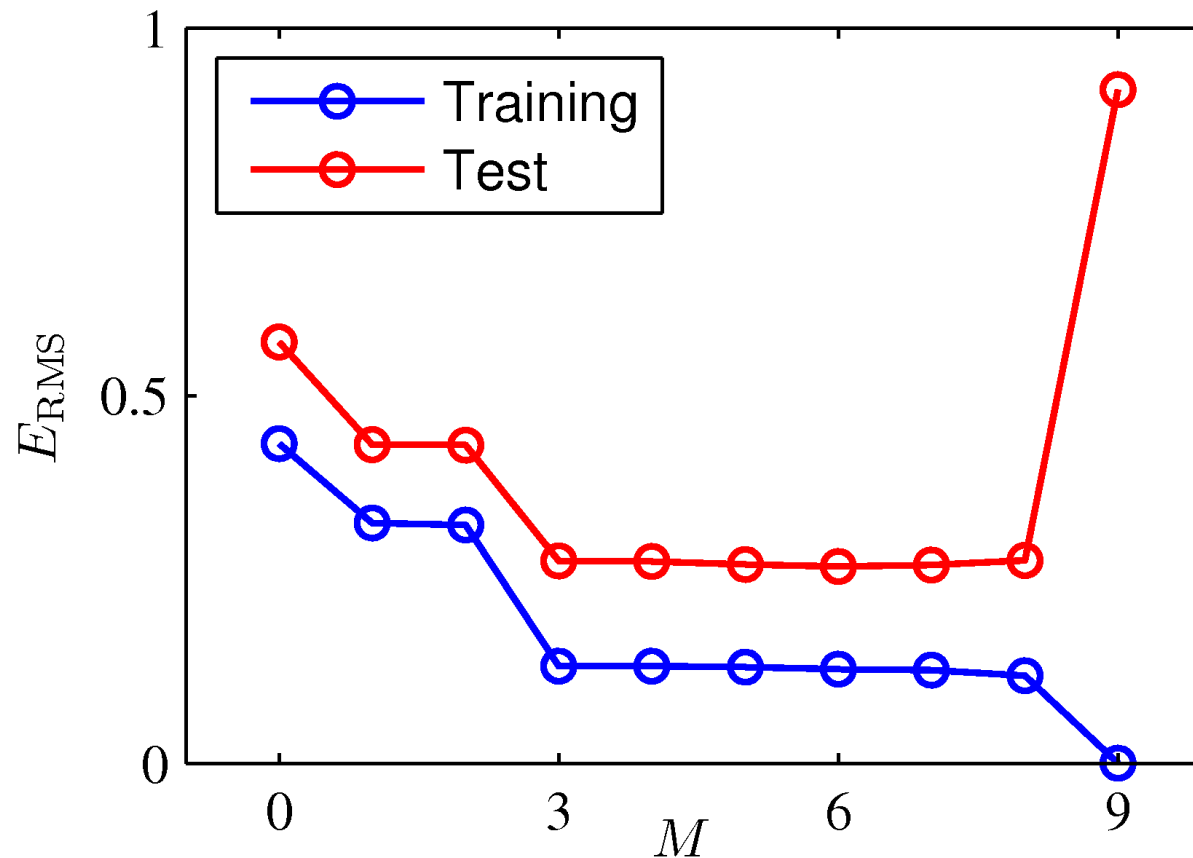


$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M = \sum_{j=0}^M w_jx^j$$



# Overfitting

- Root mean square (RMS) error:  $E_{\text{RMS}} = \sqrt{2E(\mathbf{w}^*)/N}$

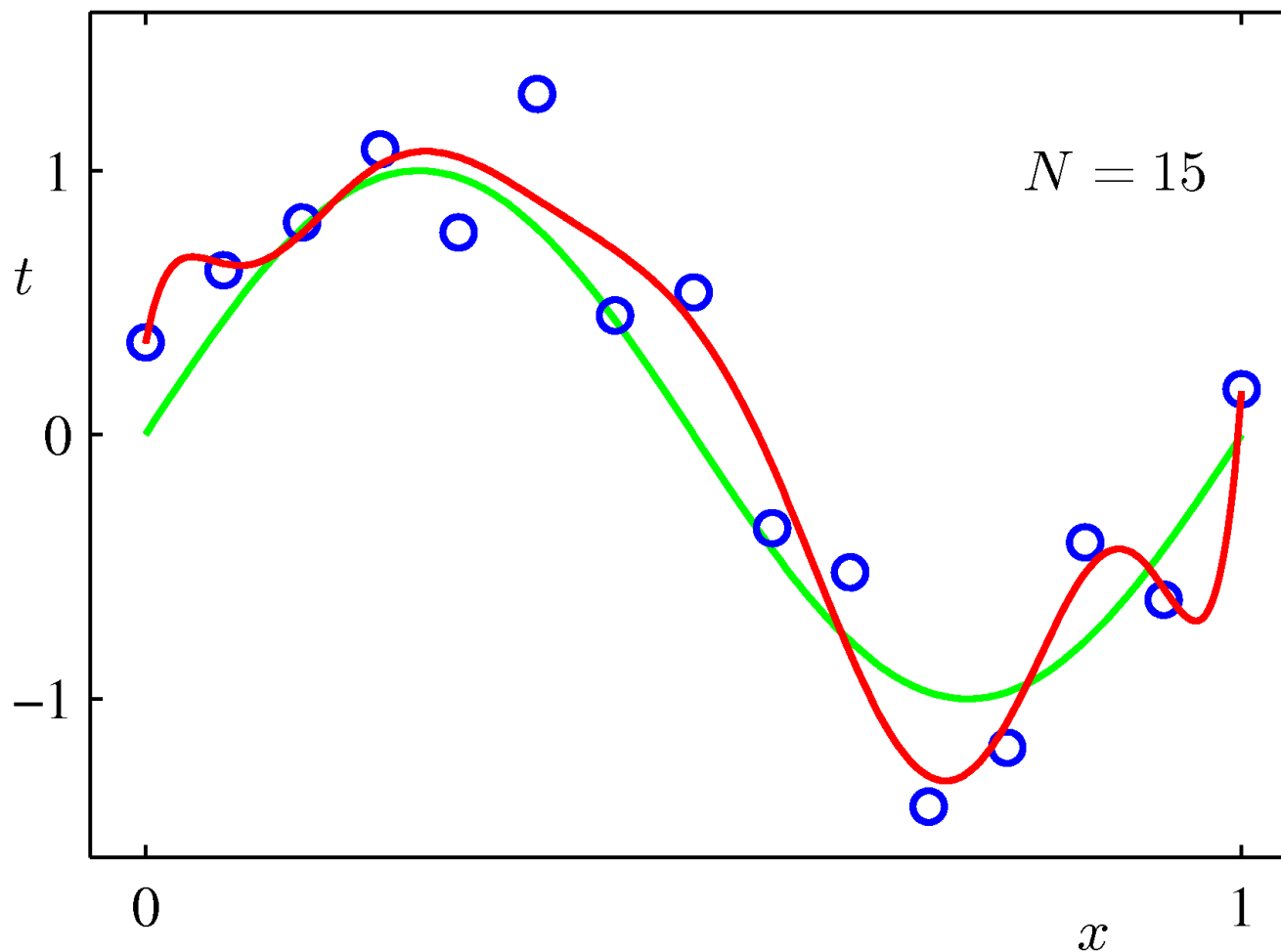


# Overfitting

	$M = 0$	$M = 1$	$M = 6$	$M = 9$
$w_0^*$	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

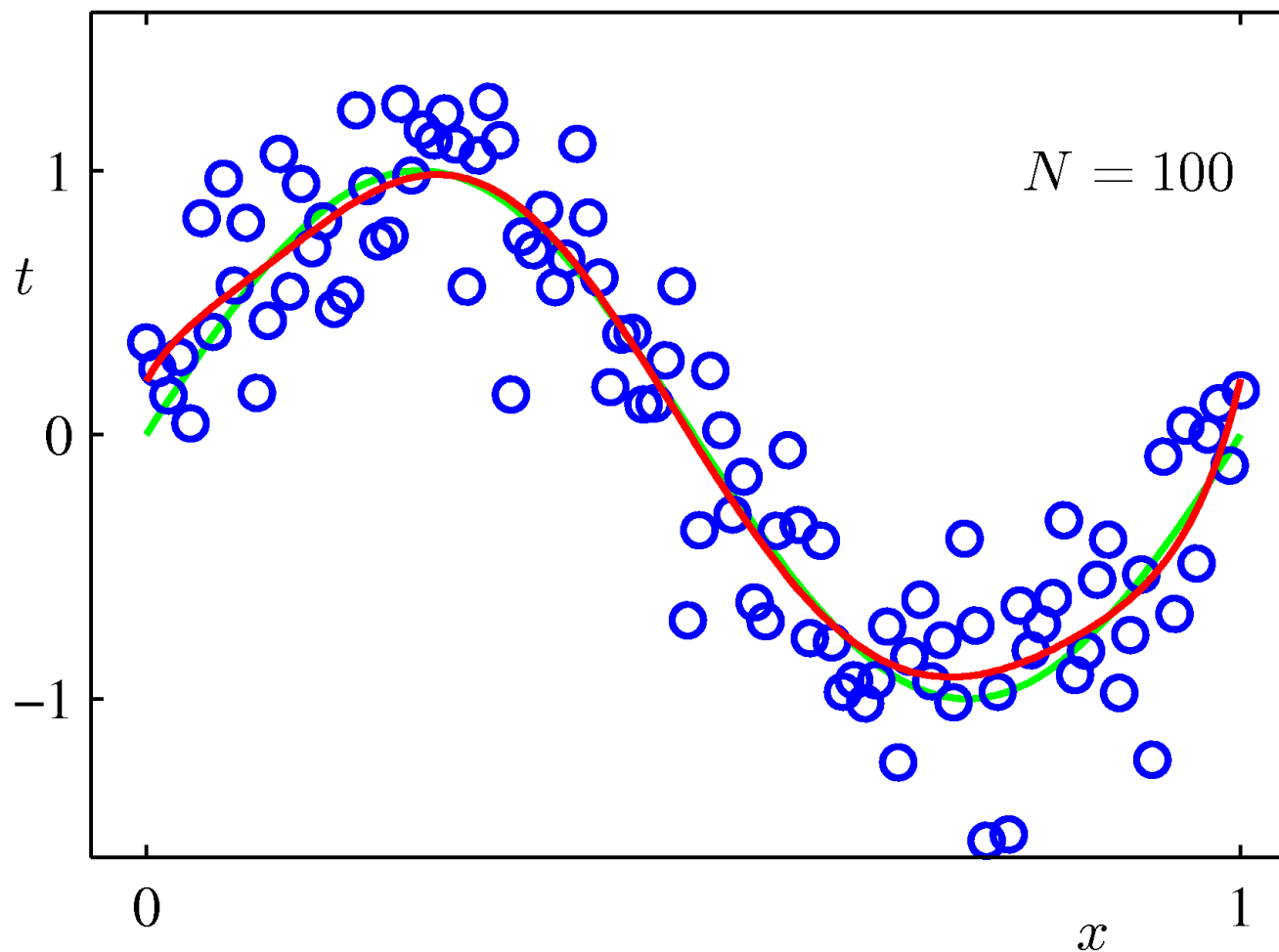
# Effect of dataset size

- Polynomial of order 9 and  $N = 15$



# Effect of dataset size

- Polynomial of order 9 and  $N = 100$



# Regularization

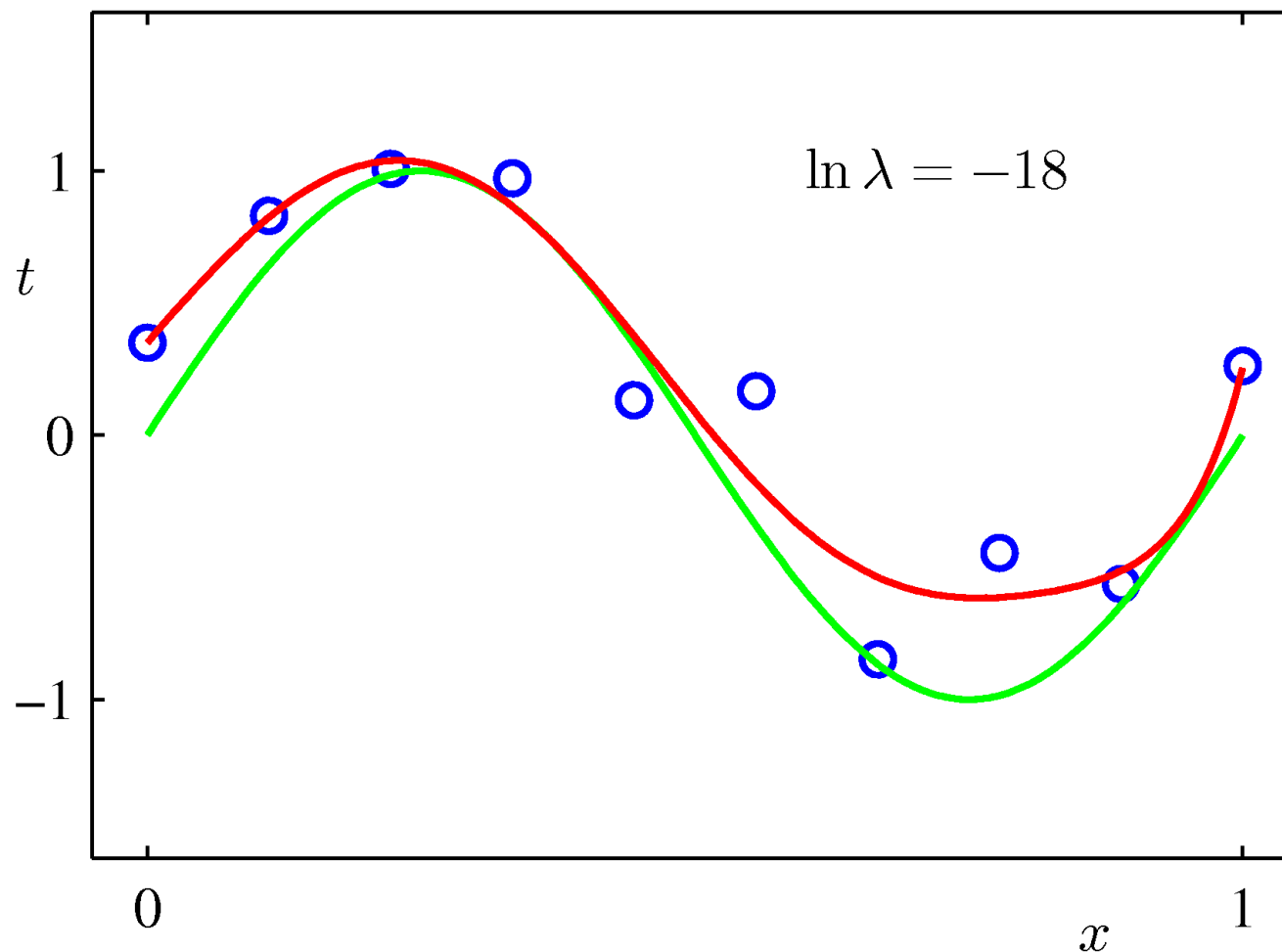
- Penalize large coefficients values

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

- $\lambda$  becomes a model parameter

# Regularization

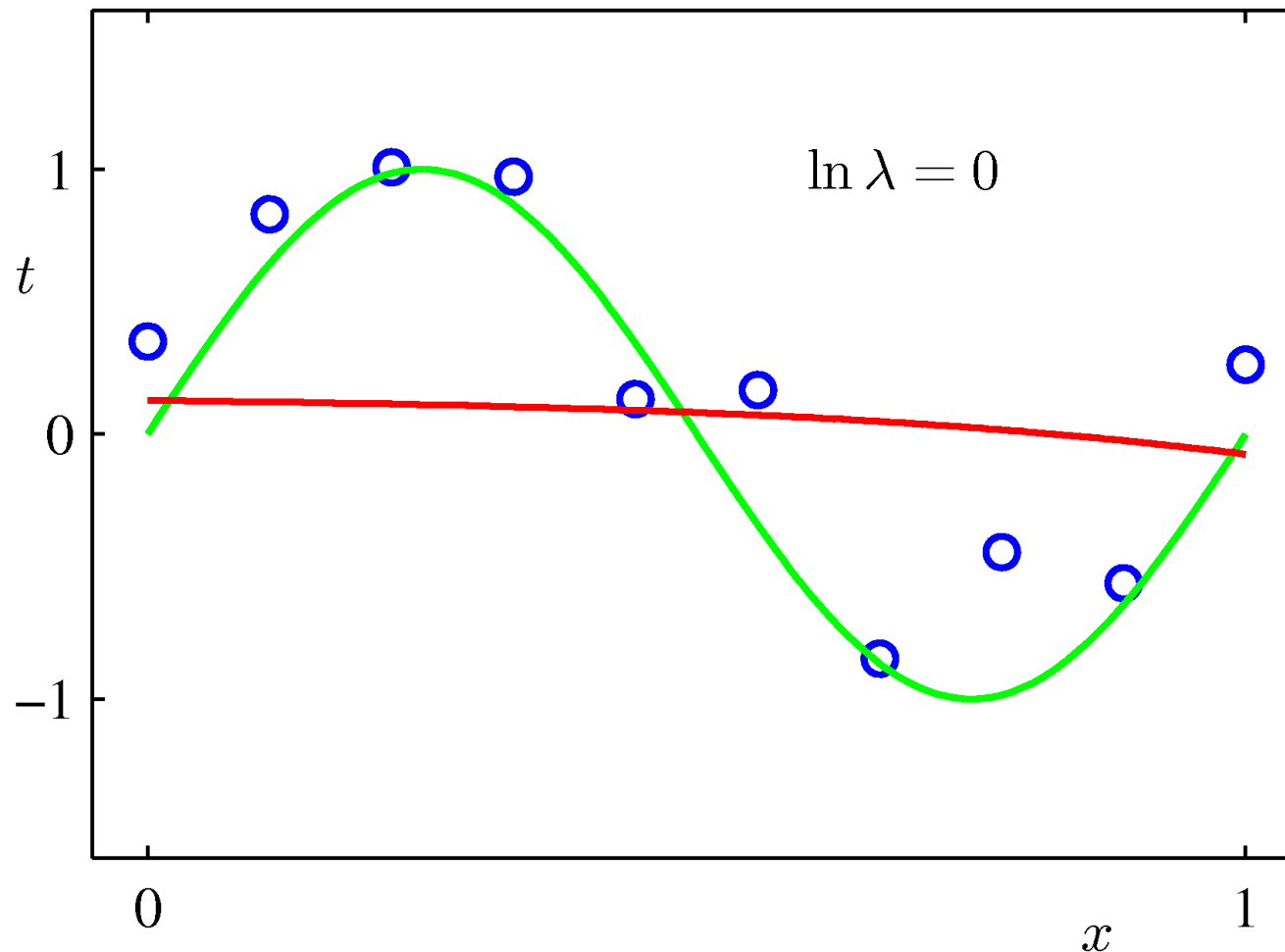
- Regularization with  $\ln \lambda = -18$





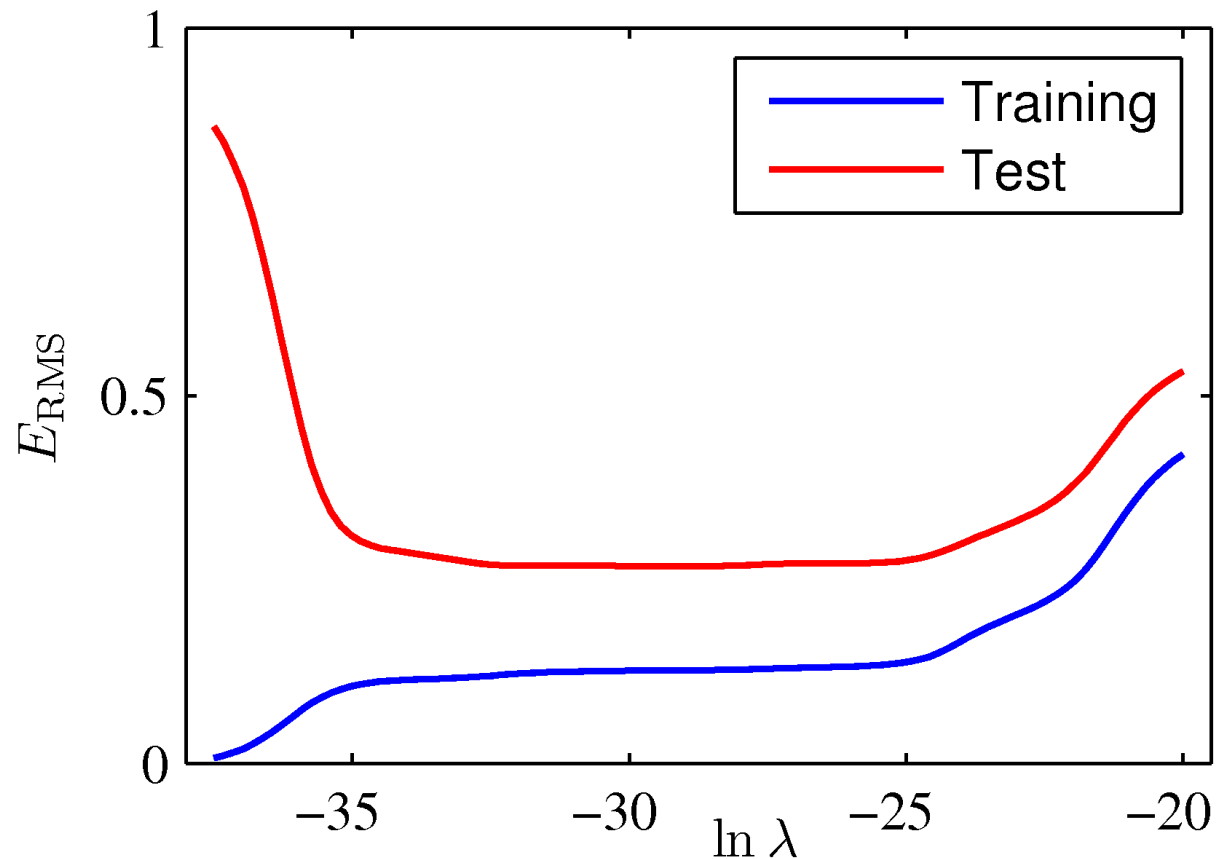
# Regularization

- Regularization with  $\ln \lambda = 0$



# Regularization

- $E_{\text{RMS}}$  versus  $\ln \lambda$



# Regularization

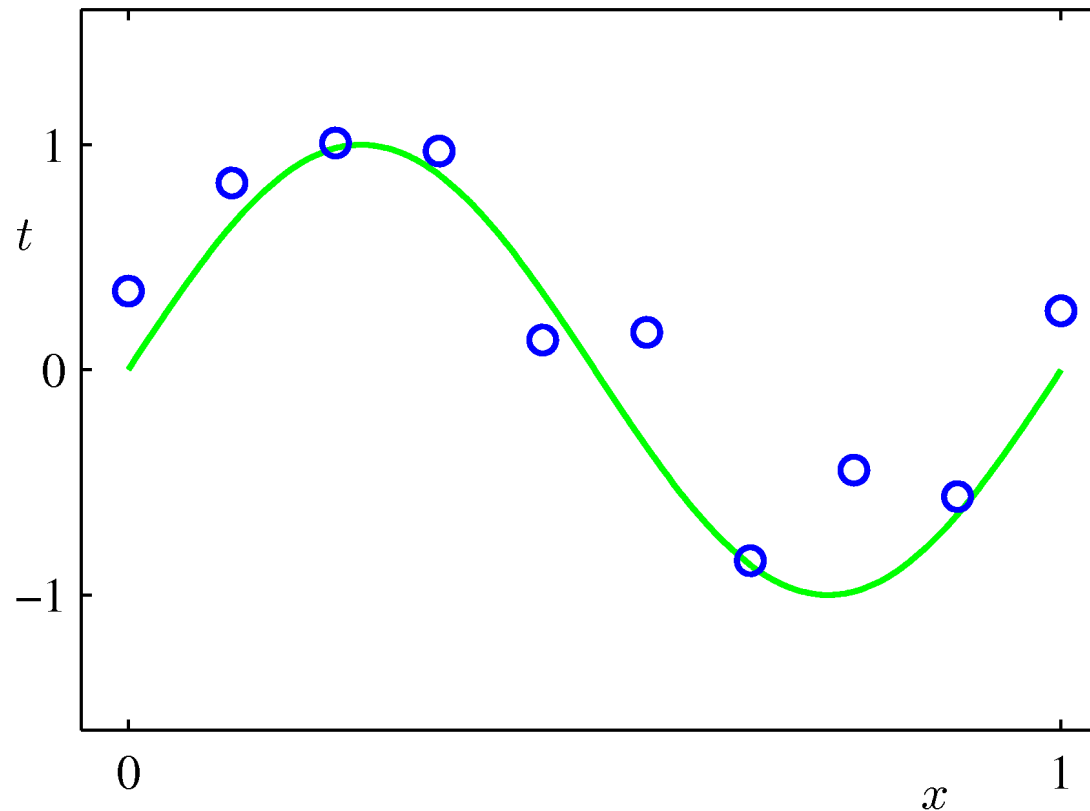
	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^*$	0.35	0.35	0.13
$w_1^*$	232.37	4.74	-0.05
$w_2^*$	-5321.83	-0.77	-0.06
$w_3^*$	48568.31	-31.97	-0.05
$w_4^*$	-231639.30	-3.89	-0.03
$w_5^*$	640042.26	55.28	-0.02
$w_6^*$	-1061800.52	41.32	-0.01
$w_7^*$	1042400.18	-45.95	-0.00
$w_8^*$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

The background of the slide is a dark blue field filled with glowing white and light blue elements. These include streams of binary code (0s and 1s) that appear to be flowing or falling, similar to the 'Matrix' effect. There are also thin, intersecting lines and some larger, fainter data-like patterns scattered across the space, creating a sense of digital complexity and movement.

# A deeper analysis

Advanced Machine Learning

# What is the issue?



$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} + \dots$$

# Linear basis function models

- General model is

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \varphi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x})$$

- $\varphi_j$  are known are basis functions
- Typically,  $\varphi_0(\mathbf{x}) = 1$  so that  $w_0$  acts as bias



# Linear basis function models

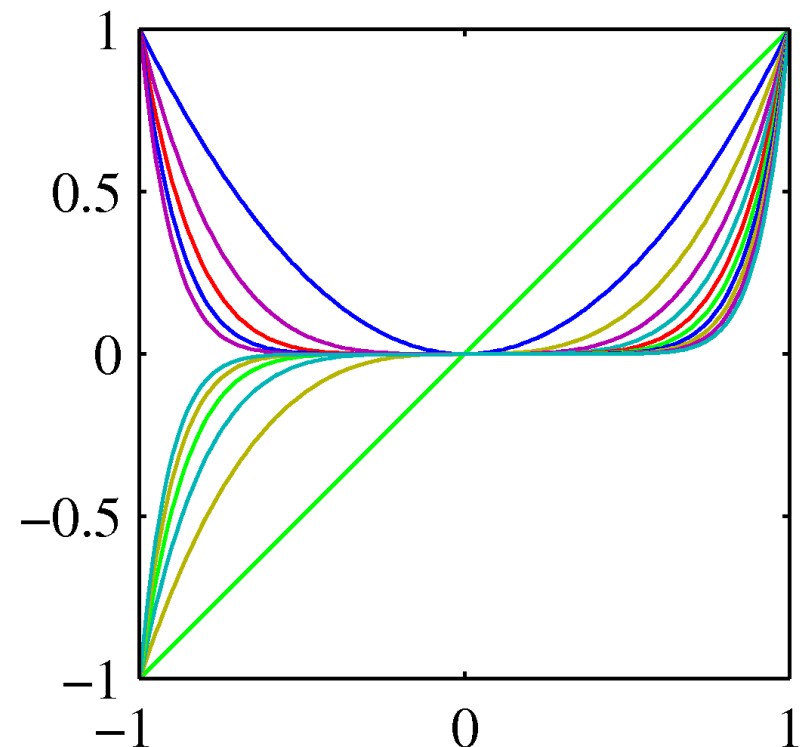
- General model is

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \varphi_j(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\varphi}(\mathbf{x})$$

- Polynomial basis functions:

$$\varphi_j(\mathbf{x}) = x^j$$

- These are global functions



# Linear basis function models

- General model is

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \varphi_j(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\varphi}(\mathbf{x})$$

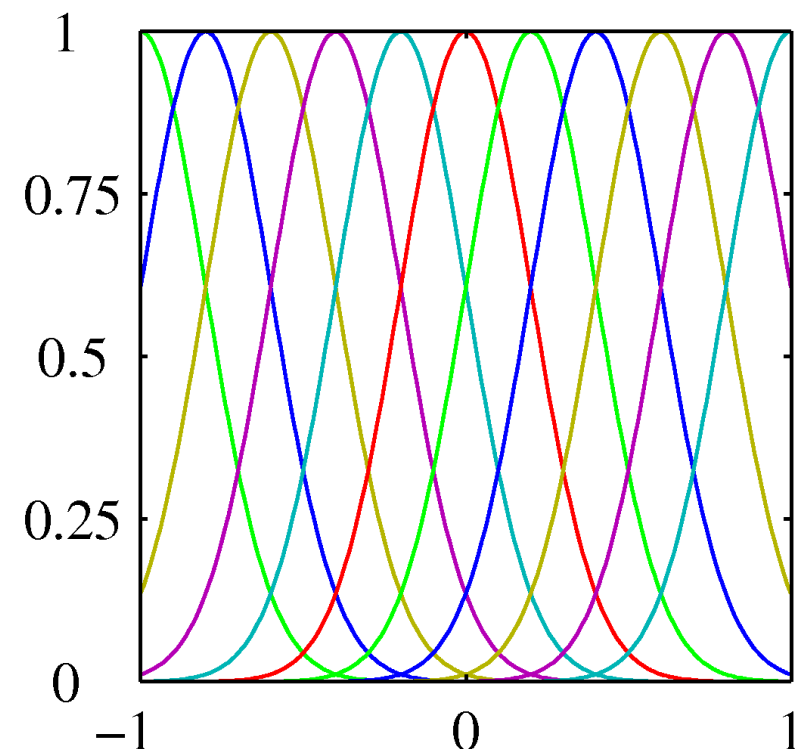
- Gaussian basis functions:

$$\varphi_j(\mathbf{x}) = \exp \left\{ -\frac{(x - \mu_j)^2}{2s^2} \right\}$$

- These are local functions

>  $\mu_j$  controls location

>  $s$  controls scale



# Linear basis function models

- General model is

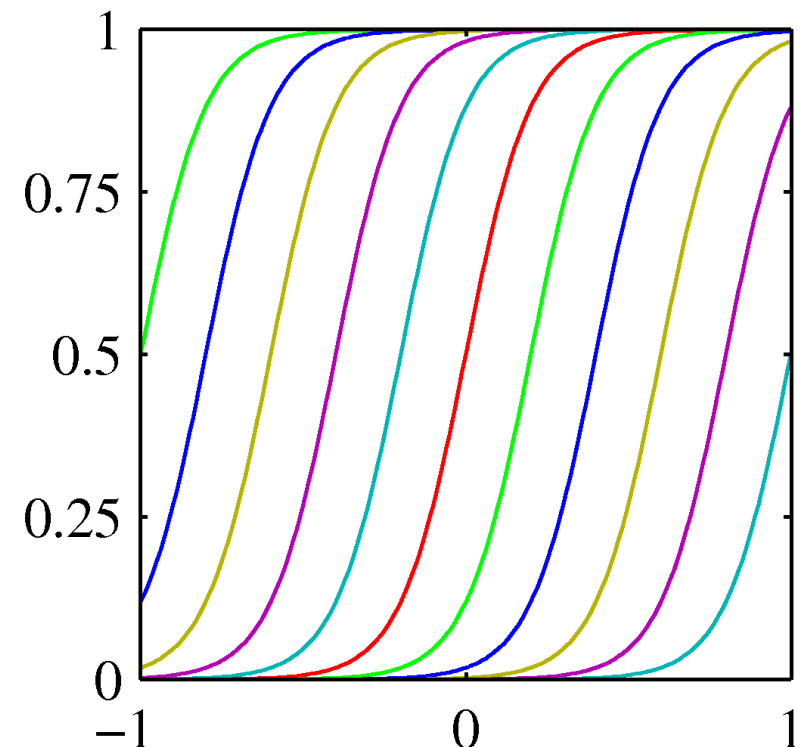
$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \varphi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x})$$

- Sigmoidal basis functions:

$$\varphi_j(\mathbf{x}) = \sigma\left(\frac{x - \mu_j}{s}\right)$$

where

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

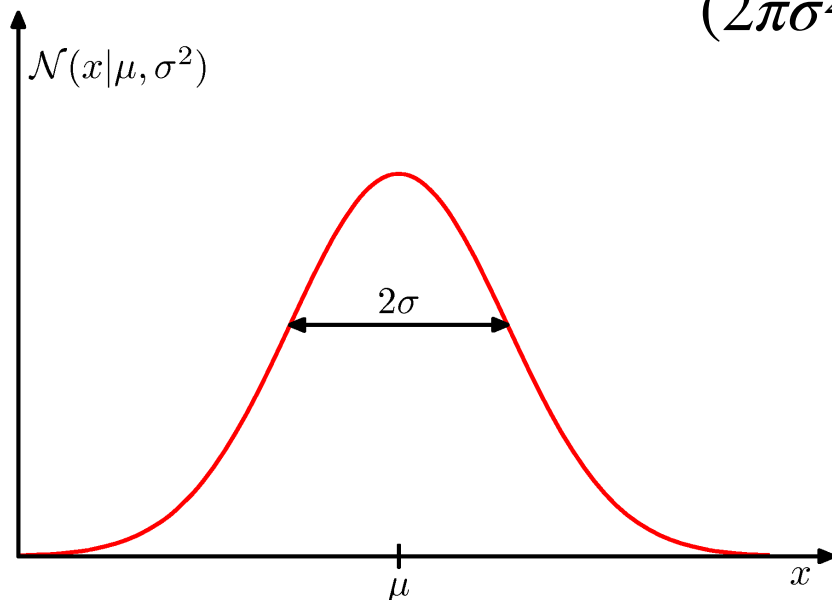


# Maximum likelihood

- Assume observations from a deterministic function with added Gaussian noise:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \quad \text{where} \quad p(\epsilon | \beta) = \mathcal{N}(\epsilon | 0, \beta^{-1})$$

- Note that  $\mathcal{N}(x | \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp \left\{ -\frac{1}{2\sigma^2}(x - \mu)^2 \right\}$



$$\beta = 1/\sigma^2$$

$$\mathcal{N}(x | \mu, \sigma^2) > 0$$

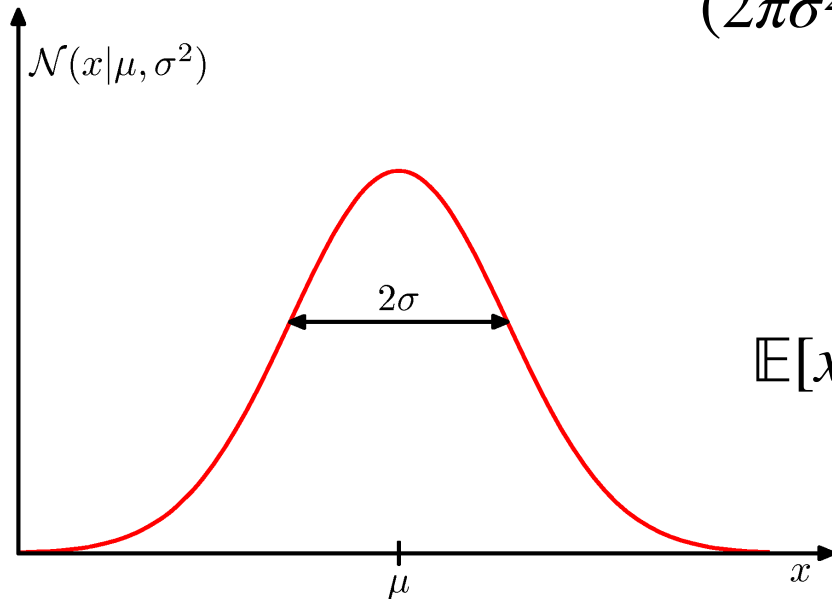
$$\int_{-\infty}^{\infty} \mathcal{N}(x | \mu, \sigma^2) dx = 1$$

# Maximum likelihood

- Assume observations from a deterministic function with added Gaussian noise:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \quad \text{where} \quad p(\epsilon | \beta) = \mathcal{N}(\epsilon | 0, \beta^{-1})$$

- Note that  $\mathcal{N}(x | \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp \left\{ -\frac{1}{2\sigma^2}(x - \mu)^2 \right\}$



$$\mathbb{E}[x] = \int_{-\infty}^{\infty} x \mathcal{N}(x | \mu, \sigma^2) dx = \mu$$

$$\mathbb{E}[x^2] = \int_{-\infty}^{\infty} x^2 \mathcal{N}(x | \mu, \sigma^2) dx = \mu^2 + \sigma^2$$

$$\text{var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \sigma^2$$

# Maximum likelihood

- Assume observations from a deterministic function with added Gaussian noise:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \quad \text{where} \quad p(\epsilon | \beta) = \mathcal{N}(\epsilon | 0, \beta^{-1})$$

- This is the same as saying

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

- Recall:  $y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \varphi_j(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\varphi}(\mathbf{x})$



# Maximum likelihood

- This is the same as saying

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

- Given observed inputs  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  and targets  $\mathbf{t} = [t_1, \dots, t_N]^\top$ , we obtain the likelihood function:

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

# Maximum likelihood

- Taking the logarithm, we get

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

where  $E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^\top \varphi(\mathbf{x}_n)\}^2$

- Recall:  $\mathcal{N}(x | \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp \left\{ -\frac{1}{2\sigma^2} (x - \mu)^2 \right\}$

# Maximum likelihood

- Computing the gradient and setting it to zero yields

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t} | \mathbf{w}, \beta) = \beta \sum_{n=1}^N \{t_n - \mathbf{w}^\top \boldsymbol{\varphi}(\mathbf{x}_n)\} \boldsymbol{\varphi}(\mathbf{x}_n)^\top = 0$$

The Moore-Penrose  
pseudo-inverse

- Solve for  $\mathbf{w}$ , we get

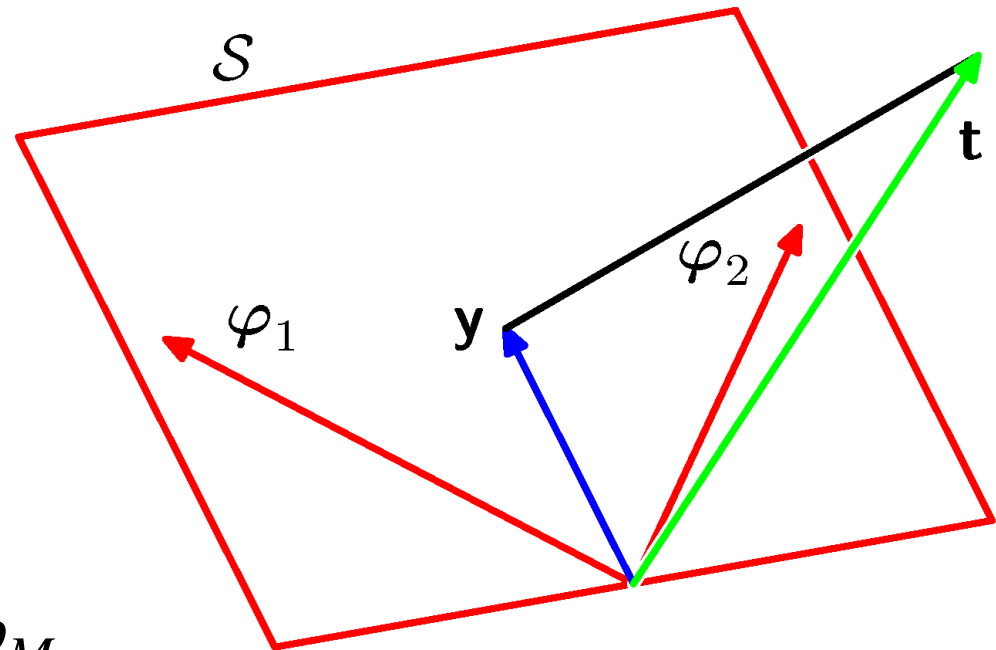
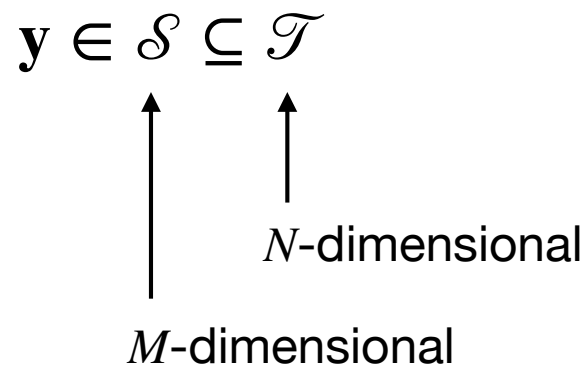
$$\mathbf{w}_{\text{ML}} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \mathbf{t}$$

with

$$\boldsymbol{\Phi} = \begin{pmatrix} \varphi_0(\mathbf{x}_1) & \varphi_1(\mathbf{x}_1) & \cdots & \varphi_{M-1}(\mathbf{x}_1) \\ \varphi_0(\mathbf{x}_2) & \varphi_1(\mathbf{x}_2) & \cdots & \varphi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_0(\mathbf{x}_N) & \varphi_1(\mathbf{x}_N) & \cdots & \varphi_{M-1}(\mathbf{x}_N) \end{pmatrix}$$

# Interpretation

- Consider  $\mathbf{y} = \Phi \mathbf{w}_{\text{ML}} = [\varphi_1, \dots, \varphi_M] \mathbf{w}_{\text{ML}}$



- $\mathcal{S}$  is spanned by  $\varphi_1, \dots, \varphi_M$
- $\mathbf{w}_{\text{ML}}$  minimizes the distance between  $\mathbf{t}$  and its orthogonal projection on  $\mathcal{S}$ , i.e.,  $\mathbf{y}$

# Regularization

- Consider the error function

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

data term + regularization term

- With the sum-of-squares error function and a quadratic regularizer, we get

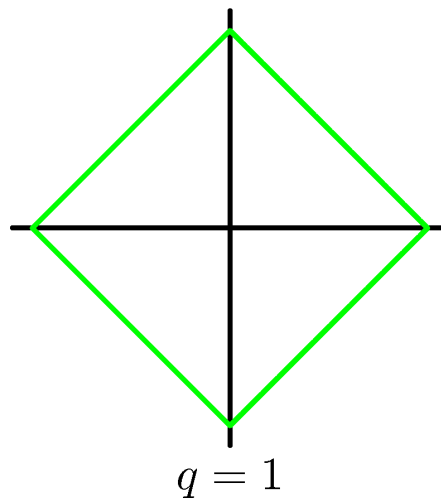
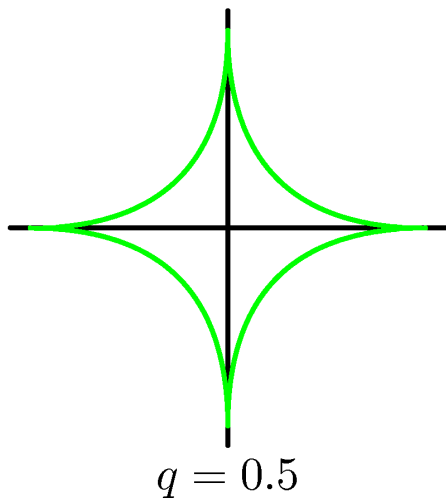
$$\frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^\top \varphi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2} \mathbf{w}^\top \mathbf{w}$$

- This is minimized by  $\mathbf{w} = (\lambda \mathbf{I} + \Phi^\top \Phi)^{-1} \Phi^\top \mathbf{t}$

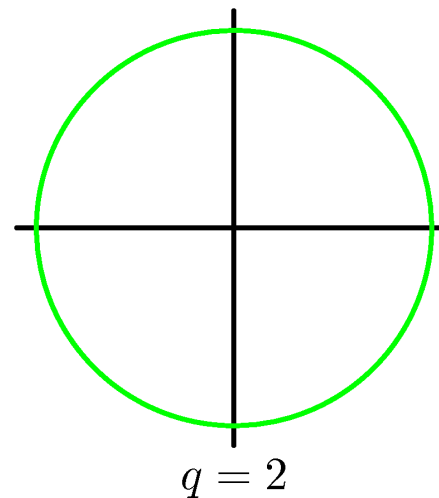
# Regularization

- With a more general regularizer, we have

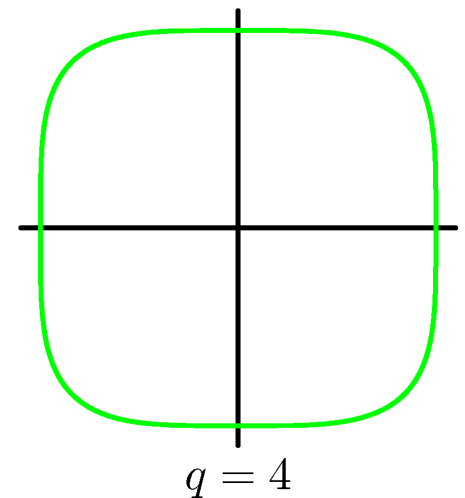
$$\frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^\top \varphi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2} \sum_{j=1}^M |w_j|^q$$



lasso



quadratic





# Regularization

- Lasso tends to generate sparser solutions than a quadratic regularizer

