Lecture 1

ARC-203
10.20-11.40 am MTh
PHYS568

Introduction

Data analysis: automated discovery of regularities & patterns in data, subsequent modeling (e.g. physical modeling)

Ex.: digit recognition: pixels → {0,...,9}
$\vec{x}$ → vector of pixels; single-digit output

Training set: $\{\vec{x}_1 \dots \vec{x}_N\}$
(e.g. hand-labeled)
⇓
$\vec{t}$ (N-dim target vector)
$t_i = \{0,\dots,9\}$

Trained model can then be used to predict digits in a test set.

Often: preprocessing / feature extraction
↓
find useful features that are fast to compute

Supervised learning problem: training, then predicting.

↓ classification [discrete categories]

↓ regression [real-valued predictions]

Unsupervised learning problem:

$\{\vec{x}_1 \dots \vec{x}_N\}$ but no target values

- ↓ clustering
- ↓ density estimation
- ↘ dimensionality reduction (visualization)

Reinforcement learning

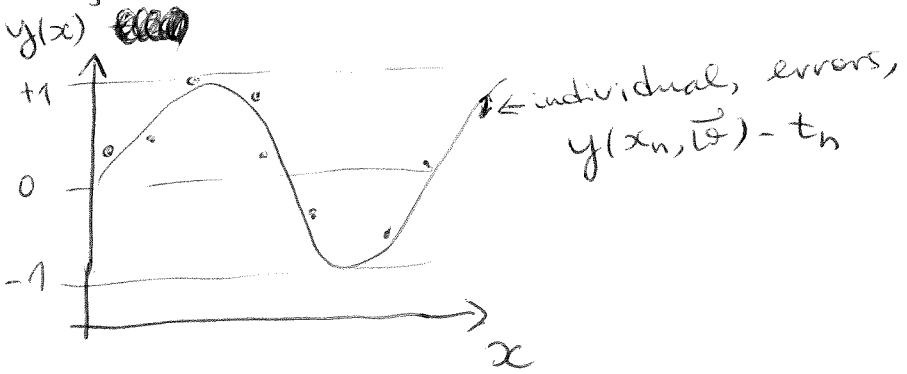
no training data at all → learn by trial and error

Objective: max score or reward

Ex.: polynomial curve fitting

$\vec{x}_N$ comes from $\sin(2\pi x)$ + noise

$(N = 10)$



$\vec{t}$ also given, real-valued

Consider curve fitting:

$$y(x, \vec{w}) = \sum_{j=0}^{M} w_j x^j \quad \leftarrow \text{linear model (in } \vec{w}\text{)}$$

M - polynomial order

Minimize the error f'n:

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

$\frac{\partial E}{\partial w_j} = 0$ has a closed-form solution → $\rightarrow \vec{w}^*$

$$\frac{\partial E(\vec{w})}{\partial w_j} = \sum_{n=1}^{N} \left(y(x_n, \vec{w}) - t_n \right) \frac{\partial y(x_n, \vec{w})}{\partial w_j} =$$

$$= \sum_{n=1}^{N} \underbrace{\left(y(x_n, \vec{w}) - t_n \right)}_{\sum_{j'=0}^{M} w_{j'} x_n^{j'}} x_n^j = 0 \qquad j = 0, \dots, M$$

$$\sum_{n=1}^{N} t_n x_n^j = \sum_{j'=0}^{M} w_{j'} \underbrace{\left[\sum_{n=1}^{N} x_n^{j'} x_n^j \right]}_{M_{j'j}}$$

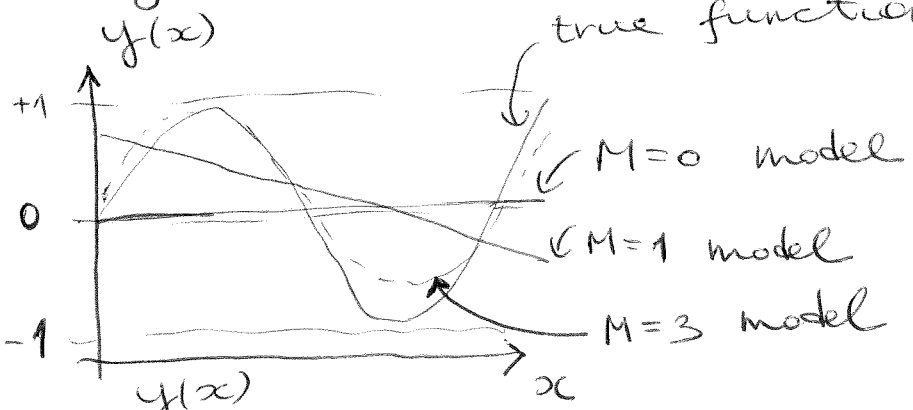
$\underbrace{\phantom{\sum_{n=1}^{N} t_n x_n^j}}_{C_j}$

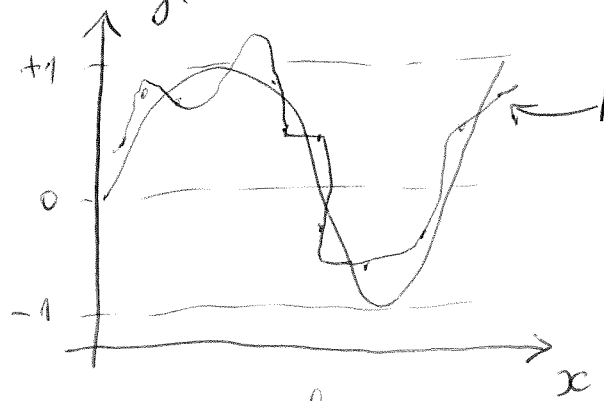
↑ linear in w_j

So, $\sum_{j'=0}^{M} w_{j'} M_{j'j} = C_j$ ← system of $M+1$ linear eq's for w_j, can be used to find w_j^{ML}.

How to choose M? (model selection)

Try M = 0, 1, 3, 9





M=9 model (overfitting)

$E(\vec{w}^*) = 0$ but the model is "bad"

In what way? Well, $|w_j^*|$ are often large & irregular when the model overfits.

How to fix it? One easy way → add more data, but the model complexity then depends on N (whereas it should depend on the complexity of the underlying process).

Minimizing least squares is a specific case of the ML approach → switching to Bayesian modeling automatically controls model complexity.

Another, more empirical technique is regularization:

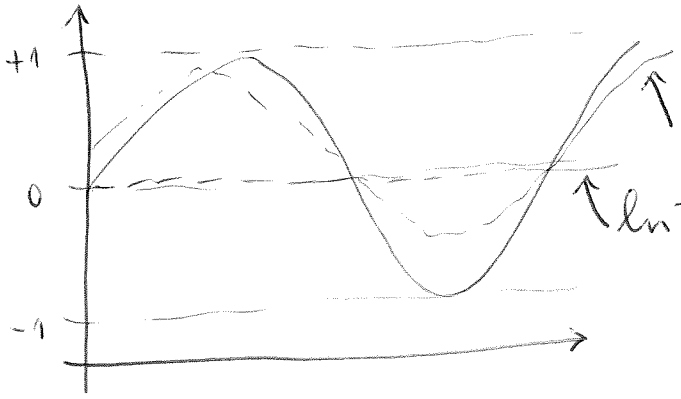
$$\tilde{E}(\vec{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(y(x_n,\vec{w}) - t_n\right)^2 + \frac{\lambda}{2}(w_0^2 + \ldots + w_M^2)$$

(w_0^2 may be omitted not to make things origin-dependent)

$$\frac{\partial \tilde{E}(\vec{w})}{\partial w_j} = 0 \quad , \forall j \text{ can still be evaluated}$$

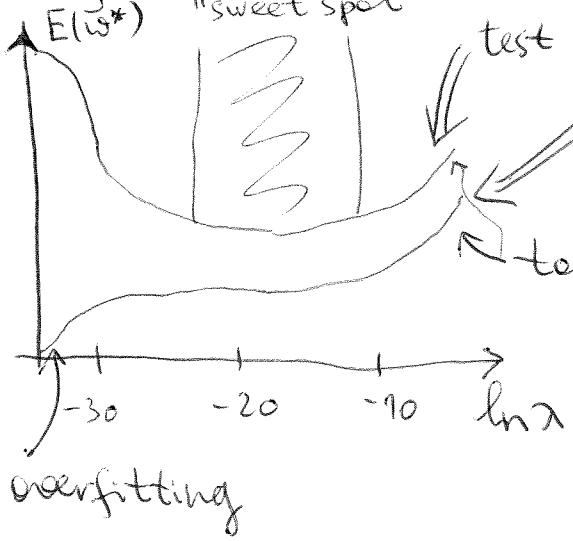
Drawback: things depend on λ.

$M=9$ fit



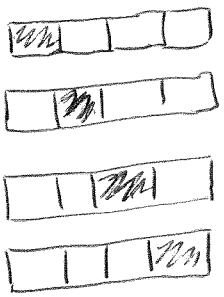
$|w_j^*|$ behave reasonably now

[How to choose λ?] Divide data into test & training sets



Can also use cross-validation

e.g. 4-fold:



» obtain $\langle E(\vec{w}_k^*)\rangle$ $k=1..4$ averaged over 4 runs

Probabilistic interpretation of curve fitting

Input values: $\vec{x} = (x_1 \dots x_N)$
Target values: $\vec{t} = (t_1 \dots t_N)$

Assume: gaus. noise distr'n

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

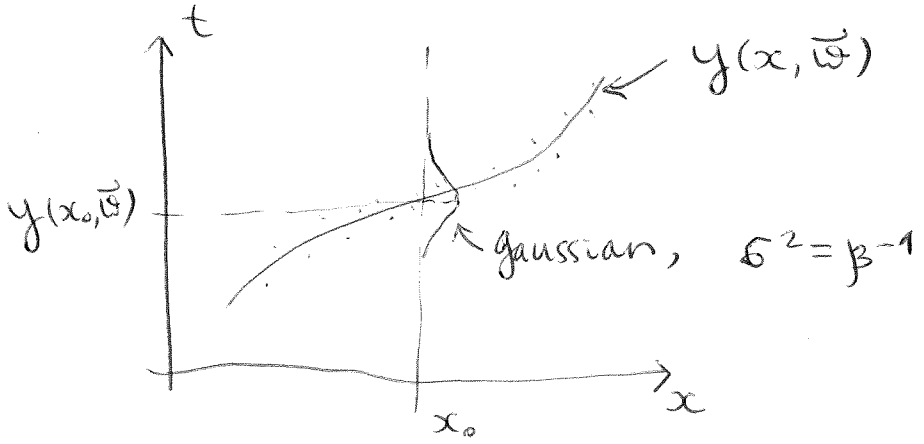
$$\mathcal{N}(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

β - precision prm

likelihood of the data:

$$P(\vec{t} \mid \vec{x}, \vec{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid y(x_n, \vec{w}), \beta^{-1})$$

↑ independent samples!



$$\log P = \underbrace{-\frac{\beta}{2} \sum_{n=1}^{N} (y(x_n, \vec{w}) - t_n)^2}_{\beta E(\vec{w})} + \underbrace{\frac{N}{2} \log\left(\frac{\beta}{2\pi}\right)}_{\text{indep. of } \vec{w}}$$

$$\frac{\partial}{\partial w_j} \log P = -\beta \frac{\partial}{\partial w_j} E(\vec{w})$$

Maximizing $\log P$ is the same as minimizing $E(\vec{w})$.
↳ find $\vec{w}_{ML}$

ML approach!

Further, $\frac{1}{\beta_{ML}} = \frac{1}{N}\sum_{n=1}^{N}(y(x_n, \vec{w}_{ML}) - t_n)^2$

Now can make predictions:

$$p(t|x, \vec{w}_{ML}, \beta_{ML}) = \mathcal{N}(t|y(x, \vec{w}_{ML}), \beta_{ML}^{-1}).$$

A bit more Bayesian: (model prms, data, likelihood, prior)

$$\left[p(\vec{w}|D) = \frac{p(D|\vec{w})\,p(\vec{w})}{p(D)} \right] \quad \text{(posterior, evidence)}$$

$$\left[p(D) = \int d\vec{w}\; p(D|\vec{w})\,p(\vec{w}) \right]$$

posterior ~ likelihood × prior [norm'n restored later]

Assume for the prior: $\sum_j w_j^2$

$$p(\vec{w}|\alpha) = \left(\frac{\alpha}{2\pi}\right)^{\frac{M+1}{2}} e^{-\frac{\alpha}{2}\vec{w}^T\vec{w}}$$

product of 1D gaussians centered on 0

α - precision prm

$$\log p(\vec{w}|\vec{x}, \vec{t}, \alpha, \beta) \sim \underbrace{\log p(t|\vec{x}, \vec{w}, \beta)}_{-\frac{\beta}{2}\sum_{n=1}^{N}(y(x_n,\vec{w}) - t_n)^2} + \underbrace{\log p(\vec{w}|\alpha)}_{-\frac{\alpha}{2}\vec{w}^T\vec{w}}$$

If we maximize the posterior prob. (instead of considering the whole distrib'n), we minimize $\frac{\beta}{2}\sum_{n=1}^{N}(\ldots)^2 + \frac{\alpha}{2}\vec{w}^T\vec{w}$

↳ same as regularization with $\lambda = \alpha/\beta$