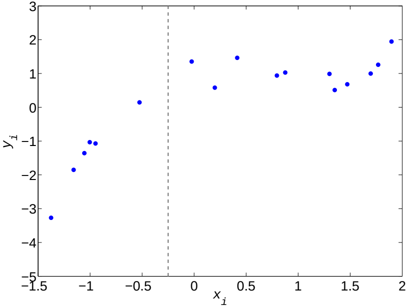
# Linear in the parameters regression

Carl Edward Rasmussen

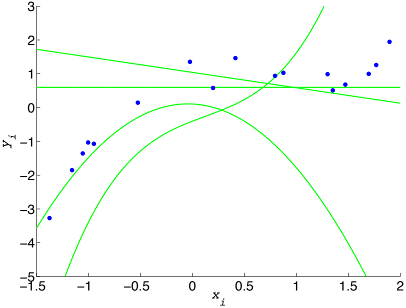
June 23rd, 2016

# How do we fit this dataset?




- Dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$ of N pairs of inputs $x_i$ and targets $y_i$.
  This data can for example be measurements in an experiment.
- Goal: predict target $y_*$ associated to any arbitrary input $x_*$.
  This is known a as a regression task in machine learning.
- Note: Here the inputs are scalars, we have a single input feature.
  Inputs to regression tasks are often vectors of multiple input features.

# Model of the data

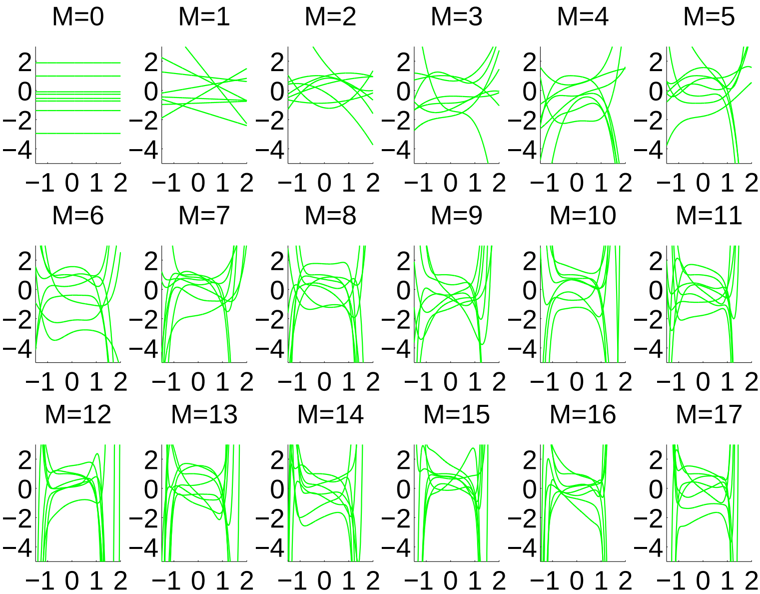



- In order to predict at a new $x_*$ we need to postulate a model of the data. We will estimate $y_*$ with $f(x_*)$.
- But what is $f(x)$? Example: a polynomial

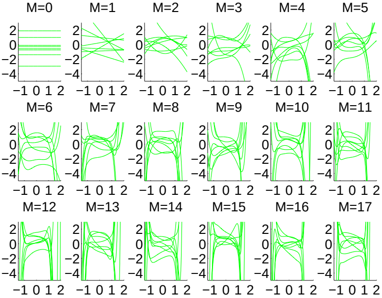
$$f_{\mathbf{w}}(x) \;=\; w_0 + w_1\, x + w_2\, x^2 + w_3\, x^3 + \ldots + w_M\, x^M$$

The $w_j$ are the weights of the polynomial, the parameters of the model.

# Model of the data. Example: polynomials of degree M

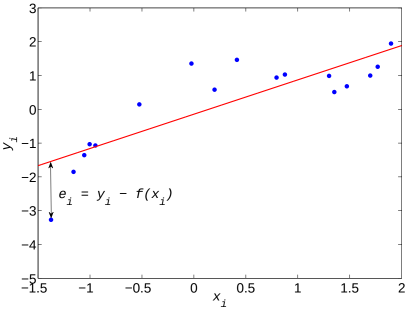


# Model structure and model parameters




- Should we choose a polynomial? model structure
- What degree should we choose for the polynomial? model structure
- For a given degree, how do we choose the weights? model parameters
- For now, let find the single "best" polynomial: degree and weights.

# Fitting model parameters: the least squares approach




- Idea: measure the quality of the fit to the training data.
- For each training point, measure the squared error $e_i^2 = (y_i - f(x_i))^2$.
- Find the parameters that minimise the sum of squared errors:

$$E(\mathbf{w}) \; = \; \sum_{i=1}^{N} e_i^2$$

$f_{\mathbf{w}}(x)$ is a function of the parameter vector $\mathbf{w} = [w_0, w_1, \ldots, w_M]^\top$.

# Least squares in detail. (1) Notation

Some notation: training targets $\mathbf{y}$, predictions $\mathbf{f}$ and errors $\mathbf{e}$.

- $\mathbf{y} = [y_1, \ldots, y_N]^\top$ is a vector that stacks the N training targets.
- $\mathbf{f} = [f_{\mathbf{w}}(x_1), \ldots, f_{\mathbf{w}}(x_N)]^\top$ stacks $f_{\mathbf{w}}(x)$ evaluated at the N training inputs.
- $\mathbf{e} = \mathbf{y} - \mathbf{f}$ is the vector of training prediction errors.

The sum of squared errors is therefore given by

$$E(\mathbf{w}) \;=\; \|\mathbf{e}\|^2 \;=\; \mathbf{e}^\top\mathbf{e} \;=\; (\mathbf{y}-\mathbf{f})^\top(\mathbf{y}-\mathbf{f})$$

More notation: weights $\mathbf{w}$, basis functions $\phi_j(x)$ and matrix $\mathbf{\Phi}$.

- $\mathbf{w} = [w_0, w_1, \ldots, w_M]^\top$ stacks the $M+1$ model weights.
- $\phi_j(x) = x^j$ is a basis function of our linear in the parameters model.

$$f_{\mathbf{w}}(x) \;=\; w_0\, 1 + w_1\, x + w_2\, x^2 + \ldots + w_M\, x^M \;=\; \sum_{j=0}^{M} w_j\, \phi_j(x)$$

- $\mathbf{\Phi}_{ij} = \phi_j(x_i)$ allows us to write $\mathbf{f} = \mathbf{\Phi}\, \mathbf{w}$.

# Least squares in detail. (2) Solution

A Gradient View. The sum of squared errors is a convex function of $\mathbf{w}$:

$$\mathsf{E}(\mathbf{w}) \;=\; (\mathbf{y}-\mathbf{f})^\top(\mathbf{y}-\mathbf{f}) \;=\; (\mathbf{y}-\boldsymbol{\Phi}\,\mathbf{w})^\top(\mathbf{y}-\boldsymbol{\Phi}\,\mathbf{w})$$

The gradient with respect to the weights is:

$$\frac{\partial \mathsf{E}(\mathbf{w})}{\partial \mathbf{w}} \;=\; 2\,\boldsymbol{\Phi}^\top(\mathbf{y}-\boldsymbol{\Phi}\,\mathbf{w}) \;=\; 2\,\boldsymbol{\Phi}^\top\,\mathbf{y} - 2\boldsymbol{\Phi}^\top\,\boldsymbol{\Phi}\,\mathbf{w}$$

The weight vector $\hat{\mathbf{w}}$ that sets the gradient to zero minimises $\mathsf{E}(\mathbf{w})$:

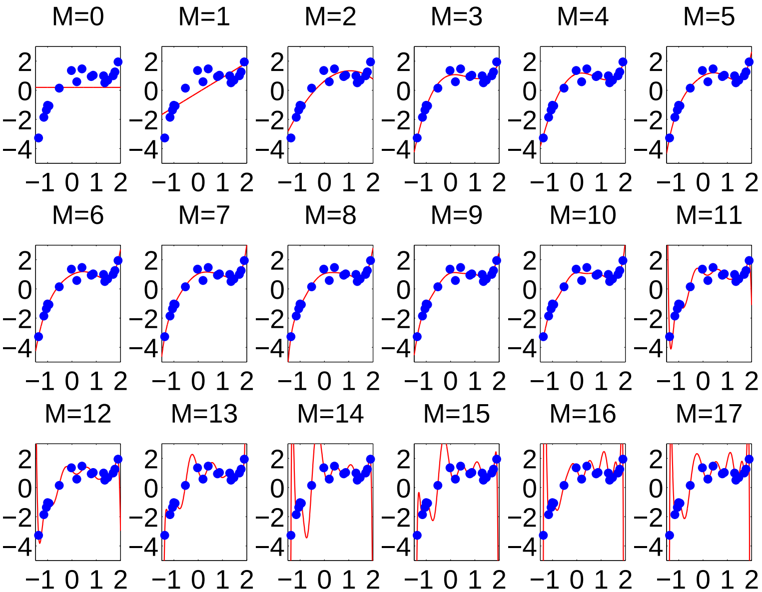
$$\boxed{\hat{\mathbf{w}} \;=\; (\boldsymbol{\Phi}^\top\,\boldsymbol{\Phi})^{-1}\,\boldsymbol{\Phi}^\top\,\mathbf{y}}$$

A Geometrical View. This is the matrix form of the Normal equations.

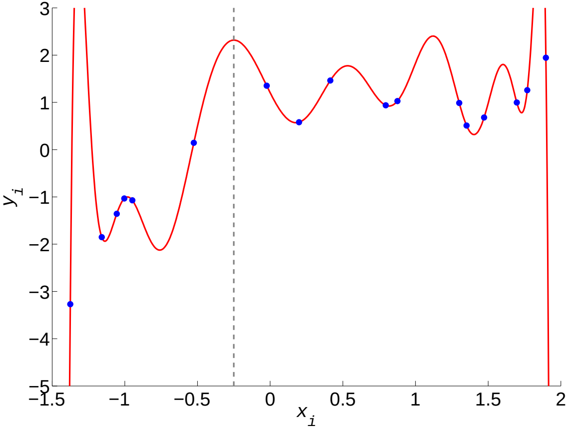
- The vector of training targets $\mathbf{y}$ lives in an N-dimensional vector space.
- The vector of training predictions $\mathbf{f}$ lives in the same space, but it is constrained to being generated by the $M+1$ columns of matrix $\boldsymbol{\Phi}$.
- The error vector $\mathbf{e}$ is minimal if it is orthogonal to all columns of $\boldsymbol{\Phi}$:

$$\boldsymbol{\Phi}^\top\,\mathbf{e} \;=\; 0 \quad\Longleftrightarrow\quad \boldsymbol{\Phi}^\top(\mathbf{y}-\boldsymbol{\Phi}\,\mathbf{w}) \;=\; 0$$

# Least squares fit for polynomials of degree 0 to 17

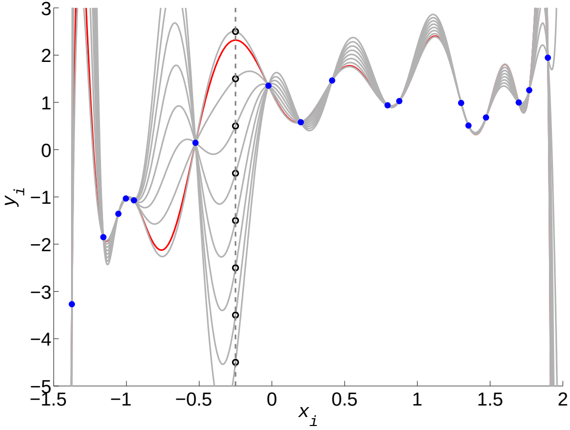


# Have we solved the problem?




- Ok, so have we solved the problem?
- What do we think $y_*$ is for $x_* = -0.25$? And for $x_* = 2$?
- If $M$ is large enough, we can find a model that fits the data

# Overfitting




- All the models in the figure are polynomials of degree 17 (18 weights).
- All perfectly fit the 17 training points, plus any desired $y_*$ at $x_* = -0.25$.
- We have not solved the problem. Key missing ingredient: **assumptions!**

# Some open questions

- Do we think that all models are equally probable... before we see any data?

  What does the probability of a model even mean?

- Do we need to choose a single "best" model or can we consider several?

  We need a framework to answer such questions.

- Perhaps our training targets are contaminated with noise. What to do?

  This question is a bit easier, we will start here.