POLITECNICO
MILANO 1863

Machine Learning Methods for Communication Networks and Systems

Francesco Musumeci

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB)

Politecnico di Milano, Milano, Italy

Part I – 1: Linear regression

Outline

- Introduction
- Univariate linear regression
- Multivariate linear regression
- Polynomial regression
- Computing parameters analytically

Outline

- Introduction
- Univariate linear regression
- Multivariate linear regression
- Polynomial regression
- Computing parameters analytically

Introduction

- **Regression** is part of *supervised* learning techniques
- Given the "ground truth" for a set of (labeled) examples $(\underline{x}^{(i)}, y^{(i)}), i=1,2,\ldots,m$ (*"training set"* with *m* examples)
- Predict (estimate) the output for new (unlabeled) examples $\underline{x}_{test}$ (i.e., find y_{test})
- General approach:
  - "guess" a model (hypothesis) for function $h(\underline{x})$
  - estimate parameters for function $h(\underline{x})$
  - perform prediction: $h(\underline{x}_{test})=y_{test}$
- N.B.: linear regression discussed here is a "parametric method", though "non-parametric" methods also exist (e.g., KNN)

Outline

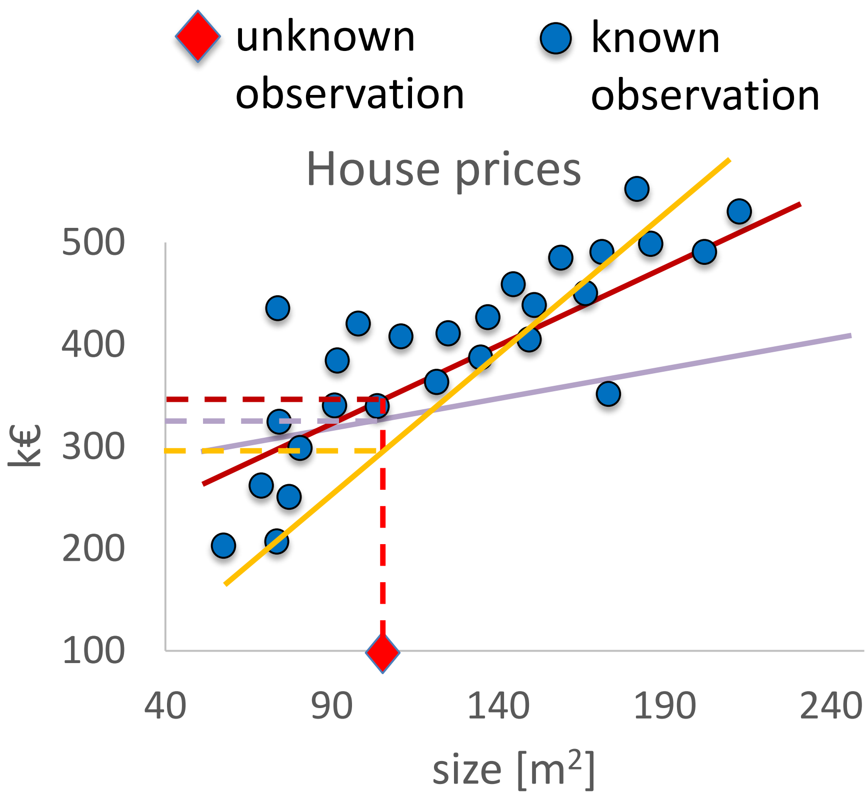
- Introduction
- Univariate linear regression
- Multivariate linear regression
- Polynomial regression
- Computing parameters analytically

Univariate linear regression
Hypothesis representation

- Simplest model we can "guess"
  - $h(\underline{x})$ is a ***linear*** function (linear)
  - $h(\underline{x})$ has only **one variable** (univariate), i.e., feature x_1

$$h(\underline{x}) = h(x_1) = \theta_0 x_0 + \theta_1 x_1 = \theta_0 + \theta_1 x_1$$

  - ○ θ_0 and θ_1 are the "weights"
  - ○ θ_0 is the "intercept" term (conventionally $x_0 = 1$)



Minimize a *Loss function*, e.g. the training mean-square error (MSE)

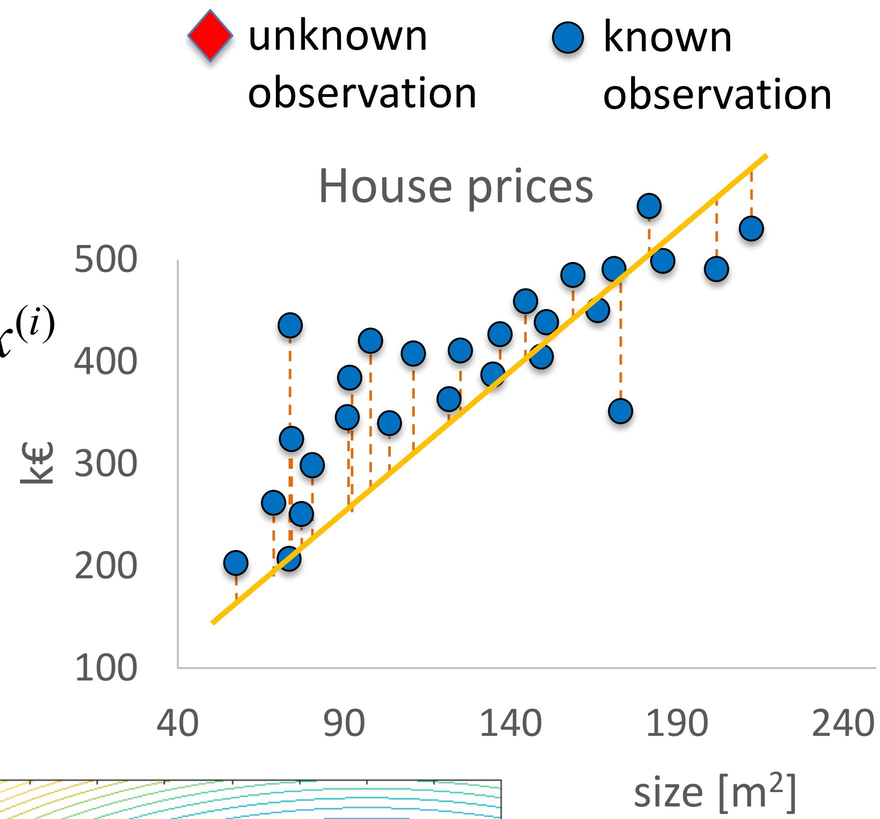
  - How to choose θ_0 and θ_1 ? $\min_{\theta_0,\theta_1}\left\{MSE = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2\right\}$

Univariate linear regression
Optimization objective

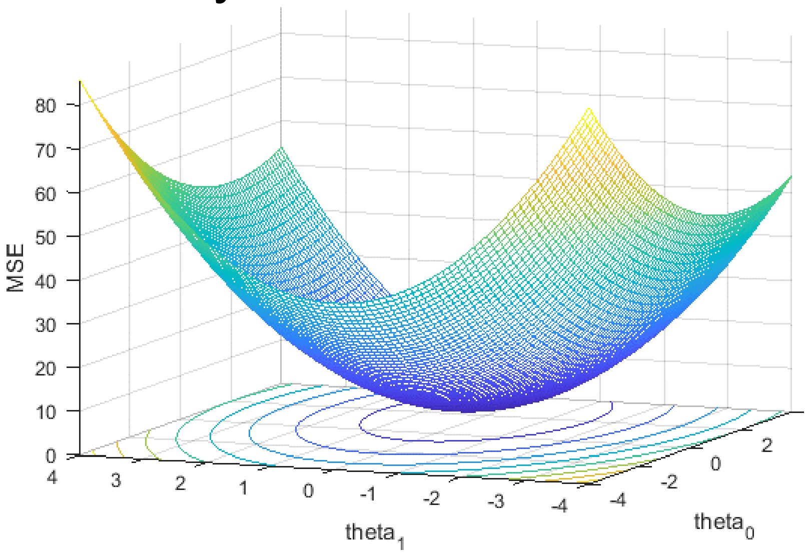
- Minimize the training MSE

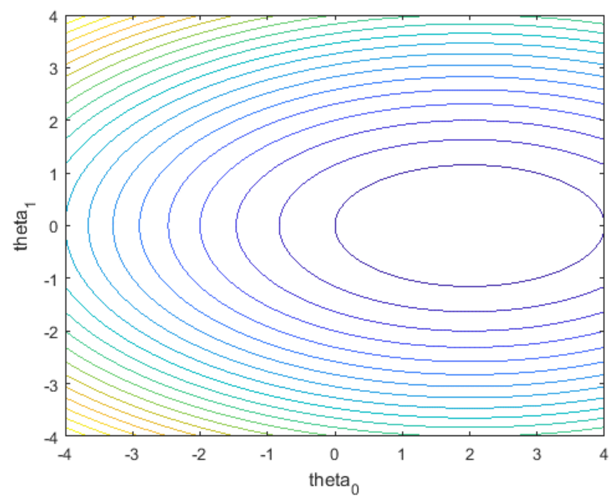
$$\min_{\theta_0,\theta_1} \{MSE(\theta_0,\theta_1)\} \qquad h(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

$$MSE(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$



- MSE for linear regression is always a convex function





Contour plot

Univariate linear regression
Parameter learning

$$MSE(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2$$

$$h(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

- Batch gradient descent algorithm
  - start with (random) initialization of θ_0 and θ_1
  - iteratively update θ_0 and θ_1 to reduce MSE

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} MSE(\theta_0, \theta_1)$$

Why the derivative?

$$= \theta_j - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \quad \mathrm{j} = 0,1$$

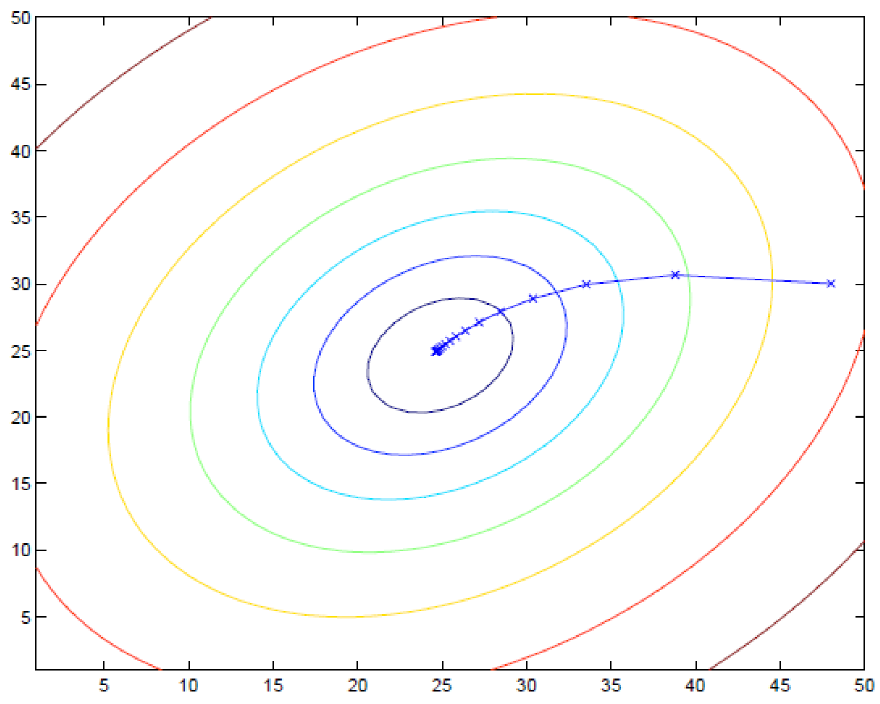
simultaneous update

    - α is the "learning rate"
  - STOP when convergence is reached
- Critical choice of α
  - small α: slow convergence
  - large α: divergence
  - Solution: plot MSE as a function of the iterations of gradient descent

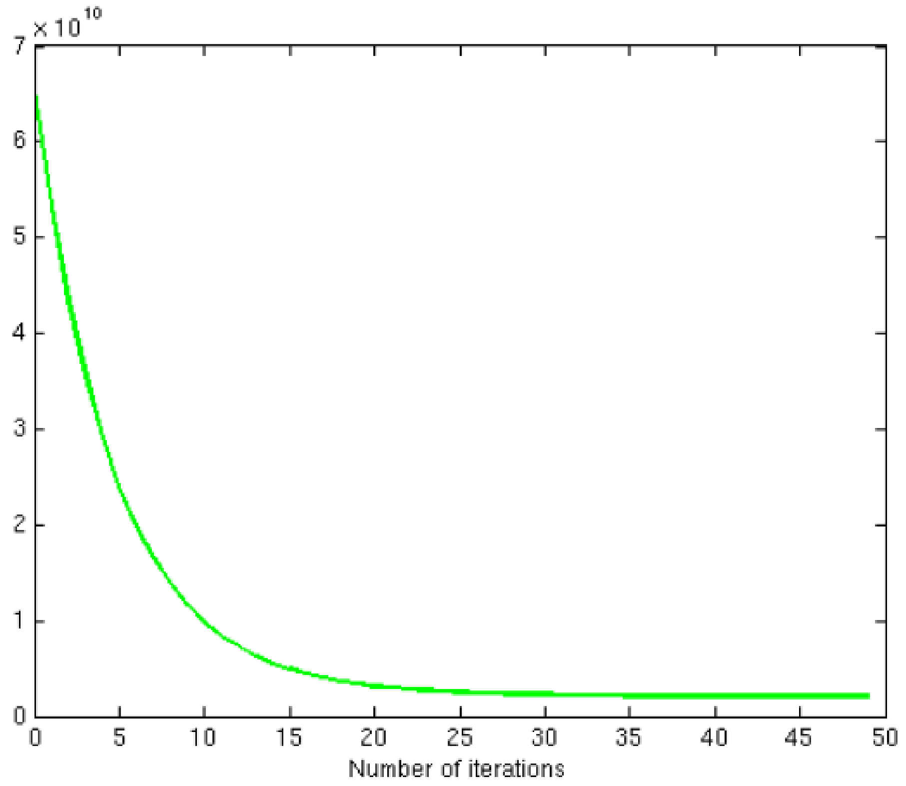
Univariate linear regression
Parameter learning

- Batch gradient descent algorithm

Parameters update



MSE minimization



Outline

- Introduction
- Univariate linear regression
- Multivariate linear regression
- Polynomial regression
- Computing parameters analytically

Multivariate linear regression

Extension of linear regression to multiple features

- *h*($\underline{x}$) has ***n*** **variables** (multivariate), i.e., we have a features **vector** $\underline{x}=(x_1, x_2, \ldots x_n)$
  - conventionally $x_0 = 1$ (θ_0 is the "intercept" term)

$$h(\underline{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

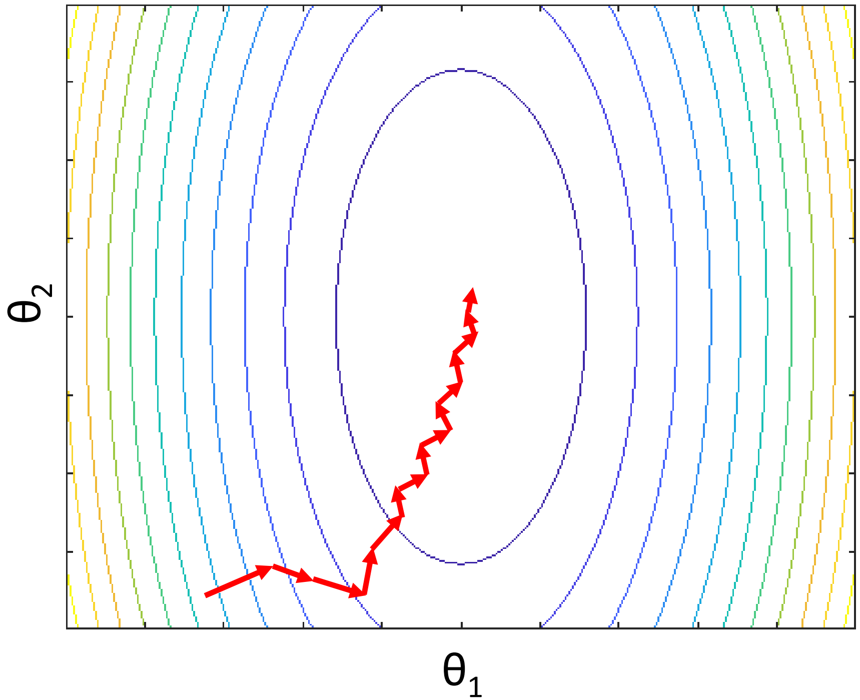
  - $x_j^{(i)}$: value of the j-th feature in the i-th example
  - $\underline{x}^{(i)}$: feature vector of i-th example
- Parameter learning can be performed via ***gradient descent***

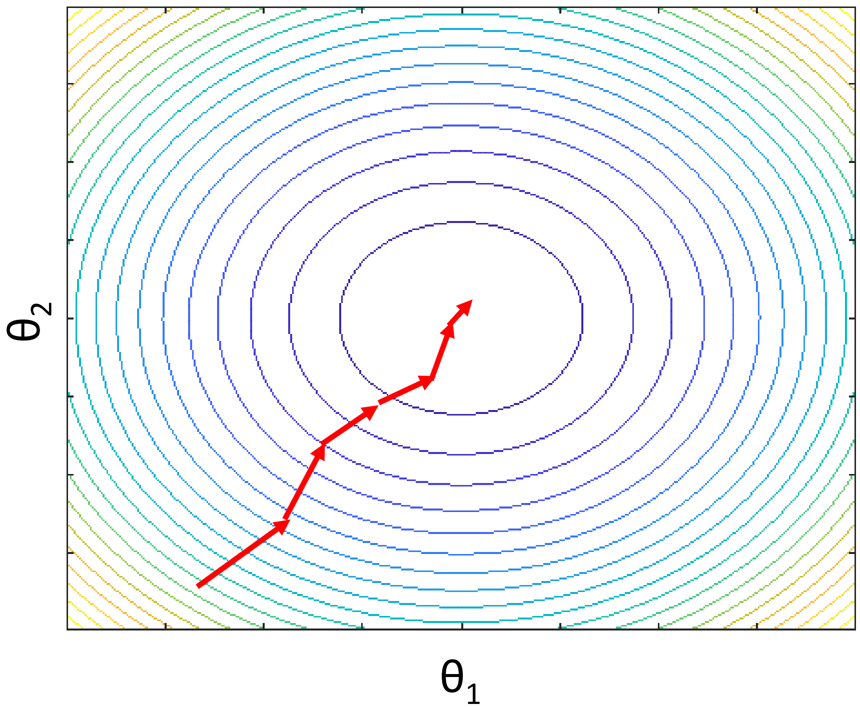
$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} MSE(\theta_0, \ldots, \theta_n) \quad \mathrm{j=0,\ldots,n}$$

simultaneous update

Gradient descent
Practical issues: features scaling

- If features take on values in (very) different ranges, convergence can be (very) slow
  - E.g.: x_1=[100;10000] Mb/s; x_2=[1;10] users
- Solution: features scaling, e.g.
  - $x_1 \rightarrow x_1/10000$
  - $x_2 \rightarrow x_2/10$





Gradient descent

Practical issues: features scaling & mean normalization

- General rule
  - features scaling: aims at providing features with values in similar ranges
  - mean normalization: aims at providing features with zero-mean values
- Mathematically:
  - $x_j^{(i)}$, $i=1,\dots,m$ (examples), $j=1,\dots,n$ (features)
  - create the "new" features $z_j^{(i)}$

mean normalization

$$z_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{s_j} \quad \mathrm{i = 1,\dots, m;\ j = 1,\dots, n}$$

features scaling

  - where

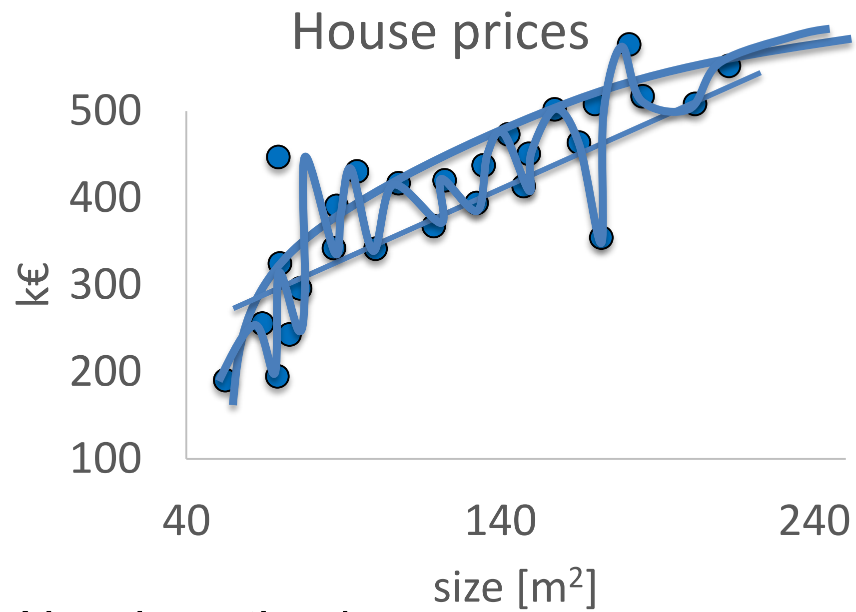
$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \qquad \mathrm{j = 1,\dots, n}$$

can be substituted by std dev. → $s_j = \max_i x_j^{(i)} - \min_i x_j^{(i)} \qquad \mathrm{j = 1,\dots, n}$

Outline

- Introduction
- Univariate linear regression
- Multivariate linear regression
- Polynomial regression
- Computing parameters analytically

Polynomial regression



- Why *linear* regression?
- Adding flexibility, i.e., increasing order of polynomials in *h(x)* can improve the fit
- E.g., starting w/ two features x_1, x_2, we can create new features by "manipulating" the original ones:
  - $x_3 = x_1^2$
  - $x_4 = x_2^2$
  - $x_5 = x_1 x_2$
  - $x_6 = x_1^3$
  - $x_7 = x_2^3$
  - $x_8 = x_1^2 x_2$
  - $x_9 = x_1 x_2^2$
  - $\ldots$

degree 2 (x_3–x_5)

degree 3 (x_6–x_9)

……

- New hypothesis:

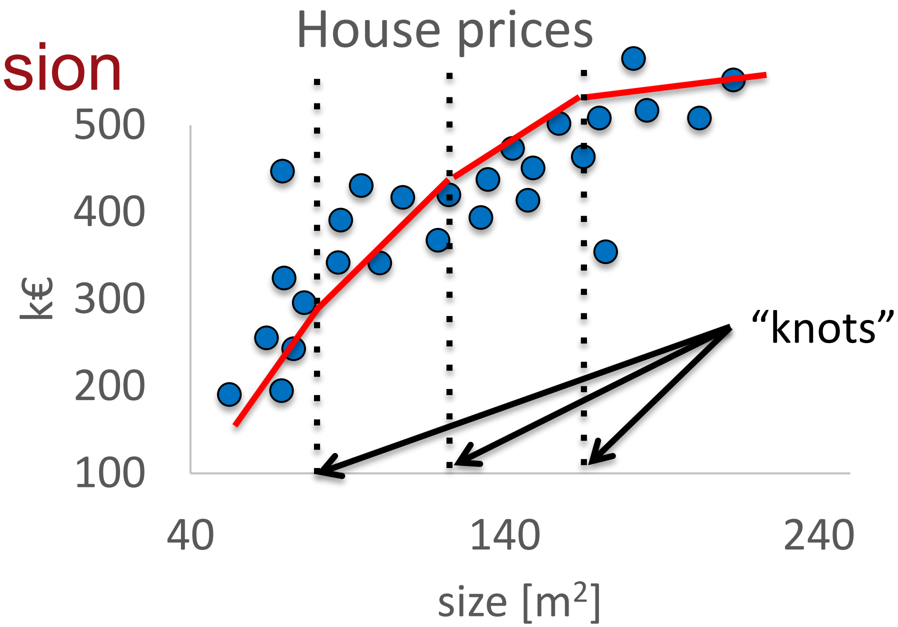
$$h(\underline{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

where added features are obtained combining 2 (or more) "original" features

- Parameters learning performed as in multivariate linear regression
- NB1: other functions can be used, e.g. *sqrt(x)*
- NB2: features scaling is even more important when using polynomial features

Polynomial regression

“Piecewise” polynomial regression



- Regression can be performed also dividing features space in “sectors”
- In each sector a different hypothesis (different parameters) is assumed
  - $h_1(x) = \theta_{01} + \theta_{11}\, x_1$
  - $h_2(x) = \theta_{02} + \theta_{12}\, x_1$
  - $h_3(x) = \theta_{03} + \theta_{13}\, x_1$
  - …
- Hypothesis in each sector can be linear, polynomial or anything else…
- With *k* knots and *d*-degree polynomial hypothesis in every sector
  - $(k+1)*(d+1)$ parameters to determine
  - Critical choice of *k* and *d*

- Further constraints can be added at “knots” to limit model flexibility and smooth the overal hypothesis:
  - Continuity
  - Continuous 1^{st}, 2^{nd} … derivatives
- Examples
  - Regression splines
  - Smoothing splines
  - …

Outline

- Introduction
- Univariate linear regression
- Multivariate linear regression
- Polynomial regression
- Computing parameters analytically

Computing parameters analytically

Normal equation

- To minimize MSE for the training set we could compute parameters $\theta_0, \theta_1, \ldots, \theta_n$ *analytically*
  - Set

$$\frac{\partial}{\partial \theta_j} MSE(\theta_0, \ldots, \theta_n) = 0 \quad \forall\ j = 0, \ldots, n$$

solve for $\theta_0, \theta_1, \ldots, \theta_n$

- In matrix form:
  - X [m x $(n+1)$]: matrix of features (m examples, n features, "+1" is used to consider the "intercept" term) for examples in the training set
  - y [m x 1]: vector of responses for examples in the training set
  - Θ [$(n+1)$ x 1]: matrix of parameters to be determined:

$$\Theta = (X^T X)^{-1} X^T y$$

- Why not to use *always* this closed-form formula?
  - Matrix inverse is slow for high-dimension
  - $(X^T X)$ can be non-invertible

$(X^T X)$ is a $(n+1)$ x $(n+1)$ square matrix

- Gradient descent works well even for large number of features (i.e., large n), BUT…
  - need to set α
  - do (several) iterations to find parameters
  - need to set a STOP condition