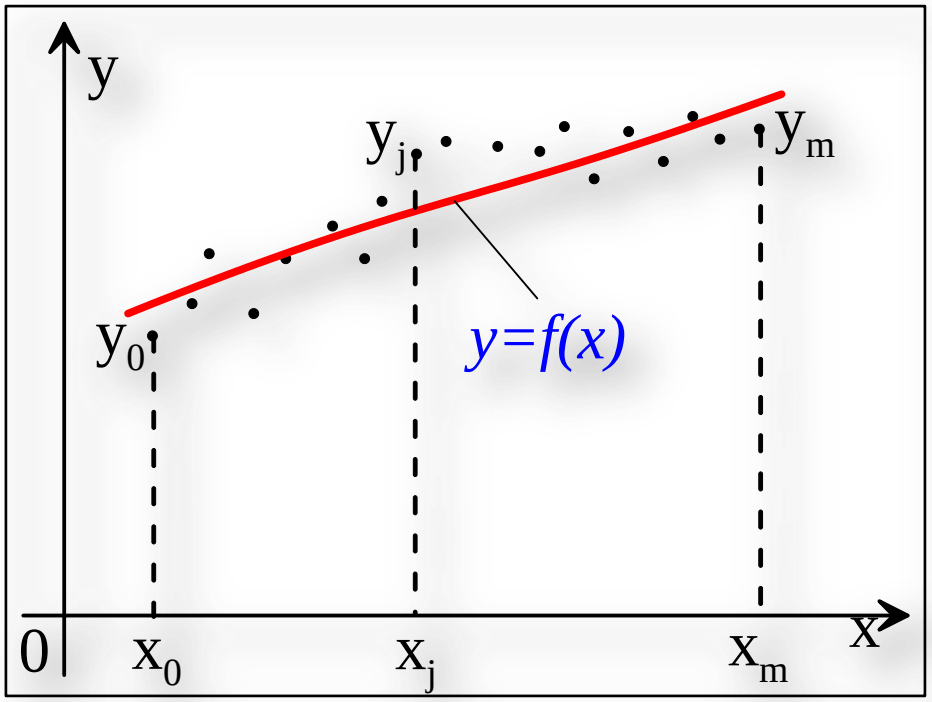
PROGRAM ROZWOJOWY
POLITECHNIKI WARSZAWSKIEJ

LECTURE 2
POLYNOMIAL APPROXIMATION

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

FORMULATION OF THE LEAST-SQUARES APPROXIMATION PROBLEM



Consider the given set of the node

$$\{(x_{k,}y_k), k = 0,..,m\}$$

Typically, the number m is large, so the standard interpolation by the global polynomial is not sensible. Such polynomial would most likely exhibit rapid oscillations and it would be also extremely sensitive to small (and inevitable) inaccuracies in the input data.

We will take a different approach. Our aim is to determine a function will capture "the general trend" (red line in the plot) presented by the given set, and such that its plot is – in a certain sense – close to all nodes.

To this aim, we choose the prescribed finite set of the basic functions (e.g., polynomials, trigonometric functions, others)

$$\{\varphi_k(x), k = 0,..,n\}$$

and then look for the function in the form of the sum

$$f(x) = a_0\varphi_0(x) + a_1\varphi_1(x) + ... + a_n\varphi_n(x) \equiv \sum_{j=0}^{n} a_j\varphi_j(x)$$

such that the following quantity is **minimal**

$$R = R(a_0, a_1, ..., a_n) = \sum_{k=0}^{m}[f(x_k) - y_k]^2 = \sum_{k=0}^{m}[\sum_{j=0}^{n} a_j\varphi_j(x_k) - y_k]^2 = \min$$

Such problem is called the **approximation in the sense of the least squares.**

Necessary conditions for the minimum of the function *R* are

$$\frac{\partial R}{\partial a_i} = 0 \ , \ i = 0,..,n$$

Then, the linear system for the unknown coefficients $\{a_j, \ j = 0,..,n\}$ is obtained as follows

$$\frac{\partial R}{\partial a_i} = 2\sum_{k=0}^{m} \varphi_i(x_k)\left[\sum_{j=0}^{n} a_j \varphi_j(x_k) - y_k\right] = 0 \quad , \ i = 0,1,..,n$$

$$\Downarrow$$

$$\sum_{j=0}^{n}[\sum_{k=0}^{m} \varphi_i(x_k)\varphi_j(x_k)]a_j = \sum_{k=0}^{m} y_k \varphi_i(x_k)$$

This system can be written is the matrix/vector form as follows ***Ma* = *z***, where

$$\boldsymbol{a} = [a_0, a_1,.., a_n]^T \quad , \qquad M_{ij} = \sum_{k=0}^{m} \varphi_i(x_k)\varphi_j(x_k) = M_{ji} \quad , \quad i, j = 0,1,..,n$$

$$z_i = \sum_{k=0}^{m} y_k \varphi_i(x_k) \quad , \quad i = 0,1,..,n$$

In particular, the basic functions can be chosen as the monomials

$$\varphi_0(x) \equiv 1,\ \varphi_1(x) = x, ..., \varphi_j(x) = x^j, ..., \varphi_n(x) = x^n$$

Then

$$M_{ij} = \sum_{k=0}^{m} x_k^{i+j} \quad , \quad i, j = 0,1,..,n \quad , \quad z_i = \sum_{k=0}^{m} y_k x_k^i \quad , \quad i = 0,1,..,n$$

Note that the matrix $\boldsymbol{M}$ can be expressed as

$$\boldsymbol{M} = \boldsymbol{W}^T \boldsymbol{W}$$

where the elements of the matrix $\boldsymbol{W}$ are

$$W_{kj} = x_k^j \quad , \quad k = 0,...,m \ , \ j = 0,...,m$$

Indeed, we have

$$(\boldsymbol{W}^T \boldsymbol{W})_{ij} = \sum_{k=0}^{m} (\boldsymbol{W}^T)_{ik} (\boldsymbol{W})_{kj} = \sum_{k=0}^{m} (\boldsymbol{W})_{ki} (\boldsymbol{W})_{kj} = \sum_{k=0}^{m} x_k^i x_k^j = (\boldsymbol{M})_{ij}$$

Also, the right-hand side vector $\boldsymbol{z}$ can be expressed as

$$\boldsymbol{z} = \boldsymbol{W}^T \boldsymbol{y} \quad , \quad \boldsymbol{y} = [y_0, y_1, ..., y_m]^T$$

This form of the obtained linear system is not accidental. In fact, the polynomial approximations problem can be viewed as the **over determined interpolation problem**.

We will take an algebraic rather that the analytical approach. Consider an over determined linear system in the form of

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \ , \ \dim(\boldsymbol{A}) = (m, n) \ , \ m > n$$

This means that the number of equations m is larger than the number of the unknowns n (the matrix $\boldsymbol{A}$ is rectangular). Usually, so the above system of equations is – there is no vector $\boldsymbol{x}$ in R^n such that all equations are **simultaneously** satisfied. Yet, there is a possibility to re-define the problem of determination of the vector $\boldsymbol{x}$ in such a way, that there **exists** a solution. Moreover, if the rank of the matrix $\boldsymbol{A}$ is equal n (i.e., it is the maximal possible) then the solution is **unique**!

The idea is simple: we will find the vector **x** which makes the residual vector

$$\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$$

as small (short) as possible. In other words, the Euclidean norm of the vector $\boldsymbol{r}$

$$\|r\|_2 := \sqrt{r_1^2 + \ldots + r_m^2}$$

must be **minimal**.

To this aim, we will refer to the concept of the **range of the matrix** $\boldsymbol{A}$. The range of $\boldsymbol{A}$ is defined as the following set of vectors (the linear subspace of R^m)

$$range(\boldsymbol{A}) := \{\boldsymbol{y} \in R^m : \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x},\ \boldsymbol{x} \in R^n\}$$

Next, we also define the **kernel** of the transpose matrix $\boldsymbol{A}^T$. The kernel of $\boldsymbol{A}^T$ is the following set of vectors (also the linear subspace of R^m)

$$\ker(\boldsymbol{A}^T) = \{\boldsymbol{y} \in R^m : \boldsymbol{A}^T\boldsymbol{y} = \boldsymbol{0} \in R^n\}$$

Now, we claim that these two subspaces in R^m are **orthogonal**, i.e.

$$range(\boldsymbol{A}) \perp \ker(\boldsymbol{A}^T) \Leftrightarrow \underset{\boldsymbol{v}\in range(\boldsymbol{A})}{\forall} \underset{\boldsymbol{w}\in \ker(\boldsymbol{A}^T)}{\forall} (\boldsymbol{v},\boldsymbol{w}) \equiv \sum_{j=1}^{m} v_j w_j = 0$$

The **proof** of this crucial fact goes as follows. First, we note that

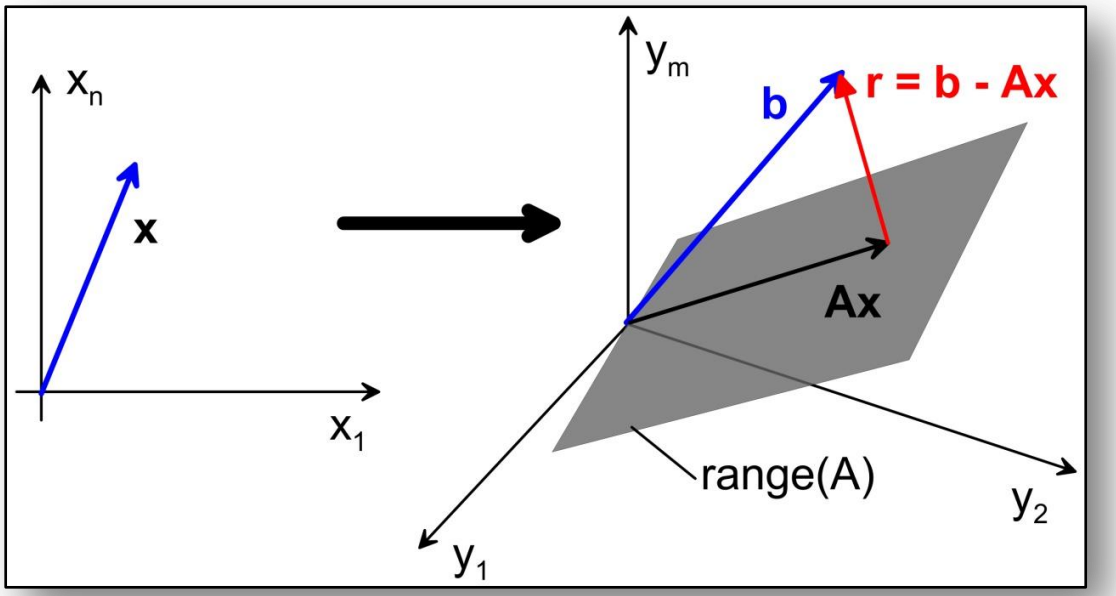
$$\boldsymbol{v} \in range(\boldsymbol{A}) \Leftrightarrow \underset{\boldsymbol{p}\in R^n}{\exists} \boldsymbol{v} = \boldsymbol{A}\boldsymbol{p}$$

Then the scalar product of the vectors $\boldsymbol{v}$ and $\boldsymbol{w}$ can be calculated

$$(\boldsymbol{v},\boldsymbol{w}) = (\boldsymbol{A}\boldsymbol{p},\boldsymbol{w}) = \sum_{j=1}^{m} (\boldsymbol{A}\boldsymbol{p})_j w_j = \sum_{j=1}^{m} (\sum_{k=1}^{n} a_{jk} p_k) w_j = \sum_{k=1}^{n} (\sum_{j=1}^{m} a_{jk} w_j) p_k =$$

$$= \sum_{k=1}^{n} (\sum_{j=1}^{m} a_{jk} w_j) p_k = \sum_{k=1}^{n} (\boldsymbol{A}^T\boldsymbol{w})_k p_k = (\boldsymbol{p}, \boldsymbol{A}^T\boldsymbol{w}) = (\boldsymbol{p},\boldsymbol{0}) = 0$$

The conclusion follows from the fact that the choice of $\boldsymbol{v}$ and $\boldsymbol{w}$ is arbitrary.



Note that (see the picture) that the length (we say – the norm) of the residual vector $\boldsymbol{r}$ is the smallest if this vector is perpendicular (we say – orthogonal) to the range($\boldsymbol{A}$). But it means that the minimal residual vector $\boldsymbol{r}$ belongs to the ker($\boldsymbol{A}^T$).

Thus, we have

$$\|\boldsymbol{r}\|_2 \equiv \sqrt{\sum_{k=1}^{m} r_k^2} = \min \quad \Leftrightarrow \quad \boldsymbol{r} \perp range(\boldsymbol{A}) \quad \Leftrightarrow \quad \boldsymbol{r} \in \ker(\boldsymbol{A}^T)$$

$$\Downarrow$$

$$\boldsymbol{A}^T(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}) = 0 \quad \Rightarrow \quad \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^T\boldsymbol{b}$$

In the context of the **polynomial approximation problem**, we can see that $\boldsymbol{A} = \boldsymbol{W}$**.** On the other hand, it is clear that the **over determined** linear system $\boldsymbol{Wa} = \boldsymbol{y}$ corresponds exactly to the "impossible" interpolation conditions

$$y_j = f(x_j) = \sum_{k=0}^{n} x_j^k a_k = \sum_{k=0}^{n} W_{jk} a_k \quad , \quad j = 0,1,...,m$$

The above mentioned method to solve the polynomial approximation problem is called the method of the **normal equations**.

In the remaining part of these notes, we present an alternative and computationally more effective method of the **orthogonal polynomials.**

METHOD OF THE ORTHOGONAL POLYNOMIALS

For the given sets of the nodes, one can define the inner (scalar) product of the functions as follows

$$\left\langle \varphi_i, \varphi_j \right\rangle := \sum_{k=0}^{m} \varphi_i(x_k)\varphi_j(x_k)$$

We say that the basic functions are (discretely) orthogonal on the given set of the nodes if and only if

$$\left\langle \varphi_i, \varphi_j \right\rangle = 0 \quad , \quad i \neq j$$

Using the orthogonal functions is advantageous because the corresponding matrix $\boldsymbol{M}$ of the system of the normal equations is purely **diagonal**

$$M_{ij} = \begin{cases} \sum_{k=0}^{m} \varphi_i^2(x_k) & , \; i = j \\ 0 & , \quad i \neq j \end{cases}$$

This means that the equations of this system are not coupled and they can be solved separately one by one. Indeed, one gets

$$\left(\sum_{k=0}^{m}\varphi_i^2(x_k)\right)a_i=\sum_{k=0}^{m}y_k\varphi_i(x_k)\quad,\quad i=0,..,n$$

$$\Downarrow$$

$$a_i=\frac{\sum_{k=0}^{m}y_k\varphi_i(x_k)}{\sum_{k=0}^{m}\varphi_i^2(x_k)}$$

The question remains how to construct efficiently orthogonal basic functions for a given set of the nodes. Since such options is particularly interesting, we will explain how to generate an **orthogonal set of polynomials**.

The computational procedure is recursive. It means that consecutive polynomials will be constructed by the use of those previously defined.

In order to initiate such procedure, two first polynomials must be prescribed. These polynomials are

$$q_0(x) \equiv 1 \quad , \quad q_1(x) = x - \alpha_1$$

The number α_1 is selected in order to assure that q_0 and q_1 are orthogonal:

$$\langle q_0, q_1 \rangle = 0 \Rightarrow \sum_{k=0}^{m} (x_k - \alpha_1) = 0 \Rightarrow \alpha_1 = \frac{1}{m+1} \sum_{k=0}^{m} x_k$$

Higher-order polynomials are obtained via two-point recurrence as follows

$$q_{j+1}(x) = x q_j(x) - \alpha_{j+1} q_j(x) - \beta_j q_{j-1}(x)$$

The coefficients α_{j+1} and β_j are chosen to assure that the polynomial q_{j+1} is orthogonal to the previously defined polynomials q_j and q_{j-1}:

$$\langle q_{j+1}, q_j \rangle = 0 \quad , \langle q_{j+1}, q_{j-1} \rangle = 0$$

These conditions lead to the following formulas:

$$\alpha_{j+1} = \frac{\langle xq_j, q_j \rangle}{\langle q_j, q_j \rangle} = \frac{\sum_{k=0}^{m} x_k q_j^2(x_k)}{\sum_{k=0}^{m} q_j^2(x_k)} \quad , \quad \beta_j = \frac{\langle xq_j, q_{j-1} \rangle}{\langle q_{j-1}, q_{j-1} \rangle} = \frac{\sum_{k=0}^{m} x_k q_{j-1}(x_k) q_j(x_k)}{\sum_{k=0}^{m} q_{j-1}^2(x_k)}$$

The crucial question arises: how come that $\langle q_{j+1}, q_i \rangle = 0 \quad , \quad i = 0,1,..,j-2$ **?!**

Surprisingly enough, this property holds "automatically"! To see this, we will calculate the inner product of the polynomial q_{j+1} and q_k for $k < j$-1:

$$\langle q_{j+1}, q_i \rangle = \langle xq_j, q_i \rangle - \alpha_{j+1} \underbrace{\langle q_j, q_i \rangle}_{0} - \beta_j \underbrace{\langle q_{j-1}, q_i \rangle}_{0} = \underbrace{\langle q_j, xq_i \rangle}_{xq_i(x) \text{ has order } i+1<j} = \sum_{p=0}^{i+1<j} \gamma_p \underbrace{\langle q_j, q_p \rangle}_{=0 \text{ for each } p<j} = 0$$

Final form of the approximating polynomial can be written as

$$f(x) = \sum_{j=0}^{n} \left[\sum_{k=0}^{m} y_k q_j(x_k) \Big/ \sum_{k=0}^{m} q_j^2(x_k) \right] q_j(x)$$

For efficient and stable method of evaluation of the polynomial *f(x)* see the **Clenshaw recurrence formula** (see, for instance, Numerical Recipes in C++, 3rd Ed., p. 222)