PROGRAM ROZWOJOWY
POLITECHNIKI WARSZAWSKIEJ

LECTURE 6
INTERPOLATION BY SPLINES

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

In this lecture we consider the problem of the **spline interpolation**. In contrast to the interpolation methods described in the **Lecture 1**, where a single polynomial is used for the whole interpolation interval, the spline technique consists in using different low-order polynomials for each subinterval between the interpolation nodes. The polynomials defined for neighboring subintervals are “glued” at the common nodes in such a way that sufficient regularity of the spline function is ensured. Here we concentrate on one but very important example of a spline function – the **cubic spline**.

Consider the set of the n interpolation nodes $\{(x_0, y_0), (x_1, y_1), ..., (x_{n-1}, y_{n-1})\}$, where $a = x_0 < x_1 < .. < x_{n-1} = b$.

The **cubic spline $C = C(x)$ is defined as follows**:

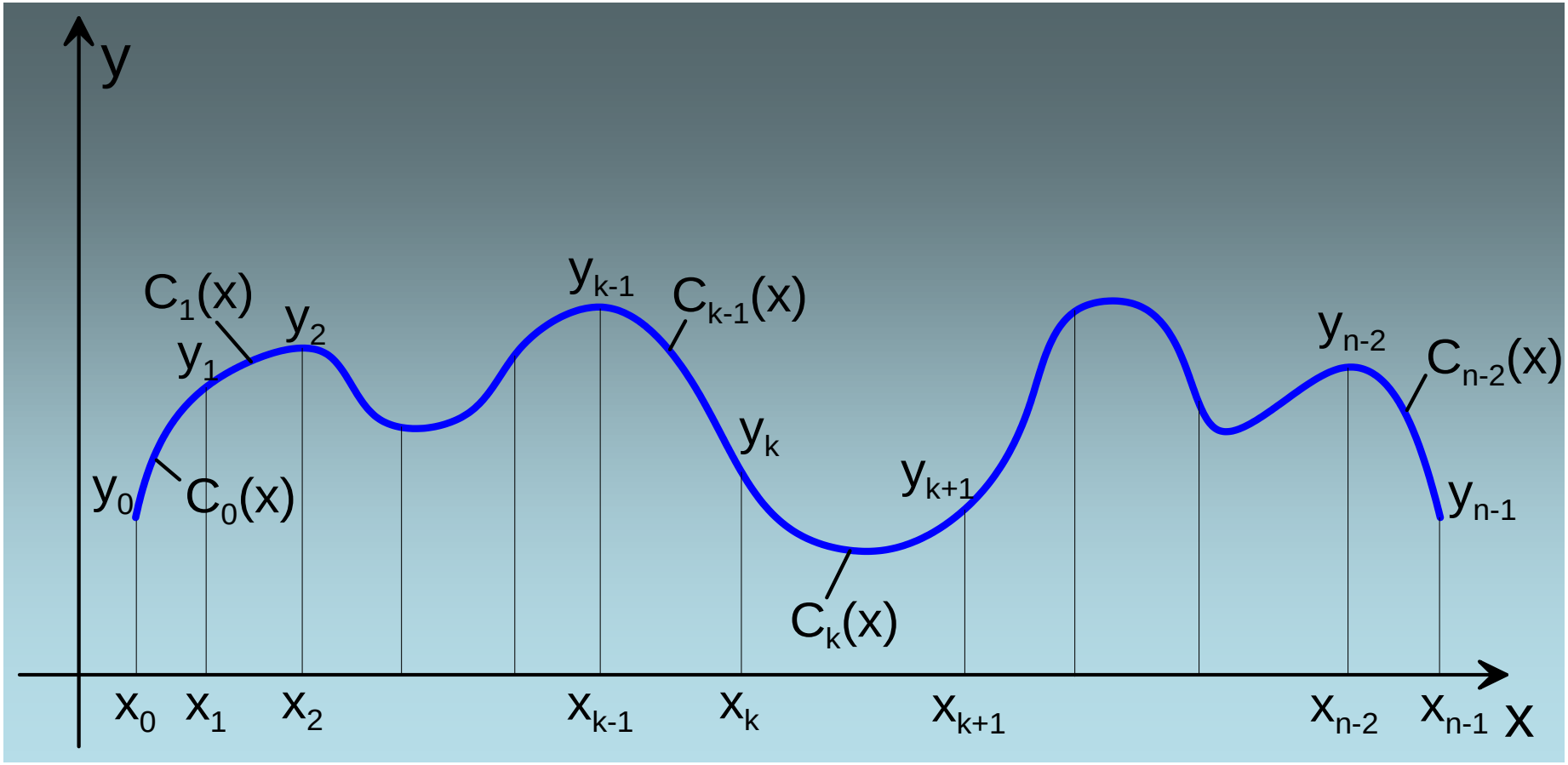
1. $C(x) \in C^2([a,b])$, i.e., it is continuous together with its first and second derivatives
2. $C_k(x) := C(x)\big|_{[x_k, x_{k+1}]} = a_{k,3}x^3 + a_{k,2}x^2 + a_{k,1}x + a_{k,0}$, $k = 0,..,n-2$, i.e. inside each subinterval it is defined as the 3rd –order (cubic) polynomial.

It follows that the local polynomials C_k, $k = 0$, must satisfy the interpolation conditions

$$C_k(x_k) = y_k \quad , \quad k = 0,..,n-1,$$

and the matching conditions

$$\begin{cases} C_{k-1}(x_k) = C_k(x_k) \\ C'_{k-1}(x_k) = C'_k(x_k) \\ C''_{k-1}(x_k) = C''_k(x_k) \end{cases} \quad k = 1,..,n-2$$



The cubic spline

Note that the overall number of these conditions is $4n - 6$. At the same time, the total number of unknown coefficients in the local polynomials is $4(n - 1)$. Thus, the problem contains two free parameters which have to be prescribed in order to find the unique spline.

Usually, the endpoint conditions are formulated: either for the first or the second derivative of the spline function:

$$(C'(x_0) = \alpha \text{ or } C''(x_0) = \beta) \text{ and } (C'(x_{n-1}) = \gamma \text{ or } C''(x_{n-1}) = \delta)$$

In principle, the **natural cubic spline** is such that

$$C''(x_0) = 0 \ , \ C''(x_{n-1}) = 0$$

The **natural spline** has a remarkable property. It turn out that among all functions which interpolate through the given nodes, the natural spline minimizes the following integral

$$\int_a^b \left[C''(x)\right]^2 dx = \min$$

More precisely, we have the following result:

Let $f \in C^2([x_{0,}x_{n-1}])$ be an arbitrary function. Assume that either $C''(x_0)=0$ and $C''(x_{n-1})=0$ or $C'(a)=f'(a)$ and $C'(b)=f'(b)$. Then

$$\int_a^b [C''(x)]^2\,dx \le \int_a^b [f''(x)]^2\,dx$$

Proof:

$$\int_a^b C''(x)[f''(x)-C''(x)]dx = C''(x)[f'(x)-C'(x)]\Big|_{x=a}^{x=b} - \int_a^b C'''(x)[f'(x)-C'(x)]dx =$$

$$= C''(b)[f'(b)-C'(b)] - C''(a)[f'(a)-C'(a)] - \int_a^b C'''(x)[f'(x)-C'(x)]dx =$$

$$= 0-0-\sum_{k=0}^{n-1}\int_{x_k}^{x_{k+1}} \underbrace{C_k'''(x)}_{6a_{k,3}=const}\,[f'(x)-C'(x)]dx = -6\sum_{k=0}^{n-1} a_{k,3}\int_{x_k}^{x_{k+1}} [f'(x)-C'(x)]dx =$$

$$= -6\sum_{k=0}^{n-1} a_{k,3}\,\underbrace{[f(x)-C(x)]\Big|_{x=x_k}^{x=x_{k+1}}}_{=0} = 0$$

because $C(x_k)=f(x_k)$, $k=0,1,..,n$

Thus, we have obtained the equality $\int_a^b C''(x) f''(x)dx = \int_a^b [C''(x)]^2 dx$.

Then

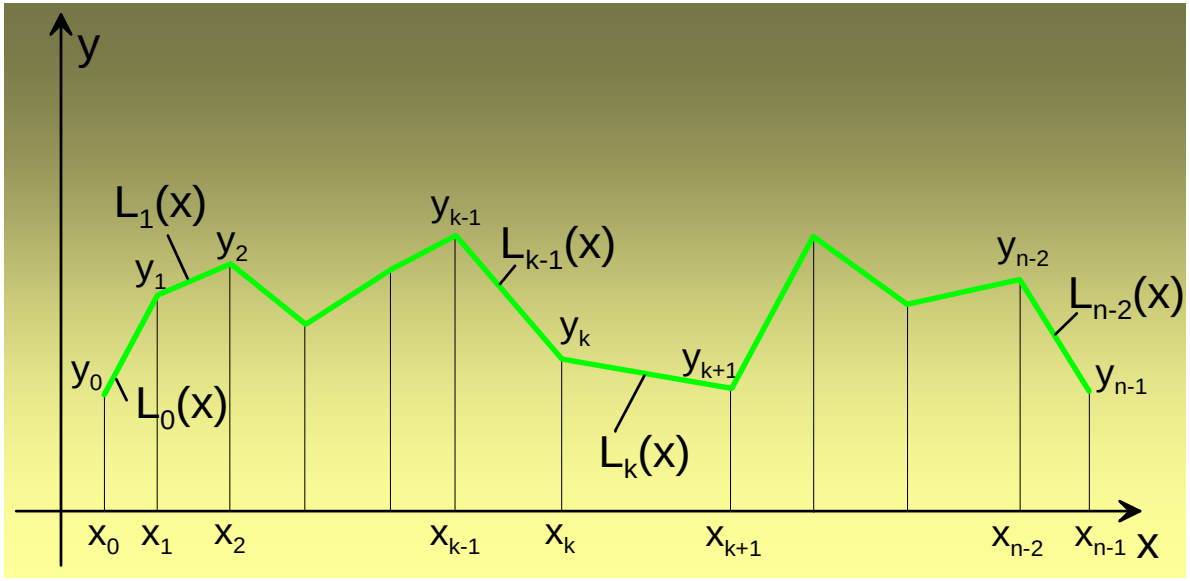
$$0 \le \int_a^b [f''(x) - C''(x)]^2 dx = \int_a^b [f''(x)]^2 dx - 2\underbrace{\int_a^b f''(x)C''(x)dx}_{= \int_a^b [C''(x)]^2 dx} + \int_a^b [C''(x)]^2 dx =$$

$$= \int_a^b [f''(x)]^2 dx - \int_a^b [C''(x)]^2 dx$$

and the conclusion follows immediately**. This ends the proof.**

The question remains how to construct the spline function. In principle, we could derive the set of $4n$-4 algebraic equations for the unknown coefficients of the local cubic polynomials using interpolation, matching and two additional conditions. We will end up with rather "awful" system of linear equation: the matrix of such system would not exhibit any convenient structure and no especially effective solution method could be applied. There exist, however, also the "smart" approach which we now describe.

Note that since the spline function is piece-cubic polynomial then its second derivative is piecewise-linear and continuous (see figure). It can be written as



$$C''(x)\Big|_{[x_k,x_{k+1}]} \equiv C_k''(x) =$$

$$= C_x''(x_k)\frac{x_{k+1}-x}{x_{k+1}-x_k} + C_x''(x_{k+1})\frac{x-x_k}{x_{k+1}-x_k}$$

or

$$C_k''(x) = m_k\frac{x_{k+1}-x}{h_k} + m_{k+1}\frac{x-x_k}{h_k}$$

where $m_k \equiv C''(x_k)$, $h_k = x_{k+1}-x_k$.

It this formula is integrated twice, we get

$$C_k(x) = \frac{m_k}{6h_k}(x_{k+1}-x)^3 + \frac{m_{k+1}}{6h_k}(x-x_k)^3 + p_k(x_{k+1}-x) + q_k(x-x_k)$$

where the symbols p_k and q_k denote the local integration constants.

These constant can be determined using the interpolation conditions

$$C_k(x_k) = y_k \quad , \; C_k(x_{k+1}) = y_{k+1}$$

One easily obtains

$$y_k = \tfrac{1}{6} m_k h_k^2 + p_k h_k \quad \Rightarrow \quad p_k = \frac{y_k}{h_k} - \tfrac{1}{6} m_k h_k$$

$$y_{k+1} = \tfrac{1}{6} m_{k+1} h_k^2 + q_k h_k \quad \Rightarrow \quad q_k = \frac{y_{k+1}}{h_k} - \tfrac{1}{6} m_{k+1} h_k$$

and the final formula for the local cubic polynomial reads

$$C_k(x) = \frac{m_k}{6h_k}(x_{k+1} - x)^3 + \frac{m_{k+1}}{6h_k}(x - x_k)^3 + \left(\frac{y_k}{h_k} - \tfrac{1}{6} m_k h_k \right)(x_{k+1} - x) +$$

$$+ \left(\frac{y_{k+1}}{h_k} - \tfrac{1}{6} m_{k+1} h_k \right)(x - x_k)$$

It remains to evaluate the **nodal values of the second derivative** $m_0, m_1, \ldots, m_{n-1}$.

Note that we haven't used the matching conditions for the first derivative yet. The local formula for this derivative is

$$C_k'(x) = -\frac{m_k}{2h_k}(x_{k+1} - x)^2 + \frac{m_{k+1}}{2h_k}(x - x_k)^2 + \frac{y_{k+1} - y_k}{h_k} - \tfrac{1}{6}(m_{k+1} - m_k)h_k$$

Consider the internal node x_k. Using the above formula one obtains

$$C_k'(x_k) = -\tfrac{1}{3}m_k h_k - \tfrac{1}{6}m_{k+1}h_k + \underbrace{\frac{y_{k+1} - y_k}{h_k}}_{d_k} = -\tfrac{1}{3}m_k h_k - \tfrac{1}{6}m_{k+1}h_k + d_k$$

Replacing k with k-1 yields immediately the formula for the derivative of $C_{k\text{-}1}$. Then insertion of x_k leads to the following result

$$C_{k-1}'(x_k) = \tfrac{1}{3}m_k h_{k-1} + \tfrac{1}{6}m_{k-1}h_{k-1} + \underbrace{\frac{y_k - y_{k-1}}{h_{k-1}}}_{d_{k-1}} = \tfrac{1}{3}m_k h_{k-1} + \tfrac{1}{6}m_{k-1}h_{k-1} + d_{k-1}$$

Finally, using the matching condition $C'_{k-1}(x_k) = C'_k(x_k)$ the following system of algebraic equations is obtained

$$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k \quad , \quad k = 1,2,..,n-2$$

where $u_k = 6(d_k - d_{k-1}) = 6\left(\frac{y_{k+1} - y_k}{h_k} - \frac{y_k - y_{k-1}}{h_{k-1}} \right).$

As we already know, two additional equations are needed to close the system. If we want to fix the endpoint values of the first derivative, then we will use the following equations

$$C'(x_0) = -\tfrac{1}{3}h_0 m_0 - \tfrac{1}{6}h_0 m_1 + d_0 = \alpha \quad \Rightarrow \quad 2h_0 m_0 + h_0 m_1 = 6(d_0 - \alpha)$$

$$C'(x_{n-1}) = \tfrac{1}{3}h_{n-2} m_{n-1} + \tfrac{1}{6}h_{n-2} m_{n-2} + d_{n-2} = \gamma \quad \Rightarrow \quad h_{n-2}m_{n-2} + 2h_{n-2}m_{n-1} = 6(\gamma - d_{n-2})$$

It we want to fix the endpoint values of the second derivative, we simply write $m_0 = \beta$ and/or $m_{n-1} = \delta$.

Summarizing, the **complete set of the linear equations for the nodal values of the second derivative of the cubic spline function** can be written as follows

$$\begin{cases} m_0 = \beta \quad or \quad 2h_0 m_0 + h_0 m_1 = 6(d_0 - \alpha) \quad , \quad k = 0 \\ h_{k-1} m_{k-1} + 2(h_{k-1} + h_k) m_k + h_k m_{k+1} = u_k \quad , \quad k = 1,..,n-2 \\ m_{n-1} = \delta \quad or \quad h_{n-2} m_{n-2} + 2h_{n-2} m_{n-1} = 6(\gamma - d_{n-2}) \quad , \quad k = n-1 \end{cases}$$

In the matrix-vector notation: $\boldsymbol{T}\,\boldsymbol{m} = \boldsymbol{r}$. Note that the matrix $\boldsymbol{T}$ has nice **3-diagonal structure**. The nonzero elements of $\boldsymbol{T}$ can be stored inside three vectors: ***a,b*** and ***c***.

$$\boldsymbol{T} = \begin{bmatrix} c_0 & -b_0 & 0 & 0 & 0 & 0 & 0 \\ -a & c_1 & -b_1 & 0 & 0 & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 & 0 \\ 0 & 0 & -a_k & c_k & -b_k & 0 & 0 \\ 0 & 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & -a_{n-2} & c_{n-2} & -b_{n-2} \\ 0 & 0 & 0 & 0 & 0 & -a_{n-1} & c_{n-1} \end{bmatrix}$$

The actual values of the matrix entries depend partly on the applied endpoint conditions:

$$\begin{cases} a_0 = 0 \quad (\textit{not used}) \\ a_k = -h_{k-1} \;\; , \;\; k = 1,..,n-2 \\ a_{n-1} = 0 \;\; \textit{or} \;\; a_{n-1} = -h_{n-2} \end{cases} \qquad \begin{cases} b_0 = 0 \quad \textit{or} \quad b_0 = -h_0 \\ b_k = -h_k \;\; , \;\; k = 1,..,n-2 \\ b_{n-1} = 0 \;\; (\textit{not used}) \end{cases}$$

$$\begin{cases} c_0 = 1 \quad \textit{or} \quad c_0 = 2h_0 \\ c_k = 2(h_{k-1} + h_k) \;\; , \;\; k = 1,..,n-2 \\ c_{n-1} = 1 \;\; \textit{or} \;\; c_{n-1} = 2h_{n-2} \end{cases}$$

In order to solve the obtained 3-diagonal system, the specially design variant of the Gauss Elimination can be used. This method is called the **Thomas algorithm**. We will derive this algorithm assuming that the system to solve has the following form

$$\begin{cases} \quad c_0 m_0 - b_0 m_1 = r_0 \\ -a_k m_{k-1} + c_k m_k - b_k m_{k+1} = r_k \quad , \quad k = 1,..,n-2 \\ -a_{n-1} m_{n-2} + c_{n-1} m_{n-1} = r_{n-1} \end{cases}$$

Thomas Algorithm

Consider two first equations of this system

$$\begin{cases} c_0 m_0 - b_0 m_1 = r_0 \\ -a_1 m_0 + c_1 m_1 - b_1 m_2 = r_1 \end{cases}$$

and assume that $c_0 \neq 0$. First, we eliminate m_0

$$\begin{cases} m_0 - \alpha_0 m_1 = \beta_0 \quad , \quad \alpha_0 = b_0 / c_0 \quad , \beta_0 = r_0 / c_0 \\ -a_1 m_0 + c_1 m_1 - b_1 m_2 = r_1 \end{cases}$$

and next we transform the second equation to the "standard" form

$$(c_1 - \alpha_0 a_1) m_1 - b_1 m_2 = r_1 + a_1 \beta_0 \quad / : (c_1 - \alpha_0 a_1)$$

$$\Downarrow$$

$$m_1 - \alpha_1 m_2 = \beta_1 \quad , \quad \alpha_1 = \frac{b_1}{c_1 - \alpha_0 a_1} \quad , \quad \beta_1 = \frac{r_1 + a_1 \beta_0}{c_1 - \alpha_0 a_1}$$

In the process, we have introduced two auxiliary numbers α_1 and β_1.

Note that the reduced system with the unknowns $m_1,\ldots,m_{n-1}$ has 3-diagonal structure and its first equation is in the form ready for further elimination. Thus, the process of elimination can be continued and after k steps we deal with the problem of elimination of the unknown m_k which proceeds as follows

$$\begin{cases} m_k - \alpha_k m_{k+1} = \beta_k & / \cdot a_{k+1} \\ -a_{k+1} m_k + c_{k+1} m_{k+1} - b_{k+1} m_{k+2} = r_{k+1} \end{cases}$$

$$(c_{k+1} - \alpha_k a_{k+1}) m_{k+1} - b_{k+1} m_{k+2} = r_{k+1} + \beta_k a_{k+1}$$

$$\Downarrow$$

$$m_{k+1} - \alpha_{k+1} m_{k+2} = \beta_{k+1} \ , \ \alpha_{k+1} = \frac{b_{k+1}}{c_{k+1} - \alpha_k a_{k+1}} \ , \ \beta_{k+1} = \frac{r_{k+1} + a_{k+1}\beta_k}{c_{k+1} - \alpha_k a_{k+1}}$$

Two more auxiliary number α_{k+1} and β_{k+1} are defined in the process.

Finally, after n-1 such steps we arrive at the end of the system. The last elimination step goes as follows

$$\begin{cases} m_{n-2} - \alpha_{n-2} m_{n-1} = \beta_{n-2} & / \cdot a_{n-1} \\ -a_{n-1} m_{n-2} + c_{n-1} m_{n-1} = r_{n-1} & \end{cases}$$

$$(c_{n-1} - \alpha_{n-2} a_{n-1}) m_{n-1} = r_{n-1} + \beta_{n-2} a_{n-1}$$

$$\Downarrow$$

$$m_{n-1} = \frac{r_{n-1} + \beta_{n-2} a_{n-1}}{c_{n-1} - \alpha_{n-2} a_{n-1}}$$

Note that since the last equation contains only two unknown the elimination process actually terminates and the last unknown $m_{n\text{-}1}$ can be effectively calculated. Note also that in the process of elimination, we arrived at the recurrent formula (see previous page) which can be now conveniently re-written in the form

$$m_k = \beta_k + \alpha_k m_{k+1} \ , \ k = n-2, n-3, ..., 1, 0$$

It means that all eliminated unknowns can be found one after another, in the reverse order.

The **Thomas algorithm** can be summarized as follows:

STAGE 1 (sweep - up)

$$\alpha_0 = b_0 / c_0 \; ; \qquad \beta_0 = r_0 / c_0 \; ;$$

for k = 0 : n - 3

$$\alpha_{k+1} = b_k / (c_{k+1} - \alpha_k a_{k+1}) ;$$

$$\beta_{k+1} = (r_{k+1} + \beta_k a_{k+1}) / (c_{k+1} - \alpha_k a_{k+1}) ;$$

end;

STAGE 2 (sweep - down)

$$m_{n-1} = (r_{n-1} + \beta_{n-2} a_{n-1}) / (c_{n-1} - \alpha_{n-2} a_{n-1}) ;$$

for k = n - 2 : 0 : -1

$$m_k = \beta_k + \alpha_k m_{k+1} ;$$

end;

Remark:

Since the **Thomas algorithm** is the variant of the **Gauss Elimination** and as such it can fail unless the matrix obeys some restriction (see **Lecture 7**). The sufficient conditions which guarantee that the calculation with the Thomas algorithm will end up successfully are:

1) $c_0 \neq 0, c_{n-1} \neq 0 \, , a_k \neq 0 \, , b_k \neq 0 \, , k = 1,..,n-2$

2) *conditions of diagonal dominance*

$$|c_k| \geq |a_k| + |b_k| \quad , \quad i = 1,..,n-2$$

$$|c_0| \geq |b_0| \quad , \quad |c_{n-1}| \geq |a_{n-1}|$$

At least one of the above inequalities must be strict!