

# Algorytmy numeryczne - projekt

Szymon Gosk, Damian Górski, Niuta Godlewska

November 23, 2020

# Contents

<b>1</b>	<b>Wstęp</b>	<b>3</b>
<b>2</b>	<b>Interpolacja wielomianowa : Przyrost naturalny</b>	<b>3</b>
2.1	Opis metod obliczeniowych . . . . .	3
2.2	Opis implementacji algorytmu . . . . .	8
2.3	Struktury danych i struktura programu . . . . .	9
2.4	Program . . . . .	9
2.5	IO (wejście-wyjście) . . . . .	11
<b>3</b>	<b>Projekt nr 2</b>	<b>11</b>
<b>4</b>	<b>Projekt nr 3</b>	<b>11</b>

# 1 Wstęp

Niniejszy dokument stanowi dokumentację wszystkich części projektu przedmiotu *Algorytmy numeryczne*. Kolejne części projektu będą opisane w kolejnych sekcjach, a kolejne elementy poszczególnych części - w podsekcjach.

Dokument ten jest również podsumowaniem pracy członków zespołu **1.1**, w którego skład wchodzi **Szymon Gosk, Damian Górski i Niuta Godlewska**. Członkowie zespołu wspólnie oświadczają, iż projekt był realizowany przez każdego z nich w równym stopniu zaangażowania.

**Kod wszystkich algorytmów** można znaleźć na repozytorium projektu:

<https://github.com/Szymon-Gosk/szymon-gosk.github.io>

Kod umieszczony w tym dokumencie stałby się nieczytelny, tak więc - z uwagi na wygodę czytającego - zachęcamy do zapoznania się z programem na stronie zamieszczonej powyżej.

Zaimplementowany projekt można znaleźć pod adresem:

<https://szymon-gosk.github.io/>

## 2 Interpolacja wielomianowa : Przyrost naturalny

### 2.1 Opis metod obliczeniowych

**Zadanie:** Program, który oszacuje przyrost naturalny na świecie w przyszłości. Węzły mają przedstawiać przyrost naturalny zmieniający się w czasie.

Metoda numeryczna użyta w tym zadaniu to **metoda newtona**. Początkowo posiadamy zbiór  $n + 1$  danych:

$$\{(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)\}$$

Metoda polega na interpolacji danych w wielomian w postaci:

$$P(x) = \sum_{k=0}^n \left( b_k \prod_{i=0}^{k-1} (x - x_i) \right)$$

która pozwala oszacować wartości dla argumentów spoza zbioru danych.

Współczynniki  $b_k$  można uzyskać korzystając z poniższego wzoru:

$$b_k = \left( \frac{y_k - \sum_{i=0}^{k-1} \left( b_i \prod_{j=0}^{i-1} (x_k - x_j) \right)}{\prod_{i=0}^{k-1} (x_k - x_i)} \right)$$

**Przykład:** Dany jest następujący zbiór danych:

$$\{(2005, 6503), (2010, 6894), (2015, 7256)\}$$

Zbiór współczynników dla tego zbioru to:

$$\begin{aligned} b_0 &= 6503 \\ b_1 &= 78.2 \\ b_2 &= -0.58 \end{aligned}$$

przez co uzyskujemy wielomian w postaci:

$$P(x) = 6503 + 78.2(x - 2005) - 0.58(x - 2005)(x - 2010)$$

lub po przekształceniach:

$$P(x) = -0.58x^2 + 2406.9x - 2487717$$

Korzystając z tego wielomianu można przybliżyć wynik dla dowolnej wartości  $x$ . Dla  $x = 2020$  podstawiamy wartość:

$$P(2020) = -0.58 \cdot (2020)^2 + 2406.9 \cdot 2020 - 2487717 = 7589$$

Kolejnym krokiem jest metoda ogólna przekształcania wielomianu w postaci Newtona w postać ogólną:

$$\sum_{k=0}^n \left( b_k \prod_{i=0}^{k-1} (x - x_i) \right) = \sum_{k=0}^n (a_k x^k)$$

Obecnie, wielomian ma postać:

$$b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + \dots + b_n(x - x_0)\dots(x - x_{n-1})$$

Pierwszym krokiem będzie wymnożenie kolejnych iloczynów  $(x - x_i)$  dla danego współczynnika  $b_k$ . W tym celu zdefiniujemy macierz  $(n + 1) \times (n + 1)$ :

$$M = \begin{pmatrix} \lambda_{(0,0)} & \lambda_{(0,1)} & \cdots & \lambda_{(0,j)} & \cdots & \lambda_{(0,n)} \\ \lambda_{(1,0)} & \lambda_{(1,1)} & \cdots & \lambda_{(1,j)} & \cdots & \lambda_{(1,n)} \\ \vdots & \vdots & & \vdots & & \vdots \\ \lambda_{(i,0)} & \lambda_{(i,1)} & \cdots & \lambda_{(i,j)} & \cdots & \lambda_{(i,n)} \\ \vdots & \vdots & & \vdots & & \vdots \\ \lambda_{(n,0)} & \lambda_{(n,1)} & \cdots & \lambda_{(n,j)} & \cdots & \lambda_{(n,n)} \end{pmatrix}$$

Zamysł elementów  $\lambda_{(i,j)}$  jest następujący - zapiszmy wielomian w postaci opisanej wyżej:

$$b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + \dots + b_n(x - x_0)\dots(x - x_{n-1}) = b_0\lambda_{(0,0)} + b_1(\lambda_{(1,1)}x + \lambda_{(0,1)}) + \dots + b_n(\lambda_{(n,n)}x^n + \lambda_{(n-1,1)}x^{n-1} + \dots + \lambda_{(0,n)}x^0)$$

Innymi słowy, są to iloczyny kolejnych podwielomianów Newtona  $\prod_{i=0}^{k-1} (x - x_i)$ .

Konsekwencją tego jest fakt, iż:

$$\forall_{j < i} \lambda_{(i,j)} = 0$$

co pozwala nam zapisać macierz  $M$  jako:

$$M = \begin{pmatrix} \lambda_{(0,0)} & \lambda_{(0,1)} & \cdots & \lambda_{(0,j)} & \cdots & \lambda_{(0,n)} \\ 0 & \lambda_{(1,1)} & \cdots & \lambda_{(1,j)} & \cdots & \lambda_{(1,n)} \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & \lambda_{(i,j)} & \cdots & \lambda_{(i,n)} \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & \lambda_{(n,n)} \end{pmatrix}$$

Dodatkowo, warto zauważyć, że:

$$\forall_{i=j} \lambda_{(i,j)} = 1$$

Do uzyskania postaci ogólnej wielomianu, należy dodać do siebie współczynniki  $\lambda$  odpowiadające danym potęgom, pomnożone przez odpowiednie współczynniki  $b$ . Wykorzystamy do tego układ liniowy:

$$\begin{pmatrix} \lambda_{(0,0)} & \lambda_{(0,1)} & \cdots & \lambda_{(0,j)} & \cdots & \lambda_{(0,n)} \\ 0 & \lambda_{(1,1)} & \cdots & \lambda_{(1,j)} & \cdots & \lambda_{(1,n)} \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & \lambda_{(i,j)} & \cdots & \lambda_{(i,n)} \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & \lambda_{(n,n)} \end{pmatrix} \cdot \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_i \\ \vdots \\ a_n \end{pmatrix}$$

Rozwiązaniem tego układu dla poszczególnych  $a_i$  jest:

$$a_i = \sum_{j=0}^i b_j \lambda_{(i,j)}$$

Do wyznaczenia wzoru na współczynniki  $\lambda$  posłużymy się kombinacjami. Niech zbiór  $X$  będzie zbiorem współczynników  $x_i$ , tzn.  $X = \{x_0, x_1, \dots, x_n\}$ . Dodatkowo niech  $X_i$  oznacza zbiór pierwszych  $i$  współczynników zbioru  $X$ :  $X_i = \{x_1, x_2, \dots, x_i\}$

Niech  $C_j^i$  będzie zbiorem kombinacji o długości  $j$  ze zbioru  $X_i$ :

$$C_j^i = \binom{X_i}{j}$$

Moc zbioru  $C_j^i$  wyraża się przez  $|C_j^i| = \binom{i}{j}$ . Poniżej przedstawimy kilka kluczowych własności tego zbioru.

Zbiór  $C_j^i$  istnieje dla negatywnych wartości - przesłanka mówiąca za tym to fakt, że funkcja  $\binom{i}{j} = \frac{\Gamma(i+1)}{\Gamma(j+1)\Gamma(i-j+1)}$  jest zdefiniowana dla ujemnych wartości.

Dodatkowo  $\forall_{n \in \mathbb{N}} \forall_{\alpha \in \mathbb{Z}^-} C_\alpha^n = \emptyset$  and  $|C_\alpha^n| = 0$

Zdefiniowawszy powyższe rzeczy, lambdę możemy otrzymać ze wzoru:

$$\lambda_{(i,j)} = (-1)^{(i+j)} \cdot \left( \sum_{K \in C_{j-i}^j} \prod_{x_k \in K} (x_k) \right)$$

co daje nam wzór ogólny na  $a_i$ :

$$a_i = \sum_{j=0}^i \left( (-1)^{(i+j)} \cdot b_j \left( \sum_{K \in C_{j-i}^j} \prod_{x_k \in K} (x_k) \right) \right)$$

Teraz możemy zapisać wzór wielomianu w postaci ogólnej:

$$P(x) = \sum_{i=0}^n \left( x^i \sum_{j=0}^i \left( (-1)^{(i+j)} \cdot b_j \left( \sum_{K \in C_{j-i}^j} \prod_{x_k \in K} (x_k) \right) \right) \right)$$

**Przykład:** Posłużymy się wielomian newtona z przykładu pierwszego:

$$P(x) = 6503 + 78.2 \cdot (x - 2005) - 0.58 \cdot (x - 2005)(x - 2010)$$

Najpierw zapiszemy macierz współczynników lambda dla tego przykładu:

$$\begin{pmatrix} \lambda_{(0,0)} & \lambda_{(0,1)} & \lambda_{(0,2)} \\ 0 & \lambda_{(1,1)} & \lambda_{(1,2)} \\ 0 & 0 & \lambda_{(2,2)} \end{pmatrix}$$

Następnie, posługując się wcześniej zdefiniowanym zbiorem  $C_j^i$ , obliczamy:

$$C_0^0 = C_0^1 = C_0^2 = \{\emptyset\}$$

$$C_1^1 = \{\{x_0\}\}$$

$$C_2^2 = \{\{x_0, x_1\}\}$$

$$C_1^2 = \{\{x_0\}, \{x_1\}\}$$

Co pozwala obliczyć nam współczynniki lambda:

$$\lambda_{(0,0)} = \lambda_{(2,2)} = \lambda_{(1,1)} = \sum_{K \in C_0^1} \prod_{x_k \in K} (x_k) = \sum_{K \in \{\emptyset\}} \prod_{x_k \in K} (x_k) = \prod_{x_k \in \emptyset} (x_k) = 1$$

$$\lambda_{(0,1)} = - \sum_{K \in C_1^1} \prod_{x_k \in K} (x_k) = - \sum_{K \in \{\{x_0\}\}} \prod_{x_k \in K} (x_k) = - \prod_{x_k \in \{x_0\}} x_k = -x_0 = -2005$$

$$\lambda_{(0,2)} = \sum_{K \in C_2^2} \prod_{x_k \in K} (x_k) = \sum_{K \in \{\{x_0, x_1\}\}} \prod_{x_k \in K} (x_k) = \prod_{x_k \in \{x_0, x_1\}} x_k = x_0 x_1 = 4030050$$

$$\lambda_{(1,2)} = - \sum_{K \in C_2^3} \prod_{x_k \in K} (x_k) = - \sum_{K \in \{\{x_0\}, \{x_1\}\}} \prod_{x_k \in K} (x_k) = - \left( \prod_{x_k \in \{x_0\}} x_k + \prod_{x_k \in \{x_1\}} x_k \right) = -(x_0 + x_1) = -4015$$

Zapisujemy równanie liniowe z wyliczonymi współczynnikami  $\lambda$ :

$$\begin{pmatrix} 1 & -2005 & 4030050 \\ 0 & 1 & -4015 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 6503 \\ 78.2 \\ -0.58 \end{pmatrix} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$$

Z czego otrzymujemy:

$$a_0 = 6503 - 2005 \cdot 78.2 - 0.58 \cdot 4030050 = -2487717$$

$$a_1 = 78.2 + 4015 \cdot 0.58 = 2406.9$$

$$a_2 = -0.58$$

A następnie podstawiając otrzymujemy poprawny wielomian w postaci ogólnej:

$$P(x) = -0.58x^2 + 2406.9x - 2487717$$

## 2.2 Opis implementacji algorytmu

Implementacja składowych całościowego algorytmu jest prosta z uwagi na matematyczny zapis obliczeń. Każda suma lub iloczyn, jest równoważny pętli w programie.

Pierwszym elementem algorytmu jest wyliczenie współczynników  $b$ . Program oblicza je zgodnie z podanym wzorem, a do liczenia każdej sumy lub iloczynu wykorzystuje pętlę.

Po obliczeniu współczynników  $b$ , program tworzy *String*, w którym zawiera dane współczynniki, wartości  $x_i$  i zwraca *String* zawierający postać Newtona wielomianu konkatenując współczynniki z nazwiasami i wartościami  $x_i$ .

Korzystając z powstałego wielomianu program pobiera argument wpisany przez użytkownika i wylicza wartość wielomianu dla tego argumentu.

Z racji na skomplikowanie algorytmu przekształcania postaci Newtona w postać ogólną, implementacja została pominięta, a metoda zostawiona w dokumencie



jako ciekawostka.

## 2.3 Struktury danych i struktura programu

Program wykorzystuje proste listy, jeden z podstawowych typów danych w języku *Javascript*.

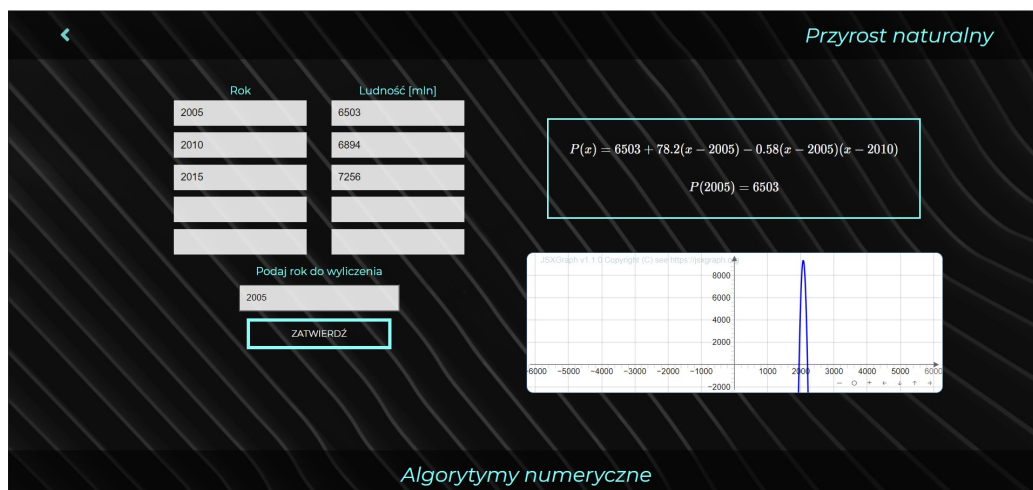
Dodatkowo przeprowadzane są operacje na liczbach - *float* - a także, w celu wyświetlania rezultatów, na danych typu *String*.

Struktura skryptu obliczeniowego odpowiada warstwie teoretycznej implementacji - tj. kolejność wykonywanych zadań w jednym skrypcie jest taka sama jak w sekcji **2.1** oraz **2.2**.

Wykorzystane zostały biblioteki KaTeX - do tworzenia tekstu matematycznego - i JsXGraph do tworzenia wykresów.

## 2.4 Program

Poniżej zostały zaprezentowane przykłady działania programu dla różnych przypadków:



<
Przyrost naturalny

Rok	Ludność [mln]
2005	6503
2010	6894
2015	

Uzupełnij pola poprawnymi zmiennymi!  
 Przynajmniej 3 lata i rok do wyliczenia

Podaj rok do wyliczenia  
  

ZATWIERDŹ

Algorytmy numeryczne

<
Przyrost naturalny

Rok	Ludność [mln]
2	50
3	68
4	89
5	99
6	124

$$P(x) = 50 + 18(x - 2) + 1.5(x - 2)(x - 3) - 2.3333333333333335(x - 2)(x - 3)(x - 4) + 1.6666666666666667(x - 2)(x - 3)(x - 4)(x - 5)$$

$$P(8) = 523$$

Podaj rok do wyliczenia  
  

ZATWIERDŹ

Algorytmy numeryczne

<
Przyrost naturalny

Rok	Ludność [mln]

Uzupełnij pola poprawnymi zmiennymi!  
 Przynajmniej 3 lata i rok do wyliczenia

Podaj rok do wyliczenia  
  

ZATWIERDŹ

Algorytmy numeryczne

## 2.5 IO (wejście-wyjście)

Wejściowe dane w programie są podawane w kontenerach typu *input*:

```
<input type="number" (...) class="input-v">
```

I pobierane przez program w skrypcie *jQuery* po naciśnięciu przycisku. Akceptowane są tylko wartości liczbowe, które mimo wszystko przechodzą do programu jako *String* - skrypt zamienia je po ich wprowadzeniu na wartości liczbowe.

Program sprawdza, czy podano poprawne dane, tj. czy liczba podanych lat i wartości zgadza się, czy nie pozostawiono jednego z pól pustego, czy podano wartość do wyliczenia. Jeśli znaleziony zostanie błąd, program wyświetli odpowiedni komunikat w kontenerze.

Dane wyjściowe programu to wartości *String* zawierające wzór wielomianu - wzór użyty do wyświetlenia wielomianu jest renderowany przez bibliotekę *KaTeX* na podstawie "brzydkiego", wyliczonego wzoru. Program na podstawie "brzydkiego" wzoru generuje również wykres funkcji przez bibliotekę *JsXGraph*. Dodatkowo pojawia się wyliczona wartość aproksymacji dla podanego roku.

## 3 Projekt nr 2

## 4 Projekt nr 3