Labolatorium 1
Metody Numeryczne
Wydział Fizyki i Informatyki Stosowanej
Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

Maciej Piwek

7 marca 2021

1 Wstęp

Na labolatoriach zapoznałem się z **metodą eliminacji Gaussa**. Na początku zajęć omówiono działanie algorytmu znanego nam z kursu Algebra I. Algorytm ten służy do rozwiązywania układów liniowych. Jego celem jest, aby macierz **A** sprowadzić do macierzy jednostkowej, gdzie wektor **b** jest wektorem wyrazów wolnych, wykorzystując operacje elementarne i finalnie rozwiązać układ równań.

2 Zadanie 1.1

Na zajęciach prowadzący omówił ważne kwestie dotyczące oscylatora harmonicznego m.in. równanie opisujące go :

$$\frac{d^2x(t)}{dt^2} = -(\frac{k}{m})x(t) = -\omega^2 x(t) \tag{1}$$

Po zastosowaniu defjnicji pochodnej:

$$\frac{d^2x(t)}{dt^2} \approx \frac{x(t+\Delta t) - 2x(t) + x(t-\Delta t)}{\Delta t^2} \tag{2}$$

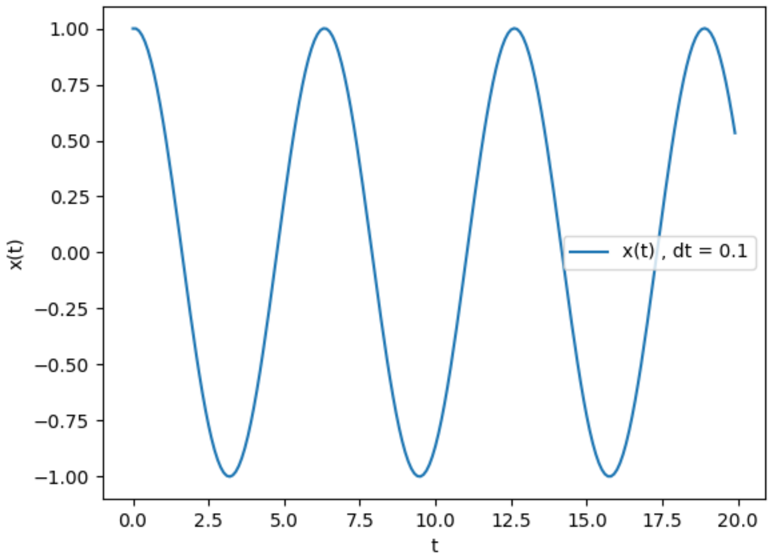
i wprowadzeniu oznaczenia $\Delta t = h$, $x_i = x(ih)$ orzymano z wcześniejszego równania iteracyjną zależność, pomiędzy x_i i x_{i-1}:

$$x_{i+1} + (\omega^2 h^2 - 2)x_i + x_{i-1} = 0 \tag{3}$$

Przy zadanych warunkach początkowych $x_0 = A$ oraz $(x_1 - x_0)/h = v_0$ miałem za zadanie rozwiązać podany układ macierzy dla siedmiu pierwszych kroków **metodą eliminacji Gaussa**:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & (\omega^2h^2-2) & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & (\omega^2h^2-2) & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & (\omega^2h^2-2) & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & (\omega^2h^2-2) & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & (\omega^2h^2-2) & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} A \\ v_0 h \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{4}$$

A - początkowe wychylenie z położenia równowagi
v_0 - prędkość początkowa ciała
Następnie narysowałem wykres zależności wychylenia z położenia róznowagi od czasu stosując **API Pylab** w Pythonie3.



Rysunek 1: Wykres zależności wychylenia od czasu $x(t)$

3 Zadanie 1.2

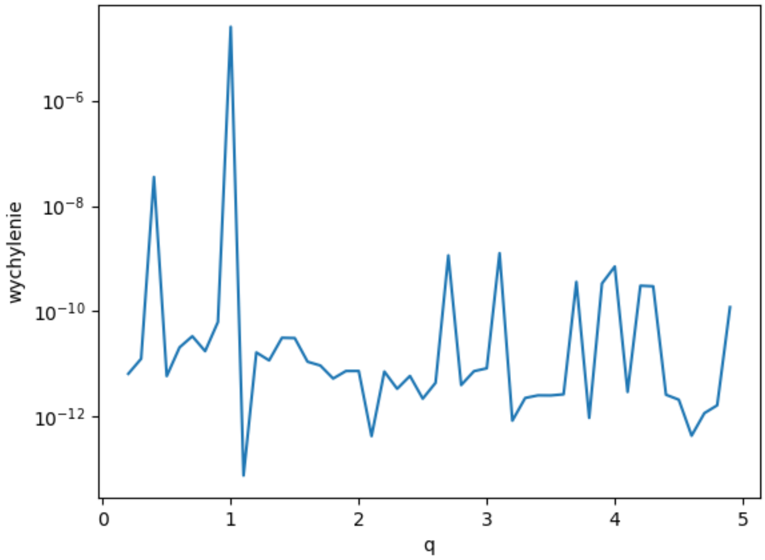
W **zadaniu 1.2** trzeba było rozwiązać układ $\boldsymbol{A}{\cdot}\boldsymbol{x} = \boldsymbol{b}$, a następnie sprawdzić poprawność prodcedury obliczając iloczyn $\boldsymbol{c} = \boldsymbol{A} \cdot \boldsymbol{x}$, gdy

$$\boldsymbol{A} = \begin{pmatrix} 2q \cdot 10^{-4} & 1 & 6 & 9 & 10 \\ 2 \cdot 10^{-4} & 1 & 6 & 9 & 10 \\ 1 & 6 & 6 & 8 & 6 \\ 5 & 9 & 10 & 7 & 10 \\ 3 & 4 & 9 & 7 & 9 \end{pmatrix}, oraz \boldsymbol{b} = \begin{pmatrix} 10 \\ 2 \\ 9 \\ 9 \\ 3 \end{pmatrix} \tag{5}$$

Postanowiłem zaimplementować ten algorytm w Python3, ponieważ dostrzegłem tam dość duże zastosowanie biblioteki **numpy**, **list** oraz jest dość intuicyjnym językiem. Program przedstawia następujące etapy:

1. Obliczanie rzędu macierzy **A**
2. Eliminacja zmiennych
3. Postępowanie odwrotne
4. Zwrócenie macierzy z rozwiązaniem i załadowanie wykresu

Po uruchomieniu już zaimplementowanego algorytmu otrzymałem następujący wykres:



Rysunek 2: Wykres zależności wychylenia od wartości parametru q, przy zadanym skoku 0.1000000234211

q - parametr
$o(q) = 1/5\sqrt{\sum_{i=1}^{5}(c_i - b_i)^2}$- zależność wychylenia od wartości parametru q

4 Wnioski:

1. Największe wychylenie możemy zaobserwować dla $q = 1$
2. Współczesne oprogramowanie może dość znacznie przyspieszyć obliczenia matematyczne.
3. Pewne koncepty z matematyki jesteśmy w stanie skutecznie przenieść do świata programowania.