Sprawozdanie projekt nr 1. MNUM

# Wstęp

Projekt zawiera w sobie 3 zadania:

1. Wyznaczenie dokładności maszynowej komputera
2. Implementacja algorytmu do rozwiązywania układów równań za pomocą metody eliminacji Gaussa z częściowym wyborem elementu podstawowego.
3. Implementacja algorytmu do rozwiązywania układów równań za pomocą metody Jacobiego.
4. Wyznaczenie błędów rozwiązań i sprawdzenie ich poprawności.

# Zad 1.

Dokładność maszynowa, oznaczana jako „eps” jest maksymalnym błędem względnym reprezentacji zmiennoprzecinkowej. Zależy ona jedynie od długości mantysy (liczby bitów). Dlatego też, jeśli komputery używają tych samych standardów zapisu liczb zmiennopozycyjnych a co za tym idzie długość mantysy jest taka sama, to i eps będzie na różnych komputerach taki sama, dla tych samych typów. Dokładność maszynową można też interpretować jako maksymalną wartość, która po dodaniu do danej liczby nie zmieni jej zapisu w komputerze. Wyraża się to następującym wzorem:

Rd(x) – x = x \* ℮, gdzie |℮| <= 2-t = eps i ‘t’ to liczba bitów mantysy

, czyli Rd(x) = x(1+℮), |℮| <= eps

Zaprojektowany przeze mnie algorytm do obliczania eps wygląda następująco:

function eps = my\_epsilon()

x = 1.0;

y = 2.0;

while (y > 1)

eps = x;

x = x/2;

y = 1.0 + x;

end

end

Wynikiem w środowisku Matlab jest zawsze 2.2204e-16. Wynika to z tego, że domyślnie liczby zmiennopozycyjne są tam zapisywane w standardzie IEEE® 754 jako liczby z podwójną precyzją, czyli ich mantysa ma długość 52 bitów. Wynik algorytmu zgadza się z wartością obliczoną teoretycznie ze wzoru

Eps = 2-t

W języku C/C++ ten sam algorytm będzie dawał różne wyniki, z tego względu, że można tam definiować jakim typem danych komputer ma się posługiwać. Dla tego też dla typu double wynik będzie taki sam jak w Matlabie a dla typu float będzie to 1.1902e-07 a dla long double 1.0842e-19.

Tak wyglądał algorytm dla języka C++.

#include<iostream>

int main()

{

//float or long double

double eps, x, y;

eps = 0;

x = 1.0;

y = 2.0;

while(y > 1)

{

eps = x;

x = x/2;

y = 1.0 + x;

}

std::cout << "machine epsilon equals " << eps << std::endl;

}