Zadanie II.9

Damian Soliński
Piotr Kulas
Marek Skiba

04.12.2012

1 Treść zadania

Dla równania $f(x) = 0$, gdzie $f(x) = e^x + x + 2$, wczytać a,b $\in \mathbf{R}$ takie, by $a < b$ oraz $f(a) * f(b) < 0$. Nastepnie, dopóki "użytkownik sie nie znudzi", wczytywać wartość $0 < \epsilon < 1$ i metoda połowienia na $[a, b]$ przybliżyć z dokładnościa ϵ rozwiazanie tego równania. Rozwiazanie to przybliżyć również metoda Newtona z $x_0 = a$, przy czym x_k bedzie dobrym przybliżeniem, gdy $|x_k - x_(k-1)| \leq \epsilon$. Porównać ilość kroków wykonanych metoda połowienia i metoda Newtona.

2 Schemat rozwiazywania zadania

Najpierw pobieramy od użytkownika dwie liczby: a i b, należace do liczb rzeczywistych, które spełniaja: $a < b$ oraz $f(a) * f(b) < 0$. Nastepnie w petli aż użytkownik sie nie znudzi pobieramy przybliżenie (ϵ) i liczymy dwoma sposobami: metoda równego podziału (metoda połowienia) oraz metoda Newtona.

Metoda równego podziału (metoda połowenia)

1. Sprawdzić, czy pierwiastkiem równania jest punkt $x_1 = \frac{a+b}{2}$, czyli czy $f(x_1) = 0$.
2. Jeżeli tak jest, algorytm kończy sie, a punkt jest miejscem zerowym. W przeciwnym razie x_1 dzieli przedział $[a, b]$ na dwa mniejsze przedziały $[a, x_1]$ i $[x_1, b]$.
3. Wybierany jest ten przedział, dla którego spełnione jest drugie założenie, tzn. albo $f(x_1)f(a) < 0$ albo $f(x_1)f(b) < 0$. Cały proces powtarzany jest dla wybranego przedziału.

Działanie algorytmu kończy sie w punkcie 2 albo po osiagnieciu żadanej dokładności przybliżenia pierwiastka.

Metoda Newtona

W pierwszym kroku metody wybierany jest punkt startowy x_0 (w naszym wypadku wartość liczby a). Wyliczamy x_1 ze wzoru $x_0 - \frac{f(x_0)}{df(x_0)}$. Nastepnie podkładamy wynik x_1 pod nasza funkcje $f(x) = e^x+x+2$. Wynikiem funkcji bedzie przybliżenie, jeśli nie jest satysfakcjonujace, wówczas punkt x_2 jest wybierany jako nowy punkt startowy i wszystkie czynności sa powtarzane. Proces jest kontynuowany, aż zostanie uzyskane wystarczajaco dobre przybliżenie pierwiastka.

3 Przykładowe rozwiazania:

Metoda równego podziału (metoda połowenia)

Ustalamy, że: $a = -3$, $b = -2$, $\epsilon = 0.01$
$f(x) = e^x + x + 2$
$f(a) = -0,950212932$
$f(b) = 0,135335283$

$x_1 = \frac{a+b}{2} = -2.5$
$f(s) = e^{-2.5} - 2.5 + 2 = -0.41791$
$a = -2.5$
$b = -2$
$x_1 = \frac{a+b}{2} = -2.25$
$f(s) = e^{-2.25} - 2.25 + 2 = -0.144600775$
$a = -2.25$
$b = -2$
$x_1 = \frac{a+b}{2} = -2.125$
$f(s) = e^{-2.125} - 2.125 + 2 = -0,00556703173$

Osiagneliśmy oczekiwane przybliżenie, pierwiastkiem równania $f(x)$ w podanym przedziale jest $x = -2.125$

Metoda Newtona

Ustalamy, że: $a = -3$, $b = -2$, $\epsilon = 0.01$
$f(x) = e^x + x + 2$
$df(x) = e^x + 1$
$x_0 = a = -3$
$x_1 = -3 - \frac{f(-3)}{df(-3)} = -2,09485175$
$f(x_1) = -0,950212932$
$x_2 = -2,09485175 - \frac{f(-2,09485175}{df(-2,09485175)} = -2,11999379$
$f(x_2) = 0,000038583910$

Osiagneliśmy oczekiwane przybliżenie, pierwiastkiem równania $f(x)$ w podanym przedziale jest $x = -2,11999379$

4 Źródło programu:

```
#include <stdio.h>
#include <math.h>
double f(double x);
double df(double x);
void bisection(double a, double b, double e);
void newton(double x, double e);

int main() {
  double a, b, e;
  int status;
  char chce_podac_x[2];
  do {
    printf("Podaj␣a:␣");
    scanf("%lf", &a);
    printf("Podaj␣b:␣");
    scanf("%lf", &b);
    if(a < b) {
      if (f(a) * f(b) < 0) {
        do {
          printf("Podaj␣przybliżenie:␣");
          status = scanf("%lf", &e);
          if(e > 0 && e < 1) {
            bisection(a, b, e);
            newton(a, e);
          }
          do {
            printf("Chcesz␣podać␣kolejne␣przybliżenie?␣(tak/nie):␣");
            scanf("%s", chce_podac_x);
            if(strcmp(chce_podac_x, "tak") == 0 || strcmp(chce_podac_x, "nie") == 0)
          } while(1);
        } while (strcmp(chce_podac_x, "tak") == 0);
      } else {
        printf("bład:␣f(a)␣i␣f(b)␣musza␣mieć␣różny␣znak\n");
      }
    } else {
      printf("bład:␣a␣musi␣być␣mniejsze␣od␣b\n");
    }
  } while(a >= b || f(a) * f(b) >= 0);

  return 0;
}

double f(double x) {
  return exp(x) + x + 2;
}
```

```c
double df(double x) {
  return exp(x) + 1;
}

void bisection(double a, double b, double e) {
  double m = (a + b) / 2;
  double fm = f(m);
  int step = 0;
  while (fabs(fm) > e) {
    if ((f(a) * fm) < 0) {
      b = m;
    } else if ((f(b) * fm) < 0) {
      a = m;
    }
    m = (a + b) / 2;
    fm = f(m);
    step += 1;
  }
  printf("Metoda połowienia:\n");
  printf("x = %.16lf, f(x) = %.16lf, Kroki: %d\n", m, fm, step);
}

void newton(double x, double e) {
  int step = 0;
  while (fabs(f(x)) > e) {
    x -= f(x) / df(x);
    step += 1;
  }
  printf("Metoda Newtona:\n");
  printf("x = %.16lf, f(x) = %.16lf, Kroki: %d\n", x, f(x), step);
}
```

5 Wynik programu:

```
[mskiba@sigma] ~/MetodyObliczeniowe $ g++ zad2.cpp -o zad2
[mskiba@sigma] ~/MetodyObliczeniowe $ ./zad2
Podaj a: -3
Podaj b: -2
Podaj przybliżenie: 0.01
Metoda połowienia:
x = -2.1250000000000000,
f(x) = -0.0055670317332805,
kroki: 2.

Metoda Newtona:
x = -2.1199937939314832,
f(x) = 0.0000385795067956,
kroki: 2.

Chcesz podać kolejne przybliżenie? (tak/nie): tak
Podaj przybliżenie: 0.001
Metoda połowienia:
x = -2.1191406250000000,
f(x) = 0.0009942000281109,
kroki: 8.

Metoda Newtona:
x = -2.1199937939314832,
f(x) = 0.0000385795067956,
kroki: 2.

Chcesz podać kolejne przybliżenie? (tak/nie): nie
[mskiba@sigma] ~/MetodyObliczeniowe $
```