Politechnika Śląska **Gliwice**
Wydział Automatyki Elektroniki i Informatyki **Rok akademicki 2007/2008**
Kierunek: Automatyka i Robotyka sem.3 **Semestr zimowy**

Metody Numeryczne
Laboratorium

Ćw.4 : Różniczkowanie numeryczne

Wykonali:
Adam DZIENDZIEL
Adrian BIELEC

Grupa 4 Sekcja 1

Data odbycia ćwiczenia:
16.11.07

Program 1

```
#include<iostream>
#include<math.h>
#include<conio.h>
#include<stdlib.h>
using namespace std;

float czytaj()
 {
  float x;
   while(!(cin>>x))
   {
    cin.clear();
    cin.ignore(1000,'\n');
   }
  return x;
 }

double silnia(int x)
 {
  double s=1;
  int i;

  for(i=1;i<=x;i++)
  {
   s=s*i;
  }
  return s;
 }

float funkcja(float x)
 {
  float y=0;

  //y=4*pow(x,4)+2*pow(x,2)+2;
  //y=6*pow(x,5)+3*pow(x,3)+x;
  y=sin(x);
  //y=log(x);
  //y=1/x;
  //y=exp(x);
  return y;
 }

int main()
 {
  int i,j,n,nn,koniec,l,optyn;
  float
a,x0,x,h,optyh,hh,fnx,optyfnx,fnxx,y,eps,b1,b2,b3,optyb1,optyb2,optyb3;
  float **tab;

  cout<<"W jakim punkcie liczyc pochodna? x0 = "; x0=czytaj();
  cout<<"Podaj krok poczatkowy h = "; h=czytaj();
  cout<<"Podaj poczatkowa ilosc krokow n = "; n=czytaj();
  cout<<"Podaj dokladnosc z jaka dokonywac obliczen eps = "; eps=czytaj();


  optyb3=9999999999.9;
  n++;
  nn=n;
  hh=h;
```

```
  tab=(float**)malloc(n*sizeof(float));
  for(i=0;i<n;i++)
  {
   tab[i]=(float*)malloc((n-i)*sizeof(float));
  }

  for(l=n;l<30;l++)
  {
   h=hh;
   do
   {
//pochodne wyznaczone analitycznie

    //y=16*pow(x0,3)+4*x0;
    //y=30*pow(x0,4)+9*pow(x0,2)+1;
    y=cos(x0);
    //y=1/x0;
    //y=-1/pow(x0,2);
    //y=exp(x0);
    for(i=0;i<n;i++)
    {
     tab[0][i]=funkcja(x0+i*h);
    }

    for(i=1;i<n;i++)
    {
     for(j=0;j<n-i;j++)
     {
      tab[i][j]=tab[i-1][j+1]-tab[i-1][j];
     }
    }

    fnxx=fnx;
    fnx=0;

    for(i=1;i<(n-1);i++)
    {
     fnx=fnx-((pow(-1,i)/float(i))*tab[i][0]);
    }

    fnx=fnx/h;
    b1=fabs((tab[n-1][0]*funkcja(x0))/(h*silnia(n)));
    b2=fabs(y-fnx);
    b3=fabs(b1-b2);

    if(fabs(fnxx-fnx)>=eps)
    {
     if(h>=0.0000001)
     {
      h=h/2.0;
      koniec=0;
     }else koniec=1;
    }else koniec=1;
```

```
//komunikaty potrzebne do sprawdzenia oprawności działania programu

    //cout<<endl<<"wynik numeryczny         >>> "<<fnx;
    //cout<<endl<<"wynik analityczny        >>> "<<y<<endl;
    //cout<<endl<<"blad metody       >>> "<<b1;
    //cout<<endl<<"blad rzeczywisty  >>> "<<b2;
    //cout<<endl<<"blad numeryczny   >>> "<<b3;
    //cout<<endl<<"dokladnosc uzyskana dla kroku h >>> "<<h*2;
    //cout<<endl<<"oraz liczby krokow n            >>> "<<n-1;
    //cout<<endl<<"test dokladnosci                >>> "<<fabs(fnxx-fnx);
    //cout<<endl<<endl;
   }while(koniec!=1);
   if(b3<optyb3)
   {
    optyb1=b1;
    optyb2=b2;
    optyb3=b3;
    optyn=n;
    optyh=h;
    optyfnx=fnx;
   }

   n++;
   tab=(float**)realloc(tab,(n)*sizeof(float));
   for(i=0;i<n;i++)
   {
    tab[i]=(float*)realloc(tab[i],(n-i)*sizeof(float));
   }
  }

  cout<<endl<<endl;
  cout<<"-----------------------------------------------------------"<<endl;
  cout<<"------------------OPTYMALNE PARAMETRY------------------"<<endl;
  cout<<endl;
  cout<<"      krok h = "<<optyh<<endl;
  cout<<"      ilosc krokow n = "<<optyn<<endl;
  cout<<endl;
  cout<<endl<<"        pochodna wynosi wtedy     >>>"<<optyfnx;
  cout<<endl<<"        przy wyniku analitycznym  >>>"<<y;
  cout<<endl<<"        blad metody       >>> "<<optyb1;
  cout<<endl<<"        blad rzeczywisty  >>> "<<optyb2;
  cout<<endl<<"        blad numeryczny   >>> "<<optyb3;



 getch();
}
```

Program 2

```
#include<iostream>
#include<math.h>
#include<conio.h>
#include<stdlib.h>
using namespace std;

float czytaj()
 {
  float x;
   while(!(cin>>x))
   {
    cin.clear();
    cin.ignore(1000,'\n');
   }
  return x;
 }

double silnia(int x)
 {
  double s=1;
  int i;

  for(i=1;i<=x;i++)
  {
   s=s*i;
  }
  return s;
 }

float funkcja(float x)
 {
  float y=0;

  //y=4*pow(x,4)+2*pow(x,2)+2;
  //y=6*pow(x,5)+3*pow(x,3)+x;
  y=sin(x);
  //y=log(x);
  //y=1/x;
  //y=exp(x);
  return y;
 }

int main()
 {
  int i,j,n,l;
  float y,a,x0,x,h,fnx,eps,xp,b1,b2,xk,b3,bmax,bmin;
  float **tab;

  cout<<"od jakiego x zaczac? xp = "; xp=czytaj();
  cout<<"na jakim skonczyc  ? xk = "; xk=czytaj();
  cout<<"Podaj krok poczatkowy h = "; h=czytaj();
  cout<<"Podaj poczatkowa ilosc krokow n = "; n=czytaj();
  bmax=0;
  bmin=9999999;
  n++;
```

```
 for(x0=xp;x0<=xk;x0=x0+0.1)
  {
   tab=(float**)malloc(n*sizeof(float));
   for(i=0;i<n;i++)
   {
    tab[i]=(float*)malloc((n-i)*sizeof(float));
   }

 //pochodne wyznaczone analitycznie

//y=16*pow(x0,3)+4*x0;
//y=30*pow(x0,4)+9*pow(x0,2)+1;
y=cos(x0);
//y=1/x0;
//y=-1/pow(x0,2);
//y=exp(x0);

   for(i=0;i<n;i++)
   {
    tab[0][i]=funkcja(x0+i*h);
   }

   for(i=1;i<n;i++)
   {
    for(j=0;j<n-i;j++)
    {
     tab[i][j]=tab[i-1][j+1]-tab[i-1][j];
    }
   }

   fnx=0;
   for(i=1;i<(n-1);i++)
   {
    fnx=fnx-((pow(-1,i)/float(i))*tab[i][0]);
   }
   fnx=fnx/h;
   cout<<fnx<<endl;

   b1=(tab[n-1][0]*funkcja(x0))/(h*silnia(n));
   b2=fabs(y-fnx);
   b3=fabs(b1-b2);

   if (b3>bmax) bmax=b3;
   if (b3<bmin) bmin=b3;
  }
  cout<<endl<<endl<<"blad maksymalny >>> "<<bmax<<"  minimalny >>> "<<bmin;

  getch();
 }
```

Program 1 wyznacza optymalne wartości n (ilość kroków) i h (szerokość przedziału) dla których błąd numeryczny różniczkowania w zadanym punkcie x_0. Został on przetestowany dla 6 funkcji w różnych miejscach x_0 z dokładnością 0.001. Program 2 natomiast dla podanych wartości n i h szuka pochodnych w kolejnych punktach począwszy od x_0 czyli $\{f'(x_0);f'(x_0+0.1);f'(x_0+0.2);\ \ldots\ f'(x_0+n*0.1)\}$ oraz zwraca największy i najmniejszy błąd numeryczny z zadanego przedziału. Dane z programu 2 posłużyły także do sporządzenia wykresów konfrontujących wartości pochodnych wyznaczonych numerycznie i analitycznie. A oto wyniki testowania programów:

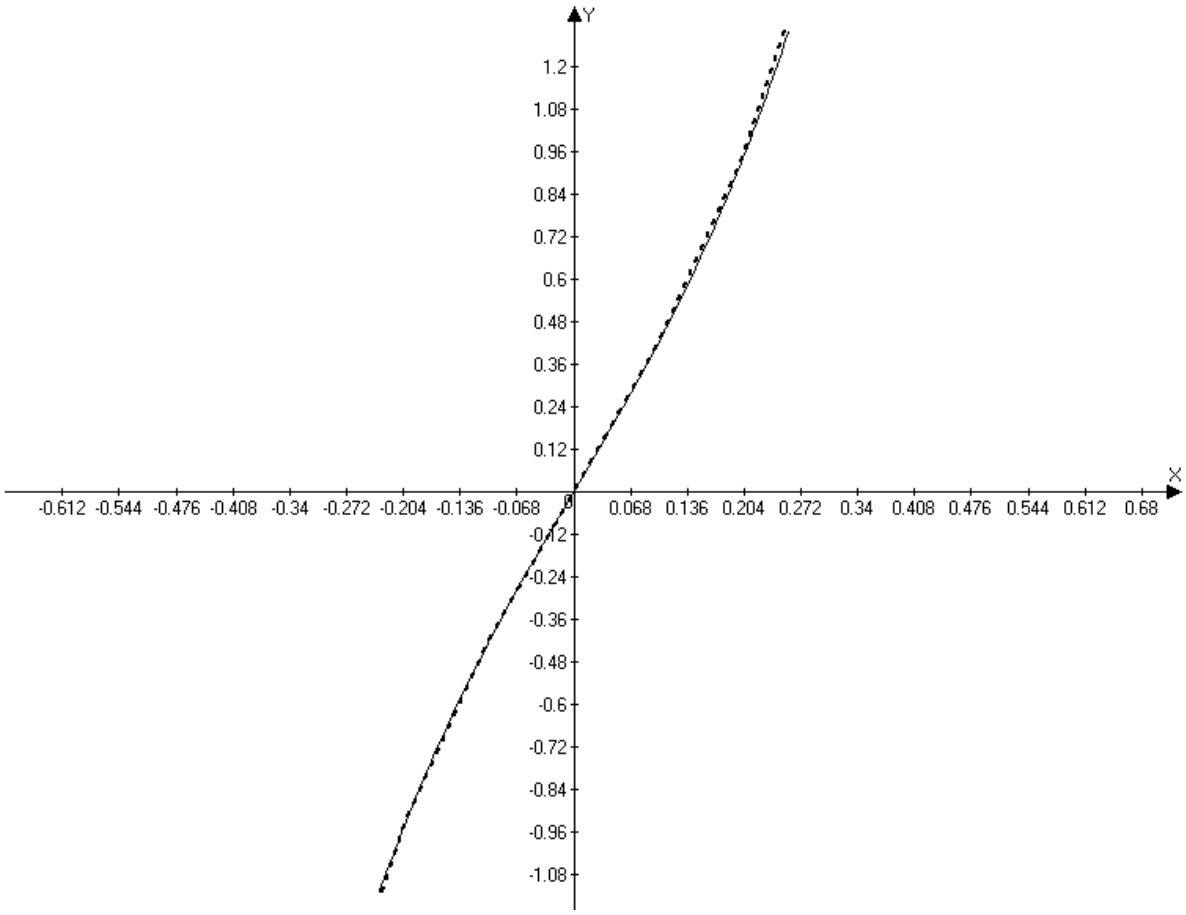
Funkcja I

$f(x) = 4x^4 + 2x^2 + 2$
$f'(x) = 16x^3 + 4x$

x_0	n	h	Błąd metody	Błąd rzeczywisty	Błąd numeryczny
	OPTYMALNE				
-0.4	9	0.15625	2.04684e-013	2.38419e-007	2.38418e-007
-0.3	9	0.15625	1.39797e-012	5.96046e-008	5.96032e-008
0	23	9.53674e-006	0	0	0
0.2	11	0.3125	1.51889e-013	2.98023e-007	2.98023e-007
0.5	11	0.3125	3.21864e-013	1.78814e-007	1.78814e-007

Poniższy wykres przedstawia pochodną analityczna (**linia ciągła**) i numeryczną (**linia przerywana**) wyznaczane w przedziale (-0.7;0.7) dla **h = 0.15** i **n = 9 .**

błąd maksymalny >>> **0.000223157** w $\mathbf{x_0 = 0.6}$
Jak widać dla optymalnych n i h wykresy niemalże się pokrywają.



Funkcja II

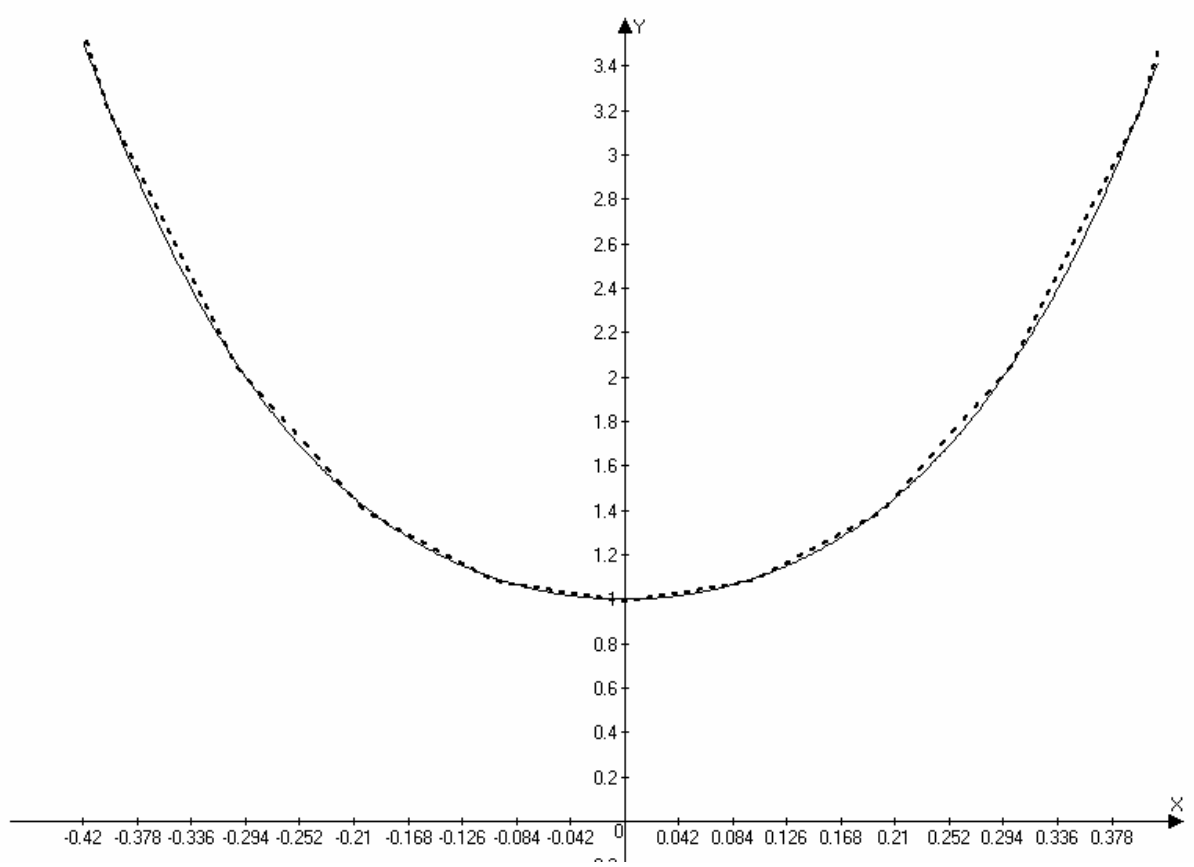
$f(x) = 6x^5 + 3x^3 + x$
$f'(x) = 30x^4 + 9x^2 + 1$

x_0	n	h	Błąd metody	Błąd rzeczywisty	Błąd numeryczny
	OPTYMALNE				
-0.9	12	0.15625	1.01112e-012	0	1.01112e-012
-0.7	11	0.15625	1.75275e-012	3.8147e-006	3.8147e-006
0	23	0.000002384	0	0	0
0.2	7	0.3125	9.30215e-009	2.15769e-005	2.15676e-005
0.8	9	0.15625	7.23595e-010	0	7.23595e-010

Poniższy wykres przedstawia pochodną analityczna (**linia ciągła**) i numeryczną (**linia przerywana**) wyznaczane w przedziale (-1;1) dla **h = 0.1** i **n = 5 .**

błąd maksymalny >>> **0.0153746** w x_0 **= -1**

Tym razem nieco odbiegamy od optymalnych wartości n i h i widać że błędy są nieco większe.



Funkcja III

f(x) = sin(x)
f’(x) = cos(x)

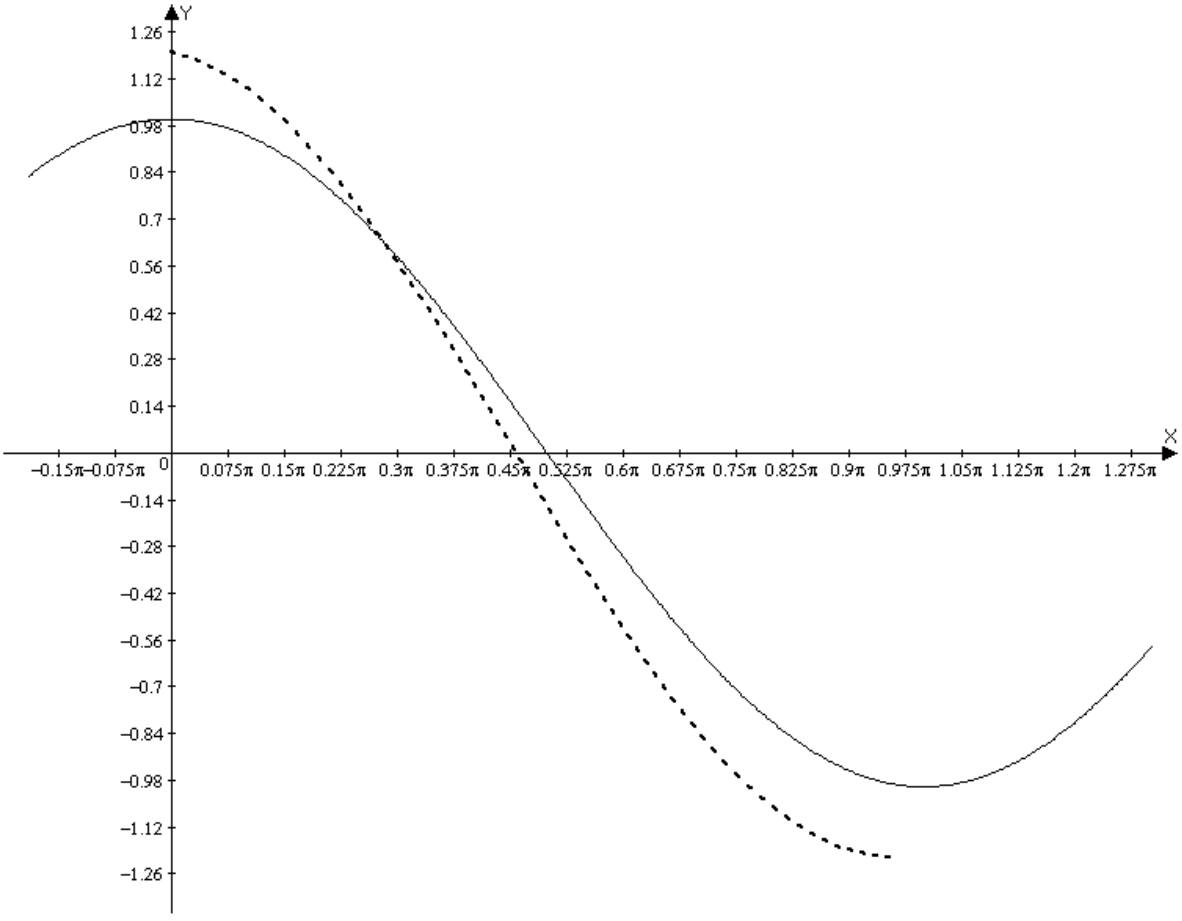
x_0	n	h	Błąd metody	Błąd rzeczywisty	Błąd numeryczny
	OPTYMALNE				
0.5	11	0.3125	3.21864e-013	1.78814e-007	1.78814e-007
0.9	12	0.3125	2.05864e-014	2.98023e-007	2.98023e-007
1.5	8	0.15625	9.43735e-012	1.2815e-006	1.28149e-006
1.9	12	0.3125	1.97825e-014	3.66569e-006	.66569e-006
2.5	13	0.3125	2.34641e-015	2.38419e-007	2.38419e-007

.

Poniższy wykres przedstawia pochodną analityczna (**linia ciągła**) i numeryczną (**linia przerywana**) wyznaczane w przedziale (0;Π) dla **h = 0.9** i **n = 3 .**

błąd maksymalny >>> 0.242279 w $\mathbf{x_0 = 2.6}$

Jeszcze bardziej odbiegamy od optymalnych wartości i błędy są większe i jeszcze bardziej widoczne.



Funkcja IV

f(x) = ln(x)
f’(x) = 1/x

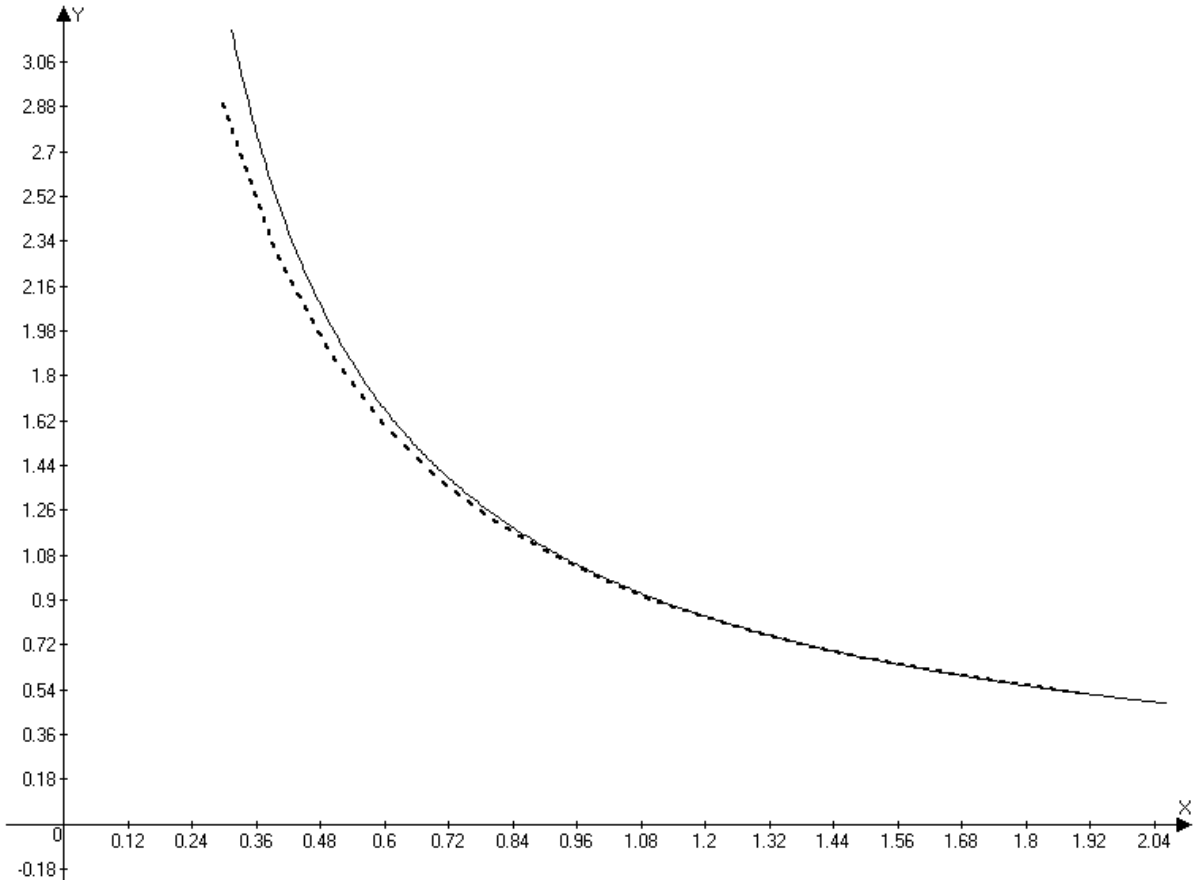
x_0	n	h	Błąd metody	Błąd rzeczywisty	Błąd numeryczny
	OPTYMALNE				
0.3	12	0.0390625	7.28708e-014	0	7.28708e-014
0.7	15	0.078125	1.63693e-016	4.64916e-006	4.64916e-006
1.1	11	0.15625	4.03959e-013	1.37091e-006	1.37091e-006
1.6	22	4.76837e-006	0	0	0
2	23	9.53674e-006	0	0	0

.

Poniższy wykres przedstawia pochodną analityczna (**linia ciągła**) i numeryczną (**linia przerywana**) wyznaczane w przedziale (0.3;2) dla **h = 0.6** i **n = 8 .**

błąd maksymalny >>> **0.444428** w $\mathbf{x_0 = 0.3}$

Kolejny raz odbiegamy od wartości optymalnych w których oba wykresy się pokrywają. Ta funkcja wymaga dużych wartości n i małych h dlatego przyjmujemy takie h i n.



Funkcja V

$f(x) = 1/x$
$f'(x) = -1/x^2$

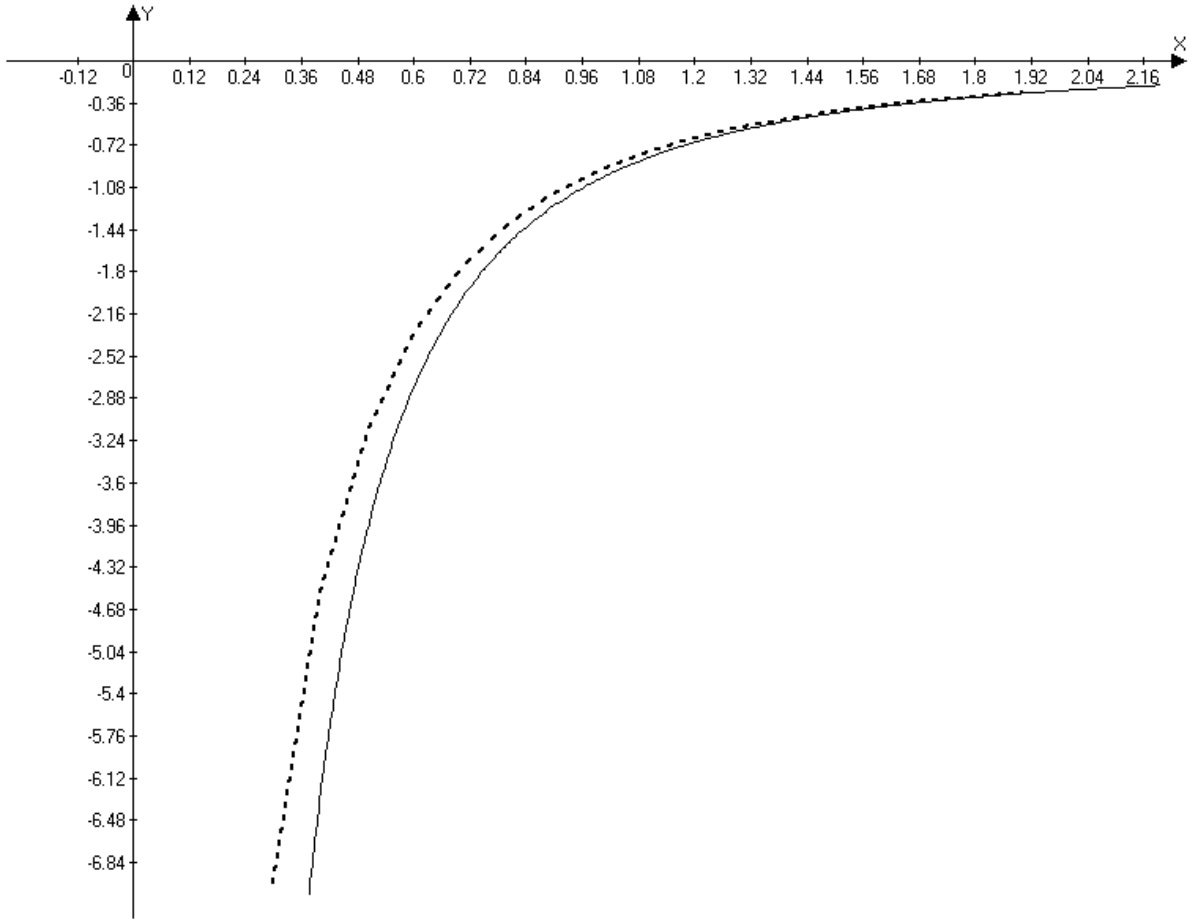
x_0	n	h	Błąd metody	Błąd rzeczywisty	Błąd numeryczny
	OPTYMALNE				
0.3	4	0.000610352	0	2.19345e-005	2.19345e-005
0.7	8	0.0390625	6.70383e-009	2.38419e-007	2.31715e-007
1.1	9	0.078125	1.8731e-010	1.13249e-006	1.1323e-006
1.7	15	0.3125	1.41632e-016	1.37091e-006	1.37091e-006
2	26	4.76837e-006	0	0	0

.

Poniższy wykres przedstawia pochodną analityczna (**linia ciągła**) i numeryczną (**linia przerywana**) wyznaczane w przedziale (0.3;2) dla **h = 0.7** i **n = 8 .**

błąd maksymalny >>> **4.10064** w $\mathbf{x_0 = 0.3}$

Analogicznie do poprzedniej funkcji, minimalnie zwiększamy tylko h.



Funkcja VI

$f(x) = e^x$
$f'(x) = e^x$

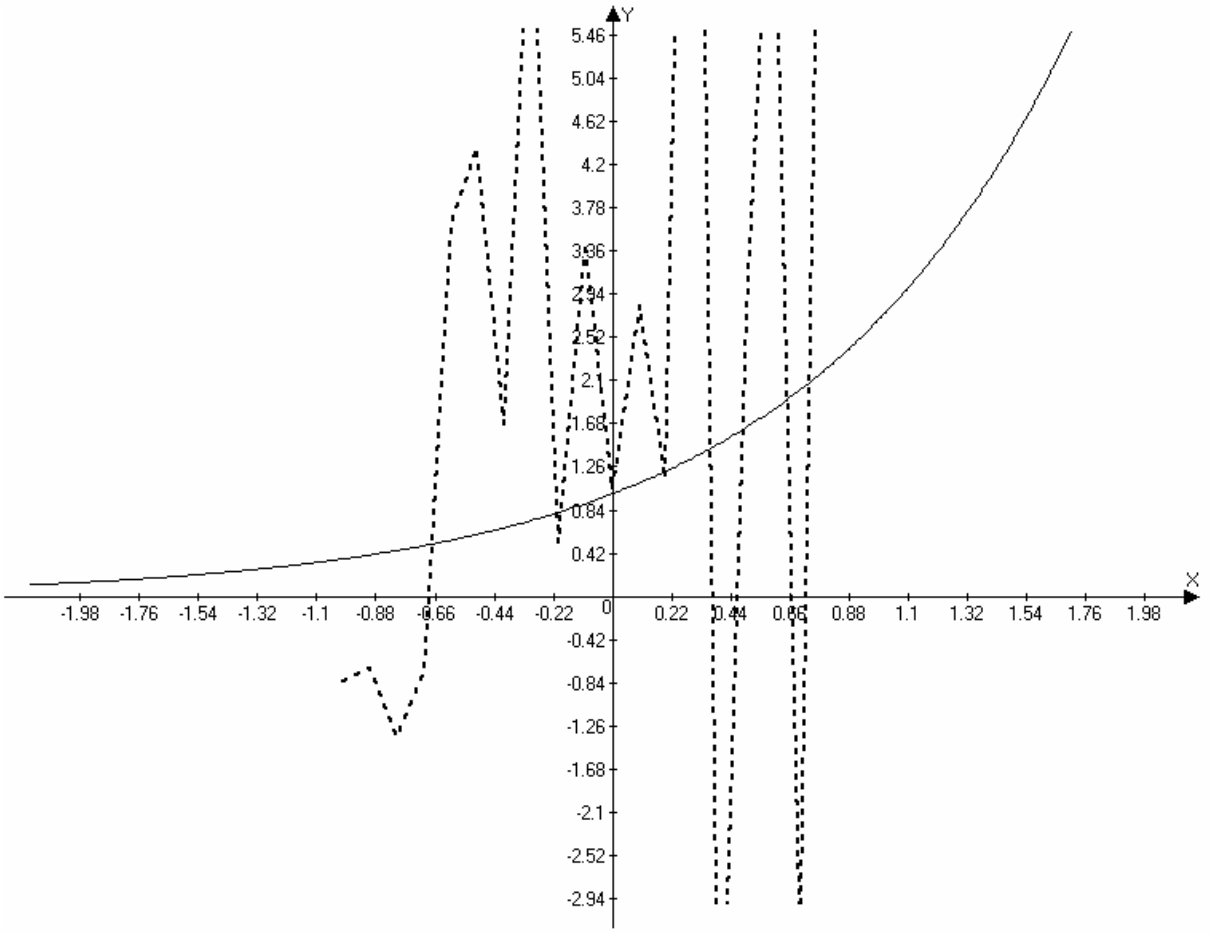
x_0	n	h	Błąd metody	Błąd rzeczywisty	Błąd numeryczny
	OPTYMALNE				
-1	6	0.078125	9.16075e-009	2.83122e-006	2.82206e-006
-0.5	10	0.15625	2.80545e-012	3.57628e-007	3.57625e-007
0	6	0.078125	6.56976e-008	6.55651e-006	6.49081e-006
0.5	7	0.078125	1.49747e-009	4.29153e-006	4.29004e-006
1	9	0.15625	1.82882e-010	7.15256e-007	7.15073e-007

.

Poniższy wykres przedstawia pochodną analityczna (**linia ciągła**) i numeryczną (**linia przerywana**) wyznaczane w przedziale (-1;1) dla **h = 0.001** i **n = 23 .**

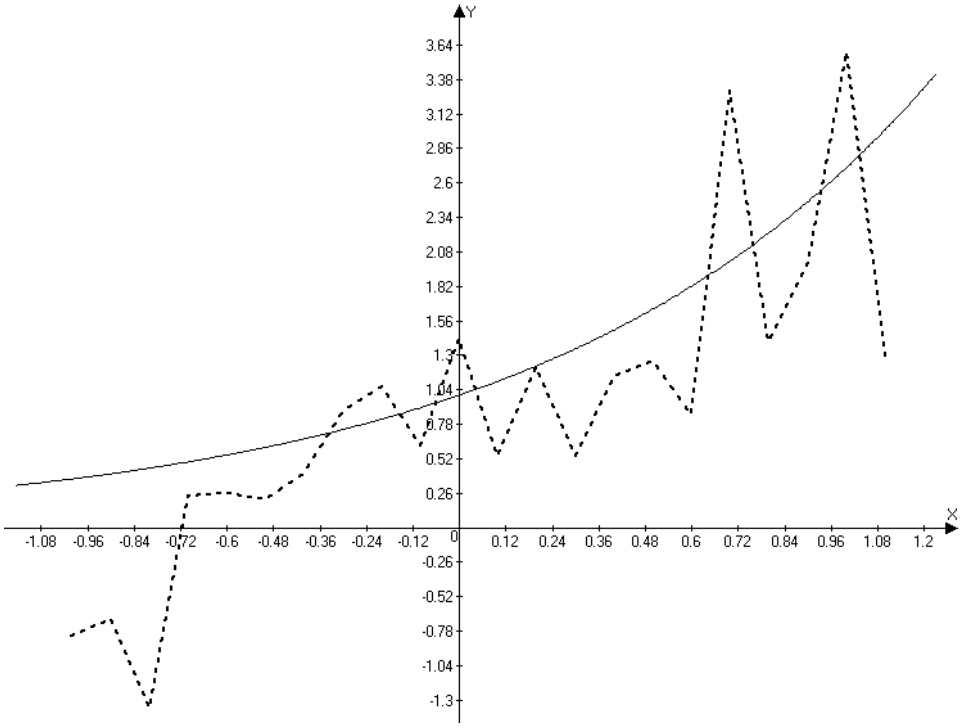
błąd maksymalny >>> **11.9342** w $\mathbf{x_0 = 0.4}$

Tym razem odbiegamy znacznie od wartości optymalnych ale w przeciwną niż w poprzednich przykładach stronę zmniejszamy h a zwiększamy n



Dla porównania pokażemy jeszcze o ile zmieni się błąd jeśli n zmienimy o 3 z 23 na 20:

błąd maksymalny >>> **1.8023** w $\mathbf{x_0 = 0.4}$



A teraz zmieniamy h: mamy wiec n = 20 i h = 0.01

błąd maksymalny >>> **0.152644** w $\mathbf{x_0 = 0.8}$