بسمه تعالی

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف

- مشخصات : عطیه جمشیدپور
- شماره دانشجویی:۹۴۱۰۳۸۳۵

تمرین اول محاسبات عددی

موعد تحویل : ۱۲/۱۵

تمرین اول

الف)

83.12512, 83.12513, 83.125514 more precise

83.1253, 83.1254, 83.1255 more accurate [1]

ب)

1. خطای مدل : ممکن است زمین کشاورزی مورد نظر به شکلی شبیه به یک مستطیل اما نه دقیقا یک مستطیل باشد، اما برای سادگی این زمین را مستطیل در نظر بگیریم.
2. خطای نمایش اعداد : به خاطر محدودیت در ذخیره بسط اعداد اعشاری ایجاد می‌شود. نمیتوان طول و عرض زمین را به طور دقیق ذخیره کرد. لذا آنها را گرد خواهیم نمود. همچنین در صورتی که زمین دایره‌ای شکل باشد به ناچار از مقدار گرد شده عدد π استفاده می‌کنیم.
3. خطای عملیات : هنگام اجرای محاسبات با اعداد تقریبی، خطاهای مربوط به داده‌های اولیه به نتیجه نهایی منتقل می‌شوند. به طور مثال خطا در اندازه گیری طول و عرض زمین به خطا در محاسبه مساحت که حاصل ضرب این دو مقدار است می‌انجامد.
4. خطای داده‌ها یا خطای اولیه : به دلیل خطای انسانی ممکن است در زمان اندازه گیری طول و عرض مقدار روی متر را به اشتباه بخوانیم[2].

تمرین دوم

الف)

$\cos(x)=1-x^2/2!+x^4/4!+\ldots+(-1)^n x^{(2n)}/(2n)!$¿

$\sin(x)=x-x^3/3!+x^5/5!+\ldots+(-1)^n x^{(2n+1)}/(2n+1)!$

$\frac{1-\cos(x)}{\sin(x)};\frac{2\sin^2(x/2)}{2\sin(x/2)\cos(x/2)}=\tan(x/2)$

$\tan(x)=x+x^3/3+2x^5/15+\ldots;\tan(x/2)=x/2+x^3/24+x^5/240+\ldots$

absolute error $x^5/240$

ب

$\frac{|A-a|}{A}<10^{-5};\ a/(1+10^{-5})<A<a/(1-10^{-5});\ 13.34449<A<13.34476$

لذا تا سه رقم اعشار میتوان به تقریب اطمینان داشت.

تمرین سوم

الف)

$f(x)=\frac{1}{\sqrt{x+1}-\sqrt{x}}; f(1{,}000{,}000)=2{,}000.0005; f(x)=\sqrt{x+1}+\sqrt{x}; f(1{,}000{,}000)=2{,}000.0004$

باتوجه به اینکه تفریق دو عدد نزدیک به هم با خطای زیادی توام است، بهتر است از رابطه دوم استفاده شود.داریم :

$\delta_{(x-y)}=\frac{|x|\delta_x+|y|\delta_y}{|x-y|}; if\ x\quad y: \delta_{(x-y)}=\infty\ proof: \delta_{(x-y)}\leq e_{(x-y)}/|x-y|=\frac{e_x+e_y}{|x-y|}=\frac{|x|\delta_x+|y|\delta_y}{|x-y|}$

درحالیکه

$\delta_{(x+y)}=\frac{|x|\delta_x+|y|\delta_y}{|x+y|};$

همچنین در خصوص تقسیم داریم

$\delta_{(x/y)}=\delta_x+\delta_y; \delta_{\left(\frac{1}{\sqrt{x+1}-\sqrt{x}}\right)}=\delta_{(\sqrt{x+1}-\sqrt{x})}$

$proof: \delta_{(x/y)}\leq e_{(x/y)}/(x/y)=\frac{\frac{x e_y+y e_x}{y^2}}{(x/y)}=e_x/x+e_y/y=\delta_x+\delta_y$

ب)

$e_f\leq e_x\vartheta f/\vartheta x+e_y\vartheta f/\vartheta y=(e_x+e_y)(x+y)^{(-0.5)}/2$

$e_g\leq e_x\vartheta g/\vartheta x+e_y\vartheta g/\vartheta y=2(e_x+e_y)(x+y)$

$\delta_f/\delta_g=e_f/e_g*g/f=0.25$

تمرین چهارم

$f(X)=\ln(X); e_f\leq e_x\vartheta f/\vartheta X; \vartheta\ln(X)=(1/X); e_f\leq e_x(1/X)=\delta_x$

تمرین پنجم

$(5.5)_{10}=(101.1)_2=1.011*2^2; (sign)(\exp)(mantiss); (0)(10, need\ bias)(011);$

$next\ num=1.01100001*2^2=(0)(10, need\ bias)(01100001); 1+x+8=16; x=7\ bit$ [5]

$1\, bit\, sign, 7\, bit\, \exp, 8\, bit\, mantiss$

$(6.8)_{10}=(110.11001100\ldots)_2=1.1011001100\ldots*2^2;$

$smaller\ estimation=(0)(10\, need\, bias)(10110011)=6.796875, \delta=0.003125\leq 2^{-8}$

$bigger\ estimation=(0)(10\, need\, bias)(10110100)=6.8125, \delta=0.0125\leq 2^{-6}$

تمرین ششم

فرض را براین می‌گیریم که کاربر بتواند هر مقدار دلخواه را به عنوان ورودی تابع وارد کند به عنوان مثال کاربر بتواند مقدار ۱/۳ را به عنوان ورودی در نظر گیرد. با توجه به اینکه مجبوریم مقدار ورودی کاربر را تقریب بزنیم، به عنوان مثال ۱/۳ را با خطای حدی $1/2*10^{-4}$ در نظر بگیریم داریم :

$\ln(x)=2\sum((1-x)/(1+x))^{(2n-1)}/(2n-1); 1\leq n\leq\infty$ [4]

$should\ be: 2((1-x)/(1+x))^{(2n-1)}/(2n-1)\leq 1/2*10^{-4}$ [3]

کد پایتون مربوط به این بخش در ضمیمه این سند در attachments/part6.py و همچنین در صفحه پنجم آمده است.

python code to calculate ln(x)

```
# getting input
x = float(input("Please, enter an input to compute natural logarithm:   "))
# limiting decimal points
x = float("{0:.4f}".format(x))


def ln(x, n):
    result = 0
    for i in range(1, n, 1):
        result += (((x - 1) / (x + 1)) ** (2 * i - 1)) / (2 * i - 1)
    #     thanks to http://math2.org/math/expansion/log.html for taylor
series
    #     enough to calculate to 5 decimal points!

    result = float("{0:.4f}".format(2 * result))
    return result


def find_optimal_n(x):
    n = 1
    while (((x - 1) / (x + 1)) ** (2 * n - 1)) / (2 * n - 1) > 0.000025:
        n += 1
    return n


result = ln(x, find_optimal_n(x))
print(result)# getting input
x = float(input("Please, enter an input to compute natural logarithm:   "))
# limiting decimal points
x = float("{0:.4f}".format(x))


def ln(x, n):
    result = 0
    for i in range(1, n, 1):
        result += (((x - 1) / (x + 1)) ** (2 * i - 1)) / (2 * i - 1)
    #     thanks to http://math2.org/math/expansion/log.htm for taylor
series
    #     enough to calculate to 5 decimal points!

    result = float("{0:.4f}".format(2 * result))
    return result


def find_optimal_n(x):
    n = 1
    while (((x - 1) / (x + 1)) ** (2 * n - 1)) / (2 * n - 1) > 0.000025:
        n += 1
    return n


result = ln(x, find_optimal_n(x))
print(result)
# test time!
from math import log

# 2.7 as input
assert log(2.7) - 0.0001 < ln(2.7, find_optimal_n(2.7))
assert log(2.7) + 0.0001 > ln(2.7, find_optimal_n(2.7))
# test time!
from math import log

# 2.7 as input
assert log(2.7) - 0.0001 < ln(2.7, find_optimal_n(2.7))
assert log(2.7) + 0.0001 > ln(2.7, find_optimal_n(2.7))
```

منابع

[1] : https://blog.minitab.com/blog/real-world-quality-improvement/accuracy-vs-precision-whats-the-difference

[2] : نیکوکار، محاسبات عددی، صفحه ۷

[3] : نیکوکار، محاسبات عددی، صفحه ۱۵

[4] : http://math2.org/math/expansion/log.htm

[5] : https://www.youtube.com/watch?v=tx-M_rqhuUA