The Chinese University of Hong Kong, Shenzhen
SDS · School of Data Science
Andre Milzarek · Fall Semester 2024-25

DDA 3005 — Numerical Methods

Exercise Sheet Nr.: 1

Name: 杨景兰 Jinglan Yang Student ID: 121090699

In the creation of this solution sheet, I worked together with:

Name: ________ Student ID: ________

Name: ________ Student ID: ________

Name: ________ Student ID: ________

For correction:

Exercise						$\sum$
Grading						

Problem 1 (Floating-Point Numbers and Sample Variance): (approx. *25* pts)
In this exercise, we want to compute the (unbiased) sample variance of a sequence of numbers $\{x_i\}_{i=1,\dots,n}$ using floating-point arithmetics. The sample variance can be calculated via one of the equivalent formulas

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \quad \text{(A)} \qquad \text{or} \qquad \sigma^2 = \frac{1}{n-1}\left[\sum_{i=1}^{n} x_i^2 - n\bar{x}^2\right] \quad \text{(B)},$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ denotes the associated sample mean.

Here, we consider the numbers $x_1 = 2021$, $x_2 = 2022$, $x_3 = 2023$, and $x_4 = 2024$ with $n = 4$. Calculate the sample variance σ^2 via the two formulas (A) and (B) for $\{x_i\}_{i=1,\dots,4}$ using a decimal, normalized floating-point system with $\beta = 10$, $p = 6$, $U = -16$, $L = 16$, and rounding to nearest. You can assume that arithmetic floating-point operations are executed in order and that square operations have priority, i.e.,

$$\sum_{i=1}^{4} x_i \approx ((x_1 \oplus x_2) \oplus x_3) \oplus x_4, \quad n\bar{x}^2 \approx n \odot (\bar{x} \odot \bar{x}), \quad \dots \quad \text{etc.}$$

Provide detailed calculations and steps that illustrate your derivation and results.

with $\beta=10$, $p=6$, $U=-16$, $L=16$:

$x_1 = 2.02100 \times 10^3$, $x_2 = 2.02200 \times 10^3$, $x_3 = 2.02300 \times 10^3$, $x_4 = 2.02400 \times 10^3$

① formulas (A):

$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{1}{n}((x_1 \oplus x_2) \oplus x_3) \oplus x_4$

$= \frac{1}{4}[((2.02100 \times 10^3 \oplus 2.02200 \times 10^3) \oplus 2.02300 \times 10^3) \oplus 2.02400 \times 10^3]$

$= \frac{1}{4}[(4.04300 \times 10^3 \oplus 2.02300 \times 10^3) \oplus 2.02400 \times 10^3]$

$= \frac{1}{4}[(6.06600 \times 10^3) \oplus 2.02400 \times 10^3]$

$= \frac{1}{4} \times 8.09000 \times 10^3$

$= 2.02250 \times 10^3$

$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{1}{3}\sum_{i=1}^{n}(x_i - \bar{x})^2$

$= \frac{1}{3}[((2.02100 \times 10^3 - 2.02250 \times 10^3)^2 \oplus (2.02200 \times 10^3 - 2.02250 \times 10^3)^2) \oplus (2.02300 \times 10^3 - 2.02250 \times 10^3)^2 \oplus (2.02400 \times 10^3 -$
$2.02250 \times 10^3)^2]$

$= \frac{1}{3}[(0.00150 \times 10^3)^2 \oplus (0.00050 \times 10^3)^2 \oplus (0.00050 \times 10^3)^2 \oplus (0.00150 \times 10^3)^2]$

$= \frac{1}{3} \times [2.25000 \oplus 0.25000 \oplus 0.25000 \oplus 2.25000]$

$= \frac{1}{3} \times (2.50000 \oplus 0.25000 \oplus 2.25000)$

$= \frac{1}{3} \times (2.75000 \oplus 2.25000)$

$= \frac{1}{3} \times 5.00000$

$= 1.66667 \times 10^0$

② formulas (B):

$n\bar{x}^2 = n \odot (\bar{x} \odot \bar{x}) = 4 \odot (2.02250 \times 10^3 \odot 2.02250 \times 10^3) = 4 \odot 4.09051 \times 10^6 = 1.63620 \times 10^7$

$\sum_{i=1}^{n} x_i^2 = (2.02100 \times 10^3 \odot 2.02100 \times 10^3) \oplus (2.02200 \times 10^3 \odot 2.02200 \times 10^3) \oplus (2.02300 \times 10^3 \odot 2.02300 \times 10^3) \oplus (2.02400 \times 10^3 \odot 2.02400 \times 10^3)$

$= (4.08444 \times 10^6) \oplus (4.08848 \times 10^6) \oplus (4.09253 \times 10^6) \oplus (4.09658 \times 10^6)$

$= 8.17292 \times 10^6 \oplus (4.09253 \times 10^6) \oplus (4.09658 \times 10^6)$

$= 1.22654 \times 10^7 \oplus (4.09658 \times 10^6)$

$= 1.63620 \times 10^7$

$\sigma^2 = \frac{1}{n-1}[\sum_{i=1}^{n} x_i^2 - n\bar{x}^2] = \frac{1}{3}[1.63620 \times 10^7 - 1.63620 \times 10^7] = \frac{1}{3} \times 0 = 0$

Problem 2 (Plotting Polynomials): (approx. *25* pts)
We consider the polynomial function $p : \mathbb{R} \to \mathbb{R}$, $p(x) = (x-3)^8$. The polynomial p has the explicit representation

$$p(x) = x^8 - 24x^7 + 252x^6 - 1512x^5 + 5670x^4 - 13608x^3 + 20412x^2 - 17496x + 6561. \quad (1)$$

a) Plot the function p for $x = 2.920, 2.921, \dots, 3.080$ using the representation shown in (1). Discuss whether your plot is an accurate representation of the mapping p.

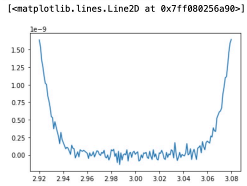
b) Recreate the plot — now using the compact formula $p(x) = (x-3)^8$.

Compare the results obtained in part a) and b) and discuss your observations. Can you explain the observed effects?

(a)

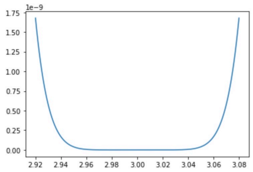
```
x = np.arange(2.920, 3.080, 0.001)
y = x**8-24*(x**7)+252*(x**6)-1512*(x**5)+5670*(x**4)-13608*(x**3)+20412*(x**2)-17496*x+6561
plt.plot(x, y)
```

```
[<matplotlib.lines.Line2D at 0x7ff080256a90>]
```



The plot is not an accurate representation of the mapping p, since it have some floating-point rounding loss, so it is not explicitly accurate.

(b)
```python
x = np.arange(2.920, 3.080, 0.001)
y = (x-3)**8
plt.plot(x, y)
```
[<matplotlib.lines.Line2D at 0x7ff0680443d0>]



The result in (b) is more accurate.

Reason: when calculate $p(x)$ when $x \to 3$, $p(x) = (x-3)^8$ will result in less floating point rounding error than explicitly calculating $p(x)$ in form of (a).

Problem 3 (A "Shaky" Recursion): (approx. *25* pts)

We consider the recursion

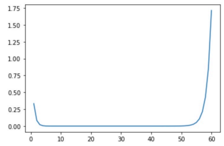
$$x_{k+1} = \frac{9}{4}x_k - \frac{1}{2}x_{k-1} \quad k = 2, 3, \ldots \tag{2}$$

with initial values $x_1 = \frac{1}{3}$ and $x_2 = \frac{1}{12}$.

a) Write a `MATLAB` or `Python` program to compute the first 60 elements of the sequence $\{x_k\}_{k=1,\ldots,60}$ using (2). Create a semilog plot of the obtained values as a function of k.

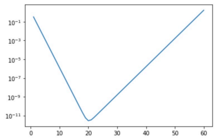
```python
# Q3.a.
```
```python
x=[x for x in range(1,61)]
```
```python
y=[]
for i in range(60):
    if i==0:
        yValue= 1/3
    elif i==1:
        yValue=1/12
    else:
        yValue=(9/4)*y[i-1]-(1/2)*y[i-2]
    y.append(yValue)
```

```python
plt.plot(x, y)
```
[<matplotlib.lines.Line2D at 0x7ff0b0708310>]



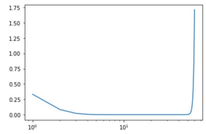
normal plot

```python
plt.semilogy(x,y)
```
[<matplotlib.lines.Line2D at 0x7ff05006fa60>]



semilogy plot

```python
plt.semilogx(x,y)
```
[<matplotlib.lines.Line2D at 0x7ff0907f6160>]

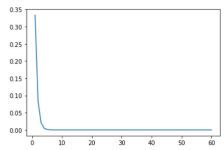


semilogx plot

b) The exact solution to the difference recursion (2) is given by $x_k = \frac{1}{3}4^{1-k}$. Plot the mapping $k \mapsto \frac{1}{3}4^{1-k}$ for $k = 1, \ldots, 60$ and compare the behavior of the obtained graph with the results from part a). What is your observation?

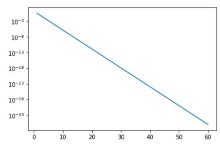
```python
x=[x for x in range(1,61)]
y =[]
for i in x:
    y.append(1/3*(4**(1-i)))
```

```python
plt.plot(x,y)
```
[<matplotlib.lines.Line2D at 0x7ff0b0732a30>]



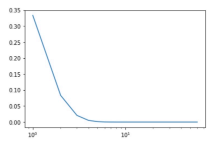
normal plot

```python
plt.semilogy(x,y)
```
[<matplotlib.lines.Line2D at 0x7ff0907fc250>]



semilogy plot

```python
plt.semilogx(x,y)
```
[<matplotlib.lines.Line2D at 0x7ff0b07543d0>]



semilogx plot

Observation: ① in recursion sequence (2), as k going larger, the resulting value first decreases then increases

② using exact solution form, the resulting value always decreases as k going larger.

c) Next, let us consider the recursion (2) with the general initial points $x_1 = a$ and $x_2 = b$, where $a, b \in \mathbb{R}$. Show that the general solution to this problem is given by

$$x_k = \frac{2^{k-1}}{7}(4b - a) + \frac{4^{2-k}}{7}(2a - b) \tag{3}$$

(e.g., through induction over k).

Assume $X_k = \frac{2^{k-1}}{7}(4b-a) + \frac{4^{2-k}}{7}(2a-b)$ is true:

① $X_1 = \frac{2^0}{7}(4b-a) + \frac{4}{7}(2a-b) = a$, $X_2 = \frac{2}{7}(4b-a) + \frac{4^0}{7}(2a-b) = b$

$X_3 = \frac{2^2}{7}(4b-a) + \frac{4^{-1}}{7}(2a-b) = \frac{9}{4}b - \frac{1}{2}a = \frac{9}{4}X_2 - \frac{1}{2}X_1$ (The general solution is true when $k=1,2,3$)

② Assume the general solution works for $k = a-1$, $k = a-2$:

$X_{a-1} = \frac{2^{a-2}}{7}(4b-a) + \frac{4^{3-a}}{7}(2a-b)$

$X_{a-2} = \frac{2^{a-3}}{7}(4b-a) + \frac{4^{4-a}}{7}(2a-b)$

For $k = a$:

$X_a = \frac{9}{4}X_{a-1} - \frac{1}{2}X_{a-2}$

$= \left(\frac{9}{4}\cdot\frac{2^{a-2}}{7} - \frac{1}{2}\cdot\frac{2^{a-3}}{7}\right)(4b-a) + \left(\frac{9}{4}\cdot\frac{4^{3-a}}{7} - \frac{1}{2}\cdot\frac{4^{4-a}}{7}\right)(2a-b)$

$= \frac{2^{a-1}}{7}(4b-a) + \frac{4^{2-a}}{7}(2a-b)$ (The general solution is true for all k)

Therefore, $X_k = \frac{2^{k-1}}{7}(4b-a) + \frac{4^{2-k}}{7}(2a-b)$ $(X_1 = a, X_2 = b)$ is proved.

d) Can you explain the numerical effects and behavior observed in part a) and b)?

The error in (a) is mainly caused by computing a small quantity as difference, since the rounding error dominates the result.
Using formula in (b) does not have the problem caused by cancellation.

*e) Can you provide a direct derivation of the (mysterious) formula (3) from the recursion (2) (with $x_1 = a$ and $x_2 = b$), i.e., without knowing the specific form of (3) already?

$$X_{k+1} = \frac{9}{4}X_k - \frac{1}{2}X_{k-1}$$
$$\Leftrightarrow X_{k+1} - \frac{1}{4}X_k = 2X_k - \frac{1}{2}X_{k-1}$$
$$\Leftrightarrow X_{k+1} - \frac{1}{4}X_k = 2\left(X_k - \frac{1}{4}X_{k-1}\right)$$
$$\Rightarrow X_k - \frac{1}{4}X_{k-1} = 2\left(X_{k-1} - \frac{1}{4}X_{k-2}\right)$$
$$= 2^2\left(X_{k-2} - \frac{1}{4}X_{k-3}\right)$$
$$= 2^3\left(X_{k-3} - \frac{1}{4}X_{k-4}\right)$$
$$= \cdots$$
$$= 2^{k-2}\left(X_2 - \frac{1}{4}X_1\right)$$
$$\Rightarrow X_k = \frac{1}{4}X_{k-1} + 2^{k-2}\left(X_2 - \frac{1}{4}X_1\right)$$
$$= 2^{-2}X_{k-1} + 2^{k-2}\left(b - \frac{1}{4}a\right)$$
$$= 2^{-2}X_{k-1} + 2^{k-4}(4b-a)$$
$$= 2^{-2}\left[2^{-2}X_{k-2} + 2^{k-5}(4b-a)\right] + 2^{k-4}(4b-a)$$
$$= 2^{-4}X_{k-2} + 2^{k-7}(4b-a) + 2^{k-4}(4b-a)$$
$$= 2^{-4}\left[2^{-2}X_{k-3} + 2^{k-6}(4b-a)\right] + 2^{k-7}(4b-a) + 2^{k-4}(4b-a)$$
$$= 2^{-6}X_{k-3} + 2^{k-10}(4b-a) + 2^{k-7}(4b-a) + 2^{k-4}(4b-a)$$
$$= \cdots$$
$$= 2^{4-2k}X_2 + (\underbrace{2^{k-4} + 2^{k-7} + 2^{k-10} + \cdots}_{k-2})(4b-a)$$
$$= 2^{4-2k}b + \frac{2^{k-4}\left[1-(2^{-3})^{k-2}\right]}{1-2^{-3}}(4b-a)$$
$$= 2^{4-2k}b + \frac{2^{k-1}\left[1-2^{6-3k}\right]}{7}(4b-a)$$
$$= \frac{2^{k-1}}{7}(4b-a) + 2^{4-2k}b - \frac{2^{5-2k}}{7}(4b-a)$$
$$= \frac{2^{k-1}}{7}(4b-a) + \frac{7\cdot 2^{4-2k} - 8\cdot 2^{4-2k}}{7}\cdot b + \frac{2^{4-2k}}{7}\cdot 2a$$
$$= \frac{2^{k-1}}{7}(4b-a) + \frac{4^{2-k}}{7}(2a-b)$$

Hence, $X_k = \frac{2^{k-1}}{7}(4b-a) + \frac{4^{2-k}}{7}(2a-b)$ with $X_1 = a$, $X_2 = b$.

Problem 4 (Squares and Error Analysis): (approx. *25* pts)
In this problem, we want to analyze numerical errors when calculating the difference of two square numbers. Specifically, for given $a, b \in \mathbb{R}$, we consider the function

$$y = f(a,b) := a^2 - b^2$$

and the associated floating-point approximation

$$\hat{y} = \hat{f}(a,b) := \mathrm{fl}(f(a,b)) = (a \odot a) \ominus (b \odot b).$$

Throughout this exercise and to simplify the analysis, we assume that a, b are machine numbers.

a) Show that the absolute forward error (total error) can be estimated via

$$|\hat{y} - y| \le \mathcal{O}(\varepsilon_{\mathrm{mach}}) \max\{a^2, b^2\}$$

(for a sufficiently small machine precision $\varepsilon_{\mathrm{mach}}$).

– You can assume that the underlying floating-point system satisfies the IEEE-standard 754, i.e., we have $u \circledast v = \mathrm{fl}(u * v) = (1+\varepsilon)(u * v)$ for all machine numbers u, v, every arithmetic operation $* \in \{+, -, \cdot, /, \sqrt{\cdot}\}$, and some ε with $|\varepsilon| \le \varepsilon_{\mathrm{mach}}$.

$$|\hat{y} - y| = |[(a\odot a)\ominus(b\odot b)] - (a^2-b^2)|$$
$$= |[(1+\varepsilon_1)a^2 \ominus (1+\varepsilon_2)b^2] - (a^2-b^2)|, \quad |\varepsilon_1| \le \varepsilon_{mach},\ |\varepsilon_2| \le \varepsilon_{mach}$$
$$= |(1+\varepsilon_3)[(1+\varepsilon_1)a^2 - (1+\varepsilon_2)b^2] - (a^2-b^2)|, \quad |\varepsilon_3| \le \varepsilon_{mach}$$
$$= |(\varepsilon_1+\varepsilon_3+\varepsilon_1\varepsilon_3)a^2 - (\varepsilon_2+\varepsilon_3+\varepsilon_2\varepsilon_3)b^2|$$
$$= |\varepsilon_4 a^2 - \varepsilon_5 b^2|, \quad \varepsilon_4 = \varepsilon_1+\varepsilon_3+\varepsilon_1\varepsilon_3, \quad \varepsilon_5 = \varepsilon_2+\varepsilon_3+\varepsilon_2\varepsilon_3$$
$$\le \max\{\varepsilon_4, \varepsilon_5\}\cdot\max\{a^2, b^2\}$$

$\because \varepsilon_4 = \varepsilon_1 + \varepsilon_3 + \varepsilon_1\varepsilon_3 \le 2\varepsilon_{mach} + \varepsilon_{mach}^2 \le O(mach)$, $\varepsilon_5 = \varepsilon_2 + \varepsilon_3 + \varepsilon_1\varepsilon_3 \le 2\varepsilon_{mach} + \varepsilon_{mach}^2 \le O(mach)$

$\therefore |\hat{y} - y| \le O(\varepsilon_{mach}) \cdot \max\{a^2, b^2\}$

b) Consider a decimal, normalized floating-point system with $\beta = 10$, $p = 4$, $U = -9$, $L = 9$, and rounding to nearest. Construct an example and choose a, b such that

$$\frac{|\hat{y} - y|}{|y|} > \frac{1}{2}.$$

$\frac{|\hat{y} - y|}{|y|} > \frac{1}{2}$

$\Leftrightarrow |\hat{y} - y| > \frac{1}{2}|y|$

$\Leftrightarrow |(a \odot a) \ominus (b \odot b) - (a^2 - b^2)| > \frac{1}{2}|a^2 - b^2|$

Choose a, b such that using the effect of rounding:

$a = 1.00003$ $\quad b = 1.00001$

$\Rightarrow$ in floating-point system with $\beta = 10, p = 4, U = -9, L = 9$:

$\Rightarrow |\hat{y} - y| = |(a \odot a) \ominus (b \odot b) - (a^2 - b^2)| = |(1.000 \times 10^0 \odot 1.000 \times 10^0) \ominus (1.000 \times 10^0 \odot 1.000 \times 10^0) - (a^2 - b^2)| = |a^2 - b^2| > \frac{1}{2}(a^2 - b^2)$

c) Can you design an alternative (more accurate) algorithm $\tilde{y} = \tilde{f}(a, b)$ satisfying

$$\frac{|\tilde{y} - y|}{|y|} = \frac{|\tilde{f}(a, b) - f(a, b)|}{|f(a, b)|} \approx \mathcal{O}(\varepsilon_{\text{mach}})$$

for all machine numbers a, b? Provide detailed explanations! Repeat the calculations for the example in part b) using the alternative algorithm $\tilde{f}$. What are your observations?

$\tilde{y} = \tilde{f}(a, b) = fl(f(a, b)) = fl((a+b)(a-b)) = (a \oplus b) \odot (a \ominus b)$

$\Rightarrow |\tilde{y} - y| = |(a \oplus b) \odot (a \ominus b) - (a^2 - b^2)|$

$= |[(1+\varepsilon_6)(a+b) \odot (1+\varepsilon_7)(a-b)] - (a^2 - b^2)|$, $|\varepsilon_6| \le \varepsilon_{mach}$, $|\varepsilon_7| \le \varepsilon_{mach}$

$= |(1+\varepsilon_8)(1+\varepsilon_6)(1+\varepsilon_7)(a^2 - b^2) - (a^2 - b^2)|$, $|\varepsilon_8| \le \varepsilon_{mach}$

$= |(\varepsilon_6 + \varepsilon_7 + \varepsilon_8 + \varepsilon_6\varepsilon_7 + \varepsilon_7\varepsilon_8 + \varepsilon_6\varepsilon_7\varepsilon_8)(a^2 - b^2)|$

$\Rightarrow \frac{|\tilde{y} - y|}{|y|} = \frac{|(\varepsilon_6 + \varepsilon_7 + \varepsilon_8 + \varepsilon_6\varepsilon_7 + \varepsilon_7\varepsilon_8 + \varepsilon_6\varepsilon_7\varepsilon_8)(a^2 - b^2)|}{|a^2 - b^2|}$

$= |(\varepsilon_6 + \varepsilon_7 + \varepsilon_8 + \varepsilon_6\varepsilon_7 + \varepsilon_7\varepsilon_8 + \varepsilon_6\varepsilon_7\varepsilon_8)|$

$\le 3\varepsilon_{mach} + 2\varepsilon_{mach}^2 + \varepsilon_{mach}^3$

$\approx O(\varepsilon_{mach})$

Repeat in (b): $\quad a = 1.00003$, $\quad b = 1.00001$

$\frac{|\tilde{y} - y|}{|y|} = \frac{|(1.000 \oplus 1.000) \cdot (1.000 \ominus 1.000) - (a^2 - b^2)|}{|a^2 - b^2|} = \frac{|(a^2 - b^2)|}{|a^2 - b^2|} = 1 > \frac{1}{2}$ The result does not change.