89

Q1.a

$$Q_B = p/B^k$$

$Q_{10} = p/10^k$ $Q_2 = p/2^k$ We use the therom: if $x \in Q_2$, then x must also in Q_{10}

1. $\frac{3}{16} = \frac{3}{2^4}$, so $\frac{3}{16} \in Q_2, Q_{10}$

2. $\frac{91}{200} = \frac{7\times13}{2^3\times5^2}$ 5 is a coprime of 2, so $91/200 \notin Q_2$ $\frac{91}{200} = \frac{455}{10^3}$, so $\frac{91}{200} \in Q_{10}$

6

3. $\frac{8}{60} = \frac{2}{3\times5}$, 3 is a coprime of 2 and 10 so $\frac{8}{60}$ so $\frac{8}{60} \notin Q_2, Q_{10}$

4. $\frac{24}{60} = \frac{2}{5} = \frac{4}{10}$, so $\frac{24}{60} \in Q_{10}$, but 5 is a coprime of 2, so $\frac{24}{60} \notin Q_2$

5. $\frac{99}{250} = \frac{3^2\times11}{2\times5^3} = \frac{396}{10^3}$, $\frac{99}{250} \in Q_{10}$ but 5 is a coprime of 2, so $\frac{99}{250} \notin Q_2$

In sum, Q_{10}: 3/16, 91/200, 24/60, 99/250. Q_2: 3/16.

Q1.b First we find the binary representation of 0.07:

$$\begin{aligned}
0.07 \times 2 &= 0.14 \longrightarrow 0 \\
0.14 \times 2 &= 0.28 \longrightarrow 0 \\
0.28 \times 2 &= 0.56 \longrightarrow 0 \\
0.56 \times 2 &= 1.12 \longrightarrow 1 \\
0.12 \times 2 &= 0.24 \longrightarrow 0 \\
0.24 \times 2 &= 0.48 \longrightarrow 0 \\
0.48 \times 2 &= 0.96 \longrightarrow 0
\end{aligned}$$

8

The binary representation $0.07 = (0.0001000)_2$ rounding by truncation to the same first (significant) three (nonzero) binary digits gives $\phi(0.07) = (0.000100)_2 = 0.0625$

Q2.a

0.4 has one significant digits so $0.4 \in F_{10}(2)$.

$0.4 = 0.(\dot{0}1\dot{1}0)_2$, which have more than 6 significant binary digits, so $0.4 \notin F_2(6)$

0.25 have two significant digits, so $0.25 \in F_{10}(2)$.

$0.25 = 2^{-2}$, so $0.25 \in F_2(6)$

0.125 has three significant digits so $0.125 \notin F_{10}(2)$.

$0.125 = 1 \times 2^{-3}$ so $0.125 \in F_2(6)$

0.0125 has three significant digits so $0.125 \notin F_{10}(2)$.

$0.0125 = (0.00000\dot{0}11\dot{0})_2$ which have more than 6 significant binary digits, so $0.0125 \notin F_2(6)$

0.0625 has three significant digits so $0.0625 \notin F_{10}(2)$.

$0.0625 = 1 \times 2^{-4}$ so $0.0625 \in F_2(6)$

0.0025 have two significant digits, so $0.0025 \in F_{10}(2)$.

$0.0025 = (0.000000001010001...)_2$ which have more than 6 significant binary digits, so $0.0025 \notin F_2(6)$

0.625 has three significant digits so $0.625 \notin F_{10}(2)$.

$0.625 = 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$ so $0.625 \in F_2(6)$

6.25 has three significant digits so $6.25 \notin F_{10}(2)$.

6

$6.25 = 1 \times 2^{2} + 1 \times 2^{1} + 0 \times 2^{0} + 0 \times 2^{-1} + 1 \times 2^{-2}$ so $6.25 \in F_2(6)$

$$F_{10}(2) : 0.4, 0.25, 0.0025$$
$$F_2(6) : 0.25, 0.125, 0.0625, 0.625, 6.25$$

Q2.b

{1/2, 3/4, 1} in $[1/2, 1]$ are in intersection of $F_{10}(2)$ and $F_2(6)$

Q2.c

6

$$F_B(2) = x| \pm (c_1 B^{-1+e} + c_2 B^{-2+e})$$

For $\epsilon = \frac{1}{2} B^{-3}$ define $x = B^{-1} + B^{-2} + B^{-3} \in R$ if F_B(2) is a dense set then there is s in $F_B(2)$ s.t. $|s - x| < \epsilon$ ie. $f(B, e) = |B^{-1} + B^{-2} + B^{-3} - c_1 B^{-1+e} + c_2 B^{-2+e}| < \epsilon$ but $f(B, e)_{min} = B^{-3}$ This is a contradiction. So $F_B(2)$ is not a dense set. Hence $F_{10}(2)$ $F_7(2)$ are not dense sets.

$F_B(2)$ do not include $\{x = \pm \sum_{j=1}^{t} c_j 7^{-j+e}\}^{|t>2}$ i.e. the integer with more than 3 digits. Hence $F_1 0(2)$ do not include $\{x = \pm \sum_{j=1}^{t} c_j 10^{-j+e} \mid t > 2\}$ $F_7(2)$ do not include $\{x = \pm \sum_{j=1}^{t} c_j 7^{-j+e} \mid t > 2\}$

Q3.a

$$u_0 = 3.4$$
$$u_1 = 0.43$$
$$u_2 = u_0 u_1$$

$$u_0 = (1 + \delta_0)3.4$$
$$u_1 = (1 + \delta_1)0.43$$
$$u_2 = (1 + \delta_2)u_0 u_1$$

By manual multiplication exact product of 3.4 and 0.43 is 1.462

In [31]:

```
#Q3.a
#apporxiamte product of 3.4*0.43
x_exact = Decimal("3.4")
y_exact = Decimal("0.43")
ans_exact = x_exact * y_exact
print("the exact answer for 3.4 is :{:10.53f}".format(ans_exact))

x=3.4
y=0.43
ans=x*y
print("the numerical answer for 3.4 is :{:10.53f}".format(ans))

print("the relative rounding error for 3.4 is :", abs((Decimal(x)-x_exact)/x_exact))
print("the relative rounding error for 0.43 is :", abs((Decimal(y)-y_exact)/y_exact))
print("the total relative rounding error for 3.4*0.43 is :", abs((Decimal(ans)-ans_exact)/ans_exact)
```

```
the exact answer for 3.4 is :1.46200000000000000000000000000000000000000000000000000
the numerical answer for 3.4 is :1.46199999999999996624922005139524117112159729003906250
the relative rounding error for 3.4 is : 2.612289469706250683349721572E-17
the relative rounding error for 0.43 is : 1.549148406453706800591113956E-17
the total relative rounding error for 3.4*0.43 is : 2.308534880205523859704405110E-17
```

10

The optimal implementation requires

$$|\delta| \le \epsilon \le \frac{1}{2B^{t-1}} = \frac{1}{2 \times 2^{52}} \approx 1.11 \times 10^{-16}$$

, We can see that the relative error for x, y and $x \times y$ all satisfy the above condition. The Python implementation of this product is optimal for this data.

Q3.b

$$u_0 = (1 + \delta_0)x_1$$
$$u_1 = (1 + \delta_1)x_2$$
$$u_2 = (1 + \delta_{u2})u_0 u_1$$

For optimal implementation, the relative error by rounding for each number is unique.

$$v_0 = (1 + \delta_1)x_2$$
$$v_1 = (1 + \delta_0)x_1$$
$$v_2 = (1 + \delta_{v2})v_0 v_1$$

because

$$u_0 u_1 = (1 + \delta_0)(1 + \delta_1)x_1 x_2 = v_0 v_1$$

. so $\delta_{u2} = \delta_{v2}$ Hence

$$u_2 = v_2$$

8

for an optimal implementation of multiplication, the rounding error is the same for the (numerical) products x1 $*$ x2 and for x2 $*$ x1.

Q3.c

$|\delta| <= 0.5B^{-t+1}$

$F_{10}(2)$: $|\delta| <= 0.5 \times 10^{-1} = 0.05$

$F_2(7)$: $|\delta| <= 0.5 \times 2^{-6} = 2^{-7}$

Comparing the results: Although the base is smaller, the rounding for longer digits is more precise.

Q4.a

$$u_1 = x_1$$

$$\dots$$

$$u_N = x_N$$

$$u_{N+1} = u_1 + u_2$$

$$u_{N+2} = u_{N+1} + u_3$$

$$\dots$$

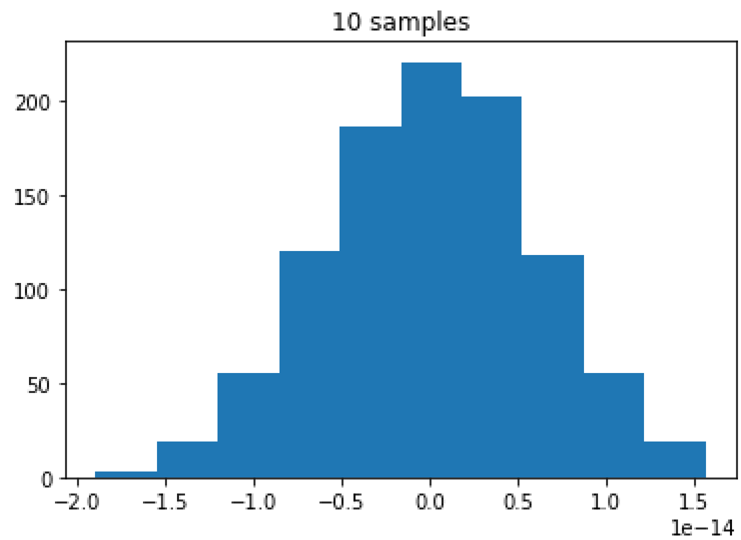
$$u_{2N-1} = u_{2N-2} + u_N$$

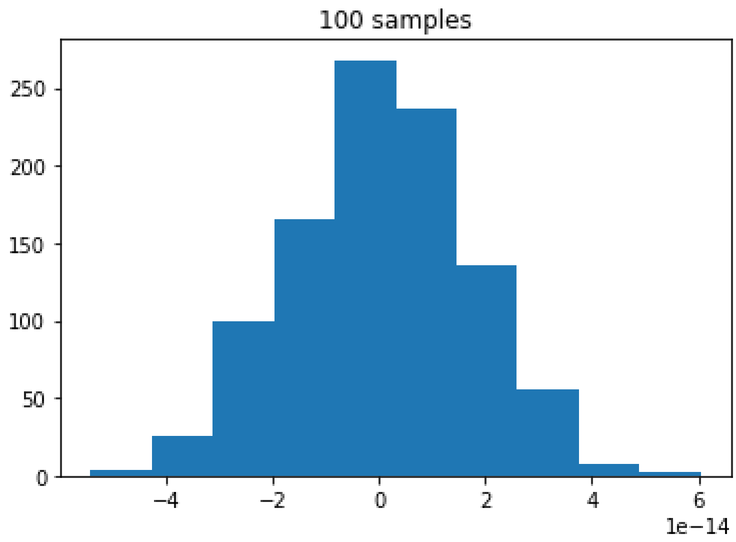
$$u_{2N} = u_{2N-1}/N$$

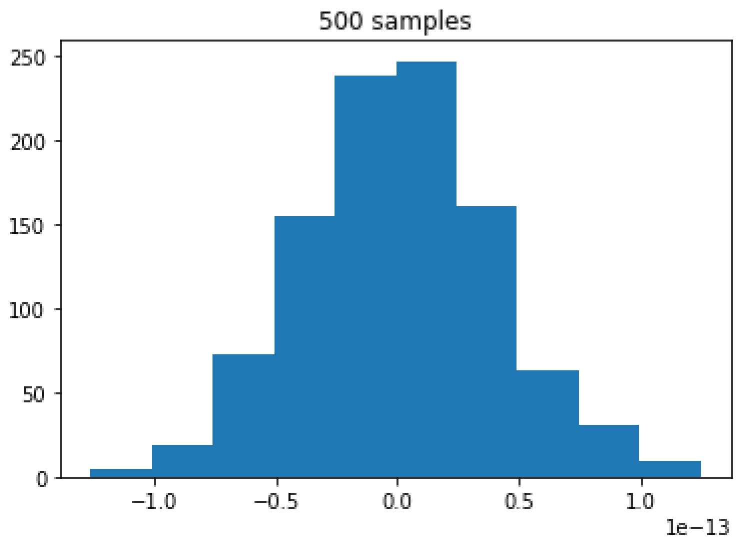
$$\begin{gathered} v_1 = (1+\delta_1)x_1 \\ \dots \\ v_N = (1+\delta_N)x_N \\ v_{N+1} = (1+\delta_{N+1})(v_1 + v_2) \\ v_{N+2} = (1+\delta_{N+2})(v_{N+1} + v_3) \\ \dots \\ v_{2N-1} = (1+\delta_{2N-1})(v_{2N-2} + v_N) \\ v_{2N} = (1+\delta_{2N})v_{2N-1}/N \end{gathered}$$

In [17]:

```
# a code to calculate error
import matplotlib.pylab as plt
import numpy as np
from decimal import Decimal, getcontext


def sum(data, delta):
    n= len(data)
    v=np.zeros(n)
    s= np.zeros(n-1)
    for i in range(n):
        v[i]=(1+delta[i])*data[i]

    for i in range(n-1):
        if i==0:
            s[i] = (1+delta[n+i])*(v[0]+v[1])
        else:
            s[i] = (1+delta[n+i])*(s[i-1]+v[i+1])

    ave=(1+delta[2*n-1])*s[-1]/n

    return ave


#error hist
t=1000 #1000 trial times
n_list = [10, 100, 500]
for n in n_list:
    error = np.zeros(t)
    epsi = 1e-14
    data = np.random.random(n)
    for k in range(t):
        delta = epsi*(np.random.random(2*n)*2-1)
        error[k] = sum(data, delta)-sum(data, np.zeros(2*n))
    plt.hist(error)
    plt.title(r"{} samples".format(n))
    plt.show()
```







10

Q4.b

$$
\begin{aligned}
u_1 &= x \\
u_2 &= cos(u_1) \\
u_3 &= u_2^2 \\
u_4 &= cos(2u_1) \\
u_5 &= 2u_3 - 1 \\
u_6 &= u_5 - u_4
\end{aligned}
$$

$$
\begin{aligned}
v_1 &= (1+\delta_1)x \\
v_2 &= (1+\delta_2)cos(v_1) \\
v_3 &= (1+\delta_3)v_2^2 \\
v_4 &= (1+\delta_4)cos(2v_1) \\
v_5 &= (1+\delta_5)(2v_3 - 1) \\
v_6 &= (1+\delta_6)(v_5 - v_4)
\end{aligned}
$$

8

So

$$
\begin{aligned}
v_6 &= (1+\delta_6)[(1+\delta_5)(2(1+\delta_3)(1+\delta_2)^2 cos^2((1+\delta_1)x) - 1) - (1+\delta_4)cos(2(1+\delta_1)x)] \\
&\approx 2(1+2\delta_2+\delta_3+\delta_5+\delta_6)cos^2((1+\delta_1)x) - (1+\delta_5+\delta_6) - (1+\delta_4+\delta_6)cos(2(1+\delta_1)x)
\end{aligned}
$$

Using

$$cos^2((1+\delta_1)x) \approx cos^2(x) - 2sin(x)cos(x)\delta_1 x$$

$$cos[2(1+\delta_1)x] = cos(2x) - 2sin(2x)\delta_1 x = 2cos^2(x) - 1 - 2sin(2x)\delta_1 x$$

We get:

$$v_6 - u_6 \approx 2(2\delta_2 + \delta_3 - \delta_4 + \delta_5)cos^2(x) + \delta_4 - \delta_5 \leq 12\epsilon$$

where we use max $|cos(x)| = cos(0) = 1$ for $x \in [0, \pi/2]$ and $|\delta| = \pm\delta \leq \epsilon$

Q5.a

$$\kappa(x) = \sup_{y \neq x} \frac{\|f(y) - f(x)\|/\|f(x)\|}{\|y - x\|/|x\|}$$

f(x)=sin(x) x [0, \pi/4]

$$\kappa(x) = \sup_{y \neq x} \frac{\|sin(y) - sin(x)\|/\|sin(x)\|}{\|y - x\|/|x\|}$$

8

$$= \sup_{y \neq x} \frac{\|sin(y) - sin(x)\|/\|y - x\|}{\|sin(x)\|/|x\|} = 1$$

Q5.b

$f(x) = \sum_{k=1}^{n} x_i$

$$\kappa(x) = \sup_{y \neq x} \frac{|\sum_{k=1}^{n}(y_i - x_i)|/|\sum_{k=1}^{n} x_i|}{\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \Big/ \sqrt{\sum_{i=1}^{n} x_i^2}}$$

$$= \sup_{y \neq x} \frac{nd/|\sum_{i=1}^{n} x_i|}{\sqrt{nd} \Big/ \sqrt{\sum_{i=1}^{n} x_i^2}}$$

$$= \sup_{y \neq x} \frac{\sqrt{n \sum_{i=1}^{n} x_i^2}}{|\sum_{i=1}^{n} x_i|}$$

$$= \sqrt{n}$$

In the last step we use:

$$\sqrt{\sum_{i=1}^{n} x_i^2} \leq \sqrt{(\sum_{i=1}^{n} x_i)^2} = |\sum_{i=1}^{n} x_i|$$

the equality holds when there is only one positive term in $\{x_i\}$.

Q6.a

$$u_1 = x$$
$$u_2 = sin(u_1)$$

with rounding errors

$$v_1 = (1+\delta_1)x$$
$$v_2 = (1+\delta_2)sin(v_1)$$

backward stable model

$$z_1 = (1+\zeta_1)x$$
$$z_2 = sin(z_1)$$

$$z_2 = v_2 = sin(z_1) = (1+\delta_2)sin(v_1)$$
$$z_1 = arcsin((1+\delta_2)sin(v_1)) = arcsin((1+\delta_2)sin((1+\delta_1)x)) = (1+\zeta_1)x$$

$$\zeta_1 = arcsin[(1+\delta_2)sin[(1+\delta_1)x]]/x - 1$$

When $sin(x) \longrightarrow \pm 1$, namely $x\ (\frac{1}{2}+n)\pi\ (1+\delta_2)sin[(1+\delta_1)x]$ may be bigger than 1, which is not the domain of arcsin. ζ_1 then can not take a value and it is not backward stable. so $[a, b]$ can not include the point near $(\frac{1}{2}+n)\pi$.

Q6.b

If $f(x, \delta)$ is backward stable and $f(x, 0)$ is well conditioned with condition number $\kappa(x)$, then there is a $C > 0$ such that the relative error satisfies

$$|e| \leq \kappa(x)\, C\epsilon.$$

3

For all rounding errors δ with $|\delta_k| \leq \epsilon$, since $|\zeta_1| \leq (\frac{\pi}{2x}+1)$,

$$|e| \leq \kappa(x)\frac{\|y-x\|}{\|x\|} \leq \frac{\pi}{2x}+1$$

In [ ]: