\$ Supplement Note 9: Linear and Quadratic Extrapolation for $log(2)$

The *alternating harmonic series* is the *conditionally convergent* series

$$\sum_{k=0}^{\infty} \frac{(-1)^{k-1}}{k} = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} + \cdots = \ln 2 = 0.6931471805599453 \tag{1}$$

Identity (1) is proved in **Example 7.2**, p.7-2 of **Lecture Note**:

$$\ln 2 = \int_1^2 \frac{dx}{x} = \int_0^1 \frac{dt}{1+t} = \int_0^1 (1-t+t^2-t^3+\cdots)dt = 1-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\frac{1}{5}+\cdots \tag{2}$$

For a numerical evaluation of the series, the consecutive alternation terms are combined together to avoid numerical instability. For an even number N, we have

$$S_N \equiv \Big(1 - \frac{1}{2}\Big) + \Big(\frac{1}{3} - \frac{1}{4}\Big) + \Big(\frac{1}{5} - \frac{1}{6}\Big) + \cdots + \Big(\frac{1}{N-1} - \frac{1}{N}\Big) = \sum_{k=1}^{N/2} \frac{1}{2k(2k-1)} \tag{3}$$

As is shown in the python code *LinearQuadraticLog2.py* and listed in Table 1, the convergence of the alternating harmonic series S_N in (3) is slow. Typical results are $S_{10} = 0.6687, S_{100} = 0.69065$, and $S_{1024} = 0.692903$.

Linear Extrapolation

The rate of convergence in the calculation of the series may be improved by making a *linear extrapolation*. Assuming that the sum S_N in (3) has an error proportional to $\frac{1}{N}$, S_N may be expressed as a linear function of $x \equiv \frac{1}{N}$: $S_N \equiv S(x) \equiv a + bx$. As $N \to \infty$, $x \to 0$. *The constant $S(0) = a$ is then the desired value of* $log(2)$. The coefficients a and b can be determined by solving the simultaneous linear equations from the two calculated results S_{N_1}, S_{N_2}:

$$S_{N_1} \equiv S(x_1) \equiv a + bx_1; \quad x_1 \equiv \frac{1}{N_1} \tag{4a}$$

$$S_{N_2} \equiv S(x_2) \equiv a + bx_2; \quad x_2 \equiv \frac{1}{N_2} \tag{4b}$$

N	S_N	error
10	0.6687714031754279	-2.44 e-02
20	0.6808033817926938	-1.23 e-02
30	0.6848832820313465	-8.26 e-03
40	0.6869362400091404	-6.21 e-03
100	0.6906534304818241	-2.49 e-03
300	0.6923145416700924	-8.33 e-04
1024	0.6929030995395841	-2.44 e-04
N_1 , N_2	values from linear extrapolation	error
10, 20	0.6928353604099597	-3.12 e-04
20, 30	0.6930430825086519	-1.04 e-04
30, 40	0.6930951139425222	-5.21 e-05
40, 100	0.6931315574636133	-1.56 e-05
100, 300	0.6931450972642266	-2.08 e-06
300, 1024	0.6931469771098154	-2.03 e-07

Table 1: Series Sums S_N and Values from Linear Extrapolation for $\ln 2$

In particular, the constant a is obtained by eliminating the linear term from Eq.(4):

$$a = \frac{x_1 \, S_{N_2} - x_2 \, S_{N_1}}{x_1 - x_2} \tag{5}$$

Some typical results from linear extrapolation of the alternating harmonic series are shown in Table 1. The rate of convergence of the series is much higher. For example, the result of the linear extrapolation from S_{20}, S_{30} (summing 30 terms only) yields 0.69304, a result better than the one obtained by summing 1024 terms of the series: $S_{1024} = 0.69290$.

Quadratic Extrapolation

A further improvement may be achieved by the *quadratic extrapolation* with $S_N \equiv S(x) \equiv a_0 + a_1 x + a_2 x^2$, $x \equiv \frac{1}{N}$, where additional error terms proportional to $(\frac{1}{N})^2$ are also included. The coefficients a_0, a_1, a_2 can be obtained from three calculated

N_1 , N_2 , N_2	results from quadratic extrapolation	error
10 ,20, 30	0.6931469435579979	-2.37 e-07
20 ,30, 40	0.6931471453763925	-3.52 e-08
30 ,40, 100	0.6931471761155095	-4.44 e-09
40 ,100, 300	0.6931471803104747	-2.49 e-10
100, 300, 1024	0.6931471805563078	-3.64 e-12

Table 2: Approximated values for log(2) from Quadratic Extrapolation

values $S_{N_1}, S_{N_2}, S_{N_3}$:

$$S_{N_i} = S(x_i) \equiv a_0 + a_1 x_i + a_2 x_i^2, \quad x_i \equiv \frac{1}{N_i}, \ i = 1, 2, 3 \ . \tag{6}$$

Here again, *the constant* $a_0 = S(0) \equiv S_\infty$ *is the desired approximation to log*(2). The coefficient a_1 was eliminated as in (5):

$$d_{12} \equiv \frac{x_1 \, S_{N_2} - x_2 \, S_{N_1}}{x_1 - x_2} = a_0 - a_2 x_1 x_2 \tag{7a}$$

$$d_{23} \equiv \frac{x_2 \, S_{N_3} - x_3 \, S_{N_2}}{x_2 - x_3} = a_0 - a_2 x_2 x_3 \tag{7b}$$

The coefficient a_0 may then be obtained by eliminating a_2 in (7) in a similar way:

$$a_0 = \frac{x_3 d_{12} - x_1 d_{23}}{x_3 - x_1} \tag{8}$$

Some typical results are listed in Table 2. The results from the quadratic extrapolation are very close to the exact value.

Generalization to extrapolation using polynomials of higher order is straightforward. However, *truncation errors* due to finite degrees of accuracy for real numbers represented by computers (usually with 64 bits) have to be carefully monitored for extrapolations of higher order. Considering the simplicity of the approach and the accuracy of the results, *linear extrapolation and quadratic extrapolation are highly recommended*.

```
# LinearQuadraticLog2.py
# calculate log2 by linear extrapolation and quadratic extrapolation
```

```
# on the alternating harmonic series log(2)= sum_{n = 1} 1/2n(2n -1)
import numpy as np
nn =[10, 20, 30, 40, 100, 300, 1024]
ni, s, log2, S_N= 0, 0, np.log(2), []
for nf in nn:
    for i in range(ni+1, nf+1):
        fac = 2*i *(2*i -1)
        s +=  1/fac
    S_N.append(s)
    ni = nf
print(' results from alternating harmonic series sum\n n   , log(2), error')
for n,s in zip(nn, S_N): print(n,s, s-log2)
print('log(2) =', log2)
print('linear extrapolation \nn1,n2  , log(2), error')
s0, n0 = S_N[0], nn[0]
n0_1 = 1/n0
for s1, n1 in zip(S_N[1:], nn[1:]):
    n1_1 = 1/n1
    s = (n0_1*s1- n1_1*s0)/(n0_1-n1_1)
    print(n0,n1,s, s-log2)
print('quadratic extrapolation\nn1,n2 ,n3 , log(2), error')
sn1, n1 = S_N[0], nn[0]
sn2, n2 = S_N[1], nn[1]
for sn3, n3 in zip(S_N[2:], nn[2:]):
    x1, x2, x3 = 1/n1, 1/n2, 1/n3
    d12 = (x1*sn2-x2*sn1)/(x1-x2)
    d23 = (x2*sn3-x3*sn2)/(x2-x3)
    a = (x3*d12-x1*d23)/(x3-x1)
    print(n1,n2,n3, a, a-log2)
    sn1, n1 = sn2, n2
    sn2, n2 = sn3, n3
```