Chapter 1

Extrapolation to zero

In this short chapter we will describe shortly the concept of *extrapolation to zero* and how we can apply it.

1.1 Motivation

Let $T :]0, \varepsilon[\to \mathbb{R}$ be function and assume that we have

$$T(h) = T_0 + ah^n + O(h^{n+1}). \tag{1.1}$$

for $h \to 0$. We are interested in computing $T_0 = \lim_{h\to 0} T(h)$ up to some desired accurracy. In order to do that we might have to compute $T(h)$ for very small h. That might not be feasible since $T(h)$ might be very expensive to compute for small h or even impossible due to numerical instabilities. Hence we would like to somehow accelerate the convergence of T to 0. A nice way to do that is the *Richardson extrapolation scheme* which goes as follows: Let $0 < r < 1$. Plug rh into (1.1). Then we get

$$T(rh) = T_0 + ar^n h^n + O(h^{n+1}). \tag{1.2}$$

Now multiply (1.1) by r^n, subtract it from (1.2) and divide the result by $1 - r^n$. Then we get:

$$R(h) = T_0 + O(h^{n+1})$$

where

$$R(h) := \frac{T(rh) - r^n T(h)}{1 - r^n}.$$

Note that $R(h)$ has $O(h^{n+1})$ convergence to T_0 while $T(h)$ has $O(h^n)$, i.e. $R(h)$ converges asymptotically faster. But what did we actually do? We took the linear polynomial in t^n which goes through $(rh, T(rh))$ and $(h, T(h))$ and let $R(h)$ be its value at 0, i.e. we interpolated the points and then evaluated the interpolation polynomial outside the interval; hence the term *extrapolation*. This should serve as a motivation for the sequel.

1.2 The extrapolation table

We always think of T as arising from some numerical scheme e.g. the trapezoidal rule and then T_0 is the integral of some function. Thus we do not require that T is necessarily defined for all values near 0, but only on set which has 0 as an accumulation point. In what follows, we will thus refer to T as a *method* for computing T_0.

Definition 1.1. Let T be a method for computing T_0. We say that T has an asymptotic expansion in h^p up to order pm if there exist constants $\tau_p, \tau_{2p}, \dots, \tau_{mp} \in \mathbb{R}$ such that

$$T(h) = T_0 + \tau_p h^p + \tau_{2p} h^{2p} + \cdots \tau_{mp} h^{mp} + O(h^{(m+1)p}) \tag{1.3}$$

for $h \to 0$.

Let $(x_1, y_1), \dots, (x_k, y_k)$ be a collection of points such that $x_1, \dots, x_k$ are distinct. Then there exists a polynomial P which interpolates the points, i.e. $P(x_i) = y_i$ for all i. We say that P is *the interpolation polynomial* for the points if P has the lowest degree among all polynomials which interpolate them. The interpolation polynomial is unique. Let $p > 0$ be an integer and points $(x_1^p, y_1), \dots, (x_n^p, y_n)$ such that x_i^p are distinct, be given. Let P be the interpolation polynomial for the points. We then call $P(h^p)$ the *interpolation polynomial in* p for the points.

Let T be a method with asymptotic expansion in p up to pm. The extrapolation process works as follows: We compute $T(h)$ for some points $h_1, h_2, \dots, h_k$ where $k \le m$. Then we compute the interpolation polynomial P in h^p which goes through $(h_1, T(h_1)), \dots, (h_k, T(h_k))$. We then hope that $P(0)$ gives a good approximation T_0.

In order to compute $P(0)$ we use the *Neville scheme*. Let $P_{ij}(h^p) \coloneqq P(h^p; h_{i-j+1}^p, h_i^p)$ be the interpolation polynomial in h^p which interpolates $(h_{i-j+1}^p, T(h_{i-j+1}), \dots, (h_i^p, T(h_i^p))$ and set $T_{ij} \coloneqq P_{ij}(0)$. Then according to the Neville scheme we can compute T_{ij}, $j \le i$, in the following recursive way:

1. $T_{i1} \coloneqq T(h_i)$ for $i = 1, \dots, k$.
2. $T_{ij} \coloneqq T_{i,j-1} + \dfrac{T_{i,j-1} - T_{i-1,j-1}}{r^p - 1} = \dfrac{r^p T_{i,j-1} - T_{i-1,j-1}}{r^p - 1}$ for $1 < j \le i$ where $r \coloneqq h_{i-j+1}/h_i$.

If we align T_{ij} to a triangular table, we call that the *extrapolation table*.

1.3 Convergence

If we have a numerical method or scheme that has an asymptotic expansion as (1.3), then the error decays polynomially as $h \to 0$. It is known (see e.g. theorem 9.22 in [1]) that T_{ij} has faster polynomial decay of higher degree, as $h \to 0$, then T. Let $\varepsilon_k \coloneqq |T_{kk} - T|$. We want to analyze how ε_k behaves as $k \to +\infty$, i.e. how ε_k behaves when we increse the number of extrapolation steps. Let $N_n k$ be some measure of the effort needed to compute T_{kk}. In what follows we will test numerically the qualitative hypothesis that the error converges exponentially with the computational effort i.e.

$$\varepsilon_k \sim A \exp(-cN_k^q) \tag{1.4}$$

for constants A, c, q. Note that if $\varepsilon_k = A\exp(-cN_k^q)$ then $\ln \varepsilon_k = b - cN_k^q$ so in order to test the hypothesis we will do the following: Assume that we have the error ε_k for $k = 1, \dots, n$. Then we will compute

$$(b, c, q) \coloneqq \arg\min \left\{ \sum_{k=1}^{n} |\ln \varepsilon_k - (b - cN_k^q)|^2 \right\} \tag{1.5}$$

and see whether the points $(N_k, \ln \varepsilon_k)$ fit well to the graph of $t \mapsto b - ct^q$.

We will also test the hypothesis that the error converges exponentially with the number of extrapolation steps, i.e. whether

$$\varepsilon_k \sim A \exp(-ck^q) \tag{1.6}$$

for constants A, c, q.

In order to validate the estimated parameters b, c, q we will do a simple "cross validation" by fitting the model to subsets of the data and see whether the parameters vary a lot. If they vary a lot, we conclude that the fitting is unstable. If they are almost the same we will be more confident in that the model is actually appropriate.

1.4 Code

The following Python mehtod computes the extrapolation table for some scheme which has an asymptotic expansion in h^p.

```python
#sc (Scheme): The scheme to extrapolate
#prob: The problem to apply the scheme to. We assume that sch is an
#      implementation of Scheme which can be applied to prob.
#seq (Sequence): The sequence to use in the extrapolation
#hp (bool): Indicates whether to use high precision arithmetic (true)
#           or standard double precision (false).
#returns: The extrapolation table as a list of lists.
def extrapolate(sc, prob, seq, hp):
  n = len(seq)
  X = [[0 for j in range(i + 1)] for i in range(n)]

  #X[i][j] = T_ij
  for i in range(n):
    X[i][0] = sc.apply(prob, seq[i])
    for j in range(1, i + 1):
      #r = h_{i-j} / h_i = seq[i] / seq[i-j]
      #rp = r^p.
      #Must cast the elements of seq to hp numbers if in hp mode.
      rp = ((mpf('1') * seq[i]) / (mpf('1') * seq[i-j]) if hp else seq[i] / seq[i-j]) ** sc.p
      X[i][j] = (rp * X[i][j-1] - X[i-1][j-1]) / (rp - 1)

  return X
```

Chapter 2

Romberg quadrature

2.1 The algorithm

Let $f : [a, b] \to \mathbb{R}$ be a function and $I \coloneqq \int_a^b f(x)dx$. The *trapezoidal rule* is a method for approaching I which works as follows: Let $a = t_0 < t_1 < \cdots < t_n = b$ be a subdivision of $[a, b]$. On each of the intervals $[t_{i-1}, t_i]$ we approximate $\int_{t_{i-1}}^{t_i} f(x)dx$ by the area of a trapezoid with verticies $(t_{i-1}, 0)$, $(t_{i-1}, f(t_{i-1}))$, $(t_i, f(t_i))$, $(t_i, 0)$ i.e. by $\frac{1}{2}(t_i - t_{i-1})(f(t_{i-1}) + f(t_i))$. Hence we approximate I by

$$I = \sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} f(x)dx \approx \sum_{i=1}^{n} \frac{1}{2}(t_i - t_{i-1})(f(t_{i-1}) + f(t_i)).$$

If $t_i - t_{i-1} = \frac{1}{n}(b - a) =: h$ for each i then the above estimate becomes

$$I \approx h\left(\frac{1}{2}(f(a) + f(b)) + \sum_{i=1}^{n-1} f(a + ih)\right) \tag{2.1}$$

We define $T_f(h)$ as the right hand side in (2.1).

Let $F : [0, n] \to \mathbb{R}$ be a $2k + 1$ times continuously differentiable function, n a positive integer. Then by Euler's summation formula (see formula 298 in [2]) we have

$$\sum_{i=0}^{n} F(i) = \int_0^n F(x)dx + \frac{1}{2}(F(0) + F(n)) + \sum_{i=1}^{k} \frac{B_{2i}}{(2i)!}(F^{(2i-1)}(n) - F^{(2i-1)}(0)) + R_k \tag{2.2}$$

where $R_k = \int_0^n P_{2k+1}(x)F^{(2k+1)}(x)dx$, B_m are the *Bernoulli numbers* and P_m the *Bernoulli polynomials.* If let $F(x) \coloneqq f(a + xh)$ then we get the following asymptotic expansion for the trapezoidal rule:

Theorem 2.1. *Let $f : [a, b] \to \mathbb{R}$ be $2k + 1$ times continuously differentiable and $h \coloneqq (b - a)/n$. Then*

$$T_f(h) = I + \sum_{i=1}^{k} \frac{B_{2i}}{(2i)!}(f^{(2i-1)}(b) - f^{(2i-1)}(a))h^{2i} + h^{2k+1}R_k(h) \tag{2.3}$$

where

$$R_k(h) = \int_a^b P_{k+1}\left(n\frac{x - a}{b - a}\right) f^{(2k+1)}(x)dx. \tag{2.4}$$

The following code is a trivial implementation of the trapezoidal rule. The *TrapezoidalRule* class in an implementation of the abstract class *Scheme* which represents a numerical scheme or method, which has asymptotic expansion in h^p. The Scheme class has a method named *apply* which takes in a problem to which the scheme is applied to. The argument m in the apply-method is the number of subintervals that should be used.

```
class TrapezoidalRule(Scheme):
    def __init__(self):
        super(TrapezoidalRule, self).__init__(2)

    def apply(self, inte, m):
        (a,b) = inte.interval
        h = (b - a) / m
        I = 0.5 * (inte.f(a) + inte.f(b))
        for i in range(1, m):
            I += inte.f(a + i * h)

        return I * h
```

Assume that we have computed the value of $T_f(h)$ for $h = h_1, \ldots, h_k$ and we want extrapolate to zero, i.e. we want to compute the value at zero of the interpolation polynomial in h^2 for $(h_i^2, T_f(h_i)$, $i = 1, \ldots, k$. Denote by T_{ij} the value at zero of the polynomial in h^2 which goes through $(h_{i-j+1}^2, T(h_{i-j+1}), \ldots, (h_i^2, T(h_i))$. The Neville scheme gives us the following algorithm for computing T_{ij}, $1 \leq j \leq i \leq k$, recursively:

1. $T_{i1} := T_f(h_i)$ for $i = 1, \ldots, k$.
2. $T_{ij} := T_{i,j-1} + \frac{T_{i,j-1} - T_{i-1,j-1}}{\left(\frac{h_{i-j+1}}{h_i}\right)^2 - 1}$ for $2 \leq j \leq i$.

2.2 Numerical experiments

In this section we are going to apply Romberg quadrature to various functions and also try different sequences. We will analyze how different sequences perform in the sense that we want to measure how many function evaluations we need to attain a prescribed precision.

We will try various functions and the following sequences:

- The harmonic sequence: $a_n = n$, $n \geq 0$.
- The Romberg sequence: $a_n = 2^{n-1}$, $n \geq 1$.
- The Bulirsch sequence: $a_1 = 1$, $a_2 = 2$, $a_3 = 3$ and $a_{n+2} = 2 \cdot a_n$ for $n \geq 2$. Its first elements are
$$1, 2, 3, 4, 6, 8, 12, 16, 24, 32, \ldots$$

Suppose that we are approximating the integral $I := \int_a^b f(x)dx$ using Romberg quadrature. We will use the stepsizes $h_k := (b-a)/a_k$ for the extrapolation. Let T_{ij}, $i \geq 0$ and $j \leq i$ be the extrapolation table we get and $\varepsilon_k := |T_{kk} - I|$ be the error on the diagnoals. Let N_k be the number of function evaluations needed to compute T_{kk}. We will use N_k as the measurement of computational effort as mentioned in section 1.3 and we will try to fit the exponential convergence model introduced there. We will also plot the logarithm of the error against the number of extrapolation steps. Note that $N_k = \sum_{i=1}^{k}(a_i + 1)$ where (a_i)

is our sequence, so in case of the Harmonic sequence, we have $N_n = n(n+3)/2 \approx n^2/2$ for n large. Hence if $\varepsilon_n \sim A\exp(-cN_n^q)$ then

$$\varepsilon_n \sim A\exp(-c/2^q n^{2q})$$

for n large. Thus if the error converges exponentially with the number of function evaluations, it will also converge exponentially with the number of extrapolation steps, and the exponent in the latter fitting will be twice the parameter from the former.

If our sequence is the Romberg sequence then $N_k = \sum_{i=1}^{n}(2^{i-1}+1) = 2^k + k - 1 \approx 2^k$ for k large, so if $\varepsilon_k \sim A\exp(-cN_k^q)$ then

$$\varepsilon_k \sim A\exp(-c2^{kq})$$

for k large, which is not exponential convergence. On the other hand, if the we have exponential convergence in the number of extrapolation steps, i.e.

$$\varepsilon_k \sim A\exp(-ck^q)$$

then since $k \approx \ln N_k/\ln 2$ we get

$$\ln \varepsilon_k \sim \ln A - c(\ln N_k/\ln 2)^q = \ln A - \frac{c}{(\ln 2)^q}(\ln N_k)^q$$

so if we consider the ln-ln plot of the error against the number of function evaluations, then the points should fall on the graph of a function of the form $t \mapsto b - ct^q$. The exponent should be the same as in the fitting for the logarithm of the error against the number of extrapolation steps.

For the model fitting we will thus plot the logarithm of the error agains the number of function evaluations, the number of extrapolation steps and the logarithm of the number of extrapolation steps. We will also consider the plot of the base 10 logarithm of the error against the number of function evaluations.

We conduct the experiments in Python 3 and use the high precision arithmetic library mpmath for all the computations. The precision will be set to 500 significant digits so will not have to worry about numerical instabilities.

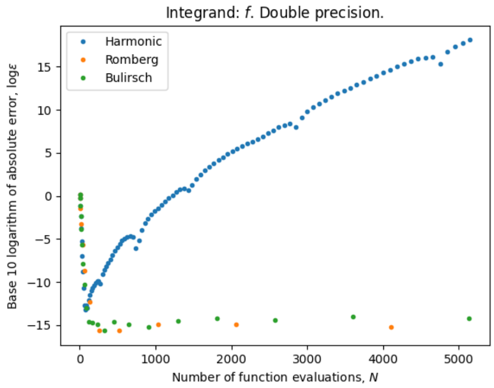
Now we will consider the results of the experiments.

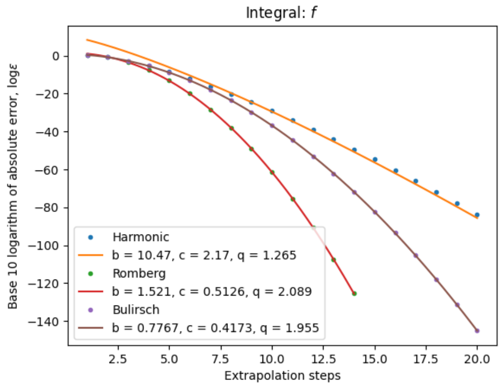
2.2.1 Cosine squared

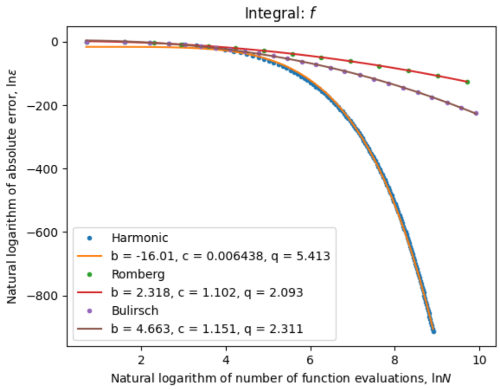
The first function we are going to try is

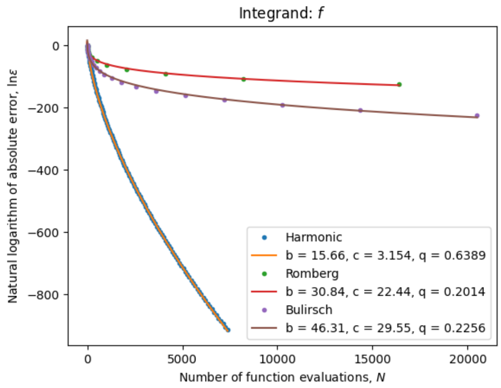
$$f : [0, \pi] \to \mathbb{R}, \quad f(x) := \cos^2(x)$$

which is entire.









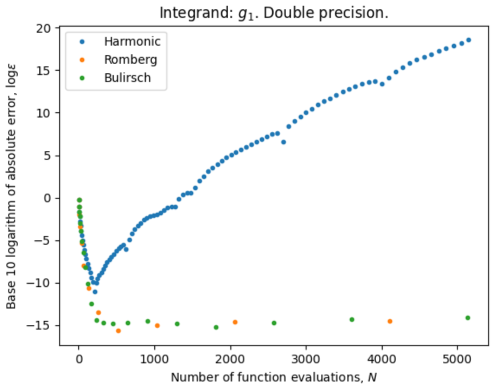
When considering the number of function evaluations against the error we seem to have exponential convergence for the Harmonic sequence. We reject the fitting for the fitting of the exponential convergence model, for Romberg and Bulirsch, since the b parameter is unreasonably large for an exponent. However, we seem to have exponential convergence in all cases when considering the number of extrapolation steps against the error.

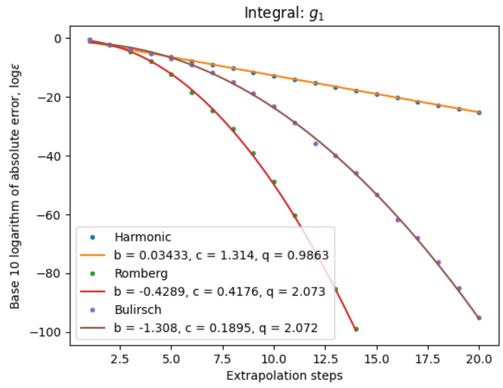
The Harmonic sequence performes best, then Bulirsch and then Romberg. In standard double precision floating point arithmetic, we get down to maching level error using any sequence. Regarding convergence with the number of extrapolation, then it looks as we have exponential convergence for the Romberg and Bulirsch sequences.

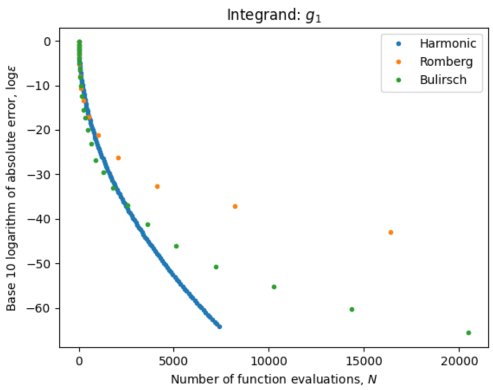
2.2.2 Function with poles

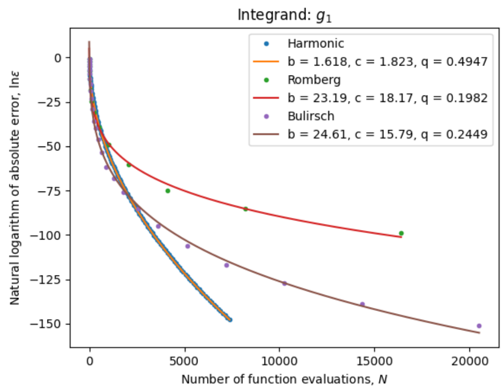
Now we will consider the following function:

$$g_a : [-1, 1] \to \mathbb{R}, \quad g_a(x) := \frac{1}{a^2 + x^2}, \, a > 0$$

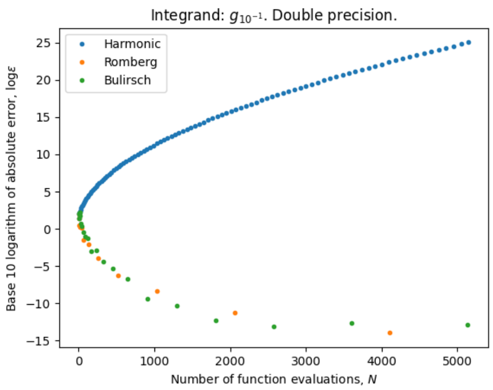


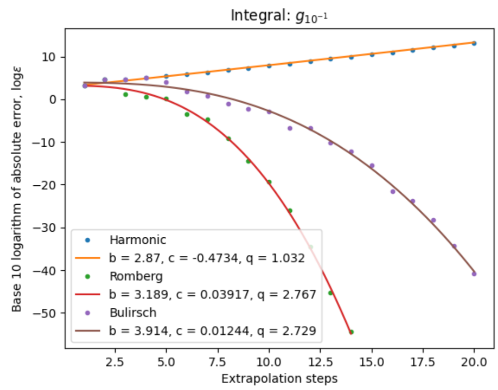


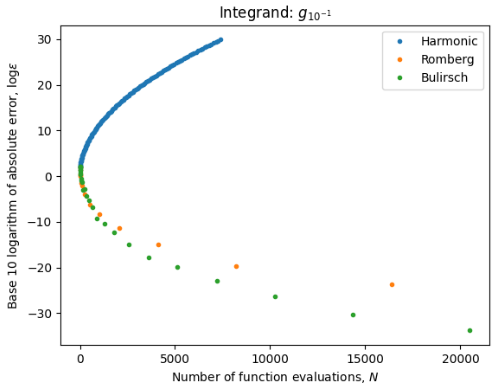


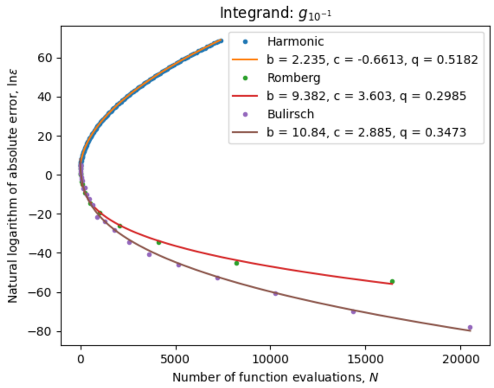


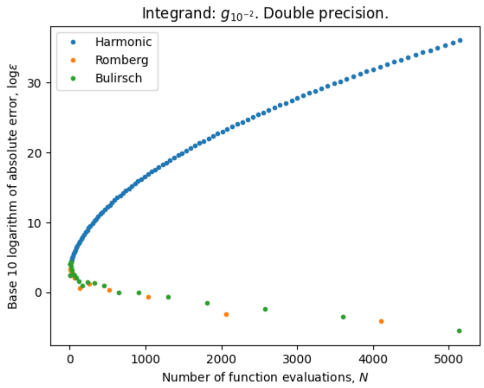
The error seems to converge exponentially with the number of function evaluations for the harmonic sequence but again the b parameters are unreasonably big for the other two sequences. For the number of extrapolation steps against the error, we seem to have exponential convergence in all cases. In standard floating point arithmetic, we get down to machine level precision for the Romberg and Bulirsch sequences but not for the Harmonic sequence. At first the sequences perform simarly but in the long run the harmonic sequence results in the fastest convergence.

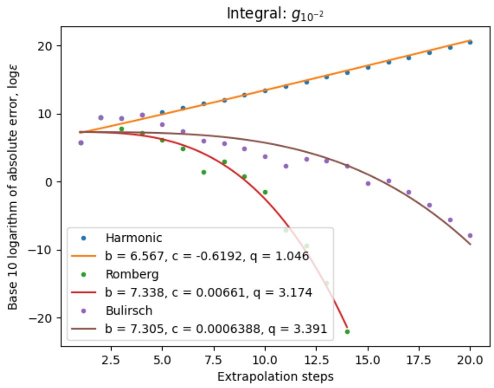


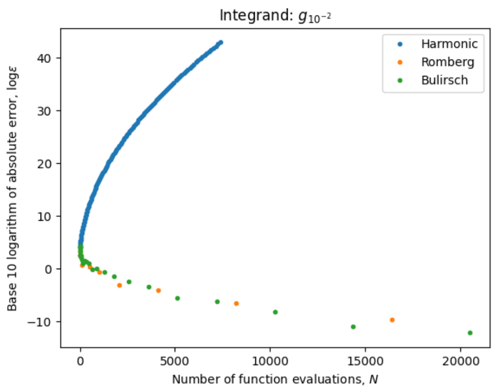


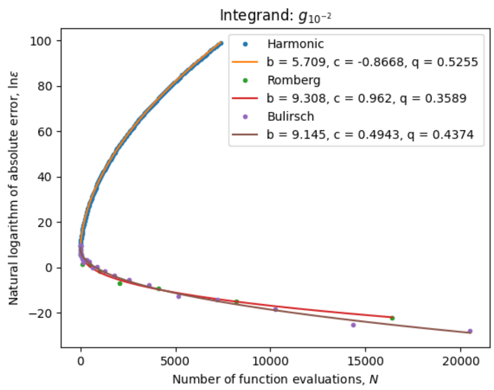






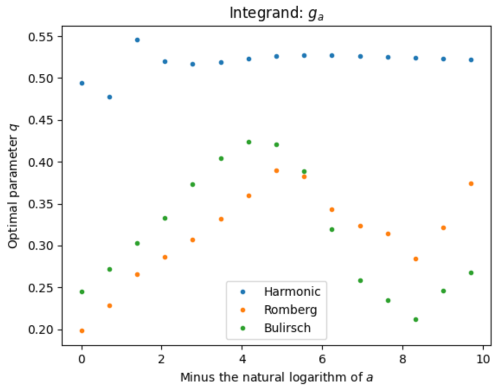






For $a = 10^{-1}, 10^{-2}$ we get divergence for the harmonic sequence, Bulirsch performes best for $a = 10^{-1}$ and Romberg for $a = 10^{-2}$. The parameters in the fitting of the exponential convergence model to the number of function evaluations against the error, are reasonable. On the contrary, the c is suspiciously small in the fitting of the extrapolation steps to the error.

If we plot the value of the optimal parameter q in the model fitting for the error against number of function evaluations, against a, the resulting graph looks as follows:

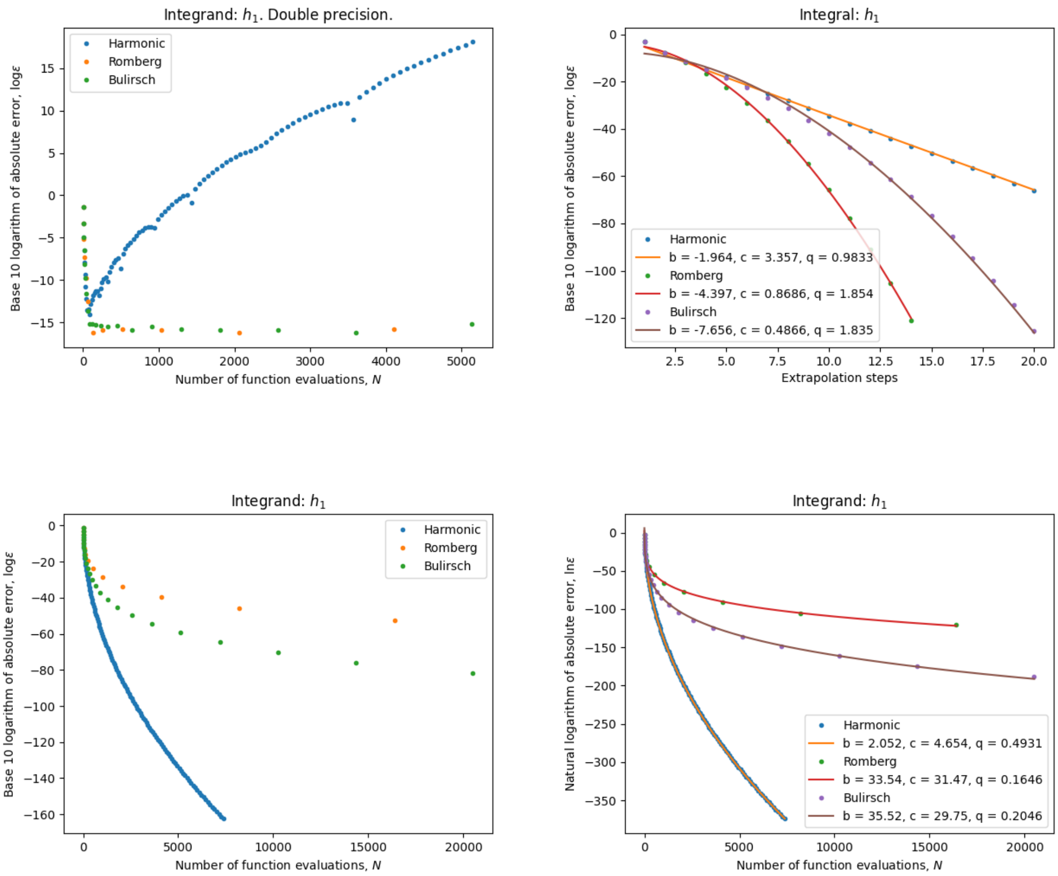


2.2.3 Logarithm

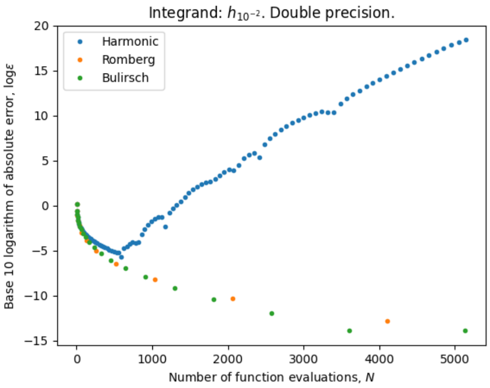
Now we will consider the following function

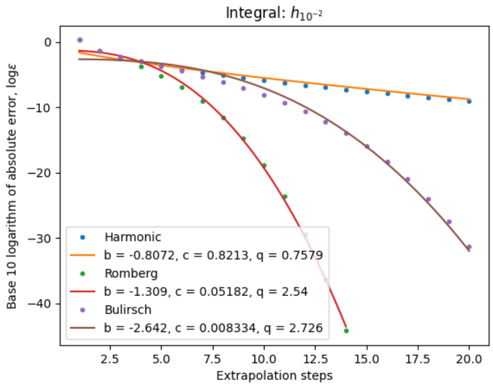
$$h_a : [0, 1] \to \mathbb{R}, \quad h_a(x) := \ln(a + x),\ a > 0.$$

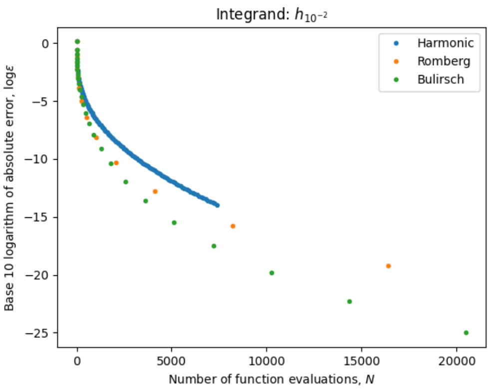
This function is analytic on neighbourhood about the interval but we have a singularity at the horizontal ray from $-a$ to $-\infty$.

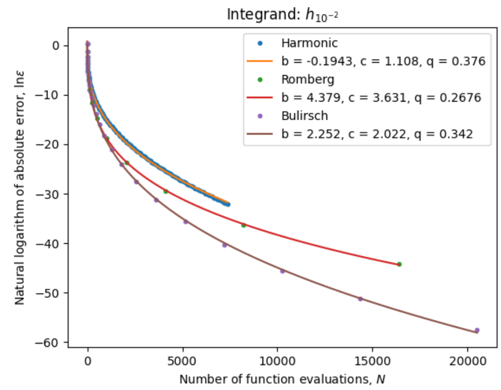


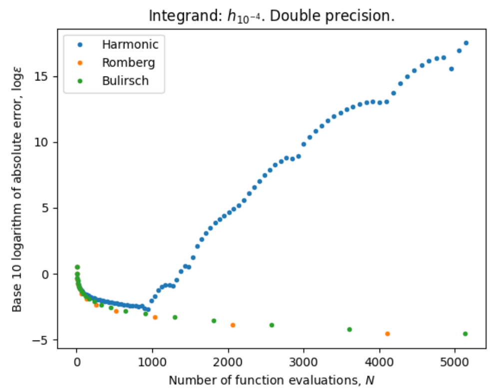
For $a = 1$ the error seems to converge exponentially with the number of extrapolation steps for the harmonic sequence but the b parameters are suspicuously large for the other sequences. The error seems to converge exponentially with the number of extrapolation steps. The harmonic sequence results in the fastest convergence in the long run, then Bulirsch and then Romberg. In standard floating point arithmetic, we get down to machine level precision using Romberg and Bulirsch but not by using the Harmonic sequence.

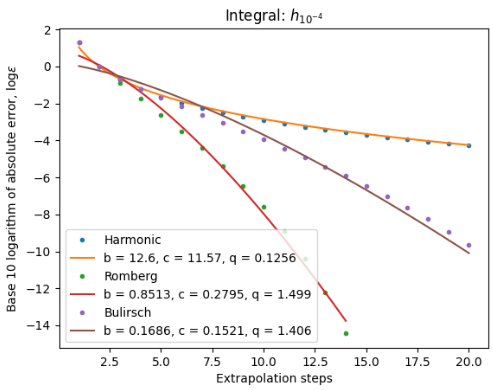


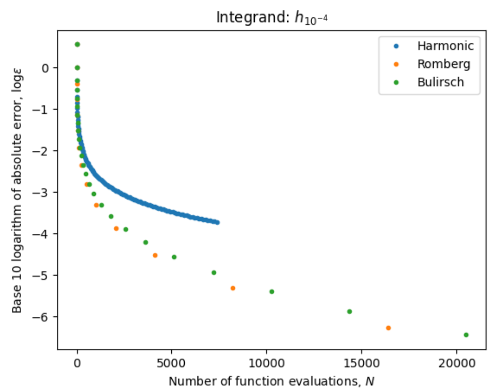


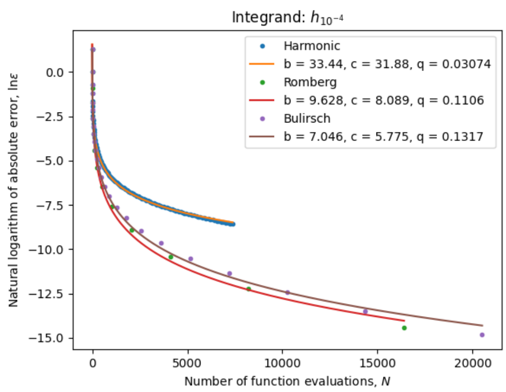






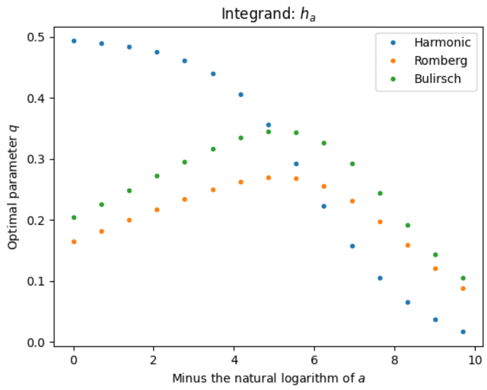






For $a = 10^{-2}, 10^{-4}$ the parameters in the fitting of the number of function evaluations against the error seem more reasonable then the ones the fitting of the number of extrapolation steps against the error. The harmonic sequence performes not as well as the other two and in standard floating point precision we do not get down to the same level of accurracy using that.

If we plot the value of the optimal parameter q against a, the resulting graph looks as follows:

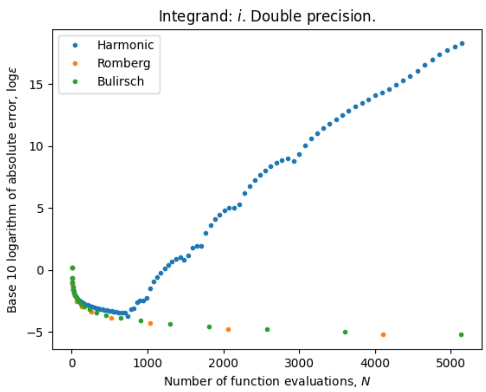


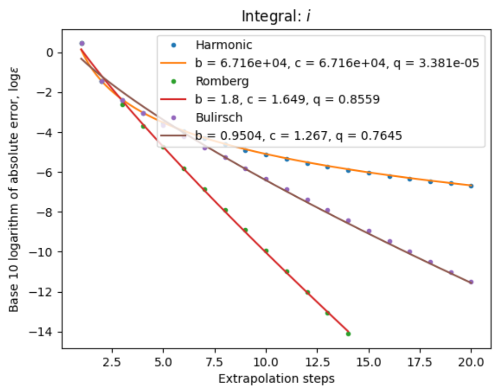
2.2.4 Area of half circle

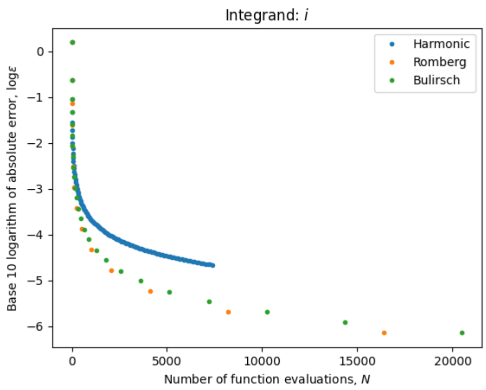
Now we will try the following function:

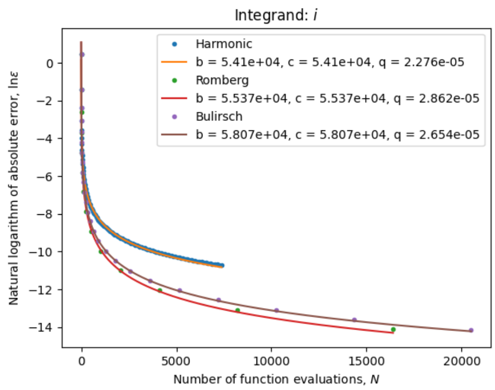
$$i : [-1, 1] \to \mathbb{R}, \quad i(x) := \sqrt{1 - x^2}.$$

This function is analytic inside the interval of definition but not at the endpoints. Its derivative has singularities at the endpoints.

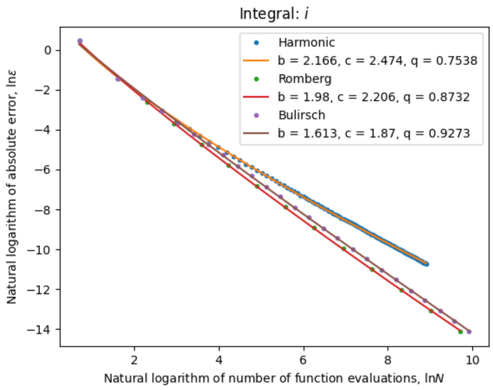








Here the fit seems to be very unstable since we are getting very small values q. Thus we shall try to plot a log-log plot of the number of evaluation agains the error. It looks as follows:

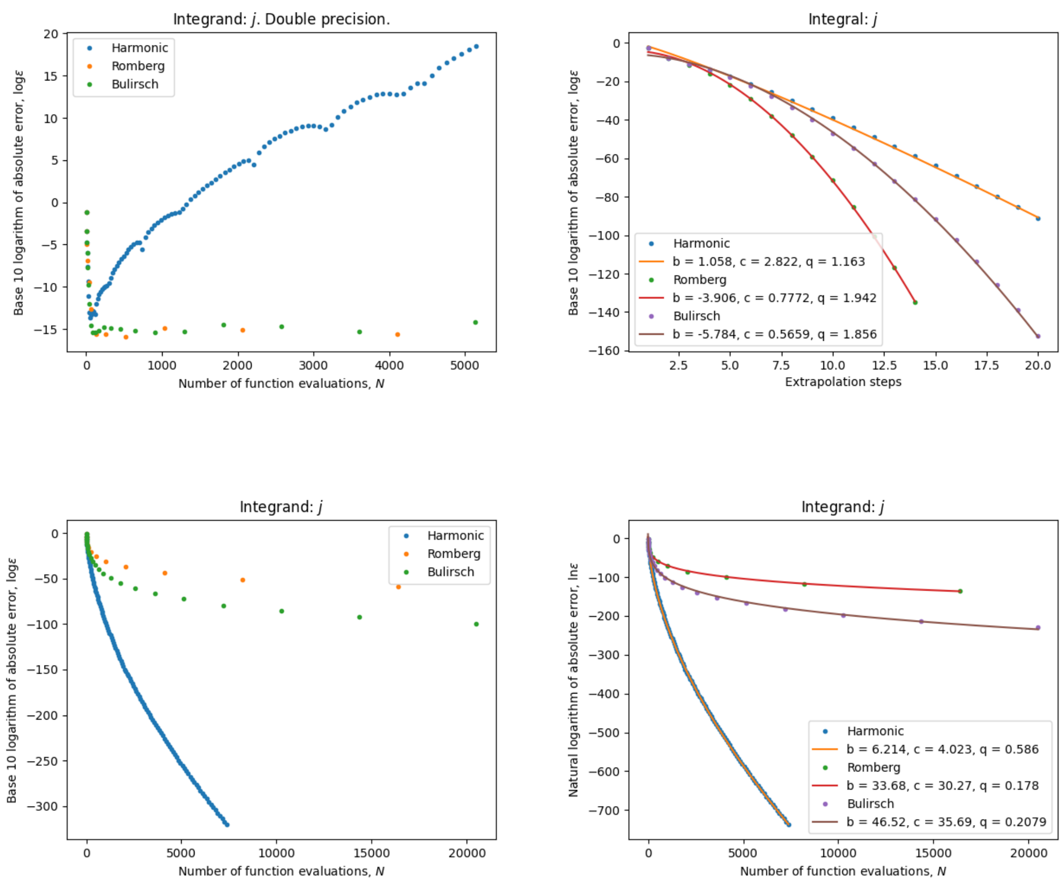


We see that the values seem to fit on a line. Thus the relation seems to be rather $\ln \varepsilon_i \sim a \ln N_i + b$ i.e. $\varepsilon_i \sim C N_i^a$ which we call algebraic convergence.

2.2.5 Gaussian

Finally we will consider the Gaussian function

$$j : [0, 1] \to \mathbb{R}, \quad k(x) := \frac{2}{\sqrt{\pi}} e^{-x^2}.$$



Here the error seems to converge exponentially with the number of function evaluations for the harmonic sequence. For the other sequences the b parameters are unreasonably large. The error seems to converge exponentially with the number of extrapolation steps, in all cases. The harmonic sequence performes best, then Bulirsch an then Romberg. Using standard double precision, we get almost down to machine level precision using Romberg or Bulirsch, but are approximately 2 digits from there, using the Harmonic sequence.

The values of the optimal parameters in the curve fitting of evaluations against the logarithm of the error are:

Integrand	Sequence	b	c	q
f	Harmonic	15.66	3.1537	0.63887
f	Romberg	30.844	22.442	0.2014
f	Bulirsch	46.309	29.549	0.22556
$g_{10^{-2}}$	Harmonic	5.7088	-0.8668	0.52546
$g_{10^{-2}}$	Romberg	9.3083	0.96199	0.35893
$g_{10^{-2}}$	Bulirsch	9.1445	0.49433	0.43743
$g_{10^{-1}}$	Harmonic	2.2352	-0.66129	0.51817
$g_{10^{-1}}$	Romberg	9.3824	3.6029	0.29851
$g_{10^{-1}}$	Bulirsch	10.844	2.8849	0.34731
g_1	Harmonic	1.6178	1.823	0.49467
g_1	Romberg	23.192	18.171	0.19817
g_1	Bulirsch	24.613	15.795	0.24492
$h_{10^{-4}}$	Harmonic	33.436	31.879	0.030738
$h_{10^{-4}}$	Romberg	9.6285	8.0889	0.1106
$h_{10^{-4}}$	Bulirsch	7.0462	5.7755	0.13169
$h_{10^{-2}}$	Harmonic	-0.19426	1.1078	0.37602
$h_{10^{-2}}$	Romberg	4.3792	3.631	0.26761
$h_{10^{-2}}$	Bulirsch	2.2519	2.0217	0.34203
h_1	Harmonic	2.052	4.6543	0.4931
h_1	Romberg	33.542	31.468	0.16462
h_1	Bulirsch	35.525	29.752	0.20461
i	Harmonic	54099	54099	$2.2756 \cdot 10^{-5}$
i	Romberg	55368	55367	$2.8621 \cdot 10^{-5}$
i	Bulirsch	58074	58073	$2.6538 \cdot 10^{-5}$
j	Harmonic	6.2138	4.0228	0.58595
j	Romberg	33.68	30.265	0.17797
j	Bulirsch	46.521	35.69	0.20788

Table 2.1: Optimal parameters by test case

The values of the optimal parameters in the curve fitting of extrapolation steps against the logarithm of the error are:

Integrand	Sequence	b	c	q
f	Harmonic	10.466	2.1696	1.2654
f	Romberg	1.5206	0.51255	2.089
f	Bulirsch	0.77673	0.41734	1.9549
$g_{10^{-2}}$	Harmonic	6.5675	−0.61916	1.0458
$g_{10^{-2}}$	Romberg	7.3378	0.0066103	3.1744
$g_{10^{-2}}$	Bulirsch	7.3047	0.00063882	3.3913
$g_{10^{-1}}$	Harmonic	2.8699	−0.47343	1.0317
$g_{10^{-1}}$	Romberg	3.1888	0.039167	2.7667
$g_{10^{-1}}$	Bulirsch	3.9142	0.012441	2.7293
g_1	Harmonic	0.034332	1.3144	0.98632
g_1	Romberg	−0.4289	0.41763	2.0726
g_1	Bulirsch	−1.3077	0.18952	2.0725
$h_{10^{-4}}$	Harmonic	12.604	11.571	0.12559
$h_{10^{-4}}$	Romberg	0.85129	0.27953	1.4991
$h_{10^{-4}}$	Bulirsch	0.16861	0.15206	1.4061
$h_{10^{-2}}$	Harmonic	−0.80722	0.82135	0.75792
$h_{10^{-2}}$	Romberg	−1.309	0.051824	2.5402
$h_{10^{-2}}$	Bulirsch	−2.6424	0.0083341	2.7259
h_1	Harmonic	−1.9642	3.3575	0.98328
h_1	Romberg	−4.397	0.86863	1.8535
h_1	Bulirsch	−7.6558	0.48664	1.8348
i	Harmonic	67160	67160	$3.3808 \cdot 10^{-5}$
i	Romberg	1.8004	1.6494	0.85593
i	Bulirsch	0.95043	1.2669	0.7645
j	Harmonic	1.0579	2.8215	1.1626
j	Romberg	−3.906	0.77717	1.9416
j	Bulirsch	−5.7837	0.56594	1.8564

Table 2.2: Optimal parameters by test case

Chapter 3

Extrapolation of difference quotients

3.1 The algorithm

Let $a \in \mathbb{R}$, $\varepsilon > 0$ and $f :]a - \varepsilon, a + \varepsilon[\to \mathbb{R}$ be differentiable at a. We are interested in estimating $f'(a)$. Assume that f is $2k+1$ times differentiable at a. Then by Taylor's theorem we have

$$f(a+h) = f(a) + f'(a)h + \frac{f''(a)}{2}h^2 + \cdots + \frac{f^{(2k)}(a)}{(2k)!}h^{2k} + \frac{f^{(2k+1)}(\xi)}{(2k+1)!}h^{2k+1} \tag{3.1}$$

where $a < \xi < a + h$. Now plug $-h$ instead of h in (3.1):

$$f(a-h) = f(a) - f'(a)h + \frac{f''(a)}{2}h^2 - \cdots + \frac{f^{(2k)}(a)}{(2k)!}h^{2k} - \frac{f^{(2k+1)}(\eta)}{(2k+1)!}h^{2k+1} \tag{3.2}$$

where $a - h < \eta < a$. If we subtract (3.2) from (3.1) and divide by $2h$ we get:

$$f'(a) = D_f(h) + \frac{f'''(a)}{3!}h^2 + \cdots + \frac{f^{(2k-1)}(a)}{(2k-1)!}h^{2k-2} + \frac{f^{(2k+1)}(\xi) + f^{(2k+1)}(\eta)}{2 \cdot (2k+1)!}h^{2k} \tag{3.3}$$

where

$$D_f(h) \coloneqq \frac{f(a+h) - f(a-h)}{2h} \tag{3.4}$$

is the *symmetric difference quotient* of f at a. Note that $\frac{1}{2}(f^{(2k+1)}(\xi) + f^{(2k+1)}(\eta))$ is in the image of $f^{(2k+1)}$ so we can rewrite (3.3) as

$$f'(a) = D_f(h) + \frac{f'''(a)}{3!}h^2 + \cdots + \frac{f^{(2k-1)}(a)}{(2k-1)!}h^{2k-2} + \frac{f^{(2k+1)}(\zeta)}{(2k+1)!}h^{2k} \tag{3.5}$$

where $a - h < \zeta < a + h$. Formula (3.5) tells us that the symmetric difference quotient method has asymptotic expansion in h^2 of order $2k-2$ if f is $2k+1$ times differentiable. Thus we can use the following scheme to extrapolate the symmetric difference quotient method:

1. $D_{i1} \coloneqq D_f(h_i)$ for $i = 1, \ldots, k$.

2. $D_{ij} \coloneqq D_{i,j-1} + \frac{D_{i,j-1} - D_{i-1,j-1}}{\left(\frac{h_{i-j+1}}{h_i}\right)^2 - 1}$ for $2 \leq j \leq i$.

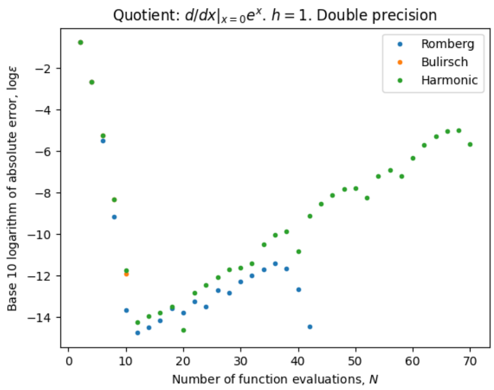
3.2 Numerical experiments

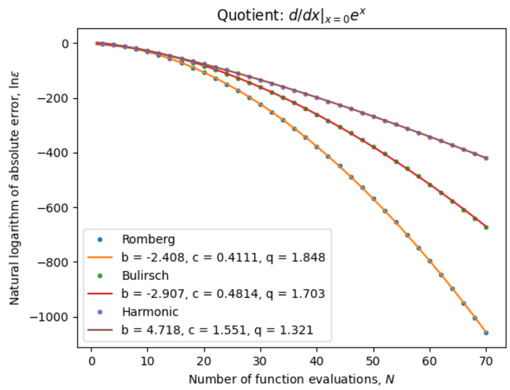
In this section we are going to extrapolate the symmetric difference quotient for approximating the derivative of a function at a given point. Let $h > 0$ be some number, $f :]a-\varepsilon, a+\varepsilon[\to \mathbb{R}$ a function differentiable at a and $n_1 < n_2 < \cdots$ a sequence of integers. Let $h_i := h/n_i$. Let D_{ij} be the extrapolation table that we get from extrapolating in h^2 using the points $(h_1^2, D_f(h_1)), (h_2^2, D_f(h_2)), \ldots$, as we described in the first chapter. We let $\varepsilon_i := |X_{ii} - f'(a)|$. We want to analyze how ε_i as i increases and we also want to do similar efficiency analyzis as in the chapter on Romberg quadrature and check whether we have exponential convergence. We will do the computations with precision up to 500 significant digits and also using standard double precision arithmetic.

Now we will consider the results of the experiments.

3.2.1 The exponential function

We begin by considering the derivative of the exponential function at zero.



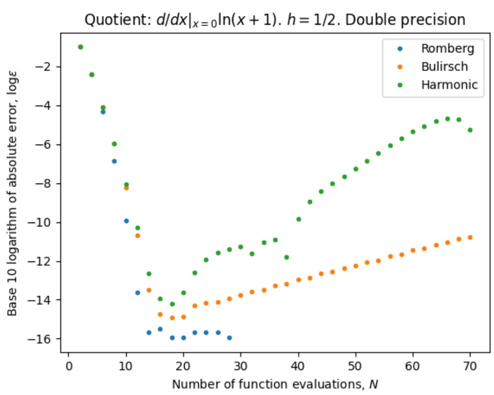


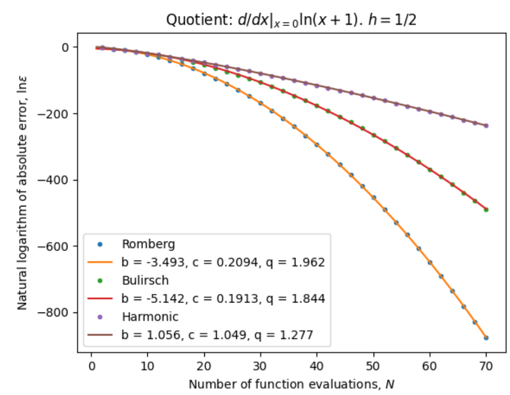
Here the model fits remarkably well. In standard floating point arithmetic, we get down to machine level precision using any sequence. Romberg results in the fastest convergence, then Bulirsch and then the harmonic sequence.

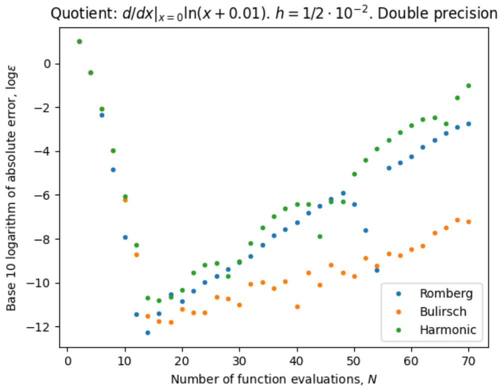
3.2.2 Logarithm

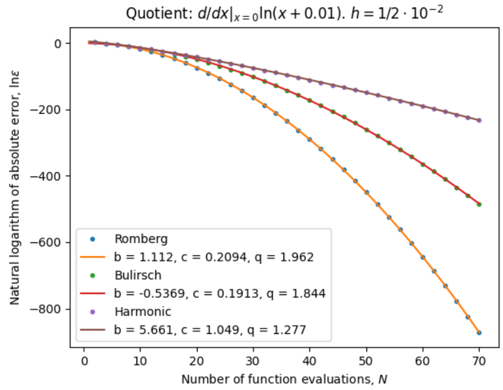
Now we will consider the dervative at zero of the function

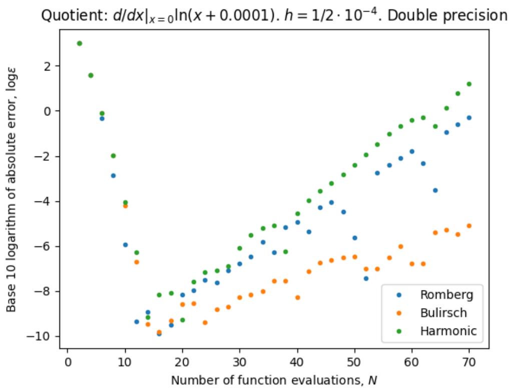
$$g_a(x) := \ln(x+a), \quad a > 0.$$

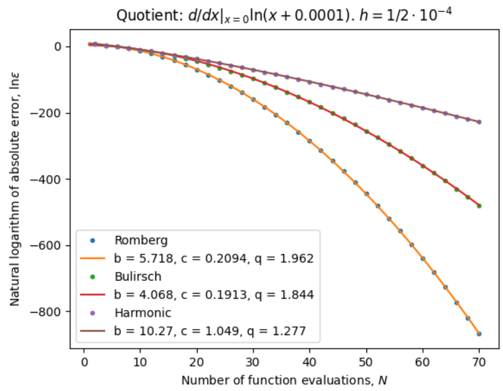










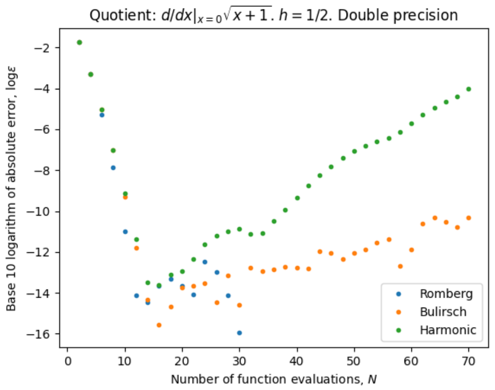


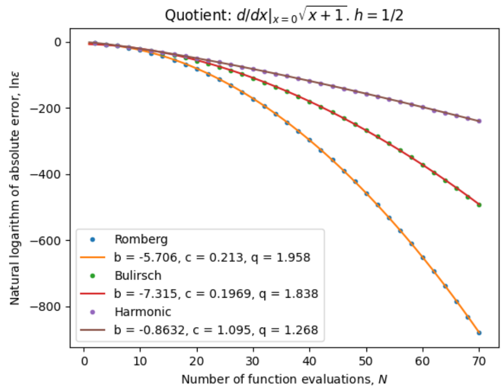
Here the model also fits remarkably well. In standard floating point arithmetic, we get down to machine level precision using any sequence. Romberg results in the fastest convergence, then Bulirsch and then the harmonic sequence. Note that the q parameter is independent of a.

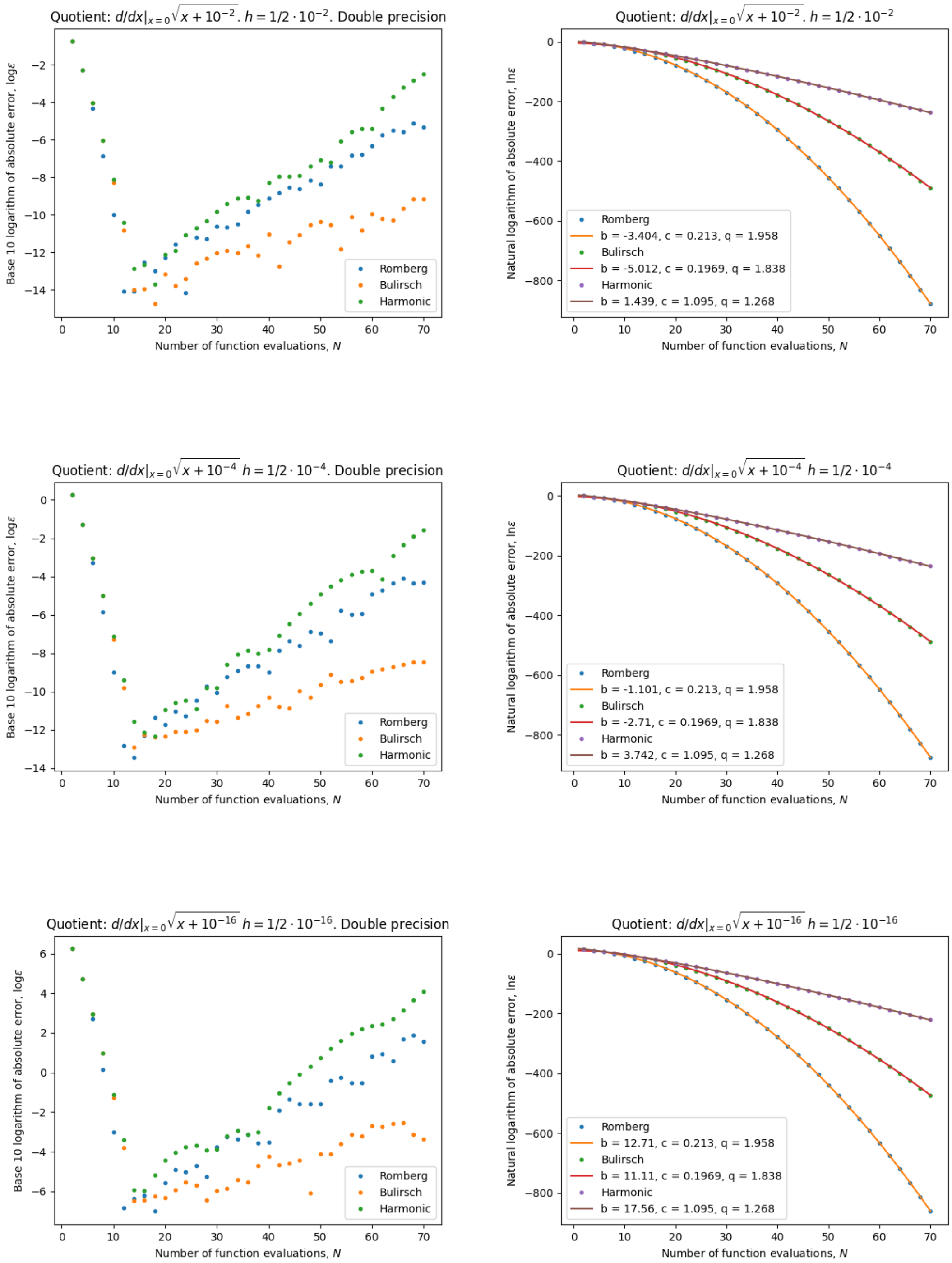
3.2.3 Square root

Now we shall consider the derivative at zero of the following function:

$$h_a(x) := \sqrt{a+x}, \quad a > 0.$$







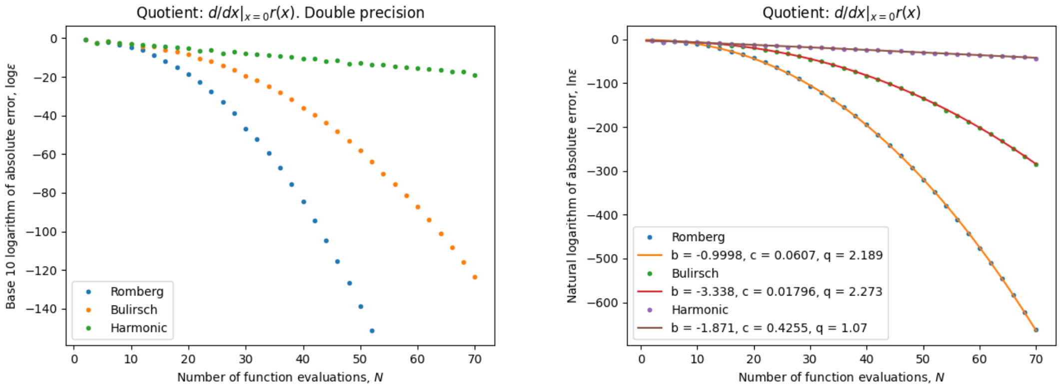
The model fits remarkably well. In standard floating point arithmetic, we get down to machine level precision using any sequence. Romberg results in the fastest convergence, then Bulirsch and then the harmonic sequence. The parameter q is also independent of a.

3.2.4 Smooth but not analytic function

Now we will consider the derivate at zero of the following function:

$$r(x) := \begin{cases} e^{-1/x} & \text{if } x > 0 \\ 0 & \text{else} \end{cases}$$

which is smooth but not analytic.



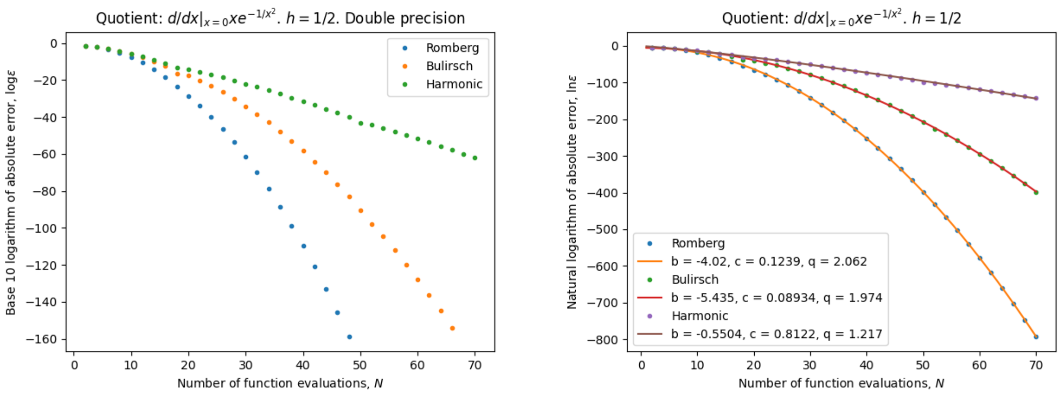
The model fits very well. Romberg results in the fastest convergence, then Bulirsch and then the harmonic sequence.

3.2.5 Another smooth but not analytic function

Now we will consider the derivative at zero of the following function:

$$i(x) := \begin{cases} xe^{-1/x^2} & \text{if } x \neq 0 \\ 0 & \text{else} \end{cases}$$

which is smooth but not analytic.

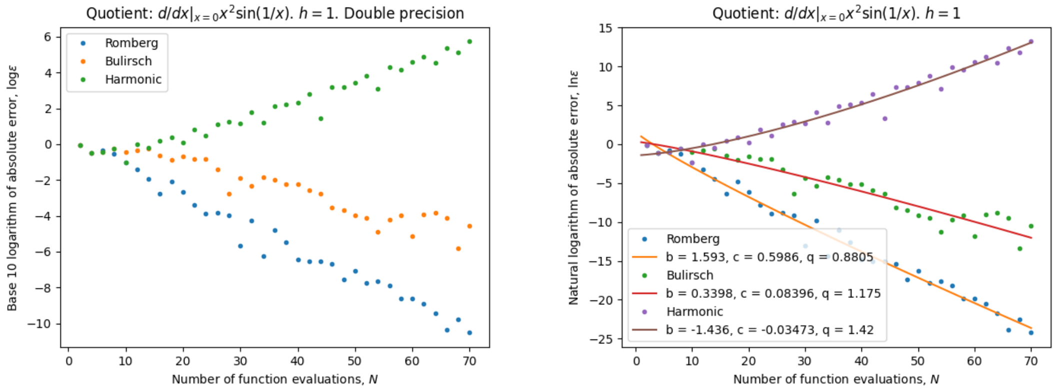


The model fits very well. Romberg results in the fastest convergence, then Bulirsch and then the harmonic sequence.

3.2.6 Only once differentiable function

Finally we will consider the derivate at zero of the following function which is only once differentiable at that point:

$$j(x) := \begin{cases} x^2 \sin \frac{1}{x} & \text{if } x \neq 0 \\ 0 & \text{else} \end{cases}.$$



Here the model does not fit very well and we do not have nice convergence in all cases; for the harmonic sequence we get divergence. Note that we do not have the asymptotic expansion for the derivate here, since the function is only once differentiable.

The parameters from the fitting are:

Derivative	Sequence	b	c	q
$d/dx\|_{x=0}r(x)$	Romberg	−0.99982	0.060703	2.189
$d/dx\|_{x=0}r(x)$	Bulirsch	−3.3384	0.017962	2.2735
$d/dx\|_{x=0}r(x)$	Harmonic	−1.8705	0.42555	1.0695
$d/dx\|_{x=0}xe^{-1/x^2}.\ h = 1/2$	Romberg	−4.0202	0.12391	2.0616
$d/dx\|_{x=0}xe^{-1/x^2}.\ h = 1/2$	Bulirsch	−5.4352	0.089335	1.9741
$d/dx\|_{x=0}xe^{-1/x^2}.\ h = 1/2$	Harmonic	−0.55045	0.81216	1.2174
$d/dx\|_{x=0} \sin x.\ h = 1/2$	Romberg	−5.3663	0.51067	1.8066
$d/dx\|_{x=0} \sin x.\ h = 1/2$	Bulirsch	−5.1899	0.64124	1.651
$d/dx\|_{x=0} \sin x.\ h = 1/2$	Harmonic	3.912	1.9876	1.2878
$d/dx\|_{x=0} \ln(x + 0.0001).\ h = 1/2 \cdot 10^{-4}$	Romberg	5.7177	0.20942	1.9619
$d/dx\|_{x=0} \ln(x + 0.0001).\ h = 1/2 \cdot 10^{-4}$	Bulirsch	4.0682	0.19126	1.8443
$d/dx\|_{x=0} \ln(x + 0.0001).\ h = 1/2 \cdot 10^{-4}$	Harmonic	10.266	1.0494	1.2768
$d/dx\|_{x=0} \ln(x + 0.01).\ h = 1/2 \cdot 10^{-2}$	Romberg	1.1125	0.20942	1.9619
$d/dx\|_{x=0} \ln(x + 0.01).\ h = 1/2 \cdot 10^{-2}$	Bulirsch	−0.53694	0.19126	1.8443
$d/dx\|_{x=0} \ln(x + 0.01).\ h = 1/2 \cdot 10^{-2}$	Harmonic	5.6607	1.0494	1.2768
$d/dx\|_{x=0} \ln(x + 1).\ h = 1/2$	Romberg	−3.4927	0.20942	1.9619
$d/dx\|_{x=0} \ln(x + 1).\ h = 1/2$	Bulirsch	−5.1421	0.19126	1.8443
$d/dx\|_{x=0} \ln(x + 1).\ h = 1/2$	Harmonic	1.0555	1.0494	1.2768
$d/dx\|_{x=0}x^2 \sin(1/x).\ h = 1$	Romberg	1.5925	0.59863	0.88054
$d/dx\|_{x=0}x^2 \sin(1/x).\ h = 1$	Bulirsch	0.33976	0.083956	1.1751
$d/dx\|_{x=0}x^2 \sin(1/x).\ h = 1$	Harmonic	−1.436	−0.034733	1.4204
$d/dx\|_{x=0}\sqrt{x + 1}.\ h = 1/2$	Romberg	−5.7063	0.21299	1.9582
$d/dx\|_{x=0}\sqrt{x + 1}.\ h = 1/2$	Bulirsch	−7.315	0.19691	1.8379
$d/dx\|_{x=0}\sqrt{x + 1}.\ h = 1/2$	Harmonic	−0.86323	1.0951	1.2682
$d/dx\|_{x=0}\sqrt{x + 10^{-2}}.\ h = 1/2 \cdot 10^{-2}$	Romberg	−3.4037	0.21299	1.9582
$d/dx\|_{x=0}\sqrt{x + 10^{-2}}.\ h = 1/2 \cdot 10^{-2}$	Bulirsch	−5.0125	0.19691	1.8379
$d/dx\|_{x=0}\sqrt{x + 10^{-2}}.\ h = 1/2 \cdot 10^{-2}$	Harmonic	1.4394	1.0951	1.2682
$d/dx\|_{x=0}\sqrt{x + 10^{-4}}\ h = 1/2 \cdot 10^{-4}$	Romberg	−1.1011	0.21299	1.9582
$d/dx\|_{x=0}\sqrt{x + 10^{-4}}\ h = 1/2 \cdot 10^{-4}$	Bulirsch	−2.7099	0.19691	1.8379
$d/dx\|_{x=0}\sqrt{x + 10^{-4}}\ h = 1/2 \cdot 10^{-4}$	Harmonic	3.7419	1.0951	1.2682
$d/dx\|_{x=0}\sqrt{x + 10^{-16}}\ h = 1/2 \cdot 10^{-16}$	Romberg	12.714	0.21299	1.9582
$d/dx\|_{x=0}\sqrt{x + 10^{-16}}\ h = 1/2 \cdot 10^{-16}$	Bulirsch	11.106	0.19691	1.8379
$d/dx\|_{x=0}\sqrt{x + 10^{-16}}\ h = 1/2 \cdot 10^{-16}$	Harmonic	17.557	1.0951	1.2682
$d/dx\|_{x=0}e^x$	Romberg	−2.408	0.41106	1.8479
$d/dx\|_{x=0}e^x$	Bulirsch	−2.9068	0.48137	1.703
$d/dx\|_{x=0}e^x$	Harmonic	4.7175	1.5509	1.3212

Table 3.1: Optimal parameters by test case

Excluding the computation of $d/dx|_{x=0}x^2 \sin 1/x$, the model fits exceptionally. We always get fast convergence except when computing $d/dx|_{x=0}x \sin 1/x$ and extrapolation with the harmonic sequence. Excluding this case, we always get almost down to machine level precision when using double precision arithmetic, using any extrapolation sequence. It is worth noting that $x \sin 1/x$ is only once differentiable at 0 so we do not have the asymptotic expansion for its derivative at 0. The Romberg sequence performes best and the harmonic sequence worst, in all cases.

Chapter 4

Initial Value Problems

4.1 The explicit midpoint rule

Let $f : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ be a smooth mapping and consider the initial value problem

$$y'(t) = f(t, y(t)), \quad y(a) = y_a, \quad t \in [a, b]. \tag{4.1}$$

The explicit midpoint methods is a method for computing an approximation to the solution of (4.1), and it goes as follows: Let $n \geq 1$ be an integer and $h \coloneqq (b-a)/2n$. We then define recursively

$$\xi_h(a) \coloneqq y_a, \quad \xi_h(a+h) \coloneqq \xi_h(a) + hf(a, \xi_h(a))$$

and

$$\xi_h(a + (i+1)h) \coloneqq \xi_h(a + (i-1)h) + 2hf(a + ih, \xi(a + ih)).$$

Then ξ_h is an approximate solution to (4.1) defined at $a, a+h, \ldots, b$. We are interested in the value $X_f(h) \coloneqq \xi_h(b)$. It is possible to show that $X_f(h)$ has an asymptotic expansion in h^2. We have the following implementation in Python of the explicit midpoint rule for computing $X_f(h)$.

```
class ExplicitMidpointRule(Scheme):

  def __init__(self):
    super(ExplicitMidpointRule, self).__init__(2)

  def apply(self, ivp, n):
    h = (ivp.b - ivp.a) / (2 * n)
    y_sl = ivp.y0
    y_l = ivp.y0 + h * ivp.f(ivp.a, ivp.y0)

    for i in range(1, 2 * n):
      tmp = y_l
      y_l = y_sl + 2 * h * ivp.f(ivp.a + i * h, y_l)
      y_sl = tmp

    return y_l
```

4.2 Numerical experiments

In this section we are going to extrapolate the explicit midpoint rule and analyze the convergence of the approximations as we extrapolate more often. Consider the initial value problem (4.1). Let $n_1 < n_2 < \cdots$ be some sequence of integers and $h_i := (b-a)/n_i$. Let X_{ij} the extrapolation table which we get from extrapolating in h^2, using the points $(h_i, X_f(h_i))$. Let $\varepsilon_i := |X_{ii} - y(b)|$ be the absolute error. We are going to do the same convergence and efficency analysis as in the two previous chapters. We will both do the computations using high precision arithmetic with 500 correct digits and also in standard double precision.

In those cases where we do not have an analytic solution to the equations, we computed a reference solution up to 500 significant digits. We did that by using extrapolation with the harmonic sequence and estimating the error as the difference between successive terms in the sequence of approximations.

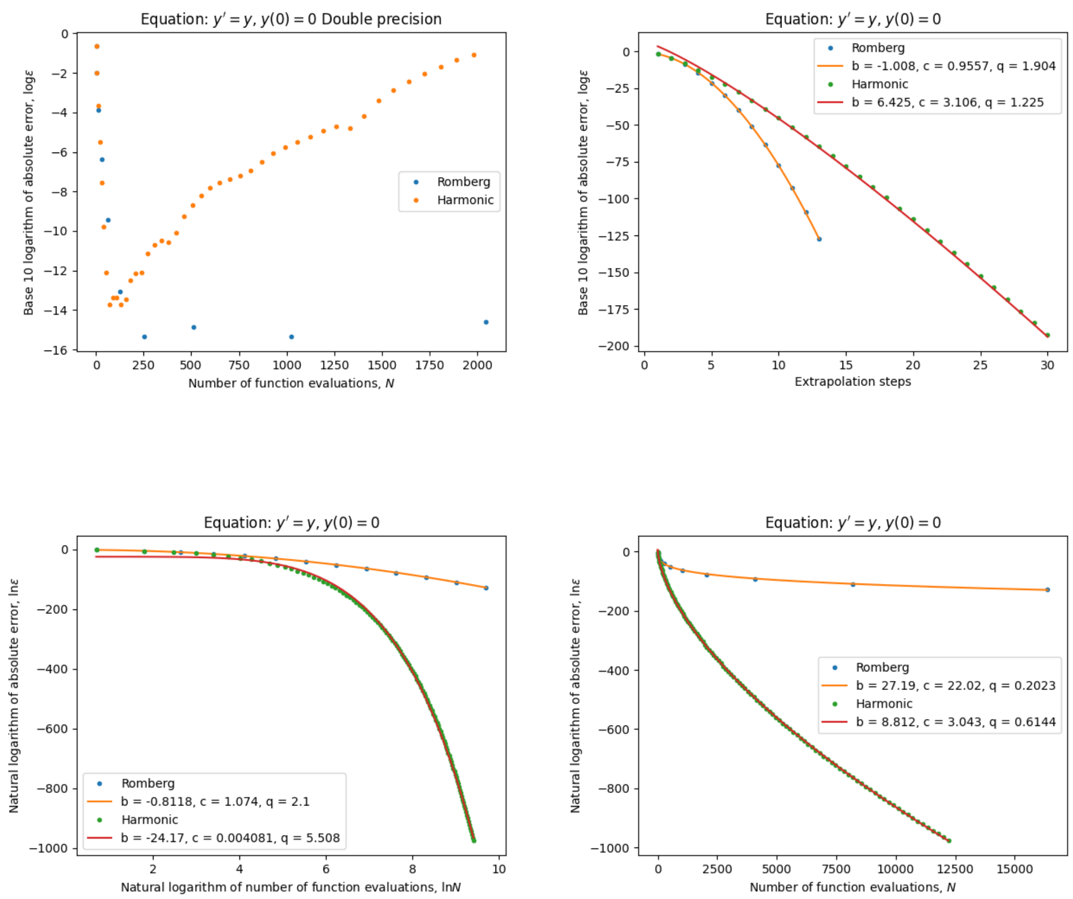
Now we will consider the results of the experiments.

4.2.1 Exponential growth

First we will consider the following initial value problem:

$$y'(x) = y(x), \quad y(0) = 0, \quad x \in [0, 1] \tag{4.2}$$

whose solution is the analytic function $y(x) = e^x$.



Here we have exponential convergence in all cases. The harmonic sequence performes better than Romberg. In standard floating point arithmetic, we get down to maching level error using either sequence.

4.2.2 Logistic curve

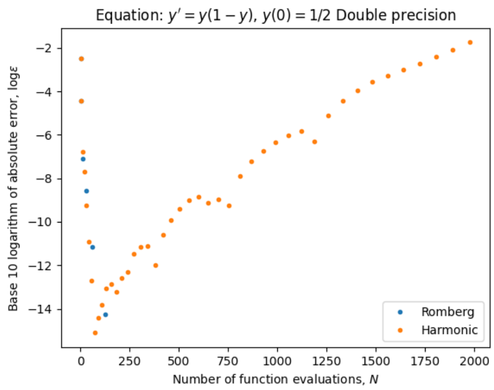
Then we will consider the following initial value problem

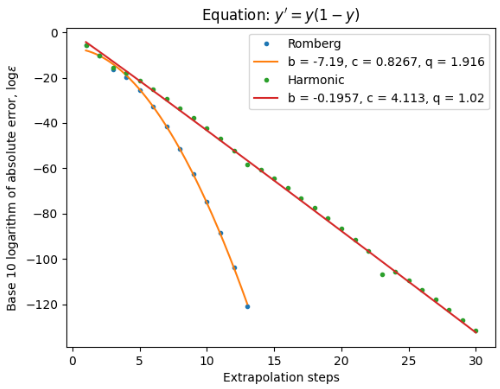
$$y'(x) = y(x)(1 - y(x)), \quad y(0) = 1/2, \quad x \in [0, 1] \tag{4.3}$$

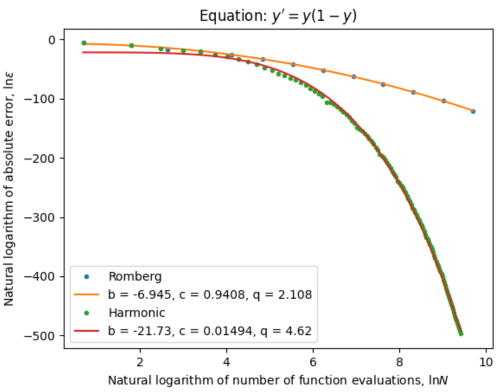
whose solution is the sigmoid function

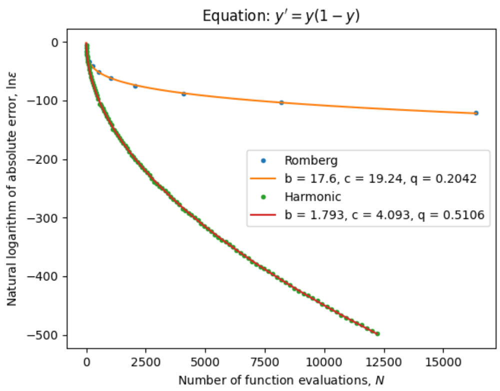
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

which is analytic.









Here the model also fits very well, the Harmonic sequence performes better and we get down to machine level precision using either sequence in standard floating point arithmetic.

4.2.3 Tangens

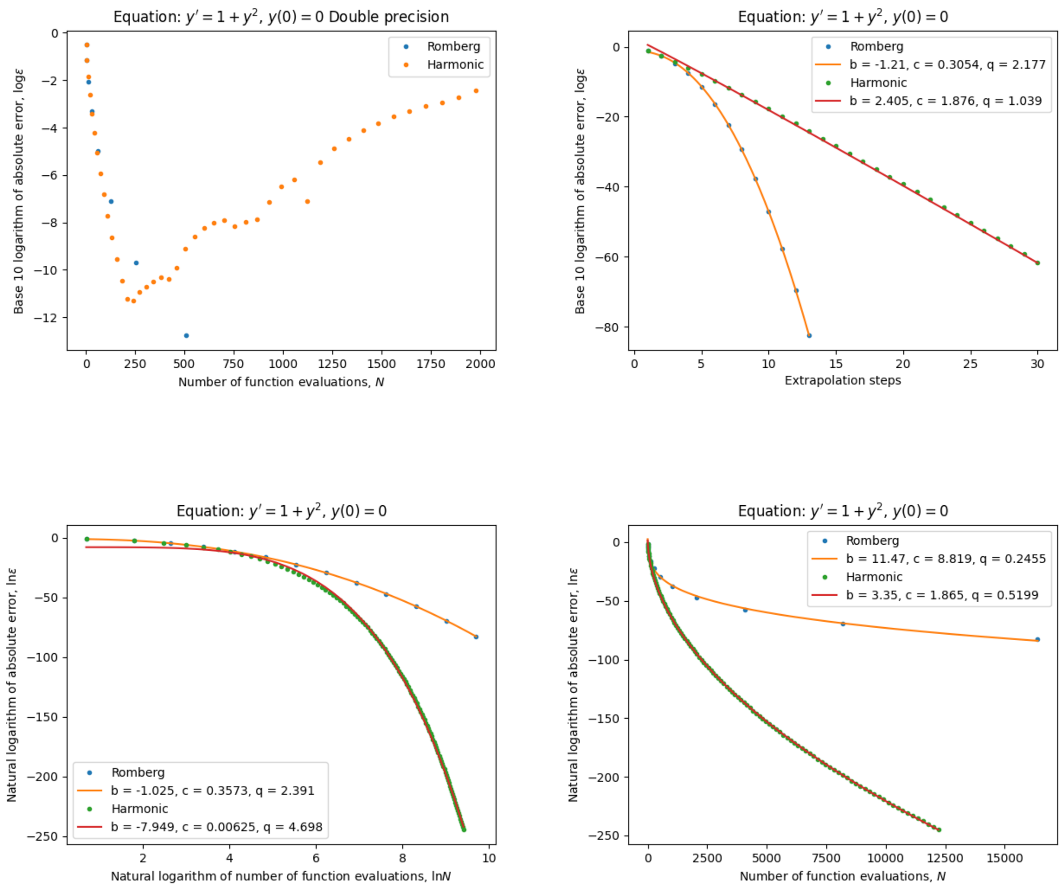
Now we will consider the following equation

$$y'(x) = 1 + y(x)^2, \quad y(0) = 0, \quad x \in [0, 1] \tag{4.4}$$

whose solution is

$$y(x) := \tan(x)$$

which is meromorphic and we are quite far from singularites.



Here we also have exponential convergence in all cases. The harmonic sequence performes better than Romberg. In standard floating point arithmetic, we get down to maching level error using either sequence.

4.2.4 Equation with singularity

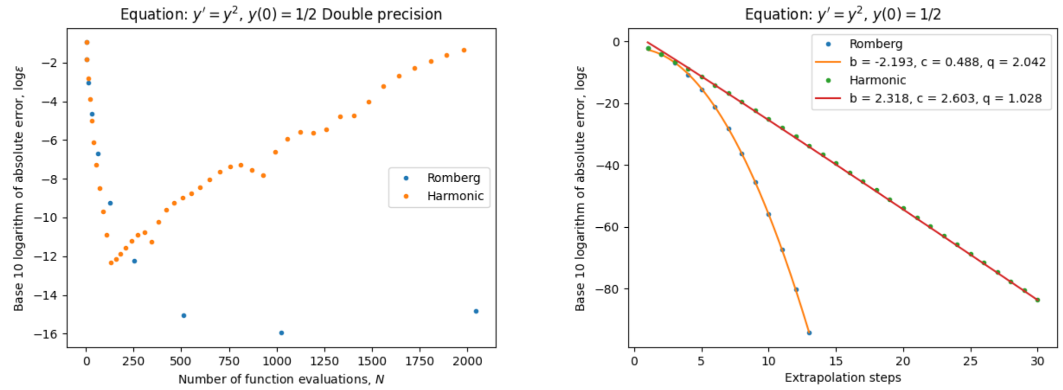
Now we will consider the following initial value problem:

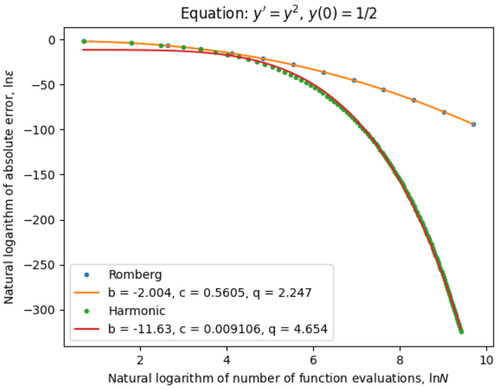
$$y'(t) = y^2(t), \quad y(0) = 1/(1+a), \quad t \in [0,1] \tag{4.5}$$

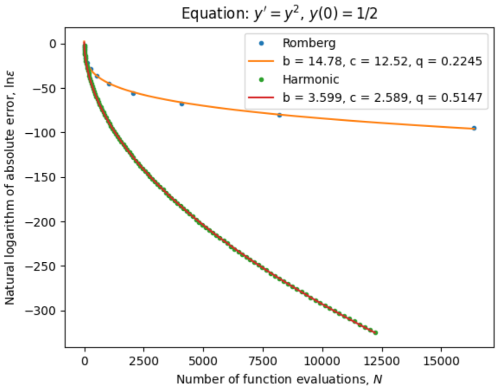
whose solution is

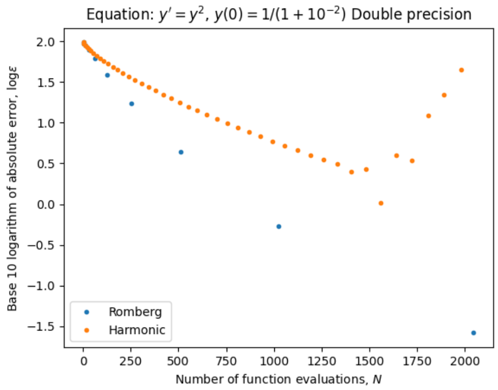
$$y(t) = \frac{1}{1-(t-a)}.$$

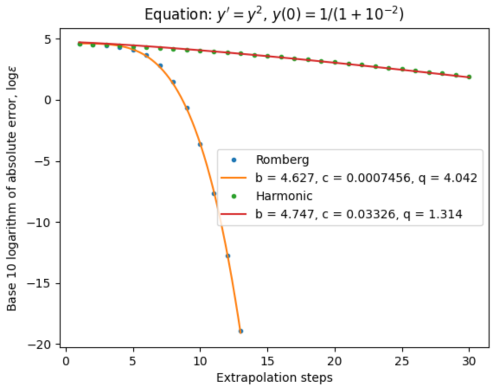
The solution is meromorphic with a pole at $1 + a$.

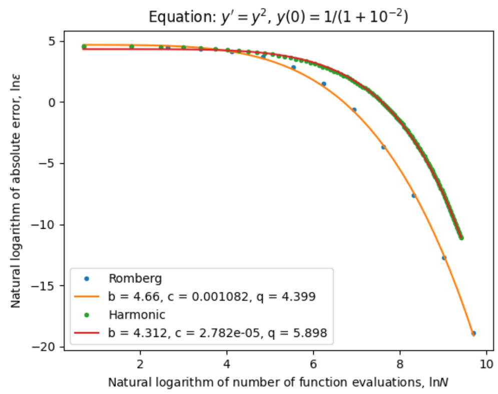


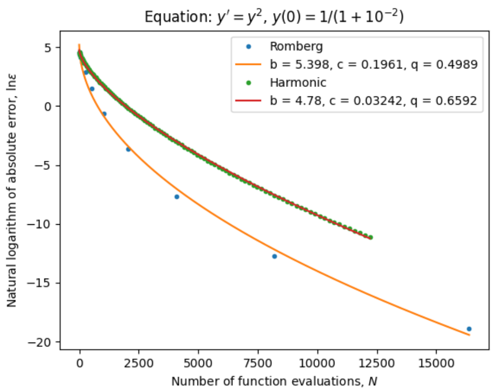

Equation: $y' = y^2$, $y(0) = 1/2$
Natural logarithm of absolute error, lnε
Natural logarithm of number of function evaluations, lnN
Romberg
b = -2.004, c = 0.5605, q = 2.247
Harmonic
b = -11.63, c = 0.009106, q = 4.654

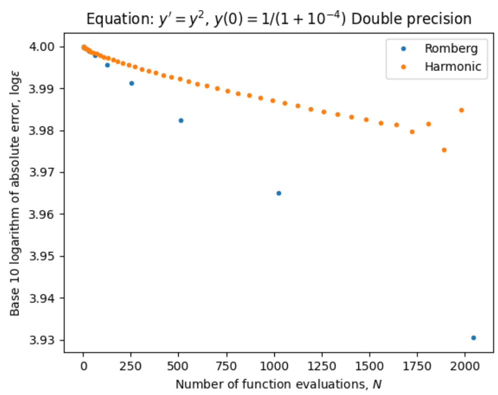

Equation: $y' = y^2$, $y(0) = 1/2$
Natural logarithm of absolute error, lnε
Number of function evaluations, N
Romberg
b = 14.78, c = 12.52, q = 0.2245
Harmonic
b = 3.599, c = 2.589, q = 0.5147

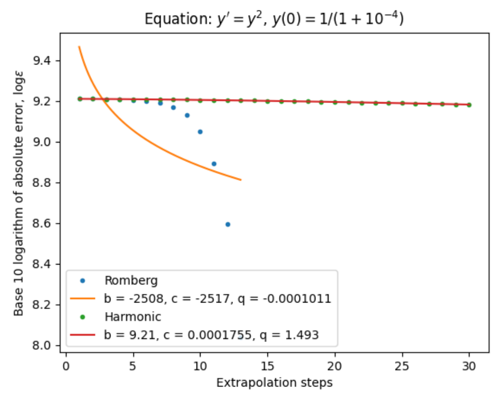

Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-2})$ Double precision
Base 10 logarithm of absolute error, logε
Number of function evaluations, N
Romberg
Harmonic

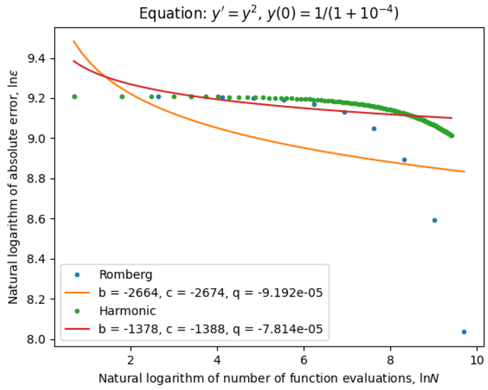

Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-2})$
Base 10 logarithm of absolute error, logε
Extrapolation steps
Romberg
b = 4.627, c = 0.0007456, q = 4.042
Harmonic
b = 4.747, c = 0.03326, q = 1.314

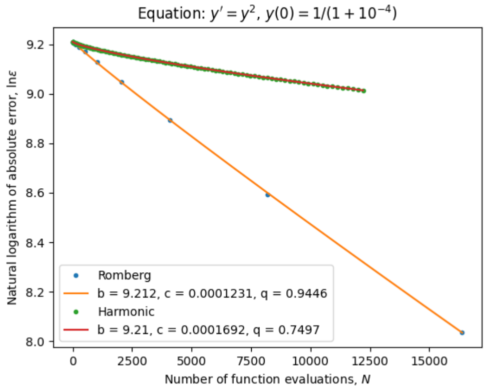

Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-2})$
Natural logarithm of absolute error, lnε
Natural logarithm of number of function evaluations, lnN
Romberg
b = 4.66, c = 0.001082, q = 4.399
Harmonic
b = 4.312, c = 2.782e-05, q = 5.898


Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-2})$
Natural logarithm of absolute error, lnε
Number of function evaluations, N
Romberg
b = 5.398, c = 0.1961, q = 0.4989
Harmonic
b = 4.78, c = 0.03242, q = 0.6592


Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$ Double precision
Base 10 logarithm of absolute error, logε
Number of function evaluations, N
Romberg
Harmonic

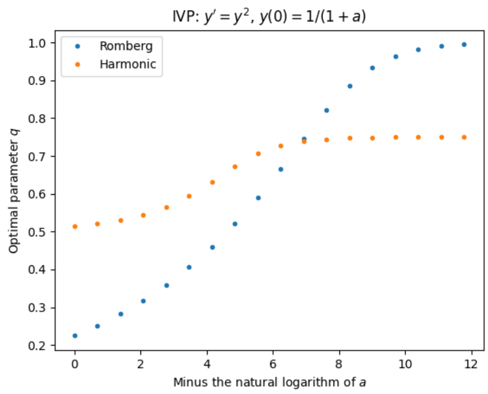

Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$
Base 10 logarithm of absolute error, logε
Extrapolation steps
Romberg
b = -2508, c = -2517, q = -0.0001011
Harmonic
b = 9.21, c = 0.0001755, q = 1.493





The model fits well for large a, and then the Harmonic sequence works better. But for small a we get a very poor fitting, and extremely slow convergence towards the solution. For $a = 10^{-4}$, when considering the number of function evaluations agains the error, we can not say that we have exponential convergence because the values on the vertical axis are on much smaller scale then the ones on the horizontal axis. The fitting fails entirely when considering the number of extrapolation steps against the error with the Romberg sequence and $a = 10^{-4}$.

The plot of q against a is as follows:



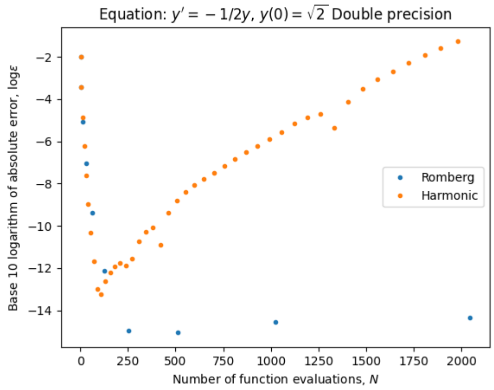
4.2.5 Equation with moderate singularity

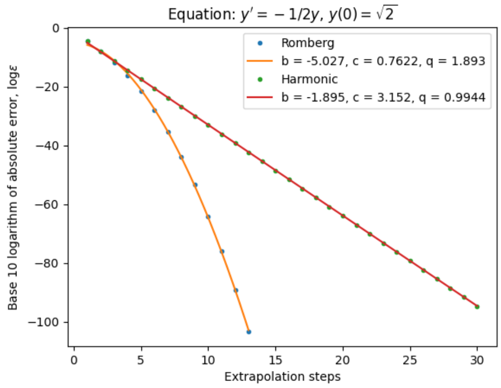
Now we will consider the following initial value problem

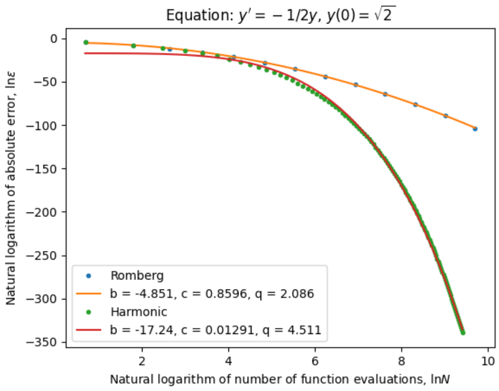
$$y'(t) = -\frac{1}{2y}, \quad y(0) = \sqrt{1+a}, \quad t \in [0,1] \tag{4.6}$$

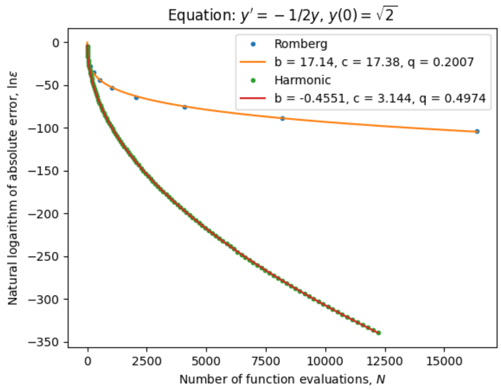
whose solution is

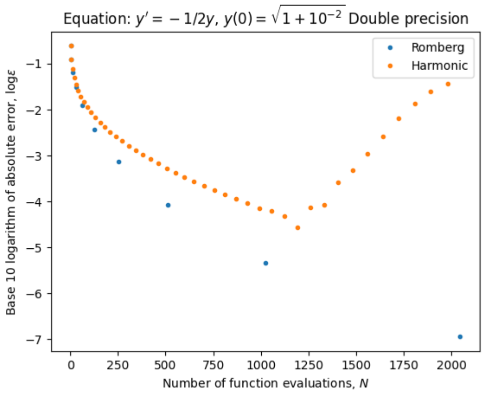
$$y(t) = \sqrt{1-(t-a)}$$

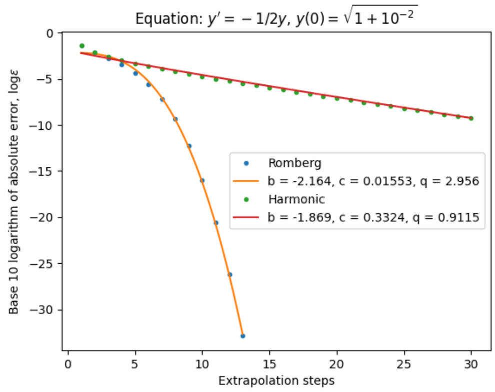

Equation: $y' = -1/2y$, $y(0) = \sqrt{2}$ Double precision
Base 10 logarithm of absolute error, $\log\varepsilon$
Number of function evaluations, N
Romberg
Harmonic

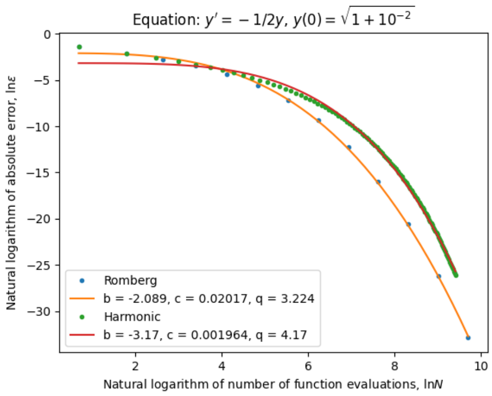

Equation: $y' = -1/2y$, $y(0) = \sqrt{2}$
Base 10 logarithm of absolute error, $\log\varepsilon$
Extrapolation steps
Romberg
b = -5.027, c = 0.7622, q = 1.893
Harmonic
b = -1.895, c = 3.152, q = 0.9944

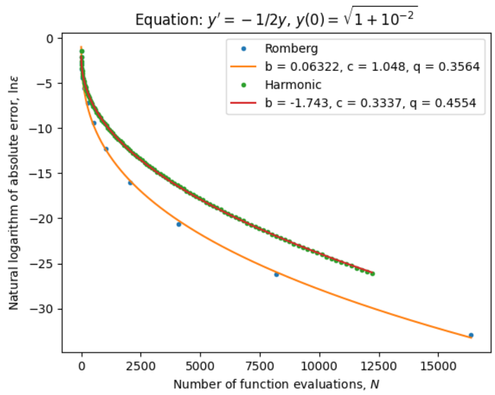

Equation: $y' = -1/2y$, $y(0) = \sqrt{2}$
Natural logarithm of absolute error, $\ln\varepsilon$
Natural logarithm of number of function evaluations, $\ln N$
Romberg
b = -4.851, c = 0.8596, q = 2.086
Harmonic
b = -17.24, c = 0.01291, q = 4.511

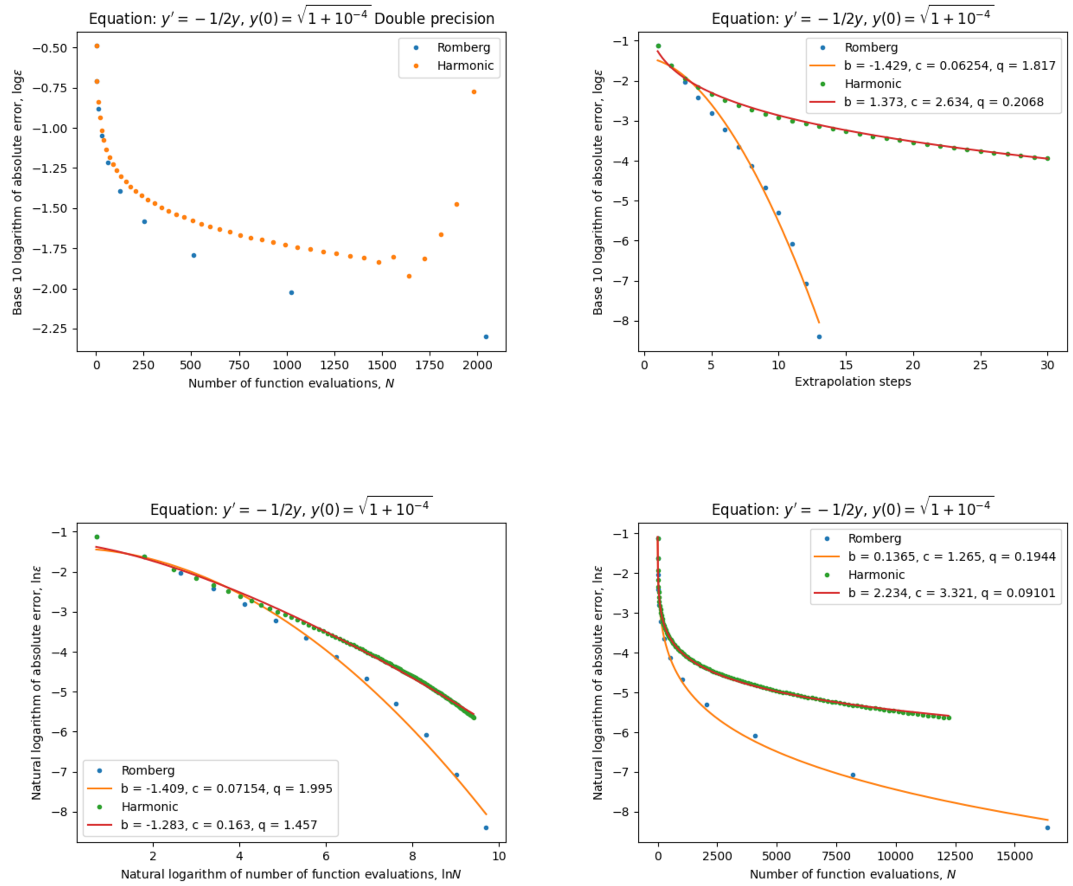

Equation: $y' = -1/2y$, $y(0) = \sqrt{2}$
Natural logarithm of absolute error, $\ln\varepsilon$
Number of function evaluations, N
Romberg
b = 17.14, c = 17.38, q = 0.2007
Harmonic
b = -0.4551, c = 3.144, q = 0.4974


Equation: $y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$ Double precision
Base 10 logarithm of absolute error, $\log\varepsilon$
Number of function evaluations, N
Romberg
Harmonic


Equation: $y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$
Base 10 logarithm of absolute error, $\log\varepsilon$
Extrapolation steps
Romberg
b = -2.164, c = 0.01553, q = 2.956
Harmonic
b = -1.869, c = 0.3324, q = 0.9115

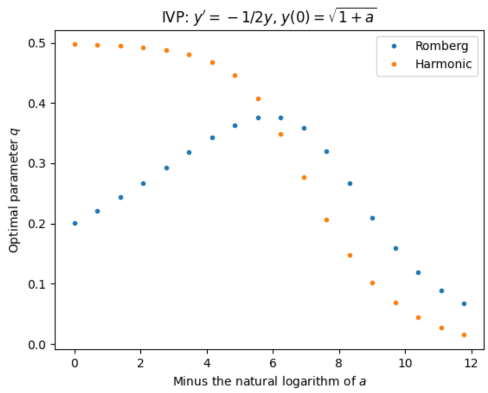

Equation: $y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$
Natural logarithm of absolute error, $\ln\varepsilon$
Natural logarithm of number of function evaluations, $\ln N$
Romberg
b = -2.089, c = 0.02017, q = 3.224
Harmonic
b = -3.17, c = 0.001964, q = 4.17


Equation: $y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$
Natural logarithm of absolute error, $\ln\varepsilon$
Number of function evaluations, N
Romberg
b = 0.06322, c = 1.048, q = 0.3564
Harmonic
b = -1.743, c = 0.3337, q = 0.4554



Here the model seems to fit in all cases. The Harmonic sequence works better for large a but the Romberg better for small a.

The plot of q against a is as follows:



4.2.6 Circular rotation

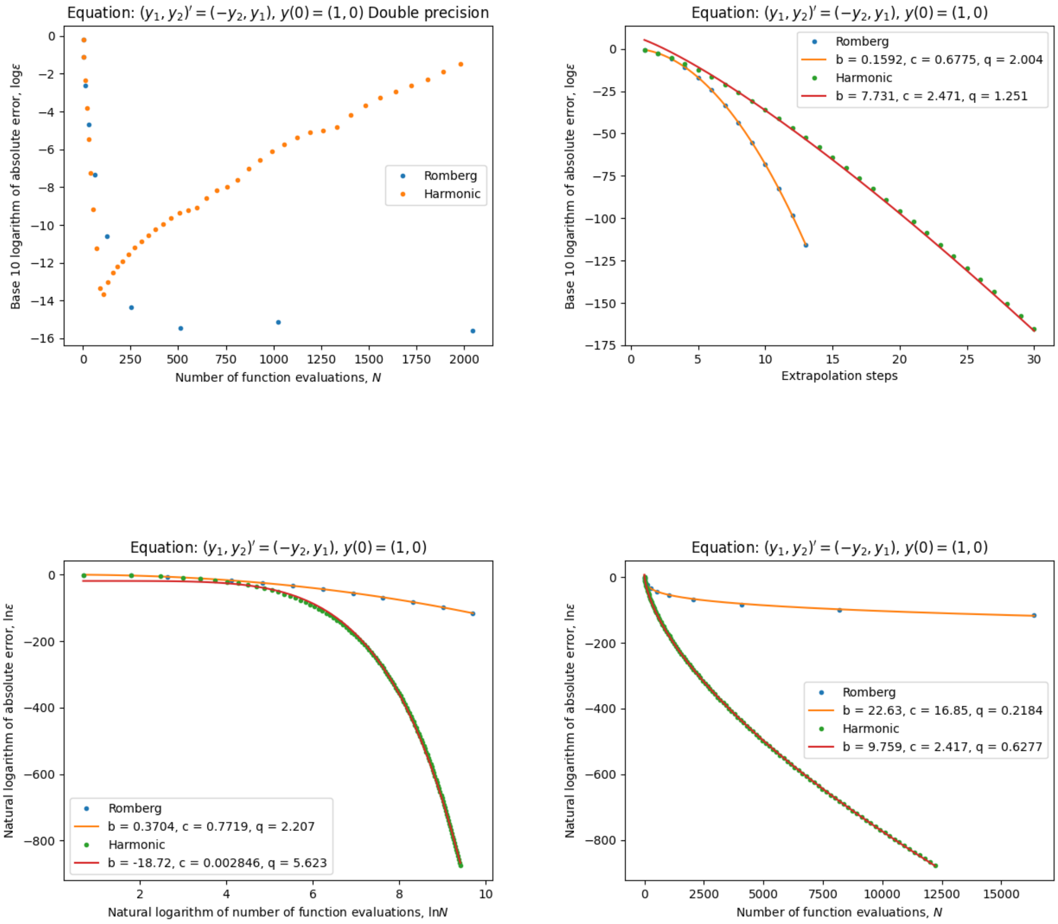
Now we will consider the following system of equations:

$$(y_1(t), y_2(t))' = (-y_2(t), y_1(t)), \quad y(0) = (1, 0), \quad t \in [0, \pi/2] \tag{4.7}$$

whose solution is

$$(y_1(t), y_2(t)) = (\cos t, \sin t)$$

which is entire.

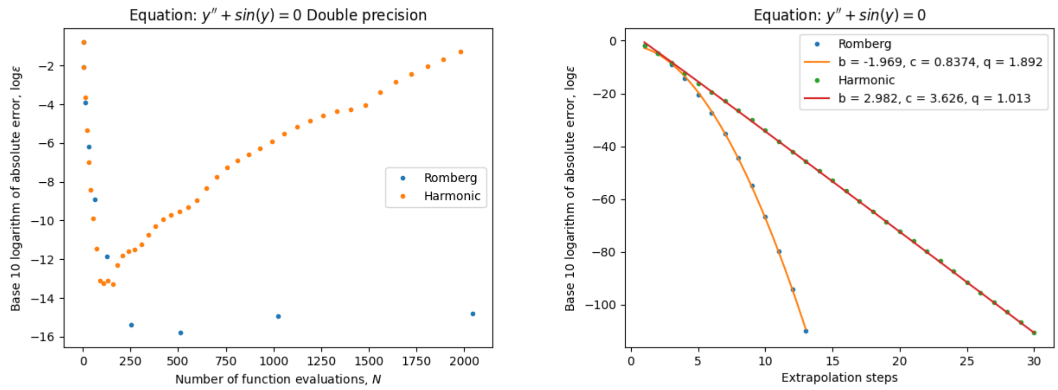


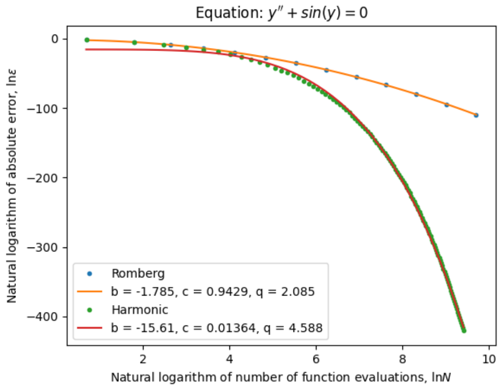
Here the model fits very well. The harmonic sequence works better then Romberg and we get down to machine level precision using either sequence when using standard floating point arithmetic.

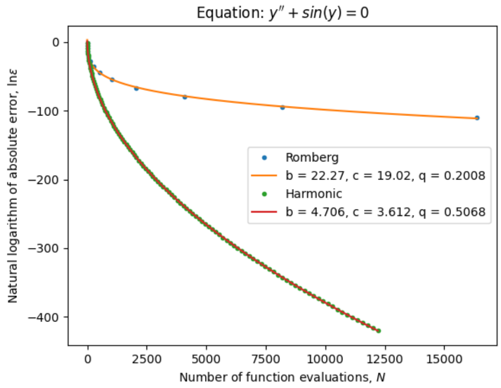
4.2.7 Mathematical pendulum

Now we will consider the mathematical pendulum equation:

$$y''(t) + \sin y(t) = 0, \quad y(0) = 0,\, y'(0) = 1, \quad t \in [0, 1]. \tag{4.8}$$







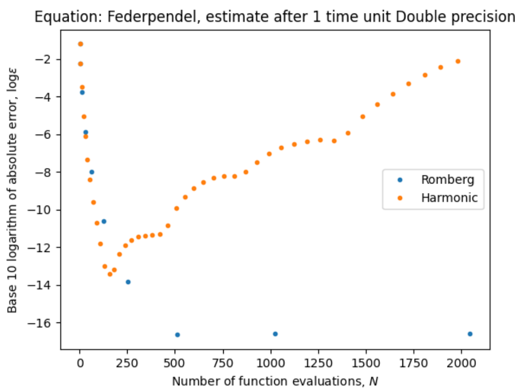
Here the model also fits very well, the harmonic sequence works better and we get down to machine level precision in standard floating point arithmetic, using either sequence.

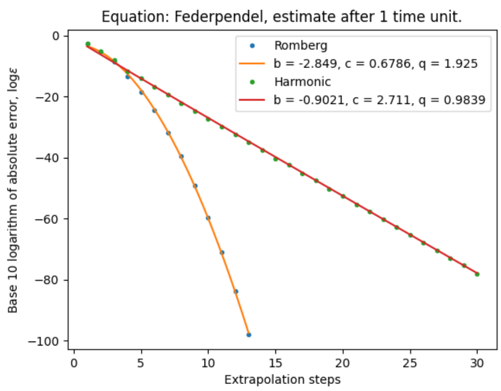
4.2.8 Federpendel

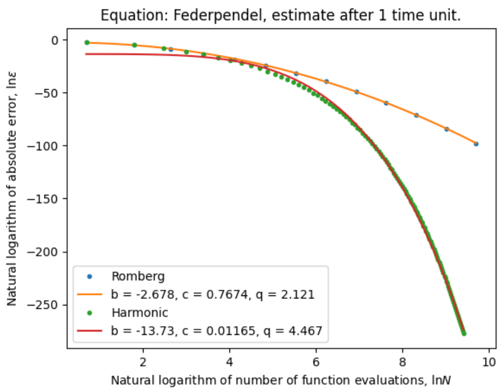
Now we will consider the equation of motion for das Federpendel or the spring pendulum:

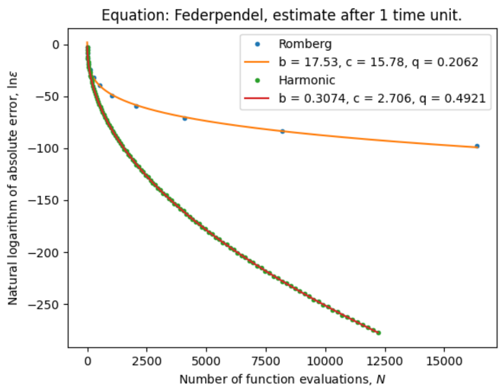
$$\mathbf{p}' = -(|\mathbf{q}| - 1)\frac{\mathbf{q}}{|\mathbf{q}|} - \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{q}' = \mathbf{p}$$

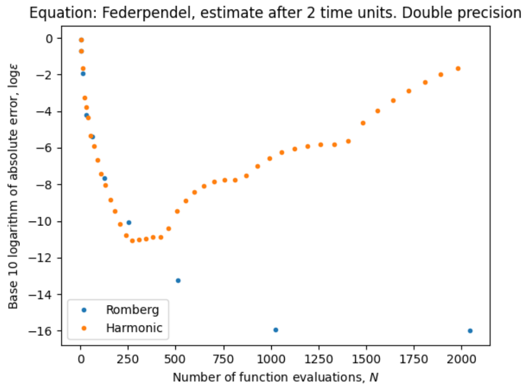
where $\mathbf{p}$ and $\mathbf{q}$ are two dimensional vectors. We will consider it with the initial condition $\mathbf{q}(0) = (1, 0)$ and $\mathbf{p}(0) = (0, 1)$ and try to both estimate the solution at time $t = 1$ and time $t = 2$.

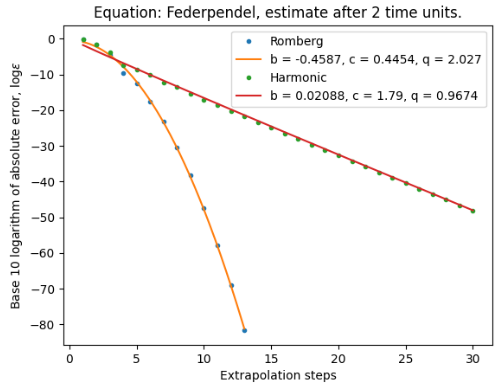


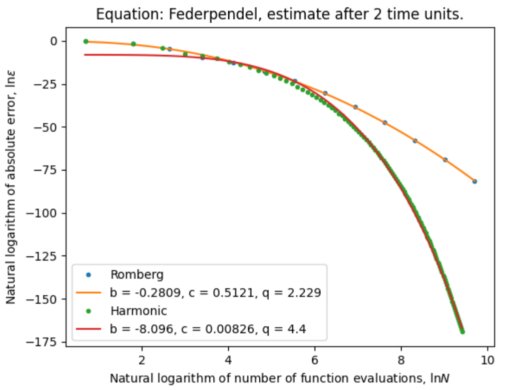


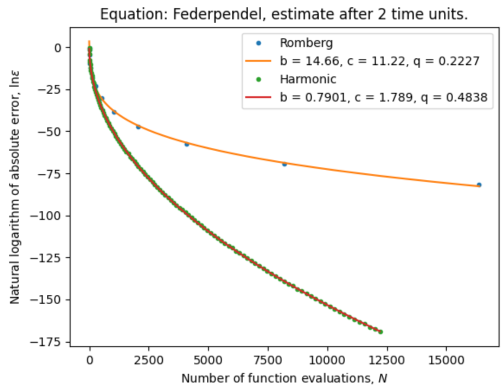












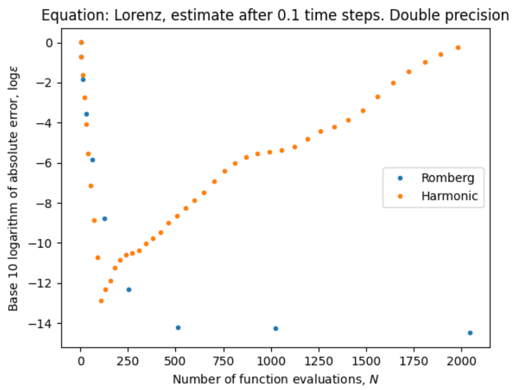
The model fits very well in all cases, the harmonic sequence works better. In standard floating point arithmetic, we obtain higher accurracy using the Romberg sequence, though we get high accurracy in both cases.

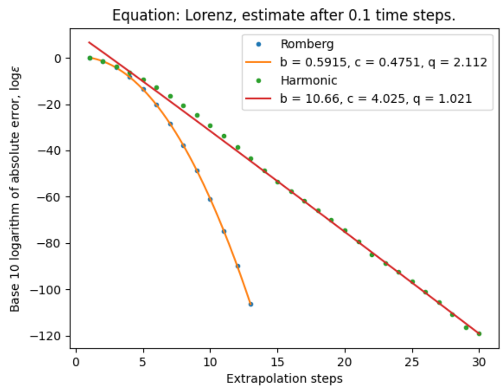
4.2.9 Lorenz equations

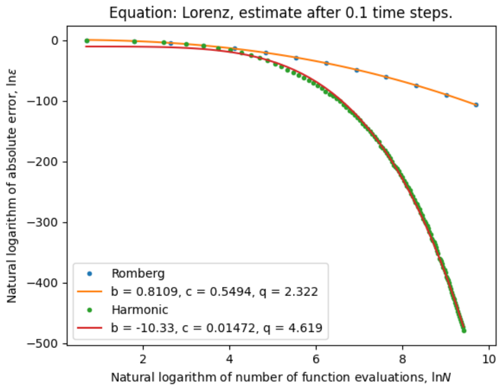
The Lorenz equations are the following system:

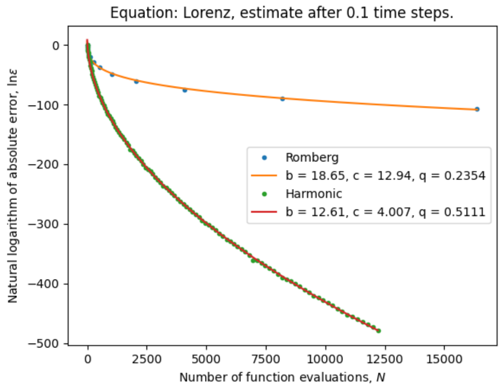
$$\frac{dx}{dt} = \sigma(y - x), \quad \frac{dy}{dt} = x(\rho - z) - y, \quad \frac{dz}{dt} = xy - \beta z$$

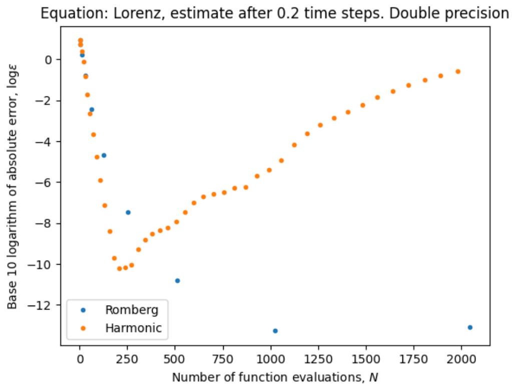
where σ, ρ and β are constants. In our experiment, the constants are set to $\sigma = 10$, $\rho = 28$ and $\beta = 8/3$. The initial condition we will consider is $(x(0), y(0), z(0)) = (1, 1, 1)$.

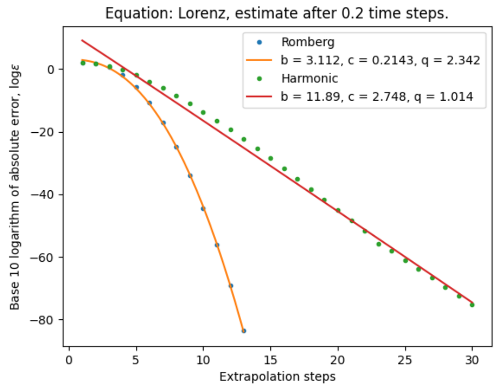


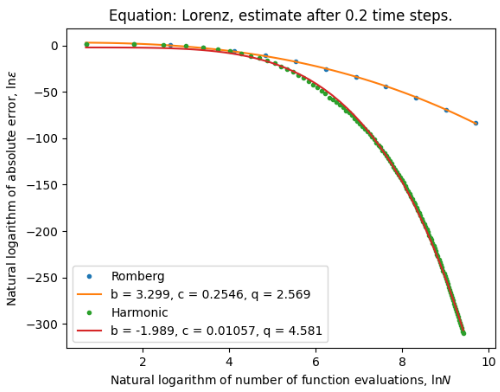


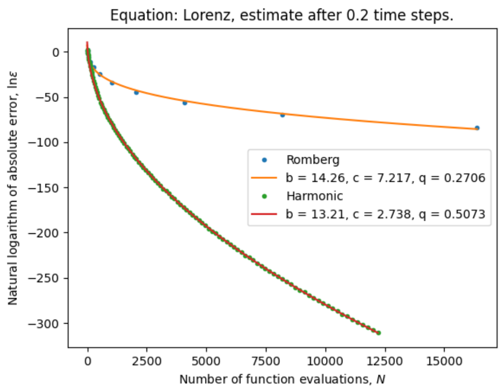












The model fits very well in both cases when we consider the number of evaluations against error. The harmonic sequence works better. The fitting is not as nice when considering the number of extrapolation seps against the error. In standard floating point arithmetic, we obtain higher accurracy using the Romberg sequence, though we get high accurracy in both cases.

The values of the optimal parameters in the fitting of the number evaluations against the error, are:

IVP	Sequence	b	c	q
$y' = y$, $y(0) = 0$	Romberg	27.187	22.015	0.20227
$y' = y$, $y(0) = 0$	Harmonic	8.8124	3.0433	0.61436
$y' = y(1-y)$	Romberg	17.604	19.237	0.20418
$y' = y(1-y)$	Harmonic	1.7927	4.0933	0.51064
$y' = 1 + y^2$, $y(0) = 0$	Romberg	11.465	8.8186	0.24549
$y' = 1 + y^2$, $y(0) = 0$	Harmonic	3.3496	1.8647	0.51993
$(y_1, y_2)' = (-y_2, y_1)$, $y(0) = (1, 0)$	Romberg	22.626	16.846	0.2184
$(y_1, y_2)' = (-y_2, y_1)$, $y(0) = (1, 0)$	Harmonic	9.7592	2.4171	0.62765
$y' = y^2$, $y(0) = 1/2$	Romberg	14.78	12.517	0.22455
$y' = y^2$, $y(0) = 1/2$	Harmonic	3.5994	2.5894	0.51472
$y' = y^2$, $y(0) = 1/(1 + 10^{-2})$	Romberg	5.3983	0.1961	0.4989
$y' = y^2$, $y(0) = 1/(1 + 10^{-2})$	Harmonic	4.7796	0.032416	0.6592
$y' = y^2$, $y(0) = 1/(1 + 10^{-4})$	Romberg	9.212	0.00012308	0.94461
$y' = y^2$, $y(0) = 1/(1 + 10^{-4})$	Harmonic	9.2104	0.00016925	0.74975
$y' = -1/2y$, $y(0) = \sqrt{2}$	Romberg	17.142	17.376	0.2007
$y' = -1/2y$, $y(0) = \sqrt{2}$	Harmonic	-0.45512	3.1436	0.49744
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$	Romberg	0.063222	1.0479	0.35645
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$	Harmonic	-1.7425	0.33371	0.45544
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-4}}$	Romberg	0.13652	1.2653	0.19436
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-4}}$	Harmonic	2.2343	3.3206	0.091009
$y'' + \sin(y) = 0$	Romberg	22.275	19.017	0.20078
$y'' + \sin(y) = 0$	Harmonic	4.7064	3.6116	0.50678
Federpendel, estimate after 1 time unit.	Romberg	17.532	15.778	0.20624
Federpendel, estimate after 1 time unit.	Harmonic	0.30737	2.706	0.49211
Federpendel, estimate after 2 time units.	Romberg	14.66	11.217	0.22272
Federpendel, estimate after 2 time units.	Harmonic	0.79006	1.789	0.48379
Lorenz, estimate after 0.1 time steps.	Romberg	18.654	12.939	0.2354
Lorenz, estimate after 0.1 time steps.	Harmonic	12.615	4.0068	0.5111
Lorenz, estimate after 0.2 time steps.	Romberg	14.264	7.2173	0.27063
Lorenz, estimate after 0.2 time steps.	Harmonic	13.212	2.7376	0.50732

Table 4.1: Optimal parameters by test case

We note that in those cases where the singularities of the solutions are not very close to our time interval, then q is close to 0.5 for the harmonic sequence and close to 0.2 for the Romberg sequence.

The values of the optimal parameters in the fitting of the number of extrapolation steps against the error, are:

IVP	Sequence	b	c	q
$y' = y$, $y(0) = 0$	Romberg	-1.0083	0.9557	1.904
$y' = y$, $y(0) = 0$	Harmonic	6.425	3.1061	1.225
$y' = y(1-y)$	Romberg	-7.1901	0.82672	1.916
$y' = y(1-y)$	Harmonic	-0.19572	4.1127	1.0204
$y' = 1 + y^2$, $y(0) = 0$	Romberg	-1.2103	0.30542	2.1771
$y' = 1 + y^2$, $y(0) = 0$	Harmonic	2.405	1.8763	1.0387
$(y_1, y_2)' = (-y_2, y_1)$, $y(0) = (1, 0)$	Romberg	0.15924	0.67746	2.004
$(y_1, y_2)' = (-y_2, y_1)$, $y(0) = (1, 0)$	Harmonic	7.7307	2.4712	1.2513
$y' = y^2$, $y(0) = 1/2$	Romberg	-2.193	0.48799	2.0424
$y' = y^2$, $y(0) = 1/2$	Harmonic	2.3181	2.6033	1.0284
$y' = y^2$, $y(0) = 1/(1 + 10^{-2})$	Romberg	4.6269	0.00074557	4.042
$y' = y^2$, $y(0) = 1/(1 + 10^{-2})$	Harmonic	4.7474	0.033263	1.3137
$y' = y^2$, $y(0) = 1/(1 + 10^{-4})$	Romberg	-2507.7	-2517.1	-0.00010114
$y' = y^2$, $y(0) = 1/(1 + 10^{-4})$	Harmonic	9.2101	0.00017545	1.4929
$y' = -1/2y$, $y(0) = \sqrt{2}$	Romberg	-5.0269	0.76221	1.8929
$y' = -1/2y$, $y(0) = \sqrt{2}$	Harmonic	-1.8945	3.1518	0.99437
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$	Romberg	-2.1638	0.015532	2.9565
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-2}}$	Harmonic	-1.8689	0.33241	0.91148
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-4}}$	Romberg	-1.4291	0.062542	1.8173
$y' = -1/2y$, $y(0) = \sqrt{1 + 10^{-4}}$	Harmonic	1.3726	2.6336	0.20682
$y'' + \sin(y) = 0$	Romberg	-1.9686	0.83745	1.8917
$y'' + \sin(y) = 0$	Harmonic	2.9822	3.6265	1.0128
Federpendel, estimate after 1 time unit.	Romberg	-2.8486	0.67856	1.9249
Federpendel, estimate after 1 time unit.	Harmonic	-0.90208	2.7108	0.98385
Federpendel, estimate after 2 time units.	Romberg	-0.45874	0.44537	2.0272
Federpendel, estimate after 2 time units.	Harmonic	0.020877	1.79	0.96743
Lorenz, estimate after 0.1 time steps.	Romberg	0.59151	0.47509	2.112
Lorenz, estimate after 0.1 time steps.	Harmonic	10.659	4.0254	1.0213
Lorenz, estimate after 0.2 time steps.	Romberg	3.1116	0.21429	2.3417
Lorenz, estimate after 0.2 time steps.	Harmonic	11.891	2.7477	1.014

Table 4.2: Optimal parameters by test case

Bibliography

[1] Perter Deuflhard and Andreas Hohmann. *Numerical Analysis in Modern Scientific Computing*, vol. 43 of Texts in Applied Mathematics, Springer, New York, 2003.

[2] Konrad Knopp. *Theorie und Anwendung der unendlichen Reihen.*, Springer Verlag, Berlin, Heidelberg, New York, (5. Auflage) 1964.