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 $\tau_p, \tau_{2p}, \ldots, \tau_{mp} \in \mathbb{R}$ 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 $(x_1, y_1), \ldots, (x_k, y_k)$ be a collection of points such that $x_1, \ldots, x_k$ 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 $(x_1^p, y_1), \ldots, (x_n^p, y_n)$ 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 $h_1, h_2, \ldots, h_k$ where $k \leq m$. Then we compute the interpolation polynomial P in h^p which goes through $(h_1, T(h_1)), \ldots, (h_k, T(h_k))$. We then hope that $P(0)$ gives a good approximation T_0.

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

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

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 $\varepsilon_k := |T_{kk} - T|$. 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.

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

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

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

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. The cross validation strategy we will use goes as follows: Suppose that we have done a curve fitting on (x_k, y_k) for $k = 0, 1, \ldots, n$. Then we will do the curve fitting for $(x_{k+3}, y_{k+3}), \ldots, (x_{k+9}, x_{k+9})$ for $k = 0, \ldots, n-6$ and compute the relative variance of the parameters. Let a_k, $k = 1, \ldots, m$ be numbers. Then we define their mean value by $\overline{a} := \frac{1}{m} \sum_{k=1}^{m} a_k$, and the relative variance by

$$\frac{1}{m\overline{a}^2} \sum_{k=1}^{m} (a_k - \overline{a})^2.$$

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. In all cases we will try to fit the points on curve of a function of the form $t \mapsto b - ct^q$ and we will report the mean and relative variance of $A \coloneqq e^b$, c and q.

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.

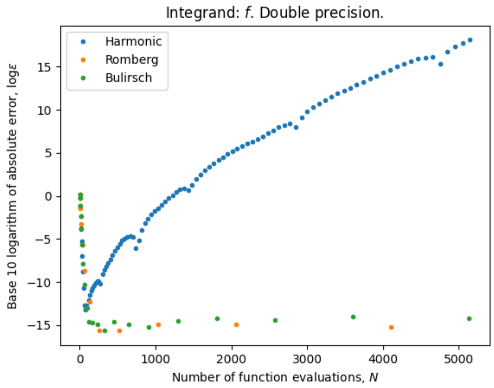
Now we will consider the results of the experiments.

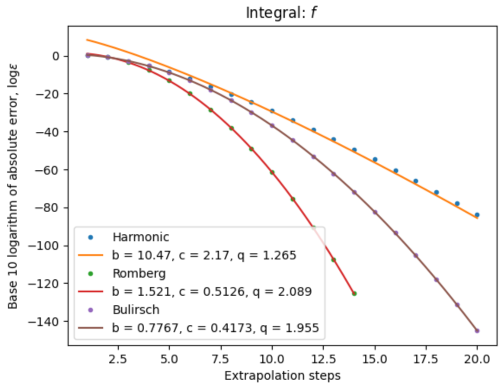
2.2.1 Cosine squared

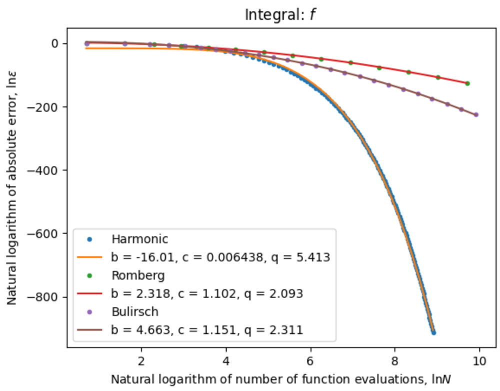
The first function we are going to try is

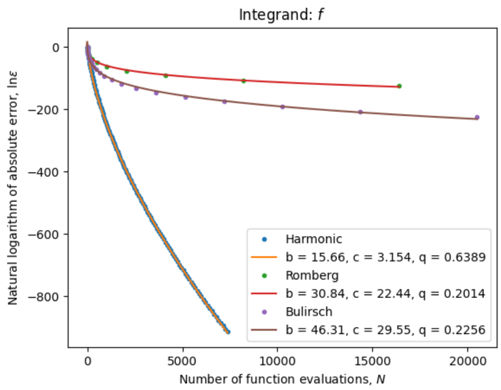
$$f : [0,\pi] \to \mathbb{R}, \quad f(x) \coloneqq \cos^2(x)$$

which is entire.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	$3.698 \cdot 10^{18}$	15.89	3.431	0.05254	0.6382	0.00378
Romberg	lin-ln evals-error	$3.478 \cdot 10^{31}$	4	34.93	0.1045	0.1765	0.02939
Bulirsch	lin-ln evals-error	$1.44 \cdot 10^{73}$	14.91	50.79	0.4318	0.2315	0.2026
Harmonic	lin-ln steps-error	$2.979 \cdot 10^{14}$	14.48	2.346	0.03992	1.261	0.002109
Romberg	lin-ln steps-error	10.06	0.06604	0.5486	0.001087	2.066	$3.842 \cdot 10^{-5}$
Bulirsch	lin-ln steps-error	1.601	0.1852	0.4143	0.001942	1.956	$5.581 \cdot 10^{-5}$
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	70.67	0.03149	1.31	0.0005279	2.022	$2.887 \cdot 10^{-5}$
Bulirsch	ln-ln evals-error	$4.442 \cdot 10^{5}$	2.965	1.416	0.08864	2.274	0.007141

We see that the harmonic sequence performes best, then Bulirsch and then Romberg. In standard double precision arithmetic, we get down to machine level precision using Romberg or Bulirsch, but we are like 2 digits from there, using the harmonic sequence.

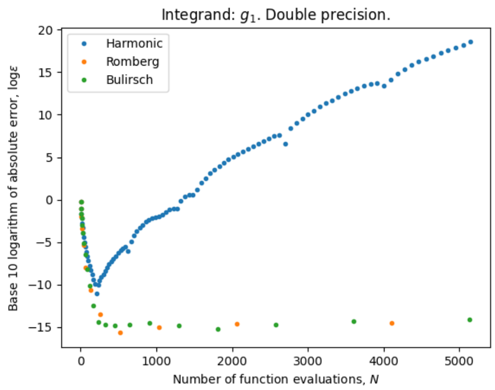
For the Romberg sequence we clearly have exponential convergence in the number of extrapolation steps and thus we also get a good fitting for the ln-ln graph. We also have exponential convergence in the number of steps for the Bulirsch sequence.

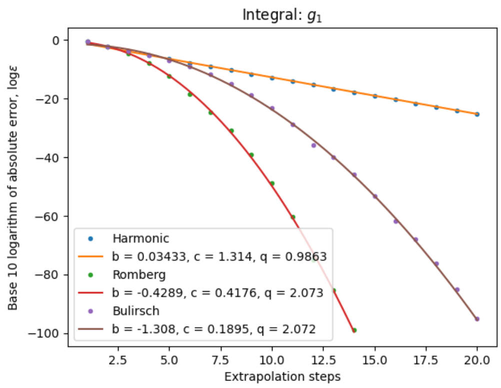
The model does not seem to fit well in any case for the Harmonic sequence since since we unreasonably big values for A in the first two cases and the curve fitting does not even converge in all cases for the last plot.

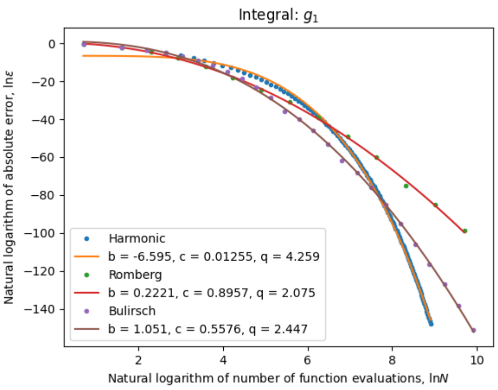
2.2.2 Function with poles

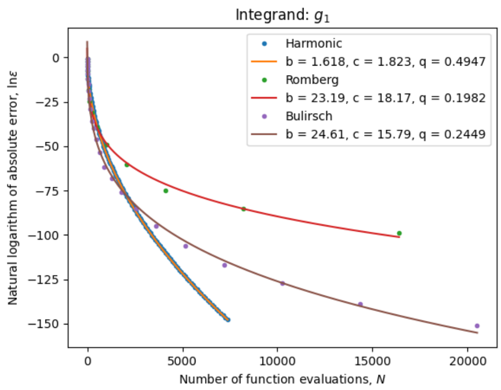
Now we will consider the following function:

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





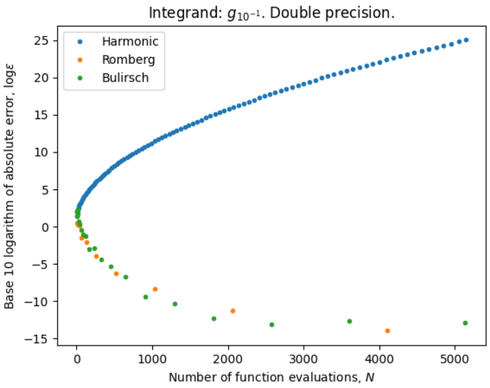


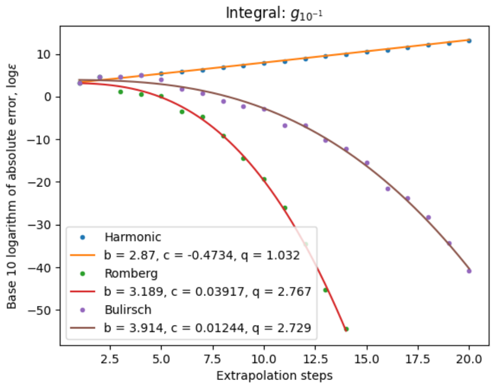


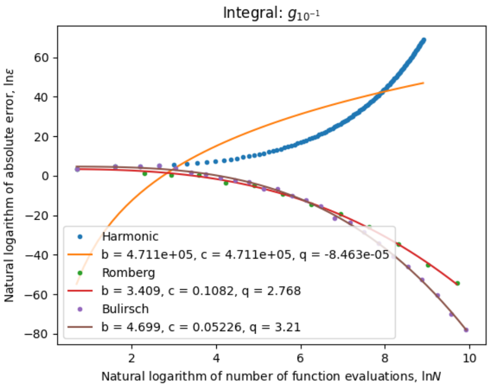
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	3.832	0.09139	1.801	0.0006569	0.4956	$3.963 \cdot 10^{-5}$
Romberg	lin-ln evals-error	$1.638 \cdot 10^{29}$	4	25.01	0.3364	0.194	0.08516
Bulirsch	lin-ln evals-error	.	.	146.8	3.207	0.2756	0.5871
Harmonic	lin-ln steps-error	0.691	0.1736	1.288	0.001823	0.9894	0.0001036
Romberg	lin-ln steps-error	9.757	1.881	0.4355	0.5242	2.21	0.03547
Bulirsch	lin-ln steps-error	$3.958 \cdot 10^{25}$	15	0.8647	1.779	2.106	0.09976
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	53.88	1.418	1.013	0.4567	2.166	0.0389
Bulirsch	ln-ln evals-error	$2.619 \cdot 10^{36}$	15	2.987	1.774	2.483	0.1401

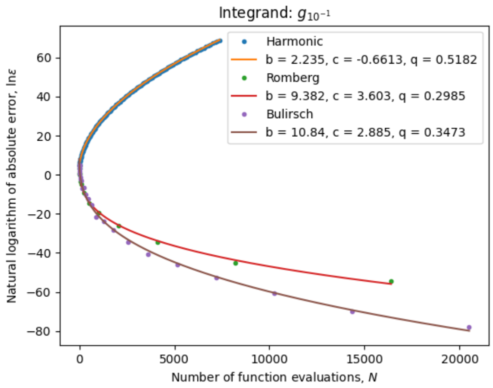
We see that the harmonic sequence performes best, then Bulirsch and then Romberg. In standard double precision arithmetic, we get down to machine level precision using Romberg or Bulirsch, but we are like 5 digits from there, using the harmonic sequence.

Here we clearly have exponential convergence in the number of evaluations for the Harmonic sequence and hence also in the number of steps. The models do not seem to fit very nicely in the other cases.



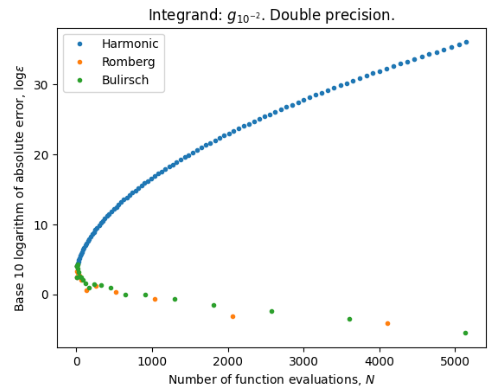


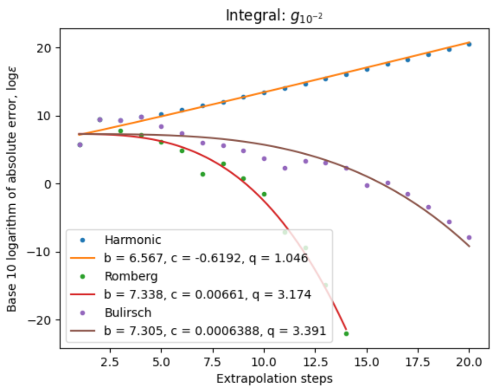


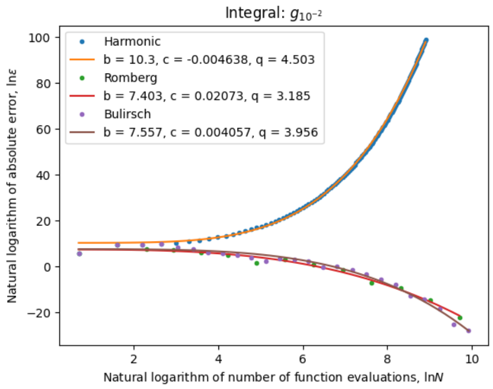


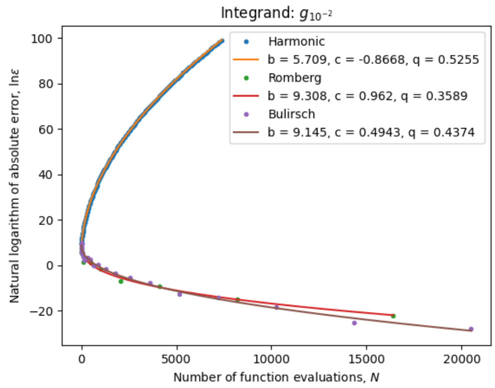
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	6.15	0.6637	-0.6969	0.01918	0.5172	0.003167
Romberg	lin-ln evals-error	$3.02 \cdot 10^7$	3.985	3.897	0.3031	0.3265	0.07301
Bulirsch	lin-ln evals-error	.	.	1123	14.56	0.3275	0.744
Harmonic	lin-ln steps-error	11.9	0.2941	-0.496	0.01497	1.029	0.00189
Romberg	lin-ln steps-error	35.64	1.539	0.03496	0.4962	2.934	0.01283
Bulirsch	lin-ln steps-error	.	.	844.6	14.99	2.519	0.2989
Harmonic	ln-ln evals-error	.	.	$4.863 \cdot 10^5$	0.03858	-0.0002903	0.3799
Romberg	ln-ln evals-error	59.88	1.575	0.1019	0.4164	2.915	0.0144
Bulirsch	ln-ln evals-error	.	.	1069	14.99	2.965	0.3358

Here we get divergence for the harmonic sequence, but convergence for the other sequences, fastest for Bulirsch. In standard double precision arithmetic, we get down to machine level precision using Romberg or Bulirsch. None of the models fits.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	268	5.059	-0.9277	0.0424	0.5267	0.007106
Romberg	lin-ln evals-error	.	.	3150	2.785	0.3352	1.082
Bulirsch	lin-ln evals-error	.	.	2763	2.795	0.6219	2.844
Harmonic	lin-ln steps-error	519.6	2.663	-0.659	0.03722	1.047	0.004672
Romberg	lin-ln steps-error	$9.388 \cdot 10^{14}$	4	4.469	3.191	3.087	0.6264
Bulirsch	lin-ln steps-error	.	.	1615	3.548	4.057	1.73
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	$2.102 \cdot 10^{56}$	4	23.8	3.7	3.032	0.6855
Bulirsch	ln-ln evals-error	.	.	1824	3.52	4.841	1.87

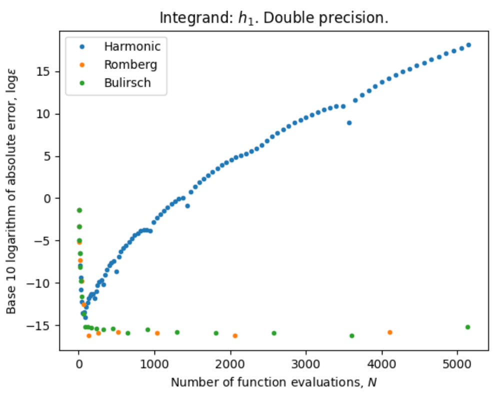
Here the same comments apply as for $a = 10^{-1}$, except that now the Romberg sequence performes better than the Bulirsch sequence and the model fitting is worse.

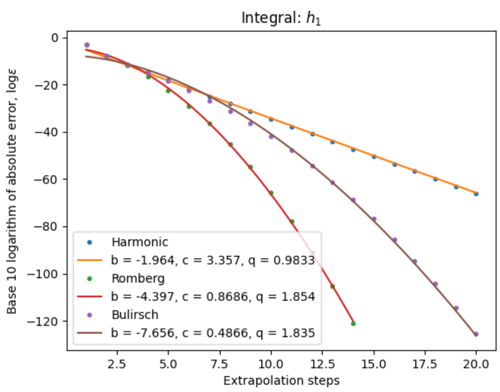
2.2.3 Logarithm

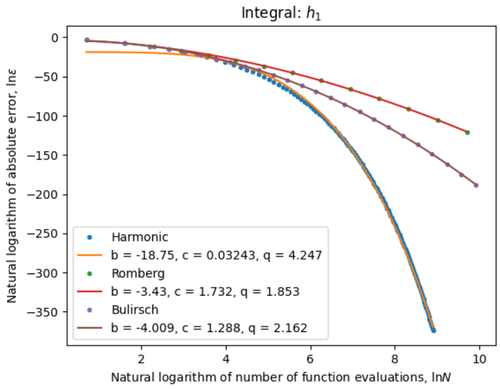
Now we will consider the following function

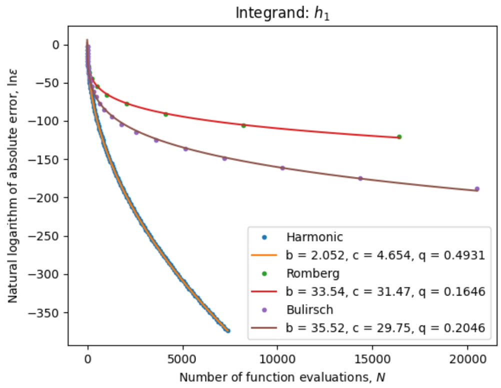
$$h_a : [0,1] \to \mathbb{R}, \quad h_a(x) := \ln(a + x),\ a > 0.$$

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





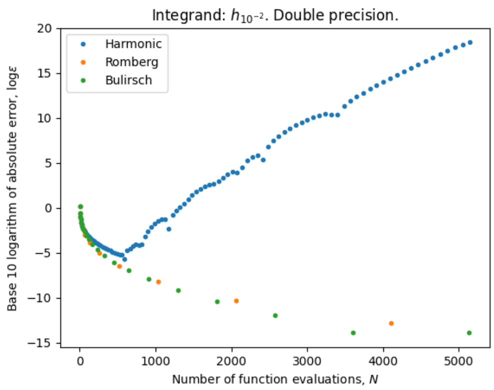


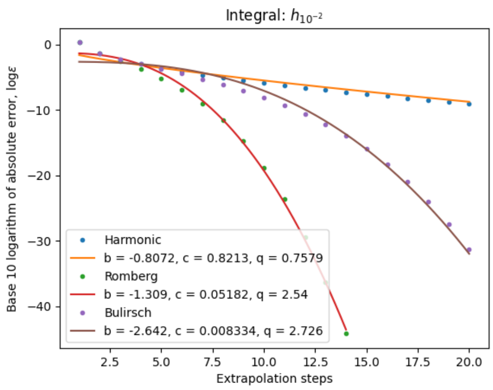


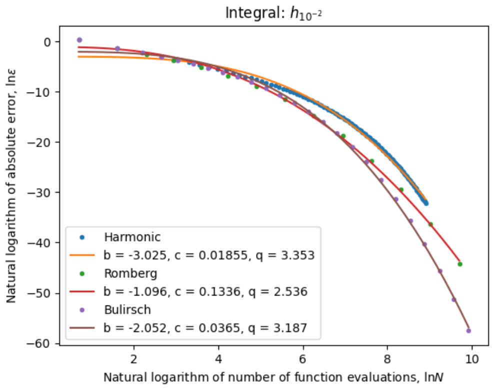
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	4.305	1.625	4.577	0.001332	0.4944	$8.445 \cdot 10^{-5}$
Romberg	lin-ln evals-error	$4.448 \cdot 10^{23}$	3.996	37.1	0.03467	0.1563	0.009509
Bulirsch	lin-ln evals-error	$1.085 \cdot 10^{43}$	13.73	38.32	0.2396	0.2067	0.06998
Harmonic	lin-ln steps-error	0.06861	3.33	3.274	0.002897	0.9871	0.000164
Romberg	lin-ln steps-error	0.0007205	0.9454	0.6403	0.01862	1.96	0.0006732
Bulirsch	lin-ln steps-error	0.0001991	8.396	0.4164	0.3211	1.901	0.006898
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	0.006917	1.189	1.488	0.01968	1.912	0.001013
Bulirsch	ln-ln evals-error	0.007191	0.3683	1.209	0.003807	2.187	0.0001579

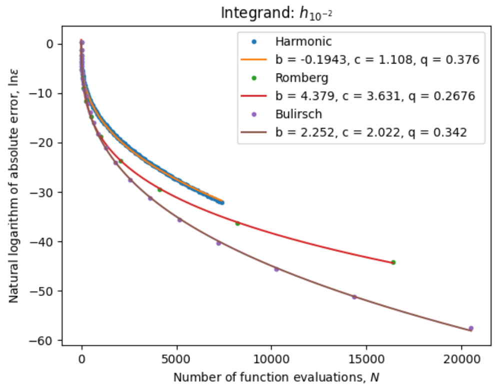
We see that the harmonic sequence performes best, then Bulirsch and then Romberg. In standard double precision arithmetic, we get down to machine level precision using Romberg or Bulirsch, but we are like 2 digits from there, using the harmonic sequence.

Here we clearly have exponential convergence in the number of evaluations for the harmonic sequence and hence also in the number of steps. For Romberg and Bulirsch we seem to have exponential convergence in the number of steps.







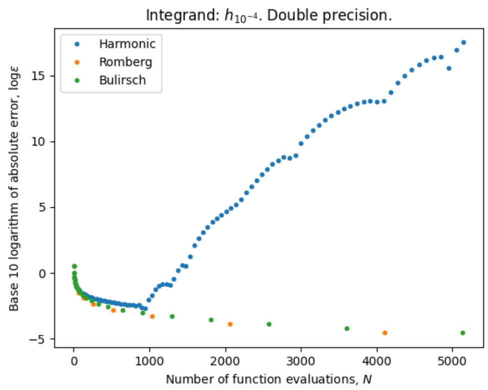


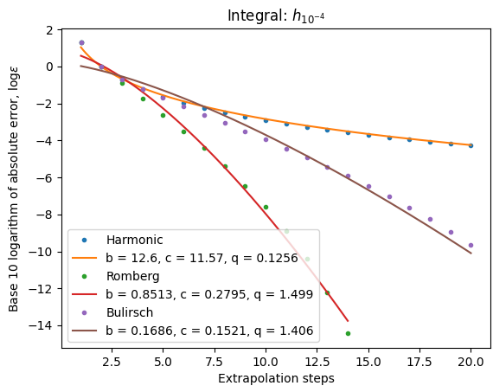
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	65.31	75.17	0.9698	1.442	0.4141	0.03051
Romberg	lin-ln evals-error	62.93	1.966	3.02	0.02769	0.2887	0.002777
Bulirsch	lin-ln evals-error	1421	10.21	2.024	0.1564	0.3492	0.01221
Harmonic	lin-ln steps-error	1.245	26.82	0.6627	1.048	0.8362	0.02436
Romberg	lin-ln steps-error	0.05696	0.5511	0.03227	0.3681	2.751	0.006105
Bulirsch	lin-ln steps-error	0.04768	2.868	0.03603	3.54	2.666	0.04769
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	0.08097	0.6227	0.09113	0.2618	2.722	0.006305
Bulirsch	ln-ln evals-error	0.08405	1.957	0.06538	1.649	3.144	0.02837

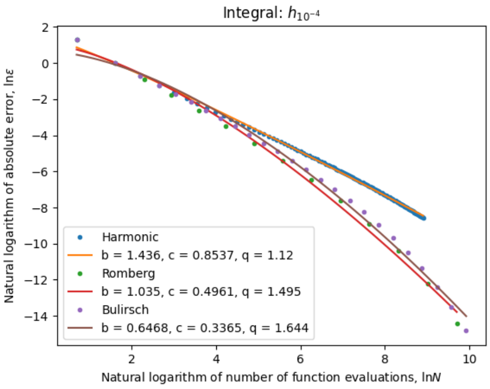
We see that we can not attain high precision using the harmonic sequence and standard double precision. It is hard to tell which sequence performes best in the long run, though we can say that Bulirsch performes better than Romberg.

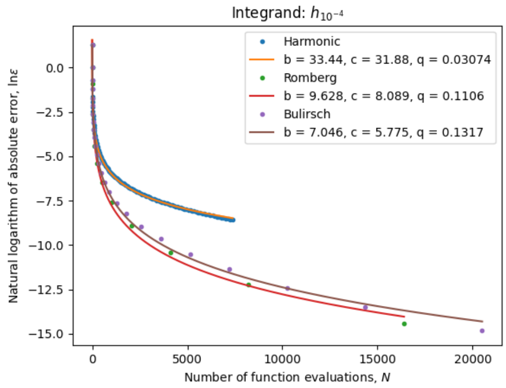
For Romberg, we get low variance in all cases. But since we can not have both exponential convergence in the number of steps and in the number of function evaluations we can not say anything definite.

For the other sequences, the results are not clear, unless that the last model clearly does not fit for the Harmonic sequence.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln evals-error	$1.218 \cdot 10^{10}$	4	9.792	0.587	0.119	0.2065
Bulirsch	lin-ln evals-error	$4.389 \cdot 10^{6}$	3.871	7.595	0.5385	0.1447	0.3134
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	0.8009	0.7945	0.1727	0.6742	1.81	0.05434
Bulirsch	lin-ln steps-error	0.551	1.326	0.1412	1.37	1.807	0.1569
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	1.356	1.045	0.3691	0.618	1.751	0.06504
Bulirsch	ln-ln evals-error	1.319	1.017	0.3206	0.7671	2.009	0.1528

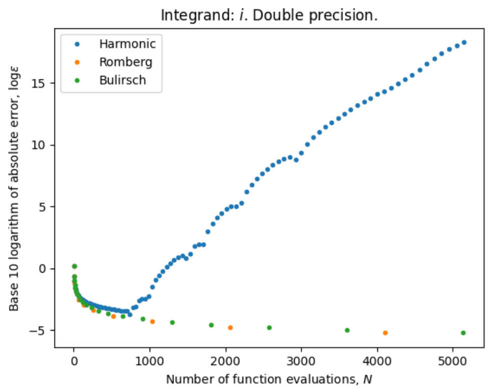
Here again, we do not attain high precision when using the Harmonic sequence in double precision arithmetic. It is hard to say which sequence performes best. None of our models seems to fit.

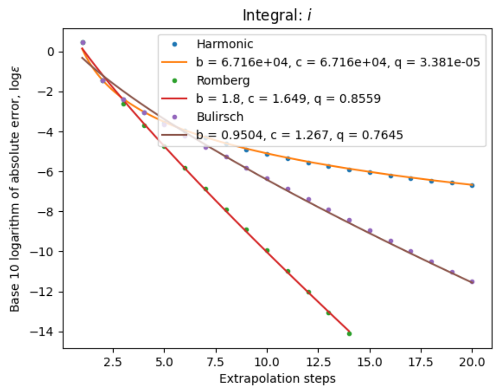
2.2.4 Area of half circle

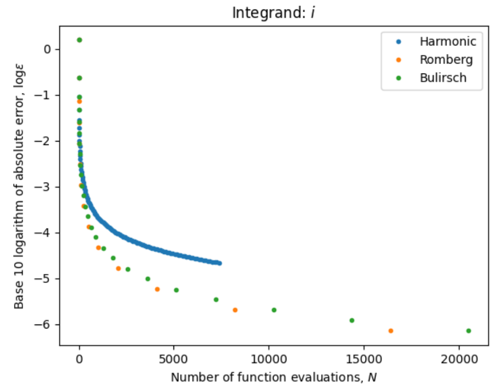
Now we will try the following function:

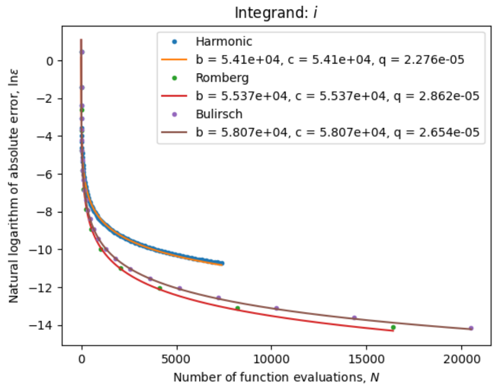
$$i : [-1, 1] \to \mathbb{R}, \quad i(x) := \sqrt{1 - x^2}.$$

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









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	.	.	$5.198 \cdot 10^4$	0.002568	$2.252 \cdot 10^{-5}$	0.001907
Romberg	lin-ln evals-error	.	.	$5.666 \cdot 10^4$	0.005933	$2.726 \cdot 10^{-5}$	0.004682
Bulirsch	lin-ln evals-error	.	.	$5.991 \cdot 10^4$	0.001785	$2.548 \cdot 10^{-5}$	0.001618
Harmonic	lin-ln steps-error	.	.	$6.686 \cdot 10^4$	0.0003218	$3.369 \cdot 10^{-5}$	0.0003045
Romberg	lin-ln steps-error	1.624	0.001777	1.056	0.0002564	0.995	$2.782 \cdot 10^{-5}$
Bulirsch	lin-ln steps-error	0.4011	0.1625	0.5825	0.0448	0.9709	0.003052
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	4.527	0.1118	1.939	0.008599	0.9166	0.001262
Bulirsch	ln-ln evals-error	3.499	0.05917	1.649	0.007759	0.9758	0.001453

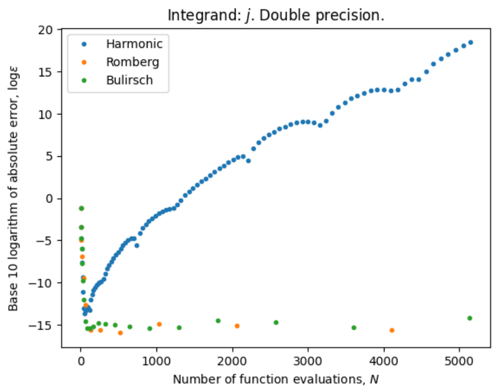
We see that we do not get high precision using double precision arithmetic, independent of sequence. The Romberg and Bulirsch sequence seem to perform similarly well but the harmonic sequence seems to be slowest.

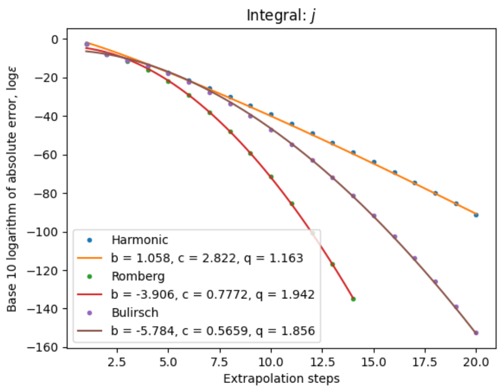
For the harmonic sequence, the error neither converges exponentally with the number of function evaluations nor the number of extrapolation steps. For Romberg we get a nice fit for the exponential convergence in number of steps. For Bulirsch neither we do not have exponential convergence in number of evaluations, and the results are unclear for the number of steps.

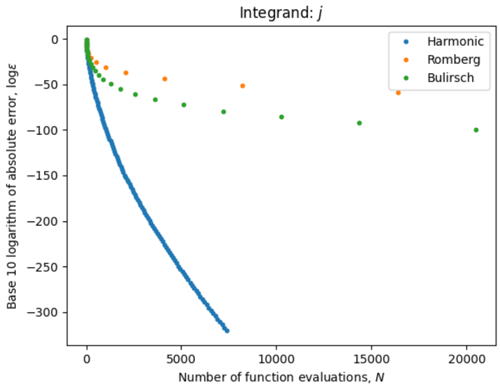
2.2.5 Gaussian

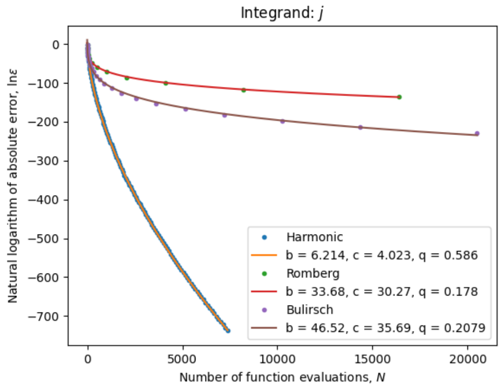
Finally we will consider the Gaussian function

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









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln evals-error	$5.057 \cdot 10^{30}$	4	41.2	0.07555	0.162	0.02295
Bulirsch	lin-ln evals-error	.	.	80.86	2.754	0.2109	0.1757
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	0.005845	0.02304	0.6857	0.0002576	1.986	$8.973 \cdot 10^{-6}$
Bulirsch	lin-ln steps-error	$3.072 \cdot 10^{23}$	15	0.6686	1.172	1.88	0.008732
Harmonic	ln-ln evals-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	0.05733	0.1689	1.603	0.001578	1.939	$7.621 \cdot 10^{-5}$
Bulirsch	ln-ln evals-error	$1.149 \cdot 10^{37}$	15	2.172	1.422	2.171	0.01192

In double precision arithmetic we get down to machine level precision using Romberg or Bulirsch, but we get down to like 2 digits from there, using the harmonic sequence. The harmonic sequence performes best, then Bulirsch and then Romberg.

For the Harmonic sequence, none of the models fits. For the Romberg sequence, we clearly have exponential convergence in the number of extrapolation steps. For Bulirsch, we clearly do not have exponential convergence in the number of evaluations and fitting of the other models should be rejected since the A constant is unreasonably big in there is a high variance in c.

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 $2 \leq j \leq i$.

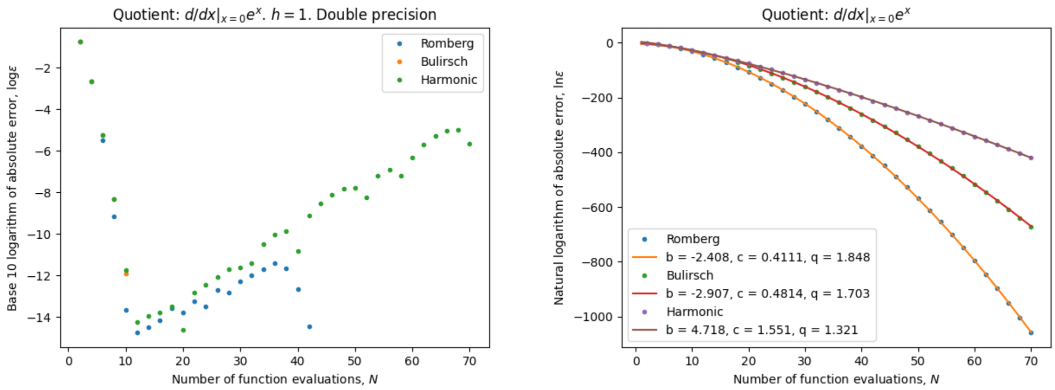
3.2 Numerical experiments

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.



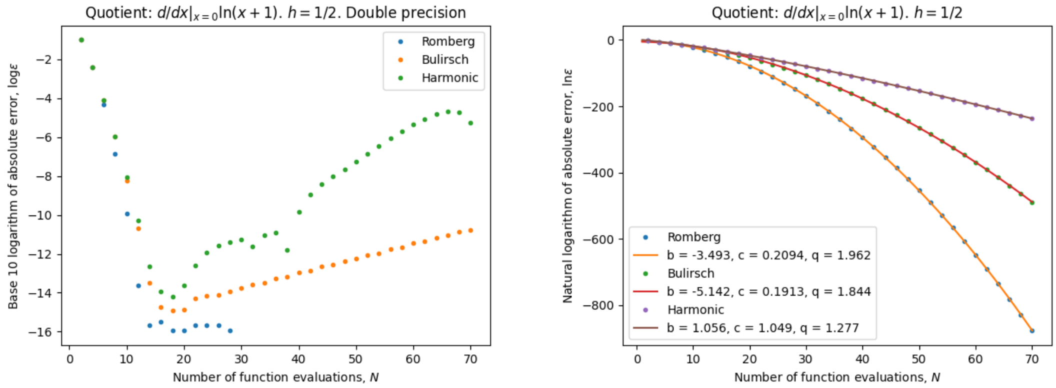
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	0.1079	3.508	0.4046	0.01413	1.85	0.0002493
Bulirsch	lin-ln evals-error	0.08951	4.763	0.463	0.04109	1.712	0.0008086
Harmonic	lin-ln evals-error	$4.149 \cdot 10^6$	6.499	1.734	0.02291	1.303	0.000902

In standard floating point arithmetic, we get down to machine level precision using any sequence. The Romberg sequence works best, then Bulirsch and then the harmonic. The model seems to fit moderately well in all cases.

3.2.2 Logarithm

Now we will consider the dervative at zero of the function

$$g(x) := \ln(x + 1).$$



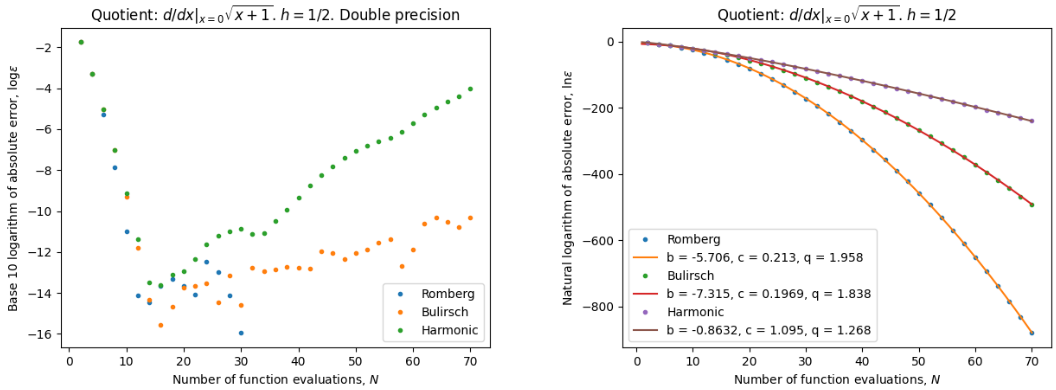
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	0.01503	2.443	0.2078	0.007242	1.962	0.000123
Bulirsch	lin-ln evals-error	0.005981	6.275	0.1873	0.08379	1.851	0.001315
Harmonic	lin-ln evals-error	96.18	2.331	1.128	0.01092	1.264	0.0004176

We get down to machine level precision using any sequence, Romberg performes best, then Bulirsch and then the Harmonic one. The model fits quite well in all cases.

3.2.3 Square root

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

$$h(x) := \sqrt{1+x}$$



Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	0.00163	3.185	0.2109	0.009876	1.959	0.000166
Bulirsch	lin-ln evals-error	0.0007538	7.186	0.1925	0.09801	1.846	0.001513
Harmonic	lin-ln evals-error	7.958	2.102	1.159	0.008158	1.259	0.0003023

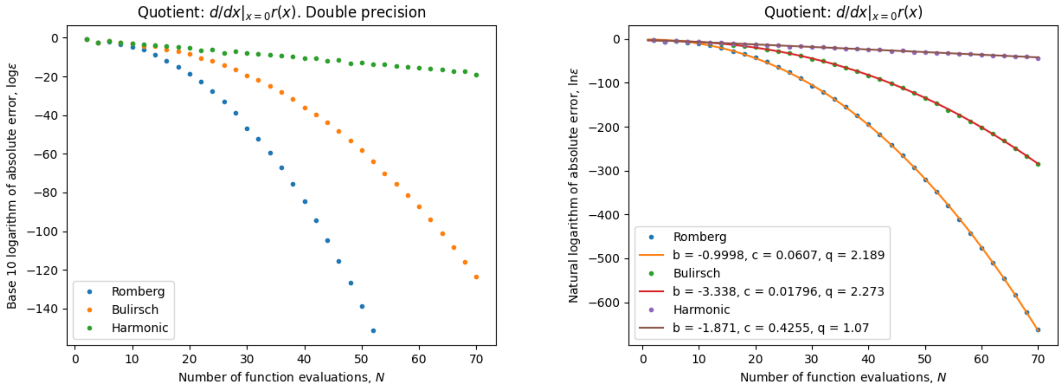
In standard double precision floating point arithmetic we get down to machine level precision using any sequence. The model fits quite well for the Romberg and the Harmonic sequence and moderately well for the Bulirsch sequence.

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.



Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$6.865 \cdot 10^{52}$	25	0.177	2.644	2.181	0.02709
Bulirsch	lin-ln evals-error	$4.186 \cdot 10^{60}$	25	0.2329	5.914	2.282	0.07529
Harmonic	lin-ln evals-error	.	.	.	.	.	.

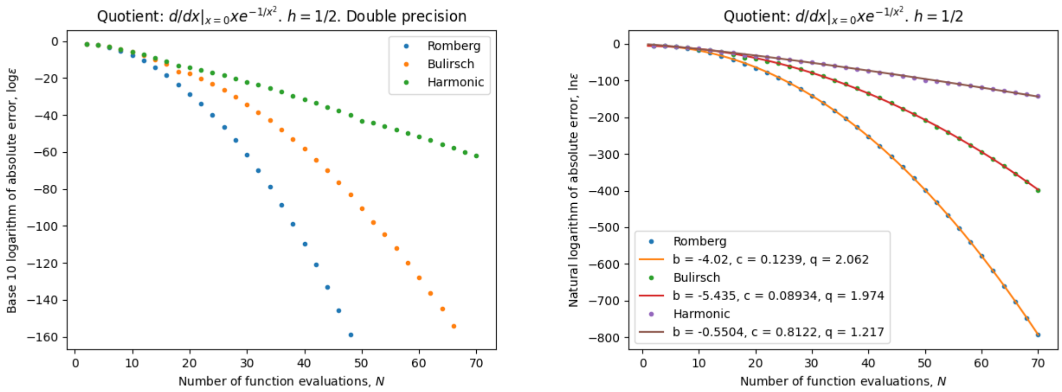
Here the model does not fit in any case.

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.



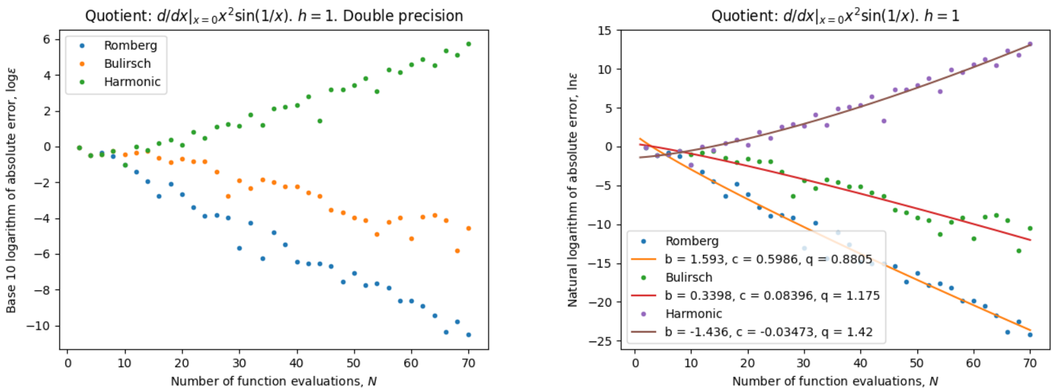
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	1228	24.39	0.1549	0.6269	2.043	0.004482
Bulirsch	lin-ln evals-error	$1.706 \cdot 10^{65}$	25	1.496	7.898	1.948	0.08245
Harmonic	lin-ln evals-error	.	.	.	.	.	.

Here the model does not fit in any case.

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:

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



Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	.	.	2607	2.282	4.553	3.923
Bulirsch	lin-ln evals-error	.	.	.	.	.	.
Harmonic	lin-ln evals-error	.	.	.	.	.	.

Here the model simply does not fit. 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
$d/dx\|_{x=0}\ln(x+1).\ h=1/2$	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
$d/dx\|_{x=0}e^x$	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.

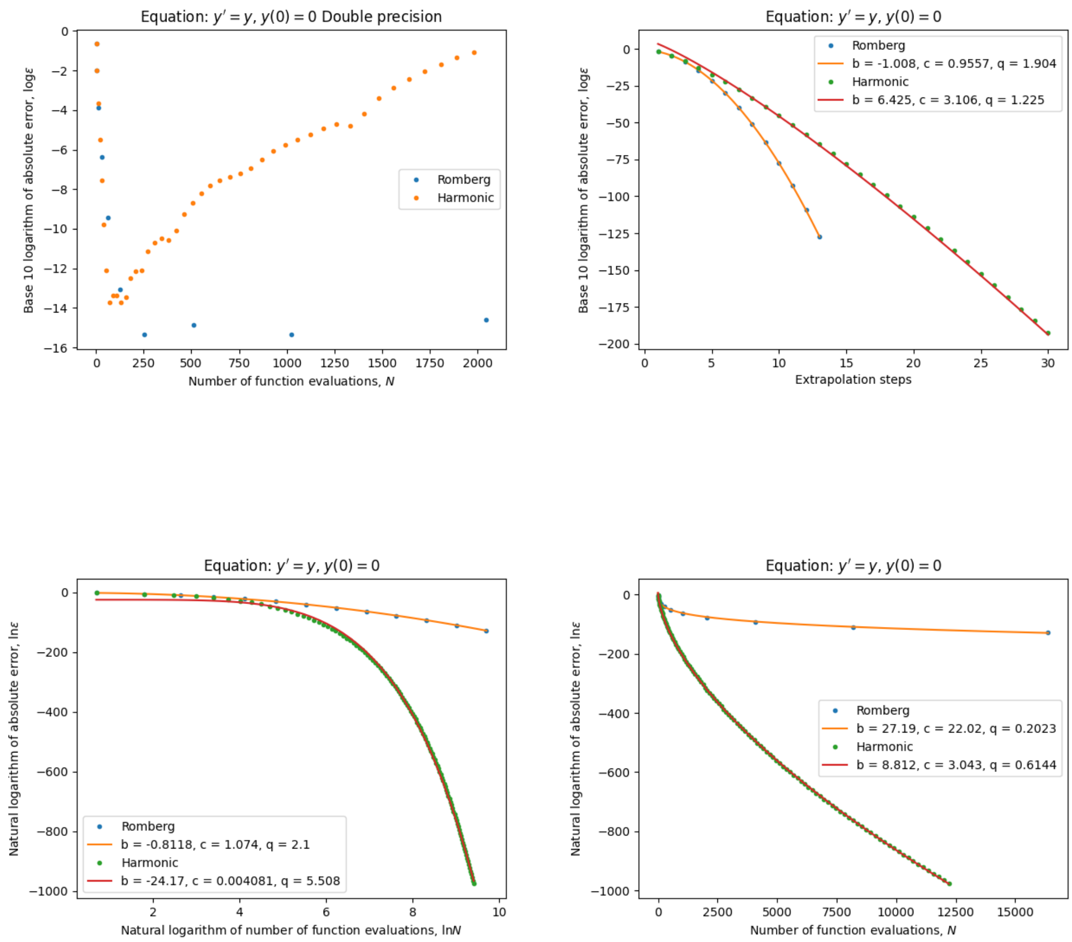
Now we will consider the results of the experiments.

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 $y(x) = e^x$.



Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$3.929 \cdot 10^{28}$	3	34.22	0.08528	0.1755	0.02121
Harmonic	lin-ln evals-error	$1.053 \cdot 10^{12}$	12.16	3.226	0.02133	0.6119	0.001013
Romberg	lin-ln steps-error	0.162	0.07703	0.8922	0.0008956	1.928	$3.583 \cdot 10^{-5}$
Harmonic	lin-ln steps-error	$2.469 \cdot 10^{10}$	11.7	3.279	0.01751	1.22	0.0007849
Romberg	ln-ln evals-error	1.232	0.1808	1.147	0.001975	2.075	$8.865 \cdot 10^{-5}$
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here we clearly have exponential convergence in the number of steps for the Romberg sequence but none of the models fits for the harmonic sequence.

4.2.2 Logistic curve

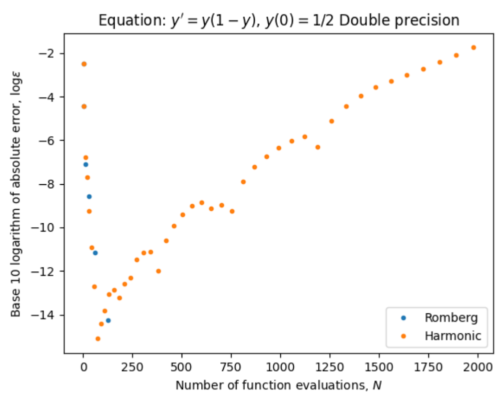
Then we will consider the following initial value problem

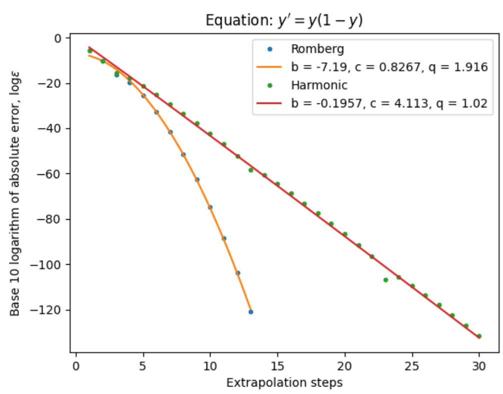
$$y'(x) = y(x)(1 - y(x)), \quad y(0) = 1/2, \quad x \in [0, 1] \tag{4.3}$$

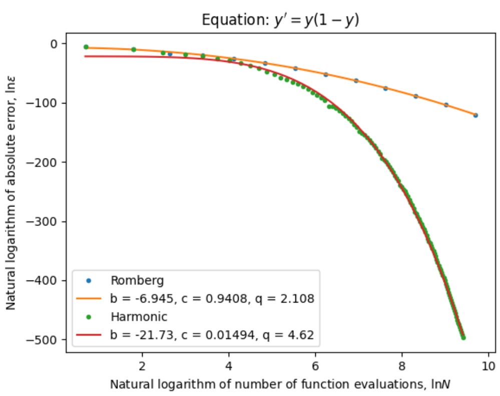
whose solution is the sigmoid function

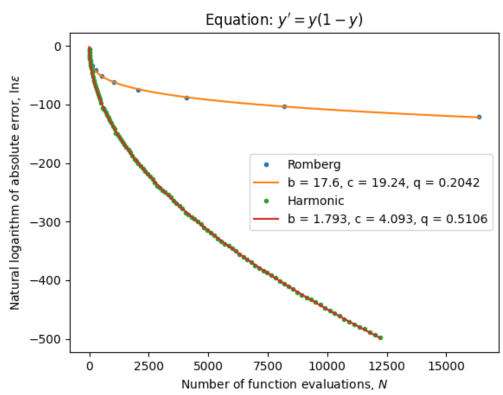
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

which is analytic.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$7.183 \cdot 10^{18}$	2.999	26.25	0.08333	0.1874	0.02078
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	$9.544 \cdot 10^{-5}$	0.1065	0.6621	0.001969	1.995	$7.267 \cdot 10^{-5}$
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	0.0004803	0.1008	0.852	0.002435	2.15	0.0001118
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here we clearly have exponential convergence in the number of steps for the Romberg sequence. The model does not fit in any case for the harmonic sequence.

4.2.3 Tangens

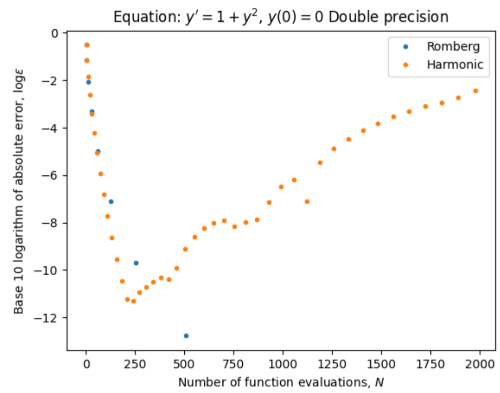
Now we will consider the following equation

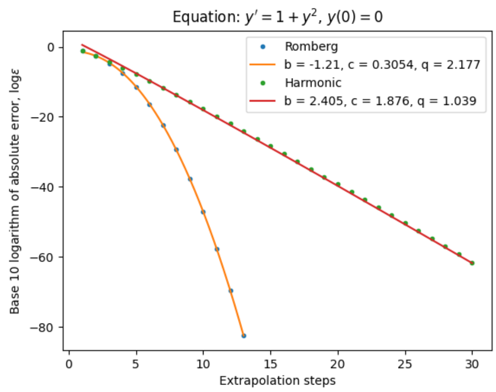
$$y'(x) = 1 + y(x)^2, \quad y(0) = 0, \quad x \in [0, 1] \tag{4.4}$$

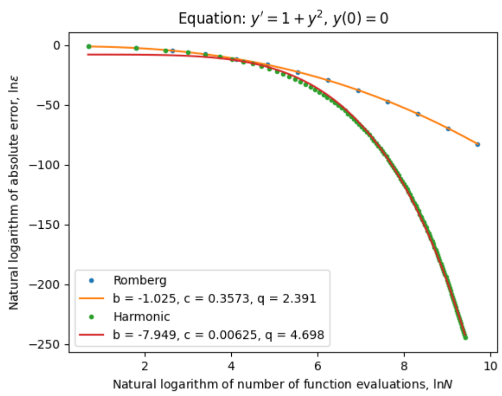
whose solution is

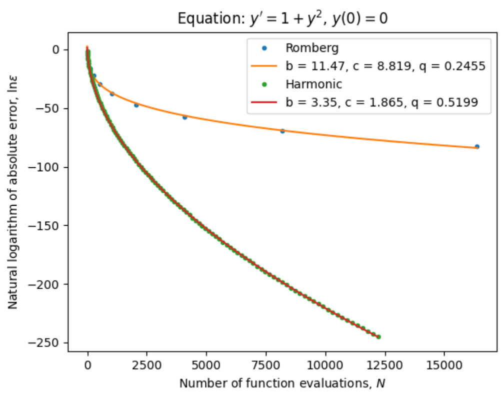
$$y(x) := \tan(x)$$

which is meromorphic and we are quite far from singularites.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$1.351 \cdot 10^{13}$	2.997	13.1	0.1255	0.224	0.02402
Harmonic	lin-ln evals-error	616.1	0.9751	1.974	0.01426	0.517	0.001138
Romberg	lin-ln steps-error	0.2218	0.00522	0.2875	0.0001785	2.2	$5.487 \cdot 10^{-6}$
Harmonic	lin-ln steps-error	204.6	0.9415	1.982	0.01258	1.033	0.0009701
Romberg	ln-ln evals-error	0.6649	0.2211	0.373	0.008112	2.379	0.0002882
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here we clearly have exponential convergence in the number of steps for the Romberg sequence. None of the models fits well for the Harmonic sequence.

4.2.4 Equation with singularity

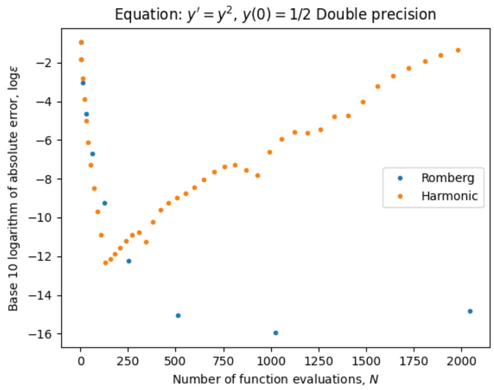
Now we will consider the following initial value problem:

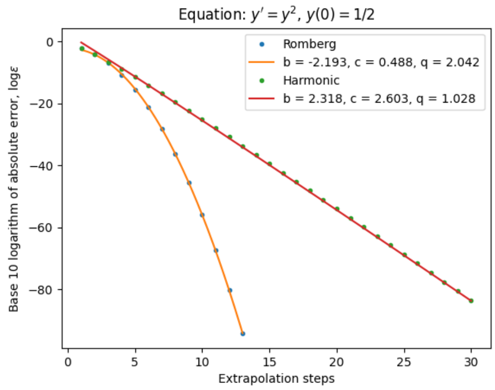
$$y'(t) = y^2(t), \quad y(0) = 1/(1+a), \quad t \in [0,1] \tag{4.5}$$

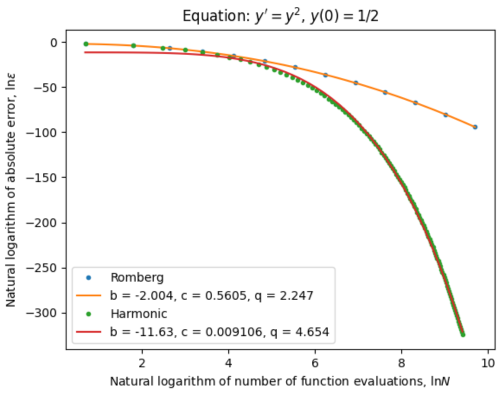
whose solution is

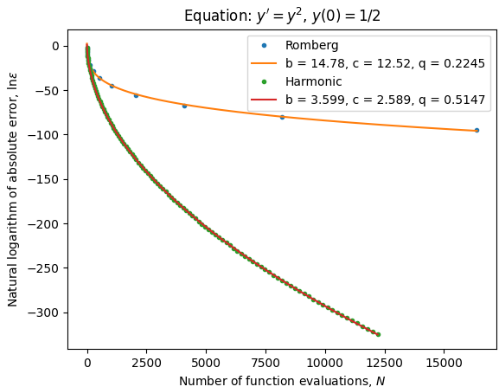
$$y(t) = \frac{1}{1-(t-a)}.$$

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



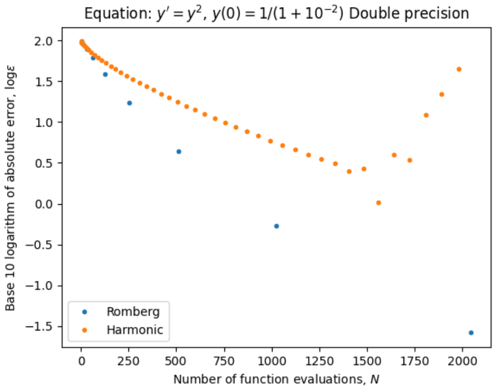


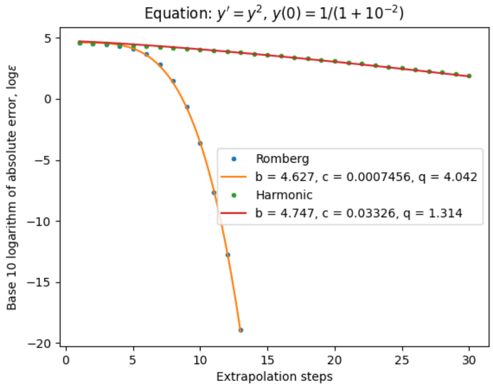


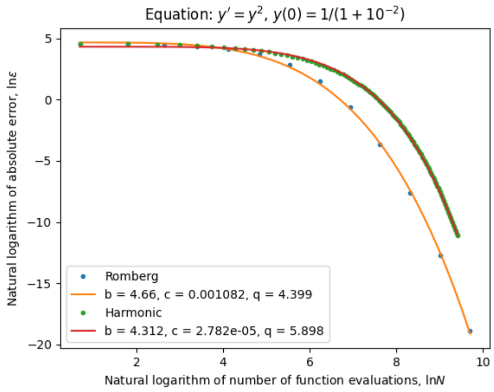


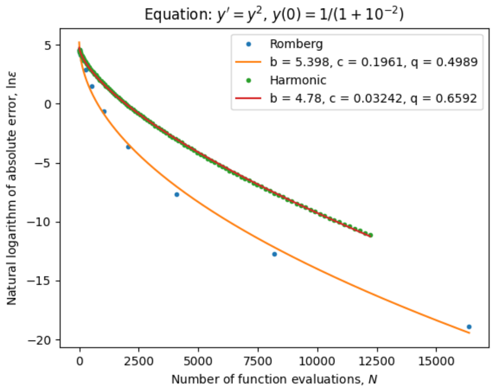
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$1.725 \cdot 10^{16}$	2.999	18.53	0.09702	0.2017	0.02064
Harmonic	lin-ln evals-error	728.6	0.9185	2.701	0.007961	0.5123	0.0006199
Romberg	lin-ln steps-error	0.05483	0.02945	0.4449	0.0008532	2.075	$3.038 \cdot 10^{-5}$
Harmonic	lin-ln steps-error	169.4	0.8824	2.711	0.006852	1.024	0.0005169
Romberg	ln-ln evals-error	0.1965	0.09582	0.5737	0.002273	2.24	$8.882 \cdot 10^{-5}$
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here we seem to have exponential convergence in the number of steps for the Romberg sequence. For the Harmonic sequence, none of the models fits nicely.



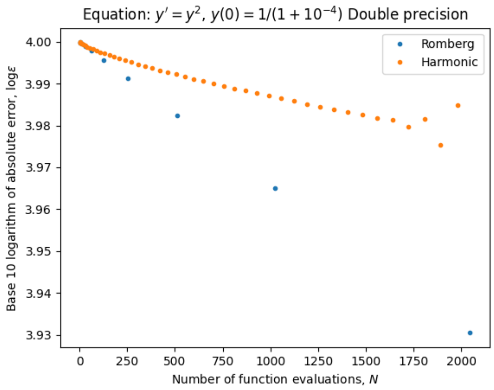


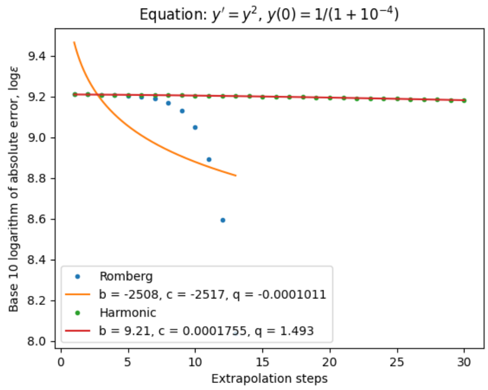


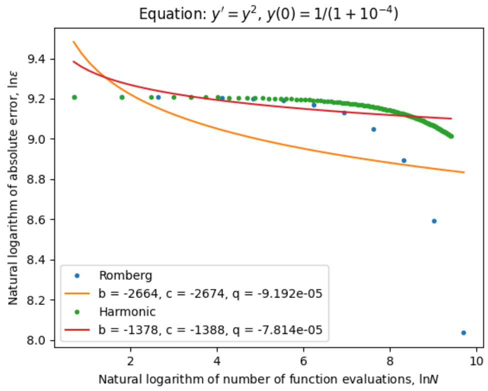


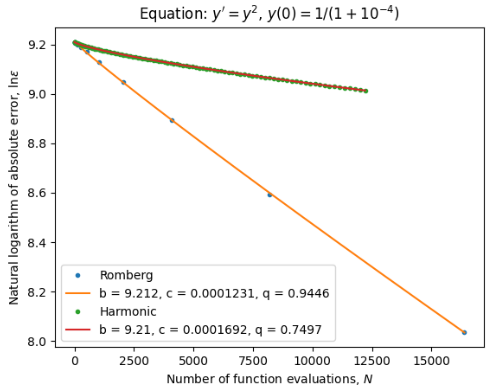
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	567.8	1.143	0.1661	0.7022	0.5832	0.03661
Harmonic	lin-ln evals-error	241.1	0.4337	0.04452	0.2373	0.6456	0.007567
Romberg	lin-ln steps-error	108	0.05406	0.0004885	0.3319	4.297	0.002772
Harmonic	lin-ln steps-error	226.2	0.4359	−25.94	51.96	1.257	0.02703
Romberg	ln-ln evals-error	117.2	0.08201	0.0007221	0.4366	4.711	0.004126
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here the model fits moderately well for Romberg sequence, when considering exponential convergence in the number of steps. We do not have nice fit for the harmonic sequence in any of the cases.





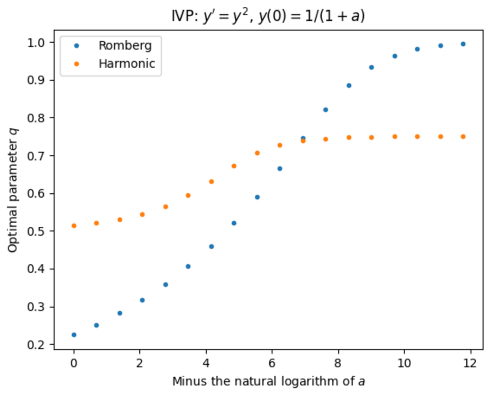




Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$1.001 \cdot 10^4$	$9.072 \cdot 10^{-7}$	$9.09 \cdot 10^{-5}$	0.01109	0.9818	0.0001641
Harmonic	lin-ln evals-error	$1 \cdot 10^4$	$2.528 \cdot 10^{-8}$	0.0001699	0.0002823	0.7496	$7.364 \cdot 10^{-6}$
Romberg	lin-ln steps-error	7433	0.3334	-67.64	3	5.423	0.3392
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	4948	1	-198.5	1.232	4.103	1.002
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here the model does not fit in any case.

The plot of q against a is as follows:



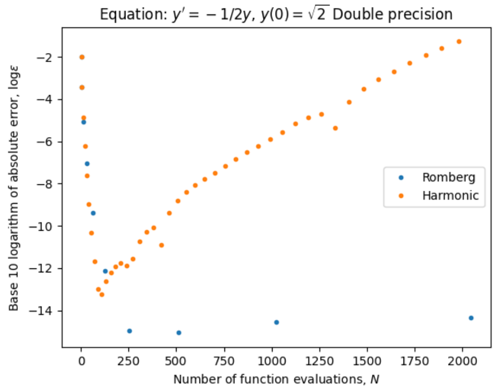
4.2.5 Equation with moderate singularity

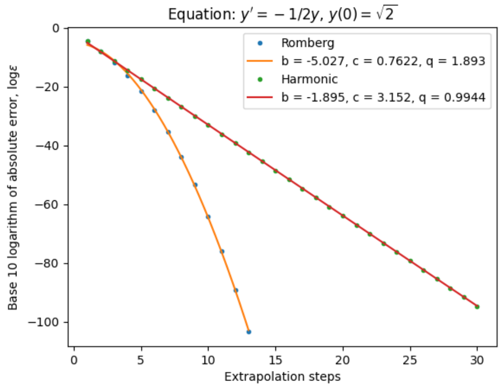
Now we will consider the following initial value problem

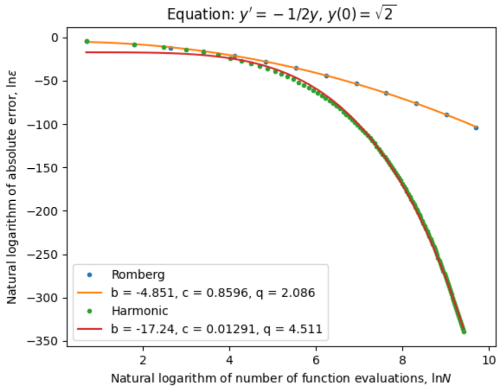
$$y'(t) = -\frac{1}{2y}, \quad y(0) = \sqrt{1+a}, \quad t \in [0,1] \tag{4.6}$$

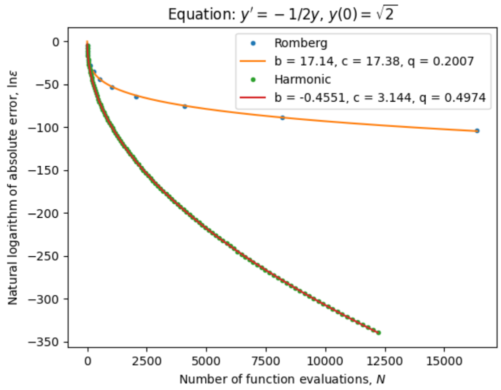
whose solution is

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



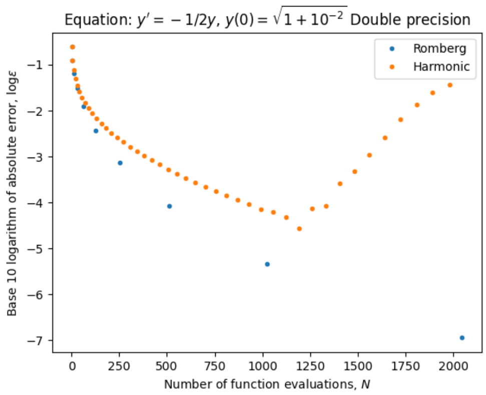


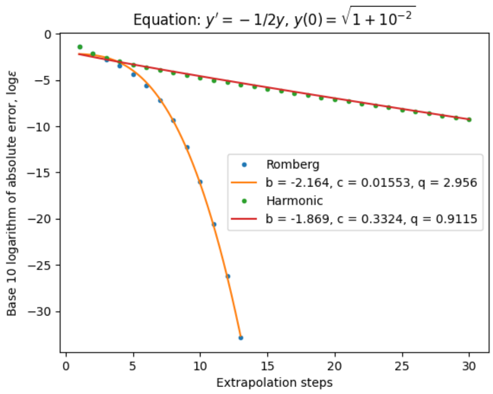


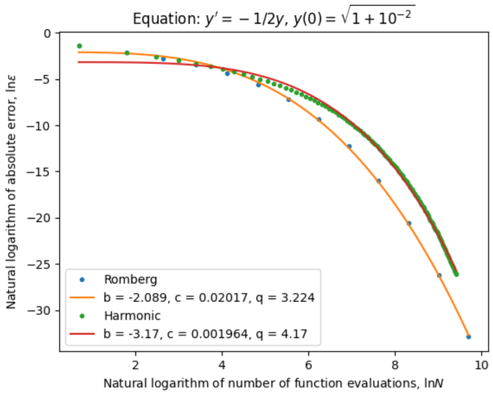


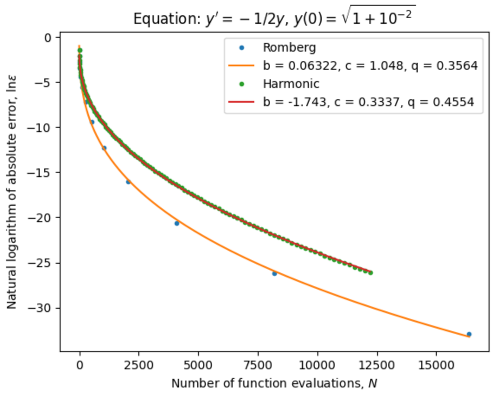
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$3.406 \cdot 10^{16}$	2.999	23.22	0.06365	0.1841	0.01443
Harmonic	lin-ln evals-error	0.4658	0.1195	3.123	0.0002015	0.4979	$1.14 \cdot 10^{-5}$
Romberg	lin-ln steps-error	0.0008592	0.3997	0.5989	0.009262	1.979	0.0003485
Harmonic	lin-ln steps-error	0.1044	0.144	3.127	0.0002611	0.9956	$1.552 \cdot 10^{-5}$
Romberg	ln-ln evals-error	0.002924	0.01955	0.7671	0.0003673	2.132	$1.55 \cdot 10^{-5}$
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here we clearly have exponential convergence in the number of steps and evaluations for the harmonic sequence. We also have it in the number of steps for the Romberg sequence.





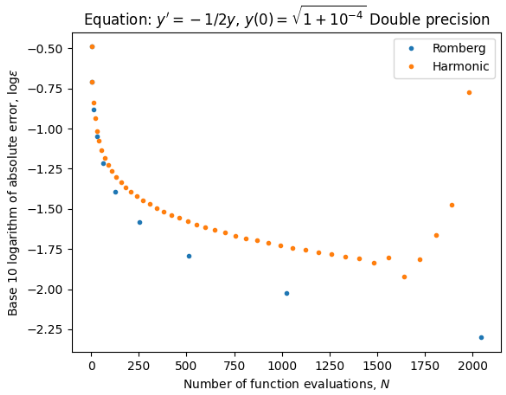


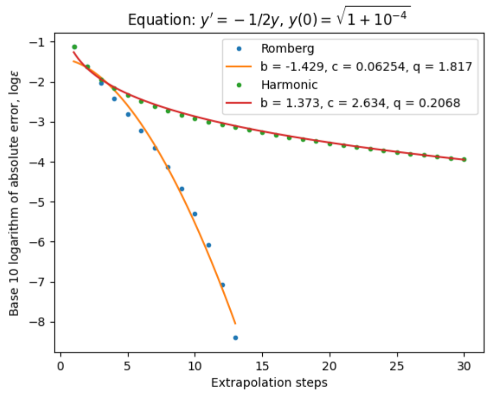


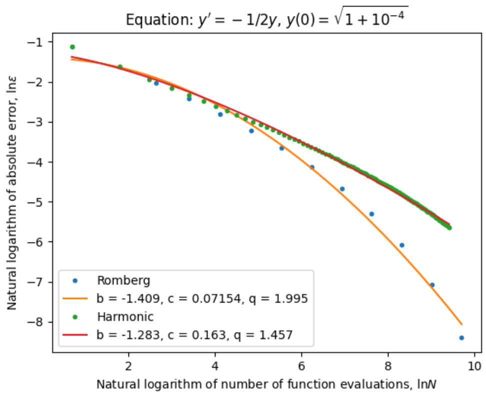
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	1.977	1.277	0.912	0.1265	0.3841	0.009173
Harmonic	lin-ln evals-error	0.1185	0.9408	0.3049	0.2691	0.4676	0.007262
Romberg	lin-ln steps-error	0.05369	0.04044	0.009886	0.08471	3.134	0.001253
Harmonic	lin-ln steps-error	0.1007	0.6539	0.2999	0.2215	0.9368	0.006574
Romberg	ln-ln evals-error	0.06264	0.01892	0.01292	0.03735	3.418	0.0006328
Harmonic	ln-ln evals-error	.	.	.	.	.	.

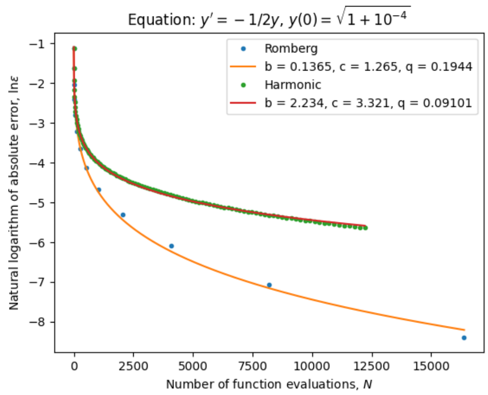
For the Romberg sequence, the model seems to fit moderately well when considering exponential convergence in the number of evaluations but it is not entirely clear since the model also fits moderately well when considering exponential converge in the number of steps and those two are mutually exclusive.

For the harmonic sequence we do not get nice fit in any case.





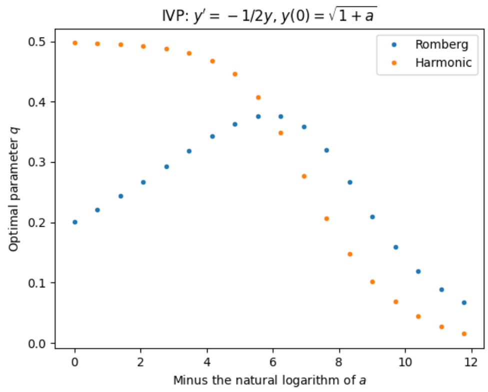




Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	1.59	1.047	1.341	0.3366	0.1989	0.08076
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	0.1375	0.1919	0.04595	0.8511	2.117	0.05595
Harmonic	lin-ln steps-error	10.05	18.03	1.442	1.062	0.3442	0.1232
Romberg	ln-ln evals-error	0.148	0.1921	0.05535	0.7589	2.281	0.05284
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here, we do not have any clear fit.

The plot of q against a is as follows:



4.2.6 Circular rotation

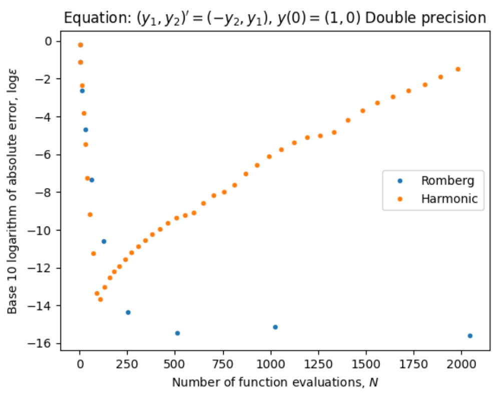
Now we will consider the following system of equations:

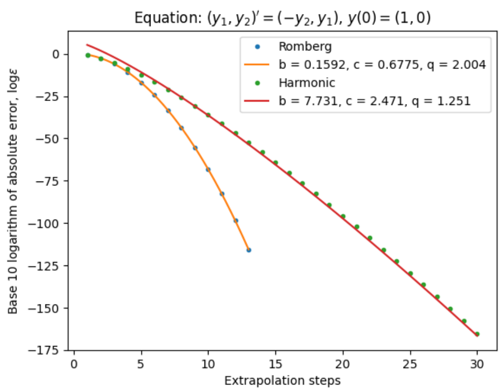
$$(y_1(t), y_2(t))' = (-y_2(t), y_1(t)), \quad y(0) = (1, 0), \quad t \in [0, \pi/2] \tag{4.7}$$

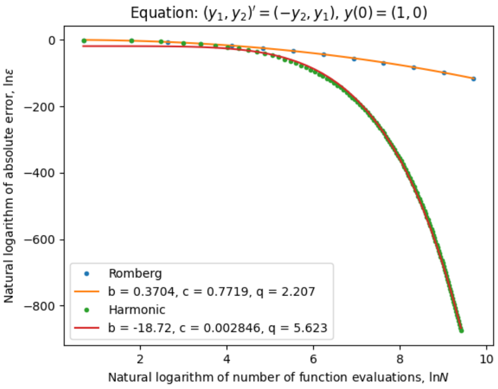
whose solution is

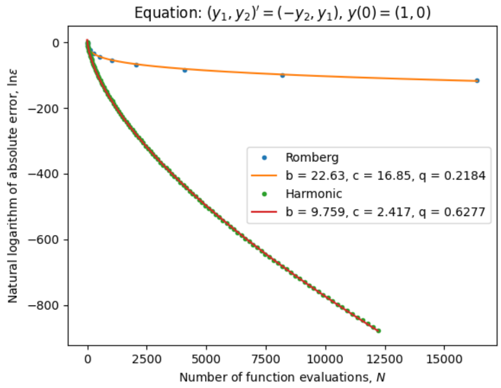
$$(y_1(t), y_2(t)) = (\cos t, \sin t)$$

which is entire.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$9.158 \cdot 10^{24}$	3	27.06	0.103	0.1896	0.02387
Harmonic	lin-ln evals-error	$1.269 \cdot 10^{13}$	12.98	2.597	0.0327	0.6252	0.001654
Romberg	lin-ln steps-error	1.09	0.0003639	0.6727	$6.223 \cdot 10^{-6}$	2.007	$2.397 \cdot 10^{-7}$
Harmonic	lin-ln steps-error	$4.499 \cdot 10^{11}$	12.52	2.643	0.02762	1.246	0.001318
Romberg	ln-ln evals-error	7.126	0.3484	0.8677	0.005226	2.163	0.0002203
Harmonic	ln-ln evals-error	.	.	.	.	.	.

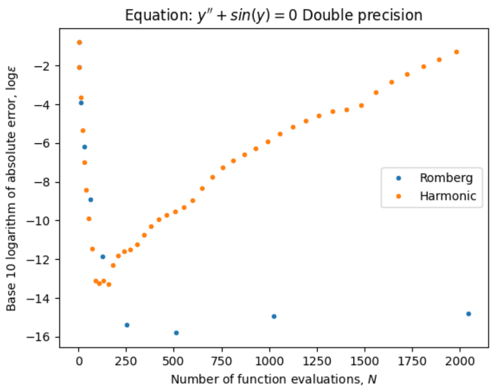
The harmonic sequence works better then Romberg and we get down to machine level precision using either sequence when using standard floating point arithmetic.

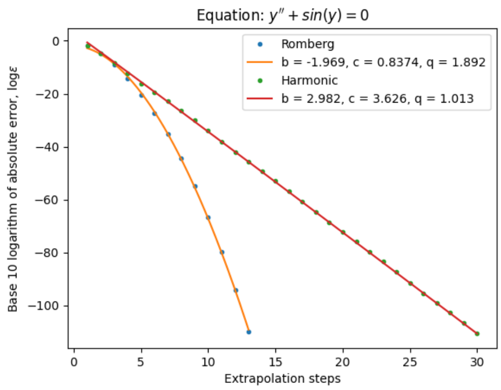
We clearly have exponential convergence in the number of steps for the Romberg sequence but we do not have any nice fit for the harmonic sequence.

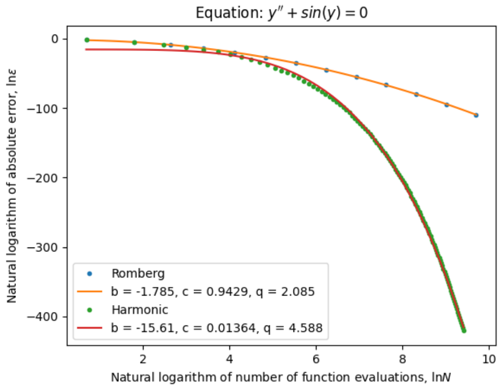
4.2.7 Mathematical pendulum

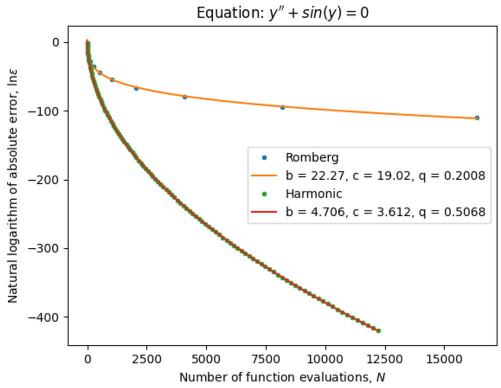
Now we will consider the mathematical pendulum equation:

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









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$4.852 \cdot 10^{18}$	3	23.49	0.06226	0.1898	0.01102
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	0.01054	1.671	0.603	0.03526	2.013	0.001162
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	0.0311	0.7449	0.7694	0.01405	2.17	0.0005522
Harmonic	ln-ln evals-error	.	.	.	.	.	.

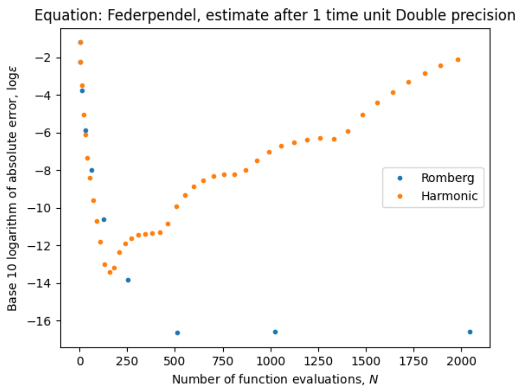
Here we seem to have exponential convergence in the number of steps for the Romberg sequence. The harmonic sequence works is faster and we almost get down to machine level precision using either sequence.

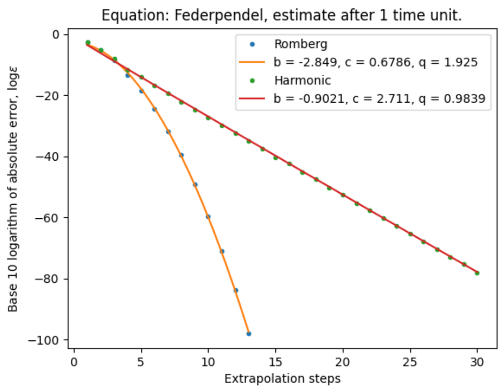
4.2.8 Federpendel

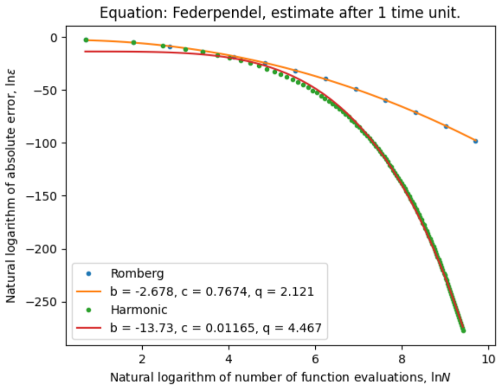
Now we will consider the equation of motion for das Federpendel or the spring pendulum:

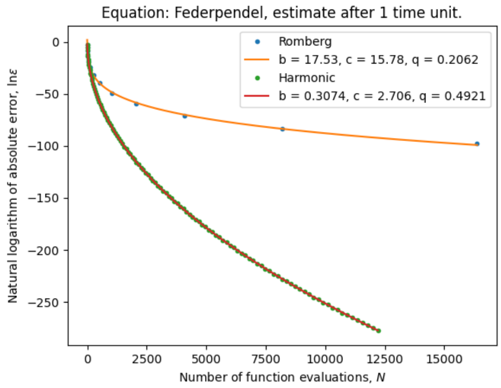
$$\mathbf{p}' = -(|\mathbf{q}| - 1)\frac{\mathbf{q}}{|\mathbf{q}|} - \begin{pmatrix}1\\0\end{pmatrix}, \quad \mathbf{q}' = \mathbf{p}$$

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 $t = 2$.





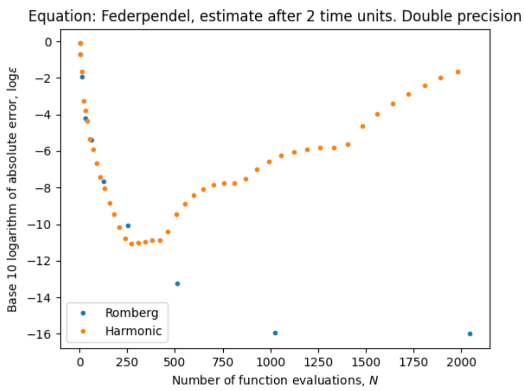


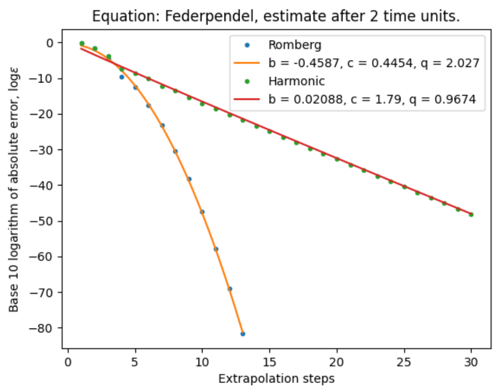


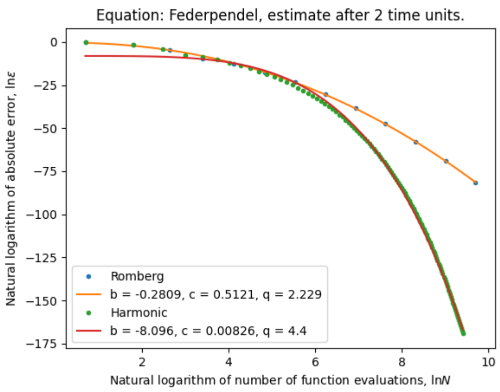
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	2.325^{15}	2.897	20.8	0.07387	0.1913	0.01729
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	0.006769	0.2266	0.5219	0.007752	2.019	0.0002881
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	0.02348	0.2334	0.6702	0.003067	2.176	0.000111
Harmonic	ln-ln evals-error	.	.	.	.	.	.

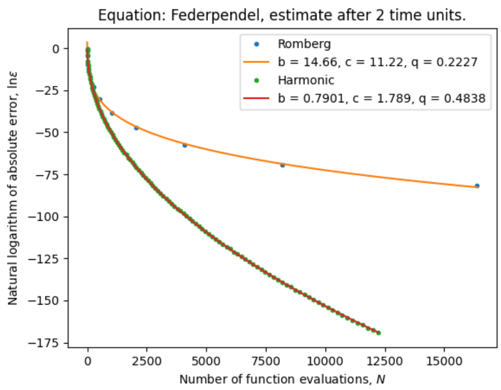
Here we seem to have exponential convergence in the number of steps for the Romberg sequence. None of the models fits well for the harmonic sequence.

The harmonic sequence performes better we almost get down to machine level precision using either sequence, in double precision floating point arithmetic.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$2.562 \cdot 10^{13}$	3	13.43	0.1715	0.224	0.04363
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	0.07725	0.4469	0.2919	0.04082	2.194	0.001626
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	0.2609	0.6696	0.3816	0.05891	2.373	0.002839
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here the model for exponential convergence in the number of steps seems to fit moderately well for the Romberg sequence but none of the models fits for the harmonic sequence.

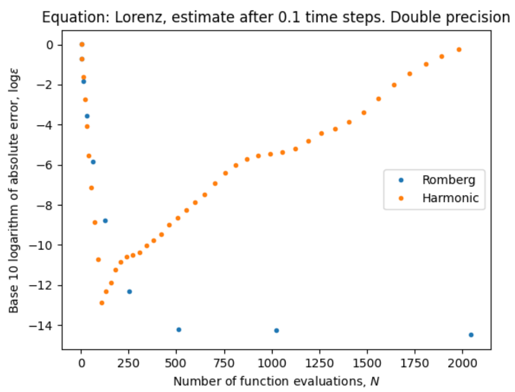
The harmonic sequence works better and we get almost down to machine level precision in standard double precision floating point arithmetic.

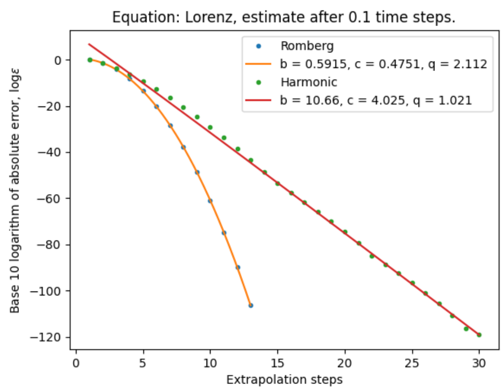
4.2.9 Lorenz equations

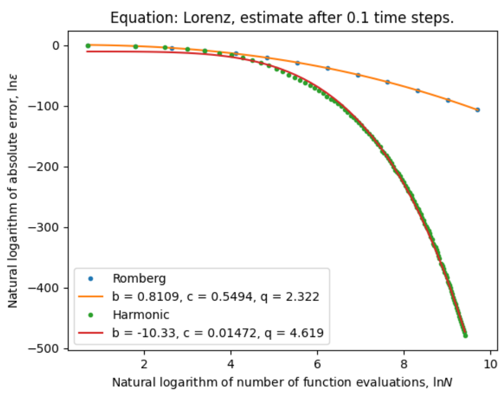
The Lorenz equations are the following system:

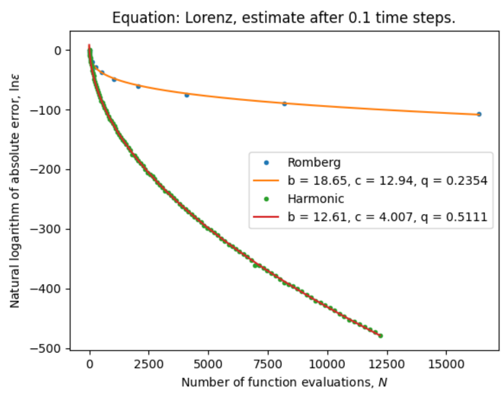
$$\frac{dx}{dt} = \sigma(y - x), \quad \frac{dy}{dt} = x(\rho - z) - y, \quad \frac{dz}{dt} = xy - \beta z$$

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 $(x(0), y(0), z(0)) = (1, 1, 1)$.





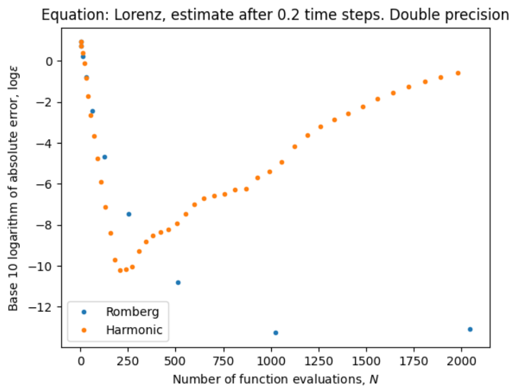


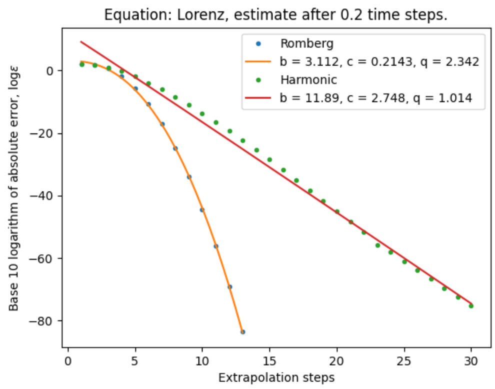


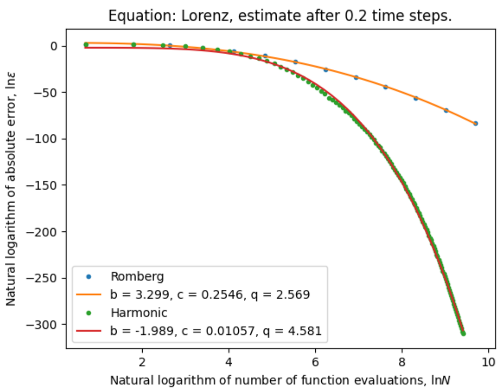
Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$3.447 \cdot 10^{21}$	3	21.2	0.1242	0.2061	0.02692
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	2.886	0.04035	0.4956	0.0009672	2.098	$3.499 \cdot 10^{-5}$
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	15.97	0.4928	0.6421	0.01116	2.265	0.0004469
Harmonic	ln-ln evals-error	.	.	.	.	.	.

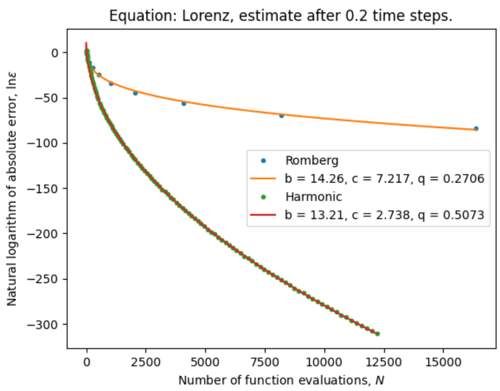
Here the harmonic sequence works better and we get down to machine level precision in standard double precision arithmetic.

The model for exponential convergence in the number of steps seems to fit well for the Romberg sequence but none of the models fits for the harmonic sequence.









Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$1.544 \cdot 10^{17}$	3	12.97	0.1833	0.2372	0.03499
Harmonic	lin-ln evals-error	.	.	.	.	.	.
Romberg	lin-ln steps-error	129.1	0.3322	0.2655	0.01477	2.27	0.0004755
Harmonic	lin-ln steps-error	.	.	.	.	.	.
Romberg	ln-ln evals-error	506.5	0.8695	0.3478	0.03817	2.457	0.00137
Harmonic	ln-ln evals-error	.	.	.	.	.	.

Here we also get down to machine level precision in standard double precision floating point arithmetic. The harmonic sequence performes better.

None of the models fits for the harmonic sequence but we seem to have exponential convergence in the number of steps for the romberg sequence.

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.