Spectral Convergence of Extrapolation

Hjalti Þór Ísleifsson

Advisor
Prof. Dr. Ralf Hiptmair

Zürich, October 2020

Contents

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. We will follow section 9.4.2 in [2].

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 a discrete set $\mathbb{H}$ 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 $h \to 0$, $h \in \mathbb{H}$.

Let (x_1, y_1), ..., (x_k, y_k) be a collection of points such that $x_1, \ldots, x_k$ are distinct. Then there exists a uniqe polynomial P of degree $k-1$ 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. Let $p > 0$ be an integer and points (x_1^p, y_1), ..., (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 these 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 $P_{ij}(h^p) \coloneqq P(h^p; h_{i-j+1}^p, h_i^p)$ be the interpolation polynomial in h^p which interpolates $(h_{i-j+1}^p, T(h_{i-j+1})), \ldots, (h_i^p, T(h_i^p))$ and set $T_{ij} \coloneqq P_{ij}(0)$. Then according to the Neville scheme we can compute T_{ij}, $j \leq i$, in the following recursive way:

1. $T_{i1} \coloneqq T(h_i)$ for $i = 1, \ldots, k$.
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 $1 < j \leq i$ 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 [2]) that T_{ij} has polynomial decay of higher degree, as $h \to 0$, than T. Let $\varepsilon_k \coloneqq |T_{kk} - T|$. We want to analyse 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 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

$$(b^*, c^*, q^*) \coloneqq \operatorname{argmin}_{(b,c,q)} \left\{ \sum_{k=1}^{n} |\ln \varepsilon_k - (b - c N_k^q)|^2 \right\} \tag{1.5}$$

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

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.

We define the following relative residuals:

$$\rho_{\ln} := \frac{\sum_{k=1}^n |\ln \varepsilon_k - (b^* - c^* N_k^{q^*})|^2}{\sum_{k=1}^n |\ln \varepsilon_k|^2}$$

and

$$\rho_{\text{lin}} := \frac{\sum_{k=1}^n |\varepsilon_k - A^* \exp(-c^* N_k^{q^*})|^2}{\sum_{k=1}^n |\varepsilon_k|^2}$$

where $A^* := \exp(b^*)$. We will use these residuals as one measurement of the goodness of fit.

In addition to that 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 $k = 0, 1, \ldots, n$. Let $r = \max 10, [n/2]$. Then we will do the curve fitting for $(x_k, y_k), \ldots, (x_{k+r-1}, x_{k+r-1})$ for $k = 3, \ldots, n - r - 1$ 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

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

In order to visualize the stability of the fit, we will plot all the curves obtained in the cross validation, on top of each other. We call this plots *stack plots*.

1.4 Code

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

```
#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 an np.array of np.arrays.
def extrapolate(sc, prob, seq, hp):
  n = len(seq)
  X = [[None] * (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 np.array([np.array(X_i) for X_i in X])
```

Chapter 2

Romberg quadrature

2.1 The algorithm

Let $f : [a, b] \to \mathbb{R}$ be a function and $I := \int_a^b f(x)dx$. The *trapezoidal rule* (see eg. section 9.1 in [2]) 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) for all $h \in \{(b - a)/n \mid n \in \mathbb{Z}_+ \ldots\} =: \mathbb{H}$.

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 [3]) 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 we let $F(x) := f(a + xh)$ 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 $h := (b - a)/n$. 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 to 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 of degree $j-1$ 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. $T_{i1} \coloneqq T_f(h_i)$ for $i = 1, \ldots, k$.
2. $T_{ij} \coloneqq T_{i,j-1} + \frac{T_{i,j-1} - T_{i-1,j-1}}{\left(\frac{h_{i-j+1}}{h_i}\right)^2 - 1}$ 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 analyse 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: $a_n = n$, $n > 0$.
- 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 $I \coloneqq \int_a^b f(x)dx$ using Romberg quadrature. We will use the stepsizes $h_k \coloneqq (b-a)/a_k$ for the extrapolation. Let T_{ij}, $i \geq 0$ and $j \leq i$ be the extrapolation table we get and $\varepsilon_k \coloneqq |T_{kk} - I|$ 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 $N_k = \sum_{i=1}^{k}(2^{i-1}+1) = 2^k + k - 1 \approx 2^k$ 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. Hence possibilities of having exponential convergence in the number of steps and in the number of evaluations are mutually exclusive for the Romberg sequence.

For the model fitting we will plot the logarithm of the error against the number of function evaluations and the number of extrapolation steps. Then 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 will also provide 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.

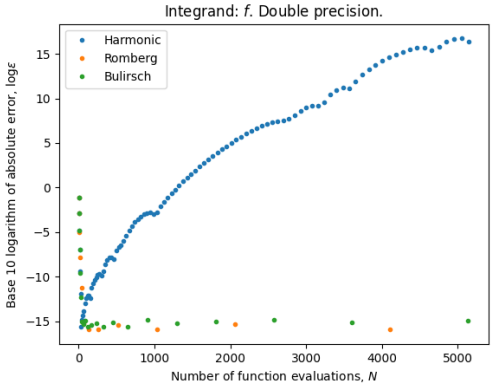
Now we will consider the results of the experiments.

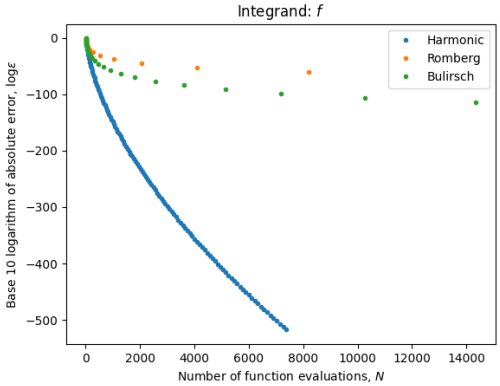
2.2.1 Cosine squared

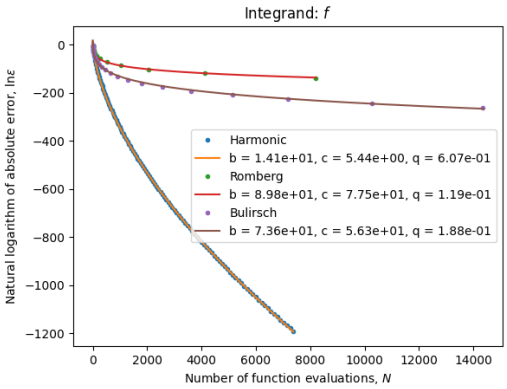
The first function we are going to try is

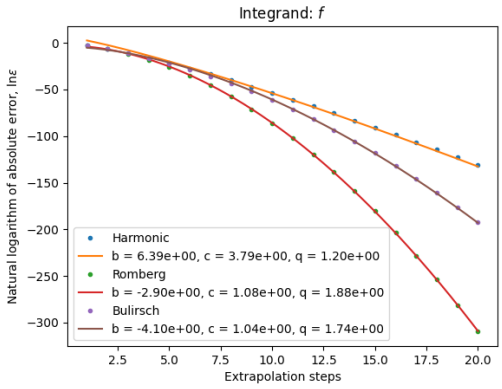
$$f : [0,1] \to \mathbb{R}, \quad f(x) \coloneqq \cos^2(x)$$

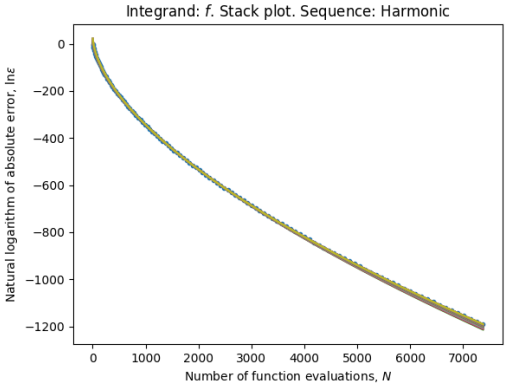
which is entire.

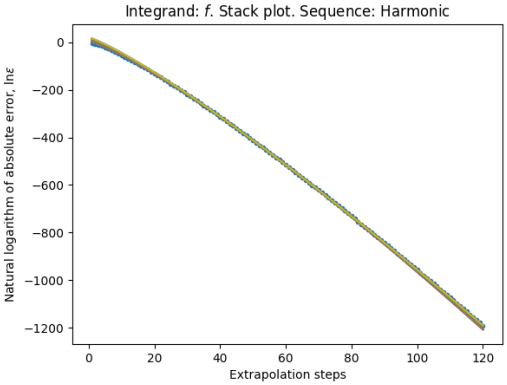


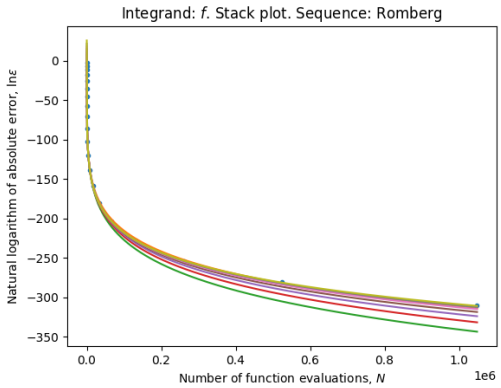


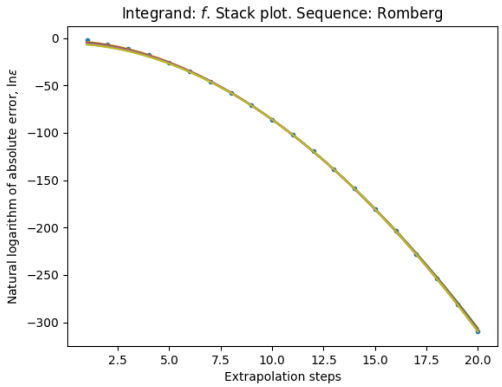

Integrand: f
Natural logarithm of absolute error, lnε
Number of function evaluations, N
Harmonic
b = 1.41e+01, c = 5.44e+00, q = 6.07e-01
Romberg
b = 8.98e+01, c = 7.75e+01, q = 1.19e-01
Bulirsch
b = 7.36e+01, c = 5.63e+01, q = 1.88e-01

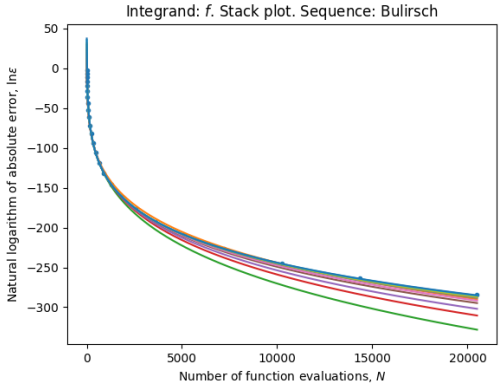

Integrand: f
Natural logarithm of absolute error, lnε
Extrapolation steps
Harmonic
b = 6.39e+00, c = 3.79e+00, q = 1.20e+00
Romberg
b = -2.90e+00, c = 1.08e+00, q = 1.88e+00
Bulirsch
b = -4.10e+00, c = 1.04e+00, q = 1.74e+00

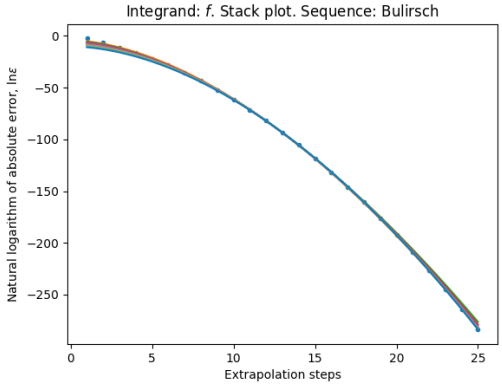

Integrand: f. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Integrand: f. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps


Integrand: f. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Integrand: f. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps


Integrand: f. Stack plot. Sequence: Bulirsch
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Integrand: f. Stack plot. Sequence: Bulirsch
Natural logarithm of absolute error, lnε
Extrapolation steps

Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	8.6[+12]	7.5[+00]	5.8[+00]	4.7[−03]	6.0[−01]	1.7[−04]	1.6[+07]	6.5[−06]
RS-evals	1.1[+73]	6.0[+00]	9.6[+01]	6.7[−02]	1.1[−01]	1.9[−02]	1.1[+07]	3.9[−04]
BS-evals	3.8[+76]	8.0[+00]	8.7[+01]	1.1[−01]	1.6[−01]	3.4[−02]	2.5[+10]	7.9[−04]
HS-steps	8.4[+07]	6.4[+00]	4.0[+00]	3.4[−03]	1.2[+00]	1.1[−04]	2.7[+04]	3.3[−06]
RS-steps	1.3[−02]	6.5[−01]	1.0[+00]	2.4[−03]	1.9[+00]	7.7[−05]	5.9[−01]	1.1[−05]
BS-steps	2.3[−03]	2.2[+00]	9.2[−01]	1.4[−02]	1.8[+00]	4.2[−04]	8.6[−01]	3.9[−05]

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 and Bulirsch sequence we clearly have exponential convergence in the number of steps but not in the number of evaluations.

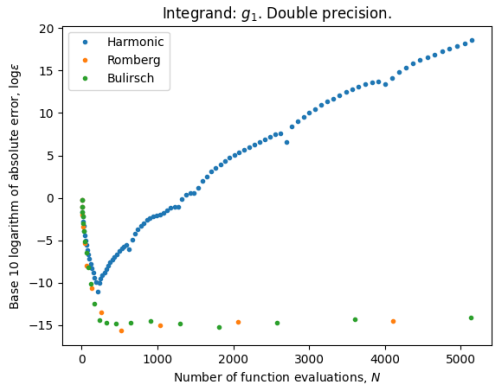
For the harmonic sequence it is hard to tell. We get very large values of A and we get large value for ρ_{lin}. On the contrary, the stack plots seem stable.

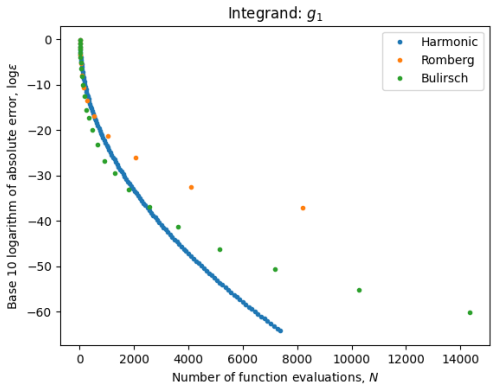
2.2.2 Function with poles

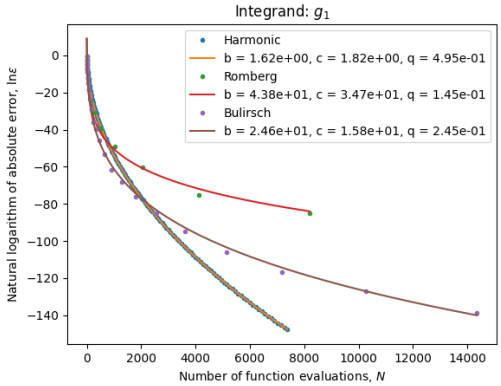
Now we will consider the following function:

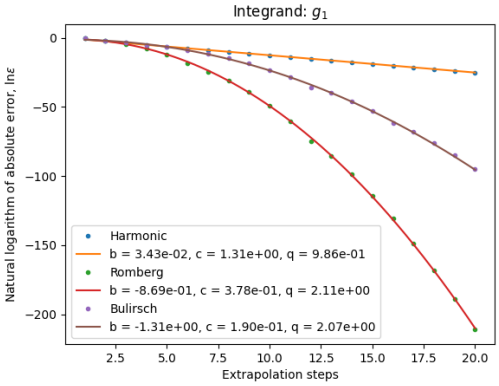
$$g_a : [-1, 1] \to \mathbb{R}, \quad g_a(x) := \frac{1}{a^2 + x^2}, \; a > 0$$

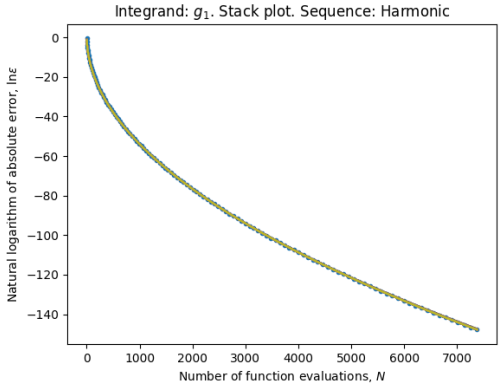
$a = 1$

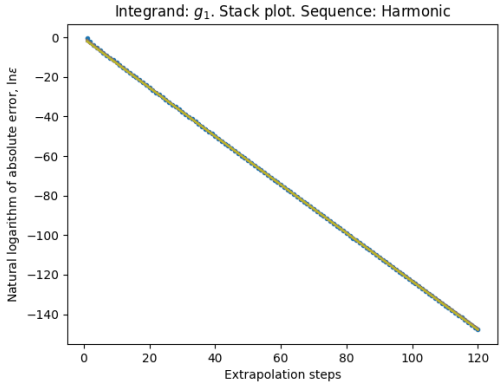


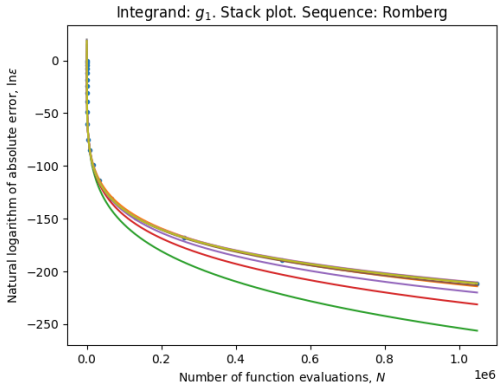


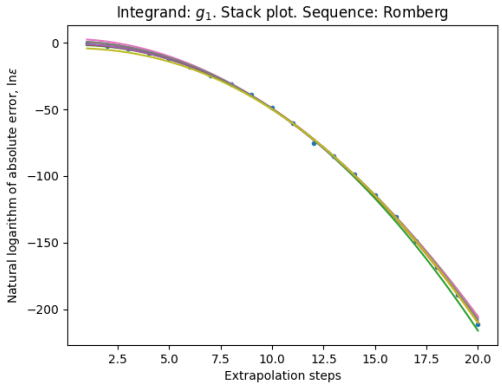


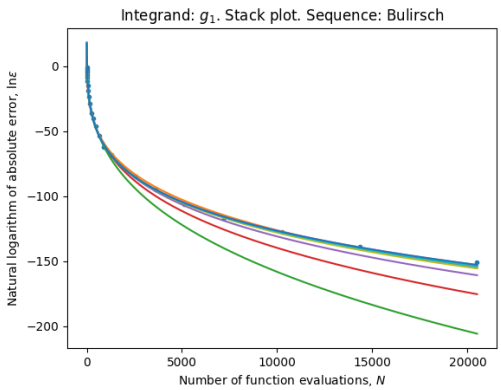


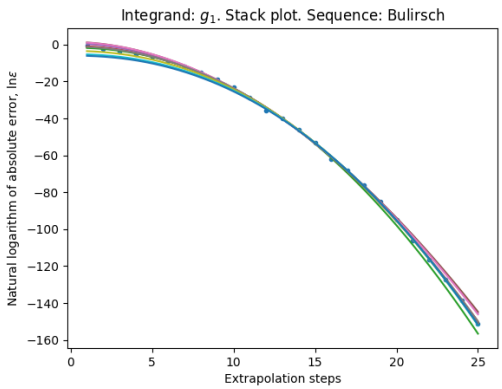










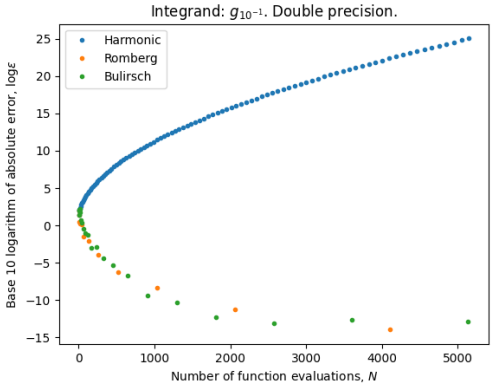


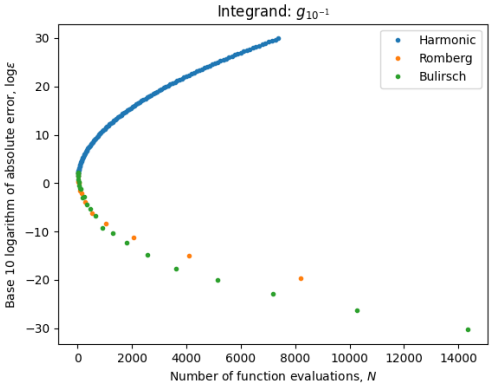
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	3.8[+00]	2.9[−02]	1.8[+00]	1.4[−04]	5.0[−01]	6.5[−06]	1.0[−01]	3.2[−07]
RS-evals	4.1[+37]	5.2[+00]	4.7[+01]	1.0[−01]	1.3[−01]	3.5[−02]	1.5[+05]	5.7[−04]
BS-evals	8.2[+22]	5.0[+00]	2.4[+01]	1.1[−01]	2.2[−01]	4.5[−02]	3.9[+05]	1.4[−03]
HS-steps	6.6[−01]	4.5[−02]	1.3[+00]	2.6[−04]	9.9[−01]	1.1[−05]	2.6[−01]	1.1[−06]
RS-steps	3.7[+00]	2.9[+00]	4.2[−01]	3.0[−02]	2.1[+00]	8.2[−04]	2.4[−01]	7.7[−05]
BS-steps	1.2[+00]	1.7[+00]	2.0[−01]	1.1[−01]	2.1[+00]	2.5[−03]	3.6[−01]	9.1[−05]

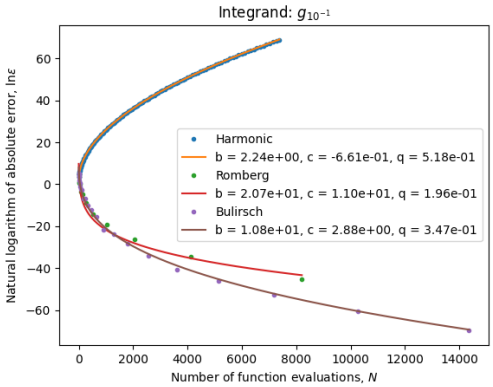
We see that the harmonic sequence performes best, then Bulirsch and then Romberg. In standard double precision arithmetic, we get down to machine level precision using Romberg or Bulirsch, but we are like 5 digits from there, using the harmonic sequence.

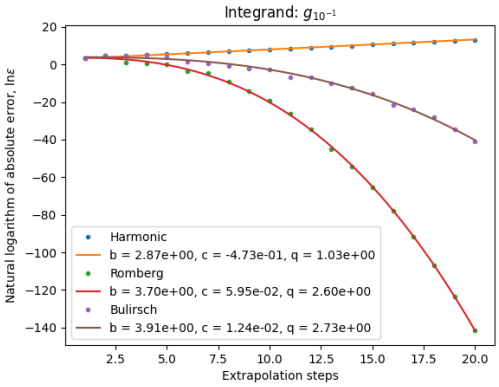
Here we clearly have exponential convergence in the number of evaluations for the Harmonic sequence and hence also in the number of steps. We do not have exponential convergence in the number of evaluations for Romberg and Bulirsch but the model for exponential convergence in the number of steps fits moderately well for those sequences.

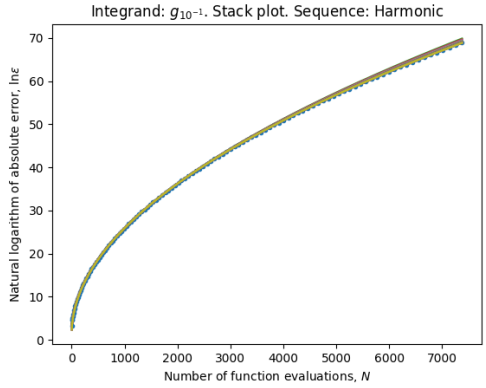
$a = 10^{-1}$

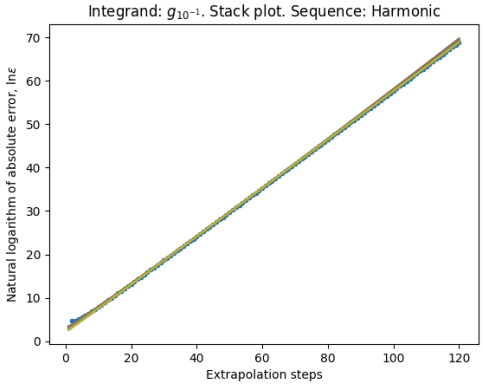


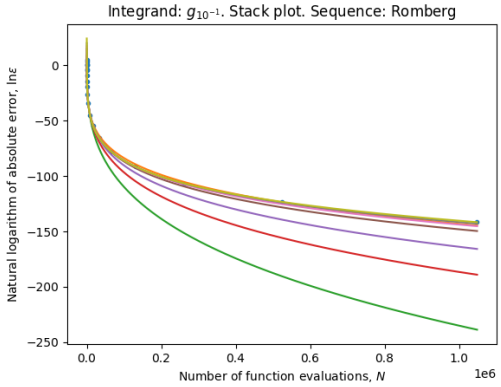


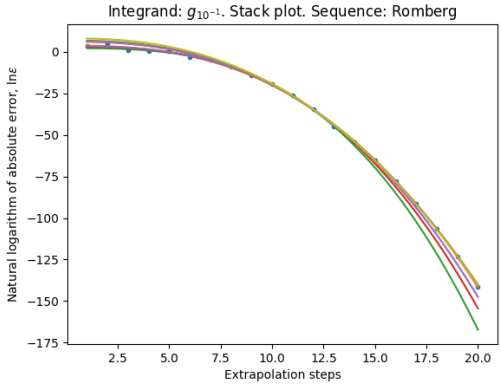


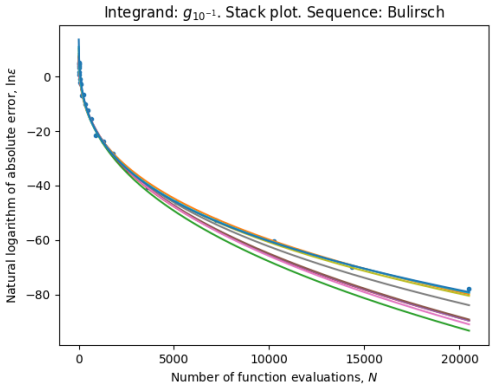


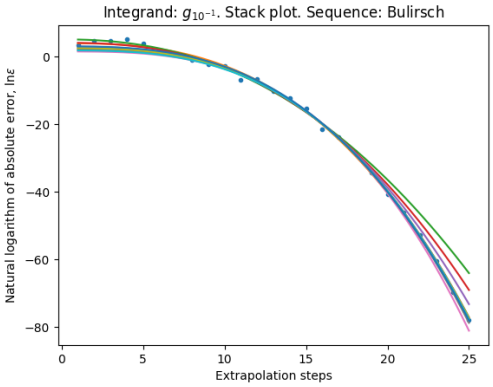








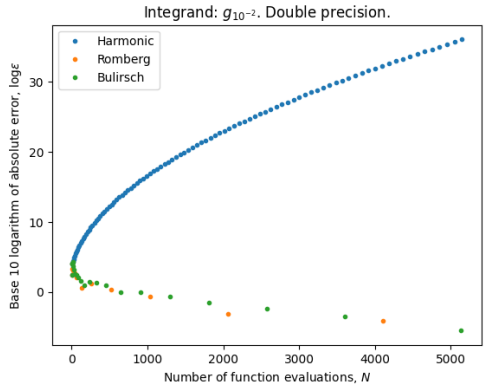


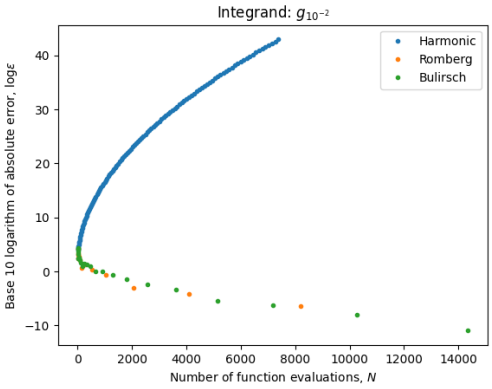


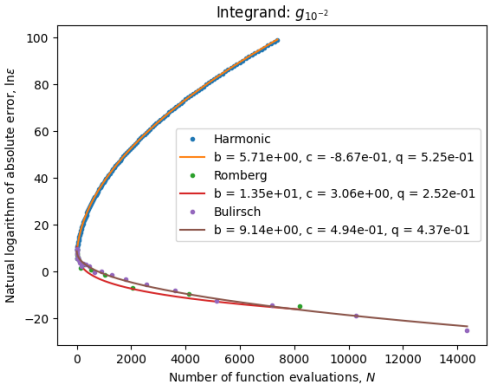
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	5.2[+00]	7.7[−02]	−7.2[−01]	1.8[−03]	5.1[−01]	9.0[−05]	1.6[−02]	8.7[−06]
RS-evals	7.5[+22]	6.0[+00]	1.4[+01]	3.3[−01]	2.0[−01]	7.9[−02]	9.7[+02]	1.4[−03]
BS-evals	9.0[+07]	7.6[+00]	3.0[+00]	1.9[−01]	3.6[−01]	1.3[−02]	3.4[+01]	1.7[−03]
HS-steps	1.1[+01]	5.0[−02]	−5.1[−01]	1.5[−03]	1.0[+00]	6.5[−05]	1.2[−02]	6.7[−06]
RS-steps	7.3[+02]	2.1[+00]	6.6[−02]	1.8[−01]	2.6[+00]	5.3[−03]	5.2[−01]	1.7[−04]
BS-steps	3.2[+01]	2.0[+00]	1.4[−02]	7.5[−01]	2.8[+00]	6.5[−03]	5.1[−01]	8.6[−04]

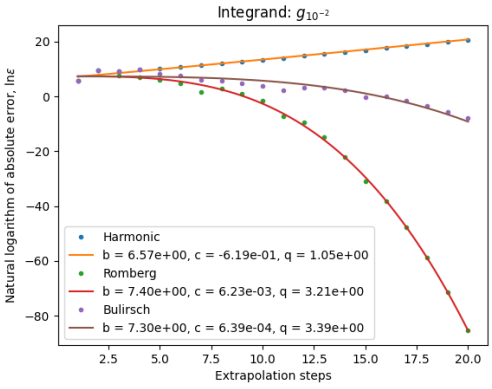
Here we get divergence for the harmonic sequence, but convergence for the other sequences, fastest for Bulirsch. In standard double precision arithmetic, we get down to machine level precision using Romberg or Bulirsch. The model for exponential convergence in number of evaluations does not fit for Bulirsch nor Romberg but it is hard to tell whether we have exponential convergence in the number of steps.

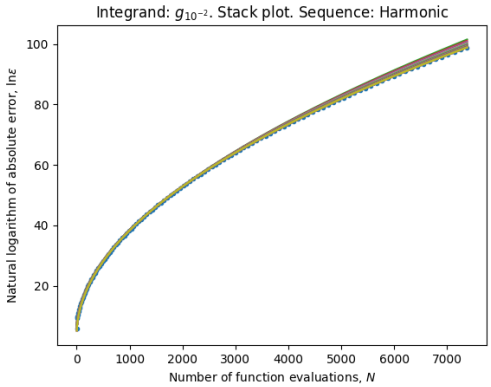
$a = 10^{-2}$

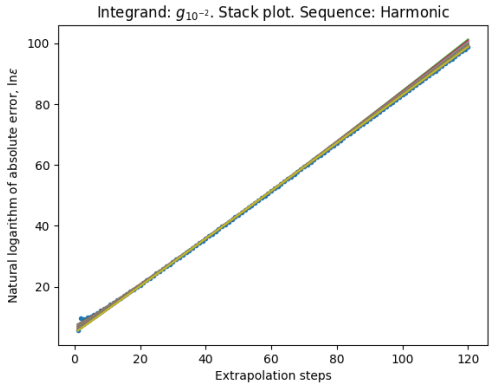

Integrand: $g_{10^{-2}}$. Double precision.
Harmonic
Romberg
Bulirsch
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

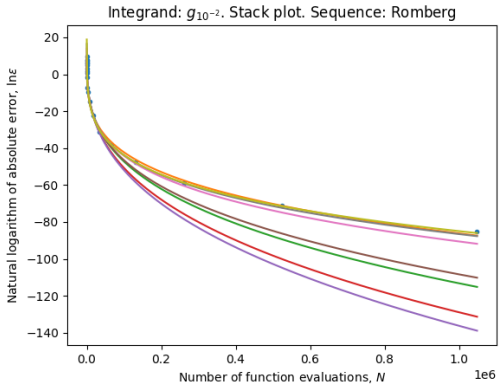

Integrand: $g_{10^{-2}}$
Harmonic
Romberg
Bulirsch
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

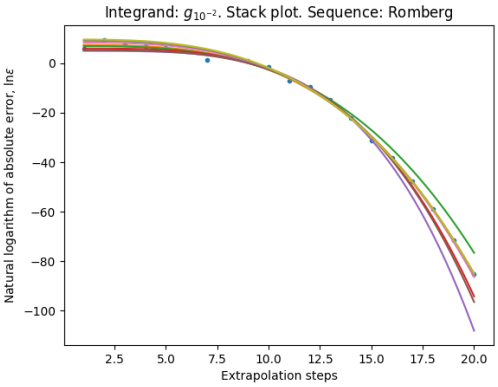

Integrand: $g_{10^{-2}}$
Harmonic
b = 5.71e+00, c = -8.67e-01, q = 5.25e-01
Romberg
b = 1.35e+01, c = 3.06e+00, q = 2.52e-01
Bulirsch
b = 9.14e+00, c = 4.94e-01, q = 4.37e-01
Natural logarithm of absolute error, lnε
Number of function evaluations, N

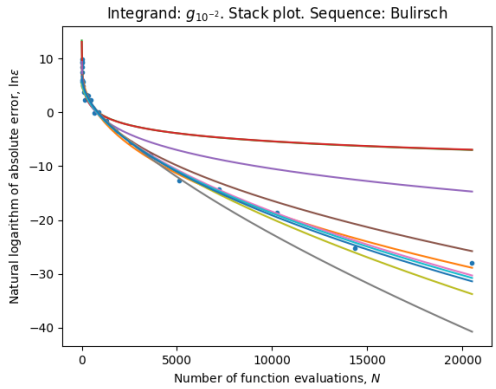

Integrand: $g_{10^{-2}}$
Harmonic
b = 6.57e+00, c = -6.19e-01, q = 1.05e+00
Romberg
b = 7.40e+00, c = 6.23e-03, q = 3.21e+00
Bulirsch
b = 7.30e+00, c = 6.39e-04, q = 3.39e+00
Natural logarithm of absolute error, lnε
Extrapolation steps

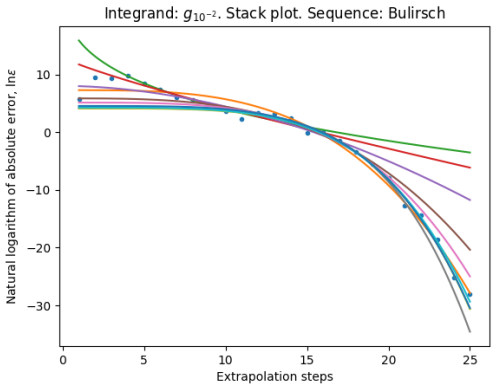

Integrand: $g_{10^{-2}}$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Integrand: $g_{10^{-2}}$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps


Integrand: $g_{10^{-2}}$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Integrand: $g_{10^{-2}}$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps





Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	1.2[+02]	8.0[−01]	−9.6[−01]	7.2[−03]	5.2[−01]	3.7[−04]	7.2[−02]	2.0[−05]
RS-evals	8.9[+10]	5.8[+00]	3.0[+00]	6.3[−01]	2.9[−01]	5.4[−02]	2.3[+00]	2.7[−03]
BS-evals	·	·	2.9[+03]	4.9[+00]	4.0[−01]	3.7[−01]	7.2[−01]	1.3[−02]
HS-steps	3.2[+02]	5.8[−01]	−6.8[−01]	6.8[−03]	1.0[+00]	3.0[−04]	5.5[−02]	1.9[−05]
RS-steps	4.4[+03]	1.5[+00]	5.4[−03]	4.5[−01]	3.4[+00]	1.0[−02]	7.6[−01]	1.1[−03]
BS-steps	1.7[+11]	8.0[+00]	1.5[+00]	6.4[+00]	3.2[+00]	2.7[−01]	8.1[−01]	1.7[−02]

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.

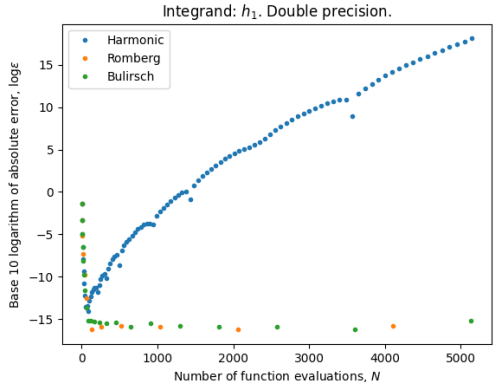
2.2.3 Logarithm

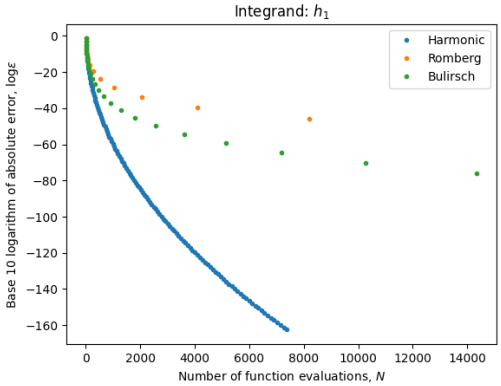
Now we will consider the following function

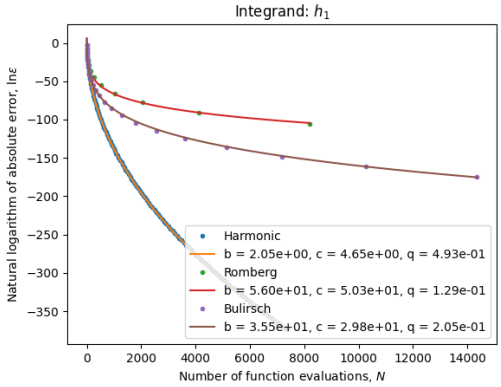
$$h_a : [0, 1] \to \mathbb{R}, \quad h_a(x) := \ln(a + x),\ a > 0.$$

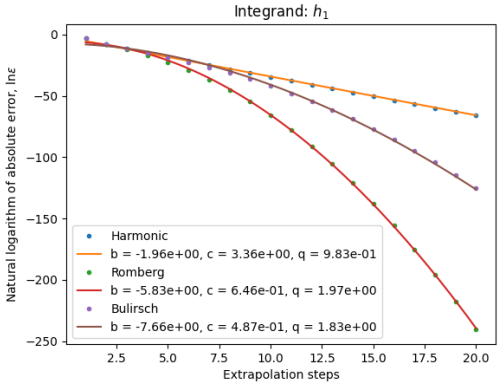
This function is analytic on neighbourhood about the interval but we have a singularity at the horizontal ray from $-a$ to $-\infty$.

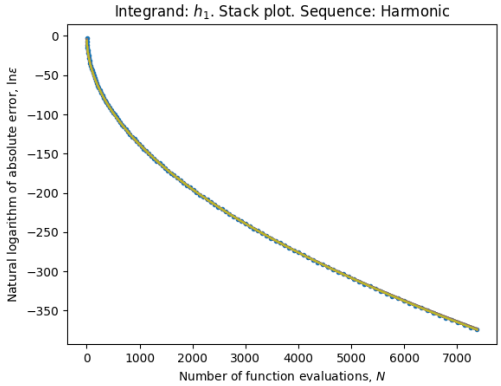
$a = 1$

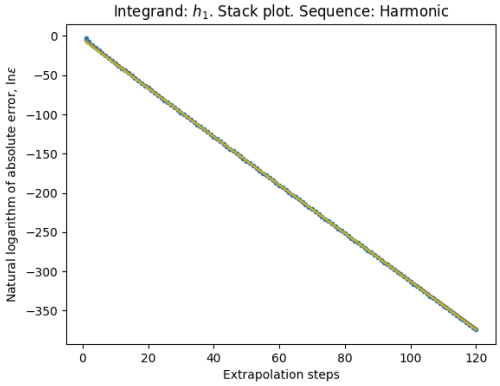


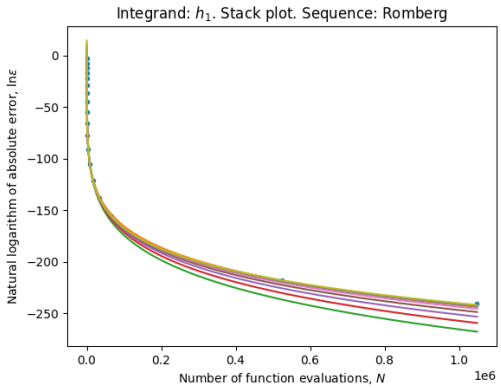


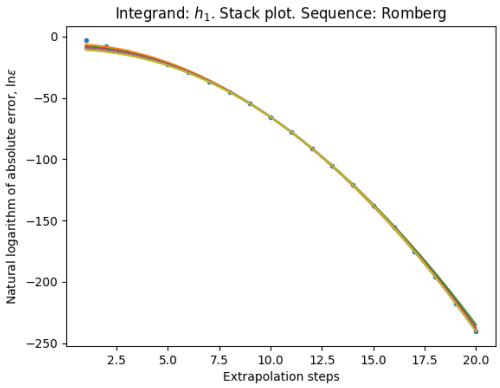

Integrand: h_1
Natural logarithm of absolute error, lnε
Number of function evaluations, N
Harmonic
b = 2.05e+00, c = 4.65e+00, q = 4.93e-01
Romberg
b = 5.60e+01, c = 5.03e+01, q = 1.29e-01
Bulirsch
b = 3.55e+01, c = 2.98e+01, q = 2.05e-01

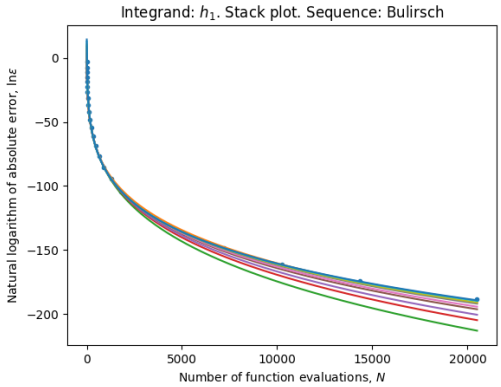

Integrand: h_1
Natural logarithm of absolute error, lnε
Extrapolation steps
Harmonic
b = -1.96e+00, c = 3.36e+00, q = 9.83e-01
Romberg
b = -5.83e+00, c = 6.46e-01, q = 1.97e+00
Bulirsch
b = -7.66e+00, c = 4.87e-01, q = 1.83e+00

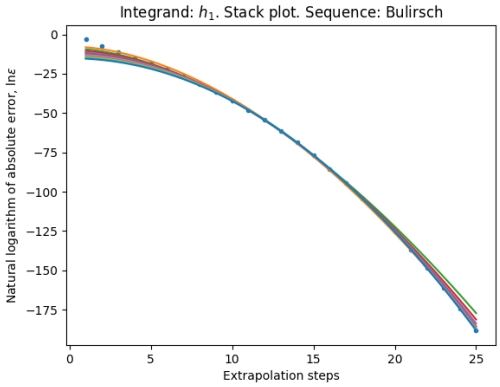

Integrand: h_1. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Integrand: h_1. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps


Integrand: h_1. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Integrand: h_1. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps


Integrand: h_1. Stack plot. Sequence: Bulirsch
Natural logarithm of absolute error, lnε
Number of function evaluations, N

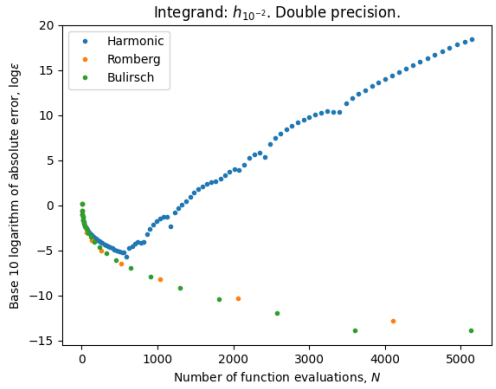

Integrand: h_1. Stack plot. Sequence: Bulirsch
Natural logarithm of absolute error, lnε
Extrapolation steps

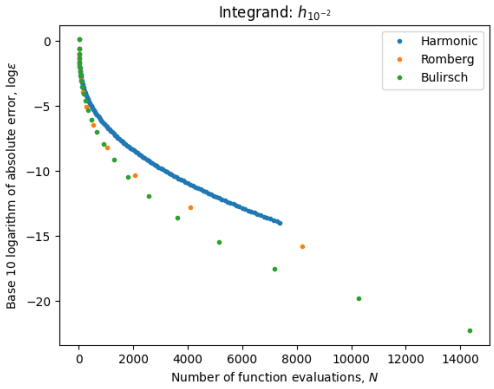
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	3.1[+00]	3.6[−01]	4.5[+00]	2.3[−04]	5.0[−01]	1.1[−05]	5.2[−01]	6.3[−07]
RS-evals	1.5[+43]	6.0[+00]	5.7[+01]	6.6[−02]	1.3[−01]	1.6[−02]	4.1[+03]	2.4[−04]
BS-evals	1.5[+31]	7.9[+00]	3.7[+01]	8.1[−02]	2.0[−01]	1.9[−02]	7.4[+03]	3.2[−04]
HS-steps	3.9[−02]	5.5[−01]	3.2[+00]	3.9[−04]	9.9[−01]	1.6[−05]	7.7[−01]	1.6[−06]
RS-steps	2.1[−04]	1.2[+00]	5.4[−01]	9.5[−03]	2.0[+00]	2.6[−04]	9.2[−01]	7.9[−05]
BS-steps	2.3[−05]	3.4[+00]	3.6[−01]	6.8[−02]	1.9[+00]	1.7[−03]	9.9[−01]	2.6[−04]

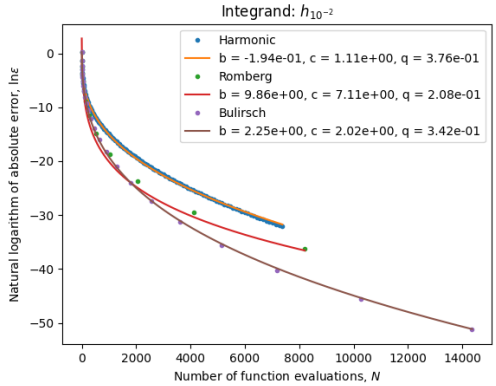
We see that the harmonic sequence performes best, then Bulirsch and then Romberg. In standard double precision arithmetic, we get down to machine level precision using Romberg or Bulirsch, but we are like 2 digits from there, using the harmonic sequence.

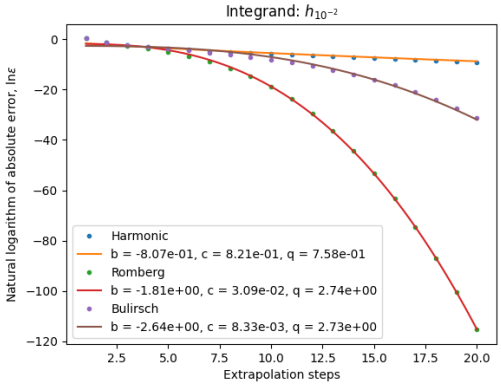
Here we clearly have exponential convergence in the number of evaluations for the harmonic sequence and hence also in the number of steps. For Romberg and Bulirsch we seem to have exponential convergence in the number of steps though the fitting is not as nice as in the case of the harmonic sequence.

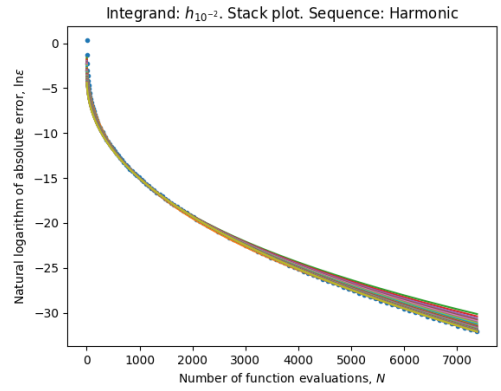
$a = 10^{-2}$

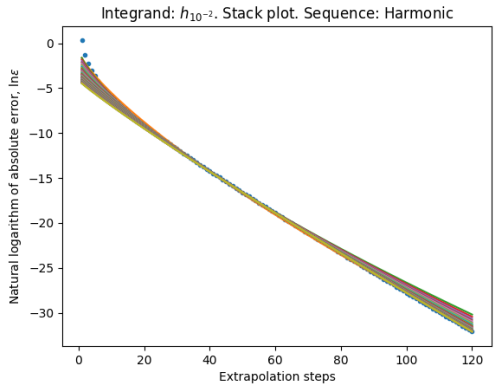


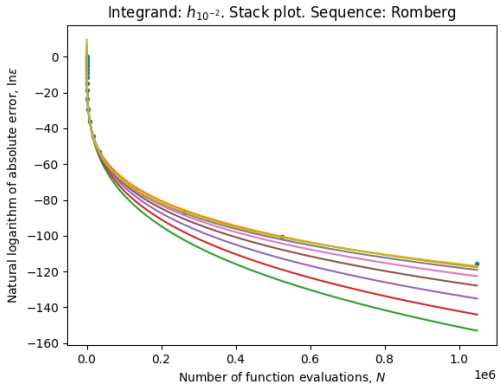


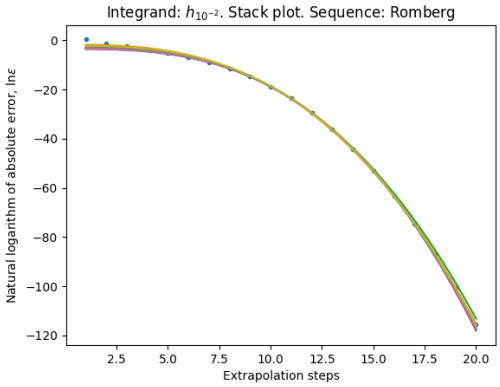


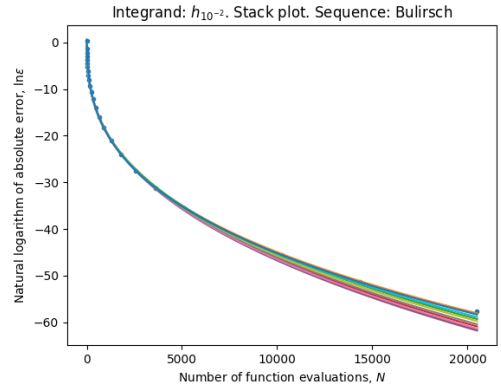


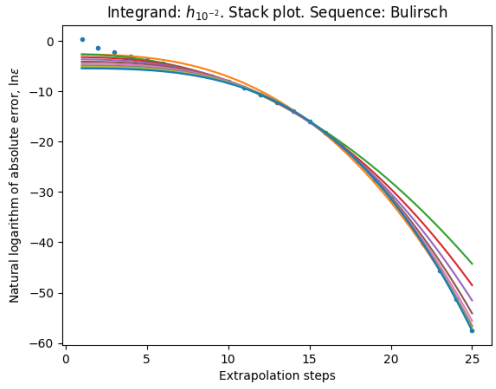










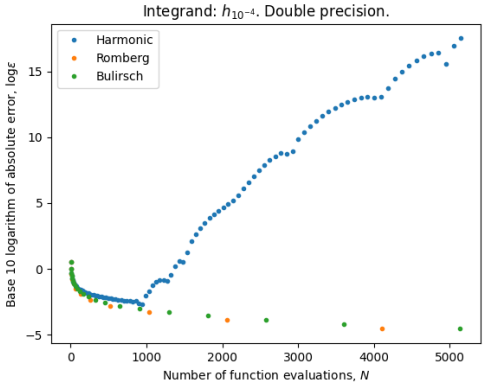


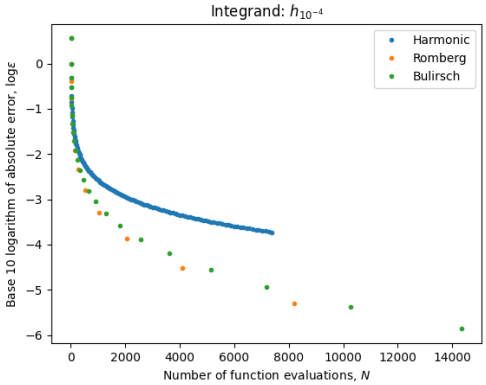
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	1.4[−01]	2.5[+00]	6.6[−01]	1.1[−01]	4.3[−01]	4.7[−03]	7.1[−01]	2.0[−04]
RS-evals	3.4[+09]	6.0[+00]	6.7[+00]	2.3[−01]	2.3[−01]	2.6[−02]	7.9[+00]	5.6[−04]
BS-evals	6.9[+00]	2.0[+00]	1.6[+00]	3.1[−02]	3.7[−01]	2.1[−03]	2.0[−01]	1.2[−04]
HS-steps	7.5[−02]	1.8[+00]	4.8[−01]	1.1[−01]	8.6[−01]	4.2[−03]	7.1[−01]	1.9[−04]
RS-steps	5.2[−02]	2.6[−01]	2.3[−02]	1.8[−02]	2.8[+00]	2.7[−04]	7.6[−01]	1.7[−04]
BS-steps	2.1[−02]	1.0[+00]	8.8[−03]	1.1[+00]	2.8[+00]	1.2[−02]	8.8[−01]	1.4[−03]

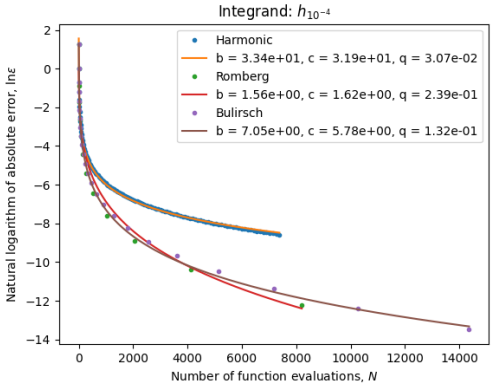
We see that we can not attain high precision using the harmonic sequence and standard double precision. It is hard to tell which sequence performes best in the long run, though we can say that Bulirsch performes better than Romberg.

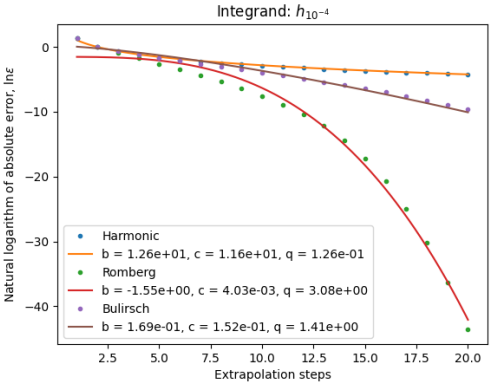
For Romberg, we get a moderately goot fit for exponential convergence in the number of steps. The fitting is quite good for the harmonic sequence but it is quite unstable for Bulirsch.

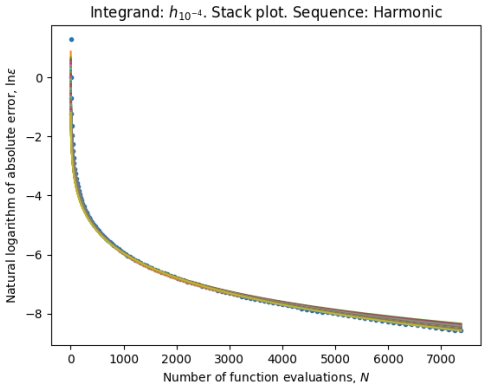
$a = 10^{-4}$

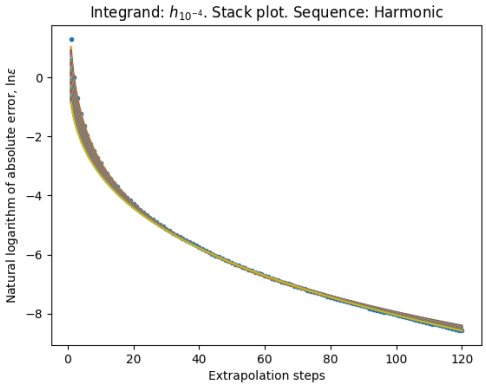


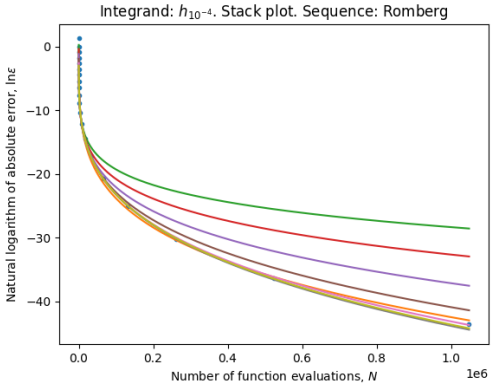


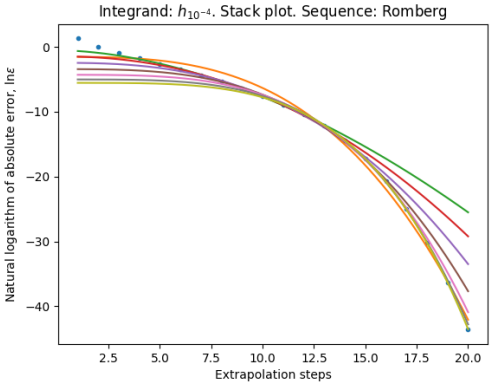


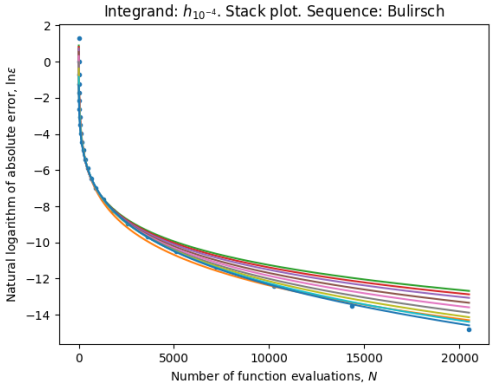


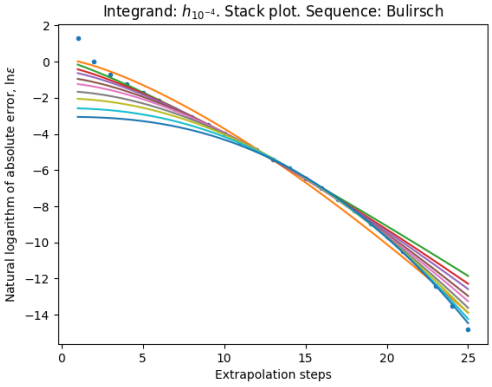












Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	1.3[+13]	5.6[+01]	8.1[+00]	5.5[−01]	9.6[−02]	9.4[−02]	1.0[−01]	9.6[−05]
RS-evals	1.1[+02]	5.4[+00]	2.0[+00]	7.3[−01]	2.4[−01]	6.4[−02]	6.1[−01]	1.3[−03]
BS-evals	1.9[+05]	5.4[+00]	6.4[+00]	3.3[−01]	1.4[−01]	1.3[−01]	1.7[−01]	8.8[−04]
HS-steps	1.4[+04]	1.7[+01]	5.3[+00]	2.0[−01]	2.2[−01]	4.7[−02]	4.4[−02]	4.0[−05]
RS-steps	1.4[−01]	2.1[+00]	2.4[−02]	2.5[+00]	3.0[+00]	7.0[−02]	8.6[−01]	4.4[−03]
BS-steps	4.1[−01]	7.0[−01]	9.5[−02]	8.4[−01]	1.7[+00]	5.4[−02]	4.7[−01]	3.6[−03]

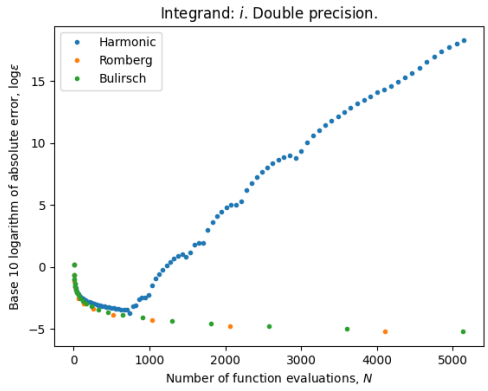
Here again, we do not attain high precision when using the Harmonic sequence in double precision arithmetic. It is hard to say which sequence performes best. We get reasonably good fit for the harmonic sequence but not for Romberg and Bulirsch.

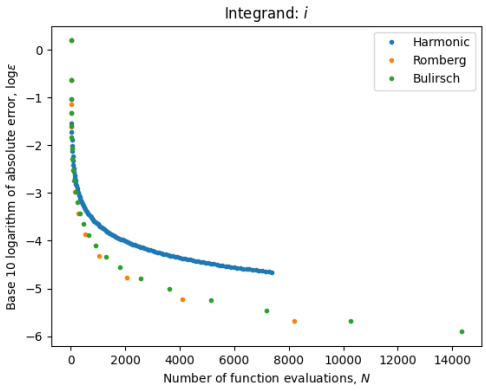
2.2.4 Area of half circle

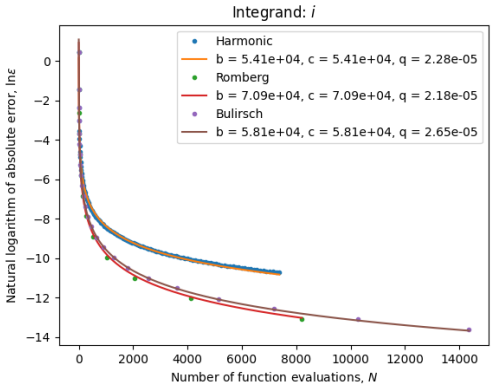
Now we will try the following function:

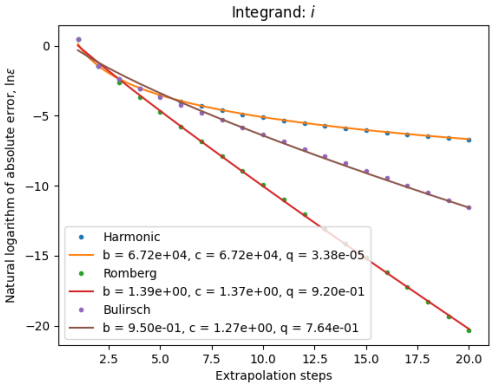
$$i : [-1, 1] \to \mathbb{R}, \quad i(x) := \sqrt{1 - x^2}.$$

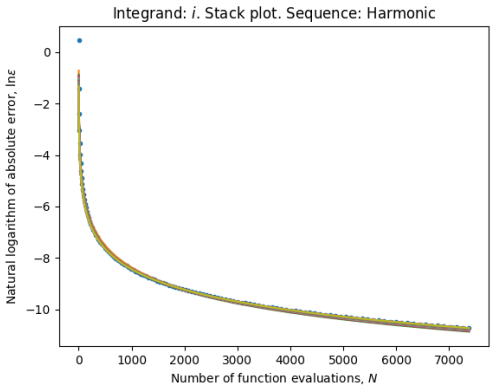
This function is analytic inside the interval of definition but not at the endpoints. Its derivative has singularities at the endpoints.

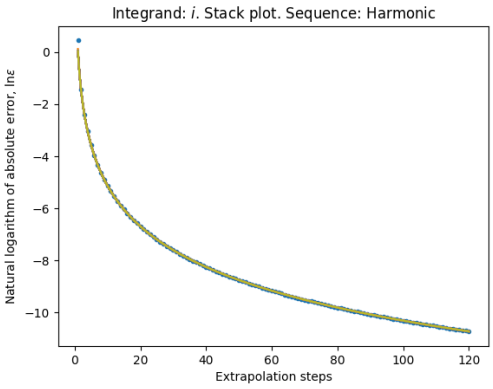

Integrand: i. Double precision.
Harmonic
Romberg
Bulirsch
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

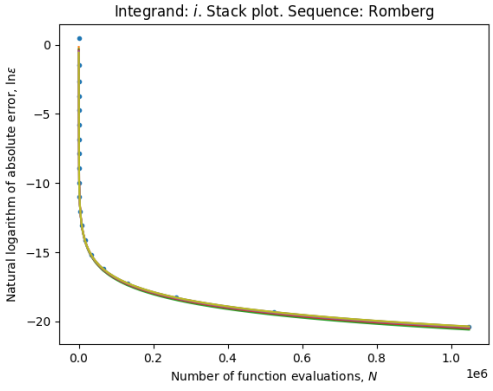

Integrand: i
Harmonic
Romberg
Bulirsch
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

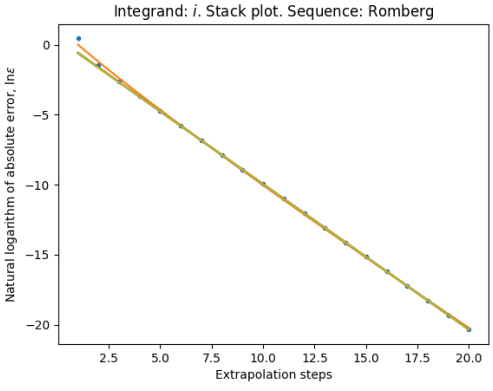

Integrand: i
Harmonic
b = 5.41e+04, c = 5.41e+04, q = 2.28e-05
Romberg
b = 7.09e+04, c = 7.09e+04, q = 2.18e-05
Bulirsch
b = 5.81e+04, c = 5.81e+04, q = 2.65e-05
Natural logarithm of absolute error, lnε
Number of function evaluations, N

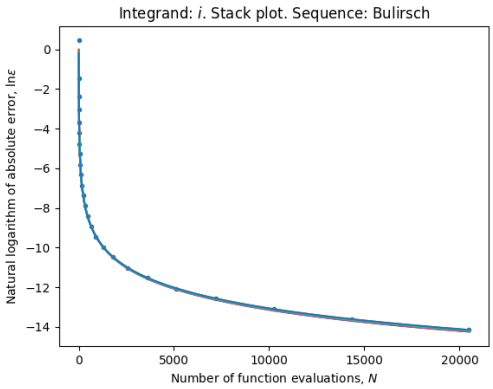

Integrand: i
Harmonic
b = 6.72e+04, c = 6.72e+04, q = 3.38e-05
Romberg
b = 1.39e+00, c = 1.37e+00, q = 9.20e-01
Bulirsch
b = 9.50e-01, c = 1.27e+00, q = 7.64e-01
Natural logarithm of absolute error, lnε
Extrapolation steps

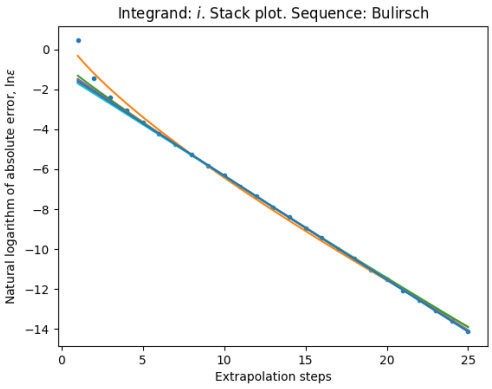

Integrand: i. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Integrand: i. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps


Integrand: i. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Integrand: i. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps



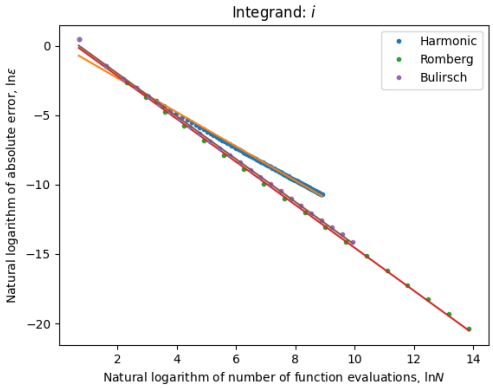


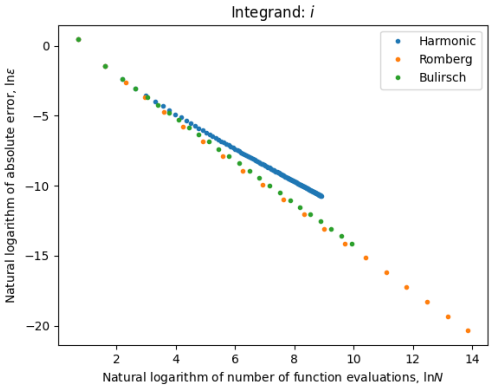
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	$\rho_{\ln}$
HS-evals	·	·	5.4[+04]	3.9[−03]	2.2[−05]	3.5[−03]	−1.0[+00]	2.5[−04]
RS-evals	·	·	7.3[+04]	5.7[−03]	2.1[−05]	5.1[−03]	−1.0[+00]	2.2[−04]
BS-evals	·	·	5.8[+04]	3.2[−04]	2.6[−05]	2.4[−04]	−1.0[+00]	1.3[−04]
HS-steps	·	·	6.7[+04]	2.4[−04]	3.4[−05]	2.5[−04]	−1.0[+00]	1.4[−05]
RS-steps	1.6[+00]	5.1[−04]	1.0[+00]	5.4[−05]	1.0[+00]	5.0[−06]	1.2[−01]	1.4[−04]
BS-steps	3.7[−01]	3.1[−02]	5.6[−01]	8.4[−03]	9.8[−01]	6.3[−04]	2.8[−01]	7.0[−04]

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 we reject the fitting because we get unreasoable values of the parameters. The model for exponential convergence in the number of steps fits well in the case of the Romberg sequence and Bulirsch.

If we plot the logarithm of the error agains the logarithm of the evaluations we see that the points seem to fit on a line. Hence, the error seems to converge algebraically with the number of evaluations.

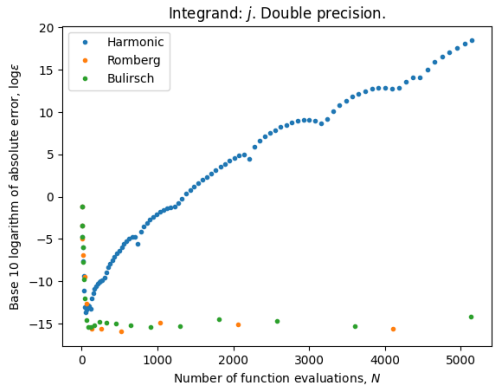


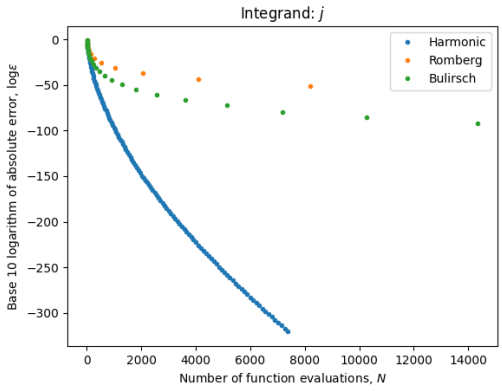


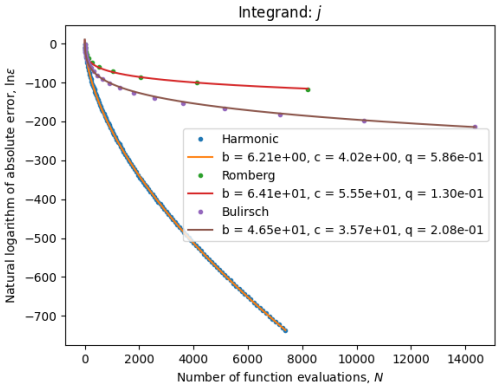
2.2.5 Gaussian

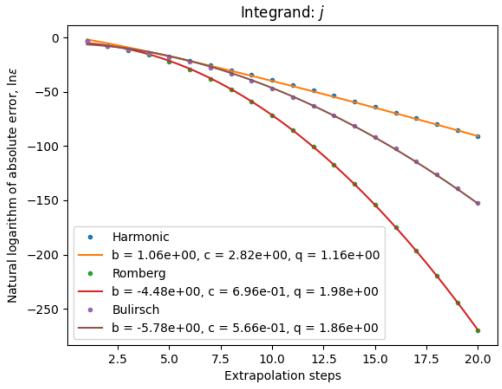
Finally we will consider the Gaussian function

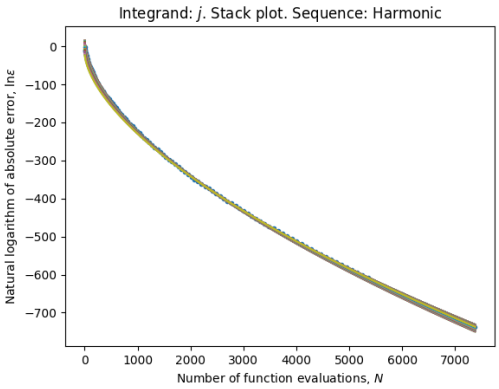
$$j : [0,1] \to \mathbb{R}, \quad k(x) := \frac{2}{\sqrt{\pi}} e^{-x^2}.$$

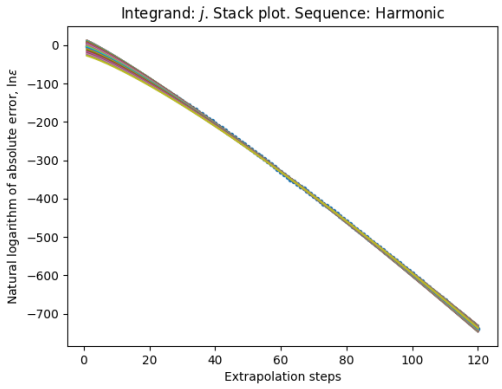

Integrand: j. Double precision.
Harmonic
Romberg
Bulirsch
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

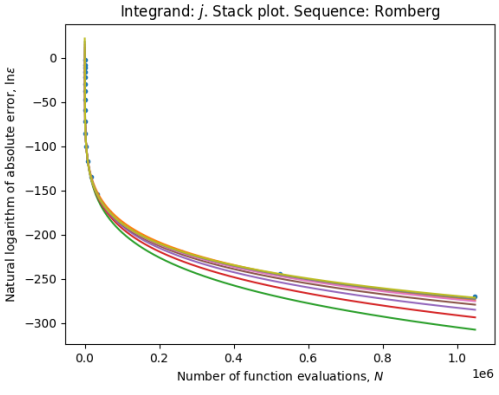

Integrand: j
Harmonic
Romberg
Bulirsch
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

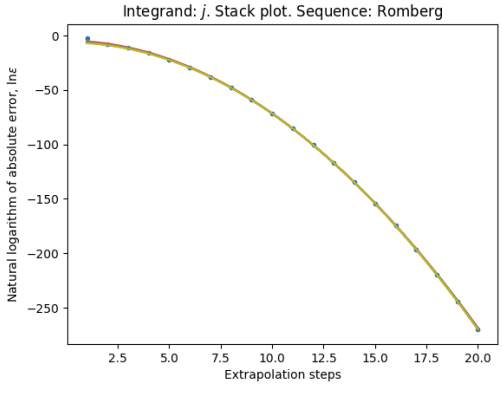

Integrand: j
Harmonic
b = 6.21e+00, c = 4.02e+00, q = 5.86e-01
Romberg
b = 6.41e+01, c = 5.55e+01, q = 1.30e-01
Bulirsch
b = 4.65e+01, c = 3.57e+01, q = 2.08e-01
Natural logarithm of absolute error, lnε
Number of function evaluations, N

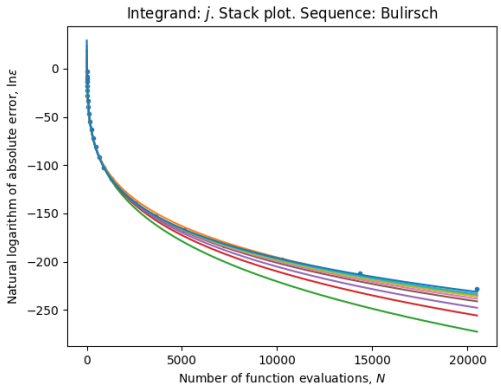

Integrand: j
Harmonic
b = 1.06e+00, c = 2.82e+00, q = 1.16e+00
Romberg
b = -4.48e+00, c = 6.96e-01, q = 1.98e+00
Bulirsch
b = -5.78e+00, c = 5.66e-01, q = 1.86e+00
Natural logarithm of absolute error, lnε
Extrapolation steps

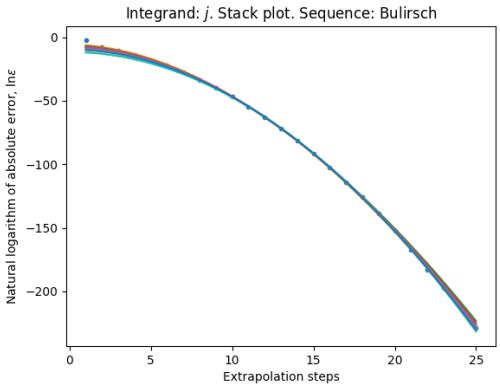

Integrand: j. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Integrand: j. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps


Integrand: j. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Integrand: j. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps





Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
HS-evals	1.0[+09]	5.5[+00]	4.3[+00]	2.1[−02]	5.8[−01]	7.3[−04]	2.5[+02]	6.2[−06]
RS-evals	1.4[+55]	6.0[+00]	6.8[+01]	8.3[−02]	1.2[−01]	2.3[−02]	1.5[+05]	4.2[−04]
BS-evals	1.4[+51]	8.0[+00]	5.0[+01]	1.2[−01]	1.9[−01]	3.0[−02]	7.9[+06]	8.2[−04]
HS-steps	6.9[+05]	5.4[+00]	2.9[+00]	2.2[−02]	1.2[+00]	6.6[−04]	2.0[+00]	4.7[−06]
RS-steps	3.9[−03]	1.5[−01]	6.6[−01]	8.3[−04]	2.0[+00]	2.4[−05]	8.5[−01]	2.6[−05]
BS-steps	4.0[−04]	1.5[+00]	4.7[−01]	2.5[−02]	1.9[+00]	7.1[−04]	9.5[−01]	7.7[−05]

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 it is hard to tell whether we have exponential convergence because we get very large value for A. For Romberg we seem to have exponential convergence in the number steps and also for Bulirsch.

2.3 Summary

When the integrand is entire, we have exponential convergence for the Romberg and Bulirsch sequence in the number of steps. The results are not as clear for the harmonic sequence in those cases. It is interesting how fast convergence we get using the harmonic sequence.

When the integrand has a pole, which is not so close to the interval of integration, we have exponential convergence for the harmonic sequence. For the Romberg and Bulirsch, the fitting is not as clear. When we move closer to the pole, the model fails in all cases.

When the integrand is not analytic on a neighbourhood around the interval of integration, we seem to have exponential convergence in the number of steps for the Romberg and Bulirsch sequence, but the fitting fails for the harmonic sequence as we get either very big or small parameters. In this case, it the error seems to follow algebraically with the number of evaluations and steps.

It should be mentioned that in all cases, we used the length of the interval as initial step size. However, whether the methods converge or not may depend on the initial step size.

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 $2 \le j \le i$.

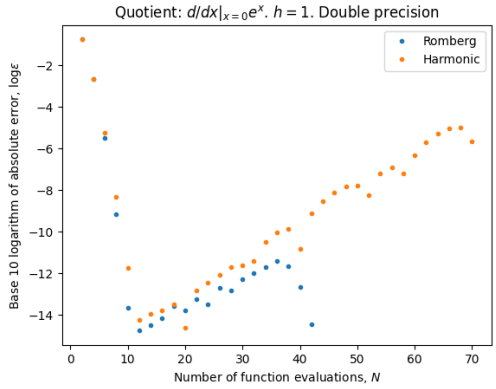
3.2 Numerical experiments

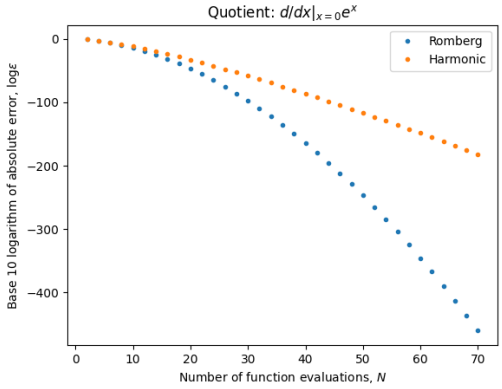
In this section we are going to extrapolate the symmetric difference quotient for approximating the derivative of a function at a given point. Let $h > 0$ be some number, $f :]a-\varepsilon, a+\varepsilon[\to \mathbb{R}$ a function differentiable at a and $n_1 < n_2 < \cdots$ a sequence of integers. 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 analyse how ε_i as i increases and we also want to do similar efficiency analysis 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.

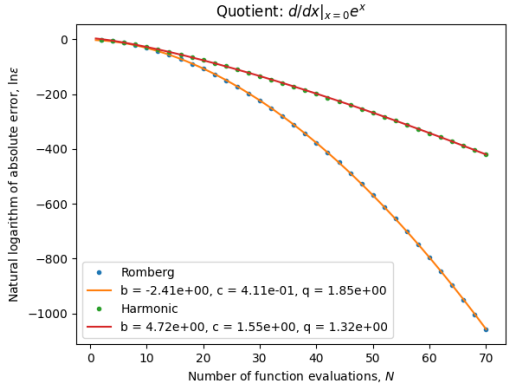
Now we will consider the results of the experiments.

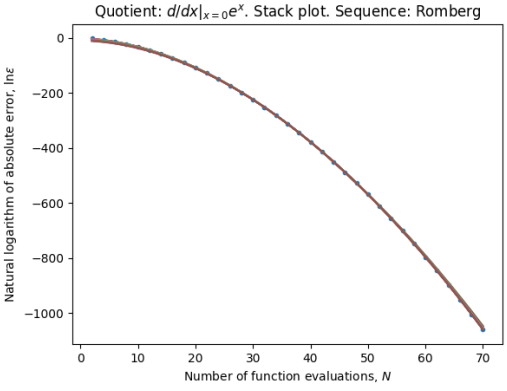
3.2.1 The exponential function

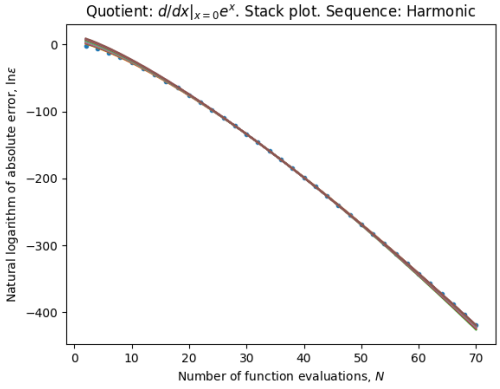
We begin by considering the derivative of the exponential function at zero.











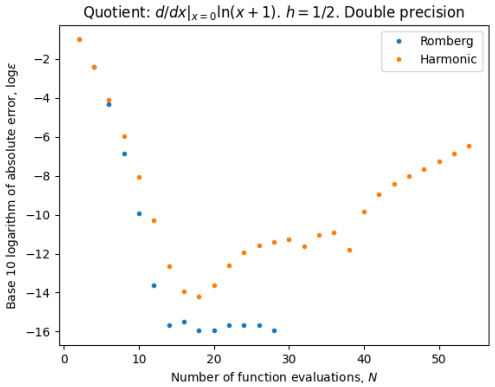
Sequence	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS	3.7[−02]	2.6[+00]	4.0[−01]	4.0[−03]	1.9[+00]	6.7[−05]	7.8[−01]	3.0[−06]
HS	1.1[+05]	3.3[+00]	1.7[+00]	7.2[−03]	1.3[+00]	2.4[−04]	1.5[+02]	1.2[−05]

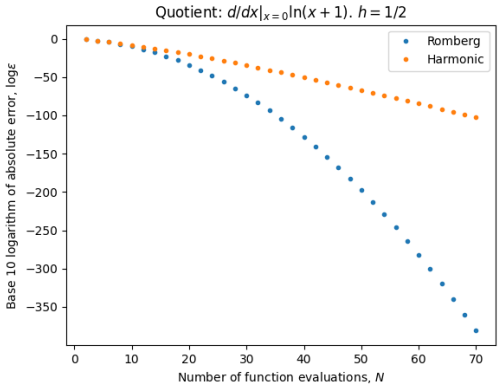
In standard floating point arithmetic, we get down to machine level precision using both sequences. The Romberg sequence works beter. The model seems to fit well in all cases.

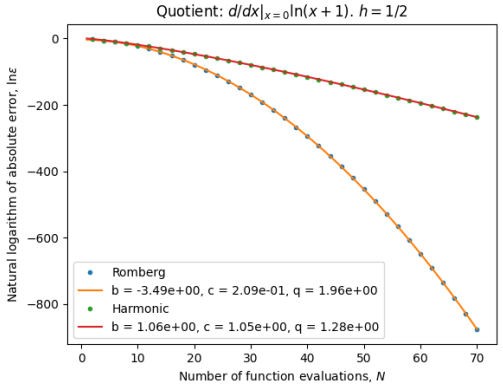
3.2.2 Logarithm

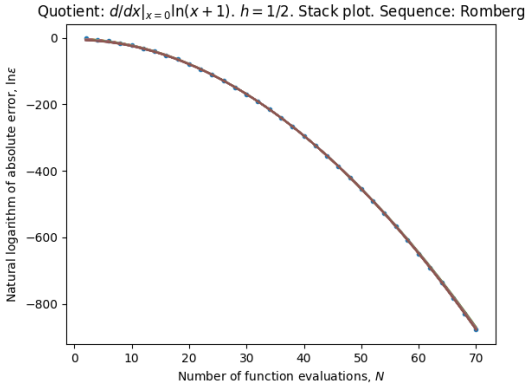
Now we will consider the dervative at zero of the function

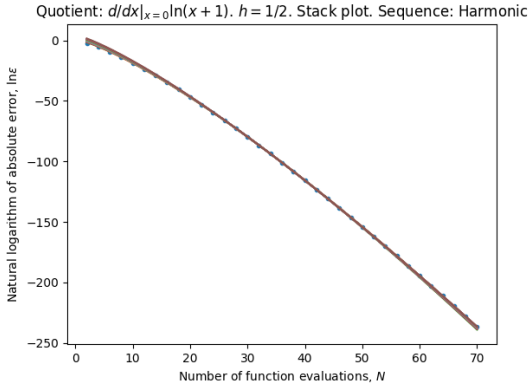
$$g(x) \coloneqq \ln(x+1).$$











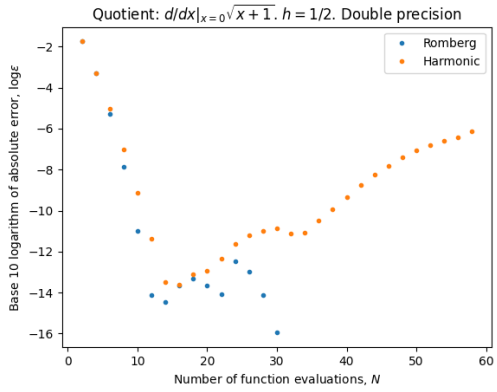
Sequence	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	$\rho_{\ln}$
RS	9.1[−03]	9.8[−01]	2.0[−01]	1.4[−03]	2.0[+00]	2.1[−05]	7.5[−01]	2.1[−06]
HS	2.8[+01]	9.6[−01]	1.1[+00]	3.5[−03]	1.3[+00]	1.2[−04]	1.7[+00]	4.5[−06]

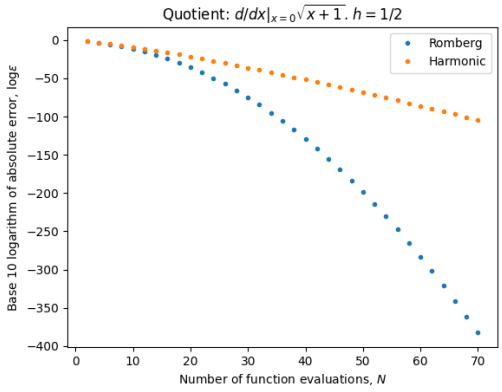
We get down to machine level precision using both sequences, Romberg performes better. The model fits well in all cases.

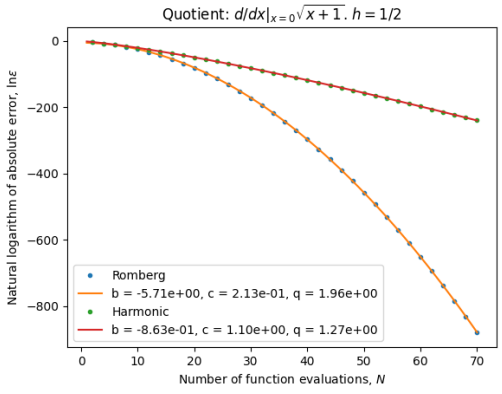
3.2.3 Square root

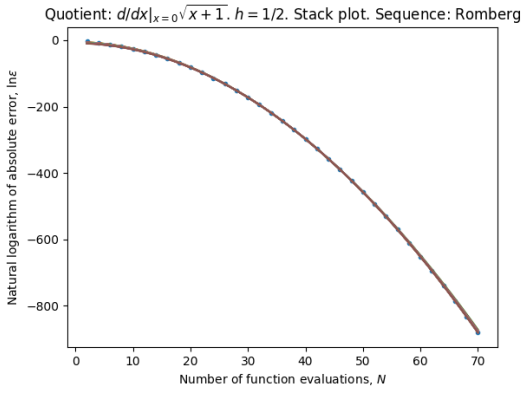
Now we shall consider the derivative at zero of the following function:

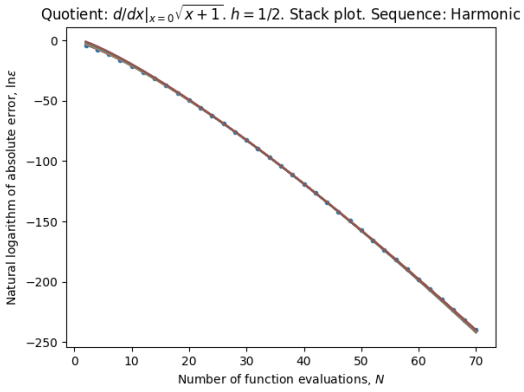
$$h(x) := \sqrt{1+x}$$











Sequence	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS	8.3[−04]	1.3[+00]	2.1[−01]	1.8[−03]	2.0[+00]	2.7[−05]	8.4[−01]	2.9[−06]
HS	2.6[+00]	8.2[−01]	1.2[+00]	2.7[−03]	1.3[+00]	8.9[−05]	5.0[−01]	2.5[−06]

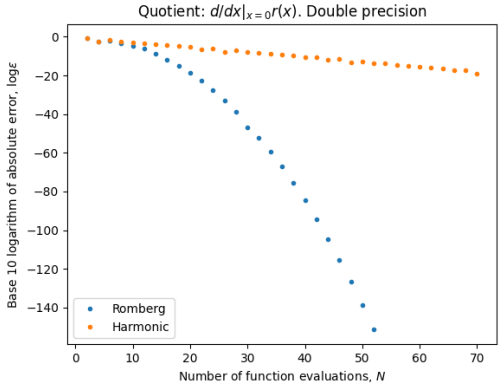
In standard double precision floating point arithmetic we get down to machine level precision using any sequence. The model fits well in all cases.

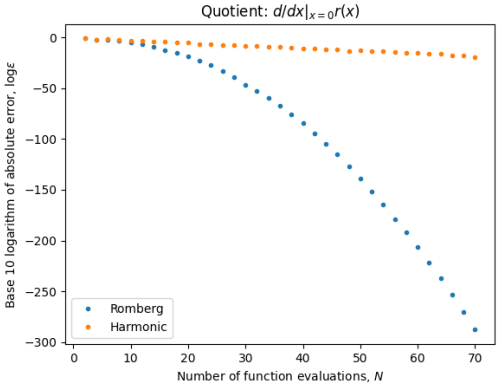
3.2.4 Smooth but not analytic function

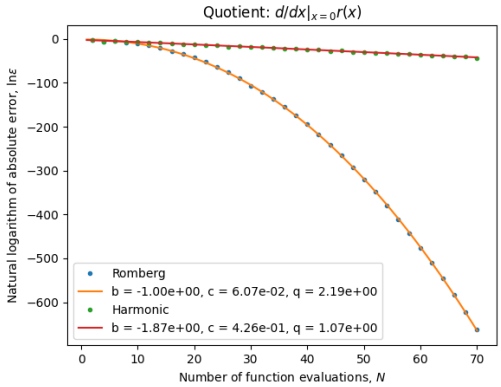
Now we will consider the derivate at zero of the following function:

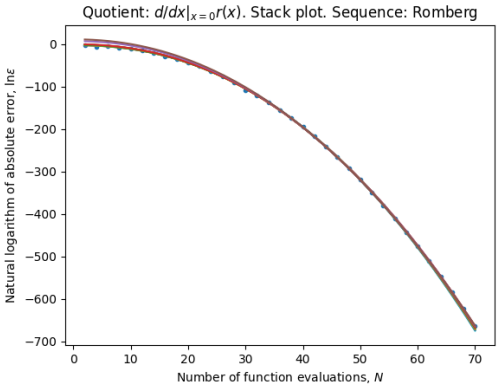
$$r(x) := \begin{cases} e^{-1/x} & \text{if } x > 0 \\ 0 & \text{else} \end{cases}$$

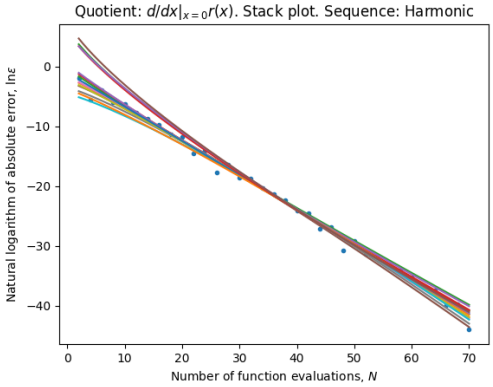
which is smooth but not analytic.











Sequence	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	$\rho_{\ln}$
RS	7.3[+03]	1.3[+01]	6.0[−02]	2.3[−02]	2.2[+00]	2.1[−04]	1.7[+00]	2.2[−05]
HS	2.9[+02]	4.8[+00]	8.2[−01]	4.8[−01]	9.8[−01]	2.3[−02]	3.1[−01]	1.2[−03]

Romberg performes better.

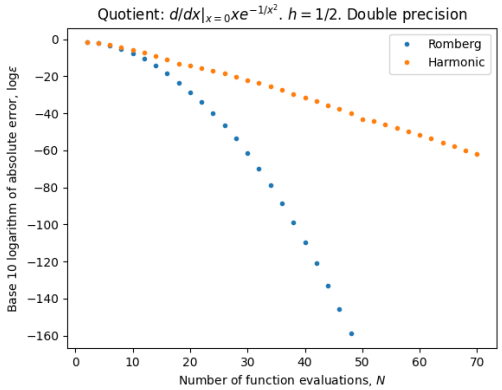
The model seems to fit reasonably well for the Romberg sequence but the fitting is not so nice for the harmonic sequence.

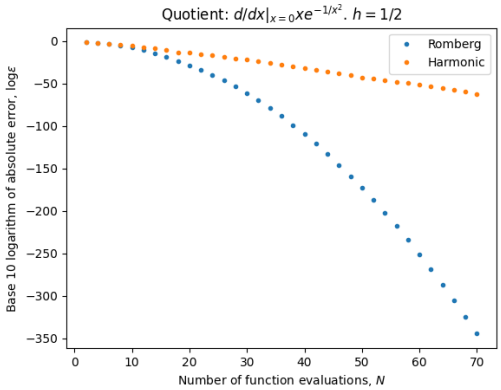
3.2.5 Another smooth but not analytic function

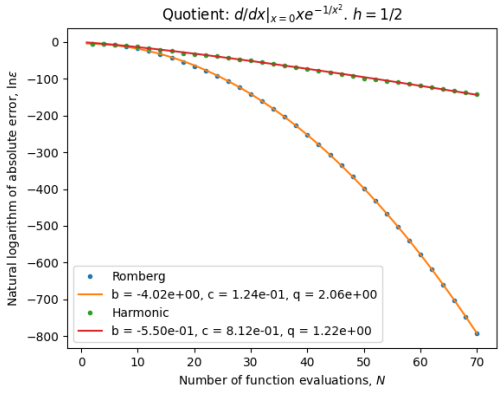
Now we will consider the derivative at zero of the following function:

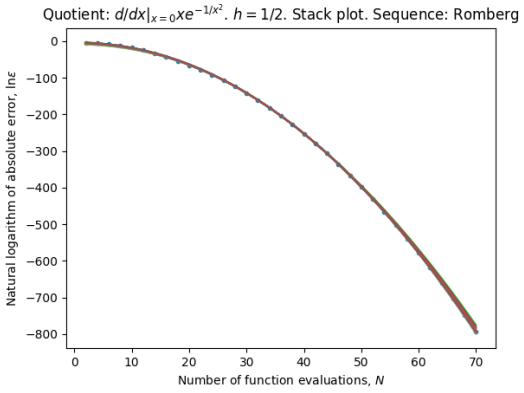
$$i(x) := \begin{cases} xe^{-1/x^2} & \text{if } x \neq 0 \\ 0 & \text{else} \end{cases}$$

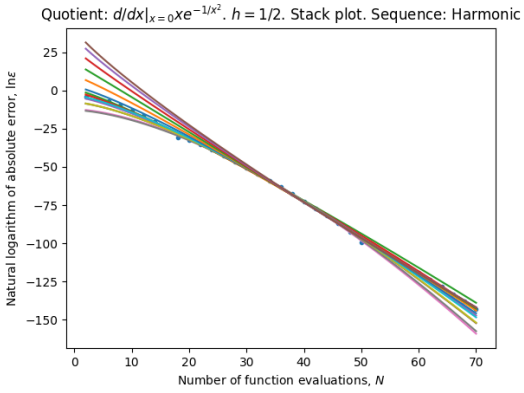
which is smooth but not analytic.











Sequence	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS	9.2[−03]	1.7[+00]	1.2[−01]	7.0[−03]	2.1[+00]	9.6[−05]	2.0[−01]	4.4[−06]
HS	2.1[+16]	1.3[+01]	1.5[+00]	1.0[+00]	1.2[+00]	3.7[−02]	1.3[+01]	1.8[−04]

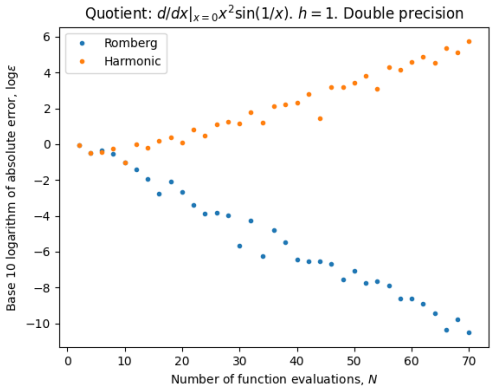
Here the Romberg sequence performes better.

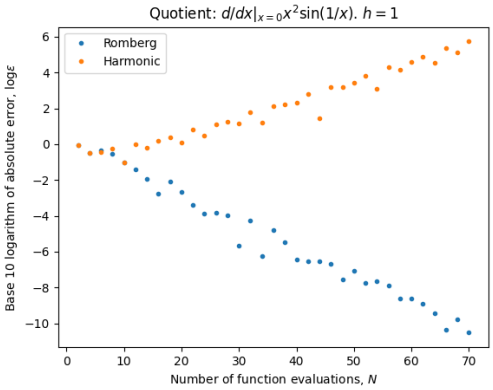
We seem to have nice fit for the Romberg sequence but not so good for the harmonic sequence.

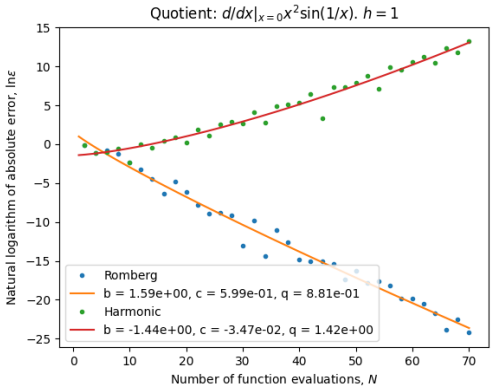
3.2.6 Only once differentiable function

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

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







Sequence	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS	2.0[+27]	1.3[+01]	6.1[+00]	3.8[+00]	8.1[−01]	4.0[−01]	7.1[−01]	4.5[−03]
HS	4.0[−01]	1.7[+00]	−3.7[−01]	6.7[+00]	1.3[+00]	9.3[−02]	9.3[−02]	1.5[−02]

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.

3.3 Summary

We the function is analytic we get exponential convergence. When the function is infinitely differentiable but not analytic, we get exponential convergence for the Romberg sequence but not for the harmonic sequence. When the function is only once differentiable, we do not get any fitting.

It is worth mentioning that the Romberg sequence works better 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 method* (see e.g. section 4.3.3 in [1]) 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 $h := (b - a)/2n$. We then define recursively

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

and

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

Then ξ_h is an approximate solution to (4.1) defined at $a, a + h, \ldots, b$. We are interested in the value $X_f(h) := \xi_h(b)$. It is shown in section 4.3.3. in [1] 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 high precision. 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.

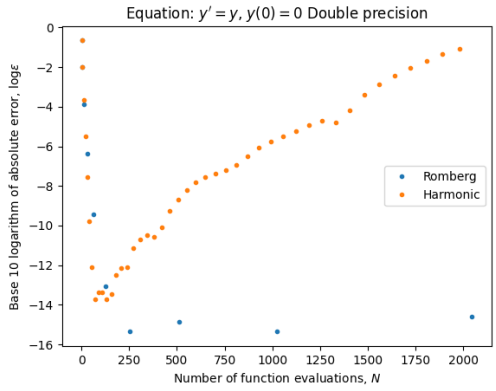
Now we will consider the results of the experiments.

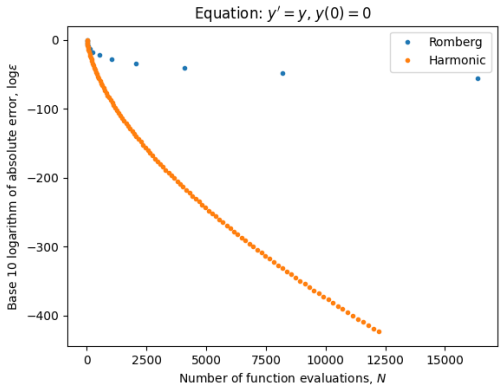
4.2.1 Exponential growth

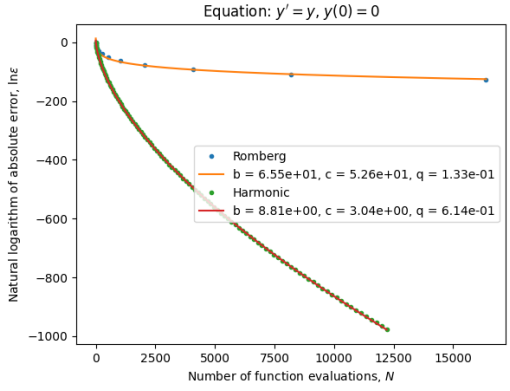
First we will consider the following initial value problem:

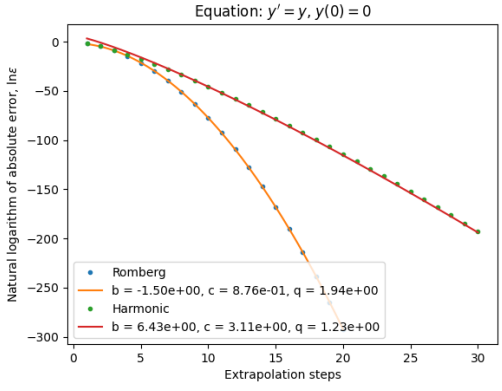
$$y'(x) = y(x), \quad y(0) = 1, \quad x \in [0, 1] \tag{4.2}$$

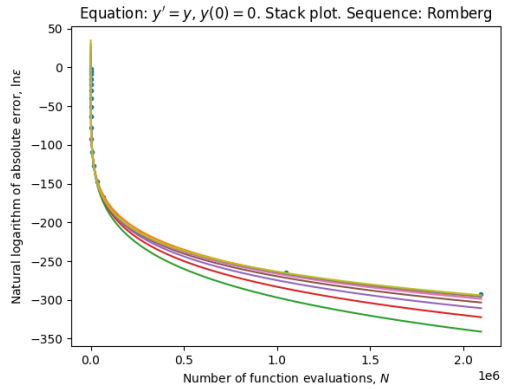
whose solution is the analytic function $y(x) = e^x$.

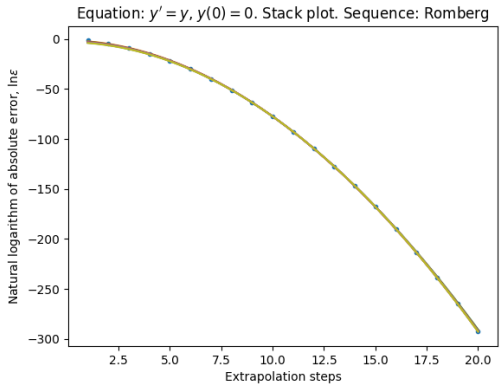


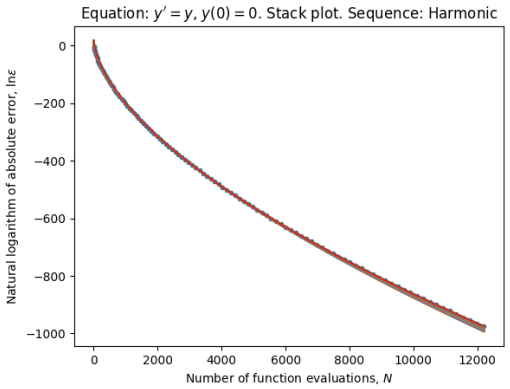


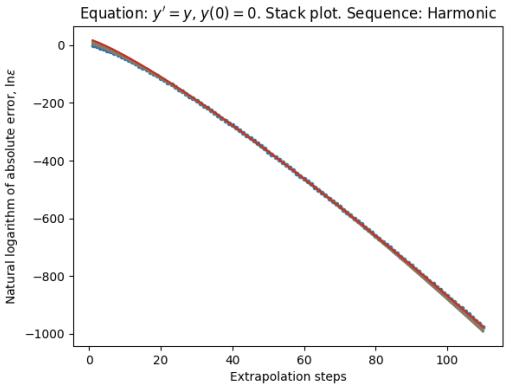












Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	$\rho_{\ln}$
RS-evals	6.2[+65]	6.0[+00]	7.2[+01]	1.2[−01]	1.2[−01]	3.1[−02]	1.4[+08]	6.9[−04]
HS-evals	1.2[+09]	6.4[+00]	3.3[+00]	5.2[−03]	6.1[−01]	1.7[−04]	8.4[+04]	5.6[−06]
RS-steps	8.7[−02]	2.2[−01]	8.4[−01]	9.2[−04]	1.9[+00]	2.8[−05]	3.3[−01]	4.5[−06]
HS-steps	3.8[+07]	6.0[+00]	3.3[+00]	4.5[−03]	1.2[+00]	1.4[−04]	1.6[+04]	4.5[−06]

The Harmonic sequence performes better. We get down to machine level precision using either sequence in double precision arithmetic.

We clearly have exponential convergence in the number of steps for the Romberg sequence and the fitting for the Harmonic sequence seems nice but we though have very big values for A.

4.2.2 Logistic curve

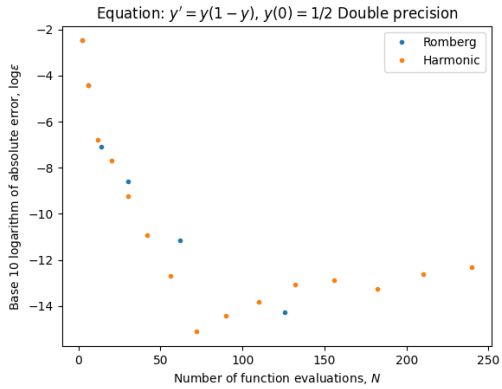
Then we will consider the following initial value problem

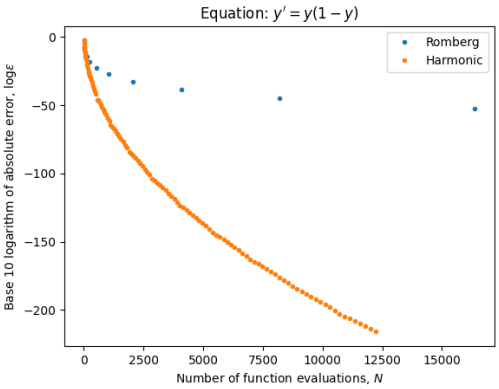
$$y'(x) = y(x)(1 - y(x)), \quad y(0) = 1/2, \quad x \in [0, 1] \tag{4.3}$$

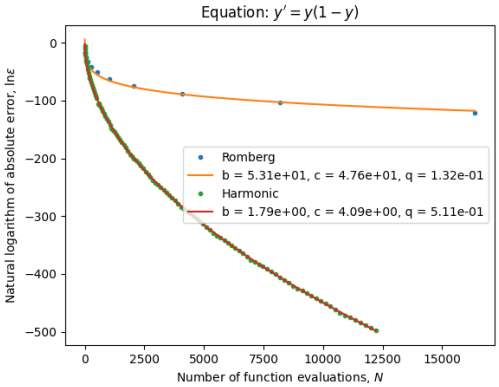
whose solution is the sigmoid function

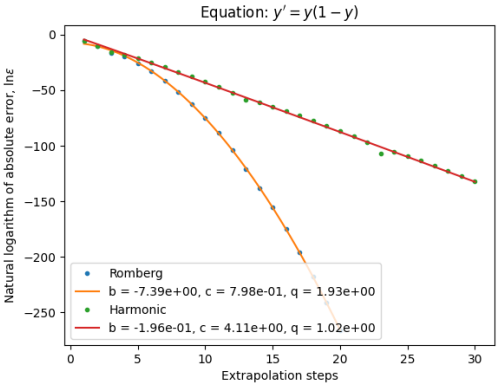
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

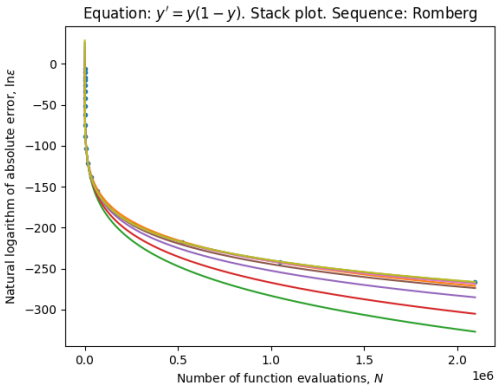
which is analytic.

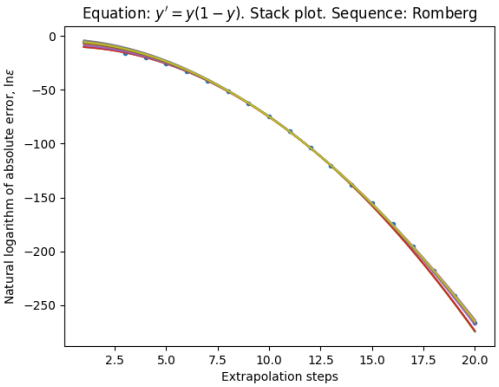

Equation: $y' = y(1-y)$, $y(0) = 1/2$ Double precision
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

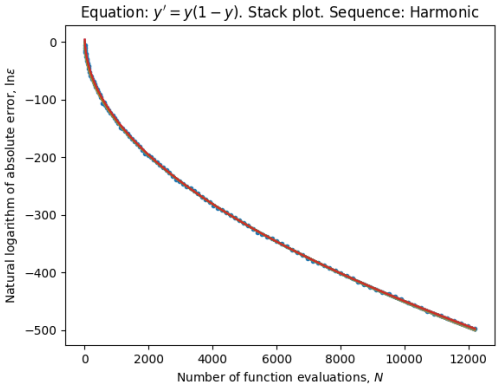

Equation: $y' = y(1-y)$
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

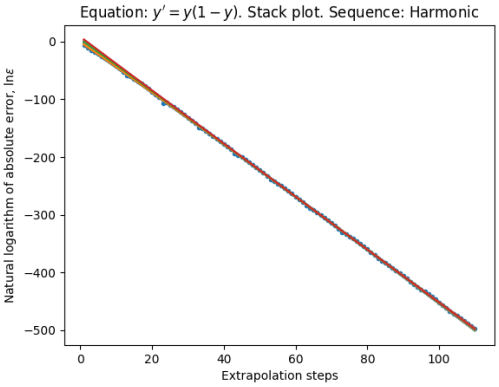

Equation: $y' = y(1-y)$
Romberg
b = 5.31e+01, c = 4.76e+01, q = 1.32e-01
Harmonic
b = 1.79e+00, c = 4.09e+00, q = 5.11e-01
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Equation: $y' = y(1-y)$
Romberg
b = -7.39e+00, c = 7.98e-01, q = 1.93e+00
Harmonic
b = -1.96e-01, c = 4.11e+00, q = 1.02e+00
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = y(1-y)$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Equation: $y' = y(1-y)$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = y(1-y)$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Equation: $y' = y(1-y)$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps

Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	8.8[+63]	6.0[+00]	7.1[+01]	2.0[−01]	1.2[−01]	6.4[−02]	6.4[+05]	6.1[−04]
HS-evals	2.5[+03]	9.4[+00]	4.2[+00]	3.6[−03]	5.1[−01]	1.3[−04]	1.9[+01]	1.3[−05]
RS-steps	1.0[−02]	1.9[+00]	8.1[−01]	3.1[−02]	1.9[+00]	1.1[−03]	8.4[−01]	4.6[−05]
HS-steps	2.6[+02]	9.2[+00]	4.3[+00]	3.5[−03]	1.0[+00]	1.3[−04]	9.3[+00]	1.3[−05]

The harmonic sequence performes better and we get down to machine level precisision using either sequence, in double precision.

We seem to have exponential convergence in the number of steps for the Romberg sequence and the models fit well for the harmonic sequence.

4.2.3 Tangens

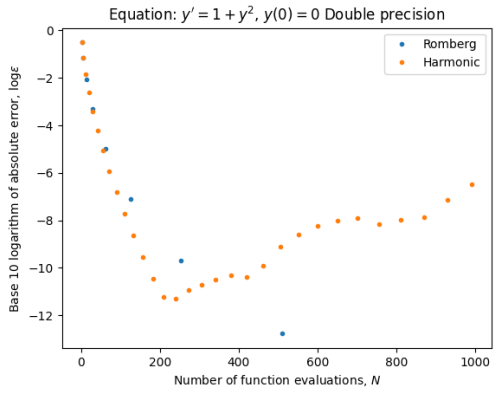
Now we will consider the following equation

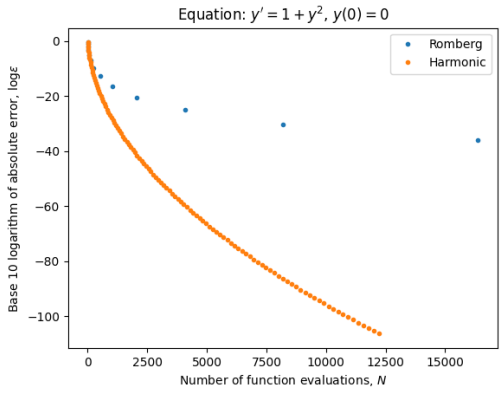
$$y'(x) = 1 + y(x)^2, \quad y(0) = 0, \quad x \in [0, 1] \tag{4.4}$$

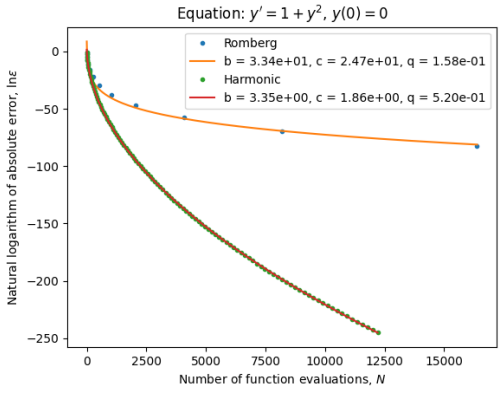
whose solution is

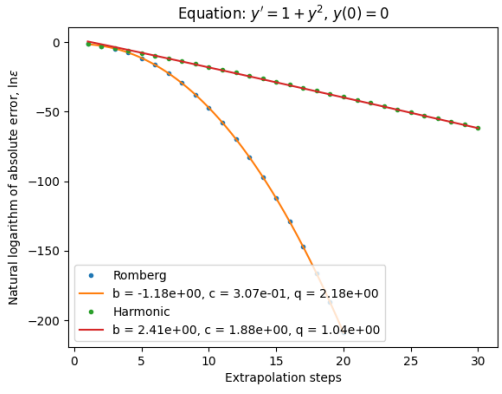
$$y(x) \coloneqq \tan(x)$$

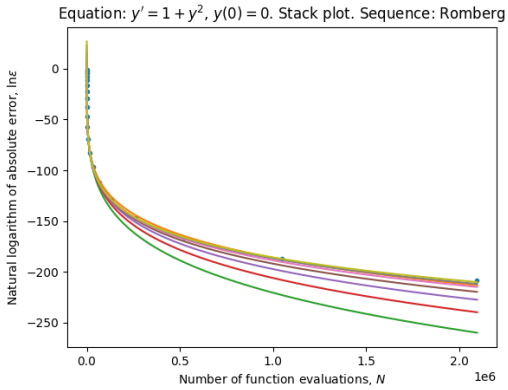
which is meromorphic and we are quite far from singularites.

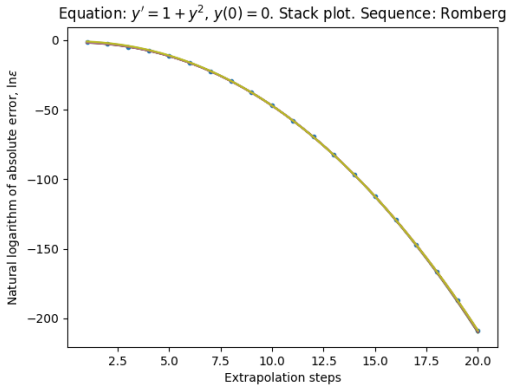


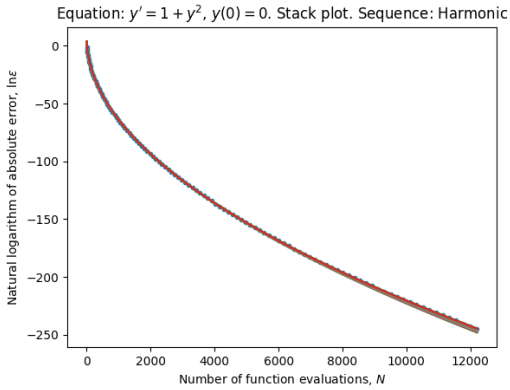


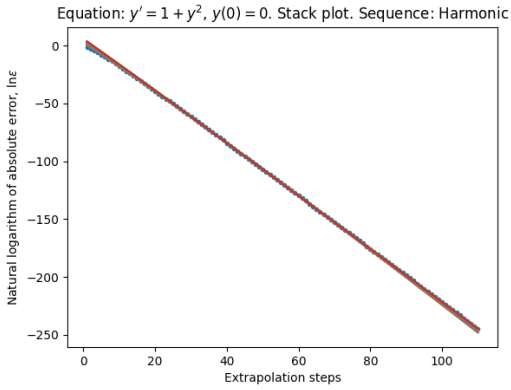












Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	$\rho_{\ln}$
RS-evals	9.5[+36]	6.0[+00]	3.3[+01]	1.7[−01]	1.5[−01]	3.6[−02]	1.1[+06]	9.5[−04]
HS-evals	3.6[+02]	5.7[−01]	2.0[+00]	2.5[−03]	5.1[−01]	1.1[−04]	2.8[+01]	5.9[−06]
RS-steps	3.5[−01]	1.1[−01]	3.0[−01]	1.0[−03]	2.2[+00]	2.6[−05]	6.7[−02]	1.3[−06]
HS-steps	1.2[+02]	5.4[−01]	2.0[+00]	2.3[−03]	1.0[+00]	1.0[−04]	2.0[+01]	5.3[−06]

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

Here we clearly have exponential convergence in the number of steps for the Romberg sequence and the fit is also very nice for the harmonic sequence.

4.2.4 The logarithm

Now we will consider the following initial value problem:

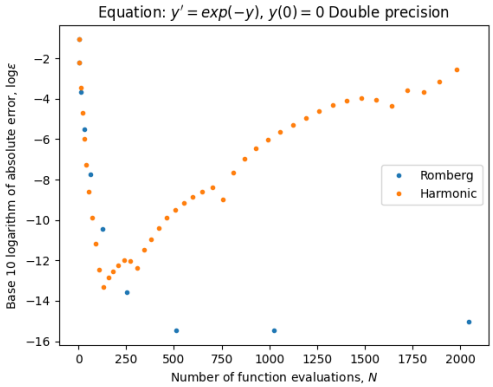
$$y'(t) = \exp(-y(t)), \quad y(0) = \ln(a), \quad t \in [0, 1]. \tag{4.5}$$

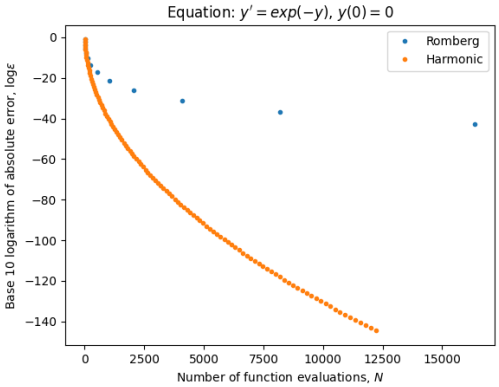
whose solution is

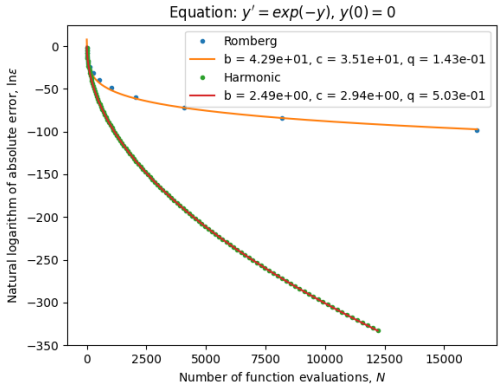
$$y(t) = \ln(a + t).$$

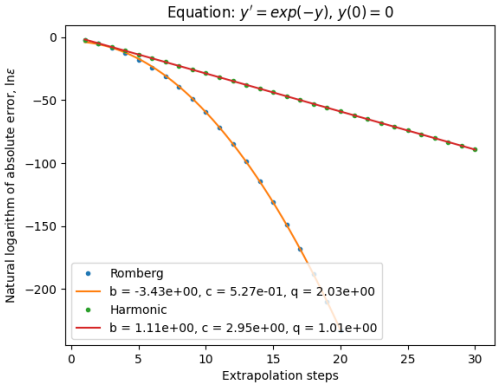
The solution is analytic on but with a singularity on the closed horizontal ray from $-a$ to $-\infty$.

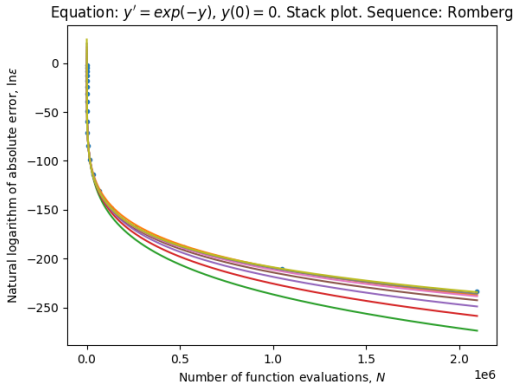
$a = 1$

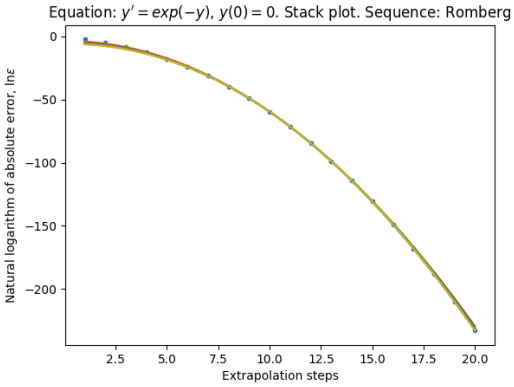

Equation: $y' = exp(-y)$, $y(0) = 0$ Double precision
Base 10 logarithm of absolute error, logε
Number of function evaluations, N
Romberg
Harmonic

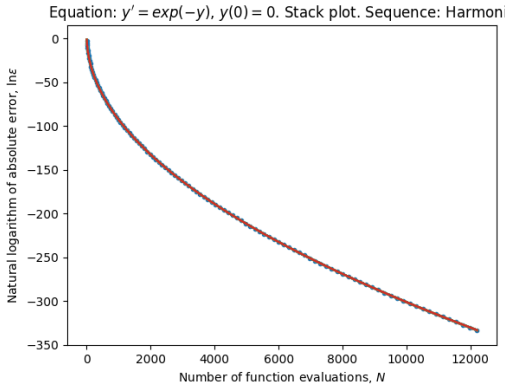

Equation: $y' = exp(-y)$, $y(0) = 0$
Base 10 logarithm of absolute error, logε
Number of function evaluations, N
Romberg
Harmonic

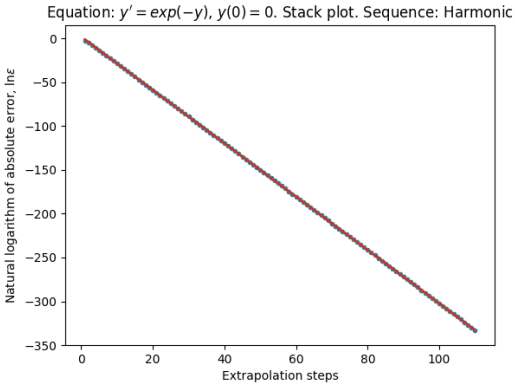

Equation: $y' = exp(-y)$, $y(0) = 0$
Natural logarithm of absolute error, lnε
Number of function evaluations, N
Romberg
b = 4.29e+01, c = 3.51e+01, q = 1.43e-01
Harmonic
b = 2.49e+00, c = 2.94e+00, q = 5.03e-01


Equation: $y' = exp(-y)$, $y(0) = 0$
Natural logarithm of absolute error, lnε
Extrapolation steps
Romberg
b = -3.43e+00, c = 5.27e-01, q = 2.03e+00
Harmonic
b = 1.11e+00, c = 2.95e+00, q = 1.01e+00


Equation: $y' = exp(-y)$, $y(0) = 0$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Equation: $y' = exp(-y)$, $y(0) = 0$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = exp(-y)$, $y(0) = 0$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N

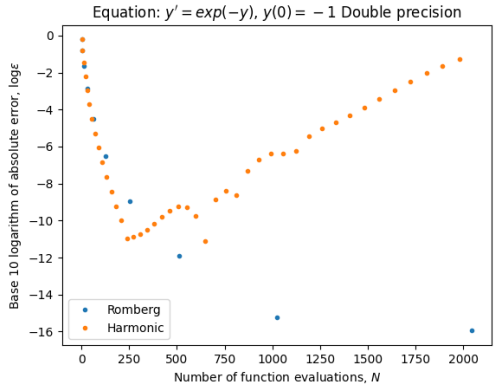

Equation: $y' = exp(-y)$, $y(0) = 0$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps

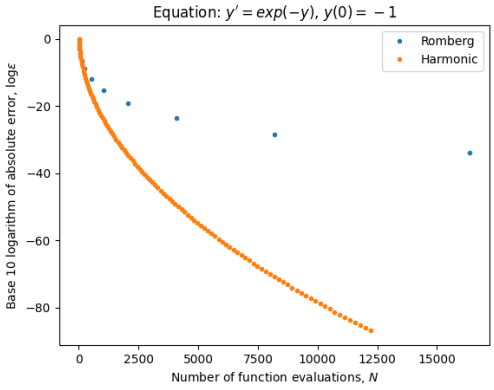
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	4.9[+42]	6.0[+00]	4.5[+01]	1.2[−01]	1.4[−01]	2.8[−02]	5.2[+05]	6.0[−04]
HS-evals	2.1[+01]	4.2[−02]	3.0[+00]	6.2[−05]	5.0[−01]	2.7[−06]	1.3[+00]	2.9[−07]
RS-steps	7.5[−03]	4.1[−01]	4.8[−01]	3.3[−03]	2.1[+00]	9.1[−05]	6.0[−01]	2.1[−05]
HS-steps	4.9[+00]	3.5[−02]	3.0[+00]	4.9[−05]	1.0[+00]	2.1[−06]	6.7[−01]	1.9[−07]

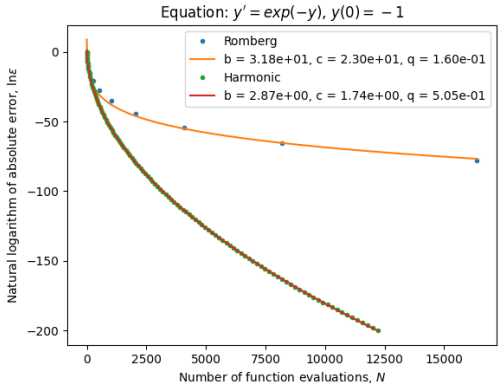
Here the harmonic sequence works better.

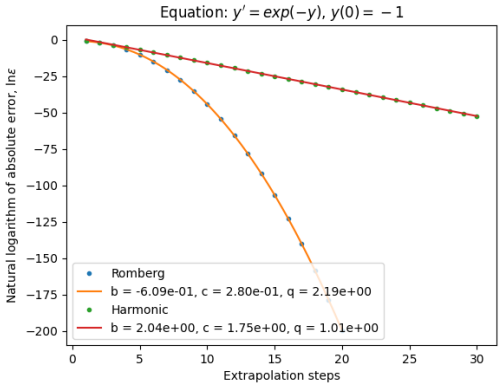
We have exponential convergence in the number of evaluations and the number of steps for the harmonic sequence and we have exponential convergence in the number of steps for the Romberg sequence.

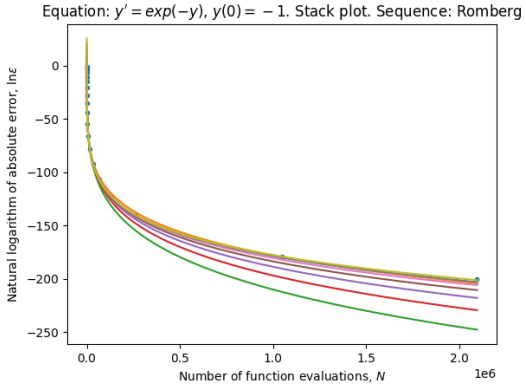
$a = e^{-1}$

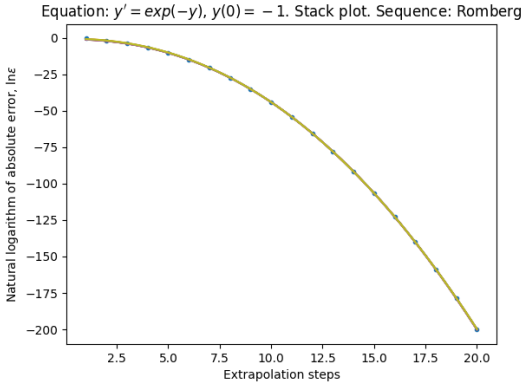


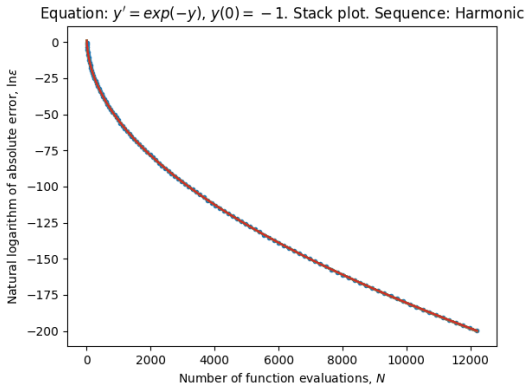


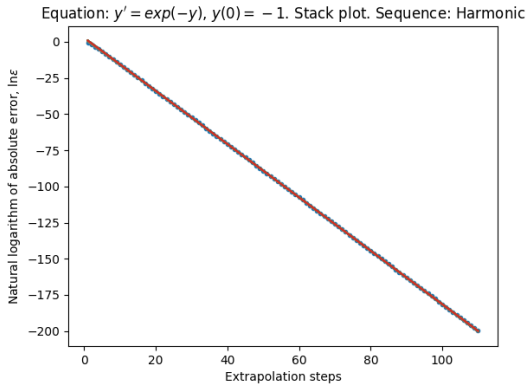








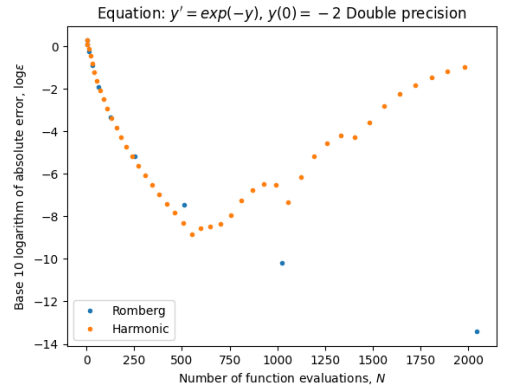


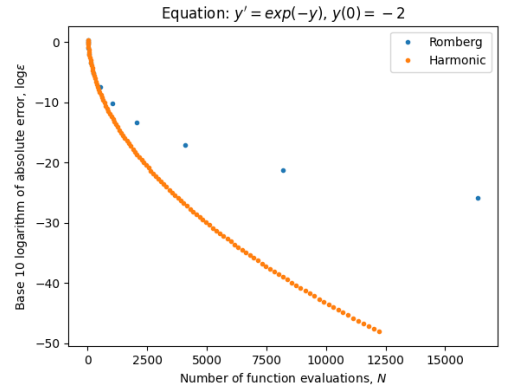


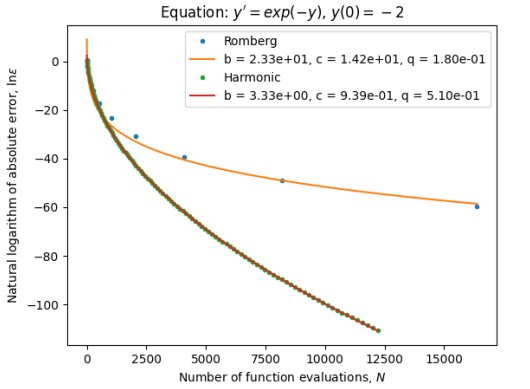
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.3[+34]	6.0[+00]	3.0[+01]	1.6[−01]	1.5[−01]	3.4[−02]	4.9[+05]	9.1[−04]
HS-evals	3.1[+01]	3.6[−02]	1.8[+00]	1.3[−04]	5.0[−01]	5.8[−06]	1.9[+00]	9.4[−07]
RS-steps	4.2[−01]	2.5[−02]	2.7[−01]	1.9[−04]	2.2[+00]	4.2[−06]	1.0[−01]	1.7[−06]
HS-steps	1.3[+01]	3.1[−02]	1.8[+00]	1.1[−04]	1.0[+00]	4.9[−06]	1.3[+00]	7.4[−07]

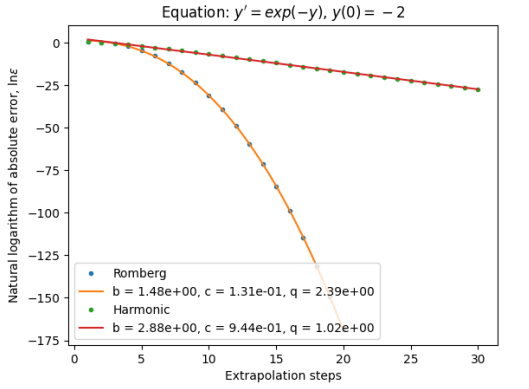
Here the same comments apply as when $a = 1$.

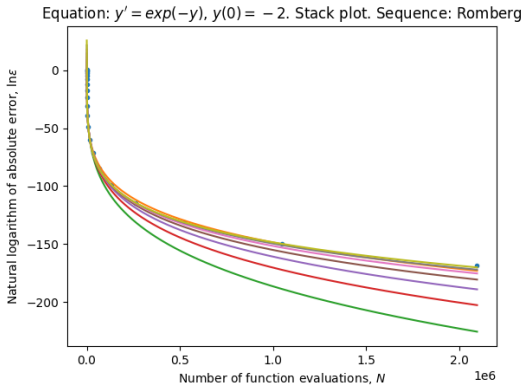
$a = e^{-2}$

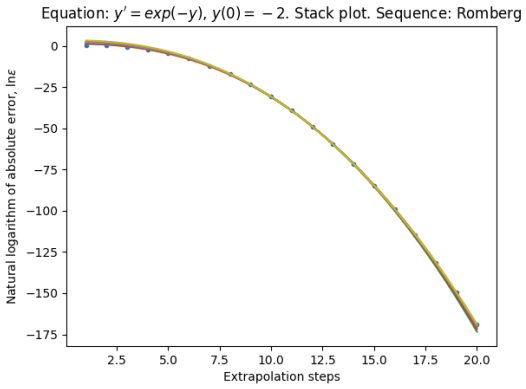


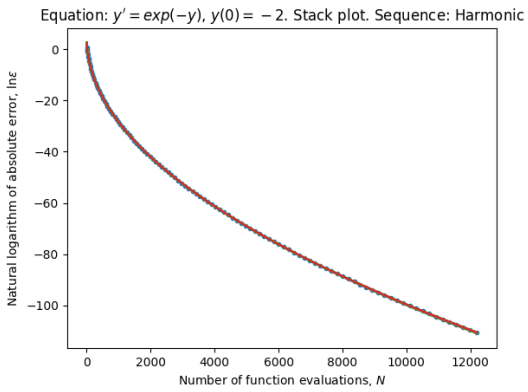


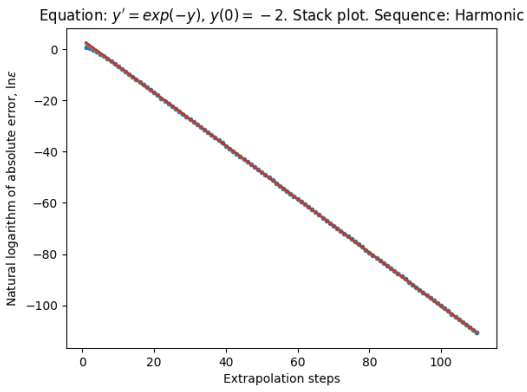








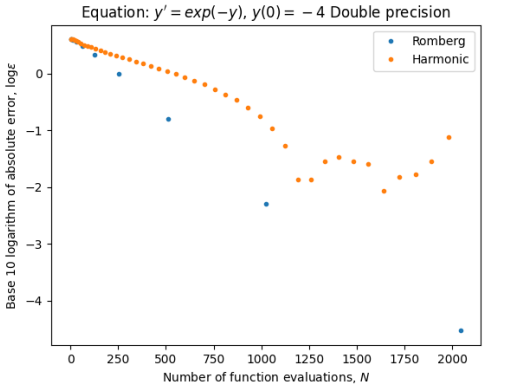


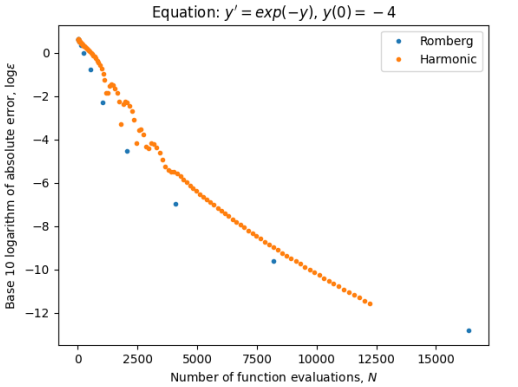


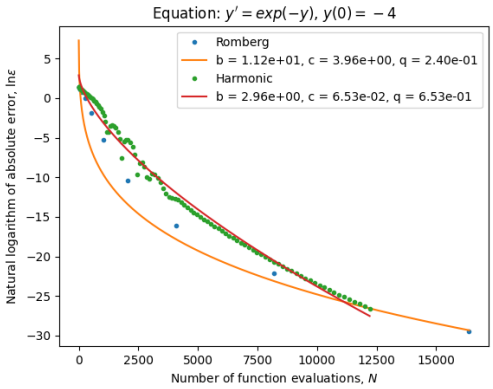
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	6.7[+26]	6.0[+00]	1.9[+01]	2.2[−01]	1.8[−01]	4.1[−02]	2.9[+05]	1.4[−03]
HS-evals	5.3[+01]	1.5[−02]	9.9[−01]	1.4[−04]	5.0[−01]	6.0[−06]	5.0[+00]	7.7[−06]
RS-steps	1.4[+01]	5.1[−01]	1.4[−01]	1.0[−02]	2.4[+00]	2.2[−04]	7.7[−01]	2.1[−05]
HS-steps	3.2[+01]	1.3[−02]	9.9[−01]	1.2[−04]	1.0[+00]	5.1[−06]	4.2[+00]	7.1[−06]

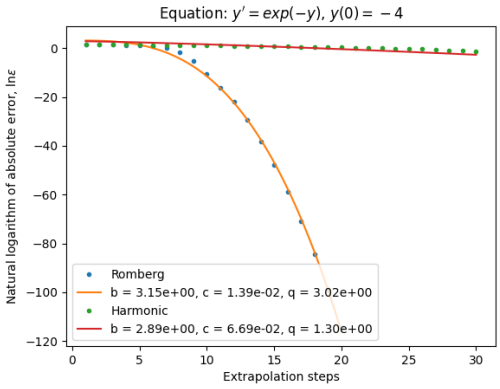
Here the same comments apply as when $a = 1$.

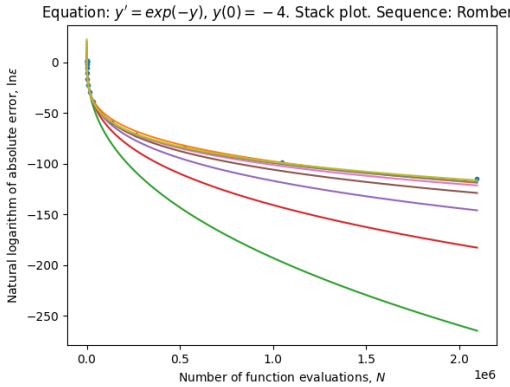
$a = e^{-4}$

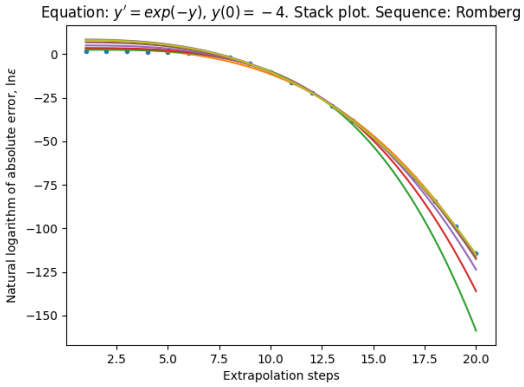


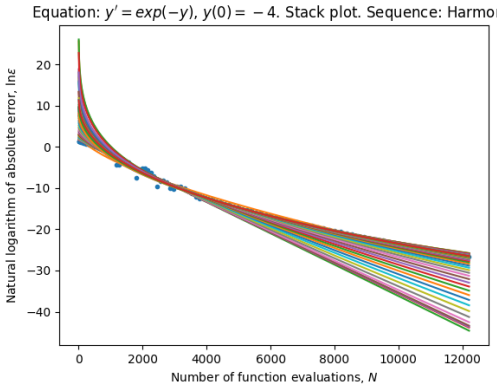


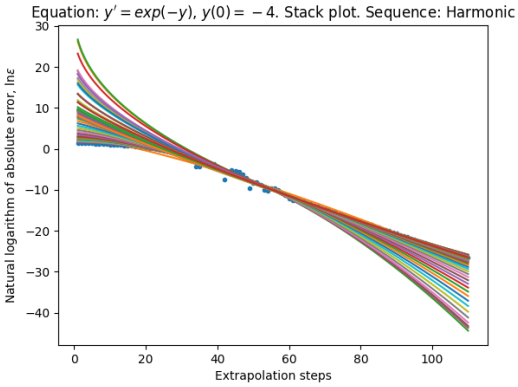








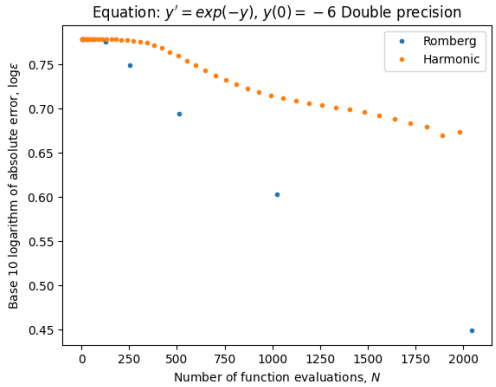


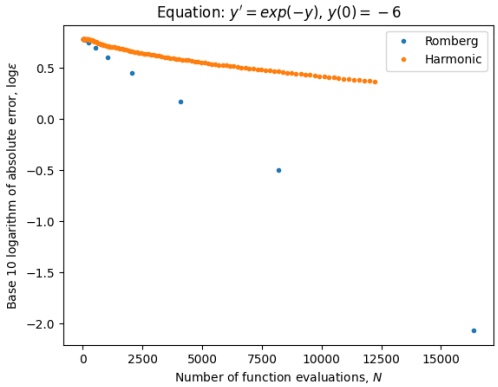


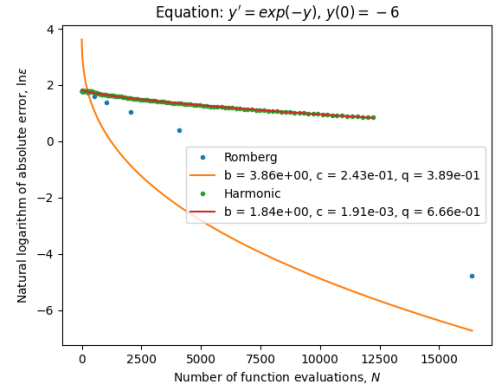
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.8[+15]	5.9[+00]	6.1[+00]	5.2[−01]	2.6[−01]	1.1[−01]	7.0[+03]	3.9[−03]
HS-evals	3.1[+13]	2.8[+01]	1.0[+00]	2.4[+00]	5.5[−01]	1.8[−01]	3.3[+00]	3.3[−03]
RS-steps	1.7[+03]	1.0[+00]	2.1[−02]	3.7[−01]	3.0[+00]	1.3[−02]	1.1[+01]	7.5[−04]
HS-steps	1.4[+13]	2.8[+01]	1.0[+00]	2.4[+00]	1.1[+00]	1.8[−01]	3.1[+00]	3.3[−03]

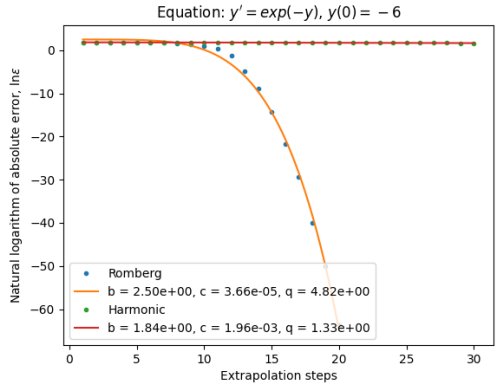
Here we get much slower convergence than in the previous cases and the model does not fit in any case.

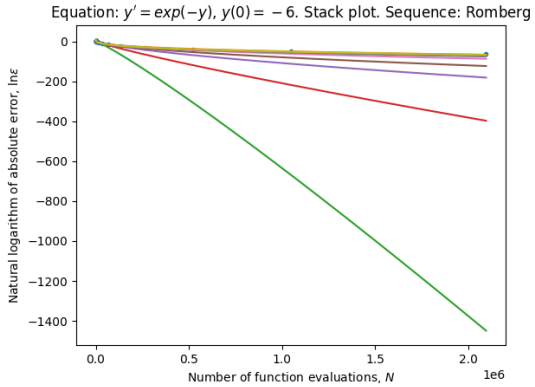
$a = e^{-6}$

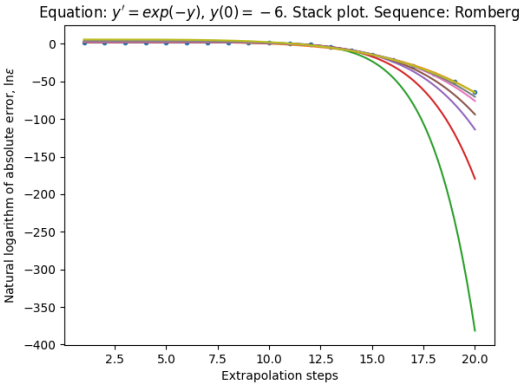

Equation: $y' = exp(-y)$, $y(0) = -6$ Double precision
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

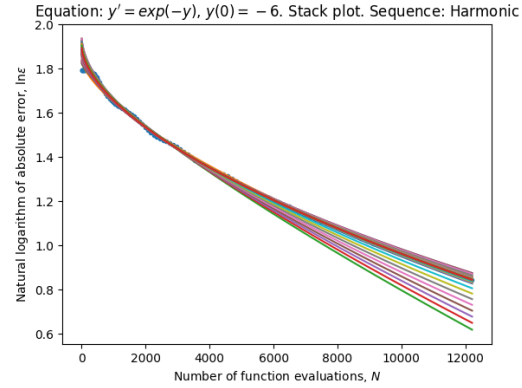

Equation: $y' = exp(-y)$, $y(0) = -6$
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

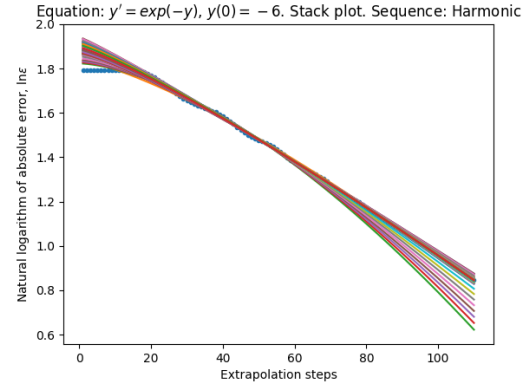

Equation: $y' = exp(-y)$, $y(0) = -6$
Romberg
b = 3.86e+00, c = 2.43e-01, q = 3.89e-01
Harmonic
b = 1.84e+00, c = 1.91e-03, q = 6.66e-01
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Equation: $y' = exp(-y)$, $y(0) = -6$
Romberg
b = 2.50e+00, c = 3.66e-05, q = 4.82e+00
Harmonic
b = 1.84e+00, c = 1.96e-03, q = 1.33e+00
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = exp(-y)$, $y(0) = -6$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Equation: $y' = exp(-y)$, $y(0) = -6$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = exp(-y)$, $y(0) = -6$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Equation: $y' = exp(-y)$, $y(0) = -6$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps

Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.1[+04]	5.8[+00]	1.7[−01]	2.5[+00]	6.3[−01]	1.7[−01]	6.6[+00]	6.1[−03]
HS-evals	6.7[+00]	9.2[−04]	4.1[−03]	1.5[−01]	6.0[−01]	9.7[−03]	3.4[−04]	1.0[−04]
RS-steps	5.4[+01]	3.3[+00]	2.9[−05]	3.8[+00]	6.4[+00]	6.7[−02]	6.6[−01]	2.1[−03]
HS-steps	6.6[+00]	8.8[−04]	4.1[−03]	1.5[−01]	1.2[+00]	9.4[−03]	3.4[−04]	1.0[−04]

Here we do simply not get convergence towards the solution and all the models fail.

4.2.5 Equation with singularity

Now we will consider the following initial value problem:

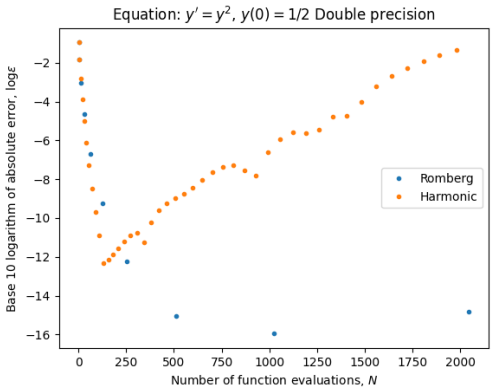
$$y'(t) = y^2(t), \quad y(0) = 1/(1+a), \quad t \in [0,1] \tag{4.6}$$

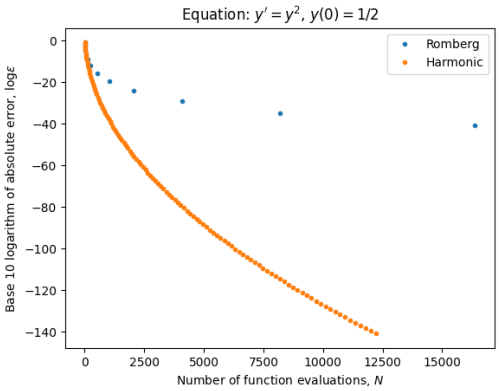
whose solution is

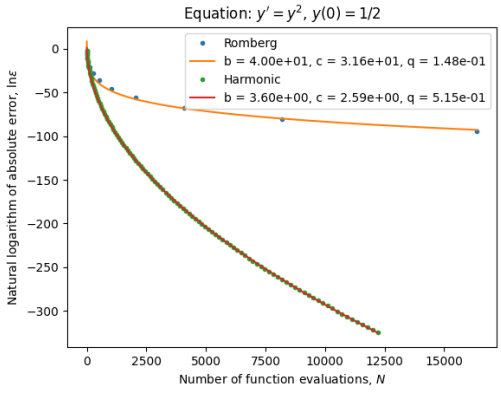
$$y(t) = \frac{1}{1-(t-a)}.$$

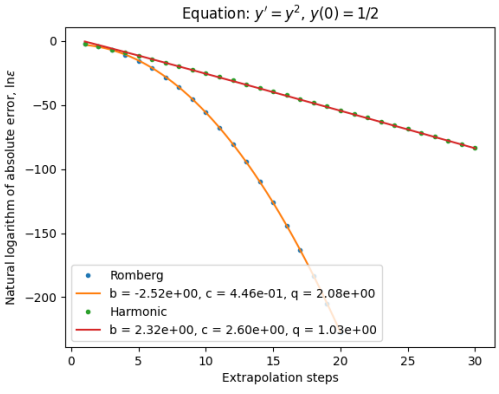
The solution is meromorphic with a pole at $1+a$.

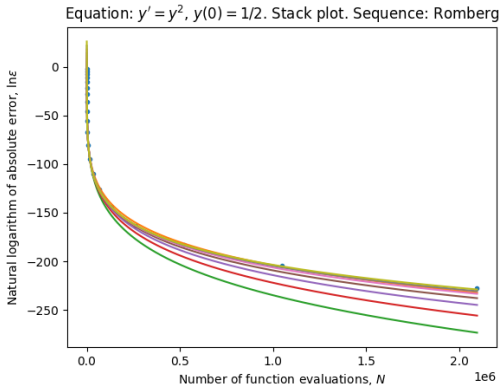
$a = 1$

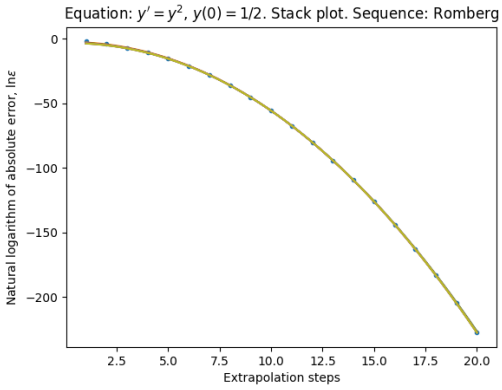


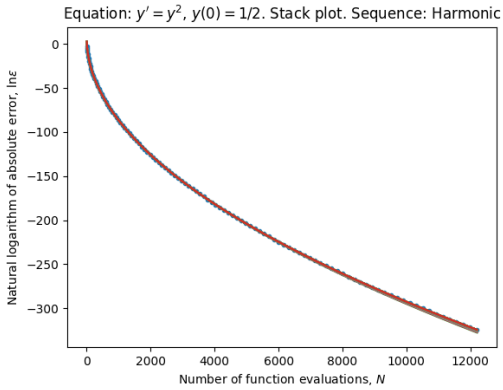


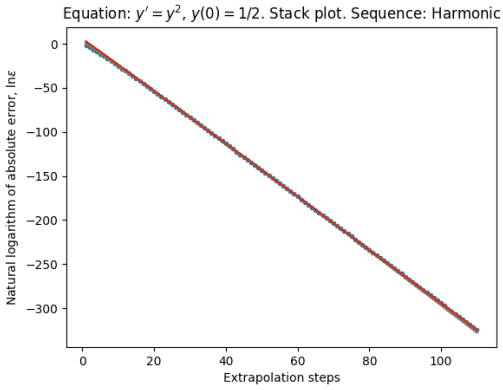










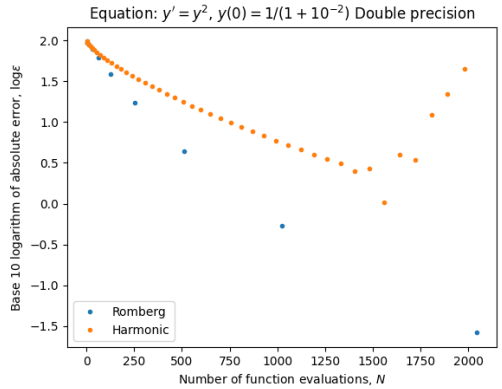


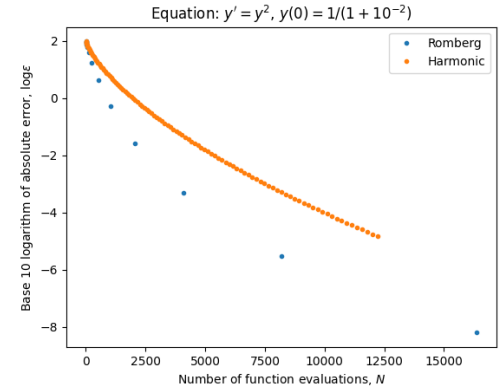
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.4[+42]	6.0[+00]	4.2[+01]	1.4[−01]	1.4[−01]	3.2[−02]	1.7[+06]	7.7[−04]
HS-evals	4.4[+02]	5.2[−01]	2.7[+00]	1.3[−03]	5.1[−01]	5.5[−05]	5.2[+01]	3.6[−06]
RS-steps	4.2[−02]	3.0[−02]	4.3[−01]	3.0[−04]	2.1[+00]	8.2[−06]	2.8[−01]	4.7[−06]
HS-steps	1.0[+02]	4.9[−01]	2.8[+00]	1.2[−03]	1.0[+00]	5.0[−05]	3.4[+01]	3.2[−06]

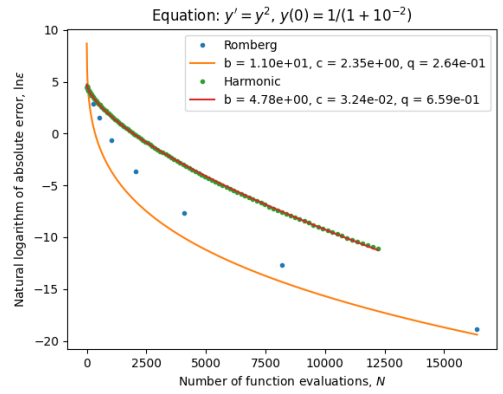
The harmonic sequence performes better and we get almost down to machine level precision using standar double precision floating point arithmetic, with either sequence.

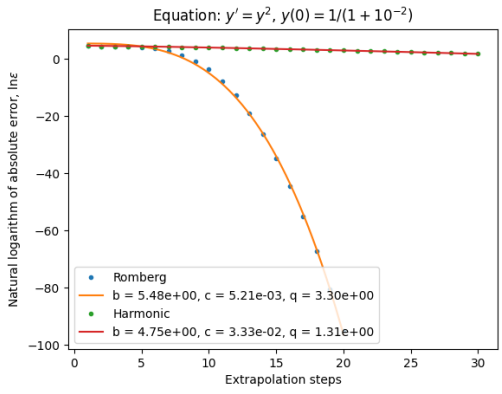
We have very nice fit for the exponential convergence in the number of steps for the Romberg sequence and also for the harmonic sequence.

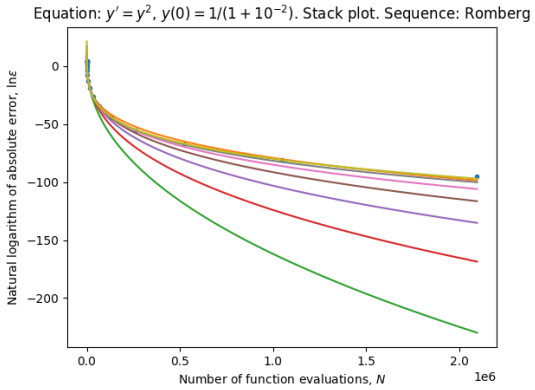
$a = 10^{-2}$

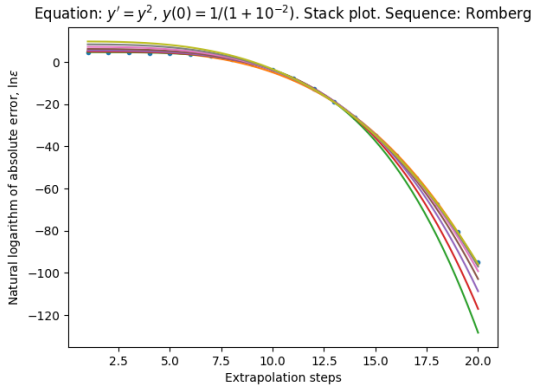


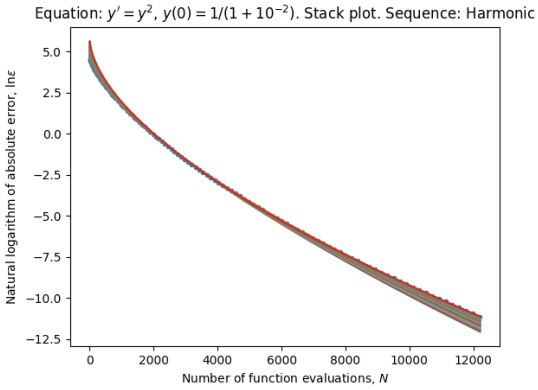


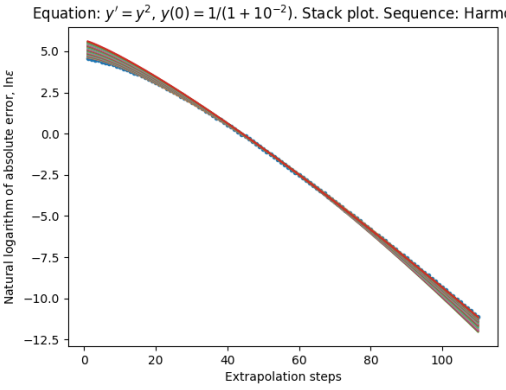










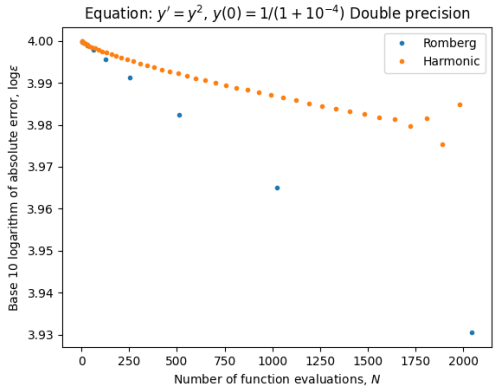


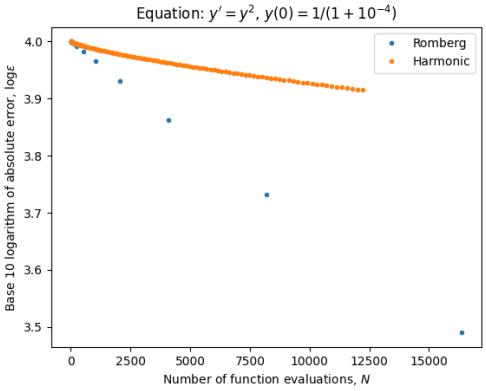
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.2[+12]	6.0[+00]	2.8[+00]	7.7[−01]	3.1[−01]	8.3[−02]	4.0[+02]	3.6[−03]
HS-evals	1.7[+02]	1.2[−01]	4.0[−02]	8.6[−02]	6.5[−01]	2.4[−03]	9.8[−03]	1.0[−04]
RS-steps	3.5[+03]	2.9[+00]	5.2[−03]	6.3[−01]	3.5[+00]	9.4[−03]	1.5[+00]	5.0[−04]
HS-steps	1.7[+02]	1.1[−01]	4.0[−02]	8.2[−02]	1.3[+00]	2.2[−03]	8.5[−03]	9.5[−05]

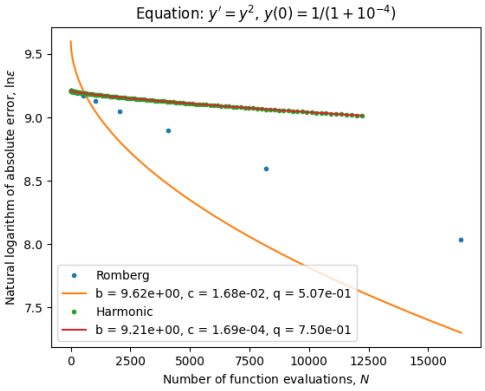
Here Romberg works better and we do not attain as high precisison using the harmonic as when using Romberg, in standard double precision arithmetic.

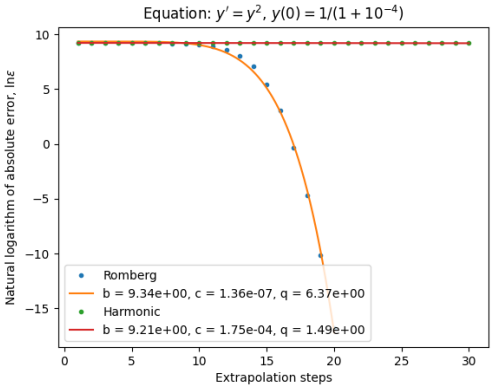
Here the model fits moderately well for Romberg sequence, when considering exponential convergence in the number of steps. We also have moderate fit for the harmonic sequence.

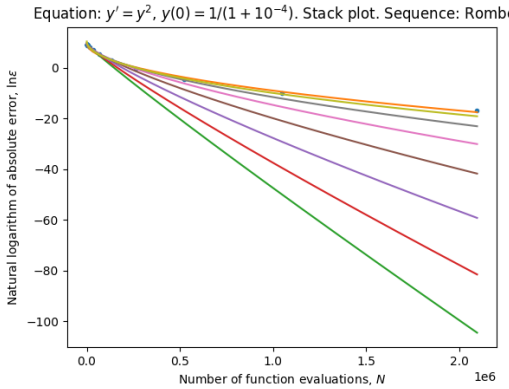
$a = 10^{-4}$

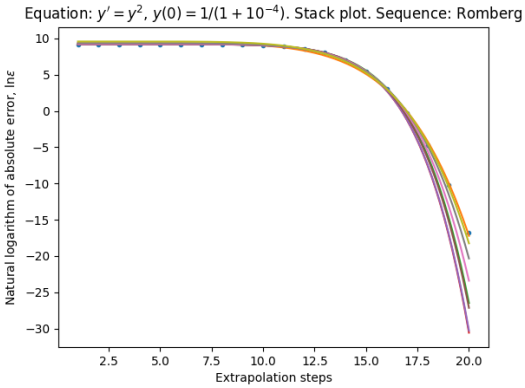

Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$ Double precision
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

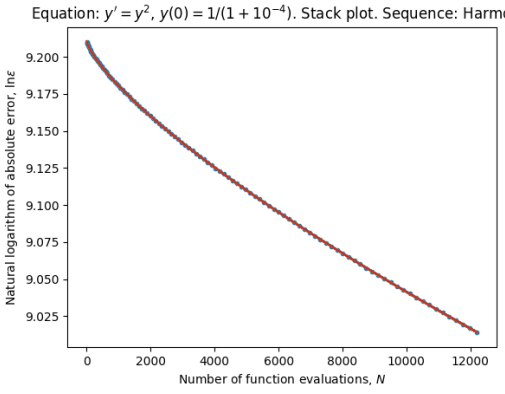

Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

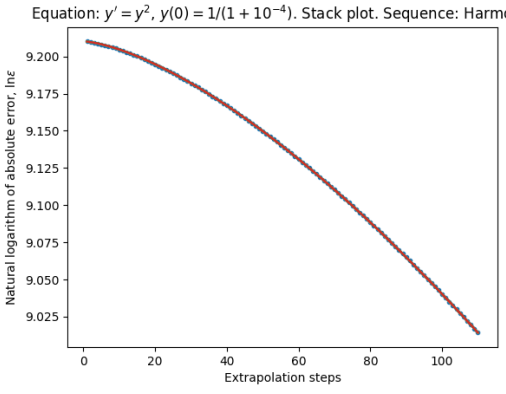

Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$
Natural logarithm of absolute error, lnε
Romberg
b = 9.62e+00, c = 1.68e-02, q = 5.07e-01
Harmonic
b = 9.21e+00, c = 1.69e-04, q = 7.50e-01
Number of function evaluations, N


Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$
Natural logarithm of absolute error, lnε
Romberg
b = 9.34e+00, c = 1.36e-07, q = 6.37e+00
Harmonic
b = 9.21e+00, c = 1.75e-04, q = 1.49e+00
Extrapolation steps


Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Equation: $y' = y^2$, $y(0) = 1/(1 + 10^{-4})$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps

Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.5[+04]	2.3[−01]	4.3[−03]	2.3[+00]	7.4[−01]	4.2[−02]	1.1[−01]	3.0[−03]
HS-evals	1.0[+04]	5.5[−09]	1.7[−04]	5.8[−05]	7.5[−01]	1.2[−06]	4.2[−10]	4.8[−12]
RS-steps	1.1[+04]	2.2[−02]	2.0[−08]	3.2[+00]	7.6[+00]	6.3[−03]	1.8[−02]	8.8[−04]
HS-steps	1.0[+04]	3.3[−10]	1.7[−04]	5.3[−06]	1.5[+00]	1.2[−07]	2.1[−10]	2.2[−12]

Here we clearly do not have exponential convergence in the number of evaluations for the Romberg sequence. We have not so good fit for exponential convergence in the number of steps.

Regarding the harmonic sequence, we note that we have extremely slow convergence, so the x and y values differ by many orders of magnitude, so the results are maybe not so reliable.

4.2.6 Equation with moderate singularity

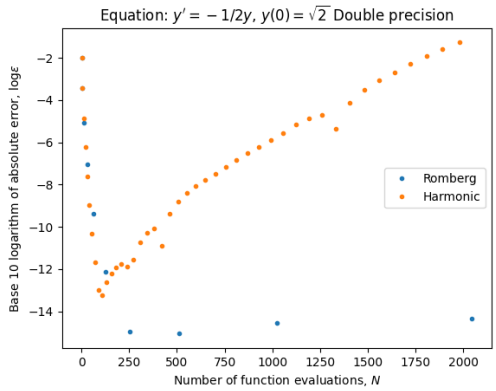
Now we will consider the following initial value problem

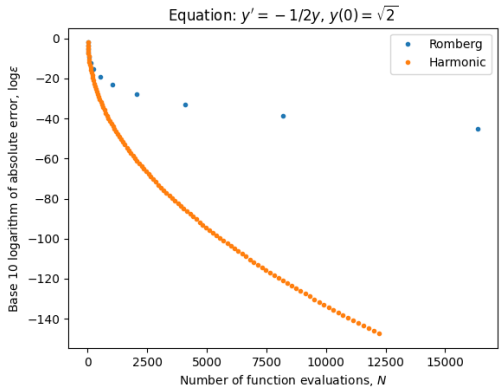
$$y'(t) = -\frac{1}{2y}, \quad y(0) = \sqrt{1+a}, \quad t \in [0,1] \tag{4.7}$$

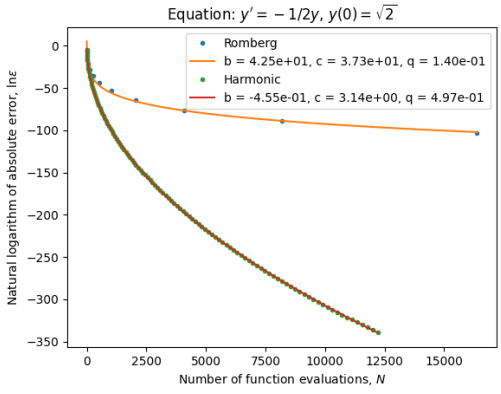
whose solution is

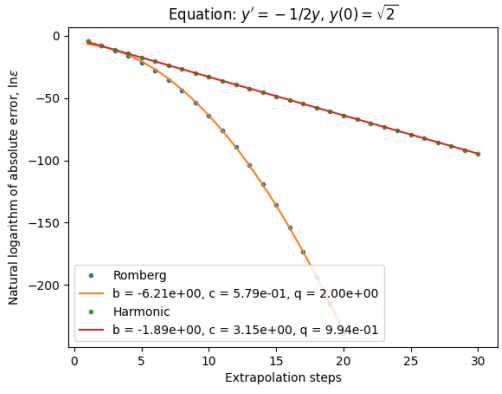
$$y(t) = \sqrt{1-(t-a)}$$

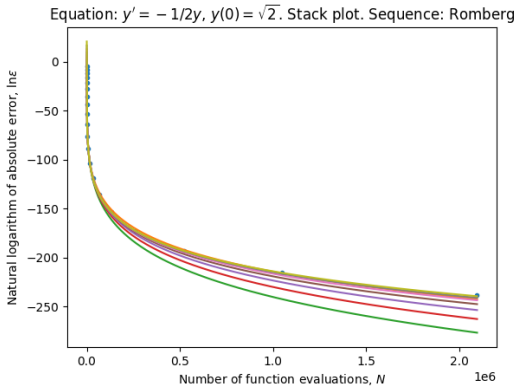
$a = 1$

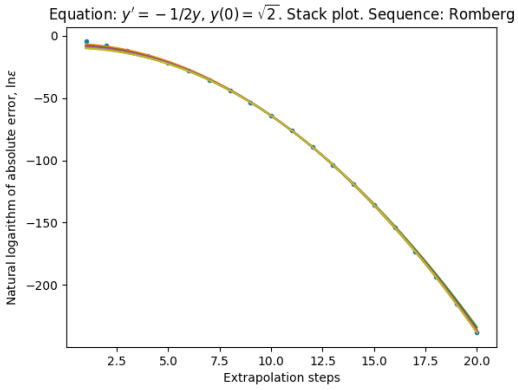


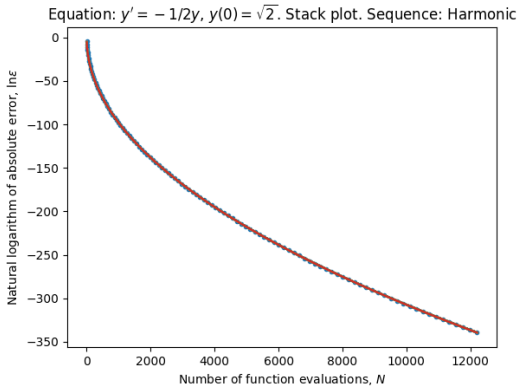


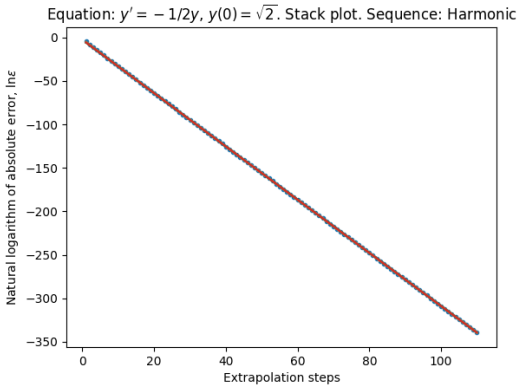










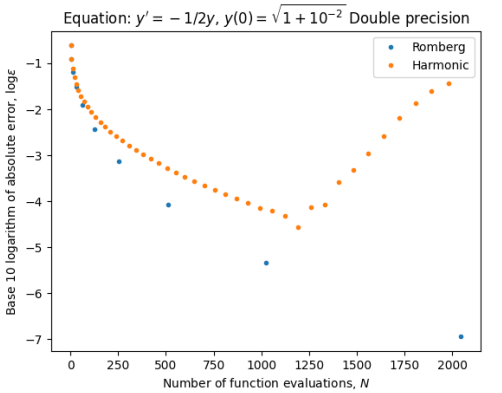


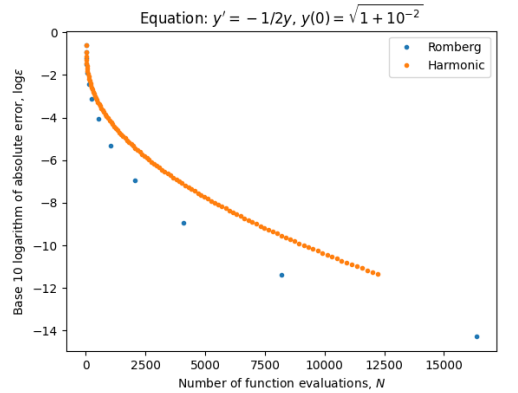
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	8.3[+41]	6.0[+00]	4.7[+01]	1.1[−01]	1.3[−01]	2.6[−02]	1.5[+05]	4.9[−04]
HS-evals	4.5[−01]	3.6[−02]	3.1[+00]	3.8[−05]	5.0[−01]	1.6[−06]	6.8[−02]	7.0[−08]
RS-steps	2.8[−04]	7.4[−01]	5.1[−01]	5.8[−03]	2.0[+00]	1.6[−04]	7.9[−01]	4.0[−05]
HS-steps	1.0[−01]	4.1[−02]	3.1[+00]	4.4[−05]	1.0[+00]	1.9[−06]	1.3[−01]	1.2[−07]

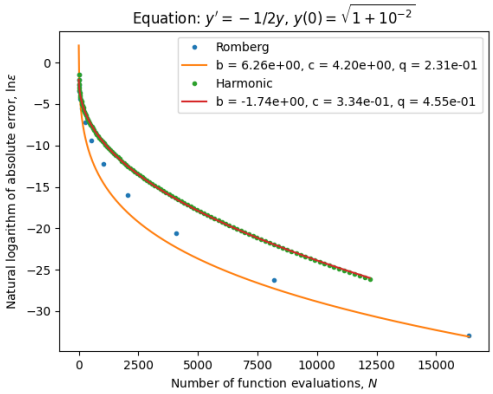
Here the harmonic sequence works better than Romberg and we get down to machine level precision using either sequence, in standard double precision floating point arithmetic.

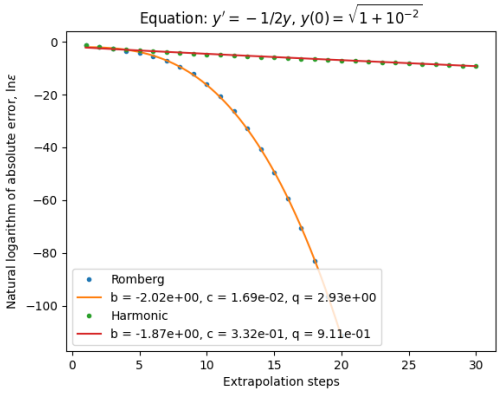
Here we clearly have exponential convergence in the number of steps and evaluations for the harmonic sequence. Regarding Romberg, we seem to have exponential convergence in the number of steps.

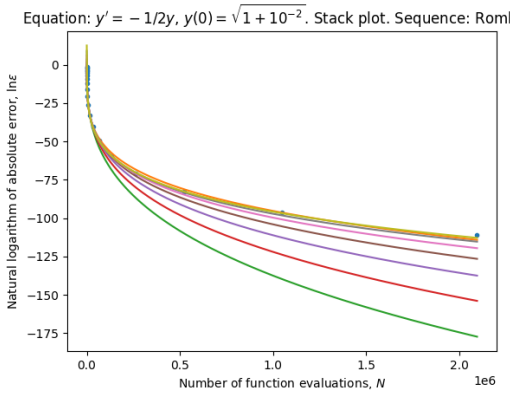
$a = 10^{-2}$

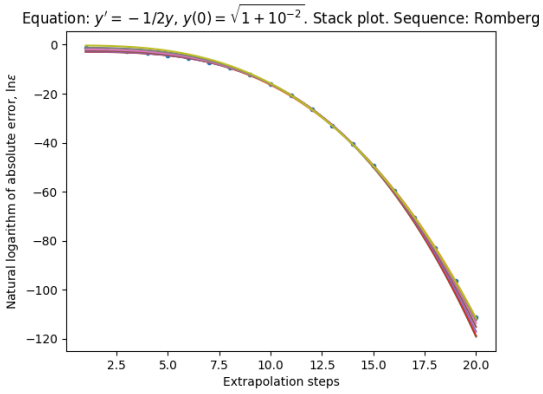


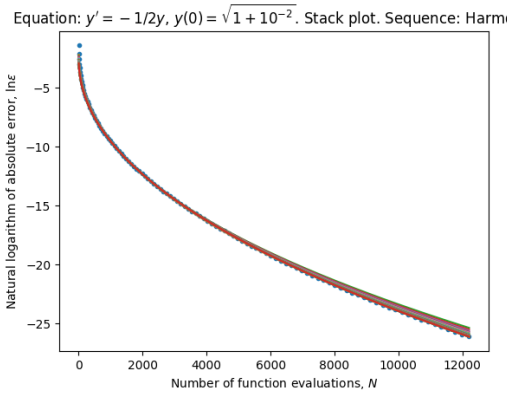


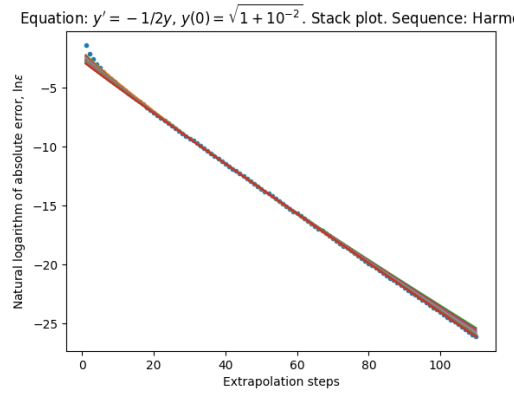










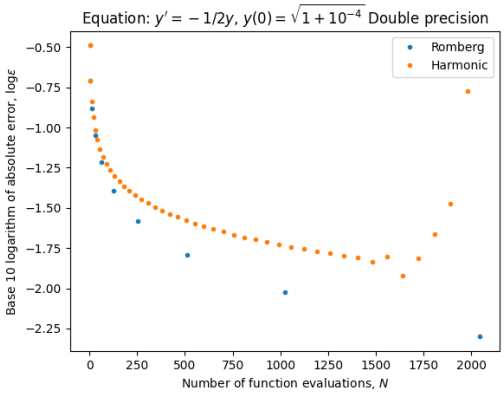


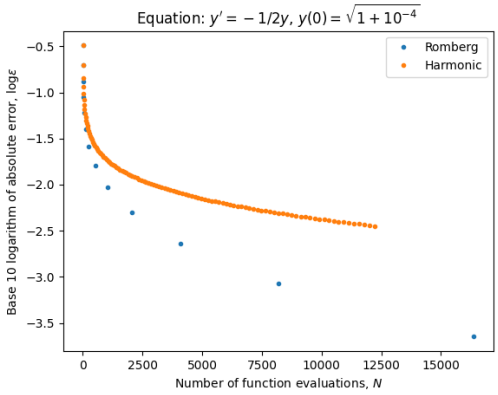
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	3.1[+09]	6.0[+00]	4.5[+00]	4.3[−01]	2.5[−01]	4.6[−02]	1.7[+02]	1.4[−03]
HS-evals	9.3[−02]	5.5[−02]	2.6[−01]	1.1[−02]	4.8[−01]	4.5[−04]	2.3[−01]	5.1[−05]
RS-steps	2.1[−01]	1.2[+00]	1.4[−02]	1.1[−01]	3.0[+00]	1.3[−03]	1.7[−01]	5.7[−05]
HS-steps	8.3[−02]	5.0[−02]	2.6[−01]	1.1[−02]	9.6[−01]	4.4[−04]	2.3[−01]	5.1[−05]

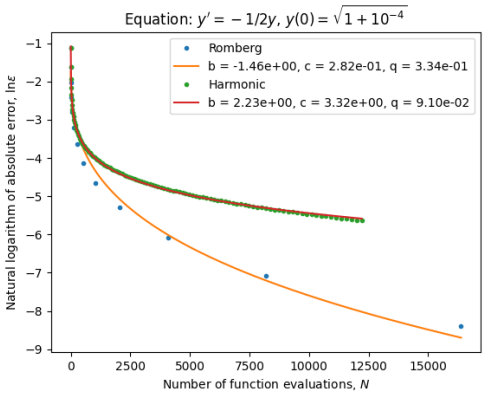
Here Romberg performes better and we do not attain as high precision using the harmonic sequence in standard double precision floating point arithmetic.

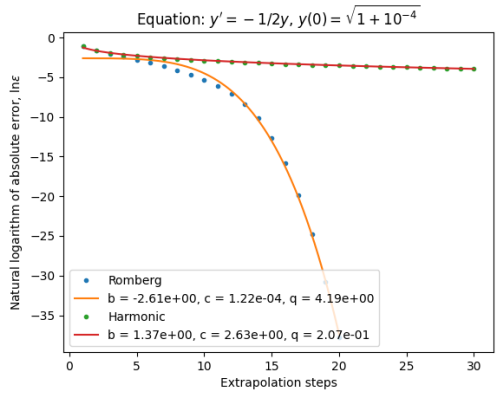
For the Romberg sequence, the model seems to fit moderately well when considering exponential convergence in the number of evaluations. We also have reasonably good fit for the harmonic sequence.

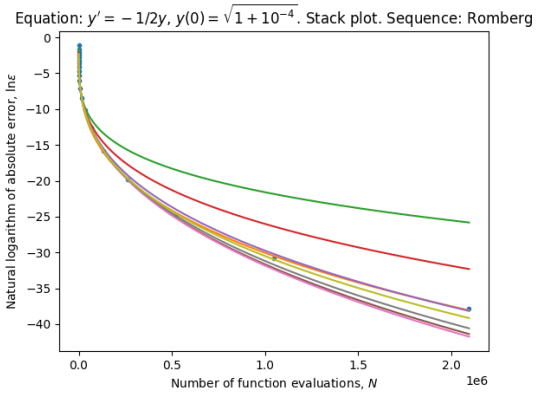
$a = 10^{-4}$

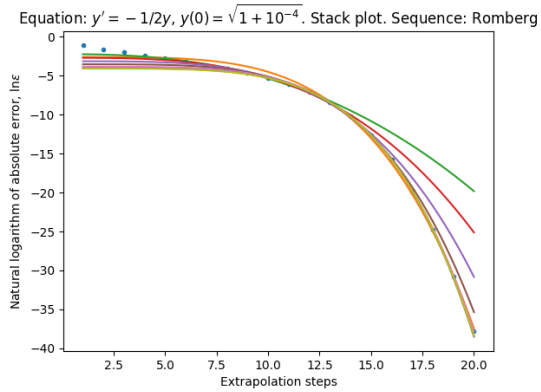

Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$ Double precision
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

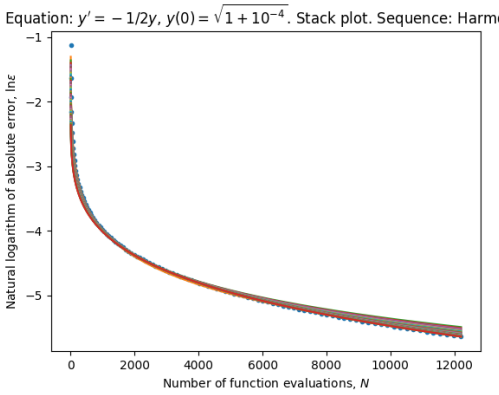

Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$
Romberg
Harmonic
Base 10 logarithm of absolute error, logε
Number of function evaluations, N

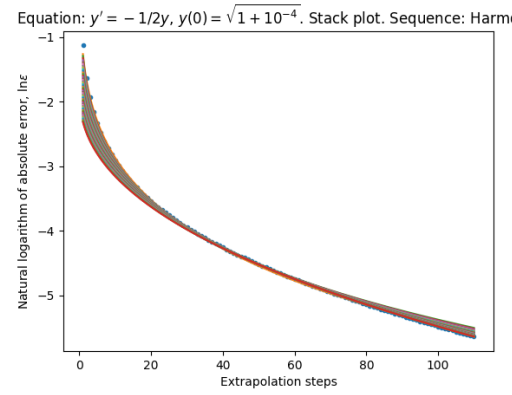

Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$
Romberg
b = -1.46e+00, c = 2.82e-01, q = 3.34e-01
Harmonic
b = 2.23e+00, c = 3.32e+00, q = 9.10e-02
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$
Romberg
b = -2.61e+00, c = 1.22e-04, q = 4.19e+00
Harmonic
b = 1.37e+00, c = 2.63e+00, q = 2.07e-01
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Number of function evaluations, N
1e6


Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$. Stack plot. Sequence: Romberg
Natural logarithm of absolute error, lnε
Extrapolation steps


Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Number of function evaluations, N


Equation: $y' = -1/2y$, $y(0) = \sqrt{1+10^{-4}}$. Stack plot. Sequence: Harmonic
Natural logarithm of absolute error, lnε
Extrapolation steps

Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.4[−01]	4.8[−01]	2.5[−01]	4.6[−01]	3.5[−01]	1.9[−02]	1.7[−01]	4.5[−04]
HS-evals	1.3[+00]	3.4[+00]	1.3[+00]	3.9[−01]	1.6[−01]	5.5[−02]	1.2[−02]	4.9[−05]
RS-steps	4.3[−02]	5.1[−01]	1.8[−03]	3.8[+00]	4.0[+00]	4.4[−02]	4.7[−01]	1.8[−03]
HS-steps	8.1[−01]	1.7[+00]	1.1[+00]	3.1[−01]	3.3[−01]	4.6[−02]	7.4[−03]	3.5[−05]

Here, we do not have any clear fit.

4.2.7 Circular rotation

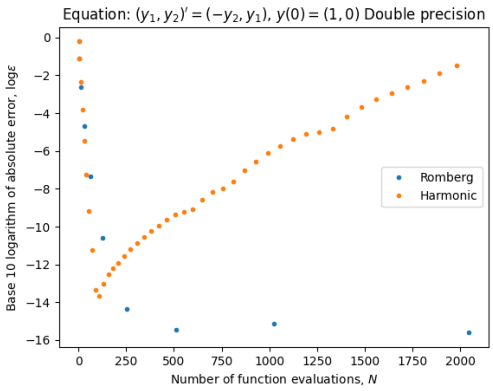
Now we will consider the following system of equations:

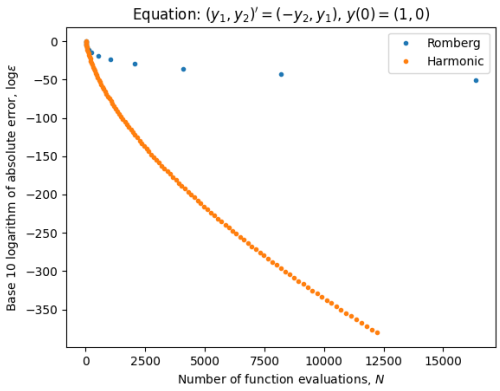
$$(y_1(t), y_2(t))' = (-y_2(t), y_1(t)), \quad y(0) = (1, 0), \quad t \in [0, \pi/2] \tag{4.8}$$

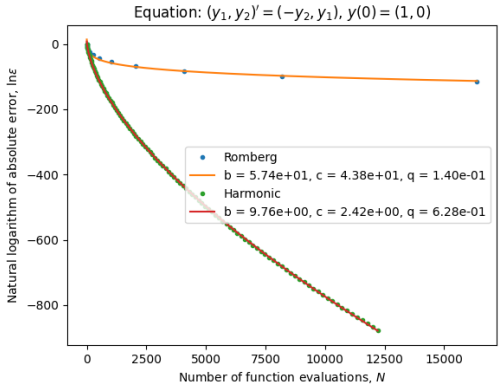
whose solution is

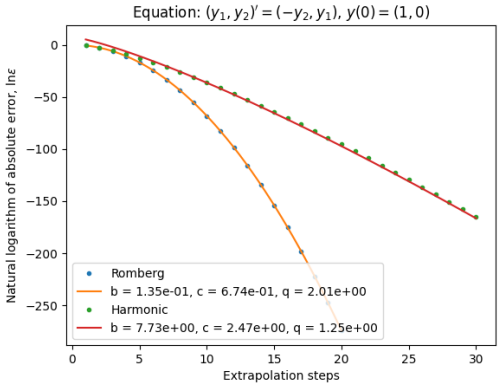
$$(y_1(t), y_2(t)) = (\cos t, \sin t)$$

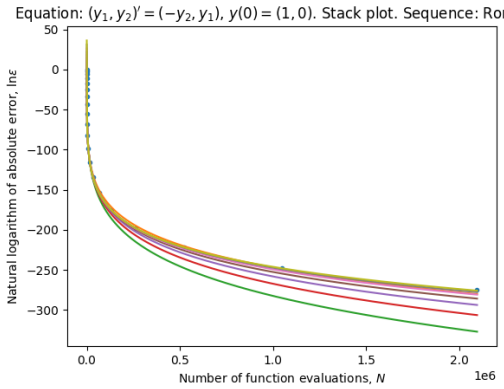
which is entire.

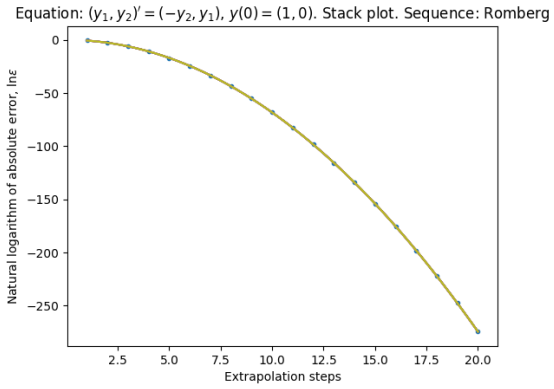


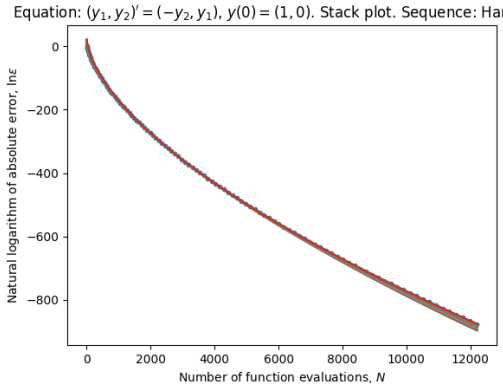


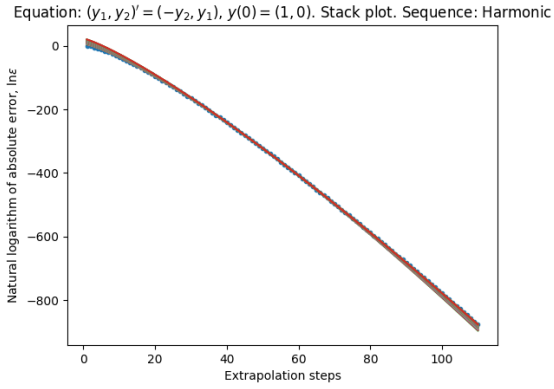












Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.2[+60]	6.0[+00]	6.1[+01]	1.4[−01]	1.3[−01]	3.5[−02]	2.3[+08]	8.6[−04]
HS-evals	9.0[+09]	6.8[+00]	2.6[+00]	7.9[−03]	6.2[−01]	2.5[−04]	4.4[+05]	8.9[−06]
RS-steps	1.1[+00]	1.0[−04]	6.7[−01]	5.0[−07]	2.0[+00]	1.5[−08]	2.9[−03]	2.0[−08]
HS-steps	4.3[+08]	6.5[+00]	2.7[+00]	6.9[−03]	1.2[+00]	2.2[−04]	9.5[+04]	7.5[−06]

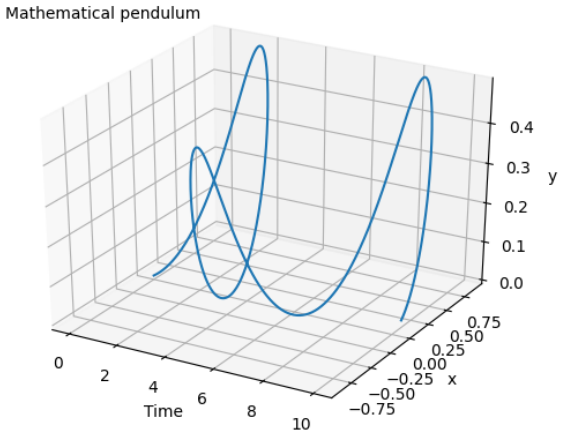
The harmonic sequence works better then Romberg and we get down to machine level precision using either sequence when using standard floating point arithmetic.

We clearly have exponential convergence in the number of steps for the Romberg sequence. For the harmonic sequence, we seem to have exponential convergence, but though we must note that the mean value of the A coefficient is quite large.

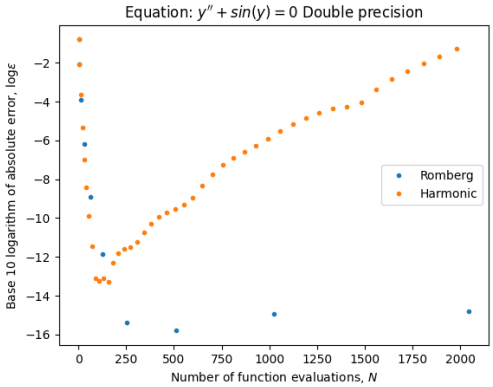
4.2.8 Mathematical pendulum

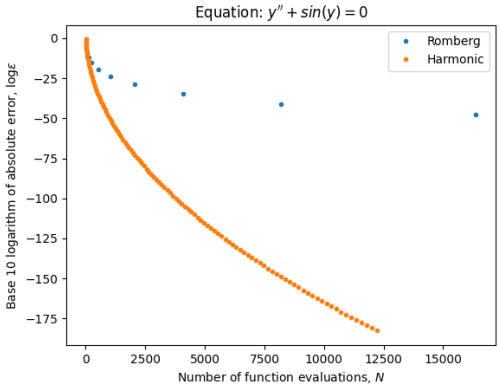
Now we will consider the mathematical pendulum equation:

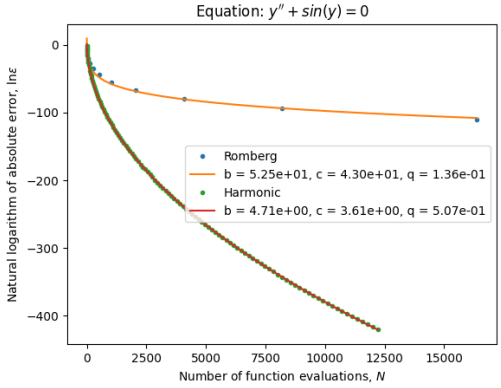
$$y''(t) + \sin y(t) = 0, \quad y(0) = 0,\ y'(0) = 1, \quad t \in [0, 1]. \tag{4.9}$$

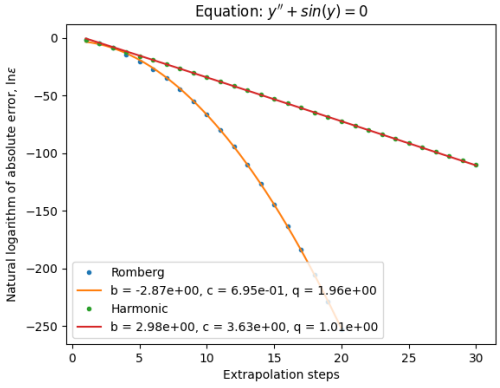


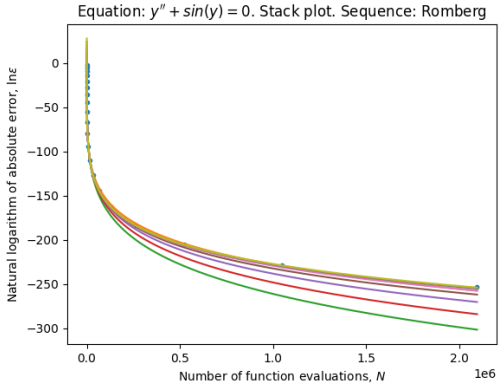
Here we computed a reference solution up to 500 digits which can be found in the code.

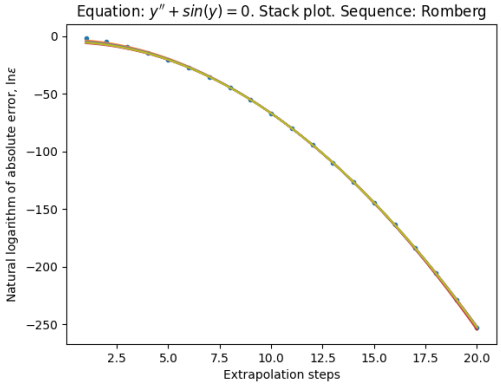


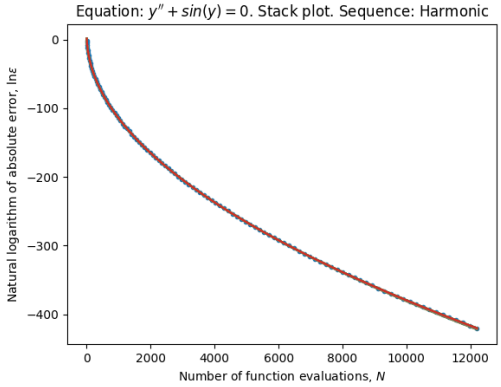


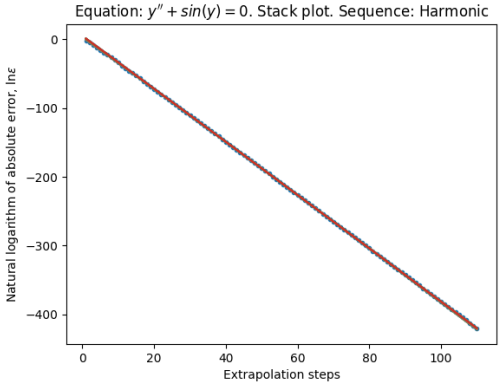












Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	3.2[+52]	6.0[+00]	5.7[+01]	1.4[−01]	1.3[−01]	3.9[−02]	1.4[+06]	5.6[−04]
HS-evals	4.7[+0]	2.7[−01]	3.7[+00]	2.6[−04]	5.0[−01]	1.1[−05]	9.6[+00]	1.1[−06]
RS-steps	1.4[−02]	3.3[−01]	6.3[−01]	2.1[−03]	2.0[+00]	6.2[−05]	6.7[−01]	2.8[−05]
HS-steps	7.3[+01]	2.5[−01]	3.7[+00]	2.3[−04]	1.0[+00]	9.9[−06]	5.2[+00]	9.2[−07]

Here the harmonic sequence works better and we get down to machine level precision in standard double precision floating point arithmetic, using either sequence.

We have nice fit for the exponential convergence in the number of steps for the Romberg sequence. We also have very nice fit for the harmonic sequence.

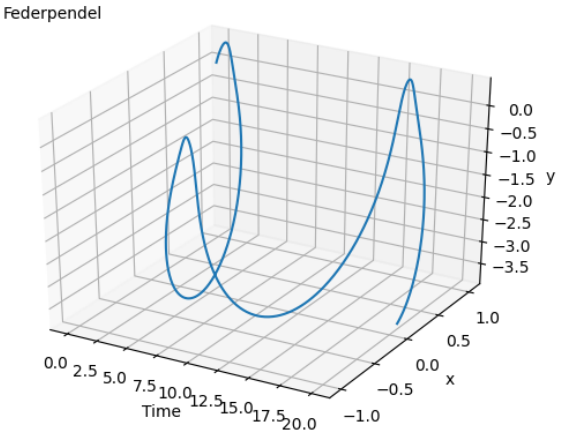
4.2.9 Federpendel

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

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

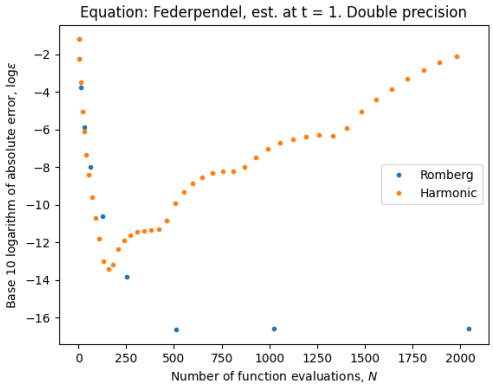
where $\mathbf{p}$ and $\mathbf{q}$ are two dimensional vectors. We will consider it with the initial condition $\mathbf{q}(0) = (1, 0)$ and $\mathbf{p}(0) = (0, 1)$ and try to both estimate the solution at times $t = 1$, $t = 2$ and $t = 10$.

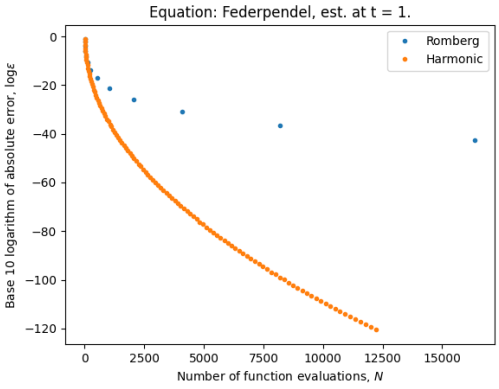
In the first two cases we use precomputed solutions with accurracy of 500 digits and in the third case we use a solution with accurracy of 250 digits. The solutions can be found in the code.

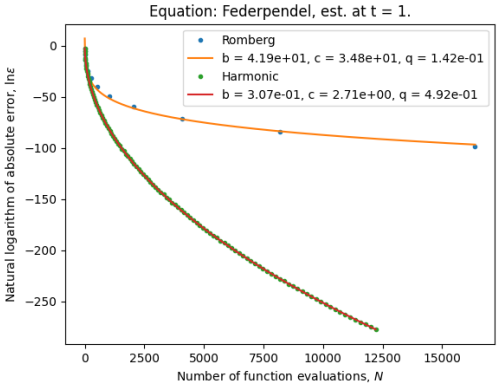


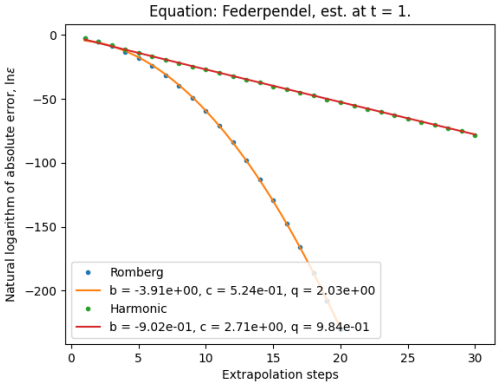
$t = 1$

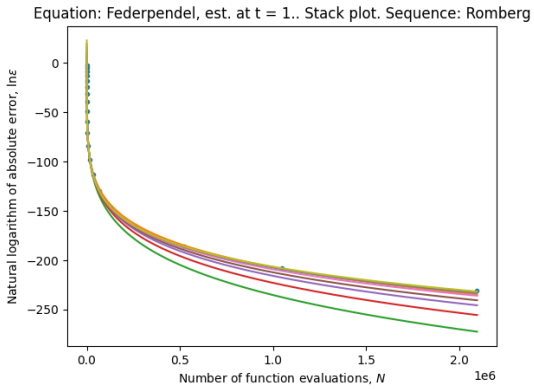
Here we use a reference solution which was computed

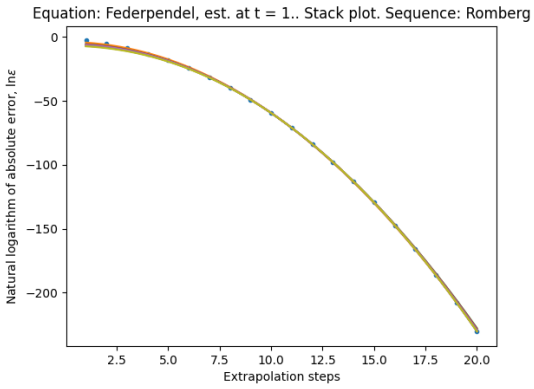


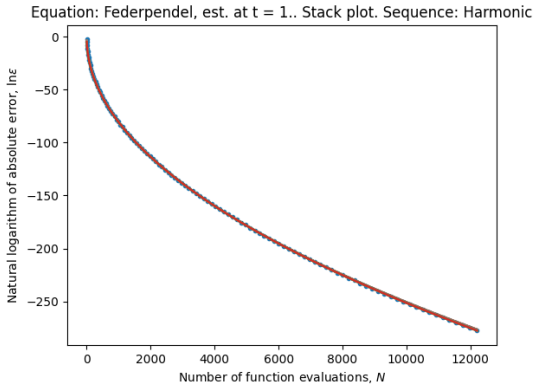


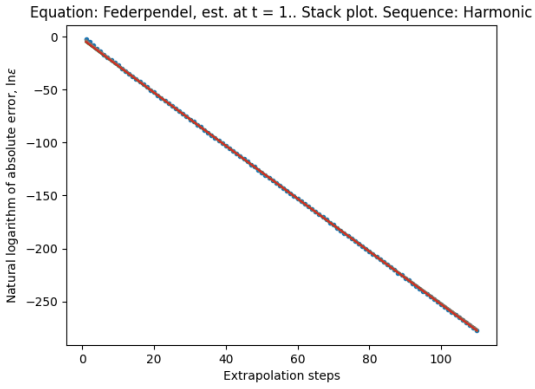












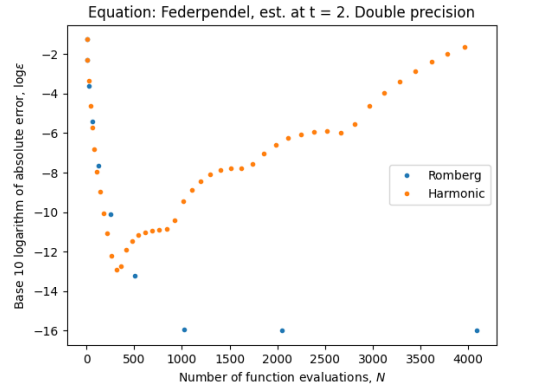
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	8.2[+40]	6.0[+00]	4.4[+01]	1.2[−01]	1.4[−01]	2.8[−02]	3.1[+05]	5.4[−04]
HS-evals	5.1[−01]	2.5[−01]	2.6[+00]	3.4[−04]	5.0[−01]	1.5[−05]	2.7[−01]	1.8[−06]
RS-steps	2.8[−03]	3.7[−01]	4.6[−01]	3.3[−03]	2.1[+00]	8.9[−05]	6.5[−01]	3.7[−05]
HS-steps	1.5[−01]	2.5[−01]	2.6[+00]	3.5[−04]	9.9[−01]	1.5[−05]	3.2[−01]	2.0[−06]

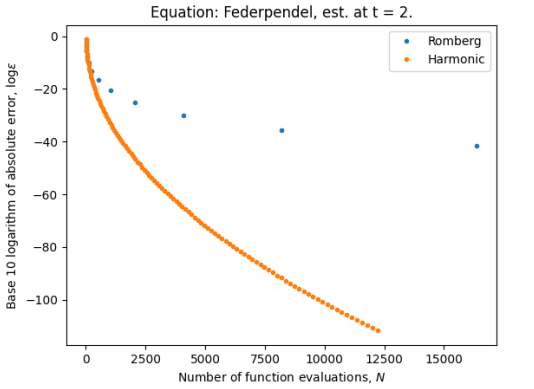
Here the harmonic sequence works better and we get down to machine level precision in standard double precision floating point arithmetic, using either sequence.

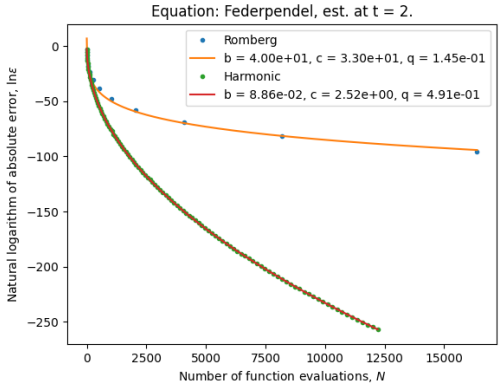
We have nice fit for exponential convergence in the number of steps for the Romberg sequence. We also have nice fit for exponential convergence for the harmonic sequence.

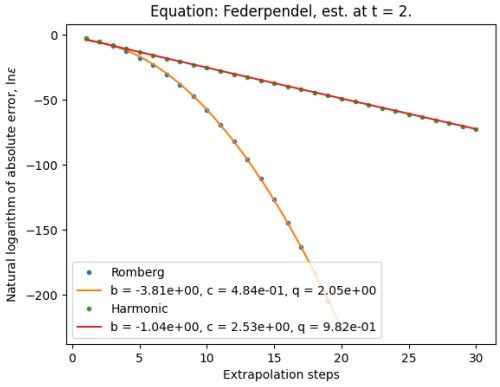
$t = 2$

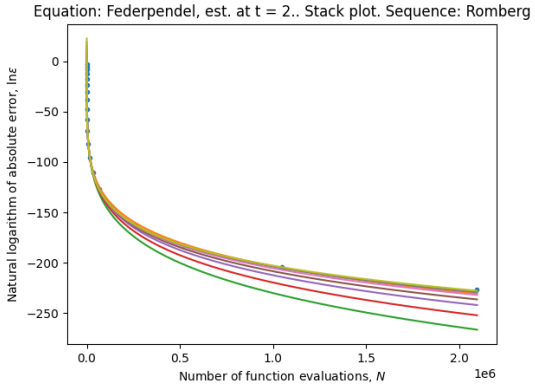
Here we use 1 as initial step length.

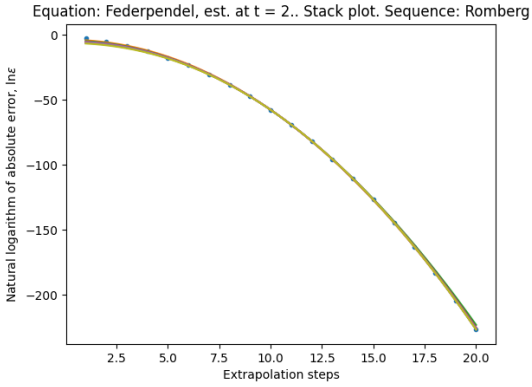


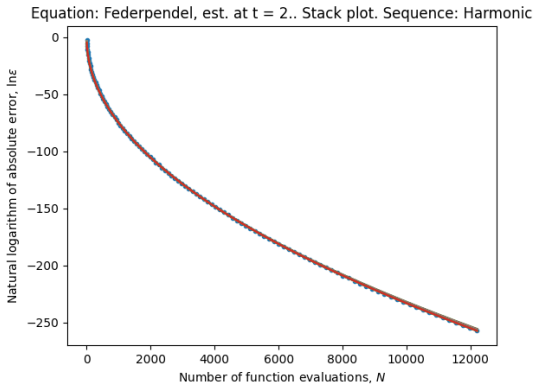


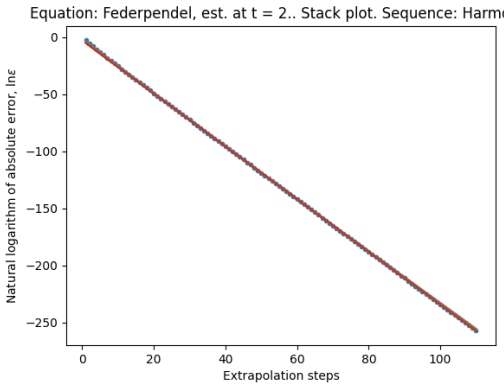












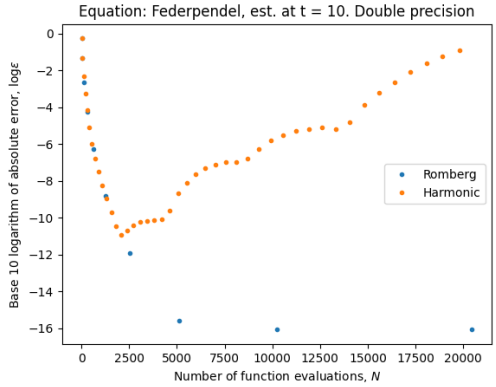
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	6.5[+39]	6.0[+00]	4.2[+01]	1.2[−01]	1.4[−01]	2.7[−02]	4.0[+05]	6.0[−04]
HS-evals	4.2[−01]	3.7[−01]	2.4[+00]	5.1[−04]	4.9[−01]	2.2[−05]	1.8[−01]	1.3[−06]
RS-steps	5.0[−03]	4.6[−01]	4.4[−01]	4.0[−03]	2.1[+00]	1.1[−04]	5.6[−01]	2.2[−05]
HS-steps	1.3[−01]	3.6[−01]	2.4[+00]	5.2[−04]	9.9[−01]	2.2[−05]	2.3[−01]	1.5[−06]

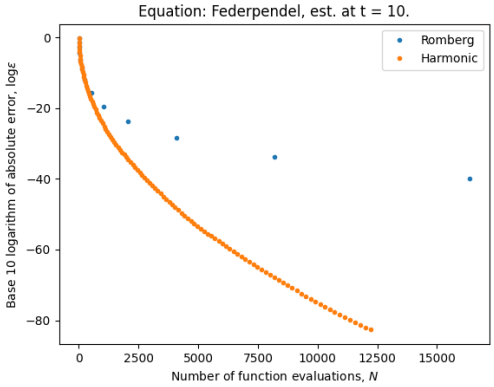
Here the harmonic sequence also works better and we attain high precision using either sequence in standard double precision floating point arithmetic.

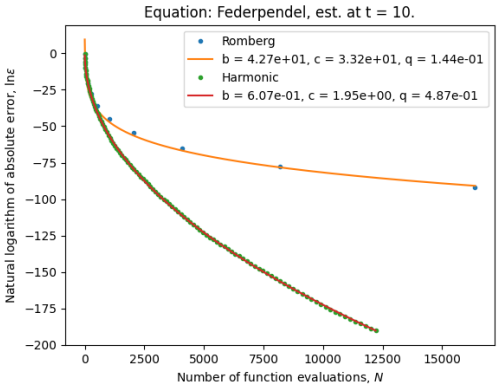
We seem to have very nice fit for exponential convergence in the number of steps for the Romberg sequence. We also have very nice fit for exponential convergence for the harmonic sequence.

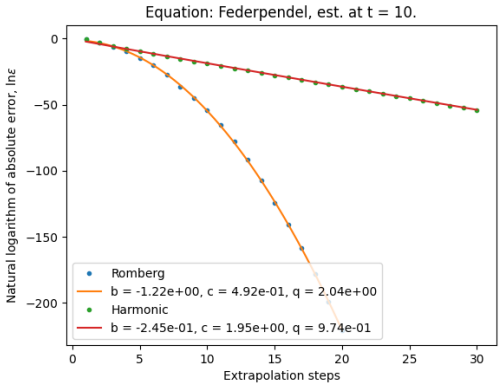
$t = 10$

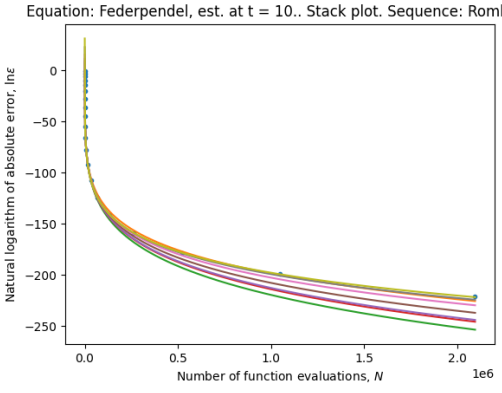
Here we use 1 as initial step length. When too big initial step length was used, the convergence failed. Hence the results are dependent upon the initial step length.

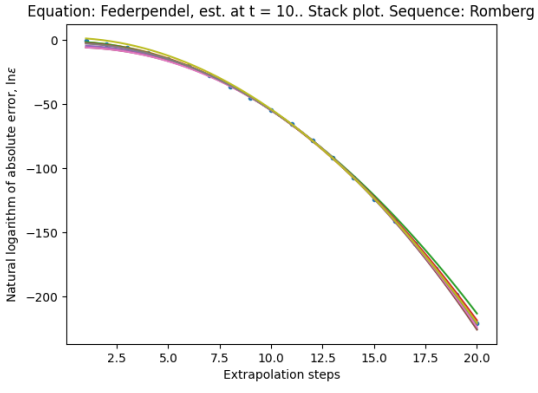


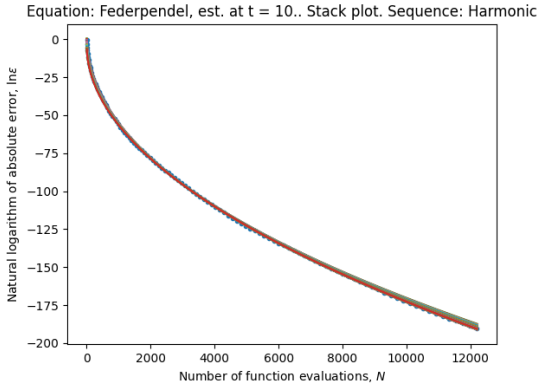


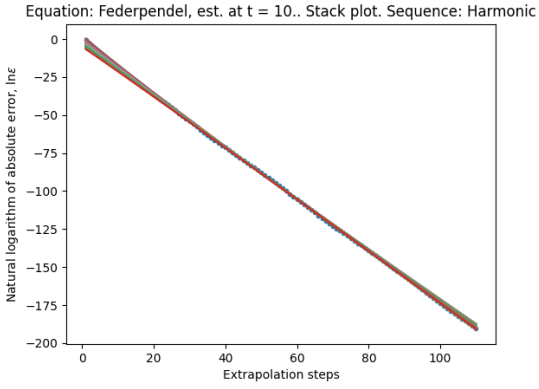












Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.9[+50]	6.0[+00]	4.3[+01]	2.0[−01]	1.4[−01]	3.0[−02]	6.4[+05]	7.4[−04]
HS-evals	3.9[+00]	3.9[+00]	1.9[+00]	1.1[−02]	4.9[−01]	4.6[−04]	6.0[−01]	1.2[−05]
RS-steps	9.7[−01]	4.8[+00]	4.4[−01]	4.0[−02]	2.1[+00]	1.1[−03]	4.4[−01]	2.7[−05]
HS-steps	1.6[+00]	3.9[+00]	1.9[+00]	1.1[−02]	9.8[−01]	4.6[−04]	6.3[−01]	1.2[−05]

Here we still have exponential convergence in the number of steps for both sequences.

4.2.10 Lorenz equations

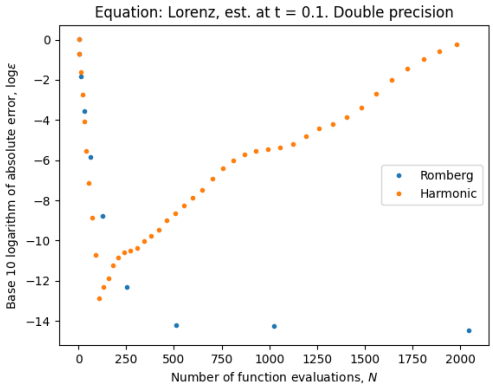
The Lorenz equations are the following system:

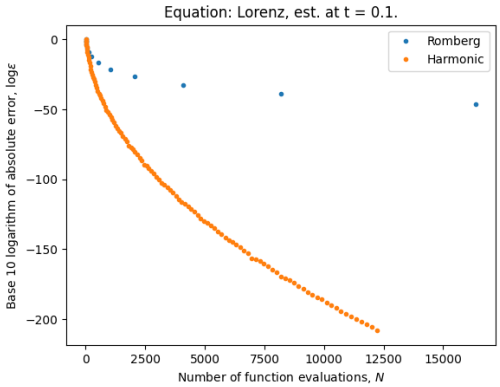
$$\frac{dx}{dt} = \sigma(y - x), \quad \frac{dy}{dt} = x(\rho - z) - y, \quad \frac{dz}{dt} = xy - \beta z$$

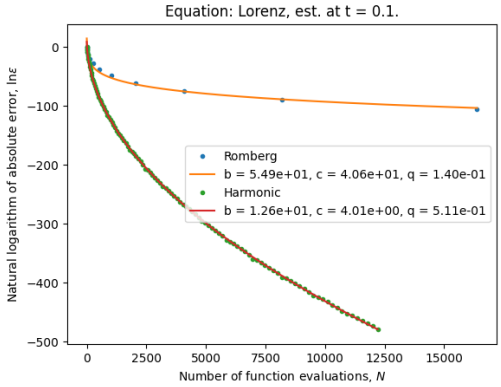
where σ, ρ and β are constants. In our experiment, the constants are set to $\sigma = 10$, $\rho = 28$ and $\beta = 8/3$. The initial condition we will consider is $(x(0), y(0), z(0)) = (1, 1, 1)$.

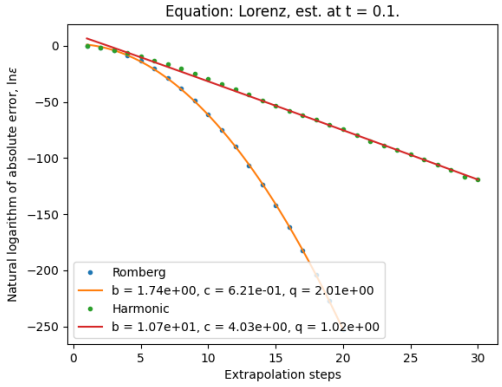
We use precomputed reference solutions with accurracy of 500 digits which can be found in the code.

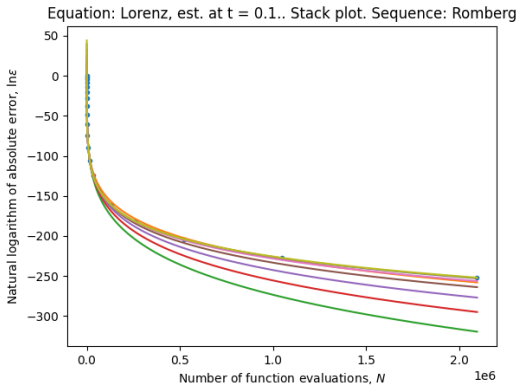
$t = 0.1$

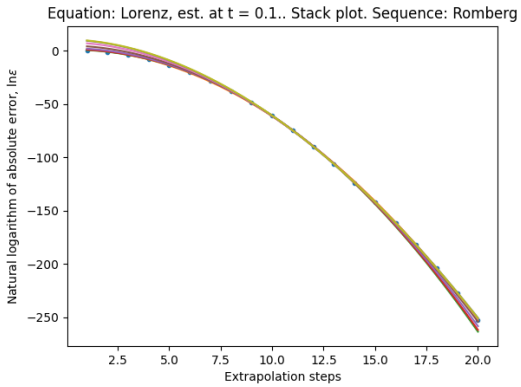


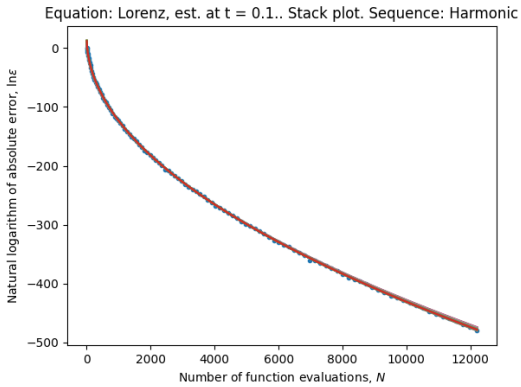


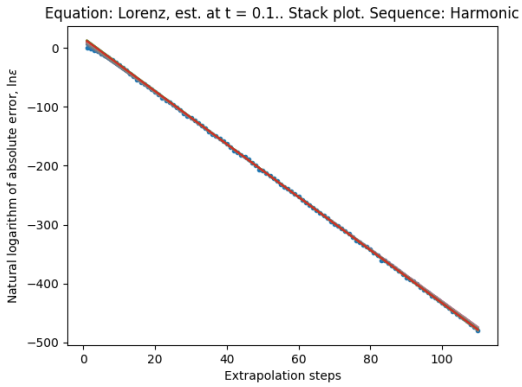












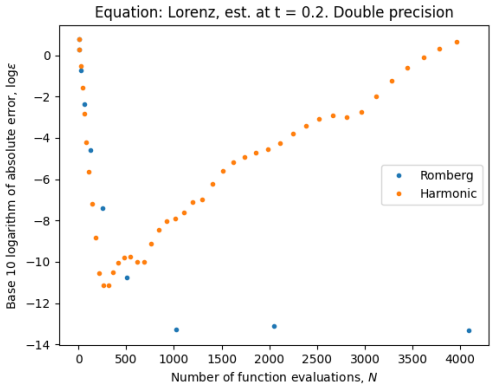
Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	1.5[+73]	6.0[+00]	6.6[+01]	2.6[−01]	1.3[−01]	7.3[−02]	5.8[+08]	1.2[−03]
HS-evals	2.6[+07]	6.8[+00]	4.2[+00]	1.8[−03]	5.1[−01]	7.2[−05]	9.2[+05]	2.2[−05]
RS-steps	8.9[+03]	2.3[+00]	7.2[−01]	6.4[−02]	2.0[+00]	2.0[−03]	4.0[+00]	3.6[−05]
HS-steps	2.9[+06]	6.6[+00]	4.2[+00]	1.8[−03]	1.0[+00]	7.1[−05]	5.4[+05]	2.1[−05]

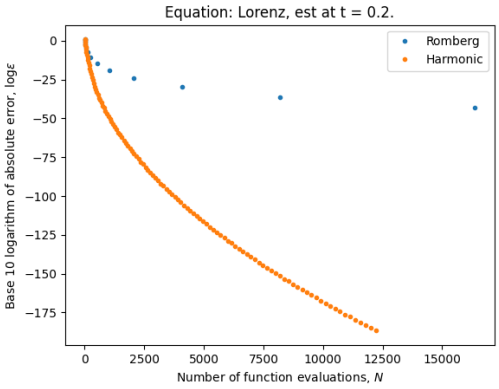
Here the harmonic sequence works better and we get down to machine level precision in standard double precision arithmetic.

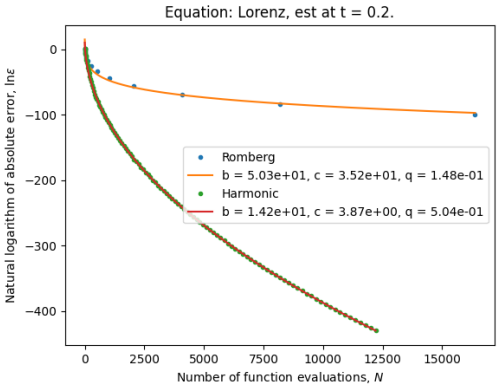
The model for exponential convergence in the number of steps fits moderately well for the Romberg sequence. We get nice fit for exponential convergence for the harmonic sequence.

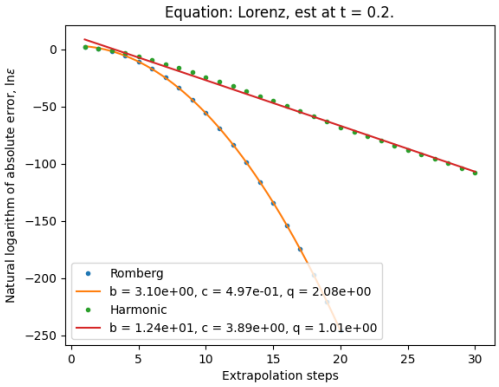
$t = 0.2$

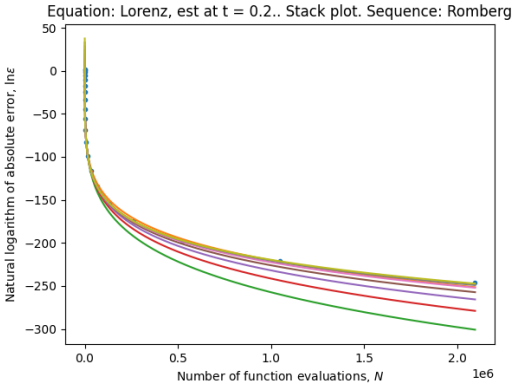
Here we use 0.1 as initial step length.

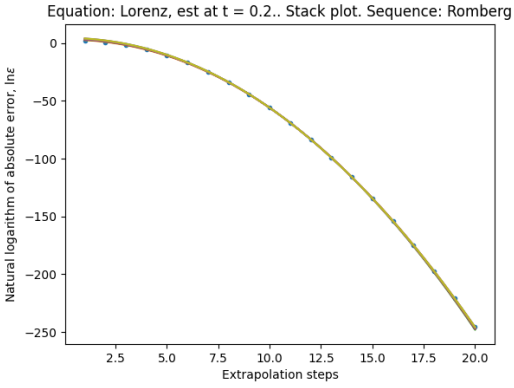


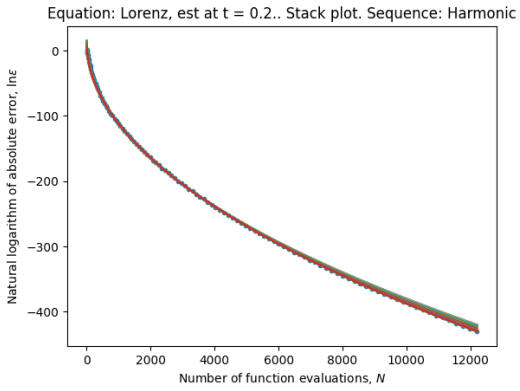


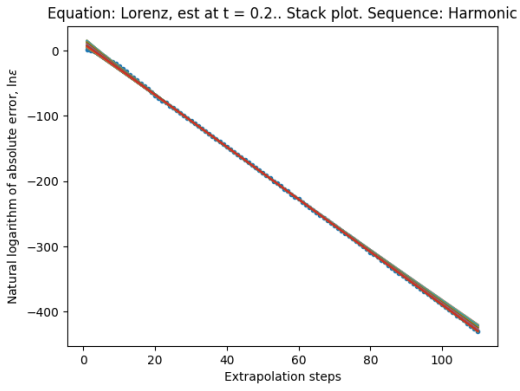












Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var	ρ_{lin}	ρ_{ln}
RS-evals	3.8[+53]	6.0[+00]	5.0[+01]	1.6[−01]	1.4[−01]	3.8[−02]	2.0[+08]	1.1[−03]
HS-evals	6.3[+08]	7.6[+00]	4.0[+00]	1.5[−02]	5.0[−01]	5.8[−04]	9.5[+05]	2.3[−05]
RS-steps	6.1[+01]	1.6[−01]	5.2[−01]	1.3[−03]	2.1[+00]	3.8[−05]	1.5[+00]	7.0[−06]
HS-steps	7.3[+07]	7.5[+00]	4.0[+00]	1.5[−02]	1.0[+00]	5.7[−04]	6.0[+05]	2.2[−05]

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

The model for exponential convergence in the number of steps fits moderately well for the Romberg sequence. We get moderate fit for exponential convergence for the harmonic sequence.

4.3 Summary

When the solutions are entire we seem to have exponential convergence in the number of steps for both the Romberg sequence and the harmonic sequence. The same holds when the solutions are analytic and we are not too close to a singularity. When we move closer to the singularities, the fitting becomes worse and the convergence begins to fail.

It is interesting how fast convergence we get with the harmonic sequence when computing nicely behaved solutions.

It would be interesting to analyze further how the convergence depends on the initial step size.

Bibliography

[1] Peter Deuflhard and Folkmar Bornemann. *Scientific Computing with Ordinary Differential Equations*, vol. 42 of Texts in Applied Mathematics, Springer, New York, 2002.

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

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