

Chapter 1

Extrapolation to zero

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

1.1 Motivation

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

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

for $h \rightarrow 0$. We are interested in computing $T_0 = \lim_{h \rightarrow 0} T(h)$ up to some desired accuracy. 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

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

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 $\tau_p, \tau_{2p}, \dots, \tau_{mp} \in \mathbb{R}$ such that

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

for $h \rightarrow 0$, $h \in \mathbb{H}$.

Let $(x_1, y_1), \dots, (x_k, y_k)$ be a collection of points such that x_1, \dots, x_k are distinct. Then there exists a unique 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 $(x_1^p, y_1), \dots, (x_n^p, y_n)$ such that x_i^p are distinct, be given. Let P be the interpolation polynomial for the points. We then call $P(h^p)$ the *interpolation polynomial in p* for 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, \dots, h_k where $k \leq m$. Then we compute the interpolation polynomial P in h^p which goes through $(h_1, T(h_1)), \dots, (h_k, T(h_k))$. We then hope that $P(0)$ gives a good approximation T_0 .

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

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

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

1.3 Convergence

If we have a numerical method or scheme that has an asymptotic expansion as (1.3), then the error decays polynomially as $h \rightarrow 0$. It is known (see e.g. theorem 9.22 in [?]) that T_{ij} has polynomial decay of higher degree, as $h \rightarrow 0$, than T . Let $\varepsilon_k := |T_{kk} - T|$. We want to analyse how ε_k behaves as $k \rightarrow +\infty$, i.e. how ε_k behaves when we increase 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.

$$\varepsilon_k \sim A \exp(-cN_k^q) \quad (1.4)$$

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

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

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

$$\varepsilon_k \sim A \exp(-ck^q) \quad (1.6)$$

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 $k = 0, 1, \dots, n$. Let $r = \max(10, [n/2])$. Then we will do the curve fitting for $(x_k, y_k), \dots, (x_{k+r-1}, y_{k+r-1})$ for $k = 3, \dots, n-r-1$ and compute the relative variance of the parameters. Let $a_k, k = 1, \dots, m$ be numbers. Then we define their mean value by $\bar{a} := \frac{1}{m} \sum_{k=1}^m a_k$, and the relative variance by

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

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 = np.array([np.array([mpf(0) if hp else 0] * (i+1)) for i in range(n)])
    #X[i][j] = T_ij
```

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

return X
```

Chapter 2

Romberg quadrature

2.1 The algorithm

Let $f : [a, b] \rightarrow \mathbb{R}$ be a function and $I := \int_a^b f(x)dx$. The *trapezoidal rule* is a method for approaching I which works as follows: Let $a = t_0 < t_1 < \dots < t_n = b$ 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 vertices $(t_{i-1}, 0)$, $(t_{i-1}, f(t_{i-1}))$, $(t_i, f(t_i))$, $(t_i, 0)$ 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

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

We define $T_f(h)$ as the right hand side in (2.1) for all $h \in \{(b - a)/n \mid n \in \mathbb{Z}_+ \dots\} =: \mathbb{H}$.

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

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

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

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

where

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

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 $h = h_1, \dots, h_k$ 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 $(h_i^2, T_f(h_i))$, $i = 1, \dots, k$. Denote by T_{ij} the value at zero of the polynomial of degree $j - 1$ in h^2 which goes through $(h_{i-j+1}^2, T(h_{i-j+1}))$, \dots , $(h_i^2, T(h_i))$. The Neville scheme gives us the following algorithm for computing T_{ij} , $1 \leq j \leq i \leq k$, recursively:

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

2.2 Numerical experiments

In this section we are going to apply Romberg quadrature to various functions and also try different sequences. We will 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

$$1, 2, 3, 4, 6, 8, 12, 16, 24, 32, \dots$$

Suppose that we are approximating the integral $I := \int_a^b f(x)dx$ using Romberg quadrature. We will use the stepsizes $h_k := (b - a)/a_k$ for the extrapolation. Let T_{ij} , $i \geq 0$ and $j \leq i$ be the extrapolation table we get and $\varepsilon_k := |T_{kk} - I|$ be the error on the diagonals. 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 $A := e^b$, 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.

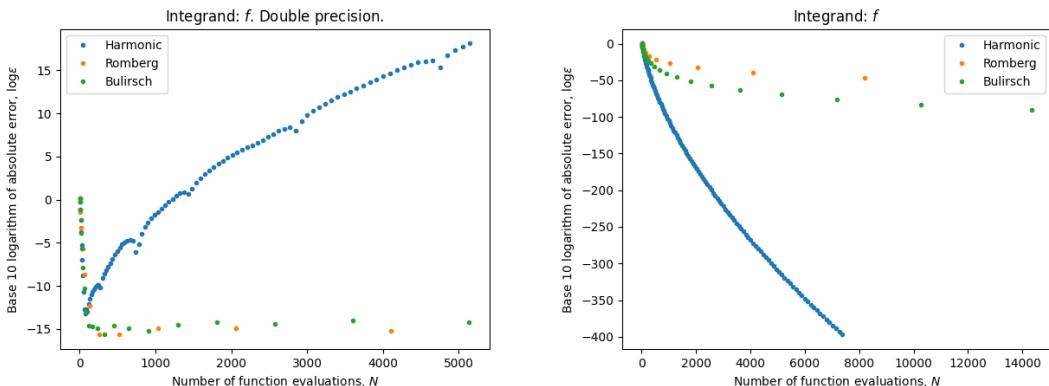
Now we will consider the results of the experiments.

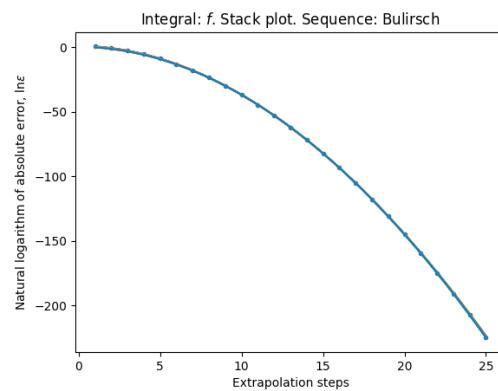
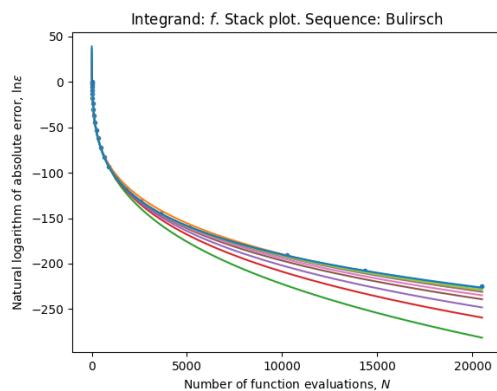
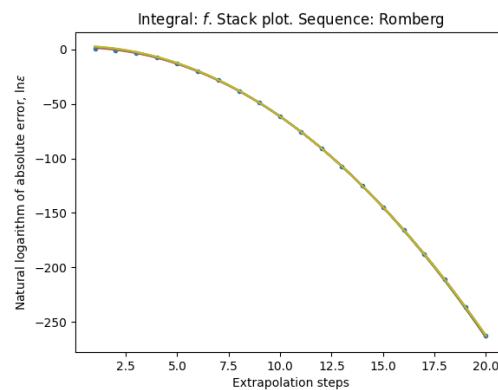
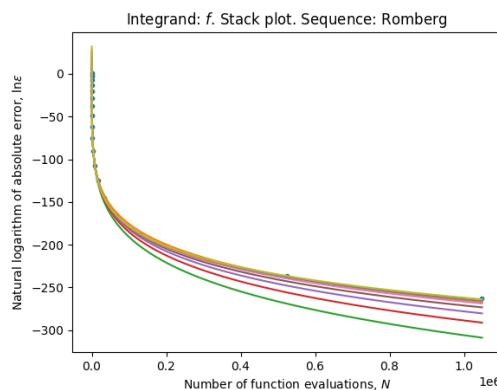
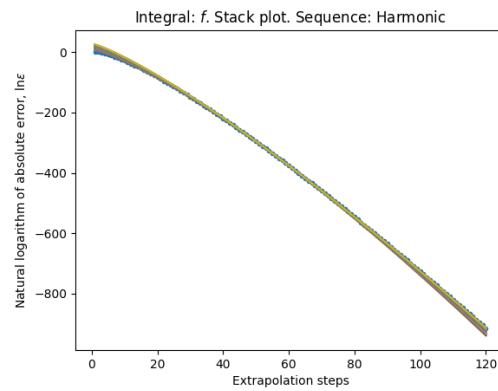
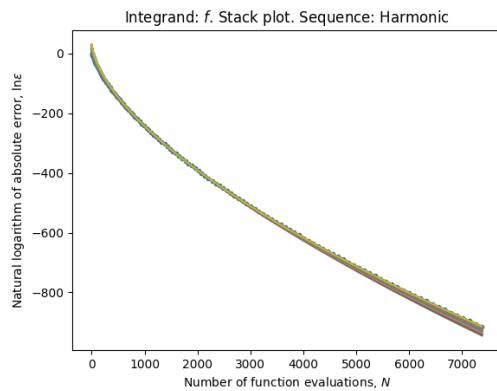
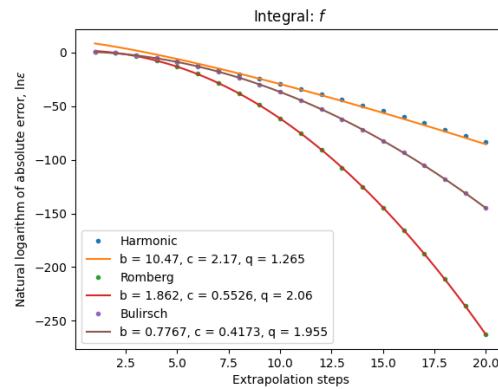
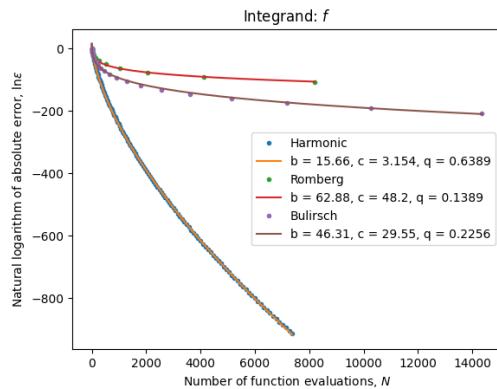
2.2.1 Cosine squared

The first function we are going to try is

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

which is entire.





Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Harmonic	lin-lin evals-error	$4.426 \cdot 10^{14}$	8.674	3.524	0.01154	0.6287	0.0004025
Romberg	lin-lin evals-error	$1.853 \cdot 10^{58}$	6	63.25	0.11	0.1312	0.02905
Bulirsch	lin-lin evals-error	$2.195 \cdot 10^{53}$	7.998	47.04	0.1714	0.2034	0.04509
Harmonic	lin-lin steps-error	$8.963 \cdot 10^{10}$	7.637	2.386	0.009485	1.249	0.0002853
Romberg	lin-lin steps-error	16.31	0.1345	0.5749	0.0008091	2.049	$2.316 \cdot 10^{-5}$
Bulirsch	lin-lin steps-error	1.658	0.06912	0.4129	0.0006159	1.957	$1.639 \cdot 10^{-5}$

We see that the harmonic sequence performs 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.

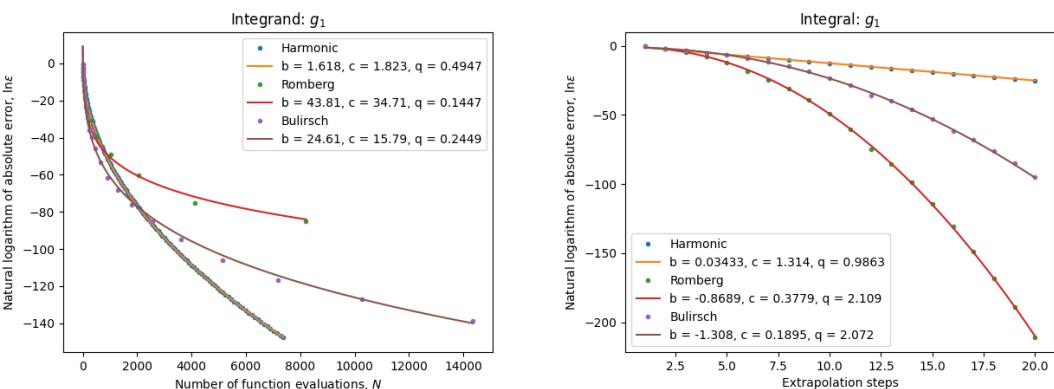
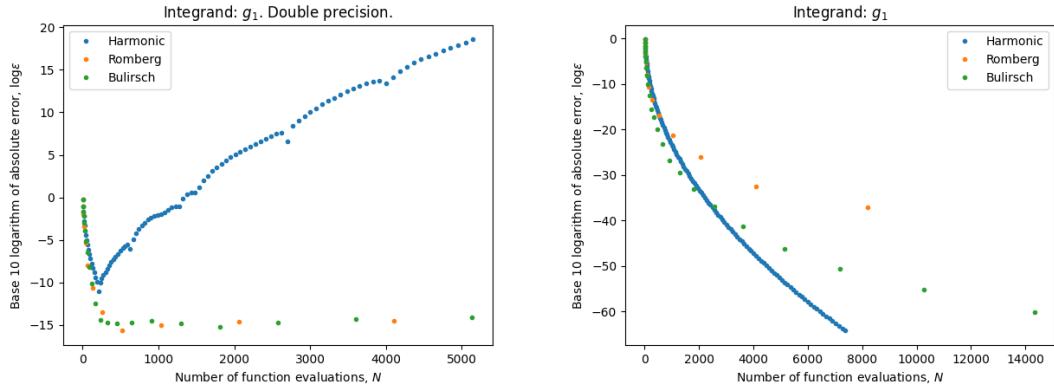
For the Romberg and Bulirsch sequence we clearly have exponential convergence in the number of steps but not in the number of evaluations.

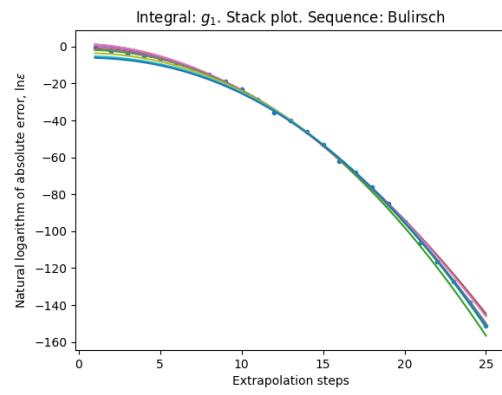
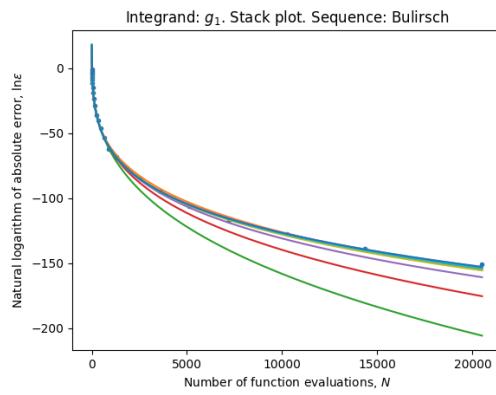
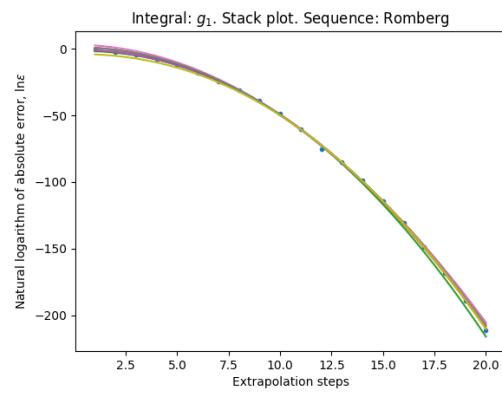
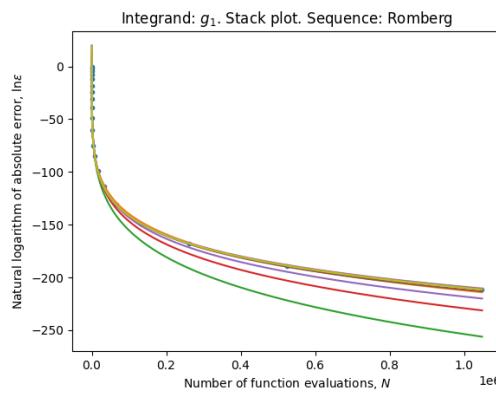
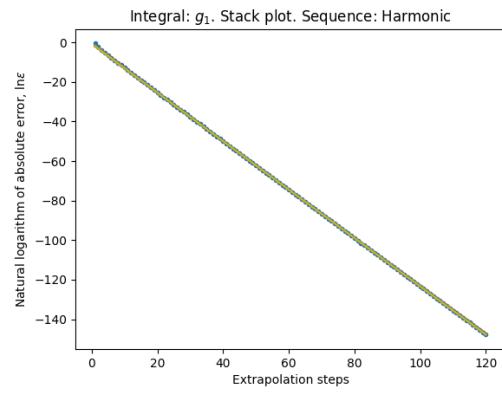
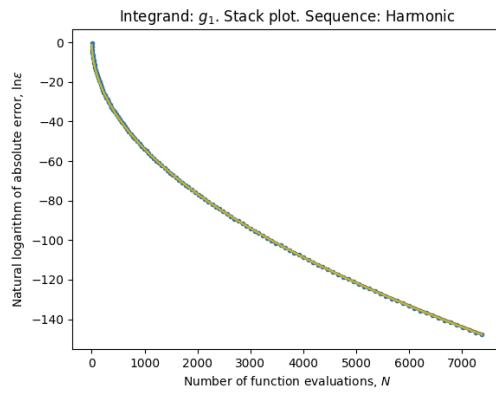
For the harmonic sequence we get unreasonably large values of A and we can see from the stack plots that the fitting is not so stable.

2.2.2 Function with poles

Now we will consider the following function:

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

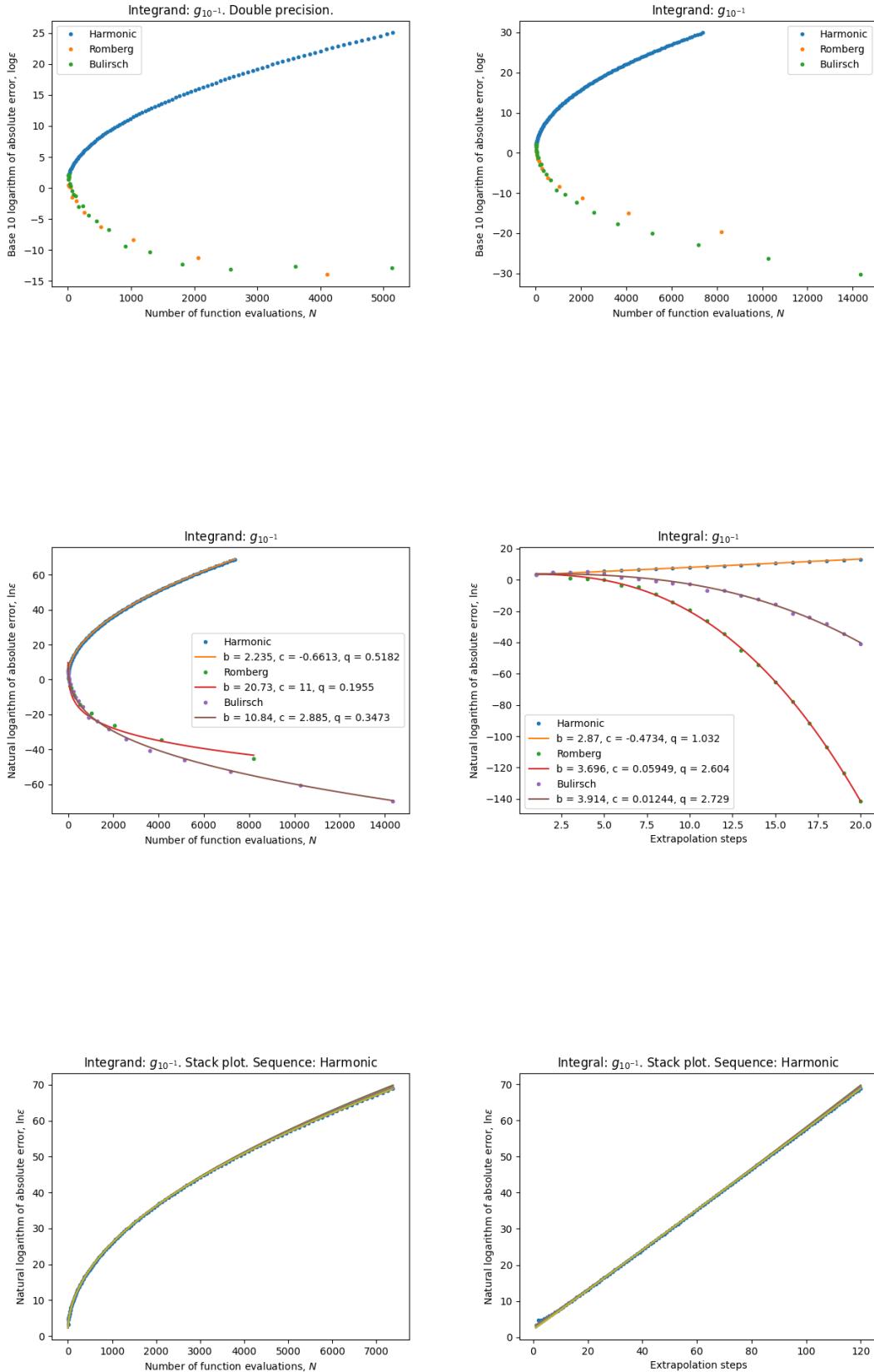


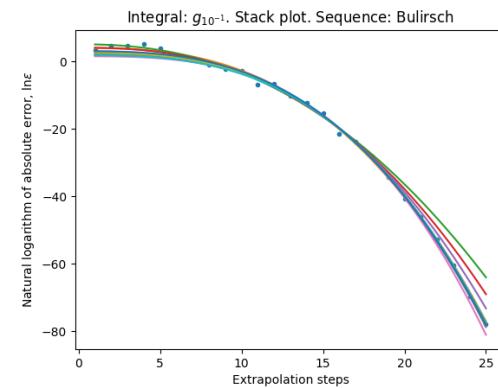
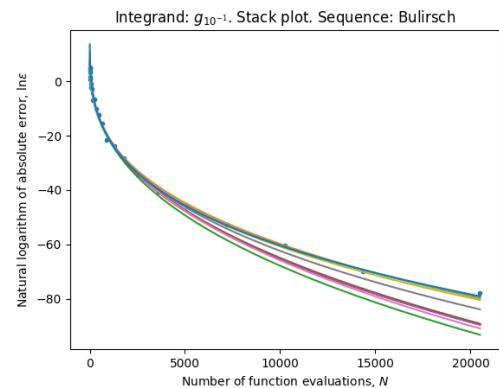
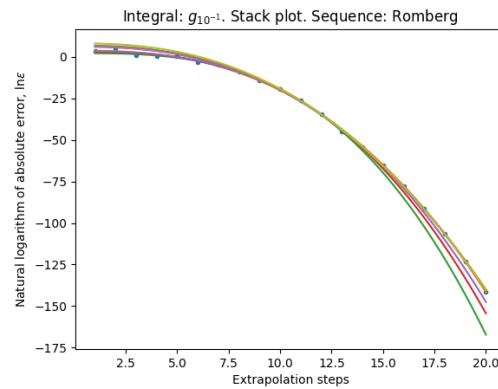
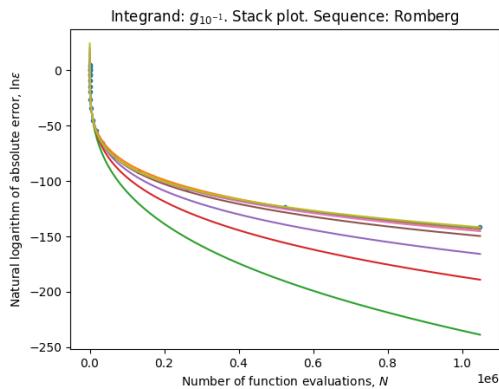


Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Harmonic	lin-ln evals-error	3.771	0.02895	1.793	0.0001375	0.4963	$6.495 \cdot 10^{-6}$
Romberg	lin-ln evals-error	$4.114 \cdot 10^{37}$	5.211	46.94	0.1026	0.135	0.03489
Bulirsch	lin-ln evals-error	$8.199 \cdot 10^{22}$	4.987	24.22	0.1103	0.2236	0.04529
Harmonic	lin-ln steps-error	0.6596	0.04548	1.277	0.0002625	0.9918	$1.093 \cdot 10^{-5}$
Romberg	lin-ln steps-error	3.747	2.921	0.4181	0.02976	2.079	0.0008172
Bulirsch	lin-ln steps-error	1.191	1.72	0.2048	0.105	2.062	0.00252

We see that the harmonic sequence performs 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.

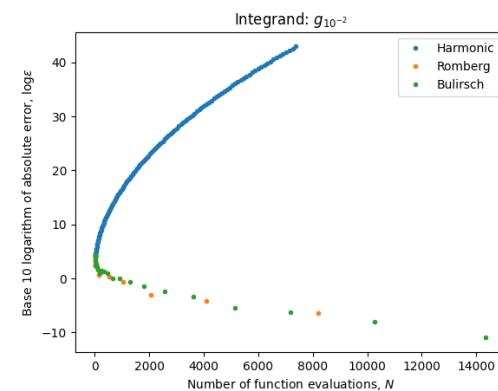
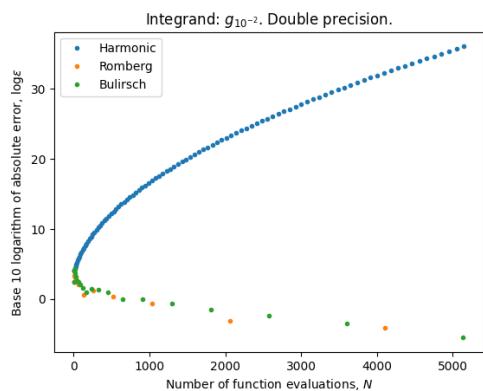
Here we clearly have exponential convergence in the number of evaluations for the Harmonic sequence and hence also in the number of steps. We do not have exponential convergence in the number of evaluations for Romberg and Bulirsch but the model for exponential convergence in the number of steps fits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

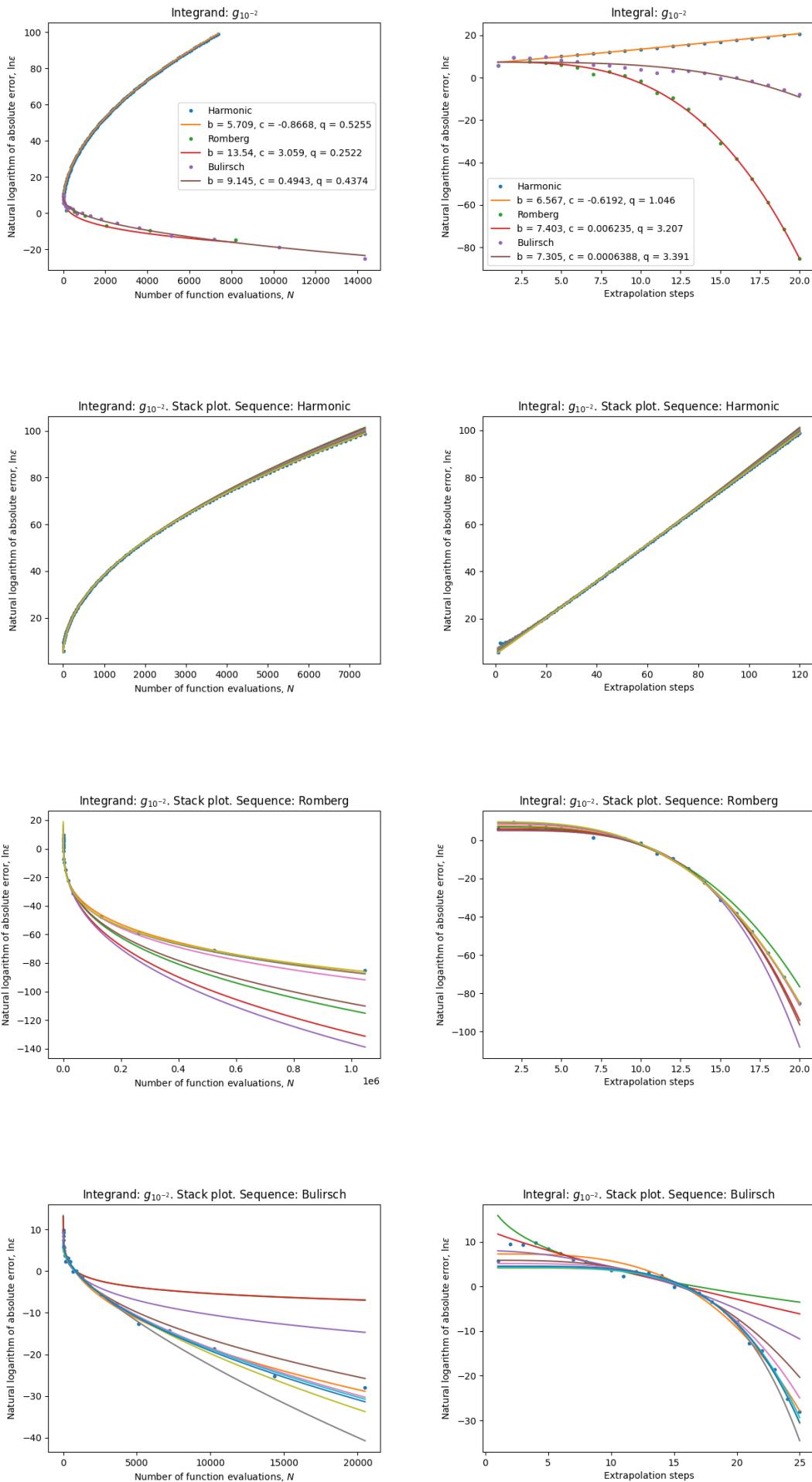




Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Harmonic	lin-ln evals-error	5.231	0.0773	-0.7178	0.001803	0.5101	$9.026 \cdot 10^{-5}$
Romberg	lin-ln evals-error	$7.533 \cdot 10^{22}$	5.999	14.35	0.3322	0.2047	0.07855
Bulirsch	lin-ln evals-error	$8.963 \cdot 10^7$	7.604	2.995	0.186	0.3602	0.0129
Harmonic	lin-ln steps-error	10.97	0.05032	-0.5086	0.001492	1.019	$6.47 \cdot 10^{-5}$
Romberg	lin-ln steps-error	727.7	2.108	0.06624	0.1819	2.623	0.005292
Bulirsch	lin-ln steps-error	32.46	2.026	0.01381	0.7483	2.764	0.006506

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.





Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Harmonic	lin-lin evals-error	122.7	0.8001	-0.9641	0.007205	0.5155	0.0003682
Romberg	lin-lin evals-error	$8.898 \cdot 10^{10}$	5.785	3.021	0.6343	0.2928	0.05447
Bulirsch	lin-lin evals-error	.	.	2876	4.874	0.4042	0.3708
Harmonic	lin-lin steps-error	323.1	0.5753	-0.682	0.006766	1.029	0.0002999
Romberg	lin-lin steps-error	4384	1.505	0.005443	0.4455	3.388	0.0103
Bulirsch	lin-lin steps-error	$1.73 \cdot 10^{11}$	8	1.498	6.41	3.17	0.272

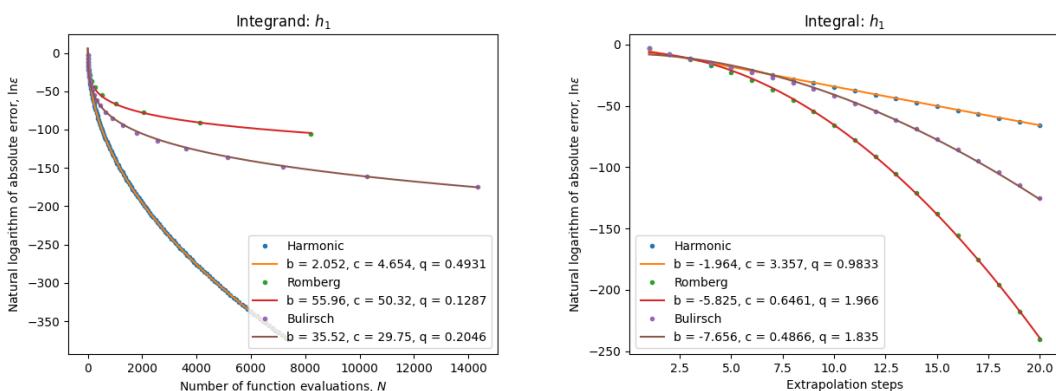
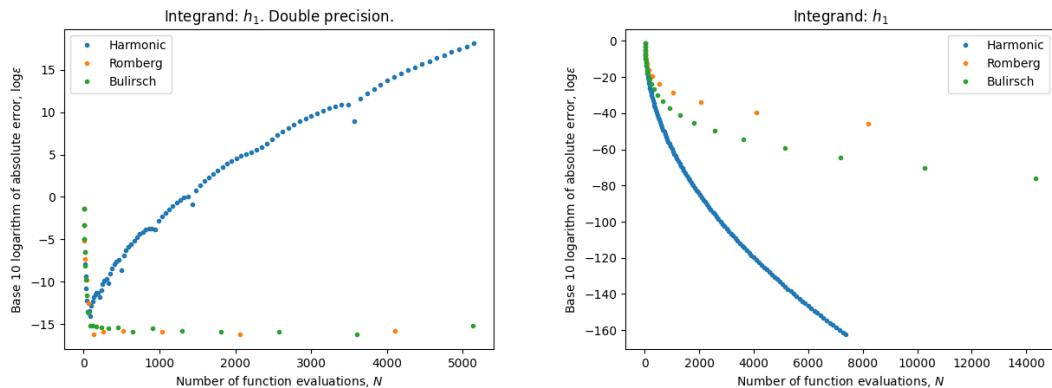
Here the same comments apply as for $a = 10^{-1}$, except that now the Romberg sequence performs better than the Bulirsch sequence and the model fitting is worse.

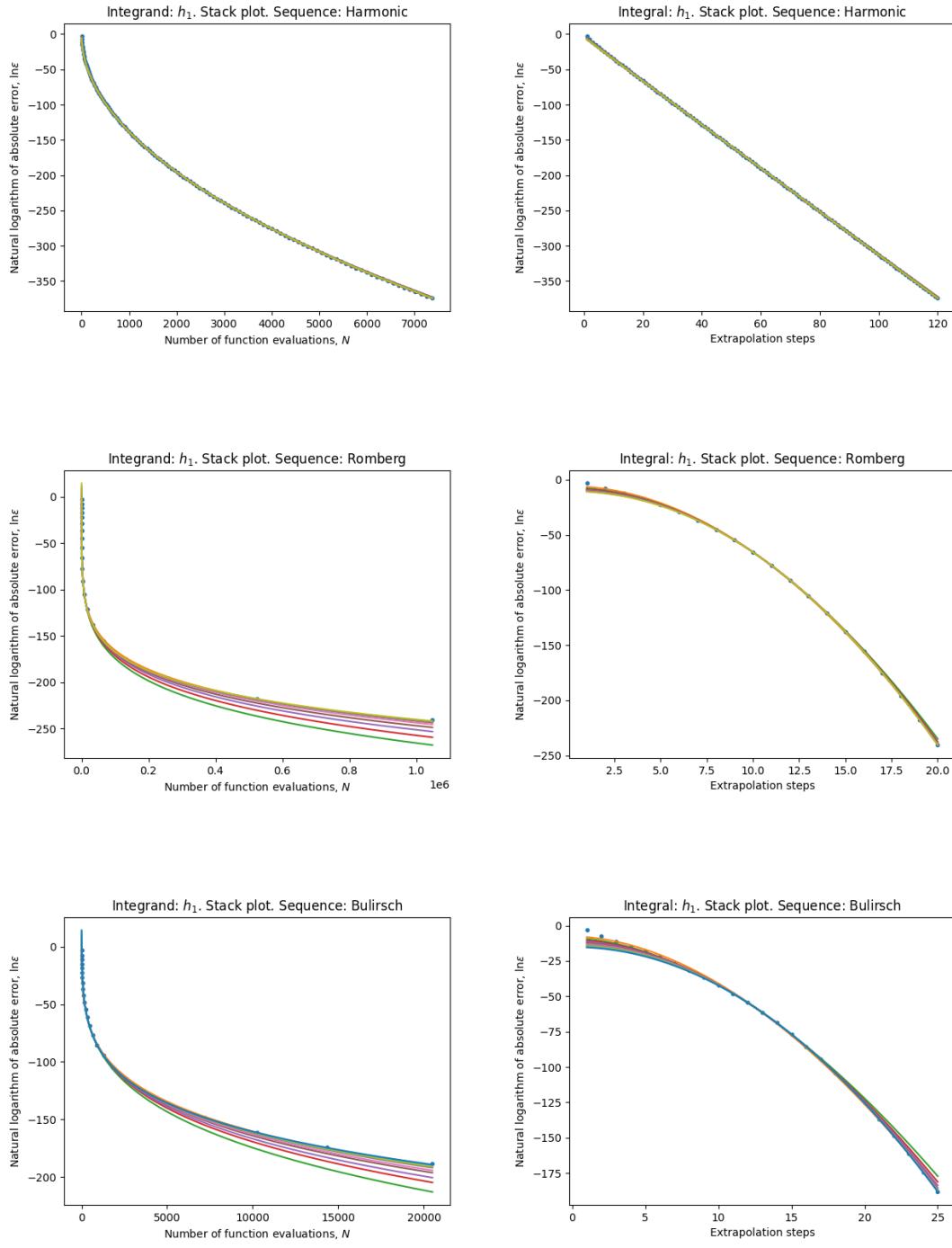
2.2.3 Logarithm

Now we will consider the following function

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

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

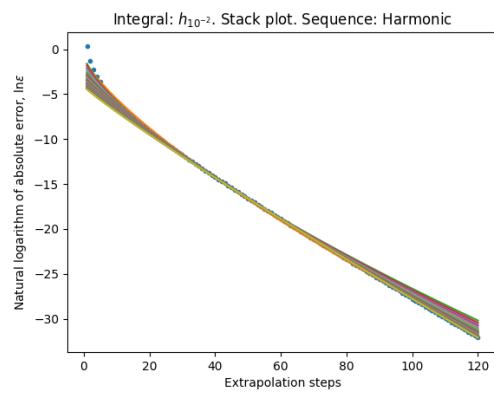
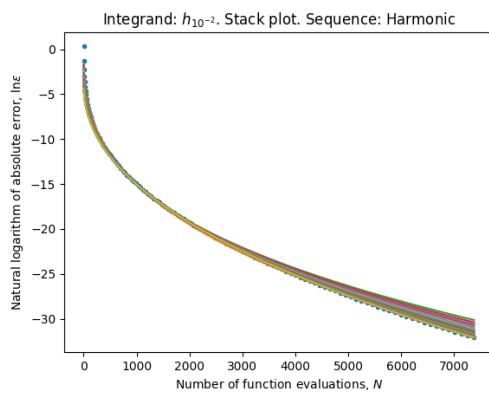
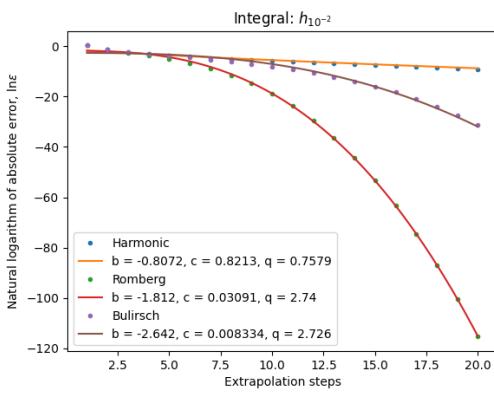
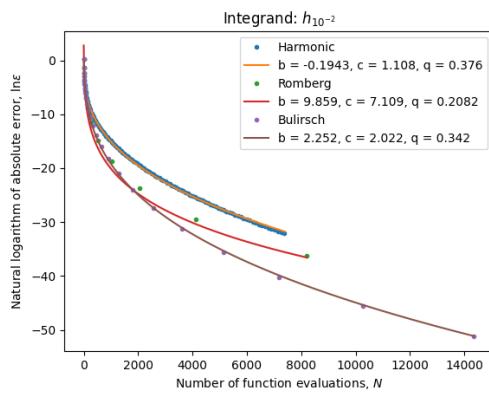
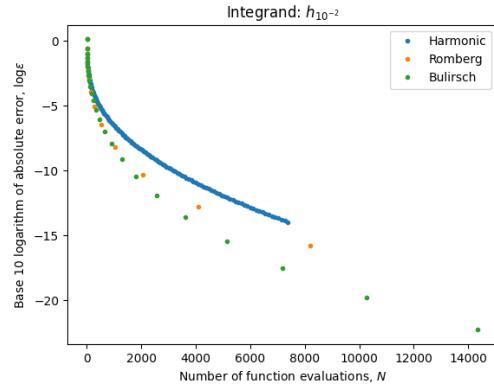
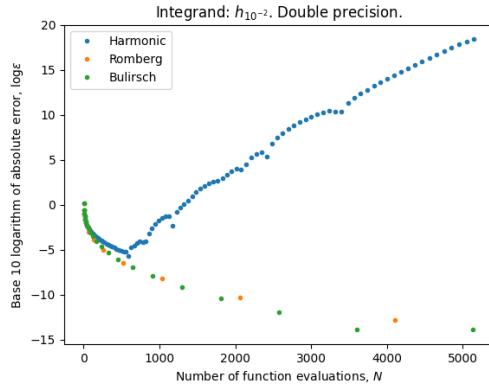


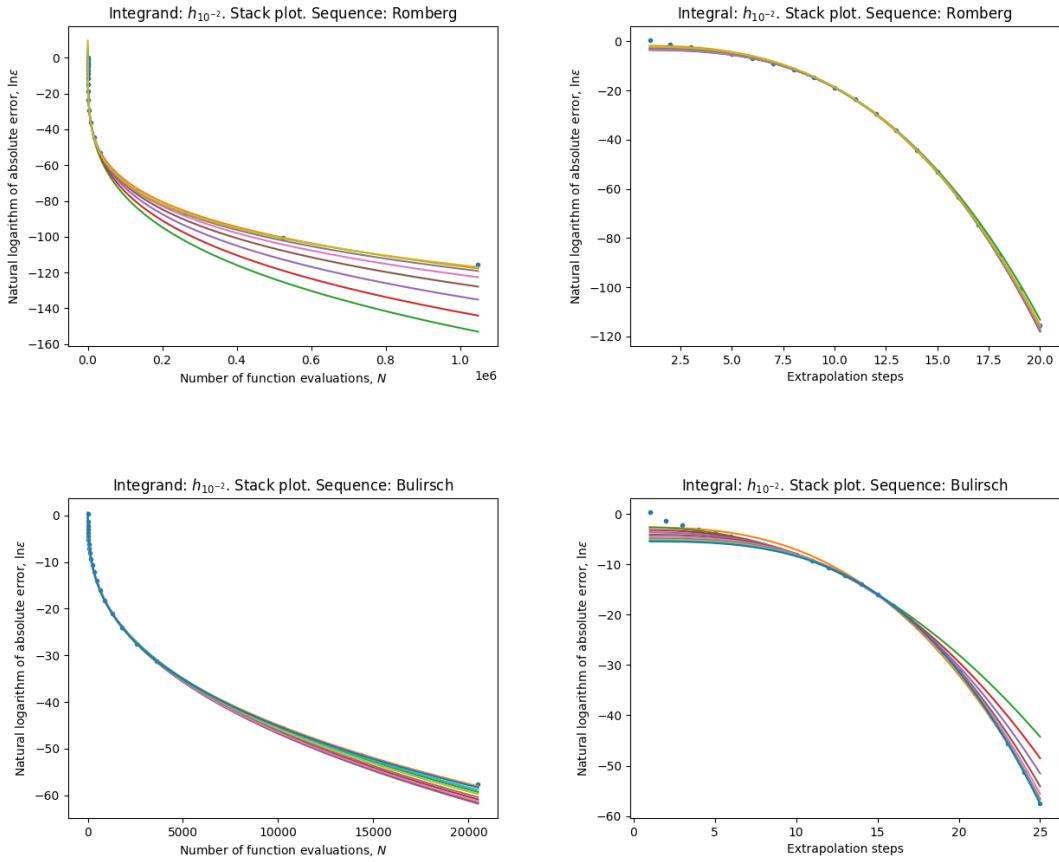


Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Harmonic	lin-ln evals-error	3.072	0.3632	4.545	0.000229	0.4954	$1.085 \cdot 10^{-5}$
Romberg	lin-ln evals-error	$1.538 \cdot 10^{43}$	6	56.82	0.06603	0.1269	0.0163
Bulirsch	lin-ln evals-error	$1.55 \cdot 10^{31}$	7.894	36.84	0.08131	0.1955	0.0187
Harmonic	lin-ln steps-error	0.03919	0.5477	3.237	0.0003865	0.9901	$1.608 \cdot 10^{-5}$
Romberg	lin-ln steps-error	0.0002063	1.214	0.5431	0.009515	2.02	0.000263
Bulirsch	lin-ln steps-error	$2.293 \cdot 10^{-5}$	3.407	0.3579	0.06839	1.929	0.001656

We see that the harmonic sequence performs 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.

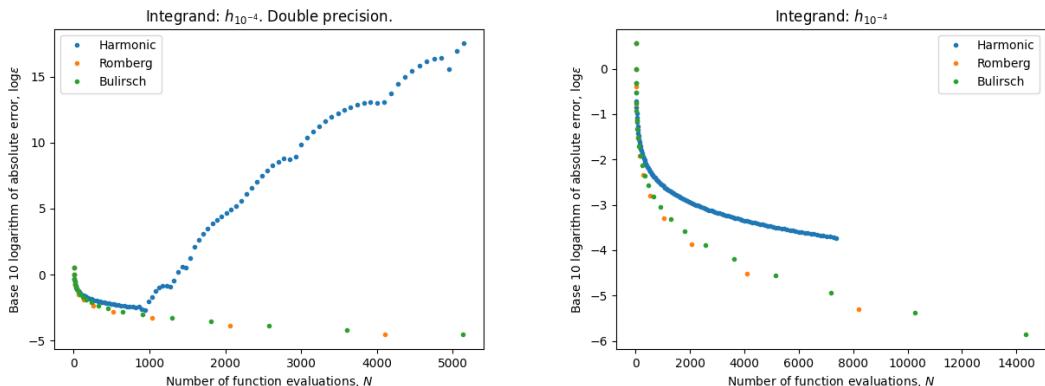


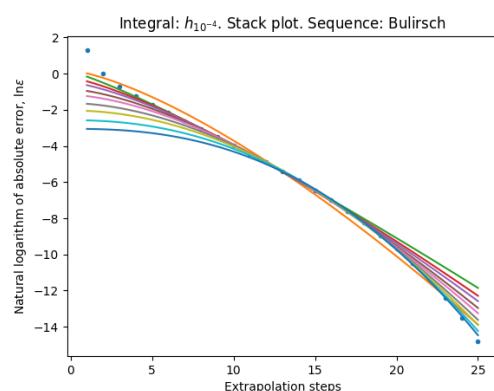
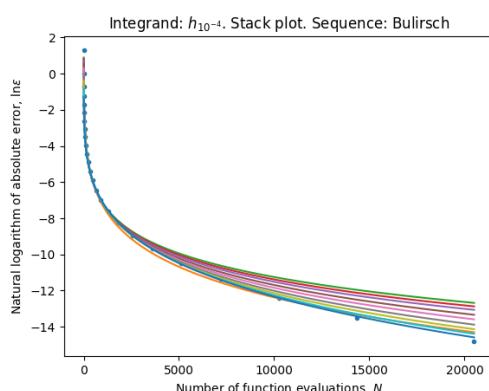
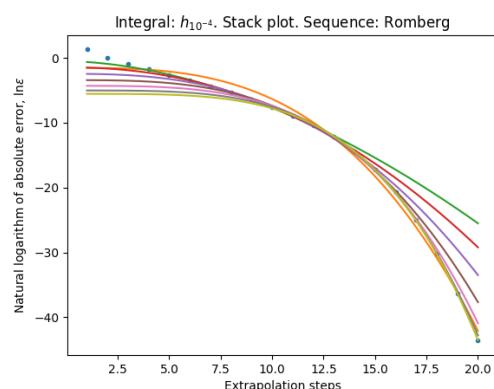
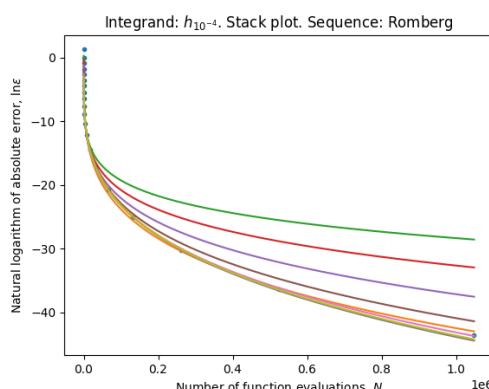
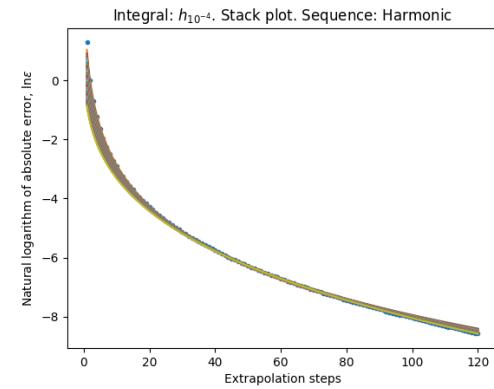
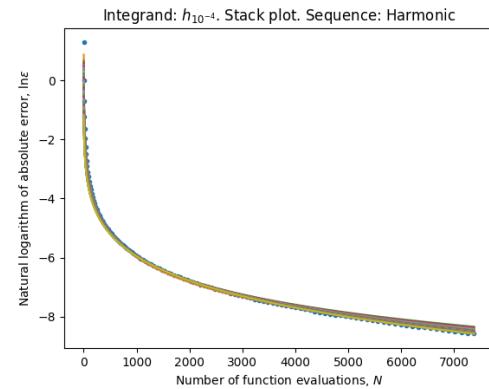
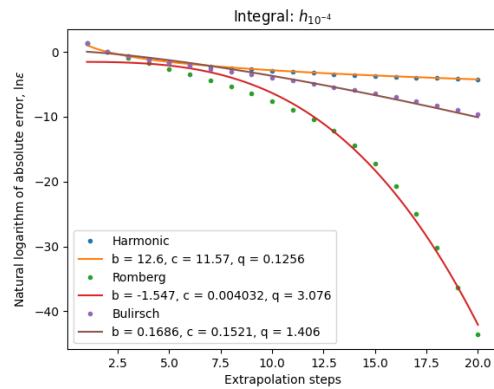
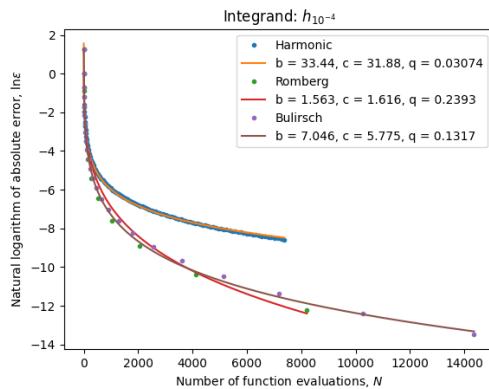


Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Harmonic	lin-ln evals-error	0.1351	2.455	0.6631	0.1084	0.4296	0.004684
Romberg	lin-ln evals-error	$3.388 \cdot 10^9$	5.969	6.696	0.2315	0.2286	0.02614
Bulirsch	lin-ln evals-error	6.859	1.956	1.626	0.0311	0.3675	0.002076
Harmonic	lin-ln steps-error	0.07535	1.798	0.4822	0.1106	0.8628	0.0042
Romberg	lin-ln steps-error	0.05205	0.2618	0.02252	0.01836	2.848	0.0002742
Bulirsch	lin-ln steps-error	0.0214	1.016	0.00882	1.141	2.822	0.01164

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

For Romberg, we get a moderately good 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.





Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Harmonic	lin-ln evals-error	$1.328 \cdot 10^{13}$	55.82	8.14	0.5474	0.09568	0.0936
Romberg	lin-ln evals-error	106.1	5.445	2.037	0.7267	0.2364	0.06414
Bulirsch	lin-ln evals-error	$1.888 \cdot 10^5$	5.399	6.394	0.3315	0.1362	0.1274
Harmonic	lin-ln steps-error	$1.395 \cdot 10^4$	17.13	5.261	0.2049	0.2157	0.04738
Romberg	lin-ln steps-error	0.1362	2.091	0.02426	2.55	3.039	0.07013
Bulirsch	lin-ln steps-error	0.4121	0.6973	0.09479	0.8435	1.695	0.05417

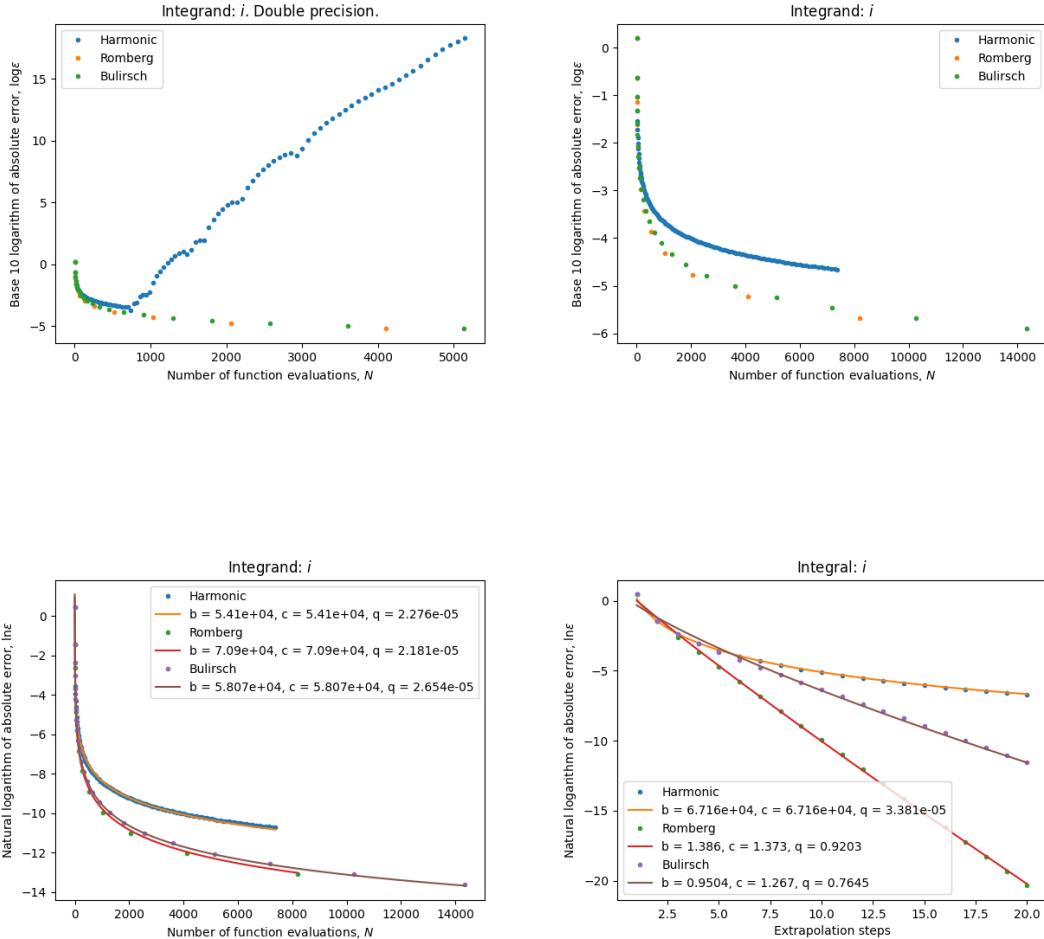
Here again, we do not attain high precision when using the Harmonic sequence in double precision arithmetic. It is hard to say which sequence performs best. We get reasonably good fit for the harmonic sequence but not for Romberg and Bulirsch.

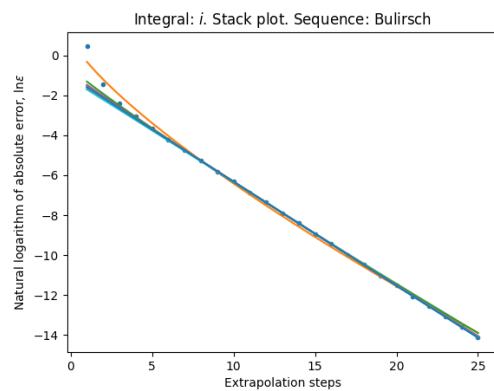
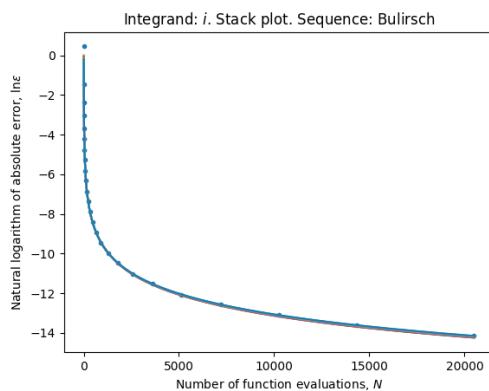
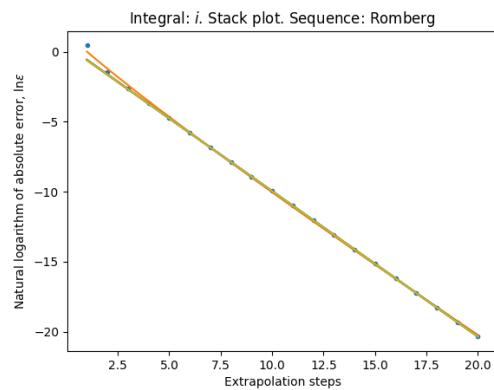
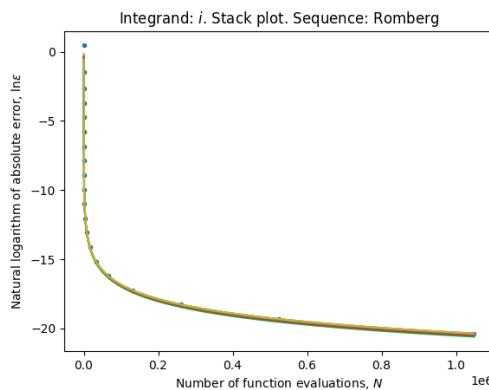
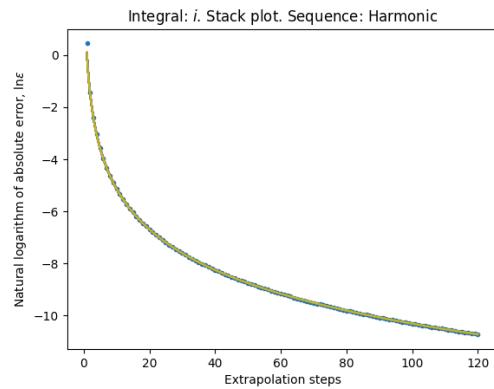
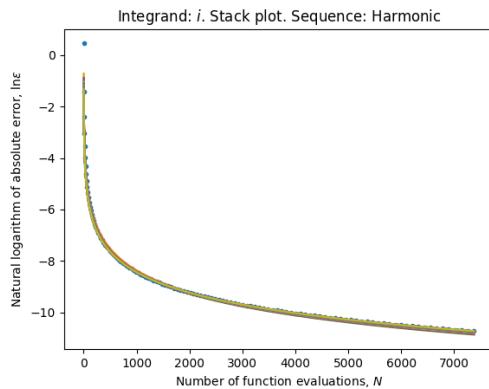
2.2.4 Area of half circle

Now we will try the following function:

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

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





Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Harmonic	lin-ln evals-error	.	.	$5.358 \cdot 10^4$	0.003851	$2.176 \cdot 10^{-5}$	0.00346
Romberg	lin-ln evals-error	.	.	$7.348 \cdot 10^4$	0.005727	$2.071 \cdot 10^{-5}$	0.005124
Bulirsch	lin-ln evals-error	.	.	$5.763 \cdot 10^4$	0.0003158	$2.647 \cdot 10^{-5}$	0.0002409
Harmonic	lin-ln steps-error	.	.	$6.694 \cdot 10^4$	0.0002388	$3.364 \cdot 10^{-5}$	0.0002515
Romberg	lin-ln steps-error	1.581	0.0005141	1.046	$5.395 \cdot 10^{-5}$	0.9983	$4.956 \cdot 10^{-6}$
Bulirsch	lin-ln steps-error	0.375	0.03054	0.5595	0.00836	0.9807	0.000633

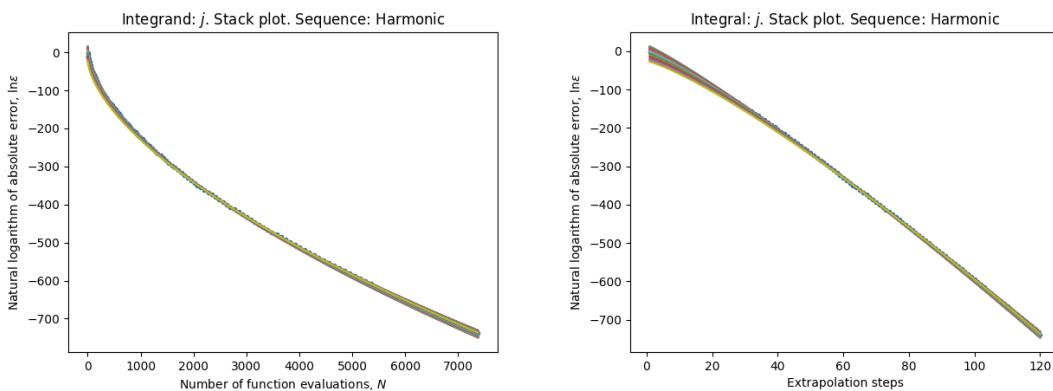
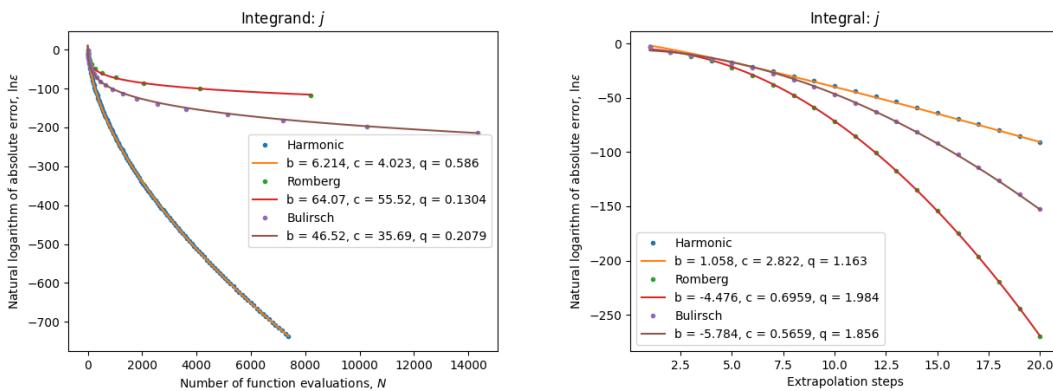
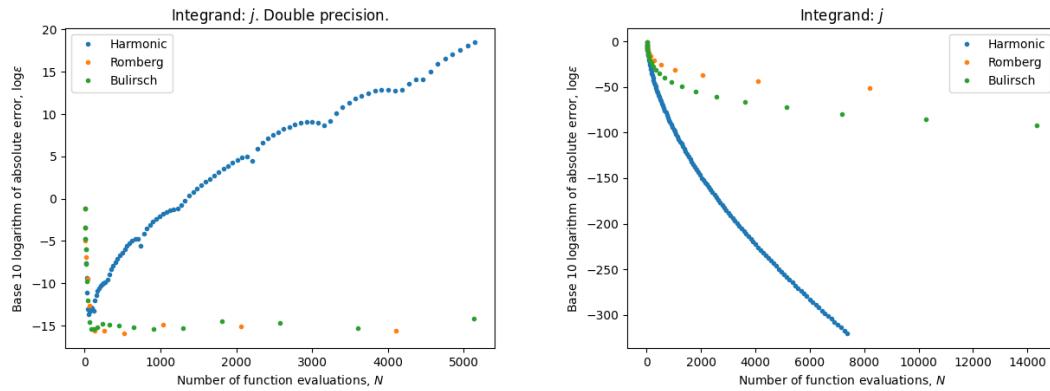
We see that we do not get high precision using double precision arithmetic, independent of sequence. The Romberg and Bulirsch sequence seem to perform similarly well but the harmonic sequence seems to be slowest.

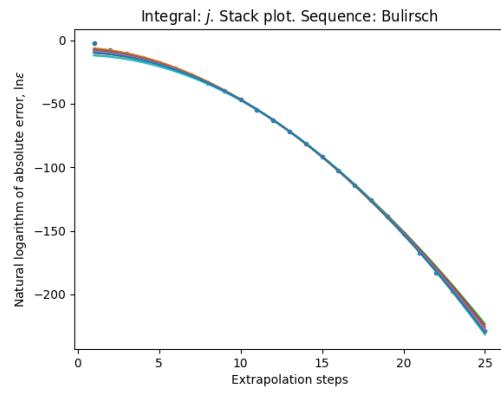
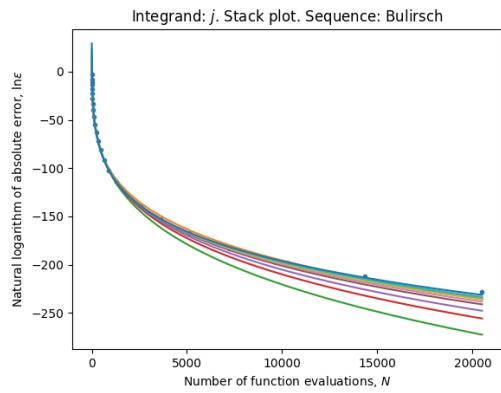
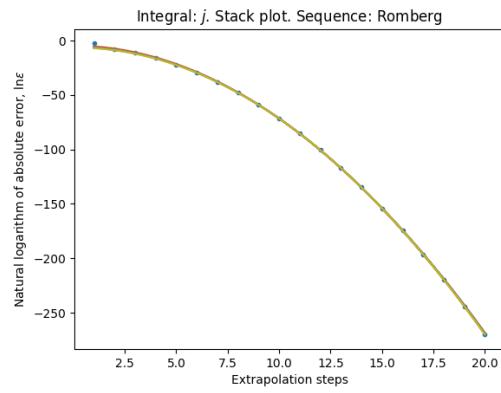
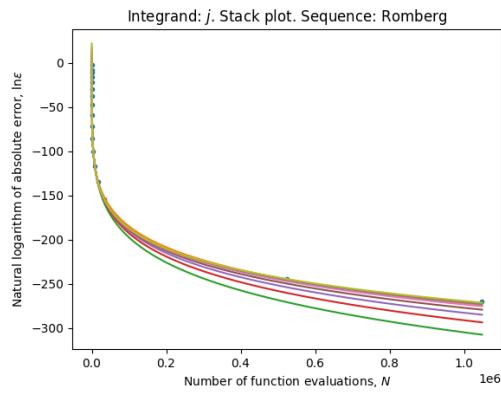
For the harmonic sequence we reject the fitting because we get unreasonale values of the parameters. The model for exponential convergence in the number of steps fits well in the case of the Romberg sequence and Bulirsch.

2.2.5 Gaussian

Finally we will consider the Gaussian function

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





Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Harmonic	lin-ln evals-error	$1.037 \cdot 10^9$	5.523	4.257	0.02142	0.5814	0.0007265
Romberg	lin-ln evals-error	$1.398 \cdot 10^{55}$	6	68.42	0.08326	0.1248	0.02265
Bulirsch	lin-ln evals-error	$1.356 \cdot 10^{51}$	8	49.55	0.1212	0.1931	0.02978
Harmonic	lin-ln steps-error	$6.922 \cdot 10^5$	5.362	2.939	0.02212	1.157	0.0006628
Romberg	lin-ln steps-error	0.003928	0.1462	0.659	0.0008259	2	$2.366 \cdot 10^{-5}$
Bulirsch	lin-ln steps-error	0.0003995	1.479	0.4704	0.02518	1.913	0.0007081

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 performs 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.

Integrand	Sequence	b	c	q
f	Harmonic	15.66	3.1537	0.63887
f	Romberg	62.881	48.203	0.13887
f	Bulirsch	46.309	29.549	0.22556
$g_{10^{-2}}$	Harmonic	5.7088	-0.8668	0.52546
$g_{10^{-2}}$	Romberg	13.538	3.0588	0.25221
$g_{10^{-2}}$	Bulirsch	9.1445	0.49433	0.43743
$g_{10^{-1}}$	Harmonic	2.2352	-0.66129	0.51817
$g_{10^{-1}}$	Romberg	20.725	10.998	0.19552
$g_{10^{-1}}$	Bulirsch	10.844	2.8849	0.34731
g_1	Harmonic	1.6178	1.823	0.49467
g_1	Romberg	43.807	34.715	0.14465
g_1	Bulirsch	24.613	15.795	0.24492
$h_{10^{-4}}$	Harmonic	33.436	31.879	0.030738
$h_{10^{-4}}$	Romberg	1.5634	1.6161	0.23927
$h_{10^{-4}}$	Bulirsch	7.0462	5.7755	0.13169
$h_{10^{-2}}$	Harmonic	-0.19426	1.1078	0.37602
$h_{10^{-2}}$	Romberg	9.8594	7.1092	0.20823
$h_{10^{-2}}$	Bulirsch	2.2519	2.0217	0.34203
h_1	Harmonic	2.052	4.6543	0.4931
h_1	Romberg	55.957	50.318	0.1287
h_1	Bulirsch	35.525	29.752	0.20461
i	Harmonic	54099	54099	$2.2756 \cdot 10^{-5}$
i	Romberg	70897	70896	$2.1811 \cdot 10^{-5}$
i	Bulirsch	58074	58073	$2.6538 \cdot 10^{-5}$
j	Harmonic	6.2138	4.0228	0.58595
j	Romberg	64.075	55.521	0.13037
j	Bulirsch	46.521	35.69	0.20788

Table 2.1: Optimal parameters by test case

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

Integrand	Sequence	b	c	q
f	Harmonic	10.466	2.1696	1.2654
f	Romberg	1.8624	0.55261	2.0602
f	Bulirsch	0.77673	0.41734	1.9549
$g_{10^{-2}}$	Harmonic	6.5675	-0.61916	1.0458
$g_{10^{-2}}$	Romberg	7.403	0.006235	3.2075
$g_{10^{-2}}$	Bulirsch	7.3047	0.00063882	3.3913
$g_{10^{-1}}$	Harmonic	2.8699	-0.47343	1.0317
$g_{10^{-1}}$	Romberg	3.6964	0.059489	2.6042
$g_{10^{-1}}$	Bulirsch	3.9142	0.012441	2.7293
g_1	Harmonic	0.034332	1.3144	0.98632
g_1	Romberg	-0.86887	0.37786	2.1086
g_1	Bulirsch	-1.3077	0.18952	2.0725
$h_{10^{-4}}$	Harmonic	12.604	11.571	0.12559
$h_{10^{-4}}$	Romberg	-1.5468	0.0040324	3.0761
$h_{10^{-4}}$	Bulirsch	0.16861	0.15206	1.4061
$h_{10^{-2}}$	Harmonic	-0.80722	0.82135	0.75792
$h_{10^{-2}}$	Romberg	-1.8118	0.030909	2.74
$h_{10^{-2}}$	Bulirsch	-2.6424	0.0083341	2.7259
h_1	Harmonic	-1.9642	3.3575	0.98328
h_1	Romberg	-5.8254	0.64611	1.9664
h_1	Bulirsch	-7.6558	0.48664	1.8348
i	Harmonic	67160	67160	$3.3808 \cdot 10^{-5}$
i	Romberg	1.3857	1.3726	0.92029
i	Bulirsch	0.95043	1.2669	0.7645
j	Harmonic	1.0579	2.8215	1.1626
j	Romberg	-4.4762	0.69593	1.9838
j	Bulirsch	-5.7837	0.56594	1.8564

Table 2.2: Optimal parameters by test case

Chapter 3

Extrapolation of difference quotients

3.1 The algorithm

Let $a \in \mathbb{R}$, $\varepsilon > 0$ and $f :]a - \varepsilon, a + \varepsilon[\rightarrow \mathbb{R}$ 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

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

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

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

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

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

where

$$D_f(h) := \frac{f(a + h) - f(a - h)}{2h} \quad (3.4)$$

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

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

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

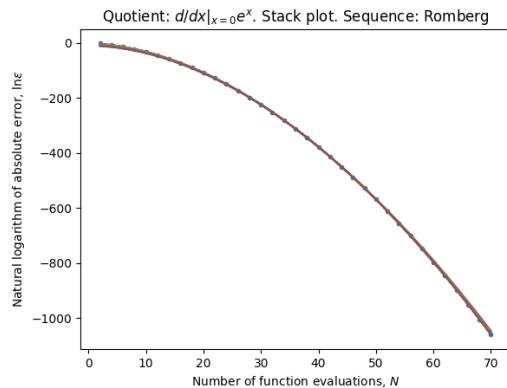
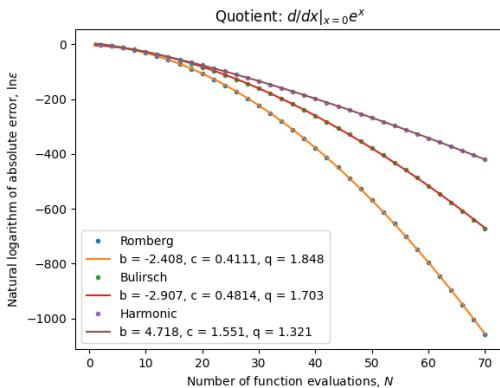
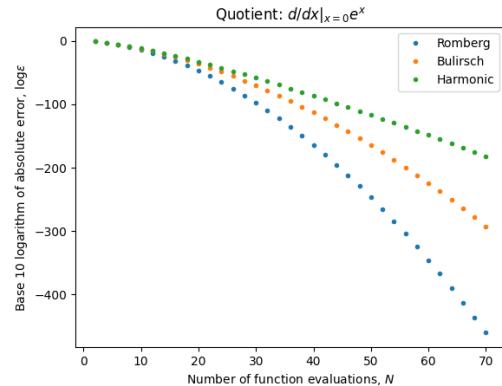
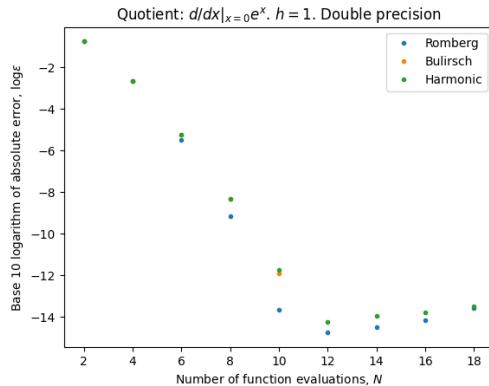
3.2 Numerical experiments

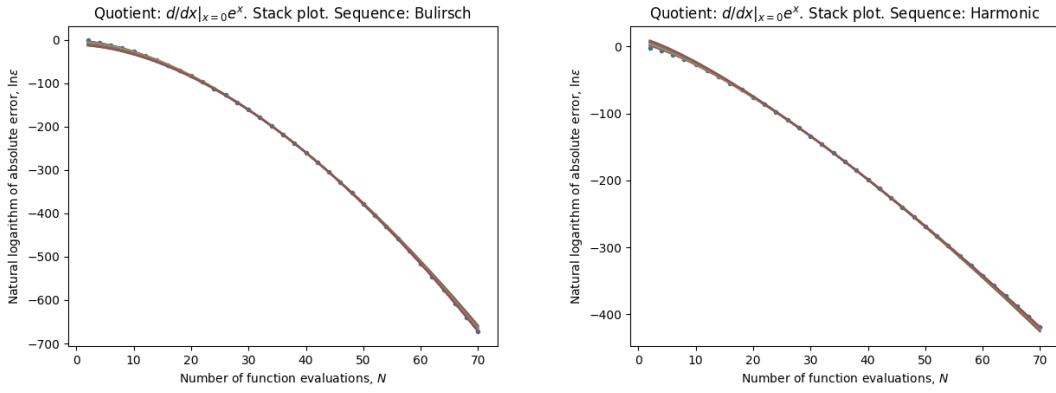
In this section we are going to extrapolate the symmetric difference quotient for approximating the derivative of a function at a given point. Let $h > 0$ be some number, $f :]a - \varepsilon, a + \varepsilon[\rightarrow \mathbb{R}$ a function differentiable at a and $n_1 < n_2 < \dots$ a sequence of 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 $(h_1^2, D_f(h_1)), (h_2^2, D_f(h_2)), \dots$, 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.





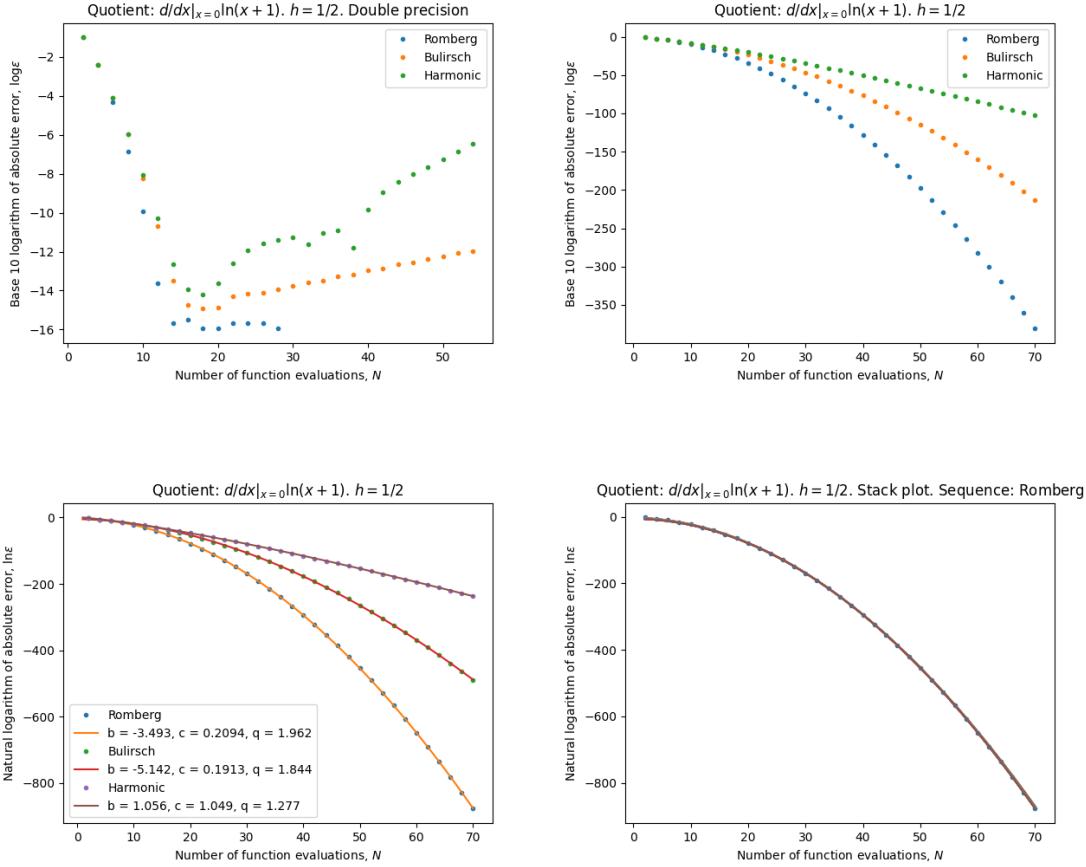
Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-ln evals-error	0.03687	2.584	0.4004	0.004019	1.853	$6.702 \cdot 10^{-5}$
Bulirsch	lin-ln evals-error	0.01953	3.508	0.4544	0.0119	1.716	0.0002258
Harmonic	lin-ln evals-error	$1.122 \cdot 10^5$	3.293	1.732	0.007202	1.299	0.0002359

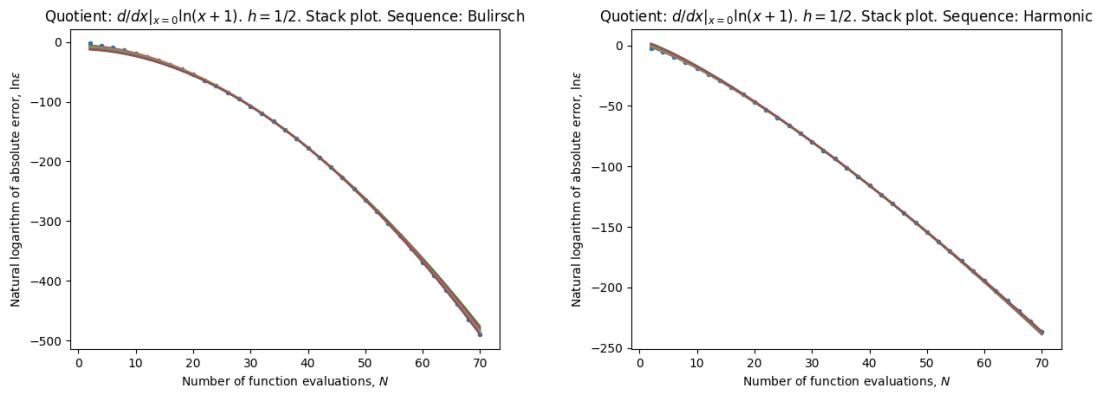
In standard floating point arithmetic, we get down to machine level precision using any sequence. The Romberg sequence works best, then Bulirsch and then the harmonic. The model seems to fit well in all cases.

3.2.2 Logarithm

Now we will consider the derivative at zero of the function

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





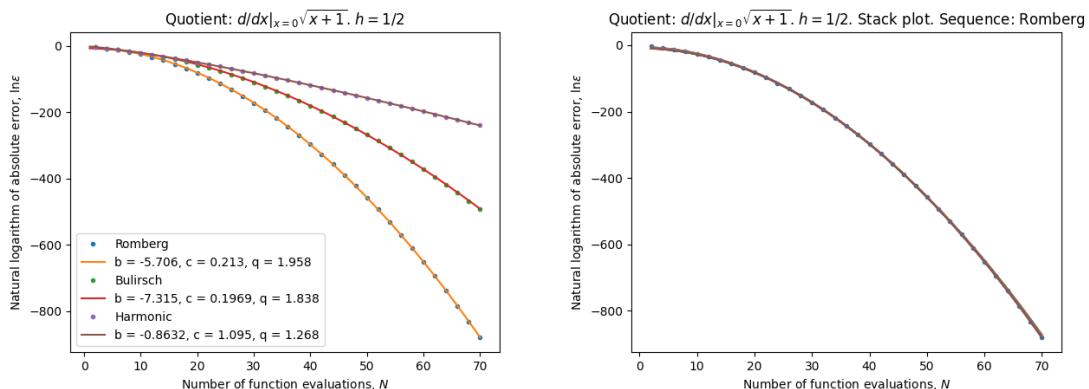
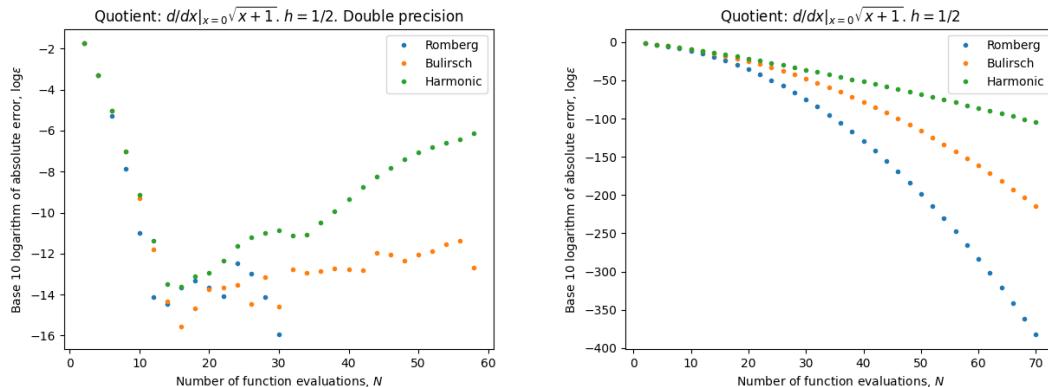
Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-lin evals-error	0.009098	0.9823	0.2041	0.001354	1.967	$2.052 \cdot 10^{-5}$
Bulirsch	lin-lin evals-error	0.001179	3.144	0.1757	0.01703	1.863	0.0002721
Harmonic	lin-lin evals-error	27.62	0.9622	1.126	0.003487	1.263	0.0001174

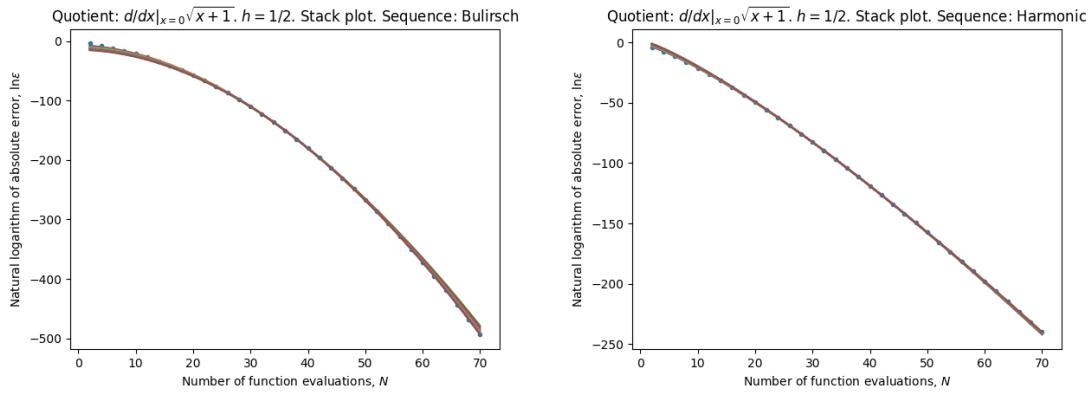
We get down to machine level precision using any sequence, Romberg performs best, then Bulirsch and then the Harmonic one. The model fits well in all cases.

3.2.3 Square root

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

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





Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin- \ln evals-error	0.0008261	1.263	0.2064	0.001755	1.965	$2.659 \cdot 10^{-5}$
Bulirsch	lin- \ln evals-error	0.0001188	3.504	0.1792	0.01927	1.859	0.0003075
Harmonic	lin- \ln evals-error	2.598	0.8154	1.155	0.002659	1.257	$8.929 \cdot 10^{-5}$

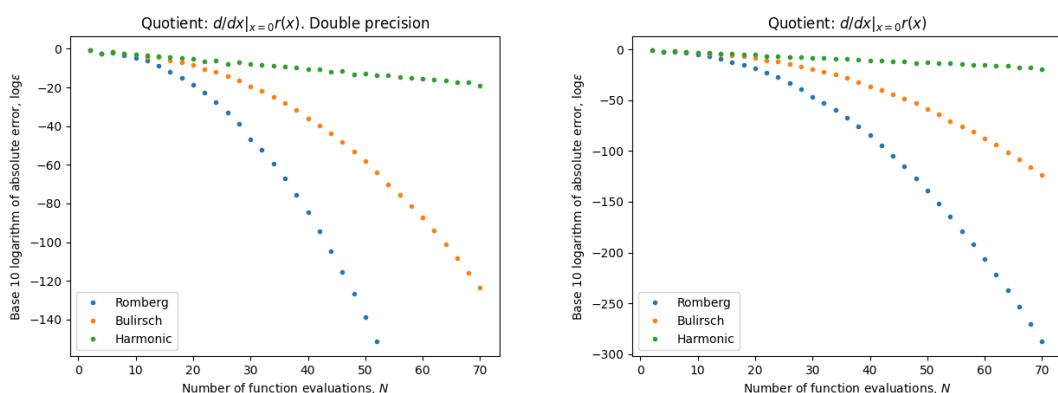
In standard double precision floating point arithmetic we get down to machine level precision using any sequence. The model fits well in all cases.

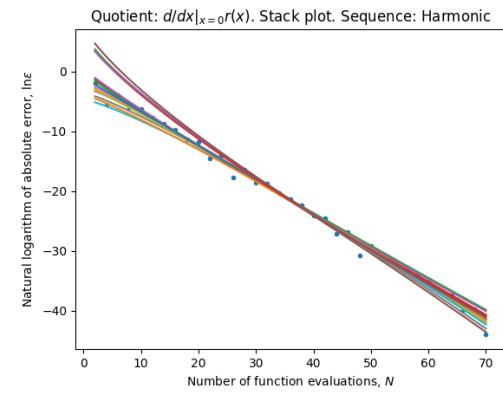
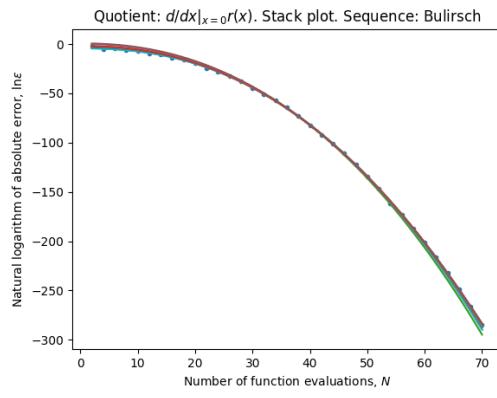
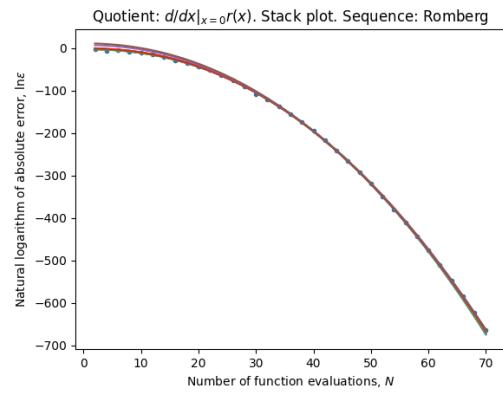
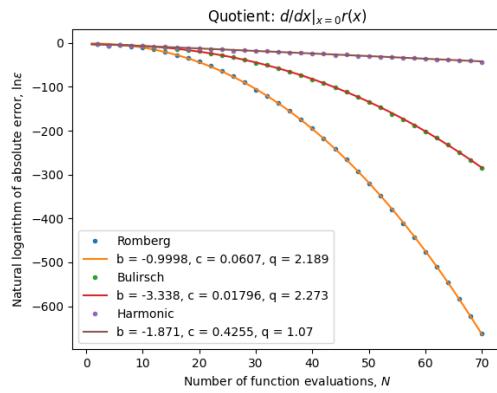
3.2.4 Smooth but not analytic function

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	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	7305	12.55	0.06014	0.02339	2.195	0.0002104
Bulirsch	lin-ln evals-error	0.2728	4.406	0.0178	0.03004	2.281	0.000316
Harmonic	lin-ln evals-error	287.5	4.85	0.818	0.4753	0.9831	0.02252

Romberg performs best and the harmonic sequence worst.

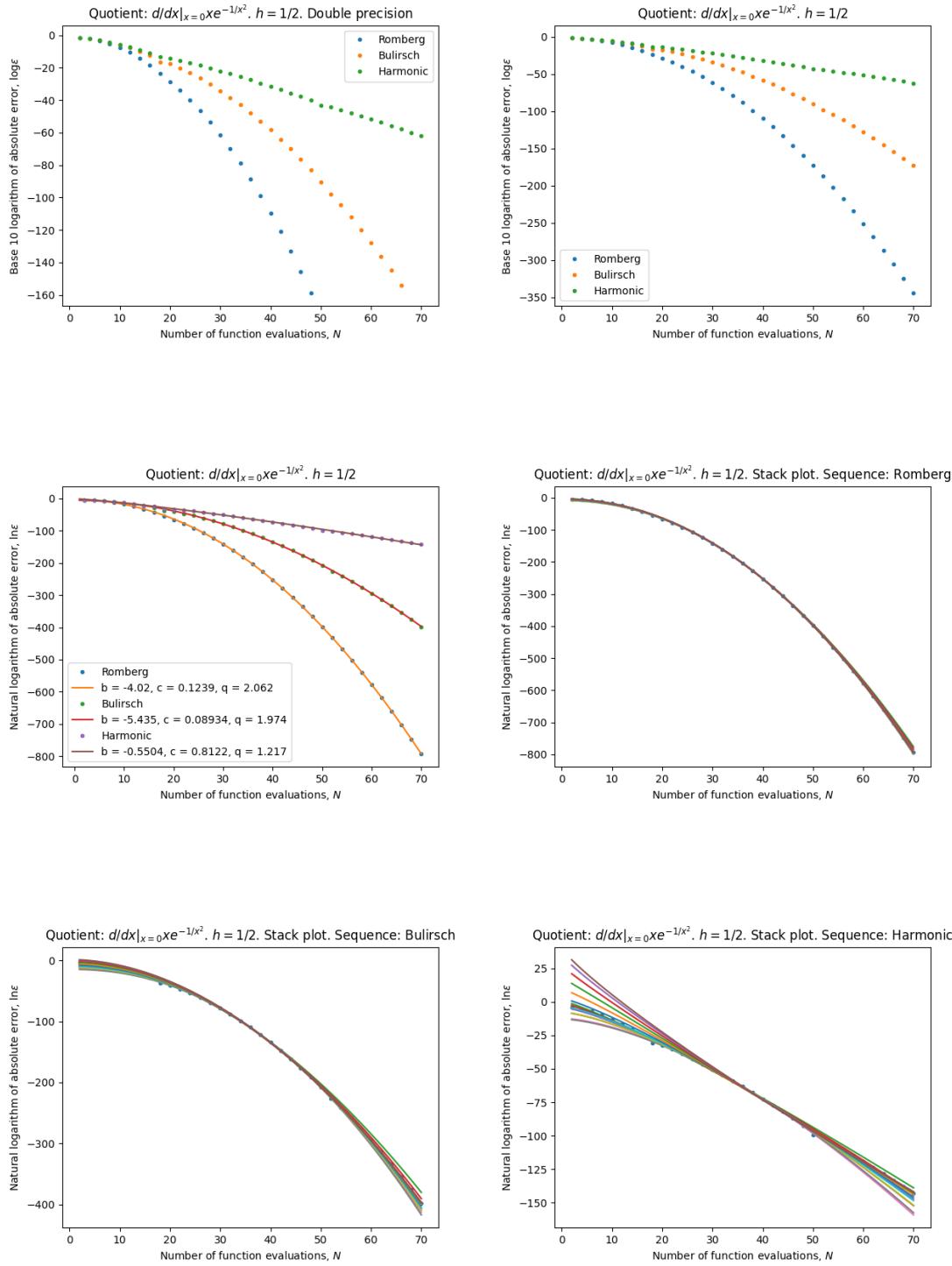
The model seems to fit reasonably well for the Romberg and Bulirsch sequence but the fitting is not so nice for the harmonic sequence.

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	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	0.009202	1.693	0.1182	0.00696	2.073	$9.612 \cdot 10^{-5}$
Bulirsch	lin-ln evals-error	0.4381	5.98	0.07965	0.09556	2.015	0.001606
Harmonic	lin-ln evals-error	$2.061 \cdot 10^{16}$	12.84	1.483	1.04	1.216	0.03736

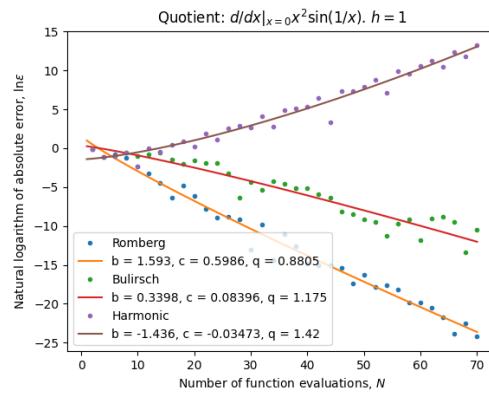
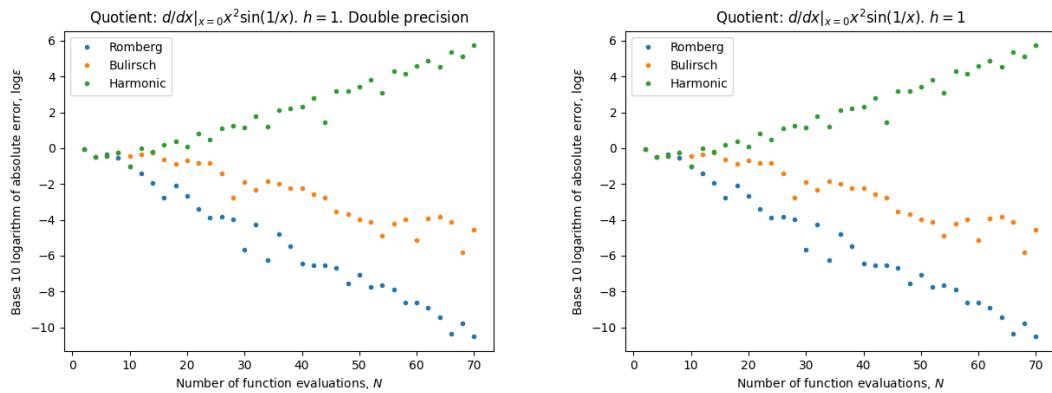
Here Romberg performs best and the harmonic sequence worst.

We seem to have nice fit for the Romberg sequence, moderately good for the Bulirsch 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	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$1.99 \cdot 10^{27}$	13	6.139	3.832	0.8069	0.397
Bulirsch	lin-ln evals-error	.	.	2316	4.918	1.24	0.4409
Harmonic	lin-ln evals-error	0.3968	1.653	-0.3656	6.72	1.307	0.09291

Here the model simply does not fit. Note that we do not have the asymptotic expansion for the derivate here, since the function is only once differentiable.

The parameters from the fitting are:

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

Table 3.1: Optimal parameters by test case

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

Chapter 4

Initial Value Problems

4.1 The explicit midpoint rule

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

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

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

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

and

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

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

```
class ExplicitMidpointRule(Scheme):

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

    def apply(self, ivp, n):
        h = (ivp.b - ivp.a) / (2 * n)
        y_s1 = 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_s1 + 2 * h * ivp.f(ivp.a + i * h, y_l)
            y_s1 = 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 $n_1 < n_2 < \dots$ 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 efficiency 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.

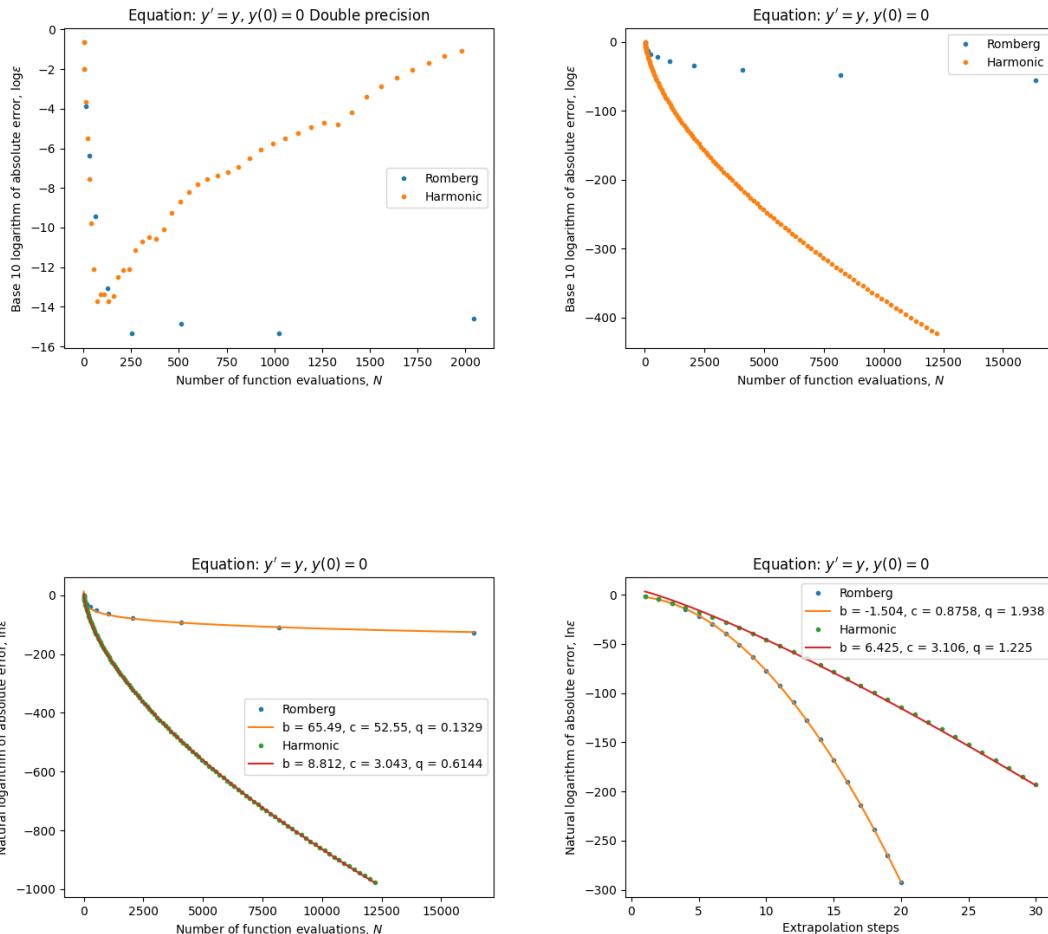
Now we will consider the results of the experiments.

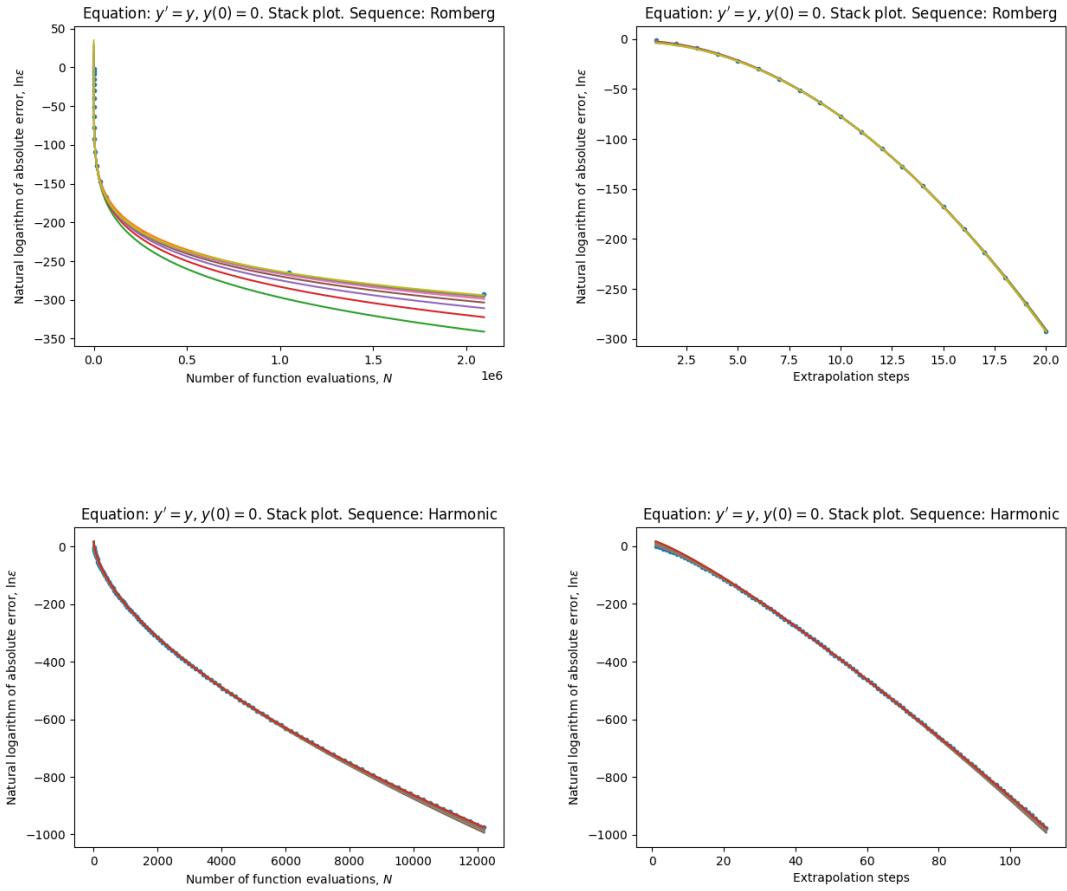
4.2.1 Exponential growth

First we will consider the following initial value problem:

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

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





Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Romberg	lin-ln evals-error	$6.231 \cdot 10^{65}$	6	71.92	0.1174	0.1236	0.03122
Harmonic	lin-ln evals-error	$1.204 \cdot 10^9$	6.365	3.264	0.005221	0.6081	0.0001653
Romberg	lin-ln steps-error	0.0873	0.2223	0.841	0.0009203	1.95	$2.797 \cdot 10^{-5}$
Harmonic	lin-ln steps-error	$3.805 \cdot 10^7$	6.024	3.309	0.004502	1.214	0.0001425

The Harmonic sequence performs 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

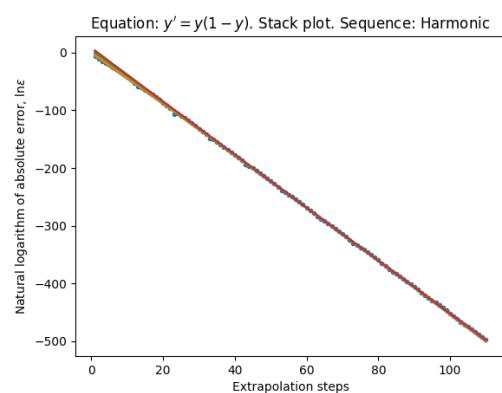
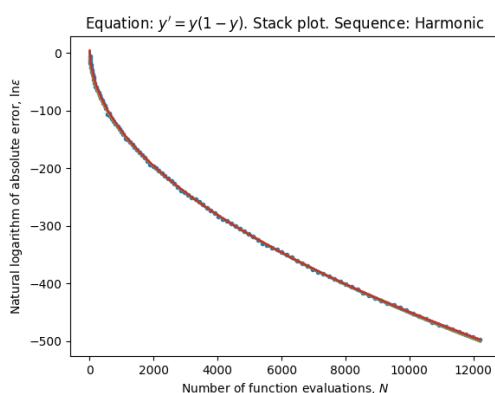
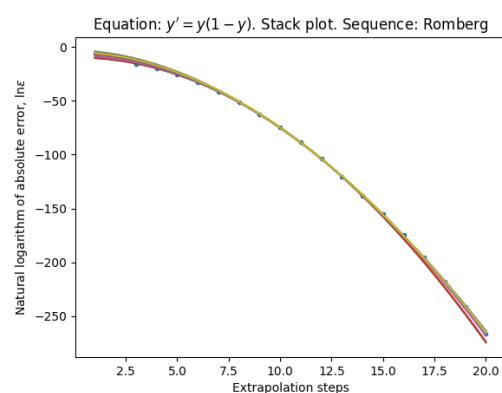
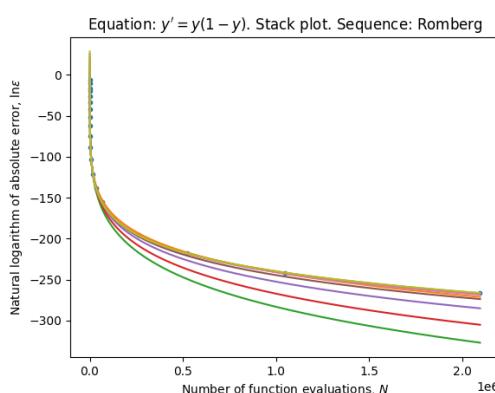
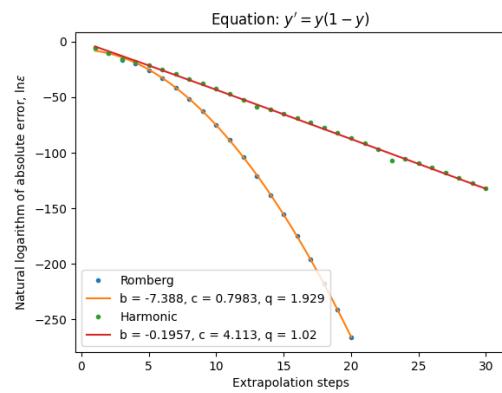
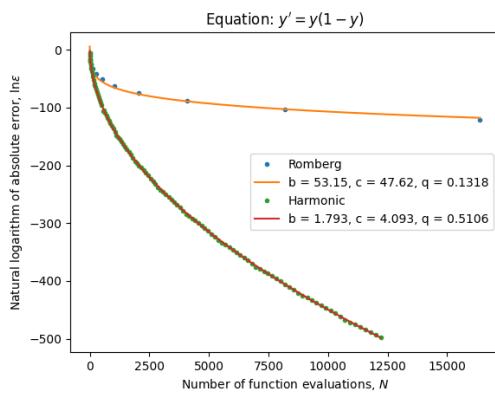
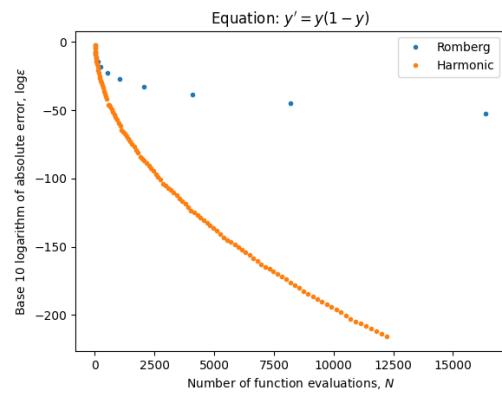
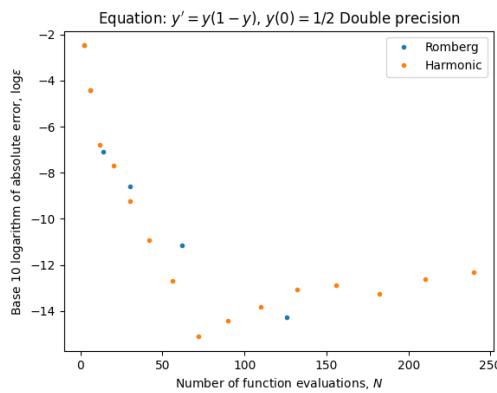
Then we will consider the following initial value problem

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

whose solution is the sigmoid function

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

which is analytic.



Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-lin evals-error	$8.848 \cdot 10^{63}$	6	70.69	0.1952	0.1226	0.0637
Harmonic	lin-lin evals-error	2510	9.384	4.249	0.00357	0.5073	0.0001329
Romberg	lin-lin steps-error	0.01026	1.856	0.814	0.03122	1.932	0.001091
Harmonic	lin-lin steps-error	257.8	9.223	4.256	0.003464	1.014	0.0001294

The harmonic sequence performs better and we get down to machine level precision 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

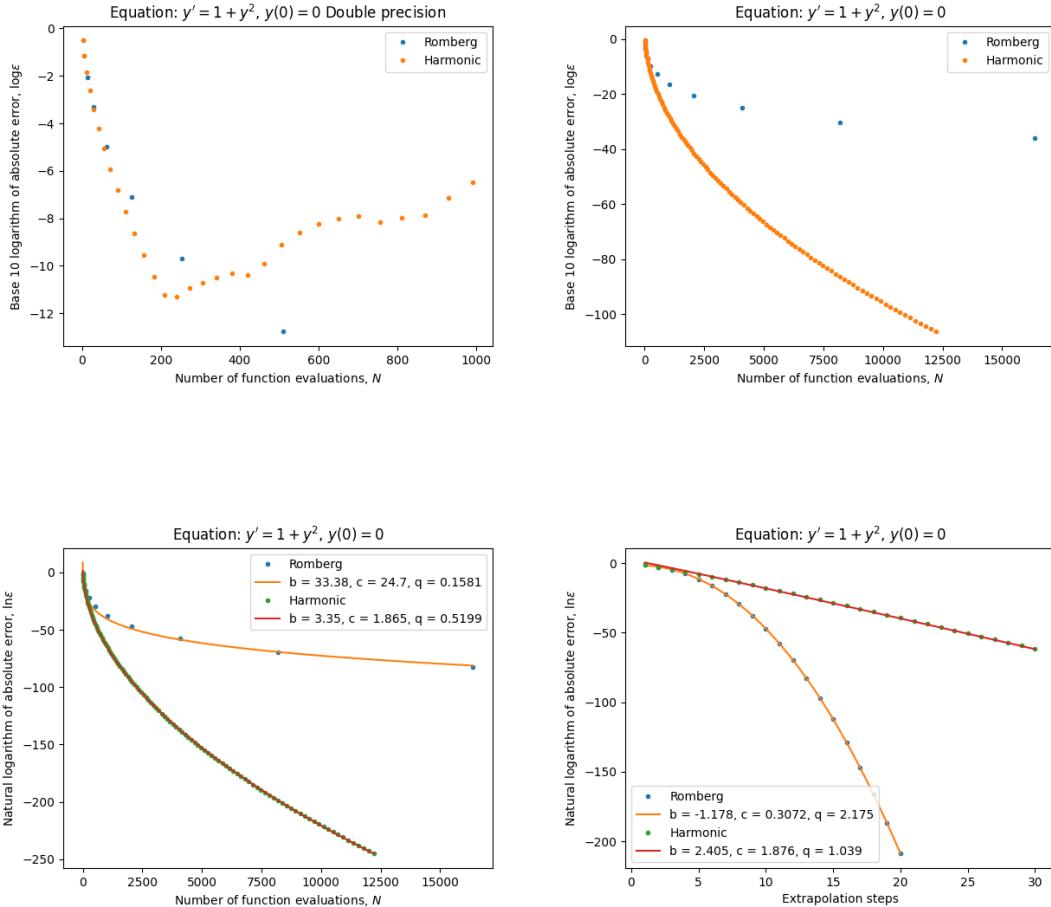
Now we will consider the following equation

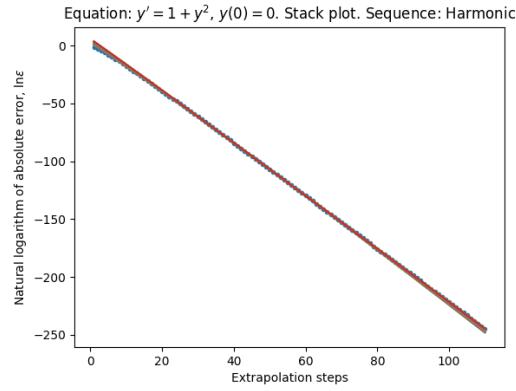
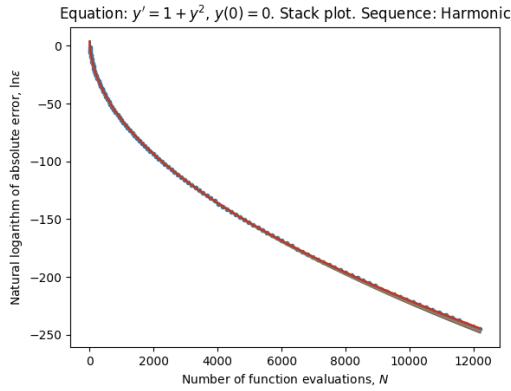
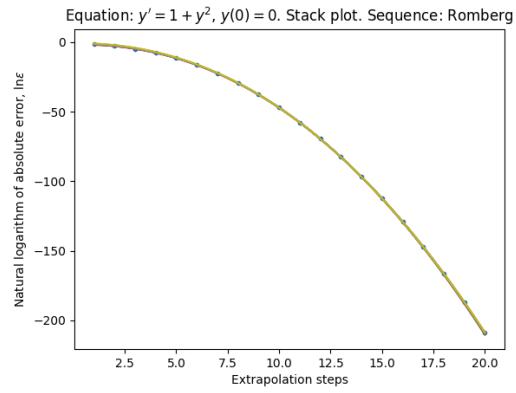
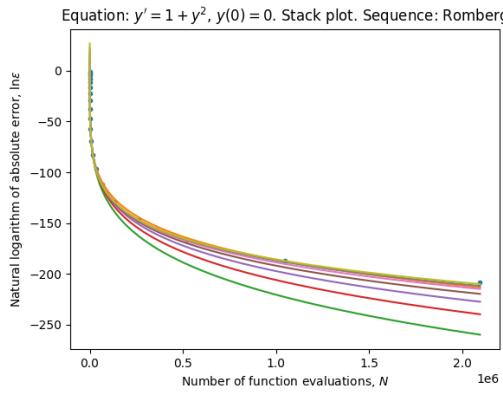
$$y'(x) = 1 + y(x)^2, \quad y(0) = 0, \quad x \in [0, 1] \quad (4.4)$$

whose solution is

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

which is meromorphic and we are quite far from singularities.





Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-lin evals-error	$9.496 \cdot 10^{36}$	6	33.25	0.1689	0.1523	0.03648
Harmonic	lin-lin evals-error	358.8	0.5652	2.017	0.002488	0.5127	0.000109
Romberg	lin-lin steps-error	0.3476	0.1097	0.3039	0.001043	2.18	$2.579 \cdot 10^{-5}$
Harmonic	lin-lin steps-error	121.5	0.5373	2.021	0.002309	1.025	0.0001013

The harmonic sequence performs 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 Equation with singularity

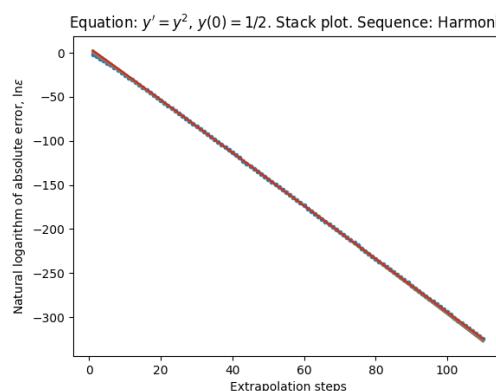
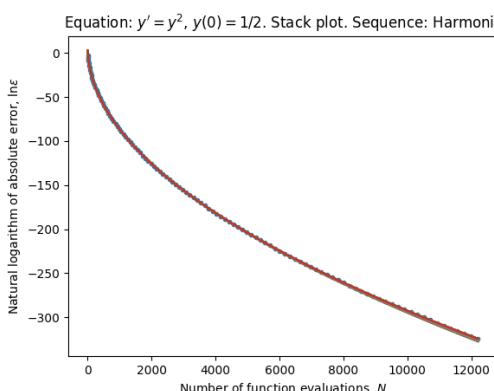
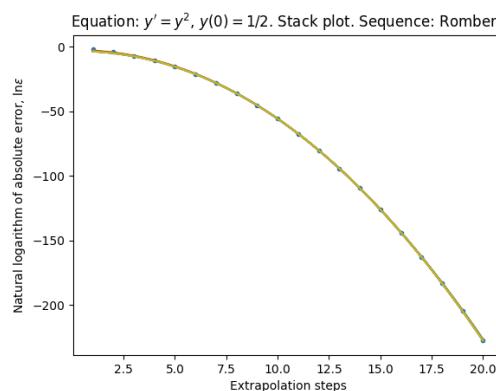
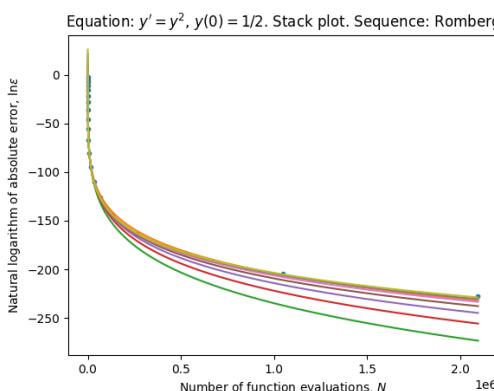
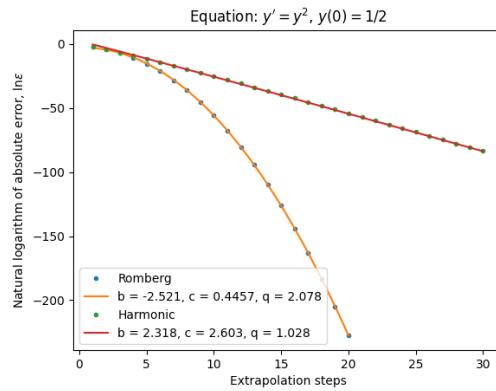
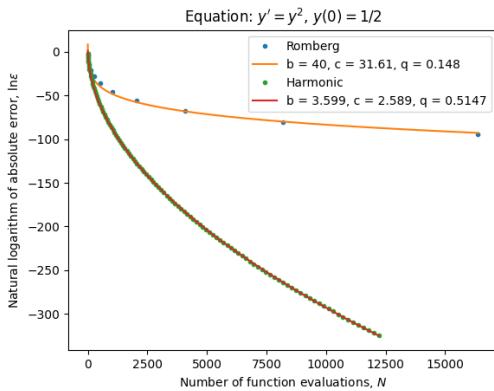
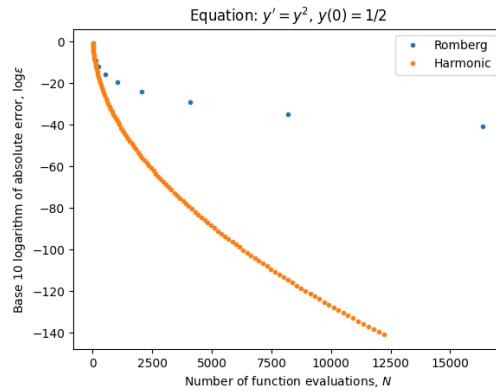
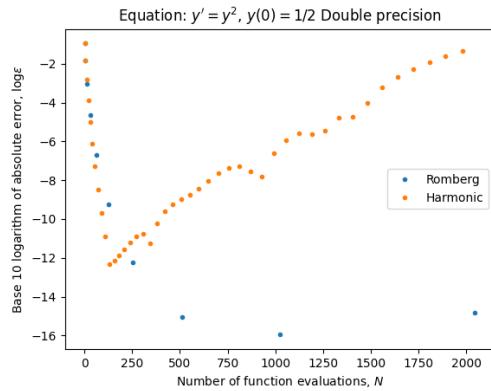
Now we will consider the following initial value problem:

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

whose solution is

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

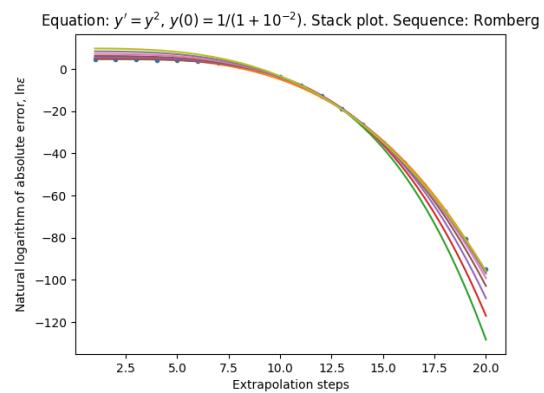
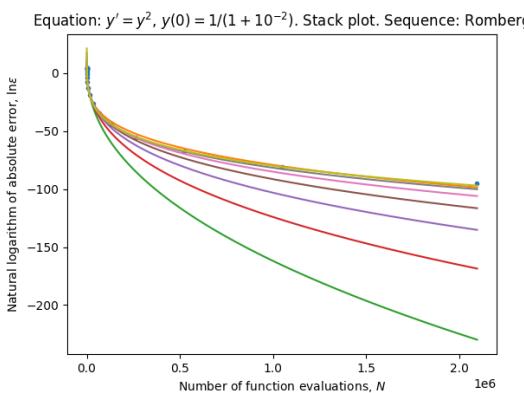
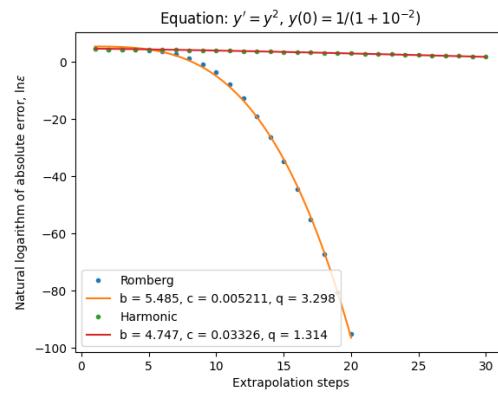
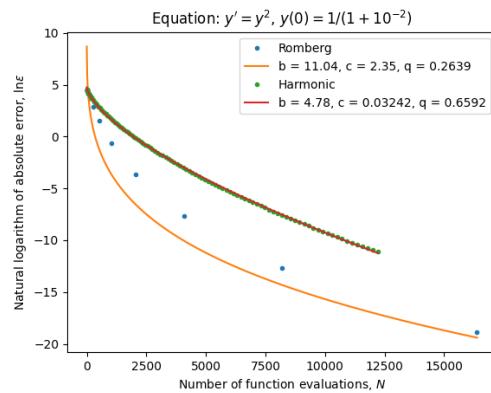
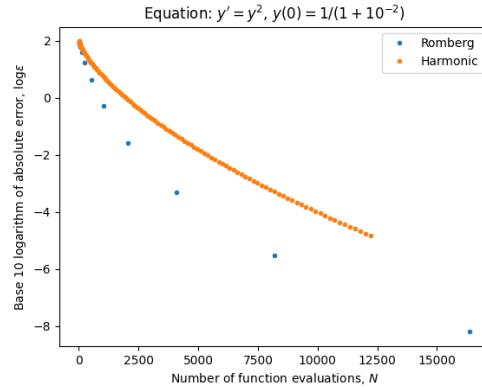
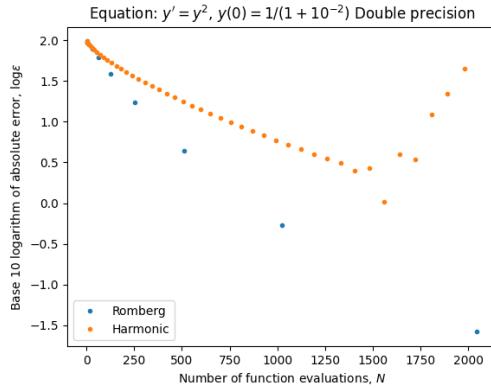
The solution is meromorphic with a pole at $1 + a$.

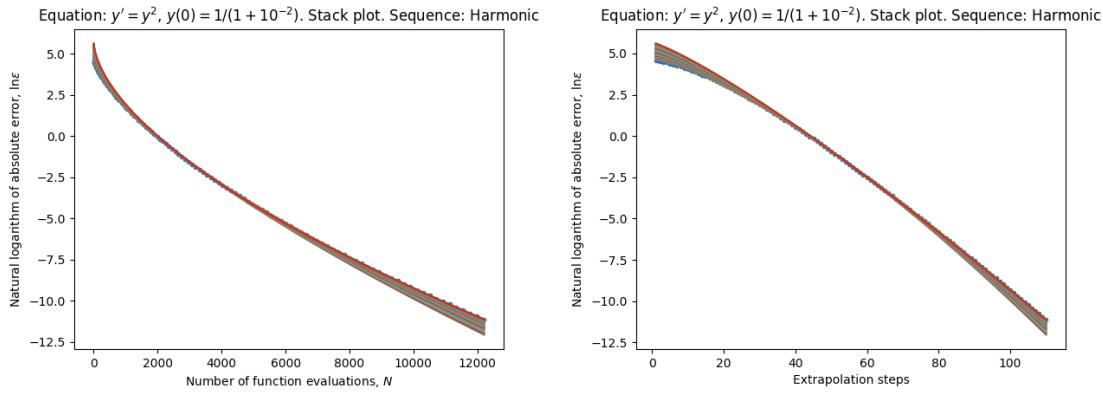


Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-lin evals-error	$1.418 \cdot 10^{42}$	6	42.1	0.1387	0.141	0.03195
Harmonic	lin-lin evals-error	440	0.5196	2.747	0.001256	0.5092	$5.472 \cdot 10^{-5}$
Romberg	lin-lin steps-error	0.04186	0.03016	0.4274	0.0002961	2.091	$8.21 \cdot 10^{-6}$
Harmonic	lin-lin steps-error	104.3	0.4903	2.752	0.001151	1.018	$5.022 \cdot 10^{-5}$

The harmonic sequence performs 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.

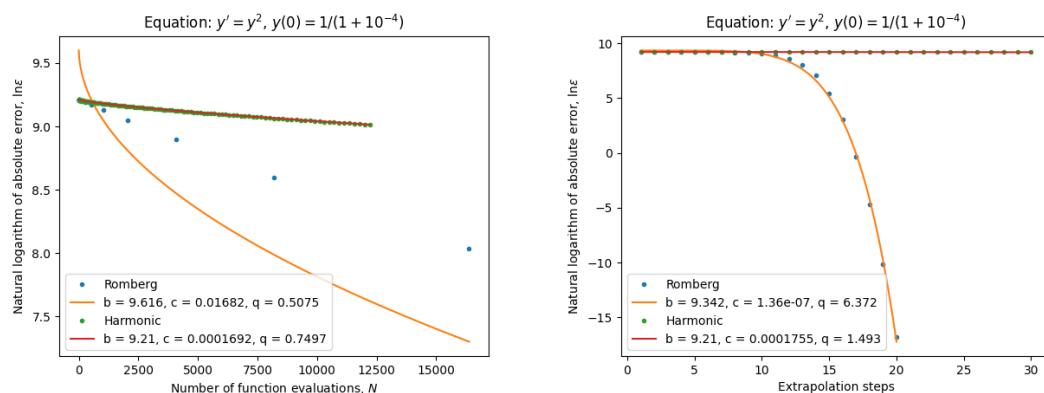
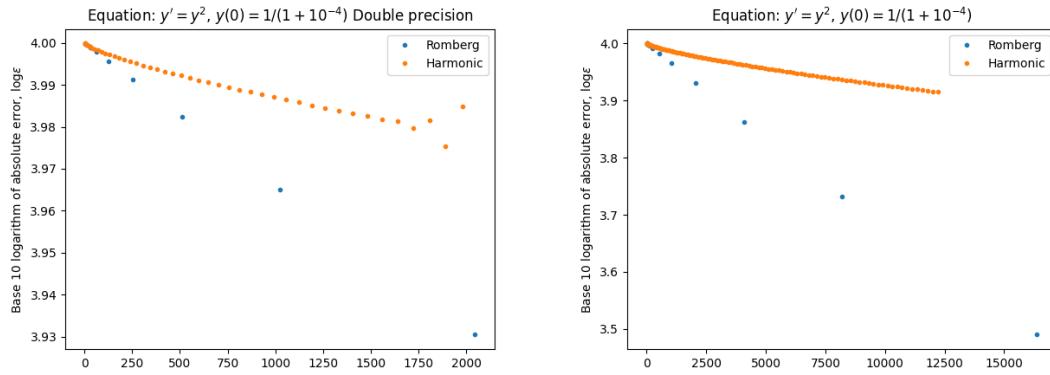


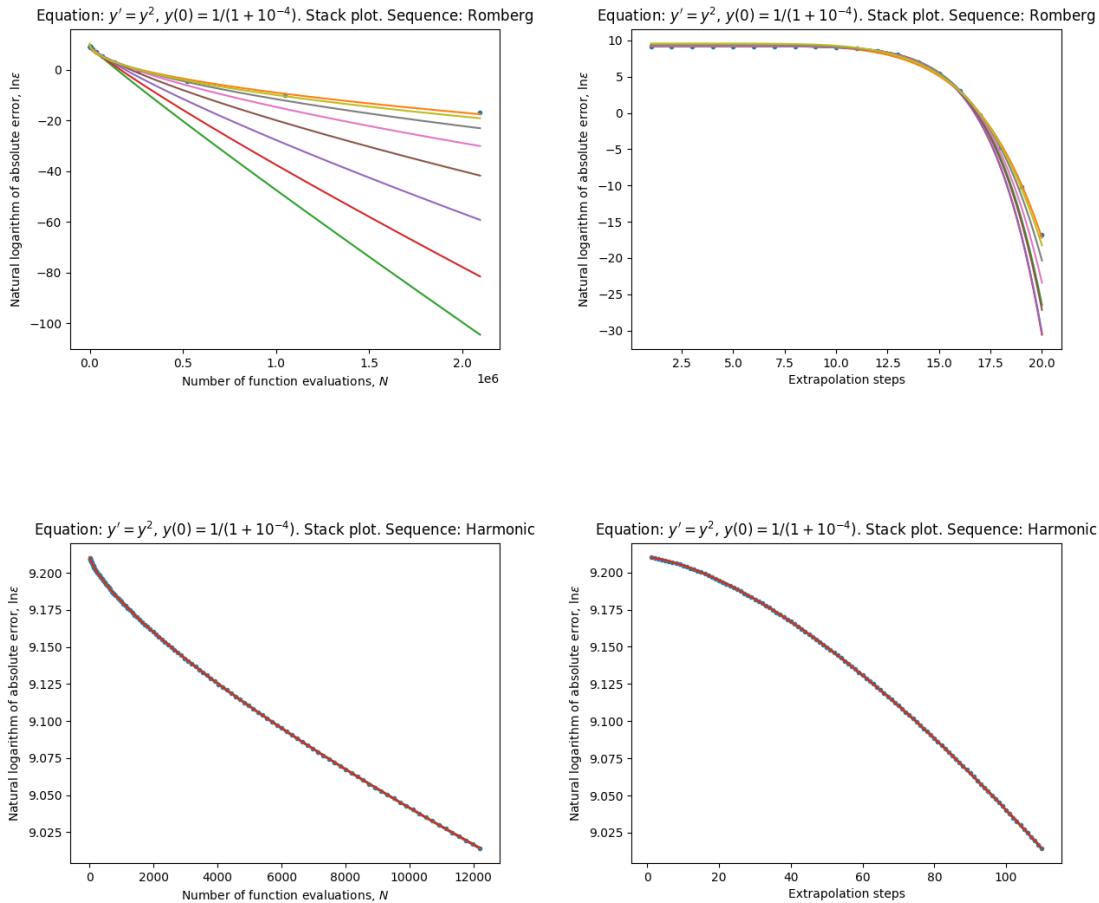


Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Romberg	lin-lin evals-error	$1.249 \cdot 10^{12}$	5.984	2.77	0.7656	0.3067	0.08315
Harmonic	lin-lin evals-error	173.2	0.1158	0.03983	0.08603	0.6452	0.002375
Romberg	lin-lin steps-error	3540	2.885	0.005212	0.6307	3.461	0.00941
Harmonic	lin-lin steps-error	165.2	0.1095	0.04047	0.08154	1.287	0.002248

Here Romberg works better and we do not attain as high precision 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.





Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$1.462 \cdot 10^4$	0.2298	0.004335	2.285	0.7393	0.04196
Harmonic	lin-ln evals-error	$1 \cdot 10^4$	$5.52 \cdot 10^{-9}$	0.0001699	$5.806 \cdot 10^{-5}$	0.7494	$1.194 \cdot 10^{-6}$
Romberg	lin-ln steps-error	$1.087 \cdot 10^4$	0.02206	$1.977 \cdot 10^{-8}$	3.201	7.604	0.006289
Harmonic	lin-ln steps-error	9997	$3.302 \cdot 10^{-10}$	0.0001749	$5.29 \cdot 10^{-6}$	1.494	$1.204 \cdot 10^{-7}$

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.

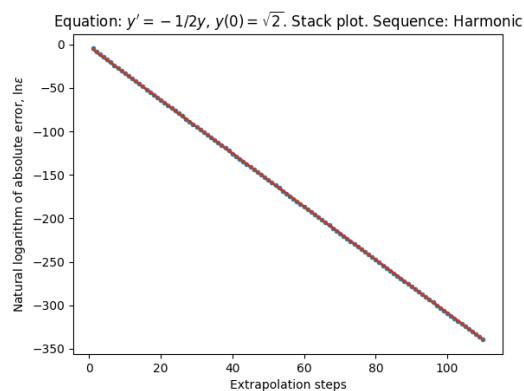
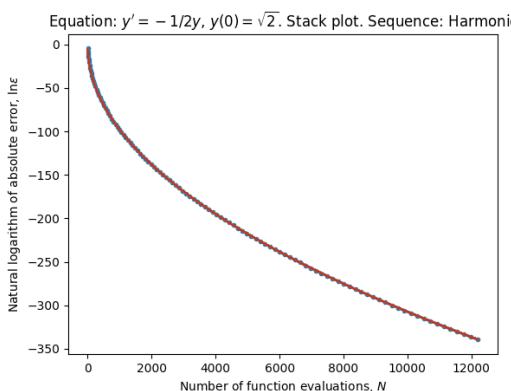
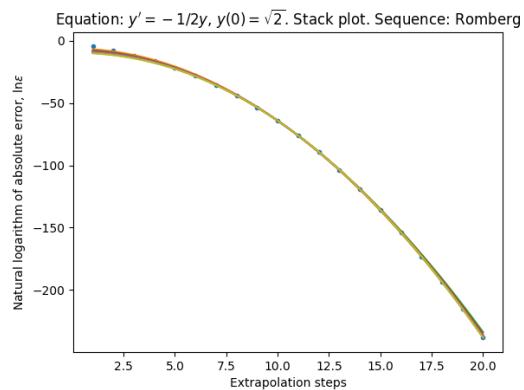
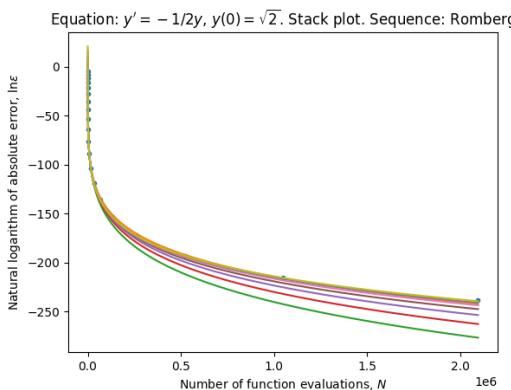
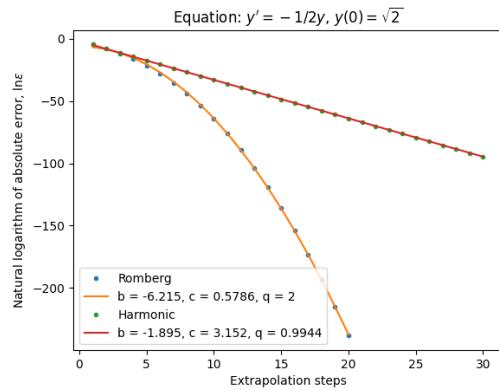
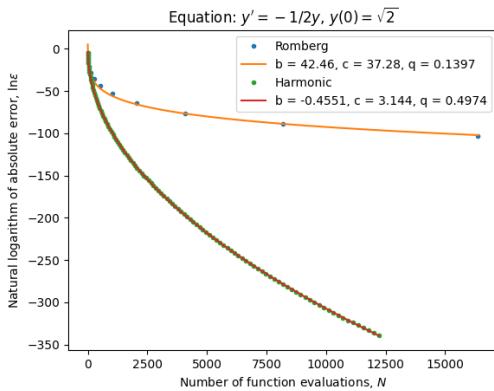
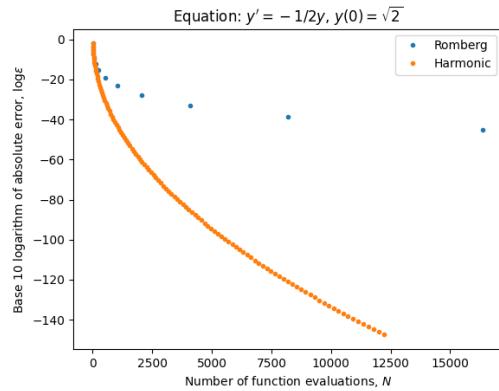
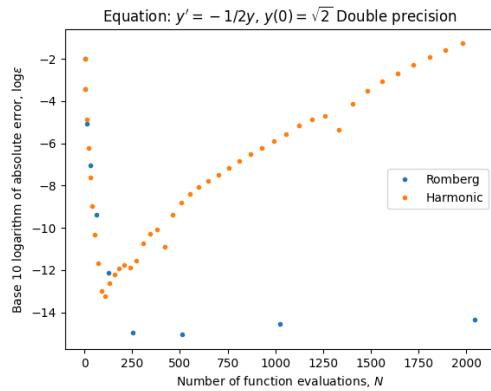
4.2.5 Equation with moderate singularity

Now we will consider the following initial value problem

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

whose solution is

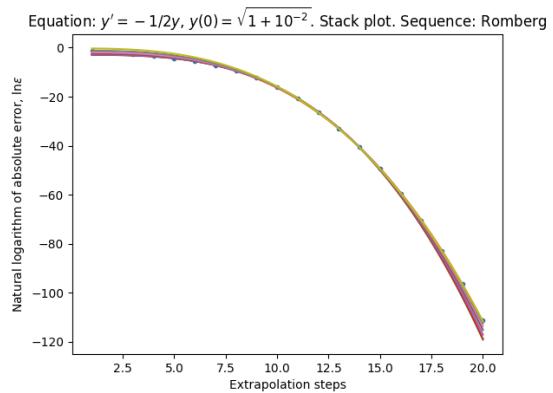
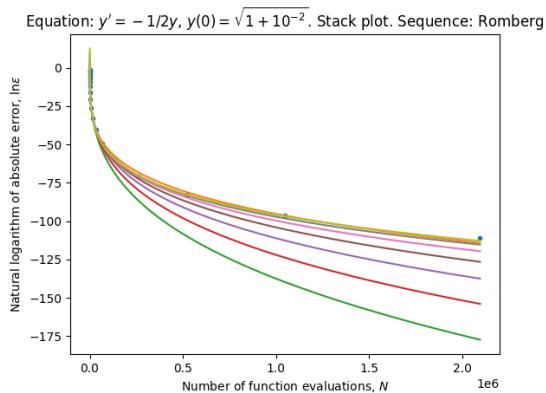
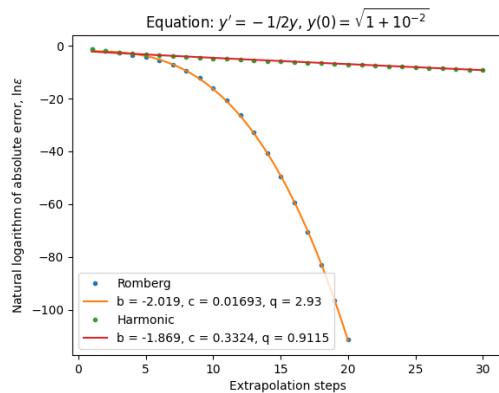
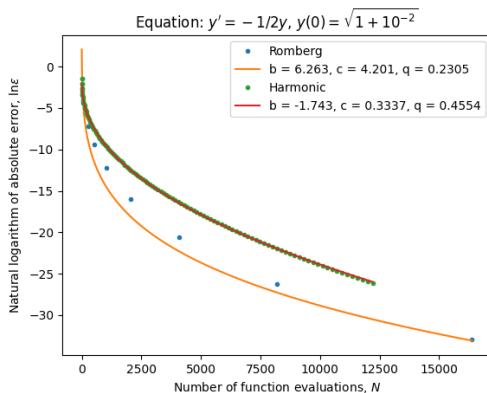
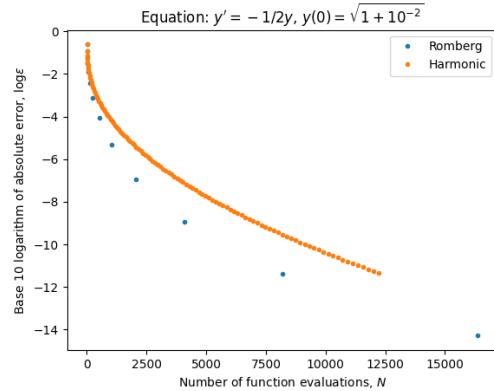
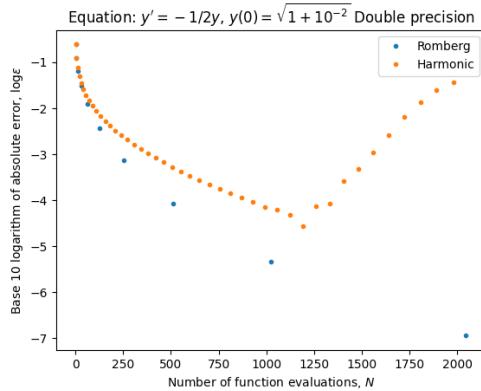
$$y(t) = \sqrt{1 - (t - a)}$$

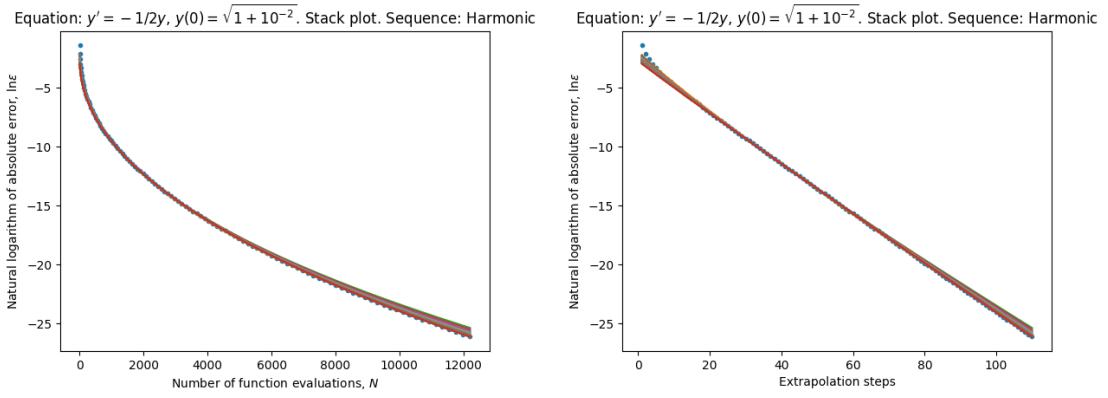


Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-lin evals-error	$8.335 \cdot 10^{41}$	6	47.04	0.1111	0.1343	0.02593
Harmonic	lin-lin evals-error	0.454	0.03631	3.115	$3.797 \cdot 10^{-5}$	0.4983	$1.623 \cdot 10^{-6}$
Romberg	lin-lin steps-error	0.0002771	0.7384	0.5104	0.0058	2.038	0.0001599
Harmonic	lin-lin steps-error	0.1009	0.0414	3.117	$4.412 \cdot 10^{-5}$	0.9965	$1.9 \cdot 10^{-6}$

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.

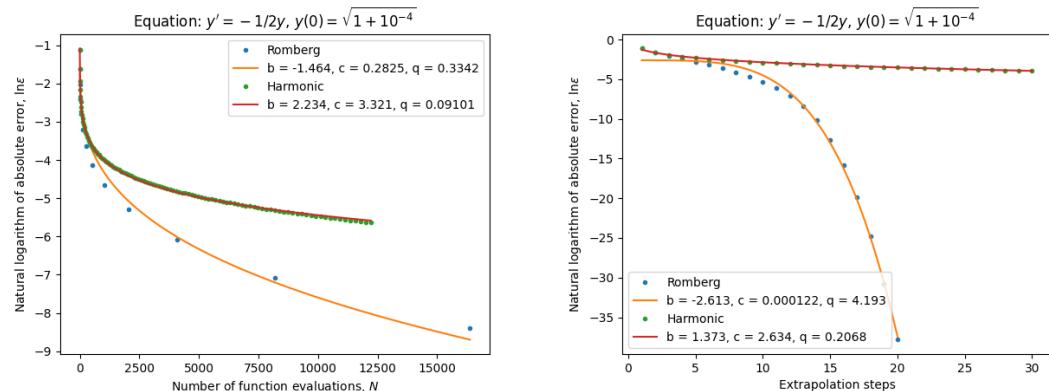
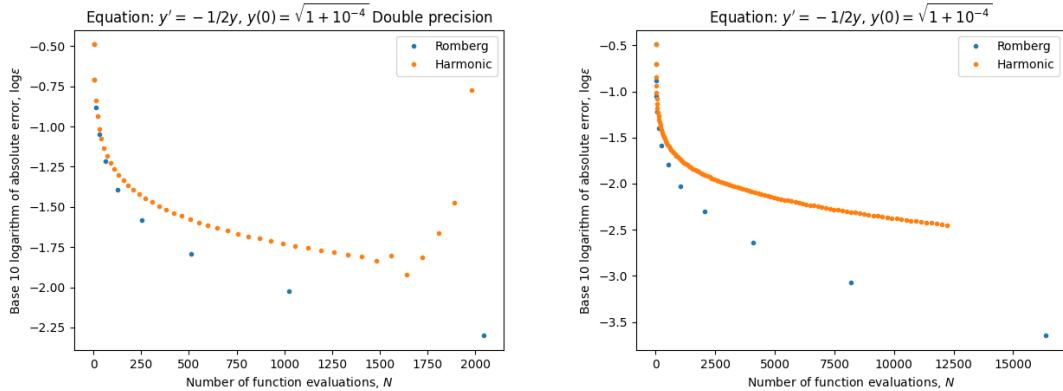


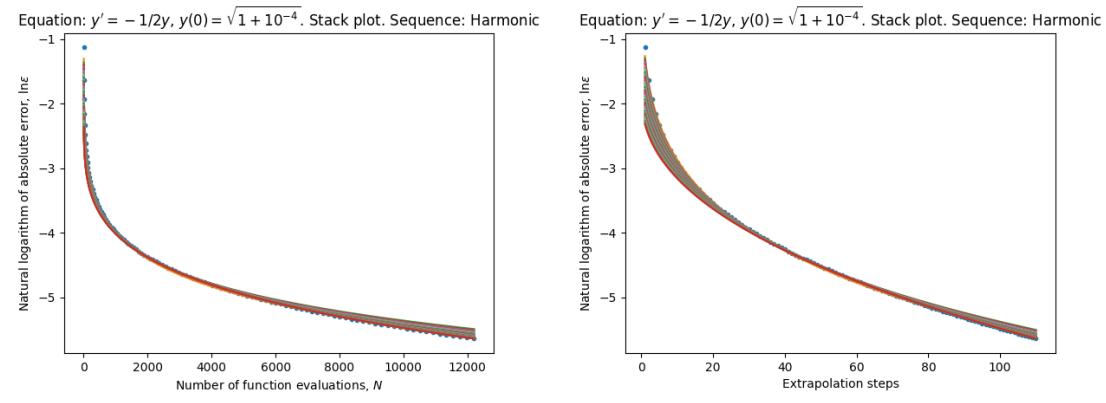
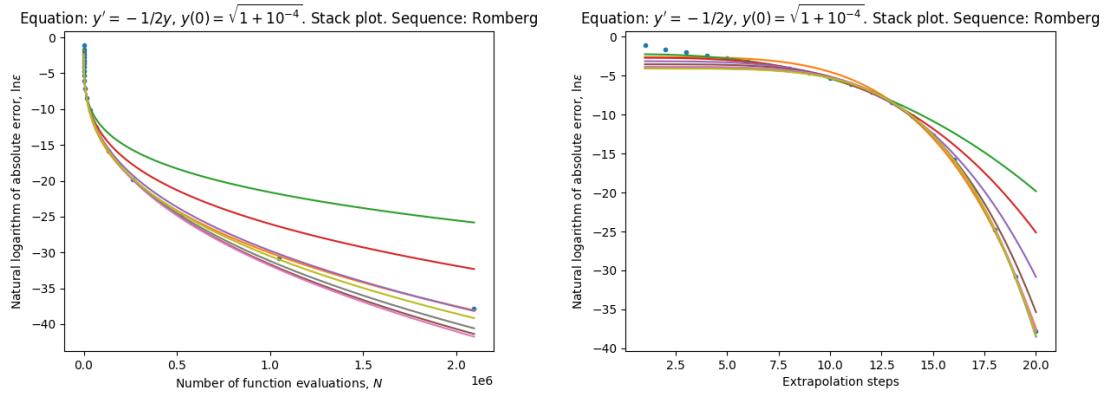


Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-lin evals-error	$3.095 \cdot 10^9$	5.978	4.521	0.4282	0.2537	0.04639
Harmonic	lin-lin evals-error	0.09273	0.0545	0.263	0.01142	0.4785	0.0004549
Romberg	lin-lin steps-error	0.2097	1.241	0.01393	0.1057	3.022	0.001297
Harmonic	lin-lin steps-error	0.08262	0.05039	0.2624	0.01092	0.9574	0.0004388

Here Romberg performs 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.





Sequence	Plot	A -mean	A -var	c -mean	c -var	q -mean	q -var
Romberg	lin-ln evals-error	0.1398	0.4846	0.2489	0.4625	0.3506	0.01881
Harmonic	lin-ln evals-error	1.282	3.403	1.258	0.3894	0.1631	0.05472
Romberg	lin-ln steps-error	0.0429	0.5116	0.001785	3.769	4.002	0.04408
Harmonic	lin-ln steps-error	0.8136	1.714	1.146	0.311	0.3342	0.04634

Here, we do not have any clear fit.

4.2.6 Circular rotation

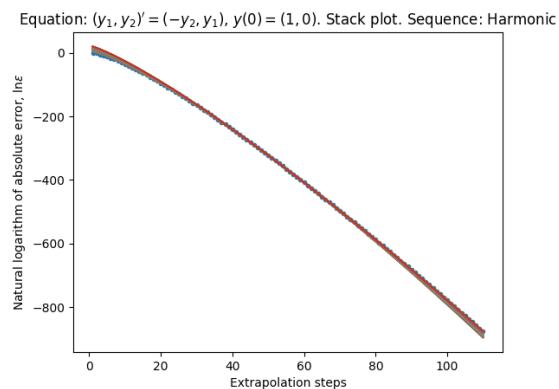
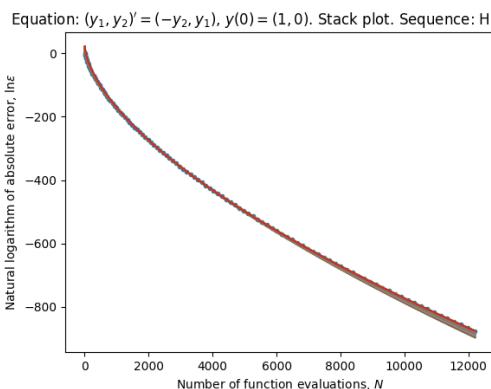
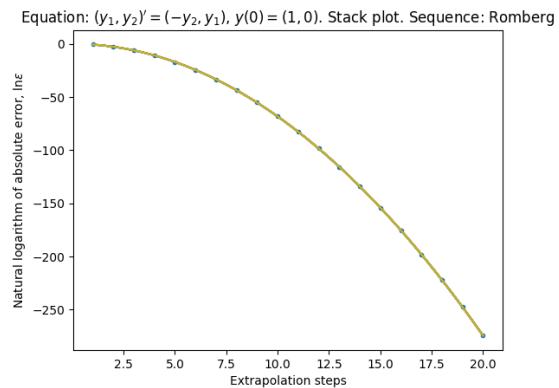
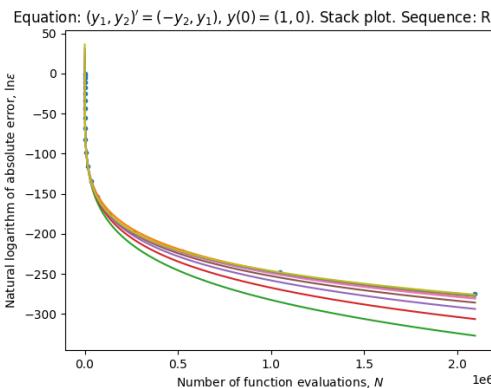
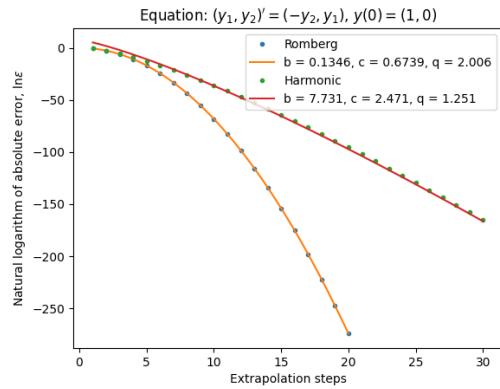
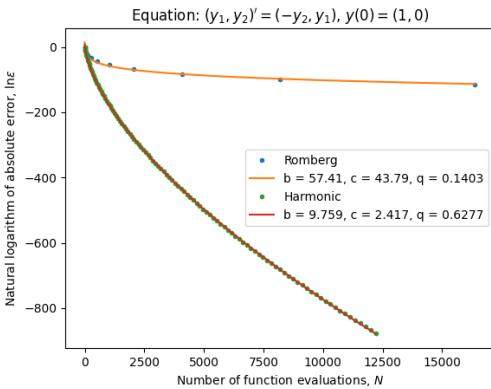
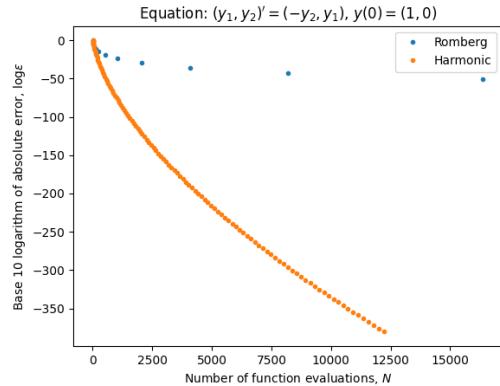
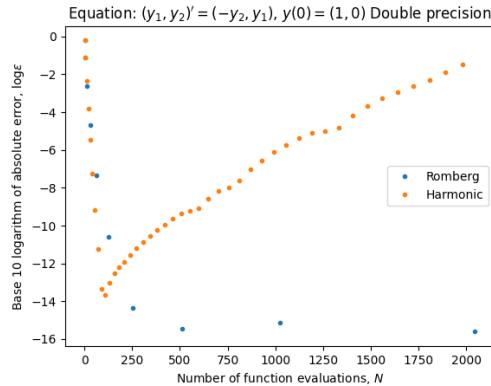
Now we will consider the following system of equations:

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

whose solution is

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

which is entire.



Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-lin evals-error	$1.215 \cdot 10^{60}$	6	61.05	0.1358	0.1308	0.03453
Harmonic	lin-lin evals-error	$9 \cdot 10^9$	6.841	2.636	0.007896	0.62	0.0002457
Romberg	lin-lin steps-error	1.083	0.000101	0.6717	$4.963 \cdot 10^{-7}$	2.007	$1.48 \cdot 10^{-8}$
Harmonic	lin-lin steps-error	$4.276 \cdot 10^8$	6.508	2.675	0.006923	1.237	0.0002153

The harmonic sequence works better than 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 suspiciously large.

4.2.7 Mathematical pendulum

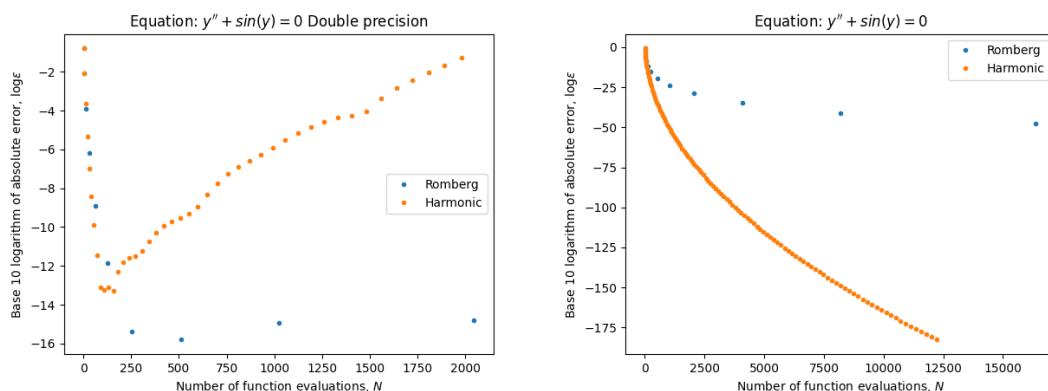
Now we will consider the mathematical pendulum equation:

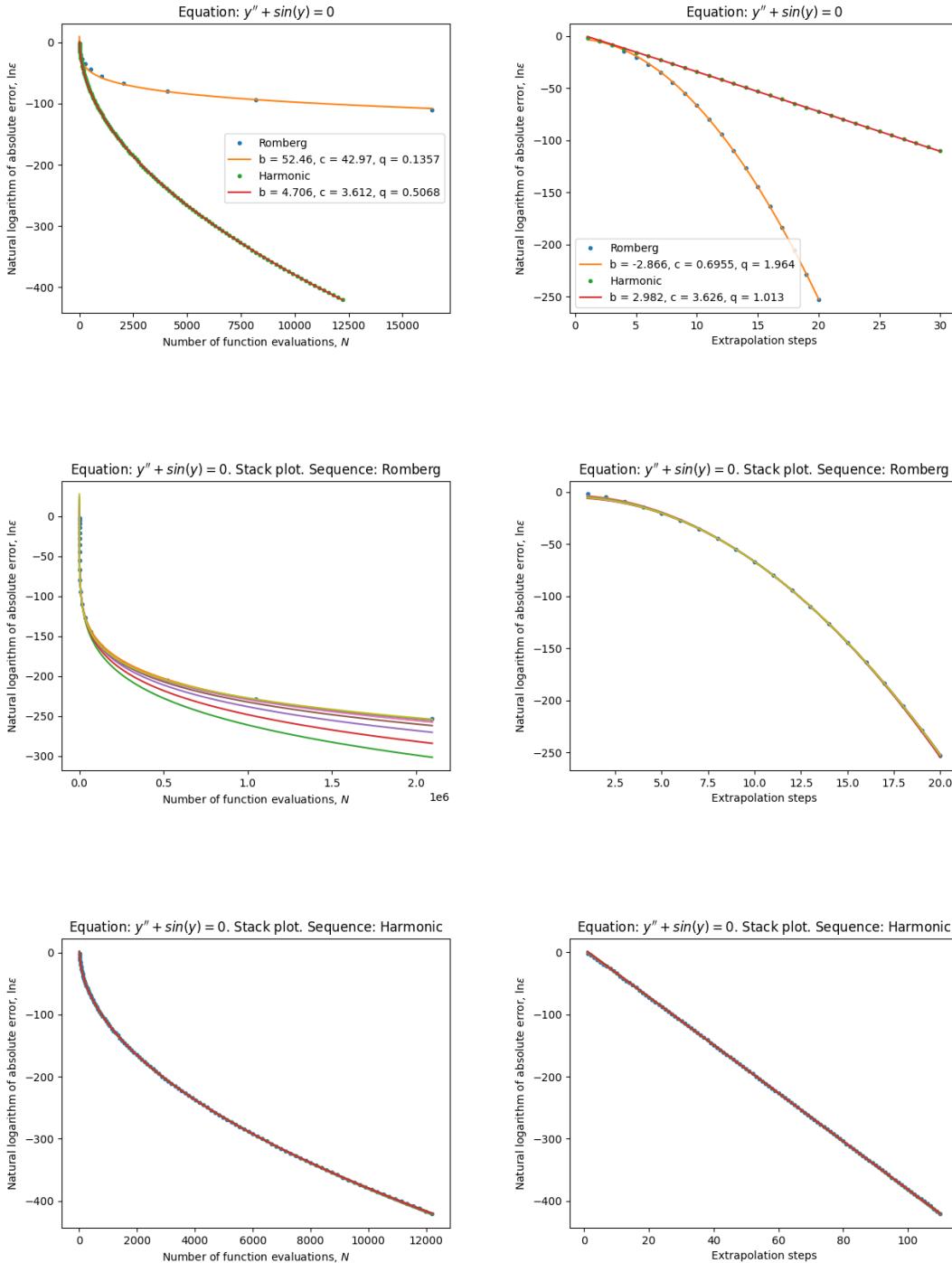
$$y''(t) + \sin y(t) = 0, \quad y(0) = 0, \quad y'(0) = 1, \quad t \in [0, 1]. \quad (4.8)$$

We use the following solution as a reference solution:

$$\begin{aligned} y(1) = & 0.8477986816771168446636783318809189285204869487805839233894518237264 \\ & 0630434688566338353630224868585976468140927461204156022529742699705231816982332 \\ & 8758573465006731337868899164920465794464433365734542344489127172608253083611368 \\ & 1006527088120002980265012291946040048160269697235789023333048631065165528080437 \\ & 9429035809638912058441220815487971711586081657146212645642189375079112255277643 \\ & 9852179622939101197041300976161428914224324373911458462193934898208401314255177 \\ & 19029786828908788598595270867026923369, \end{aligned}$$

$$\begin{aligned} y'(1) = & 0.5685689980951714899417351630224588908163828902980003832061653705295 \\ & 9766024841080148281818712844186765300851429899969930131242230797649624727126014 \\ & 4817345395779299526094029847932513513414050406462412114396118139738629227789263 \\ & 8171260577815198979573088520212213199643020386321347981028859823073409833268487 \\ & 4465028063198951013936721519329558610769737497582995347991161206624419371276898 \\ & 6489889976919046237233718506150977278781928633388966603053245457050361914127647 \\ & 64458195896384935977161329263091278345 \end{aligned}$$





Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$3.245 \cdot 10^{52}$	6	56.9	0.1423	0.1295	0.03859
Harmonic	lin-ln evals-error	471.5	0.2744	3.713	0.0002618	0.5042	$1.13 \cdot 10^{-5}$
Romberg	lin-ln steps-error	0.01384	0.3333	0.6305	0.002076	1.995	$6.207 \cdot 10^{-5}$
Harmonic	lin-ln steps-error	72.54	0.2499	3.718	0.0002297	1.008	$9.918 \cdot 10^{-6}$

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.

4.2.8 Federpendel

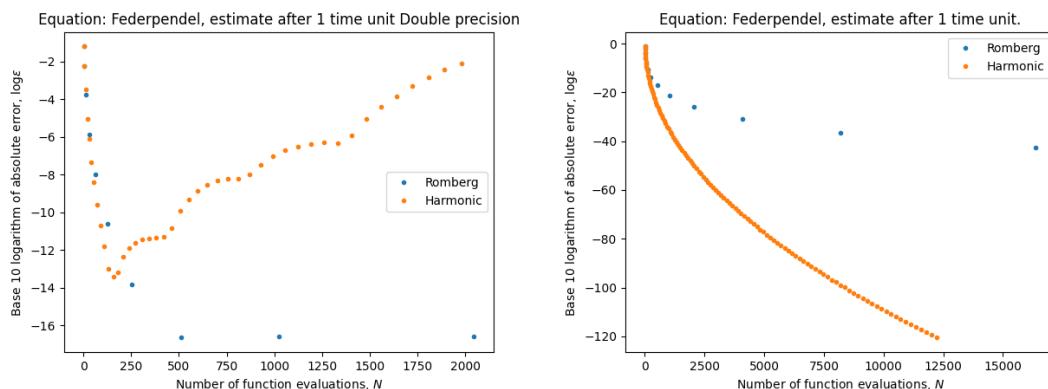
Now we will consider the equation of motion for das Federpendel or the spring pendulum:

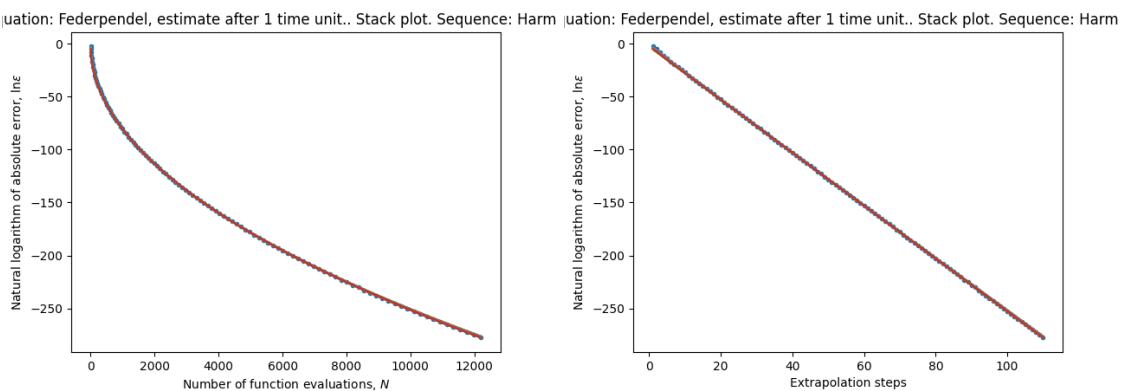
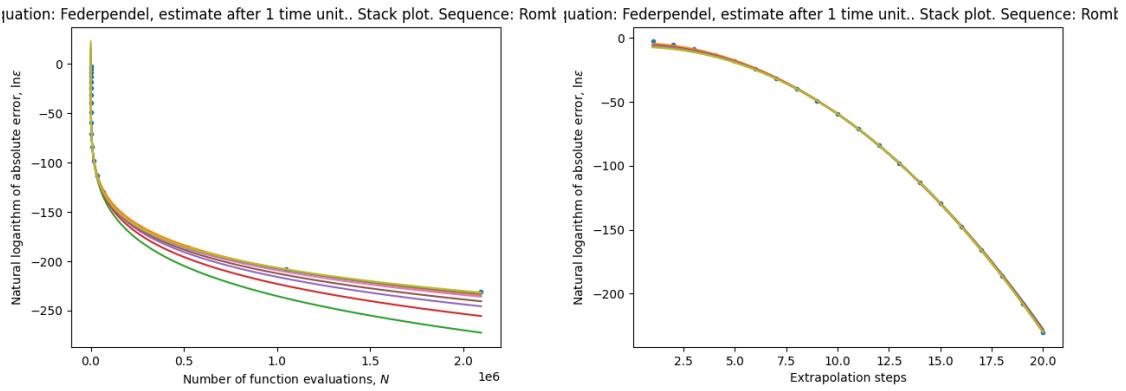
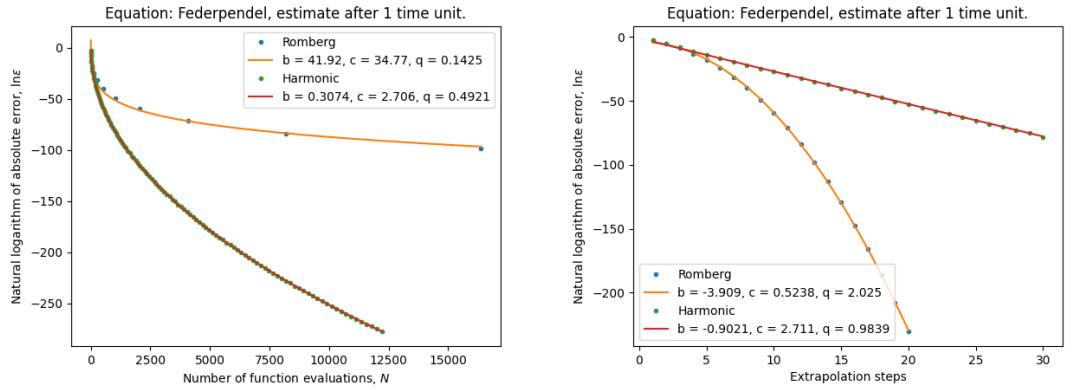
$$\mathbf{p}' = -(|\mathbf{q}| - 1) \frac{\mathbf{q}}{|\mathbf{q}|} - \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{q}' = \mathbf{p}$$

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

In the first case, we use the following solution as a reference solution:

$$\begin{aligned} p_1(2) &= 0.908485951398498847696036532914960625032705238066880976505590514636 \\ &7905962171740676512707256297490244309869210549143488281971351049854478027514056 \\ &6262920923919094308879461423242023249006528802304016469019659450867666853516641 \\ &8971605815998588638997253128128298126845123206573170821384952679781625192283326 \\ &461437859956514402961485773917489613281890858712434486197850316077660975033430 \\ &1861509139734505417881523161797903118442562808951135430345052299394995970839644 \\ &880196049228869133677552355615201639841, p_2(2) = -0.0400127440187803370433611931 \\ &1971319694982740419870372756532600713589357815311625888442455990290263338843920 \\ &0169186026061570394613581614141168758223380117460540462680460365968098566107856 \\ &7794822016124704981807041628519497684400629612447710926826683363619072300602971 \\ &3149111709247362257659423681770012878721823547658511754828372204936485071190516 \\ &7068986285810424331986673275583066137879769596277394094224029428587321605052542 \\ &16934224179224762870110338678881856667907336082354739548076217771507305280004', q_1(2) = \\ &-0.0578905193478659361183269820123810589022849245683766184320848367032663559388 \\ &5844470933229180067670016942393390771972203851305951263697059378236143483165809 \\ &7674960924145345347938203698716492781871838123431284351121896401807895747317005 \\ &7453144181599988319918727853073479669870455092154852833038253599472175252066107 \\ &8706856514085070876693593731236012696482527120526805925642310343906436262801205 \\ &7942225535815942692762334818955055874353118056680076137468438477813164951871956 \\ &636577739297079502516623843152, q_2(2) = -1.0336632796059989196849426774308533254 \\ &8477514735768044836892501581121860537920088974942190809167733252204685854070956 \\ &4361720865486118909151079825532293678649809860388715183392093022856689516474945 \\ &9579768574320635663289125546215560356989982492005036765245846059130661617492068 \\ &0036366502840459245685377224521969333538615600597727983589388152976741517156030 \\ &7613621573703750005039261045148332504554722529004095095646609612057847931708647 \\ &2139909045481432231784472145723227375922414201905077415299034086814. \end{aligned}$$

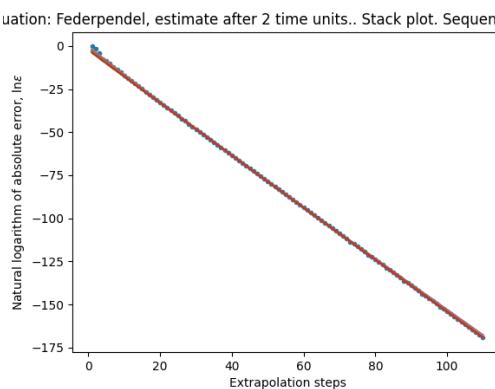
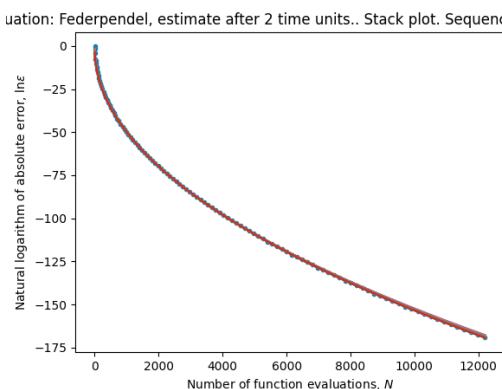
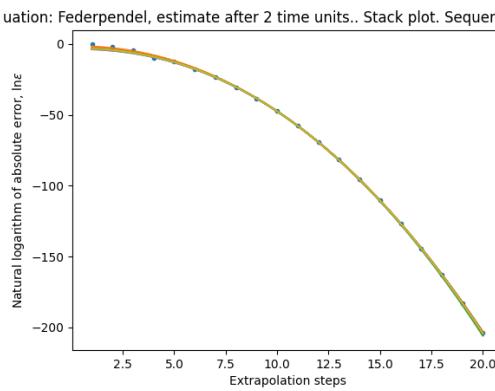
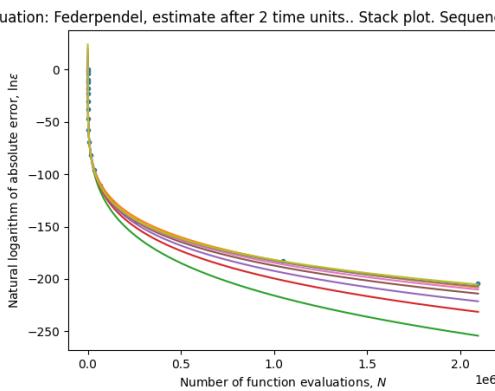
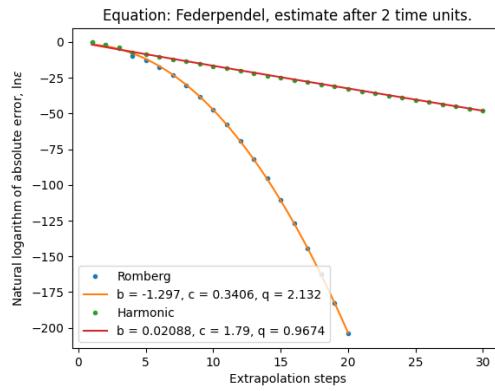
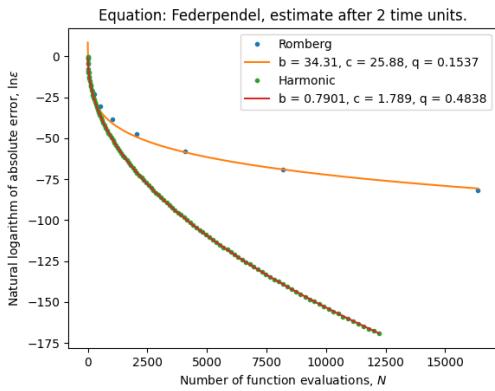
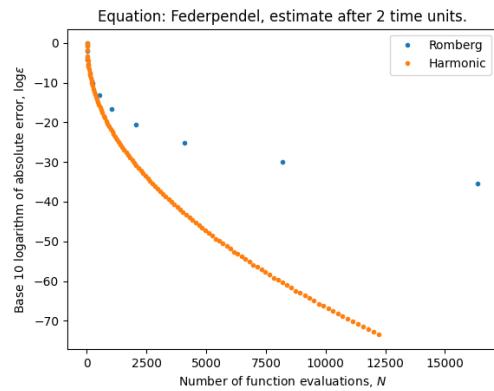
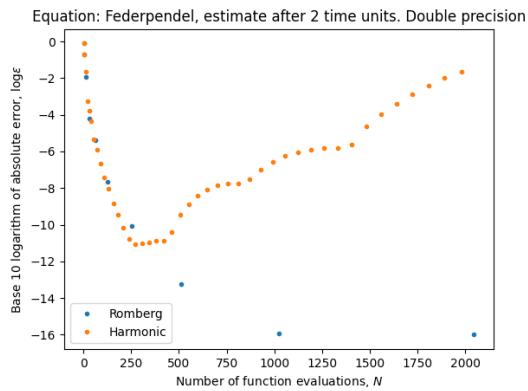




Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-ln evals-error	$8.241 \cdot 10^{40}$	6	43.86	0.1208	0.1376	0.02818
Harmonic	lin-ln evals-error	0.5082	0.2486	2.612	0.0003409	0.4955	$1.457 \cdot 10^{-5}$
Romberg	lin-ln steps-error	0.002791	0.3701	0.4615	0.003294	2.064	$8.941 \cdot 10^{-5}$
Harmonic	lin-ln steps-error	0.1465	0.2485	2.613	0.0003471	0.9909	$1.492 \cdot 10^{-5}$

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.



Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-lin evals-error	$2.335 \cdot 10^{34}$	6	32.11	0.1569	0.1519	0.03495
Harmonic	lin-lin evals-error	0.5245	0.3886	1.643	0.001271	0.492	$5.413 \cdot 10^{-5}$
Romberg	lin-lin steps-error	0.05507	0.06164	0.2956	0.001162	2.178	$3.178 \cdot 10^{-5}$
Harmonic	lin-lin steps-error	0.2422	0.3718	1.642	0.001252	0.984	$5.36 \cdot 10^{-5}$

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

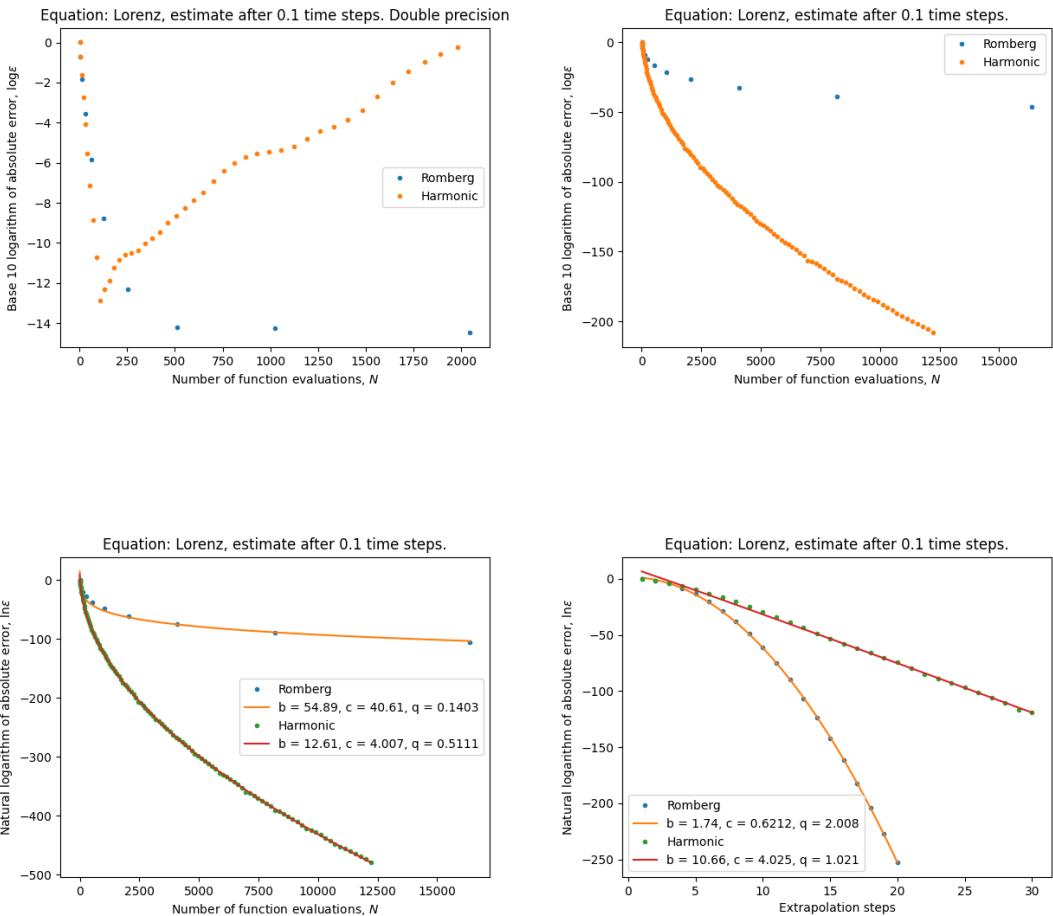
We seem to have very nice fit for exponential convergence in the number of steps for the Romberg sequence. We also have very nice fit for exponential convergence for the harmonic sequence.

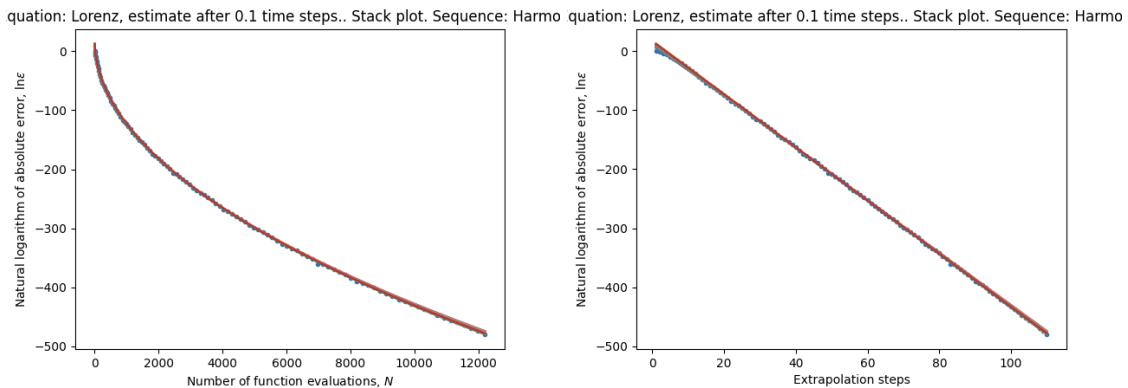
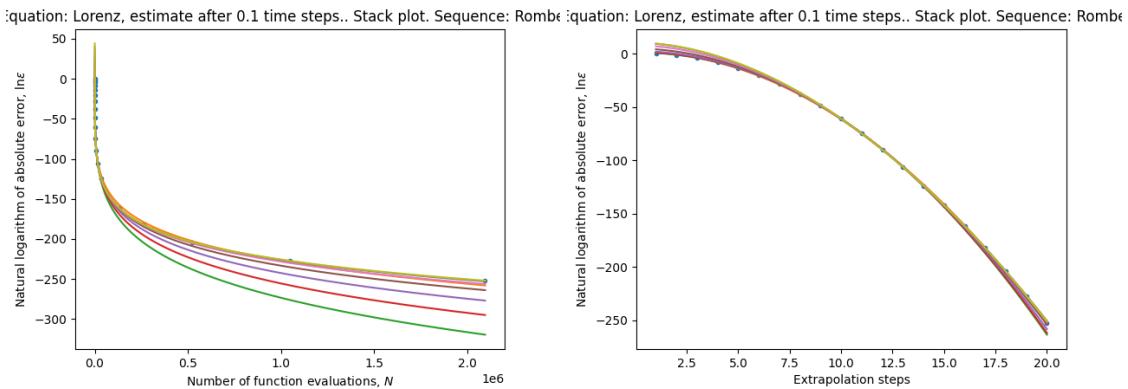
4.2.9 Lorenz equations

The Lorenz equations are the following system:

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

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

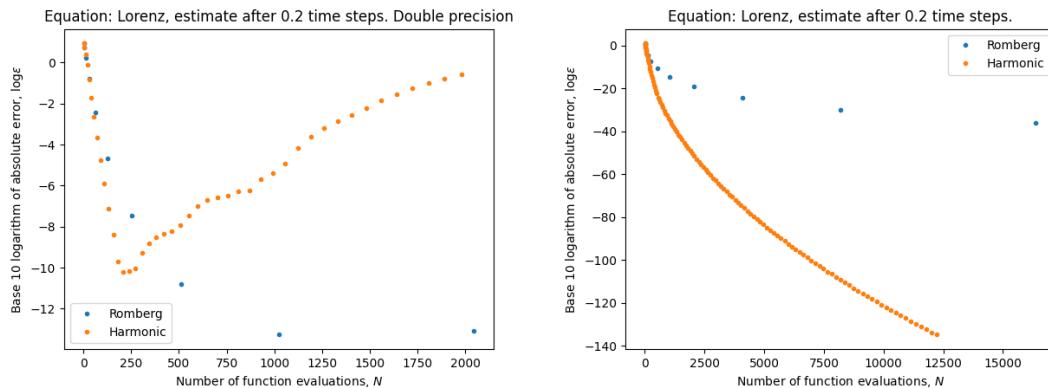


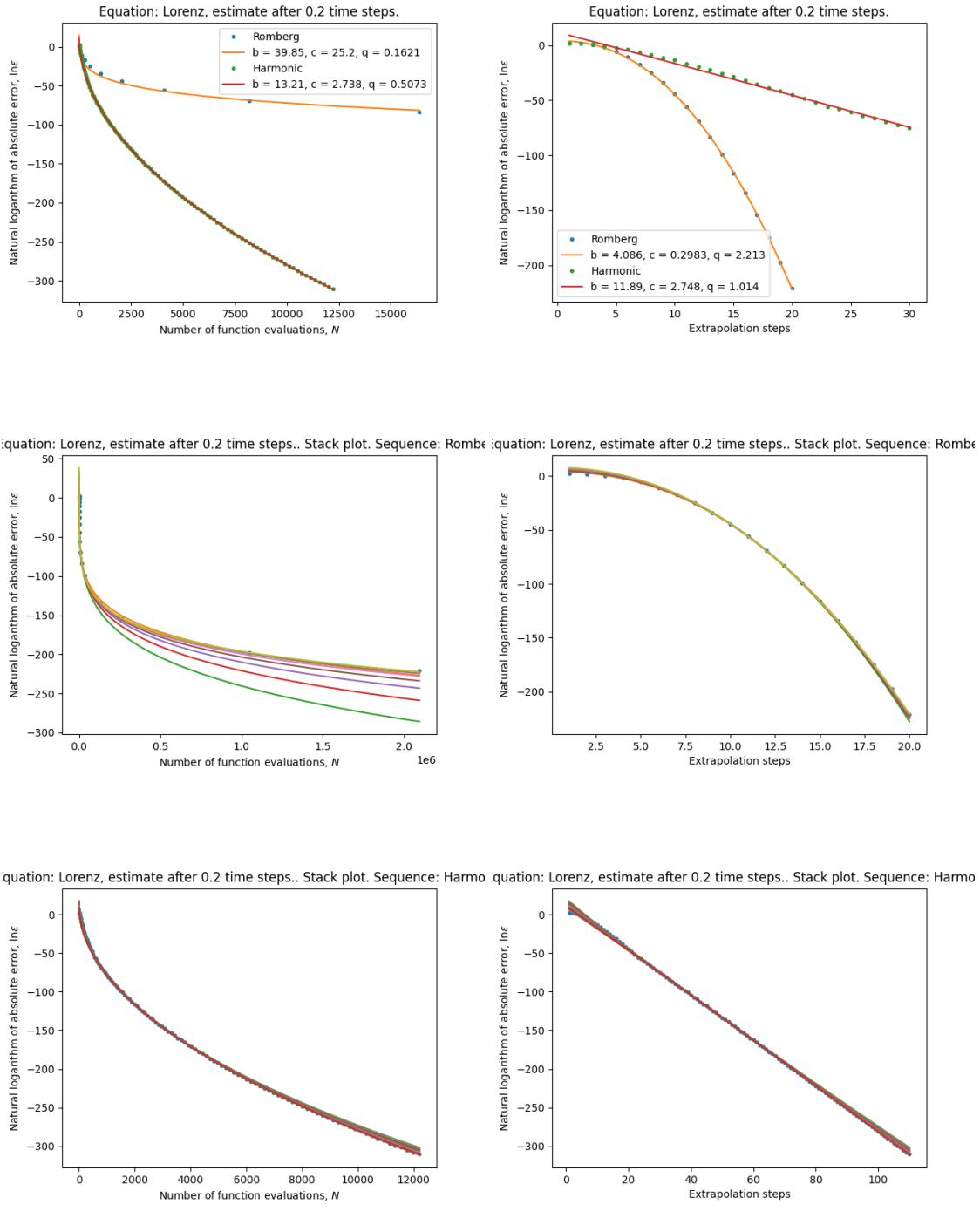


Sequence	Plot	$A\text{-mean}$	$A\text{-var}$	$c\text{-mean}$	$c\text{-var}$	$q\text{-mean}$	$q\text{-var}$
Romberg	lin-lin evals-error	$1.532 \cdot 10^{73}$	6	66.42	0.2606	0.1285	0.07315
Harmonic	lin-lin evals-error	$2.639 \cdot 10^7$	6.828	4.214	0.001814	0.5062	$7.152 \cdot 10^{-5}$
Romberg	lin-lin steps-error	8932	2.321	0.7248	0.06447	1.976	0.001984
Harmonic	lin-lin steps-error	$2.903 \cdot 10^6$	6.603	4.22	0.001799	1.012	$7.141 \cdot 10^{-5}$

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.





Sequence	Plot	A-mean	A-var	c-mean	c-var	q-mean	q-var
Romberg	lin-ln evals-error	$2.009 \cdot 10^{46}$	6	37.38	0.2052	0.1527	0.04644
Harmonic	lin-ln evals-error	$7.974 \cdot 10^8$	7.349	2.931	0.02946	0.5011	0.001063
Romberg	lin-ln steps-error	1040	1.066	0.337	0.01327	2.18	0.0003518
Harmonic	lin-ln steps-error	$1.538 \cdot 10^8$	7.211	2.931	0.02863	1.002	0.001041

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

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

IVP	Sequence	b	c	q
$y' = y, y(0) = 0$	Romberg	65.485	52.551	0.13291
$y' = y, y(0) = 0$	Harmonic	8.8124	3.0433	0.61436
$y' = y(1 - y)$	Romberg	53.149	47.621	0.13177
$y' = y(1 - y)$	Harmonic	1.7927	4.0933	0.51064
$y' = 1 + y^2, y(0) = 0$	Romberg	33.382	24.704	0.15812
$y' = 1 + y^2, y(0) = 0$	Harmonic	3.3496	1.8647	0.51993
$(y_1, y_2)' = (-y_2, y_1), y(0) = (1, 0)$	Romberg	57.413	43.792	0.1403
$(y_1, y_2)' = (-y_2, y_1), y(0) = (1, 0)$	Harmonic	9.7592	2.4171	0.62765
$y' = y^2, y(0) = 1/2$	Romberg	39.999	31.606	0.14796
$y' = y^2, y(0) = 1/2$	Harmonic	3.5994	2.5894	0.51472
$y' = y^2, y(0) = 1/(1 + 10^{-2})$	Romberg	11.039	2.3505	0.26394
$y' = y^2, y(0) = 1/(1 + 10^{-2})$	Harmonic	4.7796	0.032416	0.6592
$y' = y^2, y(0) = 1/(1 + 10^{-4})$	Romberg	9.6163	0.016825	0.50749
$y' = y^2, y(0) = 1/(1 + 10^{-4})$	Harmonic	9.2104	0.00016925	0.74975
$y' = -1/2y, y(0) = \sqrt{2}$	Romberg	42.458	37.281	0.13969
$y' = -1/2y, y(0) = \sqrt{2}$	Harmonic	-0.45512	3.1436	0.49744
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-2}}$	Romberg	6.2634	4.2006	0.23055
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-2}}$	Harmonic	-1.7425	0.33371	0.45544
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-4}}$	Romberg	-1.4645	0.28248	0.33417
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-4}}$	Harmonic	2.2343	3.3206	0.091009
$y'' + \sin(y) = 0$	Romberg	52.459	42.97	0.13575
$y'' + \sin(y) = 0$	Harmonic	4.7064	3.6116	0.50678
Federpendel, estimate after 1 time unit.	Romberg	41.924	34.768	0.14246
Federpendel, estimate after 1 time unit.	Harmonic	0.30737	2.706	0.49211
Federpendel, estimate after 2 time units.	Romberg	34.307	25.884	0.15365
Federpendel, estimate after 2 time units.	Harmonic	0.79006	1.789	0.48379
Lorenz, estimate after 0.1 time units.	Romberg	54.885	40.609	0.14032
Lorenz, estimate after 0.1 time units.	Harmonic	12.615	4.0068	0.5111
Lorenz, estimate after 0.2 time units.	Romberg	39.854	25.197	0.16206
Lorenz, estimate after 0.2 time units.	Harmonic	13.212	2.7376	0.50732

Table 4.1: Optimal parameters by test case

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

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

IVP	Sequence	b	c	q
$y' = y, y(0) = 0$	Romberg	-1.5037	0.87585	1.9378
$y' = y, y(0) = 0$	Harmonic	6.425	3.1061	1.225
$y' = y(1 - y)$	Romberg	-7.388	0.79828	1.9293
$y' = y(1 - y)$	Harmonic	-0.19572	4.1127	1.0204
$y' = 1 + y^2, y(0) = 0$	Romberg	-1.1782	0.30723	2.1752
$y' = 1 + y^2, y(0) = 0$	Harmonic	2.405	1.8763	1.0387
$(y_1, y_2)' = (-y_2, y_1), y(0) = (1, 0)$	Romberg	0.13463	0.67385	2.0061
$(y_1, y_2)' = (-y_2, y_1), y(0) = (1, 0)$	Harmonic	7.7307	2.4712	1.2513
$y' = y^2, y(0) = 1/2$	Romberg	-2.521	0.44573	2.0777
$y' = y^2, y(0) = 1/2$	Harmonic	2.3181	2.6033	1.0284
$y' = y^2, y(0) = 1/(1 + 10^{-2})$	Romberg	5.4845	0.0052111	3.2981
$y' = y^2, y(0) = 1/(1 + 10^{-2})$	Harmonic	4.7474	0.033263	1.3137
$y' = y^2, y(0) = 1/(1 + 10^{-4})$	Romberg	9.3422	$1.3598e - 07$	6.3724
$y' = y^2, y(0) = 1/(1 + 10^{-4})$	Harmonic	9.2101	0.00017545	1.4929
$y' = -1/2y, y(0) = \sqrt{2}$	Romberg	-6.2149	0.57862	1.9997
$y' = -1/2y, y(0) = \sqrt{2}$	Harmonic	-1.8945	3.1518	0.99437
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-2}}$	Romberg	-2.0186	0.016933	2.9296
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-2}}$	Harmonic	-1.8689	0.33241	0.91148
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-4}}$	Romberg	-2.613	0.00012201	4.1931
$y' = -1/2y, y(0) = \sqrt{1 + 10^{-4}}$	Harmonic	1.3726	2.6336	0.20682
$y'' + \sin(y) = 0$	Romberg	-2.8658	0.69546	1.9636
$y'' + \sin(y) = 0$	Harmonic	2.9822	3.6265	1.0128
Federpendel, estimate after 1 time unit.	Romberg	-3.9094	0.52384	2.0251
Federpendel, estimate after 1 time unit.	Harmonic	-0.90208	2.7108	0.98385
Federpendel, estimate after 2 time units.	Romberg	-1.2973	0.34056	2.132
Federpendel, estimate after 2 time units.	Harmonic	0.020877	1.79	0.96743
Lorenz, estimate after 0.1 time units.	Romberg	1.7404	0.62119	2.0083
Lorenz, estimate after 0.1 time units.	Harmonic	10.659	4.0254	1.0213
Lorenz, estimate after 0.2 time units.	Romberg	4.0863	0.2983	2.2131
Lorenz, estimate after 0.2 time units.	Harmonic	11.891	2.7477	1.014

Table 4.2: Optimal parameters by test case

Bibliography

- [1] Perter Deuflhard and Andreas Hohmann. *Numerical Analysis in Modern Scientific Computing*, vol. 43 of Texts in Applied Mathematics, Springer, New York, 2003.
- [2] Konrad Knopp. *Theorie und Anwendung der unendlichen Reihen.*, Springer Verlag, Berlin, Heidelberg, New York, (5. Auflage) 1964.