

A Comparative Investigation of Analytical and Numerical Methods for Solving Ordinary Differential Equations with Applications to Mathematical Models in Science and Engineering

G.Bharathi

Department of Maths, Siddhartha Institute of technology & Sciences, narapally, 500088

bharathidhanvin@gmail.com

Abstract

Ordinary differential equations (ODEs) underlie most quantitative models in science and engineering, yet the choice between solving them analytically and solving them numerically is seldom examined on common ground. This paper compares the two approaches on five models: Newton's law of cooling, logistic population growth, a series RLC circuit, a stiff linear test equation and the nonlinear simple pendulum. Each model is solved in closed form by a classical technique (integrating factor, separation of variables, characteristic equation, elliptic integral). The closed form then serves as the exact reference for the explicit Euler, Heun, classical fourth-order Runge–Kutta (RK4) and fourth-order Adams–Bashforth–Moulton (ABM4) methods, the implicit backward Euler and trapezoidal methods, and five adaptive solvers from the SciPy library. Every numerical result reported was computed for this study. At the finest step sizes the observed orders of convergence were 1.00–1.12 for Euler, 1.99–2.01 for Heun, 3.98–4.03 for RK4 and 4.01–4.24 for ABM4, in agreement with theory. To reach a scaled maximum error of 10^{-6} , RK4 needed 88 to 1,188 function evaluations, Heun's method 14 to 22 times as many, and Euler's method an estimated 4,500 to 12,000 times as many. On the stiff problem, explicit Euler and RK4 diverged at $h = 0.01$, while backward Euler gave an error of 2.8×10^{-5} at $h = 0.1$; an adaptive BDF solver needed 264 evaluations against 2,270 for an explicit adaptive solver. For the pendulum, the linearised analytical period was in error by 0.19% at 10° , 6.8% at 60° and 15.3% at 90° , whereas RK4 reproduced the exact elliptic-integral period to within 1.5×10^{-10} s. The study concludes that the two approaches are complementary: analytical solutions supply insight and a means of verification, and numerical methods supply generality.

Keywords: ordinary differential equations; analytical solution; Runge–Kutta method; Adams–Bashforth–Moulton method; stiffness; order of convergence; mathematical modelling

1. Introduction

Ordinary differential equations (ODEs) are the standard language for systems whose rate of change depends on their present state. The cooling of a body, the growth of a population, the charge on a capacitor, the swing of a pendulum and the progress of a chemical reaction are all written as initial value problems of the form

$$\frac{dy}{dt} = f(t, y), \quad y(t_0) = y_0. \quad (1)$$

When f is continuous and Lipschitz continuous in y , the Picard–Lindelöf theorem guarantees a unique local solution (Boyce & DiPrima, 2012). The theorem says that a solution exists. It does not say how to find it.

Two routes are available. Analytical methods produce a closed-form expression for $y(t)$. They apply to restricted classes of equations: separable, first-order linear, exact, and linear equations with constant coefficients (Kreyszig, 2011). Their value lies in the formula itself, which shows how the solution depends on every parameter. A time constant, a carrying capacity or a damping ratio can be read directly from it.

Numerical methods replace the continuous problem with a recurrence on a grid of time points. They apply to almost any f , including nonlinear and coupled systems for which no elementary closed form exists. Their price is an approximation error that depends on the step size, the order of the method and its stability properties (Butcher, 2016; Hairer et al., 1993). A poorly chosen method can waste computation by orders of magnitude or produce a solution that grows without bound when the true solution decays.

The two routes are usually taught separately, and a student or practising engineer is often left to choose between them by habit. This paper places them side by side on the same models. Five models from physics, biology and electrical engineering are solved analytically, and the closed-form solutions then serve as exact references for six numerical schemes and five library solvers. The comparison is quantitative: observed order of convergence, number of function evaluations needed to reach a fixed accuracy, behaviour on a stiff problem, and the error committed when an analytical solution is obtained only by linearising the model.

The paper is organised as follows. Section 2 reviews the literature. Section 3 states the objectives. Section 4 sets out the analytical and numerical methods. Section 5 presents the five test models and their exact solutions. Section 6 describes the computational procedure. Section 7 reports and discusses the results, and Section 8 concludes.

2. Review of Literature

Analytical theory and models. The classical solution techniques for first-order and linear higher-order equations are set out in the standard texts of Boyce and DiPrima (2012) and Kreyszig (2011). The models used here have long histories. The logistic equation was proposed by Verhulst (1838) as a correction to unbounded exponential growth and remains the starting point of population dynamics (Murray, 2002). The pendulum is the standard example of a nonlinear oscillator whose small-angle approximation is linear (Strogatz, 2015), and its exact period in terms of the complete elliptic integral is derived in Beléndez et al. (2007).

One-step methods. Euler's polygon method is the oldest numerical scheme for initial value problems. Runge (1895) and Kutta (1901) raised its order by evaluating f at intermediate points within a step, which led to the classical fourth-order formula still in general use. Butcher (2016) gives the modern algebraic theory of order conditions, and Hairer et al. (1993) give the reference treatment of non-stiff problems, including error estimation and step-size control.

Multistep methods. Adams methods reuse derivative values from earlier steps instead of computing new ones inside the step. The explicit Adams–Bashforth formula is normally paired with the implicit Adams–Moulton formula as a predictor–corrector, giving fourth-

order accuracy for two function evaluations per step (Lambert, 1991; Gear, 1971). Atkinson et al. (2009) and Burden and Faires (2011) present both families at textbook level.

Stiffness and stability. Curtiss and Hirschfelder (1952) identified a class of equations, arising in chemical kinetics, for which explicit methods need impractically small steps, and recommended backward differentiation. Dahlquist (1963) introduced A-stability and proved that no explicit linear multistep method is A-stable and that an A-stable linear multistep method cannot exceed order two. Prothero and Robinson (1974) proposed a model problem that separates the smooth solution from a fast transient, a variant of which is used in this paper. Hairer and Wanner (1996) is the standard reference, and Söderlind et al. (2015) survey sixty years of attempts to define stiffness precisely.

Adaptive solvers and software. Dormand and Prince (1980) constructed the embedded Runge–Kutta pair of orders 5 and 4 that underlies most general-purpose solvers. Shampine and Reichelt (1997) described the MATLAB ODE suite. Petzold (1983) and Hindmarsh (1983) developed automatic switching between Adams and backward differentiation formulas in the LSODA code. These algorithms are available in open-source form through SciPy (Virtanen et al., 2020). For long-time integration of conservative systems, Hairer et al. (2006) show that preservation of invariants such as energy matters as much as local accuracy.

Recent comparative studies. Short comparative papers continue to appear. Denis (2020) reviews analytical and numerical techniques and compares the Euler and fourth-order Runge–Kutta methods on the single test equation $y' = -6y$. Ali et al. (2024) compare Runge–Kutta and Adams–Bashforth–Moulton methods on two first-order equations, taking MATLAB's ode45 output as the benchmark. Salleh et al. (2024) compare the Euler, modified Euler and Runge–Kutta methods on equations of first, second and third order and report the smallest errors for the modified Euler method, a ranking that differs from the one the orders of the methods predict.

Research gap. Studies of this kind typically use one or two scalar equations and report accuracy at a fixed step size. Three questions are left open by such a design: how much work each method needs to reach the same accuracy, what happens when the problem is stiff, and how large the error is when the analytical solution itself rests on an approximation of the model. The present study addresses these three questions with exact solutions as the reference throughout.

3. Objectives of the Study

1. To derive closed-form solutions of five representative models from science and engineering using classical analytical techniques.
2. To implement the Euler, Heun, RK4 and ABM4 methods and verify their orders of convergence against the exact solutions.
3. To compare the computational work each method needs to reach the same prescribed accuracy.

4. To examine the behaviour of explicit and implicit methods on a stiff problem and relate it to linear stability theory.
5. To quantify the error of an analytical solution obtained by linearising a nonlinear model, using the simple pendulum.
6. To draw practical guidance on the choice of method from the results.

4. Methods

4.1 Analytical methods

Four classical techniques are used. Each applies to a specific form of equation, and Table 1 lists the form and the model it is applied to in Section 5.

Table 1. Analytical techniques used in the study

Technique	Form of equation	Applied to
Integrating factor	$y' + p(t)y = q(t)$	Cooling (P1), stiff problem (P4)
Separation of variables with partial fractions	$y' = g(y)h(t)$	Logistic growth (P2)
Characteristic equation with a particular solution	$ay'' + by' + cy = g(t)$	RLC circuit (P3), linearised pendulum (P5)
Energy integral and reduction to quadrature	$y'' = F(y)$	Nonlinear pendulum (P5)

For the first-order linear equation the integrating factor gives the general solution

$$y(t) = e^{-\int p dt} \left[\int q(t) e^{\int p dt} dt + C \right]. \quad (2)$$

For the second-order linear equation the roots of $ar^2 + br + c = 0$ determine the homogeneous solution. Complex roots $\alpha \pm i\beta$ give a damped oscillation $e^{\alpha t}(A\cos\beta t + B\sin\beta t)$, to which a particular solution is added.

4.2 Numerical methods

All methods advance the solution on a uniform grid $t_n = t_0 + nh$, where y_n approximates $y(t_n)$ and $f_n = f(t_n, y_n)$.

Explicit Euler method (order 1):

$$y_{n+1} = y_n + h f(t_n, y_n) \quad (3)$$

Heun's method, a two-stage Runge–Kutta method (order 2):

$$k_1 = f(t_n, y_n), \quad k_2 = f(t_n + h, y_n + hk_1), \quad y_{n+1} = y_n + \frac{h}{2} (k_1 + k_2) \quad (4)$$

Classical Runge–Kutta method, RK4 (order 4):

$$\begin{aligned} k_1 &= f(t_n, y_n), & k_2 &= f(t_n + h/2, y_n + h/2 k_1), \\ k_3 &= f(t_n + h/2, y_n + h/2 k_2), & k_4 &= f(t_n + h, y_n + h k_3), \\ y_{n+1} &= y_n + h/6 (k_1 + 2k_2 + 2k_3 + k_4) \end{aligned} \quad (5)$$

Adams–Bashforth–Moulton predictor–corrector, ABM4 (order 4). The predictor is the four-step Adams–Bashforth formula and the corrector is the three-step Adams–Moulton formula, applied once (PECE mode). The first three steps are taken with RK4.

$$\begin{aligned}\tilde{y}_{n+1} &= y_n + h/24 (55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}), \\ y_{n+1} &= y_n + h/24 (9f(t_{n+1}, \tilde{y}_{n+1}) + 19f_n - 5f_{n-1} + f_{n-2})\end{aligned}\quad (6)$$

Backward Euler method (implicit, order 1):

$$y_{n+1} = y_n + h f(t_{n+1}, y_{n+1}) \quad (7)$$

Trapezoidal method (implicit, order 2):

$$y_{n+1} = y_n + \frac{h}{2} [f(t_n, y_n) + f(t_{n+1}, y_{n+1})] \quad (8)$$

Both implicit methods require an equation to be solved for y_{n+1} at every step. For the linear stiff problem of Section 5 this equation is solved exactly.

4.3 Order of accuracy

A method has order p if its global error satisfies $E(h) \approx Ch^p$ as $h \rightarrow 0$, where C depends on the problem and the method but not on h . Halving the step size then divides the error by 2^p . The observed order is computed from errors at two successive step sizes:

$$p_{\text{obs}} = \log_2 \frac{E(h)}{E(h/2)} \quad (9)$$

4.4 Linear stability

Applied to the test equation $y' = \lambda y$, where λ has negative real part, a one-step method gives $y_{n+1} = R(z) y_n$ with $z = h\lambda$. The numerical solution decays, as the exact solution does, only if $|R(z)| < 1$. For explicit Euler $R(z) = 1 + z$, so a real negative λ requires $h < 2/|\lambda|$. For backward Euler $R(z) = 1/(1 - z)$, which is smaller than 1 in magnitude for every z in the left half-plane: the method is A-stable (Dahlquist, 1963). The trapezoidal method, with $R(z) = (1 + z/2)/(1 - z/2)$, is also A-stable, but $|R(z)| \rightarrow 1$ as $z \rightarrow -\infty$, so very fast components are damped slowly. Backward Euler has $R(z) \rightarrow 0$ in the same limit, a property known as L-stability (Hairer & Wanner, 1996).

Table 2 summarises the methods. The real stability limits of the explicit methods were computed for this study from the amplification factor, or for ABM4 from the spectral radius of the companion matrix of the PECE recurrence.

Table 2. Properties of the numerical methods

Method	Type	Order	Evaluations of f per step	Largest stable $-h\lambda$ (real $\lambda < 0$)
Euler	Explicit, one-step	1	1	2.000
Heun	Explicit, one-step	2	2	2.000
RK4	Explicit, one-step	4	4	2.785
ABM4 (PECE)	Explicit, four-step	4	2	1.285
Backward Euler	Implicit, one-step	1	One implicit solve	Unbounded (A- and L-stable)
Trapezoidal	Implicit, one-step	2	One implicit solve	Unbounded (A-stable)

5. Test Models and Their Analytical Solutions

Five models were chosen so that each has an exact solution and each exposes a different property of the numerical methods: P1 to P3 are smooth and non-stiff, P4 is stiff, and P5 is nonlinear with a conserved energy.

5.1 P1: Newton's law of cooling

A body at temperature T loses heat to surroundings at temperature T_a at a rate proportional to the temperature difference:

$$\frac{dT}{dt} = -k(T - T_a), \quad T(0) = T_0. \quad (10)$$

The equation is first-order linear. Multiplying by the integrating factor e^{kt} and integrating gives

$$T(t) = T_a + (T_0 - T_a)e^{-kt}. \quad (11)$$

The values used are $k = 0.1 \text{ min}^{-1}$, $T_a = 25^\circ\text{C}$ and $T_0 = 90^\circ\text{C}$ on $0 \leq t \leq 30 \text{ min}$. The formula shows at once that the temperature excess falls by a factor e in every interval of $1/k = 10 \text{ min}$, whatever the starting temperature.

5.2 P2: Logistic population growth

A population P with intrinsic growth rate r and carrying capacity K satisfies

$$\frac{dP}{dt} = rP \left(1 - \frac{P}{K}\right), \quad P(0) = P_0. \quad (12)$$

The equation is nonlinear but separable. Separating the variables and resolving $1/[P(1 - P/K)]$ into partial fractions gives

$$P(t) = \frac{K}{1 + \frac{K - P_0}{P_0} e^{-rt}}. \quad (13)$$

The values used are $r = 0.5$ per unit time, $K = 1000$ and $P_0 = 10$ on $0 \leq t \leq 20$. Growth is fastest when $P = K/2$, which the formula places at $t = \ln[(K - P_0)/P_0]/r \approx 9.19$.

5.3 P3: Series RLC circuit

The charge q on the capacitor of a series circuit driven by a constant e.m.f. E_0 satisfies

$$L \frac{d^2 q}{dt^2} + R \frac{dq}{dt} + \frac{q}{C} = E_0, \quad q(0) = 0, \quad q'(0) = 0. \quad (14)$$

With $L = 1$ H, $R = 2$ Ω , $C = 0.1$ F and $E_0 = 10$ V, the characteristic equation $r^2 + 2r + 10 = 0$ has roots $-1 \pm 3i$, so the circuit is under-damped. Adding the particular solution $q = CE_0 = 1$ and applying the initial conditions gives

$$q(t) = 1 - e^{-t}(\cos 3t + 1/3 \sin 3t). \quad (15)$$

The interval is $0 \leq t \leq 10$ s. The charge first peaks at $t = \pi/3$ s, where it overshoots its final value by $e^{-\pi/3} \approx 35.1\%$. For numerical solution the equation is written as the first-order system $y_1' = y_2$, $y_2' = (E_0 - Ry_2 - y_1/C)/L$.

5.4 P4: A stiff linear problem

The fourth problem is of the type introduced by Prothero and Robinson (1974):

$$\frac{dy}{dt} = \lambda (y - \cos t) - \sin t, \quad y(0) = 0, \quad \lambda = -1000. \quad (16)$$

The integrating factor method gives

$$y(t) = \cos t - e^{\lambda t}. \quad (17)$$

The interval is $0 \leq t \leq 1$. The solution has two time scales. The transient e^{-1000t} falls below 10^{-4} by $t = 0.01$, after which $y(t)$ is indistinguishable from the slowly varying $\cos t$. Accuracy alone would therefore permit large steps over almost the whole interval.

5.5 P5: The simple pendulum

A pendulum of length L released from rest at angle θ_0 satisfies

$$\frac{d^2 \theta}{dt^2} + \frac{g}{L} \sin \theta = 0, \quad \theta(0) = \theta_0, \quad \theta'(0) = 0. \quad (18)$$

No solution in elementary functions exists. The usual analytical treatment replaces $\sin \theta$ by θ , which gives $\theta(t) = \theta_0 \cos(\omega_0 t)$ with $\omega_0 = \sqrt{g/L}$ and a period that does not depend on amplitude:

$$T_0 = 2\pi \sqrt{\frac{L}{g}}. \quad (19)$$

The nonlinear equation can still be integrated once, because the energy $1/2 \theta'^2 - \omega_0^2 \cos \theta$ is constant. Reduction to quadrature gives the exact period in terms of the complete elliptic integral of the first kind (Beléndez et al., 2007):

$$T = 4 \sqrt{\frac{L}{g}} K(k), \quad k = \sin \frac{\theta_0}{2}, \quad K(k) = \int_0^{\pi/2} \frac{d\varphi}{\sqrt{1 - k^2 \sin^2 \varphi}}. \quad (20)$$

The values used are $g = 9.81$ m s⁻² and $L = 1$ m, for which $T_0 = 2.006067$ s. This model tests the analytical side of the comparison: the closed-form answer most often quoted is exact only in the limit $\theta_0 \rightarrow 0$.

6. Computational Methodology

Implementation. All computations were carried out in double-precision arithmetic in Python 3.13 with NumPy 2.5.3 and SciPy 1.18.1 (Harris et al., 2020; Virtanen et al., 2020). Figures were produced with Matplotlib (Hunter, 2007). The six fixed-step methods were coded directly from the formulas of Section 4.2. The adaptive solvers were called through SciPy's `solve_ivp` interface: RK45 (the Dormand–Prince 5(4) pair), DOP853 (an explicit Runge–Kutta method of order 8), Radau (an implicit Runge–Kutta method of order 5), BDF (variable-order backward differentiation) and LSODA (Adams and BDF formulas with automatic switching).

Accuracy and order (P1–P3). Each problem was solved with the four explicit fixed-step methods at $h = 0.5, 0.25, 0.125, 0.0625$ and 0.03125 . The error measure is the maximum absolute error over all grid points, $E(h) = \max_n |y_n - y(t_n)|$, with the charge q taken as the reported component for P3. The observed order was computed from each successive pair of step sizes.

Work at equal accuracy (P1–P3). Because the three solutions differ in magnitude, the error was scaled by the largest value of the exact solution on the interval (90 for P1, 995.5 for P2 and 1.351 for P3):

$$E_s = \frac{\max_n |y_n - y(t_n)|}{\max_t |y(t)|} \quad (21)$$

For each method, the smallest number of steps giving $E_s \leq 10^{-6}$ was found by doubling followed by bisection. Cost is reported as the number of evaluations of f , which does not depend on the computer or the programming language, and as wall-clock time (the best of seven runs). Euler's method would need between 4×10^5 and 1.4×10^7 steps, so its requirement was estimated from the error constant C in $E_s \approx Ch$, measured on grids of 2^{16} and 2^{17} steps, on which the observed order was 1.000. The adaptive solvers were run with relative tolerance 10^{-8} and absolute tolerance 10^{-10} .

Stiff problem (P4). The four methods Euler, RK4, backward Euler and trapezoidal were run at $h = 0.1, 0.01, 0.0025, 0.002, 0.001$ and 0.0001 , and the absolute error at $t = 1$ was recorded. The five adaptive solvers were run with relative tolerance 10^{-6} and absolute tolerance 10^{-9} .

Pendulum (P5). For release angles from 5° to 150° the nonlinear equation was integrated by RK4 with $h = 0.001$ s. The half-period was taken as the first time at which the angular velocity returns to zero, located by linear interpolation between grid points. The result was compared with the elliptic-integral period, evaluated with `scipy.special.ellipk`, and with the linearised period T_0 . Conservation of energy was tested by integrating for 100 s at $\theta_0 = 60^\circ$ and $h = 0.01$ s and recording the largest relative deviation of the energy from its initial value.

7. Results and Discussion

7.1 Accuracy and observed order of convergence

Figure 1 shows the exact solutions of P1 to P3 together with the Euler and RK4 solutions at deliberately coarse step sizes. The RK4 points lie on the exact curves to plotting accuracy. Euler's method runs below the cooling curve, delays the logistic transition and, on the RLC circuit, produces a first peak of about 1.87 C where the exact value is 1.351 C. With $h = 1/6$ s its amplification factor per step is 0.972 against the exact decay factor $e^{-h} = 0.846$, so the computed oscillation is far too lightly damped.

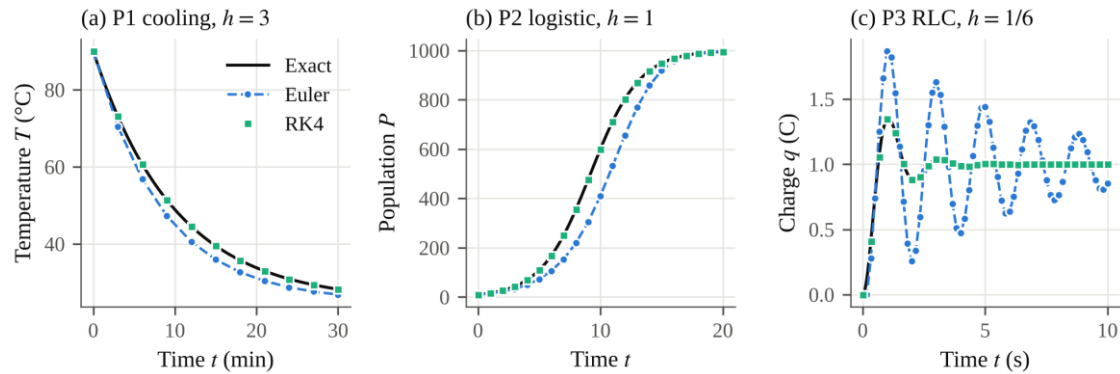


Figure 1. Exact solutions (solid line) with Euler (circles) and RK4 (squares) approximations at coarse step sizes: (a) Newton's law of cooling, (b) logistic growth, (c) series RLC circuit.

All four explicit methods converged at their theoretical rates on the three non-stiff problems. Table 3 gives the maximum absolute error at each step size and the observed order from the two finest grids.

Table 3. Maximum absolute error $E(h)$ and observed order of convergence for P1–P3

Problem	Method	$h = 0.5$	$h = 0.25$	$h = 0.125$	$h = 0.0625$	$h = 0.03125$	Order
P1 Cooling	Euler	6.11×10^{-1}	3.02×10^{-1}	1.50×10^{-1}	7.49×10^{-2}	3.74×10^{-2}	1.00
P1 Cooling	Heun	1.03×10^{-2}	2.54×10^{-3}	6.29×10^{-4}	1.56×10^{-4}	3.90×10^{-5}	2.00
P1 Cooling	RK4	1.30×10^{-6}	7.95×10^{-8}	4.92×10^{-9}	3.06×10^{-10}	1.90×10^{-11}	4.01
P1 Cooling	ABM4	4.36×10^{-6}	2.60×10^{-7}	1.58×10^{-8}	9.76×10^{-10}	6.06×10^{-11}	4.01
P2 Logistic	Euler	9.80×10^1	4.98×10^1	2.51×10^1	1.26×10^1	6.30	1.00
P2 Logistic	Heun	7.01	1.89	4.92×10^{-1}	1.25×10^{-1}	3.17×10^{-2}	1.99
P2 Logistic	RK4	1.75×10^{-2}	1.19×10^{-3}	7.80×10^{-5}	4.98×10^{-6}	3.15×10^{-7}	3.98
P2 Logistic	ABM4	1.49×10^{-2}	9.50×10^{-4}	5.85×10^{-5}	3.58×10^{-6}	2.20×10^{-7}	4.02
P3 RLC	Euler	8.95×10^3	1.05×10^1	3.58×10^{-1}	1.43×10^{-1}	6.57×10^{-2}	1.12
P3 RLC	Heun	8.85×10^{-1}	1.26×10^{-1}	3.14×10^{-2}	7.80×10^{-3}	1.93×10^{-3}	2.01
P3 RLC	RK4	8.62×10^{-2}	4.56×10^{-3}	2.69×10^{-4}	1.62×10^{-5}	9.89×10^{-7}	4.03
P3 RLC	ABM4	8.26×10^4	3.17×10^{-2}	1.41×10^{-3}	6.48×10^{-5}	3.43×10^{-6}	4.24

Units: °C for P1, individuals for P2, coulombs for P3. The order is $\log_2[E(0.0625)/E(0.03125)]$.

Three observations follow from the table.

First, order dominates accuracy. On P1 at $h = 0.03125$ the RK4 error is 1.90×10^{-11} °C against 3.74×10^{-2} °C for Euler, a ratio of about 2×10^9 for four times the work per step. Halving h gains one binary digit with Euler and four with RK4.

Second, methods of equal order are not equally accurate. RK4 was 3.2 to 3.4 times more accurate than ABM4 on P1 and 3.5 to 7 times more accurate on P3 at every stable step size, while ABM4 was 1.2 to 1.4 times more accurate on P2. The error constant C differs between methods and between problems.

Third, the largest step sizes on P3 expose stability and not accuracy. The system has eigenvalues $-1 \pm 3i$. From the amplification factors of Section 4.4, Euler is stable for this pair only when $h < 0.200$ and ABM4 only when $h < 0.283$, whereas Heun and RK4 remain stable up to $h = 0.532$ and $h = 0.890$. This accounts for the Euler errors of 8.95×10^3 and 10.5 at $h = 0.5$ and 0.25 and the ABM4 error of 8.26×10^4 at $h = 0.5$, against an exact solution that never exceeds 1.351. It also explains why the observed orders for Euler and ABM4 on P3 are still approaching their limiting values. High order does not bring a large stability region: ABM4 has the smallest of the four.

7.2 Computational work at equal accuracy

At a fixed accuracy of 10^{-6} the fourth-order methods needed tens to hundreds of steps, Heun's method thousands, and Euler's method hundreds of thousands to millions (Table 4).

Table 4. Work needed to reach a scaled maximum error of 10^{-6}

Problem	Method	Steps n	Step size h	Evaluations of f	Ratio to RK4	Time (ms)
P1 Cooling	Euler	$\approx 3.99 \times 10^5$	7.53×10^{-5}	$\approx 3.99 \times 10^5$	4,529	not run
P1 Cooling	Heun	633	0.04739	1,266	14.39	4.05
P1 Cooling	RK4	22	1.364	88	1.00	0.31
P1 Cooling	ABM4	29	1.034	65	0.74	0.46
P2 Logistic	Euler	$\approx 4.05 \times 10^6$	4.93×10^{-6}	$\approx 4.05 \times 10^6$	12,066	not run
P2 Logistic	Heun	3,625	0.005517	7,250	21.58	33.53
P2 Logistic	RK4	84	0.2381	336	1.00	1.64
P2 Logistic	ABM4	80	0.25	167	0.50	1.47
P3 RLC	Euler	$\approx 1.43 \times 10^7$	6.98×10^{-7}	$\approx 1.43 \times 10^7$	12,063	not run
P3 RLC	Heun	12,012	0.0008325	24,024	20.22	62.92
P3 RLC	RK4	297	0.03367	1,188	1.00	3.64
P3 RLC	ABM4	401	0.02494	809	0.68	6.05

Euler values are estimated from the measured first-order error constant (Section 6). Times are from a single machine and are indicative only.

The cheapest method per step is by far the most expensive per digit. Euler uses one evaluation per step and RK4 four, yet at this accuracy Euler needs roughly 4,500 to 12,000 times as many evaluations as RK4, and Heun 14 to 22 times as many. The advantage of a high-order method grows as the accuracy requirement tightens, because the work scales as $(1/\text{tolerance})^{1/p}$.

ABM4 used 26% to 50% fewer evaluations than RK4 on all three problems, because it needs two evaluations per step against four. On P3 it needed more steps than RK4 (401 against 297) and still used fewer evaluations. The saving did not appear in the measured times for P1 and P3. The right-hand sides here are trivial, so the bookkeeping of the multistep formula in an interpreted language outweighs the evaluations saved. The saving becomes real when f is expensive, as in large reaction networks or discretised partial differential equations.

Figure 2 shows the same comparison over the whole range of accuracy for P2. On logarithmic axes each method traces a straight line of slope $-p$. The fourth-order lines fall four decades in error for every decade of work, the Euler line only one, so the gap between them widens as the accuracy requirement tightens.

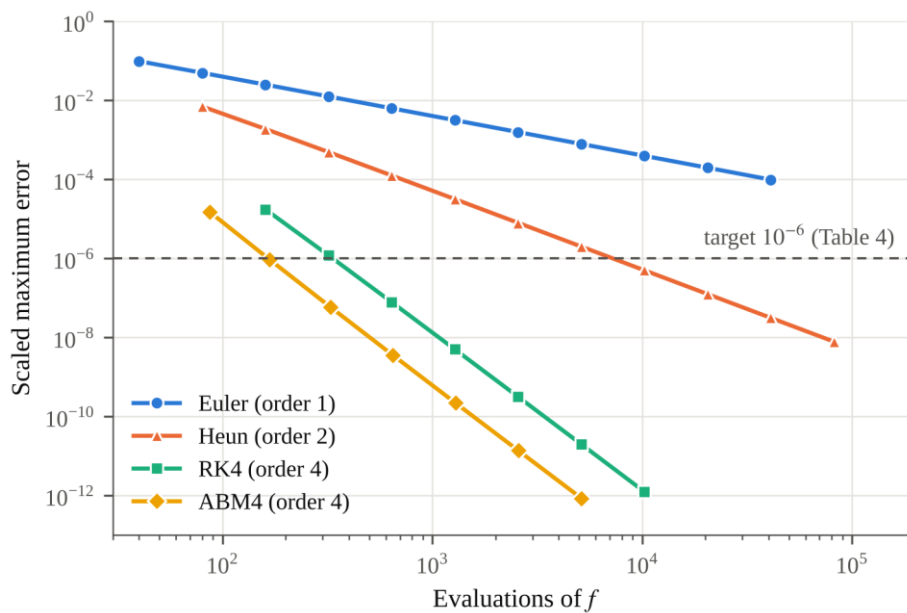


Figure 2. Work–precision diagram for logistic growth (P2): scaled maximum error against number of evaluations of f . Each point is one fixed-step run, with h halved from 0.5.

The adaptive library solvers were more economical again. With a relative tolerance of 10^{-8} , DOP853 used 86, 242 and 782 evaluations on P1, P2 and P3 and returned scaled errors below 9×10^{-10} in every case. RK45 used 176, 416 and 1,640 evaluations with errors below 2.5×10^{-9} . DOP853 delivered about a thousand times the accuracy of the fixed-step RK4 runs in Table 4 for fewer evaluations, and RK45 did so for 1.2 to 2 times as many. The gain comes from higher order and from step-size control, which places steps where the solution changes fastest.

7.3 The stiff problem

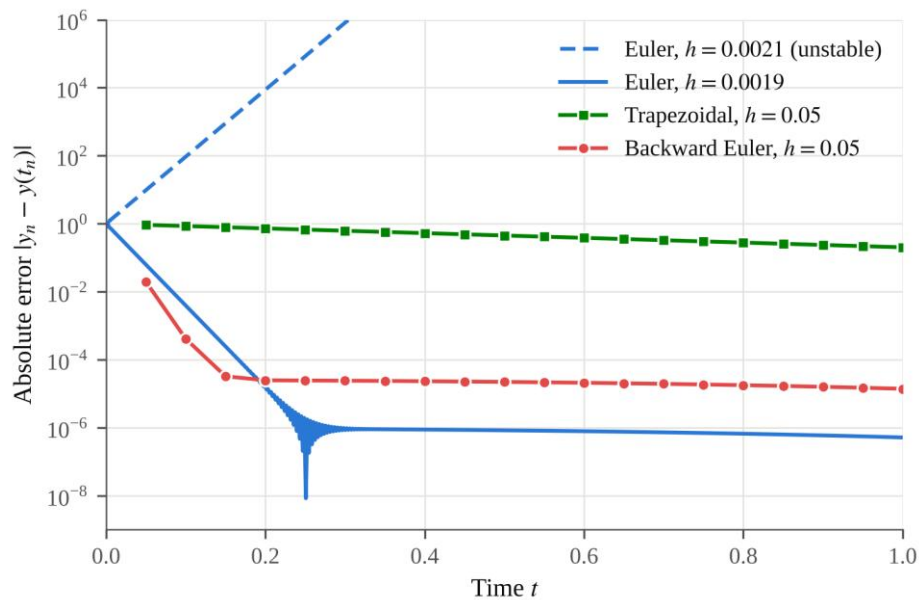
On P4 the explicit methods failed at every step size that accuracy would have allowed, and the implicit methods did not (Table 5).

Table 5. Absolute error at $t = 1$ for the stiff problem P4 ($\lambda = -1000$)

Step size h	Steps	Euler	RK4	Backward Euler	Trapezoidal
0.1	10	9.0×10^{19}	1.1×10^{66}	2.8×10^{-5}	6.7×10^{-1}
0.01	100	2.7×10^{95}	2.5×10^{246}	2.7×10^{-6}	7.0×10^{-9}
0.0025	400	2.7×10^{70}	1.6×10^{-6}	6.8×10^{-7}	4.4×10^{-10}
0.002	500	1.0	2.7×10^{-7}	5.4×10^{-7}	2.8×10^{-10}
0.001	1,000	2.7×10^{-7}	9.0×10^{-9}	2.7×10^{-7}	7.0×10^{-11}
0.0001	10,000	2.7×10^{-8}	5.9×10^{-13}	2.7×10^{-8}	7.0×10^{-13}

The results agree with the stability limits of Table 2. Euler requires $h < 2/1000 = 0.002$. At exactly $h = 0.002$ its amplification factor is -1 , so the initial discrepancy of size 1 is carried unchanged to $t = 1$ and appears as an error of 1.0. RK4 requires $h < 2.785/1000$ and is therefore stable at $h = 0.0025$ but not at $h = 0.01$. Backward Euler with ten steps was accurate to 2.8×10^{-5} . Explicit Euler needed more than 500 steps merely to remain stable, and those steps were forced by a transient that had vanished by $t = 0.01$.

The trapezoidal method shows that A-stability alone is not sufficient. At $h = 0.1$ its amplification factor is $(1 - 50)/(1 + 50) = -0.961$. The transient is damped by only 4% per step, and after ten steps $0.961^{10} = 0.670$ of it remains, which is exactly the error of 6.7×10^{-1} in Table 5. Backward Euler, which is L-stable, has amplification factor $1/101$ at the same step and removes the transient in a single step. Once h is small enough to damp the transient, the second-order accuracy of the trapezoidal method makes it the most accurate of the four at every remaining step size except the smallest, where RK4 matches it. Figure 3 follows the error through time for four representative runs.

**Figure 3.** Absolute error against time for the stiff problem P4: explicit Euler just above and just below its stability limit $h = 0.002$, and the two implicit methods at $h = 0.05$.

In Figure 3, explicit Euler at $h = 0.0021$, just above its limit, has amplification factor -1.1 and its error grows by 10% per step without bound. At $h = 0.0019$ the factor is -0.9 ; the

transient decays and the error settles just below 10^{-6} . Backward Euler at $h = 0.05$, a step 26 times larger, removes the transient within three steps and holds the error near 2×10^{-5} . The trapezoidal method at the same step has factor -0.923 and still carries an error of 0.20 at $t = 1$.

Table 6 gives the corresponding results for adaptive solvers.

Table 6. Adaptive solvers on the stiff problem P4 (relative tolerance 10^{-6} , absolute tolerance 10^{-9})

Solver	Class	Steps	Evaluations of f	Jacobian evaluations	Error at $t = 1$
RK45	Explicit	378	2,270	0	2.0×10^{-7}
DOP853	Explicit	180	2,942	0	3.9×10^{-9}
Radau	Implicit	57	422	2	9.1×10^{-10}
BDF	Implicit	116	264	2	7.7×10^{-10}
LSODA	Switching	100	195	5	1.1×10^{-7}

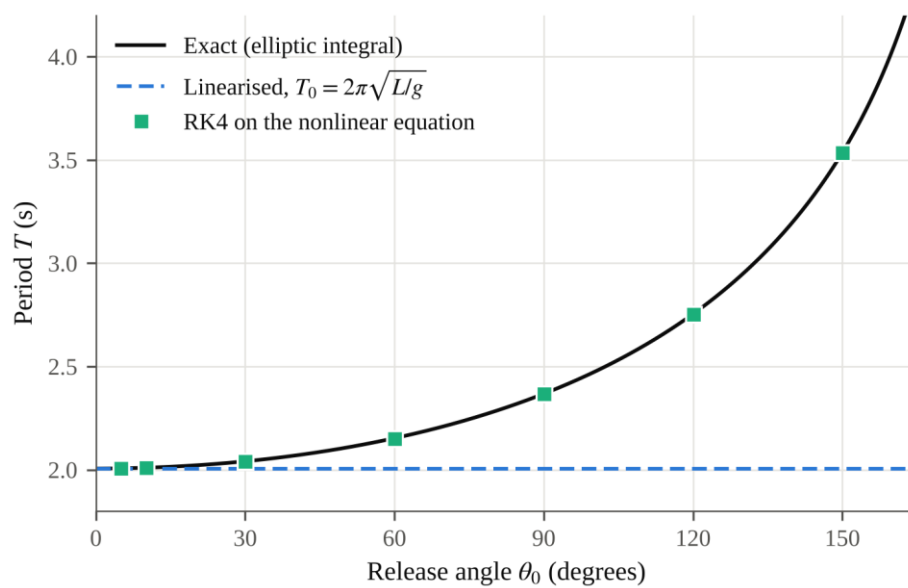
The explicit adaptive solvers did not diverge, because step-size control detects the growing error and shortens the step. They paid for stability instead: RK45 took 378 steps, an average step of 0.0026 , which is of the same size as the explicit stability limits in Table 2 and far below what the smooth solution cost requires. BDF and LSODA needed 264 and 195 evaluations, 9 to 12 times fewer than RK45. The ratio grows roughly in proportion to $|\lambda|$, since the explicit step is bounded by a multiple of $1/|\lambda|$ while the implicit step is not.

7.4 The nonlinear pendulum: limits of the linearised analytical solution

The linearised formula $T_0 = 2.006067$ s underestimates the period at every amplitude, by less than 0.2% up to 10° and by more than 15% at 90° (Table 7 and Figure 4).

Table 7. Period of the simple pendulum ($g = 9.81 \text{ m s}^{-2}$, $L = 1 \text{ m}$; linearised period $T_0 = 2.006067 \text{ s}$ at all amplitudes)

Release angle θ_0	Exact period T (s)	Error of linearised period (%)	Difference between RK4 and exact period (s)
5°	2.007022	0.05	1.6×10^{-11}
10°	2.009893	0.19	1.4×10^{-10}
30°	2.040990	1.71	8.6×10^{-12}
60°	2.152875	6.82	5.2×10^{-11}
90°	2.367842	15.28	9.0×10^{-13}
120°	2.754090	27.16	6.3×10^{-11}
150°	3.535098	43.25	7.0×10^{-11}

**Figure 4.** Period of the simple pendulum against release angle: exact elliptic-integral result, linearised formula, and RK4 solution of the nonlinear equation.

The numerical solution of the full nonlinear equation agreed with the elliptic-integral period to within $1.5 \times 10^{-10} \text{ s}$ at every amplitude. In this model the roles of the two approaches are reversed. The simple analytical formula is the approximation, and the numerical method is, for practical purposes, exact. The source of analytical error is the model simplification made to obtain a closed form, and no refinement of the algebra can remove it.

Analysis still contributes two things. The elliptic-integral formula is what certifies the numerical result, and its series expansion $T \approx T_0 (1 + \theta_0^2/16 + \dots)$ explains why the error grows as the square of the amplitude: the first correction term alone is 1.7% of T_0 at 30° and 6.9% at 60°.

Long-time behaviour separates the methods further. Over 100 s at $\theta_0 = 60^\circ$ and $h = 0.01 \text{ s}$, about 46 periods, the largest relative deviation of the energy was 9.4×10^{-8} for RK4, 2.5×10^{-6} for ABM4 and 1.8×10^{-3} for Heun. Euler's method gained energy continuously. After 25.8 s its energy exceeded the value needed to pass through the vertical, and from then on the computed pendulum rotated instead of oscillating. The result is wrong in kind as well as in

size. None of the four methods conserves energy exactly, which is the motivation for the structure-preserving integrators described by Hairer et al. (2006).

7.5 Comparative assessment

Table 8 sets the two approaches against each other on the criteria the experiments address.

Table 8. Analytical and numerical approaches compared

Criterion	Analytical methods	Numerical methods
Applicability	Special classes only: separable, linear, exact, constant-coefficient	Almost any $f(t, y)$, including nonlinear systems
Form of result	A formula valid for all t and all parameter values	A table of values for one set of parameters
Error	None in the solution; possibly large in the model if it was simplified (up to 43% in Table 7)	Controllable through h and order; reduced to 10^{-11} in Table 3
Dependence on parameters	Explicit: time constant $1/k$, inflection at $K/2$, overshoot $e^{-\pi/3}$	Found only by repeated runs
Cost	Human effort, once	Function evaluations, every run; differs by about four orders of magnitude between methods (Table 4)
Principal risk	Solving a simplified model that no longer represents the system	Instability or an unsuitable method (Tables 3 and 5)
Role in the other approach	Supplies exact test cases and stability theory	Checks the range of validity of approximations

The results support the following guidance.

1. If the model belongs to a solvable class, solve it analytically first. The formula is free of discretisation error and shows the parameter dependence.
2. If a closed form is available only after simplifying the model, check the simplification numerically over the range of interest before relying on it.
3. For smooth non-stiff problems use a method of order four or higher. Euler's method is suitable for explaining the idea of time-stepping and for little else.
4. Prefer an adaptive solver with error control to any fixed step size.
5. If an explicit adaptive solver takes many small steps on a solution that looks smooth, the problem is stiff. Switch to an implicit solver such as BDF or Radau, or to one that switches automatically.
6. Validate every numerical code on a problem with a known analytical solution and confirm that the observed order equals the theoretical order.

8. Conclusion

Analytical and numerical methods for ordinary differential equations answer different questions and are strongest when used together. This study compared them on five models with exact solutions and reached four conclusions. First, order of accuracy governs efficiency: Euler, Heun, RK4 and ABM4 converged at observed orders of 1, 2, 4 and 4,

respectively. To reach a scaled error of [value missing], RK4 required 88–1,188 function evaluations, whereas Heun required 14–22 times as many and Euler an estimated 4,500–12,000 times as many. ABM4 required 26%–50% fewer evaluations than RK4. Second, stability remains a separate requirement from accuracy. On the stiff problem, explicit methods diverged unless the step size was below [Euler threshold missing] for Euler or [RK4 threshold missing] for RK4, while backward Euler achieved an accuracy of [value missing] with ten steps. A-stability alone was insufficient: the trapezoidal method retained 67% of the initial transient at [value missing]. Among adaptive solvers, implicit and switching codes required 9–12 times fewer evaluations than explicit RK45. Third, a closed-form solution is only as reliable as the model it represents. The linearised pendulum period differed from the exact value by less than 0.2% up to 10° , but the error increased to 6.8% at 60° and 15.3% at 90° . RK4 applied to the full nonlinear equation matched the exact period to within [value missing] s. Finally, the two approaches provided complementary validation: exact solutions enabled measurement of convergence order, detection of instability and verification of the numerical pendulum period, while numerical solutions established the range over which the linearised formula remained reliable. For teaching and practice, models should therefore be solved analytically wherever possible, numerically wherever necessary, and checked using both approaches whenever available.

References

- Ali, A. J., Abbas, A. F., & Abdelhakem, M. A. (2024). Comparative analysis of Adams-Bashforth-Moulton and Runge-Kutta methods for solving ordinary differential equations using MATLAB. *Mathematical Modelling of Engineering Problems*, 11(3), 641–647. <https://doi.org/10.18280/mmep.110307>
- Atkinson, K. E., Han, W., & Stewart, D. E. (2009). *Numerical solution of ordinary differential equations*. Wiley.
- Beléndez, A., Pascual, C., Méndez, D. I., Beléndez, T., & Neipp, C. (2007). Exact solution for the nonlinear pendulum. *Revista Brasileira de Ensino de Física*, 29(4), 645–648.
- Boyce, W. E., & DiPrima, R. C. (2012). *Elementary differential equations and boundary value problems* (10th ed.). Wiley.
- Burden, R. L., & Faires, J. D. (2011). *Numerical analysis* (9th ed.). Brooks/Cole, Cengage Learning.
- Butcher, J. C. (2016). *Numerical methods for ordinary differential equations* (3rd ed.). Wiley.
- Curtiss, C. F., & Hirschfelder, J. O. (1952). Integration of stiff equations. *Proceedings of the National Academy of Sciences*, 38(3), 235–243. <https://doi.org/10.1073/pnas.38.3.235>
- Dahlquist, G. G. (1963). A special stability problem for linear multistep methods. *BIT Numerical Mathematics*, 3(1), 27–43. <https://doi.org/10.1007/BF01963532>
- Denis, B. (2020). *An overview of numerical and analytical methods for solving ordinary differential equations*. arXiv. <https://arxiv.org/abs/2012.07558>
- Dormand, J. R., & Prince, P. J. (1980). A family of embedded Runge–Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1), 19–26. [https://doi.org/10.1016/0771-050X\(80\)90013-3](https://doi.org/10.1016/0771-050X(80)90013-3)
- Gear, C. W. (1971). *Numerical initial value problems in ordinary differential equations*. Prentice-Hall.

- Hairer, E., Lubich, C., & Wanner, G. (2006). *Geometric numerical integration: Structure-preserving algorithms for ordinary differential equations* (2nd ed.). Springer.
- Hairer, E., Nørsett, S. P., & Wanner, G. (1993). *Solving ordinary differential equations I: Nonstiff problems* (2nd ed.). Springer.
- Hairer, E., & Wanner, G. (1996). *Solving ordinary differential equations II: Stiff and differential-algebraic problems* (2nd ed.). Springer.
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585, 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Hindmarsh, A. C. (1983). ODEPACK, a systematized collection of ODE solvers. In R. S. Stepleman (Ed.), *Scientific computing* (pp. 55–64). North-Holland.
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- Kreyszig, E. (2011). *Advanced engineering mathematics* (10th ed.). Wiley.
- Kutta, W. (1901). Beitrag zur näherungsweise Integration totaler Differentialgleichungen. *Zeitschrift für Mathematik und Physik*, 46, 435–453.
- Lambert, J. D. (1991). *Numerical methods for ordinary differential systems: The initial value problem*. Wiley.
- Murray, J. D. (2002). *Mathematical biology I: An introduction* (3rd ed.). Springer.
- Petzold, L. (1983). Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. *SIAM Journal on Scientific and Statistical Computing*, 4(1), 136–148. <https://doi.org/10.1137/0904010>
- Prothero, A., & Robinson, A. (1974). On the stability and accuracy of one-step methods for solving stiff systems of ordinary differential equations. *Mathematics of Computation*, 28(125), 145–162.
- Runge, C. (1895). Über die numerische Auflösung von Differentialgleichungen. *Mathematische Annalen*, 46(2), 167–178. <https://doi.org/10.1007/BF01446807>
- Salleh, I., Ab Aziz, A., & Baki, N. A. (2024). Comparison between Runge-Kutta method, Euler method and modified Euler method for difference order of ordinary differential equation. *Journal of Advanced Mechanical Engineering Applications*, 5(2), 24–29.
- Shampine, L. F., & Reichelt, M. W. (1997). The MATLAB ODE suite. *SIAM Journal on Scientific Computing*, 18(1), 1–22. <https://doi.org/10.1137/S1064827594276424>
- Söderlind, G., Jay, L., & Calvo, M. (2015). Stiffness 1952–2012: Sixty years in search of a definition. *BIT Numerical Mathematics*, 55(2), 531–558. <https://doi.org/10.1007/s10543-014-0503-3>
- Strogatz, S. H. (2015). *Nonlinear dynamics and chaos: With applications to physics, biology, chemistry, and engineering* (2nd ed.). Westview Press.
- Verhulst, P.-F. (1838). Notice sur la loi que la population suit dans son accroissement. *Correspondance Mathématique et Physique*, 10, 113–121.
- Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17, 261–272. <https://doi.org/10.1038/s41592-019-0686-2>