Homework #3

Due 11:59pm, Sunday, Feb 27, 2022.

1 Solve the Lorenz equations with a Taylor series method

Consider the following initial value problem (Lorenz).

$$Y' = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = f(t, Y) = \begin{bmatrix} 10y - 10x \\ 28x - y - xz \\ -\frac{8}{5}z + xy \end{bmatrix}, \qquad Y(0) = \begin{bmatrix} x(0) \\ y(0) \\ z(0) \end{bmatrix} = \begin{bmatrix} -4 \\ -8 \\ 15 \end{bmatrix}.$$

(a) Implement the Taylor series method to at least 4th order in Python to approximate the solution of the IVP for $t \in [0, 3]$. By my computations, $y(3) \approx -9.33907$: you can use this to check yours.

Choose several values of the number of steps to check the convergence of the approximation as $h \to 0$.

Make plots of x, y, z versus t, and also of the three projections y versus x and z versus x and z versus y. Compare your results with those you obtained with Euler's method in Homework 1 and comment.

(b) Generate the xz plot of the solution for a longer interval of time, t.

2. Deriving the family of 3rd-order 3-stage Runge-Kutta methods

Ackleh et al. Exercise 7.10.12

(a) Find the equations that must be satisfied by the coefficients of a 3rd-order 3-stage Runge-Kutta method.

(b) Find a particular set of coefficients that satisfies these equations (ideally one that is not in Wikipedia).

Suggestions and hints

Get a large quantity of paper, pencil and eraser, and a cup of strong coffee for this one: it is a heavy calculation.

Do it as follows. Consider the generic i^{th} component of the solution of a 2D autonomous system $y' = f(y)$. Making it autonomous reduces the complexity of the expressions, and doing it for 2D only keeps the expressions manageable but it will still be clear how they generalize to n dimensions.

The idea is to write Taylor expansions for the values of f_i in the Runge-Kutta formula, and match these with the Taylor expansion of $y_i(t_k + h)$. Your terms of your expansions will all be products of the RK coefficients, derivatives of the components f_j of f, and the components f_j themselves - all evaluated at the same point, so you don't need to waste time and ink writing "$(y^{\{k\}})$" to say they are evaluated at $y^{\{k\}}$, the current latest point of our numerical solution. I recommend using the compact notation $\partial_p f_q$ to denote the derivative of the q^{th} component of $f(y)$ with respect to the p^{th} component of y (at the point $y^{\{k\}}$).

The ingredients you will need are: (i) the differential equation itself, (ii) the chain rule of partial differentiation, and (iii) the multivariable Taylor expansion as given in Wikipedia, written out explicitly for 2 variables. You will not need to go beyond 2nd partials of the components of f. The key thing you need is:

$$f_i(y + v) \approx f_i + \partial_1 f_i \cdot v_1 + \partial_2 f_i \cdot v_2 + \frac{1}{2!}\left(\partial_1^2 f_i \cdot v_1^2 + 2\partial_1 \partial_2 f_i \cdot v_1 v_2 + \partial_2^2 f_i \cdot v_2^2\right)$$

where everything on the RHS is evaluated at y.

Try not to waste time figuring out terms that will contribute at order higher than h^3.