Simulation of radioactive decay: stiffness in ODE

Gabriela Pyda

16 November 2025

1 Introduction

The law of radioactive decay dictates that the rate at which an unstable atomic nucleus transforms into a more stable form is directly proportional to the number of unstable nuclei present. This fundamental principle is described by the equation $\frac{dN(t)}{dt} = -\lambda N(t)$, where $N(t)$ represents the amount (atomic nuclei or mass) of the unstable element at time t, and λ is the characteristic decay constant, often related to the half-life of the substance.

Many heavy radioactive isotopes undergo a series of transformations, forming a chain of descendant isotopes. These descendants can also be radioactive, leading to a "radioactive family" or decay chain. For a system involving three radioactive elements and one stable final product, the rates of decay are interconnected and can be described by a system of linear ordinary differential equations (SODE). This system models the changing amounts of each isotope over time.

Specifically, for a three-element family where the fourth element is stable, the system of linear ordinary differential equations (SODE) is given by the Formula (1), where $N_0(t)$, $N_1(t)$, and $N_2(t)$ represent the amounts of the initial isotope, the first daughter isotope, and the second daughter isotope, respectively, at time t. The constants λ_0, λ_1, and λ_2 are their respective decay constants. This system can also be written in a more compact vector form as $\frac{d\vec{N}}{dt} = \vec{f}(t, \vec{N})$, where $\vec{N} = [N_0, N_1, N_2]^T$ and $\vec{f}(t, \vec{N}) = [f_0, f_1, f_2]^T$ represented by the Formula (2).

$$\begin{cases} \frac{dN_0}{dt} = -\lambda_0 N_0 \\ \frac{dN_1}{dt} = \lambda_0 N_0 - \lambda_1 N_1 \\ \frac{dN_2}{dt} = \lambda_1 N_1 - \lambda_2 N_2 \end{cases} \tag{1}$$

$$\begin{cases} f_0 = -\lambda_0 N_0 \\ f_1 = \lambda_0 N_0 - \lambda_1 N_1 \\ f_2 = \lambda_1 N_1 - \lambda_2 N_2 \end{cases} \tag{2}$$

The exact solution for this SODE, calculated in the Formulae (3) - 5, can be derived analytically, providing explicit formulas for $N_0(t)$, $N_1(t)$, and $N_2(t)$ based on their initial values $N_0(0)$, $N_1(0)$, and $N_2(0)$. These exact solutions are crucial for verifying the accuracy of numerical methods.

$$N_0(t) = N_0(0)e^{-\lambda_0 t} \tag{3}$$

$$N_1(t) = \frac{\lambda_0 N_0(0)}{\lambda_0 - \lambda_1} e^{-\lambda_0 t} + \frac{\lambda_0 [N_0(0) + N_1(0)] - \lambda_1 N_1(0)}{\lambda_1 - \lambda_0} e^{-\lambda_1 t} \tag{4}$$

$$\begin{aligned} N_2(t) = {} & \frac{\lambda_0 \lambda_1 N_0(0)}{(\lambda_0 - \lambda_1)(\lambda_2 - \lambda_0)} e^{-\lambda_0 t} + \frac{\lambda_1 [\lambda_0 (N_0(0) + N_1(0)) - \lambda_1 N_1(0)]}{(\lambda_0 - \lambda_1)(\lambda_2 - \lambda_1)} e^{-\lambda_1 t} \\ & + \frac{\lambda_0 \lambda_1 [N_0(0) + N_1(0) + N_2(0)] - \lambda_0 \lambda_2 N_2(0) - \lambda_1 \lambda_2 [N_1(0) + N_2(0)] + \lambda_2^2 N_2(0)}{(\lambda_2 - \lambda_1)(\lambda_2 - \lambda_0)} e^{-\lambda_2 t} \end{aligned} \tag{5}$$

The numerical solution of the SODE given in Formula (1) is the primary goal of this project. Due to the linear nature of the SODE, it can be expressed in matrix form as $\frac{d\vec{N}}{dt} = A\vec{N}(t)$. The properties of the matrix A, specifically its eigenvalues $\lambda_0, \lambda_1, \lambda_2$, determine the "stiffness" of the system. If there is a significant disparity between these eigenvalues (i.e., $\lambda_i \gg \lambda_j$ for at least one pair of indices), the SODE is considered stiff. Stiff systems pose numerical challenges and require specialized methods for stable and accurate solutions. This project will employ an implicit trapezoidal scheme combined with Newton iteration and an adaptive time step mechanism to address the stiffness.

2 Numerical algorithms

2.1 Adaptive time step

Numerical calculations involving ordinary differential equations often benefit greatly from adaptive time step control. This approach eliminates the need for manual adjustment of the time step, as it automatically changes the step size based on predefined constraints, ensuring both accuracy and computational efficiency. The core idea is to estimate the error produced by a given time step and then adjust the step size to keep this error within a specified tolerance.

The adaptive time step algorithm works by comparing the results of two different numerical approximations for the same interval: one using a full time step Δt, and another using two half-steps, $\Delta t/2$. First, the solution is calculated for a single full time step, yielding an approximation, let's call it $\vec{N}^{(1)}$. Next, the solution is calculated using two successive half-steps. This involves calculating an intermediate solution using $\Delta t/2$ from the current state, and then using this intermediate solution to calculate the final state after another $\Delta t/2$. This yields a second, generally more accurate, approximation, $\vec{N}^{(2)}$. The difference between these two approximations, specifically $\vec{\epsilon} = \frac{\vec{N}^{(2)} - \vec{N}^{(1)}}{2^p - 1}$ (where p is the order of the method, typically 2 for the trapezoidal rule), provides an estimate of the local error. The maximal component of this error, $\epsilon_{max} = \max\{|\epsilon_0|, |\epsilon_1|, |\epsilon_2|\}$, is then compared against a user-defined tolerance (TOL).

Based on this comparison, a new time step Δt_{new} is calculated using the Formula (6), where S is a safety factor (e.g., 0.9) to prevent oscillations and ensure a conservative step size.

$$\Delta t_{new} = S \cdot \Delta t \cdot \left(\frac{TOL}{\epsilon_{max}} \right)^{\frac{1}{p+1}} \tag{6}$$

If ϵ_{max} is too large, Δt_{new} will be smaller than the current Δt, indicating that a smaller step is needed for the next iteration. Conversely, if ϵ_{max} is much smaller than TOL, Δt_{new} will be larger, allowing the algorithm to take bigger steps and accelerate the computation. The algorithm then proceeds to the next iteration using this newly calculated time step, ensuring that the error remains within acceptable bounds throughout the simulation.

2.2 Trapezoidal method

The trapezoidal method is an implicit, second-order numerical method used for solving ordinary differential equations (ODEs). It is particularly well-suited for stiff systems due to its excellent stability properties. The method averages the slopes at the beginning and end of a time interval to estimate the average rate of change, making it more accurate than explicit methods like the Euler method for the same step size.

For a system of ODEs given by $\frac{d\vec{N}}{dt} = \vec{f}(t, \vec{N})$, the trapezoidal rule advances the solution from $\vec{N}_i$ at time t_i to $\vec{N}_{i+1}$ at time $t_{i+1} = t_i + \Delta t$ using the Formula (7), where Δt is the time step.

$$\vec{N}_{i+1} = \vec{N}_i + \frac{\Delta t}{2}\left(\vec{f}(t_i, \vec{N}_i) + \vec{f}(t_{i+1}, \vec{N}_{i+1})\right) \tag{7}$$

A key characteristic of the trapezoidal method is that $\vec{N}_{i+1}$ appears on both sides of the Formula (7), making it an implicit method. This means that to find $\vec{N}_{i+1}$, one cannot simply compute it directly. Instead, $\vec{N}_{i+1}$ must be solved for, typically through iterative techniques. This implicit nature contributes to its stability, but also requires more computational effort per step compared to explicit methods. For nonlinear systems or stiff linear systems (like the radioactive decay system), solving for $\vec{N}_{i+1}$ at each step usually involves a root-finding algorithm, such as Newton's method described in the next section.

2.3 Newton iterations

As established, the trapezoidal method (Formula (7)) is an implicit scheme, meaning that $\vec{N}_{i+1}$ appears on both sides of the equation. To find the solution $\vec{N}_{i+1}$ at the next time step, we need to solve a system of non-linear equations. This is where Newton's method, specifically adapted for multi-dimensional nonlinear functions, becomes indispensable.

First, we rewrite the Formula (7) into a form where we are seeking the root of a function. Let's define a function $\vec{F}(\vec{N}_{i+1})$ such that finding its root means finding $\vec{N}_{i+1}$ (Formula (8)). Our goal is to find $\vec{N}_{i+1}$ such that $\vec{F}(\vec{N}_{i+1}) = \vec{0}$. For the specific system of radioactive decay (Equations (1) and (2)), we can write out the components of $\vec{F}(\vec{N}_{i+1}) = [F_0, F_1, F_2]^T$ in the Formula (9).

$$\vec{F}(\vec{N}_{i+1}) = \vec{N}_{i+1} - \vec{N}_i - \frac{\Delta t}{2}\left(\vec{f}(t_i, \vec{N}_i) + \vec{f}(t_{i+1}, \vec{N}_{i+1})\right) \tag{8}$$

$$\begin{cases} F_0 = N_{0,i+1} - N_{0,i} - \frac{\Delta t}{2}\left((-\lambda_0 N_{0,i}) + (-\lambda_0 N_{0,i+1})\right) \\ F_1 = N_{1,i+1} - N_{1,i} - \frac{\Delta t}{2}\left((\lambda_0 N_{0,i} - \lambda_1 N_{1,i}) + (\lambda_0 N_{0,i+1} - \lambda_1 N_{1,i+1})\right) \\ F_2 = N_{2,i+1} - N_{2,i} - \frac{\Delta t}{2}\left((\lambda_1 N_{1,i} - \lambda_2 N_{2,i}) + (\lambda_1 N_{1,i+1} - \lambda_2 N_{2,i+1})\right) \end{cases} \tag{9}$$

Newton's method is an iterative procedure. Starting with an initial guess $\vec{N}_{i+1}^{(k)}$ (where k denotes the iteration number), it refines this guess using the Formula (10). The term $\frac{\partial \vec{F}}{\partial \vec{N}_{i+1}}$ is the Jacobian matrix of $\vec{F}$ with respect to $\vec{N}_{i+1}$, denoted as $\mathbf{G}$. The elements of this matrix are given by $G_{lm} = \frac{\partial F_l}{\partial N_{m,i+1}}$. For our three-component system, $\mathbf{G}$ is a 3×3 matrix present in the Formula (11). Calculating its partial derivatives from the Formula (9), we get the final matrix in the form from the Formula (12).

$$\vec{N}_{i+1}^{(k+1)} = \vec{N}_{i+1}^{(k)} - \left[\frac{\partial \vec{F}}{\partial \vec{N}_{i+1}} \left(\vec{N}_{i+1}^{(k)} \right) \right]^{-1} \vec{F}\left(\vec{N}_{i+1}^{(k)} \right) \tag{10}$$

$$\mathbf{G} = \frac{\partial \vec{F}}{\partial \vec{N}_{i+1}} = \begin{pmatrix} \frac{\partial F_0}{\partial N_{0,i+1}} & \frac{\partial F_0}{\partial N_{1,i+1}} & \frac{\partial F_0}{\partial N_{2,i+1}} \\ \frac{\partial F_1}{\partial N_{0,i+1}} & \frac{\partial F_1}{\partial N_{1,i+1}} & \frac{\partial F_1}{\partial N_{2,i+1}} \\ \frac{\partial F_2}{\partial N_{0,i+1}} & \frac{\partial F_2}{\partial N_{1,i+1}} & \frac{\partial F_2}{\partial N_{2,i+1}} \end{pmatrix} \tag{11}$$

$$\mathbf{G} = \begin{pmatrix} 1 - \frac{\Delta t}{2}(-\lambda_0) & 0 & 0 \\ -\frac{\Delta t}{2}\lambda_0 & 1 - \frac{\Delta t}{2}(-\lambda_1) & 0 \\ 0 & -\frac{\Delta t}{2}\lambda_1 & 1 - \frac{\Delta t}{2}(-\lambda_2) \end{pmatrix} \tag{12}$$

Notice that the Jacobian matrix for this specific linear system is constant during the Newton iterations within a single time step, as it does not depend on $\vec{N}_{i+1}^{(k)}$. This simplifies the process considerably.

The Newton iteration effectively solves the linear system $\mathbf{G}\Delta\vec{N} = -\vec{F}(\vec{N}_{i+1}^{(k)})$ for the update vector $\Delta\vec{N}$, and then updates the solution with $\vec{N}_{i+1}^{(k+1)} = \vec{N}_{i+1}^{(k)} + \Delta\vec{N}$. The initial guess for $\vec{N}_{i+1}^{(0)}$ is typically taken as the solution from the previous time step, i.e., $\vec{N}_i$. The iterations continue until the magnitude of $\Delta\vec{N}$ (or ϵ_{max} as defined for the adaptive time step) falls below a predefined tolerance, or a maximum number of iterations is reached. The update vector $\Delta\vec{N} = [\Delta N_0, \Delta N_1, \Delta N_2]^T$ can be obtained either by numerically solving the linear system $\mathbf{G}\Delta\vec{N} = -\vec{F}(\vec{N}_{i+1}^{(k)})$ using a linear algebra routine, or by analytically inverting the 3×3 matrix $\mathbf{G}$ and then multiplying by $-\vec{F}$. The analytical solution for $\Delta N_0, \Delta N_1, \Delta N_2$ involves computing the determinant of $\mathbf{G}$ and using Cramer's rule or explicit inverse Formulae (13) - (15), where g_{lm} are the elements of the matrix $\mathbf{G}$ from the Formula (12), and the determinant is given by the Formula (16).

$$\Delta N_0 = \frac{-F_0 g_{11} g_{22} + F_0 g_{12} g_{21} + F_1 g_{01} g_{22} - F_1 g_{02} g_{21} - F_2 g_{01} g_{12} + F_2 g_{02} g_{11}}{\det(\mathbf{G})} \tag{13}$$

$$\Delta N_1 = \frac{-F_0 g_{10} g_{22} + F_0 g_{12} g_{20} - F_1 g_{00} g_{22} + F_1 g_{02} g_{20} + F_2 g_{00} g_{12} - F_2 g_{02} g_{10}}{\det(\mathbf{G})} \tag{14}$$

$$\Delta N_2 = \frac{-F_0 g_{10} g_{21} + F_0 g_{11} g_{20} + F_1 g_{00} g_{21} - F_1 g_{01} g_{20} - F_2 g_{00} g_{11} + F_2 g_{01} g_{10}}{\det(\mathbf{G})} \tag{15}$$

$$\det(\mathbf{G}) = g_{00} g_{11} g_{22} - g_{00} g_{12} g_{21} - g_{01} g_{10} g_{22} + g_{01} g_{12} g_{20} + g_{02} g_{10} g_{21} - g_{02} g_{11} g_{20} \tag{16}$$

The values of F_0, F_1, F_2 are evaluated using the current iterative guess $\vec{N}_{i+1}^{(k)}$ in the Formula (9). This iterative process ensures that the implicit trapezoidal equation is accurately solved at each time step, making it robust for handling stiff systems.

3 Results

Each simulation was performed with use of initial parameters $\Delta t = 10^{-2}$, $N_0 = 1$, $N_1 = N_2 = 0$, $tolerance = 10^{-10}$ and $t_{max} = 200$.

3.1 Test of correctness

The first experiment included testing the correctness of implemented radioactive decay phenomena, by setting the parameters $\lambda_0 = 1$, $\lambda_1 = 5$, $\lambda_2 = 50$ and $TOL = 10^{-4}$. The distribution of numbers of particular particles $N_x(t)$ is presented in Figure 1. Additionally, the change of Δt over time is visualized in Figure 2.

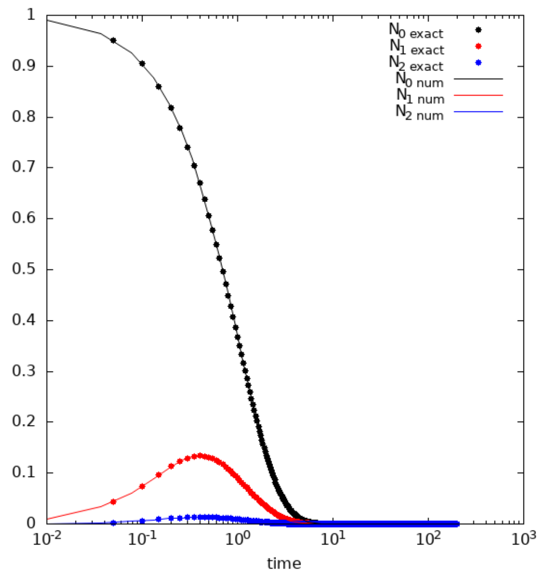

Figure 1: Distribution of the number of subsequent isotopes through time

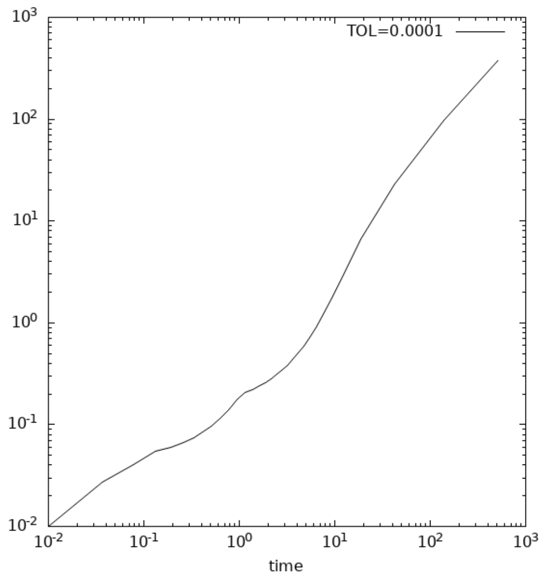

Figure 2: Change of adaptive time step Δt over time

The result correctly represents the radioactive decay of subsequent isotopes. N_0 decays exponentially over time. N_1 and N_2 increase as an effect of radioactive decay of the previous isotope in the chain, and gradually obey this law too. Characteristics of the decay speed for each isotope are also conserved - small value of λ_0 implies slower decay in comparison to λ_1 and λ_2.

It is clearly visible that the time step kept low values for significant changes in $\vec{N}(t)$ and obtained higher values in late simulation phase, when the state of the $\vec{N}(t)$ was rather stable. This proves the appropriate implementation of the adaptive time step mechanism.

3.2 Analysis of radioactive decay behavior for different tolerances

The next step was to simulate the radioactive decay with initial conditions $\lambda_0 = 100$, $\lambda_1 = 1$, $\lambda_2 = 0.01$ and for two different tolerances $TOL = 10^{-6}$ and $TOL = 10^{-3}$. The distributions of the number of isotopes for both tolerances are presented in Figures 3 and 4 with their respective adaptive time steps in Figures 5 and 6.

Both visualizations of radioactive decay seem to preserve the same phenomena as the subsequent isotope appears in the nearby of maximal value of the previous isotope state. What is striking is that for lower tolerance, the plot seems to contain negative values. By looking at the raw data:

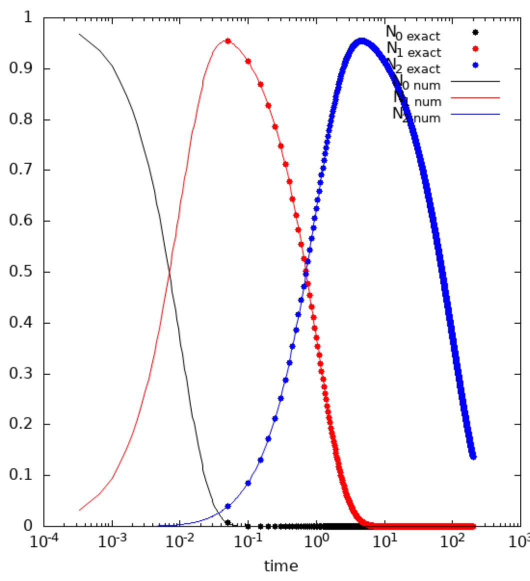

Figure 3: Distribution of the number of subsequent isotopes through time for $TOL = 10^{-6}$

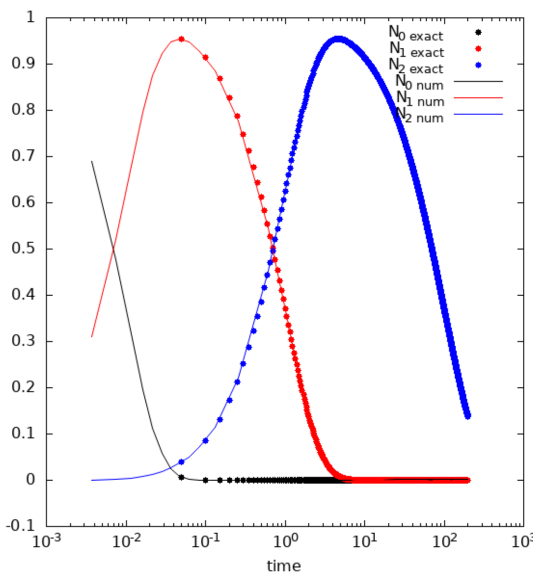

Figure 4: Distribution of the number of subsequent isotopes through time for $TOL = 10^{-3}$

```
time       N_0          N_1          N_2
7.61986    0.000274393  0.00026956   0.935536
11.1205    0.000439937 -0.000391432  0.903835
```

one may conclude that the algorithm indeed produces the negative values for N_1 and increases the value of N_0 (moves the chain in the opposite direction) which is an unexpected and nonphysical behavior.

The anomaly is also visible in the evolution of the adaptive time step Δt. For smaller tolerance value, the timestep increased slowly in the nearby of maximal number of any of isotopes and accelerated during its decay until another maximum of subsequent isotope is approached, creating the step-like shape. In contrary, the higher tolerance resulted in stabilization of timestep, which didn't change following the decay.

4 Conclusions

The laboratory proved that the trapezoidal method with Newton subiteration and adaptive time step is an excellent simulation method for radioactive decay of radiative family, closely presenting this physical phenomenon. It provided the universality of the method for various characteristics of radioactive isotopes such as their initial amount N and their characteristic constant λ. Moreover, it showed the importance of the choice of proper tolerance in adaptive time step generation, as it is crucial for simulation's quality and correctness.

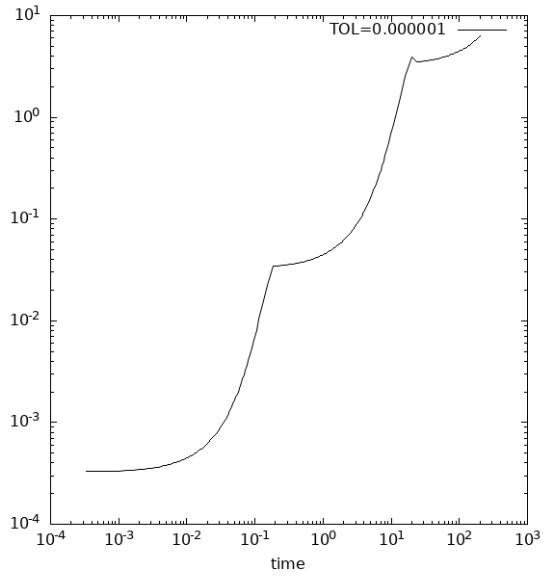

Figure 5: Change of adaptive time step Δt over time for $TOL = 10^{-6}$

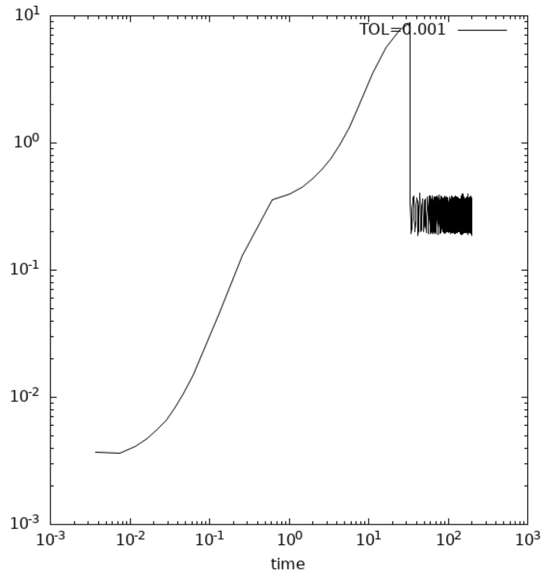

Figure 6: Change of adaptive time step Δt over time for $TOL = 10^{-3}$