PHYS 512 Assignment 2

Liam Fitzpatrick

September 2022

1 Question 1

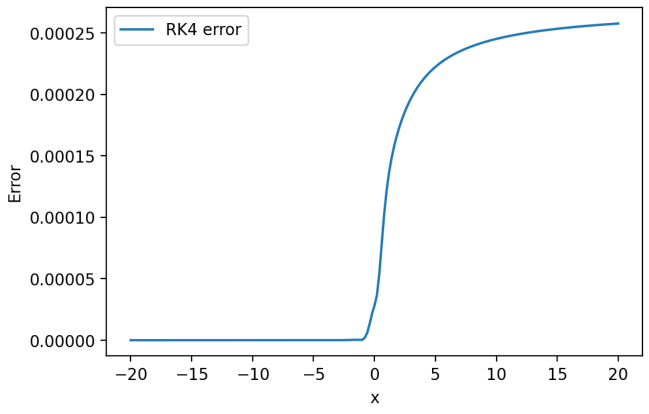
Using the same RK4 method outlined in the lecture notes, the function `rk4_step(fun,x,y,h)` was coded and the number of function calls was counted (for later):

```
def rk4_step(fun,x,y,h):              #Start count of fun calls:
    k1=h*fun(x,y)                     #+1
    k2=h*fun(x+h/2,y+k1/2)            #+1
    k3=h*fun(x+h/2,y+k2/2)            #+1
    k4=h*fun(x+h,y+k3)                #+1
    return y+(k1+2*k2+2*k3+k4)/6      #Total 4 fun calls
```

This stepper function was then iterated over each interval (x from -20 to 20 with 200 intervals):

```
nsteps=200
npt=nsteps+1
x=np.linspace(-20,20,npt)
y=np.zeros(npt)
y[0]=1  #Initial condition
for i in range(nsteps):    #Iterate stepper over each interval
    h=x[i+1]-x[i]
    y[i+1]=rk4_step(fun,x[i],y[i],h)
```

Here is the plot of the resulting error from the true solution (of order 10^{-4} for h=0.1):



In order to improve this method, we will compare 1 step of length h to 2 steps of length h/2. Since RK4 uses a Taylor expansion up to the fourth order, the error terms will be $O(h^5)$. Using $RK4_h(x)$ to define the RK4 computed step with step size h:

$$y(x+h) = RK4_h(x) + O(h^5) = RK4_{h/2}(x) + 2O((h/2)^5) = RK4_{h/2}(x) + \frac{1}{16}O(h^5)$$

Where $O((h/2)^5)$ is multiplied by 2 in the h/2 RK4 stepper since 2 steps are needed for every 1 step of h.

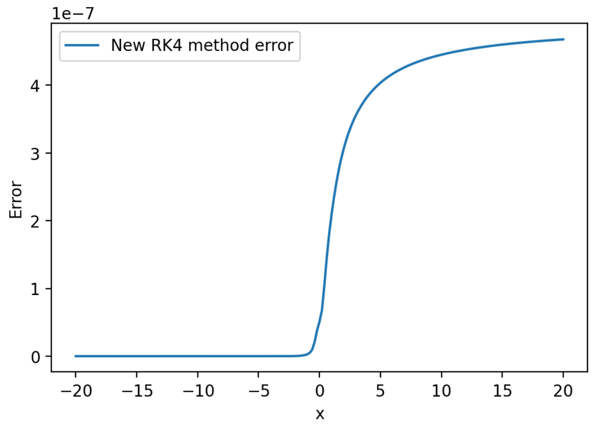
$$\Rightarrow 16\, RK4_{h/2}(x) - RK4_h(x) = 15y(x+h) + O(h^6)$$

$$\Rightarrow y(x+h) = \frac{16\,RK4_{h/2}(x) - RK4_h(x)}{15} + O(h^6)$$

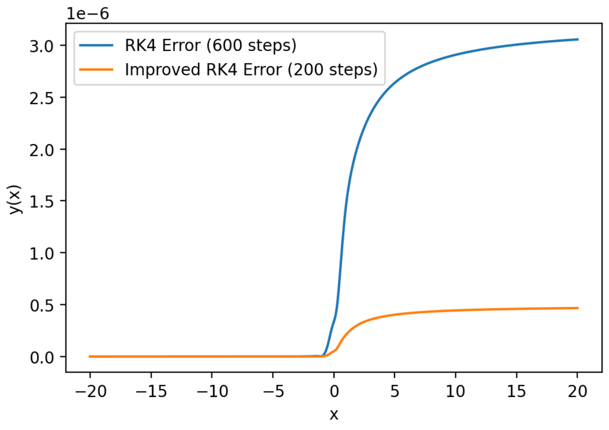
By combining step size h and h/2 in this ratio, we can obtain an error of $O(h^6)$. Writing the function `rk4_stepd(fun,x,y,h)`:

```
def rk4_stepd(fun,x,y,h):              #Start count of fun calls:
    y1=rk4_step(fun,x,y,h)             #+4
    y2=rk4_step(fun,x,y,h/2)           #+4
    y2=rk4_step(fun,x+h/2,y2,h/2)      #+4
    return (16*y2-y1)/15               #Total 12 fun calls
                                       # => use 1/3 the intervals for same call count
```

Iterating over the intervals and plotting results in:



As seen in the above code blocks, `rk4_step` calls `fun` 4 times for each step, `rk4_stepd` calls `fun` 12 times for each step. Therefore for the same number of function calls, the interval will be split into 1/3 the amount of sub-intervals for evaluating with `rk4_stepd` when compared to `rk4_step`. Using 600 sub-intervals for `rk4_step` and 200 for `rk4_stepd`, their errors were plotted:



Even when using 1/3 the amount of intervals to evaluate the stepper on, the improved RK4 method is still greater than 6 times more accurate than the original method.

2 Question 2

For this problem, a system of ODE's modelling the decay chain of Uranium-238 was solved using `scipy.integrate.solve_ivp(fun,t_span,y0)`. The system of ODE's in question is an example of the Bateman Equation: (`https://en.wikipedia.org/wiki/Bateman_equation`)

$$\frac{dN_1(t)}{dt} = -\lambda_1 N_1(t)$$

$$\frac{dN_i(t)}{dt} = \lambda_{i-1} N_{i-1}(t) - \lambda_1 N_i(t)$$

$$\frac{dN_k(t)}{dt} = \lambda_{k-1} N_{k-1}(t)$$

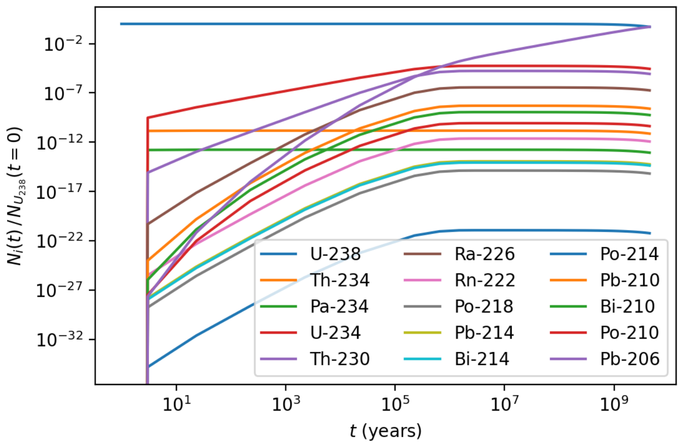
Where there are k isotopes and where $N_i(t)$ is the concentration of the i^{th} isotope with a decay rate λ_i. In this case, `y0` is the initial conditions vector and `fun` is the vector of functions on the RHS of each equation in the system. :

```python
decay_rates=np.log(2)/half_lives
n=len(decay_rates)+1

def fun(t,y):
    dydt=np.zeros(n)
    dydt[0]=-decay_rates[0]*y[0]  #First isotope is only decaying
    for i in range(1,n-1):
        #Middle isotopes decay and are produced by previous isotopes
        dydt[i]=-decay_rates[i]*y[i]+decay_rates[i-1]*y[i-1]
    dydt[n-1]=decay_rates[n-2]*y[n-2]  #Last isotope is only being produced
    return dydt
```

Where `decay_rates` is defined using a list `half_lives` which are the half lives on the lecture slides, not including Pb-206 which is stable.

Here is a loglog plot of all the isotopes' quantities divided by the initial amount of U-238, over the time span of 1 U-238 half-life:



Next is the plot for the ratio of of Pb-206 to U-238. When the half lives of all intermediate isotopes are approximated to 0 in relation to the half life of U-238, the system of differential equations becomes:

$$\frac{dN_1(t)}{dt} = -\lambda_1 N_1(t)$$

$$\frac{dN_2(t)}{dt} = \lambda_1 N_1(t)$$

Which is easy to solve analytically:

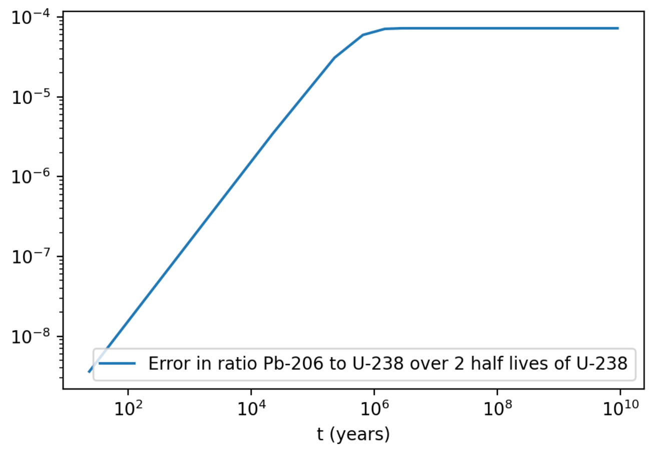
$$\frac{N_1(t)}{N_1(t_0)} = e^{-\lambda_1 t}$$

$$N_1(t)) + N_2(t) = N_1(t_0) \Rightarrow N_2(t) = N_1(t_0)(1 - e^{-\lambda_1 t})$$

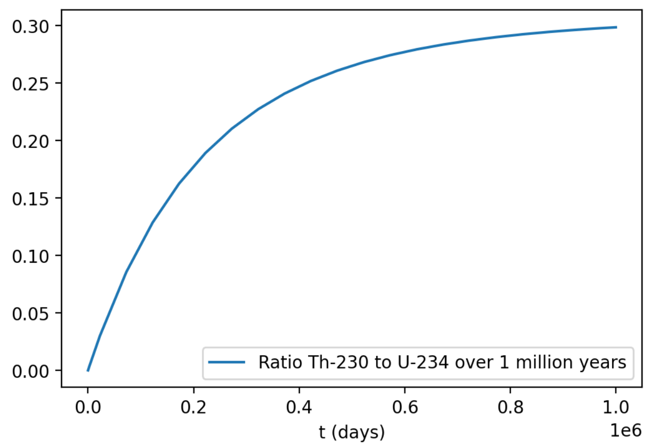
Therefore the ratio of Pb-206 to U-238 is:

$$\frac{N_2(t)}{N_1(t)} = \frac{1 - e^{-\lambda_1 t}}{e^{-\lambda_1 t}} = e^{\lambda_1 t} - 1$$

Comparing this to the numerically computed solution, the error is less than 10^{-4}:



As for the ratio of Th-230 to U-234:


Ratio Th-230 to U-234 over 1 million years
1e6
t (days)