Reading Chapter 1 2/16/25

1.1 Radioactive Decay

-A typical example of nuclear isotype ^{235}U(uranium nucleus that contains 143 neutrons and 92 protons).

-Process of radioactive decay is random in the following sense.

-It is useful to imagine that we have a sample containing a large number of ^{235}U nuclei.

-If $N_U(t)$ is the number of U nuclei that are present in a sample at time t.

$$\frac{dN_U}{dt} = -\frac{N_U}{\tau},$$

where τ is the "time constant" for decay. You can then show by direct sub:

$$N_U = N_U(0)e^{-t/\tau}.$$

1.2 A Numerical Approach

-We can now consider a simple method for solving this problem.

- Our goal is to obtain N_U as a function of t.

- Given the value of N_U at a particular value of t, we want to estimate the value at later times (initial value problem

- We can use Taylor expansion for N_U:

$$N_U(\Delta t) = N_U(0) + \frac{dN_U}{dt}\Delta t + \frac{1}{2}\frac{d^2N_U}{dt^2}(\Delta t)^2 + \dots$$

where $N_U(0)$ is the value of our func. at time $t = 0$, $N_U(\Delta t)$ is its value at $t = \Delta t$, and derivatives are evaluated at $t = 0$.

- While programming, like handwriting, is a highly individual process, there are certain recommended practices.

- It is important that we be able to understand programs written by others, as well as by ourselves.

- The first thing you should do is think. Construct an outline of how the problem is to be solved.

- <u>Example 1.1</u>

- Declare necessary variables + arrays
- Initialize variables

-Do actual calculation.

-Store results.

-Note that this is the main program.

-Consider how this looks in Fortran.

```
C Simulation of radioactive decay.
C Program to accompany "Computational Phys"
  by N. Giordano & H. Nakanishi
    Program decay
C   declare arrays
    double precision n_uranium(100), t(100)
C use subroutines to do the work
   call initialize(n_uranium, t, tau, dt, n)
   call calculate(n_uranium, t, tau, dt, n)
   call store(n_uranium, t, n)
   stop
   end
```

Example 1.2 Pseudocode for subroutine

-Prompt for and assign $N_U(0)$, τ, and Δt.

-Set initial value of time, $t(0)$.

-Set numbers of time steps for calc.

-Fortran version of subroutine.

```
      Subroutine initialize(nuclei,t,tc,dt,n)
      Initialize variables
      double precision nuclei(i), t(i)
      Print *, initial number of nuclei→'
c     read(5,*) nuclei(i)
      Print *,'time constant →'
      read(5,*) tc
      Print *,'time step →'
      read(5,*) dt
      Print *,'total time →'
      read(5,*) time
      t(1) = 0
      n=min(int(time/dt),100)
      return
      end
```

Example 1.3

- For each time step i (beginning with i=1), calculate N_U and t at step i+1:
  - $N_U(t_{i+1}) = N_U(t_i) - (N_U(t_i)/\tau)dt$ (Euler method)
  - $t_{i+1} = t_i + \Delta t$.
  - repeat for n-1 time steps.

Subroutine calculate in Fortran:

```
      Subroutine calculate(n_uranium,t,tau,dt,n)
c     Now use Euler method
c     Variable dimensioning is used for
arrays n_uranium() and t() do i=1,n-1
        n_uranium(i+1)=n_uranium(i)-(n_uranium(i)/tau)d
        t(i+1)=t(i) + dt
```

```
end do
return
 end
```

- Final store writes the result to a file.

Subroutine store in Fortran

```
 Subroutine store (n_uranium,t,n)
 double Precision n_uranium(n), t(n)
 Open(1,file='decay.dat')
 do i=1,n
        write(1,20), t(i), n_uranium(i)
end do
close(1)
format(1x,1P,2(e12.5,2x))
return
end
```

Graph:

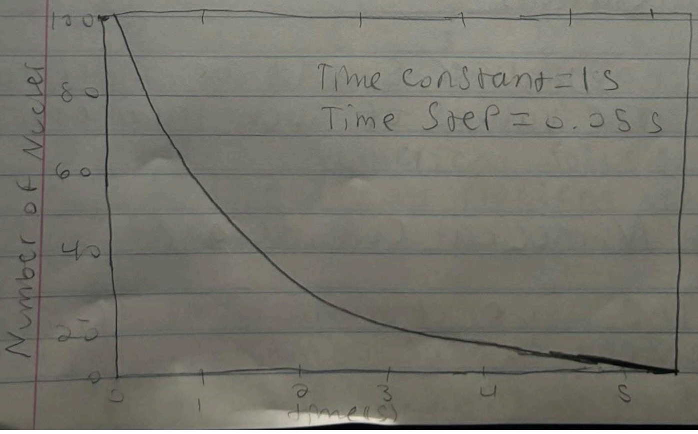

Figure 1.1; Circles indicate the numerical solution to the radioactive decay Problem

1.4 Testing Your Program.

- We should not really consider it to be a working function until we are convinced that its output is correct.

- Checking a Program is not always a trivial task, but there are some general guidelines.

- After a Program has been debugged so that it can be run without any complaints about the syntax

- Always check your Program: gives the same answer for different "step sizes" Our decay program involved a timestep variable, dt, and most other numerical calculations.

- Final answer should be independent of the values of such Parameters.

1.5 Numerical Considerations

- The issue of numerical errors is central to the computational solution of any Problem.

- Questions such as how to design or choose the best algorithm for a particular problem are central topics in many computer science and applied math courses.

- With our radioactive decay program errors were introduced by the approx used to estimate the solution of the diff eq.

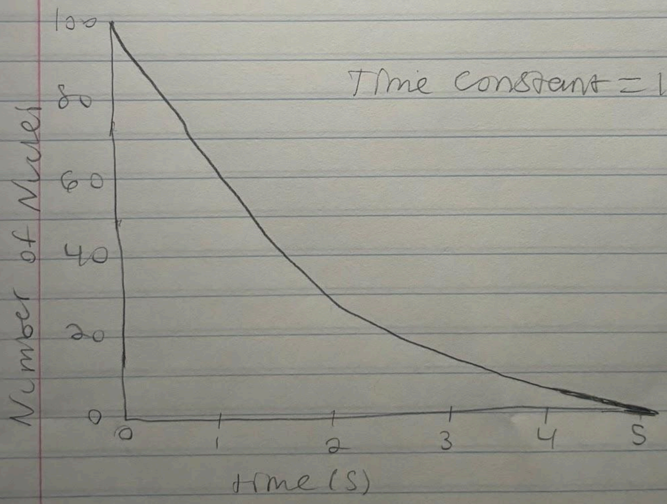

Figure 1.2: Numerical solution of the radioactive decay problem using the Euler method.