SPLINLAB, FRIB, MICHIGAN STATE UNIVERSITY

Radiation Calculator Manual

Jake Huneau

29/7/16

IMPORTANCE AND APPLICATION

The amount of radiation may be desired to know the health dangers of handling a certain isotope for a certain amount of time, or the amount of radiation that is given off my an isotope for a certain amount of time. This radiation calculator uses an isotope that is inputed by the user and computes the decay chain of that isotope and the number of atoms different time steps for each isotope in the decay chain of that initial isotope at teach time step. This is then used to calculate the amount of energy of each type of radioactive decay associated with that decay, along with plots of the activity and number of atoms over time for each isotope in the calculated decay chain.

DESCRIPTION OF PROGRAM CODE AND NECESSARY FILES

All of the files can be found at the following github link:
https://github.com/TheStrangeQuark/Radiation_Calculator

The program is written using Python version 2.7, but can be ran with any more recent version of version. To run the program, there are 2 python files and 2 text files needed:

- `decaydata.txt`: Text file which contains information on each isotope from the NNDC.[1]
- `exposure_rate_constants.txt`: file of exposure rate constants and lead shielding values for radionuclides.
- `isotopic_dicts.py`: Python file containing dictionaries of information about isotopes used in the program.

- `radiation_calculator.py`: Main program file, which calculates decay chain for user inputed isotope, the amount of each isotope, and the amount of energy for each form of radiation involved in the decay chain.

Each of these files will now be described in more detail.

DECAYDATA.TXT

This text file was provided by the NNDC as a text file of information about each isotope available to them at the time. The format of the text is the following: each unindented line has the following format:
`Z;A;Liso;E(keV),T12 (s);DT12 (s);spin;parity;ndk`
where Z is the atomic number, A is the nucleon number, Liso is the isomer (0 is ground level, 1 is the first isomer, etc.), E is the energy level of the isomer in kiloelectron volts, T12 is the half-life of the isotope in seconds, DT12 is the uncertainty in the half-life in seconds, spin is the quantum spin number, parity is spin parity, and ndk is the number of decay modes. If ndk is at least 1 then there will a number of indented lines under this line which is equal to ndk. Each of these lines described a decay mode. The decay modes have the following format:
`dm;Liso(daughter);BR;DBR;Q (keV);DQ (keV)`
where dm is the decay mode, Liso(daughter) is the isomer of the daughter isotope, BR is the branching ratio as a decimal, DBR is the uncertainty of the branching ratio, Q is the Q-value of the decay mode in kiloelecton volts. DQ is the uncertainty in the Q-value in kiloelectron volts.

For example, the following lines from the text file would describe the first isomer Ra-213, which decays via alpha decay 1% of the time and goes through an isomeric transformation to the ground state of Ra-213 before decaying further 99% of the time:

```
88;213;1;1.77000E3;0.0021;1.0E-4;-77.777;0.0;2
     alpha;0;1.00000E-2;2.00000E-3;8.63000E3;8.94427E0
     IT;0;9.90000E-1;2.00000E-3;1.77000E3;8.00000E0
```

EXPOSURE_RATE_CONSTANTS.TXT

This text file was pulled from a paper written by David S. Smith and Michael G. Stabin titled *Exposure Rate Constants and Lead Shielding Values For Over 1,100 Radionuclides.*[2] In this paper, different information is gathered for radionuclides. The format of this text file is the following:
First is the name of the nuclide; then the next three values are the exposure rate constant, first in unit (C m^2 / kg MBq s) where this value is multiplied by the second value which in the form 10jx, which represents 10^x. The second value is the exposure rate constant in unit (R cm^2 / mCi h); then the f-factor in unit (cGy/R); then the last 5 values are the the following characteristics of lead attenuation thickness in unit (mm Pb): HVL, QVL, TVL, CVL, MVL.

For example, the following line is for Ra-225:

```
Ra-225 8.04 10j14 0.415 0.924 0.0119 0.0347 0.0904 0.237 0.384
```

This would be Ra-225 with an exposure rate constant of $8.04x10^{-14}$ C m^2 / kg MBq s or 0.924 R cm^2 / mCi h, and an f-factor of 0.0119 cGy/R.

An f-factor is for the conversion between exposure rate in air to dose rate in tissue.

`ISOTOPIC_DICTS.PY`

This python file contains dictionaries that are used in `radiation_calculator.py`. The following dictionaries are included:

- dictionary of each atomic number to the symbol (ex: 88: Ra)
- exposure rates for each isotope
- atomic mass of each isotope

`RADIATION_CALCULATOR.PY`

This python file is the main program used for doing calculations. It imports `scipy, numpy, matplotlib`. It is organized into the following sections of code:

- `Conversions, Constants, Normalizations`: Contains constants and functions that define different conversions and normalizations.
- `Collect important data and user inputs`: Contains function for organizing data from decaydata.txt into workable form and functions for collecting user input either from user manually entering mass addition data or extracting from a list of mass addition data.
- `printing, saving, displaying`: Contains functions for printing and displaying important information and saving important information in text file.
- `Analyze Isotope`: Contains functions for analyzing an isotope to check for different properties.
- `Build decay chains`: Contains functions and Class for building the decay chain.
- `Decay Equations`: Contains functions for building system of equations to be solved by ODE for each isotope in decay chain.
- `process decay`: Contains functions for processing the decay chain into more usable formats.
- `Calculate solutions from equations`: contains function for solving system of equations. ODE used is described bellow.
- `Analyze solutions`: Analyzes solutions for different desired information; such as upper limit of energy for each type of radiation.

The `main` does the following:

1. Collects all isotope data.

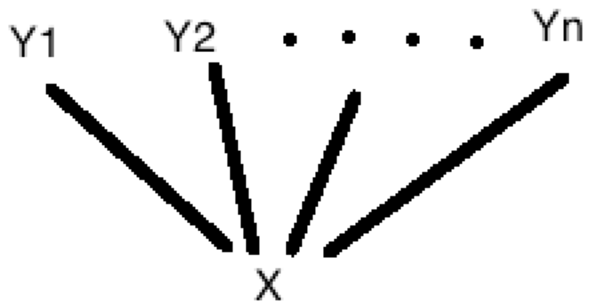
2. Collects information about initial isotope from user.
3. Creates decay chain.
4. Collect sample rate from user.
5. Builds equations.
6. Creates dictionary of initial activity for each isotope in decay order. This will be 0 for each isotope except initial isotope.
7. Solutions are found for initial chain. Then solves again for each additional mass that is added if any additional masses are added.
8. Solutions are converted to activity from total number of atoms.
9. Total number of atoms decayed for each isotope is calculated.
10. Total amount of energy is calculated for each type of radiation that is involved in decay chain.
11. Amount of energy for each radiation is printed.
12. solutions of the number of atoms for each isotope at each time step is saved in a text file.
13. plot is created of number of atoms and activity for each isotope in decay chain.

Description of how Decay Chain Equations are Made

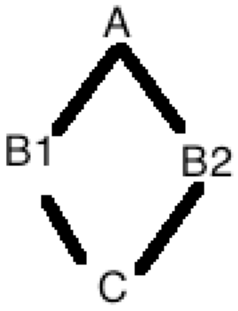
The decay chains are calculated from a system of equations build from the following equation, which would describe how a single equation is built:

$$X_f = \left(\sum_{Y_j=Y_1}^{Y_n} P_{Y_j X} Y_{j,i} / X_{j,\lambda} \right) - X_i / X_\lambda \tag{1}$$

Where X_f is the final amount of species X, Y_j is each parent species that decays into X, $P_{Y_j X}$ is the probability that Y_j will decay into X, X_i is the initial amount of X, and X_λ is the half-life of X. This equation would demonstrate the following decay:



This equation is used for any decay still. For example, the following decay tree will be described:



$$
\begin{aligned}
A_f &= -A_i / A_\lambda \\
B_{1,f} &= P_{AB_1} A_i / A_\lambda - B_{1,i} / B_{1,\lambda} \\
B_{2,f} &= P_{AB_2} A_i / A_\lambda - B_{2,i} / B_{2,\lambda} \\
C_f &= P_{B_1C} B_{1,i} / B_{1,\lambda} + P_{B_2C} B_{2,i} / B_{2,\lambda} - C_i / C_\lambda
\end{aligned}
$$

Description of ODE

The function used to solve the ODE is `scipy.integrate.odeint()` which uses `lsoda` from the FORTRAN library `odepack`. A desciption of `scipy.integrate.odeint()` is found at the following link:
scipy.integrate.odeint() Description Link
A description of `lsoda` from the FORTRAN library `odepack` is found at the following link:
lsoda Description Link
A stiff equation is an equation in which numerical solutions are numerically unstable, meaning neighboring solutions are orders of magnitude in difference, and may indicate that smaller step sizes are needed. `lsoda` initially begins with nonstiff (Adams) methods and dynamically

monitors solutions to decide if the method should remain being nonstiff or switch to stiff (BDF) methods if the solutions are stiff.

A description of Adams method, the method used for nonstiff solutions, can be found at the following:
Adams Description Link

A description of BDF method, the method used for stiff solutions, can be found at the following link:
BDF Description Link

INSTALLATION

Python is needed to run this radiation calculation. Using the Anaconda distribution of Python with automatically install all the required packages and is recommended: https://www.continuum.io/downloads. If the user decides against using Anaconda, then the following packages will be needed to run the radiation calculator: `numpy, scipy, matplotlib`.

The file named `radiation_calculator.py` can be ran on Windows by simply double-clicking on the file, it can also be ran from the command line or terminal by first changing to the directory containing the file and then typing in the command prompt or terminal:

```
python radiation_calculator.py
```

If this does not work then try the following:

```
py radiation_calculator.py
```

DESCRIPTION OF INPUTS AND OUTPUTS

INPUTS

There are two possible types of input for this program: Manual or File. The type of input will be initially asked by the program. If the user enters "manual", then the program will ask for the isotope by asking for z (atomic number), a (nucleon number), and isomer energy level (0 is ground state) which are used to determine the initial isotope in the chain. The initial activity of this isotope is then asked in mCi. Then the time scale that will be used in the program is asked. This will be the time scale that sample frequency and other functions will use. This should be determined from how long the user wants to simulate the decay for. (Ex. if the initial isotope has a half-life of 5 seconds, then the user may want to use 's' as the time scale). The Max time (in that time scale) is then asked. These inputs are what are all used to create the initial decay chain and solutions. Next, the program will ask if there are added masses. If the user wants to add mass of the initial isotope at a later time, they will enter "yes". This will then enter a loop of entering more masses at later times. This loop will ask what time (in the user inputed time scale) that the mass is added and the activity of the mass that is added in mCi. This loops will continue with these 2 questions until the user enters "n" or anything that isn't a number, which will break out of the loop.

If the user instead enters "file", then the program will ask for a file name. this filename must be in the following format: z.a.isomer. .txt where z is the atomic number, a is the nucleon number and isomer is the isomer energy level (0 is ground state), and can be anything else. The program uses this to find the initial isotope. The text file must be formatted in the following way: there are two columns in the text file separated by a tab. The first column is the date and possibly the time at which as a mass is added. The second column is the activity in mCi of the added mass coordinating to the date in that row. The date and possibly time is in the following format: year.month.date.hour.minute.second, where only the year.month.date are required and anything after is optional. After the file is inputed, the program will ask for the time scale, sample frequency, and max time for the simulation to be ran. The max time must be later than the last entry in the text file.

The user will then be asked which type of plots are desired. The two options are "log" or "linear", where log is a plot with logarithmic x-axis and y-axis.

Outputs

There are three outputs from the program. The first is the amount of energy of each associated radiation with the program. This will be printed in the terminal, or wherever the program is ran. The second is a set of plots: one is the number of atoms for each isotope in the decay chain and the second is the activity of each isotope in the decay chain. The final output of the program is a file, which is saved to the folder which contains the program. The name of this file will be z.a.isomer.out.txt and will contain the a row for each time step. The first item in the row is the time stamp, and each item after is the number of atoms of one of the isotopes in the decay chain. The corresponding isotope can be found in the header, where the isotope order is displayed in the format (z,a,isomer).

Example

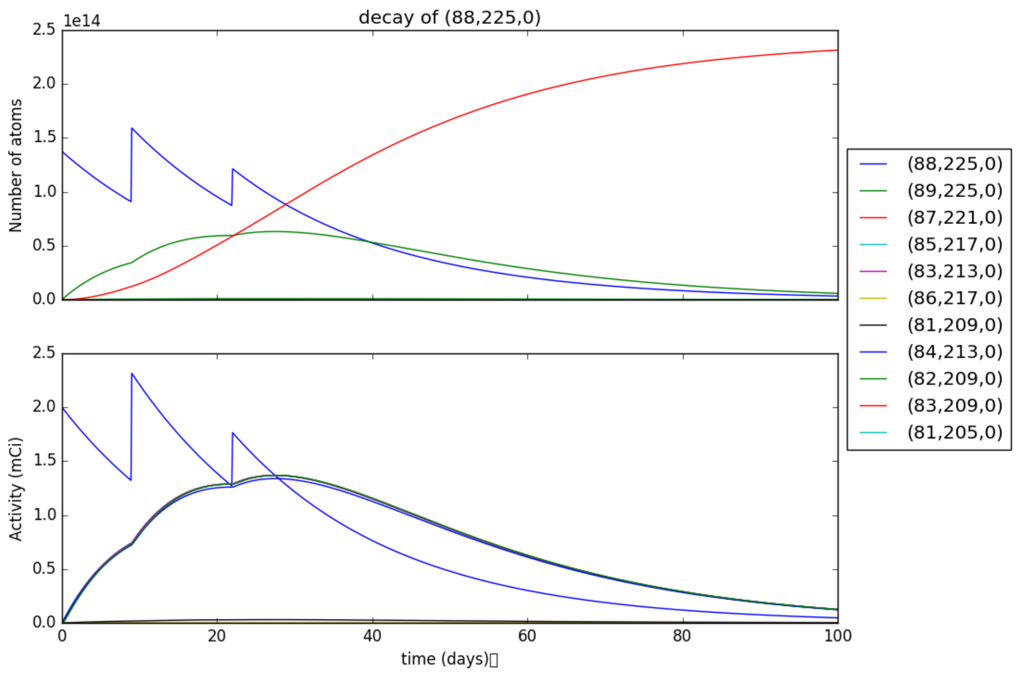
The following file input can be used with the following text file named `88.225.0.txt`

```
2016.7.11        2
2016.7.20        1
2016.8.2       0.5
```

The user then can enter "days" for the time scale, "100" for the Max Time, "linear" for the plot type, and "10" for sample frequency. The following should be printed on the terminal:

```
Radiation Energies
beta- :  51.3692592385 J
alpha :  516.51259369 J
```

The following plot is produced:



and the following text file is created, named `88.224.0.out.txt`

```
# timeyr 88,224,0 86,220,0 84,216,0 82,212,0 83,212,0 81,208,0 84,212,0 82,208,0
0.000000000000000000e+00 1.687994747457265039e+13 0.000000000000000000e+00
0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
1.000000000000000056e-01 1.468780876979152148e+13 2.582934181882912636e+09
 6.736072447955859825e+06 1.285811877101314697e+12 1.168434250635101624e+11
 2.112660219652587414e+09 6.160231084615872099e+00 7.847810721361469727e+11
2.000000000000000111e-01 0.000000000000000000e+00 0.000000000000000000e+00
 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
 0.000000000000000000e+00 2.111307223283 18649e-228 2.186219992957051630e-314
3.000000000000000444e-01 2.186216127387438568e-314 -1.260957053217541341e+44
2.186219755805541626e-314 2.186215949523806065e-314 0.000000000000000000e+00
0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
4.000000000000000222e-01 0.000000000000000000e+00 0.000000000000000000e+00
 3.575812437392816887e-245 2.186219850666145628e-314 2.186216020669259066e-314
 -1.045697856021544139e-244 2.186219945526749629e-314 2.186216091814712067e-314
5.000000000000000000e-01 0.000000000000000000e+00 0.000000000000000000e+00
 2.272701970869734103e-322 2.137758690576651115e-314 0.000000000000000000e+00
0.000000000000000000e+00 2.173888841701484794e-322 0.000000000000000000e+00
6.000000000000000888e-01 0.000000000000000000e+00 0.000000000000000000e+00
2.075075712533235486e-322 2.681561747002458153e+154 0.000000000000000000e+00
0.000000000000000000e+00 1.976262583364986177e-322 1.253070585211905243e-316
7.000000000000000666e-01 2.137758904013010119e-314 4.940656458412465442e-324
4.940656458412465442e-324 0.000000000000000000e+00 0.000000000000000000e+00
```

```
 0.000000000000000000e+00 0.000000000000000000e+00 9.734698130969006051e-309
8.000000000000000444e-01 6.953244028976664988e-310 5.387879942938418196e-315
6.953266524472469013e-310 6.953244305989391298e-310 2.137758753817053783e-314
1.390673299325810017e-308 6.953244028976664988e-310 5.387879942938418196e-315
9.000000000000000222e-01 6.953266520161845067e-310 6.953244305990972308e-310
2.137758809152406117e-314 4.227640331163685790e-307 6.953242933380957052e-310
nan nan 2.123805541568413301e-314
1.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
 0.000000000000000000e+00 0.000000000000000000e+00 2.529616106707182306e-321
2.529616106707182306e-321 4.940656458412465442e-324 0.000000000000000000e+00
```

In this text file, the rows are too long to fit on one line, and if the text size were made smaller, then each data point would fit on one line. The different time steps are still in separated rows that are separated by a return.

References

[1] National Nuclear Data Center. Isotope information. retrieved via email, February 2016.

[2] David S. Smith and Michael G. Stabin. Exposure rate constants and lead shielding values for over 1,100 radionuclides. *Health Physics*, 102(3):271–91, February 2012.