EASY – Exact Nuclear Statistical Yields
Directions and Explanations

The programs from this distribution can be used to calculate the yields of individual levels of all nuclides assuming thermal production according to a canonical ensemble followed by sequential decay. The thermal production is calculated via Mekjian's method, which allows one to statistically sum over all partitions of a system of size $Z_{\rm sys}$ and $N_{\rm sys}$ into all possible divisions into fragments strictly conserving the overall charge and nucleon number. Within the canonical ensemble, all levels of the individual nuclei are accounted for through the partition function. The initial fragmentation provides the probability $P_{z,a,i}$, the probability of producing a fragment of charge z, mass a and in the level i. The sequential decay portion of the calculation reassigns the weight $P_{z,a,i}$ into all available levels in all the daughter nuclides which are accessible given the eight decay modes which are considered.

The principal advantages to this approach are:

- No Monte Carlo is invoked which allows one to consider rare fragments.
- Charge/mass conservation is strictly enforced.
- The modeling of sequential decay accounts for non-thermal forms for the relative populations of levels.

The principal disadvantages to this approach are:

- Energy conservation is not strictly enforced. This is possible, but computationally very expensive, see S. Pratt and S. Das Gupta, Phys. Rev. C**62**, 044603 (2000). This is usually not an issue for single-particle yields.
- Explicit fragment-fragment interactions are ignored. Only mean-field or Wigner-Seitz corrections can be readily included in this approach.

Initial Statistical Fragmentation

Mekjian's method is built on knowing the partition functions of the individual nuclei. In this version, the partition functions are of a simple form,

$$Z_{z,a}(T) = \frac{f_V A_{\rm sys}}{\rho}\frac{(maT)^{3/2}}{(2\pi\hbar^2c^2)^{3/2}}\sum_i D_{z,a,i}\exp\left(-\frac{E_{z,a,i}+\Delta_{\rm coul}(z,a)}{T}\right), \tag{1}$$

Here ρ is the breakup density of one sixth $\rho_0 = 0.15$ fm^{-3}, and f_V is the fraction of the volume which is available for the center-of-mass motion, $f_V = 1.0 - 2\rho/\rho_0$. The energy and degeneracies of the levels i are $E_{z,a,i}$ and $D_{z,a,i}$, while $\Delta_{\rm coul}$ is the in-medium correction (Wigner-Seitz) to the Coulomb interaction,

$$\Delta_{\rm coul}(z,a) = -(0.7\ {\rm MeV})\frac{z^2}{a^{1/3}}\cdot(\rho/\rho_0)^{1/3} \tag{2}$$

It is not horribly difficult to incorporate more sophisticated forms for the partition functions $Z_{z,a}$. One must merely adjust the routines in *nucl_info.cc* that read the energy levels (levelinit), calculates the partition functions of individual nuclei (omegafinder), calculates the probability of occupying individual states (levelfiller), and calculates the binding energy (frldm). The binding energies are calculated with the bulk portions of the Finite-Range Liquid-Drop Model FRLDM. The degeneracies are discussed below.

Generating Levels and Degeneracies of Nuclides

For lighter nuclei, $A < A_{\mathrm{MAKEBELIEVE}}$ (defined in "easy.h"), nuclear levels are generated by reading binding energies from files in the directory *easy/real_levels.*

Levels and degeneracies of heavier nuclei are modeled by assuming that both neutrons and protons occupy uniformly-spaced single-particle levels. The spacing of the single-particle levels is given by the expression

$$\Delta E = \pi^2\alpha/a, \tag{3}$$

where α is the level density parameter LEVELPAR set to 10 MeV in *easy.h.* The degeneracy of the ground state is always unity and the degeneracy of the excited states are determined by counting the number of ways one can excite N_p protons and N_n neutrons into excited states, and summing over all N_n and N_p. This is done assuming the single-particle states are infinitely deep. See S. Pratt, Phys. Rev. Lett. **84**, 4255 (2000). Thus, the degeneracies are calculated for discrete steps in excitation energy, ΔE, and are the same for all nuclei when scaled with the step-size ΔE which varies with the fragment mass a.

For increasingly heavy nuclei, the step-size becomes smaller. When ΔE becomes smaller than a parameter ERESOLUTION (set in *easy.h*) the degeneracies are interpolated onto a mesh of granularity ERESOLUTION.

The Finite-Range Liquid-Drop Model, FRLDM, is used to generate the ground- state energies of these nuclei. But only the bulk terms are used.

Sequential Decay

All nuclear levels in all nuclides are assigned exact populations through the Mekjian method. Maintaining the exact nature, i.e. foregoing Monte Carlo, requires that the weight of each particle-unstable levels be reapportioned into the levels of the daughter nuclei through which it might decay. The terminology used throughout the program is MOTHER, DAUGHTER and BABY for the decaying nucleus and the two products. Eight modes of decay are considered, i.e. eight species of BABIES. These are proton, neutron, deuteron, dineutron, diproton, ^{3}He, tritons and αs. The BABIES are assumed to be in their ground state. More modes can easily be added by adjusting the relevant parameters defined in *easy.h.*

The relative weights for decaying into the various levels of the various daughters is determined by simple Weisskopf arguments.

$$w(- \to i) \propto D_i \mu R^2 (E_k - V_c), \tag{4}$$

where D_i is the degeneracy of the level, R is the capture radius, $R = R_{\mathrm{daughter}} + R_{\mathrm{baby}}$, E_k is the kinetic energy of the outgoing particles, V_c is the Coulomb barrier and μ is the reduced mass.

If energies are below the Coulomb barrier, a crude WKB tunneling approximation is used assuming that tunneling proceeds through an s wave.

How to run EASY

From the main directory where the makefile is stored, one compiles the program by entering

```
% make easy
```

One can then run the program by entering

```
% easy
```

The program prompts you

```
How many protons(Z) and nucleons(A) in the system?
```

At this point enter the net number of protons and the mass of the system, e.g. if you are fragmenting ^{208}Pb, enter

```
82 208
```

Next, the program will prompt for the temperature,

```
What is the TEMPERATURE? (in MeV)
```

Simple enter the temperature in MeV, e.g. for a temperature of 5 MeV,

```
5
```

Finally, the user is prompted for the highest masses which will be decayed.

```
What is the decay cutoff?
(Will only decay masses <= cutoff to save CPU)
```

One could enter the total mass of the nucleus, then all nuclei would be decayed. However, this is extremely CPU intensive for heavy nuclei. If one were only interested in Calcium isotopes, there would be no reason to decay mass-200 objects. One might then choose $a = 100$ as the cutoff. As long as the temperature is not so high that calcium nuclides would result from the decay of mass $a > 100$ objects, the choice is safe. Our experience has been that for temperatures less than 8 MeV there is no reason to decay nuclides with masses more than 20 above those you are interested in. If one wanted a cutoff of 100, one would thus simply enter

```
100
```

The program should now run without further intervention. The program may need only a few minutes or a few seconds to calculate the initial populations. One should see the following output.

```
Beginning recursive calc.s of all part.  functions
Beginning calc.  of mass dist.s
CHECK: net neutrons=126 net protons=82
```

The first two lines simply inform the user of the progress being made. The last line reports on a check that the initial populations indeed add to the correct values. The program will exit and report an error if these values are incorrect.

Finally, the decays are modeled. This may be slow and can even take several hours if one decays heavy nuclei with a large cutoff. The output at this stage should be

```
Beginning Decay Process
finished decaying a=100 nuclides
```

```
finished decaying a=99 nuclides
finished decaying a=98 nuclides
finished decaying a=97 nuclides
finished decaying a=96 nuclides
⋮
```

When the program finishes, there should be a message

```
⋮
finished decaying a=2 nuclides
finished decaying a=1 nuclides
testfinalsum=110.897 testinitialsum=110.897
```

The two sums refer to the net mass included in those nuclei with mass less than or equal to the decay cutoff. The final and initial sums should be equal. If one would have chosen the cutoff equal to the total system mass, the sums would equal the total system mass.

For the most part, if the program exits without an error message, the calculation was successful.

Adjusting other parameters

A variety of other parameters can be found in *easy.h*. The parameters one might wish to adjust are:

LEVELPAR	This is the level density parameter, set to 10 MeV by default.
ERESOLUTION	For heavier nuclei this forces the degeneracy and energies of the internal states to be interpolated onto a mesh of resolution set to 1.0 MeV by default.
A_MAKEBELIEVE	Levels for nuclides with masses greater than or equal to this parameter are generated via the description described earlier.
RHO_RATIO	This is the inverse ratio of the breakup density to nuclear matter density, and is set to 6 by default.
AVAILABLE_VOL_FRACTION	Due to excluded volume effects, only this fraction is available for motion. By default it equals 1.0-2.0/RHO_RATIO.
ZBABY,NBABY	These arrays describe the charges and neutron numbers of the decay modes.
MAXLEVELS	The maximum number of internal levels, set to 1000 by default.
LOWERLIMIT	In the summary file, no nuclides with populations lower than this level are reported. It is also used to determine when to crop the levels for individual nuclei. Set by default to 10^{-20}.
WRTEPOPS	If this is set populations of individual levels will be written. By default this is undefined.

Results

Each run produces a file in the *results* directory. The filename includes information about the charge and mass of the decaying system and the temperature. For instance, If one decayed ^{208}Pb at a temperature of 5 MeV, the summary file would be named *summary_Z82_A208_T5.dat*.

If one defines WRITEPOPS in *easy.h*, the populations of individual levels will be included in files which would be in the directories *results/finalpops* and *results/initialpops*. A separate file is created for each nuclide. For example, the ^{40}Ca information would be in a file *Z82_A208_T5_z20a40.tmp*.

There also exists a short programs *sift.cc* which reads the summary file and writes only that information pertaining to a specific element or mass to a new file. The program prompts for information about which summary file to read and which element or mass to extract. The newly created file might have the name *sorted_Z82_A208_T5_z20.dat* if one had been asking for information about Calcium isotopes created from the fragmentation of ^{208}Pb at a temperature of 5 MeV.

Description of Source Files

easy.h	Contains parameter definitions, a few global variables, class definitions and function prototypes.
easy.cc	This is the main program.
easysubs.cc	A few subroutines called by the main program, including the function "recursive" which performs the Mekjian method.
decaysubs.cc	The functions that perform the decays are included here.
nucl_info.cc	In *easy.h* a class "nucl_info_class" whose class members define properties of individual nuclei. A separate object is created for each nuclide. The member functions of this class are contained in *nucl_info.cc* and are those functions whose operations refer to only the one nuclide.
sift.cc	This routine reads through the summary file to find populations of a specific element of a specific mass.

Making Plots with PHYSICA

As a bonus, I include a directory *figs* which includes a file *tempdependence_fixedz.plt* which is a PHYSICA macro. For those who do not use PHYSICA for making plots, this is of no use. The subdirectory *sorted_results* includes output files from running the *sift* program. Also present is an eps file generated from the output files which are included as an example. PHYSICA can be downloaded from TRIUMF at *http://www.triumf.ca/physica/html/homepage.html.*

Having Problems?

Feel free to contact Scott Pratt, pratt@nscl.msu.edu, (517) 333-6438.