User Manual for DMFortFactor: A Fast Fortran Code for WIMP-Nucleus Form Factors

Oliver C. Gorton, Changfeng Jiao and Calvin W. Johnson

November 16, 2021

Contents

1 Preamble **3**
1.1 Theory . . . 3
1.2 Interface . . . 3

2 Usage guide **4**
2.1 SuperQuickstart guide . . . 4
2.2 Compiling with Cmake . . . 5
2.3 Required files . . . 6
2.3.1 Control file . . . 6
2.3.2 Nuclear density matrix file (.dres) . . . 7
2.4 Command-line interface . . . 8
2.5 Compute options explained . . . 9
2.6 Event rate spectra from the Python wrapper . . . 10

3 Nuclear structure input **11**
3.1 Filling core orbitals for phenomenological interactions . . . 12

4 Python interface **13**
4.1 Example: generating a weighted sum of event rate spectra . . . 13
4.2 Example: comparing event-rate spectra for calculations with different WIMP masses . . . 14

5 Equations found in the code **16**
5.1 Differential event rate . . . 16
5.2 Differential cross section . . . 17
5.3 Transition probability / Scattering probability . . . 18

5.4 Dark matter response functions 18
5.5 Cross terms 19
5.6 Operators 20
5.7 Nuclear response functions 21
5.8 Nuclear operators and their matrix elements 21

6 Control file keywords 23

1 Preamble

We present here a fast modern Fortran code, DMFortFactor, for computing WIMP-nucleus scattering event rates using a previously studied theoretical framework arXiv:1308.6288, but with advanced algorithmic and numerical implementation, including the ability to take advantage of multi-core CPUs. Furthermore, we enhance the accessibility by including Python wrapper and example scripts and which can be used to call the Fortran code from within a Python environment.

1.1 Theory

This program is concerned principally with computing the dark matter-nucleus differential event rate as a function of the nuclear recoil energy E_R:

$$\frac{dR_D}{dE_R} = N_T \frac{\rho_\chi}{m_\chi} \int_{v_{min}}^{v_{escape}} \frac{2m_T}{4\pi v^2} \frac{1}{2j_\chi + 1} \frac{1}{2j_T + 1} \sum_{spins} |\mathcal{M}(v,q)|^2 \tilde{f}(\vec{v}) v d^3 v \tag{1}$$

This quantity has units of events/GeV and is implicitly multiplied by an effective exposure of 1 Kilogram-Day of target nuclei. This is done by taking $N_t = 1\ kilogram \cdot day/m_T$, where m_T is the mass of the target nucleus in GeV. Recoil energies E_R are given in keV.

The cross section is determined using a user-defined WIMP-nucleus interaction within a non-relativistic effective field theory (EFT) framework. The interaction is specified by 15 coupling coefficients defining an interaction:

$$\sum_{x=p,n} \sum_{i=1}^{16} c_i^x \mathcal{O}_i. \tag{2}$$

1.2 Interface

The command line interface (CLI) to the code prompts the user for the type of calculation they wish to perform from a menu of options, then the target nucleus, given by the number of protons Z and neutrons N, two input files, and finally other CLI inputs depending on the compute-option chosen. These final inputs are typically three numbers specifying the range of recoil energies or momentum for which to compute the output. After running, the code prints the results to a plain text file in tabulated format.

The two additional input files which need to be specified in the CLI are (1) a "control file", and (2) the nuclear structure input. The control file specifies the EFT interaction and any optional settings desired.

The nuclear structure inputs needed are one-body density matrices, defined below. We supply a library of density matrices for many of the common expected targets, as listed in 3. The density matrix files are written in plain ASCII, using the format output by the nuclear configuration-interaction code BIGSTICK [?, ?]. The only assumption is that the single-particle basis states are harmonic oscillator states; the user must supply the harmonic oscillator single particle basis frequency Ω, typically given in MeV as $\hbar\Omega$, or the related length parameter $b = \sqrt{\hbar/M\Omega}$, where M is the nucleon mass.

If density matrices are generated in some other single-particle basis, such as those from a Woods-Saxon potential or a Hartree-Fock calculation, that basis must be expanded into harmonic oscillator states. By using harmonic oscillator basis states one can efficiently compute the matrix elements. One can use either phenomenological or *ab initio* model spaces and interactions; as an example, we provide density matrices for ^{12}C, both from the phenomenological Cohen-Kurath shell model interaction [?], and from an interaction derived from chiral effective field theory [?].

While the DMFortFactor executable can be run by itself, we provide Python APIs for integrating the Fortran program into Python work flows; see section **??**.

2 Usage guide

`DMFortFactor` can be used interactively from the command line, where the user is guided by prompts for a small number of datafiles and parameters. Naturally, this interactive process can be expedited by piping a pre-written input file into the command line interface (CLI).

We also provide a generic application programming interface (API) for the Python language. This API essentially offers a prescribed and easy-to-use way to run `DMFortFactor` in a programmatic way. Any sufficiently experienced linux user could probably write a script to produce any possible set of inputs to our code. But our API removes the need by making it easy for anyone who can use a Python function to write their own advanced scripts allowing them to perform parameter studies and comparisons of different inputs to the theory.

2.1 SuperQuickstart guide

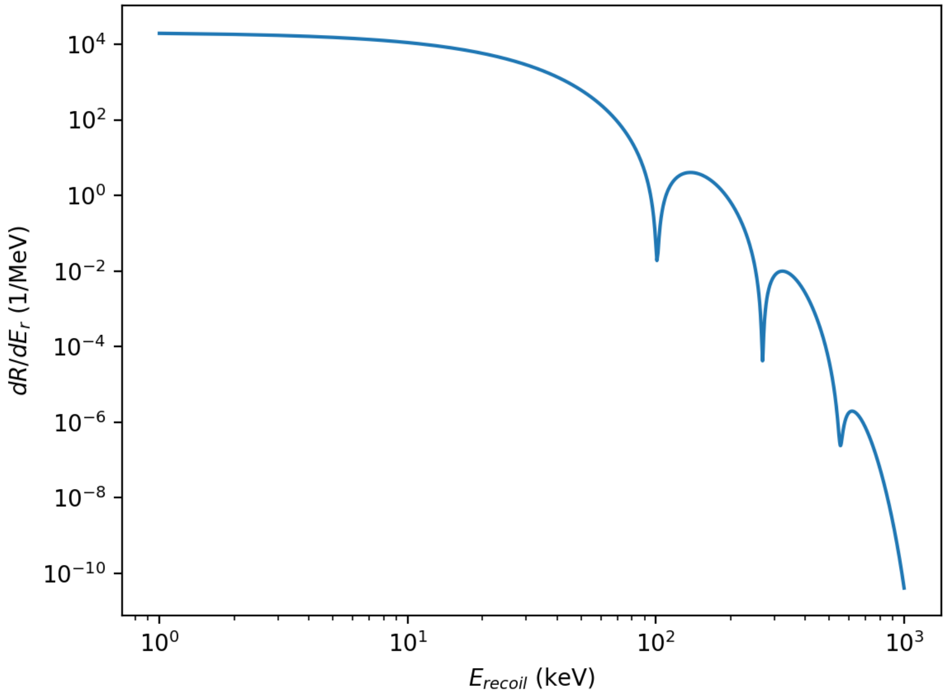
1. Navigate to the `build/` directory from wherever you have stored `darkmatter/` (e.g. `cd src/`, or `cd ~/Downloads/darkmatter/src/`)

2. Run the command: `bash ./build.sh`

3. Navigate to the directory `runs/` (e.g. `cd ../runs/`)

4. Run the command: `python3 ../examples/exampleXe.py`

This should generate the following figure:



2.2 Compiling with Cmake

There are multiple directories in the project. All of the Fortran code which needs to be compiled is found in the `src/` directory. We provide an install script `install.sh` in the `build` directory to compile the code using `Cmake`.

The easiest way to get started is simply to navigate to the `build` directory and run

```
bash build.sh
```

or, equivalently,

```
cmake ../src
cmake --build --parallel
```

This will compile `DMFortFactor` using `gfortran`. If you want to use a different compiler, you can first run `export FC=<compiler>`, setting `<compiler>` to your compiler of choice.

By default, this will compile the code with openMP. If you want a serial-only executable, you can run the `build-serial.sh script instead`, or run:

```
cmake -DSERIAL=ON ../src
cmake --build . --parallel
```

Both of these options will compile the source code and leave the executable, called `dmfortfactor` in the `build` directory. Note that if you change from a serial executable to a parallel executable (or vice versa) you should run the `clean.sh` script:

```
bash ./clean.sh
```

2.3 Required files

There are two files required for any calculation:

1. Control file (.control)
2. Nuclear density matrix file (.dres)

Additionally, if the user enables the option "usenergyfile", then a file containing the input energies or momentum will also be required.

2.3.1 Control file

Each EFT parameter is written on its own line in [mycontrolfile].control, with four values: the keyword "coefnonrel", the operator number (integer 1..16), the coupling type ("p"=proton, "n"=neutron, "s"=scalar, "v"=vecctor), and the coefficient value. For example,

```
coefnonrel    1    s     3.1
```

would set $c_1^{\tau=0}=3.1$. We take the isospin convention:

$$\begin{aligned} c^0 &= c^p + c^n \\ c^1 &= c^p - c^n \end{aligned} \tag{3}$$

Thus, the previous example is equivalent to:

```
coefnonrel    1    p     1.55
coefnonrel    1    n     1.55
```

The control file also serves a more general but optional function: to set any parameter in the program to a custom value. Simply add an entry to the control file with two values: the first should be the keyword for the parameter and the second should be the value to set that parameter to. For example, to set the velocity of the earth in the galactic frame to 240 km/s, you should add the line:

```
vearth  240.0
```

As an example, here is the complete control file used to calculate the event rate for the c_1^n coupling to ^{131}Xe shown in Table **??**:

```
# Coefficient matrix (non-relativistic)
# Ommitted values are assumed to be 0.0.
# c_i^t
# i = 1,...,16
# t: p=proton n=neutron s=scaler v=vector
coefnonrel  1  n  0.00048
wimpmass 150.0
vearth 232.0
maxwellv0 220.0
dmdens 0.3
usemomentum 0
useenergyfile 0
ntscale  2500.0
printdensities 0
#vescape 550.
```

Uncommenting the last line would set the escape velocity to 550 km/s. A complete list of keywords is given in section 6.

2.3.2 Nuclear density matrix file (.dres)

We adopt the output format from the `BIGSTICK` shell-model code. The output one-body densities are written to a file with extension `.dres`. We provide a full specification of this plain-text-file format in the `docs` directory. Here, we show the form of the file and explain its contents.

```
  State      E        Ex         J       T
    1   -330.17116   0.00000     1.500  11.500
  Single particle state quantum numbers
ORBIT      N     L   2 x J
     1     0     2     3
     2     0     2     5
     3     1     0     1
 Initial state #    1 E = -330.17117 2xJ, 2xT =    3  23
 Final state   #    1 E = -330.17117 2xJ, 2xT =    3  23
 Jt =   0, proton      neutron
    1    1   1.55844   5.40558
```

The file is comprised of three sections: (1) many-body state information, (2) single-particle state quantum numbers, (3) density matrix element blocks. Only the ground state is needed for inelastic WIMP-nucleus scattering calculations.

The single-particle state quantum numbers specify the quantum numbers for the simple-harmonic oscillator states involved in the one-body operators.

Finally, the one-body density matrix elements are listed in nested blocks with three layers: (i) the initial and final state specification (corresponding to the many-body states listed in section (1) of the file), (ii) the angular momentum carried by the one-body density matrix operator, labeled `Jt` here, and (iii) the single-particle state labels `a`, `b` in columns 1 and 2 (corresponding to the single-particle state labels listed in section (2) of the file) and the proton and neutron (isospin-0 and isospin-1) density matrix elements in columns 3 and 4.

Both (i) and (ii) must be specified along with columns 1 and 2 of (iii) in order to fully determine a matrix element $\rho_K^{f,i}(a,b)$, where $K = J_t$. Note that the values of K are restricted by conservation of angular momentum; both between the many-body states labeled i and f, and the single-particle states labeled a and b.

2.4 Command-line interface

The program will prompt the user for the minimum necessary inputs to run a calculation with default parameter values, including the name of a control file which contains the EFT coefficients, and optionally, additional changes to the calculation parameters.

After selecting the option `[er]` to compute an event rate spectra, there are six further lines of input. These will be explained by an example:

```
 Enter the target proton number
54
 Enter the target neutron number
77
 Enter name of control file (.control):
xe131
...
  Enter name of one-body density file (.dres)
xe131gcn
...
 What is the range of recoil energies in kev?
 Enter starting energy, stoping energy, step size:
0.0001 250. 1.0
```

The first two entries are self-evident: we specifiy the number of protons and neutrons in the target nucleus. In this case, 54 and 74, respectively, for ^{131}Xe.

Third is the name of the control file containing the EFT coefficients and other, optional, settings. The '.control' file extension should be omitted. This contents of this file will be explained in more detail later.

Fourth is the file containing the nuclear wave functions in the form of one-body density matrices. Only the ground-state need be included. The '.dres' file extension is omitted.

Fifth and finally are three numbers specifying the range of recoil energies E_R that the differential scattering rate should be computed for.

The event rate spectra will be written to a file, and as a side effect of the calculation, the total event rate for the energy range requested will be estimated by numerical integration. Note that the accuracy of this result will depend on the choice of the step size.

2.5 Compute options explained

There are a handful of options available from the main menu of the code:

[er] Differential event rate, for a range of recoil energies or transfer momenta

[cs] Differential cross section at a fixed recoil energy over a range of speeds

[tp] Transition/scattering probability at a fixed recoil energy over a range of speeds

[te] Total integrated events (without producing spectra file)

[wd] Nuclear response functions at a given value of $y = (qb/2)^2$

[ws] Nuclear response function spectra (for a range of q or E)

The string in square brackets [x] is the compute-option. Once a compute-option is chosen, subsequent CLI inputs are the same up until the density matrix file (.dres) has been read-in. Then, the inputs depend on the compute-option chosen.

[cs] Differential cross section per recoil energy. Four additional inputs:

- E-recoil (keV)
- v-start (km/s)
- v-stop (km/s)
- v-step (km/s)

[tp] Scattering probability. Same as [2].

[te] Total scattering events per detector (does not produce spectra data). This option uses adaptive quadrature to perform the integral of the event rate spectra with the fewest number of evaluations to reach the desired relative error. This will be much faster than the result from options [1]. Three additional inputs:

- E-start (keV)
- E-stop (keV)
- Desired relative error (decimal value)

[wd] Nuclear response function test. This compute-option allows the user to evaluate the nuclear response functions $W_i^{x,x'}(y)$ for a provided value of y. All combinations of x and x' will be printed for both isospin and proton-neutron couplings. Two additional inputs are required:

- Function number (1 - 8)
- Value of $y = (qb/2)^2$ (dimensionless)

[ws] Nuclear response function spectra. Enter a range of recoil energy or momentum values to evaluate the nuclear response funcions on. Tabulates the data to a file - one momentum per line or energy per line. Momentum/energy is written to the first column. The following 32 columns store the (8,2,2)-dimensional response functions $W_i^{x,x'}$:

```
q_1  W_1^00 W_2^00 ... W_8^00 W_1^10 ... W_8^11
q_2  W_1^00 W_2^00 ... W_8^00 W_1^10 ... W_8^11
...
q_m  W_1^00 W_2^00 ... W_8^00 W_1^10 ... W_8^11
```

The inputs are:

- E-start (keV)
- E-stop (keV)
- E-step (keV)

For the event-rate spectra [er] and for the nuclear response function spectra [ws], the range of values is either over recoil energy E_r (kev) (the default) or over the transfer momentum q (Gev/c). To use q instead of E_r, use the control word `usemomentum` set to 1.

2.6 Event rate spectra from the Python wrapper

We provide a Python wrapper for the code and a number of example scripts demonstrating its use. The wrapper comes in the form of a Python function `EventrateSpectra` which can be imported from dmfortfactor.py in the Python directory. This function has three required arguments:

1. Z the number of protons in the target nucleus

2. N the number of neutrons in the target nucleus

3. The '.dres' filename for the one-body density matrix file describing the nuclear structure.

If no other arguments are provided, default values will be used for all of the remaining necessary parameters, including zero interaction strength. To calculate an event rate with a nonzero interaction, the user should also provide one or more of the optional EFT coupling coefficient arrays: `cpvec, cnvec, csvec, cvvec`. These store the couplings to protons, neutrons, isoscalar, and isovector, respectively. Finally, the user can also pass a dictionary of valid control keywords and values to the function in order to set any of the control words defined in the manual.

To compute the eventrate spectra for ^{131}Xe with a WIMP mass of 50 GeV and a $c_3^v = 0.0048$ coupling, one might call:

```
import dmfortfactor as dm
control_dict = {"wimpmass" : 50.0}
cvvec = np.zeros(15)
cvvec[2] = 0.0048
Erkev, ER = dm.EventrateSpectra(
            Z = 54,
            N = 77,
            dres = "xe131gcn",
            controlwords = control_dict,
            cvvec = cvvec)
```

This will return the differential event rate spectra for recoil energies from 1 keV to 1 MeV in 1 keV steps. The file 'xe131gcn.dres' must exist in the current working directory and contain a valid one-body density matrix for ^{131}Xe.

3 Nuclear structure input

Users must provide nuclear one-body density matrix elements of the form:

$$\rho^{\Psi}_{K,T}(a,b) = \langle \Psi | [\hat{c}^\dagger_a \hat{c}_b]_{K,T} | \Psi \rangle, \tag{4}$$

where Ψ is the nuclear-target wave function and $\hat{c}^\dagger$, $\hat{c}$ are the one-body creation, destruction operators. The matrix elements must be stored in a file in a standard format produced by shell-model codes like BIGSTICK.

Nuclei	Isotopes	Source
Si	28, 29	[?]
Xe	128, 129, 130, 131, 132, 134, 136	
Ar	40	
C	12	[?]
He	4	

Table 1: Table of nuclear data we include with the program. Each corresponds to a (.dres) density matrix file. The source indicates the nuclear Hamiltonian that was used to generate the wave function data.

3.1 Filling core orbitals for phenomenological interactions

Since standard one-body density matrices in phenomenological model spaces contain only matrix elements for orbitals in the valence space, it is necessary to infer the matrix elements for the core orbitals. Our code does this by default, but the user can disable this option using the `fillnuclearcore` control word.

For phenomenological interactions one typically has a 'frozen' core of nucleons which do not participate in the two-body forces of the Hamiltonian. In such cases the single-particle space listed in the .dres file consists only of the valence orbitals and the one-body density matrices are only specified for the valence orbitals.

DMFormFactor reads the valence space orbitals from the .dres file and infers the number of core nucleons by subtracting the number of valence protons and neutrons from the number of nucleons in the target nucleus. The core orbitals are assumed to be one of the standard shell model orbital sets associated with possible cores: He-4, O-16, Ca-40, Ni-56, Sn-100.

The one-body density matrix elements for the core orbitals are then determined from the (full) occupation of the core orbitals. In proton-neutron format:

$$\rho^{\Psi}_{J,x=p,n}(a,b)_{(core)} = \delta_{a,b}[j_a][J], \tag{5}$$

where $[y] \equiv \sqrt{2y+1}$ and j_a is the angular momentum of a-orbit. J is the total spin of the nuclear target state Ψ. And in isospin format for a target state with total isospin T:

$$\rho^{\Psi}_{J,\tau=0}(a,b)_{(core)} = \delta_{a,b}[1/2][j_a][J][T], \tag{6}$$

$$\rho^{\Psi}_{J,\tau=1}(a,b)_{(core)} = 0.0. \tag{7}$$

4 Python interface

We provide two generic API's for interacting with the Fortran program. They are: `runTemplates` and `EventrateSpectra`. These can be imported into your own Python script by having the file `dmfortfactoy.py` in your working directory or by adding it to your path:

```
import sys
sys.path.append(’/path/to/dmfortfactor.py’)
import dmfortfactor as dm
```

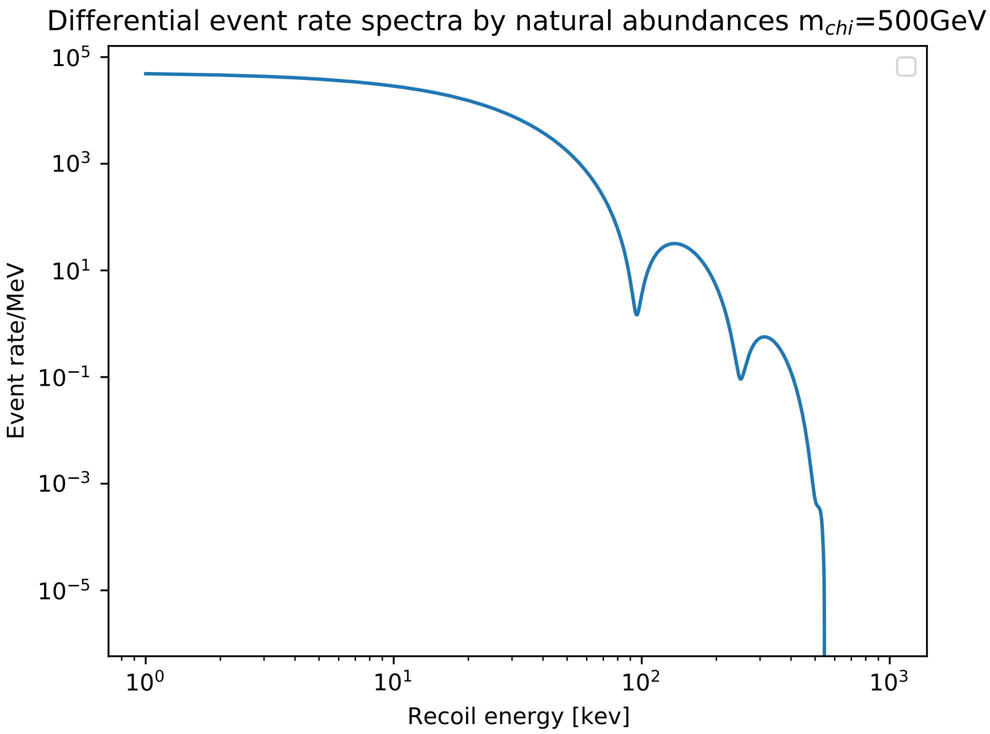
`EventrateSpectra` is essentially a wrapper for the event-rate calculation function of `dmfortfactor`. It makes use of `runTemplates` but shields the user from having to handle the input and control files needed by `dmfortfactor`.

`runTemplates` is a fairly generic function which runs an arbitrary linux program which takes an input file by a linux pipe, and makes use of a secondary 'control' file. `runTemplates` takes in templates for the input and control files, modifies them using `sed` to replace string keys with values from a Python dictionary, runs the executable, then collects outputs written to an output file, returning the data as Numpy arrays.

4.1 Example: generating a weighted sum of event rate spectra

```
import sys
sys.path
   .append("/Users/oliver/projects/darkmatter/python")
import numpy as np
import matplotlib.pyplot as plt
import dmfortfactor as dm

plt.figure()
Z = 54
isotopes = [128, 129, 130, 131, 132, 134, 136]
weights = [.01910,
    .26401, .04071, .21232, .26909, .10436, .08857]
weightedsum = 0.0

for i,isotope in enumerate(isotopes):
    N = isotope - Z
    controls = {
            "wimpmass" :500.0,
            "vescape" : 550.0}
    cp = np.zeros(15)
    cp[0] = 1
    RecoilE, EventRate = dm.EventrateSpectra(
             Z,
             N,
             dres = "xe%igcn"%isotope,
             controlwords = controls,
             cpvec = cp )

    weightedsum += EventRate * weights[i]

    label = "xe"+str(isotope)
    plt
   .plot(RecoilE, EventRate, label="$^{%s}$Xe"%isotope)
```

We now have the weighted sum of event rates stored in the variable weightedsum. Using matplotlib, we could produce the following plot:



4.2 Example: comparing event-rate spectra for calculations with different WIMP masses

```python
import sys
sys.path
   .append("/Users/oliver/projects/darkmatter/python")

import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import dmfortfactor as dm

filename_base = "GeV-rate.dat"
g = open(filename_base+".csv", "w+")
g.write("# WIMP_mass, Recoil_energy, Event_rate\n")

masses = np.arange(1.0, 300., 1) # GeV
for WIMPMASS in masses:

    print("m = %s"%WIMPMASS)
    # Control words with a variable wimp mass
    control_dict = {
            "wimpmass" : WIMPMASS,
            "vearth" : 232.0,
            "ntscale" : 2500.0,
            "maxwellv0" : 220.0
    }

    # EFT coupling: isovector
    cvvec = np.zeros(15)
    cvvec[0] = 0.00048

    # Evaluate the model
    E, R = dm.EventrateSpectra(
        Z = 54,
        N = 77,
        dres = "../dres/xe131gcn",
        epmin = 1.0,
        epmax = 1000.0,
        epstep = 1.0,
        controlwords = control_dict,
        cvvec = cvvec)

    f = open(str(WIMPMASS)+filename_base, "w+")
    f.write("
   # Event rate (events/GeV) for %s GeV WIMPs."%WIMPMASS)
    f.write
   ("# Recoil energy (keV)               # event rate")
    for ii, energy in enumerate(E):
        f.write("%10.5f    %s\n"%(energy, R[ii]))
    f.close()

    for ii, energy in enumerate(E):

      g.write("%s, %10.5f, %s\n"%(WIMPMASS,energy,R[ii]))
g.close()
```

5 Equations found in the code

We present the equations necessary to reproduce the code. For a more complete description of the theory, see Phys. Rev. C 89.065501.

5.1 Differential event rate

$$\frac{dR}{dE_r}(E_r)=N_T n_\chi \int_{v_{min}}^{v_{escape}} \frac{d\sigma}{dE_r}(v,E_r)\tilde{f}(\vec{v})vd^3v, \tag{8}$$

where E_r is the recoil energy of the WIMP-nucleus scattering event, N_T is the number of target nuclei, $n_\chi = \rho_\chi/m_\chi$ is the local dark matter number density, σ is the WIMP-nucleus cross section. The dark matter velocity distribution in the lab frame, $\tilde{f}(\vec{v})$, is obtained by boosting the Galactic-frame distribution $f(\vec{v})$: $\tilde{f}(\vec{v}) = f(\vec{v}+\vec{v}_{earth})$, where $\vec{v}_{earth}$ is the velocity of the earth in the galactic rest frame. The simplest model is a three-dimensional Maxwell distribution:

$$f(\vec{v}) \propto e^{-\vec{v}^2/v_0^2}, \tag{9}$$

where v_0 is some scaling factor (typically taken to be around 220 km/s).

In order to evaluate the integral in (8), we make the conversion to spherical coordinates, and take special care to deal with the velocity boost in (**??**). Assuming a $1/v^2$ velocity dependence of the cross section term (see section 5.2), we need to evaluate an integral of the form

$$I = \int_{v_{min}}^{v_{max}} d^3v \frac{f(\vec{v}+\vec{v}_{earth})}{v} = \int_{v_{min}}^{v_{max}} d^3v \frac{1}{v} e^{-(\vec{v}+\vec{v}_{earth})^2/v_0^2} \tag{10}$$

Noting that $(\vec{v}+\vec{v}_{earth})^2 = \vec{v}^2 + \vec{v}_{earth}^2 + 2vv_{earth}\cos(\theta)$, with $|\vec{v}| \equiv v$ and θ defining the angle between the two vectors, it's convenient to make the substitution

$d^3v = d\phi d(\cos\theta)v^2dv$:

$$\begin{aligned}
I &= \int_0^{2\pi} d\phi \int_{v_{min}}^{v_{max}} dv \int_{-1}^{1} d(\cos\theta) e^{-2vv_{earth}\cos\theta/v_0^2} v^2 \frac{1}{v} e^{-(\vec{v}^2+\vec{v}_{earth}^2)/v_0^2} \\
&= 2\pi \int_{v_{min}}^{v_{max}} dv v e^{-(\vec{v}^2+\vec{v}_{earth}^2)/v_0^2} \left(-\frac{v_0^2}{2vv_{earth}} e^{-2vv_{earth}\cos\theta/v_0^2} \right)_{-1}^{1} \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv e^{-(\vec{v}^2+\vec{v}_{earth}^2)/v_0^2} \left(-e^{-2vv_{earth}/v_0^2} + e^{+2vv_{earth}/v_0^2} \right) \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv \left(-e^{(v+v_{earth})^2/v_0^2} + e^{(v-v_{earth})^2/v_0^2} \right) \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv (g(v-v_{earth}) - g(v+v_{earth}))
\end{aligned} \tag{11}$$

where in the last equality, we have defined a one-dimensional Gaussian form

$$g(v) \propto e^{-v^2/v_0^2}. \tag{12}$$

The final expression for I can be trivially generalized to other spherically symmetric velocity-dependent forms of the differential cross section. What's important is the reduction of the velocity-boosted d^3v integral to a radial integral which can be carried out with one-dimensional quadrature:

$$\begin{aligned}
&\int_{v_{min}}^{v_{max}} d^3v \sigma(v) e^{-(\vec{v}+\vec{v}_{earth})^2/v_0^2} \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv \sigma(v) v^2 (g(v-v_{earth}) - g(v+v_{earth})).
\end{aligned} \tag{13}$$

The FORTRAN code uses equation (13) to evaluate the event rate integral in equation (8) with quadrature. Analytic solutions of (13) exist in the form of error functions; we use the above form since it makes easy to later modify the velocity distribution (as long as it remains spherically symmetric). For example, adding a velocity cut-off is as easy as changing the limit on the quadrature, with no need to write a whole new subroutine for the analytic forms found in the Mathematica script.

5.2 Differential cross section

$$\frac{d\sigma(v,E_R)}{dE_R} = 2m_T \frac{d\sigma(v)(v,\vec{q}^2)}{d\vec{q}^2} = 2m_T \frac{1}{4\pi v^2} T(v,q), \tag{14}$$

Where v is the velocity of the dark matter particles in the lab-frame, q is the momentum transfer of the scattering event, m_T is the mass of the target nucleus, and $T(v,q)$ is the transition or scattering probability. We can see here that the differential cross section has an explicit $1/v^2$ dependence, independent of any velocity dependence of $T(v,q)$.

5.3 Transition probability / Scattering probability

The scattering probability is

$$T(v,q) = \frac{1}{2j_\chi + 1} \frac{1}{2j_T + 1} \sum_{spins} |\mathcal{M}(v,q)|^2 \tag{15}$$

where j_χ is the spin of the WIMP, j_T is the spin angular momentum of the target nucleus, and $\mathcal{M}$ Galilean invariant amplitude, which is defined by

$$T(v,q) = \frac{4\pi}{2j_T + 1} \frac{1}{(4m_\chi)^2} \sum_{x=p,n} \sum_{x'=p,n}^{1} \sum_{i=1}^{8} R_i^{xx'}(v^2,q^2) W_i^{xx'}(q) \tag{16}$$

where m_χ is the mass of the dark matter particle and x is an index used to sum over isospin couplings. The coefficients $R_i^{x,x'}$ are dark matter particle response functions, to be define in another section. The operators $W_i^{xx'}(q)$ are nuclear response functions, which are sums over matrix elements of nuclear operators constructed from Bessel spherical harmonics and vector spherical harmonics.

5.4 Dark matter response functions

There are 8 dark matter response functions which group 15 operator coefficients c_i^x according the pair of nuclear response functions which they multiply.

As a shorthand, $cl(j) \equiv 4j(j+1)/3$, and $v^{\perp 2} \equiv v^2 - (q/2\mu_t)^2$.

$$\begin{aligned}
R_M^{xx'}(v,q) &= \frac{1}{4} cl(j_\chi)[v^{\perp 2}(c_5^x c_5^{x'} q^2 + c_8^x c_8^{x'}) + c_{11}^x c_{11}^{x'} q^2] \\
&\quad + (c_1^x + c_2^x v^{\perp 2})(c_1^{x'} + c_2^{x'} v^{\perp 2}) \\
R_{\Sigma''}^{xx'}(v,q) &= \frac{1}{16} cl(j_\chi)[c_6^x c_6^{x'} q^4 + (c_{13}^x c_{13}^{x'} q^2 + c_{12}^x c_{12}^{x'}) v^{\perp 2} + 2 c_4^x c_6^{x'} q^2 + c_4^x c_4^{x'}] + \frac{1}{4} c_{10}^x c_{10}^{x'} q^2 \\
R_{\Sigma'}^{xx'}(v,q) &= \frac{1}{32} cl(j_\chi)[2 c_9^x c_9^{x'} q^2 + (c_{15}^x c_{15}^{x'} q^4 + c_{14}^x c_{14}^{x'} q^2 \\
&\quad - 2 c_{12}^x c_{15}^{x'} q^2 + c_{12}^x c_{12}^{x'}) v^{\perp 2} + 2 c_4^x c_4^{x'}] + \frac{1}{8}(c_3^x c_3^{x'} q^2 + c_7^x c_7^{x'}) v^{\perp 2} \\
R_{\Phi''}^{xx'}(v,q) &= \frac{q^2}{16 m_N^2} cl(j_\chi)(c_{12}^x - c_{15}^x q^2)(c_{12}^{x'} - c_{15}^{x'} q^2) + \frac{q^4}{4 m_N^2} c_3^x c_3^{x'}
\end{aligned}$$

Table 2: Table of EFT coefficient interactions. Shows which coefficients multiply each coefficient in addition to itself.

Coefficient	Couples to
1	2, 3
2	1, 3
3	1, 2
4	5, 6
5	4
6	4
7	
8	9
9	8
10	
11	12, 15
12	11, 15
13	
14	
15	11, 12

$$
\begin{aligned}
R^{xx'}_{\tilde{\Phi}'}(v,q) &= \frac{q^2}{16m_N^2} cl(j_\chi)(c^x_{13}c^{x'}_{13}q^2 + c^x_{12}c^{x'}_{12}) \\
R^{xx'}_{\Delta}(v,q) &= \frac{q^2}{4m_N^2} cl(j_\chi)(c^x_5 c^{x'}_5 q^2 + c^x_8 c^{x'}_8) + 2\frac{q^2}{m_N^2} c^x_2 c^{x'}_2 v^{\perp 2} \\
R^{xx'}_{\Delta\Sigma'}(v,q) &= \frac{q^2}{4m_N} cl(j_\chi)(c^x_4 c^{x'}_5 - c^x_8 c^{x'}_9) - \frac{q^2}{m_N} c^x_2 c^{x'}_3 v^{\perp 2} \\
R^{xx'}_{\Phi'' M}(v,q) &= \frac{q^2}{4m_N} cl(j_\chi) c^x_{11}(c^{x'}_{12} - c^{x'}_{15} q^2) + \frac{q^2}{m_N} c^{x'}_3 (c^x_1 + c^x_2 v^{\perp 2})
\end{aligned}
$$

5.5 Cross terms

Table 2 lists all EFT coefficient cross-couplings. We can use this table to create a minimal list of inputs to validate all possible nonzero couplings. In addition to each coefficient on its own ($i = 1,...,15$), one should also test the following 9 unique combinations: (1,2), (1,3), (2,3), (4, 5), (5,6), (8,9), (11,12), (11,15), (12,15).

5.6 Operators

The code uses the EFT coefficients in explicit proton-neutron couplings, i.e. the interaction is defined by:

$$\mathcal{H}=\sum_{x=p,n}\sum_{i=1,15}c_i^x\mathcal{O}_i \tag{17}$$

and the 15 momentum-dependent operators are:

$$\mathcal{O}_1=1_\chi 1_N \tag{18}$$
$$\mathcal{O}_2=(v^\perp)^2 \tag{19}$$
$$\mathcal{O}_3=i\vec{S}_N\cdot\left(\frac{\vec{q}}{m_N}\times\vec{v}^\perp\right) \tag{20}$$
$$\mathcal{O}_4=\vec{S}_\chi\cdot\vec{S}_N \tag{21}$$
$$\mathcal{O}_5=i\vec{S}_\chi\cdot\left(\frac{\vec{q}}{m_N}\times\vec{v}^\perp\right) \tag{22}$$
$$\mathcal{O}_6=\left(\vec{S}_\chi\cdot\frac{\vec{q}}{m_N}\right)\left(\vec{S}_N\cdot\frac{\vec{q}}{m_N}\right) \tag{23}$$
$$\mathcal{O}_7=\vec{S}_N\cdot\vec{v}^\perp \tag{24}$$
$$\mathcal{O}_8=\vec{S}_\chi\cdot\vec{v}^\perp \tag{25}$$
$$\mathcal{O}_9=i\vec{S}_\chi\cdot\left(\vec{S}_N\times\frac{\vec{q}}{m_N}\right) \tag{26}$$
$$\mathcal{O}_{10}=i\vec{S}_N\cdot\frac{\vec{q}}{m_N} \tag{27}$$
$$\mathcal{O}_{11}=i\vec{S}_\chi\cdot\frac{\vec{q}}{m_N} \tag{28}$$
$$\mathcal{O}_{12}=\vec{S}_\chi\cdot\left(\vec{S}_N\times\vec{v}^\perp\right) \tag{29}$$
$$\mathcal{O}_{13}=i\left(\vec{S}_\chi\cdot\vec{v}^\perp\right)\left(\vec{S}_N\cdot\frac{\vec{q}}{m_N}\right) \tag{30}$$
$$\mathcal{O}_{14}=i\left(\vec{S}_\chi\cdot\frac{\vec{q}}{m_N}\right)\left(\vec{S}_N\cdot\vec{v}^\perp\right) \tag{31}$$
$$\mathcal{O}_{15}=-\left(\vec{S}_\chi\cdot\frac{\vec{q}}{m_N}\right)\left(\left(\vec{S}_N\times\vec{v}^\perp\right)\cdot\frac{\vec{q}}{m_N}\right) \tag{32}$$

5.7 Nuclear response functions

There are eight nuclear response functions $W_i^{xx'}(y)$ considered here. The unit-less variable y is defined

$$y = \left(\frac{qb}{2}\right)^2, \tag{33}$$

in terms of the harmonic oscillator size parameter b, which has a default value of

$$b^2 = 41.467/(45A^{-1./3} - 25A^{-2/3})\ fm^2. \tag{34}$$

$$W_M^{xx'}(y) = \sum_{even\ J} \langle j_T | M_{Jx}(y) | j_T \rangle \langle j_T | M_{Jx'}(y) | j_T \rangle \tag{35}$$

$$W_{\Sigma''}^{xx'}(y) = \sum_{odd\ J} \langle j_T | \Sigma''{}_{Jx}(y) | j_T \rangle \langle j_T | \Sigma''{}_{Jx'}(y) | j_T \rangle \tag{36}$$

$$W_{\Sigma'}^{xx'}(y) = \sum_{odd\ J} \langle j_T | \Sigma'{}_{Jx}(y) | j_T \rangle \langle j_T | \Sigma'{}_{Jx'}(y) | j_T \rangle \tag{37}$$

$$W_{\Phi''}^{xx'}(y) = \sum_{even\ J} \langle j_T | \Phi''{}_{Jx}(y) | j_T \rangle \langle j_T | \Phi''{}_{Jx'}(y) | j_T \rangle \tag{38}$$

$$W_{\tilde{\Phi}'}^{xx'}(y) = \sum_{even\ J} \langle j_T | \tilde{\Phi}'_{Jx}(y) | j_T \rangle \langle j_T | \tilde{\Phi}'_{Jx'}(y) | j_T \rangle \tag{39}$$

$$W_{\Delta}^{xx'}(y) = \sum_{odd\ J} \langle j_T | \Delta_{Jx}(y) | j_T \rangle \langle j_T | \Delta_{Jx'}(y) | j_T \rangle \tag{40}$$

$$W_{\Delta\Sigma'}^{xx'}(y) = \sum_{odd\ J} \langle j_T | \Delta_{Jx}(y) | j_T \rangle \langle j_T | \Sigma'{}_{Jx'}(y) | j_T \rangle \tag{41}$$

$$W_{\Phi'' M}^{xx'}(y) = \sum_{even\ J} \langle j_T | \Phi''{}_{Jx}(y) | j_T \rangle \langle j_T | M_{Jx'}(y) | j_T \rangle \tag{42}$$

The sums over J are determined by the transformation properties of each multipole operator and the restriction to elastic scattering in which we assume conservation of parity and CP symmetry.

5.8 Nuclear operators and their matrix elements

There are six nuclear operators constructed from Bessel spherical harmonics and vector spherical harmonics, and are evaluated here on the ground state of the target nucleus.

There are six nuclear operators $\mathcal{W}^{(f)}$, $f=1,...,6$, describing the electro-weak coupling of the WIMPs to the nucleon degrees of freedom. The six single-particle operators are given the symbols:

$$\mathcal{W}_J^{(f=1,...,6)}=M_J,\Delta_J,\Sigma'_J,\Sigma''_J,\tilde{\Phi}'_J,\Phi''_J, \tag{43}$$

and are constructed from Bessel spherical and vector harmonics [?]:

$$M_{JM}(q\vec{x}) \equiv j_J(qx)Y_{JM}(\Omega_x) \tag{44}$$

$$\vec{M}_{JML}(q\vec{x}) \equiv j_L(qx)\vec{Y}_{JLM}(\Omega_x), \tag{45}$$

where

$$Y_{JLM}(\Omega_x) = \sum_{m\lambda} \langle Lm1\lambda|\,|(L1)JM_J\rangle\, Y_{Lm}(\Omega_x)\vec{e}_\lambda. \tag{46}$$

The six multipole operators are defined as:

$$M_{JM} \tag{47}$$

$$\Delta_{JM} \equiv \vec{M}_{JJM} \cdot \frac{1}{q}\vec{\nabla} \tag{48}$$

$$\Sigma'_{JM} \equiv -i\left\{\frac{1}{q}\vec{\nabla}\times\vec{M}_{JJM}\right\}\cdot\vec{\sigma} \tag{49}$$

$$\Sigma''_{JM} \equiv \left\{\frac{1}{q}\vec{\nabla}M_{JM}\right\}\cdot\vec{\sigma} \tag{50}$$

$$\tilde{\Phi}'_{JM} \equiv \left(\frac{1}{q}\vec{\nabla}\times\vec{M}_{JJM}\right)\cdot\left(\vec{\sigma}\times\frac{1}{q}\vec{\nabla}\right)+\frac{1}{2}\vec{M}_{JJM}\cdot\vec{\sigma} \tag{51}$$

$$\Phi''_{JM} \equiv i\left(\frac{1}{q}\vec{\nabla}M_{JM}\right)\cdot\left(\vec{\sigma}\times\frac{1}{q}\vec{\nabla}\right) \tag{52}$$

The matrix elements of these operators can be calculated for standard wave functions from second-quantized shell model calculations:

$$\begin{aligned}\langle\Psi_f|\mathcal{W}_J^{(f)}|\Psi_i\rangle &= \mathrm{Tr}\left(\mathcal{W}_J^{(f)}\rho_J^{f,i}\right) \\ &= \sum_{a,b}\langle a||\mathcal{W}_J^{(f)}||b\rangle\rho_J^{fi}(ab),\end{aligned} \tag{53}$$

where single-particle orbital labels a imply shell model quantum number n_a,l_a,j_a, and the double-bar $||$ indicates reduced matrix elements [?]. We assume a harmonic oscillator single-particle basis, with the important convention that the radial nodal quantum number n_a starts at 0, that is, we label the orbitals as $0s$,$0p$,$1s0d$, etc.., and *not* starting with $1s$,$1p$, etc. Then the one-body matrix elements for operators $\langle a||\mathcal{W}_J^{(f)}||b\rangle$, built from spherical Bessel functions and vector spherical harmonics, have closed-form expressions in terms of confluent hypergeometric functions [**?**].

${}_1F_1$ is the confluent hypergeometric function:

$$ {}_1F_1(a,b,z)=\sum_{n=0}^{\infty}\frac{a^{(n)}z^n}{b^{(n)}n!}, \tag{54}$$

which makes use of the rising factorial function:

$$m^{(n)}=\frac{(m+n-1)!}{(m-1)!}. \tag{55}$$

6 Control file keywords

Keyword	Symbol	Meaning	Units	Default
dmdens	ρ_χ	Local dark matter density.	GeV/cm^3	0.3
dmspin	j_χ	Instrinsic spin of WIMP particles.	$\hbar$	$\frac{1}{2}$
fillnuclearcore		Logical flag (enter 0 for False, 1 for True) to fill the inert-core single-particle orbitals in the nuclear level densities. Phenomenological shell model calculations typically provide only the density matrices for the active valence-space orbitals, leaving it to the user to infer the core-orbital densities. This option automatically assigns these empty matrix elements assuming a totally filled core.		1 (true)
gaussorder		Order of the Gauss-Legendre quadrature to use when using Type 2 quadrature. (See quadtype.) An n-th order routine will perform n function evaluations. Naturally, a higher order will result in higher precision, but longer compute time.		12

hofrequency	$\hbar\omega$	Set the harmonic oscillator length by specifying the harmonic oscillator frequency. (b = 6.43/sqrt($\hbar\omega$)). If using an *ab initio* interaction, $\hbar\omega$ should be set to match the value used in the interaction.	MeV	See hoparameter.
hoparameter	b	Harmonic oscillator length. Determines the scale of the nuclear wavefunction interaction.	fm	See eqn. (34).
maxwellv0	v_0	Maxwell-Boltzman velocity distribution scaling factor.	km/s	220.0
mnucleon	m_N	Mass of a nucleon. It's assumed that $m_p \approx m_n$.	GeV	0.938272
ntscale	N_t	Effective number of target nuclei scaling factor. The differential event rate is multiplied by this constant in units of kilogram-days. For example, if the detector had a total effective exposure of 2500 kg days, one might enter 2500 for this value.	kg days	1.0
quadrelerr		Desired relative error for the adaptive numerical quadrature routine (quadtype 1).		10^{-6}
quadtype		Option for type of numerical quadrature. (Type 1 = adaptive 8th order Gauss-Legendre quadrature. Type 2 = static n-th order Gauss-Legendre quadrature.)		1 (type 1)
sj2tablemax		Maximum value of $2 \times J$ used when caching Wigner 3-J and 6-J functions into memory.		12
sj2tablemin		Minimum value of $2 \times J$ used when caching Wigner 3-J and 6-J functions into memory.		-2
useenergyfile		Logical flag (enter 0 for False, 1 for True) to read energy grid used for calculation from a user-provided file intead of specifying a range.		0 (false)

usemomentum		Logical flag (enter 0 for False, 1 for True) to use momentum transfer intead of recoil energy as the independent variable.		0 (false)
vearth	v_{earth}	Speed of the earth in the galactic frame.	km/s	232.0
vescape	v_{escape}	Galactic escape velocity. Particles moving faster than this speed will escape the galaxy, thus setting an upper limit on the WIMP velocity distribution.	km/s	$12 \times v_{scale}$
weakmscale	m_v	Weak interaction mass scale. User defined EFT coefficients are divided by m_v^2.	GeV	246.2
wimpmass	m_χ	WIMP particle mass.	GeV	50.0