DMFortFactor: A FORTRAN Program for Experimental WIMP Analysis

Oliver Gorton, Changfeng Jiao and Calvin Johnson

May 17, 2021

Model-Independent WIMP Scattering Responses and Event Rates

Contents

1 Preamble

This is the manual for the FORTRAN version of the model- independent WIMP scattering response and event rate code, which was originally written in Mathematica and described in arXiv:1308.6288. We call our program `DMFortFactor`, a Fortran interpretation of DMFormFactor.

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

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

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 16 coupling coefficients defining an interaction:

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

2 Quickstart 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 advanced 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 `src/` directory from wherever you have stored `darkmatter/` (e.g. `cd src/`, or `cd ~/Downloads/darkmatter/src/`)

2. Run the command: `make new`

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

4. Run the command: `../../src/dmfortfactor.x < xe131.input`

If successful, one of the last lines printed to screen should be "Event rate spectra written to eventrate˙spectra.dat".

2.2 Compiling with make

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 a `Makefile` to compile the code using a few different options.

The easiest way to get started is simply to navigate to `src/` and run

```
make dmfortfactor
```

This will compile `DMFortFactor` using `gfortran`. If you want to use a different compiler, you must edit the Makefile to change the `COMP` variable from gfortran to your compiler of choice.

Aditionally, if you want to utilize multiple CPU cores to speed up the runtime, you can compile with `openMP`:

```
make openmp
```

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

```
make clean
```

in between builds.

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.

2.4 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 $c_1^0 = 3.1$. 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
```

A complete list of keywords is given in section 5.

2.5 Event rate spectra (events per GeV) using the CLI

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 customizations to the calculation parameters.

After selecting the option 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 is 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 estimatted by numerical integration. Note that the accuracy of this result will depend on the choice of the step size.

2.6 Python interface

We provide three generic API's for interacting with the Fortran program through a python interface. They are: `runCustomControl`, `runCustomInput`, and `runTemplates`. The first runs the Fortran code with a control file which has been programatically modified by the API then returns the results to the calling environment. The second does the same, except with a custom input file, rather than a custom control file. The third combines the functionality of both. Here we demonstrate an application of `runTempaltes` to generate a plot of event rate specra for a range of WIMP masses.

2.7 Example: using runCustomInput to generate a weighted sum of event rate spectra

In this example we will construct a simple python script using the runCustomInput API to compute the differential event rate spectra for several isotopes of Xe, and compute the weighted sum according to the natural abundances of each isotope.

Firt, examining the API for runCustomInput:

```
1 def runCustomInput(exec_name, input_template, param_dict,
2     workdir='', label='runCustom',resultfile='eventrate_spectra.dat'):
3
4     """
5      Author: Oliver Gorton, 2020
6
```

```
    This is a function for running an executable program with a custom
    input file by replacing keywords in a template used by the executed
    code.

    exec_name: (string) containing the path to and name of the
        executable you want to run
    input_template: (string) this is an input or control file used by the
        executable and which will be edited by this function
    param_dict: (dictionary) of keywords and the values to which those
        keywords will be changed. The keywords are strings which should
        appear in the input_template file.
    workdir: (string) name of the working directory where you want to
        call the executable and send the output.
    label: (string) name for this job, to label the input and output
        files generated by this function.

    """
```

The first three inputs are mandatory while the last three are optional, having default values defined in the function. We begin constructing our script by creating the generic inputs for the executatble name and the directory where the calculations will be done (this is where the required files for the calculation should be). We also specify the atomic number and natural abundances for Xenon isotopes:

```
import os
import matplotlib.pyplot as plt
import runCustomInput

exec_name = "dmfortfactor.x"
workdir = os.getcwd()

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

Now, we will need the input-file template. Having created the template file, for example:

```
1
54
NEUTRONS
xe
xeAAAgcn
1. 1000. 1.0
```

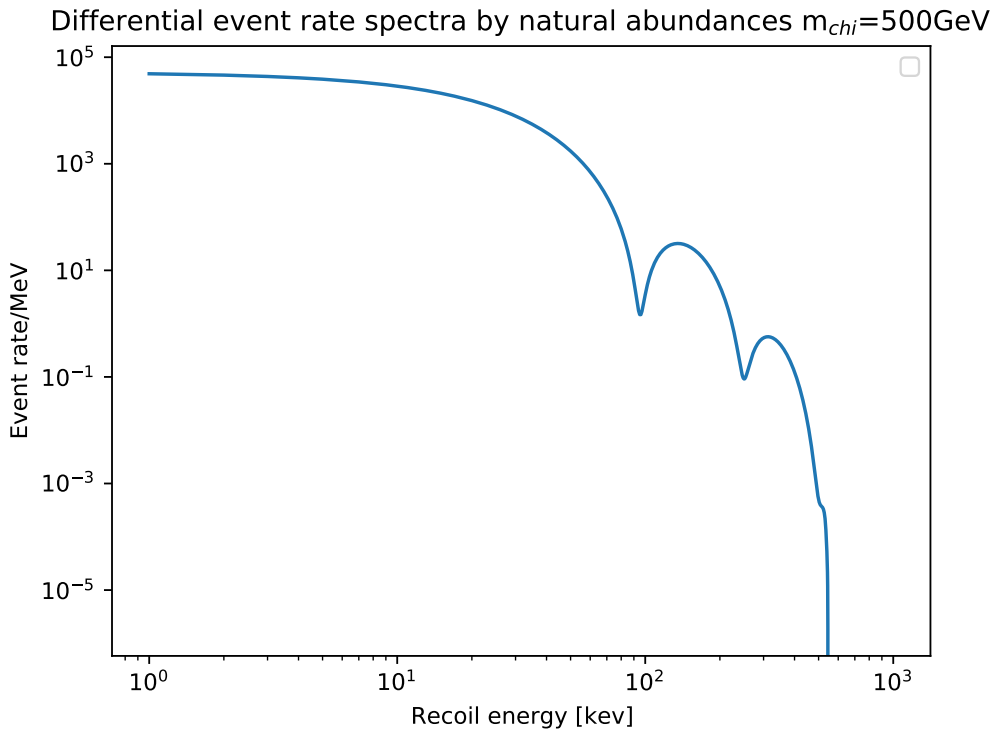
with our arbitrary choice of keywords "NEUTRONS" and "AAA", let's loop over our isotopes, and for each assemble the dictionary of keywords which the API will replace with values:

```
input_template = "input.xeAAA"
for i,isotope in enumerate(isotopes):
    N = isotope - Z
    label = "xe"+str(isotope)
    param_dict = { "NEUTRONS" : N, "AAA" : isotope }
```

We now have all of the required arguments: exec_name, input_template, param_dict, workdir, and label. We are ready to call runCustomInput, which returns the event rate spectra vs recoil energy arrays. All-together now:

```
import os
import matplotlib.pyplot as plt
import runCustomInput

exec_name = "dmfortfactor.x"
workdir = os.getcwd()

Z = 54
isotopes = [128, 129, 130, 131, 132, 134, 136]
weights = [.01910, .26401, .04071, .21232, .26909, .10436, .08857]
weightedsum = 0.0
input_template = "input.xeAAA"
for i,isotope in enumerate(isotopes):
    N = isotope - Z
    label = "xe"+str(isotope)
    param_dict = { "NEUTRONS" : N, "AAA" : isotope }

    RecoilE, EventRate = runCustomInput.runCustomInput(exec_name,
            input_template, param_dict, workdir, label)

    weightedsum += EventRate * weights[i]
```

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



2.8 Example: using runCustomControl to compare event-rate spectra for calculations with different WIMP masses

In this example we will construct a simple python script using the runCustomControl API to compute the differential event rate spectra for several values of the WIMP particle mass. This is very similar in implementation to the previous example, with the main difference being that the WIMP mass is edited by the user through the control file, rather than through the interactive input or input file. The appropriate API for this task is therefore **runCustomControl**:

```
def runCustomControl(exec_name, inputfile, control_template,
    param_dict,
    workdir='', label='runCustom',resultfile='eventrate_spectra.dat'):

    """
     Author: Oliver Gorton, 2020

     This is a function for running an executable program with a
    custom
     control file by replacing keywords in a template used by the
    executed
     code. The control file is one which is used indirectly (not a by
    a
```

```
    command line pipe). For example, an interaction file for bigstick
   .

    exec_name:
        (string) containing the path to and name of the
        executable you want to run
    inputfile:
        (string) this is an input file used by the executable
    control_template:
        (string) the base-name of a file ending in
        '.control'. "<control_template>.template" is the template
        file which the function will edit and copy into
        <control_template>.
    param_dict:
        (dictionary) of keywords and the values to which those
        keywords will be changed. The keywords are strings which
   should
        appear in the input_template file.
    workdir:
        (string) name of the working directory where you want to call
        the executable and send the output.
    label:
        (string) name for this job, to label the input and output
   files
        generated by this function.

    """
```

This API has the same inputs as runCustomInput, with 'inputfile' intead of 'input_template', and one additional argument: 'control_template'.

We again start by constructing the necessary arguments:

```
import os
import matplotlib.pyplot as plt
import runCustomControl

exec_name = "dmfortfactor.x"
inputfile = 'input.xe131'
control_template = "xe131.control"
workdir = os.getcwd()
```

As in the previous example, inputfile is a standard input file we have placed in the current working directory. This time, it is a complete input file with no replaceable keywords:

```
1
54
77
xe131
xe131gcn
1. 1000. 10.0
```

Alongside this standard input file, we also create a template control file called "xe131.control.template". When the API replaces the keywords in this file, it will strip the .template filename extension, creating a file called "xe131.control", which is referenced in the fourth line of the above input file. Within this template file, we find the line (along with other necessary and/or optional control commands not listed for brevity):

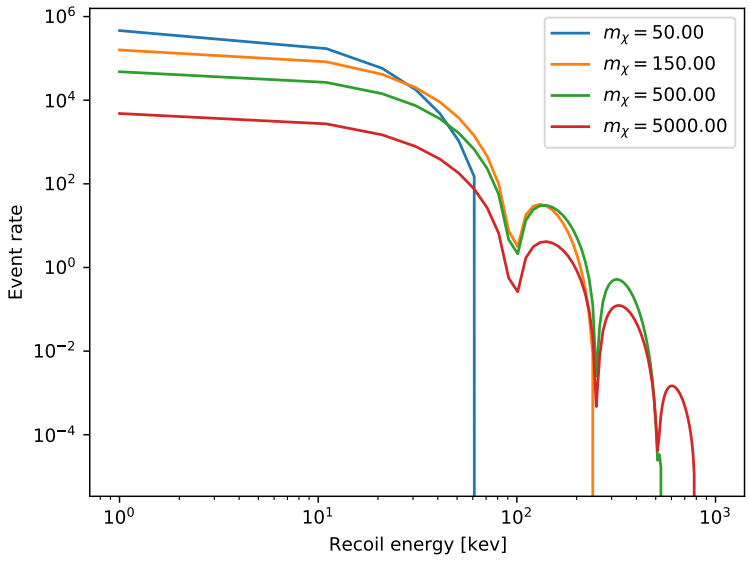
```
wimpmass MASS_KEYWORD
```

'wimpmass' is the keyword for setting the WIMP mass, and 'MASS_KEYWORD' is our arbitrary keyword for the Python API to replace.

We complete the script by looping over different WIMP masses, creating the dictionary or keywords to replace and their values, and finally call the runCustomControl API:

```
import os
import matplotlib.pyplot as plt
import runCustomControl

exec_name = "dmfortfactor.x"
inputfile = 'input.xe131'
control_template = "xe131.control"
workdir = os.getcwd()
for wimp_mass in (50.,150.0, 500.,5000.):
    param_dict = { "MASS_KEYWORD" : wimp_mass }
    RecoilE, EventRate = runCustomControl.runCustomControl(exec_name,
            inputfile, control_template, param_dict, workdir)
```

We could, for example, plot the result at each iteration and construct the following plot:



3 Other compute options

3.1 Total integrated events

Inputs 2 - 6 for this calculation will be almost identical to those required for the event rate spectra calculation. The main difference is in the sixth and final input:

```
 Using adaptive numerical integration to determine total
 integrated event rate. What are the limits of integration
 for recoil energies in kev? Enter starting energy,
 stoping energy, relative error:
0.0 250.0 0.001
```

Here, we have again been asked for a starting and stopping recoil energy, but this time the third value is the desired relative error of the integrated spectra. In this example, 0.001 corresponds to a desired uncertainty of 0.1%.

An important difference between this calculation and the event rate spectra calculation is that in this evaluation, the spectra is not written to file. This is because an adaptive integration routine is used to keep the number of function evaluations to a minimum. As a result, this calculation will be much faster than computing the entire spectra.

4 Equations found in the code

4.1 Differential event rate

$$\frac{dR_D}{dE_R} = \frac{dR_D}{d\vec{q}^2}(q) = N_T n_\chi \int_{v_{min}}^{\infty} \frac{d\sigma(v,q)}{d\vec{q}^2} \tilde{f}(\vec{v}) v d^3v \tag{4}$$

where q is the WIMP-nucleon momentum transfer, 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-nucleon cross section, and $\tilde{f}$ is dark matter velocity distribution in the lab-frame. $\tilde{f}(\vec{v})$ is obtained by boosting the Galactic-frame distribution $f(\vec{v})$,

$$\tilde{f}(\vec{v}) = f(\vec{v} + \vec{v}_{earth}), \tag{5}$$

where $\vec{v}_{earth}$ is the velocity of the earth in the galactic rest frame. The simplest model is a three-dimensional Maxwell distribution:

$$f(\vec{v}) \propto e^{-\vec{v}^2/v_0^2}, \tag{6}$$

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

In order to evaluate the integral in (4), we make the conversion to spherical coordinates, and take special care to deal with the velocity boost in (5). Assuming

a $1/v^2$ velocity dependence of the cross section term (see section 4.2), we need to evaluate an integral of the form

$$I = \int_{v_{min}}^{v_{max}} d^3v \frac{f(\vec{v}+\vec{v}_{earth})}{v} = \int_{v_{min}}^{v_{max}} d^3v \frac{1}{v} e^{-(\vec{v}+\vec{v}_{earth})^2/v_0^2} \tag{7}$$

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

$$\begin{aligned}
I &= \int_0^{2\pi} d\phi \int_{v_{min}}^{v_{max}} dv \int_{-1}^{1} d(\cos\theta) e^{-2vv_{earth}\cos\theta/v_0^2} v^2 \frac{1}{v} e^{-(\vec{v}^2+\vec{v}_{earth}^2)/v_0^2} \\
&= 2\pi \int_{v_{min}}^{v_{max}} dv v e^{-(\vec{v}^2+\vec{v}_{earth}^2)/v_0^2} \left(-\frac{v_0^2}{2vv_{earth}} e^{-2vv_{earth}\cos\theta/v_0^2} \right)_{-1}^{1} \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv e^{-(\vec{v}^2+\vec{v}_{earth}^2)/v_0^2} \left(-e^{-2vv_{earth}/v_0^2} + e^{+2vv_{earth}/v_0^2} \right) \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv \left(-e^{(v+v_{earth})^2/v_0^2} + e^{(v-v_{earth})^2/v_0^2} \right) \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv \left(g(v-v_{earth}) - g(v+v_{earth}) \right)
\end{aligned} \tag{8}$$

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

$$g(v) \propto e^{-v^2/v_0^2}. \tag{9}$$

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:

$$\begin{aligned}
&\int_{v_{min}}^{v_{max}} d^3v \sigma(v) e^{-(\vec{v}+\vec{v}_{earth})^2/v_0^2} \\
&= \frac{\pi v_0^2}{v_{earth}} \int_{v_{min}}^{v_{max}} dv \sigma(v) v^2 \left(g(v-v_{earth}) - g(v+v_{earth}) \right).
\end{aligned} \tag{10}$$

The FORTRAN code uses equation (10) to evaluate the event rate integral in equation (4) with quadrature. Analytic solutions of (10) 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.

4.2 Differential cross section

$$\frac{d\sigma(v,E_R)}{dE_R} = 2m_T \frac{d\sigma(v)(v,\vec{q}^2)}{d\vec{q}^2} = 2m_T \frac{1}{4\pi v^2} T(v,q), \tag{11}$$

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 $T(v,q)$.

4.3 Transition probability / Scattering probability

The scattering probability is

$$T(v,q) = \frac{1}{2j_\chi+1}\frac{1}{2j_T+1}\sum_{spins}|\mathcal{M}(v,q)|^2 \tag{12}$$

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

$$T(v,q) = \frac{4\pi}{2j_T+1}\frac{1}{(4m_\chi)^2}\sum_{x=p,n}\sum_{x'=p,n}^{1}\sum_{i=1}^{8} R_i^{xx'}(v^2,q^2)W_i^{xx'}(q) \tag{13}$$

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.

4.4 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.

$$\begin{aligned} R_M^{xx'}(v,q) = \frac{1}{4}cl(j_\chi)\{&[v^2-(q/2\mu_t)^2](c_5^x c_5^{x'}q^2 + c_8^x c_8^{x'}) + c_{11}^x c_{11}^{x'}q^2\} \\ &+\{c_1^x + c_2^x[v^2-(q/2\mu_t)^2]\}\{c_1^{x'} + c_2^{x'}[v^2-(q/2\mu_t)^2]\} \end{aligned} \tag{14}$$

$$\begin{aligned} R_{\Sigma''}^{xx'}(v,q) = \frac{1}{16}cl(j_\chi)\{&c_6^x c_6^{x'}q^4 \\ &+(c_{13}^x c_{13}^{x'}q^2 + c_{12}^x c_{12}^{x'})[v^2-(q/2\mu_T)^2] + 2c_4^x c_6^{x'}q^2 + c_4^x c_4^{x'}\} + \frac{1}{4}c_{10}^x c_{10}^{x'}q^2 \end{aligned} \tag{15}$$

$$R^{xx'}_{\Sigma'}(v,q) = \frac{1}{32} cl(j_\chi)\{2c^x_9 c^{x'}_9 q^2 + (c^x_{15}c^{x'}_{15}q^4 + c^x_{14}c^{x'}_{14}q^2 - 2c^x_{12}c^{x'}_{15}q^2 + c^x_{12}c^{x'}_{12})[v^2 - (q/2\mu_T)^2] + 2c^x_4 c^{x'}_4\} + \frac{1}{8}(c^x_3 c^{x'}_3 q^2 + c^x_7 c^{x'}_7)[v^2 - (q/2\mu_T)^2] \quad (16)$$

$$R^{xx'}_{\Phi''}(v,q) = \frac{q^2}{(4m_N)^2} cl(j_\chi)(c^x_{12} - c^x_{15}q^2)(c^{x'}_{12} - c^{x'}_{15}q^2) + \frac{q^2}{(4m_N)^2} q^2 c^x_3 c^{x'}_3 \quad (17)$$

$$R^{xx'}_{\tilde{\Phi}'}(v,q) = \frac{q^2}{(4m_N)^2} cl(j_\chi)(c^x_{12}c^{x'}_{12}q^2 + c^x_{12}c^{x'}_{12}) \quad (18)$$

$$R^{xx'}_{\Delta}(v,q) = \frac{q^2}{(2m_N)^2} cl(j_\chi)(c^x_5 c^{x'}_5 q^2 + c^x_8 c^{x'}_8) + 2\frac{q^2}{m^2_N} c^x_2 c^{x'}_2 [v^2 - (q/2\mu_T)^2] \quad (19)$$

$$R^{xx'}_{\Delta\Sigma'}(v,q) = \frac{q^2}{(2m_N)^2} cl(j_\chi)(c^x_4 c^{x'}_5 - c^x_8 c^{x'}_9) - \frac{q^2}{m_N} c^x_2 c^{x'}_3 [v^2 - (q/2\mu_T)^2] \quad (20)$$

$$R^{xx'}_{\Phi''M}(v,q) = \frac{q^2}{4m_N} cl(j_\chi) c^x_{11}(c^{x'}_{12} - c^{x'}_{15}q^2) + \frac{q^2}{m_N} c^{x'}_3 \{c^x_1 + c^x_2[v^2 - (q/2\mu_T)^2]\} \quad (21)$$

As a shorthand we have introduced the notation

$$cl(j) = 4j(j+1)/3. \quad (22)$$

4.5 Nuclear response functions

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

$$y = \left(\frac{qb}{2}\right)^2, \quad (23)$$

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

$$b^2 = 41.467/(45A^{-1./3} - 25A^{-2/3})\ fm^2. \quad (24)$$

$$W_{M}^{xx'}(y) = \sum_{even\ J} \langle j_T|\, M_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, M_{Jx'}(y)\, |j_T\rangle \tag{25}$$

$$W_{\Sigma''}^{xx'}(y) = \sum_{odd\ J} \langle j_T|\, \Sigma''_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, \Sigma''_{Jx'}(y)\, |j_T\rangle \tag{26}$$

$$W_{\Sigma'}^{xx'}(y) = \sum_{odd\ J} \langle j_T|\, \Sigma'_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, \Sigma'_{Jx'}(y)\, |j_T\rangle \tag{27}$$

$$W_{\Phi''}^{xx'}(y) = \sum_{even\ J} \langle j_T|\, \Phi''_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, \Phi''_{Jx'}(y)\, |j_T\rangle \tag{28}$$

$$W_{\tilde{\Phi}'}^{xx'}(y) = \sum_{even\ J} \langle j_T|\, \tilde{\Phi}'_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, \tilde{\Phi}'_{Jx'}(y)\, |j_T\rangle \tag{29}$$

$$W_{\Delta}^{xx'}(y) = \sum_{odd\ J} \langle j_T|\, \Delta_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, \Delta_{Jx'}(y)\, |j_T\rangle \tag{30}$$

$$W_{\Delta\Sigma'}^{xx'}(y) = \sum_{odd\ J} \langle j_T|\, \Delta_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, \Sigma'_{Jx'}(y)\, |j_T\rangle \tag{31}$$

$$W_{\Phi''M}^{xx'}(y) = \sum_{even\ J} \langle j_T|\, \Phi''_{Jx}(y)\, |j_T\rangle \, \langle j_T|\, M_{Jx'}(y)\, |j_T\rangle \tag{32}$$

4.6 Nuclear operators and their matrix elements

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

5 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	n	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	$(\frac{41.467}{45A^{-1/3}-25A^{-2/3}})^{1/2}$
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)
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