DMFortFactor

A Fast and Accessible Program for Computing WIMP-Nucleus-Scattering Event-Rates

Oliver Gorton[1], Calvin Johnson, Changfeng Jiao

2021 Fall Meeting of the APS Division of Nuclear Physics
Session QL:
BSM Searches in Fundamental Symmetries X: Dark Matter
October 14 2021

[1]San Diego State University

DMFortFactor: a Fortran code to compute WIMP-nucleus form factors in an EFT framework

- Our code is **hundreds to thousands of times faster** than `DMFormFactor` [2][3], the only published code for computing these nuclear form factors
- Written in Modern Fortran, with a Python wrapper
- Capability to compute multiple relevant quantities

[2] Fitzpatrick, Haxton, Katz, Lubbers, Xu, "The effective field theory of dark matter direct detection", JCAP02(2013)004

[3] Anand, Fitzpatrick, Haxton, "Weakly interacting massive particle-nucleus elastic scattering response", PRC 89, 065501 (2014)

Direct detection of dark matter is an ongoing area of research

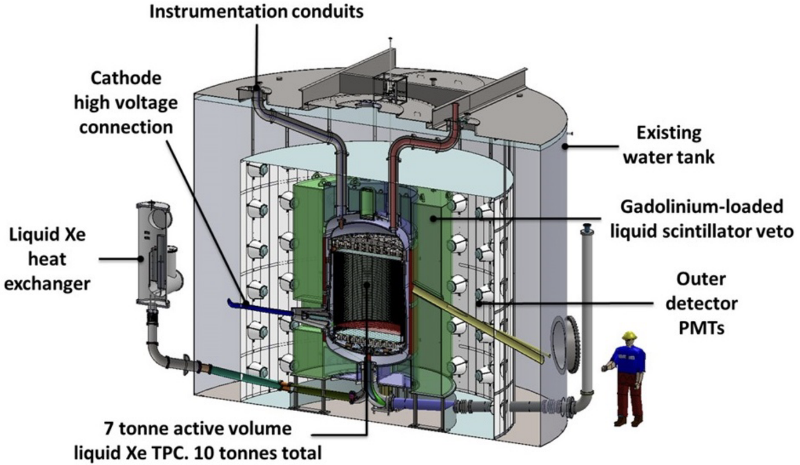


Figure: The LZ detector, which is being built at the Sanford Underground Research Facility and run by the DOE, Lawrence Berkeley Lab [arXiv:1703.09144v1]

Waiting for the right nuclear recoil

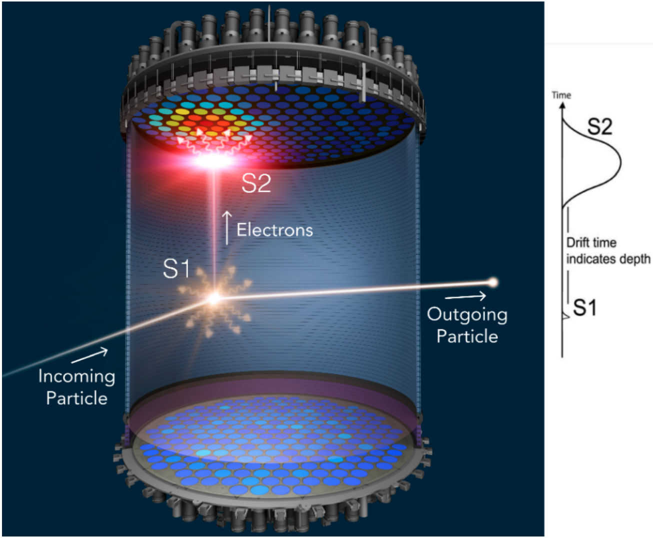


Figure: The LZ detector is waiting for a weak interaction between a WIMP and a Xenon nucleus [arXiv:1703.09144v1]

The observed event-rate sets limits on the theoretical interaction strength

- A theoretical prediction for the event rate comes from Effective Field Theory (EFT)
- Ingredients:
  1. Nuclear wave function (actually, one-body density matrices) in a H.O. basis
  2. An EFT Lagrangian cast as a Hamiltonian

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

  3. A WIMP velocity distribution
- The 15 nucleon couplings $\mathcal{O}_i$ are combinations of

$$i\frac{\vec{q}}{m_N},\ \vec{v}^{\perp},\ \vec{S}_{\chi},\ \vec{S}_N. \tag{2}$$

The event rate (or lack thereof) sets limits on the theoretical interaction strength

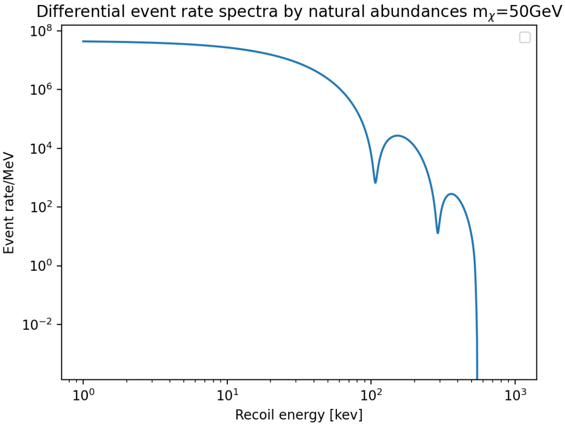


Figure: Predicted direct-dection event rate given 50 GeV WIMPs and simple coupling to protons with $c_1^p = 1$.

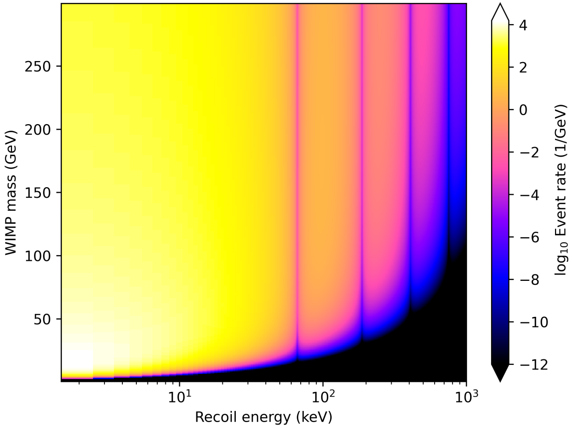
Timing

Runtime (seconds)		
Nonzero coefficients	c_1^n	All
DMFormFactor[4]	20,000	20,000
DMFortFactor (serial)	1.7	100
DMFortFactor (parallel x4)	0.5	30

Table: Program execution time for an event-rate calculation of ^{131}Xe with 1000 recoil energies with $m_\chi = 150$ GeV.

[4] arXiv:1308.6288, arXiv:1203.3542

Timing benchmark: Differential eventrate spectra for 300 possible WIMP masses



Code	Runtime
DMFormFactor	70 days (predicted)
DMFortFactor	10 minutes

[Easy to use] Use the Python wrapper to compute WIMP-nucleus scattering-event rates

```
'''
Use dmfortfactor to compute the differential event
rate spectra for Xe-131 assuming a WIMP mass of 50 GeV
and isovector coupling to operator-3.
'''
import dmfortfactor as dm
control_dict = {"wimpmass" : 50.0}
c_v = np.zeros(15)
c_v[2] = 0.0048 # Set the EFT coupling
Erkev, ER = dm.EventrateSpectra(
              Z = 54,
              N = 77,
              dres = "../dres/xe131gcn",
              controlwords = control_dict,
              cvvec = c_v)
```

[Easy to export] Generation of tabulated nuclear form factors for codes like WimPyDD

$$W_i^{x,x'}(q) = \sum_J \sum_{a,b} \rho_J^x(a,b)\rho_J^{x'}(a,b) \langle a||\mathcal{W}_J^{(f),x}||b\rangle \langle a||\mathcal{W}_J^{(f),x'}||b\rangle$$

Columnated plain text for `m` momentum values q

```
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
```

[Easy to import] Python function for nuclear form factors for codes like WimPyDD (arXiv:2106.06207)

```python
'''
Use dmfortfactor to generate form factors for Xe-131
and return a Python function for later use
'''
cwords = {
        "wimpmass" : 150.0,
        "usemomentum": 1}
Wfunc = dm.NucFormFactor(
        Z = 54,
        N = 77,
        dres = "../dres/xe131gcn",
        controlwords = cwords,
        ermin = 0.001,
        ermax = 10.0,
        erstep = 0.001)
```

[Easy to import] Python function for nuclear form factors for codes like WimPyDD

A function of momentum transfer q, and returns the $(8, 2, 2)$ array of form factors needed for WimPyDD

```
[python console] >>> Wfunc(q=0.001)
```

```
[[[ 1.62248e+03  2.78451e-02]
  [ 5.92199e-03 -3.54404e+02]]
 [[-2.61310e+02 -2.61411e-02]
  [-8.93270e-03  5.70787e+01]]
 [[-2.61310e+02 -2.61411e-02]
  [-8.93270e-03  7.84651e+01]]
 [[ 4.20853e+01  2.45414e-02]
  [ 1.34741e-02 -1.26372e+01]]
 [[ 7.74174e+01  1.39266e-02]
  [ 7.00767e-02 -4.41734e-02]]
 [[-1.71453e+01 -1.30745e-02]
  [-6.05964e-02  4.14702e-02]]
 [[-1.71453e+01 -1.30745e-02]
  [-6.05964e-02  3.81975e-02]]
 [[ 3.80365e+00  1.22745e-02]
  [ 5.23987e-02 -3.58599e-02]]]
```

DMFortFactor: a Fortran code to compute WIMP-nucleus form factors in an EFT framework

- Our code is **hundreds to thousands of times faster** than `DMFormFactor`
- Written in Modern Fortran, with a Python wrapper
- **Publication in preparation for** *Computer Physics Communications*

This work was supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics, under Award Number DE-SC0019465.

We thank Jonathan Nikoleyczik, Shaun Alsum, Kim Palladino, A. B. Balantekin and S. N. Coppersmith, and we acknowledge the pioneering Mathematica script of Anand, Fitzsimmons, and Haxton, which made the job of validating our code much easier.