


A C++ program for estimating detector sensitivities to long-lived particles: **displaced decay counter**

Zeren Simon Wang 王泽人 (合肥工业大学 HFUT)

[arXiv:2308.07371](https://arxiv.org/abs/2308.07371) (*Eur. Phys. J. C* 84, 642 (2024))

 GitHub repository of [DDC](#)

in collaboration with F. Domingo, J. Günther, and J. S. Kim

July 17th, 2025

第六届粒子物理前沿研讨会

中国吉林长春



合肥工业大学
HEFEI UNIVERSITY OF TECHNOLOGY

Motivation

- No concrete evidence so far at the LHC for heavy resonances
- Increasingly more attention given to other signatures e.g. long-lived particles (LLPs)
- LLP: well motivated, predicted in various BSM scenarios
- ATLAS, CMS, etc. performed multiple searches
- LHC far detectors proposed: FASER(2), MATHUSLA, CODEX-b, ANUBIS, MoEDAL-MAPP1(2), etc.
- Need a unified platform for sensitivity estimation at all these experimental setups

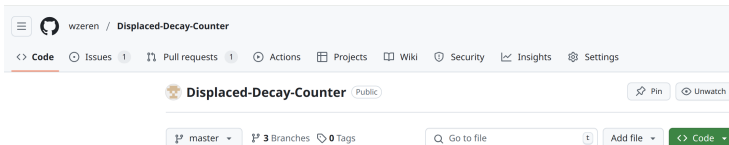
⇒ Displaced Decay Counter

What is DDC?

- **Main functionality:** compute the decay probability of an LLP within a pre-defined detector volume
- Some **LHC detector models pre-encoded**; **an editor for further constructions**
- Relies on **MC-simulation tools**, but not pre-generated tabulated spectra
- **Versatility:** different experimental facilities, MC-simulation tools, theoretical models, interaction types, personalized detector models

Getting started

- Source code downloaded from <https://github.com/wzeren/Displaced-Decay-Counter>



- Prerequisites: **Pythia8** and **HepMC2**; a **C++17** compiler
- Adjust the paths in DDC/Makefile to the local setup and **make -jN**

```
Terminal
#Home Directory
HOME = ...../Research/HEP_Software

#Pythia Directories
PYTHIA8=$(HOME)/pythia8313
PYTHIALIB=$(PYTHIA8)/lib
PYTHIAINC=$(PYTHIA8)/include
PYTHIAINCPYTHIA8=$(PYTHIA8)/include/Pythia8

#HepMC Directories
HEPMC=$(HOME)/hepmc2.06.11-install
HEPMCLIB=$(HEPMC)/lib
HEPMCINC=$(HEPMC)/include
```

General features

- Basic input:
 - PID, mass, lifetime, and 'visible' BR of the LLP
 - MC events in the LHEF or HepMC format, or Pythia8 run card; A total cross section normalizing the set of events
 - Names of the detectors that the user wants to simulate, with their corresponding integrated luminosities
- DDC computes decay probability of the LLPs identified by their PID inside the specified detectors, and then the signal event number
- Detector efficiencies not yet taken into account in the current version
 - the corresponding information is generally not available yet for far-detector experiments

Operation and output

- `cd bin/`
`./main inputEvents.dat inputLLPs.dat results.txt`
- Results saved in `bin/results.txt` and logs (availability of the detectors and summary of the input information) in `bin/Logs/`
- Results include the simulated acceptance and number of signal events for each detector

```
*****  
***** WELCOME TO THE RESULT FILE *****  
*****  
  
Number of visible LLP events in the detectors.  
Detector: simulated acceptance, number of events:  
*****  
AL3X, LLP0: 0.0333148 , 31.7906  
ANUBIS0, LLP0: 0.000384867 , 4.40712  
ANUBIS1, LLP0: 0.000308149 , 3.52861  
CODEXB0, LLP0: 6.88927e-05 , 0.078889  
CODEXB1, LLP0: 7.00049e-05 , 0.0801626  
FACET, LLP0: 0.000392674 , 4.49651  
FASER, LLP0: 1.41389e-07 , 8.09524e-05  
FASER2, LLP0: 2.78528e-05 , 0.318942  
MAPP1, LLP0: 0.000175794 , 0.0201302  
MAPP2, LLP0: 0.000417777 , 0.478396  
MATHUSLA0, LLP0: 0.000232753 , 2.66526  
MATHUSLA1, LLP0: 0.000179199 , 2.05201  
MATHUSLA2, LLP0: 0.000185024 , 2.11871
```

Sample result file

Code structure

① Read and interpret the input files

- Using the `inputInterface` class
- A class object constructed from the paths to the input files
- Input extracted and stored using the `json` interface

② Encode the properties of detector models

- Characteristics of the detector models stored in objects of the `Detector` class
- Central property of these detector models rests with their class function `computing a probability of decay within the detector`

③ Analyze the MC events against the former

- Analysis of MC events processed within the `analysis` class
- Object created in the main routine from the stored input
- The event file is read (or produced with Pythia)
- LLPs searched for by their PDG code
- LLPs checked against a list of detector models
- Outcome copied in the output file

Doxygen documentation:

<https://wzeren.github.io/Displaced-Decay-Counter>

Modeling the decay probability

- Probability that the particle decays in a detector that comes across its trajectory between distances ℓ_1 and ℓ_2 ($\ell_1 < \ell_2$):

$$P_{\text{lab}}^{\text{dec}}(\ell_1, \ell_2) = \exp\left[-\frac{\ell_1}{\gamma\|\vec{v}\|\tau}\right] - \exp\left[-\frac{\ell_2}{\gamma\|\vec{v}\|\tau}\right]$$

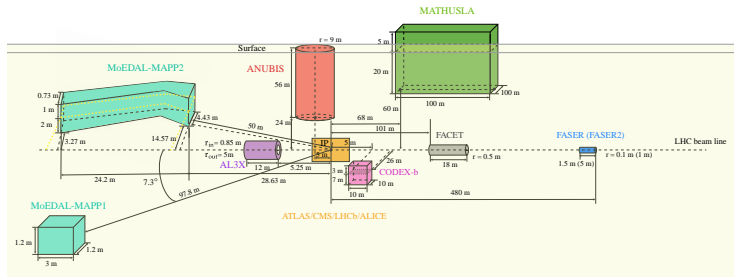
⇒ Geometrical exercise of **determining the distances at which the LLP trajectory intersects the outer layers of the detector**

- **Additivity**: given a volume V with $V = \sqcup_i V_i$,
 $\mathcal{P}(\text{decay in } V) = \sum_i \mathcal{P}(\text{decay in } V_i)$

Underlying hypotheses:

- ① LLP produced at the IP
- ② LLP flies in a straight line from the IP
- ③ LLP not interacting with the detector components

LHC far detectors



reproduced from [2410.19561] (C.-T. Lu, X. Wang, X. Wei, Y. Wu)

Transverse: ANUBIS, CODEX-b, MoEDAL-MAPP1(2), MATHUSLA

Forward: FACET, FASER(2)

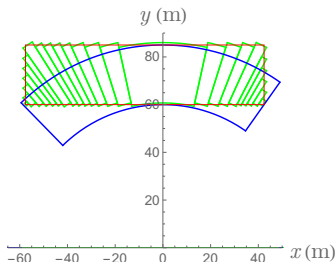
- Mostly **background-free**; assume that 3 signal events correspond to exclusion bounds at 95% C.L.

Defining detectors in cylindrical geometry

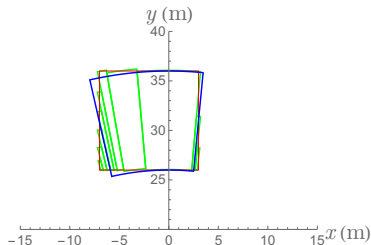
- Detectors designed in a fashion exploiting the **cylindrical symmetry** (of LLP production) around the beam axis, to **minimize the time cost** of event simulation
- LLPs emitted at **any azimuthal angle used** in the calculation of the acceptance
- Saves a factor $\approx 2\pi/\delta\varphi$ in terms of event generation, as compared to a strict three-dimensional modelization

Built-in detectors

- Multiple detector layers in the (x, y) plane



MATHUSLA (red) vs. MATHUSLA0 (blue) vs. MATHUSLA1 (green)



CODEX-b (red) vs. CODEXB0 (blue) vs. CODEXB1 (green)

- FASER(2)**: cylindrical detectors aligned with the beam axis in the far-forward region. Respects the cylindrical symmetry \Rightarrow no difficulty in modeling these detectors with a single detector layer
- Others include ANUBIS, MAPP1(2), and FACET

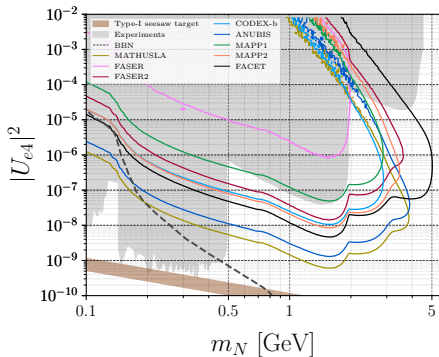
Implementing new detectors

- Possible to **implement new detectors**: studying new geometries or improving existing ones
- Run **make DetEditor** in the main folder \Rightarrow DetEditor executable created in DDC/src
- The only needed **input** is the **name** of the new detector
- DetEditor **prepares a skeleton code** and link it to the rest of the program
- The barebone script is in DDC/src/Detectors

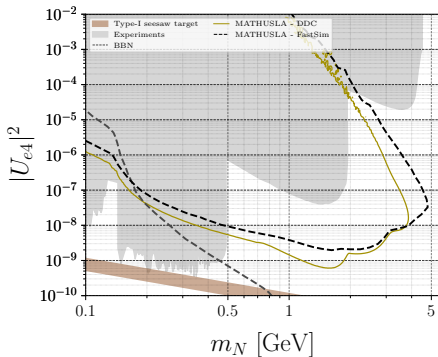
Heavy neutral lepton in the minimal scenario

- Heavy neutral leptons, SM singlet; One N and it mixes with ν_e

$$\mathcal{L} = \frac{g}{\sqrt{2}} \sum_{\alpha,j} U_{\alpha j} \bar{\ell}_{\alpha} \gamma^{\mu} P_L N_j W_{L\mu}^{-} + \frac{g}{2 \cos \theta_W} \sum_{\alpha,i,j} V_{\alpha i}^L U_{\alpha j}^* \bar{N}_j \gamma^{\mu} P_L \nu_i Z_{\mu}$$



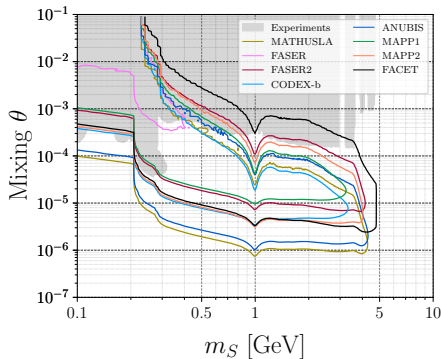
Sensitivity reach at 95% C.L.



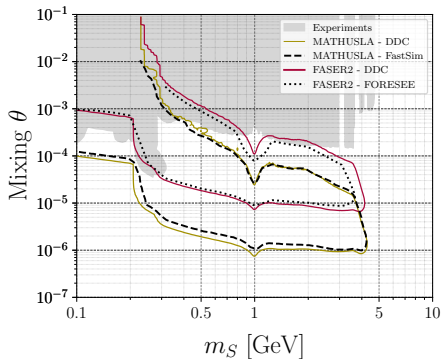
DDC vs. FastSim for MATHUSLA. The FastSim results are extracted from [\[2308.05860\]](#) General agreement

Long-lived complex scalar mixed with the SM Higgs boson and produced from B -meson decays

- Focus on $B \rightarrow K + S$, ignore trilinear scalar couplings
- Visible final states: $S \rightarrow e^-e^+, \mu^-\mu^+, \pi\pi, KK, gg$, and ss



Sensitivity reach at 95% C.L.



DDC vs. FastSim or FORESEE. FastSim results from [\[2308.05860\]](#) and FORESEE results from [GitHub](#) General agreement

Summary

- Displaced Decay Counter (DDC): a C++ program for calculating detector acceptances and signal rates, relying on MC-simulation tools
- Input: MC events in LHEF and HepMC format; or a Pythia8 run card for internal event generation
- Pre-defined far-detector proposals
- Provides the user with an editor for the implementation of further far-detector models
- Presented benchmark scenarios to illustrate the usage of our tool:
 - Validation of our results with existing ones
 - Validation between DDC and other similar tools
- Not restricted to applications in LHC experiments
- Highly versatile, offering modeling capabilities beyond those demonstrated: high potential for studying LLP models at colliders

Thank You! 谢谢!

Back-up slides

Input

Input files in **json** format:

- DDC/bin/inputEvents.dat:

```
{"input":{"input_file_format":"...",  
"input_file_path":"...", "nMC":..., "sigma":...}}
```
- DDC/bin/inputLLPs.dat:

```
{"LLP":{"LLPPID":..., "ctau":...,  
"mass":..., "visibleBR":...}}
```
- DDC/bin/detectors.dat:

```
{"detector_name":[switch,int_luminosity]}
```

More on the cylindrical definition of detectors

A set of basic objects defined in the program to facilitate the implementation of detectors in cylindrical geometry:

① Oriented cylindrical segments (class CylSeg):

- `(std::array<double,2> AA=zA,hA;)`
- A sign representing its orientation (incoming or outgoing): $\varepsilon_O = \pm 1$
- $\varepsilon_O \exp \left[-\frac{\ell}{\gamma \|\vec{v}\| \tau} \right]$

② Cylindrical detector layers (class CylDetLayer):

- A list of oriented cylindrical segments (`std::vector<CylSeg>` SegList) with a weighing factor \mathcal{W} (double)
- Sum of decay probabilities for the segments: $\mathcal{W} \cdot \sum_i \varepsilon_O \exp \left[-\frac{\ell_i}{\gamma \|\vec{v}\| \tau} \right]$
- Weight associated with the detector layer $\mathcal{W}_\varphi = \delta\varphi/2\pi$
- Constructed from either **segments**, **coordinates and weights**, or **'bricks'**

③ Detectors (class Detector):

- Combines an **identifier** (`std::string`), an **integrated luminosity** (double), and a **list of cylindrical detector layers** (`std::vector<CylDet Layer>`)

Similar tools

| Name | Code & Tool | Process Types | Models |
|----------|-------------------------|-----------------------|----------|
| MadDump | plugin for MG5, Python | LLP decays/scattering | Many |
| FORESEE | Python | LLP decays | Many |
| ALPINIST | Mathematica–ROOT–Python | LLP decays | ALP only |
| SensCalc | Mathematica | LLP decays | Many |
| FastSim | Python | LLP decays | Many |
| DDC | C++ | LLP decays | Many |

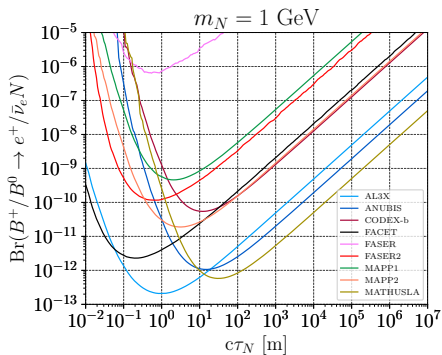
| Name | Experiments | Recon. and Effi. | Reference |
|----------|-----------------------|--|---------------------------------|
| MadDump | Beam-dump exp. mainly | ✗ | JHEP05(2019)028 |
| FORESEE | Far detectors | Cut functions ✓, decay-product effi. ✗ | Phys. Rev. D 104, 035012 (2021) |
| ALPINIST | Extracted-beam exp. | Fully reconstructed final states ✓ | JHEP07(2022)094 |
| SensCalc | Far detectors | ✗ | Phys. Rev. D 108, 075028 (2023) |
| FastSim | Far detectors | Recons. criteria for DVs ✓ | Phys. Rev. D 109, 075017 (2024) |
| DDC | Far detectors | Cut functions ✓ | Eur. Phys. J. C (2024) 84:642 |

Implementing cuts

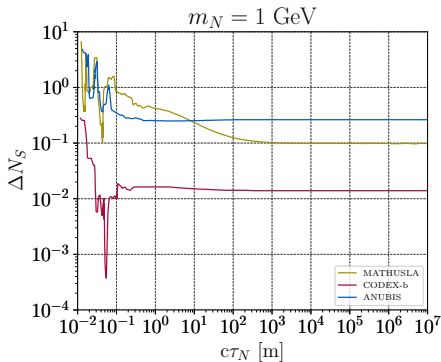
- Possible to **implement cuts** on events containing LLPs
- **A cut function is attached to each detector** in the DDC/src/Detectors folder
- **By default trivial** but can be edited
- The input of this function is a collider event in HepMC format, leaving a maximal flexibility on the possible types of conditions
- **To be filled up in a later stage** when more experimental detail becomes available

Light neutral fermion N produced in bottom decays

- Consider $B^+ \rightarrow e^+ N$ or $B^0 \rightarrow \bar{\nu}_e N$
- N can be a sterile neutrino, a light bino in the RPV-SUSY, ...



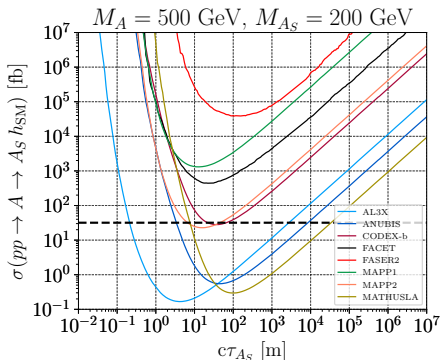
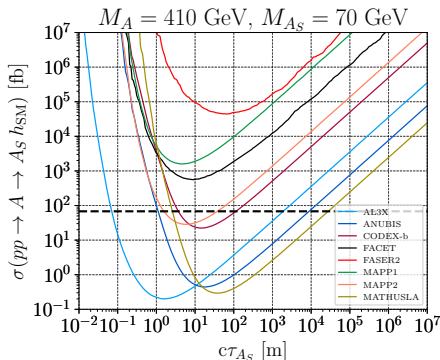
Sensitivity reach at 95% C.L.



$\Delta N_S = |N_S^1 - N_S^0|/N_S^0$
simple approx. vs. sophisticated implem.

Long-lived scalar in an extended Higgs sector

- New scalar states: H , A , H^\pm at about 500 GeV, and a lighter, mostly singlet CP-odd scalar A_S with mass in the 100 GeV range
- $pp \rightarrow A \rightarrow A_S h_{\text{SM}}$, A_S long-lived
- Configuration discussed in [[2203.05049](#)]



Long-lived fermion from a pair of heavy resonances

- $pp \xrightarrow{\text{EW}} \tilde{e}_R \tilde{e}_R$, ($\tilde{e}_R \rightarrow e\tilde{\chi}_1^0$, $\tilde{e}_R \rightarrow e\tilde{\chi}_1^0$)
- A long-lived light bino $\tilde{\chi}_1^0$ decays via RPV couplings

