SensCalc: user guide

Maksym Ovchynnikov

E-mail: o.maxim@gmail.com

Contents

Contents

1 General overview and installation

The code `SensCalc` consists of a few `Mathematica` notebooks that compute the number of events for various LLPs.[1] Four notebooks have to be run sequentially: `Acceptances.nb`, `LLP distribution.nb`, `LLP sensitivity.nb`, and `Plots.nb`, see Fig. 1.

On top of that, the recent update has added the module called `EventCalc`, which generates the decay events at various in a way similar to the traditional Monte Carlo generators. It uses the input produced by the second notebook of `SensCalc`, `LLP distribution.nb`, as well as the routines utilized by the other notebooks.

1.1 Why `Mathematica`?

Evaluating sensitivities of lifetime frontier experiments to LLPs is often much simpler than detailed event analysis or background evaluation. If computing the distributions of mother particles producing the LLPs externally, the sensitivity evaluation is split into relatively trivial computations, such as calculating the LLP distribution function and evaluating the acceptances for the LLP and its decay products.

[1] Contact the author: o.maxim@gmail.com

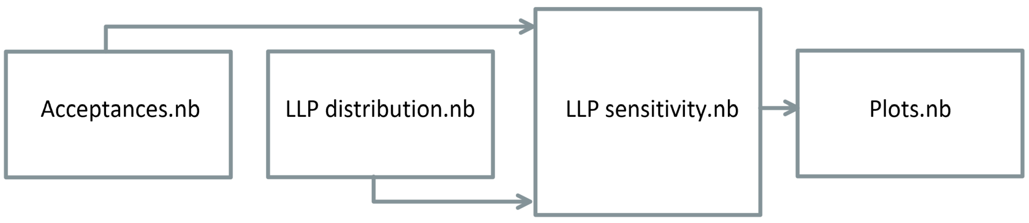

Figure 1. Description of the modular structure of `SensCalc`. The notebook `Acceptances.nb` produces the list of acceptances ϵ_{az} and ϵ_{dec} entering Eq. (2.1) of the accompanying paper for the selected experiment. The notebook `LLP distribution.nb` computes the distribution of LLPs $f(m,\theta,E)$ at the facility housing the experiment. The notebook `LLP sensitivity.nb` uses the input of the two previous notebooks to calculate the tabulated number of events, and calculates the sensitivity in the mass-coupling plane as a function of the remaining parameters, such as the minimal number of events and any additional model-specific parameters. Finally, `Plots.nb` produces the sensitivity plots from the output of the previous notebook.

`Mathematica` is a great environment for such tasks. Namely, everything from computing the matrix elements of the LLP production and decay to working with the geometry of the experiment may be made inside it. This way, it is very suitable for people who are not familiar with special software used to compute sensitivities. In addition, `Mathematica` notebooks provide a convenient visual framework. Finally, there is a very friendly and experienced `Mathematica` community whose members may help with any question (and probably optimize the performance of `SensCalc` by another factor of ten).

Pure `Mathematica` does not suit well for performing some low-level operations such as producing the phase space of many-body decays or propagating the particles through space. Fortunately, to improve this part, the `Mathematica` code has been simply compiled into a native machine code, which may improve the performance by a factor of a hundred and more.

1.2 Installation

The user needs to have installed `Wolfram Mathematica`, version 13.2 or higher. In addition, the user must install the `Mathematica` package `FeynCalc` and a `C` compiler recognized by `Mathematica`.[2] There are no other dependencies. The user can check whether the dependencies are properly installed by launching the notebook `0. Check for the needed soft.nb`.

1.3 Launching guidance

For the LLPs and experiments that are already implemented, the user just needs to launch the section *Launch this section to run the notebook* located at the top of each notebook. The relevant sections will then be launched automatically, and the notebook will prompt the user to specify the required inputs via dialog windows. If, however, users want to modify the model, geometry, or assumptions, they may change the code and inputs as described in the following sections.

1.4 Performance

Performance significantly depends on CPU and LLP. For most of the scenarios, the sensitivity may be computed from scratch – computing the tabulated acceptances, distribution, number of events, and the sensitivity – within less than an hour. For instance, Apple's new CPUs typically work much faster than Intel's CPUs. Also, if being launched once for an experiment belonging to the given facility, the output of the notebook `LLP distribution.nb` may be used for any other experiment

[2]For Windows machines, a suitable choice is `Visual studio community`, where the package "Desktop development with C++" should be installed.

from the same facility, which reduces timing substantially. Performance may be improved in the future.

2 List of implemented LLPs

The currently implemented LLPs are:

- The HNLs that are mixing with $\nu_e/\nu_\mu/\nu_\tau$. The names of these LLPs throughout the notebooks are are `HNL-mixing-e`, `HNL-mixing-mu`, `HNL-mixing-tau`, or, collectively, `HNL`.
- Dark photons (`DP`).
- ALPs coupled to gluons, fermions, or photons (`ALP-gluon`, `ALP-fermion`, `ALP-photon`).
- Dark scalars with the mixing and trilinear couplings (`Scalar`).
- Mediators coupled to the anomaly-free combinations of the baryon and lepton currents: $U_{\text{B-L}}(1)$, $U_{B-3L_\mu}(1)$, $U_{B-3L_e-L_\mu+L_\tau}(1)$, and $U_{B-L_e-3L_\mu+L_\tau}(1)$ (`B-L`, `B-3Lmu`, `B-3Le-Lmu+Ltau`, `B-Le-3Lmu+Ltau`).
- Millicharged particles (in public – only calculation of the production flux) (`MCP`).

For the definitions of the models and references on the implemented phenomenology, look at the accompanying paper.

In the future, I plan to add other models, such as inelastic light dark matter and dipole portal of HNLs. Their implementations already exist in private.

3 1. Acceptances.nb

The output of the first notebook is data that includes:

1. A table with the following columns:
$$\{m, \theta, E, z, \epsilon_{\text{az}}, \epsilon_{\text{dec}}\} \tag{3.1}$$
where $\epsilon_{\text{az}}(\theta, z)$ is the fraction of the azimuthal angle covered by the decay volume, and $\epsilon_{\text{dec}}(m, \theta, E, z)$ is the acceptance for the decay products of the LLPs that decayed inside the decay volume.
2. The experiment- or facility-specific details: at which facility the experiment is located, whether the experiment has a calorimeter and a dipole magnet, what is the number of proton-proton or proton-target collisions, what are the production fractions of various SM particles such as mesons or heavy SM bosons, and the total branching ratio of the processes visible at the experiment.

The data is located in the folder
Acceptances/⟨Given experiment⟩
Depending on the specific LLP, experiment, OS, and CPU, the notebook typically runs for $\mathcal{O}(5)$ minutes per LLP. The runtime also significantly depends on the grid density and the number of simulated decays; the latter is parameterized by the number of ϕ values considered when evaluating the decay products acceptance (`NofPhiVals`), and by the number of decays simulated for each (m, E, θ, z) point (`ivaltest`).

Below is a description of how the notebook works, assuming that all the required ingredients are provided (the requirements are given in the subsection below):

1. First, users specify the experiment for which they want to compute the acceptances (section `Choosing the experiment`), the list of the particles detectable at the experiment, and selection criteria for the decay products (section *Specifying cuts on the decay products*).

2. Next, the notebook computes the tabulated azimuthal acceptance $\epsilon_{\rm az}(\theta, z)$ (cf. *Azimuthal acceptance calculation*). It starts by evaluating the polar angle coverage of the decay volume and the detector, which is done automatically by discretizing the implemented geometry of the experiment. To ensure the correctness of the implementation of the experiment setup, the user should check the values of the parameters

   `ThetaDecVolGivenExperimentMin`,

   `ThetaDecVolGivenExperimentMax`,

   `ThetaDetGivenExperimentMin`,

   `ThetaDetGivenExperimentMax`,

   and the visualization of the decay volume and detector in section *Visualization of the geometry.* Afterward, the notebook computes the grid of polar angles θ and z within the decay volume's coverage. Next, for each set of (θ, z), the list of values of the azimuthal angle ϕ that belongs to the decay volume acceptance is generated.[3] Once the computation of the azimuthal acceptance has finished, the user should check that the volume of the decay volume matches the value obtained using the computed acceptance $\epsilon_{\rm az}$ and Eq. (5) from the accompanying paper (cf. section *Grid with azimuthal acceptance*). Another cross-check is the visualization of the generated grid of the LLP's coordinates inside the decay volume, separately for the LLPs that point or do not point to the detector, and checking that the volume integral of the azimuthal acceptance matches with the total volume of the decay volume. These tests, together with displaying the E, θ, and z grids, are made in section *Azimuthal acceptance calculations.*

2. To compute the decay products acceptance, one must first simulate LLP decays at the rest frame of the decaying LLP. Consider a particle `particle` (which may be either a LLP or a SM particle) with mass `mparticle`. The phase space for the particular decay channel `process` is generated on-flight by the block

   `PhaseSpaceDecaysRest[particle,mparticle,process,Nevents]`

   defined in the notebook *llp-decays.nb* located at the folder *codes* and evaluated externally from the main notebook. Here, `Nevents` is the number of decay events to generate. If some of the LLP's decay products are unstable, it repeats the routine

   `PhaseSpaceStableFromUnstableBlock`

   which decays it until only (meta)stable products are left: $\gamma, \pi^{\pm}, K^{\pm}, K_L^0, e^{\pm}, \mu^{\pm}, \nu, p, n$. The phase space is organized in a table, with each row representing a particular decay. Every 8 elements of the given row represent a particular decay product, with the following meanings:

   `p_x p_y p_z Energy mass pdg-identifier electric-charge stability`

   where all dimensional units are in GeV, `pdg-identifier` is the PDG ID in the form `ID.` (e.g., `22.`, note the presence of '.'!), electric charge is in units of the proton's charge, and `stability` must be `1.` (which defines metastable particles from the list above). Decays into partons (such as LLP $\to u\bar{u}, c\bar{c}$, etc.) are treated by this routine as decays into a pair of the lightest charged mesons containing the given parton.

[3] Suppose the decay volume is not itself the detector. In that case, the notebook computes two grids of ϕ: one for when the LLP decays inside the decay volume and simultaneously points to the end of the detector (`EpsilonAzPhiListToDet`), and another one for when the LLP decays inside but does not point to it (`EpsilonAzPhiListNotToDet`).

Alternatively, if the given decay is into partons, instead of generating on-flight, it may be possible to use the phase space pre-generated in `MadGraph5+pythia8` for several characteristic masses `mLLP`. In this case, the partons are replaced with a bunch of hadrons appearing because of the showering and hadronization. The hadronized phase space is called by the block

`PhaseSpaceDecaysRestJets[LLP, process, mLLP]`

located in the notebook `llp-decays.nb`. The UFO files used to generate the phase space are located in the folder *UFO files*. The phase space is stored in the folder

simulated phase space

The names of the files must have the pattern

`<LLP>_<process>_mass.m`

where `<LLP>`, `<process>` exactly matches the names of the LLP and the given decay process into the partons as defined in the notebook. An example of the filename is

`Scalar_Jets-cc_3.8.m`,

which contains the phase space of the decay of the Higgs-like scalar with mass 3.8 GeV into hadrons resulting from the jets $c\bar{c}$. The structure of the phase space must be the same as the one generated by `PhaseSpaceDecaysRest`. To have a rectangular array where all the rows have the same length, every row is supplemented with `-999` if needed.

4. When computing the decay products acceptance (cf. *Decay acceptance calculation*), the notebook simulates the phase space of the relevant decay channels and calculates the decay acceptance for several signatures. All the definitions are loaded from an external notebook `decay-acceptance.nb`. In particular, for the given LLP mass, the tabulated decay acceptance is calculated by the block

   `FinalBlockMass[LLP, mLLP, proclist, isim, HadronizationOption,`

   `AllProductsWithinAcceptance]`

   where: `proclist` is the list of the processes used to compute the decay products acceptance (the users are asked about the list of these processes right before the computation); `isim` is the number of decays simulated per the ALP mass mLLP, longitudinal displacement from the production point zLLP, the polar angle thetaLLP, the azimuthal angle phiLLP, and the energy ELLP (the optimal value is around 1000); `HadronizationOption` may be `True` of `False` depending on whether the user wants to hadronize partonic decays or not;[4] and `AllProductsWithinAcceptance` is either `>=2` or `All detectable`, depending on the number of particles that must satisfy the decay acceptance criteria – correspondingly, either at least two particles with the opposite charges, or all detectable particles with the energy above the detection threshold (this option would be relevant for many-body decays when it is necessary to reconstruct as many particles as possible in order to identify the LLP).

   The block first evaluates the purely geometric part of the decay acceptance by requiring the projection of the trajectory on the final plane of the detector to be within the detector cross-section (the routine `conditionDecayAcceptanceGeometric`). If a dipole magnet is placed somewhere, then "kicks" to the trajectory, and the momentum components of the charged decay product in the direction transverse to the magnetic field are applied at the end of the magnet. Then, if the decay products are within the geometric acceptance, it calculates the acceptance for the other cuts, such as the energy cut, etc. (see the subsection *Blocks*

[4]Importantly, for the decay into jets, if `HadronizationOption` is `True`, one may choose only the masses `mLLP` for which `PhaseSpaceDecaysRestJets` contains the data. The notebook automatically computes such a mass list.

computing various cuts from the external notebook), which serves as the event pre-selection. The experimental resolutions, such as Gaussian smearings of the energy and direction of motion, may be easily incorporated.

Overall, the calculation of all the cuts is done with the help of the routines `DecayAcceptanceComp`, `DecayAcceptancenProductsComp` or `DecayAcceptanceAllProductsComp`, depending on the type of decay product acceptance.[5] In the notebook, they are launched depending on the choice of `acceptancetypechoice`. The first routine, `DecayAcceptanceComp` (the option `>=2` of `acceptancetypechoice`), calculates the acceptance requiring *at least* two particles to satisfy all the acceptance criteria; for each pair, various cuts are applied, such as the invariant mass cut, the impact parameter cut, etc. This choice is typical among the community of Lifetime Frontier experiments; it shows the optimistic estimate for the event rate. The second one (`>=n`) requires *at least n* products to pass all the cuts; it may be useful when studying many-body decays in the presence of background: multi-particle signatures are much simpler to discriminate. So far, only cuts on separate particles are applied (but the other cuts may be easily added by the user). Finally, the last one (`All detectable`) requires all products with the energy above some threshold to satisfy the cuts; it is useful when studying so-called fully reconstructible events, when the event information may be enough to reconstruct the LLP's mass.

Within the routine, the decay products acceptance is first computed for the values of ϕ for which the LLP is inside the decay volume, and then averaged over ϕ, with each value being weighted by the corresponding azimuthal acceptance. Currently, `SensCalc` may generate on-flight 2-body, 3-body, and 4-body decays of LLPs (of course, if they contain unstable particles, they are decayed recursively). However, generic n-body decays may be implemented completely similarly to the 4-body decays.

3.1 What users have to provide

If the user wants to implement a new LLP, they need to provide the following:

1. The interpolated branching ratios of the decay processes

   `BrRatiosList[LLP, mLLP, process]`

   where `process` is the given decay process, `mLLP` is the LLP mass in GeV, and the associated list of the decay products decay channels `ListDecayProducts[LLP, process]` in the form {`product1, product2, product3, product4`}. If the processes are 2- or 3-body, the remaining products must be replaced with `"Null"`. The list of recognizable products is given in section *Properties of SM particles.* For instance, for a 3-body decay into $\pi^+\pi^-\gamma$, the list of decay products is {π^+,π^-,γ,`"Null"`}.

   For the implemented models, the branching ratios are stored in the folder

   phenomenology/LLP/decay widths

   and they are imported in the notebook via the external notebook *llp-decays.nb.*

2. The squared matrix elements of the decay at the rest frame of the decaying particle. They are needed only for n-body decays with $n \geq 3$. Currently, `SensCalc` may only include it for 3-body decays; for decays with higher multiplicities, a unit matrix element is assumed.[6] For 3-body decays, the squared matrix elements must depend on the decaying LLP mass `m`, the energy of the 1st particle `E1`, the energy of the 3rd particle `E3`. The user may enter the matrix

[5] They may be found in the folder *codes/decay-acceptance.nb*

[6] To be implemented in the future.

element and calculate its square using `FeynCalc`. The list of all squared matrix elements is given by

```
Msquared3BodyLLP[LLP, process, E1, E3, mLLP]
```

For the implemented LLP models, the matrix elements are defined in the section *Decays of various LLPs: processes, branching ratios, matrix elements.*

For the computation of the azimuthal and decay acceptances, the user has to provide:

1. The geometry and dimensions of the decay volume and detector, as well as the properties of the detector. For the detector equipment, the information that currently needs to be provided is the presence of the ECAL and of the dipole magnet, their positions, and the mean magnetic field of the magnet. This is done in the section *Geometry of different experiments and relevant cross-sections*, which loads the external notebook `experiments.nb`.

   – If the decay volume and detector have a simple shape – box, cylinder, or annular cylinder, the only required inputs are the parameters describing the dimensions; they are listed in the subsection *Geometry* belonging to the subsection of the relevant experiment. An example of such parameters is `zToDecayVolumeExperiment[<Experiment>]`

     which specifies the longitudinal distance from the collision point to the beginning of the decay volume. If the detector has a more complicated shape, the user should implement its geometry in the section *Full geometry of the decay volume and detector*. Examples of already implemented non-standard geometries include the SHiP experiment and LHCb.[7] Regardless of how they were implemented, all the geometries are finally added to unified lists. For instance,

     ```
     DecayVolumeGeometry[experiment]
     ```

     contains the full geometry of the decay volume.

2. The integrated luminosity (for colliders) or the number of protons on target (for beam dump experiments) and the relevant production cross-section for the Standard Model particles (cf. subsection *Cross-sections* of the external notebook).

3. Conditions for the LLP to decay inside the decay volume and for the decay product to point/not point to the detector. These are

   ```
   IfLLPinsideDecVol[zLLP, xLLP, yLLP, experiment]
   ```

   ```
   IfLLPtoDet[x1LLPproj, x2LLPproj, experiment]
   ```

   ```
   IfLLPnotToDet[x1LLPproj, x2LLPproj, experiment]
   ```

   where `zLLP,xLLP,yLLP` are the coordinates of the decaying LLP (in meters), and `x1LLPproj`, `x2LLPproj` are the projections of particle's (LLP or its decay product) trajectory given its production/decay point and the presence/absence of the dipole magnet. `x1LLPproj` coincides with the x coordinate, while `x2LLPproj` may be the y coordinate (if the detector plane is transverse to the proton beam axis, as, e.g., for the SHiP experiment) or the z coordinate (in the other cases).

   For the experiments with a simple shape of the decay volume and detector (e.g., box, cylinder, annular cylinder), these conditions may be calculated automatically by using the routines

   ```
   IfLLPinsideDecVolSimple[experiment]
   ```

[7] Another example is the geometry of the ANUBIS experiment in the ceiling configuration; it is more complicated and hence stored in a different notebook `1. Acceptances. ANUBIS-ceiling`.

`IfParticlePointsToDetSimple[experiment]`

For more complicated cases, this has to be defined manually (see subsections *Conditions for the LLP to decay inside the decay volume and for the products to point to the end of the detector*).

4 2. LLP distribution.nb

The output of the second notebook is a tabulated distribution of the form

$$\{m, \theta, E, f^{(i)}\}, \tag{4.1}$$

where the last column is the value of the distribution function for the given (m, θ, E), and the production mechanism i. The distribution is normalized to one. The tables are located in the following folder:

spectra/New physics particles spectra/⟨LLP⟩/⟨facility⟩/

In the case when the production is via scatterings, the output also includes the tabulated production probability, stored in the folder

phenomenology/⟨LLP⟩/Production probabilities

Depending on the specific LLP, OS, CPU, and the number of generated decays, the notebook runs for $< \mathcal{O}(20)$ minutes.

Often, if the notebook has already been run once for a given facility, it does not need to be launched again for the other experiments housed at the same facility, as the shape of LLP's distribution is determined mainly by the beam energy, while the normalization may be applied multiplicatively.

When launching the notebook, the user is prompted to choose the LLP and the facility at which the considered experiment is located. The notebook then computes the distributions of the LLPs produced by all implemented channels or by the channels selected by the user by evaluating the chapter *Choose LLP and run.* The code that computes the tabulated distribution for the given LLP `LLP` and process `proc` must located in the section with the name *LLP_proc* (for example, the code computing the distribution of HNLs produced by decays of W bosons is located in section *HNL_W*).

For the LLPs produced in decays, the calculation is organized as follows: the notebook first samples random mother particles according to their distributions (see the routine

`BlockPointsFromSmoothDistribution`

from the section *Working with mother particles distributions*). For this, it imports the distribution of the mother particle `particle` at the given facility `facility` from Sec. *Preliminary code: mother particles sampling* and accordingly generates `nsim` random mother 4-momenta. This is done by the routine `PointsSampler[Facility,particle, nsim, IfTwo]`. The last argument, `"True"` or `"False"`, tells whether there are two LLPs per single decay of `particle`.

Then (cf. section *Distribution of LLPs*), it generates the phase space of the LLPs in the rest frame of the decaying particles (the routine `ThreeBodyDecaysEventsAtRest[LLP, process, mLLP, nsim]`, boosts them (`BoostedLLPsBlock[LLP, facility, mLLP, process, nsim]`), and finally produces a smooth LLP angle-energy distribution based on the generated data

`FromDataToSmoothDistribution[boosteddata, mLLP, facility, IfTesting]`

This is systematically done with the help of the routine

`BlockTabulatedPDFsFromDecaysMass[LLP, Facility, mLLP, process, nsim, thetalist, Elist, thetaminExtr, IfTesting]`

Here, `thetalist`, `Elist` are the lists of the polar angles and LLP energies suitable for the given production channel and facility used for the tabulation, `thetaminextr` is the angle below which the

LLP's distribution is assumed to have the scaling

$$\frac{d^2f}{d\theta dE} \approx \frac{\sin(\theta)}{\sin(\theta_{\rm min,extr})}\left(\frac{d^2f}{d\theta dE}\right)_{\theta=\theta_{\rm min,extr}} \tag{4.2}$$

to avoid the lack of statistics.
The other implemented production channels include quasi-elastic scatterings (examples are the Primakov process and photon fusion for the ALPs), proton bremsstrahlung, and mixing with neutral particles. They are handled by the following blocks:

`BlockTabulatedPDFsFromPrimakov[Facility, Nucleus, thetalist, Elist]`

`BlockTabulatedPDFsFromFusion[Facility, Nucleus, thetalist, Elist]`

for the quasi-elastic scatterings, where `Nucleus` corresponds to the target nucleus;

`BlockTabulatedPDFsFromBrem[LLP, Facility,Description]`

where `Description` is the description of the bremsstrahlung channel (explained in the accompanying paper about the revision of the dark photon parameter space);

`BlockTabulatedPDFsFromMixingsMass[LLP, Facility, mLLP, process, thetalist, Elist, thetaminExtr]`

The implementation will be made more systematic in the future.
The UFO files used to generate the distributions of LLPs from the DIS processes are located in the folder *UFO files*. Those are pre-computed using `MadGraph+pythia8` and stored, for each of the LLPs, in the folder

spectra/New physics particles spectra/⟨LLP⟩/Pregenerated

4.1 What users have to provide

The users have to provide:

1. The tabulated distributions of the mother particles in the form `polar angle in rad`, `Energy in GeV`, `Value of the distribution`. The distributions are located in the folder

   spectra/SM particles/

   They must be normalized by one. To be found and imported by the notebook, the filename of the given distribution must match the pattern `DoubleDistr_<facility>_<mother>.txt`. The names of the mother particles may be found in the output cell of the section `Masses and other parameters of SM particles`.

2. The information about the given production process is to be stored in Chapter `Preliminary code: LLP production phenomenology`. The minimal information includes: the name of the process (to be added to the list `ProcessesListAll[<LLP>]`); the production type for the given process (`ProductionType[<LLP>,<process>]`), which may be "Decay", "Mixing" (if the production is via the mixing with neutral particles), or "Scattering" (if the production is via the scatterings); the list of particles participating in the production process `ReactionProductsList[<LLP>, <process>]`. The first particle is always the mother product, and by convention, LLPs must be at the end of this list. For example, for the production process `B-3-body`, when HNLs are created by 3-body decays of B mesons, it is {`"Bplus", "Dstar", "e", "HNL-mixing-e"`}. Another example is proton bremsstrahlung, where the list looks as, e.g., {`"p","DP"`}. If the decay process is a 3-body, then one needs to add the matrix element expressed in terms of the LLP mass, and the energies of the 1st and 3rd decay products (`Msquared3Body[<LLP>, <proc>, mLLP, E1, E3]`). If the same decaying particle can produce the LLP through multiple channels, one requires the branching ratios of these various production channels. The implemented examples include HNLs, which may be produced by 2- and 3-body decays of D and B mesons; see Sec. *HNL* in chapter *Preliminary code:*

LLP production phenomenology, as well as the routine `BoostedHNLsBlock[LLP, Facility, mLLP, process, nsim]`.

3. If the given LLP production process is a and the matrix elements if the production channel is a 3-body decay, cf. section *M^2 of 3-body production for different LLPs*). The implemented The branching ratios for the implemented LLPs are located in the folder

   phenomenology/⟨LLP⟩/branching ratios/

   Users can add their branching ratios in any format, as long as they are imported appropriately.

5 3. LLP sensitivity.nb

5.1 What the notebook does

Once the user specifies the experiment and LLP (section *Specifying the experiment*), the notebook interpolates the tabulated functions obtained by the previous notebook and constructs the integral of the number of events (section *Number of events*). The simplest way is to obtain the number of events using the built-in Mathematica functions `Interpolation` and `NIntegrate`; this is the function `NeventsInt[m, coupling, ProdChannel]`, where `ProdChannel` is the given LLP's production channel. It is defined in the subsection *Number of events - slow.* The resulting brute-force integrals are very slow ($\mathcal{O}$(1 second) per LLP's parameter space point "mass-coupling"). Given that the number of events has to be tabulated over a dense grid of LLP masses and couplings, the total computation time would be significant. In addition, the integration domain in θ, E, z may be much wider than the domain the events typically belong to. This is the case for the experiments where the detector covers a much narrower domain of the angles than the decay volume does. The Monte-Carlo integral (which randomly samples points inside the whole decay volume region) may return results with large errors in this situation.

To reduce the error and speedup the computation, the subsection *Number of events - fast* introduces a much faster and more stable way of calculating the number of events using quick grid mapping (the function `NeventsDiscret[m, ProdChannel, couplingslist]`, where `couplingslist` is the list of couplings, e.g., $\{10^{-7.}, 10^{-6.}\}$).

The brute-force integral representation of the number of events serves to cross-check for the fast method (section *Tests*). Typically, the agreement between the two methods is within 10%. Larger discrepancies may happen in a few cases:

1. Rarely, the θ, E, z grid density assumed by default for the computation of `NeventsDiscret`, `OutGridθfinal`, `OutGridEfinal`, `OutGridzfinal`, is insufficient. To exclude this reason, the user may just increase the grid density from 30 to a higher value in these strings:

   ```
   Do[{OutGridθfinal[prod], Δθvals[prod]} = OutGridθTemp[InGridθϵ, 30, prod, θmaxBrem],
   {prod, ProductionList}]
   ```

   ```
   {OutGridzfinal, Δzvals} = OutGridszTemp[InGridzϵ, 30, zminExp];
   ```

   to see whether the value of `NeventsDiscret` converges to `Nevents`.

2. A much more common reason is when the Monte-Carlo integration in `Nevents` fails to produce accurate results due to the abovementioned reasons.

5.1.1 Generalized acceptances

The section *Acceptances and approximate lower/upper bounds computation* produces the acceptances that give a qualitative understanding of the contribution of the factors entering Eq. (1) of the accompanying paper (the LLP distribution function, the azimuthal acceptance, the decay probability, the decay products acceptance) on the sensitivity at the lower bound, where the number of events is

$$N_{\text{ev}} = \frac{N_{\text{prod}}}{c\tau_{\text{LLP}}} \times \int d\theta dE dz f_{\text{LLP}} \epsilon_{\text{az}} c\tau_{\text{LLP}} \frac{dP_{\text{decay}}}{dz} \times \epsilon_{\text{dec}} \tag{5.1}$$

where the integral is the LLP lifetime-independent. Namely, the data exported in the folder

    Auxillary data/⟨LLP⟩/⟨experiment⟩

has the form

$\mathcal{I}_0[m_{\text{LLP}}]$, $\mathcal{I}_1[m_{\text{LLP}}]$, $\mathcal{I}_2[m_{\text{LLP}}]$, $\mathcal{I}_3[m_{\text{LLP}}]$

The integrals $\mathcal{I}_i$ are:

$$\mathcal{I}_0 = N_{\text{prod}}, \tag{5.2}$$

which shows the total number of LLPs produced at the given facility;

$$\mathcal{I}_1 = \frac{N_{\text{prod}}}{z_{\text{max}} - z_{\text{min}}} \int d\theta dE dz f_{\text{LLP}} \epsilon_{\text{az}}, \tag{5.3}$$

which shows the fraction of LLPs traveling in the direction of the decay volume times the length of the intersection of the LLP trajectory and the decay volume (normalized by the total number of the produced LLPs times the longitudinal length of the decay volume $z_{\text{max}} - z_{\text{min}}$);

$$\mathcal{I}_2 = \frac{N_{\text{prod}}}{c\tau_{\text{LLP}}} \int d\theta dE dz f_{\text{LLP}} \epsilon_{\text{az}} c\tau_{\text{LLP}} \frac{dP_{\text{dec}}}{dz}, \tag{5.4}$$

which is the fraction of the LLPs decaying inside the decay volume;

$$\mathcal{I}_3 = \frac{N_{\text{prod}}}{c\tau_{\text{LLP}}} \int d\theta dE dz f_{\text{LLP}} \epsilon_{\text{az}} c\tau_{\text{LLP}} \frac{dP_{\text{dec}}}{dz} \epsilon_{\text{dec}}, \tag{5.5}$$

which is the fraction of LLPs decaying inside the decay volume with the decay products satisfying the decay acceptance criteria. $\mathcal{I}_1$ shows the actual limitations on the sensitivity caused by the geometry of the experiments. $\mathcal{I}_2$ is the number of decays assuming that $\epsilon_{\text{dec}} \equiv 1$. $\mathcal{I}_3$ is the total number of events at the lower bound. For $\mathcal{I}_{2,3}$, the exporting is for $c\tau_{\text{LLP}} = 10^9$ m. The bound may be easily rescaled for an arbitrary $c\tau_{\text{LLP}}(m_{\text{LLP}}, g_{\text{LLP-SM}})$, where $g_{\text{LLP-SM}}$ is the LLP-SM coupling, given that these quantities scale as $(c\tau_{\text{LLP}})^{-1}$. In addition, when comparing the acceptances for various experiments, the dependence on $c\tau_{\text{LLP}}$ does not matter.
Basically, assuming that $N_{\text{ev}} \propto g^4$, where g is the LLP's coupling to the SM particles, the sensitivity at the lower bound is

$$g^2_{\text{lower}} = \sqrt{N_{\text{min}} c\tau_{\text{LLP}}}/\sqrt{\mathcal{I}_4}\Big|_{\text{g = 1}}, \tag{5.6}$$

where N_{min} is the critical number of events determining the boundary of the sensitivity region. These lower bounds estimates are also performed in the notebook.

5.1.2 Exporting tabulated number of events and sensitivities

It may be possible to set the decay products acceptance to 1 when computing the number of events by changing the line

```
integrandtab = TableIntegrandDiscret[m, ProdChannel, "True"];
```

to

```
integrandtab = TableIntegrandDiscret[m, ProdChannel, "False"];
```

in `NeventsDiscret[m, ProdChannel, couplingslist]`.

The section *Exporting tabulated number of events* then exports the tabulated number of events in the form `LLP mass in GeV`, `LLP coupling`, `Number of events` for all the relevant production channels. The output is located in the folder
`Tabulated Nevents/⟨LLP⟩/⟨experiment⟩/`
Finally, the section *Computing sensitivities* imports the tabulated numbers of events, sums them over the production channels, and computes the sensitivity. The user must specify the minimum number of events and some model-specific parameters. The sensitivity curves can then be found in the folder
Sensitivity domains/⟨LLP⟩/⟨experiment⟩/
The typical running time is $\mathcal{O}(5-10)$ minutes.

5.2 What users have to provide

The users have to provide:

1. The tabulated LLP distribution functions and acceptances generated by the previous modules. At the very beginning of the notebook, it scans the folders

   spectra/New physics particles spectra/¡LLP¿

   Acceptances

   on the available tabulated angle-energy distributions and acceptances.

2. The probabilities of producing the LLP through various mechanisms, `ProbLLP[mLLP, coupling, prod]`, where `mLLP` is the LLP mass in GeV, `coupling` is the coupling in appropriate units, and `prod` must exactly match the name of the corresponding angle-energy distribution. Typically, this probability is the branching ratio of the decay (for the decay process) or the scattering production probability per proton-proton collision (for the production by, e.g., deep inelastic scattering). For already implemented LLPs, the tabulated probabilities are located in the folder

   phenomenology/⟨LLP⟩/branching ratios/

   The probability of producing the LLP through the given process is

   `ProbMother[prod]*ProbLLP[mLLP, coupling, prod]`

   where `ProbMother` is the probability of having a mother particle per proton collision (defined in the chapter *Angle-energy distributions and differential yields for the given experiment*). If one considers `prod` as a decay of some mother particles X or the mixing with it, it is simply the fraction of the produced X per collision, `ProbMother` $= \sigma_{p\text{ collision}\to X}/\sigma_{p\text{ collision}}$. For the given experiment, it may be extracted from the `.mx` file with tabulated acceptances using the table `CrossSectionData` (assigned in the chapter *Cross-sections, acceptances*). By convention, if the mother particle is a charged particle, `CrossSectionData` includes a factor of 2 due to the charge-conjugated production channel. If one considers the scattering of protons (the proton bremsstrahlung, DIS, etc.), it is one by definition.

   They are imported and interpolated in the section *LLP phenomenology*, selected for the particular facility and convoluted with the production fractions of the mother particles (for the production via decays of secondaries) in sub-section *Importing distributions*.

3. The lifetime $c\tau$ (in meters), `cτLLP[mLLP,coupling]`. For the implemented LLPs, the decay widths are located in the folder

   phenomenology/⟨LLP⟩/decay widths/

   They are imported and interpolated in the section *LLP phenomenology*.

6 4. Plots.nb

This small notebook simply scans the folder containing the sensitivities for the chosen LLP, lists all the available curves that match the specified minimum number of events and model-specific parameters, imports the selected curves, and makes the sensitivity plot.

7 EventCalc.nb

`EventCalc` is an additional module that generates detailed LLP event information. It does the same as `SensCalc`, except for decaying unstable decay products and propagating them through the detector. Its purpose is to pass the decay events to the full simulation framework of various experiments, where they have to be processed using detailed simulations.

The simplified logic of the module is as follows. Through the notebook, the user is subsequently asked about the experiment, the LLP, the production modes, and the decay modes. Then, it computes the tabulated distribution averaged over the selected production channels. Then, if the user selects the mass and the set of couplings describing the LLP, the routine

`NeventsBlock[mass, coupling1, coupling2, Nevents]`

produces the total number of events and two tables `llpinfodata`, `decayproductsdata`. The first table is about the properties of the decaying LLP. Each row has the columns

$\{p_{\texttt{LLP},x}\texttt{,}p_{\texttt{LLP},y}\texttt{,}p_{\texttt{LLP},z}\texttt{,}E_{\texttt{LLP}}\texttt{,}m_{\texttt{LLP}}\texttt{,}\cos(\theta_{\texttt{LLP}})\texttt{,}\frac{p_{\texttt{LLP}}}{m_{\texttt{LLP}}}\texttt{,}P_{\texttt{decay}}\texttt{,}x_{\texttt{decay}}\texttt{,}y_{\texttt{decay}}\texttt{,}z_{\texttt{decay}}\texttt{,}\epsilon_{\texttt{az}}\texttt{,}N_{\texttt{prod,tot}}\texttt{,}\epsilon_{\texttt{geom}}\}$

Here, the first 5 columns are the 4-momentum and the mass of the LLP, the 6th and 7th are the cosine of the polar angle and the momentum-over-mass of the LLP, the 8th is the decay probability P_{decay}, the 9th-11th are the coordinates of the decay vertex, $N_{\text{prod,tot}}$ is the total number of produced LLPs at the given facility, and ϵ_{geom} is the azimuthal acceptance ϵ_{az} for the given LLP – the probability that the given LLP will have the azimuthal angle such that its trajectory intersects the decay volume. ϵ_{geom} and $N_{\text{prod,tot}}$ are constant for the given simulation, whereas P_{decay} and ϵ_{az} vary from event to event. The units are GeV for energies, momenta and mass, and meters for the decay coordinates.

The total number of events (that is automatically computed by the code `NeventsLLP[llpinfodata]`) is

$$N_{\text{events,total}} = N_{\text{prod,tot}} \times \epsilon_{\text{geom}} \times \frac{1}{N_{\text{simulated}}} \sum_{i=1}^{N_{\text{simulated}}} \epsilon^{i}_{\text{az}} \cdot P^{i}_{\text{decay}}, \tag{7.1}$$

where $N_{\text{simulated}}$ is the number of simulated events – the last argument in the `NeventsBlock` block.

The decay products table has the columns

$p_x\texttt{,}p_y\texttt{,}p_z\texttt{,}E\texttt{,}m$`,pdg, electric charge, stability`

The last two quantities are internal for `SensCalc` and may be dropped. Here, `stability` is 1 if the given particle is stable or metastable (such as muons, charged pions, or kaons), and 0 otherwise. `electric` charge for quarks is assumed to be an integer for internal purposes.

The block `NeventsBlock` also exports the two mentioned tables.

Currently, only dark photons are properly implemented in `EventCalc`. The other LLP models available in `SensCalc` will be added shortly after.

Some example of usage is provided in the subsection *Test* at the end of the notebook.