# Hazma[☆]

## Adam Coogan[a,b,c], Logan A. Morrison[b,c]

[a]*GRAPPA, Institute of Physics, University of Amsterdam, 1098 XH Amsterdam, The Netherlands*
[b]*Department of Physics, 1156 High St., University of California Santa Cruz, Santa Cruz, CA 95064, USA*
[c]*Santa Cruz Institute for Particle Physics, 1156 High St., Santa Cruz, CA 95064, USA*

**Abstract**

Hazma does many wonderful things.

*Keywords:* Dark matter, indirect detection

## 1. Introduction

Hazma is great.

*Notation/units.* Unless otherwise noted, the units used throughout `hazma` are

- Masses: MeV

- Decay widths: MeV

- Cross sections: $\mathrm{cm}^3/\mathrm{s}$

## 2. Installation and Package Structure

### 2.1. Installation

Before installing `hazma`, the user needs to have a few of well-established packages : cython, scipy, numpy, and scikit-image. Theses are easily installed by using PyPi. If the user has PyPi installed on their system, then these packages can be installed using

---

[☆]Code and tutorials available at github.com/LoganAMorrison/Hazma.

*Email addresses:* `a.m.coogan@uva.nl` (Adam Coogan), `loanmorr@ucsc.edu` (Logan A. Morrison)

```
pip install cython, scipy, numpy, scikit-image
```

Hazma can be installed in the same way, using:

```
pip install hazma
```

This will download a tarball from the PyPi repository, compile all the c-code and install `hazma` on the system. Alternatively, the user can install `hazma` by downloading the package from `https://github.com/LoganAMorrison/Hazma.git`. Once downloaded, navigate to the package directory using the command line and run either

```
pip install .
```

or

```
python setup.py install
```

Note that since `hazma` makes extensive usage of the package cython, the user will need to have a c-compiler installed on their system. For more information, see the cython installation guide: `https://cython.readthedocs.io/en/latest/src/quickstart/install.html`

## 3. Overview: The `theory` Class

<span style="color:red">Should we have boxes for different functions and constructors specifying type signatures? Should that be left for the actual docs in the code? Seems like it, based on other publications in this journal.</span>

Instances of the `theory` class represent dark matter models, and are the main objects used to perform analyses in Hazma. Two `theory`s, each containing a DM and mediator particle, come built into `hazma`The Lagrangians defining these theories and the domains of validity are described in the following subsection. `theory` possesses methods to compute quantities such as DM annihilation cross sections, gamma ray spectra and constraints, and CMB constraints. This section catalogues the definitions of these quantities used by `hazma` along with minimal examples of how to use the associated methods.

2

Each of the models that ship with `hazma` contain two BSM particles:

- A dark matter particle;

- A mediator $M$ that interacts with the DM as well as Standard Model particles.

The Lagrangians can be expressed as

$$\mathcal{L} = \mathcal{L}_{\mathrm{SM}} + \mathcal{L}_{\mathrm{DM}} + \mathcal{L}_{\mathrm{M}} + \mathcal{L}_{\mathrm{Int}(M)}, \tag{1}$$

which consists of the SM Lagrangian, the free Lagrangians for the mediator and DM, and the mediator's interactions with the DM and SM fields. Both the dark matter and the mediator are taken to be uncharged under the Standard Model gauge group. The Lagrangian is defined in terms of the microscopic degrees of freedom of the Standard Model (quarks, leptons and gauge bosons). However, at the energy scale of interest for self-annihilations of nonrelativistic MeV dark matter, quarks and gluons are not the correct strongly-interacting degrees of freedom. Instead, the microscopic Lagrangian must be matched onto the effective Lagrangian for pions and other mesons using the techniques of chiral perturbation theory (chPT). The models currently implemented in `hazma` utilize leading-order chPT.

As with an effective field theory (EFT), chPT has a limited range of validity. Observables in chPT are computed in terms of an expansion in a small parameter, the meson momentum $p$ divided by the mass scale $\Lambda_{\mathrm{ChPT}} \sim 4\pi f_\pi \sim 1$ GeV, where $f_\pi$ is the pion decay constant. As $p^2 \to \Lambda_{\mathrm{ChPT}}$, higher-order Feynman diagrams in the chPT expansion provide contributions to observables comparable to leading order ones. This suggests that leading-order chPT cannot be trusted for computing dark matter self-annihilation cross sections when $m_{\mathrm{DM}} \gtrsim 500$ GeV. In fact, the convergence of the chPT expansion is disrupted at a lower mass scale by the lowest-lying hadronic resonances, the $\rho$ ($m_\rho = 770$ GeV) and the $f_0(500)$ ($m_{f_0(500)} \sim 450$ MeV). Figure 1 illustrates the resulting regions
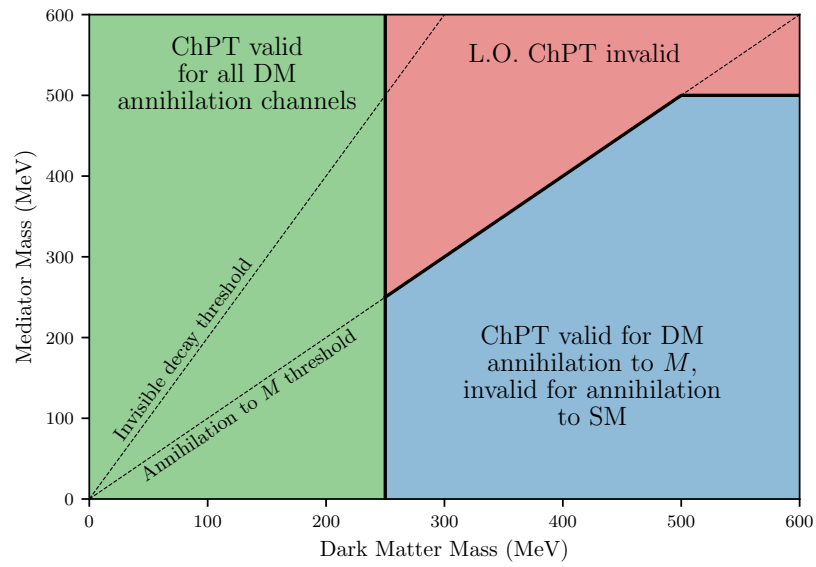
Figure 1: Values of the DM mass $m_{\mathrm{DM}}$ and mediator mass $m_M$ for which the leading-order chiral perturbation theory calculations used to implement the models built into `hazma` can be trusted.

where the leading-order chPT calculations can and cannot be trusted in the $(m_{\text{DM}}, m_{\text{mediator}})$ plane. An important consequence is that annihilation into kaons and heavier mesons can be ignored since that would require DM masses far above the chPT range of validity. This is important to keep in mind when using the models provided with `hazma`.[2] Of course, user-defined models (see Section 5) need not use leading-order chPT.

While the preceding discussion applies irrespective of the mediator and DM spin and quantum numbers, the rest of this subsection specializes to the two models that come with `hazma scalar_mediator`, which contains a real scalar mediator $S$ and `vector_mediator`, where the mediator is a vector $V$. In both cases the DM is taken to be a Dirac fermion, so that

$$\mathcal{L} = \bar{\chi}(i\slashed{\partial} - m_\chi)\chi. \tag{2}$$

The rest of this subsection presents $\mathcal{L}_{\text{Int}(S)}$ and $\mathcal{L}_{\text{Int}(V)}$ at the level of quarks and gluons as well as the Lagrangians obtained by performing the chPT matching. Snippets are provided to demonstrate how to construct each `theory` and change its parameters. The interaction Lagrangians, matching procedure and a brief overview of the chiral Lagrangian are explained in detail in a companion paper [**?**].

### 3.1.1. scalar_mediator

The free Lagrangian for a real scalar is

$$\mathcal{L}_S = \frac{1}{2}(\partial_\mu S)(\partial^\mu S) - \frac{1}{2}m_S^2 S^2, \tag{3}$$

where $m_S$ is the scalar's mass. The interactions with the light fundamental SM degrees of freedom

$$\mathcal{L}_{\text{Int}(S)} = -S\left(g_{S\chi} + g_{Sf}\sum_f \frac{y_f}{\sqrt{2}}\bar{f}f\right) \tag{4}$$

$$+ \frac{S}{\Lambda}\left(g_{SF}\frac{\alpha_{\text{EM}}}{4\pi}F_{\mu\nu}F^{\mu\nu} + g_{SG}\frac{\alpha_s}{4\pi}G_{\mu\nu}^a G^{a\mu\nu}\right). \tag{5}$$

---
2

The sum runs over fermions with mass below the GeV scale ($f = e, \mu, u, d, s$). Note that the coupling $g_{Sf}$ is outside the sum. The Yukawas are defined to be $y_f = \sqrt{2} m_f / v_h$, with the Higgs vacuum expectation value (vev) defined as $v_h = 246$ GeV. The parameter $\Lambda$ is the mass scale at which $S$ acquires (non-renormalizable) interactions with the photon and gluon.

After performing the matching onto the chiral Lagrangian and expanding to leading order in the pion fields, the resulting interaction Lagrangian is

$$\mathcal{L}_{\text{Int}(S)} = \frac{2g_{SG}}{9\Lambda} S \left[ (\partial_\mu \pi^0)(\partial^\mu \pi^0) + 2(\partial_\mu \pi^+)(\partial^\mu \pi^-) \right] \tag{6}$$

$$+ \frac{4ieg_{SG}}{9\Lambda} S A^\mu \left[ \pi^- (\partial_\mu \pi^+) - \pi^+ (\partial_\mu \pi^-) \right] \tag{7}$$

$$- \frac{B(m_u + m_d)}{6} \left( \frac{3g_{Sf}}{v_h} + \frac{2g_{SG}}{3\Lambda} \right) S \left[ (\pi^0)^2 + 2\pi^+\pi^- \right] \tag{8}$$

$$+ \frac{B(m_u + m_d)g_{SG}}{81\Lambda} \left( \frac{2g_{SG}}{\Lambda} - \frac{9g_{Sf}}{v_h} \right) S^2 \left[ (\pi^0)^2 + 2\pi^+\pi^- \right] \tag{9}$$

$$+ \frac{4e^2 g_{SG}}{9\Lambda} S \pi^+ \pi^- A_\mu A^\mu \tag{10}$$

$$- g_{S\chi} S \bar{\chi} \chi - g_{Sf} S \sum_{\ell = e, \mu} \frac{y_\ell}{\sqrt{2}} \bar{\ell}\ell + \frac{\alpha_s}{4\pi\Lambda} g_{SG} S G^a_{\mu\nu} G^{a\mu\nu}. \tag{11}$$

The parameters for the scalar model are attributes of the `scalar_mediator` class. Their names in `hazma` are

$$(m_\chi, m_S, g_{S\chi}, g_{Sf}, g_{SG}, g_{SF}, \Lambda) \leftrightarrow (\texttt{mx}, \texttt{ms}, \texttt{gsxx}, \texttt{gsff}, \texttt{gsGG}, \texttt{gsFF}, \texttt{lam}).$$

The following snippet shows how to create an instance of `scalar_mediator`, change the value of a parameter, and print its new value:

```
>>> from hazma.scalar_mediator import ScalarMediator
>>> sm = ScalarMediator(mx=150., ms=1e3, gsxx=1., gsff=0.1,
...                      gsGG=0.1, gsFF=0.1, lam=2e5)
>>> print(sm.gsff)
0.1
>>> sm.gsff = 0.5
>>> print(sm.gsff)
0.5
```

<span style="color:red">Talk about Higgs portal and heavy fermion subclasses!</span>

### 3.1.2. `vector_mediator`

For the vector mediator the free part of the Lagrangian is

$$\mathcal{L}_V = -\frac{1}{4}V_{\mu\nu}V^{\mu\nu} + \frac{1}{2}m_V^2 V_\mu V^\mu, \tag{12}$$

where $m_V$ is the mass of the vector. The interactions considered are

$$\mathcal{L}_{\text{Int}(V)} = V_\mu \left( g_{V\chi}\bar{\chi}\gamma^\mu\chi + \sum_f g_{Vf}\bar{f}\gamma^\mu f \right) - \frac{\epsilon}{2}V^{\mu\nu}F_{\mu\nu}. \tag{13}$$

The sum again runs over the light fermions ($f = e, \mu, u, d, s$), and $V$ may have different couplings to each of these. The last term is a kinetic mixing between the photon and $V$, which can be eliminated by transforming the photon $A_\mu \to A_\mu - \epsilon V_\mu$. Upon this field redefinition $V$ acquires an $\epsilon$-suppressed interaction with the SM fermions, which is captured by changing the fermion couplings

$$g_{Vf} \to g_{Vf} - \epsilon e Q_f, \tag{14}$$

where $Q_f$ is the electric charge of the fermion $f$ and $e > 0$ is the electron's charge.

Matching onto the chiral Lagrangian and isolating the terms contributing at leading order to the quantities computed in `hazma` gives

$$\mathcal{L}_{\text{Int}(V)} = -i(g_{Vu} - g_{Vd})V^\mu \left( \pi^+\partial_\mu\pi^- - \pi^-\partial_\mu\pi^+ \right) \tag{15}$$

$$+ (g_{Vu} - g_{Vd})^2 V_\mu V^\mu \pi^+\pi^- \tag{16}$$

$$+ 2e(Q_u - Q_d)(g_{Vu} - g_{Vd})A_\mu V^\mu \pi^+\pi^- \tag{17}$$

$$+ \frac{1}{8\pi^2 f_\pi}\epsilon^{\mu\nu\rho\sigma}(\partial_\mu\pi^0) \tag{18}$$

$$\times \{ e(2g_{Vu} + g_{Vd})\left[ (\partial_\nu A_\rho)V_\sigma + (\partial_\nu V_\rho)A_\sigma \right] \tag{19}$$

$$+ 3(g_{Vu}^2 - g_{Vd}^2)(\partial_\nu V_\rho)V_\sigma \}. \tag{20}$$

<span style="text-align:center">7</span>

The base `vector_mediator` class contains the parameters[3]

$$(m_\chi, m_V, g_{V\chi}, g_{Vu}, g_{Vd}, g_{Vs}, g_{Ve}, g_{V\mu})$$

$$\leftrightarrow (\mathtt{mx}, \mathtt{mv}, \mathtt{gvxx}, \mathtt{gvuu}, \mathtt{gvdd}, \mathtt{gvss}, \mathtt{gvee}, \mathtt{gvmumu}).$$

The following snippet shows how to instantiate and alter the parameter values for `vector_mediator`:

```
>>> from hazma.vector_mediator import VectorMediator
>>> vm = VectorMediator(mx=150., mv=1e3, gvxx=1., gvuu=0.1,
...                      gvdd=0.2, gvss=0.3, gvee=0.4,
...                      gvmumu=0.5)
>>> print(vm.gvee)
0.4
>>> print(vm.gvuu)
0.1
>>> vm.gvuu = 0.05
>>> print(vm.gvuu)
0.05
```

For convenience, a subclass called `KineticMixing` is also provided to handle the important case where $V$ couples to the SM purely through the kinetic mixing term ($\epsilon \neq 0$, $g_{Vf} = 0$ for all $f$). The parameters for this subclass are

$$(m_\chi, m_V, g_{V\chi}, \epsilon) \leftrightarrow (\mathtt{mx}, \mathtt{mv}, \mathtt{gvxx}, \mathtt{eps}).$$

While the underlying parameters `gvuu`, ..., `gvmumu` can be accessed by instances of `KineticMixing`, they cannot be set directly since they are fully determined by `eps`:

---

[3]Note that `gvss` is not currently used since `vector_mediator` is based on leading-order chPT.

```
>>> from hazma.vector_mediator import KineticMixing
>>> km = KineticMixing(mx=150., mv=1e3, gvxx=1., eps=0.1)
>>> km.eps
0.1
>>> km.gvuu
-0.020187846690459792  # = -0.1 * 2/3 * sqrt(4 pi / 137)
>>> km.gvuu = 0.1
AttributeError: Cannot set gvuu
```

*3.2. Cross section, decay widths and branching fractions*

*theory.list_annihilation_final_states().* Lists the final states into which the DM can potentially annihilate. These are based on the terms appearing in $\mathcal{L}_{\text{Int}(M)}$. Note that the actual branching fractions into a given final state may be zero if the relevant couplings are zero or the center of mass energy for the annihilation is smaller than the sum of the final state particles' masses.

```
>>> ScalarMediator.list_annihilation_final_states()
['mu mu', 'e e', 'g g', 'pi0 pi0', 'pi pi', 's s']
>>> VectorMediator.list_annihilation_final_states()
['mu mu', 'e e', 'pi pi', 'pi0 g', 'v v']
```

This is not quite right. For example, it excludes the final states $e^+e^-\gamma$, $S\pi^0\pi^0$, $\pi^+\pi^-\gamma\gamma$ for the scalar and $V\pi^0$ for the vector. While that's fine since those final states are so suppressed, the wording here needs to be precise.

*theory.annihilation_cross_sections(cme).* For an instance `th` of theory and a center of mass energy `cme`, this returns a `dict` whose keys are the elements of `th.list_annihilation_final_states()` and `'total'` and whose values are the corresponding cross sections for annihilating into those states. In the following example, the DM is lighter than the mediator, but the center of mass energy is large enough to permit annihilations into the mediator:

9

```
>>> sm = ScalarMediator(mx=180., ms=190., gsxx=1., gsff=0.1,
...                      gsGG=0.1, gsFF=0.1, lam=2e5)
>>> xsecs = sm.annihilation_cross_sections(400.)
>>> print(xsecs)
{'g g': 5.668702951121324e-32,
 'e e': 4.525214520210817e-32,
 'pi0 pi0': 1.8542010286144253e-27,
 'total': 4.1600065791804136e-18,
 's s': 4.160006572615015e-18,
 'mu mu': 1.1842156064646993e-27,
 'pi pi': 3.526880321768581e-27}
```

*theory.annihilation_branching_fractions(cme).* For an instance `th` of theory and a center of mass energy `cme`, this returns a `dict` whose keys are the elements of `th.list_annihilation_final_states()` and whose values are the corresponding branching fractions for annihilating into those states.

```
>>> km = KineticMixing(mx=170., mv=1e3, gvxx=1., eps=0.2)
>>> bfs = km.annihilation_branching_fractions(400.)
>>> print(bfs)
{'pi0 g': 1.5585058782195258e-05,
 'e e': 0.4855695299567697,
 'mu mu': 0.4698109970123726,
 'v v': 0.0,
 'pi pi': 0.04460388797207543}
```

*theory.partial_widths().*

*3.3.  Gamma ray and positron spectra*

*3.4.  Gamma ray limits*

Explain procedure

*3.5. Cosmic Microwave Background limits*

Explain procedure

*3.6. Gamma-ray and Positron Spectra*

In `hazma`, their are two main types of spectra: Final state radiation (FSR) and decay spectra. Processes where there is a photon or electron in the immediate final state are classified as FSR. For example, the process of two dark matter particles annihilating through a scalar mediator into two charged fermions or mesons and a photon, i.e. $\overline{\chi}\chi \to S^* \to \overline{f}f\gamma$ or $\overline{\chi}\chi \to S^* \to \pi^+\pi^-\gamma$. Decay spectra arise when one of the final state particles is unstable and is able to decay radiatively, like the muon, charged pion or neutral pion. If this is the case, the spectrum for the state $A$ decaying or annihilating into the state $XB$ with $X$ being some state and $B$ then decaying radiatively to the state $C\gamma$, can be approximated using the narrow width approximation as

$$\frac{dN(A \to XB \to XC\gamma)}{dE} \sim \text{BF}(A \to XB)\frac{dN(B \to X\gamma)}{dE} \qquad (21)$$

An example of such a proccess is two dark matter particles annihilating into two muons or two pions: $\overline{\chi}\chi \to S^* \to \mu^+\mu^-$ or $\overline{\chi}\chi \to S^* \to \pi^+\pi^-$. For both cases, the final state particles are unstable and will decay. For the decay spectra, it is easiest to compute the gamma-ray or positron spectra in the rest frame of the decaying particle. Then, to compute the spectra in another frame, the spectrum must be boosted.

*3.6.1. Boost Integrals*

## 4. How Hazma Works

## 5. Advanced Usage

- Make a list or graphic showing what the user has to implement.

## 6. Conclusion

## 7. Bibliography

Here are two sample references: [1, 2].

## References

[1] R. Feynman, F. Vernon Jr., Annals of Physics 24 (1963) 118–173. doi:`10.1016/0003-4916(63)90068-X`.

[2] P. Dirac, Physica 19 (1953) 888–896. doi:`10.1016/S0031-8914(53)80099-6`.