Cosmic-ray upscattered inelastic dark matter

This notebook contains the calculations, analysis and figures for the paper arXiv:2108.00583

Introduction

This notebook calculates the flux and observable rates of cosmic-ray upscattered inelastic dark matter.

The halo dark matter is assumed to be in the low mass state χ_1 with mass m_{χ_1}, this can be upscattered to a heavier state χ_2 with mass $m_{\chi_2} = m_{\chi_1} + \delta$
$\left(m_{\chi_1}\right.$ and δ are taken as the model parameters ,
$\left. m_{\chi_2} \text{ is not used so all refereces to mass can be assumed to be } m_{\chi_1}\right)$

Throughout the calculations units are generally assumed to be in GeV, except where explicitly named in the variable.

- Gray (initialization) cells can be evaluated automatically
- Red cells are the main computations and can take minutes to run
- Blue cells create plots
- Purple cells create output files

Constants

In[ ]:=
```
ħc=0.19732698 ;(*GeV  fm*)
Centimeter = 1/ħc×10^13/GeV;
c=2.997925 ×10^8;
Second =c 10^2Centimeter ;
Kilogram = c^2/(1.602×10^-19) GeV/10^9;
kpc=3.086×10^21Centimeter ;
KilogramDay =(1Kilogram )(24×3600 Second );
tonneYr =365250  KilogramDay ;
mp=0.93827208816 //SetPrecision [#,11]&;
Deff=0.997 kpc;
ρX=0.3 GeV/Centimeter^3;
unitsCM2S =(Centimeter^2/cm^2)(Second/s);
amu= mp/1.00727647//SetPrecision [#,10]&;
mn=0.93956542052 //SetPrecision [#,11]&;
Me=510.998950 keV;
```

Set plot styles and custom plot functions

In[ ]:=
```
SetOptions [#,Frame ->True ,FrameStyle ->Directive [Black ,18,FontFamily ->Times ,Thickness [.004]],A
SetOptions [#,Frame ->True ,LabelStyle ->Directive [Black ,16,FontFamily ->Times ,Thickness [.004]],A
logSpace [a_, b_, n_] := 10.0 ^Range [Log10 [a], Log10 [b], (Log10 [b] - Log10 [a])/(n - 1)];

topLeft [x_] := Placed [x, {Scaled [{.38 , .95}], {.9, 1}}];
topRight [x_] := Placed [x,{Scaled [{.95 , .95}], {.9, 1}}];
Legend //Clear ;
Legend [legendList_ ,opt:OptionsPattern [{Position ->"Right ",Type->"Line ",LineLegend }]] := Module
If[OptionValue [Type]=="Line ",
f=LineLegend ;];
If[OptionValue [Type]=="Swatch ",
f=SwatchLegend ;];
If[OptionValue [Position ]=="Right ",
   f[ Style[#, 14, FontFamily  -> "Times "] & /@ legendList , FilterRules [{opt},Except [{Position
If[OptionValue [Position ]=="Left ",
   f[ Style[#, 14, FontFamily  -> "Times "] & /@ legendList , FilterRules [{opt},Except [{Position

f[ Style[#, 14, FontFamily  -> "Times "] & /@ legendList , FilterRules [{opt},Except [{Position ,Ty
];

LogTicks //Clear ;
LogTicks [min_,max_,step_] := Block [{lmin , lmax , t},
   lmin  = Floor [Log10 [min]];
   lmax  = Floor [Log10 [max]];
   t=0;
   Return [{Join[
      Table [{10^i, If[Mod[++t,step]==0,Superscript [10, i],Null], {0.012 , 0}}, {i, Floor [lmin], C
       Table [{i*10^j, Null , {0.006 , 0}}, {j, Floor [lmin],
        Ceiling [lmax], 1}, {i, 0.1, 0.9, 0.1}]]],
     Join[Table [{10^i, Null , {0.012 , 0}}, {i, Floor [lmin],
     Ceiling [lmax]}], (Flatten [#1, 1] &)[ Table [{i*10^j, Null , {0.006 , 0}}, {j, Floor [lmin], Ceil
];
LogTicks [min_, max_] := LogTicks [min,max,1];
```

Cosmic-ray fluxes

Import spectra

We will consider cosmic-ray protons and helium, define their mass, charge, atomic number and spin:

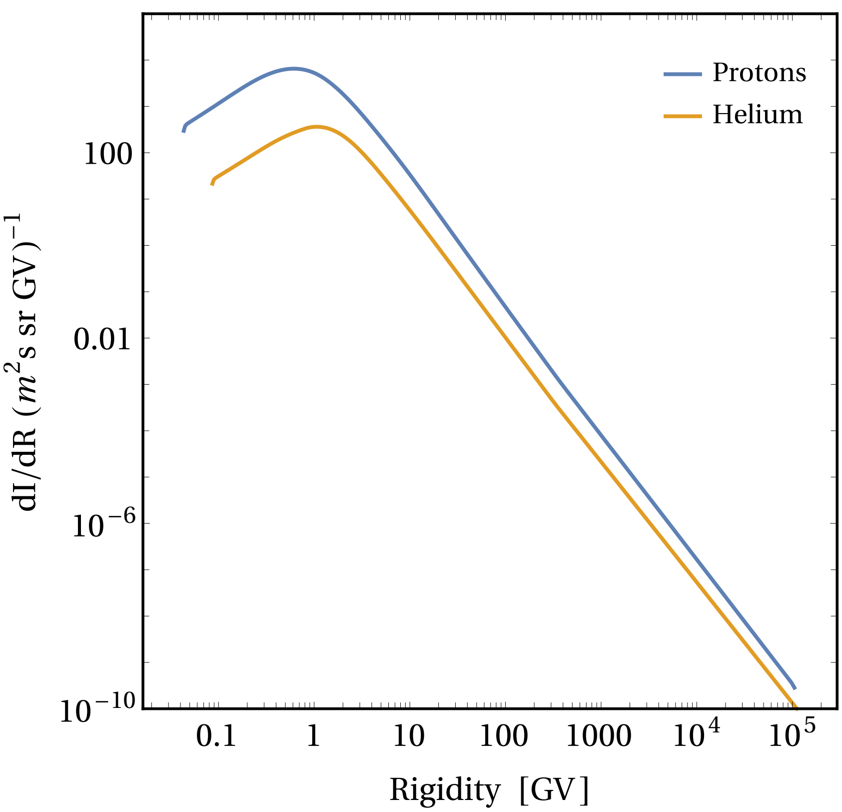
In[ ]:=
```
mi  = {mp, 4.002602 *amu};
Zei = {1,2};
Ai  = {1,4};
ji  = {0.5,0};
```

Import CR flux data tables and create interpolation functions:

In[ ]:=
```
NotebookDirectory []//SetDirectory ;
protonLISdata =Import ["data/TABLE_Protons _R.txt","Table "]//Drop[#,1]&;
HeLISdata =Import ["data/TABLE_Helium _R.txt","Table "]//Drop[#,1]&;
dIdR={Interpolation [protonLISdata ,InterpolationOrder →1],Interpolation [HeLISdata ,Interpolatio
```

Plot raw data, which is a function of rigidity:

In[ ]:=
```
ListLogLogPlot [
 {protonLISdata , HeLISdata}, Joined → True,
 FrameLabel → {"Rigidity [GV]", "dI/dR (m^2s sr GV)^-1"},
 PlotRange → {10^-10, 10^5},
 PlotLegends → Legend[{"Protons", "Helium"}]]
```

Out[ ]=


Converting to kinetic energy

Functions for rigidity (R) in terms of kinetic energy (T) and vise-versa:

In[ ]:=
```
Rt[T_,ii_]:= Sqrt[T^2+2mi[[ii]] T]/Zei[[ii]]
```

In[ ]:=
```
TR[R_,jj_]:= Sqrt[(mi[[jj]])^2+(R Zei[[jj]])^2] -mi[[jj]]
```

Max kinetic energy in provided data files:

In[ ]:=
```
TdatMin ={TR[protonLISdata [[1,1]],1],TR[HeLISdata [[1,1]],2]}
TdatMax ={TR[protonLISdata [[-1,1]],1],TR[HeLISdata [[-1,1]],2]}
```

Out[ ]= {0.000999971 , 0.00399668 }

Out[ ]= {105 099. , 420 396. }

In[ ]:=
```
dRdT =D[ Sqrt[T^2+2mi T] ,T]/Zei
```

Out[ ]= $\left\{\frac{1.8765441763 + 2\,T}{2\,\sqrt{1.8765441763\,T + T^2}},\ \frac{7.4568 + 2\,T}{4\,\sqrt{7.4568\,T + T^2}}\right\}$

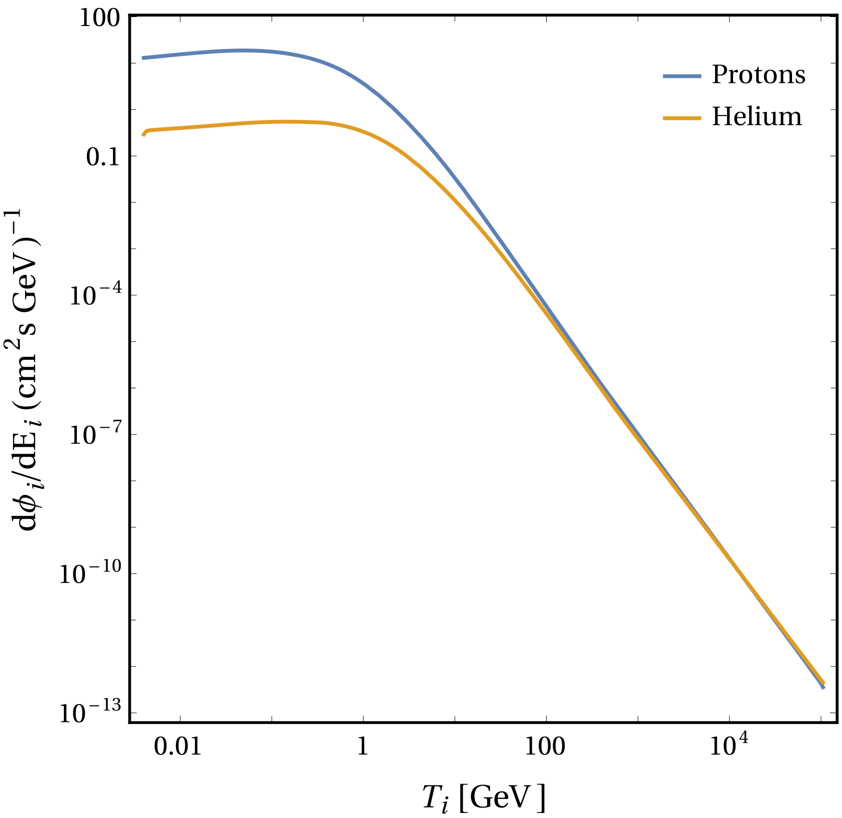
Converting to function of kinetic energy and giving flux in units of GeV^2

In[ ]:=
```
dϕdT[t_?NumericQ ,i_?IntegerQ ]:=Evaluate [4π/100^2  1/(Centimeter ^2Second  GeV)]dIdR[[i]][Rt[t ,i]] (dRdT[[i]]/.
```

In[ ]:=

```
LogLogPlot[{Centimeter^2 Second GeV dϕdT[t, 1], Centimeter^2 Second GeV dϕdT[t, 2]},
 {t, TdatMin // Max, TdatMax // Min}, FrameTicksStyle → Directive[Black, 15],
 FrameTicks → Automatic, PlotRange → All, AspectRatio → 1,
 PlotLegends → Legend[{"Protons", "Helium"}],
 FrameLabel → {Style["Tᵢ [GeV]", 18, Black, FontFamily → Times],
   Style["dϕᵢ/dEᵢ (cm²s GeV)⁻¹", 18, Black, FontFamily → Times]}]
```

Out[ ]=



Build flux tables:

In[ ]:=

```
pFlux = Table[{T, dϕdT[T, 1] GeV^-2}, {T, logSpace[TdatMin[[1]], TdatMax[[1]], 800]}] // Interpolation[#, Interpola
heFlux = Table[{T, dϕdT[T, 2] GeV^-2}, {T, logSpace[TdatMin[[2]], TdatMax[[2]], 800]}] // Interpolation[#, Interpol
```

CRDM flux

Kinematic limits

Find inelastic limits:

$p_i = p_i' + p_x$

$p_i'^2 = p_i^2 + p_x^2 - 2\, p_i\, p_x \cos\theta$

use $p^2 = E^2 - m^2$ and E = T + m

In[ ]:=

$$\text{Solve}\left[\left((\text{Tip}+\text{mii})^2-\text{mii}^2\right)==\left((\text{Ti}+\text{mii})^2-\text{mii}^2\right)+\left((\text{Tx}+\text{mxp})^2-\text{mxp}^2\right)+2\sqrt{\left((\text{Ti}+\text{mii})^2-\text{mii}^2\right)\left((\text{Tx}+\text{mxp})^2-\text{mxp}^2\right)}\ /.\ \{\text{mxp}\to\text{mx}\right.$$

Out[ ]=

$$\left\{\left\{\text{Tx}\to\frac{1}{2\,(\text{mii}+\text{mx})^2+4\,\text{mx}\,\text{Ti}}\left(2\,\text{mx}\,\text{Ti}\,(2\,\text{mii}+\text{Ti})-2\,(\text{mii}\,(\text{mii}+\text{mx})+\text{mx}\,\text{Ti})\,\delta+(\text{mii}+\text{mx}+\text{Ti})\,\delta^2-\sqrt{-\text{Ti}\,(2\,\text{mii}+\text{Ti})\left(-2\,\text{mx}\,\text{Ti}+2\,(\text{mii}+\text{mx})\,\delta+\delta^2\right)\left(2\,\text{mx}\,(\text{Ti}-\delta)-\delta^2+2\,\text{mii}\,(2\,\text{mx}+\delta)\right)}\right)\right\},\right.$$
$$\left.\left\{\text{Tx}\to\frac{1}{2\,(\text{mii}+\text{mx})^2+4\,\text{mx}\,\text{Ti}}\left(2\,\text{mx}\,\text{Ti}\,(2\,\text{mii}+\text{Ti})-2\,(\text{mii}\,(\text{mii}+\text{mx})+\text{mx}\,\text{Ti})\,\delta+(\text{mii}+\text{mx}+\text{Ti})\,\delta^2+\sqrt{-\text{Ti}\,(2\,\text{mii}+\text{Ti})\left(-2\,\text{mx}\,\text{Ti}+2\,(\text{mii}+\text{mx})\,\delta+\delta^2\right)\left(2\,\text{mx}\,(\text{Ti}-\delta)-\delta^2+2\,\text{mii}\,(2\,\text{mx}+\delta)\right)}\right)\right\}\right\}$$

In[ ]:=

$$\frac{1}{2\,(\text{mii})^2+4\,\text{mx}\,\text{Ti}}\left(2\,\text{mx}\,\text{Ti}\,(2\,\text{mii}+\text{Ti})-2\,(\text{mii}\,(\text{mii})+\text{mx}\,\text{Ti})\,\delta+(\text{Ti})\,\delta^2-\sqrt{-\text{Ti}\,(2\,\text{mii}+\text{Ti})\,(-2\,\text{mx}\,\text{Ti}+2\,(\text{mii})\,\delta)\,(2\,\text{mx}\,(\text{Ti})+2\,\text{mii}\,(2\,\text{mx}+\delta))}\right)\ //\ \text{FullSimplify}$$

Out[ ]=

$$\frac{1}{2\left(\text{mii}^2+2\,\text{mx}\,\text{Ti}\right)}\left(2\,\text{mx}\,\text{Ti}\,(2\,\text{mii}+\text{Ti})-2\left(\text{mii}^2+\text{mx}\,\text{Ti}\right)\delta+\text{Ti}\,\delta^2-2\,\sqrt{\text{Ti}\,(2\,\text{mii}+\text{Ti})\,(\text{mx}\,\text{Ti}-\text{mii}\,\delta)\,(\text{mx}\,\text{Ti}+\text{mii}\,(2\,\text{mx}+\delta))}\right)$$

Define functions for kinematic limits:

In[ ]:=

$$\text{TxMin}[\text{Ti\_?NumericQ},\text{mii\_?NumericQ},\text{mx\_?NumericQ},\delta\text{\_?NumericQ}]:=$$
$$\text{If}\Big[\delta==0,0,$$
$$\frac{1}{2\,(\text{mii}+\text{mx})^2+4\,\text{mx}\,\text{Ti}}\Big(2\,\text{mx}\,\text{Ti}\,(2\,\text{mii}+\text{Ti})-2\,(\text{mii}\,(\text{mii}+\text{mx})+\text{mx}\,\text{Ti})\,\delta+(\text{mii}+\text{mx}+\text{Ti})\,\delta^2-\sqrt{-\text{Ti}\,(2\,\text{mii}+\text{Ti})}$$

In[ ]:=

$$\text{TxMax}[\text{Ti\_?NumericQ},\text{mii\_?NumericQ},\text{mx\_?NumericQ},\delta\text{\_?NumericQ}]:=$$
$$\frac{1}{2\,(\text{mii}+\text{mx})^2+4\,\text{mx}\,\text{Ti}}\Big(2\,\text{mx}\,\text{Ti}\,(2\,\text{mii}+\text{Ti})-2\,(\text{mii}\,(\text{mii}+\text{mx})+\text{mx}\,\text{Ti})\,\delta+(\text{mii}+\text{mx}+\text{Ti})\,\delta^2+\sqrt{-\text{Ti}\,(2\,\text{mii}+\text{Ti})}$$

In[ ]:=

$$\text{TxMinGlobal}[\text{mx\_},\delta\text{\_}]:=\frac{\delta^2}{2\text{mx}}$$

Minimum incoming energy to produce outgoing Tx:

In[ ]:=

$$\text{Solve}\left[\left((\text{Tip}+\text{mii})^2-\text{mii}^2\right)==\left((\text{Ti}+\text{mii})^2-\text{mii}^2\right)+\left((\text{Tx}+\text{mxp})^2-\text{mxp}^2\right)+2\sqrt{\left((\text{Ti}+\text{mii})^2-\text{mii}^2\right)\left((\text{Tx}+\text{mxp})^2-\text{mxp}^2\right)}\ /.\ \{\text{mxp}\to\text{mx}\right.$$

Out[ ]=

$$\left\{\left\{\text{Ti}\to\frac{1}{2}\left(-2\,\text{mii}+\text{Tx}+\delta+\frac{\sqrt{\text{Tx}\,(2\,\text{mx}+\text{Tx}+2\,\delta)\left(2\,\text{mx}\,\text{Tx}-\delta^2\right)\left(4\,\text{mii}^2+2\,\text{mx}\,\text{Tx}-\delta^2\right)}}{-2\,\text{mx}\,\text{Tx}+\delta^2}\right)\right\},\right.$$
$$\left.\left\{\text{Ti}\to\frac{1}{2}\left(-2\,\text{mii}+\text{Tx}+\delta+\frac{\sqrt{\text{Tx}\,(2\,\text{mx}+\text{Tx}+2\,\delta)\left(2\,\text{mx}\,\text{Tx}-\delta^2\right)\left(4\,\text{mii}^2+2\,\text{mx}\,\text{Tx}-\delta^2\right)}}{2\,\text{mx}\,\text{Tx}-\delta^2}\right)\right\}\right\}$$

In[ ]:=
$$\text{TiMin}[\text{Tx\_?NumericQ},\text{mii\_?NumericQ},\text{mx\_?NumericQ},\delta\text{\_?NumericQ}]:=\text{If}\Big[\delta>\delta\text{Max}[\text{Tx},\text{mx}],\infty,\frac{1}{2}\left(-2\ \text{mii}+\text{Tx}+\delta+\frac{\sqrt{\text{Tx}\ (2\ \text{mx}+\text{Tx}+2\ \delta)\ (2\ \text{mx}\ \text{Tx}-\delta^2)\ (4\ \text{mii}^2+2\ \text{mx}\ \text{Tx}-\delta^2)}}{2\ \text{mx}\ \text{Tx}-\delta^2}\right)\Big];$$

Find max delta:

In[ ]:=
$$\text{Solve}\Big[\big((16\ \text{mii}\ \text{mx}\ \text{Tx}-8\ \text{mx}\ \text{Tx}^2-8\ \text{mx}\ \text{Tx}\ \delta-8\ \text{mii}\ \delta^2+4\ \text{Tx}\ \delta^2+4\ \delta^3)^2-4\ (8\ \text{mx}\ \text{Tx}-4\ \delta^2)\ (-4\ \text{mii}^2\ \text{Tx}^2-8$$

Out[ ]=
$$\left\{\left\{\delta\to\frac{1}{2}(-2\ \text{mx}-\text{Tx})\right\},\left\{\delta\to-\sqrt{2}\ \sqrt{\text{mx}}\ \sqrt{\text{Tx}}\right\},\left\{\delta\to\sqrt{2}\ \sqrt{\text{mx}}\ \sqrt{\text{Tx}}\right\},\left\{\delta\to-\sqrt{2}\ \sqrt{2\ \text{mii}^2+\text{mx}\ \text{Tx}}\right\},\left\{\delta\to\sqrt{2}\ \sqrt{2\ \text{mii}^2+\text{mx}\ \text{Tx}}\right\}\right\}$$

In[ ]:=
$$\delta\text{Max}[\text{Tx\_},\text{mx\_}]:=\sqrt{2\ \text{mx}\ \text{Tx}}$$

Global Ti minimum:

In[ ]:=
$$\text{gMin}=\text{D}\Big[\frac{1}{2}\left(-2\ \text{mii}+\text{Tx}+\delta+\frac{\sqrt{\text{Tx}\ (2\ \text{mx}+\text{Tx}+2\ \delta)\ (2\ \text{mx}\ \text{Tx}-\delta^2)\ (4\ \text{mii}^2+2\ \text{mx}\ \text{Tx}-\delta^2)}}{2\ \text{mx}\ \text{Tx}-\delta^2}\right),\text{Tx}\Big]==0\text{//Solve}[\#,\text{Tx}]\&$$

Out[ ]=
$$\left\{\left\{\text{Tx}\to\frac{(2\ \text{mii}-\delta)\ \delta\ (\text{mx}+\delta)}{2\ \text{mx}\ (\text{mii}-\text{mx}-\delta)}\right\},\left\{\text{Tx}\to\frac{\delta\ (2\ \text{mii}+\delta)\ (\text{mx}+\delta)}{2\ \text{mx}\ (\text{mii}+\text{mx}+\delta)}\right\},\left\{\text{Tx}\to\frac{-2\ \text{mii}\ \delta-\delta^2}{2\ (\text{mii}-\text{mx})}\right\},\left\{\text{Tx}\to\frac{-2\ \text{mii}\ \delta+\delta^2}{2\ (\text{mii}+\text{mx})}\right\}\right\}$$

In[ ]:=
$$\frac{1}{2}\left(-2\ \text{mii}+\text{Tx}+\delta+\frac{\sqrt{\text{Tx}\ (2\ \text{mx}+\text{Tx}+2\ \delta)\ (2\ \text{mx}\ \text{Tx}-\delta^2)\ (4\ \text{mii}^2+2\ \text{mx}\ \text{Tx}-\delta^2)}}{2\ \text{mx}\ \text{Tx}-\delta^2}\right)\text{/.gMin//Simplify}[\#,\text{Assumptio}$$

Out[ ]=
$$\left\{\begin{cases}-\frac{(2\ \text{mii}-\delta)\ (2\ \text{mx}+\delta)}{2\ \text{mx}} & 2\ \text{mii}\le\delta\veebar 2\ \text{mii}\le 2\ \text{mx}+\delta\\ \frac{\text{mii}\ \delta\ (-2\ \text{mii}+\delta)}{2\ \text{mx}\ (-\text{mii}+\text{mx}+\delta)} & \text{True}\end{cases},\ \frac{\delta\ (2\ \text{mii}+2\ \text{mx}+\delta)}{2\ \text{mx}},\ \begin{cases}-2\ \text{mii} & 2\ \text{mii}>2\ \text{mx}+\delta\\ -\frac{\delta\ (2\ \text{mx}+\delta)}{2\ (\text{mii}-\text{mx})} & \text{True}\end{cases},\ -\frac{(2\ \text{mii}+2\ \text{mx}+\delta)\ (2\ \text{mii}-\delta+\text{Abs}[-2\ \text{mii}+\delta])}{4\ (\text{mii}+\text{mx})}\right\}$$

In[ ]:=
$$\text{TiMinGlobal}[\text{mii\_?NumericQ},\text{mx\_?NumericQ},\delta\text{\_?NumericQ}]:=\frac{\delta\ (2\ \text{mii}+2\ \text{mx}+\delta)}{2\ \text{mx}}$$

Max δ for a given Ti:

In[ ]:=
```
Solve[δ (2 mii+2 mx+δ)/(2 mx)==TiMi,δ]
```

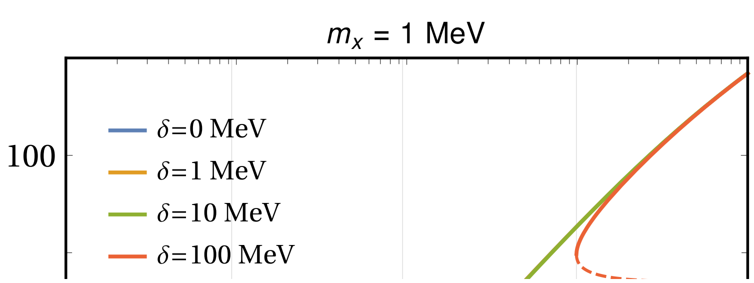
Out[ ]=
$$\left\{\left\{\delta \to -\text{mii}-\text{mx}-\sqrt{\text{mii}^2+2\,\text{mii}\,\text{mx}+\text{mx}^2+2\,\text{mx}\,\text{TiMi}}\right\},\left\{\delta \to -\text{mii}-\text{mx}+\sqrt{\text{mii}^2+2\,\text{mii}\,\text{mx}+\text{mx}^2+2\,\text{mx}\,\text{TiMi}}\right\}\right\}$$

In[ ]:=
```
δMaxTi[Ti_,mii_,mx_]:=-mii-mx+Sqrt[mii^2+2 mii mx+mx^2+2 mx Ti]
```

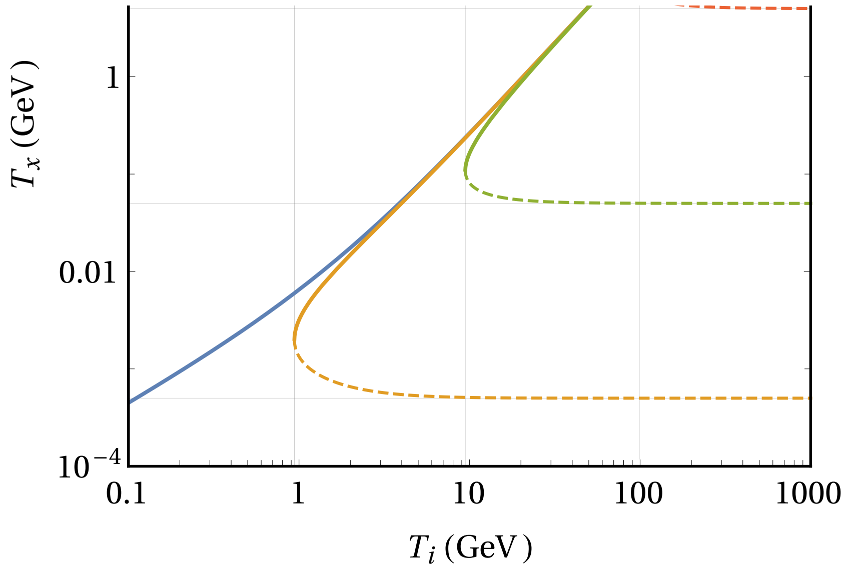
Sample phase-space plots

Plot limits of kinematic variables for some example values:

In[ ]:=
```
{Show[LogLogPlot[{TxMax[Ti, mp, .001, 0], TxMax[Ti, mp, .001, 0.001],
    TxMax[Ti, mp, .001, .01], TxMax[Ti, mp, .001, .1]}, {Ti, .1, 1000},
   FrameLabel → {"Ti (GeV)", "Tx (GeV)"}, PlotLabel → Style["mx = 1 MeV", 16, Black],
   PlotRange → {{.1, 1000}, {.0001, 1000}},
   PlotLegends →
    Legend[{"δ=0 MeV", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.2, .8}],
   GridLines → {{TiMinGlobal[mp, .001, .001],
      TiMinGlobal[mp, .001, .01], TiMinGlobal[mp, .001, .1]},
     {TxMinGlobal[.001, .001], TxMinGlobal[.001, .01], TxMinGlobal[.001, .1]}}],
  LogLogPlot[{0, TxMin[Ti, mp, .001, 0.001], TxMin[Ti, mp, .001, .01],
    TxMin[Ti, mp, .001, .1]}, {Ti, .1, 1000},
   PlotStyle → Dashed, FrameLabel → {"Ti (GeV)", "Tx (GeV)"}]],
 Show[LogLogPlot[{TxMax[Ti, mp, .1, 0], TxMax[Ti, mp, .1, 0.001], TxMax[Ti, mp, .1, .01],
    TxMax[Ti, mp, .1, .1]}, {Ti, .01, 50}, FrameLabel → {"Ti (GeV)", "Tx (GeV)"},
   PlotLabel → Style["mx = 100 MeV", 16, Black],
   PlotRange → {{.01, 10}, {.00001, 10}},
   PlotLegends →
    Legend[{"δ=0 MeV", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.2, .8}],
   GridLines → {{TiMinGlobal[mp, .1, .001], TiMinGlobal[mp, .1, .01], TiMinGlobal[mp,
        .1, .1]}, {TxMinGlobal[.1, .001], TxMinGlobal[.1, .01], TxMinGlobal[.1, .1]}}],
  LogLogPlot[{0, TxMin[Ti, mp, .1, 0.001], TxMin[Ti, mp, .1, .01],
    TxMin[Ti, mp, .1, .1]}, {Ti, .01, 50},
   PlotStyle → Dashed, FrameLabel → {"Ti (GeV)", "Tx (GeV)"}]]}
```

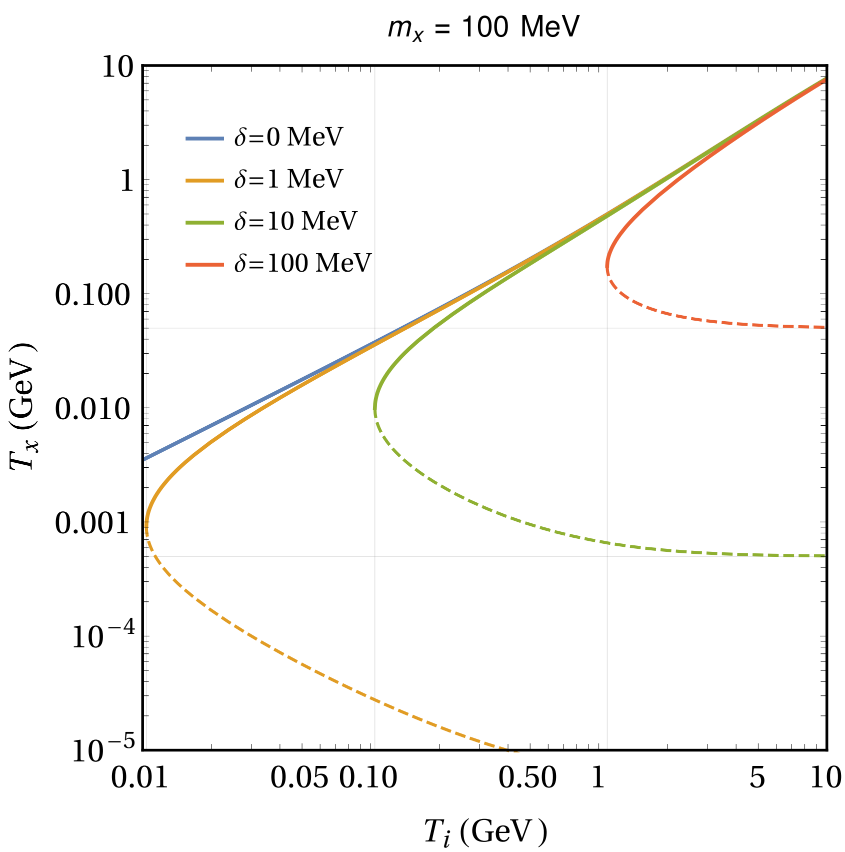


Out[•]= {



T_x (GeV)
1
0.01
10^{-4}
0.1
1
10
100
1000
T_i (GeV)

,



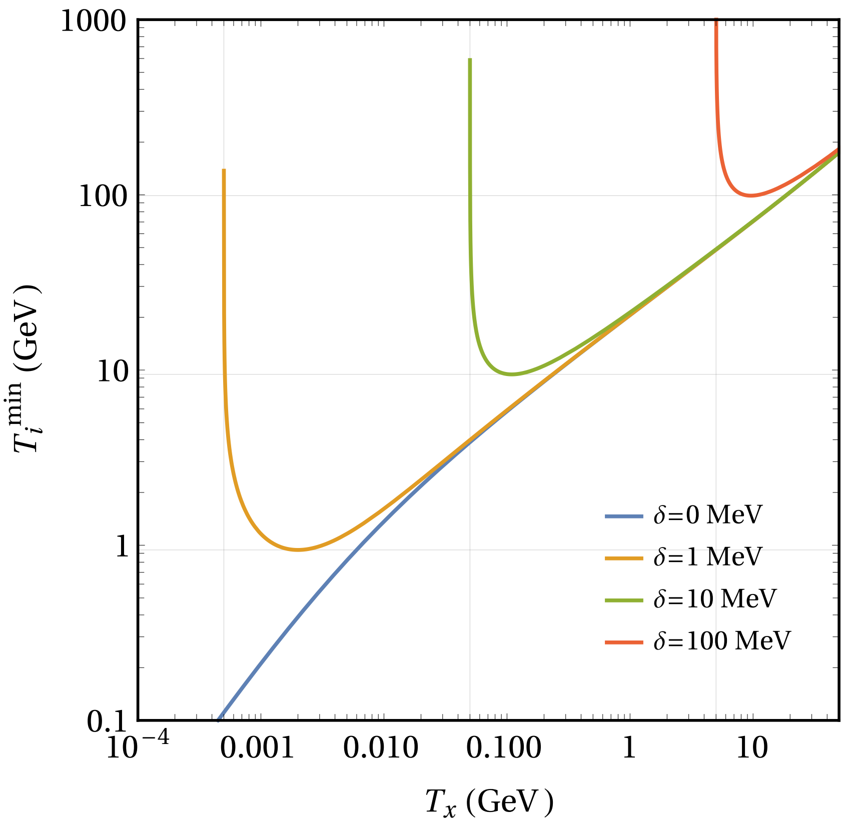
m_x = 100 MeV
δ=0 MeV
δ=1 MeV
δ=10 MeV
δ=100 MeV
T_x (GeV)
10
1
0.100
0.010
0.001
10^{-4}
10^{-5}
0.01
0.05 0.10
0.50 1
5 10
T_i (GeV)

}

In[ ]:=

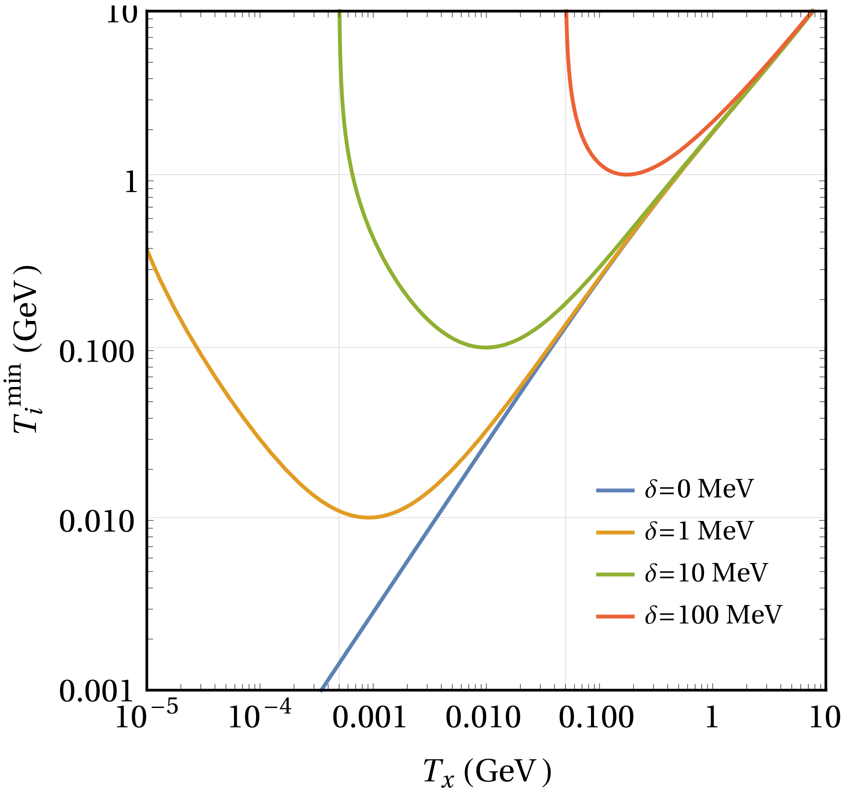
```
{LogLogPlot[{TiMin[Ti, mp, .001, 0], TiMin[Ti, mp, .001, 0.001],
    TiMin[Ti, mp, .001, .01], TiMin[Ti, mp, .001, .1]}, {Ti, .0001, 100},
  FrameLabel → {"Tx (GeV)", "Ti^min (GeV)"}, PlotLabel → Style["mx = 1 MeV", 16, Black],
  PlotRange → {{.0001, 50}, {.1, 1000}},
  PlotLegends →
   Legend[{"δ=0 MeV", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.8, .2}],
  GridLines → {{TxMinGlobal[.001, .001], TxMinGlobal[.001, .01],
     TxMinGlobal[.001, .1]}, {TiMinGlobal[mp, .001, .001],
     TiMinGlobal[mp, .001, .01], TiMinGlobal[mp, .001, .1]}}],
 LogLogPlot[{TiMin[Ti, mp, .1, 0], TiMin[Ti, mp, .1, 0.001], TiMin[Ti, mp, .1, .01],
   TiMin[Ti, mp, .1, .1]}, {Ti, .00001, 10}, FrameLabel → {"Tx (GeV)", "Ti^min (GeV)"},
  PlotLabel → Style["mx = 100 MeV", 16, Black],
  PlotRange → {{.00001, 10}, {.001, 10}},
  PlotLegends →
   Legend[{"δ=0 MeV", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.8, .2}],
  GridLines → {{TxMinGlobal[.1, .001], TxMinGlobal[.1, .01], TxMinGlobal[.1, .1]},
    {TiMinGlobal[mp, .1, .001], TiMinGlobal[mp, .1, .01], TiMinGlobal[mp, .1, .1]}}]}
```

Out[ ]= {



,

m_x = 100 MeV



}

Form factors

Helm

Standard definition of the Helm form factor:

q = momentum transfer in GeV

A = atomic number of target nuclei

In[ ]:=

```
Fhelm[q_,A_]=
```

$$\text{Piecewise}\left[\left\{\left\{3\frac{\text{Sin}\left[\frac{q\ r}{\hbar c}\right]-\left(\frac{q\ r}{\hbar c}\right)\text{Cos}\left[\frac{q\ r}{\hbar c}\right]}{\left(\frac{q\ r}{\hbar c}\right)^3}\text{Exp}\left[\frac{-\left(\frac{q\ s}{\hbar c}\right)^2}{2}\right]/.\left\{r\to\left((1.23\ A^{1/3}-.6)^2+\frac{7}{3}\pi^2(0.52)^2-5(0.9)^2\right)^{1/2},s\to0.9\right\},4>\right.\right.\right.$$

dipole form factor

Dipole form of the form factor:

In[ ]:=

$$\text{Gi}[\text{q2\_},\Lambda\text{i\_}]:=\left(1+\frac{\text{q2}}{\Lambda\text{i}^2}\right)^{-2}$$

Where Λ comes from the charge radius:

In[ ]:=

```
Λi={0.770,0.410};
```

Nuclear response functions

In[ ]:=
```
WM00[qGeV_ ,A_]:=
Switch[A,
1,0.0397887 Gi[qGeV^2,Λi⟦1⟧]^2,
4,0.31831 Gi[qGeV^2,Λi[[2]]]^2,
_,(4π/1 4)^-1 A^2 Fhelm[qGeV ,A]^2
];
```

In[ ]:=
```
FM[qGeV_ ,A_ ,jN_]:= 4π/(2jN+1) 4WM00[qGeV ,A]
```

CR-DM cross section

Reduced mass:

In[ ]:=
```
μ[m1_ ,m2_]:= m1 m2/(m1+m2)
```

Functions for NR cross section:

In[ ]:=
```
σXP[g_ ,mx_ ,mϕ_ ,kk_ :0]:= (4 g^4 μ[mx ,mp]^2)/(π mϕ^4(1+4kk^2/mϕ^2)) / Centimeter^2 / GeV^2
```

Coupling for a given cross section:

In[ ]:=
```
gg[σ_ ,mx_ ,mϕ_]:=((σ π mϕ^4)/(4 μ[mx ,mp]^2) Centimeter^2 GeV^2)^(1/4)
```

Full differential cross section (A^2 enhancement is included in the structure factor F_M):

- Tx = DM kinetic energy
- Ti = incoming CR energy
- mx = mass of dark matter
- δ = mass splitting
- gxi = to mediator coupling (takes x and nucleon coupling as the same)
- mA = mass of mediator
- iS = species of CR (1 = proton, 2 = neutron)

Units of returned differential cross section are GeV^{-3}

In[ ]:=
```
dσidTxVector [Tx_?NumericQ ,Ti_?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,gxi_?NumericQ ,mA_?NumericQ ,
((gxi^4 (4 mx (mi⟦iS⟧+Ti)^2-2 ((mi⟦iS⟧+mx)^2+2 mx Ti) Tx+2 mx Tx^2-4 mx (mi⟦iS⟧+Ti) δ+(mx-Tx) δ^2))
/(2 π Ti (2 mi⟦iS⟧+Ti) (mA^2+2 mx Tx-δ^2)^2 GeV^3))FM[Sqrt[2 mx Tx-δ^2] ,Ai⟦iS⟧,ji⟦iS⟧]
```

Define flux

Function for the upscattered χ_2 dark matter flux, contributions from protons and helium included.

Tx2 = flux is computed for the χ_2 DM kinetic energy Tx2

mx = mass of dark matter

δ = mass splitting

gxi = to mediator coupling (takes x and nucleon coupling as the same)

mA = mass of mediator

Units of returned flux are GeV^2

In[ ]:=
```
dϕX2dTx2[Tx2_?NumericQ,mx_?NumericQ,δ_?NumericQ,gxi_?NumericQ,mA_?NumericQ]:=
Deff*(ρX/(mx GeV))(
If[TxMin[TdatMax[[1]],mi[[1]],mx,δ]>Tx2,0,
NIntegrate[
pFlux[Tii]×dσidTxVector[Tx2,Tii,mx,δ,gxi,mA,1]GeV^3,
{Tii,Max[TdatMin[[1]],TiMin[Tx2,mi[[1]],mx,δ]],Max[TdatMax[[1]],TiMin[Tx2,mi[[1]],mx,δ]]},
Method→{"GlobalAdaptive","SymbolicProcessing"→0},AccuracyGoal→∞,PrecisionGoal→3]
]
+If[TxMin[TdatMax[[2]],mi[[2]],mx,δ]>Tx2,0,
NIntegrate[
heFlux[Tii]×dσidTxVector[Tx2,Tii,mx,δ,gxi,mA,2]GeV^3,
{Tii,Max[TdatMin[[2]],TiMin[Tx2,mi[[2]],mx,δ]],Max[TdatMax[[2]],TiMin[Tx2,mi[[2]],mx,δ]]},
Method→{"GlobalAdaptive","SymbolicProcessing"→0},AccuracyGoal→∞,PrecisionGoal→3]
]
)
```

When the mediator mass is small and mass splitting is large the spectrum becomes sharply peaked, this function finds that peak and does a better job sampling the spectra:

variable are same as above, but with the flux returned for nPoints sampled between {TX2MIN, TX2MAX}

In[ ]:=
```
dϕX2dTx2list [TX2MIN_ ?NumericQ ,TX2MAX_ ?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,gxi_ ?NumericQ ,mA_?N
Module [{Tx2Points ,maxF ,TMIN ,TMAX },
If[TX2MIN <TxMin [TdatMax ⟦1⟧,mi⟦1⟧,mx ,δ],TMIN =TxMin [TdatMax ⟦1⟧,mi⟦1⟧,mx ,δ]; ,TMIN =TX2MIN ];
If[TX2MAX >TxMax [TdatMax ⟦1⟧,mi⟦1⟧,mx ,δ],TMAX =TxMax [TdatMax ⟦1⟧,mi⟦1⟧,mx ,δ]; ,TMAX =TX2MAX ];
(*More careful sampling if there are sharp peaks *)
If[mA<.01&&δ>.002 ,
maxF =Txp/.(NMaximize [{dϕX2dTx2 [Txp ,mx ,δ,gxi ,mA] GeV^-2,Txp>TMIN },Txp ,PrecisionGoal →3,Method →"N
Tx2Points =Join[logSpace [TMIN ,Min[maxF ,TMAX],2nPoints /8//Round ],logSpace [maxF ,Min[400(maxF -TMIN)
,
Tx2Points =logSpace [TMIN ,TMAX ,nPoints ]];
ParallelTable [{Tx2 ,dϕX2dTx2 [Tx2 ,mx ,δ,gxi ,mA]GeV^-2},
{Tx2 ,Tx2Points }]//Return
]
```

Up-scattered DM spectra

Calculate

Choose couplings around sensitivity limit but that have round numbers:

In[ ]:= σXP[$\sqrt{.5}$, .1, 1]

Out[ ]= 1.01218×10^{-30}

In[ ]:= gxHeavy = $\sqrt{.5}$;

In[ ]:= σXP[$\sqrt{.001}$, .001, .001]

Out[ ]= 4.94718×10^{-28}

In[ ]:= gxLight = $\sqrt{.001}$;

In[ ]:=
```
datFluxVecElas = {
    ParallelTable [{Tx, dϕX2dTx2 [Tx, 0.001 , 0, gxLight , .001] Tx GeV (unitsCM2S cm^2 s)},
     {Tx, logSpace [10^-5, 100, 200]}]
,
    ParallelTable [{Tx, dϕX2dTx2 [Tx, 0.100 , 0, gxHeavy , 1.00] Tx GeV (unitsCM2S cm^2 s)},
     {Tx, logSpace [10^-5, 100, 200]}]};
```

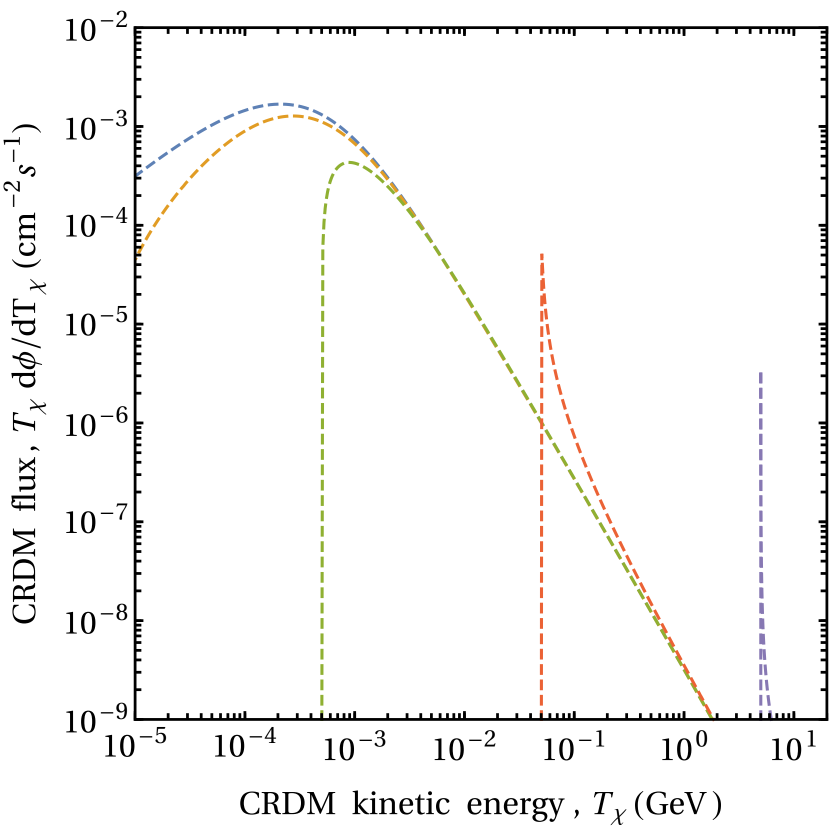
In[ ]:=
```
With[{MX = 10^-3, MA = 10^-3},

 datFluxVecM1MeVlight  = {
    ParallelTable [
{Tx, dϕX2dTx2 [Tx, MX, .0001, gxLight, MA] Tx GeV (unitsCM2S cm^2 s)},
      {Tx, logSpace [TxMin[0.99 TdatMax⟦1⟧, mi⟦1⟧, MX, .0001], 10, 400]}]
     ,
    ParallelTable [
{Tx, dϕX2dTx2 [Tx, MX, .001, gxLight, MA] Tx GeV (unitsCM2S cm^2 s)},
      {Tx, logSpace [TxMin[0.99 TdatMax⟦1⟧, mi⟦1⟧, MX, .001], 10, 400]}]
,
    ParallelTable [{Tx,
       dϕX2dTx2 [Tx, MX, 0.01, gxLight, MA] Tx GeV (unitsCM2S cm^2 s)},
      {Tx, logSpace [TxMin[0.99 TdatMax⟦1⟧, mi⟦1⟧, MX, .010], 10, 400]}]
,
    dϕX2dTx2list [.1, 10, MX, .1, gxLight, MA, 400] //
       {#[[All, 1]], #[[All, 1]] * #[[All, 2]] * unitsCM2S GeV^3 cm^2 s} & // Thread};
]
```

In[ ]:=
```
With[{MX = 10^-1, MA = 10^0},

 datFluxVecM100MeVheavy  = {
ParallelTable [
{Tx, dϕX2dTx2 [Tx, MX, .0001, gxHeavy, MA] Tx GeV (unitsCM2S cm^2 s)},
      {Tx, logSpace [TxMin[.99 TdatMax⟦1⟧, mi⟦1⟧, MX, .0001], 20, 400]}]
     ,
    ParallelTable [
{Tx, dϕX2dTx2 [Tx, MX, .001, gxHeavy, MA] Tx GeV (unitsCM2S cm^2 s)},
      {Tx, logSpace [TxMin[.99 TdatMax⟦1⟧, mi⟦1⟧, MX, .001], 20, 400]}]
,
    ParallelTable [
{Tx, dϕX2dTx2 [Tx, MX, 0.01, gxHeavy, MA] Tx GeV (unitsCM2S cm^2 s)},
      {Tx, logSpace [TxMin[.99 TdatMax⟦1⟧, mi⟦1⟧, MX, .010], 20, 400]}]
,
    ParallelTable [
{Tx, dϕX2dTx2 [Tx, MX, 0.10, gxHeavy, MA] Tx GeV (unitsCM2S cm^2 s)},
      {Tx, logSpace [TxMin[.999 TdatMax⟦1⟧, mi⟦1⟧, MX, .100], 20, 400]}]};
]
```

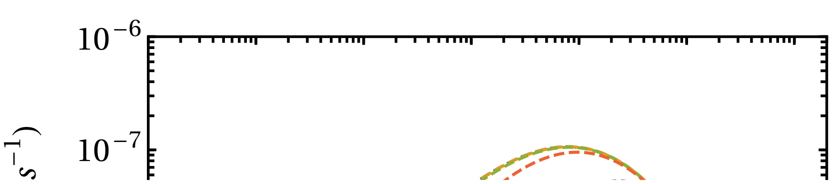
Plot spectra

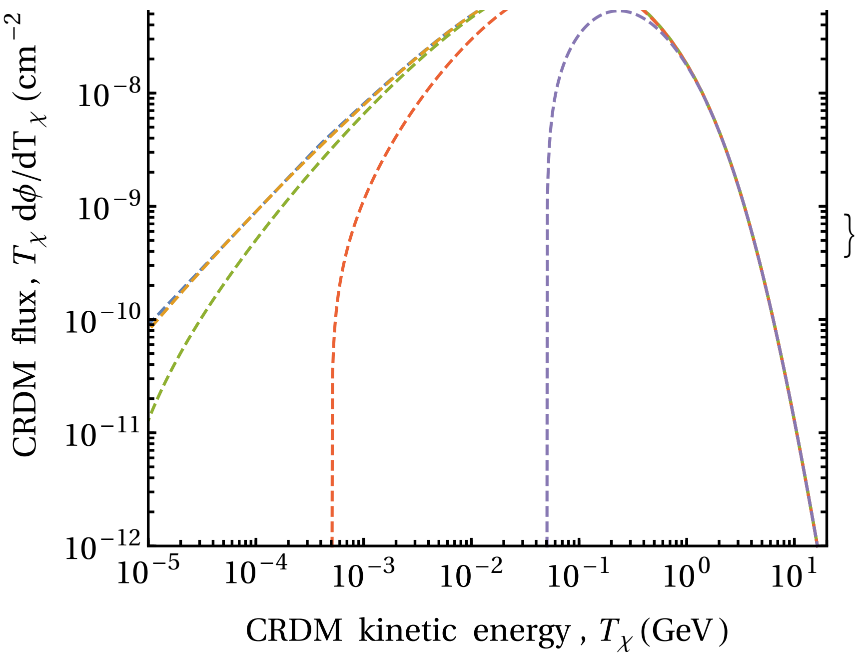
```
commonOptions ={Joined →True ,
 FrameTicks →{LogTicks [10^-16,1],LogTicks [10^-6,10^2]},
 PlotRange →{{10^-5,20},{10^-9,10^-2}},
 FrameLabel →{"CRDM  kinetic  energy , Tχ (GeV)","CRDM  flux , Tχ dϕ/dTχ (cm^-2 s^-1)"}};
```

```
inelasFluxPlot1MeVlight  = ListLogLogPlot [
    Join[{datFluxVecElas 〚1〛}, datFluxVecM1MeVlight ], PlotStyle → Dashed ,
    commonOptions ];
inelasFluxPlot100MeVheavy  = ListLogLogPlot [
    Join[{datFluxVecElas [[2]]}, datFluxVecM100MeVheavy ], PlotStyle → Dashed ,
    PlotRange → {{10^-5, 20}, {10^-12, 10^-6}}, commonOptions ];
{inelasFluxPlot1MeVlight  , inelasFluxPlot100MeVheavy }
```

{ 

,





DM decay

This section describes the decay of the upscattered χ_2 particles to χ_1+ γ and calculates the resulting χ_1 and γ spectra

Kinematics

CoM decay kinematics with 1 massless particle:

Px == px' + pγ

Ei == M + Tx

$$Ex == \frac{M^2 + mx^2}{2\,M}$$

$$E\gamma == \frac{M^2 - mx^2}{2\,M}$$

$$pxCoM == \frac{\sqrt{(M^2 + mx^2)^2 - 4\,M^2\,mx^2}}{2\,M} == \frac{M^2 - mx^2}{2\,M}$$

$$p\gamma CoM == E\gamma == \frac{M^2 - mx^2}{2\,M}$$

Take M = mx + δ and break into components, along and perpendicular to the parent particle's direction:

$$pxCoMi == \frac{(mx + \delta)^2 - mx^2}{2\,(mx + \delta)}\,Sin\theta d$$

$$\text{pxCoMj} == \frac{(\text{mx}+\delta)^2-\text{mx}^2}{2\,(\text{mx}+\delta)}\,\text{Cos}\theta\text{d}$$

$$\text{p}\gamma\text{CoMi} == -\frac{(\text{mx}+\delta)^2-\text{mx}^2}{2\,(\text{mx}+\delta)}\,\text{Sin}\theta\text{d}$$

$$\text{p}\gamma\text{CoMj} == -\frac{(\text{mx}+\delta)^2-\text{mx}^2}{2\,(\text{mx}+\delta)}\,\text{Cos}\theta\text{d}$$

Boosting, γ = Ei/M:

$$\text{pxi} == \frac{\delta\,(2\,\text{mx}+\delta)}{2\,(\text{mx}+\delta)}\,\text{Sin}\theta\text{d}$$

$$\text{pxj} == \gamma\left(\frac{\delta\,(2\,\text{mx}+\delta)}{2\,(\text{mx}+\delta)}\,\text{Cos}\theta\text{d}+\beta[\gamma]\,\text{Ex}\right) ==$$

$$\frac{\text{mx}+\delta+\text{Tx}}{\text{mx}+\delta}\left(\frac{\delta\,(2\,\text{mx}+\delta)}{2\,(\text{mx}+\delta)}\,\text{Cos}\theta\text{d}+\beta\left[\frac{\text{mx}+\delta+\text{Tx}}{\text{mx}+\delta}\right]\frac{(\text{mx}+\delta)^2+\text{mx}^2}{2\,(\text{mx}+\delta)}\right)$$

$$\text{TxB} = \sqrt{\text{px}^2+\text{mx}^2}-\text{mx} == \sqrt{\text{pxi}^2+\text{pxj}^2+\text{mx}^2}-\text{mx}$$

Define energy of outgoing DM after decay:

In[ ]:=
```
β[γ_]:=(1-γ^-2)^(1/2)
```

In[ ]:=
```
Tx1[Tx2_,mx1_,δ_,cosθd_]=
```

$$\sqrt{\left(\frac{\delta^2\ (2\ \text{mx1}+\delta)^2}{4(\text{mx1}+\delta)^2}\text{Sin}[\text{ArcCos}[\text{cos}\theta\text{d}]]^2+\left(\frac{\text{mx1}+\delta+\text{Tx2}}{\text{mx1}+\delta}\left(\frac{\delta\ (2\ \text{mx1}+\delta)}{2\ (\text{mx1}+\delta)}\text{cos}\theta\text{d}+\beta\left[\frac{\text{mx1}+\delta+\text{Tx2}}{\text{mx1}+\delta}\right]\frac{(\text{mx1}+\delta)^2+\text{mx1}^2}{2(\text{mx1}+\delta)}\right)\right)^2+\text{mx1}^2}$$

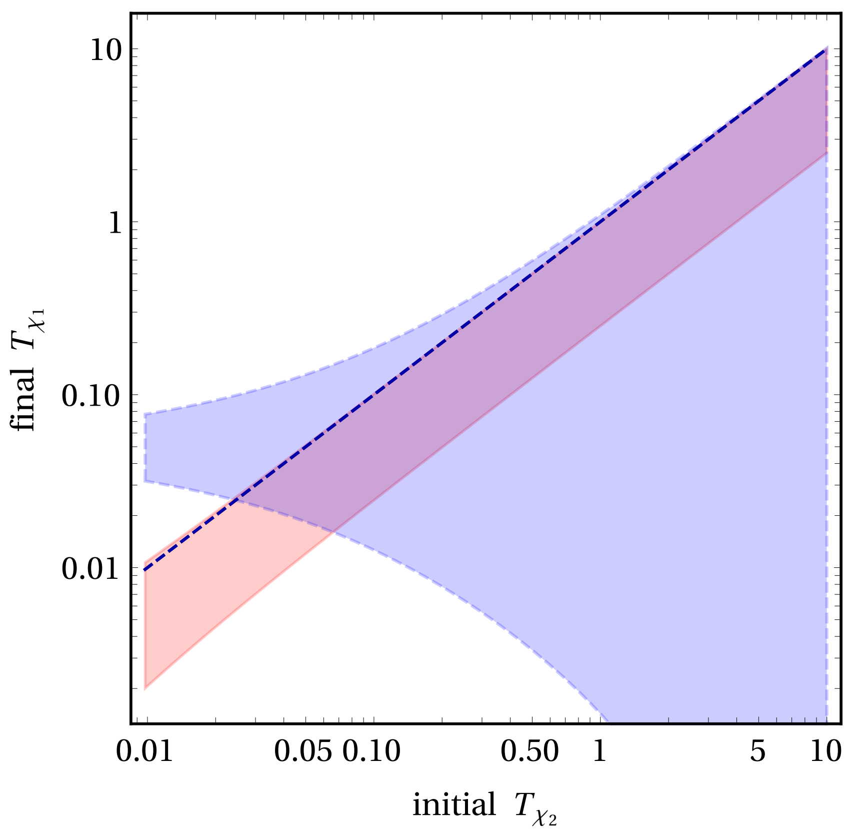
Out[ ]=

$$-\text{mx1}+\sqrt{\left(\text{mx1}^2-\frac{\left(-1+\text{cos}\theta\text{d}^2\right)\delta^2\,(2\,\text{mx1}+\delta)^2}{4\,(\text{mx1}+\delta)^2}+\frac{(\text{mx1}+\text{Tx2}+\delta)^2\left(\text{cos}\theta\text{d}\ \delta\,(2\,\text{mx1}+\delta)+\sqrt{\frac{\text{Tx2}\,(2\,\text{mx1}+\text{Tx2}+2\,\delta)}{(\text{mx1}+\text{Tx2}+\delta)^2}}\left(\text{mx1}^2+(\text{mx1}+\delta)^2\right)\right)^2}{4\,(\text{mx1}+\delta)^4}\right)}$$

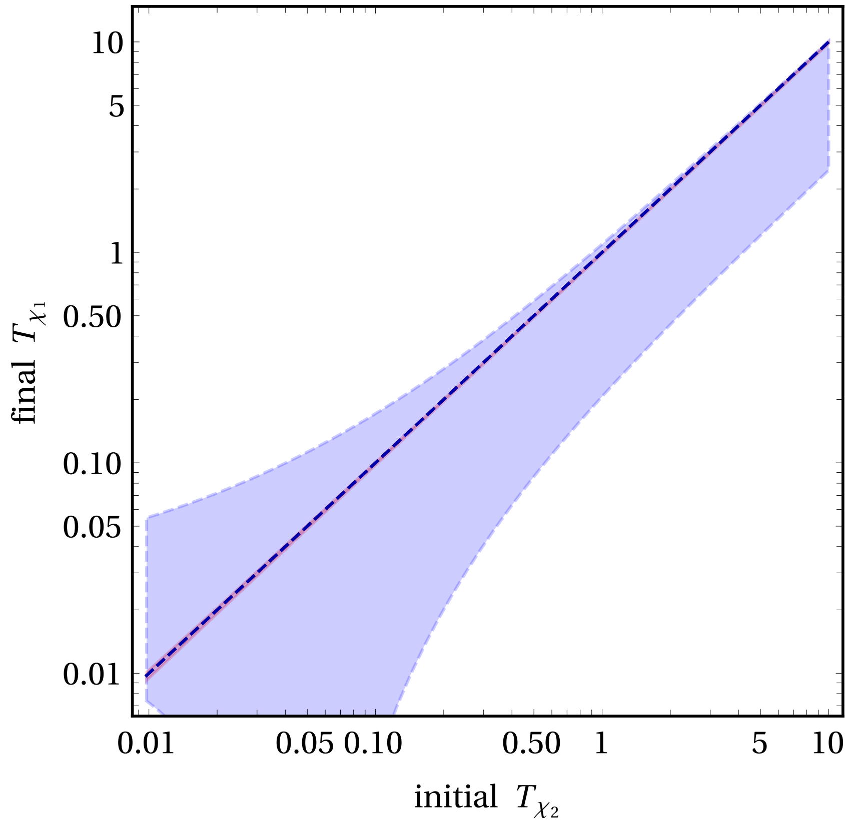
Different colors are different δ

In[ ]:=
```
{Show[
  LogLogPlot[{TX, Tx1[TX, .001, .001, -1], Tx1[TX, .001, .001, 1]}, {TX, 0, 10},
   PlotStyle → {{Dashed, Red // Darker}, None, None}, Filling → 2 → {3},
   FillingStyle → Opacity[.2, Red], FrameLabel → {"initial Tχ₂", "final Tχ₁"}],
  LogLogPlot[{TX, Tx1[TX, .001, .1, -1], Tx1[TX, .001, .1, 1]},
   {TX, 0, 10}, PlotStyle → {{Dashed, Blue // Darker}, None, None},
   Filling → 2 → {3}, FillingStyle → Opacity[.2, Blue]]],
 Show[
  LogLogPlot[{TX, Tx1[TX, .1, .001, -1], Tx1[TX, .1, .001, 1]}, {TX, 0, 10},
   PlotStyle → {{Dashed, Red // Darker}, None, None}, Filling → 2 → {3},
   FillingStyle → Opacity[.2, Red], FrameLabel → {"initial Tχ₂", "final Tχ₁"}],
  LogLogPlot[{TX, Tx1[TX, .1, .1, -1], Tx1[TX, .1, .1, 1]}, {TX, 0, 10},
   PlotStyle → {{Dashed, Blue // Darker}, None, None},
   Filling → 2 → {3}, FillingStyle → Opacity[.2, Blue]]]}
```

Out[ ]= {



,



}

In[ ]:=
```
Solve[TXb==γ(mx+δ^2/(2 (mx+δ)))+γ β[γ] Sqrt[(2 mx δ^2/(2 (mx+δ))+(δ^2/(2 (mx+δ)))^2)] cosθ-mx/.γ→(mx+δ+Tx)/(mx+δ),Tx]//FullSimpl
```

Out[ ]=

$$\Big\{\Big\{Tx \to \Big(\cos\theta^2\,\delta^2\,(mx+\delta)\,(2\,mx+\delta)^2 + (mx+\delta)\,(2\,mx\,TXb + (2\,TXb-\delta)\,\delta)\,\big(2\,mx^2 + 2\,mx\,\delta + \delta^2\big) + \sqrt{\big(\cos\theta^2\,\delta^2\,(mx+\delta)^2\,(2\,mx+\delta)^2\,\big(4\,mx^2\,TXb\,(2\,mx+TXb) + 8\,mx\,TXb\,(2\,mx+TXb)\,\delta + 4\,\big(\big(-1+\cos\theta^2\big)\,mx^2 + 2\,mx\,TXb + TXb^2\big)\,\delta^2 + 4\,\big(-1+\cos\theta^2\big)\,mx\,\delta^3 + \big(-1+\cos\theta^2\big)\,\delta^4\big)\big)}\Big) \Big/ \Big(\big(2\,mx^2 - 2\,(-1+\cos\theta)\,mx\,\delta - (-1+\cos\theta)\,\delta^2\big)\,\big(2\,mx^2 + 2\,(1+\cos\theta)\,mx\,\delta + (1+\cos\theta)\,\delta^2\big)\Big)\Big\},$$

$$\Big\{Tx \to \Big(\cos\theta^2\,\delta^2\,(mx+\delta)\,(2\,mx+\delta)^2 + (mx+\delta)\,(2\,mx\,TXb + (2\,TXb-\delta)\,\delta)\,\big(2\,mx^2 + 2\,mx\,\delta + \delta^2\big) - \sqrt{\big(\cos\theta^2\,\delta^2\,(mx+\delta)^2\,(2\,mx+\delta)^2\,\big(4\,mx^2\,TXb\,(2\,mx+TXb) + 8\,mx\,TXb\,(2\,mx+TXb)\,\delta + 4\,\big(\big(-1+\cos\theta^2\big)\,mx^2 + 2\,mx\,TXb + TXb^2\big)\,\delta^2 + 4\,\big(-1+\cos\theta^2\big)\,mx\,\delta^3 + \big(-1+\cos\theta^2\big)\,\delta^4\big)\big)}\Big) \Big/ \Big(\big(2\,mx^2 - 2\,(-1+\cos\theta)\,mx\,\delta - (-1+\cos\theta)\,\delta^2\big)\,\big(2\,mx^2 + 2\,(1+\cos\theta)\,mx\,\delta + (1+\cos\theta)\,\delta^2\big)\Big)\Big\}\Big\}$$

Initial Tx2 (in lab frame) to achieve a given boosted Tx1:

In[ ]:=
```
Tx2[Tx1_,mx1_,δ_,cosθ_]:=
(cosθ^2 δ^2 (mx1+δ) (2 mx1+δ)^2+(mx1+δ) (2 mx1 Tx1+(2 Tx1-δ) δ) (2 mx1^2+2 mx1 δ+δ^2)-√(cosθ^2 δ^2 (mx1+δ)
```

Minimum/maximum initial energy Tx2 that can be boosted to a final Tx1:

In[ ]:= Solve[Tx1[TX2, mx1, δ, 1] == TX1, TX2] // FullSimplify

Out[ ]=
$$\left\{\left\{TX2 \to -\frac{mx1\,(2\,mx1+\delta)^2 + \sqrt{TX1\,(2\,mx1+TX1)\,\delta^2\,(2\,mx1+\delta)^2} + TX1\,(2\,mx1^2 + 2\,mx1\,\delta + \delta^2)}{2\,mx1^2}\right\},\right.$$
$$\left\{TX2 \to \frac{2\,mx1^2\,TX1 + 2\,mx1\,TX1\,\delta + (mx1+TX1)\,\delta^2 - \sqrt{TX1\,(2\,mx1+TX1)\,\delta^2\,(2\,mx1+\delta)^2}}{2\,mx1^2}\right\},$$
$$\left\{TX2 \to \frac{-mx1\,(2\,mx1+\delta)^2 + \sqrt{TX1\,(2\,mx1+TX1)\,\delta^2\,(2\,mx1+\delta)^2} - TX1\,(2\,mx1^2 + 2\,mx1\,\delta + \delta^2)}{2\,mx1^2}\right\},$$
$$\left.\left\{TX2 \to \frac{2\,mx1^2\,TX1 + 2\,mx1\,TX1\,\delta + (mx1+TX1)\,\delta^2 + \sqrt{TX1\,(2\,mx1+TX1)\,\delta^2\,(2\,mx1+\delta)^2}}{2\,mx1^2}\right\}\right\}$$

In[ ]:=
$$\text{Tx2min}[\text{Tx1\_},\text{mx\_},\delta\_] := \frac{2\,mx^2\,Tx1 + 2\,mx\,Tx1\,\delta + (mx+Tx1)\,\delta^2 - \sqrt{Tx1\,(2\,mx+Tx1)\,\delta^2\,(2\,mx+\delta)^2}}{2\,mx^2}$$
$$\text{Tx2max}[\text{Tx1\_},\text{mx\_},\delta\_] := \frac{2\,mx^2\,Tx1 + 2\,mx\,Tx1\,\delta + (mx+Tx1)\,\delta^2 + \sqrt{Tx1\,(2\,mx+Tx1)\,\delta^2\,(2\,mx+\delta)^2}}{2\,mx^2}$$

Decay photon

Energy of photon in detector frame:

In[ ]:=
$$\text{E}\gamma\text{b}[\text{Tx\_},\text{mx\_},\delta\_,\cos\theta\_] := \gamma\,\frac{\delta\,(2\,mx+\delta)}{2\,(mx+\delta)}(1-\beta[\gamma]\cos\theta)\,/.\,\gamma \to \frac{mx+\delta+Tx}{mx+\delta}$$

Minimum Tx to give Eγ in detector frame:

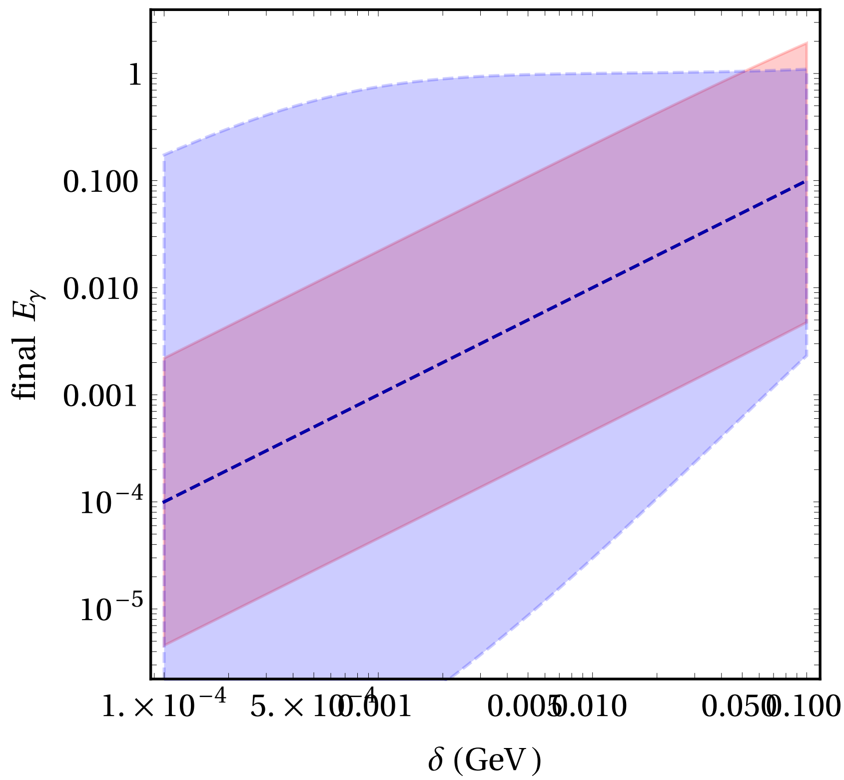
In[ ]:= Solve[Eγb[Tx, mx, δ, -1] == Eγ, Tx] // FullSimplify

Out[ ]=
$$\left\{\left\{Tx \to \frac{(-2\,E\gamma\,(mx+\delta) + \delta\,(2\,mx+\delta))^2}{4\,E\gamma\,\delta\,(2\,mx+\delta)}\right\}\right\}$$

In[ ]:=
$$\text{TxMinE}\gamma[\text{E}\gamma\text{\_?NumericQ},\text{mx\_?NumericQ},\delta\text{\_?NumericQ}] := \frac{(-2\,E\gamma\,(mx+\delta)+\delta\,(2\,mx+\delta))^2}{4\,E\gamma\,\delta\,(2\,mx+\delta)}$$

In[ ]:=
```
Show[LogLogPlot[{δ, Eγb[10, 1, δ, -1], Eγb[10, 1, δ, 1]}, {δ, .0001, .1},
  PlotStyle → {{Dashed, Red // Darker}, None, None}, Filling → 2 → {3},
  FillingStyle → Opacity[.2, Red], FrameLabel → {"δ (GeV)", "final Eγ"}],
 LogLogPlot[{δ, Eγb[1, .001, δ, -1], Eγb[1, .001, δ, 1]}, {δ, .0001, .1},
  PlotStyle → {{Dashed, Blue // Darker}, None, None},
  Filling → 2 → {3}, FillingStyle → Opacity[.2, Blue]]]
```

Out[ ]=



Define post decay flux

Function for the upscattered χ_1 dark matter flux after decay from χ_2, contributions from protons and helium included.

Tx1 = flux is computed for the χ_1 DM kinetic energy Tx1

mx = mass of dark matter

δ = mass splitting

gxi = to mediator coupling (takes x and nucleon coupling as the same)

mA = mass of mediator

Units of returned flux are GeV^2

In[∘]:=
```
dϕX1dTx1[TX1_?NumericQ, mx_?NumericQ, δ_?NumericQ, gxi_?NumericQ, mA_?NumericQ]:=
Deff*(ρX/(mx GeV))(
If[(TX1<Tx1[TxMin[TdatMax[[1]],mi[[1]],mx,δ],mx,δ,-1] ∨ TiMinGlobal[mi[[1]],mx,δ]>TdatMax[[1]]),10^-100,
NIntegrate[
pFlux[Tii]×dσidTxVector[TX2,Tii,mx,δ,gxi,mA,1]((δ (2 mx+δ) Sqrt[TX2 (2 mx+TX2+2 δ)])/(mx+δ)^2)^-1 GeV,
{TX2,Tx2min[TX1,mx,δ],Tx2max[TX1,mx,δ]},
{Tii,Max[TdatMin[[1]],Min[TiMin[TX2,mi[[1]],mx,δ],TdatMax[[1]]]],TdatMax[[1]]},AccuracyGoal→∞,Method→{'
]
+If[(TX1<Tx1[TxMin[TdatMax[[2]],mi[[2]],mx,δ],mx,δ,-1] ∨ TiMinGlobal[mi[[2]],mx,δ]>TdatMax[[2]]),10^-100,
NIntegrate[
heFlux[Tii]×dσidTxVector[TX2,Tii,mx,δ,gxi,mA,2]((δ (2 mx+δ) Sqrt[TX2 (2 mx+TX2+2 δ)])/(mx+δ)^2)^-1 GeV,
{TX2,Tx2min[TX1,mx,δ],Tx2max[TX1,mx,δ]},
{Tii,Max[TdatMin[[2]],Min[TiMin[TX2,mi[[2]],mx,δ],TdatMax[[2]]]],TdatMax[[2]]},AccuracyGoal→∞,Method→{'
]
)
```

Photon flux:

variables are same as above except Eγ is the photon energy in GeV

In[∘]:=
```
dϕγdE[Eγ_?NumericQ, mx_?NumericQ, δ_?NumericQ, gxi_?NumericQ, mϕ_?NumericQ]:=
If[Eγ>Eγb[10,mx,δ,-1]∨Eγ<Eγb[10,mx,δ,1],0,
(((mx+δ)^2-mx^2)/(2(mx+δ)))^-1 Deff*(ρX/(mx GeV))×
Sum[
NIntegrate[
dϕdT[Tii,ii]×dσidTxVector[TX,Tii,mx,δ,gxi,mϕ,ii]((2 Sqrt[((TX+mx+δ)^2-(mx+δ)^2)])/(mx+δ))^-1 GeV,
{TX,TxMinEγ[Eγ,mx,δ],100},
{Tii,Min[TiMin[100,mi[[ii]],mx,δ],TdatMax[[ii]]],TdatMax[[ii]]},AccuracyGoal→∞,PrecisionGoal→3]
,{ii,1,2}]]
```

Define function which returns of list of samples of the χ_1 spectra after decay

variable are same as above, but with the flux returned for nPoints (log sampled) between {TX1MIN, TX1MAX}

In[•]:=
```
dϕX1dTx1list [TX1MIN_ ?NumericQ ,TX1MAX_ ?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,gxi_?NumericQ ,mA_?N
Module[{fluxX2 ,Tx2Points ,maxF},
fluxX2 =dϕX2dTx2list [Tx2min[TX1MIN ,mx ,δ],Min[Tx2max[TX1MAX ,mx ,δ],1000],mx ,δ,gxi ,mA ,2nPoints ]//I
ParallelTable [
{TX1 , If[(TX1<Tx1[fluxX2 [[1,1,1]],mx ,δ,-1]),10^-100,
  NIntegrate [
   fluxX2 [TX2]((δ (2 mx+δ) Sqrt[TX2(2 mx+TX2+2 δ)])/(mx+δ)^2)^-1,
   {TX2 ,Min[Max[Tx2min[TX1 ,mx ,δ],fluxX2 [[1,1,1]]],Tx2max [TX1 ,mx ,δ]],Min[fluxX2 [[1,1,2]],Tx2max [TX1 ,m
   AccuracyGoal →∞,PrecisionGoal →3,Method→{Automatic ,"SymbolicProcessing "→0}]]}
,{TX1 ,logSpace [TX1MIN ,TX1MAX ,nPoints ]}]//Return
];
```

Define function which returns of list of samples of the γ spectra

variable are same as above, but with the flux returned for nPoints (log sampled) between {EMIN, EMAX}

In[•]:=
```
dϕγdElist[EMIN_?NumericQ ,EMAX_?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,gxi_?NumericQ ,mA_?NumericQ
Module[{fluxX2 ,TxPoints },
fluxX2 =
dϕX2dTx2list [1.00001 TxMin[TdatMax [[1]],mi[[1]],mx ,δ],Min[TxMax[TdatMax [[1]],mi[[1]],mx ,δ],1000],mx ,δ,gx
//Interpolation [#,InterpolationOrder →1]&;
ParallelTable [{Eγ,
(((mx+δ)^2-mx^2)/(2(mx+δ)))^-1 GeV^2 ×
NIntegrate [
   fluxX2 [TX]((2 Sqrt[((TX+mx+δ)^2-(mx+δ)^2)])/(mx+δ))^-1,
   {TX ,Max[fluxX2 [[1,1,1]],TxMinEγ [Eγ ,mx ,δ]],Max[TxMinEγ [Eγ ,mx ,δ],fluxX2 [[1,1,2]]]},
   AccuracyGoal →∞,PrecisionGoal →3]},
{Eγ ,logSpace [EMIN ,EMAX ,nPoints ]}]//Return
];
```

Calculate fluxes

CRDM flux

```
With[{MX = .001, MA = .001, nPoints = 200},
  datDecayDMfluxM1MeVδp1MeV =
    dϕX1dTx1list[10^-5, 10, MX, .0001, gxLight, MA, nPoints];
  datDecayDMfluxM1MeVδp1MeV[[All, 2]] *=
    datDecayDMfluxM1MeVδ1MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;

datDecayDMfluxM1MeVδ1MeV = dϕX1dTx1list[10^-5, 10, MX, .001, gxLight, MA, nPoints];
  datDecayDMfluxM1MeVδ1MeV[[All, 2]] *=
    datDecayDMfluxM1MeVδ1MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;

datDecayDMfluxM1MeVδ10MeV = dϕX1dTx1list[10^-5, 10, MX, .01, gxLight, MA, nPoints];
  datDecayDMfluxM1MeVδ10MeV[[All, 2]] *=
    datDecayDMfluxM1MeVδ10MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;

datDecayDMfluxM1MeVδ100MeV = dϕX1dTx1list[.01, 10, MX, .1, gxLight, MA, 2 nPoints];
  datDecayDMfluxM1MeVδ100MeV[[All, 2]] *=
    datDecayDMfluxM1MeVδ100MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;
]
```

```
With[{MX = .1, MA = 1, nPoints = 200},
 datDecayDMfluxM100MeVδp1MeV =
  dϕX1dTx1list[10^-5, 20, MX, .0001, gxHeavy, MA, nPoints];
 datDecayDMfluxM100MeVδp1MeV[[All, 2]] *=
  datDecayDMfluxM100MeVδ1MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;

datDecayDMfluxM100MeVδ1MeV = dϕX1dTx1list[10^-5, 20, MX, .001, gxHeavy, MA, nPoints];
 datDecayDMfluxM100MeVδ1MeV[[All, 2]] *=
  datDecayDMfluxM100MeVδ1MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;

datDecayDMfluxM100MeVδ10MeV = dϕX1dTx1list[10^-5, 20, MX, .01, gxHeavy, MA, nPoints];
 datDecayDMfluxM100MeVδ10MeV[[All, 2]] *=
  datDecayDMfluxM100MeVδ10MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;

datDecayDMfluxM100MeVδ100MeV = dϕX1dTx1list[10^-4, 20, MX, .1, gxHeavy, MA, nPoints];
 datDecayDMfluxM100MeVδ100MeV[[All, 2]] *=
  datDecayDMfluxM100MeVδ100MeV[[All, 1]] unitsCM2S GeV^3 cm^2 s;
]
```

Photon flux

In[ ]:=

```
datDecayγFluxM1MeV =
With[{MX = .001, MA = .001, EMIN = 10^-3, EMAX = 10, nPoints = 200},

   {dϕγdElist[EMIN, EMAX, MX, 0.0001, gxLight, MA, nPoints],
    dϕγdElist[EMIN, EMAX, MX, 0.001, gxLight, MA, nPoints],
    dϕγdElist[EMIN, EMAX, MX, 0.010, gxLight, MA, nPoints],
    dϕγdElist[EMIN, EMAX, MX, 0.100, gxLight, MA, 2 nPoints]}];
```

```
datDecayγFluxM100MeV =
With[{MX = .1, MA = 1, EMIN = 10^-3, EMAX = 10, nPoints = 200},

   {dϕγdElist[EMIN, EMAX, MX, 0.0001, gxHeavy, MA, nPoints],
    dϕγdElist[EMIN, EMAX, MX, 0.001, gxHeavy, MA, nPoints],
    dϕγdElist[EMIN, EMAX, MX, 0.010, gxHeavy, MA, nPoints],
    dϕγdElist[EMIN, EMAX, MX, 0.100, gxHeavy, MA, nPoints]}];
```

Fraction of flux from $b > 30^o$ is ~0.19 (see CR notebook), however IGRB is 80% accounted for, so can approximate limit based on equating total decay γ flux to the total IGRB flux.
Convert units to those used by Fermi data $\left(\mathrm{GeV}/\mathrm{cm}^2/s/\mathrm{sr}\right)$:

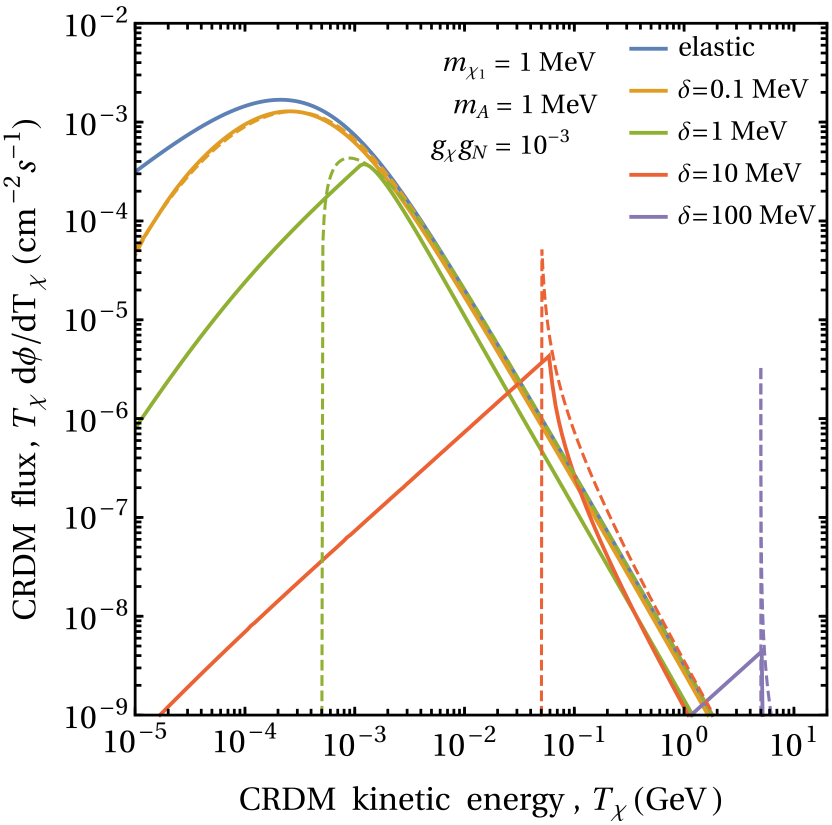
In[ ]:=
```
Do[datDecayγFluxM1MeV[[ii, All, 2]] *=
  1/(4 π) GeV (unitsCM2S cm^2 s) datDecayγFluxM1MeV[[ii, All, 1]]^2, {ii, 1, 4}];
```

In[ ]:=
```
Do[datDecayγFluxM100MeV[[ii, All, 2]] *=
  1/(4 π) GeV (unitsCM2S cm^2 s) datDecayγFluxM100MeV[[ii, All, 1]]^2, {ii, 1, 4}];
```

Dark matter flux plots

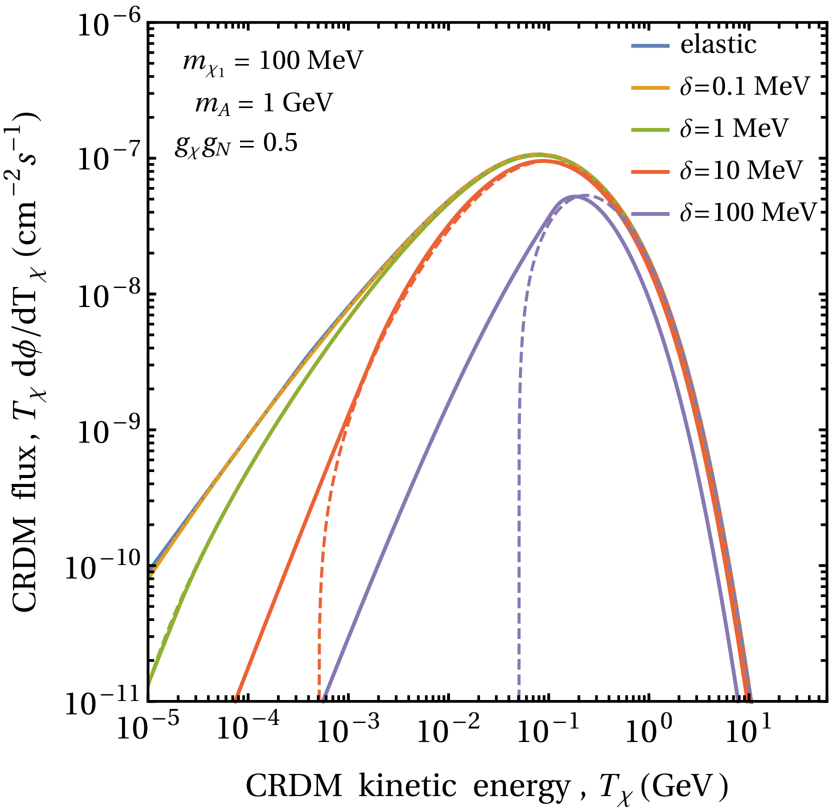
In[•]:=
```
fluxPlot1MeVlight = Show[
  ListLogLogPlot[
   {datFluxVecElas⟦1⟧, datDecayDMfluxM1MeVδp1MeV , datDecayDMfluxM1MeVδ1MeV ,
    datDecayDMfluxM1MeVδ10MeV , datDecayDMfluxM1MeVδ100MeV },
   Joined → True, commonOptions ,
   Epilog → Inset[Style[" mχ1 = 1 MeV\n    mA = 1 MeV\ngχgN = 10^-3      ",
      14, FontFamily → "Times"], Scaled[{.55, .88}]],
   PlotLegends → Legend[{"elastic", "δ=0.1 MeV", "δ=1 MeV",
      "δ=10 MeV", "δ=100 MeV"}, Position → {.85, .84}]],
  inelasFluxPlot1MeVlight ]
```

Out[•]=



In[ ]:=
```
fluxPlot100MeVheavy = Show[
  ListLogLogPlot[
   {datFluxVecElas[[2]], datDecayDMfluxM100MeVδp1MeV, datDecayDMfluxM100MeVδ1MeV,
    datDecayDMfluxM100MeVδ10MeV, datDecayDMfluxM100MeVδ100MeV},
   Joined → True, PlotRange → {{10^-5, 60}, {10^-11, 10^-6}}, commonOptions,
   Epilog → Inset[Style["    mχ1 = 100 MeV\nmA = 1 GeV\ngχgN = 0.5          ",
      14, FontFamily → "Times"], Scaled[{.17, .88}]],
   PlotLegends → Legend[{"elastic", "δ=0.1 MeV", "δ=1 MeV",
      "δ=10 MeV", "δ=100 MeV"}, Position → {.85, .84}]],
  inelasFluxPlot100MeVheavy
 ]
```

Out[ ]=



In[ ]:=
```
Export["figures/fig_flux_1MeV.pdf", fluxPlot1MeVlight];
Export["figures/fig_flux_100MeV.pdf", fluxPlot100MeVheavy];
```

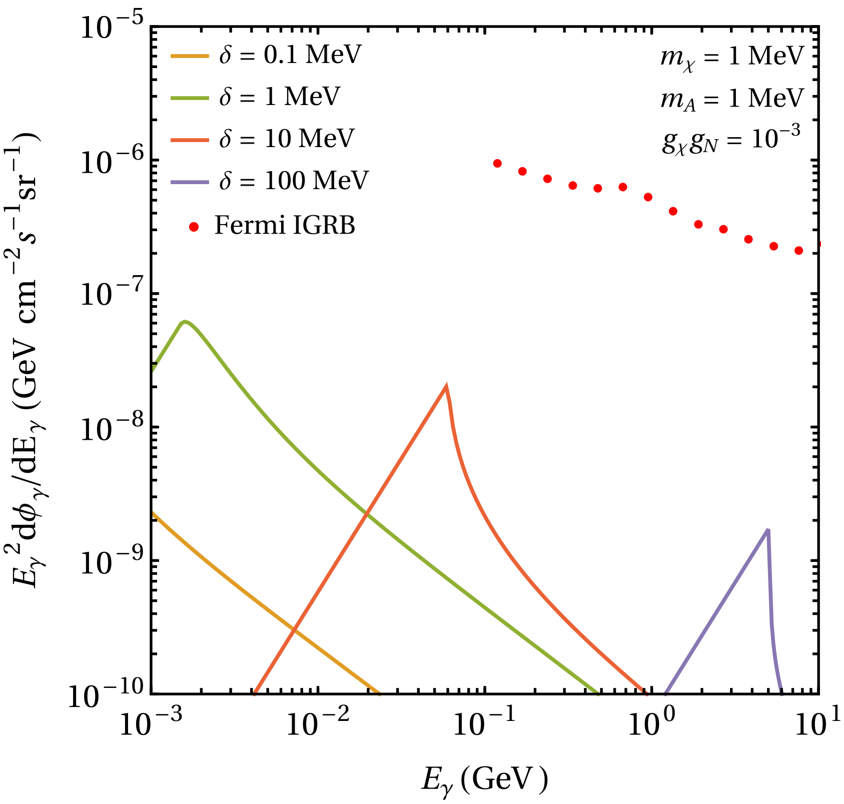
Photon flux plot

Import Fermi data from fig.5 of 1501.05464:

In[ ]:=
```
fermiIGRB = Import["data/FERMI_IGRB.csv"];
```

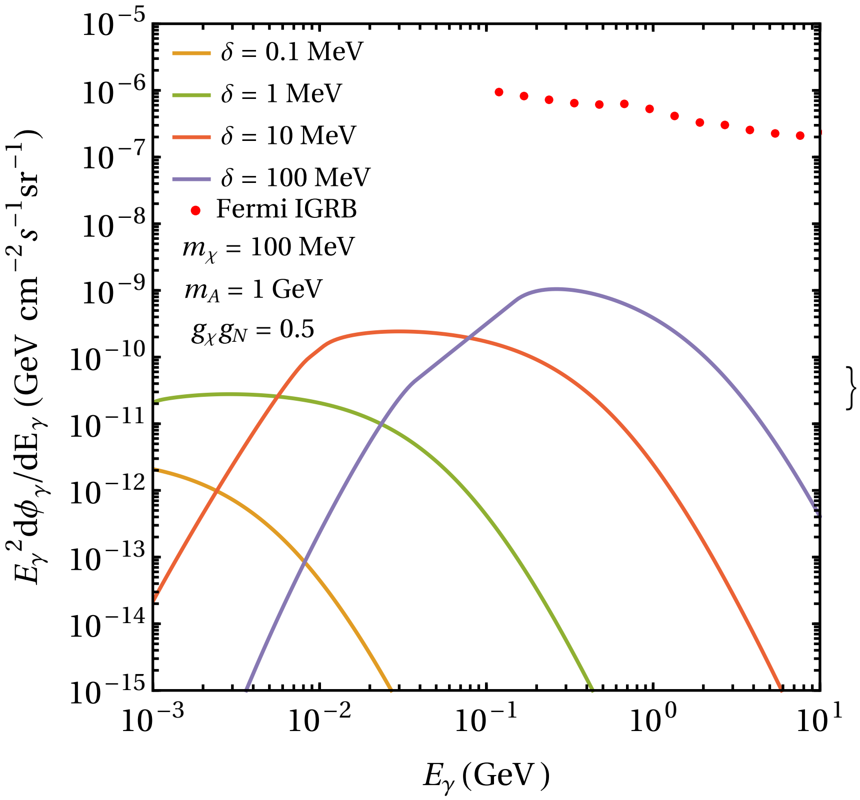
In[ ]:=
```
{photonFluxPlot1MeV = Show[ListLogLogPlot[datDecayγFluxM1MeV,
    Joined → True, FrameTicks → {LogTicks[10^-15, .1], LogTicks[10^-6, 10^3]},
    PlotRange → {{10^-3, 10}, {10^-10, 10^-5}}, PlotLegends → Legend[{"δ = 0.1 MeV",
       "δ = 1 MeV", "δ = 10 MeV", "δ = 100 MeV"}, Position → {.18, .86}],
    FrameLabel → {"Eγ (GeV)", "Eγ^2dϕγ/dEγ (GeV cm^-2s^-1sr^-1)"},
    PlotStyle → ({#, Thick} & /@ ColorData[97, "ColorList"][[{2, 3, 4, 5}]]),
    Epilog → Inset[Style["mχ = 1 MeV\nmA = 1 MeV\ngχgN = 10^-3",
       14, FontFamily → "Times"], Scaled[{.87, .89}]]]
   ,
   ListLogLogPlot[fermiIGRB, PlotStyle → Red,
    PlotLegends → Legend[{"Fermi IGRB"}, Position → {.18, .7}]]],
 photonFluxPlot100MeV = Show[ListLogLogPlot[datDecayγFluxM100MeV,
    Joined → True, FrameTicks → {LogTicks[10^-15, .1], LogTicks[10^-6, 10^3]},
    PlotRange → {{10^-3, 10}, {10^-15, 10^-5}}, PlotLegends → Legend[{"δ = 0.1 MeV",
       "δ = 1 MeV", "δ = 10 MeV", "δ = 100 MeV"}, Position → {.18, .86}],
    PlotStyle → ({#, Thick} & /@ ColorData[97, "ColorList"][[{2, 3, 4, 5}]]),
    FrameLabel → {"Eγ (GeV)", "Eγ^2dϕγ/dEγ (GeV cm^-2s^-1sr^-1)"},
    Epilog → Inset[Style["     mχ = 100 MeV\nmA = 1 GeV\ngχgN = 0.5",
       14, FontFamily → "Times"], Scaled[{.15, .6}]]]
   ,
   ListLogLogPlot[fermiIGRB, PlotStyle → Red,
    PlotLegends → Legend[{"Fermi IGRB"}, Position → {.18, .72}]]]}
```

Out[ ]= {



δ = 0.1 MeV
δ = 1 MeV
δ = 10 MeV
δ = 100 MeV
Fermi IGRB
m_χ = 1 MeV
m_A = 1 MeV
$g_\chi g_N = 10^{-3}$
$E_\gamma{}^2 d\phi_\gamma/dE_\gamma$ (GeV cm^{-2}s^{-1}sr^{-1})
E_γ (GeV)
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
10^{-10}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}

,



In[ ]:=
```
Export["./figures/fig_photon_flux_m1MeV.pdf", photonFluxPlot1MeV];
Export["./figures/fig_photon_flux_m100MeV.pdf", photonFluxPlot100MeV];
```

CRiDM scattering on Earth

This section describes the scattering of the upscattered dark matter component in a detector on Earth.

Two scenarios are considered:

I. χ_1 (produced from decay of χ_2) endothermically scatters in the detector

II. χ_2 lives long enough to exothermically scatter in the detector

Kinematics for up-scatter

Find minimum Tx to produce an upscatter with energy Er:

3-momentum conservation:

p(mx+δ) = p'(mx) + k(mT)

re-arrange and squaring:

$(p')^{\wedge}2 = (p - k)^2$

In[ ]:=
```
Solve[((Txp+mxp)^2-(mxp)^2)==((Tx+mx)^2-mx^2)+((Er+mT)^2-mT^2)+2 √(((Tx+mx)^2-mx^2)((Er+mT)^2-mT^2)) /.{mxp→mx+δ,Txp-
```

Out[ ]=
$$\left\{\left\{Tx\to\frac{Er\,(2\,Er\,mT-4\,mT\,mx+\delta\,(2\,mx+\delta))-\sqrt{Er\,(Er+2\,mT)\left(2\,Er\,mT+\delta^2\right)\left(2\,Er\,mT+(2\,mx+\delta)^2\right)}}{4\,Er\,mT}\right\},\left\{Tx\to\frac{Er\,(2\,Er\,mT-4\,mT\,mx+\delta\,(2\,mx+\delta))+\sqrt{Er\,(Er+2\,mT)\left(2\,Er\,mT+\delta^2\right)\left(2\,Er\,mT+(2\,mx+\delta)^2\right)}}{4\,Er\,mT}\right\}\right\}$$

Define functions for kinematic limits

Er = recoil energy

mT = target nuclei mass

In[ ]:=
```
TxMinEu[Er_?NumericQ ,mT_?NumericQ ,mx_?NumericQ ,δ_?NumericQ ]:=
1/(4 Er mT) (Er (2 Er mT-4 mT mx+δ (2 mx+δ))+ √(Er (Er+2 mT) (2 Er mT+δ^2) (2 Er mT+(2 mx+δ)^2)))
```

Global min:

In[ ]:=
```
TxMinU[mT_,mx_,δ_]= 1/(4 Er mT) (Er (2 Er mT-4 mT mx+δ (2 mx+δ))+ √(Er (Er+2 mT) (2 Er mT+δ^2) (2 Er mT+
```

Find the kinematic limits on the recoil energy:

In[ ]:=
```
Solve[((Txp + mxp)^2 - (mxp)^2) ==
    ((Tx + mx)^2 - mx^2) + ((Er + mT)^2 - mT^2) + 2 √(((Tx + mx)^2 - mx^2) ((Er + mT)^2 - mT^2)) /.
   {mxp → mx + δ, Txp -> Tx - Er - δ}, {Er}] // FullSimplify
```

Out[ ]=
$$\left\{\left\{Er\to-\frac{1}{2\,(mT+mx)^2+4\,mT\,Tx}\left(-2\,mT\,Tx\,(2\,mx+Tx)+2\,mx\,(mT+mx+Tx)\,\delta+(mT+mx+Tx)\,\delta^2+\sqrt{Tx\,(2\,mx+Tx)\left(2\,mT\,Tx-2\,(mT+mx)\,\delta-\delta^2\right)(-\delta\,(2\,mx+\delta)+2\,mT\,(2\,mx+Tx+\delta))}\right)\right\},\left\{Er\to\frac{1}{2\,(mT+mx)^2+4\,mT\,Tx}\left(2\,mT\,Tx\,(2\,mx+Tx)-2\,mx\,(mT+mx+Tx)\,\delta-(mT+mx+Tx)\,\delta^2+\sqrt{Tx\,(2\,mx+Tx)\left(2\,mT\,Tx-2\,(mT+mx)\,\delta-\delta^2\right)(-\delta\,(2\,mx+\delta)+2\,mT\,(2\,mx+Tx+\delta))}\right)\right\}\right\}$$

Define functions for the kinematic limits:

In[ ]:=
```
ErMinU[Tx_?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,mT_?NumericQ ]:=
If[-mT-mx+ √(mT^2+2 mT mx+mx^2+2 mT Tx) <δ,0,
- 1/(2 (mT+mx)^2+4 mT Tx) (-2 mT Tx (2 mx+Tx)+2 mx (mT+mx+Tx) δ+(mT+mx+Tx) δ^2+√(Tx (2 mx+Tx) (2 mT Tx-2
```

In[ ]:=
```
ErMaxU[Tx_?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,mT_?NumericQ ]:=
If[-mT-mx+√(mT^2+2 mT mx+mx^2+2 mT Tx) <δ,0,
1/(2 (mT+mx)^2+4 mT Tx) (2 mT Tx (2 mx+Tx)-2 mx (mT+mx+Tx) δ-(mT+mx+Tx) δ^2+√(Tx (2 mx+Tx) (2 mT Tx-2 (m
```

Kinematics for down-scatter

Excited DM comes in with kinetic energy = Tx and mass = mx’ = mx + δ, de-excited goes out

3-momentum conservation:

p(mx+δ) = p’(mx) + k(mT)

re-arrange and squaring:

$(p')^2 = (p - k)^2$

Find minimum Tx to produce a downscatter with recoil energy Er:

In[ ]:=
```
Solve[((Txp+mxp)^2-mxp^2)==((Tx+mxi)^2-mxi^2)+((Er+mT)^2-mT^2)+2 √(((Tx+mxi)^2-mxi^2)((Er+mT)^2-mT^2)) /.{mxp→mx,mxi
```

Out[ ]=

$$\left\{\left\{Tx \to \frac{Er\,(2\,Er\,mT - 4\,mT\,(mx+\delta) - \delta\,(2\,mx+\delta)) - \sqrt{Er\,(Er+2\,mT)\,(2\,Er\,mT+\delta^2)\,(2\,Er\,mT+(2\,mx+\delta)^2)}}{4\,Er\,mT}\right\},\right.$$
$$\left.\left\{Tx \to \frac{Er\,(2\,Er\,mT - 4\,mT\,(mx+\delta) - \delta\,(2\,mx+\delta)) + \sqrt{Er\,(Er+2\,mT)\,(2\,Er\,mT+\delta^2)\,(2\,Er\,mT+(2\,mx+\delta)^2)}}{4\,Er\,mT}\right\}\right\}$$

In[ ]:=
```
TxMinEd[Er_?NumericQ ,mT_?NumericQ ,mx_?NumericQ ,δ_?NumericQ ]:=
(Er (2 Er mT-4 mT (mx+δ)-δ (2 mx+δ))+√(Er (Er+2 mT) (2 Er mT+δ^2) (2 Er mT+(2 mx+δ)^2)))/(4 Er mT)
```

Find the kinematic limits on the recoil energy:

In[ ]:=
```
Solve[((Txp+mxp)^2-(mxp)^2)==((Tx+mxi)^2-mxi^2)+((Er+mT)^2-mT^2)+2 Sqrt[((Tx+mxi)^2-mxi^2)((Er+mT)^2-mT^2)] /.{Txp->Tx-Er
```

Out[ ]=

$$\left\{\left\{Er \to \frac{1}{2\left(4\,mT^2+8\,mT\,mx+4\,mx^2+8\,mT\,Tx+8\,mT\,\delta+8\,mx\,\delta+4\,\delta^2\right)}\left(16\,mT\,mx\,Tx+8\,mT\,Tx^2+8\,mT\,mx\,\delta+8\,mx^2\,\delta+16\,mT\,Tx\,\delta+8\,mx\,Tx\,\delta+4\,mT\,\delta^2+12\,mx\,\delta^2+4\,Tx\,\delta^2+4\,\delta^3-\sqrt{\left(4\,\delta^2\left(-4\,mx^2-4\,mx\,\delta-\delta^2\right)\left(4\,mT^2+8\,mT\,mx+4\,mx^2+8\,mT\,Tx+8\,mT\,\delta+8\,mx\,\delta+4\,\delta^2\right)+\left(-16\,mT\,mx\,Tx-8\,mT\,Tx^2-8\,mT\,mx\,\delta-8\,mx^2\,\delta-16\,mT\,Tx\,\delta-8\,mx\,Tx\,\delta-4\,mT\,\delta^2-12\,mx\,\delta^2-4\,Tx\,\delta^2-4\,\delta^3\right)^2\right)}\right)\right\},\right.$$

$$\left.\left\{Er \to \frac{1}{2\left(4\,mT^2+8\,mT\,mx+4\,mx^2+8\,mT\,Tx+8\,mT\,\delta+8\,mx\,\delta+4\,\delta^2\right)}\left(16\,mT\,mx\,Tx+8\,mT\,Tx^2+8\,mT\,mx\,\delta+8\,mx^2\,\delta+16\,mT\,Tx\,\delta+8\,mx\,Tx\,\delta+4\,mT\,\delta^2+12\,mx\,\delta^2+4\,Tx\,\delta^2+4\,\delta^3+\sqrt{\left(4\,\delta^2\left(-4\,mx^2-4\,mx\,\delta-\delta^2\right)\left(4\,mT^2+8\,mT\,mx+4\,mx^2+8\,mT\,Tx+8\,mT\,\delta+8\,mx\,\delta+4\,\delta^2\right)+\left(-16\,mT\,mx\,Tx-8\,mT\,Tx^2-8\,mT\,mx\,\delta-8\,mx^2\,\delta-16\,mT\,Tx\,\delta-8\,mx\,Tx\,\delta-4\,mT\,\delta^2-12\,mx\,\delta^2-4\,Tx\,\delta^2-4\,\delta^3\right)^2\right)}\right)\right\}\right\}$$

Define functions for the kinematic limits:

In[ ]:=
```
ErMinD[Tx_?NumericQ,mx_?NumericQ,δ_?NumericQ,mT_?NumericQ]:=
1/(2 (4 mT^2+8 mT mx+4 mx^2+8 mT Tx+8 mT δ+8 mx δ+4 δ^2)) (16 mT mx Tx+8 mT Tx^2+8 mT mx δ+8 mx^2 δ+16
```

In[ ]:=
```
ErMaxD[Tx_?NumericQ,mx_?NumericQ,δ_?NumericQ,mT_?NumericQ]:=
1/(2 (4 mT^2+8 mT mx+4 mx^2+8 mT Tx+8 mT δ+8 mx δ+4 δ^2)) (16 mT mx Tx+8 mT Tx^2+8 mT mx δ+8 mx^2 δ+16
```

These match with equation 11 of 2006.14089, but since we are scattering from a nucleus and not an electron, we don’t have a sharp peak in recoil energy.

Define cross sections

Upscatter

Differential cross section for the endothermic scatter of χ_1 on a nuclear target with A nucleons

Er = recoil energy

Tx1 = incoming χ_1 kinetic energy

returned cross section has units GeV^{-3}

In[ ]:=
```
dσdErUp[Er_?NumericQ,Tx1_?NumericQ,A_?NumericQ,mx_?NumericQ,δ_?NumericQ,gxi_?NumericQ,mA_?N
(gxi^4/(2π GeV^4) A^2 Fhelm[Sqrt[2 A amu Er],A]^2)(2(A amu)Er^2-Er(2(A amu+mx)^2+4 A amu Tx1+4mx δ +δ^2)+(A amu)(4
```

Downscatter

Differential cross section for the exothermic scatter of χ_2 on a nuclear target with A nucleons

Er = recoil energy

Tx2 = incoming χ_2 kinetic energy

returned cross section has units GeV^{-3}

In[ ]:=
```
dσdErDown[Er_?NumericQ ,Tx2_?NumericQ ,A_?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,gxi_?NumericQ ,mA_
(gxi^4/(2π GeV^4) A^2 Fhelm[Sqrt[2 A amu Er] ,A]^2)(2 (A amu)Er^2-Er(2(A amu+mx)^2+4 A amu Tx2+4(A amu)δ-δ^2)+(A amu
```

Define differential rates for xenon

Here we consider scattering on xenon, which we treat as a single isotope: xenon-131

In[ ]:=
```
(*Energies above this are assumed to be attenuated *)
TXMAX =10^2;
```

Define functions that return a list of { Er (keV) , rate (/tonne/yr/keV)}, sampled from ErMinGeV to ErMaxGeV with nPoints (log spaced)

In[ ]:=
```
dRdErUpList [ERMIN_ ?NumericQ ,ERMAX_ ?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,gxi_?NumericQ ,MA_?Nume
Module[{fluxX1 ,ErPoints ,TXMIN ,TXMAXU },
TXMIN =Max[TxMin[TdatMax [[1]],mi[[1]],mx ,δ],TxMinU [131 amu ,mx ,δ]];
TXMAXU =Min[TXMAX ,TxMax [TdatMax [[1]],mi[[1]],mx ,δ]];

fluxX1 =dϕX1dTx1list [Min[TXMIN ,TXMAXU ],TXMAXU ,mx ,δ,gxi ,MA ,4nPoints ]
//Interpolation [#,InterpolationOrder →1]&;

ErPoints =logSpace [Max[ErMinU [fluxX1 [[1,1,-1]],mx ,δ,131 amu]+10^-7 ,ERMIN ],Min[ErMaxU [fluxX1 [[1,1,-1]],

{10^6 ErPoints ,
ParallelTable [
  tonneYr/(131 amu (10^6(*keV*)))
  NIntegrate [
   fluxX1 [Tx1GeV ]×dσdErUp [ErGeV ,Tx1GeV ,131 ,mx ,δ,gxi ,MA]GeV^3 ,
   {Tx1GeV ,Max[Min[{TxMinEu [ErGeV ,131 amu ,mx ,δ],fluxX1 [[1,1,-1]]}],fluxX1 [[1,1,1]],fluxX1 [[1,1,-1]]},
   AccuracyGoal →Infinity ,PrecisionGoal →3,Method →{Automatic ,"SymbolicProcessing "→0},MinRecu
 ,{ErGeV ,ErPoints }]}//Thread //Return
];
```

In[•]:=

```
dRdErDownList [ERMIN_ ?NumericQ ,ERMAX_ ?NumericQ ,mx_?NumericQ ,δ_?NumericQ ,gx_?NumericQ ,MA_?Num
Module [{fluxX2 ,ErPoints ,TXMIN ,TXMAXD },
TXMIN =TxMin [TdatMax ⟦1⟧,mi⟦1⟧,mx ,δ]+10^-6;
TXMAXD =Min[TXMAX ,TxMax [TdatMax ⟦1⟧,mi⟦1⟧,mx ,δ]];

fluxX2 =dϕX2dTx2list [TXMIN ,TXMAXD ,mx ,δ,gx ,MA,4nPoints ]
//Interpolation [#,InterpolationOrder →1]&;

ErPoints =logSpace [Max[ErMinD [fluxX2 ⟦1,1,-1⟧,mx ,δ,131 amu]+10^-7,ERMIN ],Min[ErMaxD [fluxX2 ⟦1,1,-1⟧,
If[TXMIN >TXMAXD ,
Return [{ErPoints ,ConstantArray [0,Length [ErPoints ]]}//Thread ];
];

{10^6 ErPoints ,
  ParallelTable [
         tonneYr
    ─────────────────────
    131 amu (10^6(*keV*))
   NIntegrate [
     fluxX2 [Tx2GeV ] dσdErDown [ErGeV ,Tx2GeV ,131 ,mx ,δ,gx ,MA]GeV^3,
 {Tx2GeV ,Min[Max[TxMinEd [ErGeV ,131 amu ,mx ,δ],fluxX2 ⟦1,1,1⟧],fluxX2 ⟦1,1,-1⟧],fluxX2 ⟦1,1,-1⟧},
 AccuracyGoal →Infinity ,PrecisionGoal →3,Method →{Automatic ,"SymbolicProcessing "→0},MinRecurs
 ,{ErGeV ,ErPoints }]}//Thread //Return
];
```

Calculate rates

Elastic

In[•]:=

```
datDiffRateVecElas =
  {dRdErDownList [10^-6, 10^-3, .001, 0, gxLight, .001, 100],
   dRdErDownList [10^-6, 10^-3, .100, 0, gxHeavy, 1.00, 100]};
```

upscattering

In[ ]:=
```
datDiffRateVecUpM1MeV =
  With[{MX = .001, MA = .001, nPoints = 100},

   {dRdErUpList[10^-6, 10^-3, MX, 10^-3, gxLight, MA, nPoints],
    dRdErUpList[10^-6, 10^-3, MX, 10^-2, gxLight, MA, nPoints],
    dRdErUpList[10^-6, 10^-3, MX, 10^-1, gxLight, MA, nPoints]}];
```

In[ ]:=
```
datDiffRateVecUpM100MeV =
  With[{MX = .1, MA = 1, nPoints = 100},

   {dRdErUpList[10^-6, 10^-3, MX, 10^-3, gxHeavy, MA, nPoints],
    dRdErUpList[10^-6, 10^-3, MX, 10^-2, gxHeavy, MA, nPoints],
    dRdErUpList[10^-6, 10^-3, MX, 10^-1, gxHeavy, MA, nPoints]}];
```

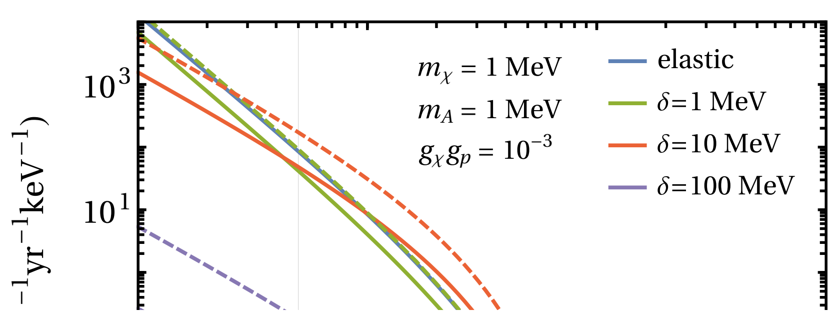
downscattering

In[ ]:=
```
datDiffRateVecDownM1MeV =
  With[{MX = .001, MA = .001, nPoints = 100},

   {dRdErDownList[10^-6, 10^-3, MX, 10^-3, gxLight, MA, nPoints],
    dRdErDownList[10^-6, 10^-3, MX, 10^-2, gxLight, MA, nPoints],
    dRdErDownList[10^-6, 10^-3, MX, 10^-1, gxLight, MA, nPoints]}];
```

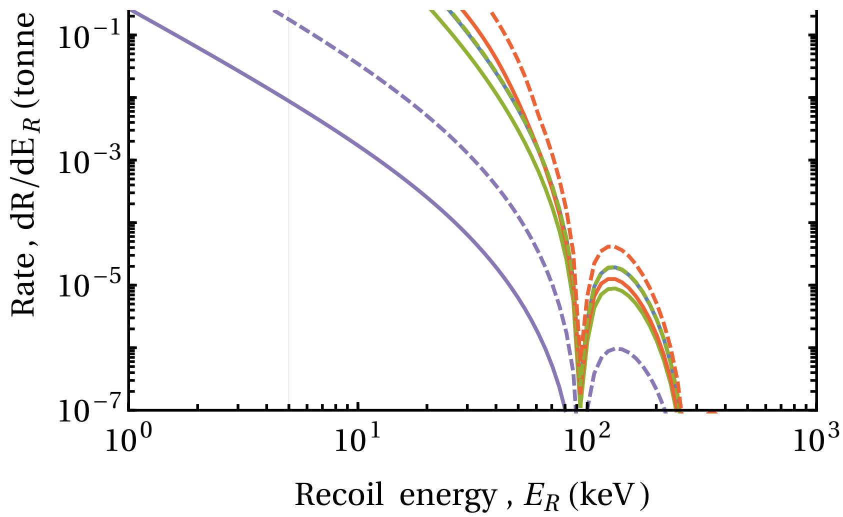
In[ ]:=
```
datDiffRateVecDownM100MeV =
  With[{MX = .1, MA = 1, nPoints = 100},

   {dRdErDownList[10^-6, 10^-3, MX, 10^-3, gxHeavy, MA, nPoints],
    dRdErDownList[10^-6, 10^-3, MX, 10^-2, gxHeavy, MA, nPoints],
    dRdErDownList[10^-6, 10^-3, MX, 10^-1, gxHeavy, MA, nPoints]}];
```

Rate plots

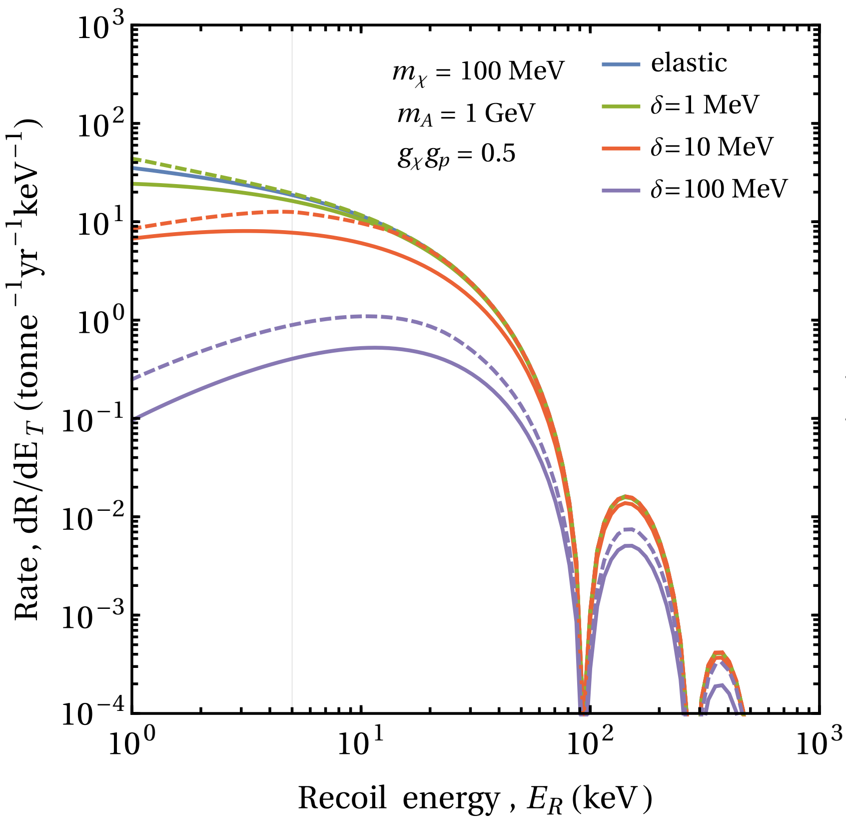
```
In[ ]:= {ratePlot1MeV = Show[
    ListLogLogPlot[
     Join[{datDiffRateVecElas[[1]]}, datDiffRateVecUpM1MeV], Joined → True,
     PlotRange → {{10^0, 10^3}, {10^-7, 10^4}},
     FrameTicks → {LogTicks[10^-16, 10^12, 2], LogTicks[10^0, 10^4]},
     PlotLegends →
      Legend[{"elastic", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.82, .85}],
     FrameLabel → {"Recoil energy, E_R (keV)", "Rate, dR/dE_R (tonne^-1yr^-1keV^-1)"},
     PlotStyle → ({#, Thick} & /@ ColorData[97, "ColorList"][[{1, 3, 4, 5}]]),
     GridLines → {{5}, None},
     Epilog → Inset[Style[" m_χ = 1 MeV \nm_A = 1 MeV\ng_χg_p = 10^-3 ",
        14, FontFamily → "Times"], Scaled[{.51, .87}]]],
    ListLogLogPlot[datDiffRateVecDownM1MeV, Joined → True,
     PlotStyle → ({#, Thick, Dashed} & /@ ColorData[97, "ColorList"][[3 ;; 5]])]
   ],
  ratePlot100MeV = Show[
    ListLogLogPlot[
     Join[{datDiffRateVecElas[[2]]}, datDiffRateVecUpM100MeV], Joined → True,
     FrameTicks → {LogTicks[10^-8, 10^12], LogTicks[10^-6, 10^3]},
     PlotRange → {{10^0, 10^3}, {10^-4, 10^3}},
     GridLines → {{5}, None},
     PlotStyle → ({#, Thick} & /@ ColorData[97, "ColorList"][[{1, 3, 4, 5}]]),
     PlotLegends →
      Legend[{"elastic", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.82, .85}],
     FrameLabel → {"Recoil energy, E_R (keV)", "Rate, dR/dE_T (tonne^-1yr^-1keV^-1)"},
     Epilog → Inset[Style["   m_χ = 100 MeV\nm_A = 1 GeV \ng_χg_p = 0.5     ",
        14, FontFamily → "Times"], Scaled[{.49, .87}]]],
    ListLogLogPlot[datDiffRateVecDownM100MeV, Joined → True,
     PlotStyle → ({#, Thick, Dashed} & /@ ColorData[97, "ColorList"][[3 ;; 5]])]
   ]}
```



Out[◦]= {



,



}

In[◦]:=

```
Export["figures/fig_rate_1MeV.pdf", ratePlot1MeV ];
Export["figures/fig_rate_100MeV.pdf", ratePlot100MeV ];
```

Normalized rate plot

```
In[ ]:= scaledDatDiffRateVecElas = datDiffRateVecElas;
scaledDatDiffRateVecElas[[1, All, 2]] *= 1 / datDiffRateVecElas[[1, 24, 2]];
scaledDatDiffRateVecElas[[2, All, 2]] *= 1 / datDiffRateVecElas[[2, 24, 2]];
```

```
In[ ]:= scaledDatDiffRateVecUpM1MeV = datDiffRateVecUpM1MeV;
scaledDatDiffRateVecUpM1MeV[[All, All, 2]] *=
  scaledDatDiffRateVecElas[[1, 24, 2]] / datDiffRateVecUpM1MeV[[All, 24, 2]];
```

```
In[ ]:= scaledDatDiffRateVecDownM1MeV = datDiffRateVecDownM1MeV;
scaledDatDiffRateVecDownM1MeV[[All, All, 2]] *=
  scaledDatDiffRateVecElas[[1, 24, 2]] / datDiffRateVecDownM1MeV[[All, 24, 2]];
```

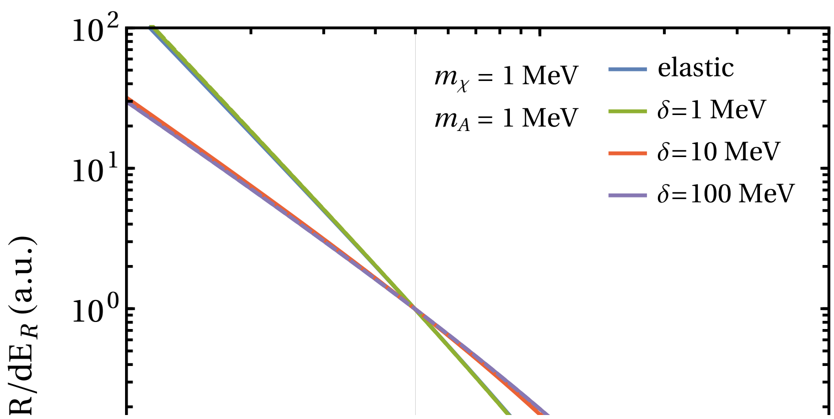
```
In[ ]:= scaledDatDiffRateVecUpM100MeV = datDiffRateVecUpM100MeV;
scaledDatDiffRateVecUpM100MeV[[All, All, 2]] *=
  scaledDatDiffRateVecElas[[2, 24, 2]] / datDiffRateVecUpM100MeV[[All, 24, 2]];
```

```
In[ ]:= scaledDatDiffRateVecDownM100MeV = datDiffRateVecDownM100MeV;
scaledDatDiffRateVecDownM100MeV[[All, All, 2]] *=
  scaledDatDiffRateVecElas[[2, 24, 2]] / datDiffRateVecDownM100MeV[[All, 24, 2]];
```

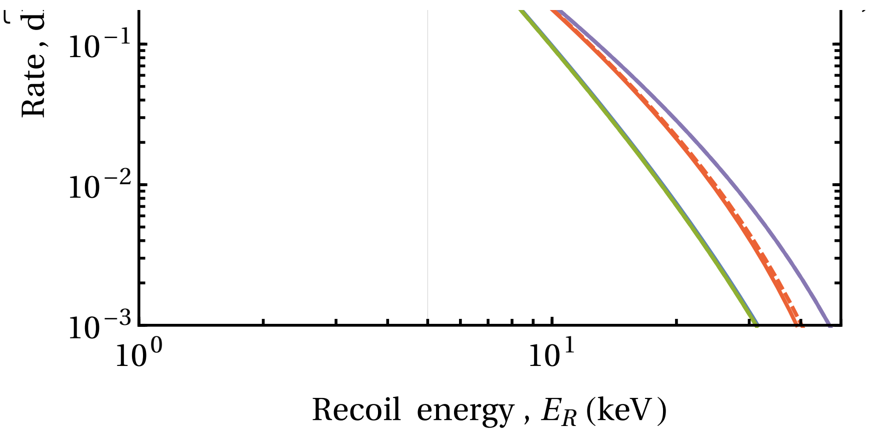
Log scale:

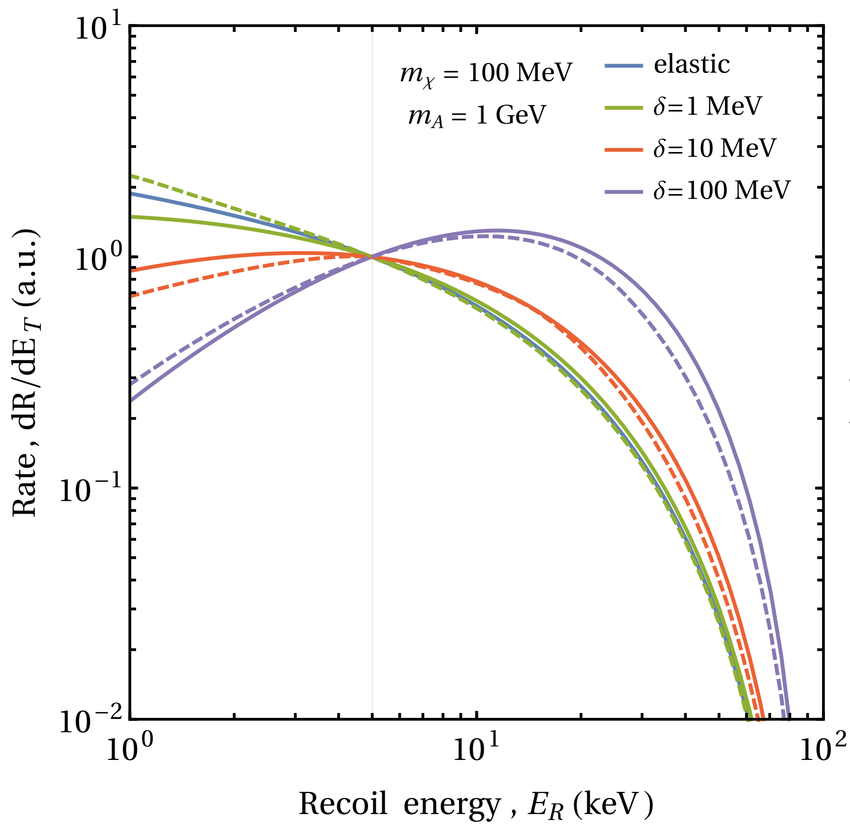
In[ ]:=
```
{Show[
  ListLogLogPlot[Join[{scaledDatDiffRateVecElas⟦1⟧},
    scaledDatDiffRateVecUpM1MeV ], Joined → True,
   PlotRange → {{10^0, 50}, {10^-3, 10^2}},
   FrameTicks → {LogTicks[10^-16, 10^12], LogTicks[10^0, 10^4]},
   PlotLegends →
    Legend[{"elastic", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.82, .85}],
   FrameLabel → {"Recoil energy, E_R (keV)", "Rate, dR/dE_R (a.u.)"},
   PlotStyle → ({#, Thick} &/@ ColorData[97, "ColorList"][[{1, 3, 4, 5}]]),
   GridLines → {{5}, None},
   Epilog → Inset[Style[" m_χ = 1 MeV \nm_A = 1 MeV",
      14, FontFamily → "Times"], Scaled[{.54, .9}]]],
  ListLogLogPlot[scaledDatDiffRateVecDownM1MeV , Joined → True,
   PlotStyle → ({#, Thick, Dashed} &/@ ColorData[97, "ColorList"][[3 ;; 5]])]
 ],
 Show[
  ListLogLogPlot[Join[{scaledDatDiffRateVecElas⟦2⟧},
    scaledDatDiffRateVecUpM100MeV ], Joined → True,
   FrameTicks → {LogTicks[10^-8, 10^12], LogTicks[10^-4, 10^2]},
   PlotRange → {{10^0, 10^2}, {10^-2, 10^1}},
   GridLines → {{5}, None},
   PlotStyle → ({#, Thick} &/@ ColorData[97, "ColorList"][[{1, 3, 4, 5}]]),
   PlotLegends →
    Legend[{"elastic", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.82, .85}],
   FrameLabel → {"Recoil energy, E_R (keV)", "Rate, dR/dE_T (a.u.)"},
   Epilog → Inset[Style["   m_χ = 100 MeV\nm_A = 1 GeV",
      14, FontFamily → "Times"], Scaled[{.50, .90}]]],
  ListLogLogPlot[scaledDatDiffRateVecDownM100MeV , Joined → True,
   PlotStyle → ({#, Thick, Dashed} &/@ ColorData[97, "ColorList"][[3 ;; 5]])]
 ]}
```

Out[ ]= {



,





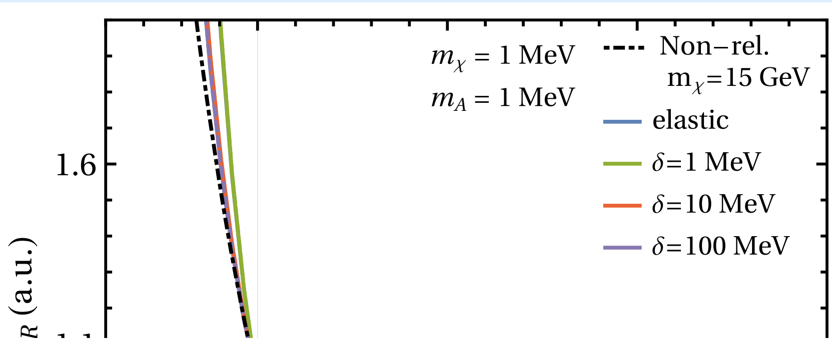
}

Linear scale:

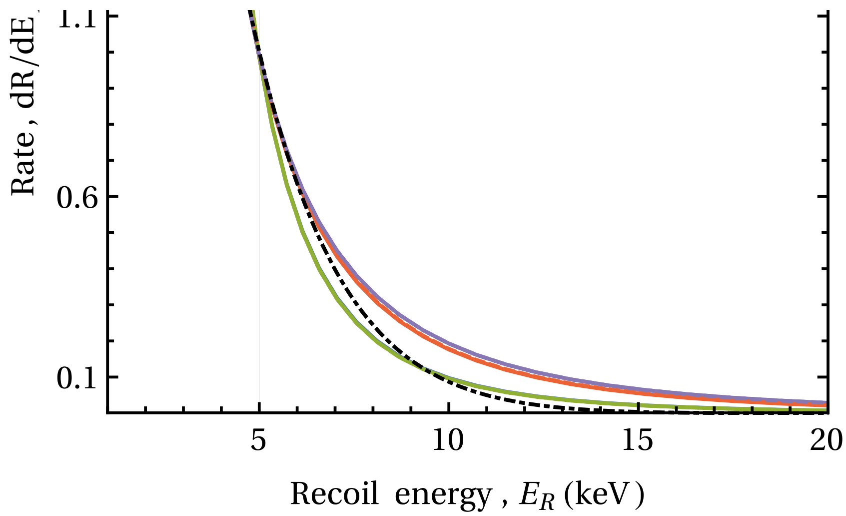
In[∘]:=

```
{scaledRatePlot1MeV = Show[
   ListPlot[Join[{scaledDatDiffRateVecElas ⟦1⟧},
     scaledDatDiffRateVecUpM1MeV ], Joined → True,
    PlotRange → {{10^0, 20}, {10^-3, 2}},
    PlotLegends →
     Legend[{"elastic", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.82, .77}],
    FrameLabel → {"Recoil energy, E_R (keV)", "Rate, dR/dE_R (a.u.)"},
```

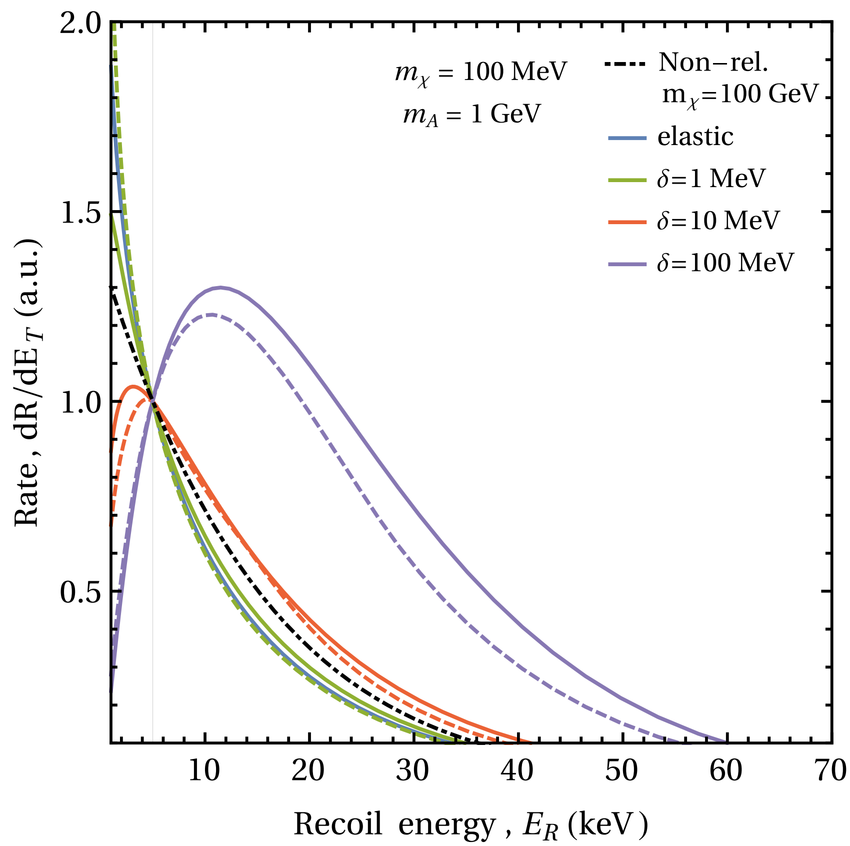
```
    PlotStyle → ({#, Thick} & /@ ColorData[97, "ColorList"][[{1, 3, 4, 5}]]),
    GridLines → {{5}, None},
    Epilog → Inset[Style[" mχ = 1 MeV \nmA = 1 MeV",
       14, FontFamily → "Times"], Scaled[{.55, .92}]]],
   ListPlot[scaledDatDiffRateVecDownM1MeV , Joined → True,
    PlotStyle → ({#, Thick, Dashed} & /@ ColorData[97, "ColorList"][[3 ;; 5]])],
   Plot[dRdEr[Er 10^-6, 0, 131, 15, 1, 1] / dRdEr[5 × 10^-6, 0, 131, 15, 1, 1], {Er, 1, 60},
    PlotStyle → {Black, DotDashed, Thick}, PlotRange → All,
    PlotLegends → Legend[{"Non-rel.       \n  mχ=15 GeV"},
      Position → {.835, .94}, LegendMarkerSize → 21]]
  ],
 scaledRatePlot100MeV = Show[
   ListPlot[Join[{scaledDatDiffRateVecElas [[2]]},
     scaledDatDiffRateVecUpM100MeV ], Joined → True,
    PlotRange → {{10^0, 70}, {10^-1, 2}},
    GridLines → {{5}, None},
    PlotStyle → ({#, Thick} & /@ ColorData[97, "ColorList"][[{1, 3, 4, 5}]]),
    PlotLegends →
     Legend[{"elastic", "δ=1 MeV", "δ=10 MeV", "δ=100 MeV"}, Position → {.82, .75}],
    FrameLabel → {"Recoil energy, ER (keV)", "Rate, dR/dET (a.u.)"},
    Epilog → Inset[Style["   mχ = 100 MeV\nmA = 1 GeV",
       14, FontFamily → "Times"], Scaled[{.50, .90}]]],
   ListPlot[scaledDatDiffRateVecDownM100MeV , Joined → True,
    PlotStyle → ({#, Thick, Dashed} & /@ ColorData[97, "ColorList"][[3 ;; 5]])],
   Plot[dRdEr[Er 10^-6, 0, 131, 100, 1, 1] / dRdEr[5 × 10^-6, 0, 131, 100, 1, 1],
    {Er, 1, 60}, PlotStyle → {Black, DotDashed, Thick},
    PlotLegends → Legend[{"Non-rel.         \n mχ=100 GeV"},
      Position → {.835, .92}, LegendMarkerSize → 21]]
  ]}
```



Out[•]= {



,



}

```
Export["figures/fig_rateScaled _1MeV.pdf", scaledRatePlot1MeV ];
Export["figures/fig_rateScaled _100MeV.pdf", scaledRatePlot100MeV ];
```

XENON1t bounds

Total rates in XENON1T

Define efficiency function, returns fraction of events observed for a given recoil energy

ErkeV = recoil energy in keV

In[ ]:=
```
xeEff[ErkeV_]:=.5(0.44*Erfc[.3*(5.4-ErkeV)]*0.5*Erfc[.14*(ErkeV-42)])
```

Total event rate fucntions in XENON1T for:

- non-relativistic DM:
- endothermic CRDM scattering:
- exothermic CRDM scattering:

mx = dark matter mass in GeV

δ = mass splitting in GeV

gxi = coupling

mA = mass of mediator in GeV

Rate is returned with units /tonne/yr

In[ ]:=
```
NReventsXe1t[mx_?NumericQ,δ_?NumericQ,gxi_?NumericQ,mA_?NumericQ]:=
```

$$\text{If}\left[\left(\frac{\mu[131\,\text{amu},\text{mx}]\left(\frac{\text{vesc+ve}}{\text{c}}\right)^2}{2}<\delta\right)\|\left(\text{VminGlobal}[\delta,\text{mx},131\ \text{amu}]>\frac{\text{ve+vesc}}{\text{c}}\right),0,\right.$$

```
NIntegrate[
dRdErFull[Er,δ,131,mx,gxi,mA]10^6,
{Er,Max[2×10^-6,ErMinGeV[δ,VMAX,mx,131amu]],ErMaxGeV[δ,VMAX,mx,131amu]},
PrecisionGoal→3]]
```

In[ ]:=
```
iCRDMeventsXe1tU[mx_?NumericQ,δ_?NumericQ,gxi_?NumericQ,mA_?NumericQ,nPoints_:50]:=
Module[{rate},

 rate=Interpolation[
 dRdErUpList[Max[ErMinU[TXMAX,mx,δ,131amu],2×10^-6],Min[ErMaxU[TXMAX,mx,δ,131amu],60×10^-6],mx,δ,g
,InterpolationOrder→1];

NIntegrate[
xeEff[ErkeV]×rate[ErkeV],
 {ErkeV,10^6Max[ErMinU[TXMAX,mx,δ,131amu],2×10^-6],10^6Min[ErMaxU[TXMAX,mx,δ,131amu],60×10^-6]},
 PrecisionGoal→2]//Return;
];
```

```
In[ ]:= iCRDMeventsXe1tD [mx_?NumericQ ,δ_?NumericQ ,gxi_?NumericQ ,mA_?NumericQ ,nPoints_ :50]:=
Module [{rate},

 rate=Interpolation [
 dRdErDownList [Max[ErMinD [TXMAX ,mx ,δ ,131 amu],2×10^-6],Min[ErMaxD [TXMAX ,mx ,δ ,131 amu],60×10^-6],mx ,δ
InterpolationOrder  →1];

NIntegrate [
xeEff [ErkeV ]×rate [ErkeV ],
 {ErkeV ,10^6 Max[ErMinD [TXMAX ,mx ,δ ,131 amu],2×10^-6],10^6 Min[ErMaxD [TXMAX ,mx ,δ ,131 amu],60×10^-6]},
 PrecisionGoal  →2]//Return ;
];
```

Sensitivity in mass vs. coupling

Assuming a 1t.y exposure, based on the second column of table I of 1805.12562:

```
In[ ]:= Nexp = 7.36;
        Nobs = 14;
```

```
In[ ]:= logL[x_, bgexp_, obs_, bgN_] = -(bgexp bgN + x) +
           obs Log[bgexp bgN + x] - Log[obs!] + Log[PDF[NormalDistribution [1, .15], bgN]];
```

Find the 90%CL:

```
In[ ]:= sol =
         FindMinimum [{(√(-2 (logL[x, Nexp, Nobs, 1] - logL[Nobs - Nexp, Nexp, Nobs, 1])) - InverseCDF [
                NormalDistribution [0, 1], 0.9])^2, x > 1}, {x, 10}]

Out[ ]= {3.66198 × 10^-16, {x → 11.9975}}
```

```
In[ ]:= NupperLimit = x /. sol[[2]];
```

We use 12 events as the upper limit.

Calculations

CRDM upscattering bounds:

```
With[{MA = 1, npoints = 100},
xe1tDatiCRDMmxGGheavMedU = {
    Table[
     {mx, Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tU [mx, .0001, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.001, 10^2, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tU [mx, .001, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tU [mx, .010, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 40, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tU [mx, .100, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 2, 100]}]};
]
```

In[ ]:=
```
del = {"p0001", "p001", "p01", "p1"};
Table[Export["Xe1t_bounds _heavyMed _del" <> del[[ii]] <> "_up.csv",
   xe1tDatiCRDMmxGGheavMedU [[ii]]], {ii, 1, 4}];
```

```
With[{MA = .001, npoints = 100},
xe1tDatiCRDMmxGGlightMedU  = {
Table[
    {mx, √(√(NupperLimit  /(iCRDMeventsXe1tU [mx, .0001, 1, MA, npoints] + 10^-100))) },
    {mx, logSpace[.001, 100, 100]}],
   Table[
    {mx,
      √(√(NupperLimit  /(iCRDMeventsXe1tU [mx, .001, 1, MA, npoints] + 10^-100))) },
    {mx, logSpace[.0001, 100, 100]}],
   Table[
    {mx,
      √(√(NupperLimit  /(iCRDMeventsXe1tU [mx, .010, 1, MA, npoints] + 10^-100))) },
    {mx, logSpace[.0001, 100, 100]}],
   Table[
    {mx,
      √(√(NupperLimit  /(iCRDMeventsXe1tU [mx, .100, 1, MA, npoints] + 10^-100))) },
    {mx, logSpace[.0001, 100, 100]}]};
]
```

In[◦]:=
```
del = {"p0001", "p001", "p01", "p1"};
Table[Export["Xe1t_bounds _lightMed _del" <> del[[ii]] <> "_up.csv",
   xe1tDatiCRDMmxGGlightMedU [[ii]]], {ii, 1, 4}];
```

CRDM downscattering bounds:

```
With[{MA = 1, npoints = 60},
xe1tDatiCRDMmxGGheavMedD = {
    Table[{mx, Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx,  0, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .0001, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .001, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .010, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001,  40, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .100, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001,  2, 100]}]};
]
```

In[ ]:=
```
del = {"0", "p0001", "p001", "p01", "p1"};
Table[Export["Xe1t_bounds_heavyMed_del" <> del[[ii]] <> "_down.csv",
   xe1tDatiCRDMmxGGheavMedD[[ii]]], {ii, 1, 5}];
```

```
With[{MA = .001, npoints = 60},
xe1tDatiCRDMmxGGlightMedD = {
    Table[{mx, Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, 0, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .0001, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .001, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .010, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 100, 100]}],
    Table[
     {mx,
       Sqrt[Sqrt[NupperLimit / (iCRDMeventsXe1tD[mx, .100, 1, MA, npoints] + 10^-100)]]},
     {mx, logSpace[.0001, 1, 100]}]};
]
```

In[ ]:=
```
del = {"0", "p0001", "p001", "p01", "p1"};
Table[Export["Xe1t_bounds_lightMed_del" <> del[[ii]] <> "_down.csv",
   xe1tDatiCRDMmxGGlightMedD[[ii]]], {ii, 1, 5}];
```

Import previously calculated bounds

In[ ]:=
```
NotebookDirectory[] // SetDirectory;
```

In[ ]:=
```
del = {"p0001", "p001", "p01", "p1"};
xe1tDatiCRDMmxGGheavMedU =
  Table[Import["results/Xe1t_bounds_heavyMed_del" <> del[[ii]] <> "_up.csv"], {ii, 1, 4}];
xe1tDatiCRDMmxGGlightMedU =
  Table[Import["results/Xe1t_bounds_lightMed_del" <> del[[ii]] <> "_up.csv"], {ii, 1, 4}];
```

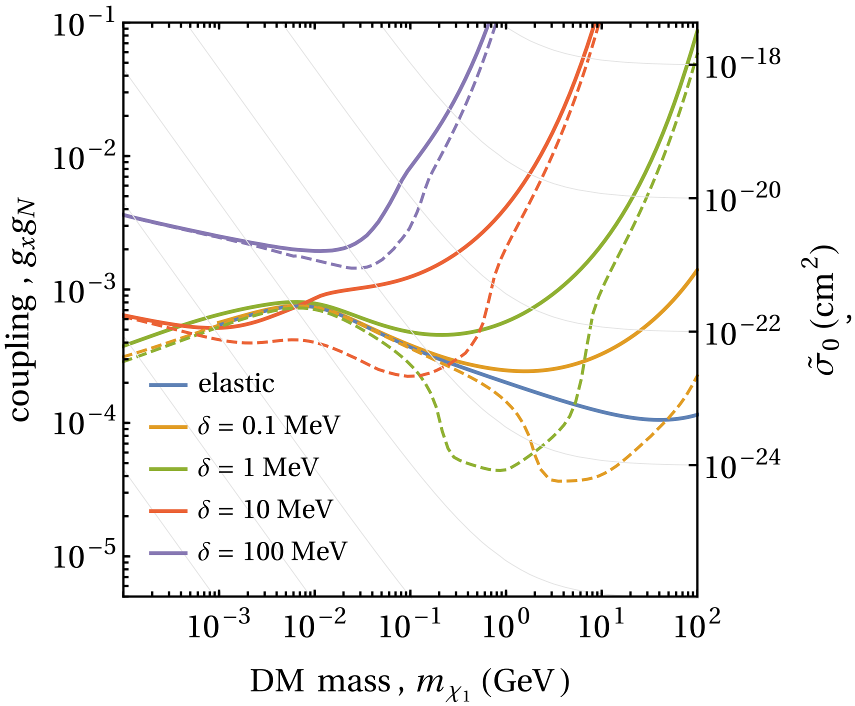
In[ ]:=
```
del = {"0", "p0001", "p001", "p01", "p1"};
xe1tDatiCRDMmxGGheavMedD = Table[
   Import["results/Xe1t_bounds_heavyMed_del" <> del[[ii]] <> "_down.csv"], {ii, 1, 5}];
xe1tDatiCRDMmxGGlightMedD = Table[
   Import["results/Xe1t_bounds_lightMed_del" <> del[[ii]] <> "_down.csv"], {ii, 1, 5}];
```

Plot sensitivity

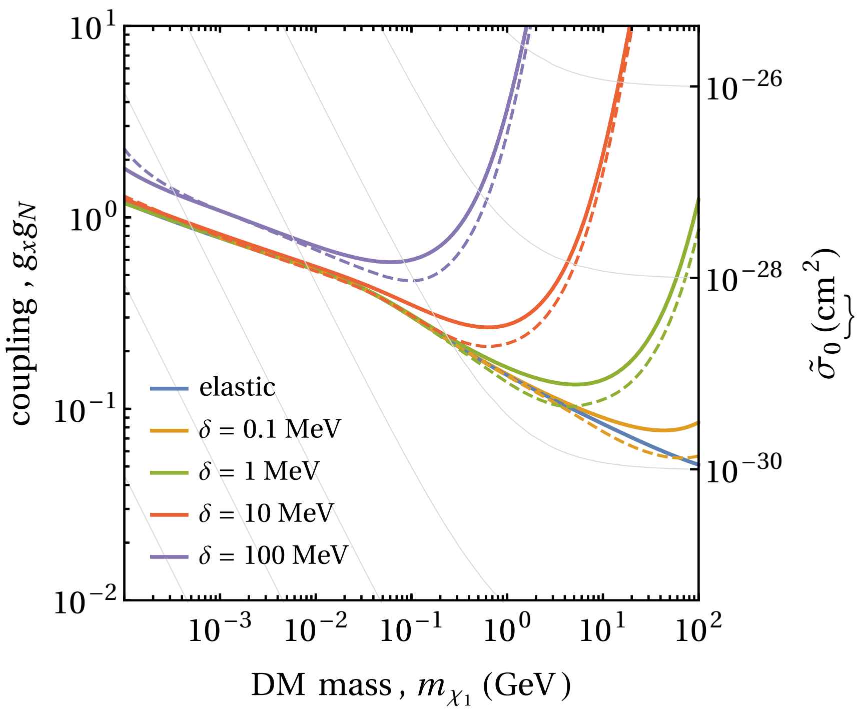
In[ ]:=
```
label[x_,y_,z_]:=Text[Superscript[10,z],{x,y},Background→White]
```

In[ ]:=
```
{boundPlotMed1MeV = Show[ListLogLogPlot[Join[
     {xe1tDatiCRDMmxGGlightMedD[[1]]}, xe1tDatiCRDMmxGGlightMedU], Joined → True,
    PlotRange → {{.0001, 10^2}, {5 × 10^-6, 10^-1}},
    PlotLegends → Legend[{"elastic", "δ = 0.1 MeV",
       "δ = 1 MeV", "δ = 10 MeV", "δ = 100 MeV"}, Position → {.22, .22}],
    FrameLabel → {{"coupling, gχgN", "σ̃0 (cm^2)"}, {"DM mass, mχ1 (GeV)", ""}},
    FrameTicks → {{LogTicks[10^-5, 10^3][[1]], Table[{gg[10^x, 100, .001]^2,
          Superscript[10, x]}, {x, -42, -14, 2}]}, LogTicks[10^-4, 10^2]}],
   ListLogLogPlot[xe1tDatiCRDMmxGGlightMedD, Joined → True, PlotStyle → Dashed],
   ContourPlot[σXP[Sqrt[gg], mx, .001] // Log10, {mx, 10^-4, 10^3}, {gg, 10^-6, 10},
    Contours → Range[-38, -14, 2], ScalingFunctions → {"Log", "Log"},
    ContourShading → None, ContourStyle → Lighter[Gray, .8]]
  ],
 boundPlotMed1GeV = Show[
   ListLogLogPlot[
    Join[{xe1tDatiCRDMmxGGheavMedD[[1]]}, xe1tDatiCRDMmxGGheavMedU], Joined → True,
    PlotRange → {{.0001, 10^2}, {10^-2, 10^1}},
    PlotLegends → Legend[{"elastic", "δ = 0.1 MeV",
       "δ = 1 MeV", "δ = 10 MeV", "δ = 100 MeV"}, Position → {.22, .22}],
    FrameLabel → {{"coupling, gχgN", "σ̃0 (cm^2)"}, {"DM mass, mχ1 (GeV)", ""}},
    FrameTicks → {{LogTicks[10^-4, 10^1][[1]], Table[{gg[10^x, 100, 1]^2, Superscript[10, x]},
         {x, -42, -20, 2}]}, LogTicks[10^-4, 10^3]}],
   ListLogLogPlot[xe1tDatiCRDMmxGGheavMedD, Joined → True, PlotStyle → Dashed],
   ContourPlot[σXP[Sqrt[g], mx, 1] // Log10, {mx, 10^-4, 10^3}, {g, 10^-4, 10},
    Contours → Range[-38, -26, 2], ScalingFunctions → {"Log", "Log"},
    ContourShading → None, ContourStyle → Lighter[Gray, .7]]
  ]}
```

Out[∘]= {



coupling , $g_\chi g_N$
DM mass , m_{χ_1} (GeV)
$\tilde{\sigma}_0$ (cm^2)
elastic
δ = 0.1 MeV
δ = 1 MeV
δ = 10 MeV
δ = 100 MeV



coupling , $g_\chi g_N$
DM mass , m_{χ_1} (GeV)
$\tilde{\sigma}_0$ (cm^2)
elastic
δ = 0.1 MeV
δ = 1 MeV
δ = 10 MeV
δ = 100 MeV

```
Export["./figures/fig_bound_m1GeV.pdf", boundPlotMed1GeV];
Export["./figures/fig_bound_m1MeV.pdf", boundPlotMed1MeV];
```

Attenuation

In this section we approximate the attenuation of the CRDM before reaching XENON1t at LNGS.

We estimate attenuation for the elastic case only and find the coupling which would attenuate 10GeV dark matter to below threshold.

Depth of LNGS:

```
Zlngs=140000 Centimeter GeV;
```

Approximate composition of the crust from Kouvaris and Emken:

```
NA=6.02×10^23/mol;
ρE=2.7g/Centimeter^3;
AN={16,28,27,56,40,39,23,24};
nN=ρE NA{.466/(16 g/mol), .277/(28 g/mol), .081/(27 g/mol), .05/(55.85 g/mol), .036/(40 g/mol), .028/(39.1 g/mol), .026/(23 g/mol), .021/(24.3 g/mol)};(
```

Average these as an effective density and atomic mass (see appendix for validity):

```
avgN=ρE NA Plus@@{.466/(16 g/mol), .277/(28 g/mol), .081/(27 g/mol), .05/(55.85 g/mol), .036/(40 g/mol), .028/(39.1 g/mol), .026/(23 g/mol), 24
avgA=Plus@@{.466×16,28×.277,27×.081,56×.05,40×.036,39×.028,23×.026,24×.021};
```

Define energy loss function:

```
dTdzUpApprox[Txx_?NumericQ,mx_?NumericQ,δ_?NumericQ,gx_?NumericQ,mA_?NumericQ]:=
NIntegrate[avgN Er dσdErUp[Er,Txx,avgA,mx,δ,gx,mA],{Er,ErMinU[Txx,mx,δ,avgA mp],ErMaxU[Txx,m
```

Solving the energy loss DE:

```
TxLNGS[TxSurface_?NumericQ,mx_?NumericQ,δ_?NumericQ,gx_?NumericQ,mA_?NumericQ]:=
Module[{Tx},
NDSolveValue[{D[Tx[z],z]==-dTdzUpApprox[Tx[z],mx,δ,gx,mA]GeV^-2,Tx[0]==TxSurface},Tx[Zlngs],{z,0,Zlr
```

```
In[ ]:= With[{mA = .001},
  gx = .435;
  attenDat10 = Table[{mx,
     TxZ = TxLNGS[10, mx, 0, gx, mA];
     While[TxZ > TxMinEd[5 × 10^-6, 131 amu, mx, 0],
      gx *= 1.0025;
      TxZ = TxLNGS[10, mx, 0, gx, mA];];
     gx^2},
    {mx, logSpace[.0001, 10, 30] // Reverse}
   ]];
```

Fermi bounds

```
In[ ]:= fermiIGRB =Import["data/FERMI_IGRB.csv"];
```

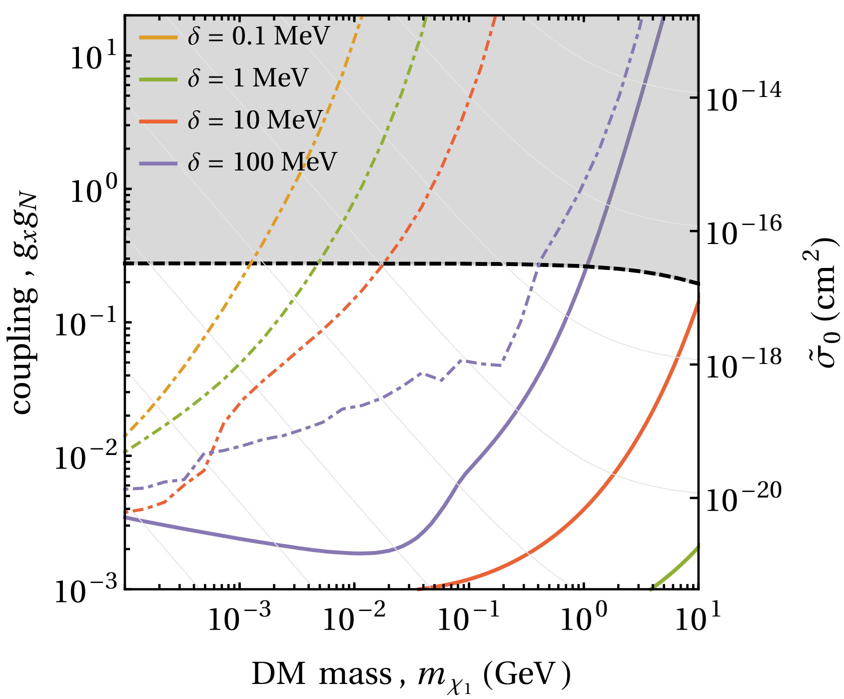
```
In[ ]:= With[{MA = 1, gx = 1, EMIN = .119, EMAX = 30.49, nPoints = 17},
  fermiLimitHeavy = Table[{MX, dϕγdElist[EMIN, EMAX, MX, δ, gx, MA, nPoints] //
        Sqrt[fermiIGRB[[1 ;; 17, 2]] / (#[[All, 2]] 1/(4 π) GeV (unitsCM2S cm^2 s) #[[All, 1]]^2 + 10^-100)]
        & // Min}
 , {δ, {.001, .01, .1}}
 , {MX, logSpace[10^-4, 10, 20]}]
];
```

```
In[ ]:= With[{MA = .001, gx = 1, EMIN = fermiIGRB[[1, 1]],
   EMAX = fermiIGRB[[-1, 1]], nPoints = Length[fermiIGRB]},
  fermiLimitLight =
Table[{MX, dϕγdElist[EMIN, EMAX, MX, δ, gx, MA, nPoints] //
        Sqrt[fermiIGRB[[All, 2]] / (#[[All, 2]] 1/(4 π) GeV (unitsCM2S cm^2 s) #[[All, 1]]^2 + 10^-100)] & //
       Min}
 , {δ, {.0001, .001, .01, .1}}
 , {MX, logSpace[10^-4, 10, 30]}]
];
```

In[ ]:=

```
fermiBoundPlot = Show[
  ListLogLogPlot[xe1tDatiCRDMmxGGlightMedU , Joined → True,
   PlotRange → {{.0001, 10}, {10^-3, 20}},
   PlotLegends → Legend[{"δ = 0.1 MeV",
      "δ = 1 MeV", "δ = 10 MeV", "δ = 100 MeV"}, Position → {.2, .85}],
   FrameLabel → {{"coupling, gχgN", "σ̃0 (cm^2)"}, {"DM mass, mχ1 (GeV)", ""}},
   PlotStyle → ({#, Thick} &/@ ColorData[97, "ColorList"][[{2, 3, 4, 5}]]),
   FrameTicks →
    {{LogTicks[10^-5, 10^3][[1]], Table[{gg[10^x, 100, .001]^2, Superscript[10, x]},
        {x, -42, -14, 2}]}, LogTicks[10^-4, 10^2]}],
  ListLogLogPlot[attenDat10 , Joined → True,
   PlotStyle → {Thick, Black, Dashed},
   Filling → Top, FillingStyle → Opacity[.3, Gray],
   PlotRange → {.1, 20}],
  ContourPlot[σXP[Sqrt[gg] , mx, .001] // Log10, {mx, 10^-4, 10^3}, {gg, 10^-6, 100},
   Contours → Range[-38, -14, 2], ScalingFunctions → {"Log", "Log"},
   ContourShading → None, ContourStyle → Lighter[Gray, .8]], ListLogLogPlot[
   Join[{{{0, 0}, {0, 0}}}, fermiLimitLight], PlotStyle → DotDashed, Joined → True]]
```

Out[ ]=



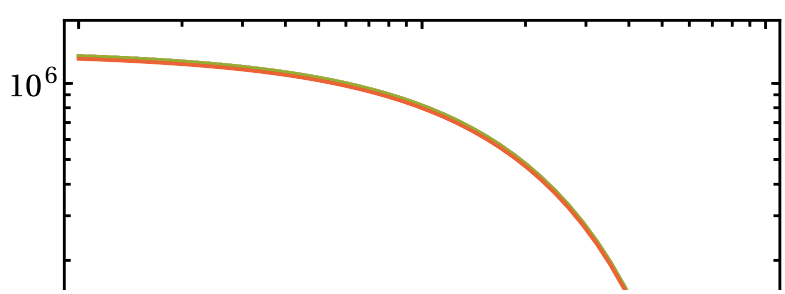
In[ ]:=
```
Export["figures/fig_fermi_atten.pdf", fermiBoundPlot]
```
Out[ ]=
```
figures/fig_fermi_atten.pdf
```

Appendix

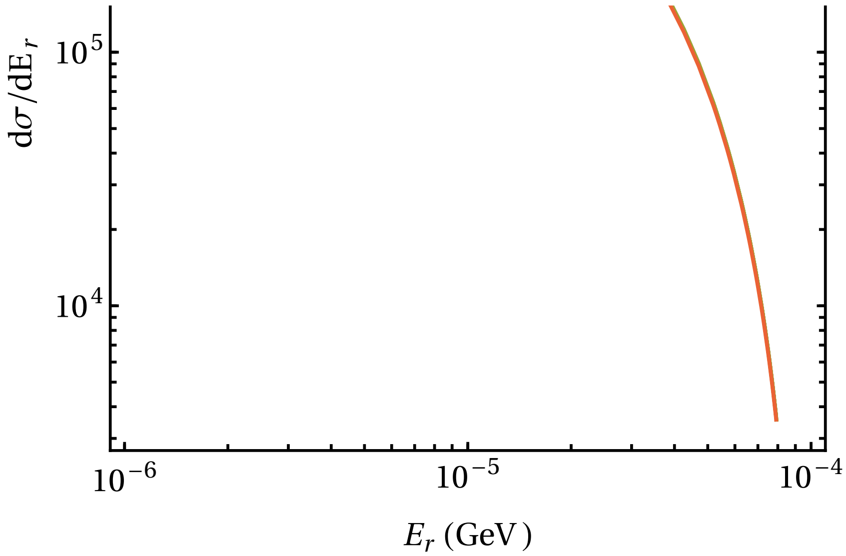
Cross sections

Since δ is small the cross sections are degenerate, therefore the differences in the rate come from the flux and kinematics.

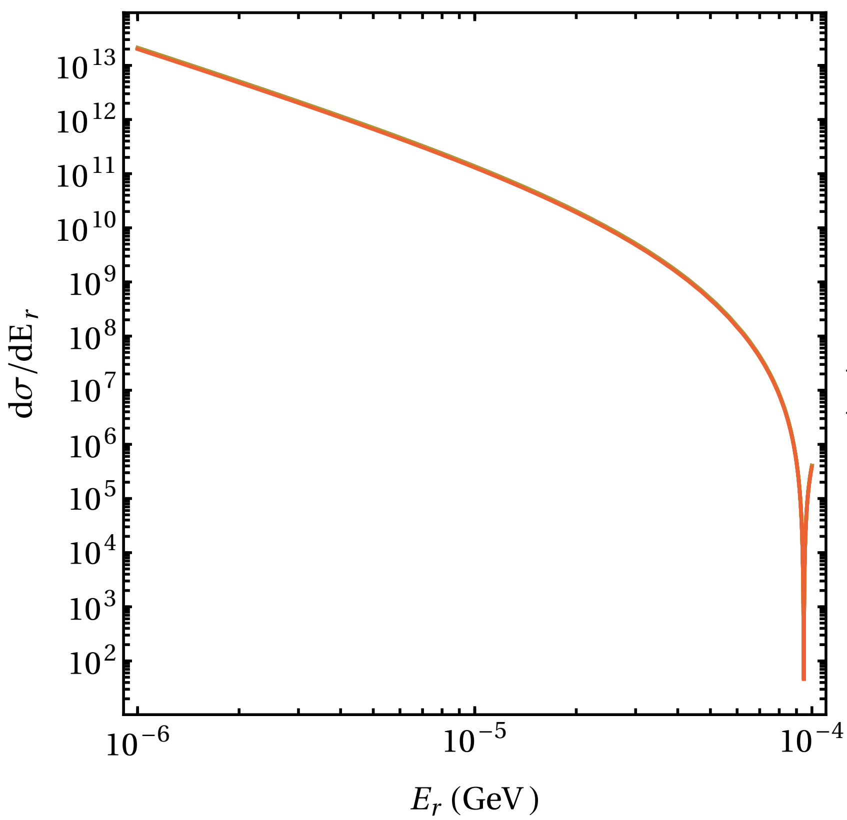
In[ ]:=
```
{With[{Tx1 = 1, mx = .1, MA = 1, gxi = 1},
  Show[LogLogPlot[{dσdErUp[ErGeV, Tx1, 131, mx, 0, gxi, MA] GeV^3,
      dσdErUp[ErGeV, Tx1, 131, mx, .001, gxi, MA] GeV^3,
      dσdErUp[ErGeV, Tx1, 131, mx, .01, gxi, MA] GeV^3,
      dσdErUp[ErGeV, Tx1, 131, mx, .1, gxi, MA] GeV^3}, {ErGeV, 10^-6, 10^-4},
    FrameTicks → {LogTicks[10^0, 10^14, 1], LogTicks[10^-6, 10^-4]},
    FrameLabel → {"E_r (GeV)", "dσ/dE_r"}],
   LogLogPlot[{dσdErDown[ErGeV, Tx1, 131, mx, 0, gxi, MA] GeV^3,
      dσdErDown[ErGeV, Tx1, 131, mx, .001, gxi, MA] GeV^3,
      dσdErDown[ErGeV, Tx1, 131, mx, .01, gxi, MA] GeV^3,
      dσdErDown[ErGeV, Tx1, 131, mx, .1, gxi, MA] GeV^3}, {ErGeV, 10^-6, 10^-4}]]]
 ,
 With[{Tx1 = 1, mx = .001, MA = .001, gxi = 1},
  Show[LogLogPlot[{dσdErUp[ErGeV, Tx1, 131, mx, 0, gxi, MA] GeV^3, dσdErUp[ErGeV, Tx1,
        131, mx, .001, gxi, MA] GeV^3, dσdErUp[ErGeV, Tx1, 131, mx, .01, gxi, MA] GeV^3,
      dσdErUp[ErGeV, Tx1, 131, mx, .1, gxi, MA] GeV^3}, {ErGeV, 10^-6, 10^-4},
    FrameTicks → {LogTicks[10^0, 10^14, 1], LogTicks[10^-6, 10^-4]},
    FrameLabel → {"E_r (GeV)", "dσ/dE_r"}],
   LogLogPlot[{dσdErDown[ErGeV, Tx1, 131, mx, 0, gxi, MA] GeV^3,
      dσdErDown[ErGeV, Tx1, 131, mx, .001, gxi, MA] GeV^3,
      dσdErDown[ErGeV, Tx1, 131, mx, .01, gxi, MA] GeV^3,
      dσdErDown[ErGeV, Tx1, 131, mx, .1, gxi, MA] GeV^3}, {ErGeV, 10^-6, 10^-4}]]]}
```



Out[ ]= {


dσ/dE_r
10^5
10^4
10^−6
10^−5
10^−4
E_r (GeV)

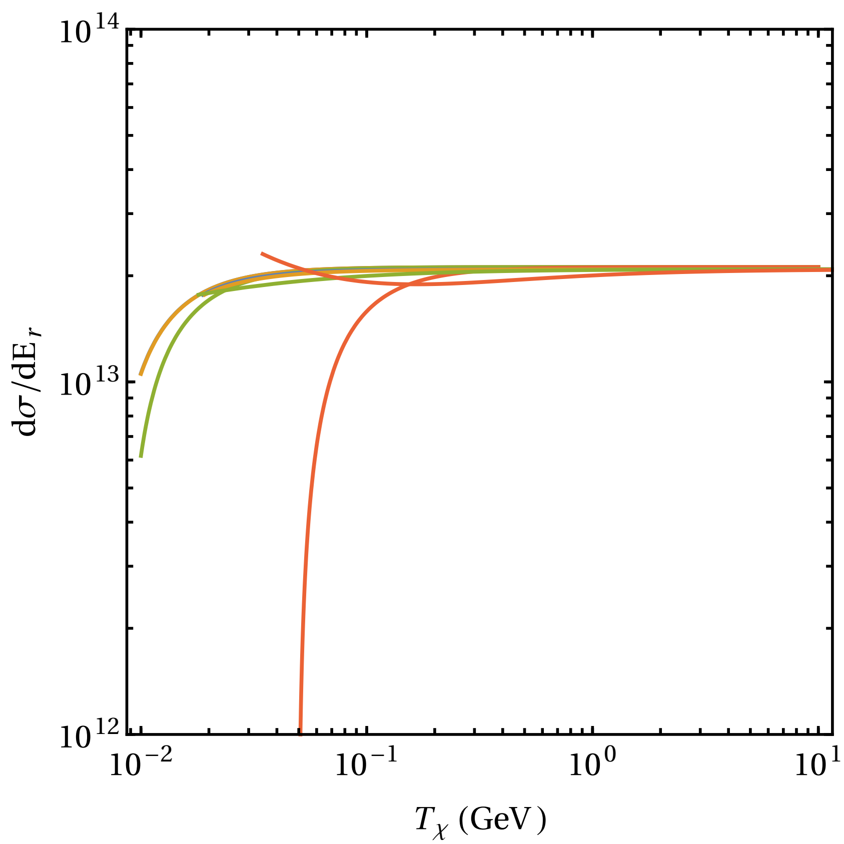
,


dσ/dE_r
10^13
10^12
10^11
10^10
10^9
10^8
10^7
10^6
10^5
10^4
10^3
10^2
10^−6
10^−5
10^−4
E_r (GeV)

}

```
In[ ]:= With[{ErGeV = 10^-6, mx = .001, MA = .001, gxi = 1},
  Show[LogLogPlot[{dσdErUp[ErGeV, Tx1, 131, mx, 0, gxi, MA] GeV^3,
     dσdErUp[ErGeV, Tx1, 131, mx, .001, gxi, MA] GeV^3,
     dσdErUp[ErGeV, Tx1, 131, mx, .01, gxi, MA] GeV^3,
     dσdErUp[ErGeV, Tx1, 131, mx, .1, gxi, MA] GeV^3}, {Tx1, 10^-2, 10^1},
    PlotRange → {10^12, 10^14},
    FrameTicks → {LogTicks[10^12, 10^14], LogTicks[10^-3, 10^3]},
    FrameLabel → {"Tχ (GeV)", "dσ/dEr"}],
   LogLogPlot[{dσdErDown[ErGeV, Tx1, 131, mx, 0, gxi, MA] GeV^3,
     dσdErDown[ErGeV, Tx1, 131, mx, .001, gxi, MA] GeV^3,
     dσdErDown[ErGeV, Tx1, 131, mx, .01, gxi, MA] GeV^3,
     dσdErDown[ErGeV, Tx1, 131, mx, .1, gxi, MA] GeV^3}, {Tx1, 10^-3, 10^3}]]]
```

Out[ ]=



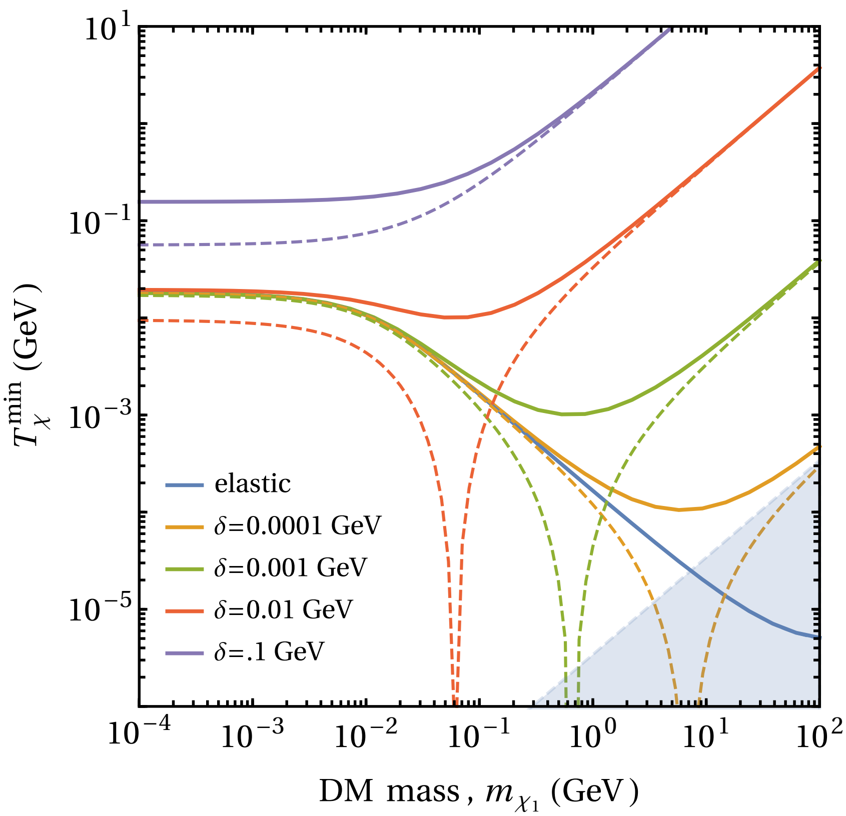
Kinematics Plots

```
In[ ]:= kinDatU = Table[
    {mx, TxMinEu[5 × 10^-6, 131, mx, δ]},
    {δ, {0, .0001, .001, .01, .1}},
    {mx, logSpace[.0001, 100, 30]}];
```

In[ ]:=
```
kinDatD = Table[
   {mx, TxMinEd[5 × 10^-6, 131, mx, δ]},
   {δ, {0, .0001, .001, .01, .1}},
   {mx, logSpace[.0001, 100, 100]}];
```

In[ ]:=
```
Show[
 ListLogLogPlot[kinDatU, Joined → True,
  PlotRange → {{10^-4, 100}, {10^-6, 10}},
  FrameTicks → {LogTicks[10^-6, 10^1, 2], LogTicks[10^-4, 10^2]},
  PlotLegends → Legend[{"elastic", "δ=0.0001 GeV",
     "δ=0.001 GeV", "δ=0.01 GeV", "δ=.1 GeV"}, Position → {.2, .2}],
  FrameLabel → {"DM mass, m_χ1 (GeV)", "T_χ^min (GeV)"}],
 ListLogLogPlot[kinDatD, Joined → True, PlotStyle → Dashed,
  PlotRange → {{10^-4, 100}, {10^-6, 10}}],
 LogLogPlot[1/2 mx VMAX^2, {mx, 10^-4, 100}, PlotStyle → None, Filling → Bottom]]
```

Out[ ]=



Find an expression for the minima in the exothermic case:

In[ ]:=
$$0 == \frac{\text{Er}\,(2\,\text{Er}\,\text{mT} - 4\,\text{mT}\,(\text{mx} + \delta) - \delta\,(2\,\text{mx} + \delta)) + \sqrt{\text{Er}\,(\text{Er} + 2\,\text{mT})\left(2\,\text{Er}\,\text{mT} + \delta^2\right)\left(2\,\text{Er}\,\text{mT} + (2\,\text{mx} + \delta)^2\right)}}{4\,\text{Er}\,\text{mT}} \;//$$

Solve[#, mx] &

Out[ ]=
$$\left\{\left\{\text{mx} \to \frac{-2\,\text{Er}\,\text{mT} - 2\,\text{Er}\,\delta + \delta^2}{2\,(\text{Er} - \delta)}\right\}\right\}$$

In[ ]:=
$$\text{mxTxMin[Er\_, mT\_, }\delta\text{\_] := } \frac{-2\,\text{Er}\,\text{mT} - 2\,\text{Er}\,\delta + \delta^2}{2\,(\text{Er} - \delta)}$$

$$\text{mxTxMin[Er\_, mT\_, }\delta\text{\_] := } \frac{\text{Er}\,\text{mT}}{(\delta)}$$

In[ ]:=
$$\frac{10 \times 10^{-6} \times 131\,\text{amu}}{(\delta)}$$

Out[ ]=
$$\frac{0.001220257270}{\delta}$$

In[∘]:=
```
Show[
  ListLogLogPlot[kinDatU, Joined → True,
   PlotRange → {{10^-4, 100}, {10^-6, 10}},
   FrameTicks → {LogTicks[10^-6, 10^1, 2], LogTicks[10^-4, 10^2]},
   PlotLegends → Legend[{"elastic", "δ=0.0001 GeV",
      "δ=0.001 GeV", "δ=0.01 GeV", "δ=.1 GeV"}, Position → {.2, .2}],
   FrameLabel → {"DM mass, m_χ1 (GeV)", "T_χ^min (GeV)"},
   GridLines → {{mxTxMin[5 × 10^-6, 131 amu, .01],
      mxTxMin[5 × 10^-6, 131 amu, .001], mxTxMin[5 × 10^-6, 131 amu, .0001]}, None}],
  ListLogLogPlot[kinDatD, Joined → True, PlotStyle → Dashed,
   PlotRange → {{10^-4, 100}, {10^-6, 10}}],
  LogLogPlot[1/2 mx VMAX^2, {mx, 10^-4, 100}, PlotStyle → None, Filling → Bottom]]
```

Out[∘]=