Correcting a cross section for a thermal target*

Documentation for LLNL's routines that thermally broaden a cross section for a heated target

Bret R. Beck
Lawrence Livermore National Laboratory

August 1, 2023

*This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract #W-7405-ENG-48.

1 Introduction

Most cross section data are given as a function of the incident particle's velocity or energy (e.g., $\sigma(E)$) assuming that the target is at rest. Most targets, while at rest on average, have a thermal motion which must be included to obtain the actual reaction rate between incident particle and target. This article explains how to correct - often called thermal broadening - a cross section for the thermal motion of the target (see section **??**) and describes [**?**] routines that can be used to thermally broaden a cross section (see section **??**).

2 Theory

For a material at temperature T, the effective cross section $\langle\sigma(v_{\rm n},T)\rangle$ for a reaction is defined as the cross section at incident velocity $v_{\rm n}$ (or energy $E_{\rm n}$) that gives the same reaction rate for a stationary target as the real cross section $\sigma(v_{\rm n})$ gives for a thermal target. For a target of density ρ and incident particle velocity $\mathbf{v}_{\rm n}$, or speed $v_{\rm n}$, the reaction rate is,

$$\rho\, v_{\rm n}\ \langle\sigma(v_{\rm n},T)\rangle = \int d\mathbf{v}_{\rm t}\rho\, v_{\rm r}\, \sigma(v_{\rm r})\, M(\mathrm{v_t}) \quad . \tag{1}$$

Here, $M(v_{\rm t})$ is the target velocity distribution function and $v_{\mathbf{r}} = |\mathbf{v}_{\rm n} - \mathbf{v}_{\rm t}|$ is the relative speed between the incident particle and a target with velocity $\mathbf{v}_{\rm t}$. The relative speed can also be written as,

$$v_{\rm r}^2 = v_{\rm n}^2 + v_{\rm t}^2 - 2\,\mu\, v_{\rm n} v_{\rm t} \quad , \tag{2}$$

where μ is the angle between the incident particle's velocity and the target's velocity. In Eq. **??** we can drop the factor ρ appearing on both sides of the equation, divide by $v_{\rm n}$ and replace $d\vec{v}_{\rm t}$ with $d\theta\, d\mu\, v_{\rm t}^2\, dv_{\rm t}$ to obtain,

$$\langle\sigma(E_{\rm n},T)\rangle = \frac{1}{v_{\rm n}}\int_{\vec{v}_{\rm t}} v_{\rm r}\,\sigma(E_{\rm r})\,M(v_{\rm t})\,d\vec{v}_{\rm t} = \frac{1}{v_{\rm n}}\int_0^{2\pi} d\theta \int_{-1}^{1} d\mu \int_0^{\infty} dv_{\rm t} v_{\rm t}^2 v_{\rm r}\,\sigma(E_{\rm r})\,M(v_{\rm t}) \quad . \tag{3}$$

Here, $\sigma(E_{\rm r})$ is written in terms of the incident particle's energy when its velocity is $v_{\rm r}$, $E_{\rm r} = m_{\rm n} v_{\rm r}^2/2$, as that is how the data are typically stored in nuclear databases. The θ integral yields 2π as no factor in the integrand depends on it. The μ integration is performed by using Eq. **??** to replace $d\mu$ by $dv_{\rm r}$, (i.e., $v_{\rm r}\, dv_{\rm r} = -v_{\rm t}\, v_{\rm n}\, d\mu$). Thus, the integral becomes,

$$\int_0^{2\pi} d\theta \int_{-1}^{1} d\mu \int_0^{\infty} dv_{\rm t} v_{\rm t}^2 v_{\rm r}\,\sigma(E_{\rm r})\,M(v_{\rm t}) = \frac{2\pi}{v_{\rm n}}\int_0^{\infty} dv_{\rm r}\, v_{\rm r}^2\,\sigma(E_{\rm r}) \int_{v_{\rm r}-v_{\rm n}}^{v_{\rm r}+v_{\rm n}} dv_{\rm t}\, v_{\rm t}\, M(v_{\rm t}) \tag{4}$$

where the integration limits for $v_{\rm t}$ have changed to be consistent with Eq. **??**. Using a Maxwellian for $M(v_{\rm t})$,

$$M(v_{\rm t}) = \left(\frac{m_{\rm t}}{2\pi T}\right)^{3/2} \exp(-m_{\rm t} v_{\rm t}^2/(2T)) \quad , \tag{5}$$

the integral over $v_{\rm t}$ yields,

$$\langle\sigma(E_{\rm n},T)\rangle = \frac{1}{v_{\rm n}^2 v_{\rm T}\sqrt{\pi}}\ \int_0^{\infty} dv_{\rm r}\, v_{\rm r}^2\,\sigma(E_{\rm r}) \left[\exp\left(-\left(\frac{v_{\rm r}-v_{\rm n}}{v_{\rm T}}\right)^2\right) - \exp\left(-\left(\frac{v_{\rm r}+v_{\rm n}}{v_{\rm T}}\right)^2\right)\right] \tag{6}$$

where $v_{\rm T} = \sqrt{2T/m_{\rm t}}$ and $m_{\rm t}$ is the target mass. This can be broken up into two terms, σ_- and σ_+. The σ_- term contains the factor $\exp(-((v_{\rm r}-v_{\rm n})/v_{\rm T})^2)$ and the other contains the factor $\exp(-((v_{\rm r}+v_{\rm n})/v_{\rm T})^2)$. The main contribution to σ_- is from the region $|v_{\rm r}-v_{\rm n}| < 5v_{\rm T}$ and the main contribution to σ_+ is from the region $v_{\rm r} < 5v_{\rm T}$.

3 Numerical algorithm

This section describes how to numerically integrate Eq. **??** for a cross section given as a function of the incident particle's energy.

3.1 Linear interpolated, point-wise cross section data

In nuclear databases, $\sigma(E_{\rm r})$ is often given at N points (i.e., E_i, σ_i) and intermediate values are interpolated using linear-linear interpolation. In other words, for $E_i < E < E_{i+1}$, $\sigma(E)$ is,

$$\sigma(E) = \left(\frac{\sigma_{i+1}-\sigma_i}{E_{i+1}-E_i}\right)(E-E_i)+\sigma_i = \left(\frac{\sigma_{i+1}-\sigma_i}{v_{i+1}^2-v_i^2}\right)\left(v^2-v_i^2\right)+\sigma_i \equiv s_i\left(v^2-v_i^2\right)+\sigma_i \quad . \tag{7}$$

Substituting this into Eq. **??** yields,

$$\begin{aligned}\langle\sigma(E_{\rm n},T)\rangle &= \frac{1}{v_{\rm n}^2 v_{\rm T}\sqrt{\pi}}\sum_{i=1}^{N-1}\int_{v_i}^{v_{i+1}} dv_{\rm r}\, v_{\rm r}^2\left(s_i\left(v_{\rm r}^2-v_i^2\right)+\sigma_i\right)\times \\ &\qquad \left[\exp\left(-\left(\frac{v_{\rm r}-v_{\rm n}}{v_{\rm T}}\right)^2\right)-\exp\left(-\left(\frac{v_{\rm r}+v_{\rm n}}{v_{\rm T}}\right)^2\right)\right] \quad .\end{aligned} \tag{8}$$

Breaking this into two parts, σ_- for the exponential with the term $v_{\rm r}-v_{\rm n}$ and σ_+ for the $v_{\rm r}+v_{\rm n}$ term, and substituting $x_- = (v_{\rm r}-v_{\rm n})/v_{\rm T}$ into the expression for σ_-, with a similar substitution for σ_+, produces

$$\langle\sigma_\mp(E_{\rm n},T)\rangle = \frac{1}{v_{\rm n}^2\sqrt{\pi}}\sum_{i=1}^{N-1}\int_{(v_i\mp v_{\rm n})/v_{\rm T}}^{(v_{i+1}\mp v_{\rm n})/v_{\rm T}} dx\, e^{-x^2}\, g_\mp(x) \tag{9}$$

where

$$\begin{aligned}g_\mp(x) &= s_i\, v_{\rm T}^4\, x^4 \pm 4\, s_i\, v_{\rm n}\, v_{\rm T}^3\, x^3 + \left(\sigma_i + s_i\left(6v_{\rm n}^2-v_i^2\right)\right) v_{\rm T}^2\, x^2 \\ &\quad \pm 2\, v_{\rm n}\left(\sigma_i + s_i\left(2v_{\rm n}^2-v_i^2\right)\right) v_{\rm T}\, x + v_{\rm n}^2\left(\sigma_i + s_i\left(v_{\rm n}^2-v_i^2\right)\right) \quad .\end{aligned} \tag{10}$$

Defining $s_{E,i} = (\sigma_{i+1}-\sigma_i)/(E_{i+1}-E_i)$ and $K = T\, m_{\rm n}/(E_{\rm n} m_{\rm t})$ the expression for $g_\mp(x)/v_{\rm n}^2$ becomes,

$$\begin{aligned}\frac{g_\mp(x)}{v_{\rm n}^2} &= s_{E,i}\, K^2\, E_{\rm n}\, x^4 \pm 4\, s_{E,i}\, K^{3/2}\, E_{\rm n}\, x^3 + \left(\sigma_i + s_{E,i}\left(6E_{\rm n}-E_i\right)\right) K\, x^2 \\ &\quad \pm 2\left(\sigma_i + s_{E,i}\left(2E_{\rm n}-E_i\right)\right) K^{1/2}\, x + \left(\sigma_i + s_{E,i}\left(E_{\rm n}-E_i\right)\right) \quad .\end{aligned} \tag{11}$$

3.1.1 Dealing with end points.

In equation **??** the thermally corrected cross sections for the last points (e.g., E_N, E_{N-1} or any point within $5\, v_{\rm T}$ of E_N) will be incorrect unless the integral is extended beyond the last given point E_N. There are three trivial extensions one can make (and maybe more). The simplest extension is to do nothing. For this extension it is easy to show that the last point will be low by about a factor-of-two, since the integral in Eq. **??** ignores the contribution for $E > E_N$.

A second possible extension is to assume that the cross section is flat for $E \geq E_N$ and to use the cross section at E_N as its value. In this case the extension yields,

$$\begin{aligned}\sigma_{\text{extension}} &= \frac{\sigma(E_N)}{v_n^2\,\sqrt{\pi}}\left[2\,v_{\text{T}}\,v_{\text{n}}\,e^{-x_2^2} + \int_{x1}^{x2} dx e^{-x^2}\left(v_{\text{T}}^2\,x^2 + 2\,v_{\text{T}}\,v_{\text{n}}\,x + v_{\text{n}}^2\right)\right] \\ &= \frac{\sigma(E_N)}{4\,v_{\text{n}}^2\,\sqrt{\pi}}\Bigg[\left(4\,v_{\text{T}}\,v_{\text{n}} + 2\,v_{\text{T}}^2 x_1\right)e^{-x_1^2} + \left(4\,v_{\text{T}}\,v_{\text{n}} + 2\,v_{\text{T}}^2 x_2\right)e^{-x_2^2} \\ &\quad + \sqrt{\pi}\left(2\,v_{\text{n}}^2 + v_{\text{T}}^2\right)\Big(\text{erf}(x_2) - \text{erf}(x_1)\Big)\Bigg] \end{aligned} \tag{12}$$

where $x_1 = (v_N - v_{\text{n}})/v_{\text{T}}$, $x_2 = (v_N + v_{\text{n}})/v_{\text{T}}$ and v_N is the incident particle's speed evaluated at energy E_N.

A third possible extension is to assume that the cross section is $1/v$ for $E \geq E_N$ and to use $\sigma(E > E_N) = \sigma(E_N)\,v_N/v(E)$. In this case the extension yields,

$$\begin{aligned}\sigma_{\text{extension}} &= \frac{\sigma(E_N)\,v_N}{v_{\text{n}}^2\,\sqrt{\pi}}\left[v_{\text{T}}\,e^{-x_2^2} + \int_{x1}^{x2} dx\,e^{-x^2}\,(v_{\text{T}}\,x + v_{\text{n}})\right] \\ &= \frac{\sigma(E_N)\,v_N}{2\,v_{\text{n}}^2\sqrt{\pi}}\left[v_{\text{T}}\left(e^{-x_2^2} + e^{-x_1^2}\right) + \sqrt{\pi}\,v_{\text{n}}\Big(\text{erf}(x_2) - \text{erf}(x_1)\Big)\right] \end{aligned} \tag{13}$$

3.2 Multi-group cross section data

In deterministic transport, the cross section data are divided into G groups with energy boundaries E_i and the cross section σ_g for group g is a constant. Using $v_i = \sqrt{2E_i/m_{\text{n}}}$, Eq. **??** becomes,

$$\langle\sigma_{\mp}(E_{\text{n}}, T)\rangle = \frac{1}{v_{\text{n}}^2\sqrt{\pi}}\sum_{i=1}^{G}\sigma_g\int_{(v_i \mp v_{\text{n}})/v_{\text{T}}}^{(v_{i+1}\mp v_{\text{n}})/v_{\text{T}}} dx\,e^{-x^2}\left(v_{\text{T}}^2\,x^2 \pm 2v_{\text{n}}\,v_{\text{T}}\,x + v_{\text{n}}^2\right) \quad . \tag{14}$$

This equation is the same as Eq. **??** with $s_i = 0$. Thus, $g_{\mp}(x)/v_{\text{n}}^2$ becomes,

$$\frac{g_{\mp}(x)}{v_{\text{n}}^2} = \sigma_i K\,x^2 \pm 2\,\sigma_i\,K^{1/2}\,x + \sigma_i \quad . \tag{15}$$

3.3 Evaluating integrals of the form $\int dx\,x^n \exp(-x^2)$

In sections **??** and **??**, integrals of the form

$$h_{\text{n}}(a,b) = \frac{1}{\sqrt{\pi}}\int_a^b dx\,x^n e^{-x^2} \tag{16}$$

are to be evaluated. These integrals are

$$\begin{aligned}\frac{1}{\sqrt{\pi}}\int dx\,e^{-x^2} &= \frac{\text{erf}(x)}{2} \\ \frac{1}{\sqrt{\pi}}\int dx\,xe^{-x^2} &= \frac{-e^{-x^2}}{2\sqrt{\pi}}\end{aligned}$$

$$\begin{aligned}
\frac{1}{\sqrt{\pi}} \int dx\, x^2 e^{-x^2} &= \frac{\mathrm{erf}(x)}{4} - \frac{x\, e^{-x^2}}{2\sqrt{\pi}} \\
\frac{1}{\sqrt{\pi}} \int dx\, x^3 e^{-x^2} &= \frac{-1}{2\sqrt{\pi}} \left(x^2+1\right) e^{-x^2} \\
\frac{1}{\sqrt{\pi}} \int dx\, x^4 e^{-x^2} &= \frac{3\,\mathrm{erf}(x)}{8} - \frac{x}{4\sqrt{\pi}} \left(2\,x^2+3\right) e^{-x^2} \quad . \qquad (17)
\end{aligned}$$

When using a computer to evaluate these expressions, the forms in Eq. **??** will not produce good results when $b-a$ is small. For small $b-a$ it is better to Taylor series expand the integrands about the mid point $x_0 = (b+a)/2$. Let $C_{n,i}(x_0)$ be the coefficient for the term $\exp(-x_0^2)\,(x-x_0)^i/i!$ and $\epsilon = b-a$, then the Taylor series is,

$$\begin{aligned}
\int_a^b dx \sum_{i=0}^{\infty} \frac{C_{n,i}(x_0)}{i!} \exp(-x_0^2)\,(x-x_0)^i &= \exp(-x_0^2) \sum_{i=0}^{\infty} \frac{C_{n,i}(x_0)}{i!} \int_{x_0+\epsilon/2}^{x_0+\epsilon/2} dx (x-x_0)^i \\
&= \exp(-x_0^2) \sum_{i=0 \text{ by } 2}^{\infty} \frac{C_{n,i}(x_0)\,\epsilon^{i+1}}{2^i\,(i+1)!} \qquad (18)
\end{aligned}$$

For even i to 6 the $C_{n,i}(x)$ are,
for n = 0,

$$\begin{aligned}
C_{0,0} &= 1 \\
C_{0,2} &= 2(2x^2-1) \\
C_{0,4} &= 4(4x^4-12x^2+3) \\
C_{0,6} &= 8(8x^6-60x^4+90x^2-15) \qquad (19)
\end{aligned}$$

for n = 1,

$$\begin{aligned}
C_{1,0} &= x \\
C_{1,2} &= 2x(2x^2-3) \\
C_{1,4} &= 4x(4x^4-20x^2+15) \\
C_{1,6} &= 8x(8x^6-84x^4+210x^2-105) \qquad (20)
\end{aligned}$$

for n = 2,

$$\begin{aligned}
C_{2,0} &= x^2 \\
C_{2,2} &= 2(2x^4-5x^2+1) \\
C_{2,4} &= 4(4x^6-28x^4+39x^2-6) \\
C_{2,6} &= 8(8x^8-108x^6+390x^4-375x^2+45) \qquad (21)
\end{aligned}$$

for n = 3,

$$\begin{aligned}
C_{3,0} &= x^3 \\
C_{3,2} &= 2x(2x^4-7x^2+3) \\
C_{3,4} &= 4x(4x^6-36x^4+75x^2-30) \\
C_{3,6} &= 8x(8x^8-132x^6+630x^4-945x^2+315) \qquad (22)
\end{aligned}$$

for n = 4,

$$\begin{aligned} C_{4,0} &= x^4 \\ C_{4,2} &= 2x^2(2x^4 - 9x^2 + 6) \\ C_{4,4} &= 4(4x^8 - 44x^6 + 123x^4 - 84x^2 + 6) \\ C_{4,6} &= 8(8x^{10} - 156x^8 + 930x^6 - 1935x^4 + 1170x^2 - 90) \end{aligned} \tag{23}$$

4 Heating a heated cross section

Often the cross section to be heated to temperature T as already been heated to temperature T_1. As long as the $T \geq T_1$, Eq. **??** is still valid provided [**?**] the temperature used in Eq. **??** is $T_2 = T - T_1$. This sections goes through the proof of this statement.

If a cross section heated to temperature T_1 is inserted into the right-hand-side of Eq. **??**, we have

$$\begin{aligned} \langle\sigma(v_\mathrm{n},T)\rangle &= \frac{1}{v_\mathrm{n}^2 v_{\mathrm{T}_2}\sqrt{\pi}} \int_0^\infty dv_\mathrm{r}\, v_\mathrm{r}^2\, \langle\sigma(v_\mathrm{r},T_1)\rangle \left[e^{-((v_\mathrm{r}-v_\mathrm{n})/v_{\mathrm{T}_2})^2} - e^{-((v_\mathrm{r}+v_\mathrm{n})/v_{\mathrm{T}_2})^2}\right] \\ &= \frac{1}{v_\mathrm{n}^2 v_{\mathrm{T}_2}\sqrt{\pi}} \int_0^\infty dv_\mathrm{r}\, v_\mathrm{r}^2 \left\{ \frac{1}{v_\mathrm{r}^2 v_{\mathrm{T}_1}\sqrt{\pi}} \int_0^\infty dv_{\mathrm{r}'}\, v_{\mathrm{r}'}^2\, \sigma(v_{\mathrm{r}'}) \right. \\ &\quad \left. \times \left[e^{-((v_{\mathrm{r}'}-v_\mathrm{r})/v_{\mathrm{T}_1})^2} - e^{-((v_{\mathrm{r}'}+v_\mathrm{r})/v_{\mathrm{T}_1})^2}\right] \right\} \left[e^{-((v_\mathrm{r}-v_\mathrm{n})/v_{\mathrm{T}_2})^2} - e^{-((v_\mathrm{r}+v_\mathrm{n})/v_{\mathrm{T}_2})^2}\right] \\ &= \frac{1}{v_\mathrm{n}^2\, v_{\mathrm{T}_2}\, v_{\mathrm{T}_1}\, \pi} \int_0^\infty dv_{\mathrm{r}'}\, v_{\mathrm{r}'}^2\, \sigma(v_{\mathrm{r}'}) \int_0^\infty dv_\mathrm{r} \\ &\quad \times \left[e^{-((v_{\mathrm{r}'}-v_\mathrm{r})/v_{\mathrm{T}_1})^2} - e^{-((v_{\mathrm{r}'}+v_\mathrm{r})/v_{\mathrm{T}_1})^2}\right] \left[e^{-((v_\mathrm{r}-v_\mathrm{n})/v_{\mathrm{T}_2})^2} - e^{-((v_\mathrm{r}+v_\mathrm{n})/v_{\mathrm{T}_2})^2}\right] \end{aligned} \tag{24}$$

The integration over v_r is

$$\int_0^\infty dv_\mathrm{r} \left[e^{-((v_{\mathrm{r}'}-v_\mathrm{r})/v_{\mathrm{T}_1})^2} - e^{-((v_{\mathrm{r}'}+v_\mathrm{r})/v_{\mathrm{T}_1})^2}\right] \left[e^{-((v_\mathrm{r}-v_\mathrm{n})/v_{\mathrm{T}_2})^2} - e^{-((v_\mathrm{r}+v_\mathrm{n})/v_{\mathrm{T}_2})^2}\right] \tag{25}$$

After substituting $v_\mathrm{r} \to -v_\mathrm{r}$ into two of the terms, one each of the positive and negative terms, this equation simplifies to

$$\int_{-\infty}^\infty dv_\mathrm{r} \left\{ \left[e^{-((v_{\mathrm{r}'}-v_\mathrm{r})/v_{\mathrm{T}_1})^2}\, e^{-((v_\mathrm{r}-v_\mathrm{n})/v_{\mathrm{T}_2})^2}\right] - \left[e^{-((v_{\mathrm{r}'}+v_\mathrm{r})/v_{\mathrm{T}_1})^2}\, e^{-((v_\mathrm{r}-v_\mathrm{n})/v_{\mathrm{T}_2})^2}\right] \right\} \tag{26}$$

Using the equation for the convolution of two Gaussians,[1] Eq. **??** integrates to

$$\frac{\sqrt{\pi}\, v_{\mathrm{T}_1}\, v_{\mathrm{T}_2}}{v_\mathrm{T}} \left[e^{-((v_{\mathrm{r}'}-v_\mathrm{n})/v_\mathrm{T})^2} - e^{-((v_{\mathrm{r}'}+v_\mathrm{n})/v_\mathrm{T})^2}\right] \tag{27}$$

[1] The equation for the convolutions of two Gaussians is

$$\int_{-\infty}^\infty dx' \frac{e^{-(x'-x_1)^2/(2\,\sigma_1^2)}}{\sqrt{2\pi}\,\sigma_1} \frac{e^{-(x-x')^2/(2\,\sigma_2^2)}}{\sqrt{2\pi}\,\sigma_2} = \frac{e^{-(x-x_1)^2/(2\,(\sigma_1^2+\sigma_2^2))}}{\sqrt{2\pi\,(\sigma_1^2+\sigma_2^2)}}$$

where $v_{\rm T} = \sqrt{v_{\rm T_1}^2 + v_{\rm T_2}^2} = \sqrt{2\,(T_1 + T_2)/m_{\rm n}} = \sqrt{2\,T/m_{\rm n}}$. Therefore, Eq. **??** becomes

$$\langle \sigma(v_{\rm n}, T) \rangle = \frac{1}{v_{\rm n}^2 v_{\rm T} \sqrt{\pi}} \int_0^\infty dv_{\rm r'}\, v_{\rm r'}^2\, \sigma(v_{\rm r'}) \left[\exp\left(-\left(\frac{v_{\rm r'} - v_{\rm n}}{v_{\rm T}} \right)^2 \right) - \exp\left(-\left(\frac{v_{\rm r'} + v_{\rm n}}{v_{\rm T}} \right)^2 \right) \right] \qquad (28)$$

and completes the proof.

5 Routine to thermally broaden a cross section

A C routine called crossSectionAdjustForHeatedTarget has been written to thermally broaden a piecewise, linear-linear cross section that is a function of the incident particle's energy. This section describes it and a python interface to it.

5.1 crossSectionAdjustForHeatedTarget C library

A library called libcrossSectionAdjustForHeatedTarget.a contains[2] the C-routine crossSectionAdjustForHeatedTarget for heating cross sections. The file crossSectionAdjustForHeatedTarget.h is its header file. The routine crossSectionAdjustForHeatedTarget has the following C declaration:

```
int crossSectionAdjustForHeatedTarget(
        crossSectionAdjustForHeatedTarget_limit lowerlimit,     /* Input */
        crossSectionAdjustForHeatedTarget_limit upperlimit,     /* Input */
        crossSectionAdjustForHeatedTarget_info *info,           /* Input/Output */
        double EMin,                                            /* Input */
        double massRatio,                                       /* Input */
        double T,                                               /* Input */
        double fInterpolation,                                  /* Input */
        int nPairs,                                             /* Input */
        double *E_cs_in,                                        /* Input */
        double **E_cs_out );                                    /* Output */
```

Here, **Input** (**Output**) means the argument is used for input (output). The cross section to be heated (see section **??** if it has already been heated) contains **nPairs** of piecewise, linear-linear (energy,cross section) points given by **E_cs_in**. For example, if at energies 1e-11, 1e-10, 1e-4, 1. and 20 the cross section is 12.3, 9.2, 9.1, 4.3 and 11.9 respectively, then a C calling routine could define **nPairs** and **E_cs_in** as:

```
int nPairs = 5;
double E_cs_in[10] = { 1e-11, 12.3, 1e-10, 9.2, 1e-4, 9.1, 1., 4.3, 20., 11.9 };
```

T is the temperature of the target (see section **??**), **massRatio** is the ratio of the target's mass to the incident particle's mass and **fInterpolation** is the desired accuracy of the heated, piecewise, linear-linear cross section (e.g., **fInterpolation** = 1e-3 means that the heated cross section data should be piecewise, linear-linear to 0.1%). Currenlty, **fInterpolation** is restricted to be between 10^{-6} and 0.1.

[2]This library also contains the external routine crossSectionAdjustForHeatedTarget_integrate_xn_qauss. This routine is only for testing and should not, in general, be used.

The arguments **EMin**, **T** and the energy values in **E_cs_in** must all be in the same energy unit (e.g., MeV). If the return value from crossSectionAdjustForHeatedTarget is positive, then it is the number of (energy, cross section) data pairs return in **E_cs_out**. The memory for **E_cs_out** is allocated by crossSectionAdjustForHeatedTarget and must be freed by the calling routine. If the return value from crossSectionAdjustForHeatedTarget is negative, then an has error occurred. The following table list the error values and their meaning:

```
 -1      nPair is less than 2.
 -2      massRatio is less than or equal to 0.
 -3      The first energy point is less than or equal to 0.
 -4      The temperature is less than or equal to 0.
 -5      The i^th energy value is greater than the i^th + 1 energy value.
 -6      Could not allocate internal memory.
 -7      Could not allocate memory for E_cs_out.
-11      Could not allocate internal memory.
```

The inputted cross section has domain $E_{\min}$ = E_cs_in[0] to $E_{\max}$ = E_cs_in[2 × nPairs - 2]. The arguments **lowerlimit** and **upperlimit** define how the cross section data should be extended below $E_{\min}$ and above $E_{\max}$, respectively (see section **??**). The type crossSectionAdjustForHeatedTarget_limit is a C enum with the following defined values:

crossSectionAdjustForHeatedTarget_limit_one_over_v: The cross section is extended as $1/v$.

crossSectionAdjustForHeatedTarget_limit_constant: The first (last) cross section point is used as the value below (above) the first (last) point.

crossSectionAdjustForHeatedTarget_limit_threshold: The cross section is zero outside the domain. For example, the reaction has a threshold at $E_{\text{threshold}}$.

The heated data domain is never extended above $E_{\max}$. However, if **lowerlimit** is **crossSectionAdjustForHeatedTarget_limit_threshold** the domain will be extended below $E_{\min}$ to the greater of **EMin** and E_{cutoff} where $E_{\text{cutoff}} = m_{\text{n}} v_{\text{cutoff}}^2/2$ and v_{cutoff} is the incident particle's velocity at $E_{\min}$ minus 5 times the target's thermal velocity v_{T}. If the heated domain must not extend below the unheated domain, set **EMin** = $E_{\min}$. Setting **EMin** > $E_{\min}$ is the same as setting **EMin** = $E_{\min}$.

The argument **info** is a C struct with the following elements:

```
int mode;                   /* Input */
int verbose;                /* Input */
int InfoStats;              /* Currently not used. */
int WarningStats;           /* Output */
int ErrorStats;             /* Currently not used. */
int bytes;                  /* Output */
int evaluations;            /* Output */
double fInterpolation;      /* Output */
```

Bits in the element **mode** are used to control how crossSectionAdjustForHeatedTarget choose the energy points to heat and whether the heated cross section data should be thinned. Currently, only two bits are used and they can be set with the following macros:

crossSectionAdjustForHeatedTarget_mode_all: By default, crossSectionAdjustForHeatedTarget does not heated every point in the input energy domain. Instead, it attempts to judiciously pick the points to obtain the desired requested accuracy via **fInterpolation** (e.g., for energies less than T / massRatio the heated cross section will vary approximately as 1/v, and for a given **fInterpolation** the required step size can be calculated). By judiciously picking points, the code can be much faster (a-factor-of 100 as been observed) if the unheated cross section contains many resonances (i.e., points). Or-ing this macro with **mode** causes every point in the unheated data to be heated. Independent of how the points are picked, crossSectionAdjustForHeatedTarget always test if a point should be added between to consecutive energy points based on **fInterpolation**. And it iterates this until one is not required.

crossSectionAdjustForHeatedTarget_mode_do_not_thin: By default, the returned cross section is thinned. Or-ing this macro with **mode** results in the data not being thinned.

This routines as 3-levels[3] of messages (informative, warning and error). The number of occurrences of each is returned by **mode** elements **InfoStats**, **WarningStats** and **ErrorStats**. In addition, if **mode** element **verbose** is greater than 0, then a message is printed for each error. If it is greater than 1, a message is printed for each error and warning. And, if it is greater than 2, a message is printed for each error, warning and informative. If memory allocation fails, then **mode** element **bytes** is set to the number of bytes that were requested.

The **mode** element **evaluations** is set to the number of points heated and the element **fInterpolation** is set to the actual fInterpolation used.

5.2 Python interface to the crossSectionAdjustForHeatedTarget library

The routine crossSectionAdjustForHeatedTarget can be called from python via the module crossSectionAdjustForHeatedTarget. This module has one function, also called crossSectionAdjustForHeatedTarget, which has the effective[4] python definition of:

```
def crossSectionAdjustForHeatedTarget( massRatio, T, E_cs_in,
    lowerlimit = 'constant', upperlimit = 'oneOverV', fInterpolation = 2e-3,
    heatAllPoints = False, doNotThin = False, EMin = 1e-11 )
```

Currently, E_cs_in must be a python list of [E,$\sigma(E)$] points. Using the example on page **??**, E_cs_in would be:

```
    E_cs_in = [ [ 1e-11, 12.3 ], [ 1e-10, 9.2 ], [ 1e-4, 9.1 ],
                [ 1., 4.3 ], [ 20., 11.9 ] ]
```

The arguments **massRatio**, **T**, **lowerlimit**, **upperlimit**, **fInterpolation** and **EMin** are as per the C-routine's crossSectionAdjustForHeatedTarget arguments. However, the arguments **lowerlimit** and **upperlimit** require string values, and they must be one of the following: 'constant', 'oneOverV' or 'threshold'. Finally, the arguments **heatAllPoints** and **doNotThin** are used to set crossSectionAdjustForHeatedTarget's **mode** argument.

The heated data is returned as a list of [E,$\sigma(E)$] points. If crossSectionAdjustForHeatedTarget returns an error, as raise is executed.

[3] Currently, only warning is implemented.

[4] Effective because it is also a C-routine implementing python functions.

References

[1] There is another heating package called SIGMA1 written by D. Cullen that heats ENDF formatted files. It is not described here but can be found at http://www-nds.iaea.org as part of the PREPRO package.

[2] Private communications with Dermott (Red) Cullen.