Description of utility-programs for scattering problem in the DBSR complex (folder SCT_JJ)

SEC_DCS_JJ

Description:	provides differential and angle-integrated (ordinary and momentum transfer) cross sections for given transition
Input files:	**zarm.tma** (or zarm.tmb, tmat.done), **target_jj**
Output files:	**dcs_ii_ff, mt_ii_ff**
Call as:	**sec_dif_JJ** itr1=ii itr2=jj i16=-1|0|. ifano=0|1 ek1=... ek2=... or ekk=… Glow=... Ghigh=... Gstep=... dcs=0|1 tdone=0|1 JJ_extend=…

All arguments are optional.

itr1 [1]	index of initial state (default - 1)
itr2 [1]	index of final state (default - 1)
i16 [16]	i16 - controls the output units: = 0 - sigma in a.u. > 0 - sigma in 10^{-i16} cm^2
ifano [0]	= 0 - Condon-Shortly phase convention, default = 1 - Fano phase convention
ek1 [0]	if > 0, restriction on minimum electrovn energy (in Ry)
ek2 [0]	if > 0, restriction on maximum electron energy (in Ry)
ekk [0]	if > 0, exact electron energy (ek1=ek2=ekk) (output in **tmat.done_inp)**
Glow [0]	lowest scattering angle
Ghigh [180]	highest scattering angle
Gstep [1]	step fort scattering angle
dcs [1]	if = 0, skip the calculations of differential cross sections
tdone [0]	if =1, redirect input from **zarm.tma** to **tmat.done** file
JJ_extend [0]	if > 0, extrapolate T-matrix elements to JJ_extend value (input **tmat.done_inp ->** output **tmat.done_out)**

The utility SEC_DIF_JJ first check **zarm.tma** (zarm.tmb) file and create **tmat.done** file with T-matrix elements, specific for the given transition. The tmat.done file has the same format as in program MJK (Grum-Grzhimailo 2003). If **ekk** parameter is not equal 0, the program additionally analyzes the T-matrix elements for the given energy and prepares them for extrapolation to higher J-values. To do it, the program first divided all matrix elements on subsets with the same changes of involved l- and j-values. The values in subsets are supposed to reduce as in geometric series. The corresponding coefficients are found as ratio of two highest values in the series, T(n)/T(n-1). This information is recorded in the **tmat.done_inp** file and the program stops. The user may check the extrapolation coefficients (in the end of the **tmat.done_inp** file) and rerun the program with **JJ_extend** parameter. The program with extrapolate the T-matric coefficients with 2J values up two JJ_extend. The resulting T-matrix elements are recorded in **tmat.done_out** file and program stops. The user may check extrapolated data and copy this file to **tmat.done**. Then, in order to get differential cross sections, he can use SEC_DIF_JJ with **tdone=1** option, or any other program, which employ the **tmat.done** input. Note that for high J-values of J (> 50), the SEC_DIF_JJ program may take too much time due to big number of A_λ coefficients (see below). In this case, it is advised to use SEC_DIF_JJ_ampl program (described below), which is much faster.

Related theory:

JJ –coupling (neutral case)

$$\sigma = \frac{\pi a_0^2}{2(2J_0+1)k_0^2}\sum_{l_0 l_1 j_0 j_1 J}(2J+1)\left|T^{\pi J}(J_0 l_0 j_0 \to J_1 l_1 j_1)\right|^2$$

$$\frac{d\sigma}{d\Omega} = \frac{a_0^2}{8(2J_0+1)k_0^2}\sum_{\lambda\ge 0}A_\lambda P_\lambda(\cos\theta) = N\sum_{\lambda\ge 0}A_\lambda P_\lambda(\cos\theta)$$

$$A_\lambda = \sum_{\substack{l_0 l_1 j_0 j_1 J\\ l'_0 l'_1 j'_0 j'_1 J'}}(-1)^{J_0-J_1} i^{l_0-l_1+l'_0-l'_1}\hat{l}_0\hat{l}'_0\,\hat{l}_1\hat{l}'_1\,\hat{j}_0\hat{j}'_0\,\hat{j}_1\hat{j}'_1\,(2J+1)(2J'+1)$$

$$\times(l_0 0 l'_0|\lambda 0)(l_1 0 l'_1|\lambda 0)\begin{Bmatrix} j_0 & j'_0 & \lambda\\ l'_0 & l_0 & \frac{1}{2}\end{Bmatrix}\begin{Bmatrix} j_1 & j'_1 & \lambda\\ l'_1 & l_1 & \frac{1}{2}\end{Bmatrix}\begin{Bmatrix} j_0 & j'_0 & \lambda\\ J' & J & J_0\end{Bmatrix}\begin{Bmatrix} j_1 & j'_1 & \lambda\\ J' & J & J_1\end{Bmatrix}$$

$$\times T^{\pi J}(J_0 l_0 j_0 \to J_1 l_1 K_1)\left[T^{\pi' J'}(J_0 l'_0\, j'_0 \to J_1 l'_1\, j'_1)\right]^*$$

$\hat{a} = \sqrt{2a+1}$; $\quad i^{l_0-l_1+l'_0-l'_1}$ - Fano factor, used only for Fano phase convention

Angle-integrated and momentum-transfer cross section can be express as:

$$\sigma = 4\pi\, N\, A_0 \qquad \sigma_{MT} = 4\pi\, N\,(A_0 - 1/3A_1)$$

JJ –coupling (Coulomb case)

DCS can be expressed as pure Coulomb scattering, interference term and potential term (first two only for elastic scattering, when initial state, *i*, equal final, *f*).

$$\frac{d\sigma_c}{d\Omega} = \frac{(Z/k_0)^2}{4k_0^2\sin^4\theta/2}\delta(i,f)$$

$$\frac{d\sigma_{\text{int}}}{d\Omega} = \frac{(z/k_0)\delta(i,f)}{4(2J_0+1)k_0^2\sin^2\theta/2}\operatorname{Re}\left\{e^{-i(z/k_0)\ln[\sin^2(\theta/2)]})\sum_l Q_l P_l(\cos\theta)\right\}$$

$$Q_l = \sum_{J\pi} e^{i2(\sigma_l-\sigma_0)}(2J+1)T_{ll}^{J\pi}(\alpha_0 J_0 \to \alpha_0 J_0) \qquad \text{(should be checked)}$$

$$\frac{d\sigma_M}{d\Omega} = \frac{a_0^2}{8(2J_0+1)k_0^2}\sum_{\lambda\ge 0}A_\lambda^{Coul}P_\lambda(\cos\theta); \qquad A_\lambda^{Coul} = A_\lambda \times e^{i(\sigma_{l_0}+\sigma_{l_1}-\sigma_{l'_0}-\sigma_{l'_1})}$$

SEC_DCS_JJ_AMPL

Description:	provides differential and angle-integrated (ordinary and momentum transfer) cross sections for given transition
Input files:	**zarm.tma** (or zarm.tmb, tmat.done), **target_jj**
Output files:	**dcs_ii_ff, mt_ii_ff**
Call as:	**sec_dif_JJ_ampl** itr1=ii itr2=jj i16=-1|0|. ifano=0|1 ek1=... ek2=... or ekk=… Glow=... Ghigh=... Gstep=... dcs=0|1 tdone=0|1 JJ_extend=…

The utility SEC_DIF_JJ_AMPL used the direct calculations of scattering amplitude instead of analytical approach implemented in SEC_DIF_JJ. It has the same input argument and the same file structure. The SEC_DIF_JJ_AMPL turned out to be much faster then SEC_DIF_JJ, especially for big T-matrix sets with high maximum J-values (50 and more).

Related theory:

$$f_{01}(J_0,M_0,\mu_0 \to J_1,M_1,\mu_1;\theta,\varphi) = f_c + f_m = \delta(all)\frac{(z/k_0)}{2k_0\sin^2(\theta/2)}e^{i2\sigma_0}e^{i(z/k_0)\ln[\sin^2(\theta/2)]}$$

$$-i\sqrt{\frac{\pi}{k_0k_1}}\sum_{l_0,l_1,j_0,j_1,J,\pi} i^{(l_0-l_1)}\sqrt{(2l_0+1)}e^{i(\sigma_{l_0}+\sigma_{l_1})}C^{l_0\frac{1}{2}j_0}_{0\mu_0m_{j_0}}C^{l_1\frac{1}{2}j_1}_{m_1\mu_1m_{j_1}}C^{J_0j_0J}_{M_0m_{j_0}M}C^{J_1j_1J}_{M_1m_{j_1}M}\times T^{J\pi}(\alpha_0J_0l_0j_0\to\alpha_1J_1l_1j_1)Y_{l_1m_1}(\theta,\varphi)$$

(relative phase between the Coulomb and potential part ($\pm i$) is different in different publications)

$$Y_l^m(\theta,\varphi) = \Theta_{lm}(\theta)\Phi_m(\varphi)$$

$$\Phi_m(\varphi) = \sqrt{\frac{1}{2\pi}}e^{im\varphi}$$

$$\Theta_{lm}(\theta) = (-1)^m\sqrt{\frac{(2l+1)}{2}\frac{(l-m)!}{(l+m)!}}P_l^m(\cos\theta) = (-1)^m\overline{P}_l^m(\cos\theta), \qquad (m>0)$$

$$\Theta_{l-m}(\theta) = (-1)^m\Theta_{lm}(\theta) = \overline{P}_l^m(\cos\theta)$$

Considering $\varphi=0$, we may rewrite expression as

$$f_{01}(J_0,M_0,\mu_0 \to J_1,M_1,\mu_1;\theta,\varphi) = f_c + f_m = \delta(all)\frac{(z/k_0)}{2k_0\sin^2(\theta/2)}e^{i2\sigma_0}e^{i(z/k_0)\ln[\sin^2(\theta/2)]}$$

$$-i\sqrt{\frac{1}{2k_0k_1}}\sum_{l_0,l_1,j_0,j_1,J,\pi} i^{(l_0-l_1)}\sqrt{(2l_0+1)}e^{i(\sigma_{l_0}+\sigma_{l_1})}C^{l_0\frac{1}{2}j_0}_{0\mu_0m_{j_0}}C^{l_1\frac{1}{2}j_1}_{m_1\mu_1m_{j_1}}C^{J_0j_0J}_{M_0m_{j_0}M}C^{J_1j_1J}_{M_1m_{j_1}M}\times T^{J\pi}(\alpha_0J_0l_0j_0\to\alpha_1J_1l_1j_1)\overline{P}_{l_1}^{m_1}(\theta)$$

($\overline{P}_{l_1}^{m_1}(\theta)$ **-** is given by program ALEGFM**)**

$$\frac{d\sigma}{d\Omega} = \frac{1}{2(2J_0+1)} \frac{k_1}{k_0} \sum_{M_0, M_1, \mu_0, \mu_1} \left| f(\theta, \varphi) \right|^2$$

So, we may use direct calculation of scattering amplitude for different M's and them sum their modules. Angle-integrated cross sections (if needed) are then obtain direct integration of $d\sigma$ by angles.

$$\sigma_{ij} = \int_0^{2\pi} d\phi \int_0^{\pi} \sin\theta d\theta \frac{d\sigma_{ij}}{d\Omega}(\theta) \qquad \sigma_M = \int_0^{2\pi} d\phi \int_0^{\pi} \sin\theta (1-\cos\theta) d\theta \frac{d\sigma}{d\Omega}(\theta)$$