Short description of EIKONXS program

June 4, 2010

EIKONXS - eikonal (classical trajectory) program can accept adiabatic or diabatic molecular data, uses definition of the CTF of Errea et al.[1] and can compute a classical trajectory conserving energy for several choices of initial condition, or just use a straight line. This program is based on the philosophy of Pampa (Salin and Piacentini)[2] but uses an integration method due to Billing and Baer[3]. Author: R.J. Allan STFC, Daresbury Laboratory, Daresbury, Warrington WA4 4AD

For using this program your need a computer with *gcc*, *make* and any *Fortran 90* compiler.

Theoretical background

EIKONXS uses the impact parameter approximation to study of ion-atom collisions, which is valid when the energy of collision is greater than electronic excitation energy. For describing the electronic motion, molecular wavefunction is expanded over the finit basis set. This molecular approach is suitable for velocity of collision much smaller then velocity of the electron motion.

Let's look at the collision of particles A and B. The Hamiltonian of the system is given by

$$H = H_{el} + \frac{Z_A Z_B}{R(t)},$$

where Z_A and Z_B are the charges of A and B respectively, $R(t)$ defines the relative position of A and B, H_{el} is the Hamiltonian of the motion of the electrons in the field of the nuclei. To find the wavefunction for electronic states, we need to solve the time-dependent Schrodinger equation

$$\left(H - i\frac{d}{dt}\right)\Psi'(\mathbf{r},t) = 0,$$

here $\mathbf{r}$ - the set of electronic coordinates. For now, let us reject the internuclear potential, because there is no electronic coordinates in it, or just use the following transformation with the same result:

$$\Psi'(\mathbf{r},t) = \Psi(\mathbf{r},t)\exp\left(-iZ_AZ_B\int\limits_t \frac{dt'}{R(t')}\right).$$

Doing so, we get for $\Psi(\mathbf{r},t)$ equation

$$\left(H_{el} - i\frac{d}{dt}\right)\Psi(\mathbf{r},t) = 0. \tag{1}$$

The time-dependent electronic wavefunction is approximated by the expansion

$$\Psi(\mathbf{r},t) = \sum_{j=1}^{N} b_j(t)\chi_j\left(R(t),\mathbf{r}\right) \tag{2}$$

with time-dependent complex amplitudes $b_j(t)$ for the occupation of states $\chi_j\left(R(t),\mathbf{r}\right)$. In the straight-line impact parameter approximation we put

$$\mathbf{R}=\boldsymbol{\rho}+\boldsymbol{v}t=\boldsymbol{\rho}+z\hat{\boldsymbol{v}}.$$

Taking into account that $\chi_j\left(R(t),\mathbf{r}\right)$ are the eigenstates of H_{el} with eigenvalues $\epsilon_j(R)$ and that $z=vt$, we can rewrite equation (2) as

$$\Psi(\mathbf{r},t)=\sum_{j=1}^{N}a_j(z)\chi_j\left(R(z),\mathbf{r}\right)\exp\left(-\frac{i}{v}\int_{z_0}^{z}\epsilon_j(R(z'))dz'\right). \tag{3}$$

Inserting equation (3) into (1), the Schrodinger equation is reduced to the system of coupled equations for the occupation amplitudes $a_j(z)$

$$\frac{d}{dz}a_j(z)=-\sum_{k=1}^{N}a_k(z)\left\langle\chi_j\left|\frac{d}{dz}\right|\chi_k\right\rangle\exp\left(-\frac{i}{v}\int_{z_0}^{z}(\epsilon_k-\epsilon_j)dz'\right),\qquad i=1,2,\dots,N. \tag{4}$$

Now transform matrix elements in (4) by using

$$\frac{d}{dz}=\frac{dR}{dz}\frac{d}{dR}+\frac{d\theta}{dz}\frac{d}{d\theta},$$

where θ is the angle between $\boldsymbol{v}$ and $\boldsymbol{R}$. Also we replace $\frac{d}{d\theta}$ by iL_y, where L_y is the component of the total electronic orbital angular momentum operator along a direction perpendicular to the plane of collision, and get

$$\left\langle\chi_j\left|\frac{d}{dz}\right|\chi_k\right\rangle=\frac{z}{R}\left\langle\chi_j\left|\frac{d}{dR}\right|\chi_k\right\rangle+\frac{\rho}{R^2}\left\langle\chi_j\left|iL_y\right|\chi_k\right\rangle. \tag{5}$$

Using (5), we finally produce the semiclassical coupled equations

$$\frac{d}{dz}a_j(z)=-\sum_{k=1}^{N}a_k(z)\left(\frac{z}{R}\left\langle\chi_j\left|\frac{d}{dR}\right|\chi_k\right\rangle+\frac{\rho}{R^2}\left\langle\chi_j\left|iL_y\right|\chi_k\right\rangle\right)\exp\left(-\frac{i}{v}\int_{z_0}^{z}(\epsilon_k-\epsilon_j)dz'\right),\; i=1,2,\dots,N.$$

In the matrix form this system of equations can be rewritten as

$$\dot{a}=U(R(z))a,$$

where a is a column vector and U is the coupling matrix. This system is integrated from $-|z_{max}|$ to final value $|z_{max}|$ by using a matrix propagator method [3] with several choices of initial conditions $a_j(-|z_{max}|)=\delta_{jk},\ i,k=1,\dots,N$. And as a result of calculations, we have a set of $a_j(|z_{max}|)$ which determinate amplitudes of transitions to j-state from different k-states.

The total cross section is given by

$$\sigma_{i\to f}(v)=2\pi\int_0^\infty bP_{i\to f}(b)db,$$

where the probability $P_{i\to f}(b)$ is calculated from the a_f as

$$P_{i\to f}(b)=\left|a_f(|z_{max}|)\right|^2.$$

Molecular input

Let me explain how to use this program in a very simple way, jast to get a total cross-sections for charge exchange between different molecular states. First of all, before start to use this program your need to have *energy curves* for a set of molecular states (adiabatic states) and *couplings* between them (also adiabatic). To put your molecular data in the program you must use *readubu.f* file containing READ1 subroutine. This subroutine must read the following data:

- NE (integer) - number of points in energy curves
- NS (integer) - number of states
- NC (integer) - number of couplings (NC=NS*(NS-1)/2)
- R(I) (real array) - internuclear distances in the energy curves; I=1,NE
- E(I,J) (real array) - energies; I denote to internuclear distance, J is a number of state.
- NR (integer) - number of points in couplings
- BRR(I) (real array) - internuclear distances in the couplings curves; I=1,NR[1]
- TR(I,J) (real array) - transition elements (couplings) between different states; I denote to internuclear distance, J defines pair of states in the following way: if, for example, you have four molecular states E1, E2, E3, E4, than

  ```
  J=1 defines 1-2 coupling
  J=2         1-3
  J=3         1-4
  J=4         2-3
  J=5         2-4
  J=6         3-4
  ```

There is a possibility to take into account Common Translation Factor (CTF) of Errea et al. To do this, you must modify your input data by filling E2(I,J) and TR2(I,J) arrays in addition. But you need $\left\langle \Psi_k \left| z\frac{d}{dz} \right| \Psi_n \right\rangle$ and $\left\langle \Psi_k \left| x^2 \right| \Psi_n \right\rangle$ for this. In my calculation I always put 0 instead this matrix elements.

eikonxs.con

If you successfully put your molecular data in the READ1 subroutine, then you need to write a little file (eikonxs.con) to describe and control calculations. This is an example of this file for adiabatic calculations:

```
1     8 diffxs
2     APROB
3     0.
4     2
5     .01
6     .1
7       1  1.0
8     470  0.1
9     -49.
10    49.
11    0.1
```

[1]If you want to do diabatic calculation, you must care about this conditions: NE=NR,BRR(1)=R(1),BRR(NR)=R(NR).

```
12    0.8
13    49.
14      0  0
15    -2.1448181-2.1261778-2.1225612-2.1237669
16      0  0  0  0
17    F
18    0.
19    TOTAL
20    STOP
```

Let look through this file line by line:

1. This is a trash from previous parallel version of this program, you can put anything here
2. APROB means we want to do an *adiabatic* calculation
3. This is also trash from the old version of this program, just put here zero any time (variable isn't using during calculation)
4. Integer number defines number of velocities you want to calculate (we want 2)
5. velocity 1 in atomic units
6. velocity 2 in atomic units
7. Integer number defines how many sets of impact parameter we want to calculate, and then we put initial point for impact parameter (here we start from 0.1 as in molecular input)
8. we want 470 points[2] of impact parameter with a step 0.1[3]
9. Starting R distance for calculation, usually should be a little smaller then maximum R in molecular input
10. Ending R distance for calculation, usually should be the same as in line 9
11. Step in R propagation (here from -49 to 49), significantly effect in time of calculations
12. Reduced mass of the system
13. One defines eikonal phase for calculation, must be the same or greater then maximum R in line 9 and 10.
14. Two integer number define initial conditions for calculation (NCOL1 and NCOL2). If we put ncol1=0 (and anything in ncol2), then program will be using a straight line trajectory; in this case it works fine without any errors and relatively fast. If ncol1=1,2,...,NS and ncol2=0,1,...,NS, then program uses conserving energy classical trajectory and makes averaging of the result over initial state ncol1[4]; it takes a long time for calculations and crashes time to time.
15. Asymptotic energies of molecular states
16. Projection of angular momentum of electon on internuclear axis for different molecular states, 0 for Σ states and 1 for Π states

[2]You should care about number of points and step, 0.1+470*0.1 mustn't be greater then numbers in 9 and 10 line

[3]If you want, for example, 2 sets impact parameters, the next line should describe the second set (number of points and step).

[4]I'm not fully understand this trick in the program

17. TRUE/FALSE, if we need or not intermediate output of the program
18. the same as line 3
19. TOTAL means us want total cross-sections
20. STOP the program

This is a simple output of the program:

```
*****PROGRAM.TOTAL*****
RELATIVE VELOCITY 0.10000E-01 [AU] 1.9984 [eV] TOTAL CROSS SECTIONS [CM~2*10~-16]
471 IMPACT PARAMETERS FROM 0.10000 TO 47.100 [ao]
0.0100000 1.9984111 1 1 2262.3910109
0.0100000 1.9984111 1 2    1.2437946
0.0100000 1.9984111 1 3    0.0114241
0.0100000 1.9984111 1 4    0.3550571
0.0100000 1.9984111 2 1    1.2438064
0.0100000 1.9984111 2 2 3758.5819163
0.0100000 1.9984111 2 3    0.5607343
0.0100000 1.9984111 2 4    3.4922183
0.0100000 1.9984111 3 1    0.0114212
0.0100000 1.9984111 3 2    0.5607542
0.0100000 1.9984111 3 3  909.1109100
0.0100000 1.9984111 3 4    5.5976539
0.0100000 1.9984111 4 1    0.3550649
0.0100000 1.9984111 4 2    3.4921678
0.0100000 1.9984111 4 3    5.5976354
0.0100000 1.9984111 4 4 3787.1198912
*****PROGRAM.TOTAL*****
RELATIVE VELOCITY 0.10000 [AU] 199.84 [eV] TOTAL CROSS SECTIONS [CM~2*10~-16]
471 IMPACT PARAMETERS FROM 0.10000 TO 47.100 [ao]
0.1000000 199.8411146 1 1  672.3875136
0.1000000 199.8411146 1 2   87.8643503
0.1000000 199.8411146 1 3   11.6180805
0.1000000 199.8411146 1 4    7.7344885
0.1000000 199.8411146 2 1   87.8638022
0.1000000 199.8411146 2 2 2085.2939069
0.1000000 199.8411146 2 3   73.4646589
0.1000000 199.8411146 2 4   20.7079418
0.1000000 199.8411146 3 1   11.6181271
0.1000000 199.8411146 3 2   73.4629910
0.1000000 199.8411146 3 3  395.3864244
0.1000000 199.8411146 3 4    6.6237661
0.1000000 199.8411146 4 1    7.7345859
0.1000000 199.8411146 4 2   20.7091689
0.1000000 199.8411146 4 3    6.6223980
0.1000000 199.8411146 4 4 1783.6553846
```

Each line of output contains velocity of collision, full energy of collision in center of mass system, initial and final states, total cross-section in $10^{-16} cm^2$.

There is a possibility to do calculation in *diabatic* representation. For this purpose you need to make a diabatisation of your molecular input by using DIAB module of eikonxs. After diabatisation all other numbers are the same as in adiabatic case except DPROB instead APROB. The example of *eikonxs.con* for diabatic calculations is here.

```
8 diffxs
DIAB                                       <- using DIAB module
0.                                         <- always zero
-2.1448181-2.1261778-2.1225612-2.1237669   <- asymptotic energies
  0  0  0  0                               <- 1/0 for Σ/Π states
100.                  <- defines step in R, if it is bigger then step is smaller
F                                          <- intermediate output
50.0                                       <- Maximum R in input data
DPROB
0.
2
.01
.1
  1  1.0
470  0.1
-49.
49.
0.1
0.8
49.
  0  0
-2.1448181-2.1261778-2.1225612-2.1237669
  0  0  0  0
F
0.
TOTAL
STOP
```

At the end I want to say, that at this moment it is better to use straight line trajectory calculation (but at this case there is no mass dependence). In another case program some time crashes (but there is a mass dependence), i try to fix it now. Also diabatic calculations take less time than adiabatic, but result is the same if the step of propagation and in impact parameter is rather small (for example 0.1). If the step is rather big (for example 0.5), then adiabatic calculations gives different result in comparison with diabatic (they in 2-2.5-3 times bigger); but diabatic calculation still gives reasonable answer.

Compile and run

As I said at the beginning, to compile eikonxs you mast have computer with *linux OS, gcc*, *make* and *Fortran 90* compiler. Before compile the program set a correct name of fortran compiler (f90,f95,gfortran,ifort,...) of your system in *makefile:*

```
FC=f90 -c
LD=f90 -o
```

To easy compile the program just command

```
make Absoft
```

in the folder with sources of the program, and you will get executable file *eikonxs.out.* After that just run it by

```
./eikonxs.out
```

command. The output of the program also can be redirected to file by

```
./eikonxs.out > nameOFfile
```

for subsequent analysis using, for example, *grep* utility.

The program is written using infernal mixing of fortran 77 and 90, and consist of files with *.f, *.f90 and *.F extensions. If you want to change something in the source of program, you must change *.F files (if there are several files with same name but different extensions), if no such file, then change *.f90 file, and the last candidate for changing is *.f file. After that type

```
make clean
```

and again

```
make Absoft && ./eikonxs.out
```

References

[1] L F Errea et al 1994 J. Phys. B: At. Mol. Opt. Phys. 27 3603

[2] C. GAUSSORGUES, R.D. PIACENTINI, A. SALIN COMP. PHYS. COMMUN. 10 (1975) 223

[3] G. D. BILLING, M. BAER CHEM. PHYS. LETT. V.48 N.2 (1977) 37