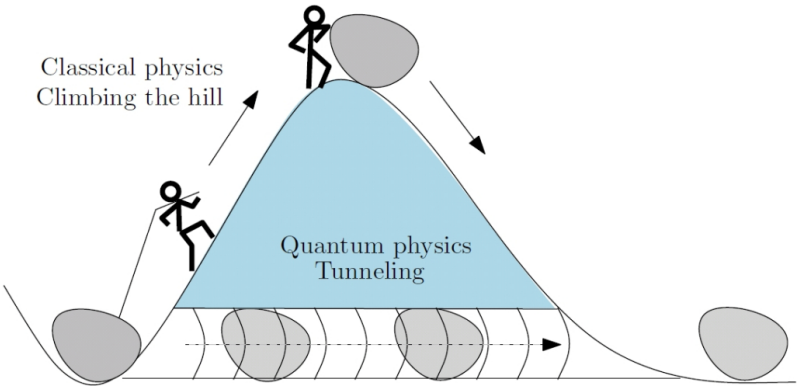
kSC.py – User Guide

v2021.03.29

Marc Serra Peralta



Description

The kSC.py script can determine the rate constant (and other parameters) including tunneling effect of any reaction. It incorporates several types of potential barriers, from squared to Eckart potentials. This manual provides an overview of the usage of the script and a description of the input, output and keywords.

Contents

1 General Overview

`kSC.py` is an open source script to numerically calculate the rate constant (and other parameters) of a given reaction including the tunneling correction. The package can be found in the following *GitHub* repository

```
https://github.com/MarcSerraPeralta/k_tunneling
```

The instructions of installation and the requisites (specially *Python 3*) are described in the `README.md` and `requirements.txt` files.

This manual is the user guide that provides an overview of the script usage and it contains command and input examples for a better understanding. For more information about the rate constants expressions and the numerical aspects of the script, please read the `documentation/theoretical_background.pdf` file in the *GitHub* repository.

For a quick summary of this manual, run the following command

```
python kSC.py --help_adv
```

It is more comfortable to add the repository's path in `$PATH` variable (in Linux distributions it is usually stored at `/etc/environment`). Then, the previous command is just

```
kSC.py --help_adv
```

and this is going to be the nomenclature for all the manual's commands.

1.1 What does `kSC.py` include?

The script includes the tunneling probability $p(E)$, the tunneling coefficient $\kappa(T)$ and the rate constant $k(T)$.

Table 1: List of all the potential barriers included in `kSC.py`.

$p(E)$	$\kappa(T)$	$k(T)$
classical	classical	classical
squared	squared	squared
	parabolic	parabolic
Eckart	Eckart	Eckart
	Eckart approx.	Eckart approx.
	Wigner	Wigner

1.2 What does `kSC.py` need for the calculations?

In order to calculate any of the parameters mentioned in the previous section, the `kSC.py` only needs the energy of the reactants (E_r), the energy of the transition state ($E^\ddagger$), the energy of the products(E_p), the imaginary frequency of the TS ($\nu^\ddagger$) and the $\Delta G^\ddagger$ from the reactants to the transition state. All energies must include the Zero Point Energy (ZPE).

2 List of Useful Commands and Tips

`kSC.py --help_adv`

Prints the `kSC.py` general information (input file structure, units and types of barriers included)

`kSC.py -n input.csv -log`

Calculates the rate constants specified in `input.csv` and saves the data in Arrhenius format (i.e. $1/T$ and $\ln k$), so it is easier to create a plot in Excel or LibreOffice.

`kSC.py -gaussian reactant.out TS.out product.out -o input.csv`

Creates input file for `kSC.py` from Gaussian output files.

`kSC.py -gaussian_T input_name.gjf -T Tmin Tmax DeltaT`

Creates Gaussian input file for multiple thermochemical calculations at different temperatures (i.e. from `Tmin` to `Tmax` in `DeltaT` steps).

`sed -i 's/0 1/0 3/g' input_name.gjf`

Replace all `0 1` in `input_name.gjf` by `0 3`.

`Steps to save Excel file to CSV format`

Click `this link` and go to *How to save CSV file as UTF-8 in Microsoft Excel?*.

`Steps to open CSV file in Excel`

Click `this link` or `this other one` and follow the steps.

3 Input Structure

The input file is based on the *Comma Separated Value* (CSV) format, which is a representation of a table where the cells are identified by commas. Therefore the decimal symbol must be the dot '.', for example `3.14159` (instead of `3,14159`).

The structure of the input is the following:

line 1 List of the potential barriers to calculate. The possible options are:

`classical`, `squared`, `parabolic`, `eckart`, `eckart-approx`, and `wigner`

The names have to be separated by a space, for example

`eckart eckart-aprox wigner`.

lines ≥ 2 Each line must have the values of T, E_r, $E^\ddagger$, E_p, $\nu^\ddagger$, G_r and $G^\ddagger$ in Hartrees. The values have to be separated by commas: `T,Er,ETS,Ep,freqTS,Gr,GTS`.

There is no limit in the number of lines.

Important!

Energies must include the Zero Point Energy (ZPE). The units must be Hartrees, K and cm^{-1} for Energy, Temperature and Frequency, respectively.

Tip

If only the $\Delta G^\ddagger$ is known, it can be specified by $G_r = 0$ and $G^\ddagger = \Delta G^\ddagger$.

Example of input

The example file `documentation/input_example.csv` can be found in the *GitHub* repository. A shorter example is shown in Table 2.

Table 2: Input data to be fed into `kSC.py` (left) and CSV input file (right). The units are Hartree, K and cm^{-1}. The origin of energies does not have to be the reactants.

Barriers: Eckart, Wigner						
T	E_r	$E^\ddagger$	E_p	$\nu^\ddagger$	G_r	$G^\ddagger$
10	0	0.0185	-0.0037	1300	0	0.019
15	0	0.0185	-0.0037	1300	0	0.019

$\Rightarrow$

eckart wigner
10,0,0.0185,-0.0037,1300,0,0.019
15,0,0.0185,-0.0037,1300,0,0.019

3.1 Create input file from Gaussian output file

Option 1: using `kSC.py`

Run the following command to create the input file:

```
kSC.py -gaussian reactant.out TS.out product.out -o input.csv
```

This command can be used with Gaussian output files that contain multiple thermochemical calculations in several temperatures.

Important!

Thermochemical calculations of all the three ouputs must be done in the same temperature (or same range of temperatures).

Tip

The sign of the $\nu^{\ddagger}$ is not important and can be either positive or negative. Nevertheless, when using `-gaussian` argument the frequencies in `input.csv` must be negative (meaning that they are imaginary). If not, this means that the TS calculations may have not found a TS.

Option 2: using `grep`

Another option is using the `grep` command to find the desired values (T, E, G, $\nu^{\ddagger}$):

```
grep -i 'Temperature ' output_gaussian.out
grep -i 'Sum of electronic and zero-point Energies=' output_gaussian.out
grep -i 'Sum of electronic and thermal Free Energies=' output_gaussian.out
grep -i 'Frequencies --' output_gaussian.out
```

Nevertheless, for Gaussian output files that contain multiple thermochemical calculations, the `kSC.py` option is recommended.

Create Gaussian input for multiple thermochemical calculations

The Gaussian input template for the multiple thermochemical calculations can be created by the command:

```
kSC.py -gaussian_T input_name.gjf -T Tmin Tmax DeltaT
```

and then the keywords inside `input_name.gjf` can be changed using the Replace tool of any text editor. For example, if the range of temperatures to be studied is 10, 15, 20 and 25 K, then the command would be

```
kSC.py -gaussian_T input_name.gjf -T 10 25 5
```

then if the desired spin multiplicity is 3 and the D3 empirical dispersion has to be included, the following replacements have to be done: `Replace all: '0 1' by '0 3'` and `'B3LYP/6-311G(d,p)' by 'B3LYP/6-311G(d,p) empiricaldispersion=GD3BJ'`.

This can be done on console using the following command (replacing `'0 1' by '0 3'`)

```
sed -i 's/0 1/0 3/g' input_name.gjf
```

Tip

To calculate the rate constant at different temperatures, create an input file with the previous commands and then copy the cells changing the temperature (see the section of Excel or LibreOfficle Calc to know how to easly open and modify the file). Remember that this is an approximation as $\Delta G^{\ddagger}$ varies with temperature. For a more precise calculation use the multiple thermochemical calculations in Gaussian (using `link1`).

3.2 Create input file from Crystal output file

Not implemented yet.

3.3 Create input file using Excel

1. Write the potential barriers to calculate separated by spaces in cell A1.
2. Write the values of T, E_r, $E^{\ddagger}$, E_p, $\nu^{\ddagger}$, G_r and $G^{\ddagger}$ in Hartrees (in this order and using decimal point '.') in the cells starting from A2.
3. Save the file following the steps from this link (go to *How to save the CSV file as UTF-8 in Microsoft Excel?*).

3.4 Create input file using LibreOffice Calc

1. Create an empty file named `input_name.csv` in a folder.
2. Open this file using LibreOffice Calc.

   Important!

   Specify `UTF-8` as a *Character set* and `Separated by Tab, Comma, Semicolon` in *Separation options*.

3. Write the potential barriers to calculate separated by spaces in cell A1.
4. Write the values of T, E_r, $E^{\ddagger}$, E_p, $\nu^{\ddagger}$, G_r and $G^{\ddagger}$ in Hartrees (in this order and using decimal point '.') in the cells starting from A2.
5. Save the file as CSV.

4 Selecting Calculation

4.1 Crossover Temperature

The crossover temperature (T_x) is calculated for every row in the input file with the following command:

```
kSC.py -Tx -n input.csv
```

and the T_x is printed on console (each line corresponds to each row of the input file).

Tip

For the crossover temperature, only $\Delta E^\ddagger = E^\ddagger - E_r$ and $\nu^\ddagger$ are needed so the rest of the cells can be 0 (but not empty).

4.2 Tunneling Probability

In this calculation, the temperature column of the input file is used as the energy input (because $p(E)$). The command to be used is:

```
kSC.py -n input.csv -prob [-o output_prob.csv] [-log]
```

where `-o` argument can be omitted, although it is not recommended. The default output file name is the input name with `_output` added at the end (for example, in this case the output name would be `input_output.csv`). With the `-log` argument, the probability is saved as E and $\ln p$.

Tip

The $\Delta G^\ddagger = G^\ddagger - G_r$ is not needed so these cells can be set to 0 (but not empty).

4.3 Tunneling Coefficient

The coefficient probability is calculated using the following command:

```
kSC.py -n input.csv -coeff [-o output_coeff.csv] [-log]
```

where `-o` argument works the same as in tunneling probability and, with the `-log` argument, the coefficient is saved as $1/T$ and $\ln \kappa$.

Tip

The $\Delta G^\ddagger = G^\ddagger - G_r$ is not needed so these cells can be set to 0 (but not empty).

4.4 Rate Constant

The coefficient probability is calculated using the following command:

```
kSC.py -n input.csv [-o output_k.csv] [-log]
```

where `-o` argument works the same as in tunneling probability and, with the `-log` argument, the rate constant is saved as $1/T$ and $\ln k$.

5 Output Structure

The output file is also written in CSV format. Data is arranged in columns and labelled on top, while the order of the barriers is preserved from the input. The option `-log` saves the data in Arrhenius format (for the rate constant and tunneling coefficient): $1/T$ and $\ln k$, so it is easier to plot in Excel or LibreOffice Calc.

The units of the output file are K for Temperature, eV for Energy and s^{-1} for Rate Constant (the tunneling probability and coefficient are dimensionless).

Important!

When using `-log` option, there may occur indefinite errors due to $\log(0) \to -\infty$ and the output file may have `-inf` (specially when temperatures are low). To solve this problem, just delete these `-inf` cells.

5.1 Open output file with Excel

Follow the instructions from one of these links

1. Microsoft Support
2. Steps with images

Important!

Be aware that the decimal point in Excel is the comma ',' but in the CSV format it is the dot '.'. In order to solve any problems, it is possible to replace all dots '.' to commas ',' in Excel.

5.2 Open output file with LibreOffice Calc

1. Open this file using LibreOffice Calc.

Important!

Specify `UTF-8` as a *Character set* and `Separated by Tab, Comma, Semicolon` in *Separation options*.

Tip

If some calculations or plots are performed, save the file as `.ods` to keep the formulas and plots (otherwise, they will be deleted in the CSV file).