Excitation Dynamics of Ag20 Nanocluster from Surface Hopping Simulations

Jicun Li
Brandeis University
July 1, 2023

Contents

- Background/context
- Practice details
- Remarks

Context

- **Noble metal nanoclusters have been widely used in sensing and imaging, catalysis, energy conversion, non-linear optics, ...**
- **Lack of understanding of their structure-property relationships limits their applications.**
- **Quantum chemical modeling can provide useful insights into the relationships.**

Context

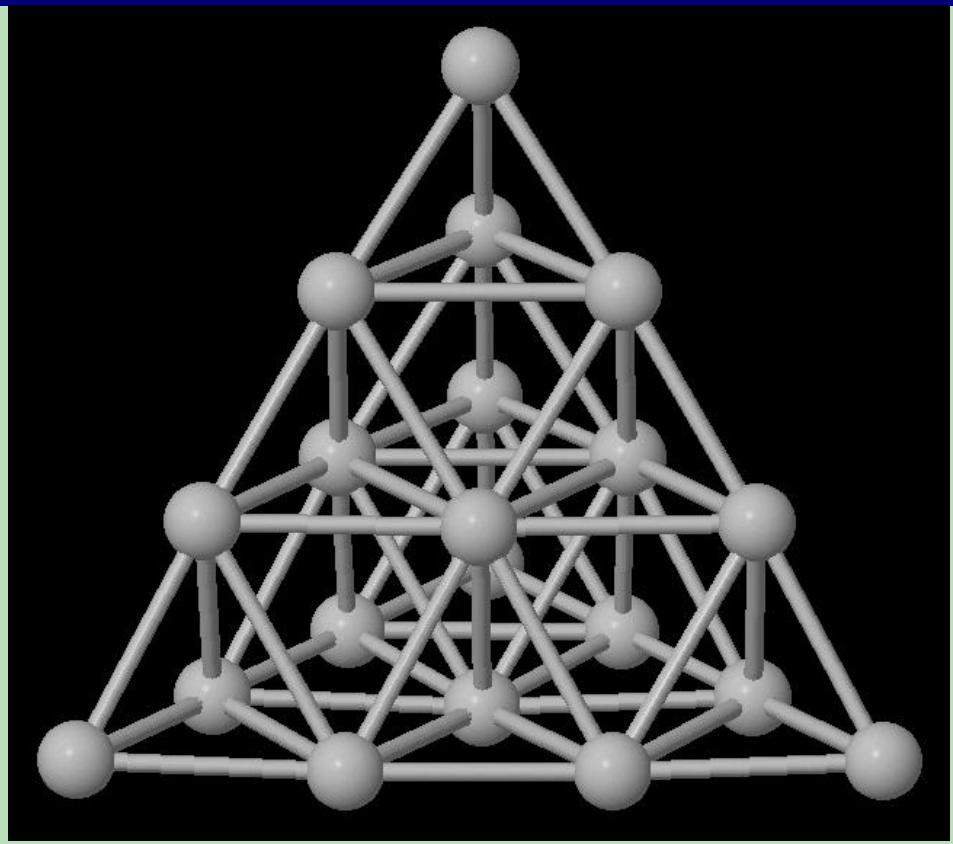
- **Most of the applications involve the excited-state properties.**
- **Understanding the excitation dynamics is essential for modifying or designing novel structures**
- **Surface hopping is a simple but widely-used methods for excitation dynamics studies**

Aim

- **Run SH simulations for Ag cluster**
- **Understand the concepts in SH**
- **Practice the procedure of SH**

Ag20 cluster

- neutral, singlet
- A widely-used model system
- Extracted from Ag crystal
- 4 layers with 1, 3, 6, 10 atoms in each layer
- bond length ~2.8 Å



Electronic structure calculations

- **DFTB**
- **Parameter set: `hyb-0-2`**
- **Dispersion: UFF**

`https://dftb.org/parameters/download/hyb/hyb-0-2-cc`

Steps of SH simulation

- 1. MD+TD
  - MD: trajectory snapshots
  - TD: excitation information
- 2. NACS: couplings between snapshots
- 3. mapping: Slater determinate
- 4. NAMD: SH part
- 5. analysis: decay kinetics

Step 1: MD

```
#   This file is to run the MD using DFTB+

Geometry = GenFormat {
    <<< "x1.gen"
}

Driver = VelocityVerlet {
    TimeStep [fs] = 0.5
    Steps = 200000

    MovedAtoms = 1:-1
    KeepStationary = Yes

    Thermostat = None {
        InitialTemperature [K] = 300
    }
}

Hamiltonian = DFTB {
    SCC = Yes
    Charge = 0

    SlaterKosterFiles = Type2FileNames {
        Prefix = "./hyb-0-2/"
        Separator = "-"
        Suffix = ".skf"
    }
    MaxAngularMomentum = {
        Ag = "d"
    }

    Dispersion = LennardJones {
        Parameters = UFFParameters{}
    }
}
```

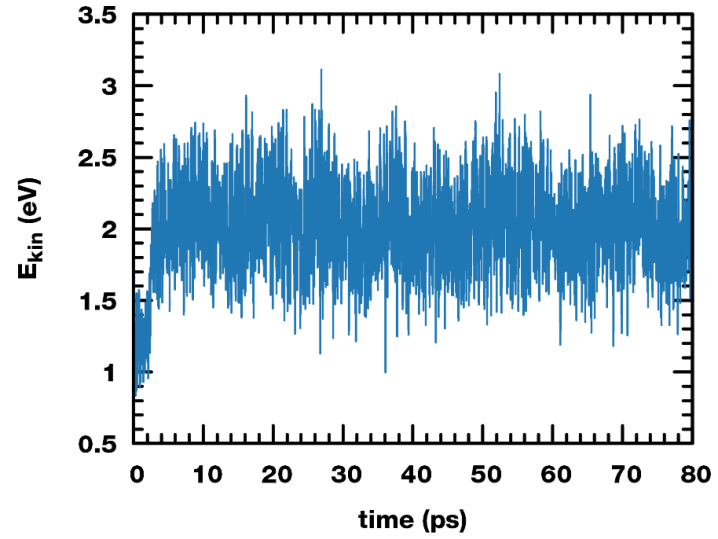
100 ps

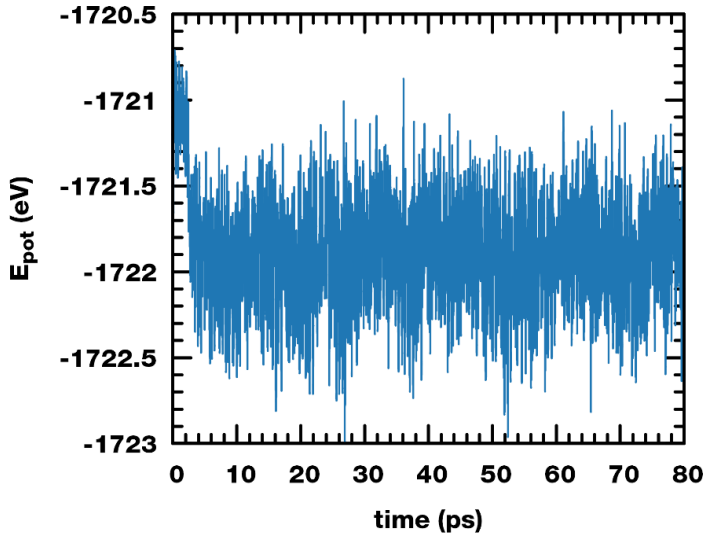
All mobile

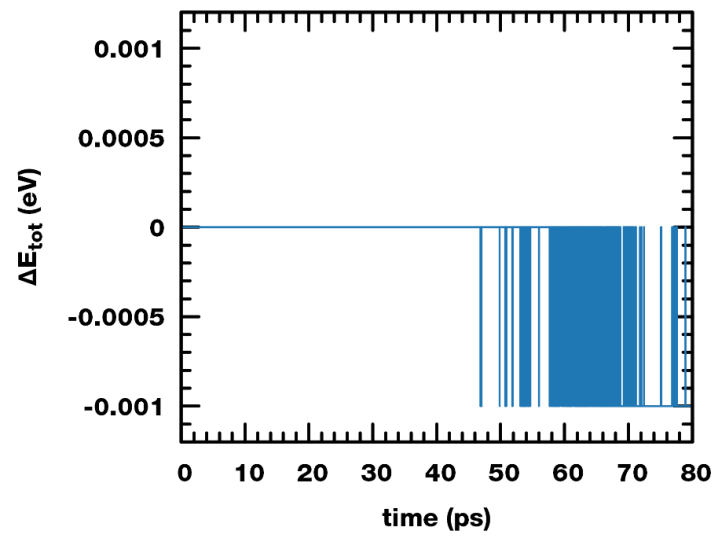
NVE from 300 K

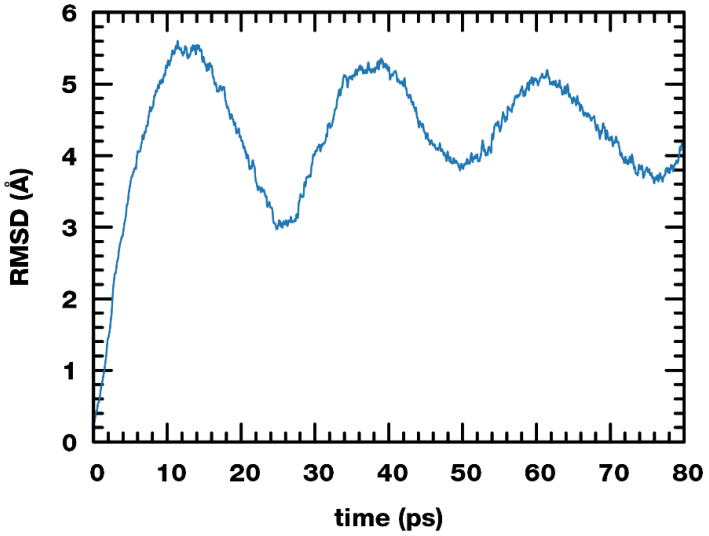
MD analysis

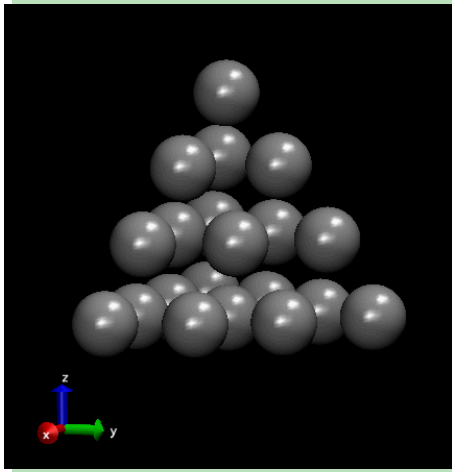
```
> bash mdinfo.bsh
```











Step 1: TD

- information of excited states
  - HOMO, LUMO
  - active space of MO for NACS
  - mapping of excited states
- First 10 excited states
- Every 25 fs

Transition analysis

> band.out → HOMO 110, LUMO 111
> bash excinfo.bsh
  time: 41000-43000 fs
  MO: 108-117 → 103-123

Step 2: NACS

```
import os
import sys

# Fisrt, we add the location of the library to test to the PYTHON path
if sys.platform=="cygwin":
    from cyglibra_core import *
elif sys.platform=="linux" or sys.platform=="linux2":
    from liblibra_core import *
from libra_py.packages.dftbplus import methods as DFTB_methods
import libra_py.workflows.nbra.step2_dftb as step2
from libra_py import units

params = { "EXE":"dftb+",
           "mo_active_space":list(range(103, 123)),
           "md_file":"Ag20-md.xyz",
           "sp_gen_file": "x1.gen",
           "ovlp_gen_file": "x2.gen",
           "syst_spec" : "C",
           "scf_in_file" : "dftb_in_ham1.hsd",
           "hs_in_file" : "dftb_in_ham2.hsd",
           "ovlp_in_file" : "dftb_in_overlaps.hsd",
           "do_tddftb": False,
           "dt":0.5*41.0, "isnap":82000, "fsnap":86000, "out_dir":"res",
           "tol":0.5
         }

step2.run_step2(params)
```

Step 3: mapping

```
> bash excinfo.bsh
```

```
9839        S9 =[ 9,-6, 7,-7, 8,-8];
9840        S10=[ 6,-6, 7,-7, 8,-8];
9841    41000.000  108  117    0  0  0  0  0  0  0 -1 -2 -2
9842        S0 =[ 6,-6, 7,-7, 8,-8];
9843        S1 =[ 6,-6, 7,-7, 9,-8];
9844        S2 =[ 6,-6, 7,-7,10,-8];
9845        S3 =[ 6,-6, 7,-7,11,-8];
9846        S4 =[ 6,-6, 7,-7,12,-8];
9847        S5 =[ 6,-6, 7,-7,13,-8];
9848        S6 =[ 6,-6, 7,-7,14,-8];
9849        S7 =[ 6,-6, 7,-7,15,-8];
9850        S8 =[ 6,-6, 9,-7, 8,-8];
9851        S9 =[ 9,-6, 7,-7, 8,-8];
9852        S10=[ 6,-6,10,-7, 8,-8];
9853    41050.000  108  117    0  0  0  0  0  0  0 -1 -1 -2
9854        S0 =[ 6,-6, 7,-7, 8,-8];
9855        S1 =[ 6,-6, 7,-7, 9,-8];
9856        S2 =[ 6,-6, 7,-7,10,-8];
9857        S3 =[ 6,-6, 7,-7,11,-8];
9858        S4 =[ 6,-6, 7,-7,12,-8];
9859        S5 =[ 6,-6, 7,-7,13,-8];
9860        S6 =[ 6,-6, 7,-7,14,-8];
9861        S7 =[ 6,-6, 7,-7,15,-8];
9862        S8 =[ 6,-6, 9,-7, 8,-8];
9863        S9 =[ 6,-6,10,-7, 8,-8];
9864        S10=[ 9,-6, 7,-7, 8,-8];
9865    41100.000  108  117    0  0  0  0  0  0  0 -1 -2 -2
9866        S0 =[ 6,-6, 7,-7, 8,-8];
9867        S1 =[ 6,-6, 7,-7, 9,-8];
9868        S2 =[ 6,-6, 7,-7,10,-8];
9869        S3 =[ 6,-6, 7,-7,11,-8];
9870        S4 =[ 6,-6, 7,-7,12,-8];
9871        S5 =[ 6,-6, 7,-7,13,-8];
9872        S6 =[ 6,-6, 7,-7,14,-8];
9873        S7 =[ 6,-6, 7,-7,15,-8];
9874        S8 =[ 6,-6, 9,-7, 8,-8];
9875        S9 =[ 9,-6, 7,-7, 8,-8];
9876        S10=[ 6,-6,10,-7, 8,-8];
9877    41150.000  108  117    0  0  0  0  0  0  0 -1 -1 -2 -2
9878        S0 =[ 6,-6, 7,-7, 8,-8];
9879        S1 =[ 6,-6, 7,-7, 9,-8];
9880        S2 =[ 6,-6, 7,-7,10,-8];
```

```
16  # Set variables based on your data in step2. Indexing is from 1
17  num_alpha_ks_orbs = 20 # Number of alpha spin-orbtials in the alpha spin-block
18                         # By extension, this is also the number of beta spin-or
19  start_time  = 82000    # Start reading step2 data at this index
20  finish_time = 85997  # Stop  reading step2 data at this index
21
22  data_dim  = num_alpha_ks_orbs # Total number of rows or columns in the step2 d
23  act_sp    = range(data_dim)   # Consider every spin-orbital to be in our activ
24
25  # Make a parameters dictionary with the relevant information about the step2 d
26  params = {"data_set_paths" : [res_dir],
27            "data_dim":data_dim, "active_space":act_sp,
28            "isnap":start_time,  "fsnap":finish_time,
29           }
30
31  # These files contain N x N matrices
32  params.update( {"read_S_data" : 0,
33                  "S_data_re_prefix": "S_",  "S_data_re_suffix": "_re",
34                  "S_data_im_prefix": "S_",  "S_data_im_suffix": "_im",
35                  "read_St_data" : 1,
36                  "St_data_re_prefix": "St_",  "St_data_re_suffix": "_re",
37                  "St_data_im_prefix": "St_",  "St_data_im_suffix": "_im",
38                  "read_hvib_data" : 1,
39                  "hvib_data_re_prefix": "hvib_",  "hvib_data_re_suffix": "_re"
40                  "hvib_data_im_prefix": "hvib_",  "hvib_data_im_suffix": "_im"
41                )
42  S, St, Hvib_ks = step3.get_step2_data(params)
43
44  S0 =[ 6,-6, 7,-7, 8,-8];
45  S1 =[ 6,-6, 7,-7, 9,-8];
46  S2 =[ 6,-6, 7,-7,10,-8];
47  S3 =[ 6,-6, 7,-7,11,-8];
48  S4 =[ 6,-6, 7,-7,12,-8];
49  S5 =[ 6,-6, 7,-7,13,-8];
50  S6 =[ 6,-6, 7,-7,14,-8];
51  S7 =[ 6,-6, 7,-7,15,-8];
52  S8 =[ 6,-6, 9,-7, 8,-8];
53  S9 =[ 9,-6, 7,-7, 8,-8];
54  S10=[ 6,-6,10,-7, 8,-8];
55
56  basis = [ S0, S1, S2, S3 ]
```

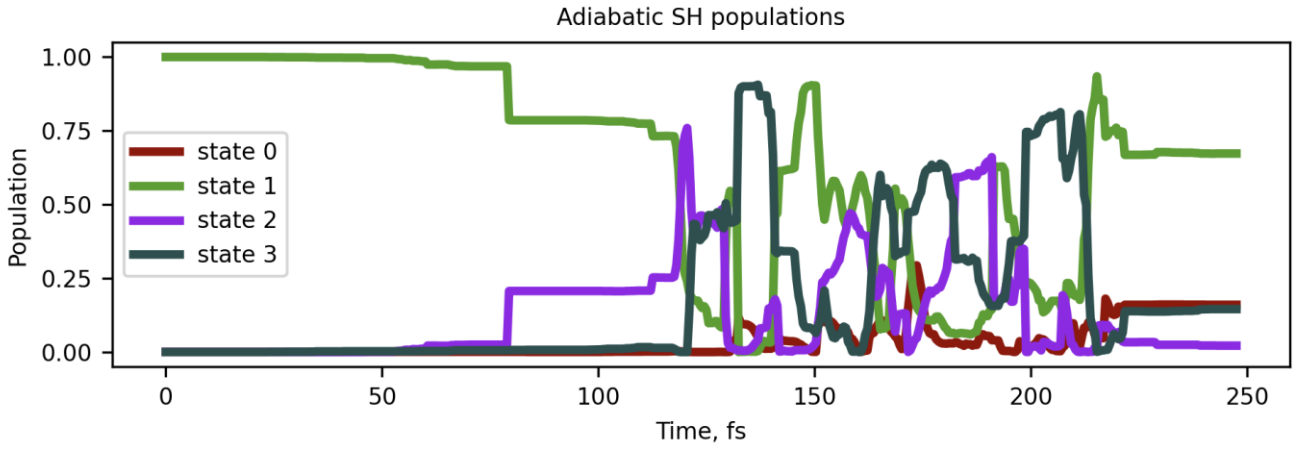
Step 4: NAMD

```
22  colors.update({"12": "#FF4500"})  # orangered
23  colors.update({"13": "#B22222"})  # firebrick
24  colors.update({"14": "#DC143C"})  # crimson
25  colors.update({"21": "#5e9c36"})  # green
26  colors.update({"22": "#006400"})  # darkgreen
27  colors.update({"23": "#228B22"})  # forestgreen
28  colors.update({"24": "#808000"})  # olive
29  colors.update({"31": "#8A2BE2"})  # blueviolet
30  colors.update({"32": "#00008B"})  # darkblue
31  colors.update({"41": "#2F4F4F"})  # darkslategray
32
33  clrs_index = ["11", "21", "31", "41", "12", "22", "32", "13","23", "14", "24"]
34
35  istate=int(sys.argv[1])
36  istep =int(sys.argv[2])    # the first timestep to read
37  fstep = min(istep+500,3996)   # the last timestep to read
38
39  pref = F"S{istate}_{istep}-{fstep}"
40
41  dt = 0.5*41.0   # integration time-step [a.u. of time]
42
43  print(pref)
44
45  nsteps = fstep - istep
46  NSTEPS = nsteps
47  print(F"Number of steps = {nsteps}")
48
49  x = np.loadtxt(F'ham_sd/hvib_sd_0_im')
50  nstates = x.shape[0]
51  NSTATES = nstates
52  print(F"Number of states = {nstates}")
53
54  #================= Read energies ====================
55  Hvib, NAC, HAM = [], [], []
56  for step in range(istep,fstep):
```

Results

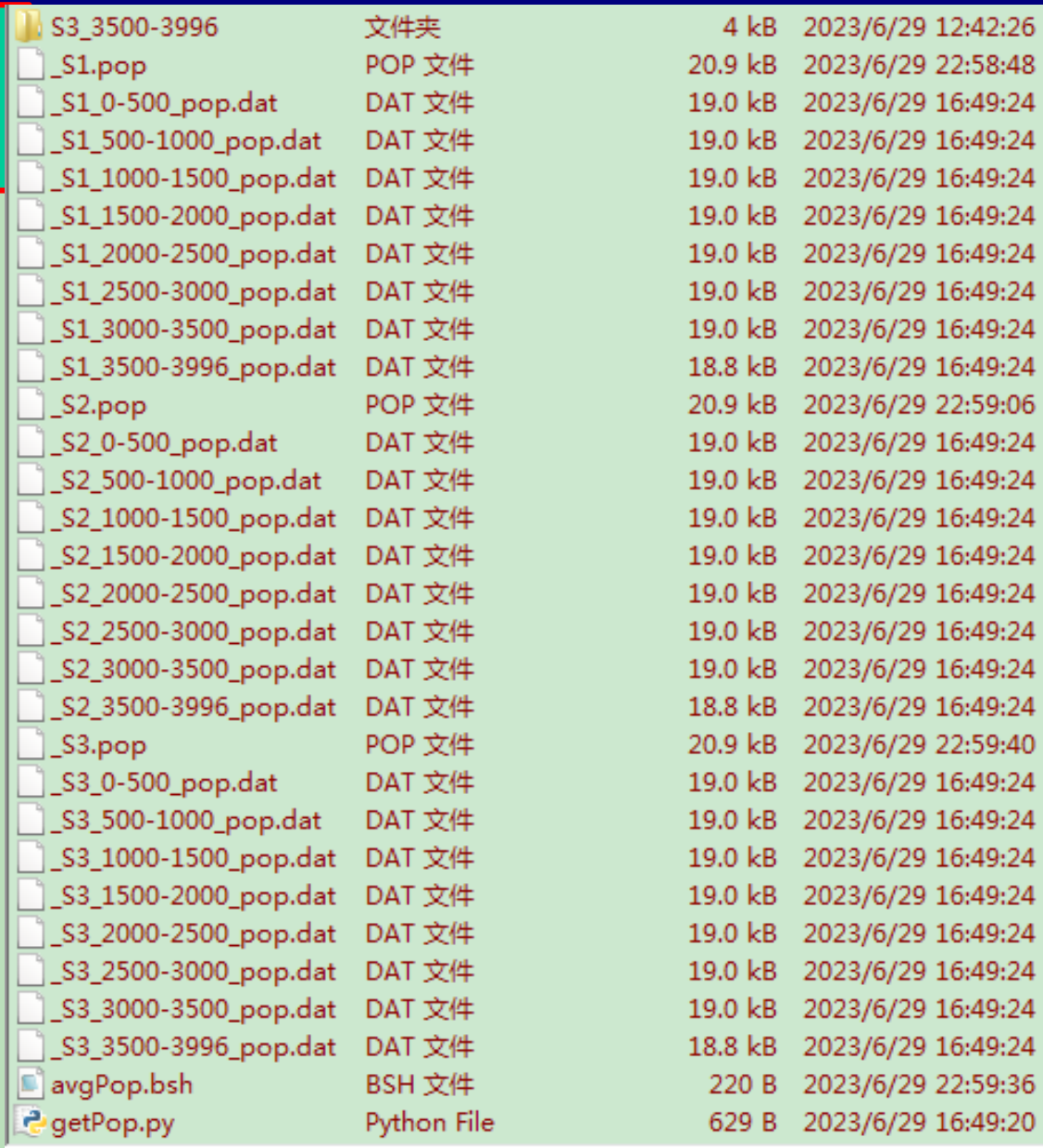
S1_0-500	文件夹	4 kB	2023/6/29 16:40:02
S1_500-1000	文件夹		
S1_1000-1500	文件夹		
S1_1500-2000	文件夹		
S1_2000-2500	文件夹		
S1_2500-3000	文件夹		
S1_3000-3500	文件夹		
S1_3500-3996	文件夹	4 kB	2023/6/29 11:13:08
S2_0-500	文件夹		
S2_500-1000	文件夹		
S2_1000-1500	文件夹		
S2_1500-2000	文件夹		
S2_2000-2500	文件夹		
S2_2500-3000	文件夹		
S2_3000-3500	文件夹		
S2_3500-3996	文件夹		
S3_0-500	文件夹		
S3_500-1000	文件夹	4 kB	2023/6/29 12:30:44
S3_1000-1500	文件夹	4 kB	2023/6/29 12:32:50
S3_1500-2000	文件夹	4 kB	2023/6/29 12:34:46
S3_2000-2500	文件夹	4 kB	2023/6/29 12:36:44
S3_2500-3000	文件夹	4 kB	2023/6/29 12:38:40
S3_3000-3500	文件夹	4 kB	2023/6/29 12:40:34
S3_3500-3996	文件夹	4 kB	2023/6/29 12:42:26

_dyn_params.txt	文本文档	1.7 kB	2023/6/29 11:05:02
_model_params.txt	文本文档	54 B	2023/6/29 11:05:02
mem_data.hdf	HDF 文件	45.4 kB	2023/6/29 11:05:46
se_pop_adi.png	IrfanView PNG ...	235.8 kB	2023/6/29 11:05:48
sh_pop_adi.png	IrfanView PNG ...	135.2 kB	2023/6/29 11:05:48



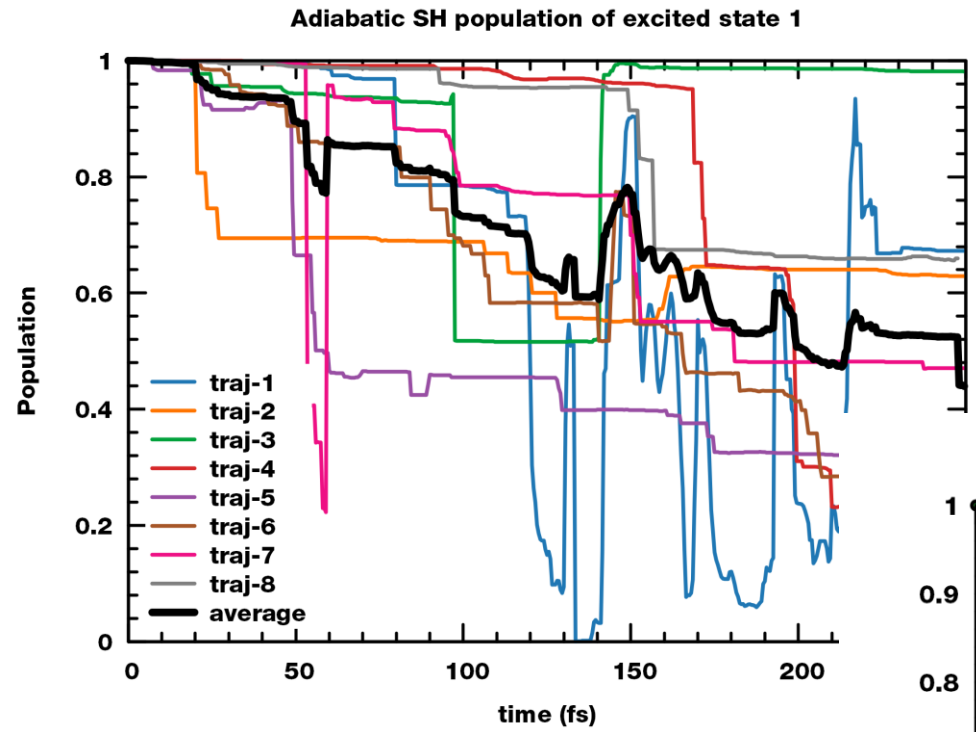
data

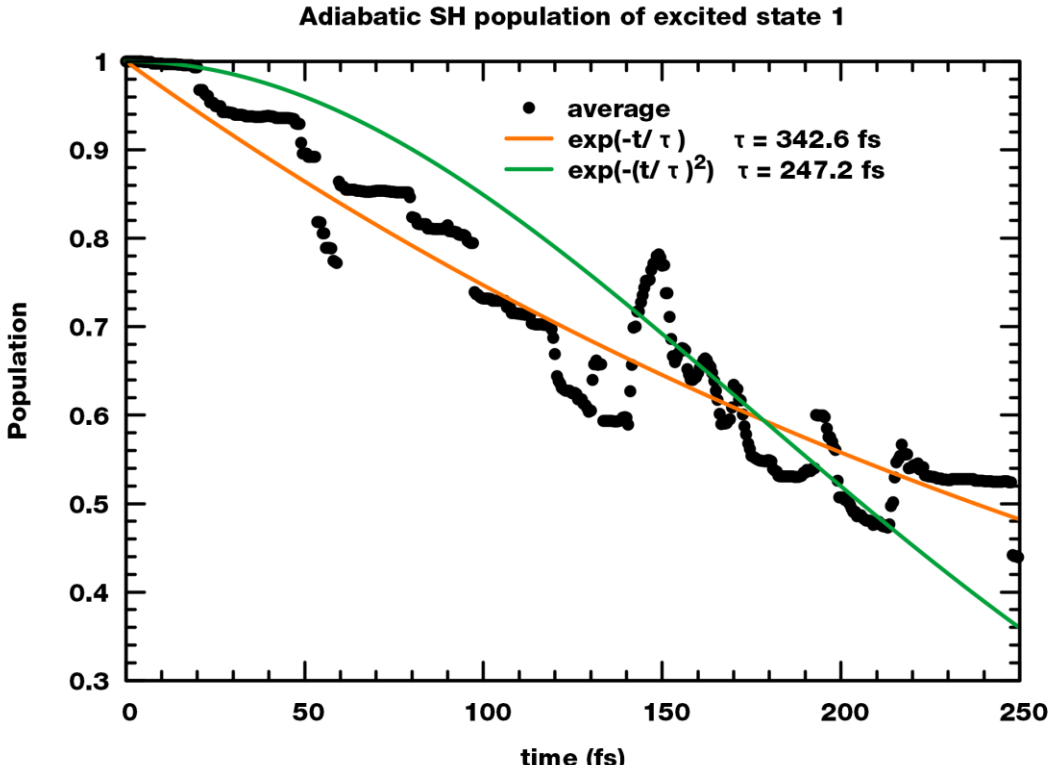
```
> python getPop.py
> bash   avgPop.bsh
```



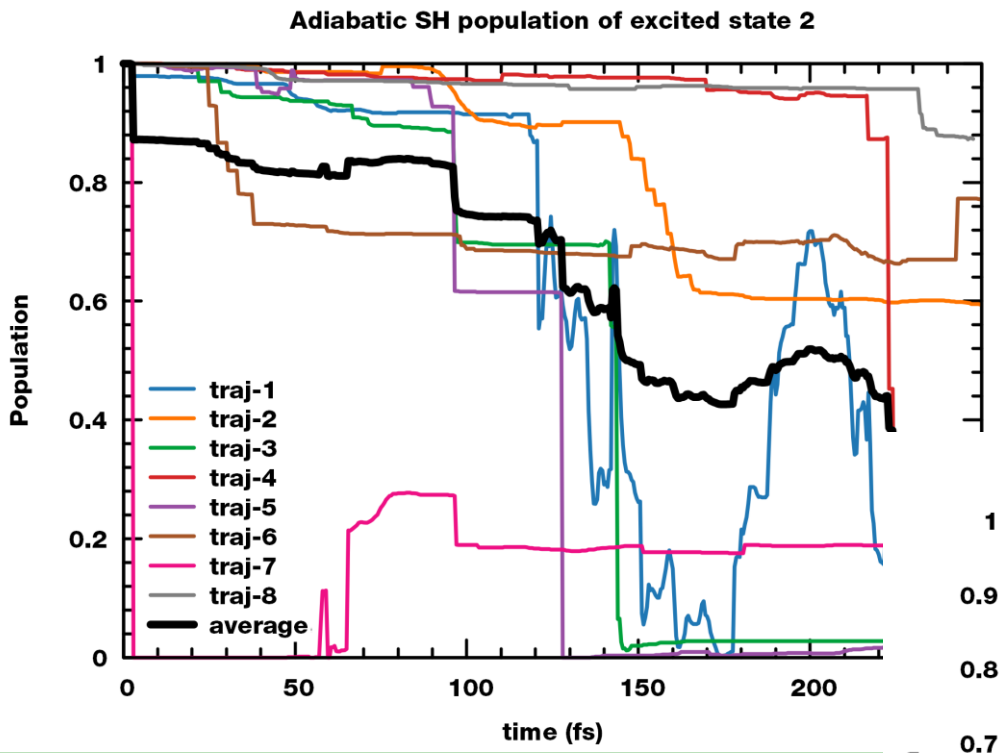
S3_3500-3996	文件夹	4 kB	2023/6/29 12:42:26
_S1.pop	POP 文件	20.9 kB	2023/6/29 22:58:48
_S1_0-500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S1_500-1000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S1_1000-1500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S1_1500-2000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S1_2000-2500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S1_2500-3000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S1_3000-3500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S1_3500-3996_pop.dat	DAT 文件	18.8 kB	2023/6/29 16:49:24
_S2.pop	POP 文件	20.9 kB	2023/6/29 22:59:06
_S2_0-500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S2_500-1000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S2_1000-1500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S2_1500-2000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S2_2000-2500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S2_2500-3000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S2_3000-3500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S2_3500-3996_pop.dat	DAT 文件	18.8 kB	2023/6/29 16:49:24
_S3.pop	POP 文件	20.9 kB	2023/6/29 22:59:40
_S3_0-500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S3_500-1000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S3_1000-1500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S3_1500-2000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S3_2000-2500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S3_2500-3000_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S3_3000-3500_pop.dat	DAT 文件	19.0 kB	2023/6/29 16:49:24
_S3_3500-3996_pop.dat	DAT 文件	18.8 kB	2023/6/29 16:49:24
avgPop.bsh	BSH 文件	220 B	2023/6/29 22:59:36
getPop.py	Python File	629 B	2023/6/29 16:49:20

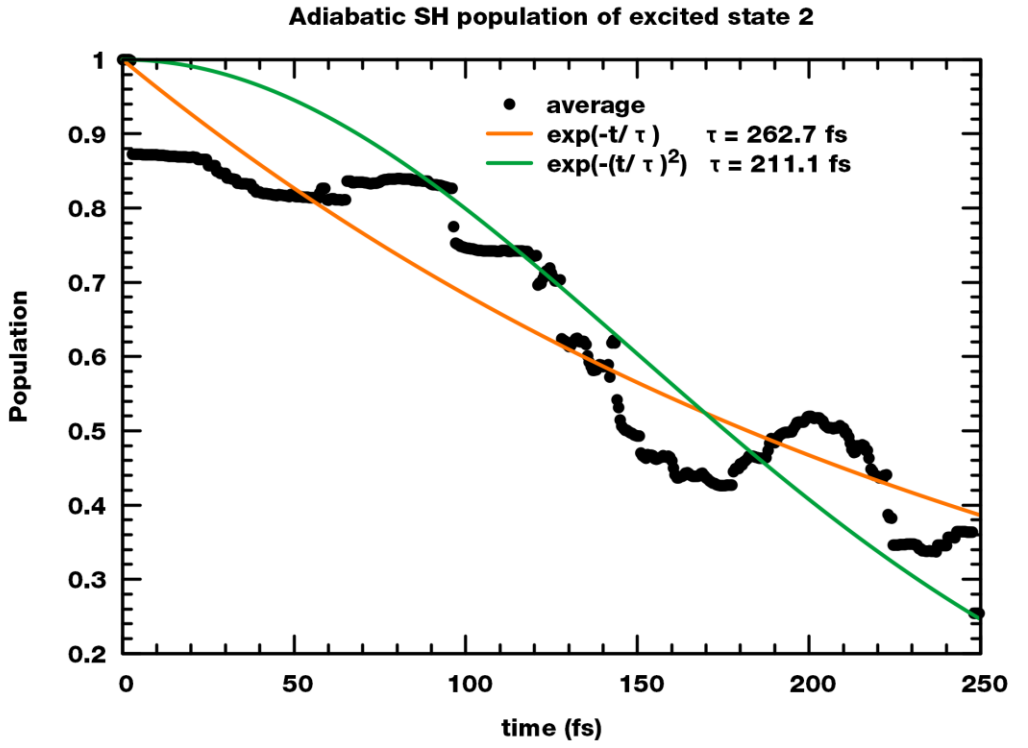
Decay of S1


Adiabatic SH population of excited state 1
Population
time (fs)
traj-1
traj-2
traj-3
traj-4
traj-5
traj-6
traj-7
traj-8
average

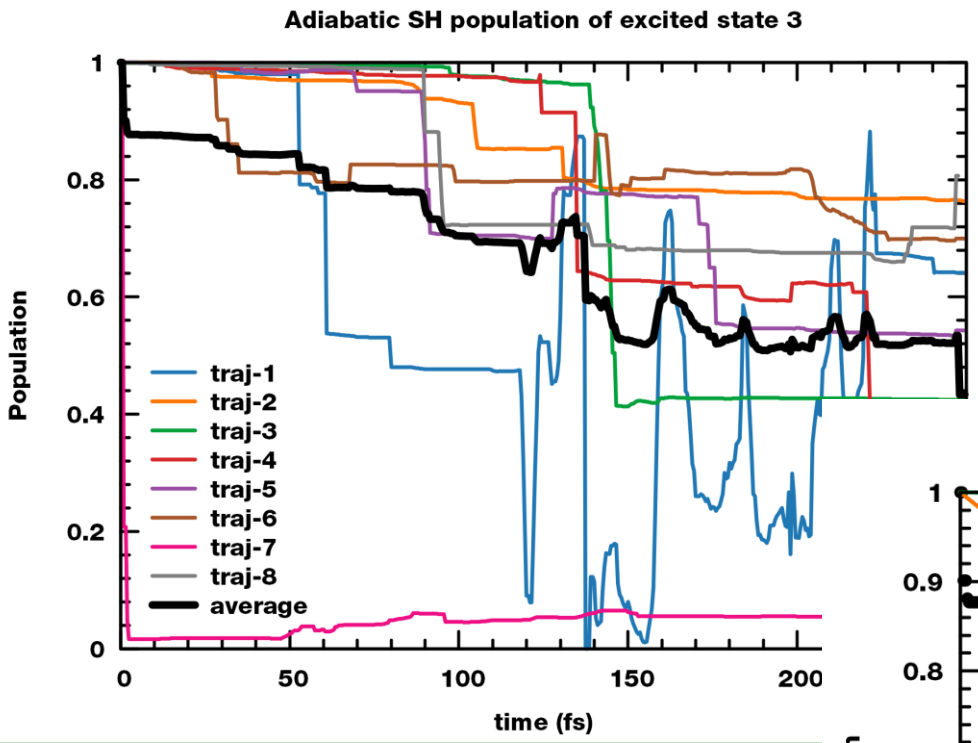

Adiabatic SH population of excited state 1
Population
time (fs)
average
exp(-t/ τ) τ = 342.6 fs
exp(-(t/ τ)2) τ = 247.2 fs

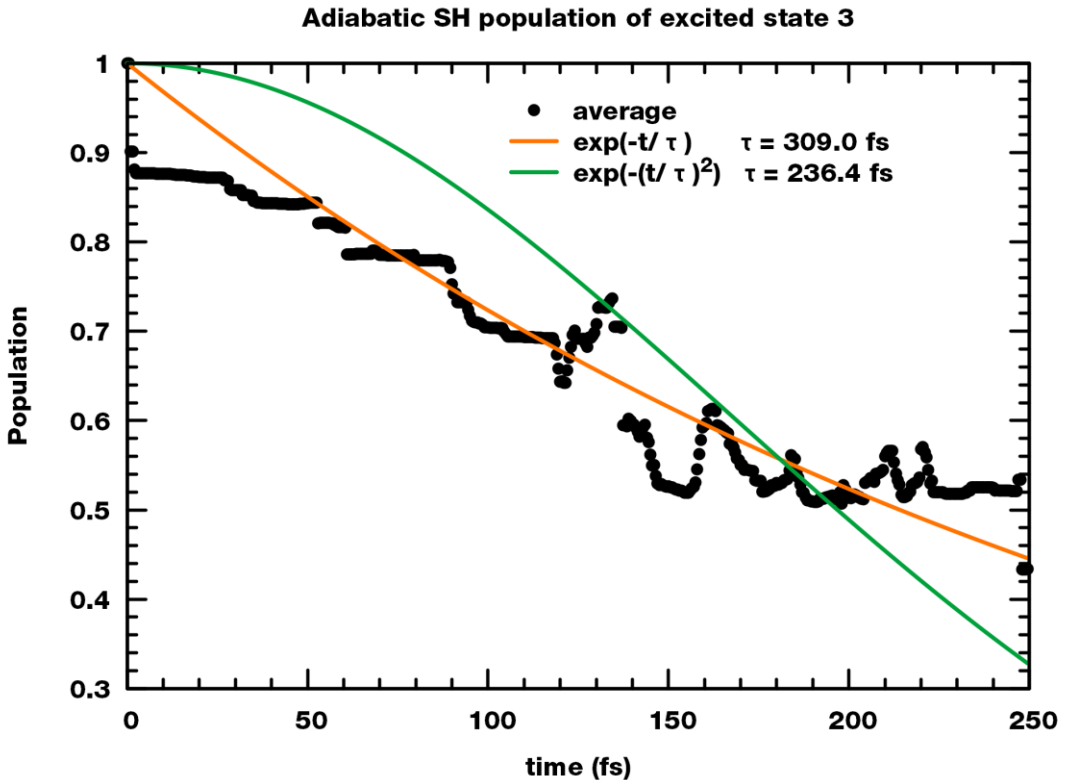
Decay of S2


Adiabatic SH population of excited state 2
Population
time (fs)
traj-1
traj-2
traj-3
traj-4
traj-5
traj-6
traj-7
traj-8
average


Adiabatic SH population of excited state 2
Population
time (fs)
average
exp(-t/ τ) τ = 262.7 fs
exp(-(t/ τ)2) τ = 211.1 fs

Decay of S3


Adiabatic SH population of excited state 3
Population
time (fs)
traj-1
traj-2
traj-3
traj-4
traj-5
traj-6
traj-7
traj-8
average


Adiabatic SH population of excited state 3
Population
time (fs)
average
exp(-t/ τ) τ = 309.0 fs
exp(-(t/ τ)2) τ = 236.4 fs

Remarks

- DFTB may be not a good choice for Ag cluster, or the parameter set of Ag is not good enough
- The decay behavior of the first 3 excited states is described better with exponential kinetics
- The relaxation time is more than 250 fs, which means the simulation should be long enough
- More trajectories are necessary to converge the decay curve
- Other SH method should be used to validate the FSSH results

Thank You