

CyberTraining project: Implementing a numerical efficient formulation of FSSH in Libra

Agenda

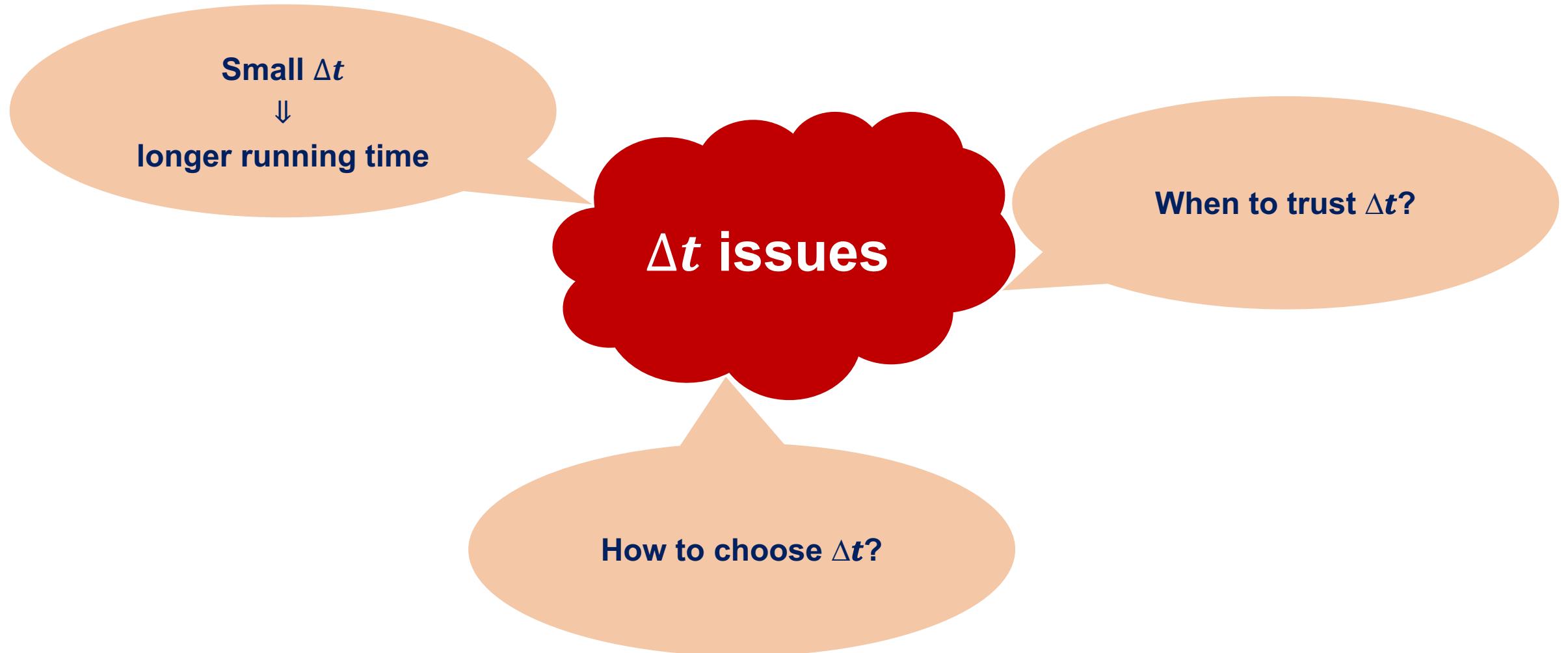
Motivation & Goal

Formulation of a new hop probability

Numerical experiment

Time & workflow overview

Motivation & Goal



Motivation: FSSH procedure from time t to $t + \Delta t$

1) Propagate classical coordinates from t to $t + \Delta t$ at active state m:

$$\begin{aligned}\dot{x} &= p \\ \dot{p} &= F_m^{adi}(x)\end{aligned}$$

2) Propagate TDSE of the electronic amplitudes from t to $t + \Delta t$:

$$i\epsilon \dot{c}^{adi}(t) = H^{adi}(t)c^{adi}(t)$$

with $H_{m,n}^{adi}(t) = \delta_{m,n}E_m(x) - i\epsilon d^{m,n}(x)^T p$

3) Hop to state n with probability:

$$P_{m,n}^{FSSH}(t, t + \Delta t) = \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

with $\rho_{n,m}^{adi} = c_n^{adi} \overline{c_m^{adi}}$

Issue: energy conservation is not enough as indicator

1) Propagate classical coordinates from t to $t + \Delta t$ at active state m:

$$\begin{aligned}\dot{x} &= p \\ \dot{p} &= F_m^{adi}(x)\end{aligned}$$

2) Propagate TDSE of the electronic amplitudes from t to $t + \Delta t$:

$$i\epsilon \dot{c}^{adi}(t) = H^{adi}(t)c^{adi}(t)$$

with $H_{m,n}^{adi}(t) = \delta_{m,n}E_m(x) - i\epsilon d^{m,n}(x)^T p$

3) Hop to state n with probability:

$$P_{m,n}^{FSSH}(t, t + \Delta t) = \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

with $\rho_{n,m}^{adi} = c_n^{adi} \overline{c_m^{adi}}$

**Energy conservation
only indicates how well
step 1) is propagated
but is no indicator of the
convergence of the
other steps.**

Issue: typical time step size hierarchy

1) Propagate classical coordinates from t to $t + \Delta t$ at active state m:

$$\begin{aligned}\dot{x} &= p \\ \dot{p} &= F_m^{adi}(x)\end{aligned}$$

2) Propagate TDSE of the electronic amplitudes from t to $t + \Delta t$:

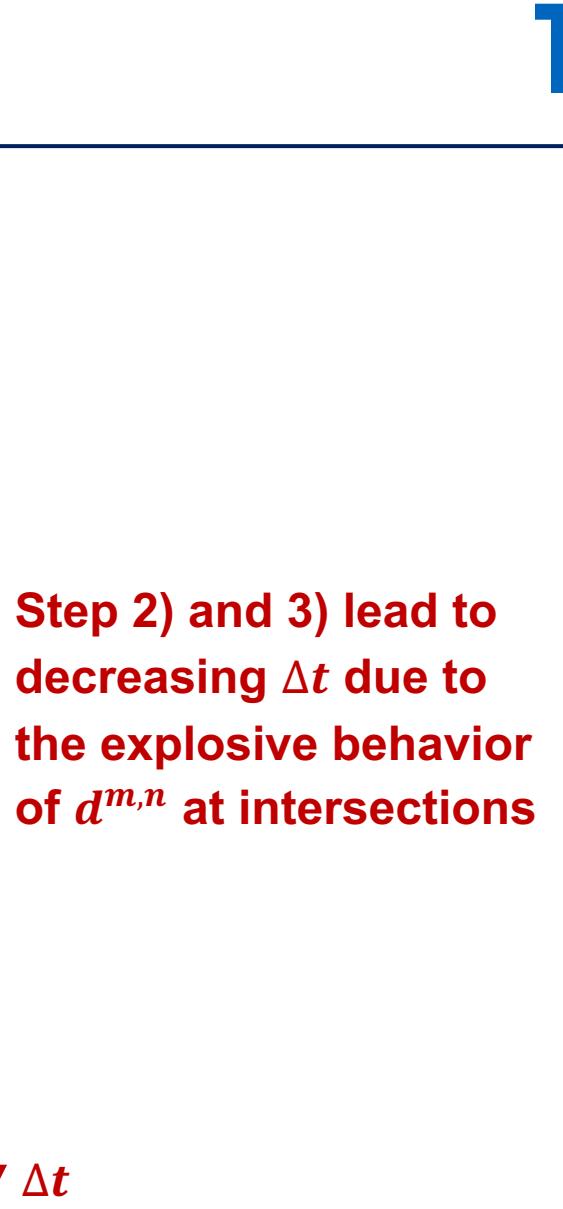
$$i\epsilon \dot{c}^{adi}(t) = H^{adi}(t)c^{adi}(t)$$

with $H_{m,n}^{adi}(t) = \delta_{m,n}E_m(x) - i\epsilon d^{m,n}(x)^T p$

3) Hop to state n with probability:

$$P_{m,n}^{FSSH}(t, t + \Delta t) = \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

with $\rho_{n,m}^{adi} = c_n^{adi} \overline{c_m^{adi}}$



Find numerical efficient formulations and methods of step 2) and 3) such that:



FSSH boosts up and converges faster



Step 2) and 3) become less sensitive then step 1)



Energy conservation can be used as a convergence indicator for FSSH

Goal of this particular presentation

1) Propagate classical coordinates from t to $t + \Delta t$ at active state m:

$$\begin{aligned}\dot{x} &= p \\ \dot{p} &= F_m^{adi}(x)\end{aligned}$$

2) Propagate TDSE of the electronic amplitudes from t to $t + \Delta t$:

$$i\epsilon \dot{c}^{adi}(t) = H^{adi}(t)c^{adi}(t)$$

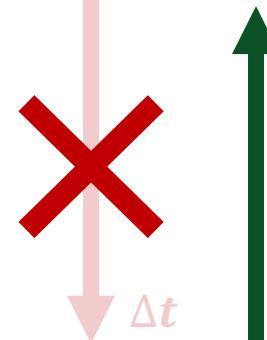
with $H_{m,n}^{adi}(t) = \delta_{m,n}E_m(x) - i\epsilon d^{m,n}(x)^T p$

Find a numerical efficient formulation of $P_{m,n}^{FSSH}$

3) Hop to state n with probability:

$$P_{m,n}^{FSSH}(t, t + \Delta t) = \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

with $\rho_{n,m}^{adi} = c_n^{adi} \overline{c_m^{adi}}$



Formulation of a new hop probability

Derivation of the hop probability

$$P_{m,out}(t, t + \Delta t) = \frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)}$$

Derivation of the hop probability: FSSH (1/3)

FSSH

$$P_{m,out}(t, t + \Delta t) = \frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} \stackrel{FEM}{\approx} \frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)}$$

Derivation of the hop probability: FSSH (2/3)

FSSH

$$P_{m,out}(t, t + \Delta t) = \frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} \underset{\text{FEM}}{\approx} \frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)} = \sum_{n \neq m} \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

Derivation of the hop probability: FSSH (3/3)

FSSH

$$P_{m,out}(t, t + \Delta t) = \frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} \underset{\text{FEM}}{\approx} \frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)} = \sum_{n \neq m} \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

$$P_{m,out} \stackrel{!}{=} \sum_{n \neq m} P_{m,n}$$

$$P_{m,n}^{FSSH}(t, t + \Delta t) := \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

Derivation of the hop probability: FSSH2 (1/3)

FSSH

$$P_{m,out}(t, t + \Delta t) = \frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} \underset{\text{FEM}}{\approx} \frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)} = \sum_{n \neq m} \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

$$P_{m,out} \stackrel{!}{=} \sum_{n \neq m} P_{m,n}$$

$$P_{m,n}^{FSSH}(t, t + \Delta t) := \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

FSSH2

$$\sum_n \rho_{n,n}^{adi}(t) = 1 \Leftrightarrow \rho_{m,m}^{adi}(t) = 1 - \sum_{n \neq m} \rho_{n,n}^{adi}(t)$$

Derivation of the hop probability: FSSH2 (2/3)

FSSH

$$P_{m,out}(t, t + \Delta t) = \frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} \underset{\text{FEM}}{\approx} \frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)} = \sum_{n \neq m} \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

$$P_{m,out} \stackrel{!}{=} \sum_{n \neq m} P_{m,n}$$

$$P_{m,n}^{FSSH}(t, t + \Delta t) := \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

FSSH2

$$\sum_n \rho_{n,n}^{adi}(t) = 1 \Leftrightarrow \rho_{m,m}^{adi}(t) = 1 - \sum_{n \neq m} \rho_{n,n}^{adi}(t)$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} = \sum_{n \neq m} \frac{\rho_{n,n}^{adi}(t + \Delta t) - \rho_{n,n}^{adi}(t)}{\rho_{m,m}^{adi}(t)}$$

Derivation of the hop probability: FSSH2 (3/3)

FSSH

$$P_{m,out}(t, t + \Delta t) = \frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} \underset{\text{FEM}}{\approx} \frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)}$$

$$\frac{\dot{\rho}_{m,m}^{adi}(t) \Delta t}{\rho_{m,m}^{adi}(t)} = \sum_{n \neq m} \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

$$P_{m,out} \stackrel{!}{=} \sum_{n \neq m} P_{m,n}$$

$$P_{m,n}^{FSSH}(t, t + \Delta t) := \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

FSSH2

$$\sum_n \rho_{n,n}^{adi}(t) = 1 \Leftrightarrow \rho_{m,m}^{adi}(t) = 1 - \sum_{n \neq m} \rho_{n,n}^{adi}(t)$$

$$\frac{\rho_{m,m}^{adi}(t) - \rho_{m,m}^{adi}(t + \Delta t)}{\rho_{m,m}^{adi}(t)} = \sum_{n \neq m} \frac{\rho_{n,n}^{adi}(t + \Delta t) - \rho_{n,n}^{adi}(t)}{\rho_{m,m}^{adi}(t)}$$

$$P_{m,out} \stackrel{!}{=} \sum_{n \neq m} P_{m,n}$$

$$P_{m,n}^{FSSH2}(t, t + \Delta t) := \frac{\rho_{n,n}^{adi}(t + \Delta t) - \rho_{n,n}^{adi}(t)}{\rho_{m,m}^{adi}(t)}$$

Theoretical advantages of FSSH2 over FSSH

$$P_{m,n}^{FSSH}(t, t + \Delta t) := \frac{2\Delta t \operatorname{Re}(\rho_{n,m}^{adi}) d^{m,n}(x)^T p}{\rho_{m,m}^{adi}}$$

vs.

$$P_{m,n}^{FSSH2}(t, t + \Delta t) := \frac{\rho_{n,n}^{adi}(t + \Delta t) - \rho_{n,n}^{adi}(t)}{\rho_{m,m}^{adi}(t)}$$



Numerical stable probability formula



For a total of only two electronic states, we have $P_{m,n}^{FSSH2} = P_{m,out}^{\text{FEM}} \approx P_{m,n}^{FSSH}$



FSSH2 can handle trivial crossings (depending on the TDSE propagator)

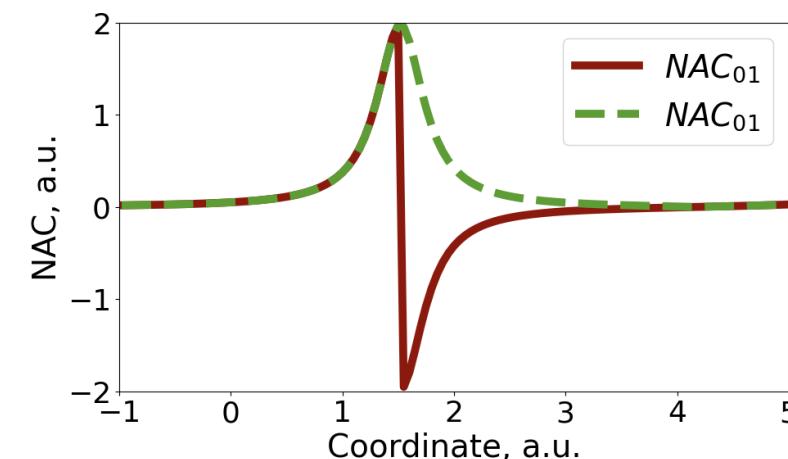
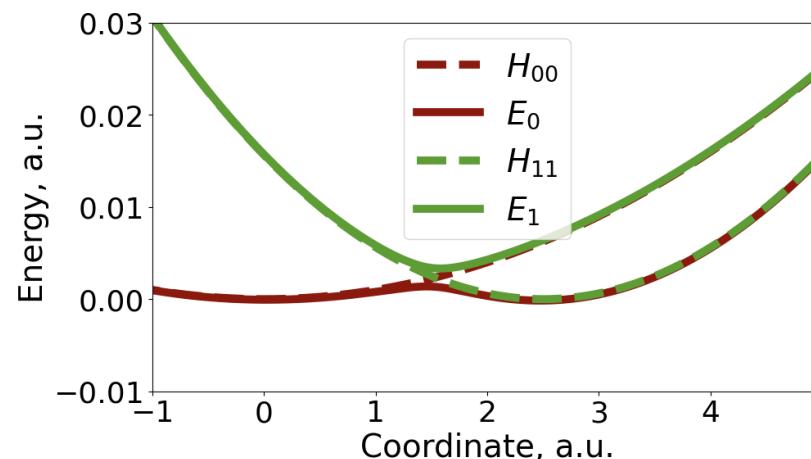
Numerical experiment

The Holstein model

For our simulations, we choose the one-dimensional Holstein model given by the following diabatic potential energy matrix for two electronic states,

$$V^{dia}(x) = \begin{pmatrix} U_1 + \frac{k_1}{2}(x - y_1)^2 & V \\ V & U_2 + \frac{k_2}{2}(x - y_2)^2 \end{pmatrix}.$$

We set: $U = \begin{pmatrix} 0 \\ -0.001 \end{pmatrix}$, $y = \begin{pmatrix} 0 \\ 0.5 \end{pmatrix}$, $k = \begin{pmatrix} 0.002 \\ 0.008 \end{pmatrix}$, and $V = 0.001$.



Initial wavefunction

Gaussian wave packet centered at:

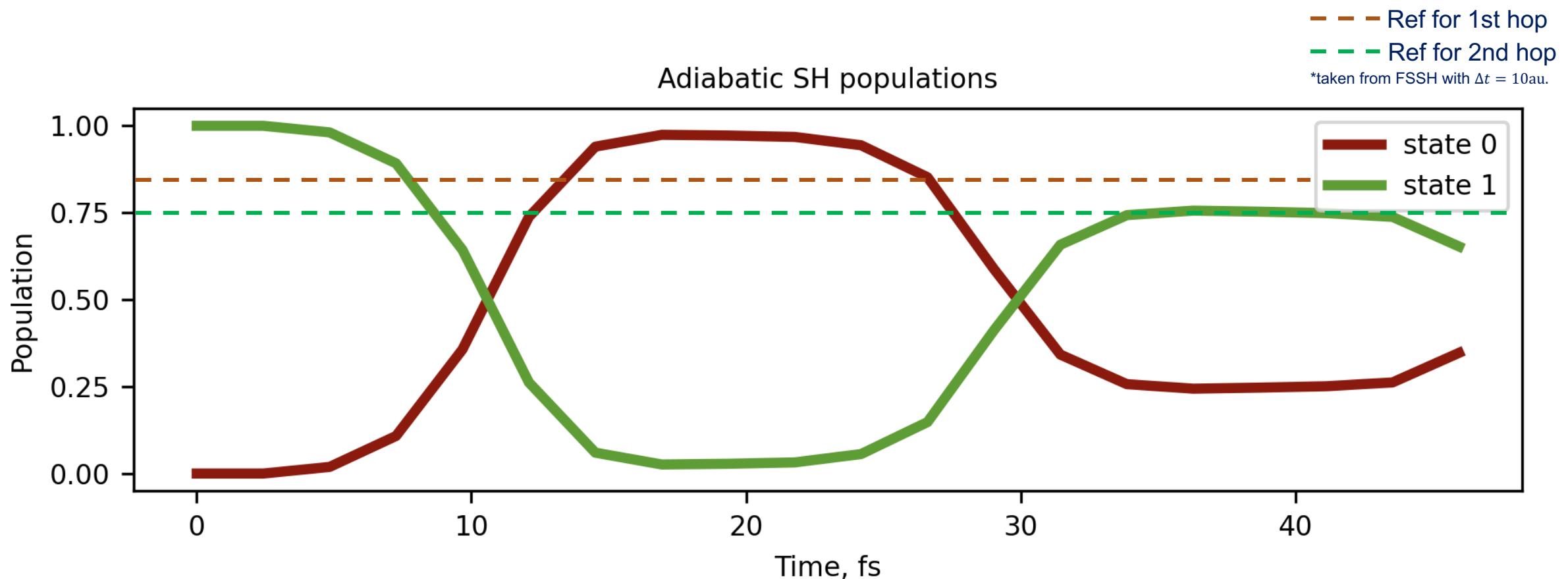
➤ $\begin{pmatrix} q \\ p \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix}$

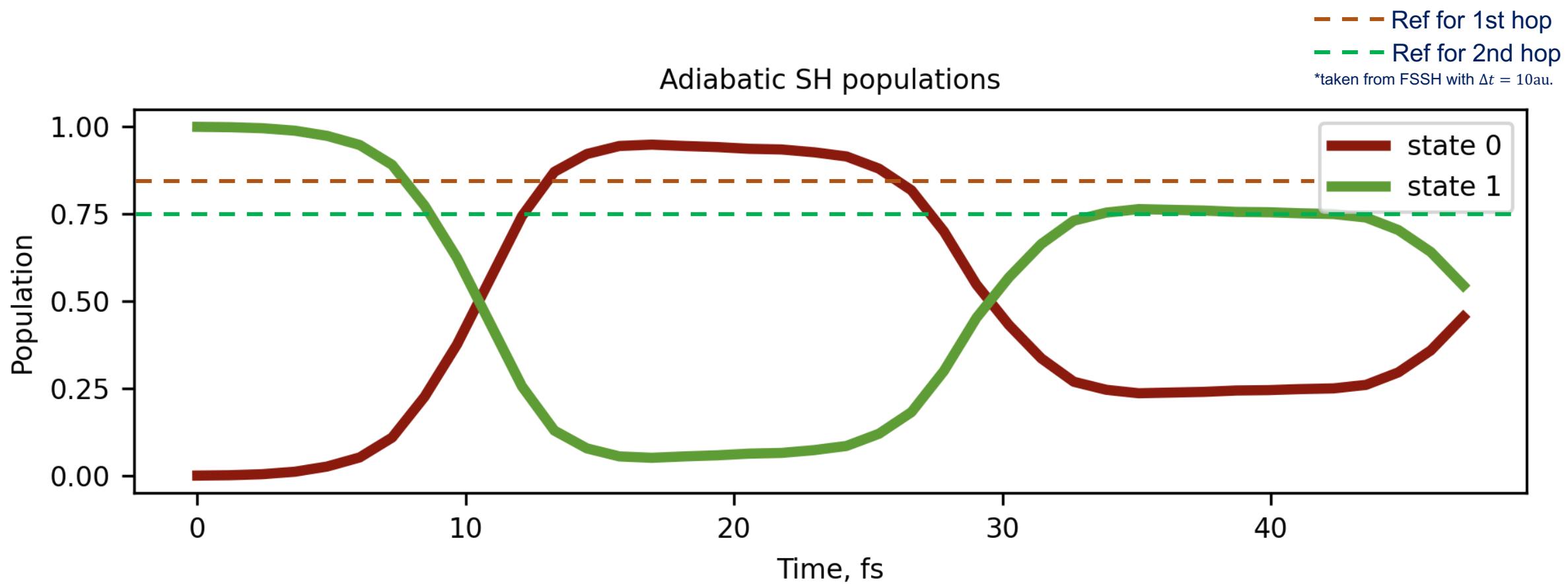
➤ width matrix $\Sigma = \begin{pmatrix} \sqrt{\frac{1}{2\sqrt{0.01 \cdot M}}} & 0 \\ 0 & \sqrt{\frac{\sqrt{0.01 \cdot M}}{2}} \end{pmatrix}$

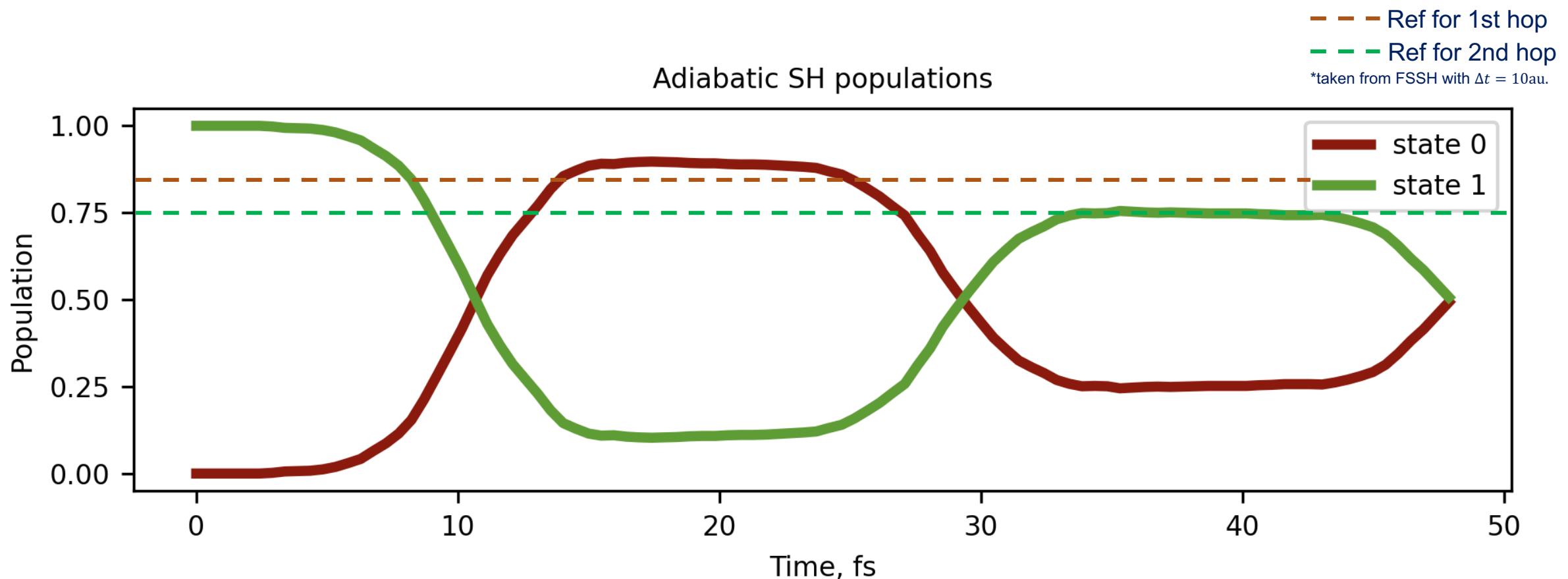
➤ located at the second diabatic state

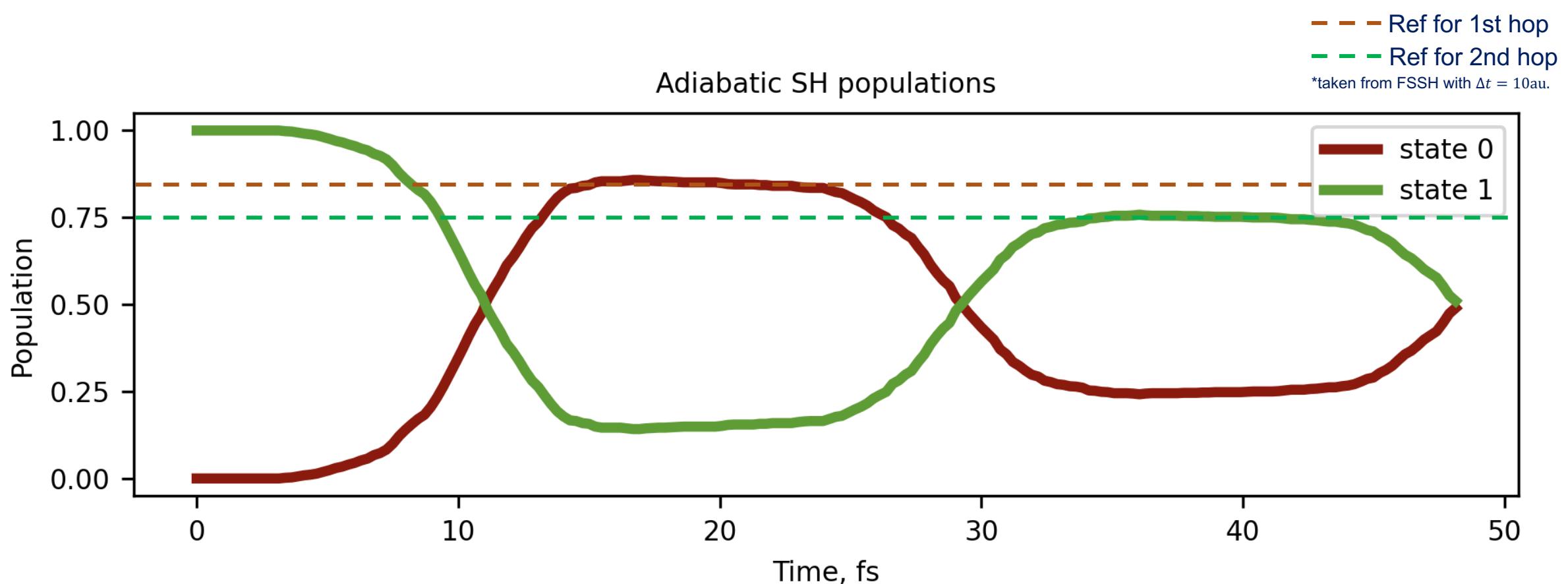
Further parameters

- **Mass M:** 2000au.
- **Simulation time:** 2000au. / ca. 50fs
- **Time step sizes Δt in au:** 100, 50, 20, 10
- **Methods:** FSSH, FSSH2, and GFSH (for Alexey)
- **Number of trajectories:** 1000 (5000 would be better to observe fewer deviations – but longer running time)



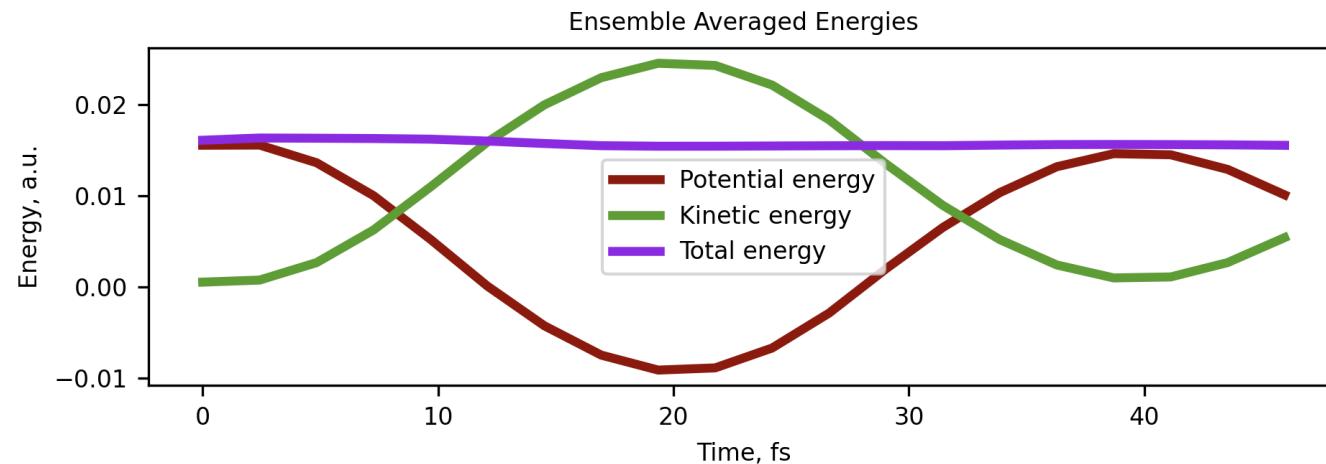




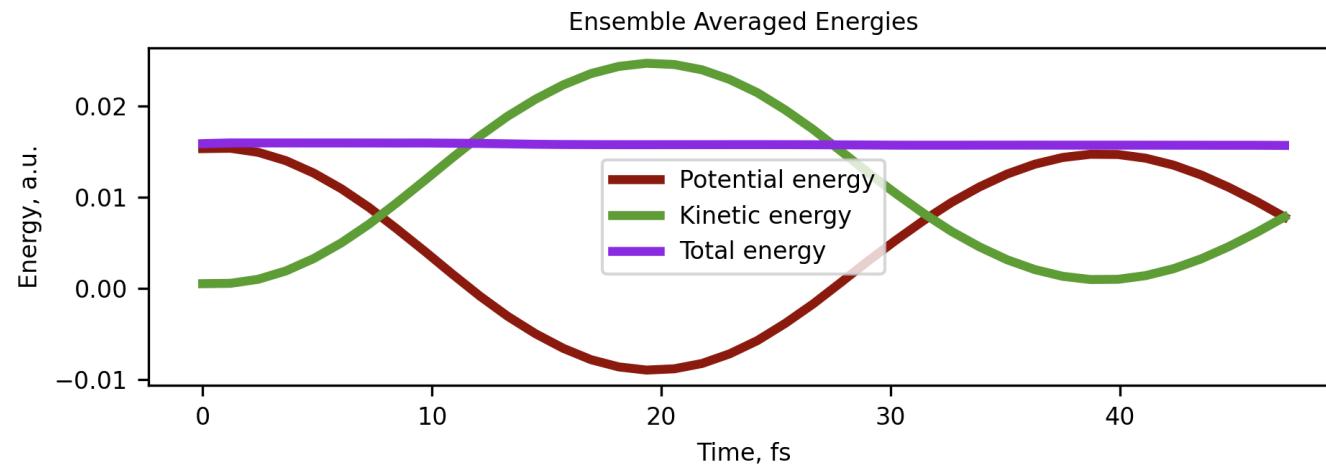


FSSH – energy conservation

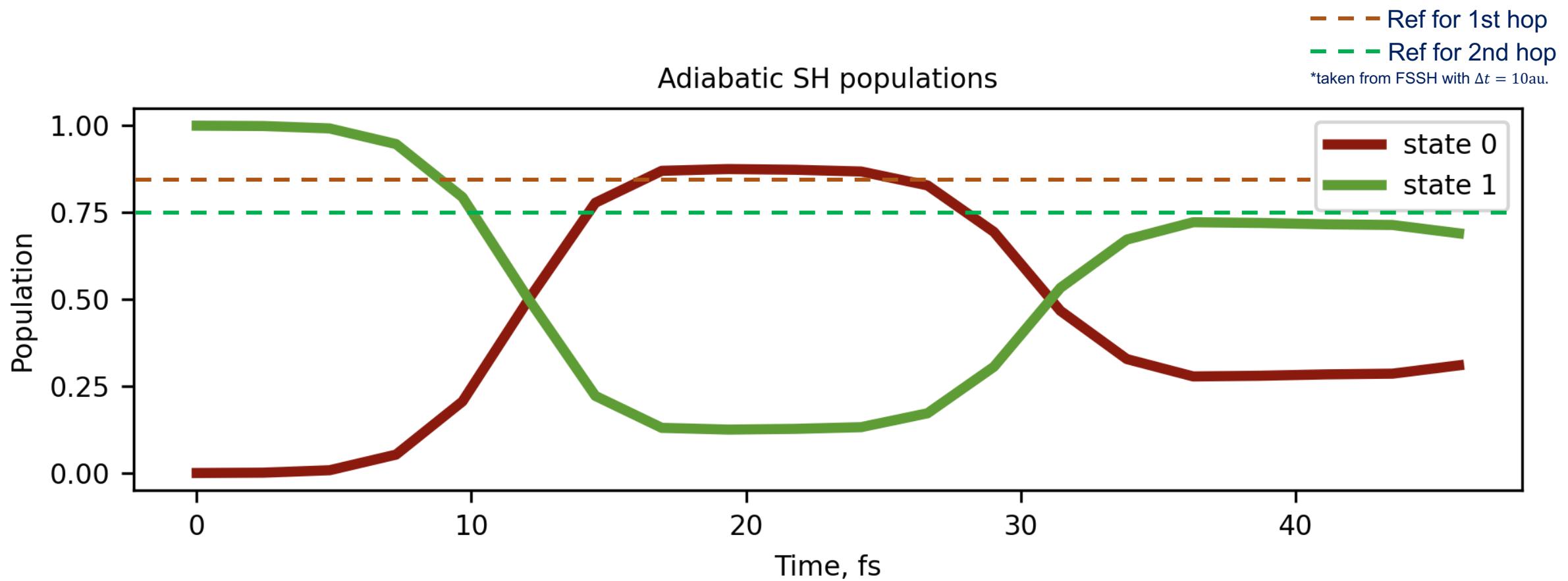
$\Delta t = 100$:

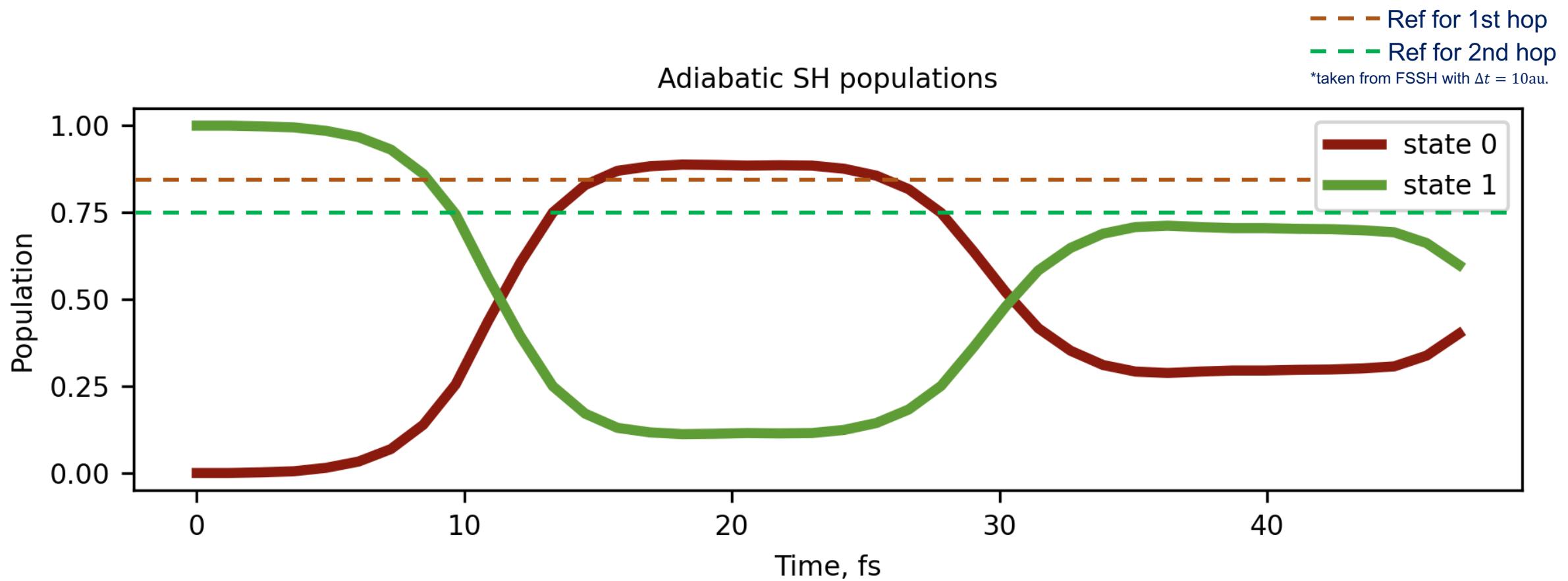


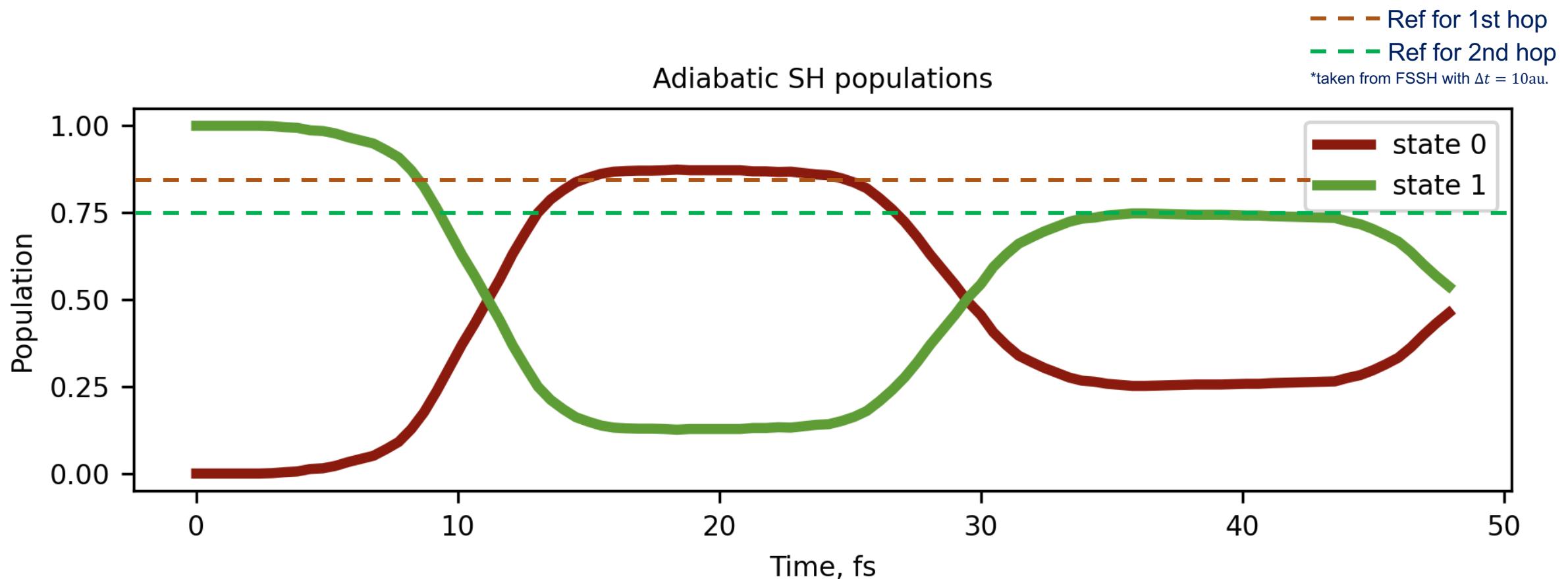
$\Delta t = 50$:

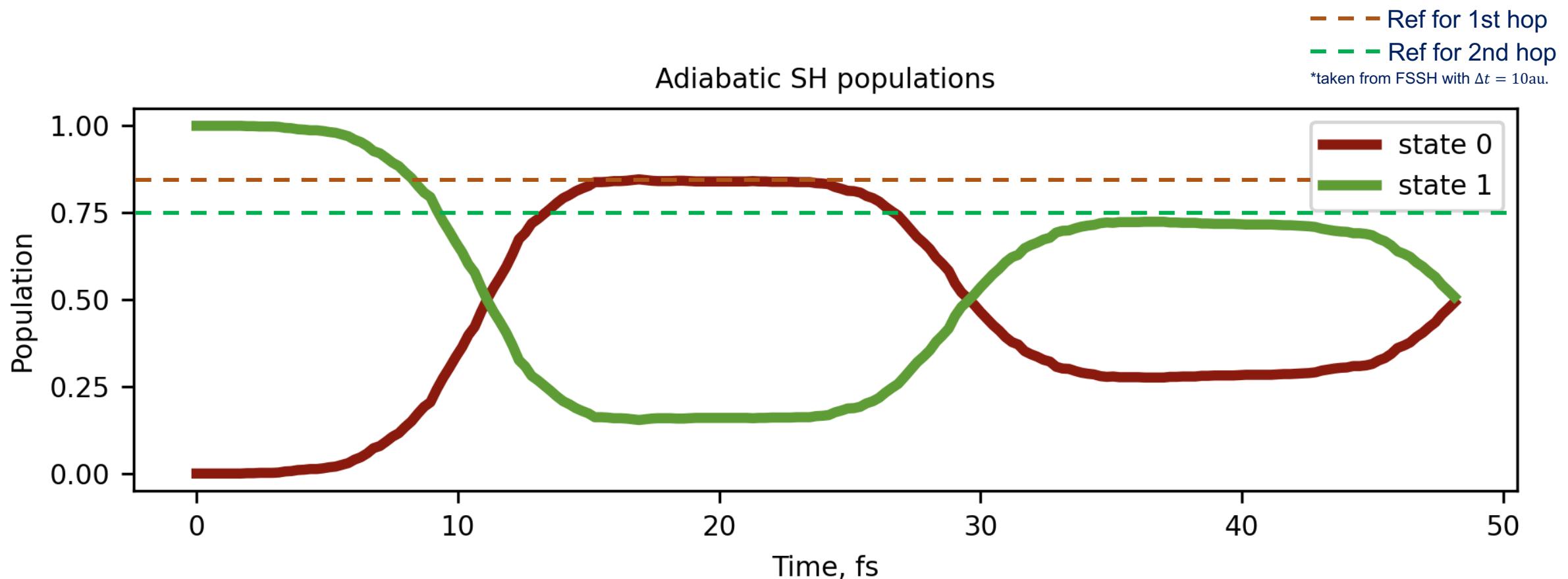


- Energy was already conserved at $\Delta t = 100\text{au}$.
- However, FSSH started only to converge at around $\Delta t = 10$ to 20au .



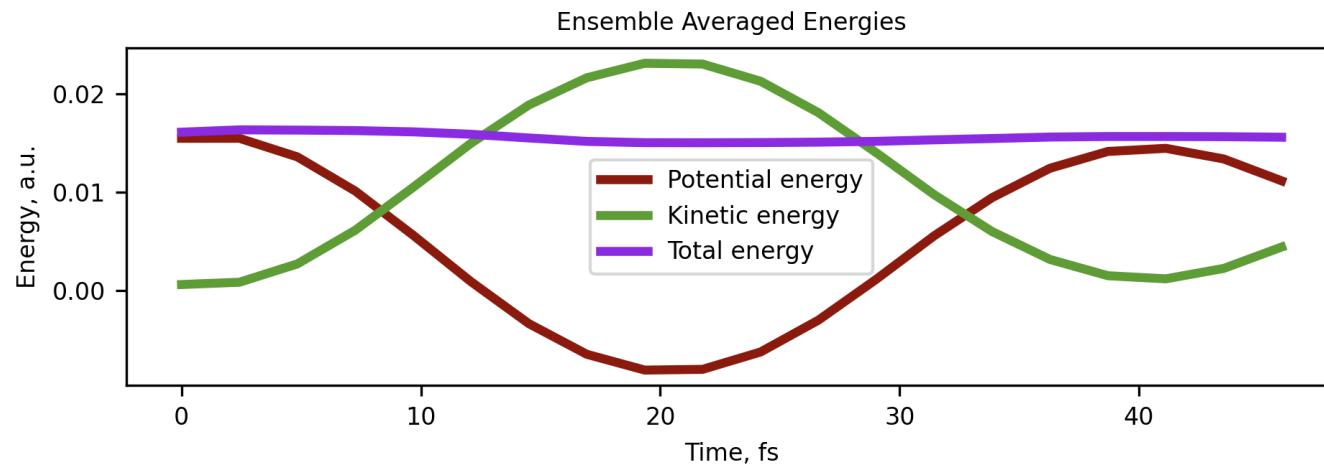




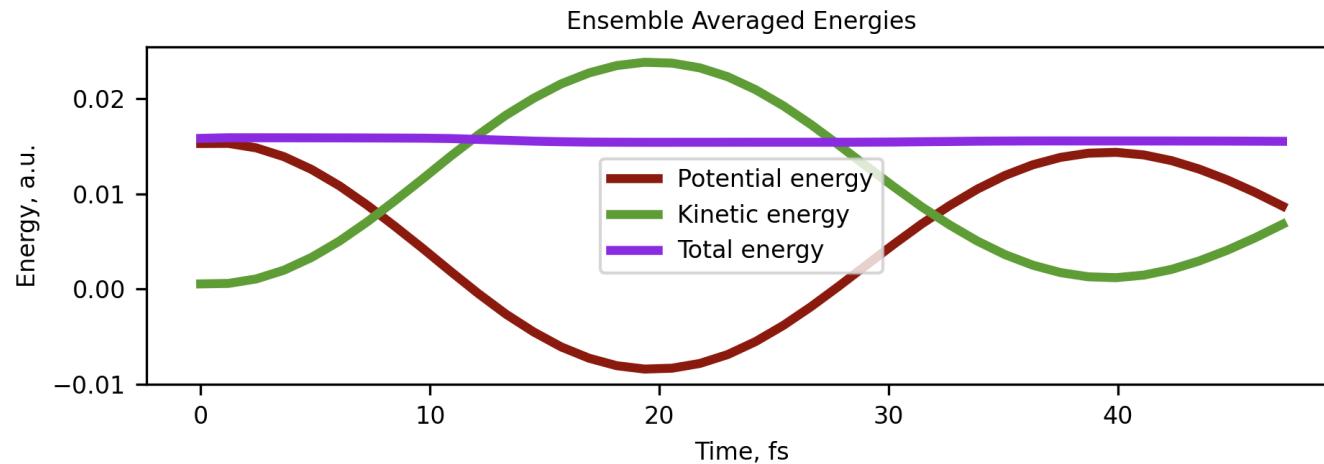


FSSH2 – energy conservation

$\Delta t = 100$:

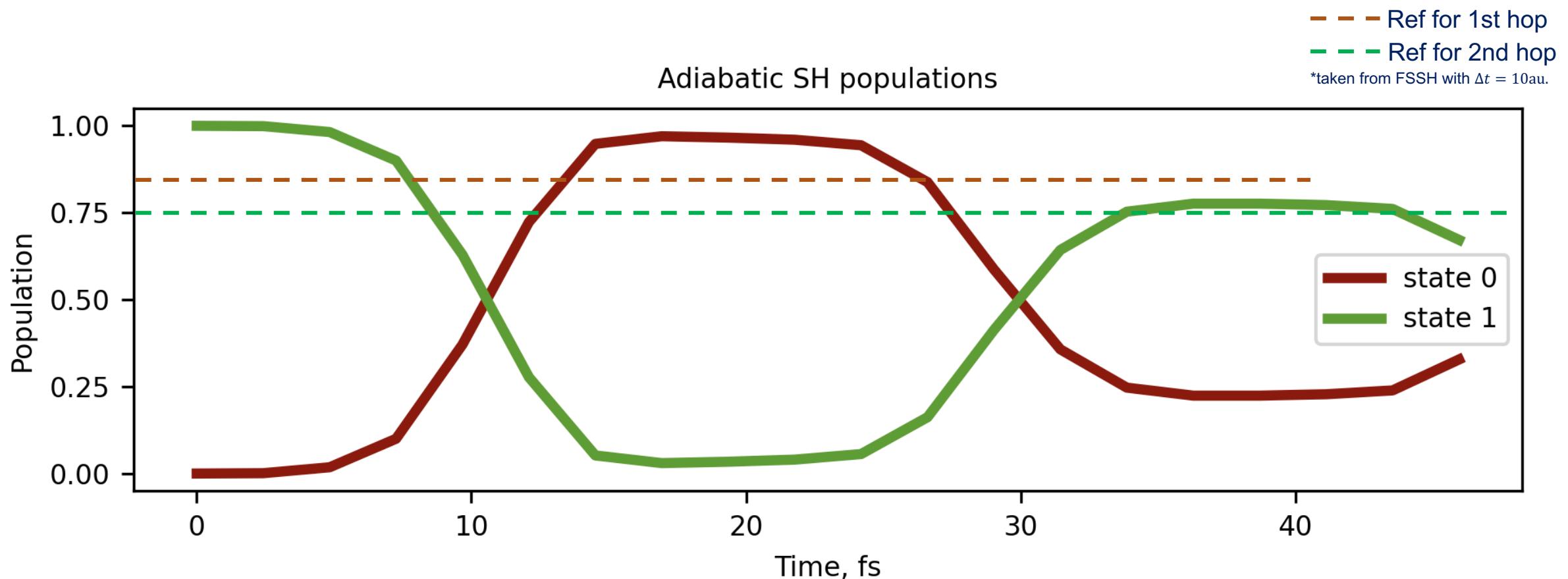


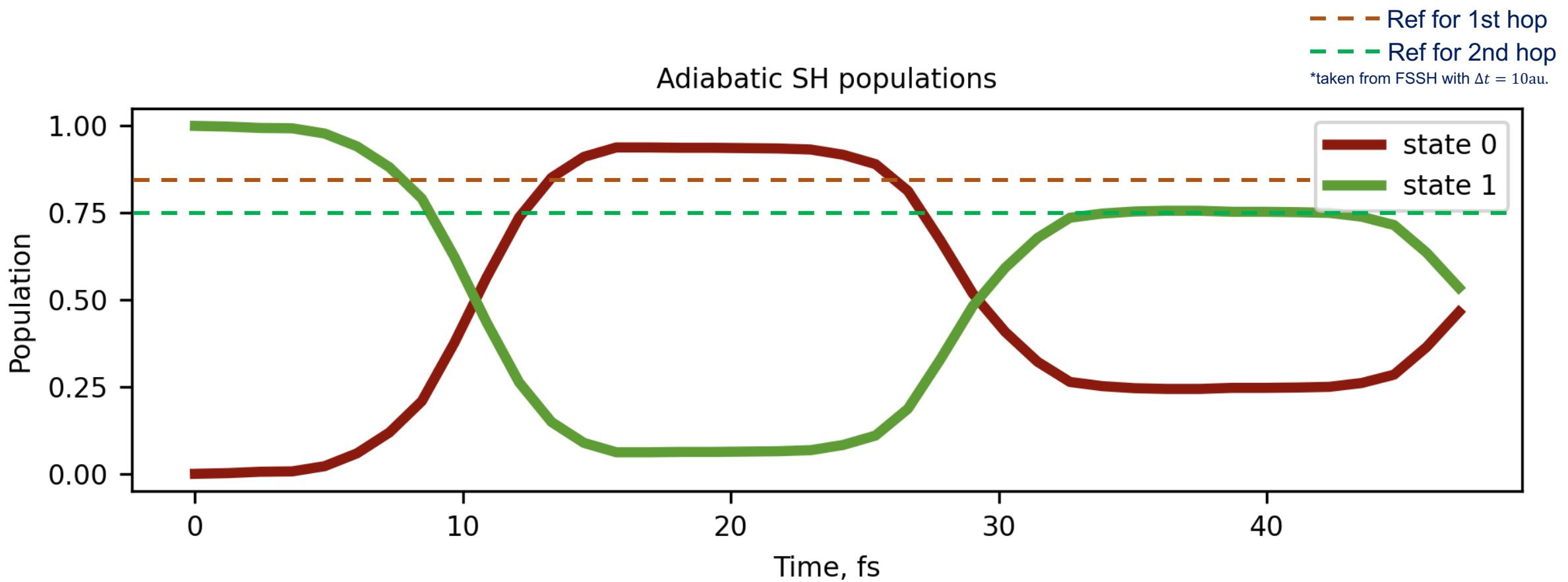
$\Delta t = 50$:

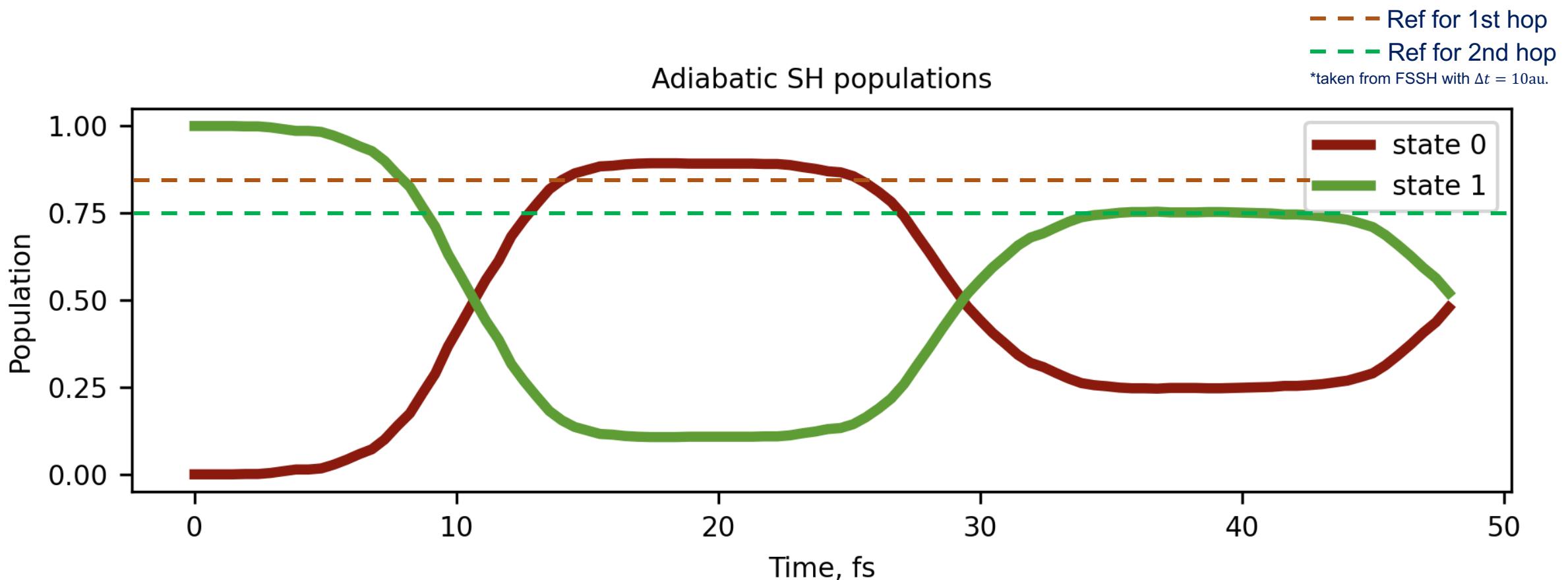


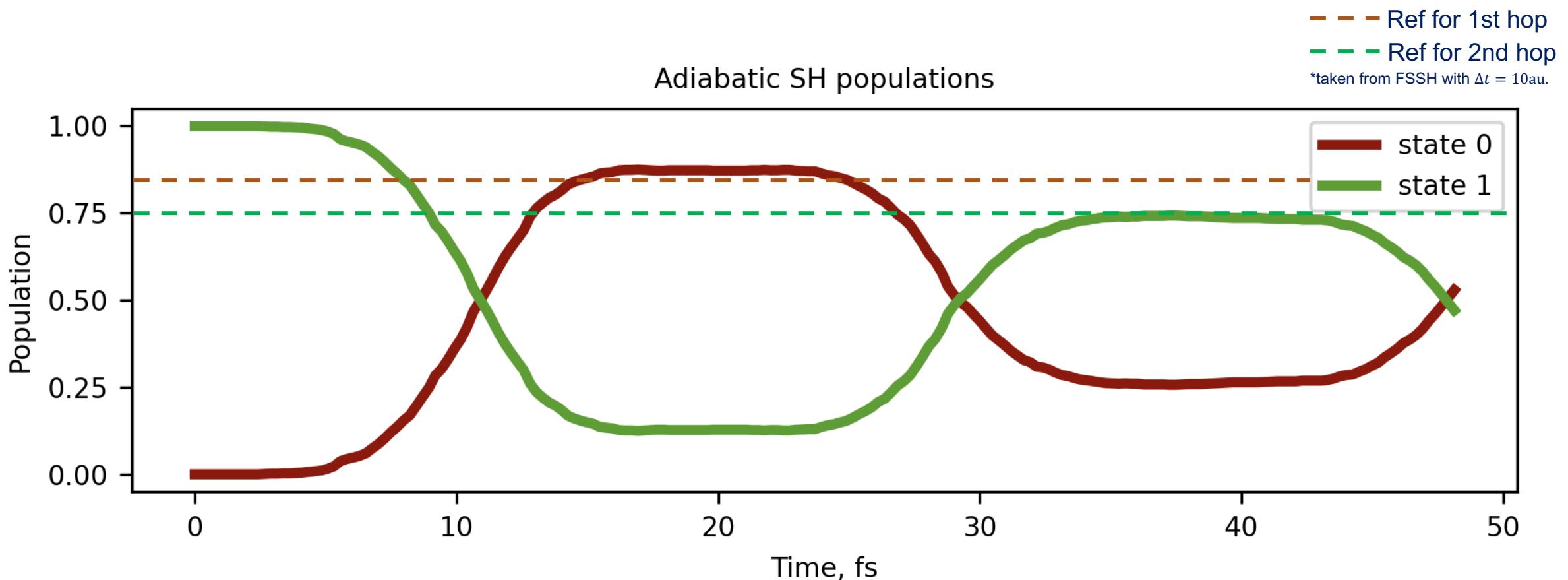
FSSH2 – observations

- Energy was already conserved at $\Delta t = 100\text{au}$.
- Accordingly, FSSH2 provided converged results at around $\Delta t = 100\text{au}$. (I could not try larger time step sizes since the system showed an error: division by zero)



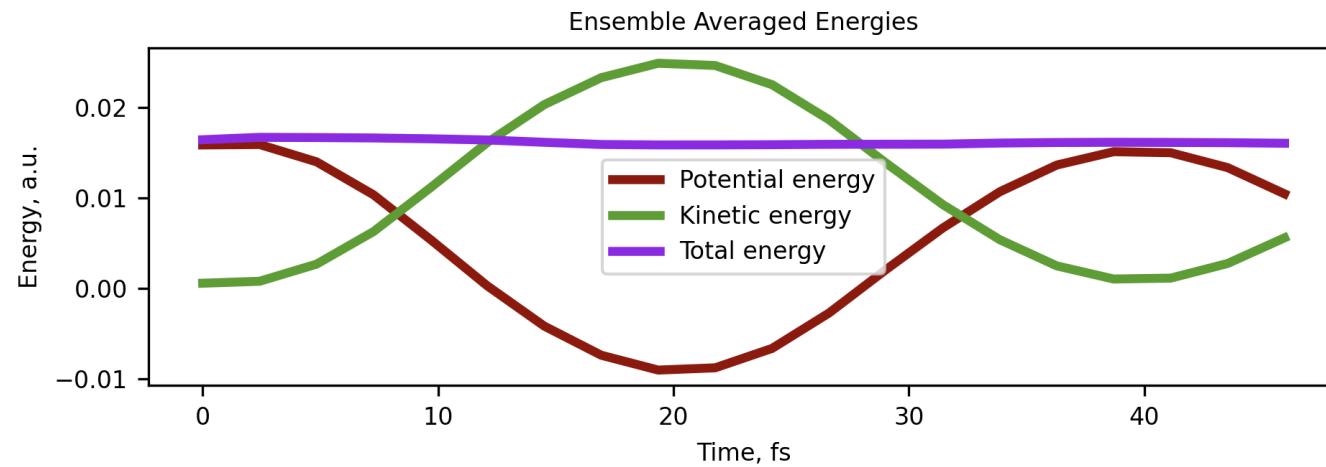




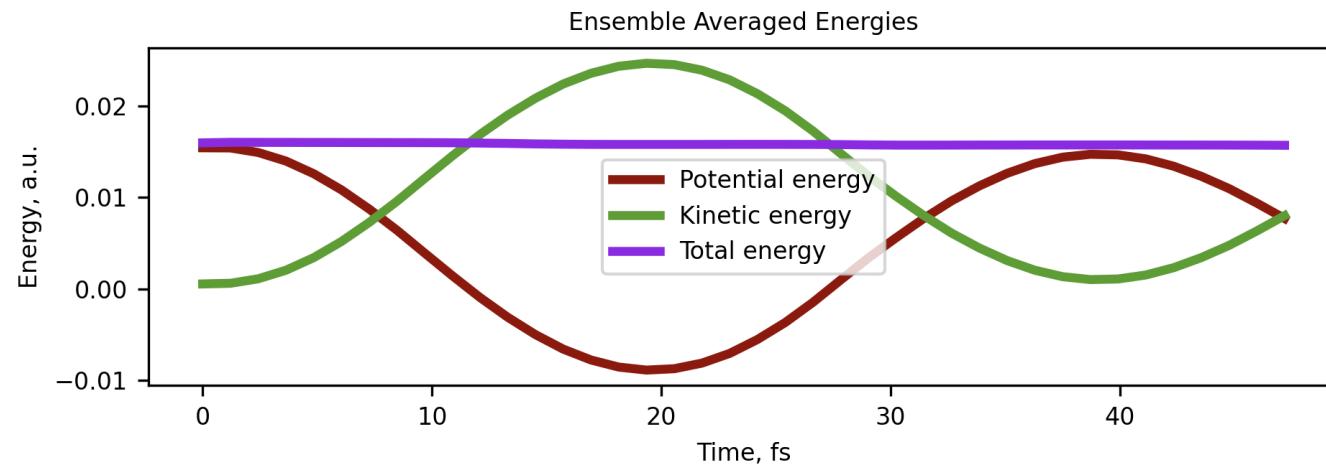


GFSH – energy conservation

$\Delta t = 100$:



$\Delta t = 50$:



- Same observations as for FSSH.
- It underperforms FSSH2, although their hop formulas are similar, because the implementation in Libra contains a Finite Element Method approximation of the original GFSH formula.



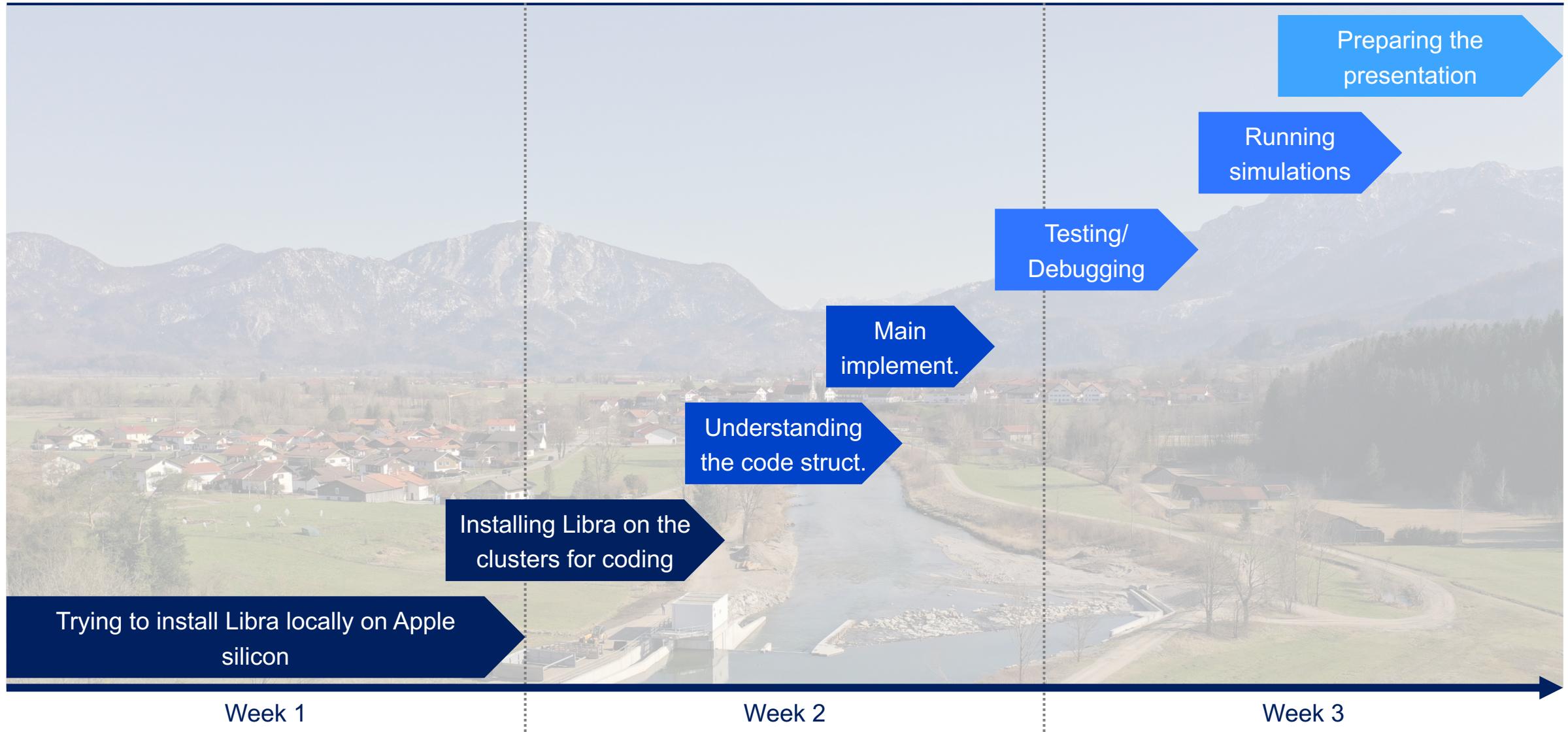
FSSH2 overperformed FSSH by significantly improving the populations for larger time steps and showing less sensitivity.



Energy conservation was an indicator of the convergence of FSSH2 but not of FSSH and GFSH.

Time & workflow overview

Time overview

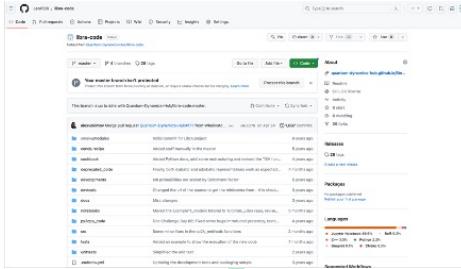


Workflow

CCR

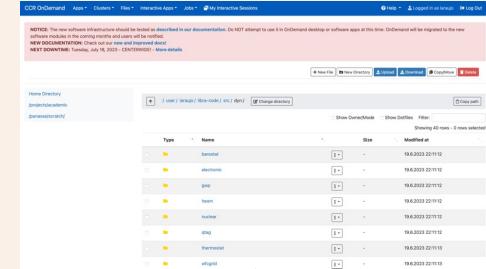
GitHub (Fork)

Pull from the Libra package



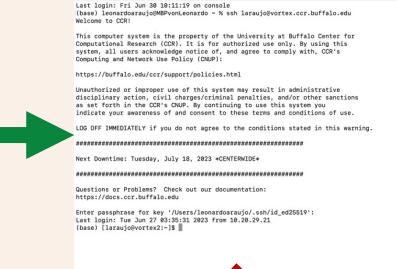
CCR OnDemand

Upload modifications



Terminal

Compile/test/simulate

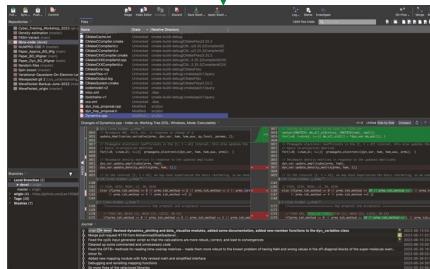


CCR OnDemand

Download results

Type	Name	Size	Modified at
modest-FSSH-100B		-	29.6.2023 22:08:49
modest-FSSH-10B		-	29.6.2023 22:08:23
modest-FSSH-20B		-	29.6.2023 22:16:29
modest-FSSH-00B		-	29.6.2023 22:15:07
modest-FSSH-100B		-	22.6.2023 22:31:09
modest-FSSH-10B		-	29.6.2023 22:23:23
modest-FSSH-20B		-	29.6.2023 22:18:29
modest-FSSH-00B		-	29.6.2023 22:15:10
modest-FSSH-100B		-	29.6.2023 22:11:27
modest-FSSH-10B		-	29.6.2023 22:26:27
modest-FSSH-20B		-	29.6.2023 22:16:08
modest-FSSH-00B		-	29.6.2023 22:20:08
test.o		13.8 KB	29.6.2023 22:08:08

Loc

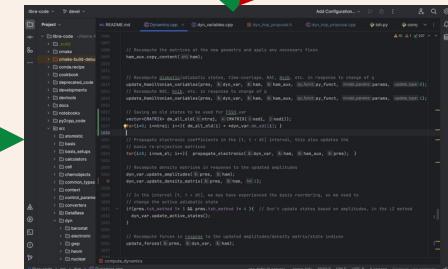


SmartGit

Pull locally/observe modifications

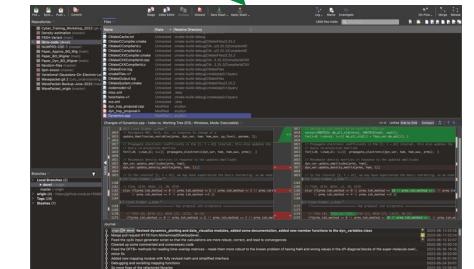
CLion (C++ IDE)

Code/check syntax



SmartGit

Push modifications



Implementation
loop

Modified files: Dynamics.cpp

Saving densities of the previous time step:

```
1012 // Recompute diabatic/adiabatic states, time-overlaps, NAC, Hvib, etc. in response to change of q
1013 update_Hamiltonian_variables(prms, dyn_var, ham, ham_aux, py_funct, params, 0);
1014 // Recompute NAC, Hvib, etc. in response to change of p
1015 update_Hamiltonian_variables(prms, dyn_var, ham, ham_aux, py_funct, params, 1);
1016
1017 // Saving as old states to be used for FSSH_var
1018 vector<CMATRIX> dm_all_old(ntraj, CMATRIX(nadi, nadi));
1019 for(i=0; i<ntraj; i++){ dm_all_old[i] = *dyn_var.dm_adi[i]; }
1020
1021 // Propagate electronic coefficients in the [t, t + dt] interval, this also updates the
1022 // basis re-projection matrices
1023 for(i=0; i<num_el; i++){ propagate_electronic(dyn_var, ham, ham_aux, prms); }
1024
```

Setting tsh_method=10 as FSSH2:

```
1165 // FSSH, GFSH, MSSH, LZ, ZN, DISH
1166 else if(prms.tsh_method == 0 || prms.tsh_method == 10 || prms.tsh_method == 1 || prms.tsh_method == 2 ||
1167     || prms.tsh_method == 4 || prms.tsh_method == 5 ){
1168
1169
1170     vector<int> old_states(dyn_var.act_states);
1171
1172     //===== Hop proposal and acceptance =====
1173
1174     // FSSH (0), FSSH_var (10), GFSH (1), MSSH (2), LZ(3), ZN (4)
1175     if(prms.tsh_method == 0 || prms.tsh_method == 10 || prms.tsh_method == 1 || prms.tsh_method == 2 || pr
1176
1177         /// Compute hop proposal probabilities from the active state of each trajectory to all other states
1178         /// of that trajectory
1179         vector< vector<double> > g;
1180         g = hop_proposal_probabilities(prms, dyn_var, dm_all_old, ham, ham_aux);
1181
1182         // Propose new discrete states for all trajectories
1183         vector<int> prop_states( propose_hops(g, dyn_var.act_states, rnd) );
1184
```

Modified files: dyn_hop_proposal.h

Introducing a new method to compute hop probabilities with the previous densities as additional input:

```
66 vector< vector<double> > hop_proposal_probabilities(dyn_control_params& prms,
67     dyn_variables& dyn_var, nHamiltonian& ham, nHamiltonian& ham_prev);
68
69 vector< vector<double> > hop_proposal_probabilities(dyn_control_params& prms,
70     dyn_variables& dyn_var, vector<CMATRIX>& dm_all_old, nHamiltonian& ham, nHamiltonian& ham_prev);
71
72 //vector< vector<double> > hop_proposal_probabilities(dyn_control_params& prms,
73 //     dyn_variables& dyn_var, nHamiltonian& ham, vector<MATRIX>& prev_ham_dia);
74
```

Modified files: dyn_hop_proposal.cpp (1/2)

Introducing a new option in
hop_proposal_probabilities(...) to compute FSSH2:

```
1284 if(prms.tsh_method == 0){ // FSSH
1285     g[traj] = hopping_probabilities_fssh(prms, dm, Hvib, dyn_var.act_states[traj]);
1286 }
1287 else if(prms.tsh_method == 10){ // FSSH_var
1288     //CMATRIX& dm_old = *dyn_var.old.dm_adi[traj];
1289     CMATRIX dm_old = dm_all_old[traj];
1290     g[traj] = hopping_probabilities_fssh_var(prms, dm, dm_old, Hvib, dyn_var.act_states[traj]);
1291 }
1292 else if(prms.tsh_method == 1){ // GFSH
1293     g[traj] = hopping_probabilities_gfsh(prms, dm, Hvib, dyn_var.act_states[traj]);
1294 }
```

Modified files: dyn_hop_proposal.cpp (2/2)

Compute FSSH2 in a new method
hopping_probabilities_fssh_var(...):

```
375     // Now calculate the hopping probabilities
376     i = act_state_idx;
377
378     sum = 0.0;
379     double a_ii_old = denmat_old.get(i,i).real();
380 // cout<<"a_ii = "<<a_ii<<endl;
381
382     for(j=0;j<nstates;j++){
383
384         if(i!=j){
385             /**
386              dc/dt = -(i/hbar) * Hvib * c
387              (dc/dt)^+ = i/hbar * c^+ * Hvib^+
388
389              rho = c * c^+
390
391              P(i->j) = ( rho_jj - rho_old_jj ) / rho_ii
392
393             */
394
395             double a_jj_old = denmat_old.get(j,j).real();
396             double a_jj      = denmat.get(j,j).real();
397
398             if(a_ii_old<1e-8){ g_ij = 0.0; } // avoid division by zero
399             else{
400                 g_ij = (a_jj - a_jj_old)/a_ii_old; // This is a general case
401
402                 if(g_ij<0.0){ g_ij = 0.0; }
403
404             } // else
405
406             g[j] = g_ij;
407             sum = sum + g_ij;
408         }
409         else{ g[j] = 0.0; }
410
411     } // for j
412
413     g[i] = 1.0 - sum;
414
415     return g;
```

Thank you for your attention!

Any questions?



Proooost!!