TUNA

Theoretical Unification of Nuclear Arrangements

Version 0.5.0

August 2024

Foreword

Welcome to TUNA!

Beginning as a project in my spare time, developing TUNA was intended to deepen my understanding of the theory and algorithmic implementation of quantum chemistry. It's a simple, command line-based Python program designed to calculate properties of diatomic molecules. While modest in scope, TUNA aims to be a helpful tool at the intersection of molecular physics and quantum chemistry, providing an accessible way to explore quantum mechanical calculations in small molecules.

One of TUNA's strengths is its user-friendly interface, which makes it a suitable teaching aid. With clear command line outputs, the program offers an straightforward experience for those looking to learn about quantum chemistry. On the more technical side, TUNA most unique feature is combining the calculation of potential energy surfaces with the numerical solution of the nuclear Schrödinger equation, enabling users to compare anharmonic vibrational frequencies of diatomics using a variety of (post-)Hartree-–Fock methods. The program is also surprisingly efficient, considering it's written in Python.

While TUNA is modest in scope, its capability will improve in the near future. The primary long term aim is to enable it to act as a testing ground for the performance of density-functional approximations on diatomics.

One day, somebody might find the program useful and TUNA could make waves!

Harry Brough

Contents

1 Getting Started **3**

1.1 Installation . . . 3

1.2 Simple Input Line . . . 3

1.3 Units and Constants . . . 4

1.4 Program Components . . . 5

2 Detailed Documentation **6**

2.1 Calculation Types . . . 6

2.1.1 Single Point Energy . . . 6

2.1.2 Coordinate Scan . . . 6

2.1.3 Geometry Optimisation . . . 6

2.1.4 Harmonic Frequency . . . 7

2.1.5 Anharmonic Frequency . . . 8

2.1.6 *Ab Initio* Molecular Dynamics . . . 8

2.2 Electronic Structure Methods . . . 10

2.2.1 Hartree–Fock . . . 10

2.2.1.1 SCF Convergence . . . 10

2.2.2 Møller–Plesset Perturbation Theory . . . 10

2.2.3 Basis Sets . . . 10

2.2.4 Dispersion Correction . . . 11

2.3 Properties . . . 11

2.3.1 Rotational Constants . . . 11

2.3.2 Multipole Moments . . . 12

2.3.3 Population Analysis . . . 12

2.3.4 Electron Density Plot . . . 14

2.3.5 Thermochemistry . . . 14

3 List of Keywords **16**

Bibliography **18**

1 Getting Started

1.1 Installation

Installing TUNA has only been tested on Windows 11 — and not tested very well. The easiest way to install TUNA is first ensuring Python 3.12 or higher is installed, then from a terminal, run:

```
pip install QuantumTUNA
```

If this proceeds according to plan, next locate the folder where TUNA was installed, and add it to PATH. On Windows, where $*$ is your user, this may be:

```
C:\Users\*\AppData\Local\Programs\Python\Python312\Lib\site-packages\TUNA
```

Once the folder containing the TUNA Python files is added to PATH, open a new terminal and test that everything is working correctly by running:

```
TUNA --version
```

If the version of TUNA you installed prints to the terminal, you're good to go!

While it hasn't been tested on these systems, there's a good chance TUNA works on Mac and Linux and with earlier versions of Python 3. Let me know if it doesn't!

1.2 Simple Input Line

The input line reads "TUNA calculation-type: atom1 [atom2 bond-length] ! method basis-set ! parameters". The input line is not case sensitive and spaces around colons are optional.

```
TUNA [Calculation] : [Atom A] [Atom B] [Distance] : [Method] [Basis]
```

For instance, a single point energy calculation on an H2 molecule with bond length of 1.0 Angstroms, with Hartree–Fock in the STO-3G basis set is:

```
TUNA SPE : H H 1.0 : HF STO-3G
```

Additional parameters can be added by adding another colon at the end, and appending the parameters.

An optimisation calculation with the D2 dispersion scheme on a H—He$^+$ molecule starting at a bond length of 0.8 Å , using the 6-31++G basis and spin-component-scaled MP2, is:

```
TUNA OPT : H He 0.8 : SCS-MP2 6-31++G : D2 CHARGE +1 DENSPLOT
```

As many keywords as you like can be written after the second exclamation mark. Some keywords such as `CHARGE` require a value (such as +1 here) — just write this directly after the keyword, and then more keywords can be written afterwards.

The output prints to the terminal by default, but by replacing the command `TUNA` with `TUNALOG` the output will print to a file called "`tuna-calculation.log`" in the directory the terminal was called.

1.3 Units and Constants

In the TUNA output, distances are expressed in angstroms, times in femtoseconds and all other quantities in atomic units unless otherwise stated.

Units in TUNA are set by the CODATA 2022 recommendations [1], shown in Table 1.1. All other units and values are derived from these constants.

Table 1.1 Unit conversions and fundamental constants.

Constant	Symbol	Value
Planck Constant	h	$6.62607015 \times 10^{-34}$ J s
Elementary Charge	e	$1.602176634 \times 10^{-19}$ C
Electron Mass	m_e	$9.1093837139 \times 10^{-31}$ kg
Vacuum Permittivity	$4\pi\varepsilon_0$	$1.11265005620 \times 10^{-10}$ F m^{-1}
Speed of Light	c	2.99792458×10^{8} m s^{-1}
Boltzmann Constant	k_B	1.380649×10^{-23} J K^{-1}
Atomic Mass Unit	amu	$1.660539068911 \times 10^{-27}$ kg

1.4 Program Components

This section lists the functions of the different modules in the TUNA program, which call each other during a calculation. The main program, `tuna.py`, initiates all calculations when TUNA is called from a terminal.

Table 1.2 List of Python modules that constitute TUNA, in alphabetical order.

Module	Description
tuna	Main program, input parsing
tuna_anharm	Anharmonic frequencies
tuna_basis	Main program, input parsing
tuna_dispersion	Semi-empirical dispersion correction
tuna_energy	Calculating energies, building molecules
tuna_integral	Evaluating one- and two-electron integrals
tuna_md	*Ab initio* molecular dynamics
tuna_mp2	Second order Møller–Plesset perturbation theory energy and density
tuna_optfreq	Calculating gradients, optimisation algorithm, harmonic frequencies
tuna_postscf	Calculating one-electron properties
tuna_scf	Main self-consistent field loop, convergence acceleration
tuna_thermo	Thermochemistry after frequency calculation
tuna_util	General utility, units, useful functions

2 Detailed Documentation

2.1 Calculation Types

2.1.1 Single Point Energy

2.1.2 Coordinate Scan

2.1.3 Geometry Optimisation

By default, an approximate Hessian is used to update the geometry in an optimisation, but using the `CALCHESS` keyword, the "exact" Hessian can be calculated numerically for each step. This reduces the number of optimisation steps substantially, but increases the computational cost much more. Therefore, this is probably pointless except maybe for optimising on a very flat potential energy surface.

The approximate Hessian, H, of the ith optimisation step is calculated simply by

$$H_i = \frac{\nabla E_i - \nabla E_{i-1}}{r_i - r_{i-1}} \, . \tag{2.1}$$

For optimisations to a local minimum, either the exact or approximate Hessian is used in the convex region of the potential energy surface, whereas for optimisations to maxima, dynamic Hessians are used in the concave region. Outside of these regions, a default Hessian with value $\frac{1}{4}$ is used.

The bond length is updated by

$$r_i = r_{i-1} + \frac{\nabla E_i}{H_i} \tag{2.2}$$

The maximum value of step allowed is 0.2 Angstrom, to prevent ruining optimisations from a bad starting guess.

By default, the density matrix is read in from each optimisation step; this can be disabled using the `NOMOREAD` keyword.

The trajectory of a geometry optimisation can be printed to an `.xyz` file using the `TRAJ` keyword. The structures and energies at each point are printed to a file called `"tuna_trajectory.xyz"` in the directory of the terminal.

Geometries can also be optimised to a local maximum, rather than a local minimum, using the `OPTMAX` keyword. This simply inverts the treatment of the Hessian and step update calculations, bringing the structure to a local maximum on the potential energy surface. This probably isn't very useful, but could be used, for instance, to find the unphysical local maximum predicted by MP2 on the potential energy surface of H_2.

Four optimisation convergence criteria are available: loose, medium, tight (default) and extreme. For optimisations, SCF convergence criteria is set to extreme by default to prevent build up of noise.

Table 2.1 Optimisation convergence criteria in TUNA

Convergence	**Gradient / hartree bohr^{-1}**	**Step / bohr**
Loose	0.001	0.01
Medium	0.0001	0.0001
Tight	0.000001	0.00001
Extreme	0.00000001	0.0000001

2.1.4 Harmonic Frequency

For instance, a 50 step AIMD calculation using MP2/6-31G with a timestep of 0.2 fs at an initial temperature of 350 K can be called by:

```
TUNA FREQ : H H 0.6750 : RHF 6-311++G
```

```
TUNA OPTFREQ : H H 1.0 : RHF 6-311++G : TEMP 350 PRES 150000
```

$$\omega = \sqrt{\frac{k}{\mu}} \tag{2.3}$$

$$k = \frac{\partial^2 E}{\partial r^2} \tag{2.4}$$

Table 2.2 Atomic masses in TUNA.

Atom	Major Isotope	Mass / amu
Hydrogen	^{1}H	1.00782503223
Helium	^{4}He	4.00260325413

A thermochemistry calculation, detailed in section 2.3.5 is performed after the frequency calculation. A temperature and pressure can be specified for this, for instance:

```
TUNA FREQ : H H 0.6750 : RHF 6-311++G : TEMP 340 PRES 101500
```

2.1.5 Anharmonic Frequency

2.1.6 *Ab Initio* Molecular Dynamics

The implementation in TUNA of *ab initio* molecular dynamics (AIMD) is Born–Oppenheimer molecular dynamics [2], where nuclear positions are updated by the classical equations of motion, $\mathbf{F} = m\mathbf{a}$, over the quantum chemical potential energy surface. Therefore, unlike other AIMD methods such as Car–Parrinello molecular dynamics where the electronic degrees of freedom are also dynamic [3], the *ab initio* energy and gradient must be recalculated at every timestep.

To run an AIMD calculation in TUNA, use the MD calculation type with any electronic structure method and basis, and set the timestep (fs), initial temperature (K) and number of steps with the keywords `TIMESTEP`, `TEMP` and `MDNUMBER` respectively. By default, `TIMESTEP` is 0.1 fs, `TEMP` is 0 K and `MDNUMBER` is 30.

For instance, a 50 step AIMD calculation using MP2/6-31G with a timestep of 0.2 fs at an initial temperature of 350 K can be called by:

```
TUNA MD : H H 1.2 : MP2 6-31G : TIMESTEP 0.2 TEMP 350 MDNUMBER 50
```

As positions and velocities are continually updated, a trajectory is mapped out. By default, TUNA prints this trajectory to a file called "`tuna-trajectory.xyz`" in the directory where the program was called — this can be disabled using the `NOTRAJ` keyword.

The output prints the step number, time, bond length and temperature. The classical kinetic energy of the nuclei, as well as their potential energy (the total electronic energy) is also printed, with the total molecular energy. The timestep should be chosen to conserve the total system energy. At each timestep, all energy and gradient information can be printed using the additional print keyword, `P`, although this makes a big mess.

Because the Velocity Verlet algorithm [4] yields velocities and positions at the same point in time and is numerically stable, it is implemented in TUNA. Positions are updated by

$$\mathbf{r}(t+\Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}\mathbf{a}(t)\Delta t^2, \tag{2.5}$$

and velocities are updated by

$$\mathbf{v}(t+\Delta t) = \mathbf{v}(t) + \frac{\mathbf{a}(t) + \mathbf{a}(t+\Delta t)}{2}\Delta t\ . \tag{2.6}$$

By default, the AIMD calculation begins with the atoms at rest. However, by using the `TEMP` keyword an initial temperature can be specified from which the total classical kinetic energy of the molecule is calculated by

$$K = 3k_{\mathrm{B}}T\ , \tag{2.7}$$

where

$$K = \frac{1}{2}m_1\mathbf{v}_1^2 + \frac{1}{2}m_2\mathbf{v}_2^2\ . \tag{2.8}$$

From this kinetic energy, velocities are randomly determined to satisfy the Maxwell–Boltzmann distribution. Next, net molecular translations are removed before the velocities are rescaled to produce the specified temperature exactly. Net rotations are not removed, so to handle this three-dimensional data — which is stressful for TUNA — the coordinates are rotated onto one dimension to calculate the energy and gradient before forces are back-transformed into three dimensions. The initial specified temperature will not be conserved, as thermostats don't really make sense for diatomics so all AIMD calculations are run in the NVE ensemble.

2.2 Electronic Structure Methods

2.2.1 Hartree–Fock

2.2.1.1 SCF Convergence

In level shifting, the energies of the virtual orbitals are shifted so that after diagonalisation of the Fock matrix, the occupied and virtual orbitals mix less strongly and the calculation smoothly converges. Level shifting is applied via a parameter, b, which updates the Fock matrix by

$$\mathbf{F}' = \mathbf{F} - b\mathbf{P} \tag{2.9}$$

In TUNA, $b = 0.2$ by default. Level shift can be activated by the `LEVELSHIFT` keyword.

2.2.2 Møller–Plesset Perturbation Theory

In TUNA, conventional and spin-component-scaled (SCS) second-order Møller–Plesset perturbation theory (MP2) are implemented.

2.2.3 Basis Sets

Presently, only s orbital integrals are implemented in TUNA, so basis sets are limited to period 1 elements without polarisation functions. Only Pople-style basis sets are available, and custom sets can not yet be read in.

Table 2.3 Basis sets in TUNA

Basis Set	Description
STO-3G	Minimal basis set
STO-6G	Minimal basis set
3-21G	Double-zeta basis set
4-31G	Double-zeta basis set
6-31G	Double-zeta basis set
6-31+G	Double-zeta basis set with diffuse functions on heavy atoms
6-31++G	Double-zeta basis set with diffuse functions on all atoms
6-311G	Triple-zeta basis set
6-311+G	Triple-zeta basis set with diffuse functions on heavy atoms
6-311++G	Triple-zeta basis set with diffuse functions on all atoms

Note: the 6-311++G basis set is not parameterised for helium.

2.2.4 Dispersion Correction

Currently, the only available form of dispersion correction is the D2 semi-empirical scheme by Grimme [5]. Here, the energy due to dispersion is calculated by

$$E_{\text{disp}} = -s_6 \frac{C_6^{ij}}{R_{ij}^6} \frac{1}{f_{\text{damp}}} \ , \tag{2.10}$$

where the damping function is given by

$$f_{\text{damp}} = 1 + \text{exp}\left[-d \left(\frac{R_{ij}}{R_r} - 1 \right) \right] . \tag{2.11}$$

The default value of the damping factor d is 20, and the s_6 value is 1.2 in line with the parameterisation of Hartree–Fock in ORCA. The values of the Van der Waals radii, R_r, and C_6 coefficients are element-specific.

2.3 Properties

The calculation of many molecular properties requires a density matrix, which is trivial for Hartree–Fock calculations but not for correlated calculations. Currently, the MP2 unrelaxed density is implemented for restricted Hartree–Fock references only. All other calculations (such as SCS-MP2 and MP2 on unrestricted references) will print a warning and use the Hartree–Fock density matrix.

After all calculations, parameters using Koopmans' theorem [6] are printed. These are the ionisation energy, $-\epsilon_{\text{HOMO}}$, electron affinity, $-\epsilon_{\text{LUMO}}$ and HOMO–LUMO gap, $\epsilon_{\text{LUMO}} - \epsilon_{\text{HOMO}}$. Calculations of rotational constants, multipole moments and population analysis is disabled for single atoms.

2.3.1 Rotational Constants

Linear molecules only have a single rotational constant, which is simply calculated by

$$B = \frac{1}{2\mu(\mathbf{R}_{\text{A}} - \mathbf{R}_{\text{B}})^2} \ , \tag{2.12}$$

where μ is the reduced mass of the molecule, using the masses shown in section 2.1.4. The value is given by TUNA in gigaherz, and printed just above the dipole moment.

2.3.2 Multipole Moments

After single point energy calculations and at the end of optimisations, TUNA prints out dipole moment information. The origin for all dipole moment calculations is the centre of mass, $\mathbf{R}_0$, and the atomic masses used are the isotopic masses of the most common isotopes, shown in section 2.1.4. This includes the nuclear dipole moment,

$$\boldsymbol{\mu}_{\text{nuc}} = Z_{\text{A}}(\mathbf{R}_{\text{A}} - \mathbf{R}_0) + Z_{\text{B}}(\mathbf{R}_{\text{B}} - \mathbf{R}_0) \ , \tag{2.13}$$

and the electronic dipole moment,

$$\boldsymbol{\mu}_{\text{ele}} = -\sum_{ij} P_{ij} D_{ij} \ , \tag{2.14}$$

where D_{ij} are the dipole moment integrals, calculated with the rest of the one-electron integrals at the start of a calculation.

In addition to the values of the dipole moments, and total dipole moment, a diagram of the molecular structure is printed with an arrow pointing in the direction of the negative charge.

2.3.3 Population Analysis

By default, TUNA prints Mulliken [7], Löwdin [8] and Mayer [9] population analysis after single point energy calculations and optimisations. Note that all the equations in this section are true for diatomics, not necessarily molecules in general.

The Mulliken analysis is very widely used (despite its considerable weaknesses [10]), and partitions the electron density by the basis functions assigned to each atom. The number of electrons, N, is the integral of the density and can be calculated by

$$N = \sum_{\mu\nu} P_{\mu\nu} S_{\mu\nu} \ . \tag{2.15}$$

When the shared density is partitioned equally between atoms, the number of electrons on atom A is

$$N_{\text{A}} = \sum_{\mu\in\text{A}} \sum_{\nu\in\text{A}} P_{\mu\nu}^{\text{AA}} S_{\mu\nu}^{\text{AA}} + \frac{1}{2} B_{\text{AB}} \ , \tag{2.16}$$

where B_{AB} is the Mulliken bond order, given by

$$B_{\mathsf{AB}} = 2\sum_{\mu\in\mathsf{A}}\sum_{\nu\in\mathsf{B}} P_{\mu\nu}^{\mathsf{AB}} S_{\mu\nu}^{\mathsf{AB}} \ . \tag{2.17}$$

The charge of an atom in a molecule is then given by $Q_{\mathsf{A}} = Z_{\mathsf{A}} - N_{\mathsf{A}}$. At the bottom of the single point energy output, TUNA prints these charges and the Mulliken bond order, as well as the sum of the charges which should equal the requested molecular charge.

Because the molecular orbitals than generate a specific energy and density are not unique, population analysis can be performed with non-localised orthogonal basis functions, in Löwdin analysis. Here, the density matrix is rotated by

$$\mathbf{P}^{\mathsf{L}} = \mathbf{S}^{\frac{1}{2}}\mathbf{P}\mathbf{S}^{\frac{1}{2}} \ . \tag{2.18}$$

Using this density matrix, atomic populations are calculated by

$$N_{\mathsf{A}} = \sum_{\mu\in\mathsf{A}} \mathbf{P}_{\mu\mu}^{\mathsf{L}} \ , \tag{2.19}$$

and the bond order is calculated by

$$B_{\mathsf{AB}} = \sum_{\mu\in\mathsf{A}}\sum_{\nu\in\mathsf{B}} (P_{\mu\nu}^{\mathsf{L}})^2 \ . \tag{2.20}$$

Similarly to the Mulliken analysis, Löwdin charges and bond order are printed at the end of a calculation. With these two outputs, the results of the Mayer analysis are also printed.

The Mayer charges are the same as the Mulliken charges, but the bond order (and bonded valence) is given by

$$B_{\mathsf{AB}} = \sum_{\mu\in\mathsf{A}}\sum_{\nu\in\mathsf{B}} (\mathbf{PS})_{\mu\nu}(\mathbf{PS})_{\nu\mu} + (\mathbf{RS})_{\mu\nu}(\mathbf{RS})_{\nu\mu} \ , \tag{2.21}$$

where $\mathbf{R}$ is the spin density matrix, $\mathbf{P}_\alpha - \mathbf{P}_\beta$. In Mayer analysis the valence of each atom is calculated. The total valence, which is similar to the expected valence of an atom, is

$$V_{\mathsf{A}} = 2N_{\mathsf{A}} - \sum_{\mu\in\mathsf{A}}\sum_{\nu\in\mathsf{B}} (\mathbf{PS})_{\mu\nu}(\mathbf{PS})_{\nu\mu} \tag{2.22}$$

and the free valence is calculated by $F_{\mathsf{A}} = V_{\mathsf{A}} - B_{\mathsf{AB}}$, which is a measure of the ability to form further bonds. These valences are printed under the Mayer analysis heading at the end of calculations.

2.3.4 Electron Density Plot

A three-dimensional plot of the electron density can be calculated using the `DENSPLOT` keyword after an energy calculation. This can be combined with, for instance, optimisations or MD runs, to show the change in electron density over time.

However, the calculation of the cube files is very slow and the default parameters to view the iso-surface may crop the density incorrectly in some cases. But can be fun to see the density converge in an optimisation.

2.3.5 Thermochemistry

After the vibrational frequency calculation is finished TUNA begins a quick calculation of thermochemical parameters. The contributions to internal energy,

$$U = E + E_{\text{zero-point}} + E_{\text{translational}} + E_{\text{vibrational}} + E_{\text{rotational}} \ , \tag{2.23}$$

and entropy,

$$S = S_{\text{electronic}} + E_{\text{translational}} + S_{\text{vibrational}} + S_{\text{rotational}} \ , \tag{2.24}$$

are first calculated, before the enthalpy,

$$H = U + k_{\text{B}}T \ , \tag{2.25}$$

and Gibbs free energy,

$$G = H - TS \ , \tag{2.26}$$

are evaluated and printed. All these values, including entropies, are expressed in hartree.

The zero-point energy is calculated by $E_{\text{zero-point}} = \frac{1}{2}\omega$, and is excluded if the vibrational frequency is imaginary. The translational contribution to energy is $E_{\text{translational}} = \frac{3}{2}k_{\text{B}}T$ and the rotational contribution is $E_{\text{rotational}} = k_{\text{B}}T$. Finally, the vibrational contribution is calculated by

$$E_{\text{vibrational}} = \frac{\omega}{\exp\left(\omega/k_{\text{B}}T\right) - 1} \ , \tag{2.27}$$

assuming the molecule behaves as a harmonic oscillator — at high temperatures this may be a poor approximation.

The electronic contribution to entropy is assumed to be zero, but the translational, vibrational and rotational contributions are calculated by three fairly horrible formulas, where p is the pressure and B is the rotational constant:

$$S_{\text{translational}} = k_{\text{B}}\left[\frac{5}{2} + \ln\left(\frac{Mk_{\text{B}}T}{h}\right)^{\frac{3}{2}} + \ln\left(\frac{k_{\text{B}}T}{p}\right)\right] \tag{2.28}$$

$$S_{\text{vibrational}} = k_{\text{B}}\left[\frac{\omega}{k_{\text{B}}T\Big(\exp(\omega/k_{\text{B}}T) - 1\Big)} - \ln\Big(1 - \exp\Big(-\frac{\omega}{k_{\text{B}}T}\Big)\Big)\right] \tag{2.29}$$

$$S_{\text{rotational}} = k_{\text{B}}\left[1 + \ln\left(\frac{k_{\text{B}}T}{\sigma Bhc}\right)\right] \tag{2.30}$$

For the rotational entropy, the symmetry number, σ, is determined from the molecular point group. Symmetric, $D_{\infty\text{h}}$ molecules have $\sigma = 2$ while asymmetric, $C_{\infty\text{v}}$ molecules have $\sigma = 1$.

The entropy, and therefore the Gibbs free energy depends on the temperature and pressure. Using the `TEMP` and `PRES` keywords, these quantities can be specified. For instance, a thermochemistry calculation at 340 K and 101500 Pa can be called by:

```
TUNA FREQ : H H 0.6750 : RHF 6-311++G : TEMP 340 PRES 101500
```

3 List of Keywords

Table 3.1 All of the mandatory and optional keywords in TUNA, grouped by function.

Parameter	Description
P	Print more information to the terminal
T	Print less information to the terminal
CH [int] or CHARGE [int]	Molecular charge
ML [int] or MULTIPLICITY [int]	Molecular multiplicity
DECONTRACT	Decontract basis set
ROTATE	Rotate orbitals for initial guess
NOROTATE	Don't rotate orbitals for initial guess
LOOSE or LOOSESCF	Loose SCF convergence
MEDIUM or MEDIUMSCF	Medium SCF convergence
TIGHT or TIGHTSCF	Tight SCF convergence
EXTREME or EXTREMESCF	Extreme SCF convergence
DAMP	Dynamic damping for SCF convergence
NODAMP	No dynamic damping for SCF convergence
LEVELSHIFT	Level shift for SCF convergence
NOLEVELSHIFT	No level shift for SCF convergence
DIIS	Fock matrix extrapolation for SCF convergence
NODIIS	No Fock matrix extrapolation for SCF convergence
SLOWCONV	High static damping (50%) for SCF convergence
VERYSLOWCONV	Extreme static damping (85%) for SCF convergence

Parameter	Description
`MAXITER` [int]	Maximum number of SCF iterations
`LOOSEOPT`	Loose geometry convergence
`MEDIUMOPT`	Medium geometry convergence
`TIGHTOPT`	Tight geometry convergence
`EXTREMEOPT`	Extreme geometry convergence
`CALCHESS`	Calculate exact Hessian for optimisation
`D2`	Semi-empirical dispersion correction with D2
`MOREAD`	Use density from previous step
`NOMOREAD`	Recalculate density from scratch each step
`OPTMAX`	Optimise to a local maximum, rather than a minimum
`TRAJ`	Write trajectory to `tuna-trajectory.xyz`
`NOTRAJ`	Do not write trajectory
`MAXGEOMITER` [int]	Maximum number of optimisation steps
`SCANSTEP` [float]	Distance increment for coordinate scan
`SCANNUMBER` [int]	Number of distance increments for coordinate scan
`SCANPLOT`	Plot potential energy surface from coordinate scan
`DENSPLOT`	Plot 3D electron density
`TIMESTEP` [float]	Timestep for MD simulation
`MDNUMBER` [int]	Number of timesteps for MD simulation
`TEMP` [float] or `TEMPERATURE` [float]	Sets temperature for thermochemistry and MD
`PRES` [float] or `PRESSURE` [float]	Sets pressure for thermochemistry calculations

Bibliography

(1) P. J. Mohr, E. Tiesinga, D. B. Newell and B. N. Taylor, *Codata Internationally Reconmmended 2022 Values of the Fundamental Physical Constants*, en, 2024.

(2) R. Iftimie, P. Minary and M. E. Tuckerman, *Proc. Natl Acad. Sci.*, 2005, **102**, 6654–6659.

(3) R. Car and M. Parrinello, *Phys. Rev. Lett.*, 1985, **55**, 2471–2474.

(4) W. C. Swope, H. C. Andersen, P. H. Berens and K. R. Wilson, *J. Chem. Phys.*, 1982, **76**, 637–649.

(5) S. Grimme, *J. Comput. Chem.*, 2006, **27**, 1787–1799.

(6) T. Koopmans, *Physica*, 1934, **1**, 104–113.

(7) R. S. Mulliken, *J. Chem. Phys.*, 1955, **23**, 1833–1840.

(8) P.-O. Lowdin, *J. Chem. Phys.*, 1950, **18**, 365–375.

(9) I. Mayer, *Chem. Phys. Lett.*, 1983, **97**, 270–274.

(10) E. R. Davidson and A. E. Clark, *Int. J. Quantum Chem.*, 2022, **122**, e26860.