Contents

Installation and Environment Setup

1. WSL and MobaXterm Setup

Use MobaXterm for connecting and editing files remotely.
To use WSL with Ubuntu on Windows:

- Install WSL using:

```
wsl --install
```

- Use Ubuntu from the Microsoft Store.
- In MobaXterm, connect via SSH to the WSL instance.

2. Backing up WSL Linux

To backup your WSL setup:

```
wsl --export Ubuntu backup.tar
```

To restore:

```
wsl --import UbuntuRestored C:\WSL\UbuntuRestored backup.tar
```

3. Useful Linux Tools

Install commonly used tools:

```
sudo apt update
sudo apt install build-essential python3 python3-pip unzip git
```

4. Working with the Tutorial

Download the tutorial repository:

```
git clone https://github.com/jcroldao123/chem-scripts.git
cd chem-scripts
make install
```

To generate documentation:

```
cd docs
make html
```

5. Links and Shortcuts

- Use `Ctrl+Shift+C/V` for copy/paste in terminal.
- Gaussian Basis Sets: `https://www.basissetexchange.org`
- MobaXterm download: `https://mobaxterm.mobatek.net`
- Ubuntu WSL install: `https://learn.microsoft.com/en-us/windows/wsl/install`

Preface

This manual was developed with the goal of supporting students, researchers, and practitioners in the field of quantum chemistry and molecular simulations. It compiles, in a unified and didactic way, practical instructions and theoretical notes on some of the most widely used computational chemistry packages: Gaussian, ORCA, Dalton, GAMESS, Turbomole, Molcas, FCclasses, GROMACS, and LAMMPS.

The content reflects the author's experience in teaching, research supervision, and collaborative projects — with emphasis on usability, reproducibility, and automation of simulations. Many of the scripts and examples presented here come directly from real research contexts and student theses. The manual is meant to be dynamic. As tools evolve, so will this tutorial — especially thanks to the valuable feedback from students and colleagues who apply it in their own workflows.

To the students: never hesitate to explore, question, and break things (on purpose). Science moves forward through curiosity.

Juan Carlos Roldao

Chapter 1

Gaussian

1.1 Introduction

Gaussian is a widely used quantum chemistry program that supports a range of methods including HF, DFT, MP2, and TDDFT.

1.2 Basic Input

```
#P B3LYP/6-31G(d) Opt Freq

Title Card Required

0 1
C  0.000  0.000  0.000
H  0.000  0.000  1.089
...
```

1.3 Functional Tuning (ω tuning)

Generating Additional Files

- Use `formchk` to convert `.chk` to `.fchk`.
- Use `cubegen` to generate cube files for densities or MOs.
- Use `pop=full` or `pop=chelpg` to extract atomic charges.

Example:

```
formchk molecule.chk molecule.fchk
cubegen 0 density=SCF molecule.fchk density.cube -2 h
```

Plotting Electrostatic Potential

- Use GaussView to open `.fchk` and generate electrostatic potential maps.
- Export 2D or 3D plots as `.eps` for publications.

Functional Tuning (ω tuning)

- Use CAM-B3LYP and related functionals for charge-transfer states.
- Optimize the ω parameter using ionization potential/electron affinity matching.

Chapter 2

Dalton

2.1 Introduction

Dalton is a powerful electronic structure program for calculating molecular properties such as polarizabilities, electronic excitation energies, and multipole moments.

2.2 Input Files

A Dalton calculation requires at least two files:

- `.mol` – molecular structure and geometry
- `.dal` – specification of the calculation

Example `.mol` file

```
Charge=0  Multiplicity=1
O     0.000000    0.000000    0.000000
H     0.757160    0.586260    0.000000
H    -0.757160    0.586260    0.000000
```

Example `.dal` file

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.HF
**END OF INPUT
```

2.3 Running the Calculation

You can execute the calculation using:

```
dalton input -mol input.mol -dal input.dal
```

Adjust this command as necessary for SLURM or PBS batch submission.

2.4 Output Analysis

Dalton produces a `.out` file which contains the results of the calculation. Key elements include:

- Total energy
- Final geometry
- MO coefficients
- Transition properties

Examine the output with terminal tools like `less`, `grep` or use a text editor.

Notes

- Dalton requires properly formatted input files to function.
- Errors are usually caught during the input parsing phase.

Advanced Usage and TDDFT in Dalton

TDDFT Setup

To run TDDFT in Dalton, your `.dal` file should include:

```
**DALTON INPUT
.RUN RESPONSE
**RESPONS
.TDDFT
*READ
NSTATE=10
**END OF INPUT
```

Spin-Orbit Coupling (SOC)

For SOC calculations, Dalton requires proper symmetry and relativistic settings:

```
**HAMILTONIAN
.SPIN-ORBIT
```

Reusing Occupation Information

You may reuse converged occupation from one run to initialize another. Use `.mol` with previous orbital information or rename generated files accordingly.

Solvent Effects

Use PCM for implicit solvent:

```
**PCM
.SOLVENT
WATER
```

Transition Properties and Analysis

Dalton can compute transition dipole moments, oscillator strengths, and excitation energies. These appear in the `.out` file and can be parsed with scripts.

Tips for Calculations with Metal Complexes

- Use effective core potentials (ECPs) if available.
- Choose suitable basis sets (e.g., def2-TZVP for metals).
- Symmetry can affect convergence and property calculations — disable if needed.

Chapter 3

Turbomole

3.1 Introduction

Turbomole is an efficient quantum chemistry software package for performing Hartree-Fock, DFT, MP2 and related calculations.

It is driven primarily through terminal-based scripts and text files.

3.2 Setting Up a Calculation

To initialize a working directory, use the `define` script:

```
define
```

You will be prompted to input:

- Molecular geometry (cartesian or internal coordinates)
- Basis sets and atomic types
- Charge and multiplicity
- SCF and calculation method

After setup, several files are generated including:

- `control` – main input file
- `coord` – geometry
- `basis` – basis set definitions

3.3 Running Jobs

Use specific modules for different job types. For example:

```
dscf
ridft
mpgrad
escf
```

These programs read the `control` file and produce output in standard text files.

3.4 Analyzing Output

After execution, key results are stored in:

- `job.last` – final log
- `energy` – total energy summary
- `gradient` – forces
- `mos` – molecular orbitals

You can visualize orbitals with external tools like `Turbomole-TurboPlot` or export for other viewers.

3.5 Sample Workflow

```
define
dscf > dscf.out
ridft > ridft.out
```

Check each step and review `control` parameters as needed.

Notes

- Use `t2x` to convert Turbomole output to other formats.
- All files are text-based and editable.
- Scripts like `jobex` automate geometry optimization loops.

Advanced Tips for Turbomole

Input File Setup

Use the `define` script interactively to prepare input files:

- Geometry: read from `coord` or manually input
- Charge and multiplicity: set in `control`
- Basis set: select from internal list (e.g., def-SVP, def2-TZVP)

Geometry Conversion

Convert coordinates from Gaussian (.gjf) or ORCA (.xyz) using external tools or scripts. Make sure the atom ordering and units are correct.

Vertical Transitions with ESCF

Add to the `control` file:

```
$escf
  irrep=a1
  nroots=10
```

Then run:

```
escf > escf.out
```

Job Management

Use `jobex` for geometry optimization:

```
jobex -gcart > opt.out
```

Use `ridft`, `dscf` and `grad` for specific tasks.

Output Files and Analysis

Key outputs:

- `control` – main configuration
- `energy` – SCF energy values
- `mos` – molecular orbitals
- `gradient` – for optimization steps

Visualization and Conversion

Use `t2x` and tools like Avogadro or VMD for orbital and structure visualization. You may convert Turbomole output to cube or Molden formats.

3.6 Getting Started

TmoleX GUI

TmoleX is a graphical interface for Turbomole that allows easier creation and submission of jobs. It is available for Linux and Windows and is useful for geometry visualization, setup, and basic analysis.

Shell Setup

To load Turbomole:

```
source /opt/turbomole/Config_turbo_env
```

3.7 Ground-State Optimization

Use the following commands in the working directory:

```
define
jobex -ri -c 200
```

After geometry optimization:

```
aoforce > freq.out
```

To extract frequencies:

```
grep -A1 'frequency' freq.out
```

3.8 Excited-State Calculations

Perform TDDFT vertical excitation calculations:

```
escf > escf.out
```

For excited-state optimizations:

```
jobex -ri -level escf -c 200
```

Calculate frequencies in excited states:

```
escf > escf.out
aoforce -level escf > freq_es.out
```

3.9 UV-Vis Spectrum Generation

Use the `t2x` and `spectrum` tools or post-process with Multiwfn.

Plot oscillator strengths vs excitation energy using:

```
grep 'excitation energy' escf.out
```

3.10 Troubleshooting and Cluster Execution

- Fix permission issues with:

  ```
  chmod -R u+rwX .
  ```

- Floating-point exceptions may indicate poor geometry or convergence failure.
- For cluster usage, prepare a `.pbs` file:

Example PBS Script

```
#!/bin/bash
#PBS -N turbomole_job
#PBS -l nodes=1:ppn=8
cd $PBS_O_WORKDIR
source /opt/turbomole/Config_turbo_env
jobex > job.out
```

Chapter 4

FCclasses

4.1 Introduction

FCclasses is a tool for simulating vibronic spectra using the Franck-Condon (FC) approximation and Duschinsky rotation. It allows the modeling of absorption and emission spectra including vibrational structure.

4.2 Input Preparation

FCclasses requires:

- Normal mode frequencies and displacement vectors for ground and excited states
- Harmonic force constants
- Duschinsky rotation matrix

These can be obtained from quantum chemistry software (e.g., Gaussian) and post-processed.

4.3 Key Files and Format

Main input files:

- `freq.g`: frequencies and normal modes of ground state
- `freq.e`: frequencies and normal modes of excited state
- `geom.g`, `geom.e`: equilibrium geometries
- `fcclasses.inp`: parameter file

Sample `fcclasses.inp`

```
&input
  temp = 298.0,
  e0 = 0.0,
  broad = 200.0,
  maxquanta = 10,
  intensity = 1,
  npoints = 1000,
  xmin = 0.0,
  xmax = 5.0
/
```

4.4 Running FCclasses

Run the program using:

```
fcclasses < fcclasses.inp > fcclasses.out
```

Outputs include:

- `fcclasses.out` – summary of input and computational results
- `absorption.dat`, `emission.dat` – spectra
- `stick.dat` – discrete transitions

4.5 Analysis and Plotting

Spectra can be visualized using plotting software (e.g., gnuplot, matplotlib):

```
plot "absorption.dat" using 1:2 with lines
```

Notes

- Frequencies must be in the same units and consistent across files.
- Geometry alignment between states is critical.
- Scripts can automate file extraction from Gaussian outputs.

Advanced Use Cases for FCclasses

Simplified Workflow

FCclasses can work with simplified files from Gaussian:

- Convert log files to `.fchk` using `formchk`
- Extract normal modes and displacements using custom Python scripts or Multiwfn

Absorption and Emission Spectra

You can simulate both:

- AH (Adiabatic Hessian) approach – uses optimized geometries for each state
- TI (Time-Independent) approximation – using a single geometry

Dark States and Intensity

Some excited states may be dark (oscillator strength ≈ 0), but FCclasses can still simulate vibrational transitions for these states. Adjust broadening or restrict modes to better match experimental shapes.

Troubleshooting Tips

- Use consistent basis sets and functionals across states
- Check Duschinsky rotation matrix dimensions and sanity
- Align geometries before computing displacements

Automated Input Preparation

Scripts can assist in generating:

- Geometry files from Gaussian logs
- Displacement vectors
- FCclasses input files (`fcclasses.inp`)

Visualization

Plot stick spectra and convoluted spectra for absorption/emission using:

```
gnuplot
matplotlib
xmgrace
```

Chapter 5

ORCA

5.1 Introduction

ORCA is a flexible and powerful quantum chemistry package designed for modern electronic structure methods, including DFT, TDDFT, and multireference methods.

5.2 TDDFT Calculations

To run TDDFT with ORCA:

```
! B3LYP def2-SVP TightSCF TDDFT
%tddft
  nroots 10
end
```

5.3 Spin-Orbit Coupling

To include spin-orbit coupling (SOC):

```
! B3LYP def2-TZVP SOC TDDFT
```

5.4 Solvation Models

You can use CPCM or SMD for solvation:

```
%cpcm
  smd true
  solvent "Acetonitrile"
end
```

5.5 Visualization and Analysis

- Use `orca_plot` to extract orbital plots.
- Use Chemcraft or Avogadro to visualize orbitals and cube files.
- Use Multiwfn to analyze electron density and ESP maps.

5.6 Excited-State Analysis

Check the `.out` file for:

- Excitation energies and oscillator strengths
- Orbital contributions: e.g., HOMO $\rightarrow$ LUMO
- Natural transition orbitals (NTOs)

5.7 Tips

- Use `maxdim` and `tol` in `%tddft` block for convergence tuning.
- Inspect orbital contributions for charge transfer vs local excitations.
- Use `NumFreq` for numerical frequencies if analytic gradients are not available.

Chapter 6

LAMMPS

6.1 Introduction

LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) is a classical molecular dynamics code designed for parallel computation of materials science and biomolecular systems.

6.2 Input Files

A LAMMPS simulation generally requires:

- `in.lammps` – input script with simulation parameters
- `data.lammps` – atomic coordinates, bonds, angles, etc.
- Potential files – e.g., Lennard-Jones, EAM, ReaxFF parameters

Example: `in.lammps`

```
units real
atom_style full
read_data data.lammps

pair_style lj/cut 10.0
pair_coeff * * 0.1 3.0

fix 1 all nvt temp 300.0 300.0 100.0
timestep 1.0
run 10000
```

6.3 Running LAMMPS

To execute:

```
lmp_serial -in in.lammps
```

Or for parallel execution:

```
mpirun -np 4 lmp_mpi -in in.lammps
```

6.4 Output Files

Key outputs:

- `log.lammps` – simulation log
- `dump.lammpstrj` – trajectory file
- `thermo.out` – temperature, pressure, energy

6.5 Visualization

Use `VMD`, `Ovito` or similar tools to view trajectories:

```
vmd -lammpstrj dump.lammpstrj
```

Notes

- Units must match the chosen style (e.g., `real`, `metal`, `lj`).
- LAMMPS is script-driven and highly customizable.
- Use `fix` commands to control dynamics, thermostats, constraints.

Chapter 7

Molcas

7.1 Introduction

OpenMolcas is a powerful quantum chemistry package particularly suited for multireference methods like CASSCF and CASPT2. It supports a wide range of post-HF methods and spin-orbit coupling calculations.

7.2 Active Space Selection

Use the Grid Viewer (GV) to inspect molecular orbitals and guide the choice of active space:

- Choose π and π^* orbitals for conjugated systems
- Include lone pairs or nonbonding orbitals when relevant
- Exclude very low or very high virtual orbitals unless involved in excitation

7.3 Input Structure

Molcas input files typically include modules:

```
&GATEWAY
Title = molecule
Coord = molecule.xyz
Basis = cc-pVDZ

&SEWARD

&SCF

&RASSCF
Inactive = 10
Active = 4
```

```
Ras2 = 4
Nactel = 4 0 0

&CASPT2

&RASSI
NrofJobI = 2
```

7.4 Optimization and State Averaging

Use `SLAPAF` for geometry optimization.

State averaging:

```
CIROOT = [2,2,1]
```

Distribute weights across singlet/triplet states appropriately.

7.5 Spectroscopic Applications

Use `RASSI` for transition dipoles, SOC, and transition intensities.

- Combine ground and excited states from RASSCF/CASPT2
- Calculate absorption and emission spectra (ESA/GSA)

7.6 Best Practices

- Use Cholesky decomposition to speed up integrals
- Start with small active spaces, then increase
- Always check orbitals and CI vectors visually
- Use `&ALASKA` for analytical gradients where possible

Chapter 8

GROMACS

8.1 Introduction

GROMACS is a versatile package for performing molecular dynamics, primarily for biochemical molecules such as proteins, lipids, and nucleic acids.

8.2 Basic Workflow

Typical GROMACS workflow includes:

1. Preparing the structure (`.gro`, `.pdb`)
2. Generating topology files
3. Defining the simulation box and solvation
4. Adding ions
5. Energy minimization
6. Equilibration (NVT, NPT)
7. Production MD

8.3 Key Commands

1. Structure and Topology

```
gmx pdb2gmx -f molecule.pdb -o processed.gro -water spce
```

2. Define Box

```
gmx editconf -f processed.gro -o newbox.gro -c -d 1.0 -bt cubic
```

3. Solvate

```
gmx solvate -cp newbox.gro -cs spc216.gro -o solvated.gro -p topol.top
```

4. Add Ions

```
gmx grompp -f ions.mdp -c solvated.gro -p topol.top -o ions.tpr
gmx genion -s ions.tpr -o solv_ions.gro -p topol.top -pname NA -nname CL -neutral
```

5. Energy Minimization

```
gmx grompp -f minim.mdp -c solv_ions.gro -p topol.top -o em.tpr
gmx mdrun -v -deffnm em
```

6. Equilibration and Production

```
gmx grompp -f nvt.mdp -c em.gro -p topol.top -o nvt.tpr
gmx mdrun -deffnm nvt

gmx grompp -f npt.mdp -c nvt.gro -p topol.top -o npt.tpr
gmx mdrun -deffnm npt

gmx grompp -f md.mdp -c npt.gro -p topol.top -o md.tpr
gmx mdrun -deffnm md
```

8.4 Analysis

Use `gmx energy`, `gmx rms`, `gmx gyrate`, etc.

```
gmx energy -f md.edr -o energy.xvg
```

Plot using `xmgrace`, `matplotlib` or `gnuplot`.

Notes

- All steps require `.mdp` parameter files.
- GROMACS is case-sensitive and file naming consistency is important.
- Output trajectory files: `.trr`, `.xtc`, energy files: `.edr`, logs: `.log`.

Appendix

Helpful Online Resources

- Gaussian Documentation: `https://gaussian.com`
- ORCA Input Library: `https://orcaforum.kofo.mpg.de`
- Dalton Project: `https://daltonproject.org`
- FCclasses GitHub: `https://github.com`
- Turbomole: `https://www.turbomole.org`
- LAMMPS Manual: `https://docs.lammps.org`
- GROMACS: `https://manual.gromacs.org`

Shell and Script Tips

- Use `grep`, `awk`, and `sed` for batch parsing
- Use SLURM or PBS for cluster submissions
- Automate analysis with Python or Bash scripts

Bibliography

- Cramer, C. J. *Essentials of Computational Chemistry.* Wiley.
- Jensen, F. *Introduction to Computational Chemistry.*
- Szabo, A., and Ostlund, N. *Modern Quantum Chemistry.*
- Gaussian 16 User Reference.
- ORCA Manual, Frank Neese, Max-Planck Institut.
- GROMACS and LAMMPS user guides.
- Tutorials and internal notes by Prof. Juarez C. Rolim.