Manual for simulating synthetic proteins in Gromacs

Jonas Båtnes

August 17, 2025

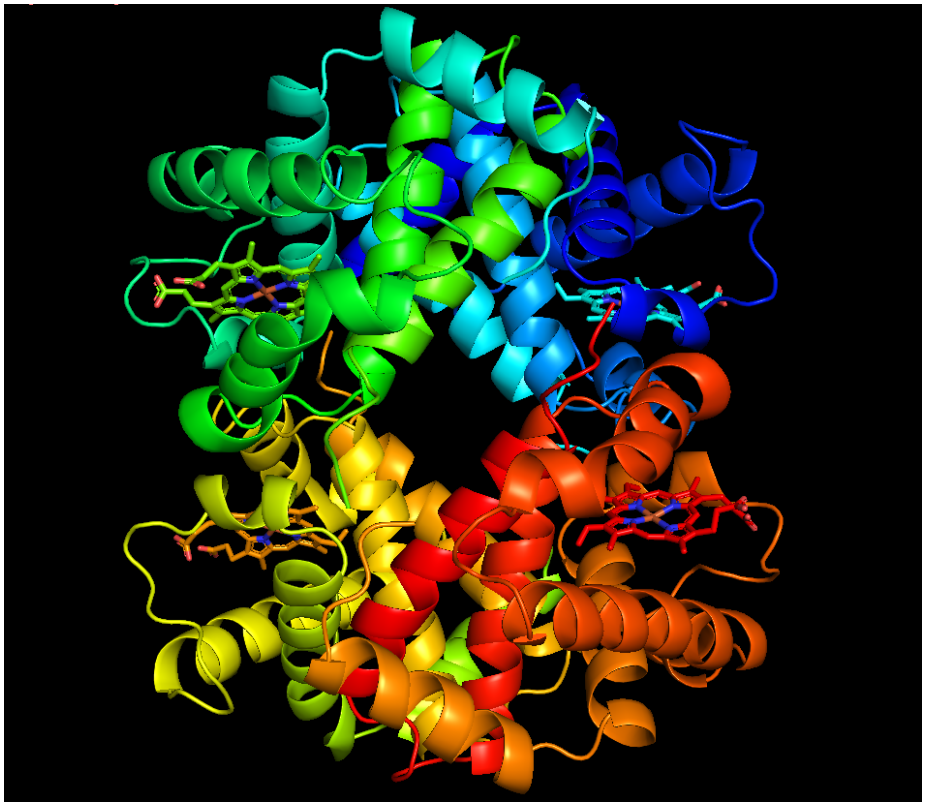

Table of Contents

Requirements

The manual are developed and tested for Gromacs version 2025.2 specifically with the following requirements:

- gcc version 11.4
- cmake version 3.28.3
- NVIDIA HPC SDK version 25.1
- PyMOL version 3.1

However, the manual can be used for all versions. The above requirements only need to be updated accordingly. It should also be mentioned that the manual are created with the assumption that the programs are running on the "greatfacet" node on the CCSE cluster. Pymol are the only program that must be run in a laptop as supercomputers does not have a Graphical User Interphase.

Protein Database

Before any simulation can be done, it is imperative to have a coordinate file for the system. For proteins the easiest way to obtain such a file is through downloading it from Protein Data Bank (PDB), which contain experimentally determined 3D structures of proteins.
As an example, try typing "Hemoglobin" in the search bar. The result should be a list of several hemoglobin quaternary structures and subunits from different organisms. The left menu "Refinements" is an advanced filter if specific requirements are needed. The most relevant options for human studies are to choose "Homo Sapiens", "Sus scrofa" (Wild Boar) or "Drosophila melanogaster" (Fruit fly) for organism and "X-RAY DIFFRACTION" or "ELECTRON MICROSCOPY" as experimental method. All depends on what structures are available. After options are chosen, the green play button on the right side of "Refinements" must be clicked.
After searching for "Hemoglobin", on top there should be a structure with ligand ID: 2PGH. This ID is important as it is a scientific reference ID. It can be used in the search bar to find the structure directly and to easily let other know what structure are used for simulations. When clicking on the ID: 2PGH, a new window with more info appears. More accurate experimental data are available when scrolling down, where sequence length and Reference sequence of the individual proteins/chains often are most important. The Reference sequence will be used to add missing aminoacids in PyMOL. To download the structure, there are a scroll down menu that says "Download Files" in the top right corner. Choose "Legacy PDB Format" or "PDBx/mmCIF Format" if the first is not available due to structure size.
The downloaded file must be uploaded to PyMOL which is free for students, such that the structure can be refined for your own purposes.

Pymol

When opening PyMOL the file of study can be chosen via File/Open... in the top left menu. From here it is possible to do a range of modifications to prepare for simulation. First we can start by displaying the sequence via Display/Sequence and choosing the standard aminoacid 3 letter codes via Display/Sequence Mode/Residue Names. Now, the aminoacid of choice can be selected by clicking on them either one by one or by clicking on one and then use SHIFT + click to select all the aminoacids in between. Try to do this with the water molecules with residue numbers 142-162. Then go to the menu on the right and click on the A (Action) dropdown menu on the "(sele)" entry. By clicking "remove atoms" the selected water molecules are deleted. We are lucky that for water specifically we can click A/remove waters from the "All" entry without selecting any of the water molecules as they will be added later in Gromacs. The most common amendment that we must do in PyMOL besides removing water is adding a missing aminoacids as experimental uncertainties are present in the uploaded file. They appear with grey color in the displayed sequence. In our Hemoglobin sequence we have no missing aminoacids, but for practice we can remove aminoacids 6-8 (ASP-LYS-ALA) as already learned. Before or after the removal, click on the S (Show) dropdown menu from the "All" entry and then on "licorice". Then click H (Hide) and "cartoon". This makes it easier to see the individual aminoacids and atoms. Then click A/zoom on the "sele" entry (if the 3 aminoacids are removed, select the 2 neighboring aminoacids). It is good practice to orient the structure such that the righthand neighbor of the missing aminoacids are on the right in the viewer. Now click "Builder..." above the right hand menu. This open a new window where we are going to use the "protein" section. First we have to select the secondary structure that our missing aminoacids has. We know from the cartoon view we have hidden, that they are part of an Alpha Helix, so select this in the middle dropdown menu. Now to adding the missing aminoacids, we can use the Reference Sequence in PDB as described in the previous section and map them to the correct aminoacid using Figure 1. Just click the aminoacid to add in the builder window, and then click the correct site that PyMOL suggest with spheres. When the last aminoacid are added, there should be a short distance between the carboxylic (COOH) carbon and the amine (NH2) Nitrogen. These can be connected by clicking/selecting them in the viewer and then choosing a single bond in the builder menu. If the distance between the aminoacids are not close, double check that the correct secondary structure are added. If there are long sequences that are missing from the structure, the 3D orientation can be approximated with help of images from Alpha Fold. Just be certain that to find the same protein from the same organism with the same sequence length as the structure used from PBE. Saving the structure final step and are done by clicking File/Save Session to save the structure. After that, go to File/Export Structure.../Export Molecule.... A new window will open, where a selection of atoms to export must be made. The "enabled" alternative is choosing all the entries in the right menu that has a green dot at the start. Click on the green dot to deselect it. The State

should always be "-1 (current)" and the only boxes that need to be checked off is "Write segment identifier (segi) column" under PDB Options and "one single file" under Multi-File. Finally, click Save.... To learn more advanced features of PyMOL, watch the "PyMOL 101" series from Molecular Memory on Youtube.

Amino acid	Three letter symbol	One letter symbol*
Alanine	Ala	A
Arginine	Arg	R
Asparagine	Asn	N
Aspartic acid	Asp	D
Cysteine	Cys	C
Glutamic acid	Glu	E
Glutamine	Gln	Q
Glycine	Gly	G
Histidine	His	H
Isoleucine	Ile	I
Leucine	Leu	L
Lysine	Lys	K
Methionine	Met	M
Phenylalanine	Phe	F
Proline	Pro	P
Serine	Ser	S
Threonine	Thr	T
Tryptophan	Trp	W
Tyrosine	Tyr	Y
Valine	Val	V

Figure 1: Translation table for Aminoacids.

After saving the pdb file, open it and remove all lines that do not start with "ATOM", "HETATM" or "TER" including the bottom CONNECT lines (Gromacs has inbuild connect features). If the structure is bigger than 99999 atoms it is necessary to do some modifications to the file. Specifically, the "atom numbers" must start from 1 again after 99999. The "chain ID" can be a capital or small letter or a digit but if there are more than 62 Chain IDs, we must

switch between 2 or more of the alternatives and add a "secondary Chain ID" in between the "temperature factor" and the "element name" in Figure 2.

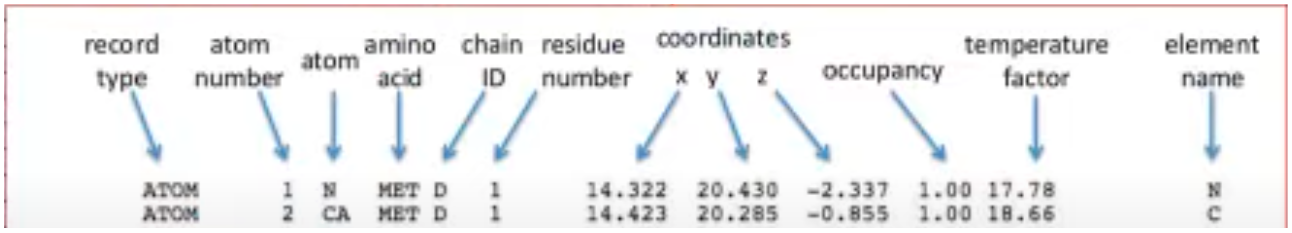

Figure 2: pdb file structure.

The following python file are doing all of the mentioned changes for big pdb files: Fix3column.py. In addition, here is a python file for searching through the residue numbers for finding missing residues and inserting a TER line between the two neighboring residues: FindmissingTERandinsert.py. This can be smart to do for all file sizes in case aminoacids are overlooked in PyMOL. Gromacs will cause errors if it detects a missing residue and no TER line separates the neighboring residues.

Gromacs

Now that the pdb file is ready for simulation we can start with the gromacs commands for doing simulations.

Step 1: Generating Topology and Converting pdb to gro

```
gmx_mpi pdb2gmx -f Finalproteinstructure.pdb -o processed.gro -p topol.
    top -ignh
```

This command adds a force field and a water model of choice. The `-ignh` flag ignores all hydrogens and adds new hydrogens based on the chosen force field and water model. The output format changes from `.pdb` to `.gro` (GROMACS format). This step prevents the protein from interacting with itself across periodic boundaries.

Step 2: Setting Up Periodic Boundary Conditions

```
mpirun -np 1 gmx_mpi editconf -f processed.gro -o box.gro -c -d 2.0 -bt
    triclinic
```

This command handles periodic boundary conditions and box dimensions:

- `-c`: Centers the protein in the box.

- `-d 2.0`: Sets a minimum distance of 2.0 nm between the protein and box edge.
- `-bt triclinic`: Creates a rectangular box (important for periodic boundary conditions). This can be changed to cubic if needed.

Make sure to take a picture of the output, the box center and dimensions will be important for specifying the Quantum Mechanical (QM) box if Cp2k are used.

Step 3: Solvating the System

```
mpirun -np 1 gmx_mpi solvate -cp box.gro -cs tip4p -o solvated.gro -p
    topol.top
```

This command solvates the system with TIP4P water:

- `-cp box.gro`: Specifies the coordinate file.
- `-cs tip4p`: Specifies the TIP4P water model.
- `-o conf.gro`: Specifies the output coordinate file.
- `-p topol.top`: Amends the topology file by changing `SOL 0` to `SOL "number of water molecules"`.

Step 4: Preparing for Ion Addition

```
mpirun -np 1 gmx_mpi grompp -f ../Configfiles/ions.mdp -c solvated.gro -p
     topol.top -r solvated.gro -o ions.tpr -maxwarn 1
```

This command creates an `ions.tpr` file necessary for the `genion` step. The mdp file must be created and can for example look like this ions.mdp. Remove the `-r solvated.gro` if the line `define = -DPOSRES` is removed from `ions.mdp`. This restrains the protein, enabling a two-step minimization where water settles first, then the protein.

Step 5: Adding Ions with Specific Concentrations

```
mpirun -np 1 gmx_mpi make_ndx -f solvated.gro -o index.ndx
```

Run this command only if you want to specifically decide ions and concentration of choice. This creates an index file specifying molecules to replace with ions. When prompted, select "SOL" and then `q` to quit.

```
mpirun -np 1 gmx_mpi genion -s ions.tpr -o neutralized.gro -p topol.top -
    pname NA -nname CL -neutral -n index.ndx
```

Run this command with the index.ndx to neutralize the system before adding specific ions and concentrations. If only neutralizing is necessary, drop the first

command in step 5 and run this command by removing the index.ndx in the end. Select "SOL" when prompted to add `Na+` and `Cl-` to neutralize the system.

```
mpirun -np 1 gmx_mpi genion -s ions.tpr -o solv_with_Na.gro -p topol.top
    -pname NA -np X -n index.ndx
mpirun -np 1 gmx_mpi genion -s ions.tpr -o solv_with_Cl.gro -p topol.top
    -pname CL -conc Y -n index.ndx
mpirun -np 1 gmx_mpi genion -s ions.tpr -o solv_with_K.gro -p topol.top -
    pname K -conc Z -n index.ndx
mpirun -np 1 gmx_mpi genion -s ions.tpr -o solv_with_Mg.gro -p topol.top
    -pname MG -conc U -n index.ndx
mpirun -np 1 gmx_mpi genion -s ions.tpr -o solv_with_Zn.gro -p topol.top
    -pname ZN -conc V -n index.ndx
mpirun -np 1 gmx_mpi genion -s ions.tpr -o solv_with_Ca.gro -p topol.top
    -pname CA -conc W -n index.ndx
```

Examples of ions that can be added are given in the last command above. Gromacs ensures electro-neutrality by adding complementary opposite ions when using `-conc X`. The concentrations should be added as moles (M). If a specific number of ions want to be added, subsitute the `-conc X` flag with `-np X`.

Step 6: Creating Index Files for CP2K (Optional)

If using CP2K:

```
mpirun -np 1 gmx_mpi make_ndx -f neutralized.gro -o temp.ndx
```

This stores the atom numbers of the MM (classical) part of the simulation in the temp.ndx file. Press `q` and enter to quit the prompt.

```
mpirun -np 1 gmx_mpi select -f neutralized.gro -s neutralized.gro -on
    qm_water.ndx -select '"QMatoms" name OW HW1 HW2 and same residue as
    (resname SOL and name OW and x > 16.530 and x < 17.730 and y >
    16.615 and y < 17.815 and z > 6.390 and z < 2.590)'
```

This selects the atoms in a specific QM region based on coordinates from the simulation box in units nm, creating `qm_water.ndx`. Edit `qm_water.ndx` to rename the group to `QMatoms`. A cube with 1.2 nm sides yields approximately 56 water molecules and 1 nm sides yields approximately 21 water molecules.

```
mpirun -np 1 gmx_mpi make_ndx -f neutralized.gro -o index.ndx -n temp.ndx
     qm_water.ndx
```

This command merge `temp.ndx` and `qm_water.ndx` into index.ndx. Press `q` to quit when prompted. Verify in `index.ndx` that `QMatoms` contains only hydrogens and oxygens, with no virtual sites, by mapping atom numbers to `neutralized.gro`.

Step 7: Energy Minimization

If Cp2k are not linked with Gromacs, remove the -n index.ndx flag and the -qmi ../Configfiles/.inp flags for all the following grompp commands. Except these flags, the commands are exactly similar for Gromacs alone, where the Cp2k sections in the mdp file are removed/commented out.

```
mpirun -np 1 gmx_mpi grompp -f ../Configfiles/emcp2k.mdp -c neutralized.
    gro -p topol.top -r neutralized.gro -n index.ndx -o em.tpr -maxwarn
    1 -qmi ../Configfiles/em_cp2k.inp
```

This creates an `em.tpr` file for energy minimization. The emcp2k.mdp file must be created. Remove `-r neutralized.gro` if `define = -DPOSRES` is removed from `emcp2k.mdp`. Remove `-qmi ../Configfiles/em_cp2k.inp` if using a basic functional like PBE and not an input file as specified in the Cp2k section in the mdp.

Run energy minimization:

```
mpirun -np 2 gmx_mpi mdrun -v -deffnm em -cpt 5
```

The -cpt 5 command creates a checkpoint every 5 minutes such that the run can be started again if it is interrupted prematurely. Substitute this flag with -cpi em.cpt to start the simulation again from the last checkpoint.

Extract potential energy:

```
mpirun -np 1 gmx_mpi energy -f em.edr -o potential.xvg
```

Select `13 0` to choose potential energy. Use plotxvgfiles.ipynb to visualize the potential energy.

Step 8: NVT Equilibration

```
mpirun -np 1 gmx_mpi grompp -f ../Configfiles/nvtcp2k.mdp -c em.gro -r em
    .gro -p topol.top -n index.ndx -o nvt.tpr -maxwarn 1 -qmi ../
    Configfiles/nvt_cp2k.inp
```

This creates an `nvt.tpr` file for the first equilibration step (NVT ensemble, stabilizing temperature). The nvtcp2k.mdp file must be created. Remove `-r em.gro` if `define = -DPOSRES` is removed from `nvtcp2k.mdp`. Remove `-qmi ../Configfiles/nvt_cp2k.inp` if using a basic functional like PBE and not an input file as specified in the Cp2k section in the mdp.

Run NVT equilibration:

```
mpirun -np 2 gmx_mpi mdrun -deffnm nvt -cpt 5
```

The -cpt 5 command creates a checkpoint every 5 minutes such that the run can be started again if it is interrupted prematurely. Substitute this flag with -cpi nvt.cpt to start the simulation again from the last checkpoint. To continue to the next step without finishing the whole run, use "gmx_mpi trjconv -s nvt.tpr -f nvt.cpt -o nvt.gro -dump 0" and choose 0 when prompted, to create the necessary nvt.gro

Extract temperature:

```
mpirun -np 1 gmx_mpi energy -f nvt.edr -o temperature.xvg
```

Select `18 0` to choose temperature.

Step 9: NPT Equilibration

```
mpirun -np 1 gmx_mpi grompp -f ../Configfiles/nptcp2k.mdp -c nvt.gro -r
    nvt.gro -t nvt.cpt -p topol.top -n index.ndx -o npt.tpr -qmi ../
    Configfiles/npt_cp2k.inp
```

This creates an `npt.tpr` file for the second equilibration step (NPT ensemble, stabilizing pressure). The nptcp2k.mdp file must be created. Remove `-r nvt.gro` if `define = -DPOSRES` is removed from `nptcp2k.mdp`. Remove `-qmi ../Configfiles/npt_cp2k.inp` if using a basic functional like PBE and not an input file as specified in the Cp2k section in the mdp.

Run NPT equilibration:

```
mpirun -np 2 gmx_mpi mdrun -deffnm npt -cpt 5
```

The -cpt 5 command creates a checkpoint every 5 minutes such that the run can be started again if it is interrupted prematurely. Substitute this flag with -cpi npt.cpt to start the simulation again from the last checkpoint. To continue to the next step without finishing the whole run, use "gmx_mpi trjconv -s npt.tpr -f npt.cpt -o npt.gro -dump 0" and choose 0 when prompted, to create the necessary npt.gro

Extract pressure and density:

```
mpirun -np 1 gmx_mpi energy -f npt.edr -o pressure.xvg
mpirun -np 1 gmx_mpi energy -f npt.edr -o density.xvg
```

Select `20 0` for pressure and `26 0` for density.

Step 10: Molecular Dynamics (MD)

```
mpirun -np 1 gmx_mpi grompp -f ../Configfiles/mdcp2k.mdp -c npt.gro -t
    npt.cpt -p topol.top -n index.ndx -o md_0_1.tpr -qmi ../Configfiles/
    md_cp2k.inp
```

This creates an `md_0_1.tpr` file for the MD step. The mdcp2k.mdp file must be created. Amend `mdcp2k.mdp` to set `nstxout` and `nstvout` to non-zero values for position and velocity output. Remove `-qmi ../Configfiles/md_cp2k.inp` if using a basic functional like PBE and not an input file as specified in the Cp2k section in the mdp.

Run MD:

```
mpirun -np 2 gmx_mpi mdrun -deffnm md_0_1 -cpt 5
```

If the simulation stops prematurely, substitute `-cpt 5` flag with `-cpi md_0_1.cpt` to resume from the last checkpoint.

Extract MD data:

```
mpirun -np 1 gmx_mpi energy -f md_0_1.edr -o md.xvg
```

There are many ways of analyzing the trajectory file (.xtc) from the simulation. Look at the Gromacs homepage to learn the different analysis options.

Setup and Running on Supercomputer

Gromacs

Running simulations on a supercomputer requires specific knowledge of the hardware that will be used. Run "lscpu" in the terminal to find the specification of the node cpu, where AVX version is the most important to notice together with number of cores. Also run "nvidia-smi" to see the available GPUs on the node. The GPU type must be noticed, as it is important for the build process of Gromacs and Cp2k. Download Gromacs from their home page. Source the build environment via the terminal (source hpc_build_env.sh) such that Gromacs can find all the necessary paths, and then create a build directory in the gromacs-2025.2 folder. After that, the following cmake command builds Gromacs on the "greatfacet" node supporting CPUs with AVX2_256 and NVIDIA A100 GPUs:

```
cmake .. \
    -DGMX_BUILD_OWN_FFTW=ON \
    -DREGRESSIONTEST_DOWNLOAD=OFF \
    -DCMAKE_INSTALL_PREFIX=~/gromacs \
    -DBUILD_SHARED_LIBS=OFF \
    -DGMX_PREFER_STATIC_LIBS=ON \
    -DGMXAPI=OFF \
    -DGMX_INSTALL_NBLIB_API=OFF \
    -DGMX_FFT_LIBRARY=mkl \
    -DGMX_DOUBLE=OFF \
    -DGMX_SIMD=AVX2_256 \
    -DGMX_OPENMP=ON \
```

```
-DGMX_MPI=ON \
-DGMX_GPU=CUDA \
-DCUDA_TOOLKIT_ROOT_DIR=/opt/nvidia/hpc_sdk/Linux_x86_64/25.1/cuda \
-DGMX_CUDA_TARGET_SM="80" \
-DGMX_USE_CUFFTMP=ON \
-DcuFFTMp_ROOT=/opt/nvidia/hpc_sdk/Linux_x86_64/25.1/math_libs \
-DCMAKE_C_COMPILER=nvc \
-DCMAKE_CXX_COMPILER=nvc++
```

Notice that it includes the cuFFTMP, which allows Particle-Mesh Ewald (PME) calculations to be done on GPUs, significantly speeding up simulations. After cmake, run these to build and install, also inside the build folder :"

```
make -j32
make check -j32
make install
```

When Gromacs is installed according to these steps, it should be able to run on several GPUs. Use the Slurmjob.sh to specify number of GPUs and number of CPUs per GPU for running Energy minimization, NVT ensambe, NPT ensamble or Molecular Dynamics. Be careful to set numbers of processes -np equal to –ntasks= and –gres=gpu:.

Cp2k

Cp2k should be built with NVIDIA HPC SDK. This means that cuFFTMP, which allows Particle-Mesh Ewald (PME) calculations to be done on GPUs, are included and multi-node runs are possible. However, there are some challenges with optimization of quantum mechanical simulations, which I have not yet solved (invalid pointer() error at runtime). If removing ELPA from the Cp2k download the simulation runs fine, and to download ELPA successfully the following can be added "`-L$CUDA_HOME/lib64 -L/opt/nvidia/hpc_sdk/Linux_x86_64/25.1/math_libs/lib64`" into the LDFLAGS of the configure command in

`/home/users/jonasbaa/WaterPhase/cp2k/tools/toolchain/scripts/stage5/install_elpa.sh`

This indicates that ELPA does not recognize the NVIDIA HPC SDK folder architecture by itself, which most likely are the cause of the invalid pointer() error. However, to build Cp2k, download Cp2k from this github repository. Source the build environment via the terminal (source hpc_build_env.sh) such that Cp2k can find all the necessary paths, and then navigate to cp2k/tools/toolchain. Run the following toolchain command to download all the necessary dependencies:

```
./install_cp2k_toolchain.sh --install-all --with-tblite=no --with-ace=no
--with-gcc=system --math-mode=mkl --with-mkl=/opt/intel/oneapi/mkl/2025.2
--mpi-mode=openmpi --enable-cuda=yes --gpu-ver=A100 --target-cpu=znver2 -j 32
```

Use:

- `./install_cp2k_toolchain.sh --help`

to get an overview of available toolchain arguments if modifications are necessary.

Now, navigate to the cp2k folder. Make sure to clean the build if several builds are done:

- `make distclean ARCH=local_cuda VERSION=psmp`
- `rm -rf ~/WaterPhase/cp2k/obj/ ~/WaterPhase/cp2k/lib/`

Then run these commands for preparation:

- `source /home/users/jonasbaa/WaterPhase/cp2k/tools/toolchain/install/setup`
- `cp /home/users/jonasbaa/WaterPhase/cp2k/tools/toolchain/install/arch/* /home/users/jonasbaa/WaterPhase/cp2k/arch/`

Now, to make Cp2k for CUDA and OpenMPI compatibility run:

- `make -j 32 ARCH=local_cuda VERSION=psmp`

For other options, read the Cp2k INSTALL.md for other versions than psmp. To create a static linker file run:

- `make -j 32 ARCH=local_cuda VERSION=psmp libcp2k`

And if you have patience, check if the build works. Often, it takes shorter time to just build Gromacs over again with linker flags to Cp2k and test it if runs correctly. But here is the command if for some reason it is necessary:

- `make -j 32 ARCH=local_cuda VERSION=psmp test`

After Cp2k is built, it is necessary to build Gromacs over again with the exact steps described earlier, but with the following cmake command (insert your specific path to Cp2k):

```
cmake .. \
  -DGMX_BUILD_OWN_FFTW=ON \
  -DREGRESSIONTEST_DOWNLOAD=OFF \
  -DCMAKE_INSTALL_PREFIX=~/gromacs \
  -DBUILD_SHARED_LIBS=OFF \
  -DGMX_PREFER_STATIC_LIBS=ON \
  -DGMXAPI=OFF \
  -DGMX_INSTALL_NBLIB_API=OFF \
  -DGMX_FFT_LIBRARY=mkl \
  -DGMX_DOUBLE=OFF \
  -DGMX_SIMD=AVX2_256 \
  -DGMX_MPI=ON \
  -DGMX_GPU=CUDA \
  -DGMX_CUDA_TARGET_SM="80" \
  -DGMX_USE_CUFFTMP=ON \
  -DcuFFTMp_ROOT=/opt/nvidia/hpc_sdk/Linux_x86_64/25.1/math_libs \
```

```
-DGMX_CP2K=ON \
-DCMAKE_C_COMPILER=...tools/toolchain/install/openmpi-5.0.8/bin/mpicc \
-DCMAKE_CXX_COMPILER=...tools/toolchain/install/openmpi-5.0.8/bin/mpicxx \
-DCMAKE_Fortran_COMPILER=...tools/toolchain/install/openmpi-5.0.8/bin/mpifort \
-DCP2K_DIR=.../cp2k/lib/local_cuda/psmp \
-DCP2K_LINKER_FLAGS='
  -L/opt/intel/oneapi/mkl/2025.2/lib
  -L/opt/nvidia/hpc_sdk/Linux_x86_64/25.1/math_libs/12.6/targets/x86_64-linux/lib
  -L/opt/nvidia/hpc_sdk/Linux_x86_64/25.1/comm_libs/12.6/hpcx/hpcx-2.21/ucc/lib
  -L/opt/nvidia/hpc_sdk/Linux_x86_64/25.1/comm_libs/12.6/hpcx/hpcx-2.21/ucx/lib
  -L...tools/toolchain/install/COSMA-2.7.0-cuda/lib
  -L...tools/toolchain/install/dbcsr-2.8.0-cuda/lib
  -L...tools/toolchain/install/deepmd-kit-3.1.0/lib
  -L...tools/toolchain/install/dftd4-3.7.0/lib
  -L...tools/toolchain/install/elpa-2024.05.001/nvidia/lib
  -L...tools/toolchain/install/fftw-3.3.10/lib
  -L...tools/toolchain/install/gmp-6.3.0/lib
  -L...tools/toolchain/install/greenX-2.2/lib
  -L...tools/toolchain/install/gsl-2.8/lib
  -L...tools/toolchain/install/hdf5-1.14.6/lib
  -L...tools/toolchain/install/libint-v2.6.0-cp2k-lmax-5/lib
  -L...tools/toolchain/install/libsmeagol-1.2/lib
  -L...tools/toolchain/install/libtorch-2.7.1/lib
  -L...tools/toolchain/install/libvdwxc-0.4.0/lib
  -L...tools/toolchain/install/libvori-220621/lib
  -L...tools/toolchain/install/libxc-7.0.0/lib
  -L...tools/toolchain/install/libxsmm-e0c4a2389afba36c453233ad7de07bd92c715bec/lib
  -L...tools/toolchain/install/openmpi-5.0.8/lib
  -L...tools/toolchain/install/plumed-2.9.3/lib
  -L...tools/toolchain/install/pugixml-1.15/lib
  -L...tools/toolchain/install/sirius-7.7.0/cuda/lib
  -L...tools/toolchain/install/SpFFT-1.1.1/lib/cuda
  -L...tools/toolchain/install/spglib-2.5.0/lib
  -L...tools/toolchain/install/SpLA-1.6.1/lib/cuda
  -L...tools/toolchain/install/trexio-2.5.0/lib
  -Wl,-rpath,...tools/toolchain/install/COSMA-2.7.0-cuda/lib
  -Wl,-rpath,...tools/toolchain/install/dbcsr-2.8.0-cuda/lib
  -Wl,-rpath,...tools/toolchain/install/deepmd-kit-3.1.0/lib
  -Wl,-rpath,...tools/toolchain/install/dftd4-3.7.0/lib
  -Wl,-rpath,...tools/toolchain/install/elpa-2024.05.001/nvidia/lib
  -Wl,-rpath,...tools/toolchain/install/fftw-3.3.10/lib
  -Wl,-rpath,...tools/toolchain/install/gmp-6.3.0/lib
  -Wl,-rpath,...tools/toolchain/install/greenX-2.2/lib
  -Wl,-rpath,...tools/toolchain/install/gsl-2.8/lib
  -Wl,-rpath,...tools/toolchain/install/hdf5-1.14.6/lib
  -Wl,-rpath,...tools/toolchain/install/libint-v2.6.0-cp2k-lmax-5/lib
```

```
-Wl,-rpath,...tools/toolchain/install/libsmeagol-1.2/lib
-Wl,-rpath,...tools/toolchain/install/libtorch-2.7.1/lib
-Wl,-rpath,...tools/toolchain/install/libvdwxc-0.4.0/lib
-Wl,-rpath,...tools/toolchain/install/libvori-220621/lib
-Wl,-rpath,...tools/toolchain/install/libxc-7.0.0/lib
-Wl,-rpath,...tools/toolchain/install/libxsmm-e0c4a2389afba36c453233ad7de07bd92c715bec/l
-Wl,-rpath,...tools/toolchain/install/openmpi-5.0.8/lib
-Wl,-rpath,...tools/toolchain/install/plumed-2.9.3/lib
-Wl,-rpath,...tools/toolchain/install/pugixml-1.15/lib
-Wl,-rpath,...tools/toolchain/install/sirius-7.7.0/cuda/lib
-Wl,-rpath,...tools/toolchain/install/SpFFT-1.1.1/lib/cuda
-Wl,-rpath,...tools/toolchain/install/spglib-2.5.0/lib
-Wl,-rpath,...tools/toolchain/install/SpLA-1.6.1/lib/cuda
-Wl,-rpath,...tools/toolchain/install/trexio-2.5.0/lib
-Wl,--start-group -lsirius_cxx -lsirius -Wl,--end-group
-lplumed -lplumedKernel -lplumedWrapper
-ldbcsr -ldeepmd_c -ldftd4 -lmctc-lib -lmstore -lmulticharge
-lsmeagol -lelpa_openmp
-Wl,--start-group -lcosta -lcosma_prefixed_pxgemm -lcosma_pxgemm_cpp -lcosma -lTiled-MM
-lmkl_scalapack_lp64 -lmkl_blacs_openmpi_lp64 -lmkl_gf_lp64
-lmkl_gnu_thread -lmkl_core -lxsmmnoblas -lxsmm -lxsmmf -lxsmmext
-lspfft -lfftw3 -lfftw3_omp -lxcf03 -lxc -lvdwxc -lpugixml
-ltrexio -lspla -lsymspg -lspglib_f08 -lint2 -lvori
-lGXCommon -lgx_ac -lgx_minimax -lgsl -lgslcblas
-lhdf5_fortran -lhdf5_hl_fortran -lhdf5_hl -lhdf5 -lz
-ltorch -ltorch_cpu -lc10 -lcudart -lcublasLt -lcublas -lcusolver
-lgfortran -lgomp -lgmp -lgmpxx -lstdc++ -lm -lpthread -lrt -ldl
-rdynamic -Wl,-Bsymbolic -Wl,--enable-new-dtags
-lmpi_usempif08 -lmpi_mpifh -lmpi -lcuda -lcusolverMp -lcal -lnvrtc -lucc -lucs -lucm'
```