GROMACS

Receptor–Ligand Complex Simulation

Pipeline Specification · Steps 1 – 10
Craft Lab Procedures and Protocols
Authors: Anuj Kurella, Oscar Garza, Eliason Lewis, John Craft

Force Field · GROMOS 54A7 (United-Atom)
Water Model · SPC (Simple Point Charge)
Simulation Engine · GROMACS
Docking Platform · AutoDock Vina

Nomenclature Standard ·
`[ProteinName]-sim[PDB_Code]-[Residue]-qm[7-Digit_ID]-lib[5-Digit_ID]`
e.g. `talin-sim2mwn-WN0D-qm1234567-lib12345`

STEP 1 Prepare the Protein · protein.pdb → protein.gro

The first step translates a static crystallographic receptor file (PDB format) into a GROMACS-compatible coordinate file and a force-field topology. This produces the two files consumed by all downstream steps.

Natural Language Trigger

"Take the protein structure talin-2mwn-str01.pdb and convert it into a GROMACS topology and coordinate file. Apply the SPC water model, ignore existing hydrogens so they are rebuilt geometrically, and name the outputs using the standard nomenclature."

Command

```
gmx pdb2gmx \
  -f talin-2mwn-str01.pdb \
  -o talin-sim2mwn-WN0D-QM1234567-lib12345_protein.gro \
  -p talin-sim2mwn-WN0D-QM1234567-lib12345_topol.top \
  -water spc \
  -ignh
```

Parameter Reference

Parameter	Flag	Description
Input file	-f	Ingests the raw atomic coordinates of the receptor from the designated PDB file.
Coordinate output	-o	Generates the simulation-ready protein structure file (.gro). The _protein.gro suffix is mandatory to distinguish it from ligand and complex files sharing the same prefix.
Topology output	-p	Generates the physics rulebook (.top) defining all bonds, angles, and charges under GROMOS 54A7. A posre.itp position-restraints file is also auto-generated and must not be renamed.
Water model	-water spc	Embeds SPC water model parameters into the topology as a prerequisite for the downstream solvation phase.
Hydrogen rebuild	-ignh	Strips all crystallographic hydrogens and rebuilds them from pure chemical geometry, eliminating steric clashes at simulation start.

Historical Examples

```
gmx pdb2gmx -f talin-3g9w-align-atoms.pdb -o talin-3g9w-NOH.gro -water spc
```

```
gmx pdb2gmx -f talin-6vgu-add-sideH.pdb   -o talin-6vgu-NOH.gro  -water spc
-ignh
gmx pdb2gmx -f talin_2mwn_align_405.pdb   -o talin_2mwn_405.gro  -water spc
-ignh
```

VALIDATION Confirm the output .gro contains the expected number of protein heavy atoms. Record this value — it is required for the atom arithmetic in Step 3.

STEP 2 Prepare the Ligand · ligand.pdb → ligand.gro

This step converts the docked ligand pose from AutoDock PDB format into a GROMACS coordinate file. It does not generate a topology — force-field parameters are supplied later via the pre-generated ITP file referenced in Step 4.

Natural Language Trigger

"Take the ligand structure WN0D-unitedatom-pose2-reordered.pdb and convert it to a GROMACS coordinate file. Name the output using the standard nomenclature for the WN0D ligand residue name, quantum mechanics ID 1234567, and ligand library 12345."

Command

```
gmx editconf \
  -f WN0D-unitedatom-pose2-reordered.pdb \
  -o talin-sim2mwn-WN0D-qm1234567-lib12345_ligand.gro
```

Parameter Reference

Parameter	Flag	Description
Input file	-f	Ingests the docked ligand pose. This file must already contain correctly positioned united-atom hydrogens from the upstream parameterization pipeline.
Coordinate output	-o	Generates the ligand coordinate file (.gro). The _ligand.gro suffix distinguishes it from the protein and complex files sharing the same prefix.

Historical Examples

```
gmx editconf -f KJVC_unitedatom_original_geometry.pdb -o
KJVC_unitedatom_original_geometry.gro
gmx editconf -f J211_unitedatom_original_geometry.pdb -o
J211_unitedatom_original_geometry.gro
gmx editconf -f WN0D-unitedatom-pose2-reordered.pdb   -o
WN0D-unitedatom-pose2-reordered.gro
gmx editconf -f WN0D-6vgu-dock-pose1HA-reorder.pdb    -o
WN0D-unitedatom-pose1-clusterA.gro
```

Template (ZZZZ substitution)

```
# Replace ZZZZ with your 4-character residue code
gmx editconf -f ZZZZ_unitedatom_original_geometry.pdb \
             -o ZZZZ_unitedatom_original_geometry.gro
```

SCOPE editconf performs coordinate conversion only — no chemical interpretation. The ligand PDB must already have correctly placed united-atom hydrogens.

VALIDATION Check line 2 of the output .gro for the atom count. Record this value — it is required for the atom arithmetic in Step 3.

STEP 3 Assemble the Complex · protein.gro + ligand.gro → complex.gro

This step merges the receptor and docked ligand into a single unified coordinate file with a valid atom-count header. The result feeds directly into grompp for energy minimisation and MD setup.

Natural Language Trigger

"Assemble the receptor–ligand complex for the WN0D ligand, QM ID 1234567, library 12345. The protein has 3857 atoms and the ligand has 86 atoms (total: 3943). Append the ligand coordinate block to the protein file and update the atom count header."

Output Filename

```
talin-sim2mwn-WN0D-QM1234567-lib12345_complex.gro
```

Commands

```
# 1. Copy protein .gro to the versioned complex filename
cp talin-sim2mwn-WN0D-QM1234567-lib12345_protein.gro \
   talin-sim2mwn-WN0D-QM1234567-lib12345_complex.gro

# 2. Append ligand atom block (strip 2-line header + box-vector line)
tail -n +3 talin-sim2mwn-WN0D-QM1234567-lib12345_ligand.gro | head -n -1 \
  >> talin-sim2mwn-WN0D-QM1234567-lib12345_complex.gro

# 3. Update atom count on line 2  [3857 + 86 = 3943]
sed -i '2s/.*/3943/' talin-sim2mwn-WN0D-QM1234567-lib12345_complex.gro
```

Atom Count Reference

Protein Atoms	Ligand Atoms	Complex Total	System / Notes
2246	86	2332	3G9W (SYK)
3857	86	3943	6VGU (Talin)
2627	32	2659	SYK ZINC000262720815
2627	35	2662	SYK ZINC000158625452
10211	48	10259	Target 1
10211	72	10283	THI 349

10211	52	10263	THI 349 v2
10211	45	10256	THI 935

Process Mechanics

- Stream edit {sed -i} protein file is edited in place. Copy the file if you want to preserve the original for reference and re-use.
- Ligand coordinate block extraction: strip the 2-line header (title on line 1, atom count integer on line 2) and the final box-vector line using tail -n +3 … | head -n -1.
- Atom count header update: line 2 of every .gro is the total system atom count. An incorrect value causes gmx grompp to abort. Always verify the arithmetic before writing.
  e.g. 3857 (protein) + 86 (ligand) = 3943 (complex)
- Residue name consistency: the ligand residue label in the appended block must exactly match the code substituted in the [molecules] directive in Step 4.

VALIDATION After assembly, count coordinate lines and confirm they equal the integer on line 2. Confirm the ligand residue name in the appended block matches the code to be used in Step 4.

STEP 4 Modify the Topology · Register the Ligand in .top

The topology (.top) is the physics rulebook for the entire system. This step registers the ligand as a chemically active participant — not merely a geometric presence — by substituting the ZZZZ placeholder, including the ligand ITP file, and adding a molecule count entry.

Natural Language Trigger

"Update the GROMACS topology for the talin complex with the WN0D ligand, QM ID 1234567, library 12345. Replace the ZZZZ placeholder with the WN0D residue code, include the ligand force-field ITP file, and set the molecule count entry to 1."

Output Filename

```
talin-sim2mwn-WN0D-QM1234567-lib12345_complex_topol.top
```

Commands

```
# 1. Copy base topology to versioned complex topology
cp talin-sim2mwn-WN0D-QM1234567-lib12345_topol.top \
   talin-sim2mwn-WN0D-QM1234567-lib12345_complex_topol.top

# 2. Substitute all ZZZZ placeholders with the ligand residue code
sed -i 's/ZZZZ/WN0D/g' \
    talin-sim2mwn-WN0D-QM1234567-lib12345_complex_topol.top

# 3. Verify ITP include line is present (before [ system ] directive)
#    Expected: #include "WN0D_GROMACS_G54A7FF_unitedatom.itp"
grep -q 'WN0D_GROMACS_G54A7FF_unitedatom.itp' \
     talin-sim2mwn-WN0D-QM1234567-lib12345_complex_topol.top \
  || echo "WARNING: ITP include missing — insert before [ system ]"

# 4. Verify ligand entry in [ molecules ] directive
grep -q 'WN0D' talin-sim2mwn-WN0D-QM1234567-lib12345_complex_topol.top \
  || echo "WARNING: WN0D not in [ molecules ] — append: WN0D  1"

# 5. Dry-run grompp to validate topology + coordinate consistency
gmx grompp \
  -f minim.mdp \
  -c talin-sim2mwn-WN0D-QM1234567-lib12345_complex.gro \
  -p talin-sim2mwn-WN0D-QM1234567-lib12345_complex_topol.top \
  -o talin-sim2mwn-WN0D-QM1234567-lib12345_validation.tpr \
  -maxwarn 1
```

Template Examples (ZZZZ Substitution)

```
sed -i -e s/ZZZZ/1II5/g syk_4fl2_topol.top
sed -i -e s/ZZZZ/M0C4/g syk_4fl2_topol.top
sed -i -e s/ZZZZ/2JAR/g syk_4fl2_topol.top
sed -i -e s/ZZZZ/N0BD/g syk_4fl2_topol.top
```

Process Mechanics

- Base topology copy: the protein topology from Step 1 is copied to a new versioned complex topology. The original is preserved.
- Placeholder substitution: ZZZZ is substituted globally. Expect a minimum of 2 occurrences — one in the #include path and one in [molecules]. Fewer indicates a malformed template.
- ITP include directive: the ligand force-field ITP must appear after all standard force-field includes and before the [system] directive. Absence causes unknown atom type errors in grompp.
- [molecules] block: the ligand must appear as WN0D 1 after the protein entry, matching the atom ordering in the complex .gro from Step 3. Order mismatch is fatal.
- Force field consistency: the ligand ITP must be parameterised against the same force field used in Step 1 (GROMOS 54A7). Mixing force fields produces undefined energetics.

VALIDATION Run the gmx grompp dry run. A clean exit with ≤ 1 warning is required before proceeding to the solvation phase.

STEP 5 Solvate the System · Define Box & Add Water

This step establishes periodic boundary conditions by centering the complex in a defined simulation box, then fills that box with pre-equilibrated SPC water molecules. Both sub-steps are strict prerequisites for all downstream electrostatic calculations.

Natural Language Trigger

"Define a cubic simulation box around the talin complex with a 1.0 nm buffer, then solvate with SPC water and update the topology. Use the standard nomenclature for the 2MWN system."

Part 1 · Define the Simulation Box

```
gmx editconf \
  -f talin_2mwn_405.gro \
  -o newbox_2mwn.gro \
  -bt cubic \
  -d 1.0
```

Parameter	Flag	Description
Input file	-f	The assembled complex coordinate file from Step 3.
Output file	-o	New coordinate file with the box geometry embedded.
Box type	-bt cubic	Specifies a cubic box geometry. Cubic boxes are standard for globular proteins; dodecahedral boxes may be used to reduce solvent volume.
Buffer distance	-d 1.0	Enforces a minimum 1.0 nm gap between any periodic image of the solute, preventing unphysical self-interactions across the boundary.

Part 2 · Solvate with SPC Water

```
gmx solvate \
  -cp newbox_2mwn.gro \
  -cs spc216.gro \
  -p talin_2mwn_topol.top \
  -o solv_2mwn.gro
```

Parameter	Flag	Description
Complex input	-cp	The boxed coordinate file produced by Part 1.

Solvent config	-cs spc216.gro	The GROMACS pre-equilibrated SPC water configuration library file. Must match the water model declared in the topology (-water spc in Step 1).
Topology	-p	The topology file is updated in-place: solvate appends the SOL molecule count automatically. No manual editing is required.
Output file	-o	Fully solvated coordinate file. This is the input for Step 6.

Process Mechanics

- Box sizing: the 1.0 nm buffer is a minimum. For highly charged or flexible systems, 1.2–1.5 nm may be appropriate to avoid periodic image artifacts.
- Topology auto-update: gmx solvate modifies the [molecules] block of the .top file, appending SOL <N> where N is the number of added water molecules. This count is used in Step 6 for ion replacement.
- SPC consistency: the spc216.gro solvent library must match the -water spc flag declared during pdb2gmx in Step 1. Mismatched water models corrupt the topology.

VALIDATION Open the topology after solvation and confirm a SOL entry has been appended to [molecules] with a non-zero count. Record this count — it will be needed to verify ion replacement in Step 6.

STEP 6 Neutralise the System · Add Counter-Ions

This step evaluates the net electrostatic charge of the solvated system and achieves electrical neutrality by replacing selected solvent molecules with sodium (NA) or chloride (CL) counter-ions. A net-neutral cell is mandatory for accurate long-range electrostatics in all subsequent MD phases.

Natural Language Trigger

"Preprocess the solvated talin 2MWN system to determine its net charge, then add the appropriate counter-ions to neutralise it. Replace solvent molecules in Group 15 (SOL) with NA or CL as needed."

Part 1 · Preprocess & Read Net Charge

```
gmx grompp \
  -f em.mdp \
  -c solv_2mwn.gro \
  -p talin_2mwn_topol.top \
  -o ions.tpr \
  -maxwarn 4

# READ the terminal output carefully.
# Look for a line such as:
#   System has non-zero total charge: -8.0
# Record the sign and magnitude for Part 2.
```

Part 2 · Add Counter-Ions

```
# If system charge is NEGATIVE → add NA (sodium) with -np
gmx genion \
  -s ions.tpr \
  -o newsolv_ions.gro \
  -p talin_2mwn_topol.top \
  -pname NA \
  -nname CL \
  -np X          # X = number of NA ions required

# If system charge is POSITIVE → add CL (chloride) with -nn
gmx genion \
  -s ions.tpr \
  -o newsolv_ions.gro \
  -p talin_2mwn_topol.top \
  -pname NA \
  -nname CL \
  -nn X          # X = number of CL ions required

# When prompted, select Group 15 (SOL) as the replacement group.
```

Parameter	Flag	Description
Input TPR	-s ions.tpr	Binary topology produced by Part 1. Contains the preprocessed system with charge information.
Output file	-o	New coordinate file with water molecules replaced by ions.
Topology	-p	Updated in-place: ion entries are added to [molecules] and SOL count is decremented accordingly.
Positive ion	-pname NA	Ion species used for positive counter-ions (sodium).
Negative ion	-nname CL	Ion species used for negative counter-ions (chloride).
Add NA ions	-np X	Number of sodium ions to add. Use when the system net charge is negative.
Add CL ions	-nn X	Number of chloride ions to add. Use when the system net charge is positive.

Ion Selection Logic

NET CHARGE −8 → add 8 sodium ions → use -np 8
NET CHARGE +5 → add 5 chloride ions → use -nn 5
NET CHARGE 0 → system already neutral; proceed to Step 7 without genion.

Process Mechanics

- Group selection: when prompted interactively, always select Group 15 (SOL). Replacing protein or ligand atoms with ions is a fatal error.
- Topology auto-update: genion decrements the SOL count and appends NA and/or CL entries to [molecules]. Verify these changes after the run.
- Ion count arithmetic: the number of ions added must exactly equal the absolute value of the system net charge to achieve neutrality.

VALIDATION Re-run gmx grompp on newsolv_ions.gro after neutralisation. The preprocessor output should report System has total charge: 0 (or a value within floating-point rounding, < ±0.001 e).

STEP 7 Energy Minimisation · Relax Atomic Geometry

This step removes steric clashes and unphysical geometric strains introduced during solvation and ion placement by relaxing the system to a local potential energy minimum. All subsequent dynamic phases require a strain-free starting structure.

Natural Language Trigger

"Compile the neutralised talin 2MWN system with energy minimisation parameters and submit the minimisation job to the HPC cluster via SLURM."

Part 1 · Compile Binary Run Input

```
gmx grompp \
  -f em_real.mdp \
  -c newsolv_ions.gro \
  -p talin_2mwn_topol.top \
  -o em.tpr \
  -maxwarn 4
```

Part 2 · Submit to HPC Cluster

```
sbatch THem.sh

# Monitor job status:
squeue -u $USER

# Output on completion:
#   em.gro   — energy-minimised coordinate file (input for Step 8)
#   em.edr   — energy data file (inspect with gmx energy)
#   em.log   — full minimisation log
```

Parameter	Flag	Description
MDP file	-f em_real.mdp	Energy minimisation parameter file. Typically specifies steepest-descent or L-BFGS algorithm with a convergence criterion on the maximum force (Fmax).
Coordinates	-c	Neutralised, ion-containing coordinate file from Step 6.
Topology	-p	The complex topology file carrying all bonded and non-bonded parameters.
Output TPR	-o em.tpr	Output of the binary simulation file used as input for gromacs submission.

Max warnings	-maxwarn 4	Permits up to 4 non-fatal warnings from grompp. Review all warnings in the output before proceeding.
SLURM submit	sbatch THem.sh	Submits the minimisation job to the HPC cluster. The script references em.tpr and defines resource allocation (nodes, CPUs, walltime).

Process Mechanics

- Minimisation convergence: the run terminates when the maximum force on any atom falls below the Fmax threshold defined in em_real.mdp (typically 1000 kJ mol^{-1} nm^{-1} for a first pass).
- Output em.gro is the sole input for Step 8. Do not proceed if the minimisation log reports failure to converge — diagnose and resolve clashes first.
- Energy inspection: after completion, run gmx energy -f em.edr and plot the potential energy. It must be negative and stable (not drifting upward).

VALIDATION Confirm the minimisation log ends with Potential Energy showing a large negative value and that Fmax is below the threshold. Any crash or divergence must be resolved before Step 8.

STEP 8 NVT Equilibration · Thermal Stabilization

This step heats the energy-minimized system to the target simulation temperature under constant volume (NVT ensemble), distributing kinetic energy evenly across all atoms. Proper thermal equilibration is a strict prerequisite for pressure equilibration and production MD.

Natural Language Trigger

"Compile the energy-minimised talin 2MWN system with NVT equilibration parameters and submit the thermal equilibration job to the HPC cluster."

Part 1 · Compile Binary Run Input

```
gmx grompp \
  -f nvt.mdp \
  -c em.gro \
  -p talin_2mwn-topol.top \
  -o nvt.tpr \
  -maxwarn 4
```

Part 2 · Submit to HPC Cluster

```
sbatch THnvt.sh

# Monitor job status:
squeue -u $USER

# Output on completion:
#   nvt.gro   — thermally equilibrated coordinates (input for Step 9)
#   nvt.edr   — energy file (inspect temperature via gmx energy)
#   nvt.log   — full equilibration log
```

Parameter	Flag	Description
MDP file	-f nvt.mdp	NVT equilibration parameter file. Defines thermostat (V-rescale or Nosé-Hoover), target temperature, and equilibration duration (typically 100 ps).
Coordinates	-c em.gro	Energy-minimized coordinate file from Step 7. This is the sole structural input.
Topology	-p	Complex topology unchanged from prior steps.
Output TPR	-o nvt.tpr	Output of the binary simulation file used as input for gromacs submission.

SLURM submit	sbatch THnvt.sh	Submits the NVT job. The script references nvt.tpr and handles GPU/CPU resource allocation and walltime.

Process Mechanics

- Ensemble: NVT fixes particle count (N), volume (V), and temperature (T). Volume is held constant intentionally — premature pressure coupling before thermal equilibration causes instability.
- Thermostat: the V-rescale or Nosé-Hoover thermostat in nvt.mdp controls temperature. Confirm the target temperature (e.g. 300 K) matches the intended simulation conditions.
- Temperature convergence: after the run, use gmx energy -f nvt.edr and select Temperature. The value must plateau at the target temperature with low fluctuation before Step 9 is started.
- Position restraints: nvt.mdp typically applies position restraints to the protein backbone (referencing posre.itp from Step 1) to allow solvent to equilibrate around a fixed solute.

VALIDATION Plot the temperature time series from nvt.edr. It must reach and hold the target temperature (e.g. 300 K ± 5 K) for the final third of the run before proceeding.

STEP 9 Production MD · Multi-Time segment Trajectory Execution

This step executes the production molecular dynamics simulation as a chained sequence of discrete trajectory windows (time segments), capturing the real-time structural evolution and intermolecular interactions of the complex over an extended timescale. The multi-time segment approach accommodates HPC execution limits while preserving thermodynamic continuity.

Natural Language Trigger

"Update all MD submission scripts with the PWTE residue name substitution of ZZZZ in preparation to submit each production MD time segment sequentially. Each time segment takes the final coordinate file of the preceding time segment as its starting structure."

Part 1 · Update Submission Scripts

```
# Replace ZZZZ placeholder with the active environment prefix
sed -i -e s/ZZZZ/PWTE/g THmd1.sh
sed -i -e s/ZZZZ/PWTE/g THmd2.sh
sed -i -e s/ZZZZ/PWTE/g THmd3.sh
sed -i -e s/ZZZZ/PWTE/g THmd4.sh
sed -i -e s/ZZZZ/PWTE/g THmd5.sh
```

Part 2 · Sequential Time segment Submission

```
# ── MD TIME SEGMENT 1 ──────────────────────────────────────────
# Input: nvt.gro  (NVT equilibrated structure from Step 8)
gmx grompp -f md.mdp  -c nvt_syk.gro  -p syk_4fl2_topol.top  -o md_syk.tpr
-maxwarn 4
sbatch THmd1.sh

# ── MD TIME SEGMENT 2 ──────────────────────────────────────────
# Input: md_syk.gro  (generated on completion of THmd1.sh)
gmx grompp -f md2.mdp -c md_syk.gro  -p syk_4fl2_topol.top  -o md2_syk.tpr
-maxwarn 4
sbatch THmd2.sh

# ── MD TIME SEGMENT 3 ──────────────────────────────────────────
# Input: md2_syk.gro  (generated on completion of THmd2.sh)
gmx grompp -f md3.mdp -c md2_syk.gro -p syk_4fl2_topol.top  -o md3_syk.tpr
-maxwarn 4
sbatch THmd3.sh

# ── MD TIME SEGMENT 4 ──────────────────────────────────────────
# Input: md3_syk.gro  (generated on completion of THmd3.sh)
gmx grompp -f md4.mdp -c md3_syk.gro -p syk_4fl2_topol.top  -o md4_syk.tpr
-maxwarn 4
sbatch THmd4.sh
```

```
# ── MD TIME SEGMENT 5 ──────────────────────────────────────────
# Input: md4_syk.gro  (generated on completion of THmd4.sh)
gmx grompp -f md5.mdp -c md4_syk.gro -p syk_4fl2_topol.top  -o md5_syk.tpr
-maxwarn 4
sbatch THmd5.sh
```

Time segment Chain — Data Flow

Time segment	MDP File	Input Coordinates	Output TPR	Submit
1	md.mdp	nvt_syk.gro	md_syk.tpr	THmd1.sh
2	md2.mdp	md_syk.gro	md2_syk.tpr	THmd2.sh
3	md3.mdp	md2_syk.gro	md3_syk.tpr	THmd3.sh
4	md4.mdp	md3_syk.gro	md4_syk.tpr	THmd4.sh
5	md5.mdp	md4_syk.gro	md5_syk.tpr	THmd5.sh

Process Mechanics

- Serial dependency: each time segment must complete successfully before the next is compiled. The output .gro of time segment N is the input -c of time segment N+1. Submitting out of order corrupts the trajectory.
- Script placeholder (ZZZZ): the sed substitution propagates the environment/project prefix into all SLURM scripts before any jobs are submitted. Omitting this step causes the scripts to reference incorrect file paths.
- MDP continuity: md2.mdp through md5.mdp should set gen_vel = no and continuation = yes to preserve velocities and avoid re-initializing the thermostat between time segments.
- Walltime management: time segment boundaries are chosen to fit within the cluster's maximum walltime per job. Typical time segment sizes are 1–2 ns per submission.
- Trajectory files: each time segment produces .xtc (compressed trajectory), .trr (full-precision trajectory), .edr (energy), and .gro (final coordinates). Retain all files until analysis is complete.

VALIDATION After each time segment completes, inspect the .log file for Finished mdrun and confirm the output .gro exists before submitting the next time segment. A crashed time segment must be diagnosed and restarted before the chain can continue.

STEP 10 Post-Processing · Strip Solvent & Export to PDB

This step cleans and reformats the raw MD output for downstream analysis and visualization. Explicit solvent and bulk ions are stripped from all trajectory and coordinate files to isolate the solute complex, dramatically reducing file sizes. The final water-free coordinate file is then converted from GROMACS .gro format to the universally recognized PDB format for use in PyMOL, VMD, or other external tools.

Natural Language Trigger

"Strip water and ions from the final talin 2MWN MD trajectory and coordinate files, then convert the cleaned structure to PDB format. Label all outputs with simulation number X."

Part 1 · Strip Solvent from Trajectory & Coordinates

```
# Strip water from compressed trajectory (.xtc)
gmx trjconv -s md5.tpr -f md5.xtc -o syksim0X.xtc

# Strip water from full-precision trajectory (.trr)
gmx trjconv -s md5.tpr -f md5.trr -o syksim0X.trr

# Strip water from final coordinate file (.gro)
gmx trjconv -s md5.tpr -f md5.gro -o syksim0X.gro

# X = simulation number (e.g. syksim01.xtc, syksim02.xtc …)
# When prompted, select Group 1 (Protein) or the appropriate
# solute group to exclude SOL and ions from output.
```

Parameter	Flag	Description
Reference TPR	-s md5.tpr	The binary topology from the final MD time segment. Provides atom group definitions and PBC information for correct trajectory processing. Output of the binary simulation file used as input for gromacs submission.
Input trajectory	-f	The raw output trajectory file (.xtc or .trr) or coordinate file (.gro) from the final MD time segment.
Output file	-o	Cleaned, water-free output. Filename includes the simulation number X for version control.

Part 2 · Convert .gro to .pdb

```
gmx editconf \
  -f syksim0X.gro \
```

```
  -o syksim0X.pdb

# X = simulation number
# Output: universal PDB file ready for PyMOL, VMD, or RCSB submission.
```

Parameter	Flag	Description
Input file	-f syksim0X.gro	Water-stripped final coordinate file from Part 1.
Output file	-o syksim0X.pdb	PDB-format coordinate file. Compatible with all major molecular visualisation and analysis platforms.

Process Mechanics

- Group selection: when trjconv prompts for an output group, select Protein (Group 1) or a custom index group containing the receptor and ligand only. Selecting the System group retains water and defeats the purpose of this step.
- File naming — simulation number X: increment X for each independent simulation run (e.g. 01, 02, 03). This index distinguishes trajectories from different docking poses, replicate runs, or ligand variants.
- Trajectory types: .xtc files are compressed and suitable for most analyses (RMSD, RMSF, clustering). .trr files retain full precision and velocities — required for energy recalculation or entropy estimation.
- PDB format utility: the .pdb output is the entry point for PyMOL visualization, RCSB structural deposition, and ligand interaction fingerprinting tools such as PLIP or ProLIF.
- File size reduction: stripping solvent typically reduces trajectory file sizes by 60–80%, making storage and transfer practical for large multi-time segment simulations.

VALIDATION Open syksim0X.pdb in PyMOL or VMD and confirm only the protein and ligand are present — no water molecules or ions should appear. Verify the ligand residue (e.g. WN0D) is intact and correctly positioned within the binding site.

End of Pipeline Specification · Steps 1 – 10