Technical Tricks of Coarse-Grained MD Visualization with VMD

Olaf Lenz

Institut für Computerphysik, Universität Stuttgart
Stuttgart, Germany

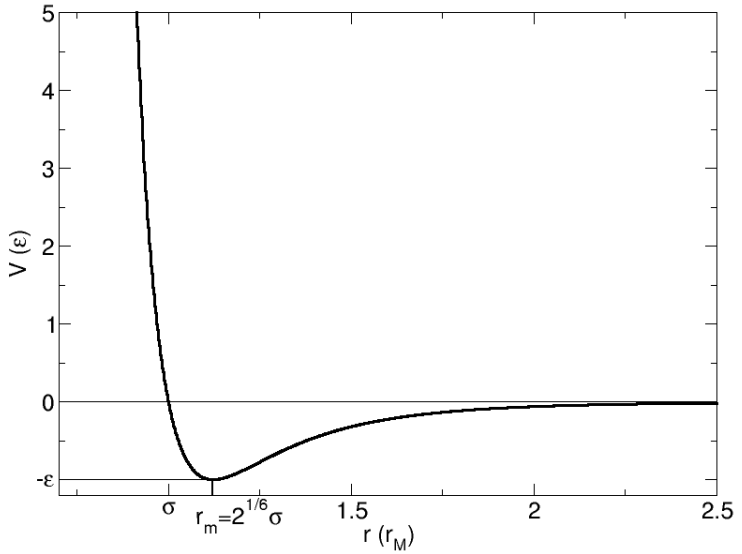
University of Stuttgart
Germany

INSTITUTE FOR COMPUTATIONAL PHYSICS

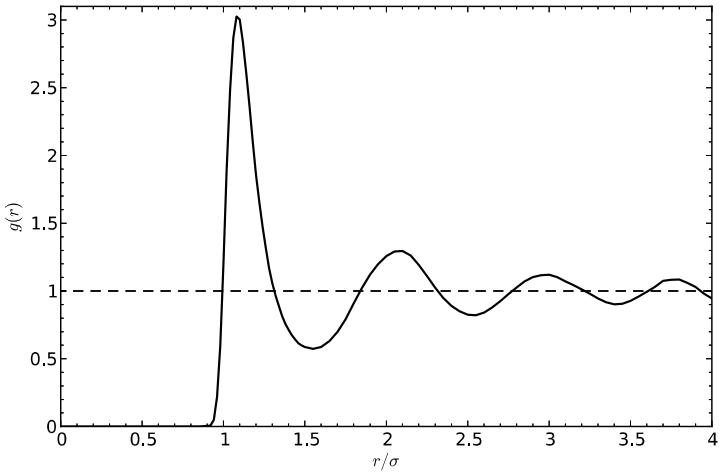
Arbitrary Units

- All-atom MD simulations
  - Lengths in nm
  - Energies in kJ/mol
- Coarse-grained MD
  - No particular units!
  - Units by user choice
- Two independent reference units
  - Use typical length and energy scale
  - Keep quantities in the order of 1
    - Better numerical accuracy
    - Easier to handle
  - e.g. Length unit: LJ parameter σ (i.e. σ=1)
  - e.g. Energy unit: LJ parameter ε (i.e. ε=1)
- Relation Temperature/Energy: $k_B = 1$
- Can be translated to any other unit system

Lennard-Jones-Potential

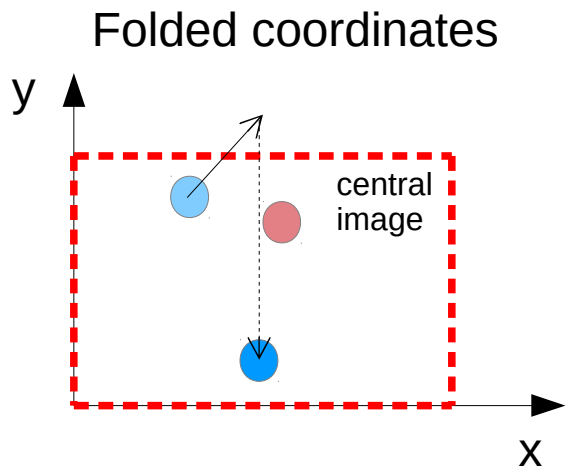


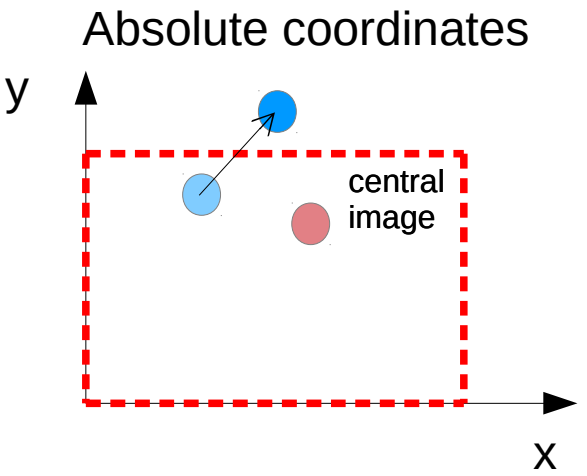
RDF of LJ fluid



Absolute vs. Folded Coordinates

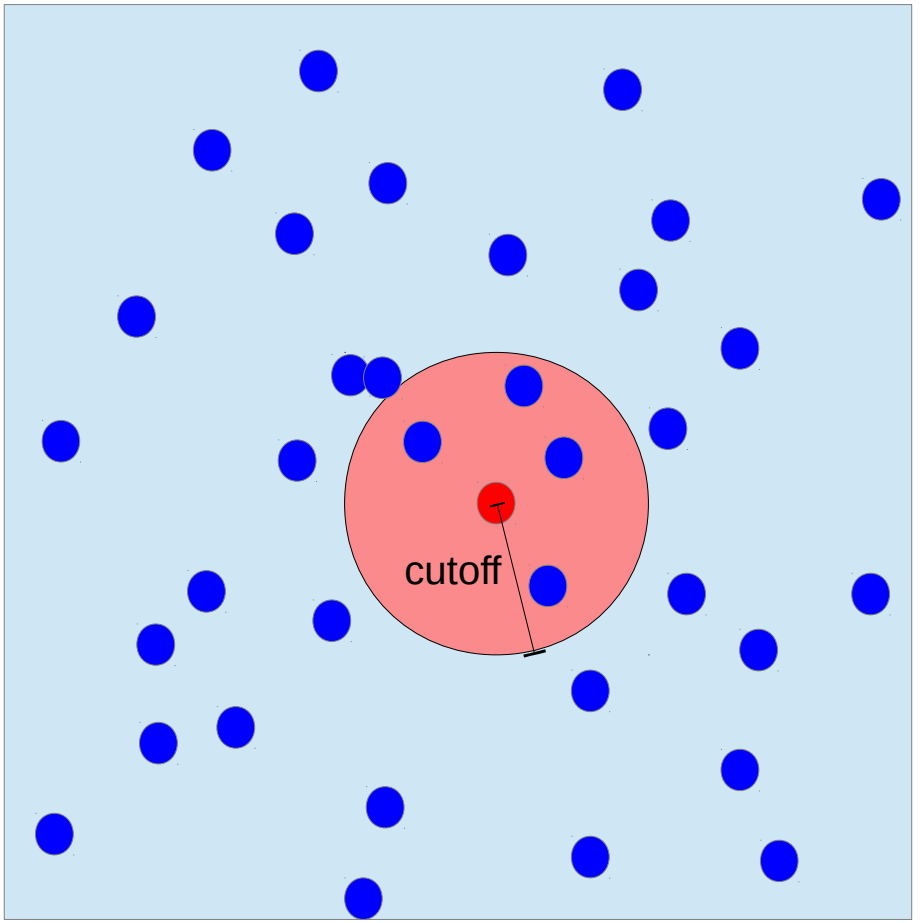
- How to store positions in Periodic Boundary Conditions (PBC)?
- Folded Coordinates
  (a.k.a. “Wrapped Coordinates”)
  - Coordinates are always in central image
  - When a particle leaves the box to any direction, the coordinates are folded
  - Problem: Folding is irreversible
- Absolute Coordinates
  - Coordinates are not folded
  - Coordinates can be outside the central image
  - Coordinates are continuous, no “jumps”
- Computing Distances:
  PBC need to be taken into account in both cases
- Pros of Absolute Coordinates
  - Possible to measure MSD
  - Visualization
    - No overstretched bonds
    - Molecules stay together





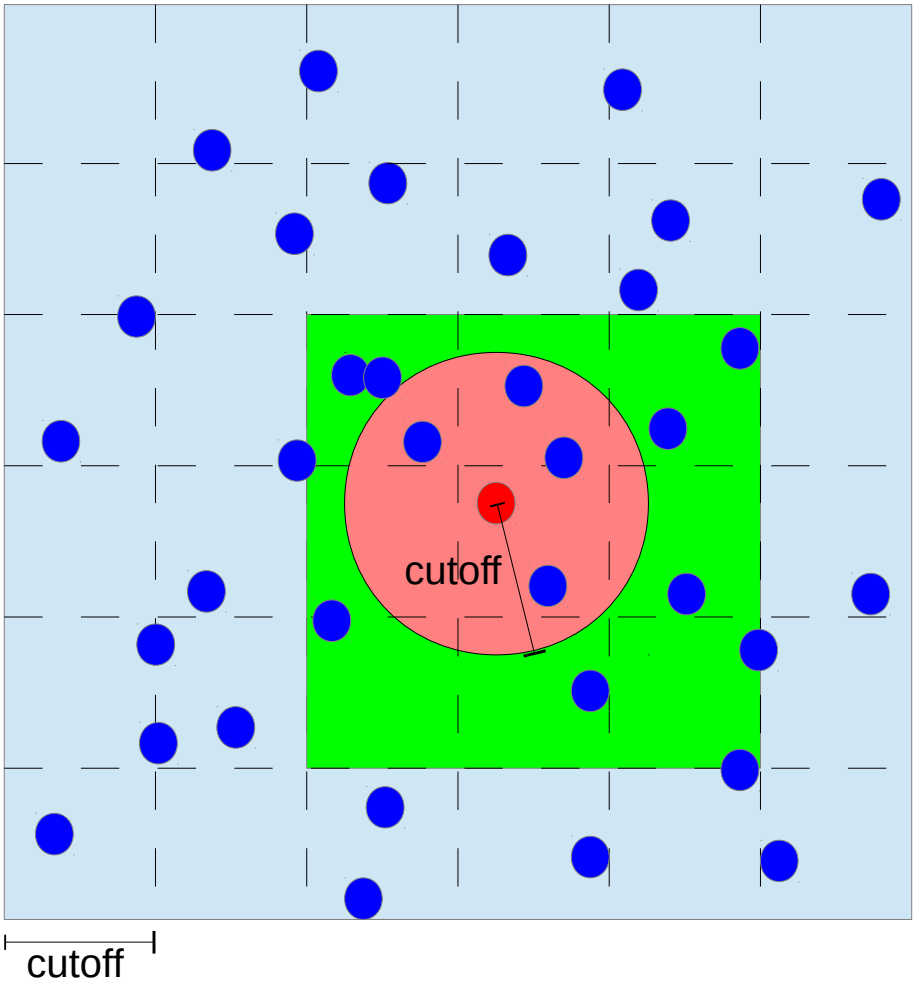
Speeding Up MD

- MD simulation with *N* particles
  - Propagation: *O(N)*
  - Force computation: in theory: $O(N^2)$
- … but many interactions are short-ranged
  - … or at least long tails can be neglected
  - e.g. Lennard-Jones interaction
  - Particle pairs with a distance larger than maximal interaction range (*cutoff*) can be ignored
  - Can be used to improve speed
  - However: only coordinates are known
- Long-ranged interactions are trickier
  - Coulomb interaction → Thursday
  - Hydrodynamics → Friday



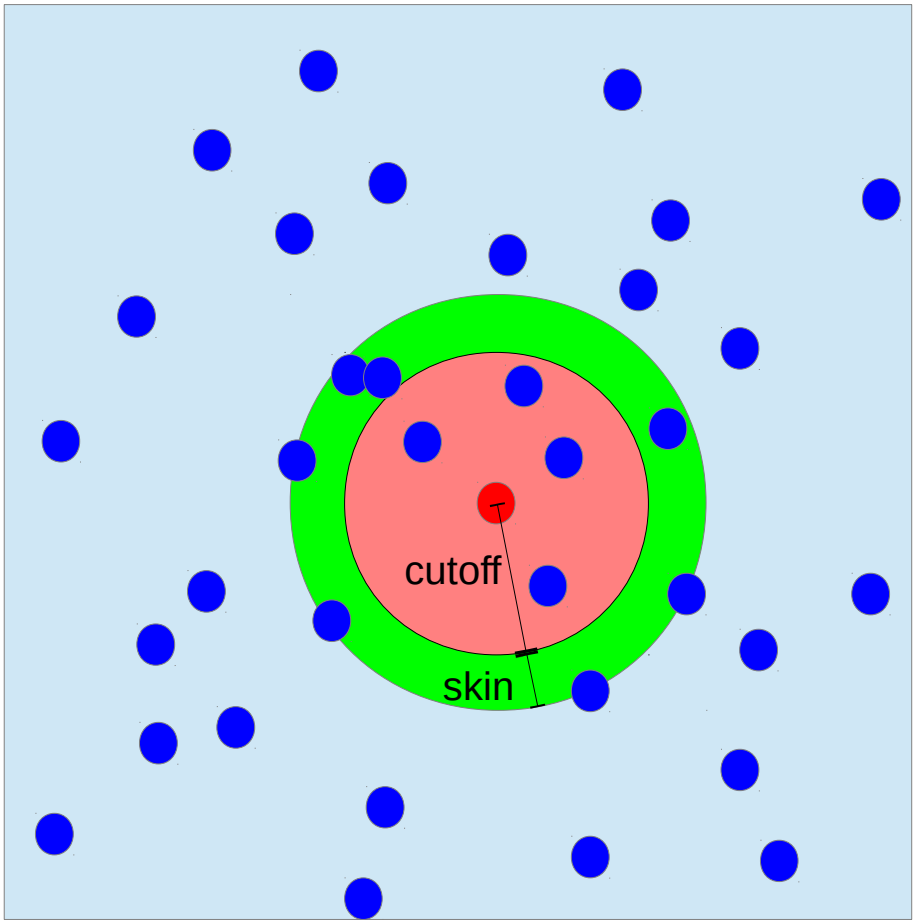
Cell Lists (a.k.a. Linked Cell List)

- Split box into *cells*
- Cell size ≥ *cutoff*
- Store a list of particles per cell
- Interaction partners must all be in neighboring 27 cells
- When a particle moves, move it to new cell
- Reduces complexity to *O(N)* (at constant density)
- Requires at least 3 cells per direction
  - Otherwise physically questionable (particle interacts with its own image)
- Used by ESPResSo/++



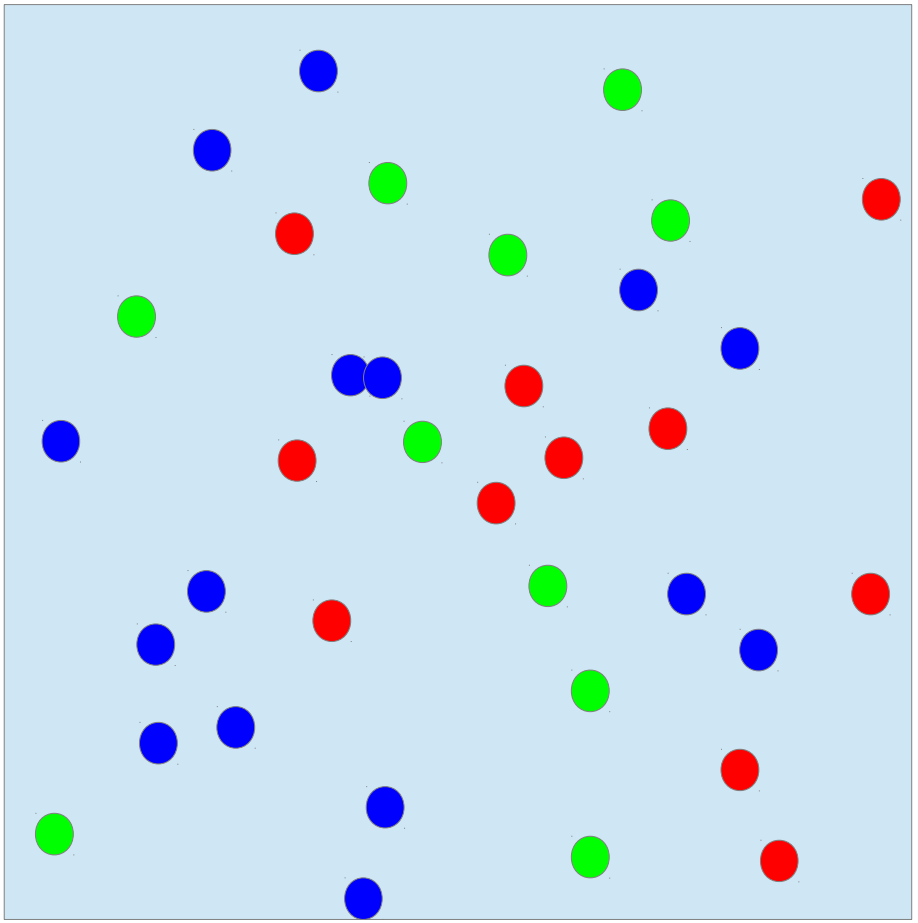
Verlet Lists (a.k.a. Neighborhood List)

- Further improvement
- Compute list of pairs within *cutoff+skin* (e.g. via Cell Lists)
- Store *Verlet lists*
- Compute interaction only for pairs in the list
- Reduces number of interactions to be computed (>60% of Cell lists)
- Verlet lists need to be recomputed when a particle has moved further than ½ *skin ("Verlet update")*
- Skin size: Trade-off
  - Larger skin: more pairs in list
  - Smaller skin: more frequent update
- Used by ESPResSo/++



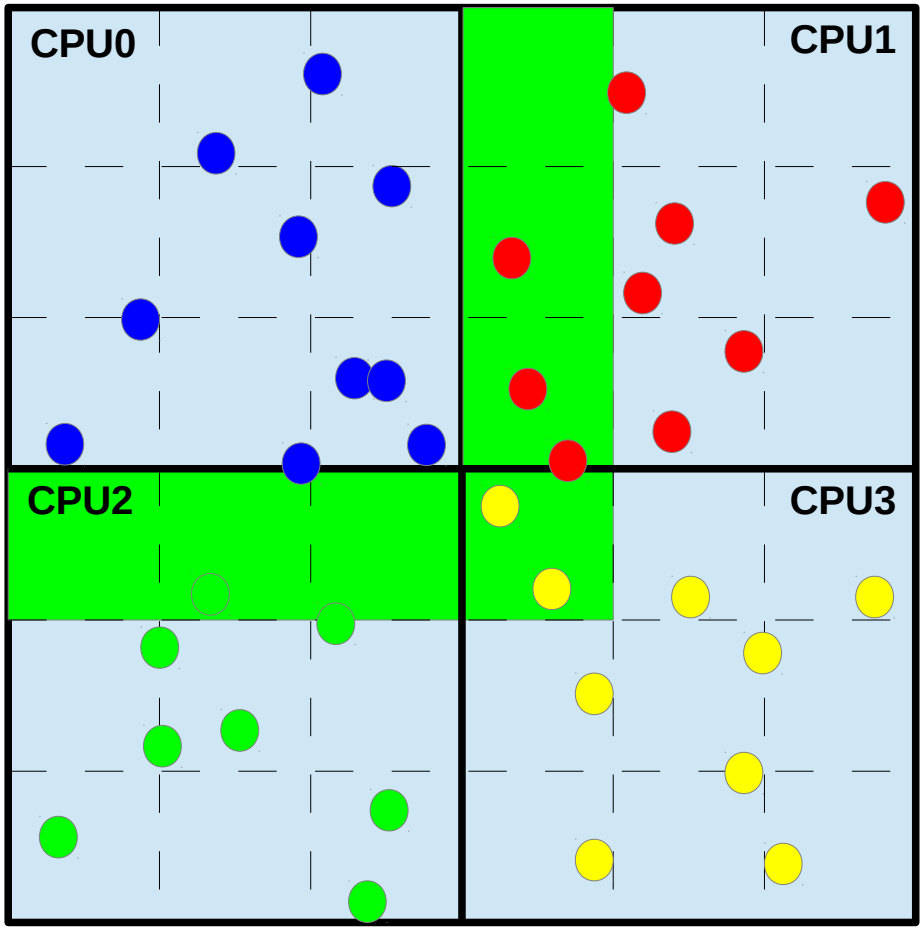
Parallelization: Atomic Decomposition

- How to do MD in parallel with multiple processes (CPUs, cores, nodes, whatever)?
  - Copy all particles to all processes
  - Compute forces of “own” particles and propagate own particles
  - All-to-all communication of particles after each step
- Pros
  - Simple to implement
  - Simple load balancing
- Cons
  - Lots of communication $O(P^2)$
  - Double computation of forces (or even more communication)
  - Bad for lager systems



Parallelization: Domain Decomposition

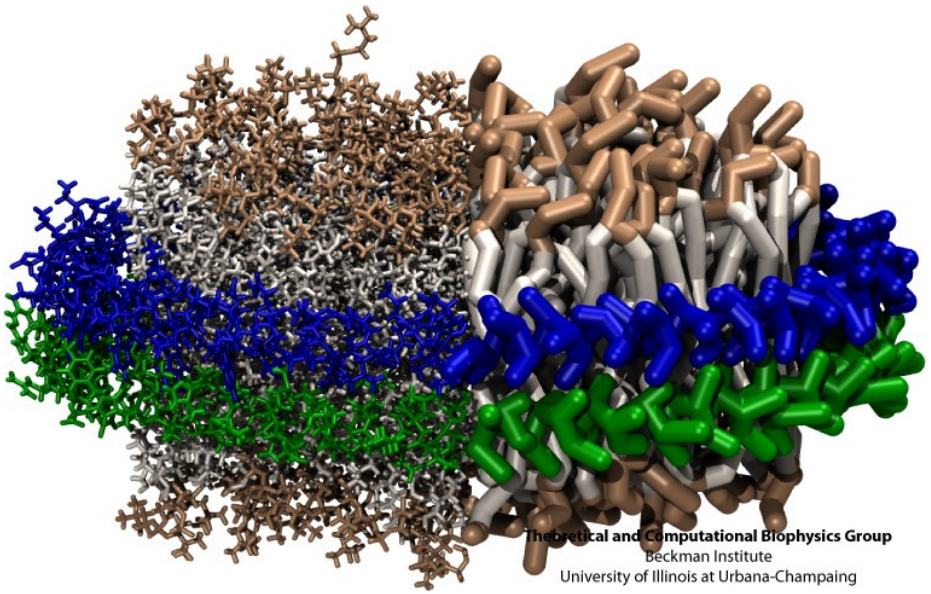
- Decompose system into spatial domains
- Each process
  - is responsible for one *local* domain
  - stores local particles
  - and boundaries of neighboring domains (*ghost particles*)
- Only particles in domain boundaries need to be communicated to neighbor domains *O(P)*
- Pseudo-algorithm
  1. Domain decompose system
  2. Compute forces for all local particles
  3. Propagate local particles
  4. Communicate boundary particles to neighbor process
- Can profit from Cell lists
- Used by ESPResSo/++



Visualization with VMD

- **V**isual **M**olecular **D**ynamics
- Developed at K. Schulten's group in Urbana-Champagne
- Made mostly for proteins, therefore strange vocabulary
- But also very flexible for CG
- Useful for online visualization and publication-quality rendering
- Tcl-scripted (like ESPResSo)
- Free of charge, open source
- … but not FOSS

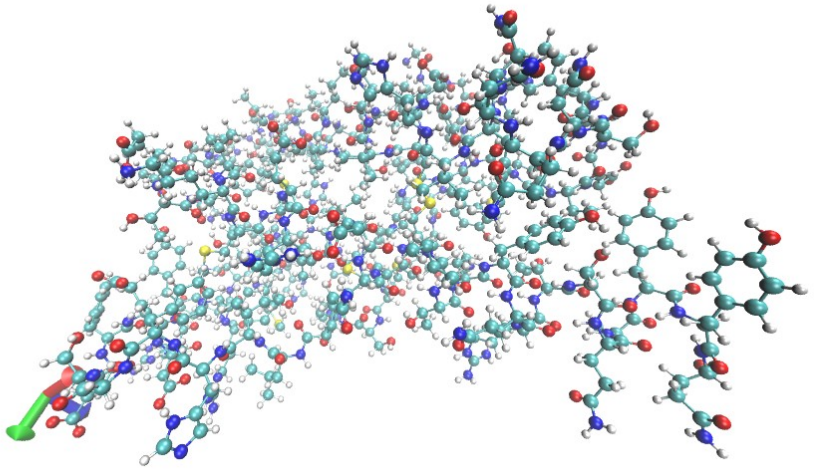
VMD
Visual Molecular Dynamics



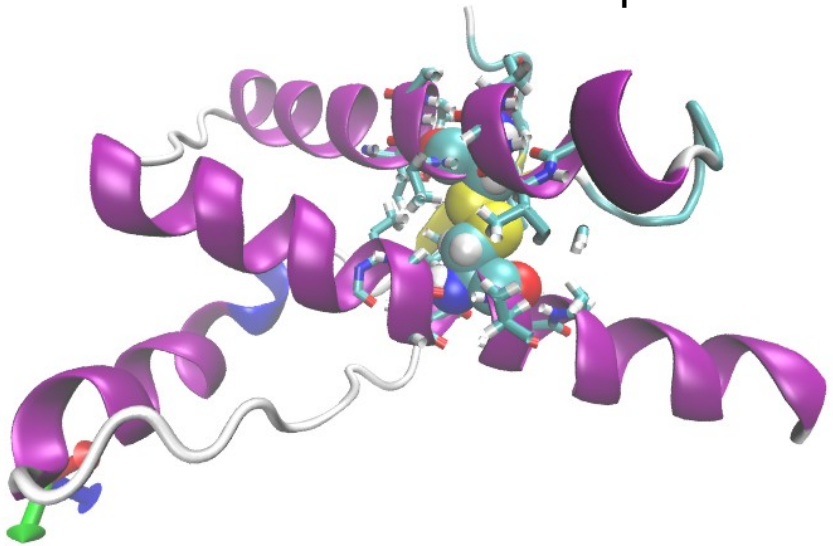
Humphrey, W., Dalke, A. and Schulten, K., "VMD - Visual Molecular Dynamics", J. Molec. Graphics, 1996, vol. 14, pp. 33-38.

Visualization with VMD: Concepts

- Representation
  - Display of a Selection of atoms
  - With a specific Drawing style (e.g. VDW, lines, bonds, …)
  - And a specific Coloring style (e.g. by atom type, by residue, by observable, …)
- Multiple representations allow for fancy visualization
- Most important dialog: Graphics → Representations

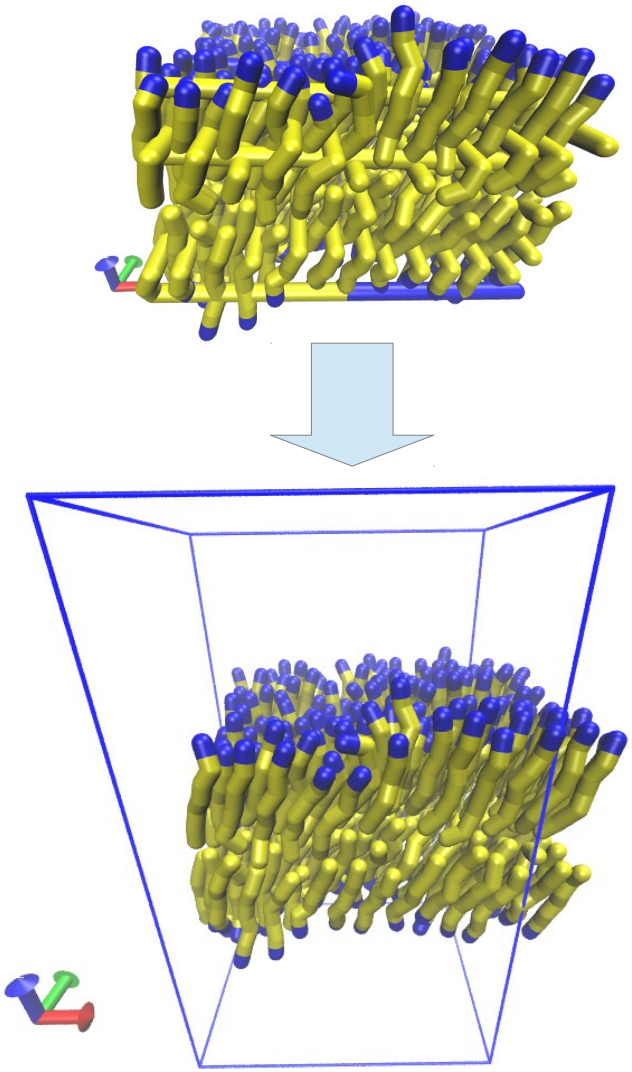


Two visualizations of bovine prion 1DX0



Visualization with VMD: PBCTools

- PBC make visualization complex
  - Overstretched bonds
  - Jumps between frames
  - What is in the center?
- PBCTools can help
  - Draw a box
    `>pbc box`
  - Wrap the coordinates into an image
    `>pbc wrap`
  - “Unwrap” coordinates
    (remove wrapping jumps)
    `>pbc unwrap`
  - Not trivial, as the task is not trivial
  - Tcl commands, no GUI



… and now try it out ...