Review

Molecular Dynamics Simulations: Principles, Algorithms, and Emerging Applications

Stuti , Shivani Yaduvanshi , Dushyant Sharma , Vansh Kashyap , Veerendra Kumar *

Posted Date: 31 March 2026

doi: 10.20944/preprints202603.2311.v1

Keywords: MD simulation; force fields; periodic boundary conditions; protein dynamics; computational biophysics

 doi:10.20944/preprints202603.2311.v1

Review

Molecular Dynamics Simulations: Principles, Algorithms, and Emerging Applications

Stuti, Shivani Yaduvanshi, Dushyant Sharma, Vansh Kashyap and Veerendra Kumar *

Amity Institute of Molecular Medicine and Stem Cell Research (AIMMSCR), Amity University, Noida, Uttar Pradesh, 201303, India

* Correspondence: vkumar34@amity.edu

Abstract

Molecular dynamics (MD) simulation is a fundamental technique for resolving biomolecular structures and functions at atomic resolution. Accelerated by GPU computing and machine learning-integrated force fields (FF), modern MD simulation facilitates the study of large-scale systems and rare biological events, such as protein folding, allosteric transitions etc. While advanced sampling methods and AI integration have significantly enhanced efficiency in drug discovery and protein engineering, the field still faces challenges regarding FF accuracy, timescale constraints, and quantum effects. Continued development of hybrid quantum and molecular mechanics methods and standardized workflows is essential to further improve the predictive power and reproducibility of MD in biotechnological research. In this review, we attempted to provide the latest developments in the MD simulations.

Keywords: MD simulation; force fields; periodic boundary conditions; protein dynamics; computational biophysics

1. Introduction

In biological molecules, the atoms are always moving. They will never reach or remain at a low energy level. The dynamics of atoms determine how biomolecules work. Numerous proteins and enzymes change their shape in order to perform their functions. Consequently, the entire process cannot be explained by a static image. Additionally, complicated processes like protein folding, ligand interactions, protein-protein interactions (PPIs), etc., cannot be fully comprehended by experimental data alone [1–3]. The majority of proteins interact with other proteins to carry out their tasks [4–6]. Protein functions should therefore be characterized in relation to their interacting partners. Actually, PPIs are a crucial target for drug discovery since they are universal biological processes [3,7]. PPIs are characterized using a variety of methods, including far-western blot analysis, pull-down testing, crosslinking, label transfer, and co-immunoprecipitation (co-IP). Biophysical techniques such as co-IP and pull-down experiments make it simple to isolate stable interacting proteins. But to stabilize the connecting proteins, weak or fleeting connections must be crosslinked [8]. To characterize the interactions between proteins, a variety of methods are often needed. Therefore, unless a complicated structure is determined, experimental characterization of PPIs is expensive, time-consuming, and does not provide molecular details.

MD simulation is a computer-based simulation method that examines the physical motions of atoms within molecules by numerically solving Newton's equations of motion. The roles of molecules in their natural environments are expressed by these atomic motions [9]. As a result, MD simulation documents biomolecular processes by accurately simulating what atoms do in real life at femtoseconds (fs, 10-15 s). One may determine the forces applied to each atom by every other atom in a biomolecular system (such as a protein encircled by water), given the locations of every atom (Figure 1A and 1B) [10]. After determining each atom's starting position and velocity, it computes the forces operating on each atom. A "movie" of the system's trajectory is produced by repeatedly

 doi:10.20944/preprints202603.2311.v1

using these forces to forecast the next position and velocity in tiny time steps (Figure 1A). The forces acting on atoms in MD simulations are obtained from molecular mechanics FFs, which are verified against pertinent experimental data and parameterized using quantum mechanical computations [11]. The most costly component of MD simulations is usually force computation. The Boltzmann distribution—that is, the likelihood of seeing a specific atom as a function of potential energy—is sampled by the simulation. Every atom in a typical simulation system is seen as a charged sphere. It is believed that the covalent connections that join atoms act as a spring [10].

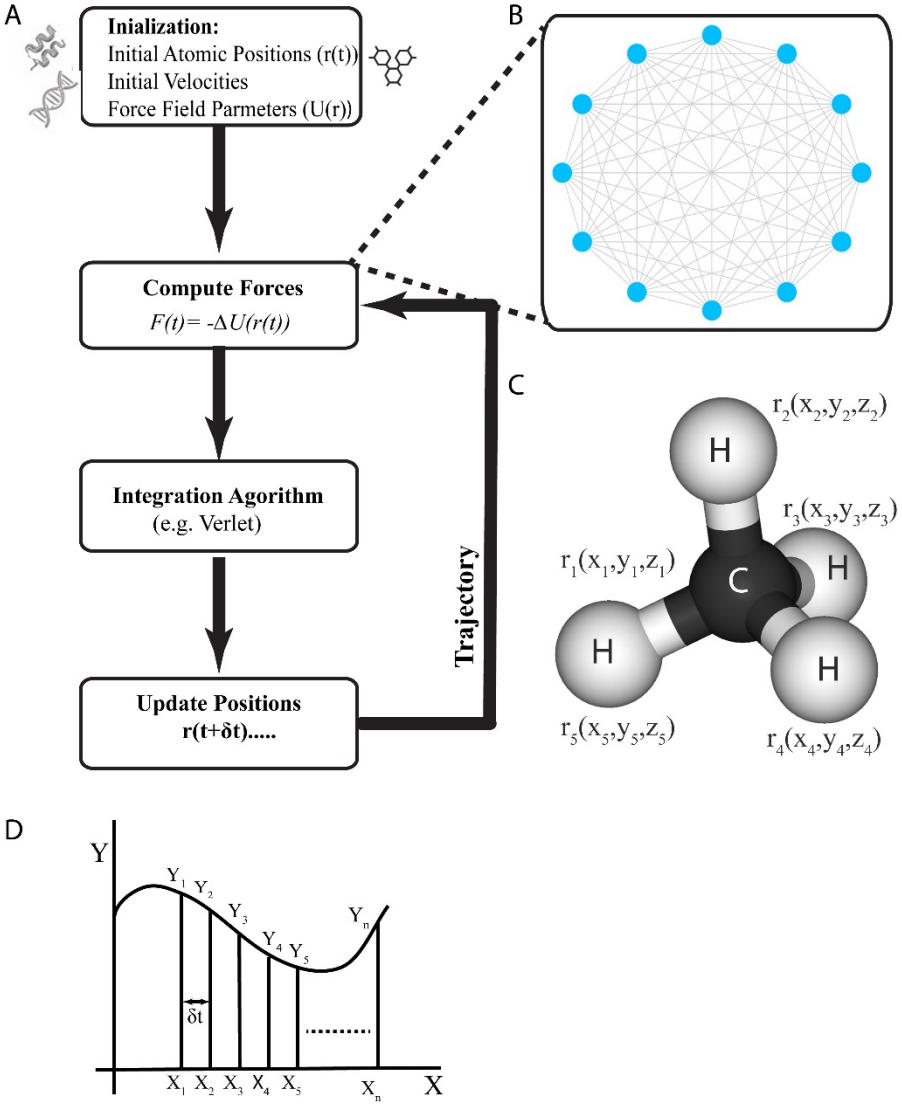


Figure 1. MD simulation algorithm. **A)** MD simulation begins with the initialization step. In this step, the position and velocity of each atom are calculated, and force field parameters are computed. The forces acting on each atom are then used to predict the next position and velocity in small time steps. These steps are repeated to create a "movie" of the system's trajectory. **B)** The molecular geometry R is expressed in terms of atomic positions. **C)** The complexity of interatomic interactions in a system. **D)** Numerical integration methods, e.g., finite difference method (FDM), are employed to approximate the motion of atoms over time. The integration can be divided into many small finite time steps δt. The time step should be small enough to accurately describe the process of interest.

In 1957, the first MD simulation for gases was carried out (Alder and Wainwright, 1957). But in the late 1970s, the first protein MD simulation was carried out [12]. Since its beginnings, MD has evolved into a significant and popular technique that enables researchers to describe the intricate

 doi:10.20944/preprints202603.2311.v1

microscopic dynamical behaviour of a wide range of systems, including gases, liquids, solids, surfaces, and clusters [13] For scientists working in biology, physics, and chemistry, it is now an essential tool. Because MD simulations may access temporal and length scales of microseconds and nanometers, they are able to overcome experimental limitations. For example, the discovery of particular molecular interactions and understanding of how proteins undergo structural changes due to binding are made possible by the comprehensive atomistic depiction and dynamic behaviour inherent in MD simulations. Therefore, biological responses to different alterations, such as mutation, phosphorylation, protonation, or contact with other molecules/materials, can be predicted using MD simulations [10].

For the study of the dynamics and temporal evolution of molecular systems, MD is a fundamental technique in computational design and molecular modelling. Since MD simulations were the first technique to be effectively transferred to the graphical processing unit (GPU) architecture, they have benefited greatly from the rapid increase in computational capacity that has characterized the past decades of computational research [14]. Researchers can now model processes even on the microsecond time scale thanks to the use of GPUs, which increases the possible length of MD simulations.

2. Theory

Newton's laws describe how forces affect atoms' movements. The force applied to every atom by every other atom is computed during simulation (Figure 1B). The spatial positions of individual atoms as a function of time are then predicted using Newton's classical law of motion. The second-order differential equations that Newton's second law expresses are resolved to achieve this:

$$f_i(t) = m_i a_i(t) \tag{1}$$

Where $f_i(t)$ is the net force acting on the ith atom of the system at time t, $a_i(t)$ is the corresponding acceleration, and m_i is the mass of the ith atom [10]. Therefore, by integrating Newton's equations, many configurations of the evolving system are generated. As a result, the trajectory that is produced is a three-dimensional film that shows how each atom in the system is configured at each point during the simulation period. Since analytical solutions to the intricate interactions among many particles are typically unattainable, atomic motion in simulations is integrated using numerical techniques. The nuclear forces in MD simulations cover a wide variety of characteristic time scales. Even slowly changing force components must be integrated at the tiny intervals needed by the quickest motions in a typical MD simulation with a fixed time step, even if these slower forces might theoretically be precisely calculated at larger time discretization [14].

Solving the equations of motion for every atom is a difficult aspect of MD modelling (Figure 1B and 1C). The molecular geometry R of a molecule with N atoms at any given time t can be written as follows in terms of atomic positions:

$$R = (r_1, r_2, \ldots\ldots, r_N)$$

Where r1, r2,…,rN are atomic positions as shown in Figure 1B.

However, acceleration is a second-order derivative of the position r with respect to time t. Therefore, for each atom, the second law can be written as:

$$f_i(t) = m_i a_i(t) = m_i \frac{\partial^2 r_i}{\partial t^2} \tag{2}$$

Furthermore, the conservative forces acting on each atom are equal to the negative gradient of the potential energy V, and can be expressed mathematically as:

$$F_i = -\frac{\partial V}{\partial r_i} \tag{3}$$

 doi:10.20944/preprints202603.2311.v1

Therefore, equation [1] becomes:

$$f_i(t) = m_i a_i(t) = -\frac{\partial V}{\partial r_i} = m_i \frac{\partial^2 r_i}{\partial t^2} \quad (4)$$

The complex nature of interatomic interactions makes an analytical solution to this equation impractical for molecules with more than two atoms (Figure 1B). Consequently, the mobility of atoms over time is approximated using numerical integration techniques, such as the finite difference approach.

3. Finite Difference Method (FDM)

FDM is a class of numerical methods that approximate derivatives using finite differences in order to solve differential equations (Figure 1D). Numerous tiny, finite time increments δt can be used to break up the integration. For the process of interest to be adequately described, the time step should be sufficiently tiny. A decent δt should be in the order of femtoseconds because molecular motions occur in the range of 10−14 s. It is advisable to use a time step that is less than the model system's fastest time step. For atomistic simulations, 1-2 fs is typically utilized [10].

At each simulation step, all N equations of motion are solved simultaneously to determine the forces acting on each atom $(F_1, F_2, \ldots, F_N)$ at a given time t. The potential energy's negative gradient, which is computed using preset FFs, yields the force acting on each atom. After the forces are known, each atom's acceleration can be calculated as follows:

$$a_i = \frac{F_i}{m_i} = -\frac{\partial V}{m_i \partial r_i} \quad (5)$$

We can predict the position and velocity at the time step based on the molecule's position, acceleration, and velocity at time t. To create a trajectory that shows a particle's locations and velocities over time, this process is repeated numerous times (Figure 1A). Therefore, the final trajectory is not continuous but a series of configurations at each discrete time step (δt).

4. MD Simulation Algorithm Steps

1) The initial conditions, including the positions (r_i) and velocities (v_i) of all atoms, are defined at a specific time t. For systems derived from experimentally solved structures, atomic positions can be directly obtained from the crystallographic data available in PDB files. In contrast, for disordered systems, atomic positions can either be generated randomly or constructed as an ordered structure that is subsequently melted. The initial velocities of particles are typically assigned according to a Maxwell–Boltzmann distribution corresponding to the desired temperature, followed by adjustments to ensure zero total angular momentum and centre-of-mass velocity for the entire system. The potential energy (V) arising from interatomic interactions is then computed using FFs Equation 12 (described below).

2) The forces acting on each atom are derived from the potential energy function (V). This step is the most computationally demanding, as it requires evaluating complex interatomic interactions governed by numerous parameters. The force on each atom is expressed as:

$$\mathrm{F_i} = -\frac{\partial \mathrm{V}}{\partial \mathrm{r_i}}$$

3) Newton's equation of motion is solved to find the positions of atoms in the next step (t+δt).

$$\mathrm{a_i} = \frac{\mathrm{F_i}}{\mathrm{m_i}}$$

The above process is iterated for as many steps as needed. In each step, the initial configuration is updated with the new one (Figure 1A).

 doi:10.20944/preprints202603.2311.v1

4.1. The Verlet Algorithm

After computing the accelerations, the Verlet algorithm is employed as a widely used numerical method to integrate Newton's equations of motion [15]. It updates the positions of atoms over time using their current positions and accelerations, offering a stable and accurate approach for simulating atomic trajectories. The position and other dynamical properties of a molecule at time $t + \delta t$can be approximated using a Taylor series expansion around time t.

$$r(t + \delta t) = r(t) + \delta t v(t) + \frac{1}{2}\delta t^2 a(t) + \frac{1}{6}\delta t^3 b(t) + \cdots$$

$$v(t + \delta t) = v(t) + \delta t a(t) + \frac{1}{2}\delta t^2 b(t) + \cdots$$

$$a(t + \delta t) = a(t) + \delta t b(t) + \cdots$$

where v is the velocity, a is the acceleration, b is the third derivative of the position with respect to time, and so on. This approximation gets better when the δt is smaller (Figure 1D).

The Verlet algorithm utilises the current positions $r(t)$and accelerations $a(t)$, along with the positions from the previous time step $r(t - \delta t)$, to predict the positions at the next step $r(t + \delta t)$. The algorithm is derived by expressing the equations of motion for both past and future positions as follows:

$$r(t + \delta t) = r(t) + \delta t v(t) + \frac{1}{2}\delta t^2 a(t) + \cdots$$

$$r(t - \delta t) = r(t) - \delta t v(t) + \frac{1}{2}\delta t^2 a(t) + \cdots$$

Adding these together:

$$r(t + \delta t) + r(t - \delta t) = 2r(t) + \delta t^2 a(t) + \cdots$$

$$r(t + \delta t) = 2r(t) - r(t - \delta t) + \delta t^2 a(t) + \cdots$$

This is the basic Verlet algorithm equation. To determine the position of an atom at $r(t + \delta t)$, the positions at $r(t)$, $r(t - \delta t)$, and the acceleration $a(t)$ are required (Figure 2A). However, the Verlet algorithm has two main limitations. First, it does not explicitly include the velocity term $v(t)$. Therefore, additional calculations are required for the velocity term. Although velocities are not essential for tracing the molecular trajectory, they are required for computing properties such as kinetic energy. The velocity can be approximated using the mean value theorem, where $v = \delta r/\delta t$.

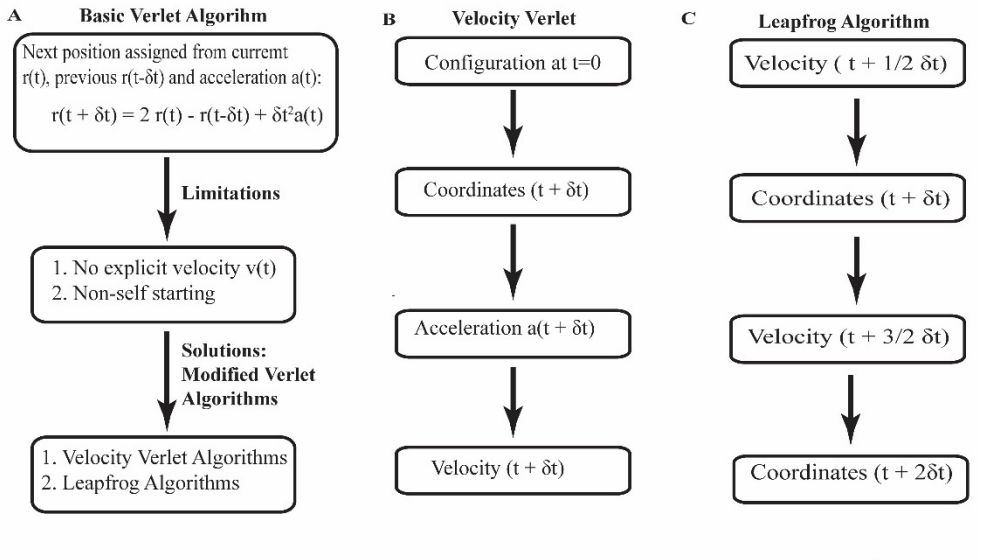
$$v(t) = \frac{\{r(t + \delta t) - r(t - \delta t)\}}{2\delta t}$$

$$v(t + \delta t) = \frac{\{r(t + \delta t) - r(t)\}}{2\delta t}$$

However, this calculation generally leads to large errors.

The second limitation of the Verlet algorithm is that it requires the position from the previous step, making it a non–self-starting algorithm. Consequently, at $t = 0$, the previous position $r(t - \delta t)$ must be estimated through alternative methods. To address these limitations, several modified versions of the Verlet algorithm have been developed. Two commonly used variants are discussed below.

 doi:10.20944/preprints202603.2311.v1



D

Integrators	Coordinates (r) and velocities (v) stored at each time step
Velocity Verlet	$[r_t = 0, V_t = 1], [r_t = 2, V_t = 2], [r_t = 3, V_t = 3]$
Leap Frog	$[r_t = 0, V_t = 0, 5], [r_t = 1, V_t = 1, 5], [r_t = 2, V_t = 2, 5], [r_t = 3, V_t = 3, 5]$

Figure 2. Method to integrate Newton's equations of motion. **A)** Basic Verlet algorithm method to integrate Newton's equations of motion. The atomic positions are updated using their current positions $r(t)$ and accelerations $a(t)$ along with the positions from the previous time step $r(t - \delta t)$, to predict the positions at the next step $r(t + \delta t)$. The limitations of the Verlet algorithm are addressed by **B)** Velocity Verlet and **C)** Leapfrog algorithm. **D)** During the simulation, switching between the Velocity Verlet and Leapfrog integrators is not possible because the velocities are defined and stored at different time steps in each method.

4.2. Velocity Verlet

The Velocity Verlet algorithm is one of the most widely used integration schemes in MD simulations (Figure 2B). It is derived from the Taylor expansions of position and velocity:

$$\boldsymbol{r}(t + \delta t) = \boldsymbol{r}(t) + \delta t \boldsymbol{v}(t) + \frac{1}{2}\delta t^2 \boldsymbol{a}(t)$$

$$v(t + \delta t) = v(t) + \frac{1}{2}\delta t[a(t) + a(t + \delta t)]$$

In this approach, the current position $r(t)$, velocity $v(t)$, and acceleration $a(t)$ are used to compute the molecular motion. First, the new position $r(t + \delta t)$ is determined from the current parameters. The system configuration is then updated, and the resulting acceleration $a(t + \delta t)$ is recalculated from the new forces acting on the atoms. Finally, the updated acceleration is used to compute the future velocity $v(t + \delta t)$ using the equation above (Figure 2B).

4.3. Leapfrog Algorithm

The Leapfrog algorithm is another integration method commonly used in MD simulations [16]. It is based on updating positions and velocities at staggered time intervals, where velocities are evaluated at half-time steps ($t + \frac{1}{2}\delta t$) relative to positions (Figure 2C). The equations are given by:

$$r(t + \delta t) = r(t) + \delta t v\left[t + \frac{1}{2}\delta t\right]$$

 doi:10.20944/preprints202603.2311.v1

$$v\left(t+\frac{1}{2}\delta t\right)=v\left(t-\frac{1}{2}\delta t\right)+\delta t a(t)$$

Initially, velocities are computed at $t+\frac{1}{2}\delta t$, and these are used to determine the positions at the next time step $t+\delta t$. The velocities then "leap" over the positions to the next half-step $t+\frac{3}{2}\delta t$, enabling the computation of subsequent positions at $t+2\delta t$. This iterative process continues over many time steps, producing the molecular trajectory. The name "leapfrog" arises because the positions and velocities advance alternately in time, much like two frogs jumping over each other (Figure 2C).

It is not possible to switch between the Velocity Verlet and Leapfrog integrators during a simulation because the velocities are defined and stored at different time steps in each method (Figure 2D). In the Velocity Verlet algorithm, both positions and velocities are evaluated at the same time points, whereas in the Leapfrog scheme, velocities are computed at half-time steps between successive position updates. Switching integrators mid-simulation would therefore introduce a discontinuity, effectively corresponding to reassigning new initial velocities to the system and disrupting the continuity of the trajectory (Figure 2D).

5. Time Steps

In MD simulation, the atomic motion is discretized into small finite time steps. The total simulation time T is the product of the timesteps δt and the number of timesteps N_t:

$$T=N_t\delta t$$

Appropriate time step selection is critical for accurately describing molecular motion in MD simulations (Figure 1D). The time step depends on the required numerical accuracy and the integrating algorithm [14]. It must be small enough to capture the fastest motions within a molecule. In biological systems, the fastest motions are represented by the oscillations of the hydrogen atom bound to a heavy atom (e.g., carbon). The bond oscillation of a C-H bond occurs ~10 fs (Table 1). Therefore, the size of the integration time step should be in the order of ~fs to capture these motions. The default choice is to employ a 1fs long-time step as it provides a reasonably accurate trajectory. However, if our interest is not in sampling the fastest bond oscillations, longer time steps can be used [17]. Most of the events of biochemical interest, for example, functionally important structural changes in proteins, take place on timescales of nanoseconds, microseconds, or longer. A typical simulation thus involves millions or billions of time steps [18,19].

Table 1. Bond oscillation time of various bonds found in biological molecules.

S.No.	Bonds Vibrations	Time (~fs)
1	X–H bond	10
2	O–H bond (water, hydroxyl groups)	9–10
3	N–H bond	10–12
4	C–H bond	10–15
5	C–C single bond	30–40
6	C=C	15–20
7	C≡C	10–15
8	C–N / C–O single bond	25–35
9	Angle-bending motions	40–100
10	Dihedral (torsional) motions	>100

 doi:10.20944/preprints202603.2311.v1

6. Algorithms to Constrain the Bond Lengths

In MD simulations, very short bond vibrations—especially those involving hydrogen atoms—occur at extremely high frequencies, requiring very small-time steps for accurate integration. To overcome this limitation and enable the use of longer time steps (typically 1–2fs), constraint algorithms are applied to fix bond lengths or angles during the simulation [20,21]. These algorithms ensure that certain internal coordinates remain constant while allowing the rest of the system to evolve dynamically according to Newton's equations of motion [22].

6.1. SETTLE

It is used to constrain the geometry of water molecules, specifically the O-H bond length and H-O-H bond angle, to rigid values. This constraint helps to improve the efficiency and stability of MD simulations, particularly when simulating systems containing water [23].

6.2. SHAKE (Shaking Algorithm for Kinetics and Equilibrium)

It is a computational technique used in MD simulations to enforce constraints, particularly those involving bond lengths and angles. It allows for larger time steps in simulations, leading to faster and more efficient calculations [24].

6.3. LINCS (Linear Constraint Solver)

It is an algorithm used in MD simulations to enforce constraints on bond lengths and angles within a molecule. It is a non-iterative method that efficiently and stably constrains bonds, allowing for larger time steps in MD simulations. LINCS is particularly useful for Brownian dynamics simulations and is the standard constraint algorithm in GROMACS [25]

6.4. RATTLE Algorithm

A velocity-based extension of SHAKE, RATTLE, corrects both positions and velocities to maintain constraints throughout the simulation. It is particularly suited for use with the Velocity Verlet integration scheme [26].

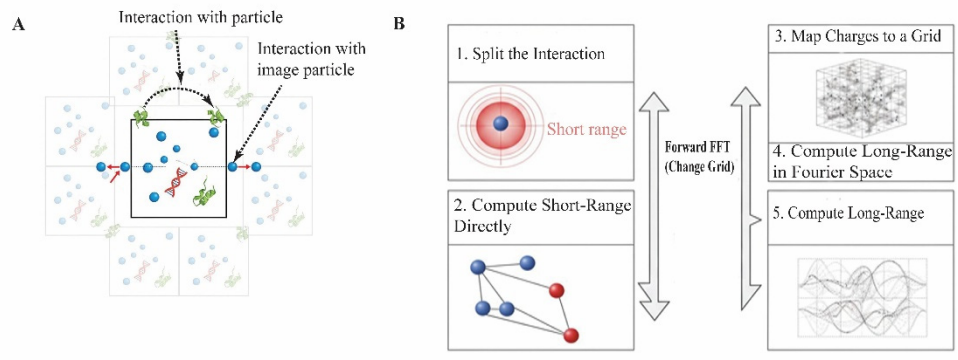
7. Periodic Boundary Conditions (PBCs)

While it is possible to simulate a system of N particles in a vacuum, studying molecular behaviour in bulk solvent requires boundary conditions to more accurately mimic real-world environments. The boundary conditions define the interaction of a system with its surroundings at the edges or surfaces (Figure 3A). They specify how the system behaves at these boundaries, providing essential constraints for solving governing equations and accurately modelling phenomena like fluid flow or heat transfer [27].

Using rigid or reflective walls introduces artefacts, as a significant fraction of particles interact with these boundaries. This fraction is proportional to $N^{-1/3}$, which, although negligible in macroscopic systems, becomes considerable in typical molecular simulations. It becomes approximately 0.49 for $N = 10^3$. Therefore, unless the simulation specifically focuses on confined systems (e.g., liquids in nanopores), it is essential to apply PBCs.

In PBCs, the simulation box is conceptually surrounded by infinite replicas of itself (Figure 3A). Only the N atoms within the central box are explicitly simulated, but when an atom exits the box through one side, an identical image atom enters from the opposite side, thereby preserving the density and continuity of the simulated bulk environment [28].

 doi:10.20944/preprints202603.2311.v1



Water Models	X (e)	Y (e)	Z (e)	Θ (°)	d_{OH} (Å)	r_{OM} (Å)
SPC	0.41	0.41	-0.82	109.47	1.00	_
SPC-E	0.42	0.42	-0.84	109.47	1.00	0.05
TIP3P	0.41	0.41	-0.83	104.52	0.96	_
TIP4P	0.52	0.52	-1.04	104.52	0.96	0.15
TIP4PEW	0.52	0.52	-1.04	104.52	0.96	0.125
TIP4P-FB	0.59	0.59	-1.96	104.52	0.96	0.15
OPC3	0.45	0.52	-0.90	103.62	0.97	

Figure 3. Periodic boundary conditions (PBC) and Particle Mesh Ewald (PME) in MD simulations. **A)** The simulation box is surrounded by infinite replicas of itself, ensuring bulk-like behaviour by allowing atoms leaving one side of the box to re-enter from the opposite side, thereby minimizing surface and finite-size effects. **B)** Electrostatic interactions are computed by splitting them into short-range real-space and long-range reciprocal-space components, enabling efficient and accurate calculations under periodic boundary conditions. **C)** Different water models used in MD simulations.

8. Particle Mesh Ewald (PME)

The PME method is used for computing long-range electrostatic calculations in periodic systems. It calculates the total Coulombic interaction using PBC and charge neutrality of the molecular system (Figure 3B). It is based on the Ewald summation, a technique that transforms the summation of interactions in real space into a summation in Fourier space [29]. This transformation, coupled with the use of fast Fourier transforms (FFTs), allows for a significant speedup in calculations compared to direct summation.

The core concept is to split the electrostatic interaction into two parts: a short-range part calculated directly in real space and a long-range part calculated in Fourier space, where it is represented as a sum of waves (Figure 3B). To enable the use of FFTs, the charge density is represented on a discrete grid or mesh in space. By combining real and Fourier space calculations with FFTs, PME achieves a computational complexity of O (N log N), where N is the number of particles, making it a highly efficient method for simulating systems with many charged particles [30].

9. Water Models in MD Simulations

In MD simulations, water models are computational representations of water molecules that simplify their interactions, allowing for efficient simulation of water's behaviour in various systems (Figure 3C). These models, which can be rigid or flexible and explicit or implicit, are crucial for simulating biological, chemical, and materials systems where water is a key solvent. Several water models are commonly used in simulations, each with varying levels of complexity and accuracy [31]. Rigid three-point models like SPC, SPC/E, TIP3P, and TIP4P are frequently used due to their computational efficiency, while more complex models, like TIP3P-FB, TIP3P-EW, and OPC3, offer improved accuracy in specific properties (Table 2 and Figure 3C) [32]

 doi:10.20944/preprints202603.2311.v1

Table 2. Description of various water model types used in MD simulation.

S. No	Water Model	Description	Reference
1	SPC	A three-site model with fixed charges on the oxygen and hydrogen atoms.	[136]
2	SPC/E	An improved version of SPC with refined charges to better reproduce water's properties.	[137]
3	TIP3P	Another three-site model with fixed charges and a Lennard-Jones interaction.	[138]
4	TIP4P	An extension of TIP3P, adding a fourth site to represent the charge dipole.	[139]
5	TIP4P-FB	A variant of TIP4P that uses a finite basis for the Lennard-Jones interactions.	[140]
6	TIP4P-EW	A variant of TIP4P that uses the Ewald summation method for long-range electrostatic interactions.	[141]
7	OPC3	A more complex model that optimizes the point charges and Lennard-Jones parameters for improved accuracy.	[142]

The choice of water model depends on the specific simulation and desired properties. The explicit water models represent individual water molecules and accurately capture hydrogen bonding and solvation effects. In contrast, implicit water models treat the solvent as a continuous medium using dielectric approximations, which greatly reduces computational cost but lacks detailed water–solute interactions [33]. Explicit models are suitable for simulations requiring an accurate representation of water's structure and dynamics, particularly when investigating solute-water interactions [33]. On the other hand, implicit models are faster and more computationally efficient, suitable for simulations where the explicit structure of water is not crucial, such as free energy calculations or long-time simulations. Factors which influence water model choice are 1) Computational Cost: Explicit models are computationally more expensive than implicit models, 2) Desired Properties: Different models excel at reproducing specific properties like density, surface tension, and solvation free energies, and 3) System of Interest: For example, simulations of protein-water interactions might require explicit water models, while simulations of large-scale systems might benefit from implicit models [34].

10. Force Fields (FFs)

The forces acting on atoms in a system are calculated using FFs, a collection of empirical energy functions and parameters. It is a mathematical statement that explains how a system's energy depends on its particle coordinates. During the simulation, these forces dictate how the atoms move. The complicated interactions between atoms are approximated by FFs, which offer a computer method of simulating molecular behavior [35–37]. High-level quantum mechanical computations and experiments are used to develop the FFs. (I) All-atom, (II) United-Atom, and (III) Coarse-grained FFs are used in the design of FFs. Any little atom in the system, including hydrogen, can benefit from all-atom FFs. For certain big groups, such as the methyl molecule, United-Atom force fields are helpful. Lastly, the long-term protein simulation uses the coarse-grained FFs [38].

Covalent bonds, angles, dihedrals, van der Waals, and electrostatic potentials are among the formulas found in the FFs. Lennard-Jones potentials for van der Waals, periodic functions for bond rotations, springs for bond length and angles, and Coulomb's law for electrostatic interactions are the molecular properties represented by FF (Figure 4A) [39].

 doi:10.20944/preprints202603.2311.v1

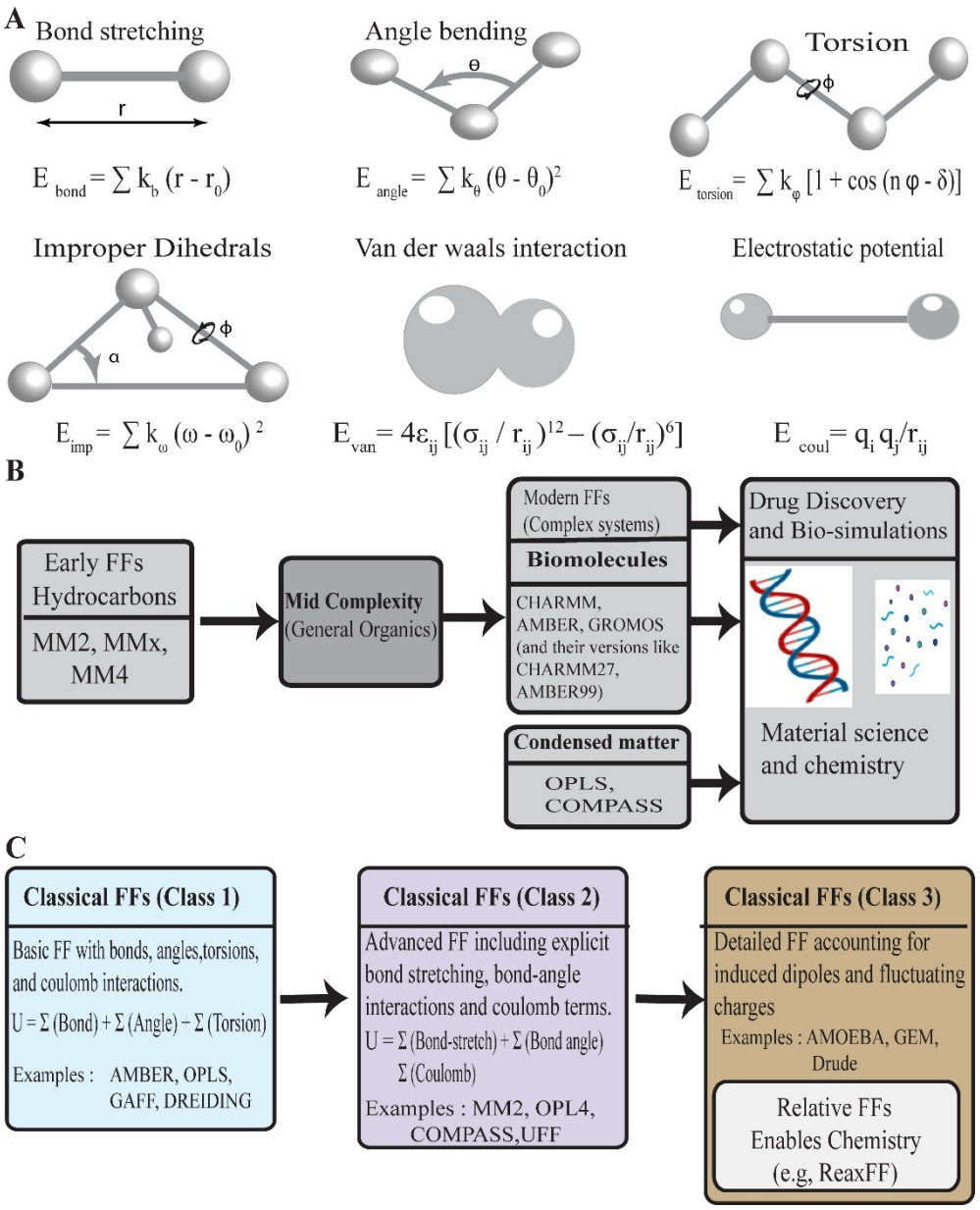


Figure 4. A) The force field comprises bonded terms, including bond stretching, angle bending, torsional, and improper dihedral interactions, while non-bonded interactions are described using Lennard–Jones potentials for van der Waals forces and Coulomb's law for electrostatics. **B)** FFs have evolved from describing simple hydrocarbons to modelling a wide range of organic molecules. **C)** FFs are classified as classical, polarizable, and reactive based on how they treat electronic effects such as polarization.

The FF mathematical statement is composed of an analytical form of the interatomic potential energy, U(r1, r2,..., rN), and a set of parameters that enter this form. The parameters can be obtained by a variety of techniques, including ab initio and semi-empirical quantum mechanical calculations and fitting to experimental data from neutron, X-ray, and electron diffraction, NMR, infrared, Raman, and neutron spectroscopy. The FF replaces the actual potential with a simplified model that works in the simulated region. A group of atoms joined by basic elastic (harmonic) forces is what is known as a molecule. The mathematical expression for an FF equation is the sum of classical potential energy, plus a few more parameters [39,40]. The sum of the intramolecular energies of the constituents and the sum of the intermolecular interaction energies between every atom are added to determine the potential energy:

 doi:10.20944/preprints202603.2311.v1

$$E(\overrightarrow{X}) = \sum_{a<b} E_{ab} + \sum_{a} E_a^{int} \tag{6}$$

The intramolecular or local energy is made up of bond stretching, angle bending, and improper and dihedral torsions. The length of covalent bonds is regulated by a straightforward harmonic function that represents bond stretching. Bond stretching intramolecular potential energy concepts are explained by:

$$E_{bond} = \sum_{i} k_{b,i}(r_i - r_{0,i})^2 \tag{7}$$

Experimentally solved structures can yield approximate values for r0. Raman or infrared spectra can be used to determine the spring constant (kb). For bond displacements greater than 10% from the equilibrium value, however, the harmonic potential is not a reliable approximation. Furthermore, the harmonic function suggests that the connection cannot be disrupted, making it impossible to study chemical processes.

A harmonic potential represents angle bending:

$$E_{bend} = \sum_{i} k_{\vartheta,i}(\vartheta_i - \vartheta_{0,i})^2 \tag{8}$$

Molecules with more than four atoms require the use of dihedral or torsional terminology. Rigid constraints can typically be used to represent high-frequency motions such as bond stretching and angle bending, which are frequently not significant in long-timescale simulations. Torsional motions, on the other hand, are essential for accurately representing the molecule's flexibility and are comparatively less rigid than bond vibrations [41]. These motions are particularly crucial for illustrating significant conformational changes caused by rotations around single bonds. Torsional concepts are therefore essential for figuring out the local structure of macromolecules and for comprehending the relative stability of their different conformational states [43]. A dihedral potential with variable depths of minima can be created by mixing several torsional terms with different n values. Usually obtained from ab initio quantum mechanical calculations, the torsional parameters are further adjusted using experimental data, such as vibrational spectra or molecule geometries, to increase their precision and dependability [44]. A cosine function is typically used to represent torsional energy:

$$E_{torsion} = \sum_{i} \{V_{1,i}\,(1 + \cos\varphi_i)/2 + V_{2,i}\,(1 - \cos 2\varphi_i)/2 + V_{3,i}V_{1,i}\,(1 + \cos 3\varphi_i)/2\} \tag{9}$$

At last, to preserve the planarity of certain chemical groups, like sp^2 hybridized carbon atoms in carbonyl groups or aromatic rings, an extra energy component known as the incorrect torsion term is included. This is required since such geometrical constraints are frequently not enforceable by the ordinary torsional potential alone. The structural integrity of planar groups is maintained throughout the simulation because the incorrect torsion term takes into consideration the energy penalty related to out-of-plane deviations. Typical mathematical expressions used to represent the improper torsion term are as follows:

$$U_{imp} = \sum_{impropers} \frac{k_{imp}}{2}[1 + \cos(2\omega - \pi)], \text{ or}$$

$$U_{imp} = \sum_{impropers} \frac{k_{imp}}{2}(\omega - \omega_0)^2$$

 doi:10.20944/preprints202603.2311.v1

ω is the improper angle corresponding to the deviation from planarity.

Repulsive and attractive forces combine to create Van der Waals interactions between two atoms. Attractive forces result from the interactions between induced dipoles, while repulsive forces are caused by the overlap of the electron clouds of both atoms. The force of attraction changes as r−6. These interactions are represented by the Lennard-Jones (LJ) potential in conjunction with a few "softer" terms to characterize the repulsive portion, such as the exponential function employed in the Buckingham potential [43]. Van der Waals forces can act between any two atoms in distinct molecules, but they can also act between atoms in the same molecule that are far enough apart [46].

Electrostatic interactions are taken into consideration by the FF expression's last term. One popular method is to give nuclei partial atomic charges and use Coulomb's equation to compute electrostatic contributions. Partial charges for tiny molecules can be calculated using atomic electronegativities or obtained by fitting to experimental thermodynamic data. Another popular technique is to perform ab initio QM calculations and then fit the electrostatic potential to obtain partial charges. However, because atomic charges are not directly measurable experimental observables, this process creates uncertainty [45]. It is crucial to remember that polarization effects in condensed-phase systems are not entirely captured by fixed partial charges. Furthermore, because electrostatic interactions are long-range, they need to be taken into account when using cutoff methods in simulations. Partial charges are usually only allocated to atomic centres for computational efficiency, while there is no fundamental prohibition against removing them from an atom if needed. Coulomb and Lennard-Jones interactions represent intermolecular energy terms for atoms separated by three or more bonds (Eq. 10). Intermolecular energy also includes the "nonbonded" interactions between intermolecular atom pairs (Eq. 11):

$$E_{nb} = \sum_{i<j} \left\{ q_i q_j e^2 / r_{ij} + 4\varepsilon_{ij} \left[\left(\sigma_{ij} / r_{ij} \right)^{12} - \left(\sigma_{ij} / r_{ij} \right)^{6} \right] \right\} \quad (10)$$

$$E_{ab} = \sum_{i}^{\epsilon a} \sum_{j}^{\epsilon b} \left\{ q_i q_j e^2 / r_{ij} + 4\varepsilon_{ij} \left[\left(\sigma_{ij} / r_{ij} \right)^{12} - \left(\sigma_{ij} / r_{ij} \right)^{6} \right] \right\} \quad (11)$$

Therefore, the final FF equation takes the following form-

$$\boldsymbol{U} = \sum_{bonds} \frac{1}{2} k_b (r - r_0)^2 + \sum_{angles} \frac{1}{2} k_a (\theta - \theta_0)^2 + \sum_{torsion} \frac{V_n}{2} [1 + cos(n\emptyset - \delta)] + \sum_{improper} V_{imp} + \sum_{LJ} 4\epsilon_{ij} \left(\frac{\sigma_{ij}^{12}}{r_{ij}^{12}} - \frac{\sigma_{ij}^{6}}{r_{ij}^{6}} \right) + \sum_{elec} \frac{q_i q_j}{r_{ij}} \quad (12$$

11. Important Force Fields

The FFs were developed with the advancement of the molecular mechanics (MM) method for the prediction of molecular structures, vibrational spectra and enthalpies of small organic molecules [46]. Earlier FFs, such as MM2, MM3, and MM4, were developed to study hydrocarbons [47]. However, these FFs were later modified to deal with various organic molecules (Figure 4B).

As the scope of research has advanced, the FFs have been modified to handle increasingly complex systems. The Dreiding and universal FFs consist of parameters for all the atoms present in the periodic table [48,49]. Other commonly used FFs are CHARMM [50], AMBER [51], GROMOS [52], OPLS [53], and COMPASS [54]. CHARMM, AMBER, and GROMOS are often used for biomolecule simulations, whereas OPLS and COMPASS are used for condensed matter simulations. Several variants of these FFs—CHARMM19, CHARMM22, CHARMM27, GROMOS96, GROMOS45A3, GROMOS53A5, GROMOS53A6, AMBER91, AMBER94, AMBER96, AMBER99, AMBER02, etc.—are accessible after additional modifications. These modified versions of FFs can handle proteins and nucleic acids [55].

 doi:10.20944/preprints202603.2311.v1

FFs are classified as classical, polarizable, and reactive force fields. Classical force fields are further divided into three classes based on the complexity of their potential energy functions (Figure 4C) [56]. Class I FFs consist of the potential energy U term as bond, angle, torsion and LJ and electrostatic (ES) interaction (non-bonded, electrostatic and van der Waals). DREIDING, AMBER, GAFF and OPLS are examples of Class I FFs [57]. Class II force fields include bond-bond and bond-angle coupling terms in addition to the Class I potential energy [58]. The examples of class II force fields are COMPASS, UFF, MM2, MM3, MM4, CFF (consistent force field) and MMFF (Merck molecular force field). Class III FFs account for fluctuating charges or dipoles to include the effects of polarization. They are used to model systems where polarization is important, such as biomolecules in aqueous environments (Figure 4C). Drude oscillators, AMOEBA, GEM, and pGM are examples of polarizable force fields [59,60]. Now, the polarization has been introduced into CHARMM using fluctuating charges or the shell model. Similarly, AMBER, OPLS and GROMOS have also been extended to include polarization effects. [57,61]

Reactive FFs are bond order-based force fields that allow for bond breaking and formation during the simulation. These are the powerful computational tools for exploring, developing and optimizing material properties. They are useful for studying chemical reactions and other processes where bond changes occur. ReaxFF is a well-known example of a reactive force field [62,63].

12. Accelerating MD Simulations

Millions to trillions of time steps must be handled by MD simulations. A significant amount of processing time is needed at each stage. Every pair of atoms has non-bonded interactions; for instance, the number of non-bonded terms in a system with N atoms is proportional to N^2 [64]. As these forces diminish rapidly with distance, van der Waals interactions between atoms separated by a certain distance can be disregarded. However, because electrostatic interactions weaken with distance, they cannot be disregarded. Large macromolecule MD simulations used to require a supercomputer. But these days, it's a lot easier to get.

Powerful simulations may now be run locally because of developments in simulation methods and computer hardware, especially graphics processing units (GPUs) [65,66]. GPUs are specialized hardware devices that can efficiently and quickly carry out parallel floating-point arithmetic operations (Figure 5A). Simulation times can be accelerated by GPUs since they can process data more quickly and efficiently than CPUs. By utilizing the GPU's parallel processing capabilities, MD simulations can be substantially accelerated by using a GPU workstation. Using the GPU to compute non-bonded interactions (van der Waals and real-space Coulomb forces) and the CPU to compute other interactions (PME, bonded interactions, NMR restraint, etc.) can further boost performance [67]. This method assures the appropriate use of the CPU's power. Significant time and money savings may arise from this. The benefits of parallelization and accelerators to speed up computing processes are features of the current generation of computers. GPUs are perfect for the parallel nature of MD calculations since they can handle thousands of threads at once. When combined with several cores, the messaging passing interface (MPI) can significantly shorten computation times. Additionally, GPUs' extremely parallel internal architecture makes them effective computing accelerators. Several hundred independent processing units (cores), far more than those typically found on CPUs, make up GPUs. In contrast to CPUs, each GPU core is built to carry out a large number of tasks—thousands, even—known as threads, which carry out extremely basic processes. Because of these factors, GPUs are now effectively employed as computational accelerators in both the scientific and gaming domains [68]. The MPI and GPU accelerators work with the main simulation applications. Thus, MD simulation operations have been significantly accelerated using GPU simulation in conjunction with MPI (Figure 5A).

 doi:10.20944/preprints202603.2311.v1

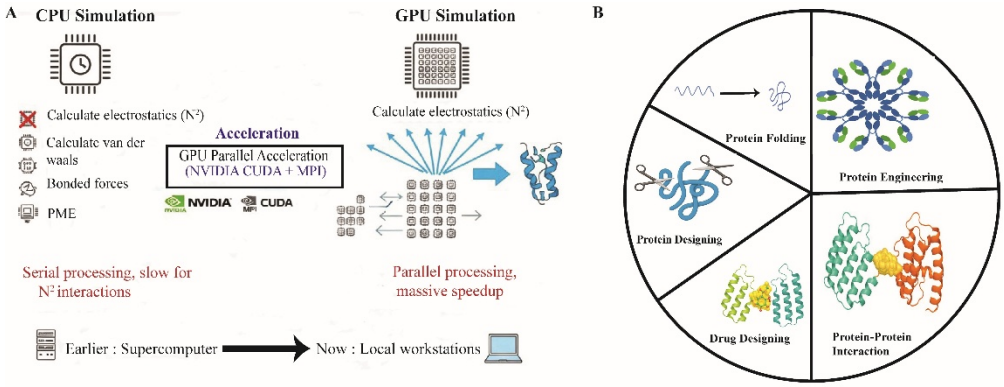


Figure 5. Accelerating MD simulations. **A)** Graphics processing units (GPUs) are specialized hardware devices that can significantly accelerate MD simulations by leveraging the GPU's parallel processing capabilities. **B)** Application of MD simulations in protein-protein interactions, protein engineering, drug designing, protein designing, and protein folding.

Compute Unified Device Architecture (CUDA) is a parallel computing platform and programming model developed by the NVIDIA Corporation (Santa Clara, California, USA). MD simulations are a great fit for GPUs with CUDA support [69]. Quadro and Tesla are two significant GPU card families that are built for demanding and extended workloads. The 24 GB of onboard RAM is more than other GPUs' 6–11 GB. These GPUs are more expensive because they include error correcting codes (ECC), which call for additional hardware. Nonetheless, the Giga Texel Shader eXtreme (GTX) card family of gaming GPUs is becoming more and more popular in MD simulations. GPUs from the GTX family offer amazing performance at incredibly low costs [70,71]. Thus, people can choose their compute cluster based on the MD simulation program, the size and complexity of the simulation system, and the available budget. The list of NVIDIA GPUs that are suggested for MD simulation is provided below (Table 3):

Table 3. List of GPUs from NVIDIA used in MD simulations.

S. No.	Graphics	Features	References
1	NVIDIA A100	Strong computing capabilities. High memory bandwidth.	**[143]**
2	NVIDIA H100	Advanced architecture. High performance.	[144]
3	NVIDIA A40	High-performance computing. Perfect for mixed workloads.	[145]
4	RTX 4090	More advanced architecture. Dedicated hardware for real-time tracing.	[146]

12.1. NVIDIA A100

Because of its powerful processing power and large memory capacity, the NVIDIA A100, which is powered by the NVIDIA Ampere Architecture, is an excellent GPU for MD simulations. It is perfect for simulating big biomolecular systems because of its capacity to handle heavy tasks and vast datasets. With over 2 TB/s of memory bandwidth, the A100 80GB can handle the biggest models and datasets in the world [72,73].

12.2. NVIDIA H100

The H100 is specifically designed for combining FP8 precision, AI, and FP16. It works nicely with MD simulation programs like AMBER and GROMACS. It is a suitable option for demanding MD workloads due to its sophisticated architecture and high performance. However, it might not be

 doi:10.20944/preprints202603.2311.v1

the best option for exclusively MD simulations due to its high cost and more specialized focus on AI and FP16/FP8 precision [74].

12.3. NVIDIA A40

The NVIDIA A40 GPU is excellent for high-performance computing and is perfect for virtual production, simulation, and ray-traced rendering. It is ideal for mixed workloads, such as extensive MD simulations, because of its strong capabilities, which support both visuals and computationally demanding activities [75].

12.4. RTX 4090

The primary distinction between GTX and RTX (Ray Tracing Texel eXtreme) cards is that GTX cards lack the specialized technology needed for real-time ray tracing, which produces more lifelike lighting, reflections, and shadows. Because of its more sophisticated design and dedicated cores for high-performance computing, an NVIDIA RTX card is far superior to a GTX card for MD simulations. The newest RTX cards offer more processing power and capabilities that significantly speed up the intricate calculations needed for MD, whereas prior GTX cards were an option for entry-level simulations [76]. The RTX 4090 offers the lowest price per core and per FP32 throughput ratio. For MD simulation, the RTX 4090 has excellent performance. Scalability is the RTX 4090's drawback. Scaling requires significant changes like water cooling, a customised chassis, PCIe risers, etc. RTX 4090 is therefore a viable option for independent researchers who operate locally.

13. The Popular MD Simulation Programs

The most widely used MD simulation programs are GROMACS, LAMMPS, AMBER, NAMD, and CHARMM. GROMACS and LAMMPS are open source, while AMBER, NAMD, and CHARMM are either commercial or have commercial versions [79].

GROMACS is a versatile package to perform molecular dynamics, i.e., simulate the Newtonian equations of motion for systems with hundreds to millions of particles. GROMACS is a community-driven and open-source, high-performance software used to simulate biomolecules such as proteins, lipids, and nucleic acids. GROMACS is known for its exceptional speed and efficiency, making it one of the fastest tools. GROMACS's unique features include support for a wide range of FFs, built-in tools for system setup and analysis, and advanced algorithms for energy conservation and long-range interactions [78]. It efficiently handles large biomolecular systems and supports parallel computing on CPUs and GPUs [79]. Its versatility and strong community support make it a preferred choice for simulations in structural biology, chemistry, and materials science. NVIDIA H100, A40 and T4 can be used to run GROMACS. The A40 combines professional graphics with powerful compute capabilities. T4 is a cost-effective alternative; the T4 provides excellent energy efficiency and sufficient power for most GROMACS workloads [80].

LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) is a powerful, open-source molecular dynamics software developed by Sandia National Laboratories (USA). It is widely used in materials science, soft matter physics, and biomolecular simulations [81]. One of its key strengths is its modular and extensible architecture, allowing users to easily implement custom potentials, FFs, and simulation methods. LAMMPS supports a wide variety of FFs, including classical (Lennard-Jones), reactive (ReaxFF), and coarse-grained models. It is designed for massively parallel computing using MPI, making it suitable for simulations involving millions of atoms on high-performance computing platforms. LAMMPS also supports various ensembles (NVE, NVT, NPT), flexible boundary conditions, and advanced integration schemes [82]. It can be accelerated using GPUs and Kokkos for improved performance, and integrates well with external tools like VMD, OVITO, and Python scripting for analysis and visualization [81]. Its versatility and scalability make LAMMPS an excellent choice for researchers working on complex simulations in both materials and biological systems.

 doi:10.20944/preprints202603.2311.v1

AMBER (Assisted Model Building with Energy Refinement) is a widely used MD software suite primarily designed to simulate biomolecules. Originally developed at the University of California (San Francisco), AMBER is both a set of FFs and a collection of simulation programs, with Sander and PMEMD being its core engines. One of AMBER's unique features is its highly accurate and extensively validated FFs, particularly ff14SB, GAFF, and ff19SB, which are well-suited for biomolecular simulations [83,84]. It supports free energy calculations, enhanced sampling techniques, and QM/MM hybrid simulations, making it a powerful tool for drug discovery and enzyme mechanism studies. AMBER is optimized for high-performance computing, with GPU acceleration via pmemd. CUDA provides substantial speed-ups for large systems. It also features tight integration with preparation and analysis tools like LEaP (for system setup), cpptraj (for trajectory analysis), and Antechamber (for small molecule parameterization). Its focus on biological relevance, precision, and methodological depth makes AMBER a preferred tool in computational structural biology and pharmaceutical research [85]. Both NVIDIA A100 and H100 GPUs are used to ensure efficient processing of AMBER workloads. The H100 offers exceptional performance with advanced features like NVLink and NVSwitch for efficient GPU communication [88].

NAMD (Nanoscale Molecular Dynamics) is a high-performance MD program designed for simulating large biomolecular systems with high accuracy and efficiency. Developed by the Theoretical and Computational Biophysics Group at the University of Illinois (USA), NAMD is especially well-known for its excellent scalability across parallel computing platforms, from desktops to supercomputers [87]. One of its unique features is its tight integration with the molecular visualization program VMD, which simplifies system setup, simulation monitoring, and data analysis. NAMD supports multiple FFs, including CHARMM, AMBER, and OPLS-AA, and provides robust implementations for constant pressure and temperature ensembles, free energy perturbation (FEP), and replica exchange molecular dynamics (REMD). It also supports hybrid QM/MM simulations for studying enzymatic reactions and other complex processes. Additionally, NAMD offers efficient GPU acceleration, allowing users to simulate systems with millions of atoms. User-friendliness, performance, and support for advanced simulation techniques make NAMD a preferred tool in structural biology, drug discovery, and membrane protein research (Melo et al., 2018). NVIDIA A40's high memory capacity and robust performance make it ideal for scaling NAMD workloads. The L40S is designed for multi-workload acceleration, making it ideal for complex NAMD simulations [89]

CHARMM (Chemistry at HARvard Macromolecular Mechanics) is a versatile and widely used MD simulation program developed for modelling and analysing the behaviour of biological macromolecules, including proteins, nucleic acids, lipids, and carbohydrates [90,91]. One of its defining strengths is its extensive and highly refined FFs (such as CHARMM36), which are particularly well-suited for accurate simulations of complex biomolecular systems. CHARMM supports a broad range of simulation methods, including energy minimization, molecular dynamics, Monte-Carlo simulations, and hybrid QM/MM calculations. A unique feature of CHARMM is its high degree of flexibility and scriptability, allowing users to define custom workflows, restraints, and algorithms with ease. It also offers specialized tools for solvation, membrane modelling, and ligand docking, and is compatible with enhanced sampling techniques such as umbrella sampling, replica exchange, and free energy perturbation. Although CHARMM was initially command-line-driven, it has evolved to support parallel computing and interfaces with graphical tools like CHARMM-GUI, which simplifies system preparation for novices [92,93]. Its scientific rigour, comprehensive capabilities, and ongoing development make CHARMM a powerful platform for detailed biomolecular simulation studies.

AI²BMD: An AI-based ab initio biomolecular dynamics system (AI²BMD) that can efficiently simulate full-atom large biomolecules with ab initio accuracy [94]. AI²BMD uses a protein fragmentation scheme and a machine learning FF to achieve generalizable ab initio accuracy for energy and force calculations for various proteins comprising more than 10,000 atoms. It saves several orders of magnitude in calculation time as compared to density functional theory. Through many

 doi:10.20944/preprints202603.2311.v1

hundreds of nanoseconds of dynamics simulations, AI2BMD proved its capacity to effectively investigate the conformational space of proteins and peptides, obtaining precise 3J couplings that correspond to nuclear magnetic resonance investigations and illustrating the folding and unfolding processes of proteins. Moreover, accurate free-energy calculations for protein folding are made possible by AI2BMD, and the projected thermodynamic properties are in good agreement with experimental. AI2BMD may be able to detect the dynamic processes of bioactivities, supplement wet-lab studies, and facilitate biomedical research that is currently unfeasible [94].

14. Application of MD simulation

Protein-protein interactions (PPI): Many protein functions are based on PPIs, and understanding the three-dimensional (3D) structures of protein-protein complexes offers the structural, mechanical, and dynamical information needed to comprehend these functions (Figure 5B). Numerous groups have used simulations to describe PPI interfaces [4,5,22] . MD simulations have been used to examine the dynamic involvement of several ions, including water, in the interaction of proteins [95]. Additionally, the development of a complex involving two or more binding partners can be tracked. MD simulation has been used to decipher the full complex building process of the gating brake (GB) peptide and calmodulin (CaM) from T-type ion channels (Figure 6). After separating the CaM and GB peptides into non-interacting locations, the simulation was initiated. The two parts approached one another, scanned the binding interface, and created a stable complex during the simulation [96].

At the protein-protein interface, small, low-frequency interactions can be investigated. Analysis of Residue Interaction Networks (RINs) is a method for finding these crucial residues. By combining all of the interactions found in the MD-simulated structures, RIN analysis could more effectively identify the crucial residues of the binding of RBD and ACE2 [97].

Binding interactions are quantified by dissociation constants Kd, while binding and nonbinding interactions are characterized by second osmotic virial coefficients B2. B2 is calculated in relation to liquid–liquid phase separation in solutions containing proteins and nucleic acids. Implicit solvents can be used in Monte Carlo simulations or molecular dynamics to compute B2 [98].

Using an MD-based sampling strategy, the association and dissociation of five protein pairs were investigated. Instead of lengthy interfacial exploration while the proteins were in contact, the native state was attained by dissociation and eventual reassociation near the native interface if the proteins made contact far from the native interface [99].

Protein folding: Protein folding is still, in many respects, a mystery to be solved despite advancements in techniques. Protein and medication design, as well as genetic interpretation, may benefit from precise protein folding energetics calculations. A useful method for researching protein folding mechanisms is MD simulation (Figure 5B). A better understanding of these intricate biological processes has resulted from simulations of protein folding on realistic timeframes made possible by advances in computation, sampling approaches like replica-exchange MD, and enhanced FFs. Realistic empirical FFs can now be used to mimic the folding of tiny proteins in all-atom detail, thanks to programs like Folding@home [100]. The folding time of proteins in explicit solvents, which ranges from microseconds to tens of seconds, is the main obstacle [2] As a result, research is limited to tiny fast-folding proteins. The energetics (ΔH and ΔCp) of protein folding were computed by calculating the energy of the folded and unfolded states at various temperatures [101].

The fast-folding WW domain mutants fold in three to five microseconds. A 10-microsecond simulation of the Pin1 WW domain was recently produced, starting from a fully unfolded state using a specially tailored version of NAMD. Regretfully, the protein in this simulation misfolded into an alpha helical state that held steady for most of the route (Freddolino et al., 2008).

With the explicit solvent model under the AMBER14SB force field, the eight helical proteins (2I9M, TC5B, 1WN8, 1V4Z, 1HO2, 1HLL, 2KFE, and 1YYB) were folded into their native structures at room temperature using both traditional MD and accelerated molecular dynamics (AMD) from the linear structures. Within 40–180 ns, folding took place in AMD simulations. not using

 doi:10.20944/preprints202603.2311.v1

conventional MD simulation, but beginning with the linear structures of the eight proteins at 300 K [103].

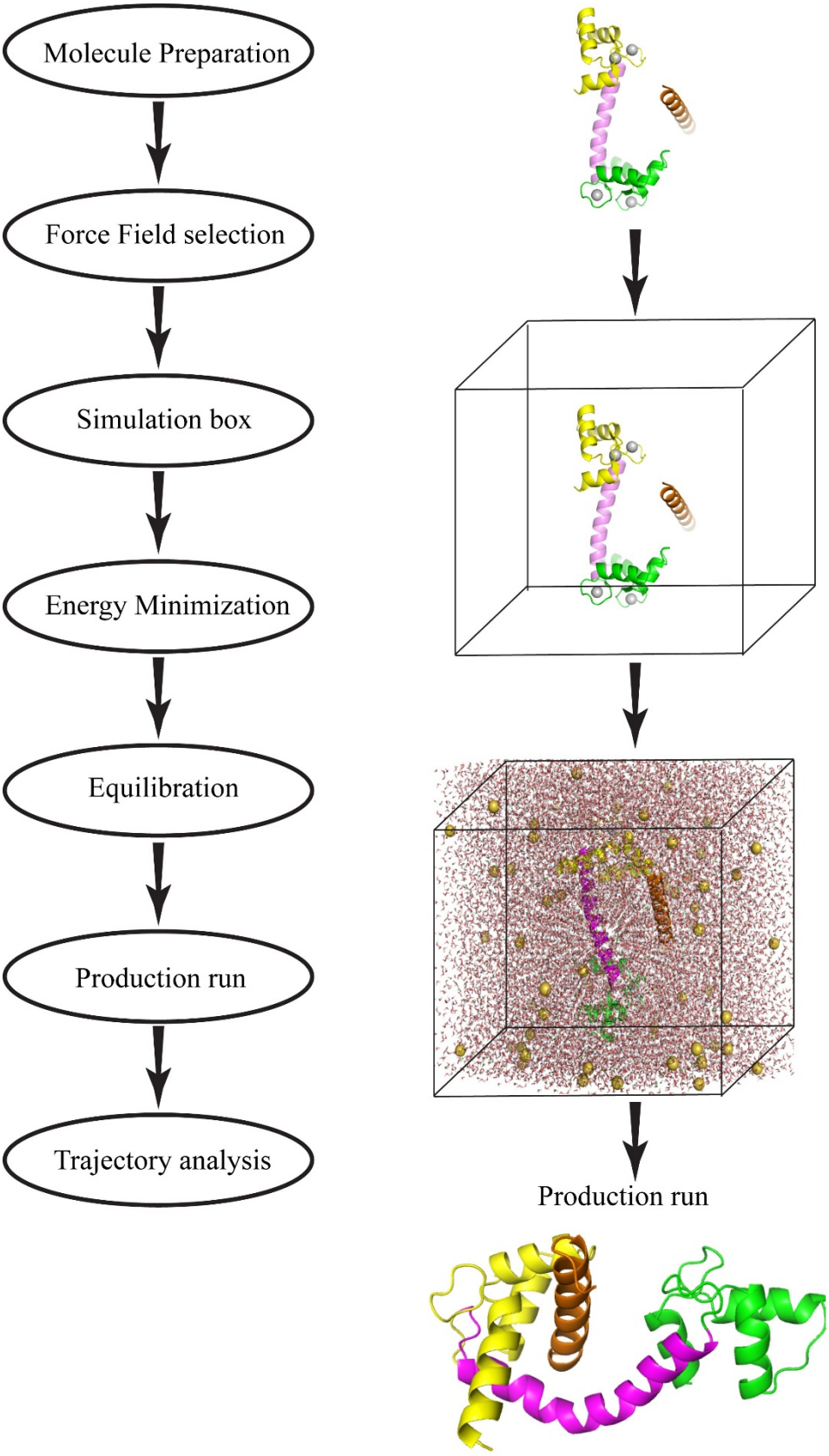


Figure 6. Protein-protein interactions study using MD simulation. Calmodulin (CaM) and Gating brake (GB) peptide from T-type calcium channels were placed apart in non-interacting positions. During the simulation, the GB peptides moved towards each other and bound at their natural binding site.

 doi:10.20944/preprints202603.2311.v1

The structural characteristics of protein-folding transition states are often mapped using Φ-values. About half (30 runs) of the sixty MD simulations of the putative transition-state ensemble (TSE) conformations of the src SH3 protein were able to reach the folded state. This suggests that Φ-value analysis can offer insightful information about the transition state's structural characteristics. The study also showed that quick folding requires tight side-chain packing between the distal hairpin and the diverging turn. Additionally, interactions between the N- and C-terminal strands are less important kinetically than those within the central β-sheet, which is the TSE's most organized area [1]

Dynamics conformations fall into two general kinds and can occur in many different contexts. Protein intrinsic factors are what drive the first category. Higher flexibility, for instance, is produced by disordered regions that lack α-helices or β-sheets; transitions between various conformations are also made easier by relative rotations or alterations between structural domains. Alternative conformations impacted by outside environmental factors fall under the second group. On the one hand, interactions with other macromolecules or the binding of tiny ligands can cause various conformational states. Conversely, the stability and shape of the protein can be directly impacted by variations in environmental parameters, including temperature, pH, and ion concentration. It is interesting to note that both internal and external stimuli influence the dynamic conformations of proteins.

Protein design: The large conformational space and the requirement for stability and specificity make designing new proteins with desired functionalities extremely difficult (Figure 6B). Proteins with specific functionalities can be designed by using MD simulations to investigate the conformational flexibility of proteins. The creation of new proteins has been made easier by recent developments in artificial intelligence. Protein sequences and structures that satisfy particular functional criteria have been produced using generative models, such as diffusion models and protein language models [104]. MD simulations are then used to improve these AI-generated designs in order to evaluate their dynamics, stability, and functional qualities. MD simulations can be used to study three crucial elements of protein function: flexibility, conformational changes, and allosteric control. These dynamic elements can be included in the design process to produce proteins with improved specificity and usefulness [105]. For de novo decapeptide design, sophisticated computational techniques such as genetic algorithms, deep learning algorithms, and reinforcement learning were employed. Aggregation propensity and solvent-accessible surface area can be assessed using coarse-grained MD simulations [106]

In order to use the designed protein model, it is crucial to estimate its quality. Information on model quality can be obtained using MD simulations. The model quality was evaluated using three features generated from MD: root-mean-square deviation (RMSD), the proportion of secondary structure, and the fraction of native contacts to the original structure [109].

Drug design: Drug design has been guided by the use of deep learning (DL) and MD simulation. Drug development also depends on an understanding of the protein-protein/ligand complex's binding energy (Figure 6B). DL and MD simulations can be used to tackle problems in molecular docking, drug virtual screening, protein structure and binding prediction, and complex evolution. Without sampling the protein conformational rearrangements during ligand binding, MD simulation can solve the issues with ligand docking computations [110].

Drug design relies heavily on accurate binding affinity prediction. The two most used techniques for determining free energies of binding are thermodynamic integration (TI) and free-energy perturbation (FEP). Improved sampling techniques have been used to speed up these computations because these methods need a large sampling to reach convergence. Furthermore, ML-based potentials have been used in MD simulations to enhance the precision and effectiveness of free-energy computations [109].

Protein engineering: Protein engineering is used to create proteins with enhanced properties or to display new or desired functions. Experimental procedures need a great deal of time and effort. On

 doi:10.20944/preprints202603.2311.v1

the other hand, in silico protein design is a quicker and less expensive option. In protein design, MD simulation can be used to precisely model the dynamics of the target protein.

To determine the appropriate locations for adding new N-glycosylation sites on human beta interferon (rhuIFN-β), Samoudi et al. used a computational approach. The anticipated structures' molecular stability and flexibility were assessed using the MD simulation. Human coagulation factor hyperglycosylated forms were created using a similar methodology. IV. MD modelling was used to identify the novel glycosylation sites [110,111].

PEGylation is a widely accepted approach to circumvent the cytotoxicity problem of drug molecules. It also improves the solubility and circulation time of the drug molecules. The MD simulation was used to assess the interaction between insulin and the different chain lengths of the PEG polymer [112].

15. Types of Simulation

The MD simulations are broadly classified as classical MD and ab-initio MD (AIMD). The difference lies in how they calculate inter-particle interactions. Classical MD provides information on large-scale structural and thermodynamic properties of systems, while ab-initio AIMD yields highly accurate, electronic-level details, including chemical reactions and charge transfer, for smaller systems and shorter timescales [113]. Classical MD uses simplified FFs, while AIMD solves the Schrödinger equation to determine the forces. In essence, classical MD excels at observing the collective, long-term behaviour of large systems, while AIMD provides fundamental, highly accurate insights into the specific chemical and electronic processes that govern those behaviours at a microscopic level.

Other important variations include coarse-grained MD, which simplifies the system by modelling fewer particles, and those based on specific ensembles like NVT or NPT [38,114].

Classical Molecular Dynamics (MD): Uses empirical FFs to approximate the potential energy of the system. It is computationally less expensive, allowing for larger systems and longer simulation times [115,116]. Classical MD simulates the physical movements of atoms and molecules by numerically solving Newton's equations of motion as described above. It provides insight into a system's structure and dynamic evolution by calculating the forces between particles and tracking their positions over time, making it a powerful tool for materials science, chemistry, and biology [10]. The following type of information can be obtained using Classical MD simulations-

- Structural dynamics and conformational changes, e.g., insights into protein folding, ligand binding, and large-scale motions of biomolecules.
- Thermodynamic and bulk properties, e.g., calculation of diffusion coefficients, viscosity, pressure, temperature, and material properties like elastic moduli and phase transitions.
- Solvent and ion behaviour, e.g., understanding the dynamics of water molecules and ions around solutes.
- Interaction mechanisms, e.g., elucidation of non-covalent interactions and how mutations or ligands affect protein structure and function (though without explicit electronic detail).

Ab-initio *MD (AIMD)*: Based on Density Functional Theory (DFT), it calculates the forces directly from the electronic structure of the atoms, without relying on empirical parameters. This provides a more accurate and unbiased characterization of molecular systems, especially where electronic effects are critical. However, the high computational cost limits simulations to smaller systems (hundreds of atoms) and much shorter timescales (picoseconds). It combines quantum mechanics with molecular dynamics to model chemical systems accurately by calculating atomic forces "on-the-fly" from first principles [117–119]. This makes AIMD powerful for predicting physical and chemical properties with high accuracy. The following information can be obtained from AIMD simulations-

- Direct observation of bond breaking and formation events, which is impossible with standard classical MD force fields.

 doi:10.20944/preprints202603.2311.v1

- Detailed information on charge distribution, polarization effects, and electronic transitions.
- Precise structural information, such as accurate bond angles and lengths, and the energy barriers along reaction pathways.
- Ability to incorporate quantum effects of light nuclei (like protons), which are important in processes such as proton transfer or tunnelling.
- Calculation of vibrational spectra (e.g., IR spectra) that can be directly compared to experimental results.

Machine Learning Interatomic Potentials (MLIPs): A newer approach that uses machine learning to create highly accurate potential energy functions, combining the accuracy of AIMD with the speed of classical MD. MLIPs are advanced computational models that use machine learning to accurately predict the energies and forces acting on atoms within a molecular or material system [120,121]. Instead of relying on traditional FFs or performing expensive QM calculations, such as DFT for every configuration, MLIPs learn the underlying relationship between atomic arrangements and their corresponding energies from large datasets generated using high-quality quantum calculations. These models use mathematical descriptors—such as symmetry functions, SOAP vectors, or graph-based representations—to capture the local atomic environment in a way that is independent of rotation, translation, and atom order. ML algorithms, including neural networks, Gaussian approximation methods, and graph neural networks, are then trained to reproduce the potential energy surface (PES) of the system [122,123]. Once trained, MLIPs provide energies and forces with near-quantum accuracy but at a speed comparable to classical FFs, enabling MD simulations of complex systems over long timescales. Although highly accurate and flexible, these models depend heavily on the quality and diversity of the training data and must be carefully validated to ensure reliable predictions outside their training domain.

Coarse-Grained Molecular Dynamics (CG MD): Simplifies the system by modelling groups of atoms as single "beads" or particles, reducing the number of degrees of freedom and increasing computational efficiency for a large system. CG MD is a simplified form of molecular dynamics in which groups of atoms are combined into single interaction units called beads. By reducing atomic detail, CG MD dramatically lowers computational cost, allowing simulations of very large biomolecular systems—such as membranes, protein assemblies, nucleic acids, and viral particles—over much longer timescales than all-atom MD can typically achieve. The interactions between these coarse-grained beads are described using effective potentials that are parameterized to reproduce essential structural, thermodynamic, or dynamical properties of the underlying atomistic system. Popular CG models like the MARTINI force field and UNRES provide standardized bead types and interaction rules, enabling efficient simulations of processes such as membrane fusion, protein folding, aggregation, and self-assembly [38,114]. Although CG MD sacrifices atomic-level detail and may not accurately represent fine interactions, it serves as a powerful tool for studying large-scale biological phenomena that are otherwise inaccessible to conventional atomistic simulations.

16. Recent Advancements in MD Simulations

Enhanced Sampling Techniques: To overcome the limitations of traditional MD, enhanced sampling methods have been developed. Techniques such as adaptive sampling, metadynamics, and replica exchange molecular dynamics (REMD) enable the exploration of rare events and high-dimensional free energy landscapes. Adaptive sampling utilizes ML algorithms to identify under-sampled regions of the conformational space, directing simulations to these areas to improve sampling efficiency (Kleiman et al., 2023). Simulated annealing is used to characterize very flexible systems. Earlier annealing methods were used for small protein simulations. However, a variant of the method called generalized simulated annealing can be employed for bigger macromolecular complexes [124,125].

ML algorithms have been utilized to enhance sampling in MD simulations. By predicting regions of interest in the conformational space, AI can guide simulations to explore these areas more

 doi:10.20944/preprints202603.2311.v1

efficiently. This integration allows for a more comprehensive understanding of protein dynamics and interactions, leading to more accurate predictions of drug efficacy and protein function [126].

Real-Time Feedback in Design: AI-MD integration also enables real-time feedback during the design process. As simulations progress, ML models can analyse the data to identify promising candidates and suggest modifications. This iterative process accelerates the design cycle and improves the quality of the final product [127].

Quantum-Level Simulations: The integration of QM with MD simulations holds the potential to provide more accurate descriptions of molecular interactions. QM calculations can capture electronic effects that classical methods may miss, leading to improved predictions of binding affinities and reaction mechanisms [128]. It employs QM principles and quantum computing to model protein behaviour with unprecedented accuracy. Due to the immense size and complexity of full protein systems, researchers employ fragmentation and hybrid methods to make these simulations computationally feasible [129].

Multi-Scale Modelling: Multi-scale modelling and simulation is an approach that combines multiple models and simulation methods to study complex systems across different spatial and temporal scales [130]. Combining MD simulations with other modelling techniques, such as coarse-grained models and continuum mechanics, allows for the study of larger systems over longer timescales. This multi-scale approach provides a more comprehensive understanding of biomolecular behaviour and facilitates the design of complex systems, such as drug delivery vehicles and synthetic tissues [131].

High-Throughput Screening: Advancements in computational power and algorithm efficiency enable high-throughput screening of large compound libraries using MD simulations. Coupled with ML, this approach allows for the rapid identification of potential drug candidates and accelerates the drug discovery process [132].

Standardization and Reproducibility: To ensure the reliability and reproducibility of MD simulations, there is a growing emphasis on standardizing protocols and benchmarking methods. Initiatives aimed at developing standardized workflows and reporting guidelines are essential for the consistent application of MD simulations in drug and protein design [133].

Once the system reaches a certain level of complexity, molecular simulations will inevitably contain inaccuracies [134]. They discovered, using separate tools, that while the anticipated densities were quite close, the observations frequently slipped outside of the total statistical uncertainties of the various simulations. In order to do TRUE (Transparent, Reproducible, Usable-by-others, and Extensible) simulation studies, the Molecular Simulation Design Framework (MoSDeF) workflow package was created [135].

17. Limitations of MD Simulation

MD simulation has certain drawbacks, which are covered below -

FF Dependency: A specific biomolecule system is the focus of each FF. For instance, GROMOS for proteins, AMBER for DNA, etc. Only when an appropriate force field is used in accordance with the biological samples does MD simulation yield good results. Otherwise, it will impact the outcomes [56].

Neglect of Electronic Motion: Only the particle's nuclear motion is taken into account by the classical MD; electronic motion and quantum effects are disregarded. Biomolecular MD simulation can be performed using classical MD simulation. Nevertheless, chemical processes and ion bonding are inappropriate for the traditional MD. For this reason, quantum dynamics methodologies are employed, which makes designing the simulation extremely difficult and demands a lot of processing power.

Critical frequencies: The system's coupled atoms oscillate spontaneously at the crucial frequency. At low temperatures, quantum effects are more appropriate for calculating such vibrations than classical MD.

 doi:10.20944/preprints202603.2311.v1

Time Scale: For stability, MD simulation is run in short time steps (≈ 2 fs or 2×10–15 s). Proteins, on the other hand, undergo structural alterations in nanoseconds, microseconds, milliseconds, or even longer. It goes without saying that nanosecond to millisecond events require millions to trillions of consecutive time steps. Therefore, to fully comprehend the structural changes in proteins, a very high processing power is needed.

Covalent bonds: The majority of the covalent bonds in proteins do not form or break during normal function after synthesis. Certain covalent connections do, nevertheless, form and break more frequently. For instance, hydrogen can be gained or lost from acidic or basic amino acid residues and disulfide linkages between cysteines. It is challenging to account for these occurrences in MD simulations.

Author Contributions: Conceptualization – Veerendra Kumar; methodology – Stuti, Veerendra Kumar; software - Stuti; validation – Veerendra Kumar; formal analysis – Stuti, Veerendra Kumar; investigation – Stuti, Shivani Yaduvanshi, Vansh Kashyap, Dushyant Sharma, Veerendra Kumar.; resources - NA.; data curation - NA.; writing—original draft preparation - Stuti, Veerendra Kumar; investigation – Stuti, Shivani Yaduvanshi, Vansh Kashyap, Dushyant Sharma, Veerendra Kumar.; .; visualization - Stuti, Veerendra Kumar; investigation – Stuti, Shivani Yaduvanshi, Vansh Kashyap, Dushyant Sharma, Veerendra Kumar.; supervision – Veerendra Kumar.; project administration - Veerendra Kumar.; funding acquisition - Veerendra Kumar. All authors have read and agreed to the published version of the manuscript.

Funding: We would like to acknowledge the Ramalingaswami Fellowship (BT/RLF/Re-entry/64/2017), Department of Biotechnology, Govt of India (VK). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No data were generated or analyzed during this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gsponer J, Caflisch A. Molecular dynamics simulations of protein folding from the transition state. Proc Natl Acad Sci U S A [Internet]. 2002 [cited 2025 Nov 27];99(10):6719.
2. Lane TJ, Shukla D, Beauchamp KA, et al. To milliseconds and beyond: challenges in the simulation of protein folding. Curr Opin Struct Biol [Internet]. 2013 [cited 2025 Nov 27];23(1):58–65.
3. Lu H, Zhou Q, He J, et al. Recent advances in the development of protein–protein interactions modulators: mechanisms and clinical trials. Signal Transduction and Targeted Therapy 2020 5:1 [Internet]. 2020 [cited 2022 Apr 17];5(1):1–23.
4. Chothia C, Janin J. Principles of protein-protein recognition. Nature [Internet]. 1975 [cited 2022 Apr 17];256(5520):705–708.
5. Jones S, Thornton JM. Principles of protein-protein interactions. Proc Natl Acad Sci U S A [Internet]. 1996 [cited 2022 Apr 17];93(1):13–20.
6. Titeca K, Lemmens I, Tavernier J, et al. Discovering cellular protein-protein interactions: Technological strategies and opportunities. Mass Spectrom Rev [Internet]. 2019 [cited 2022 Apr 17];38(1):79–111.
7. Scott DE, Bayly AR, Abell C, et al. Small molecules, big targets: drug discovery faces the protein–protein interaction challenge. Nature Reviews Drug Discovery 2016 15:8 [Internet]. 2016 [cited 2022 Apr 17];15(8):533–550.
8. Xing S, Wallmeroth N, Berendzen KW, et al. Techniques for the Analysis of Protein-Protein Interactions in Vivo. Plant Physiol [Internet]. 2016 [cited 2025 Dec 8];171(2):727.
9. Rizzuti B. Molecular simulations of proteins: From simplified physical interactions to complex biological phenomena. Biochimica et Biophysica Acta (BBA) - Proteins and Proteomics [Internet]. 2022 [cited 2025 Dec 8];1870(3):140757.

 doi:10.20944/preprints202603.2311.v1

10. Hollingsworth SA, Dror RO. Molecular dynamics simulation for all. Neuron [Internet]. 2018 [cited 2021 Jul 12];99(6):1129.
11. Guvench O, MacKerell AD. Comparison of protein force fields for molecular dynamics simulations. Methods Mol Biol [Internet]. 2008 [cited 2025 Dec 9];443:63–88.
12. McCammon JA, Gelin BR, Karplus M. Dynamics of folded proteins. Nature [Internet]. 1977 [cited 2025 Jul 15];267(5612):585–590.
13. Tuckerman ME, Martyna GJ. Understanding Modern Molecular Dynamics: Techniques and Applications. Journal of Physical Chemistry B [Internet]. 1999 [cited 2025 Nov 26];104(2):159–178.
14. Zhou R, Harder E, Xu H, et al. Efficient multiple time step method for use with Ewald and particle mesh Ewald for large biomolecular systems. J Chem Phys [Internet]. 2001 [cited 2025 Dec 9];115(5):2348–2358.
15. Verlet L. Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. Physical Review [Internet]. 1967 [cited 2025 Dec 9];159(1):98.
16. Van Gunsteren WF, Berendsen HJC. A Leap-frog Algorithm for Stochastic Dynamics. Mol Simul [Internet]. 1988 [cited 2025 Dec 9];1(3):173–185.
17. Braun E, Gilmer J, Mayes HB, et al. Best Practices for Foundations in Molecular Simulations [Article v1.0]. Living J Comput Mol Sci [Internet]. 2018 [cited 2025 Dec 9];1(1):5957.
18. Hsu T, Sadigh B, Bulatov V, et al. Score Dynamics: Scaling Molecular Dynamics with Picoseconds Time Steps via Conditional Diffusion Model. J Chem Theory Comput [Internet]. 2024 [cited 2025 Dec 9];20(6):2335–2348.
19. Hopkins CW, Le Grand S, Walker RC, et al. Long-Time-Step Molecular Dynamics through Hydrogen Mass Repartitioning. J Chem Theory Comput [Internet]. 2015 [cited 2025 Dec 9];11(4):1864–1874.
20. Hess B, Bekker H, Berendsen HJC, et al. LINCS: A linear constraint solver for molecular simulations. J Comput Chem [Internet]. 1997 [cited 2025 Dec 9]; doi: 10.1002/(SICI)1096-987X(199709)18:12.
21. Yoneya M, Hirasawa K, Berendsen HJC. A Non-Iterative Matrix Method for Constraint Molecular Dynamics Simulations. Mol Simul [Internet]. 1994 [cited 2025 Dec 9];13(6):395–405.
22. Sahil M, Sarkar S, Mondal J. Long-time-step molecular dynamics can retard simulation of protein-ligand recognition process. Biophys J [Internet]. 2023 [cited 2025 Dec 9];122(5):802–816.
23. Miyamoto S, Kollman PA. Settle: An analytical version of the SHAKE and RATTLE algorithm for rigid water models. J Comput Chem [Internet]. 1992 [cited 2025 Dec 9];13(8):952–962.
24. Macuglia D. SHAKE and the exact constraint satisfaction of the dynamics of semi-rigid molecules in Cartesian coordinates, 1973–1977. Archive for History of Exact Sciences 2023 77:4 [Internet]. 2023 [cited 2025 Nov 27];77(4):345–371.
25. Hess B, Bekker H, Berendsen HJC, et al. LINCS: A linear constraint solver for molecular simulations. J Comput Chem [Internet]. 1997 [cited 2025 Nov 27];18(12):1463–1472.
26. Andersen HC. Rattle: A "velocity" version of the shake algorithm for molecular dynamics calculations. J Comput Phys [Internet]. 1983 [cited 2025 Dec 9];52(1):24–34.
27. Yang JZ, Li X. Comparative study of boundary conditions for molecular dynamics simulations of solids at low temperature. Phys Rev B [Internet]. 2006 [cited 2025 Dec 9];73(22):224111.
28. Kuzkin VA. On angular momentum balance for particle systems with periodic boundary conditions. ZAMM Zeitschrift fur Angewandte Mathematik und Mechanik [Internet]. 2015 [cited 2025 Dec 9];95(11):1290–1295.
29. Wells BA, Chaffee AL. Ewald Summation for Molecular Simulations. J Chem Theory Comput [Internet]. 2015 [cited 2025 Dec 9];11(8):3684–3695.
30. Cheatham TE, Miller JL, Fox T, et al. Molecular Dynamics Simulations on Solvated Biomolecular Systems: The Particle Mesh Ewald Method Leads to Stable Trajectories of DNA, RNA, and Proteins. J Am Chem Soc [Internet]. 2002 [cited 2025 Dec 9];117(14):4193–4194.
31. Kadaoluwa Pathirannahalage SP, Meftahi N, Elbourne A, et al. Systematic Comparison of the Structural and Dynamic Properties of Commonly Used Water Models for Molecular Dynamics Simulations. J Chem Inf Model [Internet]. 2021 [cited 2025 Nov 27];61(9):4521–4536.
32. Izadi S, Onufriev A V. Accuracy limit of rigid 3-point water models. J Chem Phys [Internet]. 2016 [cited 2025 Nov 27];145(7).

 doi:10.20944/preprints202603.2311.v1

33. Zhang J, Zhang H, Wu T, et al. Comparison of Implicit and Explicit Solvent Models for the Calculation of Solvation Free Energy in Organic Solvents. J Chem Theory Comput [Internet]. 2017 [cited 2026 Jan 7];13(3):1034–1043.
34. Skyner RE, McDonagh JL, Groom CR, et al. A review of methods for the calculation of solution free energies and the modelling of systems in solution. Physical Chemistry Chemical Physics [Internet]. 2015 [cited 2025 Dec 10];17(9):6174–6191.
35. Heinz H, Lin TJ, Kishore Mishra R, et al. Thermodynamically consistent force fields for the assembly of inorganic, organic, and biological nanostructures: The INTERFACE force field. Langmuir [Internet]. 2013 [cited 2022 Apr 15];29(6):1754–1765.
36. Sun H, Mumby SJ, Maple JR, et al. An ab Initio CFF93 All-Atom Force Field for Polycarbonates. J Am Chem Soc [Internet]. 2002 [cited 2022 Apr 17];116(7):2978–2987.
37. Wang J, Wolf RM, Caldwell JW, et al. Development and testing of a general amber force field. J Comput Chem [Internet]. 2004 [cited 2022 Apr 17];25(9):1157–1174.
38. Emperador A, Guàrdia E. Accurate coarse grained models for protein association and recognition. Adv Protein Chem Struct Biol [Internet]. 2025 [cited 2025 Dec 9];145:1–21.
39. Jorgensen WL, Tirado-Rives J. Potential energy functions for atomic-level simulations of water and organic and biomolecular systems. Proc Natl Acad Sci U S A [Internet]. 2005 [cited 2022 May 30];102(19):6665–6670.
40. MacKerell AD, Bashford D, Bellott M, et al. All-atom empirical potential for molecular modeling and dynamics studies of proteins. J Phys Chem B [Internet]. 1998 [cited 2022 May 30];102(18):3586–3616.
41. Prasad K, Nikzad M, Nisha SS, et al. On the Use of Molecular Dynamics Simulations for Elucidating Fine Structural, Physico-Chemical and Thermomechanical Properties of Lignocellulosic Systems: Historical and Future Perspectives. Journal of Composites Science 2021, Vol 5, Page 55 [Internet]. 2021 [cited 2025 Nov 27];5(2):55.
42. Robertson MJ, Tirado-Rives J, Jorgensen WL. Improved Peptide and Protein Torsional Energetics with the OPLSAA Force Field. J Chem Theory Comput [Internet]. 2015 [cited 2025 Nov 27];11(7):3499–3509.
43. Swindells N, Sykes C. The classical equation of state of gaseous helium, neon and argon. Proc R Soc Lond A Math Phys Sci [Internet]. 1938 [cited 2025 Nov 27];168(933):264–283.
44. Ángyán J, Dobson J, Jansen G, et al. Introduction. RSC Theoretical and Computational Chemistry Series [Internet]. 2020 [cited 2025 Nov 27];2020-January(16):1–434.
45. Geidl S, Bouchal T, Raček T, et al. High-quality and universal empirical atomic charges for chemoinformatics applications. Journal of Cheminformatics 2015 7:1 [Internet]. 2015 [cited 2025 Nov 27];7(1):59-.
46. Dauber-Osguthorpe P, Hagler AT. Biomolecular force fields: where have we been, where are we now, where do we need to go and how do we get there? Journal of Computer-Aided Molecular Design 2018 33:2 [Internet]. 2018 [cited 2025 Dec 10];33(2):133–203.
47. Chen KH, Allinger NL. Molecular mechanics (MM4) study of saturated four-membered ring hydrocarbons. Journal of Molecular Structure: THEOCHEM [Internet]. 2002 [cited 2025 Dec 10];581(1–3):215–237.
48. Rappé AK, Casewit CJ, Colwell KS, et al. UFF, a Full Periodic Table Force Field for Molecular Mechanics and Molecular Dynamics Simulations. J Am Chem Soc. 1992;114(25):10024–10035.
49. Mayo SL, Olafson BD, Goddard WA. DREIDING: A generic force field for molecular simulations. Journal of Physical Chemistry. 1990;94(26):8897–8909.
50. Brooks BR, Bruccoleri RE, Olafson BD, et al. CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. J Comput Chem [Internet]. 1983 [cited 2025 Dec 10];4(2):187–217.
51. Bayly CI, Merz KM, Ferguson DM, et al. A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules. J Am Chem Soc [Internet]. 1995 [cited 2025 Dec 10];117(19):5179–5197.
52. Schuler LD, Daura X, Van Gunsteren WF. An improved GROMOS96 force field for aliphatic hydrocarbons in the condensed phase. J Comput Chem. 2001;22(11):1205–1218.
53. Jorgensen WL, Tirado-Rives J. The OPLS Force Field for Proteins. Energy Minimizations for Crystals of Cyclic Peptides and Crambin. J Am Chem Soc. 1988;110(6):1657–1666.

 doi:10.20944/preprints202603.2311.v1

54. Sun H. COMPASS: An ab Initio Force-Field Optimized for Condensed-Phase ApplicationsOverview with Details on Alkane and Benzene Compounds. Journal of Physical Chemistry B [Internet]. 1998 [cited 2025 Dec 10];102(38):7338–7364.
55. Vanommeslaeghe K, Mackerell AD. CHARMM additive and polarizable force fields for biophysics and computer-aided drug design. Biochim Biophys Acta [Internet]. 2014 [cited 2025 Nov 27];1850(5):861.
56. González MA. Force fields and molecular dynamics simulations. Collection SFN. 2011;12:169–200.
57. Vanommeslaeghe K, Guvench O, Alexander D MacKerell J. Molecular Mechanics. Curr Pharm Des [Internet]. 2014 [cited 2025 Nov 27];20(20):3281.
58. Maple JR, Hwang M -J, Stockfisch TP, et al. Derivation of class II force fields. I. Methodology and quantum force field for the alkyl functional group and alkane molecules. J Comput Chem [Internet]. 1994 [cited 2025 Dec 10];15(2):162–182.
59. Ponder JW, Wu C, Ren P, et al. Current Status of the AMOEBA Polarizable Force Field. Journal of Physical Chemistry B [Internet]. 2010 [cited 2025 Dec 29];114(8):2549–2564.
60. Lemkul JA, Huang J, Roux B, et al. An Empirical Polarizable Force Field Based on the Classical Drude Oscillator Model: Development History and Recent Applications. Chem Rev [Internet]. 2016 [cited 2025 Dec 29];116(9):4983–5013.
61. Lin FY, MacKerell AD. Force fields for small molecules. Methods Mol Biol [Internet]. 2019 [cited 2025 Nov 27];2022:21.
62. Van Duin ACT, Dasgupta S, Lorant F, et al. ReaxFF: A reactive force field for hydrocarbons. Journal of Physical Chemistry A. 2001;105(41):9396–9409.
63. Senftle TP, Hong S, Islam MM, et al. The ReaxFF reactive force-field: development, applications and future directions. npj Computational Materials 2016 2:1 [Internet]. 2016 [cited 2025 Dec 10];2(1):15011-.
64. Mao Q, Feng M, Jiang XZ, et al. Classical and reactive molecular dynamics: Principles and applications in combustion and energy systems. Prog Energy Combust Sci [Internet]. 2023 [cited 2025 Nov 27];97:101084.
65. Pandey M, Fernandez M, Gentile F, et al. The transformational role of GPU computing and deep learning in drug discovery. Nature Machine Intelligence 2022 4:3 [Internet]. 2022 [cited 2025 Dec 10];4(3):211–221.
66. Kang P, Kang P. Evaluation of GPU-Accelerated Edge Platforms for Stochastic Simulations: Performance and Energy Efficiency Analysis. Mathematics 2025, Vol 13, [Internet]. 2025 [cited 2025 Dec 10];13(20).
67. Stone JE, Hardy DJ, Ufimtsev IS, et al. GPU-accelerated molecular modeling coming of age. J Mol Graph Model [Internet]. 2010 [cited 2025 Nov 27];29(2):116–125.
68. Biagini T, Petrizzelli F, Truglio M, et al. Are Gaming-Enabled Graphic Processing Unit Cards Convenient for Molecular Dynamics Simulation? Evolutionary Bioinformatics. 2019;15.
69. Karunadasa NP, Ranasinghe DN. Accelerating high performance applications with CUDA and MPI. ICIIS 2009 - 4th International Conference on Industrial and Information Systems 2009, Conference Proceedings. 2009;331–336.
70. Biagini T, Chillemi G, Mazzoccoli G, et al. Molecular dynamics recipes for genome research. Brief Bioinform [Internet]. 2018 [cited 2022 Apr 16];19(5):853–862.
71. Biagini T, Petrizzelli F, Truglio M, et al. Are Gaming-Enabled Graphic Processing Unit Cards Convenient for Molecular Dynamics Simulation? Evol Bioinform Online [Internet]. 2019 [cited 2022 Apr 16];15.
72. Adjoua O, Lagardère L, Jolly LH, et al. Tinker-HP: Accelerating Molecular Dynamics Simulations of Large Complex Systems with Advanced Point Dipole Polarizable Force Fields Using GPUs and Multi-GPU Systems. J Chem Theory Comput [Internet]. 2021 [cited 2025 Dec 10];17(4):2034–2053.
73. Musaelian A, Johansson A, Batzner S, et al. Scaling the leading accuracy of deep equivariant models to biomolecular simulations of realistic size. 2023;
74. Lawrence R, Chakravorty DK, Perez LM, et al. Performance of Molecular Dynamics Acceleration Strategies on Composable Cyberinfrastructure. PEARC 2024 - Practice and Experience in Advanced Research Computing 2024: Human Powered Computing [Internet]. 2024 [cited 2025 Dec 10]; doi: 10.1145/3626203.3670631;WGROUP:STRING:ACM.
75. Lastovetsky A, Manumachu RR. Energy-Efficient Parallel Computing: Challenges to Scaling. Information (Switzerland). 2023;14(4).

 doi:10.20944/preprints202603.2311.v1

76. Thanh PTH, Tran TT, Nguyen TH, et al. ULDepth: Transform Self-Supervised Depth Estimation to Unpaired Multi-Domain Learning. IEEE Open Journal of Signal Processing. 2025;6:1004–1016.
77. Salsbury FR. Molecular Dynamics Simulations of Protein Dynamics and their relevance to drug discovery. Curr Opin Pharmacol [Internet]. 2010 [cited 2025 Nov 27];10(6):738.
78. Abraham MJ, Murtola T, Schulz R, et al. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. SoftwareX [Internet]. 2015 [cited 2025 Dec 10];1–2:19–25.
79. Pronk S, Páll S, Schulz R, et al. GROMACS 4.5: a high-throughput and highly parallel open source molecular simulation toolkit. Bioinformatics [Internet]. 2013 [cited 2025 Nov 27];29(7):845–854.
80. Pronk S, Páll S, Schulz R, et al. GROMACS 4.5: a high-throughput and highly parallel open source molecular simulation toolkit. Bioinformatics [Internet]. 2013 [cited 2025 Dec 10];29(7):845–854.
81. Thompson AP, Aktulga HM, Berger R, et al. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. Comput Phys Commun. 2022;271.
82. Singraber A, Behler J, Dellago C. Library-Based LAMMPS Implementation of High-Dimensional Neural Network Potentials. J Chem Theory Comput [Internet]. 2019 [cited 2025 Dec 10];15(3):1827–1840.
83. Case DA, Aktulga HM, Belfon K, et al. AmberTools. J Chem Inf Model [Internet]. 2023 [cited 2025 Dec 11];63(20):6183.
84. Weiner PK, Kollman PA. AMBER: Assisted model building with energy refinement. A general program for modeling molecules and their interactions. J Comput Chem [Internet]. 1981 [cited 2025 Dec 11];2(3):287–303.
85. Case DA, Cerutti DS, Cruzeiro VWD, et al. Recent Developments in Amber Biomolecular Simulations. J Chem Inf Model [Internet]. 2025 [cited 2025 Dec 11];65(15):7835.
86. Case DA, Cheatham TE, Darden T, et al. The Amber Biomolecular Simulation Programs. J Comput Chem [Internet]. 2005 [cited 2025 Nov 27];26(16):1668.
87. Melo MCR, Bernardi RC, Rudack T, et al. NAMD goes quantum: an integrative suite for hybrid simulations. Nat Methods [Internet]. 2018 [cited 2025 Dec 11];15(5):351–354.
88. Melo MCR, Bernardi RC, Rudack T, et al. NAMD goes quantum: An integrative suite for QM/MM simulations. Nat Methods [Internet]. 2018 [cited 2025 Dec 11];15(5):351.
89. Phillips JC, Braun R, Wang W, et al. Scalable molecular dynamics with NAMD. J Comput Chem [Internet]. 2005 [cited 2025 Nov 27];26(16):1781–1802.
90. Brooks BR, Brooks CL, Mackerell AD, et al. CHARMM: The biomolecular simulation program. J Comput Chem [Internet]. 2009 [cited 2025 Dec 11];30(10):1545–1614.
91. Brooks BR, Bruccoleri RE, Olafson BD, et al. CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. J Comput Chem [Internet]. 1983 [cited 2025 Dec 11];4(2):187–217.
92. Lee J, Cheng X, Swails JM, et al. CHARMM-GUI Input Generator for NAMD, GROMACS, AMBER, OpenMM, and CHARMM/OpenMM Simulations Using the CHARMM36 Additive Force Field. J Chem Theory Comput. 2016;12(1):405–413.
93. Jo S, Kim T, Iyer VG, et al. CHARMM-GUI: A web-based graphical user interface for CHARMM. J Comput Chem. 2008; doi: 10.1002/jcc.20945.
94. Wang T, He X, Li M, et al. Ab initio characterization of protein molecular dynamics with AI2BMD. Nature. 2024;635(8040):1019–1027.
95. Martin J, Frezza E. A dynamical view of protein-protein complexes: Studies by molecular dynamics simulations. Front Mol Biosci [Internet]. 2022 [cited 2025 Nov 27];9:970109.
96. Yaduvanshi S, Kumar V. Interaction studies between calmodulin and gating brake peptides from T-type channels. Frontiers in Biophysics [Internet]. 2025 [cited 2025 Dec 11];3:1569091.
97. Sriramulu DK, Lee SG. Analysis of protein-protein interface with incorporating low-frequency molecular interactions in molecular dynamics simulation. J Mol Graph Model [Internet]. 2023 [cited 2025 Dec 11];122:108461.
98. Jost Lopez A, Quoika PK, Linke M, et al. Quantifying Protein–Protein Interactions in Molecular Simulations. J Phys Chem B [Internet]. 2020 [cited 2025 Nov 27];124(23):4673.

 doi:10.20944/preprints202603.2311.v1

99. Pan AC, Jacobson D, Yatsenko K, et al. Atomic-level characterization of protein–protein association. Proc Natl Acad Sci U S A [Internet]. 2019 [cited 2025 Nov 27];116(10):4244–4249.
100. Voelz VA, Pande VS, Bowman GR. Folding@home: Achievements from over 20 years of citizen science herald the exascale era. Biophys J [Internet]. 2023 [cited 2025 Dec 29];122(14):2852–2863.
101. Galano-Frutos JJ, Nerín-Fonz F, Sancho J. Calculation of Protein Folding Thermodynamics Using Molecular Dynamics Simulations. J Chem Inf Model [Internet]. 2023 [cited 2025 Dec 11];63(24):7791–7806.
102. Freddolino PL, Liu F, Gruebele M, et al. Ten-Microsecond Molecular Dynamics Simulation of a Fast-Folding WW Domain. Biophys J [Internet]. 2008 [cited 2025 Nov 27];94(10):L75.
103. Duan L, Guo X, Cong Y, et al. Accelerated Molecular Dynamics Simulation for Helical Proteins Folding in Explicit Water. Front Chem [Internet]. 2019 [cited 2025 Nov 27];7:428425.
104. Mardikoraem M, Wang Z, Pascual N, et al. Generative models for protein sequence modeling: recent advances and future directions. Brief Bioinform [Internet]. 2023 [cited 2025 Dec 11];24(6):1–19.
105. Wang M, Ma A, Wang H, et al. Atomic molecular dynamics simulation advances of de novo-designed proteins. Q Rev Biophys [Internet]. 2024 [cited 2025 Dec 11];57:e14.
106. Wang J, Liu Z, Zhao S, et al. Deep Learning Empowers the Discovery of Self-Assembling Peptides with Over 10 Trillion Sequences. Adv Sci (Weinh) [Internet]. 2023 [cited 2025 Nov 27];10(31).
107. Kurniawan J, Ishida T. Protein Model Quality Estimation Using Molecular Dynamics Simulation. ACS Omega [Internet]. 2022 [cited 2025 Nov 27];7(28):24274.
108. Sun Y, Jiao Y, Shi C, et al. Deep learning-based molecular dynamics simulation for structure-based drug design against SARS-CoV-2. Comput Struct Biotechnol J [Internet]. 2022 [cited 2025 Nov 27];20:5014–5027.
109. Wang X, Wu X, Brooks BR, et al. Accurate Free Energy Calculation via Multiscale Simulations Driven by Hybrid Machine Learning and Molecular Mechanics Potentials. J Chem Theory Comput [Internet]. 2025 [cited 2025 Dec 11];21(14):6979–6987.
110. Samoudi M, Tabandeh F, Minuchehr Z, et al. Rational design of hyper-glycosylated interferon beta analogs: A computational strategy for glycoengineering. J Mol Graph Model [Internet]. 2015 [cited 2025 Dec 11];56:31–42.
111. Samoudi M, Minuchehr Z, Harcum SW, et al. Rational design of glycoengineered interferon-β analogs with improved aggregation state: experimental validation. Protein Engineering, Design and Selection [Internet]. 2017 [cited 2025 Dec 11];30(1):23–30.
112. Yang C, Lu D, Liu Z. How PEGylation enhances the stability and potency of insulin: a molecular dynamics simulation. Biochemistry [Internet]. 2011 [cited 2025 Nov 27];50(13):2585–2593.
113. Tuckerman ME, Martyna GJ. Understanding Modern Molecular Dynamics: Techniques and Applications. Journal of Physical Chemistry B [Internet]. 1999 [cited 2025 Dec 11];104(2):159–178.
114. Stevens MJ. Coarse-grained simulations of lipid bilayers. J Chem Phys [Internet]. 2004 [cited 2025 Dec 11];121(23):11942–11948.
115. Jász Á, Rák Á, Ladjánszki I, et al. Classical molecular dynamics on graphics processing unit architectures. Wiley Interdiscip Rev Comput Mol Sci [Internet]. 2020 [cited 2025 Dec 9];10(2):e1444.
116. Hug S. Classical molecular dynamics in a nutshell. Methods Mol Biol [Internet]. 2013 [cited 2025 Dec 11];924:127–152.
117. Tse JS. Ab initio molecular dynamics with density functional theory. Annu Rev Phys Chem [Internet]. 2002 [cited 2025 Dec 11];53(Volume 53, 2002):249–290.
118. Boek ES, Sprik M. Ab Initio Molecular Dynamics Study of the Hydration of a Sodium Smectite Clay. Journal of Physical Chemistry B [Internet]. 2003 [cited 2025 Dec 11];107(14):3251–3256.
119. Iftimie R, Minary P, Tuckerman ME. Ab initio molecular dynamics: Concepts, recent developments, and future trends. Proc Natl Acad Sci U S A [Internet]. 2005 [cited 2025 Dec 11];102(19):6654–6659.
120. Qi J, Ko TW, Wood BC, et al. Robust training of machine learning interatomic potentials with dimensionality reduction and stratified sampling. npj Computational Materials 2024 10:1 [Internet]. 2024 [cited 2025 Dec 11];10(1):43-.
121. Kulichenko M, Nebgen B, Lubbers N, et al. Data Generation for Machine Learning Interatomic Potentials and Beyond. Chem Rev [Internet]. 2024 [cited 2025 Dec 11];124(24):13681–13714.

 doi:10.20944/preprints202603.2311.v1

122. Bartók AP, Payne MC, Kondor R, et al. Gaussian Approximation Potentials: The Accuracy of Quantum Mechanics, without the Electrons. Phys Rev Lett [Internet]. 2010 [cited 2025 Dec 11];104(13):136403.
123. Szlachta WJ, Bartók AP, Csányi G. Accuracy and transferability of Gaussian approximation potential models for tungsten. Phys Rev B [Internet]. 2014 [cited 2025 Dec 11];90(10):104108.
124. Kulichenko M, Nebgen B, Lubbers N, et al. Data Generation for Machine Learning Interatomic Potentials and Beyond. Chem Rev [Internet]. 2024 [cited 2025 Dec 11];124(24):13681–13714.
125. Bernardi RC, Melo MCR, Schulten K. Enhanced Sampling Techniques in Molecular Dynamics Simulations of Biological Systems. Biochim Biophys Acta [Internet]. 2014 [cited 2025 Dec 11];1850(5):872.
126. Dao TM, Rahman T, Dao TM, et al. Deep Generative Modeling of Protein Conformations: A Comprehensive Review. BioChem 2025, Vol 5, [Internet]. 2025 [cited 2025 Dec 11];5(3).
127. Visan AI, Negut I, Visan AI, et al. Integrating Artificial Intelligence for Drug Discovery in the Context of Revolutionizing Drug Delivery. Life 2024, Vol 14, [Internet]. 2024 [cited 2025 Dec 11];14(2).
128. König G, Hudson PS, Boresch S, et al. Multiscale Free Energy Simulations: An Efficient Method for Connecting Classical MD Simulations to QM or QM/MM Free Energies Using Non-Boltzmann Bennett Reweighting Schemes. J Chem Theory Comput [Internet]. 2014 [cited 2025 Dec 11];10(4):1406–1419.
129. Atchade-Adelomou P, Sala LC, Atchade-Adelomou P, et al. A Quantum Strategy for the Simulation of Large Proteins: From Fragmentation in Small Proteins to Scalability in Complex Systems. Electronics 2025, Vol 14, [Internet]. 2025 [cited 2025 Dec 11];14(13).
130. Barbhuiya S, Jivkov A, Das BB. A review of multi-scale modelling of concrete deterioration: Fundamentals, techniques and perspectives. Constr Build Mater [Internet]. 2023 [cited 2025 Dec 11];406:133472.
131. Poma AB, Hinostroza Caldas A, Cofas-Vargas LF, et al. Recent advances in machine learning and coarse-grained potentials for biomolecular simulations. Biophys J [Internet]. 2025 [cited 2025 Dec 11]; doi: 10.1016/J.BPJ.2025.06.019.
132. Sadybekov A V., Katritch V. Computational approaches streamlining drug discovery. Nature 2023 616:7958 [Internet]. 2023 [cited 2025 Dec 11];616(7958):673–685.
133. Caparotta M, Perez A. Advancing Molecular Dynamics: Toward Standardization, Integration, and Data Accessibility in Structural Biology. J Phys Chem B [Internet]. 2024 [cited 2025 Dec 11];128(10):2219–2227.
134. Schappals M, Mecklenfeld A, Kröger L, et al. Round Robin Study: Molecular Simulation of Thermodynamic Properties from Models with Internal Degrees of Freedom. J Chem Theory Comput [Internet]. 2017 [cited 2025 Dec 11];13(9):4270–4280.
135. Craven NC, Singh R, Quach CD, et al. Achieving Reproducibility and Replicability of Molecular Dynamics and Monte Carlo Simulations Using the Molecular Simulation Design Framework (MoSDeF). J Chem Eng Data [Internet]. 2025 [cited 2025 Dec 11];70(6):2178–2199.
136. Taylor RS, Dang LX, Garrett BC. Molecular Dynamics Simulations of the Liquid/Vapor Interface of SPC/E Water. Journal of Physical Chemistry [Internet]. 1996 [cited 2026 Jan 8];100(28):11720–11725.
137. Lee SH. Temperature dependence of the thermal conductivity of water: A molecular dynamics simulation study using the SPC/E model. Mol Phys [Internet]. 2014 [cited 2026 Jan 8];112(16):2155–2159.
138. Rosas JimEnez JG, Fábián B, Hummer G. Faster Sampling in Molecular Dynamics Simulations with TIP3P-F Water. J Chem Theory Comput [Internet]. 2024 [cited 2026 Jan 8];20(24):11068–11081.
139. Neumann M. Dielectric relaxation in water. Computer simulations with the TIP4P potential. J Chem Phys. 1985;85(3):1567–1580.
140. Wang LP, Martinez TJ, Pande VS. Building Force Fields: An Automatic, Systematic, and Reproducible Approach. Journal of Physical Chemistry Letters [Internet]. 2014 [cited 2026 Jan 8];5(11):1885–1891.
141. Horn HW, Swope WC, Pitera JW, et al. Development of an improved four-site water model for biomolecular simulations: TIP4P-Ew. J Chem Phys [Internet]. 2004 [cited 2026 Jan 8];120(20):9665–9678.
142. Pan Z, Mu J, Chen HF. Balanced Three-Point Water Model OPC3-B for Intrinsically Disordered and Ordered Proteins. J Chem Theory Comput [Internet]. 2023 [cited 2026 Jan 8];19(15):4837–4850.
143. Gygi F. A single-GPU implementation of first-principles molecular dynamics. J Chem Phys [Internet]. 2025 [cited 2026 Jan 8];163(15).

 doi:10.20944/preprints202603.2311.v1

144. Boku T, Sugita M, Kobayashi R, et al. Improving Performance on Replica-Exchange Molecular Dynamics Simulations by Optimizing GPU Core Utilization. ACM International Conference Proceeding Series [Internet]. 2024 [cited 2026 Jan 8];1082–1091.
145. Cui S, Patke A, Nguyen H, et al. Story of Two GPUs: Characterizing the Resilience of Hopper H100 and Ampere A100 GPUs. 2025 [cited 2026 Jan 8];1.
146. Galvelis R, Varela-Rial A, Doerr S, et al. NNP/MM: Accelerating molecular dynamics simulations with machine learning potentials and molecular mechanics. J Chem Inf Model [Internet]. 2023 [cited 2026 Jan 8];63(18):5701.