Prof. Dr. rer. nat. habil. Martin O. Steinhauser

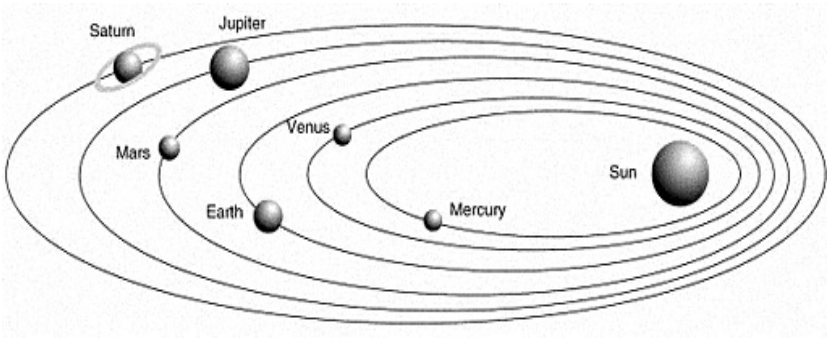
Frankfurt University of Applied Sciences, Germany

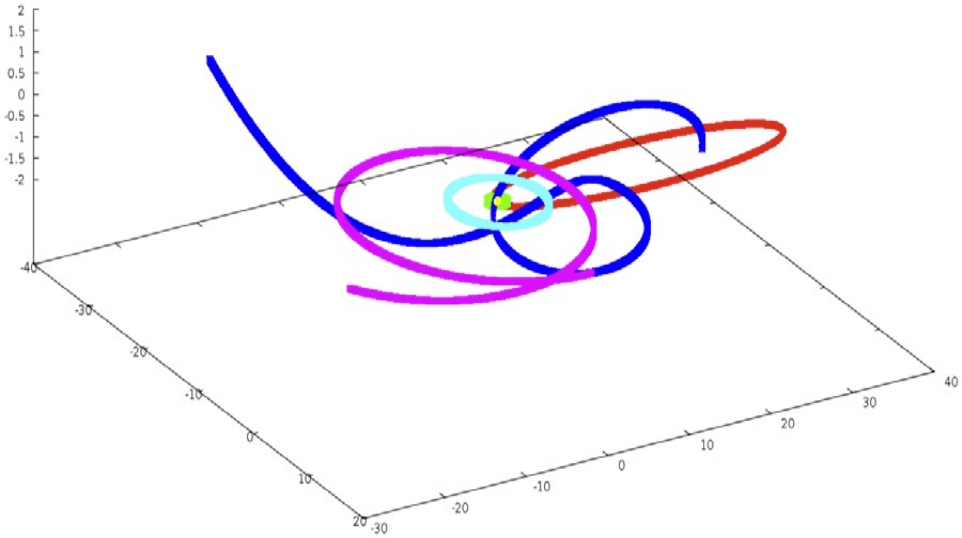
Faculty of Computer Science and Engineering

Short Lecture Course:

Introduction to Computational Science with Applications in Molecular Dynamics

Session 3: The Molecular Dynamics Method





Overview of this short course

Topics Covered (subject to change)

- 1st Session: Lec. 1-2 — Introduction & Bits and Bytes
- 2nd Session: Lec 3 (2x) — Bits and Bytes continued
- 3rd Session: Lec 4-6 — Molecular Dynamics (MD)
- 4th Session: Lec 7-8 — Problem of Sorting
- 5th Session: Lec 9 (2x) — Problem of Sorting
- 6th Session: Lec 10-11 — Monte Carlo/Statistical Physics
- 8th Session: Lec 12-13 — Monte Carlo/Random Numbers

Session 3: Overview

OUTLINE OF LECTURE

- What is the MD method?
- Newtonian/Lagrangian/Hamiltonian Dynamics
- A Molecular Dynamics Program: Planetary Motion

- **Handout 4:** Introduction to Molecular Dynamics Simulations

  (Original Publication by M. O. Steinhauser)

- **Handout 5:** C-Code: PMC.zip

To download lecture material, please go to Github:

https://github.com/Kosmokrat/JapanLecture2024

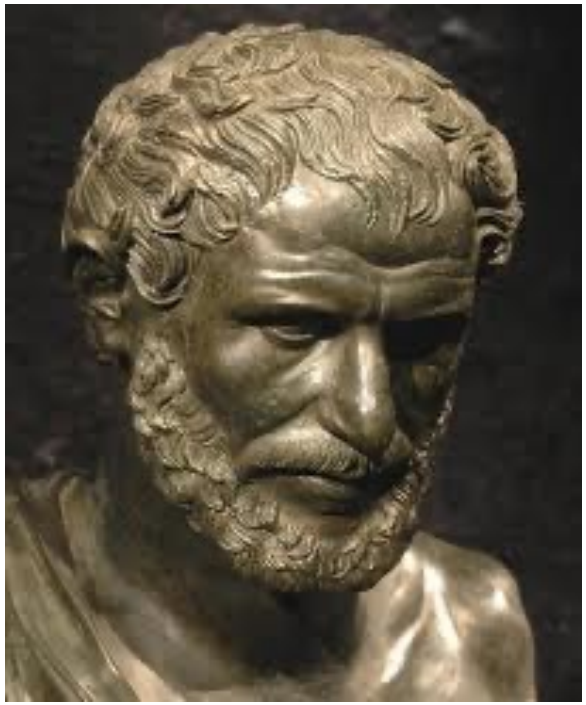
Session 3

1 The Molecular Dynamics Method

What is Molecular Dynamics?

You cannot step twice in the same river

Heraclitus



What is Molecular Dynamics?

- MD follows Newton‘s classical equations of motion (EOM) for a system of N interacting nuclei (or particles)
- MD *numerically solves* the classical EOM (Newton)
- MD keeps track of the trajectories of N particles in a system at finite temperature
- In MD, one is usually most interested in the *avaraged thermodynamic properties* of a N-body system
- MD was *first introduced* for the study of liquids for which an analytical theory is extremely difficult to be formulated
  (1964 landmark paper by Rahman)

Remark:

- Solid states are *periodic* and thus simpler in the theoretical treatment than liquids. For liquids, long-range disorder is an essential part of the system.

Newton‘s Equations and Laplace‘s Demon

- Follow the dynamics (the motion) of all the atoms in your material
- Numerically solve classical equations of motion (Newton)

$$m_i \frac{d^2\vec{r}}{dt^2} = \vec{F}_i\left(\vec{r}_1, \cdots, \vec{r}_N\right)$$

LAPLACE:

Nous devons donc envisager l'état présent de l'universe comme l'effet de son état antérieur et comme la cause de delui qui va suivre. Une intelligence qui, pour un instant donné, connaîtrait toutes les forces dont la nature est animée et la situation respective des êtres qui las composent, si d'ailleurs elle était assez vaste pour soumettre ces données à l'Analyse, embrasserait dans la même formule les mouvements des plus grands corps de l'univers et ceux du plus lèger atome : rien ne serait incertain pour elle, et l'avenir, comme le passé, serait présent à ses yeux.

Calculating the Forces

- Forces on the atoms come from the interaction with other atoms:

Total potential energy
(from QM or interatomic potentials)

Approximated
(in almost all cases)

$$\vec{F}\left(\vec{r}_1, \cdots \vec{r}_N\right) = -\vec{\nabla}_{\vec{r}_i} \Phi\left(\{\vec{r}_j\}\right)$$

Solve Differential Equations:

- One needs *initial conditions* and the EOMs
- An absolute *deterministic view* of the physical world

Review of Classical Mechanics: Newton

- Different formulations of classical mechanics:
- **Newton:** Direct description of a mechanical system in position space
- **Equations of Motion:** $\vec{F}_i = \sum_{i=1}^{N} m_i \ddot{\vec{r}}_i = \sum_{i=1}^{N} \dot{\vec{p}}_i = -\vec{\nabla}_{\vec{r}_i} \phi\left(\left\{\vec{r}_i\right\}\right)$

Remark:

- If a potential exists, the system is called *conservative:* $\oint_{curve} \vec{F}\, \mathrm{d}\vec{s} = 0$
- Important, because here, the total energy is *conserved:*

$$E = \sum_{i=1}^{N} 1/2\, m_i \vec{v}_i^{\,2} + \phi\left(\vec{r}_i\right) \Rightarrow \frac{dE}{dt} = \sum_{i=1}^{N} m_i \vec{v}_i \dot{\vec{v}}_i + \frac{d\phi}{dt}$$

$$= \sum_{i=1}^{N} m_i \dot{\vec{v}}_i \vec{v}_i - \vec{F}_i \vec{v}_i = 0, \text{ because } \frac{d\phi}{dt} = \frac{d\phi}{d\vec{r}_i} \frac{d\vec{r}_i}{dt} = F_i \vec{v}_i$$

Review of Classical Mechanics: Lagrange

- **Lagrange:** There exists a function $L = L\left(\dot{q}_i, q_i, t\right)$

  for which the following *variational principle* holds:

$$I = \int_{t_0}^{t} L\left(\dot{q}_i, q_i, t\right) dt = 0$$

- **Equations of Motion: Lagrange Equations of the 2nd kind:**

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = 0$$

Remark: $L = L\left(\dot{q}_i, q_i, t\right) = E_{kin} - E_{pot} = K\left(\dot{q}_i, q_i, t\right) - \phi\left(q_i\right)$

- Advantage of Lagrange formulation: L can be formulated in *any system* using generalized velocities $\dot{q}_i$ and coordinates q_i .

Review of Classical Mechanics: Hamilton

- Reformulation of classical mechanics (1800‘s):

- **Hamiltonian:** Description of a mechanical system in 6N-dim. phase space

  There exists a function $H\left(p_i, q_i, t\right) = \sum_{i=1}^{3N} p_i q_i - L\left(\dot{q}_i, q_i, t\right)$

  where $p_i = \frac{\partial L}{\partial \dot{q}_i}$ is the *generalized momentum*

- **Equations of Motion** are the *canonical* EOM:

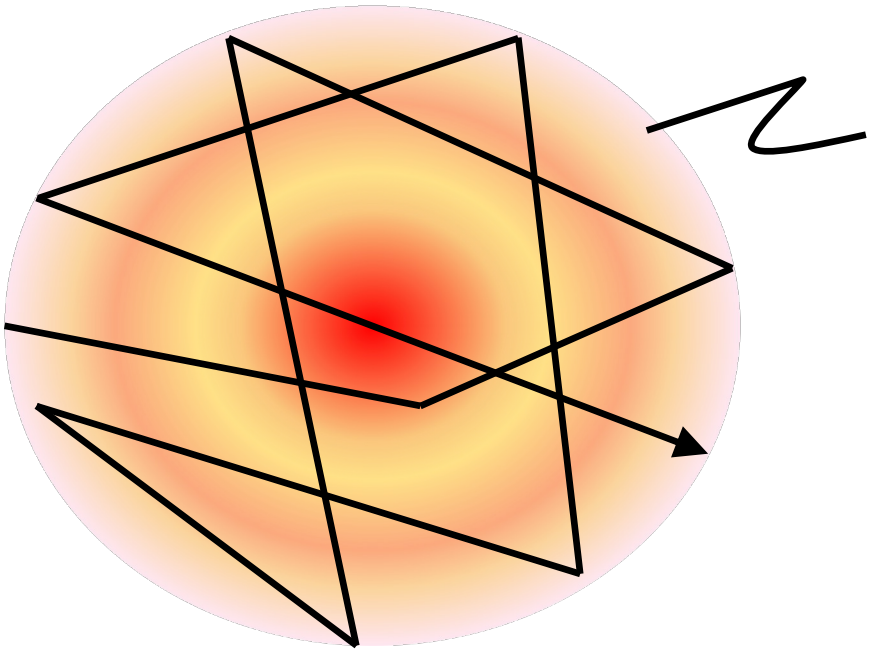
$$\dot{q}_i = \frac{\partial H}{\partial p_i} \qquad \dot{p}_i = -\frac{\partial H}{\partial q_i} \qquad i = 1, \ldots, 3N$$

Remark:

- $H\left(p_i, q_i, t\right)$ and $L\left(\dot{q}_i, q_i, t\right)$ are connected via a

  Legendre-Transformation

Ergodicity Hypothesis



6N-dimensional phase space

All parts of phase space are eventually touched:
Time-average = Ensemble Average

- There is **no general proof** for ergodicity!

Basic MD Integration Algorithm

Algorithm 1.1 Basic Algorithm

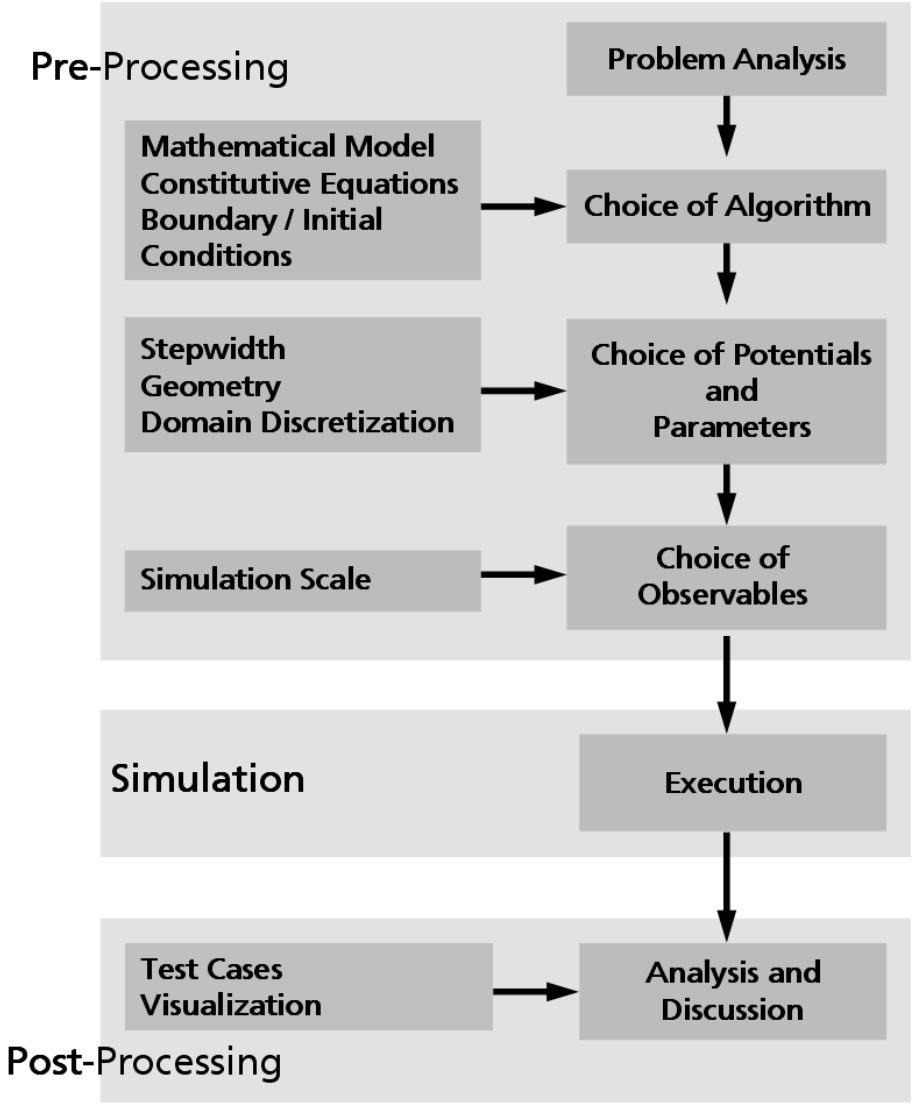
```
real t = t_start;
for i = 1, ..., N
  set initial conditions x_i (positions) and v_i (velocities);
while (t < t_end) {
  compute for i = 1, ..., N the new positions x_i and velocities v_i
    at time t + delta_t by an integration procedure from the
    positions x_i, velocities v_i and forces F_i on the particle at
    earlier times;
  t = t + delta_t;
}
```

The Computational MD Experiment

- **Initialize:** select positions and velocities
- **Integrate:** compute all forces, and determine new positions
- **Equilibrate:** let the system reach equilibrium

  (i.e. lose memory of initial conditions)
- **Average:** accumulate quantities of interest

Scheme of any Computer Simulation



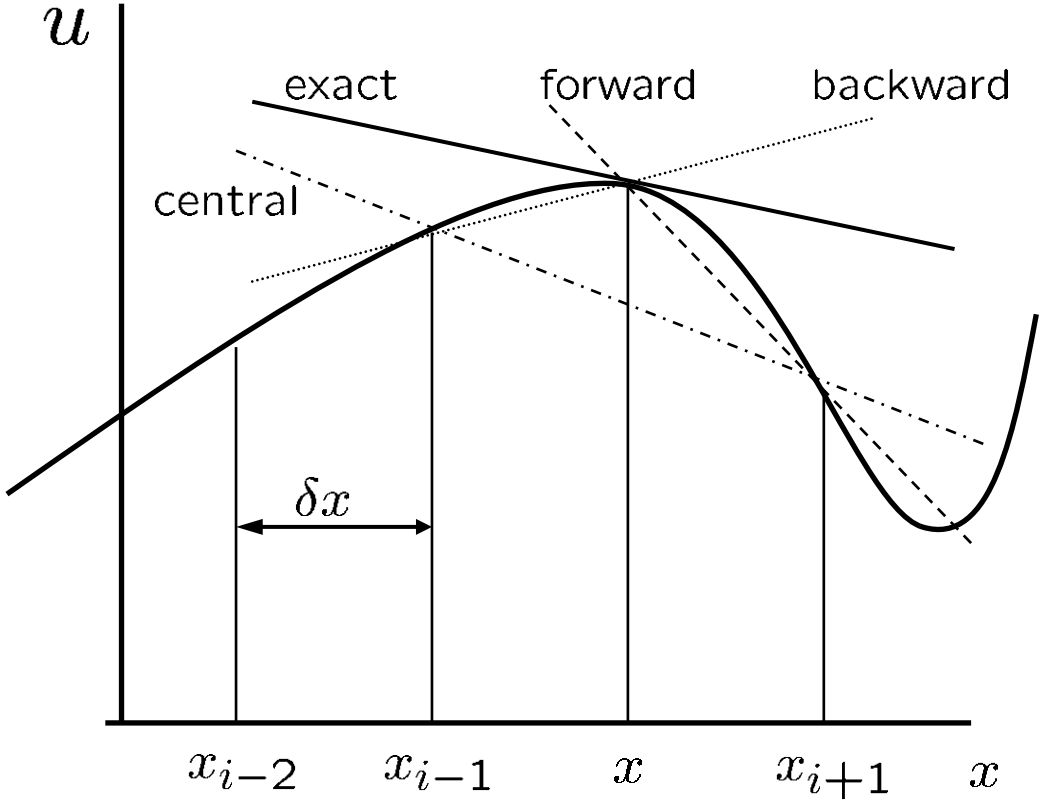
Taken from:
M. O. Steinhauser: Multiscale Modeling of Fluids and Solids – Theory and Applications, Springer, 2nd edition, Berlin, Heidelberg, Boston, 2008

Integration: Many Variants of MD According to the Ensemble

- Use an integrator (Verlet, leapfrog, velocity verlet, Gear-predictor-corrector…)
- **Robust**, long-term **conservation of the constants of motions**, **time-reversible**, constant volume in phase space
- Choose the desired **thermodynamic ensemble** (microcanonical NVE, or canonical NVT using a thermostat, isobaric-isothermic NOT with a barostat,…)
- **Stochastic** (Langevin), **constrained** (velocity re-scaling,…), **extended system** (Nosé-Hover)

Spatial and Temporal Discretization

- Numerical integration
  - Forward difference
  - Backward difference
  - Central difference



Molecular Dynamics Solves the N-Body Problem
Naive Approach: Taylor Expansion

Classical N-body initial value problem:

Can only be solved numerically (except in very special cases)

How?

$$X(t+\Delta t) = X(t) + \dot{X}(t)\Delta t + \frac{1}{2!}\ddot{X}\left(t\right)\Delta t^2 + \frac{1}{3!}\dddot{X}\left(t\right)\Delta t^3 + \cdots$$

Molecular Dynamics Solves the N-Body Problem
Naive Approach: Taylor Expansion

Classical N-body initial value problem:

Can only be solved numerically (except in very special cases)

How?

$$X(t+\Delta t) = X(t) + \dot{X}(t)\Delta t + \frac{1}{2!}\ddot{X}\left(t\right)\Delta t^2 + \frac{1}{3!}\dddot{X}\left(t\right)\Delta t^3 + \cdots$$

Solving the N-Body Problem
Naive Approach: Taylor Expansion

Classical N-body initial value problem:

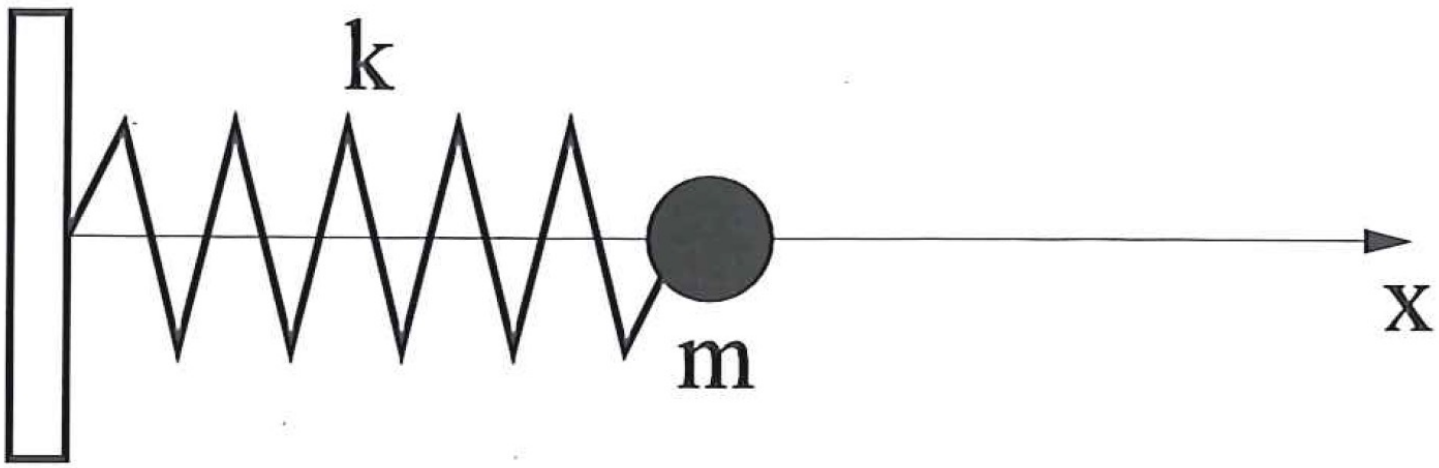
Can only be solved numerically (except in very special cases)

How? Truncate the Taylor Expansion

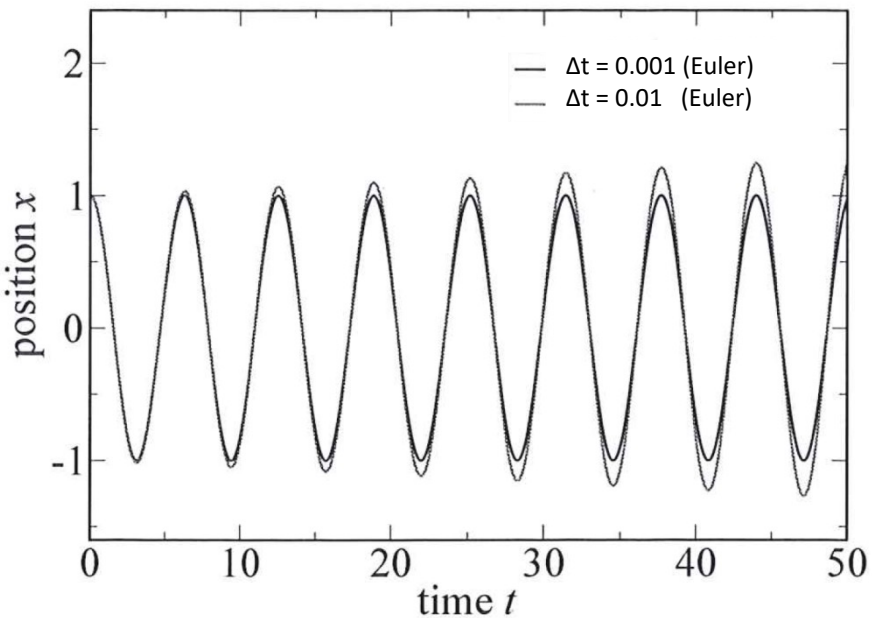
$$X(t+\Delta t) = X(t) + \dot{X}(t)\Delta t + \frac{1}{2!}\ddot{X}\left(t\right)\Delta t^2$$

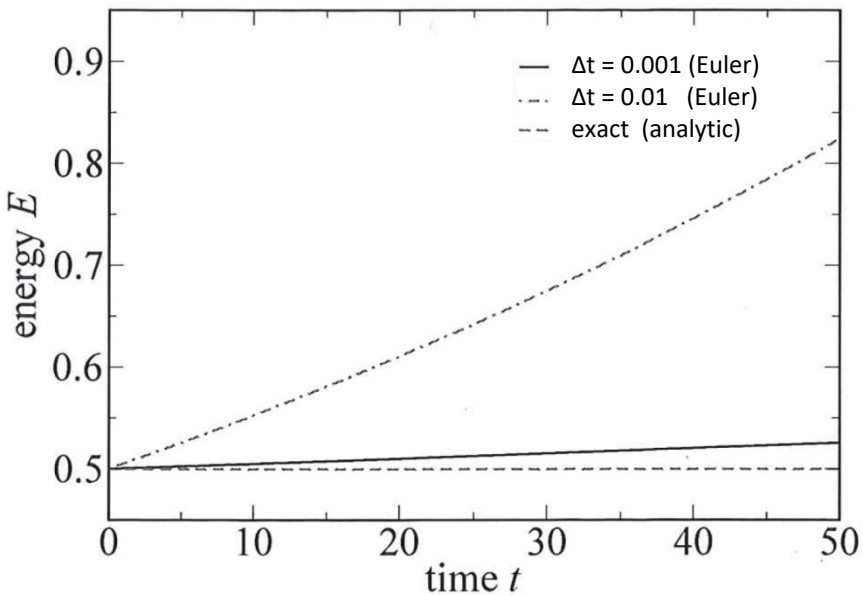
Absolutely Forbidden!

Simple Example: 1D Harmonic Oscillator



Simple Example: 1D Harmonic Oscillator





Forward Euler Method:

- Is not time reversible
- Does not conserve volume in phase space
- Suffers from energy drift

Solving the N-Body Problem
Naive Approach: Taylor Expansion (Forward Euler Method)

Forward Euler Method:

- Is not time reversible
- Does not conserve volume in phase space
- Suffers from energy drift

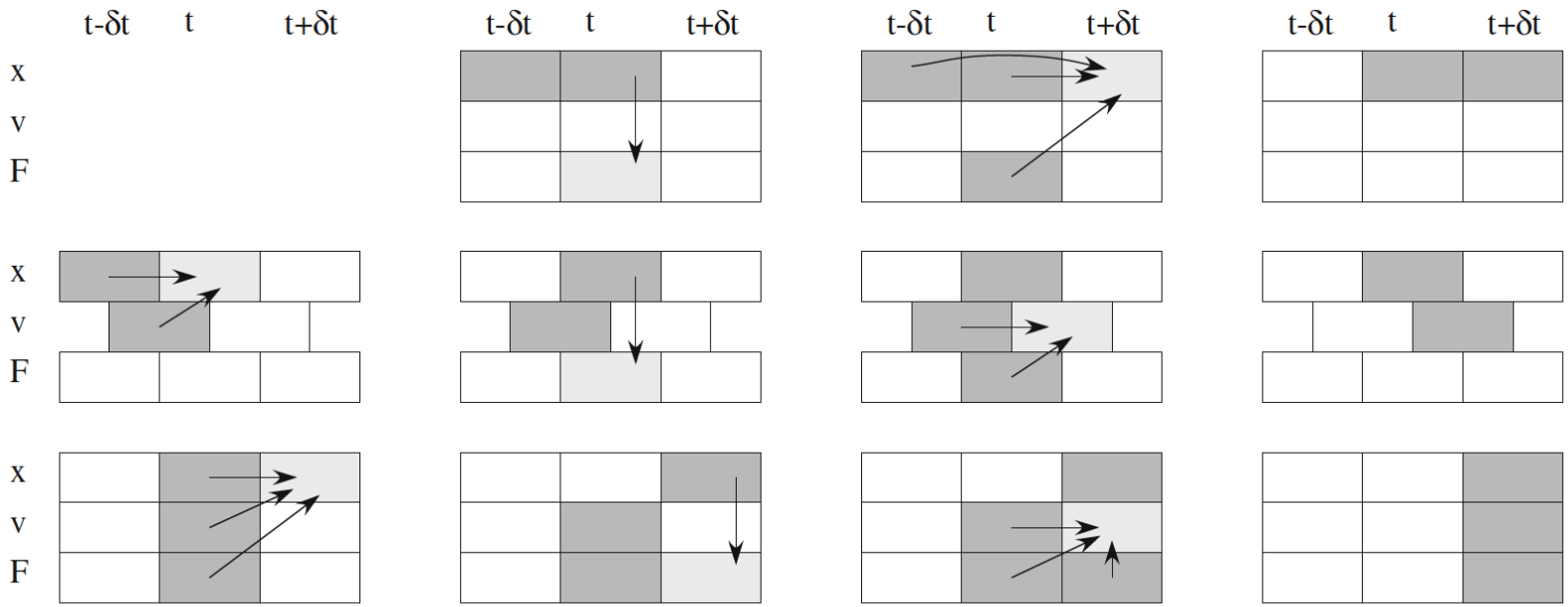
Absolutely NOT !

The Standard Velocity Verlet Algorithm

Algorithm 1.2 Velocity-Störmer-Verlet Method

```
//  start with initial data x, v, t
//  auxiliary vector F^old;
compute forces F;
while (t < t_end) {
  t = t + delta_t;
  loop over all i {                                    // update x
    x_i = x_i + delta_t * (v_i + .5 / m_i * F_i* delta_t); // using (6*)
    F_i^old = F_i;
  }
  compute forces F;
  loop over all i                                      // update v
    v_i = v_i + delta_t * .5 / m_i * (F_i + F_i^old);  // using (7*)
  compute derived quantities as for example kinetic or potential energy;
  print values of t, x, v as well as derived quantities;
}
```

Different Schemes of The Verlet Algorithm



- Top: Standard Verlet Scheme
- Middle: Leapfrog Scheme
- Bottom: Velocity Verlet

Example: Planetary Motion Code (PMC)

Data structure Particle

```
typedef struct {
  real m;          // mass
  real x[DIM];     // position
  real v[DIM];     // velocity
  real F[DIM];     // force
} Particle;
```

Algorithm Velocity-Störmer-Verlet Method

```
void timeIntegration_basis(real t, real delta_t, real t_end,
                           Particle *p, int N) {
  compF_basis(p, N);
  while (t < t_end) {
    t += delta_t;
    compX_basis(p, N, delta_t);
    compF_basis(p, N);
    compV_basis(p, N, delta_t);
    compoutStatistic_basis(p, N, t);
    outputResults_basis(p, N, t);
  }
}
```

Example: Planetary Motion Code (PMC)

Algorithm ... Routines for the Velocity-Störmer-Verlet Time Step for a Vector of Particles

```
void compX_basis(Particle *p, int N, real delta_t) {
  for (int i=0; i<N; i++)
    updateX(&p[i], delta_t);
}
void compV_basis(Particle *p, int N, real delta_t) {
  for (int i=0; i<N; i++)
    updateV(&p[i], delta_t);
}
```

Algorithm Computation of the Force with $\mathcal{O}(N^2)$ Operations

```
void compF_basis(Particle *p, int N) {
  for (int i=0; i<N; i++)
    for (int d=0; d<DIM; d++)
      p[i].F[d] = 0;                    // set F for all particles to zero
  for (int i=0; i<N; i++)
    for (int j=0; j<N; j++)
      if (i != j) force(&p[i], &p[j]);     // add the forces F_ij to F_i
}
```

Example: Planetary Motion Code (PMC)

Algorithm Gravitational Force between two Particles

```
void force(Particle *i, Particle *j) {
  real r = 0;
  for (int d=0; d<DIM; d++)
    r += sqr(j->x[d] - i->x[d]);              // squared distance r=r_ij^2
  real f = i->m * j->m /(sqrt(r) * r);
  for (int d=0; d<DIM; d++)
    i->F[d] += f * (j->x[d] - i->x[d]);
}
```

Code fragment Allocate and Free Memory Dynamically

```
Particle *p = (Particle*)malloc(N * sizeof(*p));     // reserve
free(p);                                             // and release memory
```

Example: Planetary Motion Code (PMC)

Algorithm Main Program

```
int main() {
  int N;
  real delta_t, t_end;
  inputParameters_basis(&delta_t, &t_end, &N);
  Particle *p = (Particle*)malloc(N * sizeof(*p));
  initData_basis(p, N);
  timeIntegration_basis(0, delta_t, t_end, p, N);
  free(p);
  return 0;
}
```

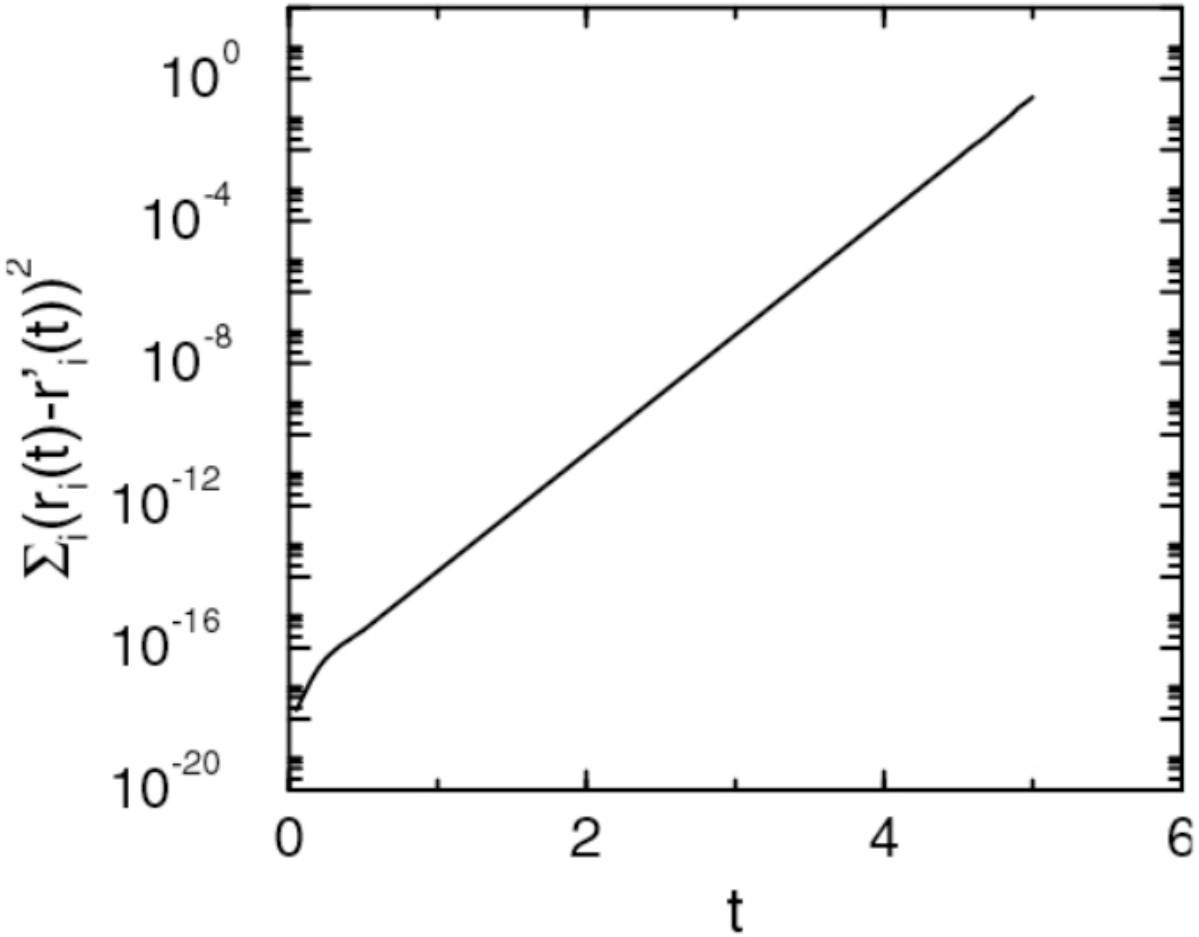
Lyapunov Instabilities

- The dynamics of a well-behaved classical many body (N particle) system is **chaotic!**

- Consequence: Trajectories of particles that differ very slightly in their initial conditions, **diverge exponentially ! (Lyapunov Instability)**

Lyapunov Instabilities

- The Lyapunov desaster in action…



Lyapunov Instabilities

- Any small error in the numerical integration of the equations of motion **will blow up exponentially…**

always…

and… for *any* algorithm !

So…

Why should anyone believe in Molecular Dynamics Simulation?

What is the point of simulating dynamics if we cannot trust the resulting time-evolution?

Answer: We‘re interested in *Statistical* Properties

Here, everything works out fine!

Analysis and Interpretation of MD

Relate **microscopic** phenomena simulated with the MD method and **macroscopic** properties:

Given a thermodynamic state of a material, what are the probabilities of finding the system in the various possible microscopic states?

Or: Given a series of microscopic states, what is the corresponding macroscopic state?

→ To answer this question, we need Statistical Mechanics !

Live Demo

My University Research Page: https://www.frankfurt-university.de/steinhauser

Contact Me: martin.steinhauser@fb2-fra.uas.de

Research Gate: https://www.researchgate.net/profile/Martin-Steinhauser