1.021, 3.021, 10.333, 22.00 Introduction to Modeling and Simulation
Spring 2018

Property calculation

Lecture 4

Markus J. Buehler

Laboratory for Atomistic and Molecular Mechanics
Department of Civil and Environmental Engineering
Massachusetts Institute of Technology

E-mail: mbuehler@MIT.EDU
URL: http://web.mit.edu/mbuehler/www/

Massachusetts Institute of Technology

Content overview

I. Fundamentals of particle methods

Lectures 1-12
February/March

1. Atoms, molecules, chemistry
2. Statistical mechanics
3. Molecular dynamics, Monte Carlo
4. Visualization and data analysis
5. Mechanical properties – application: how things fail (and how to prevent it)
6. Multi-scale modeling paradigm
7. Biological systems (simulation in biophysics) – how proteins work and how to model them

II. Advanced topics in particle methods

Lectures 13-24
March/April/May

1. Quantum Weirdness: The Theory of Quantum Mechanics
2. The Many-Body Problem: From Many-Body to Single-Particle
3. Quantum modeling of materials
4. From Atoms to Solids
5. Basic properties of materials
6. Advanced properties of materials
7. Materials Informatics

Lecture 4: Property calculation

Outline:

1. How to calculate properties from atomistic simulation
2. How to solve the equations
    2.1 Ergodic hypothesis
3. Monte Carlo (MC) method
    3.1 Application to integration

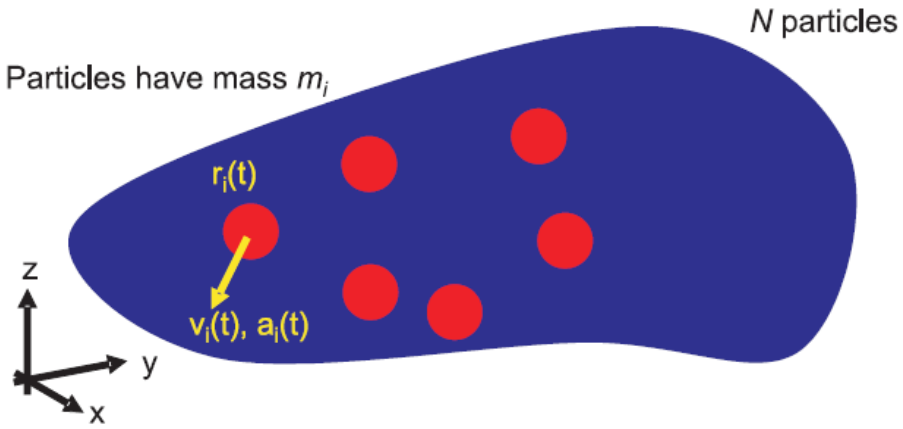
Goals of today's lecture:

- Understand connection between micro- and macro-states
- Relevance of statistical mechanics & sampling
- How to solve the equations – MD or MC
- First introduction to MC

1. How to calculate properties from atomistic simulation

Tool: statistical mechanics

Molecular dynamics



Follow trajectories of atoms (*classical mechanics, Newton's laws*)

"Verlet central difference method"

$$r_i(t_0+\Delta t) = -r_i(t_0-\Delta t) + 2r_i(t_0)\Delta t + a_i(t_0)(\Delta t)^2 + \ldots$$

Positions at t_0-Δt | Positions at t_0 | Accelerations at t_0

$$a_i = f_i / m$$

Property calculation: Introduction

Have:

$$\vec{x}(t), \dot{\vec{x}}(t), \ddot{\vec{x}}(t)$$

“microscopic information”

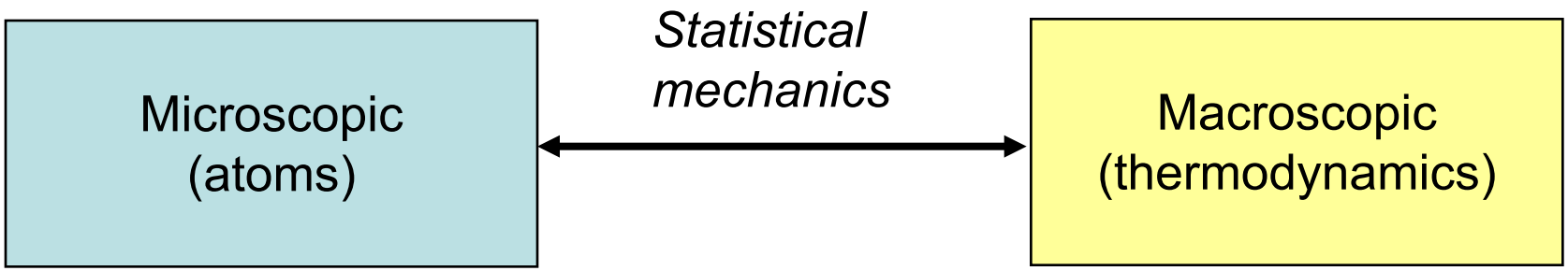
Want:

- Thermodynamical properties (temperature, pressure, stress, strain, thermal conductivity, ..)
- State (gas, liquid, solid)
- …

(properties that can be measured in experiment!)

Goal: To develop a robust framework to calculate a range of “macroscale” properties from MD simulation studies (“microscale information”)

Link between statistical mechanics and thermodynamics



Macroscopic conditions (e.g. constant volume, temperature, number of particles…) translate to the microscopic system as boundary conditions (constraints)

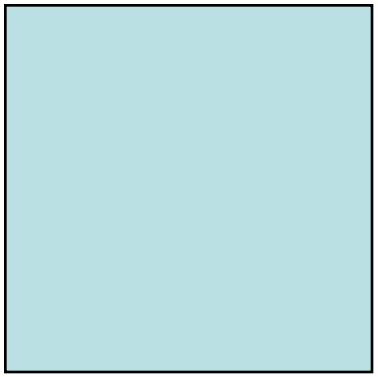
Macroscopic system: defined by extensive variables, which are constant: E.g. (N,V,E)=NVE ensemble

The behavior of the **microscopic system** is related to the macroscopic conditions. In other words, the distribution of microscopic states is related to the macroscopic conditions.

To **calculate macroscopic properties** (via statistical mechanics) from microscopic information we need to know the distribution of microscopic states (e.g. through a simulation)

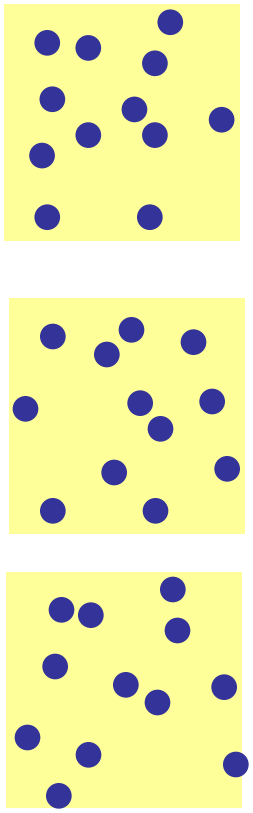
Macroscopic vs. microscopic states

Canonical ensemble



N, V, T

$\equiv$



$C_1 \longleftrightarrow r_1, p_1$

$C_2 \longleftrightarrow r_2, p_2$

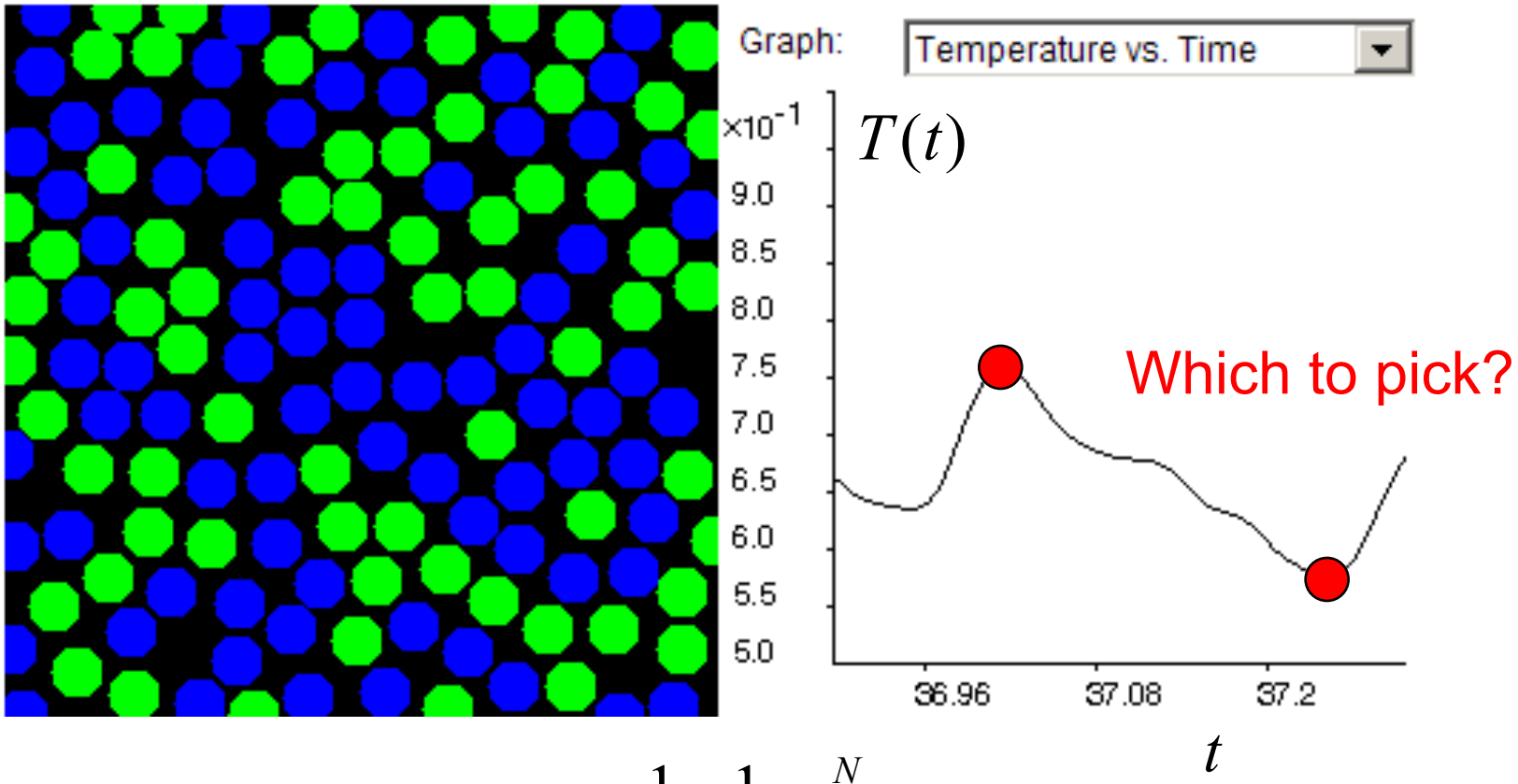
$C_3 \longleftrightarrow r_3, p_3$

...

$C_N \longleftrightarrow r_N, p_N$

Same macroscopic state is represented by many different microscopic configurations

Micro-macro relation



$$T(t) = \frac{1}{3}\frac{1}{Nk_B}\sum_{i=1}^{N} m_i \vec{v}_i^{\,2}(t)$$

Specific (individual) microscopic states are insufficient to relate to macroscopic properties

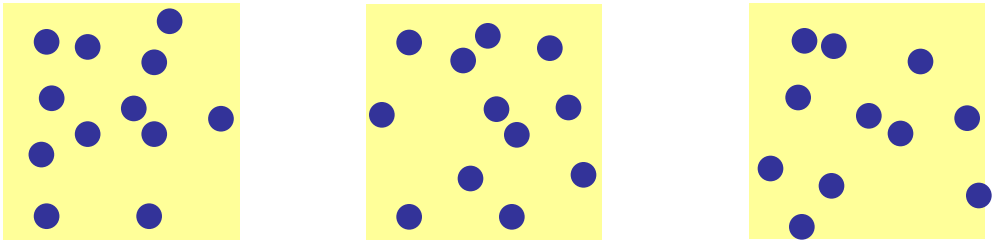
Averaging over the ensemble

- Rather than taking single measurement, need to average over “all” microscopic states that represent the corresponding macroscopic condition

- This averaging needs to be done in a suitable fashion, that is, we need to **consider the specific distribution of microscopic states** (e.g. some microscopic states may be more likely than others)

What about trying this….

Property A_1 Property A_2 Property A_3



C_1 C_2 C_3

r_1, p_1 r_2, p_2 r_3, p_3

$$A_{\text{macro}} = \frac{1}{3}\left(A_1 + A_2 + A_3\right)$$

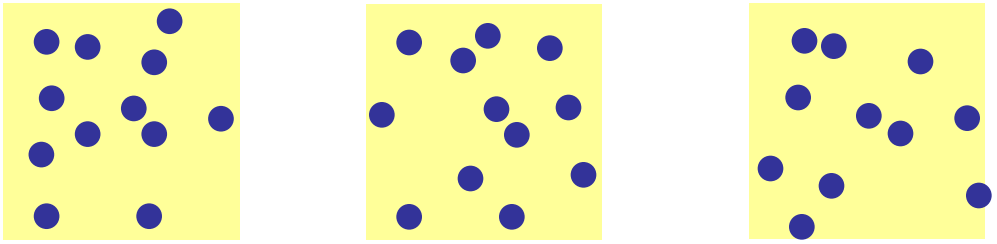
Averaging over the ensemble

- Rather than taking single measurement, need to average over “all” microscopic states that represent the corresponding macroscopic condition

- This averaging needs to be done in a suitable fashion, that is, we need to **consider the specific distribution of microscopic states** (e.g. some microscopic states may be more likely than others)

What about trying this….

Property A_1 Property A_2 Property A_3



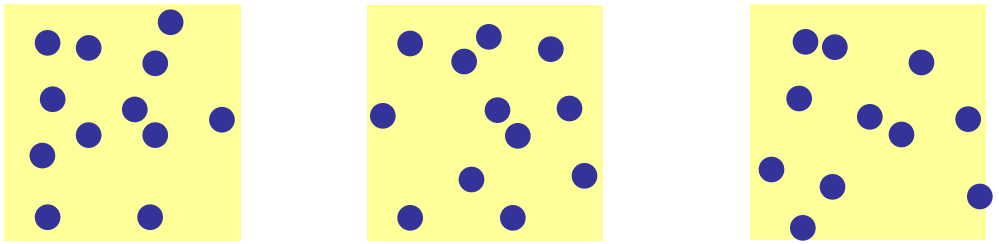
C_1 C_2 C_3

$$A_{\text{macro}} \neq \frac{1}{3}\left(A_1 + A_2 + A_3\right)$$

Generally, NO!

Averaging over the ensemble

Property A_1 Property A_2 Property A_3



C_1 C_2 C_3

$$A_{\text{macro}} \neq \frac{1}{3}\left(A_1 + A_2 + A_3\right)$$

Instead, we must average with proper weights that represent the probability of a system in a particular microscopic state!

(I.e., not all microscopic states are equal)

$$A_{\text{macro}} = \rho_1 A_1 + \rho_2 A_2 + \rho_3 A_3 =$$

$$\rho_1(r_1, p_1)A_1(r_1, p_1) + \rho_2(r_2, p_2)A_2(r_2, p_2) + \rho_3(r_3, p_3)A_3(r_3, p_3)$$

Probability to find system in state C_1

How to relate microscopic states to macroscopic variables?

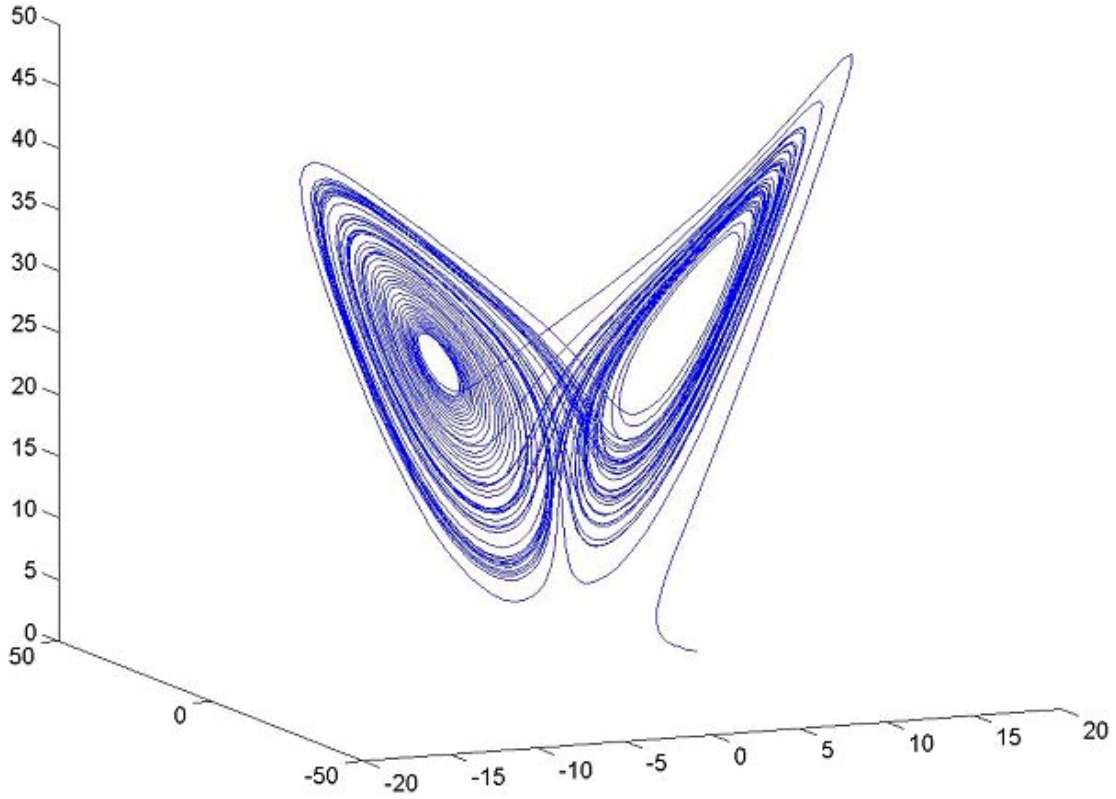
$A(r, p)$ Property due to specific microstate

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp$$

- Ensemble average, obtained by integral over all microscopic states

- Proper weight $\rho(p,r)$ - depends on ensemble

Illustration/example: phase space



6N-dimensional phase space

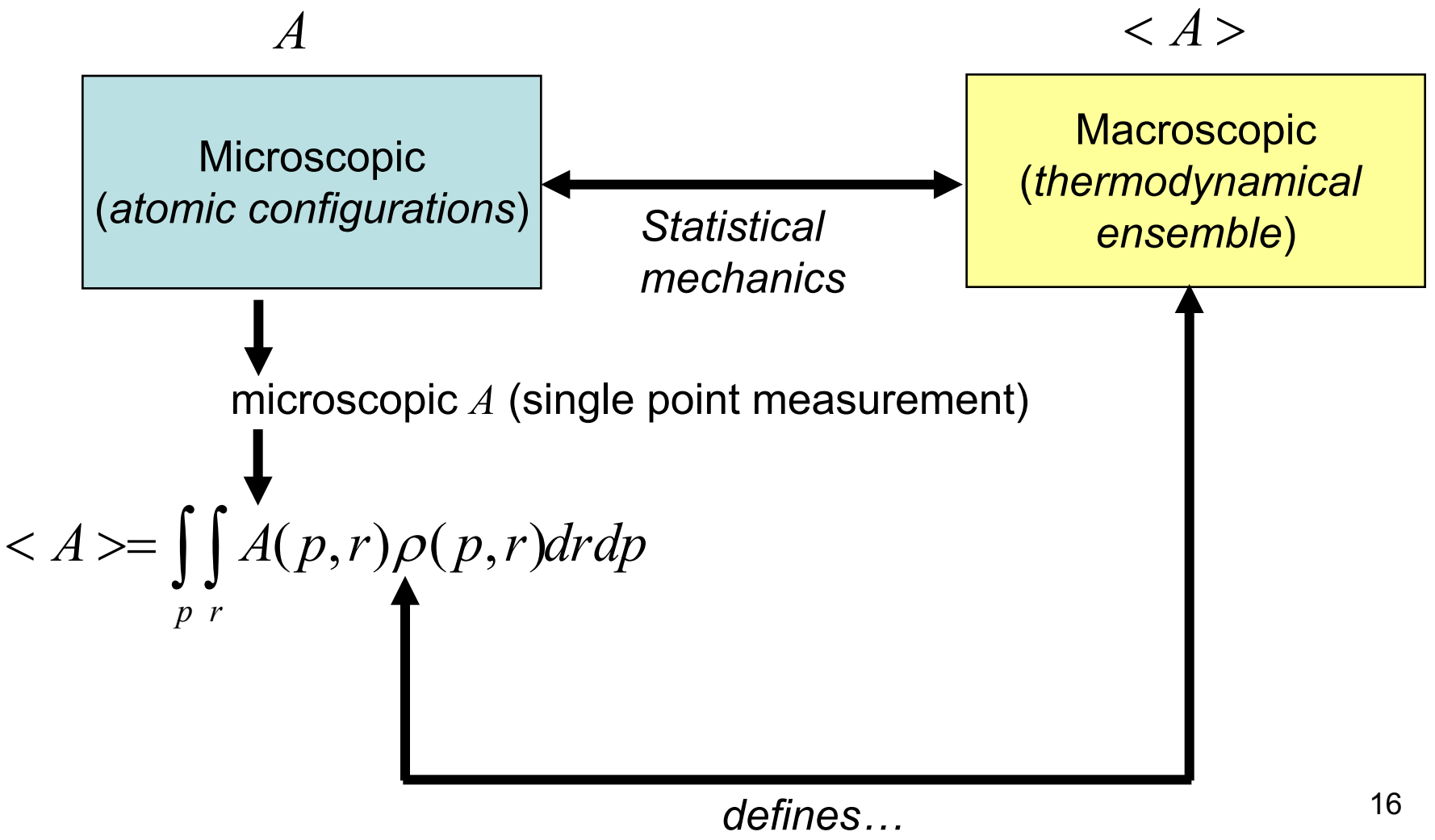
$$r, p$$

$$\downarrow$$

$$\rho(p, r)$$

http://www.ace.gatech.edu/experiments2/2413/lorenz/fall02/

Summary: How macro-micro relation works



2. How to solve the equations

Approaches in solving this problem

- **Method of choice: *Numerical simulation***

- Two major approaches:

  1. Using **molecular dynamics (MD)**: Generate microscopic information through dynamical evolution of microscopic system (*i.e.,* simulate the "real behavior" as we would obtain in lab experiment)

  2. Using a numerical scheme/algorithm to randomly generate microscopic states, which, through proper averaging, can be used to compute macroscopic properties. Methods referred to as "**Monte Carlo**"

Monte Carlo (MC) scheme

- Concept: Find simpler way to solve the integral

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp$$

- Use idea of “random walk” to step through relevant microscopic states and thereby create proper weighting (visit states with higher probability density more often)

=*ensemble (statistical) average*

MC algorithm result

Final result of MC algorithm:

Algorithm that leads to proper Distribution of microscopic states…

Ensemble (statistical) average

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp \quad \longleftrightarrow \quad \boxed{< A > \frac{1}{N_A}\sum_i A_i}$$

Carry out algorithm for N_A steps
Average results
..done!

Ergodicity

- MC method is based on directly computing the ensemble average

  Define a series of microscopic states that reflect the appropriate ensemble average; weights intrinsically captured since states more likely are visited more frequently and vice versa

- **Egodicity**: The ensemble average is equal to the time-average during the dynamical evolution of a system under proper thermodynamical conditions.

  In other words, the set of microscopic states generated by solving the equations of motion in MD “automatically” generates the proper distribution/weights of the microscopic states

  This is called the Ergodic hypothesis:

$$< A >_{Ens} = < A >_{Time}$$

Ergodic hypothesis

- Ergodic hypothesis:

Ensemble (statistical) average = time average

- All microstates are sampled with appropriate probability density over long time scales

$$\underbrace{\frac{1}{N_A}\sum_{i=1..N_A} A(i)}_{\text{MC}} = < A >_{Ens} = < A >_{Time} = \underbrace{\frac{1}{N_t}\sum_{i=1..N_t} A(i)}_{\text{MD}}$$

Example: Temperature

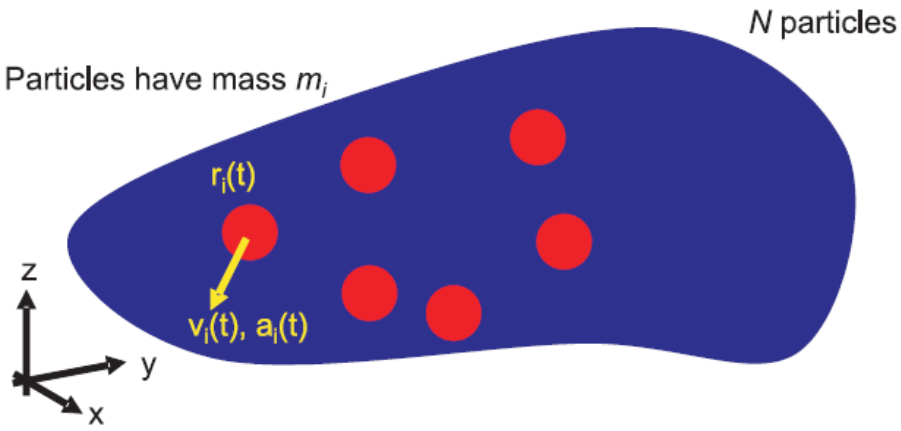
- Ergodic hypothesis:

Ensemble (statistical) average = time average

- All microstates are sampled with appropriate probability density over long time scales

$$\underbrace{\frac{1}{N_A}\sum_{i=1..N_A}\left(\frac{1}{3}\frac{1}{Nk_B}\sum_{i=1}^{N}m_i\vec{v}_i^{\,2}\right)}_{\text{MC}} = < A >_{Ens} = < A >_{Time} = \underbrace{\frac{1}{N_t}\sum_{i=1..N_t}\left(\frac{1}{3}\frac{1}{Nk_B}\sum_{i=1}^{N}m_i\vec{v}_i^{\,2}\right)}_{\text{MD}}$$

Importance for MD algorithm



Follow trajectories of atoms (*classical mechanics, Newton's laws*)

"Verlet central difference method"

$$r_i(t_0+\Delta t) = -r_i(t_0-\Delta t) + 2r_i(t_0)\Delta t + a_i(t_0)(\Delta t)^2 + \ldots \quad a_i = f_i / m$$

Positions at t_0-Δt | Positions at t_0 | Accelerations at t_0

It's sufficient to simply average over all MD steps…

$$< A >_{Time} = \frac{1}{N_t} \sum_{i=1..N_t} A(i)$$

Molecular dynamics

- During integration of equations of motion – must impose thermodynamical constraints
- For example, *Verlet central difference method* leads to a microcanonical ensemble (NVE)
- Other integration methods exist to generate NVT, NpT ensembles etc.

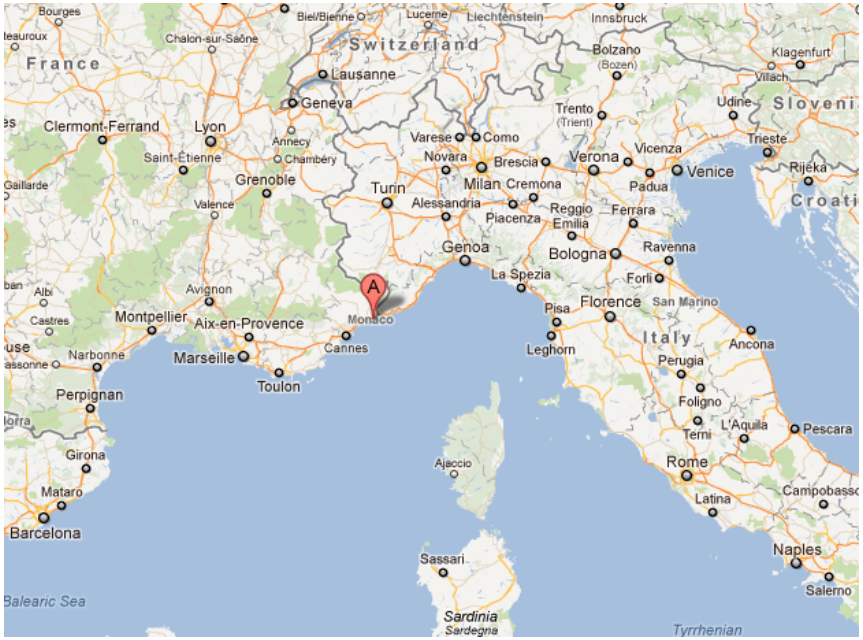
$$r_i(t_0 + \Delta t) = -r_i(t_0 - \Delta t) + 2r_i(t_0)\Delta t + a_i(t_0)(\Delta t)^2 + \ldots$$

Positions at t_0-Δt | Positions at t_0 | Accelerations at t_0

$$a_i = f_i / m$$

3. Monte Carlo method

Monte Carlo Casino
Monte Carlo, Monaco



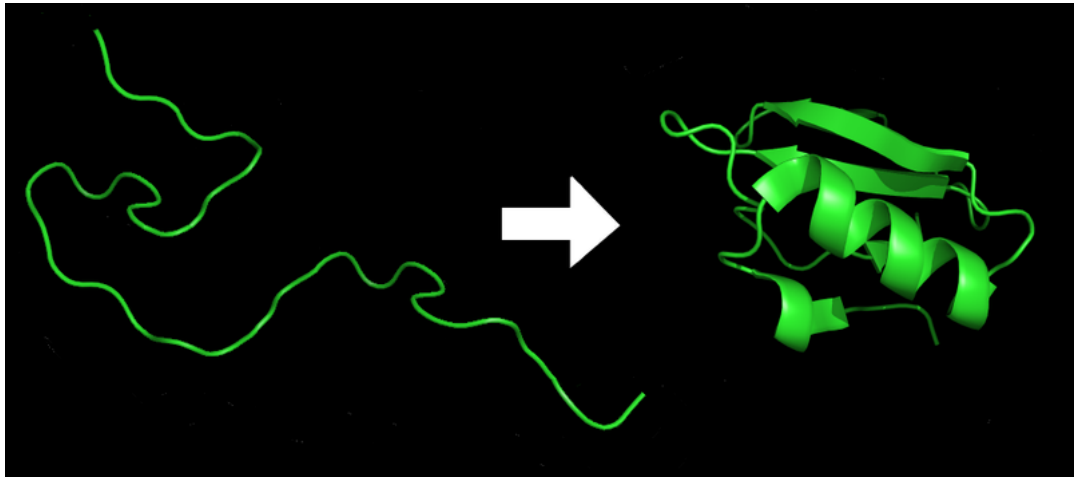
History of Monte Carlo techniques

Developed in the 1940s with the advent of computers – idea: solve differential equations by randomly generating solutions

Term originated in 1946 by **von Neumann and Ulam** – code name for secret project based on the concept outlined above

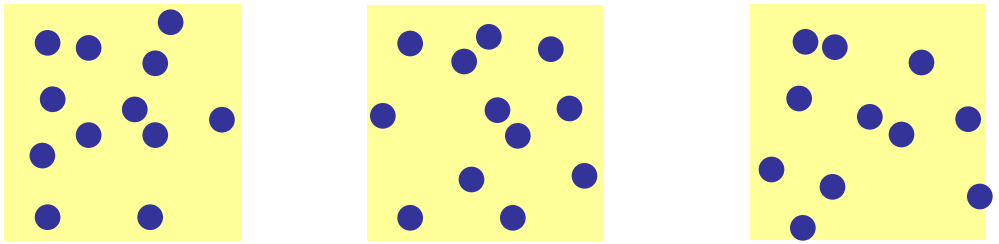
Manhattan project

Protein folding



Averaging over the ensemble

Property A_1 Property A_2 Property A_3



C_1 C_2 C_3

$$A_{\text{macro}} \neq \frac{1}{3}\left(A_1 + A_2 + A_3\right)$$

Instead, we must average with proper weights that represent the probability of a system in a particular microscopic state!

(I.e., not all microscopic states are equal)

$$A_{\text{macro}} = \rho_1 A_1 + \rho_2 A_2 + \rho_3 A_3 =$$

$$\rho_1(r_1, p_1)A_1(r_1, p_1) + \rho_2(r_2, p_2)A_2(r_2, p_2) + \rho_3(r_3, p_3)A_3(r_3, p_3)$$

Probability to find system in state C_1

How to solve…

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp$$

Probability density distribution

Virtually impossible to carry out analytically

Must know all possible configurations

Therefore: Require numerical simulation

Molecular dynamics OR Monte Carlo

3.1 Application to integration

“Random sampling”

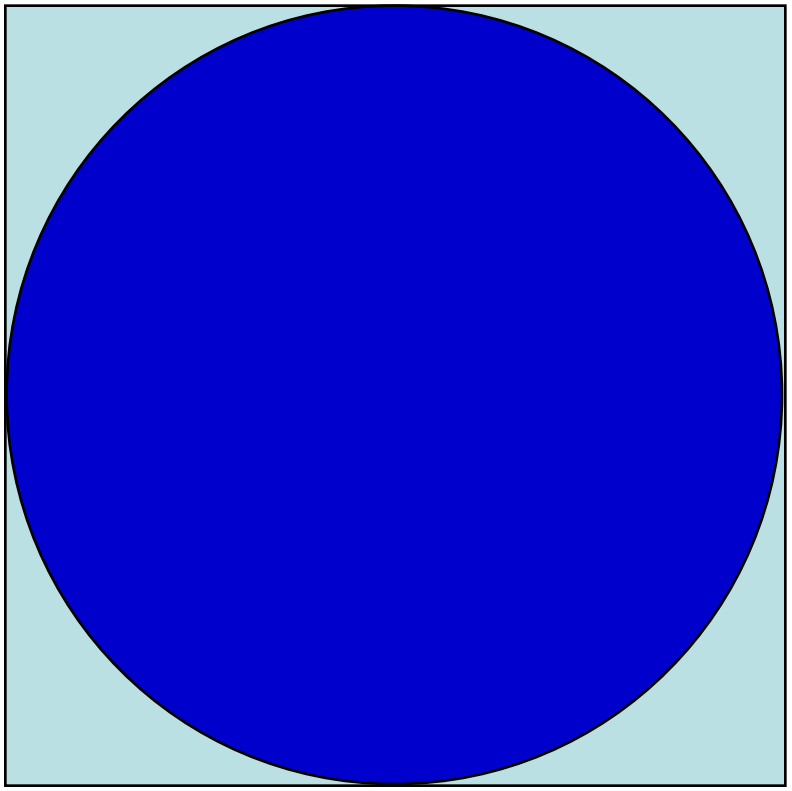
Monte Carlo scheme

Method to carry out integration over “domain”

Want:

$$A = \int_{\Omega} f(\vec{x}) d\Omega$$

Ω



E.g.: Area of circle (value of π)

$$A_C = \frac{\pi d^2}{4} \qquad A_C = \frac{\pi}{4}$$

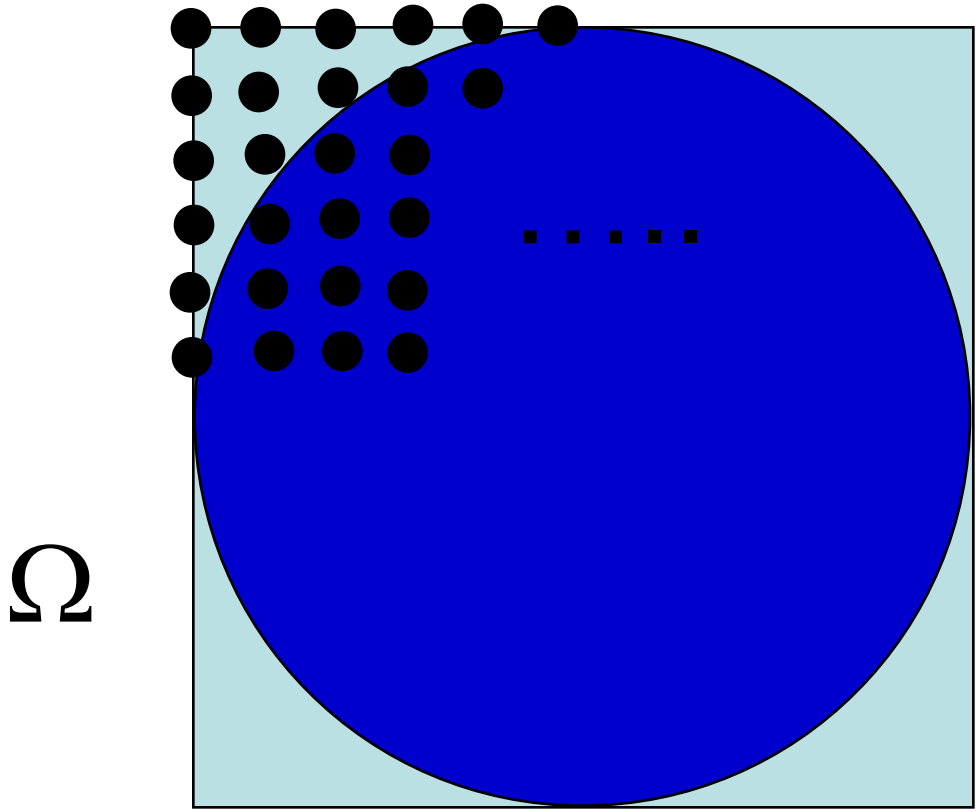
$$\pi = 4 A_C$$

$$d = 1$$

$$f(\vec{x}) = \begin{cases} 1 & \text{inside} \\ 0 & \text{outside} \end{cases}$$

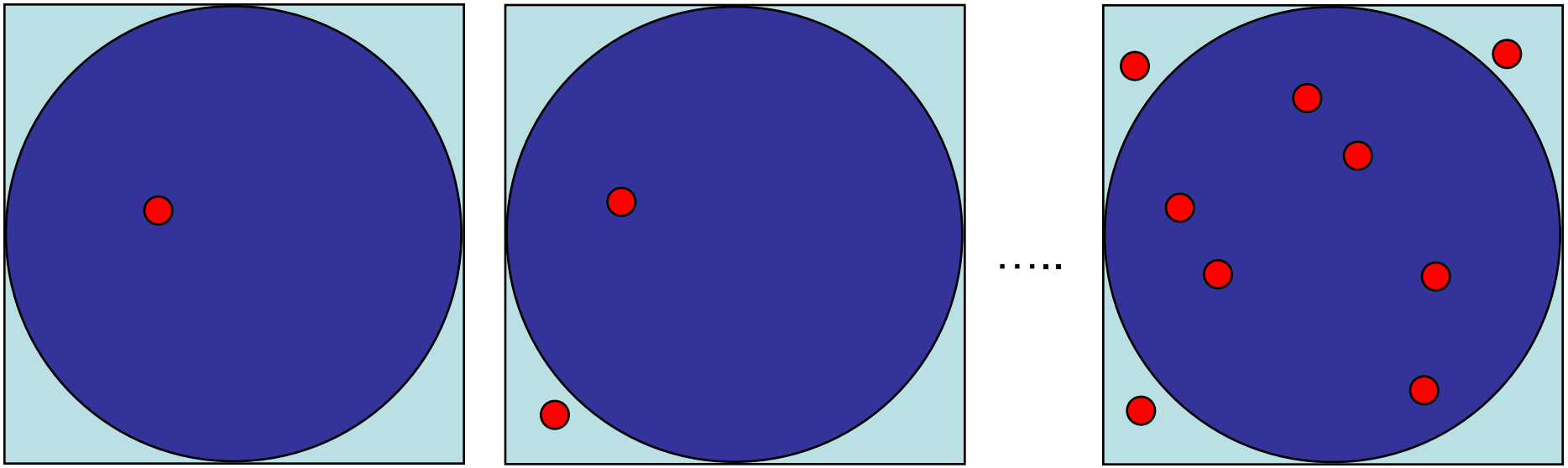
Conventional way…

- Evaluate integrand at predetermined values in the domain (e.g. quadratic grid)
- Evaluate integral at discrete points and sum up



What about playing darts..

Alternative way: integration through MC

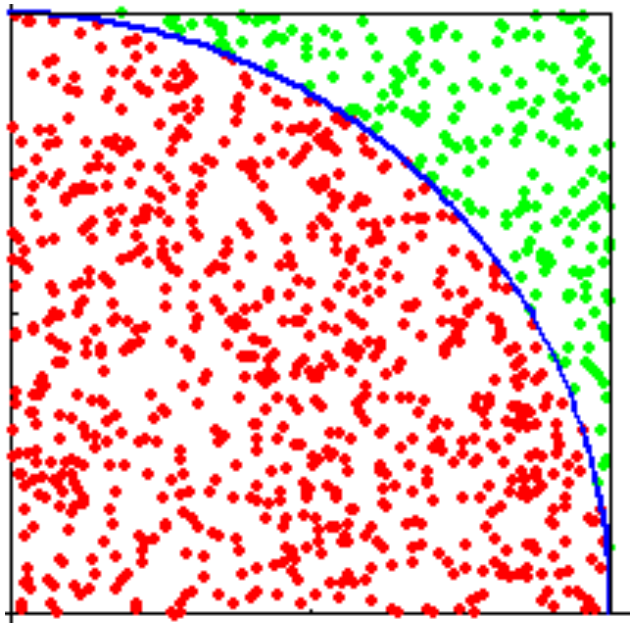


Playing darts: Randomly select point in domain
Evaluate integral a these points
Sum up results to solve integral

Monte Carlo scheme for integration

- **Step 1**: Pick random point $\vec{x}_i$ in Ω
- **Step 2**: Accept/reject point based on criterion (e.g. if inside or outside of circle and if in area not yet counted)
- **Step 3**: If accepted, add $f(\vec{x}_i) = 1$ to the total sum

length r=1



$$A_C = \int_{\Omega} f(\vec{x}) d\Omega \qquad A_C = \frac{\pi}{16}$$

$$A_C = \frac{1}{N_A} \sum_i f(\vec{x}_i)$$

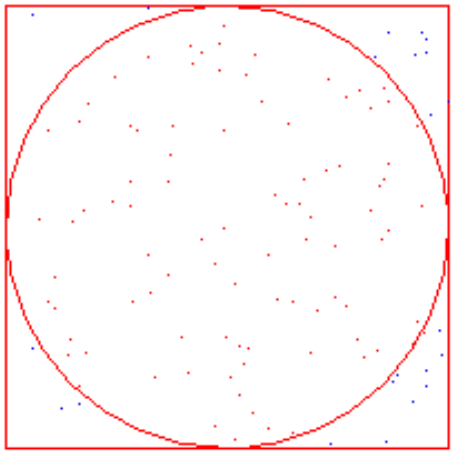
N_A: Attempts made

http://math.fullerton.edu/mathews/n2003/MonteCarloPiMod.html

Java applet: how to calculate pi

- http://polymer.bu.edu/java/java/montepi/montepiapplet.html

Monte Carlo JAVA Applet



Throw 1 Dart Reset

100 Darts

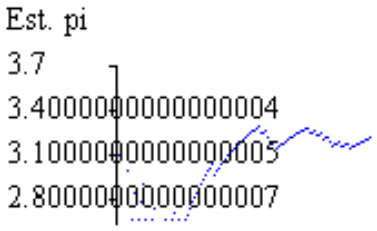
1000 Darts

10000 Darts

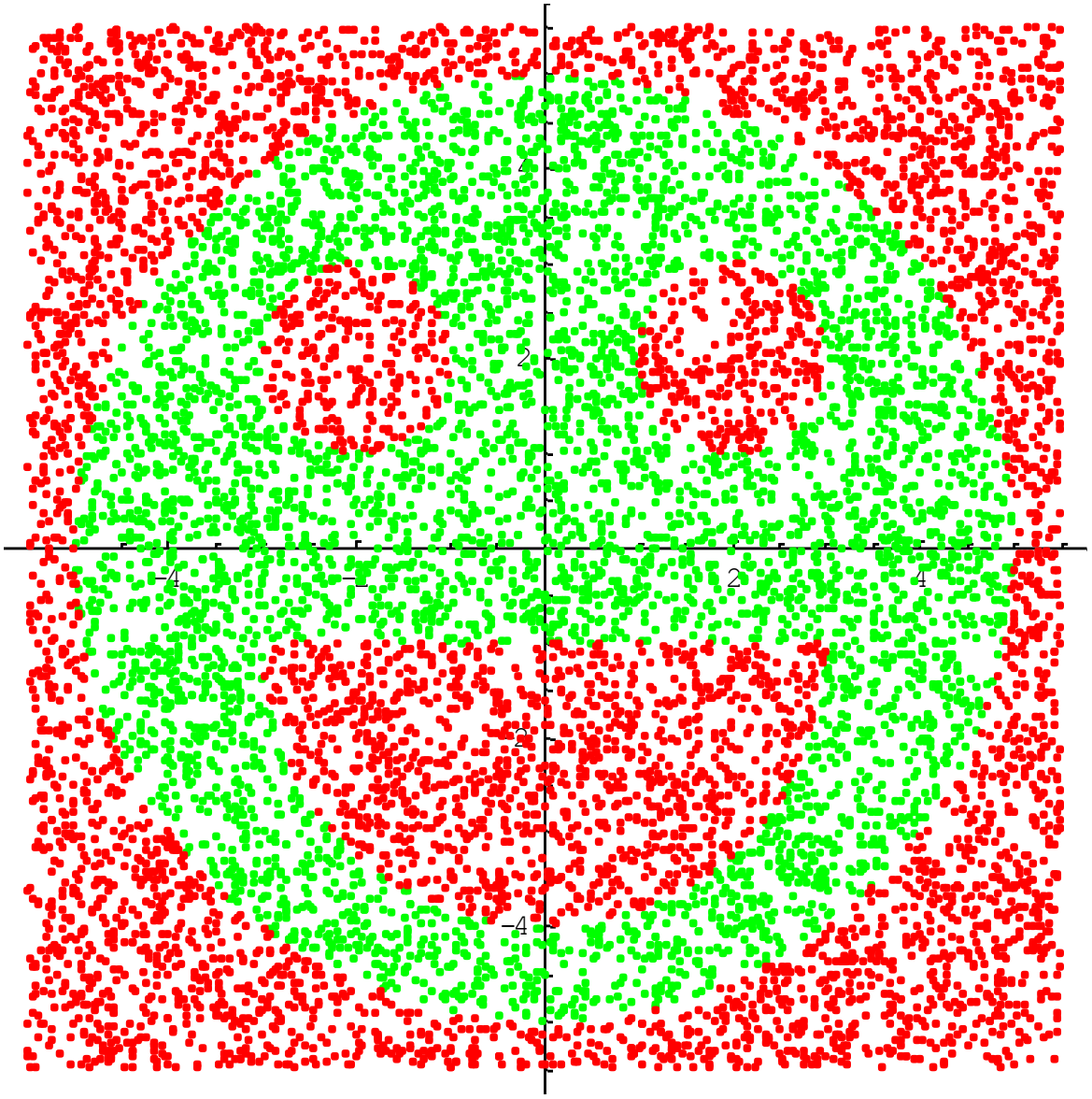
Number of darts in circle: 92

Number of darts in square: 114

Estimate for pi: 3.22807017543



Example: more complicated shapes



After N. Baker (WUSTL)

How to apply to ensemble average?

- Similar method can be used to apply to integrate the ensemble average
- Need more complex iteration scheme (replace "*random sampling*" by "*importance sampling*")

- E.g. Metropolis-Hastings algorithm

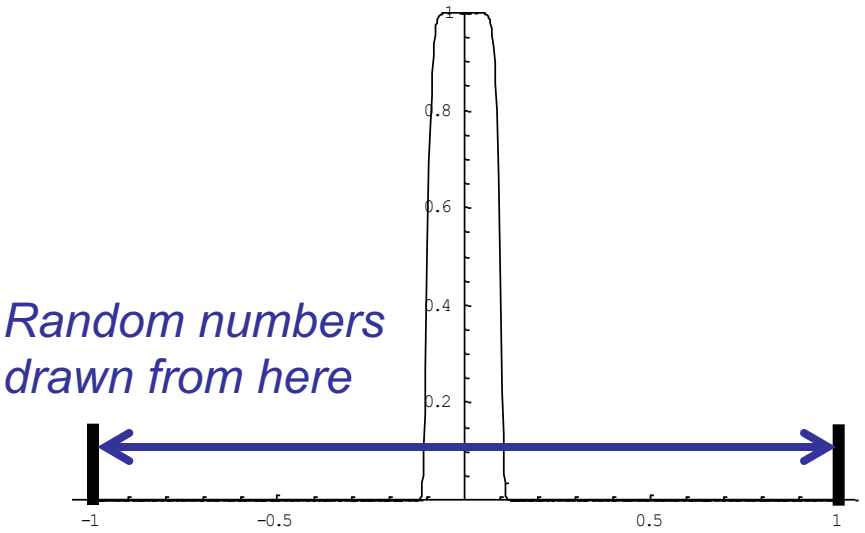
Want:

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp \quad \longleftrightarrow \quad \boxed{< A > \frac{1}{N_A}\sum_i A_i}$$

Challenge: sampling specific types of distributions

- We want to
  - Integrate a sharply-peaked function
  - Use Monte Carlo with uniformly-distributed random numbers (e.g. here from -1 to 1)

$$f(x) = \exp\left(-(100x)^{12}\right)$$

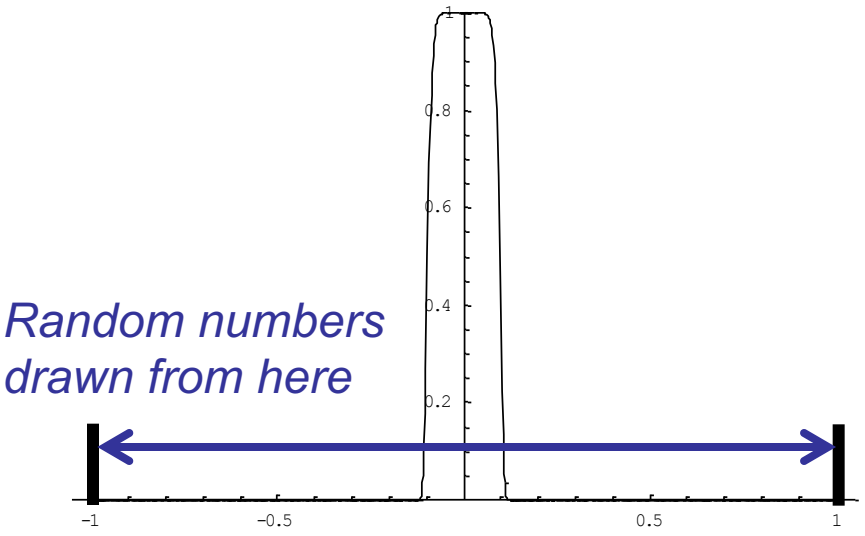


After N. Baker (WUSTL)

Challenge: sampling specific types of distributions

- We want to
  - Integrate a sharply-peaked function
  - Use Monte Carlo with uniformly-distributed random numbers (e.g. here from -1 to 1)
- What happens?
  - Very few points contribute to the integral (~9%)
  - Poor computational efficiency/convergence
- Solution: use a different distribution of random numbers to sample "*importance sampling*"

$$f(x) = \exp\left(-(100x)^{12}\right)$$



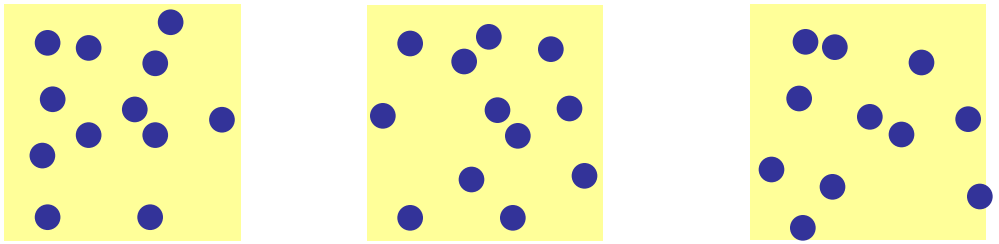
After N. Baker (WUSTL)

3.2 Metropolis-Hastings algorithm

"Importance sampling"

Averaging over the ensemble

Property A_1 Property A_2 Property A_3



C_1 C_2 C_3

$$A_{\text{macro}} \neq \frac{1}{3}\left(A_1 + A_2 + A_3\right)$$

Instead, we must average with proper weights that represent the probability of a system in a particular microscopic state!

(I.e., not all microscopic states are equal)

$$A_{\text{macro}} = \rho_1 A_1 + \rho_2 A_2 + \rho_3 A_3 =$$

$$\rho_1(r_1, p_1)A_1(r_1, p_1) + \rho_2(r_2, p_2)A_2(r_2, p_2) + \rho_3(r_3, p_3)A_3(r_3, p_3)$$

Probability to find system in state C_1

How to apply to ensemble average?

- Similar method can be used to apply to integrate the ensemble average

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp$$

$$\rho(p,r) = \frac{1}{Q}\exp\left[-\frac{H(p,r)}{k_B T}\right]$$

"discrete"

$$< A >= \sum_{i=1}^{N_A} \frac{A\exp\left(-H(r_A,p_A)/\left(k_B T\right)\right)}{\sum_{i=1}^{N_A}\exp\left(-H(r_A,p_A)/\left(k_B T\right)\right)}$$

Computationally inefficient: If states are created "randomly" that have low probability….

- To be computationally more effective, need more complex iteration scheme (replace "*random sampling*" by "*importance sampling*")

Importance sampling

- **Core concept:** Picking states with a biased probability: Importance sampling (sampling the "correct" way…)

$$< A >= \sum_{i=1}^{N_A} \frac{A \exp\left(-H(r_A, p_A)/\left(k_B T\right)\right)}{\sum_{i=1}^{N_A} \exp\left(-H(r_A, p_A)/\left(k_B T\right)\right)}$$

$$< A >= \frac{1}{N_A} \sum_{i=1}^{N_A} A(r_A, p_A)$$

Corresponding to…

$$< A >= \frac{1}{3}\left(\rho_1 A_1 + \rho_2 A_2 + \rho_3 A_3\right) \rightarrow < A >= \frac{1}{3}\left(A_1 + A_2 + A_3\right)$$

Importance sampling

- **Core concept:** Picking states with a biased probability: Importance sampling (sampling the "correct" way…)

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp \qquad \rho(p,r) = \frac{1}{Q}\exp\left[-\frac{H(p,r)}{k_B T}\right]$$

Notice: Probability (and thus importance) related to energy of state

Importance sampling: Metropolis algorithm

- Leads to an appropriate "chain" of states, visiting each state with **correct probability**

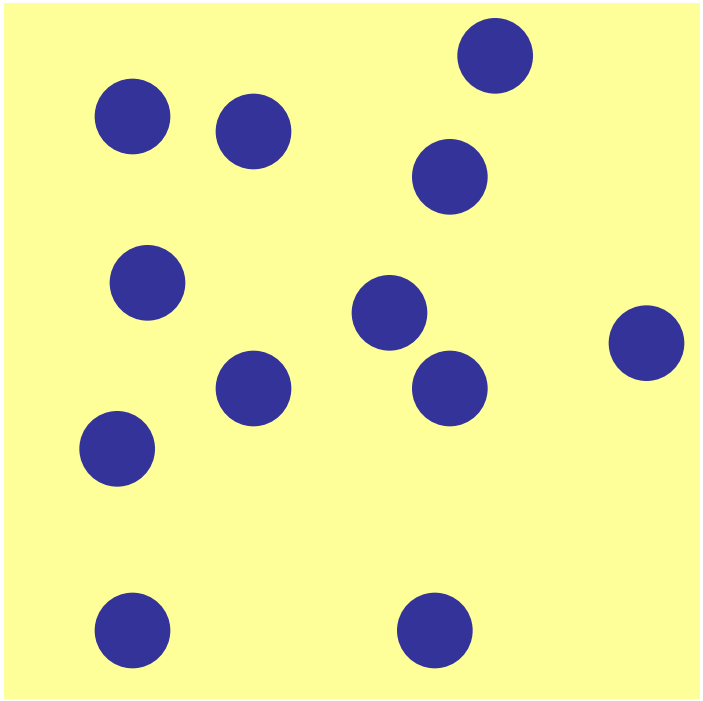
- Concept:

  - Pick random initial state
  - Move to trial states
  - Accept trial state with certain probability (based on knowledge about behavior of system, *i.e.*, energy states)

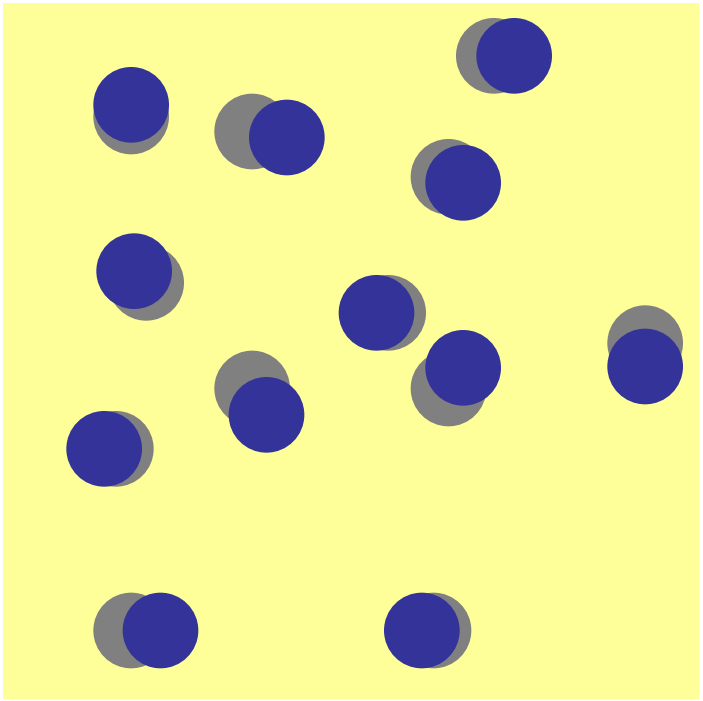
Original reference: *J. Chem. Phys.* **21**,1087, 1953

Metropolis-Hastings Algorithm

Concept: Generate set of random microscopic configurations
Accept or reject with certain scheme



State A



State B

Random move to new state B

Metropolis-Hastings Algorithm: *NVT*

Have: State A (initial state) + energy function $H(A)$

Step 1: Generate new state B (random move)

Metropolis-Hastings Algorithm: *NVT*

Have: State A **(initial state) + energy function** $H(A)$

Step 1: **Generate new state** B **(random move)**

Step 2: **if** $H(B)<H(A)$ **then** $a = 1$

else

Draw random number $0 < p < 1$

a = true[1]/false[0] for acceptance

"Downhill" moves always accepted

Metropolis-Hastings Algorithm: *NVT*

Have: State A **(initial state) + energy function** $H(A)$

Step 1: **Generate new state** B **(random move)**

Step 2: **if** $H(B)<H(A)$ **then** $a = 1$

else

a = true[1]/false[0] for acceptance

Draw random number $0 < p < 1$

if $p < \exp\left[-\dfrac{H(B)-H(A)}{k_B T}\right]$ $a = 1$

else

$a = 0$

endif

a=variable either 0 or 1 (used to detect acceptance

"Downhill" moves always accepted, uphill moves with finite ("thermal") probability

Metropolis-Hastings Algorithm: *NVT*

Have: State A **(initial state) + energy function** $H(A)$

Step 1: **Generate new state** B **(random move)**

Step 2: **if** $H(B)<H(A)$ **then** $a = 1$

a = true[1]/false[0] for acceptance

else

Draw random number $0 < p < 1$

if $p < \exp\left[-\dfrac{H(B)-H(A)}{k_B T}\right]$ $a = 1$

else

$a = 0$

endif

a=variable either 0 or 1 (used to detect acceptance of state B when a=1)

endif

Step 3: **if** $a = 1$ **then accept state** B

endif

Metropolis-Hastings Algorithm: *NVT*

Have: State A **(initial state) + energy function** $H(A)$

repeat N_A times

Step 1: **Generate new state** B **(random move)**

Step 2: **if** $H(B)<H(A)$ **then** $a = 1$

else

Draw random number $0 < p < 1$

if $p < \exp\left[-\frac{H(B)-H(A)}{k_B T}\right]$ $a = 1$

else

$a = 0$

endif

endif

Step 3: **if** $a = 1$ **then accept state** B

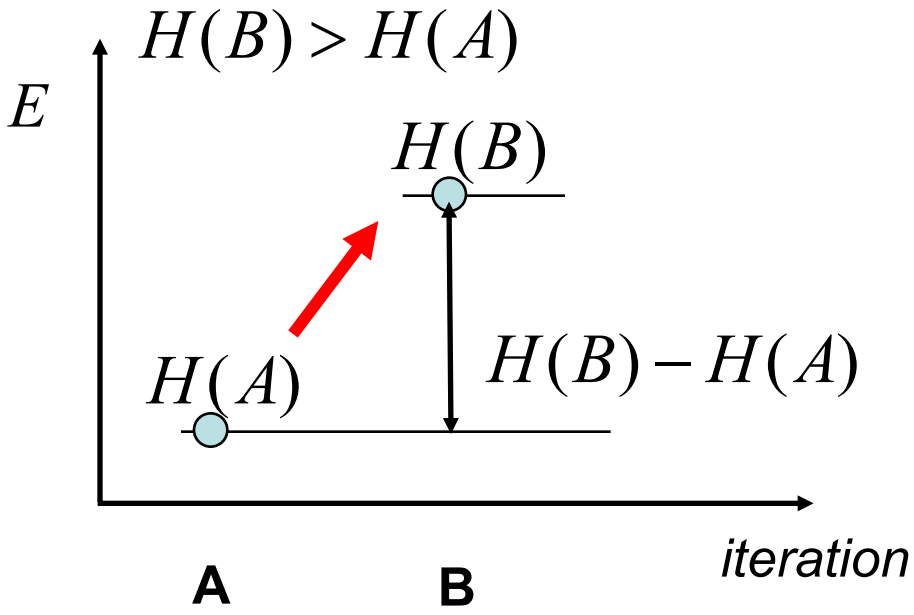
endif

a = true[1]/false[0] for acceptance

a=variable either 0 or 1 (used to detect acceptance of state B when a=1)

$$< A >= \frac{1}{N_A} \sum_{i=1..N_A} A(i)$$

Arrhenius law - explanation

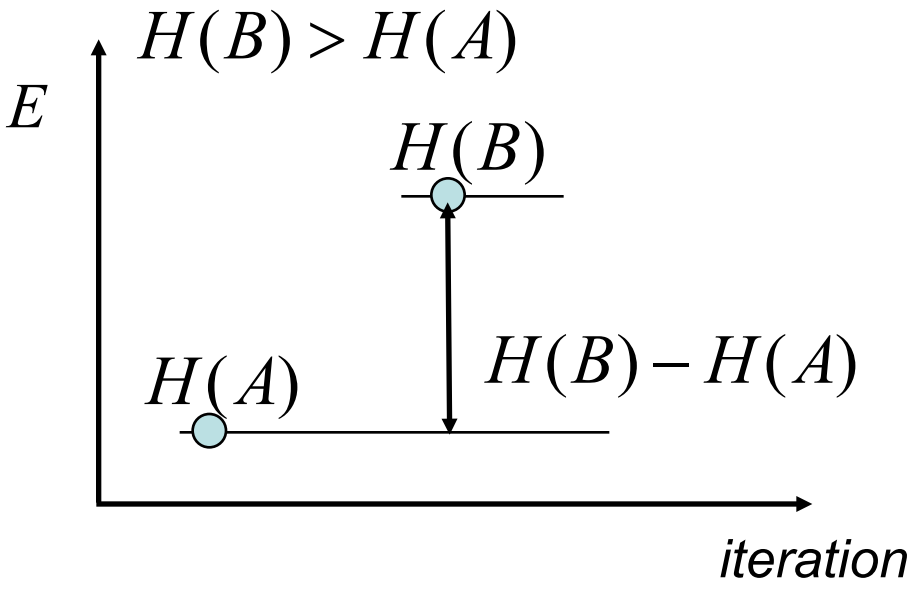


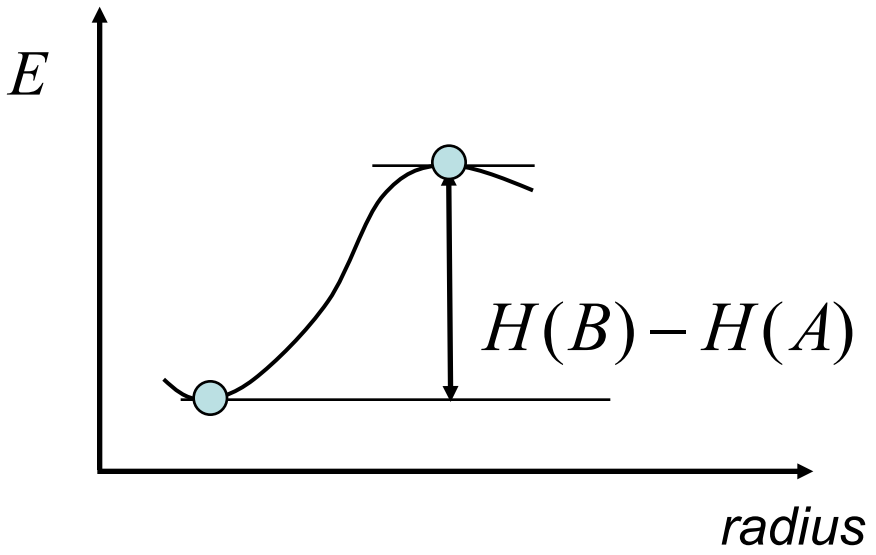
Consider two states, A and B

State B has higher energy than state A

Otherwise accepted anyway!

Arrhenius law - explanation





Energy difference between states A and B ("uphill")

Probability of success of overcoming the barrier at temperature T

$$\exp\left[-\frac{H(B)-H(A)}{k_B T}\right]$$

Arrhenius law - explanation

Probability of success of overcoming the barrier

$$\exp\left[-\frac{H(B)-H(A)}{k_B T}\right]$$

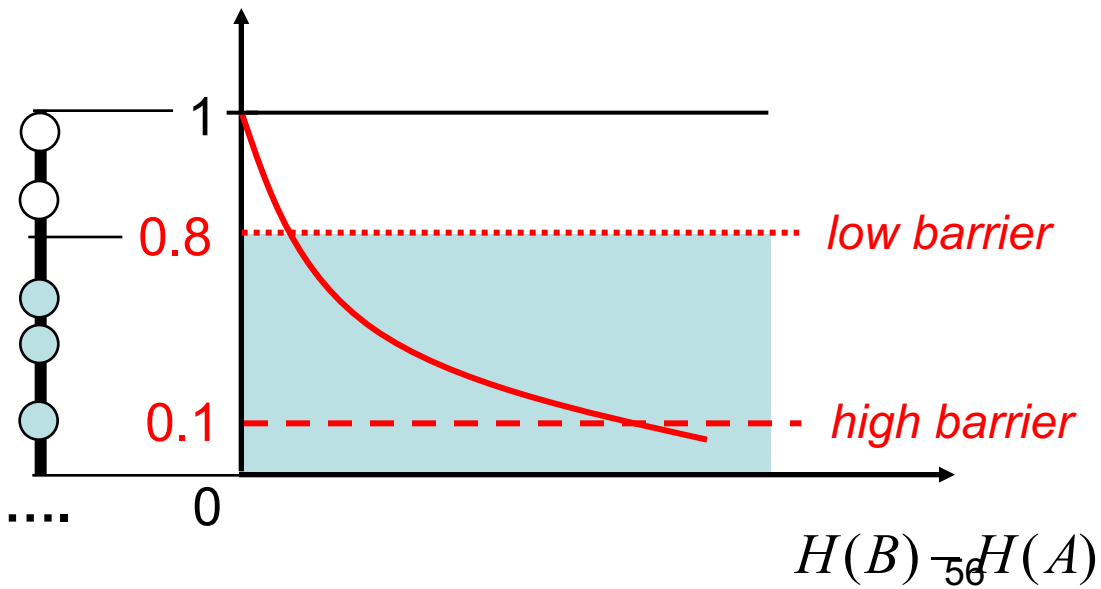
Random number $0 < p < 1$

(equal probability to draw any number between 0 and 1)

Acceptance if:

$$p < \exp\left[-\frac{H(B)-H(A)}{k_B T}\right]$$

E.g. when $\exp(..) = 0.8$ most choices for p will be below, that is, higher chance for acceptance



Play “1D darts”

Summary: Metropolis-Hastings Algorithm

Have: State A (initial state) + energy function $H(A)$

repeat N_A times

Step 1: **Generate new state B (random move)**

Step 2: **if** $H(B)<H(A)$ **then** $a = 1$

else

Draw random number $0 < p < 1$

if $p < \exp\left[-\frac{H(B)-H(A)}{k_B T}\right]$ $a = 1$

else

$a = 0$

endif

endif

Step 3: **if** $a = 1$ **then accept state** B

endif

a = true[1]/false[0] for acceptance

a=variable either 0 or 1 (used to detect acceptance of state B when a=1)

$$< A >= \frac{1}{N_A} \sum_{i=1..N_A} A(i)$$

Summary: MC scheme

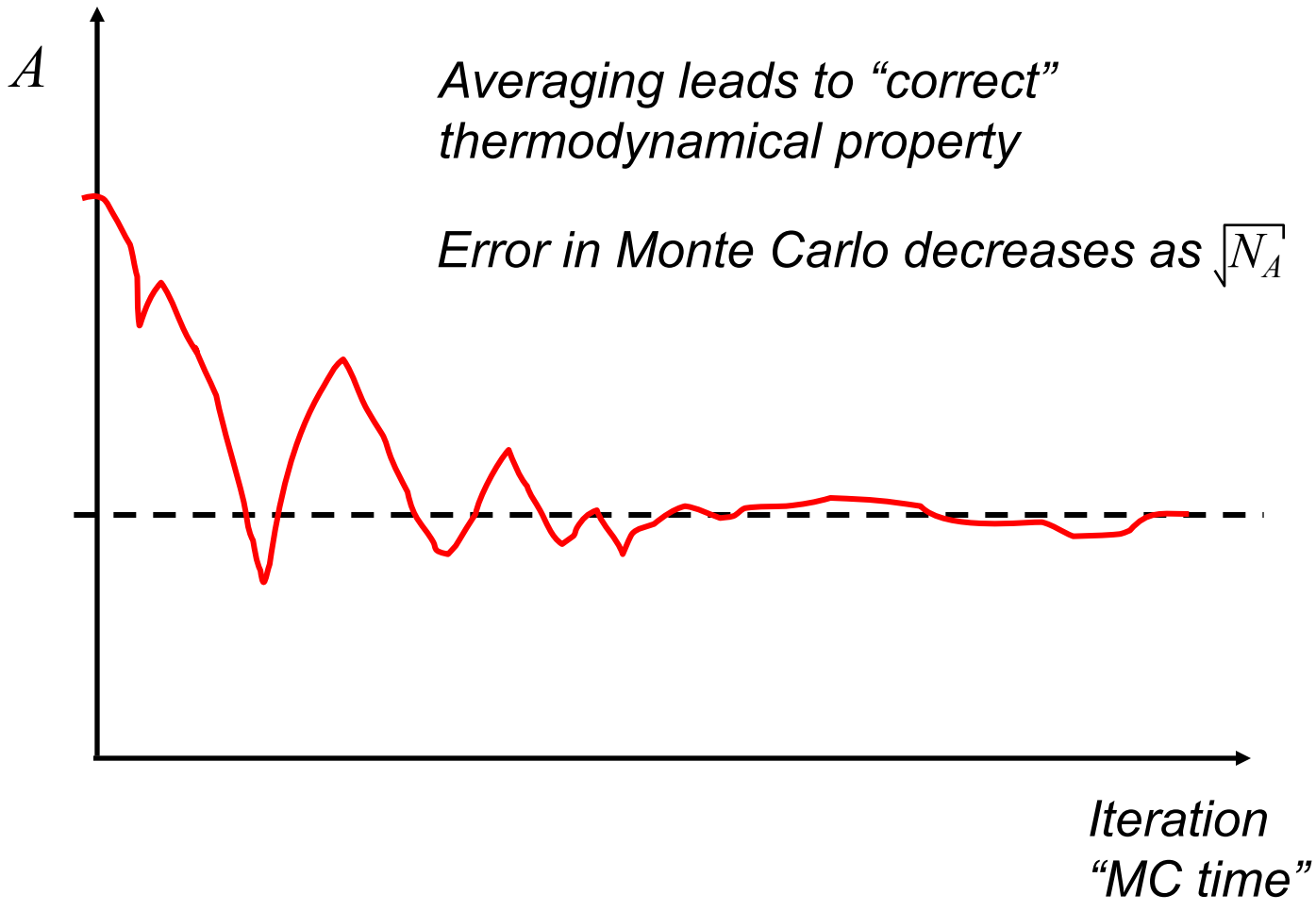
Have achieved:

$$< A >= \int_p \int_r A(p,r)\rho(p,r)drdp \quad \longleftrightarrow \quad \boxed{< A > \frac{1}{N_A} \sum_{i=1..N_A} A_i}$$

Note:

- Do not need forces between atoms (for accelerations)
- Only valid for equilibrium processes

Property calculation with MC: example



Other ensembles/applications

- Other ensembles carried out by modifying the acceptance criterion (in Metropolis-Hastings algorithm), e.g. *NVT, NPT;* **goal is to reach the appropriate distribution of states according to the corresponding probability distributions**

- Move sets can be adapted for other cases, e.g. not just move of particles but also **rotations of side chains** (=rotamers), **torsions**, etc.

  E.g. application in protein folding problem when we'd like to determine the 3D folded structure of a protein in thermal equilibrium, *NVT*

After: R.J. Sadus

3.3 Complex moves

Complex moves

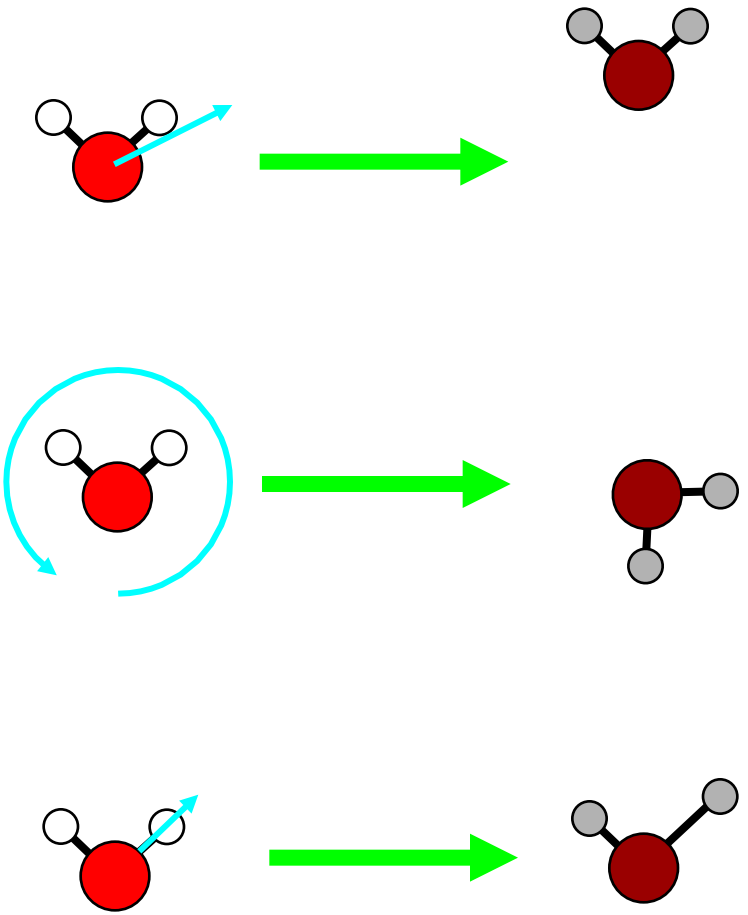
- Move sets can be adapted for other cases, e.g. not just move of particles but also **rotations of side chains** (=rotamers), **torsions**, etc.

  E.g. application in protein folding problem when we'd like to determine the 3D folded structure of a protein in thermal equilibrium

After: R.J. Sadus

Possible Monte Carlo moves

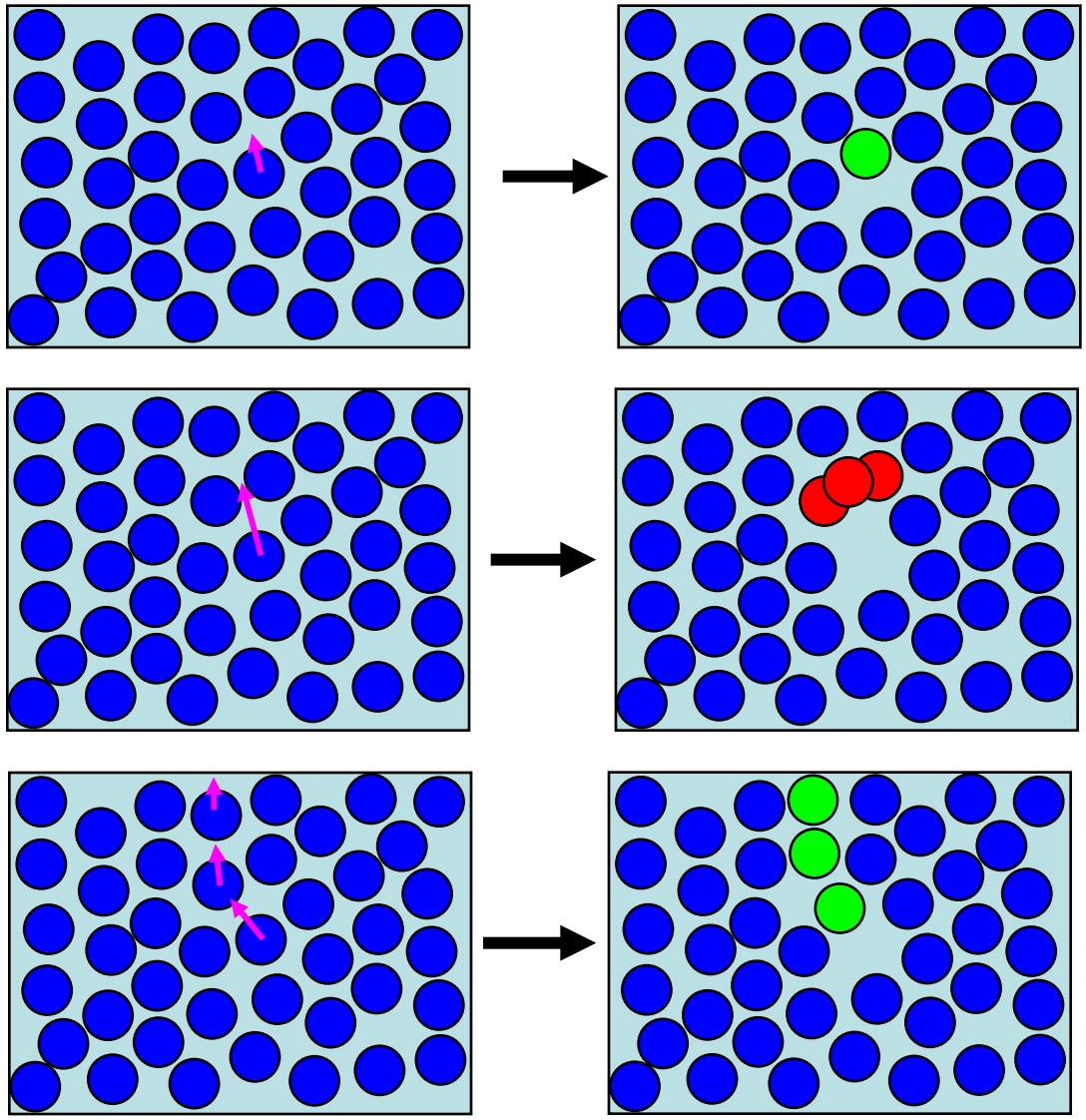
- Trial moves
  - Rigid body translation
  - Rigid body rotation
  - Internal conformational changes (soft vs. stiff modes)
  - Titration/electronic states
  - …
- Questions:
  - How “big” a move should we take?
  - Move one particle or many?



After N. Baker (WUSTL)

Monte Carlo moves

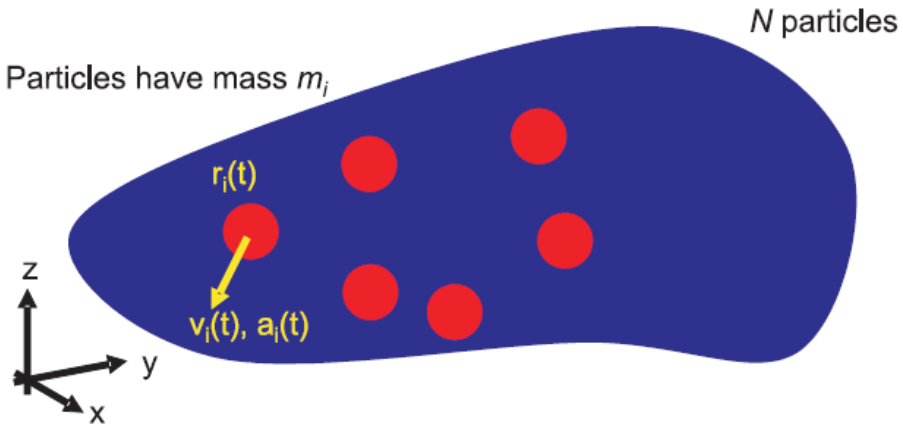
- How “big” a move should we take?
  - **Smaller moves**: better acceptance rate, slower sampling
  - **Bigger moves**: faster sampling, poorer acceptance rate
- Move one particle or many?
  - Possible to achieve more efficient sampling with *correct* multi-particle moves
  - One-particle moves must choose particles at random



After N. Baker (WUSTL)

4. How to model chemical interactions
INTRODUCTION

Molecular dynamics: A “bold” idea



$$r_i(t_0+\Delta t) = -r_i(t_0-\Delta t) + 2r_i(t_0)\Delta t + a_i(t_0)(\Delta t)^2 + \ldots$$

Positions at t_0-Δt

Positions at t_0

Accelerations at t_0

$$a_i = f_i / m$$

Forces between atoms… how to obtain?

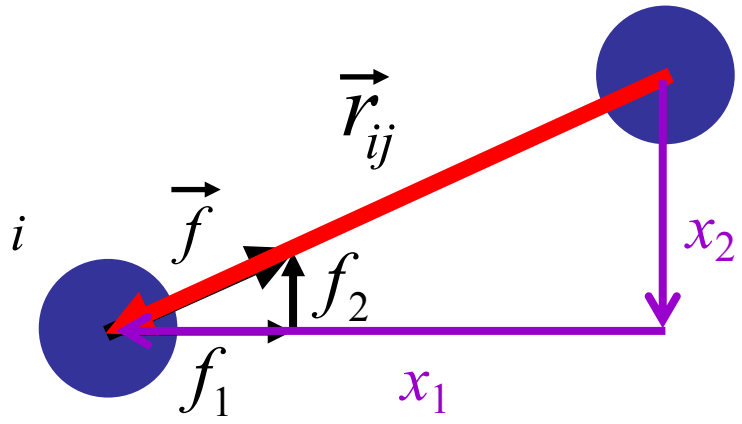
How are forces calculated?

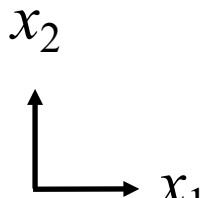
Force magnitude: Derivative of potential energy with respect to atomic distance

$$f = -\frac{\mathrm{d}U(r)}{\mathrm{d}r}$$

To obtain force vector f_i, take projections into the three axial directions

$$f_i = f\frac{x_i}{r}$$

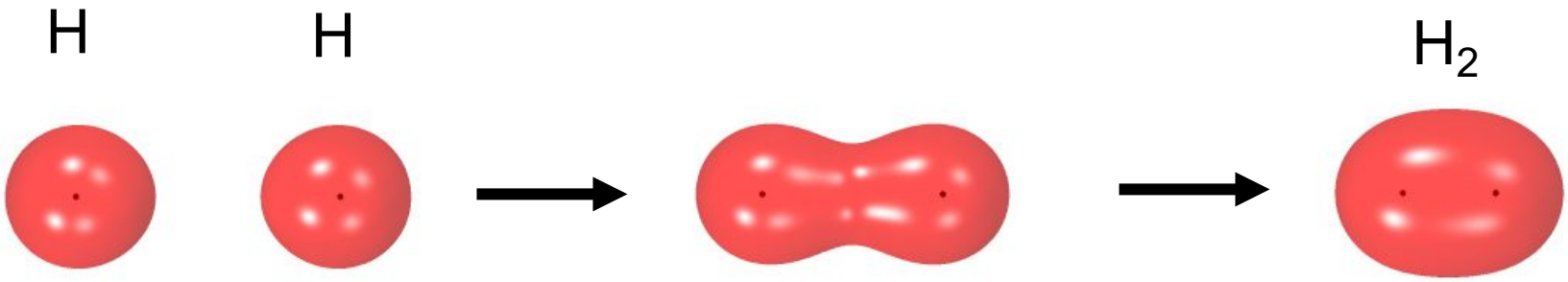




Atomic interactions – quantum perspective

How electrons from different atoms interact defines nature of chemical bond

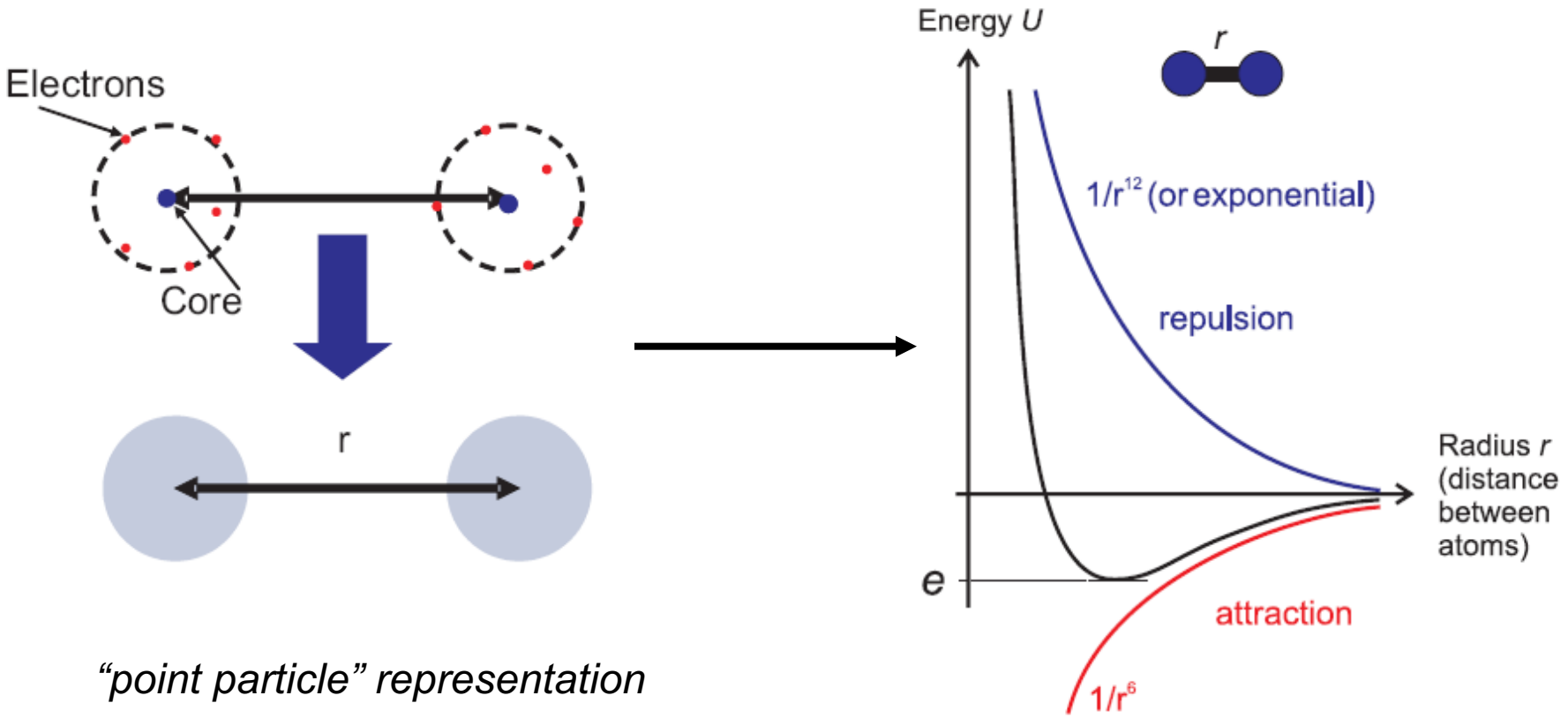
Density distribution of electrons around a H-H molecule



Much more about it in part II

http://winter.group.shef.ac.uk/orbitron/MOs/H2/1s1s-sigma/index.html

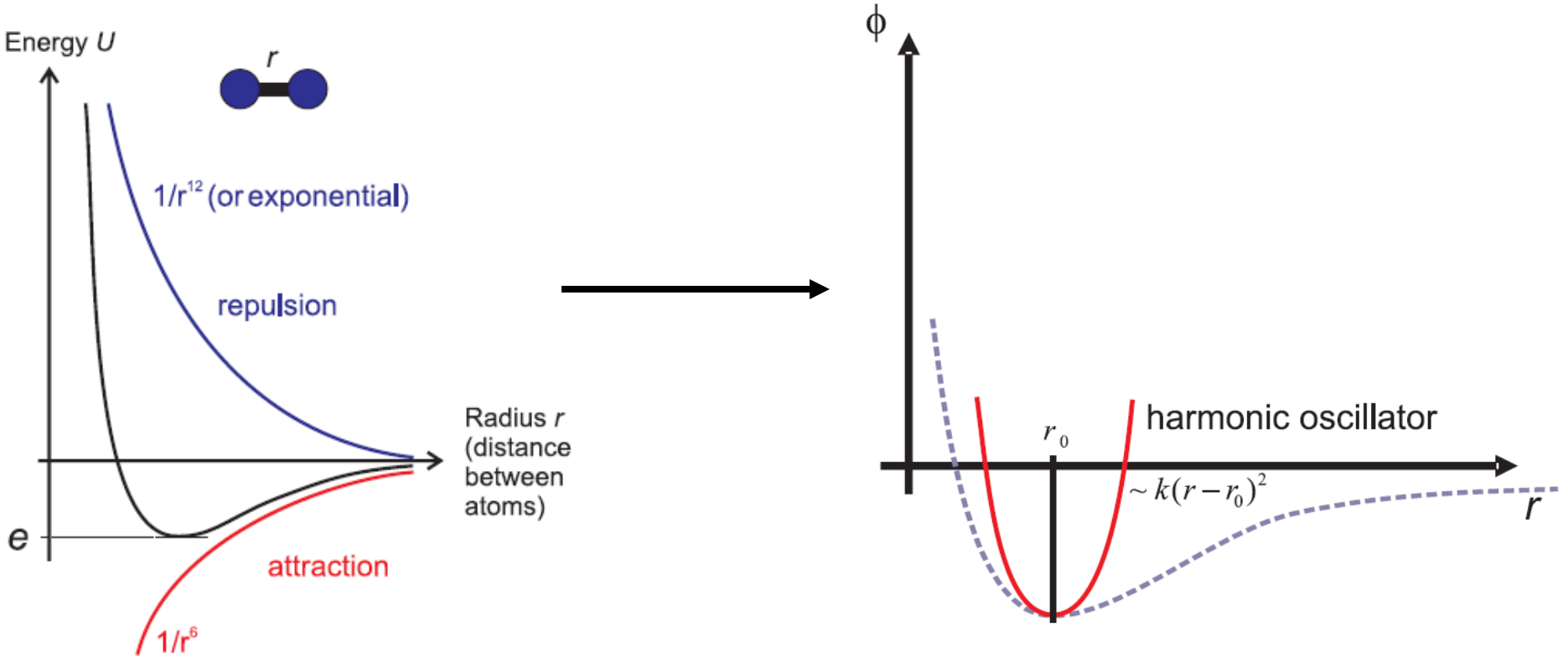
Concept: Interatomic potential



"point particle" representation

Attraction: Formation of chemical bond by sharing of electrons
Repulsion: Pauli exclusion (too many electrons in small volume)

Interatomic bond - model



Attraction: Formation of chemical bond by sharing of electrons
Repulsion: Pauli exclusion (too many electrons in small volume)

4.1 Different types of chemical bonds

Atomic interactions – different types of chemical bonds

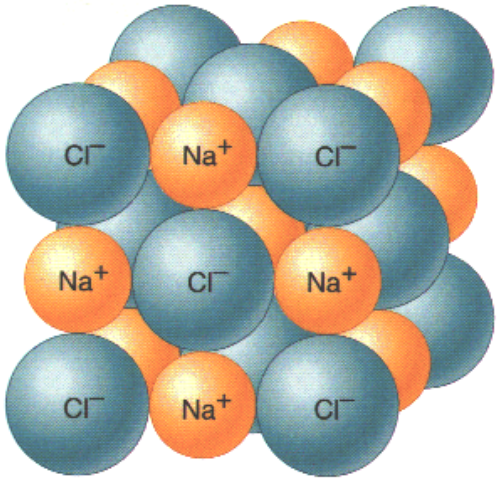
- **Primary bonds ("strong")**
  - Ionic (ceramics, quartz, feldspar - **rocks**)
  - Covalent (**silicon**)
  - Metallic (copper, nickel, **gold**, silver)
    (high melting point, 1000-5,000K)

- **Secondary bonds ("weak")**
  - Van der Waals (**wax**, low melting point)
  - Hydrogen bonds (proteins, **spider silk**)
    (melting point 100-500K)

- Ionic: Non-directional (point charges interacting)
- Covalent: Directional (bond angles, torsions matter)
- Metallic: Non-directional (electron gas concept)

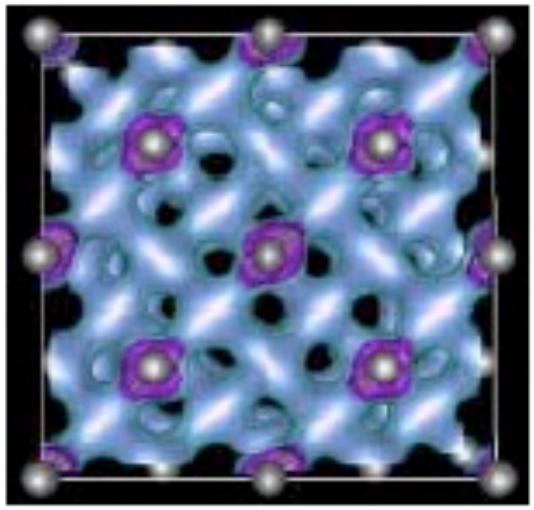
Difference of material properties originates from different atomic interactions

Types of bonding (illustrations)



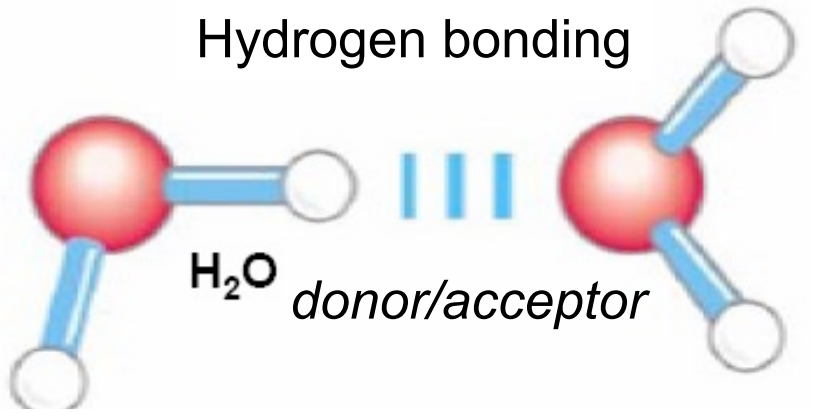
charge
+/-

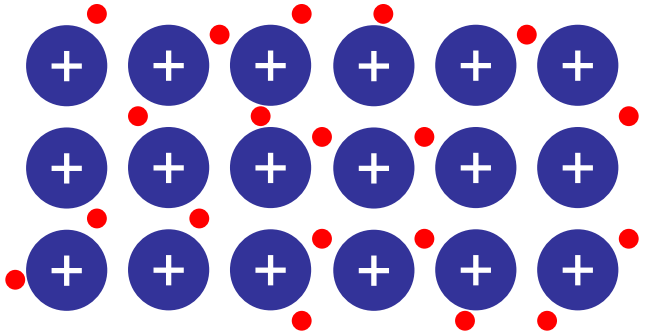
Ionic bonding



electron density (localized!)

Covalent bonding





Metallic bonding

After: Pavel Korzhavyi, *Royal Institute of Technology (KTH)*

Wax

Soft, deformable, does not break under deformation

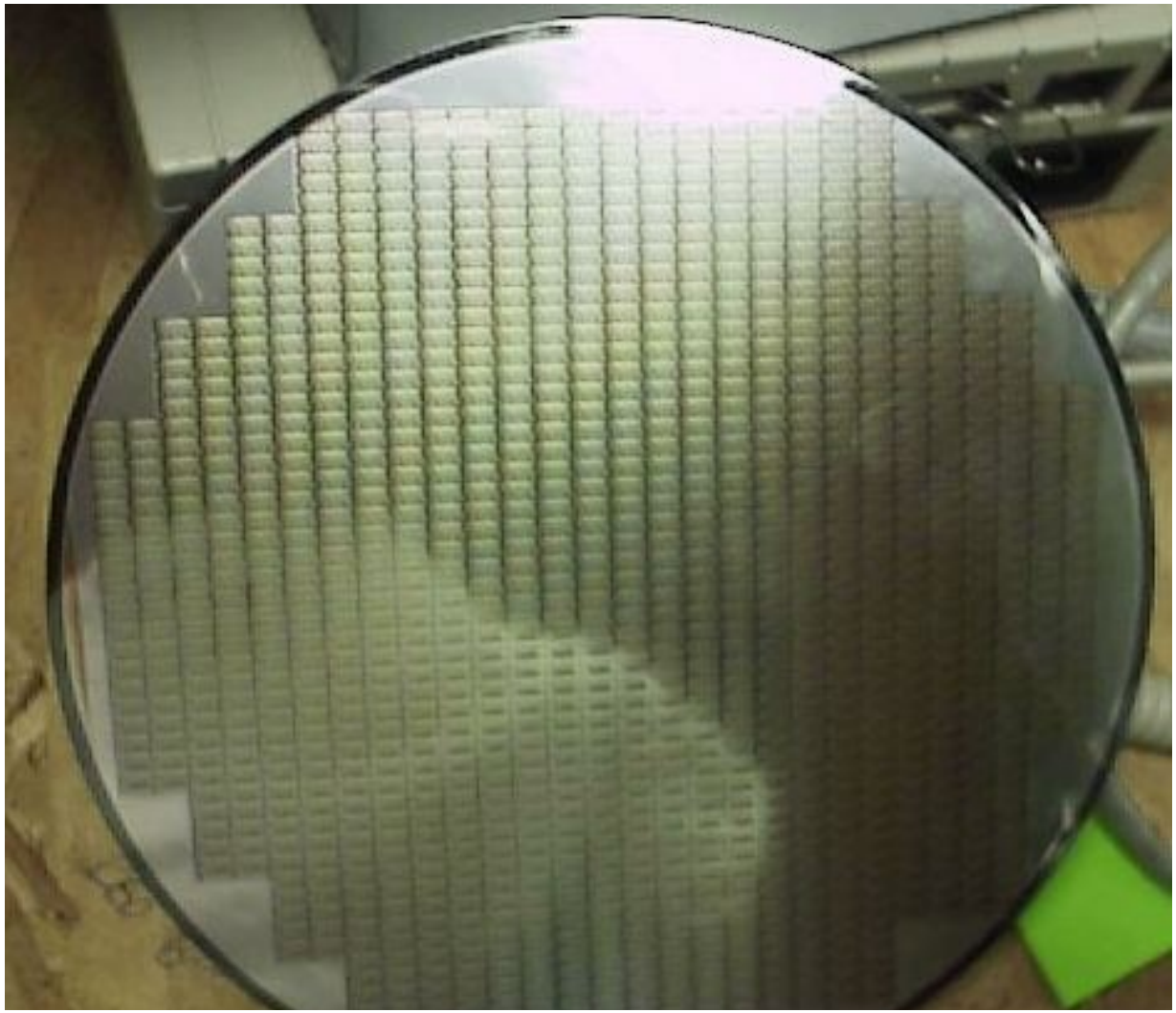
Rocks

Quite brittle (breaks e.g. during earthquake)

Gold

Very “soft” metal, deformable, high density

Silicon



Very brittle – shatters into many pieces if dropped

Spider web

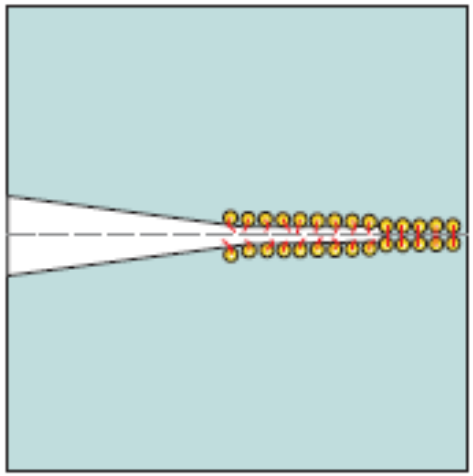
Very extensible, deformation, yet very strong (similar to steel)

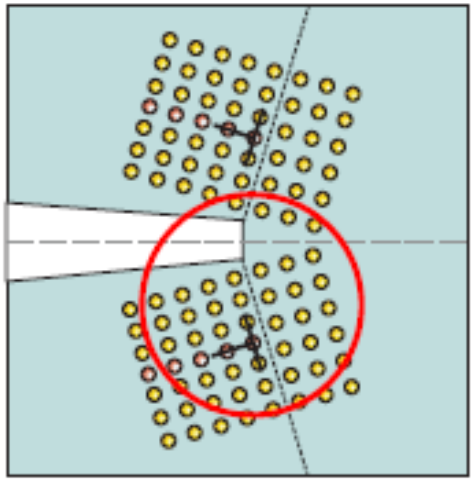
Brittle or ductile?

brittle

ductile

Glass,
Polymers,
Ice…





Copper,
Gold,
…

shear load

Outline

- Goal: model chemical bonds with the objective to enable force calculation (basic MD algorithm) or energy calculation (TODAY, MC)

- **Two-step approach**:

  1. Define energy landscape, *i.e.* defines how distance between particles controls the energy stored in the bond

  2. Then take derivatives to obtain forces, to be used in the MD algorithm

"Modeling and simulation" paradigm:

- First, develop mathematical expressions (modeling)
- Second, use model in numerical solution (simulation, =MD)

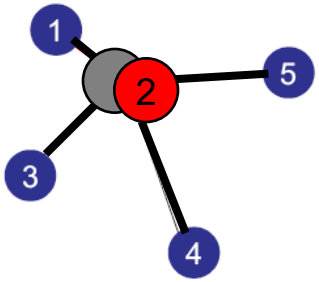
Models for atomic interactions

- Define interatomic potentials that describe the energy of a set of atoms as a function of their coordinates r:

$$r = \{\vec{r}_j\} \quad j = 1..N$$

$$U_{total} = U_{total}(r)$$

Depends on position of all other atoms



$$\boxed{\vec{F}_i = -\nabla_{\vec{r}_i} U_{total}(r)} \quad i = 1..N$$

$$\nabla_{\vec{r}_i} = \left(\frac{\partial}{\partial r_{1,i}}, \frac{\partial}{\partial r_{2,i}}, \frac{\partial}{\partial r_{3,i}} \right)$$

Change of potential energy due to change of position of particle i ("gradient")