Various topics

Coarse graining; hard bodies; Monte Carlo techniques

Marcus Elstner and Tomáš Kubař

June 15, 2016

United-atom force fields

- early biomolecular FF (e.g. Weiner84), popular in the 1990's
- hydrogen atoms considered as condensed to the heavy atom
- mass and charge represent such a group of atoms as a whole
- number of atoms reduced considerably relative to all-atom FF

United-atom force fields

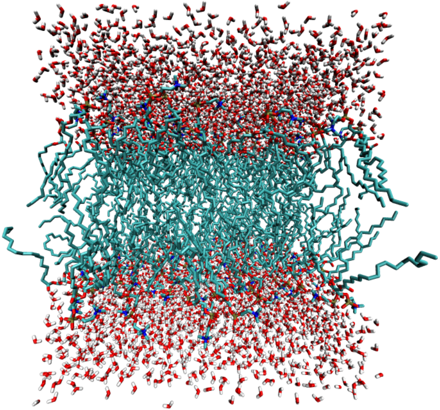
- early biomolecular FF (e.g. Weiner84), popular in the 1990's
- hydrogen atoms considered as condensed to the heavy atom
- mass and charge represent such a group of atoms as a whole
- number of atoms reduced considerably relative to all-atom FF
- good for non-polar C–H bonds – so CH_3 is one united atom

United-atom force fields

- early biomolecular FF (e.g. Weiner84), popular in the 1990's
- hydrogen atoms considered as condensed to the heavy atom
- mass and charge represent such a group of atoms as a whole
- number of atoms reduced considerably relative to all-atom FF
- good for non-polar C–H bonds – so CH_3 is one united atom
- polar O–H group by a single 'atom' – too crude
  → only non-polar hydrogens usually condensed with heavy

United-atom force fields

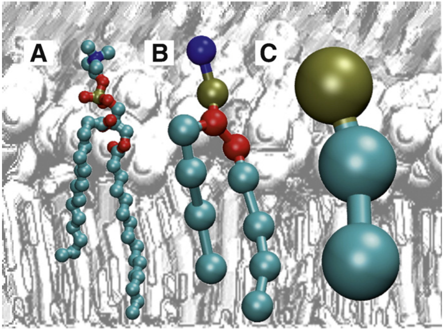
still sometimes used e.g. for lipids – each CH_2 is a united atom



(simulation of a DOPC bilayer in water – Berger FF for the lipid)

from the website of Rainer Böckmann

United-atom and coarse-grained force fields



(A) united-atom, (B) specific and (C) generic coarse-grained

from Marrink et al., Biochim. Biophys. Acta 2009

Coarse-grained models

Coarse graining – an advanced and sophisticated approach to reduce the computational expense of simulations

Coarse-grained models

Coarse graining – an advanced and sophisticated approach
to reduce the computational expense of simulations

The same idea – reduction of the number of particles
Considered are particles composed of several atoms – beads
Fewer inter-particle interactions → reduced computational expense

Coarse-grained models

Coarse graining – an advanced and sophisticated approach
to reduce the computational expense of simulations

The same idea – reduction of the number of particles
Considered are particles composed of several atoms – beads
Fewer inter-particle interactions → reduced computational expense

The necessary parameters – often obtained
by fitting to all-atom force fields

Coarse-grained models

Every bead usually represents several atoms,
and a molecule is composed of several beads

Solvent – e.g. a ‘water bead’ composed of 4 H_2O molecules

Coarse-grained models

Every bead usually represents several atoms,
    and a molecule is composed of several beads
Solvent – e.g. a ‘water bead’ composed of 4 H_2O molecules

Some of the transferability of all-atom FF is lost
    – e.g. secondary structure of proteins is fixed with Martini FF
Also, hydrogen bonding cannot be described with beads!
    solution – compensation with Lennard-Jones contributions

Coarse-grained models

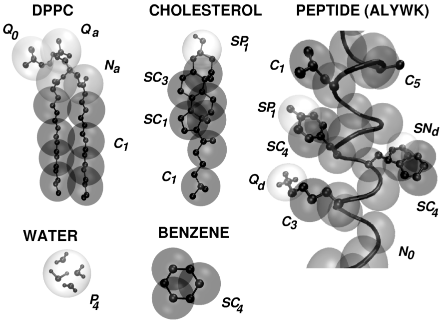
Every bead usually represents several atoms,
and a molecule is composed of several beads
Solvent – e.g. a ‘water bead’ composed of 4 H_2O molecules

Some of the transferability of all-atom FF is lost
– e.g. secondary structure of proteins is fixed with Martini FF
Also, hydrogen bonding cannot be described with beads!
solution – compensation with Lennard-Jones contributions

Especially useful for large-scale conformational transitions involving
exceedingly large molecular systems, excessive time scales,
or both

Martini force field

mapping of beads onto molecular fragments with Martini FF

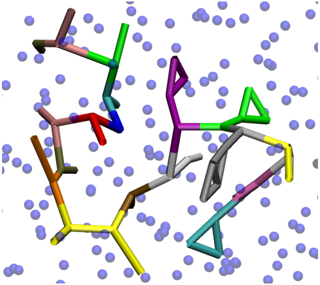


- 3 to 4 heavy atoms compose one bead ('4-to-1 mapping')
- mass of beads – 72 u (= 4 H_2O), or 45 u in ring structures

from the Martini website

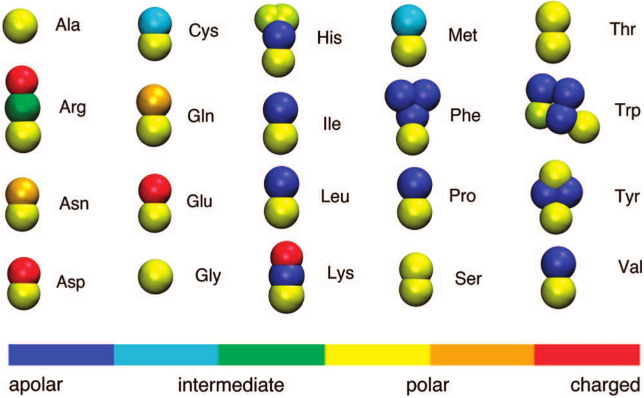
Martini force field

a solvated peptide with Martini FF



Martini force field

the amino acids:



from Monticelli et al., J. Chem. Theory Comput. 2008

Acceleration of the simulation

Why does a coarse-grained simulation run faster?

- smaller number of particles → fewer interactions to compute

Acceleration of the simulation

Why does a coarse-grained simulation run faster?

- smaller number of particles → fewer interactions to compute
- long integration time step due to large masses of beads
  - 25 fs with Martini (i.e. 100 fs effectively, see below)

Acceleration of the simulation

Why does a coarse-grained simulation run faster?

- smaller number of particles → fewer interactions to compute
- long integration time step due to large masses of beads
  - 25 fs with Martini (i.e. 100 fs effectively, see below)
- FF often constructed for use with faster simulation algorithms
  - e.g. cut-off for electrostatics with Martini

Acceleration of the simulation

Why does a coarse-grained simulation run faster?

- smaller number of particles → fewer interactions to compute
- long integration time step due to large masses of beads
  - 25 fs with Martini (i.e. 100 fs effectively, see below)
- FF often constructed for use with faster simulation algorithms
  - e.g. cut-off for electrostatics with Martini
- smaller number of DOF → smoother free energy surfaces
  → fewer barriers → acceleration of all processes
  (by a factor of 3 to 8 for Martini, but not uniformly!
  – factor of 4 for acceleration of diffusion in water)

Acceleration of the simulation

Why does a coarse-grained simulation run faster?

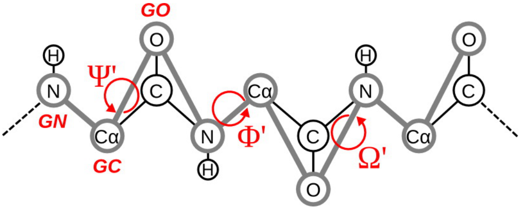
- smaller number of particles → fewer interactions to compute
- long integration time step due to large masses of beads
  - 25 fs with Martini (i.e. 100 fs effectively, see below)
- FF often constructed for use with faster simulation algorithms
  - e.g. cut-off for electrostatics with Martini
- smaller number of DOF → smoother free energy surfaces
  → fewer barriers → acceleration of all processes
  (by a factor of 3 to 8 for Martini, but not uniformly!
  – factor of 4 for acceleration of diffusion in water)

"… length and time scales that are 2 to 3 orders of magnitude larger compared to atomistic simulations, providing a bridge between the atomistic and the mesoscopic scale."

Coarse-grained models

SIRAH force field

- somewhat less coarse-grained, closer to united-atom
- representation of backbone dihedral angles retained

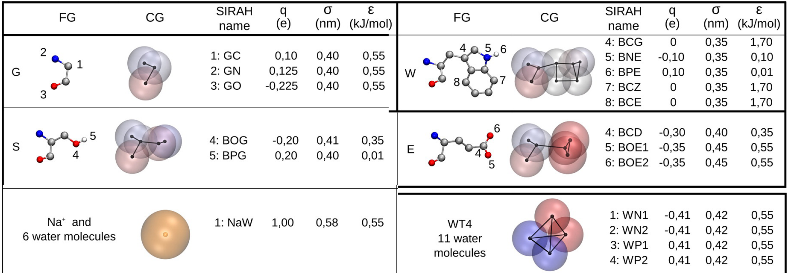


from Pantano et al., J. Chem. Theory. Comput. 2015

Coarse-grained models

SIRAH force field

- less coarse-grained → possibly improved transferability
- explicit solvent, long-range electrostatics (no cut-off)



	FG	CG	SIRAH name	q (e)	σ (nm)	ε (kJ/mol)
G			1: GC 2: GN 3: GO	0,10 0,125 -0,225	0,40 0,40 0,40	0,55 0,55 0,55
S			4: BOG 5: BPG	-0,20 0,20	0,41 0,40	0,35 0,01
	Na⁺ and 6 water molecules		1: NaW	1,00	0,58	0,55

	FG	CG	SIRAH name	q (e)	σ (nm)	ε (kJ/mol)
W			4: BCG 5: BNE 6: BPE 7: BCZ 8: BCE	0 -0,10 0,10 0 0	0,35 0,35 0,35 0,35 0,35	1,70 0,10 0,01 1,70 1,70
E			4: BCD 5: BOE1 6: BOE2	-0,30 -0,35 -0,35	0,40 0,45 0,45	0,35 0,55 0,55
	WT4 11 water molecules		1: WN1 2: WN2 3: WP1 4: WP2	-0,41 -0,41 0,41 0,41	0,42 0,42 0,42 0,42	0,55 0,55 0,55 0,55

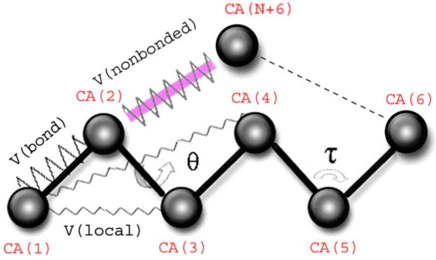
- illustration – different compromises may be made

from Pantano et al., J. Chem. Theory. Comput. 2015

Coarse-grained models

VAMM force field for proteins

- every amino acid represented by a single bead at C_α



- more coarse-grained than Martini

from Korkut & Hendrickson 2009

MD simulation of hard bodies

first MD simulation of a system in the condensed phase

- used the model of hard spheres
  (Alder & Wainwright, J. Chem. Phys. 1957)
- first step from the ideal gas towards realistic molecules
- valuable tool in statistical thermodynamics
  → equations of state and virial expansions

The hard-sphere potential

- pairwise potential
- potential energy of a system of two hard spheres with radius R
  - is zero for distances larger than the diameter of the spheres
  - is infinity for shorter distances, when the spheres overlap:

$$V(r) = \begin{cases} 0 & \text{if } r > 2R \\ +\infty & \text{otherwise} \end{cases}$$

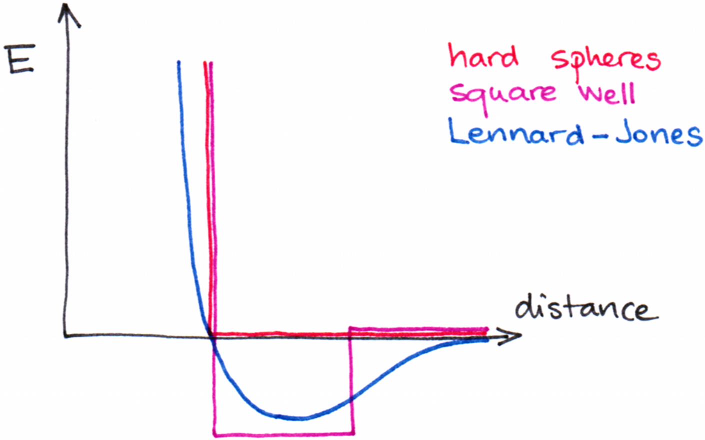
The hard-sphere potential

- pairwise potential
- potential energy of a system of two hard spheres with radius R
  - is zero for distances larger than the diameter of the spheres
  - is infinity for shorter distances, when the spheres overlap:

$$V(r) = \begin{cases} 0 & \text{if } r > 2R \\ +\infty & \text{otherwise} \end{cases}$$

- is discontinuous → not differentiable
- different from potentials typically used in biomolecular simulation

The hard-sphere potential



The square-well potential

a more realistic description preserving the simplicity of the model?

square well model

- region of negative potential energy (attractive interaction) starting at the contact distance $2R$
- goes in the direction of the Lennard-Jones potential, which describes nonpolar fluids very well

Hard convex bodies

- another extension used in statistical thermodynamics
- potential energy function is discontinuous, still:

  zero if the bodies do not intersect; infinity if they do

Hard convex bodies

- another extension used in statistical thermodynamics
- potential energy function is discontinuous, still:
  zero if the bodies do not intersect; infinity if they do
- enhancement – the bodies are not spherical anymore,
  but rather ellipsoidal or polyhedral
- may describe e.g. diatomic molecules better than hard sphere

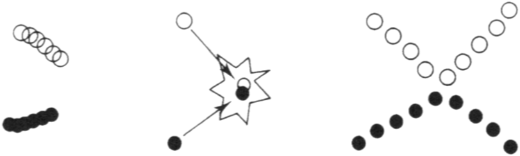
Simulation protocol

propagation of Newton's EOM with e.g. Verlet integrator
  – continuous and smooth potential required
otherwise – sudden 'jumps' in forces lead to unstable simulations,
  or at least wrong sampling of the configuration space

Simulation protocol

propagation of Newton's EOM with e.g. Verlet integrator
– continuous and smooth potential required

otherwise – sudden 'jumps' in forces lead to unstable simulations,
or at least wrong sampling of the configuration space



reprinted from Leach, Molecular Modelling

Simulation protocol

Hard spheres cannot be simulated with a usual integrator
– explosions caused by sudden clashes of atoms would occur
(similar to those in usual MD simulations with too large Δt)
However, with hard spheres, any arbitrarily short Δt is ‘too long’

Simulation protocol

Hard spheres cannot be simulated with a usual integrator
  – explosions caused by sudden clashes of atoms would occur
  (similar to those in usual MD simulations with too large Δt)
However, with hard spheres, any arbitrarily short Δt is 'too long'

What would a simulation of hard spheres with Verlet look like?
  There are no forces in any initial configuration,
  and so the spheres move with their initial velocities
  until, all of a sudden, two spheres start to overlap.
  The energy and forces are infinite, and the simulation crashes.

Simulation protocol

The protocol has to be adjusted to the discontinous potential
– event-driven protocol

The spheres move along straight lines between collisions, which are perfectly elastic and instantaneous

Simulation protocol

The protocol has to be adjusted to the discontinous potential
– event-driven protocol

The spheres move along straight lines between collisions,
which are perfectly elastic and instantaneous

1. Identify the next pair of spheres to collide,
   and calculate when this collision will occur
2. Calculate the positions of all spheres at the collision time –
   conservation of linear momentum and of kinetic energy
3. Determine the new velocities of the two spheres after collision
4. Repeat from start

Simulation protocol

No further approximations are involved in this protocol
→ simulation will be exact within the model of hard spheres

Note: With continuous potentials, we had to make approximations, like a stepwise integration of the eqns of motion

Simulation protocol

No further approximations are involved in this protocol
→ simulation will be exact within the model of hard spheres

Note: With continuous potentials, we had to make approximations, like a stepwise integration of the eqns of motion

Potential energy – constant (zero) throughout the simulation
Conservation of total energy → conservation of kinetic energy
→ temperature is constant in any hard-spheres simulation

Monte Carlo simulation

The main objective of molecular dynamics –
    mostly not to study how the molecular system evolves in time,
    rather to generate configurations of the system
    (sampling → calculation of thermodynamic quantites)
MD is not the only possibility to do this . . .

Monte Carlo simulation

The main objective of molecular dynamics –
mostly not to study how the molecular system evolves in time,
rather to generate configurations of the system
(sampling → calculation of thermodynamic quantites)
MD is not the only possibility to do this . . .

Another possibility – Monte Carlo methods (MC),
which involve random number generators Actually, first
computer simulations of molecular systems were MC
(Metropolis et al., J. Chem. Phys. 1953)

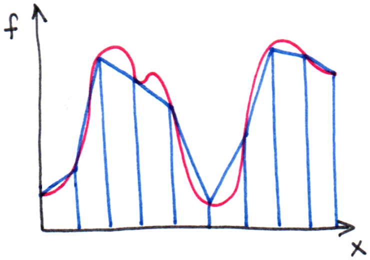
Monte Carlo integration

Major goal of molecular simulation – calculation of thermodynamic properties – integration (formally)

Can we use a method based on randomness for integration?

Monte Carlo integration

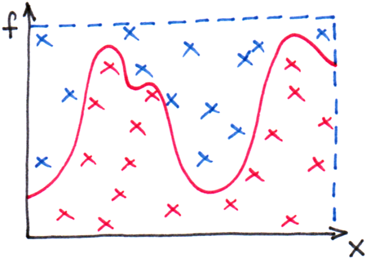
Major goal of molecular simulation – calculation
of thermodynamic properties – integration (formally)
Can we use a method based on randomness for integration?



Possibility – trapezium rule
– comes intro trouble
for functions of many variables
– we always have many variables
molecular systems

Monte Carlo integration

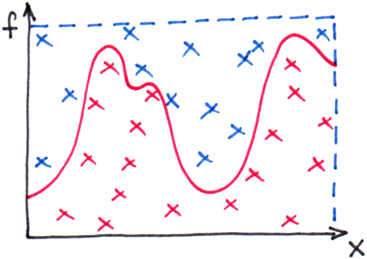
Major goal of molecular simulation – calculation
of thermodynamic properties – integration (formally)
Can we use a method based on randomness for integration?



Alternatively
- generate N points randomly
- count points (n) under curve
- area under the curve relative to the rectangle $\approx n/N$

Monte Carlo integration

Major goal of molecular simulation – calculation
of thermodynamic properties – integration (formally)
Can we use a method based on randomness for integration?



Alternatively
- generate N points randomly
- count points (n) under curve
- area under the curve relative to the rectangle $\approx n/N$

Apply the Monte Carlo idea to calculate π as follows:

Generate pairs of random number between 0 and 1 (x, y).
Count the pairs for which $x^2 + y^2 < 1$, i.e. the point (x, y) lies within the circle centered at (0,0) with a radius of 1.

The ratio of this number to the total number of pairs approaches $\pi/4$.

Monte Carlo integration

Importantly:
Extension of this ansatz to many dimensions is straightforward
– useful for studies of molecular systems

Monte Carlo integration

Importantly:
Extension of this ansatz to many dimensions is straightforward
– useful for studies of molecular systems

Groundbreaking idea (Metropolis):
Generate the configurations with the right probability,
creating the correct thermodynamic (e.g. canonical) ensemble

Such importance sampling will make it trivial to average
thermodynamics quantities over the generated configurations

Metropolis' method

Typical MC simulation of a molecular system
- a sequence of configurations is generated in an iterative way
- in every iteration, one configuration is produced.

Usually:

1. A trial configuration is constructed from the current one
   by randomly shifting one randomly chosen particle (atom).

Metropolis' method

Typical MC simulation of a molecular system

- a sequence of configurations is generated in an iterative way
- in every iteration, one configuration is produced.

Usually:

1. A trial configuration is constructed from the current one
   by randomly shifting one randomly chosen particle (atom).
2. It is tested if this configuration shall be accepted or not.
   For this, potential energy of the entire system is calculated.
   (possible optimization – only small part of the system changes,
   → only a small fraction of the interactions changes)

Metropolis' method

Trial set of coordinates is calculated with random $\xi_{x,z,y} \in (0,1)$:

$$x_{\text{trial}} = x + (2\xi_x - 1)\cdot\delta r$$
$$y_{\text{trial}} = y + (2\xi_y - 1)\cdot\delta r$$
$$z_{\text{trial}} = z + (2\xi_z - 1)\cdot\delta r$$

δr – maximum allowed displacement

Metropolis' method

Trial set of coordinates is calculated with random $\xi_{x,z,y} \in (0,1)$:

$$\begin{aligned} x_{\text{trial}} &= x + (2\xi_x - 1)\cdot\delta r \\ y_{\text{trial}} &= y + (2\xi_y - 1)\cdot\delta r \\ z_{\text{trial}} &= z + (2\xi_z - 1)\cdot\delta r \end{aligned}$$

δr – maximum allowed displacement

acceptance probability of the trial configuration is obtained from potential energy – current U, of trial config U_{trial}:

$$\mathcal{P} = \begin{cases} 1 & \text{if } U_{\text{trial}} < U \\ \exp\left[-\frac{U_{\text{trial}} - U}{k_{\text{B}}T}\right] & \text{otherwise} \end{cases}$$

Metropolis' method

Trial set of coordinates is calculated with random $\xi_{x,z,y} \in (0,1)$:

$$\begin{aligned} x_{\text{trial}} &= x + (2\xi_x - 1) \cdot \delta r \\ y_{\text{trial}} &= y + (2\xi_y - 1) \cdot \delta r \\ z_{\text{trial}} &= z + (2\xi_z - 1) \cdot \delta r \end{aligned}$$

δr – maximum allowed displacement

acceptance probability of the trial configuration is obtained from potential energy – current U, of trial config U_{trial}:

$$\mathcal{P} = \begin{cases} 1 & \text{if } U_{\text{trial}} < U \\ \exp\left[-\frac{U_{\text{trial}} - U}{k_{\text{B}} T}\right] & \text{otherwise} \end{cases}$$

The trial configuration is accepted if $\mathcal{P} >$ random $\zeta \in (0,1)$ otherwise it is discarded and another trial is generated

Acceptance ratio

The percentage of accepted configurations (among all generated) governed by max. allowed displacement δr – adjustable parameter

- usually chosen so that $\frac{1}{3}$ to $\frac{1}{2}$ of all configs are accepted
- this was shown to lead to the most efficient sampling

Acceptance ratio

The percentage of accepted configurations (among all generated) governed by max. allowed displacement δr – adjustable parameter

- usually chosen so that $\frac{1}{3}$ to $\frac{1}{2}$ of all configs are accepted
- this was shown to lead to the most efficient sampling

δr too small $\rightarrow$ most configurations are accepted though,
but the configurations are very similar $\rightarrow$ slow sampling
δr too large $\rightarrow$ too many trial configurations are rejected
Often – δr adjusted in the course of the simulation
in order to reach a certain target acceptance ratio

Properties of MC

- generates a correct thermodynamic ensemble (canonical)
- involves temperature naturally
  - no additional thermostat necessary
  - difference from MD
- no kinetic information (velocities, E_{kin})

MC protocol – variations

Possible modifications to the algorithm:

- move the atoms sequentially, in a preset order, instead of selecting one randomly – one fewer random number needed
- move several atoms at once, instead of a single atom – very efficient sampling of config space (with appropriate δr)

Generators of pseudorandom numbers

Several random numbers in every iteration have to be obtained
and a large number of iterations is needed
→ reliable and efficient source of random numbers needed.

Most convenient – ‘calculate’ random numbers in some way
paradoxical requirement (computers are deterministic)

Generators of pseudorandom numbers

Several random numbers in every iteration have to be obtained
and a large number of iterations is needed
→ reliable and efficient source of random numbers needed.

Most convenient – ‘calculate’ random numbers in some way
paradoxical requirement (computers are deterministic)

There are ways to generate sequences of pseudorandom numbers
not actually random, but still independent enough of each other,
with right statistical properties → useful for MC

Linear congruential generators

- most commonly used generators
- produce sequences of pseudorandom numbers
- a following number in the sequence ξ_{i+1} is obtained

Linear congruential generators

- most commonly used generators
- produce sequences of pseudorandom numbers
- a following number in the sequence ξ_{i+1} is obtained
  1. from the previous number ξ_i
  2. multiplying by a constant a
  3. adding another constant b
  4. and taking the remainder when dividing by a constant m

Linear congruential generators

- most commonly used generators
- produce sequences of pseudorandom numbers
- a following number in the sequence ξ_{i+1} is obtained
  1. from the previous number ξ_i
  2. multiplying by a constant a
  3. adding another constant b
  4. and taking the remainder when dividing by a constant m
- initial value (seed) has to be chosen (often – system time)

$$\begin{aligned} \xi_0 &= \text{seed} \\ \xi_{i+1} &= (a \cdot \xi_i + b) \bmod m \end{aligned}$$

Linear congruential generators

- most commonly used generators
- produce sequences of pseudorandom numbers
- a following number in the sequence ξ_{i+1} is obtained
  1. from the previous number ξ_i
  2. multiplying by a constant a
  3. adding another constant b
  4. and taking the remainder when dividing by a constant m
- initial value (seed) has to be chosen (often – system time)

$$\begin{aligned} \xi_0 &= \text{seed} \\ \xi_{i+1} &= (a \cdot \xi_i + b) \bmod m \end{aligned}$$

value $\in (0, 1)$ is obtained by dividing ξ_{i+1} by the modulus m

Linear congruential generators

Very important – choose appropriate values of a, b and m

Then, the generator will produce all possible values $0, \ldots, m-1$ and will start to repeat the sequence only after m numbers.

Otherwise – the sequence starts to repeat itself much earlier, and the randomness is severely limited.

Linear congruential generators

Very important – choose appropriate values of a, b and m
Then, the generator will produce all possible values $0, \dots, m-1$
and will start to repeat the sequence only after m numbers.
Otherwise – the sequence starts to repeat itself much earlier,
and the randomness is severely limited.

Disadvantage – if we generate points in an N-dimensional space,
these are not distributed uniformly in the space,
but rather they lie on at most $\sqrt[N]{m}$ $(N-1)$-dimensional planes
(i.e. on straight lines if we have a 2D space).
With really poor generators – much fewer than $\sqrt[N]{m}$ hyperplanes.

Linear congruential generators

Very important – choose appropriate values of a, b and m
Then, the generator will produce all possible values $0, \ldots, m-1$
and will start to repeat the sequence only after m numbers.
Otherwise – the sequence starts to repeat itself much earlier,
and the randomness is severely limited.

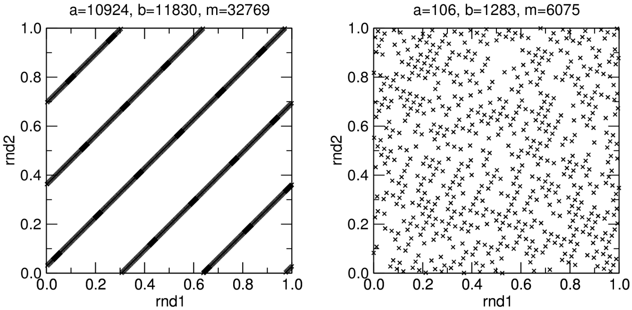
Disadvantage – if we generate points in an N-dimensional space,
these are not distributed uniformly in the space,
but rather they lie on at most $\sqrt[N]{m}$ $(N-1)$-dimensional planes
(i.e. on straight lines if we have a 2D space).
With really poor generators – much fewer than $\sqrt[N]{m}$ hyperplanes.

An example is RANDU: ξ_0 is odd and $\xi_{i+1} = 65539 \cdot \xi_i \bmod 2^{31}$.
All generated values are odd, the period is only 2^{29},

and the points $(\xi_i, \xi_{i+1}, \xi_{i+2})$ cumulate on as few as 15 planes in space.

Linear congruential generators

A bad and a good generator of pseudorandom numbers.



Each point (rnd1,rnd2) is a pair of consecutive numbers from LCG

Generators of higher quality

Still, LCG are often used in MC simulations
because of extreme simplicity and computational efficiency.

Higher-quality pseudorandom number generators:

linear feedback shift register generators

- uses several bits from current number to generate new ones
- does not cumulate the generated numbers on hyperplanes

Mersenne twister

- current state of the art among generators
- extremely long period of $2^{19937} - 1$
- no cumulation of numbers on hyperplanes up to 623 dim.
- even suitable for cryptographic applications

Alternative generators of random numbers

In Unix-like operating systems (with Linux being the first),
`/dev/random` (or `/dev/urandom`) is a special file
that serves as a random or pseudorandom number generator.
It accesses environmental noise collected from device drivers etc.

from Wikipedia

Monte Carlo simulation of molecules

Easiest implementation – system of monoatomic molecules
(translational degrees of freedom only)

Polyatomic molecules – more complex situation,
most difficult if there is large conformational flexibility

Monte Carlo simulation of molecules

Easiest implementation – system of monoatomic molecules
(translational degrees of freedom only)
Polyatomic molecules – more complex situation,
most difficult if there is large conformational flexibility
Then, the internal degrees of freedom have to be free to vary
→ overlap of atoms → energy grows steeply
→ extremely low acceptance ratio

Monte Carlo simulation of molecules

Easiest implementation – system of monoatomic molecules
(translational degrees of freedom only)

Polyatomic molecules – more complex situation,
most difficult if there is large conformational flexibility

Then, the internal degrees of freedom have to be free to vary
→ overlap of atoms → energy grows steeply
→ extremely low acceptance ratio

Rigid molecules – still quite easy to simulate with MC
– orientation in space being varied beside position in space
– rotation along an axis x, y or z by randomly chosen angle

Monte Carlo simulation of molecules

Particularly rich area of application of MC
– macromolecular chemistry

Monte Carlo simulation of polymers

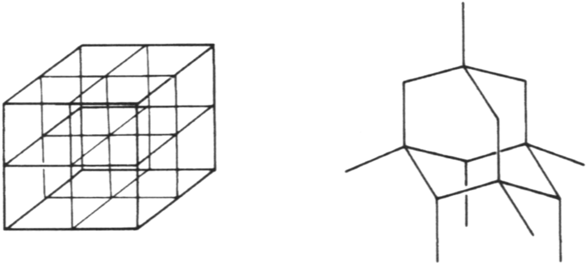
Many approximative polymer models are suitable for MC simulation

Convenient – a chain of monomer units,
which are elementary particles (no further internal structure)

Monte Carlo simulation of polymers

Many approximative polymer models are suitable for MC simulation
Convenient – a chain of monomer units,
which are elementary particles (no further internal structure)

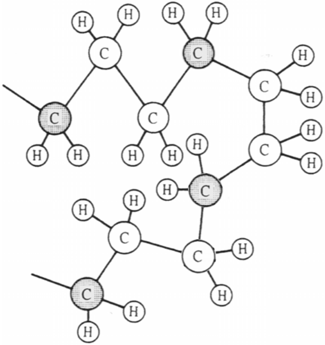
Lattice models – very simple and useful for efficient studies
– monomer units connected with a bond can occupy
neighboring lattice points in a cubic or tetrahedral lattice

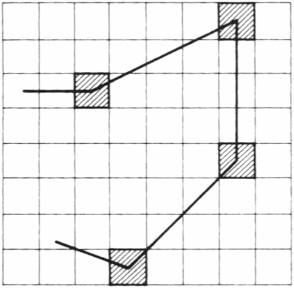


Monte Carlo simulation of polymers

Usually – very simple expressions for potential energy
(simplicity of the model + requirement of efficiency)

More realistic and complex – bond fluctuation model
- lattice is finer-grained compared to the bond length
- ‘effective’ bonds are not constrained to the edges of lattice





Monte Carlo simulation of polymers

Simplest type of simulation – random walk
- the chain grows in a random direction until the desired length
- first implementation – excluded volume of previous segments is not considered, so the chain is free to cross itself

Monte Carlo simulation of polymers

Simplest type of simulation – random walk

– the chain grows in a random direction until the desired length

– first implementation – excluded volume of previous segments is not considered, so the chain is free to cross itself

Various structural properties can be evaluated by averaging the results over many growing 'simulations'.

Quantities of interest: end-to-end distance R_n, radius of gyration s_n

$$\begin{aligned} \left\langle R_n^2 \right\rangle_0 &= n \cdot l^2 \\ \left\langle s^2 \right\rangle_0 &= \left\langle R_n^2 \right\rangle / 6 \end{aligned}$$

for a chain composed of n bonds with length l

Monte Carlo simulation of polymers

Excluded volume no described – may seem to be crude, but this may not be a problem

Monte Carlo simulation of polymers

Excluded volume no described – may seem to be crude,
but this may not be a problem

theta state (ϑ state) of a polymer

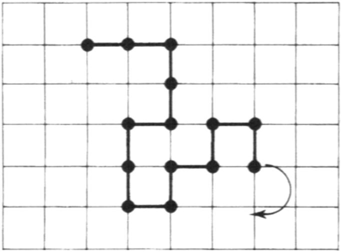
- excluded volume and attractive interactions cancel exactly
  (also, the second virial coefficient vanishes)
- results derived with simple random walk are actually valid
  (often designated with the subscript ‘0’)

Monte Carlo simulation of polymers

How to take the excluded volume into account?

– do not allowing the chain to extend to already occupied points

– self-avoiding walk



Monte Carlo simulation of polymers

How to take the excluded volume into account?
– do not allowing the chain to extend to already occupied points
– self-avoiding walk

SAW was used to generate all possible configurations
of a polymer of given length on a given lattice
→ partition function → all thermodynamic properties

'potential energy' – simple interaction model for nearby monomers
also – copolymers with two different types of monomer units

particular attention – structural properties – end-to-end distance:

$$\langle R_n^2 \rangle \approx n^{1.18} \cdot l^2 \qquad \text{for } n \to \infty$$

Monte Carlo simulation of polymers

to modify the configuration with MC – nearly impossible
because no free space (espec. for densely packed polymers)

Monte Carlo simulation of polymers

to modify the configuration with MC – nearly impossible
because no free space (espec. for densely packed polymers)

slithering snake model – a trick to avoid the problem:

1. choose one end of the polymer chain randomly as the head
2. make MC attempt to connect a new monomer unit to head
3. on success, then remove one monomer from the other end
4. repeat

Monte Carlo simulation of polymers

to modify the configuration with MC – nearly impossible
because no free space (espec. for densely packed polymers)

slithering snake model – a trick to avoid the problem:

1. choose one end of the polymer chain randomly as the head
2. make MC attempt to connect a new monomer unit to head
3. on success, then remove one monomer from the other end
4. repeat

How to improve the lattice models? . . . cancel the lattice

– 'continuous' polymer models – string of connected beads
– freely connected, with a spherically symmetric potential (LJ)
– do not generally correspond to monomer units
– links ('bonds') either fixed or with a harmonic potential

Monte Carlo simulation of polymers

most unrealistic property – continuous variation of link angles

Monte Carlo simulation of polymers

most unrealistic property – continuous variation of link angles

freely rotating chain model – link angles are fixed

- – free rotation about the links (‘dihedral angles’) allowed
- – this changes the overall structure of the polymer chain!

Monte Carlo simulation of polymers

most unrealistic property – continuous variation of link angles

freely rotating chain model – link angles are fixed

- free rotation about the links ('dihedral angles') allowed
- this changes the overall structure of the polymer chain!
- the extension of a chain with interlink angle φ,
  with respect to the freely connected chain:

$$C_n = \frac{\langle R_n^2 \rangle}{n \cdot l^2} \qquad \rightarrow \qquad C_\infty = \frac{1 - \cos\varphi}{1 + \cos\varphi}$$

so, for a tetrahedral bond angle $\varphi = 109^\circ$: $C_\infty \approx 2$

Monte Carlo simulation of polymers

rotational isomeric state model (Flory, 1969)

- every link is always in one of defined rotational states
- these states, dihedral angles, are minima of pot. energy
- e.g. trans, gauche(+) and gauche(−) in polyalkane chain

Monte Carlo simulation of polymers

rotational isomeric state model (Flory, 1969)

- every link is always in one of defined rotational states
- these states, dihedral angles, are minima of pot. energy
- e.g. trans, gauche(+) and gauche(–) in polyalkane chain
- conformation-dependent properties described with matrices
- best known approximative description of polymer chains
- combined with MC → wide range of properties
    - conformations of chain are generated with probability distributions corresponding to their statistical weights, which are a component of the model (in a matrix form)

Monte Carlo simulation of polymers

rotational isomeric state model

matrix of statistical weights for an example of polyalkane chain:

$$U \equiv \begin{pmatrix} u_{tt} & u_{tg^+} & u_{tg^-} \\ u_{g^+t} & u_{g^+g^+} & u_{g^+g^-} \\ u_{g^-t} & u_{g^-g^+} & u_{g^-g^-} \end{pmatrix} = \begin{pmatrix} 1.00 & 0.54 & 0.54 \\ 1.00 & 0.54 & 0.05 \\ 1.00 & 0.05 & 0.54 \end{pmatrix}$$

u_{ab} – statistical weight of dihedral state b
following a link in the dihedral state a

Monte Carlo simulation of polymers

rotational isomeric state model

Starting on one end of the chain, a conformation is generated
by calculating the dihedral angles sequentially,
until the whole chain is done.
The probability of each dihedral angle is determined with MC
using the a priori probabilities of the dihedral states
and the state of the previous dihedral angle.

Monte Carlo simulation of polymers

rotational isomeric state model

Starting on one end of the chain, a conformation is generated
by calculating the dihedral angles sequentially,
until the whole chain is done.
The probability of each dihedral angle is determined with MC
using the a priori probabilities of the dihedral states
and the state of the previous dihedral angle.

In a typical study, a large number of such chain will be grown,
and the properties of interest will be calculated and averaged.
Estimated can be pair correlation functions, scattering functions
and force–elongation profiles.