GPU-powered Molecular Dynamics Simulations in Statistical Physics

Felix Höfling

Department of Mathematics and Computer Science

Freie Universität Berlin, Germany

Perspectives of GPU Computing in Science

Rome, 26–28 September 2016

Freie Universität Berlin

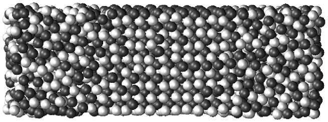
Why molecular dynamics simulations?

- clean “experiments”, detailed view at nanoscale processes

Why molecular dynamics simulations?

- clean “experiments”, detailed view at nanoscale processes
- make predictions (qualitative and quantitative)
    - materials research: surface tension, stress–strain relations
    - drug development: protein conformations, reaction pathways

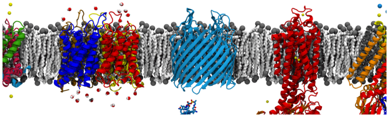
materials research



Kerrache, Horbach & Binder, EPL (Europhys. Lett.) 2008

liquid/crystal interfaces of $Al_{50}Ni_{50}$ alloy (EAM potentials)

cell biology



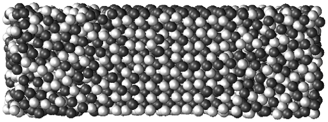
http://simbac.gatech.edu

membrane proteins embedded in a phospholipid bilayer

Why molecular dynamics simulations?

- clean "experiments", detailed view at nanoscale processes
- make predictions (qualitative and quantitative)
  - materials research: surface tension, stress–strain relations
  - drug development: protein conformations, reaction pathways
- test microscopic theories, e.g., in statistical physics
  rheology of polymer composites, glass transition dynamics, nucleation theory
- requires good models, coarse-grained descriptions

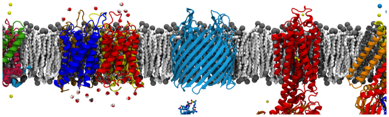
materials research



Kerrache, Horbach & Binder, EPL (Europhys. Lett.) 2008

liquid/crystal interfaces of $Al_{50}Ni_{50}$ alloy
(EAM potentials)

cell biology



http://simbac.gatech.edu

membrane proteins embedded
in a phospholipid bilayer

Molecular dynamics simulations

- integration of Newton's equations of motion for N particles, N large

$$m_i \dot{\boldsymbol{r}}_i = \boldsymbol{p}_i \,, \qquad \dot{\boldsymbol{p}}_i = \underbrace{\boldsymbol{F}_i(\{\boldsymbol{r}_j\})\,,}_{\text{interaction}} \qquad i = 1, \dots, N$$

- conservation laws:
  - total momentum, total energy $H = \sum_i \boldsymbol{p}_i^2/2m_i + V(\{\boldsymbol{r}_j\}), \dots$
  - phase space volume (symplectic flows)

Molecular dynamics simulations

- integration of Newton's equations of motion for N particles, N large

$$m_i \dot{\boldsymbol{r}}_i = \boldsymbol{p}_i\,, \qquad \dot{\boldsymbol{p}}_i = \underbrace{\boldsymbol{F}_i(\{\boldsymbol{r}_j\})\,,}_{\text{interaction}} \qquad i = 1, \dots, N$$

- conservation laws:
  - total momentum, total energy $H = \sum_i \boldsymbol{p}_i^2/2m_i + V(\{\boldsymbol{r}_j\}), \dots$
  - phase space volume (symplectic flows)
- exact solution: $\{\boldsymbol{r}_i, \boldsymbol{p}_i\} \mapsto \mathrm{e}^{\mathrm{i}\mathcal{L}\tau}\{\boldsymbol{r}_i, \boldsymbol{p}_i\}, \qquad \mathcal{L} = \mathcal{L}_{\boldsymbol{r}} + \mathcal{L}_{\boldsymbol{p}}$

$$\mathrm{i}\mathcal{L}_{\boldsymbol{r}} = \sum_i (\boldsymbol{p}_i/m_i) \cdot \partial/\partial \boldsymbol{r}_i\,, \qquad \mathrm{i}\mathcal{L}_{\boldsymbol{p}} = -\sum_i \boldsymbol{F}_i(\{\boldsymbol{r}_j\}) \cdot \partial/\partial \boldsymbol{p}_i$$

→ velocity-Verlet algorithm: $\mathrm{e}^{\mathrm{i}\mathcal{L}\tau} = \mathrm{e}^{\mathrm{i}\mathcal{L}_{\boldsymbol{p}}\tau/2}\,\mathrm{e}^{\mathrm{i}\mathcal{L}_{\boldsymbol{r}}\tau}\,\mathrm{e}^{\mathrm{i}\mathcal{L}_{\boldsymbol{p}}\tau/2} + O\left(\tau^2\right)$

Molecular dynamics simulations

- integration of Newton's equations of motion for N particles, N large

$$m_i \dot{\boldsymbol{r}}_i = \boldsymbol{p}_i\,, \qquad \dot{\boldsymbol{p}}_i = \boldsymbol{F}_i(\{\boldsymbol{r}_j\})\,, \qquad i = 1, \dots, N$$

interaction

- conservation laws:
  - total momentum, total energy $H = \sum_i \boldsymbol{p}_i^2/2m_i + V(\{\boldsymbol{r}_j\}), \dots$
  - phase space volume (symplectic flows)
- exact solution: $\{\boldsymbol{r}_i, \boldsymbol{p}_i\} \mapsto \mathrm{e}^{\mathrm{i}\mathcal{L}\tau}\{\boldsymbol{r}_i, \boldsymbol{p}_i\}, \qquad \mathcal{L} = \mathcal{L}_{\boldsymbol{r}} + \mathcal{L}_{\boldsymbol{p}}$

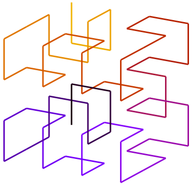
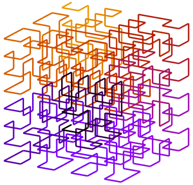
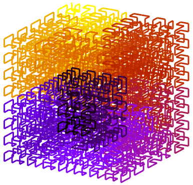
$$\mathrm{i}\mathcal{L}_{\boldsymbol{r}} = \sum_i (\boldsymbol{p}_i/m_i) \cdot \partial/\partial \boldsymbol{r}_i\,, \qquad \mathrm{i}\mathcal{L}_{\boldsymbol{p}} = -\sum_i \boldsymbol{F}_i(\{\boldsymbol{r}_j\}) \cdot \partial/\partial \boldsymbol{p}_i$$

→ velocity-Verlet algorithm: $\mathrm{e}^{\mathrm{i}\mathcal{L}\tau} = \mathrm{e}^{\mathrm{i}\mathcal{L}_{\boldsymbol{p}}\tau/2}\,\mathrm{e}^{\mathrm{i}\mathcal{L}_{\boldsymbol{r}}\tau}\,\mathrm{e}^{\mathrm{i}\mathcal{L}_{\boldsymbol{p}}\tau/2} + O(\tau^2)$

- parallelisation is trivial once the forces $\{\boldsymbol{F}_i\}$ are known:
  - thread $\#i$ → particle $\#i$ linear memory access, complexity $O(N)$
- interactions require communication naïvely: $O(N^2)$
  - Verlet neighbour lists for short-ranged pair forces → $M \cdot O(N)$
  - algorithmic primitives: radix sort and reduction → $O(N \log(N))$

Data locality: Hilbert's space-filling curve

- positions of neighbours have random locations in memory
  - fetch coordinates via read-only texture cache
  - limited cache size → memory locality

→ periodically re-order particle data in memory

J. A. Anderson *et al.*, J. Comp. Phys. **227** 5342 (2008)

- Hilbert's space-filling curve maps 3D space to 1D memory
  - Hilbert curve is recursively generated *on the GPU*
  - generate permutation using radix sort
  - rearrange particle data using texture reads and coalescable writes

Conservation laws: floating-point precision

- theory: conservation of total momentum and total energy
- reality: drift due to accumulation of round-off errors

P. H. Colberg and F. Höfling, Comput. Phys. Commun. **182**, 1120 (2011)

Conservation laws: floating-point precision

- theory: conservation of total momentum and total energy
- reality: drift due to accumulation of round-off errors
- solution: use multi-precision floating-point arithmetics (double-single)
    - for the summation of forces (group opposite cells!)
    - in the velocity-Verlet algorithm
- but evaluate potentials in single precision
  saves 50% read/write access to global GPU memory
    - smooth truncation of potentials (C^2-continuous at the cutoff)

P. H. Colberg and F. Höfling, Comput. Phys. Commun. **182**, 1120 (2011)

Conservation laws: floating-point precision

- theory: conservation of total momentum and total energy
- reality: drift due to accumulation of round-off errors
- solution: use multi-precision floating-point arithmetics (double-single)
  - for the summation of forces (group opposite cells!)
  - in the velocity-Verlet algorithm
- but evaluate potentials in single precision
  saves 50% read/write access to global GPU memory
  - smooth truncation of potentials (C^2-continuous at the cutoff)

➔ energy and momentum drifts essentially eliminated
momentum: less than 10^{-7} per 10^7 steps of $\delta t^* = 0.001$
energy: less than 10^{-5} per 10^8 steps for $\delta t^* = 0.001$, $h = 0.005$

- execution times increase by less than 10% (Nvidia Tesla K20X)

P. H. Colberg and F. Höfling, Comput. Phys. Commun. **182**, 1120 (2011)

Double-single floating-point precision

- poor double precision performance on Nvidia GTX and Maxwell
  no support for double precision prior to compute capability 1.3

→ double-single arithmetic based on native single precision instructions

  - DSFUN90 package provides double-single routines in Fortran

    D. H. Bailey (2005)

  - porting to CUDA straightforward, use C++ operator overloads
  - effective precision of 44 bits

- `dsfloat=(hi,lo)` $\leftrightarrow$ `float2` (lossless)

- wishlist: `double` $\leftrightarrow$ `float2=(float, rest)`

- store high and low floats in two different arrays
  → efficient access to positions in single precision (force computation)

Velocity-Verlet in double-single precision (CUDA & C++ templates)

```
template <int dimension, typename gpu_vector_type>
__global__ void integrate(
    float4* g_position, gpu_vector_type* g_image, float4* g_velocity, gpu_vector_type const* g_force
  , float timestep, fixed_vector<float, dimension> box_length
)
{
    unsigned int const thread = GTID;               // global thread ID
    unsigned int const nthread = GTDIM;             // total number of threads

     // load double-single precision values from global memory (2 float4)
     fixed_vector<dsfloat, dimension> r, v;
     unsigned int species; float mass;
     tie(r, species) <<= tie(g_position[thread], g_position[thread + nthread]);
     tie(v, mass)    <<= tie(g_velocity[thread], g_velocity[thread + nthread]);

    // load single precision values from global memory (1 float4)
    fixed_vector<float, dimension> f = g_force[thread];

     // actual computations with 2D/3D vectors in double-single precision
     v += f * (timestep / 2) / mass;
     r += v * timestep;
     fixed_vector<float, dimension> image = box_kernel::reduce_periodic(r, box_length);

    // write results to global memory
    tie(g_position[thread], g_position[thread + nthread]) <<= tie(r, species);
    tie(g_velocity[thread], g_velocity[thread + nthread]) <<= tie(v, mass);
    if (!(image == float_vector_type(0))) {
        g_image[thread] = image + static_cast<float_vector_type>(g_image[thread]);
    }
}
```

Break-down of the algorithm

Binary mixture of 256,000 Lennard-Jones particles

HALMD ● HAL’s MD package
Highly Accelerated Large-scale Molecular Dynamics

task	time [ms]	#calls	share	complexity
MD integration step	4.1	10,000	99%	
dump system state (GPU $\rightarrow$ disk)	410	1	1%	
compute short-ranged forces	2.9	10,000	68%	$M \cdot O(N)$
velocity-Verlet integration	.41	10,000	10%	$O(N)$

(Nvidia Tesla K20Xm, $r_c = 2.5\sigma, r_{\text{skin}} = 0.5\sigma, \delta t^* = 0.001, \rho^* = 1.2$)

Break-down of the algorithm

Binary mixture of 256,000 Lennard-Jones particles

HALMD ● HAL's MD package
Highly Accelerated Large-scale Molecular Dynamics

- use generic algorithms for complex tasks, express them in terms of
  - radix sort and reduction operations $O(N \log(N))$
  - simple operations $O(N)$

task	time [ms]	#calls	share	complexity
MD integration step	4.1	10,000	99%	
dump system state (GPU $\rightarrow$ disk)	410	1	1%	
compute short-ranged forces	2.9	10,000	68%	$M \cdot O(N)$
velocity-Verlet integration	.41	10,000	10%	$O(N)$
generate neighbour lists	21	163	8%	$O(N \log(N))$
re-order particle data	15	163	6%	$O(N \log(N))$
generate cell lists	6.1	163	2%	$O(N \log(N))$
neighbour list criterion	.08	10,000	2%	$O(N \log(N))$

(Nvidia Tesla K20Xm, $r_c = 2.5\sigma, r_{\text{skin}} = 0.5\sigma, \delta t^* = 0.001, \rho^* = 1.2$)

What to do with the vast amount of data generated?

- avoid disk I/O → data locality!
- exploit parallel computing also for the data analysis

→ online evaluation of relevant quantities
thermodynamic variables, spatial profiles, time correlation functions, coarse-grained variables, . . .

- decide *before* the simulation what is relevant
the information needed is determined by the questions asked

What to do with the vast amount of data generated?

- avoid disk I/O → data locality!
- exploit parallel computing also for the data analysis

→ online evaluation of relevant quantities
thermodynamic variables, spatial profiles, time correlation functions, coarse-grained variables, . . .

- decide *before* the simulation what is relevant
the information needed is determined by the questions asked

H5MD!

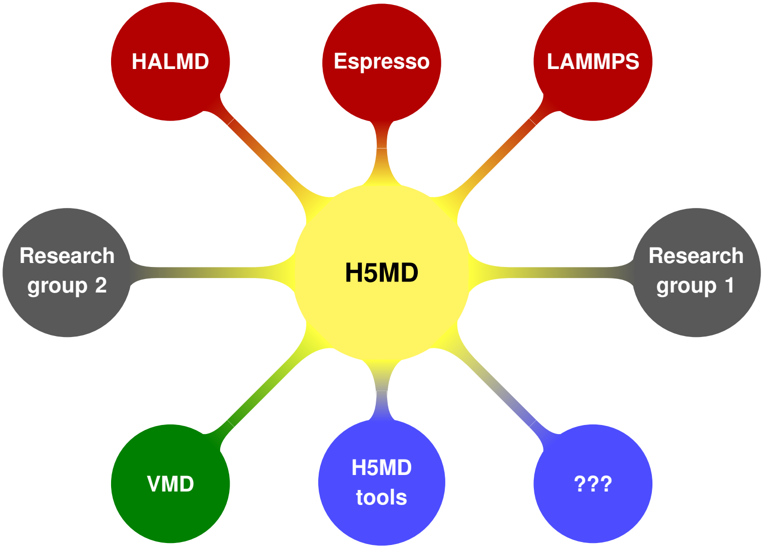
- HDF5 for Molecular Data: `http://nongnu.org/h5md`

P. de Buyl, P. H. Colberg, and F. Höfling, Comput. Phys. Commun. **185**, 1546 (2014)

- efficient, structured, and portable storage of heterogeneous data
- binary format, data compression, fast and parallel I/O
- based on the HDF5 library
bindings for C, C++, Fortran, Python; support by Matlab, Mathematica, . . .

The H5MD universe

simulation software visualisation software analysis tools people



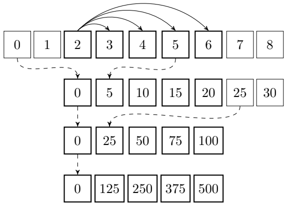
Evaluation of time correlation functions

- slow complex dynamics extends over many decades in time
- evaluate time correlation functions *in situ* on the GPU

$$C_{AB}(t) = \langle A(t)^* B(0) \rangle = \lim_{T\to\infty} \frac{1}{T} \int_0^T A(t+\tau)^* B(\tau)\, \mathrm{d}\tau$$

→ multiple-τ correlator ("blocking scheme"):

- time interval Δt increases geometrically with block level → logarithmic time grid
- calculate correlations only within the same block → modest memory usage



$$C_{AB}(m\Delta t) \approx \frac{1}{M'} \sum_{j=0}^{M'} A^*_{m+j} B_j$$

$$M' = T/\Delta t - m, \; A_k = A(k\Delta t), \; \ldots$$

- compute particle averages using parallel reduction algorithms

HALMD ● HAL's MD package

Highly Accelerated Large-scale Molecular Dynamics

- **acceleration:** specifically designed to run on GPU processors
  1 GPU comparable to **3–4 Ivy Bridge nodes** à 20 cores
  [HALMD @ "Kepler" K20Xm *vs.* LAMMPS @ supercomputer *Hydra* of the Max Planck Society]
- **applications:** the statistical physics of inhomogeneous fluids
  glass transition, liquid–vapour interfaces, demixing of binary fluids,
  confined fluids, porous media, 2D/3D systems, . . .
- **precision:** excellent numerical long-time stability
  conservation laws → double-single floating-point arithmetics, C^2-smooth potentials
- **efficient:** online evaluation of dynamic correlations minimises disk I/O
- output as structured, compressed, and portable H5MD files

P. de Buyl, P. H. Colberg, and F. Höfling, Comput. Phys. Commun. **185**, 1546 (2014)

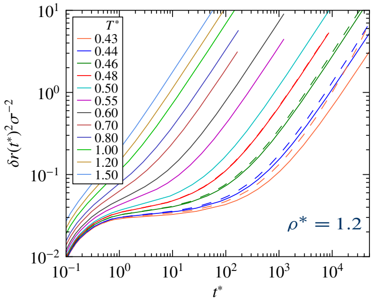
- **modular** & generic design → user scripts define complex tasks
- **educational** tool for M.Sc. students, lab courses → LGPL license

http://halmd.org

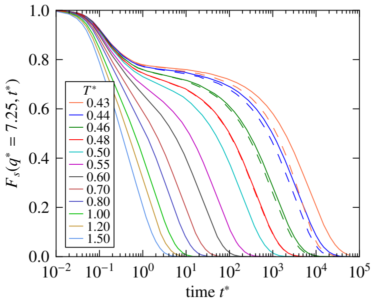
P. H. Colberg and F. Höfling, Comput. Phys. Commun. **182**, 1120 (2011)

Glassy dynamics of a supercooled liquid

- binary mixture of 50,000 LJ spheres
  “Kob–Andersen”, $r_c = 2.5\sigma$, $\rho^* = 1.2$
  production runs of 10^7 NVE steps finished within 8 hours on a Tesla T10 GPU
- slow dynamics upon cooling (or compression)
- single precision: quantitatively and qualitatively wrong results



mean-square displacement

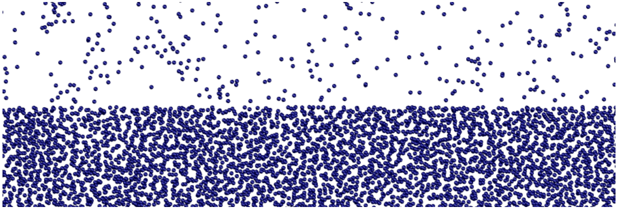


self-intermediate scattering function

Colberg & Höfling, Comput. Phys. Commun. (2011)

Application: structure of liquid–vapour interfaces

- planar interface between coexisting liquid and vapour phases

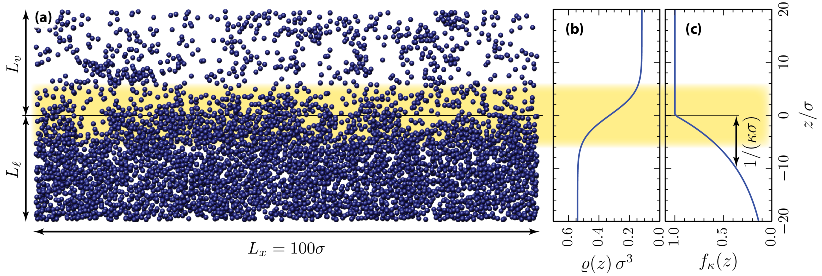


Lennard-Jones fluid: cross section of 3D simulation at $T^* = 1.0 \approx 0.82 T_c^*$

Application: structure of liquid–vapour interfaces

- planar interface between coexisting liquid and vapour phases
- broadened by thermal fluctuations → mean profile $\rho(z)$

→ determined by temperature T and interparticle attraction
coexisting densities ρ_v, ρ_ℓ and interface width ζ



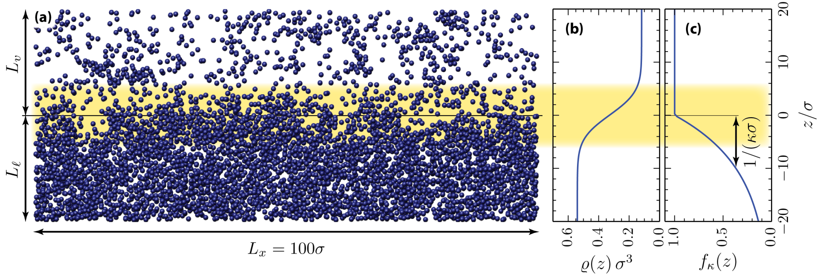
Lennard-Jones fluid: cross section of 3D simulation at $T^* = 1.15 \approx 0.94 T_c^*$

Application: structure of liquid–vapour interfaces

- planar interface between coexisting liquid and vapour phases
- broadened by thermal fluctuations → mean profile $\rho(z)$

→ determined by temperature T and interparticle attraction
coexisting densities ρ_v, ρ_ℓ and interface width ζ

- "surface roughness" → bulk-like fluctuations & capillary waves
- CWs are controlled by mesoscopic surface tension $\gamma(q)$



Lennard-Jones fluid: cross section of 3D simulation at $T^* = 1.15 \approx 0.94 T_c^*$

Grazing-incidence small-angle X-ray scattering (GISAXS)

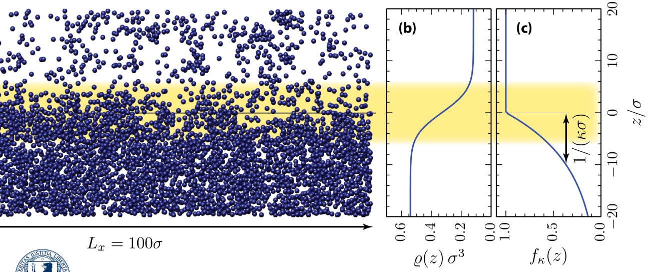
- liquid side: evanescent wave with penetration depth $1/\kappa(\alpha_i, \alpha_f)$
- scattering cross section proportional to [Dietrich & Haase, Phys. Rep. (1995)]

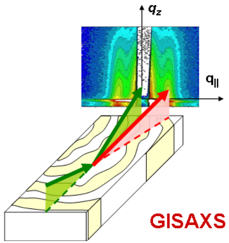
$$I_{\text{tot}}(\boldsymbol{q};\kappa) = \iint_{-L_\ell}^{L_v} \mathrm{d}z\,\mathrm{d}z'\; f_\kappa(z)^*\; f_\kappa(z')\; \boxed{G(\boldsymbol{q}, z, z')}$$

$$\int \mathrm{d}^2 R\, \mathrm{e}^{-\mathrm{i}\boldsymbol{q}\cdot\boldsymbol{R}} \left[\langle\hat\rho(\mathbf{0}, z)\hat\rho(\boldsymbol{R}, z')\rangle - \rho(z)\rho(z')\right]$$

- weighting factor $f_\kappa(z) = \mathrm{e}^{-\kappa|z|}$ for $z < 0$
- sufficiently deep penetration required $(1/\kappa \gg \zeta)$

  but: sizeable background from bulk on top of interface signal

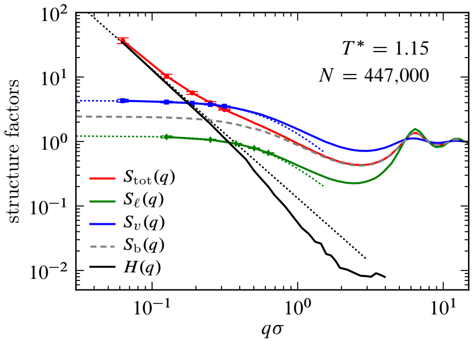


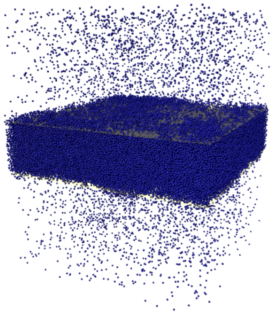


Simulation results: interface structure factor

- synthetic scattering data $S_{\text{tot}}(q)$, similar to GISAXS $\sim 1/\gamma_0 q^2$
- determine properties of liquid and vapour bulk separately $\rightarrow S_b(q)$

$\rightarrow$ interface structure factor $\widetilde{H}(q) = S_{\text{tot}}(q) - S_{\text{b}}(q)$





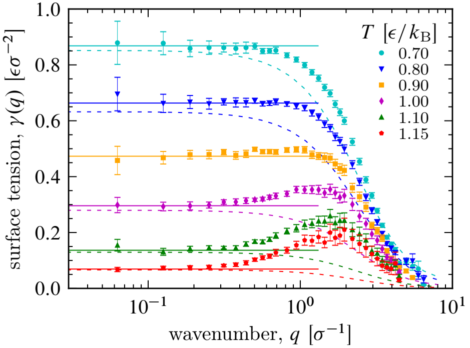
447,000 Lennard-Jones particles, cutoff $r_c = 3.5\sigma$, $T^* = 1.15 \approx 0.94T_c^*$
box size $L_x = L_y = 100\sigma$, $L_z = 200\sigma$, liquid slab $w = 50\sigma$
30 runs over 10^7 steps for $S_{\text{tot}}(q) \rightarrow$ 840 GPU hours (Tesla K20Xm)

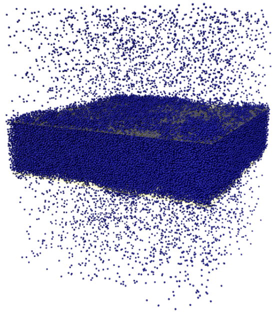
Höfling and Dietrich, EPL (Europhys. Lett.) **109**, 46002 (2015)

Simulation results: interface structure factor

- synthetic scattering data $S_{\text{tot}}(q)$, similar to GISAXS $\sim 1/\gamma_0 q^2$
- determine properties of liquid and vapour bulk separately $\rightarrow S_b(q)$

$\rightarrow$ interface structure factor $\widetilde{H}(q) = S_{\text{tot}}(q) - S_{\text{b}}(q) \sim 1/\gamma(q)\, q^2$

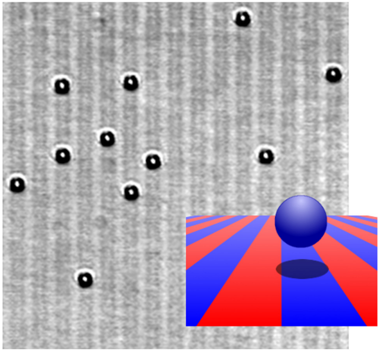




447,000 Lennard-Jones particles, cutoff $r_c = 3.5\sigma$, $T^* = 1.15 \approx 0.94 T_c^*$
box size $L_x = L_y = 100\sigma$, $L_z = 200\sigma$, liquid slab $w = 50\sigma$
30 runs over 10^7 steps for $S_{\text{tot}}(q) \rightarrow$ 840 GPU hours (Tesla K20Xm)

Höfling and Dietrich, EPL (Europhys. Lett.) **109**, 46002 (2015)

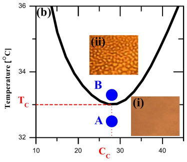
Colloids in a critical water–oil solvent

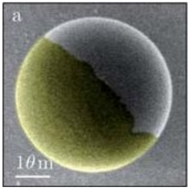


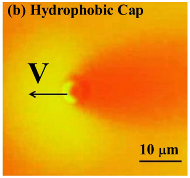
Courtesy of M. Tröndle

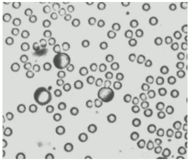
Tröndle, Bechinger, Dietrich *et al.*, Mol. Phys. **109**, 1169 (2011)

critical Casimir effect





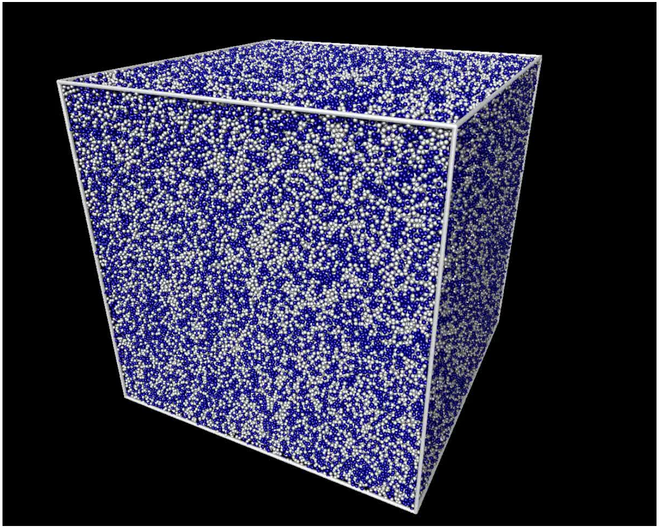


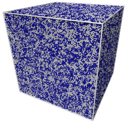


Bechinger *et al.*, Soft Matter **7**, 8810 (2011); J. Phys.: Condens. Matter **24**, 284129 (2012)

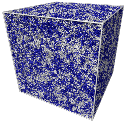
self-propelled Janus particle

Spinodal decomposition of a binary mixture

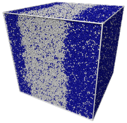




$T > T_c$



$T = T_c$



$T < T_c$

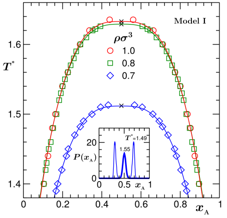
700k particles, 10^8 NVE steps, temperature quench to $T = .93T_c$ (5 days on a GTX980)

Continuous demixing transition of a binary liquid

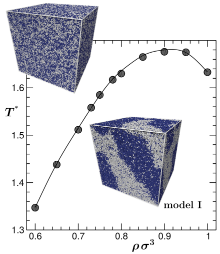
S. Roy, S. Dietrich & F.H., J. Chem. Phys. **145**, 134505 (2016)

- critical phenomenon: universality class of model H'
  - conserved scalar order parameter: local composition fluctuation
  - couples to local number density and transverse momentum
- fluctuations of the local composition:
  divergence of both correlation length and relaxation time
  $\xi \sim |T - T_c|^{-\nu}$, $\nu \approx 0.630$ and $\tau_R \sim \xi^z$, $z \approx 3.06$

→ critical slowing down, interdiffusion ceases at T_c



coexistence curves
(ρ = const)



λ-line $T_c(\rho)$
(x_A = const)

Structure: Ising universality class

- static structure factor $S_{cc}(|\boldsymbol{k}|) = N^{-1}\langle \delta c_{\boldsymbol{k}}^{*}\, \delta c_{\boldsymbol{k}} \rangle$

→ correlation length ξ and "susceptibility" χ

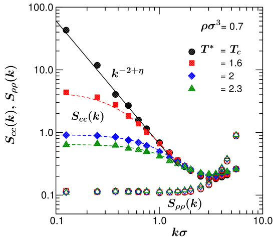
from extended Ornstein–Zernike form: $S_{cc}(k \ll \sigma) \simeq \rho k_{\mathrm{B}} T \chi / [1 + (k\xi)^2]^{1-\eta/2}$

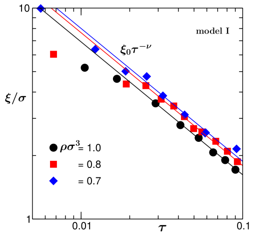
- critical scaling of Ising universality class

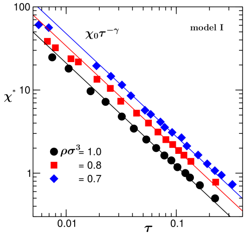
$$S_{cc}(k) = k^{-2+\eta} s(k\xi), \quad \xi \simeq \xi_0 \tau^{-\nu}, \; \chi \simeq \chi_0 \tau^{-\gamma}$$

$$d = 3: \quad \nu \approx 0.630, \quad \eta \approx 0.036, \quad \gamma = \nu(2-\eta)$$

- amplitudes ξ_0, χ_0 depend on the specific system (e.g., fluid density ρ)

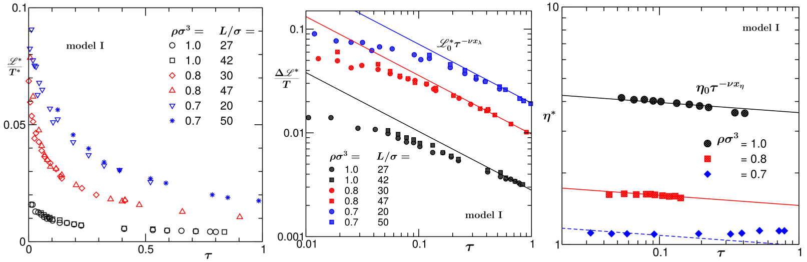






Critical singularities of transport coefficients (model H')

- interdiffusion vanishes: $D_{\mathrm{AB}} \sim \xi^{-x_D}$ $\quad x_D \approx 1.068$
  - use Onsager coefficient $\mathscr{L} = \chi D_{\mathrm{AB}}$ instead
  - critical enhancement: $\Delta\mathscr{L}(T) = \mathscr{L}(T) - \mathscr{L}_b k_{\mathrm{B}} T \simeq \mathscr{L}_0 k_{\mathrm{B}} T \tau^{-\nu x_\lambda}$ $\quad \nu x_\lambda \approx 0.567$
- shear viscosity diverges slowly: $\bar{\eta} \simeq \eta_0 \tau^{-\nu x_\eta}$ $\quad \nu x_\eta \approx 0.043$
- transport coefficients are calculated from time correlation functions



87,500 particles ($L = 50\sigma$), 20 runs over 10^7 steps → 6 h on 20 Tesla K20Xm for each T, ρ

Universal ratios of critical amplitudes

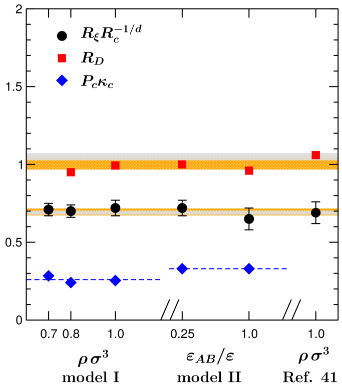
- some ratios of (non-universal) critical amplitudes are universal
  example: correlation length $\xi \simeq \xi_0^{\pm}|\tau|^{-\nu} \rightarrow \xi_0^+/\xi_0^- \approx 2.02$
- "static" amplitude ratio:
  involves only structural properties

$$R_\xi^+ R_c^{-1/d} = \xi_0^+ \left(\frac{\varphi_0^2}{k_\mathrm{B} T \chi_0^+}\right)^{1/d} \approx 0.69$$

- "dynamic" amplitude ratio:
  involves also transport coefficients

$$R_D = \frac{6\pi \eta_0^+ \xi_0^+ \mathcal{L}_0^+}{\chi_0^+} \approx 1.0$$

- also: $P_c \kappa_c \approx$ const (?!)



Acknowledgements

Siegfried Dietrich, Sutapa Roy

MPI für Intelligente System, Stuttgart, Germany

Jürgen Horbach, Nicolas Höft

Universität Düsseldorf, Germany

Peter Colberg

University of Toronto, Canada

Pierre de Buyl

Katholieke Universiteit Leuven, Belgium

supercomputer *Hydra*

Max Planck Computing and Data Facility Garching, Germany
700 Kepler GPUs (K20Xm), recent upgrade: 200 Maxwell GPUs (GTX980)

Summary

- **HAL's MD package:** specifically designed for use with GPUs
- floating-point precision crucial for numerical long-time stability

  Colberg & Höfling, Comput. Phys. Commun. (2011)
- online evaluation of time correlation functions avoids disk I/O
- **H5MD:** HDF5 for Molecular Data `http://nongnu.org/h5md`

  de Buyl, Colberg & Höfling, Comput. Phys. Commun. (2014)
- applications: inhomogeneous fluids at the molecular scale
  glass transition, liquid–vapour interfaces, demixing of binary fluids,
  confined fluids, porous media, 2D/3D systems, . . .

  Höfling & Dietrich, EPL (2015)
  Roy, Dietrich & Höfling, J. Chem. Phys. (2016)

→ scope for multi-GPU simulations (in the near future)

Come to Berlin for a free tutorial on HAL's MD package!

HALMD ● HAL's MD package
Highly Accelerated Large-scale Molecular Dynamics

`http://halmd.org`