Meso– and multi–scale modelling E-CAM modules II

E-CAM Deliverable 4.3
Deliverable Type: Other
Delivered in March, 2018

E-CAM
The European Centre of Excellence for
Software, Training and Consultancy
in Simulation and Modelling

Funded by the European Union under grant agreement 676531

Project and Deliverable Information

Project Title	E-CAM: An e-infrastructure for software, training and discussion in simulation and modelling
Project Ref.	Grant Agreement 676531
Project Website	https://www.e-cam2020.eu
EC Project Officer	Juan PELEGRIN
Deliverable ID	D4.3
Deliverable Nature	Other
Dissemination Level	Public
Contractual Date of Delivery	Project Month 28(January, 2018)
Actual Delivery Date	30th March, 2018
Deliverable Description	9 software modules delivered to the E-CAM repository in the area of meso– and multi–scale modelling, based on user requests and their documentation.

Document Control Information

Document	Title:	Meso– and multi–scale modelling E-CAM modules II
	ID:	D4.3
	Version:	As of 30th March, 2018
	Status:	Accepted by WP leader
	Available at:	https://www.e-cam2020.eu/deliverables
	Document history:	Internal Project Management Link
Review	Review Status:	Reviewed
Authorship	Written by:	Burkhard Dünweg (DE-SMSM, Max Planck Institute for Polymer Research)
	Contributors:	Jony Castagna and Silvia Chiacchiera (Science and Technology Facilities Council); Hideki Kobayashi (Max Planck Institute for Polymer Research); Christian Krekeler (Free University of Berlin)
	Reviewed by:	Ana Catarina Mendonça (EPFL)
	Approved by:	Burkhard Dünweg (Max Planck Institute for Polymer Research)

Document Keywords

Keywords:	E-CAM, CECAM, Module, DL_MESO_DPD, ESPResSo++, GC-AdResS, GPU, DPD, dipole moments, polymer melts, equilibration, coarse–graining

30th March, 2018
***Disclaimer**:This deliverable has been prepared by the responsible Work Package of the Project in accordance with the Consortium Agreement and the Grant Agreement. It solely reflects the opinion of the parties to such agreements on a collective basis in the context of the Project and to the extent foreseen in such agreements.*

[1] duenweg@mpip-mainz.mpg.de

Contents

List of Figures

Executive Summary

In this report for Deliverable 4.3 of E-CAM, nine software modules in meso– and multi–scale modelling are presented. Two of the modules have been implemented in DL_MESO_DPD:

- Multi GPU version of the DL_MESO_DPD code
- Using SIONlib (parallel I/O library) to write/read HISTORY files in DL_MESO_DPD.

Furthermore, there are five modules that have been implemented in ESPResSo++. Together they form an implementation of a novel hierarchical strategy to generate equilibrium structures of polymer melts:

- Coarse-Graining
- Fine-Graining
- Reinsertion
- Feedback Control Mechanism
- Constrain-gyration radius.

Finally, two modules applying the grand canonical adaptive resolution scheme (GC-AdResS) have been developed, implemented and tested in/with GROMACS 5.1.0. The GC-AdResS scheme is implemented in several MD packages, with GROMACS 5.1.0 as an example for a very complex one. The modules provide a recipe to simplify the implementation and to make it more general and easier to implement into other codes.

- Abrupt-GC-AdResS
- Abrupt-AdResS_forcecap

A manual explaining the use of GC-AdResS implemented in GROMACS is also presented here, as supporting information.

A short description is written for each module, followed by a link to the respective Merge-Request on the GitLab service of E-CAM. These merge requests contain detailed information about the code development, testing and documentation of the modules.

1 Introduction

Work package 4 deals with "mesoscale" simulations that are characterized by a description of the physics "somewhere in between" atomistic Molecular Dynamics and engineering–style continuum mechanics. Many mesoscale methods have been developed, and this is still a highly active field. Of particular importance is the development of systematic "coarse–graining" approaches, where the same physics is being described on two or more differing length and time scales, and relations are established between these in a quantitative fashion.

The software modules of the present report are all integrated into big software packages and thus extend their applicability. The first package is DL_MESO_DPD [1, 2] that provides a toolbox to do DPD simulations. DPD ("dissipative particle dynamics") is essentially Molecular Dynamics, however not for atoms but for effective particles that represent many microscopic degrees of freedom. These effective particles interact via soft potentials, and pairwise friction augmented with thermal Langevin noise represents the effect of those degrees of freedom not explicitly taken into account.

The second package is ESPResSo++ [3, 4], which is based upon Molecular Dynamics but adds lots of mesoscale aspects like DPD, coupling to a Lattice Boltzmann fluid, and, in particular, is well suited for systematic coarse–graining applications.

Finally, there are modules for a new implementation of the Grand Canonical Adaptive resolution scheme (GC-AdResS) scheme in the MD package GROMACS. Briefly, AdResS is a scheme to partition a simulation box and treat different regions with different degrees of accuracy, and combine the advantages of classical atomistic simulations with those from coarse-grained simulations. GC-AdResS is the grand-canonical version, which couples the system to a particle reservoir. For more details please see Refs. [5, 6, 7]. GC-AdResS is already implemented in several program packages, among which GROMACS has posed the biggest coding challenge. The existing solution of this problem is far from satisfactory, which was the motivation for the development of the present suite of modules, which is partly presented in Sec. 4. The new approach significantly simplifies the GC-AdResS implementation and also generalizes it. The benefit is a much easier interfacing with existing codes (such as GROMACS), combined with an improvement of performance. The new implementation not only scales much better, but it is also hardware platform independent.

The modules that are being presented here have been developed by the E-CAM programmer Jony Castagna (Sec. 2.1) and the E-CAM Postdoctoral Research Associate (PDRA)s Silvia Chiacchiera (Sec. 2.2), Hideki Kobayashi (Sec. 3), and Christian Krekeler (Sec. 4). The PDRAs worked within the framework of pilot projects which implied a strong interaction with industry (S. Chiacchiera: Unilever; H. Kobayashi: Michelin, C.Krekeler: MODAL AG).

All three packages have previously (in deliverable 4.1 [8]) been identified as important codes for mesoscale simulations within E-CAM.

1.1 Overall scope of the module set

The modules presented in this report were generated within three projects.

1.1.1 Optimization of DL_MESO_DPD

Related to the DPD code DL_Meso_DPD, we present two modules in this deliverable that are related to its performance optimization. The first involves the porting of DL_Meso_DPD to multiple graphics cards (GPUs). This follows the previous work presented in deliverable D4.2 [9] where a first version on a single GPU has been presented.

The second module utilizes the SIONlib library, a scalable I/O library for parallel access to task-local files, to write/read the trajectory (HISTORY) files in DL_MESO_DPD.

1.1.2 Fast equilibration of dense polymer melts in ESPResSo++

The present document presents four ESPResSo++ modules that deal with dense polymer systems represented by atomistic models or simplified bead–spring models. For such systems, it is exceedingly difficult to computationally generate samples that are in proper thermal equilibrium, where the chains take random coil conformations and are nicely packed with respect to each other. This problem becomes progressively worse with increasing molecular weight and has plagued the polymer simulation community for decades. It cannot be solved by simply adding more computer power, because in a straightforward Molecular Dynamics simulation it is not due to insufficient sample sizes but rather insufficient observation time. Recently, there was a breakthrough made by Guojie Zhang et al. [10], where a system of high molecular weight is mapped onto an equivalent one with lower molecular weight (i. e. a more coarse-grained system). The latter system can be equilibrated more easily. After generating such a conformation, the more detailed description is re-introduced. This procedure can be done recursively, such that one obtains a method that is in spirit quite similar to multigrid methods. The modules provided here aim at making this method available to a wide audience of academic and industrial users.

1.1.3 New implementation of the GC-AdResS scheme in GROMACS

This deliverable presents two modules mainly describing an Abrupt-GC-AdResS setup. In the previous GROMACS implementation, GC-AdResS appeared as a generic force kernel in a very specific way, which had to be updated for each and every new GROMACS release. Furthermore, this approach provided only poor performance and scalability, which severely limits its applicability to "real–life" (or industrial) problems. The goal of the E-CAM modules was to improve the implementation and relieve this bottleneck.

We therefore present a way to turn GC-AdResS into a set of modules that are fairly independent from the main program. The basic idea is to first define the spatial regions corresponding to the atomistic or the coarse-grained simulations. Then these regions are in turn directly described by their respective neighbor lists. The interactions are treated by using the pre-defined standard interactions of GROMACS. In this way, the generic force kernel is redundant and can be removed. AdResS-specific code (which is of course somewhat slower) is then only executed in the interfacial region — but this typically comprises only a fairly small fraction of the overall system.

First tests and simulations show an increase in performance and scalability. Since the generic force kernel is removed, the integrated hardware specific force kernels can be used. For this reason, the new implementation is not anymore limited to CPUs. Furthermore, the removal of AdResS from the central force kernel means that the AdResS-enhanced version of GROMACS is now much more in line with the "plain" version (i. e. without AdResS), such that it automatically takes full advantage of further optimizations of the latter.

1.2 General applications and possible exploitation of the codes

DPD is routinely used in an industrial context to find out the static and dynamic behavior of soft-matter systems. Examples include colloidal dispersions, emulsions and other amphiphilic systems, polymer solutions, etc. Such materials are being produced or processed in industries like cosmetics, food, pharmaceutics, biomedicine, etc. Porting the method to GPUs (Sec. 2.1) is thus inherently useful in order to provide cheaper calculations.

The implementation of SIONlib in DL_Meso_DPD allows the users to reduce several output trajectory files to just a single one, which is a big advantage, for example for maintaining the simulation output, in addition to possibly improving the scalability of the code.

Concerning the modules in ESPResSo++, it should be mentioned that there is direct interest in the complex mechanical (or viscoelastic) behavior of dense polymer systems e. g. in industries like oil, plastics, and rubber (last not least car tires). The modules have been developed in direct collaboration with the tire industry (Michelin). Equilibrated samples can be subjected to externally imposed deformations, such that the (linear and nonlinear) viscoelastic response (i. e. the time-dependent stresses in the material) can be determined. This opens the perspective of computer-assisted material design, where the influence of molecular weight, molecular architecture and molecular chemistry can be studied. There is particular interest in the behavior of block copolymers, and future software development steps will aim at the ability to study such more complex architectures.

Concerning the GC-AdResS scheme, it should be realized that it is a very general method which can be applied to any classical MD simulation or beyond. The main purpose of the GC-AdResS modules presented here is to reduce the method to be merely an interface in physical space that is added to the MD code, whose force kernel remains untouched. Thus GC-AdResS is independent of the hardware architecture, and does not hinder the code's performance and scalability considerably.

The previous implementation was a part of the official GROMACS release up to version 5.1.4. Since GROMACS is one of the most prominent packages in the field of MD simulations (mainly because of its high performance and scalability) one of our goals is to encourage the re-implementation into the most current GROMACS version. And since GROMACS is a rather complicated code, our wish is also to present a scheme which can be plugged in easily, and that can also work with other codes. This new procedure removes the biggest argument against the previous implementation of GC-AdResS, where the forces had to be modified during the simulation, reducing the efficiency of the force kernel and thus the efficiency of the MD code.

This new setup retains all the features of merit of the previous GC-AdResS, which means it is possible to reduce the size of the simulation and production time. It is possible to use it as a tool to look at local properties, without the need to ignore the fact that the local degrees of freedom are strongly coupled with a much larger environment. We have tested and reproduced previous results for water (as the most simple case), tetrahedral molecules and 1,3-dimethyl-imidazolium chloride. Currently we are in contact with Animesh Argawal (previous student and now postdoc researcher in Los Alamos) and Christoph Junghans (implemented the first GC-AdResS in GROMACS and is also now in Los Alamos). Animesh Argawal is currently applying this scheme to path integral simulations, while Christoph Junghans is generally active in the field of development of MD codes.

1.3 How to read this report

For each module, we give a short overview, followed by links to the `Merge-Request` and the `Documentation` on the GitLab service of E-CAM, which shows detailed information about code development, testing and documentation. The documentation shows how to use the modules in practice, while possible practical exploitation and industrial applications have been outlined above (partly, some additional remarks are added below).

2 Modules based on DL_MESO_DPD

The base code for the following two modules is DL_MESO_DPD [1, 2], the Dissipative Particle Dynamics (DPD) code from the mesoscopic simulation package DL_MESO, developed by M. Seaton at Daresbury Laboratory. This open source code is available from Science and Technology Facilities Council (STFC) under both academic (free) and commercial (paid) licenses.

2.1 Multi GPU version of the DL_MESO_DPD code

A preliminary multi-GPU version of the DL_MESO_DPD code has been developed. This follows the previous work presented in deliverable D4.2 [9] where a first version on a single GPU has been presented.

2.1.1 Module description

In the current module, the main framework of this multi-GPU version has been developed. To achieve good scaling, the exchange of data between GPUs overlaps with the computation of the forces for the internal cells of each partition assigned to each GPU (a domain decomposition approach based on the MPI parallel version of DL_MESO_DPD has been followed, with each GPU assigned to one MPI task).

The current implementation relies on a slow transfer of the data GPU-CPU-CPU-HOST. Faster implementations will be explored in next modules. However, the transfer of data occurs in 3 steps: x-y planes first, x-z planes with halo data (i.e. the values which will fill the ghost cells) from previous swap and finally the y-z planes with all halos. This avoid the problems of the corner cells, which usually requires a separate communication reducing the sendreceive call from 14 to 6.

The multi-GPU version has been currently tested with 8 GPUs and successfully reproduced same results of a single GPU within machine accuracy resolution. Future plan are the benchmarking of the code with different data transfer implementations than the current trivial GPU-host-GPU transfer. These are: of Peer To Peer communication within a node, CUDA-aware MPI and CUDA-aware MPI with Direct Remote Memory Access (DRMA).

2.1.2 Motivation and exploitation

Due to the limit of the memory of GPUs (nowadays around 16GB), the current version of DL_MESO_DPD on a single GPU is limited to a maximum of 5M particles. This represent a strong limit to real application where complex systems can easily overcome that number (polymers or surfactant are examples of large systems with hundred millions of particles).

To overcome this limit, a multi-GPU version is currently under development. This will allow to use latest supercomputers with hydbrid architectures (cluster of nodes with CPU and GPU in each of them) and then targeting real exascale applications.

Merge Request	Merge Request for multi-GPU version of DL_MESO_DPD
Direct Documentation Link	Module under review. Soon available in our Software Library

2.2 Using SIONlib (parallel I/O library) to write/read HISTORY files in DL_MESO_DPD

This module makes use of the SIONlib library to write/read the trajectory (HISTORY) files in DL_MESO_DPD. In the last release (2.6, dating November 2015), the MPI version of DL_MESO_DPD generates *multiple* trajectory files, one for each MPI task (which is common bottleneck for extreme scalability since it stresses the underlying filesystem).

2.2.1 Module description

The use of SIONlib allows to minimally modify the writing so that just *one* physical file (history.sion) is produced. An analogous modification has to be implemented in the post-processing utilities that read the HISTORY files. As an example, here the modifications are implemented for one specific utility, `format_history_sion.f90`, a formatting tool analogous to `format_history.f90`. Beside showing how to adapt the reading, this allows a robust check of the implementation, since the output is human readable, contains the full trajectories, and can be readily compared with that obtained using `format_history.f90` with the standard version of DL_MESO_DPD.

Note that the next release version of DL_MESO_DPD (2.7, in development) will tackle the writing of files differently, producing a single trajectory file from the outset using MPI I/O. However, the interface proposed here provides this feature to the users of version 2.6, and represents an alternative solution for the handling of the trajectories.

2.2.2 Motivation and exploitation

We would like to emphasize that, while SIONlib is optimized for a large number of MPI tasks, the reduction from several output files to just one is in any case a benefit, for example when it comes to the maintenance of the simulation output.

The interfacing presented in this module has been conceived during the WP 4 Extended Software Development Workshop (ESDW) held in Barcelona, in July 2017. It is available under BSD license, for the post-processing part, and under DL_MESO licence, for the base code.

Merge Request	Merge Request for Interface of DL_MESO_DPD and SIONlib
Direct Documentation Link	Documentation for the Interface of DL_MESO_DPD and SIONlib

3 Modules based on ESPResSo++: Hierarchical strategy for simple one-component polymer melts

ESPResSo++ [3, 4] (“Extensible Software Package for Research in Soft Matter based upon C++”) is a general-purpose simulation package for soft-matter research, mainly developed at the Max Planck Institute for Polymer Research Mainz. It is freely available under the GNU Public License.

We here present a suite of programs which implement the recursive equilibration strategy of Guojie Zhang et al. [10] within ESPResSo++. In our opinion, the strategy is so far the best method to efficiently find well-equilibrated polymer configurations. It is hoped that the port into a freely available package will make this novel method usable and popular within the community of polymer simulators at large, both in an academic and an industrial context. Figure 1 illustrates the hierarchical equilibration strategy used, which is further explained below.

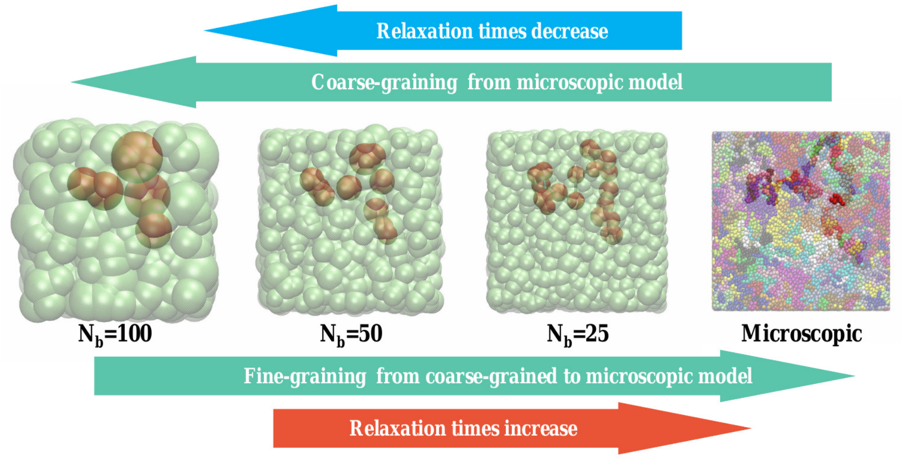
Relaxation times decrease
Coarse-graining from microscopic model
N_b=100
N_b=50
N_b=25
Microscopic
Fine-graining from coarse-grained to microscopic model
Relaxation times increase

Figure 1: Schematic illustration of the hierarchical equilibration strategy.

To decrease the relaxation time, microscopic monomers are Coarse Grained (CG) by mapping each subchain with N_b monomers onto a soft “blob”. The CG system is then characterized by a much lower molecular weight and thus is equilibrated quickly. One thus obtain a configuration that is equilibrated on large scales but does not provide information about the structure on smaller (i. e. more Fine Grained (FG)) scales.

To obtain the latter, the resolution is increased by applying a fine-graining procedure to the previous (more coarse-grained) level. In such a fine-graining step, each CG polymer chain is replaced with a more fine-grained chain, by dividing a CG blob into several FG blobs.

Both coarse-graining and fine-graining are done recursively such that the lowest level is a polymer melt with full atomistic details, while the top level is a melt that contains chains with only a very low degree of polymerization (in the extreme case, just $N = 1$), composed of monomers that are very soft blobs.

The FG blobs that result from the fine-graining step are set up in such a way that their conformation is consistent with the conformation at the more coarse-grained level. After this setup, the local FG conformation is relaxed into a local equilibrium, again consistent with the (fixed) CG blobs.

3.1 Coarse-Graining

The actual coarse-graining step is done by (i) a C++ code (module Md-Softblob, described in Deliverable D4.2 [9]), plus (ii) a controlling Python script, which is delivered presently. For the convenience of the reader, let us describe the procedure again:

A polymer chain, originally consisting of N monomers, is replaced by a coarse-grained (CG) chain consisting of N/N_b soft blobs linked by a harmonic bond potential, $V_{bond} = 3k_B T d^2/2b_{CG}^2$, and an angular bond-bending potential $V_{bend} = k_B T k_{bend}(1+\cos(\theta))/2$. Here d is the distance and θ is the angle between consecutive bonds. The interactions between non-bonded soft blobs are taken into account by a repulsive pair potential $V_{nb} = k_B T \epsilon U_G(r_{ij})$. Here r_{ij} is the center-to-center distance between the two blobs, $U_G(r_{ij})$ is a Gaussian function with variance $\overline{\sigma}^2 = \sigma_i^2 + \sigma_j^2$ and σ_i is the gyration radius of blob number i. The gyration radius σ is in turn fluctuating. This fluctuation is controlled by the potential $V_{sphere} = k_B T(a_1 N_b^3 \sigma^{-6} + a_2 N_b^{-1} \sigma^2 + a_3 \sigma^{-3})$. Equilibrated configurations of soft blobs are generated by Molecular Dynamics (MD) simulations based on the above model.

Merge Request	Merge Request for Coarse-Graining
Direct Documentation Link	Documentation for Coarse-Graining

3.2 Fine-Graining

Here, the resolution is step-by-step increased by recursively applying a fine-graining procedure to the previous (more coarse-grained) level. In such a fine-graining step, each CG polymer chain is replaced with a more fine-grained chain, by dividing a CG blob into several FG blobs.

The present module provides the python script which performs this fine-graining procedure.

Merge Request	Merge Request for Fine-Graining
Direct Documentation Link	Documentation for Fine-Graining

3.3 Reinsertion

For re-inserting the microscopic details, the set of FG blobs is set up in such a way that its conformation is consistent with the conformation at the more coarse–grained level. After this setup, the local FG conformation is relaxed into a local equilibrium, again consistent with the (fixed) CG blobs.

The reinsertion procedure is divided into two parts. Firstly, monomers are treated as mass points without non-bonded interaction. Starting from this state, repulsive non-bonded interactions are gradually introduced according to the feedback control mechanism explained in Ref. [11]. This procedure makes sure that the final fine-grained conformation is consistent with the conformation at the more coarse-grained level.

The present module provides the python script which performs the reinsertion procedure.

Merge Request	Merge Request for Reinsertion
Direct Documentation Link	Documentation for Reinsertion

3.4 Feedback Control Mechanism (Fbloop)

The present module provides the python script which performs the feedback control mechanism mentioned in the previous subsection.

Merge Request	Merge Request for Fbloop
Direct Documentation Link	Documentation for Fbloop

3.5 Constrain-RG

A re-inserted set of FG blobs must be set up in such a way that its conformation is consistent with the conformation at the more coarse-grained level. Therefore the gyration radii of the subchains must coincide with the gyration radii of the corresponding blobs, during an initial equilibration period. The present module provides the C++ class for applying a suitable constraint that conserves the gyration radius of N microscopic monomers.

Merge Request	Merge Request for Constrain-RG
Direct Documentation Link	Documentation for Constrain-RG

4 Modules based on the Grand Canonical Adaptive Resolution Scheme

The Grand Canonical Adaptive resolution scheme gives a methodological description to partition a simulation box into different regions with different degrees of accuracy. For more details on the theory see Ref. [5, 6, 7]. The coupling between the regions is done via an interpolation of forces. Technically, the simulation box is divided into 3 regions, the atomistic (AT) and the coarse grained (CG), which are coupled via a transition region. This hybrid region (name in previous GC-AdResS) or Δ (name in new implementation) has to be of the size of the cut-off radius, or larger, to make sure that the AT and the CG regions do not interact. In the previous ansatz (upper half of Fig. 2) the coupling of the AT and the CG regions was done via a smooth interpolation of the forces in the hybrid region, where w_i(x) is a $\cos^2$ function. In the AT region w_i(x)=1, while it is w_i(x)=0 in the CG region.

$$F_i = \sum_j w_i(x) w_j(x) F_{ij}^{at} + \sum_j (1 - w_i(x) w_j(x)) F_{ij}^{cg} + F_{th}$$

The F_{th} represents the thermodynamic force and ensures that the chemical potential in the AT and the CG regions are the same.

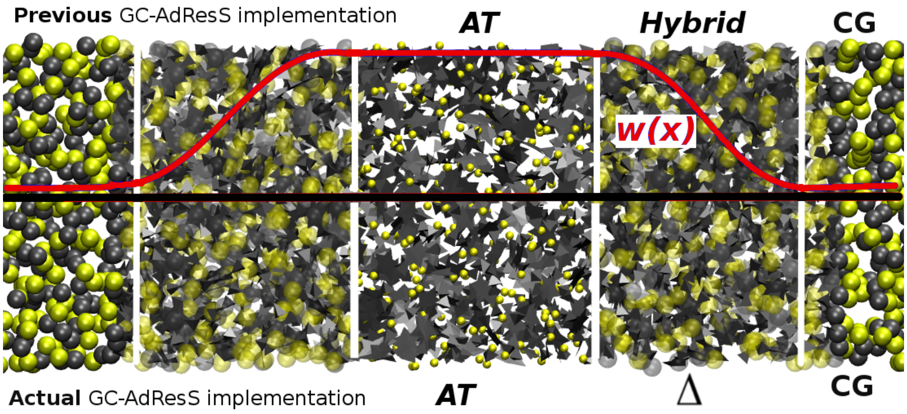
Previous GC-AdResS implementation
AT
Hybrid
CG
w(x)
Actual GC-AdResS implementation
AT
Δ
CG

Figure 2: Schematic comparison between the previous GC-AdResS implementation and the actual one presented here, based on 1,3-Dimethylimidazolium chloride simulations. The old scheme requires a weighting function in the hybrid region w(x). In this way the atomistic degrees of freedom are slowly switched on in a controlled way. The new scheme (lower half) still requires a transition region Δ, but here the switching function is implicitly included in the interactions and the thermodynamic force.

This scheme is implemented in several MD packages in some form (GROMACS [12, 13], ESPResSo++ [3, 4], and LAMMPS). In GROMACS it is implemented up to version 5.1.4. So far, the speedup was only up to a factor of 2, independent of the studied systems. The reason for that is the above shown force interpolation is working on the forces directly. In GROMACS, this scheme is implemented in a generic force kernel. The kernel is not optimized and not parallelized, which creates a bottleneck. Our main project goal is to implement GC-AdResS in other codes, and since one has to touch very well optimized force kernels, this makes it difficult. The best way to achieve our goal was to change our ansatz (lower half of Fig. 2). Instead of trying to modify the force kernel, we are now using the existing internal structures of the code, the force kernel, the standard interactions and the neighbor list search. With this approach the theoretical conditions to couple two systems are still intact. And it is technically much easier to implement a partitioning into a code than adjusting forces kernels. The new implementation is now merely an interface introduced into the code, the force kernel is left untouched, i.e. GC-AdResS is now independent of the hardware architecture, and scales considerably better than the old implementation. This is a huge step towards our final goal, the spreading of GC-AdResS as a standard method.

Two of the modules presented show a new ansatz to partition the neighbor list and how to remove GC-AdResS from the force kernel while at the same time retain the features of the scheme, thus making the new implementation as general as possible. Our goal is to encourage the re-implementation into the current GROMACS version and implementation in other codes.

A manual to start/run GC-AdResS implemented in GROMACS (up to version 5.1.4) is presented in this deliverable as supporting information (see section A).

4.1 Abrupt-GC-AdResS

This module presents a very straightforward way to implement a new partitioning scheme and how to utilize a constant weighting function. This also solves two problems with the performance: the neighbor list search and the generic force kernel.

The original idea of our project was to work on a general implementation of GC-AdResS in classical MD packages. However the previous implementation of GC- AdResS in GROMACS has several performance problems. We know that the main performance losses of the old GC-AdResS simulations in GROMACS are in the neighbor list search and the generic force kernel, linking the At and CG forces together via a smooth weighting function. The neighbor list search and the force kernel are linked. Thus, to remove the bottleneck with respect to the performance and the hindrance regarding the easy/general implementation into other MD codes, and also to avoid the problem of the not optimized (and not parallelized) force kernel used in GROMACS, we changed the neighbor list search. In GROMACS, the neighbor list is put together and organized in the file *ns.c*. In GROMACS 5.1.0 there are two functions which basically sort the incoming particles into the different neighbor lists. In the previous implementation of GC-AdResS everything other than CG (with w_i=w_j=1) or AT (with w_i=w_j=0) is sorted into the neighbor lists. Any other particles are sorted into a special neighbor list designed only for AdResS.

We have now changed this neighbor list sorting, and everything is taken into account other than: (AT and (w_i=0 or w_j=0)) or (CG and (w_i>=0 and w_j>=0)). This leads to 5 distinct interactions: (1) AT-AT in the AT region, (2) CG-CG in the CG region, (3) AT-AT between particles in the hybrid region, (4) AT-AT between particles of the AT region with the hybrid region and (5) CG-CG between particles of the CG region with the hybrid region. The if statement excludes the CG-CG interaction in the hybrid region.

Then, we basically switched from the generic kernel to the standardized (optimized and paralellized) force kernels predefined in GROMACS. This procedure lead to a considerable speed up of the code not only compared to the previous GC-AdResS but also compared to full atomistic simulations. The result is shown in Fig. 3.

We tested this new implementation on SPC water with varying system sizes (a paper has been submitted to Physics Review Letters). GROMACS is optimized especially for handling of bio-systems, i.e. GROMACS has the best performance in case of water simulations. We set up a couple of Abrupt GC-AdResS simulations ranging from small 6912 water molecules to 48k water molecules. We used a standard desktop machine (Intel Core i5-4590 CPU @ 3.30GHz x4) and run small 20 ps runs. We can see that our performance is much improved.

Another important part of this new implementation is that it decouples GC-AdResS directly from the core of any MD code (the highly optimized force kernel). This has a couple of implications:

- For GROMACS, that means it leaves the selection of the force kernel to the program, which means it will also run with the CUDA supported version of GROMACS.
- This enables us to implement the scheme into other codes much easily, or convince the developers to do that.
- We now might even be able to use external libraries/patches like MIST or PLUMED.

Merge Request	Abrupt GC-AdResS: A new and more general implementation
Direct Documentation Link	Module under review. Soon available in our Software Library

4.2 Abrupt-AdResS_forcecap

The implementation of Abrupt GC-AdResS is in itself only working for the smallest and simplest of molecules without problems. For larger and more complex molecules the simulation crashes. However, that is nothing new. As studies of ionic liquids and polymer melts have shown us for large and complicated molecules even the standard GC-AdResS is not working. The reason for that is when a molecules from the coarse grained region enters the hybrid region the atomistic representations, which are present due to the technically necessary double resolution, interact. It is possible for atoms to be too close together, which results in a too high forces and thus in too high velocities for those

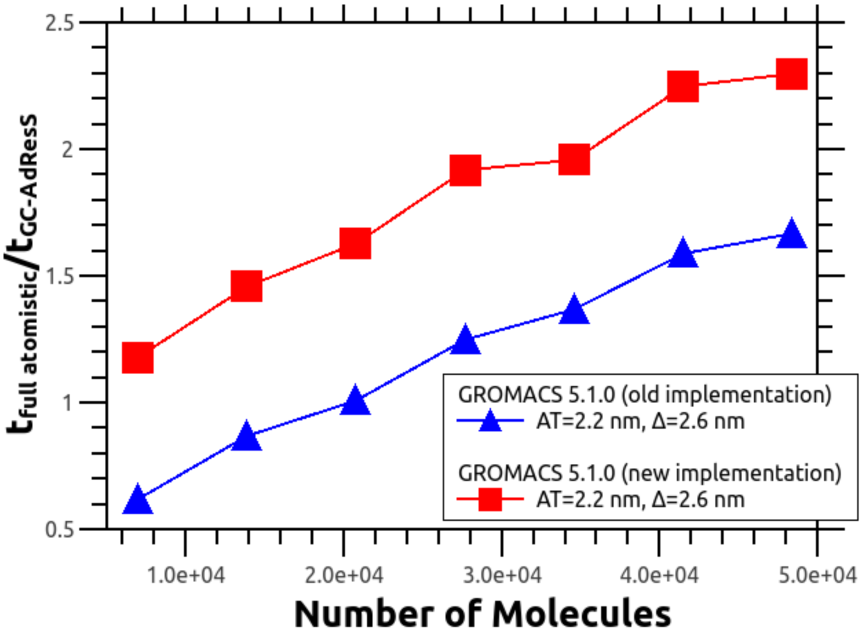

Figure 3: Performance test of the new GC-AdResS implementation in GROMACS 5.1.0 compared to full atomistic and old GC-AdResS simulations. The test was done for SPC water at varying system sizes. The atomistic region was 1.6 nm, the Δ region was 2x1.1 nm, the rest was coarse grained.

particles. Since in Abrupt GC-AdResS (previous module) we can avoid the generic force kernel from GROMACS the force capping (which was previously implemented at the end of the force calculation) had to be shifted. Several trials to put it directly after the force calculation did not work out. So we finally looked at the one place where each force has to be read and handled, the (in our case) stochastic dynamics integrator. The force capping is rather simple. If the force gets too high it is re-scaled to a given value and therefore does not destroy the molecule and causes the program to crash.

Recipe:

```
fc = Chosen upper force limit for the ionic liquids simulations
fn = force acting on a particle
if (absolute value (fn) larger then fc)
then
fn = fc*fn/(absolute value (fn))
end if
```

A note of caution: one has to choose a high enough force, otherwise normal interactions will trigger the force capping and unnecessary change the interactions. That would change the dynamics and the structure of a system. If chosen too high it might run into an impossible and unstable configuration, which will also result in a program crash.

We have tested the new addition to the code on the same water systems as before and the performance is not affected. Furthermore we run a couple of ionic liquids simulations (1,3-Dimethyl-imidazolium chloride from 1000 ion pairs to 50000 ion pairs) and measured the performance, see Fig. 4. For the dynamics and structure check we looked at the 1000 ion pair system. We can reproduce the radial distribution functions in the atomistic region. The density in the atomistic, transition (Δ) region is less than 3% different from full atomistic simulations and the density diffusion profile shows a complete mixing of the different regions.

Merge Request	Abrupt_AdResS_Forcecapping
Direct documentation link	Module under review. Soon available in our Software Library

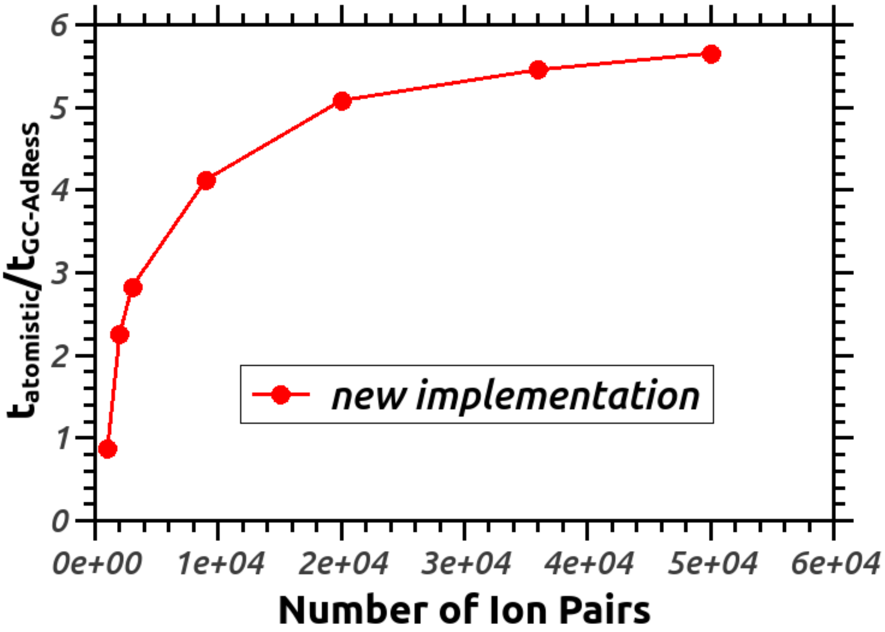

Figure 4: Performance test (on 1 core) of the new GC-AdResS implementation in GROMACS 5.1.0 with the force capping compared to full atomistic simulations. The test was done for 1,3-Dimethyl-imidazolium chloride at varying system sizes with 3 nm atomistc region, 2x3 nm Δ region and the rest is coarse grained.

5 Outlook

The deliverable contains nine modules that are integrated in three big software packages: DL_MESO_DPD, ESPResSo++, and GROMACS, that focus on meso– and multi–scale modelling.

The usability of DL_MESO_DPD has been significantly enhanced by porting it to Multi-GPU platforms, and by offering the opportunity to analyze trajectories based upon just a single file. Development along these lines will continue in the future; in particular the study of dielectric systems and of electrostatics shall continue.

The implementation of the hierarchical strategy for dense polymer systems is now close to complete. We are currently exploring new variants where the interaction between coarse-grained blobs is modeled in a more accurate way, based upon liquid-state theory.

We furthermore deliver a recipe for a more widely implementable version of the GC-AdResS scheme and two related modules. We have tested this ansatz on various systems (water, tetrahedral, and ionic liquids) and found a much better performance than with the old implementation. Furthermore, we removed a critical obstacle for other developers to introduce GC-AdResS into their MD packages. This is a significant step towards exporting this scheme into new MD packages. Our next steps will be combining the quantum methods from E-CAM WP 3 (Quantum Dynamics) with GC-AdResS.

All of the modules have been accepted into the E-CAM software library or were under final review at the time of submission.

A Appendix

A.1 AdResS-manual

This manual explains the use of GC-AdResS implemented in GROMACS (up to version 5.1.4).

For the simulation to work, one has to have tabulated coarse grained (CG) potentials. The source of those CG potentials is not important. We concentrated, for simplicity, on ones generated via inverse Boltzman iteration (as included in the VOTCA package).

The other ingredient is the thermodynamic force. VOTCA (up to version 1.3) provides a routine to calculate this force (for details see: Delle Site L., Agarwal A., Junghans C., Wang C., https://arxiv.org/abs/1412.4540; or Wang, H., Hartmann, C., Schütte, C., Delle Site, L. (2013). Grand-canonical-like molecular-dynamics simulations by using an adaptive-resolution technique. Physical Review X, 3(1), 011018).

Additional information:

- The configuration of the system requires a double resolution setup, i.e. CG particles and the AT presentation of molecules are co-existing in the same box (this is one of the bottlenecks in the simulation).
- Setup of the coarse grained potential via VOTCA program (see: www.votca.org).
- Setup of double resolutions start configuration:
  After generating an optimized full atomisitc simulation box, the "csg_dump" command shows the particles included. The resulting information can be used to setup a "mapping" xml files (mapping from AT to CG particles). The tool "csg_map" then allows to use the mapping to generate the double resolution setup, i.e. CG and the atomistic particles are in the same simulation box. This setup has to be considered in the topology file. In GROMACS the CG particles have to be added as virtual particles to the topology file and the definition of the virtual sides have to be added in the force field file.
- Inverse Boltzmann Iteration (for details of the required options needed see: www.votca.org) is used in our current simulations. In theory any CG potential is working, the main requirement is that the CG region is properly represented (e.g. chemical potential, density, solution).
- The thermodynamic force (TF) (for details of the required options needed see: www.votca.org) is used to determine the TF for simulations with the currently implemented GC-AdResS in the official GROMACS releases (up to version 5.1.4). After a converged thermodynamic force is found, it can be used for longer production runs or as initial guess for similar systems.
- The necessary changes in the GROMACS input.mdp file for the GC-AdResS simulation:
  - In GROMACS the stochastic dynamics (integrator = sd) integrator is the only one which can be used for GC-AdResS.
  - Then currently (as another bottleneck) only tabulated potentials are implemented, thus the following lines have to be added and adjusted: energygrps = AT CG
    energygrp_table
  - Furthermore in its current implementation only "cutoff-scheme = group" works. The coulomb type has to be set to "coulombtype = reaction-field" or "coulombtype = reaction-field-zero" and van der Waals interaction type to "vdw-type = user".
  - The important part of the input file is the following section, which describe the setup of the simulation box. In GROMCAS, two choices are possible. The first type is spherical around the provided reference coordinate and the second type is a slab like setup (called xsplit). If the type constant is chosen, the value "adress_const_wf" (wf = weighting function) determines how the entire simulation box is treated (wf = 1 is full atomistic, wf=0 is CG, 0<wf<1 is in the hybrid region).

```
adress = yes ;no
adress_type = sphere ;xsplit sphere or constant
adress_const_wf = 1 ;nur für constant
adress_ex_width = radius/size of the atomistic region
adress_hy_width = size of the hybrid region
adress_ex_forcecap = prevents 2 particles coming too close in the hybrid region
adress_interface_correction = off ;thermoforce
adress_site = com
adress_reference_coords = reference point of the atomistic region
adress_tf_grp_names = the name for the thermoynamic force tables
adress_cg_grp_names = name of the CG particles in the CG region
adress_do_hybridpairs = no
```

Merge Request	Manual to start/run Grand Canonical Adaptive Resolution Simulations
Direct documentation link	Manual under review. Soon available in our Software Library

References

Acronyms Used

WP	Work-package
ESDW	Extended Software Development Workshop
DPD	Dissipative Particle Dynamics
STFC	Science and Technology Facilities Council
GC-AdResS	Grand Canonical Adaptive resolution scheme
PDRA	Postdoctoral Research Associate
CG	Coarse Grained
FG	Fine Grained
WP	Work-package

URLs referenced

Page ii

https://www.e-cam2020.eu . . . https://www.e-cam2020.eu
https://www.e-cam2020.eu/deliverables . . . https://www.e-cam2020.eu/deliverables
Internal Project Management Link . . . https://redmine.e-cam2020.eu/issues/159
duenweg@mpip-mainz.mpg.de . . . mailto:duenweg@mpip-mainz.mpg.de
http://creativecommons.org/licenses/by/4.0 . . . http://creativecommons.org/licenses/by/4.0

Page 1

DL_MESO_DPD . . . https://www.scd.stfc.ac.uk/Pages/DL_MESO.aspx
ESPResSo++ . . . http://www.espresso-pp.de/
GROMACS 5.1.0 . . . http://www.gromacs.org/

Page 2

GROMACS . . . http://www.gromacs.org/
SIONlib . . . http://www.fz-juelich.de/ias/jsc/EN/Expertise/Support/Software/SIONlib/_node.html

Page 4

GitLab service of E-CAM . . . https://gitlab.e-cam2020.eu/

Page 5

DL_MESO . . . https://www.scd.stfc.ac.uk/Pages/DL_MESO.aspx
Merge Request for multi-GPU version of DL_MESO_DPD . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/26
Software Library . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modules/index.html
SIONlib . . . http://www.fz-juelich.de/ias/jsc/EN/Expertise/Support/Software/SIONlib/_node.html
SIONlib . . . http://www.fz-juelich.de/ias/jsc/EN/Expertise/Support/Software/SIONlib/_node.html

Page 6

Merge Request for Interface of DL_MESO_DPD and SIONlib . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/
merge_requests/4
Documentation for the Interface of DL_MESO_DPD and SIONlib . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modules/modules/DL_MESO_DPD/sionlib_dlmeso_dpd/readme.html

Page 7

Merge Request for Coarse-Graining . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules
merge_requests/39
Documentation for Coarse-Graining . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Mo
modules/hierarchical-strategy/simple_one-component_melts/coarse-graining/readme.html
Merge Request for Fine-Graining . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules/
merge_requests/40

Documentation for Fine-Graining . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modu
modules/hierarchical-strategy/simple_one-component_melts/fine-graining/readme.html
Merge Request for Reinsertion . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules/merge_requests/41
Documentation for Reinsertion . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Module
modules/hierarchical-strategy/simple_one-component_melts/reinsertion/readme.html
Merge Request for Fbloop . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules/merge_requests/42
Documentation for Fbloop . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modules/modules/hierarchical-strategy/simple_one-component_melts/fbloop/readme.html

Page 8

Merge Request for Constrain-RG . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules/merge_requests/31
Documentation for Constrain-RG . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modu
modules/hierarchical-strategy/components/constrain-rg/readme.html

Page 9

Abrupt GC-AdResS: A new and more general implementation . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules/merge_requests/38
Software Library . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modules/index.html

Page 10

Abrupt_AdResS_Forcecapping . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules/merge_requests/43
Software Library . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modules/index.html

Page 13

https://arxiv.org/abs/1412.4540 . . . https://arxiv.org/abs/1412.4540
www.votca.org . . . www.votca.org
www.votca.org . . . www.votca.org

Page 14

Manual to start/run Grand Canonical Adaptive Resolution Simulations . . . https://gitlab.e-cam2020.eu/e-cam/Meso-Multi-Scale-Modelling-Modules/merge_requests/33/diffs
Software Library . . . http://e-cam.readthedocs.io/en/latest/Meso-Multi-Scale-Modelling-Modules/index.html

Citations

[1] http://www.cse.clrc.ac.uk/ccg/software/DL_MESO/.

[2] M. A. Seaton, R. L. Anderson, S. Metz, and W. Smith, "DL_meso: highly scalable mesoscale simulations," *Molecular Simulation*, vol. 39, no. 10, pp. 796–821, Sep. 2013.

[3] http://www.espresso-pp.de/.

[4] J. D. Halverson, T. Brandes, O. Lenz, A. Arnold, S. Bevc, V. Starchenko, K. Kremer, T. Stuehn, and D. Reith, "ESPResSo++: A modern multiscale simulation package for soft matter systems," *Computer Physics Communications*, vol. 184, no. 4, pp. 1129–1149, Apr. 2013.

[5] L. Delle Site and M. Praprotnik, "Molecular Systems with Open Boundaries: Theory and Simulation," *Phys. Rep.*, vol. 693, pp. 1–56, 2017.

[6] H. Wang, C. Schütte, and L. Delle Site, "Adaptive Resolution Simulation (AdResS): A Smooth Thermodynamic and Structural Transition from Atomistic to Coarse Grained Resolution and Vice Versa in a Grand Canonical Fashion," *J. Chem. Theory Comput.*, vol. 8, pp. 2878–2887, 2012.

[7] H. Wang, C. Hartmann, C. Schütte, and L. Delle Site, "Grand-Canonical-Like Molecular-Dynamics Simulations by Using an Adaptive-Resolution Technique," *Phys. Rev. X*, vol. 3, p. 011018, 2013.

[8] I. Pagonabarraga, B. Duenweg, V. Lobaskin, G. Sutmann, C. Hartmann, and L. Petit, “Identification/Selection of E-CAM meso and multi- scale modelling codes for development,” Oct. 2016. [Online]. Available: https://doi.org/10.5281/zenodo.841696

[9] B. Duenweg, J. Castagna, S. Chiacchera, and H. Kobayashi, “Meso– and multi–scale modelling E-CAM modules I,” Oct. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.1191428

[10] G. Zhang, L. A. Moreira, T. Stuehn, K. C. Daoulas, and K. Kremer, “Equilibration of High Molecular Weight Polymer Melts: A Hierarchical Strategy,” *ACS Macro Letters*, vol. 3, no. 2, pp. 198–203, Feb. 2014. [Online]. Available: http://dx.doi.org/10.1021/mz5000015

[11] Moreira Livia A., Zhang Guojie, Müller Franziska, Stuehn Torsten, and Kremer Kurt, “Direct Equilibration and Characterization of Polymer Melts for Computer Simulations,” *Macromolecular Theory and Simulations*, vol. 24, no. 5, pp. 419–431, Jun. 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/mats.201500013

[12] http://www.gromacs.org/.

[13] B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl, “GROMACS 4: Algorithms for Highly Efficient, Load-Balanced, and Scalable Molecular Simulation,” *J. Chem. Theory Comput.*, vol. 4, p. 435, 2008.