GROMACS - Bug #1400

Can't reproduce results on DPPC with reaction field under version 4.5.3 compared to 4.0.7

12/10/2013 02:46 PM - Patrick Fuchs

Status:	Closed		
Priority:	Normal		
Assignee:	Berk Hess		
Category:	mdrun		
Target version:	4.6.6		
Affected version - extra info:		**Difficulty:**	uncategorized
Affected version:	4.5.3		

Description

I ran a 100 ns simulation of DPPC / water at 323K using Reaction Field electrostatics. Using gromacs 4.0.7, I was able to reproduce the results described in Poger's paper (http://dx.doi.org/10.1002/jcc.21396), that is, area per DPPC ~ 0.63 nm^2.
When I use the exact same input files under gromacs 4.5.3, the area drops to ~ 0.59 nm^2. I'm suspecting a problem with the implementation of reaction field in 4.5.3.
In Poger's paper (http://dx.doi.org/10.1002/jcc.21396), gromac 3.2.1 was used. In a more recent paper by the same authors, they used gromacs 3.3.3 (http://dx.doi.org/10.1021/ct300675z) and reported the same area (0.63 nm^2). So it looks like versions 3.3.* and 4.0.* are fine, and the problem arises for versions 4.5.* (I haven't tested 4.6.*).
Last, when I do a zero step md (thus single point energy calculation, it seems impossible to use steepest descent zero step with reaction field) with both gromacs versions 4.0.7 and 4.5.3, I don't see any difference in potential energy up to the 4th or 5th decimal.

A few details:
- I'm attaching a tarball (pb_RF_version453.tgz) with all the input files
- the files lipids_vdw.itp, example_lipids.top and dppc.itp were downloaded from David Poger's website (http://compbio.chemistry.uq.edu.au/~david/research/lipids.htm) on April 2012
- when I test the same simulations with PME electrostatics, I don't get any difference between 4.0.7 and 4.5.3
- both gromacs versions 4.0.7 and 4.5.3 were compiled with gcc 4.6.3
- I used (default) domain decomposition for both versions

Related issues:

Related to GROMACS - Bug #2884: GROMACS can not reproduce properties with the...	**Closed**

Associated revisions

Revision 0278ef87 - 05/22/2014 09:13 AM - Mark Abraham

Add fatal errors for VV and twin-range MTS

Michael never implemented the multiple-time stepping with the VV integrator family and constraints (see code that calls combine_forces() from update_coords() in src/mdlib/update.c). Probably that means the multiple-time-step regime was not tested with VV either. Strictly speaking, these new fatal errors have scope that is wider than is clearly warranted, but it is not clear the no-constraints VV path was only ever as bad as the broken leap-frog path is (see #1400).

I suspect VV+constraints will work with the incoming fixes for leap-frog, but until someone wants to use it (and why would they?), then I'm not going to test that it works as well as it does with leap-frog.

Change-Id: Ib61d0fb7661bca2101c04423a6af1744420c06ab

Revision a75db633 - 06/05/2014 02:18 PM - Berk Hess

Fix constraint virial with multiple time stepping

With multiple time stepping the additional nstcalclr-1 force contribution was constrained to remove it from the virial. This procedure neglected the non-linear contribution due to the rotation of constraints. Now the contribution of this force component to the coordinate update is constrained instead and

the corresponding virial contribution is subtracted from the
constraint virial.

Fixes #1400

Change-Id: If3217f52808bf7491998324f8dc3161bc003ec1b

History

#1 - 12/10/2013 04:57 PM - Justin Lemkul

Can you provide an .mdp or .tpr file? Which RF method did you use? It will be quite important to select the same one that the original authors did, which would correspond to "Reaction-Field-nec" in Gromacs 4.5 and beyond, given the old versions (3.2.1 and 3.3.3) cited.

#2 - 12/10/2013 05:41 PM - Patrick Fuchs

- File md_gmx407.tpr added

- File md_gmx453.tpr added

- File mdin.mdp added

- File area_RF.ps added

All the input files are in pb_RF_version453.tgz, I attach again the mdp and in addition I attach a tpr for both versions 4.0.7 and 4.5.3.
The RF method is "Generalized-Reaction-Field" as is cited in the two papers by Poger and Mark.

I also attach for convenience a plot of the area per lipid. The mdp file that I attached to this message correspond to RF2_gmx407 (solid red line) and RF2_gmx453 (dashed red line). There is in addition another simulation with 407 (named RF_gmx407, solid black line).

#3 - 12/11/2013 12:17 AM - Justin Lemkul

IIRC, the RF methods changed between 4.0.x and 4.5, so it is expected that the algorithms are producing dissimilar results. Can you please try again with Reaction-Field-nec under 4.5.3? This should be the equivalent algorithm to what the original authors used.

#4 - 12/11/2013 09:55 AM - Samuli Ollila

It is said in the manuals for 4.0 and 4.5.6 that Reaction-Field-nec is "The same as Reaction-Field, but implemented as in GROMACS versions before 3.3". Also I did not find any mention about this in the release notes. If I understand this correctly, there should not be difference between versions 4.0 and 4.5, or did I miss something?

#5 - 12/11/2013 02:48 PM - Justin Lemkul

Samuli Ollila wrote:

> It is said in the manuals for 4.0 and 4.5.6 that Reaction-Field-nec is "The same as Reaction-Field, but implemented as in GROMACS versions before 3.3". Also I did not find any mention about this in the release notes. If I understand this correctly, there should not be difference between versions 4.0 and 4.5, or did I miss something?

Presumably, 4.0.x and 4.5.x should be the same in this regard, but it needs to be established whether the expected algorithm (in this case, Reaction-Field-nec) is working as intended, which thus far has not been demonstrated. The lack of mention in the release notes is unfortunate, but for a long time, the notes only focused on highlights and were not necessarily as comprehensive as they are now. Sometimes things changed and were not mentioned there, but the manual has always been accurate.

#6 - 12/11/2013 04:04 PM - Samuli Ollila

According to Patrick's original post, the versions 3.2.1, 3.3.3 and 4.0.7 gives the same area per molecule 0.63 nm^2, but the version 4.5.3 gives 0.59 nm^2. If the manual is correct, then using Reaction-Field-nec in versions 3.3 and newer would be the same as Reacion-Field in 3.2.1. Since changing from version 3.2.1 to versions 3.3.3 and 4.0.7 does not change the area per molecule this does not seem to be the issue. However, something seems to be changed between 4.0.7 and 4.5.3. It would be nice to know what is this? Maybe running 4.5.3 simulation with Reaction-Field-nec would help in debugging, though?

#7 - 12/11/2013 04:06 PM - Justin Lemkul

Samuli Ollila wrote:

> Maybe running 4.5.3 simulation with Reaction-Field-nec would help in debugging, though?

That was my original suggestion. We need to be sure we're comparing apples to apples here. Running with different algorithms over different versions would expand the available data to help track down the issue.

#8 - 12/11/2013 04:15 PM - Patrick Fuchs

Justin Lemkul wrote:

> Samuli Ollila wrote:
>
> > Maybe running 4.5.3 simulation with Reaction-Field-nec would help in debugging, though?
>
> That was my original suggestion. We need to be sure we're comparing apples to apples here. Running with different algorithms over different versions would expand the available data to help track down the issue.

I launched a new simulation with 4.5.3 and the exact same input files but "coulombtype = Reaction-Field-nec". I will update once completed.

#9 - 12/18/2013 06:37 PM - Justin Lemkul

Filing this here so it doesn't get forgotten. Potentially very useful debugging information posted to gmx-users: https://mailman-1.sys.kth.se/pipermail/gromacs.org_gmx-users/2013-December/086257.html

#10 - 12/20/2013 06:49 PM - Patrick Fuchs

- File area_RF+RFnec.ps added

The simulation with gmx 4.5.3 and Reaction-Field-nec has completed. Attached in area_RF+RFnec.ps are the results, to sum up:
1st simulation
- RF gmx407 with Generalized-Reaction-Field : 62.7 A^2
2nd simulation with another seed:
- RF2 gmx407 with Generalized-Reaction-Field : 63.6 A^2
- RF2 gmx453 with Generalized-Reaction-Field : 58.6 A^2
- RF2_gmx453 with Reaction-Field-nec : 60.0 A^2

It seems that Reaction-Field-nec with gmx453 doesn't solve the problem and gives lower area compared to gmx407 Generalized-Reaction-Field.
Maybe Lutz can give feedback on the simulation with nstlist=1?

#11 - 12/21/2013 11:49 PM - Lutz Maibaum

- File comparison.png added

The attached graph provides some additional data that seems consistent with the ideas discussed here and on the related thread on the mailing list [1]:

- Changing the coulombtype from Reaction-Field to Reaction-Field-nec in Gromacs 4.6.5 doesn't seem to affect the resulting area per lipid.
- Changing nstlist from 5 to 1 in Gromacs 4.6.5 seems to reproduce the area per lipid obtained in Gromacs 4.0.7 with nstlist=5 (at severe computational cost).

[1] [http://gromacs.5086.x6.nabble.com/Gromos-DPPC-bilayer-different-results-for-4-0-7-and-4-6-5-td5013351.html]

#12 - 03/25/2014 11:20 AM - Patrick Fuchs

- File area_RF+rlist_eq_rcoulomb.pdf added

I finally tested the suggestion of Samuli (https://mailman-1.sys.kth.se/pipermail/gromacs.org_gmx-users/2013-December/086261.html), that is, increase rlist up to rlist=rcoulomb in Gromacs 4.5.3. Here are the results (see area_RF+rlist_eq_rcoulomb.pdf). We recover a correct area (63 A^2, see magenta curve "RF2_gmx453_rlistequalrcoulomb") similar to that obtained with 4.0.7.
I will post a message to the mailing-list to sum-up all these things and launch again the discussion.

#13 - 04/22/2014 07:06 PM - Patrick Fuchs

Since there hasn't been any reply to my last message in the mailing list, I put here a link to it so that it doesn't get lost: https://mailman-1.sys.kth.se/pipermail/gromacs.org_gmx-users/2014-March/088108.html. There my conclusion is: if for some reason someone wants to use the twin-range cut-off as it is done in the GROMOS software (with all versions of GROMOS force fields), one has to stick to GROMACS 4.0.* or lower.

#14 - 04/22/2014 09:23 PM - Mark Abraham

There's pretty limited interest from GROMACS devs in RF and/or twin-range setups, and the group-style cutoff-scheme itself is on the table for removal when we think the Verlet scheme does pretty much everything people should want to do. So, even if there was a bug introduced around the time of 4.5, the code path that contains it is likely going to go away. Moreover, for lipid systems, long-range effects are known to be important, and PME (and LJ-PME) seem so clearly the right way to do things that investing time fixing implementations of short-range-only methods that have known flaws does not seem like a good idea.

So, Berk's old comments (https://mailman-1.sys.kth.se/pipermail/gromacs.org_gmx-developers/2011-December/005506.html) are mostly addressed in the 4.6 Verlet scheme, except that RF intrinsically has a force discontinuity at the cutoff if the dielectric is finite, and the actual twin-range multiple-time-stepping itself is not supported. However, in the GPU era, the extra code complexity just to reduce flops is really not worthwhile. I can't see an obvious reason why single-range 1.4 should be less accurate than 0.8/1.4 unless the original parametrization was too tight to that scheme.

Ideally, the 4.6 Verlet scheme would already work for the single-range version of this setup, but it does not seem anybody has tried it. I have a run going. How are you guys measuring area per lipid? I've never done it.

#15 - 04/23/2014 07:46 PM - Mark Abraham

Alternatively, if we want to try to identify the underlying issue, then the best approach is likely to do a git bisect. That means building mdrun for a large series of versions between 4.0.7 and something known bad, and observing when the behaviour changes. But running tens of nanoseconds of a membrane to get a statistically significant converged area per lipid over around a dozen code versions does not sound like fun, and will take weeks. Is there a simpler observable that seems likely to show the problem?

#16 - 04/23/2014 09:50 PM - Patrick Fuchs

I understand that GROMACS devs are not keen in maintaining twin-range cutoff / RF scheme in new versions. My concern is just that a system that was working with versions <= 4.0.* no longer works in versions > 4.5.* and it was hard to understand why (at least to me when I first uncountered it). So I just wanted to make users know clearly not to use versions >= 4.5.* with this scheme. I guess it's too late to write it in the old manuals or to add a grompp warning in the old versions, so at least now within this redmine report #1400 it is documented.
As commented above, the two workarounds we found is to i) decrease nstlist to 1 (or 2), or ii) to increase the cutoff up to 1.4. According to your remark about the GPU era, it looks like the latter will be usable with reasonable computational cost.

About your questions:
1) For calculating the area per lipid, you just take: (Box-X * Box-Y) / nb_of_lipids_per_leaflet. nb_of_lipids_per_leaflet is 64 in the files posted above.

2) Unfortunately it's difficult to draw a clear conclusion before the area converges. However, an easy way to see if the twin-range cutoff / RF scheme works is to simulate a system with explicit hydrogens (e.g. a protein in water). As described by Miguel Machuqueiro (https://mailman-1.sys.kth.se/pipermail/gromacs.org_gmx-developers/2011-December/005489.html), it crashes right away with versions >= 4.5.3. Now that I know, on my side I stick to version 4.0.7 when I use this scheme.

#17 - 04/24/2014 07:48 PM - Mark Abraham

Patrick Fuchs wrote:

> I understand that GROMACS devs are not keen in maintaining twin-range cutoff / RF scheme in new versions. My concern is just that a system that was working with versions <= 4.0.* no longer works in versions > 4.5.* and it was hard to understand why (at least to me when I first uncountered it). So I just wanted to make users know clearly not to use versions >= 4.5.* with this scheme. I guess it's too late to write it in the old manuals or to add a grompp warning in the old versions, so at least now within this redmine report #1400 it is documented.

Indeed, thanks for your concern. Keeping everything working is a nightmare problem, even if you restrict the effort to common use cases.

> As commented above, the two workarounds we found is to i) decrease nstlist to 1 (or 2), or ii) to increase the cutoff up to 1.4. According to your remark about the GPU era, it looks like the latter will be usable with reasonable computational cost.

Neighbourlist duration should not affect the physics, which requires that an adequate buffer exists, which was hard to do with early versions of GROMACS. I don't think there is any reasonable way to do that for the twin-range regime, so support for that regime is going to disappear entirely along with the group cut-off scheme. I've no idea how well-tuned the RF (or GRF) GPU kernels are, but using them will pretty much guarantee wasting the CPU (because we have not prioritised doing any CPU-GPU load-balancing of short-ranged work, because the need for it with PME is fairly low). If GROMACS 5.0 PME+LJPME is faster than GRF at 1.4nm, then we'd probably consider that a win for the world ;-)

> About your questions:
> 1) For calculating the area per lipid, you just take: (Box-X * Box-Y) / nb_of_lipids_per_leaflet. nb_of_lipids_per_leaflet is 64 in the files posted above.

Thanks

> 2) Unfortunately it's difficult to draw a clear conclusion before the area converges. However, an easy way to see if the twin-range cutoff / RF scheme works is to simulate a system with explicit hydrogens (e.g. a protein in water). As described by Miguel Machuqueiro (https://mailman-1.sys.kth.se/pipermail/gromacs.org_gmx-developers/2011-December/005489.html), it crashes right away with versions >= 4.5.3. Now that I know, on my side I stick to version 4.0.7 when I use this scheme.

Good suggestion, after some pain getting set up, I can indeed observe nstlist=5 lead to LINCS issues and crashes once development proceeds sufficiently past 4.0.7. That might let me find the issue reasonably quickly, which could lead to a fix in 4.6.6 or 5.0.

#18 - 04/25/2014 11:25 AM - Mark Abraham

Bisection showed the issue is very likely in the Trotter-decomposition patch introduced after 4.0.7, 9487b9511. No great surprise there. The old code accumulated the force and the shift-force from the long-range contribution on each time step, which was irreversible and bad. The new code computed the long-range force and shift-force correctly, and saved a copy of the long-range force. Then during the update phase in combine_forces(), that long-range force is constrained (for reasons I don't understand), and then added to the rest of the forces, multiplied by nstlist-1 to achieve the reversible multiple time-step. But nothing is done to the shift force, so the virial is computed from a force and a shift force that are inconsistent. Even without constraints, a mismatch still occurs. This is consistent with the observations that either rlist=rlistlong or nstlist=1 lead to reasonable simulations.

I think the solution is to take the (usually constrained) long-range force, scale it by nstlist-1, and accumulate it to the shift force in combine_forces() in the normal way. Or maybe the shift forces need to be saved earlier and themselves constrained. Not sure. This should lead to computing a virial from things that are consistent.

#19 - 05/07/2014 08:48 AM - Berk Hess

You can't update electrostatic forces in water every 10 fs, since water rotates too fast. This twin-range setup was stable before, because the, very incorrect, integration scheme added artificial stabilization. With the Trotter decomposition some runs with this setup now even become unstable, as they should. If you really want to run with this setup, you should use a single-range 1.4 nm cut-off.

Mark, are you sure there is an issue with the shift forces? I thought this through well when implementing this Trotter scheme and also now I don't see why there should be an issue.

#20 - 05/07/2014 10:01 AM - Berk Hess

I now checked the old dppc benchmark system single-range vs twin-range with a time step of 0.2 fs and I do see some differences in the virial, pressure and surface tension. But I don't understand what can be wrong.

The Trotter scheme works like this at nstlist steps:
Calculate f_sr and f_lr
f1=f_sr+f_lr
calculate virial contribution of f1 and its shift forces
constrain f_lr to get f_lr_c
f2 = f1+(nstlist-1)*f_lr_c
update x,v with f2
constrain and get the constraint virial contribution, which should only count the f_lr part once, not nstlist times, since we constrained f_lr before adding the extra nstlist-1 to f2.

#21 - 05/07/2014 12:53 PM - Berk Hess

This issue is definitely not in the shift forces. For dppc with nstlist=10 I also get an error of 0.5% in the virial without pbc.

#22 - 05/07/2014 03:22 PM - Mark Abraham

OK. I attempted two solutions along the lines that I speculated, and certainly they did not work.

#23 - 05/07/2014 03:38 PM - Berk Hess

I see now that the difference in the virial comes from the difference in the constraint virial. Alternatively one could not constrain f_lr, but calculate it's virial contribution and subtract that nstlist-1 times. That gives exactly the same incorrect result. At least that's consistent. But that doesn't tell us where the problem is. I thought there might be some non-linearity in the constraining. But even a timestep of 0.02 fs with nstlist=10 gives the same deviation. It seems that there is some additional constraint contribution that appears when adding f_lr more than once.

#24 - 05/07/2014 04:53 PM - Berk Hess

I think I know what the issue is. It is actually the non-linearity in the constraints. When forces work perpendicular to constraints, they still give a constraint distribution, since they will lengthen the constraint. This constraint force does not depend on the time step (unless it's very large). So we should not constraint the long range force, but update x with the long range force and determine the constraint contribution of that, which we can then subtract. Alternatively we can do a single time step update to determine the constraint distribution and ignore the other results.

#25 - 05/07/2014 05:06 PM - Mark Abraham

Berk Hess wrote:

> I think I know what the issue is. It is actually the non-linearity in the constraints. When forces work perpendicular to constraints, they still give a constraint distribution, since they will lengthen the constraint. This constraint force does not depend on the time step (unless it's very large). So we should not constraint the long range force, but update x with the long range force and determine the constraint contribution of that, which we can then subtract.

Clarifying, you mean at nstlist steps, we temporarily update x with just the (unconstrained) long-range force, determine vir_lr_c (the constraint contribution to the virial of the long-range force), do the normal update of x from all the forces (per comment 20). Then I'm not sure what you mean we might subtract off. (nstlist-1)*vir_lr_c is subtracted from the virial?

Non-nstlist steps remain the way they are.

> Alternatively we can do a single time step update to determine the constraint distribution and ignore the other results.

#26 - 05/07/2014 05:45 PM - Berk Hess

non-nstlist steps remain as they are, since we don't calculate any virial there.
At nstlist steps we do either 1) or 2):

1)
dummy update with the long-range force only added once to determine the constraint virial
real update with the long-range force added nstlist times, ignore the constraint virial

2)

dummy update without v: xp = x + 0.5*dt^2*f_lr, constrain xp, determine the lr constraint virial
real update with the long-range force added nstlist times, correct the constraint virial using the term above

#27 - 05/08/2014 05:46 PM - Mark Abraham

I have a working version of 1) based on the Trotter patch, but it's a hack. The virial produced by single- and twin-range used to differ and is now identical (at double precision). I'll work on a version that functions on release-4-6, where the context for integrators and updates is... umm... different!

#28 - 05/12/2014 02:55 AM - Mark Abraham

I now have code on release-4-6 that gets the right constraint virial for twin-range with MD integrator.

Twin-range plus constraints has never been implemented for VV integrator, because the force-constraining was disabled, so I will introduce fatal errors for all of twin-range plus VV. It's probably straightforward to implement, but I don't know that anybody actually wants it and don't want to test all the range of possibilities.

However there's a subtle trap. The temporary velocities from the single-step update at nstlist steps are the ones we should be using for (e.g.) computing KE for temperature coupling. But vsites should be constructed from the multiple-step velocities. The special box update for MTTK looks like it wants full velocities, but we don't care about VV. And naturally, the vsite construction and box update happen between the coordinates update and the global communication for KE, so fancy footwork has to happen in more than one place.

The DPPC system exposed the issue with KE after a few nanoseconds; I have another run going with what I think is a fix.

Question for Berk: In forcerec_set_ranges() in forcerec.c, fr->f_twin seems like it is (re)alloced inside too many conditionals?

#29 - 05/13/2014 05:35 PM - Mark Abraham

Mark Abraham wrote:

> I now have code on release-4-6 that gets the right constraint virial for twin-range with MD integrator.
>
> Twin-range plus constraints has never been implemented for VV integrator, because the force-constraining was disabled, so I will introduce fatal errors for all of twin-range plus VV. It's probably straightforward to implement, but I don't know that anybody actually wants it and don't want to test all the range of possibilities.
>
> However there's a subtle trap. The temporary velocities from the single-step update at nstlist steps are the ones we should be using for (e.g.) computing KE for temperature coupling. But vsites should be constructed from the multiple-step velocities. The special box update for MTTK looks like it wants full velocities, but we don't care about VV. And naturally, the vsite construction and box update happen between the coordinates update and the global communication for KE, so fancy footwork has to happen in more than one place.

The observations I made that led to this realization were from DPPC at 2fs and nstlist=3. That setup with the KE fix above is still unstable - presumably the known issue with water libration. So far (6.5ns), 2fs and nstlist=2 is fine. So I will upload a fix.

> The DPPC system exposed the issue with KE after a few nanoseconds; I have another run going with what I think is a fix.
>
> Question for Berk: In forcerec_set_ranges() in forcerec.c, fr->f_twin seems like it is (re)alloced inside too many conditionals?

#30 - 05/13/2014 09:10 PM - Gerrit Code Review Bot

Gerrit received a related patchset '1' for Issue #1400.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: Ib61d0fb7661bca2101c04423a6af1744420c06ab
Gerrit URL: https://gerrit.gromacs.org/3438

#31 - 05/13/2014 09:10 PM - Gerrit Code Review Bot

Gerrit received a related patchset '1' for Issue #1400.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: I92957304cd8ef46963ebe8157efcf6d2e8994e32
Gerrit URL: https://gerrit.gromacs.org/3439

#32 - 05/13/2014 09:23 PM - Mark Abraham

Confirmed with the code fix in Gerrit that the global communication stage for KE must use the single-step velocities to reproduce the KE for single-range. All the energies now agree after one step of single- and twin-range (except the obvious partioning of the nonbonded components).

#33 - 05/14/2014 10:54 PM - Gerrit Code Review Bot

Gerrit received a related DRAFT patchset '1' for Issue #1400.
Uploader: Berk Hess (hess@kth.se)
Change-Id: If3217f52808bf7491998324f8dc3161bc003ec1b
Gerrit URL: https://gerrit.gromacs.org/3443

#34 - 05/31/2014 02:06 PM - Mark Abraham

- File grompp.mdp added

- Category set to mdrun

- Status changed from New to Fix uploaded

- Assignee set to Berk Hess

- Target version set to 4.6.6

I tested Berk's draft fix on a corrected version of the protocol used in the error report (nstlist=2 is the only stable option; old code had compensating errors that appeared to make nstlist=5 stable). That version produces average APL=0.619 nm^2 over 100ns with variation comparable with the 4.0.7 version. My .mdp is attached. I'll get Berk to publish it for review and then this issue will be resolved.

#35 - 06/05/2014 02:30 PM - Berk Hess

- Status changed from Fix uploaded to Resolved

- % Done changed from 0 to 100

Applied in changeset a75db633a2e363d48c66a440d2890592f4431d52.

#36 - 06/12/2014 01:50 AM - Erik Lindahl

- Status changed from Resolved to Closed

#37 - 03/26/2019 05:06 PM - Mark Abraham

- Related to Bug #2884: GROMACS can not reproduce properties with the GROMOS force fields added

Files

pb_RF_version453.tgz	305 KB	12/10/2013	Patrick Fuchs
md_gmx407.tpr	761 KB	12/10/2013	Patrick Fuchs
mdin.mdp	1.26 KB	12/10/2013	Patrick Fuchs
md_gmx453.tpr	761 KB	12/10/2013	Patrick Fuchs
area_RF.ps	43.9 KB	12/10/2013	Patrick Fuchs
area_RF+RFnec.ps	56.9 KB	12/20/2013	Patrick Fuchs
comparison.png	13.4 KB	12/21/2013	Lutz Maibaum
area_RF+rlist_eq_rcoulomb.pdf	32.6 KB	03/25/2014	Patrick Fuchs
grompp.mdp	1.37 KB	05/31/2014	Mark Abraham