# GROMACS - Bug #2849

## Free energy discrepancies between GROMACS versions

01/30/2019 11:17 AM - Miroslav Suruzhon

| | | | |
|---|---|---|---|
| **Status:** | Closed | | |
| **Priority:** | High | | |
| **Assignee:** | Berk Hess | | |
| **Category:** | mdrun | | |
| **Target version:** | 2019.1 | | |
| **Affected version - extra info:** | Potentially all versions since 2018.3 | **Difficulty:** | uncategorized |
| **Affected version:** | 2018.4 | | |

**Description**

Hello,

I was running some free energy perturbation simulations in GROMACS, and I had problems reproducing data from other people's simulations in GROMACS. After several days of trying to find out the issue, I realised that different versions of GROMACS give very different results in the .xvg file for the same input .tpr file and .trr trajectory (using gmx mdrun -rerun). I have attached the relevant .tpr and .trr files for the first 5 ps for two lambda values, alongside with the .gro, .top and .mdp files I used to create the .tpr. I have also attached the .xvg files generated for every stable version of GROMACS since 2018.1.

Ignoring the floating point errors, we can see that 2018.1, 2018.2 and 2018.3 give more or less the same results, which are very different from those produced by 2018.4 and 2018.5, resulting in extremely large discrepancies in free energies - more than 20 kJ/mol using gmx bar.  However, these discrepancies only seem to hold for the delta H values and not dH/dl. Finally, free energies calculated from 2019 using gmx bar seem to reproduce 2018.1 - 2018.3 but we can see that there is a missing column of data compared to other versions and most of the dH/dl derivatives do not agree with other versions of GROMACS.

Note that this issue only seems to persist when perturbing the Coulomb / bonded interactions. Van der Waals perturbation energies seem consistent between versions (not shown here).

Since the only variable for all of these simulations is the GROMACS version (which was compiled on the same computer (Ubuntu 18.04.1) with the same libraries (cmake 3.10.2, Open MPI 2.1.1, gcc 7.3.0), I highly suspect that there is (at least one) bug with the free energy implementation of the code (or the output thereof). However, I am not sure which version gives the "correct" results. Any feedback would be highly appreciated.

Thank you very much.

Kind regards,

Miroslav

**Related issues:**

| | |
|---|---|
| Related to GROMACS - Bug #2640: coul-lambda affects the pressure computation ... | **Closed** |
| Related to GROMACS - Bug #2703: constraint and mass contributions missing fro... | **Closed** |
| Related to GROMACS - Task #1868: implement mdrun -rerun better, simplifying d... | **Closed** |

### Associated revisions

**Revision 13f60c8a - 02/09/2019 02:17 PM - Berk Hess**

Disallow rerun with perturbed masses or constraints

Since rerun no longer integrates, free-energy contributions due to
changing masses and constraints are no longer computed. Now rerun
exits with a fatal error when such a system is provided.

Fixes #2849

Change-Id: Ic1550fce22f68c573b52d6895e1e61398b671e51

**Revision a741d07b - 02/15/2019 10:35 AM - Berk Hess**

Fix mass Delta H double counting

With separate mass lambda parameters, the perturbed mass contribution
to Delta H (not dH/dlambda) was double counted. This was due to an
incorrect fix for issue #2703 for missing Delta H contribution.

Refs #2703
Fixes #2849

Change-Id: I1349260e1e90a17a6b7dbe0d239d9474dcfa928c

**Revision 499ab90c - 02/20/2019 09:00 AM - Berk Hess**

Fix mass Delta H double counting

With separate mass lambda parameters, the perturbed mass contribution
to Delta H (not dH/dlambda) was double counted. This was due to an
incorrect fix for issue #2703 for missing Delta H contribution.

Refs #2703
Fixes #2849

Change-Id: I1349260e1e90a17a6b7dbe0d239d9474dcfa928c

## History

**#1 - 01/30/2019 12:08 PM - Mark Abraham**

*- Related to Bug #2640: coul-lambda affects the pressure computation of a ligand with zero partial charge added*

**#2 - 01/30/2019 12:09 PM - Mark Abraham**

*- Related to Bug #2703: constraint and mass contributions missing from foreign Hamiltonian values added*

**#3 - 01/30/2019 12:09 PM - Mark Abraham**

I referenced some potentially relevant bug fixes

**#4 - 02/07/2019 08:51 AM - Berk Hess**

*- Category set to mdrun*

*- Status changed from New to Accepted*

The missing constraint contribution to the foreign lambda Hamiltonian differences was added in 2018.4. So that should explain the difference between when switching from 2018.3 to 2018.4. But 2019 should be identical to 2018.4/5, so there seems to be something wrong there.

**#5 - 02/07/2019 11:19 AM - Berk Hess**

I think the issue is that is 2019 the -rerun option no longer does integration and constraining, which means that the constraint contributions to delta H and dH/dlambda are 0.
We should disable these outputs when constraints are perturbed.

**#6 - 02/07/2019 11:22 AM - Miroslav Suruzhon**

*- File 181L_stability.xlsx added*

Berk Hess wrote:

> The missing constraint contribution to the foreign lambda Hamiltonian differences was added in 2018.4. So that should explain the difference between when switching from 2018.3 to 2018.4. But 2019 should be identical to 2018.4/5, so there seems to be something wrong there.

I have been running more tests on the same system, and I think that the problem might be even deeper than that.

I ran three different combinations of 11 lambda values for 5 ns each with the following scenarios in both 2018.3 and 2018.4:
*unified protocol: 11 equidistant lambda values from 0 to 1 perturbing VdW, Coulomb and bonded interactions simultaneously with LJ and Coulomb softcore
*split protocol no softcore: 0.0, 0.2, 0.4, 0.6, 0.8, 1 for VdW and then perturbing Coulomb and bonded interactions in the same way simultaneously with LJ but without Coulomb softcore
*split protocol with softcore: same as before, but with Coulomb softcore

I then analysed the trajectories using an external tool (alchemical analysis) using several methods: TI, TI with cubic interpolation, EXP evaluated at ensemble A (backward EXP), EXP evaluated at ensemble B (forward exp), BAR and MBAR. Here I used the data generated from calc-lambda-neighbors=-1. I then took the trajectories generated from 2018.4 and fed them back into *both* 2018.3 and 2018.4 to manually generate potential energy output for EXP, BAR and MBAR.

Naturally, all of the above simulations should result in more or less the same energies. The worst case scenario should be that 2018.3 to gives inconsistent results but 2018.4 gives results that are fairly independent of the protocol used to generate the trajectories and the method of analysis.

I have attached the results as an Excel spreadsheet. We can make several important points:
*TI always gives consistent energies between GROMACS versions and free energy protocols (note that this does *not* apply to 2019, where I haven't performed these more sophisticated tests yet)
*2018.3 gives consistent energies for EXP, BAR, MBAR in the split protocol but different from TI
*Unified protocol for 2018.3 gives yet another result different to all of the above but with a similar consistency
*Output calculated with BAR from 2018.4 is consistent with TI
*However, all other methods give extremely different results between each other and between different protocols
*Note that all of these differences can be traced only to the lambda windows there Coulomb / bonds are perturbed and not van der Waals interactions
*When rerunning the trajectories from 2018.4 with both 2018.4 and 2018.3 this time we get very similar results from both 2018.4 and 2018.3
*However, these results are different from the XVG output with calc-lambda-state=-1. Again, this difference is only present when it comes to the Coulomb / bonded interactions.
*The BAR and MBAR results seem to mimic those from 2018.3, while the EXP results seem to be completely different to anything we have seen.

From these we can draw some tentative conclusions:
*The issue is probably related to the Coulombic / bonded forces / potential energies. Seeing the previous bug fix, it is more likely related to the PME implementation
*2018.4 gives output for dH in the XVG file which is inconsistent with the potential energy difference between the two lambda states generated from the same version (again only for Coulomb / bonded) - there probably is a bug here
*Since both 2018.3 and 2018.4 give similar outputs for potential energies when rerun from the same 2018.4 trajectories and these are largely similar to 2018.3 results, we can tentatively say that the sampling is probably consistent between versions, so the forces seem to be correct
*However, seeing that the unified approach still gives very different results from the split approaches, I would say that the potential energy might be missing some terms

**tl;dr: I think that there is an issue with both XVG output for BAR/MBAR (and possibly TI) and potential energy calculation with changing lambda values. However, these are probably only related to the PME implementation and not to other energetic components. The forces also seem to be correct in all cases**

#### #7 - 02/07/2019 12:11 PM - Berk Hess

So issue I am aware of:
<= 2018.3: mass and constraint contributions missing in Delta H (dH/dlambda is correct)
2018.4: none
2019: no mass and constraint contributions in both Delta H and dH/dlambda

Thus I would expect TI to give identical results for 2018.3 and 2018.4. Is that the case?

Also I would expect that BAR differences between 2018.3 and 2018.4 would only be present when masses or constraints are perturbed. Can you easily separate the charge perturbation from the mass and constraints perturbation?

#### #8 - 02/07/2019 12:16 PM - Miroslav Suruzhon

Berk Hess wrote:

> So issue I am aware of:
> <= 2018.3: mass and constraint contributions missing in Delta H (dH/dlambda is correct)
> 2018.4: none
> 2019: no mass and constraint contributions in both Delta H and dH/dlambda
>
> Thus I would expect TI to give identical results for 2018.3 and 2018.4. Is that the case?
>
> Also I would expect that BAR differences between 2018.3 and 2018.4 would only be present when masses or constraints are perturbed. Can you easily separate the charge perturbation from the mass and constraints perturbation?

Yes, TI is the same for 2018.3 and 2018.4 but different in 2019. When I ran a small test on 2019, TI agreed with all other methods in 2019. However, split and unified protocols gave very different results. This is the opposite situation to 2018.4 where TI and BAR always agree for both split and unified, but don't agree with MBAR and EXP.

Also, I am only perturbing charges, van der Waals and bonds with fep-lambdas set to whatever the default is. I therefore think that I am not changing any masses or constraints in any of these simulations, unless the constraints are a part of the bonded lambdas? In this case I can run the bonded lambdas separately.

#### #9 - 02/07/2019 12:39 PM - Berk Hess

The tprs you attached perturb both charges and masses. So my hypothesis is still that the Coulomb is correct.

I would also think 2019 is correct when computing delta H and dH/dl directly, not using rerun. Or did you find differences there with 2018.4?

#### #10 - 02/07/2019 12:52 PM - Miroslav Suruzhon

Berk Hess wrote:

> The tprs you attached perturb both charges and masses. So my hypothesis is still that the Coulomb is correct.
>
> I would also think 2019 is correct when computing delta H and dH/dl directly, not using rerun. Or did you find differences there with 2018.4?

Sorry, I got confused, I do perturb everything with fep-lambdas in the tprs I attached but on the tests I did afterwards (the Excel spreadsheet) I only perturbed bonds, Coulomb and LJ. I only tested 2019 with the tprs that I uploaded, so that's when I perturbed everything. 2019 without rerun seems to generate Delta H values like 2018.3 and TI values unlike any other versions. Therefore, 2019 and 2018.4 produce extremely different results. I will try to do the same tests with 2019 and this time split bonded and Coulomb interactions.

#### #11 - 02/07/2019 01:13 PM - Berk Hess

Your lambda 0 tpr gives me identical dhdl.xvg files with 2018.4 and 2019, as I would expect.
With rerun 2018.4 is the same as without, for 2019 it is different, as I expected.

So all evidence I have seen is consistent with what I know about the code. 2018.4: always correct, 2019.4 mdrun correct, rerun incorrect.

#### #12 - 02/07/2019 03:53 PM - Miroslav Suruzhon

Berk Hess wrote:

> Your lambda 0 tpr gives me identical dhdl.xvg files with 2018.4 and 2019, as I would expect.
> With rerun 2018.4 is the same as without, for 2019 it is different, as I expected.
>
> So all evidence I have seen is consistent with what I know about the code. 2018.4: always correct, 2019.4 mdrun correct, rerun incorrect.

OK, so we agree on the part which is 2018.4 fixed some behaviours from 2018.3 and 2019 is the same as 2018.4 except that the rerun is broken. I also agree that regenerating the XVG using rerun for < 2019 produces the same results as the regular mdrun.

However, in the rerun I reported, I output potential energies (with both 2018.3 and 2018.4) instead of Delta H from 2018.4's trajectories. From these potential energies I created input for pymbar's (M)BAR which gives me results close to the ones reported in 2018.3 (using the regular dhdl.xvg output) and not 2018.4 (as shown in the Excel spreadsheet). This is true no matter which version of GROMACS we output these potential energies with. Again, the differences could only be attributed to the non-van der Waals legs (so bonded / Coulomb). Alternatively, one can also use gmx energy to calculate the EXP free energies from the potential energies and they only agree with BAR for the van der Waals parts. I only perturb Coulomb, bonded and van der Waals in these simulations. That's why I suspected the potential energies don't give correct values (because they don't agree with Delta H for Coulomb / bonded).

Another issue with 2018.4 is that MBAR produces very different results from BAR, which shouldn't be the case.

Right now I have split Coulomb and bonded interactions for both 2018.4 so that we can determine for sure where the discrepancy lies (EXP, BAR and MBAR should give the same results for 2018.4 if it is correct and potential energies regenerated from 2018.4's mdrun rerun should yet again give the same results, which is not the case in Coulomb / bonded). I will keep you posted.

#### #13 - 02/07/2019 05:49 PM - Mark Abraham

*- Assignee set to Mark Abraham*

*- Target version set to 2019.1*

At the very least, we will stop 2019 rerun writing wrong numbers. Probably by blocking such tpr files, because I don't think restoring the 2018.4 rerun behaviour is feasible.

#### #14 - 02/07/2019 05:53 PM - Mark Abraham

*- Related to Task #1868: implement mdrun -rerun better, simplifying do_md added*

#### #15 - 02/07/2019 08:27 PM - Berk Hess

I don't understand what you mean with "potential energies". The Coulomb+bonded leg has both mass and constraint perturbation terms which are not part of the potential energy. I could be that if you consistently ignore those terms, you get consistent results. But they will be consistently incorrect.

#### #16 - 02/08/2019 11:00 AM - Gerrit Code Review Bot

Gerrit received a related patchset '1' for Issue #2849.
Uploader: Berk Hess (hess@kth.se)
Change-Id: gromacs~release-2019~Ic1550fce22f68c573b52d6895e1e61398b671e51
Gerrit URL: https://gerrit.gromacs.org/9113

#### #17 - 02/08/2019 11:07 AM - Berk Hess

*- Subject changed from XVG Energy Discrepancies Between GROMACS Versions to Free energy discrepancies between GROMACS versions*

*- Status changed from Accepted to Fix uploaded*

*- Priority changed from Normal to High*

I have uploaded a change disabling rerun with perturbed masses or constraints.

But we should be sure there are no other issues with 2018.4/5 and 2019.

**#18 - 02/08/2019 11:20 AM - Miroslav Suruzhon**

Berk Hess wrote:

> I don't understand what you mean with "potential energies". The Coulomb+bonded leg has both mass and constraint perturbation terms which are not part of the potential energy. I could be that if you consistently ignore those terms, you get consistent results. But they will be consistently incorrect.

Thanks for the bug fix for rerun!

I didn't know that the constraint contributions were implemented outside of the potential energy. That would explain the rerun discrepancy. Does that mean that if I want to regenerate data for MBAR post-run I need to output total energy instead?

I split the bonded and Coulomb contributions and TI / EXP / BAR / MBAR agree for all of the Coulomb contributions (using the regular XVG output in 2018.4), which means that it is the bonds that create the large hysteresis between forward and backward EXP. Oddly enough, this happens whether or not a constraint algorithm is used (I am not perturbing masses either), although the discrepancy is markedly lower if there are no constraints. I suspect (and hope) it is a poor convergence issue rather than a bug so I will increase the lambda window resolution.

I will report back when I get some more results.

**#19 - 02/08/2019 12:11 PM - Berk Hess**

The constraint contribution can not be obtained from the potential, there is a kinetic contribution.

I don't understand what you mean with "no constraint algorithm". Or have you tried with all bonds flexible?

**#20 - 02/08/2019 12:20 PM - Miroslav Suruzhon**

Berk Hess wrote:

> The constraint contribution can not be obtained from the potential, there is a kinetic contribution.
>
> I don't understand what you mean with "no constraint algorithm". Or have you tried with all bonds flexible?

Okay, that makes sense. The total energy should then contain all terms we would want for a correct free energy calculation post-run then, is that correct (e.g. with gmx energy -fee)?

Yes, one of the scenarios I am trying is with all bonds flexible and a 1 fs timestep instead of 2 fs (I have set constraints to none in the mdp file).

**#21 - 02/08/2019 05:34 PM - Berk Hess**

> Okay, that makes sense. The total energy should then contain all terms we would want for a correct free energy calculation post-run then, is that correct (e.g. with gmx energy -fee)?

For mass perturbation you would need to compute the kinetic energy difference/derivative, but in 2019 that can not be done with rerun. The constraint contribution can never be obtained from computing energies. You need to compute dH/dlambda with includes momentum contributions. Delta H for constraints is approximated at Delta lambda * dH/dlambda.

**#22 - 02/09/2019 02:30 PM - Berk Hess**

*- Status changed from Fix uploaded to Resolved*

Applied in changeset 13f60c8aa0ba2fd60fe2f2e57b480ec93a9a0539.

**#23 - 02/11/2019 09:54 AM - Mark Abraham**

*- Status changed from Resolved to Closed*

*- Assignee changed from Mark Abraham to Berk Hess*

**#24 - 02/11/2019 11:24 AM - Miroslav Suruzhon**

*- File Bonded.xlsx added*

Berk Hess wrote:

> > Okay, that makes sense. The total energy should then contain all terms we would want for a correct free energy calculation post-run then, is that correct (e.g. with gmx energy -fee)?
>
> For mass perturbation you would need to compute the kinetic energy difference/derivative, but in 2019 that can not be done with rerun. The constraint contribution can never be obtained from computing energies. You need to compute dH/dlambda with includes momentum contributions. Delta H for constraints is approximated at Delta lambda * dH/dlambda.

That was very informative and helpful, thanks for the information.

Back to the topic of constraints, I have now run simulations which are split into two groups: constrained (with LINCS) and unconstrained (with 1 fs timestep), and then split into another three groups: perturb bonded lambdas only (set fep lambdas explicitly to 0), perturb bonded and fep lambdas together, perturb bonded and mass lambdas together. In all of the simulations the VdW and the Coulomb lambdas have been explicitly set to be constant in all 10 lambda windows (so I only have bonded, constraint and mass contributions). Since I am not changing any temperatures or restraints in my simulations, the ones where fep lambdas are perturbed and mass lambdas are perturbed should give exactly the same results as each other.

Here are the results:
*All the free energy methods agree in the simplest case of no constraints and only bonded lambdas perturbed, as they should - this baseline case is correct and issue-free
*Simulations where fep and mass lambdas are perturbed only agree in terms of TI. The other methods differ both among each other and between simulations - this is clearly erroneous behaviour for the Delta H values when mass is perturbed. If you take a look at the actual numbers, I think that the issue might be double-counting of the mass contribution when mass lambdas are perturbed.
*Constrained simulations where only the bonded lambdas are perturbed give results where TI, BAR and MBAR agree but forward and backward EXP give very different values, which is again erroneous behaviour. Interestingly enough, the average of these values, no matter how far apart they might be, gives back a value in agreement with BAR. Maybe this could be related to the one-sided approximation of Delta lambda * dH/dlambda and a symmetrised approximation should be applied instead (e.g. Delta lambda * ((dH/dlambda)0 + (dH/dlambda)1) / 2 ? I am not sure about this one, since the discrepancies seem to be far bigger than this correction.

So in conclusion, I think that the constraint and mass lambda correction implementation is still not issue-free. I think you will agree with me that different methods of calculating free energies should largely agree with each other (and they do in the baseline case) and not exhibit such behaviour. All data was taken directly from the resulting XVG files with 2018.4 and no rerun was performed. I have attached an Excel spreadsheet which visualises the data better.

Many thanks.

**#25 - 02/11/2019 02:44 PM - Berk Hess**

The exponential is a crude approximation, so I'm not surprised that is far off.
But the (M)BAR results deviating from the other methods seem to indicate a bug. With which version is this? As I said, there was a big exactly with this in 2018.3 and before.

**#26 - 02/11/2019 02:58 PM - Miroslav Suruzhon**

Berk Hess wrote:

> The exponential is a crude approximation, so I'm not surprised that is far off.
> But the (M)BAR results deviating from the other methods seem to indicate a bug. With which version is this? As I said, there was a big exactly with this in 2018.3 and before.

Yes, it is probably the crudest and that's why I ran finely spaced lambda values, but I still get this huge hysteresis. And note that this doesn't happen in the case of no constraints / no mass perturbations (in this case the agreement is very good), so I don't think this is a fault of the method.

I have used 2018.4 and the discrepancy between (M)BAR and TI seems to be only triggered by explicit setting of mass-lambdas, instead of implicit using fep-lambdas. That's why I think there is some double-counting (or half counting!) of this correction. The reason is that the difference between TI of unperturbed mass and TI of perturbed mass seems to be the same as the difference of TI of perturbed mass and BAR of perturbed mass.

**#27 - 02/12/2019 09:44 AM - Berk Hess**

But how do you compute the exponential free-energy difference? gmx energy can not compute the mass and constraint terms.

I just checked a trivial case of changing the masses in an LJ liquid with 2018.5 and TI and BAR give me the same answer.

**#28 - 02/12/2019 10:27 AM - Miroslav Suruzhon**

Berk Hess wrote:

> But how do you compute the exponential free-energy difference? gmx energy can not compute the mass and constraint terms.
>
> I just checked a trivial case of changing the masses in an LJ liquid with 2018.5 and TI and BAR give me the same answer.

I use an external script called alchemical analysis and it uses the Delta H values from the original XVG file to compute the EXP difference. So the mass / constraint contributions should be there.

It is strange that you don't get the discrepancy I am getting. Maybe it is triggered by me setting the bonded-lambdas as well? I will try running mass lambdas only also. Could you upload your MDP for that run? I will try to see where the differences are.

Many thanks

**#29 - 02/12/2019 11:39 AM - Berk Hess**

I see the exponential issue now. The mass and constraint terms do not affect the Boltzmann distribution. If you want to do exponential averages (which you don't), you should separate the mass and constraint terms and add them without exponential averaging.

I manually change the mass of my LJ atom in my single atom molecule and my mdp file uses:

```
free-energy          = yes
fep-lambdas           = 0 1
init-lambda-state      = 0
delta-lambda          = 0
```

**#30 - 02/12/2019 11:46 AM - Miroslav Suruzhon**

Berk Hess wrote:

> I see the exponential issue now. The mass and constraint terms do not affect the Boltzmann distribution. If you want to do exponential averages (which you don't), you should separate the mass and constraint terms and add them without exponential averaging.
>
> I manually change the mass of my LJ atom in my single atom molecule and my mdp file uses:
>
> ```
> free-energy          = yes
> fep-lambdas           = 0 1
> init-lambda-state      = 0
> delta-lambda          = 0
> ```

Ah, that would explain this EXP hysteresis, I didn't think of that!

As for the mdp file - setting the masses using fep-lambdas doesn't result in the discrepancy but setting mass-lambdas explicitly does for some reason (that may be why your BAR and TI agree). That's why I thought there was a small bug related to the implementation of mass-lambdas. So if you put fep-lambdas to 0 0 and mass-lambdas to 0 1 you might see it.

**#31 - 02/13/2019 10:02 AM - Berk Hess**

*- Status changed from Closed to Fix uploaded*

I found the mass Delta H double counting issue and uploaded a fix to 2019. We should backport this to 2018.

**#32 - 02/13/2019 10:12 AM - Miroslav Suruzhon**

Berk Hess wrote:

> I found the mass Delta H double counting issue and uploaded a fix to 2019. We should backport this to 2018.

Many thanks for that! I think we've explained everything I observed now.

**#33 - 02/13/2019 10:47 AM - Berk Hess**

Thanks for reporting this and your extensive analysis efforts!

**#34 - 02/15/2019 10:45 AM - Berk Hess**

*- Status changed from Fix uploaded to Resolved*

Applied in changeset a741d07b604e2952a0113203faccd310271dc233.

**#35 - 02/15/2019 12:03 PM - Paul Bauer**

*- Status changed from Resolved to Closed*

## Files

| | | | |
|---|---|---|---|
| Input.tar.xz | 2.91 MB | 01/30/2019 | Miroslav Suruzhon |
| 181L_stability.xlsx | 11.7 KB | 02/07/2019 | Miroslav Suruzhon |
| Bonded.xlsx | 9.97 KB | 02/11/2019 | Miroslav Suruzhon |