**GROMACS - Bug #1232**

**Incorrect calculation of TI kinetic energy contribution -- no zeroing**

04/28/2013 02:12 AM - Michael Shirts

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>High</td>
</tr>
<tr>
<td>Assignee:</td>
<td>Michael Shirts</td>
</tr>
<tr>
<td>Category:</td>
<td>mdrun</td>
</tr>
<tr>
<td>Target version:</td>
<td>4.6.2</td>
</tr>
<tr>
<td>Affected version - extra info:</td>
<td>4.6 and 4.6.1</td>
</tr>
<tr>
<td>Affected version:</td>
<td>4.6</td>
</tr>
<tr>
<td>Difficulty:</td>
<td>uncategorized</td>
</tr>
</tbody>
</table>

**Description**

There is an incorrect accumulation of the thermodynamic integration kinetic energy contribution. This appears to be because the accumulator term:

```c
dekindl_sum = &ekind->ekin_work[thread][opts->ngtc][0][0];
```

is never zeroed. It does not show up in some simulations because of another error in handling F_DVDL and F_DKDL that in some situations causes F_DKDL not to be printed.

I am getting a patch ready to post in the next 30 min or so.

**Associated revisions**

**Revision b8440aeb - 04/30/2013 07:40 PM - Michael Shirts**

Fixing handling of perturbation mass changes.

Fixes redmine #1232

in force.c, sum_dhdl

- moved F_DKDL to match the order in ept_names. Not required, but harmonizes the code (lack of clarity probably helped cause the problems before), has no code effect.

- no longer treating the F_DKDL term separately from the other derivative components. Will be added to F_DVDL if the mass-lambda term is not separately specified. Results in a bit of a misnomer (F_DVDL becomes the derivative of the entire hamiltonian), but makes it much easier to collapse all molecular perturbation terms into a single component for output, where it is no longer really F_DVDL. I think that's better than always printing out a F_DVDL and a F_DKDL for everything where F_DKDL will probably usually be zero.

in md_support.c, compute_globals

- Synchronize the behaviors of the dhdls by writing first to the linear component corresponding to the mass, and then later transferring it to F_DKDL

in group.h, struct gmx_ekindata_t * add pointer to per-thread accumulation variable for dekindl

in tgroup.c, sum_ekin

- For velocity verlet integrators, computes the dekindl correctly as the derivatives of the current ekin. Shouldn't really affect the results in any significant way, since the average contribution will be the same regardless, but this is more consistent.

in tgroup.c, init_ekindata

- reduce use of numeric constants in allocating memory

- initialize new ekindata_t member

03/20/2020
in update.c, calc_ke_part_normal

- zero the accumulator for dekindl before using it, fixing bug introduced in 7b6508e8

in update.c, in calc_ke_part_normal and calc_ke_part_visc

- sign error in mass change; if mass B is greater than mass A, then the change in free energy is positive, not negative.

Change-Id: l9deaf546bca66d400e0eb2c4015a6eda302dd1d

Revision 3e918e6d - 06/13/2013 09:45 AM - Michael Shirts

Fixing handling of perturbation mass changes.

Fixes redmine #1232

in force.c, sum_dhdl

- moved F_DKDL to match the order in efpt_names. Not required, but harmonizes the code (lack of clarity probably helped cause the problems before), has no code effect.

- no longer treating the F_DKDL term separately from the other derivative components. Will be added to F_DVDL if the mass-lambda term is not separately specified. Results in a bit of a misnomer (F_DVDL becomes the derivative of the entire hamiltonian), but makes it much easier to collapse all molecular perturbation terms into a single component for output, where it is no longer really F_DVDL. I think that's better than always printing out a F_DVDL and a F_DKDL for everything where F_DKDL will probably usually be zero.

in md_support.c, computeGlobals

- Synchronize the behaviors of the dhdls by writing first to the linear component corresponding to the mass, and then later transferring it to F_DKDL

in group.h, struct gmx_ekindata_t * add pointer to per-thread accumulation variable for dekindl

in tgroup.c, sum_ekin

- For velocity verlet integrators, computes the dekindl correctly as the derivatives of the current ekin. Shouldn't really affect the results in any significant way, since the average contribution will be the same regardless, but this is more consistent.

in tgroup.c, init_ekindata

- reduce use of numeric constants in allocating memory

- initialize new ekindata_t member

in update.c, calc_ke_part_normal

- zero the accumulator for dekindl before using it, fixing bug introduced in 7b6508e8

in update.c, in calc_ke_part_normal and calc_ke_part_visc

- sign error in mass change; if mass B is greater than mass A, then the change in free energy is positive, not negative.

Change-Id: l9deaf546bca66d400e0eb2c4015a6eda302dd1d

History

#1 - 04/28/2013 02:59 AM - Michael Shirts

- Assignee changed from Berk Hess to Michael Shirts

#2 - 04/28/2013 02:59 AM - Michael Shirts

- Target version set to 4.6.2
#3 - 04/28/2013 03:01 AM - Michael Shirts
- % Done changed from 50 to 90
- Affected version - extra info set to 4.6 and 4.6.1

OK, posted a fix.

#4 - 04/29/2013 04:03 PM - Mark Abraham
- Status changed from New to Accepted

#5 - 04/29/2013 04:03 PM - Mark Abraham
- Status changed from Accepted to In Progress

Indeed. Bug was introduced in commit 7b6508e8 (the big squashed Verlet commit).

#6 - 04/29/2013 04:03 PM - Mark Abraham
- Status changed from In Progress to Fix uploaded

#7 - 05/22/2013 04:51 AM - Mark Abraham
- Status changed from Fix uploaded to Resolved

#8 - 05/31/2013 10:45 AM - Mark Abraham
- Status changed from Resolved to Closed