**Gromacs - Bug #1297**

infinite relative permittivity is incompletely implemented

07/02/2013 10:50 AM - Mark Abraham

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>Mark Abraham</td>
</tr>
<tr>
<td>Category:</td>
<td>mdrun</td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
</tr>
<tr>
<td>Affected version - extra info:</td>
<td>Affected version: 4.6.2</td>
</tr>
</tbody>
</table>

**Description**

0 ir->epsilon_r[f] is supposed to signal infinite relative permittivity, i.e. fully conducting, so no electrostatic potential.

As partly reported on gmx-developers today by Mark Tianwu Zang:

- ewald_LRcorrection, ewald_charge_correction, do_ewald and calc_verlet_buffer_size use ir->epsilon_r without checking for zero
- solve_pme_yzx does likewise (indirectly)
- generalized Born code does likewise (genborn.c, and all the way into the group kernels)

So various code will divide by zero.

I think the PME, GB and ic data structures should store a multiplicative constant (as fr and ic try to do), that their init routines should all apply the same logic when using inputrec->epsilon_r[f], and that only inputrec should have members epsilon_r[f].

Because this goes everywhere, it's probably easiest to do in master branch with some tests that (0 epsilon_r[f]) => (no crash and no Coulomb) for all coulombtype and GB. I have some as-yet unpublished toys that will support doing that. Then we cherry-pick the result back.

**Associated revisions**

Revision af3e1a13 - 06/20/2015 08:09 PM - Erik Lindahl

Disallow infinite epsilon with PME or GB.

This leads to divide-by-zero errors, and since it just turns off electrostatics it will be faster to use a plain cutoff, which we now recommend instead.

Fixes #1297.

Change-Id: la5eba7a1d4bd3b387174c9cb4878a2d65ead2c6c

**History**

**#1 - 07/02/2013 01:39 PM - Mark Abraham**

Of course, it doesn't make much sense to trigger these code paths with 0 == epsilon_r[f] - all you do is waste time. tpbconv -zeroq supports doing a rerun that happens to have no/few electrostatic interactions, because the searching notices atoms have no charge and the kernels become an empty loop.

We can probably extend coulombtype to include None in 5.0 - a null electrostatics model might be useful for testing code, or with rerun.

The question becomes "why do we support infinite relative permittivity?"
Infinite relative permittivity is the best way of using RF and related methods. Even thought RF is not great we should support it for historical reasons.

Infinite epsilon_rf is fully and properly supported. Infinite epsilon_r simply completely turns off all electrostatic interactions. I don't see any need to support this with PME, GB, etc. grompp should either give a fatal_error and suggest to switch to plain cut-off with epsilon_r=0 for optimal performance.

Gerrit received a related patchset '1' for Issue #1297. 
Uploader: Erik Lindahl (erik.lindahl@gmail.com)
Change-Id: Ia5eba7a1d4bd3b387174c9cb4878a2d65ead2c6c
Gerrit URL: https://gerrit.gromacs.org/4747

- Status changed from New to Fix uploaded

- Status changed from Fix uploaded to Closed