GROMACS - Bug #774

flag -mol in g_msd

07/07/2011 03:16 PM - Florian Dommert

Status:	Closed		
Priority:	Normal		
Assignee:			
Category:			
Target version:			
Affected version - extra info:		**Difficulty:**	uncategorized
Affected version:	4.5.1		

Description

The tool g_msd gives inconsistent results with the flag -mol. If the diffusion coefficient is calculated for a single molecule using a corresponding index file, the result differs from an analysis for the same molecule, but using the
flag -mol.

Associated revisions

Revision e030b3a4 - 11/10/2012 10:06 PM - Roland Schulz

Fix g_msd -mol

Fixes #774

Change-Id: I6fa564754bcbc898a5da6428729775c669cd72d2

History

#1 - 07/27/2011 12:14 AM - Simon Butler

- File gmx_msd.mol.patch added

- File gmx_msd.mem.patch added

I've just posted this to gmx-users but I guess it should probably go here too. My tests suggest these patches resolve both problems with the -mol flag.

Regarding the excessive memory usage of g_msd when using the mol flag, as reported recently, I believe I've identified the source of the problem: the array of per-molecule MSD data in curr->lsq. In particular, it appears that the call to gmx_stats_add_point on line 476 is the key offender.

I haven't the time at the moment to dig any further than this, unfortunately, but the attached patch (gmx_msd.mem.patch) may be of temporary use to users of g_msd. It merely comments out the call to gmx_stats_init and the printmol routine. If anybody wishes to retain the calculation of D for each individual molecule, they will need to recode the data accumulation in the style of that for the overall calculation I expect.

There is also the problem of the large difference in the calculated results when using the -mol flag, as reported by Florian. I think the cause of this is an error in the logic within the corr_loop routine. The original sequence is:

1. If first iteration, copy current frame to previous
2. If -mol, make molecules whole
3. Remove PBC jumps
4. If -mol, calculate molecule COMs and copy to xa array

I believe the correct order for this sequence should be 2, 4, 1, 3. The original order results in xa[prev] containing zero for every position on step 1 and also in the PBC step operating on an array that hasn't been repopulated yet, with the result that jumps are not removed correctly.

I've tested the new sequence for a single PF6 molecule and for the corresponding P atom (which is almost exactly at the COM) and obtained very nearly identical results for each (which wasn't the case previously). I've attached a separate patch to reorder these function calls (gmx_msd.mol.patch). Both patches are for the v4.5.4 copy of gmx_msd.c, by the way.

#2 - 10/17/2012 10:50 PM - Roland Schulz

Could you please upload these patches to gerrit.gromacs.org?

#3 - 11/10/2012 10:10 PM - Roland Schulz

- Status changed from New to In Progress

The memory issue is unrelated and you should open a separate issue for it. I uploaded the mol patch to https://gerrit.gromacs.org/#/c/1639/ .

#4 - 02/24/2014 11:14 AM - Rossen Apostolov

- Status changed from In Progress to Closed

- Affected version set to 4.5.1

It's been fixed by Roland's patch.

Files

gmx_msd.mol.patch	592 Bytes	07/26/2011	Simon Butler
gmx_msd.mem.patch	460 Bytes	07/26/2011	Simon Butler