Gromacs - Feature #1346
COM pulling increased flexibility and clarification
09/30/2013 12:42 PM - Berk Hess

Status: Closed
Priority: Normal
Assignee: Berk Hess
Category: mdrun
Target version: 5.x

Description
The pull code in Gromacs, although very widely used, has always been somewhat limited in features. Also the mdp options can be confusing.
For version 5.0 I want to, at the least, remove the restriction of a single reference group and separate the group definitions from the pull coordinate definitions. Please have a look at the example for the suggested setup below.

The pull code can, of course, be generalized in infinitely many ways. But that comes at the cost of more (bug-prone) code and a more complex interface. Here are some possibilities for generalization we could consider, if a significant fraction of the users would be interested:
1) Allowing a mix of constraint and potential pulling (requires some work to get working efficiently, probably requires two global communication steps).
2) Add a flat-bottomed potential (this I will probably add: little work and only one extra parameter)
3) Allow multiple distances (the average of N distances) in a single pull coordinate (could be specified through multiple group pair on the pull_coord1_group line).
4) Putting the free-energy B-state parameter for k (and now also for init) on the same line (no extra kB mdp option), see redmine #1303
5) Anything more you can come up with (that might seem useful to more users)?

mdp setup in 4.6:

```md
pull                     = constraint
pull_geometry            = direction
pull_dim                 = N N Y
pull_ngroups             = 1
pull_group0              = GR1
pull_group1              = GR2
pull_vec1                = 0 0 1
pull_init1               = 0.5
```

Suggested mdp setup for 5.0:

```md
pull                     = constraint

pull_ngroups             = 2
pull_group1              = GR1
pull_group1_weights      = ...
pull_group1_pbcatom      = ...
pull_group2              = GR2

pull_ncoords             = 1
pull_coord1_geometry     = direction
pull_coord1_dim          = N N Y
pull_coord1_groups       = GR1 GR2
pull_coord1_vector       = 0 0 1
pull_coord1_init         = 0.5
```

Associated revisions
Revision cc819af8 - 12/03/2013 09:13 AM - Berk Hess

Removed restriction of one reference pull group

The pull code now uses separate pull groups and coordinates, which removes any restriction on group pairing.
The pull geometry "position" has been removed, but the origin option replaces this functionality in case of an absolute reference. Removed triple use of the init option. Refs #1346

Change-Id: i90736ac6b6f18e3bdf26a344072172a69627566e9

---

**History**

**#1 - 09/30/2013 03:31 PM - Erik Marklund**

I've implemented the pulling along a helical coordinate, initially intended for examining processes on DNA helices. It's a generalisation of angular pulling and can do things that can't really be accomplished with the geometries that are currently implemented. I'd be happy to adapt this to 5.0.

**#2 - 09/30/2013 03:33 PM - Erik Marklund**

I forgot to add that most of that code is separate from the actual pulling. Most of it is geometrical analysis of how the COM are positioned with respect to each other and the calculation of a pulling force. Hence it is not so intrusive into the actual pull code.

**#3 - 09/30/2013 04:27 PM - Berk Hess**

But is this helical pulling working on top of angular pull code, or the normal pull code? The changes I am proposing are only for the normal pull code, which is (currently) completely separated from the angular pull code.

**#4 - 09/30/2013 04:28 PM - Bogdan Costescu**

I'd like to suggest ending the run when the distance between groups becomes lower or higher than a given value. This should eliminate CPU time uselessly burned in studies involving two molecules (or parts of molecule) being pulled together or apart, when only distances within a certain range are relevant. The distance between the group and reference is computed anyway, except for constant force, so the code additions should be very simple. In my implementation for 4.5.x, the biggest part of the code is involved in reading the values from mdp and performing a graceful exit - I chose to save the current distances in the pull struct and test against the given values in the do_md() loop.

**#5 - 10/01/2013 10:29 AM - Erik Marklund**

The helical pulling was based on the ordinary pull code.

**#6 - 10/02/2013 03:13 PM - Jochen Hub**

We would be happy if multiple reference groups would be possible. Such that for each pull group, an mdp options specifies the reference group for each pull group. E.g:

```plaintext
; First, the reference groups
pull_nrefgrp   = 2
pull_refgroup0 = ProteinSubunitA
pull_refgroup1 = ProteinSubunitB

; Then the actual pull groups
pull_group0   = molecule1
pull_refgrp0  = 1
pull_group1   = molecule2
pull_refgrp1  = 2
pull_group2   = molecule3
pull_refgrp2  = 1
...
```

11/25/2015
In case it is helpful, here is a diff of the standard 4.0.5 source code and a version that was modified for us by Pawel Pomorski (Working for SHARCNET at the University of Waterloo, Canada)

The attached file was created by:

```bash
for i in $(find gromacs-4.0.5_pawel/ -name "*.ch"); do j=$(echo $i | sed "s/_pawel//"); n=$(diff $j $i|wc -l); if((n)); then diff -u $j $i; fi; done > diff_4.0.5.multiPull
```

The usage for that modified version is:

```bash
:::PULL STUFF
num_pull_loop=10
pull=umbrella
pull_geometry = distance
pull_dim = Y Y Y
pull_nstxout = 50
pull_0nstfout = 50
pull_0ngroups = 1
pull_0group0 = a_200
pull_0pbcatom0 = 0
pull_0group1 = a_1000
pull_0pbcatom1 = 0
pull_0init1 = 0.828897
pull_0k1 = 100
pull_0kB1 = 100

pull_1nstfout = 50
pull_1ngroups = 1
pull_1group0 = a_300
pull_1pbcatom0 = 0
pull_1group1 = a_1100
pull_1pbcatom1 = 0
pull_1init1 = 0.794814
pull_1k1 = 100
pull_1kB1 = 100

pull_2nstfout = 50
pull_2ngroups = 1
pull_2group0 = a_945
pull_2pbcatom0 = 0
pull_2group1 = a_4640
pull_2pbcatom1 = 0
pull_2init1 = 0.836636
pull_2k1 = 100
pull_2kB1 = 100
...
```

The limitations are that you can not have different pull, pull_geometry, pull_dim, or pull_nstxout values for the different groups.
The two suggested/modified mdp layouts seem to do exactly what I propose, but with more mdp options and restrictions on them. Since there doesn't seem to be a big need for certain extensions, I'll go with my suggestion for now.

Note that many users will want many different additional pull functionalities (such as terminating a simulation as suggested above), but we want to keep to code a simple and manageable as possible, since such extensions will be very bug prone and somewhat hard to test properly.

I noticed the group selection for pull coordinates in my suggestion was incorrect, here is a corrected version:

```python
pull = constraint

pull_nstxout = ...
pull_nstfout = ...

pull_ngroups = 2
pull_group1 = GR1
pull_group1_weights = ...
pull_group1_pbcatom = ...
pull_group2 = GR2

pull_ncoords = 1
pull_coord1_geometry = direction
pull_coord1_dim = NN
pull_coord1_groups = 1 2
pull_coord1_vector = 0 0 1
pull_coord1_init = 0.5
```

---

#10 - 05/22/2014 09:55 AM - Erik Lindahl
- Target version changed from 5.0 to 5.x

#11 - 08/25/2015 09:34 PM - Berk Hess
- Status changed from New to Closed

In version 5.1 the pull code setup is now a lot more flexible and there are more pull geometries. The next release will add angle and dihedral pulling. So we can close this feature; a new one can be opened if someone wants even more features.

Files

diff_4.0.5.multiPull.gz 7.3 KB 10/05/2013 Chris Neale