GROMACS - Task #2569
announce deprecations in GROMACS 2019
07/11/2018 10:52 PM - Mark Abraham

Status: Closed
Priority: Normal
Assignee: Mark Abraham
Category: core library
Target version: 2019
Difficulty: uncategorized

Description
Suggestions for functionality to announce as deprecated, either to remove or replace at some later time.

Several of these are intended to clarify what combinations of things are actually supported in mdrun, simplifying its documentation challenge. Many things have custom input and output files, and we are a long way from all-works-with-all support. Many of the current decisions date from the time when we did not want to increase our install footprint of 80+ tools.

Current list:

- aromatic vsite construction in pdb2gmx
- gmx mdrun -rerun (replace with gmx rerun and gmx tpi; see also #1868)
- gmx mdrun -membed (replace with gmx membed; we had the old g_membed, whose implementation we fixed, but the distinct tool is useful)
- non-integrators in the integrator .mdp option (replace with gmx em along with an mdp option to choose an EM flavour; gmx nm)
- gmx mdrun -gcom (need freedom to refactor integrators, and hopefully remove the motivation for this option; see also #1925)
- Benchmarking functionality (gmx mdrun -nsteps n -resetstep x -resethway are not something normal mdrun users should use, and creates questions about gmx mdrun -c confout.gro and checkpointing; whereas gmx benchmark has a clear use and more options for implementations)
- Aspects of editconf, trjconv, eneconv and trjcat that belong as separate tools (e.g. convert renumber cat subsample demux)
- gmx do_dssp (at some point we will want to reimplement this natively)

We should also have a docs page for all features that are deprecated, along with the version in which they got that status. Release notes will record when they were removed.

Related issues:
Related to GROMACS - Task #1925: remove concept of unilateral global communic... In Progress
Related to GROMACS - Task #1868: implement mdrun -rerun better, simplifying d... Closed
Related to GROMACS - Task #1781: re-design benchmarking functionality Accepted
Related to GROMACS - Task #2495: replace -noconfout with mdp option Accepted
Related to GROMACS - Bug #2717: mdrun runs infinitely when checkpoint file is... Closed
Related to GROMACS - Bug #2757: mdrun refuses to start with .cpt if nsteps is... Closed
Related to GROMACS - Task #3256: Remove -nsteps option from mdrun New
Related to GROMACS - Feature #3242: Please do not remove the -nsteps flag New

Associated revisions
Revision cf2d8336 - 10/13/2018 09:53 PM - Mark Abraham
Deprecate various functionality in GROMACS 2019

Published a deprecation policy.

Updated the release notes to refer also to previously deprecated features.

Announced intent to change some functionality:

- gmx mdrun -membed options (but not feature)
- gmx mdrun -rerun option (but not feature)
- integrator .mdp field will contain only integrators
- gmx do_dssp to be replaced by gmx dssp
- gmx trjconv and friends to be split and rewritten

02/24/2020
List of newly deprecated functionality:
- conversion of aromatic rings to virtual sites
- gmx mdrun -table options (but not feature)
- gmx mdrun -gcom option and feature
- gmx mdrun -nsteps option and feature
- gmx mdrun -nsteps -resetstep -resethway moved to
  a gmx benchmark tool
- gmx mdrun -confout removed

Also updated release notes for functionality removed in GROMACS 2019.
Refs #2495, #1781
Fixes #2569, #1925
Change-Id: I1d00859d0f15409a472984f5a65347a50c71ad17

Revision 21f509ef - 11/05/2018 02:28 PM - Paul Bauer

Brought all the old interfaces back to a state where they can always compile regardless of the build configuration, and give fatal errors if used from a configuration that didn't support the method.

When configured, this should work as before, but we have no ability to test that in Jenkins.

Added some necessary const correctness.

Did QM/MM preparation all in the same place, to simplify runner.cpp

Added deprecation status to release notes.
Refs #2706, #2569
Change-Id: I14a6566c60bbf27a7b1916be1874b36987b7da5

History

#1 - 09/10/2018 11:47 AM - Mark Abraham
- Description updated

#2 - 09/10/2018 03:33 PM - Mark Abraham

Mark Abraham wrote:

At today's Stockholm dev meeting we were generally in favour of 2019 noting the intent to remove or change the UI for most of these things. Notes below.

- aromatic vsite construction in pdb2gmx

Deprecate in 2019 and remove for 2020.

- gmx mdrun -rerun (replace with gmx rerun and gmx tpr; see also #1868)

2019 log files will note the intent to switch away from mdrun-based implementation for these two and normal-mode analysis (but without committing us to an approach now)

Separate discussion on energy groups. Rerun currently defaults to CPU if energy groups are on. Proposed that grompp warn when energy groups + PME are both on (as this combination is not clearly useful and can't run efficiently on a GPU). Agreed not nice that energy group .tpr + GPU NB writes zeroes (and we have had gmx-user posts confused about this). 2019 will make an energy group run default to the CPU, and issue a warning (ie. to console). (This is different from the way PME + FEP will not run PME on a GPU, because there is only one way to implement the user's requirements in that case, and those requirements are clearly useful ones.)

- gmx mdrun -membed (replace with gmx membed; we had the old g_membed, whose implementation we fixed, but the distinct tool is useful)

Didn't discuss, but Mark observes that the same considerations apply as for rerun.

- non-integrators in the integrator .mdp option (replace with gmx em along with an mdp option to choose an EM flavour; gmx nm)
This is a less clearly useful change. gmx em gives a little bit more freedom to implement warnings and have docs that make clear what command-line options work and don't work (but much is common with gmx mdrun).

- gmx mdrun -gcom (need freedom to refactor integrators, and hopefully remove the motivation for this option; see also #1925)

2019 to note that this is deprecated, but we are not sure whether recent improvements are enough to remove this option, nor how bad a problem it might prove to be for integrator refactoring projects.

- Benchmarking functionality (gmx mdrun -ntsteps n -resetstep x -resethway are not something normal mdrun users should use, and creates questions about gmx mdrun -c confout.gro and checkpointing; whereas gmx benchmark has a clear use and more options for implementations)

Deprecate in 2019, probably implement in gmx benchmark for 2020.

- Aspects of editconf, trjconv, eneconv and trjcat that belong as separate tools (e.g. convert renumber cat subsample demux)

Several of the command-line options of these tools should note on the terminal our intent to change how this functionality is supported.

- gmx do_dssp (at some point we will want to reimplement this natively)

Christian to see if there is any low-hanging fruit to improve usability. MDAnalysis also uses a DSSP binary that users have to install.

#3 - 09/11/2018 04:58 PM - Szilárd Páll
What's the policy on backward-compatibility? For one year kept + warning, after we can brake things?

#4 - 09/13/2018 01:47 AM - Mark Abraham
- Related to Task #1925: remove concept of unilateral global communication added

#5 - 09/13/2018 01:47 AM - Mark Abraham
- Related to Task #1868: implement mdrun -rerun better, simplifying do_md added

#6 - 09/19/2018 03:07 PM - Mark Abraham
- Related to Task #1781: re-design benchmarking functionality added

#7 - 09/19/2018 03:14 PM - Mark Abraham
- Related to Task #2495: replace -noconfout with mdp option added

#8 - 09/27/2018 08:00 PM - Mark Abraham
Szilárd Páll wrote:

What's the policy on backward-compatibility? For one year kept + warning, after we can brake things?

That's the practice, but I don't think we've written it down anywhere.

#9 - 09/27/2018 08:01 PM - Mark Abraham
Not strictly a deprecation, but we've negotiated with NVIDIA (who originally contributed the feature) that we will remove NVML support in GROMACS 2019.

#10 - 10/05/2018 10:29 AM - Mark Abraham
We should also deprecate mdrun -table -tablep -tableb, since in future tables will be handled by grompp

#11 - 10/05/2018 11:49 AM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #2569
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: gromacs~master~I1d00859d0f15409a472984f5a65347a50c71ad17
Gerrit URL: https://gerrit.gromacs.org/8488

#12 - 10/08/2018 06:48 PM - Mark Abraham
- Status changed from New to Fix uploaded
Further suggestion:

That we deprecate QM/MM other than Gaussian and MiMiC (I have a patch that will make them build and pass checks in normal GROMACS build configurations, but unless someone wants to contribute test cases (and the ability to run them), and do a proper build system for finding and linking to libraries, then we will remove these for GROMACS 2020. See #2706 and linked issues

Actually we already did (runner issues a notice), so all we need to do is update the deprecation documentation. I propose we remove this stuff in master immediately after 2019 release

How do people feel about deprecating the mdrun-only build? It was intended as a low-dependency version of mdrun for running on HPC clusters, but if we're aiming at a rosy gmxapi exascale future, then that's the opposite use case. We'll still have a version of libgromacs buildable without python for a long time yet. The mdrun-only build has not been tested in Jenkins for several years, because it requires someone to do the work of having another build of gromacs available that has gmx grompp, gmx check, etc. Neither the mdrun integration tests nor gmxapi will be suitable for testing it.

We tried to have a few full gmx builds and several mdrun-only builds, but in the end, it wasn't as convenient as we expected. tpr version mismatches were typical (in daily lab work, when you a have a set of projects with different timespans and based on different gromacs version). And the reduction in compiling efforts (even on the supercomputer) and binary size was almost negligible.

mdrun-only builds are extremely useful on heterogeneous clusters. When I build a specific GROMACS version, I build everything in single-prec, without GPU support, without dynamic linking. This gmx executable is used for GROMACS tools invocations on diverse workstations without problems. Additionally, I build mdrun-only with various SIMD settings in single, with GPU support, and in double, so that for each CPU type, the optimal SIMD instructions are used.

So, if we had a "portable x86" build that offered full performance, there'd be no need for that setup?

We tried to have a few full gmx builds and several mdrun-only builds, but in the end, it wasn't as convenient as we expected. tpr version mismatches were typical (in daily lab work, when you a have a set of projects with different timespans and based on different gromacs version). And the reduction in compiling efforts (even on the supercomputer) and binary size was almost negligible.
That sounds like issues with people loading the wrong modules, which you'd have had whether or not you had mdrun-only builds as part of those module loads.

#22 - 10/30/2018 04:56 PM - Carsten Kutzner
Mark Abraham wrote:

Carsten Kutzner wrote:

mdrun-only builds are extremely useful on heterogeneous clusters. When I build a specific GROMACS version, I build everything in single-prec, without GPU support, without MPI, without dynamic linking. This gmx executable is used for GROMACS tools invocations on diverse workstations without problems. Additionally, I build mdrun-only with various SIMD settings in single, with GPU support, and in double, so that for each CPU type, the optimal SIMD instructions are used.

So, if we had a "portable x86" build that offered full performance, there'd be no need for that setup?

A portable x86 build including various SIMD would be great. But still one might want single/double, and, more importantly, build the gmx* tools without MPI and CUDA. This way you don't have to source some CUDARC or add MPI and CUDA modules before you can execute gmx.

#23 - 10/30/2018 06:13 PM - Arthur Zalevsky
Mark Abraham wrote:

Arthur Zalevsky wrote:

We tried to have a few full gmx builds and several mdrun-only builds, but in the end, it wasn't as convenient as we expected. tpr version mismatches were typical (in daily lab work, when you a have a set of projects with different timespans and based on different gromacs version). And the reduction in compiling efforts (even on the supercomputer) and binary size was almost negligible.

That sounds like issues with people loading the wrong modules, which you'd have had whether or not you had mdrun-only builds as part of those module loads.

Oh, i wasn't clear enough. That definitely was a human problem. But having full gmx binary (rather than mdrun-only) simplifies rebuilding of the tpr inplace. But again it can just be a local peculiarity because we usually perform several steps of mechanical calculations before going into qm/mm. And sometimes switch between mm - qm/mm - mm several times (for a multistep reaction, for example) and thus have to use the whole zoo of version during one project.

#24 - 11/08/2018 12:24 PM - Mark Abraham
- Related to Bug #2717: mdrun runs infinitely when checkpoint file is beyond the designated end point added

#25 - 11/15/2018 12:27 AM - Mark Abraham
- Related to Bug #2757: mdrun refuses to start with .cpt if nsteps is -1 in .tpr added

#26 - 11/29/2018 10:54 AM - Mark Abraham
Summarizing - nobody has yet identified a clear need for the mdrun only build. Things like low-dependency builds and SIMD-portable builds are just as readily satisfied with the normal build.

#27 - 12/17/2018 01:45 PM - Paul Bauer
- Status changed from Feedback wanted to Resolved

I think this has been resolved for now. Any further deprecations should go in an issue for 2020.

#28 - 12/17/2018 01:45 PM - Paul Bauer
- Status changed from Resolved to Closed

#29 - 01/09/2020 02:01 PM - Mark Abraham
- Related to Task #3256: Remove -nsteps option from mdrun added

#30 - 01/09/2020 02:02 PM - Mark Abraham
- Related to Feature #3242: Please do not remove the -nsteps flag added