June 14, 2022 by Stig Syndergaard.

Notes related to Ian's 'Carrried over ROPP 9.1 tickets' v1.4 document in email of 31/8 2021.

Mostly ROPP 10 changes (or already in ROPP 9.1)
- #529: OK!
- #530: Seems that Ian forgot the last bug-fix in 5189.
- #531: OK!
- #532: OK!
- #533: OK!
- #534: OK!
- #535: OK!
- #536: OK!
- #537: OK!
- #538: OK!
- #539: OK!
- #540: OK!
- #541: OK!
- #542: OK!
- #543: Minor differences in the setting of some quality variables in ropp_fm_bg2ro_1d.f90
- #544: OK!
- #545: OK!
- #546: OK!
- #547: OK!
- #548: We have a difference of opinion here. I don't understand why a warning should be issued (with exit code = 1) when the user deliberately set --no-ranchk. It is anyways not done consistently in all tools.
- #549: OK!
- #550: OK! The puzzling loop near the end could be removed - left-over from earlier where it made better sense because last loop used to be conditional (r4924).
- #551: Additional bug-fix was implemented in dmi_trunk_10.0 (r6897). Corresponding adjoint code not fixed yet. Also many other changes related to 1dvar in dmi_trunk_9.0 (r5847-r5853, r5857-r5858, r6170, r6175, r6594) that are not in ROPP 11. Note also Ian's last comment in this ticket.
- #552: This is implemented in ROPP110_DMI_changes. The solution in the dmi_trunk is simpler and the code is closer to what is in ROPP 11.0, although with important differences. It was documented in #498, last comment. The solution in ROPP110_DMI_changes could still go into ROPP 11.1, since a lot of testing and new test files was made that are useful. The solution in dmi_trunk_11.0 could be implemented afterwards, and should then pass tests easily. See also #711.
- #553: OK! Depends on #552
- #554: OK!
- #555: OK!
- #556: OK!
- #557: OK!
- #558: OK!
- #559: OK!

- #560: OK!
- #561: OK!
- #562: OK!
- #563: OK!
- #564: See #548. Also other warnings in ropp2ropp, e.g., --no-zapem.
- #565: OK!
- #566: OK!
- #567: OK!
- #568: OK!
- #569: See #564 and #548. Also warnings in ucar2ropp and eum2ropp.
- #570: OK!
- #571: See #569.
- #572: OK!
- #573: OK!
- #574: OK!
- #575: OK!
- #576: OK!
- #577: OK!
- #578: OK!
- #579: OK!
- #580: OK!
- #581: OK!
- #582: OK!
- #583: OK!
- #584: OK!
- #585: OK! We need to be aware if UCAR changes/has changed their definition of impact height in newer data (e.g., with COSMIC-2 and new COSMIC-1 reprocessing).
- #586: OK!
- #587: OK!
- #588: OK!
- #589: OK!

Mostly ROPP110_DMI_changes:
- #590: OK!
- #591: OK!
- #592: OK!
- #593: OK!
- #594: OK!
- #595: OK!
- #596: OK!
- #597: OK!
- #598: OK!
- #599: OK!
- #600: OK! Wonder why we don't just set ts = 0.02 or ts = 0.001 based on the (ts > 0.00105) check. That would need some testing.
- #601: OK!
- #602: OK!

- #603: OK!
- #604: OK!
- #605: OK!
- #606: OK!
- #607: OK!
- #608: OK! Added documentation is fine, but perhaps it should say 'Parts of the preprocessing implemented ...' in line 159 in pp_background.tex.
- #609: OK!
- #610: OK! Added documentation is fine.
- #611: OK!
- #612: OK!
- #613: OK!
- #614: OK!
- #615: OK! The go_l1_ip is not used. ENDIF in section 14 has been moved up so that TPs are calculated even when obs_ok = FALSE. Why the checks on (.NOT. invtool) in section 17 and 18? Why the smt_ba stuff and check on (invtool) in section 18 right before section 18.1? (this was already done right before section 7a). Why ropp_io_free(dum_metro) in section 21? It is also in section 24. The extra diagnostics in section 22 seems to be a remnant from an old version of ROPP (e.g., IC_badness_score was renamed to SO_badness_score years ago; cf. ropp_pp_diag2roprof.f90). Turns out that all of the above is due to additional changes in ropp_pp_occ_tool.f90 since r4607, although some later revisions have been included. The whole thing looks much better in ROPP111_prototype. Only real difference to dmi_trunk_11.0 there is the ropp_io_free(dum_metro) and the extra sobgr variables. Extra checks in ropp_pp_interpolate_latlonaz.f90 are fine!
- #616: OK!
- #617: OK!
- #618: OK!
- #619: OK!
- #620: OK!
- #621: OK!
- #622: OK!
- #623: OK!
- #624: OK!
- #625: OK!
- #626: OK!
- #627: OK!
- #628: OK!
- #629: OK!
- #630: OK!
- #631: OK!
- #632: OK! Minor differences to dmi_trunk_11.0 in comments and formatting.
- #633: OK! I think this is a good example of how inconvenient reference files are. Often when we make changes we make them to get different results (new configuration, new features, improvements, bug-fix, etc..). Then the tests fail, not necessarily because of coding mistakes, but because the reference files then needs to be updated. So what are they really good for? When we make changes that have no influence on results, we usually know that, and then tests pass. In a few cases tests do catch coding limitations or mistakes. But are

they really worth the trouble? In many of the above tickets the reference files were not configured to capture the changes, and tests passed without new features actually being tested, presumably because it is a lot of work to make new reference files that can capture all different combinations of processing options. Aren't there better ways of testing?
- #634: OK!
- #635: OK! Note that r5427 was the result of the merge of dmi_trunk_8.1 and ROPP 9.0 to become dmi_trunk_9.0. This is where DMI started using parallel/serial instead of convoluted/regular because we in the merge accepted the nomenclature in ROPP 9.0.
- #636: OK!
- #637: OK!
- #638: OK!
- #639: OK!
- #640: OK!
- #641: OK!
- #642: OK!
- #643: OK!
- #644: OK!
- #645: OK!

<u>Subsequent relevant tickets:</u>
- #646: OK!
- #647: OK! DMI ticket. This change somehow didn't make it to dmi_trunk_10.0. Probably I missed it in the merge.
- #652: OK!
- #654: OK! See #615
- #661: OK! See #552
- #662: DMI ticket.
- #694: DMI request. Fixed in dmi_trunk_11.0 but not in ROPP111_prototype.
- #696: DMI ticket. Fixed in dmi_trunk_11.0
- #697: DMI ticket. Fixed in dmi_trunk_11.0
- #698: DMI ticket. Fixed in dmi_trunk_11.0
- #699: OK!
- #702: OK!
- #707: DMI ticket. Fixed in dmi_trunk_11.0
- #708: DMI ticket.
- #709: OK!
- #711: OK! for me to do.