2020-08-27: Obs Diagnostics

Yannick opened the meeting with two announcements:

1. Please continue to fill out your Google profiles. Put in your organization in the "Company" entry, and enter your real name. These are for tracking and reporting the work being done, and for easy identification of users.
2. We have a couple in-kind contributors for the upcoming documentation and test sprints. Thank you for signing up! We could use a couple more volunteers. Please send email to Yannick if you want to volunteer.

Then we proceeded to the topic of the day, observation diagnostics, with presentations from JJ and Travis.

JJ presented the following slides (which can also be found on Google drive here):

PANDA-C_obs-spa...nostics_JJG.pdf

Here is a summary of the presentation. Please see the slides for details.

JJ presented a system for processing and viewing data related to observations from a JEDI run which are saved in the IODA output files. The process consists of two steps and is implemented in Python. The first is to create a "statistics database" file from the IODA file, and the second is to read the "statistics database" file into a StatDB object (wrapper around a Pandas dataframe) and produce an analysis (plots, gross statistics, etc.) from the StatsDB object. JJ showed several examples of what can be produced by this system which include O-minus-F, gross statistics, time series, profiles, etc for a wide variety of observation types. See the slides for example plots.

JJ explained that the large amount of data that are produced from the JEDI run need to be organized and trimmed down to be manageable for diagnosis. The system takes a binning approach that creates subsets of data which can be processed in parallel. Once statistical quantities of the subsets are calculated, these can be aggregated into the statistics for the whole data set. The binning process is configurable and offers a wide variety of selection choices. JJ mentioned that this flow is extensible and should work for diagnostics such as: obs-space score cards, model-space diagnostics and correlation. JJ noted that while the first two examples would be readily available, the correlation would require some additional work.

Iliana asked if this system could be used with our small test cases (~100 observations). JJ answered yes, but cautioned that the resulting diagnostics may not be statistically significant.

Yannick asked how the system is relying on the ioda file format. JJ responded that there is a single class that handles reading the ioda file, and he plans to refactor this to use the new ioda-engines interface.

Ting Lei asked if all data from all cycles have to be read to obtain a bootstrap generated statistic such as a confidence interval. JJ acknowledged that for highest accuracy the bootstrap process does require this. However, the bootstrap can also be accomplished with subsets and aggregation (as shown in the presentation) which should run much faster, but will cause some accuracy loss.

A discussion started at this point in regard to the overlap between this project and other ongoing projects, for example FSOI monitoring, MET+ and ioda-plots (see below, Travis' presentation). The different systems each have their advantages and it was agreed that it would be worthwhile to compare the different projects and to explore collaboration and the possibilities of converging to a common system.

Travis presented the following slides (also available here on Google Drive):

ioda-plots.pdf

Here is a summary of the presentation. Please see the slides for details.

Travis presented a system for doing observation diagnostics which is implemented in Python and currently resides in the JCSDA/ioda-plots repository. This system is similar to that presented by JJ, with a couple distinctions. ioda-plots is configurable from YAML (JJ's system uses Python dictionaries embedded in the code), and is designed to automatically dump out a set of plots for the near-real time (NRT) SOCA website with as little configuration as possible. The steps in the ioda-plots flow are similar: bin the data down to smaller data sets (saved in files) followed by the generation of diagnostics (plots, statistics, etc.). O-minus-B, O-minus-A, obs values and counts binned into 1 degree lat,lon along with vertical profiles are the defaults for the NRT website. Travis pointed out that it is easy to compare two different experiments (ie, generate difference plots) with ioda-plots.

The discussion concerning project overlap and project convergence started up again. It was noted that we have several AOP Epics about diagnostics (SOCA, OBS, JEDI projects) plus we have a number of JCSDA github repositories (ioda-plots, fsoi, plotting_diagnostics, DAdiagnostics) dedicated to diagnostics. It was agreed that the AOP Epics and repositories need to be consolidated. Yannick set the goal of converging to one repository and suggested the creation of a focused group to work toward this goal. There was also some discussion of which plotting/diagnostics tools/repos, if any, should be included in the release.

Chris S asked the group for a list of what was missing from the diagnostics systems presented and discussed today. We heard details about what the RUC group is doing from Ming and what EMC is doing from Rahul. Points that were made include plotting long term bias and make sure to preserve the O-minus-F from runs as they progress (since O-minus-F is used by several groups for diagnostics). Tom summarized nicely by describing a 3-tiered approach to diagnostics:

1. Tier 1: cached pre-made plots (readily available for posting, viewing)
2. Tier 2: on demand statistics (pre-made stats, accumulated)
3. Tier 3: exotic plots (post process after DA run completes)

The tiers can be modified as we move forward. For example a tier 3 plot that becomes popular could be moved to tier 2 so it becomes available on demand.

Steve H noted that a Jupyter notebook tutorial exists in the ioda-engines repository when can be used to familiarize oneself to the Python API.

At this point, Yannick closed the meeting.