NOAA Model Diagnostics Task Force (MDTF)
Description of diagnostics Applied Programming Interface (API)

1. Motivation

A critical need exists to improve the diagnosis of global climate and forecasting models. A key need is incorporation of process-oriented diagnostics into standard diagnostics packages that can be applied to development versions of the models, allowing the application of diagnostics to be repeatable across multiple model versions. A "process-oriented diagnostic" characterizes a specific physical process or emergent behavior that is hypothesized to be related to the ability to simulate an observed phenomenon. A significant barrier is the lack of a mechanism for getting community-developed diagnostics into the modeling center development process.

To reduce this barrier, the MDTF aims to create a diagnostic package that is portable, extensible, usable, and open for contribution from the community. A goal is to allow diagnostics to be repeatable inside, or outside, of modeling center workflows. These are diagnostics focused on model improvement, and as such a slightly different focus from other efforts. Hopefully these diagnostics will be compatible with other efforts.

The Application Programming Interface (API) described below takes this motivation and is developed with several additional practical principles. (A) Diagnostics need to be open source and (B) need to be able to run on Climate and Forecast (CF) model output formats (i.e. CMIP format).

The API described below is being developed by the NOAA Model Diagnostics Task Force provides the specifications for a common and extensible mechanism for rapid dissemination of process-oriented diagnostics across modeling centers.

2. API Description

Two current options are to embed diagnostics into an existing variability package (e.g. that which runs at NCAR and GFDL, originally the NCAR AMWG Variability Package), or to use a new diagnostic script detailed in this document. They should be interoperable, and the new script a lightweight version of the package. A script will provide a portable mechanism for developing and running diagnostics that an individual user can run. It can also serve as the basis for collecting diagnostics into the center workflows.

API Sample

The Application Programming Interface (API) for the diagnostic script consists of a python script that does 3 things:

1. Set up paths and variable names
2. Call custom diagnostics to generate plots
3. Compose plots into a web page

The python script can be stand alone, or can be built into the NCAR and GFDL workflows.

The prototype implementation of the concept is now available in a demonstration for internal testing by the MDTF team, which runs a sample diagnostic on CESM model output, and composes a web page with a set of figures. These steps are described below.

User Diagnostics

Diagnostic utilities (user developed code that generates analysis and plots) just have to be callable from python by an open source package. Python would be recommended for the diagnostic utilities, but not required. NCL or Fortran are other recommended options. We will not specify the language of the code, although it is strongly recommended to use open source code, and not write diagnostics in a proprietary package/program that needs a license. While diagnostic code in a proprietary package might be able to run locally in a modeling center given appropriate licenses, such code may limit wider dissemination. Note that there exist many open source approximations (and even specific python approximations) of many common proprietary packages (e.g. matplotlib in python is based on MATLAB).

Observational datasets

Naturally observations for model evaluation are an important part of the package. Observations should be focused on the diagnostic at hand, and 'preprocessed' as much as possible (e.g. climatologies, distribution functions) for direct and efficient application to model results. Ideally, plots based on observations can be produced in advance and included in the package. The goal is not to dumb down the evaluation, but to pre-process observations to improve efficiency and processing speed, and to reduce the need to send around large volumes of data with the package.

How does this work in practice? A satellite data set that is processed for example on a profile by profile basis (L2) and then gridded into seasonal climatologies for comparison to observations should be preprocessed so only the final gridded seasonal field is present in the package. Or, a high frequency diagnostic for precipitation intensity can be pre-processed into a PDF of several values (or raw counts in bins) in space and time. Diagnostics can process raw observations, but should have an option to read in a pre-calculated curve from a netCDF file to save time (and the option to write the file for the curve if desired). The pre-processed observations could be supplied with the package code.

Output

The output will be a series of figures (suggest gif, png, jpg) and HTML code that links the figures together with links, and then compresses the whole package of figures and HTML so they can be posted on a web server. Figures are thus aimed at screen/web resolution. If postscript is generated they should be converted to something commonly displayed on a web page. An option to save postscript is available in the sample package.

The diagnostics should output a list of figure file names. There is a block of code that can parse the file names into links for HTML. An example is contained in the demonstration code being tested by the MDTF. Note that a diagnostic could build a custom web page that could be linked to the diagnostics as well with a single link.

The method permits an iterative approach: diagnostics run in series (now), but independently, and contribute to a common web page. The result is a very extensible package, where a user can work on one diagnostic, submit it, and grow a library of diagnostics.

In addition, the MDFT will aim to produce a routine to calculate terms for the Moist Static Energy (MSE) budget from both GFDL and NCAR models. This might be one routine for each model. We might include both a direct calculation of the vertical advection term and also the vertical advection term computed as a residual. The goal will be to provide standard output for the MSE budget terms that other diagnostics can use, and maybe some basic plots. This can be built into the MDTF diagnostics. Other common calculations are possible as well.

Detailed Elements of the Diagnostic API

The initial package contains a simple python script (mdtf.py) that executes a single sample diagnostic, easily extensible to a broader set of diagnostics.

A. Paths and Variable Names

The input is model output file names and a structure of paths to the data, diagnostics and output. We will set up appropriate variable names for CESM and GFDL, as well Climate and Forecast (CF) metadata variable names for CMIP5/6 model output. This should enable users to read any model format. The demonstration code currently produces an example for CESM. We may eventually read in these names from python files for clarity, and the eventual goal is to convert the internal names to use CF conventions.

B. Call custom diagnostics to generate plots

This is a simple part of the overall script that calls a series of python interfaces to diagnostics. This could most simply be a python script that has inherited all the paths from the main diagnostic script. The example shown actually uses 'sub-process' calls to call an NCL (NCAR Command Language) script. Fortran would be another example of something that could commonly be called. As mentioned above, proprietary packages such Matlab would be discouraged.

C. Compose plots into a web page

The last component occurs in two places: the overall driver script sets up the web page and header before diagnostics are called (eventually this will be a sub-process/subroutine/function call). Finally, after diagnostics it will also finish the footer for the web page, and then compress the html and plots together.

The individual diagnostics have to process files (for example, convert to a web readable format from postscript) and make sure they are in an appropriate directory. They also need to write to the common HTML web page the links to the images, and any desired text. A sample of how to do this simply is provided, and can be copied and pasted into the diagnostics.

A diagnostic can also build its own separate web page, as long as it is linked to in the master web page.

The goal is that the web page is extensible. Eventually it could be formatted into a series of pages, but for now (first phase) a single page is appropriate.

The goal is to have the architecture that implements the API in a series of scripts available for download (likely from NCAR). Ideally we would include user-developed diagnostics back into the script, with options to turn them on and off as desired.

3. Opportunities for Extension to the Broader Community

Long term, the diagnostics are complementary with the ESMval (https://www.esmvaltool.org/) package format, and should be extensible into that framework. However, opportunities may be available for more immediate dissemination to other modeling centers. While the package being developed should be sufficiently user-friendly such that it easily adopted, dissemination to other modeling centers will likely be aided through development of collaborations with centers such as NASA Goddard, Lawrence Livermore National Laboratory, and the UK Met Office, as well as collaborations with international bodies such as WMO's WCRP Working Group on Climate Modeling (WGCM), Working Group on Numerical Experimentation (WGNE), and the WGNE Madden-Julian Task Force that are promoting dissemination of process-oriented diagnostics. Members of the NOAA Model Diagnostics Task Force have existing collaborations at these institutions that might be exploited to ease incorporation of these diagnostics into the respective modeling center evaluation packages. Fostering such collaborations will be encouraged as this diagnostic development activity matures. Encouragement will also be made for external diagnostic developers to contribute to the package. Opportunities for advertisement to the broader diagnostic community will exist at scientific conferences, postings on websites (e.g. MAPP program), and in online newsletters.