Geosci. Model Dev. Discuss., 8, 3823–3859, 2015
www.geosci-model-dev-discuss.net/8/3823/2015/
doi:10.5194/gmdd-8-3823-2015

Geoscientific Model Development Discussions
Open Access

A new ensemble-based consistency test for the Community Earth System Model

A. H. Baker, D. M. Hammerling, M. N. Levy, H. Xu, J. M. Dennis, B. E. Eaton, J. Edwards, C. Hannay, S. A. Mickelson, R. B. Neale, D. Nychka, J. Shollenberger, J. Tribbia, M. Vertenstein, and D. Williamson

The National Center for Atmospheric Research, Boulder, CO, USA

Received: 15 April 2015 – Accepted: 16 April 2015 – Published: 8 May 2015

Correspondence to: A. H. Baker (abaker@ucar.edu)

Published by Copernicus Publications on behalf of the European Geosciences Union.

Abstract

Climate simulations codes, such as the Community Earth System Model (CESM), are especially complex and continually evolving. Their on-going state of development requires frequent software verification in the form of quality assurance to both preserve the quality of the code and instill model confidence. To formalize and simplify this previously subjective and computationally-expensive aspect of the verification process, we have developed a new tool for evaluating climate consistency. Because an ensemble of simulations allows us to gauge the natural variability of the model's climate, our new tool uses an ensemble approach for consistency testing. In particular, an ensemble of CESM climate runs is created, from which we obtain a statistical distribution that can be used to determine whether a new climate run is statistically distinguishable from the original ensemble. The CESM Ensemble Consistency Test, referred to as CESM-ECT, is objective in nature and accessible to CESM developers and users. The tool has proven its utility in detecting errors in software and hardware environments and providing rapid feedback to model developers.

1 Introduction

The Community Earth System Model (CESM) is a state-of-the-art fully-coupled, global climate model whose development is centered at the National Center for Atmospheric Research (NCAR) (Hurrell et al., 2013). The Earth's global climate is complex, and CESM is widely-used by scientists around the world to further our understanding of the future, present and past states of the climate system. For large simulation models such as CESM, verification and validation are critical to establishing and maintaining a model's credibility, particularly when the model is used to make decisions (e.g., Carson II, 2002). Note that differences in interpretation exist among scientific communities in regards to the terms verification and validation (e.g., Oberkamf and Roy, 2010). Validation generally focuses on how well the model represents the real world phenomena

GMDD
8, 3823–3859, 2015

that are being modeled. Verification involves determining whether the implementation of a model is correct and matches the intended description and assumptions for the model (see, e.g. Carson II, 2002; Sargent, 2011; Whitner and Balci, 1989; Oberkamf and Roy, 2010; Goosse et al., 2014).

Software verification necessarily requires the detection and reduction of errors or "quality assurance" (Oberkamf and Roy, 2010), and we focus on this component of verification for CESM. As with many scientific codes, development of CESM is on-going: features are continually added; improvements are made; software and hardware environments change. The primary motivation for this work is to ensure that changes during the development life cycle of CESM do not adversely affect the simulation. In particular, changes during CESM development that result in simulation output that is no longer bit-for-bit (BFB) identical to previous output data require attention to ensure that the output still produces the same climate (i.e., an error has not been introduced). Note that CESM simulations are expected to produce BFB reproducible output on the same machine and processor counts when the CESM version and parameters are identical." The approach to detecting potential errors in CESM has historically been a cumbersome process at best. For example, porting the CESM code to a new machine architecture results in non-BFB model output, and the current approach is as follows. First, a climate simulation of several hundred years (typically 400) is run on the new machine. Next, data from the new simulation is analyzed and compared to data from the same simulation run on a "trusted" machine, and, lastly, all results are given to a senior climate scientist for approval. This informal process is not overly rigorous and relies largely on subjective evaluations. Further, running a simulation for hundreds of years is resource intensive, and this expense is exacerbated as the model grows larger and more complicated. Clearly a more rapid, objective, and accesible solution is needed, particularly because a port of CESM to a new machine is just one example of a non-BFB change that requires quality assurance testing. Other common situations that can lead to non-BFB results include experiments with new compiler versions or optimizations, code modifications that are not expected to be climate-changing, and

GMDD
8, 3823–3859, 2015

many new exascale-computing technologies. The lack of a straightforward metric for accessing the quality of the simulation output has limited the ability of CESM users and developers to introduce potential code modifications and performance improvements that result in non-BFB reproducibility. The need for a more quantitative solution for ensuring code quality prompted our development of a new tool for assessing the impact of non-BFB changes in CESM. While verification always involves some degree of subjectivity and one cannot absolutely prove correctness (Carson II, 2002; Oberkamf and Roy, 2010), we aim to facilitate the detection of hardware, software, or human errors introduced into the simulation.

The quality assurance component of code verification implies that a degree of consistency must exist (Oberkamf and Roy, 2010). Our new method evaluates climate consistency in CESM via an ensemble-based approach that simplifies and formalizes the quality assurance piece of the current verification process. In particular, the goal of our new CESM Ensemble Consistency Test tool, referred to as CESM-ECT, is to easily determine whether or not a change in a CESM simulation is statistically significant. The ability of this simple tool to quickly assess changes in simulation output is a significant step forward in the pursuit of more qualitative metrics for the climate modeling community. The tool has already proven invaluable in terms of providing more feedback to model developers and increasing confidence in new CESM releases. Note that we do not discuss verification of the underlying numerical model in this work, which is considered at other stages in the development of individual CESM components. Further, we do not address model validation, but mention that it is primarily conducted via hindcasts and comparisons to real world data, e.g., the Intergovernmental Panel on Climate Change Data Distribution Center has a large collection of observed data (IPCC Data Collection Center 2015).

This paper is organized as follows. We give additional background information in Sect. 2. We describe the new CESM-ECT tool in Sect. 3. In Sect. 4, we provide results from experiments with CESM-ECT, and, in Sect. 5, we give examples of the utility of

GMDD
8, 3823–3859, 2015

the new tool in practice. Finally, we give concluding remarks and discuss future work in Sect. 6.

2 Background

Climate science has a strong computational component, and the climate codes used in this discipline are typically complex and large in size (e.g., Easterbrook et al., 2011; Pipitone and Easterbrook, 2012), making the thorough evaluation of climate model software quite challenging (Clune and Rood, 2011). In particular, the CESM code base, which has been developed over the last twenty years, currently contains about one and a half million lines of code. CESM consists of multiple geophysical component models of the atmosphere, ocean, land, sea ice, land ice, and rivers. These components can all run on different grid resolutions, exchanging boundary data with each other through a central coupler. Because CESM supports a variety of spatial resolutions and time-scales, simulations can be run on both state-of-the-art supercomputers as well as on an individual scientist's laptop. The myriad of model configurations available to the user contribute to the difficulty of exhaustive software testing (Clune and Rood, 2011; Pipitone and Easterbrook, 2012). A particularly fascinating and in-depth description of the challenges of scientific software in general, and climate modeling software in particular, can be found in (Easterbrook and Johns, 2009). Furthermore, the societal importance of better understanding Earth's climate is such that every effort must be made to verify climate codes as well as possible (e.g., Easterbrook et al., 2011).

In general, scientific codes are often in a near-constant state of development as new science capabilities are added and requirements change, and this is certainly true for CESM and other global climate models. However, despite the complexity of climate software, both the constant enrichment of the code base and the manner in which it has evolved over time has resulted in an overall quality of software superior to that of other open-source projects (Pipitone and Easterbrook, 2012). Yet the pace of evolution of the code requires that issues of correctness, reproducibility and software

GMDD
8, 3823–3859, 2015

quality are frequently being addressed. Coarse-grained testing is a common practice in climate modeling, and this global approach is useful for detecting the existence of errors in the software or input stack or the software and hardware environment (Clune and Rood, 2011). This approach does not offer information as to the source of the error, but rather as to whether or not one may exist. The goal of coarse-grained testing is not to prove correctness, but to point out potential incorrectness. Fine-grained testing is needed to identify the source of errors, and typically occurs within the individual CESM component models. Our focus in this work is on a coarse-grained approach to software quality assurance, and for climate models, this global approach typically takes the form of analysis of simulation output (Easterbrook and Johns, 2009). Visualizations of model output are commonly examined by climate scientists, and achieving bit-for-bit (BFB) identical results has been quite important to the climate community (Easterbrook and Johns, 2009; Pipitone and Easterbrook, 2012). If changes in the source code or software and hardware environment yield BFB results to the previous version, then this verification step is trivial. However, depending on the nature of the change, achieving BFB results from one run to the next is not always possible. For example, in the context of porting the code to a new machine architecture, machine-rounding level changes can propagate rapidly in a climate model (Rosinski and Williamson, 1997). In fact, changes in hardware, software stack, compiler version, and CESM source code can all cause round-off level or larger changes in the model simulation results, and the emergence of some heterogeneous computing technologies inhibit BFB reproducibility as well.

Some of the difficulties caused by differences due to truncation and rounding in climate codes that result in non-BFB simulation data are discussed in (Clune and Rood, 2011). In particular, the authors cite the need for determining acceptable error tolerances and the concern that seemingly minor software changes can result in a different climate if the simulation is not run for a sufficient amount of time. The work in (Rosinski and Williamson, 1997) is also of interest and aims to determine the validity of a simulation when migrating to a new architecture. They minimize the computational expense of a long run by setting tolerances for rounding accumulation growth based on the growth

GMDD
8, 3823–3859, 2015

of a small perturbation in the atmospheric temperature after several days. However, this test is no longer applicable to the atmospheric component of CESM, called the Community Atmosphere Model (CAM), because the parameterizations in CAM5 are ill-conditioned in the sense that small perturbations in the input produce large perturbations in the output. The result is that the tolerances for rounding accumulation growth are exceeded within the first few time steps. Our work builds on this idea of gauging the effects of a small temperature perturbation on the simulation, though improvements in software and hardware allow us to extend the simulation duration well beyond several days. Further, by looking only at climate signals, we relax the restriction on how the parameterizations respond.

3 A new method for evaluating consistency

In this section, we present and discuss a new ensemble consistency test for CESM, called CESM-ECT. We first give a broad overview, followed by more details in the subsequent subsections. As noted, CESM's evolving code base and the demand to run on new machine architectures often result in data that are not BFB identical to previous data. Therefore, our new tool for CESM must determine whether or not the *new* configuration (e.g., code generated with a different compiler option, on a new architecture, or after a non-climate changing code modification) should be accepted. For our purposes, we accept the new configuration if its output data is statistically *indistinguishable* from the *original* data, where the *original* data refers to data generated on a trusted machine with an accepted version of the software stack. Our tool must:

- determine whether or not data from a new configuration is consistent with the original data
- indicate the level of confidence in its determination (e.g., false positive rate)
- be user-friendly in terms of ease of use and minimal computational requirements for the end-user.

GMDD
8, 3823–3859, 2015

Note that this new tool takes a coarse-grained approach to detecting statistical differences. Its purpose is not to isolate the source of an inconsistency, but rather to indicate the likelihood that one exists. To this end, the CESM-ECT tool works as follows. The first step requires the creation of an ensemble of simulations in an accepted environment representing the original data. The second step uses the ensemble data to determine the statistical distributions that describe the original data. Next, several simulations representing the new data are obtained. And finally, a determination is made as to whether the new data is statistically similar to the original ensemble data.

3.1 Preliminaries

CESM data are written to "history" files in time slices in NetCDF format for post-processing analysis. Data in history files are single-precision (by default). For this initial work, we focus on history data from the Community Atmosphere Model (CAM) component in CESM, which is actively developed at NCAR. We chose to begin with CAM because the time-scales for changes propagating through the atmosphere are relatively short compared to the longer time-scales of other components, such as the ocean, ice, or land models. Further, the set of CAM global output variables is diverse, and the default number for our CESM configuration (detailed in the next section) is on the order of 130. An error in CAM would certainly affect the other component models in fully-coupled CESM situations; however, we cannot assume that CAM data passing CESM-ECT implies that the remaining components would also pass. Data from other components (e.g. ocean, ice, and land) will be addressed in future work, though we give an example in Sect. 5 of detecting errors stemming from the ice component with CESM-ECT.

3.2 An ensemble method

The development of a tool like CESM-ECT necessitates the determination of error tolerances that can be used to evaluate whether differences in climate data are sig-

GMDD
8, 3823–3859, 2015

nificant. Requiring that the difference be less than the natural variability of the climate system makes sense intuitively and is along the lines of Condition 2 in (Rosinski and Williamson, 1997). However, characterizing the natural variability is difficult with a single run of the original simulation. Therefore, we extend the sampling of the original data to an ensemble from which we can obtain a statistical distribution. An ensemble refers to a collection of multiple realizations of the same model simulation, generated to represent possible states of the system (e.g., Dai et al., 2001). Generally, small perturbations in the initial conditions are used to generate the ensemble members, and the idea is to characterize the climate system with a representative distribution (as opposed to a single run). Ensembles are commonly used in climate modeling and weather forecasting (see, e.g., Dai et al., 2001; Zhu and Toth, 2008; von Storch and Zwiers, 2013; Zhu, 2005; Sansom et al., 2013) to enhance model confidence, indicate uncertainly, and improve predictions. For example, the ensemble in (Kay et al., 2015) was created by small perturbations to the initial temperature condition in CAM and is being used to study internal climate variability.

We generate our ensemble for CESM-ECT by running simulations that differ only in a random perturbation of the initial atmospheric temperature condition of $\mathcal{O}(10^{-14})$. These perturbations grow to the size of NWP (Numerical Weather Prediction) analysis errors in a few hours. Each simulation is one-year in length, which is short enough to be computational reasonable, yet of sufficient length to allow the effects of the perturbation to propagate though the system. A perturbation of this size should not be climate-changing, and, while one year is inadequate to establish a climate, it is sufficient for generating the statistical distribution that we need. In particular, while the trajectories of the ensemble members will rapidly diverge due to the chaotic nonlinearity of the model, the statistical properties of the ensemble members are expected to be the same. Determining the appropriate number of ensemble members requires a balance between computational and storage costs and the quality of the distribution. The lower bound on the size is constrained by our use of Principal Component Analysis (PCA), which is described in the next subsection. PCA requires that the number of

GMDD
8, 3823–3859, 2015

ensemble members be larger than the number of CAM variables. We chose an initial ensemble size, denoted by N_{ens}, of 151 for CESM-ECT. At this size, the coefficient of variation for each CAM variable is well under five percent, save for two variables that are known to have large distributions across the ensemble. The cost to generate the ensemble is reasonable because all N_{ens}, members can be run in parallel, resulting in a much faster turn around time than for a single multi-century run (a single one-year simulation can run in a couple hours on less than a thousand cores). Note that, as explained further in Sect. 3.5, an ensemble is only generated for the control and not for the code to be tested. Hence, the ensemble creation does not impact the CESM-ECT user.

In summary, the CESM-ECT ensemble consists of $N_{ens} = 151$ one-year climate simulations, denoted by $E = \{E_1, E_2, \ldots, E_{N_{ens}}\}$, and is produced on a trusted machine with an accepted version, model, and configuration of the climate code. The data for these one-year ensemble runs consists of annual temporal averages at each grid point for the selected grid resolution for all N_{var} variables, which are either two- or three-dimensional. Retaining only the annual temporal averages for each variable helps to reduce the cost of storing the ensemble simulation output and has proved sufficient for our purposes. We denote the dataset for a variable X as $X = \{x_1, x_2, \ldots, x_{N_X}\}$, where x_i is a scalar that represents the annual (temporal) average at grid point i and N_X is the total number of grid points in X (determined by whether X is a 2-D or 3-D variable).

3.3 Characterizing the ensemble data

The next stage in our process is the creation of the statistical distributions that describe the ensemble data. In particular, information collected from the ensemble simulations helps to characterize the internal variability of the climate model system. Results from new simulations (resulting from a non-BFB change) can then be compared to the ensemble distribution to determine consistency.

First, based on the ensemble simulation output, CESM-ECT calculates the global area-weighted mean and root mean squared Z-score (RMSZ) distributions. These

GMDD
8, 3823–3859, 2015

measurements supply tangible information to climate scientists that together give them an indication of the average state and variability across the ensemble. In particular, for each ensemble member m, the global area-weighted mean is calculated for each variable X across all grid points i and denoted by $\overline{X}^m$. Next, to create the RMSZ distribution, recall that a Z-score is simply an indication of how many standard deviations a value is from its mean. Therefore, for each variable X, at each grid point i, we compare the value of x_i of ensemble member m (denoted by x_i^m) to the values of x_i in sub-ensemble $E\backslash m$, where $E\backslash m$ consists of all the ensemble members of E except for member m. This Z-score calculation requires the computation of mean and standard deviation of x_i at each grid point i in the *sub-ensemble* $E\backslash m$, which are referred to as $\overline{x}_i^{E\backslash m}$ and $\sigma_{x_i}^{E\backslash m}$, respectively. The Z-score for ensemble member m at each grid point i is then

$$Z_{x_i}^m = \frac{x_i^m - \overline{x}_i^{E\backslash m}}{\sigma_{x_i}^{E\backslash m}}. \tag{1}$$

To indicate the average Z-score for ensemble member m over all grid points, we calculate the root mean squared Z-score (RMSZ) for each variable X:

$$\mathrm{RMSZ}_X^m = \sqrt{\frac{1}{N_X}\sum_i \left(Z_{x_i}^m\right)^2}. \tag{2}$$

Repeating this process for each ensemble member results in 151 RMSZ scores for *each* output variable. Finally, to facilitate the computation of RMSZ score for new runs, we also compute both the mean and standard deviation at each grid point i for each variable X in ensemble E, denoted by $\overline{x}_i^E$ and $\sigma_{x_i}^E$, respectively. To summarize, the first stage produces the following global mean and RMSZ information describing the *original* data:

– $N_{\mathrm{var}} \times N_{\mathrm{ens}}$ global means,

GMDD
8, 3823–3859, 2015

– $N_{var} \times N_{ens}$ RMSZ scores,

– $N_{var} \times N_X$ per-point mean and standard deviations,

which are written to the CESM-ECT ensemble summary file.

Both the mean and RMSZ scores are of considerable value to scientists in terms of providing insight on the distribution across the control ensemble for each variable. However, determining whether or not the climate in the new run is consistent with the ensemble data based on the number of variables that fall within the distribution (or other specified tolerance) is difficult without a linearly independent set of variables. For the CESM 1.3.x series, 134 variables are output by default for CAM. We exclude several redundant variables as well as those with zero variance across the ensemble (e.g., specified variables common to all ensemble runs) from our analysis, resulting in $N_{var} = 120$ variables total. A correlation analysis shows that many of these variables are highly correlated (> 0.9). In fact, 52 variables are highly correlated in the global mean, and 43 in the RMSZ score, and 16 pairs are common to both. Determining objective and statistically-motivated criteria (such as false positive rates) necessitated a transformation of our variable-based data to a linearly independent data space. We use Principal Component Analysis (PCA), a popular tool in data analysis, to determine the orthogonal transform needed to convert the ensemble variable values into a set of principal component scores. The principal components are orthogonal and indicate the directions in which there is the most variance, i.e. in which the data is the most "spread out", thereby exposing underlying structure in the data that might otherwise be overlooked (e.g., Shlens, 2014). A second well-known advantage of PCA is that most of the variance in the system ends up being represented by many fewer components than the original number of variables, which simplifies analysis, particularly when there are large number of variables.

CESM-ECT applies PCA-based testing only to the global mean data, as we expect similar information to emerge from the RMSZ data. The implementation of this strategy into our tool entails the following steps. First, we standardize the $N_{var} \times N_{ens}$ matrix

GMDD
8, 3823–3859, 2015

containing the global means for each variable in each ensemble member and denote the result by $\mathbf{V}_{gm}$. Note that $N_{var} = 120$ and $N_{var} < N_{ens}$. Standardization of the data involves subtracting the ensemble mean and dividing by the ensemble standard deviation for each variable and is important because the CAM variables have vastly different units and magnitudes. Next, we calculate the transformation matrix, or "loadings", that project the variable space $\mathbf{V}_{gm}$ into principal component (PC) space. Loading matrix $\mathbf{P}_{gm}$ is size $(N_{var} \times N_{var})$ and corresponds to the eigenvector decomposition of the covariance of $\mathbf{V}_{gm}$, ordered such that the first PC corresponds to the largest eigenvalue and decreasing from there. Finally, we apply the transformation to $\mathbf{V}_{gm}$ to obtain the PC scores, $\mathbf{S}_{gm}$, for our ensemble:

$$\mathbf{S}_{gm} = \mathbf{P}_{gm}{}^{T}\mathbf{V}_{gm}. \tag{3}$$

Now instead of using a distribution of variable global means to represent our ensemble, the $N_{var} \times N_{ens}$ matrix $\mathbf{S}_{gm}$ forms a distribution of PC scores that represents the variance structure in the data. These scores have a mean of zero, so we only need to calculate the standard deviation of the ensemble scores in $\mathbf{S}_{gm}$, which we denote by $\sigma_{\mathbf{S}_{gm}}$. To summarize, this first stage computes the following data related to the PCA-based testing:

- N_{var} means of ensemble global mean values ($\mu_{\mathbf{V}_{gm}}$)
- N_{var} standard deviations of ensemble global mean values ($\sigma_{\mathbf{V}_{gm}}$)
- $N_{var} \times N_{var}$ loadings ($\mathbf{P}_{gm}$)
- N_{var} standard deviations of ensemble global mean scores ($\sigma_{\mathbf{S}_{gm}}$)

which are also written to CESM-ECT ensemble summary file.

The distribution of global mean *scores* from the ensemble, represented by the standard deviations in $\sigma_{\mathbf{S}_{gm}}$, can be used to evaluate data from a new simulation. Note that

GMDD
8, 3823–3859, 2015

most of the variance in the data is now largely represented by a few PCs. In fact, the coefficients on the first PC explain about 21 % of the variance and the coefficients on the second explain about 17 % of the variance. Figure 1 illustrates the importance of first several PC scores. Typically the less important PC scores are neglected, and one only examines the first N_{PC} components in an analysis. In this work, we use $N_{PC} = 50$, as little additional information is gained after the first 50 PC scores.

The CESM-ECT ensemble summary file (in NetCDF format) described here is generated for each CESM software tag on the Yellowstone machine at NCAR with the default compiler options (more details follow in Sect. 3.5).

3.4 Determining a pass or fail

The last step in the CESM-ECT procedure evaluates whether the new output data that has resulted from the non-BFB change is statistically distinguishable from the original ensemble data, as represented by the ensemble summary file. For simplicity of discussion, assume that we want to evaluate whether the results obtained on a new machine, Yosemite, are consistent (i.e. not statistically distinguishable) with those on Yellowstone. To do this, we collect data from a small number (N_{new}) of randomly selected ensemble runs on Yosemite. Variables in the new datasets are denoted by $\widetilde{X}$, where $\widetilde{X} = \{\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_{N_X}\}$. The CESM-ECT tool then decides whether or not the output data from simulations on Yosemite are consistent with the ensemble data and issues an overall pass or fail result.

CESM-ECT determines an overall pass or fail in the following manner. First, the weighted area global means for each variable $\widetilde{X}$ in all N_{new} runs are calculated, $\overline{\widetilde{X}^k}$ ($k = 1 : N_{new}$). These new variable means are then standardized using the mean and standard deviations of the control ensemble given in the summary file ($\mu_{\mathbf{V}_{gm}}$ and $\sigma_{\mathbf{V}_{gm}}$). Second, the standardized means are converted to scores via the loading matrix $\mathbf{P}_{gm}$ from the summary file. Next, we determine whether the first N_{PC} scores of the new runs are within m_σ standard deviations of the mean, using the standard deviation of

GMDD
8, 3823–3859, 2015

the zero-mean scores for the ensemble in the summary file ($\sigma_{\mathbf{s}_{gm}}$). Then for each of the N_{new} Yosemite simulations, the PC scores that fall outside the m_σ confidence interval are tagged as a "fail" for that particular run. Finally, CESM-ECT decides whether the simulations overall on Yosemite are consistent with those on Yellowstone by counting the number of times that each PC failed at least $N_{runFails}$ runs, where $N_{runFails} \leq N_{new}$. If at least $N_{pcFails}$ PCs fail at least $N_{runFails}$ runs, then CESM-ECT returns an overall "failure".

The parameters m_σ, N_{new}, $N_{pcFails}$, and $N_{runFails}$ are chosen to obtain a desired false positive rate. We performed an empirical simulation study and tested a variety of combinations of parameters. We found that choosing $m_\sigma = 2$ (which corresponds to the 95% confidence level), $N_{new} = 3$, $N_{pcFails} = 3$, and $N_{runFails} = 2$ yields our desired false positive rate of 0.5%. To summarize, we run 3 simulations on Yosemite, and if at least 3 of the same PCs fail for at least 2 of these runs, then CESM-ECT issues a "failure". We intentionally err on the conservative side by choosing a low false positive rate, hedging against the possibility that our ensemble may not be capturing all the variability that we want to accept. Also note that while perturbing the initial temperature condition is a common method of ensemble creation for studying climate variability, other possibilities exist, and we are currently conducting further research on the initial ensemble composition and its representation of the range of variability, particularly in regard to compilers and machine modifications.

If desired, CESM-ECT also provides information on whether the variable global means and RMSZ scores fall outside of the corresponding ensemble distributions in the summary file. While this information is often of interest to climate scientists, it does not directly affect the "pass/fail" determination made by CESM-ECT, as that is based purely on the PCA testing. For each variable $\widetilde{X}$ in all N_{new} runs, CESM-ECT can optionally calculate the Z scores that compare the values of $\widetilde{x}_i^k$ for each new run to that

GMDD
8, 3823–3859, 2015

of the original ensemble E by

$$Z^k_{\widetilde{x}_i} = \frac{\widetilde{x}_i^{\,k} - \overline{x}_i^E}{\sigma^E_{x_i}}. \quad (4)$$

The RMSZ scores for each variable in each run are then obtained from the Z-scores as before. The N_{new} global means and RMSZ scores are examined independently for all N_{var} variables. CESM-ECT can report whether for each variable $\widetilde{X}$ in the new data, the global mean, $\overline{\widetilde{X}}$, falls within the original ensemble distribution's range, and, similarly for the RMSZ of $\widetilde{X}^k$.

3.5 CESM-ECT software tools

Finally, we further discuss the software tools needed to test for ensemble consistency that are included in the CESM public releases (see Sect. 6 for details). Generating the ensemble simulation data by setting up and running the $N_{\text{ens}} = 151$ one-year simulations is the most compute-intensive step in this ensemble consistency-testing process. The CESM Software Engineering group generates ensembles as needed. For example, generating new ensemble simulation data is now routine when a CESM software tag is created that contains a scientific change known to alter the climate from the previous tag. (The frequency of such tag creation varies, but is several times a year on average). While the utility used to generate the ensemble runs is included in CESM releases, a typical end-user does not need to generate their own ensemble. Note that our consistency-testing methodology can be extended to other simulation models, and, in that case, an application-specific tool to facilitate the generation of N_{ens} simulations would be needed for the new application.

Whenever a new ensemble of simulations is generated, a summary file (as described in Sect. 3.3) must be created for the ensemble. The ensemble summary utility (pyEnsSum), written in parallel Python, creates a NetCDF summary for any specified number (N_{ens}) of output files. This step requires far less time than it takes to run the

GMDD
8, 3823–3859, 2015

simulations themselves. As an example, generating the summary file for 151 ensemble members on 42 cores of Yellowstone takes about 20 minutes (we chose the number of cores to be equal to the number of 3-D variables). Note that the summary creation takes less than a minute if we only compute the information needed for the PCA test (i.e., exclude the most computationally expensive part, the RMSZ calculations). Each CESM software tag now includes the corresponding ensemble summary file. Including the summary file in the CESM releases facilitates tracking data changes in the software life cycle and enables CESM users to run CESM-ECT without creating an ensemble of simulations themselves. Note that the storage cost for a single summary file is minor compared to the cost of storing the simulation output for the entire ensemble.

In addition to an ensemble summary file, our Python tool CESM-ECT (pyCECT) requires $N_{new} = 3$ one-year simulations from the configuration that is to be tested. For a CESM developer or advanced user, this may mean using a development version of code with a modification that needs to be tested. For a basic CESM-user, this may mean verifying that the user's installation of CESM on their personal machine is acceptable. In either case, a simple shell script that creates one-year CESM run cases (with random initial perturbations) for this purpose is also included in CESM releases, though advanced users can certainly generate more custom simulations if desired. Regardless, after the N_{new} simulations have completed, pyCECT determines whether results from the new configuration are consistent with the original ensemble data based on the supplied new CAM output files and specified ensemble summary file. Then pyCECT reports whether of not the new configuration has passed or failed the consistency test, as well as which PCs in particular have passed or failed each of the N_{new} simulations contributing to the overall pass/fail rating. In addition, the user may assign values to the pyCECT parameters m_σ, N_{new}, $N_{pcFails}$, $N_{runFails}$, and N_{PC} via input parameters if the defaults are not desired.

For clarity, Fig. 2 illustrates the workflow for the CESM-ECT process. The two Python tools are indicated by green circles. The dashed blue box delineates the work done

GMDD
8, 3823–3859, 2015

pre-release by the CESM-software engineers. If a CESM user wants to evaluate a new configuration, the user simply executes the steps in the dashed red box.

4 Experimental studies

As noted in Sect. 1, a verification process necessarily includes some degree of subjectivity. The decision to designate our initial ensemble distribution as "accepted" is critical to our methodology and yet, despite on-going research, we cannot (ever) be absolutely sure that this distribution is "correct" in terms of capturing all signatures that lead to the same climate. Our confidence in this initial ensemble distribution is due, in part, to the vast experience and intuition of the CESM climate scientists. However, we gain further confidence with a series of tests of trusted scenarios (i.e., scenarios that we expect to produce the same climate) and verify that those scenarios pass the CESM-ECT. Similarly, we sample scenarios that we expect to be climate-changing and should, therefore, fail.

4.1 Preliminaries

The results in this work were obtained from the 1.3 release series of CESM, using a present-day F compset (active atmosphere and land, data ocean, and prescribed ice concentration) and CAM5 physics. We examine 120 (out of a possible 134) variables from the CAM history files, as redundant variables and those with no variance are excluded. Of the 120 variables, 78 are two-dimensional and 42 are three-dimensional variables. This spectral-element version of CAM uses a ne= 30 resolution, which corresponds approximately to a 1° global grid, containing a total of 48 602 horizontal gridpoints and 30 vertical levels. Unless otherwise noted, simulations were run with 900 MPI tasks and two OpenMP threads per task on the Yellowstone machine at NCAR. The default compiler on Yellowstone for our CESM version is Intel 13.1.2 with −O2 optimization.

GMDD
8, 3823–3859, 2015

4.2 Non-climate changing modifications

First we look at modifications that lead to non-BFB results but are *not* expected to be climate-changing. Such modifications include equivalent code formulations that result in the reordering in floating-point arithmetic operations, thus affecting the rounding error. Two common CESM configurations that induce reordering in arithmetic operations include removing thread-level parallelism from the model and certain compiler changes. We expect that the following tests on Yellowstone will not be climate-changing, and thus, will be consistent with our initial ensemble distribution:

- *NO-OPT*: changing the Intel compiler flag to remove optimization (−O0)
- *INTEL-15*: changing the Intel compiler version to 15.0.0
- *NO-THRD*: compiling CAM without threading (MPI-only)
- *PGI*: using the CESM-supported PGI compiler (13.0)
- *GNU*: using the CESM-supported GNU compiler (4.8.0)

These five scenarios differ from the control run used to generate the ensemble only in the single aspect listed above. We first generate $N_{new} = 3$ simulations on Yellowstone corresponding to each test scenario, where each simulation is given a perturbation selected at random from the perturbations used to create the initial ensemble. Table 2 lists the pass/fail result from pyCECT and indicates that none of these modifications caused a failure. Recall that our criteria for failure in pyCECT is that at least three PCs must fail at least two of the runs. Table 2 shows that at most two PCs failed two runs for these particular test scenarios.

4.3 CAM climate-changing parameter modifications

CESM-ECT also must successfully detect changes to the simulation results that are known to be climate-changing and return a failure. To this end, climate scientists pro-

GMDD
8, 3823–3859, 2015

vided a list of CAM input parameters thought to affect the climate in a non-trivial manner. Parameter values were modified to be those intended for use with different CAM configurations (e.g. high-resolution, finite volume, etc.). We ran the following test scenarios which were identical to the default ensemble case with the exception of the noted CAM parameter change (the name of the CAM parameter is indicated in italics, and its original default value in parenthesis):

- *DUST*: dust emissions; *dust_emis_fact = 0.45* (0.55)
- *FACTB*: wet deposition of aerosols convection factor; *sol_factb_interstitial = 1.0* (0.1)
- *FACTIC*: wet deposition of aerosols convection factor; *sol_factic_interstitial = 1.0* (0.4)
- *RH-MIN-LOW*: min. relative humidity for low clouds; *cldfrc_rhminl = 0.85* (0.8975)
- *RH-MIN-HIGH*: min. relative humidity for high clouds; *cldfrc_rhminh = 0.9* (0.8)
- *CLDFRC-DP*: deep convection cloud fraction; *cldfrc_dp1 = 0.14* (0.10)
- *UW-SH*: penetrative entrainment efficiency – shallow; *uwschu_rpen = 10.0* (5.0)
- *CONV-LND*: autoconversion over land in deep convection; *zmconv_c0_lnd = 0.0035* (0.0059)
- *CONV-OCN*: autoconversion over ocean in deep convection; *zmconv_c0_ocn = 0.0035* (0.045)
- *NU-P*: hyperviscosity for layer thickness (vertical lagrangian dynamics); *nu_p* = 1.0×10^{-14} (1.0×10^{-15})
- *NU*: dynamics hyperviscosity (horizontal diffusion); nu = 9.0×10^{-14} (1.0×10^{-15})

GMDD
8, 3823–3859, 2015

From Table 1, most of these tests fail by a lot more than 3 PCs, indicating that the new simulation data is quite different from the original ensemble data. However, contrary to our initial expectations, one scenario was found to be consistent and passed. Upon further investigation, the change caused by NU likely did affect some aspects of the climate in a way that would not be detected by the test. The issue is that modifications to NU cause changes to the small-scales (but not to the mean of the field the diffusion is applied to) and generally affect the extremes of climate variables (such as precipitation). Because CESM-ECT looks at variable annual-global means, the "pass" result is not entirely surprising as errors in small-scale behavior are unlikely to be detected in a yearly global mean. Developing the capability to detect the influence of small-scale events is a subject for future work.

4.4 Modifications with unknown outcome

Now we present results for simulations in which we had less confidence in the expected outcome. These include running our default CESM simulation on other CESM-supported machines as well as changing to a higher level of optimization on Yellowstone (−O3). We expected that the tests on other machines supported by CESM would pass, and, for each machine, we list the machine name and location below (and give the processor and compiler type in parentheses). The affect of −O3 compiler options was not known as the CESM codebase is large and level three optimizations can be quite aggressive. The following simulations were performed:

- *HOPPER*: National Energy Research Scientific Computing Center (Cray XE6, PGI)
- *EDISON*: National Energy Research Scientific Computing Center (Cray XC30, Intel)
- *TITAN*: Oakridge National Laboratory (AMD Opteron CPUs, PGI)
- *MIRA*: Argonne National Laboratory (IBM BG/Q system, IBM)

GMDD
8, 3823–3859, 2015

- *JANUS*: University of Colorado (Intel Westmere CPUs, Intel)
- *BLUEWATERS*: University of Illinois (Cray XE6, PGI)
- *EOS*: Oakridge National Laboratory (Cray XC30, Intel)
- *GOLDBACH-INTEL*: NCAR (Intel Xeon CPU cluster, Intel)
- *GOLDBACH-PGI*: NCAR (Intel Xeon CPU cluster, PGI)
- *INTEL13-O3* Yellowstone with default Intel compiler and −O3 option
- *INTEL14-O3* Yellowstone with Intel 14.0.2 compiler and −O3 option
- *INTEL15-O3* Yellowstone with Intel 15.0.0 compiler and −O3 option

Note that we use the CESM-specified default compiler option for each CESM-supported machine. Table 3 indicates that most of the CESM-supported machine configurations pass (the nine test scenarios above the horizontal line), and the few that fail are all near the pass/fail threshold. In other words, these machine failures are in contrast to the more egregious failures obtained by changing CAM parameters as in Table 1. However, ideally all CESM-supported machines would pass our test (assuming the absence of error in their hardware and software environments), and a better understanding of the variability introduced by the environments of other machines (i.e., not Yellowstone) is needed. Therefore, as a first step, we ran additional tests on Mira and Bluewaters with the goal of better understanding (and substantiating) the failures in Table 3. For each machine, we ran 7 more sets of three randomly perturbed simulations. Thus we have a total of 8 experiments each for Mira and Bluewaters, counting the original in Table 3. Furthermore, we created three *additional* ensembles of 151 simulations based on the PGI, GNU, and NO-OPT scenarios listed in Sect. 4.2 and created a summary file for each. Thus, we can test the 8 new cases for consistency on both machines against a total of four ensembles to better understand the effect of the compiler on the consistency assessment. Results from these experiments are shown

GMDD
8, 3823–3859, 2015

for Bluewaters and Mira in Figs. 3 and 4, respectively. Note that the Intel ensemble is the default "accepted" ensemble that we have used thus far in our experiments and the No-Opt option is also the Intel compiler (with −O0).

The results in Figs. 3 and 4 indicate that the compiler choice for the control ensemble on Yellowstone results in differences in the numbers of PC scores that fail each individual test case. However, the overall outcome from all four control ensembles is similar in that the test results are split in terms of passes and fails, indicating that these are in fact borderline cases for CESM-ECT with the current failure criteria, which requires at least 3 PCs to fail at least two runs. Test scenarios that very nearly pass or fail, such as these for Bluewaters and Mira underscore the difficulty in distinguishing a bug in the hardware or software from the natural variability present in the climate system. Certainly we do not expect to perfect CESM-ECT to the point where a pass or fail is a definitive indication of the absence or presence of a problem, though we have obtained a large amount of data to date that we will explore in detail to better characterize the effects of compiler and architecture differences on the variability. We expect to report on our further analysis in future work. Finally, another difficulty for our tool is that while PCA will indicate the existence of different signatures of variability between new simulations and the ensemble, the differences detected may not necessarily be important in terms of the produced climate and the decision on whether to accept or reject that climate (e.g., because the definition of climate requires more than one year and involves spatial distributions).

The last three experiments listed above and in Table 3 involve either modifying the optimization to a more agressive level (INTEL13-O3) or additionally upgrading the compiler version (INTEL14-O3 and INTEL15-O3). Our results for INTEL15-O3 suggest that there is a compiler issue with that version. Note that because of the size of the CESM code base, pinpointing a problem with a specific compiler version is time-intensive, and we find it more productive not to use that compiler.

GMDD
8, 3823–3859, 2015

5 CESM-ECT in practice

CESM-ECT has already been successfully integrated into the CESM software engineering workflow. In particular, the creation of a new beta release tag in the CESM development trunk (that is *not* BFB with the previous tag) requires that CESM-ECT be run for the new tag on all CESM-supported platforms (e.g. the machines listed in Sect. 4.4 and the supported compilers on those platforms (e.g, Intel, GNU and PGI, all with −O2, on Yellowstone). Results from these tests are kept in the CESM testing database. Failure on one or more of the test platforms signals that an error may exist in the new tag or on a particular machine, spawning an investigation and delay of the beta tag release.

CESM-ECT has proven its utility on numerous occasions, and we now provide several specific examples of the success of this consistency testing methodology in practice. The first example concerns an early success for our ensemble-based testing methodology. The consistency test for a CESM.1.2 series beta tag test on the Mira machine failed decisively, while the consistency tests on all other platforms passed. The CESM-ECT failure prompted an extensive investigation of the Mira simulation data which resulted in the discovery that the CAM energy balance was incorrect. Eventually an error was discovered in the stochastic cloud generator code that only manifested itself on big-endian systems (Mira was the only big-endian machine in the group of CESM-supported machines). Because this particular success occurred early in the research and development stages of CESM-ECT (when we were only looking at RMSZ scores), it provided the impetus to move forward and further refine our ensemble-based consistency testing strategy.

A second, more recent success for CESM-ECT was the detection of errors in a new version of the Community Ice Code (CICE). In particular, CICE5 replaced CICE4 in the CESM.1.3 series development trunk, and this upgrade was purported to not change the climate. However, when the software tag with CICE5 was tested with CESM-ECT, failures occurred on all of the CESM-supported platforms. Recall that CESM-ECT uses

GMDD
8, 3823–3859, 2015

an F compset (e.g., Sect. 4.1), which means that CICE runs in prescribed mode. Prescribed mode is intended for atmospheric experiments and uses the thermodynamics in the sea ice model (the dynamics are deactivated) with a pre-specified ice distribution. The CESM-ECT failures for the new development tag raised a red flag that resulted in the detection and correction of a number of errors and necessary tuning parameter changes in CICE5 prescribed mode. Pre-integration component-level testing for standalone CICE, however, allowed errors to go undetected in prescribed mode until run with CESM-ECT. Table 4 lists the results of CESM-ECT for three test scenarios on Yellowstone (Intel, GNU, and PGI compilers) with CICE5 and CICE4, showing that the difference was quite significant.

Finally, CESM-ECT has been essential in the evaluation of lossy compression schemes for CESM climate data. Lossy compression schemes result in data loss when the compressed data is reconstructed (i.e., uncompressed). Evaluating the impact of the loss in precision and/or accuracy in the reconstructed data is critical to the adoption of lossy compression methods in the climate modeling community. In particular, we advocate for compression levels that result in reconstructed data that is not statistically distinguishable from the original data. The CESM ensemble-consistency methodology has been invaluable in making this determination (e.g., Baker et al., 2014).

6 Conclusions and future work

Software quality assurance is critical for building (and retaining) confidence in widely-used scientific codes such as the Community Earth System Model. The size of the code, diversity of both the user and developer base, societal impact, and near-constant state of development for CESM require a verification technique that is easy to use and has minimal computational requirements. Further, the increasing difficulty in achieving BFB identical results due to differences across hardware and software environments dictates that a verification tool determines acceptable error tolerances. This manuscript presents a ensemble-based consistency test that evaluates whether a new CESM con-

GMDD
8, 3823–3859, 2015

figuration (e.g., resulting from a code modification, compiler change, or new hardware platform) is consistent with the original "accepted" (or control) configuration. The original configuration is represented by an ensemble that captures the natural variability in the modeled climate system. CESM-ECT has already been effectively incorporated into the CESM software development workflow. Our many experiments and its successes in practice have increased our confidence in this methodology for detecting and reducing errors in CESM. Furthermore, the utility of CESM-ECT in a number of scenarios has become apparent:

- port-verification (new CESM-supported machines);
- quality assurance for software release tags;
- exploration of new algorithms, solvers, compiler options;
- feedback for model developers;
- detection of errors in the software or hardware environment; and
- assessment of the effects of lossy data compression.

Despite our successes with this new consistency-testing methodology, the natural variability present in the climate system makes the detection of subtle errors in CESM challenging. While no verification tool can be absolutely correct, we consider CESM-ECT in its current form to be preliminary work as many avenues remain to be explored. We are currently conducting a more detailed analysis of large ensembles from different compilers and machines in an attempt to better characterize the effects of those types of perturbations. We have also begun to evaluate spatial patterns in addition to global (spatial) means, as these patterns may be revealing in such contexts as boundaries between ocean and land, and less chaotic systems like the coarse-resolution ocean. In addition, we are interested in other important climate statistics like extremes. Finally, we intend to evaluate relationships between variables in cross-covariance studies.

GMDD
8, 3823–3859, 2015

Code availability

The software tools needed to test for ensemble consistency are included in the CESM public releases beginning with the version 1.4 series, which are available at https://github.com/CESM-Development/cime. Note that the Python testing tools can also be downloaded independently of CESM from the collection of parallel Python tools available on the NCAR's Application Scalability and Performance website (https://www2.cisl.ucar.edu/tdd/asap/application-scalability) or obtained directly from NCAR's public Subversion repository (https://proxy.subversion.ucar.edu/pubasap/pyCECT/tags/1.0.0/). CESM simulation data is available from the corresponding author upon request.

Acknowledgements. This research used computing resources provided by the Climate Simulation Laboratory at NCAR's Computational and Information Systems Laboratory (CISL), sponsored by the National Science Foundation and other agencies. This research also used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the US Department of Energy under Contract No. DE-AC05-00OR22725. This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the US Department of Energy under Contract No. DE-AC02-05CH11231. This work utilized the Janus supercomputer, which is supported by the National Science Foundation (award number CNS-0821794) and the University of Colorado Boulder. The Janus supercomputer is a joint effort of the University of Colorado Boulder, the University of Colorado Denver and the National Center for Atmospheric Research.

References

Baker, A. H., Xu, H., Dennis, J. M., Levy, M. N., Nychka, D., Mickelson, S. A., Edwards, J., Vertenstein, M., and Wegener, A.: A methodology for evaluating the impact of data compression on climate simulation data, in: Proceedings of the 23rd international symposium on High-Performance Parallel and Distributed Computing, HPDC '14, 203–214, 2014. 3847

GMDD

8, 3823–3859, 2015

Carson II, J. S.: Model verification and validation, in: Proceedings of the 2002 Winter Simulation Conference, 52–58, 2002. 3824, 3825, 3826

Clune, T. and Rood, R.: Software testing and verification in climate model development, IEEE Software, 28, 49–55, doi:10.1109/MS.2011.117, 2011. 3827, 3828

Dai, A., Meehl, G., Washington, W., Wigley, T., and Arblaster, J. M.: Ensemble simulation of 21st century climate changes: business as usual vs. CO_2 stabilization, B. Am. Meteorol. Soc., 82, 2377–2388, 2001. 3831

Easterbrook, S. M. and Johns, T. C.: Engineering the software for understanding climate change, Comput. Sci. Eng., 11, 65–74, doi:10.1109/MCSE.2009.193, 2009. 3827, 3828

Easterbrook, S. M., Edwards, P. N., Balaji, V., and Budich, R.: Guest editors' introduction: climate change – science and software, IEEE Software, 28, 32–35, 2011. 3827

Goosse, H., Barriat, P., Lefebvre, W., Loutre, M., and Zunz, V.: Introduction to climate dynamics and climate modeling, available at: http://www.climate.be/textbook (last access: 5 May 2015), 2014. 3825

Hurrell, J. W., Holland, M. M., Gent, P. R., Ghan, S., Kay, J. E., Kushner, P. J., Lamarque, J. F., Large, W. G., Lawrence, D., Lindsay, K., Lipscomb, W. H., Long, M. C., Mahowald, N., Marsh, D. R., Neale, R. B., Rasch, P., Vavrus, S., Vertenstein, M., Bader, D., Collins, W. D., Hack, J. J., Kiehl, J., and Marshall, S.: The Community Earth System Model: a framework for collaborative research, B. Am. Metereol. Soc., 94, 1339–1360, doi:10.1175/BAMS-D-12-00121.1, 2013. 3824

IPCC Data Collection Center 2015, available at: http://www.ipcc-data.org/ (last access: 5 May 2015), 2015. 3826

Kay, J., Deser, C., Phillips, A., A. Mai, A., Hannay, C., Strand, G., Arblaster, J., Bates, S., Danabasoglu, G., Edwards, J., Holland, M., Kushner, P., Lamarque, J.-F., Lawrence, D., Lindsay, K., Middleton, A., Munoz, E., Neale, R., Oleson, K., Polvani, L., and Vertenstein, M.: The Community Earth System Model (CESM) Large Ensemble Project: A Community Resource for Studying Climate Change in the Presence of Internal Climate Variability, B. Am. Meteorol. Soc., doi:10.1175/BAMS-D-13-00255.1, in press, 2015. 3831

Oberkamf, W. and Roy, C.: Verification and Validation in Scientific Computing, Cambridge University Press, Cambridge, 2010. 3824, 3825, 3826

Pipitone, J. and Easterbrook, S.: Assessing climate model software quality: a defect density analysis of three models, Geosci. Model Dev., 5, 1009–1022, doi:10.5194/gmd-5-1009-2012, 2012. 3827, 3828

GMDD

8, 3823–3859, 2015

Rosinski, J. M. and Williamson, D. L.: The accumulation of rounding errors and port validation for global atmospheric models, SIAM J. Sci. Comput., 18, 552–564, doi:10.1137/S1064827594275534, 1997. 3828, 3831

Sansom, P. G., Stephenson, D. B., Ferro, C. A. T., Zappa, G., and Shaffrey, L.: Simple uncertainty frameworks for selecting weighting schemes and interpreting multimodel ensemble climate change experiments, J. Climate, 26, 4017–4037, 2013. 3831

Sargent, R. G.: Verification and Validation of Simulation Models, in: Proceedings of the 2011 Winter Simulation Conference, 183–198, 2011. 3825

Shlens, J.: A Tutorial on Principal Component Analysis, CoRR, abs/1404.1100, available at: http://arxiv.org/abs/1404.1100 (last access: 5 May 2015), 2014. 3834

von Storch, H. and Zwiers, F.: Testing ensembles of climate change scenarios for "statistical significance", Climatic Change, 117, 1–9, 2013. 3831

Whitner, R. B. and Balci, O.: Guidelines for selecting and using simulation model verification techniques, in: Winter Simulation Conference, 559–568, 1989. 3825

Zhu, Y.: Ensemble forecast: a new approach to uncertainty and predictability, Adv. Atmos. Sci., 22, 781–788, 2005. 3831

Zhu, Y. and Toth, Z.: Ensemble based probabilistic forecast verification, in: 19th Conference on Probability and Statistics, Amer. Meteor. Soc., 1–6, 2008. 3831

GMDD

8, 3823–3859, 2015

Table 1. CESM modifications expected to change the climate.

Test name	CESM-ECT Results	Number of PCs failing at least 2 runs
DUST	FAIL	9
FACTB	FAIL	36
FACTIC	FAIL	43
RH-MIN-LOW	FAIL	44
RH-MIN-HIGH	FAIL	30
CLDFRC-DP	FAIL	27
UW-SH	FAIL	24
CONV-LND	FAIL	33
CONV-OCN	FAIL	45
NU-P	FAIL	35
NU	PASS	1

GMDD
8, 3823–3859, 2015

Table 2. CESM modifications expected to produce the same climate.

Test name	CESM-ECT Results	Number of PCs failing at least 2 runs
NO-OPT	PASS	1
INTEL-15	PASS	1
NO-THRD	PASS	0
PGI	PASS	0
GNU	PASS	2

GMDD
8, 3823–3859, 2015

Table 3. CESM modifications with unknown outcomes.

Test name	CESM-ECT Results	Number of PCs failing at least 2 runs
HOPPER	PASS	1
EDISON	PASS	1
TITAN	PASS	0
MIRA	FAIL	5
JANUS	PASS	1
BLUEWATERS	FAIL	5
EOS	FAIL	4
GOLDBACH-INTEL	PASS	0
GOLDBACH-PGI	PASS	0
INTEL13-03	PASS	1
INTEL14-03	PASS	1
INTEL15-03	FAIL	38

GMDD
8, 3823–3859, 2015

Table 4. CESM development tag with two versions of the CICE component run with different compilers on Yellowstone.

Test name	CESM-ECT Results	Number of PCs failing at least 2 runs
CICE4-INTEL	PASS	1
CICE4-GNU	PASS	0
CICE4-PGI	PASS	0
CICE5-INTEL	FAIL	19
CICE5-GNU	FAIL	20
CICE5-PGI	FAIL	19

GMDD
8, 3823–3859, 2015

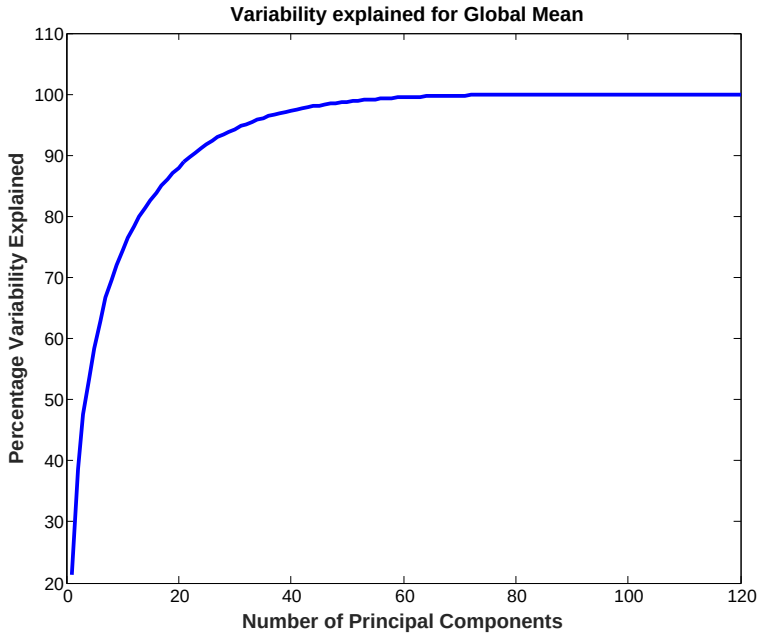

Figure 1. Percentage of variability explained for global mean by component scores.

GMDD
8, 3823–3859, 2015

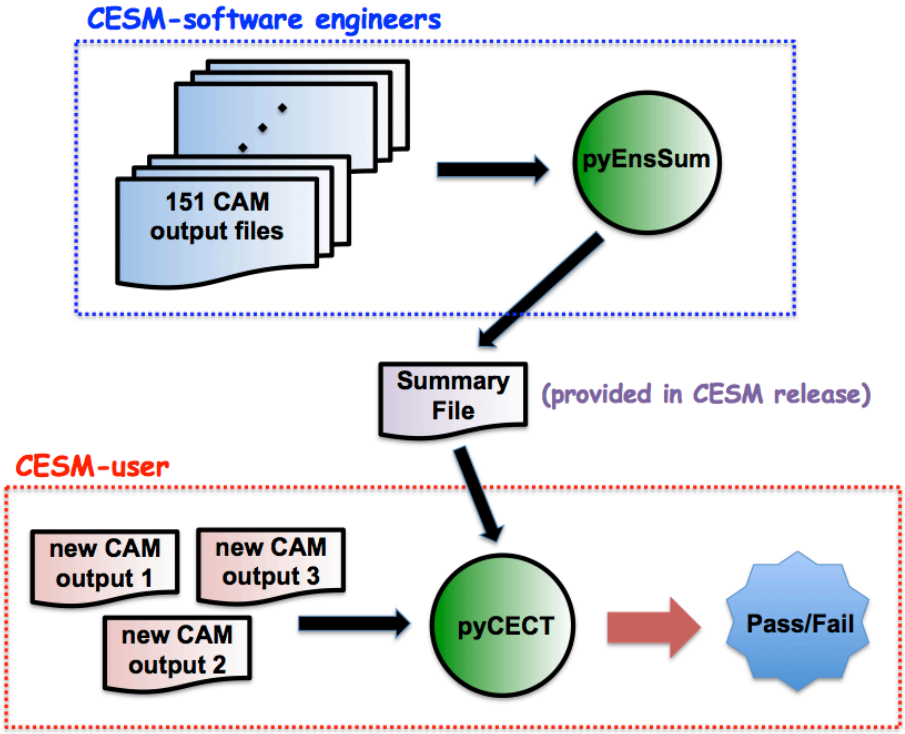

Figure 2. Graphic of CESM-ECT software tools (circles) and workflow.

GMDD
8, 3823–3859, 2015

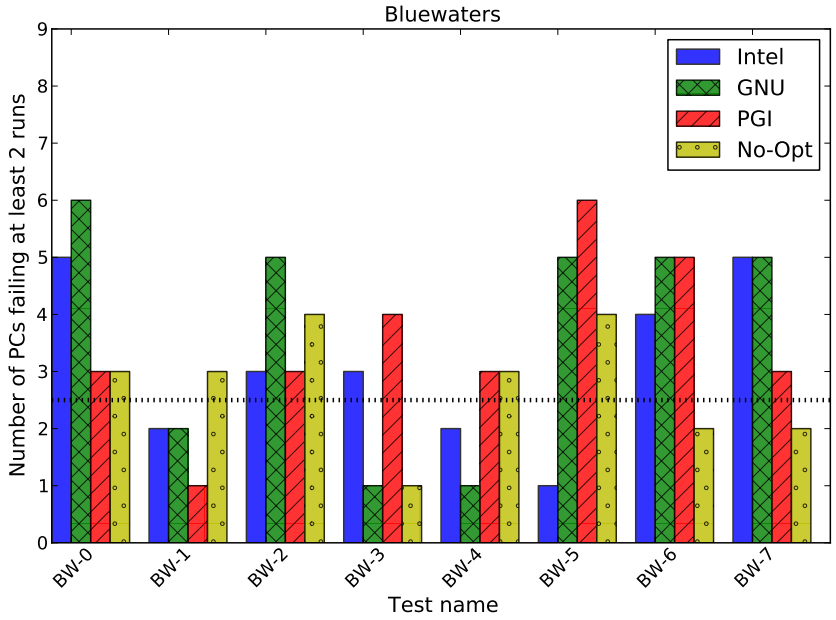

Figure 3. Additional CESM-ECT results on Bluewaters, comparing against four different ensemble distributions. Bars extending above the dashed line indicate an overall failure.

GMDD
8, 3823–3859, 2015

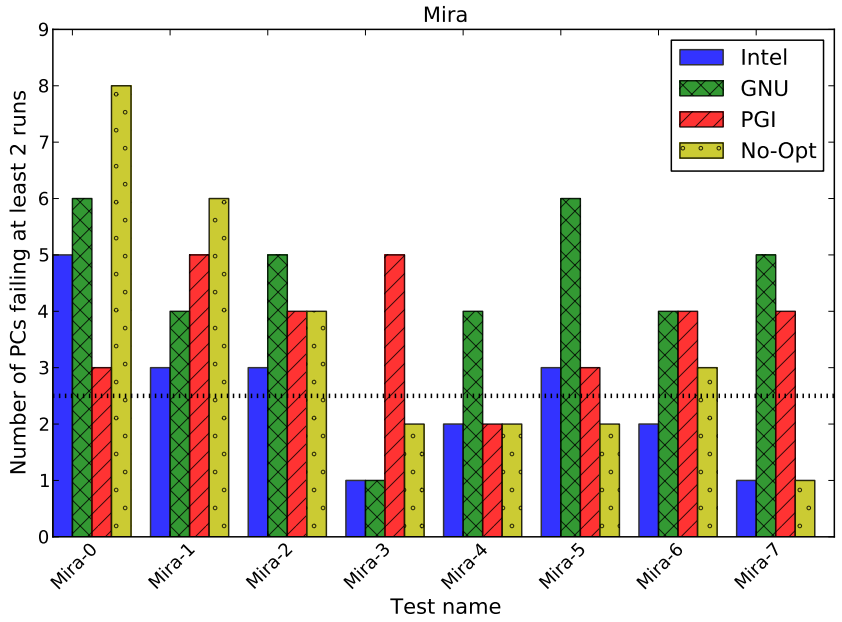

Figure 4. Additional CESM-ECT results on Mira, comparing against four different ensemble distributions. Bars extending above the dashed line indicate an overall failure.

GMDD
8, 3823–3859, 2015