Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2017-185-AC1, 2017

Geoscientific Model Development Discussions
Open Access
EGU

Interactive comment on "Automated model optimisation using the Cylc workflow engine (Cyclops v1.0)" *by* Richard M. Gorman and Hilary J. Oliver

Richard M. Gorman and Hilary J. Oliver

r.gorman@niwa.co.nz

Received and published: 20 November 2017

The reviewer has made some very constructive comments and suggestions, which we believe can be addressed, leading to an improved manuscript.

It is suggested that the paper needs to be expanded by trialling two or three more additional algorithms, and also by including a study starting from a more extreme parameter set. We accept that suggestion, and intend to undertake some further studies to include in the paper. Our aim was to introduce Cyclops as a potentially useful tool to apply any of a wide set of algorithms to optimise a numerical modelling system, in a way that can be quite readily adapted to different modelling systems. That still

Printer-friendly version

Discussion paper

leaves a lot of work to do in designing the modelling system itself (e.g. designing verification metrics, selecting parameters to optimise and choosing the best optimisation algorithm). We did not intend those issues to be a major focus of the paper, and took the view that it would suffice to present one example application for which we made a set of choices that was seen as reasonable if not definitive. But in hindsight, it would help support our aim if we were to show several different optimisation algorithms being applied within Cyclops.

Ideally, if we included at least one truly global algorithm in these further tests, this could go a long way toward clarifying whether we have located a global rather than merely local optimum to our wave hindcast calibration study. In the present version of the paper we merely discussed this issue as possibly needing further investigation. The required computational resources are, however, likely to be prohibitive, so we may need to confine ourselves to local algorithms, and following the suggestion of trying some initialisations far from the default as a simpler way to at least partially address this question. Among local algorithms, we would be amenable to trying the Gauss-Newton approach, but would prefer not to hold up the work to add it to the set of choices available in Cyclops through the NLopt toolbox, if this proves problematic.

The other comments can be addressed through changes in the text. While a revised draft needs to await completion of the further studies noted above, I can make a few points on how we intend to address those comments.

1. The interface between the optimisation algorithm and the model

As noted in the review, the optimisation suite simply outputs a single namelist file. This contains names and values for each parameter, which can be grouped by related sets of parameters. The variable names and allowed ranges are set in a “parameter definition” file that the user prepares. The model suite then needs to include a task that takes this namelist file as one of its inputs, and prepares whatever input files are needed for the model(s) to run. Because the formats are highly model-specific, this task needs to

be tailored for the particular model suite. For example, in our wave hindcast application, this task consists of a shell script which simply includes the namelist file verbatim as part of an ASCII control file, which also has various timing parameters provided from environment variables. In this case, Wavewatch interprets the namelist groups as referring to sets of parameters for different physical processes (e.g. wind input, nonlinear interactions), and we don't need to parse this information in the preprocessing step.

Incidentally, the bold labels in Tables 2-4 refer to these groupings, which was not made sufficiently clear in the paper. Breaking up the tables according to these groupings, as suggested, would also be helpful.

In other cases, some slightly more complicated scripting may be required to generate model inpjut files from the single namelist file. For example, I am presently working on a coupled ADCIRC-SWAN model for hydrodynamic & wave simulations. This model runs as a single coupled executable, but separate SWAN and ADCIRC control files are needed. The cylc suite which controls this coupled simulation includes tasks to prepare those files, consisting of shell scripts using parameter values from environment variables. To apply the optimisation suite to this coupled simulation (which I haven't quite done yet!) will just need to add in a task that parses a single namelist file to set the relevant environmental variables. In that example, there might be "SWAN" and "ADCIRC" namelist groups to make that straightforward. I'm not experienced with HadAM3, but imagine that something similar could be done to prepare the control files it needs. But I guess if the namelist format was inadequate to supply the needed information, we could change that format within the optimisation suite. I should stress that none of this needs any change to the main model codes: they can run as standard release versions under a separate task within the model suite.

When it comes to computing the cost function values, again the details are up to the model suite, but communication with the optimisation suite is very simple: some task within the model suite needs to write that single number to a file, which the optimisation suite reads when that particular implementation of the model suite has completed. The

optimisation suite then appends that value, along with the corresponding parameter values, to a simple ASCII file which serves as the "lookup table".

Really, from the optimisation suite's point of view, the model suite is just a black box that takes a namelist file with parameter names and values, written to a specified path relative to a new directory created for each iteration of the model suite, and computes a single objective function value which it writes to another specified path within that directory.

2. Concurrency

The description in Section 2.3 of how Cyclops can run several iterations of the model suite concurrently is perhaps not as clear as it could be. As noted, whether or not concurrent simulations can be run at any particular stage of the optimisation process does indeed depend on the particular algorithm being used. Also, any particular execution of the optimisation algorithm is done in a purely serial way.

Because of the particular generic "user supplied objective function" subroutine we have implemented, any run of the optimisation task ("optimise_step") simply amounts to reading previous results from the lookup table file and deciding, based on those results, what parameter set the particular algorithm would call for next. At that point it stops, and, unless convergence has been reached, those parameter values are sent off to our model "black box", and eventually the lookup table will be appended with the results. So the next time the optimisation task is run, it will get one step further in the iteration sequence.

Now instead of just waiting for the latest model run to finish (in maybe an hour or more), we might wonder if the answer that comes back will make any difference to the next parameter set the optimisation engine will call for. Rather than rely on knowing any details on how each algorithm works, we can decide that empirically, running the optimisation process several times with randomly generated "answers" from the simulation still in progress appended to the lookup table. If it still calls for exactly the same choice

of parameters each time, clearly there is no need to wait for the actual value to be computed, as we already know what simulation needs to be run, we can start it straight away.

Now once we have more than one simulation running, we need to know if the results of any of those runs will affect the selection of the next parameter set. So we keep track of all the parameter values which are currently being worked on, and do randomised tests on sensitivity to those results.

3. Effect of noise in the optimisation algorithm, and choice of cost function

Tett et al (2017) point out that the inherently chaotic nature of the climate system means that a certain level of noise is introduced into evaluations of an atmospheric model simulation, which can cause problems in evaluating the termination criteria. They describe a procedure to rerun a simulation that had nominally satisfied the prescribed convergence criteria, with randomised perturbations before determining whether or not to terminate.

Unlike the atmosphere, ocean surface waves are an essentially dissipative system, and perturbations introduced in the initial conditions and forcing will tend to diminish, rather than grow, with time. As a result, noise in the objective function was not so relevant for our wave hindcast application as for atmospheric models. Nevertheless, we can envision Cyclops being applied to optimisation of an atmospheric model, or some other system with un underlying chaotic nature. So we will add a comment on this issue, suggesting that a measure such as that described by Tett et al (2017) could be introduced into Cyclops for use in such applications. Other references to Tett et al. (2017), which was published subsequent to our manuscript submission, will be added as appropriate e.g. in addition to our reference to Roach et al (2017).

Similarly, the dissipative nature of ocean waves means that a cost function based on a spatial average of the (temporal) RMSE of model-data comparisons will not be subject to the level of chaotic variability seen in similar measures for atmospheric models.

Small scale variability in wave model output is therefore more likely to be genuinely sensitive to parameter variation. In that case it is worth capturing such variability in the cost function, whereas for a chaotic system it may be wiser to average out such variability before evaluating the cost function. Again, we agree it would be helpful to mention this issue in the context that applications to different model systems may require variations in approach.

4. Task interaction

The way that task interaction is handled could be better described. This function is inherent to Cylc, so belongs in its description (Section 2.1). Essentially, Cylc maintains a database for each suite, keeping track of the status of each cycle of each task within the suite. "Status" includes whether the task has started, is running, is stopped, and any messages that the task has sent. Tasks within a Cylc suite can interrogate the suite database, and the databases of other running suites, as configured when task dependencies are defined.

Incidentally, it would perhaps also be useful for us to say a little more about the nomenclature, pointing out the subtle spelling differences. "Cylc" (pronounced "silk") was named partly in a slightly modified reference to the "cycle" concept. "Cyclops" is a Cylc-based Optimisation Suite, but we switched the letters back to sound a little more familiar!

5. Selection of convergence criteria

In the convergence criteria, the "change in parameter values" means the magnitude of the vector difference, i.e. $\sqrt{(\sum\_(n = 1)N(\Delta x\_n)\hat{}2}$). Section 2.2 will be changed to clarify this. The choice of 0.02 was somewhat arbitrary initial choice. As we noted in Section 3.3, the suite can be restarted with revised criteria after stopping, either having met an initial set of convergence criteria, or through manual intervention. So in practice, you can start with quite loose criteria, then either decide to carry on further with tighter criteria, or reconfigure something and start again. For the two three-month

Printer-friendly version

Discussion paper

hindcasts, the aim was to explore differences between the two model configurations, and guide the choice of settings to use for the more thorough 12 month optimisation. For that purpose, the results reached with the 0.02 criterion may have been sufficient, but perhaps it would be better serve the paper to plot the results with an extended iteration under tighter criteria.

6. Tables

The parameters in Table 2 are described in the Appendix, but a reference to this needs to be made in the Table caption. The “n” values were retained in the Tables with the intention of providing a key to the Figures. A clearer alternative way to identify which variable is which in the Figures will be investigated.

Parameters with fixed, default values were added to the Tables for completeness (for other wave modellers who might ask “what value did you use for X?”), but they could indeed be better given elsewhere, if at all.

7. Figures

Figure 1 was generated with software that forms part of the Cylc GUI. It should be straightforward to redo these with improved fonts and colouring. Arrows represent task dependencies, as explicitly specified in defining the suite, so redrawing them to loop back to the start of a cycle would violate this meaning. In Cylc, each task at each cycle can be considered independently, and can run as soon as those explicit dependencies are met. This can allow tasks from one cycle to run simultaneously with tasks from other cycles if their explicit dependencies allow it. This is a feature we exploit in allowing for concurrent model simulations. Actually, a clearer version of Figure 1 could help illustrate that discussion. So my preference is to do that, with other software if necessary, and retain the meaning of the arrows.

8. Minor comments not addressed above

L21, P7 L21: Inconsistencies in referencing the Wavewatch model will be addressed,

in line with the model's licensing terms.

P5, L14. Agreed, it would be more robust and "pythonic" to return "None" than -1 in such cases, so I will change the code before any applications where it could matter.

P8, L2. Yes, dot means d/dt. We will clarify this in the text and improve the equation layout

P8, L35. We will change "zero" to "negligible"

P8 We will rewrite this section in a hopefully clearer manner

P9, L11. The two ice parameters might, a priori, be expected to have more influence than some of the other parameters that were already included, which indeed turned out to be the case. In hindsight they should have been included from the start.

P9, L24. More discussion of parameter sensitivity will be added, using the Delta parameter in Table 4

Other points are minor edits which will be implemented as suggested.

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-185, 2017.

Printer-friendly version

Discussion paper