## BON in a Box – Pipeline Peer Review Form

### Pipeline Name:

### Reviewer Name:

### Date:

### 1. Methods evaluation

**Objective**: Evaluate whether the methodology is scientifically sound, well-documented, and understandable to the user.

* **Evaluation of the method** – Does the method follow an established protocol that has been peer reviewed and published or verified in some other way? If it is a GBF indicator, does it follow the methods laid out in the [GBF indicator metadata](https://www.gbf-indicators.org/)?
* **Novelty of the method** – Does the pipeline add a new or improved method to the pipeline toolbox or is it redundant with a pipeline already present within BON in a Box? If there is overlap between methods, are the reasons and situations in which to choose one method over another clearly communicated?
* **Assumptions and limitations** – Are the assumptions and limitations of the models clearly communicated? If applicable, are there measures of uncertainty?
* **Data types** – Does the pipeline use data that is appropriate for the method? Are the limitations of this data stated? If there is an option to input custom data, are the guidelines for the types and format clearly communicated?

* **Transparency of methodology** – Does the pipeline communicate the methodology so that the average user can understand what is happening within the analysis? Are there guidelines for when and in what context to use the analysis?

**Comments and Suggestions**:

### 2. Code Quality and Accuracy of Implementation

**Objective**: Assess whether the code implements the methods correctly, is reproducible, can run with different geographical areas and parameters, and meets the minimum standards outlined in the pipeline standards document.

* **Method implementation** – Does the code correctly implement the cited methods, free of mistakes? Do the results make sense?
* **Generalizability** – Is the analysis pipeline generalizable to different areas, species, scales of analysis, etc. or is it specific to one use case?
* **Pipeline** – Is the full analysis pipeline assembled correctly? Are the necessary outputs of one script connected to the next script? Is the pipeline divided into logical steps (scripts)?
* **Documentation**
  + **Scripts** –Scripts can be found in the “scripts” folder of the cloned BON in a Box repository. The pull request on GitHub can help identify the new or modified scripts. Within the scripts, is the code well commented and understandable? Are there print messages within the code so the user knows what steps of the code are running from the logs? Does it meet the [code standards](https://docs.google.com/document/u/0/d/1YjEMoB5DaMFVChreUa1CRzDWuHk1dMf9fLaspGg-0tY/edit)?
  + **Script metadata** – The script metadata YAML can be found in the UI by clicking on the individual scripts on the left sidebar of the pipeline editor, or directly in the YAML description files in the “scripts” folder.

Are the YAML description files for each script well documented? Are all fields filled out? Is the description of the script sufficient for understanding what it does? Are the inputs and outputs well described so the user can understand how to parameterize the model? Are all packages that are used cited? Are all methods used cited? Does it meet the YAML file requirements in the [pipeline standards](https://docs.google.com/document/u/0/d/1YjEMoB5DaMFVChreUa1CRzDWuHk1dMf9fLaspGg-0tY/edit)?

* + **Pipeline metadata** – Pipeline metadata can be found in the “Run a pipeline” tab by choosing the pipeline to review from the dropdown list. It can also be found in YAML format in the “metadata” and “inputs, outputs” tabs on the right side of the pipeline editor, when the pipeline is loaded.

Is the pipeline well described for a non-expert user to understand what it does? Are all of the metadata sections (name, description, author(s), license, citations) filled out? Are all of the citations in each individual script plus any additional relevant citations in the metadata? Is the pipeline appropriately named in a descriptive way? Are the inputs and outputs described in a way that helps the user parameterize the pipeline?

* + **Markdown tutorial** – Is the pipeline accompanied by a markdown (.md) tutorial that walks the user through how the pipeline works, how to use it, and shows example inputs and outputs?

* **Usability** – Can the pipeline be run successfully with the provided instructions? Are there clear instructions for how to parameterize the models? Are the inputs of the pipeline clear? Can the pipeline be parameterized and run by a non-expert? Are the outputs described in a way that helps interpret its results?
* **Data handling**
  + **Input data** – For publicly available data, does the code use APIs (if available) to make sure the user is running with the most up-to-date data available? For user input data, does it meet the “Standards for user defined input data” in the [pipeline standards](https://docs.google.com/document/u/0/d/1YjEMoB5DaMFVChreUa1CRzDWuHk1dMf9fLaspGg-0tY/edit)?
  + **Output data** – Do the output data formats meet the [pipeline standards](https://docs.google.com/document/u/0/d/1YjEMoB5DaMFVChreUa1CRzDWuHk1dMf9fLaspGg-0tY/edit)?
* **Error handling and robustness** – Does the code handle common errors with informative messages using the biab\_error\_stop() message? If not, what were some of the errors received that were difficult to troubleshoot?

**Test results**: Test the generalizability of the pipeline with different parameters. Identify whether the pipelines work, whether the results make sense, and if there are any limitations that should be clearly communicated.

Evaluate generalizability of the pipeline across space, in several different countries and sub-national entities if applicable. Test with different input parameters to make sure the pipeline runs and the results make sense.

For each test run, copy the YML file associated with the run and attach to the end of this document, with a note about whether it ran successfully. This will allow us to see the combinations of parameters that were used for each run. You can find the input YAML by clicking on the “Input yaml” next to the “input form” tab on the “Pipeline run” user interface.

* Number of test runs: \_\_\_
* Number of successful runs: \_\_\_
* Number of failures or anomalies (incorrect results): \_\_\_

Details:

**Comments and Suggestions**:

### 

### 

### Final Assessment

* **Overall score (1–5) for Scientific Rigor**: \_\_\_
* **Overall score (1–5) for Code and Implementation**: \_\_\_

**Would you recommend this pipeline for inclusion in BON in a Box?**

* Yes, without modifications
* Yes, with minor revisions
* No, major revisions needed

**General Comments**:

**YAML files of test runs:**