JOSS Review for stitches, pulled into one place.

https://github.com/openjournals/joss-reviews/issues/5525

- Comments in the review checklists, separate review comments in the github discussion, a PR and several issues in the stitches github
- All revisions have been merged to `main` and a new release has been performed (https://github.com/JGCRI/stitches/releases/tag/v0.11.0).

Section 1 - Paper comments

- agree that summary and statement of need are backwards
- Author Response: Thank you! We have made this change

- The summary could be made clearer for non-specialist audiences, but it is there. (As a suggestion, the summary could say something more like, "There is a need to inspect the interaction between climate change and impacts. At the moment this isn't possible with our most expensive tools. This package provides a way to build that link and examine the interaction without the computational cost. In a scientific paper, it has been shown that this method does not come with unworkably large errors [cite Tebaldi paper])"
- Author Response: Thank you! We have made edits to the summary (previously statement of need)

- Also there and states needs very clearly. References to other work could be better explained and fleshed out I think, particularly discussion of why other tools (e.g. MESMER, METEOR I would guess?) don't achieve what STITCHES does. A clearer reference to the full scientific explanation in this section would also be helpful I think as that would be what scientific readers need to fully understand how the tool works.
- Author Response: Thank you! We have added text in an attempt to address this point. We have pointed only to MESMER; I am embarrassed to admit that I could not find a citation of published work for METEOR to point to. I will happily include a citation if someone can please point me.

- don't know of any other package which attempts to do such stitching. This package does quite a lot of ESGF data handling so it could be compared to other packages which do such handling and processing such as ESMValTool, but that could also be out of scope.
- Author Response: We appreciate this note but it does seem out of scope, as ESMValTool is aiming to accomplish such a different thing to `stitches`.

- Summary and Statement of Need appear to be mislabelled.
- Author Response: Thank you! We have made this change

- Summary (Statement of Need) is relatively clear and highlights well how stitches is unique in its approaches.

- Author Response: Thank you!

- Summary (Statement of Need) could be shortened and made more accessible to general audiences.
- Author Response: Thank you! We have attempted to do so.

- Statement of Need (Summary) covers the value of stitches within the field of impact analysis.
- Author Response: Thank you!

- The value of output from stitches to help in impact modelling is made clear, but the intended audience is only explicitly stated in the final paragraph. Should be stated earlier.
- Author Response: Thank you! We have moved this earlier.

- Stitches builds on existing science frameworks with a unique approach, but could be clearer in stating the relative strengths/weaknesses of its approach versus others (speed? versatility? portability? statistical/physical consistency?)
- We have adjusted text to more explicitly compare to the other most relevant package currently available, MESMER.

- I'm not familiar with python packages that aid in generating new variable curves from ESM runs via stitched model data, but it would be good to briefly talk about others that may exist in the scientific Python community (I will search around as well).
- To our knowledge, there aren't any that generate new curves via stitching. MESMER is the closest model available in terms of outcome, but they take a fundamentally different approach and to our knowledge are not presently producing outputs of multiple variables jointly.

- It would be interesting to mention the unique value of stitches in scenario development as compared to multimodel ensemble statistics.
- Author Response: Thank you, we have added text covering this in the final paragraph.

- Other packages are not explored. Mention is made to PANGEO and PANGEO-backed software (xarray), but no comparisons are made to other scenario-building packages.
- Thank you, we have tried to be more explicit in our text where comparisons do occur, and we have added some additional comparisons.

- paper could benefit from some editing; some repetitive statements, some redundant phrases could be removed.
- Author Response: We have attempted to do so

- A few statements require citations; "While many existing ESM emulation methods rely on 'bottom up' methods", "Research from the climate science community has indicated that

many ESM output variables are tightly dependent upon the GSAT trajectory and thus scenario independent", "emulators trained with bottom- up methods often can only handle a small number of variables jointly (e.g. temperature and precipitation)", etc.
- Thank you, we have clarified text and added citations

- The description of the library could benefit from an explanation of the data structure or a brief summary of the capabilities or core functionality.
- Author Response: Thank you! We have tried to do so

- An example of the expected outputs from the tool (e.g. Quickstarter examples) would help showcase the capabilities of stitches.
- Author Response: Thank you! A quickstarter has been available since submission as noted in the paper draft. It is present in the `notebooks` directory of the `stitches` repository (https://github.com/JGCRI/stitches/blob/main/notebooks/stitches-quickstart.ipynb) and its contents are also automatically reproduced in the `stitches` website (https://jgcri.github.io/stitches/), accessible from the github landing page. However, since submitting the repository to JOSS, we actually prepared extensive training materials for a workshop our institute hosted:
    1. https://github.com/JGCRI/stitches/blob/main/notebooks/stitches_training_GCAMAnnualMeeting2023.ipynb - This has actually been used as the basis to update the quickstarter. It preserves many of the content pieces both reviewers mentioned about the original notebook, and we have followed the suggestion in one of the reviewer issues to provide more detail on how the archive data (which is the `stitches` package data) was arrived at, and we have hopefully done a better job in the revised quickstarter of showcasing the expected outputs and capabilities.
    2. https://github.com/JGCRI/stitches/blob/main/notebooks/stitches_takehome_GCAMAnnualMeeting2023.ipynb

Code-related Reviews

We have also clarified in the JOSS paper draft that STITCHES is intended for specific use cases with CMIP6/ScenarioMIP experiments as outlined in the ESD scientifically focused paper. And we have added a small amount of text to emphasize that CMIP6/ScenarioMIP is a widely used product in international efforts to support our claims that the code has general use for the scientific community that regularly interacts with that data.

Section 2- Review Comments from the github discussion

https://github.com/openjournals/joss-reviews/issues/5525#issuecomment-1625193014:

- In general, I think I could use the package without too much trouble but I would really struggle to extend it beyond its main use case. The major reason is that many of the internal data is built around pandas data frames, but it wasn't clear to me where I should look to understand what sort of form these data frames should take or what the data in them should be. This may require a separate section on the various 'models' used in the code (e.g. https://pyam-iamc.readthedocs.io/en/stable/data.html). Such sections can be a bit painful to write and maintain, but they are generally very helpful for helping new users and maintainers to understand how the system works. In particular, it wasn't clear to me what the recipe should look like nor the archive data (I could sort of guess from the examples, but I am not sure I would guess correctly so explicit docs could be very helpful). The other option would be to use something like pandera to add more structure to the data frames and communicate more explicitly what each column should be and means. Each option comes with pro's and con's, but I think I would find it very hard to build a mental model of the package as it is currently written and documented.
- Thank you for raising these ideas! As noted in the JOSS paper draft, `stitches` is designed for use with CMIP6/ScenarioMIP experiments, for data that has been hosted on pangeo (which encompasses the majority of CMIP6/ScenarioMIP data currently available).We have revised the manuscript to reflect this more explicitly.

  Your suggestions and ideas have been captured in an enhancement issue in the stitches repository (https://github.com/JGCRI/stitches/issues/82). We are in the process of implementing this for cmiply held CMIP6-structured data for a future release. In general, this will be a less-supported use-case, as a portion of the code in `stitches` is addressing the various modeling-center-specific quirks of the CMIP6 data. Requiring the use of pangeo-hosted CMIP6 data helped cut down on the number of edge cases in files we had to address. We can't guarantee that `stitches` code will work on locally held CMIP6 data, as users may have made changes to some portions of the file, even if the core information held in the local NetCDF is the same. Support for CMIP5 or CORDEX style files is out of scope for the project that funds `stitches`. Updates for CMIP7 will be made when data become available, which we assume will highlight places we can generalize some of the `stitches` code.

  The package functions stitches.make_matching_archive() and stitches.make_recipe() take care of formatting for the pangeo-hosted CMIP6/ScenarioMIP data we do support.
  We do support custom target GSAT time series outside of the available CMIP6/ScenarioMIP archive (such as from Magicc or Hector or FAIR), and the revised quickstarter provides more clarity on processing target time series for matching with stitches. The previous quickstart is retained, as the `stitches-barebones-quickstart` notebook. At the very least, it seemed a useful reference point for the reviewers to have a side by side with the updated quickstart.

Arising from the above, there were a few areas where the functionality of the package wasn't super clear to me. I think adding a section like the 'model' section above and/or refactoring would make it much clearer what is going on. The issues were:

- why are 'ensemble', 'experiment' and 'model' needed as part of the target_data? Could this not work just based on grouping by everything except the key columns of variable, year and value?
    - Author response: We chose to keep these columns intentionally - a user can overwrite them to contain any information they like but they're columns whose entries are used in forming the ID of the stitched files. When targeting all ensemble members of an experiment, knowing which stitched output was formed by targeting which specific ensemble member has proven useful for us.

- In addition, shouldn't there be some units in target_data and perhaps also a reference period? Using a different reference period from that which was used to create the archive in the first place could cause havoc no?
    - Author response: This is an excellent idea and we have added these columns to our data. Because the anomaly relative to a reference period is only used in the code for GSAT related quantities (such as the matching), it is also something a user can easily shift in pre or post-processing if their GSAT data is relative to a different reference period. All of the stitched, gridded products that `stitches` generates as netcdfs are *not* in terms of anomaly, they are the actual values held in the original ESM netcdf files hosted on Pangeo. We have also added notes to this effect in the quickstart (under 'Install the Package data from Zenodo').

- Is the pangeo table hard-coded? This is probably fine for CMIP6, but seems problematic as we rapidly move beyond CMIP6 (perhaps this can be left for future work, but it seemed a bit odd to me to not give users a way to use a different archive if they want)
    - Author response: The pangeo table can be called from pangeo at any time for updates via a call to stitches.fx_pangeo.fetch_pangeo_table(). It is a data table that Pangeo hosts, cataloging features of available data and file locations. In practice, there are few updates to the data, so we ship a copy with the package data for the version of the package.
      Any future era of data that is hosted on Pangeo will hopefully have a similar structure but we will certainly adjust this function to pull a file for future eras, once available. The code does expect field names as they presently appear in the pangeo table format. We have also added a note to this effect in the quickstart (under 'Install the Package data from Zenodo').

- Is the historical/scenario split also intentionally hard-coded to know about the SSPs? This also seems like it could be problematic if anyone wanted to use STITCHES in a different context or after CMIP6 (or before, e.g. CMIP5).
    - While we tried to be general, it is most likely parts of `stitches` are hard coded based on the CMIP6/ScenarioMIP experiment names because those are the only experiment types STITCHES is intended for use with and validated for.
      At the very least, 'historical' is an experiment that gets appended into every SSP to create the full time series

https://github.com/JGCRI/stitches/blob/bee5fa147b16c2115c8e2cc37cf5ff162abbf754/stitches/make_tas_archive.py#L141).
There is one function that is hardcoded for 2014 vs 2015
https://github.com/JGCRI/stitches/blob/bee5fa147b16c2115c8e2cc37cf5ff162abbf754/stitches/fx_recipe.py#L460 .
Unfortunately, it is out of scope for our supporting project to make adjustments to be back-compatible with CMIP5. We will offer support for CMIP7 when data becomes available, and we will assume that this will highlight places we can generalize the `stitches` code. CMIP5 files may be supported at that time.

- Does the stitching only pull data from one model or can you end up with stitching that joins together windows/samples from multiple different models (e.g. CanESM5 and NASA-GISS output)? Reading the diagram, I think the answer is you can have more than one model but looking at the code, it wasn't super clear to me (e.g. this line csv_to_load = [file for file in all_files if (model in file)][0] in gmat_stitching confused me, maybe the stitching_id handles this?)
    - Author Response: No, you cannot mix ESMs, and we do not believe it makes scientific sense to. The first step in the diagram is 'Choose ESM' singular, and the immediate text after the diagram in the webpage and quickstart are:

      "To use stitches, there are a number of decisions users have to make, perhaps the most important being:

      - Which ESM will stitches emulate?
      - What scenario will be the target of the emulation?
      - Which available CMIP6 experiments, from the ESM to be emulated, will constitute the archive, i.e., the building blocks that stitches will use to construct the target scenario?"

      We have added a clarification to the quickstart additionally.

- **In general, the repo would also benefit from the use of some other tools e.g. code linters and auto-formatters. They would make the code much more readable and introduce very little cost (at least in my opinon).**

https://github.com/openjournals/joss-reviews/issues/5525#issuecomment-1625663505:
(Hi all, I've been meaning to open some issues/PRs in the repo (and will when I find a minute), but I wanted to piggyback on Zeb's great comments here)

In general, I think I could use the package without too much trouble but I would really struggle to extend it beyond its main use case. The major reason is that many of the internal data is built around pandas data frames, but it wasn't clear to me where I should look to understand what sort of form these data frames should take or what the data in them should be. This may require a separate section on the various 'models' used in the code (e.g.

https://pyam-iamc.readthedocs.io/en/stable/data.html). Such sections can be a bit painful to write and maintain, but they are generally very helpful for helping new users and maintainers to understand how the system works.

- I am very much in agreement with this. Having worked with CORDEX/CMIP5/CMIP6 data for many years, I also felt it wasn't entirely clear how I could format my local NetCDF collections for use in STITCHES. A data model would provide a first step towards implementing methods for generalizing use-cases.
- Author Response: This is an excellent point and has been captured in an issue for future releases (https://github.com/JGCRI/stitches/issues/82). Please see our reply to a similar comment above.
  In the paper, in the Statement of Need (now Summary) section, we have added further text to more explicitly highlight that `stitches` is built specifically to interact with pangeo-hosted CMIP6/ScenarioMIP data, as well as highlighting the broad use of CMIP6/ScenarioMIP data to support the case that `stitches` has broad use cases, even restricted only to CMIP6/ScenarioMIP.
  We have clarified in the quickstart that the only required data that must be downloaded for `stitches` to work is downloaded in 5-10 minutes, unless the CMIP6 data on pangeo has had an update (rare).

- I mentioned in my review that xarray and intake are used to fetch data by STITCHES and I had found it odd that efforts were done to convert their Dataset format to CSVs and Pandas DataFrames. The xarray format is quite robust/extensible (https://docs.xarray.dev/en/stable/user-guide/data-structures.html) and preserves the metadata fetched from Pangeo. With tools intake and dask, it allows for methods of structuring subsetted requests for data in advance of GET requests, significantly reducing download time/memory requirements. I'm not familiar with Pandera, but it seems to have integrated dask support as well. Adherence to a standardized data structure would be essential to making the project much more portable.
- Thank you for raising this suggestion! I have included it in an issue (https://github.com/JGCRI/stitches/issues/82) as a potential future optimization. Currently, we find the speeds in our code to be adequate for our purposes, and we do not make specific functionality or performance claims that we fail to meet due to using DataFrames. Operating on csvs and DataFrames actually widens the use cases for `stitches` as well:
  We use xarray for calculating GSAT time series from the raw netcdfs hosted on pangeo, of course, see stitches.fx_make_tas_archive.get_global_tas(). This step is, however, only ever done once upon an initial setup of `stitches`, as it is called within the stitches.make_tas_archive() function to process raw pangeo netcdfs into GSAT anomaly time series that are much slimmer and used for matching. More details are provided below in the comment about the run time on the stitches.make_tas_archive() function , which IS long.
  Because most of the internal functions in stitches (the matching, drawing permutations of

recipes) are based on GSAT time series rather than gridded quantities, operating on csvs and DataFrames actually improves the use cases stitches can be used on, allowing integration with results of SCMs for novel GSAT scenarios that are interior to but not part of the ScenarioMIP design. These novel GSAT scenarios are likely to come from simple climate models such as Hector (our in-house SCM that we prioritize compatibility with), or Magicc or FAIR. Hector at least outputs its time series of global average temperature anomalies as csv files rather than netcdfs. I believe at least Magicc-live does as well.
For a little more detail:, the recipe-related functions in `stitches` are not simply operating on GSAT time series. They operate on 'chunked' GSAT time series that we have processed into windows of time and characterized with the median temperature and rate of change of temperature in that window (more details on this in our revised quickstart). We certainly could structure this as an xarray dataset with two variables` (median temperature and rate of change) and the time windows as custom time variable labels, but to us that was overkill.
There **is** an opportunity to rely more heavily on xarray functions in our gridded stitching process to speed that up and we have opened an issue for future improvements to include this (https://github.com/JGCRI/stitches/issues/82). However, this is not currently a prohibitive time slowdown that causes the `stitches` package to fail to meet its functionality and performance claims.

In general, the repo would also benefit from the use of some other tools e.g. code linters and auto-formatters. They would make the code much more readable and introduce very little cost (at least in my opinon).

- In my work, I do a fair amount of package maintenance and coding standards enforcement, and would be willing to give a hand in this way. If my schedule allows for it, I'd be more than happy to open a PR.
- Thank you! Your PR (https://github.com/JGCRI/stitches/pull/81) has been merged into the package.

Section 3 - Stitches Github Issues:

In checklist code comments:

- ☑ ~~https://github.com/JGCRI/stitches/issues/80 : tests~~
  - Replied in issue
- ☑ ~~https://github.com/JGCRI/stitches/issues/79 : community guidelines~~
  - Replied in issue
- ☑ ~~https://github.com/JGCRI/stitches/issues/78 : notebooks~~
  - Thank you for your suggestions! We have cleaned up the notebooks directory and added the additional training materials we developed after submitting to JOSS. The quickstart preserves many of the content pieces both reviewers

mentioned and we have followed the suggestion in this issue to provide more detail on how the archive data was arrived at.

- ☑ ~~https://github.com/JGCRI/stitches/issues/77 : read me~~
  - Replied in issue
- ☑ ~~https://github.com/JGCRI/stitches/issues/76 : installation~~
  - Replied in issue

Section 4 - non-paper comments raised in the JOSS issue review checklists:

https://github.com/openjournals/joss-reviews/issues/5525#issuecomment-1584832423 :

- Functionality: Have the functional claims of the software been confirmed?
  - The quick-start examples can be reproduced locally on Linux. Code logic assumes POSIX environment (no Windows support).
  - Generating new data does not seem to be feasible (stitches.make_tas_archive) as fetch calls to pangeo are not "lazy" (estimated ~8h to collect data values on a reasonably fast connection; memory requirements are probably significant).
  - Thank you for highlighting this! We do offer a pre-generated data option that bypasses this part which would be what I assume most users would use; we have clarified in our README.
    To be clear, this step IS very slow; it was about 2-4 hours to completely rebuild the package data with this step on my home wifi. But it is also a step that only must be run once when setting up a `stitches` installation *with pangeo-hosted CMIP6/ScenarioMIP data that has been updated since the last time a user set up `stitches`* (which is rare). It is not necessary to run every time a new realization of emulated ESM data is generated by `stitches`. Note, we re-built the package data as part of this review process to address another comment that we add units columns, but there was no change in the data otherwise.
    In other words, stitches.make_tas_archive() is not a piece of code generating new outputs from stitches, it is a data processing step for setting up the collection of available matching data. It only ever has to be run once for the pangeo CMIP6/ScenarioMIP data that `stitches` is designed to work with. New data isn't published very often, so this is almost never called in the practice of actually using or even setting up `stitches`. Therefore, we ship a version of the required package data that is output by stitches.make_tas_archive() that is not large and is easily fetched in 5-10 minutes of download time with stitches.install_package_data(), with more details now added to the quickstarter. That 5-10 minutes itself is also only necessary to do once when setting up stitches, not every time one wants to generate new realizations of gridded climate data. We have also added text to the revised quickstart to clarify more of this. Thank you for highlighting a place to improve clarity!

- Stitches largely operates with direct calls to values (loading operations) and uses pandas DataFrames for internal logic before converting back to xarray/NetCDF. Xarray is used for fetching data, and nearly all operations are xarray-compatible. Why not leverage xarray/intake data stores to reduce computational/memory/bandwidth load?
- See above in our notes about using DataFrames rather than xarray in some places.

- Data values are loaded early and/or copied very often within the logic (slowdowns, thread safety concerns).
- This is a helpful idea for future optimization that we have captured in an issue (https://github.com/JGCRI/stitches/issues/82), thank you!

- Installation instructions: Is there a clearly-stated list of dependencies? Ideally, these should be handled with an automated package management solution.
    - Dependencies for installation of the package are present in requirements.txt. Requirements pin pandas<1.5 (should be updated) and use pkg_resources (deprecated).
        - [CRV] we updated the pandas dependency
        - [CRV] changed pkg_resources to importlib
    - Documentation build requirements (recipe with sphinx + sphinx extensions) are not present in setup.py or repository.
        - [CRV] added in these in setup.py extras
    - Package metadata in setup() doesn't adhere to PEP standards, no use of package metadata classifiers (https://pypi.org/classifiers/).
        - [CRV] done
- Example usage:
    - Documentation is unclear on the data schema that is expected for the primary analysis operations (Necessary fields? Data formats? Expected outputs?)
        - This is a good point. Hopefully the revised quickstart and some of the example notebooks we developed after submitting this work address many of these concerns.
    - It isn't clear if the library can work with other existing intake-esm data stores or is hardcoded for the data facets seen in https://storage.googleapis.com/cmip6/pangeo-cmip6.json.
        - It does expect those facets, yes. We have made that explicit in the revised quickstart, thank you for highlighting it!

- Functionality documentation: Is the core functionality of the software documented to a satisfactory level (e.g., API method documentation)?
    - Docstrings do not really follow a consistent formatting, mostly adhere to conventions (https://peps.python.org/pep-0257/).
        - [CRV] done. Docstrings are now PEP 257 compliant and consistent.
    - Package API imports all functions to the top-level (would significantly benefit from modular divisions between testing/setup functions, analysis functions, and visualization functions).
        - [CRV] I agree. But we have users that are using this current format. We will schedule this in a major version release so we can notify the user community ahead of time. So no changes to this at this time. Thanks!
- Automated tests: Are there automated tests or manual steps described so that the functionality of the software can be verified?
    - There are GitHub Workflow CI tests configured for the repository. Tests are found within the package (stitches.tests) but should be moved to top-level / excluded from wheel.
        - [CRV] done. Tests are now outside of the package.
    - Testing configuration relies on outdated installation method ($ python setup.py install).

        - [CRV] testing config uses a standard pip install procedure in the action.
    - Testing setup relies on python calls to library (python -c 'import stitches; stitches.install_package_data()') but this should be made part of the testing setup stage or exposed via a CLI.
        - [CRV] fixed. Moved to a conftest.py file to conduct using a pytest wrapper.
    - Only Linux * Python3.9 is tested (would benefit from a testing matrix). GitHub Workflows are using deprecated Actions (actions/checkout@v1). Running tests locally shows that they pass (unittest or pytest) but shows many DeprecationWarnings.
        - [CRV] Action is generated using v3. We are also now running 3.9-11 on linux, mac, and windows in our matrix.
    - Code coverage reporting via Codecov seems to be misconfigured and not up-to-date (local testing shows 52%; Badge shows 8%).
        - [CRV] badge is linked to main. Higher coverage will show up once we merge dev to main.
- Community guidelines: Are there clear guidelines for third parties wishing to 1) Contribute to the software 2) Report issues or problems with the software 3) Seek support
    - Repository contains a CONTRIBUTING.md guide specific for the project.
        - [CRV] yes
    - Authors ask that contributors clarify whether contributions are copyright-restricted (mentions of a project called "Hector"?) - this could be made easier with Pull Request Templates.
        - [CRV] fixed mention of Hector typo. PR template will come in a later version as we are coordinating a standard one for use across our suite of packages.
    - No Issue Templates or Pull Request Templates.
        - [CRV] PR and Issue templates will come in a later version as we are coordinating a standard one for use across our suite of packages.
    - Testing/tooling setup and metrics are not mentioned in contributor guidelines.
        - [CRV] added a few comments about these in contributing.
    - No guidance is mentioned on how to report problems with software
        - [CRV] this is located in the README contributing section
    - Repository and ReadMe identify Code of Conduct (Contributor Covenant v2.0).
        - [CRV] yes
    - No contact method is listed for reporting abuse.
        - [CRV] this is covered in issue reporting which can act as a point of contact
    - Thank you, this has been addressed in https://github.com/JGCRI/stitches/pull/84
        - [CRV] AMAZING REVIEW! THANK YOU!