Response to the reviewers

Dear Reviewers,

Thank you for giving me the opportunity to submit a revised draft of my manuscript titled *Epwshiftr: incorporating open data of climate change prediction into building performance simulation for future adaptation and mitigation*. We appreciate the time and effort that you have dedicated to providing valuable feedback and insightful comments on our work. We have been able to incorporate changes to reflect most of the suggestions and have highlighted the changes within the uploaded manuscript.

Below is a point-by-point response to the reviewers' comments and concerns.

Reviewer 1

Reviewer Point P 1.1 — This paper developed a free open-source future weather generator epwshiftr, which can generate future weather data for 11 meteorological variables.

Reply: Thank you for the comments. We hope epwshiftr can ease the data-processing burden and reduce the costs of energy modelers when preparing future weather files for assessing climate change impacts on building energy performance.

Reviewer 2

Epwshiftr appears to be a useful tool for modelers investigating likely performance of buildings in the future.

Reviewer Point P 2.1 — The format and writing quality of the paper is acceptable once minor errors are corrected. I noticed as least one incorrect word usage ("further" instead of "future"). Please carefully proof-read the paper.

Reply: Thank you for pointing this out. The typo has been corrected. The manuscript has been proofread with spelling and grammatical errors corrected.

Reviewer Point P 2.2 — Figures 1 and 2 are both unreadable in a printout of the paper. They are not particularly clear even when viewing the PDF at high magnification. The listings are fuzzy; listings 4 and 5 are also too small. The manuscript is significantly shorter than the 8 page limit, so space is available to enlarge and clarify these items.

Reply: Thank you for the suggestions.

The manuscript has been rewritten using BS2023 LaTeXtemplate for better image and listing formatting. The image file of Figure 1 and 2 have been replaced with high-resolution ones. Also, both have been enlarged and placed in full page width.

In the original draft, all listings were screenshots of code snippets in small sizes. Now we replace all of them by rewriting using LaTeXlistings. Clarifications of code have been added as comments in each listing. The fonts have been enlarged to make sure they are readable in the printout version.

Reviewer Point P 2.3 — You refer to the morphing method as "reliable". I am not sure there is consensus on that. Morphing modifies individual weather items independently and may not preserve physically realistic relationships among them. Further, unconstrained morphing can produce physically impossible values, such as wet bulb temperature greater than dry bulb temperature. Finally, morphing does not capture time-dependent effects such as concentration of temperature increases in nighttime hours. You should amplify your discussion of how morphing is implemented.

Reply:

Thank you for the insightful comments and suggestions.

We have changed the description of the morphing method in both the Abstract and Introduction section. Both advantages and disadvantages of it have been added in the Introduction section:

> Morphing can capture the average future weather conditions from GCM while preserving historical weather sequences. It requires low computational power, making it possible to create many weather files from worldwide locations. However, morphing may under- or overestimate climate change impacts because of the lacking of ability to capture future extreme weather conditions and potential differences in the reference time frame of the TMY and GCM data (Moazami et al., 2019). Moreover, careful consideration should be given to morphing when modifying individual meteorological variables independently, breaking their physical relationships. Despite these shortcomings, this method is still widely used because of its simple and flexible characteristics.

Description about how the morphing method is implemented in epwshiftr has also been added:

> To avoid unrealistic results, the *Morphing Module* has taken extra data validation and calculation steps, including but not limited to:
>
> - Warnings are generated if there are any missing values in the input EPW and GCM data.
> - Unit conversions between data of EPW and GCM are automatically performed using the units (Pebesma et al., 2016) R package, e.g. all temperature data have been converted to Celsius before calculation.
> - Calculation of dew point temperature is performed based on dry-bulb temperature and relative humidity using the psychrolib (Meyer and Thevenard,
>
> 2019) R package.
>
> - Input values of relative humidity that exceed 100% will be reset to 100%.
> - A threshold value is set for the stretch factor (α), i.e. monthly-mean fractional change, when performing morphing operations. The default value is set to `3`. If the absolute α exceeds this threshold value, warnings are issued to suggest users further investigate the input data before continuing. Moreover, the morphing method will use the shift factor (Δx) to to avoid unrealistic morphed values.

Reviewer Point P 2.4 — Further, you should comment on the adaptability of your framework to support alternative extrapolation algorithms.

Reply:

Thank for the suggestion.

Currently, epwshiftr only supports the morphing method. We are happy to explore the feasibility of supporting alternative extrapolation algorithms in the future. Clarifications have also been added in the *Morphing Module* section:

> Currently, epwshiftr only supports the morphing method. But the `morphing_epw` interface provides parameters to modify which factors should be used for each meteorological variable, with meaningful defaults value given. For example, radiation-related variables are, by default, morphed using the stretch factor, avoiding unrealistic positive values at nighttime. The modular design pattern of epwshiftr makes it decouple the data structure and the actual extrapolation algorithm used. We are happy to explore the feasibility of supporting alternative extrapolation algorithms in the future.

Reviewer Point P 2.5 — In addition, it would be helpful to provide at least general information on testing and validation. Have simulation results been compared for studies done with epwshiftr weather data to those conducted with some of the other future weather methods you cite? Is the NetCDF data reliable and what happens when it is not? Can potential users be confident that epwshiftr will "do the job"?

Reply:

Thank you for the comments and suggestions.

We take seriously about code correctness. For details about the test coverage, please see epwshiftr code coverage report[1]. A description of the code quality assurance has been added in the 3rd section:

> The epwshiftr package follows the Test-Driven Development (TDD) process. Around 450 unit tests are carefully made, covering 94% of the codebase. They are automatically run on Windows, macOS, and Linux whenever changes are made in epwshiftr on CRAN and GitHub.

In terms of validation, besides data validation steps in epwshiftr itself described in Response , epwshiftr uses Defensive Programming pattern[2]. Careful efforts have been made to validate user inputs including data types, value length and range, ESGF data node availability, NetCDF file accessibility, etc., avoiding common mistakes before computation.

epwshiftr stores and returns data in a consistent tidy format, making it easy for users to examine the returned data of each module further. In terms of the calculated morphing factors, a description of the structure of the returned data from the *Morphing Module* is also added:

> Besides the efforts above, the *Morphing Module* always returns the calculated Δx and α values in dedicated columns, which provides opportunities for detailed examination and custom statistical analyses.

[1] `https://app.codecov.io/gh/ideas-lab-nus/epwshiftr?branch=master`

[2] `https://en.wikipedia.org/wiki/Defensive_programming`