We thank the reviewer for the constructive comments and suggestions. The comments are very helpful and will certainly strengthen the quality of the manuscript. Our response to the review can be found in the attached document.

R: Referee's comment
A: Author's response
C: Proposed changes in the manuscript
Blue letters: Suggested changes in the text

R: The manuscript by Sauter et al., describes the COSIPY v1.2 open-source coupled snowpack and ice surface energy and mass balance model. This model is designed to simulate the energy and mass balance of snow and ice covered surfaces, with applications for glacier mass balance simulations. The model builds on several decades of research in the field of snow cover and ice simulations. The main originality of this model is that it is implemented in Python. Given the scope and the content of the manuscript, it is fully appropriate for publication in Geoscientific Model Development. Overall, the manuscript reads well and I have not identified major flaws in the manuscript. Note, however, that I haven't checked one by one all the equations in Sections 2 and 3, which are based on classical concepts and frameworks for snow and ice energy and mass balance. I have several comments, which can rather be seen as suggestions, to the authors, and a series of minor comments.

Section 4 : While Sections 2 and 3 are in fact of limited added-value given that the equations are concepts are already outlined in a number of previous publications (it is fine to leave them in the manuscript, this is a useful reference for users of the model or its output, perhaps complemented by recent publications such as Essery et al., 2013, http://dx.doi.org/10.1016/j.advwatres.2012.07.013 and Lafaysse et al., 2017, http://dx.doi.org/10.5194/tc-11-1173-2017), I find section 4, addressing "Model architecture", quite short and it could be expanded to better address the novelty and added value of the model compared to previously existing models. For example, I think that it could be useful to provide more details regarding the Python libraries used for this model, their common dependencies, added-value, etc., and how the "modularity" of the model structure is implemented. This could be addressed not only by adding text, but also figures, providing an overview description of the model structure and the interlinkages between them.

A: We will complement the Sections with the work done by Essery et al. (2013) and Lafaysse et al. (2017) at the corresponding locations. Both are excellent references.

Originally we had planned another paragraph about the modular structure of the model. However, we decided to not integrate it, as we felt that this technical information would be better presented in the readthedocs online documentation (https://cosipy.readthedocs.io/en/latest). We will, however, add a paragraph about the code implementation, dependencies and structure of the model. Some information is already given in Section 7 'Code availability, documentation, and software requirements', which is mandatory in GMD. Additionally, we will also discuss the added-value of COSIPY compared to previous glacier mass balance models. This has already been criticized by the first reviewer.

C: In Section 4, we will take up this issue again and highlight the special features. These include

- is completely written in Python, modular and object-oriented
- completely based on open-source libraries

- has a readthedocs documentation (which is still in the development phase)
- parameterizations can be easily extended or modified by the user
- NetCDF IO routines
- easy integration of Weather Research and Forecast (WRF) forcing
- adapted for distributed glacier mass balances simulations; it needs to be pointed out in this regard that COSIPY is <u>not</u> a snow cover model
- has a community platform (Slack) and code is actively maintained
- new git commits are automatically tested via travis and codecov
- each model version gets a DOI
- has a restart option for operational applications

R: Section 5 : The section 5 provides an example of the model use for the Zhadang glacier, High Mountain Asia, with illustrations of model output (Figures 1 and 2) and model performance (Figure 3) for this case study. The results appear to be reasonable for a typical energy and mass balance model applied to a glacier setting. However, this does not correspond to a full model evaluation exercise, and I think this model description article would greatly benefit from a more robust evaluation. In this respect, I think the dataset used for the ESM-SnowMIP intercomparison could be particularly useful. All the relevant data have been made available in Ménard et al., 2019 (https://doi.org/10.5194/essd-11-865-2019), and paper such as Krinner et al., 2018 (https://doi.org/10.5194/gmd-11-5027-2018) can be used as inspiration for providing the evaluation indicators of snow models. Regardless of how this is handled, I consider useful for this model description article to provide some evaluation metrics relevant to the performance of the model described in this article.

A: Thank you for the suggestion. We agree it would be of great benefit to include a comparison with other models, however a full model intercomparison is beyond the scope of this manuscript.. We will use the data from Ménard et al., 2019 as forcing data to reproduce the metrics and compare COSIPY to other models similar to Krinner et al., 2018. It might be that we can not reproduce all metrics since COSIPY is a glacier energy and mass balance model and not a snowmodel, i.e., the soil heat flux is not parameterized in the same way, for example. Furthermore, we will extend the model evaluation for the Zhadang glacier with more ablation-stake data and present profile plots of the layer properties for the Hintereisferner in the European Alps. This exercise is also in response to comments of the other reviewer.

C: We will make the appropriate changes to the existing text in Chapter 5 and will introduce new paragraphs/sub-chapters on the new datasets and evaluations.

R: Page 4, line 15 : how do the re-meshing algorithms compare to existing re-meshing algorithms used in other snow cover models ? I think in particular of Crocus (Vionnet et al., 2012, https://doi.org/10.5194/gmd-5-773-2012), there are other models with remeshing approaches. I think it would be good to position the approach taken here within other existing models.

A: Similar to CROCUS, COSIPY uses a set of criteria that determine when two layers are merged or splitted. As already indicated in the text the user can choose between two options:

(i) Logarithmic profile: This method is only suitable for simulations where the layering of the snowpack is not relevant (bulk). Layer thicknesses are calculated starting from the top layer, which always remains at a constant thickness, and gradually increases with depth by a constant stretching factor. Thus the layers close to the surface have a higher spatial resolution, which is advantageous for the computation of the energy and mass fluxes at the surface.

(ii) Adaptive profile: The adaptive algorithm runs in three consecutive steps: (1) adding/removing snow/ice at the surface, (2) adjusting the first layer, (3) updating internal layers.

> In the first step it is checked whether snow falls or melts away (note: internal layers can also melt). If snow falls on the glacier surface, it will only remain on the surface if it reaches a user-defined minimum snow thickness. If it falls on an existing snowpack, any snowfall that exceeds a user-defined minimum threshold is added to the snowpack. Melt is removed from the first layers and internal layers. After this step, layers can become very small and the thickness of the first layer no longer corresponds to the user-specified constant thickness. Therefore, it is necessary to remesh the layers.
>
> In the second step, the top layer is adjusted first. The top layer is remeshed so that this layer always has the user-defined layer thickness (default value is 0.01 m). The adaptation of the top layers together with internal melting processes can reduce the internal layers to a very low thickness. To avoid thin layers, the layers are merged or split in the last step (see next paragraph).
>
> In the last step, internal layers are splitted or merged. For each layer, a check is made to identify layers with a thickness of less than a defined minimum layer thickness. Such thin layers are merged with the layer below. Also if the differences in temperature and density of two subsequent layers are less than a user defined threshold (similarity criteria), they will be merged. How often a merging/splitting can take place per time step is also defined by the user (correction steps). Unlike CROCUS, internal remeshing always starts from the surface, i.e. the uppermost layers are adapted first. Depending on how many correction steps are set by the user, it can happen that only the uppermost layers are remeshed.

C: We will extend the description of the meshing algorithms (beginning from page 4 line 15) and describe them in more detail as outlined above.

R: Page 4, line 24 : “useful feature” : would it be possible to elaborate on what is meant by “useful feature” ? What metric was used to address the “usefulness” ?

A: The term 'useful feature' should indicate that the adaptive algorithm is reasonable when one is interested in the stratification of the snowpack. In contrast to the logarithmic profile, one obtains well resolved layers. This can be important for some glaciological issues, but it is computationally more expensive than the logarithmic algorithm.

C: We will rewrite the sentence “*The adaptive re-meshing proves to be a useful feature, but slightly increases both the computing time and the data volume*” to “***Unlike the logarithmic approach, adaptive re-meshing resolves individual layers but slightly increases both computing time and data volume.***”

R: Page 7, line 3 : I suggest to use the LaTeX symbol \varepsilon instead of \epsi, this seems to better match the graphical design of the "epsilon" symbol, when it refers to the emissivity.

A/C: Will be done.

R: Page 9, line 3, I suggest replacing "Von" by "von" for the name of "von Karman" (ideally with "accents" on the "a"s).

A/C: Will be done.

R: Page 10, line 7 : I don't think it is adequate to refer to "snow grain settling", but "Snow settling" would be less ambiguous and more accurate.

A/C: We agree and will change it to "snow settling".

R: Page 11, line 20 : I think more details should be given on what is referred to here as "dynamic mesh" ?

A: The wording is probably a bit confusing. The term should refer to the computational mesh which can be (dynamically) adapted by the re-meshing algorithms.

C: To clarify this misunderstanding we will remove the term 'dynamic'.

R2: P12 L9: More explanations could be given to better explain the content of the parenthesis "(not recommended for distributed simulations)"

A: The user can specify in a file which data should be stored. In addition to the atmospheric variables, COSIPY calculates the state of the snow/ice layers. Since the number of vertical layers and grid cells can be very high, it is recommended to store only those variables that are necessary for later evaluation. We recommend dropping the states of the layers in distributed simulations to save memory space.

C: We will change the sentence to: "*Besides the standard output variables there is also the possibility to store vertical snow profile information****, although to save memory we can only recommend this for single point simulations.***"

R2: P12 L31 : I think more explanations are needed for "driven by ERA-5", in particular whether downscaling was applied, and if yes, how.

A: We extracted the needed input data from the nearest ERA5 grid point and applied downscaling methods to the variables. We will further clarify this in the revised version of the manuscript.

C: We will add a table to the manuscript with the applied downscaling approaches and change the sentence to: *"The model was driven by ERA5 data instead of in-situ observations.* ***The ERA5 data were downscaled to the site using straightforward approaches. Temperature, T_{zr}, and humidity, RH_{zt}, were corrected to the altitude of the grid cell using empirical lapse rates. For pressure, p_{zt}, the barometric formula was used. The radiation model of Wohlfahrt et al (2016) was used for the incoming shortwave radiation to account for effects of shadowing, slope and aspect. Total precipitation RRR, N and U_{zv} were used directly from the closest ERA5 grid point."***

R2: P13 Figure 1: I suggest replacing "modelled" by "simulated" in the legend and captions.

A/C: We agree and will change it accordingly in all legends and captions.

R2: Page 14, Figure 3: Would it be possible to provide a definition for the term "Speedup"? I think this would be a useful clarification. If possible, it would be useful to provide a comparison of this metric with other existing models, in order to address to what extent the scalability of this Python-based model is comparable to implementations using other programming language.

A: Thank you for the important comment. The speedup is the ratio between the single-core execution time and the execution time of the corresponding multiple-core simulation. We wanted to point this out with the sentence: "..., i.e. the ratio of the original execution time with the execution time of the corresponding node test." In the present case, it is difficult to compare this value with other models because we would have to run the other models to the same test case. The speedup in the present case should rather show the performance gain when using multiple cores in contrast to a single core computer setup.

C: We will change the caption of Figure 3 to: *"Speedup* ***(execution time of single-core simulation divided by execution time of the corresponding multiple-core simulation)*** *for computing a 10-year distributed COSIPY run on Zhadang glacier with 206 grid points."* and the respective sentence to: *"..., i.e. the ratio of the original execution time* ***(single core)*** *with the execution time of the corresponding test* ***(multiple cores)****."*

R: Page 15, line 15 : I think it would be appropriate to also refer to multiphysics modelling, and it would be good to know to what extend COSIPY can be used for such applications (see e.g. Pritchard et al., 2020, https://doi.org/10.5194/tc-14-1225-2020).]

A: Multiphysical modeling is a very exciting topic and we are convinced that it will become even more important in the future. COSIPY is a modeling platform designed to test and apply different parameterizations. In principle it is already possible to generate ensemble simulations with different physical parameterizations and solvers, but COSIPY is not yet an ensemble multiphysics modeling environment. As a vision for the future it is conceivable to extend COSIPY for automatic ensemble simulations. Various uncertainties could be included - multiphysics modeling, perturbed input data or parameter uncertainty. So far, we have not yet thought about how an ensemble can be realized in a single simulation. Until we have a feasible idea, we have no choice but to run COSIPY with different combinations of physical parameterizations or input uncertainty and evaluate the statistics afterwards.

C: At this point we will, as an addition to the existing manuscript, only provide an outlook on what might be possible with COSIPY in the future, e.g. ensemble simulations.