Thank you so much for your valuable comments and suggestions that led us making improvements. Below, we present each comment for the Reviewer #1 in blue and our respective answer in green.

Review #2:

Points for Improvement

2.1 Data cube definition

There is no definition of "data cube" in the paper. The example provided by the paper uses a small area with specific characteristics. Different definitions of a data cube exist. Appel and Pebesma (2019) see it as a regular space-time partition with all images sharing the same coordinate projection. Simoes et al. (2019) extend this to include tiles with different coordinate projections but same spatial and temporal resolutions. The stricter definition requires all tiles to be converted to a single projection, which is hard for large areas. The extended definition allows a data cube to cover large areas. The authors need to clarify what a "data cube" means for them. The examples suggest they adopt the definition from the "cubo" package, which is a limited version of Appel and Pebesma (2019), focusing on small patches in the UTM coordinate system.

We thank the reviewer for noting the absence of an explicit definition, and for laying out the relevant distinctions so clearly. We have added a definition and state precisely which one we adopt. In the revised manuscript we explicitly cite Appel and Pebesma (2019) for the definition of a data cube.

stac2cube constructs a single-projection space-time cube in the sense of Appel and Pebesma (2019): all Sentinel-2 items contributing to a cube are loaded onto one common grid with a single coordinate reference system and a single spatial resolution, yielding dimensions (time, band, y, x) on projected, metric (UTM easting/northing) coordinates. We do not adopt the extended Simoes et al. (2019) definition: the package does not retain tiles in their native UTM zones. When an area of interest spans several MGRS/UTM zones, all items are warped onto one common CRS.

We have also clarified how our cube relates to cubo, since this bears on the reviewer's observation. While stac2cube builds on the cube foundation established by cubo, it is not restricted to cubo's fixed-size patches. The area of interest is an arbitrary user-supplied polygon or bounding box; items from multiple adjacent MGRS tiles that overlap the AOI are mosaicked into a single continuous timestep per solar day (with an optional mode that instead keeps each tile as its own timestep); and multi-zone AOIs are accepted rather than refused. In this sense our cube sits between the cubo restriction and the extended definition: strict single-projection in the Appel & Pebesma (2019) sense, but supporting arbitrary, multi-tile areas rather than isolated patches. We additionally attach a cube-level attribute system recording provenance and build configuration (contributing MGRS tiles, sensor, resampling, cloud-processing status, source

STAC API, and AOI), and per-scene coordinates for scene coverage and, when cloud detection is enabled, cloud percentage.

We have also refined how the common projection is chosen. When the items returned for an area of interest span several MGRS/UTM zones, stac2cube computes each candidate projection's share of the area of interest and selects the projection that covers the largest share, so that the least data has to be reprojected. If the user specifies a target CRS, that choice is used, and a warning is issued when a chosen or selected projection is not native to all scenes. We note as a limitation that, for very large multi-zone areas, a single common projection still involves some distortion, which we discuss together with possible future refinements.

2.2 Author's contribution

The authors developed an extension for the "cubo" package, including functions to enhance cloud-cover removal, image registration, and produce super-resolution data. However, this does not constitute an "open-source package for customized processing of Sentinel-2," as claimed. Their work lacks functions typical of EODC tools, like raster analysis and machine learning classification. A more accurate description is that they created pre-processing tools for Sentinel-2 data in cube environments. For this reason, the manuscript title does not accurately reflect the authors' contributions. It is suggested that the title be revised to better reflect what the authors have done. One option is to use a title such as "An Open-Source Package for Customised Pre-processing of Sentinel-2 Data for Data Cube Analytics", or something along these lines.

We thank the reviewer, and we largely agree. We accept that "processing" overstates the contribution: stac2cube does not provide the downstream raster-analysis and machine-learning-classification functions typical of full EODC tools, and its role is the preparation of analysis-ready Sentinel-2 cubes. We have revised the title to reflect this. We have adopted a formulation close to the reviewer's suggestion, along the lines of "An Open-Source Package for Customised Pre-processing of Sentinel-2 Data for Data Cube Analytics," and harmonised it with the related change requested by RC1 (removal of the "scalable/HPC" wording).

We would, however, like to clarify the scope of "pre-processing," as our contribution is more than a lightweight extension of cubo. Implementing probabilistic cloud masking (s2cloudless), geometric co-registration (AROSICS) and super-resolution (SEN2SR) as a coherent, cube-based workflow required a dedicated pipeline rather than a plug-in to an existing package. For example, the s2cloudless workflow requires Sentinel-2 L1C input alongside L2A, which entails precise handling of STAC metadata (e.g. the processing baseline) and specific resampling choices that existing packages do not expose. As described in our response to §2.1, the package also constructs single-projection cubes over arbitrary, multi-tile areas of interest (mosaicking adjacent MGRS tiles), which goes beyond cubo's fixed-patch scope.

In addition, stac2cube exposes a range of transparent, user-controllable parameters that the previously mentioned packages do not surface, giving users explicit control over the pre-processing (for example the cloud-probability threshold and masking strategy, co-registration

configuration, resampling method, tile-handling and partial-scene-coverage behaviour). As of v1.5.0 we also provide a Jupyter-based graphical user interface that lowers the entry barrier and guides users through an easy-to-follow workflow, further improving usability. We have described these aspects in the revised manuscript so that the nature and extent of the pre-processing contribution are accurately conveyed.

2.3 Relation to previous work

When stating their contributions relative to previous work, researchers need to be careful on how they state previous work. In the paper, the authors state: "We identify three recurring deficiencies in both open-source and commercial EODC tool chains that limit their utility for challenging, dynamic land surfaces (e.g., braided rivers) and for long-term time-series analysis." Strong arguments need solid evidence. The terms "deficient" and "limited" suggest these tools can't perform data analysis, but this isn't self-evident. To support this, authors should specify the reviewed "open-source and commercial EODC tool chains" and detail their deficiencies. Fairness in evaluating third-party tools is lacking; examples like Pebesma et al. (2026) and Gomes et al. (2020) illustrate proper evaluation. Despite not being a review, authors should have briefly described the EODC tool chains they consider and pointed out their capacities and limitations. Considering the current EODC tool chains like Google Earth Engine, openEO, sits, and Open Data Cube, calling these tools "deficient" and "limited" is unwarranted. The authors should highlight their work positively, emphasizing the benefits their work offers to data cube users.

We thank the reviewer, and we agree completely. Looking backward, we recognise that describing established EODC tool chains as having "recurring deficiencies" that are "limited" was both unsupported and not appropriately framed. Tools such as Google Earth Engine, openEO, sits, and Open Data Cube are mature, capable, and widely used, and we do not intend to suggest they cannot perform advanced data analysis. We have adjusted the framing in the revised version of the manuscript.

We have replaced the "deficiencies" statement with a positive framing of our own contribution: rather than claiming these tools fail, we state what stac2cube adds that is not currently readily available in STAC-based Python cube workflows, and for which some demanding Sentinel-2 applications (such as dynamic braided-river systems) create a need. A concrete example is probabilistic cloud masking (s2cloudless): while available within Google Earth Engine as a precomputed collection, to our knowledge it is not readily accessible from a STAC-based Python package, which is a gap our integration fills.

Following the reviewer's suggestion and drawing on the evaluation style of Pebesma et al. (2026) and Gomes et al. (2020), we have also added a brief, fair description of the EODC tool chains we refer to, noting their capabilities as well as the specific points at which our integrated pre-processing adds value, rather than characterising them negatively. We are grateful to the reviewer for prompting this correction, which we agree strengthens the contribution.

2.4 Lack of API package description

When examining similar works on EO cloud computing like Gorelick et al. (2017), Simoes et al. (2019), and Schramm et al. (2021), all describe their packages' APIs, including their functionality, usage, benefits, and limitations. It's important for authors to provide a similar API discussion.

We thank the reviewer for this suggestion. We agree that, following the example of Gorelick et al. (2017), Simoes et al. (2019), and Schramm et al. (2021), the manuscript should describe the package itself more concretely. In the revised version we have added a dedicated subsection describing stac2cube's functionality, usage, benefits, and limitations.

This description covers: the main entry point for cube construction and the parameters through which the user controls it (data source and Sentinel-2 level, area of interest, bands, spatial resolution and resampling method, tile-handling and partial-scene-coverage behaviour); the optional processing steps and how they are invoked (probabilistic cloud masking via s2cloudless with a user-set cloud-probability threshold, cloud- and shadow-masking strategy, geometric co-registration via AROSICS with a user-selectable master scene, and super-resolution via SEN2SR); and the outputs, an Xarray cube with dimensions (time, band, y, x) on a single-projection metric grid, per-scene coordinates (scene coverage and, when cloud detection is enabled, cloud percentage), and a cube-level attribute block recording the build configuration and provenance (contributing MGRS tiles, sensor, resampling, cloud-processing status, source STAC API, and AOI) that also enables a cube to be reproducibly extended.

We have also stated the benefits (an integrated, transparent, parameter-controlled pre-processing workflow producing analysis-ready cubes, with output to NetCDF, GeoTIFF, and Zarr) and the limitations (for example the single-CRS behaviour for large multi-zone AOIs noted in our response to §2.1, and the per-scene application of super-resolution discussed in §2.5). As of v1.5.0 a Jupyter-based graphical user interface additionally guides users through this workflow. We have kept this description focused on functionality and usage appropriate to the journal's scope, rather than a comprehensive API reference.

2.5 Limitations of the work

Careful authors inform the reader of their contributions. They frame the work as necessary to avoid "deficiencies" of existing EODC tools, which is not true. Existing tools are widely used without the authors' pre-processing tools. The authors should present their results positively and openly discuss the benefits and limitations of their tool, including scenarios where it is particularly valuable. Consider an EO data cube or analysis-ready data provider like Microsoft Planetary Computer, Digital Earth Africa, the Copernicus Data Space Environment, or the Brazil Data Cube. These providers typically get ARD tiles from ESA with cloud info in the SCL band, already split into MGRS tiles. If they switched to "s2cloudless" and AROSICS tools, they'd need to reprocess all Sentinel-2 images, a significant task. Do the authors see reprocessing as necessary for all EO cloud providers? Are there specific cases where their tools are essential? The authors should be more rigorous about the super-resolution analysis. They assume "more is better" and

that super-resolved S2 images are superior in all cases, which entails a 25-fold increase in database size and processing. Is this justified, especially since 10-meter Sentinel-2 images often suffice? Also, they must consider how super-resolution affects long-term time series consistency to fully justify their tool's value.

We thank the reviewer, and we agree the work should be framed around benefits and clearly stated limitations rather than the "deficiencies" of other tools; we address the tone in our response to §2.3 and have applied the same positive framing here.

About the providers: We do not argue that ARD providers such as Microsoft Planetary Computer, Digital Earth Africa, the Copernicus Data Space Environment, or the Brazil Data Cube should reprocess their Sentinel-2 archives. These providers deliver SCL-based, MGRS-tiled ARD that is well suited to the large majority of applications. Our pre-processing is not intended as a replacement for that model but as an on-demand, user-side option for specific cases where the standard products are less adequate, for example dynamic, heterogeneous surfaces such as braided rivers, where probabilistic cloud masking over bright gravel bars, intra-cube co-registration, and optional super-resolution provide tangible benefit. We have stated this scope explicitly, including the cases where our tools are and are not essential.

About super-resolution: We agree that super-resolution is not universally beneficial, and we would like to note that the manuscript already reflects this: in the Discussion we recommend the 10 m cube as the default and present the 2.5 m super-resolved product as suited to specific tasks with high spatial heterogeneity, and in the Figure 10 (UAV) comparison we document a fundamental limitation, where the model does not introduce genuinely new surface information (units already spectrally mixed at 10 m, such as small vegetation patches, remain aggregated after super-resolution). We have strengthened and signposted this discussion, so the trade-off is stated more explicitly and accurately. In particular, super-resolving from 10 m to 2.5 m increases the pixel count 16-fold (areal 4x4) and the computational cost scales accordingly, but the on-disk storage increase is smaller, approximately 8-fold at default settings, because the super-resolved output is now packed to int16 at half the bytes per sample (as of v1.5.0) (and can be reduced further, to roughly 4 to 5-fold, with optional lossless compression). We have updated the paper accordingly, replacing the earlier figure with these measured values. As the benefit remains task-dependent, for many applications the 10 m cube will remain the recommended default.

We have additionally extended the discussion to address temporal consistency directly. Because SEN2SR is applied per scene, stacking super-resolved scenes could in principle introduce learned, scene-to-scene variation distinct from real surface change. In examining this, we observed that SEN2SR inference is not bit-reproducible when run on GPU: two runs of the identical model on the identical input scene, at full single precision with identical settings, differed in a large fraction of pixels, though the magnitude was very small (a maximum deviation of the order of 1e-3 in reflectance units, and per-band mean absolute deviations of the order of 1e-5, roughly 0.01% of each band's value range). Repeated CPU execution, by contrast, reproduced outputs bit-identically, which localises this variability to the CUDA execution path

rather than to the model, the input data, or the storage format; the implementation enables cuDNN kernel autotuning and sets no deterministic-algorithm constraint, which is a plausible but not yet isolated mechanism. These observations come from a single test site and a single GPU, and we did not compare GPU- against CPU-produced outputs, so we make no claim about any cross-hardware discrepancy. Given this, we currently recommend the super-resolved product primarily for enhanced visual and geomorphic interpretation, and we treat its suitability as a validated input for quantitative multi-temporal analysis, including classification and segmentation, as an open question for dedicated follow-up work rather than an established benefit.

3 Final remarks

The manuscript introduces tools for Earth observation data cubes, but flaws in presentation undermine its message. Instead of highlighting benefits, the authors adopt an adversarial tone, criticising earlier work in offensive terms. These negative claims lack rigorous justification, weakening the overall contribution. The manuscript needs major revisions. The authors should reconsider how they describe their work relative to the literature. Their work is valid. With careful, positive rewriting, it merits publication.

We thank the reviewer for the constructive spirit of these closing remarks, and in particular for recognising that the underlying work is valid. We take the criticism of tone seriously. As detailed in our responses to §2.2, §2.3, and §2.5, we have revised the manuscript to remove the adversarial framing of existing tools, replace unsupported negative characterisations with a fair account of those tools' capabilities, and present our contribution in terms of the specific benefits it offers rather than the shortcomings of others. We are grateful for the reviewer's assessment that, with this positive rewriting, the work merits publication, and we have aimed to reflect that guidance throughout our revisions.

Code Repository for stac2cube v1.5.0:

Code availability: The stac2cube source code is openly available on GitHub (https://github.com/BaturalpArisoy/stac2cube) and archived on Zenodo. The version described in this revision is v1.5.0 (DOI: 10.5281/zenodo.20666787), released under the Apache License 2.0.