The manuscript presents UpsFrac, a MATLAB-based workflow that links (i) generation of deterministic and stochastic discrete-fracture models (DFM) and (ii) flow-based upscaling of their equivalent permeability using the Multiple-Boundary Method (MBM) implemented with TPFA/MPFA discretization. The authors provide validation against analytical solutions, a field-scale demonstration, and release the code under GPL-3.0 on GitHub and Zenodo. The topic is timely; few open-source tools bridge fracture modelling and rigorous flow upscaling in a single framework. The paper would be publishable after a major revision that addresses methodological clarifications, strengthens validation, and improves presentation.

General Comments

1. While UpsFrac provides a useful integrated workflow, the individual components (ADFNE, MRST, MBM) are not novel. The paper would benefit from a clearer positioning of what is truly new: Is it the integration? Is it the application to fractal systems? Is it the ability to handle both deterministic and stochastic fractures? In the introduction section, clearly delineate the innovation over existing tools like PorePy, DFNWorks, or OpenGeoSys-DFM modules.

2. The accuracy of upscaling strongly depends on mesh size, particularly for fractures aligned with or against the grid. Include a sensitivity analysis showing how grid refinement or fracture-matrix contrast influences permeability estimates. Report convergence trends and potential sources of numerical error.

3. The code assumes constant aperture per fracture and a simplified grid-based representation. Discuss the implications of this assumption for faults with variable aperture or roughness. It would be better to quantify the error induced by this simplification. Consider adding a test where a fracture with log-normal aperture variability is upscaled with (i) constant-aperture assumption and (ii) a higher-fidelity solver (e.g., locally refined mesh or transfer-function approach)

4. The current implementation is 2D, which is acceptable for early-stage validation and pedagogy but limits industrial applicability. Provide a roadmap or discussion of how 3D capabilities could be implemented.

5. Computational performance and scalability are not analyzed. Provide runtime and memory statistics for the field-scale case (number of grids and fractures, CPU/GPU specs). Compare

with at least one alternative open-source package (e.g., OGS) to highlight UpsFrac's efficiency or identify bottlenecks.

6. Two validation tests (power-law DFN and single fracture with variable aperture) are useful but narrow. Add a heterogeneous matrix-fracture system where analytical REV permeability is known. Then compare it with embedded-discrete-fracture simulation on the full domain to confirm that the upscaled EFM reproduces domain-scale responses

7. The abstract claims that UpsFrac can easily run DFM ensembles for uncertainty analysis, yet no ensemble results are shown. Include an example with 50–100 stochastic realisations, report statistics (mean, variance) of the upscaled tensor components, and discuss convergence

8. Algorithm description lacks reproducibility details. Scripts such as "SubDivideToGrid.m" or "CalcuKeq.m" are mentioned but not summarized. Add a concise pseudo-code or flowchart (in the paper) that maps each processing step to its script name; list key input parameters

9. Frequent grammatical errors and misspellings distract from the content. Thorough language editing is required; see minor comments below for examples.

Technical corrections

1. In the abstract, clarify that the current implementation is 2-D and MATLAB-based.
2. Fig. 2 workflow: fonts are blurry; consider vector graphics.
3. Fig. 7 permeability map: add color-bar units
4. Fig. 9 ellipses: scale key missing (are axes normalized?).
5. Consistency in references: some entries list full journal names, others abbreviations; unify style.