

## ***Interactive comment on “Slate: extending Firedrake’s domain-specific abstraction to hybridized solvers for geoscience and beyond” by Thomas H. Gibson et al.***

### **Anonymous Referee #1**

Received and published: 29 May 2019

This is a strong contribution that combines state-of-the-art (albeit existing) numerical approaches of great interest to the geosciences community, with a well-engineered and extendable software implementation (Slate).

The authors state that the contribution is in the automated translation of mathematics to compiled code, and on this the paper delivers. The topic is clearly suitable within the context of the Firedrake special issue. Having implemented cell local operations manually using Eigen and C++, I can attest to the usefulness of having something like Slate available to the community.

It would be useful to have the following questions addressed in the manuscript:

C1

1) How is the `.inv()` command translated to generated code? Are you using LU/Cholesky or directly inverting the matrix? It is clear the former for the `A.solve()` command as the decomposition is passed, but not for `.inv()`. 2) How would you deal in practice with non-linear problems solved using, e.g. a Newton-Krylov method. Is it possible to compose SLEPc and the Preconditioned Krylov Solvers? How would one setup SLEPc to call the code to recover the internal/local variables between Newton iterations? 3) What does the generated Eigen code look like? 4) It is not totally clear how the output from TSFC is fed into the linear algebra compiler. Do you call the TSFC kernel, get the complete cell tensor, split it and then perform the dense linear algebra operations? Or is everything 'interleaved' into one single cell tensor kernel by the linear algebra compiler? Or do you call multiple TSFC kernels, one for each sub-block and then perform the dense linear algebra operations? 5) The high-level problem setup in sections 1 and 2 is pretty terse. I don't think someone who has some knowledge of FEM but is not a real subject expert could get through this section. Some of the wording is quite heavy on jargon too, e.g. global data structure  $\sim$  sparse matrix? I appreciate you don't have time to do a deep-dive into FE, but a bit more text and some pointers to more detailed explanations (other Firedrake papers?) would be useful.

Small notational comments:

\* You do not mention the bold symbol = vector function convention. \* Your convention of non-bold capitals being (local? - not sure) discrete linear operators (matrices and vectors) is also not mentioned, although  $K$  (the finite element cell) would break this 'convention'. I'd also note that  $K$  is used twice for different objects (the finite element cell and a matrix in eq.76) It would probably aid readability if it was possible to distinguish between matrices and vectors, global and local, especially when the operations become more complex in the later sections. There also seem to be (global? - not sure) discrete linear operators in bold.

2019.

C3