High-level python abstractions for data management in inversion problems

Checkpointing, done beautifully.

Navjot Kukreja
Imperial College London
London, UK

Jan Hückelheim
Imperial College London
London, UK

Michael Lange
Imperial College London
London, UK

Who Else
Imperial College London
London, UK

ABSTRACT

The computation of adjoints in optimisation and inverse design problems requires storing intermediate data. The available memory size sets a limit to this, and necessitates recomputation in some instances. Revolve checkpointing offers an optimal schedule that trades computational cost for smaller memory footprints. Integrating Revolve into a modern python HPC code and combining it with code generation is not straightforward. We present an API that makes checkpointing accessible from a DSL-based code generation environment and present a benchmark study. TODO.

CCS CONCEPTS

• **Software and its engineering → Software design engineering**;

KEYWORDS

HPC, Code generation, API, Checkpointing, Adjoint, Inverse Problems

ACM Reference format:
Navjot Kukreja, Jan Hückelheim, Michael Lange, and Who Else. 1997. High-level python abstractions for data management in inversion problems. In *Proceedings of SuperComputing, Denver, Colorado, USA, November 2017 (SC17),* **??** pages.
https://doi.org/10.475/123_4

1 INTRODUCTION

Motivation: Why do we need to connect revolve and a python seismic inversion code? What is there to learn for others? How does our work help readers and the world?

2 SEISMIC INVERSION

Some context on seismic imaging, the PDEs, the method. Why use adjoints. Why is this problem time-dependent. Why do we need a lot of data.

SC17, November 2017, Denver, Colorado, USA

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

3 DEVITO

Code generation, references to previous papers about Devito. Why not be brave and write this in Fortran.

4 REVOLVE

How does revolve work and save memory. How do others use it, why is this painful.

5 API: CONNECTING REVOLVE WITH DEVITO

This is the core of the paper.

6 TEST CASE, RESULTS

A nice test case that can be scaled up in size easily. Timings for different mesh sizes, with different amounts of memory set as the upper limit.

7 CONCLUSIONS

This work was highly successful, but more work could be done.

ACKNOWLEDGMENTS

The authors are very grateful.