Efficient I/O and Data Management for Exascale Earthquake Simulation and Analysis

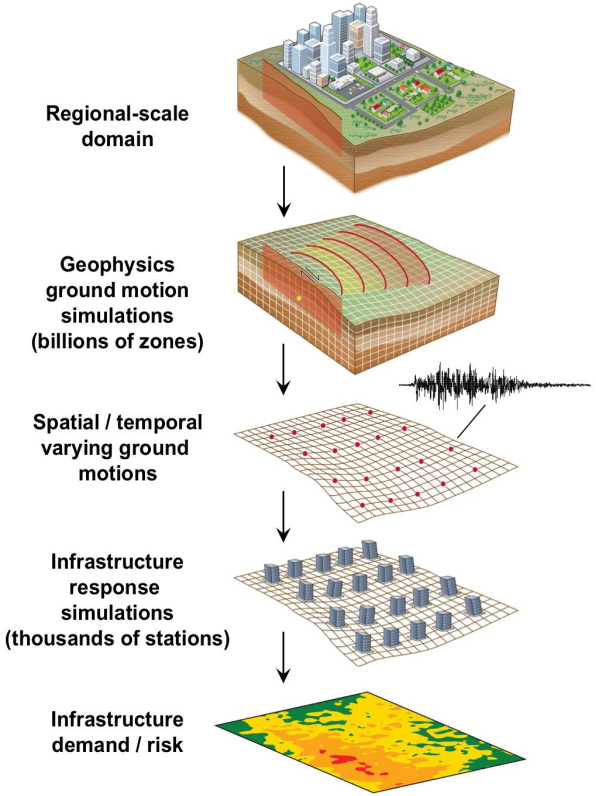
Houjun Tang

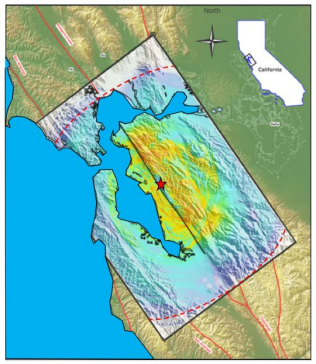
Berkeley Lab

Collaborators: David Mccallen, Anders Petersson, Arthur Rodgers, Arben Pitarka, Mamun Miah, Floriana Petrone, Bjorn Sjogreen, Ramesh Pankajakshan

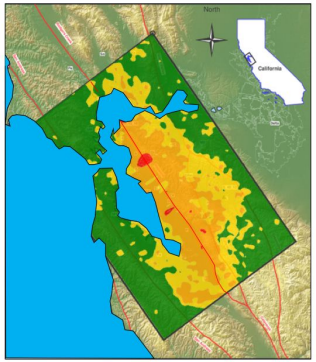
BERKELEY LAB
Lawrence Livermore National Laboratory
University of Nevada, Reno
ECP EXASCALE COMPUTING PROJECT

EQSIM: A Framework for Regional-scale Fault-to-structure Simulations





What is the distribution of ground motions?



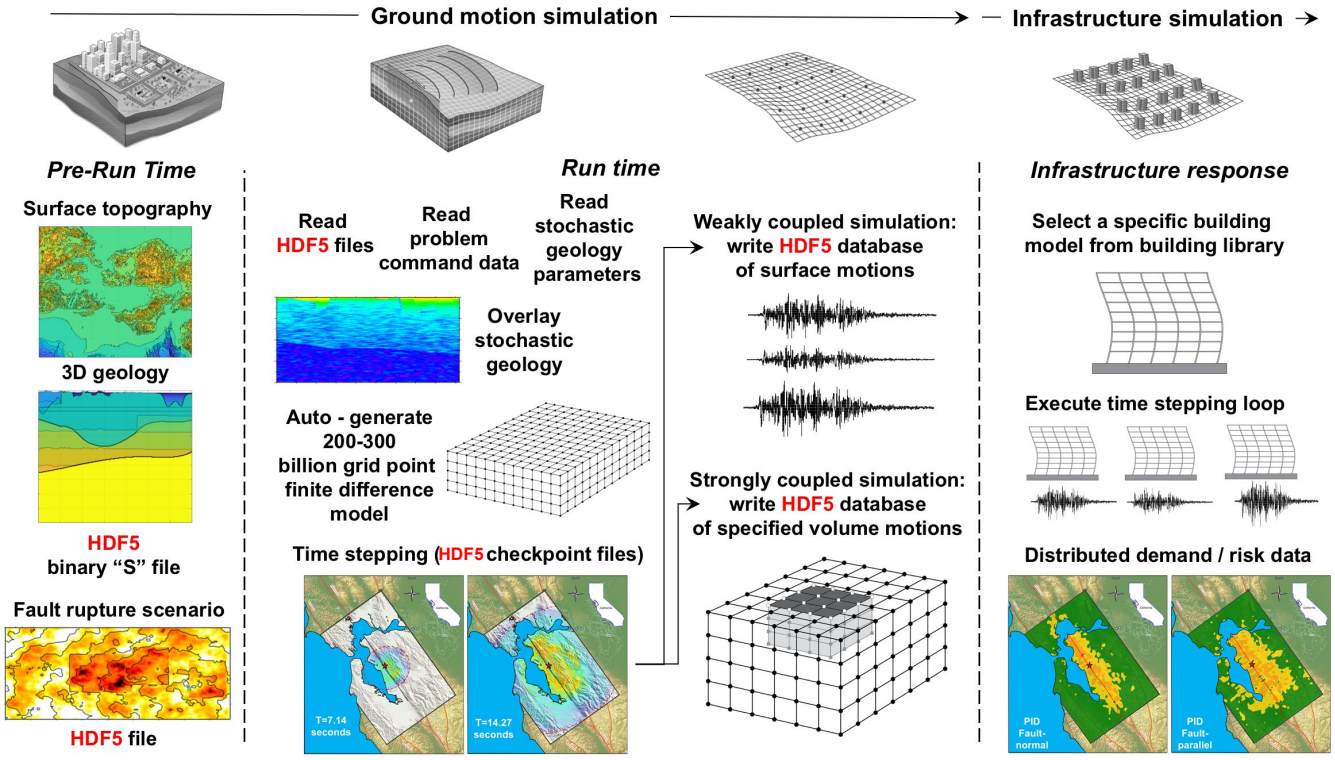
What is the distribution of infrastructure risk?

EQSIM I/O and Data Management Goals

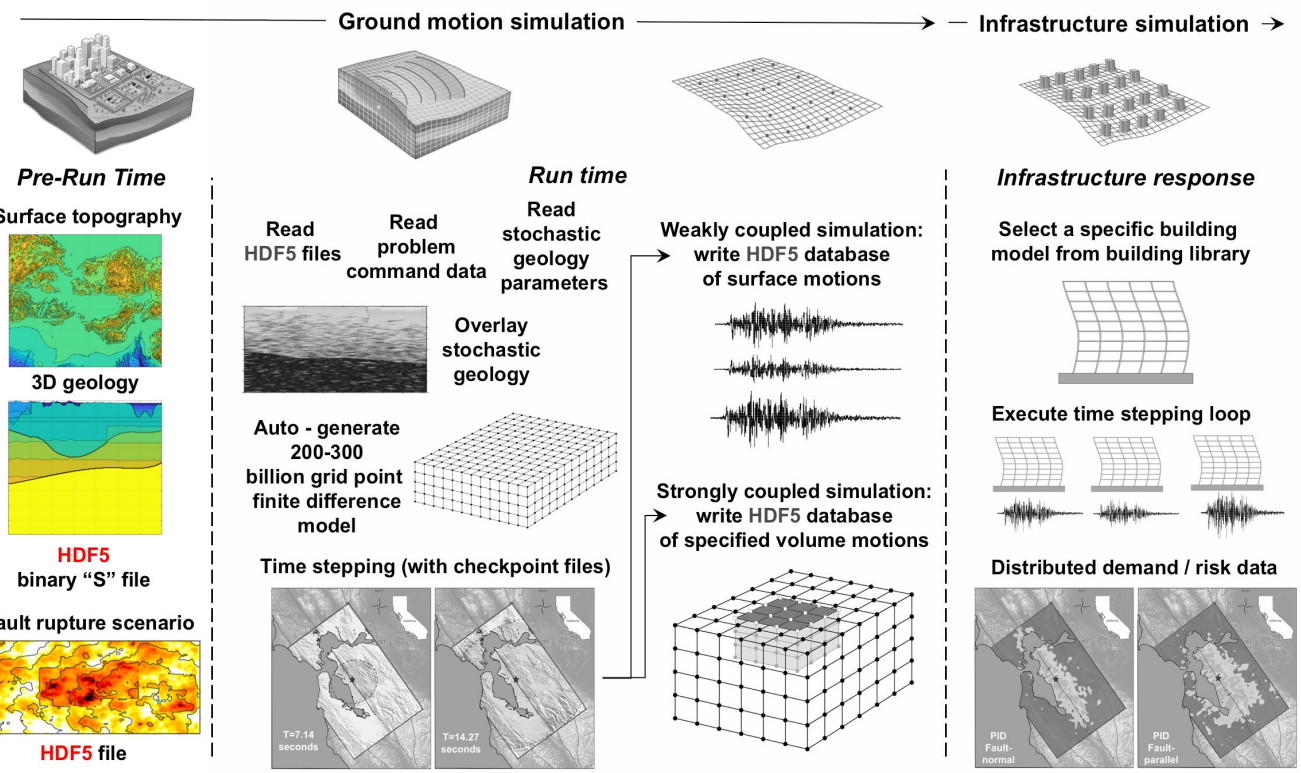
- Moving toward exascale earthquake simulations, I/O and data management becomes increasingly challenging
  - Increased volume of input and output data significantly affects the overall simulation run time.
  - New requirements for I/O and data emerge as simulation code evolves.
  - Easy-to-access data format enables efficient analysis and data sharing.
  - New techniques such as compression is required to enable large scale data analysis.

- HDF5 integration is important to improve the workflow efficiency
  - HDF5 is a high performance software library and file format.
  - HDF5 supports heterogeneous data, easy sharing, cross platform, fast I/O, and keep metadata with data.

EQSIM Workflow

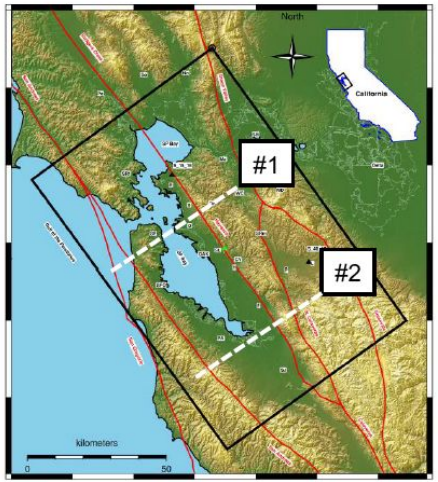


Input Model Data

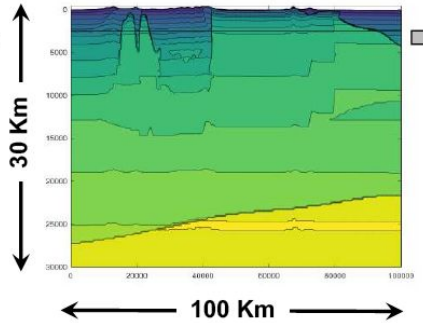


Material Sfile

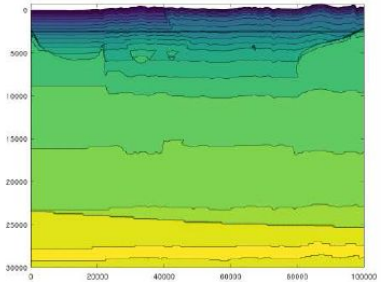
USGS geologic data



Earth Cross Section #1



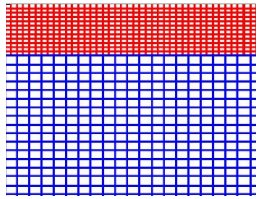
Earth Cross Section #2



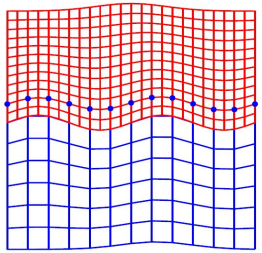
Vertical scale exaggerated

Newly developed "S" file for the 3D geologic model

- Enhanced material model inspection and visualization with the HDF5 format
- Enables material model output for both forward and inverse problems with SW4
- Allows converting existing material model data to an S file with SW4 grid and mesh refinement levels
- Allows horizontal and/or vertical down sampling to reduce the data size with acceptable interpolation error bounds



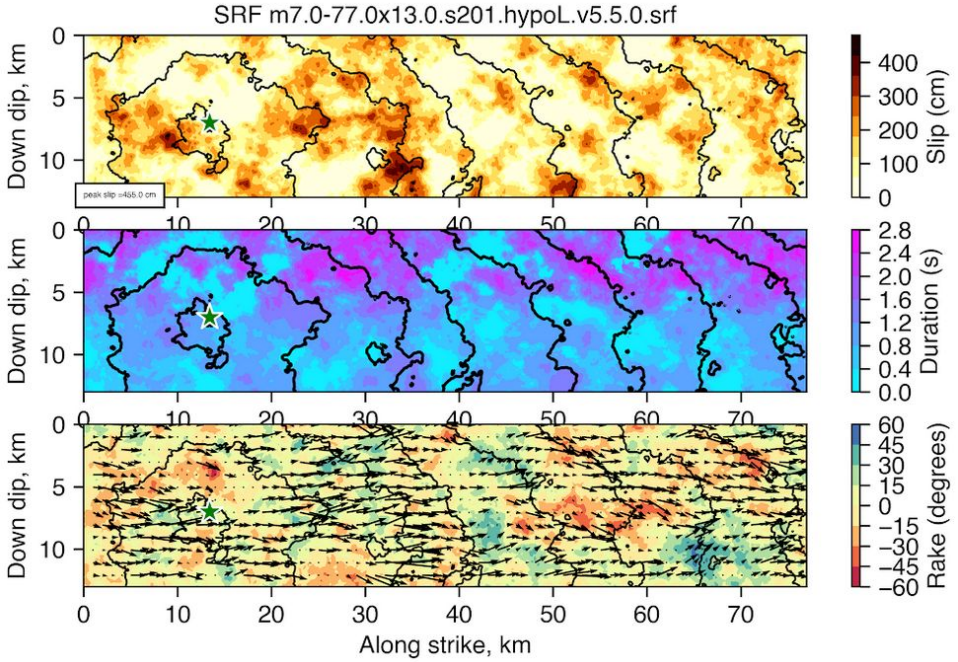
R file



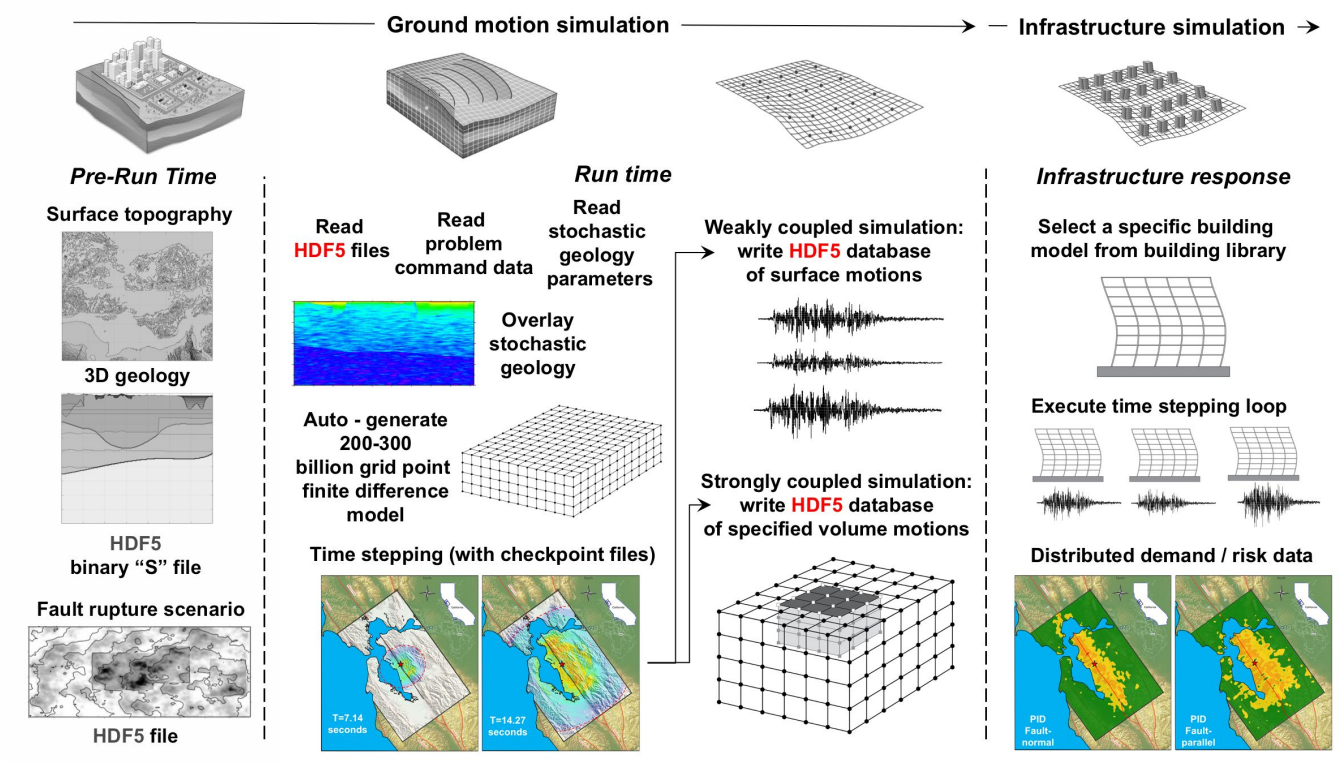
S file

Rupture-HDF5

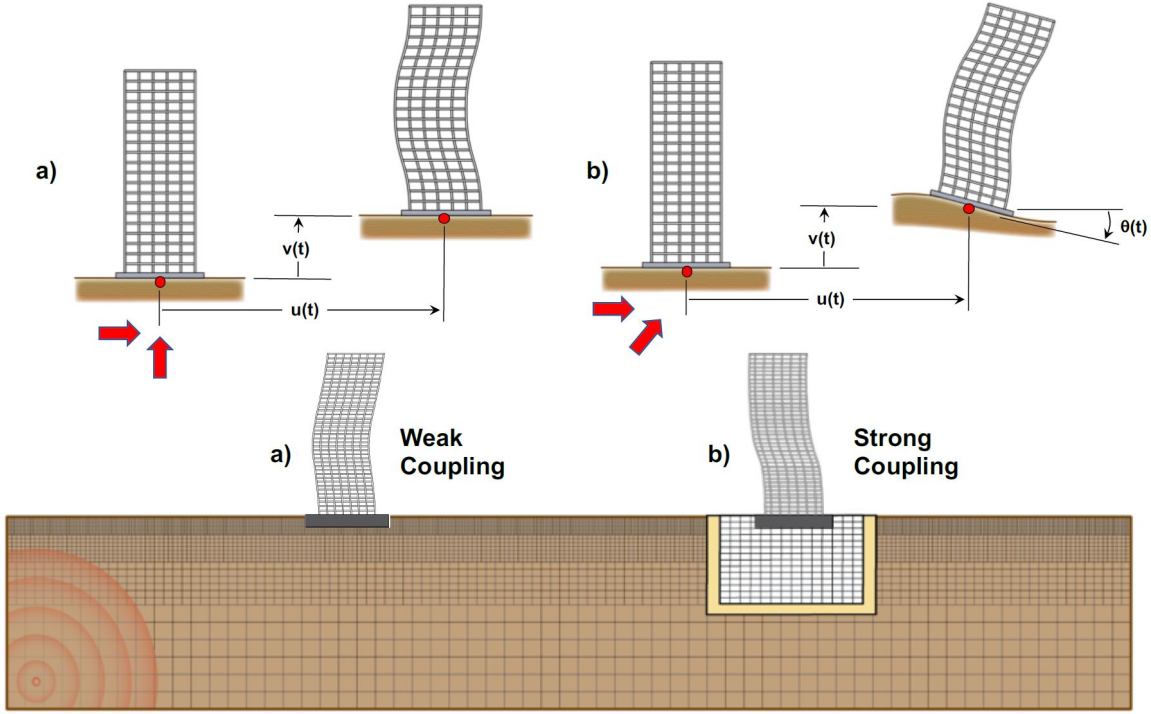
- Originally in SRF format (ASCII)
- Converted HDF5 file is ⅓ the original size and can be read much more efficiently in parallel (hours to minutes in 922 and 3600 Summit nodes run).
- Easy to share and visualize with Python, MATLAB, R, etc.



Output and Analysis

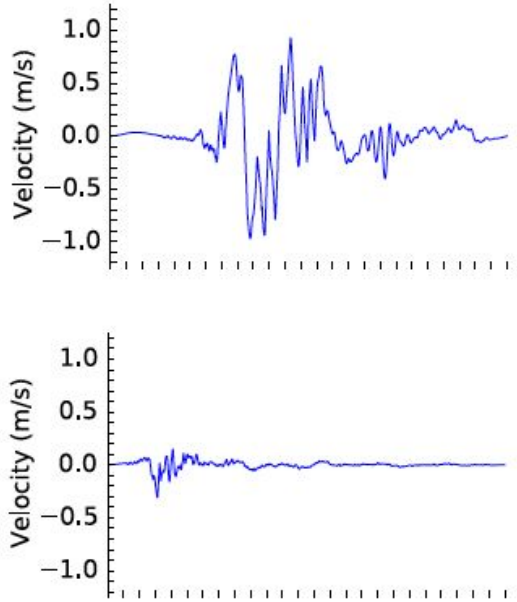


Weak and Strong Code Coupling

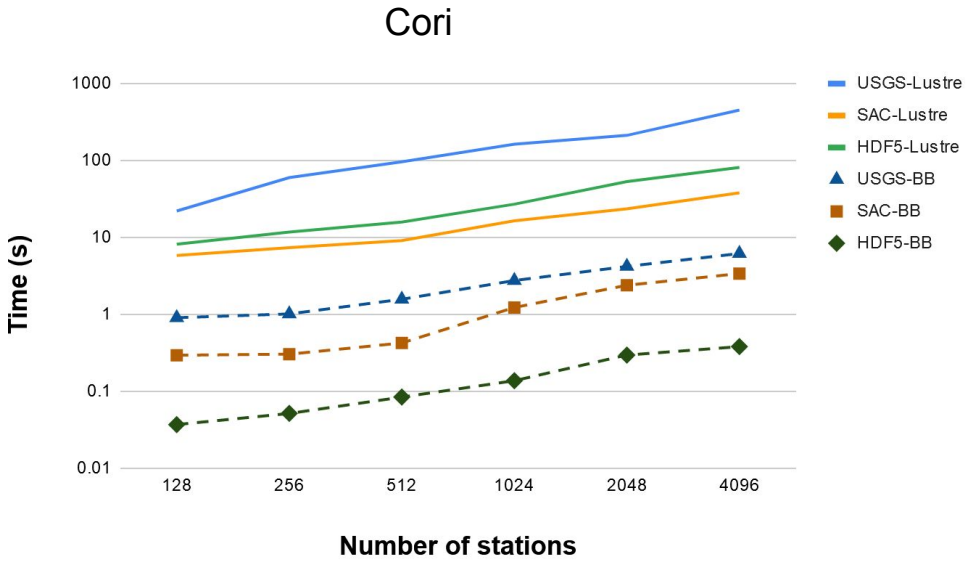


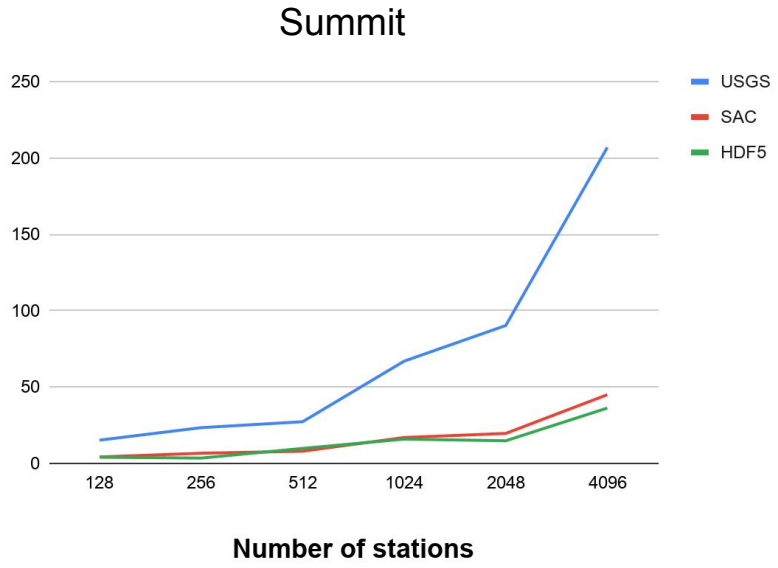
SAC-HDF5

- Time-history of receiver stations at user-specified location.
- SAC format
  - 3 files per station (x,y,z direction), may result in more than 10k files for a large scale run.
  - Each file is relatively small (<10MB)
  - Required special reader to parse data.
- SAC-HDF5 format
  - Single HDF5 file for all stations.
  - Allows down-sampling.
  - Easy to read and visualize.
  - Write time is comparable to SAC when writing to Lustre and GPFS, up to 9x faster to burst buffer.



I/O Time Comparison

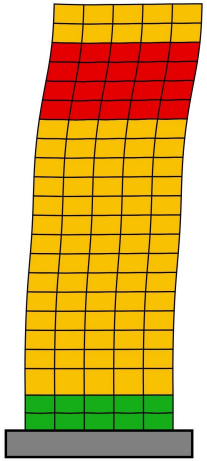




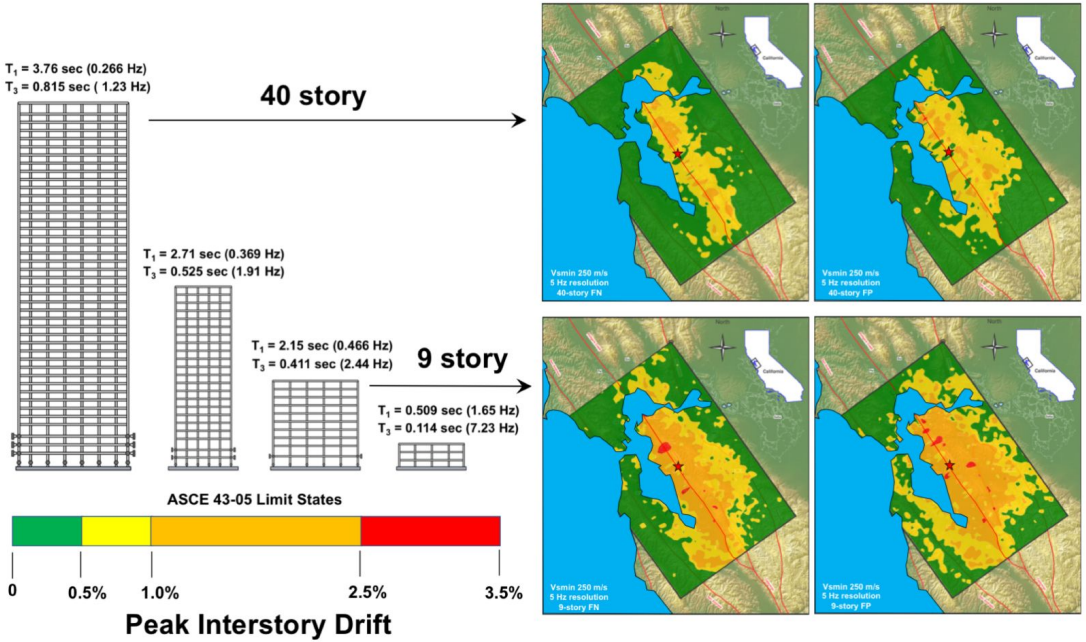
HDF5 is *1.5-2.3X slower* than SAC on Lustre
*5-9X **faster*** than SAC on Burst Buffer (BB)

HDF5 is up to *1.2X faster* than SAC

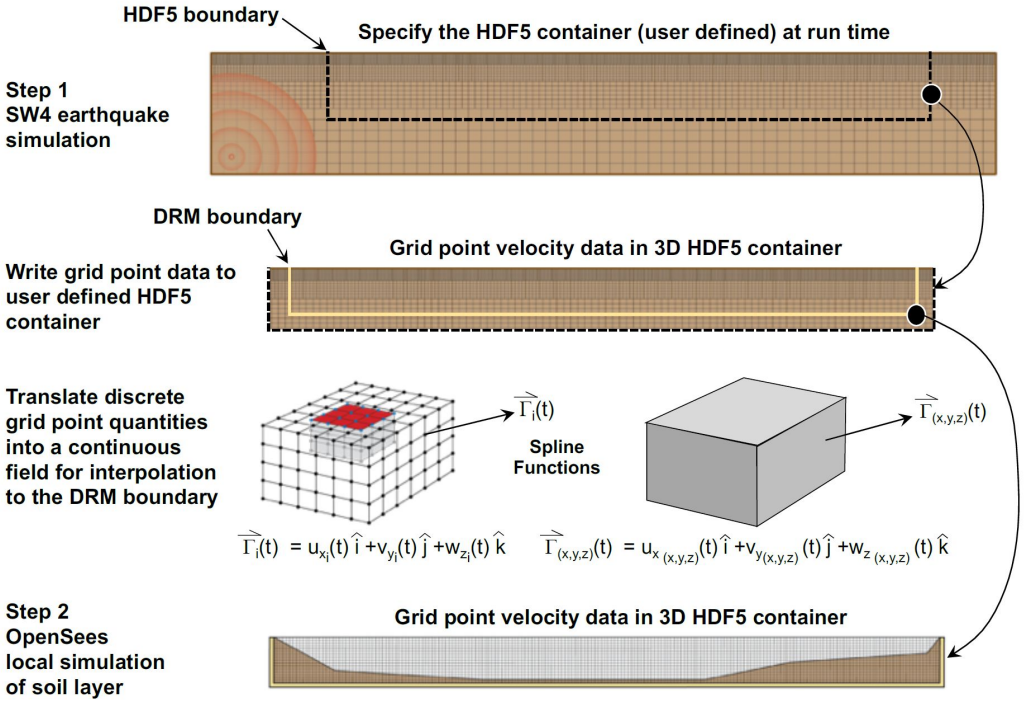
Weak Coupling of Time-history Data



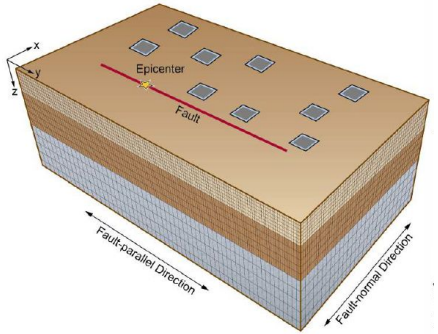
In the weak coupling case the surface ground motions computed at a point on the earth surface are applied directly and uniformly across the base of an infrastructure model



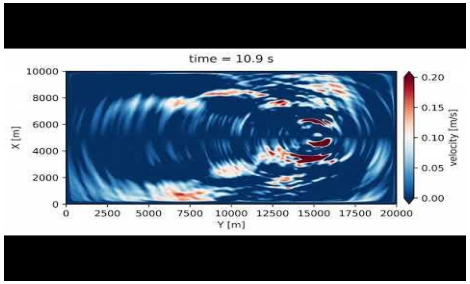
Strong Coupling of Geophysics and Engineering Models



Compression

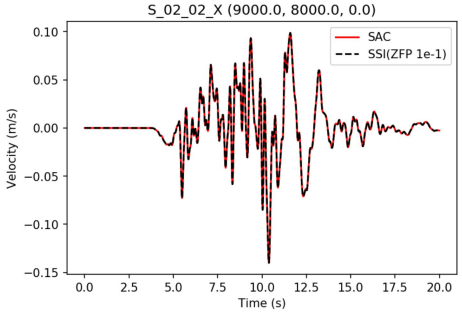


HDF5 output with compression enables saving velocity time-history at *every grid point* in a near-surface volume (e.g. down to 150m depth)



ZFP is a library for compressing floating-point arrays with lossless and lossy but optionally error-bounded compression.

Config	CR	HDF5 File Size
Default	1	(76 TB)
zfp-acc=1e-2	261	**293 GB**
zfp-acc=1e-1	494	**155 GB**




```
# pip install hdf5plugin
import h5py
import hdf5plugin

h5file = h5py.File('data.h5')
data = h5file['vel'][:]
plot...
h5file.close()
```

ZFP: *https://computing.llnl.gov/projects/zfp*

Summary

Integration of HDF5 greatly improves the EQSIM workflow efficiency to generate, process, analyze, and visualize data.

- HDF5's self-describing format and portability allows convenient data sharing among scientists.
- Various programming language interfaces and tools from HDF5 provide easy data access.
- Improved I/O performance for both input and output data.
- Reduced number of time-history files from thousands to 1 per simulation.
- Transparent compression capability allows saving and analyzing more data pain-free.

Thanks!

Questions?