SWQU: Improving Space Weather Predictions with Data-Driven Models of the Solar Atmosphere and Inner Heliosphere

N.V. Pogorelov,[1] C. N. Arge,[2] R. Caplan,[3] P. Colella,[4] C. Gebhard,[4] S. Jones,[2] T. K. Kim,[1] J. Linker,[3] T. Singh,[1] B. Van Straalen,[4] L. M. Upton,[5] C. Downs,[3] M. Stulajter,[3] M. S. Yalim[1]

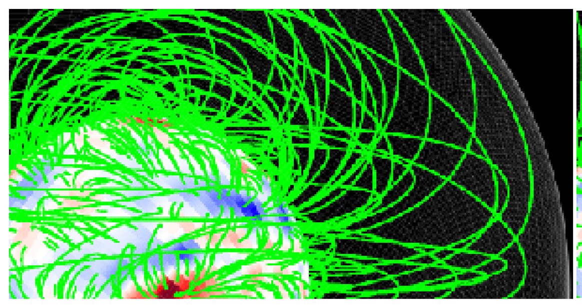
[1]University of Alabama in Huntsville, [2]Goddard Space Flight Center, [3]Predictive Science, [4]Lawrence Berkeley National Laboratory, [5]Southwest Research Institute

Abstract

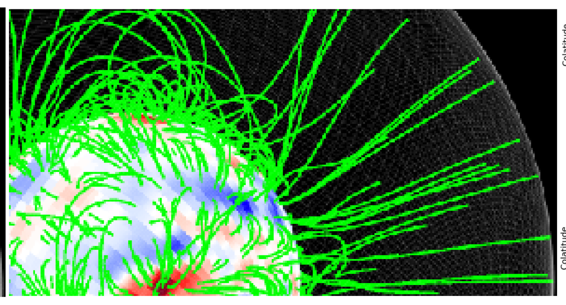
Our science goal is to develop a new software to support data-driven, time-dependent models of the solar atmosphere and heliosphere suitable for near real-time predictions of the SW properties at Earth's orbit and in the interplanetary space.

1. A new Open surface Flux Transport (OFT) model which evolves information to the back side of the Sun and its poles and update the model flux with new observations using data assimilation methods.
2. A new potential field solver (POT3D) combined with the output from the traditional WSA model, and with remote coronal and in situ solar wind observations. WSA and the new potential field solver (PFSS and PFCS) are both be validated using the maps from the OFT.
3. A highly parallel, adaptive mesh refinement (AMR) code (HelioCubed) for the Reynolds-averaged, ideal MHD equations describing the solar wind flow in the region between R = 10−20 $R_\odot$ and 1–3 au. These equations will be accompanied by the equations describing the transport of turbulence. We are building on the Multi-Scale Fluid-Kinetic Simulation Suite (MS-FLUKSS) collaboratively developed at UAH using the Chombo AMR framework (Colella et al., 2007). The new version of our software is built on Chombo 4 and allow us to perform simulations with the fourth order of accuracy in time and space, and use cubed spheres to generate meshes around the Sun.
4. The developed software will run on both GPUs and CPUs and will be made publicly available.

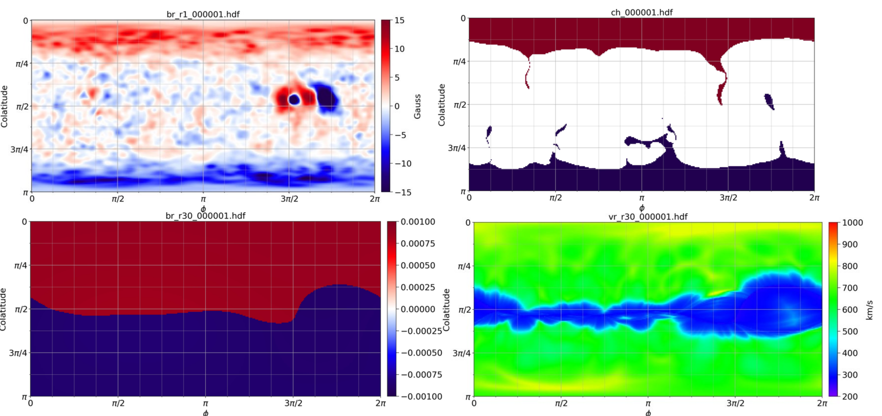
POT3D (Caplan et al., 2021) is a high performance Fortran-90 MPI code that solves Laplace's equation using finite differences on a non-uniform spherical grid. It has been ported to GPUs using OpenACC. The code is open source. It is available to public via the GitHub repository.



Potential field (closed wall)

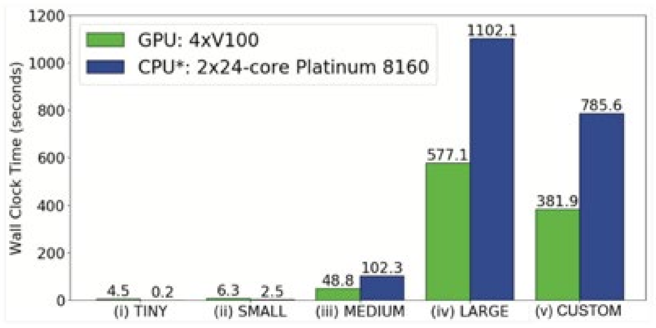


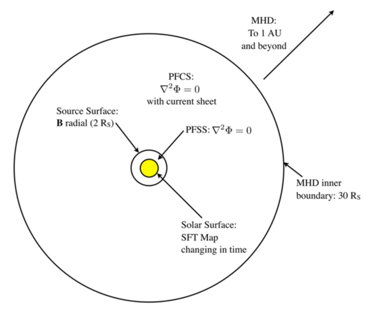
Potential Field Source Surface

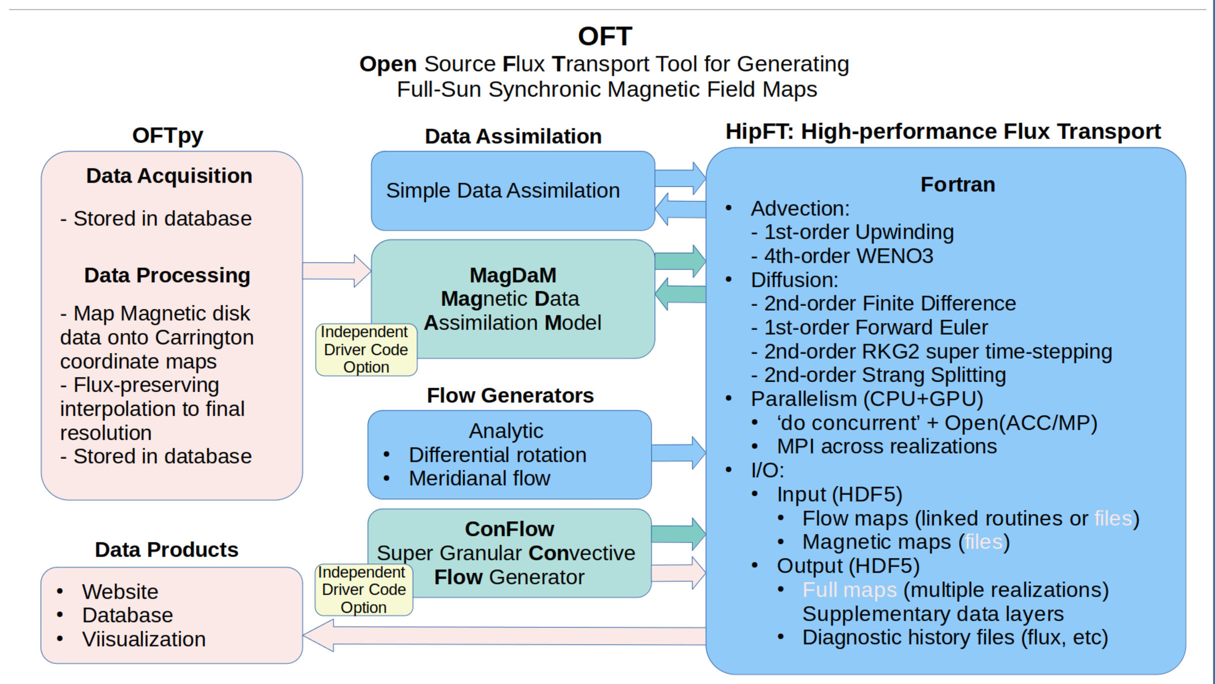


Demonstration of POT3D & WSA

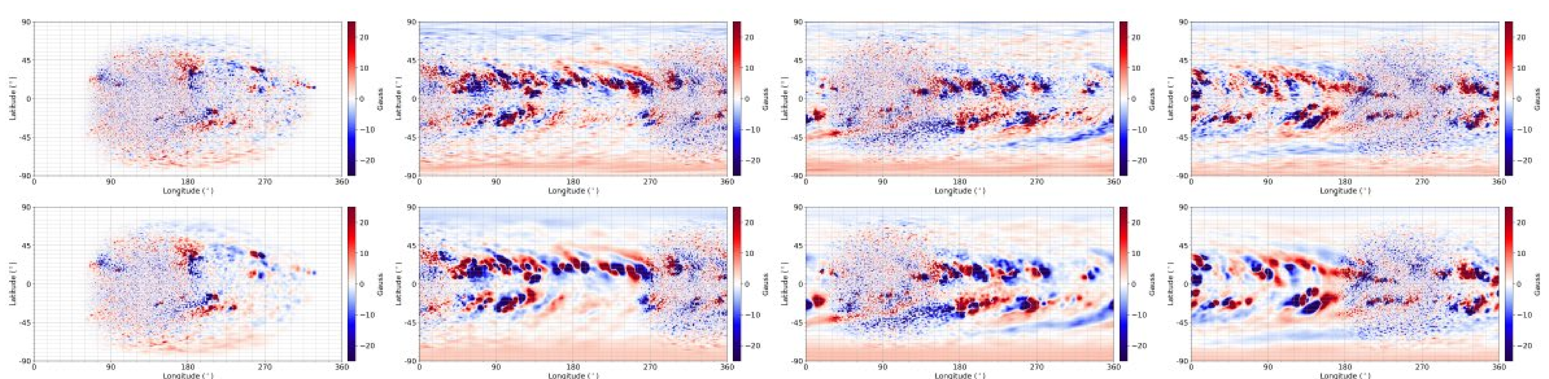
Case	Resolution (r x theta x phi)	# cells	Theta-Phi Mesh
(i) Tiny	27x90x180	0.4x10⁶	uniform
(ii) Small	54x180x360	3.5x10⁶	uniform
(iii) Medium	135x450x900	55x10⁶	uniform
(iv) Large	207x900x1800	335x10⁶	uniform
(v) Custom	216x742x1095	175x10⁶	nonuniform





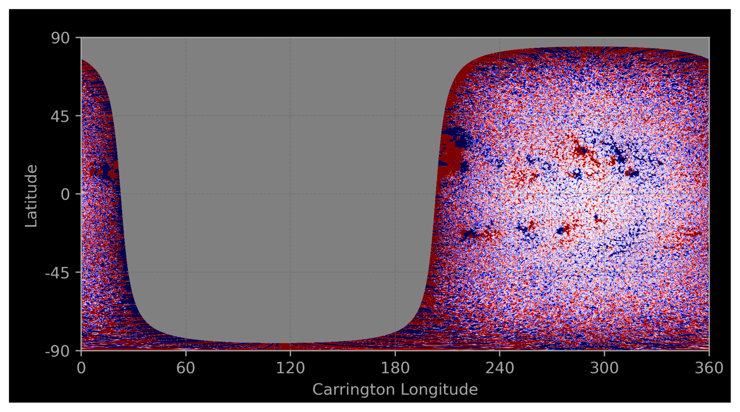


In addition to OFTpy and data assimilation modules, OFT involves a high-performance (implicit) and high-accuracy (2nd order of accuracy in space in time) FT module (HipFT). A key feature of HipFT that makes it high performance is the ability to run the code on GPU accelerators. HipFT models advection and diffusion of magnetic flux over the surface of the Sun. The AFT model is unique in that it explicitly models the turbulent surface flows produced by convection (Hathaway et al., 2010; Hathaway & Upton, 2021). Together with our progress towards including supergranular convective flows, these make it possible to dramatically improve the quality and generation time of full-Sun maps. As an example, OFT performs a few tens of times faster than the AFT model. Snapshots of HipFT data simulation test. The top images are from the simulation with no diffusion, while the bottom are with a diffusion of 200 km²/s. The snapshots are shown at times 9.5, 349.5, 694.5, and 874.5 days.



Data Acquisition and Processing: OFTpy

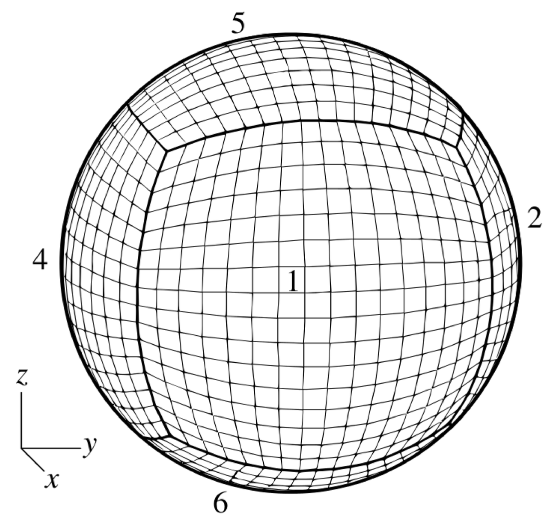
We have developed OFTpy, a python codebase that facilitates acquisition and processing of data products to a computation-ready format for HipFT. The current code supports full processing of JSOC HMI M720s line-of-sight (LOS) magnetograms to a radial-flux Carrington Rotation (CR) Map. This extensible software will also serve as a prototype/example for future users to incorporate magnetograms from alternative observatories. For data acquisition of HMI LOS magnetograms, OFTpy acts as a Python wrapper for product query and download from the Stanford JSOC). Figure at the bottom shows an example. We are presently comparing this mapping to a mapping product developed by the Stanford team for use by ADAPT.



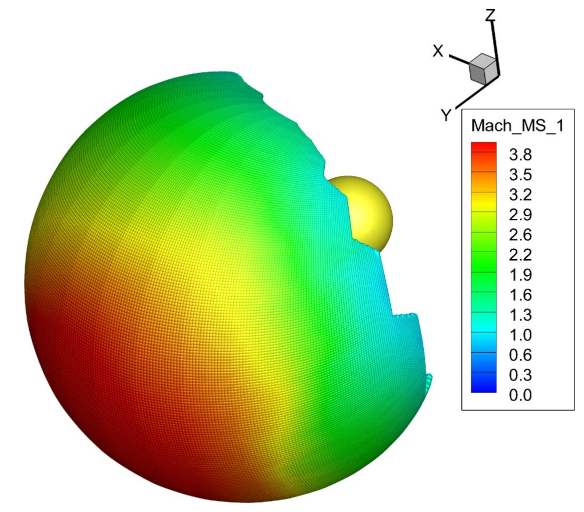
An HMI 720s magnetogram, observed on 2012/01/15 00:59:52, converted to radial flux and projected into Carrington Coordinates with OFTpy.

The Inner Heliospheric Model: HelioCubed

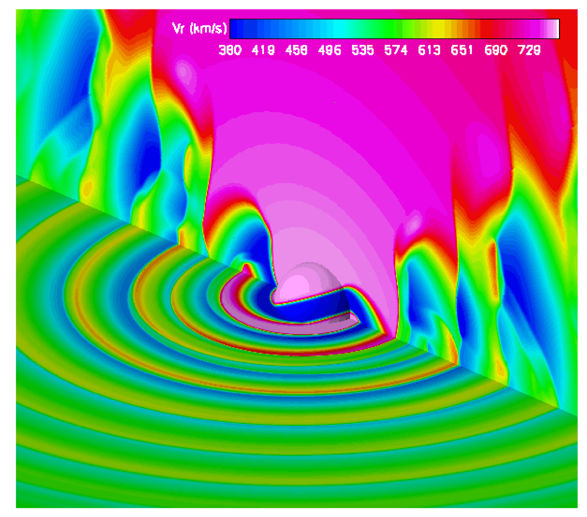
1. We use finite volume method to solve hyperbolic, Reynolds-averaged MHD equations in conservative form with 4th order of accuracy in space and time.
2. The average values of primitive variables are calculated on R and L side of the faces with the 4th order accuracy and a Riemann problem solver is used to find the 4th order accurate fluxes through these faces.
3. The 4th order accurate RK method is used to integrate the equations with time. Limiters specially designed for the 4th order accurate methods are used (McCorquodale et al 2011).
4. Our approach solving this problem is based on the following ideas: a cubed-sphere representation of space, that has most of the same advantages as the widely-used spherical coordinate system, but does not have a polar singularity; a method-of-lines discretization of the evolution equations, with high-order accurate discretizations in both space and time for non-linear problems; and block-structured AMR.



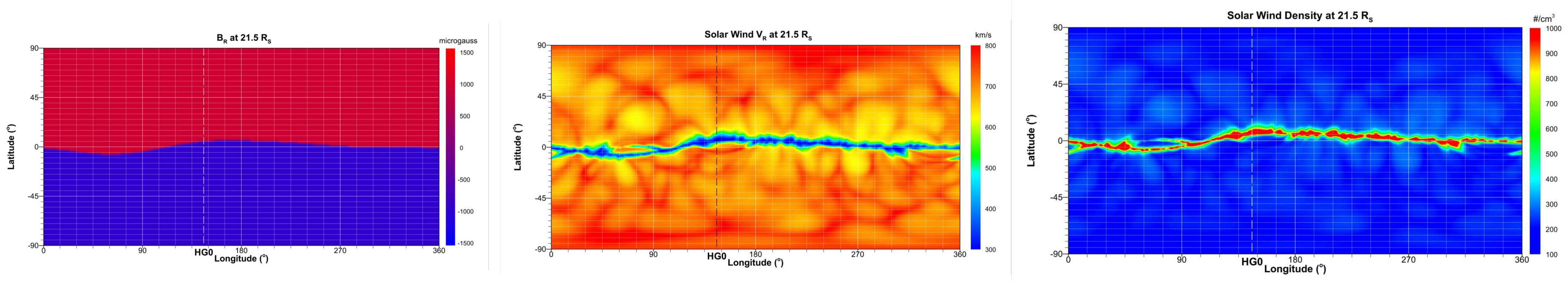
Mapped-multiblock gridding of the surface of a sphere



A shock in front of the model CME

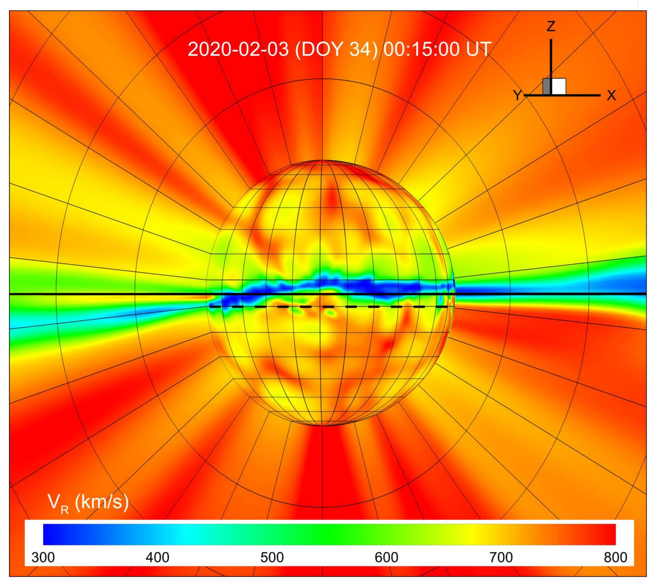


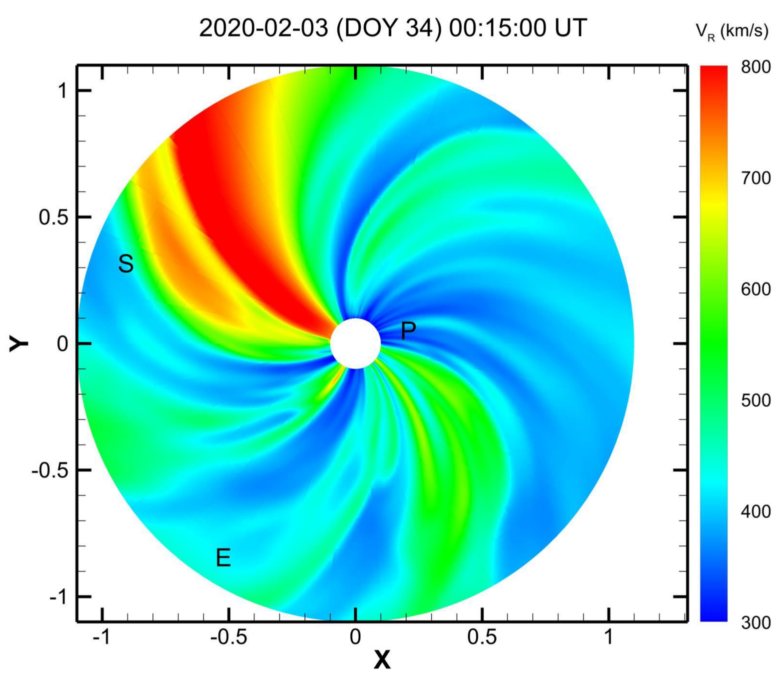
Corotating interaction regions



A preliminary simulation in the IH using OFT-POT3D-WSA-MS-FLUKSS. Top row: the initial distributions at the inner boundary at 21.5 Rs.

(Right panels): the 3D distribution (left panel) and the equatorial cut of the SW. The projected positions of Earth, Parker Solar Probe, and Stereo A are indicated with letters E, P, and S, respectively.





Software Architecture: Chombo for GPUs

Our software approach is based on Chombo, a C++ library developed at LBNL for implementing structured grid algorithms on distributed-memory multiprocessors. The design of Chombo is based on the following:

• Simulation data is defined on unions of rectangular patches in one of a nested hierarchy of uniform grid index spaces. The union of rectangles induces a domain decomposition onto a distributed memory set of processes.

• The fundamental unit of performance-critical computation is to apply discrete operators to data on a single patch. On CPU-based systems, this has been done using Fortran, or a low-level C-subset of C++, in order to obtain high performance.

While this has been a successful strategy for CPU-based distributed-memory systems, the extension to GPU-based systems introduces two new features that are not present in the CPU-based model:

1. Heterogeneous processors and memory on a node: there are both CPU and GPU systems on a single node, each with their own memory.
2. High-performance for floating point intensive operations is to be obtained using SIMT parallelism on the GPUs.

Acknowledgments. This work is supported jointly by NSF and NASA through the program "Next Generation Software for Data-driven Models of Space Weather with Quantified Uncertainties (SWQU)"