Predictive Science Inc.

NASA SwRI AFRL

HipFT v1.0

An Open Source High-Performance Flux Transport Model

Ronald M. Caplan[1], Miko M. Stulajter[1], James A. Turtle[1], Jon A. Linker[1], Cooper Downs[1], Lisa A. Upton[2], Charles Nicholas Arge[3], Raphael Attie[3], Carl J. Henney[4], and Nikolai Pogorelov[5]

(1) Predictive Science Inc., (2) Southwest Research Institute, (3) NASA Goddard Space Flight Center, (4) Air Force Research Laboratory Kirkland AFB, (5) University of Alabama in Huntsville

Download Free NOW!

github.com/predsci/hipft

AGU23 SH13C 2686
San Francisco, CA & Online Everywhere
11-15 December 2023

INTRODUCTION

Maps of the global solar photospheric magnetic field play an important role in solar and heliospheric physics. Routine measurements of the surface field occur only along the Sun-Earth line. Flux transport models attempt to mitigate this limitation by modeling the surface evolution of the field.

Here we present the first public release of the High-Performance Flux Transport code (HipFT), which implements advection, diffusion, and data assimilation over the solar surface. It can produce multiple realizations of the evolving flux, allowing uncertainty quantification. The code is modular, so users can easily add custom flow, diffusion, source, and data assimilation models.

HipFT is the computational core of the upcoming Open Source Flux Transport (OFT) model, which will be a complete system for generating full-Sun magnetograms through acquiring & processing observational data, generating realistic convective flows, and running the flux transport model.

Research supported by NASA and NSF

MODEL

$B_r(t,\theta,\phi)$ Surface radial magnetic field

$$\frac{\partial B_r}{\partial t} = -\nabla_s\cdot(\mathbf{v}\,B_r) + \nabla_s\cdot(\nu\,\nabla_s B_r) + D + S$$

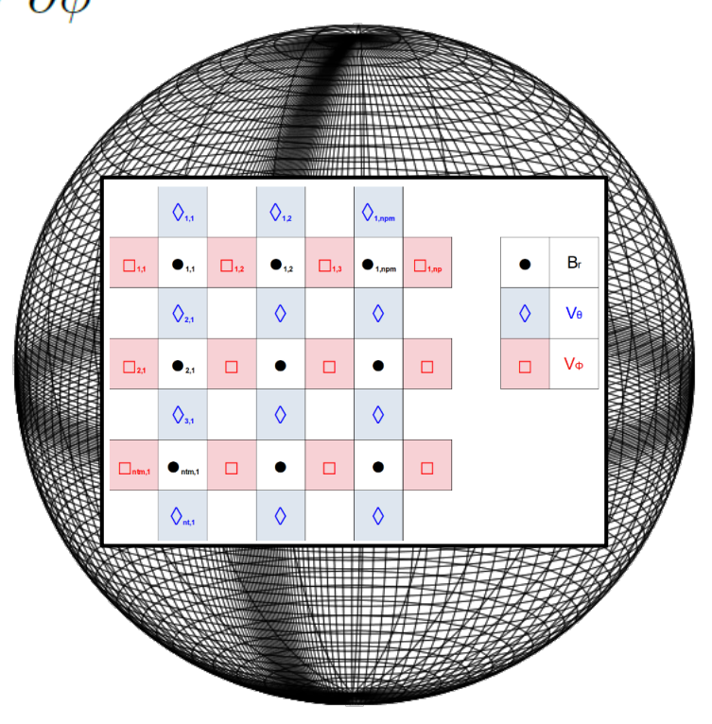
$$\nabla_s\cdot(\nu(\theta,\phi)\,\nabla_s B_r) = \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\nu(\theta,\phi)\,\sin\theta\,\frac{\partial B_r}{\partial\theta}\right) + \frac{1}{\sin^2\theta}\frac{\partial}{\partial\phi}\left(\nu(\theta,\phi)\frac{\partial B_r}{\partial\phi}\right)$$

$$\nabla_s\cdot(B_r\,\mathbf{v}_s(\mathbf{x},t)) = \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}(\sin\theta\,B_r\,v_\theta) + \frac{1}{\sin\theta}\frac{\partial}{\partial\phi}(B_r\,v_\phi)$$

$\mathbf{v}_s = (v_\theta, v_\phi)$ Flow velocities
D Data assimilation
S Sources (e.g. emerging flux)
ν Diffusion coefficient

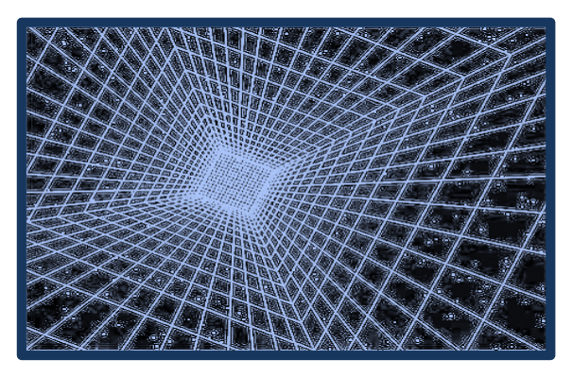
GRID

Non-uniform, logically rectangular, with field and velocities staggered



NUMERICAL METHODS

SPATIAL METHODS



FLOW: 1st-order Upwind
4th*-order WENO3 [Smit et. al. (2005)]

DIFFUSION: 2nd-order central finite difference

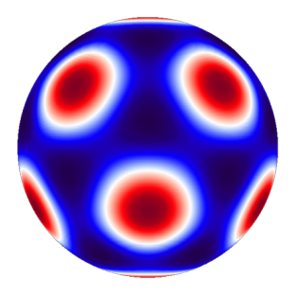
TEMPORAL METHODS:

Flow and diffusion are 2nd-order Strang operator-split [McLachla et. al. (2003)]

FLOW: 3rd-order RK3TVD [Gottlieb et. al. (1998)]

DIFFUSION: 2nd-order Runge-Kutta-Gegenbauer [Skaras et. al. (2021)]

VALIDATION



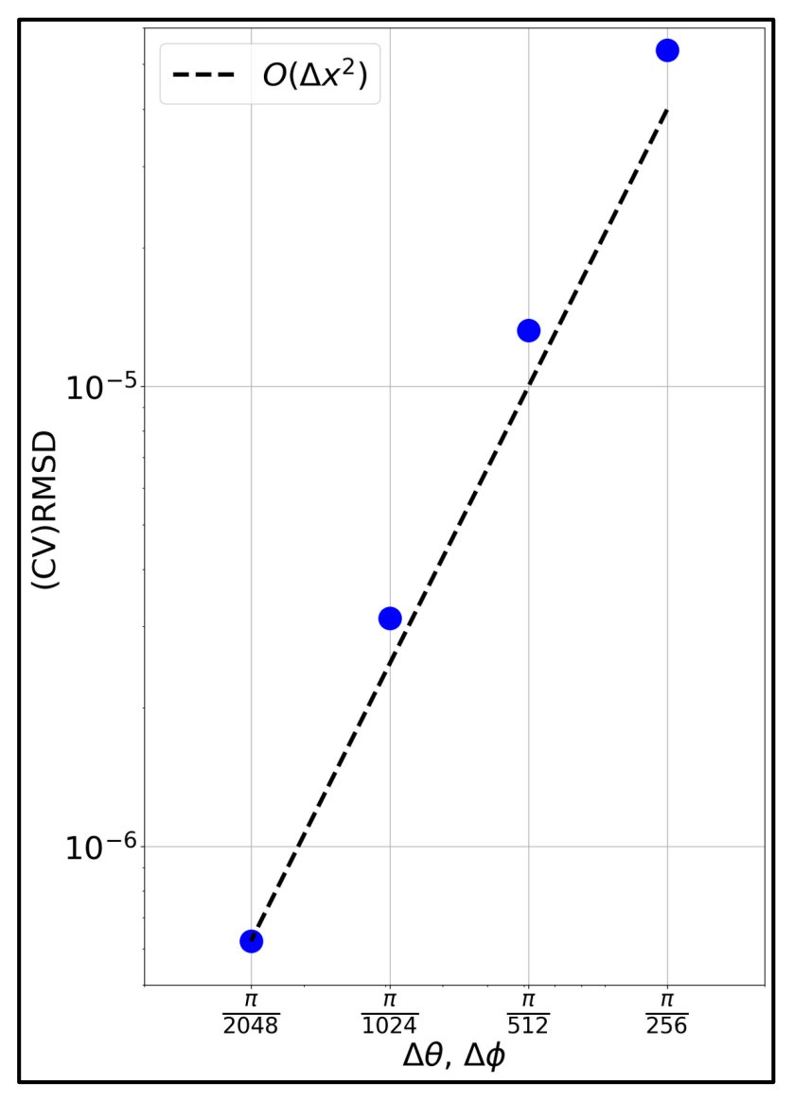
$$u(\theta,\phi,t) = 1000\,e^{-42\,\nu\,t}\left(Y_6^0(\theta,\phi) + \sqrt{\frac{14}{11}}\,Y_6^5(\theta,\phi)\right)$$

End Time: 28 days

$Y_l^m(\theta,\phi)$ Tesseral spherical harmonics

$\nu = 500\,\mathrm{km}^2/\mathrm{s}$

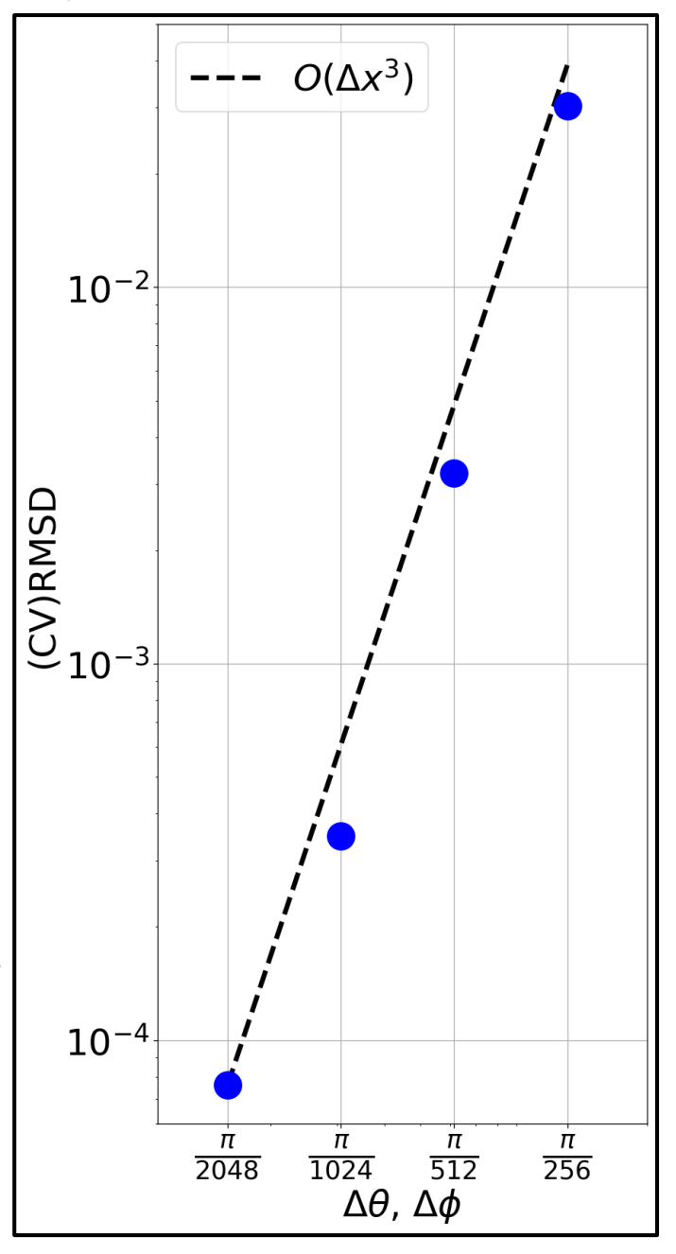
DIFFUSION ONLY



DIFFUSION+ADVECTION

Combine with rigid rotation with constant angular velocity in phi over 1 full rotation

$v_\phi = \Omega\,\sin\theta \quad \Omega = 1.8076\ldots\,\mathrm{km/s}$



CODE & PARALLELISM

- Written in Fortran 2023
- Fortran standard parallelism (do concurrent) for multi-core CPU or GPU
- OpenMP Traget for GPU data management
- MPI is used to parallelize realizations across multiple CPUs/GPUs

OpenMP MPI

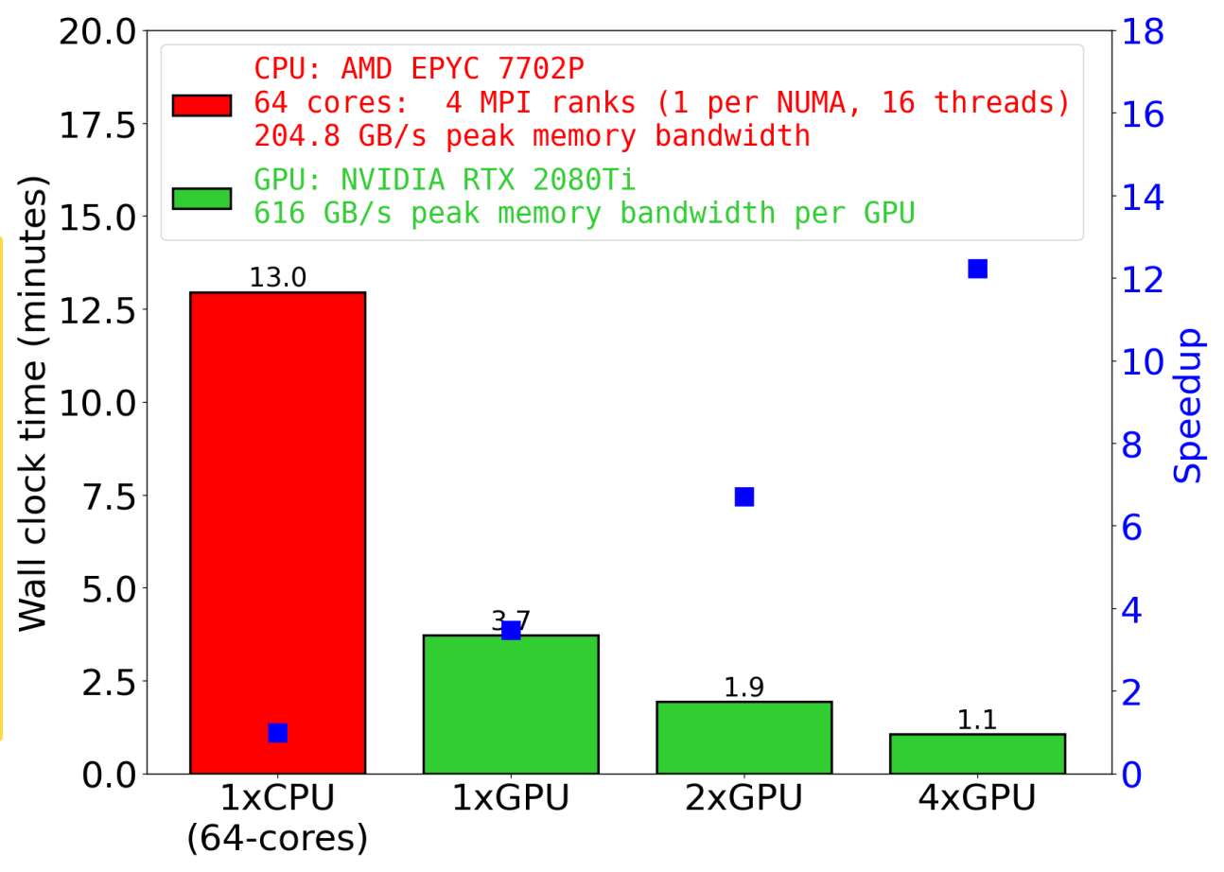
do concurrent

```
do concurrent (k=2:npm-1,j=2:ntm-1)
  y(j,k) =  coef(j,k,1)*x(j,  k-1) &
          + coef(j,k,2)*x(j-1,k  ) &
          + coef(j,k,3)*x(j  ,k  ) &
          + coef(j,k,4)*x(j+1,k  ) &
          + coef(j,k,5)*x(j,  k+1)
enddo
```

PERFORMANCE

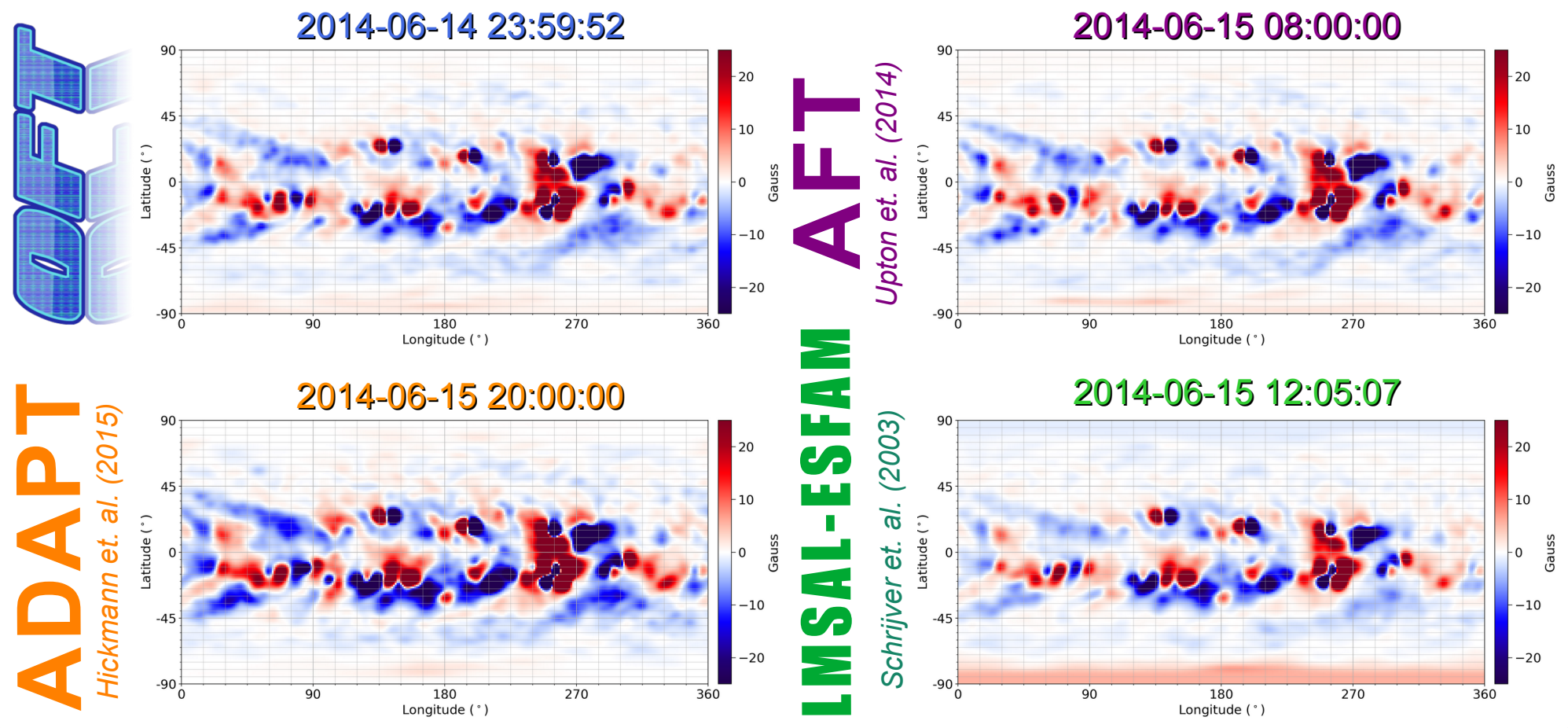
Test: 28-day run at 1024x512 with analytic flow models and diffusion. Eight realizations spanning various levels of diffusion and flow attenuation

In-house workstation:
EPYC 7702P 64-core CPU
Four RTX 2080Ti GPUs



COMPARISON TO OTHER FLUX TRANSPORT MODELS

- Maps from FT models are processed by interpolating to 300x150 resolution, flux balancing, and smoothing
- Note some models apply scaling factors to the HMI data



DATA ASSIMILATION

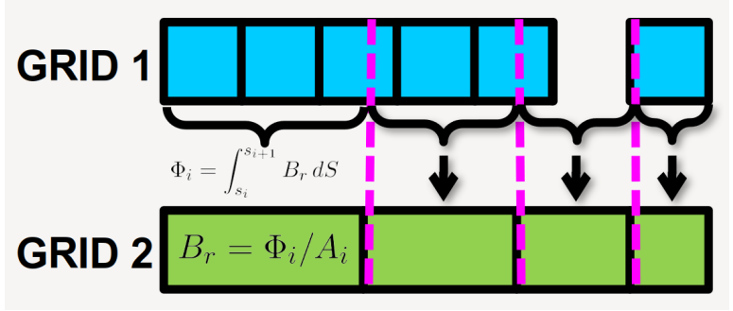
OFTpy

Acquire data (e.g. HMI M720s LOS through JSOC `drms` py package)

Convert line-of-sight field into radial field:
$B_r = B_{\mathrm{los}}/\mu$

Map to Carrington frame with resolution **10240 x 5120** to avoid under-sampling

Reduce size with flux-preserving integral binning:

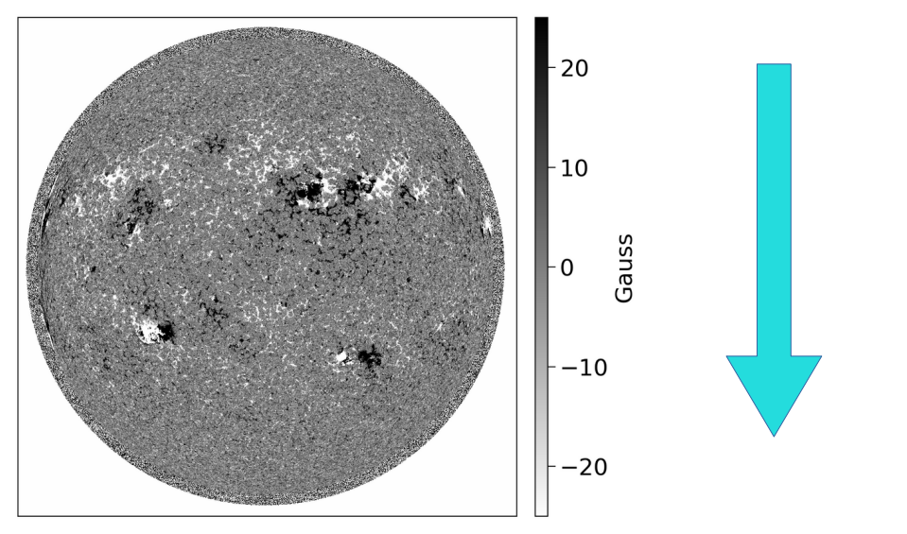


Set quality weights:
$\mu = \cos\theta_d \in [0,1]$
θ_d is the center to limb angle

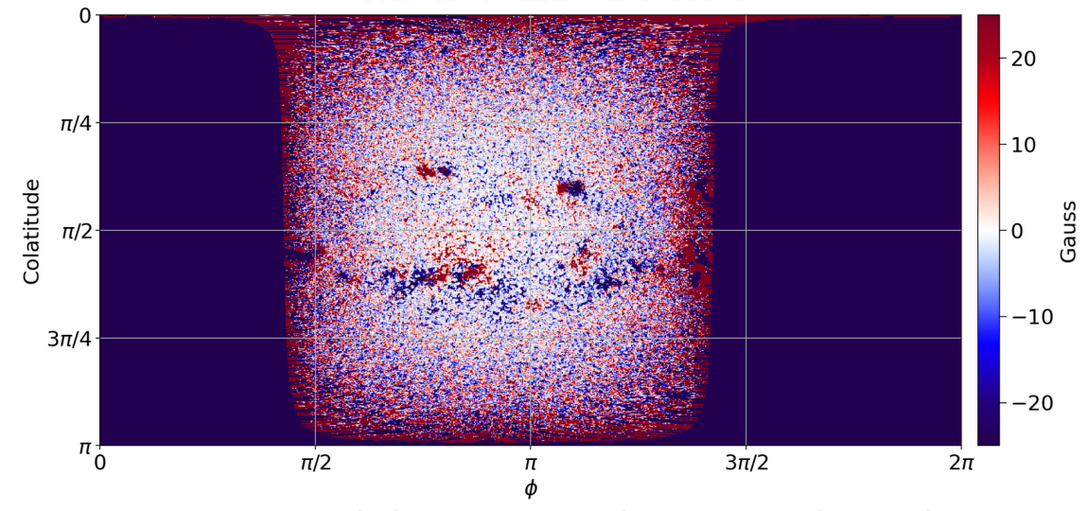
Use weights with power and cutoff parameters to assimilate data into HipFT:

$$F = \mu^{\alpha_\mu} \qquad \mu < \mu_{\mathrm{lim}}$$
$$F = 0 \qquad \text{o.w.}$$
$$D = \frac{1}{\Delta t}\left(F\,B_{r;d} - F\,B_r\right)$$

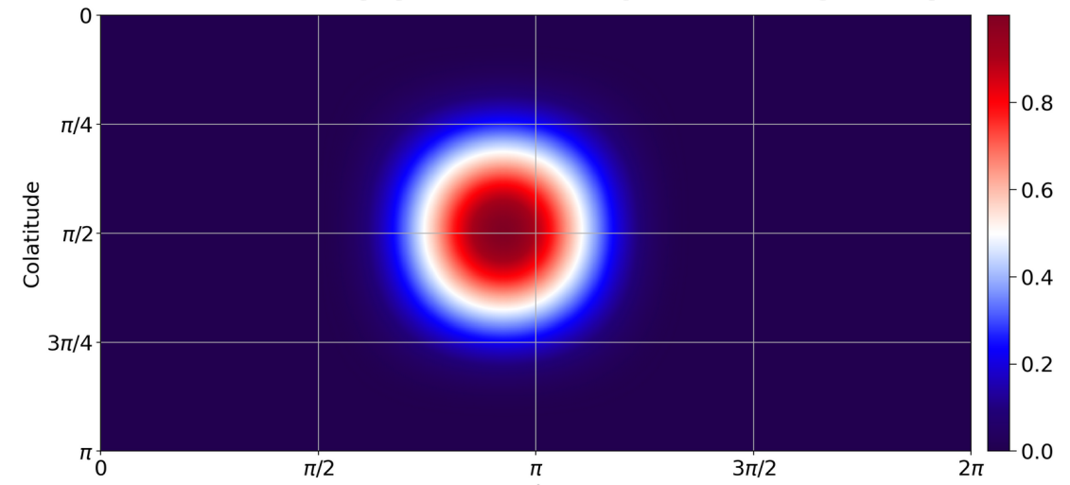
DISK LOS DATA



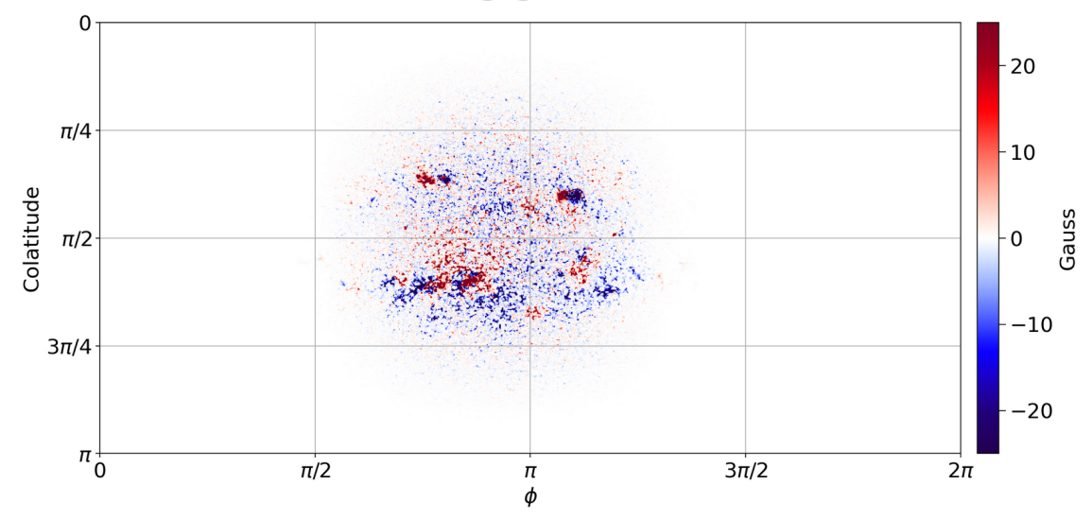
MAPPED DATA



DATA ASSIMILATION WEIGHTS

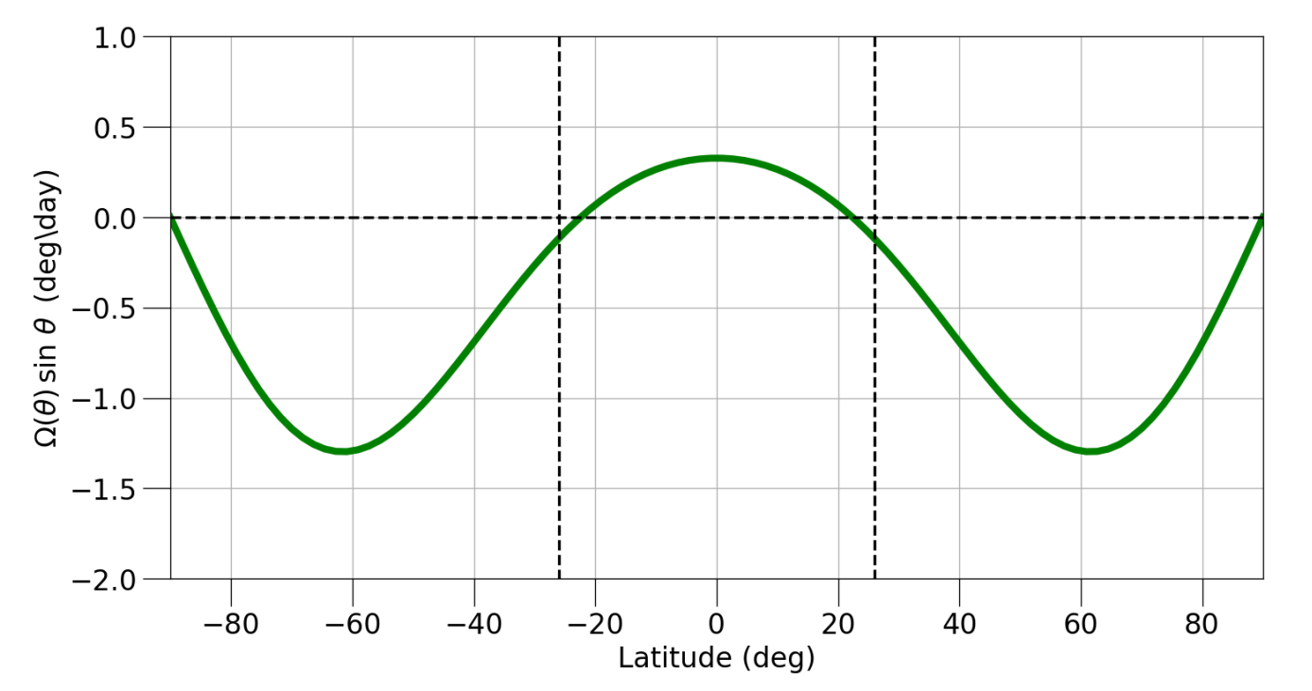


EXAMPLE ASSIMILATED DATA

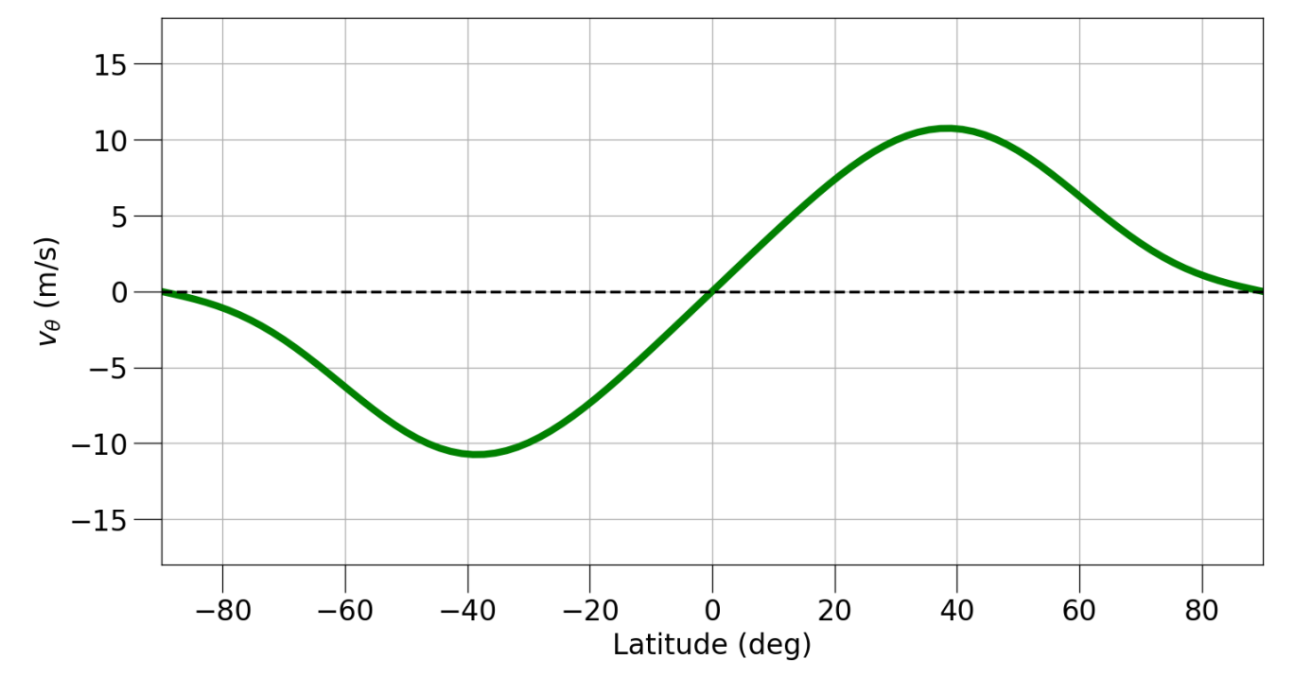


FLOW MODELS

Differential Rotation $v_\phi(\theta) = \left[d_0 + d_2\,\cos^2\theta + d_4\,\cos^4\theta\right]\,r\,\sin\theta$



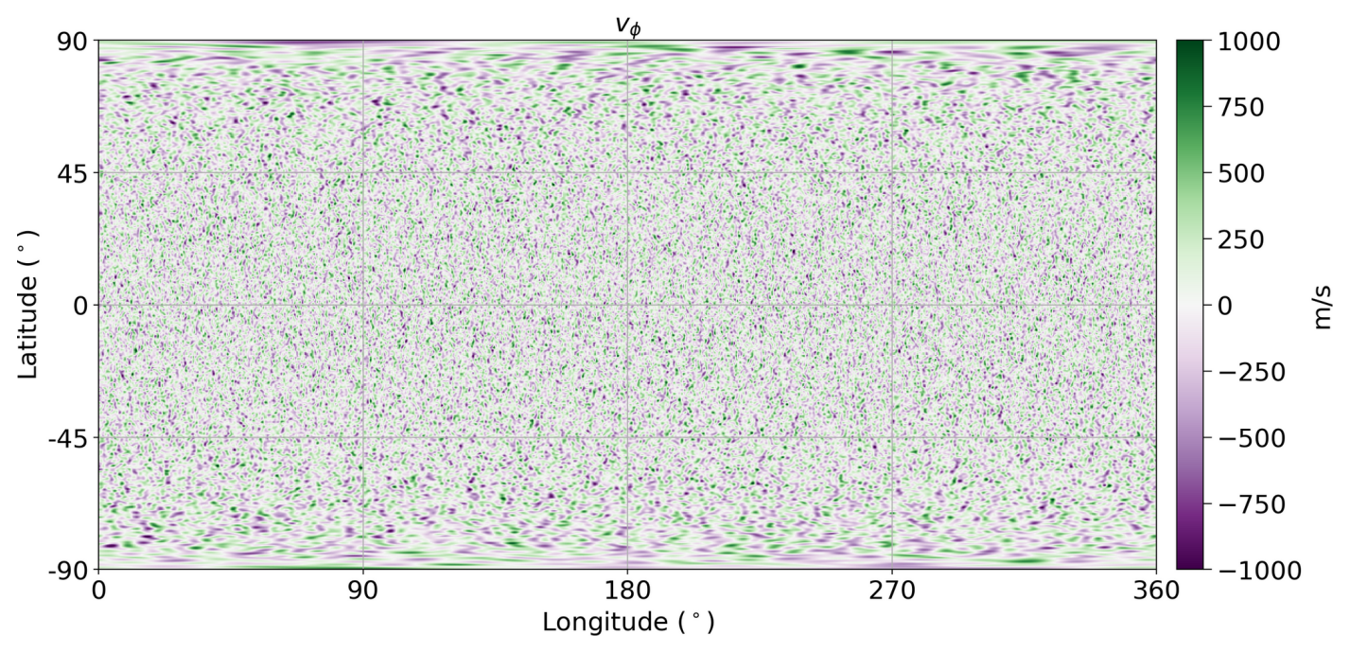
Meridianal Flows $v_\theta(\theta) = \left[-m_1\,\cos\theta - m_3\,\cos^3\theta - m_5\,\cos^5\theta\right]\,\sin\theta$



Flow Attenuation (reduces flow velocities in active regions)

$$v \rightarrow v\left[1.0 - \tanh\left(\frac{|B_r|}{B_0}\right)\right]$$

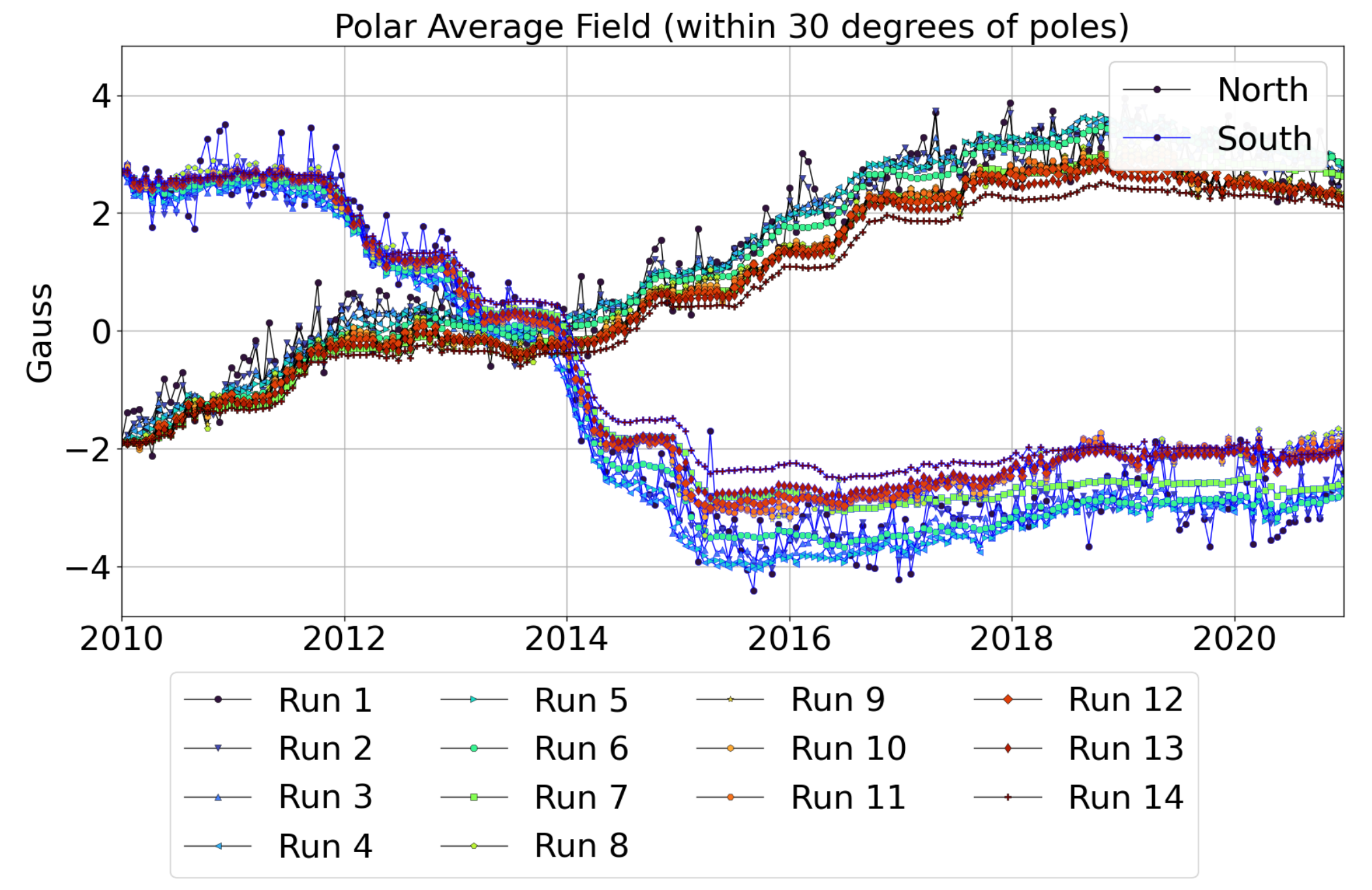
Super Granular Convective Flows [Hathaway et. al. (2010,2015)]



ConFlow

MULTIPLE REALIZATIONS

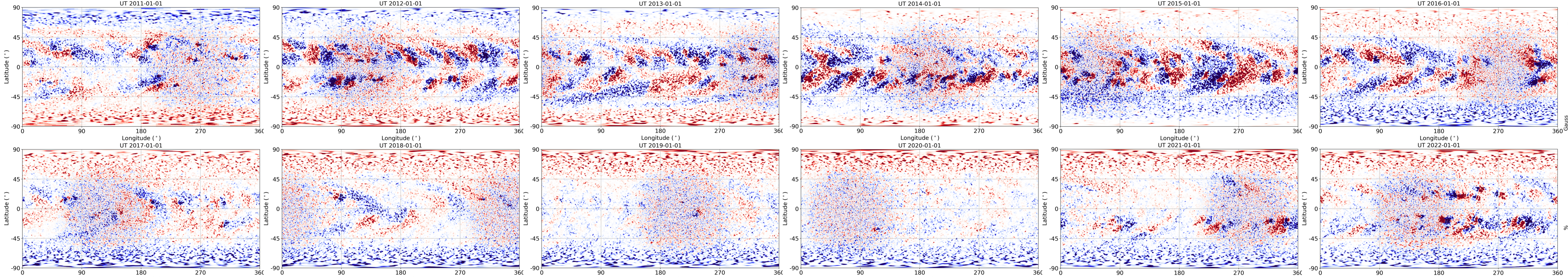
- Uncertainty quantification is very important when there are variations in observed data and model parameters
- To facilitate this, HipFT has been designed to compute many realizations of maps spanning several user-defined data assimilation and model parameters
- MPI is used to spread the realizations evenly across compute units (GPUs or CPU-sockets) to allow the efficient computation of a large number of realizations in a single run

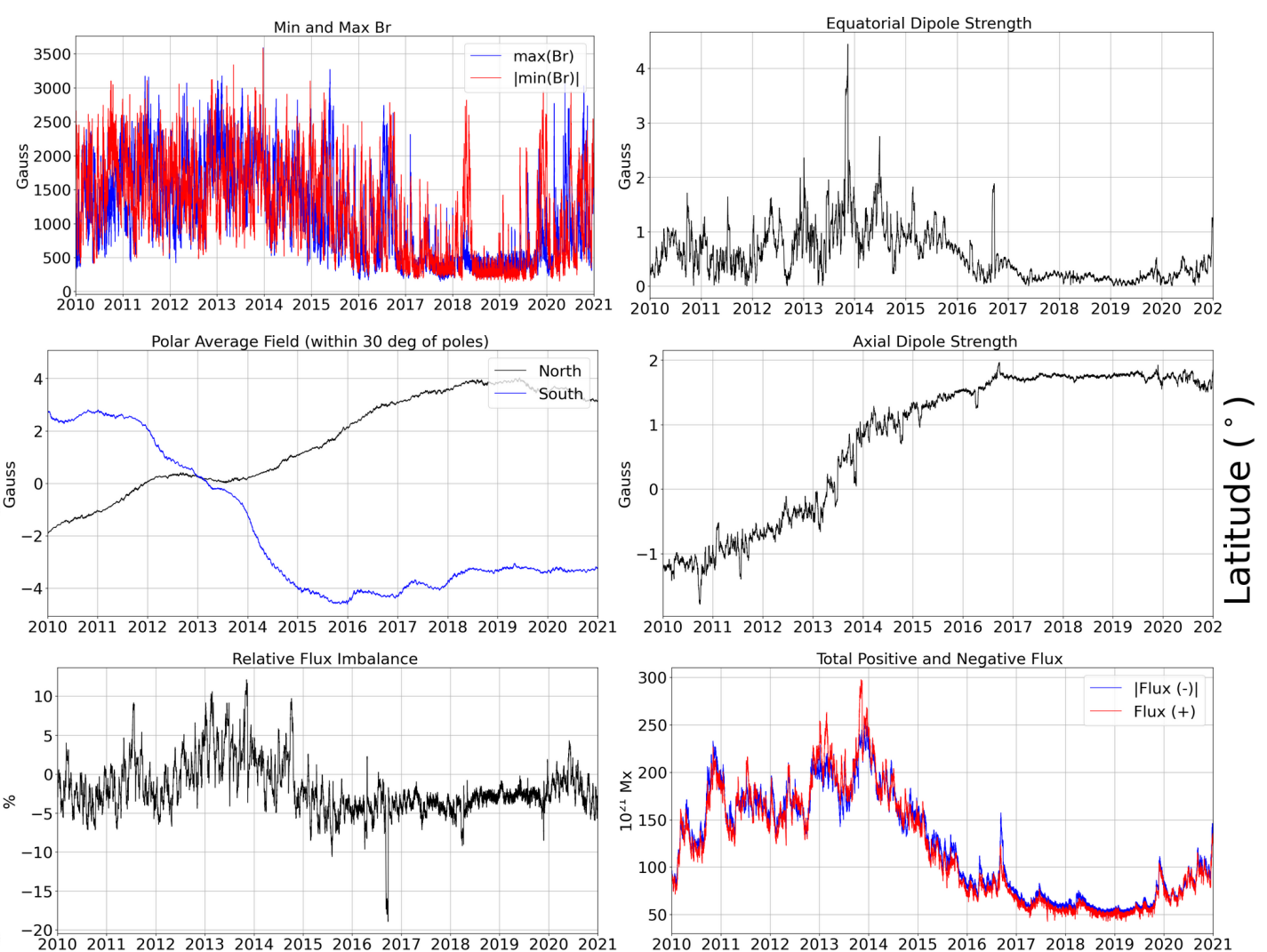


EXAMPLE PRODUCTION RUN

Initial map from AFT model, HMI data assimilation (1-hour cadence), ConFlow (1CR) and analytic flows with 500G attenuation, diffusion of 175 km²/s

Runtime on an NVIDIA RTX 2080Ti GPU: 13:00:00





Butterfly Diagram (1CR average)