New climatology modeling methodologies

Paul O'Brien and Alex Boyd

5/20/2025

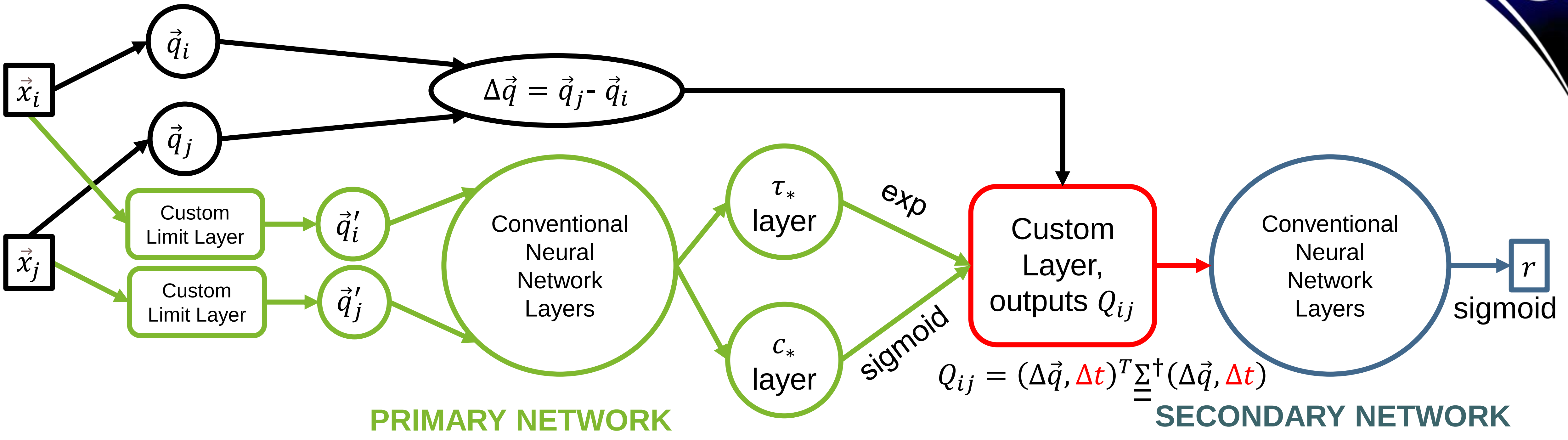
OTR 2025-00766

Analytical Covariance Matrices

Background

- Climatology models that go beyond a mean environment require spatial and spatiotemporal covariance to estimate confidence levels
- The model domains typically have 3 or 4 “spatial” dimensions, e.g., E, K, Φ
- For a 3-d model, the spatiotemporal covariance is a 7-dimensional function:
  - $r(x_1, x_2, x_3, x_1', x_2', x_3', \Delta t) = \mathrm{cov}\left(j(x_1, x_2, x_3, t), j(x_1', x_2', x_3', t + \Delta t)\right)$
  - j is flux or log flux or a Gaussian replacement of flux
  - This function has several constraints, for example, when $\Delta t = 0$ and $\vec{x} = \vec{x}'$, then $r =1$
- ONERA has a simplified solution [Brunet+2021]:
  - Obtain spatial correlations from a simulation (Salammbo) and data set (Van Allen Probes)
  - Obtain spatiotemporal correlations with a temporal decorrelation function fitted to data
  - $r(x_1, x_2, x_3, x_1', x_2', x_3', \Delta t) = \mathrm{cov}\left(j(x_1, x_2, x_3, t), j(x_1', x_2', x_3', t)\right) \exp\left(-\frac{\Delta t^2}{2\tau(x_1,x_2,x_3)\tau\left(x_1',x_2',x_3'\right)}\right)$
  - Does not handle asymmetric temporal correlations
  - Cannot interpolate or extrapolate spatially if the simulation or data set do not cover the entire domain of the climatology model
- We expand on prior works in Gaussian Processes [Paciorek+Schervish, 2004; Rasmussen+Williams, 2006] to develop a flexible non-stationary covariance function in the form of a neural network
- We train the neural network with data from a numerical simulations, electrons so far:
  - VERB [Saikin+2021] that uses a V, K, L coordinate system (already in use)
  - BAS-RBM [Glauert+2018] uses E, K, L* coordinate system (not yet in use)

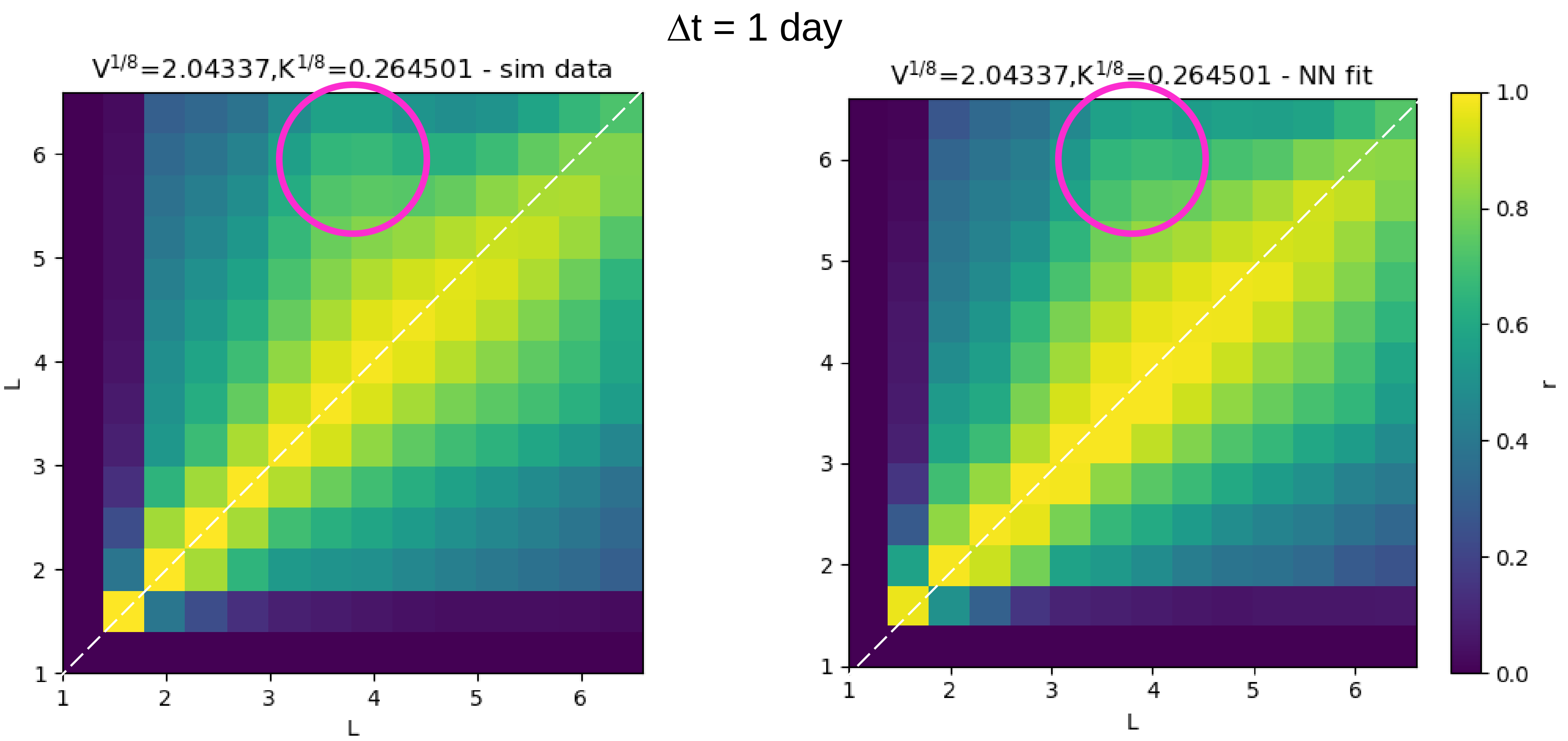
Neural network model of spatio-temporal covariance



- We extend the space to a fourth dimension (Δt), which allows for $\mathrm{r}(\vec{x}_i, \vec{x}_i; \Delta t) < 1$
- We can employ an input swap requirement to ensure symmetry $\mathrm{r}\left(\vec{x}_i, \vec{x}_j; \Delta t\right) = \mathrm{r}\left(\vec{x}_j, \vec{x}_i; -\Delta t\right)$
- We train a separate network at each Δt. In practice the network only knows that $\Delta t > 0$, and actually uses Δt=1 in all cases for numerical stability. The real value of Δt is only used to build training data and to label the network

$$\underline{\underline{\Sigma}} = \begin{bmatrix} \tau_1^2 & c_{12}\tau_1\tau_2 & c_{13}\tau_1\tau_3 & c_{1t}\tau_1\tau_t \\ c_{12}\tau_1\tau_2 & \tau_2^2 & c_{23}\tau_2\tau_3 & c_{2t}\tau_2\tau_t \\ c_{13}\tau_1\tau_3 & c_{23}\tau_2\tau_3 & \tau_3^2 & c_{3t}\tau_3\tau_t \\ c_{1t}\tau_1\tau_t & c_{2t}\tau_2\tau_t & c_{3t}\tau_3\tau_t & \tau_t^2 \end{bmatrix}$$

Example NN Spatio-Temporal Covariance Fit (1 day time lag)

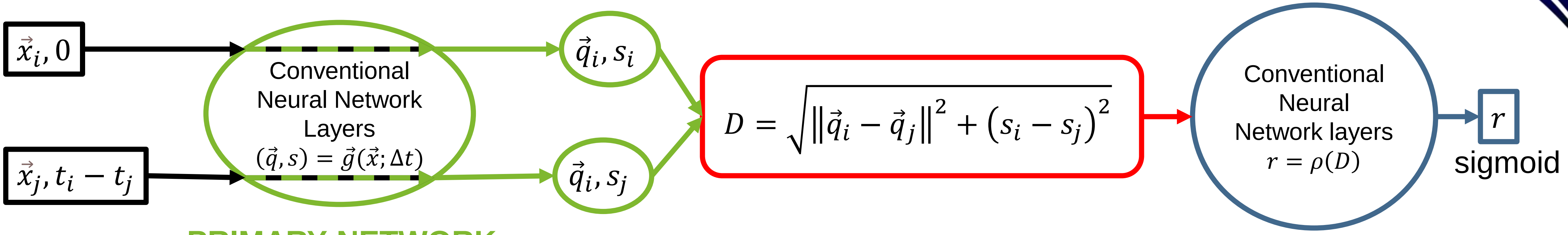


NN covers all V, K, L in sim. Only one slice shown here.
Notice slight asymmetry across diagonal, circled O

A new idea inspired by Reimann spaces

- Build a space-time coordinate transform to a system where decorrelation is isotropic (a "flat" space):
  - $(\vec{q}, s) = \vec{g}(\vec{x}; \Delta t)$
  - $r(\vec{q}_i, \vec{q}_j, t_i - t_j) = \rho\left(\sqrt{\left\|\vec{q}_i - \vec{q}_j\right\|^2 + \left(s_i - s_j\right)^2}\right)$
- Constraints:
  - $\frac{\partial g_i}{\partial x_i} > 0$ (monotonic)
  - $\frac{\partial g_i}{\partial x_j} = \frac{\partial g_j}{\partial x_i}$ (symmetric)
  - $\left|\frac{\partial \vec{g}}{\partial \vec{x}}\right| > 0$(non-singular matrix determinant)
  - $\rho(0) = 1$ (consistent)
  - $\vec{\nabla}\rho \cdot \vec{\nabla}\left(\sqrt{\left\|\vec{q}_i - \vec{q}_j\right\|^2 + \left(s_i - s_j\right)^2}\right) < 0$ (decorrelation with distance)
- Implementing constraints is fairly easy in modern NN codes (e.g., PyTorch): it's just another term in the penalty function
- Constraints can/should be applied over entire valid range of $\vec{x}, \Delta t$, not just where there's data
- This approach avoids a lot of the structural complexity of the prior approach, but it should be able to capture the same structural details – a spatial transform of this type is implicit in the 7-D covariance function

Reimann-inspired NN model of spatio-temporal covariance



PRIMARY NETWORK (Called twice)

$$\frac{\partial g_i}{\partial x_i} > 0 \qquad \frac{\partial g_i}{\partial x_j} = \frac{\partial g_j}{\partial x_i} \qquad \left|\frac{\partial \vec{g}}{\partial \vec{x}}\right| > 0$$

SECONDARY NETWORK

$$\rho(0) = 1 \qquad \frac{d\rho}{dD} < 0$$

- This neural network can be trained on the entire data set of correlations
  - Training tuples: $(\vec{x}_i, t_i, \vec{x}_j, t_j) \rightarrow r$
  - Observed correlations
  - Model correlations
  - Constraints
- Constraints should be applied at points within the domain randomized at every training step
- Weighting strategy among observed correlations, model correlations, and constraints will likely require some hand tuning

What we learned about using simulation data

- Sims often include a boundary region to separate the main physics domain from highly uncertain boundary conditions. Correlations computed in this boundary region are not valid and must be excluded from training data.
- The VERB sim appears to have certain grid points with anomalously low/no variation. Correlations involving these points are not valid and must be excluded from training data.
- The VERB sim appears to require a few years to “warm up”, especially at low L. Correlations should be computed only after this warm-up period.

New Marginal Distributions for Climatology Models

Acknowledgement: Constantinos Papadimitriou of SPARC contributed substantially to the investigation of new approaches to marginal distributions.

Introduction

- Marginal distributions describe the variability of particle flux at a specific point in the climatology model's grid
- Current climatology models take two different approaches to marginal distributions:
  - Tables of fluxes at prescribed percentiles
  - Parameters of analytical distributions (e.g., Weibull, Log-Normal which have 2 parameters)
- Tables are more precise, but it is complex to represent the uncertainty on them: $O(N^2)$ problem for table of N percentiles
- Parameters are easier to work with, but are less precise and have discontinuities if different functional forms are used for the same species (e.g., plasma electrons and radiation belt electrons)
- With Constantinos Papadimitriou at SPARC, we have developed a hybrid approach that combines tables for describing the marginal distribution with a 2-parameter approach to uncertainty

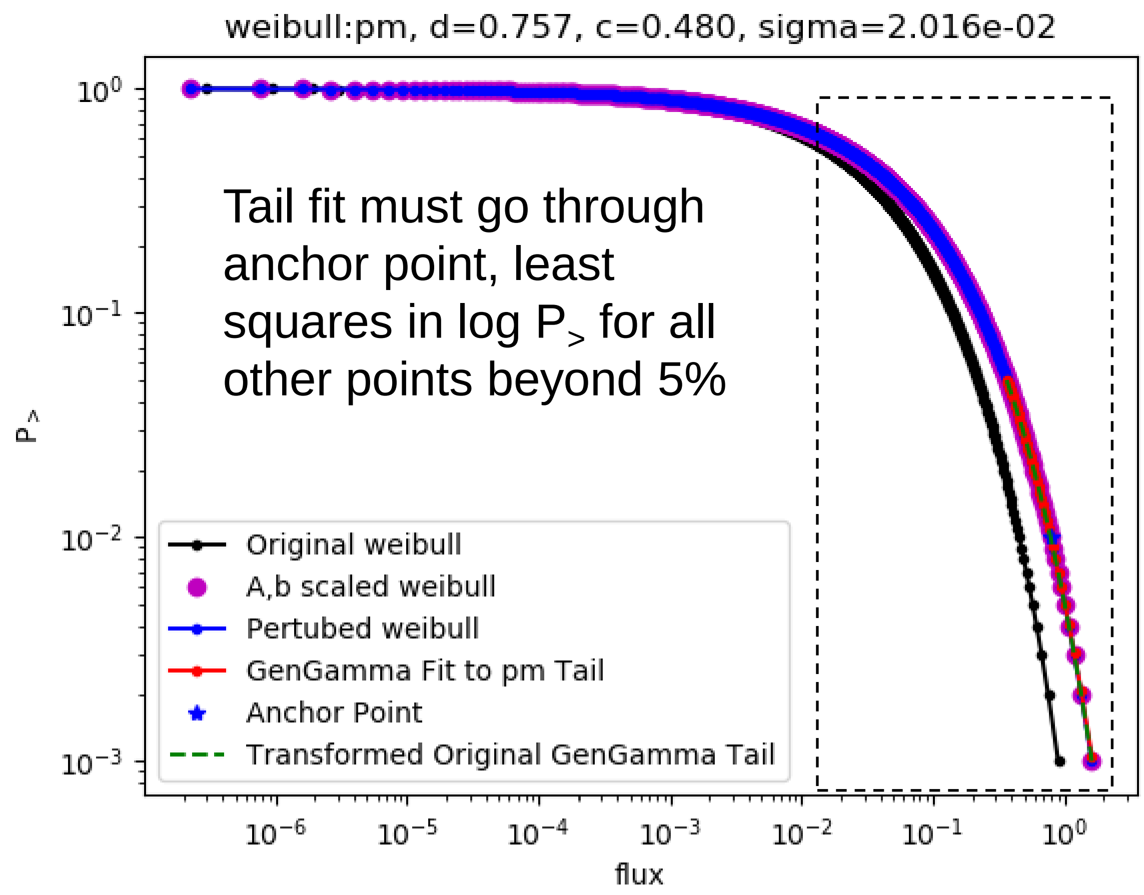
- Briefly: the marginal distribution is given as a table with a generalized gamma distribution* as its tail, while the uncertainty is tracked only for the 50^{th} and 95^{th} percentile. To perturb the entire distribution, first perturb the 50^{th} and 95^{th} percentiles, then interpolate/extrapolate those changes (in a power-law sense) to other percentiles

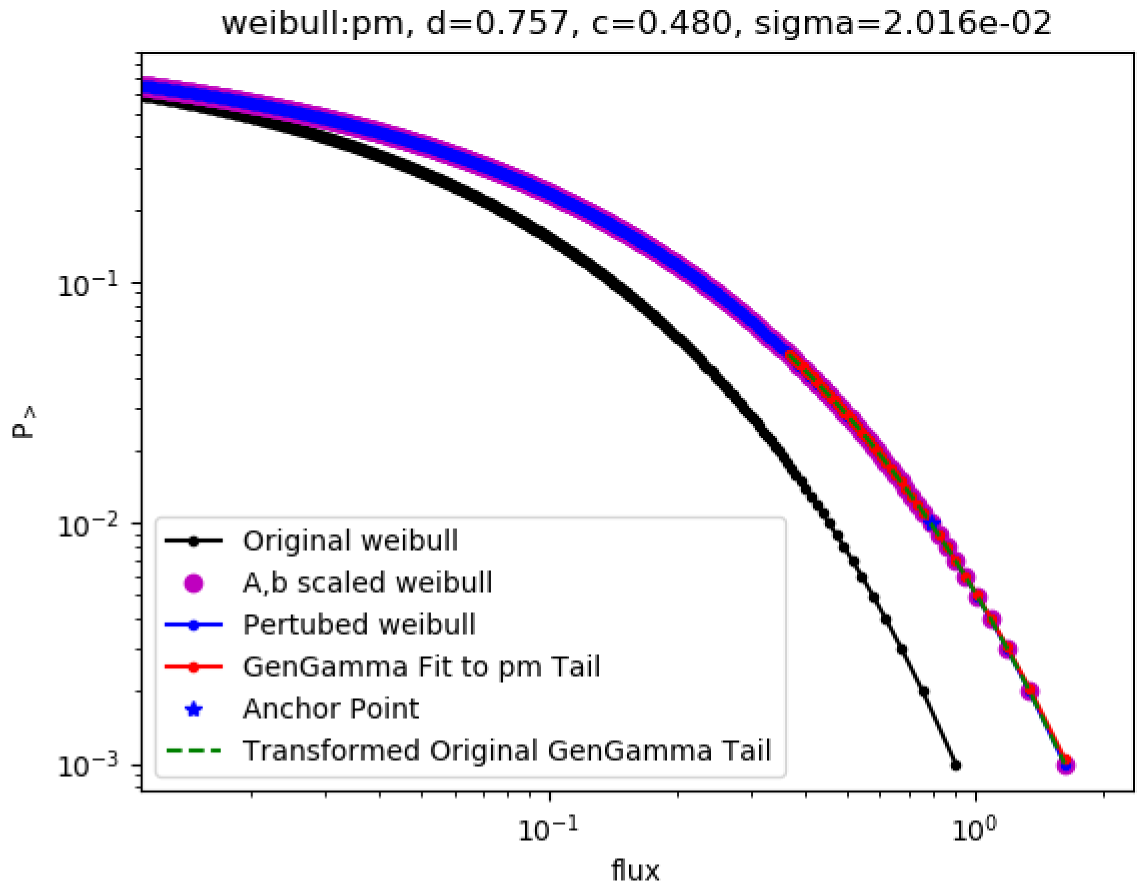
- *generalized gamma is a gamma distribution with a power-law transform applied to the variate

Math

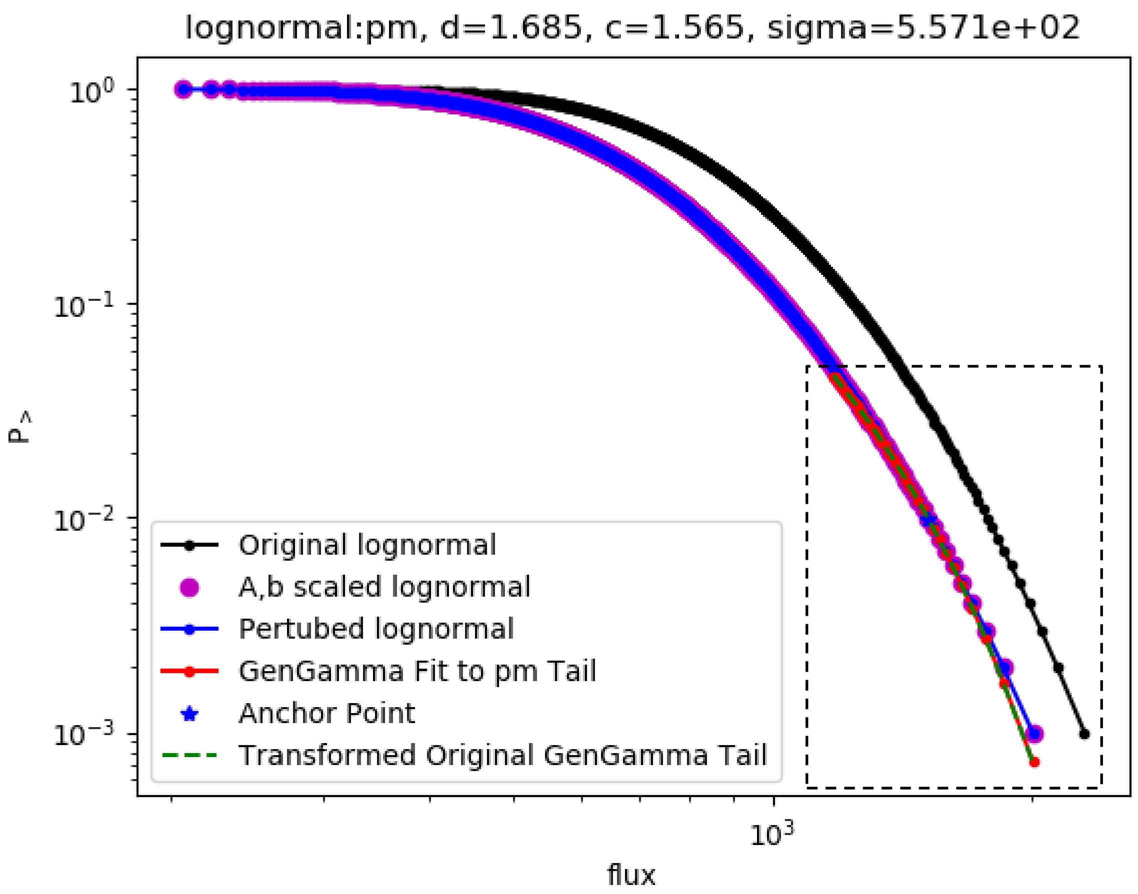
- At each grid point, we have a table of percentiles: $\vec{m} \in \{F(m_i) = p_i\}$
  - m_i is the flux at percentile p_i
  - m_{50} denotes the median, and m_{95} denotes the 95th percentile
- Error is represented on transformed variables:
  - $\theta_1 = \ln m_{50}, \theta_2 = \ln(m_{95} - m_{50})$
  - Global error matrix $\underline{\underline{S}}_\Theta$ tracks correlated errors at all grid points for θ_1 and θ_2
  - Perturb all θ using: $\vec{\Theta}' = \vec{\Theta} + \underline{\underline{S}}_\Theta \vec{\varepsilon}$, then obtain m'_{50} and m'_{95} at each grid point
- Assume a power-law perturbation function: $m'_i = A{m_i}^b$
  - $b = \ln \frac{m'_{50}}{m'_{95}} / \ln \frac{m_{50}}{m_{95}}$,$A = \frac{m'_{50}}{{m_{50}}^b}$
  - Apply this transform to all percentiles: $\vec{m}' \in \left\{F'(m'_i) = F'\left(A{m_i}^b\right) = p_i\right\}$
- Tail:
  - Original generalized gamma is: $F(x) = \gamma\left(\frac{d}{c}, \left(\frac{x}{\sigma}\right)^c\right) / \Gamma\left(\frac{d}{c}\right)$
  - Weibull when $\frac{d}{c} = 1$, log-normal as $\frac{d}{c} \to \infty$
  - Perturbed is $F'\left(x' = Ax^b\right) = \gamma\left(\frac{d\prime}{c\prime}, \left(\frac{x\prime}{\sigma\prime}\right)^{c\prime}\right) / \Gamma\left(\frac{d\prime}{c\prime}\right)$, $\sigma' = A\sigma^b, c' = c/b,\ d' = d/b,\ \frac{d'}{c'} = \frac{d}{c}$

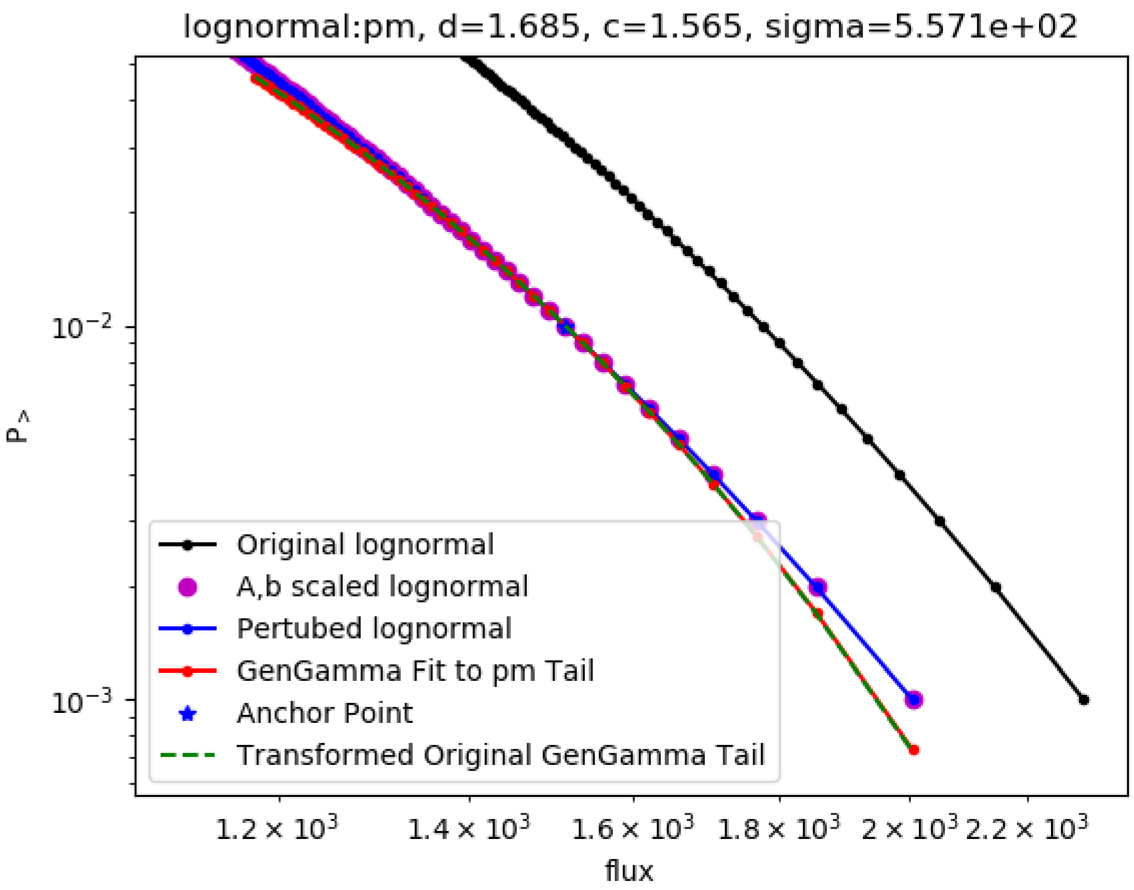
Weibull Example





Log-Normal Example





Summary

- New approach:
  - Provides flux at a fixed set of percentiles
  - Includes a generalized gamma distribution for the right tail
  - Allows power-law perturbation of entire distribution and tail based on perturbing 50^{th} and 95^{th} percentiles
- Improvements
  - Achieves greater accuracy via tabular percentiles
  - Maintains simple uncertainty quantification via errors on 50^{th} and 95^{th} percentile
  - Seamlessly incorporates Weibull and Log-Normal tails in generalized gamma
- To be determined:
  - How to decide which percentiles to measure directly when samples are small, and which ones to extrapolate with a fit
  - How best to fit the tails, especially in bins with few samples
  - How to spatially interpolate/extrapolate tabular values and tail fits
  - Should we be using SPINNs instead?

New Directions

Should we be Using Statistics and Physics Informed Neural Networks?

- Our models now look like this:
  - A spatio-temporal joint probability distribution over time
  - Represents likelihood of flux j at two points $(\vec{x}_i, t_i)$ and $(\vec{x}_k, t_k)$
  - $p\big(j(\vec{x}_i, t_i), j(\vec{x}_k, t_k)\big) = p(j_i; \vec{x}_i)p(j_k; \vec{x}_k)c(j_i, j_k; \vec{x}_i, \vec{x}_k, t_i - t_k)$
  - $p(j; \vec{x}) = \left.\frac{dF}{dj}\right|_{\vec{x}}$ - marginal distribution of flux at $\vec{x}$
  - $c(j_i, j_k; \vec{x}_i, \vec{x}_k, \Delta t)$ – copula describing covariance structure as a function of time lag Δt (C is cumulative copula)
- New developments in neural networks allow us to represent unknown functions F, C as neural networks
- Training could be done on heterogeneous data:
  - Simulation data
  - Unidirectional differential fluxes
  - Integral fluxes
  - Raw sensor counts (accounting for sensor response that integrates over grid)
- Training could include statistical constraints:
  - Marginals F have 0,1 bounds and positive derivative everywhere
  - Integral copulas C have boundary conditions and all positive derivatives everywhere
- Training could include physical constraints:
  - Marginal gradients approximate steady state diffusion equation
  - Marginals obey boundary conditions (loss cone, zero gradient at magnetic equator)

SPINN challenges

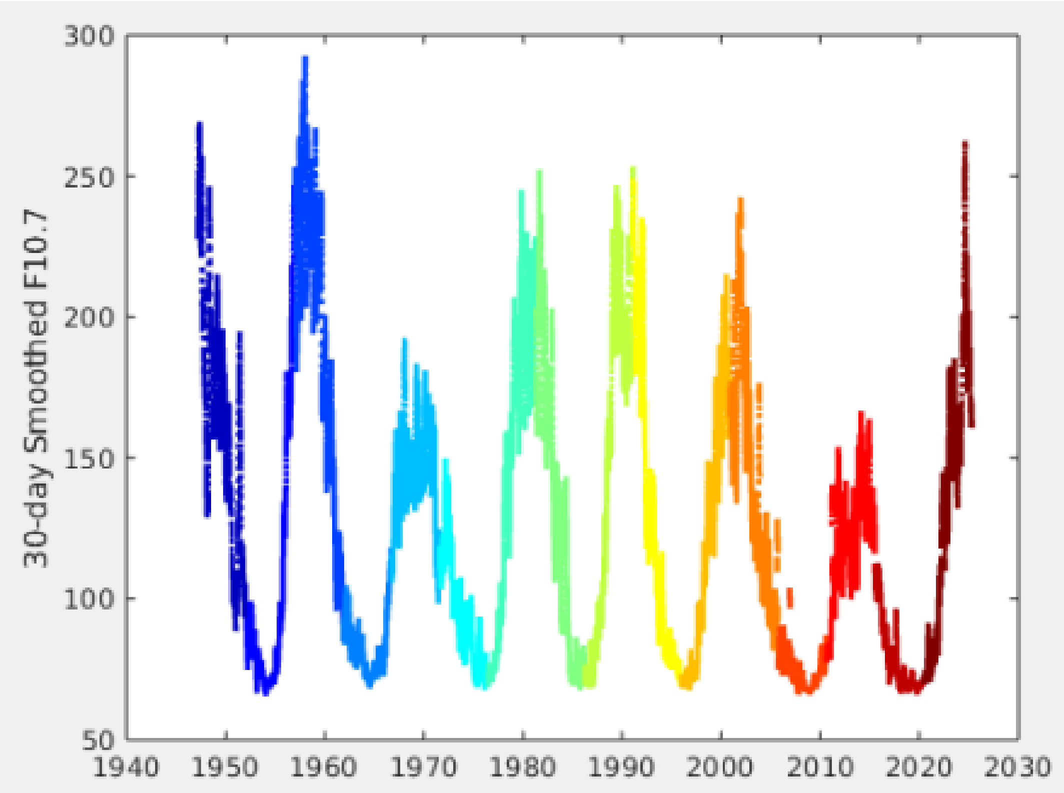
- How do we capture model uncertainty in a SPINN framework? I.e., how do we try to make today's model bound tomorrow's data?
- How do we train a SPINN?
  - Lots of work to figure out how to build penalty functions for legacy data sets
  - Enormous data sets (all flight data ever)
  - Probably some hand tuning to balance data constraints, statistical constraints, physical constraints
- How do we use a trained SPINN?
  - Are we still doing Monte-Carlo sampling to generate scenarios?
  - Will it run quickly enough?
  - Do we need, instead, to use something like cumulants to avoid Monte-Carlo?

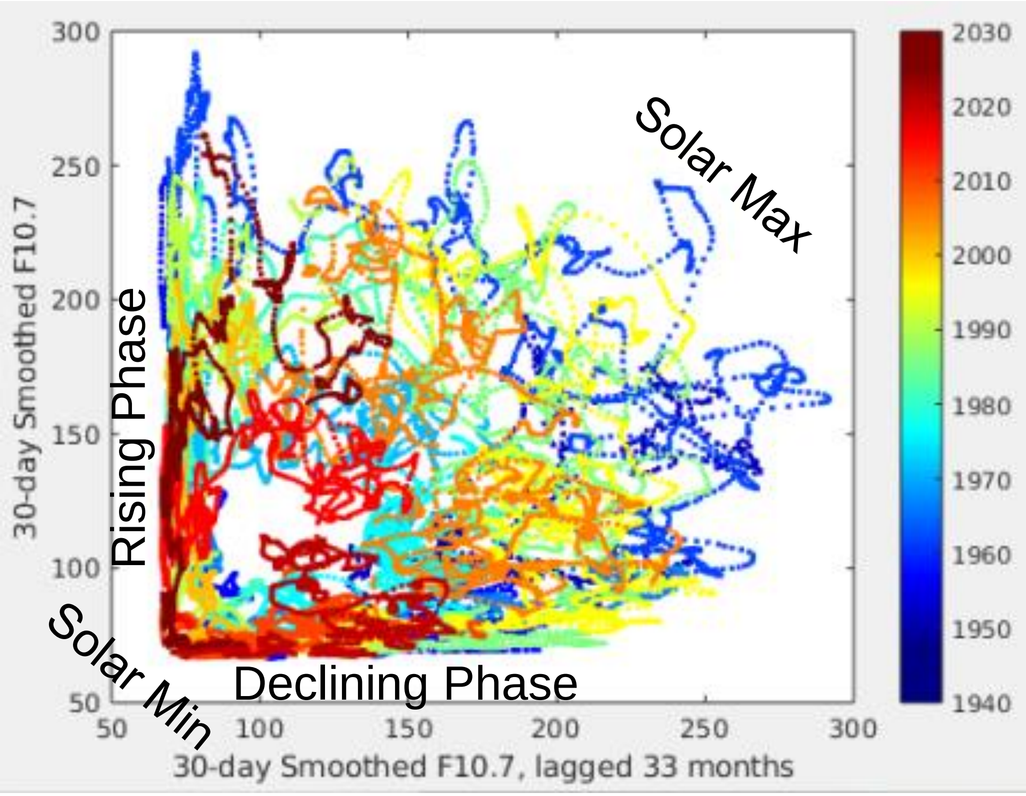
Can we get away from magnetic coordinates?

- One of the computationally expensive and complex pieces of evaluating a radiation environment model is converting from the model's magnetic coordinates to the spacecraft location
- Using principal component compression, we might be able to store everything on a time, altitude, latitude, longitude, energy, pitch-angle grid in a reasonable amount of data
  - ~300 altitude steps from 100 km altitude to R=12 RE. 50 km steps in LEO, ~0.7% dR/R
  - ~100 L/latitude steps, < 3 degrees latitude step, < 0.25 L step
  - 15 longitude steps
  - ~50 energy steps from 1 keV to 1 GeV, 30% dE/E
  - ~10 pitch angle steps (5, 10, 20, … 90 degrees)
  - ~4E8 grid points, stored a single precision (1.6 Gb for unidirectional flux, 0.16 Gb for omni)
- Variation:
  - ~100 principal components (165 Gb for unidirectional flux, 17 Gb for omni)
  - ~300,000 time steps (31 years at 1 hour resolution)
  - 2.7E10 principal component amplitude points
  - ~100 Gb
- Developer interpolates their model onto such a grid before software delivery
- Greatly simplifies delivered software:
  - Computing full flux maps from PCs and amplitudes
  - Interpolating and integrating over flux maps
  - No field model, no field line tracing, no drift shell coordinates
- Does not reduce time for Monte Carlo uncertainty and dynamics

How to parameterize solar cycle variation?

- Solar cycle drives systematic variation in low-altitude proton fluxes (atmospheric density)
- Solar cycle drives systematic variation in the statistics of high-altitude proton and electron fluxes (storms)
- Solar cycles are only quasi-periodic, which poses a challenge for ensuring continuity at solar cycle boundary
- Proposed solution: represent solar cycle phase as two parameters:
  - Current monthly average sunspot number
  - Monthly average sunspot number lagged by ¼ solar cycle (33 months)





Summary

- Neural networks can allow us to define very flexible covariance models
- Covariance NNs can be trained on real and simulation data

- Generalized gamma distribution combined with tabulated percentiles can replace the log-Normal and Weibull formulations we have used previously

- SPINNs might offer a framework to build a grid-free model that can fit heterogeneous data without having to convert everything to unidirectional differential flux

- Principal component compression might allow us to build real-space dynamic maps rather than using magnetic coordinates

- It might be good to parameterize solar cycle dependence in terms of F10.7 and lagged F10.7

References

- Brunet, A., et al., (2021) OMEP-EOR: A MeV proton flux specification model for electric orbit raising missions, J. Space Weather and Space Climate, 11(55), https://doi.org/10.1051/swsc/2021038
- Glauert, S. A., et al., (2018). A 30-year simulation of the outer electron radiation belt. Space Weather, 16, 1498–1522.
- Paciorek, C. and Schervish, M. J. (2004). Nonstationary Covariance Functions for Gaussian Process Regression. In Thrun, S., Saul, L., and Scholkopf, B., editors, Advances in Neural Information Processing Systems 16. MIT Press. http://machinelearning.wustl.edu/mlpapers/paper_files/NIPS2003_AA35.pdf
- Rasmussen, C. E., and C. K. I. Williams, Gaussian Processes for Machine Learning The MIT Press, 2006. ISBN 0-262-18253-X, http://www.gaussianprocess.org/gpml/
- Saikin, A. A., et al. (2021). Reconstruction of the radiation belts for solar cycles 17–24 (1933–2017). Space Weather, 19, e2020SW002524.