A Framework for Applying Neural Networks to Eddy Covariance Data

Dr. June Skeeter

june.skeeter@ubc.ca

University British Columbia

Eddy Covariance

Ecosystem-scale fluxes energy, water, and trace gases.

- Voluminous data sets
  - Noisy & gap-prone
- Ideally suited for machine learning!

Burns Bog EC Station
Delta, BC

Neural Networks

Universal approximators: they can map any continuous function to an arbitrary degree accuracy.

- With sufficient hidden nodes, they will fit **any** pattern in a dataset
  - Care must be taken to ensure the patterns are real
    - Have often been treated as “black boxes”
- Well suited for non-linear, multi-variate response functions
  - Capable *interpolation* and **extrapolation**

Commonly Cited Limitations

Issue	Solutions
Over-fitting	* Model ensembles * 3-way cross validation * pruning
Black boxes models	* Plot partial derivatives * feature importance
Computationally expensive	* Tensorflow * GPU processing

Objective

Provide a framework for applying NN models can be applied to EC data for both descriptive analysis and inferential modelling.

- The github repository linked to this presentation has functional examples that can be used to apply NN models.
  - Runs in Python and Tensorflow: *GPU support not required*

Example Data

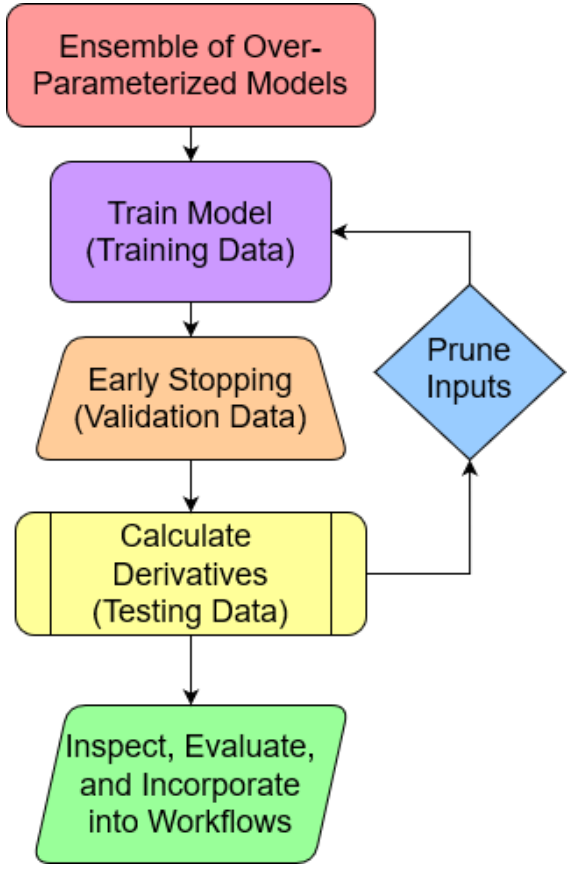
BB1 EC station

- Beakrush-Sphagnum ecosystem undergoing active restoration
- 8+ years of CO_2 and CH_4 flux observations

Training Procedures

An iterative process:

- Three way cross-validation
  - Train & validate - random split by model
  - Test - consistent between models
- Ensemble-size
  - Larger ensemble > more robust model
  - N = 10 likely suitable for most applications

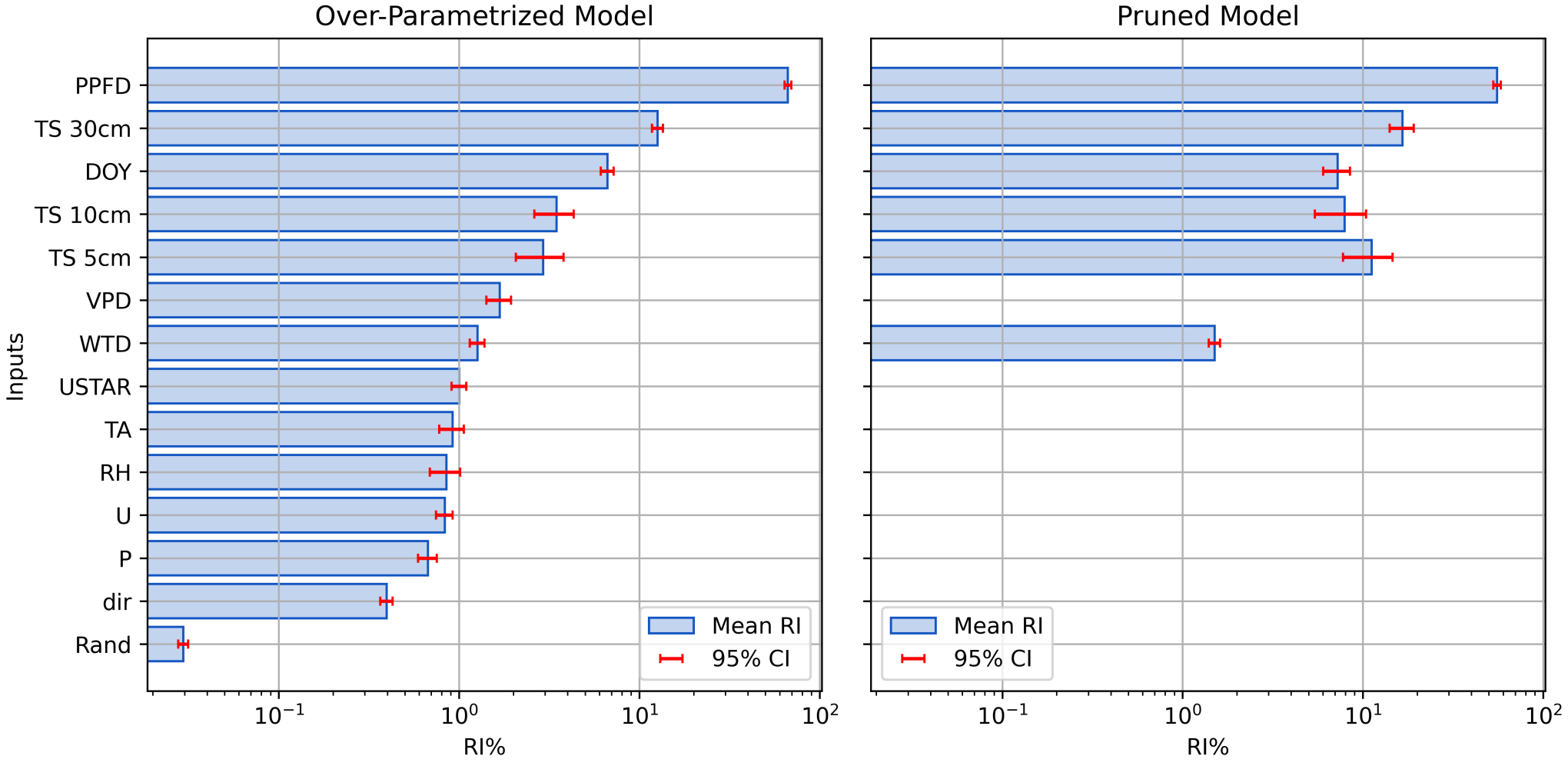


How to Prune a Model

Calculate partial first derivative of the output with respect to each input over test data domain.

- **Relative Influence (RI)**: Normalized Sum of squared derivatives (SSD)
- Iteratively prune inputs with RI below a reference threshold
  - Random input assess base-level performance
- Train final model without random scalar

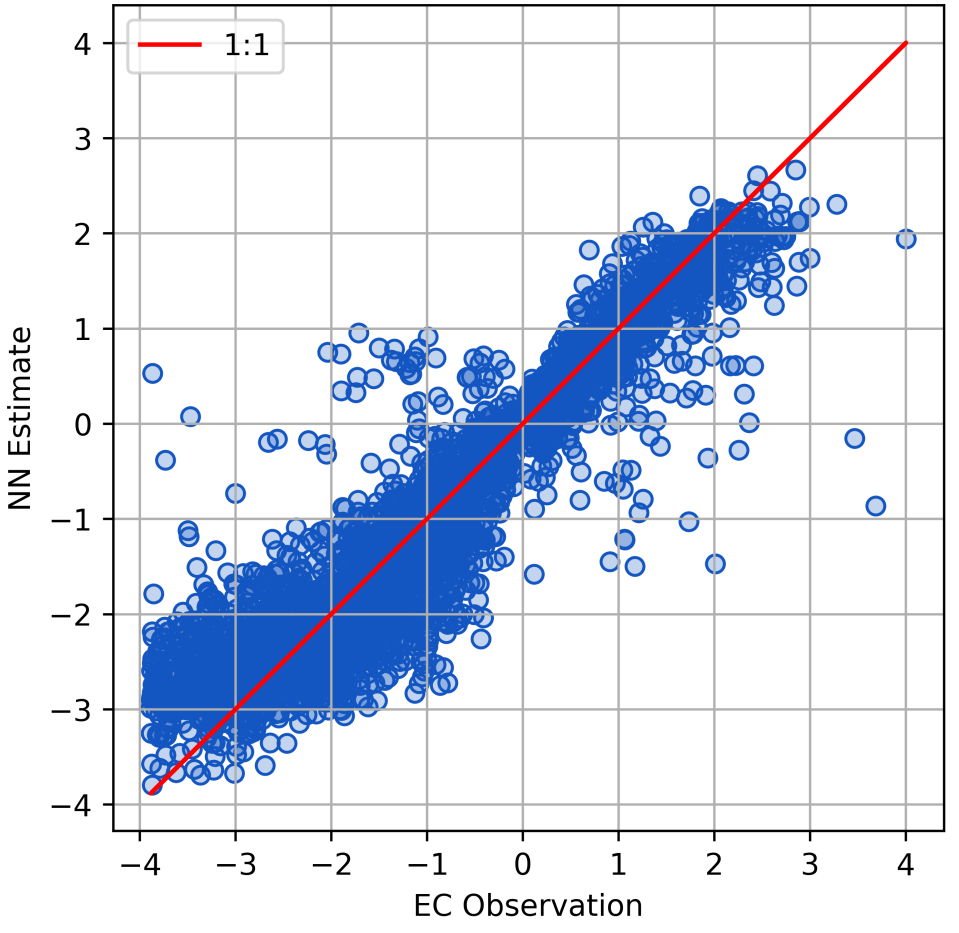
Pruning a FCO_2 Model



Model Inspection

Plot the model outputs using the test data.

Metric	Score
RMSE	0.59 $\mu mol\, m^{-2} s^{-1}$
r^2	0.86

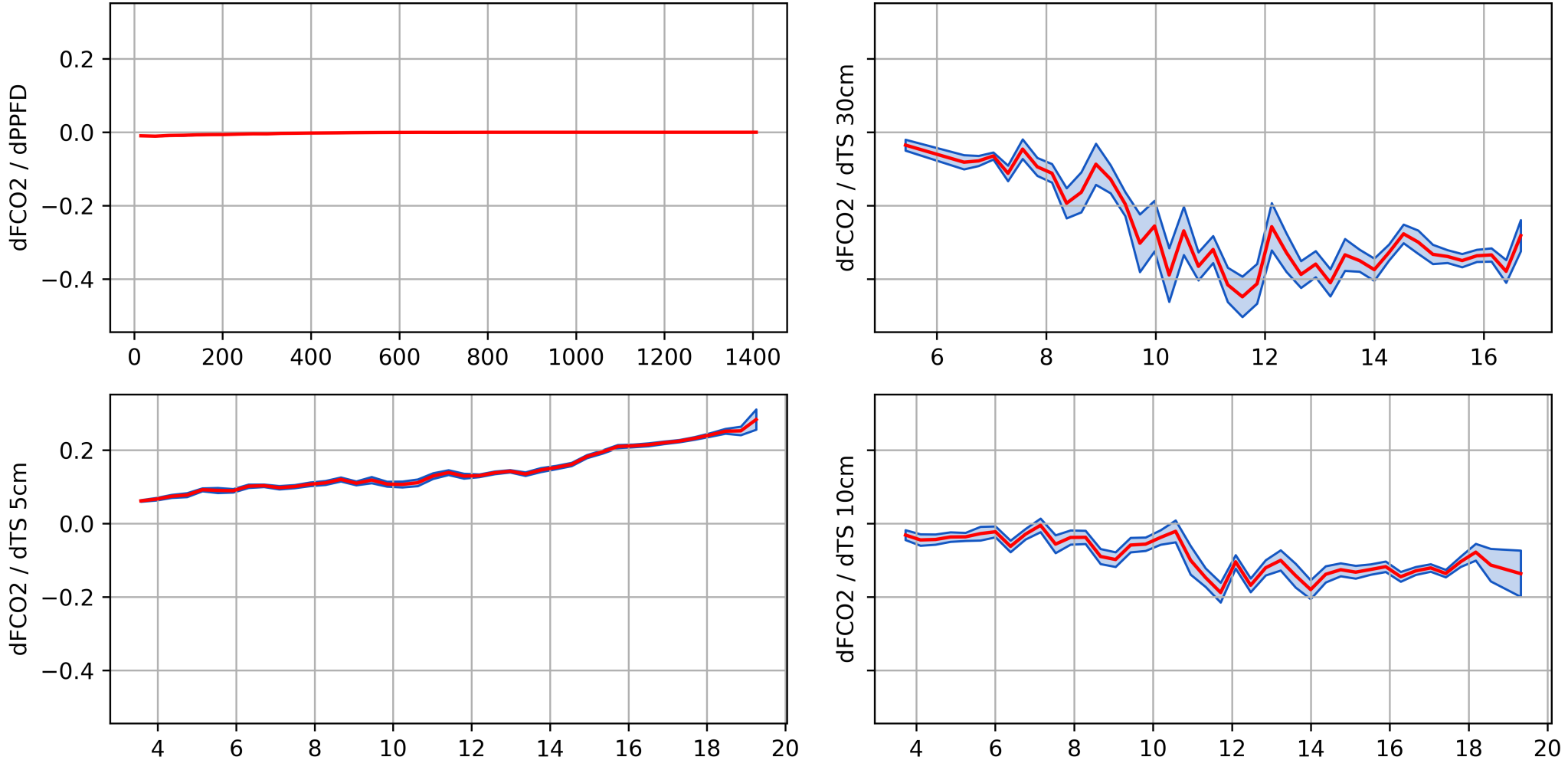


Plotting Derivatives

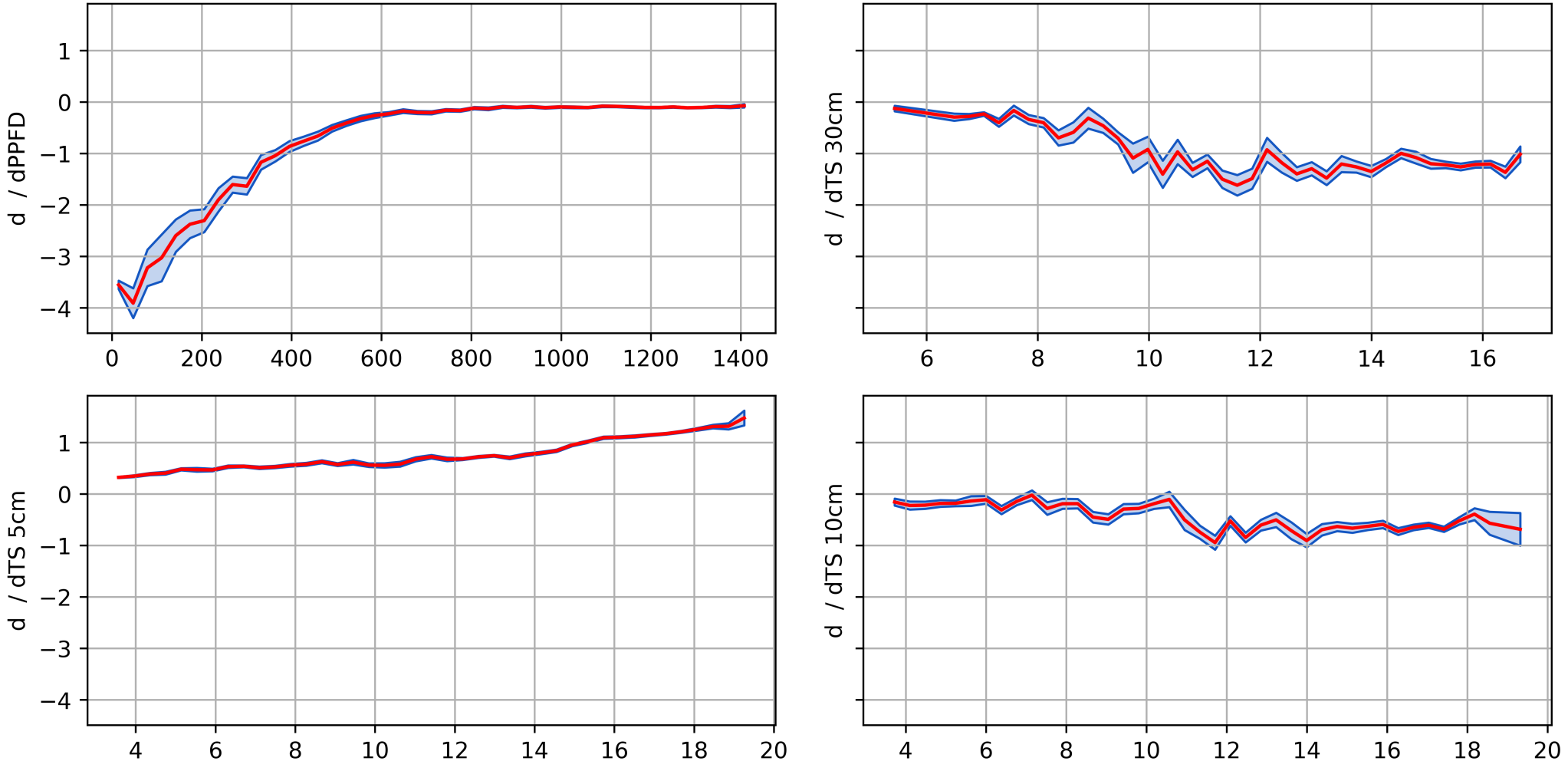
Help ensure mapped relationships are physically plausible

- An **essential step** and **key advantage** NN models
- Raw derivatives show true feature responses
  - 95% confidence intervals indicate model confidence in relationships
- Normalized derivatives scaled by input variance
  - View relative on common scale

Partial Derivatives of FCO_2



Normalized Partial Derivatives



Next Steps

- Custom NN architecture: Separating input layers may allow us partition fluxes.
  - FCO_2 into GPP and ER
  - FCH_4 into methanogenesis, methanotrophy, transport
- Flux footprint analysis: Models can help account for spatial heterogeneity within a a footprint
  - NN could also be trained to calculate/classify footprints!
- u* filtering: Partial derivatives of u* could give thresholds for filtering

Thank You

Questions?

Why not use a Random Forest?

RF models are great! … for classifying discrete objects

- But, it's my view that applying them to continuous data is misguided
- They are poorly suited for interpolation and incapable of extrapolation