Deriving Inherent Optical Properties Using Bayesian Neural Networks

Susanne E. Craig, Erdem M. Karaköylü

September 14, 2018

Contents

1 Introduction

2 Methods

2.1 Data Collection

2.2 Data Preparation for Machine Learning

- A note on reproducibility: All code available for download on github/data available on
- Feature engineering and data transformation:
  - transformation of lat/lon
  - transformation of time
  - Rayleigh/Fresnel-corrected TOA radiance
- Train/Test Split: While bayesian models are robust to overfitting, the data was nevertheless split into 90%/10% training/testing sets, so as to enable possible future comparisons with non-bayesian machine learning implementations.
- Standardization:
- Pairwise Relationship: Figure 1 below summarizes the dataset at hand.

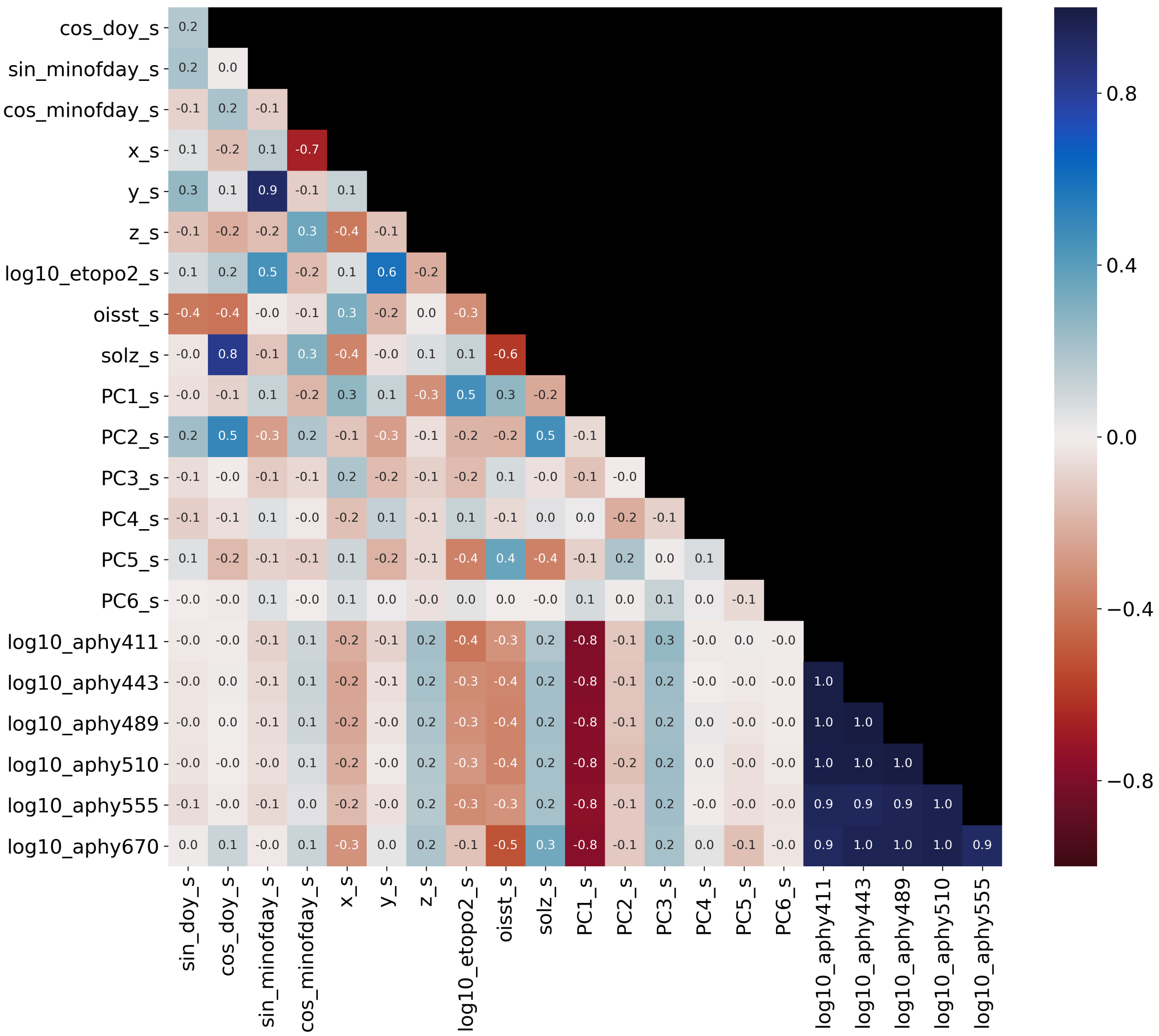
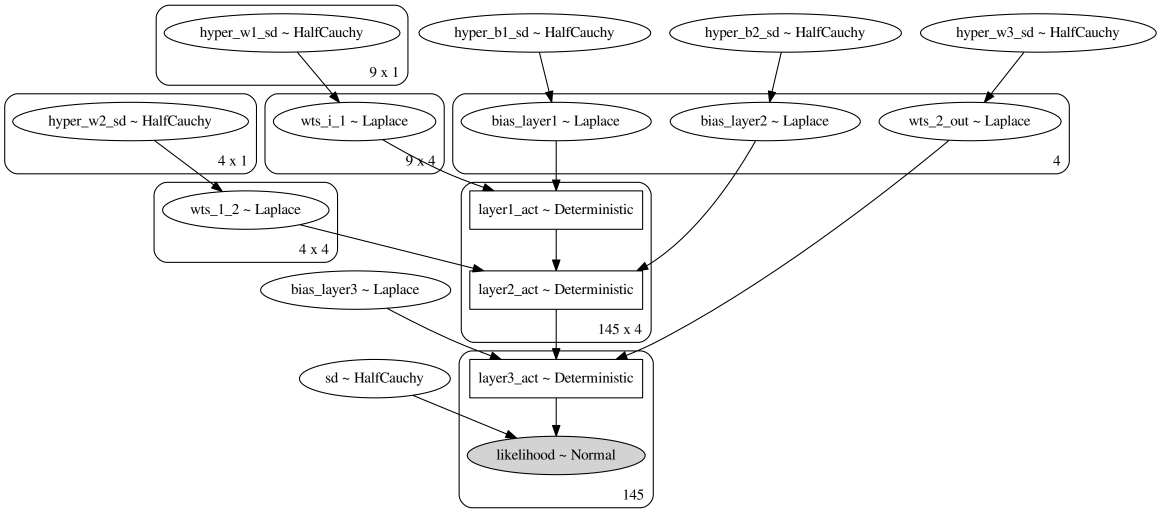


Figure 1: Pairwise correlation plot of features and targets. Features are standardized; targets are log-transformed.



2.3 Bayesian Neural Network Implementation

2.3.1 Neural Networks and the Bayesian Paradigm

- Neural Networks as function approximators
- Weights as distributions rather than scalar parameters

2.3.2 Model Architecture and the Automatic Relevance Determination (ARD) Framework

- A bayesian hierarchical model is one in which the parameterization of coefficient distributions are dependent on an overarching distribution [include kruschke figure?]
- A bayesian neural network with ARD is one where the variance of the distributions of the weights comming out of each input unit is controlled by a hierarchical distribution. This allows uncovering how relevant each feature is in predicting the target.
- Prior specification

2.4 Data and Software Access, and Reproducibility

- Code written in Python
- Data preparation using pandas, numpy and scikit-learn
- probabilistic programming in PyMC3
- All code was written in python and is available as jupyter (formerly ipython) notebooks on github (→ give reference).
- Data available on project repository, Open Science Framework (→ give reference), and includes
  - full and train/test-split dataset
  - Python standard scaler and PCA transformer objects parameterized for the data used here
  - PyMC3 models fitted to the data, as serialized Python objects stored in binary files.

3 Results

3.1 Model Comparison

- WAIC
- LOOCV

3.2 Test Set Validation And Posterior Predictive Checks

- Test set observed/predicted comparison
- Fig w/ r2 and mae
- Fig w/ 95% and 50% HPD]

3.3 Posterior Distribution of Weights

- Input unit weights posterior distribution in view of ARD
- Hidden unit weights posterior distribution
- Bias weights posterior distribution

4 Discussion

5 Conclusion

6 References

- McElreath Book
- Kruschke Puppy Book
- BDA 3
- PyMC3 paper
- Radford Neal's book on BNN
- SVGD paper
- McElreath Book chapter on WAIC
- LOOCV paper
- McKay on ARD
- Neal on ARD