UNIVERSITY OF CALIFORNIA
Santa Barbara

UNDERSTANDING THE INFLUENCE OF PARAMETER VALUE UNCERTAINTY ON CLIMATE MODEL OUTPUT

A Capstone Project submitted in partial satisfaction of the requirements for the degree of Master of Environmental Data Science for the Bren School of Environmental Science & Management

by

HEATHER CHILDERS
SOFIA INGERSOLL
SUJAN BHATTARAI

Committee in charge:
DR. DANIEL KENNEDY
DR. CARMEN GALAZ GARCÍA

JUNE 2024

UNDERSTANDING THE INFLUENCE OF PARAMETER VALUE UNCERTAINTY ON CLIMATE MODEL OUTPUT

HEATHER CHILDERS

SOFIA INGERSOLL

SUJAN BHATTARAI

CARMEN GALAZ GARCÍA

DANIEL KENNEDY

JUNE 2024

Table of Contents

Abstract/Project Summary

Climate models are computer simulations that attempt to replicate the complex interactions between Earth's systems. Improving the accuracy of climate models relies on evaluating uncertainty and minimizing error. The Climate and Global Dynamics Lab at the National Center for Atmospheric Research (NCAR) has recently carried out a Parameter Perturbation Experiment (PPE) to understand how the uncertainty of parameter values affected the output of their model, the Community Land Model (CLM); which simulates terrestrial processes. While the necessary data for the PPE has been collected, the data is stored in a collection of files that are difficult to interpret in their current form. The current website hosts visualizations for a portion of the PPE data, but contains no visualizations for data that more closely simulates Earth system interactions. These issues can be mitigated by developing an emulator with the internal complexity to isolate a one-to-one relationship between a parameter and climate variable, then display the predicted relationship. A publicly available emulator with these capabilities will allow scientists to easily interpret complex climate model outputs and offer insights on parameter-variable relationships that are not being predicted accurately by the model; which can lead to increased accuracy and precision of climate models.

Acknowledgements

Bren School of Environmental Science & Management: Satie Airamé, Assistant Dean; Emily Case, Capstone Project Coordinator; Carmen Galaz García, Assistant Teaching Professor.

Columbia University: Linnia Hawkins, Associate Research Scientist;

National Center for Atmospheric Research: Nick Cote, Software Engineer; Daniel Kennedy, Project Scientist

Executive Summary

Climate change is a real and threatening problem facing today's society. Advancements in climate modeling have become one of our best tools to support research, policy, and mitigation strategies to address climate change. The National Center for Atmospheric Research (NCAR) has allocated substantial resources into developing a large-scale climate model, the Community Earth Systems Model (CESM), which consists of land, oceanic, and atmospheric climate sub-models created by various labs at NCAR.

This project has worked primarily with the Climate and Global Dynamics Lab (CGDL) at NCAR, and the data they have generated as part of the Community Land Model Parameter Perturbation Experiment, which focuses primarily on terrestrial climate change predictors. The PPE involved varying over 200 land-parameters one at a time across 2500 simulations, and varying 32 land-parameters via advanced sampling techniques across 500 simulations to test for parameter interactions.

While all the necessary data for the PPE has been collected, the data is being stored in a collection of files that are difficult to interpret in their current form. There is an existing website that has pre-processed data visualizations for the one at a time data, but there are no visualizations for the data that factors in parameter interactions. There is also minimal documentation for the parameter/variable metadata, which makes interpreting the current visualizations nearly impossible for scientists outside of the Climate and Global Dynamics Lab. Furthermore, scientists utilizing this data are limited to the 500 parameter sets because running the simulations for new parameter values is time consuming and computationally inefficient. To address these issues, the project deliverables include:

- Develop an emulator, using machine learning techniques, that has the internal complexity to parse out a one-to-one relationship between a parameter and climate variable output
- Creating an interactive dashboard that allows users to select a parameter and variable of interest and displays visualizations for the predicted values and relative parameter importance
- Improving the metadata documentation by constructing a splash page that includes experimental setup, full names of variables/parameters and the associated units.

The initial phase of creating the interactive emulator is to create a standardized workflow of functions to output a formatted dataset. This dataset was used as the input data for the machine learning model that will become our emulator. The machine learning model implemented a form of regression that can identify the individual relationship between a variable and parameter of

interest and quantify the uncertainty around the predicted relationship. After developing the key functionalities of the emulator, the results were then converted into figures that display the predicted relationship between a user-selected parameter and variable, a cross-validation plot showing the accuracy of the emulators predictions compared to the actual climate variable outputs, and a parameter influence plot to show the parameters with the highest influence on the selected climate variable.

These figures were then embedded into an interactive python dashboard using the Panel package. The structure of the dashboard will allow users to select a parameter and variable of interest, then display the visualizations with an accessible link to an HTML page that will include the metadata documentation.The dashboard will also be equipped with continuous integration, so that NCAR staff can build upon this emulator to add in extended functionality. The associated GitHub repository for this project will also act as a template for other departments at NCAR to develop similar tools for visualizing parameter sensitivity experiments.

By providing a publicly accessible emulator equipped with these capabilities, scientists gain effortless access to interpreting intricate climate model outputs. This, in turn, fosters an environment where subject matter experts can contribute insights into parameter-variable relationships currently overlooked by the model, thereby enhancing the accuracy and precision of climate forecasts.

Problem Statement

Climate models are one of the best tools we have for predicting the most harmful effects of climate change, and it's essential that climate scientists work to maintain the accuracy of the models as they become more complex. The Parameter Perturbation Experiment aimed to do just that by using two parameter perturbation methods to check for model inaccuracies. The first method involved changing one parameter at a time across a range of values, with no parameter interactions. While this data can be useful, it is not the most accurate representation of Earth's interconnected systems. The second sampling method implemented Latin Hypercube (LHC) sampling to develop a range of realistic parameter sets to test against real-world observations. Since the working group has yet to release the raw data from their parameter perturbation experiment, the CLM working group has created a preliminary website where users can quickly reference the data. However, the point-and-click navigation is time consuming, irreproducible, and is only available for the one-at-a-time dataset. Additionally, the few visualizations that are available have little-to-no aesthetic elements that allow users who aren't familiar with the acronyms used by the CLM working group to interpret the visualizations. These constraints make it very difficult to pull meaningful insights from the impactful work of the PPE. Without access to the raw dataset, there is no way to truly compare the climate variable outputs to the observations collected in the real world, which is the best way to test for model accuracy.

Specific Objectives

The primary objectives for this project are:

- Development of a **user-friendly, interactive dashboard** that will allow for users to view the results of the parameter perturbation experiment easily and efficiently. The interactive dashboard will provide an interface where new or experienced users can select a parameter, variable, and time range of interest to quickly visualize the predicted relationship. The dashboard will contain a link to a splash page with documentation describing the parameters to allow for a broader scope of individuals to explore the data.
- Creation and validation of a **machine learning model** that can produce a reduced, yet representative, dataset that will be used for data visualizations displaying the predicted relationship between user-defined parameters and variables of interest. This feature will allow scientists to visualize the results from multiple simulations simultaneously, increasing the efficiency of model insight generation and exposing flaws in the model.
- Improve the **metadata documentation** by consolidating existing documentation onto an easily accessible splash page and producing a clean GitHub repository that allows users to access the results of the Parameter Perturbation Experiment, as well as our approach for developing a data-driven emulator as a form of scientific communication.

Summary of Solution Design

8.1 Design Choices

8.1.a Emulator Components

Defining the emulator components

As outlined below in *Figure 1*, the two components that undergo data preprocessing functions are LHC perturbed parameter simulations and accepted perturbed parameter values. The pre-processing of the LHC data includes dimensional reduction of the gridcell and time dimensions, yielding each simulation's global annual mean. These values are used to train the emulator to predict the model output for the climate variable. Both datasets were normalized between 0 and 1. This enables the emulator to further enhance its performance. The normalization step guarantees data uniformity within a Gaussian assumption. This pre-processing step improves the accuracy and interpretability of subsequent analyses and predictions, fostering a more robust understanding of the relationships between perturbed parameter values and climate variable predictions.

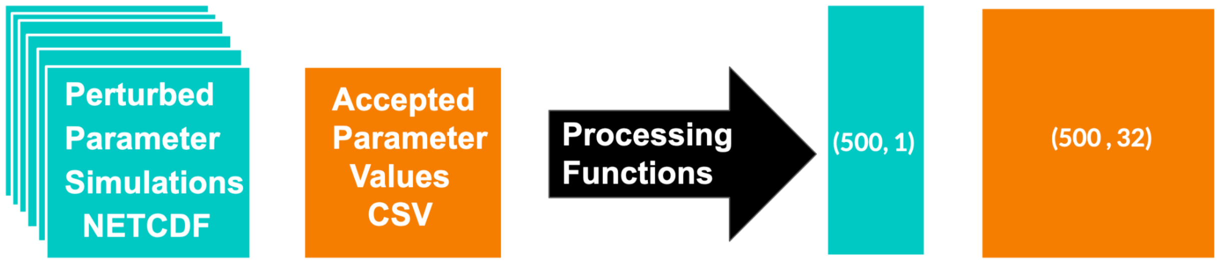

Perturbed Parameter Simulations NETCDF
Accepted Parameter Values CSV
Processing Functions
(500, 1)
(500 , 32)

Figure 1: Diagram of CLM-5 PPE data processing workflow: cluster of PPE5 simulations associated with user selected climate variable (**blue**), cluster of accepted parameter values (**orange**).

Perturbed Parameter Simulations (500, 1)

In this workflow, the Latin HyperCube (LHC) data generated by the PPE is subset as a cluster of 500 files containing sets of simulations using time ranges varying in increments of 5 years. Each simulation contains 32 parameter features that were perturbed according to reasonable bounds. All parameters were assigned independent, quasi-random values for each simulation. The timeline selected for the emulator was 1995-2015. This 20 year time range was chosen to provide more informative insights into parameter value influence on climate variables. The development and application of satellite data did not begin until the 1980s, so there is a greater degree of uncertainty associated with climate model predictions (Yang, Jun, et. al, 2013). Therefore, a decision was made to use data from 1995 and onward.

Accepted Perturbed Parameter Values (500, 32)

A cluster of accepted parameter values for 32 parameters that influence the environment.

8.1.b Gaussian Process Regression Machine Learning (GPR ML) Emulator

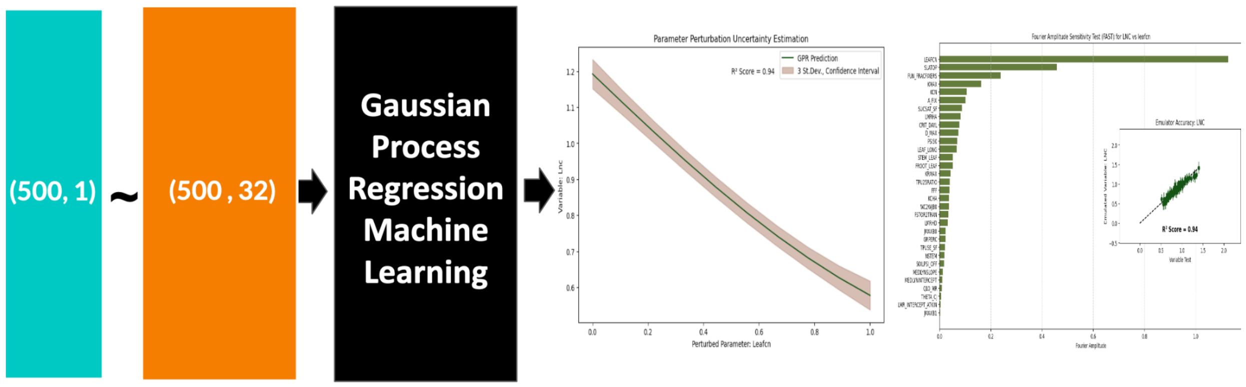


Figure 2: Diagram of CLM-5 PPE Emulator workflow: cluster of PPE simulations associated with user selected climate variable (blue), cluster of accepted parameter values (orange), 'black box' GPR ML emulator (black), GPR plot of emulation prediction of climate variable model output vs perturbed parameter with associated uncertainty (left), Fourier Amplitude Sensitivity Transformation (FAST) parameter sensitivity analysis plot and Cross Validation accuracy inset plot (right).

Defining GPR Emulator Capabilities

A Gaussian Process Regression Machine Learning (GPR ML) emulator serves as a versatile tool in the context of climate modeling calibration and understanding the influence of parameter value uncertainty on climate model outputs. Emulators are stand-ins for full climate models. The computational burden required to compute predictive outcomes is alleviated when replacing the model with an emulator. Insights that invite calibration of the climate model are made available more quickly.

GPR emulators are capable of predicting climate variable outcomes for complex relationships within climate models with confidence. The backbone of this 'black box' emulator is a nonparametric Bayesian machine learning approach (Rasmussen, 2004). When handling multivariate data, each perturbed parameter is distributed normally and their joint distribution is also Gaussian [normal]. The multivariate Gaussian distribution is defined by a mean vector, μ,and a covariance matrix, Σ. The relationship between the perturbed parameters and climate variable output is estimated using the distribution of possible functions. The emulator then computes the mean prediction based on the estimated distribution (Görtler, et al., 2019).

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix} \sim \mathcal{N}(\mu, \Sigma)$$

Eq 1

X, the mean prediction, follows a normal distribution for the mean vector, μ, and covariance matrix, Σ (Görtler, et al., 2019).

$$\Sigma = \mathrm{Cov}(X_i, X_j) = E\left[(X_i - \mu_i)(X_j - \mu_j)^T\right]$$

Eq 2

The covariance matrix, Σ, describes the shape of the distribution. It is defined in terms of the expected value E, as described above in *Eq 2* (Görtler, et al., 2019).

Leveraging a GPR ML Emulator

To put it more simply and in the context of this project, The CLM-5 PPE Emulator initially trains on a set of accepted values for perturbed parameters associated with the climate variable of interest. It then leverages this information to construct a probabilistic emulation that captures the relationships between the perturbed parameters and the climate variable of interest. The mean prediction represents the weighted combination of these probabilistic functions for the expected value of the climate variable output, while the covariance provides information about the uncertainty associated with the prediction (Görtler, et al., 2019). A unique feature of GPR is that all 32 perturbed parameter features passed through it are in a normalized fashion both individually and combined. Predictions are made by the emulator through estimating μ and Σ of the climate variable output distribution at each input point. The emulator provides predictions for the climate variable output at unobserved perturbed parameter values, complete with uncertainty estimates. Thus, alleviating the hefty computational burden associated with repeatedly running complex climate models. Enabling climate scientists the ability to gather insightful interpretations of the climate predictions alongside the confidence provided with the emulator's predictions.

Through the quantification of uncertainty, insights into the reliability and accuracy of the model predictions are gained. These emulation outputs can be used to refine and calibrate the CLM-5 climate model, leading to more confidence in the climate simulations created as a result of recorded reliability and trustworthiness. Leveraging the uncertainty alongside the emulation predictions, developments to refine climate models may be made through the climate model calibrations for emulated predictions with high uncertainty. Leading to enhancements in the ability to simulate and predict future climate scenarios more accurately. To curate stakeholder trust and build confidence in climate predictions, it is important to communicate uncertainty. When provided the full picture, stakeholders gain access to a clear understanding of the uncertainties involved. Leading to higher likelihood of these stakeholders trusting and acting upon model outputs. Establishing a standard for transparency is essential for effective science communication and policy formulation. The quantified uncertainty provided by our dashboard emulation visualizations allows policymakers and climate scientists, alike, to make informed decisions based on the reliability of the model outputs. Understanding the range of possible outcomes and their associated uncertainties helps in assessing risks and developing appropriate adaptation and mitigation strategies.

8.2 The CLM5 PPE Uncertainty Emulator

8.2.a. Emulator Specifications

This is the earliest iteration of the uncertainty emulator, therefore the foundation for the essential emulator specifications were defined clearly in this project. When tabulating uncertainty, the emulator is confined to 3 standard deviations. The choice to have a confidence interval of approximately 99.7% was to strike a balance between coverage and precision and provide a comprehensive representation of uncertainty.

Increasing the number of standard deviations would widen the uncertainty bounds and provide a more conservative estimate, it may also lead to overly broad intervals that lack precision. Juxtaposing the former, by quantifying fewer standard deviations, narrower intervals would result and potentially exclude important outlier information. For these reasons, 3 standard deviations was deemed the most robust level of uncertainty that allows climate scientists to gain insights into the accuracy and validity of the predictions. Leading to further developments in climate model calibration, uncertainty quantification of parameter value influence, and educational resources to document climate model behaviors.

8.2.b. Kernel Configuration

What is a kernel?

In GPR, a kernel function, $k(x_i, x_j)$, further defines the statistical structure used to compute the covariance between two input data points x_i and x_j (Rasmussen & Williams, 2006). The kernel function is a typically specified component of the emulator that measures the correlation between the values in the perturbed parameter space. It serves as a keystone element of GPR emulators and defines the shape and characteristics of the covariance function, enabling insights into the underlying relationships interwoven within the data. Different kernel functions contain specific flexibilities that allow them to capture various patterns embedded in the data, enabling unique uncertainty estimations at varying levels of confidence.

In simpler terms, the kernel in a GPR model encapsulates the specifications to assess the relationship between data points. It quantifies the degree of similarity or dissimilarity between the perturbed parameter values, thereby determining their influence on each other during the regression process (Rasmussen & Williams, 2006).

Kernels are of paramount importance in GPR models, as they directly impact the model's flexibility and predictive performance. The choice of kernel function influences the smoothness and complexity of the model, thereby affecting its ability to capture underlying patterns and dependencies in the data (Rasmussen & Williams, 2006). The implementation of a custom kernel configuration allows the emulator to be more tailored to specific characteristics within data. Therein, ultimately enhancing the emulator's robustness in interpretability and predictive accuracy.

Kernel configurations for this model

Below there are a collection of kernel functions that may be combined in a variety of ways to best interpolate the intricate relationships within the data. Refer to the Future Works section of this Technical Document to see the outline for potential applications of custom kernel combinations, leveraging the `select_best_kernel()` function.

Table 1: Types of kernel components that are appropriate to use to make configurations using the scikit-learn package, sklearn.gaussian_process.kernels (scikit-learn, 2024).

Kernel Name	Equation	Defining Variables	Background

Constant	$k(x_i, x_j) = constant\ value\ \forall\ x_i, x_j$	Constant value is the overall variance, x_i, x_j are parameter input vectors.	The constant kernel assigns a constant value to all pairs of data points. It is often used as a baseline to capture the overall variance in the data (scikit-learn, 2024).
Linear (sklearn DotProduct)	$k(x_i, x_j) = \sigma_0^2 + x_i \cdot x_j$	σ_0^2 represents the variance, x_i, x_j is the dot product of parameter input vectors.	The linear kernel models linear relationships between perturbed parameter values assuming a linear correlation between them. It computes the dot product between input feature vectors (scikit-learn, 2024).
RBF	$k(x_i, x_j) = exp\left(-\frac{d(x_i, x_j)^2}{2l^2}\right)$	$d(x_i, x_j)^2$ is the Euclidean distance between the parameter input vectors, l is the length scale of the parameter	The Radial Basis Function (RBF) kernel, also known as the Gaussian kernel, measures the similarity between data points based on their Euclidean distance. It assigns higher weights to nearby data points, making it suitable for capturing non-linear relationships (scikit-learn, 2024).

Matern 32 & 52	Eq 3, Matern 32, for v = $\frac{3}{2}$ $$k(x_i, x_j) = \left(1 + \frac{\sqrt{3}}{l} d(x_i, x_j)^2\right) \bullet exp\left(-\frac{\sqrt{3}}{l} d(x_i, x_j)\right)$$ Eq 4, Matern 52, for v = $\frac{5}{2}$ $$k(x_i, x_j) = \left(1 + \frac{\sqrt{5}}{l} d(x_i, x_j) + \frac{5}{3l} d(x_i, x_j)^2\right) \bullet exp\left(-\frac{\sqrt{5}}{l} d(x_i, x_j)\right)$$	$d(x_i, x_j)^2$ is the Euclidean distance between the parameter input vectors, l is the length scale of the parameter, $\left(1 + \frac{\sqrt{n}}{l} d(x_i, x_j)^2\right)$ is a flexible linear term that increases with distance, $exp\left(-\frac{\sqrt{n}}{l} d(x_i, x_j)\right)$ is the flexible exponential decay term, it decreases with distance.	The Matern 32 kernel is a specific case of the Matern family of kernels, specified in the top equation. Eq 3 is characterized by its smoothness hyperparameter setting of v = 3/2. It balances between flexibility and smoothness, making it suitable for modeling datasets with moderate levels of noise and non-linearities. The Matern 52 kernel, shown in Eq 4, has a smoothness hyperparameter setting of v = 5/2, resulting in a smoother function compared to Matern 32. Matern kernels pair nicely with the RBF kernel to increase flexibility (scikit-learn, 2024).
Polynomial	$$k(x_i, x_j) = (\delta < x_i, x_j > + coef0)^{degree}$$	δ is a scalar for the dot product of the input parameter vectors, $coef0$ is a constant term, degree is the power of the polynomial.	The polynomial kernel interprets polynomial relationships between parameter values, allowing for the capture of non-linear dependencies (scikit-learn, 2024).
White Noise	$$k(x_i, x_j) = noise\ level\ if x_i == x_j\ else\ 0$$	When the input parameter vectors are not equal, a constant noise value is applied.	The white noise kernel, also known as the bias kernel, as aids in the standardization of the noise-level of the signal (scikit-learn, 2024).

As outlined in *Table 1*, there are a collection of kernel functions that may be combined mathematically to better interpolate various relationships. For the earliest iteration of The CLM5

PPE Emulator, the following kernel configuration shown in *Eq 5,* was selected for this early iteration due to its simplicity and manageability.

$$kernel = ConstantKernel(constant\_value = 3,\ constant\_value\_bounds = (1e - 2,\ 1e4)) \bullet RBF(length\_scale = 1,\ length\_scale\_bound = (1e - 4,\ 1e8)) \qquad \text{Eq 5}$$

The RBF kernel is highly flexible and can capture complex non-linear relationships in the data. This is crucial for environmental data, which often exhibits intricate patterns and interactions between variables. The Constant kernel is applied to set boundaries to capture model variance. The boundaries established for the hyperparameters were selected based on the range of values provided by the emulation outputs. Limitations associated with the kernel figuration are that it lacks dimensionality in terms of being able to assess each perturbed parameter feature on their own individual length scale. Instead all perturbed parameters are assessed in the same dimensional space. To work around these limitations currently within the kernel, an iteration clause was crafted to enable each perturbed parameter to be individually assessed in the parameter space. This allows the emulator to be trained to interpolate each parameter's influence on the climate variable output. Upon training, the trained emulator is pickled and stored for later use upon user-querying.

8.2.c Pickling the GPR Emulator

What's a pickled emulator?

The Pickle package in Python that aids the serializing and deserializing trained emulators and their associated data (Python Software Foundation, 2024). It saves emulation objects as a file in a format that can be easily read and reused in the workflow at a later time. Pickle essentially allows the dashboard to save our trained emulator data and load it back into memory when selected by a user, preserving all the information and functionalities of the original object.

How is it applied in our model?

In the emulator workflow, we utilize the Pickle package to store emulation results for future use and 'unpickle' those objects employing an if-else statement. At the beginning of our workflow, we embedded a conditional statement that checks if a climate variable has been selected in the past. If the relationship has been previously emulated and stored, the dashboard retrieves the saved emulation results and displays them for the user. Otherwise, if the climate variable is new and has not yet been previously computed, the model proceeds to run the emulation and stores the data for future use.

Why is it beneficial?

The use of pickling in our model offers several benefits. Firstly, it saves significant time and computational resources by avoiding the need to rerun emulations for relationships that have already been computed. This efficiency not only streamlines the user experience but also reduces the computational burden and carbon impact associated with running the model repeatedly. Additionally, pickling ensures that previously computed results are readily available for analysis, enabling seamless exploration and interpretation of model outputs. Overall, pickling enhances the efficiency, accessibility, and sustainability of our model workflow.

Products and Deliverables

The goal of this project is to develop an interactive emulator for the results of the Parameter Perturbation Experiment. The emulator will allow NCAR scientists to view the parameter and variable definitions, the relationship between a variable and parameter of interest, and the uncertainty of that predicted relationship. In order to achieve this, the dashboard will have the following components:

- A splash page with a description of the experiment setup, how the simulation data was collected, and a full-length description of the 32 perturbed parameters and the 10 most commonly used climate output variables
- A main page that will allow users to select a parameter and variable of interest from a drop down menu, and display the following visualizations
    - The predicted relationship between the selected parameter (x-axis) and variable (y-axis), along with the 99% confidence interval for the estimate
    - The results of a parameter sensitivity analysis for the selected variable, showing which parameters have the highest influence on the selected variable
    - An accuracy plot to show the comparison between the measured and predicted values

Our team has structured the dashboard so that staff at NCAR will be able to expand upon the capabilities of the dashboard after the project is completed in June. To help accommodate NCAR staff as they build upon the existing scaffolding, the scope of the project includes:

- Implementing continuous integration into the containerization of the dashboard to launch future work automatically
- Adding a substantial 'Maintenance and Future Work' section to the Technical Documentation
- Developing a cleaned, organized GitHub repository with Jupyter Notebooks outlining the progress made on suggested future work

Summary of Testing

10.1 Effective and Insightful Data Visualization

Quality Assurance Measures

Throughout the development of the workflow, the emulator components continually underwent dimensional assessment. This was to guarantee the components being considered contain all of the desired information required to interpret quality predictions. When configuring our emulator, a series of qualitative checks were conducted using testing metrics such as the R^2, Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE). The acceptable range of emulation accuracy defined by the team was for the top 10 most common climate variables to provide an R^2 of 0.65 or greater. Additionally, visualizations such as GPR emulation plot, Fourier Amplitude Sensitivity Transformation, and a Cross Validation Accuracy plot were produced to assess the performance of the emulator and accuracy.

Initially, when testing functionality of the emulator, the climate variable (Leaf Nitrogen Concentration, LNC) and perturbed parameter (Leaf Carbon to Nitrogen Ratio, Leafcn) relationship was selected due to their highly correlated relationship. The GPR emulation plot was the primary indicator to determine how well the model was performing, the expected output was a negative linear trend with high correlation, as seen below in *Figure 3*.

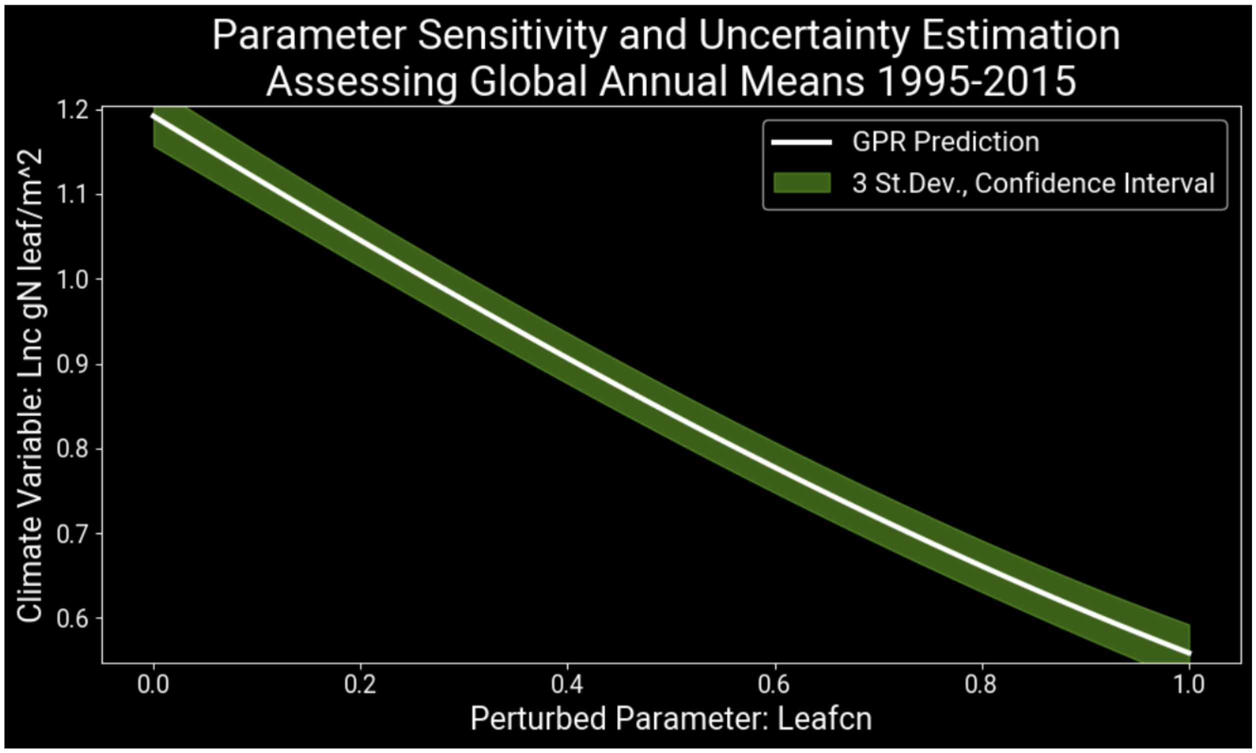


Figure 3: GPR Emulation Prediction plot for the Climate Variable Leaf Nitrogen Concentration (LNC) and Leafcn. Shaded region displays uncertainty up to 3 standard deviations.

Once the initial testing relationship revealed the expected predictive trends with minimal uncertainty, the emulator testing expanded to the 10 most common climate variables. A script was created to save png images in a designated folder of all the plots generated for the model predictions. The team later reviewed these images to assess the reliability, validity, and accuracy of the emulator.

GPR Emulation Prediction Plot

The GPR emulation plot visualizes the predicted relationship between the user-selected perturbed parameter and the climate variable. It provides an intuitive understanding of how variations in parameter values influence predicted climate variable outcomes. The plot also incorporates uncertainty estimation, represented by shaded regions indicating up to three standard deviations, offering insight into the reliability of predictions.

Fourier Amplitude Sensitivity Transformation (FAST) Plot

The Fourier Amplitude Sensitivity Transformation (FAST) plot is a technique used for conducting parameter sensitivity analysis in computational modeling (Fang, Gertner, et. al., 2003). It decomposes the variance of climate variable output into contributions from individual perturbed parameters. Thus, allowing for the assessment of each parameter's impact on model outcomes. *Figure 4* displays the sensitivity of the predicted climate variable to variations in each perturbed parameter, providing valuable insights into the relative importance and influence of different parameters on that climate variable.

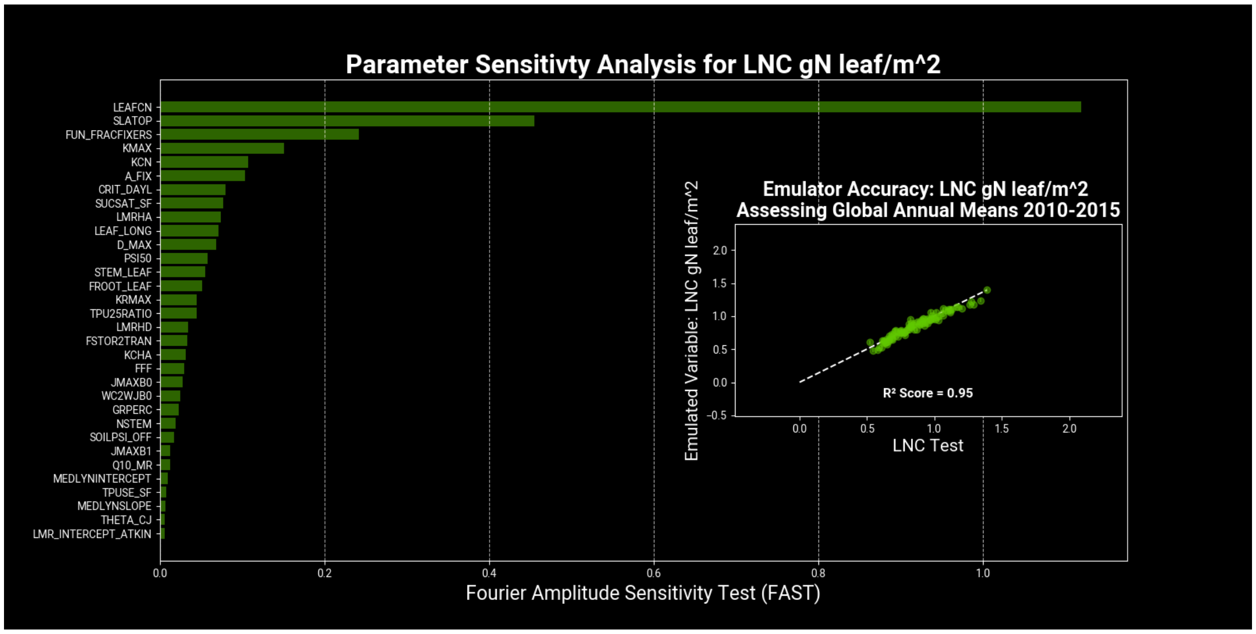


Figure 4: Emulation Fourier Amplitude Sensitivity Transformation plot with Cross Validation Accuracy plot inset for the Climate Variable Leaf Nitrogen Concentration (LNC) and Leafcn. Shaded region displays uncertainty up to 3 standard deviations, R^2 is 0.95.

The FAST plot serves as a tool for parameter sensitivity analysis, leveraging variance to assess the influence of individual parameters on the selected climate variable. By examining this plot, users can discern which parameters exert the most significant impact on the climate variable under study, aiding in understanding key drivers of model outputs.

Accuracy Plot

Within each FAST plot, a cross validation plot is produced to provide quality assurance alongside the prediction using a subset of the predicted climate variable output to a subset of the test data. This plot facilitates the comparison between predicted climate variable values generated by the emulator and test values. Additionally, the plot includes the coefficient of determination R^2 derived from the emulation, quantifying the model's predictive performance. A perfect alignment with the reference line denotes flawless prediction accuracy and would yield an R^2 of 1. The Accuracy Plot enables users to evaluate the emulator's ability to faithfully model the data and predict outcomes, along with their associated uncertainty.

User Documentation

11.1 Dashboard User Manual

Welcome to the User Manual for the Community Land Model 5 Parameter Perturbation Experiment Dashboard! This tool is designed to allow users to choose a parameter, variable, and time-frame of interest to create custom visualization for analysis within the dashboard. Below is a quick overview of the experimental setup, followed by step-by-step instructions on how to effectively use this tool.

11.1.a Overview of the Project

This project focuses on using Gaussian Process Regression (GPR) to model and predict climate data. Specifically, the project involves perturbing a global land model with 32 different parameters. Each time a single parameter is altered, the responses of the other 31 parameters are recorded. This process produces a comprehensive sample of the parameter space, which includes 471 different output variables such as surface temperature, leaf nitrogen content, and soil moisture. The data is stored in NetCDF format for efficient storage and retrieval.

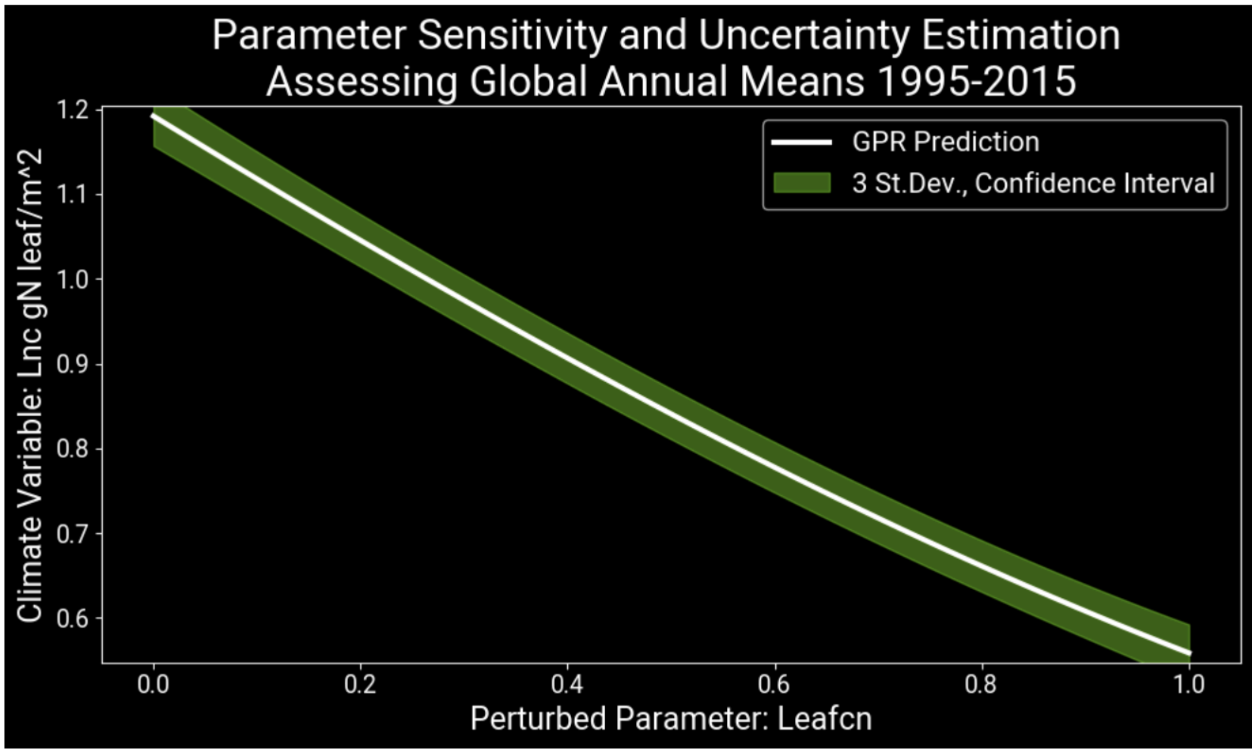
The Gaussian process regressor is trained on this sample data to learn the relationships between parameters. Once trained, the model can predict the values of output variables based on new input values. For instance, a sample of 10 different values between 0 and 1 can be provided to the trained model, and it will predict the corresponding output values.

11.1.b Information on Gaussian Process Regression

Gaussian Process Regression (GPR) is a machine learning model used for predicting outcomes based on input data. It assumes the data follows a Gaussian distribution and makes predictions by learning the relationships between input and output variables from training data. When a new input is provided, the GPR predicts the possible outputs and also provides a measure of uncertainty through standard deviation. This standard deviation indicates how confident the model is about its predictions; a smaller standard deviation means higher confidence, and a larger one indicates more uncertainty. GPR is particularly useful in fields like climate science, where understanding the variability and confidence in predictions is crucial. Unlike polynomial regressors, which fit a fixed polynomial equation to the data, GPR models the data using a probability distribution, making it more flexible and capable of capturing complex relationships and uncertainties. This flexibility helps scientists make more informed decisions based on the model's output and its associated confidence levels.

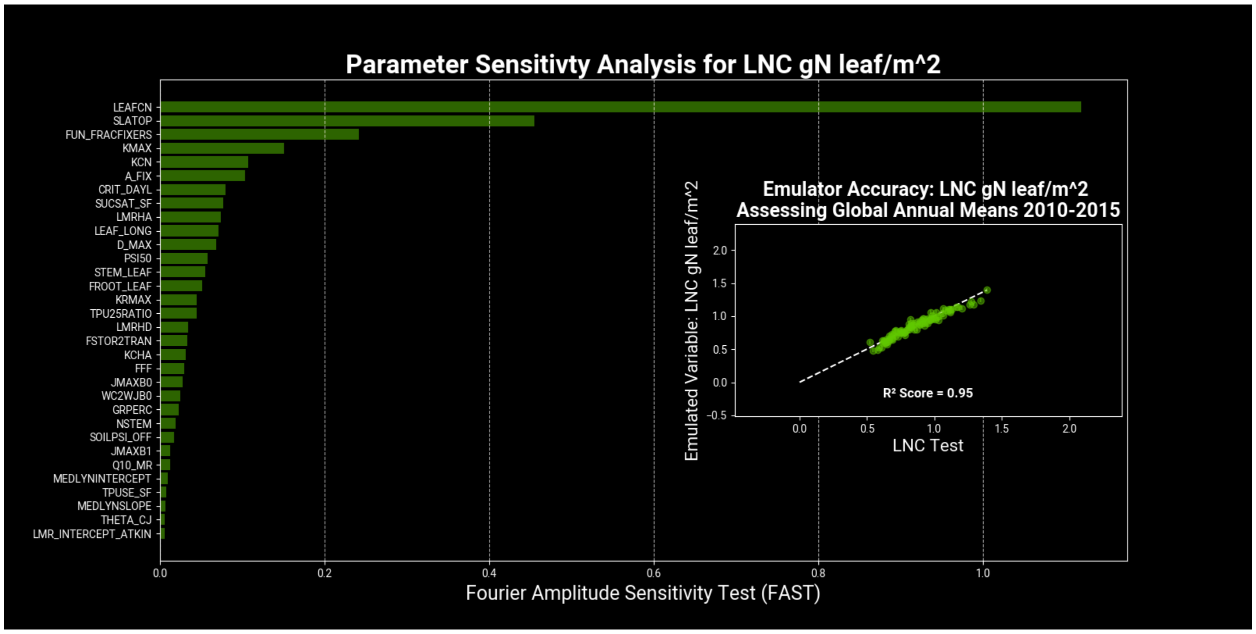
11.1.c Dashboard Output Explanation

The Dashboard generates two visualizations for each new selection made using the drop down menus. The first visualization is an emulator plot featuring the climate variable predictions, and a three standard deviation range of uncertainty. In the emulator plot. The emulator plot shown below shows the relationship between a parameter (Leafcn, the Carbon to Nitrogen Ratio in a Leaf) and a variable (LNC, the nitrogen concentration in a leaf) for the time range 1995-2015.



Another plot produced below consists of two subplots. The horizontal bar plot represents the sensitivity of each of the 32 parameters. The parameter with the greatest influence on a variable has the greatest height, and so on.

The smaller nested plot within the second visualization is a cross-validation plot comparing the emulator's predicted result with the output data from the Community Land Model 5. This comparison is measured using the R^2 value. An R^2 value greater than 0.7 indicates good performance of the model, while values lower than that indicate poor performance.



11.2 Maintenance Documentation for Software Personnel

This document provides detailed instructions for using and maintaining the GPR Emulator with NetCDF datasets. It is intended for scientists at NCAR who work with climate datasets but have no experience with supercomputing or Gaussian process regression in a supercomputing environment. The guide covers setting up the Python environment, configuring resources on the remote server, running the emulator, using the dashboard interface, and troubleshooting.

Python Environment Setup

Establishing a suitable Python environment is crucial for running the Emulator (machine learning model) smoothly. The runtime environment should have following packages and versions:

Package	Version	Description
xarray	2024.1.1	A powerful library for working with labeled multi-dimensional arrays, particularly useful for handling netcdf data with complex dimensions and metadata.
pandas	2.2.0	Used for data manipulation and analysis of, especially for tabular data structures like dataframes.

numpy	1.26.3	Essential for numerical computations, providing efficient array operations and mathematical functions.
matplotlib	3.8.2	A versatile plotting library for creating various types of visualizations, enabling to explore and visualize climate data effectively.
scikit-learn	1.4.0	Offers a wide range of machine learning algorithms and tools, including Gaussian Process Regression, which is utilized in the emulator.
panel	1.3.8	Enables the creation of interactive dashboards, allowing users to interactively explore and analyze emulator results.
dask	2024.1.1	Ideal for parallel computing and handling large datasets, which is common in climate research.

Cluster Configuration

To speed up data processing, use the get_cluster() function to set up a Dask cluster. This function, located in the project's utils folder, configures resources with 40 cores, 40 threads, and 4GB RAM per core. If get_cluster() is not called, set the following options:

- **Number of Cores**: 40
- **Memory Allocation**: 4GB per core
- **Walltime**: 12 hours (adjust as necessary)

This configuration maximizes computational resources and expedites the emulation process.

Data Preparation and Emulator run

Ensure the following data and information are ready:

Parameter Data Array: Collect and preprocess the required parameter data using functions from the utils folder. Use read_all_simulation() to read the parameter set and variables of interest.

Running the Emulator

Execute the train_emulator() function, which includes the following steps:

1. **Data Splitting**: Divide the data into training and testing sets (80:20 ratio)

2. **Model Training**: Train a Gaussian Process Regression model using the training data
3. **Prediction**: Make predictions using the trained model
4. **Evaluation**: Assess the model's performance using evaluation metrics like R squared
5. **Visualization**: Plot predicted values, uncertainty estimates, and regression lines

Troubleshooting

Common Issues and Solutions

1. **Data Availability**: Ensure input data has the shape [500, 32]. Adjust rows as necessary but maintain 32 columns
2. **Function Arguments**: Verify the arguments passed to functions are correct and consistent. Information about each function can be found ‘?function_name`
3. **Error Messages**: Pay attention to error messages and traceback information. Ensure output variables are single arrays with shape [500,]
4. **Utils Package**: Ensure the utils package is included in the ‘meds conda’ environment
5. **Resource Cutoff**: Periodic maintenance at NCAR may limit Dashboard operation. If the emulator is slow, check NCAR resources

Best Practices

To maximize the effectiveness of the Emulator, consider the following best practices:

- **Optimize Resources**: Adjust cluster resources based on analysis complexity and available computational resources
- **Validation**: Validate emulator results using known data or benchmarks
- **Exploration**: Explore different parameter ranges and scenarios to understand the sensitivity of climate variables
- **Documentation**: Document emulator settings, parameter ranges, evaluation metrics, and findings for reproducibility

Archive Access

While the data used for this project is not technically proprietary, the dataset is very large and currently being stored on the NCAR server. The National Center for Atmospheric Research has plans to publish the full dataset using their current standard for data archival and access. The code produced during this project is available through the GitHub Organization and the much smaller, pre-processed emulator datasets are available through Dryad using the DOI: 10.5281/zenodo.11291029.

References

1.7. Gaussian Processes. (n.d.). Scikit-Learn. Retrieved March 15, 2024, from https://scikit-learn/stable/modules/gaussian_process.html

A Visual Exploration of Gaussian Processes, Görtler, et al., Distill, 2019. https://distill.pub/2019/visual-exploration-gaussian-processes/

Academic Resources for Current Students. (n.d.). Capstone Project Guidelines | MEDS Class of 2024. UCSB Bren School of Environmental Science & Management. Retrieved March 15, 2024, from https://bren.ucsb.edu/academic-resources-current-students

Diagnostics - NCAR static site with preprocessed images (n.d.). Retrieved March 15, 2024, from https://webext.cgd.ucar.edu/I2000/PPEn11_OAAT/

EDS 411 - EDS 411: MEDS Capstone. (n.d.). Retrieved February 19, 2024, from https://carmengg.github.io/eds-411-website/

Good enough practices to manage your project data—Managing your data. (n.d.). Retrieved March 15, 2024, from https://ucsb-library-research-data-services.github.io/project-data-management/manage.html

Gaussian Processes for Machine Learning. Rasmussen, C. E., & Williams, C. K. I. (2006). MIT Press. Accessed 3 March 2024.

Gaussian Processes in Machine Learning. In: Bousquet, O., von Luxburg, U., Rätsch, G. (eds) Advanced Lectures on Machine Learning. Rasmussen, C.E. (2004). ML 2003. Lecture Notes in Computer Science(), vol 3176. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-28650-9_4

Improved generalized Fourier amplitude sensitivity test (FAST) for model assessment. Fang, S., Gertner, G.Z., Shinkareva, S. *et al.* | Statistics and Computing **13**, 221–226 (2003). https://doi.org/10.1023/A:1024266632666

NCAR Climate Capstone Proposal—Childers, H., Kennedy, D., Ingersoll, S. (n.d.). Retrieved March 15, 2024

Pickle - Python Object Serialization. Python Documentation, docs.python.org/3/library/pickle.html. Accessed 13 May 2024.

Sklearn.Gaussian_process.Kernels.DotProduct. Scikit, scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.kernels.DotProduct.html. Accessed 13 May 2024.

Sklearn.Metrics.Pairwise.Polynomial_kernel. Scikit, scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.polynomial_kernel.html. Accessed 13 May 2024.

The CLM5 Parameter Perturbation Experiment | Climate & Global Dynamics. (n.d.). Retrieved March 15, 2024, from https://www.cgd.ucar.edu/events/seminar/2023/katie-dagon-and-daniel-kennedy-132940

The role of satellite remote sensing in climate change studies. Yang, Jun & Gong, Peng & Fu, Rong & Zhang, Minghua & Chen, Jing & Liang, Shunlin & Xu, Bing & Shi, Jiancheng & Dickinson, Robert. (2013). Nature Climate Change. 3. 875-883. 10.1038/nclimate2033. DOI: 10.1038/nclimate2033

What Is a Climate Model? NCAS, National Centre for Atmospheric Science, ncas.ac.uk/learn/what-is-a-climate-model/. Accessed 3 May 2024.