Creating a Benchmark for Data-Driven Climate Projections

Keith Kwong
khkwong@ucsd.edu

Jun Zhang
juz018@ucsd.edu

Jack Kai Lim
jklim@ucsd.edu

Duncan Watson Paris
dwatsonparris@ucsd.edu

Abstract

Climate emulation has long been limited by the amount of resources needed to use legacy built Earth system models. As such, exploration into the many different and possible emission pathways have been relegated to one-dimensional impulse response or simple pattern scaling models, neither of which are capable of accounting for finer details when emulating the Shared Socioeconomic Pathways. Here we introduce ClimateBench - (Watson-Parris et al. 2022), the first set of baseline machine learning models capable of emulating the response of full complexity Earth system models to forcers using a fraction of the time and resources and a benchmarking framework to compare any future models with these baselines. These models are capable of predicting annual mean global distributions of temperature, diurnal temperature range, and precipitation given the emissions and concentrations of carbon dioxide, methane, sulfur dioxide, and black carbon. We also discuss the accuracy and interpretability of the models, as well as its robustness to known physical constraints.

Code: https://github.com/khkwong/ClimateBench-remake

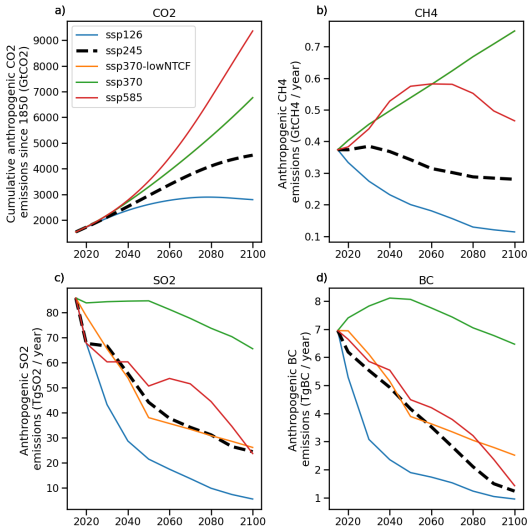
1 Introduction

Many emissions pathways can achieve the goal of the Paris Agreement which limits the increase of global temperature under 2 celsius degrees. The policymakers need to assess different social and economic impacts under different emission scenarios to reduce the effect of climate change and achieve the goal. The machine learning models that emulate different emissions scenarios play an important role in this process of weighing. Our models aim to predict the global temperature, diurnal temperature, and precipitation(including extreme precipitation) under different emission scenarios which help the policymakers and general public assess which possible emission scenarios can achieve the targetWatson-Parris et al. (2022).

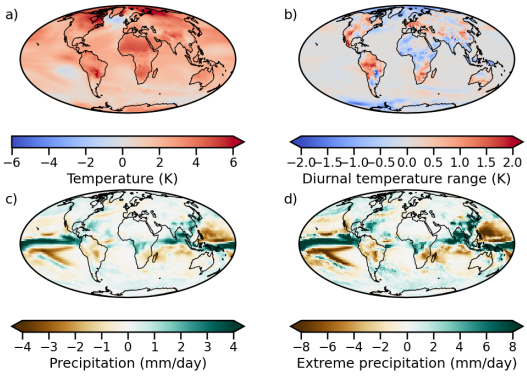
There are a number of previous models that have been used in a similar manner to ClimateBench - Watson-Parris et al. (2022), such as impulse response models and pattern scaling models. Both have their demerits, however, with impulse response models having the ability to capture general non-linearity in the system, but being unable to model regional climate changes and pattern scaling models' reliance on scaling of spatial distributions of global mean temperature changes meaning it is unable to accurately emulate precipitation due to its strong non-linear relationship with temperature. Statistical emulators for regional climate have been built in the past, however they tend to be too regionally specific or only capable of simple emulations of temperature. Furthermore, none of these current approaches to climate emulation account for the influence of aerosols such as black carbon and sulfur dioxide, which are capable of affecting both regional temperature and precipitation. In the realm of machine learning, there have been recent studies diving into the potential of using Gaussian process regression and non-linear pattern scaling for climate emulation, but due to their recency, their performance cannot be appropriately judged yet due to a lack of a benchmark to compare withWatson-Parris et al. (2022).

The training data that is going to be used for the models training is from the Norwegian Earth System Model (NorESM2; Seland et al., 2020) which is generated data from simulations performed by the NorESM2 model. This was done as part of the sixth coupled model intercomparison project(CMIP6 ; Eyring et al.,2016). The data is used by the policymakers when deciding climate policies, so that the emissions data that come from the NorESM2 and CMIP allows ClimateBench - Watson-Parris et al. (2022) to predict outcomes of different possible scenarios that align with the policymakers' want to reduce climate change. The data that is extracted from NorESM2 and CMIP are netcdfs which are multi-dimensional containing data on every latitude and longitude for the emissions and aerosols that we are looking at i.e Carbon Dioxide, Methane, Sulfur Dioxide, and Black Carbon and span from the 1850s to the end of 2014.

The input variables include emission data of **CO_2**, **SO_2**, **CH_4**, and **Black Carbon** by different emission scenarios data and historical data. We took the global average of input variables Co_2 and CH_4 by year. We convert the units of SO_2 and Black Carbon into Tera gram and take the global sum. Through this process, we acquire the patterns of four different input variables from 2020 to 2100. The output variables are the temperature (T), Precipitation, 90th percentile precipitation, and daily diurnal temperature range.



(a) Input Variables



(b) Output Variables

Figure 1: (a) Line Graphs of Input Variables from Multiple SSP (b) Map of Output Variables from SSP245

2 Methods

As for the models, as our goal is to reproduce professor Duncan's ClimateBench - (Watson-Parris et al. 2022) work, we also worked on 3 of the same models that we did which are a **Gaussian Process Regressions**, a **Random Forest** and a **Convoluted Neural Network**.

2.1 Gaussian Process Regression

A Gaussian Process is a probabilistic statistical model which is used to describe the Bayesian a priori uncertainty about a latent function. It is different from many other machine learning models, a Gaussian Process is non-parametric meaning it can adapt to any complexity of data and not be constrained by predetermined number of parameters. The way the Gaussian Process works is by first starting with a prior distribution which should reflect the beliefs of a function space of a kernel. The kernel which is also known as the covariance function is an important part of the Gaussian Process as it is what defines the covariance between 2 points which the the same as stating the similarities between 2 points.

Advantages that come with using the Gaussian Process are that one can get uncertainty estimates from the Gaussian Process as at each prediction point the model also gives a measure of confidence of its predictions. In addition to that, Gaussian Processes are also extremely flexible due to the choice of a kernel, as there are a large variety of different possible kernels allowing various different fittings of the data. Examples of some common kernels are the **RBF**, **Matérn**, **Periodic**, **Linear**, and **Polynomial** Kernels. At the same time however, the kernel is also one of the reasons why Gaussian Processes are complex as the choice of kernel does affect the results drastically but the choice of the right or the best

kernel is incredibly non-trivial.

When fitting the data on to the Gaussian Process, we notice that the dimensions of $\mathbf{SO_2}$ and **BC** have 5 dimensions each. The reason for this can be found in 5.1. So to account for that we used Principle Component Analysis which explains around 96% to 98% of the variance and built a 'pca_solver'. This is important as the Gaussian Process will have Automatic relevance determination (ARD) kernels which allow the model to treat each principle component individually and eventually output a value in one dimension which will need to be solved using the 'pca_solver' in order to be used in validation.

For further understanding of Gaussian Processes and how they worked, Görtler, Kehlbeck and Deussen (2019) is a good reference and is the reference I used as well.

2.2 Random Forest

The Random Forest model is a way to combine the predictions of multiple decision trees into the result. Each decision tree has root nodes, leaf nodes, and decision nodes. The leaf nodes include different questions which continue to split the data points in order to reach a final decision. The advantage of Random Forest models when it comes to climate data is the high accuracy and robustness to outliers. Yet, the forest is not linear and tends to result in overfitting. It requires hyperparameters tuning and cross-validation to alleviate that overfitting. Thus, we have applied the Randomized Search to the random forest models. To use it, We have defined a parameter grid, such as the number of levels for the tree, maximum depth for the tree, and number of samples for the leaf nodes. Due to the length of our input and output variables, the process of fitting Random Forest models with hyperparameters requires an enormous time to be finished. The generation of Random Forest also requires more memory. For fitting different training data, we got different structures of forests for four different variables as follows:

Table 1: A 6x5 Table

Hyperparameter	**Number of trees**	**Min samples split**	**Min samples leaf**	**Maxdepth**
Temperature	100	10	4	10
Diurnal Temperature	150	25	4	45
Precipitation	100	25	12	30
90th Precipitation	100	15	12	25

In terms of data, the Random Forest model is also required to apply principal component analysis to reduce the dimensions of aerosol emission maps. For example, the first five principal components of SO_2 and BC across different scenario datasets would be used as the input variables. Meanwhile, the global emission maps CO_2 and CH_4 from different scenario datasets would be the input variables. The goal of the Random Forest model is to predict the four climate variables. Therefore, there are four separate Random Forests that correspond to four different output variables temperature, diurnal temperature, and precipitation(including extreme precipitation).

2.3 Neural Networks

Artificial Neural Networks were chosen to be the third baseline model for ClimateBench -(Watson-Parris et al. 2022) because there are several different types of neural networks that have been shown to be able to handle spatial and temporal data. Recurrent Neural Networks(RNN) have been used to process time series and sequential data while Convolutional Neural Networks (CNN) have been used in great effect in fields such as computer vision and image processing due to its capabilities in learning spatial patterns. As such, these two architectures can be utilized in sequence in order to effectively garner results from the time series of spatial data found in the ClimateBench data set. The resulting bench marking model consists of a CNN followed by a long short term-memory (LSTM) network which has been proven to be an effective RNN in modeling time series data.

The final architecture of our CNN-LSTM approach is as follows: a convolutional layer with 20 3x3 filters and ReLU activation followed by an average and global average pooling layer, reducing the input shape down to two dimensions for the upcoming LSTM layer using 25 units and ReLU activation function. The final two layers are a dense and reshaping layer of size (96, 144) to regain the latitude and longitude dimensions for our final outputs. All of this was built using Tensorflow and the Keras library. The input data was also split into chunks of 10-years, moving in one-year increments prior to training the CNN. Each of the different output variables was trained on separate versions of the architecture outlined above, resulting in four emulators that were trained for 30 epochs each and a batch size of 16 using mean squared error as the optimizer function.

3 Evaluation

The goal of the machine learning models in ClimateBench - (Watson-Parris et al. 2022) is to predict the global temperature, diurnal temperature, and precipitation given the emissions of Carbon Dioxide CO_2, Methane CH_4, Sulfur Dioxide SO_2, and Black Carbon under the test scenario ssp245. As such, performance of the models are evaluated against the actual values provided by the NorESM2 model under the same scenario. RMSE (root mean squared error) have historically been used in the evaluation of regression tasks, as a loss function that provides a numerical value that is the same unit as the outputs. Taking into account that climate predictions also have a spacial component to them that need to be accounted for, the final metric we chose to evaluate the baseline models is a combination ($NRMSE_t$) of the normalized, global mean root-mean square error ($NRMSE_s$) and the NRMSE in the global mean ($NRMSE_s$). These are calculated as follows:

$$NRMSE_s = \sqrt{\langle (|x_{i,j,t}|_t - |y_{i,j,t,n}|_{n,t})^2 \rangle / |\langle y_{i,j} \rangle|_{t,n}} \tag{1}$$

$$NRMSE_g = \sqrt{\langle |(|x_{i,j,t}| - |y_{i,j,t,n}|_n)^2|_t \rangle / |\langle y_{i,j} \rangle|_{t,n}} \tag{2}$$

$$NRMSE_t = NRMSE_s + \alpha * NRMSE_g \tag{3}$$

4 Results

Throughout the reproduction of Duncan Watson Paris' ClimateBench - (Watson-Parris et al. 2022) paper, we have also reproduced 3 baseline emulators which demonstrated different methods to use the data set from NorESM2 in dealing with the problem posed to us. Below we can find the table which represents the Spatial, Global and Total NRMSE of all the emulators that were build as part of this project which we compared against the future scenarios **SSP245**. Table 2 is the table of NRMSEs that our models produced.

Table 2: The Spatial, Global and Total NRMSE of the Different Baseline Emulators for the Years 2080–2100 Against the ClimateBench Task of Estimating Key Climate Variables Under Future Scenario SSP245 By Our Models

	NRMSE Temperature			NRMSE Diurnal Temperature Range			NRMSE Precipitation			NRMSE 90th Precipitation		
	Spatial	Global	Total	Spatial	Global	Total	Spatial	Global	Total	Spatial	Global	Total
Gaussian Process	0.091	0.044	0.309	9.195	2.650	22.443	2.330	0.378	4.222	2.605	0.395	4.582
Neural Network	0.097	0.044	0.317	8.431	1.226	14.563	2.264	0.178	3.153	2.516	0.373	4.380
Random Forest	0.452	0.398	2.442	13.161	2.771	27.017	5.599	0.924	10.217	6.734	0.996	11.712

now below here is the Table 3 from Duncan's ClimateBench paper,

Table 3: The Spatial, Global and Total NRMSE of the Different Baseline Emulators for the Years 2080–2100 Against the ClimateBench Task of Estimating Key Climate Variables Under Future Scenario SSP245 By ClimateBench

	NRMSE Temperature			NRMSE Diurnal Temperature Range			NRMSE Precipitation			NRMSE 90th Precipitation		
	Spatial	Global	Total	Spatial	Global	Total	Spatial	Global	Total	Spatial	Global	Total
Gaussian Process	0.109	0.074	0.478	9.207	2.675	22.582	2.341	0.341	4.048	2.556	0.429	4.702
Neural Network	0.107	0.044	0.327	9.917	1.372	16.778	2.128	0.209	3.175	2.610	0.346	4.339
Random Forest	0.108	0.058	0.400	9.195	2.652	22.457	2.524	0.502	5.035	2.682	0.543	5.399

And here is another Table 4 to show the absolute differences between the reproduction (Our Models) compared to the table presented on ClimateBench

Table 4: The Absolute Differences of the Spatial, Global and Total NRMSE of the Different Baseline Emulators for the Years 2080–2100 Against the ClimateBench Task of Estimating Key Climate Variables Under Future Scenario SSP245 Between ClimateBench and Our Models

	NRMSE Temperature			NRMSE Diurnal Temperature Range			NRMSE Precipitation			NRMSE 90th Precipitation		
	Spatial	Global	Total	Spatial	Global	Total	Spatial	Global	Total	Spatial	Global	Total
Gaussian Process	0.066	0.017	0.169	6.557	6.52	0.139	1.963	1.989	0.174	2.161	2.176	0.12
Neural Network	0.063	0.053	0.01	8.691	7.059	2.215	1.95	2.055	0.022	2.237	2.17	0.041
Random Forest	0.29	0.394	2.042	6.424	10.509	4.56	1.6	5.097	5.182	1.686	6.191	6.313

As we can see in Table 4, for most of the NRMSE for *temperature*, there was very little variability between our models and the ClimateBench - (Watson-Parris et al. 2022) with most of the NRMSEs ≈ 0. This is also attribute to the low NRMSEs values from the models which imply that the models were able to predict the *temperature* variability based of the Aerosols relatively well.

As for the *Diurnal Temperature Range*, the difference in the NRMSEs scores between our models and ClimateBench - (Watson-Parris et al. 2022) differed a little more. There were outliers like **Random Forest's Global NRMSE** as there was a difference of **10.509** which is the largest among all of the differences. We are not entirely sure why this is the case, but it could be something to explore to understand why this variability exists.

Now looking at both *Precipitation* and 90^{th} *Precipitation* the NRMSEs differences between our models and ClimateBench - (Watson-Parris et al. 2022) vary more than for *temperature* but less then *Diurnal Temperature Range*. With some of the variables varying less than others, key outliers that stand out is the **Total and Global NRMSEs for Random Forests** while all other NRMSEs scores stay relatively below 2 which is a good range.

In addition to the tables that compare the NRMSE performance of our Models against ClimateBench - Watson-Parris et al. 2022 Figure 2 is a graph that shows the mean difference **Temperature**, **Diurnal Temperature Range**, **Precipitation** and **Extreme Precipitation** between our models and ClimateBench.

Looking at the graph we can see that the variation between the output variables between our models and ClimateBench do not vary much. With **Temperature** only varying from around $[1.5, -1.5]$, for **Diurnal Temperature Range** it is around $[1, -1]$. For **Precipitation** it is $[1.2, -1.2]$ and finally for **Extreme Precipitation** it has the largest variance with around $[3, -3]$. However, it shows that our model on average got extremely close to the results obtain by ClimateBench which shows it's reproducibility.

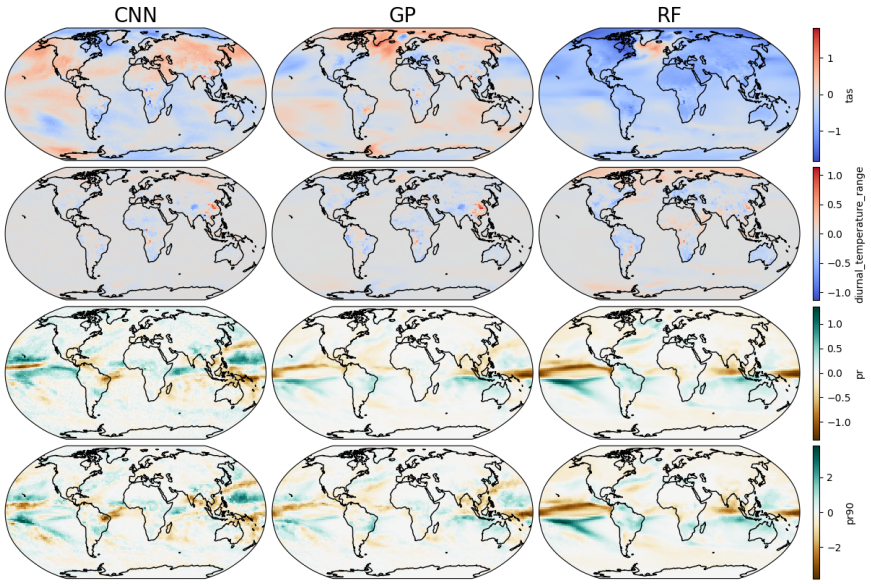


Figure 2: Graph to show the mean difference of output variables between Climate-Bench and our Reproduction

5 Discussion

5.1 Aerosol Dimension Differences

As we can see from the cleaned data, for $\mathbf{CO_2}$ and $\mathbf{CH_4}$ there is only 1 dimension and for $\mathbf{SO_2}$ and **BC** has 5 dimensions. The reason for this is because $\mathbf{SO_2}$ and **BC** would only stay in the atmosphere for a few days to a week. While $\mathbf{CH_4}$ would stay in the atmosphere for years and millennia for $\mathbf{C0_2}$. This is why they have more dimensions, as they would be out of our atmosphere before they get mixed completely into our environment and have a global effect. Hence, the dimensions give us the distributions of $\mathbf{SO_2}$ and **BC** to account for where they are being emitted.

5.2 Models Trustworthiness

Furthermore, each model leads to different predictions. As we can see, the Random Forest models have higher global NRMSE scores for the diurnal temperature. The scores for precipitation data are also worse than other models. It indicates that this model might need to be optimized by a deeper tree structure. Some of the precipitation training data are also more extreme than the test data. It could cause the prediction of test data to vary from the actual data and lead to higher NRMSE scores. The scores of diurnal temperature range variables also are significantly different from scores of ClimateBench's paper. We couldn't identify what caused that huge difference. The possible reasons can be overfitting or the variable itself.

For the Gaussian Process, overall performed extremely well, getting relatively low NRMSEs scores across the board, even comparing it with the Neural Network and the **ssp245** emulator. However, there is a little question to be asked on the reproducibility of the Gaussian Process, as a Gaussian Process essentially works as a Regressor so it has the chance during training to get a different minimal even under the same circumstances. This cause an interesting comparison to happen between my figure (Jack) for the **temperature** compared with Duncan's ClimateBench - (Watson-Parris et al. 2022). Where when validating the trustworthiness of both Gaussian Process Models by calculating the mean difference against the **ssp245** model my Gaussian Process had a very different map and distribution of the temperature change compared to ClimateBench. Which as mentioned earlier is likely due to the Gaussian Process both reaching a different minima. This can be something to look into deeper to understand the reason for this variance and whether it is a problem and that the reason why the model works so well has a little to do with random chance as well. The figures can be located A.2.1

Finally, the combined CNN and LSTM model proved to be all around the best and most trustworthy model, having the lowest total NRMSE of three of the four categories (Diurnal Temperature Range, Precipitation and 90th Precipitation). In comparison to the results found in the original ClimateBench paper, there is a small amount of variation due to the randomness inherent in training a CNN without a preset seed, but overall the difference is small, with the highest being 2.215 for the Durnal Temperature Range. The CNN then, was able to achieve the about the same amount of trustworthiness as that of the one Professor Duncan trained in ClimateBench.

5.3 Future Research

For future research, we would like to explore more possibilities for creating a better Baseline for climate data prediction through machine learning modeling. The current models are just re-creating three models from Duncan's paper. In order to have more accurate predictions from our current models, we would like to optimize our models.

For example, the Neural Networks can be improved by incorporating the original ssp245 data that we compare against into the training data. While this was first avoided due to fears of overfitting, we believe that ignoring it all together doesn't make too much sense considering we are trying to predict that scenario. To combat overfitting, dropout can be incorporated when we add the extra data, which we believe could improve the model's performance overall. Different layers and different activation functions could also be beneficial, so further study could be derived from trying multiple different architectures.

The Gaussian Process model's performance can be enhanced by more carefully optimizing the selection of kernels. Tailoring kernel choices to fit each individual input variable can improve the model's fit to the training data, thereby enhancing the confidence intervals and the overall predictive accuracy of the model. Additionally, experimenting with different 'mean_functions' and optimizer could yield varied training outcomes, potentially leading to further improvements in the model.

The Random Forest models can be improved by XGBoost techniques. XGBoost can correct the errors in current models and add models sequentially to improve further. The more advanced model structures, hyperparameter tuning, and other improvements in techniques might allow us to get better NRSME scores than the Baseline Models and explore more climate input variables and output variables. Through that, the optimized models might be able to create better baselines to estimate the future climate and make the public understand what we should do to control climate change.

6 Conclusion

To create a Climate Bench, the machine learning models aim to predict various climate data by giving aerosol emission input data. Through the predictions, the policymakers and the general public are able to assess which emission pathway is most suitable for achieving the goal of reducing global warming. By applying the three machine learning models from the Climate Bench paper, we can predict target variables like temperature, diurnal temperature range, and precipitation. The NRMSE scores for our prediction are varied by different models and target variables. Also, the scores are slightly different from the scores of the original Climate Bench paper. There could be multiple factors contributing to these variations, such as input variables, the model itself, and target variables. For example, all of our three models perform not so well for the diurnal temperature range variable. So, the next step for the research should dive deeper into the model optimization and try to create a better baseline for future climate prediction.

References

Görtler, Jochen, Rebecca Kehlbeck, and Oliver Deussen. 2019. "A visual exploration of gaussian processes." *Distill* 4(4), p. e17

Watson-Parris, Duncan, Yuhan Rao, Dirk Olivié, Øyvind Seland, Peer Nowack, Gustau Camps-Valls, Philip Stier, Shahine Bouabid, Maura Dewey, Emilie Fons et al. 2022. "ClimateBench v1. 0: A Benchmark for Data-Driven Climate Projections." *Journal of Advances in Modeling Earth Systems* 14(10), p. e2021MS002954

Appendices

A.1 Training Details

A.1.1 Gaussian Process

For the reproduction of Duncan's ClimateBench - (Watson-Parris et al. 2022), we use the Gaussian Process Regression from the package `GPflow`, and for the generation of the kernel, we used a Matérn32 kernel from `GPflow` where the documentation can be found here. Next for each output variable I created a basic Matérn32 kernels for each of the 4 aerosols and added them together to create the final kernel which can be seen below,

```
from gpflow.kernels import Matern32

# Set up Kernels for each predictor using matern32 kernel
kernel_CO2 = Matern32(active_dims=[0],\
    variance=1.0, lengthscales=1.0)
kernel_CH4 = Matern32(active_dims=[1],\
    variance=1.0, lengthscales=1.0)
kernel_BC = Matern32(active_dims=[2, 3, 4, 5, 6],\
    variance=1.0, lengthscales=5 * [1.])
kernel_SO2 = Matern32(active_dims=[7, 8, 9, 10, 11],\
    variance=1.0, lengthscales=5 * [1.])

kernel_matern32 = kernel_CO2 + kernel_CH4 + kernel_BC + kernel_SO2
```

Then next is to minimize the loss via a mean function and the one that I used was the `GPflow.mean_functions.Constant()`, and then I trained the model allowing it only a max iteration of 1000 to find it's minima.

```
from gpflow.mean_functions import Constant
from gpflow.models import GPR
from gpflow.optimizers import Scipy

# Setting up model
```

```
mean_function = Constant()
model = GPR(
    data = (X_train.astype(float), y_train_tas.astype(float)),
    kernel = kernel_matern32,
    mean_function = mean_function,
)

# Define models and optimizers
optimizer = Scipy()
# Train model
optimizer.minimize(model.training_loss,
             variables=model.trainable_variables,
             options=dict(disp=True, maxiter=1000))
```

And below we have Figure A 1 which shows the training parameters from the Gaussian Process for **Temperature**

<gpflow.models.gpr.GPR object at 0x00000210EC152A30>

name	class	transform	prior	trainable	shape	dtype	value
GPR.mean_function.c	Parameter	Identity		True	(1,)	float64	[0.62667]
GPR.kernel.kernels[0].variance	Parameter	Softplus		True	()	float64	8.72645
GPR.kernel.kernels[0].lengthscales	Parameter	Softplus		True	()	float64	8.55292
GPR.kernel.kernels[1].variance	Parameter	Softplus		True	()	float64	0.03471
GPR.kernel.kernels[1].lengthscales	Parameter	Softplus		True	()	float64	0.55315
GPR.kernel.kernels[2].variance	Parameter	Softplus		True	()	float64	0.08214
GPR.kernel.kernels[2].lengthscales	Parameter	Softplus		True	(5,)	float64	[0.01929, 16.47391, 0.27919...
GPR.kernel.kernels[3].variance	Parameter	Softplus		True	()	float64	0.62918
GPR.kernel.kernels[3].lengthscales	Parameter	Softplus		True	(5,)	float64	[9.79738, 21.18801, 32.72901...
GPR.likelihood.variance	Parameter	Softplus + Shift		True	()	float64	0.07134

Figure A 1: TAS Gaussian Process Model

A.1.2 Random Forest

To reproduce the Random Forest from ClimateBench(Watson-Parris et al. 2022), we applied the Random Forest Regressor from the scikit-learn. To initialize the Randomized Search from scikit-learn package, we have set up a random gird as hyperparameters as following:

```
n_estimators = [
    int(x) for x in np.linspace(start = 100, stop = 300, num = 5)
]

# Number of features to consider at every split
max_features = ['auto', 'sqrt']
```

```
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(5,55, num = 11)]
max_depth.append(None)

# Minimum number of samples required to split a node
min_samples_split = [5, 10, 15, 25]

# Minimum number of samples required at each leaf node
min_samples_leaf = [4, 8, 12]

# Method of selecting samples for training each tree
bootstrap = [True, False]

# Create the random grid
random_grid = {
            'n_estimators': n_estimators,
            'max_features': max_features,
            'max_depth': max_depth,
            'min_samples_split': min_samples_split,
            'min_samples_leaf': min_samples_leaf,
            'bootstrap': bootstrap
            }
```

After initializing the grid, we can fit the training data into four different random forests by output variables. Each model should have the same hyperparameters for Randomized Search.

```
#fitting the random forest for tas
reg0 = RandomForestRegressor(random_state=0)
rf_tas = reg0.fit(X_train_tas,y_train_tas)
rf_random0 = RandomizedSearchCV(estimator = reg0,
param_distributions = random_grid, n_iter = 29, cv = 3, verbose=2, n_jobs = -1)
rf_tas = rf_random0.fit(X_train_tas,y_train_tas)

#fitting the random forest for pr
reg1 = RandomForestRegressor(random_state=0)
rf_pr = reg0.fit(X_train_pr,y_train_pr)
rf_random1 = RandomizedSearchCV(estimator = reg1,
param_distributions = random_grid, n_iter = 29, cv = 3, verbose=2, n_jobs = -1)
rf_pr = rf_random1.fit(X_train_pr,y_train_pr)
```

```
#fitting the random forest for pr90
reg2 = RandomForestRegressor(random_state=0)
rf_pr90 = reg0.fit(X_train_pr90,y_train_pr90)
rf_random2 = RandomizedSearchCV(estimator = reg2,
param_distributions = random_grid, n_iter = 29, cv = 3, verbose=2, n_jobs = -1)
rf_pr90 = rf_random2.fit(X_train_pr90,y_train_pr90)

#fitting the random forest for dtr
reg3 = RandomForestRegressor(random_state=0)
rf_dtr = reg3.fit(X_train_dtr,y_train_dtr)
rf_random3 = RandomizedSearchCV(estimator = reg3,
param_distributions = random_grid, n_iter = 29, cv = 3, verbose=2, n_jobs = -1)
rf_dtr = rf_random3.fit(X_train_dtr,y_train_dtr)
```

A.1.3 Convoluted Neural Network

To reproduce the CNN and LSTM deep learning model from ClimateBench(Watson-Parris et al. 2022), we made use of the machine learning libraries Tensorflow and Keras. We used a convolutional layer with ReLU activation, followed by both an average and global average layer, then the LSTM layer with ReLU activation before reshaping the final outputs with a Dense and Reshape layer. The structure is pictured below.

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 time_distributed (TimeDist  (None, 10, 96, 144, 20)   740
 ributed)
 time_distributed_1 (TimeDi  (None, 10, 48, 72, 20)    0
 stributed)
 time_distributed_2 (TimeDi  (None, 10, 20)            0
 stributed)
 lstm (LSTM)                 (None, 25)                4600
 dense (Dense)               (None, 13824)             359424
 activation (Activation)     (None, 13824)             0
 reshape (Reshape)           (None, 1, 96, 144)        0
=================================================================
Total params: 364764 (1.39 MB)
Trainable params: 364764 (1.39 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

A.2 Additional Figures

A.2.1 ClimateBench GP vs Our GP

Here are the figures that represent the interesting validation result that I (Jack) got when validation my Gaussian Processes **Temperature Predictions**.

If you close in on the Gaussian Process **temperature** for ClimateBench and our Gaussian Process you can see a clear difference bewteen the predicted values which is extremely interesting.

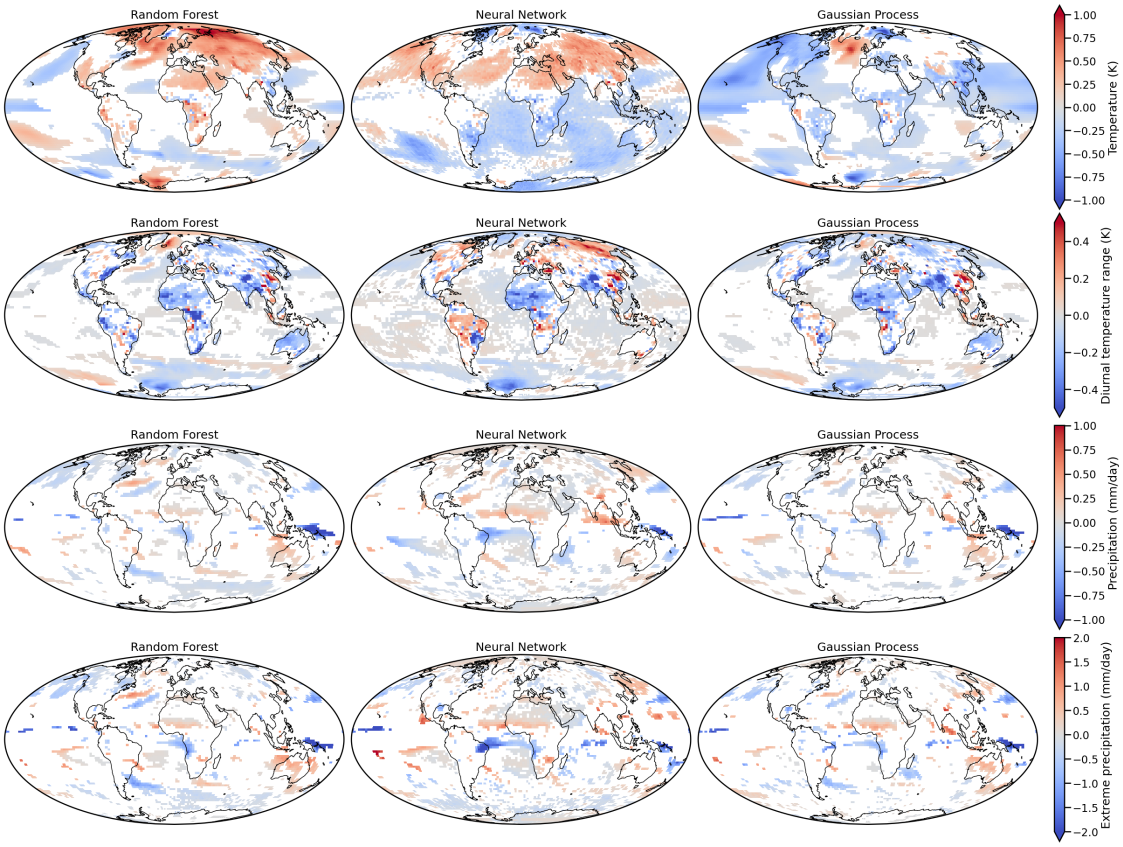


Figure A 2: Figure from ClimateBench showing the mean difference between the models against the **ssp245** emulators

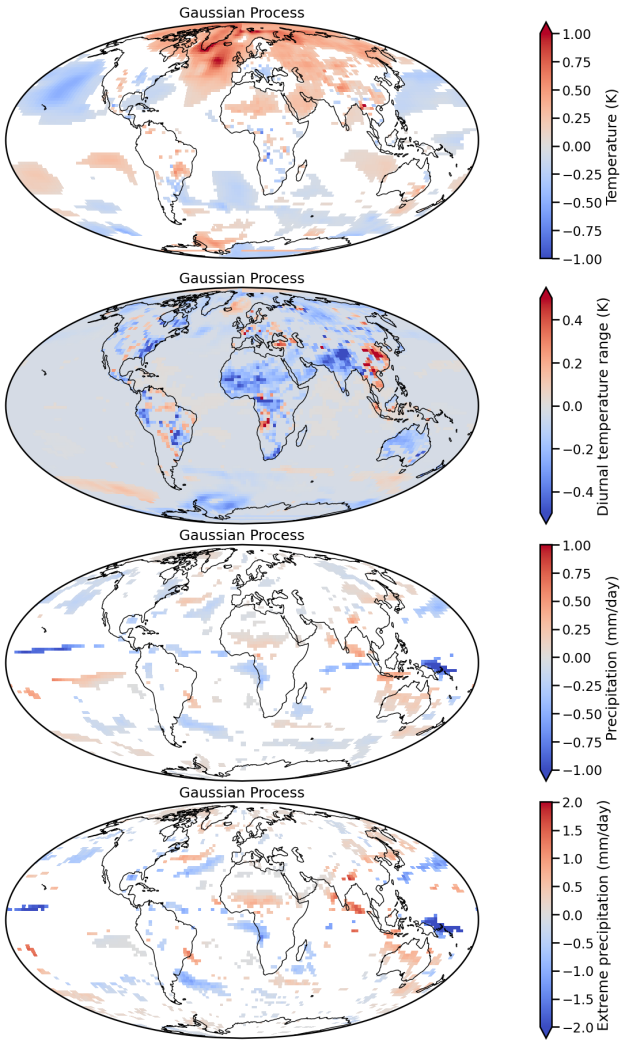


Figure A 3: Figure from Jack's Gaussian Process Mean difference against **ssp245** emulators