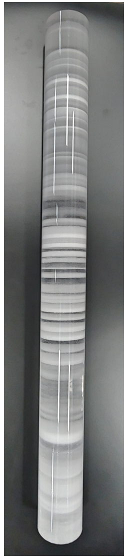
Data Science for Polar Ice-Core Climate Reconstructions

Team: Daniel Cairns, Jakob Thoms, Shirley Zhang (MDS, UBC)
Partner: Anais Orsi (EOAS department, Faculty of Science, UBC)
Mentor: Alexi Rodríguez-Arelis (MDS, UBC)

Challenge

How can we use data from ice cores to predict what the weather was like in Antarctica in the past?



?

Weather conditions

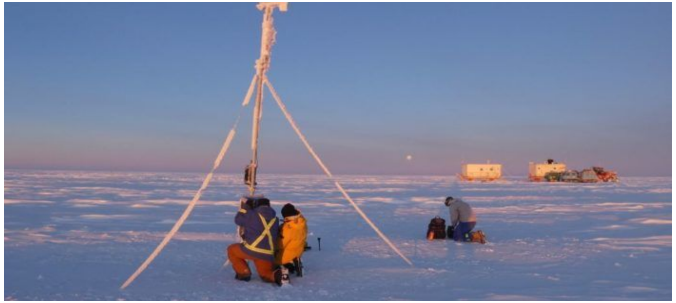


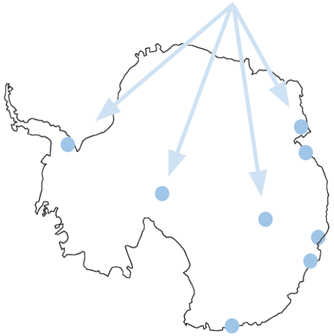
Image: Craig Butsch

Image: Pixabay

Contents

Data

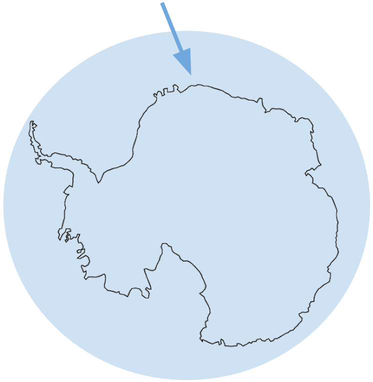
Observed ice-core and weather data



+
Mathematical weather relationships

Computationally intensive simulations

Artificial ice-core and weather data



Climate model

Training and testing data

Machine learning/ statistical model

X: simulated ice-core data

Y: simulated weather data

Surrogate model

A **simplified** version of a more **complex** and **computationally intensive** model.

Delta-18-O and Climate Variables

Ice-Core Data (Model inputs)

Delta-18-O

H_2O^{18}

H_2O^{16}

Weather Data (Model Outputs)

Temperature

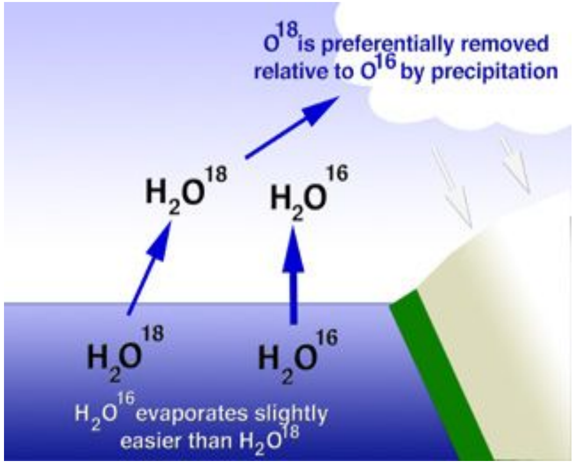
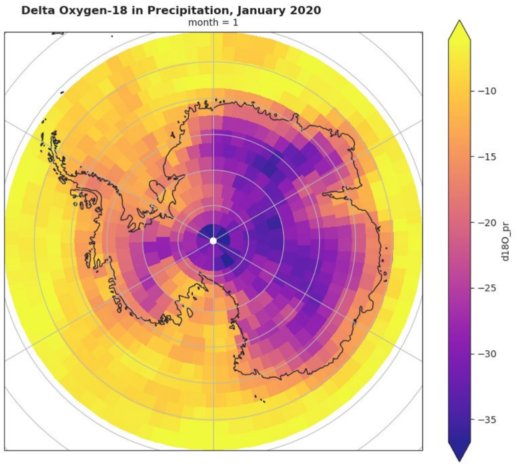


Image: University of Michigan

Geopotential Height and Precipitation



"Can we build **surrogate models** using simulated climate data which yield *accurate* and *precise predictions* of **temperature, geopotential height, and precipitation** across Antarctica?"

<u>Objectives</u>

1. Implement Gaussian Process and Neural Network models ✅
2. Evaluate accuracy and precision of predictions ✅
3. Create a workflow notebook and reproducible package for others to use ✅

Project Overview

1
Preprocessing

2
Modelling

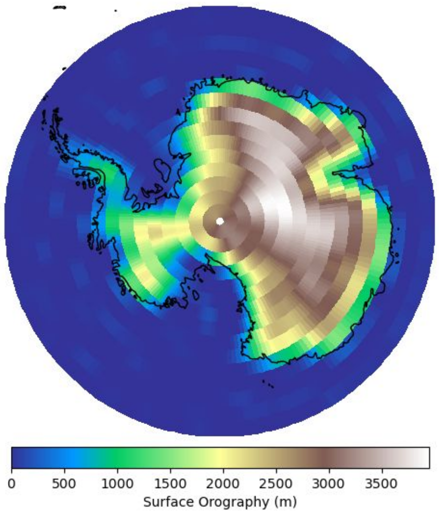
- OLS Baseline
- Gaussian Process (GP)
- Neural Network (NN)

3
Evaluation

4
Data Products

Preprocessing: Split Data and Add Spatial Features

- Polar Coordinates
- Distance to Coast
- Surface Orography (elevation)
- Land Boolean Mask



Preprocessing: Deseasonalize and Scale

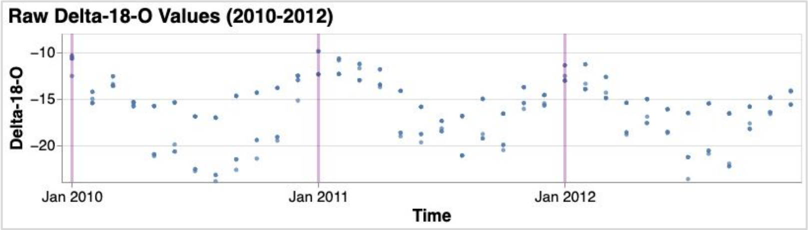
Let v be a spatial-temporal variable (e.g. Delta 18-O).

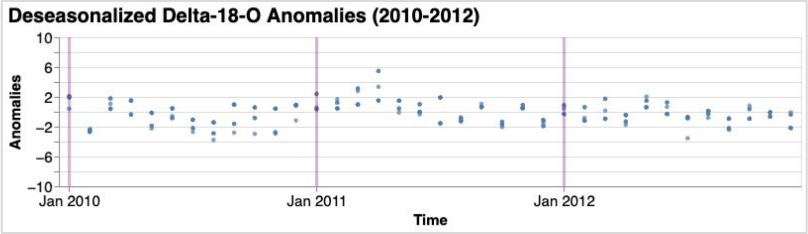
Deseasonalize

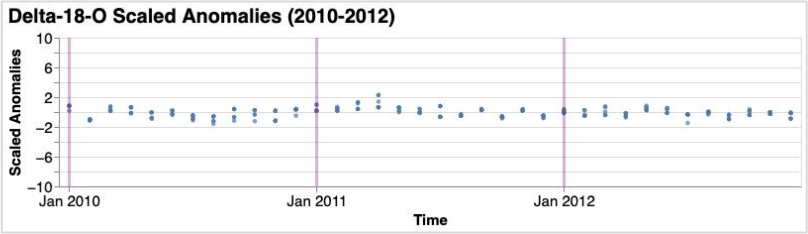
$$v^{(\text{anomaly})}_{\text{lat,lon,month}} = v_{\text{lat,lon,month}} - \overline{v}_{\text{lat,lon,month}}$$

Scale

$$v^{(\text{anomaly, scaled})}_{\text{lat,lon,month}} = \frac{v^{(\text{anomaly})}_{\text{lat,lon,month}} - \overline{v}^{(\text{anomaly})}}{\hat{\sigma}^{(\text{anomaly})}}$$







Preprocessing

Model inputs	Model outputs
Scaled Delta 18-O Anomalies	Scaled Temperature Anomalies
Scaled Easting Coordinate	Scaled Geopotential Height Anomalies
Scaled Northing Coordinate	Scaled Precipitation Anomalies
Scaled Distance to Coast	
Scaled Surface Orography	
Land / Sea Boolean Mask	

GP Model Architecture

$$y_i = \mu(\mathbf{x}_i) + \text{ error term (stochastic)}$$

$$\mu(\mathbf{x}_i) = \beta_0 + \sum_{k=1}^{5} \beta_k x_{i,k}$$

$$\mathbf{x}_i = (x_{i,1}\ x_{i,2}\ x_{i,3}\ x_{i,4}\ x_{i,5})^\mathsf{T}$$

$$x_{i,1} = \text{delta-18O}$$
$$x_{i,2} = \text{UPS Easting}$$
$$x_{i,3} = \text{UPS Northing}$$
$$x_{i,4} = \text{Distance-to-coast}$$
$$x_{i,5} = \text{Surface Altitude}$$

$$i = 1, \ldots, n$$
$$n = \text{number of training examples}$$

$$\mathrm{Var}(Z(\mathbf{x}_i)) = \sigma^2$$
$$\mathrm{Cov}(Z(\mathbf{x}_i), Z(\mathbf{x}_j)) = \sigma^2 R(\mathbf{x}_i, \mathbf{x}_j)$$

$$R(\mathbf{x}_i, \mathbf{x}_j) = \prod_{k=1}^{5} \mathcal{K}_{\mathrm{RBF}}(x_{i,k},\ x_{j,k})$$

GP Training

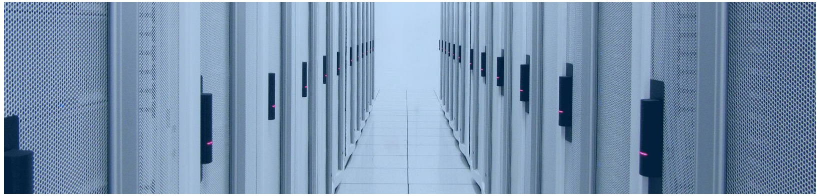


Image: UBC ARC Sockeye

Training data

> 700,000 examples

Split #	Examples	Time Range	Time	Memory	Train RMSE
1	~ 87,000	Jan 1995 - Nov 1997	2 hours	150 GB	0.967
2	~ 87,000	Dec 1997 - Oct 2000	1.5 hours	150 GB	0.846
…	…		…	…	…
8	~ 87,000	Mar 2015 - Jan 2018	1.5 hours	150 GB	0.867
9	~ 87,000	Feb 2018 - Dec 2020	2 hours	150 GB	0.821

Final Model Parameters

Variable: Temperature
Kernel: RBF Kernel
Epochs: 10
Learning rate: 0.0015

NN Architecture: Six 2D-convolutional layers

***6 Inputs:** δ-Oxygen-18, Easting, Northing, Altitude, Coastal Distance, Land Mask*

Layer	Input Channels	Output Channels
1	6	32
2	32	32
3	32	16
4	16	16
5	16	8
6	8	3

***3 Outputs**: Temperature, Precipitation, Geopotential Height*

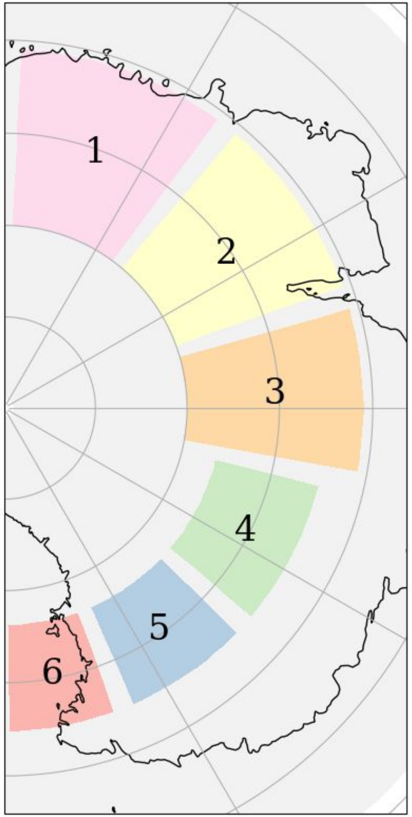
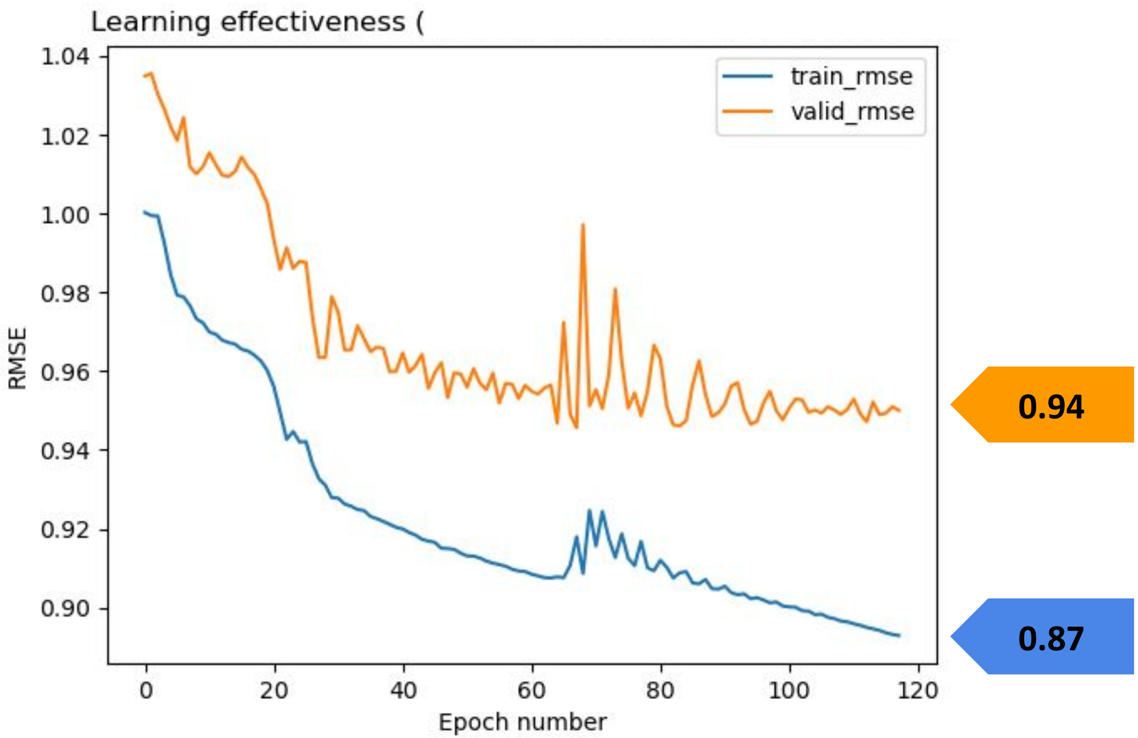


Illustration of each layer's kernel size at latitude -75.

NN Training



117
Epochs

26
Minutes

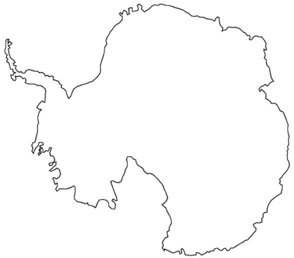
+10% vs. Baseline

Model Evaluation

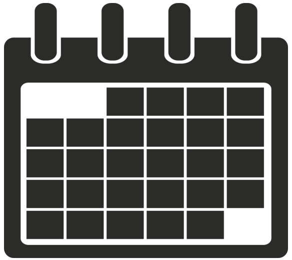
1. Calculate RMSE

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}\left(y_i^{(\text{anomaly, scaled})} - \hat{y}_i^{(\text{anomaly, scaled})}\right)^2}{n}}$$

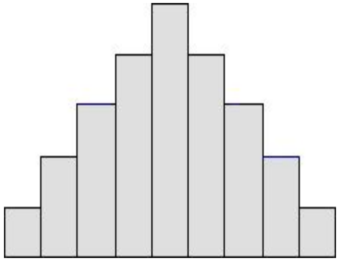
2. Analyze Residuals



Spatial Patterns



Temporal Patterns



Residual Distribution

Precipitation

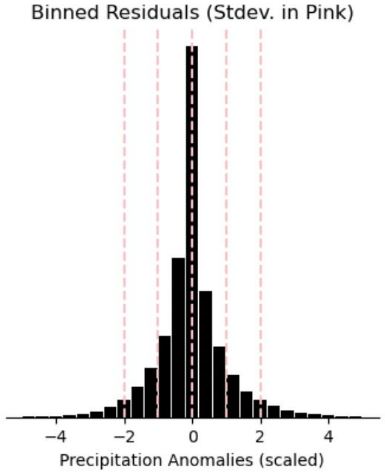
<u>OLS Baseline</u>
1.07

<u>GP Model</u>
1.04

<u>NN Model</u>
1.05

Test RMSE scores
on scaled anomalies

Model Residual Analysis - Gaussian Process (kernel 1) with Split 6/9
Precipitation Anomalies (scaled)



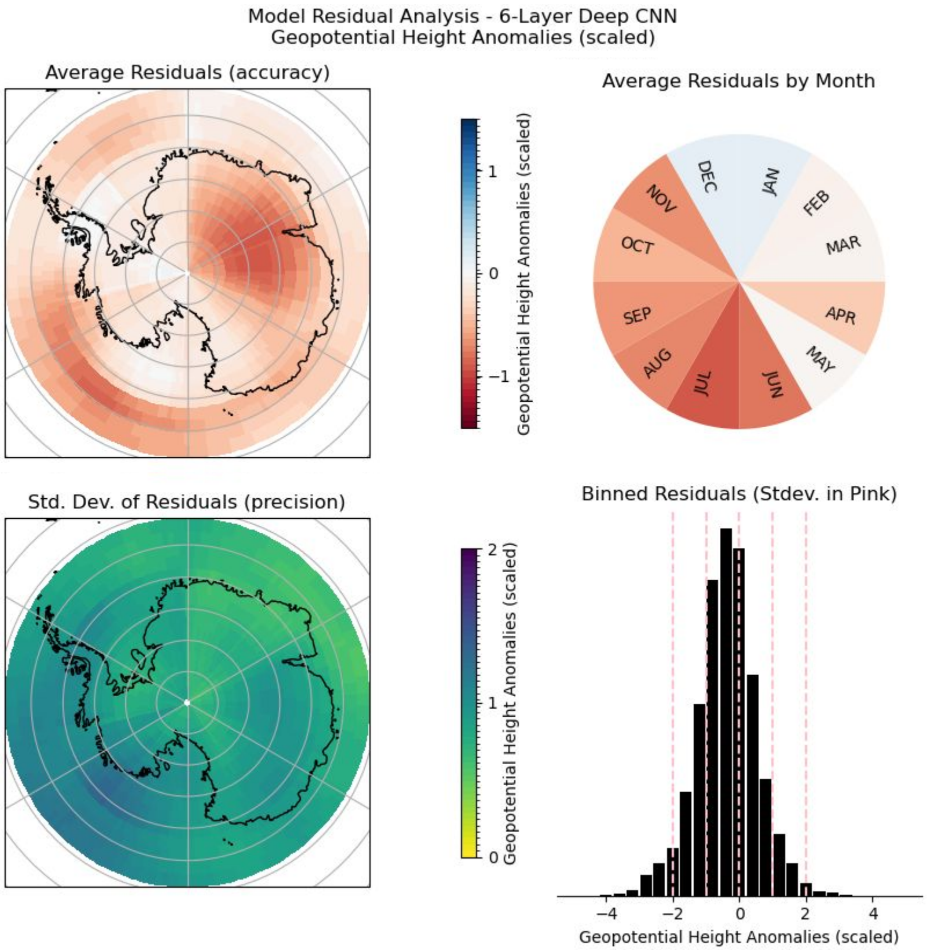
Geopotential Height

OLS Baseline
1.11

GP Model
1.04

NN Model
0.97

Test RMSE scores on scaled anomalies



Temperature

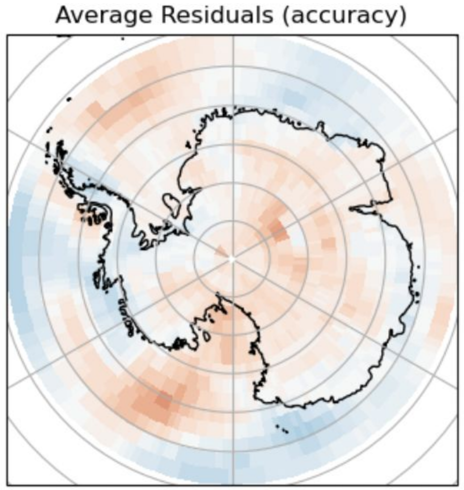
OLS Baseline
1.11

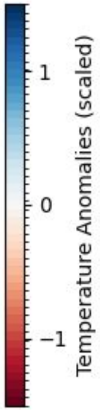
GP Model
1.03

NN Model
0.98

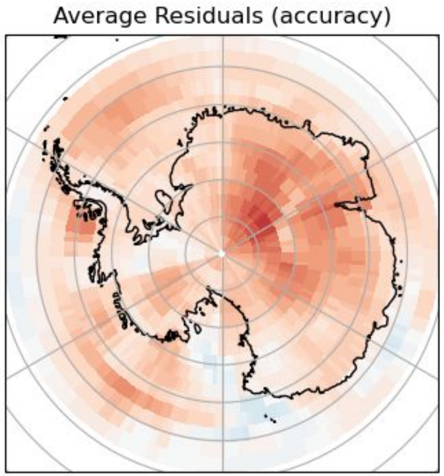
Test RMSE scores on scaled anomalies

GP Model





NN Model



1 Background

2 Research Question and Objectives

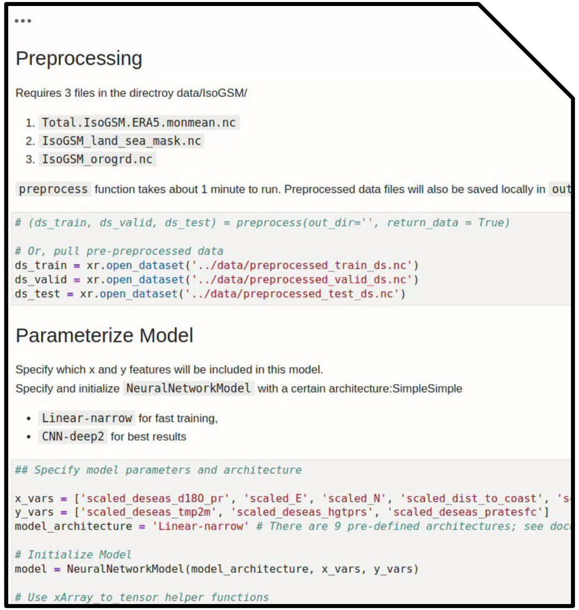
3 Preprocessing

4 Modelling

5 Results and Discussion

6 Future Steps

Data Product: Notebook and GitHub Repo



Preprocessing

Requires 3 files in the directroy data/IsoGSM/

1. `Total.IsoGSM.ERA5.monmean.nc`
2. `IsoGSM_land_sea_mask.nc`
3. `IsoGSM_orogrd.nc`

`preprocess` function takes about 1 minute to run. Preprocessed data files will also be saved locally in `out`

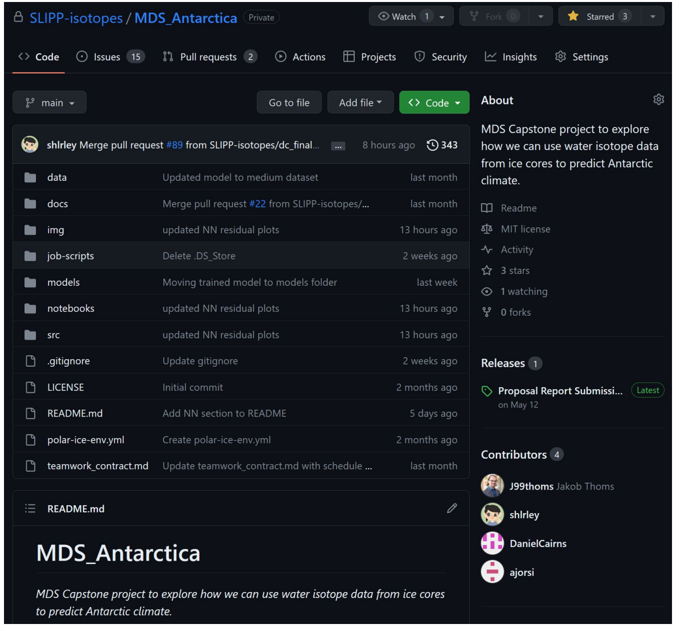
```
# (ds_train, ds_valid, ds_test) = preprocess(out_dir='', return_data = True)

# Or, pull pre-preprocessed data
ds_train = xr.open_dataset('../data/preprocessed_train_ds.nc')
ds_valid = xr.open_dataset('../data/preprocessed_valid_ds.nc')
ds_test = xr.open_dataset('../data/preprocessed_test_ds.nc')
```

Parameterize Model

Specify which x and y features will be included in this model.
Specify and initialize `NeuralNetworkModel` with a certain architecture:SimpleSimple

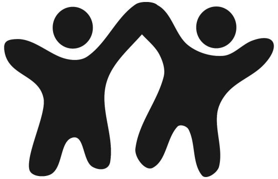
- `Linear-narrow` for fast training,
- `CNN-deep2` for best results

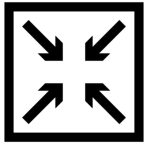
```
## Specify model parameters and architecture

x_vars = ['scaled_deseas_d18O_pr', 'scaled_E', 'scaled_N', 'scaled_dist_to_coast', 's
y_vars = ['scaled_deseas_tmp2m', 'scaled_deseas_hgtprs', 'scaled_deseas_pratesfc']
model_architecture = 'Linear-narrow' # There are 9 pre-defined architectures; see doc

# Initialize Model
model = NeuralNetworkModel(model_architecture, x_vars, y_vars)

# Use xArray to tensor helper functions
```



Future Steps

Ensemble model





Local scaling

Different climate models

Challenge

How can we use data from ice-cores to predict what the weather was like in Antarctica in the past?

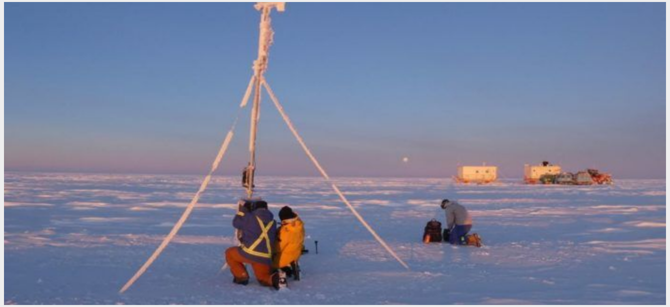


Image: Craig Butsch

Questions?