GCAM Annual Meeting 2023 - `stitches` training

The purpose of this tutorial is to demonstrate how `stitches` can be used as an emulator. While `stitches` can emulate a number of CMIP6 models, this example will focus on emulating CanESM5 SSP245 results.

This tutorial also assumes that the user has either seen a talk on `stitches` or read the paper published in *Earth System Dyanmics* (Tebaldi et al 2022). This tutorial is aimed at highlighting the flexibility of functions in `stitches`.

A simpler quickstart notebook comes in every stitches download and includes installation instructions:

https://github.com/JGCRI/stitches/blob/main/notebooks/stitches-quickstart.ipynb

A more detailed and involved version of this training is available as well:

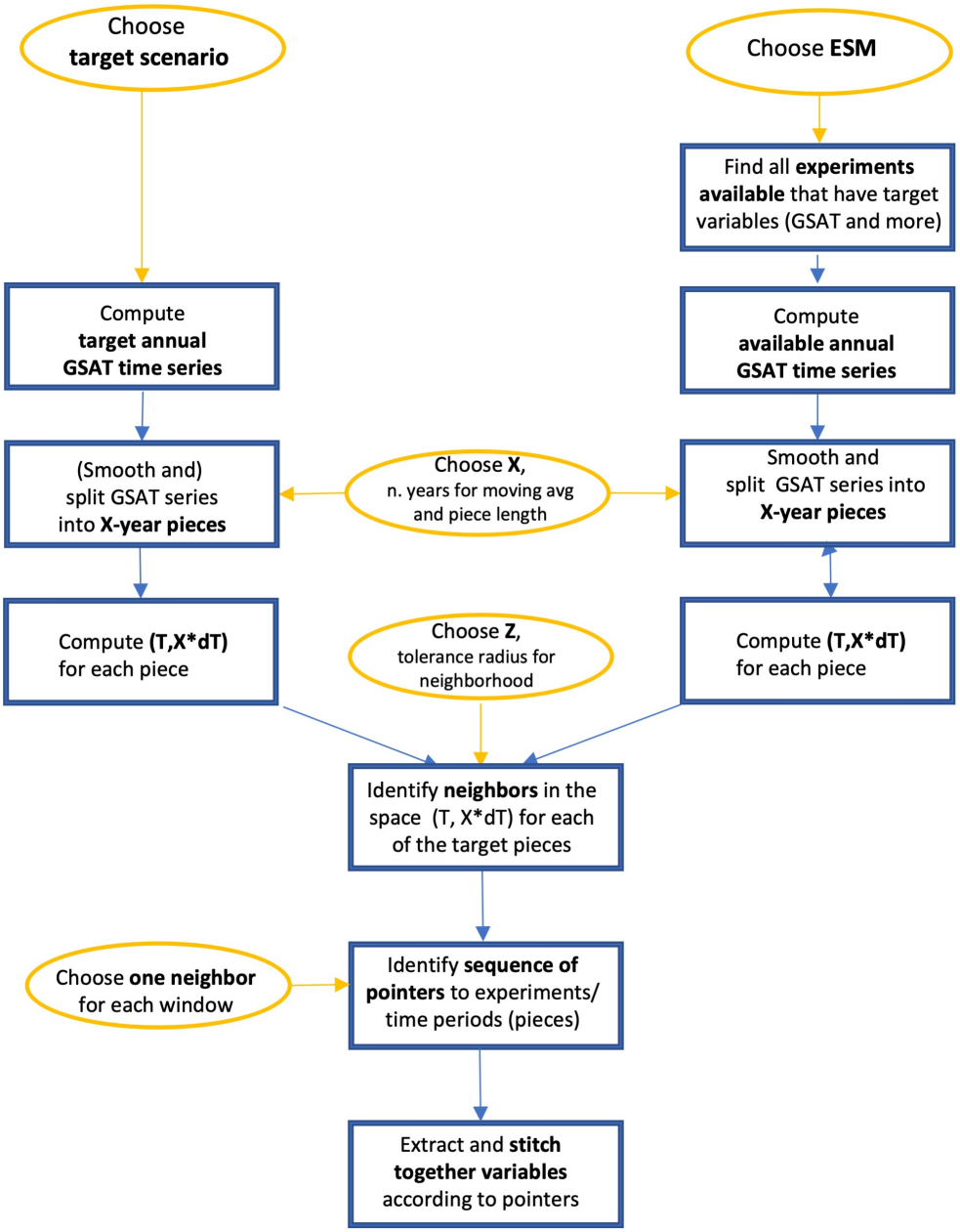
https://github.com/JGCRI/stitches/blob/dev/notebooks/stitches_takehome_GCAMAnnualMeeting2023.ipynb

All notebooks assume a familiarity with CMIP-style data.

To use `stitches`, there are a number of decisions users have to make, perhaps the most important being:

- Which ESM will `stitches` emulate?
- What *archive data* will be used? These are values of global temperature (in the following referred to as GSAT, or Tgav, or tas) for experiments/time periods (e.g., historical and SSP realizations) that the target data will be matched to. It should only contain data for the specific ESM that is being emulated.
- What *target data* will be used? This data frame represents the temperature pathway the stitched product will follow. The contents of this data frame may come from computing GSAT from CMIP6 ESM temperature output (as we do in this tutorial) according to a standard SSP scenario, or it may follow some alternate pathway that a simple model like Hector or FaIR has produced.

A diagram illustrating the `stitches` process is included for reference:

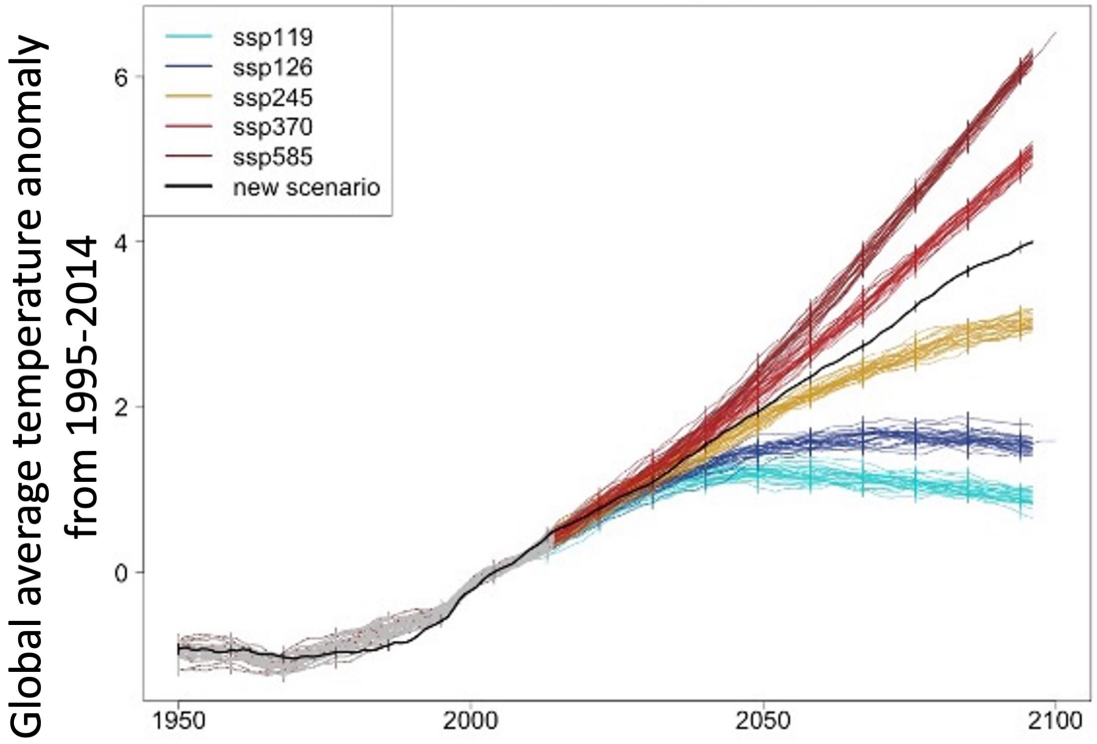


- `stitches` defaults to $X = 9$ year windows.

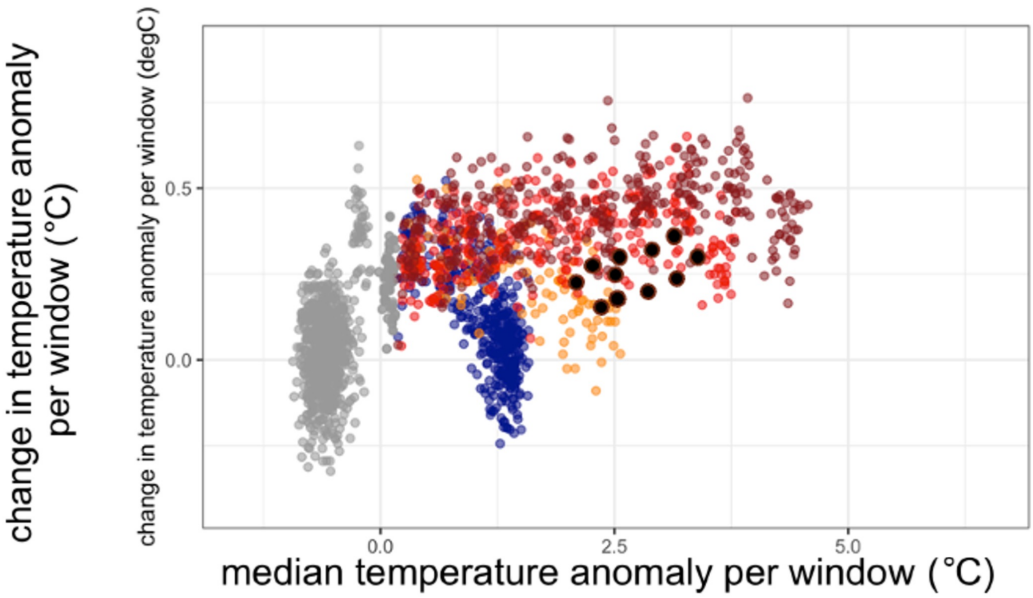
STITCHES works by matching X-year segments of a target GSAT trajectory to X-year segments of available GSAT trajectories, computed from the archived output of an ESM that has been run according to SSP scenarios.

The choice of X balances the need to maintain temporal consistency with the need to have flexibility in matching GSAT anomalies and rates of change. STITCHES uses X=9 year windows.

So, if one were to take ESM data for many scenarios and plot the corresponding global average temperature trajectories after smoothing, the vertical lines drawn here represents a potential segmenting of the data into those 9-year windows of both available simulations (colored lines) and a target trajectory (the back line, here as an example, the new trajectory intermediate to those available).



For each segment, the median temperature value and the change in value per segment can be plotted in a two-dimensional space where now the windows of the available scenarios are the colored dots, and those of the target scenario are the black dots:



This two dimensional space is where matching between target points (black) and available archive points (colorful) occurs, using a nearest neighbor approach.

Getting Started

```
import stitches
```

Load the additional python libraries that will be used in this example. These packages are installed as `stitches` dependencies.

```
import os
import pkg_resources
import warnings

import numpy as np
import pandas as pd
from matplotlib import pyplot as plt



# For help with plotting
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
plt.rcParams['figure.figsize'] = 12, 6
```

Install the package data from Zenodo

The package data is all data that has been processed from raw Pangeo data and is generated with package functions. For convenience and rapid cloning of the github repository, the package data is also minted on Zenodo and can be quickly downloaded for using the package.

For this training, it has been installed for you already (5-10 min to download).

```
# stitches.install_package_data()
```

Example Set Up

We will use `stitches` to emulate CanESM5 SSP245 results (an experiment run under CMIP6). This is our *Target Data*.

Then we will compare the `stitches` results with actual CanESM5 SSP245 output data.

For CMIP6 results, Earth system model data runs from 1850-2100 (or 2099, depending on the ESM). This tutorial will focus on emulating that entire period.

Decide on the target data

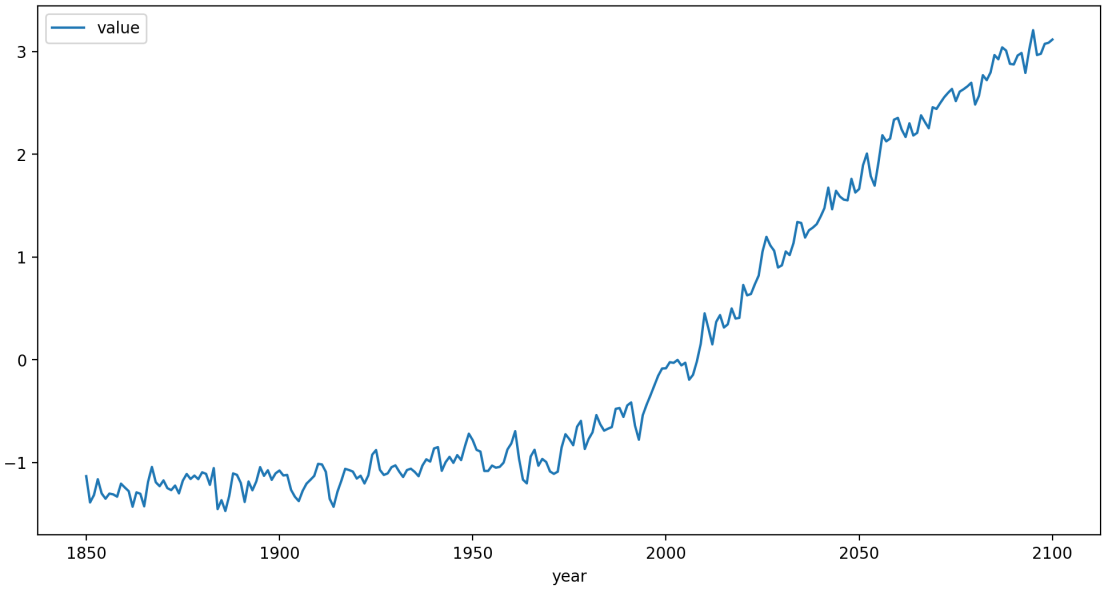
- The primary input to `stitches` functions that most users will adjust is the target data.
- The target data is the temperature pathway the stitched (emulated) product will follow. This data can come from an ESM or another class of climate models, for a specific SSP scenario or an arbitrarily defined scenario.
- Similarly to the archive data, the target data should contain the mean temperature anomaly and rate of temperature change for every X-year window into which the target GSAT trajectory has been subdivided. `stitches` includes functions for processing raw ESM Tgav data into the structure it needs for matching.

```
# Load time series and subset to target time series if needed:
data_directory = pkg_resources.resource_filename('stitches', "data")
targ = pd.read_csv(os.path.join(data_directory, "tas-data",
"CanESM5_tas.csv"))
target_data = targ.loc[(targ["model"] == "CanESM5")
                       & (targ["experiment"] == 'ssp245')].copy()

target_data  = target_data[target_data["ensemble"].isin(['r1i1p1f1'])].copy()

target_data = target_data.drop(columns='zstore').reset_index(drop=True)
```

Take a look at the structure and a plot of the time series we will be targeting:

```
print(target_data.head())
target_data.plot(x='year', y='value')
plt.show()
plt.close()
```

```
  variable experiment  ensemble    model  year     value
0      tas     ssp245  r1i1p1f1  CanESM5  1850 -1.133884
1      tas     ssp245  r1i1p1f1  CanESM5  1851 -1.389375
2      tas     ssp245  r1i1p1f1  CanESM5  1852 -1.318175
3      tas     ssp245  r1i1p1f1  CanESM5  1853 -1.163771
4      tas     ssp245  r1i1p1f1  CanESM5  1854 -1.302066
```



- Critically, these time series are *global average temperature anomaly from 1995-2014 average*.
- In this demonstration, we will specifically be targeting ensemble member 1 of the CanESM5 SSP245 simulations. The entire SSP245 ensemble may be jointly targeted by omitting the line `target_data = target_data[target_data["ensemble"].isin(['r1i1p1f1'])].copy()`

Decide on the archive data

- Limit the archive matching data to the model we are trying to emulate, CanESM5 in this case.
- In this example, we treat SSP245 as a novel scenario rather than one run by the ESM and available, so we exclude it from the archive data.
- `stitches` actually provides two files in its pacakge data.

- `matching_archive.csv` can be considered the default (for now). Starting in 1850, nine year windows are sliced forward and don't overlap.
- The final window ends up beginning in 2093, and is only 8 years long to terminate in 2100 (7 years if the ESM ends in 2099).

This training will demonstrate the more flexible archive setting options.

- `stitches` includes `matching_archive_staggered.csv` as package data as well. The difference is that this file does every possible full 9-year chunk, not just slicing sequentially from the starting point.

```
# read in the package data of all ESMs-Scenarios-ensemble members avail.
data_directory = pkg_resources.resource_filename('stitches', "data")
path = os.path.join(data_directory, 'matching_archive_staggered.csv')
data = pd.read_csv(path)

staggered_archive = data.copy()

end_yr_vector = [1858, 1866, 1875,1884, 1893,
                 1902, 1911, 1920, 1929, 1938, 1947,
                 1956,  1965, 1974, 1983, 1992, 2001,
                 2010, 2019, 2028, 2037, 2046, 2055,
                 2064, 2073, 2082, 2091, 2100]

tmp = staggered_archive.loc[(data["experiment"].isin(['ssp126', 'ssp370',
'ssp585']))
                        & (data["model"] == "CanESM5")].copy()

archive_data = stitches.fx_processing.subset_archive(staggered_archive = tmp,
                              end_yr_vector = end_yr_vector)

print(archive_data)
```

```
     experiment variable    model   ensemble  start_yr  end_yr  year  \
0        ssp126      tas  CanESM5  r10i1p1f1      1850    1858  1854
1        ssp126      tas  CanESM5  r11i1p1f1      1850    1858  1854
2        ssp126      tas  CanESM5  r12i1p1f1      1850    1858  1854
3        ssp126      tas  CanESM5  r13i1p1f1      1850    1858  1854
4        ssp126      tas  CanESM5  r14i1p1f1      1850    1858  1854
...         ...      ...      ...        ...       ...     ...   ...
2095     ssp585      tas  CanESM5   r9i1p1f1      2056    2064  2060
2096     ssp585      tas  CanESM5   r9i1p1f1      2065    2073  2069
2097     ssp585      tas  CanESM5   r9i1p1f1      2074    2082  2078
2098     ssp585      tas  CanESM5   r9i1p1f1      2083    2091  2087
2099     ssp585      tas  CanESM5   r9i1p1f1      2092    2100  2096

            fx        dx
0    -1.360399  0.016472
1    -1.235783 -0.013422
```

```
2    -1.378939  0.017703
3    -1.390443  0.014038
4    -1.401410  0.010122
...        ...       ...
2095  3.250968  0.080163
2096  4.020579  0.088755
2097  4.731989  0.076436
2098  5.508628  0.089533
2099  6.249420  0.058199

[2100 rows x 9 columns]
```

Target data pre-processing

- We had decided to target SSP245 realization 1 to emulate
- The first step of pre-processing any target data is to smooth it.

```
target_data = stitches.fx_processing.calculate_rolling_mean(target_data,
                                                            size=31).copy()
```

For consistency with how we set up the archive, we will have the target window ending in 2100 be a complete 9 years and work back.

You can use the `base_chunk=8` argument to do that. `base_chunk=8` means the target starts in 1850+8 = 1858 and cuts every 9 years after that, ending in 2100.

```
target_data = stitches.fx_processing.get_chunk_info(
    stitches.fx_processing.chunk_ts(df = target_data,  n=9,
                                    base_chunk=8)).copy()
print(target_data)
```

```
    ensemble experiment variable    model  start_yr  end_yr  year        fx
\
0   r1i1p1f1     ssp245      tas  CanESM5      1858    1866  1862 -1.254450
1   r1i1p1f1     ssp245      tas  CanESM5      1867    1875  1871 -1.242954
2   r1i1p1f1     ssp245      tas  CanESM5      1876    1884  1880 -1.214581
3   r1i1p1f1     ssp245      tas  CanESM5      1885    1893  1889 -1.195424
4   r1i1p1f1     ssp245      tas  CanESM5      1894    1902  1898 -1.200969
5   r1i1p1f1     ssp245      tas  CanESM5      1903    1911  1907 -1.172773
6   r1i1p1f1     ssp245      tas  CanESM5      1912    1920  1916 -1.145221
7   r1i1p1f1     ssp245      tas  CanESM5      1921    1929  1925 -1.091947
8   r1i1p1f1     ssp245      tas  CanESM5      1930    1938  1934 -1.022043
9   r1i1p1f1     ssp245      tas  CanESM5      1939    1947  1943 -0.994382
10  r1i1p1f1     ssp245      tas  CanESM5      1948    1956  1952 -0.955606
11  r1i1p1f1     ssp245      tas  CanESM5      1957    1965  1961 -0.945947
12  r1i1p1f1     ssp245      tas  CanESM5      1966    1974  1970 -0.879419
13  r1i1p1f1     ssp245      tas  CanESM5      1975    1983  1979 -0.761327
14  r1i1p1f1     ssp245      tas  CanESM5      1984    1992  1988 -0.506838
15  r1i1p1f1     ssp245      tas  CanESM5      1993    2001  1997 -0.268266
```

```
16  r1i1p1f1     ssp245      tas  CanESM5      2002    2010  2006  0.030355
17  r1i1p1f1     ssp245      tas  CanESM5      2011    2019  2015  0.420281
18  r1i1p1f1     ssp245      tas  CanESM5      2020    2028  2024  0.791804
19  r1i1p1f1     ssp245      tas  CanESM5      2029    2037  2033  1.149695
20  r1i1p1f1     ssp245      tas  CanESM5      2038    2046  2042  1.481969
21  r1i1p1f1     ssp245      tas  CanESM5      2047    2055  2051  1.819203
22  r1i1p1f1     ssp245      tas  CanESM5      2056    2064  2060  2.127866
23  r1i1p1f1     ssp245      tas  CanESM5      2065    2073  2069  2.402196
24  r1i1p1f1     ssp245      tas  CanESM5      2074    2082  2078  2.636278
25  r1i1p1f1     ssp245      tas  CanESM5      2083    2091  2087  2.832238
26  r1i1p1f1     ssp245      tas  CanESM5      2092    2100  2096  2.937192

          dx
0   0.003499
1   0.000194
2   0.004563
3  -0.001245
4   0.000351
5   0.004305
6   0.002159
7   0.005531
8   0.008648
9   0.003807
10  0.001086
11  0.006385
12  0.013736
13  0.016582
14  0.028356
15  0.032369
16  0.038950
17  0.037003
18  0.041818
19  0.039828
20  0.036774
21  0.034212
22  0.033031
23  0.030943
24  0.023368
25  0.012831
26  0.014289
```

Emulate

This occurs with two functions:

- `stitches.make_recipe()` does the matching between a target and archive, and gives the pointers to all of the pangeo-hosted netcdf files of data.

- `stitches.gmat_stitching()` or `stitches.gridded_stitching()` then stitch either global average temperature anomaly trajectories or gridded, multivariate netcdfs from those recipes.

Matching and making the recipe.

The arguments for making recipes are relatively simple. You specify the target data, the archive data, how many matches you want to try to make for each realization of the target data, and whether you want your results to be reproducible.

Two other optional variables include

- `non_tas_variables` - which variables in addition to tas do you think you want to have gridded results for? The default is to only provide tas recipes.
- `res` - do you want to stitch monthly (`'mon'`) or daily (`'day'`) gridded results? The default is to monthly as daily files are very large to work with and create.

The remaining argument, `tol` specifies the matching tolerance - for each target window, how far out away in the archive are we willing to look for similar points?

- `stitches` prioritizes providing a nearest neighbor match, which dictates how it currently uses `tol`
- `tol=0.0` corresponds to providing the nearest neighbor match.
- Each target window gets its own, custom nearest neighbor match that is some distance away, `dist_nn`.
- For each target window, we center a circular matching neighborhood on the target point. In a radius of `dist_nn`, we know the nearest neighbor is the only available point by definition.
- Therefore, we expand the matching neighborhood for each target point to search for matches up to a distance of `dist_nn + tol` away.
- So if target window A has a nearest neighbor 0.1degC away, and `tol=0.01`, then a circle centered on A with radius 0.11degC contains all possible matches. If target window B has a nearest neighbor 0.05degC away, then its matching neighborhood is a circle centered on B with a radius of 0.06degC.
- In the paper, we provide `z_cutoff` values that are 'safe' maximum tolerances to use for every ESM examined. We will be adding more ESMs in the future.

```
# the nearest neighbor recipes
nn_recipes = stitches.make_recipe(target_data = target_data,
                                  archive_data=archive_data,
                                  tol=0.0,
                                  N_matches=4,
                                  reproducible=True)
```

```
print(nn_recipes.head())
print(nn_recipes.tail())
```

```
You have requested more recipes than possible for at least one target
trajectories, returning what can
The following target windows have a nearest neighbor in T, dT space
that is more than 0.25degC away. This may or may not result in poor
matches and we recommend validation.
  target_variable target_experiment target_ensemble target_model  \
0             tas            ssp245        r1i1p1f1      CanESM5

   target_start_yr  target_end_yr  target_year  target_fx  target_dx  \
0             2083           2091         2087   2.832238   0.012831

  archive_experiment  ... archive_model archive_ensemble archive_start_yr  \
0             ssp370  ...       CanESM5        r17i1p1f1             2056

   archive_end_yr  archive_year  archive_fx  archive_dx   dist_dx   dist_fx
\
0            2064          2060    2.815717    0.045724  0.263149  0.016521

    dist_l2
0  0.263667

[1 rows x 21 columns]
--------------------------------------------------------------------------------
------------
  target_start_yr target_end_yr archive_experiment archive_variable  \
0            1858          1866         historical              tas
1            1867          1875         historical              tas
2            1876          1884         historical              tas
3            1885          1893         historical              tas
4            1894          1902         historical              tas

  archive_model archive_ensemble       stitching_id archive_start_yr  \
0       CanESM5         r3i1p1f1  ssp245~r1i1p1f1~1             1858
1       CanESM5        r19i1p1f1  ssp245~r1i1p1f1~1             1858
2       CanESM5         r1i1p1f1  ssp245~r1i1p1f1~1             1867
3       CanESM5        r17i1p1f1  ssp245~r1i1p1f1~1             1903
4       CanESM5         r5i1p1f1  ssp245~r1i1p1f1~1             1894

  archive_end_yr                                          tas_file
0           1866  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
1           1866  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
2           1875  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
3           1911  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
4           1902  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
   target_start_yr target_end_yr archive_experiment archive_variable  \
23            2056          2064             ssp370              tas
```

```
24            2065          2073             ssp370              tas
25            2074          2082             ssp370              tas
26            2083          2091             ssp370              tas
27            2092          2100             ssp370              tas

   archive_model archive_ensemble       stitching_id archive_start_yr  \
23       CanESM5         r4i1p1f1  ssp245~r1i1p1f1~1             2047
24       CanESM5        r20i1p1f1  ssp245~r1i1p1f1~1             2047
25       CanESM5        r11i1p1f1  ssp245~r1i1p1f1~1             2056
26       CanESM5        r17i1p1f1  ssp245~r1i1p1f1~1             2056
27       CanESM5         r2i1p1f1  ssp245~r1i1p1f1~1             2056

   archive_end_yr                                          tas_file
23           2055  gs://cmip6/CMIP6/ScenarioMIP/CCCma/CanESM5/ssp...
24           2055  gs://cmip6/CMIP6/ScenarioMIP/CCCma/CanESM5/ssp...
25           2064  gs://cmip6/CMIP6/ScenarioMIP/CCCma/CanESM5/ssp...
26           2064  gs://cmip6/CMIP6/ScenarioMIP/CCCma/CanESM5/ssp...
27           2064  gs://cmip6/CMIP6/ScenarioMIP/CCCma/CanESM5/ssp...
```

More flexible matches

```
# additional recipes
my_recipes = stitches.make_recipe(target_data = target_data,
                                  archive_data=archive_data,
                                  tol=0.05,
                                  res='mon',
                                   non_tas_variables=['pr'],
                                  N_matches=4,
                                  reproducible=True)
print(my_recipes.head())
print('-----------------------------------')
# you can take a look at one of the actual file addresses to get a sense of
# what the Pangeo file addresses look like.:
print(my_recipes['pr_file'].iloc[0])
```

```
You have requested more recipes than possible for at least one target
trajectories, returning what can
  target_start_yr target_end_yr archive_experiment archive_variable  \
0            1858          1866         historical              tas
1            1867          1875         historical              tas
2            1876          1884         historical              tas
3            1885          1893         historical              tas
4            1894          1902         historical              tas

  archive_model archive_ensemble       stitching_id archive_start_yr  \
0       CanESM5         r1i1p1f1  ssp245~r1i1p1f1~1             1858
1       CanESM5        r25i1p1f1  ssp245~r1i1p1f1~1             1858
2       CanESM5        r17i1p1f1  ssp245~r1i1p1f1~1             1903
3       CanESM5         r2i1p1f1  ssp245~r1i1p1f1~1             1912
4       CanESM5         r1i1p1f1  ssp245~r1i1p1f1~1             1903
```

```
  archive_end_yr                                              tas_file  \
0           1866  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
1           1866  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
2           1911  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
3           1920  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
4           1911  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...

                                              pr_file
0  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
1  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
2  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
3  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
4  gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical...
-----------------------------------
gs://cmip6/CMIP6/CMIP/CCCma/CanESM5/historical/r1i1p1f1/Amon/pr/gn/v20190429/
```

If you wanted to include sea level pressure in addition to precipitation, you would use `non_tas_variables=['pr', 'psl']`.

stitching and plotting

Nearest neighbor result

Stitch the global average temperature for the nearest neighbor result, and see it in the context of the actual ESM data that was not used in the archive at all.

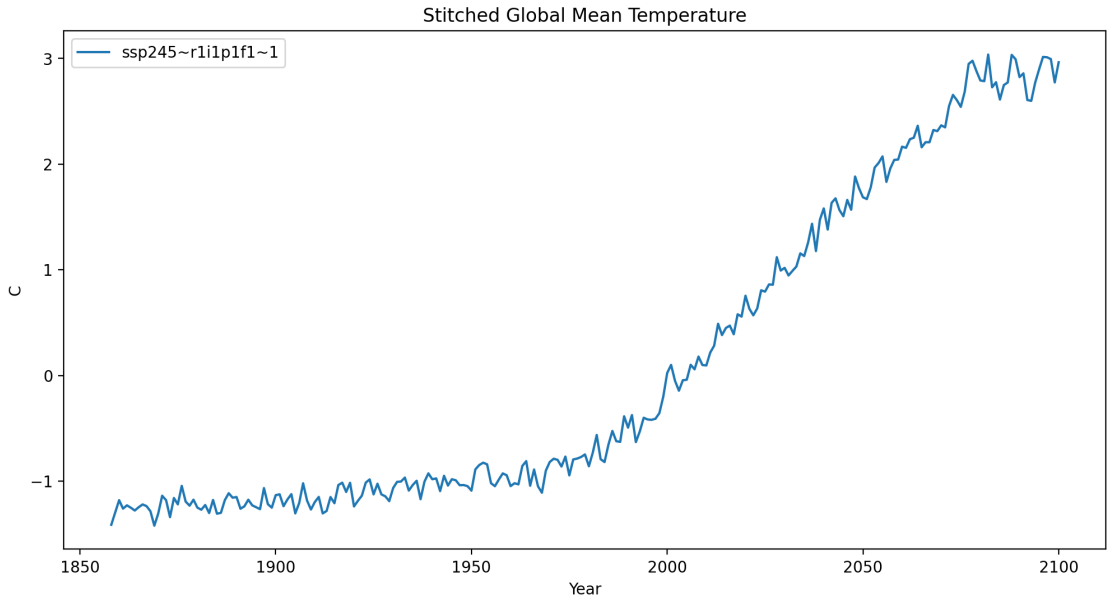
```
stitched_global_temp = stitches.gmat_stitching(nn_recipes)
```

nearest neighbor stitched realization

```
groups = stitched_global_temp.groupby('stitching_id')

for name, group in groups:
    plt.plot(group.year, group.value, label = name)

plt.xlabel("Year")
plt.ylabel("C")
plt.title("Stitched Global Mean Temperature")
plt.legend()
plt.show()
plt.close()
```

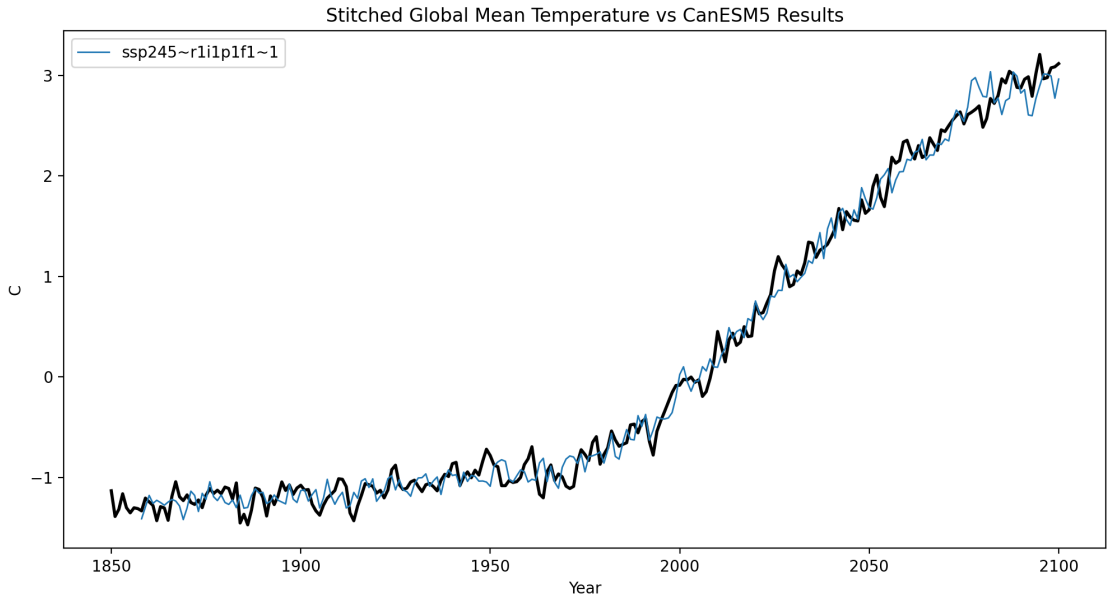
```
# Load the comparison GSAT data
data_directory = pkg_resources.resource_filename("stitches", "data")
data_path = os.path.join(data_directory, "tas-data", "CanESM5_tas.csv")
```

```
comp_data = pd.read_csv(data_path)
comp_data = comp_data.loc[comp_data["experiment"] == "ssp245"]
```



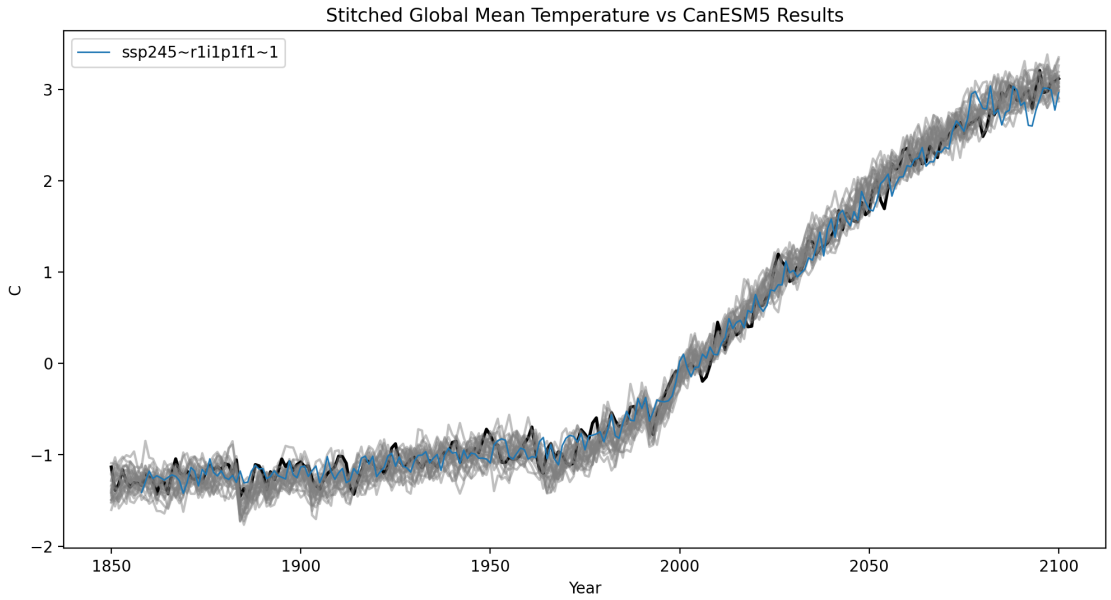
stitched realization and the target ensemble member

```
groups = comp_data.groupby('ensemble')
for name, group in groups:
    if(group.ensemble.unique() == 'r1i1p1f1'):
        plt.plot(group.year, group.value, color = "black", linewidth = 2.0)
# The stitched realizations:
groups = stitched_global_temp.groupby('stitching_id')
for name, group in groups:
    plt.plot(group.year, group.value, linewidth= 1.0, label = name)

plt.legend()
plt.xlabel("Year")
plt.ylabel("C")
plt.title("Stitched Global Mean Temperature vs CanESM5 Results")
plt.show()
plt.close()
```



stitched realization and the entire scenario ensemble

```
# full ensemble of actual ESM runs:
groups = comp_data.groupby('ensemble')
for name, group in groups:
    if(group.ensemble.unique() == 'r1i1p1f1'):
        plt.plot(group.year, group.value, color = "black", linewidth = 2.0)
    else:
        plt.plot(group.year, group.value, color = "0.5", alpha=0.5)

# The stitched realizations:
groups = stitched_global_temp.groupby('stitching_id')
for name, group in groups:
    plt.plot(group.year, group.value, linewidth= 1.0, label = name)

plt.legend()
plt.xlabel("Year")
plt.ylabel("C")
plt.title("Stitched Global Mean Temperature vs CanESM5 Results")
plt.show()
plt.close()
```



gridded stitching of the non-NN recipes

This is a little slow, but it will create the netcdfs according to our stitched recipes that we can load in and work with.

```
stitches.gridded_stitching(out_dir='.',  rp=my_recipes)
```

```
['Stitching gridded netcdf for: CanESM5 tas ssp245~r1i1p1f1~1']
['Stitching gridded netcdf for: CanESM5 tas ssp245~r1i1p1f1~2']
['Stitching gridded netcdf for: CanESM5 tas ssp245~r1i1p1f1~3']
```

```
['./stitched_CanESM5_tas_ssp245~r1i1p1f1~3.nc',
 './stitched_CanESM5_pr_ssp245~r1i1p1f1~3.nc']
```

now you have created multiple gridded, monthly tas and pr files that are statistically consistent with the target: SSP245 realization 1.

```
import xarray as xr

gen_tas = xr.open_dataset('stitched_CanESM5_tas_ssp245~r1i1p1f1~1.nc')
gen_tas
```

```
<xarray.Dataset>
Dimensions:  (time: 2916, lat: 64, lon: 128)
Coordinates:
  * time     (time) datetime64[ns] 1858-01-31 1858-02-28 ... 2100-12-31
  * lat      (lat) float64 -87.86 -85.1 -82.31 -79.53 ... 79.53 82.31 85.1
87.86
  * lon      (lon) float64 0.0 2.812 5.625 8.438 ... 348.8 351.6 354.4 357.2
Data variables:
    tas      (time, lat, lon) float32 ...
```

```
gen_pr = xr.open_dataset('stitched_CanESM5_pr_ssp245~r1i1p1f1~1.nc')
gen_pr
```

```
<xarray.Dataset>
Dimensions:  (time: 2916, lat: 64, lon: 128)
Coordinates:
  * time     (time) datetime64[ns] 1858-01-31 1858-02-28 ... 2100-12-31
  * lat      (lat) float64 -87.86 -85.1 -82.31 -79.53 ... 79.53 82.31 85.1
87.86
  * lon      (lon) float64 0.0 2.812 5.625 8.438 ... 348.8 351.6 354.4 357.2
Data variables:
    pr       (time, lat, lon) float32 ...
```

Pull comparison netcdfs

```
# Fetch the actual data directly from pangeo
data_directory = pkg_resources.resource_filename("stitches", "data")
pangeo_path = os.path.join(data_directory, "pangeo_table.csv")

pangeo_data = pd.read_csv(pangeo_path)

pangeo_data = pangeo_data.loc[(pangeo_data['variable'].isin(['tas', 'pr']))
                              & (pangeo_data['domain'].str.contains('mon'))
                              & (pangeo_data['experiment'].isin(['ssp245']))
                              & (pangeo_data['ensemble'].isin(['r1i1p1f1']))
                              &
(pangeo_data['model'].isin(['CanESM5']))].copy()

# load the target tas netcdf files
tas_address = pangeo_data.loc[pangeo_data['variable']== 'tas'].zstore.copy()
tar_tas = stitches.fetch_nc(tas_address.values[0])

# load the target pr netcdf files
pr_address = pangeo_data.loc[pangeo_data['variable']== 'pr'].zstore.copy()
tar_pr = stitches.fetch_nc(pr_address.values[0])
```

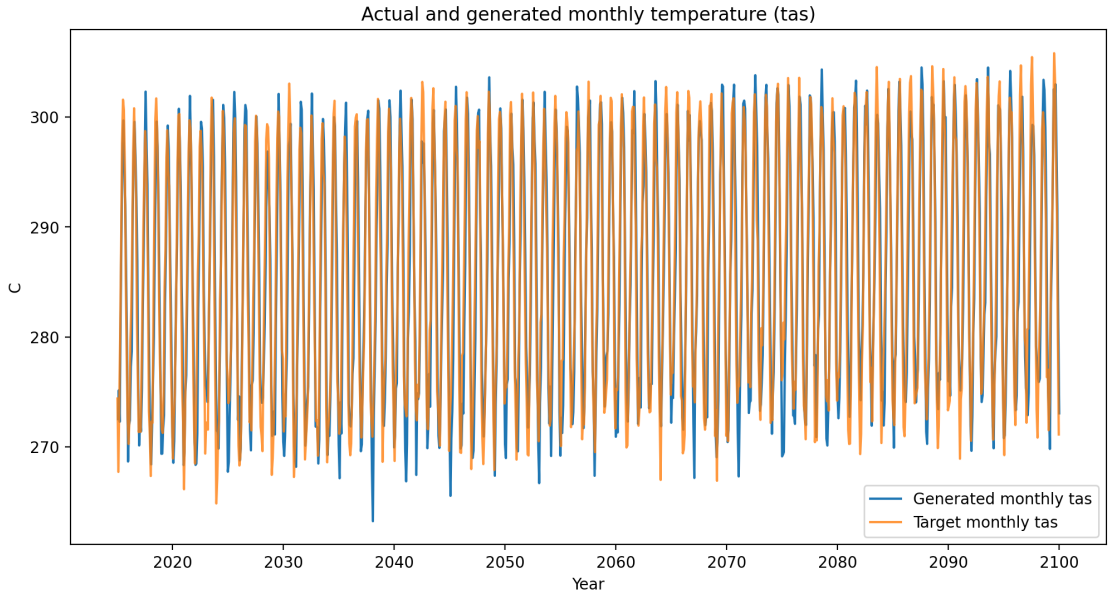
Visualize

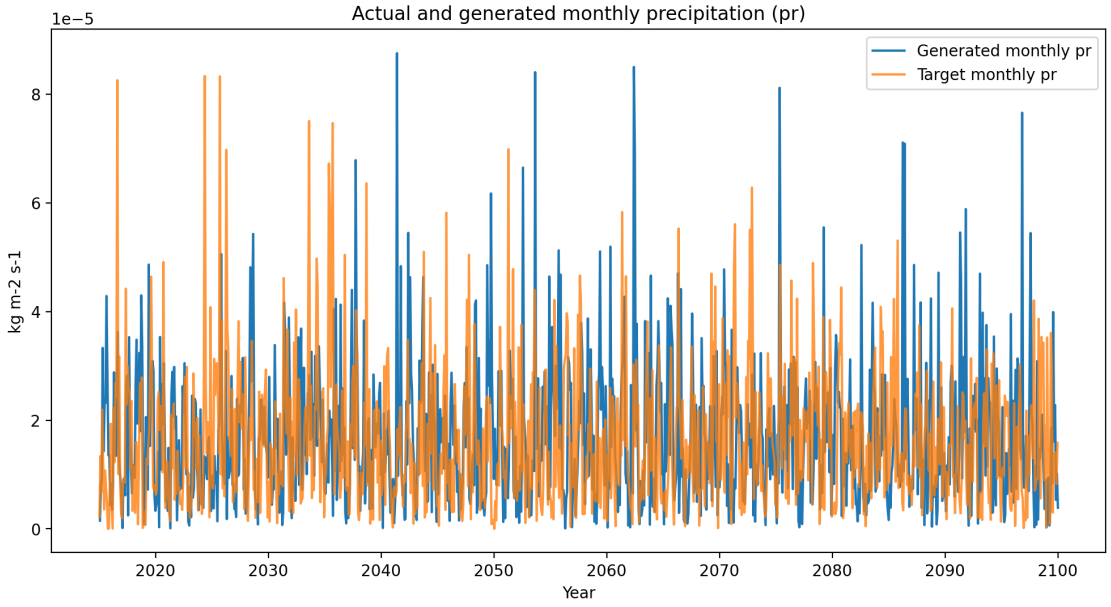
Select a grid cell and plot the generated and target tas, pr data for first-cut comparison

```
# define a helper function
def plot_comparison(generated_data,
                    target_data,
                    variable,
                    alpha=0.8):
    """Plot comparision between target variable time series and generated
data"""
```

```
    if variable.casefold() == "pr":
        variable_name = "precipitation"
        units = "kg m-2 s-1"
    else:
        variable_name = "temperature"
        units = "C"

    # temperature (tas)
    plt.plot(generated_data.time,
             generated_data[variable],
             label=f"Generated monthly {variable}")

    with warnings.catch_warnings():
        warnings.filterwarnings("ignore")

        plt.plot(target_data.indexes['time'].to_datetimeindex(),
                 target_data[variable],
                 alpha=alpha,
                 label = f"Target monthly {variable}")

    plt.legend()
    plt.xlabel("Year")
    plt.ylabel(units)
    plt.title(f"Actual and generated monthly {variable_name} ({variable})")
    plt.show()
    plt.close()


# lon and lat values for a grid cell near the Joint Global Change Research
Institute in College Park, MD, USA
cp_lat = 38.9897
cp_lon = 180 + 76.9378

# lat and lon coordinates closest
abslat = np.abs(gen_tas.lat - cp_lat)
abslon = np.abs(gen_tas.lon-cp_lon)
c = np.maximum(abslon, abslat)
([lon_loc], [lat_loc]) = np.where(c == np.min(c))
lon_grid = gen_tas.lon[lon_loc]
lat_grid = gen_tas.lat[lat_loc]

cp_tas_gen = gen_tas.sel(lon=lon_grid,
                         lat=lat_grid,
                         time=slice('2015-01-01', '2099-12-31')).copy()

cp_tas_tar = tar_tas.sel(lon=lon_grid,
                         lat=lat_grid,
                         time=slice('2015-01-01', '2099-12-31')).copy()
```

```python
cp_pr_gen = gen_pr.sel(lon=lon_grid,
                       lat=lat_grid,
                       time=slice('2015-01-01', '2099-12-31')).copy()

cp_pr_tar = tar_pr.sel(lon=lon_grid,
                       lat=lat_grid,
                       time=slice('2015-01-01', '2099-12-31')).copy()

# temperature (tas)
plot_comparison(generated_data=cp_tas_gen,
                target_data=cp_tas_tar,
                variable="tas")

# precipitation (pr)
plot_comparison(generated_data=cp_pr_gen,
                target_data=cp_pr_tar,
                variable="pr")
```





Visual validation of the complex spatial, temporal, and cross-variable relationships present in ESM outputs is not possible. We extensively validate that the method reproduces ESM internal variability in the ESD paper, but this visual plotting at least suggests that nothing is obviously wrong. In particular, there are no obvious artifacts occurring every 9-years in the generated time series.

In other words, it's not inconceivable from these plots that the orange time series were sampled from the same underlying multivariate distribution that generated the blue time series.