DUPLEXITY

A Python package for weather forecast validation

AGU24

Bodeker Scientific

Lexi Xu, Emily O'Riordan, Greg Bodeker, William Naylor
Bodeker Scientific, Alexandra, New Zealand

Contact: emily@bodekerscientific.com

What is Duplexity?

While developing our own AI-based weather forecast models and validating them against other data sources, we found a substantial amount of time was spent researching validation metrics, and we often needed to implement them from scratch.

We built **Duplexity** to resolve this issue, by providing:

- An **open-source Python package** with many types of validation metrics suitable for various weather data types;
- Comprehensive **documentation** to help users **find appropriate metrics** for their data types;
- A **contributable library** for others in the field to add their own metrics.

duplexity.pixelwise.gilbert_skill_score(*observed: ndarray | DataArray | Dataset | DataFrame | List[ndarray | DataArray | Dataset | DataFrame], output: ndarray | DataArray | Dataset | DataFrame | List[ndarray | DataArray | Dataset | DataFrame], threshold: float, var: str = None)→ float* [source]

Calculate the Gilbert Skill Score (GSS), also known as the Equitable Threat Score, between observed and model output values based on a specified threshold. It adjusts the Critical Success Index by accounting for hits that could occur by random chance.

Parameters:
- **observed** (*Union[np.ndarray, xr.DataArray, xr.Dataset, pd.DataFrame, List[Union[np.ndarray, xr.DataArray, xr.Dataset, pd.DataFrame]]]*) – Array of shape (h, w) or (n, h, w) containing observed binary or continuous values, where n is the number of samples, h is the height, and w is the width.
- **output** (*Union[np.ndarray, xr.DataArray, xr.Dataset, pd.DataFrame, List[Union[np.ndarray, xr.DataArray, xr.Dataset, pd.DataFrame]]]*) – Array of shape (h, w) or (n, h, w) containing model output binary or continuous values, where n is the number of samples, h is the height, and w is the width.
- **threshold** (*float*) – A threshold value used to convert continuous output values into binary classifications (0 or 1). Values greater than or equal to the threshold will be classified as 1, and values below the threshold will be classified as 0.
- **var** (*str (default: None)*) – The name of the variable to be used in the calculation. If *var* is None, the function will use the first variable in the dataset. Only applicable when the inputs are provided as xarray Datasets.

Returns: The Gilbert Skill Score (GSS), also known as the Equitable Threat Score (ETS), which adjusts the Critical Success Index (CSI) by accounting for hits that could occur due to random chance. GSS = (TP - CH) / (TP + FN + FP - CH) where CH (Chance Hits) = (TP + FN) * (TP + FP) / (TP + FN + FP + TN)

Return type: float

Notes

The Gilbert Skill Score (GSS) is a metric used in binary classification to assess the skill of a model by considering both correct predictions and the impact of random chance. It is particularly useful in cases involving rare events or imbalanced datasets, where traditional metrics like accuracy may be misleading.

If the inputs *observed* and *output* are provided as lists of *xr.DataArray*, *np.array*, or *pd.DataFrame*, the function will calculate the GSS for each pair of elements in the lists and then return the average GSS.

Fig 1. An example of the Duplexity documentation

Figure 1 shows an example of the **documentation** available for functions in the Duplexity package.

We aim to create a useful resource for researchers to choose the correct metrics for their use case. We highlight **the benefits and drawbacks of specific metrics**, and outline **how the metric is calculated**.

Many implementation examples are given in the documentation.

What data types does Duplexity support?

pandas xarray NumPy

Duplexity currently supports the following data types:

- **Xarray** Datasets and DataArrays;
- **Numpy** N-Dimensional Arrays;
- **Pandas** DataFrames and Series.

Support for other data types, such as Iris cubes, may be added in the future if there is appetite.

Statistical comparisons with Duplexity

Comparisons between datasets is made easy using Duplexity. We compare ERA5 precipitation reanalysis (Figure 2a) to a quantitative precipitation estimation (QPE) product (Figure 2b) for Aotearoa New Zealand, which we take to be the ground truth.

Figure 3 shows a **single line function call with Duplexity calculates a range of metrics** to evaluate ERA5 against the QPE truth data.

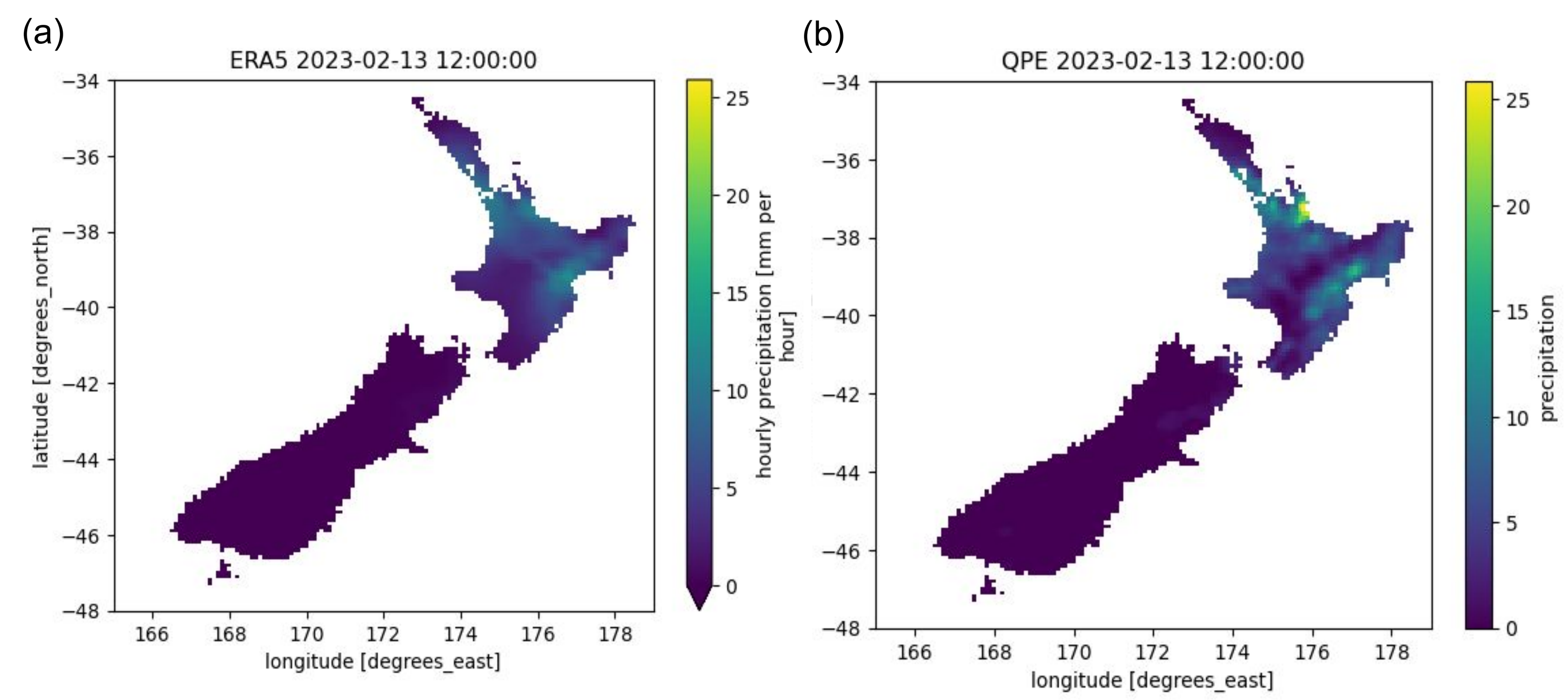

Fig 2. (a) ERA5 precipitation and (b) Quantitative Precipitation Estimation (QPE) for Aotearoa New Zealand during Cyclone Gabrielle, February 2023. These datasets are compared in Figure 3.

```python
from duplexity import pixelwise
pixelwise.calculate_pixelwise_metrics(QPE['precipitation'].values,
                                      ERA5['precipitation'].values)
```

✓ 0.0s

	Metric Name
{'cm': array([[1141, 80], [71, 40919]]),	Confusion matrix
'precision': 0.9414191419141914,	Precision
'recall': 0.9344799344799345,	Recall
'f1': 0.9379367036580353,	F1 score
'accuracy': 0.9964227334107223,	Accuracy
'csi': 0.8831269349845201,	Critical success index
'far': 0.05858085808580858,	False alarm ratio
'pod': 0.9344799344799345,	Probability of detection
'gss': 0.8798671272049028,	Gilbert skill score
'hss': 0.9360950191337625,	Heidke skill score
'pss': 0.9327478046921814,	Peirce skill score
'sedi': 0.9848144937204649,	Symmetric extremal dependence index
'mae': 0.7376003125799381,	Mean absolute error
'mse': 2.385185508820302,	Mean squared error
'rmse': 1.5444045806783602,	Root mean squared error
'bias': -0.11431841170119432,	Bias
'drmse': 1.5401677861734475,	Debiased RMSE
'corr': 0.8893871637047687}	Pearson correlation

Fig 3. An example of using Duplexity to calculate a range of pixel-wise metrics.

Plotting Capabilities

```python
from duplexity.imagewise import rapsd
from duplexity.plot import plot_rapsd

fig, ax = plt.subplots(figsize=(6, 4))

profile_era5, freqs_era5 = rapsd(ERA5['precipitation'].values)
profile_qpe, freqs_qpe = rapsd(QPE['precipitation'].values)

kwargs = {'x_units': "km", 'y_units': "Power", 'ax': ax}

plot_rapsd(freqs_era5, profile_era5, color='blue', label='ERA5', **kwargs)
plot_rapsd(freqs_qpe, profile_qpe, color='red', label='QPE', **kwargs)
```

✓ 0.3s

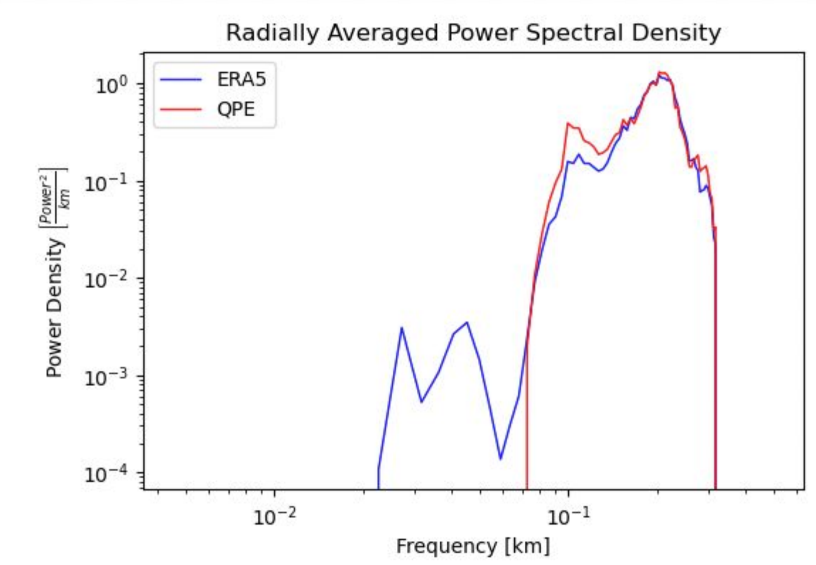

Fig 4. Using Duplexity to plot the radially averaged power spectral density of QPE and ERA5 for New Zealand.

In Figure 4, we use Duplexity to calculate and plot the radially averaged power spectral density (RAPSD) of the two datasets shown in Figure 2.

What is the RAPSD metric? The docstring in Figure 5 tells us:

```python
rapsd?
```

✓ 0.0s

```
Signature:
rapsd(
    data: Union[numpy.ndarray, xarray.core.dataarray.DataArray, xarray.core.datas
    fft_method: Optional[str] = None,
    return_freq: bool = True,
    pixel_spacing: float = 1.0,
    normalize: bool = False,
    epsilon: float = 1e-10,
) -> Union[numpy.ndarray, Tuple[numpy.ndarray, numpy.ndarray]]
Docstring:
Compute the Radially Averaged Power Spectral Density (RAPSD) of a 2D field.

This function calculates the RAPSD of a 2D data field, which is a measure of the
distribution of power into frequency components composing that field. The method
uses the 2D Fourier Transform to compute the power spectral density and then
averages this radially from the center of the frequency domain.
```

Fig 5. Docstring of the radially averaged power spectral density

What metrics are available?

Duplexity is split into several submodules:

- **Pixelwise**: Metrics in this category perform comparisons on a pixel-wise level, such as the mean_absolute_error, bias and pearson_correlation, confusion_matrix, f1_score and false_alarm_ratio.
- **Probabilistic:** Metrics in this category evaluate probabilistic forecasts, examples include the CRPS (continuous ranked probability score) and the ROC_AUC (area under receiver operator curve).
- **Imagewise:** These metrics evaluate the perceptual similarity between datasets using image processing methods. Metrics include PSNR (peak signal-to-noise ratio), SSIM (structural similarity index) and RAPSD (radially averaged power spectral density).
- **Spatial:** Metrics in this category consider neighbourhood accuracy, including FSS (fractional skill score).

Many other metrics are available, take a look at the documentation at **duplexity.readthedocs.io**

Try out Duplexity yourself!

https://github.com/lexixu19/Duplexity

This package is still under development and we appreciate any feedback. We are keen to implement new and useful metrics, so please get in touch if there is a new feature you'd like to see added to Duplexity!