**Manual for** **NoiseQC\_V1.0**

Yichen Pan

University of Science and Technology of China

January, 26, 2025

This is a ***python*** program developed for quality control of horizontal component ambient noise cross-correlation functions (CCFs) based on the Rayleigh wave phase-matching method. Regarding the Rayleigh wave phase-matching method, you can refer to Baker & Stevens (2004) for the principles of this method applied to seismic events, and to Zha et al. (2013) for its principles when applied to noise data. About the details of our practice, you can refer to Pan & Yao (in prep.).

In this program, we provide the code to compute the correlation coefficient between and (or ) and perform quality control on the dispersion data, including the code to exhibit the figure. To help users better understand how to use the program, we also provide an example for the CCFs of a single pair of stations, followed by a statistical example for a set of data from a dense array. If there are any questions, comments, and bug reports, please feel free to email [pyc2020@mail.ustc.edu.cn](mailto:pyc2020@mail.ustc.edu.cn) or [ycpan03@gmail.com](mailto:ycpan03@gmail.com).

**References**

Baker, G. E., & Stevens, J. L. (2004). Backazimuth estimation reliability using surface wave polarization. *Geophysical Research Letters*, *31*(9), 2004GL019510. <https://doi.org/10.1029/2004GL019510>

Zha, Y., Webb, S. C., & Menke, W. (2013). Determining the orientations of ocean bottom seismometers using ambient noise correlation. *Geophysical Research Letters*, *40*(14), 3585–3590. <https://doi.org/10.1002/grl.50698>

Pan, Y., & Yao, H. (in prep). Quality Control Method for Horizontal-component Ambient Noise Cross-correlations and its Application to Radial Anisotropy Tomography of the Central Tanlu Fault Zone.

**Links**

1. NoiseQC\_V1.0
2. [EGFAnalysisTimeFreq](https://yaolab.ustc.edu.cn/_upload/tpl/10/f0/4336/template4336/pdf/EGFAnalysisTimeFreq_version_2015.zip) (by Huajian Yao)
3. [NoiseCorr\_SAC](https://yaolab.ustc.edu.cn/_upload/tpl/10/f0/4336/template4336/pdf/NoiseCorr_2016Jul_v4_2.zip) (by Huajian Yao)
4. [FastXC](https://github.com/wangkingh/FastXC) (by Jingxi Wang)

**Content**

[1 Structure 3](#_Toc190292780)

[2 QuickStart 3](#_Toc190292781)

[2.1 Installation 3](#_Toc190292782)

[2.2 File Location 3](#_Toc190292783)

[2.3 Basic Parameters for a Quick Start 4](#_Toc190292784)

[2.4 Input and Output 4](#_Toc190292785)

[3 Parameters and Functions 4](#_Toc190292786)

[4 Examples 6](#_Toc190292787)

[4.1 Single Mode 6](#_Toc190292788)

[4.2 Statistic Mode 10](#_Toc190292789)

[5 Details of the Code 11](#_Toc190292790)

[5.1 Data Read 11](#_Toc190292791)

1 Structure

- ***NoiseQC***

1. ***main.py***, defines the basic classes and functions we used in this program

2. ***run\_single.py***, (**Single Mode**) the example for calculating the QC parameters (including the correlation coefficient ( and ), optimal correction angle, SNR and lag-time () between and ) for the CCFs between a single pair of stations.

3. ***run\_statistic.py***, (**Statistic Mode**) the statistical example showing how to conduct the quality control process and set the threshold of QC parameters.

2 QuickStart

This guide provides a brief introduction to program installation, data file preparation, and essential parameter configuration, ensuring an efficient start for users. **Chapter 3** and **4** will provide detailed explanations and examples of this program.

2.1 Installation

This program can be downloaded from [NoiseQC.](https://github.com/Ycpan-seis/NoiseQC)

2.2 File Location

In **Single Mode**, we utilize the ***tkinter*** package to provide an interactive interface for selecting the Z-Z component CCF of the station pair to be processed. The storage structure of the CCFs files follows the output format of [***NoiseCorr\_SAC***](https://yaolab.ustc.edu.cn/_upload/tpl/10/f0/4336/template4336/pdf/NoiseCorr_2016Jul_v4_2.zip) (Yao et al., 2006, 2011), as shown below,

users only need to select the CCF data in the **Z-Z**, while the other components can be read automatically.

In **Statistic Mode**, users should modify the following variables to select the path:

* ***Flag.data\_path***, the directory path of the folder containing the Z-Z component CCFs (the other will be read automatically)
* ***Flag.disp\_path***, the directory path of the folder containing the dispersion data for quality control.
* ***Flag.run\_path***, the directory path for running an example, contains **INPUT** and **OUTPUT** folders.
* ***stafile***, file containing the information of the stations, with the default being ‘**./INPUT/sta\_all.txt**’. (Format: station name; array name; latitude; longitude)

2.3 Basic Parameters for a Quick Start

The parameters are defined in the Para and Flag classes, and the meaning of each parameter can be found in the accompanying comments. This section will focus on the basic parameter settings required to run the code. For more advanced usage and customization to fit your own data, please refer to the detailed explanation in the following chapters.

**Para**

* **Sample\_rate**, sample rate of CCFs data.
* **Vlow/Vhigh**, the lower/upper limit of group wave velocity
* **Ts/Te**, the period range; **dT**, the interval of period

**Flag**

* **Corr\_Method**, 1: non-normalized correlation coefficient; 2: normalized
* **Auto**, 0: **Single Mode**; 1: **Statistic Mode**

2.4 Input and Output

Users need to prepare the following input files:

* Nine-component CCFs
* Dispersion data that requires quality control. (Only in **Statistic Mode**)
* Reference dispersion data to choose the period. (Only in **Single Mode** and **Period\_setting** = 1)

After preparing the input data and parameters, run **run\_single.py** or **run\_statistic.py**. In **Single Mode**, figures are output to ***run\_path****/single*, and results are displayed in the terminal. and results are displayed in the terminal. In **Statistic Mode**, figures are output to ***run\_path****/OUTPUT*.

3 Parameters and Functions

An introduction to the meaning and purpose of the parameters and functions defined in ***main.py***, methods will also be discussed.

**Parameters**

The parameters are categorized into two classes, **Para** defines values for data processing, computation, and QC thresholds, and **Flag** contains parameters for selecting methods.

**Para**

* **Sample\_rate**, sample rate of CCFs data.
* **nsta**, the number of stations to be included as source in the statistic (not the total number of stations)
* **dv**, velocity interval of group velocity spectrum (refer to EGFAnalysisTimeFreq, no need to change).
* **Vlow/Vhigh**, the lower/upper limit of group wave velocity to set the signal window.
* **Ts/Te**, the period range; **dT**, the interval of period.
* **Low\_bound/High\_bound**, the lower/upper boundary of the angle search range (default: -180/180), the interval is 1°.
* **T\_show**, the period for which you want to show its angle-correlation coefficient diagram (**Figure 4**). (Only set for **Single Mode**)
* **T\_num**, the window length of time variable filtering. (refer to EGFAnalysisTimeFreq, no need to change)
* **minSNR/minCorr/maxLag/maxAngle**, the thresholds of quality control parameters - SNR/**/****/**Optimal Angle. Users can choose the values based on statistical analysis, this is a trade-off between data quantity and quality, details are discussed in Pan & Yao (in prep.). (Only set for **Statictic Mode**)

**Flag**

* **Corr\_Method**, choose the method calculating the correlation coefficient, 1: non-normalized correlation coefficient; 2: normalized.
* **Half**, choose which half of CCFs to use, 1: Positive half-axis data; -1: Use negative half-axis data; 0: Average of positive and negative half-axis
* **Auto**, 0: **Single Mode**; 1: **Statistic Mode.**
* **Period\_setting**, 1: get the period from reference dispersion file; 0: use the period set in **Para.T**. (Only works when **Auto** = 0, the reference option is not set when **Auto** = 1)
* **Rotate**, set whether to rotate the data (T-T component CCFs) passed the QC process based on the orientation correction angle. 1: Rotate and save the CCFs that meet the conditions; 0: Don't rotate and directly restore the dispersion after the QC process. (Only works when **Auto** = 1)
* **Save\_png**, set whether to restore the figures, 1: Save image; 0: Do not save image. (**Single Mode**: runPath/single; **Statistic Mode**: runPath/OUTPUT/Figure)
* **data\_path/disp\_path/run\_path**, paths discussed in 2.2.
* **ref\_disp**, the directory path of the folder containing the reference dispersion data. (Only set when **Period\_setting** = 1)

**Functions**

**DataRead**

Once the data has been prepared in the previously specified format, this function will retrieve all nine components file names based on the Z-Z component file name users provide and load the data. You can select which half of the data to use by setting **Flag.Half**.

**AngleSearch**

This is the core function of the Rayleigh wave phase matching method, used to calculate the correlation between the Z-Z and Z-R/R-Z component CCFs.

Firstly, we conduct preprocessing for the data, including:

* setting the signal window based on the reference group velocity (**Vlow/Vhigh**) – Fun **VelocityWindow**
* setting the time-varying filter. – Fun **CutWindow**
* applying the bandpass filter to the target frequency band (**Ts/Te**). – Fun **Bandpass**

Then traverse the angle range (**Low\_bound/High\_bound**) and calculate the correlation coefficient according to the following equations,

in which represents 90° phase-shifted , is the zero-lag cross correlation between time series and . Note that the correlation coefficient between and is also calculated.

We will get the optimal angle by searching for the maximum value of , and the corresponding is the key parameter for the QC process.

**AngleSearchT**

The frequency band divided version of the function **AngleSearch**, only works in Single Mode.

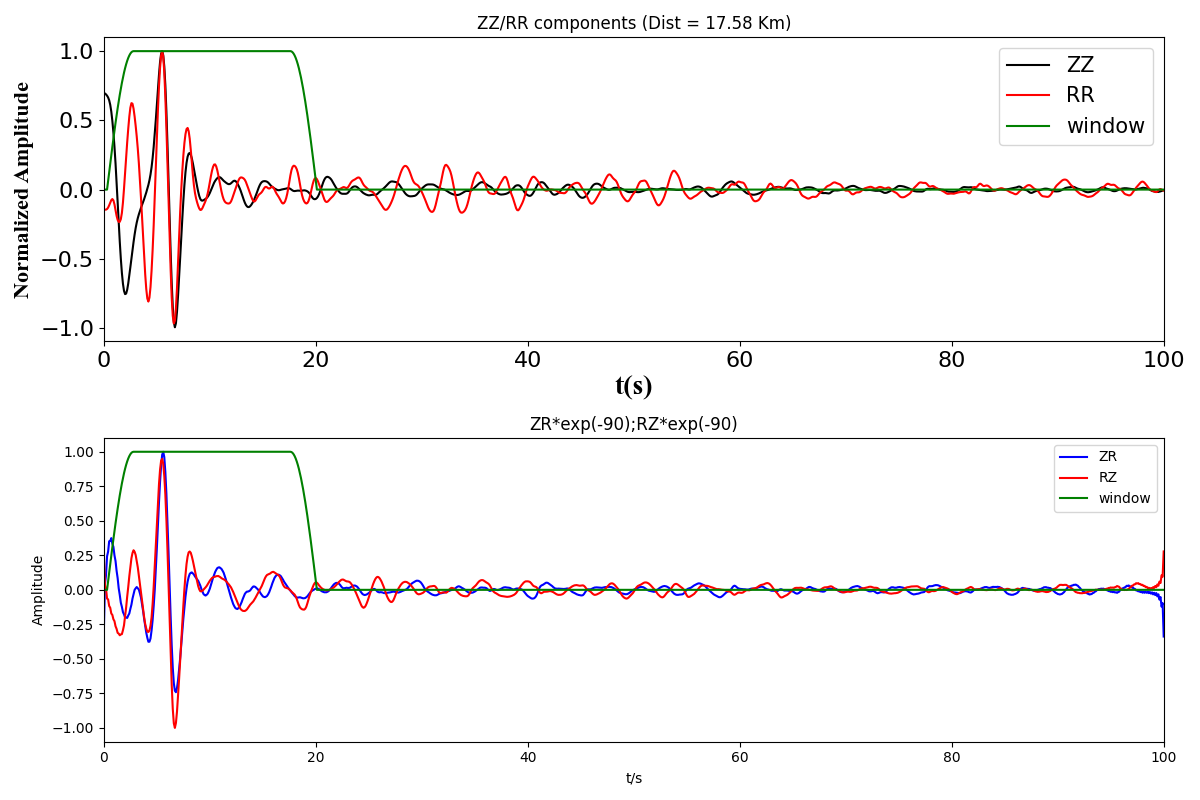
Filter the data to each sub-band within the range Ts:dt:Te, calculate the correlation coefficient, and search for the optimal angle. If the chosen frequency bands are appropriate, the optimal angle should remain relatively stable (**Figure 5**).

4 Examples

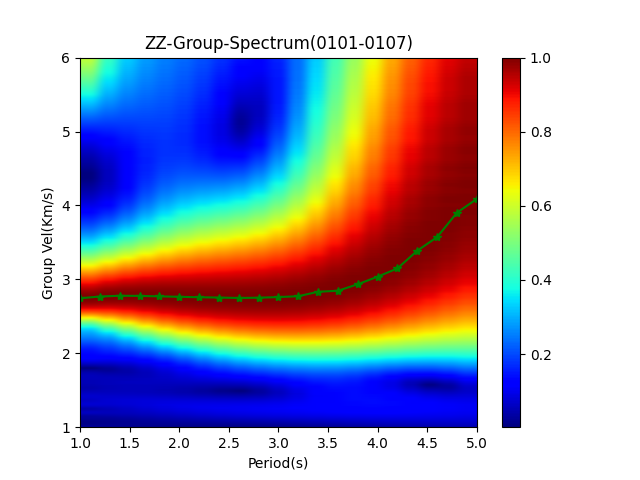
To help users better understand this project, we provide test data in *./EXAMPLE/INPUT*, along with a demonstration of the execution process.

4.1 Single Mode

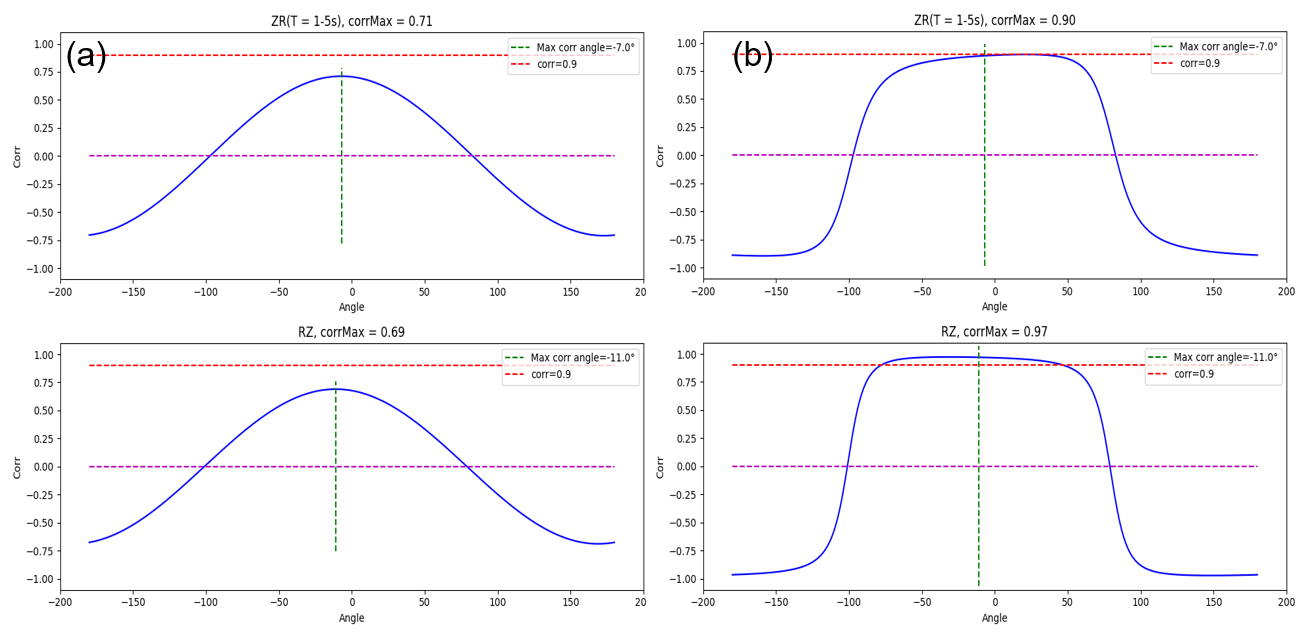
After the data is prepared in the required format, users can set the parameters as described in **Chapter 3** (**Flag.Auto** must be 0), then run ***run\_single.py*** and select the Z-Z component of the data to be used. Station information and the execution results will be output to the terminal, and several figures will be generated as well.



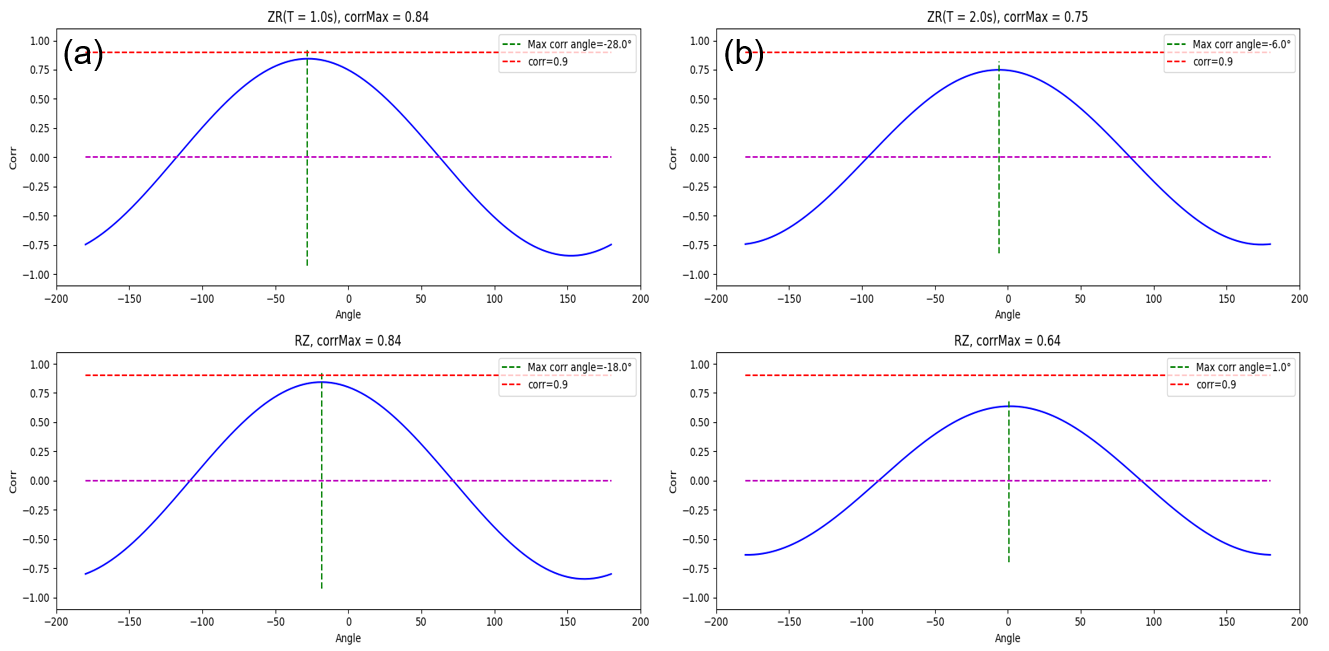
**Figure 1.** Multi-component CCFs of the selected station pair. (Fun **PlotSeis**)



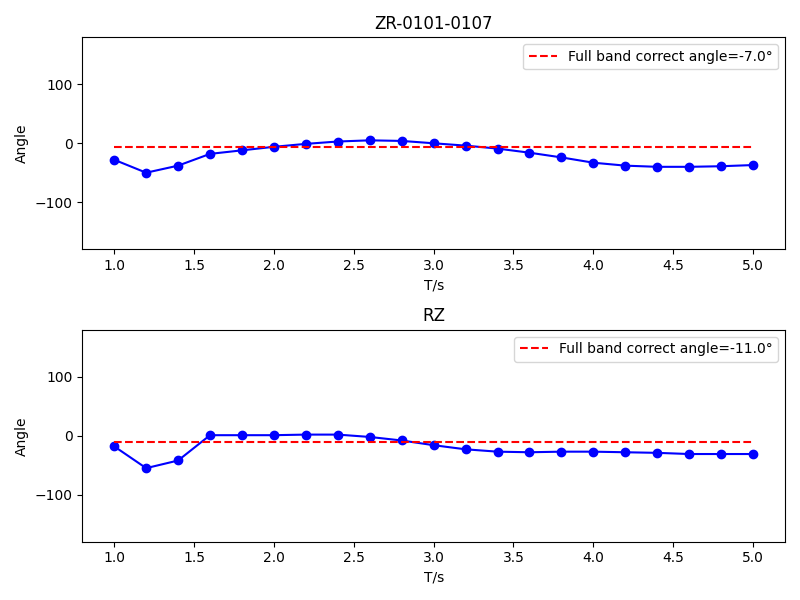
**Figure 2.** Group velocity dispersion analysis, the green line is the traced group velocity dispersion. This figure can help verify whether the data within the selected frequency band is reliable. (Fun **PlotSpectrum**)



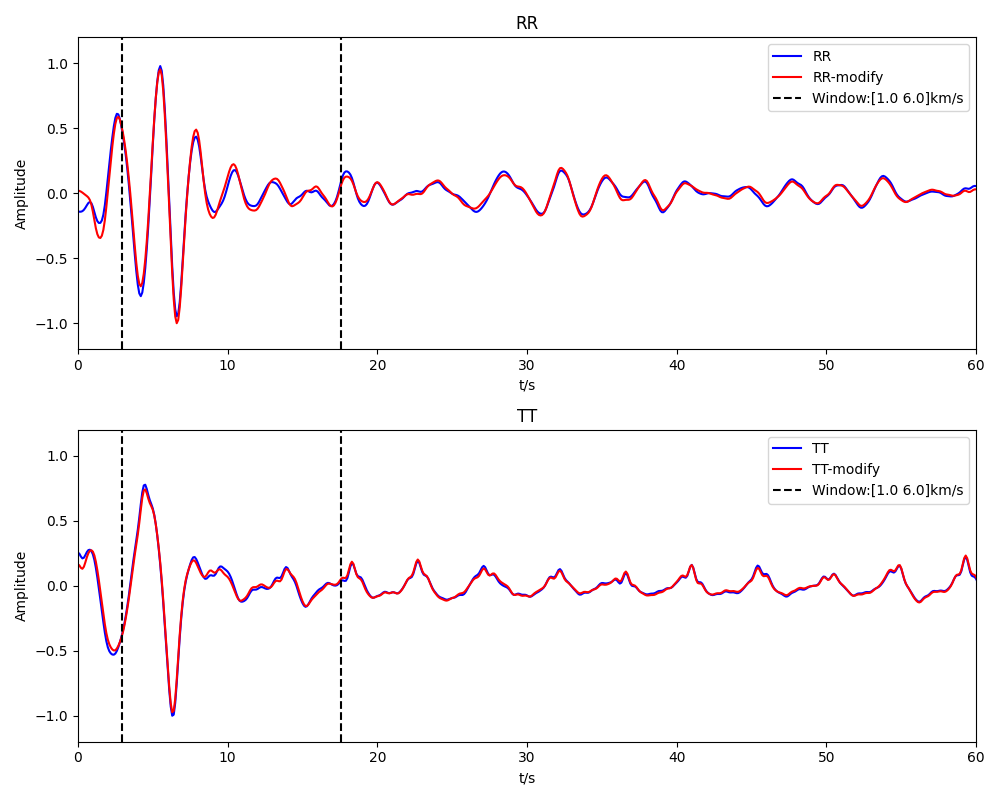
**Figure 3**. Correlation coefficient changes with rotation angle, use the whole frequency band. (a) Non-normalized, **Corr\_Method** = 1; (b) Normalized, **Corr\_Method** = 2. The optimal angle (green dashed line) is selected according to . (Fun **PlotAngleCorr**)



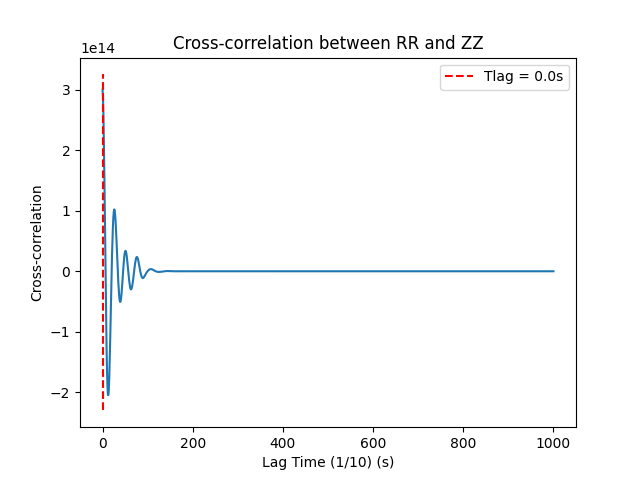
**Figure 4**. Non-normalized (**Corr\_Method** = 1) correlation coefficient changes with rotation angle, use the frequency set in **Para.T\_show**. (a) T=1.0s; (b) T=2.0s. (Fun **PlotAngleCorr**, called within Fun **Base\_SearchT**)



**Figure 5**. The optimal correction angle changes with period.



**Figure 6**. Comparison of R-R and T-T components of CCFs before and after rotation using the optimal correction angle.



**Figure 7**. The zero-lag cross-correlation between R-R and Z-Z components of CCFs. The red dashed line marks the position of the maximum cross-correlation value, corresponding to .

4.2 Statistic Mode

Edit the path parameters in the **Flag** as described in 2.2, set Para.Auto = 1, and adjust the other parameters as needed. In the example, we provide the CCFs between station **Sta-0101** and the other stations in the array (so we set **nsta** = 1), then run ***run\_statistic.py*** to calculate the QC parameters and generate statistics for this station. Of course, users can edit the loop statements in the code to select which stations to calculate.

The generated histogram of the QC parameters will be saved in *EXAMPLE/OUTPUT/Figure* (**Figure 8**).

If **Rotate** = 0, the dispersion data passed the QC process will be stored in *EXAMPLE/OUTPUT/dispData*. (**Note**: In the provided examples and code, the QC process is applied to the dispersion data. However, the QC process can also be directly applied to the CCFs data, which can help reduce the effort required for extracting dispersion. With only minor adjustments to the output, this can be easily achieved.)

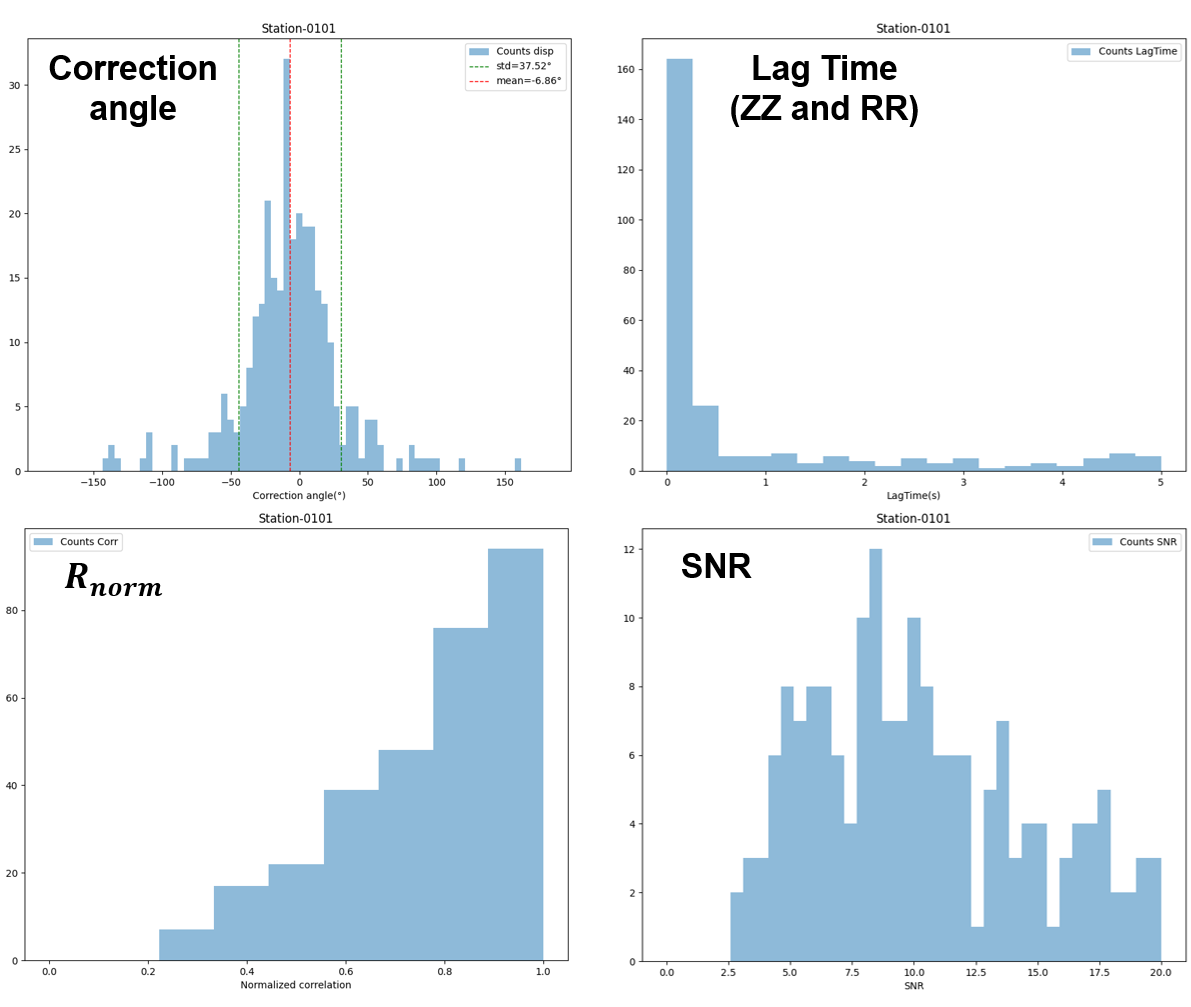
If **Rotate** = 1, the data (T-T component CCFs) that passed the QC process will be rotated based on the optimal correction angle and stored in *EXAMPLE/OUTPUT/rotateData*. 

Figure 8. Histogram of the QC parameters.

5 Details of the Code

5.1 Data Read

This section explains which parts of the code need to be modified to read the data if its format differs from the reference format.

The process of loading the stations also requires editing if your CCFs file naming format differs from the example.

