# GNSS_Vel_95CI.py: A Python Module for Calculating the Uncertainty of GNSS-Derived Site Velocity

Brendan Cornelison[1] and Guoquan Wang, M.ASCE[2]

**Abstract:** GNSS_Vel_95CI.py is an open-source Python-3 module for calculating the 95% confidence interval (95% CI) for the site velocity derived from global navigation satellite systems (GNSS) daily positions, which are often affected by time-correlated noises. The detailed methodology for calculating the 95% CI is documented in a recent article (Wang 2022) and initially programmed in Fortran. However, few young researchers (e.g., graduate students) are familiar with Fortran now. In an effort to support a broader user community, we have realized the method in the Python programming language, which has been commonly taught in current college curriculums. Through the use of this module, researchers and engineers can focus on the applications of GNSS time series, rather than on coding and data processing. In particular, the module has the option to output the autocorrelation function (ACF) and the GNSS time-series decomposition results: the linear, nonlinear, seasonal, and residual components, which allow students and researchers having little coding experience to conduct advanced GNSS time-series analysis. The module is versatile, easy to install through the pip installer and GitHub, and simple to use. An example Python program is provided to illustrate the use of this module. **DOI:** **10.1061/(ASCE)SU.1943-5428.0000410.** © *2022 American Society of Civil Engineers.*

**Author keywords:** 95% confidence interval (95% CI); Decomposition; Global navigation satellite systems (GNSS); Python; Site velocity; Time series; Uncertainty.

## Motivation

Site velocities are the primary products of long-term global navigation satellite systems (GNSS) monitoring projects. A site velocity must be used with a full understanding of its uncertainty. The uncertainty is important for long-term site-stability assessment and structural health monitoring (SHM) projects with site velocities (or deformation rates) at a few millimeters per year. There are sophisticated mathematical methods for calculating the linear trend and its uncertainty for stationary time series in statistics. A stationary time series has statistical properties that do not change over time. However, GNSS time series often exhibit nonstationary behaviors, such as nonlinear motions, seasonal motions, and random walks. Consequently, the uncertainty estimated with the conventional method is unrealistically small. In practice, different scaling and/or noise-modeling approaches have been applied by different research groups to correct the uncertainty. Thus, the geodetic literature documents considerably different uncertainty estimates, which have caused confusion among GNSS data users in research and engineering, although the velocity estimates from different research groups are similar. The geodesy research and engineering communities have been struggling for decades to develop a consistent and robust method for quantifying the uncertainties of GNSS-derived site velocities. A detailed review of current approaches to assessing the uncertainty is found in Wang (2022).

Wang (2022) published an analytical methodology for determining the uncertainty, that is, the 95% confidence interval (95% CI), of GNSS-derived site velocities. The method decomposes the GNSS time series into four components: a linear component, a nonlinear component, a seasonal component, and residuals. The determination of 95% CI accounts for the linear trend of the nonlinear component, the linear trend of the seasonal component, and the autocorrelation of the residuals. An effective sample size ($N_{eff}$) rather than the total sample size ($N$) is used to calculate the standard error of the site velocity, which is the major component of the 95% CI. Two empirical formulas have been established for estimating the 95% CI of horizontal and vertical site velocities based on the time span of the GNSS time series.

The Wang (2022) methodology was initially realized in a Fortran program (Fortran77), which calls for numerous Fortran subroutines. While Fortran is the first high-level computing language, it is now rarely taught in universities. Consequently, few young researchers (e.g., graduate students) are familiar with Fortran now. Instead, Python has become the most popular and powerful general-purpose programming language since its release in 1991, and has been commonly offered in college curriculums. To support a broader user community, we have implemented the methodology of calculating the 95% CI into a Python module, *GNSS_Vel_95CI.py*. The Python language simplifies the overall programming process and increases the code's readability.

## Software Description

The module is designed as a single Python function that reads in GNSS time series and calculates its site velocity and 95% CI according to the methodology presented in Wang (2022). The module

[1]Ph.D. Candidate, Dept. of Earth and Atmospheric Sciences, Univ. of Houston, Houston, TX 77204. Email: bscornelison@uh.edu

[2]Professor, Dept. of Earth and Atmospheric Sciences, Univ. of Houston, Houston, TX 77204 (corresponding author). ORCID: https://orcid.org/0000-0003-3731-3839. Email: gwang@uh.edu

Note. This manuscript was submitted on December 10, 2021; approved on June 15, 2022; published online on September 16, 2022. Discussion period open until February 16, 2023; separate discussions must be submitted for individual papers. This technical note is part of the ***Journal of Surveying Engineering***, © ASCE, ISSN 0733-9453.

is built on Python common libraries, such as pandas, numpy, math, statsmodels, and matplotlib. The core process is the decomposition of the original position time series ($y_i$), as follows:

$$y_i = L_i + NL_i + S_i + r_i \tag{1}$$

where $L_i$ = linear component; $NL_i$ = nonlinear component; $S_i$ = seasonal component; and $r_i$ = residuals. The final 95% CI is determined according to the following formula:

$$b_{95\%CI} \approx 1.96 \times SE_{bc} + |b_{NL}| + |b_S| \tag{2}$$

where $SE_{bc}$ = corrected standard error of the slope obtained from an ordinary linear regression on the GNSS time series; and $b_{NL}$ and $b_S$ = slopes of the nonlinear ($NL_i$) and seasonal ($S_i$) components, respectively.

GNSS time series are often biased by outliers and steps. Preprocessing is needed to remove obvious outliers and steps before conducting the decomposition analysis. The authors have developed Python modules for identifying and removing outliers and steps superimposed into the GNSS time series. A detailed review of the methods for handling the outliers and steps is found in Wang et al. (2022).

The module framework is developed according to the time series decomposition described in Eq. (1), which comprises the following five main steps:

Step 1: Apply an ordinary linear regression to the GNSS time series ($y_i$) and build the linear component: $L_i = a + b_L t_i$, where $b_L$ represents the slope of the linear regression, known as the site velocity ($b_L$). The Python module "*OLS*" from the statsmodels library is used for the linear regression (OLS 2021). The GNSS time series is detrended to get the residual time series, $R_i = y_i - L_i$. The standard error ($SE_b$) of $b_L$ is calculated using the conventional method for stationary time-series analysis

$$SE_b = \sqrt{\frac{\sum_{i=1}^{N} R_i^2}{N(N-2)}} \times \frac{1}{\sigma_t} \tag{3}$$

where $\sigma_t$ = standard deviation of the observation days ($t_i$) in decimal years.

Step 2: Apply locally weighted scatterplot smoothing (LOWESS) to the delinear-trended time series ($R_i$). The smoothed time series is known as the nonlinear component ($NL_i$) of the GNSS time series. The "*LOWESS*" function from the statsmodels library is imported into the module; "*frac*" is an input for determining the fraction of the data used for the local regression, which controls the smoothness of the curve; and "*it*" is an input for determining the number of residual-based reweightings to perform (LOWESS 2021). In our practice for processing GNSS data in the Houston, Texas, region, "*frac*" is set as 0.4, and "*it*" is set as 2. Users may use slightly different parameters for their specific datasets. An ordinary linear regression is applied to the nonlinear component for calculating its slope $b_{NL}$.

Step 3: Determine the seasonal component ($S_i$) as follows:

$$\begin{aligned} S_i = c_0 &+ c_1 \cos(2\pi \times (t_i - t_0)) + d_1 \sin(2\pi \times (t_i - t_0)) \\ &+ c_2 \cos(4\pi \times (t_i - t_0)) + d_2 \sin(4\pi \times (t_i - t_0)) \end{aligned} \tag{4}$$

where $t_i$ = date series with a unit in decimal years (e.g., 2014.3956, 2014.3984, 2014.4011, ...); and $t_0$ = initial date (e.g., 2014.3956). The coefficients $c_1$, $d_1$, $c_2$, and $d_2$ are calculated using the fast Fourier transform (FFT) method, computing the discrete Fourier transform coefficients of the residual time series ($Res_i = y_i - L_i - NL_i$); $c_1$ and $d_1$ are the Fourier coefficients of the 1-year-period signals; $c_2$ and $d_2$ are the Fourier coefficients of the half-year-period signals. The peak-to-trough amplitude of the annual signal can be estimated by $p_1 = \sqrt{c_1^2 + d_1^2}$; the peak-to-trough amplitude of the semiannual signal can be estimated by $p_2 = \sqrt{c_2^2 + d_2^2}$. The peak-to-trough amplitude of the seasonal motions can be estimated by $P = 2 \times \sqrt{p_1^2 + p_2^2}$. The detailed method for calculating the annual and half-annual seasonal parameters is coded in a Python function. Users may read the source code at our GitHub site (Cornelison and Wang 2021a) for details. It should be noted that the function only uses the integer years (e.g., 7 years, not 7.6 years) of the residuals ($Res_i$) in calculating the seasonal parameters.

The FFT requires a continuous time series. There are often data gaps from a few days to a few months in GNSS time series. The "*resample*" module in the pandas library is used to resample the date series (fill date gaps) and fill each data gap in the residual time series ($Res_i$) as a 'NaN' (Resample 2021). The "random.choice" module in the numpy library is used to replace each data gap (NaN) with a random number borrowed from the residual time series (Random 2021). In the original methodology presented in Wang (2022), we used the "hot-deck imputation" to fill in the missing data, which borrows (in a random way) information from a set of nearby observations to fill the missing information. We found that the "random.choice" module results in very similar filling data with the "hot-deck imputation." Both methods work well for small gaps (a few days to a few months) and poorly for large gaps (several months to a few years). The effects of data gaps on the estimation of 95% CI are discussed in Wang (2022).

Step 4: Determine the effective sample size ($N_{eff}$) based on the analysis of autocorrelation of the residuals, $r_i = y_i - L_i - NL_i - S_i$. For GNSS measurements, the position of each day is partially correlated with the position-values of the previous and following days. Therefore, there are actually fewer independent measurements contributing to the standard error ($SE_b$) of the estimated slope ($b_L$). In theory, only the number of independent samples should be used to calculate the standard error. "Effective sample size" ($N_{eff}$) is a term used in statistics to represent the independent samples among autocorrelated time series. $N_{eff}$ is obtained from a high-order autoregressive model developed from the autocorrelation function (ACF) of the residual time series ($r_i$). ACF describes how well the present value of the series is related to its past values. The "*acf*" module in the statsmodels library is used for calculating ACF (ACF 2021). The detailed method for calculating $N_{eff}$ is documented in Wang (2022). The standard error $SE_b$ [Eq. (3)] is corrected according to the effective sample size as follows:

$$SE_{bc} \approx \sqrt{\frac{N}{N_{eff}}} \times SE_b \tag{5}$$

where $N$ = original sample size.

Step 5: Determine the final 95% CI according to Eq. (2). To provide a useful GNSS data analysis tool for GNSS data users, the module also outputs the ACF, and the decomposition results as figures and ASCII text files.

## Illustrative Example

The module reads GNSS time series and returns site velocity ($b_L$) and its uncertainty (95% CI). Importing the module "*GNSS_Vel_95CI*" will enable a Python program to use the module. GNSS measurements often comprise three components: north-south (NS), east-west (EW), and up-down (UD). Users (or callers) need to call the module for each component in their Python program. Here is a brief example of using the module in a Python program, Main_cal_95CI.py:

```
#! /usr/bin/python3
import os
import pandas as pd
from GNSS_Vel_95CI import cal_95CI
# An example of 'fin': MRHK_GOM20_neu_cm.col
# Decimal-Year NS(cm) EW(cm) UD(cm) sigma-NS(cm) sigma-
EW(cm) sigma-UD(cm)
# 2014.3956  0.2037 0.0395 -0.0516 0.0548 0.0255 0.0605
# 2014.3984  0.1714 0.0465 -0.7940 0.0542 0.0241 0.0593
# 2014.4011 -0.1916 0.0269 -0.2318 0.0538 0.0246 0.0591
# 2014.4038 -0.2069 0.0400  1.0380 0.0517 0.0231 0.0566
# . . . . . .
directory = './'
for fin in os.listdir(directory): # Put all GNSS data files under this
directory
  if fin.endswith("neu_cm.col"):
    GNSS = fin[0:4] # Station name, e.g., MRHK
    ts_enu = []
    ts_enu = pd.read_csv (fin, header=0, delim_whitespace=True)
    year = ts_enu.iloc[:,0] # Decimal year, 2014.3596,
    2014.3984, . . .

    dis = ts_enu.iloc[:,1] # NS
    result_NS=cal_95CI(year,dis,GNSS,DIR='NS',output='on',
    pltshow='on')
    b_NS=round(result_NS[0],2) # Site velocity
    b_NS_95CI=round(result_NS[1],2) # The 95%CI of the site
    velocity

    dis = ts_enu.iloc[:,2] # EW
    result_EW=cal_95CI(year,dis,GNSS,DIR='EW',output='on',
    pltshow='on')
    b_EW=round(result_EW[0],2)
    b_EW_95CI=round(result_EW[1],2)

    dis = ts_enu.iloc[:,3] # UD
    result_UD=cal_95CI(year,dis,GNSS,DIR='UD',output='on',
    pltshow='on')
    b_UD=round(result_UD[0],2)
    b_UD_95CI=round(result_UD[1],2)

  else:
    pass
```

Two options are included in the inputs of the module: (1) pltshow = 'on' or 'off', enable or disable displaying the ACF versus time-lag plot and the decomposition plot at the end of this process; and (2) output='on' or 'off', enable or disable outputting the ACF and decomposition data. The module outputs two figures and four ASCII text files if the user sets output = on. The two figures are an ACF plot (GNSS_DIR_ACF.pdf) and a decomposition plot (GNSS_DIR_Decomposition.pdf). "GNSS" represents the name of the GNSS station (e.g., MRHK). "DIR" represents the direction of the time series, such as NS, EW, or UD. Both GNSS and DIR are the inputs from callers. The four text files are: (1) GNSS_DIR_ACF.txt, which comprises the time-lag and ACF; (2) GNSS_DIR_Linear_Nonlinear.txt, which comprises the columns of date (decimal year), the original displacement, the linear trend, and the nonlinear trend; (3) GNSS_DIR_SeasonalM.txt, which comprises the columns of continuous date (gap filled), the delinear-nonlinear-trended time series ($Res_i$), the seasonal model ($S_i$), and the final residuals ($r_i$); and (4) GNSS_DIR_AllParameters.txt, which comprises all parameters involved in calculating the 95% CI. Users who need to process a massive number of GNSS files with the sole purpose of getting site velocities and their 95% CI may set "output" and "pltshow" as off.

Fig. 1 illustrates the ACF versus the time lag for the residual time series ($r_i$) of the vertical displacements (UD) at GNSS station MRHK. ACF is used to determine the effective sample size ($N_{eff}$) in this module. MRHK is a continuous GNSS station located in Katy, Texas, where moderate land subsidence (~1.5 to 2 cm/year) is ongoing (Agudelo et al. 2020).

Fig. 2 illustrates the decomposition plot of GNSS time series at MRHK (UD component). Fig. 2(a) is the linear component; Fig. 2(b) is the nonlinear component; Fig. 2(c) is the seasonal component; and Fig. 2(d) is the residuals. The final site velocity and its 95% CI are marked in Fig. 2(a). For the UD component, the calculated site velocity is $-17.21 \pm 0.56$ mm/year over the observation period from 2014 to 2021. Wang (2022) also provides empirical formulas for projecting the 95% CI of site velocities solely depending on the length of the observation period ($T$) in years. The formulas are developed based on 9,700 globally distributed GNSS stations, representing an average 95% CI for the datasets with a similar year range. The global datasets show that $95\%CI = 1.8/T$ for the horizontal components (NS and EW) and $95\%CI = 5.2/T^{1.25}$ for the vertical component (UD). The projected 95% CI is in millimeters per year (mm/year). A comparison of the calculated 95% CI value and the projected one is also marked in Fig. 2(a). For the UD component of MRHK, the projected 95% CI is 0.43 mm/year, which is slightly smaller than the calculated one, 0.56 mm/year. The comparison suggests that this site is slightly noisier than the global average. This site is located in an area experiencing long-term groundwater pumping (Wang et al. 2022). The fluctuations of groundwater levels caused significant irregular ground surface oscillations; as a consequence, the calculated 95% CI is larger than the global average. The estimates of $SE_b$ and $SE_{bc}$ are marked in Fig. 2(a). The estimates of $b_{NL}$ and $b_S$ are marked in Figs. 2(b and c), respectively. The seasonal model [Eq. (4)] for this site is marked in Fig. 2(c). The peak-to-trough amplitude of the seasonal movements is also marked in Fig. 2(c). The root mean squares (RMS) of the delinear-trended time series, de-nonlinear-trended time series, and the residuals are marked in Figs. 2(b–d), respectively. There are two data gaps in late 2015 and late 2016, as shown in Figs. 2(a and b). These gaps were filled before determining the seasonal coefficients. Figs. 2(c and d) depict the seasonal and residual time series that gaps have been filled. Visually, there is no remarkable difference between the filled and the original portions.

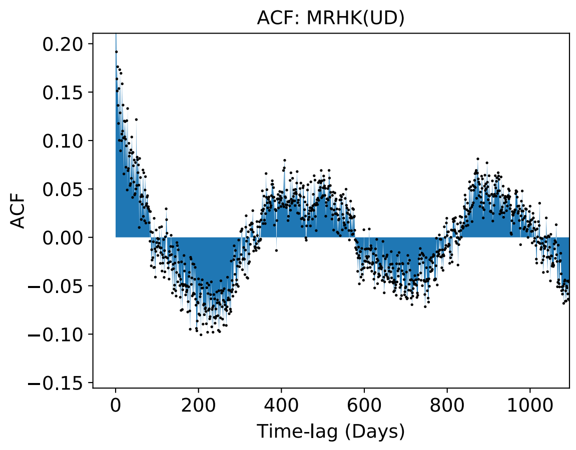

**Fig. 1.** Example of autocorrelation function (ACF) versus time-lag plot output from the Python module.

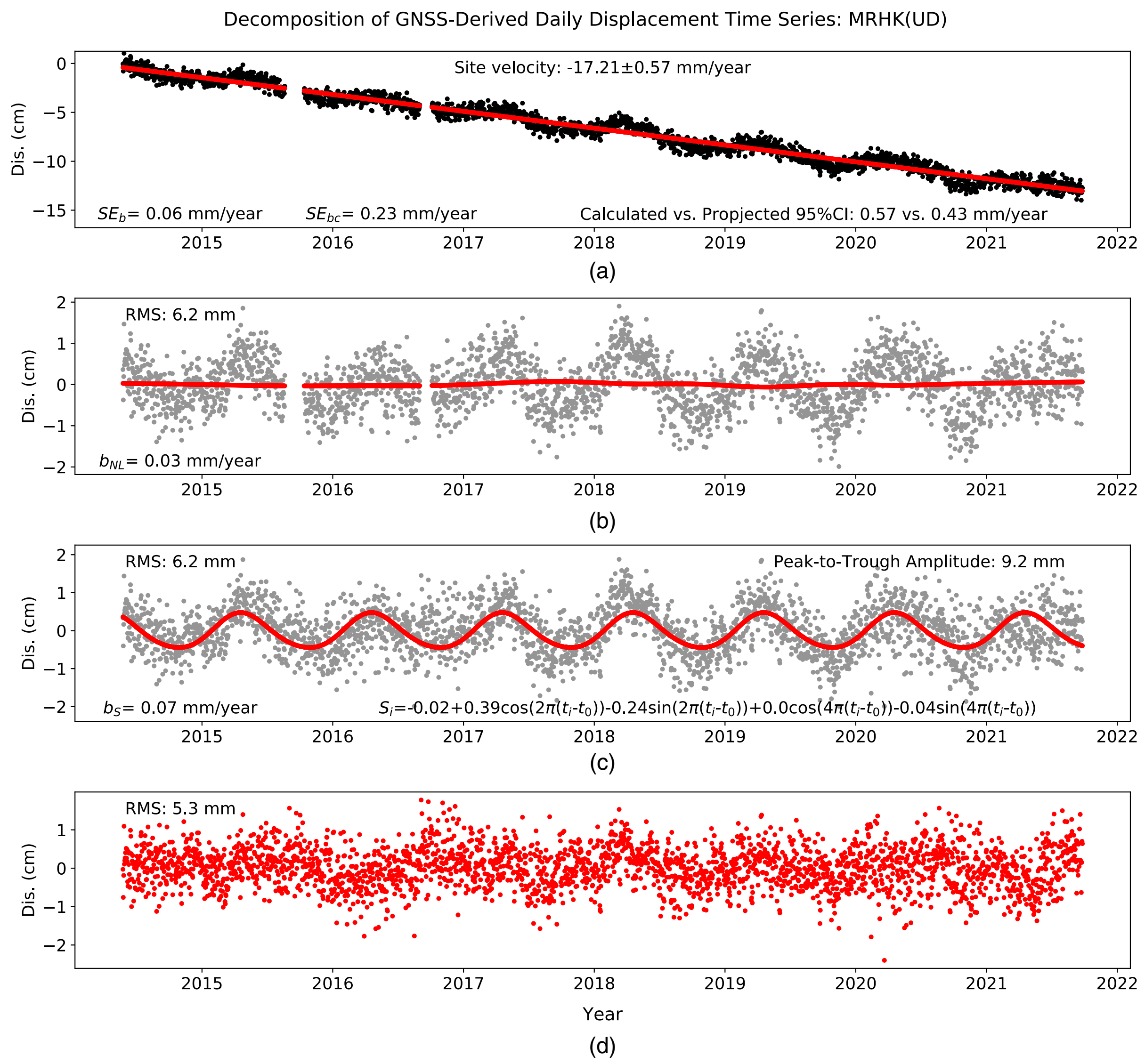

**Fig. 2.** Example of the decomposition plot output from the Python module: (a) displacements and linear component; (b) nonlinear component; (c) seasonal component; and (d) residuals.

## Conclusions and Impacts

The Python module was developed to make it easier for the geodesy education, research, and engineering communities to calculate the uncertainty (95% CI) of GNSS-derived site velocities. The module is versatile, easy to install, and simple to use. We have tested the module on computers with different operating systems (Microsoft Windows, Apple macOS, Linux). An example Python program for calling the module is provided in this article. The module is released under the GNU Public License (version 3). The source code is hosted on GitHub (Cornelison and Wang 2021a). Python wheels are provided on the Python Package Index (PyPi) website for easy installation with the Package Installer for Python (pip), using "pip install GNSS_Vel_95CI" (Cornelison and Wang 2021b).

The module outputs both the calculated (site specific) and projected (global average) 95% CI estimates, as shown in Fig. 2(a). If the calculated 95% CI is significantly larger than the project one, this site is likely noisy or experienced significant nonlinear motions. Thus, the module shows promise for automatically sorting well-behaved and poorly behaved GNSS datasets from large GNSS networks. We have used this module for quality control (QC) of the Houston GNSS Network (HoustonNet) data (Wang et al. 2022). The module also outputs the ACF and the decomposition results, which are fundamental for GNSS data analysis. With the presented module, researchers and engineers are able to focus on the application of GNSS time series, rather than spend excessive time on coding and data processing. As of spring 2022, the module has been used by the second author, G. Wang, in his geodesy classes at the University of Houston, where it was found to be a useful tool in introducing students to computer programming and facilitating large GNSS dataset analysis on their laptops. Considering the broad applications of GNSS time series in science and engineering and the popularity of Python as a programming language, the module has the potential for broad applications in education, earth sciences, and civil engineering.

## Data Availability Statement

All data, models, or code that support the findings of this study are available from the corresponding author (gwang@uh.edu) upon reasonable request. The source code generated or used during the study is available in an online GitHub repository, https://github.com/bob-Github-2020/GNSS_Vel_95CI. The module can be installed on users' computers through the Package Installer for Python (https://pypi.org/project/GNSS-VEL-95CI), using "pip install GNSS_Vel_95CI".

## Acknowledgments

We thank Dr. Linqiang Yang (University of Hawaii at Manoa), Dr. Lin Xiong (East Carolina University), Dr. Hanlin Liu (Shenzhen University), and many graduate students at the University of Houston for testing the module on their computers.

## References

ACF (Auto-Correlation Function). 2021. "statsmodels.tsa.stattools.acf." Accessed May 8, 2022. https://www.statsmodels.org/devel/generated/statsmodels.tsa.stattools.acf.html.

Agudelo, G., G. Wang, Y. Liu, Y. Bao, and M. J. Turco. 2020. "GPS geodetic infrastructure for subsidence and fault monitoring in Houston, Texas, USA." *Proc. IAHS* 382 (Apr): 11–18. https://doi.org/10.5194/piahs-97-1-2020.

Cornelison, B., and G. Wang. 2021a. "GNSS_Vel_95%CI." Accessed May 8, 2022. https://github.com/bob-Github-2020/GNSS_Vel_95CI.

Cornelison, B., and G. Wang. 2021b. "GNSS-Vel-95CI, pip install GNSS-Vel-95CI." Accessed May 8, 2022. https://pypi.org/project/GNSS-Vel-95CI.

LOWESS (Locally Weighted Scatterplot Smoothing). 2021. "statsmodels.nonparametric.smoothers_lowess.lowess." Accessed May 8, 2022. https://www.statsmodels.org/dev/_modules/statsmodels/nonparametric/smoothers_lowess.html.

OLS (Ordinary Least Squares). 2021. "statsmodels.regression.linear_model.OLS." Accessed May 8, 2022. https://www.statsmodels.org/dev/generated/statsmodels.regression.linear_model.OLS.html.

Random. 2021. "numpy.random.choice." Accessed May 8, 2022. https://numpy.org/doc/stable/reference/random/generated/numpy.random.choice.html.

Resample. 2021. "pandas.DataFrame.resample." Accessed May 8, 2022. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.resample.html.

Wang, G. 2022. "The 95% confidence interval for the GNSS-derived site velocities." *J. Surv. Eng.* 148 (1): 04021030. https://doi.org/10.1061/(ASCE)SU.1943-5428.0000390.

Wang, G., A. Greuter, C. M. Petersen, and M. J. Turco. 2022. "Houston GNSS network for subsidence and faulting monitoring: Data analysis methods and products." *J. Surv. Eng.* 148 (4): 04022008. https://doi.org/10.1061/(ASCE)SU.1943-5428.0000399.