OPEN

A novel Python module for statistical analysis of turbulence (P-SAT) in geophysical flows

Mayank Agarwal¹, Vishal Deshpande^{2✉}, David Katoshevski³ & Bimlesh Kumar⁴

We present Python Statistical Analysis of Turbulence (P-SAT), a lightweight, Python framework that can automate the process of parsing, filtering, computation of various turbulent statistics, spectra computation for steady flows. P-SAT framework is capable to work with single as well as on batch inputs. The framework quickly filters the raw velocity data using various methods like velocity correlation, signal-to-noise ratio (SNR), and acceleration thresholding method in order to de-spike the velocity signal of steady flows. It is flexible enough to provide default threshold values in methods like correlation, SNR, acceleration thresholding and also provide the end user with an option to provide a user defined value. The framework generates a .csv file at the end of the execution, which contains various turbulent parameters mentioned earlier. The P-SAT framework can handle velocity time series of steady flows as well as unsteady flows. The P-SAT framework is capable to obtain mean velocities from instantaneous velocities of unsteady flows by using Fourier-component based averaging method. Since P-SAT framework is developed using Python, it can be deployed and executed across the widely used operating systems. The GitHub link for the P-SAT framework is: <https://github.com/mayank265/flume.git>.

We have developed Python-Statistical Analysis of Turbulence (P-SAT), an open-source, lightweight Python framework that can de-spike (identify spikes and replace them) the raw velocity time series data obtained from an acoustic Doppler velocimeter (ADV) device using various filtering methods. The P-SAT framework also computes a range of turbulent statistics like mean velocities, variance, skewness, kurtosis, Reynolds stresses, third order correlations, 2D as well as 3D fluxes of turbulent kinetic energy, turbulent kinetic energy dissipation, conditional statistics of quadrants, octants and their corresponding probabilities, spectrum of a given signal etc. The P-SAT framework also exports all the turbulent statistics into a .csv so that it can be used in future for analysis.

Authors of the present study believe that the P-SAT framework would be of a significant help to those researchers who are associated with research involving statistical analyses of the turbulent flows in steady as well as in unsteady flow environments. The motivation behind development of the P-SAT framework was the authors' experience on working with various third party tools and utilities that are commercially available in the scientific community. The authors found few major issues that led to the development of P-SAT framework : (1) To the best of our knowledge, there is no single tool which can handle instantaneous velocity signals of steady flows and unsteady flows together, and is able to compute majority of the turbulent statistics that the P-SAT framework computes.

As working with the turbulent flows involves complex computations (such as: calculating mean velocities in study/unsteady flow environments, Reynolds shear stress (RSS) calculations, turbulent kinetic energy (TKE) calculations, finding third order moments of velocity fluctuations, TKE dissipation, calculations of contributions towards total RSS production from different quadrants, octant analysis and determination of the octant probabilities, calculations of spectral density functions of velocities etc.), it becomes cumbersome to compute these quantities in *Microsoft Excel* or write custom scripts for the same. (2) Support for batch processing of files: Experiments performed on a laboratory flume often consists of multiple readings at multiple points thus requiring a tool to work on batch of input files. (3) Commercial tools are expensive to use and are often closed source. Also, the flexibility provided to the end users is often limited.

¹Department of Computer Science and Engineering, Indian Institute of Technology Patna, Patna, India. ²Department of Civil and Environmental Engineering, Indian Institute of Technology Patna, Patna, India. ³Department of Civil and Environmental Engineering, Ben-Gurion University of the Negev, Beer-Sheva, Israel. ⁴Department of Civil Engineering, Indian Institute of Technology Guwahati, Guwahati, India. ✉email: vishal0092@gmail.com

The P-SAT framework is capable of handling all of the above issues. It provides a rich turbulent statistics for the given set of input files. It is an open source tool developed in Python with the source code available on GitHub for the scientific community to use it and modify it. We have chosen Python for developing P-SAT because Python has been widely adopted in the recent years because of the availability of various libraries and extensive community support. Many modules for varying applications like modeling, analysis, and optimization of electric power systems¹, SciPy², Gene Ontology analyses³, ice sheet models⁴, machine learning⁵, data logger for coupled fluid-structure simulations⁶, have used Python exclusively and made available to the scientific community.

The authors have been studying fluvial geomorphology and transport behavior of sediments (erosion/deposition) in alluvial channels, where turbulence plays as a key parameter in various fluvial environments such as: flow in a channel with curvilinear cross-section⁷, flows over bedforms⁸, flows with vegetation⁹, flows around bridge piers¹⁰, and flows in sand mined channels¹¹. For the present study, we carried out extensive experiments in a laboratory flume setup at Department of Civil Engineering, Indian Institute of Technology Guwahati. The experimental setup is shown in Fig. 1 and more details regarding the experiments and measurements are explained in Sect. 3. Instantaneous velocity readings along the vertical plane were taken using a four-beam, down-looking, Vectrino+ acoustic Doppler velocimeter (ADV) probe manufactured by Nortek. Measurements were carried out on multiple heights, and at each height, 30,000 samples were collected for 5 min (i.e., sampling rate of 100 Hz).

ADVs demonstrate a proven technology for capturing 3D velocity signals in environmental flows e.g.,^{12–15}. However, the raw velocity signals captured by ADVs may contain spikes caused by aliasing of the Doppler signal. Another issue with the raw signal maybe the Doppler noise floor also. To clean the velocity record from the previously mentioned issues, post-processing of the raw velocity signal may be necessary. These spikes have an adverse effect on the turbulent statistics that are computed and hence, they must be removed from the time series data collected by the ADV. The process of removal of spikes is known as de-spiking and there are various approaches for the same¹⁶. The P-SAT framework implements three de-spiking approaches: velocity correlation, SNR and acceleration thresholding method.

The paper is organized as follows: Sect. 2 consists of definitions of various statistical parameters of turbulence for the steady flow. It also provides an insight into the computation of the mean velocity component from the instantaneous velocities for highly unsteady flows. Section 3 describes the experimental setup in details and provides the various measurements involved. The P-SAT framework is described in Sect. 4. It details out all the requirements for the P-SAT framework, the data format, execution steps, and the parameters computed by the P-SAT framework. Section 5 provides an insight into the discussion about the P-SAT framework. Section 6 concludes the paper and provides a possible future course of work for the P-SAT framework. We also include an appendix section where more information about the individual Python files that are used along with the files that are created by the P-SAT framework are described. This would be useful for the developers who wish to enhance the P-SAT framework and add features to it.

Theoretical foundations

This section has been divided into two subsections: In Sect. 2.1, definitions of various statistical parameters of turbulence for the steady flow are presented. In Sect. 2.2, calculation of the mean velocity component from the instantaneous velocities for highly unsteady flow is presented.

Definitions of various statistical parameters of turbulence for steady flow. Since the tool calculates several statistical parameters from raw data, it is better if these statistical parameters are defined first. All the notation relating to the statistical parameters are depicted in Table 1. Time-averaged streamwise (U), lateral (V), and vertical (W) velocities have been calculated as:

$$U = \frac{1}{n} \sum_{i=1}^n U_i \quad (1)$$

$$V = \frac{1}{n} \sum_{i=1}^n V_i \quad (2)$$

$$W = \frac{1}{n} \sum_{i=1}^n W_i \quad (3)$$

where, U_i , V_i , and W_i are the instantaneous velocities in the streamwise, lateral and vertical directions, respectively, and n is the number of samples taken. Velocity variance for all three components of the velocity can be given by:

$$u_var = \overline{u'^2} = \frac{1}{n} \sum_{i=1}^n u_i'^2 \quad (4)$$

$$v_var = \overline{v'^2} = \frac{1}{n} \sum_{i=1}^n v_i'^2 \quad (5)$$

f_{cutoff}	Cut-off frequency
f_{TKEu}	Flux of the turbulent kinetic energy in streamwise direction
f_{TKEw}	Flux of the turbulent kinetic energy in vertical direction
$2DF_{TKEu}$	(Non-dimensional) 2D Flux of the turbulent kinetic energy in streamwise direction
$2Df_{TKEu}$	2D flux of the turbulent kinetic energy in streamwise direction
$2DF_{TKEw}$	(Non-dimensional) $2Df_{TKEw}$
$2Df_{TKEw}$	2D flux of the turbulent kinetic energy in vertical direction
$3DF_{TKEu}$	(Non-dimensional) $3Df_{TKEu}$
$3Df_{TKEu}$	3D flux of the turbulent kinetic energy in streamwise direction
$3DF_{TKEw}$	(Non-dimensional) $3Df_{TKEw}$
$3Df_{TKEw}$	3D flux of the turbulent kinetic energy in vertical direction
d_{50}	Median particle diameter
$\delta_{i,H}$	Indicator function for quadrant analysis
e	Dissipation of the turbulent kinetic energy
ED	(Non-dimensional) e
g	Gravitational constant
h	Flow depth
H	Hole size
k	Fourier component
λ_a	Parameter for acceleration thresholding method
$M_{30}, M_{03}, M_{12}, M_{21}$	Third order moments of velocity
n	Number of samples
Q	Flow discharge
Re	Reynolds number
ρ_{water}	Density of water
$S_{i,H}$	Fractional contribution towards RSS production
σ_g	Gradation coefficient for sand
$\sigma_u, \sigma_v, \sigma_w$	Standard deviation of velocity in the streamwise, lateral, and vertical directions, respectively
τ_{uw}, τ_{uv} , and τ_{vw}	Reynolds shear stresses
T	Time period of measurement
u', v', w'	Fluctuating components of streamwise, lateral and vertical velocity, respectively
U, V, W	Time-averaged streamwise, lateral and vertical velocity, resp
U_{uf}	Instantaneous velocities
U_*	Shear velocity
$\overline{U_{uf}}$	Mean velocity component for unsteady flow
U_i, V_i, W_i	Instantaneous streamwise, lateral and vertical velocity, resp.
φ^o	Angle of repose

Table 1. List of symbols.

$$w_{var} = \overline{w'^2} = \frac{1}{n} \sum_{i=1}^n w'^2 \quad (6)$$

where u' , v' and w' are the fluctuating components of velocities in the streamwise, lateral, and vertical directions, respectively. The square root of the velocity variance is the standard deviation. The shape of the probability density function is illustrated by its skewness. The skewness also depicts the relative contribution of positive and negative velocity fluctuations to the formation of the velocity pattern. Skewness for the velocities in all three directions can be given by:

$$u_{skew} = \left(\overline{u'^2} \right)^{-3/2} \frac{1}{n} \sum_{i=1}^n u'^3 \quad (7)$$

$$v_{skew} = \left(\overline{v'^2} \right)^{-3/2} \frac{1}{n} \sum_{i=1}^n v'^3 \quad (8)$$

$$w_skew = \left(\overline{w'^2}\right)^{-3/2} \frac{1}{n} \sum_{i=1}^n w'^3 \quad (9)$$

Kurtosis is a statistical measure that defines how heavily the tails of a distribution differ from the tails of a normal distribution. In other words, kurtosis identifies whether the tails of a given distribution contain extreme values. Kurtosis of the probability distribution of a real-valued random variable (3D velocity in the present case) is given by:

$$u_kurt = \left(\overline{u'^2}\right)^{-2} \frac{1}{n} \sum_{i=1}^n u'^4 \quad (10)$$

$$v_kurt = \left(\overline{v'^2}\right)^{-2} \frac{1}{n} \sum_{i=1}^n v'^4 \quad (11)$$

$$w_kurt = \left(\overline{w'^2}\right)^{-2} \frac{1}{n} \sum_{i=1}^n w'^4 \quad (12)$$

Reynolds stresses furnish very important information about the transfer of momentum in turbulent flows. Reynolds stresses are the components of a symmetric second order tensor where the diagonal components are called the Reynolds normal stresses (RNS) and the off diagonal components are called the Reynolds shear stresses (RSS)¹⁷. Reynolds shear stresses (τ_{uw} , τ_{uv} , and τ_{vw}) can be calculated as:

$$\tau_{uw} = -\rho_{water} \overline{u'w'} \quad (13)$$

$$\overline{u'w'} = \frac{1}{n} \sum_{i=1}^n (U_i - U)(W_i - W) \quad (14)$$

$$\tau_{uv} = -\rho_{water} \overline{u'v'} \quad (15)$$

$$\overline{u'v'} = \frac{1}{n} \sum_{i=1}^n (U_i - U)(V_i - V) \quad (16)$$

$$\tau_{vw} = -\rho_{water} \overline{v'w'} \quad (17)$$

$$\overline{v'w'} = \frac{1}{n} \sum_{i=1}^n (V_i - V)(W_i - W) \quad (18)$$

where, ρ_{water} is the density of water. The degree of flow anisotropy is measured by the ratio, σ_w/σ_u and can be given by following expression:

$$Anisotropy = \frac{\sigma_w}{\sigma_u} = \frac{\sqrt{\overline{w'w'}}}{\sqrt{\overline{u'u'}}} \quad (19)$$

where, σ_w and σ_u are the standard deviations of the vertical and streamwise velocities, respectively.

The third statistical moments or the skewness indicate non-symmetric distributions. Zero skewness indicate symmetric distribution about the mean or the Gaussian distribution, while negative and positive values of skewness show that the distribution is skewed towards left and right, respectively, around the mean. According to Raupach¹⁸, the third moments of velocity fluctuations, $M_{jk} = \overline{\hat{u}^j \hat{w}^k}$, where $j + k = 3$, and $\hat{u} = \frac{u}{(\overline{u'u'})^{0.5}}$,

$\hat{w} = \frac{w'}{(\overline{w'w'})^{0.5}}$ can be expressed as:

$$M_{30} = \frac{\overline{u'^3}}{(\overline{u'u'})^{1.5}} \quad (20)$$

$$M_{03} = \frac{\overline{w'^3}}{(\overline{w'w'})^{1.5}} \quad (21)$$

$$M_{21} = \frac{\overline{u'^2 w'}}{(\overline{u'u'}) (\overline{w'w'})^{0.5}} \quad (22)$$

Octant name	U'	V'	W'
-1	>0	>0	<0
+1	>0	>0	>0
-2	<0	<0	>0
+2	<0	>0	>0
-3	<0	<0	<0
+3	<0	>0	<0
-4	>0	<0	<0
+4	>0	>0	<0

Table 2. Determination of octant.

$$M_{12} = \frac{\overline{u'w'^2}}{(\overline{u'u'})^{0.5}(\overline{w'w'})} \quad (23)$$

The total Reynolds shear stress ($-\overline{u'w'}$) at any given point is the sum of different types of bursting events. Thus, depending upon the relative sign of instantaneous values of velocity fluctuations u' and w' , the bursting events can be plotted in four different quadrants ($i = 1, 2, 3, 4$) of the (u', w') plane¹⁹ i.e., outward interactions ($i = 1 : u' > 0, w' > 0$), ejections ($i = 2 : u' < 0, w' > 0$), inward interactions ($i = 3 : u' < 0, w' < 0$), and sweeps ($i = 4 : u' > 0, w' < 0$). At any point in the flow field, the contribution to the total Reynolds shear stress through different ways of momentum transfer can be calculated as:

$$\langle u'w' \rangle_{i,H} = \lim_{T \rightarrow 0} \frac{1}{T} \int_0^T u'(t)w'(t)\delta_{i,H}[u'(t), w'(t)]dt \quad (24)$$

where angle brackets correspond to conditional averaging, T is the sampling time, and $\delta_{i,H}$ is the indicator function. The definition of the indicator function can be given as:

$$\delta_{i,H}[u'(t), w'(t)] = \begin{cases} 1, & \text{if } (u', w') \text{ are in quadrant } i \text{ and if } |u'w'| \geq H\sigma_u\sigma_w \\ 0, & \text{otherwise.} \end{cases} \quad (25)$$

where H is the parameter defined by the hyperbolic hole region²⁰ which allows the investigation of larger contribution to the total Reynolds shear stress from various quadrants. Fractional contribution to the total Reynolds shear stress from different quadrants can be defined as:

$$S_{i,H} = \frac{\langle u'w' \rangle_{i,H}}{\overline{u'w'}} \quad (26)$$

$S_{i,H}$ is negative for outward and inward interactions ($i = 1, 3$) and is positive for ejections and sweeps ($i = 2, 4$). Eq. (26) implies that for $H = 0$ when hole size disappears, sum of the fractional contributions from all the quadrants equals to unity ($\sum_{i=1}^4 S_{i,0} = 1$).

Octant analysis is advantageous when the user wishes to analyze turbulent structures with a strong three dimensionality. Octant analysis is carried out by considering all three components of velocity fluctuations (Table 2).

2D fluxes of the turbulent kinetic energy in the streamwise ($2Df_{TKEu}$) and vertical ($2Df_{TKEw}$) directions¹⁸ can be calculated as:

$$2Df_{TKEu} = \frac{3}{4}(\overline{u'^3} + \overline{u'w'^2}) \quad (27)$$

$$2Df_{TKEw} = \frac{3}{4}(\overline{u'^2w'} + \overline{w'^3}) \quad (28)$$

Further, these 2D turbulent kinetic energy fluxes have been made non-dimensional by dividing them by the shear velocity (U_*):

$$2DF_{TKEu} = 2Df_{TKEu} / U_*^3 \quad (29)$$

$$2DF_{TKEw} = 2Df_{TKEw} / U_*^3 \quad (30)$$

3D fluxes of the turbulent kinetic energy in the streamwise ($3Df_{TKEu}$) and vertical ($3Df_{TKEw}$) directions can be calculated as:

$$3Df_{TKEu} = \frac{1}{2} \left(\overline{u'^3} + \overline{u'v'^2} + \overline{u'w'^2} \right) \quad (31)$$

$$3Df_{TKEw} = \frac{1}{2} \left(\overline{u'^2w'} + \overline{v'^2w'} + \overline{w'^3} \right) \quad (32)$$

Similarly, these 3D turbulent kinetic energy fluxes can be made non-dimensional by dividing them by the shear velocity (U_*):

$$3DF_{TKEu} = 3Df_{TKEu} / U_*^3 \quad (33)$$

$$3DF_{TKEw} = 3Df_{TKEw} / U_*^3 \quad (34)$$

The turbulent kinetic energy (TKE) is defined as half the sum of the variances of the velocity components and can be calculated as:

$$TKE = \frac{1}{2} \left(\overline{u'u'} + \overline{v'v'} + \overline{w'w'} \right) \quad (35)$$

Dissipation of the turbulent kinetic energy (e) and its non-dimensional form (ED) can be calculated by the following expressions:

$$\text{Turbulent kinetic energy dissipation } e = \frac{15\nu}{U^2} \left(\frac{\partial u'}{\partial t} \right)^2 \quad (36)$$

$$ED = e \frac{h}{U_*^3} \quad (37)$$

The P-SAT framework also allows user to compute the power spectra of filtered/unfiltered three-dimensional velocities by converting a time domain signal into frequency domain using a discrete Fast Fourier Transform (FFT). The spectrum of a filtered signal represents the mean square amplitude of that signal. In other words, spectrum shows the energy of a signal at any given frequency²¹. Energy in turbulence is received at large scales, and its dissipation occurs at small scales. Spectra allows us to think about the way, in which the energy is exchanged among eddies of different sizes.

Calculation of the mean velocity component from the instantaneous velocities for highly unsteady flow. In order to analyze structure of the unsteady flows, determination of the mean velocity component ($\overline{U_{uf}}$) from the instantaneous velocities (U_{uf}) is an essential parameter. From the definition point of view, instantaneous velocities of the unsteady flow can be decomposed into the mean velocities and fluctuating components in a following way:

$$U_{uf}(t) = \overline{U_{uf}}(t) \pm u' \quad (38)$$

There are several methods available to find out the mean velocity component from the instantaneous velocities²². However, the Fourier-component method has been found most suitable for the determination of the mean velocity component in unsteady flows²³. The determination of the mean velocities in unsteady flows using the Fourier-component method can be done as follows: Time dependent instantaneous velocities U_{ufi} (where $i = 1, 2, \dots, n$) are transformed into the frequency domain by using a discrete Fourier transform and only the frequency components lower than a cutoff frequency (f_{cutoff}) are taken as the representative values of the mean velocities ($\overline{U_{ufi}}$) as follows:

$$f_{cutoff} = (k-1)/2T. \quad (39)$$

$$\overline{U_{ufi}} = \frac{1}{2} a_0 + \sum_{j=1}^{(k-1)/2} (a_j \cos \omega_{ij} + b_j \sin \omega_{ij}) \quad (40)$$

where,

$$a_j = \frac{2}{n} \sum_{i=1}^n U_i \cos \omega_{ij} \quad (41)$$

$$b_j = \frac{2}{n} \sum_{i=1}^n U_i \sin \omega_{ij} \quad (42)$$

and

S. no.	Bed material	d_{50} (mm)	Angle of repose (dry sand), ϕ°	Gradation coefficient, σ_g (Marsh et al. 2004)	Flow depth, h (m)	Discharge, Q (m ³ /s)	Reynolds number (Re) of flow
1	River sand	1.1	31.154	1.03	0.135	0.0205	31418.26

Table 3. Details of various experimental parameters.

Figure 1. Schematic diagram of the experimental flume.

$$\omega_{ij} = 2\pi j \left(\frac{i}{N} \right) \quad (43)$$

for $j = 0, 1, 2, \dots, (k-1)/2$. Here, n is the number of samples collected in the time period (T) of a measurement. Nezu et al.²² have suggested to adopt the value of cutoff frequency (f_{cutoff}) to be smaller than the burst frequency of turbulence, and they considered to select the number of Fourier components (k) as seven.

Experimental setup and measurements

The dataset used in the present study to test the P-SAT framework, has been obtained from the experimental study carried out by Deshpande and Kumar²⁴. Experiments were performed in a glass-walled tilting flume of dimensions 20 m X 1 m X 0.72 m (length X width X depth). A schematic diagram of the flume has been provided in Fig. 1.

An upstream collection tank of dimensions 2.8 m X 1.5 m X 1.5 m (length X width X depth) was provided with a couple of wooden baffles installed in it to quieten the flow before entering the channel. Uniform river sand of median diameter $d_{50} = 1.1$ mm was used as bed material in the experiments. Table 3 provides information on various physical characteristics of bed material and experimental parameters.

Velocity measurements. Instantaneous velocity readings along the vertical plane were taken using a four-beam, down-looking, Vectrino+ acoustic Doppler velocimeter (ADV) probe manufactured by Nortek. The instrument collects data in a cylindrical remote sampling volume located at 5 cm below the central transmitter. The height of the sampling volume was set at 1 mm when measurements were taken very near the bed such that the sampling volume did not touch the particles on the bed surface, and at 4 mm when measurements were taken away from the bed. Data were collected at the center line of the channel cross-section at a distance of 8 m from the downstream end of the flume to minimize the effects of flow entrance and exit conditions on the measurement location. Measurements were carried out on multiple heights, and at each height, 30,000 samples were collected for 5 min (i.e., sampling rate of 100 Hz).

Figure 2. Flow diagram of the P-SAT framework.

P-SAT configuration, usage, features, performance and limitations

Flow diagram of the P-SAT framework. The P-SAT framework has been designed for working with steady as well as unsteady flows. Figure 2 shows the flow diagram of the P-SAT framework. The complete flow can be divided into the following main components.

- **Steady flow analysis**

1. **Spike detection and removal methods** It consists of three Spike Detection and Removal Methods: (a) Velocity correlation filter, (b) Signal-to-Noise Ratio (SNR) filter, (c) Acceleration thresholding method¹⁶. The P-SAT framework de-spikes the contaminated data points obtained from the ADV and replaces them via interpolation between ends of spike method. Each of the above filtering method, requires taking an input threshold from the user for filtering purposes (a default value is also stored in the program in case the user does not wish to specify or is unsure, for example, the default value of Correlation Filter is set as 70). Once the input threshold is set, the P-SAT framework scans for the .dat files and processes them one by one. For each filtering method, it computes the spikes and replaces them by interpolation between the ends of spikes. The algorithm for the above three filtering methods along with replacements algorithm is explained in Sect. 4.4. After running each filtering method, the relevant output files are saved into .csv format by the P-SAT framework.
2. **Turbulent statistics** The filtering methods mentioned earlier, de-spikes the noisy raw signal and converts it into a clean signal. The filtered signal is then further used for the calculations of various turbulent statistical parameters such as: time-averaged 3D velocities, velocity variances, skewness, kurtosis, Reynolds shear stresses (RSS), third order correlations, 2D as well as 3D fluxes of the turbulent kinetic energy, turbulent kinetic energy dissipation, conditional statistics of quadrants, computation of octants²⁵ and their corresponding probabilities of occurrence. All these are also saved for each input file and upon every execution, these parameters are appended in the file, so that previous computation are intact with the user.
3. **Computation of the spectral density functions of velocities** This component is used for the computation of the spectral density functions for the given input velocities. Since the ADV provides 3D velocities, the P-SAT framework computes the spectra for the velocities in streamwise, lateral, and vertical

Time (s)	SL	counter	U (m/s)	V (m/s)	W (m/s)	W1 (m/s)	AMP-U	AMP-V	AMP-W	AMP-W1	SNR-U	SNR-V	SNR-W	SNR-W1	Corr-U	Corr-V	Corr-W	Corr-W1
0	1	100011	0.2525	0.0081	0.0003	0.0226	108	124	131	106	19.1	20.3	21.5	19.7	44	71	73	58
0.01	2	100011	0.3524	0.0602	0.0212	0.0163	146	176	171	138	21.7	23.3	23.8	22	82	88	98	93

Table 4. An instance of the *.dat* file obtained from Vectrino software.

directions. A final spectra that combines all the above spectra is also plotted for user convenience. The P-SAT framework saves both the data and visualization in individual files for future reference.

4. **File organization** The P-SAT framework creates '14' files for every input *.dat* file it reads (for steady flow data). In order to ensure that it does not clutter the workspace of the user, the P-SAT framework creates a folder and organizes all the saved file and appends timestamp to all files. This is convenient from the user point of view as the user can perform further computation on the saved files. It also deletes any temporary files that are created during the program execution.

- **Unsteady flows**

1. **Determination of the mean velocity component** As has been discussed in Sect. 2.2, several methods are available for the calculation of the mean velocity component from the instantaneous velocities in unsteady flow environments²². The P-SAT framework computes mean velocities using the Fourier-component based method, which has been found most suitable for the determination of the mean velocity component in unsteady flows²³. The P-SAT framework can work with multiple files of unsteady flows specified in the "input_ensemble_files.txt". This file contains list of all those files that contain unsteady flow data. For each of the file, the P-SAT framework asks for the number of components and computes the mean velocity computation via Fourier Component based Averaging method. The user has to provide a *.csv* file having 4 columns: *t*, *u*, *v*, *w* representing time and the 3D velocities.

Importing data from raw files.

- **Input files (*.dat)** : P-SAT framework requires that the input should be in *.dat* format and the data should be comma separated. This restriction has been kept since it ensures that the large code is easily managed. There are various tools available that can help to convert a space/tab separated file(s) into a comma separated format. Table 4 shows a sample *.dat* file obtained from the Vectrino+ software.

When the raw velocity data is collected in any environmental flow experiment using a Nortek made Vectrino+ ADV and the Vectrino+ software, a *.vno* file is generated. This *.vno* file is further processed by the Vectrino+ software and five more files are created with *.adv*, *.dat*, *.hdr*, *.pck*, and *.ssl* extensions. The *.dat* files are standard ASCII files that contain velocity time series data obtained from Vectrino+ ADV. Typically the *.dat* files obtained from this software consists of the data as shown in Table 4. The columns as explained below:

- **Time** Contains the time information, depending on the sampling frequency. In this case the sampling frequency was 100 Hz and so the time interval changes every $\frac{1}{100}$ s.
- **SL** Routine serial counter from 1 till n.
- **Counter** A value provided by Vectrino software. It is not required by the P-SAT framework.
- **U_i , V_i , W_i , $W1_i$** : U_i , V_i , W_i represent the three dimensional velocity data collected by the ADV. $W1_i$ is a redundant component of W and is not required for the current study. These velocities are raw signal (contains spikes) that need filtering methods to de-spike the noisy data.
- **AMP- U_i , AMP- V_i , AMP- W_i , AMP- $W1_i$** : The signal amplitude of the U_i , V_i , W_i , $W1_i$ as measured by the Vectrino+ ADV instrument. These values are not used by the P-SAT framework.
- **SNR- U_i , SNR- V_i , SNR- W_i , SNR- $W1_i$** : The SNR values of the U_i , V_i , W_i , $W1_i$ as measured by the Vectrino+ ADV instrument. These values are used for the SNR filtering method.
- **Corr- U_i , Corr- V_i , Corr- W_i , Corr- $W1_i$** : The correlation values of the U_i , V_i , W_i , $W1_i$ as measured by the Vectrino+ ADV instrument. These values are used for the correlation filtering method.

It is important for the P-SAT framework that the columns of the input *.dat* files must be exactly in the same order for the successful execution.

Dataset processing. A key component of the P-SAT framework is that it uses the Numpy arrays that loads the *.dat* file efficiently into the memory. The Numpy module enables us to easily import specific columns from

the input *.dat* files and process them efficiently. After the raw *.dat* files are filtered, the P-SAT framework also converts them into usable *Microsoft Excel* file format. This feature was added, if the user wants to do some analysis on the filtered data using *Microsoft Excel*. We used *.xls* format so that we can maintain compatibility with the older version of *Microsoft Excel*. The final turbulent statistics are exported in a *.csv* format. The *.csv* format can be opened in any text editor or in *Microsoft Excel* for plotting various visualizations. Keeping *.csv* format enables the scientific community to write their own codes for further manipulations.

De-spiking and replacement algorithms. The P-SAT framework implements three de-spiking method: (a) Correlation Filter (shown in Algorithm 1), (b) SNR Filter (shown in Algorithm 2), (c) Acceleration Threshold Filtering (shown in Algorithm 3).

Algorithm 1: Algorithm for detecting and replacing spikes using Correlation Method

Input: 3D velocities measured using ADV, U_i , V_i ; and W_i representing instantaneous velocities in the streamwise, lateral and vertical directions, respectively. Correlation values obtained for U_i , V_i ; and W_i from the ADV. **User input** for Correlation Threshold.

Output: Filtered (Spikes Removed) 3D velocities via Correlation Method

```

1 for each  $U_i$ ,  $V_i$ ; and  $W_i$  do
2   if Correlation of  $U_i$  or Correlation of  $V_i$  or Correlation of  $W_i$  < Correlation Threshold then
3     Mark the respective data point as spike.
4     Replace the spike using interpolation between the ends of spikes.
5   else
6     The point is not a spike.
7   end
8 end

```

We will explain the algorithm for detecting spikes using **Correlation method**. The others filtering methods can be explained similarly. The input for Correlation Method Algorithm are the 3D velocities measured using ADV, U_i , V_i ; and W_i represents instantaneous velocities in the streamwise, lateral and vertical directions, respectively, and their corresponding correlation values (**Corr- U_i** , **Corr- V_i** , **Corr- W_i**). For every row (Line 1, Algorithm 1) that is read by the P-SAT framework (a sample row consists of the data shown in Table 4), it checks if **any** of the correlation values for U_i , V_i ; and W_i is less than the *Correlation Threshold* (Line 2, Algorithm 1), the corresponding velocity point is marked as spike (Line 3, Algorithm 1). The P-SAT framework provides flexibility: it provides default values for correlation threshold and also provides an option to allow the end user to enter a custom value for correlation threshold. The spike is replaced by the interpolation between the ends of spike (Line 4, Algorithm 1). The replacement algorithm is depicted in Algorithm 4. If the correlation of all the velocities are above the correlation threshold, the velocities are untouched (as they are not spike) and the next row is processed (Line 6, Algorithm 1).

The other two filtering methods are described below.

Algorithm 2: Algorithm for detecting and replacing spikes using SNR Method

Input: 3D velocities measured using ADV, U_i , V_i ; and W_i representing instantaneous velocities in the streamwise, lateral and vertical directions, respectively. SNR values obtained for U_i , V_i ; and W_i from the ADV. **User input** for **SNR** Threshold.

Output: Filtered (Spikes Removed) 3D velocities via SNR Method

```

1 for each  $U_i$ ,  $V_i$ ; and  $W_i$  do
2   if SNR of  $U_i$  or SNR of  $V_i$  or SNR of  $W_i$  < SNR Threshold then
3     Mark the respective data point as spike.
4     Replace the spike using interpolation between the ends of spikes.
5   else
6     The point is not a spike.
7   end
8 end

```

SNR filter Similar to the correlation filter but uses SNR (Signal-to-Noise ratio) values (using values of **SNR- U_i** , **SNR- V_i** , **SNR- W_i** as explained earlier).

Algorithm 3: Algorithm for detecting and replacing spikes using Acceleration Thresholding Method

Input: 3D Velocities measured using ADV, U_i is the velocity component for the current timestamp and u_{i-1} is the velocity component for the previous timestamp. Δt represents the time difference between the two timestamps. λ_a is a parameter that provides a good approximation if chosen between $1 - 1.5^{15}$. g is the gravitational constant

Output: Filtered (Spikes Removed) 3D Velocities via Acceleration Thresholding Method

```

1 while no more negative spikes is left do
2   Compute the acceleration from  $a_i = U_i - u_{i-1} / \Delta t$ .
3   Identify those points where  $a_i < -\lambda_a * g$  and replace them.
4 end
5 Use the above filtered list and identify the positive spikes as computed below.
6 while no more positive spikes is left do
7   Compute the acceleration from  $a_i = U_i - u_{i-1} / \Delta t$ .
8   Identify those points where  $a_i < +\lambda_a * g$  and replace them.
9 end

```

Algorithm 4: Algorithm for spike replacement via interpolation between ends to spike

Input: 3D velocities measured using ADV, U_i ; V_i ; and W_i representing instantaneous velocities in the streamwise, lateral and vertical directions, respectively. Spikes obtained from via Acceleration Thresholding Method or Correlation or SNR algorithm

Output: Filtered (Spikes Removed) 3D velocities via Interpolation between ends of Spike

```

1 for each spike identified via Acceleration Thresholding Method or Correlation or SNR algorithm do
2   Locate the last good point that was not identified as spike.
3   Locate the next good point that was not identified as spike.
4   Replace the spike with the interpolated value of the last and next good points.
5 end

```

Acceleration thresholding method We have used the method as proposed in Goring and Nikora¹⁶. The acceleration thresholding method first detects the spikes and then replaces them in two phases. First phase detection is done for the negative accelerations (Lines 1–4, Algorithm 3), while the second phase detection is performed on positive accelerations (Lines 6–8, Algorithm 3). In each of the phase, there are multiple passes made through the raw data until all the data points of the given sample conform to the acceleration criteria. General formula for acceleration is $a_i = \frac{U_i - u_{i-1}}{\Delta t}$. For negative accelerations all those points where $a_i < -\lambda_a g$, are marked as spike and replaced by linear interpolation between the ends of spike. The process is repeated until no more negative spikes are left. The same procedure is repeated for the positive accelerations. However, the check for positive spikes is $a_i > \lambda_a g$. The replacement strategy continues to be the same as that applied for the negative spikes (linear interpolation between the ends of spikes). The criteria is $\lambda_a g$, where λ_a is a user defined value and g is the universal gravitational constant. Goring and Nikora¹⁶ mention that values for λ_a should be kept between 1 and 1.5.

Finally, the spike replacement used is linear interpolation between ends of spike and is shown below. Given a spike data point, the algorithm takes the **last good point** that was **not** identified as spike and the **next good point** that was **not** identified as spike and performs spike replacement via linear interpolation between **last good point** and the **next good point**.

Figure 3 shows the raw velocity signal captured by the ADV and filtered signals by the P-SAT framework in a stepwise manner. The raw velocity signal at a particular measurement location has been depicted in Fig. 3A. Whereas, the raw signal filtered by using the SNR method, velocity correlation method, acceleration thresholding method, and by the combination of all the methods mentioned above are shown in Fig. 3B–E, respectively.

Sample code execution. For testing the P-SAT framework, we took the raw **.dat** files generated by the Vectrino+ software. A sample row from the raw file is displayed in Table 4. For user convenience we have presented the sample code which is ready for the execution. After downloading the entire GitHub repository (<https://github.com/mayank265/flume.git>), the user just needs to extract the zip archive (Password: “PoTs_Turbulence” (password does not contain quotes)) execute `python3 pots_module.py`. We have taken two sample **.dat** files and have shown the execution of the P-SAT framework. All **.dat** files that need to be processed must be in the same folder where the `pots_module.py` is placed. Also the `input_files.txt` and `input_files_corresponding_depths.txt` should be in same folder and should be correctly mapped (see Fig. 4). The P-SAT framework creates a **Logs** folder that contains all the errors that were recorded during the execution of the P-SAT framework. This can help for the debugging purposes.

Figure 3. (A) Raw velocity signal, (B) Signal filtered with SNR method only, (C) Signal filtered with correlation method only, (D) Signal filtered with acceleration thresholding method only, (E) Signal filtered using all the methods mentioned above.

Figure 4. Mapping the filenames of *input_files.txt* with their corresponding depths in *input_files_corresponding_depths.txt*. We see that *file_1.dat* has a depth of 0.0717 and *file_2.dat* has a depth of 0.0773. The user can place any number of files in *input_files.txt* but should map the corresponding depths in *input_files_corresponding_depths.txt*.

External libraries. The P-SAT framework requires the following libraries for the successful execution.

- **.csv**²⁶ Useful for writing files into *.csv* format. The final analysis is written into a *.csv* file.
- **datetime** Computes datetime for generation of various time format. In order to prevent conflicting filenames, we append a timestamp format to every file that is created.
- **glob** Organizing files and folders using regular expression patterns.

- **numpy**²⁷ The core library for manipulating the *.dat* files and applying various operations on them. Numpy processes the *.dat* files efficiently by converting them to numpy arrays.
- **scipy** Provides libraries for various statistical parameters like kurtosis, skewness etc.
- **os** All the filtered files are stored in a “Filtered_ Timestamp” folder which is generated by the os module.
- **timeit** For computing the code execution time.
- **xlwt**²⁸ For converting *.dat* files into *Microsoft Excel* format *.xls*.

The user can use `pip install package_name` to install the libraries. The authors would recommend installing the Anaconda²⁹ environment which is cross platform package and installs all the above packages with the inbuilt installer.

Performance. The P-SAT framework is an open source module written in Python 3+ and is tested both on *Linux* and *Windows* platforms. We have used few common libraries that are easily available. Python packages are easy to install using the standard Python package installer *pip*.

We tested the performance on the following system: OS Name: Microsoft Windows 10 Pro, OS Version: 10.0.17763 Build 17763, System Type: x64-based PC, Processor: Intel(R) Core(TM) i5-6200U CPU @ 2.30 GHz, RAM: 12 GB DDR4. We have executed the P-SAT framework and provided it with a set of 33 *.dat* files with their respective depths. The statistical parameters of turbulence obtained via P-SAT for these 33 files are shown in Tables 5, 6, 7, 8, 9 and 10. The table header is kept self-explanatory for easier understanding. Due to space constraint, the statistical parameters of turbulence obtained via P-SAT are split across multiple tables. However, the P-SAT framework stores all the statistical parameters of turbulence in a single file.

Experimental conditions are given as follows: The velocities were measured in an open channel flow experiment at the sampling rate of 100 Hz using the Nortek made Vectrino+ ADV. The ADV was suspended from a position in the flow in order to get the 3D velocities. The U_i , V_i , W_i , represent the instantaneous velocities in the streamwise (U_i), lateral (V_i), and vertical (W_i) directions, respectively. Each experiment was run for a period of 300 s. So in a given run, the total sample velocities obtained are $100 \times 300 = 30,000$. For 33 measurements the total number of sample readings we obtained is $30000 \times 33 = 990,000$ values. The P-SAT framework takes each input *.dat* file as input, does the pre-processing, then applies three filters to de-noise the velocity time series data and then calculates various turbulent parameters. During this process the P-SAT framework generates *.xls* files for all input files, the files obtained after each filtering, and a final analysis file (Parameters.csv) having the turbulent characteristics of all the input files.

The execution time for the code was 12 min 22 s. The memory usage varied between 30 MB – 160 MB in a single execution. Considering the various parameters the P-SAT framework calculates, and the conversion it does, the time taken can be justified. During different runs the time taken may reduce or increase depending on the quality of the raw file. As seen in the acceleration thresholding method, the process needs to be repeated till all the spikes are removed. A noisy velocity time series data may have a large number of spikes taking more time, while a sample having few spikes is executed quickly. So, we can conclude that the P-SAT framework does not hog system resources and can be run even on a basic desktop machine.

Tables and visualizations from the P-SAT framework. As mentioned already, the P-SAT framework generates a rich set of turbulent statistics like mean velocities, variance, skewness, kurtosis, Reynolds stresses, third order correlations, 2D as well as 3D fluxes of turbulent kinetic energy, turbulent kinetic energy dissipation, conditional statistics of quadrants, computation of Octants and their corresponding probability. Based on the rich statistics obtained, we plotted few visualizations as shown below.

Tables 5, 6, 7, 8, 9 and 10 show all the turbulent statistics generated by the P-SAT framework. We have split the tables into multiple pages as all the turbulent statistics generated by the P-SAT framework cannot be displayed using a single table.

The visualization of the turbulent statistics shown in Tables 5, 6, 7, 8, 9 and 10 is depicted in Figs. 5a, 6, 7 and 8b. Time-averaged velocities in all three directions (U , V , and W) are plotted against flow depth in Fig. 5a. Vertical distributions of three components of velocity variances are shown in Fig. 5b. Fig. 5c, d depict the vertical distribution of velocity skewness, and velocity kurtosis, respectively. Reynolds shear stresses (off diagonal components of a symmetric second order tensor) are plotted against the flow depth in Fig. 5e. Vertical distribution of the flow anisotropy is shown in Fig. 5f. Figure 6a shows the distribution of the third statistical moments (M_{30} , M_{03} , M_{12} , and M_{21}) plotted against the flow depth. Vertical distributions of the turbulent kinetic energy dissipation (ϵ) and its non-dimensional form (ED) are shown in Fig. 6b. 2D as well as 3D fluxes of the turbulent kinetic energy are plotted in Fig. 6c and d, respectively. Fractional contribution to the total Reynolds shear stress from different quadrants (quadrant analyses) for hole sizes (H) = 0, and 2 are presented in Fig. 6e and f, respectively. The 3D view of octants is shown in Fig. 7a, and vertical distributions of the octant probabilities are presented in Fig. 7b. Power spectral density functions for all three components of velocities are depicted in Fig. 8b.

P-SAT framework is also able to compute the mean velocities ($\overline{U}_{uf}$) of unsteady flows from the instantaneous velocities of the unsteady flows (U_{uf}) by using the Fourier-component method described in the theoretical development section. Figure 8a shows an example of the time series of instantaneous velocities $U_{uf}(t)$ and the mean velocity component $\overline{U}_{uf}(t)$. The P-SAT framework allows the user to select any random number of Fourier components. It can be observed from Fig. 8a that the mean velocity component comes very close to the instantaneous velocities for the number of Fourier components (k) at ninety one.

Figure 5. Visualizations that can be derived from the P-SAT framework.

Limitations and precautions to be taken while using the P-SAT module. Following points need to be taken care by a user executing the P-SAT framework.

1. The P-SAT framework requires a strict file naming convention for the input *.dat* input files. It is preferred that the input files should not have spaces or any special character except “_”.
2. Ensure all source files are strictly *comma* separated files. That implies that all source files have values separated by “,”.
3. The *input_files.txt* and *input_files_corresponding_depths.txt* are correctly mapped (see Fig. 4).
4. The libraries mentioned in Sect. 4.6 must be installed before running P-SAT framework.
5. Ensure that no file/filename is changed while the P-SAT framework is executing. It may result in an erroneous output.

Figure 6. Visualizations that can be derived from the P-SAT framework (Continued...).

Discussion

There are various commercial tools as well as free tools available for calculation of the turbulent parameters. Commercial software provide users with a simple Graphical User Interface (GUI) and allows them to filter the raw velocity time series data, but they are usually expensive. If we check out the formulas provided in Sect. 2, it can be seen that few of them can be computed in *Microsoft Excel* while for others formulas can be built. In fact the authors initially began with turbulent analysis on *Microsoft Excel* as it provided a lot more flexibility in terms of plotting graphs and was helpful for analysis. However, the authors faced the following issues while working with *Microsoft Excel*: (a) For Quadrant and Octant analyses, the formulation in *Microsoft Excel* was tedious. (b) Also, an oversight in one of the formula can hamper the results of all subsequent formulas. (c) With the data size growing (as in our case, nearly ≈ 1 million values over 33 files), the computations began to get heavy resulting in *Microsoft Excel* occasionally freezing. (d) The process to merge all the statistical data after analysis was to be done manually making the whole process cumbersome. (e) Working on unsteady flows requires Fourier averaging which is challenging to compute in *Microsoft Excel* and to the best of author's knowledge, no such tool exists

Figure 7. Visualizations that can be derived from the P-SAT framework (Continued...).

that provides for the computation of unsteady flows with a user chosen values of k (number of components). All this took a lot of time, and there were always room for errors.

We wanted to provide the community with a free and open source module that can perform all the tasks hassle free. The P-SAT framework is completely open source, which allow the developers and scientific community to extend it to meet their needs and purposes.

Conclusion and future work

We provide the end user with an open source P-SAT framework that can enable the user to filter the raw velocity time series data obtained from the Nortek Vectrino+ ADV and compute various turbulent parameters. We believe that P-SAT framework is a first of its kind framework that can completely automate the process of parsing, filtering, computation of various turbulent statistics, spectra computation for steady flows. For unsteady flows the P-SAT framework obtains mean velocities from instantaneous velocities by using Fourier-component based averaging method. P-SAT framework also saves all the processed data files so that the end user can use it for future purposes without actually needing to run the P-SAT framework again on the same set of files.

The authors have put the source code on GitHub and would welcome suggestion and improvements for the same. We believe that the P-SAT framework will help the scientific community working with environmental hydraulics by providing a means to easily filter and compute the parameters. As the core code is developed in Python, it runs smoothly on Windows, Linux and MAC based operating system.

Right now the P-SAT framework works on a command line based environment, in the future we would like to build a GUI module for the same. The GUI module will allow for richer user experience, however, for this paper, we restrict it to command line environment only.

(a) Time series of instantaneous velocity for unsteady flow and mean velocities obtained for Fourier-components $K = 3, 61,$ and 91 .

(b) (A) Velocity power spectra for U , (B) Velocity power spectra for V , (C) Velocity power spectra for W , and (D) Power spectral density functions for all three components of the time-averaged velocities.

Figure 8. Visualizations that can be derived from the P-SAT framework (Continued...).

Filename	Depth	U	V	W	u_var	v_var	w_var	u_skew	v_skew	w_skew	u_kurt	v_kurt	w_kurt
File1	0.0773	0.40903	0.02397	-0.01251	0.00372	0.00777	0.0007	-0.27849	0.33847	0.01724	2.31085	2.44263	1.02965
File2	0.0717	0.36454	-0.00931	-0.01033	0.00595	0.00151	0.0003	-0.04423	-0.27285	0.22283	0.92397	17.36672	0.0482
..	..	..	..	..	..	..	..	..	..	..	..	..	..
File33	0.0015	0.2306	0.00225	0.00204	0.00335	0.00227	0.00028	0.2295	0.81738	-0.20091	2.75305	6.51751	1.47359

Table 5. Statistical parameters of turbulence obtained via P-SAT.

Filename	Depth	u_stdev	v_stdev	w_stdev	$\overline{u'v'}$	$\overline{u'w'}$	$\overline{v'w'}$	Anisotropy	M_{30}	M_{03}	M_{12}	M_{21}
File1	0.0773	0.06101	0.08817	0.02637	-7.24E-05	-0.000105663	2.27E-05	0.43212	-0.27848	0.01718	-0.0393	0.05581
File2	0.0717	0.07717	0.0388	0.01718	4.65E-05	-0.0001211	-1.02E-05	0.22264	-0.04428	0.22277	-0.02444	-0.04098
..	..	..	..	..	..	..	..	..	..	..	..	..
File33	0.0015	0.05789	0.04767	0.01669	-3.66E-06	-0.000151116	6.45E-05	0.28823	0.22944	-0.20087	0.01918	-0.00581

Table 6. Statistical parameters of turbulence obtained via P-SAT (Continued...).

Filename	Depth	$2Df_{TKEu}$	$2Df_{TKEu}$	$2Df_{TKEw}$	$2Df_{TKEw}$	$3Df_{TKEu}$	$3Df_{TKEu}$	$3Df_{TKEw}$	$3Df_{TKEw}$	TKE
File1	0.0773	-4.87E-05	-2.77E-06	4.34E-06	2.47E-07	-4.86E-05	-2.77E-06	1.01E-05	5.77E-07	0.006095886
File2	0.0717	-1.57E-05	-8.92E-07	-2.30E-06	-1.31E-07	-1.05E-05	-5.95E-07	-2.63E-06	-1.49E-07	0.003877757
..	..	..	..	..	..	..	..	..	..	..
File33	0.0015	3.36E-05	1.91E-06	-9.44E-07	-5.37E-08	1.49E-05	8.50E-07	4.76E-06	2.71E-07	0.002951285

Table 7. Statistical parameters of turbulence obtained via P-SAT (Continued...).

Filename	Depth	$S_{1,0}$	$S_{2,0}$	$S_{3,0}$	$S_{4,0}$	$S_{1,2}$	$S_{2,2}$	$S_{3,2}$	$S_{4,2}$	e	ED
File1	0.0773	-1.98977	2.64558	-1.99061	2.3348	-0.52656	1.06369	-0.64186	0.66729	0.00046	0.00036343
File2	0.0717	-1.39548	1.91653	-1.46716	1.94611	-0.41369	0.60727	-0.43675	0.50796	0.00078	0.000621967
..	..	..	..	..	..	..	..	..	..	..	..
File33	0.0015	-0.72003	1.16876	-0.67938	1.23064	-0.23043	0.51603	-0.05579	0.42531	0.00041	0.000330071

Table 8. Statistical parameters of turbulence obtained via P-SAT (Continued...).

Filename	Depth	Octant+1	Octant-1	Octant+2	Octant-2	Octant+3	Octant-3	Octant+4	Octant-4	Total Sample
File1	0.0773	3516	3813	3592	3200	3784	3639	4005	4445	29994
File2	0.0717	3383	4623	3763	3786	3883	3143	3216	4197	29994
..	..	..	..	..	..	..	..	..	..	..
File33	0.0015	3121	3944	4404	3358	3889	3956	2963	4358	29993

Table 9. Statistical parameters of turbulence obtained via P-SAT (Continued...).

Filename	Depth	Probability Octant+1	Probability Octant-1	Probability Octant+2	Probability Octant-2	Probability Octant+3	Probability Octant-3	Probability Octant+4	Probability Octant-4	Min Octant Count	Max Octant Count
File1	0.0773	0.11722	0.12713	0.11976	0.10669	0.12616	0.12132	0.13353	0.1482	3200	4445
File2	0.0717	0.11279	0.15413	0.12546	0.12623	0.12946	0.10479	0.10722	0.13993	3143	4623
..	..	..	..	..	..	..	..	..	..	..	..
File33	0.0015	0.10406	0.1315	0.14683	0.11196	0.12966	0.1319	0.09879	0.1453	2963	4404

Table 10. Statistical parameters of turbulence obtained via P-SAT (Continued...).

Methods

We have uploaded the required dataset and the source code for the P-SAT module on Zenodo. The url for accessing the data and code is: (<https://doi.org/10.5281/zenodo.4097839>). After downloading the zip file from the Zenodo repository, the user just needs to extract the zip archive and execute `python3 pots_module.py`.

Received: 28 October 2020; Accepted: 14 January 2021

Published online: 17 February 2021

References

1. Thurner, L. et al. Pandapower—an open-source python tool for convenient modeling, analysis, and optimization of electric power systems. *IEEE Trans. Power Syst.* 33, 6510–6521 (2018).
2. Virtanen, P. et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nat. Methods* 17, 261–272 (2020).
3. Klopfenstein, D. et al. GOATOOLS: a python library for gene ontology analyses. *Sci. Rep.* 8, 1–17 (2018).
4. Kennedy, J. H. et al. LIVVkit: an extensible, python-based, land ice verification and validation toolkit for ice sheet models. *J. Adv. Model. Earth Syst.* 9, 854–869. 2017, doi: 10.1002/2017MS000916.
5. Ukkonen, P. & Mäkelä, A. Evaluation of machine learning classifiers for predicting deep convection. *Journal of Advances in Modeling Earth Systems* 11, 1784–1802. 2019, doi: 10.1029/2018MS001561.
6. Thomas, D. et al. Cudydo: an integrated python environment for coupled fluid-structure simulations. *Adv. Eng. Softw.* 128, 69–85. 2019, doi: 10.1016/j.advengsoft.2018.05.007.
7. Deshpande, V. & Kumar, B. Effect of downward seepage on the shape of an alluvial channel. In *Proceedings of the Institution of Civil Engineers-Water Management* Vol. 170, 3–14 (Thomas Telford Ltd, 2017).
8. Patel, M., Majumder, S. & Kumar, B. Effect of seepage on flow and bedforms dynamics. *Earth Surf. Process. Landf.* 42, 1807–1819 (2017).
9. Devi, T. B. & Kumar, B. Channel hydrodynamics of submerged, flexible vegetation with seepage. *J. Hydraul. Eng.* 142, 04016053 (2016).
10. Lade, A. D., Deshpande, V., Kumar, B. & Oliveto, G. On the morphodynamic alterations around bridge piers under the influence of instream mining. *Water* 11, 1676 (2019).
11. Barman, B., Kumar, B. & Sarma, A. K. Impact of sand mining on alluvial channel flow characteristics. *Ecol. Eng.* 135, 36–44 (2019).
12. Guala, M., Singh, A., BadHeartBull, N. & Fofoula-Georgiou, E. Spectral description of migrating bed forms and sediment transport. *J. Geophys. Res. Earth Surf.* 119, 123–137 (2014).
13. Parsheh, M., Sotiropoulos, F. & Porté-Agel, F. Estimation of power spectra of acoustic-doppler velocimetry data contaminated with intermittent spikes. *J. Hydraul. Eng.* 136, 368–378 (2010).
14. Lacey, R. J. & Roy, A. G. A comparative study of the turbulent flow field with and without a pebble cluster in a gravel bed river. *Water Resour. Res.* 43 (2007).
15. Nikora, V. & Goring, D. Flow turbulence over fixed and weakly mobile gravel beds. *J. Hydraul. Eng.* 126, 679–690 (2000).
16. Goring, D. G. & Nikora, V. I. Despiking acoustic doppler velocimeter data. *J. Hydraul. Eng.* 128, 117–126 (2002).
17. Pope, S. B. *Turbulent flows* (Cambridge University Press, Cambridge, 2000).
18. Raupach, M. Conditional statistics of reynolds stress in rough-wall and smooth-wall turbulent boundary layers. *J. Fluid Mech.* 108, 363–382 (1981).
19. Lu, S. & Willmarth, W. Measurements of the structure of the reynolds stress in a turbulent boundary layer. *J. Fluid Mech.* 60, 481–511 (1973).
20. Nezu, I., Tominaga, A. & Nakagawa, H. Field measurements of secondary currents in straight rivers. *J. Hydraul. Eng.* 119, 598–614 (1993).
21. Hinze, J. O. *Turbulence: an introduction to its mechanism and theory*. McGraw-Hill (1959).
22. Nezu, I., Kadota, A. & Nakagawa, H. Turbulent structure in unsteady depth-varying open-channel flows. *J. Hydraul. Eng.* 123, 752–763 (1997).
23. Nezu, I. Turbulent structure over dunes and its role on suspended sediments in steady and unsteady open-channel flows. In *Proceedings of International Symposium on Transport of Suspended Sediments and its Mathematical Modeling*, 165–189 (IAHR, 1991).
24. Deshpande, V. & Kumar, B. Turbulent flow structures in alluvial channels with curved cross-sections under conditions of downward seepage. *Earth Surf. Process. Landf.* 41, 1073–1087 (2016).
25. Keylock, C. J., Lane, S. N. & Richards, K. S. Quadrant/octant sequencing and the role of coherent structures in bed load sediment entrainment. *J. Geophys. Res. Earth Surf.* 119, 264–286 (2014).
26. jasontrigg0. python-csv 0.0.11. <https://pypi.org/project/python-csv/> (2018).
27. Oliphant, T. NumPy: a guide to NumPy. USA: Trelgol Publishing (2006–). Accessed 15 Dec 2019 (Online).
28. Newman, M. E. J. Network data. <http://www-personal.umich.edu/~mejn/netdata/> (2013).
29. Anaconda - the world's most popular data science platform. <https://www.anaconda.com> (2020).

Acknowledgements

This research was undertaken with the assistance of resources from the Water Resources Engineering Laboratory, Department of Civil Engineering, Indian Institute of Technology Guwahati and Unit of Environmental Engineering, Ben-Gurion University of the Negev, Beer-Sheva, Israel. We are thankful to the assistance provided by both the research labs in conducting the experiments.

Author contributions

M.A.: Writing, Data Curation, Software. The entire Python coding work for computing turbulent intensities has been done by M.A. V.D.: Conceptualization, Data Curation, Writing. Carried out all the experiments under various conditions and obtained the results. D.K.: Technical advice and instruments for capturing the parameters relating to unsteady flows. B.K.: Experiments relating to the steady flows were performed at the laboratory flume setup at Department of Civil Engineering, Indian Institute of Technology Guwahati. B.K. helped us setup the entire flume.

Funding

The authors have no external funding agency to mention.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to V.D.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2021