**UTESPac Users Instructions**

Utah Turbulence in Environmental Studies Process and Analysis Code (UTESpac)

Created by: Derek Jensen

[derek591@gmail.com](mailto:derek591@gmail.com)

Version 4.1 Version

1 January 2017

Document Edited by: Eric Pardyjak

Pardyjak@eng.utah.edu

University of Utah

Department of Mechanical Engineering

Environmental Fluid Dynamics Lab

1. **About UTESPac**
   1. **Overview**

UTESpac is a set of Matlab scripts and functions that may be used to analyze high-frequency atmospheric turbulence surface-layer data. It was designed specifically for use with Campbell Scientific dataloggers and its accompanying LoggerNet software. The code has native support for

* Sonic Anemometers: RMYOUNG 8100, Campbell Sci CSAT3
* Open Path Gas Analyzers: Licor 7500, Campbell Sci EC150, Campbell Sci IRGASON, and Krypton Hygrometers
* Finewire thermocouples for sensible heat flux computations
* Propeller Anemometers
* Mean meteorological sensors (e.g., T/RH, Pressure, Solar, cup anemometers, etc.)
  1. **File Formats**

UTESpac expects 24 or 48-hr CSV tables as input files, it quality controls the data, and then computes means, fluxes, variances and derived temperatures (potential temperature, virtual potential temperature) and stores the output in a MatLab structure or NetCDF file.

1. **Steps for Using UTESPac**
   1. Convert Campbell Binary files to CSV files using the Card Convert Program in LoggerNet
      * Options: File Processing - Use Time, set to 2 days 00 h under Time Settings
      * File naming - Use TimeDate Filenames and Append to Last File if multiple site files exist Array CSV Options
      * Timestamp Options - Include year, day, hour/minutes, seconds, don't include midnight is 2400, Array ID,
      * Array Datalogger Format = Hour/Minutes and Seconds
   2. Create a folder for the individual site. The folder name needs to be preceded by the keyword "site". For example, for a site named Playa the folder name is sitePlaya. Place the .csv files within the site folder.
   3. Create a subfolder named *output* in the site folder, this is where the output data from UTESPac will be stored.
   4. Create a file called siteInfo.m and store it in the site folder where site specific information (e.g. orientation of sonic are placed). See example file below.

Figure ??? – Example siteInfo.m file.

% site specific script to load site information

% enter orientation of sonics. Sonic order: tables sorted alphabetically followed by columns sorted in ascending order

info.sonicOrientation = [49]; %UU1 Chateau Nipun/Florian Measured

% enter manufacturer of SAT. 1 for Campbell, 0 for RMYoung. RMYoung v = Campbell u!

info.sonicManufact = [1]; % UU1

% enter orientation of tower relative to sonic head

info.tower = 180; % UU1

% tower elevation

info.siteElevation = 283; % (m)

% enter expected table names. Missing tables will be filled with NaNs to create consistency

% when multiple output files are concatenated with getData.m

info.tableNames = {'UU1\_20Hz'};

% enter table scan frequencies corresponding to tableNames

info.tableScanFrequency = [20]; %[Hz]

% enter number of columns in each .csv table. Note that the number of columns in the output structure will

% be 3 less than the number in the .csv file. This is because the 4 column date vector is replaced with a Matlab's

% single-column serial time. Also, note that View Pro frequently cuts of column 1 (the year!) of the .csv file.

info.tableNumberOfColumns = [10];

The info.tower variable is specified as follows and as illustrate in Fig. XX below. The idea is that the sonic head is the origin and the angle is from the origin to the tower.  For example if the sonic is due south of the tower, info.tower=0.  This variable is nice for flagging wind directions though since the wind direction is the angle from whence the wind cometh; therefore, wind directions equal to info.tower ± are flagged.

A close up of text on a whiteboard

Description automatically generated

Figure XX – Illustration of the info.tower angle in the site.info file.

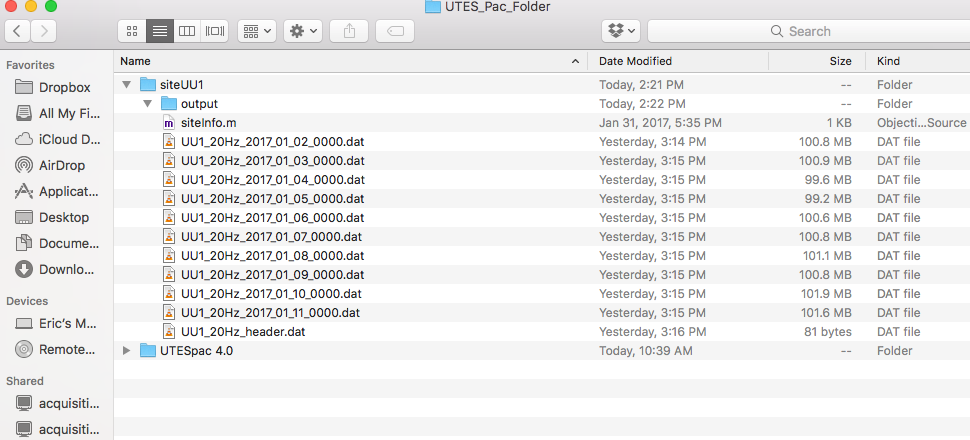


Figure ?? – Example of an acceptable file structure with CSV files.

1. Create header files for each data table. The syntax is tableName\_header.dat (e.g., "Playa\_1HZ\_header.dat", "Playa\_20HZ\_header.dat). Note that tableName must be consistent with the .csv tableNames created in step 1. The header file is a single line .dat, comma delimited file containing variable names and heights for all columns within the respective data table. The header file is 3 columns shorter than the .csv data file. This is because UTESpac immediately calculates the serial date numbers from the date vectors (columns 1 - 4) contained in the data tables. The serial dates are stored in column 1 and columns 2 - 4 are deleted, thus becoming consistent with the header file. The easiest way to create the header file is with Card Convert. Create an ASCII T0A5 file, there is no need to run the whole binary file, simply stop the conversion immediately and only a few hundred lines will be created. Open the file in a text editor and delete all lines outside of the variable headers (typically line 5). The variable names within the header and the sensor templates (defined on lines 155-169) must be consistent. The template is used by UTESpac to identify specific sensors in the header. The rules for creating the template and header variable names are:
   1. the template and variable name are the exact same except the sensor height is replaced with the wildcard \* in the template. e.g., template = 'Ux\_\*', header variable name = 'Ux\_0.5', 'Ux\_10'
   2. The sensor height must be the last numeric value in the header variable name - All sensors (with exception of solar and battery) need an associated height in meters - Heights within the header variable name at a given tower height need to exactly match. e.g. 'FW\_5','Ux\_5','RH\_5'
2. Computing a Global Planar Fit. If a global planar fit is used, a PFinfo structure, containing global planar fit coefficients, will be stored in the site folder. There is no need to do anything with it. Note: For the Global Planar Fit, there must be 1 and only 1 set of 5 minute, local planar fit data. That is, the global planar fit will fail if there are two files such as, '5minAvg\_LPF\_linDetrend' and '5minAvg\_LPF\_constDetrend' in the output folder. There must be only one or the other (it doesn't matter which!).
3. Fill out the information section of the code (lines 56 - 116) and run the code. A full example study is included in UTESpac.zip
4. Use getData(), structFill() and structConcat() to produce complete (no missing days) datasets over the full experiment. See example

Global planar fit

% select 'local' or 'global' planar fit, 'local' computes coeffiecients from local file only, 'global' computes user-defined, multi-sector, multi-datebin coefficients from all site data - the sector and datebins are defined

% graphically when the code is executed - for 'global' calculations, all data must first be run with a 'local' planar

% fit and 5-min averaging

1. **Theory and Basic Algorithms**
2. **Steps for Using UTESPac**
3. **Theory and Basic Algorithms**