10 October 2018

TO: WECAN file
FROM: Al Cooper
SUBJECT: Workflow for WindInWECAN.Rnw

The main document itself contains a reasonably detailed description of the development of the report on wind measurements in WECAN. That development does not need much elaboration in this document, but some functions were developed in the process of constructing this document and some further details and supporting explanation may help anyone wanting to duplicate or extend this work. This workflow therefore focuses mostly on those functions and a few related aspects like the sources of data used in the report. It is admittedly wordy and chatty because it is intended as possible guidance for someone wanting to duplicate the work reported here. It describes extraneous material like lines of investigation that didn't work out and are not included in the report or asides regarding incidental problems like incomplete data files that could have been fixed but were not. I also apologize for not editing this document as carefully as usual, with the excuse that this is meant to be relatively informal so readers may be hoped to excuse grammatical awkwardness.

Sources of Data

At this time, only the data files produced during preliminary processing are available for WECAN, and the data files used for the WINTER, NOMADSS and FRAPPE projects are similarly flagged as "preliminary data that should not be used for critical analysis". Even the WINTER data are so flagged, but they have been used in preference to the "production" files because many variables of interest are not included in the final production files. To facilitate reproducibility, subset data files as used by this program are saved in the EOL directory ~cooperw/RStudio/Reprocessing, with the names AKRDforC130.Rdata and ARISTO-LAMS.Rdata. EOL data-access rules prevent making these files public until the data are released for public access, so for now only internal-to-EOL personnel with responsibility for data processing for the project should use these files. The program has the code to reconstruct these files from the temporary EOL data repository (/scr/raf_data/WECAN, etc.), and later could do so from the released data files. The data files may change as EOL personnel review them and correct errors in or make changes in processing; that is one reason for caution when using these data files. The author has a personal copy of the data files used on /scr/raf_data at the time of this project and could make those available when it becomes permissible to do so.

Two data files used here need some additional discussion:

AKRDforC130.Rdata Code in the main routine constructs this file from a repository like /scr/raf_data/ if it is not present, but skips this processing and loads the existing file if the file is already present. To regenerate this file, delete the existing version before running the main routine. During construction of the data.frame contained in this file, variables needed for the complementary-filter representation of angle of attack have been added; see, e.g., "QR"

which is ADIFR/QCF and QRS and QRF which are the low-pass-filtered and high-pass-filtered versions of QR. A reference value called AOAREF is also added with filtered components; it is based on Eqn. (2) in the main document. The file is about 144 MB so it is not available in the GitHub archive; retrieve it from EOL workspace /h/eol/cooperw/RStudio/Reprocessing/AKRDforC130.Rdata.
That version included data from WINTER and FRAPPE as well as WECAN, although the fit recommended in the conclusions of this report is only based on the WECAN data, Some flights, listed in the "Bad" vector of file names, have been excluded from the data.frame because they appeared questionable for various reasons so it was thought better to omit them to avoid distorting the fit. The remaining flights included more than 720,000 measurements included in the fits and included measurements from 7 WINTER, 15 FRAPPE, and 16 WECAN flights. On the first attempt, NOMADSS flights were also included, but the results did not prove acceptable for NOMADSS and distorted the fit results for the other projects so NOMADSS was excluded. This may have been caused by the unavailability of GGVSPD in NOMADSS; that variable was used in the other calculations of vertical wind, and the substitute used in NOMADSS (VSPD_G) is an avionic-system variable with poor response characteristics compared to GGVSPD, so this may have been the reason that the results for NOMADSS were inadequate. Replacement of GGVSPD with the calculated variable "ROC" also did not work well.

If this file is to be re-created, the corresponding netCDF files for these projects need to be located in directories as specified by the Ranadu function DataDirectory() on the local machine.

ARISTO-LAMSB.Rdata This data file was generated during the ARISTO-LAMS study and is re-used here. Its generation was described in "ARISTO-LAMS.pdf", a study of how the LAMS performed in ARISTO-2015 and 2016 and a record of the recommendations from that study regarding pressure corrections and angles of attack and sideslip as determined from LAMS. The .Rdata file is only about 4 MB in size so it is included with the zip archive and as a file in the GitHub repository.

Special Functions

Some functions were isolated to the 'chunks' directory because they were expected to be of further use, for example by processing routines that implement the algorithms recommended in the "WindInWECAN" report. Here is a list of the functions that are loaded from the "chunks" subdirectory during processing:

removeSpikes.R: This function will identify spikes, remove them, and interpolate to replace the omitted points. This is unused as of now but was important in the related SOCRATES document so it has been retained here. The function calculates the rolling mean and standard deviation (width 99) of a variable "v", identifies values of the variable that differ from the

rolling mean by more standard deviations than a limit specified by sdLimit (default 4), and replaces those values by interpolation in the returned variable.

SplitCV.R: Components of a variable are split into low-pass-filtered and complementary high-pass-filtered components, as is needed for the implementation of the complementary-filter representation of AKRD.

SummarizeFit.R: The output is an abbreviated version of the fit information provided in full by "summary()" to characterize the fit produced by the R routine "lm()".

AddWindC130.R: This file incorporates two functions, "AddWindC130()" to add the calculated vertical wind to a data.frame, and "AddWind()" which is a non-functioning remnant from the SOCRATES version of this project. "AddWindC130()" uses the coefficients determined in this study to add "AKY" to the data.frame representing the new angle of attack, and it calculates the new vertical wind from the simplified formula:

$$\mathrm{WIY} = \mathrm{WIC} + \mathrm{TASX} * (\mathrm{AKY} - \mathrm{AKRD}) * \pi/180$$

The function has appropriate code to do a full wind calculation from the new angle-of-attack, but the problem is that the current high-rate files only have GGVSPD at 1 Hz so the wind calculation would have to use a smoothed version of the vertical rate-of-climb of the aircraft. It would be possible to solve this by running the high-rate files again and specifying GGVSPD output at 25 Hz, which would use the 10-Hz sample-rate data interpolated to 25 Hz as is done in conventional high-rate processing. I chose not to do this to avoid the processing required, but it would still be possible. That would improve the result for WIY, especially in turns. The full wind calculation is being done now in this function and is just over-ridden by the above formula in the last statement before the return statement from AddWindC130(), so it would only be necessary to comment that last statement. Another approach that was tried was to substitute the variable "ROC" (see the Processing Algorithms document) for GGVSPD, but an attempt to do this led to too much noise arising from noise in PSFC. Code for that is also included, but is suppressed. The variable ROC works well as a substitute for GGVSPD for the GV, so it is not clear why this attempt failed. It is possible that the test regions I used were regions with low real variability in the aircraft motion so that the noise level dominated while it might not have for a more turbulent region.

processWind.R This is left from the GV study of LAMS and is a remnant not used in the present study. Instead, a modified version of this function is included in the R code as the Rnw chunk "processWind", a modified version of the GV routine that matches variable names and other aspects of the C-130 LAMS files. The reference to "PCA" refers to Scott's approach to finding the beam speeds which was based on principal component analysis.

mergeMatt.R This function was used to combine the processed line-of-sight beam speeds in Matt Hayman's processing into the standard netCDF file produced by normal "nimbus" processing, optionally processed also by the Python script "LAMS_ARISTO.py" discussed below. The reason for needing a special step (vs. normal data.frame merging) was that the "Time"

variable in Matt's file was not the usual variable used in nimbus-produced netCDF files because it lacked date information, so some additional processing was needed to revise the time variable. It has not been used in the present study but was needed to produce the "ARISTO-LAMS.Rdata" file that is used here.

Others: Some other functions are left in the "chunks" directory although they are not used. They might be in a version that added new variables to the netCDF file. "AddWind.R is the old GV version, and "DemingFit.R" was a modified version of the Ranadu function that now has been incorporated into the Ranadu package; it is left here as a record of what was used in the GV study. "VSpec.R" is a similar working version of a Ranadu function that is not used in the present work; see "?Ranadu::VSpec" for usage information or the extended description in the document "WorkflowWindInSOCRATES.pdf".

Comments on Individual Sections of the Report

Section 1: Introduction

The embedded R-code "chunk"[1] named "initialization" includes this code:

1. Loading of library packages. In addition to knitr and Ranadu, some others that are used include "zoo" (for interpolation functions), "ggplot2" (for some plots including those produced for variance spectra by VSpec()), and some others used in association with ggplot2. "signal" is also used for fitting and should be available, but it is not loaded as a library; instead, where used, it is referenced via, e.g., signal::filtfilt()" or "signal::butter()". This is partly to avoid a package conflict in function names that results from loading the library "signal". The function "filtfilt" filters in both the forward and backward directions and

2. Also defined in this section are the vector VarList (specifying the variables to include when constructing the all-project data.frame) and the CutoffPeriod (600 s) used for the break between low-pass and high-pass components of the complementary filter for AKRD. This list includes some other variables (e.g., GGVEW) that are needed if the three-dimensional wind is to be calculated with the new angle of attack, although a simplified equation for the new vertical wind was used in this study for reasons explained elsewhere in the document and this description of the workflow. Note that the variable GGVSPD is not included here; instead, it is added later. The reason was related to attempts to include NOMADSS flights for which GGVSPD was unavailable. Once it was decided to exclude NOMADSS, the code structure was not changed but it could be simplified by adding GGVSPD to this list and omitting the NOMADSS-related code in the R-code chunk named "construct-dataframe".

[1] not to be confused with the functions contained in the subdirectory named "chunks" and loaded from there before execution. Here "chunk" refers to a section in the .Rnw document that contains R code and is identified by the indicated name at the start of the chunk of R code, in the header beginning with "<<initialization, ...".

3. Several functions used in this study are loaded from the "chunks" directory during initialization. Part of the reason for this was that the earlier study for SOCRATES included the capability of adding new variables to the netCDF file representing the result from the complementary-filter representation of AKRD. The separate processing code re-used these chunks of code to ensure consistency between the analysis document and the reprocessed files. That capability is not included as part of the present study because in the meantime appropriate code has been incorporated into nimbus and checked, so any processing for WECAN will be done using nimbus with that code and the new coefficients in this report.

Section 2: Relevant Results from ARISTO-2016

LAMS in ARISTO-2016 provided an independent reference for angle of attack, but the previous report used that reference only to find an empirical fit to the conventional formula for AKRD as contained in the ProcessingAlgorithms.pdf document. That analysis was repeated here in order to find the indicated best estimate of the high-pass-filtered portion of AKRD for use in the present study. The low-pass-filtered component was not addressed in this section because the orientation angle of the LAMS differs from that of the radome and therefore the slowly varying component of AKRD is not represented reliably by LAMS. Furthermore, there is some evidence (described in Section 3) that the mean attack angle in WECAN may differ from that in previous projects, so the results from ARISTO-2016 may not be relevant to the slowly varying component of AKRD.

In regard to sideslip, three circle maneuvers were used to determine the separate offsets in heading and sideslip for the C-130. The three were from different projects, but their consistency suggests that they might be combined. The circle maneuver provides an indication of the combined effects of offsets in heading and sideslip by considering how the measured wind varies around the circle. When the aircraft is flown at right angles to the wind, an error is introduced by a (heading+sideslip) offset, and that error changes sign when the wind is from starboard vs. when it is from port. A fit to the sinusoidal variation in the measured wind speed therefore can indicate the magnitude of this combined error. Separation of the heading from sideslip offsets uses measurements from two circles, one flown clockwise and one counterclockwise. Because in turns the roll of the aircraft is associated with a nose-up change in orientation, a component of sideslip is introduced. Because the roll angle is the same in magnitude but reverses sign for the two turn directions, the sideslip changes sign, but the lift of the aircraft remains the same to maintain level flight. Therefore an error in sideslip will appear as a difference in sideslip for the two directions of the circles. The heading has no effect on this lift, so the change in measured sldeslip for the two turn directions indicates the magnitude of the sideslip correction. This is the only maneuver that can separate the effects of heading offset and sideslip offset; reverse-heading maneuvers, often used to check the sideslip offset, actually detect the combined effects of offsets in heading and sideslip. Such maneuvers have been used to adjust the sideslip offset or the heading offset in the past, but the circle maneuvers allow determination of the separate contributions of heading and sideslip offset. For this reason, the results for the two offsets should both be used in processing; adjusting only the sideslip coefficient as indicated here will introduce a significant error because the heading offset in current use is -0.1° while the result from the circle maneuvers is significantly different.

These results are presented first in the report so they can be used in combination with the complementary-filter representation of AKRD to determine coefficients for WECAN.

The embedded R chunks associated with this section are these:

1. *processWind:* This section is used to calculate wind variables from the line-of-sight measurements of wind speeds from LAMS. The process is described in ARISTO-LAMS.pdf so no further discussion will be included here.

2. *LAMSfit:* In this section of R code, the data.frame saved as ARISTO-LAMS.Rdata is loaded and, after some preliminary calculations not included in the report but examined in the course of developing the report, the data.frame is processed in "processWind()" to add the angle of attack determined from the measurements from LAMS. There appeared to be a change in offset angle between the LAMS and radome sensors with time, perhaps caused by fuel use and the associated decreased weight and decrease in wing flex. Therefore a linear fit to the time dependence of the difference between the LAMS-based angle of attack and AKRD was used to correct the LAMS-based measurement. This is a relic from earlier use and did not affect determination of the high-pass component of angle of attack in the present study. In addition, a variable named TASCSQ was required to be smaller than 0.1 for the corresponding measurement of angle of attack from LAMS to be included in the fit. This variable was calculated in "processWind()" and represented the chisquare for the four-beam determination of true airspeed. Restricting it eliminated cases of inconsistency among the four beams and so provided some level of assurance that the measurements were valid. In addition, some apparent outlier measurements (identified by an absolute value of the difference between the time-adjusted measurement from LAMS and AKRD that exceeded 0.8°) were also excluded from the fit. Then the remaining measurements were used to determine a complementary-filter fit to the high-pass-filtered LAMS-based measurements of angle of attack as a function of the high-pass variable ADIFR/QCF (QRF in the code). Examination of the results indicated that the residual standard deviation of the fit was reduced when a time shift forward by 750 ms was introduced to the LAMS-based reference value. This sign of the shift adjusts the measurement later in time and so would be consistent with a LAMS measurement ahead of the aircraft. The magnitude of the shift that gave the smallest residual time shift was larger than the expected time shift arising from the positions of the LAMS sensitive volumes and the airspeed, but there may be an additional time shift arising from the different data systems used to record the LAMS measurements and the other standard measurements used to measure ADIFR/QCF. The fit argument was AOAREFF ~ 0 + QRF; this forces the fit to have zero constant and only a slope coefficient, as required for the high-pass component of angle of attack. A scatterplot is generated in this section that shows the correspondence between the resulting high-pass component of angle of attack provided by LAMS and the result of multiplying the fit coefficient by the high-pass-filtered value of ADIFR/QCF. The coefficient is saved in the variable "cfcff" for inclusion in the text and later use in the recommendations as the best coefficient for the high-pass-filtered component of angle of attack.

Section 3: Finding the "Slow" Component ...

This section first constructs the data.frame containing the measurements from several projects and multiple flight from each project, as described above (cf. "AKRDforC130.Rdata"). To repeat this study for a new project, delete (or rename) the .Rdata file, change the "Project" vector, and if necessary modify the "Bad" list of flights to exclude. If you want to exclude only a part of some flight, see the statement starting "if (fno == 303) ..." for a model for such an exclusion. This section is also where the low-pass-filter and high-pass-filter components of the reference variable and the dependent variables for the fit are constructed. See the discussion of the "construct-dataframe" code chunk below. The report itself contains a relatively complete discussion of how the data.frame was constructed so no further elaboration is included here.

A variable named "RF" is added to the data when the data.frame is constructed, so it might be worthwhile explaining how that is used. The concatenated data.frame contains all projects and all flights, but if it is desirable during exploratory analysis to look at individual projects or an individual flight that can be done via restricting the data via Data[Data$RF == 310,] (for WECAN flight 10) or Data[Data$RF > 300,] (for all WECAN flights. Note the ",]" closing; that includes all variables in the resulting data.frame, and the statement won't work without that.

The last part of this section discusses the fits that use the data.frame. Some exploration of different fit variables was explored but not included in the report because the result of that exploration was to settle on the same variables used for the GV. There are outside reports that the sensitivity of the 5-hole Rosemount 858 probe depends on Mach number so that sensitivity was explored here (and is the reason that filtered values of the Mach number are included in the data.frame), but inclusion of that term into either the "fast" or "slow" fit did not result in significant improvement (as was also the case for the GV), so no Mach-number dependence is included in the representation developed here.

I have usually approached these fits as follows:

1. Construct as large a list as practical of variables, products of variables, and powers of variables and their products that might be correlated with the reference variable in the fit. Use the R routine "lm()" to find a linear-model fit using all those variables. A typical statement might be:

   $$\mathrm{f} < -\mathrm{lm}(\mathrm{RefValue}, \mathrm{Var1}, \mathrm{Var2}, \mathrm{I}(\mathrm{Var1} * \mathrm{Var2}), \mathrm{I}(\mathrm{Var\hat{}2}), \mathrm{data} = \mathrm{Data})$$

   where "Data" is the data.frame containing the variables. See the documentation for lm() to understand why some of these are enclosed in "I()".

2. Use "summary(f) to see the list of terms and their significance levels. Alternately, use "anova(f)". Identify the least significant dependent variable and eliminate it, then repeat the fit. Check that the residual standard deviation (called in "summary" the residual standard error) did not increase significantly.

3. Keep repeating until all variables are indicated to be significant or the residual standard deviation does increase significantly.

4. Keep trying to eliminate variables, using especially "anova(f)", as long as the residual standard deviation does not increase significantly (where "significantly" depends on the application but often would be more than a few percent). Try to avoid overly complicated final fits because they may produce spurious results when applied to cases not incorporated in the dataset being used for the fit.

R-code chunks related to this section are discussed here:

1. *construct-dataframe:* This is discussed extensively in the report and earlier sections of this workflow document, so only a few comments will be added here:

   (a) The filter function used here, signal::filtfilt(), will crash if it encounters missing values (NA in R) in the sequence, so a preliminary step used in filtering is to replace NA values with interpolated values. This is accomplished here using the routine zoo:approx(). That requires a vector argument, so that is why the data.frame component being filtered is enclosed in "as.vector()". The argument "na.rm=FALSE" is required because, if NAs remain and are removed, the resulting vector has the wrong length for incorporation into the data.frame as a new variable. Instead, some replacement for remaining NAs is used, often "0". This occurs rarely but when it does it is usually at the start or end of data segments where the effect of the substituted value is minor and isolated.

   (b) "signal::filtfilt()" is used with Butterworth filter coefficients provided by signal::butter. The frequency argument to butter() is 2/CutoffPeriod because the values are scaled to the Nyquist frequency. 1/CutoffPeriod is the desired cutoff frequency, but for a 1-Hz signal the Nyquist frequency is 0.5. 2/CutoffPeriod gives twice the desired cutoff frequency, but the argument is interpreted as a multiple of the Nyquist frequency so a CutoffPeriod of 600 would give an argument to butter() of 2/600 which, multiplied by the Nyquist frequency, would specify an appropriate cutoff at 1/600.[2] "filtfilt" averages in both forward and backward directions to reduce the phase shift introduced by unidirectional filtering, but tests of the implementation in nimbus (which is necessarily unidirectional) indicate that the results are not significantly different in this application. An argument could be made that "filter" should be used instead because the intent here is to develop an algorithm for implementation in nimbus.

2. *buildDF:* A minor chunk of code that restricts the data to measurements with TASX > 65 and abs(ROLL) < 2. In a side effect that I don't understand, the first statement results in NA values in DF$RF that weren't there before that statement, apparently arising when TASX or ROLL is missing (NA), so the second statement keeps those values from propagating into the data.frame used for fitting.

3. *fits-to-DF:* This R-code chunk has "include=FALSE" in the header, so results from the print statements do not get included in the PDF-format file that is the report. That was used in a

[2] ... I think ... I may be mixed up regarding this interpretation, but I have read through the documentation several times and even tested this numerically. I'm still somewhat worried about this interpretation.

preliminary search for flights that looked suspicious and perhaps should be excluded from the fit, so several runs repeated this section with different netCDF files excluded.[3] Several fits are included in this chunk that were only used for exploration and have not entered the final document. The section is not written economically because some of that exploratory code is still here but doesn't affect the results. An example is the calculation of "AKYA", the new angle-of-attack based on the three-project dataset. A variable "AKY" was defined tentatively that did not include the first coefficient cfs[1] (representing d_0), and then a project-dependent offset (e.g., cfs1d for WECAN) was found from the mean difference between the low-pass-filtered value of AOAREF and the low-pass-filtered representation of angle of attack without the first coefficient. That difference was then used to adjust the fit to each project to give fits that matched the mean value of AOAREF for that project. As described in the report, that fit was not as good as needed, so a WECAN-only fit replaced that as the fit that was eventually recommended. See fsWECAN and cfsWECAN for that fit and the corresponding coefficients. Variables AKYW and WIYW were calculated from that fit. WIYW. AKYW, and cfsWECAN are suggested as the best results to use for WECAN processing. The averages listed for each flight in the data.frame are based on WIYW for WECAN but WIY calculated with a project-dependent offset for the other projects. My use of WIY is inconsistent and may be confusing: Here WIY refers to the calculations with a project-dependent offset imposed, but later WIY is the result of using the recommended coefficients in cfsWECAN and replaces what is here called WIYW. (Apologies for the inconsistency!)

4. *contour-plot:* This section generates a plot that attempts to convey the distribution of measurements about the fit in a way that is less confusing than a scattergram. A scattergram was tried here first, but with more than 440,000 points it was not evident how well the points were distributed about a best-fit line. Here I used a filled.contour() plot with a matrix that was filled with binned values in the two coordinates for the plot (AOAREF and AKYW, labeled "AKY" in the plot). The range $(-1, 5)^\circ$ was divided into 0.05° increments for each variable and then the two-dimensional matrix of values was incremented with the appropriate index values for each available measurement. The number of items in each two-dimensional bin was then converted to a base-10 logarithm (with zero events assigned -1) and color-mapped by the filled.contour() function. The key was generated specially to convert back from the logarithms to the number of events. I think the result shows the central trend of the measurements and the degree of agreement with the fit much better than was possible with a scattergram. Perhaps the code in this section can serve as a model for other similar applications, and maybe should be incorporated as a new plot function in the package Ranadu, but this hasn't been done yet.

[3] It is a useful feature in RStudio that a code chunk like this can be executed repeatedly by placing the cursor somewhere in the code segment and typing "CTRL-ALT-C". "CTRL-ALT-N" runs the next chunk and moves the pointer there.

Section 4: Plots of the Results

No special comments regarding this section, except to note that the processing code reloads individual-flight netCDF files for each of the flights so access to the data files in "Directory/WECAN/" (where Directory is provided by Ranadu::DataDirectory()) is required for this section to work.

Section 5: High-Rate Variance Spectra

The intent of this section was to investigate some aspects of the high-rate calculations of vertical wind, but this was hindered by the available 25-Hz files because the convention when producing these files is to include variables not measured at 25 Hz at 1 Hz. Therefore the available files have GGVSPD available only at 1 Hz and, although GGVSPD contributes little at high frequency, this is still a hindrance to processing to calculate a new 25-Hz vertical wind. What I should have done, and perhaps still should do, is re-run nimbus but specify 25-Hz output for even 10-Hz output (the sample rate) or GGVSPD. In my haste to get at least a preliminary version of this report available, I didn't do that. If I had, I could use the results and the new 25-Hz value of angle-of-attack from the complementary-filter representation, with the Ranadu function "WindProcessor()" to recalculate the wind in its full 3-D representation, duplicating the normal processing. Instead, here I have used the following substitute:

$$\mathrm{WIY} = \mathrm{WIC} + \mathrm{TASX} * (\mathrm{AKY} - \mathrm{AKRD}) * \pi/180$$

This should be a reasonable representation of the vertical wind except in turns, where there will be errors associated with the roll of the aircraft. It appears, however, that this substitute for complete wind calculation still is adequate to represent the problems with the variance spectra that are reported in the main document.

It might be worth mentioning why loading of the high-rate data.frames is isolated into an initial code chunk called "get-hr-data". At least temporarily, the Ranadu function getNetCDF() prints some debug information about rate conversion for high-rate files, and that information ended up being printed into the final document if the getNetCDF() call was in the same code chunk as the code generating the variance-spectra plots. There is a way to suppress this output, but I couldn't remember it; I tried "sink()" commands and the "invisible()" command without success. The easy solution for now is to isolate the getNetCDF calls to a separate code chunk and specify "include=FALSE" for that chunk.

The plots of variance spectra are generated by the Ranadu function "VSpec()", which has many options not used here that are described in the "help" function accessible as ?Ranadu::VSpec , The default method uses the R function "spectrum()", which generates a periodigram via fast Fourier transformation and then smooths that periodigram with Daniell smoothing, a moving-average smoother. The default argument "spans=49" smooth over 49 frequencies. As used here, successive plots can be superimposed using the "ADD" argument, and the theme used for the plot (here "Ranadu::theme_WAC()" can be changed easily, for example by omitting the "+theme_WAC()" call to get the standard ggplot2 theme or by adding another theme via an argument like "g +

theme_classic()". "VSpec() also can produce spectra via the "maximum entropy" method or the Welch method, which implements smoothing by segmenting and averaging the segments. The final spectrum can also be smoothed by smoothing in logarithmic intervals, although that smoothing is not used for the plots generated in this document.

In the last two plots that compare longitudinal and lateral variance spectra, Figs. 4 and 5, the longitudinal spectra are generated by multiplying the variable UXC by $\sqrt{3/4}$ to compensate for the expected 4/3 ratio between longitudinal and lateral spectra in an inertial subrange. The modified variable is called UXCA in the plot labels.

Section 6: Conclusions and Recommendations

(no additional comments on this section)

– End of Memo –