19 September 2020

TO: SensibleHeatFlux Archive: workflow document
FROM: Al Cooper
SUBJECT: Workflow comments for the AMT paper re sensible-heat flux

1 Purpose

This workflow description provides additional details leading to the submitted paper titled, "On Measuring Sensible-Heat Flux with Airborne Thermometers." The intent is to support replication of the analyses and figures presented in the papers, to facilitate changes based on new data or new analysis approaches, and to make it practical to apply the proposed algorithms to additional data collected by research aircraft used in atmospheric studies. The associated ".Rmd" file contains both text (in LaTeX format) and R processing scripts for the analyses in the resulting paper and is the definitive reference for this work, but there are details here that are not evident in that script. Those details include a description of the process of collecting the observations and processing them to data files, the data archives used, the steps required to generate the plots and other results including the instances where manual intervention is required to identify appropriate subsets of the data, references to the relevant R code and LaTeX documents, and all the steps leading to the generation of the text in the papers. This overview and these diagrams will help explain the workflow at a general level and so should substitute for reading the R and LaTeX code in most cases.

For brevity the subject paper will be referenced as "the paper" here. It includes these sections:

1. Introduction
2. Determining the Transfer Function
3. Correcting for Dynamic Heating
4. The Flux Density of Sensible Heat
5. Conclusions

After some general comments that apply to all, there are separate discussions for each section.

2 Acquisition of the primary data

The measurements used in these papers were collected using the NSF/NCAR research aircraft during various research projects that are described in included references. The onboard data-acquisition program 'aeros' recorded the data in digital format, and those data files were then processed by the program 'nimbus' to produce an archive in NetCDF format. The software management group of NCAR/EOL maintains a version-controlled archive of these programs, so if they

are of interest they can be obtained by contacting the data-management group of EOL (at this or this address). The data files available from NCAR/EOL can be found at links on this URL, where FTP access can be requested. The details of the processing algorithms including those for the calculation of wind are documented in this report on Processing Algorithms and in Lenschow and Spyers-Duran [1989]. These procedures as they pertain to the measurement of wind are also documented in Cooper et al. [2016]. The resulting data files contain measurements in scientific units and brief descriptions of each measurement, included as netCDF attributes for the files and for each variable.

3 The .Rmd files (RMarkdown)

The .Rmd file is basically LaTeX text, generated for simplicity using LyX, exported to .Rnw format, converted to .Rmd using the "mdsr" package for R (Baumer et al. [2019]), and then placed into the Copernicus template provided by Daniel Nüst which uses the "rticles" package for R (Allaire et al. [2020]). After some necessary modifications to the output from "mdsr", this was processed to .pdf format using RStudio (RStudio [2009]). RMarkdown is described by Xie et al. [2018]. Within the .Rmd file or within the .lyx file there are "chunks" of R code (R Core Team [2019]), delineated by a header having this format:

```
‘‘‘{title, option = setting, ...}
...R code...
‘‘‘
```

These chunks generate plots and other results of analyses that are incorporated into the manuscript using 'knitr' (Xie [2013, 2014]). In RStudio, the chunks appear as gray sections in the file when it is edited. Where tasks involve execution of R code, the chunk containing the code is referenced in the discussion below. Any results from the processing can be incorporated into the LaTeX text via "\Sexpr{}" calls embedded in the LaTeX portion of the file.

Two "switches" serve to speed execution of the code: (1) the "cache" option which uses previous results without recalculation, and (2) "generateFigures" which when TRUE causes the figures to be generated and placed into the text. When FALSE, previously generated figures saved in the sub-directory "figure/" are used instead. The first switch significantly speeds execution in cases where small changes are made. To ensure a clean run, set the "cache" option to FALSE and remove all entries from the "cache" subdirectory. In some cases, the figures generated from "generateFigures <- TRUE" were renamed in order to be supplied separately for the AMT submission; e.g., fig10.pdf is the reference used for the include_graphics argument but the code chunk generating that figure is "SOCp1" so the figure placed in the "figures" directory with that name has been changed to "fig10.pdf".

4 Required R packages including Ranadu

The R code used for analysis reported in this paper relies heavily on a package of routines for R called "Ranadu." This is a set of R functions for working with the NetCDF archive data files produced by NCAR/EOL/RAF, and it includes some convenience routines for generating plots and performing other data-analysis tasks with the archived data. The Ranadu package as used here is preserved at this reference: Cooper [2020]. To run the R code referenced here, that version of the package should be included in the R installation. The .Rmd routine requires that package and also some others referenced in the file, including "knitr", "ggplot2", "grid", "ggthemes", "zoo", "signal" and "magrittr". In addition, Ranadu requires "ncdf4", "tcltk" and "stats". Some parts of "Ranadu" reference additional packages as needed, but they are not used in these papers so do not need to be available for this routine to run.

The data processing for this manuscript involved revising some parts of the Ranadu package, as listed below. The most significant change was the addition of "flux()" and "rk4.integrate()" functions to the package. The first function is used to generate and plot the cospectrum used to calculate the flux, and it can optionally apply the correction procedure developed in the paper. The second is used to integrate the differential equations in the paper.

It may be worthwhile to call attention to the "pipe" that is used frequently in the R code. This relies on the "magrittr" package for R. The general structure resembles the following:

```
DataFrame %>%
  select(Time, ATH1, ATF1, Ts) %>%
  selectTime(114500, 115000) %>%
  plotWAC()
```

At each occurrence of %>% the output from the previous command is piped to the next command as the first argument. This provides clear documentation of the sequences used to construct various plots. When the plot is generated using the "ggplot2" package, this pipe is often followed by a sequence of ggplot2 functions linked by "+".

The analysis uses R data.frames that are generated from the archived netCDF files by the function "Ranadu::getNetCDF()". Most of the plots are generated using "Ranadu::ggplotWAC()" or "Ranadu::plotWAC()", which are simple convenience routines that set various options preferred by the author before calling the R "ggplot()" or "plot()" routines.

There is a manual for the Ranadu package, included in the above package reference, that includes information on how to install it. See also this URL.

4.1 Ranadu::flux ()

This function calculates the appropriate cospectrum, generates a plot of that cospectrum, calculates the total flux and the partial flux associated with fluctuations smaller than a specified wavelength, and returns a data.frame containing the frequency, the smoothed cospectrum (not weighted

by frequency), and the exceedance values. The data.frame also has attributes "Flux" and "FluxL" representing the total flux and the flux from wavelengths smaller than the wavelength "wavelengthLimit". The "wavelengthLimit" is also included as an attribute of the returned data.frame.

An optional argument to this function is "Par" which can be set to appropriate response parameters for a temperature sensor. In that case, a response function is generated using those parameters and the cospectrum is corrected using that response function.

In the code used for this paper, this was replaced by "plotCS()", a function that incorporated much of the same code. This was done partly to facilitate placing the two panels of "fig10.pdf" into a single plot.

4.2 Ranadu::CohPhase ()

Another Ranadu function used for the calculations in the paper is "Ranadu::CohPhase()", which calculates and plots the squared coherence and phase between two time series. That function was recently modified for the needs of the analysis leading to the paper. Like all Ranadu functions, it can be obtained from the Zenodo reference cited above and its documentation can be checked using R help facilities once the package is installed. It was used to generate the cospectra used in the determination of the phase between the measured recovery temperature and the expected effect of dynamic heating.

A modified version of this routine is included in the .Rmd file, named CohP( ). It was modified to return binned values of the phase and amplitude ratio labeled with bin numbers so that multiple cross-spectra covering the same frequency interval could be averaged to obtain these values. This special version was used to find the averaged values entering, for example, Figs. 2 and 4.

4.3 Ranadu::VSpec ()

Plots of variance spectra, as shown in Figs. 3 and 6, were generated using the Ranadu function "VSpec()". There is a brief discussion of this function in "RanaduManual.pdf" p. 39, available at this URL or in the Zenodo reference. A tutorial Shiny-app is also available here that discusses plotting spectral variance and describes some of the options available in "VSpec()". R help facilities also provide documentation once the "Ranadu" package is installed.

4.4 Ranadu::SmoothInterp( )

Many functions including the standard spectral analysis functions and stats::fft will fail if there are missing values in the processed time series. To avoid this problem, "SmoothInterp( )" has been used extensively. It calls "zoo::na.approx( )" to substitute interpolated values for missing values, and it uses the "rule = 2" argument to that function to extend values to beginning and ending sequences with missing values. ("zoo:na.spline( )" has been used in this function in other

applications, but it tends to introduce extraneous variance.) By default, "SmoothInterp( )" also applies Savitzky-Golay smoothing, so this needs to be suppressed by the ".Length=0" argument to "SmoothInterp( )" when smoothing is not desired (which applies to all uses in these papers). "SmoothInterp( )" is called internally from within some of the functions that deal with variance spectra, including "VSpec( )" and "CohPhase( )".

The following construction can be used to remove all the missing values in a data.frame and substitute interpolated values:

```
removeNA <- function (D) {D <- SmoothInterp(D, .Length=0)}
DSW <- as.data.frame(lapply(DS11, removeNA))
DSW <- transferAttributes(DS11, DSW)
```

The last line is needed because the "lapply( )" function doesn't carry forward the variable attributes so this transfers them from the original data.frame.

4.5 rk4.integrate()

For use in this study, a new function was defined that implements the Cash-Karp method of step adjustment (Cash and Karp [1990]) for a fourth-order Runge-Kutta integration. This was done when it became apparent that the standard R function runge.kutta() in the "rmutil" package (Swihart and Lindsey [2019]) did not perform adequately for step sizes of 0.04 s (as required for processing 25-Hz files) because one time constant for the unheated Rosemount sensor is smaller than this step size. This new integration routine is based on the description of the method in Press et al. [1992] but has been coded independently in R for use here. Unlike the standard method, it does not increase the step size above the time increment in the file, but if the error estimate for a particular step does not meet the specified tolerance then the step is divided into multiple smaller steps. If the tolerance is still not met, further subdivision occurs.

One aspect of how this routine is used makes it problematic for general use: The integration takes as an argument a function that provides the derivative to be integrated, but that function takes a single argument. As applied in this analysis, however, additional variables are needed to calculate the derivative. Here is an example of a function definition used to find the temperature of the support that contacts the wire:

```
fS <- function(y, i) {
    ((1/a) * (tau1 * DS11$DTMDT[i] + DS11$RTF1[i] - (1-a) * y) - y) / (Rate * tau2)
}
```

This function only works by using quantities that are in the function environment (here, DS11, a, tau1, Rate, tau2). A better structure would provide those as additional arguments to the function, but that will require restructuring the "rk4.integrate( )" function as well. The specific function that provides the derivative is defined each time an integration is performed, usually just before the integration, but the practice of referencing the calling environment might lead to problems for someone wanting to extend or revise the program embedded with this document in the ".Rmd" file.

5 Comments on Section 1 (Introduction)

The short introductory section just provides some context. One matter to which the introduction only mentions briefly is that of terminology. "Sensible heat flux" is the commonly used term, but it seems inappropriate to this author because "heat" is not a property of the air but the flow of thermal energy as represented by the first law of thermodynamics. "Heat flux" is therefore at best ambiguous because heat is necessarily energy in motion. The primary formula strictly represents the transfer of enthalpy for an ideal gas, but "enthalpy flux" would not exclude the reactive enthalpy embedded in the water vapor so that isn't a good substitute. "Flux of thermal energy" also isn't appropriate because that would carry the connotation of the internal energy of a perfect gas; i.e., $C_v\rho_a < w'T' >$ with C_v in place of C_p. A further problem is that what is conventionally referred to as the flux is really a flux density. Maybe the best compromise would be a term like "non-reactive enthalpy flux density" but to my knowledge that term has never been used or defined. In the end I decided to stay with "flux of sensible heat". (I have the same reservations about the term "flux of latent heat" for the same reasons: It is not the heat but the energy that is latent.)

6 Comments on Section 2 (Determining the Transfer Function)

This section presents the steady-state solution to the differential equations (Eqns. (4) and (6))[1] for a sinusoidal input, as Eqns. (7)–(12), and determines the frequency-domain "transfer function" (ratio of output to input for sinusoidal input) from that solution. The solution was obtained using Laplace transforms, which may not be familiar to some potential users of this work, so extensive documentation of the approach is included in this section of the workflow document. Also included is a direct verification of the solution obtained by substituting into the governing differential equations and showing that those equations are satisfied. An additional derivation from the impulse function is included and shown to be equivalent. This detail did not seem appropriate to include in the paper but it may be of interest to someone wanting to replicate or extend this work. In particular, it may be useful if additional terms are needed to describe the time response.

6.1 Re: Section 2.1

Here is exhaustive detail on the Laplace-transform solution. Someone expert in Laplace transforms could surely do this more elegantly and concisely. I went into this much detail to be convinced that the solution proposed was valid. This material is mostly superfluous because the solution has been verified several ways, so most readers can skip this subsection. This material will be unnecessary for those already expert in Laplace transforms because they will be able to see the solution immediately from the differential equations, as co-author Carnes has demonstrated.

[1] Equations in the submitted paper are denoted here using the notation "Eq. (n)" as in that paper, while references to equations in this workflow document use the format "(n)".

Consider a sine-wave input $T(t) = \sin \omega t$ representing the normalized actual history of the air temperature. The temperature of the sensor and that of the support must both also be sine waves, possibly of different amplitudes and phases: $T_m(t) = c\sin(\omega t + \phi)$ and $T_s(t) = b\sin(\omega t + \zeta)$.

The equation for the support temperature$T_s(t)$, given by TN Eq. (4) and repeated here, is

$$\frac{dT_s(t)}{dt} = \frac{(T(t) - T_s(t)}{\tau_2}$$

Taking Laplace transforms (denoted by $\mathscr{L}$) gives

$$s\tau_2\mathscr{L}(T_s) - \tau_2 T_s(0) = \frac{\omega}{s^2+\omega^2} - \mathscr{L}(T_s) \tag{1}$$

$$\begin{aligned}\mathscr{L}(T_s) &= \frac{T_s(0) + \frac{1}{\tau_2}\frac{\omega}{s^2+\omega^2}}{\frac{1}{\tau_2}+s} \\ &= \frac{T_s(0)}{\frac{1}{\tau_2}+s} + \frac{\omega}{\tau_2}\frac{1}{\left(\frac{1}{\tau_2}+s\right)(s+i\omega)(s-i\omega)}\end{aligned}$$

Expanding the last term in partial fractions leads to:

$$\frac{T_s(0)}{\frac{1}{\tau_2}+s} + \frac{\omega}{\tau_2}\left(\frac{c_1}{\frac{1}{\tau_2}+s} + \frac{c_2}{s+i\omega} + \frac{c_3}{s-i\omega}\right)$$

where, from Kreyszig [1962] p. 219, and with $\tau = \tau_2$ so the equations can be reused later:

$$c_1 = \frac{1}{\omega^2 + (1/\tau)^2} = \frac{\tau^2}{1+\omega^2\tau^2} \tag{2}$$

$$c_2 = -\frac{1}{2i\omega(1/\tau - i\omega)} = \frac{\tau(i-\omega\tau)}{2\omega(1+\omega^2\tau^2)} \tag{3}$$

$$c_3 = \frac{\tau(-i-\omega\tau)}{2\omega(1+\omega^2\tau^2)} \tag{4}$$

The solution is then obtained by taking the inverse Laplace transform:

$$T_s(t) = T_s(0)e^{-t/\tau_2} + \left(\frac{\omega\tau_2}{1+\omega^2\tau_2^2}e^{-t/\tau_2} + \frac{(i-\omega\tau_2)}{2(1+\omega^2\tau_2^2)}e^{-i\omega t} + \frac{(-i-\omega\tau_2)}{2(1+\omega^2\tau_2^2)}e^{i\omega t}\right) \tag{5}$$

After any initial transient response decays, the long-time solution is then

$$T_s(t) = \frac{1}{(1+\omega^2\tau_2^2)}(-\omega\tau_2\cos\omega t + \sin\omega t) = b\sin(\omega t+\zeta) = b(\cos\zeta\sin\omega t + \sin\zeta\cos\omega t) \tag{6}$$

which leads to the solution for b and ζ:

$$b\cos\zeta = -\frac{1}{1+\omega^2\tau_2^2}$$

$$b\sin\zeta = \frac{-\omega\tau_2}{1+\omega^2\tau_2^2}$$

$$\tan\zeta = -\omega\tau_2 \tag{7}$$

$$b = \frac{1}{1+\omega^2\tau_2^2}\sqrt{(1+\omega^2\tau_2^2)} = \frac{1}{\sqrt{1+\omega^2\tau_2^2}} \tag{8}$$

This illustrates the use of the Laplace-transform solution that will next be applied to Eq. (6) from the paper. A specific input, $T(t) = \sin\omega t$, has been used, but a more general solution would replace the Laplace transform of this input ($\mathscr{L}(T) = \omega/(s^2+\omega^2)$) with the general form $\mathscr{L}(T)$ in (1). The transfer function in Laplace-transform notation then is the ratio of the Laplace transform of the output to that of the input, with initial-condition-transients neglected; in this case the transfer function is $G(s) = 1/(1+s\tau_2)$. For a general input function, this leads to the Laplace transform of the response via $\mathscr{L}(T_s) = G(s)\mathscr{L}(T)$, so the inverse transform of this equation will specify the solution for a general input. For example, for $T = B\cos\omega t$ with Laplace transform $\mathscr{L}(T) = Bs/(s^2+\omega^2)$, $\mathscr{L}(T_s) = Bs/\left((1+s\tau_2)(s+i\omega)(s-i\omega)\right)$ or

$$\frac{\tau_2}{B}\mathscr{L}(T_s) = s\left(\frac{c_1}{s+1/\tau_2} + \frac{c_2}{s+i\omega} + \frac{c_3}{s-i\omega}\right)$$

with coefficients c_i as found above. Apart from an initial decaying exponential, the solution is obtained from the inverse Laplace transform:

$$T_s(t) = B\left(-\frac{i\omega(i-\omega\tau_2)}{2\omega(1+\omega^2\tau_2^2)}e^{-i\omega t} + \frac{i\omega(-i-\omega\tau_2)}{2\omega(1+\omega^2\tau_2^2)}e^{i\omega t}\right) = \left(\frac{B}{1+\omega^2\tau_2^2}\right)(\cos\omega t - \omega\tau_2\sin\omega t)$$
$$= \frac{B}{1+\omega^2\tau_2^2}\cos(\omega t+\zeta)$$

where the gain is $1/(1+\omega^2\tau_2^2)$ and the phase shift is $\tan\zeta = \omega t$.

As a check, the solution can also be obtained as follows:

$$b\omega\cos(\omega t+\zeta) = \frac{\sin\omega t - b\sin(\omega t+\zeta)}{\tau_2}$$

$$b\omega\tau_2[\cos\omega t\cos\zeta - \sin\omega t\sin\zeta] = \sin\omega t - b[\sin\omega t\cos\zeta + \cos\omega t\sin\zeta]$$

Rearranging:

$$[1 - b\cos\zeta + b\omega\tau_2\sin\zeta]\sin\omega t = b[\sin\zeta + \omega\tau_2\cos\zeta]\cos\omega t$$

This can only be valid if each term in square brackets is zero. The right-side term leads to a simple result for the phase:

$$\tan\zeta = -\omega\tau_2 \tag{9}$$

The amplitude-ratio b then follows from the left-side term:

$$b = \frac{1}{(\tan\zeta \sin\zeta + \cos\zeta)} = \cos\zeta = 1/\sqrt{1+\omega^2\tau_2^2} \tag{10}$$

The support temperature also can be written as follows: $T_s(t) = b(\cos\zeta \sin\omega t + \sin\zeta \cos\omega t)$.

Once b and ζ are known functions of frequency, they can be used in Eq. (6) from the paper to represent the support temperature $T_s(t)$ to find the amplitude c and phase ϕ of the response $T_m(t)$.

The same approach can then be applied to the equation for the sensor response $T_m(t)$ in response to a real temperature $T(t) = \sin\omega t$. The Laplace transform of Eq. (6) leads to

$$\tau_1(s\mathscr{L}(T_m) - T_m(0)) = a\mathscr{L}(T) + (1-a)b(\cos\zeta \frac{\omega}{s^2+\omega^2} + \sin\zeta \frac{s}{s^2+\omega^2}) - \mathscr{L}(T_m) \tag{11}$$

$$\tau_1(s\mathscr{L}(T_m) - T_m(0)) = \mathscr{L}(T)\left(a + (1-a)\frac{\omega}{\tau_2}\frac{1}{\left(\frac{1}{\tau_2}+s\right)(s+i\omega)(s-i\omega)}\right) - \mathscr{L}(T_m)$$

$$\mathscr{L}(T_m) = \frac{T_m(0) + \left(a + (1-a)\frac{\omega}{\tau_2}\frac{1}{\left(\frac{1}{\tau_2}+s\right)(s+i\omega)(s-i\omega)}\right)\mathscr{L}(T)}{\tau_1(s+1/\tau_1)} \tag{12}$$

The ratio of the output to input transform, if the constant term is dropped, is

$$\mathscr{H} = \frac{a(s+\frac{1}{\tau_2})(s+i\omega)(s-i\omega) + (1-a)\omega/\tau_2}{\tau_1(s+1/\tau_1)(s+1/\tau_2)(s+i\omega)(s-i\omega)} = \frac{a}{\tau_1}\frac{1}{(s+\frac{1}{\tau_1})} + \frac{(1-a)\omega}{\tau_1\tau_2(s+1/\tau_1)(s+1/\tau_2)(s+i\omega)(s-i\omega)}$$

The inverse Laplace transform then leads to the general solution for $T_m(T)$ for sine-wave input:[2]

$$T_m(t) = \frac{T_m(0)}{\tau_1}e^{-t/\tau_1} \tag{13}$$

$$+\mathscr{L}^{-1}\left(\frac{a+(1-a)b\cos\zeta}{\tau_1}\frac{\omega}{(s+1/\tau_1)(s^2+\omega^2)}\right) \tag{14}$$

$$+\mathscr{L}^{-1}\left(\frac{(1-a)b\sin\zeta}{\tau_1}\frac{s}{(s+1/\tau_1)(s^2+\omega^2)}\right) \tag{15}$$

Use partial fractions to write, following Kreyszig [1962] p. 219,

$$\frac{1}{(s+1/\tau_1)(s^2+\omega^2)} = \frac{c_1}{s+1/\tau_1} + \frac{c_2}{s+i\omega} + \frac{c_3}{s-i\omega} \tag{16}$$

[2]Here the notation $\mathscr{L}^{-1}()$ denotes the inverse Laplace transform.

The coefficients c_i are the same as found previously, (2) to (4), except now $\tau = \tau_1$. The Laplace-transform solutions for the three terms in (16), omitting the coefficients, are respectively e^{-t/τ_1}, $e^{-i\omega t}$ and $e^{i\omega t}$. The three terms arising from representing the last term in (13) by partial fractions are the same as the previous term but multiplied by s instead of ω. Because multiplying the Laplace transform by s corresponds to differentiation, differentiating the three terms gives $-(1/\tau_1)e^{-t/\tau_1}$, $-i\omega e^{-i\omega t}$ and $i\omega e^{i\omega t}$, apart from delta functions at $t = 0$ that integrate to constants. Then the solution, without the exponentially decaying terms, is

$$T_m(t) = \frac{a+(1-a)b\cos\zeta}{\tau_1}\omega\left(c_2 e^{-i\omega t} + c_3 e^{i\omega t}\right) + \frac{(1-a)b\sin\zeta}{\tau_1}\left(-c_2 i\omega e^{-i\omega t} + c_3 i\omega e^{i\omega t}\right) \quad (17)$$

This can be transformed to sine and cosine factors using the relationships $e^{i\omega t} = \cos\omega t + i\sin\omega t$ and $e^{-i\omega t} = \cos\omega t - i\sin\omega t$:

$$\begin{aligned} T_m(t) = &\left(\frac{a+(1-a)b\cos\zeta}{1}\right)\left\{\left(\frac{(i-\omega\tau)}{2(1+\omega^2\tau^2)}\right)(\cos\omega t - i\sin\omega t) + \left(\frac{(-i-\omega\tau)}{2(1+\omega^2\tau^2)}\right)(\cos\omega t + i\sin\omega t)\right\} \\ &+ \left(\frac{(1-a)b\sin\zeta}{1}\right) i\left\{\left(\frac{-(i-\omega\tau)}{2(1+\omega^2\tau^2)}\right)(\cos\omega t - i\sin\omega t) + \left(\frac{(-i-\omega\tau)}{2(1+\omega^2\tau^2)}\right)(\cos\omega t + i\sin\omega t)\right\} \\ &= C_1\cos\omega t + C_2\sin\omega t \end{aligned}$$

where

$$C_1 = (a+(1-a)b\cos\zeta)\left(\frac{-\omega\tau}{1+\omega^2\tau^2}\right) + (1-a)b\sin\zeta\left(\frac{1}{1+\omega^2\tau^2}\right)$$

$$C_2 = (a+(1-a)b\cos\zeta)\left(\frac{1}{1+\omega^2\tau^2}\right) + (1-a)b\sin\zeta\left(\frac{\omega\tau}{1+\omega^2\tau^2}\right)$$

In this equation, C_1, C_2, b and ζ are all functions of frequency. It is useful to convert this solution into the form $T_m(t) = c(\omega)\sin(\omega t + \phi(\omega))$, so that the functions $c(\omega)$ and $\phi(\omega)$ represent the amplitude and phase of the response to an input of $\sin\omega t$. This then will characterize the transfer function. The result is that $c(f)\sin(2\pi ft + \phi(f)) = c(f)\left(\sin 2\pi ft\cos\phi(f) + \cos 2\pi ft\sin\phi(f)\right)$ so $C_1 = c\sin\phi$ and $C_2 = c\cos\phi$. Thus, the amplitude and phase of the transfer function defining the response to sine waves of various frequencies are given by

$$c = \sqrt{C_1^2 + C_2^2} \quad (18)$$

$$\phi = \arctan(C_1/C_2) \quad (19)$$

The unheated Rosemount 102A2EL sensor can be represented by the parameters called "set 1" (a =0.73, τ_1 =0.03, τ_2 =0.45), as discussed in Section 2.4.1 of the paper. Figure 2 shows the resulting frequency-dependent transfer function for sine-wave input.

There is a different route to the transfer function that starts with Eq. (3) of the paper, which is an expression for the response to an inverse step function (i.e., a change from unity to zero). The

corresponding response to a positive step is

$$\Theta^*(t) = 1 - A_1 e^{-t/\tau_1} - A_2 e^{-t/\tau_2} \tag{20}$$

The derivative of that response is the impulse function $I(t)$, and the Fourier transform of that impulse function is the transfer function $H(\nu)$:

$$I(t) = (A_1/\tau_1)e^{-t/\tau_1} + (A_2/\tau_2)e^{-t/\tau_2} \text{ for} t \geq 0 \tag{21}$$

$$H(\nu) = \int_0^\infty I(t)e^{-2\pi i\nu t}dt = \frac{A_1}{2\pi i\nu\tau_1 + 1} + \frac{A_2}{2\pi i\nu\tau_2 + 1} \tag{22}$$

In terms of angular frequency ω,

$$H(\omega) = \frac{A_1(1 - i\omega\tau_1)}{(1+\omega^2\tau_1^2)} + \frac{A_2(1 - i\omega\tau_2)}{(1+\omega^2\tau_2^2)} = \left(\frac{A_1}{1+\omega^2\tau_1^2} + \frac{A_2}{1+\omega^2\tau_2^2}\right) - i\omega\left(\frac{A_1\tau_1}{1+\omega^2\tau_1^2} + \frac{A_2\tau_2}{1+\omega^2\tau_2^2}\right)$$

$$|H(\omega)| = \sqrt{C_1^2 + C_2^2} \text{ and } \text{Arg}(H(\omega)) = -\arctan(C_1/C_2)$$

where

$$C_1 = -\omega\left(\frac{A_1\tau_1}{1+\omega^2\tau_1^2} + \frac{A_2\tau_2}{1+\omega^2\tau_2^2}\right) \tag{23}$$

$$C_2 = \left(\frac{A_1}{1+\omega^2\tau_1^2} + \frac{A_2}{1+\omega^2\tau_2^2}\right) \tag{24}$$

The asserted equivalence in coefficients is $A_2 = (1-a)\tau_2/(\tau_2 - \tau_1)$ and $A_1 = 1 - A_2 = (a\tau_2 - \tau_1)/(\tau_2 - \tau_1)$, or $a = A_1(1 - \tau_1/\tau_2) + \tau_1/\tau_2 = A_1 + (1 - A_1)(\tau_1/\tau_2) = 1 - A_2(1 - \tau_1/\tau_2)$. Those substitutions can be used to show the equivalence of (23) and (24) to the corresponding expressions Eq. (13) and Eq. (14):

$$\begin{aligned}
C_1 &= -\omega\left(\frac{1}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\right)\left(A_2\tau_2(1+\omega^2\tau_1^2) + (1-A_2)\tau_1(1+\omega^2\tau_2^2)\right) \\
&=(\text{''})\left(\tau_2^2(1-a)(1+\omega^2\tau_1^2)/(\tau_2-\tau_1) + \tau_1(a\tau_2-\tau_1)(1+\omega^2\tau_2^2)/(\tau_2-\tau_1)\right) \\
&=(\text{''})\left(\frac{a\left(\tau_1\tau_2(1+\omega^2\tau_2^2) - \tau_2^2(1+\omega^2\tau_1^2)\right) + \tau_2^2(1+\omega^2\tau_1^2) - \tau_1^2(1+\omega^2\tau_2^2)}{\tau_2 - \tau_2}\right) \\
&=(\text{''})\left(\frac{a\left(\tau_2(\tau_1-\tau_2) + \tau_1\tau_2^2\omega^2(\tau_2-\tau_1)\right) + (\tau_2^2 - \tau_1^2)}{\tau_2 - \tau_1}\right) \\
&= \left(\frac{-\omega}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\right)\left(a\tau_2(-1+\tau_1\tau_2\omega^2) + \tau_2 + \tau_1\right)
\end{aligned}$$

$$
\begin{aligned}
C_2 &= \left(\frac{1}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\right)\left(A_2(1+\omega^2\tau_1^2)+(1-A_2)(1+\omega^2\tau_2^2)\right)\\
&=\left(\frac{\tau_2}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\right)\left(\frac{1-a}{\tau_2-\tau_1}(1+\omega^2\tau_1^2)+\frac{a-\tau_1/\tau_2}{\tau_2-\tau_1}(1+\omega^2\tau_2^2)\right)\\
&=\left(\frac{\tau_2}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\right)\left(\frac{a(1+\omega^2\tau_2^2-1-\omega^2\tau_1^2)+1+\omega^2\tau_1^2-(\tau_1/\tau_2)(1+\omega^2\tau_2^2)}{\tau_2-\tau_1}\right)\\
&=\left(\frac{\tau_2}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\right)\left(\frac{a\omega^2(\tau_2^2-\tau_1^2)}{\tau_2-\tau_1}+\frac{1-\tau_1/\tau_2+\omega^2\tau_1(\tau_1-\tau_2)}{\tau_2-\tau_1}\right)\\
&=\left(\frac{1}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\right)\left(a\omega^2\tau_2(\tau_2+\tau_1)+1-\omega^2\tau_1\tau_2\right)
\end{aligned}
$$

which are the same as Eqs. (13) and (14) in the paper, obtained from the differential equation using

$$\tan\zeta = -\omega\tau_2$$

$$\cos\zeta = \frac{1}{\sqrt{\tan^2\zeta+1}} = \frac{1}{\sqrt{1+\omega^2\tau_2^2}} = b$$

$$\sin\zeta = \sqrt{1-\cos^2\zeta} = \frac{-\omega\tau_2}{\sqrt{1+\omega^2\tau_2^2}} = -b\omega\tau_2$$

$$
\begin{aligned}
C_1^* &= \frac{-\omega}{(1+\omega^2\tau_1^2)}\left(\tau_1 a+\frac{(1-a)(\tau_1+\tau_2)}{(1+\omega^2\tau_2^2)}\right)\\
&=\frac{-\omega}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\left(a(\tau_1(1+\omega^2\tau_2^2)-(\tau_2+\tau_1))+(\tau_1+\tau_2)\right)\\
&=\frac{-\omega}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\left(a\tau_2(-1+\tau_1\tau_2\omega^2)+\tau_1+\tau_2\right)
\end{aligned}
$$

$$
\begin{aligned}
C_2^* &= \left(\frac{1}{1+\omega^2\tau_1^2}\right)\left(a+\frac{(1-a)(1-\omega^2\tau_1\tau_2)}{(1+\omega^2\tau_2^2)}\right)\\
&=\frac{1}{(1+\omega^2\tau_1^2)(1+\omega^2\tau_2^2)}\left(a\omega^2\tau_2(\tau_2+\tau_1)+1-\omega^2\tau_1\tau_2\right)
\end{aligned}
$$

The code chunk "checkImpulseFnSoln" (in this project archive) was used to check that the expressions used for the transfer function were indeed equivalent. An example comparison is shown in Fig. 1.

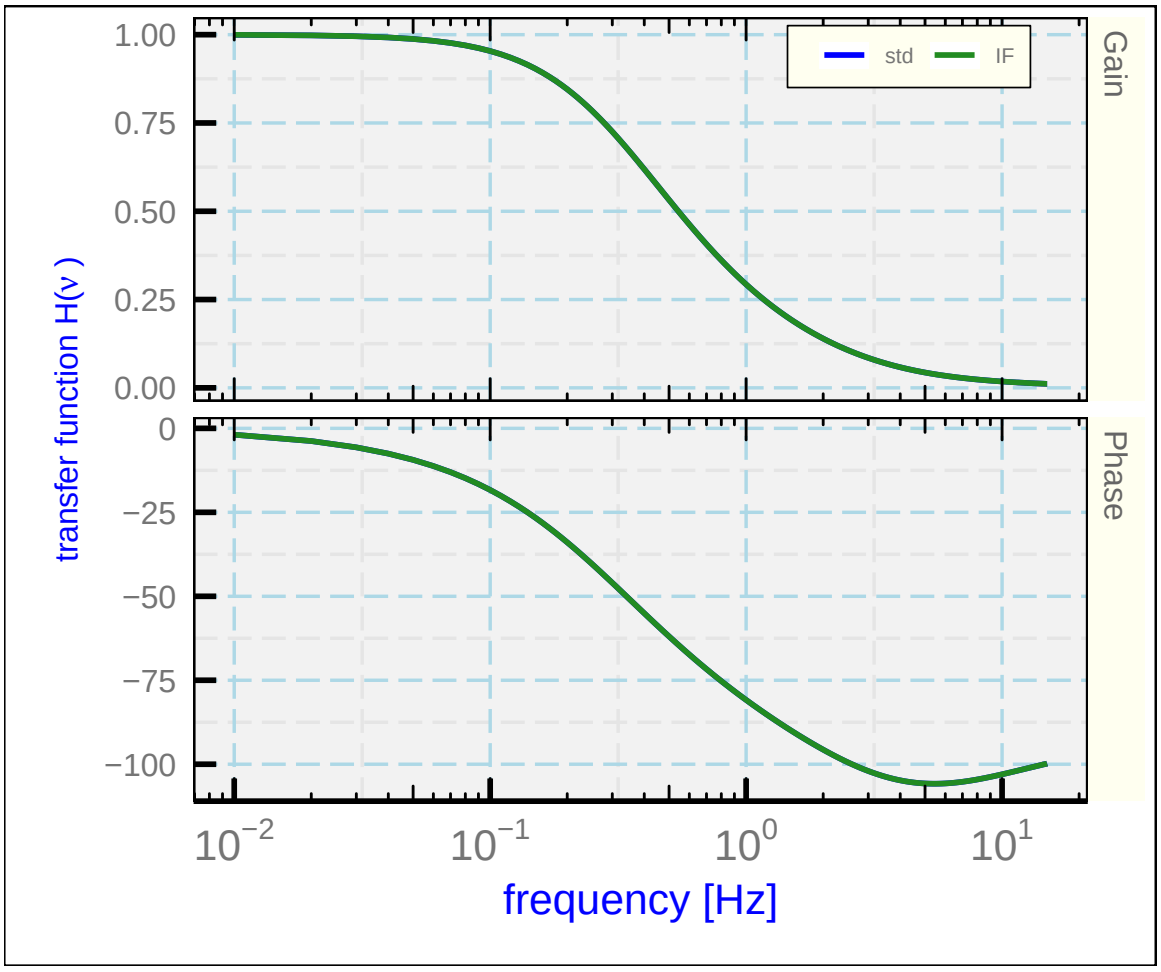

Figure 1: Example transfer function for a=0.5, τ_1=0.05, τ_2=0.5, for the Laplace-transform solution ("std") and for the corresponding solution obtained from the impulse-response function ("IF"). The lines overlap in this plot and were verified to agree to near machine-precison limits.

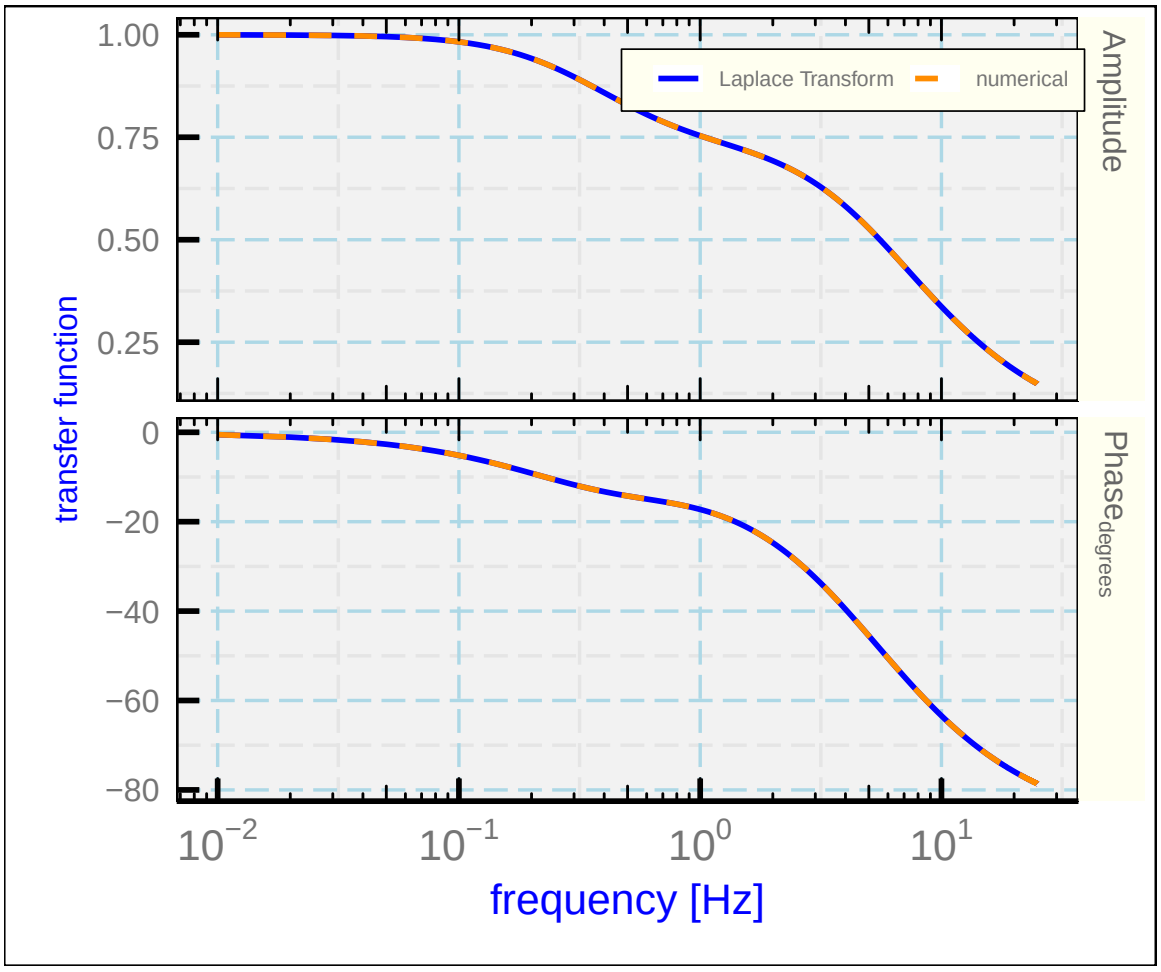

Figure 2: The amplitude and phase for the response of the Rosemount 102E4AL temperature sensor to a signal of the form $\sin(2\pi ft)$ where f is the frequency, for the Laplace-transform solution and for a numerical solution of simultaneous equations.

As still another check on the preceding derivations, an alternate solution was obtained as follows. Again isolate the factors multiplying $\sin \omega t$ and $\cos \omega t$:

$$c\omega\tau_1[\cos\omega t\cos\phi - \sin\omega t\sin\phi]$$

$$= a\sin\omega t + (1-a)b(\cos\zeta\sin\omega t + \sin\zeta\cos\omega t) - c[\cos\phi\sin\omega t + \sin\phi\cos\omega t]$$

Gathering terms:

$$[c\omega\tau_1\sin\phi + a + (1-a)b\cos\zeta - c\cos\phi]\sin\omega t$$

$$= [c\omega\tau_1\cos\phi - (1-a)b\sin\zeta + c\sin\phi]\cos\omega t$$

The simultaneous equations to be solved are then obtained by setting the factors in square brackets to zero:

$$[c\omega\tau_1\sin\phi + a + (1-a)b\cos\zeta - c\cos\phi] = 0 \tag{25}$$

$$[c\omega\tau_1\cos\phi - (1-a)b\sin\zeta + c\sin\phi] = 0 \tag{26}$$

Because it is assumed that a is known and because b and ζ are known from the solution above, the two unknowns for which (25) and (26) must be solved are c and ϕ, giving the amplitude and phase of the response to the input. They must be solved independently at each frequency because all the parameters $\{b, \zeta, c, \phi\}$ are functions of frequency. R code for this solution using the R package "nleqslv" is included in the "Rnw"-format file used to generate this workflow memo. The results of the two methods of solution are the same to nearly the numerical precision of the software.

One final check was to simulate sine waves of various frequencies and then use a numerical solution of Eq. (6) to calculate the expected measurement. The results were consistent with Fig. 2; for example, the 1 Hz simulation agreed in amplitude to within 1% and in phase within 1° even for a simple Euler-method integration.

The following is a final verification of the solution by substituting into the differential equations. For (3),

$$b\omega\tau_2\cos(\omega t + \zeta) \stackrel{?}{=} \sin(\omega t) - b\sin(\omega t + \zeta)$$

$$b\omega\tau_2(\cos\omega t\cos\zeta - \sin\omega t\sin\zeta) \stackrel{?}{=} \sin(\omega t) - b(\sin\omega t\cos\zeta + \sin\zeta\cos\omega t)$$

$$\cos\omega t(b\omega\tau_2\cos\zeta + b\sin\zeta) \stackrel{?}{=} \sin(\omega t)(b\omega\tau_2\sin\zeta + 1 - b\cos\zeta)$$

The asserted solution gives zero for the coefficient multiplying $\cos\omega t$ on the left side. On the right side, the coefficient multiplying $\sin\omega t$ is $1-b\cos\zeta(\omega^2\tau_2^2+1)$ or, using $\cos\zeta = 1/\sqrt{1+\tan^2\zeta} = 1/\sqrt{1+\omega^2\tau_2^2}$, $1-b\sqrt{1+\omega^2\tau_2^2}=0$ so the equality is satisfied.

For (4), the approach is similar:

$$-C_1\omega\tau_1\sin\omega t + C_2\omega\tau_1\cos\omega t$$
$$\stackrel{?}{=} \{a\sin\omega t + (1-a)b(\sin\omega t\cos\zeta + \cos\omega t\sin\zeta)\} - C_1\cos\omega t - C_2\sin\omega t$$

Again gathering coefficients multiplying $\sin\omega t$ and $\cos\omega t$ separately:

$$\cos\omega t\,(C_2\omega\tau_1 + C_1 - (1-a)b\sin\zeta) \stackrel{?}{=} \sin\omega t\,(a + (1-a)b\cos\zeta + C_1\omega\tau_1 - C_2)$$

Substituting the solutions for C_1, C_2, b and ζ gives, for the left side coefficient, after dividing by $b\cos\zeta$,

$$\left((\frac{a\sqrt{1+\omega^2\tau_2^2}}{\cos\zeta}+(1-a))\left(\frac{1}{1+\omega^2\tau_1^2}\right)-(1-a)\omega\tau_2\left(\frac{\omega\tau_1}{1+\omega^2\tau_1^2}\right)\right)\omega\tau_1$$
$$+\left((\frac{a\sqrt{1+\omega^2\tau_2^2}}{\cos\zeta}+(1-a))\left(\frac{-\omega\tau_1}{1+\omega^2\tau_1^2}\right)-(1-a)\omega\tau_2\left(\frac{1}{1+\omega^2\tau_1^2}\right)\right)$$
$$+(1-a)\omega\tau_2$$

which is zero. The right-side coefficient becomes, after similar division,

$$\frac{a\sqrt{1+\omega^2\tau_2^2}}{\cos\zeta}+(1-a)+\left((\frac{a\sqrt{(1+\omega^2\tau_2^2}}{\cos\zeta}+(1-a))\left(\frac{-\omega\tau_1}{1+\omega^2\tau_1^2}\right)-(1-a)\omega\tau_2\left(\frac{1}{1+\omega^2\tau_1^2}\right)\right)\omega\tau_1$$
$$-\left((\frac{a\sqrt{(1+\omega^2\tau_2^2}}{\cos\zeta}+(1-a))\left(\frac{1}{1+\omega^2\tau_1^2}\right)-(1-a)\omega\tau_2\left(\frac{\omega\tau_1}{1+\omega^2\tau_1^2}\right)\right)$$
$$=\frac{a\sqrt{1+\omega^2\tau_2^2}}{\cos\zeta}+(1-a)\left(1-\frac{1+\omega^2\tau_1^2}{1+\omega^2\tau_1^2}\right)-\frac{a\sqrt{(1+\omega^2\tau_2^2}}{\cos\zeta}\left(\frac{1+\omega^2\tau_1^2}{1+\omega^2\tau_1^2}\right)$$

here is awhich is also zero. These results then verify that the differential equation is satisfied for the case where the true signal is a continuous sine wave.

Comments regarding Fig. 1: The solution in Sect. 2.1 of the paper specifies the amplitude and phase of the response to a unit-amplitude sinusoidal input, and so specifies the magnitude and phase of the transfer function. This section uses assumed values for the response parameters (a, τ_1 and τ_2) as examples because those parameters have not been determined yet, but the values are those that result from subsequent analysis. The function "LTphase(f, P)", included in the code chunk labelled "LTsolution", was constructed to represent the solution so that it could be used throughout the paper. Its arguments are the frequency "f", which can be a vector, and the values of the parameters, provided as "P" which should be a list with named components "a", "tau1" and "tau2." The function returns a list with two component vectors representing the amplitude (or gain) and phase at the specified frequencies.

It is useful for the "LTphase()" function to be able to accept a frequency of zero because conventional FFT results in R have the first coefficient equal to the constant for zero frequency. The function appropriately returns gain = 1 and phase = 0 for zero frequency. When given negative frequencies, it returns the complex conjugates of the values at the corresponding positive frequencies; i.e., the same amplitudes but (-1) times the phase.

The two-component faceted plot of the transfer function (Fig. 1) is generated by "Ranadu::ggplotWAC()", a plot function based on the "ggplot2" package (Wickham [2009]) that combines the plots of amplitude (or gain) and phase of the transfer function into one plot. See "help(ggplotWAC)" in R for information on the use of this routine to generate plots with multiple panels like this, and the "ggplot2" book in the reference list for complete information on using that package.

At the end of Sect. 2.1 there is a brief discussion of how the transfer function can be used to correct the measurements to compensate for the time response of the sensor. This is included here as a brief reference because the correction procedure is used in some subsequent sections. It is developed in more detail in Appendix A, but that discussion is relegated to an appendix because the correction methods are straightforward and have been used and discussed in previous work.

6.2 Re: Section 2.2 (The response to dynamic heating):

Section 2 determines the transfer function for some representative airborne thermometers by observing the response to fluctuations in dynamic heating. The first equation (Eq. (15)) is so complicated because the simple version ($\alpha_r V^2/(2C_p)$) uses the measured true airspeed (V) and the standard calculation of that involves the ambient temperature. If there are errors in the measurement of ambient temperature, they enter the estimate of Q. For that reason, the expression is revised to depend only on the absolute recovery temperature (T_r) and other factors that can be determined without reference to the ambient temperature. This makes possible the iteration process used here.

6.3 Re Sect. 2.3, Data Sources

No additional information seems needed here. Someone wanting to reproduce this work can request the data files as described here. This work is all based on publicly available data sets that are preserved with associated Document Object Identifiers (DOIs).

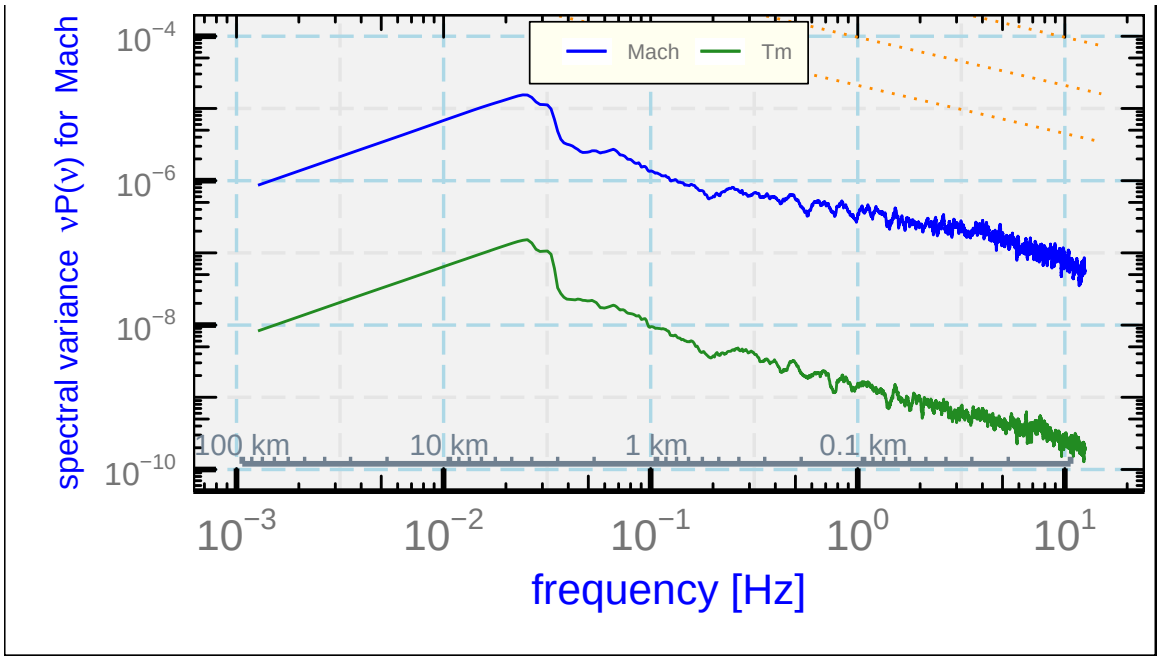

Figure 3: Frequency-weighted spectral variance for $\delta T_m/T_m$ and $0.4\delta M/M$ as functions of frequency (ν) for a low-level flight segment from VOCALS flight 3, 21 Oct 2008 11:39:00 – 11:52:00 UTC. The wavelength scale shows the correspondence between frequency and wavelength at the average airspeed. The two terms are labeled "Tm" and "Mach" in the legend.

6.4 Re Sect. 2.4, Fits to the Measurements

The initial few paragraphs are justification for using the dynamic-heating term to characterize the response of the temperature sensor. In the course of investigating this, some plots were constructed that were not included in the final paper. An example was the variance spectra for the two terms on the right side of Eq. (16). That figure is included here as Fig. 3. It shows the extent to which the second term dominates over the first for the entire frequency range. That helps justify attributing the variance in dynamic heating primarily to the variance in the Mach number, so that to a first approximation the true variance in the temperature-dependent first term in Eq. (16) can be neglected.

6.4.1 Re. Sect. 2.4.1: An unheated temperature sensor

The danger in the approach taken is that, if fluctuations in dynamic heating (via fluctuations in airspeed) are not the dominant source of variation in the measured recovery temperature (T_m) then the measured phase and amplitude ratio for the cross-spectrum of T_m and Q will be contaminated by other causes of fluctuation in the measured recovery temperature. Indeed, if such fluctuations in T_m are large compared to fluctuations in Q then the amplitude ratio determined from the variance spectra will not reflect the transfer function and cannot be used to determine it. Therefore regions

and frequency intervals were sought where it appeared that the fluctuations in T_m arose primarily from fluctuations in Q. This should be the case where the intensity of turbulence is high and would produce much larger fluctuations in T_m than the existing fluctuations in ambient temperature.

The selection of data and combination into a unified data set is described in the text of the paper and won't be repeated here. Some trial-and-error investigation of the number of bins to use in the construction of Fig. 2 in the paper led to the choice of 200 bins, which determines the number of plotted points and error bars in the plot. Although it leads to reduced clarity in the plots, a large number of bins was settled upon to avoid significant distortion of the mean when calculated for a region where the variable changes rapidly, as for example approaching 10 Hz in this figure. Too large an interval, with the average defined at the mid-point of the interval, could distort the means toward higher values compared to the true distribution and hence would bias the determination of response variables. These plots have a more convincing appearance when a smaller number of bins is used, and became vary noisy if a much larger number of bins is used, so 200 bins was the compromise used in the paper for the unheated Rosemount sensor. Because the data sets used for other sensors were much smaller, smaller numbers of bins were used for those sensors.

A modified version of the Ranadu function "CohPhase()", named "CohP()" in the .Rmd code, was used to combine the measurements of phase and amplitude for the flight segments listed in Table 1 of the paper. That function was modified to return a special data.frame containing the frequency limits of the bins and, for each frequency in the values returned by R function "spec.pgram()", the assignment of that frequency to one of the 200 logarithmically spaced bins in the phase and amplitude of the two spectra at that frequency. This function "CohP()" was called for each of the six ten-minute flight segments, with the measured recovery temperature and the expected forcing by dynamic heating (i.e., T_m and Q) and then the results from all calls were averaged in each of the frequency bins. The result was an average value (with standard deviation) for the phase and amplitude ratio, as needed to determine the transfer function. The observed phase at each bin-average frequency then was plotted in Fig. 2a to show the results for the phase. (The measurement used for T_m was "TTRR" from the netCDF file.)[3] Two-standard-deviation values for the standard deviation of the mean are plotted because one-standard-deviation limits appeared hard to distinguish in this plot. The error-bar plot was generated using the ggplot2 function "geom_errorbar()" with the limits specified using the bin-average mean values and standard deviations.

The plotted "theoretical response" was generated by "LTphase()" using the standard values of the response parameters as found later by fitting to these values and the corresponding values for the amplitude ratio. The plotted frequencies are limited to those above 10^{-2}Hz because, below that frequency, bins had too few independent measurements to provide good statistics, although the available measurements below 0.01 Hz were consistent with zero phase shift as would be expected from the theoretical response. The frequency interval was also limited to <12 Hz because the highest-frequency points in the plot otherwise departed significantly from the trend by being much higher (about -43°) and so appearing to be outliers that should be excluded from the fit. The

[3] At the time of the VOCALS project the recovery temperatures had names like TTRR instead of RTRR. Those names were changed subsequently to reflect that these are better described as recovery temperatures rather than total temperatures. In this analysis, the variable TTRR has been renamed RTRR to reflect the current usage. There is no variable RTRR in the referenced data files.

highest frequency included in the plot also shows some possible deviation and maybe should also be excluded.

Figure 2b was generated in similar fashion, using the ratio of amplitudes (the square roots of the ratio of variance spectra) in place of the phase. For the fit the calculation of the chi-square included the deviations from the experimental phases and the experimental amplitude ratios with equal weights, except that the experimental amplitude ratios were only included for the frequency range from 0.1 to 3 Hz. This range was selected because outside that range the values departed significantly and systematically from the theoretical prediction. The R function "optim()", part of the standard "stats" package, was used for this fit. In the case of the unheated Rosemount sensor, the fit used the default Nelder-Mead minimization method without imposed constraints on the values of the three parameters. This function returned a calculated Hessian and that was used to determine estimates of the uncertainty in the parameters. The function reported that it converged to a solution, and at the point of convergence the chi-square was 3892 for 218 degrees of freedom. In the calculation of the reported uncertainty limits the reported Hessian was divided by 4 to account for the use of two-standard-deviation standard deviations and then the square roots of the diagonal elements of the inverted Hessian matrix were reported. (This ignores significant correlations among the error estimates that perhaps should be considered and reported.)

The paper then reports on an iteration in which the measured recovery temperature was first corrected using the parameters as determine in the fit and then the fit was repeated. This process was performed outside the .Rmd code by changing the dynamic-heating term in lines 866 to 871 to "DH2", which was calculated by using the corrected "RT" variable in place of the measured variable ("TTRR"). The result produced only a small change to the fit parameters, within estimated uncertainty, even after only one iteration so the iteration procedure was not included in the processing code. Instead, the values obtained after iteration were those quoted in the paper.

At the end of the discussion of the unheated sensor, there is mention of a similar evaluation performed on the GV. The .Rmd file includes code for this, but the results were not included in the paper except to quote the results. The relevant code chunk is labelled "GVcheck". That code will generate figures similar to Fig. 2 and will fit to the observations to determine the fit parameters.

6.4.2 The heated Rosemount sensor

A study of the heated Rosemount sensor was included in the original file but excluded from the paper to reduce its length. The sensor is seldom used and has largely been replaced by the heated HARCO sensor, which is designed to have the same characteristics as the original. In the .lyx file the inclusion of this material is controlled by the "Extra" branch which is not included in the paper but could be added by activating this branch.The results are included in the summary of parameters (Table 3). Here is the text that would have been included in the paper but was excluded:

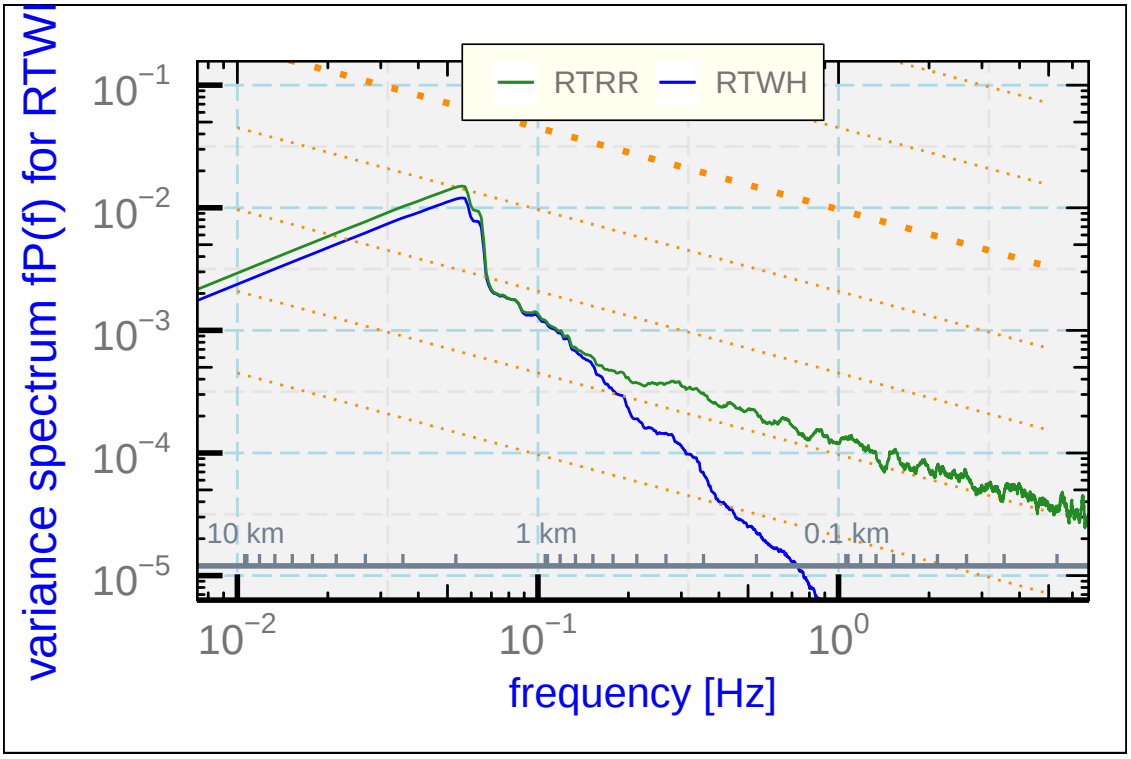

Figure 4: Variance spectrum for the recovery temperature from the heated Rosemount sensor (RTWH) and for the unheated Rosemount 102E4AL sensor (RTRR), for a low-level flight segment in the marine boundary layer.

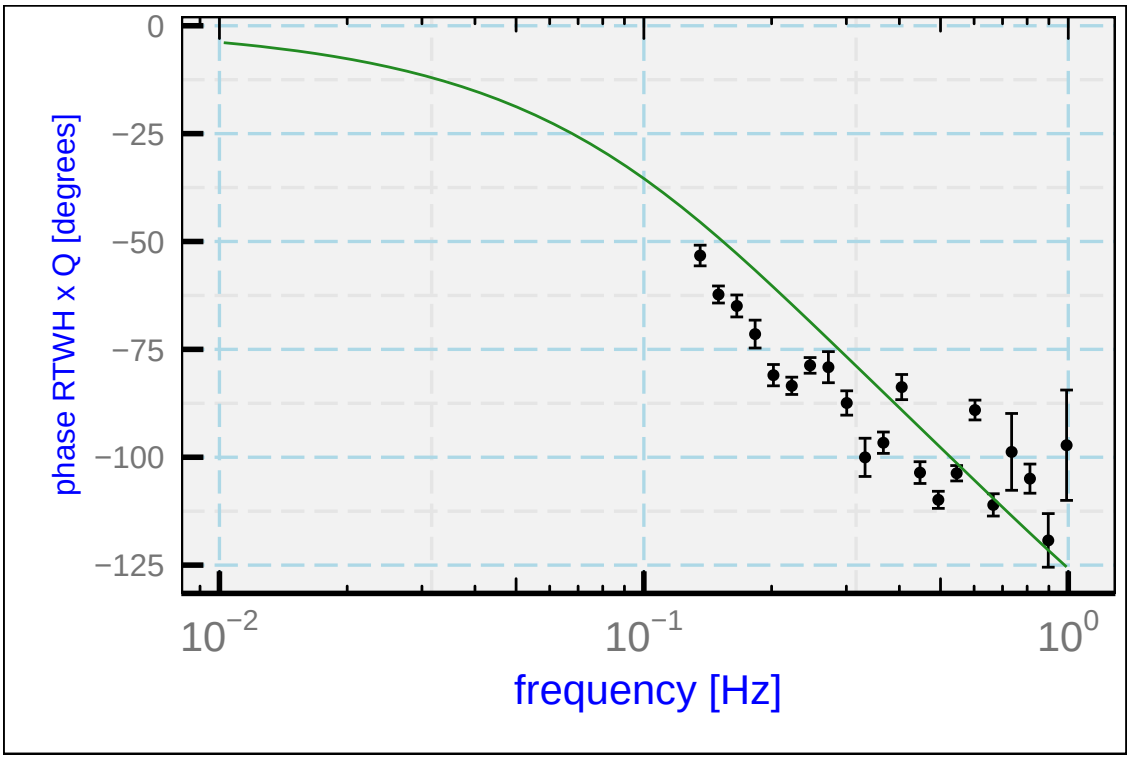

Figure 5: Phase lag of recovery temperature vs.\\ dynamic heating, for the measurements (error bars) from a Rosemount heated sensor, and the theoretical response for the best-fit parameters to the measurements with frequency below 1 Hz (green line).

The heated Rosemount sensor

During the same flight used to study the unheated sensor, a heated Rosemount sensor was also present. Data were assembled as for the unheated sensor and the preceding analysis was repeated, except that two of the flight segments were omitted because they led to erratic indications of the phase. The heated sensor had almost no response to any fluctuations above 1 Hz, as shown in Fig. 3.

The measured phase between the measured recovery temperature and the calculated dynamic heating is shown in Fig. 2. Above 1 Hz, the lack of response led to erratic estimates of the phase and the coherence between the measured recovery temperature and dynamic heating was consistent with zero, so those measurements of the phase were omitted from the fit. Similarly, the amplitude for frequencies above 1 Hz was not used in the fit, and measurements of amplitude for frequency below 0.13 Hz were similarly omitted because for these frequencies there appeared to be significant fluctuations in recovery temperature caused by real changes in air temperature and not primarily changes in dynamic heating. The best-fit parameters so obtained were a =0, τ_1 =0.16 s and τ_2 =0.9 s; these values are the basis for the theoretical lines in Figs. 4 and 5.[4]

[4]The fit was constrained to keep $a \geq 0$. Still smaller values of the χ^2 were obtained for negative a, but this would

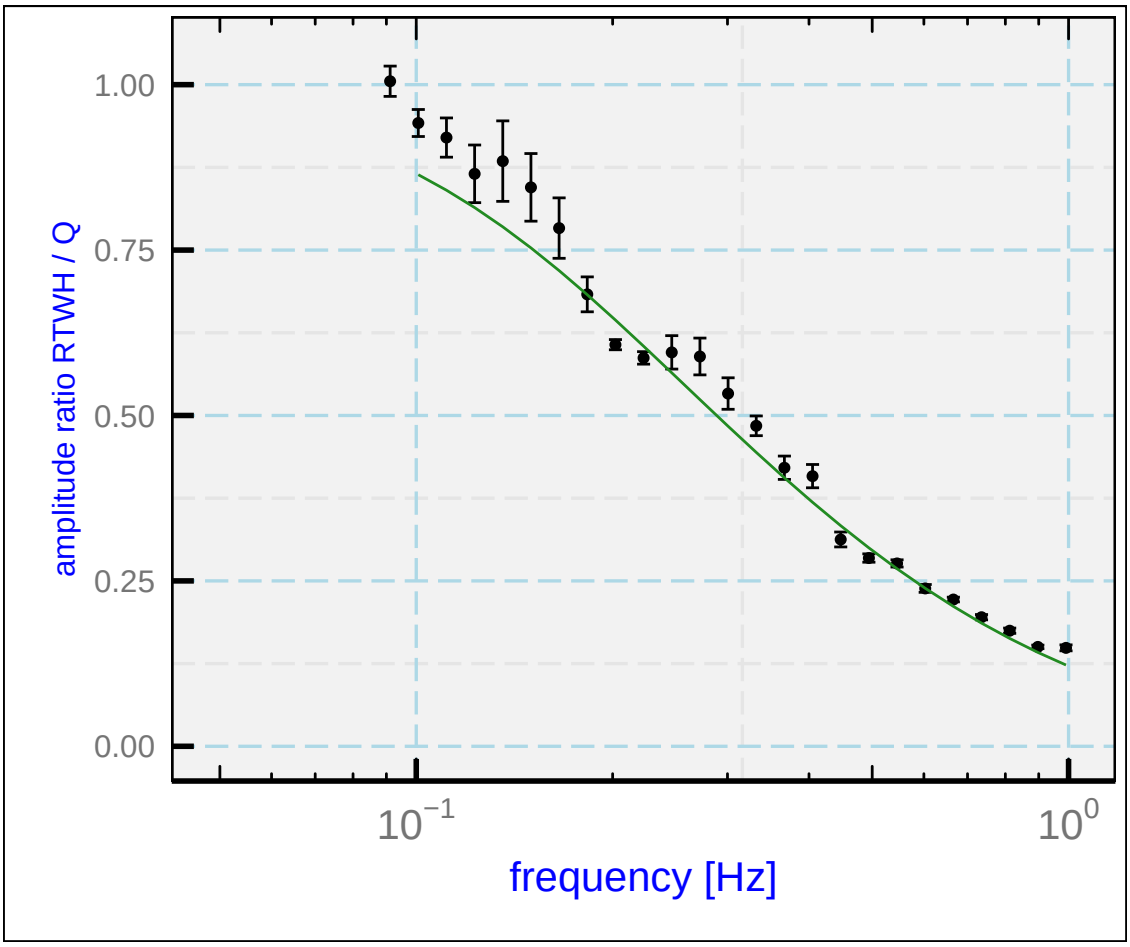

Figure 6: The ratio of the spectral amplitude for the measurement of recovery temperature ($T_m(t)$) from the heated Rosemount sensor to that for dynamic heating (Q), shown as the plotted data points. The green line is the prediction from the transfer function determined from the best-fit values matching the phase lag and amplitude ratio between these variables.

The phase shift leads to these conclusions:

1. A two-time-constant formula does not provide a very good representation of the low-frequency portion of the observations that is between 0.01 and 0.04 Hz. This may indicate that still another time constant is involved in the response of this sensor and that the two-equation representation provided by Part 1 Eqns. (3) and (4) is incomplete.

2. The observed phase shift falls below $-90°$ above about 0.3 Hz. This is impossible for a single-time-constant sensor but is possible, as the theoretical line shows, for a two-time-constant representation of the response, so it is conclusive that at least two time constants are involved. This also indicates that wrong-sign contributions to sensible-heat flux would be measured by this sensor above about 0.3 Hz.

3. The low value of a, 0, indicates that the primary heat transfer from the wire is not to the air but to the support. For a equal to zero, the differential equations TN Eq. (3) and (4) do not separately constrain τ_1 and τ_2. The fit results show complete negative correlation for the errors in these parameters, so they cannot be determined independently. The result, however, is not equivalent to a single time constant, as demonstrated by the measured phase shifts below $-90°$. The solution to the differential equations with $a = 0$ becomes

$$C_1 = b\cos\zeta\left(\frac{-\omega\tau_1}{1+\omega^2\tau_1^2}\right) + b\sin\zeta\left(\frac{1}{1+\omega^2\tau_1^2}\right)$$

$$C_2 = b\cos\zeta\left(\frac{1}{1+\omega^2\tau_1^2}\right) + b\sin\zeta\left(\frac{\omega\tau_1}{1+\omega^2\tau_1^2}\right)$$

$$c = \sqrt{C_1^2+C_2^2} = 2b^2\frac{1+\omega^2\tau_1^2}{(1+\omega^2\tau_1^2)^2} = 2\left(\frac{1}{1+\omega^2\tau_2^2}\right)\left(\frac{1}{1+\omega^2\tau_1^2}\right)$$

$$\tan\phi = (C_1/C_2) = \frac{-\omega\tau_1\cos\zeta+\sin\zeta}{\cos\zeta+\omega\tau_1\sin\zeta} = \frac{\tan\zeta-\omega\tau_1}{1+\omega\tau_1\tan\zeta} = \frac{-\omega(\tau_2+\tau_1)}{1-\omega^2\tau_1\tau_2}$$

These equations show that, for any solution, the values of τ_1 and τ_2 can be exchanged and the solution will remain the same. The particular solution found was influenced by the initial parameters for the search function, but reversed value for τ_1 and τ_2 could be obtained for other initial conditions. Nevertheless, it appears that the best interpretation is that the support responds more slowly, with τ_2 =0.9, and the sensing wire responds faster with time constant τ_1 =0.16.

The transfer function for this sensor is shown in Fig. 4. It does not appear to be feasible to measure sensible-heat flux with this sensor, even with the correction procedures to be developed in the Appendix, because above about 1 Hz there does not appear to be enough signal to amplify by using the transfer function. Also, the phase shift is less than $-90°$ above this frequency.

(this is the end of the section re the heated Rosemount sensor, excluded from the paper.)

be inconsistent with the assumed model of heat transfer to the wire.

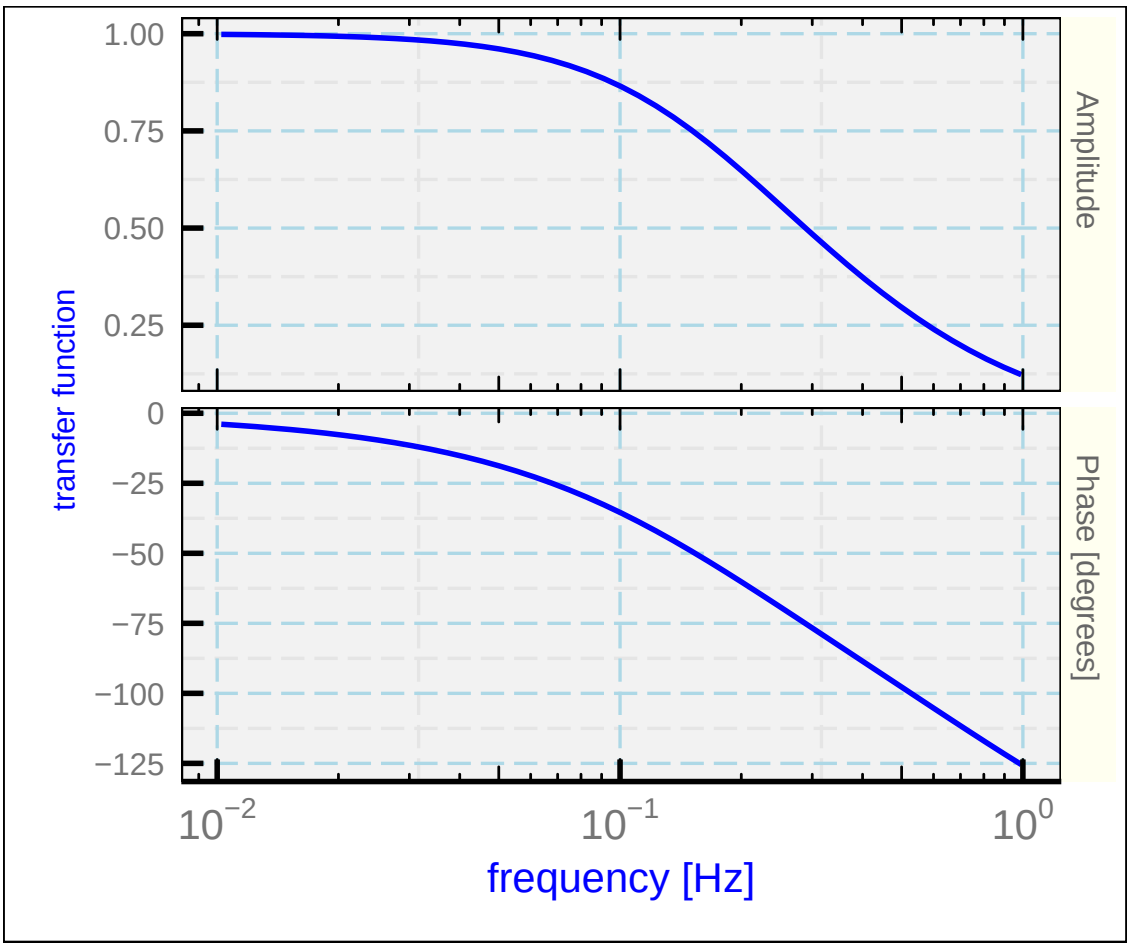

Figure 7: The transfer function for the heated Rosemount sensor on the C-130 research aircraft, based on the best-fit parameters that match the observed phase shift relative to dynamic heating.

6.4.3 Comments on Sect. 2.4.2, Heated sensors

The first try was based on WECAN data, but the results did not seem to give a response dominated by dynamic-heating forcing. That code still resides in the .Rmd file, along with an early analysis based on limited SOCRATES data, but those were superceded by a set of flight segments from the SOCRATES and CSET experiments, and that is the data set used in the paper to characterize the HARCO sensor. Procedures followed the pattern explained above for the unheated Rosemount sensor, and a transfer function for the HARCO was plotted as Fig. 4 in the paper using the same procedures. The fit provided by Eqs. (17)–(19) has no theoretical basis, but it provides a reasonable fit to the observations and is a considerable improvement over the three-parameter fit. The variable ν_0 was defined only to provide a dimensionless argument to the log() function.

6.4.4 Comments on Sect. 2.4.3, Expected dependence on flight conditions

The dependence attributed to Stickney et al. [1994] was obtained from my approximation to the central line in their Fig. 12. The uncertainty in that estimate is large; slopes ranging from about -0.6 to -0.75 fall within the uncertainty limits of the plot. Collis and Williams [1959] and Khan et al. [2004] indicate that the Nusselt number should vary as $\mathrm{Re}^{0.5}$ for a cylindrical wire, suggesting that τ_1 should vary as $\mathrm{Re}^{-0.5}$, but that does appear to be outside the likely uncertainty limits in the referenced figure. It would seem appropriate to use Re at the sensor wire rather than the free-stream Re – i.e., use the airspeed, air density, and dynamic viscosity μ evaluated for recovery conditions: $V_p = \sqrt{1-\alpha_r}V$, $p_p = p+q \approx p(1+M^2/5)^{-2/7}$, $\rho_p = p_p/(R(273.15+T_r))$, leading to

$$\mathrm{Re}^* = \rho_p V_p d/\mu$$

where the temperature dependence of μ is approximately proportional to $\sqrt{273.15+T_r}$. Because of this, the Stickney et al. parameter Z has almost the same temperature dependence, because $Z = M\rho/\rho_0 = V\rho/(\rho_o\sqrt{\gamma R T_K})$ with $T_k = T+273.15$. It is then possible to compare the ratios of Re^* and Z for different flight conditions. For $Z = 0.3$, Re^* is about 75, so it is useful to normalize both to those values and then consider how the time constant would vary with either $(Z/0.3)^{-0.68}$ or $(\mathrm{Re}^*/75)^{-0.5}$. Appropriate flight conditions against which to test this prediction were difficult to find, but one candidate was from the "DC3" project, flight 11, 21:00 to 22:00 UTC, a flight segment at about 11.5 km altitude. An estimate of τ_1 from that flight leg was $\tau_1 = 0.037$ s, while $Z^{-0.68}$ predicts 0.036 and $(\mathrm{Re}^*)^{-0.5}$ predicts about 0.033. The results aren't very different but they seem closer to the Stickney representation. Therefore, it seems better to stay with the Stickney representation, which was based on wind-tunnel measurements with the sensor rather than idealized-cylinder results. It was interesting that there did not seem to be a similar variation in τ_2; if anything, the results favored a smaller value. Using the expected dependence for the sensor is probably not appropriate for the support, so no change was suggested for the second time constant, but it would be useful to find more information on that time constant. This would be a useful topic for future work.

6.5 Sect. 2.5, response to a step change

The VOCALS project provided a good opportunity to search for discontinuities topping the marine boundary layer because most flights included numerous climbs and descents through that structure. A large number of these were examined in a search for sharp transitions, but almost all showed evidence of mixing at the top and were not sharp as would be desired to test response to a step change. The case selected was the best case, and indeed the only good case, from about 50 examined. Some other projects were also considered, but good cases where the unheated Rosemount was in use were not found. It appears that this is a difficult way to test the response characteristics of the sensor, although other situations (terrestrial boundary layer, frontal surface, etc.) might provide better opportunities.

6.6 Application to a "speed run"

Initial drafts of this paper included a section showing how the sensor response introduces hysteresis into measurements during a "speed run", where the airspeed changes during level flight and causes a change in dynamic heating. This material is included here but was excluded from the submitted version of the paper.

An example is shown in Fig. 8, where the airspeed was increased steadily in level flight from near the lower limit of the flight envelope to near the upper limit and then was decreased back to the starting value. A plot of recovery temperature should also increase and decrease as the dynamic heating changes, but with a lag caused by the sensor response. This lag will produce hysteresis in the measured temperature during the speed run.

Figure **??**a shows this hysteresis, which appears as the difference between the segment with increasing speed and that with decreasing speed. A correction based on simply shifting the measurements in time works reasonably but doesn't take into account that the Mach number and hence the time parameters vary significantly during the speed run. A better test of the time-response parameters is to apply the first correction scheme outlined in the Appendix to the measurements, with varying response parameters dependent on the Mach number. The specific correction equation used is Eq. (A2) for the heated sensor. Figure 9b shows that the delay is mostly removed by this procedure. The residual standard deviation about the regression fit for recovery temperature as a function of $V^2/(2C_p)$ is reduced from 0.26°C before correction to 0.10°C after correction. The minimum standard deviation results from increasing the time constants an additional 10%, so measurements from this speed-run maneuver are consistent with the predicted time response as found in Sect. 2.4 of the paper to within about this uncertainty.

The initial approach was to shift the measurements in time to minimize the standard deviation about the fit, but the problem with this is that the Mach number varies significantly during the speed run so the appropriate time shift should also vary significantly. A confusing and unresolved conflict with this measurement is that delaying the temperature measurements by 2.25 s produces the minimum residual standard deviation in the fit, but that is significantly longer than the predicted

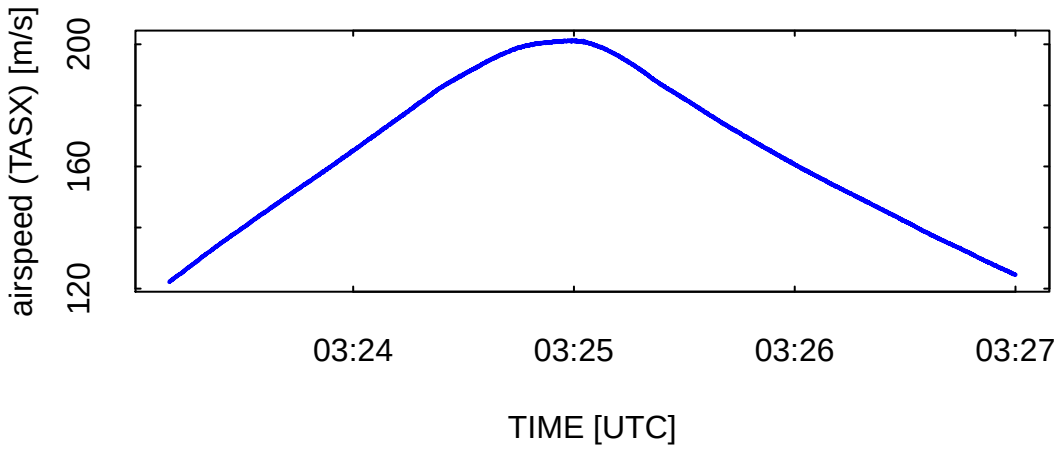

Figure 8: History of the airspeed during a "speed-run" maneuver where the airspeed varied during level flight over the available speed range of the aircraft.

shift of about 1.35 s. There may be some additional timing error that affects the measurement, but none has been found. Inconsistency in these results led to omission of this evidence from the paper.

7 Comments on Section 3: Correcting for Dynamic Heating

7.1 The Digital Filter Coefficients

The paper presents only a cursory discussion of the digital filter used to correct dynamic heating for the response of the sensor. Here some additional details are included as well as the specific coefficients used. The filter was specified in the code chunk labeled “designFilter”, following suggestions from Press et al. [1992]. These steps were followed:

1. The impulse function was found from the inverse Fourier transform of the transfer function. For 25 Hz measurements, the transfer function was evaluated at frequencies in the range 0–12.5 Hz with a resolution of (1/600) Hz and at the corresponding negative frequencies. This solution is stored in a vector with frequencies in the order 0–12.5 Hz, then -12.5 to $-(1/600)$ Hz as needed for the inverse Fourier transform.

2. The resulting impulse function then has 15,000 values, but most except for those near the start and end of the sequence are very small. Therefore only the initial 101 and trailing 100 coefficients were retained and all others were set to zero. (For the slower heated sensor, a

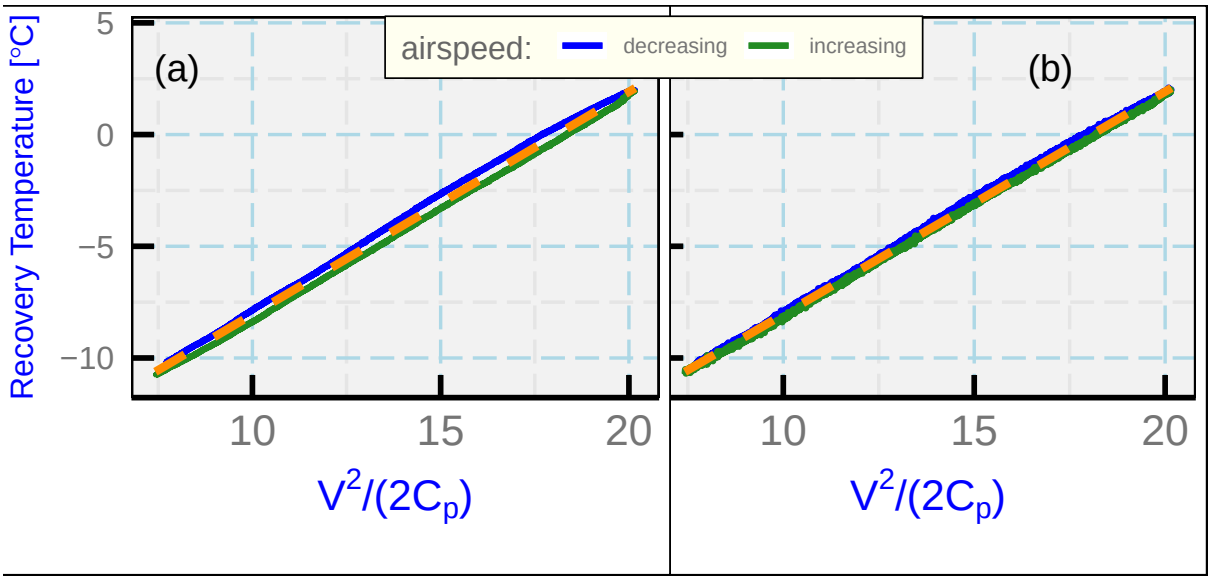

Figure 9: The recovery temperature as measured by a heated HARCO sensor during the speed-run maneuver shown in the previous figure (a) and the same measurement after correction (b). The abscissa is the dynamic-heating term without the recovery factor, where V is the airspeed and Cp is the specific heat at constant pressure. The measurements while the airspeed was increasing are shown by the green lines and those for decreasing airspeed by the blue lines. The dashed orange lines indicate the regression fits, with standard deviation about the fit of 0.26°C (uncorrected) and 0.10°C (after correction).

length of 301 samples was used.) Specifically, all values in the array representing the impulse function were set to zero for indices k with values $M+2 \leq k \leq N-M$ where N=15000 is the length of the calculated impulse function and $M = 100$ or 150 to leave 201 or 301 non-zero coefficients. This gave coefficients spanning about 8 s at 25 Hz for the unheated sensor and 12 s for the heated sensor, times long compared to the expected impulse response of the sensor. The sum of the coefficients obtained in this was was larger than 0.99, but to ensure proper normalization they were scaled to give a sum of unity.

3. Appropriate moving-average coefficients were then constructed by shifting the trailing negative-frequency components to the start of the array to give an eight-second or 12-s moving-average filter that represents the response of the sensor.

4. After filtering, the result then must be shifted backward in time to compensate for the 4-s or 6-s delays.

A sample of the approach used is listed below. For more details, see the "designFilter" code chunk.

```
NP <- 15000  ## Assume q 10-min segment
df <- 25 / NP  ## The frequency resolution
frq <- c(seq(0, 12.5, by=df), seq(-12.5+df, -df, by=df))
E <- LTphase(frq, P)  ## This function returns the gain and phase in deg.
G <- complex(modulus=E$Amp, argument=E$Phase * pi / 180) # the transfer fn
NG <- length(G)
GT <- fft(G, inverse=TRUE) / NG  # get the impulse function
## Limit to 200 coefficients (8 s at 25 Hz)
Lshift <- 100  ## below, will need to shift by 100 40-ms samples
GT[(Lshift + 2):(NP - Lshift)] <- complex(modulus=0)
GTT <- GT[GT != complex(modulus=0)]
## Reorder:
GTT <- c(GTT[(Lshift + 2):length(GTT)], GTT[1:(Lshift + 1)])
AR <- Arma(Re(GTT) / sum(Re(GTT)), 1) # Normalize to avoid <1% bias
## Result is appropriate coefficients to use for filtering;
## e.g., if DF is data.frame with Q measured dynamic heating:
DF$QF <- as.vector(signal::filter(AR, DF$Q)) ## standard R
## the following is a function to shift in time:
DF$QF <- ShiftInTime(DF$QF, .shift=-Lshift * 40, .rate=25)
```

Figure 10a shows the impulse function for the unheated and heated sensors, and Fig. 10b shows corresponding moving-average coefficients for a filter obtained from this impulse function. There is significant ringing in the filter for the unheated sensor because the shorter time constant for the sensor, 0.03 s, is smaller than the time between 25-Hz samples. The impulse response for the slower heated sensor leads to a much broader impulse response function.

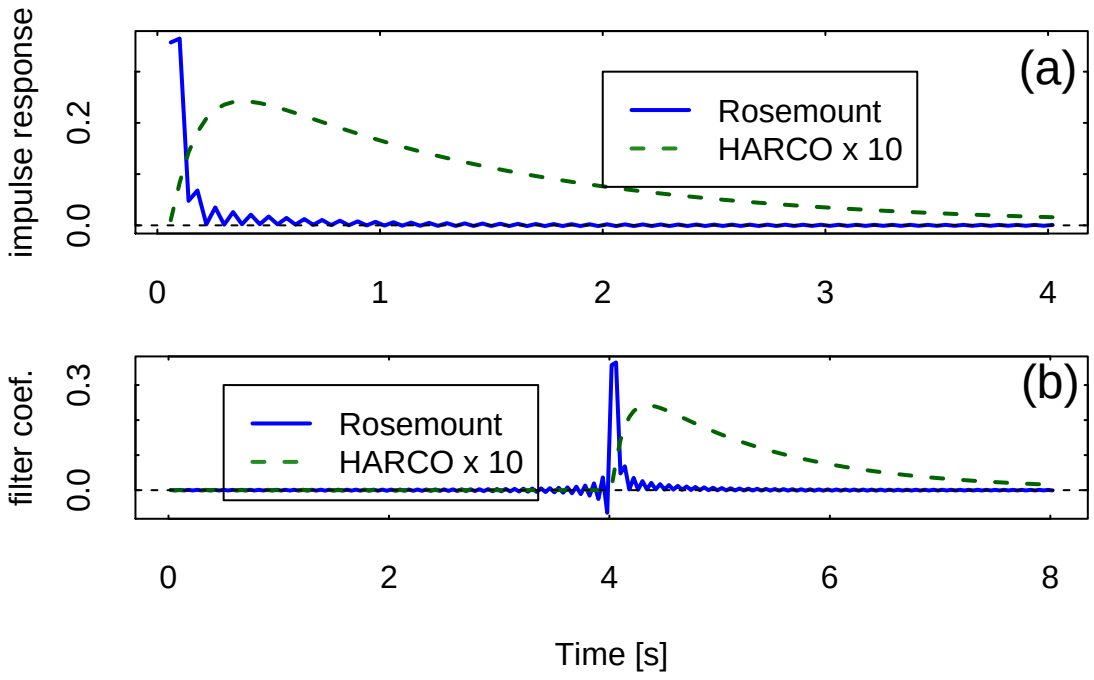

Figure 10: (a) The impulse response function found from the inverse Fourier transform of the transfer function for the unheated Rosemount 102E4AL sensor (Rosemount) and for the heated HARCO sensor (HARCO), using the response parameters from Table 3 of the paper. The impulse response for the HARCO sensor is multiplied by 10.
(b) A filter function (moving-average coefficients spanning 8 s) obtained from the impulse response function. The filtered result must be shifted forward in time by 4 s to compensate for the delay introduced by the filter. The coefficients are multiplied by 10 for the HARCO sensor.

The digital-filter coefficients and the transfer-function parameters on which they are based are stored in two data file, respectively "ARF.Rdata" and "PAR.Rdata". The following table documents the correspondence between the two files:

aircraft / sensor	ARMA coefficients	Data Rate	Parameter file	required time shift, samples
C-130 / unheated	AR	25 Hz	Param1	100
C-130 / heated	ARH	25 Hz	ParamSH	150
GV / unheated*	ARG	25 Hz	ParamSF	100
C-130 / unheated	AR1	1 Hz	Param1	10
C-130 / heated	ARH1	1 Hz	ParamSH	10
GV / unheated*	ARG1	1 Hz	ParamSF	10

*The listed values for GV / unheated are not used in the paper because it is thought that they were not determined as well as for the C-130 / unheated sensor. The C-130 / unheated values are used instead.

7.2 Other filtering methods

Two other calculations can produce the appropriately filtered response to dynamic heating $Q'(t)$:

1. Section 2.1 in the paper showed that the transfer function is represented reasonably by the solution to two coupled differential equations. Numerical integration of those equations can then produce the predicted response of the sensor to dynamic heating. The equations, revised to apply to dynamic heating, are these:

$$\frac{dQ_{qs}(t)}{dt} = \frac{Q(t) - Q_{qs}(t)}{\tau_2}$$

$$\frac{dQ'(t)}{dt} = \frac{\{aQ(t) = (1-a)Q_{qs}(t)\} - Q'(t)}{\tau_1}$$

   where the first equation describes the effect of dynamic heating on the support structure of the temperature sensor, leading to support-structure variations in temperature of $Q_{qs}(t)$ and the second describes the response of the sensing wire to the combined effects of this influence on the support temperature and the dynamic-heating term. This separation relies on the linearity of the underlying equations, which makes it possible to represent the effect of dynamic heating in isolation from real fluctuations in temperature. Euler integration of these differential equations led to erroneous results at high frequency arising from inadequate resolution in the integration, so a fourth-order Runge-Kutta integration with dynamic adjustment of the time step, as described by Cash and Karp [1990], was used.

2. The response specified by the frequency-domain transfer function $H(\nu)$ can be realized by Fourier transforms, by first calculating the Fourier transform of the dynamic-heating signal (here denoted $\hat{Q}(\nu) = \mathscr{F}^{-1}(Q(t))$ where $\mathscr{F}$ denote the Fourier transform) and then using the inverse Fourier transform $\mathscr{F}^{-1}$ to find the sensor response from $Q'(t) = \mathscr{F}^{-1}(H\{\nu)\hat{Q}(\nu))$.

As noted in the paper, for 25 Hz measurements from the unheated sensor the numerical integration appeared to provide less satisfactory results than the other two methods, even with the step-adjusted numerical integration technique. The first time constant for that probe is comparable to the time between samples, apparently leading to accumulation of errors during the integration. To see if this was indeed the problem, the 25-Hz measurements were interpolated to 125-Hz samples, smoothed with a low-pass Butterworth filter with 25-Hz cutoff, and then integrated with 125-Hz resolution. The result was in much better agreement with the results from the other two methods, so this is consistent with attribution of the discrepancy to problems with the 25-Hz integration. (This was done in special separate steps, not incorporated into the RMarkdown file, so the code does not reflect this check.)

A special processing script, "CorrectTemperature.R", is included in the project archive. It will process an existing netCDF data file to incorprate this filtering, and it adds a new measurement of ambient temperature to the archive which corrects for the sensor response to dynamic heating as described in the paper.

8 Comments on Section 4 (Flux of Sensible Heat)

Figure 10 was generated using some special code developed for this purpose, function "plotCS()" defined in the "initial3" code chunk. A related plot function is available from "Ranadu::flux()" but in order to combine the two plots from SOCRATES and CSET into one plot "plotCS()" was used to place the appropriate plots into viewports that were passed to the function. There is an inactive remnant call to "flux()" in the "SOCp1" code chunk that was used originally but has been replaced by calls to plotCS(). The earlier calls to "Ranadu:CohSpec)" were also replaced by direct calls to the "fft()" function in the same way as was done in "CohSpec()". The smoothing of the cospectrum discussed in the paper was accomplished using the "zoo::rollapply()" function to calculate the running means with various lengths.

Following the introductory discussion of the cospectrum, there was a now-suppressed section that plotted and discussed the quadrature spectrum. This section is included in the "Extra" branch in the .lyx file so it is accessible to anyone interested, but the material and discussion did not seem be contribute to the main theme of this section in any significant way so it has been omitted from the standard branch. It is worth noting, however, that the quadrature spectrum showed more indication of significant values than might have been expected, and some aspects of that distribution were not understood or explained convincingly in the suppressed material. An alternative to the correction procedure used in this section would be to transform between the cospectrum and quadrature spectrum according to the transfer-function phase angle, and that was explored but was abandoned when it did not appear to produce results consistent with the approach taken here.

The following are comments on the simulation at the end of this section. The generation of a variance spectrum with $-5/3$ slope (spectral variance vs frequency) started by using work from an earlier study, documented here. In the course of working on this paper, it became evident that there is a better way: generate a Gaussian noise spectrum (flat vs. frequency) and then modify

the spectral density to have $-5/3$ slope before transforming back to the simulated measurement sequence. This was the approach used in the paper. It is much faster than the original method used in the referenced document above. However, there appeared to be some advantages to the original method in terms of accuracy and consistency of results for different random sequences. Perhaps those advantages arose from generating originally at 50 Hz and then reducing the sample rate, or they may have arisen from generation as complete sine waves spanning the entire time period (which is unrealistic). In the code chunk "generate" in the .Rmd file, some truncation is included for frequency components smaller than 0.02 Hz, as a commented option. This seemed to improve the consistency of the results at the highest frequency and so was included in the latest simulation used in the paper. The code in "generate" may be useful elsewhere. The resulting time series are saved in "DF.Rdata" for that reason and to avoid needing to re-generate the series with each processing run.

9 Comments on Appendix A: Correcting the Temperature

9.1 Integrating the equations

When this was written it was developed from the differential equations, and it was only later realized that the solution is the same as that obtained by Inverarity by a quite different method. That is the primary reason that this was relegated to an appendix. The integration of Eqns. (4) and (6) in the paper is straightforward, but the reduction to a single integration as in Eqn. (A1) in the paper replaces one of the integrations with a more straightforward evaluation using a fourth-order central-difference representation for the derivative $dT_m(t)/dt$. The formula used for the derivative is:

```
DTMDT <-  (c(0, 8*diff(TTRR, 2), 0) -
              c(0, 0, diff(TTRR, 4), 0, 0)) * Rate / 12
## The second-order expression would be:
## DTMDT <- c(0, diff(TTRR, 2), 0) * Rate / 2
```

Other possible representations of the derivative, including the second-order centered finite-difference expression above and Savitzky-Golay calculation of the first derivative, were also tried but the above choice worked best.

The use of the new routine "rk4.integrate()" was important in the case of 25-Hz measurements from the unheated sensor because the standard Runge-Kutta package for R (and an Euler integration as used by Inverarity) led to obvious numerical problems associated with the small time constant τ_1 for the unheated sensor.

For the heated HARCO sensor, another finite-difference expression is needed for the second derivative. Because this sensor is slow, a second-order centered expression was adequate, so this expression was used for $d^2T_m(t)/dt^2$:

```
DTM2DT2 <- (c(diff(RTH1), 0) - c(0, diff(RTH1))) * Rate^2
```

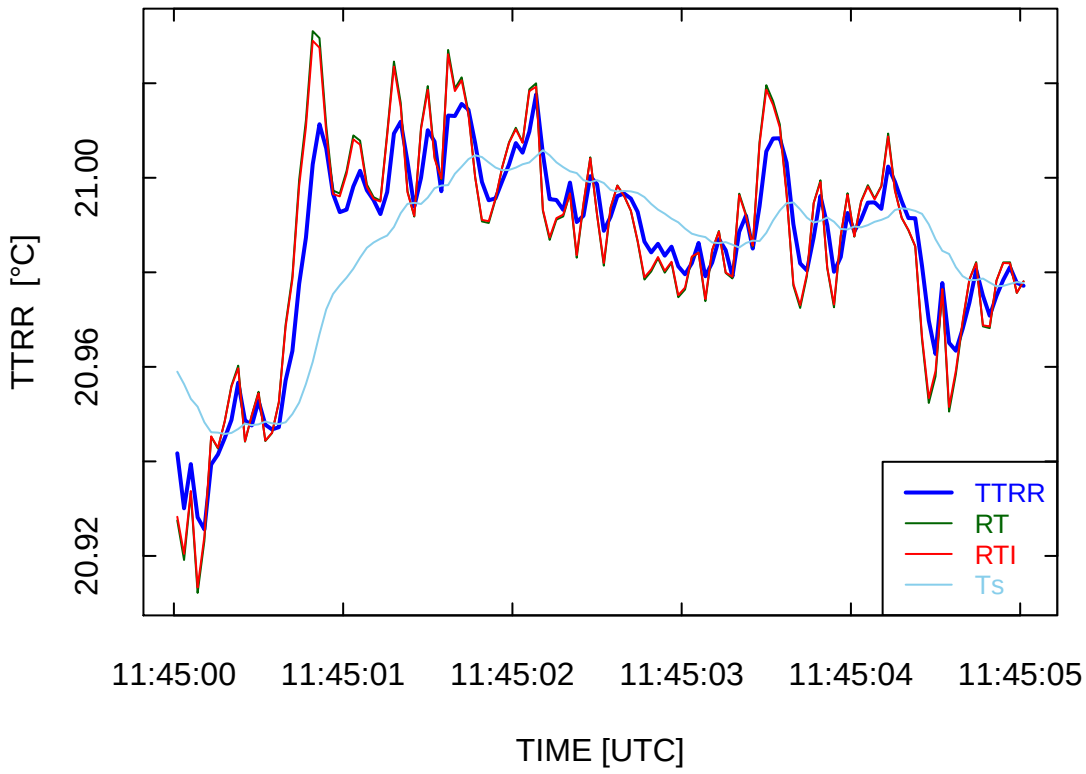

Figure 11: Comparison between the solution developed in the paper (green line, RT) and that proposed by Inverarity (red line, RT1). The original measurement is shown as TTRR (blue line), and the calculated temperature of the support is shown as Ts (cyan line).

9.2 Comparison to the Inverarity solution

Inverarity [2000] developed an equation equivalent to the following expression for the solution of the differential equations:

$$T_a(t_k) = T_a(t_0) + \frac{1}{2\alpha\delta t}\frac{dT_m(t_k)}{dt} - \beta\Delta T_m(t_k) + \gamma(1-\beta)I_k$$

where $\alpha = (A_1/\tau_1 + A_2/\tau_2), \beta = \frac{1}{\alpha}(1/\tau_1 + 1/\tau_2) - \frac{1}{\alpha^2\tau_1\tau_2}$, $\gamma = \frac{1}{\alpha\tau_1\tau_2}$ and

$$I_k = e^{-\gamma\delta t} I_{k-1} + \frac{\delta t}{2}\left(e^{-\gamma\delta t}(T_m(t_{k-1}) - T_a(t_0)) + T_m(t_k) - T_a(t_0)\right)$$

with $I_0 = 0$.

Figure 11 shows the equivalence between this solution and the one proposed in this paper, for corresponding response functions.

9.3 Fourier transformation

There are some standard considerations when dealing with Fourier transforms that are perhaps worth discussing here. The standard representation of the Fourier transform has frequencies ordered from 0 to the Nyquist frequency in steps given by $\delta f = \text{Rate}/\text{N}$ where Rate is the sample rate (e.g., 25 Hz) and N is the length of the time series. These are followed by negative frequencies ranging from $(-\nu_{Nq} + \delta f)$ to $-\delta f$, for a total length of N. When the transfer function is specified, it needs to be specified for these same frequencies. This is done conveniently by the "LTphase()" function if the first argument to the function is frequencies as specified. Then normal R operations can be used with the resulting complex vectors. The basic correction procedures is then concisely specified as

```
RTcorr <- Re(fft(fft(RTm) / H, inverse = TRUE)) / N
```

where RTm is the original measurement, RTcorr is the corrected measurement, H is the transfer function, and N is the length of the time series.

This will fail if there are missing-value measurements in the time sequence, so interpolation is used routinely via the Ranadu routine "SmoothInterp()", but with argument .Length=0 to suppress smoothing.

Because of the cyclic nature of the Fourier transform, spurious effects often arise at the start and end of the time series unless modifications are introduced. In the cases presented in this paper, means and trends are removed and then the time series is padded at the start and end with zeroes. The results are then used only for a subset of the original valid time sequence.

References

JJ Allaire, Yihui Xie, R Foundation, Hadley Wickham, Journal of Statistical Software, Ramnath Vaidyanathan, Association for Computing Machinery, Carl Boettiger, Elsevier, Karl Broman, Kirill Mueller, Bastiaan Quast, Randall Pruim, Ben Marwick, Charlotte Wickham, Oliver Keyes, Miao Yu, Daniel Emaasit, Thierry Onkelinx, Alessandro Gasparini, Marc-Andre Desautels, Dominik Leutnant, MDPI, Taylor and Francis, Oğuzhan Öğreden, Dalton Hance, Daniel Nüst, Petter Uvesten, Elio Campitelli, John Muschelli, Zhian N. Kamvar, Noam Ross, Robrecht Cannoodt, Duncan Luguern, and David M. Kaplan. *rticles: Article Formats for R Markdown*, 2020. URL `https://CRAN.R-project.org/package=rticles`. R package version 0.14.

Ben Baumer, Nicholas Horton, and Daniel Kaplan. *mdsr: Complement to 'Modern Data Science with R'*, 2019. URL `https://CRAN.R-project.org/package=mdsr`. R package version 0.1.7.

J. R. Cash and A. H. Karp. A variable order runge-kutta method for initial value problems with rapidly varying right-hand sides. *ACM Transactions on Mathematical Software (TOMS)*, 16(3): 201–222, 1990.

DC Collis and MJ Williams. Two-dimensional convection from heated wires at low reynolds numbers. *Journal of Fluid Mechanics*, 6(3):357–384, 1959.

W. A. Cooper. Ranadu: R-based analysis for ncar aircraft data users (version 2.6_20.05.15)., May 2020. URL `http://doi.org/10.5281/zenodo.3830427`.

W. A. Cooper, R. B. Friesen, M. Hayman, J. B. Jensen, D. H. Lenschow, P. A. Romashkin, A. J. Schanot, S. M. Spuler, J. L. Stith, and C. Wolff. Characterization of uncertainty in measurements of wind from the NSF/NCAR Gulfstream V research aircraft. NCAR technical note NCAR/TN-528+STR, Earth Observing Laboratory, NCAR, Boulder, CO, USA, jul 2016. URL `http://n2t.net/ark:/85065/d7qr4zqr`.

G. W. Inverarity. Correcting airborne temperature data for lags introduced by instruments with two-time-constant responses. *Journal of Atmospheric and Oceanic Technology*, 17(2): 176–184, 2000. doi: 10.1175/1520-0426(2000)017<0176:CATDFL>2.0.CO;2. URL `https://doi.org/10.1175/1520-0426(2000)017<0176:CATDFL>2.0.CO;2`.

W. A. Khan, J. R. Culham, and M. M. Yovanovich. Fluid Flow Around and Heat Transfer From an Infinite Circular Cylinder. *Journal of Heat Transfer*, 127(7):785–790, 10 2004. ISSN 0022-1481. doi: 10.1115/1.1924629. URL `https://doi.org/10.1115/1.1924629`.

E. Kreyszig. *Advanced Engineering Mathematics*. Wiley, 1962. ISBN 047133328X. URL `https://books.google.com/books?id=eAK7QgAACAAJ`.

D. H. Lenschow and P. Spyers-Duran. Measurement techniques: air motion sensing. Technical report, National Center for Atmospheric Research, 1989. URL `https://www.eol.ucar.edu/raf/Bulletins/bulletin23.html`.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C (2nd Ed.): The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 1992. ISBN 0-521-43108-5.

R Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019. URL `http://www.R-project.org`.

RStudio. *RStudio: Integrated development environment for R (Version 0.98.879)*, 2009. URL `http://www.rstudio.org`.

Truman M Stickney, Marvin W Shedlov, and Donald I Thompson. Goodrich total temperature sensors. *Goodrich Corporation*, 1994.

B. Swihart and J. Lindsey. *rmutil: Utilities for Nonlinear Regression and Repeated Measurements Models*, 2019. URL `https://CRAN.R-project.org/package=rmutil`. R package version 1.1.3.

H. Wickham. *ggplot2: elegant graphics for data analysis*. Springer New York, 2009. ISBN 978-0-387-98140-6. URL `http://had.co.nz/ggplot2/book`.

Y. Xie. *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2013. URL `http://yihui.name/knitr/`. ISBN 978-1482203530.

Y. Xie. *knitr: A general-purpose package for dynamic report generation in R*, 2014. URL `http://yihui.name/knitr/`. R package version 1.6.

Yihui Xie, J.J. Allaire, and Garrett Grolemund. *R Markdown: The Definitive Guide*. Chapman and Hall/CRC, Boca Raton, Florida, 2018. URL `https://bookdown.org/yihui/rmarkdown`. ISBN 9781138359338.

– End of Memo –