The NCAR *In Situ* Turbulence Detection Algorithm

Gregory Meymaris
Robert Sharman
Larry Cornman
Wiebke Deierling
The Technical Notes series provides an outlet for a variety of NCAR Manuscripts that contribute in specialized ways to the body of scientific knowledge but that are not yet at a point of a formal journal, monograph or book publication. Reports in this series are issued by the NCAR scientific divisions, serviced by OpenSky and operated through the NCAR Library. Designation symbols for the series include:

**EDD – Engineering, Design, or Development Reports**  
Equipment descriptions, test results, instrumentation, and operating and maintenance manuals.

**IA – Instructional Aids**  
Instruction manuals, bibliographies, film supplements, and other research or instructional aids.

**PPR – Program Progress Reports**  
Field program reports, interim and working reports, survey reports, and plans for experiments.

**PROC – Proceedings**  
Documentation or symposia, colloquia, conferences, workshops, and lectures. (Distribution maybe limited to attendees).

**STR – Scientific and Technical Reports**  
Data compilations, theoretical and numerical investigations, and experimental results.

The National Center for Atmospheric Research (NCAR) is operated by the nonprofit University Corporation for Atmospheric Research (UCAR) under the sponsorship of the National Science Foundation. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

National Center for Atmospheric Research  
P. O. Box 3000  
Boulder, Colorado 80307-3000
The NCAR In Situ Turbulence Detection Algorithm

Gregory Meymaris  
Research Applications Laboratory,  
National Center for Atmospheric Research, Boulder, CO

Robert Sharman  
Research Applications Laboratory,  
National Center for Atmospheric Research, Boulder, CO

Larry Cornman  
Research Applications Laboratory,  
National Center for Atmospheric Research, Boulder, CO

Wiebke Deierling  
Research Applications Laboratory,  
National Center for Atmospheric Research, Boulder, CO
Table of Contents

1 Acknowledgments ........................................................................................................... 3

2 Introduction ...................................................................................................................... 3

3 EDR Algorithm Overview ................................................................................................. 6
  3.1 Requirements .................................................................................................................. 9
  3.2 Quality Control Algorithm ......................................................................................... 13
  3.3 Vertical Wind Calculation ............................................................................................ 18
  3.4 EDR Estimation ............................................................................................................ 19
  3.5 Mean and Peak EDR Estimation ................................................................................... 25
  3.6 EDR Algorithm Steps ................................................................................................... 26
  3.7 Performance .................................................................................................................. 27

4 Event Based Triggering Logic ............................................................................................ 35
  4.1 Turbulence Event Reports .......................................................................................... 35
  4.2 Routine Reports ........................................................................................................... 37

5 Deployment ...................................................................................................................... 39

6 Conclusions ...................................................................................................................... 40

7 References ......................................................................................................................... 41

8 Appendix: Detailed Deployment Steps .............................................................................. 43
  8.1 First steps .................................................................................................................... 43
The NCAR In Situ Turbulence Detection Algorithm

8.2 Deployment Target .................................................................................................................. 43
8.3 Boeing ACMS Solution ........................................................................................................... 46
8.4 Existing Teledyne ACMS Solution for an existing aircraft type ................................. 46
8.5 New Teledyne ACMS Implementation for an existing aircraft type ......................... 46
8.6 New Teledyne ACMS Implementation for a new aircraft type ................................. 47
8.7 New ACMS Solution on Teledyne ACMS for an existing aircraft type ............... 47
8.8 New ACMS Solution on Teledyne ACMS for a new aircraft type ....................... 48
8.9 New ACMS with a new manufacturer on an existing aircraft type ...................... 48
8.10 New ACMS with a new manufacturer on a new aircraft type ............................ 49
8.11 Non-ACMS Solutions ....................................................................................................... 49
Acknowledgments

This research is in response to requirements and funding by the Federal Aviation Administration (FAA). The views expressed are those of the authors and do not necessarily represent the official policy or position of the FAA. Thanks are given to Jason Craig, Julia Pearson, Gary Cunning, Bill Myers, Gary Blackburn, Steve Carson, Martha Limber, Cory Morse, and Andrew Weekly for their contributions to this project over the years. The authors would also like to extend our sincere gratitude to our sponsors at the FAA, who made this work possible.

Introduction

Historically, there has been one main source for routine turbulence observations for the aviation community, namely pilot reports (PIREPS/AIREPS). Although useful for tactical turbulence avoidance, there have been problems using PIREPS for turbulence forecasting and verification including (Schwartz 1996, Sharman et al. 2006, Sharman et al. 2014):

- reported time/position is often significantly displaced from when and where the turbulence actually occurred
- the turbulence intensity reported is subjective and aircraft-dependent
- very few reports of smooth conditions (NULL reports) are made
- most reports are only made at the pilot’s discretion

These deficiencies make PIREPs inadequate for providing maps of atmospheric (i.e., aircraft independent) turbulence levels along aircraft flight routes. To address these difficulties, NCAR, under sponsorship from the FAA Aviation Weather Research Program (AWRP), has developed an in situ turbulence reporting system (Cornman et al. 1995, 2004, Sharman et al. 2014) which has now been implemented on several major U.S. commercial air carriers composed of numerous
The NCAR *In Situ* Turbulence Detection Algorithm

aircraft types. These reports are routine, completely automated, are aircraft independent, with high temporal and spatial accuracy, and are therefore ideal for operational purposes (Farrar et al. 2013) and turbulence case studies (e.g., Lane et al. 2012, Sharman et al. 2012).

The NCAR *In Situ* Turbulence Detection Algorithm (referred to as the *In Situ* EDR Algorithm, in this document) is coded as a C library which is typically integrated into an onboard Aircraft Conditioning and Monitoring System (ACMS) or other suitable onboard computer. The complete software package (available upon request) also includes Matlab® software used to check data quality, tune, and validate the onboard software. The *In Situ* EDR Algorithm estimates the cube root of the energy (or eddy) dissipation rate, referred to as EDR, an atmospheric turbulence intensity metric. Once on board the aircraft, the software runs continuously when the aircraft is off the ground, producing mean and peak EDR over the previous minute of flight once every minute.

While there are a few adjustable parameters that are weakly aircraft-dependent, the EDR output itself is developed to be aircraft-independent. This is important since an aircraft-dependent turbulence metric would need to be accompanied by aircraft data, such as airspeed, altitude, mass, and type in order for others to be able to interpret it. The impact of the turbulence on a vehicle, an aircraft-dependent measure, can always be estimated from the state of the atmosphere, given knowledge about the vehicle and its operating condition (e.g., airspeed, weight, altitude). However, without also transmitting all of that information from the reporting vehicle, it is more difficult to estimate the state of the atmosphere. For this reason, EDR has been the ICAO standard for aircraft reporting since 2001 (ICAO 2010). EDR also has the advantage of being consistent with other observations (e.g. radar-based detection algorithms as described by Williams and Meymaris (2016)) and is the preferred metric for forecasting and nowcasting turbulence.
An important component of the EDR algorithm is the inclusion of quality control (QC) algorithms, which are implemented on board as part of the software package. The software package also includes an EDR Support component that assists in the checking of data quality, tuning, and verification of the onboard software.

While for the original accelerometer-based version (Cornman et al. 1995), all one minute mean and peak EDR estimates were downlinked, the newer vertical wind-based version, running currently on about 1,400 aircraft (as of June 2019), has the capability to report turbulence events plus occasional (e.g. 15-30 minute) routine reports (Sharman et al. 2014). This was done to save communication costs associated with downlinking the mostly smooth reports. Figure 1 shows the global distribution of in situ EDR data received from Delta, Southwest, and United Airlines during the first part of 2019. The In Situ EDR Algorithm described in this document pertains only to the vertical wind-based algorithm (described in Sharman et al. 2014) and not the older accelerometer
The NCAR In Situ Turbulence Detection Algorithm

based one (described in Cornman et al. 1995). Besides documenting the algorithm, this document also describes the integrated quality control algorithm, the event triggering logic, the algorithm performance evaluated using simulations, and steps needed for deployment.

3 EDR Algorithm Overview

To clarify semantics and notation, the energy or eddy dissipation rate, typically denoted by \( \varepsilon \), is the rate at which turbulent kinetic energy is dissipated into heat (units m\(^2\)s\(^{-3}\)). There is some confusion because, in the aviation industry, references to EDR are usually in regards to the cube root of \( \varepsilon \). So from here on, the acronym EDR (units m\(^{2/3}\)s\(^{-1}\)) is used to indicate the cube root of eddy dissipation rate (\( \varepsilon \)). At first, this may seem like an unnecessary source of confusion, but there is a natural reason for reporting \( \varepsilon^{1/3} \) rather than \( \varepsilon \). \( \varepsilon^{1/3} \) scales with the standard deviation of the winds (Sharman et al. 2014), thus making interpretation simpler.

An important aspect of EDR is that since it is an atmospheric turbulence intensity metric, it is obviously aircraft independent. Two different aircraft will respond to the same atmospheric turbulence differently depending on aircraft type, altitude, airspeed, weight, and flight conditions (maneuvers, etc.). If one used an aircraft dependent metric to disseminate turbulence information, then one would have to know all of this extra information to be able to interpret a turbulence report. If instead one uses an aircraft independent metric, then all one has to know is how to interpret EDR turbulence intensity levels relative to their own application. It should be noted that aircraft dependent turbulence metrics can be useful as well, for example, to describe what conditions the aircraft and its passengers experienced. However, EDR is generally preferred for use in the development and verification of turbulence forecasting, nowcasting, and remote sensing products,
The NCAR In Situ Turbulence Detection Algorithm

and in some cases real-time use (Sharman and Pearson 2017, Pearson and Sharman 2017, Williams and Meymaris 2016).

Additionally, EDR is a metric that does not depend, to a large extent, on the scales (or eddy sizes) used to estimate it if one avoids considering larger eddy sizes, say above 500m or so (below is a further discussion of this), or quite small eddy sizes (sub-centimeter scales) where viscosity becomes a factor. If one algorithm uses scales between 100m and 200m to estimate turbulence intensity and another uses scales between 50m and 100m, they will obtain, on average, the same EDR value for the same turbulence event. The scales for which EDR is a good atmospheric turbulence intensity metric, also happen to be the scales which affect aircraft negatively. Very large or very small scale eddies in comparison to the aircraft size are not hazardous or even uncomfortable.

The In Situ EDR Algorithm is a vertical winds-based maximum-likelihood algorithm which can be conceptually thought of as having three basic components: the vertical wind calculation, the maximum likelihood EDR estimation, and, integrated into the previous two, a quality control algorithm. The term maximum likelihood refers to a statistical method for estimating an unknown quantity, and is discussed further below. The algorithm takes as inputs (see Table 1) the left and right angle of attack ($\alpha_1$ and $\alpha_2$; deg.), true airspeed ($V$; m s$^{-1}$), roll ($\phi$; deg.), pitch ($\theta$; deg.), and inertial vertical velocity ($\dot{Z}$; ms$^{-1}$; positive up) and outputs, once per minute, the mean and peak EDR ($\hat{\varepsilon}$; m$^{2/3}$ s$^{-1}$; the “hat” over the $\varepsilon$ denotes that the value is an estimate), along with associated quality control metrics. Pressure altitude, $Z$ (m), and vertical acceleration, $\ddot{Z}$ (g), are not used in the algorithm, but are used in the tuning.
## The NCAR In Situ Turbulence Detection Algorithm

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Required for Algorithm</th>
<th>Units</th>
<th>Minimum Sampling Frequency (Hz)</th>
<th>Minimum precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTC date and time</td>
<td>NA</td>
<td>No</td>
<td>NA</td>
<td>8</td>
<td>0.125 s</td>
</tr>
<tr>
<td>Aircraft type</td>
<td>NA</td>
<td>Yes¹</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>True Airspeed</td>
<td>( V_T )</td>
<td>Yes</td>
<td>( \text{m s}^{-1} )</td>
<td>8</td>
<td>0.036 ( \text{m s}^{-1} )</td>
</tr>
<tr>
<td>Left vane angle of attack (+ nose up)</td>
<td>( \alpha_1 )</td>
<td>Yes</td>
<td>( \text{deg} )</td>
<td>8</td>
<td>0.05 deg</td>
</tr>
<tr>
<td>Right vane angle of attack (+ nose up)</td>
<td>( \alpha_2 )</td>
<td>Yes</td>
<td>( \text{deg} )</td>
<td>8</td>
<td>0.05 deg</td>
</tr>
<tr>
<td>Pitch (+ nose up)</td>
<td>( \theta )</td>
<td>Yes</td>
<td>( \text{deg} )</td>
<td>8</td>
<td>0.02 deg</td>
</tr>
<tr>
<td>Body pitch rate</td>
<td>( \dot{\theta} )</td>
<td>No</td>
<td>( \text{deg s}^{-1} )</td>
<td>8</td>
<td>0.02 ( \text{deg s}^{-1} )</td>
</tr>
<tr>
<td>Roll (+ right wing down)</td>
<td>( \phi )</td>
<td>Yes</td>
<td>( \text{deg} )</td>
<td>8</td>
<td>0.02 deg</td>
</tr>
<tr>
<td>Inertial Vertical Velocity (+ upward)</td>
<td>IVV</td>
<td>Yes</td>
<td>( \text{m s}^{-1} )</td>
<td>8</td>
<td>0.00508 ( \text{m s}^{-1} )</td>
</tr>
<tr>
<td>Barometric Altitude</td>
<td>( Z )</td>
<td>No</td>
<td>( \text{feet MSL} )</td>
<td>8</td>
<td>1 ft</td>
</tr>
<tr>
<td>Vertical Acceleration</td>
<td>( \ddot{z} )</td>
<td>No</td>
<td>( \text{g} )</td>
<td>8</td>
<td>0.003 g</td>
</tr>
<tr>
<td>Latitude</td>
<td></td>
<td>No</td>
<td>( \text{deg} )</td>
<td>1</td>
<td>0.0002 deg</td>
</tr>
<tr>
<td>Longitude</td>
<td></td>
<td>No</td>
<td>( \text{deg} )</td>
<td>1</td>
<td>0.0002 deg</td>
</tr>
<tr>
<td>Wind Speed</td>
<td></td>
<td>No</td>
<td>( \text{knots} )</td>
<td>1</td>
<td>1 knot</td>
</tr>
<tr>
<td>Wind Direction</td>
<td></td>
<td>No</td>
<td>( \text{deg} )</td>
<td>1</td>
<td>1 deg</td>
</tr>
</tbody>
</table>

Table 1: List of required parameters and update frequencies for vertical winds-based algorithm including running real-time, tuning, and verification.

¹ Via adaptable parameter initialization constant values.
3.1 Requirements

There are a number of requirements for the *In Situ* EDR Algorithm as well as for the software to work properly. Regarding the software, what is required is a computer system that is capable of having a C code library integrated in and that has access to the required avionics data with the required data rates (Table 1), and ideally an accessible mechanism to transmit the data to the ground. It is equally feasible to use the software to post-process recorded data, though integration is still necessary.

The requirements for the algorithm to produce well-calibrated EDR data are more stringent. Special attention must be taken with respect to the data rates. Ideally, all input fields should be sampled at a common uniform sampling rate, $f_s$, of at least 8 Hz (8 times per second) with a dealiasing filter applied\(^2\). Alternatively, having one or more of the fields available at a data rate less than $f_s$, can have minor to extreme deleterious effects on the spectral shape of the winds, depending on the field(s). Practically speaking, the critical fields appear to be the angle of attacks, $\alpha_1$ and $\alpha_2$, as they contribute the most to the spectral content of the wind spectrum, though a rigorous analysis of this needs to be performed. Recommended *minimum* data sampling rates are provided in Table 1.

In the discussion that follows, it is important to understand the difference between the sampling frequency of the data, and a signal frequency contained in that data. The sampling frequency refers to the rate at which data is available. An 8 Hz sampling frequency indicates that data is available every 1/8\(^{th}\) of a second. A 2 Hz signal (e.g. a sine wave that fully repeats 2 times

\(^2\) Standard practice for sampling an analog device is to sample at $10f_s$, apply a dealiasing (low-pass) filter with stop-band cutoff around $0.4f_s$, and then downsample to $f_s$. This is to mitigate sampling artifacts.
per second), can be sampled using the 8 Hz sampling frequency. In fact, within data that is sampled at 8 Hz, one can resolve any number of frequencies. For real-valued data (rather than complex), one can resolve frequencies between 0 and half the sampling frequency, or 4 Hz, in this case. In the context of this technical note, the frequencies correspond to eddy scales (sizes). An aircraft sampling at 8 Hz, travelling at 250 m/s are able to observe eddies at certain scales (and their corresponding frequencies) and not at others.

A minimum common sampling rate of 8 Hz is important for a number of reasons, all quite technical. We will discuss two here. The first comes from the fact that turbulence over certain eddy scales has been seen to follow a so-called *inertial range behavior* where the spectrum falls off with frequency or wavenumber according to a -5/3 power law in log-log space (see e.g. Sharman 2016). An example of this is shown in Figure 2, which contains two sets of power spectra (the left is weaker, and the right stronger) from vertical wind data, along with a reference -5/3 slope line. The x-axis is the frequency, and the y-axis is the spectral power. The left vertical dotted line is at 0.5 Hz. For the vertical component of the wind velocity, this power law holds very well.

Figure 2: Example output from the winds-based EDR algorithm for two different turbulence levels [peak=0.16 m$^{2/3}$ s$^{-1}$ panel (a) and peak=0.35 m$^{2/3}$ s$^{-1}$panel (b)]. For both cases the individual 12 spectra developed over a one-minute time interval are shown as well as a reference $f^{-5/3}$ line and the algorithm cutoff frequencies of 0.5 Hz and 3.5 Hz. Figure originally from Sharman et al. 2014, copyright American Meteorological Society.
Certainly for scales between a meter\(^3\) and 500 meters\(^4\), but not necessarily outside of that interval. For an aircraft flying at 250 m/s, that works out to a frequency range of between 0.5 Hz and 250 Hz. Thus, using data below about 0.5 Hz may be prone to error because the algorithm relies on the turbulence following the -5/3 power rule. Additionally, the algorithm applies a high-pass filter (a linear de-trend filter) which also contaminates the lower frequencies. If the data is sampled at 8 Hz, then one only gets usable information up to a maximum of half that amount (the Nyquist frequency), i.e. 4 Hz. Often, there is contamination near that high end (from noise and from sampling artifacts), and thus we have found it prudent to exclude the upper 0.5 Hz, giving us an effective frequency range of 0.5 to 3.5 Hz (the right vertical dotted line in Figure 2). The spectra in Figure 2 can be seen to well follow the -5/3 power law quite well. It is important to note that the data has a lot of variation, which is due to the limited sampling of the turbulence, which has a random component. Also, note that for slower moving aircraft, one can set the lower bound to a smaller number in proportion to the decrease in airspeed except for the fact that the algorithm uses the high-pass filter.

If one attempts to use data sampled at a lower sampling frequency than 8 Hz, then one very quickly begins to have very little data in the desired frequency range. A sampling rate of 4 Hz would give an effective frequency range of 0.5 to 1.5 Hz. Based on simulations, this results in about a 50% increase in the standard deviation of the estimate for a single EDR (from 10 seconds of data). The 95% confidence interval goes from +/- 20% for a single EDR to +/- 30%. In other words, the errors start to become quite large. At a sampling frequency of 1 Hz, there is simply no longer an acceptable sampling frequency range, and the errors become very large (-96% to +183%.

\(^3\) Actually, more on the order of centimeters.
\(^4\) Examination of 8 Hz data recorded from commercial aircraft do seem to indicate that the -5/3 power law holds up to even larger scales than 500 m, though it does seem to depend on the event, and the results are preliminary.
for the 95% confidence interval; i.e. any number between 0 to three times the value of the true EDR), though actual results do depend on algorithm specifics (e.g. the frequency cutoffs).

The next reason for a minimum requirement of an 8 Hz sampling frequency builds on the previous issue. The current EDR algorithm calculates 12 EDRs per minute and reports the maximum as the peak EDR (this is discussed further below). If the standard deviation of each single EDR were larger, then the expected peak EDR value also goes up for homogenous turbulence, but oddly, not so for a short isolated turbulence event. In other words, there would be a comparability problem between the peak EDR from a 4 Hz algorithm versus an 8 Hz algorithm. For example, from turbulence simulations (based on a method described by Frehlich et al., 2001), for an 8 Hz algorithm in homogenous turbulence, we have found that the peak EDR would be expected to be about 15% larger than the mean EDR. For a 4 Hz version of the algorithm, the peak would be expected to be 25% larger than the mean EDR. In essence, the peak EDRs would be “biased” roughly 10% high relative to an 8 Hz algorithm, in homogenous turbulence. For short isolated events, there would not be a bias, just a larger standard deviation. This is because for a short isolated event (less than 10 seconds), the peak EDR statistics are dominated by the statistics of the one or two EDRs that encompassed the event. The individual EDRs are unbiased, and so the peak EDR in this case would also be unbiased.

At this point, sampling frequencies between 4 and 8 Hz would, in some cases, be marginally acceptable, with higher sampling frequencies being better. However, it is recommended that sampling frequencies below 4 Hz should not be considered for use with the current algorithm.

A further requirement is that the data quality of the algorithm inputs be “good”. For example, there should be relatively few bad or missing data and there should be little to no
contamination, including from filtering, noise, vibrations, etc. These problems are generally examined and identified during the tuning phase of implementation.

3.2 Quality Control Algorithm

The use of quality control algorithms within the onboard software is crucial since minor spikes or steps in the data can result in substantially erroneous EDR values, and, in our experience, these spikes are more frequent (often several per flight) than actual “real” elevated turbulence events. In other words, failure to mitigate these could result in a high (turbulence) false alarm rate.

The overarching idea for the quality control algorithm is that (generally) each value of each data field used within the algorithm is assigned a confidence value from 0 to 1 (0 representing low and 1 representing high confidence) and a QC flag (0/1 indicating no problem/problem found). Initially, confidences are set to 1 and QC flags are set to 0. If a problem is identified with a value, then its confidence is lowered and its QC flag is set (to 1). The confidence indicates the current estimation of a data value, whereas the QC flag indicates whether this data value has been determined to be “bad” previously. For example, if a data value, which was determined to be bad, is “fixed”, then its confidence may be set to 1, but its QC flag will remain set at 1.

There are several processes by which problems are identified. First, the data can be marked as “bad” on input to the software (e.g. the flag from the databus may be set indicating that the data value, as indicated on the bus, is bad). Second, the algorithm inputs and the (derived) vertical winds ($w$) are bounds-checked. Third, derived quantities such as $w$ have an inherited confidence from their input fields. Fourth, $V_T$ and $w$ are processed through the median QC algorithm described below. If the data is found to be suspicious using the first two methods, the confidences are set to 0 and the QC flag is set to 1. The third and fourth QC processes require further explanation.
The NCAR In Situ Turbulence Detection Algorithm

There are several derived high-rate quantities in the algorithm: the averaged angle of attack $(\alpha_a = (\alpha_1 + \alpha_2) / 2)$, the body-axis angle of attack $(\alpha_b)$, and $w$ using eqns. (2) and (3) below, respectively. The general paradigm is that if an input field’s flag is set then the derived field’s is also, and thus, the initial QC flag for the derived field is simply the logical OR of its inputs. The derived field’s confidence is initially set to the minimum input confidence discounted by a factor based on the other input confidences. Specifically, the derived confidence is $c_j \prod_{i \neq j} f(c_i)$ where $c_i$ are the enumerated input confidences, $j = \arg \min_i c_i$ and $f$ is the piecewise linear function with vertices (0,0), (0.5,1), (1,1). Thus, the discounted factor only has an impact if one or more of the remaining input confidences is below 0.5. $\alpha_a$ is a special case because it is nominally just the average of two values that should be about the same. If one is bad but the other is good, then the good value can be used and the bad ignored. So, for $\alpha_a$, if both input confidences are above the threshold (0.5) then the confidence is the average of the two input confidences. If only one input is above the threshold confidence, then the resultant confidence is the confidence of the good value scaled by a penalty (nominally 0.8). If both inputs are below the threshold then the confidence for $\alpha_a$ is set to 0.

The median QC algorithm is designed to detect spikes and steps. Because of the novelty of the median QC algorithm, it is described in detail here. There are three separate components. The first component, the so-called median QC z-confidence, is somewhat inspired by the median absolute deviation (MAD) method discussed in Iglewicz and Hoaglin (1993). The idea is to look for points that seem to deviate more than would be expected given the variation of the data in a window centered on that point. This algorithm uses percentiles rather than averages and standard
The NCAR In Situ Turbulence Detection Algorithm

deviations because percentiles are more robust in the presence of outliers, which are what the algorithm is trying to detect. Details of this procedure are as follows.

First, the 5\textsuperscript{th}, 10\textsuperscript{th}, 15\textsuperscript{th}, 20\textsuperscript{th}, 25\textsuperscript{th}, 30\textsuperscript{th}, 50\textsuperscript{th}, 70\textsuperscript{th}, 75\textsuperscript{th}, 80\textsuperscript{th}, 85\textsuperscript{th}, 90\textsuperscript{th}, and 95\textsuperscript{th} percentiles are computed on a sliding window of data. Currently, the window size is 31. The value to be quality controlled is always the center value of the window. Values in the window that have below-threshold confidence (0.5), as input into the median QC algorithm (e.g. based on bounds checking and the validity flag, if available), are not used to compute the percentiles. As mentioned above, the idea for z-confidence median QC confidence is to compute a robust version of the z-statistic. Namely,

$$z = \frac{x - P_{50}}{R_n}$$

where \(R_n = \max (P_{100-n} - P_n, \text{min}_IQR)\), and \(P_n\) is the \(n\textsuperscript{th}\) percentile, and \(\text{min}_IQR\) is a field-dependent constant, which is chosen to ensure the denominator does not get too small. The algorithm to choose \(n\) is described below. The value of \(z\) is then input into a piece-wise linear confidence map to generate the so-called z-confidence. The vertices of the piece-wise linear confidence map may in general depend on the field, but the ones used here are (0,1), (conf\_lb,1), (conf\_ub,0), (\(\infty\),0), where conf\_lb and conf\_ub are 1.21 and 2.21, respectively. Robust statistics are used to mitigate the effects of large outliers in the window that would significantly bias the mean and standard deviation. The dynamic choice of \(n\) is guided by the balance of two competing issues. If \(n\) is chosen to be small (say 5), then most points (nominally, all but the highest and lowest two points out of the 31) in the window are used to determine the percentile range, used in the denominator of eqn. (1). Thus, if there is a short, high intensity turbulence event within the window, the value of \(R_n\) will be larger, ensuring that \(z\) is small. This is the desired effect in the case that the point being evaluated (center) is part of the turbulence. On the other hand, if a larger
number of points are spikes or constitute a step, then the small value of $z$ is deleterious. In other
words, small values of $n$ have fewer false alarms for spikes, but also a lower detection rate.
Contrariwise, larger values of $n$ have more false alarms for spikes/steps with a higher detection
rate. To improve the skill, instead of picking only one value, $R_n$ is computed for $n$ at 5, 10, 15, 20,
25, and 30. The adjacent ratios are computed (namely $R_5/R_{10}$, $R_{10}/R_{15}$, $R_{15}/R_{20}$, $R_{20}/R_{25}$, and
$R_{25}/R_{30}$). The condition $R_{n-5} / R_n > T_R$ ($T_R$ is currently set at 4.5) is checked for $n = 10, 15, \ldots, 30$.
If the condition is true for any $n$, then the $R_n$ for the largest such $n$ is used to compute $z$. If all ratios
are below threshold, then $R_5$, is used. This procedure will tend to eliminate spikes and steps from
the calculation of $R_n$. The principal behind this is that in the case of good winds data these
percentile-ranges decrease more slowly/continuously than in the case of the existence of data
quality problems. In the latter case, there is often a big drop (corresponding to a large ratio). This
will not work if there are a very large number of spikes/steps or if the data gradually transitions
from nominal into a step. However, these cases appear to be quite rare.

To illustrate the choice of $R_n$, consider an example in which the vertical wind data is a
constant 0 throughout a 31 point time sequence, except that the last 3 values are 30 m/s. Let the
$min_{IQR}$ be 1.0. $P_5$ through $P_{85}$ would be 0, $P_{90}$ would be 12 (although it depends exactly on how
the percentile is calculated) and $P_{95}$ would be 30. Thus, $R_5 = 30$, $R_{10} = 12$, $R_{15} = R_{20} = R_{25} = R_{30} = min_{IQR} = 1.0$. Finally, $R_5/R_{10} = 2.5$, $R_{10}/R_{15} = 12.0$, $R_{15}/R_{20} = R_{20}/R_{25} = R_{25}/R_{30} = 1.0$.
Since $R_{10}/R_{15} = 12.0 > T_R = 4.5$, then $n = 15$ is chosen (i.e. $R_{15}$ is used in eqn. (1)).

Note that the thresholds for the piecewise-linear confidence map are the same even if $R_5$ is
not used. This is done for simplicity, but it has the benefit that if a different $R_n$ is picked than $R_5$,
the result is a generally larger $z$, and hence, a lower resultant confidence. To illustrate this, consider
that $R_{n-5} \geq R_n$, and therefore $z$ calculated from $R_n$ will be larger than if it was calculated using $R_{n-5}$, which will have a tendency to lower the confidence.

Additionally, two more confidences are computed and combined with the $z$-confidence to arrive at the final median QC confidence. The first, the so-called percentile-range confidence, is computed by applying a piecewise-linear confidence map to $R_n$, with larger values of $R_n$ associated with low confidences: $(0,1)$, $(IQR_{conf_{lb}},1)$, $(IQR_{conf_{ub}},0)$, $(\infty,0)$ This helps guard against the case where the $z$-confidence is low because there are a large number of spikes or a long step that results in a large $R_n$. In other words, the percentile-range confidence is lowered in cases with abnormally large variation within the 31 point window. Of course, care must be taken to set the parameters of the confidence map large enough to ensure that legitimate severe turbulence is not penalized. The values for this map are highly dependent on the field and reflect how much the field can vary within the 31 points before we should consider the data as bad.

The second, the so-called percent-good confidence, is computed by applying a piecewise-linear confidence map to the number of points in the window with higher than threshold (nominally 0.5) confidence as determined before applying the median QC algorithm. If this was done afterwards then the algorithm would frequently get stuck in a state determining all data as bad. The vertices are $(0,0)$, $(num_{good_{lb}},0)$, $(num_{good_{ub}},1)$, $(\infty,1)$, with $num_{good_{lb}}$ and $num_{good_{ub}}$ set to 20 and 25, respectively. The final median QC confidence is simply the product of the three intermediate confidences ($z$-, percentile-range, and percent-good confidences).

For each data field, the following parameters need to be tuned: the upper and lower bounds for the bounds check, $min_{IQR}$, $T_R$, conf_{lb}, conf_{ub}, IQR_{conf_{lb}}, IQR_{conf_{ub}}$. Currently, the values of $num_{good_{lb}}$ and $num_{good_{ub}}$ are set to 20 and 25, respectively.
After the initial input bounds check and, in some cases, the median QC algorithm, data with confidence below the confidence threshold can be linearly interpolated over, as long as the number of “bad” data points in a row is three or less. The confidences at those places are also linearly interpolated over, as well. Otherwise, the data are set to “bad” (e.g. NaN) and the confidences are set to 0. The flags, however, are always left set to 1.

3.3 Vertical Wind Calculation

To compute the vertical winds from the inputs, the body-axis angle of attack ($\alpha_b$) needs to be computed. Raw vane measurements are biased due to airflow effects and hence need to be calibrated. A linear model seems to be appropriate, and is standard practice in the industry. Calibration coefficients may be available from the airframe manufacturer, but in order to not require non-proprietary data, we perform our own calibration using empirical data.

First, if available, the left and right angle of attack ($\alpha_1$ and $\alpha_2$) are averaged ($\alpha_a$). If one of them has been found to be bad (below threshold confidence), then the other is used. If both are bad, then it is flagged as bad and the confidence is set to 0. The body-axis angle of attack is computed by

$$\alpha_b = a_1 \alpha_a + a_0$$

Determining, $a_0$ and $a_1$ can be done by considering the following. If the aircraft is not changing altitude and is flying under nominally smooth conditions, then, on average, $\alpha_b \approx \theta$. The vertical wind will cause this to not be instantaneously true. Utilizing data from many flights under the proper conditions, preferably including aircraft flying on different days and different wind conditions, a least-squares linear fit can be performed for the following model $\theta = a_1 \alpha_a + a_0$. Care should be taken to limit the data to straight and level flight under smooth conditions. This is
accomplished by requiring: \( Z > 15,000 \text{ ft} \), \(|IVV| < 1.0 \text{ms}^{-1}\), \(|\ddot{Z} - 1| < .075 \text{g} \), and \(|\phi| < 3^\circ\). In practice, we employ a more involved technique because, at times, the above procedure can result in biased values of \( a_i \). The full procedure is not described here, but it is included in the EDR support package that is included in the full software package.

Finally, the vertical winds are computed from \( V_T \), \( \alpha_b \), \( \theta \), \( \phi \), and \( \dot{Z} \) (Parks, et al. 1985; Cornman, et al. 2004, Sharman et al. 2014):

\[
w = V_T (\cos \theta \sin \alpha_b \cos \phi - \cos \alpha_b \sin \theta) - \dot{Z} \tag{3}
\]

Note that a pitch rate correction factor could be included, as in Lenschow (1972), namely

\[
\frac{\pi}{180} M \dot{\theta} \cos \theta , \text{ where } M \text{ is the distance from the angle of attach vane to the center of gravity of the aircraft, in meters, but in practice this term is negligible and leads to nearly identical EDR estimates under most circumstances.}
\]

The computation of \( w \) is performed at the common sampling rate \( (f_s) \) for all of the data fields.

### 3.4 EDR Estimation

The spectral-domain based maximum likelihood estimation method of estimating \( \varepsilon \) is described in Smalikho (1997). The basic calculation is to divide, frequency bin by frequency bin, the empirical wind power spectrum by the theoretical power spectrum (i.e. the power spectrum of a turbulence model with \( \varepsilon^{2/3} = 1 \text{ m}^{4/3} \text{ s}^{-2} \)) over a certain range of frequencies, and then average the ratios to compute \( \varepsilon^{2/3} \).

To compute the empirical wind power spectrum, the data are first linearly de-trended, using a least-squares fit.

\[
w'_k = w_k - a_1 \left( k - \text{floor} \left( \frac{n'}{2} \right) \right) - a_0 \tag{4}
\]
The NCAR In Situ Turbulence Detection Algorithm

for \( k = 0, \ldots, m-1 \), where \( m = 10 f_s \) (corresponding to 10 s of wind data). \( \omega_1 \) and \( \omega_0 \) are linear and constant (resp.) coefficients from the least-squares best fit of \( w_k \), computed using the normal equations, with \( x \)-values \( k - \text{floor}\left(\frac{n}{2}\right) \) for \( k = 0, \ldots, m-1 \), and \( y \)-values \( w_k \). Note that much of the computations in the normal equations rely only on the \( x \)-values, which are fixed, and thus can be precomputed.

Next, a power normalized Tukey (tapered cosine; Harris 1978) window is applied. The \( m \)-point Tukey window can be formulated as

\[
\tau_k = \begin{cases} 
\frac{1}{2} \left(1 - \cos\left(\frac{k\pi}{M+1}\right)\right) & \text{if } 0 \leq k \leq M \\
1 & \text{if } M < k \leq m-M-2 \\
\frac{1}{2} \left(1 - \cos\left(\frac{(m-k-1)\pi}{M+1}\right)\right) & \text{if } m-M-2 < k < m-1
\end{cases}
\]  

(5)

Where the taper width \( M = \text{floor}\left(0.1m-0.2\right) \) is about 10%. The power normalized Tukey window is

\[
\tilde{\tau}_k = \frac{\tau_k}{\sqrt{\frac{1}{m} \sum_{j=0}^{m-1} \tau_j^2}}
\]  

(6)

The linear de-trended and windowed vertical wind time-series is

\[
w_{k}^{\text{dW}} = \tilde{\tau}_k w_k^d
\]  

(7)

Linear de-trending and the window function application mitigate spectral leakage, though at the cost of corrupting the wind spectrum at lower frequencies. The Fourier transform is now applied:

\[
\hat{S}_k = \frac{1}{f_s m} \left| \sum_{j=0}^{m-1} w_{j}^{\text{dW}} e^{-2\pi i jk/m} \right|^2
\]  

(8)
The NCAR In Situ Turbulence Detection Algorithm

where \( k = 0, \ldots, m-1 \), and \( i \) is the complex imaginary number. Note that the \( k^{th} \) frequency corresponds to \( kf_i / m \) Hz. This computation is typically done via fast Fourier transform (FFT), appropriately scaled, and the frequencies above \( f_s / 2 \) removed. Finally, \( \hat{\varepsilon}^{2/3} \) is computed by

\[
\hat{\varepsilon}^{2/3} = \frac{1}{k_h - k_l + 1} \sum_{k=k_l}^{k_h} \frac{\hat{S}_k}{S_k}
\]

where \( S \) is a spectral theoretical model of turbulent vertical wind and \( k_l \) and \( k_h \) are the lower and upper index bounds over which the average is taken. These index bounds correspond to the frequencies \( f_l \) and \( f_h \). The associated confidence of \( \hat{\varepsilon}^{2/3} \) is the minimum of two confidences: the average of the input \( w \) confidences, and the so-called, number-of-good-inputs confidence. The latter is computed by counting the number of times the flag for \( w \) is set. Since \( w \) is derived from all of the inputs, if any of the inputs’ flags are set, then the derived \( w \) flag will be set. This count will be a number between 0 and \( m \). A piece-wise linear function is applied to this count with vertices \((0,1)\), \((0.0625m,1)\), \((0.1250m,0)\), and \((m,0)\) to obtain the confidence.

We examined several different spectral models of turbulence for \( S \). One model that includes representations of both the inertial subrange and the larger scales beyond it, and has been used extensively by both the aerodynamics and meteorological communities is the von Kármán spectral model (e.g., Hinze 1959; Hoblit 1988; Murrow 1987; Murrow et al., 1982; Founda et al. 1997; Kristensen and Lenschow 1987). The von Kármán model is parameterized by a length scale as well as turbulence intensity (Sharman et al. 2014). Roughly speaking, the length scale\(^5\) parameterizes the scale at which the von Kármán spectrum deviates from the -5/3 power law. Ideally, the model spectrum needs to take into account the procedure for generating the empirical spectrum or else the result can be biased and will not truly be a maximum likelihood method.

\(^5\) There are actually a small number of equivalent length scales that can be used for parameterization.
The second model, which was employed in earlier versions of the wind-based algorithm, is the so-called von Kármán aliased model, here denoted as $S^{va}$, as described in Sharman et al. (2014). This spectral model is derived from the von Kármán transverse autocorrelation function (Frehlich, et al. 2001) and takes into account discrete and finite-length sampling effects as well as the window function. $S^{va}$ is the exact form of the spectrum model given that a frozen von Kármán wind field is sampled $m$ times with a $V_T / f_s$ m spacing, and including a window, $\tilde{\tau}$, applied to the data before the power spectrum is computed. The only aircraft dependent parameter for $S^{va}$ is $V_T$, and therefore, in practice, a 2-D table (frequency index vs. $V_T$) of $S^{va}$ for different true airspeeds can be computed offline and simply stored in the software. For this model, we had computed and stored model spectra for $V_T = 65, 70, \ldots, 295$ m s$^{-1}$, which is sufficient for the aircraft used to date. At run-time, the $S^{va}$ used was taken from the median $V_T$ over the same time interval as the vertical winds used to compute the power spectrum, and interpolating into the table.

The third model, and that described in this paper, and the one that is now currently used operationally, is the Kolmogorov model, $S^K$, for which the temporal spectral form is given in Sharman et al. (2014) as:

$$S^K_s = \frac{12}{55} (2\pi)^{-2/3} AV_r^{-2/3} f^{-5/3}$$

(10)

where A has been estimated to be between 1.5 to 1.7 (Kristensen and Lenschow 1987; Mann 1994; Cornman et al. 1995). This is a similar formulation to that used by MacCready (1962, 1964), though those papers described a spatial spectral form. While this spectrum does not take a number of data contamination issues into account like the previous von Kármán aliased model does, it does have a number of advantages. Obviously, it is simpler. It also happens to match better with observed data recorded from commercial aircraft. The reason for this is that the input data on
commercial aircraft have almost always been filtered during the digitization process at the sensor. This largely mitigates most of the contaminants that the procedure for generating the empirical spectrum in the previous method accounted for. Also, it is generally desirable to have one piece of software that can be deployed on different aircraft types; failure to do so drives up the cost of deployment. Avionics boxes, like the aircraft monitoring system (ACMS) have limited processing capabilities and thus directly computing the wind spectrum model in such a system has not been feasible. Even if it were feasible, there is still the issue that one might encounter an aircraft that employs different filtering techniques that are not captured by the implementation. Passing a table of model values via an initialization routine was also considered but ultimately rejected because of the risk that not all systems may be able to pass approximately 1600 (spectra for about 40 values of V each with about 40 frequency bins) values to the In Situ EDR Algorithm library. Therefore, the Kolmogorov model seems to be a suitable choice.

When using the Kolmogorov model, five issues are, thus far, unaccounted for: the unknown true length scale of the data, the linear de-trending of the data, the noise signal, the fact that the vertical winds are typically computed from filtered fields, and the effects from the sampling from a finite length window. The first three are all handled in the same way, namely by use of the cutoffs \( k_l \) and \( k_h \) in eqn (9). \( f_l \) and \( f_h \), the frequencies corresponding to indices \( k_l \) and \( k_h \) (resp.) are set to 0.5 and 3.5 Hz for the 8 Hz implementations. The effects due to the unknown length scale and the linear-detrending of the data are largely limited to the lowest frequencies. Noise mostly affects the higher frequencies since the turbulence spectrum has the lowest power there, and only during smooth conditions. The impact of noise in even quite small levels of turbulence is negligible.
The NCAR In Situ Turbulence Detection Algorithm

The aircraft fields used to compute the winds are typically filtered in the analog-to-digital (A/D) conversion. In the case of commercial aircraft the details of the filtering are often difficult to obtain. Additionally, $w$ is a non-linear function of the input fields (see eqn. (3)), though most of the time, for small angle fluctuations, it is approximately linear. However, the filters used from aircraft type to aircraft type seem to vary.

To address this issue, we utilized a so-called pseudo-maximum likelihood method, i.e. a maximum likelihood method employing a model that does not exactly match the data (e.g. Gong and Samaniego, 1981; Gourieroux, et al., 1984). In our case, instead of using the tailored filtered, aliased, von Kármán model (denoted here as $S^{\text{vaf}}$), the Kolmogorov model is used. This will lead to a biased estimate of $\varepsilon^{2/3}$. However, the multiplicative bias due to using the “wrong” model is generally fixed, assuming the correct model for the data still scales with $\varepsilon^{2/3}$. Namely, the expected relative bias can be shown to be given by:

\[
\frac{1}{k_h - k_i + 1} \sum_{k=k_i}^{k_h} \frac{S_k^{\text{vaf}}}{S_k^K}
\]

(11)

Assuming the independence of the errors of $\hat{S}_k$ for different $k$, the variance can thus be shown to be increased by the multiplicative factor:

\[
\frac{\gamma^2}{k_h - k_i + 1} \left( \sum_{k=k_i}^{k_h} \frac{S_k^{\text{vaf}}}{S_k^K} \right)^2
\]

(12)

where the correction factor, $\gamma$, is set to the inverse of eqn. (11). Thus, if the exact correct model is known, the bias induced from using $S^K$ can be corrected, at the cost having an increased variance. This allows the software to always use the same model, namely $S^K$, regardless of the actual correct model. Note that if the two models differ greatly, the increase in variance can be significant.

However, the exact “correct” model is typically not known, and therefore a new procedure was developed. The idea behind this procedure is that effects from all of the contaminating sources
found to date, do not affect the spectrum in the 0.5 to 1.5 Hz frequency range for an 8 Hz sampling frequency. The following procedure can be performed to arrive at the correction factor. Over a large dataset of aircraft data with a number of turbulence events, compute \( \hat{e}^{2/3}_K \) (\( \hat{e}^{2/3} \) based on \( S^K \)) with no correction factor, but only over a limited band of frequencies 0.5 to 1.5 Hz. Second, compute \( \hat{e}^{2/3}_K \) over the full band of desired frequencies (e.g. 0.5 Hz to 3.5 Hz for \( f_s = 8 \) Hz). Lastly, compute the slope of the errors-in-variables linear regression (Parks, et al., 1989). This has experimentally been shown to well approximate \( \gamma \).

### 3.5 Mean and Peak EDR Estimation

The bias corrected \( \hat{e}^{2/3} \) is computed every 5 s. While in principal, these “raw” values could be reported, the downlink costs would currently be prohibitive. Instead, the mean and peak EDR values are computed and reported. The computation of the Mean and Peak EDR, along with their confidences are as follows.

Once per minute, the peak and confidence weighted \( \hat{e}^{2/3} \) are computed. The confidence weighted mean is computed using only those \( \hat{e}^{2/3} \) that are above a defined confidence threshold (nominally 0.5). The confidence of the mean is simply the average of the input confidences where confidences below threshold are set to 0. The peak is the largest \( \hat{e}^{2/3} \) above the defined confidence threshold, and its corresponding confidence is simply the confidence of that particular \( \hat{e}^{2/3} \) discounted by 1/12 (since there are 12 \( \hat{e}^{2/3} \) measurements per minute) for every \( \hat{e}^{2/3} \) that was greater but had a below-confidence threshold. The square root of the mean and peak \( \hat{e}^{2/3} \) are reported along with their associated confidences.
3.6 EDR Algorithm Steps

1. The software is initialized with parameters specific to that aircraft type, including the angle of attack calibration constants, the frequency cutoffs, and the EDR scale factor, $\gamma$.

2. After the aircraft has taken off, the required data is passed into the EDR software.

3. Each field is checked to see whether the values are valid and in-bounds. If not, the corresponding flags are set to 1 and the confidences are set to 0.

4. Values with flags set are interpolated over, if possible (less than 3 bad in a row), as well as the corresponding confidences. The flags remain unchanged.

5. The average angle of attack is computed as well as its confidence and flag.

6. The body axis angle of attack is computed, as well as its confidence and flag (eqn. (2)).

7. The vertical wind is computed as well as its confidence and flag (eqn. (3)).

8. The vertical wind is checked to see whether the values are in-bounds. If not, the corresponding flags are set to 1 and the confidences are set to 0.

9. The true airspeed and the vertical wind is then processed using the median QC algorithm, and their confidences and flags are updated.

10. Values with flags set are interpolated over, if possible (less than 3 bad in a row). The Median QC confidence is now recomputed using the newly interpolated values and combined with the interpolated confidence. The flags remain unchanged.

11. Every 5 seconds, an $\hat{\epsilon}^{2/3}$ is computed using 10 seconds of wind data. If there are any missing wind data, the $\hat{\epsilon}^{2/3}$ is simply set to missing and its confidence set to 0.
   a. The vertical wind data are de-trended.
   b. The de-trended wind data are windowed (eqn. (7)).
c. The empirical power spectrum of the windowed de-trended data is computed (eqn. (8)).

d. The model spectrum is retrieved from a table based on the median TAS.

e. $\hat{\varepsilon}^{2/3}$ is computed (eqn. (9)) and scaled by $\gamma$. Its confidence is computed based and the average of the confidences of the input winds, and the confidence based on the number of flagged values within the 10 second window.

12. Every minute, the mean and peak $\hat{\varepsilon}^{2/3}$, along with their associated confidences are computed using the last 12 $\hat{\varepsilon}^{2/3}$ and their confidences. The square root of the mean and peak $\hat{\varepsilon}^{2/3}$ are taken to produce the mean and peak EDR.

13. The Mean and Peak EDR, their confidences, along with the total number of flagged winds (which inherits any flagged values from inputs), the peak location (a fraction that describes when within the minute the peak occurred), the number of $\hat{\varepsilon}^{2/3}$ that went into the mean and peak calculation that had above threshold confidence (0.5), along with a version number, are all encoded for downlinking.

3.7 Performance

Statistical evaluation of the In Situ EDR algorithm performance on an aircraft is made difficult owing to the lack of “truth”, i.e. baseline values to compare against. An alternative is to use turbulence simulations (Frehlich et al. 2001, Cornman 2016). Thousands of realizations of a turbulent vertical wind time-series can be readily created with a known mean EDR and then input into the In Situ EDR Algorithm. This does not address the question of how well the vertical winds can be estimated (section 3.3), but rather only the performance of the maximum likelihood estimation algorithm (section 3.4). However, one can add artifacts to the simulated data to study the effects from noise, spikes, up-stream filtering, etc. Also, the vertical wind time-series can be
modulated to emulate short discrete events of varying length and intensity. Whether simply modulating a homogenous turbulence wind field to simulate a discrete event is realistic or not, is an open question (Cornman 2016).

A significantly more complicated approach (Emanuel et al. 2014, 2017b) is to input the vertical wind time-series into an aircraft simulator to produce aircraft data, including angle of attack, true airspeed, pitch, roll, etc. These aircraft data can incorporate artifacts from sampling, filtering, quantization, etc. The data can then be input into the In Situ EDR algorithm and compared against the original EDR used to generate the simulated vertical winds. The use of an aircraft simulator adds significant complication, and now one needs to verify that the aircraft simulator produces “realistic” results. This full procedure has been done by Emanuel et al. (2014, 2017b), for a Boeing 747-100 aircraft. In fact, this approach is the basis for the EDR algorithm testing procedure presented in Emanuel et al. (2017a). The results, however, from this more complicated approach are not much different than the results from the simpler method, and thus we only present results using the latter.

Here we provide some indication of the accuracy of the algorithm based on simulated turbulence data. For each scenario below, 6,000 realizations of vertical wind time-series corresponding to homogeneous turbulence, sampled at 8 Hz by an “aircraft” flying at 250 ms\(^{-1}\), are created using a von Kármán turbulence model with integral length scale 500 m. Further, the winds are simulated to include the effects from a 2-pole Butterworth analog low-pass filter with cutoff frequency at 3 Hz, using the Matlab® function butter\(^6\). The Butterworth filter is a simple infinite impulse response (IIR) filter that has a monotonic amplitude response in both the passband and stop band, and is a plausible filter to be used for the purpose of dealing with the data in the...

\(^6\) https://www.mathworks.com/help/signal/ref/butter.html
The NCAR In Situ Turbulence Detection Algorithm

process of digitization of an analog sensor data. While the EDR performance will vary depending on the filter used, our experience is that the dependence is weak due to the fact that the frequency range used by the algorithm, 0.5 to 3.5 Hz, is mostly within the pass-band of any filter likely to be used for the purposes of digitization. A much larger impact would be seen if no de-aliasing filter had been applied, though this situation is rare in our experience. Finally, the scale factor $\gamma$ (eqn (12)) is estimated as described in section 3.4.

The performance of the EDR estimates for the In Situ EDR Algorithm are shown in Figure 3 through Figure 5 as box plots. Since the details of how box plots are defined vary, we describe our approach here. The x-axis corresponds with the simulation input EDR. The top, “middle”, and bottom of each box are the 75th, 50th, and 25th percentiles (resp.) of the estimated EDRs. The upper dashed black lines (the so-called whiskers) extend to the largest data value less than 1.5 times the interquartile range, defined as the difference between the 75th and 25th percentile, above the 75th percentile. The lower dashed black lines are defined analogously. Red plus symbols are used to mark where location of any remaining estimates beyond the whiskers.

---

7 It is of general practice to sample an analog sensor at a rate significantly higher than the desired final sampling rate, applying a low-pass filter to significantly attenuate the signal beyond the Nyquist frequency (half of the desired sampling rate), and finally down sampling the signal to the final sampling rate.
Figure 3: Box plot of the performance of the In Situ EDR estimator, for varying values of the simulation input EDR (x-axis). N=6,000 vertical wind time-series were used for each value of simulation input EDR. The solid black line shows the y=x reference line. The relative statistics of bias, standard deviation, and 95% confidence interval for the simulation input EDR of 0.3 m$^{2/3}$s$^{-1}$ are provided at the top left. The relative statistics are calculated relative to the simulation input (0.3 m$^{2/3}$s$^{-1}$). However, they are representative for all simulation input EDRs.

Figure 3 shows a box plot of the individual EDR estimates for the In Situ EDR Algorithm, estimated from 10 seconds of vertical wind time-series (80 points at 8 Hz). As can be seen, the spread varies with simulation input EDR. However, the relative bias, standard deviation, and 95% confidence interval statistics, relative to the simulation input EDR, are independent of the simulation input EDR. The relative bias is 0.1%, the relative standard deviation is 9.7%, and the 95% confidence interval is (-15.9%,+16.2%).
Figure 4: Box plot of the performance of the In Situ Mean EDR estimator, for varying values of the simulation input EDR (x-axis). N=6,000 vertical wind time-series were used for each value of simulation input EDR. The solid black line shows the y=x reference line. The relative statistics of bias, standard deviation, and 95% confidence interval for the simulation input EDR of 0.3 m$^{2/3}$s$^{-1}$ are provided at the top left. The relative statistics are calculated relative to the simulation input (0.3 m$^{2/3}$s$^{-1}$). However, they are representative for all simulation input EDRs.

Figure 4 shows a box plot of the Mean EDR estimates for the In Situ EDR Algorithm, estimated from 12 individual EDRs. For these simulations, the length of each realization of the vertical wind time-series is 520 points, corresponding to one minute of data$^8$. The results, as expected, are better than for individual EDRs. The relative bias is 0.2% (slightly larger, though still very small), the relative standard deviation is 3.8%, and the 95% confidence interval is (-5.9%,+6.7%).

$^8$ EDRs are computed using 10 s of data every 5 s. Thus, in an operational system, the first EDR of the minute contains 5 s of data from the previous minute. Thus, we have 480+40=520, for 8 Hz data. If the time-series length was only 480, then only 11 EDR values would be able to be computed.
Figure 5: Box plot of the performance of the In Situ Peak EDR estimator, for varying values of the simulation input EDR (x-axis). N=6,000 vertical wind time-series were used for each value of simulation input EDR. The relative statistics of bias, standard deviation, and 95% confidence interval for the simulation input EDR of $0.3 \text{ m}^{2/3}\text{ s}^{-1}$ are provided at the top left. The relative standard deviation and 95% confidence interval are relative to the average Peak EDR. However, they are representative for all simulation input EDRs. The solid black line shows the expected average Peak EDR based on the reference peak EDR increase for each simulation input EDR.

Figure 5 shows a box plot of the Peak EDR estimates for the In Situ EDR Algorithm, estimated from 12 individual EDRs. For these simulations, the length of the vertical wind time-series is also 520 points. As before, it can be seen the estimator spread increases with increasing simulation input EDR. It can also be seen that the median of the Peak EDR values are larger than the simulation input EDRs. Thus, if the simulation input EDR values were used to compute the relative statistics, the results would appear to be strange. To see what the “expected” Peak EDR of homogeneous turbulence should be, an exploration of the distribution of the peak or maximum of 12 individual EDR values is required.
Consider the following thought experiment. Assume that random variables $X_i$, for $i=1,\ldots,12$, are independent and normally distributed, with mean 0.3 and standard deviation 9.7% of 0.3 (≈0.0292), with cumulative distribution function denoted $F$. This is a reasonable approximation of the distribution of the individual EDR estimates\(^9\) for 1 minute, though certainly the EDRs are not truly independent. The cumulative distribution function of the $\max_i\{X_i\}$, for $i=1,\ldots,12$, is simply $F^{12}$ (i.e. the 12th power of the cumulative distribution function). It can be shown that if $X_i$ are only positive real-valued (very approximately true), then the expected value of $\max_i\{X_i\}$ can be calculated, using a well-known relation between the CDF of a distribution and its expected value, as $\int_0^{\infty} 1 - F^{12}(x) \, dx$. This integral can be approximated numerically to be about 0.3473, an increase of about 15.8% from 0.3. We call this the reference Peak EDR increase. Now clearly the assumptions in this thought experiment are not strictly true. However, the empirically estimated increase of the Peak EDR compared to the simulation input EDR is approximately 15.0%, and thus this thought experiment does capture the essence of the issue. The approximated bias of the average Peak EDR relative to the reference Peak EDR increase (i.e. relative to 0.3473 for the simulation input EDR of 0.3 m\(^{2/3}\)s\(^{-1}\)) is -0.7%. Both the relative standard deviation (5.5%) and the 95% confidence interval (-8.5%,+9.5%) are relative to the empirically calculated average Peak EDR.

To continue on with the thought experiment, we can explore the resultant reference Peak EDR increase as a function of the standard deviation. The procedure above is repeated, but with varying the standard deviation of $X_i$, from 0% to 20% of 0.3. The results are shown in Figure 6\(^\text{Error! Reference source not found.}\), with the relative standard deviation as the $x$-axis, and the expected relative increase of the Peak EDR as the $y$-axis. What this shows in that the performance

\(^9\) In reality, the distribution of EDR estimates are better approximated by a generalized gamma distribution.
The NCAR *In Situ* Turbulence Detection Algorithm

Relative increase of peakEDR to EDR vs Relative Stddev of EDR

Figure 6 A line plot showing the relationship between the relative increase of Peak EDR from the simulation input EDR as a function of the relative standard deviation of EDR. This is labelled *reference peak EDR increase* in Figure 5, and is the expected relative increase of the peak EDR relative to the simulation input EDR due to the procedure of taking the maximum value of 12 EDRs, assuming the EDRs are independent and normally distributed with relative bias of 0% and varying relative standard deviations (x-axis).

The relative standard deviation of the individual EDR estimator is an important contributor to the expected value of the Peak EDR. If the relative standard deviation of the individual EDR is about 15%, then the expected Peak EDR estimate would be about 25% larger than the simulation input EDR, rather than the approximately 15% of the current algorithm. The slope of the line is approximately 1.6, signifying that for each 1% increase in the relative standard deviation of the EDR value, results in an increase of 1.6% larger expected Peak EDR relative to the simulation input EDR. Thus, we conclude that the “expected” peak EDR for homogeneous turbulence depends on the standard deviation of the EDR estimator. For example, employing the same algorithm design on 4 Hz data will, all things being
The NCAR In Situ Turbulence Detection Algorithm

equal, increase the standard deviation of the EDR estimate, and will therefore increase the expected peak EDR for homogeneous turbulence.

4 Event Based Triggering Logic

Ideally, all once-per-minute turbulence measurements would be downlinked, but, using current standard air-to-ground communications (e.g. ACARS), this would result in substantial costs to downlink mostly null turbulence (smooth) reports. Some null turbulence reports are helpful. For example, a pilot or dispatcher can see where aircraft are experiencing smooth conditions and possibly route other aircraft into that area. Additionally, forecasting models need both positive (turbulence) and negative (smooth) events to be skillful. To try to balance these needs, a report triggering logic was developed and implemented to decide which EDR measurements should be immediately reported.

The reporting scheme augments routine reports (sometimes referred to as heartbeats), spaced 15 to 20 minutes apart and consisting of just one mean/peak EDR measurement, with turbulence event-based triggering logic.

4.1 Turbulence Event Reports

There are 3 types of triggering events.

- Type 1: Higher intensity experienced.
- Type 2: Fairly consistent medium intensity experienced over the last 6 minutes.
- Type 3: Consistent lower intensity experienced over the last 6 minutes.
More formally:

- **Type 1:** peak EDR ≥ T1 from the last minute. 6 minutes later a follow-up report is also generated.
- **Type 2:** peak EDR ≥ T2 for at least 3 out of the last 6 minutes. All 6 measurements are reported. 6 minutes later a follow-up report is also generated.
- **Type 3:** mean EDR ≥ T3 for at least 4 out of the last 6 minutes. All 6 minutes are reported. No follow-up report is reported.

Follow-up reports are generated after type 1 and 2 events so that users can see “both sides” of the event (before and after). Otherwise, users would only see data leading up to and including the event. Follow-up reports always contain 6 measurements.

The chosen values of T1, T2, and T3 in all of the implementations thus far are:

\[
T_1 = 0.18 \, m^{2/3} s^{-1} \\
T_2 = 0.12 \, m^{2/3} s^{-1} \\
T_3 = 0.06 \, m^{2/3} s^{-1}
\]
In our experience, the bulk of the event triggered reports are Type 1 and their associated follow-ups will vary according to the turbulence climatology over the route structure of the airline.

Further, the logic is designed to limit the possibility that mean/peak EDR measurements are reported more than once by keeping track of which measurements have been previously downlinked.

4.2 Routine Reports

The goal for downlinking routine reports is to provide some null turbulence data. Also, some turbulence information can be inferred between reports (whether routine or event-based). Namely, the turbulence was not strong enough to trigger and event-based report, and thus the peak EDR had to be less than \(0.18 \text{ m}^{2/3} \text{s}^{-1}\). Using this information, it is possible to fill in the gaps. The times of these “missing” (i.e. not reported) EDRs is easy to approximate since EDR measurements are spaced by 1 minute. The 3-D positions can be linearly interpolated, which generally should lead to good results. It is possible to use the Aircraft Situational Display to Industry (ASDI) data stream to derive better positional information if required.

The first routine report is generated immediately after the first mean/peak EDR is computed, which ideally should be one minute after the take-off. Routine reports are generated thereafter at a configurable but constant time interval, ideally less than 20 minutes. The only time routine reports are not generated is if an event-based report is being generated at the same time. If there are any unreported mean/peak EDRs at wheels-down, the last routine report is generated and downlinked.
An example showing the behavior of the event-based triggering logic can be seen in Figure 8. Two flight tracks are shown from different flights. The eastern track is from an aircraft flying north-northwest, and the western track is from an aircraft flying south-southeast. The aircraft corresponding to the eastern track, experienced light turbulence just north of the Kentucky border, which triggered a type 3 report. Next, a routine report was generated over southern Indiana. Further along in Indiana, it experienced stronger turbulence, causing a number of EDRs to be downlinked from three type 1 reports and a follow-up report. Finally, another routine report was generated over the Illinois-Wisconsin border. The aircraft corresponding to the western track did not experience enough turbulence to trigger any event-based reporting, and thus only four routine reports are seen.

Figure 8: An example of the reports that were downlinked as a result of the event-based triggering logic. The flight track to the east is from a Delta flight going north, which experienced light turbulence just north of the Kentucky border, and again through much of Indiana. Two routine reports can be seen as the isolated reports. The flight track to the west is from a Delta flight that experienced minimal turbulence, and so only routine reports were downlinked. Figure originally from Sharman et al. 2014, copyright American Meteorological Society.
5 Deployment

The NCAR In Situ Turbulence Algorithm generally requires a number of steps in order to be deployed, whether operationally on a fleet of commercial aircraft, or for post-processing of recorded research aircraft data. However, under certain circumstances, “off the shelf” systems including the software are available. The steps in general depend on the target aircraft type and avionics system. See Section 8: Appendix: Detailed Deployment Steps.

The general steps are as follows. The requirements need to be checked for both the software and the algorithm. To prepare for data quality checking and tuning, the data fields in Table 1 need to be recorded from a number of flights from the target aircraft type. To be able to have confidence in the tuning, it is required to have at least 20 or so flights, though initially, one might record only a small number of flights and do a visual inspection of the data. Once enough data is collected, the data is input into the EDR Support software, which checks for data quality problems, correct orientation of certain fields (e.g. inertial vertical velocity), reasonable correlations between certain fields, etc. Further, it calibrates the angle of attack (into body-axis), and estimates $\gamma$, and outputs the final collection of adjustable parameters for that aircraft type, along with diagnostic information that can be checked to ensure that the tuning process operated smoothly.

Once this is done, or in parallel with the previous step, the C code is integrated into whatever system will run the code, e.g. an avionics box (ACMS), including handling of the reporting. This task can be quite difficult and technical, depending on the system. On a Linux system for processing recorded data, this is quite simple, but avionics boxes can be quite closed requiring working with the avionics manufacturer.
The NCAR In Situ Turbulence Detection Algorithm

After the C code is integrated into the system, the ideal scenario is to again record the high-rate data in Table 1, along with the output. The EDR Support software can then be used to check that the onboard software is working properly. Additionally, the onboard software can be queried for some diagnostic information.

6 Conclusions

An In Situ Turbulence Detection Algorithm has been developed and implemented in a C code library for use by the aviation community, and is described in detail in this report. The In Situ EDR Algorithm estimates a mean and peak EDR every minute and includes an event-based report triggering algorithm to lower the downlink costs without sacrificing too much data. An integrated quality control algorithm was designed to limit the amount of false turbulence detections while not impacting estimates of actual turbulence. The algorithm has been carefully tested against simulations that indicate very good statistical performance. The algorithm is, at the time of writing, operating on about 1,400 aircraft worldwide, with many more expected.
7 References


Farrar, T. J., 2013: Integration of Real-time In Situ Turbulence Reports Into Airline Flight Operations.  *16th Conf. of Aviation, Range and Aerospace Meteorology, Austin, TX.*


8 Appendix: Detailed Deployment Steps

The work flow for an EDR implementation can vary widely depending on a number of circumstances. This section attempts to break down the steps for deployment of the software as of the writing of this report. However, it should be noted that available options are changing rapidly as avionics manufacturers and airlines continue to work to incorporate the software onto new fleets.

Note that the references to specific companies in the text below do not constitute an endorsement of them or their products. These are all of the companies that worked with NCAR to integrate the NCAR In Situ Turbulence Detection Algorithm into the ACMS, and currently offer it, as of the writing of this document. This appendix is meant only to provide information that might be useful for those looking to make use of the NCAR In Situ Turbulence Detection Algorithm.

8.1 First steps

Identify the preliminary deployment target(s) by answering the following questions:

1. What aircraft types are under consideration?
2. For each aircraft type, what is the desired computing platform?
   ○ ACMS
   ○ AID/EFB
   ○ Other

   Next explore the ramifications of this by working through the deployment target section.

   Iterate as needed to determine the finalized deployment target.

8.2 Deployment Target

Is the AID/EFB type solution desired or required?
An AID/EFB solution is generally always possible, but none have been deployed to date. See Non-ACMS Solutions.

Assuming an ACMS/F deployment is desired or required, what kind of aircraft are targeted?
The NCAR *In Situ* Turbulence Detection Algorithm

Option Matrix

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>ACMS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Teledyne</td>
</tr>
<tr>
<td>Boeing 777, 787, 777X, 737MAX</td>
<td>1,5</td>
</tr>
<tr>
<td>Boeing 737NG</td>
<td>2,3</td>
</tr>
<tr>
<td>Boeing 767-300/400, Airbus A319, A320, A321, A330</td>
<td>3,5</td>
</tr>
<tr>
<td>All others</td>
<td>6,7</td>
</tr>
</tbody>
</table>

1. Boeing has developed an ACMS/F solution. See Boeing ACMS Solution
2. Teledyne has developed an ACMS solution. See Existing Teledyne ACMS Solution for an existing aircraft type.
3. For a Teledyne ACMS, it is possible for integration to be performed by the airline. See New ACMS Solution on Teledyne ACMS for an existing aircraft type.
4. For a different ACMS type, see New ACMS with a new manufacturer on an existing aircraft type.
5. Teledyne has not developed an ACMS solution. They could be approached to port the existing solution from the 737NG to these aircraft types. See New Teledyne ACMS Solution on an existing aircraft type.
6. Teledyne has not developed an ACMS solution. They could be approached to port the existing solution from the 737NG to these aircraft types. See New Teledyne ACMS Implementation for a new aircraft type.
7. For a Teledyne ACMS, it is possible for integration to be performed by the airline. See New Teledyne ACMS Implementation for a new aircraft type.
8. For a different ACMS type, see New ACMS with a new manufacturer on a new aircraft type.

8.3 Boeing ACMS Solution

In this scenario, Boeing is simply approached, and the option for the EDR software is purchased. In most cases, all Boeing 777, 787, 777X, and 737MAX being delivered from the factory come equipped with the EDR software already pre-installed on the aircraft and need only be activated (a trial/testing period is also possible). In other cases, the EDR software will need to be installed through a software update. The only alternative on these aircraft is to do a non-ACMS solution.

8.4 Existing Teledyne ACMS Solution for an existing aircraft type

In this scenario, Teledyne is simply approached, and the option for the EDR software is purchased. The EDR software will need to be installed through a software update.

8.5 New Teledyne ACMS Implementation for an existing aircraft type

In this scenario, Teledyne has not ported the software to this type of aircraft, but a tuning exists for this aircraft type. Teledyne could be approached to port the software to the ACMS for this type of aircraft. Steps include:

1. Airline approaches Teledyne
2. Teledyne ports EDR software to the ACMS for this type of aircraft
3. NCAR works with Teledyne
   a. Ensure that the sources and labels for data inputs match previous implementation of this aircraft type
   b. Provide tuning for this aircraft type
   c. Assist in testing
4. Airline tests ACMS new build with EDR software included on a small number of aircraft
5. Once the build is working properly, it can be deployed throughout the fleet.
8.6 New Teledyne ACMS Implementation for a new aircraft type

In this scenario, Teledyne has not ported the software to this type of aircraft, but a tuning does not exist for this aircraft type. Teledyne could be approached to port the software to the ACMS for this type of aircraft. This presents a more difficult process than if the tuning for this aircraft type already exists because recording of the high-rate input data for a number of flights will be required. Steps include:

1. Airline approaches Teledyne
2. The data requirements (available fields and sampling frequencies) are verified by Teledyne.
3. Airline works with Teledyne to record high-rate input data for a number of flights (~20-30). Note: QAR data has not been sufficient. This step has required an airline to install a card onto the ACMS for storage and a special report is generated and recorded to the card. Delta Airlines has done this, and can be contacted for more information.
4. The data is processed through the EDR Support software to test the quality of the data and produce a tuning. It may be required to iterate on steps 3 and 4 if data quality problems are identified.
5. Teledyne ports EDR software to the ACMS for this type of aircraft
6. NCAR works with Teledyne
   a. Ensure that the sources and labels for data inputs match previous implementation of this aircraft type
   b. Provide tuning for this aircraft type
   c. Assist in testing
7. Airline tests ACMS new build with EDR software included on a small number of aircraft
8. Once the build is working properly, it can be deployed throughout the fleet.

8.7 New ACMS Solution on Teledyne ACMS for an existing aircraft type

In this scenario, the airline wishes to integrate the software themselves to this type of aircraft, and a tuning exists for this aircraft type. Delta Airlines has accomplished their deployments this way, and they can be consulted for help with integration. Steps include:

1. Airline receives software and documentation.
2. Airline reviews documentation.
3. Airline integrates EDR software to the ACMS for this type of aircraft
4. NCAR works with the Airline to
The NCAR In Situ Turbulence Detection Algorithm

- Ensure that the sources and labels for data inputs match previous implementation of this aircraft type
- Provide tuning for this aircraft type
- Assist in testing

5. Airline tests ACMS new build with EDR software included on a small number of aircraft
6. Once the build is working properly, it can be deployed throughout the fleet.

8.8 New ACMS Solution on Teledyne ACMS for a new aircraft type

In this scenario, the airline wishes to integrate the software themselves to this type of aircraft, and a tuning exists for this aircraft type. Delta Airlines has accomplished their deployments this way, and they can be consulted for help with integration. Steps include:

1. Airline receives software and documentation.
2. Airline reviews documentation.
3. The data requirements (available fields and sampling frequencies) are verified by Teledyne.
4. Airline records high-rate input data for a number of flights (~20-30). Note: QAR data has not been sufficient. This step has required an airline to install a card onto the ACMS for storage and a special report is generated and recorded to the card. Delta Airlines has done this, and can be contacted for more information.
5. The data is processed through the EDR Support software to test the quality of the data and produce a tuning. It may be required to iterate on steps 4 and 5 if data quality problems are identified. This work can be done by the airline or by NCAR.
6. Airline integrates EDR software to the ACMS for this type of aircraft
7. NCAR works with the Airline to
   a. Ensure that the sources and labels for data inputs match previous implementation of this aircraft type
   b. Assist in testing
8. Airline tests ACMS new build with EDR software included on a small number of aircraft
9. Once the build is working properly, it can be deployed throughout the fleet.

8.9 New ACMS with a new manufacturer on an existing aircraft type

In this scenario, the ACMS manufacturer is going to do the software integration to this type of aircraft, and a tuning exists for this aircraft type. Steps include:

1. ACMS manufacturer receives software and documentation.
2. ACMS manufacturer reviews documentation.
3. ACMS manufacturer integrates EDR software to the ACMS for this type of aircraft
4. NCAR works with the ACMS manufacturer to
   a. Ensure that the sources and labels for data inputs match previous implementation
      of this aircraft type
   b. Provide tuning for this aircraft type.
   c. Assist in testing
5. Airline tests ACMS new build with EDR software included on a small number of aircraft
6. Once the build is working properly, it can be deployed throughout the fleet.

### 8.10 New ACMS with a new manufacturer on a new aircraft type

In this scenario, the ACMS manufacturer is going to do the software integration to this type
of aircraft, and a tuning does not exist for this aircraft type. Steps include:

1. ACMS manufacturer receives the EDR Support software, documentation and EDR
   Algorithm software and documentation.
2. ACMS manufacturer reviews documentation.
3. Airline records high-rate input data for a number of flights (~20-30). Note: QAR data
   has not been sufficient. This step has required an airline to install a card onto the ACMS
   for storage and a special report is generated and recorded to the card.
4. The data is processed through the EDR Support software to test the quality of the data
   and produce a tuning. It may be required to iterate on steps 3 and 4 if data quality
   problems are identified. This work can be done by the airline or by NCAR.
5. ACMS manufacturer integrates EDR software to the ACMS for this type of aircraft
6. NCAR works with the ACMS manufacturer to
   a. Ensure that the sources and labels for data inputs match previous implementation
      of this aircraft type
   b. Assist in testing
7. Airline tests ACMS new build with EDR software included on a small number of aircraft
8. Once the build is working properly, it can be deployed throughout the fleet.

### 8.11 Non-ACMS Solutions

In this scenario, the airline wishes to integrate the software themselves on a box other than
an ACMS. Steps include:

1. Airline receives EDR Algorithm software and documentation.
2. Airline reviews documentation.
3. The data requirements (available fields and sampling frequencies) are verified.
4. [Skip if tuning exists for this aircraft] Airline records high-rate input data for a number of flights (~20-30). Note: QAR data has not been sufficient. This step has required an airline to install a card onto the ACMS for storage and a special report is generated and recorded to the card. Delta Airlines has done this, and can be contacted for more information.
5. [Skip if tuning exists for this aircraft] The data is processed through the EDR Support software to test the quality of the data and produce a tuning. It may be required to iterate on steps 4 and 5 if data quality problems are identified. This work can be done by the airline or by NCAR.
6. Airline integrates EDR software to the target system for this type of aircraft
7. NCAR works with the Airline to
   a. Ensure that the sources and labels for data inputs match previous implementation of this aircraft type, if known
   b. Provide tuning for this aircraft type, if the tuning existed previously.
   c. Assist in testing
8. Airline tests new build with EDR software included on a small number of aircraft
9. Once the build is working properly, it can be deployed throughout the fleet.