Behavior log data QA/QC: false killer whales

Michaela A. Kratofil

2/17/2025

## Overview

This document steps through the QA/QC process of behavior log data from SPLASH satellite tags, using false killer whales as an example. There are a number of things to check to make sure that the data is valid before moving on to formal analysis. These steps were informed from previous work ([Baird et al. (2019)](https://www.cascadiaresearch.org/files/publications/Bairdetal2019_Hatteras.pdf), [Shearer et al. 2019](https://royalsocietypublishing.org/doi/10.1098/rsos.181728)) and discussions with Wildlife Computers and Will Cioffi.

The first step is to read in the behavior logs for all tags you’re working with. I created a function to read in files and format them, because some false killer whale tags were programmed in HST, but you could easily do this by reading each file individually, formatting, and then combining them into a single dataframe.

## Overlapping messages

The first thing we are going to do is check the “Message” records for any overlapping messages. Messages contain the behavior logs, or the dive/surface records. A single “Message” row in the -Behavior file summarizes the information for the dives/surfaces following that record, until the next Message block. Sometimes, corrupt messages can be erroneously retained in the behavior log. Usually these are pretty obvious: a message block that extends a really long time, often several days (or weeks) over the next message block. These potentially corrupt messages usually contain dive/surface records with extreme values. The following code creates a function to identify gaps (positive gaps in time from gaps in transmissions) and overlaps (negative gaps) between message blocks. It flags overlapping messages over 60 seconds (within a couple of mins or so is fine), so that you can then review those cases in detail to decide whether the data should be removed or not. If any messages should be removed, then you remove all dives/surfaces within that message (not just the ones with extreme values).

## [1] "PcTag026 0 flagged message overlap"  
## [1] "PcTag028 3 flagged message overlap"  
## [1] "PcTag028, overlap #1: -84 seconds"  
## [1] "PcTag028, overlap #2: -66 seconds"  
## [1] "PcTag028, overlap #3: -68 seconds"  
## [1] "PcTag028: overlap from 2010-10-27 02:48:00 to 2010-10-27 04:27:24"  
## [1] "maximum depth during this block: 331.5"  
## [1] "maximum duration during this block: 1887"  
## [1] "PcTag028: overlap from 2010-10-27 20:15:00 to 2010-10-28 01:13:06"  
## [1] "maximum depth during this block: 64.5"  
## [1] "maximum duration during this block: 7423"  
## [1] "PcTag028: overlap from 2010-11-22 04:19:00 to 2010-11-22 07:17:08"  
## [1] "maximum depth during this block: 125.5"  
## [1] "maximum duration during this block: 5231"  
## [1] "PcTag030 7 flagged message overlap"  
## [1] "PcTag030, overlap #1: -69 seconds"  
## [1] "PcTag030, overlap #2: -113 seconds"  
## [1] "PcTag030, overlap #3: -78 seconds"  
## [1] "PcTag030, overlap #4: -129 seconds"  
## [1] "PcTag030, overlap #5: -72 seconds"  
## [1] "PcTag030, overlap #6: -85 seconds"  
## [1] "PcTag030, overlap #7: -257 seconds"  
## [1] "PcTag030: overlap from 2010-12-18 13:35:00 to 2010-12-18 17:01:09"  
## [1] "maximum depth during this block: 283"  
## [1] "maximum duration during this block: 5871"  
## [1] "PcTag030: overlap from 2010-12-22 10:11:00 to 2010-12-22 18:54:53"  
## [1] "maximum depth during this block: 467"  
## [1] "maximum duration during this block: 17663"  
## [1] "PcTag030: overlap from 2010-12-24 12:04:00 to 2010-12-24 17:23:18"  
## [1] "maximum depth during this block: 189"  
## [1] "maximum duration during this block: 7103"  
## [1] "PcTag030: overlap from 2010-12-26 04:54:00 to 2010-12-26 07:04:09"  
## [1] "maximum depth during this block: 62"  
## [1] "maximum duration during this block: 2511"  
## [1] "PcTag030: overlap from 2010-12-28 09:47:00 to 2010-12-28 13:33:12"  
## [1] "maximum depth during this block: 347"  
## [1] "maximum duration during this block: 8791"  
## [1] "PcTag030: overlap from 2010-12-31 05:25:00 to 2010-12-31 13:05:25"  
## [1] "maximum depth during this block: 185"  
## [1] "maximum duration during this block: 23167"  
## [1] "PcTag030: overlap from 2010-12-31 15:05:00 to 2010-12-31 17:04:17"  
## [1] "maximum depth during this block: 189"  
## [1] "maximum duration during this block: 4016"  
## [1] "PcTag032 4 flagged message overlap"  
## [1] "PcTag032, overlap #1: -173 seconds"  
## [1] "PcTag032, overlap #2: -110 seconds"  
## [1] "PcTag032, overlap #3: -169 seconds"  
## [1] "PcTag032, overlap #4: -78 seconds"  
## [1] "PcTag032: overlap from 2010-12-28 09:21:00 to 2010-12-28 19:03:53"  
## [1] "maximum depth during this block: 157"  
## [1] "maximum duration during this block: 28451"  
## [1] "PcTag032: overlap from 2010-12-30 17:15:00 to 2010-12-30 22:13:50"  
## [1] "maximum depth during this block: 663"  
## [1] "maximum duration during this block: 6527"  
## [1] "PcTag032: overlap from 2010-12-31 14:56:00 to 2010-12-31 18:24:49"  
## [1] "maximum depth during this block: 443"  
## [1] "maximum duration during this block: 6079"  
## [1] "PcTag032: overlap from 2010-12-31 18:22:00 to 2011-01-01 00:10:18"  
## [1] "maximum depth during this block: 567"  
## [1] "maximum duration during this block: 6463"  
## [1] "PcTag035 0 flagged message overlap"  
## [1] "PcTag037 2 flagged message overlap"  
## [1] "PcTag037, overlap #1: -62 seconds"  
## [1] "PcTag037, overlap #2: -66 seconds"  
## [1] "PcTag037: overlap from 2013-07-31 01:49:00 to 2013-07-31 17:25:02"  
## [1] "maximum depth during this block: 935"  
## [1] "maximum duration during this block: 43770"  
## [1] "PcTag037: overlap from 2013-07-31 22:21:00 to 2013-08-01 12:05:06"  
## [1] "maximum depth during this block: 443"  
## [1] "maximum duration during this block: 27270"  
## [1] "PcTag049 1 flagged message overlap"  
## [1] "PcTag049, overlap #1: -62 seconds"  
## [1] "PcTag049: overlap from 2015-09-14 05:15:00 to 2015-09-14 16:37:02"  
## [1] "maximum depth during this block: 599"  
## [1] "maximum duration during this block: 24870"  
## [1] "PcTag055 0 flagged message overlap"  
## [1] "PcTag074 0 flagged message overlap"  
## [1] "PcTag090 0 flagged message overlap"  
## [1] "PcTag092 0 flagged message overlap"  
## [1] "PcTagP09 0 flagged message overlap"

For this dataset, there are a handful of tags that have message overlaps > 60 seconds. After a more detailed inspection into these situations, they are all not extreme overlaps (within ~2 minutes) and the data in the message blocks are reasonable. Therefore, because the duration data are accurate and data seem normal, we will retain all message blocks.

Example code is provided above (commented out) if you do have more obvious corrupt/erroneous message blocks and need to remove them from the dataset.

## Cleaning: pre-deployment and post-last status messages

Now we will address any records that occurred before the deployment datetime and remove any records after the last good (CRC-checked) status message. Records that occurred before the deployment datetime should be surface records, and where applicable, we will replace the start time of the surface record with the deployment datetime. If this step applies, then the code will also correct the duration min and max estimates. We remove data after the last good status message to be conservative in only analyzing data that is likely free of pressure transducer failures.

## [1] "PcTag026 416 original"  
## [1] "PcTag026 398 within-status"  
## [1] "PcTag028 1710 original"  
## [1] "PcTag030 748 original"  
## [1] "PcTag030 739 within-status"  
## [1] "PcTag032 497 original"  
## [1] "PcTag032 481 within-status"  
## [1] "PcTag035 110 original"  
## [1] "PcTag037 258 original"  
## [1] "PcTag049 559 original"  
## [1] "PcTag055 787 original"  
## [1] "PcTag074 189 original"  
## [1] "PcTag074 189 start time corrected"  
## [1] "PcTag090 95 original"  
## [1] "PcTag090 95 start time corrected"  
## [1] "PcTag090 94 within-status"  
## [1] "PcTag092 75 original"  
## [1] "PcTag092 75 start time corrected"  
## [1] "PcTag092 67 within-status"  
## [1] "PcTagP09 217 original"

## Checking for tag transducer failures

Assessment for tag pressure transducer failures here follows what was reported in [Baird et al. (2019)](https://www.cascadiaresearch.org/files/publications/Bairdetal2019_Hatteras.pdf).

For all deployments, -Status.csv files were assessed for indication of pressure transducer failure. This was done by checking values recorded in the ‘Depth’ column, which represents the value immediately recorded before location transmission. As stated in [Baird et al. (2019)](https://www.cascadiaresearch.org/files/publications/Bairdetal2019_Hatteras.pdf), values exceeding +/- 10 indicate potential transducer failure. The ZeroDepthOffset value was also checked and this should be within +/- 9 m. NOTE: we only assess CRC-checked status values; any questionable values that were NOT CRC-checked can be ignored (per Greg Schorr).

### Maximum depth values recorded in -Status files - ONLY report CRC-checked values:

| DeployID | PTT | maxDepth | maxZeroDepthOffset |
| --- | --- | --- | --- |
| PcTag026 | 94804 | 4.0 | -8 |
| PcTag028 | 94805 | 3.5 | -2 |
| PcTag030 | 23747 | 1.0 | 0 |
| PcTag032 | 94788 | NA | NA |
| PcTag035 | 102474 | 1.0 | 2 |
| PcTag037 | 98359 | 3.0 | -2 |
| PcTag049 | 144031 | NA | NA |
| PcTag055 | 168684 | 0.0 | 0 |
| PcTag074 | 222020 | 3.0 | 0 |
| PcTag090 | 252367 | 1.0 | 0 |
| PcTag092 | 253681 | 1.0 | 1 |
| PcTagP09 | 241403 | -1.0 | 2 |

All tag files meet acceptable levels of depth and zero-depth-max values in their status files, so no need to remove any records based on status file readings.

## Ascent/descent rates

Calculate estimated dive ascent/descent rates following [Baird et al. (2019)](https://www.cascadiaresearch.org/files/publications/Bairdetal2019_Hatteras.pdf). Dive ascent/descent rates were calculated as: 2\*Dive depth (avg., meters) / Dive duration (seconds). While this is a coarse estimation of ascent/descent rates, extreme values could indicate the presence of some tag malfunctioning. The highest ascent/descent rate measured from a time-depth-recorder on a false killer whale was 5.8 meters per second, and thus we use this to gauge reasonable versus extreme values.

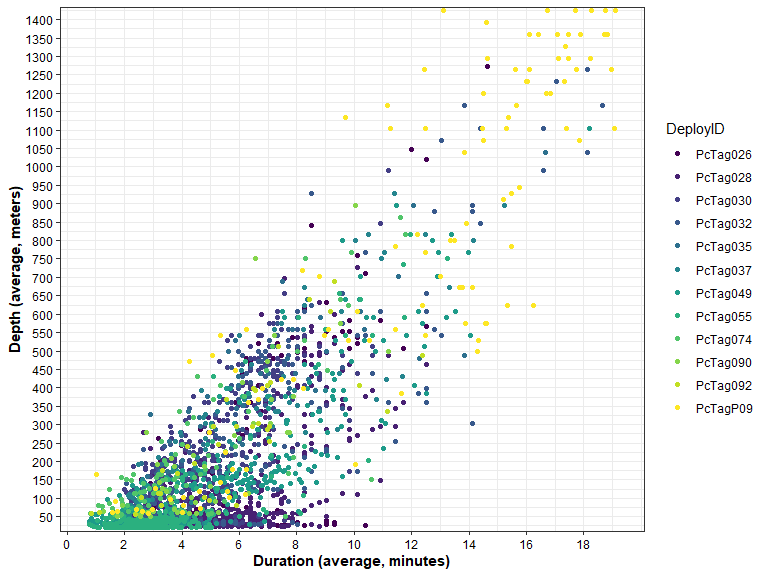
| DeployID | n\_Dives | avg\_depth | min\_depth | max\_depth | avg\_dur | min\_dur | max\_dur | avg\_rate | min\_rate | max\_rate |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PcTag026 | 202 | 151.6 | 24.8 | 1271.8 | 5.0 | 2.0 | 14.7 | 0.80 | 0.09 | 3.29 |
| PcTag028 | 860 | 80.4 | 24.8 | 727.8 | 4.8 | 2.0 | 12.5 | 0.50 | 0.09 | 3.52 |
| PcTag030 | 371 | 211.7 | 30.0 | 991.5 | 4.7 | 2.0 | 11.2 | 1.37 | 0.16 | 3.17 |
| PcTag032 | 238 | 269.2 | 30.0 | 1263.5 | 6.1 | 2.0 | 18.6 | 1.26 | 0.12 | 3.63 |
| PcTag035 | 55 | 357.0 | 36.0 | 1039.5 | 6.1 | 2.1 | 16.7 | 1.89 | 0.24 | 3.76 |
| PcTag037 | 128 | 232.4 | 30.0 | 927.5 | 5.1 | 2.0 | 15.2 | 1.32 | 0.22 | 3.02 |
| PcTag049 | 279 | 202.5 | 30.0 | 1103.5 | 5.8 | 0.8 | 18.2 | 1.08 | 0.16 | 2.78 |
| PcTag055 | 394 | 57.0 | 20.0 | 815.5 | 3.3 | 0.8 | 13.2 | 0.48 | 0.13 | 2.82 |
| PcTag074 | 94 | 205.1 | 51.5 | 863.5 | 4.2 | 0.8 | 11.8 | 1.58 | 0.23 | 3.17 |
| PcTag090 | 47 | 211.9 | 51.5 | 895.5 | 4.8 | 1.3 | 11.2 | 1.33 | 0.39 | 3.81 |
| PcTag092 | 34 | 281.6 | 55.5 | 687.5 | 5.9 | 0.8 | 12.4 | 1.57 | 0.45 | 2.53 |
| PcTagP09 | 110 | 755.3 | 59.5 | 1423.5 | 11.6 | 1.0 | 19.1 | 2.04 | 0.51 | 5.27 |

All tags have ascent/descent rates within a reasonable range, and there don’t appear to be any extreme depth or duration values.

## Summary plots

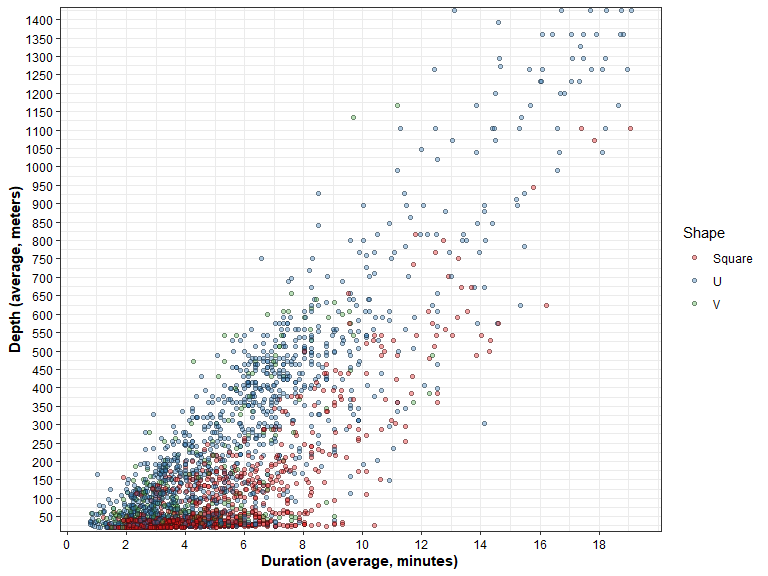
Make simple plots of dive depth, duration, and ascent/descent rates to identify any other outliers in the data.

### Dive depth vs duration

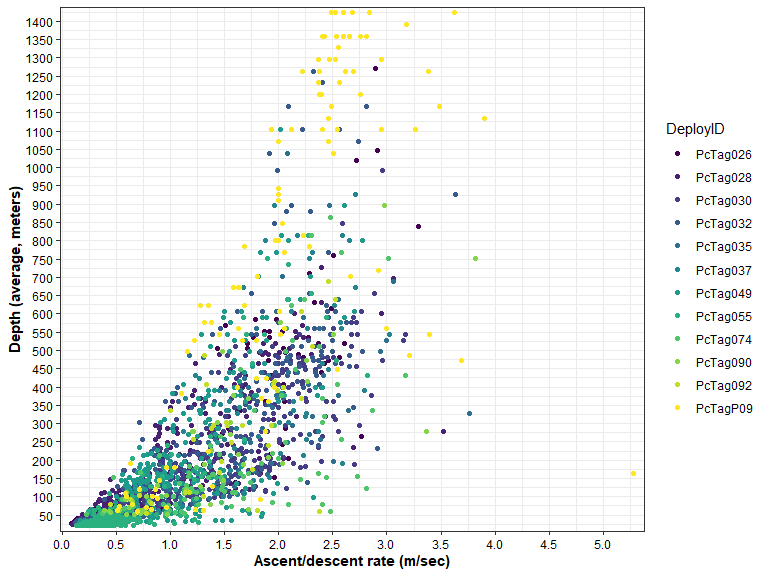


We can see that dive duration generally increases with dive depth, and there are no clear outliers that we should investigate.

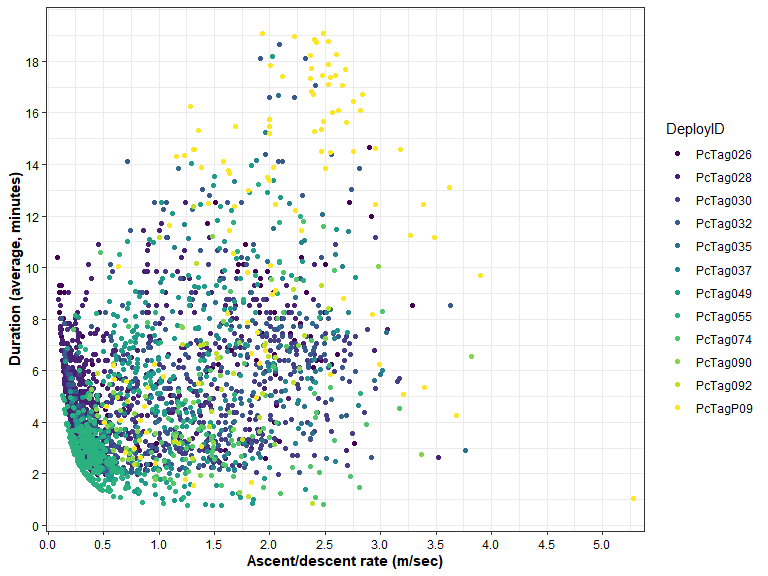
### Dive shape vs dive duration



### Dive depth vs ascent/descent rates



### Dive duration vs ascent/descent rates



## Review time series data

Some tags were programmed to transmit time series data, which can be found in the -Series and -SeriesRange files in applicable tag folders. These data report the depth of the animal at the programmed sampling interval (75, 150, 300, 450, 600 seconds). For more recent deployments, time series data is programmed to transmit on a coarser duty cycle as to provide a means for QA/QC of the behavior log data, and not to obtain a dive record over the entire deployment. Because time series data is collected at a finer temporal scale than the behavior log data, we can use it to check for any potential tag malfunctions that may have occurred. Specifically, if there are obvious discrepancies between the two data streams at any given point, this could be a result of some corruption or malfunction, and we can remove the behavior log data during these periods. See the protocol word document for an example on a killer whale deployment from Greg Schorr.

This step relies on plotting functions that are stored in a separate R file, so we will load those functions first. The functions will make plots of small time periods (easier to see) and save those plots to a specific folder for each tag. To save space in this markdown document, I did not include those plots here; some tags may be longer and/or have a lot of time series/behavior log data, and thus there will be a lot of plots. After running the code below, review those plots in detail to determine whether any data should be removed.

NOTE: in this markdown file, I have set eval=F to not run this code, so that it doesn’t run everytime I knit the document (plots are already generated for these tags).

After reviewing the time series vs behavior log plots, it appears the PcTag028 has one suspect time period where the dives reported by the time series datastream are impossible; looking at the behavior log, we don’t see these values, so they are okay to retain. If we were using the time series data in analyses, then we would exclude this portion of the time series data. For context: Looking at the SeriesRange file, this particular time series message was only received once (Count column), and thus could be corrupt. Per Heather Baer (Wildlife Computers): when a message is received more than once, you can be confident there isn’t any corruption, because the chance of the exact same corruption happening twice is slim. Example code for removing this data when applicable is below.

## Identify minor gaps or overlapping dive/surface records within each tag

After removing all outliers, now we can summarize basic information on gaps. The function will calculate gaps between dive/surface records in the behavior log, including positive gaps resulting from gaps in transmissions to satellites, and negative gaps representing overlaps between consecutive message blocks. Overlapping end/start times occurs when the end of one message block happens after the start of the next one; within 60 seconds or so, this is OK and (presumably) related to how the tag encodes the information (e.g., truncating seconds to save space). For general purposes, these overlaps are OK (e.g., Shearer et al. 2019 retained normal overlapping behavior logs).

## Summarize and save final data

| DeployID | BL\_tracking\_days | BL\_data\_days | BL\_data\_coverage | n\_data\_gaps | max\_data\_gap |
| --- | --- | --- | --- | --- | --- |
| PcTag026 | 11 days | 7.01 | 63.7 | 31 | 57229 |
| PcTag028 | 39 days | 15.72 | 40.3 | 124 | 610007 |
| PcTag030 | 30 days | 19.98 | 66.6 | 50 | 116924 |
| PcTag032 | 26 days | 20.12 | 77.4 | 32 | 129522 |
| PcTag035 | 6 days | 6.59 | 109.8 | 5 | 56 |
| PcTag037 | 16 days | 10.17 | 63.6 | 18 | 101536 |
| PcTag049 | 15 days | 14.54 | 96.9 | 25 | 17304 |
| PcTag055 | 9 days | 6.54 | 72.7 | 34 | 56996 |
| PcTag074 | 10 days | 9.19 | 91.9 | 10 | 36080 |
| PcTag090 | 17 days | 16.23 | 95.5 | 9 | 2176 |
| PcTag092 | 9 days | 3.75 | 41.7 | 7 | 160448 |
| PcTagP09 | 13 days | 8.39 | 64.5 | 23 | 68346 |

| DeployID | n\_Dives | min\_depth | avg\_depth | max\_depth | min\_dur | avg\_dur | max\_dur | perc\_time\_surf |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PcTag026 | 202 | 24.8 | 151.6 | 1271.8 | 2.0 | 5.0 | 14.7 | 90.0 |
| PcTag028 | 860 | 24.8 | 80.4 | 727.8 | 2.0 | 4.8 | 12.5 | 81.8 |
| PcTag030 | 371 | 30.0 | 211.7 | 991.5 | 2.0 | 4.7 | 11.2 | 93.9 |
| PcTag032 | 238 | 30.0 | 269.2 | 1263.5 | 2.0 | 6.1 | 18.6 | 94.9 |
| PcTag035 | 55 | 36.0 | 357.0 | 1039.5 | 2.1 | 6.1 | 16.7 | 96.5 |
| PcTag037 | 128 | 30.0 | 232.4 | 927.5 | 2.0 | 5.1 | 15.2 | 95.6 |
| PcTag049 | 279 | 30.0 | 202.5 | 1103.5 | 0.8 | 5.8 | 18.2 | 92.3 |
| PcTag055 | 394 | 20.0 | 57.0 | 815.5 | 0.8 | 3.3 | 13.2 | 86.5 |
| PcTag074 | 94 | 51.5 | 205.1 | 863.5 | 0.8 | 4.2 | 11.8 | 97.0 |
| PcTag090 | 47 | 51.5 | 211.9 | 895.5 | 1.3 | 4.8 | 11.2 | 99.1 |
| PcTag092 | 34 | 55.5 | 281.6 | 687.5 | 0.8 | 5.9 | 12.4 | 96.3 |
| PcTagP09 | 110 | 59.5 | 755.3 | 1423.5 | 1.0 | 11.6 | 19.1 | 89.5 |