**Estimating body density of cetaceans: the manual**

Tomoko Narazaki ([naratomoz@gmail.com](mailto:naratomoz@gmail.com); IGOR Pro, R)

Lucia Martina Martin Lopez ([luciamartinaml@gmail.com](mailto:luciamartinaml@gmail.com); Matlab)

Saana Isojunno ([si66@st-andrews.ac.uk](mailto:si66@st-andrews.ac.uk); R, jags)

12 June 2016

This document describes the sequence of operations needed to estimate body density of whales from tag data using a hydrodynamic glide model. For this analysis, it is needed either IGOR Pro (WaveMetrics, https://www.wavemetrics.com) or Matlab (MathWorks, http://www.mathworks.com/products/matlab/) for tag data processing. Then, R Studio (https://www.rstudio.com) will be used for Bayesian estimation.

Data required for the analysis:

* High-resolution tag data (Dtag, Little Leonardo 3MPD3GT etc.)
* 3-axis acceleration (essential)
* depth (essential)
* speed (preferable)
* 3-axis magnetism (preferable)
* temperature (preferable)
* CTD profile at the field site

***Part 1: Tag data processing using either IGOR Pro or Matlab***

1) Before the analysis, the tag data must be prepared as follows:

* Complete all calibrations (i.e. temperature drift of sensors, convert the recordings into proper units etc)
* Convert into whale-frame
* Calculate pitch, roll and heading (i.e. prh format in Dtag data)

IGOR Pro Users:

* Save “Body density” folder that include all IGOR Pro macro required for body density analysis into “User Procedure” folder in the IGOR Pro Folder.
* To load a txt file, open IGOR Pro, go to: ‘Data’ 🡪 ‘Load waves’ 🡪 ‘Load general text file’.
* Change interval of data to be based on sampling frequency by going to: ‘Data’ 🡪 ‘Change wave scaling’; alter the delta value (e.g. for 5Hz, Δ=0.2 s)
* In this analysis (using IGOR Pro), the waves are named as follows:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| surgeAw, swayAw, heaveAw | Surge, sway & heave accelerations, respectively, at whale-frame |
| surgeMw, swayMw, heaveMw | Surge, sway & heave magnetism, respectively, at whale-frame |
| D | Depth |
| pitch | Pitch in degrees |
| roll | Roll in degrees |
| head | Heading in degrees |

Matlab Users:

* Save “Body density” folder and add it into your Matlab path.
* Use ‘File’ 🡪 ‘Set path’ in the pull-down menu at the top of the Matlab screen and add the “Body density analysis” folder into your path. Press save button.
* Open “body\_density” script and follow the instructions there.
* Load prh.

|  |  |
| --- | --- |
| **Wave names Dtagdata** | **Description** |
| Aw | 3-axis accelerometer data at the whale frame; i.e., Aw(:,1) longitudinal x axis; (:,2) lateral y axis; (:,3) dorso-ventral z axis. |
| Mw | 3-axis magnetometer data at the whale frame; i.e., Mw(:,1) longitudinal x axis; (:,2) lateral y axis; (:,3) dorso-ventral z axis. |
| p | Depth in meters |
| pitch | Pitch in radians |
| roll | Roll in radians |
| head | Heading in radians |

* Type help and the name of the function for a description of input and output of each function; e.g. help finddives

2) Define dives – then, make a summary table describing the characteristics of each dive.

Here, dives are defined as any submergence deeper than 2 m (i.e. dive definition = 2 m). But, in this analysis, only deep dives of which max depth is bigger than 10m (i.e. deep dive definition = 10 m) will be extracted. Change the definitions appropriately according to your data.

IGOR Pro:

* Type **#include “FindDives”** in the procedure window.
* In the macro **FindDives,** set **DiveDef** (dive definition), **D\_DiveDef** (Deep dive definition) and **fs** (sampling frequency of depth data).
* Run the macro by typing **FindDives (D)** into the command window.
* Type **edit Dive** in the command window to see the summary table
* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Dive[][0] | Start time of each dive (i.e. No of seconds since 1904/1/1) |
| Dive[][1] | End time of each dive (i.e. No of seconds since 1904/1/1) |
| Dive[][2] | Dive duration in seconds |
| Dive[][3] | Post-dive surface duration in seconds |
| Dive[][4] | Time of the deepest point of each dive (i.e. No of seconds since 1904/1/1) |
| Dive[][5] | Max dive depth (m) |
| Start | Dive start point |
| Fin | Dive end point |
| DiveNumber | Dive ID. Starting from 1. |
| MaxD | Maximum dive depth (m) |
| Duration | Dive duration (s) |

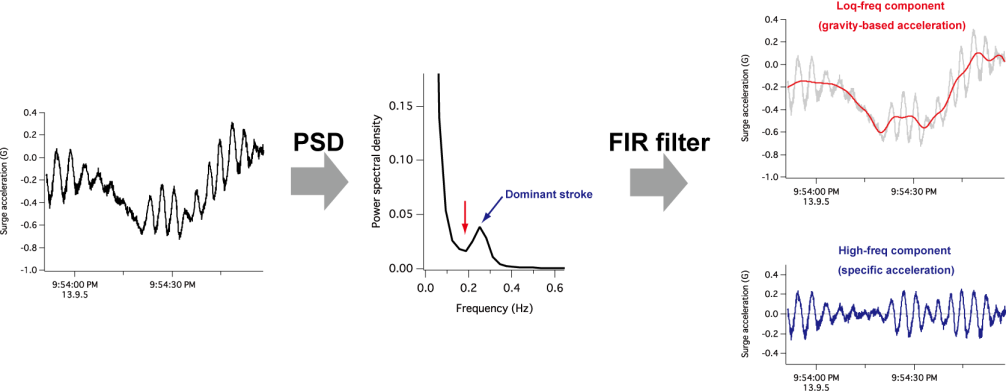
Matlab:

* Described in **body\_density.m** script as 2nd step.
  + Run **T=finddives(p,fs,mindivedep,[],1)**
  + Create matrix **D**

|  |  |
| --- | --- |
| **D matrix** | **Description** |
| D(:,1) | Start time of each dive since tag on time (in seconds) |
| D(:,2) | End time of each dive since tag on time (in seconds) |
| D(:,3) | Dive duration in seconds |
| D(:,4) | Post-dive surface duration in seconds |
| D(:,5) | Time of the deepest point of each dive since tag on time (in seconds) |
| D(:,6) | Max dive depth of each dive (in meters) |
| D(:,7) | Dive ID. Starting from 1. |

3) Separate gravity-based and specific acceleration.

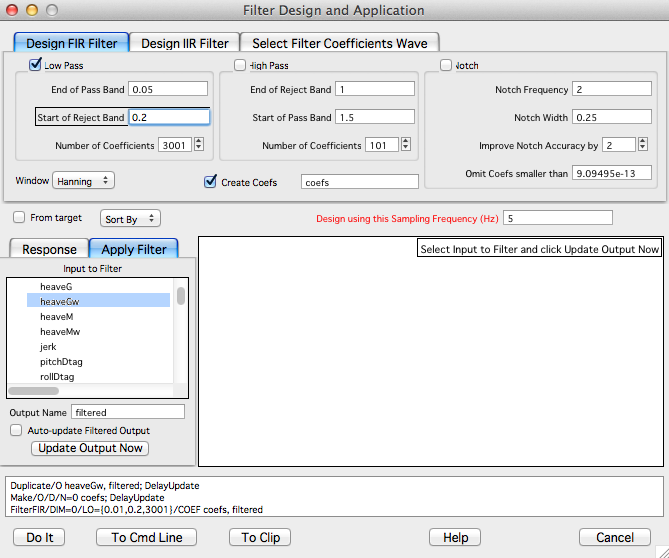
The accelerometers record both gravity-based acceleration (i.e. slowly-changing body orientations) and dynamic acceleration (i.e. resulting from body motions such as body rotations and specific accelerations ). In steady swimming, the mean body posture typically varies slowly relative to the stroking rate, while dynamic acceleration occurs at the stroking rate. Thus, we use a frequency-based filter to separate the body posture (low-frequency) from the dynamic acceleration (high-frequency) data.



To decide the cut-off value of the filter, first the power spectral density (PSD) of the acceleration should be calculated. The positive peak of PSD shows the dominant stroke frequency. Use the frequency at the negative peak (red arrow) as the threshold of the low-pass filter to extract the gravity-based acceleration components.

IGOR Pro:

* To make a PSD plot, type **#include <power spectral density>** in the procedure window and compile.
* Then, ‘Macro’ 🡪 ‘PowerSpectralDensity’
* Once, the cut-off value is decided, perform filter. ‘Analysis’ 🡪 ‘Filter…’
* In the ‘Design FIR Filter” tab, check ‘Low Pass’
* Set the cut-off values by inserting appropriate values to ‘End of Pass Band’ and ‘Start of Reject Band’. Recommended number of coefficients is 1001 – 3001.
* Window should be Hanning
* In the ‘Apply Filter’ tab, select the wave (e.g. heaveAw)
* Specify output name (e.g. heaveAwFilt).
* Click ‘Do It’. The output wave is low-frequency acceleration components.



* Repeat for surgeAw and swayAw
* To obtain specific acceleration, type the followings into the command window:

**Duplicate surgeAw surgeSA swaySA heaveSA**

**surgeSA = surgeAw – surgeAwFilt**

**swaySA = swayAw – swayAwFilt**

**heaveSA = heaveAw – heaveAwFilt**

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| surgeAwFilt, swayAwFilt, heaveAwFilt | Low-freq component (gravity-based) acceleration at surge, sway and heave axis |
| surgeSA, swaySA, heaveSA | High-freq component (dynamic) acceleration at surge, sway and heave axis |

Matlab:

* Described in **body\_density.m** script as 3rd step.
  + 3.1\_Quick separation of descent and ascent phases; index of descent and ascent phases are defined under DES and ASC matrix.
  + 3.2\_Separate low- and high- frequency acceleration signals:
    - 3.2.1\_Select periods of analysis:

(a) whole deployment record.

(b) during descent and ascent phases where steady stroking occurs, the power spectrum of these signals will give a much clearer peak than for the whole record deployment.

* + - 3.2.2\_Determine stroking rate (**FR**) and cut-off frequency using figure (1).
    - 3.2.3\_Calculate low- and high-pass-filtered acceleration signal using the function **[Anlf,Ahf,GL,KK]=Ahf\_Anlf(Aw,fs,FR,f,n,k,J,tmax);**
    - 3.2.4\_Create the low-pass-filtered pitch, named **smoothpitch.**

Output:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Anlf | Normalized low-pass-filtered 3-axis acceleration signal; Anlf(:,1) longitudinal x axis; (:,2) lateral y axis; (:,3) dorso-ventral z axis. |
| Ahf | High-pass-filtered 3-axis acceleration signal; Ahf(:,1) longitudinal x axis; (:,2) lateral y axis; (:,3) dorso-ventral z axis. |

4) Separate each dive into descent, bottom and ascent phases. Here, a descent phase is defined as the period between the beginning of each dive and the time when whale’s pitch first exceeded 0° (i.e. pitch definition). An ascent phase was defined to start at the last point in time when whales’ pitch was < 0° and ended at the end of each dive. Bottom phase is the period of the dive between the descent and the ascent phase.

IGOR Pro:

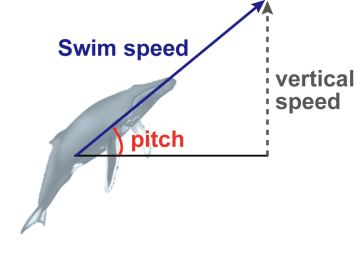
* Type **#include “DivePhase”** in the procedure window.
* In the macro **FindBottom,** set **PitchDef** (pitch definition), **D\_DiveDef** (Deep dive definition) and **fs** (sampling frequency of depth data).
* Run the macro by typing **FindBottom (D, pitch, Dive)** into the command window.
* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Phase | Coded by each phase (-1 for descent, 0 for bottom, 1 for ascent, NAN for not during dive) |
| Bottom[][0] | Time at bottom phase start (i.e. No of seconds since 1904/1/1) |
| Bottom[][1] | Depth at bottom phase start |
| Bottom[][2] | Time at bottom phase end (i.e. No of seconds since 1904/1/1) |
| Bottom[][3] | Depth at bottom phase end |

Matlab:

* Described in **body\_density.m** script as 4th step.
* It creates a graph showing not diving periods in red and diving periods in green. Click on the last dive you want to analyze; sometimes the tag detaches during the ascent of the last dive and that part of the tag excursion from the bottom to the surface should not be incorporated in the analysis. Descent phases are colored in blue, ascent phases in black and bottom phases in green.
* Check that the definition of the phases makes sense. In the case that a phase is shorter than expected due to a small ascent period during the descent phase or a small descent period during the ascent phase,the end and start of the descent and ascent phases can be selected manually respectively by running the 7 lines that are commented (%) within the loop.
* Output would be

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Phase | Coded by each phase (-1 for descent, 0 for bottom, 1 for ascent, NAN for non dive phase) |
| Bottom(:,1) | Time at the start of the bottom phase since tag on time (in seconds) |
| Bottom(:,2) | Depth at the start of the bottom phase (in meters) |
| Bottom(:,3) | Time at the end of the bottom phase since tag on time (in seconds) |
| Bottom(:,4) | Depth at the end of the bottom phase (in meters) |



5) Estimate swim speed.

Here, swim speed is calculated as vertical speed divided by sine of pitch only when pitch angle is sufficiently steep (default is 30 degrees). If speed sensor data (i.e. propeller, acoustic etc.) is available, the calibration to convert the data into m/s has to be done.

IGOR Pro:

* Type **#include “EstimateSpeed”** in the procedure window.
* In the macro **EstimateSpeed,** and set **PitchDef** (cut-off value for pitch, default is 30 degree) and **fs** (sampling frequency of depth data).
* Run the macro by typing **EstimateSpeed(pitch, D)** into the command window.
* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| SwimSp | Swim speed in m/s |

Matlab:

* Described in **body\_density.m** script as 5th step.
* Set pitch (degrees) threshold above which speed can be estimated.
* Run function **[SwimSp]=inst\_speed(p,smoothpitch,fs,FR,f,k,thdeg)**
* Output

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| SwimSp | Swim speed in m/s |

6) Estimate seawater density around the tagged whale.

Prepare CTD data that were collected at your field site. Firstly,seawater density from CTD’s temperature, salinity and depth data should be calculated using an equation of state of seawater (UNESCO, 1983) to make a seawater density-depth plot. If the CTD data already have seawater density, the first part of the estimation can be skipped. Then, seawater density at the depth where the animal visits will be extracted from the density-depth profile.

IGOR Pro:

*Step 1: Make a density-depth plot*

* Type **#include “SWdensityFromCTD”** in the procedure window.
* Run the macro by typing **SWdensityFromCTD(DPT, TMP, SL)** into the command window. To run this macro, prepare the following data:
  + DPT – CTD depth data
  + TMP – CTD temperature data
  + SL – CTD salinity data

In addition, set the variable **Deepest** before running the macro. **Deepest** must be set deeper than animal’s maximum dive depth.

* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| SWdensity | Seawater density calculated from CTD’s depth, temperature and salinity measurements |
| depCTD | Depth data at which SWdensity was recorded |

*Step 2: estimate seawater density around the whale (Dsw)*

Type **#include “EstimateDsw”** in the procedure window.

* Run the macro by typing **EstimateDsw(SWdensity, depCTD, D)** into the command window.
* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Dsw | Seawater density around the tagged animal |

Matlab:

* Described in **body\_density.m** script as 6th step.

*Step 1: Make a density-depth plot*

* Create the following vectors:
  + DPT – CTD depth data
  + TMP – CTD temperature data
  + SL – CTD salinity data
* Run the function **[SWdensity,depCTD]=SWdensityFromCTD(DPT,TMP,SL,D)**
* Output:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| SWdensity | Seawater density calculated from CTD’s depth, temperature and salinity measurements |
| depCTD | Depth data at which SWdensity was recorded |

*Step 2: estimate seawater density around the whale (Dsw)*

* Output

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Dsw | Seawater density around the tagged animal |

7) Extract strokes and glides.

Glides are defined as periods with small fluctuations in heave specific acceleration during dives. To extract glides, firstly, the threshold that determine whether the fluctuation is large (i.e. stroke) or not (i.e. glide) should be determined.

In the left figure, green circles show all positive and negative peaks in heave specific acceleration. A boundary between small noise signals and strokes (right figure) is expected to be seen in the histogram of the absolute amplitude of positive and negative peaks. The boundary is used as the threshold and any fluctuations larger than the threshold are considered as stroke (red dots in the left figure).

IGOR Pro:

*Step 1: Detect strokes*

* Type **#include “ExtractStroke”** in the procedure window.
* In the macro **ExtractStroke\_1,** specify the appropriate values for **fs** (sampling frequency of acceleration data) and **s\_dur** (minimum stroke duration in seconds).
* Run the macro **ExtractStroke\_1(heaveSA)** to find out the threshold
* Examine the histogram and select the threshold.
* Open the macro **ExtractStroke\_2(stroke**) and change the **stroke\_def** (threshold of strokes).
* Run the macro **ExtractStroke\_2(stroke)** to extract strokes.
* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| stroke | Indicates positive and negative peaks of each stroke |

*Step 2: classify ‘strokes’ or ‘glides’*

* Type **#include “StrokeOrGlide”** in the procedure window.
* Open the macro **StrokeOrGlide (stroke**) and specify the appropriate values for **interval** (the macro will check if strokes occur at this interval) and **fs** (sampling
* Run the macro **StrokeOrGlide (stroke**).
* After running these macros, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| SGtype | Indicates whether it is stroking (SGtype = 1) or gliding (SGtype = 0) |

Matlab:

* Described in **body\_density.m** script as 7th step.
* 7.1\_Extract strokes and glides using the high-pass filtered acceleration signal, using function **Ahf\_Anlf.m**
* 7.2\_Extract strokes and glides using the body rotations obtained with the magnetometer method, using function **magnet\_rot\_sa.m or magnet\_rot\_sa\_AB.m.** The differences between these two functions are explained in the comments and relate to how ambiguous one-way fluking signals are treated.

Output:

* **GL** = matrix containing the start time (first column) and end time (2nd column) of any glides (i.e., no zero crossings in tmax or more seconds).Times are in seconds.
* **KK** = matrix of cues to zero crossings in seconds (1st column) and zero-crossing directions (2nd column). +1 means a positive-going zero-crossing. Times are in seconds.

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| SGtype | Indicates whether it is stroking (SGtype = 1) or gliding (SGtype = 0) |

8) Make 5-s sub-glides.

Glides are segmented into 5-s sub-glides. Then, make a summary table describing key characteristics of each 5-s sub-glides (e.g. mean depth, pitch, speed, acceleration etc.) that is required for body density estimation. If necessary, add more information (e.g. if the sub-glides are associated with lunge feeding, presence/absence of sonar exposure etc.).

Save the summary table for glide characteristics in csv format (*whaleID*.csv) for the further analysis in R. The summary table must be prepared for each whale (or each deployment).

IGOR Pro:

* Type **#include “GlideCharacteristics”** in the procedure window.
* In the macro **SubGlides\_summary,** set the following variables before running the macro: **sub\_dur** (duration of sub-glides; default is 5 seconds), **Ddef** (any sub-glides shallower than Ddef will be removed) and **fs** (sampling frequency of depth data).
* Run the macro by typing **SubGlides\_summary (SGtype, D, SwimSp, Temp, Dive, DiveNumber, Start, Fin, Dsw, pitch, roll, head)** into the command window.

Note that the sampling frequency of **SGtype, D, SwimSp, Temp, Dsw, pitch, roll, head** must be the same to run this macro.

* Type **edit Glide** in the command window to see the summary table
* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Glide[][0] | Sub-glide start point |
| Glide[][1] | Sub-glide end point |
| Glide[][2] | Sub-glide duration (s) |
| Glide[][3] | Mean depth (m) during the sub-glide |
| Glide[][4] | Total depth change (m) during the sub-glide |
| Glide[][5] | Mean swim speed during the sub-glide |
| Glide[][6] | Mean pitch (degree) during the sub-glide |
| Glide[][7] | Sine of pitch |
| Glide[][8] | SD of pitch during the sub-glide |
| Glide[][9] | Mean temperature |
| Glide[][10] | Mean seawater density (kg/m3) during the sub-glide |
| Glide[][11] | Acceleration during the sub-glide (m/s2) |
| Glide[][12] | R-value for the regression swim speed vs time during the sub-glide |
| Glide[][13] | SE of the gradient for the regression swim speed vs time during the sub-glide |
| Glide[][14] | Dive phase: 0 for bottom phase, -1 for descent phase, 1 for ascent phase |
| Glide[][15] | Dive number in which the sub-glide recorded |
| Glide[][16] | Maximum dive depth (m) of the dive |
| Glide[][17] | Dive duration (s) of the dive |
| Glide[][18] | Mean pitch (deg) calculated using circular statistics |
| Glide[][19] | Measure of concentration (r) of pitch during the sub-glide (i.e. 0 for random direction, 1 for unidirectional) |
| Glide[][20] | Mean roll (deg) calculated using circular statistics |
| Glide[][21] | Measure of concentration (r) of roll during the sub-glide |
| Glide[][22] | Mean heading (deg) calculated using circular statistics |
| Glide[][23] | Measure of concentration (r) of head during the sub-glide |

* Save the summary table (Glide) in csv format. Name the file as “*whaleID*.csv”.

Matlab:

* Described in **body\_density.m** script as 8th step.
* 8.1\_Make 5 sec sub-glides
  + Specify the minimum duration in seconds of sub-glides; dur=5;
  + Look at figure to check that it make sense.
* 8.2\_Create a glide summary table
* Output:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| Glide(:,1) | Sub-glide start point since tag on time (in seconds) |
| Glide(:,2) | Sub-glide end point since tag on time (in seconds) |
| Glide(:,3) | Sub-glide duration (in seconds) |
| Glide(:,4) | Mean depth during the sub-glide (in meters) |
| Glide(:,5) | Total depth change during the sub-glide (in meters) |
| Glide(:,6) | Mean swim speed during the sub-glide (in m/s) |
| Glide(:,7) | Mean pitch during the sub-glide (in degrees) |
| Glide(:,8) | Mean sine of pitch during the sub-glide (in degrees) |
| Glide(:,9) | SD of pitch during the sub-glide (in degrees) |
| Glide(:,10) | Mean temperature (in °C) |
| Glide(:,11) | Mean seawater density during the sub-glide (in kg/m3) |
| Glide(:,12) | Acceleration during the sub-glide (in m/s2) |
| Glide(:,13) | R2-value for the regression swim speed vs time during the sub-glide |
| Glide(:,14) | SE of the gradient for the regression swim speed vs time during the sub-glide |
| Glide(:,15) | Dive phase: 0 for bottom phase, -1 for descent phase, 1 for ascent phase |
| Glide(:,16) | Dive number in which the sub-glide recorded |
| Glide(:,17) | Maximum dive depth of the dive (in meters) |
| Glide(:,18) | Dive duration of the dive (in seconds) |
| Glide(:,19) | Mean pitch calculated using circular statistics (in degrees) |
| Glide(:,20) | Measure of concentration (r) of pitch during the sub-glide (i.e. 0 for random direction, 1 for unidirectional) |
| Glide(:,21) | Mean roll calculated using circular statistics (in degrees) |
| Glide(:,22) | Measure of concentration (r) of roll during the sub-glide |
| Glide(:,23) | Mean heading calculated using circular statistics (in degrees) |
| Glide(:,24) | Measure of concentration (r) of head during the sub-glide |

* Save the summary table (Glide) in csv format. Type: **csvwrite(‘WhaleID.csv’,Glide)**.

9) Calculate glide ratio during descent and ascent phases. This analysis is not required for body density estimation. However, it can be useful for validation as denser whales are expected to glide more during descent (& more buoyant whales are expected to glide less during descent).

IGOR Pro:

* Type **#include “GetGlideRatio”** in the procedure window.
* Set the following variables before running the macro: **fs** (sampling frequency of D and SGtype) and **DeepDef** (threshold – get glide ratio deeper than this depth only).
* Run the macro by typing **GetGlideRatio (Dive, Bottom, SGtype, pitch, D)** into the command window.
* After running the macro, the following waves will be created:

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| G\_ratio[][0] | Total duration during descent phase (in seconds) |
| G\_ratio[][1] | Glide duration during descent (in seconds) |
| G\_ratio[][2] | Glide ratio during descent (ranges 0 – 1) |
| G\_ratio[][3] | Mean pitch during descent (in degrees) |
| G\_ratio[][4] | Descent rate (in m/s) |
| G\_ratio[][5] | Total duration during ascent phase (in seconds) |
| G\_ratio[][6] | Glide duration during ascent (in seconds) |
| G\_ratio[][7] | Glide ratio during ascent (ranges 0 – 1) |
| G\_ratio[][8] | Mean pitch during ascent (in degrees) |
| G\_ratio[][9] | Ascent rate (in m/s) |

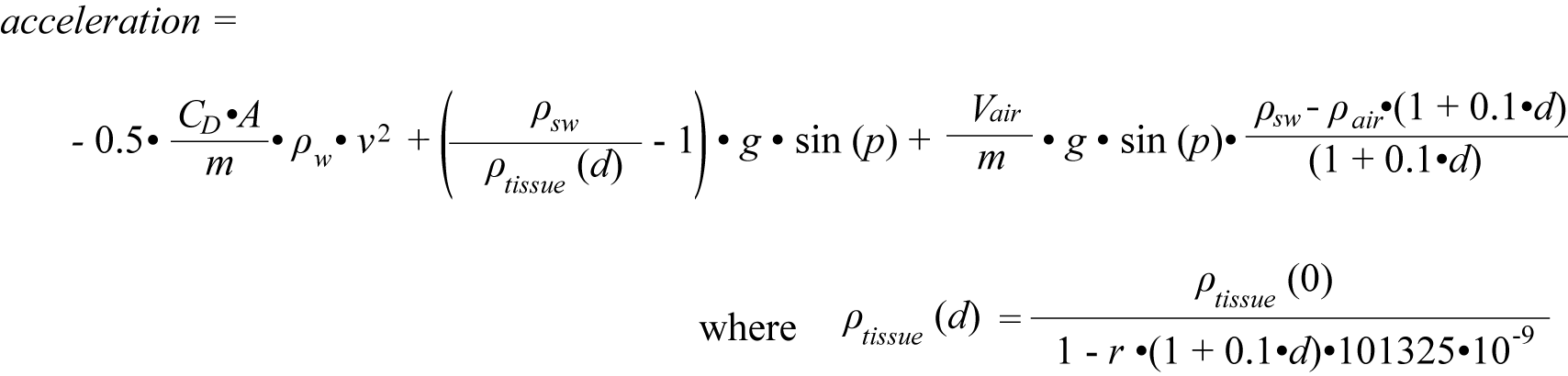
Matlab:

* Described in **body\_density.m** script as 9th step.
* Output: for each dive

|  |  |
| --- | --- |
| **Wave names** | **Description** |
| G\_ratio(:,1) | Total duration of the descent phase (in seconds) |
| G\_ratio(:,2) | Glide duration during descent (in seconds) |
| G\_ratio(:,3) | Glide ratio during descent (ranges 0 – 1) |
| G\_ratio(:,4) | Mean pitch during descent (in degrees) |
| G\_ratio(:,5) | Descent rate (in m/s) |
| G\_ratio(:,6) | Total duration of the ascent phase (in seconds) |
| G\_ratio(:,7) | Glide duration during ascent (in seconds) |
| G\_ratio(:,8) | Glide ratio during ascent (ranges 0 – 1) |
| G\_ratio(:,9) | Mean pitch during ascent (in degrees) |
| G\_ratio(:,10) | Ascent rate (in m/s) |

***Part 2: Estimating unknown parameters using R***

Based on hydrodynamic glide model, acceleration during sub-glides can be expressed as below.



*CD*: drag coefficient *ρsw*: density of the surrounding seawater (kg m-3)

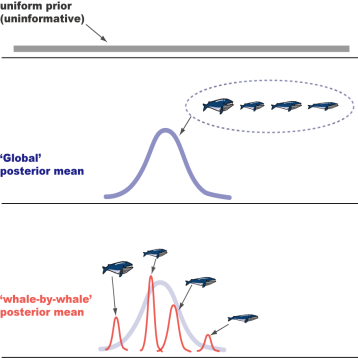
*A*: relevant surface area (m2) *ρtissue*: density of the non-gas component of the whale body (kg m-3)

*m*: mass of the whale (kg) *Vair*: volume of air inspired at the surface (ml)

*g*: gravity *r*: compressibility for animal tissue

*p*: animal pitch (radians) *ρair*: density of air (kg m-3)

*d*: glide depth (m)

Here, four unknown parameters in the equations (drag term CDA/m, body density ρtissue (0), diving gas volume Vair/m, tissue compressibility r) are estimated using Bayesian Gibbs sampling in freely available software JAGS within R.

In this analysis, a total of 12 models were evaluated to explore variability in body density, the drag term and diving gas volume. We use hierarchical model structure in which each individual (body density, drag term) or dive-by-dive (diving gas volume) estimate was considered to be a sample from a global or ‘population’ distribution with an estimated global mean, and an estimated variance across the dives and individuals. After running all the models, best model is selected based on the deviance information criterion (DIC).

The following R codes are originally created by Dr. Saana Isojunno (University of St Andrews) and were slightly modified by Tomoko Narazaki.

10) Install JAGS and R2jags

Both JAGS and R2jags are required in this analysis. JAGS (Just Another Gibbs Sampler) is a tool for analysis of Bayesian hierarchical models using Markov Chain Monte Carlo (MCMC) simulation. See <http://mcmc-jags.sourceforge.net> for the details. R2jags is an R package used to call JAGS from R (See <https://cran.r-project.org/web/packages/R2jags/R2jags.pdf> for the details).

* JAGS
  + Download files at <https://sourceforge.net/projects/mcmc-jags/files/JAGS/>.
  + Follow the instructions to install.
* R2jags
  + Open R (or R-studio)
  + Type **install.packages("R2jags")** and hit enter. To check if it is properly installed, type **library(R2jags).**
* rjags
  + You might need to install rjags to install R2jags. In this case, type **install.packages(“rjags”)** to install.

11) Read glide summary tables in R.

* Open **make\_all\_whales\_data.R**.
* Set work directory
* **dataDir** 🡪 where the glide summary tables are stored
* **modelDir** 🡪where the model files etc. are stored
* **whales** 🡪 List of whaleID. Should be the same as the names in your glide summary tables (*whaleID*.csv)
* Run the codes to make **all\_whales.Rd** that includes all glide data.

12) Set priors.

To estimate four unknown parameters using Bayesian framework, we need to set a specific prior range for each parameter. Here, we use a uniform (non-informative) prior for compressibility, body density and diving gas volume. For example, prior range used for Northern bottlenose whales (*Hyperoodon ampullatus*) were:

**compr ~ dunif (0.3, 0.7) # x 10-9 Pa-1**

**Vair ~ dunif (5, 50) # ml kg-3**

**body.density.g ~ dunif(800, 1200) # kg m-3**

For the combined drag term (CDA/m), we used an informative prior that was estimated from previous research. In the case of Northern bottlenose whales, for example, body length ranges from 5.8 to 9.8 m, surface area (mean 23.0 m2, range 12-36 m2) and mass (mean: 6816 kg; range: 3027-12739 kg) were estimated using the equation derived for sperm whales (Miller et al., 2004). Drag coefficient was estimated to be roughly 0.0030 based on previous research of some cetaceans. Using these values, the prior is set to be a normal distribution with mean of 10.0x10-6 m2 kg-1 and standard deviation of 2.0x10-6 m2 kg-1.

**Table:** CD values reported in previous research

|  |  |
| --- | --- |
| **Species** | **CD** |
| Killer whale | 0.0029 (Fish 1998) |
| Fin whale | 0.0026 (Bose and Lien, 1989) |
| Sperm whale | 0.0031 (Miller et al. 2004) |

**CdAM.g ~ dnorm(10, 0.25)T(5,20) # x10-6 m2 kg-1, dnorm(mean, precision)**

**# so, if SD = 2, precision = 1/22**

Once you decide the prior range for each of four unknown priors, set prior range of the models by following steps:

* Open text file named **model(*number*).txt**.
* For each text file, change the prior range appropriately and save.

13) Set initial values.

* Open **make\_inits.R**
* Initial values must be within the prior range. Set the initial values according to your prior range.
* Run the codes to make **sp.inits.**

14) Estimate models.

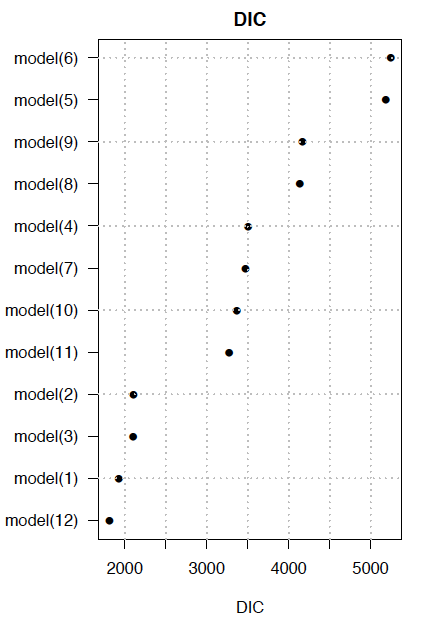
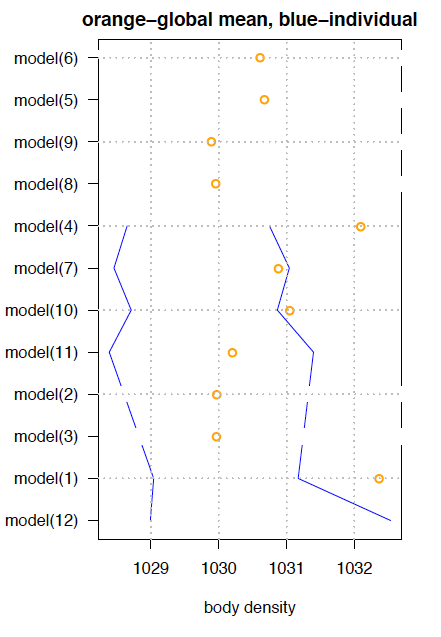
* Open **estimate\_all\_models.R**
* Make sure **setwd, dataDir, modelDir** are OK.
* Set **filterBool** 🡪 criteria for selecting ‘good’ glides for analysis.

Default is mean pitch > 30 degree, maximum dive depth > 100 m, variation of roll > 0.9, during descent and ascent only, no missing data in acceleration, swim speed, seawater density. Change the settings depending on your data.

* Set **fitName** 🡪 you can change this according to your filterBool criteria.
* Run the codes to estimate models
* Result of each model is stored as a Rd file named **model(*number*)\_*fitName*.Rd** in the work directory.

15) Model selection

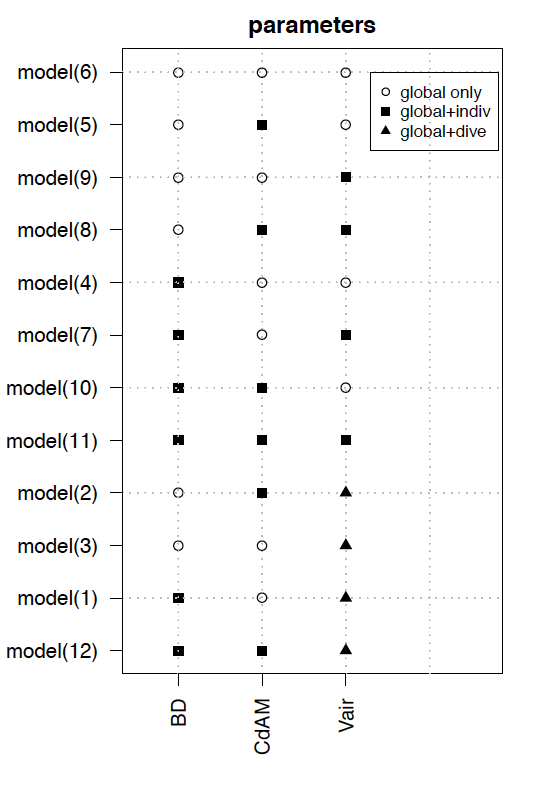
* Once estimating all the models, open **compare\_models\_DIC.R**
* Make sure **setwd, dataDir, modelDir** are OK.
* Make sure **fitName** and **filterBool** are correct
* Run the code and you will get the following files in your work directory.
  + **compare\_models.pdf** 
    - Deviance, DIC etc. of each model
    - Parameter estimates of each model (to check if estimated value change across different models)

**○** global mean estimate

**—** whale-by-whale estimates

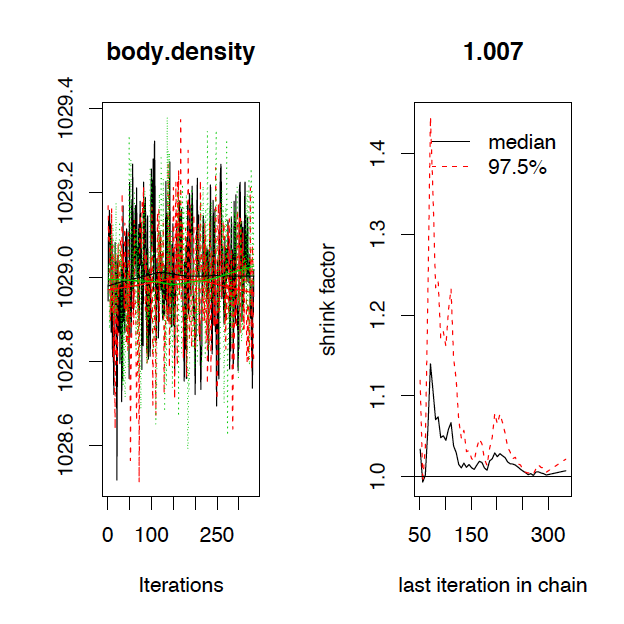
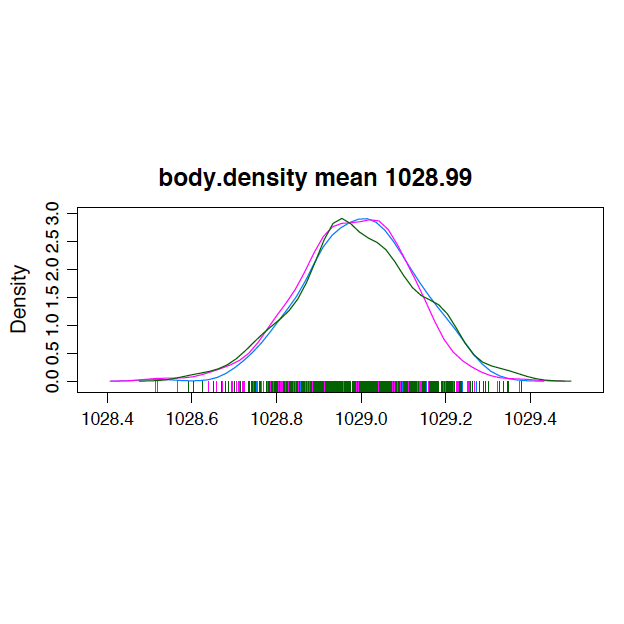
* + **all\_model\_estimates.csv**
    - a summary table of DIC, deviance and parameter estimates for all models
  + model\_parameters.pdf
    - shows combination of global, individual-specific, dive-specific parameters used in each model.



* Select the best model based on DIC.

16) Check the selected model.

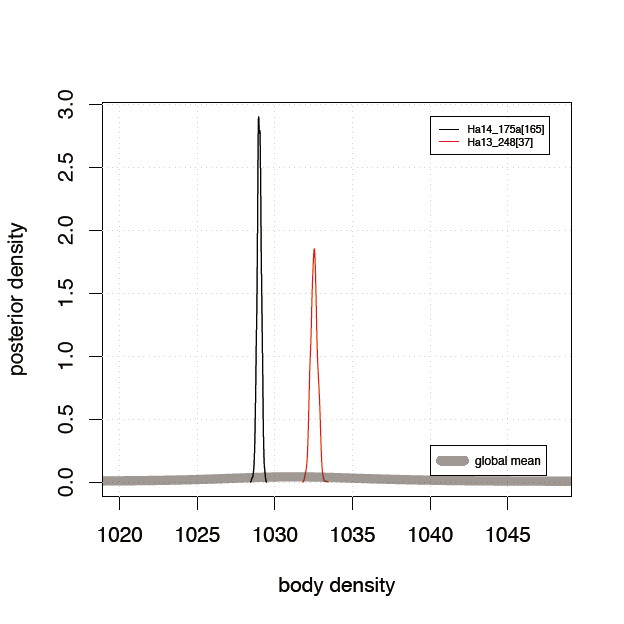
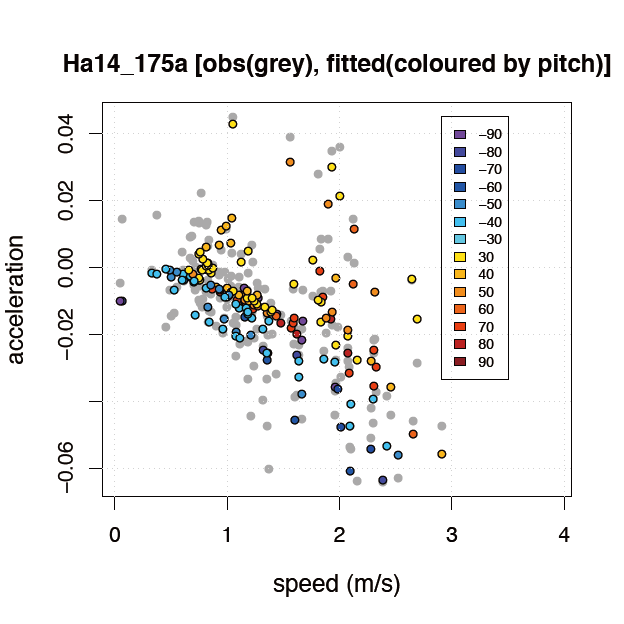
* Open **best\_model\_tracehistories.R**
* Make sure **setwd, dataDir, modelDir** are OK.
* Make sure **fitName** and **filterBool** are correct.
* Set **modelName** to the best model (e.g. **modelName<-c(“model(12)”)**)
* Run the codes and you will get **plot\_best\_model(*number*)\_tracehistories.pdf**

* Check the trace histories and posterior density plots to assess the convergence.

17) Make result plots of the selected model.

* Open **best\_model\_plots.R**
* Make sure **setwd, dataDir, modelDir** are OK.
* Make sure **fitName** and **filterBool** are correct.
* Set **modelName** to the best model (e.g. **modelName<-c(“model(12)”)**)
* Run the codes and you will get **plot\_best\_model(*number*).pdf** that include posterior density of each parameters, observed and estimated acceleration vs swim speed etc.

18) Make a summary table for the selected model.

* Open **Make\_summary\_table.R**
* Make sure **setwd, dataDir, modelDir** are OK.
* Make sure **fitName** and **filterBool** are correct.
* Set **modelName** to the best model (e.g. **modelName<-c(“model(12)”)**)
* Run the codes and you will get the following csv files:
  + **model(*number*)\_*fitName*\_Estimates\_global.pdf**

|  |  |
| --- | --- |
| meanBD.g | Mean global body density (kg/m3) |
| L95BD.g | Lower 95% CRI global body density (kg/m3) |
| U95BD.g | Upper 95% CRI global body density (kg/m3) |
| 95rangeBD.g | 95% CRI range of global body density (kg/m3) |
| meanCdAM.g | Mean global Cd\*A/m (x10-6 m2/kg) |
| L95CdAM.g | Lower 95% CRI global Cd\*A/m (x10-6 m2/kg) |
| U95CdAM.g | Upper 95% CRI global Cd\*A/m (x10-6 m2/kg) |
| 95rangeCdAM.g | 95% CRI range of global Cd\*A/m (x10-6 m2/kg) |
| meanVair | Mean global Vair/m (mL/kg3) |
| L95Vair | Lower 95% CRI global Vair/m (mL/kg3) |
| U95Vair | Upper 95% CRI global Vair/m (mL/kg3) |
| 95rangeVair | 95% CRI range of global Vair/m (mL/kg3) |
| meanCompr | Mean global compressibility (x10-9 /Pa) |
| L95Compr | Lower 95% CRI global compressibility (x10-9 /Pa) |
| U95Compr | Upper 95% CRI global compressibility (x10-9 /Pa) |
| 95rangeCompr | 95% CRI range of global compressibility (x10-9 /Pa) |

* + **model(*number*)\_*fitName*\_Estimates\_indiv.pdf** (created only when the selected model includes individual-specific parameters)

|  |  |
| --- | --- |
| ID | Whale ID |
| numGlides | Number of sub-glides used in the model |
| meanBD | Mean whale-by-whale body density (kg/m3) |
| L95BD | Lower 95% CRI whale-by-whale body density (kg/m3) |
| U95BD | Upper 95% CRI whale-by-whale body density (kg/m3) |
| 95rangeBD | 95% CRI range of whale-by-whale body density (kg/m3) |
| meanCdAM | Mean whale-by-whale Cd\*A/m (x10-6 m2/kg) |
| L95CdAM | Lower 95% CRI whale-by-whale Cd\*A/m (x10-6 m2/kg) |
| U95CdAM | Upper 95% CRI whale-by-whale Cd\*A/m (x10-6 m2/kg) |
| 95rangeCdAM | 95% CRI range of whale-by-whale Cd\*A/m (x10-6 m2/kg) |
| meanVair | Mean whale-by-whale Vair/m (mL/kg3) |
| L95Vair | Lower 95% CRI whale-by-whale Vair/m (mL/kg3) |
| U95Vair | Upper 95% CRI whale-by-whale Vair/m (mL/kg3) |
| 95rangeVair | 95% CRI range of whale-by-whale Vair/m (mL/kg3) |

* + **model(*number*)\_*fitName*\_Estimates\_global.pdf** (created only when the selected model includes dive-specific Vair).

|  |  |
| --- | --- |
| ID | Whale ID |
| dive.number | Dive ID |
| max.dive.depth | Maximum dive depth (m) |
| dive.duration | Dive duration (s) |
| meanVair.d | Mean dive-by-dive Vair/m (mL/kg3) |
| L95Vair.d | Lower 95% CRI dive-by-dive Vair/m (mL/kg3) |
| U95Vair.d | Upper 95% CRI dive-by-dive Vair/m (mL/kg3) |
| 95rangeVair.d | 95% CRI range of dive-by-dive Vair/m (mL/kg3) |