

# Package: Xcertainty (via r-universe)

October 22, 2024

**Type** Package

**Title** Estimating Lengths and Uncertainty from Photogrammetric Imagery

**Version** 1.0.0

**Date** 2024-09-24

**Author** Joshua Hewitt [aut], K.C. Bierlich [aut, cre], Enrico Pirotta [aut]

**Maintainer** K.C. Bierlich <bierlick@oregonstate.edu>

**Description** Implementation of Bayesian models for estimating object lengths and morphological relationships between object lengths using photographic data collected from drones. The Bayesian model is described in "Bayesian approach for predicting photogrammetric uncertainty in morphometric measurements derived from drones" (Bierlich et al., 2021, <[doi:10.3354/meps13814](https://doi.org/10.3354/meps13814)>).

**URL** <https://github.com/MMI-CODEX/Xcertainty>

**BugReports** <https://github.com/MMI-CODEX/Xcertainty/issues>

**License** MIT + file LICENSE

**Depends** R (>= 3.0.2), nimble

**Imports** tidyr, dplyr, coda

**Suggests** testthat, knitr, rmarkdown, stringr, ggdist, tidyverse

**VignetteBuilder** knitr, rmarkdown

**LazyData** true

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**Repository** <https://mmi-codex.r-universe.dev>

**RemoteUrl** <https://github.com/mmi-codex/xcertainty>

**RemoteRef** HEAD

**RemoteSha** 8b6915926d40c2bcdaf02b4d8aef644bcf0d072e

## Contents

body_condition . . . . .	2
body_condition_measurements . . . . .	5
body_condition_measurement_estimates . . . . .	7
breakFun . . . . .	7
calibration . . . . .	8
calibration2 . . . . .	9
calibration_sampler . . . . .	9
combine_observations . . . . .	11
co_data . . . . .	12
flatten_data . . . . .	13
growth_curve_sampler . . . . .	15
gw_data . . . . .	17
independent_length_sampler . . . . .	18
nondecreasing_length_sampler . . . . .	20
parse_observations . . . . .	22
whales . . . . .	24
whale_info . . . . .	25
<b>Index</b>	<b>27</b>

---

body_condition	<i>Compute body condition metrics for a set of measurements</i>
----------------	---

---

## Description

Function that post-processes posterior samples from a sampler, such as `independent_length_sampler()`.

## Usage

```
body_condition(
  data,
  output,
  length_name,
  width_names,
  width_increments,
  summary.burn = 0.5,
  height_ratios = rep(1, length(width_names)),
  metric = c("surface_area", "body_area_index", "body_volume", "standardized_widths")
)
```

## Arguments

data	The output from <code>parse_observations</code>
output	The return object from a sampler
length_name	The name of the total-length measurement in the dataset

width_names	Character vector with the names of the width measurements in the dataset
width_increments	Numeric vector indicating which perpendicular width segment each width_names entry corresponds to, reported as a percentage along an animal's total length (i.e., 5 for "5%", etc.)
summary.burn	proportion of posterior samples to discard before computing posterior summary statistics
height_ratios	numeric vector used to compute 'body_volume' metric. the 'body_volume' metric assumes the animal's height at a width_increment is the measured width (estimate) times the corresponding entry in height_ratios. By default, all height_ratios are assumed to equal 1, which reflects a default assumption that an animal's vertical cross sections are circular rather than elliptical.
metric	Character vector of the body condition metrics to compute

### Value

outputs a list with five elements:

**surface\_area** a list containing the surface area samples and summaries for each Subject

**body\_area\_index** a list containing the body area index samples and summaries for each Subject

**body\_volume** a list containing the body volume samples and summaries for each Subject

**standardized\_widths** a list containing the standardized width samples and summaries for each Subject

**summaries** a list for each body condition metric containing summaries for each Subject

### Examples

```
library(stringr)
library(dplyr)

#
# parse data for Xcertainty
#

data("calibration2")
data("body_condition_measurements")

body_condition_measurements <- body_condition_measurements %>%
  select(!c(TL.10.0..Width, TL.15.0..Width, TL.5.0..Width, TL.90.0..Width,
            TL.95.0..Width))

# parse calibration study
calibration_data = parse_observations(
  x = calibration2,
  subject_col = 'L_train',
  meas_col = 'RRR.pix',
  tlen_col = 'L_train',
  image_col = 'Images',
  barometer_col = 'Baro...Ht',
```

```

    laser_col = 'Laser_Alt',
    flen_col = 'Focal.length',
    iwidth_col = 'Iw',
    swidth_col = 'Sw',
    uas_col = 'Aircraft'
  )

# identify the width columns in the dataset
width_names = grep(
  pattern = 'TL\\.\\.\\.\\.*',
  x = colnames(body_condition_measurements),
  value = TRUE
)

# parse whale data
whale_data = parse_observations(
  x = body_condition_measurements, #[1:5,],
  subject_col = 'Animal_ID',
  meas_col = c('TL', width_names),
  image_col = 'Image',
  barometer_col = 'BaroAlt',
  laser_col = 'LaserAlt',
  flen_col = 'Focal_Length',
  iwidth_col = 'Iw',
  swidth_col = 'Sw',
  uas_col = 'Aircraft',
  alt_conversion_col = 'BaroAlt'
)

#
# fit a basic model or load model output
#

if(interactive()) {

  # build sampler
  sampler = independent_length_sampler(
    data = combine_observations(calibration_data, whale_data),
    priors = list(
      image_altitude = c(min = 0.1, max = 130),
      altimeter_bias = rbind(
        data.frame(altimeter = 'Barometer', mean = 0, sd = 1e2),
        data.frame(altimeter = 'Laser', mean = 0, sd = 1e2)
      ),
      altimeter_variance = rbind(
        data.frame(altimeter = 'Barometer', shape = .01, rate = .01),
        data.frame(altimeter = 'Laser', shape = .01, rate = .01)
      ),
      altimeter_scaling = rbind(
        data.frame(altimeter = 'Barometer', mean = 1, sd = 1e1),
        data.frame(altimeter = 'Laser', mean = 1, sd = 1e1)
      ),
      pixel_variance = c(shape = .01, rate = .01),

```

```

        object_lengths = c(min = .01, max = 20)
      )
    )

    # run sampler
    body_condition_measurement_estimates = sampler(niter = 1e4, thin = 100)

  } else {
    data("body_condition_measurement_estimates")
  }

#
# post-process data
#

# enumerate the width locations along the animal's length
width_increments = as.numeric(
  str_extract(
    string = width_names,
    pattern = '[0-9]+'
  )
)

# compute body condition scores
body_condition_output = body_condition(
  data = whale_data,
  output = body_condition_measurement_estimates,
  length_name = 'TL',
  width_names = width_names,
  width_increments = width_increments,
  summary.burn = .5
)

body_condition_output$summaries

```

---

body\_condition\_measurements

*Humpback whale measurement data from Duke University's Marine Robotics and Remote Sensing (MaRRS) Lab*

---

### **Description**

Photogrammetric measurements of humpback whales to estimate total body length and body condition.

### **Usage**

```
body_condition_measurements
```

**Format**

A data frame with 29 rows and 28 columns:

**Animal\_ID** unique ID for the individual whale

**TL** total body length measurement (m)

**TL.10.0..Width** Width of whale (m), pre-computed from pixels using the reported laser altimeter measurement. Width is taken at a cross-section perpendicular to the whale's center line, running from the middle of the rostrum (loosely, the whale's beak/nose) to the middle of the peduncle (the point where the tail connects to the rest of the body). The cross-section is taken 10 from the animal's rostrum to its peduncle.

**TL.15.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 15 to its peduncle.

**TL.20.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 20 to its peduncle.

**TL.25.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 25 to its peduncle.

**TL.30.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 30 to its peduncle.

**TL.35.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 35 to its peduncle.

**TL.40.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 40 to its peduncle.

**TL.45.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 45 to its peduncle.

**TL.50.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 50 to its peduncle.

**TL.55.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 55 to its peduncle.

**TL.60.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 60 to its peduncle.

**TL.65.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 65 to its peduncle.

**TL.70.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 70 to its peduncle.

**TL.75.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 75 to its peduncle.

**TL.80.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 80 to its peduncle.

**TL.85.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 85 to its peduncle.

**TL.90.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 90 to its peduncle.

**TL.95.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 95 to its peduncle.

**TL.5.0..Width** Same as TL.10.0..Width, but taken at a cross-section that is 5 to its peduncle.

**Image** image name

**BaroAlt** the barometer altitude adjusted for the launch height of the drone

**LaserAlt** the altitude recorded by the laser (LiDAR) altimeter

**Focal\_Length** focal length of the camera (mm)

**Iw** image width (px)

**Sw** sensor width (mm)

**Aircraft** the unoccupied aircraft system (UAS), or drone, used in data collection

**Source**

<<https://doi.org/10.3389/fmars.2021.749943>>

---

body\_condition\_measurement\_estimates  
*Sample MCMC output*

---

**Description**

Posterior estimates for lengths and widths of a whale. See `help("body_condition")` for computation details.

**Usage**

```
body_condition_measurement_estimates
```

**Format**

A list with 5 elements:

**altimeters** Posterior samples and summaries for altimeters

**images** Posterior samples and summaries for images

**pixel\_error** Posterior samples and summaries for pixel error component of measurement error model

**objects** Posterior samples and summaries for unknown object lengths that were estimated

**summaries** `data.frames` with posterior summaries, collated from all other list elements.

---

breakFun *Break function (required in models)*

---

**Description**

Implements Heaviside step function for use in nimble models,  $H(B) = 1$  if  $B \leq \delta$ . For internal use only. Not intended to be called directly by users.

**Usage**

```
breakFun(B, delta)
```

**Arguments**

B argument to evaluate function at

delta breakpoint location

**Value**

1 if  $B \leq \delta$ , and 0 otherwise

**Examples**

```
breakFun(B = 1, delta = 0)
```

---

calibration

*Calibration (training) data*

---

**Description**

Photogrammetric measurements of known-sized calibration objects to be used as training data.

**Usage**

```
calibration
```

**Format**

A data frame with 657 rows and 10 columns:

**CO.ID** the calibration object ID in training data

**Lpix** length measurement (px)

**CO.L** the true length of the calibration object (m)

**image** image name

**Baro\_Alt** the barometer altitude adjusted for the launch height of the drone:  $\text{Baro\_raw} + \text{Launch\_Ht}$

**Laser\_Alt** the altitude recorded by the laser (LiDAR) altimeter

**Focal\_Length** focal length of the camera (mm)

**Iw** image width (px)

**Sw** sensor width (mm)

**uas** the unoccupied aircraft system (UAS), or drone, used in data collection

**Source**

<<https://doi.org/10.1111/gcb.17366>>



---

calibration2	<i>Calibration (training) data from Duke University's Marine Robotics and Remote Sensing (MaRRS) Lab</i>
--------------	--

---

**Description**

Photogrammetric measurements of known-sized calibration objects to be used as training data.

**Usage**

calibration2

**Format**

A data frame with 46 rows and 9 columns:

**L\_train** the true length of the calibration object (m)

**RRR.pix** length measurement (px)

**Images** image name

**Baro...Ht** the barometer altitude adjusted for the launch height of the dronet

**Laser\_Alt** the altitude recorded by the laser (LiDAR) altimeter

**Focal.length** focal length of the camera (mm)

**Iw** image width (px)

**Sw** sensor width (mm)

**Aircraft** the unoccupied aircraft system (UAS), or drone, used in data collection

**Source**

<<https://doi.org/10.3389/fmars.2021.749943>>

---

calibration_sampler	<i>MCMC sampler for calibration data</i>
---------------------	--

---

**Description**

Build an MCMC sampler that only uses calibration data to estimate measurement error parameters

**Usage**

calibration\_sampler(data, priors, package\_only = FALSE)

**Arguments**

<code>data</code>	Photogrammetric data formatted for Xcertainty models, required to be an object with class <code>obs.parsed</code> , which can be obtained by running <code>parse_observations()</code>
<code>priors</code>	list with components that define the model's prior distribution. See <code>help("flatten_data")</code> for more details.
<code>package_only</code>	TRUE to return the formatted data used to build the sampler, otherwise FALSE to return the sampler

**Value**

outputs a function to run a sampler, the function arguments are:

**niter** set the number of iterations

**burn** set the number samples to discard

**thin** set the thinning rate

**Examples**

```
# load example wide-format data
data("calibration")

# parse calibration study
calibration_data = parse_observations(
  x = calibration,
  subject_col = 'CO.ID',
  meas_col = 'Lpix',
  tlen_col = 'CO.L',
  image_col = 'image',
  barometer_col = 'Baro_Alt',
  laser_col = 'Laser_Alt',
  flen_col = 'Focal_Length',
  iwidth_col = 'Iw',
  swidth_col = 'Sw',
  uas_col = 'uas'
)

# build sampler
sampler_data = calibration_sampler(
  data = calibration_data,
  priors = list(
    image_altitude = c(min = 0.1, max = 130),
    altimeter_bias = rbind(
      data.frame(altimeter = 'Barometer', mean = 0, sd = 1e2),
      data.frame(altimeter = 'Laser', mean = 0, sd = 1e2)
    ),
    altimeter_variance = rbind(
      data.frame(altimeter = 'Barometer', shape = .01, rate = .01),
      data.frame(altimeter = 'Laser', shape = .01, rate = .01)
    ),
    altimeter_scaling = rbind(
```

```

      data.frame(altimeter = 'Barometer', mean = 1, sd = 1e1),
      data.frame(altimeter = 'Laser', mean = 1, sd = 1e1)
    ),
    pixel_variance = c(shape = .01, rate = .01)
  ),
  # set to false to return sampler function
  package_only = TRUE
)

```

---

combine\_observations *Combine parsed observations into a single parsed object*

---

### Description

Combine parsed observations, such as calibration and observation (whale) data into a single parsed object. This combined, single parsed object can then be used as the data input for one of the samplers.

### Usage

```
combine_observations(...)
```

### Arguments

... Parsed datasets to combine (i.e., outputs from `Xcertainty::parsed_observations`)

### Value

outputs a list with four elements:

**pixel\_counts** a tibble containing the measurements in pixels linked with Subject, Measurement description, Image, and the Timepoint

**training\_objects** a tibble containing the Subject, Measurement, Length, and Timepoint. NULL if no training objects were included

**prediction\_objects** a tibble containing the Subject, Measurement, and Timepoint. NULL if no prediction data included

**image\_info** a tibble containing the Image, Barometer, Laser, FocalLength, ImageWidth, Sensor-Width, and UAS

### Examples

```

# load example wide-format data
data("calibration")
data("whales")

# parse calibration study
calibration_data = parse_observations(
  x = calibration,

```

```

subject_col = 'CO.ID',
meas_col = 'Lpix',
tlen_col = 'CO.L',
image_col = 'image',
barometer_col = 'Baro_Alt',
laser_col = 'Laser_Alt',
flen_col = 'Focal_Length',
iwidth_col = 'Iw',
swidth_col = 'Sw',
uas_col = 'uas'
)

# parse field study
whale_data = parse_observations(
  x = whales,
  subject_col = 'whale_ID',
  meas_col = 'TL.pix',
  image_col = 'Image',
  barometer_col = 'AltitudeBarometer',
  laser_col = 'AltitudeLaser',
  flen_col = 'FocalLength',
  iwidth_col = 'ImageWidth',
  swidth_col = 'SensorWidth',
  uas_col = 'UAS',
  timepoint_col = 'year'
)

# combine parsed calibration and observation (whale) data
combined_data = combine_observations(calibration_data, whale_data)

```

---

co\_data

*Calibration (training) data for gray whale example*


---

### Description

Photogrammetric measurements of known-sized calibration objects to be used as training data.

### Usage

co\_data

### Format

A data frame with 118 rows and 15 columns:

**uas** the unoccupied aircraft system (UAS), or drone, used in data collection

**CO.ID** the calibration object ID in training data

**CO.L** the true length of the calibration object (m)

**year** Year

**image** image name  
**date** Date  
**Sw** sensor width (mm)  
**Iw** image width (px)  
**Focal\_Length** focal length of the camera (mm)  
**Focal\_Length\_adj** the adjusted focal length (mm) to account for internal processing that corrects for barrel distortion  
**Baro\_raw** raw altitude recorded by the barometer altimeter  
**Launch\_Ht** the launch height of the drone  
**Baro\_Alt** the barometer altitude adjusted for the launch height of the drone: Baro\_raw + Launch\_Ht  
**Laser\_Alt** the altitude recorded by the laser (LiDAR) altimeter  
**Lpix** length measurement (px)

### Source

<<https://doi.org/10.1139/dsa-2023-0051>>

---

flatten\_data

*Reformat photogrammetric data for model-based analysis*

---

### Description

For internal use only. Not intended to be called directly by users.

### Usage

```

flatten_data(
  data = NULL,
  priors,
  pixel_counts = data$pixel_counts,
  training_objects = data$training_objects,
  image_info = data$image_info,
  prediction_objects = data$prediction_objects
)
  
```

### Arguments

data	A list object, or similar that includes components that describe observations to analyze. Components are automatically extracted into this function's other arguments. See the remaining documentation for details about required components.
priors	list with elements altitude, lengths, bias, and sigma that parameterize the prior distributions for the Bayesian model. The bias components may specify separate priors for each UAS/altimeter type combination, or for all barometers at once based on the information provided for joining.

```

pixel_counts    data.frame with columns Subject, Measurement, Image, and PixelCount
                 that describe the length measurements taken from images
training_objects
                 data.frame with columns Subject, Measurement, and Length that describe
                 the known lengths of the objects used to calibrate the photogrammetric model
image_info      data.frame with columns Image, Barometer, Laser, FocalLength, ImageWidth,
                 and SensorWidth that describe the images used in the photogrammetric study
prediction_objects
                 data.frame with elements Subject, Measurement, and Timepoint that de-
                 scribe the unknown lengths of objects that should be estimated

```

## Details

Assemble data.frame objects into a format that can be analyzed using numerical methods. This function is analogous to `stats::model.matrix`, which generates design matrices for models that are specified via formulas.

## Examples

```

# load example wide-format data
data("calibration")
data("whales")

# parse calibration study
calibration_data = parse_observations(
  x = calibration,
  subject_col = 'CO.ID',
  meas_col = 'Lpix',
  tlen_col = 'CO.L',
  image_col = 'image',
  barometer_col = 'Baro_Alt',
  laser_col = 'Laser_Alt',
  flen_col = 'Focal_Length',
  iwidth_col = 'Iw',
  swidth_col = 'Sw',
  uas_col = 'uas'
)

# parse field study
whale_data = parse_observations(
  x = whales,
  subject_col = 'whale_ID',
  meas_col = 'TL.pix',
  image_col = 'Image',
  barometer_col = 'AltitudeBarometer',
  laser_col = 'AltitudeLaser',
  flen_col = 'FocalLength',
  iwidth_col = 'ImageWidth',
  swidth_col = 'SensorWidth',
  uas_col = 'UAS',
  timepoint_col = 'year'
)

```

```
)

# combine parsed calibration and observation (whale) data
combined_data = combine_observations(calibration_data, whale_data)
```

---

growth\_curve\_sampler *MCMC sampler for measurements of individuals with replicates and age information to generate growth curve*

---

## Description

Build an MCMC sampler that uses calibration data to estimate the total length of animals. The total lengths are assumed to follow a growth curve model, so replicates across time points that include age information are required to fit the model. The length model is a von-Bertalanffy-Putter growth model, following Pirota & Bierlich et al., (in revision).

## Usage

```
growth_curve_sampler(data, priors, subject_info, package_only = FALSE)
```

## Arguments

data	Photogrammetric data formatted for Xcertainty models, required to be an object with class <code>obs.parsed</code> , which can be obtained by running <code>parse_observations()</code>
priors	list with components that define the model's prior distribution. See <code>help("flatten_data")</code> for more details.
subject_info	data.frame with elements Year, Subject, Group, ObservedAge, and AgeType. See <code>help("whale_info")</code> for descriptions of data.frame columns.
package_only	TRUE to return the formatted data used to build the sampler, otherwise FALSE to return the sampler

## Value

outputs a function to run a sampler, the function arguments are:

**niter** set the number of iterations

**burn** set the number samples to discard

**thin** set the thinning rate

## Examples

```
# load example wide-format data
data("calibration")
data("whales")
data("whale_info")

# parse calibration study
```

```

calibration_data = parse_observations(
  x = calibration,
  subject_col = 'CO.ID',
  meas_col = 'Lpix',
  tlen_col = 'CO.L',
  image_col = 'image',
  barometer_col = 'Baro_Alt',
  laser_col = 'Laser_Alt',
  flen_col = 'Focal_Length',
  iwidth_col = 'Iw',
  swidth_col = 'Sw',
  uas_col = 'uas'
)

# parse field study
whale_data = parse_observations(
  x = whales,
  subject_col = 'whale_ID',
  meas_col = 'TL.pix',
  image_col = 'Image',
  barometer_col = 'AltitudeBarometer',
  laser_col = 'AltitudeLaser',
  flen_col = 'FocalLength',
  iwidth_col = 'ImageWidth',
  swidth_col = 'SensorWidth',
  uas_col = 'UAS',
  timepoint_col = 'year'
)

# build sampler
sampler_data = growth_curve_sampler(
  data = combine_observations(calibration_data, whale_data),
  priors = list(
    image_altitude = c(min = 0.1, max = 130),
    altimeter_bias = rbind(
      data.frame(altimeter = 'Barometer', mean = 0, sd = 1e2),
      data.frame(altimeter = 'Laser', mean = 0, sd = 1e2)
    ),
    altimeter_variance = rbind(
      data.frame(altimeter = 'Barometer', shape = .01, rate = .01),
      data.frame(altimeter = 'Laser', shape = .01, rate = .01)
    ),
    altimeter_scaling = rbind(
      data.frame(altimeter = 'Barometer', mean = 1, sd = 1e1),
      data.frame(altimeter = 'Laser', mean = 1, sd = 1e1)
    ),
    pixel_variance = c(shape = .01, rate = .01),
    # priors from Agbayani et al.
    zero_length_age = c(mean = -5.09, sd = 0.4),
    growth_rate = c(mean = .18, sd = .01),
    # additional priors
    group_asymptotic_size = rbind(
      Female = c(mean = 12, sd = .5),

```



```

      Male = c(mean = 12, sd = .5)
    ),
    group_asymptotic_size_trend = rbind(
      Female = c(mean = 0, sd = 1),
      Male = c(mean = 0, sd = 1)
    ),
    subject_group_distribution = c(Female = .5, Male = .5),
    asymptotic_size_sd = c(min = 0, max = 10),
    min_calf_length = 3.5,
    # To model break points between 1990 and 2015
    group_size_shift_start_year = c(min = 1990, max = 2015)
  ),
  subject_info = whale_info,
  # set to false to return sampler function
  package_only = TRUE
)

```

---

gw\_data

*Gray whale measurement data*


---

## Description

An example dataset of gray whale measurements from drone-based photogrammetry.

## Usage

```
gw_data
```

## Format

A tibble with 15 rows and 34 columns:

**whale\_ID** unique individual

**image** image name

**year** Year

**DOY** Day of Year

**uas** the unoccupied aircraft system (UAS), or drone, used in data collection

**Focal\_Length** focal length of the camera (mm)

**Focal\_Length\_adj** the adjusted focal length (mm) to account for internal processing that corrects for barrel distortion

**Sw** sensor width (mm)

**Iw** image width (px)

**Baro\_raw** raw altitude recorded by the barometer altimeter

**Launch\_Ht** the launch height of the drone

**Baro\_Alt** the barometer altitude adjusted for the launch height of the drone:  $\text{Baro\_raw} + \text{Launch\_Ht}$

**Laser\_Alt** the altitude recorded by the laser (LiDAR) altimeter  
**CO.ID** the calibration object ID in training data  
**TL\_px** total body length measurement (px)  
**TL\_w05.00\_px** Body width measurement (px) at 5% of total length  
**TL\_w10.00\_px** Body width measurement (px) at 10% of total length  
**TL\_w15.00\_px** Body width measurement (px) at 15% of total length  
**TL\_w20.00\_px** Body width measurement (px) at 20% of total length  
**TL\_w25.00\_px** Body width measurement (px) at 25% of total length  
**TL\_w30.00\_px** Body width measurement (px) at 30% of total length  
**TL\_w35.00\_px** Body width measurement (px) at 35% of total length  
**TL\_w40.00\_px** Body width measurement (px) at 40% of total length  
**TL\_w45.00\_px** Body width measurement (px) at 45% of total length  
**TL\_w50.00\_px** Body width measurement (px) at 50% of total length  
**TL\_w55.00\_px** Body width measurement (px) at 55% of total length  
**TL\_w60.00\_px** Body width measurement (px) at 60% of total length  
**TL\_w65.00\_px** Body width measurement (px) at 65% of total length  
**TL\_w70.00\_px** Body width measurement (px) at 70% of total length  
**TL\_w75.00\_px** Body width measurement (px) at 75% of total length  
**TL\_w80.00\_px** Body width measurement (px) at 80% of total length  
**TL\_w85.00\_px** Body width measurement (px) at 85% of total length  
**TL\_w90.00\_px** Body width measurement (px) at 90% of total length  
**TL\_w95.00\_px** Body width measurement (px) at 95% of total length

### Source

<<https://mmi.oregonstate.edu/gemm-lab>>

---

independent\_length\_sampler

*MCMC sampler for individuals with independent measurements.*

---

### Description

Build an MCMC sampler that uses calibration data to estimate independent, unknown measurements. This model assumes all Subject/Measurement/Timepoint combinations are independent. So, this sample is well suited for data containing individuals that either have no replicate samples or have replicate samples that are independent over time, such as body condition which can increase or decrease over time, as opposed to length which should be stable or increase over time. It can also be used to estimate lengths when there are replicate measurements. However, since the model assumes all Subject/Measurement/Timepoint combinations are independent, no strength will be borrowed across temporal replication of a subject's measurements, for example.

**Usage**

```
independent_length_sampler(data, priors, package_only = FALSE)
```

**Arguments**

<code>data</code>	Photogrammetric data formatted for Xcertainty models, required to be an object with class <code>obs.parsed</code> , which can be obtained by running <code>parse_observations()</code>
<code>priors</code>	list with components that define the model's prior distribution. See <code>help("flatten_data")</code> for more details.
<code>package_only</code>	TRUE to return the formatted data used to build the sampler, otherwise FALSE to return the sampler

**Value**

outputs a function to run a sampler, the function arguments are:

**niter** set the number of iterations

**burn** set the number samples to discard

**thin** set the thinning rate

**Examples**

```
# load example wide-format data
data("calibration")
data("whales")
data("whale_info")

# parse calibration study
calibration_data = parse_observations(
  x = calibration,
  subject_col = 'CO.ID',
  meas_col = 'Lpix',
  tlen_col = 'CO.L',
  image_col = 'image',
  barometer_col = 'Baro_Alt',
  laser_col = 'Laser_Alt',
  flen_col = 'Focal_Length',
  iwidth_col = 'Iw',
  swidth_col = 'Sw',
  uas_col = 'uas'
)

# parse field study
whale_data = parse_observations(
  x = whales,
  subject_col = 'whale_ID',
  meas_col = 'TL.pix',
  image_col = 'Image',
  barometer_col = 'AltitudeBarometer',
  laser_col = 'AltitudeLaser',
```

```

    flen_col = 'FocalLength',
    iwidth_col = 'ImageWidth',
    swidth_col = 'SensorWidth',
    uas_col = 'UAS',
    timepoint_col = 'year'
  )

  # build sampler
  sampler_data = independent_length_sampler(
    data = combine_observations(calibration_data, whale_data),
    priors = list(
      image_altitude = c(min = 0.1, max = 130),
      altimeter_bias = rbind(
        data.frame(altimeter = 'Barometer', mean = 0, sd = 1e2),
        data.frame(altimeter = 'Laser', mean = 0, sd = 1e2)
      ),
      altimeter_variance = rbind(
        data.frame(altimeter = 'Barometer', shape = .01, rate = .01),
        data.frame(altimeter = 'Laser', shape = .01, rate = .01)
      ),
      altimeter_scaling = rbind(
        data.frame(altimeter = 'Barometer', mean = 1, sd = 1e1),
        data.frame(altimeter = 'Laser', mean = 1, sd = 1e1)
      ),
      pixel_variance = c(shape = .01, rate = .01),
      object_lengths = c(min = .01, max = 20)
    ),
    # set to false to return sampler function
    package_only = TRUE
  )

```

---

nondecreasing\_length\_sampler

*MCMC sampler for measurements of individuals with replicates but no age information.*

---

### Description

Build an MCMC sampler that uses calibration data to estimate measurements that are assumed to be non-decreasing in time. This sampler is well suited for when individuals have replicate measurements across time points but do not have age information. The model estimates changes in unique combinations of Subject/Measurement pairs over Timepoints.

### Usage

```
nondecreasing_length_sampler(data, priors, package_only = FALSE)
```

**Arguments**

<code>data</code>	Photogrammetric data formatted for Xcertainty models, required to be an object with class <code>obs.parsed</code> , which can be obtained by running <code>parse_observations()</code>
<code>priors</code>	list with components that define the model's prior distribution. See <code>help("flatten_data")</code> for more details.
<code>package_only</code>	TRUE to return the formatted data used to build the sampler, otherwise FALSE to return the sampler

**Value**

outputs a function to run a sampler, the function arguments are:

**niter** set the number of iterations

**burn** set the number samples to discard

**thin** set the thinning rate

**Examples**

```
# load example wide-format data
data("calibration")
data("whales")
data("whale_info")

# parse calibration study
calibration_data = parse_observations(
  x = calibration,
  subject_col = 'CO.ID',
  meas_col = 'Lpix',
  tlen_col = 'CO.L',
  image_col = 'image',
  barometer_col = 'Baro_Alt',
  laser_col = 'Laser_Alt',
  flen_col = 'Focal_Length',
  iwidth_col = 'Iw',
  swidth_col = 'Sw',
  uas_col = 'uas'
)

# parse field study
whale_data = parse_observations(
  x = whales,
  subject_col = 'whale_ID',
  meas_col = 'TL.pix',
  image_col = 'Image',
  barometer_col = 'AltitudeBarometer',
  laser_col = 'AltitudeLaser',
  flen_col = 'FocalLength',
  iwidth_col = 'ImageWidth',
  swidth_col = 'SensorWidth',
  uas_col = 'UAS',
```

```

    timepoint_col = 'year'
  )

# build sampler
sampler_data = nondecreasing_length_sampler(
  data = combine_observations(calibration_data, whale_data),
  priors = list(
    image_altitude = c(min = 0.1, max = 130),
    altimeter_bias = rbind(
      data.frame(altimeter = 'Barometer', mean = 0, sd = 1e2),
      data.frame(altimeter = 'Laser', mean = 0, sd = 1e2)
    ),
    altimeter_variance = rbind(
      data.frame(altimeter = 'Barometer', shape = .01, rate = .01),
      data.frame(altimeter = 'Laser', shape = .01, rate = .01)
    ),
    altimeter_scaling = rbind(
      data.frame(altimeter = 'Barometer', mean = 1, sd = 1e1),
      data.frame(altimeter = 'Laser', mean = 1, sd = 1e1)
    ),
    pixel_variance = c(shape = .01, rate = .01),
    object_lengths = c(min = .01, max = 20)
  ),
  # set to false to return sampler function
  package_only = TRUE
)

```

---

parse_observations	<i>Pre-process training and experimental data from wide-format to long-format</i>
--------------------	---

---

## Description

Photogrammetric data are often recorded in a wide-format data frame, in which each row contains all measurement information for a single animal. The row contains the image information (i.e., observed altitude and sensor information) as well as all measurements for a given subject. This function parses the wide-format data into a normalized list of data.frame objects that separately describe the image and measurement data. This function can process observations of calibration data as well as experimental data.

## Usage

```

parse_observations(
  x,
  subject_col,
  meas_col,
  tlen_col = NULL,
  image_col,
  barometer_col = NULL,

```

```

    laser_col = NULL,
    flen_col,
    iwidth_col,
    swidth_col,
    uas_col,
    timepoint_col = NULL,
    alt_conversion_col = NULL
  )

```

### Arguments

<code>x</code>	Wide-format data frame describing images and measurements
<code>subject_col</code>	column name in <code>x</code> for subject IDs
<code>meas_col</code>	character vector of column names in <code>x</code> with pixel-counts for each measurement of a subject
<code>tlen_col</code>	column name in <code>x</code> with the true length value (i.e., in meters) of a measurement; primarily used to specify the true length value for an observation of a calibration object. If <code>NULL</code> , then no true length will be associated with the measurement.
<code>image_col</code>	column name in <code>x</code> containing names of images from which measurements are taken
<code>barometer_col</code>	column name in <code>x</code> with Barometer altimeter values
<code>laser_col</code>	column name in <code>x</code> with Laser altimeter values
<code>flen_col</code>	column name in <code>x</code> with camera focal lengths (mm)
<code>iwidth_col</code>	column name in <code>x</code> with image widths (pixels)
<code>swidth_col</code>	column name in <code>x</code> with camera sensor widths (mm)
<code>uas_col</code>	column names in <code>x</code> with UAS name or ID
<code>timepoint_col</code>	column name in <code>x</code> with a timepoint value of a measurement. If <code>NULL</code> , then all measurements are assumed to be at the same timepoint, or equivalently, that time does not matter for the analysis
<code>alt_conversion_col</code>	if not <code>NULL</code> , column name in <code>x</code> with an altitude used to convert measurement columns from lengths to pixels

### Value

outputs a list with four elements:

**pixel\_counts** a tibble containing the measurements in pixels linked with Subject, Measurement description, Image, and the Timepoint

**training\_objects** a tibble containing the Subject, Measurement, Length, and Timepoint. `NULL` if no training objects were included

**prediction\_objects** a tibble containing the Subject, Measurement, and Timepoint. `NULL` if no prediction data included

**image\_info** a tibble containing the Image, Barometer, Laser, FocalLength, ImageWidth, Sensor-Width, and UAS

**Examples**

```

# load example wide-format data
data("calibration")
data("whales")

# parse calibration study
calibration_data = parse_observations(
  x = calibration,
  subject_col = 'CO.ID',
  meas_col = 'Lpix',
  tlen_col = 'CO.L',
  image_col = 'image',
  barometer_col = 'Baro_Alt',
  laser_col = 'Laser_Alt',
  flen_col = 'Focal_Length',
  iwidth_col = 'Iw',
  swidth_col = 'Sw',
  uas_col = 'uas'
)

# parse field study
whale_data = parse_observations(
  x = whales,
  subject_col = 'whale_ID',
  meas_col = 'TL.pix',
  image_col = 'Image',
  barometer_col = 'AltitudeBarometer',
  laser_col = 'AltitudeLaser',
  flen_col = 'FocalLength',
  iwidth_col = 'ImageWidth',
  swidth_col = 'SensorWidth',
  uas_col = 'UAS',
  timepoint_col = 'year'
)

# combine parsed calibration and observation (whale) data
combined_data = combine_observations(calibration_data, whale_data)

```

---

whales

*Gray whale metadata*


---

**Description**

Gray whale information and metadata that pairs with 'whales' data by "Subject"

**Usage**

whales



**Format**

A data frame with 826 rows and 14 columns:

**whale\_ID** unique individual

**sex** Female, Male, or NA

**Age** age in years

**AgeType** either 'known age' if individual was seen as a calf, or 'min age' from the date of date sighting

**year** Year

**date** Date

**Image** image name

**AltitudeBarometer** the barometer altitude adjusted for the launch height of the drone

**AltitudeLaser** the altitude recorded by the laser (LiDAR) altimeter

**FocalLength** focal length of the camera (mm)

**ImageWidth** image width (px)

**SensorWidth** sensor width (mm)

**UAS** the unoccupied aircraft system (UAS), or drone, used in data collection

**TL.pix** the total body length measurement in pixels

**Source**

<<https://doi.org/10.1111/gcb.17366>>

---

whale\_info

*Gray whale metadata*

---

**Description**

Gray whale information and metadata that pairs with 'whales' data by "Subject"

**Usage**

whale\_info

**Format**

A data frame with 293 rows and 5 columns:

**Year** year

**Subject** unique ID for individuals

**Group** sex; Male, Female (F), or NA

**ObservedAge** age in years

**AgeType** either 'known age' if individual was seen as a calf, or 'min age' from the date of date sighting

**Source**

<<https://doi.org/10.1111/gcb.17366>>

# Index

## \* datasets

- body\_condition\_measurement\_estimates,  
7
- body\_condition\_measurements, 5
- calibration, 8
- calibration2, 9
- co\_data, 12
- gw\_data, 17
- whale\_info, 25
- whales, 24

  

- body\_condition, 2
- body\_condition\_measurement\_estimates,  
7
- body\_condition\_measurements, 5
- breakFun, 7

  

- calibration, 8
- calibration2, 9
- calibration\_sampler, 9
- co\_data, 12
- combine\_observations, 11

  

- flatten\_data, 13

  

- growth\_curve\_sampler, 15
- gw\_data, 17

  

- independent\_length\_sampler, 18

  

- nondecreasing\_length\_sampler, 20

  

- parse\_observations, 22

  

- whale\_info, 25
- whales, 24