The data object

The `colonytrack_data` object contains the raw and preprocessed data along with various metadata elements describing the experimental set-up and the layout of the cage. The raw data and metadata files are read in by the `read_data` function and all timestamps are converted into the the Unix time format. Based on the start/end of the days (which is defined by the 'LightsOn' time in the Events file), the data are split into days and the partial days at the start and end of the experiment are padded with missing data placeholders. It almost always makes no sense to analyse these padded days, so they can be trimmed during later steps—but no data is discarded by default.

Components of the data object

Data processing

The raw data consist of antenna contacts and contain the time of the contact, the RFID tag of the animal and the ID of the RFID antenna where the contact was registered. The duration of the contact (in milliseconds) is also recorded. The `raw.data` element simply contains this information with the timestamp converted into Unix time.

Next, the consecutive antenna contacts for each subject are used to infer the cage in which it was at any time. If a subject is localised to a tunnel, then it is considered to remain in the cage from which it entered the tunnel—until it can be unequivocally located to a new cage. This cage presence information is contained in the `data` element, alongside the time at which the subject *moved into* the specified cage. In the case of multiple contacts at the same antenna, it is actually impossible to decide whether the subject moved back-and-forth *through* the tunnel or just triggered the antenna repeatedly from within the one cage (which is possible due to the confined layout of the cage rack and the required antenna sensitivity). For this reason, repeated antenna contacts are merged so that only *unequivocal cage transitions* are recorded in the processed data. Similarly, only pairs of (de-duplicated) antenna contacts that correspond to a cage transition are retained in the processed data. The information about which antenna pairs are informative comes from a cage layout network derived from the Cage layout metadata file and which is described in some more detail below. The timestamp information is given across several columns in different formats as described below.

Timestamps

The timestamp is given in seconds since the start of Unix time (00:00:00 UTC on 1 January 1970) and is also, for convenience, exploded into its date and hour components. Based on the light/dark cycle information provided in the Events metadata file, times are additionally converted to Zeitgeber time (ZT) where ZT0 is the time at which lights come on and ZT12 is when lights go off. A column of logical values specifying whether it is 'Nighttime' (lights off) or not is also included.

Time range

Because the Metrics calculation requires full 24-hour blocks of data, the range of the raw timestamps are padded so that they start at ZT0 preceding the first recorded antenna contact and end at ZT24 (equivalent to ZT0) following the last recorded antenna contact. The range of the first and last antenna contacts is given by `timestamp.range` in the `info` element and the padded range by `padded.timestamp.range`. The `days` (ZT0–ZT12, when lights are on) and `nights` (ZT12–ZT24, when the lights are off) define the timestamp ranges of each 12-hour block.

The cage network

The ColonyRack system consists of cage joined by tunnels and can thus be represented by an undirected network. This, indeed, is exactly what the ColonyTrack software does internally. This network is defined by the Cage layout description file and considers cages as vertices and tunnels/RFID antennae as edges. Details of the cage network are stored in the element `layout`. Using the capabilities of the `igraph` package, a network/graph is created Centrality.

Quality control

A `qc` element in the data object contains some quality control information that might be useful for identifying problems in the experiment. In some cases, an antenna contact may not be able to be assigned to one of the subjects in the subject metadata file. Such contacts are flagged as 'dirty data' and not used to generate the processed data found in the `data` element. The number of such unassigned events is given in `dirty.data.count` and details in `dirty.data`. These are usually just control lines generated by the ColonyRack software or can arise from the use of test tags (used, for example, to verify recording when setting up a new experiment). They can, however, also rarely occur when a tag is misread. If a very large number of 'dirty' reads are generated by the same SubjectID, this may indicate an error in your Subject metadata file—double-check that all of the subjects are correctly entered and that the tags have not become mutated by spreadsheet software.

As described above, only consecutive antenna contacts that span a cage are retained in the data object. If a subject is not recorded for whatever reason as it passes a reader, then the previous and next contacts may not flank a cage as defined in the Cage layout description file. In this case, the subject cannot be unequivocally located to a cage and is considered to remain in the cage it was last seen until its position can be unequivocally updated. Any such antenna contacts are referred to as 'non-trajectory reads' and the number of these that are encountered for each animal is given in the `qc` element under `non.trajectory.read.count`. Details on the timestamps and antenna identities are recorded under `non.trajectory.reads`.

In a few rare cases, an antenna may be triggered by the same subject at sub-millisecond intervals. This means that the timestamps are identical an is most likely due to the the subject resting in proximity to the antenna. These duplicate reads are discarded and a count of their occurrence is recorded under `submillisecond.count`.

Structure of the data object

Below is an overview of the hierarchy of the `colonytrack_data` object together with the names and classes of each component. Where the class is not part of the R base package, the parent package is given in square brackets after the class name. Round brackets indicate multiple elements (i.e. one for each subject or time window) of the same structure.

```
data : colonytrack_data [ColonyTrack]
    info : list
        timestamp.range : numeric
        padded.timestamp.range : numeric
        days : colonytrack_windows [ColonyTrack]
            id : character
            start : numeric
            end : numeric
        nights : colonytrack_windows [ColonyTrack]
            id : character
            start : numeric
            end : numeric
        subjects : character
        subject.info : data.frame
            SubjectID : character
            Tag : character
```

```
        (optional additional user-defined columns)
    processed : POSIXct
    version : character
qc : list
    dirty.data.count : integer
    dirty.data : data.frame
        Timestamp : numeric
        SubjectID : character
        ReaderID : character
        Duration : integer
    trajectory : list
        (SubjectID) : list
            non.trajectory.read.count : numeric
            non.trajectory.reads : data.frame
                Timestamp : numeric
                Cage : character
                ReaderPairID : character
            submillisecond.count : numeric
data : list
    (SubjectID) : data.frame
        Timestamp : numeric
        DateTime : POSIXct
        Day : character
        Hour : numeric
        ZTDay : character
        ZT : numeric
        Nighttime : logical
        Cage : character
raw.data : data.frame
    Timestamp : numeric
    TagID : character
    SubjectID : character
    ReaderID : character
    Duration : integer
layout : list
    cages : character
    cage.quality : data.frame
        Start : POSIXct
        End : POSIXct
        (CageID) : character
    readers : character
    transitions : data.frame
        Reader1 : character
        Reader1 : character
        Cage : character
    network : data.frame
        Sort : integer
        Source : character
        SourceType : character
        Link : character
        Target : character
        TargetType : character
    graph : igraph [igraph]
```

```
        shortest.paths : numeric matrix
        centrality : numeric
```