Ice Beacon Cookbook

CENTRE FOR EARTH OBSERVATION SCIENCE

Contents

Chapter 1

Ice Beacon Data

1.1 Critical Summary

This is a data management workflow 1 process.

Ice beacons were built by Solara Communications and David Babb for the BaySys project /emphTeam 1 - Climate and Marine System - Sea Ice. 16 ice beacons were deployed on the *2017 Churchill River and Mobile Ice Survey* campaign, and 10 ice beacons were deployed on the *2018 Hudson Bay Amundsen* campaign. The GPS location data recorded by the ice beacon is transmitted via Iridium to the Solara online data portal. The 2017 data tracks the drift of individual ice floes and the relative drift of ice beacons deployed in pairs or in array of the mobile ice pack offshore from Cape Churchill in southwestern Hudson Bay. The 2018 data tracks the movement of the ice pack and gains insight into the double gyre current movement about the Leg 1 region of the Hudson Bay.

1.1.1 Data Management Tools

1. **Ice Beacon Python Script:** Written by Victory Iyakoregha. Inputs both the Raw Churchill .csv ice beacon files and the Raw Amundsen .xlsx ice beacon files and outputs both as .csv files. The python script strips unnecessary columns, and calculates the speed and distance travelled of ice beacons using python functions that read the GPS and timestamp data. The processed ice beacon data-files are organized as a single processed ice beacon dataset.

1.3 Initial Data

1.3.1 Raw Ice Beacon Files

Dataset Description

The ice beacons record their GPS location at regular intervals set by the researcher until the ice floe breaks up and they sink. Two raw ice beacon datasets exist which are both **Data Level 0**:

1. Raw Churchill ice beacon files `.xlsx`
2. Raw Amundsen ice beacon files `.csv`

File Name(s)

- **Raw Churchill Ice Beacon Files:** `01.xlsx`, `02.xlsx`, `03.xlsx`, `04.xlsx`, `05.xlsx`, `06.xlsx`, `07.xlsx`, `08.xlsx`, `09.xlsx`, `10.xlsx`, `11.xlsx`, `12.xlsx`, `14.xlsx`, `15.xlsx`, `16.xlsx`, `CT.xlsx`
- **Raw Amundsen Ice Beacon Files:** `13.csv`, `17.csv`, `18.csv`, `19.csv`, `20.csv`, `21.csv`, `22.csv`, `23.csv`, `25.csv`, `26.csv`

File Source and Location

Raw CSV files are received from Dr. David Babb. These files are then stored `on GitLab`.

1.2 Ice Beacon Flow Chart

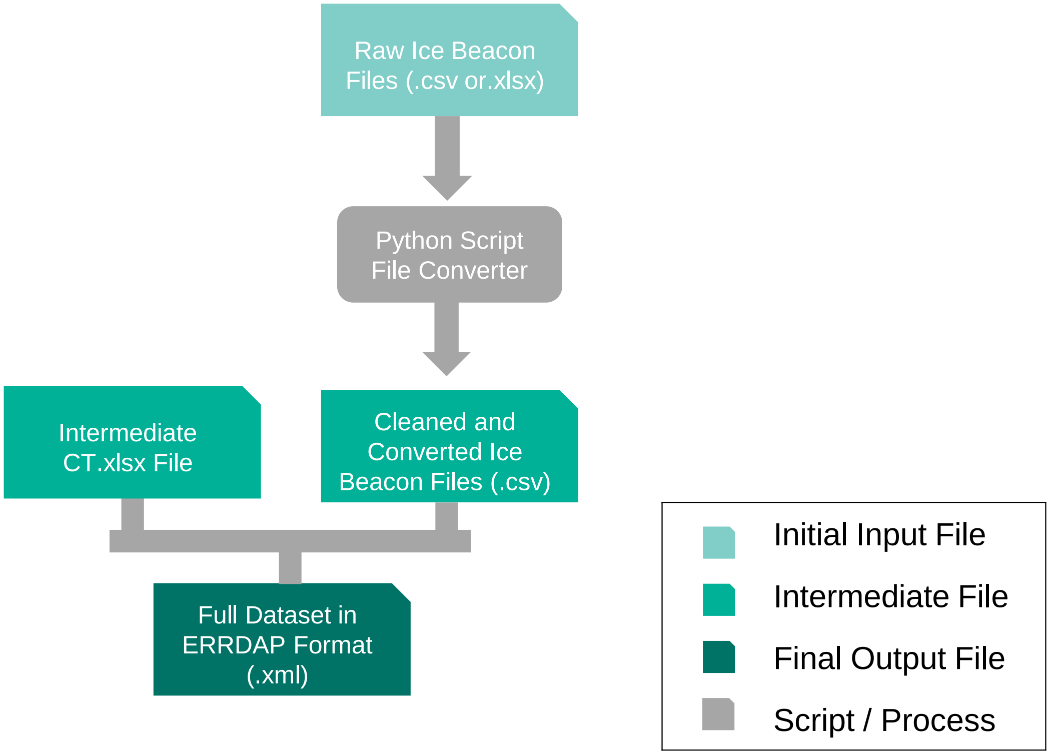
Raw Ice Beacon Files (.csv or.xlsx)
Python Script File Converter
Intermediate CT.xlsx File
Cleaned and Converted Ice Beacon Files (.csv)
Full Dataset in ERRDAP Format (.xml)
Initial Input File
Intermediate File
Final Output File
Script / Process

Figure 1.1: Flow chart of ice beacon data management

Dataset Variables

Table 1.1: Variables in Raw Churchill data files

Header	Description	Data Type	Range or Expected Values	Units
V1	Beacon ID	Identification name and number	N/A	(assigned name of beacon) CEOS (IMEI)
V2	GPS Latitude	Latitude	[-90, 90]	DDmm.mmmmn (n='N' or 'S')
V3	GPS Longitude	Longitude	[-180, 180]	DDDmm.mmmmn (n = 'E' or 'W')
V4	Year	date	N/A	YYYY
V5	Month	date	N/A	MM
V6	Day	date	N/A	DD
V7	Hour	time	N/A	HH
V8	Minute	time	N/A	MM
V9	Second	time	N/A	SS
V10	Battery Voltage of ice beacon	rational	N/A	Volts
V11	Internal Temperature of ice beacon	rational	N/A	Degrees Celcius

Table 1.2: Variables in Raw Amundsen data files

Header	**Description**	**Data Type**	**Range or Expected Values**	**Units**
Name	Beacon ID	Identification name and number	N/A	(assigned name of beacon) CEOS (IMEI)
Lat	GPS Latitude	Latitude	[-90, 90]	DDmm.mmmmn (n='N' or 'S')
Long	GPS Longitude	Longitude	[-180, 180]	DDDmm.mmmmn (n = 'E' or 'W')
Timestamp	Date and time of measurement	datetime	N/A	YYYY-MM-DD HH:MM:SS (time zone)

Note: The following variables/column headers exist on the file, but are not relevant to this analysis and were stripped by the Python script: “id”, “serial”, “message”, “hdop”, “pdop”, “speed”, “altitude”, and “heading”.

1.4 Scripts and Analytical Processes

1.4.1 Step 1: Importing Python and File Components

Script Type: Python

Files(s) In: clean CSV files

File(s) Out: converted CSV files

User Instructions

1. Download Python script process_Ice_beacon_files.py from Gitlab repository ceos_codebooks_and_scripts
2. Download Ice_beacon_files.ini to configure script details through input directory

1.4.2 Analytical Processes

1. **Import:** os.
2. **Import:** configparser.
3. **Import:** pandas **as** pd.
4. **Import:** datetime **from** datetime.
5. **Import:** math **as** m.
6. **Import:** lonlat, distance **from** geopy.distance.
7. Run **Python** script detailed below.

1.5 Step 2: Ice Beacon File Conversion

Script Type: Python Script

File(s) In: The collection of cleaned Ice Beacon CSV files

File(s) Out: The collection of converted Ice Beacon CSV files

1.5.1 Intermediate File 1 Name

User Instructions

1. Download Python script process_Ice_beacon_files.py from Gitlab repository ceos_codebooks_and_scripts
2. Download Ice_beacon_files.ini to configure script details through input directory
3. In Ice_beacon_files.ini provide the path to the ice beacon data directory in the Input Directory
4. In Ice_beacon_files.ini provide the path to the destination folder for converted Ice Beacon files in Processed Directory
5. Click "Run" at the top of the script window. The script may take several minutes to complete

Back-End Details

Back-End Script Location: Python script and [.ini] file located under Gitlab repository: Data Cleaning -> CEOS_codebooks_and_scripts -> Scripts -> Python -> Ice_Beacon

Libraries Used: Python os, configparser, pandas (pd), datetime, math (m), lonlat and distance (geopy.distance)

Analytical Process

This is a single-script Python application that performs the following tasks:

1. **Open_files** imports the previously cleaned **CSV** files from the `Basys` and `Churchill folders`.
2. **Clean_csv** accepts a single parameter, taking the dataframe for the CSV file, processing it, and returning it to open_files location. Clean_csv takes the original timestamp format and changes it to match the format of year-month-day and hours-minutes-seconds.
3. Column "message" takes "Battery Voltage" and "Internal Temperature", which are split by a whitespace, and put into their own columns "Battery Voltage" and "Internal Temperature".
4. "name", "lat", "long", and "timestamp" are renamed as: "Beacon ID", "Latitude", "Longitude", and "Timestamp", respectively.
5. "hdop", "pdop", "speed", "altitude", and "heading" are removed.
6. **Clean_excel** accepts a single parameter, taking the dataframe for the CSV file, processing it, and returning it to open_files location. Clean_excel takes the original timestamp format and changes it to match the format of year-month-day and hours-minutes-seconds.
7. Clean_excel adds the column titles: "Beacon ID", "Latitude", "Longitude", "Timestamp", "Battery Voltage", and "Internal Temperature".

8. **Clean_CT_file** accepts a single parameter, taking the dataframe for the CSV file, processing it, and returning it to open_files location. Clean_CT_file takes the original timestamp format and changes it to match the format of year-month-day and hours-minutes-seconds.

9. Clean_CT_file adds the column titles: "Beacon ID", "Latitude", "Longitude", "Timestamp", "Battery Voltage", and "Internal Temperature".

10. **Calc_distance** accepts a single parameter, taking the dataframe for the CSV file, processing it, and returning it to open_files location.

11. This function calculates the distance between every two points, in this case, the two points being each two rows of data using the difference between the latitude and longitude of each point.

12. **Calc_speed** accepts a single parameter, taking the dataframe for the CSV file, processing it, and returning it to open_files location.

13. This function calculates the speed between every two points, in this case, the two points being each two rows of data using the difference between the distance calculated in calc_distance() and timestamp.

14. **Main()** runs the script.

1.6 Final Output Data Files

1.6.1 Cleaned and Compiled Ice Beacon Data

Dataset Description

These are intermediate output files which display the calculated metadata. The intermediate files are partially cleaned before running the script. The script removes columns "hdop", "pdop", "speed", "altitude", and "heading" and gives the calculations for: time, latitude, longitude, distance, and speed. There is one CT.xlsx file for intermediate and cleaned data files used in combination with ice beacon files.

File Name(s)

Ice beacon files are named: 01.csv, 02.csv, 03.csv, 04.csv, 05.csv, 06.csv, 07.csv, 08.csv, 09.csv, 10.csv, 11.csv, 12.csv, 13.csv, 14.csv, 15.csv, 16.csv, CT.csv.

File Source and Location

Ice beacon files can be located on `Gitlab` under CanWIN –> Ice Beacon Use Case –> Ice Beacon Data –> Baysys –> Processed

Dataset Variables

Table 1.3: Variables in intermediate data file

Header	Description	Data Type	Range or Expected Values	Units
Beacon_ID	Ice beacon identification number	Index	N/A	None
Latitude	North-south directing lines of parallels	Rational	(0-90)	Degrees
Longitude	East-west directing lines of meridians	Rational	(0-180)	Degrees
Timestamp	Timestamp at the start of collection	Datetime	N/A	None
Battery Voltage	Operating battery voltage of the ice beacon	Rational	Range	Volts (V)
Internal Temperature	Internal temperature of the ice beacon	Rational	Range	Degrees Celcius
Calc_distance	Distance calculated between two points of data using latitude and longitude	Rational	Range	Metres
Calc_speed	Speed calculated between two points using the difference between the distance calculated in calc_distance() and timestamp	Rational	Range	Metres/Second

Chapter 2

Reference Tables

2.1 Data Levels

Level 0 – Raw data: unprocessed data and data products that have not undergone quality control. Depending on the data type and data transmission system, raw data may be available within seconds or minutes after real-time. Examples include real-time precipitation, streamflow, and water quality measurements

Level 0.1 – First pass QC: A first quality control pass has been performed to remove out of range and obviously erroneous values. These values are deleted from the record. E.g: Online Environment Canada stream-flow data, laboratory data

Level 1 – Quality Controlled Data: Data that have passed quality assurance procedures such as Level 0.1 and have been further quality controlled by data provider before being submitted to CanWIN (e.g. Idronaut data with only downwelling (upwelling data removed) data included.

Level 1.5 – Advanced Quality Controlled Data: Data have undergone complete data provenance (i.e. standardized) in CanWIN. Metadata includes links to protocols and methods, sample collection details, incorporates CanWIN's or another standardized vocabulary, and has analytical units standardized. Note: Process still under development in CanWIN (as of May 13, 2020).

Level 2 – Derived Products: Derived products require scientific and technical interpretation and can include multiple data types. E.g.: watershed average stream runoff derived from stream-flow gauges using an interpolation procedure.

Level 3 – Interpreted Products: These products require researcher (PI) driven analysis and interpretation and/or model-based interpretation using other data and/or strong prior assumptions. E.g.: watershed average stream runoff and flow using streamflow gauges and radarsat imagery

Level 4 – Knowledge Products: These products require researcher (PI) driven scientific interpretation and multidisciplinary data integration and include model-based interpretation using other data and/or strong prior assumptions. E.g.: watershed average nutrient runoff concentrations derived from the combination of stream-flow gauges and nutrient values.

2.2 Result Value Qualifiers

ADL - Above Detection Limit

BDL - Below Detection Limit

FD - Field Duplicate

LD - Lab Duplicate

$ - Incorrect sample container

EFAI - Equipment failure, sample lost

FEF - Field equipment failed

FEQ - Field Equipment Questionable

FFB - Failed. Field blank not acceptable

FFD - Failed. Field Duplicate

FFS - Failed. Field spike not acceptable

H - Holding time exceeded

ISP - Improper sample preservation

ITNA - Incubation time not attained

ITNM - Incubation temperature not maintained

JCW - Sample container damaged, sample lost

NaN - Value is missing and reason is not known

NC - Not collected

ND - Not detected

NR - Sample taken/measured on site but information in this field not recorded

NS - Sample collected but not submitted

OC - Master Coordinate List Used

P - Analysis requested and result pending

prob_good - probably good value. Data value that is probably consistent with real phenomena but this is unconfirmed or data value forming part of a malfunction that is considered too small to affect the overall quality of the data object of which it is a part

prob_bad - probably bad value. Data value recognised as unusual during quality control that forms part of a feature that is probably inconsistent with real phenomena

Interpolated - This value has been derived by interpolation from other values in the data object

Q - Below limit of quantification (LOQ). The value was below the LOQ of the analytical method. The value in the result field is the limit of quantification (limit of detection) for the method

Chapter 3

Options and Packages

3.1 Python

3.1.1 Python Script-Specific Options

- **os** -os is a Python module, part of the standard library on Python. Therefore, it comes with Python but still needs to be imported.
- **Configparser** - Configparser is a Python class which implements a basic configuration language for Python programs. It provides a structure similar to Microsoft Windows INI files. ConfigParser allows to write Python programs which can be customized by end users easily.
- **pandas** - Pandas is another Python module that allows the user to import CSV (comma seperated values) files. Importing pandas using the pd prefix avoids overlap with other Python tools.
- **datetime** - Datetime is the Python module containing: date, time, and datetime.
- **math** - m is a mathematical calculations module of Python.
- **lonlat, distance** - Importing lonlat, distance from geopy.distance takes x-longitude, y-latitude, and optionally, z-altitude values and gives you the coordinates as a point.

3.1.2 Python Packages

Visit `https://docs.python.org/3/library/` to learn more about python packages

- **DateTime** - Date/time values
- **geopy** - Uses geocoding to locate addresses, cities, towns, etc.

Example: Section 2.1 from Victory's semi-hemi codebook.

Glossary

Data Management Workflow 1 - one of three data management workflows used at CEOS. Workflow 1 is used for data that is managed directly by CEOS from instrument collection to data sharing. 1