Manual for the TOADSuite for the Acoustic Tracking of Animals

Acoustic and Functional Ecology Group, MPI for Ornithology, Seewiesen, Germany
Holger R. Goerlitz (hgoerlitz@orn.mpg.de)

The TOADSuite is a software toolbox **to calculate the three-dimensional position of a sound source** ("acoustic tracking"). To do this, the same sound (e.g., a bat call) is recorded on an **array of multiple microphones** (at least four) whose relative positions must be precisely known. The differences in arrival time (**TOAD**: time-of-arrival-difference) of the sound between the different microphones is calculated by cross-correlation. These TOADs are converted into 3D-positions based on the speed of sound. Since speed of sound depends on air temperature and relative humidity, also **weather data** need to be measured during acoustic tracking. The TOADSuite also analyses the **acoustic parameters** of the recorded calls, and as emitted at the bat's position. Lastly, the TOADSuite allows to **visualize the recorded calls** and their positions for error checking and **reconstruction of flight trajectories**.

© **TOADSuite**: Dr. Peter Stilz, Dr. Jens Koblitz, Dr. Holger R. Goerlitz. The TOADSuite is developed and maintained by Dr. Peter Stilz, Dr. Jens Koblitz and Dr. Holger R. Goerlitz. The largest part of the code is developed by Peter Stilz, based on code by Jens Koblitz, with additional functionality by Holger Goerlitz.
© **TOADSuite Manual**: Dr. Holger R. Goerlitz. The manual is written and maintained by Holger Goerlitz.
Usage of the TOADSuite and any parts of it is only allowed in agreement and collaboration with Holger Goerlitz, Jens Koblitz and/or Peter Stilz.

Table of Contents

Future topics to be included into this manual:

- How to calibrate microphones (2018)
- Mergearrays incl. figure legend (see email; 2018?)
- Results of the calibration of the TOADSuite (2018)

Change log of the manual:

- V 0.9.6, 9.8.2017: corrected information about AGM-file location and configuration
- V 0.9.7, Oct. 2017: new: position and orientation in mic-AGM files / mic calibration / upper-case WAV
- V 0.9.10, Nov. 2017: new: updated info about mic-AGM files and call analysis.
- V 0.9.11, Nov 17 – Jan 18: new: info on potential problems with date/time format in weather CSV and potential solution; added chapter on Solving Problems; added info about call analysis, call data and XLS files.
- V 0.9.12, Feb 18: description of final version of smalltoadcallanalysis added, incl. all errors and solutions that came up during the analysis of the Tuebingen calibration data.
- V 0.9.13, 26.03.18: Reference distance for SL added (1 m)
- V 0.9.14, 16.05.18: Corrected information about XIV and CIV added
- V 0.9.15, 08.09.19: Added exact information about the structure of the weather CSV file

1 Definitions

We use the TOADSuite program to analyse sound data, which were recorded with microphone arrays to your PC, and associated data like weather and other. For everything to work, a range of settings, definitions and naming conventions is required. Those are detailed here. Read them **carefully** and adhere to them **strictly**, the TOADSuite will not work, and/or you will lose loads of time for error checking, or potentially even data.

1.1. Program and Data

The TOADSuite is a **Program**, which is used to analyse **Data** (= sound files and all associated data files). Despite that, the TOADSuite obviously also consists of files on your computer. Thus, we clearly distinguish between the **program files** which you keep in the **program folder "TOADSuite"**, and the data files which you keep in the data folders of your research project (specifically, in a **subfolder called "actrackdata"**). Realize this difference between program and data.

In contrast to standard programs that are started by double-clicking on their icon (e.g., Word, Excel), the TOADSuite is running within another program: Matlab, a powerful programming environment. In Matlab, you run and control the TOADSuite by entering the TOADSuite commands, for example to analyse your sound files, or to open the graphical user interface (GUI) for reconstructing trajectories.
Like any other program, the TOADSuite has a range of **configurations** (= settings/preferences) that you can adjust. In contrast to standard programs, however, these configurations are not adjusted by opening a menu and doing some clicking, but by opening, changing and then saving again the configuration files of the program. Don't be confused by this difference in HOW you adjust the configurations, the effect is the same: you are changing the configuration of the program, which will then affect how the program is processing your data. But you are not adjusting the data. (On a side note: these configuration files are stored in the program folder, this is the default configuration. However, copies of the configuration files will be copied to your data folder: this is the specific configuration for this data set. We'll come to that later; just be aware of that and don't be confused.)

1.2. Coordinate System and Microphone Arrays

The purpose of the TOADSuite is to obtain the **position of a sound-emitting animal in three-dimensional space**. Thus, we need a coordinate system (CS) to represent the positions. We use a Cartesian CS (i.e., X, Y, Z-coordinates) with the following definitions:

- We use a right-handed CS with X/Y-plane being the ground, Z is upwards.
- The coordinates of the mics (receiver) in the array are defined as follows: Mic0 is the central mic with the coordinates 0/0/0. The other mic positions are arranged around the central mic in the X/Z-plane and are directed towards the positive Y-axis (**Fig. 1**). This is our **standard vertical orientation**. **Fig. 2** shows all our mic arrays and our numbering of the mics.
- In the field, the mic array is placed into the real-world CS. It has a specific height above ground (h1) and it can be rotated backwards to change its elevation angle (counter-clockwise around X: α) and horizontally to change its azimuth angle (counter-clockwise around Z: β; see **Fig. 3**).
- The origin (0/0/0) of the real world CS is thus defined to be on the ground exactly below Mic0 of the array, mic0 then has coordinates (0/0/h1).
- h1 and α are measured exactly in the field, and β is estimated in the field. h1 and α are required for correct localisation of the bats relative to the ground; β is only used to orient the bat trajectories relative to a potentially interesting linear structure in the field (river, path, hedge row,…; see **Fig. 3**).

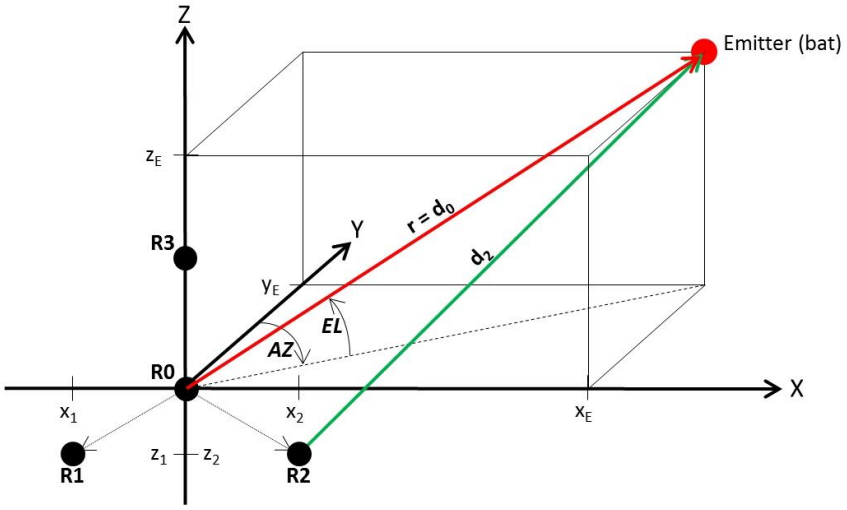

Fig. 1: Orientation of a 4-receiver-array (mics R0-R3) in the mic-centered CS. The array is facing towards the positive Y-axis, Z-axis is upwards.

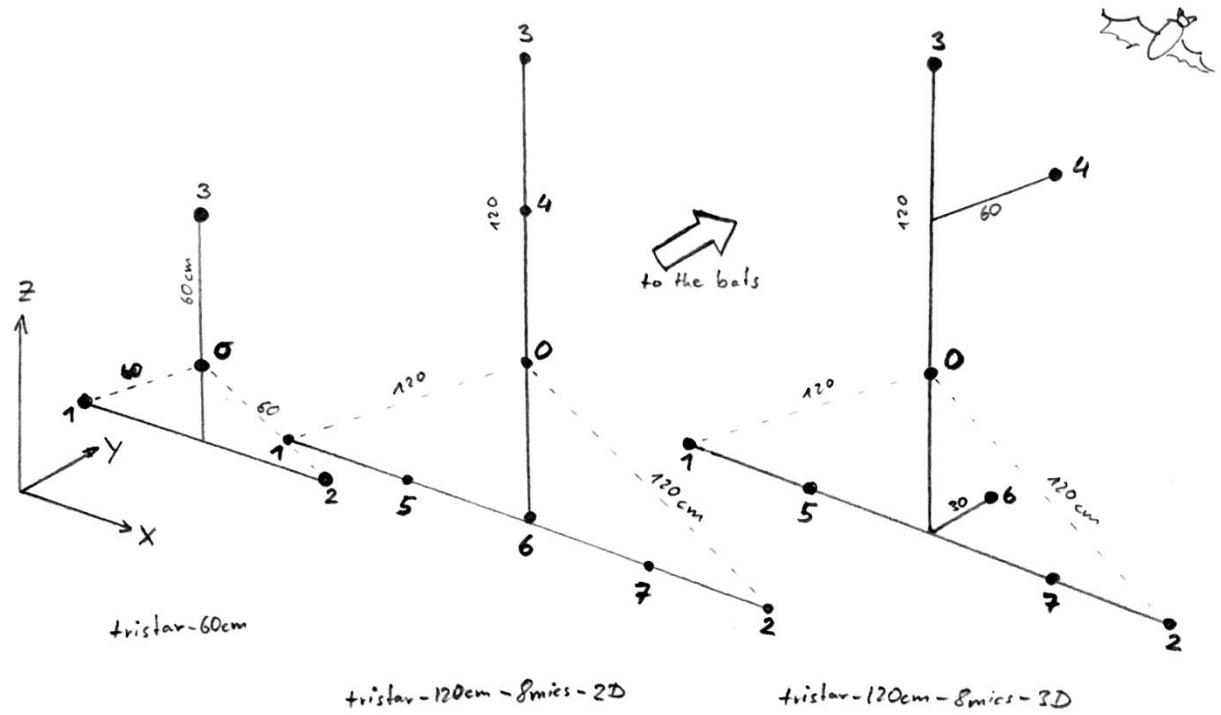

Fig. 2: Position and numbering scheme of the microphones for all arrays. Mic0 is the central microphone, numbers 1-3 are used for the outer microphones in a counter-clockwise fashion when looking from behind the array.

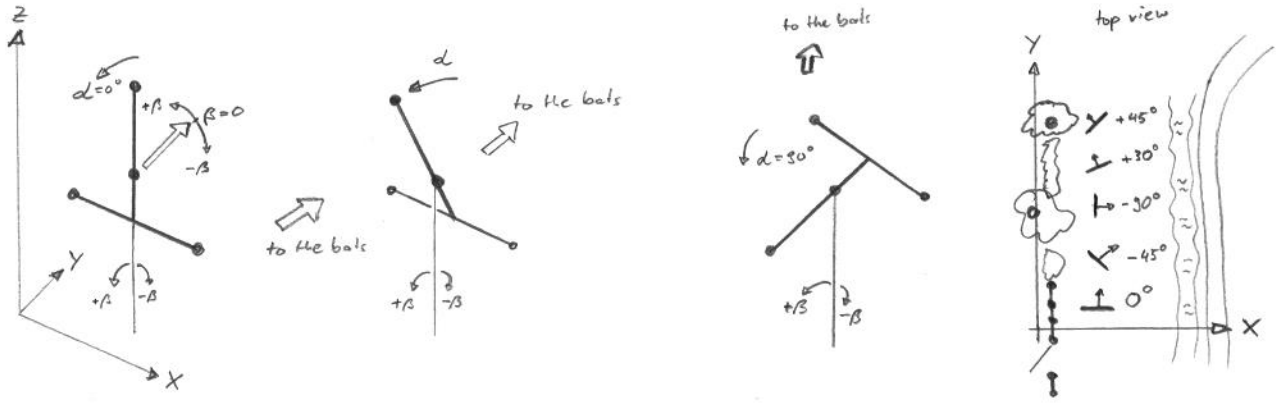

Fig. 3: Angles α and β for orientation of the array in the field. For α = 0°, the array is oriented vertically and facing forward. When rotating the array backwards, α increases to up to 90°, where the array is horizontal and facing upwards. β is the angle that the array is rotated towards an arbitrary linear horizontal structure. β is positive when the array is rotated to the left (counter-clockwise around Z) and positive when the array is rotated to the right (clockwise around Z). See the top view on the right showing exemplary β values for the array oriented differently relative to the environment with trees, hedge, fence, river and street.

- Definition of α: α is the elevation angle, defined as rotation counter-clockwise around the X-axis. α = 0° when the array is in the standard vertical orientation, and α = 90° when it is rotated 90° backwards facing upwards (standard horizontal orientation). α can take values between 0 and 90° (**Fig. 3**).
- Definition of β: β is the azimuth angle for rotations counter-clockwise around Z, relative to any arbitrary linear structure in the field that you want to use as reference, e.g. a hedge row or path. β = 0°: the array is facing parallel to that structure (the array's Y-axis is parallel to that structure). β = positive = rotation to the left (array facing the structure), β = negative = rotation to the right (array facing away from the structure; **Fig. 3**).

We use different microphone arrays, with at least 4 microphones. The geometry of the array, i.e., the microphones coordinates, needs to be exactly known for acoustic tracking. They are saved in ***_agm.csv-files** stored in the global-folder of the TOADSuite (see next section "Microphone coordinates: AGM-files" below). The *_agm.csv files contain the RELATIVE array geometry, not the absolute coordinates in the field.
Based on the angle and height information in the tracking list (see section "Tracking list" below), the mic positions are automatically rotated and translated by the TOADSuite to obtain the mic positions as in the field.

!!! Ensure that the recorded wav-files have the correct mic-number (i.e., that cables from mics to sound card are correctly connected)!

1.3. Microphone coordinates and orientation: AGM-files

The relative **positions** (**coordinates: X, Y, Z**) of all microphones of an array and potentially each microphone's **orientation (= pointing direction: azimuth, elevation, phi)** is stored in so-called AGM-files (ARRAY_NAME_agm.csv). The AGM-file has one row per microphone, with columns containing mic number, X, Y, Z, azimuth, elevation, phi (Table 1).

Table 1: Structure of the AGM-files (ARRAY_NAME_agm.csv) containing the position and orientation of each microphone in an array.

Mic #	X	Y	Z	AZ	EL	phi
0	x0	y0	z1	az1	el1	phi1
1	x1	y1	z2	az2	el2	phi2
2	x2	y2	z3	az3	el3	phi3
3	x3	y3	z4	az4	el4	phi4
…	…	…	…	…	…	…

X, Y, and Z are the position of the microphone in the array. Azimuth, Elevation and phi describe the direction into which the microphone is pointing (**Fig. 4**):

- X, Y, Z: **Cartesian coordinates** (=position) of the microphone in the array
- AZ: **Azimuth**: Angle relative to the positive x-axis
- EL: **Elevation**: Angle relative to the X-Y-plane
- Phi: **Rotation**: rotation angle around the microphone axis, starting at top (= positive Z)

All angles are given in **mathematical positive rotation**[1] in degrees [0-360°].

The standard orientation is the only one, which we use so far in all of our arrays: the microphone is pointing into the positive Y-axis, i.e. it has the orientation: (0 / 0 / 90).

Note: When a call analysis is performed, then the microphone direction is required in the AGM-file to correct for the microphone characteristics (see next section, „Microphone calibration files“)! If no direction is present, the TOADSuite just assumes the (wrong!) direction (0/0/0). If only bat positions are calculated, without call analysis, the direction information could be omitted from the AGM file.

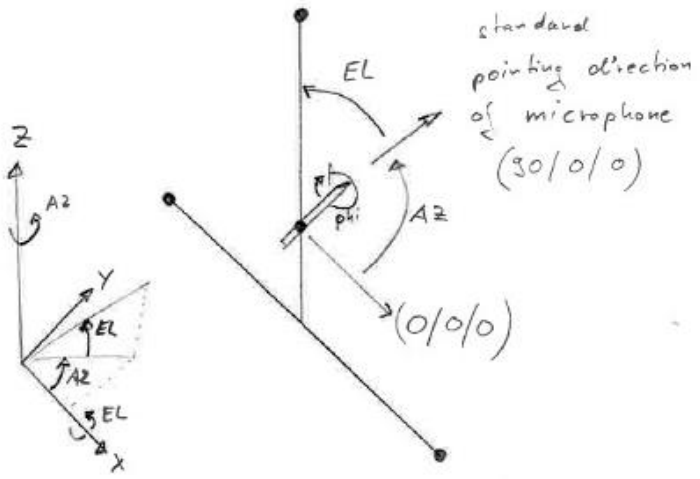

Fig. 4: Definition of the microphone direction.

[1] **Mathematical positive rotation** is defined as: clockwise-rotation when looking into the direction of the vector around which the rotation takes place; or counter-clockwise rotation when looking against the direction of the vector around which the rotation takes place.
Use the “right-hand rule to remember: your thumb points into the direction of the vector. Then your bend fingers give the rotation direction.
It can also be remembered that a mathematical positive rotation transfers one axis on the shortest way onto the next axis. I.e., in a rotation matrix, the first rotation (azimuth) transfers the positive X-axis onto the positive Y-axis on the shortest possible way. The second rotation (elevation) transfers the positive Y-axis onto the positive Z-axis on the shortest possible way. The third rotation (phi) is then another last positive rotation around the current axis.

1.4. Microphone calibration information

To correctly calculate the call frequencies and the call level (source level) as emitted by the bat, the **sensitivity** of the microphone and the whole recording system as well as its **frequency response** must be known. The TOADSuite is able to correct the received call for the frequency response of the microphone, the distance- and frequency-dependent atmospheric attenuation, and the distance-dependent geometric attenuation, to obtain the call as emitted by the bat into the direction of the array.
Therefore, microphones and recording system must be calibrated (for details, see chapter "Microphone Calibration").

Two sets of calibration information are required:

1. **Overall sensitivity of the recording system:**
   - The values are directly set in the `smalltoadcallanalysis.m` code file.
   - The following sensitivities[2] can be set (**Fig. 5**):
     - **`SENS_MIC`** (dB re. 1 V/1 Pa): Microphone sensitivity: how many volts are generated per incoming Pascal of sound pressure, expressed in dB relative to 1 V / 1 Pa (6 dB = 2 V generated / 1 Pa).
     - **`SENS_AMP`** (dB re. 1 V out / 1 V in): Amplifier gain: how many volts out are generated per incoming Volt in, expressed in dB (6 dB = 2 V out / 1 V in).
     - **`SENS_DIG`** (dB re. 1 FS / 1 V in): Digitizer gain: how many volts in correspond to full scale, expressed in dB (6 dB = 1 FS / 0.5 V in = 2 FS / 1 V in)
     - **`SENS_COR`** (dB gain correction): any further corrections in dB.
   - All four sensitivity values add up (dB-scale). If individual values are unknown, they can thus be combined and given as one single value (e.g., `SENS_ALL` could be defined to describe how many incoming Pascal of sound pressure relate to full scale in the WAV-file [dB re. 1 FS / 1 Pa].
   - `SENS_P_REF`: This is the reference sound pressure for the calculating the sound pressure level. Standard reference pressure in air is 20 µPa (= $2*10^{-5}$ Pa = 0.00002 Pa).

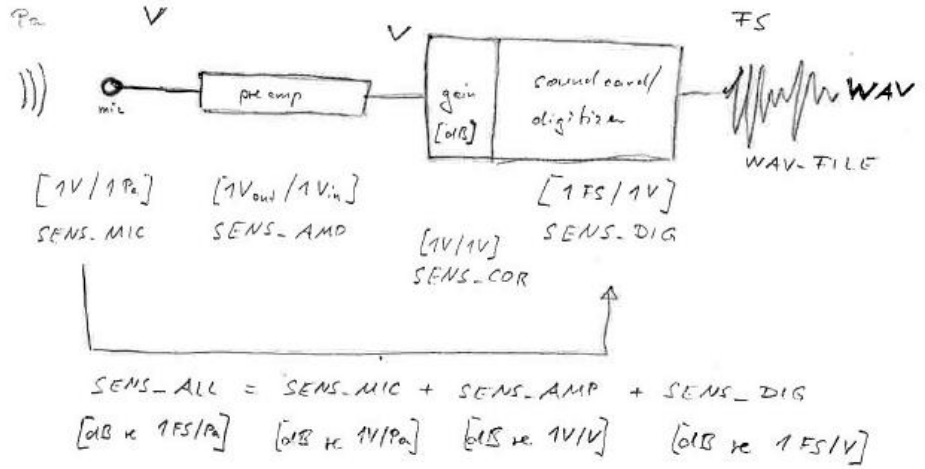

Fig. 5: Definition of sensitivity settings required for the call analysis.

[2] **Sensitivity = output per input**. Higher sensitivity = higher output per same input. Sensitivity is given in dB relative to a standard sensitivity. For example, if the standard is 1 V output per 1 Pa input, then a microphone that only generates 0.1 V output per 1 P input is 20 dB less sensitive, its sensitivity is -20 dB re. 1 V / 1 Pa. On the other hand, if the microphone generates 2 V output per 1 Pa input, then it is 6 dB more sensitive: 6 dB re. 1 V / 1 Pa.

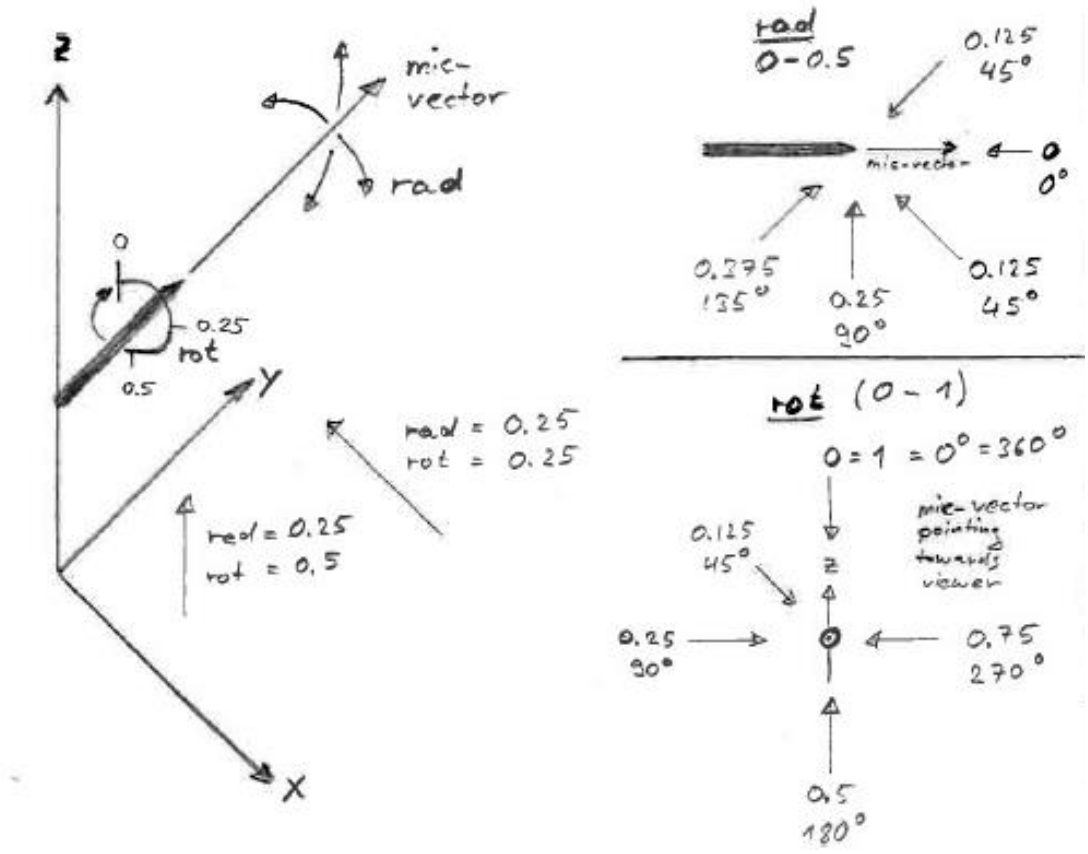

Fig. 6: Definition of the angles to describe the direction of sound impinging on the microphone.

2. **Direction-dependent microphone frequency response** and **relative sensitivity:**
   - Stored as a microphone calibration .mat-file (any name possible)
   - in the global folder of the TOADSuite
   - containing the structure "`mcmstruct`" with four fields:
     - `mcmstruct.freq` frequency vector [Hz], length N
     - `mcmstruct.rad` radial angle = off-axis-angle [norm. angle], vector, length M
     - `mcmstruct.rot` rotation-angle around mic-axis [norm. angle], vector, length L
     - `mcmstruct.A` relative sensitivity in negative dB, matrix, N x M x L
   - `rad` and `rot` are given in normalized angles from 0 to 1, 1 = 360° (**Fig. 6**)
   - rad = 0 is on-axis of the microphone, rad = 0.25 is 90° to the side, rad = 0.5 is behind the mic
   - rot = 0 is microphone top, from there we turn in mathematical positive rotation

This calibration information for each microphone and the recording system is obtained by calibrating the recording system against a known sound source. For details, see chapter "Microphone Calibration".

NOTE: only include frequencies into the calibration file that are sensible. During call analysis, the compensation for attenuation and microphone characteristics will only be conducted up to the highest frequency in the calibration files. All frequencies above the highest frequency in the calibration file will be completely filtered out.
When sampling at high sampling rates (500 kHz), this allows to exclude very high frequencies, which would be problematic when compensating for atmospheric attenuation.

1.5. Source Level

Reference distance for all source levels is **1 m distance** to the bat's mouth. To convert to 10 cm to the bat's mouth, add 20 dB (= 20*log10(1/0.1)).

1.6. Tracking list

The tracking list is a file where some basic information about each recording session is stored (date, orientation of the array in the field); the file is called **tracking_list_YYYY.xls** (substitute YYYY by the year) and must be in the **meta-folder. Don't use file extension *.xlsx.** A template is in the template folder of the TOADSuite.
Additional information, such as location, species, and comments, can also be entered.

There is also a tracking_list_YYYY.csv in the template folder, which is not used at the moment (but its use is implemented in the TOADSuite and can be activated by a switch, see Chapter on available switches).

1.7. Naming scheme

All **data files and folders** are named following a fixed scheme that you must exactly followed so that the TOADSuite will work. File and folder names consist of certain blocks; blocks are separated by underscore, values within a block are separated by dashes. Naming blocks, with format and example, are:

- [Session-Date]: date of the recording session (does NOT change after midnight)
  Format: `YYYY-MM-DD` example: `2016-08-26`
- [Session-Number]: 3-digit number of the recording sessions, multiple sessions per day possible
  Format: `SSS` example: `001`
- [Rec-Date]: date of the recording (changes from BEFORE to AFTER midnight)
  Format: `YYYY-MM-DD` example: `2016-08-27`
- [Rec-Time]: time of the recording
  Format: `HH-MM-SS` example: `01-11-57`
- [Rec-Number]: a unique 7-digit recording number for each recording
  Format: `NNNNNNN` example: `00000361`
- [TRJ-Number]: 2-digit trajectory number (within a given sound recording)
  Format: `NN` example: `01`
- [Mic-Number]: 2-digit number of the microphone, from 0 to N-1 (N = total number of mics)
  Format: `NN` example: `01`
- Mic…: keyword for single-channel-WAV-files
- ACTxx…: keyword for multi-channel-WAV-files
- weather…: keyword for weather-CSV-data

1.8. Data-Folder and Data Structure

To make the **data folder structure,** make a **Project Folder** for your research project. In this folder, make the acoustic tracking data folder named **actrackdata**, with the subfolders **fieldnotes**, **meta**, **wav**, and **weather** (**Fig. 7**). Note that the TOADSuite will generate additional subfolders: analysis, config, pos, tmp.

Info about the folders (Fig. 7):

- fieldnotes: store scans of your field notes
- meta: store meta data: currently only the tracking_list_[SessionYear].xls
- wav: store the sound recordings in one subfolder per recording session.
- weather: store the weather data: weather_[SessionDate]_[SessionNumber].csv

- analysis: will be generated by the TOADSuite, contains intermediate analysis results in one subfolder per recording session.
- config: will be generated by the TOADSuite, contains the analysis configuration in one subfolder per recording session.
- pos: will be generated by the TOADSuite, contains the final data in one file ("T-file") for each trajectory in one subfolder per recording session.
- tmp: will be generated by the TOADSuite, contains temporary data that may be deleted.

When you collect sound data in the field, we speak of a **RECORDING SESSION**, identified by the **session date** (=the date at the start of the session; the session date does not change from before to after midnight. Within a recording session, you might thus have recording dates of two successive days) and the **session number**, which is simply a number for each session of that date.

All data is therefore organized in **Session Folders.** Session folders are subfolders within the folders wav, analysis and pos, containing the sound recordings, the intermediate analysis results, and the final positions, respectively, of a recording session.

- Name format of the session folders: [SessionDate]_[SessionNumber]: **`YYYY-MM-DD_SSS`**
- Example: **`2016-08-27_001`**
- Note that the year, month and day are separated by a dash, and the session is separated by an underscore.
- It is possible to record multiple sessions per day, e.g., at different locations, or with the array in different positions or orientations, or with a different gain setting of your microphone amplifiers, etc. Within a single session, nothing changes and you simply record one WAV-file after the other. If you change something, you start a new session.

!!! NOTE on **SESSION DATE** and **RECORDING DATE**: Since bats are nocturnal, a recording session can last until after midnight, meaning that the date changes! Therefore, we distinguish between **session date** (this is the date of the session folder) and **recording date** (this is the date of the WAV-file, which can be the same as the session date, or one day later).

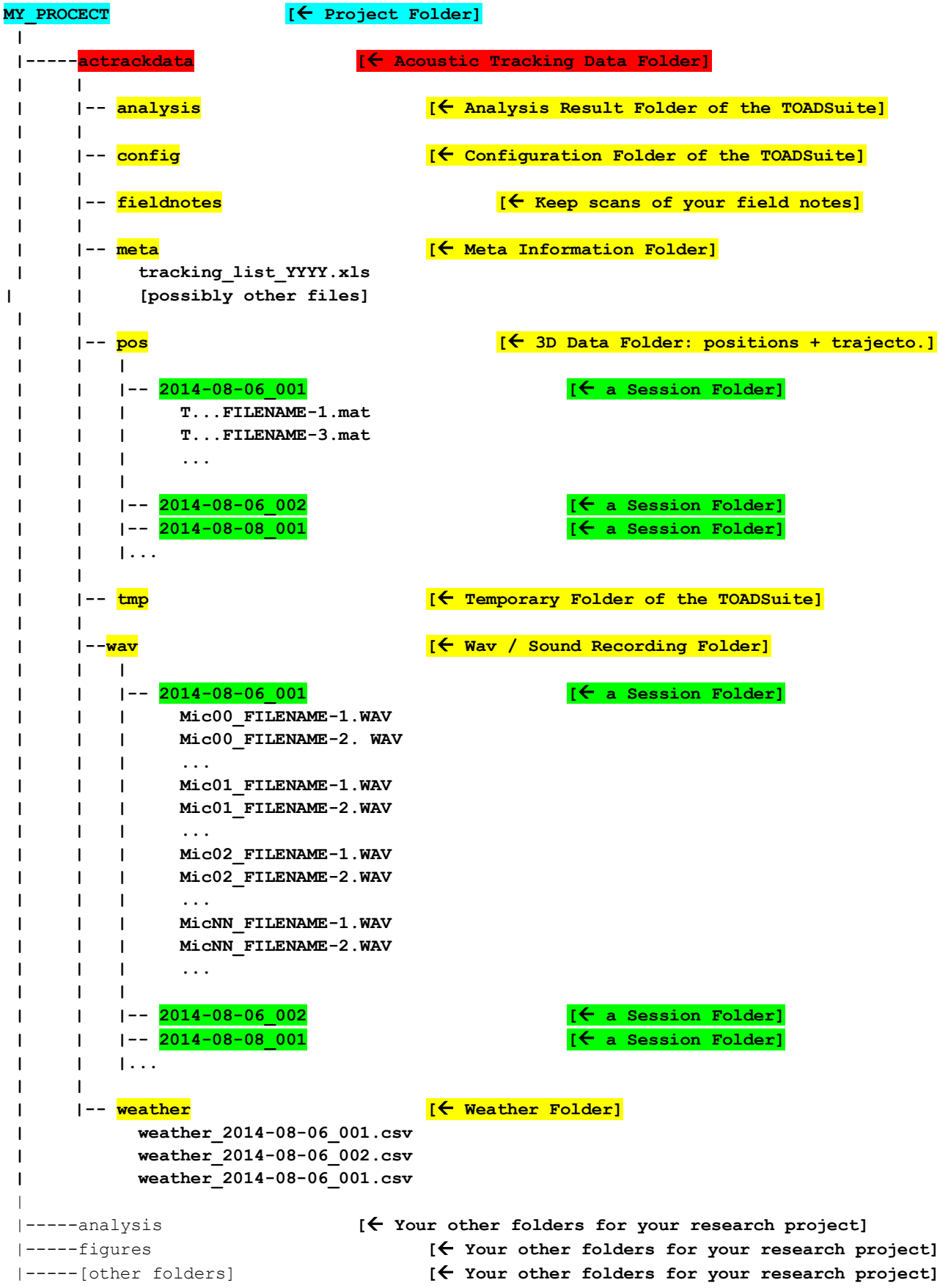

Fig. 7: Folder and file structure required by the TOADsuite.

1.9. File Names

The folder structure was introduced above. Like the folders, our files need a determined file name structure, using the naming blocks introduced above. Generally, our files are identified by the **session date** and **session number** and a **unique recording number**.

!!! Take care that you use EXACTLY the required format, including correct number of letters/digits and upper/lower case.

- **MULTI-CHANNEL SOUND RECORDINGS (WAV-files):**
  - Multi-channel WAV-files are only recorded with Avisoft Recorder Software and are later split into single-channel WAV-files (see below for their structure)
  - **Structure**: ACTxx_[Rec-Date]_[Rec-Time]_[Rec-Number][_...].WAV
  - **Structure:** `ACTxx_YYYY-MM-DD_HH-MM-SS_NNNNNNN[_...].WAV`
  - **Example:** `ACTxx_2016-08-27_21-24-02_0000256[_...].WAV`
  - In addition to the naming blocks keyword "ACTxx", **recording** date, recording time, and unique file number, potential additional fields can be added, separated by underscores.
  - **Upper-case** file extension (**.WAV**). (**Note**: Avisoft Recorder changed the WAV-ending from uppercase to lowercase in early 2017. The original multi-channel recordings saved by Avisoft Recorder will thus have lower case. Our function splittoadwavs will then generate single-channel files with uppercase WAV. If you use other recording systems, ensure that WAV-files have an upper-case ending. The uppercase ending is mostly relevant for compatibility reasons with Peter's system.)

- **SINGLE-CHANNEL SOUND RECORDINGS (WAV-files):**
  - Single-channel WAV-files are the sound recording format used by the TOADSuite.
  - **Structure:** Mic[MicNumber]_[Rec-Date]_[Rec-Time]_[Rec-Number][_...].WAV
  - **Structure:** `MicNN_YYYY-MM-DD_HH-MM-SS_NNNNNNN[_...].WAV`
  - **Example:** `Mic00_2016-08-27_21-24-02_0000256[_...].WAV`
  - In addition to the naming blocks keyword "MicNN", **recording** date, recording time, and unique file number, potential additional fields can be added, separated by underscores.
  - Upper-case file extension (.WAV).

- **WEATHER DATA (CSV-files):**
  - The logged weather (T, RH) is stored in one CSV file per recording session.
  - Structure: weather_[Session-Date]_[Session-Number].csv
  - Structure: `weather_YYYY-MM-DD_SSS.csv`
  - **Example:** `weather_2016-08-27_001.csv`
  - File name contains the key word "weather", the **session** date and the session number.
  - Lower case file extension (.csv).

- **TRACKING_LIST (XLS-file):**
  - One file for each year of the whole research project, containing some basic information about each recording session.
  - **Structure**: tracking_list_[year].xls
  - **Structure**: `tracking_list_YYYY.xls`
  - **Example:** `tracking_list_2016.xls`
  - File name contains the key word "tracking_list" and the year (four digits)
  - Lower case file extension (.xls). **Don't** use the extension **.xlsx!**
  - There is a template for this file in the templates folder of the TOADSuite

- **TRAJECTORY FILES / "T-files" (MAT-files) :**
  - **Final data files** for each trajectory, containing the raw positions and times, interpolated positions, interpolation formula, meta data and acoustic call data.
  - Note: **one T-file is stored for each trajectory**. Thus, there can be multiple T-files per recording (if multiple trajectories/bats where in the recording); or no T-file (if no good trajectories where in the recording).
  - **Structure**: T[Rec-Number]__[TRJ-Number]__[Rec-Date]_[Rec-Time][_...].mat
  - **Structure**: `TNNNNNNN__NN__YYYY-MM-DD_HH-MM-SS_NNNNNNN[_...].mat`
  - **Example**: `T0000256__01__2016-08-27_21-24-02_0000256[_...].mat`
  - File name contains the keyletter "T", followed by the unique recording number and the trajectory number, followed by the original file name of the recording (Rec-data, rec-time, rec-number, and any other potentially added fields).
  - Lower-case file extension (.mat)
  - T-files and their content are described in a separate chapter later.

- **OTHER FILES** such as the output files of the TOADSuite use an analogue scheme based on the WAV-files. As a user, however, you do not interact with these files.

- **MICROPHONE ARRAYS (CSV-files):**
  - The coordinates of the microphone in the microphone-centered CS are stored in one CSV file per array. The files end with …-agm.csv and are otherwise a descriptive name for the array. They are in the global-folder of the TOADSuite.

1.10. a comment on CSV-files

CSV-files (*.csv: comma-separated files) are used for different data, with DIFFERENT field separators (FS):

- Coordinates of the microphone arrays: semicolon as FS
- TOADSuite output files: semicolon as FS
- Weather data: comma as FS
- [Tracking_list_YYY: comma as FS]

WARNING: In general, using the FS within the fields of a CSV messes up the data structure of the CSV file and makes it unreadable. E.g., any additional commas in a CSV-file with commas as FS indicates a new data cell.
Thus, when using the tracking_listYYYY.CSV (instead of .XLS) NEVER input the commas in the cells. We try to minimize potential problem by entering text only in the last columns of the tracking list, which are ignored by the TOADSuite. So, any commas you may enter in the location, species and comment columns will be ignored. Likewise, wrong comma will mess up the weather data CSV, while wrong semicolons will mess up the TOADSuite output files.

2 Microphone Calibration for call analysis

[How to calibrate microphones and recording system. TBA]

3 Installation

1. Install/Use Matlab R2007b ("for all" folder on the server) and the Kestrel Communicator (server).
2. Make a local copy of the TOADSuite on your computer:
   - make a folder called **TOADSuite** where you want to store the program TOADSuite This is the folder for the software (NOT for the data!), referred to as TOADSuite-folder.
   - On our server, go to the current release of the TOADSuite (for all/ software/AcTrack_current_release) and copy the following folders to your TOADSuite folder: **bin, documentation, fx, global, templates, tmp**.

   Folderinformation:

   - bin: contains all code required to run the TOADSuite.
   - documentation: Manual and Documentation of the TOADSuite.
   - fx: contains additional code. If needed, copy it to bin.
   - global: contains global configuration files and array coordinate files (AGM).
   - templates: contains several templates: tracking_list_YYYY.xlx, field_notes,…
   - tmp: temporary folder where the TOADSuite will store temporary data.
   - [code_Peter: contains the TOADSuite code of Peter Stilz (don't copy).]
   - [support_code_HRG: contains additional code by Holger Goerlitz (don't copy).]

3. Copy **`isposintscalaR.m`** from **fx** to **bin** (this file seems to be required for our Matlab version).

   [Info: The fx folder contains other files that might be required by certain Matlab versions. If the TOADSuite throws an error because a specific file is missing, copy it from fx to bin.]

4. Open **`toadbaseconfig.m`** (bin-folder of the TOADSuite) and:
   - set the GLOBALPATH and TMPPATH to the global- and tmp-folder of your TOADSuite. Ensure that they **end with a backslash**.
   - check that the following entry is set: `CHANNELFORMAT='Mic%02.0f'`

5. Open **`toadconfig.m`** (global-folder of the TOADSuite) and set the following entries:
   - `SPG_CHANNELS = [0:N-1];  % # of mics to show as spectrogram`
   - where N is the number of the microphones (i.e., [0:3] for 4 mics, [0:7] for 8 mics) you use.
   - This setting determines which channels are shown as external spectrogram in the chktoadpos-GUI. Less mics than the maximum number is possible if not all channels shall be displayed.
   - NOTE: when using a different number of mics, `toadconfig.m` needs to be adapted!

6. Open **`sessionconfig.m`** (global-folder of the TOADSuite) and ensure that the following entries are set:
   - `VPOSORD = [0 1 1];  % vector for bat position order`
   - `CH_CDT = 0;  % channel # for click detection`
   - `TRACKINGLISTFORMAT_XLS = true;  % true: use XLS-file, false: use CSV-file`
   - `CHANNELS_READ = [0:N-1];  % # of mics read`
   - `CHANNELS_USE = [0:N-1];  % # of mics used for localization`
   - where N is the number of the microphones (i.e., [0:3] for 4 mics, [0:7] for 8 mics) that you most often use.
   - `CHANNELS_READ` and `CHANNELS_USE` determine the number of channels that are read and used for calculating the positions. A lower number than the maximum number of mics is possible if not all channels shall be used.
   - To be on the safe side, set `CHANNELS_READ` and `CHANNELS_USE` to the same value!
   - **NOTE**: if you use an array with a different number of microphones, `sessionconfig.m` needs to be adapted!
   - `DO_SMALLTOADCALLANALYIS = ...` : set it to "true" or "false" to activate or deactivate the call analysis
7. Open **`act.m`**, **`make_T_files_toadpos.m`**, **`write_T2xls.m`** and **`trj_viewer.m`** (bin-folder) and set your default project folder.
8. **Tracking_list:** Place a copy of the tracking list (`tracking_list_YYYY.xls`, from the template folder) into the meta-folder. Use one tracking list per year and change the last four digits of the file name to the respective year.
9. **Array coordinates:** If not in the global folder yet, make a new array coordinate file with the coordinates of the microphones in your array ("AGM-file") and give it a new, clear name (`ARRAY_NAME-agm.csv`) and place in the global folder of the TOADSuite. Compare with the other AGM-files in the global folder that your new AGM-file uses the correct formats and field separators! The TOADSuite will read your microphone coordinates from the AGM-file in its global folder.
10. Set your **Windows date format**:
    - Go to Start > Control Panel > Region and Language.
    - Set the Format to: English (United Kingdom) and set the Short Date to yyyy-MM-dd and short / long time to HH:mm / HH:mm:ss.
    - Open a weather CSV file in a text editor and check that the first column is displayed as "yyyy-MM-dd HH:MM:SS". Otherwise, the TOADSuite cannot read the weather data.

- Check that the 'List Separator' is set to a Comma (,): Click 'Additional settings…' on the bottom and check/set the List Separator.

NOTE on the date format, and potential ERROR solving:

The TOADSuite reads the weather data from the weather CSV file at the date and time of the bat call. Therefore, the TOADSuite must be able to read the date and time from the weather CSV file; and it expects that the date/time are in the format:"YYYY-MM-DD HH:MM:SS".

However, in the data files of our Kestrel weather loggers, the seconds in the timestamp are often not displayed (when opening the file in Excel). Sometimes, this might also result in the problem that the TOADSuite then cannot read the file. This is still unclear to me…

SOLUTION: If you have the problem that the TOADSuite (act(11) = starttoadpos) cannot read the weather file (→ check the Matlab error message: problem in datestr/datenum conversion?), try the following:

1. Open the weather CSV in Excel and manually set the format of the cells with date/time (in the first colum): mark the date-time-cells, right-click, select 'Format cells', select 'Custom' at the bottom of the list on the left, and type the full format in the box 'Type' on the right.

Another problem occurs in case there are additional EMPTY lines in the CSV file, after the weather data. To solve this, mark the rows below the data and delete them (backspace). Possible check in another text editor (not Excel) whether there are really no empty lines. (Note that Excel will always show empty cells, independent if they are contained in the CSV file or not).

4 Recording bats in the field

4.1. Overview:

- setting up the array in the field, measure and note its height and orientation, draw a sketch, check clocks of weather logger and PC.
- log weather data (Kestrel weather logger).
- record (not too long) sound files of bat activity.
- **Do not record too many files per session (less than 100)!!** Because multiple figures are generated per recording during analysis, Matlab crashes eventually if too many figures are open. Just start a new session if you want to record more files.

4.2. Logging weather data (Kestrel weather logger)

- Set up the logger early enough before starting the sound recordings, so that the logger can adjust to the ambient temperature conditions.
- Switch logger on/off: long press on red button.
- Data logging: set the data storage rate to **at least every 10 min** (or more often), and store at least **Temperature and relative humidity**. If possible, also record atmospheric pressure. Note that the logger records may only store 250 data points – so download them on the next day!
- **Standard logger settings**: record T (in °C), RH (in %) & pressure (in hPa). Correct time and date. Overwrite On (risk of losing previous data if not downloaded! But overwrite off does not allow to store new data if memory is full).
- Check that the clock of the logger is set to the correct time and date (required for synchronization with the sound recordings).
- CHECK before recording sound files that the weather logger is really storing data (→ Memory > Auto Store > ON)! Without weather data, recordings cannot be analyzed!
- You can switch off the logger during recording to save battery (it is still storing data if the auto store is set to ON).
- SWICH OFF auto-storage after the recording so that your data are not overwritten (→ Memory > Auto Store > OFF), and switch off the data logger (long press on red button).

  See 5.5 Workflow, step 2, about the required data structure of the weather file. If you use a different weather logger, make sure that you have the exactly identical data structure. Otherwise, manually arrange your data accordingly! **Required data**: date & time, T (°C), RH (%), pressure (hPa).

4.3. Field notes

- Fill out the field notes data sheet: date, session number, location, (expected) species, type of array (number of mics, type of mics, mic arrangement), array height and angles, gain settings, weather conditions, etc.
- Draw a sketch of your recording situation: the array's position and orientation relative to prominent features in the field (lake, road, path, trees, hedges, etc), your position, position of equipment, etc. Sketch the typical bat behavior and flight direction.

- Note relevant information, such as time of first bat appearance, time of first recording, recording number of calibration recordings, and anything else that seems interesting.
- A template for the field notes data sheet is in the TOADSuite folder.

4.4. Recording with Avisoft Recorder Software

4 Channels can be recorded with Avisoft Recorder Software and Avisoft USG SoundCards (USG 416H at 500 kHz), using Avisoft microphones (Knowles FG-O. The CM16/CMPA has a too high directionality).

!!! Note: record all channels into one single multi-channel WAV-file! Otherwise (when recording 4 separate WAV-files), synchronicity between channels is not ensured! Before running the TOADSuite, the multi-channel WAV-file needs to be split into 4 separate correctly named WAV-files (`act(2)`, `splittoadwavs.m`).

Setting the WAV-filename:

- Option > Configuration > Filenames
- Untick "Channel name"
- Enter Filename prefix: "ACTxx_"
- Tick "Date", "Time", "Track Number" (with 7 digits) and "convert space characters to underscore"
- "add last playback filenames and its start time": tick only when playbacks are presented.

Setting the recording configuration:

- Option > Configuration
- Ensure to tick "Save the active channels of each device into a multichannel file" at the bottom to record synchronized multi-channel WAV-files!
- Set all other details as needed (Device, active channels, pretrigger, trigger, base directory, sampling rate, etc.)

4.5. Recording with Matlab (avr.m)

- Use Matlab 2007b, set current directory to the folder with the recording software, type avr at prompt.
- Pressing any key should alternately start and stop a recording.
- Add settings of the Fireface?!

4.6. General comments on good recordings

- Listen to a bat detector while recording, and/or watch the display of incoming calls. Aim to start/stop recordings so that they cover a full bat pass.
- Ensure that the clock of the weather logger and the recording PC are set to the correct time. This is required for synchronization of the data.
- Using a pre-trigger of about 4 sec allows to catch the bat's activity before you hit the record button.
- 3D data become difficult to analyze when too many positions are within one recording. Thus, do not record for too long. Usually, a few seconds of constant bat activity are sufficient.
- Increase the gain as high as possible without obtaining too many clipped calls.
- Take as many notes as possible in the field on general activity, times, weather and location.

- Copy, store and pre-process your data in the same night or the next day after field work and have a first look at your data on the day after (Avisoft SASLab; or other sound software). Make additional notes if needed. Scan and store your field notes. Empty / low quality / unwanted recordings can be deleted at this point (WAV-files of ALL channels!).

5 Using the TOADsuite

5.1. OVERVIEW of the TOADSuite program files (m-files)

The TOADSuite is basically a set of Matlab files, mostly written by Peter Stilz, with some additional files by Holger Goerlitz. Most files are needed internally by the TOADSuite. The user runs the TOADSuite by calling the user functions at the Matlab prompt.

To run the TOADSuite, start Matlab 2007b, set Matlab's current folder to the TOADSuite bin folder (ONLY the bin folder! The other folders of the TOADSuite should not be on Matlab's search paths), and call a TOADSuite function at the prompt.

The user commands/functions are:

- **`act.m`** A wrapper command that allows to call all user functions via this single command. Just typing `act` provides a list for choosing between the user functions. Alternatively, type `act(N)` to call the user function with the number N directly. Or type **act(N,M,O,…)** to run multiple programs (N, M, O,…) one after the other.
- **`plottoadweather.m`** Plot recorded weather data, soundspeed and atmospheric attenuation for a recording session. **`act(1)`**
- **`splittoadwavs.m`** Split multi-channel-WAV-files into multiple single-channel-WAV-files for processing with the TOADSuite. **`act(2)`**
- **`resampletoadwavs.m`** Resample WAV-files to 500 kHz sampling rate. Needed for the WAV-files recorded with 192 kHz sampling rate (e.g, Fireface 802) **`act(3)`**
- **`starttoadpos.m`** A wrapper command to start the batch processing of WAV-files. It internally calles `mktoadpos.m` for each recording. **`act(11)`**
- **`mktoadpos.m`** Does the main processing of call detection, TOAD-calculation via crosscorrelation and position-calculation based on TOADs. **`act(12)`**
- **`chktoadpos.m`** Opens the GUI for viewing positions, manual checking, trajectory reconstruction, adding additional calls etc. **`act(13)`**
- **`maketoadsummary.m`** Combines all data into the summary CSV-file of the TOADSuite. **`act(14)`**
- **`write_T_files_toadpos.m`** Interpolate trajectories and summarize all data (incl. acoustic analysis) into our final T-file-MAT-format. For each trajectory, one T-file is written. If a WAV-recording contains multiple trajectories, multiple T-files are written for this recording. **`act(21)`**
- **`trj_viewer.m`** Opens a simple trajectory viewer to view single and multiple trajectories. **`act(22)`**
- **`write_T2xls.m`** Convert relevant data (meta, trajectories, call data) from T-files into XLS-files One XLS file per T-file, with same filename in the same folder. **`act(23)`**

Type "`help COMMANDNAME`" at the prompt to get some help on these files.

5.2. Configuration: OVERVIEW

Many parameters of the TOADSuite can and must be configured, e.g., to adapt the TOADSuite to your computer, microphone array, and bat species. This must and can be done on **multiple levels**: for the **whole TOADSuite**, for a **whole recording session**, and for **individual recordings**; and separately for the **call analysis**. This is done by changing the corresponding configuration files (m-files).

GENERAL IDEA of the TOADSuite configuration: [e-mails Peter 7.4.2016, 18:03; 02.12.2016, 19:28]

- The TOADSuite sequentially reads multiple configuration files. The ones that are earlier read are supposed to determine more general settings that are valid for a longer time. Configurations that are set later on are more specific for specific session or recordings.
- Configurations that are set later in that process overwrite values that were set earlier.
- We use the TOADSuite in an automatic sequential way to process all recordings in a session-WAV-folder. The first call in this process is starttoadpos:
    - Starttoadpos only reads toadbaseconfig.m.
    - It then checks whether the config-files toadbaseconfig. and sessionbaseconfig.m exist in the config-folder of the current session. Only if not, they are copied there from the program's global folder.
    - Additionally, starttoadpos generates for each recording ("sequence") two individual config-files, namely autosequenceconfig-...m and manualsequenceconfig-…m – but again ONLY, if they do NOT yet exist (i.e., at the first run of starttoadpos for this session).
    - If any of these files already exist in the data session folder, they are not overwritten.
    - Starttoadpos then cycles through all recordings, calling mktoadpos for each recording to calculate TOADs and positions.
    - mktoadpos will read the config-files for this session, which were copied/generated by starttoadpos in the config-folder of this session (not those in the program's global folder).
- In addition to processing all recordings of a whole recording session sequentially, single files can be processed (by mktoadpos). This enables to apply changes to the configuration of a single file (via uncommenting code in its manualsequence-config-file) and then process this file again.
- Alternatively, the whole folder can be processed again (by starttoadpos), after having changed (uncommented, changed and/or added code) one or multiple manualsequence-config-files.

IMPORTANT NOTES AND CONSEQUENCES:

- The config-files that are stored in a session's config-folder in the actrackdata folder determine how the TOADSuite analyses the recordings for that session. These config-files thus are **the rules for the analysis and a record of how the data were analyzed**.
- If starttoadpos is run again for the same session, it will use the config-files that were previously copied/generated in the session-config-folder. It will not use those of the global folder.

- Therefore, if you want to **change settings for the whole session after the first run**, change sessionconfig.m in the session's config-folder. Or delete the complete session-config-folder to do a complete fresh start.
- Since starttoadpos copied the current settings of the TOADSuite (from the program's global folder) to the session's config folder, you can safely change the program's config-files afterwards for processing another session. **Basically, changing the config-files of the TOADSuite corresponds to changing the options/preferences/etc. of any other program** – except that here they are changed by changing some code, while in other programs you usually do this in a user interface.
- It is possible to **change the settings for an individual recording** by changing its **manualsequence-config-file** (for example to set a different call detection threshold, call detection filters, and/or call analysis filters, or also different detection and reference channels). Since each run of starttoadpos/mktoadpos reads these files, another run of mktoadpos or starttoadpos will analyze the recordings again with the altered settings.
- **Don't touch** the **autosequence-config**-files.

CONFIGURATION FILES: The following configuration files are available:
(The folder in brackets is where you find them in the TOADSuite program for editing. Remember that the first run of starttoadpos copies these files into the session-config-folder.)

1. **`toadbaseconfig.m`** (TOADSuite bin folder)
    - Global configurations, such as the paths to the global- and tmp-folder
    - Only set once during installation.

2. **`toadconfig.m`** (TOADSuite global folder)
    - sets parameters for chktoadpos and other non-sequentially working files. Originally, it was also used to set parameters for mktoadpos. However, in our case, since we use a sequential processing of all recordings in a session-WAV-folder (by starttoadpos), `toadconfig.m` is functionally replaced by `sessionconfig.m` for setting the parameters for mktoadpos.

      For us, toadconfig.m sets the following parameters for chktoadpos:
    - XC_TIMEWINDOW_ARRAY: set the displayed times in the XCorr display
    - KEY_3D_LEFT/RIGHT/DOWN/UP: set the keys used to rotate the 3D display
    - SPG_CHANNELS: sets the number of channels for the external spectrogram display
    - CYCLECLICKINDICES: cycle the focal call temporally through all calls, i.e, after the last call, return focus to the first call (=true); or do not cycle (=false).
    - Usually, this file is not changed.

3. **`sessionconfig.m`** (TOADSuite global folder)
    - sets **session-wide** default parameters for mktoadpos.m.

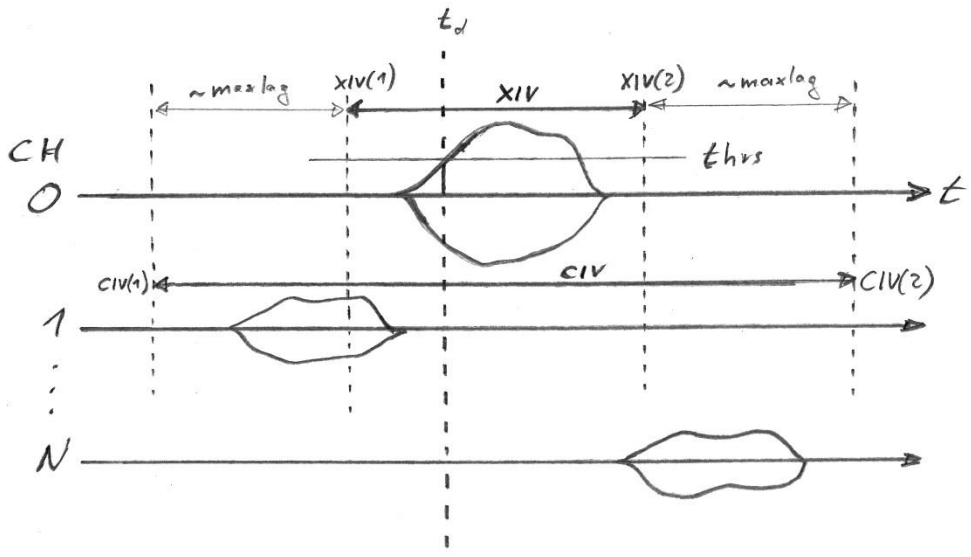

Fig. 8: Definitions of XIV and CIV. Channel 0 is the reference channel used for call detection. A call is detected when its envelope crosses a threshold thrs at time t_d. Then, the call is cut out from the waveform by the interval XIV around t_d and used as a template for the crosscorrelation with the waveform of the other channels. The waveform of the other channels will be cut by the interval CIV around t_d. The maximum lag (maxlag) of the calls in the other channels is defined by the geometry of the array: the largest distance from the reference channel to any other channel determines maxlag. For 60 cm, maxlag is about 2 ms, and for 120 cm, maxlag is about 4 ms (with a sound speed of 34 cm/ms).

- **array information** (AGMFILE, CHANNELS_READ, CHANNELS_USE, CH_CDT, VPOSORD)
- **Filter** for click detection and click-cross-correlation (BANDPASS_...)
- **ICIMIN: Minimum interval for next click detection** = refractory period of the click detector. The click detector only will detect a new call after ICIMIN has passed.
- **XIV: XCorrInterval**: Interval for the call template of the reference channel. The call template of the reference channel will be cut from the interval XIV around the detection time of the call. Adapt the duration of this interval to the typical call duration of your species. The first value should be negative (to include parts of the call that are before the detection time). **NOTE**: The call detection time t_d is not necessarily at the start of the call! It can also be at the end (e.g. in downward-FM calls). Thus, the XIV (and CIV) windows should be wide enough in both directions to include the whole call, no matter where it was detected. Rule of thumb: set XIV to the maximum call duration to expect: XIV = [-max_call_dur +max_call_dur].
- **CIV: ClickInterval**: Interval for cutting out the call for the call analysis in the reference channel, and cutting out the calls in the non-reference channels for crosscorrelation with the template of the reference channel. Adapt this to the XIV and the maximum lag of the array: CIV = XIC +/- mlag. (mlag for 60 cm: ~2 ms, mlag for 120 cm: ~4 ms). Also, CIV is the duration of the spectrogram that is displayed in the chktoadpos GUI.
- **DO_SMALLTOADCALLANALYIS**: activate/deactivate the call analysis

4. `toadnameconfig.m` (TOADSuite global folder)

- determines the list of species names and other tags for chktoadpos:
- Adapt SPECIESPROPNAMES to contain all the species you are tracking.

5. `toadthrehsold.csv` (TOADSuite global folder)

- a single value that determines the initial default threshold for call detection.
- Note that the threshold is automatically changed via user-input at the start of starttoadpos.

6. `Manualsequenceconfig_FILENAME.m` (session config folder)

- Sets **recording-specific** parameters for mktoadpos.
- Generated (not copied) by starttoadpos.m on the first run of a session.
- By default, it contains the same value as the whole session, and all text is commented. To manually change any of the settings, uncomment the required code and manually change the required values. Changes will take effect after a rerun of mktoadpos/starttoadpos.

CALL ANALYIS: The special case to configure the call analysis:

The analysis of acoustic call parameters by the TOADSuite is implemented by the code **`smalltoadcallanalysis.m`**.

- All its **configuration is directly done within the m-file**. Its settings are NOT stored anywhere else! Thus, changing this file deletes any record of your previous analysis settings! It is thus advisable to keep a copy of the file in the session-config-folder, or make a note of the settings.
- However, the settings are likely to be used across all analyses since these are standard settings for call analysis where no change should be required.
- In addition, it requires a **microphone calibration file**, whose path and name is also set in the m-file.
- For details, see **chapter 5.4.**

5.3. Configuration: EACH NEW SESSION

Each new session needs to be correctly configured. This includes for example specifying the correct array-file, the channels to be analyzed, the filters for call detection and call analysis, etc.
The configuration of a session is done via the **`sessionconfig.m`** (global-folder of the TOADSuite).

Open `sessionconfig.m` (global folder) and set the following entries:

1. Set your microphone array: **`AGMFILE = 'ARRAY_NAME-agm.csv'`**;
   Check that the specified microphone array file ("AGM-file") is in the TOADSuite's global folder.
2. Set the cannels according to the number of mics:
   - **`CHANNELS_READ = [0:N-1]`**
   - **`CHANNELS_USE = [0:N-1]`**
   - where N is the number of the microphones ([0:3] for 4 mics, [0:7] for 8 mics)
3. Ensure that: **`CH_CDT = 0`**;
4. Ensure that: **`VPOSORD = [0 1 1]`**;
5. **Filter for click detection** (kHz): BANDPASS_CDT_HP, BANDPASS_CDT_LP
   - These filter settings are used for detecting the calls. It can be useful to use a narrow bandpass around the main frequency band of the call to maximize signal-to-noise ratio.
6. **Filter for XCorr = TOAD-calculation** (kHz): BANDPASS_HP, BANDPASS_LP
   - These filter settings are used for the cross-correlation to calculate the TOADs and for all other further displays (spectrograms in the chktoadpos-GUI) and the call analysis. Select these filter to include all frequencies of the call.
7. **ICIMIN** (sec): Minimum interclick-interval before a new call can be detected.
   - The standard value of 0.02 works for search (and approach) phase calls. For tracking calls during a buzz, this value needs to be decreased.
8. **XIV** (sec): template-interval for the reference channel for the XCorr (**Fig. 4**)
   - This is the interval for cutting out the call-template from the reference channel for the XCorr. The interval is relative to the detection time of the call.
   - Should be matched to maximum call duration and to the expected time after which the call crosses the threshold
9. **CIV** (sec): Call-interval for all other channels (**Fig. 3**)
   - This is the interval for cutting out the call-waveform from all other channels to then be cross-correlated with the template from the reference channel. The interval is relative to the detection time of the call in the reference channel.
   - Should be matched to XIV and the maximally expected TOAD between the reference channel and any other microphone.
10. **DO_SMALLTOADCALLANALYIS**: activate ("true") or deactivate ("false") the call analysis

5.4. Configuration: CALL ANALYSIS

Background: The automatic call analysis analyzes each call that was detected by starttoadpos/mktoadpos. It uses the determined call position and a provided microphone calibration file to backcalculate the call as emitted by the bat (@ 1 m distance to the mouth), by compensating for geometric and atmospheric attenuation and for microphone characteristics. **NOTE** that this automatic correction can result in problems in specific cases, e.g. if the calculated distance is exceedingly large (→ very strong overcompensation of atmospheric attenuation of high frequencies). Therefore, approaches to deal with several issues are implemented. They will be explained below, and require some checking of the final data once the TOADSuite has finished.

Information: The call analysis is done by the file `smalltoadcallanalysis.m` (bin-folder). It is automatically run from `startoadpos` after each `mktoadpos`-analysis, but can also be started manually. Its standalone syntax is:

```
smalltoadcallanalysis(ANALYSISDIRECTORY,FILENAMECORE);
for example
smalltoadcallanalysis('/somedirectory/Project_Bsp_1/actrackdata/anal
ysis/2014-08-08_001/', '2014-08-08_20-57-56_0000004_Moth_000000');
or
smalltoadcallanalysis('daythree/','daythree_2013-03-25_12-17-58');
```

How the call analysis works:
The call analysis works on exactly those calls that were detected and bandpass-filtered by mktoadpos, as follows:

1. Reconstructing the call-waveform as emitted by the bat, by correcting for the microphone frequency response, for the recording system sensitivity, and for the geometric and atmospheric attenuation from the bat to the microphone.
2. Optional bandpassfilter (settings: `DO_BANDPASS = true/false`, with corresponding `BANDPASS_HP` and `BANDPASS_LP`)
3. Calculation of the Hilbert-envelope of the call, with optional smoothing of the envelope (settings: `DO_SMOOTH = true/false` and `T_SMOOTH`). When strong multi-path interferences are present, the algorithm has difficulties in finding the proper call start and end, which causes the call envelope to oscillate to amplitudes close to zero, which results in improper bandwidths and call-durations. To some degree, this can be overcome by smoothing the envelope. Default value for `T_SMOOTH` is 0.000002 s = 0.02 ms = 10 samples @ 500 kHz FS. The shorter `T_SMOOTH` is, the less is its smoothing effect, but the better it can capture temporal variation in the call envelope.
4. Setting the reference time point t_ref. This is either the time point of the maximum Hilbert-envelope ("t_ref = max") or the time point of click detection ("t_ref = cdt"). This is set by `REFERENCE_LEVEL_IS_MAX = true/false`. t_ref = max is the better standard method; max is defined as the time point with the largest envelope between the time of click detection (cdt) and the first time point where the envelope falls again below the level at cdt.

5. Calculation of a spectrogram with specified settings (`SPG_NFFT`, `SPG_WINDOW`, `SPG_OVERLAPRATIO`), which will be time-aligned with the call and its Hilbert-envelope.
6. Call parameters will be analyzed for multiple call windows which are defined by their level relative to the reference level in the vector `Threshold_Levels`. Additionally, calls can be defined by an energy criterion within the window determined by the call threshold (usually 5-95% of the total call energy).
7. For each threshold in Threshold_levels, the analysis determines the time points where the envelope exceeds the ref-level + threshold for the first time ("init") and the last time ("term"). These are the start and end times of the call for the full call window (100% energy window) determined by the current threshold.
8. In addition, a window with less energy is determined based on full call window, containing X% of the total call energy (e.g., 90% total energy, 5-95%). Again "init" and "ref" are determined as the time points where the envelope reaches 5% and 95% of the total energy.
9. For all time points of "init" and "term" and for "ref", the following parameters are calculated from the spectrogram: amplitude (linear), level (dB) and frequency-maximum.
10. Also, the RMS amplitude (linear) and level (dB) and peak frequency is calculated for the whole call window.
11. Additionally, the maximum amplitude and maximum level will be kept from the mktoadpos analysis, and the amplitude and level at cdt, max and t_ref will be calculated.
12. This data (and some more…) are summarized in an output-CSV-file with one row per call.
13. The most relevant data are copied from there into our final T-file by `make_T_files_toadpos`.

Configuration:

- Place the microphone calibration files in the 'global' folder of the TOADSuite
- Open `smalltoadcallanalysis.m` to configure the analysis parameters, in the "user configuration" and "advanced configuration" parts of the code:
- `REFERENCE_LEVEL_IS_MAX`: Determines the reference time point of the signal. This can either be the maximum of the envelope (our default: `REFERENCE_LEVEL_IS_MAX = true`), or the time point of click detection (`REFERENCE_LEVEL_IS_MAX = false`).
- `THRESHOLD_LEVELS`: a vector with multiple thresholds for level-dependent definition of call start and end, e.g., -6, -12, -20 and -30 dB. For signals with low SNR, it might be necessary to add -3 and not use -20 and -30.
- `ENERGY_QUANTIL`: a vector specifying the energy quantiles to define a shorter call window within the call window defined by THRESHOLD_LEVEL for analysing the call. The shorter call window is defined based on the total energy within the larger call window and ranges within the two energy quantiles defined here.
- Spectrogram parameters: FFT-size, window type and overlap (`SPG_NFFT`, `SPG_WINDOW`, `SPG_OVERLAP`).

- `FILE_IN_SUFFIX / FILE_OUT_SUFFIX`: sets the suffixes of the input and output files. No need to change
- To apply an additional filter (in addition to the filtering already performed by mktoadpos), activate and set the HP- and LP-filters (`DO_BANDPASS = true; BANDPASS_HP, BANDPASS_LP`). **NOTE**: Set this filter to a sensible maximum frequency. Otherwise
- The algorithm has difficulties in finding the proper call start and end when strong multi-path interferences are present, which cause the call envelope to oscillate to amplitudes close to zero, which results in improper bandwidths and call-durations. To some degree, this can be overcome by smoothing the envelope. If this is the case, activate the smoothing (`DO_SMOOTH = true`) and set an appropriate smoothing time (`T_SMOOTH`).
- Limits for distance and correction of atmospheric attenuation (AA) and geometric attenuation (GA):
    - MAXDISTANCEREPLACE = [X Y]. For calculating AA and GA, the distance of all positions with a distance of larger X will be set to Y. I.e., the position will not be changed, but if this position is more than Y meter away from the array, a distance of Y will be assumed. Set it to [100 100]: then for all positions, that are more than 100 m away (which is a wrong position anyway), the position will be assumed to be 100 m.
    - MAX_ATMATTCOMP: This is the maximum AA that will be compensated for. Consider a sampling rate of 500 kHz, where the maximum frequency is 250 kHz. At 250 kHz, AA is ~11-12 dB/m. For a bat at 10 m away, this causes a compensation of AA only of 110-120 dB! This will extremely exaggerate the level of the noise at this frequency and is not sensible to fully correct. Thus, we apply a maximum correction. Peter's recommendation is to set it to 20 dB, maximally 40 dB. I prefer 40 dB to have sufficient correction at relevant frequencies and distances (at 100 kHz, AA is ~3.3 dB/m). Consider a sensible value for your situation incl. maximum bat frequency and expected position.
- To calculate signal-to-noise-ratio: the signal-to-noise-ratio (of the uncompensated WAV-file as recorded, in dB, see SCL_SNR) are calculated based on the X% faintest part ('noise') and the Y% strongest part ('signal') within CIV. P_QUANTIL_NOISE and P_QUANTIL_SIGNAL specify the X and Y% as quantiles. P_QUANTIL_NOISE = [0 0.1] uses the 10% faintest signal, P_QUANTIL_SIGNAL = [.9 1] uses the 10% loudest signal.

  SCL_SNR specifies the Signal Compensation Level for the SNR calculation. 1 = uncompensated WAV, 2 = Ungained, 3 = BP, 4 = received, 5 = emitted.
- `CHANNEL`: Determine which channel is used for the call analysis. This is set to the same channel which is used for detecting the calls, by setting it to `D.ch_cdt`: `CHANNEL = D.ch_cdt`
    - The click-detection-channel is determined in session-config (`CH_CDT`)
- `DO_CALCULATE_SOURCE_SIGNAL`: activate (`true`) or deactivate (`false`) the calculation of the call waveform as emitted by the bat. If set to true, microphone characteristics, geometric attenuation and atmospheric attenuation is compensated. If set to false, the call is analysed as received by the microphone.
- `MICROCALIBRATION_PATH / _FILE`: Enter the path and filename for the microphone calibration file
- `SENS_MIC / _AMP / _DIG`: Set the sensitivities of the recording system. See chapter "Definitions" for definitions and chapter "Microphone Calibration" for how to calculate them.

  - NOTE 1: The sensitivities are added to obtain the total sensitivity of the recording system. So, if only the overall sensitivity of the system is known, it can be ascribed to any of the values.

- `SENS_COR`: Set any additional correction sensitivities, particularly the recording gain.
  - NOTE 2: We calibrate the recording system for a recording gain of ZERO. The recording gain which was set at the sound card can then be easily entered here, allowing for different recording gains between sessions.
  - NOTE 3: You might want to change the recording gain of the sound card over the course of an evening in the field, or between evenings. Since **one gain setting is used throughout the whole call analysis of any given session**, you must **start a new recording session whenever you change the recording gain**.
- `SENS_P_REF`: reference pressure for the calculation of the sound pressure level. Standard reference sound pressure in air is 20 µPa (= 2 * 10^{-5} Pa = 0.00002 Pa). Alternatively, it can be set to any other value. A value of 1 makes sense if just a dB relative to full scale are required.
- `SCL_USE, I_SCL_PRIMARY, SCL_P_REF:` **TBA - DON'T TOUCH!**
- `EP_THL, EP_RESTR, EP_DB, EP_DB2:` **TBA - DON'T TOUCH!**
- `ICLICKPLOT, FASTIVSEARCH:` **TBA - DON'T TOUCH!**

Comment on the microphone calibration file:
only include frequencies into the calibration file that are sensible. During call analysis, the compensation for attenuation and microphone characteristics will only be conducted up to the highest frequency in the calibration files. All frequencies above the highest frequency in the calibration file will be set to Zero level.
When sampling at high sampling rates (500 kHz), this allows to exclude very high frequencies, which would be problematic when compensating for atmospheric attenuation.

Output:
The output table (*-toadcallanalysis.csv) is written into the analysis-output directory (where also *-mktoadpos-out.mat is found). From there, `make_T_files_toadpos.m` will copy the most relevant data into the final T-files.

[For detailed information on the stored call analysis data, see the code (`smalltoadcallanalysis.m`) and particularly the e-mails be Peter Stilz from 02.12.2016 (first version), 29.08.2017 (second version incl. SL), both included in the appendix, and 01.02.2018.]

How to run it:
The call analysis is started automatically with every call to `starttoadpos.m`, given it was activated in the sessionfonfig-file.

How to run it later, or run it again:

- The call analysis does not alter any of the other output files of `mktoadpos.m` or `chktoadpos.m`. It simply generates its own output files. Therefore, the call analysis could also be run AFTER trajectories had been reconstructed, for example to change call analysis parameters.
- To run it again, (re-)configure the call analysis and then start `starttoadpos.m` again. This will overwrite the result files of `mktoadpos.m` (because `starttoadpos.m` calls `mktoadpos.m`)

and will generate/overwrite the result files of the call analysis. It will not alter the result files of `chktoadpos.m` (= the manual trajectory reconstruction).

- **NOTE**: The different result files are linked only based on **call number** (not call time). Thus, very great care has to be taken that the new run of starttoadpos detects exactly the same call as the previous run, which was used to reconstruct trajectories! To achieve this, follow these steps:
  1. Just to be safe, backup your existing data at a different location.
  2. Delete the config-files in the session-folders of those sessions that you are going to re-analyse.
  3. Configure the session exactly the same as the previous time (check in the config-files of your backupped data). Particularly, ensure that the call detection threshold has the exact same value by typing it at the prompt when running `starttoadpos.m`.
  4. `Starttoadpos.m` will then cycle through all the recordings in your session, call `mktoadpos.m` and `smalltoadcallanalysis.m` for each recording, and save/overwrite the call detection and call analysis results. The result file of `chktoadpos.m` will not be altered.
  5. Run `summarytoadpos.m` again to combine all three dataset (call detection, trajectory reconstruction, call analysis).
  6. Run `make_T_files_toadpos.m` again to generate our final trajectory data files.

Potential problems that cause program termination; the measures that we applied to prevent this; and ensuing data filtering and selection requirements:

Potential problems and how the TOADSuite deals with them:

1. **Extremely short detected calls (of less than 5 samples**). This causes several missing temporal positions for call analysis. ***Solution***: Missing individual time points are set to i_cdt. Missing interval-end-time points are set to the interval start time point. If no spectrogram slice is present during the call, the spectrogram slice closest to the inveral start will be used. (e-mail Peter 2018-01-26).
2. **Extremely large bat distance**. This leads to a failure of the source-signal calculation due to the extreme required compensation for atmospheric and geometric attenuation. ***Solution***: For positions with too large distances, a maximum distance is assumed for the source-signal calculation (see parameter MAXDISTANCEREPLACE). (e-mail Peter 2018-01-29).
3. **Extreme (over-)compensation at high frequencies.** At very high frequencies, the compensation of atmospheric attenuation becomes very large and will only compensate noise, not signal, possibly leading to various artefacts (e.g., >95% of energy within only one sample of the thl-window, leading to a missing thl_q-window). ***Solution***: thl_q-window is forced to a minimum duration of one sample; parameter MAX_ATMATTCOMP to avoid over-compensation. (e-mail Peter 2018-02-01).
4. **No bat position**. This happens if the TOAD-values are so unsuitable that the position-calculation fails; thus, no source-signal can be calculated since distance and direction is unknown. Solution: in this case, a standard position (distance = MAXDISTANCEREPLACE(2), elevation = 0, rotation = 0) is assumed for the source-signal calculation. (e-mail Peter 2018-02-06).

Required data selection:

1. **Exclude all calls with very short duration.** Use only calls that are longer than 5 samples, or rather also a biologically plausible value (> 0.1 or 0.2 ms).
2. **Exclude all positions with large distance**. We do this anyway in chktoadpos since we cannot track sensibly further than 20-30 m. But it can also be done automatically by deleting all positions and calls at too large distances; at least larger than the value set by MAXDISTANCEREPLACE(2).

3. **No specific data selection required.** Some problematic calls might be deleted already by excluding very short calls above. Also consider before the analysis the maximum frequency and distance at which the source-signal can sensibly be calculated. Set your parameters and microphone calibration file accordingly. Possibly restrict analysis to positions within a maximum distance; and to do not put too much trust on high-frequency measurements.
4. **Exclude all calls without a position.** Ideally, do this automatically by excluding all calls that do not have a position; but it will also be obvious in chktoadpos.
5. Note that the TOADSuite automatically assigns **error flags** in certain cases (if there was a mathematical error during position calculation), and based on this information excludes positional and temporal (call) data. Exclude all calls and positions with an error-flag > 0.

5.5. Workflow

To use the TOADSuite, **open Matlab 2007b**, set the **current directory to the TOADSuite bin-folder**.

Steps 1-5: Data storage and security. Do this immediately the next day after field work! Reasons: weather data is stored in a circular buffer on the Kestrel and may be overwritten. You will catch missing data and other errors early on before the next recording session. You might have forgotten important details after a few days.

1. Copy your **WAV-files** to the WAV-folder in actrackdata (if not already there from the recording). Do an initial quality check: watch your recordings (Avisoft SASLab, or other sound software). Empty / low quality / unwanted recordings can be deleted at this point / don't need to be copied (delete the WAV-files of ALL channels per recording!)
2. **Download weather data:** download weather data using the Kestrel Communicator Software ("Kestrel Tracker" Tab, COM8 (?) Port). Export ("Data Log" Tab) and store the CSV-file in the weather folder, with filename: weather_YYYY-MM-DD_SSS.csv (use session date!).

   Important information on the **data structure of the weather file**:

   The TOADSuite (code starttoadpos.m, subfunction getsoundspeed.m) reads the date&time, air temperature, relative humitidy and pressure from the weather data file, expecting the following structure and units:
   - Rows 1 + 2 are header rows, data starts in the 3. row.
   - Date & time: in the 1. column, with format: yyyy-mm-dd HH:MM:SS
   - Temperature: in the 4. column, in °C
   - Relative humidity: in the 6. column, in %
   - Pressure: In the 10. column, in hPa

   Example: Below is the output format of our Kestrel 4000. The required data columns are highlighted in green; the remaining columns are not used by the TOADSuite. Ensure that your data has the same structure for date&time, temperature, relative humidity and pressure.

FORMATTED DATE-TIME	DT	WS	TP	WC	RH	HI	DP	WB	BP	AL	DA
YYYY-MM-DD HH:MM:SS	s	m/s	Â°C	Â°C	%	Â°C	Â°C	Â°C	hPa	m	m
2015-08-12 20:22:00	4.93E+08	0	26.3	26.3	63.2	27.4	18.8	21	939.3	635	1266
2015-08-12 20:24:00	4.93E+08	0	26.2	26.2	68.1	27.7	19.8	21.6	939.5	634	1266

3. **Plot weather data**: run `act(1)` or `plottoadweather.m` to plot an overview figure with T, RH, c, and AA plotted over time for your recording session (will be stored as TIF and FIG with the same filename as the weather data in the weather folder).
   - Check that weather data has the correct format: open the weather-CSV data in a text editor. The date / time string in the first column must have the format "yyyy-MM-dd HH:mm:ss". If this is not the case, check your Region and Language settings (see installation).
   - Note: the format of the weather CSV file (as displayed in a text-editor) depends on the Region and Language settings when the file was **generated**.
4. **Enter data into tracking_list_YYYY.xls**: one row per recording session.
5. Check your **field notes** again; add any information that might still be missing (such as problems with weather data observed in previous step!). Scan your field notes and store them with your acoustic tracking data (folder field_notes in actrackdata).

Steps 6-7: Pre-processing. Prepare WAV-files for analysis.

6. **In case of multi-channel-WAV-files from Avisoft recorder**: split them first into single-channel-WAV-files (`splittoadwavs, act(2)`).
    - The single-channel files will be in the session-folder, the original multi-channel files will be moved to a subfolder "ORIG_RECs".
7. **In case of WAV-files recorded at lower sampling rate than 500 kHz (Fireface)**: resample them first to 500 kHz (`resampletoadwavs, act(3)`).
    - The resampled files will be in the session-folder, the original recording files will be moved to a subfolder "ORIG_RECs_192".

Steps 8-9: Configuration. Configure the program parameters of the TOADSuite.

8. **Configure the session** according to your needs for this session, e.g., species etc. (see 5.3 above).
9. **Configure the call analysis** according to your needs for this session, e.g., species etc. (see 5.4 above).

Step 10: MAIN ANALYSIS. Analyze the sound recordings to obtain TOADs and 3D-positions.

10. **Run the TOADSuite to calculate TOADs and 3D-positions**, either for a whole session (`starttoadpos, act(11)`) or a single recording (`mktoadpos, act(12)`).
    - A window appears to set the call detection threshold by clicking, then hit enter. If you want to manually type in a threshold, you still must first set any threshold by clicking, then type in the manual value at the command window and hit enter.

Step 11: MAIN MANUAL ANALYSIS. Visual checking, error control, trajectory reconstruction:

11. **Run the TOADSuite file by file** to visualize the positions, error-check them, assign species names, reconstruct trajectories, etc (`chktoadpos, act(13)`).
    - Manual trajectory reconstruction and visual control. See chapter 5 for details.

Steps 12-13/14: Post-processing. Summarize data and store final data format.

12. **Run the TOADSuite to generate the summary data** of the TOADSuite (`maketoadsummary, act(14)`).
13. **Run the TOADSuite to interpolate trajectories and to combine all data into our final T-file-MAT-format** (`write_T_files_toadpos, act(21)`).
    - The T-files are designed to provide all the data we need for further analysis in a complete and concise style, including the raw-positions, the interpolated positions and the interpolation functions, the acoustic data and the meta data (T, RH, c, and many more). See chapter 7 for details on the structure and content of the T-files.
14. **Run the TOADSuite to convert data from T-MAT-files to XLS-files** (`write_T2xls, act(23)`).
    - If you prefer to work with XLS-data files instead of MAT-data files, the relevant data from the T-files are exported by this function to XLS-files. XLS files have the same name and are

in the same folder as T-files. See chapter 7 for details on the structure and content of the XLS-files.

Steps 15-16: Further work.

15. Open the **quick viewer for trajectory data** (`trj_viewer, act(22)`).
    - This program allows you to easily view the raw and interpolated data of single trajectories or all trajectories of a recording, both in 3D and over time. It also provides a minimum of visual analysis (flight speed, error between raw and interpolated data).
16. **Further analyses**
    - Further analysis can investigate trajectory shape and its changes, call structure, etc. A range of Matlab functions exist to aid in this analysis. ASK before you reinvent the wheel.

NOTES:

- **EXCEL-Error:** If at any point the TOADSuite throws an error about Excel, such as "??? Undefined function or variable "ExcelWorkbook", try it again. Sometimes Matlab has difficulties accessing xls-files, and this error seems to happen regularly on the first run.
- **If errors occur, read them carefully.** Though many potential things can go wrong, first steps might be to check your configuration, data files (weather, tracking_list), and filenames.

6 TOADSuite GUI (`chktoadpos, act(13) )`

After the TOADs and 3D-positions are calculated (by mktoadpos for a single recording, or starttoadpos for a whole session), these positions need to be assigned to one or multiple trajectories. This is done by the TOADSuite GUI. Additionally, erroneous positions can be deleted, species ID can be set, and the time points of additional calls that were not detected by the TOADSuite can be manually added.

1. **[If needed, first adjust the channel numbers for the external spectrogram: Open `toadfonfig.m` (global folder) and set the following entries:**
   - **`SPG_CHANNELS = [0:N-1];`**
   - where N is the number of the microphones ([0:3] for 4 mics, [0:7] for 8 mics)
   - a lower number than the used mics is possible if not all channels shall be displayed.]

2. **Open Matlab 2007b, set Matlab's current folder to the bin-folder of the TOADSuite, and open the TOADSuite GUI by typing `chktoadpos` or `act(13).`**
   - Select a recording from the list of TOADSuite output files.
   - The TOADSuite will calculate all required information and open the main GUI (**Fig. 5**):

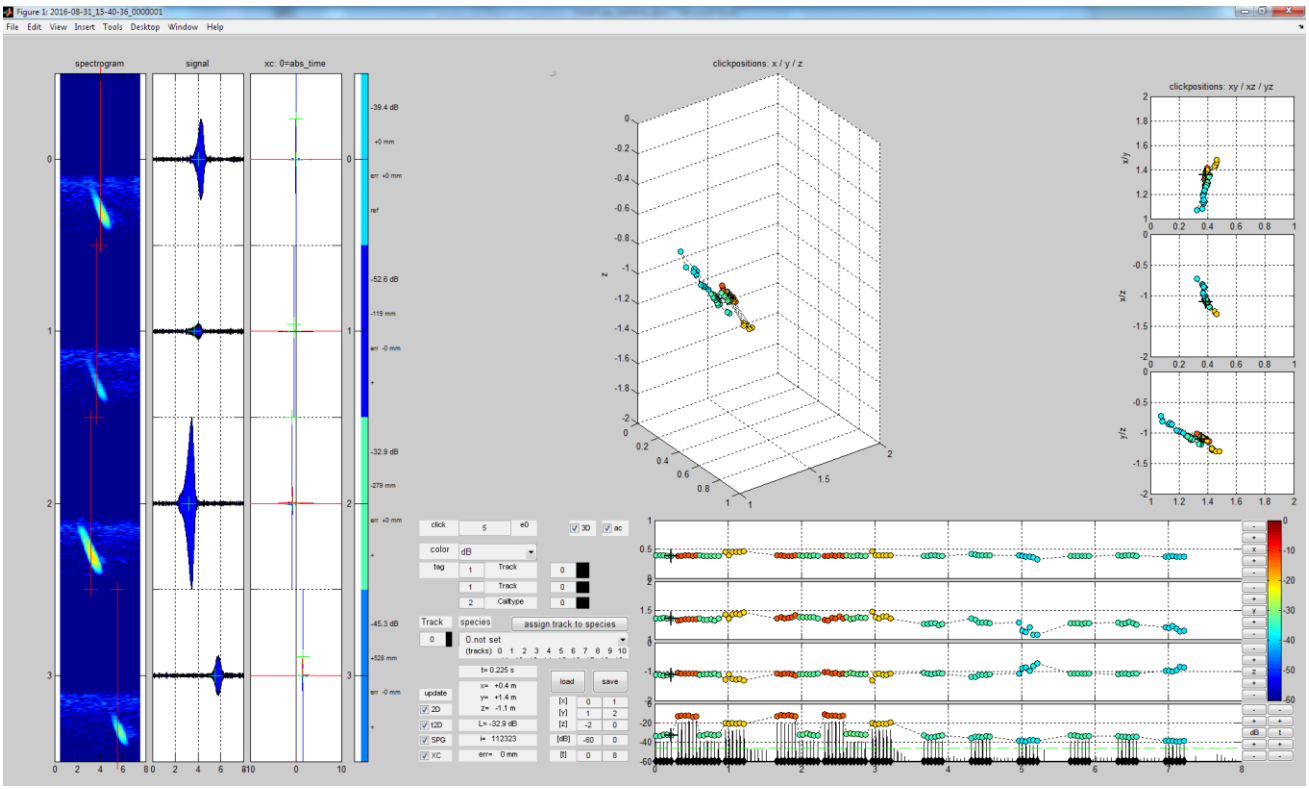

Fig. 5: Main GUI for path reconstruction. The GUI consists of a 3D-view of the positions (top center), 2D-views of the positions (right), temporal views of the X-/Y-/Z-coordinates and received level (bottom), and call plots on the left (spectrogram, signal, XCorr). All positions are color-coded; color code can be set by the drop-down menu "color" or the keys c/C (level = dB, Track = trajectory number, Calltype, Quality).

- Arrange the Matlab command window and the GUI so that you can see both (to see the GUI output on the command window).

- The GUI can be partially navigated by mouse clicks, and by the keyboard. Keyboard is faster, so it pays to learn this. If the GUI does not react to typing a key, click on the figure first. To learn about all hotkeys, type 'h' (help):

```
<- , ->     : navigate through clicks
- , +        : dis- / enable current click for 3D
q , Q        : dis- / enable current click accoustic
0 - 9        : set click tag to number
^ , v        : in- / decrement tag value
R            : set clicks in time range to tag value
c , C        : cycle color display: dB, tags ...
t , T        : cycle current tag
D            : toggle display tag group track
I            : write tag group track timing info
x            : cycle base for xcorr: abs time, trigger, expected
X            : toggle (time consuming) xcorr display
G            : toggle (time consuming) spectrogram display
u , U        : toggle update of (time consuming) 2D-plots
a, d, s, w   : rotate 3D display (can be adjusted in toadconfig.m)
L            : reset diagram limits to enabled clicks
N            : toggle click number text display
z            : decrement current species
Z            : increment current species
A            : assign track to species
l            : load tag data
S            : save data
W            : save workspace
o            : open overview spectrogram function
O            : plot overview figure
Y            : plot hyperbola figure
i            : plot ici figure
f            : plot external figures
g            : plot click spectrogram
*            : add external clicks
h            : write help text
```

- **The purpose of this GUI is to combine multiple positions into a trajectory, i.e., assign them a common trajectory number and a bat species, and to exclude wrong positions.**
- Therefore, you select individual calls/position by mouse click or by navigating with the left/right arrow keys, or by typing the call number into the field "click".
- You then set their trajectory number by the up/down arrows, or by typing the number into the tag-field.
- Multiple calls can be selected based on time by typing "R" and then entering start time, end time, and trajectory number ("tag value").
- **A comment on the tags**: The general idea is that each call/position has multiple tags assigned to it. Each tag can take on multiple values ("tag value") and specifies a different property of the call. Tag 1 is the trajectory number. By default, it is set to 0 (= not assigned to any trajectory). To assign this call/position to a trajectory, we change the tag value of tag 1 to a different value, i.e., to 1 for trajectory 1, 2 for trajectory 2, etc. So, all we do in this GUI is to change tag values, and thereby assigning calls to a given trajectory and setting other properties.
- To **exclude/include a 3D-position**, but keep the timing, uncheck/check "3D" or type - / +. This is needed for calls that had been recorded (i.e., they are present and their timing is correct), but the position is wrong.

- To completely exclude a 3D-position and its call timing (e.g., when the detected signal was not a call), uncheck “3D” and “ac”, or type “-“ and “q”. To include, check the boxes again or type “+” and “Q”.
- To set the species for a trajectory, type the trajectory number under the field “Track” and select the species from the drop down menu “species”. Then click “assign track to species”. Type “z” or “Z” to decrement/increment species assignment.
- To **save**, type “S”, or click on “save”.
- Load existing trajectory data (i.e., the assigned tags) by typing “l” or clicking “Load”.
- Due to the many displays, the GUI is not very fast. To speed it up a bit, different displays can be switched off by unchecking them in the list “update” or typing the corresponding keys.
- A range of additional plots can be generated, including spectrograms of the call on all mics, spectrogram of the call sequence, 3D-plots and the TOAD-hyperbolas. See help for further information.
- The TOADSuite might not detect all calls, depending on the detection threshold and the signal-to-noise ratio. Therefore, we can **manually add more calls** (i.e., their timing; there is obviously no position for them. But we can use the time to later interpolate their position based on the other call positions). Save data first, then type “*”. After confirming the correct WAV and CSV-file, the **extra-click-GUI** (`addtoadextraclicks.m`) will open (**Fig. 6**):

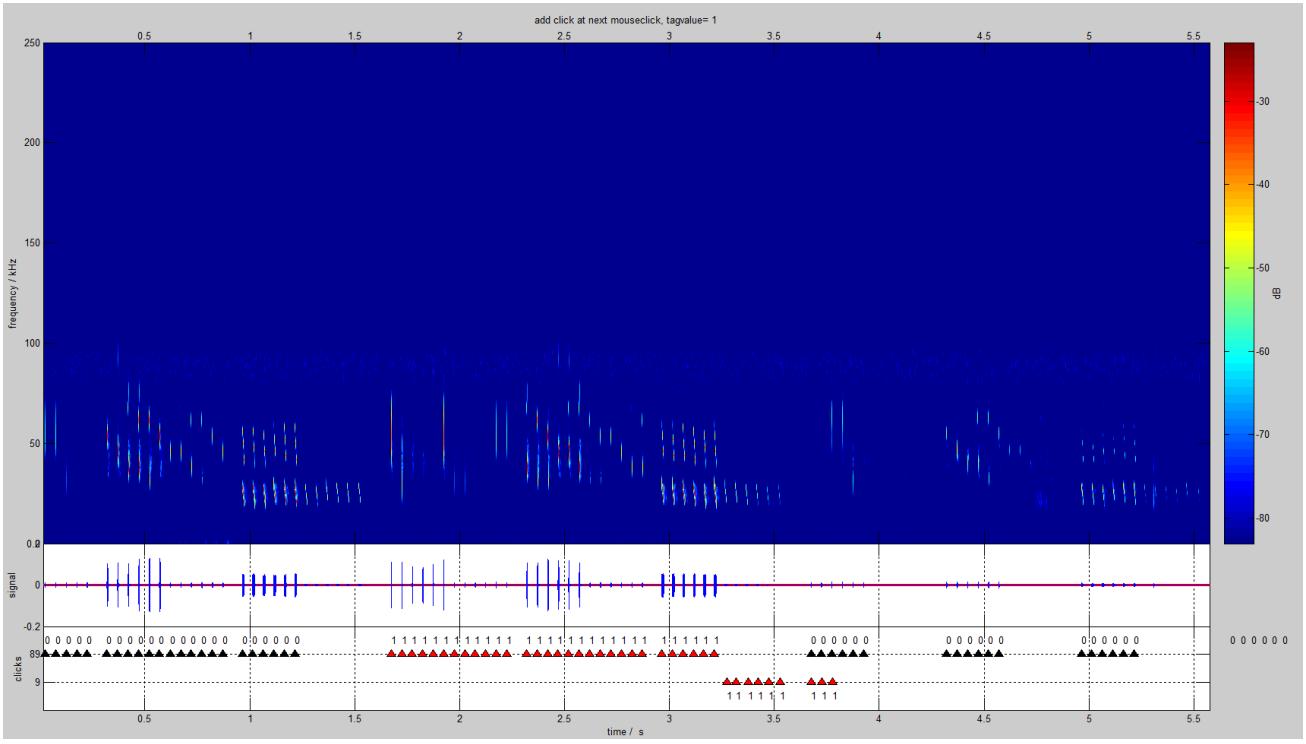

Fig. 6: Extra-click GUI for adding the times of additional calls. The GUI consists of a spectrogram (top) and oscillogram (middle) of the call recording and a display of the detected calls with their trajectory number (bottom). Calls with trajectory number 0 (= not assigned) are displayed in black; calls assigned to a trajectory are displayed in different colors with their trajectory number on top. The top row shows the automatically detected calls, the bottom row shows the manually added calls.

- Type “h” to get help about all available hotkeys:

```
escape              reset settings
0-9                 set current tagvalue to value; enter add mode
T                   increment current tagvalue
t                   decrement current tagvalue
D                   enter delete mode
+                   zoom in    *2
-                   zoom out   *2
rightarrow          move right 1/8
leftarrow           move left  1/8
uparrow             move right 7/8
downarrow           move left  7/8
home                move to begin
end                 move to end
S                   save extraclicks
h                   display help
mouse click         add click here with current tag value
mouse click         delete closest click (delete mode)
mouse zoom          zoom to interval
```

- Use the mouse to zoom in to a region by holding down the mouse key and dragging along the display. Different keys (see help) are available for zooming in and out and for moving the display.
- To **add calls** to a trajectory, enter **Add Mode** by hitting the trajectory number to which you want to add a call. Then click on the call. It will be marked and the time where you clicked will be stored as its time. Add Mode and the current trajectory number are shown in the title.
- To **delete a call** from a trajectory, enter **Delete Mode** by hitting “D” and then clicking at/next to the call. Delete Mode is indicated in the title.
- To **save**, hit “S”.
- To work on a new recording, save your data, close the figure, and start chktoadpos / act(13) again.

7 Output: structure & content of the T/XLS- data files

After trajectories were reconstructed, maketoadsummary (act(14)) and write_T_files_toadpos (act(21)) write our final data files. One T-file is stored for each trajectory, also for multiple trajectories originating from one recording. T-files store the raw-3D-positions, interpolated 3D-positions, the interpolation code, call data, and meta data.

T-files are named as follows:

- Structure: T[Rec-Number]__[TRJ-Number]__[Rec-Date]_[Rec-Time][_...].mat
- Structure: `TNNNNNNN__NN__YYYY-MM-DD_HH-MM-SS_NNNNNNN[_...].mat`
- Example: `T0000256__01__2016-08-27_21-24-02_0000256[_...].mat`

T-files contain three structure variables with various fields. The variables are:

- **`TRJ`** contains the raw and interpolated trajectory data and acoustic call data.
- **`meta`** contains meta data.
- **`pp`** contains the interpolation coefficients, called ppform by Matlab

Interpolation of the trajectories: write_T_files_toadpos.m using cubic smoothing splines (code interp_3D.m using the Matlab function csaps) to interpolate each trajectory with three different smoothing factors (0.1 = little smoothing, 1 = medium smoothing, best option in most cases, 10 = strong smoothing). Interpolation is done separately for the X-, Y- and Z-coordinates: each of the coordinates is smoothed over time. The coefficients of the smoothing are called ppform by Matlab; we store them for further use in the variable pp. Since we use three different smoothing factors, `TRJ` and `pp` are structures of the size 1x3, i.e., one column for each of the three different smoothing factors. Each of the columns contain the same data fields as specified next.

`TRJ`: Trajectory data (variable `TRJ`)

- **size**: 1x3, each column corresponds to one smoothing factor (0.1, 1, 10).
- **fields**: raw, interp, cont, TC1_ta2, TC2_ta2, cd, cd_header, cd_thrs
- **access the data by**:
    1. `TRJ(1,n).FIELDNAME` accesses the data from the `nth` smoothing factor stored in the field `FIELDNAME`
    2. `TRJ(1,1).raw` retrieves the raw-data for the first smoothing factor. Note that all raw-data obviously are the same, so `TRJ(1,1).raw` has the same content as `TRJ(1,2).raw` and `TRJ(1,3).raw`.
    3. `coords = TRJ(1,2).interp(:,3:6)`: the interpolated coordinates of the second smoothing factor
- **`raw`**: raw 3D-coordinates, one row for each call as detected and calculated by the TOADSuite. Each row is one call, columns are:
    1. trajectory number
    2. ta (call arrival time at the mic)

3. X
4. Y
5. Z
6. – 20: additional columns of only NaN (Not a Number)

- **`interp`**: interpolated 3D-coordinates, one row for each call. This includes the manually added calls; interp thus can have more rows than raw. Each row is one call, columns are:
    1. trajectory number
    2. ta (call arrival time at the mic)
    3. te (call emission time at the bat)
    4. X
    5. Y
    6. Z
    7. e1 absolute error (= distance) between raw and interpolated coordinates (m)
    8. d1 distance from raw-coordinate to Mic0
    9. e2 normalized error between raw and interpolated coordinates (%): e1/d1*100

    Note: e1 / e2 are quality measures of the interpolation, they tell how much the interpolated data deviate from the measured raw data, both absolute (e1) and relative (e2). However, they are not a quality measure of the tracking itself, i.e., they do not tell how precise the raw-coordinates are.

- **`cont`**: interpolated 3D-coordinates, interpolated at time points with 10 ms intervals. Use this for plotting nice continuous trajectories. Note that time axis has been extended by 100 ms before and after the tracked positions, i.e., positions are EXTRAPOLATED for 100 ms. Each row is one time point, columns are:
    1. trajectory number
    2. te (call emission time at the bat)
    3. X
    4. Y
    5. Z

- **`TC1_ta2`**: Timecode (based on arrival time) for all calls (including those without position), in the format: `HH:MM:SS:FF:0.xxx`

- **`TC2_ta2`**: Timecode (based on arrival time) for all calls (including those without position), in the format: `HH:MM:SS.xxx`

- **`cd`**: call data: results of the acoustic call analysis, NxM double array. Each row is one position, each column a call parameter. The data is organized in blocks:

    1. The first block contains call detection time as arrival time at the mic (dta) and as emission time at the bat (dte) and call interval (CI).

    2. The 7 further blocks contain different call data types extracted from differently sized windows around each calls. The 7 blocks are: (1) call duration, (2) start frequency, (3) frequency at maximum amplitude, (4) peak frequency, (5) end frequency, (6) peSPL, (7) RMS-SPL.

    3. For each data-type block, there are multiple window sizes: (1) the full call window used for analysis as defined by the CIV-parameter. This is much larger than the call and thus usually not meaningful. (2-x) threshold used for call detection/separation/cutting. Thresholds are the -X dB thresholds below the peak of each call, and for each -X dB threshold, there is the full (100%) window, and the X-Y% energy window (usually 5-95% = 90% energy window).

4. Note that not all calls have acoustic data: positions which were interpolated based on manually added extraclicks only have time, but no further analysis.
5. Note that start- and end-frequency data for the full call window (CIV) are very likely (totally) wrong and meaningless. This is because frequencies are linearly interpolated between neighboring time slices of the spectrogram. For the full CIV-window, the outermost time slices are used, so it cannot be interpolated, but it is extrapolated. Due to the usually lacking call data at the edges of the CIV-window and the noise, this results in meaningless values. This does not matter, since we don't use such a large window anyway.
6. **Reference distance** for all source levels is **1 m distance** to the bat's mouth. To convert to 10 cm to the bat's mouth, add 20 dB (= 20*log10(1/0.1)).

- **`cd_header`**: column headers for the call data.
  1. row: description of acoustic parameter
  2. row: parameter counter
  3. row: analysis window size: 100 = full window determined by the –X dB call threshold, 90 = X-Y% Energy window (usually 5-95% call energy)
  4. row: threshold value (-X dB)
- **`cd_thrs`**: threshold values (-X dB) used for the call analysis

`meta`: meta data (variable `meta`)

- **size**: 1x1, with multiple fields
- **fields**: most of the fields are self-explanatory, here is a short summary:
  1. `pro_ / wav_ / pos_ / meta_ / weather_ / analysis_ / config_path`: full path name of the project- / wav- / position- / meta- / weather- / analysis- and config-folder for this trajectory
  2. `session_folder`: name of the session folder
  3. `ses_Y / _Mo / _D / _num`: Year/Month/Day/Number of the recording session
  4. `fn`: filename of the summary.mat file
  5. `rec_num`: continuous 7-digit recording number
  6. `y / mo / d / h / mi / s`: year / month / day / hour / minute / second of the recording
  7. `tjr_nums_in_rec`: all recording numbers in this sound recording
  8. `T / RH / c`: Temperature / relative humidity / sound speed
  9. `w_time`: time of weather measurement
  10. `c_toadsuite`: sound speed as used by the TOADSuite to calculate TOADs
  11. `mic_pos`: vector with microphone positions
  12. `mic0_wav_fn`: filename of the recording at Mic0

13. `FS / nbits`: sampling rate and resolution of the WAV-files
14. `loc`: location, as taken from the tracking list
15. `array_alpha / _beta / height`: angles and height of the microphone array (taken from the tracking list)
16. `gain`: recording gain (dB), as taken from the tracking list
17. `d1 / a1 / h1`: distance / angle / height to/of reference object 1 (from tracking list)
18. `ref1_pos`: position of reference object 1
19. `ref2_pos`: position of reference object 2 (currently unused)
20. `refs_pos`: position of reference objects, size 8x4 (currently unused)
21. `curr_trj_num`: number of current trajectory (i.e., of this T-file)
22. `rec_spec`: recorded bat species of this trajectory

`pp`: interpolation coefficients (variable `meta`)

- **size**: 1x3, each column corresponds to one smoothing factor (0.1, 1, 10).
- **fields**: smoothing_factor, X, Y, Z, TE
- **access the data by**:
    1. `pp(1,n).FIELDNAME` accesses the data from the `nth` smoothing factor stored in the field `FIELDNAME`
    2. `pp(1,2).smoothing_factor` retrieves the smoothing factor for the second interpolation (should be 1).
    3. `pp(1,2).X` retrieves the ppform (i.e., the coefficients) for the second interpolation for the X-data.
- **`smoothing_factor`**: stores the three smoothing factors (0.1, 1, 10).
- **`X / Y / Z / TE:`** ppform for the X- / Y- / Z-coordinates and the emission times.

To use the ppforms, use the Matlab function `ppval`. For example, if you want to calculate the X-/Y-/Z-coordinates for a certain (set of) call emission time(s) te and for the second smoothing factor, use the ppform for X / Y / Z and the call emission times as input:

```
X2 = ppval(pp(1,2).X, te)
Y2 = ppval(pp(1,2).Y, te)
Z2 = ppval(pp(1,2).Z, te)
```

Note: the ppforms of X, Y, and Z take EMISSION times as input. The ppform for the emission times (pp.TE) takes ARRIVAL times as input.

```
te2 = ppval(pp(1,2).TE, ta)
```

XLS-DATA FILES:

The XLS data files have the same name as the T-files, are stored in the same folder (pos), and there is one XLS file per trajectory.

XLS-data files have three main blocks (**INFO, TRJ, AC**), containing meta-information, the trajectory data and the acoustic call data.

META-INFOrmation: recording number, trajectory number, session folder name, date & time, location, recorded species, number of trajectories in the recording, temperature, relative humidity, sound speed for T and RH, sound speed as used by the TOADSutide, sampling rate, recording gain.

TRJ-Data: call arrival time at microphone, emission time at bat, raw data (X, Y, Z), interpolated data for weak (int_01), medium (int_1) and strong (int_10) smoothing. Per interpolated data block, there are the interpolated positions (X, Y, Z) and three error measures: error = distance between raw and interpolated position (E(raw2int)), distance from bat to microphone (distB2Mic), and relative error (normE, in %) = distance-error normalized to bat-microphone-distance.

ACoustic call data: follow the same structure as the call-data in the T-file (TRJ(1,x).cd): each row is one call, each column is one data point. See above at the description of the call data in the TRJ file for details.

8 Solving TOADSuite Problems

General advice: If the TOADSuite stops working and throws an error in the Matlab command window, read the red error message very carefully! Try to understand where/what the problem is; this will help in figuring out a solution and to find the correct answer here.

Regular problems/solutions:

1. Are there **errors in the format of all files and folder, or typos in the tracking_list**? Check dashes/underscores, date and time format, etc for all files and folders (particularly session folder, weather CSV file), and the data in the tracking list.
2. If there seems to be a **problem with reading/converting the weather data**: Open the weather CSV file and check the format of the date-time-column. It should be "yyyy-mm-dd HH:MM:SS". You can try to manually format all the cells: mark all date-time-cells, right click, Format Cells, select 'Custom' at the bottom of the list on the left, and enter the format (yyyy-mm-dd HH:MM:SS) in the box called 'Type' on the right.
3. If there seems to be a **problem with reading/converting the weather data**: Check that the windows list separator (= CSV file delimiter) is set to comma (see installation 3.10): open Windows Region and Language settings, click on 'Additional Settings…' and check the entry for the List Separator. You can also check again that the date-time-format settings are correct (see installation 3.10).
4. **EXCEL-Error:** If at any point the TOADSuite throws an error about Excel, such as "??? Undefined function or variable "ExcelWorkbook", try it again. Sometimes Matlab has difficulties accessing xls-files, and this error seems to happen regularly on the first run.

9 Appendix

9.1. Switches available in the TOADSuite

The behavior and setting of the TOADSuite are defined by many parameters. Their default value can be overwritten by specifying them in the config-files to adapt the TOADSuite to our need. Here is a list of available switches. Many more (undocumented ones) are available, contact me / Peter.

Kanäle für Click detection und Referenzkanal für XCorr, e-mail vom 26.11.2016 15:45:

In session-config:

```
CH_CDT=n (clickdetektionskanal) und
CH_REF=m (referenzkanal für xcorr)
sind prinzipiell unabhängig voneinaner beliebig auf irgendwelche
Microphone zu legen.

CH_CDT=-1 nimmt automatisch den Kanal, der der Geometrischen Arraymitte
am nächsten ist
CH_REF=-2 setzt automatisch CH_REF=CH_CDT
CH_REF=-1 nimmt als Referenzkanal automatisch das kompatibelste
Laupattern zu allen anderen Kanälen enthält (patternscore) d.h. die
summe der Maxima der Kreuzenergie-normierten aller kreuzkorrelationen
mit allen verwendeten anderen kaneälen ist für diesen Kanal maximal.

Wird der CH_REF anders gewählt als CH_CDT, enstehen daraus gewisse
Schwierigkeiten, daß die Laute deutlich verschoben im CIV-fenster liegen
können liegen, das durch die Timings der Detektionen in CH_CDT definiert
wird. Das wird zwar eigentlich kompensiert, aber dadurch entstehen
wieder besonders leicht in der kreuzkorrelation Kollistionen mit den
Fenstergrenzen des CIV-Fensters, die dann näher an den Patterns liegen
können.

Daher sollte es sehr gute Gründe geben, wenn der Referenzkanal anders
als der Clickdetektionskanal gewählt werden soll.

bei Euch ist standardmägig eingestellt:

CH_CDT=0; d.h. ch0 ist clickdetektionskanal
und
CH_REF=-2; d.h. der als xcorr-referenz wird immer das pattern aus dem
gewähltem clickdetektionskanal verwendet.
```

e-mail 26.11.2016, 19:19:

```
- mktoadpos.m (not really neccessary or important in this context, but I
have implemented an improved handling of microphone-positionfiles with
oddly sorted micro-lines, and set two default values for the
```

crosscorrelation and clickdetection to take the time exactly at the peak, not shortly before: XC_RATIO=1 and THR_RATIO=1. Probably most intuitive this way, but also can be changed in any config file.)

switch for tracking list CSV/XLS

SPG_CHANNELS

Number of microphone digits: channelformat

- Sets the number of digits used for the microphone number in the WAV-filenames
- We use two digits: channelformat = 2
- Set this value in WHICH CONFIG FILE?
- Changes behavior of mktoadpos and starttoadpos
- Emails from 30.11.2016, 21:40

9.2. Questions and explanations about the TOADSuite by Holger and Peter, e-mail 02.12.2016

Und die Fragen - Wenn Du mal Zeit hast, würde ich mich über Antworten freuen. Muss nicht gleich sein, da das vielleicht länger dauert. Los geht's:

1. SPG_CHANNELS / CHANNELS_READ / CHANNELS_USE:

CHANNELS_READ = Menge der kanäle die Eingelesen wird
CHANNELS_USE = Menge der Kanäle die zur Positionsberechnung verwendet wird

bitte vorsichtshalber CHANNELS_READ und CHANNELS_USE identisch setzen, also bei Euch entweder

CHANNELS_READ=[0:7];
CHANNELS_USE=[0:7];

oder

CHANNELS_READ=[0:3];
CHANNELS_USE=[0:3];

SPG_CHANNELS = Menge der Kanäle, die auf tastendruck von chktoadpos aus als externes spectrogram dargestellt werden. Nur hier relevant.

Sofern sinnvoll werden die Mengen Durch sukzessive Masken eingeschränkt, also die datsächlich benutzen Kanäle können beispielsweise nicht mehr sein als CHANNELS_READ, auch wenn CHANNELS_USE so gesetzt wäre.

In toadconfig hatten wir SPG_CHANNELS auf [0:3] gesetzt. Soll das für den 8-mic array auf [0:7] angepasst werden? Oder hat das für uns keinen Einfluss, da session_config die relevante config-Datei ist (oder müsste SPG_CHANNELS sogar in die session_config rüber und gesetzt werden)?

In session_config gibt es CHANNELS_READ und CHANNELS_USE: sollen wir die ebenfalls auf [0:3] setzen für den kleinen und [0:8] für den großen array?

```
Es ist noch viel mehr anders möglich, aber ich umreiße nochmal kurz das
Prinzip und wie die Nutzung Eurer derzeitigen Konfiguration vorgesehen ist.

Es werden der Reihe nach verschiedene Konfigurationsdateien gelesen, die
früheren sind meist längere Zeit konstant, die späteren sukzessive auf
session und sequenz spezialisierter. Später gesetzte werte überschreiben
frühere. wenn früher gesetzte werte nicht mehr überschrieben werden,
bleiben sie gültig.

starttoadpos.m liest nur toadbaseconfig.m, testet ob bereits
sessionconfig.m und toadbaseconfig.m im individuellen sessionconfigdir/
liegen, und kopiert anderenfalls diese dateien aus global/ dorthin. dann
wird sessionconfig.m dort ausgewertet.

starttoadpos erzeut weiterhin automatisch für jede sequenz einer session
ein individuell angpasstes autosequenceconfig.m (das beispielsweise
namen und abschwächung enthält) und ein manualsequenceconfig, in dem die
konfiguration für jede einzelsequenz nochmal individuell angepasst
werden kann.

nach abfrage der detektionsschwelle zykliert startoadpos.m dann
mktoadpos.m durch alle sequenzen der session.


mktoadpos.m liest alle der folgenden konfigurationsfiles:


0) toadbaseconfig.m: Der grundlegende Ablauf des Programms wird erstmal
in toadbaseconfig in bin/ festgelegt, unter anderem welche
konfigurationsdateien in welchen Verzeichnissen überhaupt gelesen
werden, andererseits, welche Dateien von global/ ins individuelle
sessionconfigdir/ kopiert werden.

Diese Datei wird zuallererst gelesen. Sie befindet sich als einzige in
bin/ - im executeable-path, da die anderen Verzeichnisse erst in ihr
konfiguriert werden.

Der folgende Ablauf gilt für Eure derzeitige Konfiguration in
toadbaseconfig.m


1) toadconfig.m: anschließend wird toadconfig.m im sessionconfigdir/
gelesen.
(bei anderen Nutzern ist toadconfig.m die einzige allgemeine
konfigurationsdatei, aufgrund von automatischer konfiguration, kopieren
von konfigdateien, dateisystemstruktur, sessions/sequenz, abarbeitung
mehrerer aufnahmen pro aufruf bei Euch waren weitere dateien erforderlich.

toadconfig.m im sessionconfigdir/ ist (neben toadbaseconfig.m und
toadnameconfig.m) die einzige konfigurationsdatei, die von chktoadpos
gelesen wird, und sollte bei Euch in erster linie zur Konfiguration von
chktoadpos.m verwendet werden.
Hierzu zählen beispielsweise Tastenbelegungen, oder SPG_CHANNELS;

2) autosequenceconfig-*.m: (im sessionconfigdir/)
automatisch von starttoadpos.m für jede sequenz erstellt. nicht zum
editieren gedacht.

3) sessionconfig.m: (im sessionconfigdir/)
konfiguration der gesamten session. wird sofern vorhanden von
```

```
starttoadpos.m nicht berührt, anderenfalls wird das defaultmuster dieser
datei aus dem globaldir/ hierher kopiert.

die sessionconfig im globaldir/ stellt die defaultkonfguration für die
normalerweise verarbeiteten sessions dar.
Dieses ist Eure Haupt-konfigurationsdatei für die gegenwärtig
durchgeführten Analysen

die sessionconfig im sessionconfigdir/ stellt die konkrete für diese
session angewandte und von Euch ggf. angepasste konfiguration dar.

4) manualsequenceconfig.m (im sessionconfigdir/)
beinhaltet ggf.für eine bestimmte sequenz der session manuell optimierte
werte
(z.b, anderer THR_CDT, Filter, etc.)



Die meisten werte könnt Ihr in einer Beliebigen datei setzen und ggf. in
einer späteren nochmal neu setzen und überschreiben.

toadnameconfig.m (in global/) ist seperat und beinhaltet nur die
artnamenskonfiguration von chktoadpos.m

toadbaseconfig.m (in bin/) bestimmt den grundlegenden ablauf.

toadconfig.m (in global/) soll bei Euch in erster linie für die
Konfiguration von chktoadpos.m verwendet werden.
hier solltet Ihr
CHANNELS_SPG=[0:7]; (oder eine kleinere Menge, wenn Ihr nicht alle
kanäle als externes spektrogram ausgegeben bekommen wollt) anpassen.

sessionconfig.m (in global/) ist eure standardkonfigurationsdatei für
die toadanalyse.

sessionconfig.m und manualsequenceconfig.m (im sessionconfigdir/) können
nochmal individuell für die btreffende session oder sequenz angepasst
werden.



Ihr könntet also allgemeine Defaulteinstellugen für einen Arbeitsplatz
in toadbaseconfig.m setzen, etwa

CHANNELS_READ=[0:3];
CHANNELS_USE=[0:3];

und dann für die derzeit bearbeiteten daten in der sessionkonfig.m auf

CHANNELS_READ=[0:7];
CHANNELS_USE=[0:7];

(zuzüglich weiterer infos, etwa CH_CDT, CH_REF, CHANNELFORMAT,
Bandpassfilter,etc)

umstellen.


CHANNELS_SPG=[0:7]; muss ggf. in toadconfig.m angepasst werden.
```

```
2. nur zur Bestätigung: die array- AGM-files müssen ins global-dir,
korrekt?

Korrekt. (Dort liegen config-Dateien etc., die immer wieder verwendet
werden)


3. XIV/CIV in sessionconfig: um sicher zu gehen: XIV ist das Intervall
des Musters (im Referenzkanal), CIV das Intervall für alle anderen
Kanäle, korrekt? CIV sollte also um ca ± maxlag größer sein als XIV,
korrekt?

Genau!


4. Was bedeutet TRACKINGLISTFORMAT_XLS = false: dass tracking_list-CSV
files gelesen werden? Was muss ich setzen, damit XLS-files gelesen werden?

wenn xls-files gelesen werden sollen, muß der wert auf "true" gesetzt
werden.


5. CALL ANALYSIS (smalltoadcallanalysis):

5.1 wird der BP von mktoadpos hier genutzt? Also, nutzen wir den BP der
smalltoadcallanalysis nur, wenn wir noch weiter filtern wollten?

ja.

es gibt für mktoadpos 2 Filter

BANDPASS_CDT_HP = 20000;
BANDPASS_CDT_LP = 90000;
BANDPASS_HP = 20000;
BANDPASS_LP = 90000;

Der CDT-Bandpass wird nur zur clickdetektion verwendet.

Der allgemeine Filter wird auf die Signale angewendet, die in der
Kernsignalmatrix in mktoadpos abgespeichert wird, auf deren Basis dann
die Lokalisation vorgenomen wird.
Das smalltoadcallanalysis.m-programm operiert auf dieser
Kernsignalmatrix, in der die bereits zurechtgeschnittenen und
Bandpassgefilterten Signale vorliegen.

Engere Filter zur Lautanalyse werden (derzeit) durch editieren des
Programms smalltoadcallanalysis.m gesetzt. (das kann auch nochmal
umgestellt werden, Nachteile können wir nochmal telefonisch besprechen.
Umstellung aber lieber erst ab Februar.)

relevante Parameter für den Zusaztlichen Bandpass in
smalltoadcallanalysis.m werden dort in diesen Zeilen gesetzt:

DO_BANDPASS=false;
%DO_BANDPASS=true;
BANDPASS_HP=20000;
BANDPASS_LP=90000;
```

5.2 wie funktiniert die Analyse? Ich sehe, dass mit Hilfe des THRESHOLD_LEVELS Vektors verschiedene Start- und End-zeiten berechnet werden basierend auf dem enveloep. Welche Min- und Max-Frequenzen werden aber ausgewertet - wird dazu auch derselbe THRESHOLD_LEVELS Vektor genommen, um dann -X dB below peak frequency zu gehen? Oder berechnest Du die peak-frequency zu den verschiedenen Start- und End-zeiten?

5.3 REFERENCE_LEVEL_IS_MAX: das scheint mir eine Referenz-zeit zu sein. Welche genau, was muss ich beachten, um diesen Parameter zu setzen?

1a) Bandpassfilter
1b) Berechnung hilbert envelope
1c) optional smoothing der envelope entspechend Zeitangabe T_SMOOTH;

Darüberhinaus wird ein Spektrogramm entsprechende den parameterangaben berechnet und und Zeitlich mit ausgangssignal und hilbertamplitude zru Deckung gebracht.

Der Zeitpunkt der clickdetektion ("cdt") im Signal ist gegeben.

Es wird ein Referenzpunkt ("ref") im Signal festgelegt.
Wenn REFERENCE_LEVEL_IS_MAX=true; wird der Zeitpunkt mit der größten hilbert-envelope ("ref"="max") im signal genommen (was ich für das beste standardverhalten halte), anderenfalls der Zeitpunkt der clickdetektion ("ref"="cdt").

"max" ist definiert als der Zeitpunkt der größten hilbertenvelope zwischen "cdt" und dem ersten Zeitpunkt, zu dem die envelope wieder auf einen geringeren pegel als bei "cdt" gefallen ist.

Sampleindex, Zeit, Frequenz, Amplitude, Pegel dieses referenzpunktes "ref" dienen ans referenz für weitere Berechnungen.

In Tabelle:

Ausgegebene Sampleindices ("i_*" Tabelleneinträge) und Zeiten ("t_*" Tabelleneinträge) sind allgemein schlichtweg über die samplingrate verbunden und prinzipiell redundant. Ebenso sind "mxamp" und "maxampdb" beziehunsgweise "amp" und "level"-werte jeweils schlichtweg über level|dB=20*log10(amp) verbunden.

1. Block:

n_click (clicknummer)
*_cdt (zeitpunkt clickdetektion)
*_ici (differenz aufeinenderfolgernder clickdetektionszeitpunkte)
mxamp, mxampdb (maximalamplitude und pegel)
dist (Distanz des lokalisierten Punktes vom betrachteten Kanal)
g_att (geometrische abschwächung über diese Distanz)

werden aus den Daten von mktoadpos übernommen. Die letzen beiden werte sind als hilfestellung gedacht um absolute Pegel (aus gometrischer abschwächung und mxampdb) schnell grob abschätzen zu können

2. Block:

für die Zeitpunkte "cdt", "max" und "ref" (je nach dem wie dieser definiert wurde) wird jeweils
i (sample),
t (zeit)
amp (amplitude)
level (dB-Pegel)

ausgegeben.

3. Block

für jeden der angegebenen thresholdlevel wird der erste und letzte Zeitpunkt gesucht, zu dem die hilberteinhüllende zum ersten mal über "ref"level + threshold gestiegen ist ("init"), bzw. diesen das letzte mal überschreitet ("term"). Diese Zeitpunkte definieren Beginn und Ende des Lautes entsprechend diesem Pegelkriterium.

Für die 3 Zeitpunkte "init", "ref" und "term" wird jeweils berechnet:

i (sample),
t (zeit)
amp (amplitude)
level (dB-Pegel)
frq (frequenz) - maximum aus spektrogram zu diesem Zeitpunkt.

Zusätzlich wird die Differenz (bzw. für amp der Quotient) der Werte von f("term") - f("init") berechnet, und unter der Spalte "diff" abgelegt. Dieses entspricht der "Lautdauer", "Bandbreite" (vorzeichen!), etc. für dieses Threshold-Kriterium.

5.4 werden die settings der smalltoadcallanalysis irgendwo gespeichert? Wenn ja, wo?

Nein, bislang nicht. Ist derzeit auch nicht als detaillierte Lautanalyse, sondern nur als schnelle Hilfsanalyse zur Lautauswahl konzipiert. Vielleicht sollte die Konfiguration anderenfalls doch mittelfristig in die anderen Konfigurationsfiles übertragen werden.

Für die meisten Konfigurationswerte werdet Ihr vermutlich recht daurhafte Standardwerte benutzen, aber der Bandpassfilter könnte durchaus sinnvollerweise flexibel gesetzt werden.

6. Wie nutze ich die auto/manual-configs nochmal? Starttoadpos erzeugt die auto- und manualsequenceconfig-Datein für alle recordings - aber warum zwei verschiedene und nicht nur eine? In der manual-config kann ich andere Filtersettings einstellen; was mache ich mit der auto-config?

autoconfig.m in ruhe lassen, manualconfig nach belieben anpassen.

(Die auto-config ist die zuerst in mktoadpos eingelesene config-datei und beinhaltet in erster linie sequenzspezifische namensangaben und pfade, sowie die schallgeschwindigkeit für diesen Zeitpunkt der sequenz.

Grund:
Die editierbare sequenceconfig.m muß nach der sessionconfig.m eingelesen

werden, damit ihre Werte diejenigen der Sessionconfig überschreiben können.

mktoadpos.m muß aber mit einer konfigurationsdatei gestartet werden,
welche die Sequenzspezifische Information beinhaltet, welche weiteren
konfigurationsdateien für die aktuelle sequenz zu lesen sind. Dieseskann
die sessionconfig.m nicht leisten, da diese ja identisch für alle
sequenzen einer session ist. Daher kann mktoadpos nicht mit der primären
Konfigurationsdatei sessionconfig.m aufgerufen werden; ansonsten wüßte
mktoadpos nicht, welche sequenz er session nun zu behandeln ist.

Umständlich, aber daher in mktoadpos eingelesen:
1) toadconfig.m (ggf. für kompatible einstellungen mit chktoadpos.m)
2) autosequenceconfig-*.m
3) sessionceconfig.m
4) manualsequenceconfig-*.m

Könnte ich in manualconfig prinzipiell alle Parameter setzen, die ich
auch in der session-config einstelle?

ja.

9.3. Explanations about the extended smalltoadcallanalysis by Peter, e-mail 29.08.2017

Hallo Holger,

ich hab jetzt ein neues smalltoadcallanaysis inclusive einiger integrierter
nebenprogramme geschrieben.

Die Wetter-Daten etc. werden irgendwo aus dem Filesystem hoffentlich gefunden,
zusammengetragen und weitergeleitet, atmospaherische und geometrische
Abschwächung für ein Referenzmicro rückgerechnet, die mehrdimensionale
Richtcharakteristik und Frequenzgang korrekt entsprechend einer ganzen Reihe
Koordinatentransformationen und interpolationen über die fft vor und zurück
kompensiert.

Es werden nun die Parameter für die Quellaussendung (rückgerechnet auf 1m,
atmosphaerische abschwächung vollständig entfernt).

Kalirationsdatei des Microphons: beliebig benannt, inhalt mcmstruct:

```
 >> load microcalib-tristar-60-001-ch0.mat;mcmstruct

mcmstruct =

     frq: [20000 30000 40000 50000 60000 70000 80000 90000 100000]
     rad: [0 0.1250 0.2500 0.3750 0.5000]
     rot: [0 0.1250 0.2500 0.3750 0.5000 0.6250 0.7500 0.8750]
       A: [9x5x8 double]
```

```
mcmstruct hat obige vier felder,
- die abtastfrequenzen [hz]
- die radiärwinkel von der Microphonachse Weg
  [winkel normalisiert auf Vollwinkel; 1 =~360°]
- die rotationswinkel, 0 ist oben, von da aus math. positiver drehsinn in
ausrichtung des Microphones
- die Sensitivitäts-messmatrix mit empfindilichkeiten in -dB in passender größe

--------------

Konfiguration von smalltoadcallanalysis weiterhin im Programm selber.
Neue relevante parameter in user configuration:

* THRESHOLD_LEVELS --- wie gehabt
* ENERGY_QUANTIL --- die Energiequantile, die für die entsprechenden Parameter
berücksichtigt werden
* DO_CALCULATE_SOURCE_SIGNAL=true; --- schaltet die Quellparametergberechnung an
* SENS_MIC, SENS_AMP, SENS_DIG, SENS_CORR (in dB) ergeben summiert mit der
Kalibrationsmatrix aus mcmstruct die absolute Empfindlichkeit der Microphone re 1
wav_fullscale / 1 Pa.
* SENS_P_REF gibt an, ob die Ergebnisse gegen 1Pa, oder 2*10^-5Pa (dB SPL)
referenziert sein sollen.

Weiterhin wird Name und Pfad der Kalibrationsdatei angegeben und der Fragliche
Kanal festgelegt.


% user configuration

     THRESHOLD_LEVELS=[-6 -20]; %dB
     ENERGY_QUANTIL=[.05 .95];

% [...]

%   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% advanced configuration

     file_in=([analysispath filenamecore FILE_IN_SUFFIX]);
     file_out=([analysispath filenamecore FILE_OUT_SUFFIX]);

     fprintf(1,'smalltoadanalysis started\n');
     fprintf(1,'extracting call params of %s\n',file_in);

     D=load(file_in);

     CHANNEL=D.ch_cdt;
     % DO_CALCULATE_SOURCE_SIGNAL=false;
     DO_CALCULATE_SOURCE_SIGNAL=true;
     MICROCALIBRATION_PATH=D.GLOBALPATH;
     MICROCALIBRATION_FILE='microcalib-tristar-60-001-ch0.mat';
     SENS_MIC=0; % dB gain microphone re 1_V/1_Pa
     SENS_AMP=0; % dB gain re 1_V_out/1_V_in
```

```
     SENS_DIG=0; % dB gain of digitizer re 1_wav_fullscale/1_V_in
     SENS_COR=0; % dB gain correction microphone
     SENS_P_REF=1; % reference pressure P_ref/Pa
     %SENS_P_REF=2e-5; % reference pressure P_ref/Pa

%   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


-----------------------------------

Die Ergebnisse werden weiterhin in
*-toadcallparams.csv in eine Tabelle geschrieben.

Jede Zeile ein Laut

für jeden definitions-threshold für die Laute (etwa -20dB, -40dB)
wird mit obiger konfiguration der [.05 .95] Energie-Quantil hinausgeschnitten.

Es sind neue Spalten hinzugekommen, Angaben für die Quantil-Anfangs- und
Endzeitpunkte beinhalten.

% each 7 reference points [init, init_q reference(cdt|max), ...
% ... term, term_q, call_difference, call_quantil_difference]

als neuer Parameter ist für Jeden Zeitpunkt auch der zu diesem Zeitpunkt
zurückliegende Energiequantil des Lautes angegeben.

Abschließend ist für jeden Threshold-Block noch ein 8-Spalten-Block Angefügt mit

[ Gesamtenergie_Laut | RMS_Laut ]
x
Angabe in [ Absolutamplitude(ggf Pa) | dB ]
x
[ Bezogen_auf_Gesamtlaut | bezogen_auf_Quantilausschnitt ]

-------------------------

function out=smalltoadcallanalysis(analysispath,filenamecore)
% .
% out=smalltoadcallanalysis(ANALYSISPATH/,FILENAMECORE)
%
% reads ANALYSISPATH/FILENAMECORE-mktoadpos-out.mat and does automated simple
call analysis
% and writes results to table ANALYSISPATH/FILENAMECORE-toadcallparams.csv
%
% version 2017-08-27
%
% output table format:
%
% 4 header lines
%
% each row a call
%
```

```
% columns:
%
%  1    click number
%  2    detection sample
%  3    detection time
%  4    interclickinterval samples
%  5    interclickinterval time
%  6    max wav amplitude
%  7    max wav level dB
%  8    source distance
%  9    geometric att dB
%
% 10:21
% 3 reference sample blocks: [@sample_cdt, @max_intensity @reference_sample]
% x each 4 parameters: [win-sample, win-time, wav-amplitude, wav-level_dB]
%
% 22:71 | 72:121 | 122:171 ...
% n threshold blocks [-12dB, -20dB, ..., -40dB] as defined
% x (
% each 7 reference points [init, init_q reference(cdt|max), ...
% ... term, term_q, call_difference, call_quantil_difference]
% x each 6 parameters: [win-sample, win-time, wav-amplitude, wav-level, ...
% ... frequency, cumulative_normalized_call_energy]
% (amplitude-"diffs" denote amplitude ratios)
% +
% each 2 call parameters: [ mean_rms, overall_call_energy ]
% x each 4 measures [full_call(amp), full_call(dB), quantil(amp), quantil(dB)]
% )


%   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

ich habe alle notwendigen Dateien die verändert wurden beigefügt. Fast alle müssen ins "bin/" verzeichnis gehen.

Weiterhin eine Beispielhafte Microphonkalibrationsdatei (-> "global/"), und 2 Skripte, mit denen sich leicht dummy-microkalibrationsfiles für tests erstellen lassen und eine beispielhafte funktionierende sessionconfig (ggf. -> "global/"), die aber je nach Bedürfnissen (Anzahl Microphonstellen, etc ...) angepasst werden muß.

Bei mir wirds nächste Woche sicher wieder sehr knapp; Schreib mir am besten frühzeitig, wann wir telefonieren sollen.

Viele Grüße
Peter

10 Notes

Potential issues:

- **(Too??) many open figures in starttoadpos when many files were recorded**
- **Cannot enter click detection threshold manually**