The Auditory Modeling Toolbox Structure

Vienna, 2nd February 2022

Paths and configuration

```
» amt_start;
» [flags, kv] = amt_configuration;


» amt_stop; %remove all AMT paths, delete vars
```

http://amtoolbox.org/amt-1.1.0/doc/core/index.php

Trade-offs in AMT development

Provide a wide collection of models

- Documentation
- Arbitrary programming languages
- (Nearly) arbitrary licenses

Simplify scientific auditory modeling

- Unification
- Verification
- Large amounts of data
- Long computation times

Documentation 1/2

- Online: **amtoolbox.org/doc.php**
- Generated directly from the code via **mat2doc**
  - Follow the rules when writing your help-section:

    http://ltfat.org/notes/ltfatnote023.pdf

    (pages 4 and 5)
  - existing AMT code: online documentation

    => click **'View the code'** button

Rules for the help section

```
function [out1, …, outn] = <surname><year>(in1,…, inn, varargin)
%<surname><year> short description
%
%   Usage: out1 = <surname><year>(in1,…, inn);
%     [out1, out2] = <surname><year>(in1,…, inn, varargin);
%
%   Input parameters:
%     in1 : description of input 1
```

Rules for the help section

```
function [out1, …, outn] = <surname><year>(in1,…, inn, varargin)
%<surname><year> short description
%
%   Usage: out1 = <surname><year>(in1,…, inn);
%          [out1, out2] = <surname><year>(in1,…, inn, varargin);
%
%   Input parameters:
%   in1 : description of input 1
```

Spacing

Keyword

Everything matters

How to write good AMT help files?

Content:

- Describe all input and output parameters
  - format/data type
  - dimensions
  - physical units
- Long description:
  bullet points preferred
- Add references

```
function [out] = model2022(in)
Short description
Usage:
Input parameters:
Output parameters:
Long Description
Optional input arguments
See also & References
Author
```

Space

Documentation 2/2

- Use **amt_disp** for any text output
  - ‘verbose‘… like MATLAB’s disp()
  - ‘silent’… no output
  - and several more...check the help

```
amt_start(‘silent’);
amt_disp(‘hello’);
amt_stop;
```

Check in the online documentation:

- demo_verhulst2015
- demo_barumerli2021

Programming languages and licensing

Arbitrary programming languages

- **amt_mex**…compiled languages
- **amt_extern**…interpreted languages and system messages

(Nearly) arbitrary licenses

- licenses need to be compatible with the GPL3
- **amt_info**… display license information

```
amt_info('modelname');
```

Verification: Model and data rankings

Not everything is perfect.

Check the model status on:

http://amtoolbox.org/models.php or via

```
amt info(`modelname`);
```

Peripheral models	Function	Doc	Code	Verification
Gammatone filterbank	gammatone	✔	✔	✔
Linear filtering for monaural masking (basic)	dau1996	○	✔	?


unknown/OK

good

untrusted

perfect

Trade-offs in AMT development

Provide a wide collection of models

- Documentation: **amt_disp;**
- Arbitrary programming languages: **amt_mex; amt_extern;**
- (Nearly) arbitrary licenses: **amt_info;**

Simplify scientific auditory modeling

- Unification
- Large amounts of data
- Long computation times

Simplify scientific auditory modeling

Unifying the input parameters

...a real example...

```
[output,info] = bruce2018(stim, fsstim, fc, ag_fs,
ag_db, cohcs, cihcs, nl, nm, nh, lls, lms, lhs,
SRL, SRM, SRH, n, spont, tabs, trel, psthbw,
winft, winmr, nrep, rt, fsmod, 'fitaudiogram',
'human', 'varFGn', 'actualPL', 'outputPerCF',
'specificSR');
```

Simplify scientific auditory modeling

Unifying the input parameters

...a real example...

```
[output,info] = bruce2018(stim, fsstim, fc, ag_fs,
ag_db, cohcs, cihcs, nl, nm, nh, lls, lms, lhs,
SRL, SRM, SRH, n, spont, tabs, trel, psthbw,
winft, winmr, nrep, rt, fsmod, 'fitaudiogram',
'human', 'varFGn', 'actualPL', 'outputPerCF',
'specificSR');
```

```
[output,info] = bruce2018(stim, fsstim, fc, 'human')
```

Simplify scientific auditory modeling

```
[flags,keyvals] = ltfatarghelper({},definput,varargin);
```

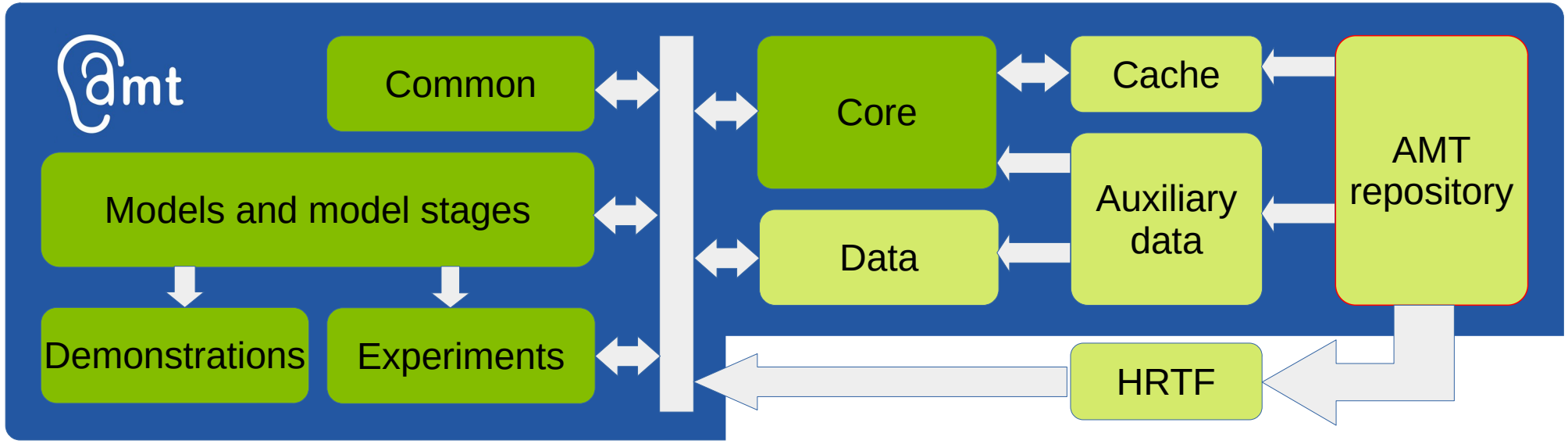
overwrites

- **write struct with flag options and default values**
- definput.flags.adt = {'adt','no_adt'}; => 'adt'
- definput.keyvals.limit=10; => 'limit', 10
- definput.import = 'bruce2018' => from defaults folder

outputs: **2 structs**

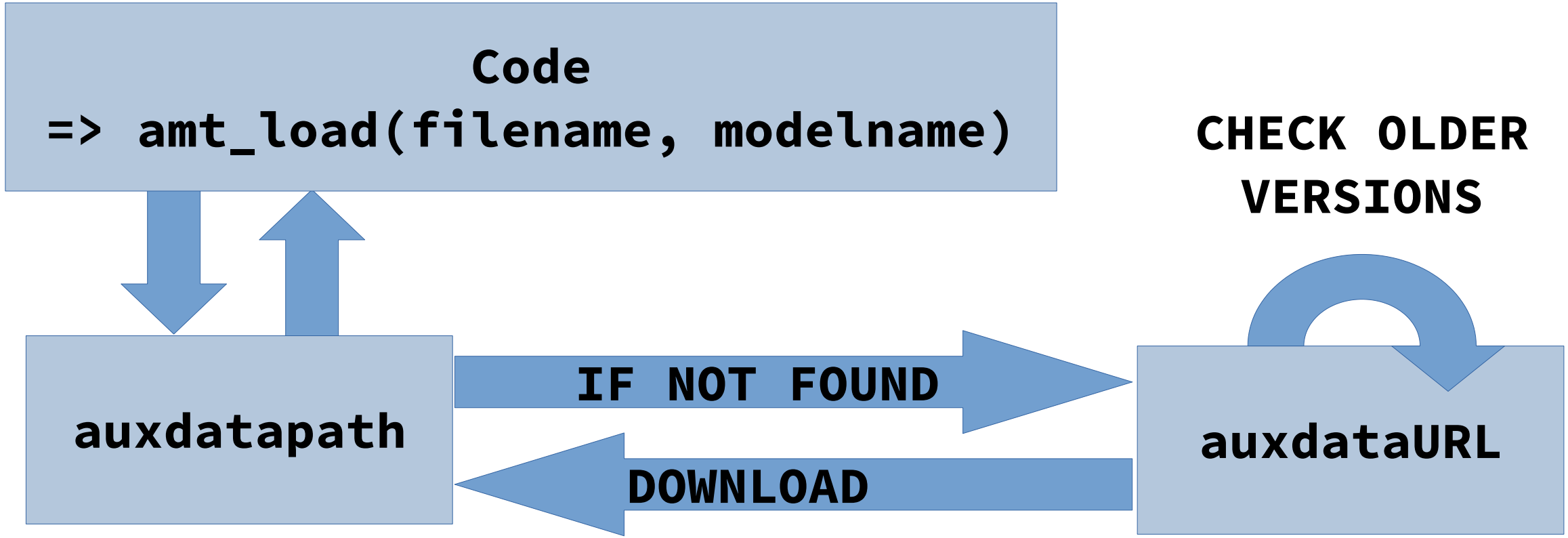
- keyvals: key-value pairs
- flags: flags

Data in the AMT: Summary



- **Data functions:** experimental data
- **HRTF**: handled via the SOFA toolbox
- **Auxiliary data**: large data
- **Cache**: storing data to avoid long computations

Handle auxiliary data: amt_load



basepath/auxdata

http://amtoolbox.org/amt-version/auxdata/modelname

Long computation times: amt_cache

- Works only in functions:
  - Store result: amt_cache('set', package, variables);
  - Get result: var = amt_cache('get', package, flags);
- Flags:
  - 'normal': only recalculate if package not found online
  - 'redo' : always recalculate
  - 'online' : never recalculate

http://amtoolbox.org/amt-version/cache/modelname

Trade-offs in AMT development

Provide a wide collection of models

- Documentation: **amt_disp;**
- Arbitrary programming languages: **amt_mex; amt_extern;**
- (Nearly) arbitrary licenses: **amt_info;**

Simplify scientific auditory modeling

- Unification: **ltfatarghelper**
- Verification: **Model ranking**
- Large amounts of data: **amt_load;**
- Long computation times: **amt_cache;**

Most relevant coding conventions

- Model: <surname_first_author><publication_year>
- Each model needs an experiment
- No hardcoded paths
- No eval and disp => amt_disp()
- Local functions prefixed with ‚local‘
- Cache long calculations
- Put each file into the folder where it belongs
- HRTFs in SOFA format

How to contribute?

1) Have publication accepted
2) Contact us
3) Create files:
   - Model: <name><year> (and modelstages…)
   - Experiment
   - Write documentation in help section
   - Replace paths in code with amt_load, SOFAload
   - Output text via amt_disp (get rid of evals)
   - Helper/common functions: it depends…
   - Demos
4) Once finished, we rank & include code with next release

You should by now...

- Know the AMT core functions and why they are there
- Know the AMT's documentation and coding requirements and where to look them up
- Know how to contribute to the AMT

What questions do you have?