Use of monaural auditory models from the AMT toolbox

AMT Workshop

ENS, PSL University, Paris, France

17 May 2022

Alejandro Osses, Ph.D.

Posdoctoral researcher, ENS Paris, France

ale.a.osses@gmail.com, alejandro.osses@ens.psl.eu

v2, 18/05

What can you expect from this presentation?

- I will show how to use and customise a monaural model of the auditory periphery
- The models:
  - dau1997.m
  - zilany2014.m
  - verhulst2015.m
  - verhulst2018.m
  - bruce2018.m
  - relanoiborra2019_featureextraction.m
  - king2019.m
  - osses2021.m
- In other words you will be able of:
  - Understanding the inputs and outputs of the models
  - How to enable / disable modules of the models (**flags**)
  - How to change parameters of the models (**keyvals**)

Use of **flags** and **keyvals**

Useful references to keep in mind

- Comparison of the eight models: Osses et al. (2022, Acta Acustica)

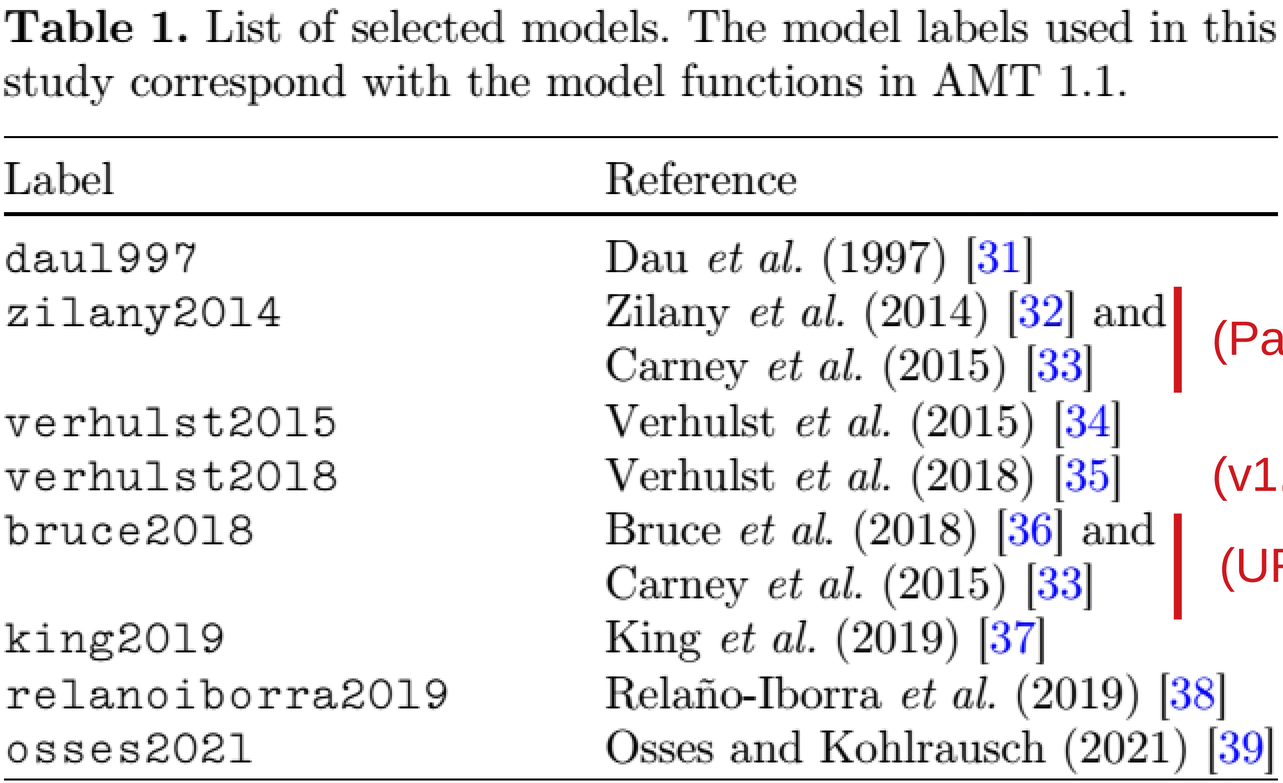


Table 1. List of selected models. The model labels used in this study correspond with the model functions in AMT 1.1.

Label	Reference
`dau1997`	Dau *et al.* (1997) [31]
`zilany2014`	Zilany *et al.* (2014) [32] and Carney *et al.* (2015) [33]
`verhulst2015`	Verhulst *et al.* (2015) [34]
`verhulst2018`	Verhulst *et al.* (2018) [35]
`bruce2018`	Bruce *et al.* (2018) [36] and Carney *et al.* (2015) [33]
`king2019`	King *et al.* (2019) [37]
`relanoiborra2019`	Relaño-Iborra *et al.* (2019) [38]
`osses2021`	Osses and Kohlrausch (2021) [39]

(Paper here)

(Paper by Maxwell et al. 2020 here)

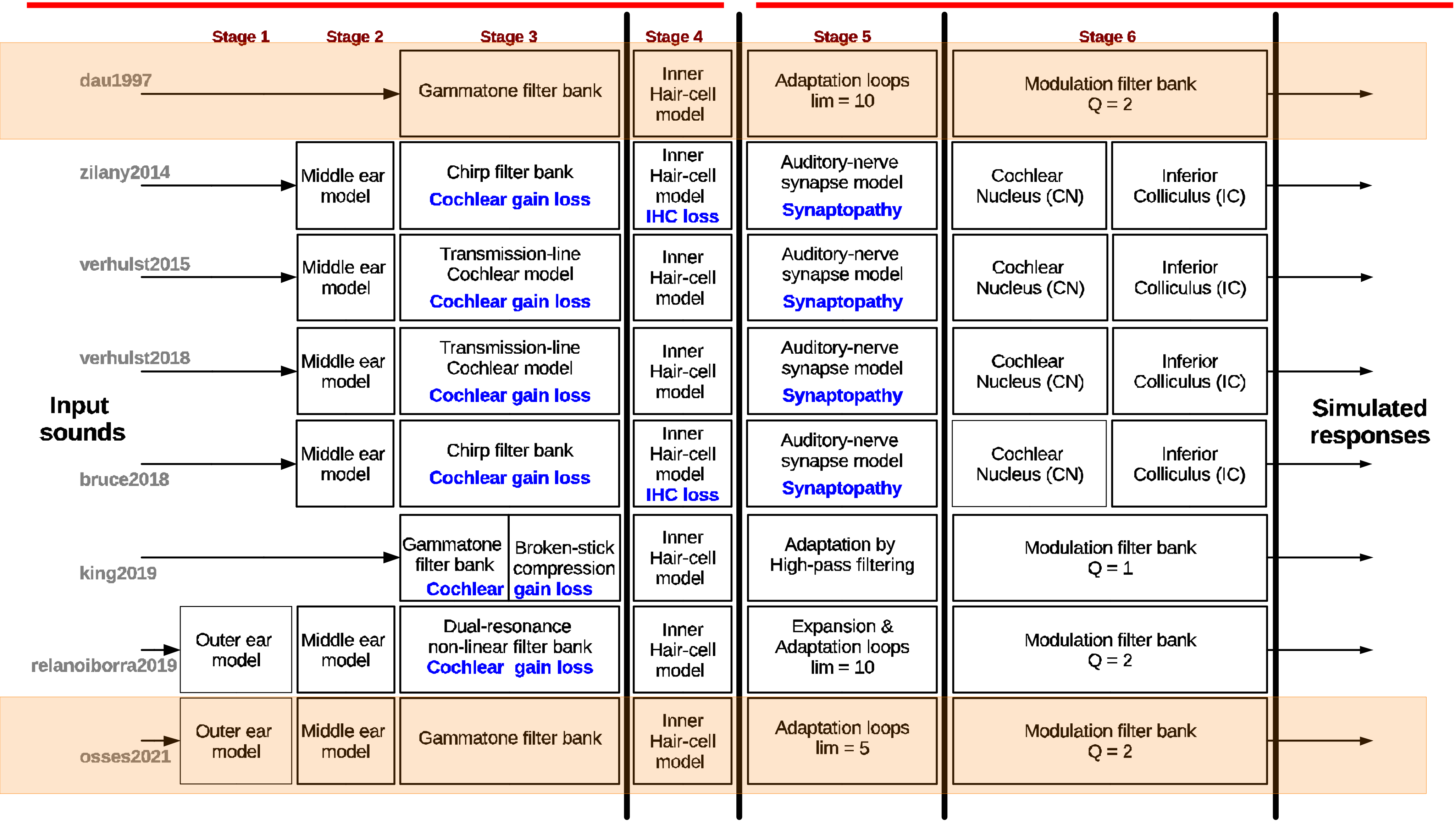
(v1.2 of the model here)

(UR EAR toolbox here)

(Paper here)

(Paper here)

(Paper here)



Input sounds

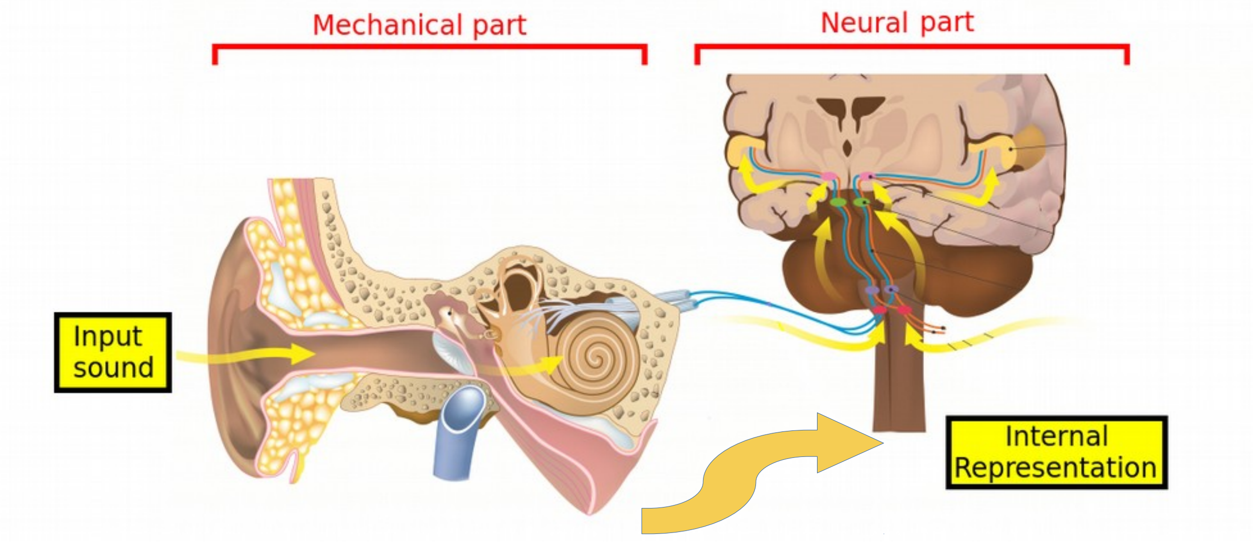
	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6
dau1997			Gammatone filter bank	Inner Hair-cell model	Adaptation loops lim = 10	Modulation filter bank Q = 2
zilany2014		Middle ear model	Chirp filter bank **Cochlear gain loss**	Inner Hair-cell model **IHC loss**	Auditory-nerve synapse model **Synaptopathy**	Cochlear Nucleus (CN) Inferior Colliculus (IC)
verhulst2015		Middle ear model	Transmission-line Cochlear model **Cochlear gain loss**	Inner Hair-cell model	Auditory-nerve synapse model **Synaptopathy**	Cochlear Nucleus (CN) Inferior Colliculus (IC)
verhulst2018		Middle ear model	Transmission-line Cochlear model **Cochlear gain loss**	Inner Hair-cell model	Auditory-nerve synapse model **Synaptopathy**	Cochlear Nucleus (CN) Inferior Colliculus (IC)
bruce2018		Middle ear model	Chirp filter bank **Cochlear gain loss**	Inner Hair-cell model **IHC loss**	Auditory-nerve synapse model **Synaptopathy**	Cochlear Nucleus (CN) Inferior Colliculus (IC)
king2019			Gammatone filter bank **Cochlear** Broken-stick compression **gain loss**	Inner Hair-cell model	Adaptation by High-pass filtering	Modulation filter bank Q = 1
relanoiborra2019	Outer ear model	Middle ear model	Dual-resonance non-linear filter bank **Cochlear gain loss**	Inner Hair-cell model	Expansion & Adaptation loops lim = 10	Modulation filter bank Q = 2
osses2021	Outer ear model	Middle ear model	Gammatone filter bank	Inner Hair-cell model	Adaptation loops lim = 5	Modulation filter bank Q = 2

Simulated responses

Auditory modelling framework

- Extended Dau et al. model:

We will work today with the model internal representations



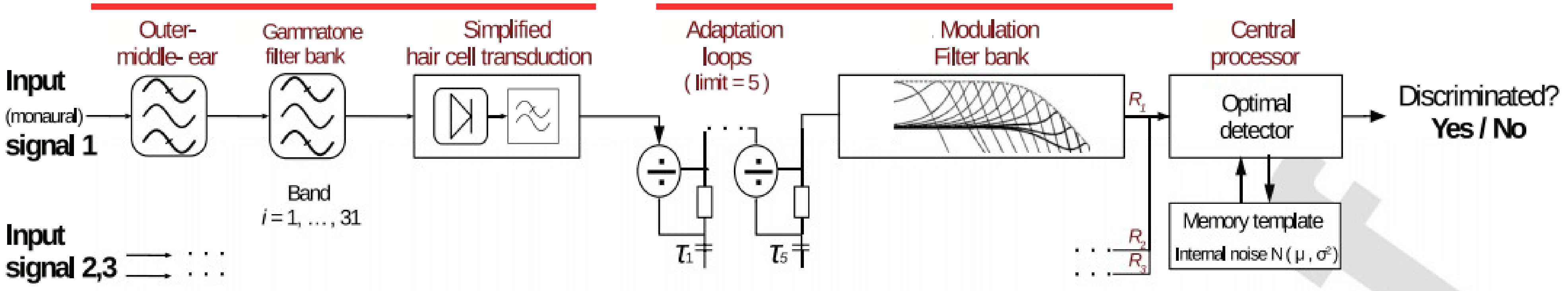
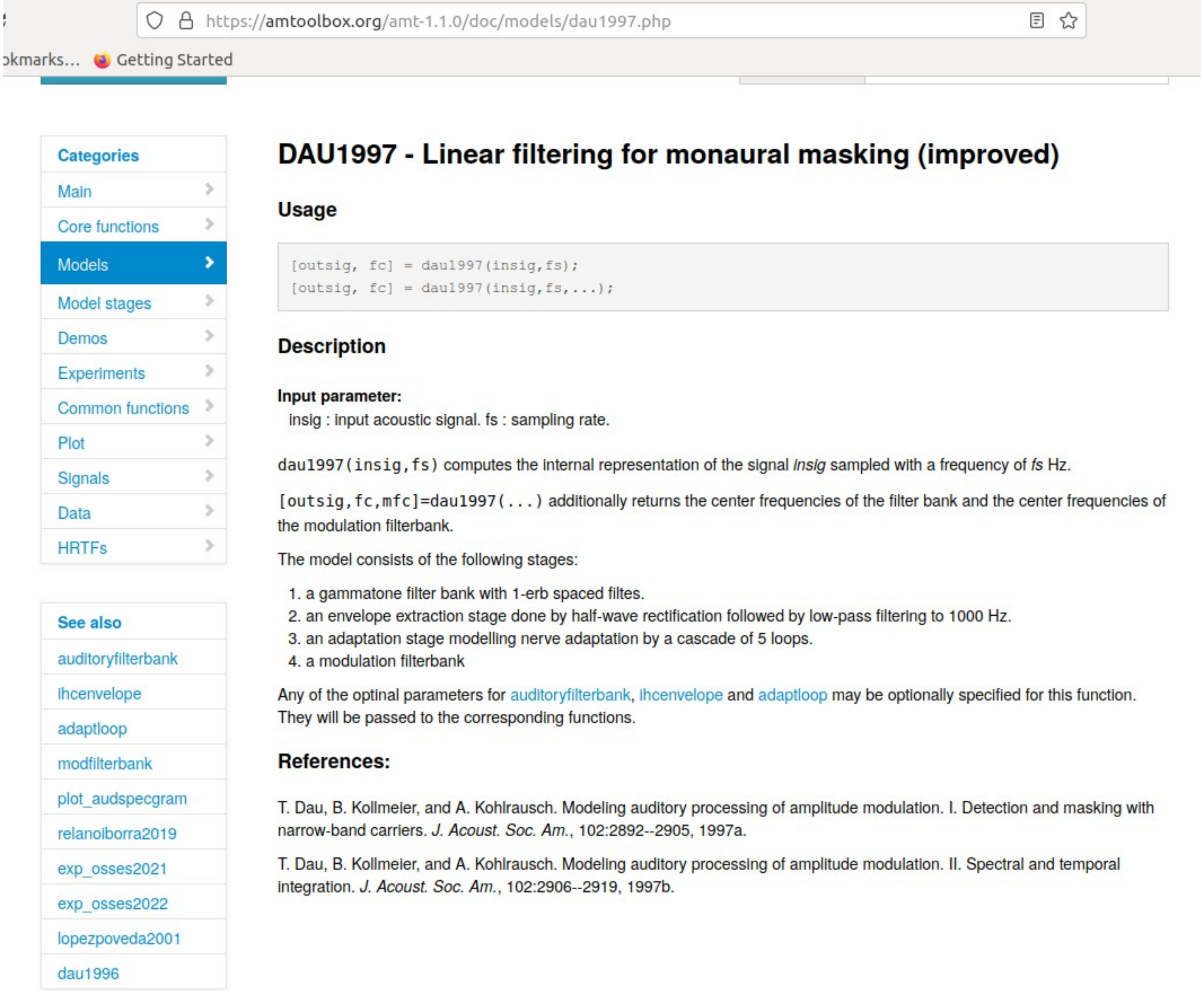


FIG. 1. (Color online) Block diagram of the updated PEMO model. All processing stages and their corresponding set of parameters are explained in the text.

(Figure from Osses & Kohlrausch, 2021, JASA)



DAU1997 - Linear filtering for monaural masking (improved)

Usage

```
[outsig, fc] = dau1997(insig,fs);
[outsig, fc] = dau1997(insig,fs,...);
```

Description

Input parameter:

insig : input acoustic signal. fs : sampling rate.

`dau1997(insig,fs)` computes the internal representation of the signal *insig* sampled with a frequency of *fs* Hz.

`[outsig,fc,mfc]=dau1997(...)` additionally returns the center frequencies of the filter bank and the center frequencies of the modulation filterbank.

The model consists of the following stages:

1. a gammatone filter bank with 1-erb spaced filtes.
2. an envelope extraction stage done by half-wave rectification followed by low-pass filtering to 1000 Hz.
3. an adaptation stage modelling nerve adaptation by a cascade of 5 loops.
4. a modulation filterbank

Any of the optinal parameters for auditoryfilterbank, ihcenvelope and adaptloop may be optionally specified for this function. They will be passed to the corresponding functions.

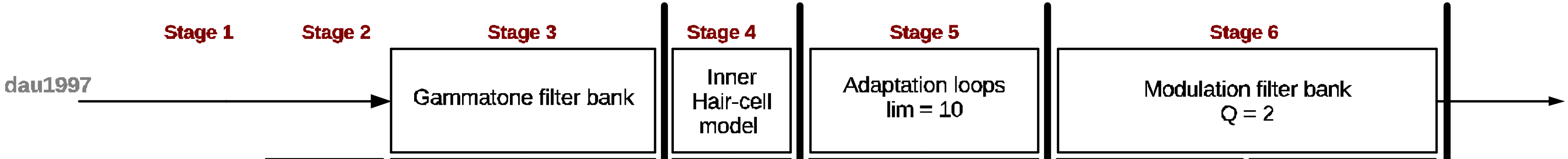
References:

T. Dau, B. Kollmeier, and A. Kohlrausch. Modeling auditory processing of amplitude modulation. I. Detection and masking with narrow-band carriers. *J. Acoust. Soc. Am.*, 102:2892--2905, 1997a.

T. Dau, B. Kollmeier, and A. Kohlrausch. Modeling auditory processing of amplitude modulation. II. Spectral and temporal integration. *J. Acoust. Soc. Am.*, 102:2906--2919, 1997b.

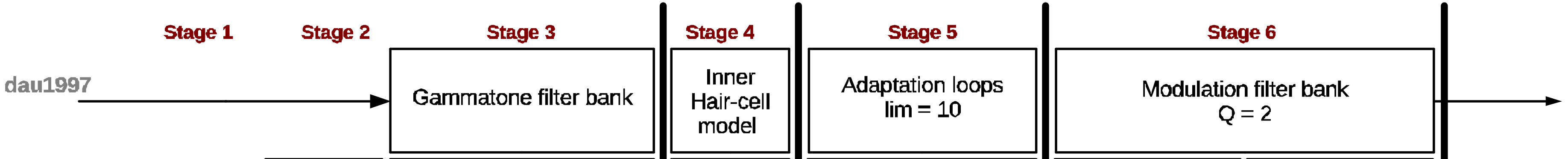
Default parameters for dau1997: Flags

- Outer ear: No
- Middle ear: No
- Gammatone filter bank: Yes (it cannot be by passed)
- Inner hair cell: Yes
- Adaptation loops: Yes
- Modulation filter bank: Yes



Default parameters for dau1997: Keyvals

- Outer ear: No
- Middle ear: No
- Gammatone filter bank: **flow, fhigh, bwmul, basef**
- Inner hair cell: cutofffreq (hard coded), **ihc_filter_order (flag ‘ihc_dau1996’)**
- Adaptation loops: **limit**, **minspl, tau (flag adt_dau1997)**
- Modulation filter bank: **mfc_upper_limit_max (flag mfb_dau1997)**



Stage 1

Stage 2

Stage 3

Stage 4

Stage 5

Stage 6

dau1997

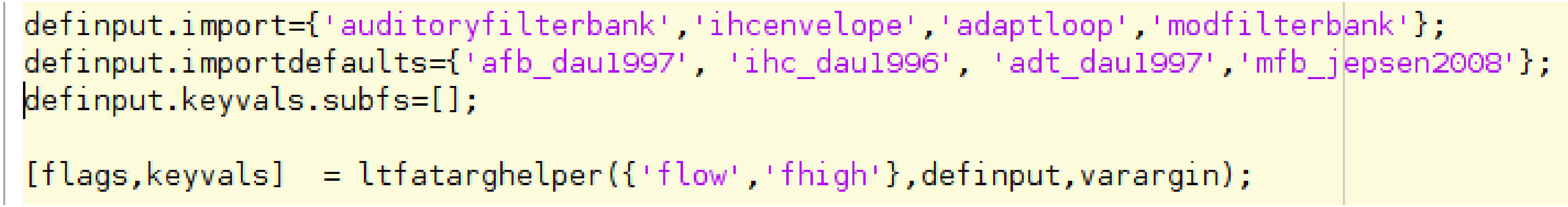
Gammatone filter bank

Inner Hair-cell model

Adaptation loops
lim = 10

Modulation filter bank
Q = 2

Where to find this information?

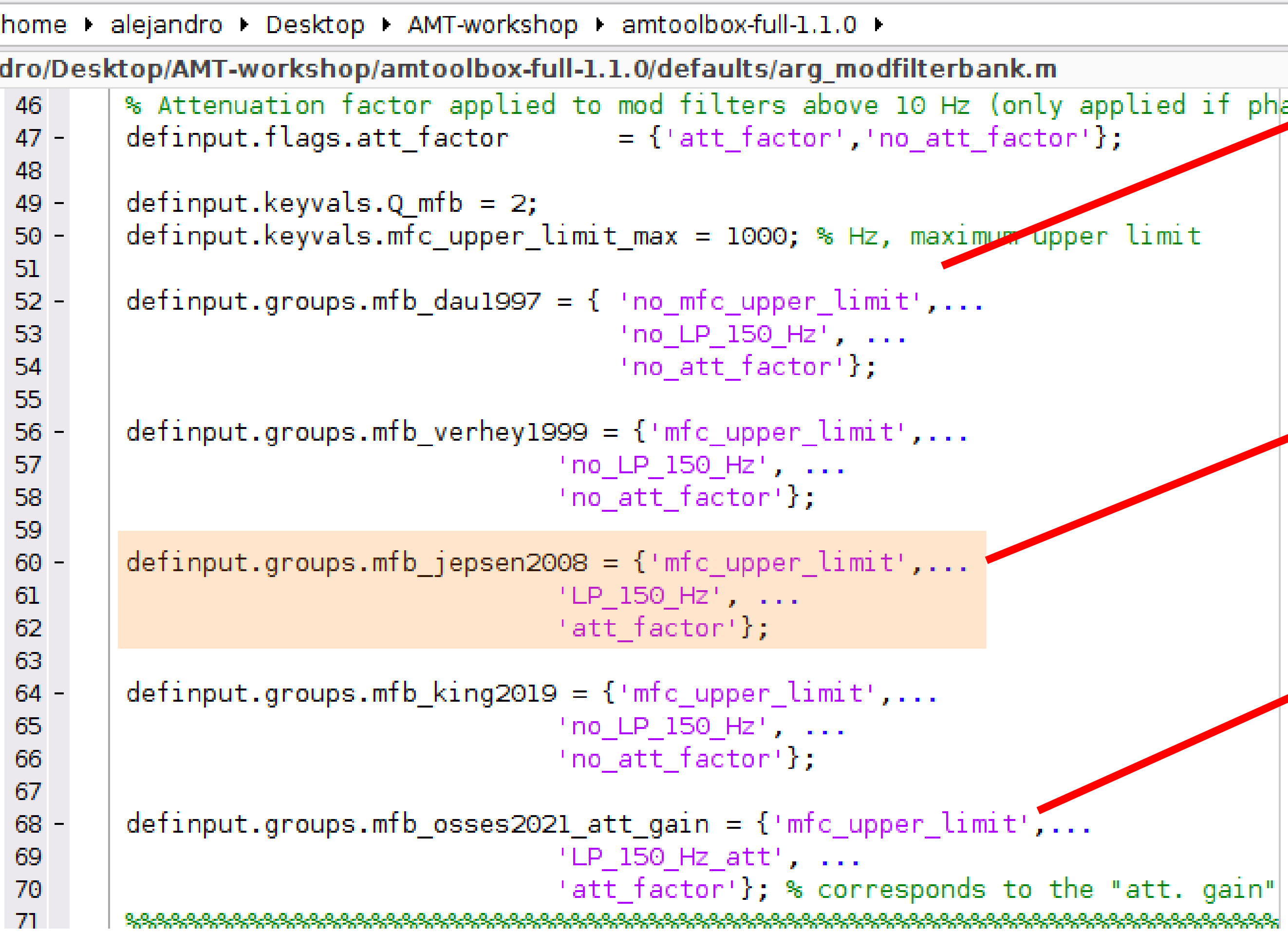


```
definput.import={'auditoryfilterbank','ihcenvelope','adaptloop','modfilterbank'};
definput.importdefaults={'afb_dau1997', 'ihc_dau1996', 'adt_dau1997','mfb_jepsen2008'};
definput.keyvals.subfs=[];

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

“Use these configurations” if other input parameters are not specified:

arg_auditoryfilterbank.m	afb_dau1997
arg_ihcenvelope.m	ihc_dau1996
arg_adaptloop.m	adt_dau1997
arg_modfilterbank.m	mfb_jepsen2008

Group of flags / values



home ▸ alejandro ▸ Desktop ▸ AMT-workshop ▸ amtoolbox-full-1.1.0 ▸

dro/Desktop/AMT-workshop/amtoolbox-full-1.1.0/defaults/arg_modfilterbank.m

```
% Attenuation factor applied to mod filters above 10 Hz (only applied if phase_insens
definput.flags.att_factor       = {'att_factor','no_att_factor'};

definput.keyvals.Q_mfb = 2;
definput.keyvals.mfc_upper_limit_max = 1000; % Hz, maximum upper limit

definput.groups.mfb_dau1997 = { 'no_mfc_upper_limit',...
                                'no_LP_150_Hz', ...
                                'no_att_factor'};

definput.groups.mfb_verhey1999 = {'mfc_upper_limit',...
                            'no_LP_150_Hz', ...
                            'no_att_factor'};

definput.groups.mfb_jepsen2008 = {'mfc_upper_limit',...
                            'LP_150_Hz', ...
                            'att_factor'};

definput.groups.mfb_king2019 = {'mfc_upper_limit',...
                            'no_LP_150_Hz', ...
                            'no_att_factor'};

definput.groups.mfb_osses2021_att_gain = {'mfc_upper_limit',...
                            'LP_150_Hz_att', ...
                            'att_factor'}; % corresponds to the "att. gain" condition fr
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

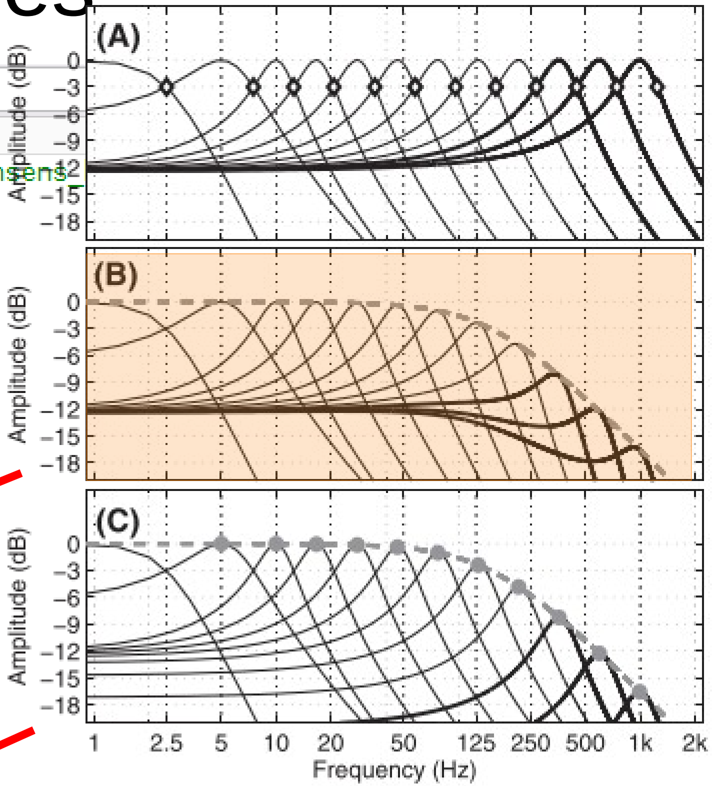
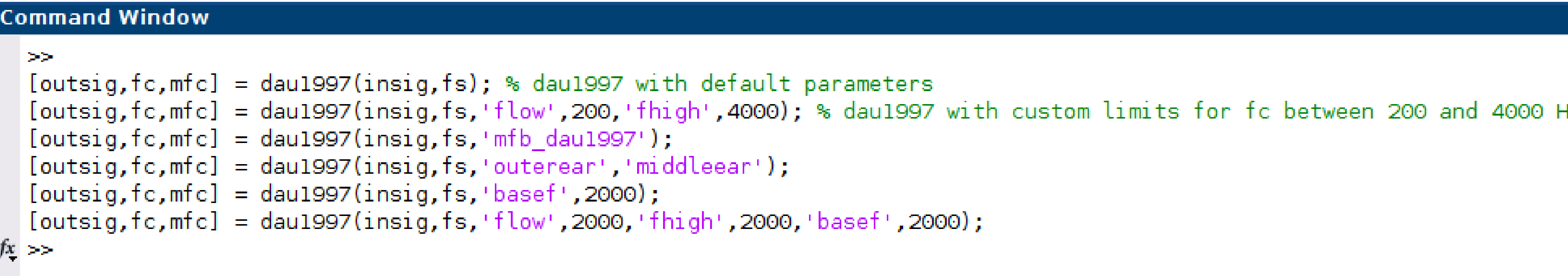


FIG. 14. MTFs of the modulation filter bank without (A) and with (B) the 150-Hz LPF filter. In (C), MTFs using a different implementation of the 150-Hz LPF are shown (see model configuration “mod att. gain,” in the text). The MTFs of filters 10–12, which have frequencies mf_c > 250 Hz, are drawn in thick black lines. The cut-off frequencies of Table II are indicated by open diamonds in (A). The transfer function of the 150-Hz LPF is shown by the dashed gray line.

Running the model with different configurations

dau1997



```
>>
[outsig,fc,mfc] = dau1997(insig,fs); % dau1997 with default parameters
[outsig,fc,mfc] = dau1997(insig,fs,'flow',200,'fhigh',4000); % dau1997 with custom limits for fc between 200 and 4000 H
[outsig,fc,mfc] = dau1997(insig,fs,'mfb_dau1997');
[outsig,fc,mfc] = dau1997(insig,fs,'outerear','middleear');
[outsig,fc,mfc] = dau1997(insig,fs,'basef',2000);
[outsig,fc,mfc] = dau1997(insig,fs,'flow',2000,'fhigh',2000,'basef',2000);
>>
```

The task of today
Process a wave file and obtain its “internal representation”

1) Install the AMT toolbox: https://amtoolbox.org/download.php
2) Pick up one of the eight models
3) Pick up one sound and load it using audioread: suggested names “insig” for the signal, and “fs” for the sampling frequency:

   ababa1.wav (French utterance): Download here

   greasy.wav: type [insig,fs] = greasy;
4) Obtain the “internal representation” by running the model

- Force the model to have a band centred at 200 Hz
- Plot the model output of the band at 200 Hz
- Run the model again but now only up to the adaptation stage (for zilany2014 and bruce 2018 this is already the last stage)

ENS DEC DÉPARTEMENT D'ÉTUDES COGNITIVES

PSL UNIVERSITÉ PARIS

Hands on:
Use of monaural auditory models from the AMT toolbox

AMT Workshop

ENS, PSL University, Paris, France

17 May 2022

Alejandro Osses, Ph.D.

Posdoctoral researcher, ENS Paris, France

ale.a.osses@gmail.com, alejandro.osses@ens.psl.eu

The task of today
Process a wave file and obtain its “internal representation”

1) Install the AMT toolbox: https://amtoolbox.org/download.php
2) Pick up one of the eight models
3) Pick up one sound and load it using audioread, suggested names “insig” for the signal, and “fs” for the sampling frequency:

   ababa1.wav (French utterance): Download here

   greasy.wav: type [insig,fs] = greasy;
4) Obtain the “internal representation” by running the model

- Force the model to have a band centred at 200 Hz
- Plot the model output of the band at 200 Hz
- Run the model again but now only up to the adaptation stage (for zilany2014 and bruce 2018 this is already the last stage)

The task of today
Step by step

1) Install the AMT toolbox: https://amtoolbox.org/download.php

- I recommend “Download the AMT full package” (follow the instructions of the website): i.e., open MATLAB and type amt_start;
- If needed, type amt_mex;

The task of today
Step by step

2) Pick up one of the eight models

- If you pick up verhulst2015 or verhulst2018, note that these models require Python preinstalled and that they are slow models, so at first, use short sounds to be processed.
- zilany2014, bruce2018, verhulst2015, verhulst2018 have many outputs and configurable parameters
- dau1997, king2019, relanoiborra2019_featureextraction, and osses2021 share a similar structure (as I showed earlier in this presentation). **So, if you don’t know which model to use, pick up one of these models.**

The task of today
Step by step

3) Pick up one of sound:

   ababa1.wav (French utterance): Download here

   greasy.wav: type [insig,fs] = greasy; (this option doesn’t require a download)

4) Obtain the “internal representation” by running the model

- Force the model to have a band centred at 200 Hz
- Plot the model output of the band at 200 Hz
- Run the model again but now only up to the adaptation stage (for zilany2014 and bruce 2018 this is already the last stage)

The task of today
Step by step

4) Obtain the “internal representation” by running the model

- Force the model to have a band centred at 200 Hz.

  dau1997 or similar: This requires to specify the keyval ‘basef’

  zilany2014, bruce2018: set the input parameter cf to 200 Hz

  verhulst2015, verhulst2018: use either (1) cf_flag=‘abr’ and then get the closest frequency to 200 Hz, or (2) use cf_flag=200

The task of today
Step by step

4) Obtain the “internal representation” by running the model

- Plot the model output of the band at 200 Hz
- **dau1997** or similar: You get the modulation filter bank outputs, plot one of the bands (the one you want, look at the parameter mfc) or several bands superimposed.
- **zilany2014, bruce2018**: You get the auditory-nerve synapse outputs. For simplicity, I suggest to plot the “mean_rate” outputs
- **verhulst2015, verhulst2018**: You get several inferior colliculus (IC) outputs.
- Run the model again but now only up to the adaptation stage (for zilany2014 and bruce 2018 this is already the last stage)

The task of today
Step by step

4) Obtain the “internal representation” by running the model

- Run the model again but now only up to the adaptation stage.
- For this stage you need to identify the name of the last module, so that you can bypass the flag (e.g., ‘mfb’ needs to be specified as ‘no_mfb’)

Some of the questions during the workshop

Pablo: Is there a way to easily guide us in choosing a specific model for a specific application?

Alejandro: I am afraid that that’s a very difficult question. I recommend to read papers, and when you see an interesting model, check whether there is an implementation available somewhere. My recent paper (Osses et al., 2022) gives a nice overview about monaural models, exploring what each model can do, but we do not compare models in an application context.

If you find in the literature a “nice model” I recommend you to check the list of experiment functions from AMT, because there are quite some models implemented here, and some of the codes to reproduce figures can be retrieved there. I believe this is a good way to check what models can be useful for yourself.

For instance, you can reproduce some of the figures of the DRNL cochlear filter bank using exp_lopezpoveda2001 using, e.g., the argument ‘fig2’. Have a look at other experiments in AMT…

In addition, there are a number of models that have been used in specific applications (e.g., speech: joergensen2011, joergensen2013, relanoiborra2019; binaural models: baumgartner2014, mclachlan2021), have a look at their corresponding demo / exp scripts, maybe some of these models can be interesting for you...

Some of the questions during the workshop

Christian: I am running the model king2019.m but I get an error “not enough input parameters” what is going on here?

Alejandro: You have to keep in mind that the models are in developing phase and they require some compulsory inputs. In particular to this model, I realised there are two small bugs which you can’t identify check the demo or exp scripts first always, because those scripts are always bug-free.

In the case of this model, there is a demo (demo_king2019.m) but no experiment script yet, but what I realised today is that you should always specify one of the following inputs, both related to the characteristic frequencies (CFs) to be simulated:

- ‘flow’, and ‘fhigh’ to specify the minimum and maximum frequencies at which CFs spaced at 1 ERB_N will be used.
- Alternatively, you need to specify the keyvalue ‘basef’ to be set to the centre frequency you want to analyse. Then, two CFs below and 2 CFs above will also be simulated as adopted in your paper together with Nicolas Wallaert (Wallaert et al. 2018, JASA)

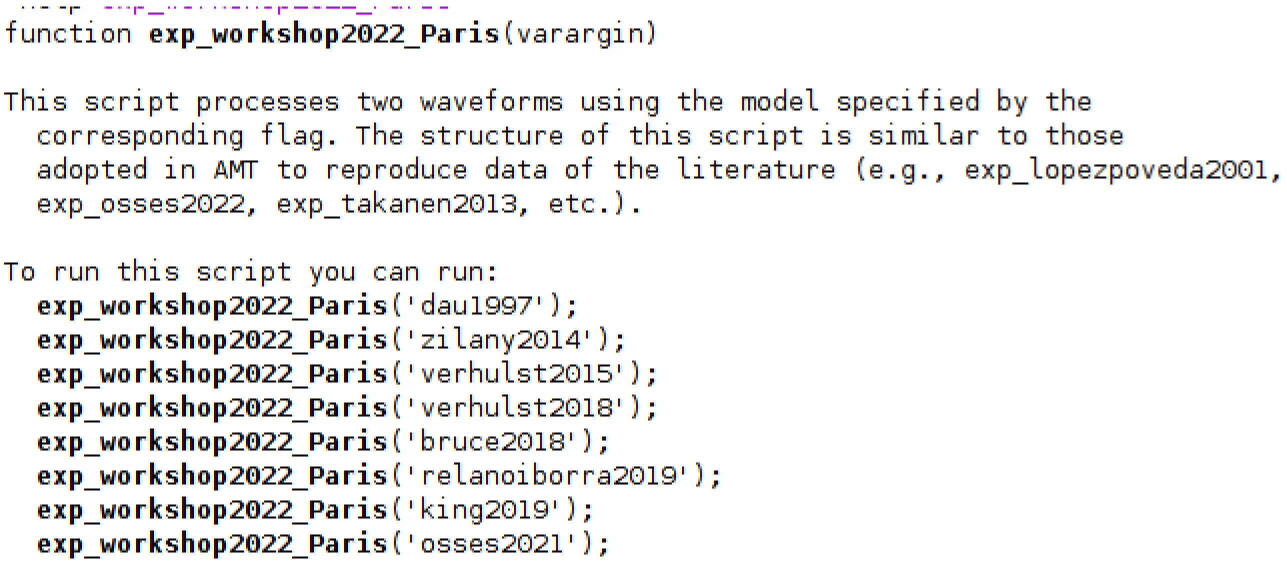
Because it is not so clear in the current implementation that one of these two CFs implementations should be adopted, we will try to fix this in the documentation of the coming king2019.m versions.

Some of the questions during the workshop

- Now that we are talking about ‘basef’, this is a standard parameter in dau1997, relanoiborra2019, king2019, and osses2021. Basef is used as a keyvalue pair to obtain at least one CF centred at that specified frequency.
- Actually, all these models use a fixed frequency step between the simulated CFs of 0.5 ERB_N for relanoiborra2019 or 1 ERB_N for the others, but if no basef is defined, the exact CFs are obtained from the frequency ‘flow’ and ‘fhigh’, normally at fixed steps starting from the closest frequency ‘flow’ such that its value using ERB_N is an integer. With defaults ‘flow’ and ‘fhigh’ of 80 and 8000 Hz, 31 bands between 3 and 33 ERB_N are obtained.

Solution to the activities

- Run the script exp_workshop2022_Paris:

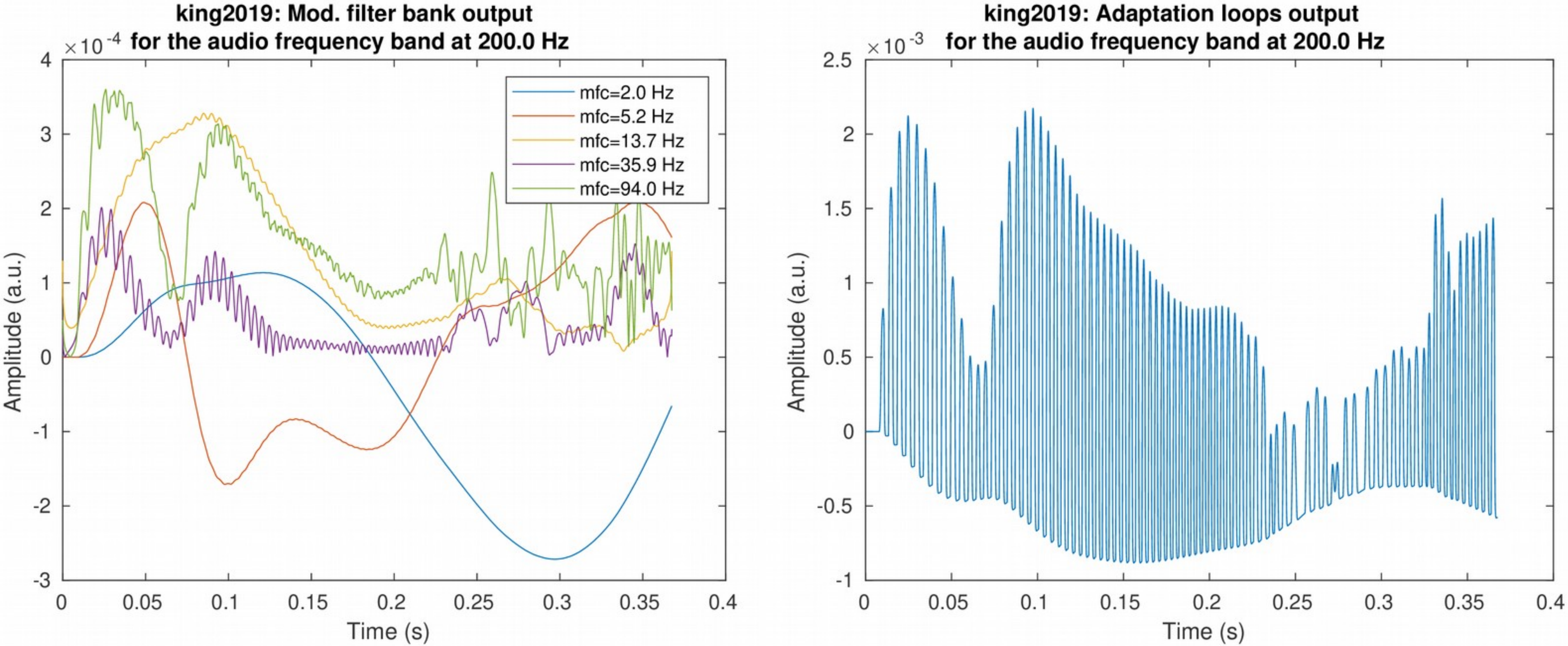


```
function exp_workshop2022_Paris(varargin)

This script processes two waveforms using the model specified by the
  corresponding flag. The structure of this script is similar to those
  adopted in AMT to reproduce data of the literature (e.g., exp_lopezpoveda2001,
  exp_osses2022, exp_takanen2013, etc.).

To run this script you can run:
  exp_workshop2022_Paris('dau1997');
  exp_workshop2022_Paris('zilany2014');
  exp_workshop2022_Paris('verhulst2015');
  exp_workshop2022_Paris('verhulst2018');
  exp_workshop2022_Paris('bruce2018');
  exp_workshop2022_Paris('relanoiborra2019');
  exp_workshop2022_Paris('king2019');
  exp_workshop2022_Paris('osses2021');
```

- In the script I provide many “extra details” about the models and I generate two figures (only for the waveform “greasy”)

Solution to the activities

- The outputs for the “ENS model”, king2019:


king2019: Mod. filter bank output
for the audio frequency band at 200.0 Hz
×10^-4
Amplitude (a.u.)
Time (s)
mfc=2.0 Hz
mfc=5.2 Hz
mfc=13.7 Hz
mfc=35.9 Hz
mfc=94.0 Hz
king2019: Adaptation loops output
for the audio frequency band at 200.0 Hz
×10^-3
Amplitude (a.u.)
Time (s)