Machine Hearing:
A Research Agenda and an Approach

Richard F. Lyon

Google, Inc.

July, 2009

Banff MM, Math, and ML Workshop

http://www.dicklyon.com

Machine Hearing Research Agenda

Why an agenda?

- Help “Machine Hearing” become a first-class academic and commercial field like “Machine Vision”
- Motivate, plan, and promote my project activities
- Do something useful with all the uninterpretable audio media out there
- Motivate the recording of more…

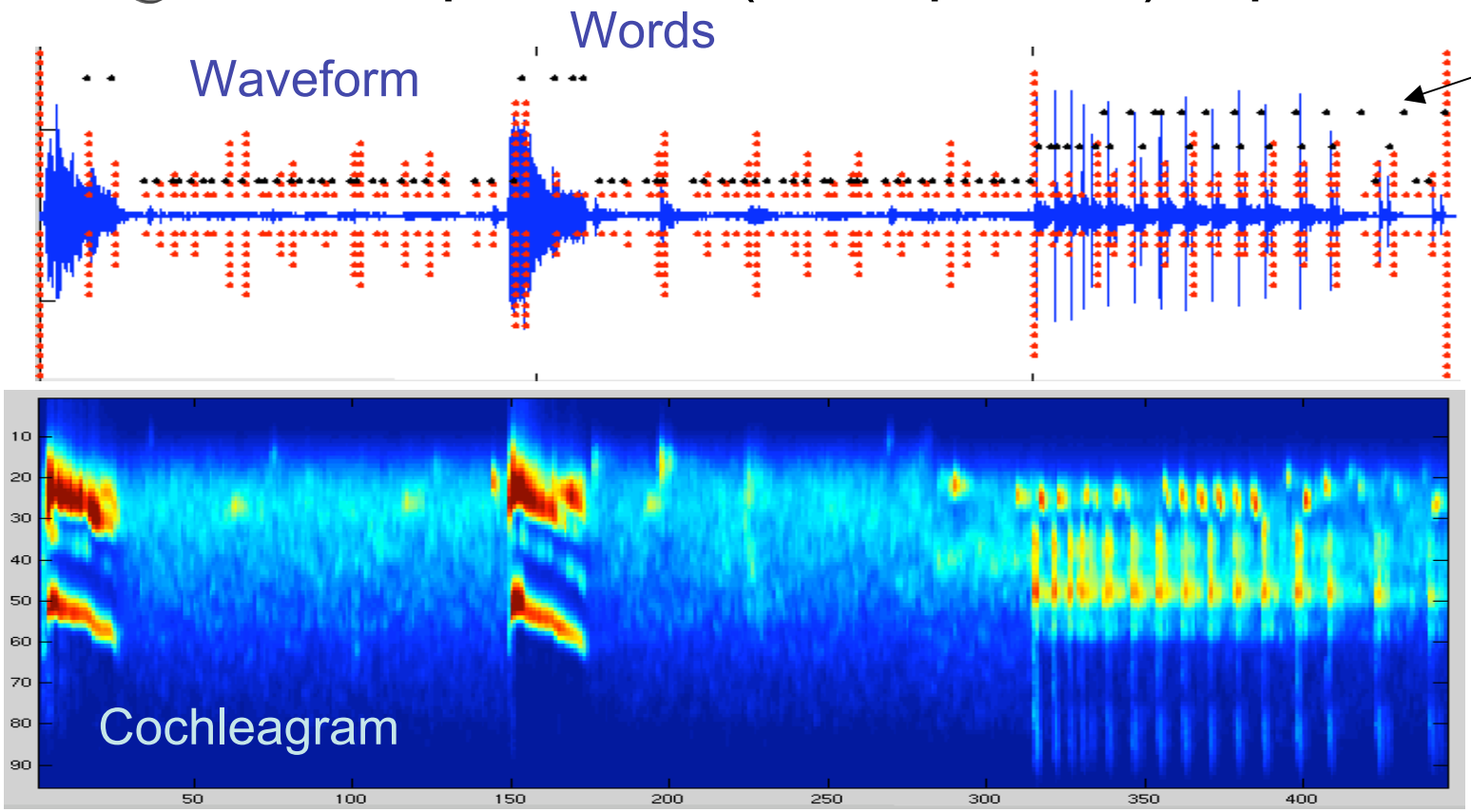
Approach – four areas to get right:

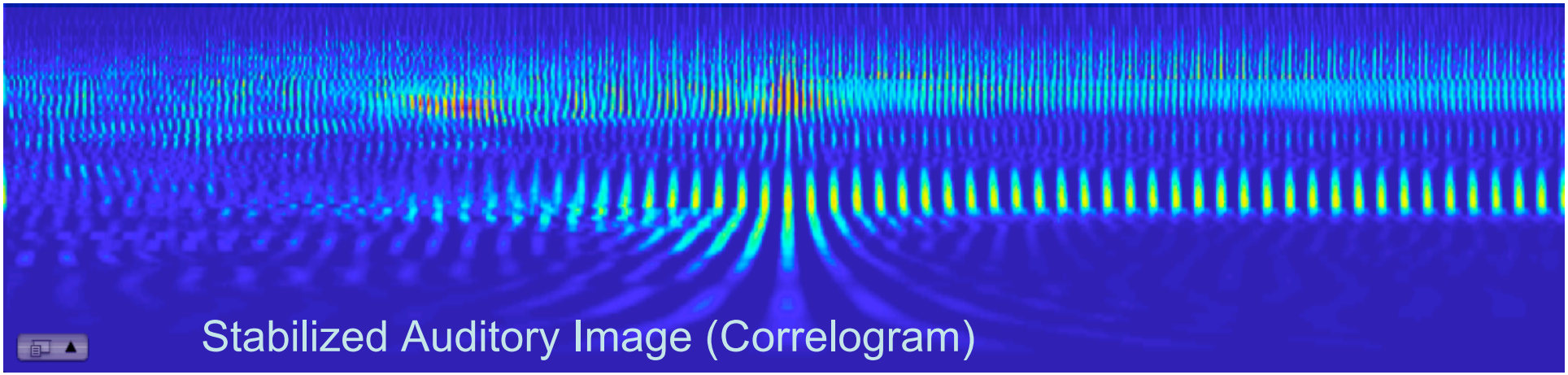
1. Leveraging techniques already developed in the machine-vision and machine-learning fields (Li Deng's **"cross-column breeding"**)
2. Productive interaction with the wider field of hearing research, to keep models honest and motivate better experiments
3. **Focus on applications** for which the challenge has to do with what things sound like, as opposed to specialized domain knowledge ("non-speech non-music audio")
4. Share and promote common base tools and models, tasks, datasets, etc.

How we proceed…

- Good auditory front end, based on good hearing research, leads to representation of what things “sound like”
- A good hard application: Content-based retrieval of sound tracks (or other audio content) based on what it “sounds like is going on”
- Analogy to content-based image retrieval, based on features that encode what things “look like,” leads to workable system structure
- Noisy data is good data – robustness from the outset

“Sapsucker” (woodpecker) representations





Searching for Sounds...

Imagine this sort of system... (not a real product)

- Ranking audio documents based on text queries
  - For example: the query 'lion' should return lion sounds *based on the audio file content*

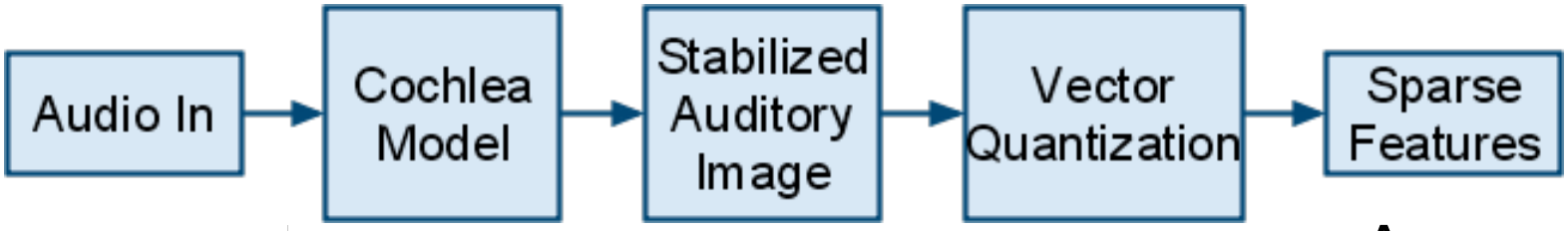
Ranking Task

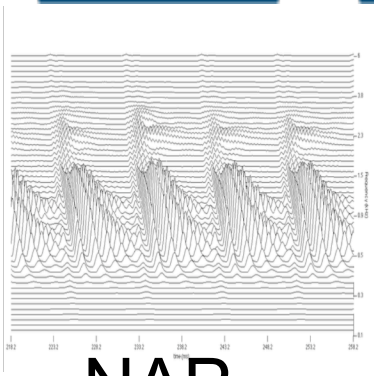
Learn to match acoustic features with text tags

Use a small labeled data set for training

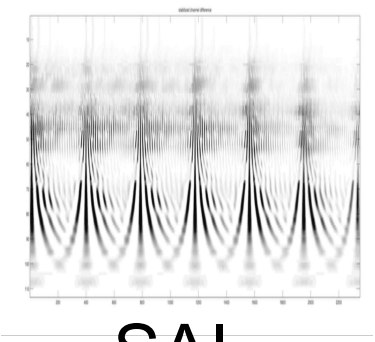
File	Tags
lion.wav	lion, roar, mammal, bigcat
4e8fcd312.mp3	car, tire, squeal, chase, engine
home_video.avi	baby, laugh

Auditory Models





NAP



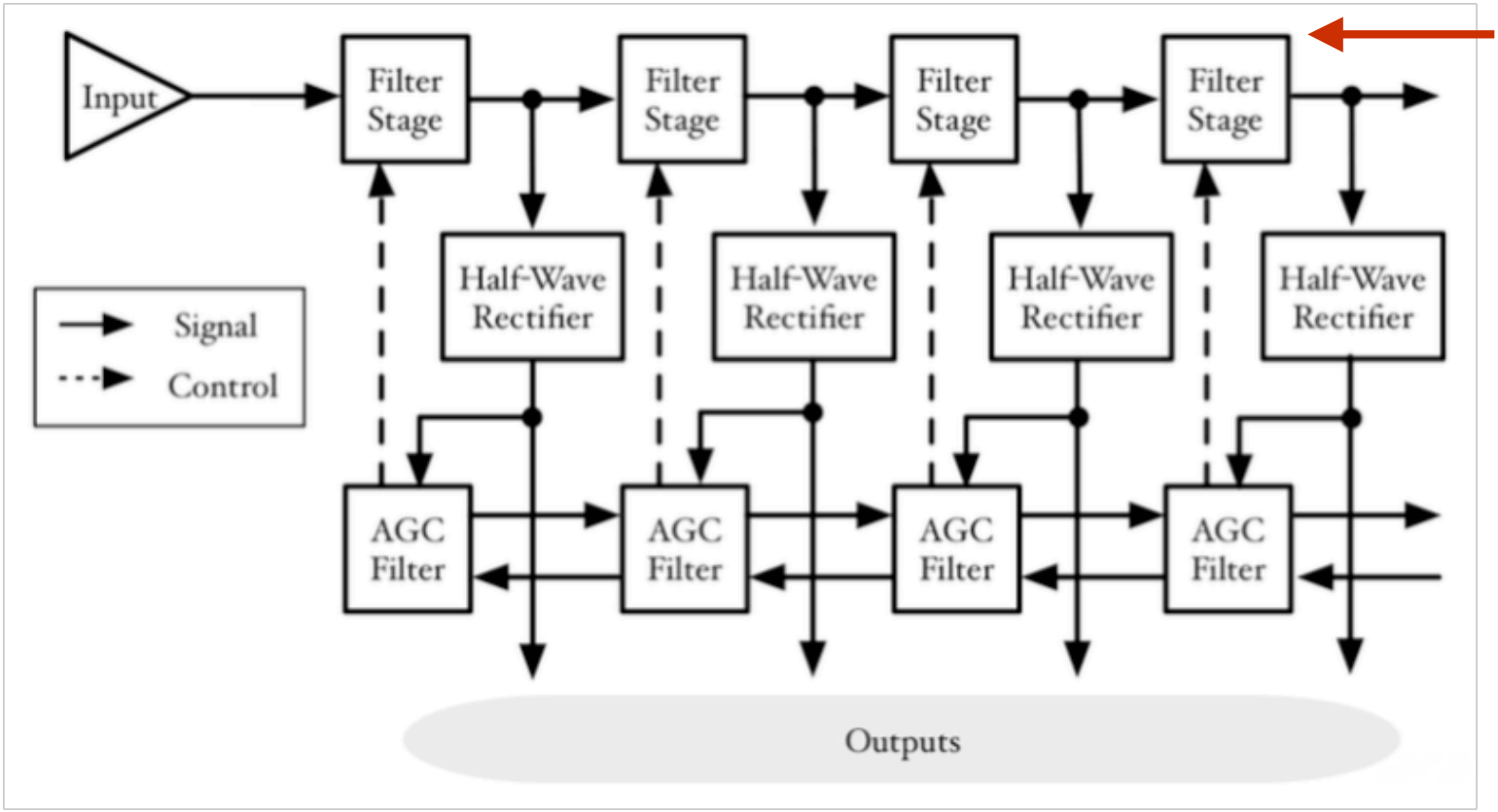
SAI

Accumulate "bag of auditory words"

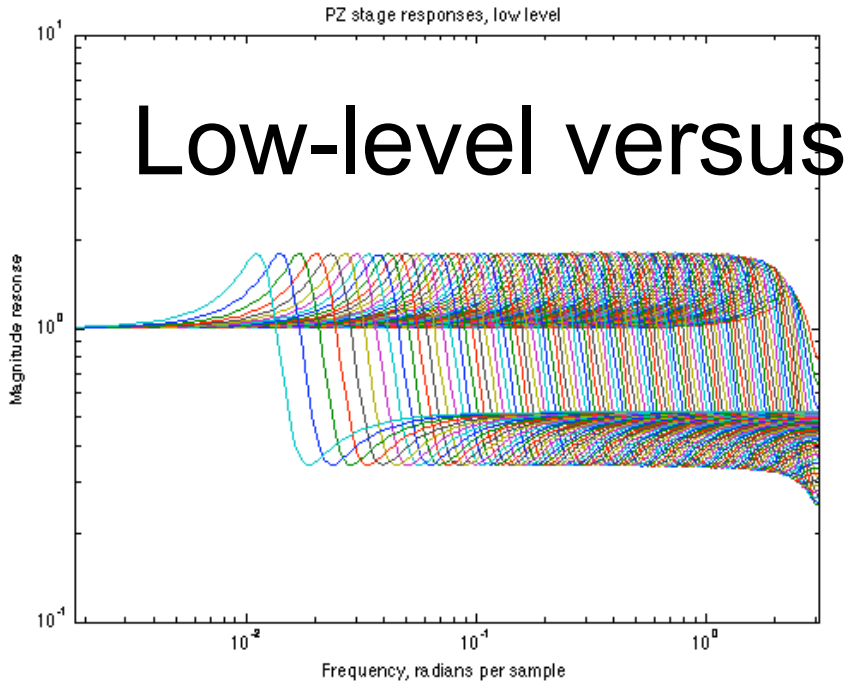
We also compared with vector-quantized MFCCs

Cochlea Model

'Pole-Zero Filter Cascade': Cascade model of the cochlea

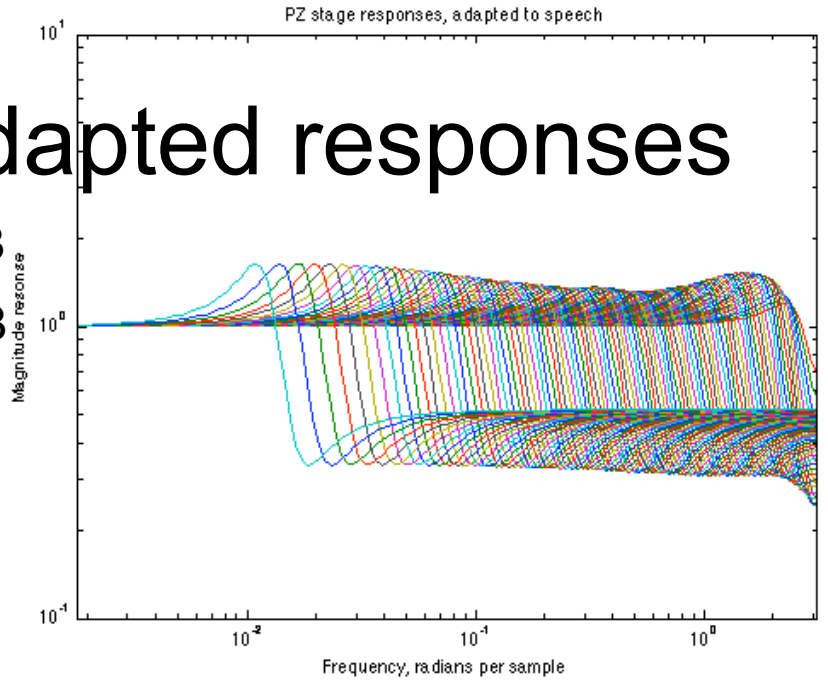


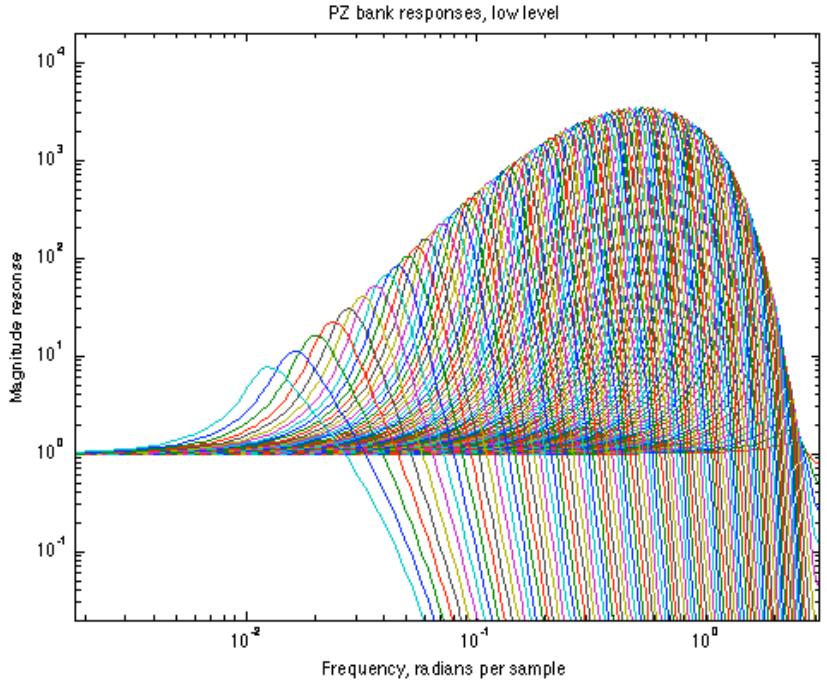
Low-level versus adapted responses



6 dB

0 dB





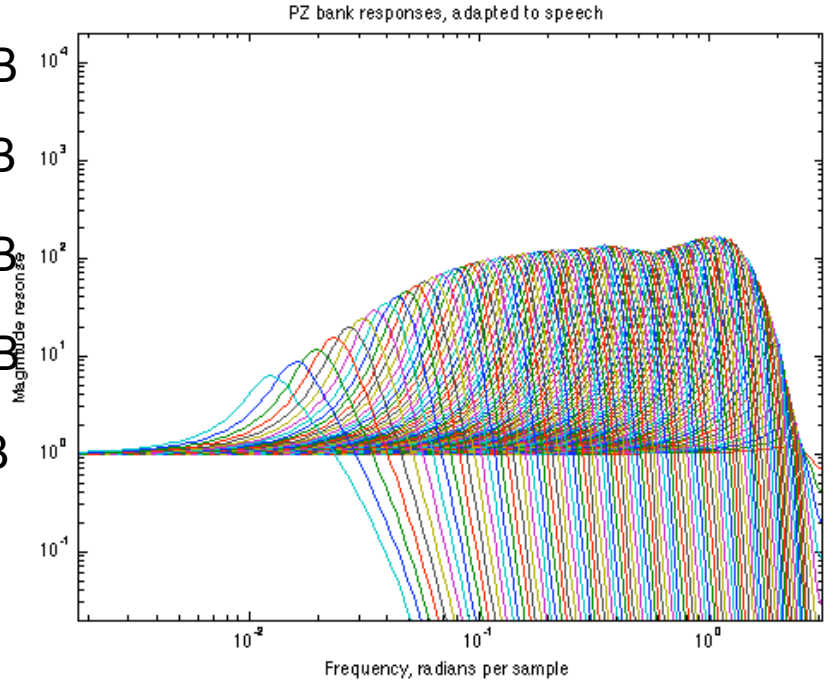
80 dB

60 dB

40 dB

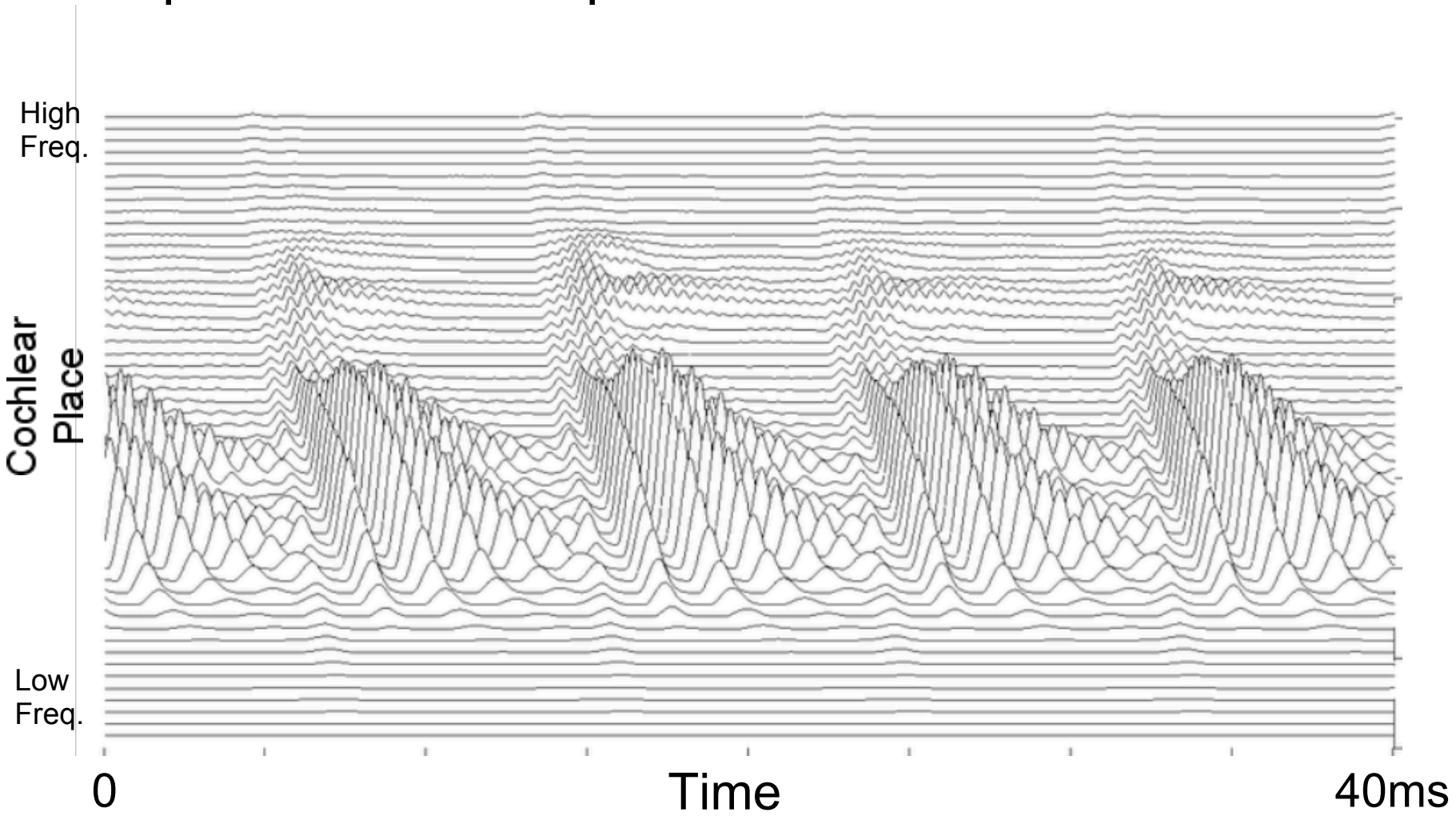
20 dB

0 dB



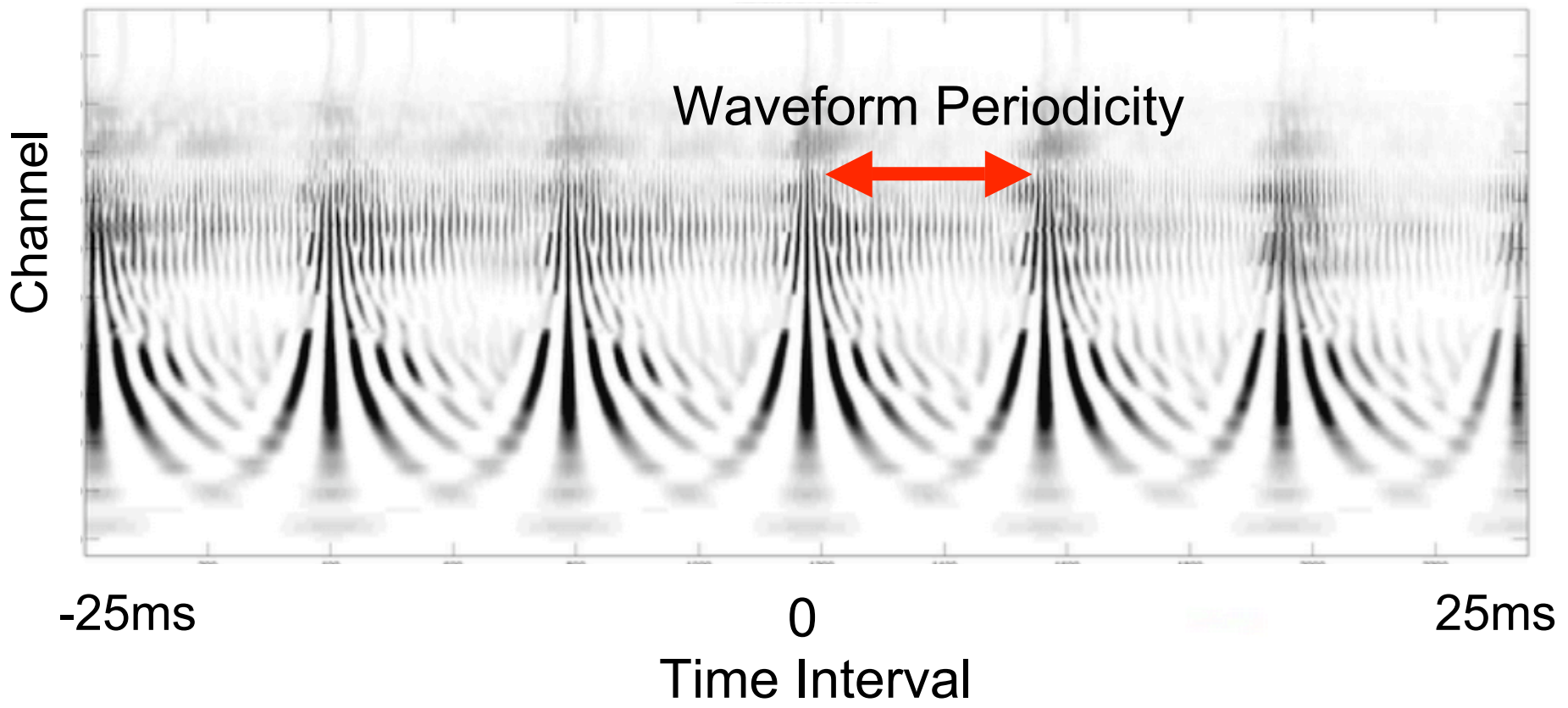
Neural Activity Pattern (NAP)

Example filterbank output for 40ms of a human vowel /a/

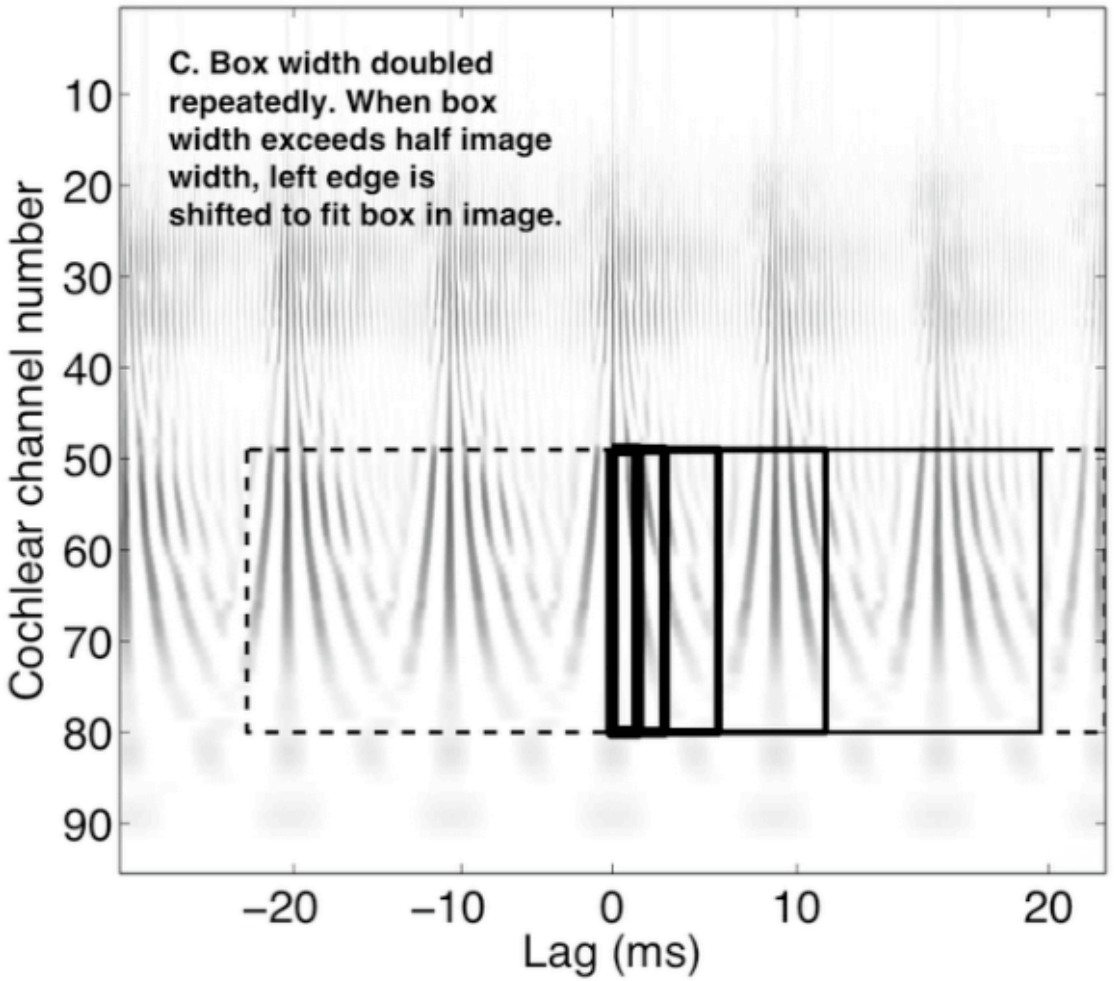


Strobed Temporal Integration

Stabilized Auditory Image (SAI)



Dense Feature Extraction

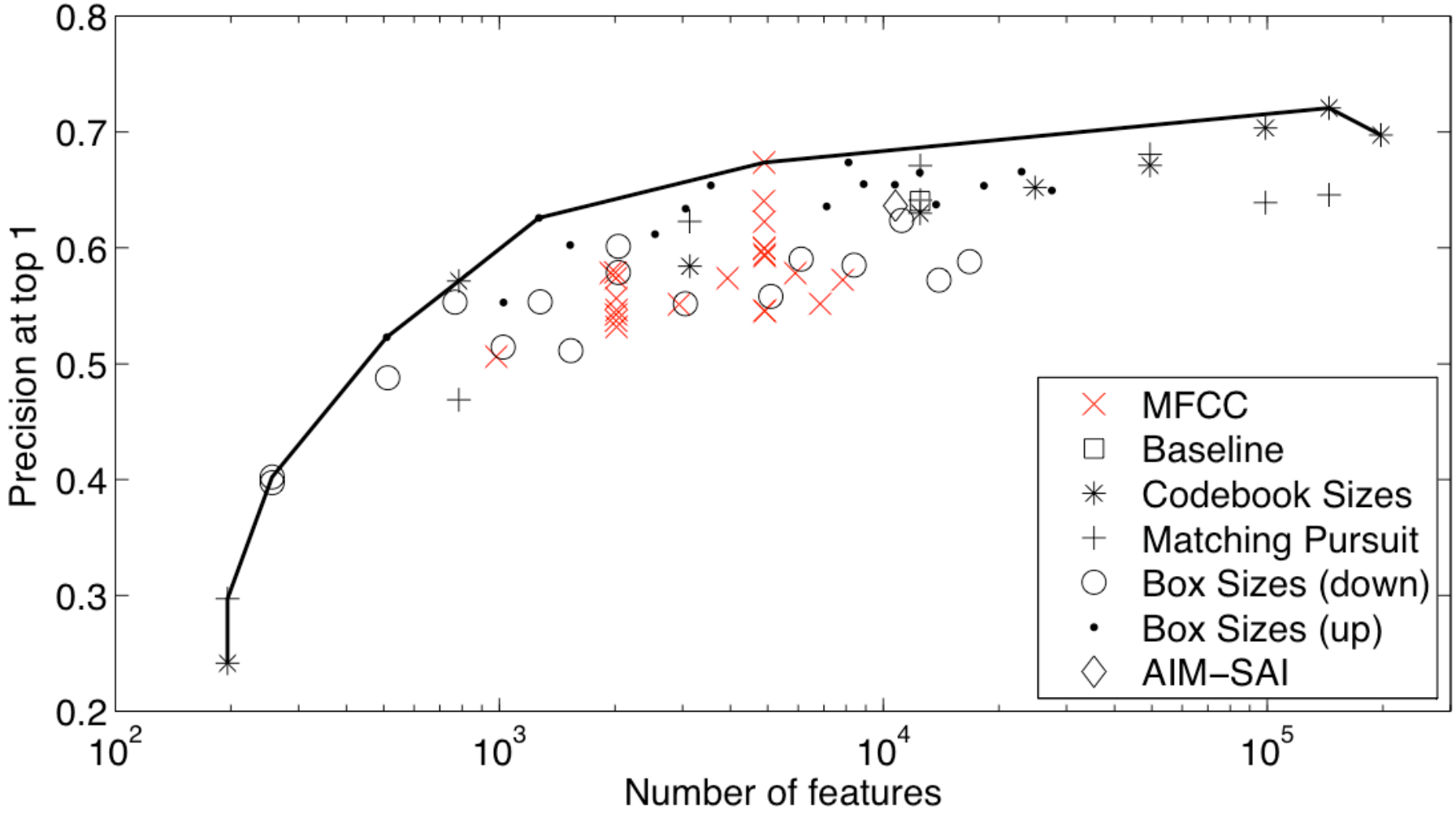


Map “bag of audio words” to query terms “bag of words”

- Vector quantize dense features to get sparse features; bag them
- Use Samy Bengio’s “PAMIR”:

  Passive–Aggressive Model for Image Retrieval
- Efficient and effective use of thousands to millions of sparse feature dimensions
- Simple linear model with robust efficient training; sparse updates
- Scales to many millions of “documents” for ranking and retrieval in response to queries

Results: Precision vs. Feature Count



Other ways to leverage machine vision

- In combination:
  - Audio/visual robots and 3D perception
  - Content-base retrieval, content classification, etc., based on joint sound/image features
- By analogy:
  - Object tracking –> sound source tracking
  - Key-point features –> key sound features?

Other good applications
(mostly in Multi-Media)

- Indexing, retrieval, summarization of personal audio diaries, movie soundtracks, etc.
- Real-time and retrospective analysis of audio security/surveillance recordings
- Front end for speech transcription systems
- Music retrieval, recommendation, etc.
- “Sound perception” and “understanding” of all sorts; let our computers help us deal with audio media…

Bake-offs

- As in speech and vision, good shared datasets with defined tasks can lead to good competitive/cooperative progress
- Quantitative performance testing requires lots of labeled training and testing data. How will we get it?

Auditory Filters:

recent work connecting cochlear mechanics, psychophysics, physiology, and neuromorphic implementations

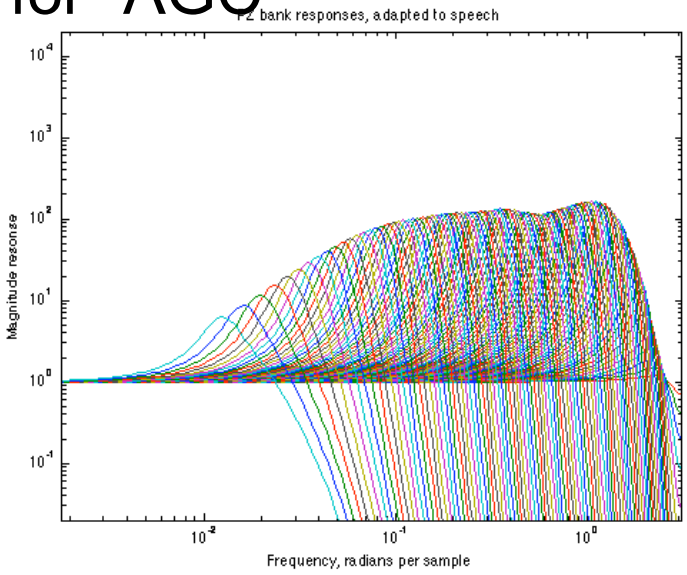
Richard F. Lyon

Google, Inc.

(from 2008 Auditory Filters workshop at Cambridge)

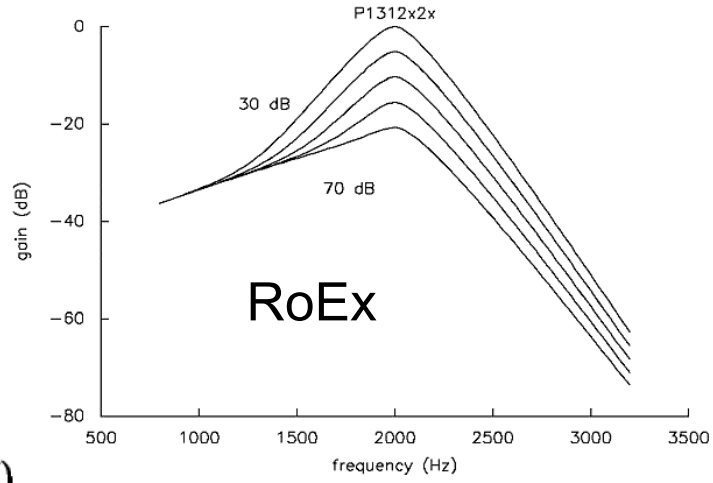
Pole–Zero Filter Cascade (PZFC) – preview

- Good fit to human masking data with simple parameters
- As with APFC, connection to traveling wave allows natural coupling effects, for masking, adaptation, etc.
- Like APGF & OZGF, unity-gain tail models lossless propagation of low-frequency energy; tail doesn't wag with Q or other parameters
- Easy to implement directly as standard second-order (pole–zero) filter sections, without further approximation
- Easy to vary parameters dynamically for "AGC"
- Low total order (complexity) for multi-channel filterbank, by sharing filter sections – total complexity just 2nd-order per channel, compared to 8th-order for gammatones



“Auditory filter” shapes and models

Roy Patterson



Auditory Filter Shapes and their Pole Orders (N)

Resonance
N=1
Schafer et al. 50
Patterson 70

Gaussian
N=∞
Swets et al. 62
Patterson 76

RoEx
Patterson and Nimmo-Smith 80

RoEx(p), RoEx(p,r), RoEx(p,t,w)
Patterson et al. 82

APGF, OZGF
N=8, 16, 32
Lyon 96

Symmetric approx. GTF
N=2
Patterson 74

GTF
N=4
Patterson 88

GCF
N=6
Irino 95

GTF, OZGF, ++
N=3
Flanagan 60

Two-side RoEx
Moore and Glasberg 87

Two-sided GTF
Rosen and Baker 94

Except for Flanagan 60, only psychophysical models are included

It would be good to connect to physiology…

Jim Flanagan 1960–62 filter models of basilar membrane

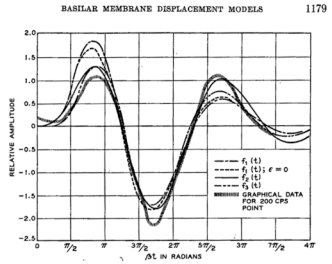


Fig. 12 — Impulse responses of the models. These displacement functions are the inverse transforms of the frequency-domain data in Figs. 10 and 11. Time delay has been equalized to compare waveforms. Locations of absolute origins are given in the text.

order-3 “gamma-tone” filter

a reasonable fit to BÉKÉSY’s results is

$$F_l(s) = c_1 \beta_l^4 \left(\frac{2000\pi\beta_l}{\beta_l + 2000\pi}\right)^{0.8} \left(\frac{s+\varepsilon_l}{s+\beta_l}\right) \left[\frac{1}{(s+\alpha_l)^2+\beta_l^2}\right]^2 e^{\frac{-3\pi s}{4\beta_l}}$$

The membrane response at any point is therefore approximated in terms of the poles and zeros of the rational function part of $F_l(s)$. As

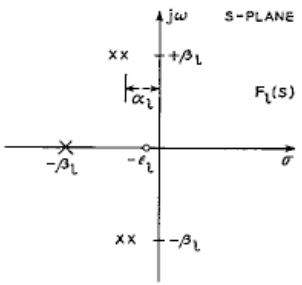


Fig. 4.17a. Pole-zero diagram for the approximating function $F_l(s)$ (after FLANAGAN, 1962a)

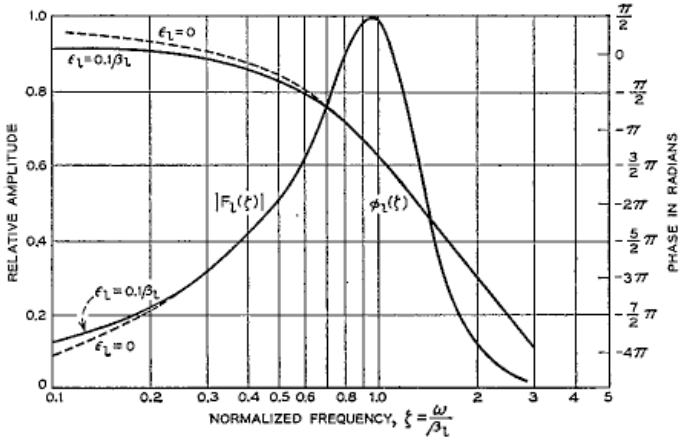


Fig. 4.17b. Amplitude and phase response of the basilar membrane model $F_l(s)$. Frequency is normalized in terms of the characteristic frequency β_l

good approximation to the displacement impulse response of the basilar membrane, at a point maximally responsive to radian frequency β, is

$$\left.\begin{aligned} p(t) &= (\beta t)^2 e^{-\beta t/2} \sin \beta t \\ &= h_{bm}(t) \sin \beta t. \end{aligned}\right\} \quad (5.9)$$

The time window for the basilar membrane, according to this modeling[1], is the “surge” function plotted in Fig. 5.6. One notices that the time window has a duration inversely related to β. It has its maximum at $t_{max} = 4/\beta$. If, as a crude estimate, $2t_{max}$ is taken as the effective duration D of the window, then for several membrane places:

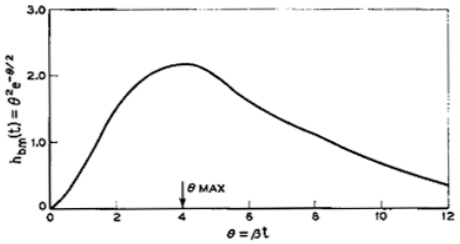


Fig. 5.6. The effective time window for short-time frequency analysis by the basilar membrane in the human ear. The weighting function is deduced from the ear model discussed in Chapter IV

$\beta/2\pi$ (cps)	$D = 2t_{max}$ (msec)
100	12.0
1000	1.2
5000	0.2

For most speech signals, therefore, the mechanical analysis of the ear apparently provides better temporal resolution than spectral resolution. Generally, the only harmonic component resolved mechanically is the fundamental frequency of voiced segments. This result is borne out by observations on the models described in Chapter IV.

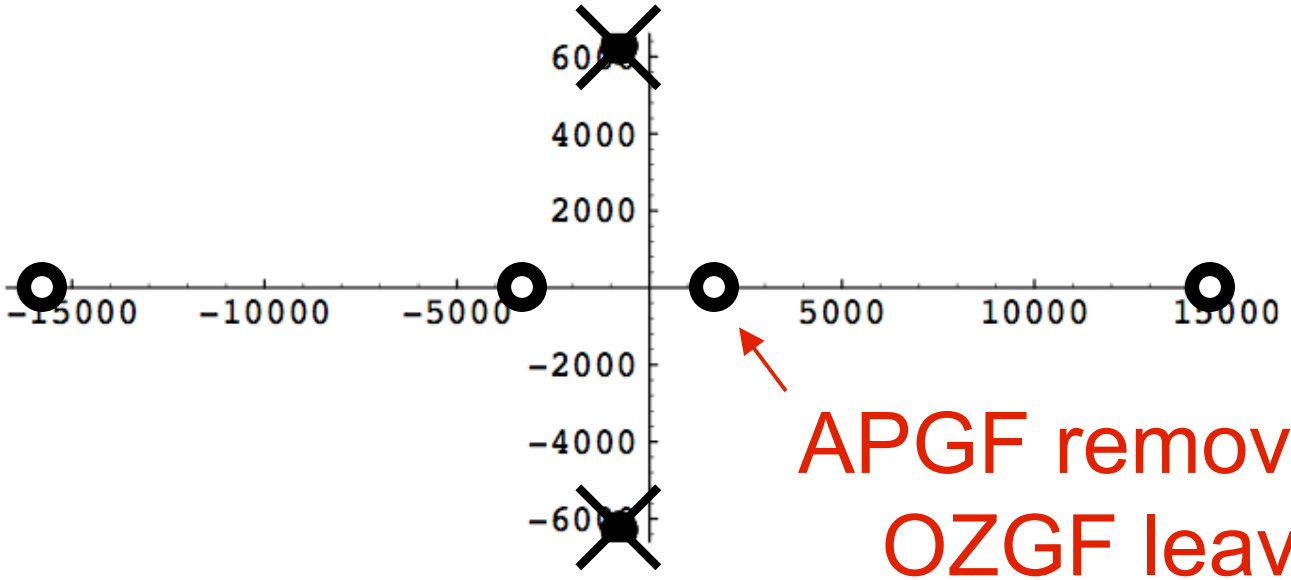
We can now plot the location of these poles and zeros assuming a gammatone filter with a bandwidth of 125Hz and a center frequency of 1000Hz (this approximates a 1000Hz cochlear channel.) In the graph below, the four zeros are shown along the real (horizontal) axis. There are four sets of poles at each of the dots indicated near ±6000j (which indicates a resonance near 1000Hz.)

```
PlotPolesWithDots[complexpolelist_]:=
    ListPlot[Map[{Re[#],Im[#]}&,complexpolelist],
        Prolog->{AbsolutePointSize[10]},
        AspectRatio->Automatic,
        DisplayFunction->Identity];

Show[{PlotPolesWithDots[N[Map[Part[#,1,2]&,poles]/.
            B->2 Pi 125/.w->2 Pi f/.f->1000]],
    PlotPolesWithDots[N[Map[Part[#,1,2]&,zeros]/.
            B->2 Pi 125/.w->2 Pi f/.f->1000]]},
        DisplayFunction->$DisplayFunction];
```

Gamma-tone's zeros

Slaney 1993



APGF removes these; OZGF leaves one

suitable for calculations in a digital computer. One such digital simulation represents the membrane motion at 40 points (FLANAGAN, 1962b).

As might be done in realizing the analog electrical circuit, the digital representation of the model can be constructed from sampled-data equivalents of the individual complex pole-pairs and the individual real poles and zeros. The sampled-data equivalents approximate the continuous functions over the frequency range of interest. The computer operations used to simulate the necessary poles and zeros are shown in Fig. 4.25. All of the square boxes labelled D are delays equal to the time between successive digital samples. The input sampling frequency, $1/D$, in the present

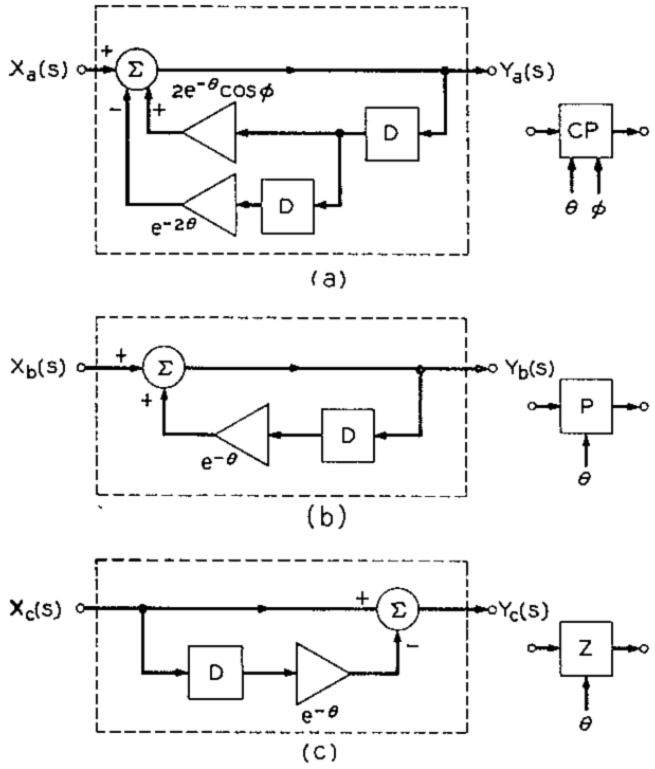


Fig. 4.25. Sampled-data equivalents for the conjugate complex poles, real-axis pole, and real-axis zero

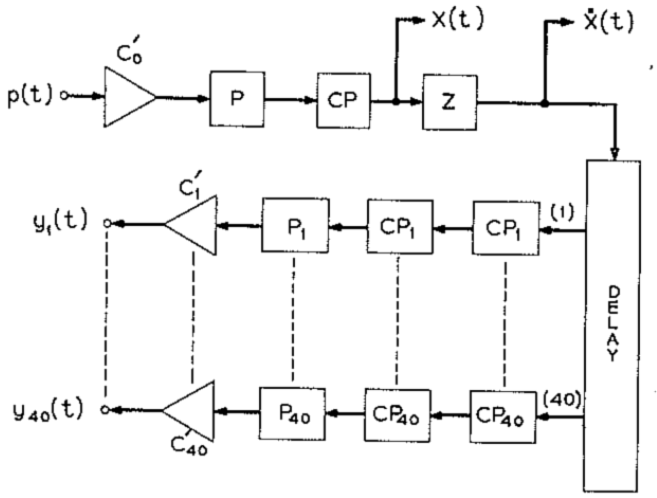
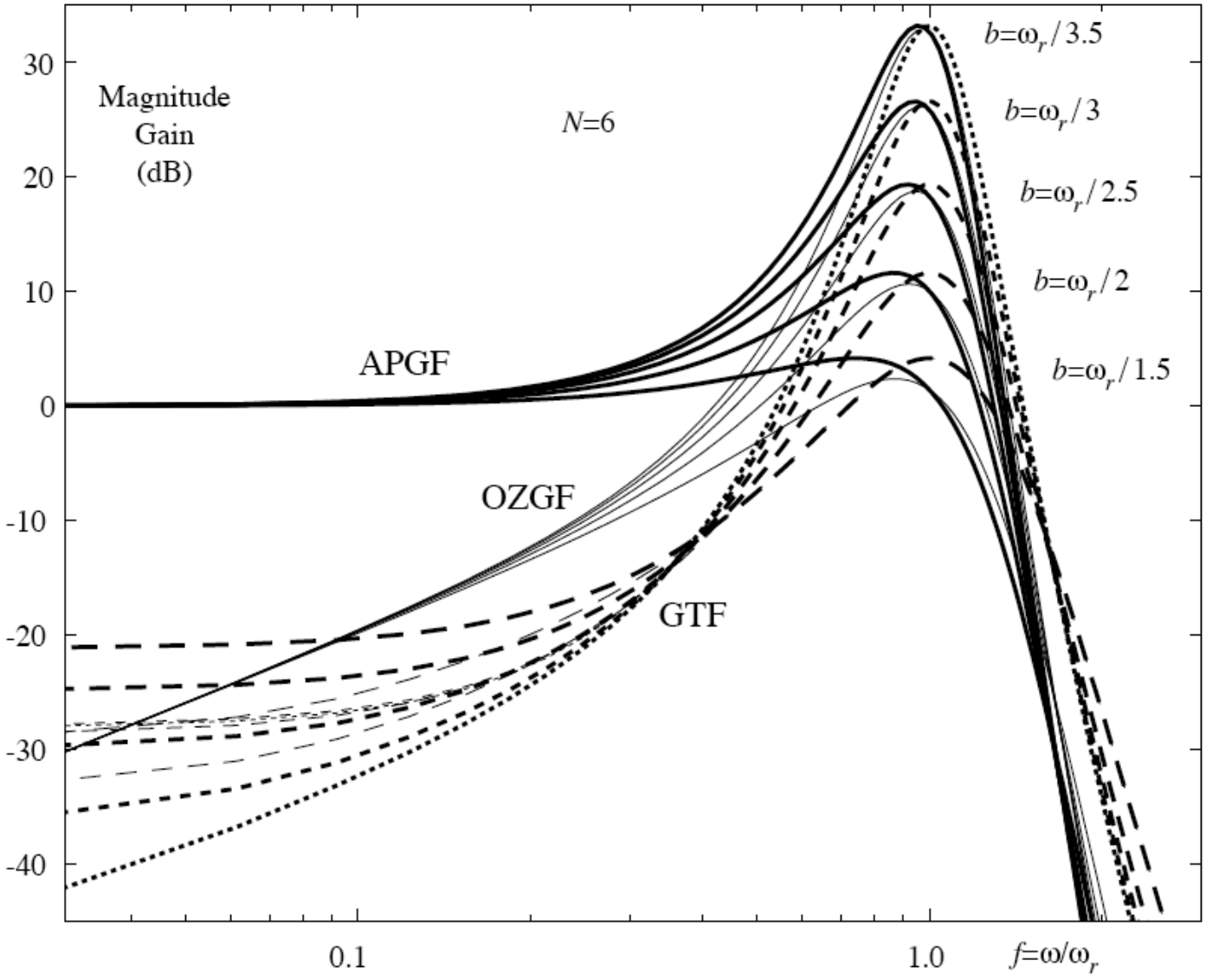


Fig. 4.26. Functional block diagram for a digital computer simulation of basilar membrane displacement

Such filters have easy implementations: Flanagan 1962

Don’t wag the tail when changing pole damping –
Like APGF & OZGF, unlike Gammatone & Gammachirp



Waves in uniform media: sinusoidal functions of *x* and *t*

In a uniform medium, a wave propagating toward increasing values of the place dimension is given by

- Complex wavenumber $k(\omega)$ controls loss or gain

$$W(x) = A\exp(i\omega t - ikx)$$

By examining the ratio of waves at two places separated by a distance Δx, we see that the wave at the farther place is equal to the wave at the nearer place multiplied by $\exp(-ik\Delta x)$, representing a frequency-dependent *filter* characterizing the stretch of length Δx.

- But in a non-uniform medium, the wavenumber *k* depends on place (*x*) as well as on frequency: $k(\omega, x)$
- Net result is a **cascade of filters** $\exp(-ik(\omega, x)dx)$.

CMOS VLSI Cochlea: an all-pole filter cascade (APFC) – Lyon & Mead 1988

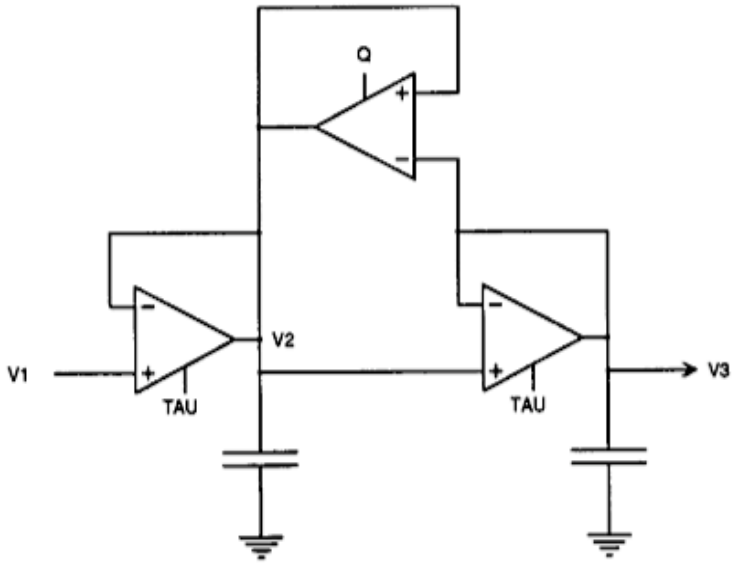


Figure 1: Second-order filter-section circuit.

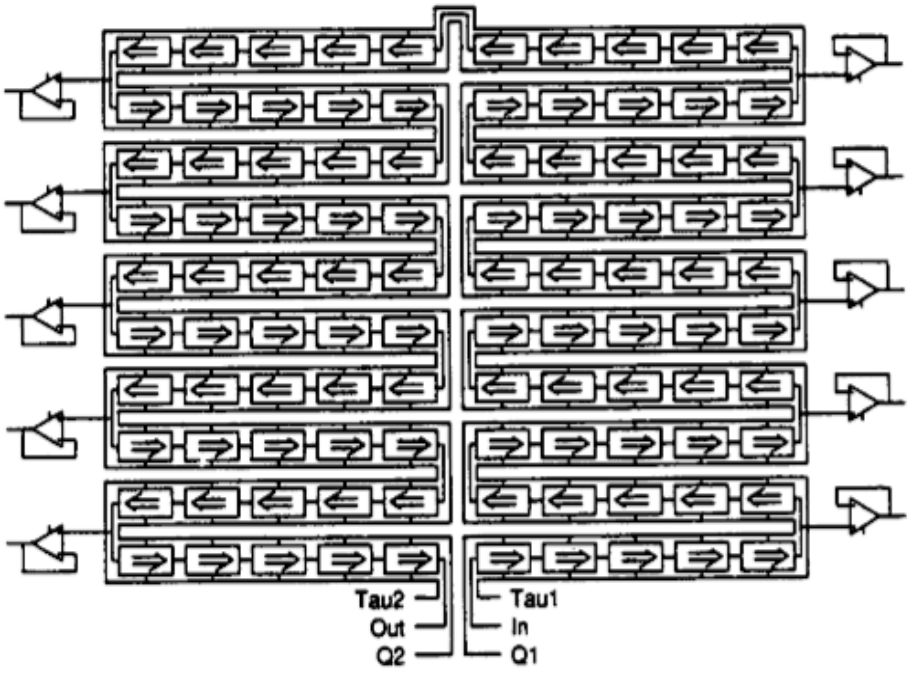
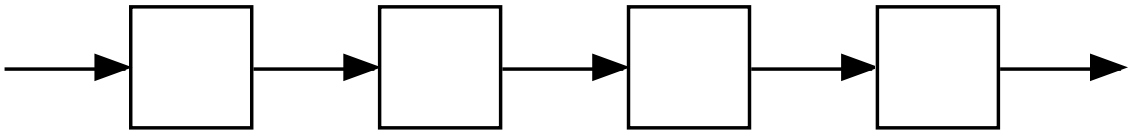


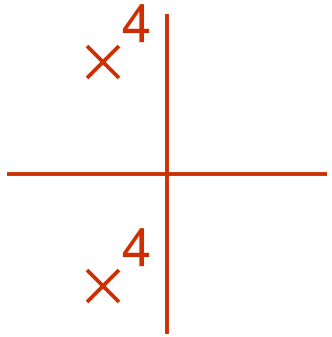
Figure 2: Floorplan of 100-stage cochlea chip.

2-pole stage is an order-1 GTF or APGF

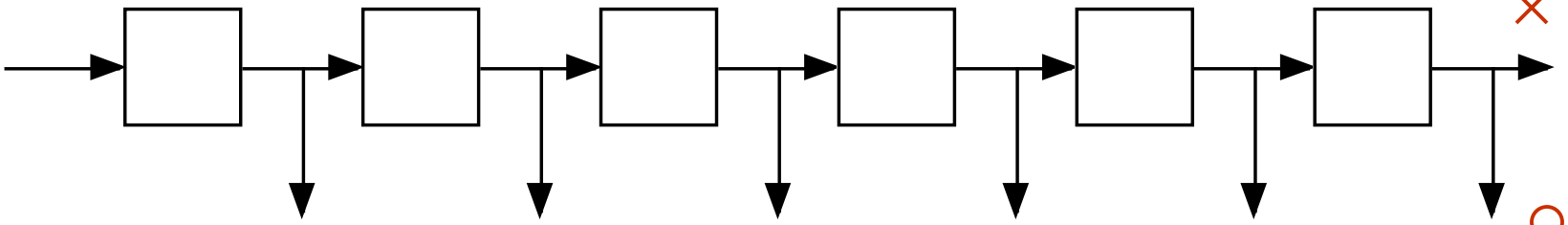
Filter cascades

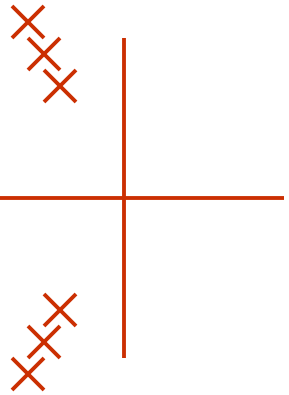
A 4th-order All-Pole Gammatone Filter is an 8th-order filter, a cascade of 4 identical pole-pair filters:



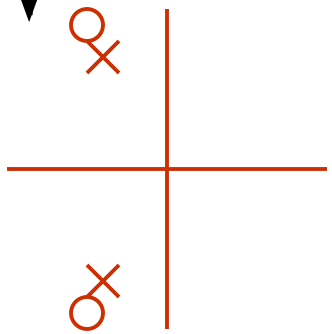


An All-Pole Filter Cascade looks the same, but non-identical stages, and outputs at every step:





A Pole-Zero Filter Cascade looks the same, but each stage has both a pair of poles and a pair of zeros.



Some possible cascade stage frequency responses

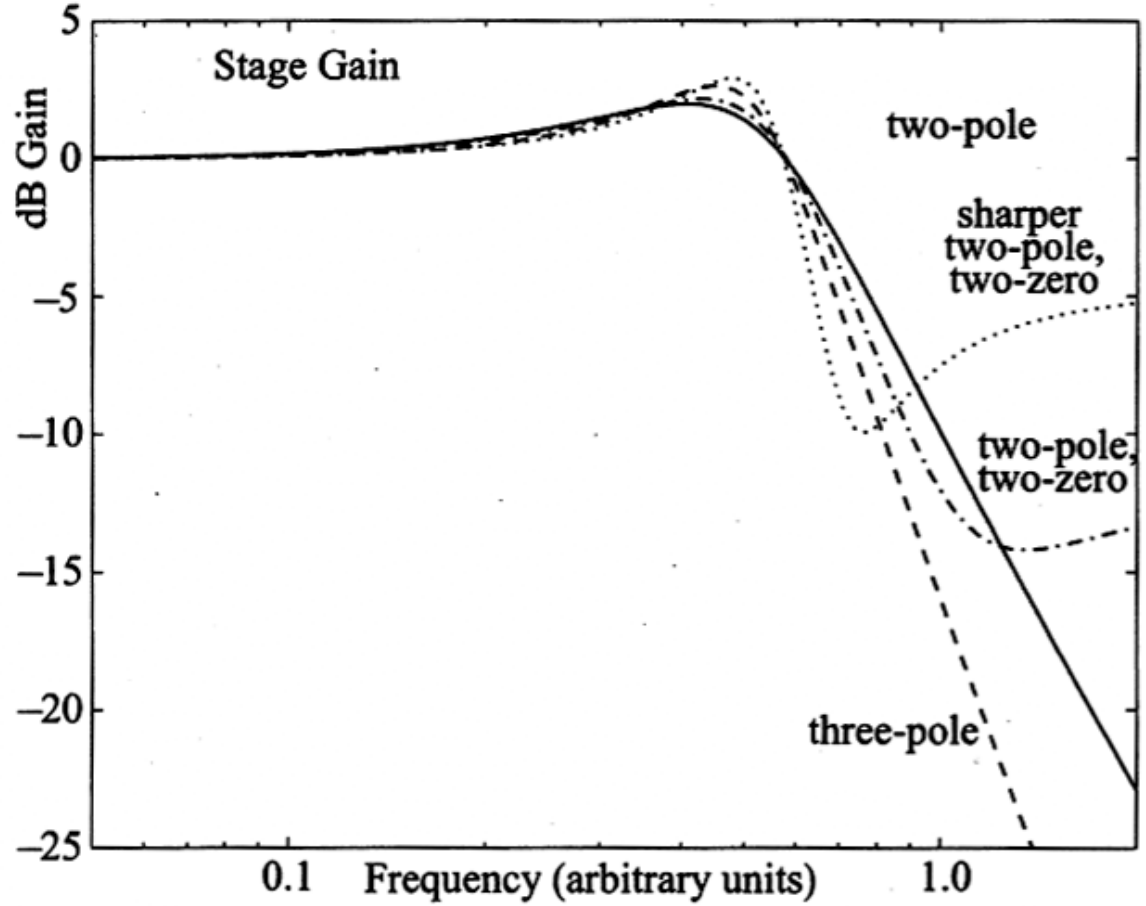
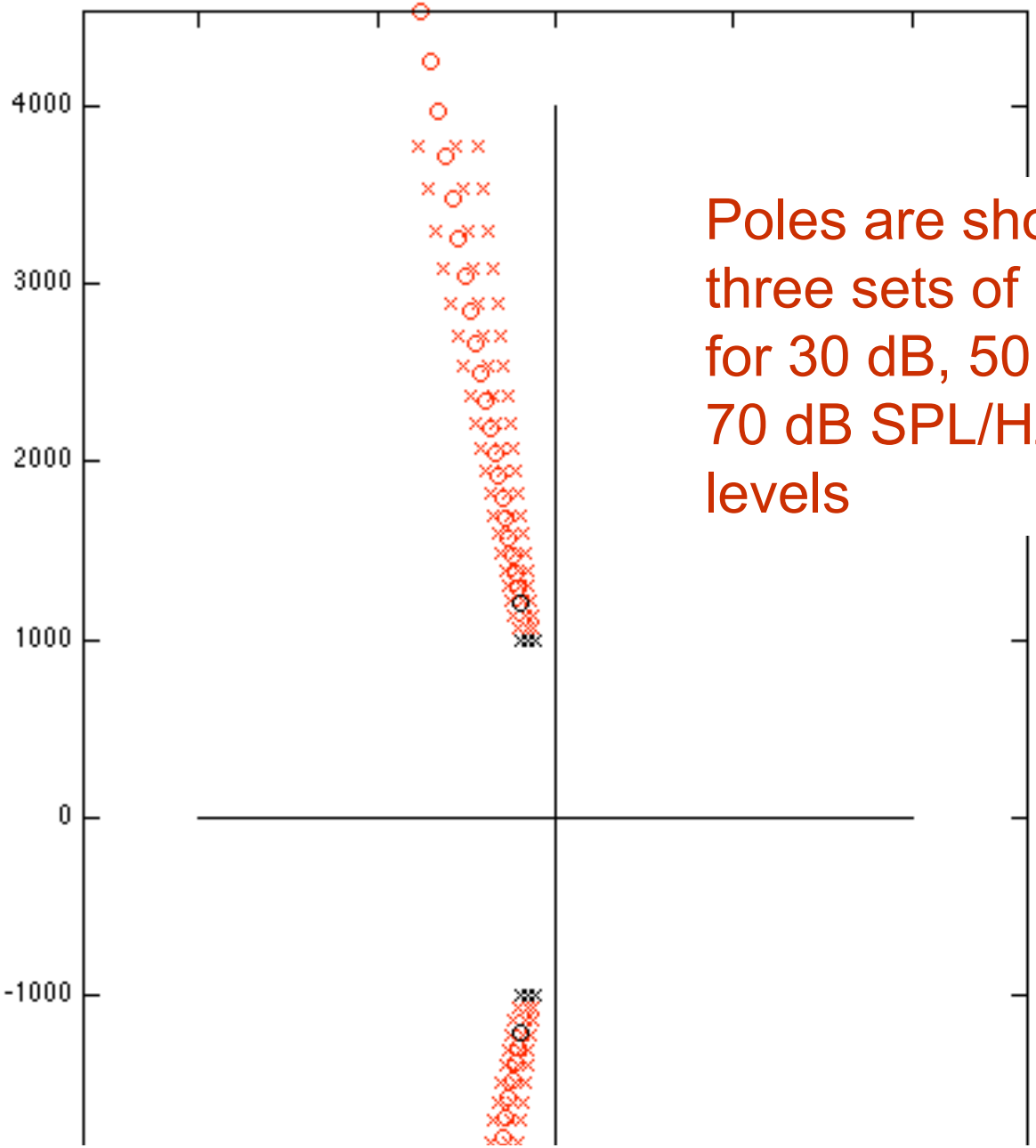


Figure 1.3 Stage transfer-functions gains. For each of the four filter designs of Figure 1.2, the magnitude of the stage transfer function is plotted.

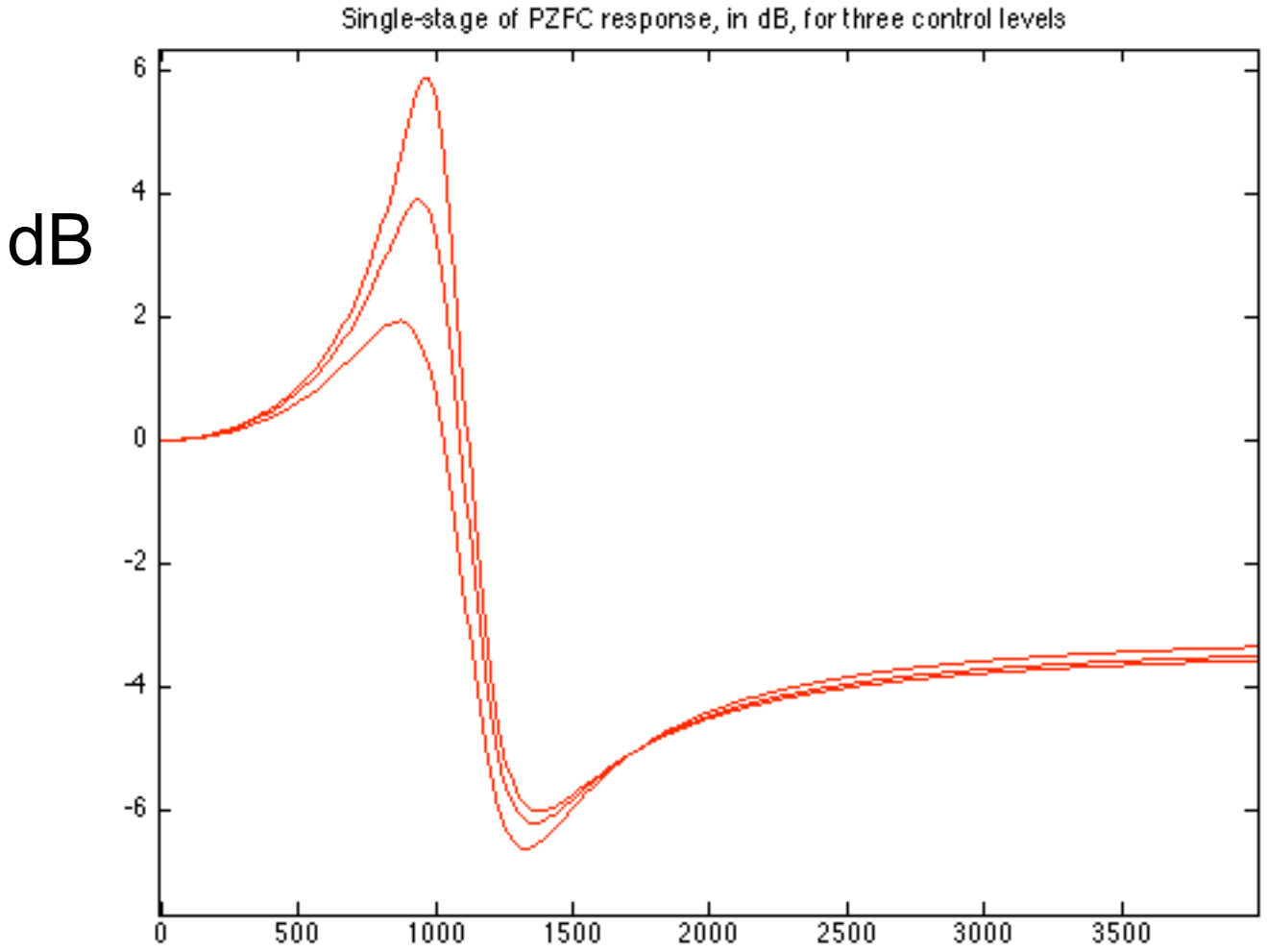
PZFC poles and zeros in the s plane



Poles are shown at three sets of locations, for 30 dB, 50 dB, and 70 dB SPL/Hz noise levels

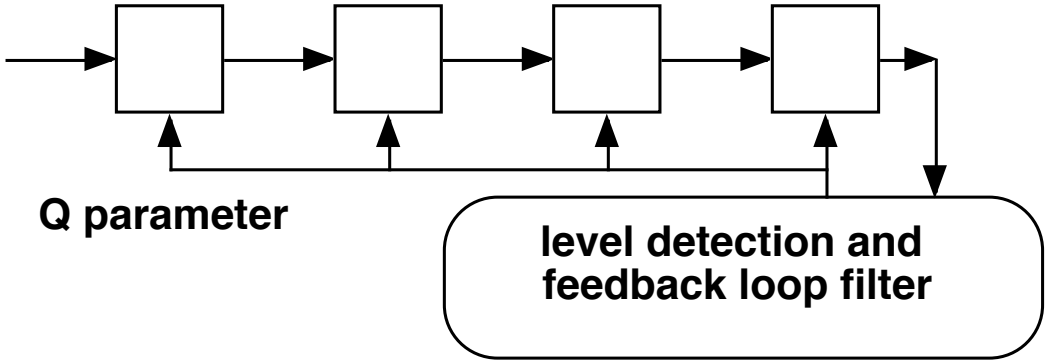
PZFC stage frequency response:

pole pair makes a bump (variable via pole Q),and **zero** pair makes a dip

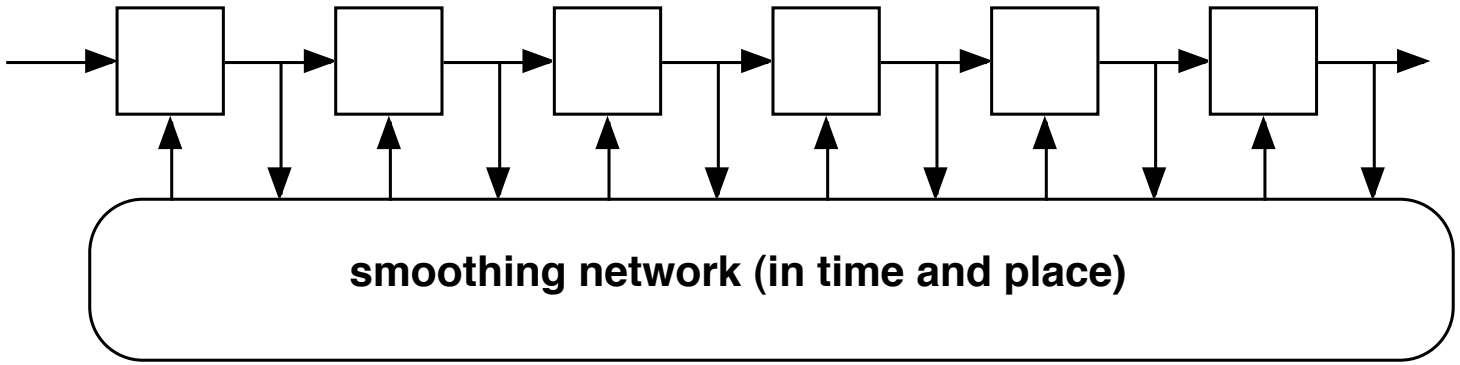


AGC (AQC) via feedback
(automatic gain or Q control)

APGF (or OZGF) in feedback configuration

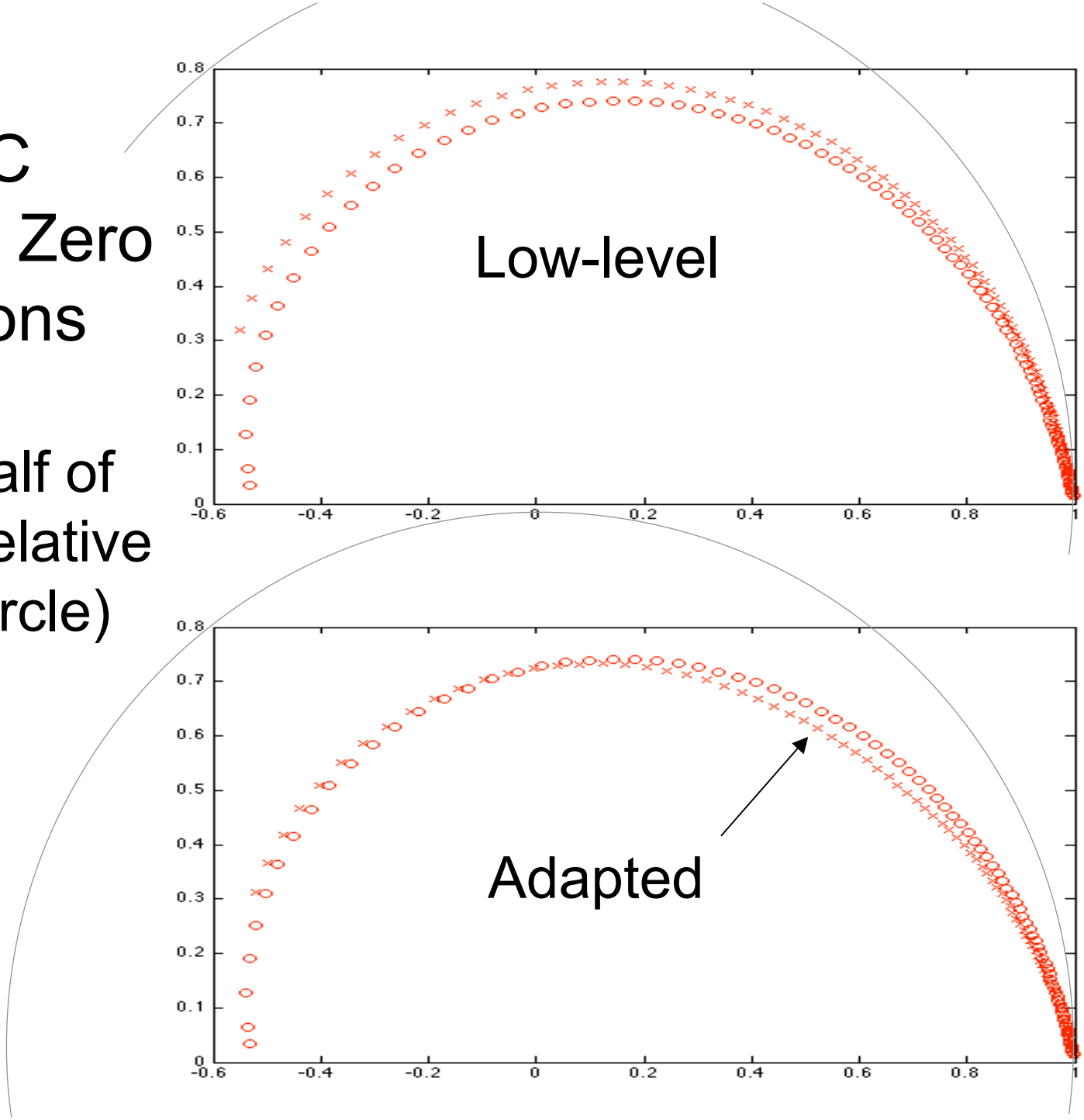


A Filter Cascade in feedback configuration uses all outputs to affect parameters of all stages, through a sort of diffusion spreading effect

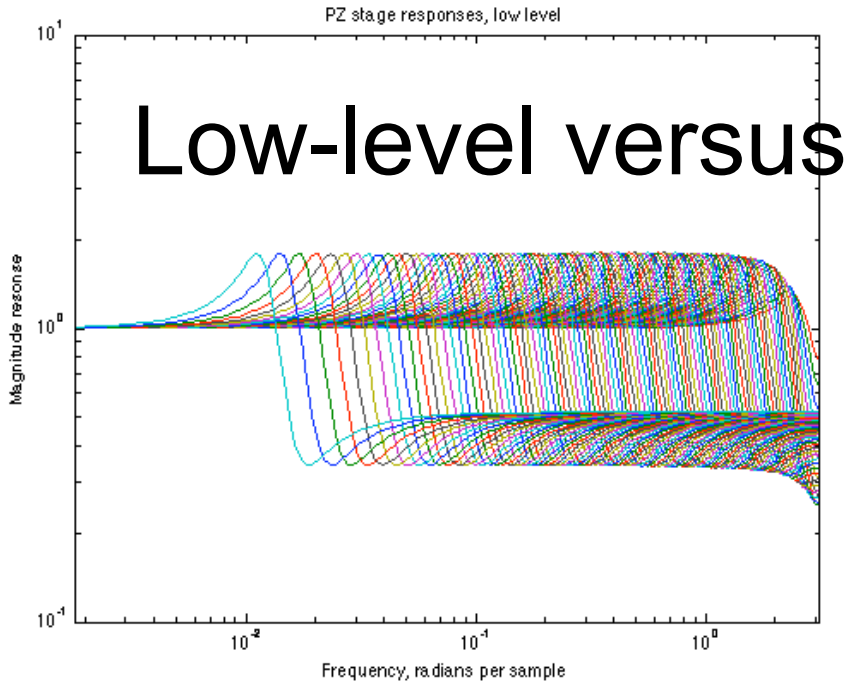


PZFC Pole and Zero Locations

(in top half of Z plane, relative to unit circle)

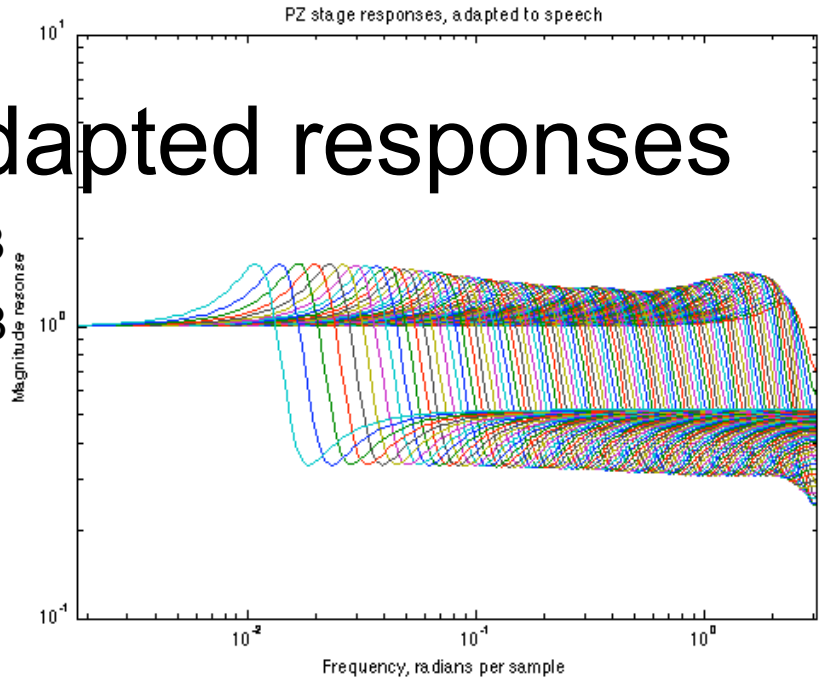


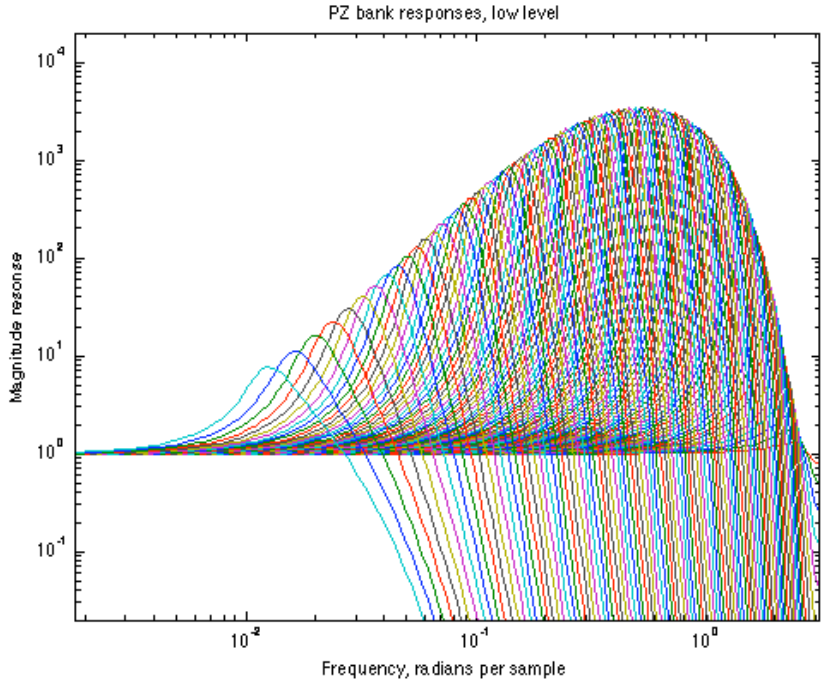
Low-level versus adapted responses



6 dB

0 dB





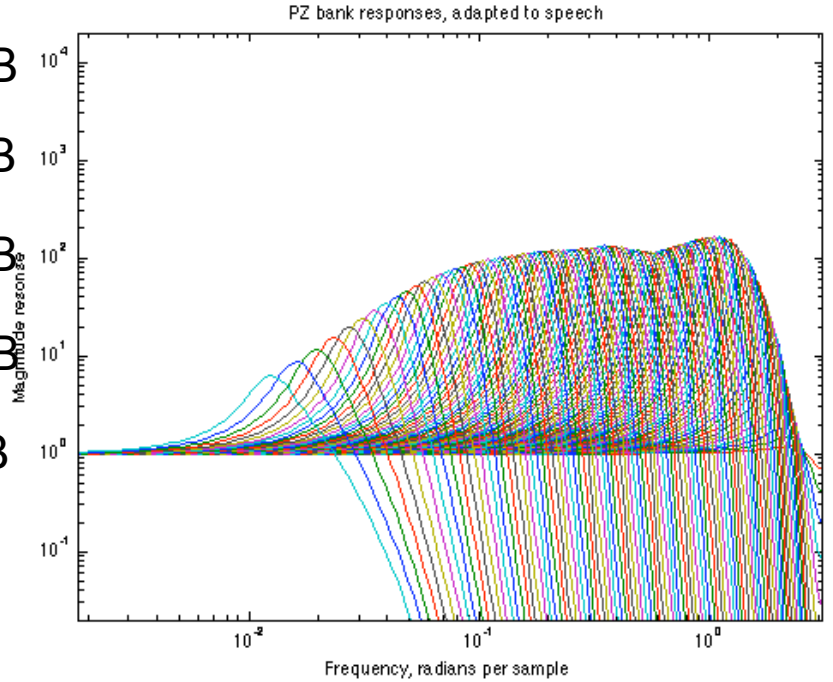
80 dB

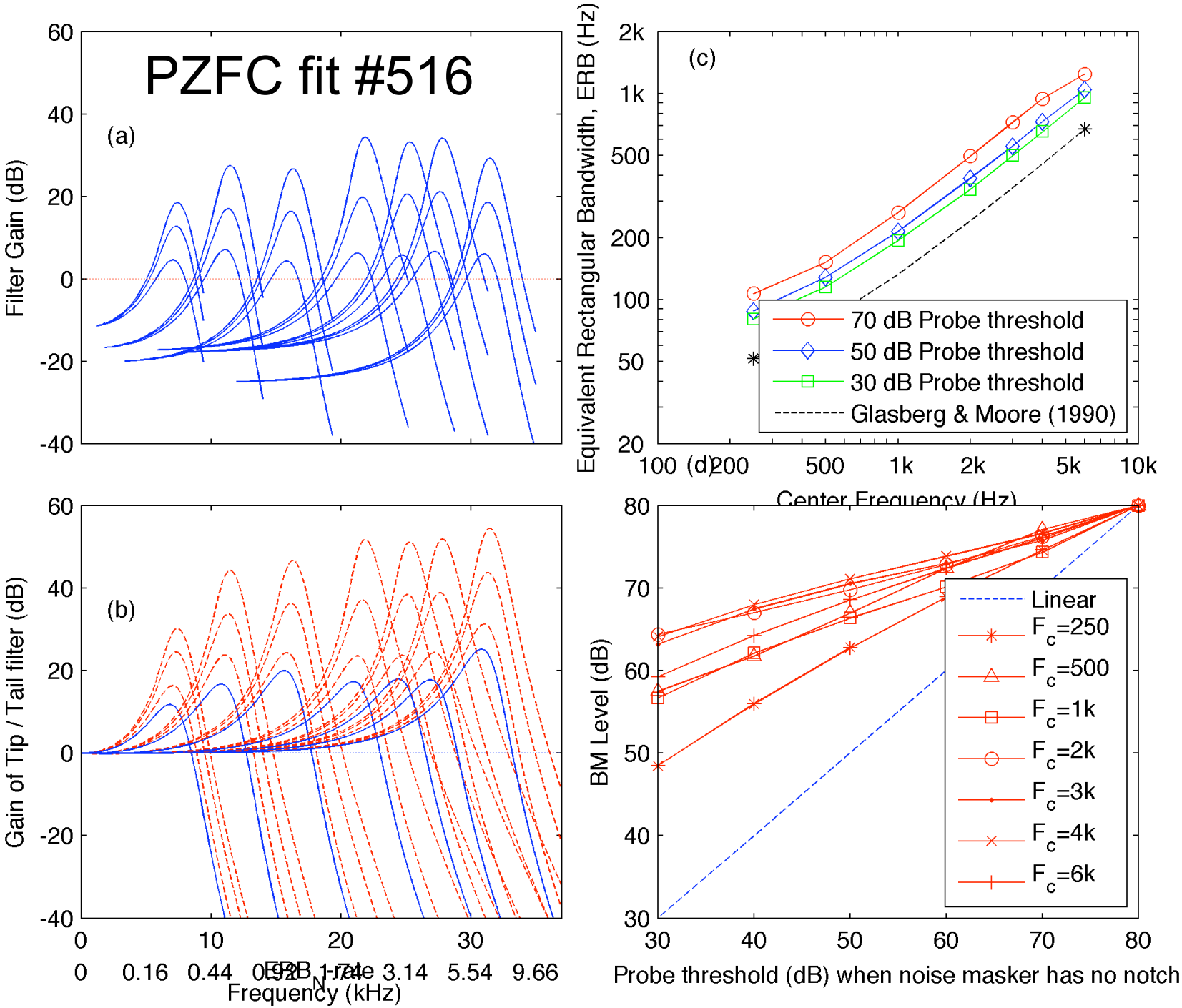
60 dB

40 dB

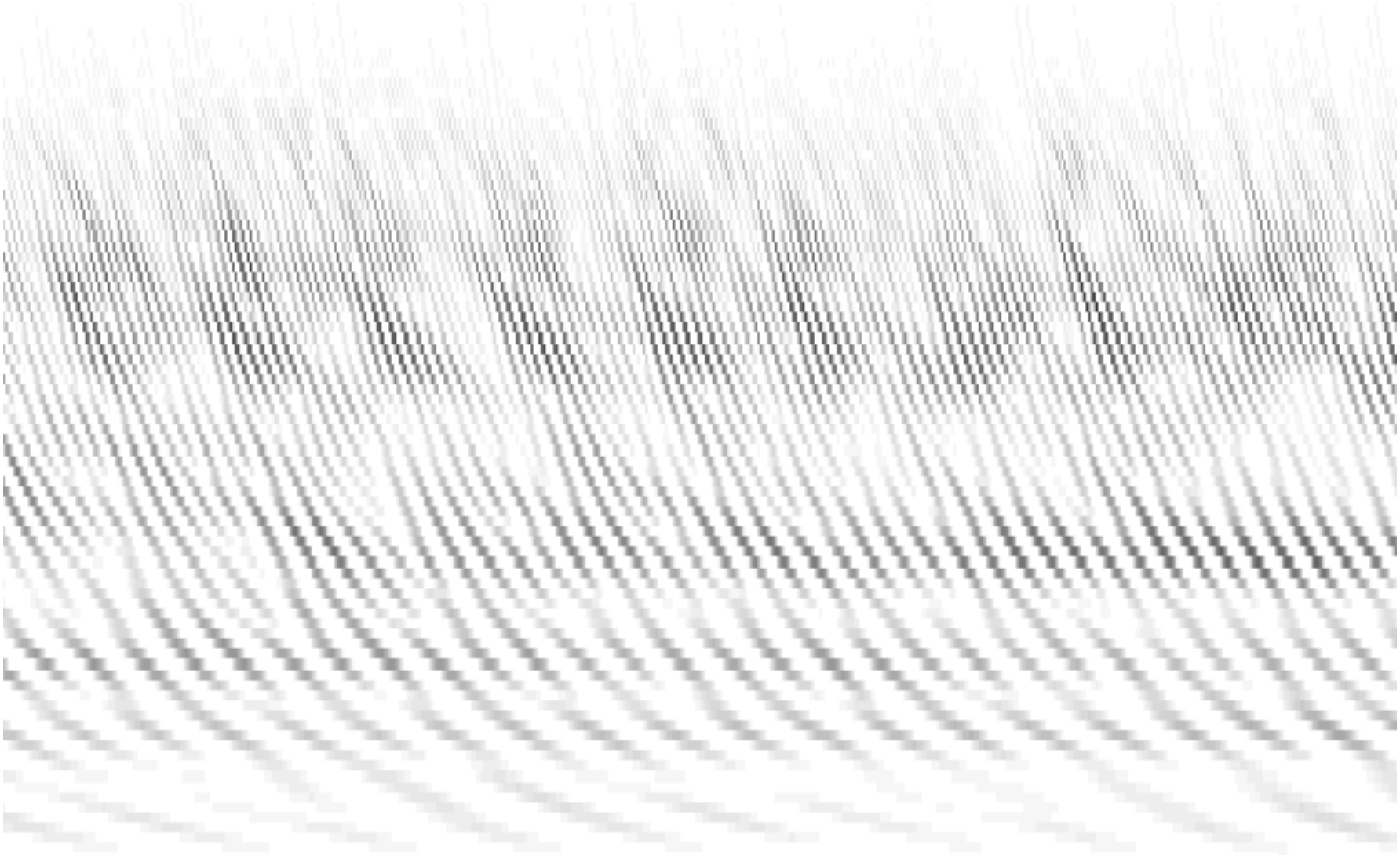
20 dB

0 dB

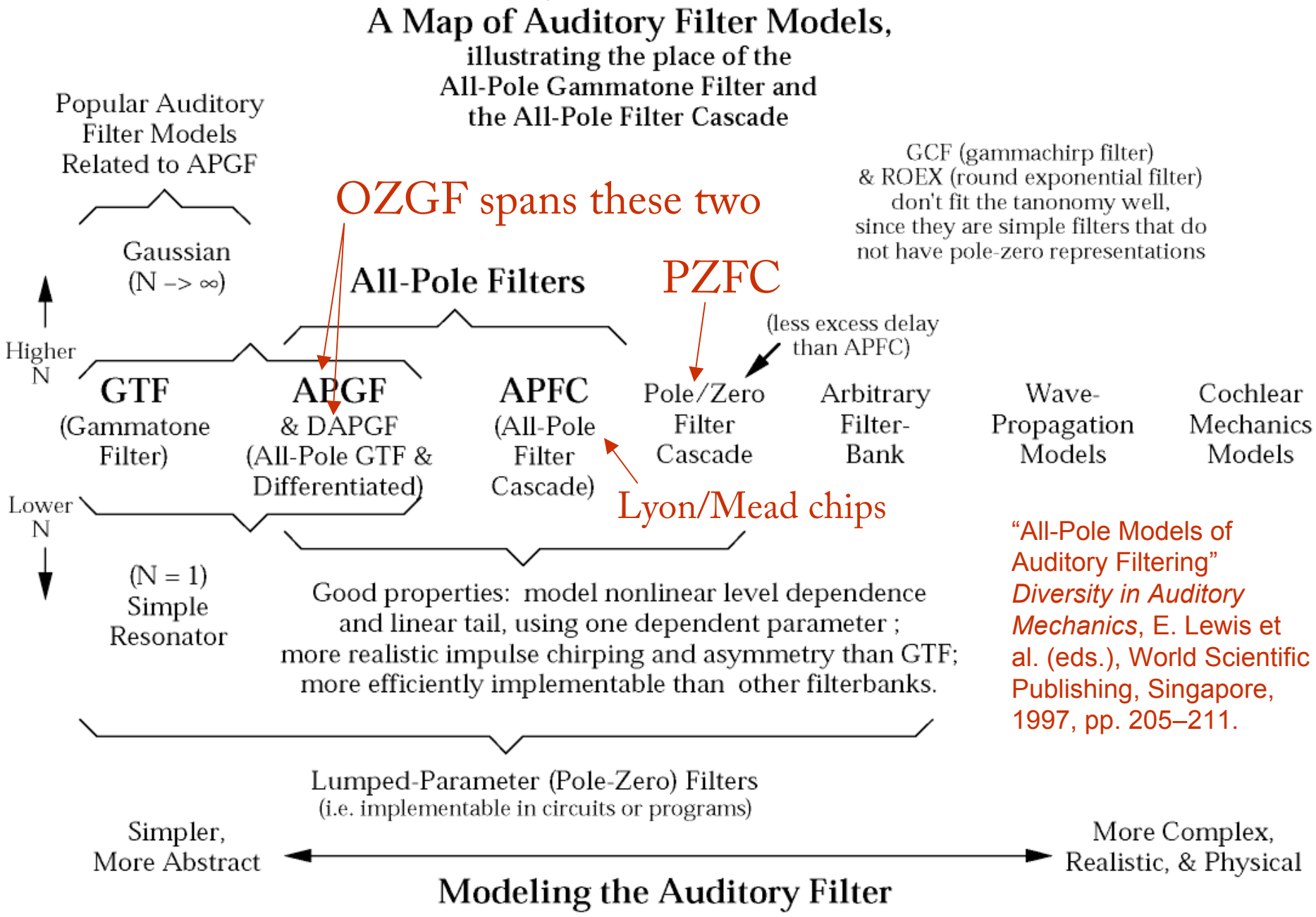



PZFC fit #516
(a)
Filter Gain (dB)
60
40
20
0
-20
-40
(b)
Gain of Tip / Tail filter (dB)
0
10
20
30
0
0.16
0.44
0.92
1.44
2.14
3.14
5.54
9.66
ERB rate
Frequency (kHz)
(c)
Equivalent Rectangular Bandwidth, ERB (Hz)
2k
1k
500
200
100
50
20
100
200
500
1k
2k
5k
10k
Center Frequency (Hz)
70 dB Probe threshold
50 dB Probe threshold
30 dB Probe threshold
Glasberg & Moore (1990)
BM Level (dB)
80
70
60
50
40
30
Linear
F c =250
F c =500
F c =1k
F c =2k
F c =3k
F c =4k
F c =6k
Probe threshold (dB) when noise masker has no notch

Response to sounds – no further compressive (log) nonlinearity needed



Lyon 1997



"All-Pole Models of Auditory Filtering" *Diversity in Auditory Mechanics*, E. Lewis et al. (eds.), World Scientific Publishing, Singapore, 1997, pp. 205–211.

PZFC Advantages – review

- Good fit to human masking data with simple params
- Connection to traveling wave allows natural coupling effects, for masking, adaptation, etc.
- Like APGF & OZGF, unity-gain tail models lossless propagation of low-frequency energy; tail doesn't wag with Q or other parameters
- Easy to implement directly as standard second-order filter sections, without further approximation
- Easy to vary parameters dynamically for "AGC"
- Low total order (complexity) for multi-channel filterbank, by sharing filter sections – total complexity just 2nd-order per channel, compared to 8th-order for gammatones

Conclusion – continuous improvement path

- RoEx family – good parameterized shapes, but not the best; no corresponding real filters
- Gammatone – too symmetric, but otherwise a good filter, not hard to implement; tail problems in real case
- Gammachirp – parameterized asymmetry, dynamically-varying peak gain; good improvement over the others, but the real filter still has tail issues
- AP/OZ GF – a chirping asymmetric filter much like GC, but with rock-steady tail behavior as gain varies; exact easy implementation, even dynamic
- PZFC – tied to traveling-wave concept; more efficient filterbank; shape fit at least as good as any others, with few parameters; can work on phase fits, too

Fitting Nonlinear Auditory Filters:
OZGF, PZFC, feedback versions, etc.

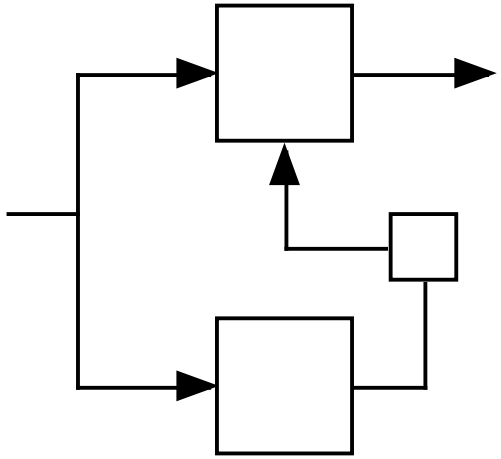
(Deriving the parameters for the human-calibrated versions of the OZGF and the PZFC from simultaneous masking data)

Irino & Unoki's Framework

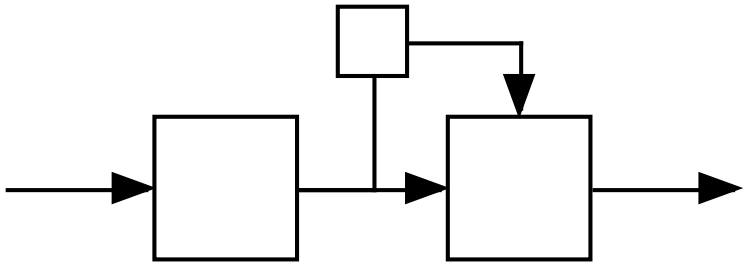
- Nonlinear optimization of parameters based on minimizing squared error in predicting masked thresholds based on tone SNR at filter output
- Also optimize over filter CF to get best SNR of the masked tone
- Level-dependent parameters depend on output of a "passive" filter with noise only (or with noise plus target tone)
- Nonlinear fit search also optimizes P0 and K
- Flexible frequency dependence of selected parameters: linear or quadratic on ERBrate scale

Level control via detection of output of “passive” filter

“passive” filter’s output level controls the “active” filter’s shape or gain, like this for parallel filters (PrlGC)

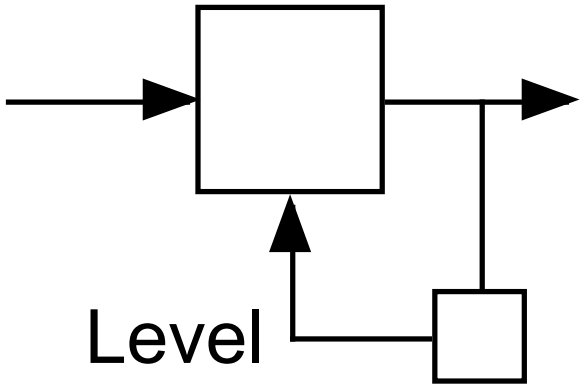


in “cascade” version (CasGC), the “passive” filter comes first, followed by the level-dependent part



Level control via detection at filter's own output

in "feedback" configuration, the main filter's output is used for level detection and control

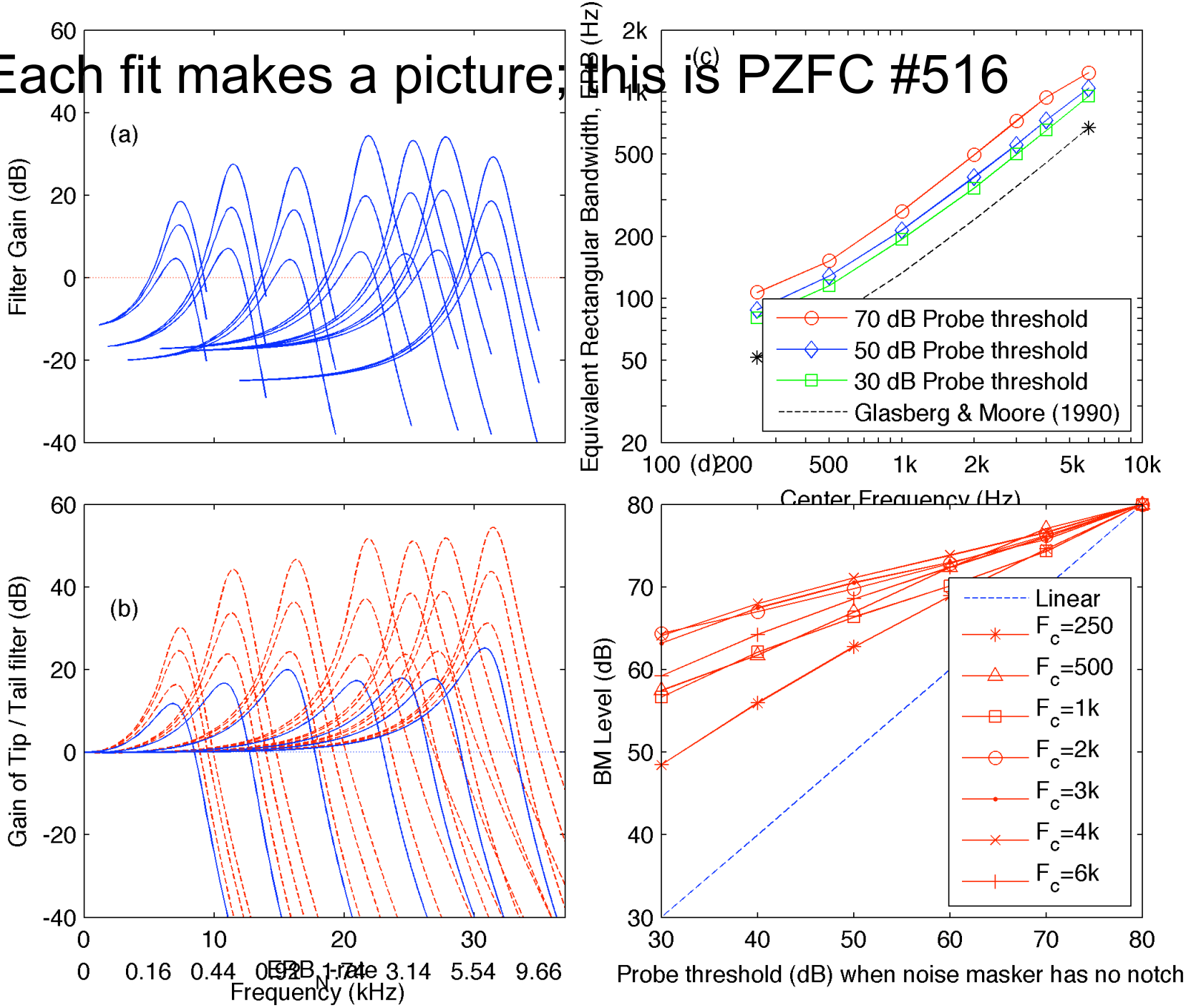


Somewhat trickier, since you need a guaranteed stable way to let it settle to a consistent level

Framework Modifications

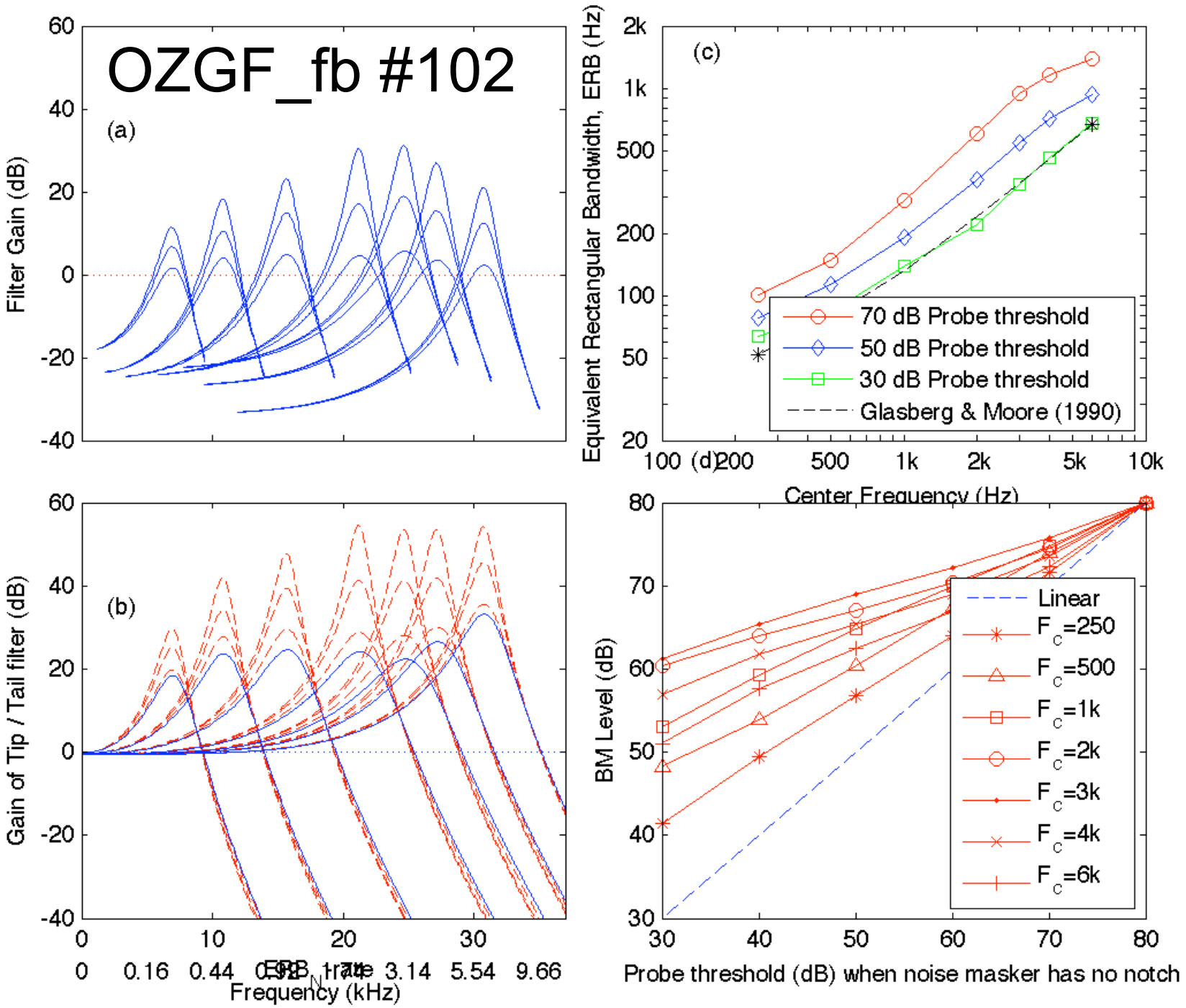
- Better convergence: near-continuous CF search to minimize confusion of estimated gradients
- Robust small-signal behavior: include P0 as an input-referred noise, accounted for in the SNR maximization; noise-only level parameters include P0
- Easily obtain optimal detection K given other parameters, instead of adding K to the search
- Feedback-type models: include a level-parameter convergence step when the main filter's output is used instead of a "passive" filter.
- Allow all models (GC, OZGF, PZFC, RoEx, etc.) to run in the same code, with minimal case switching
- Cache the level and CF between evaluations

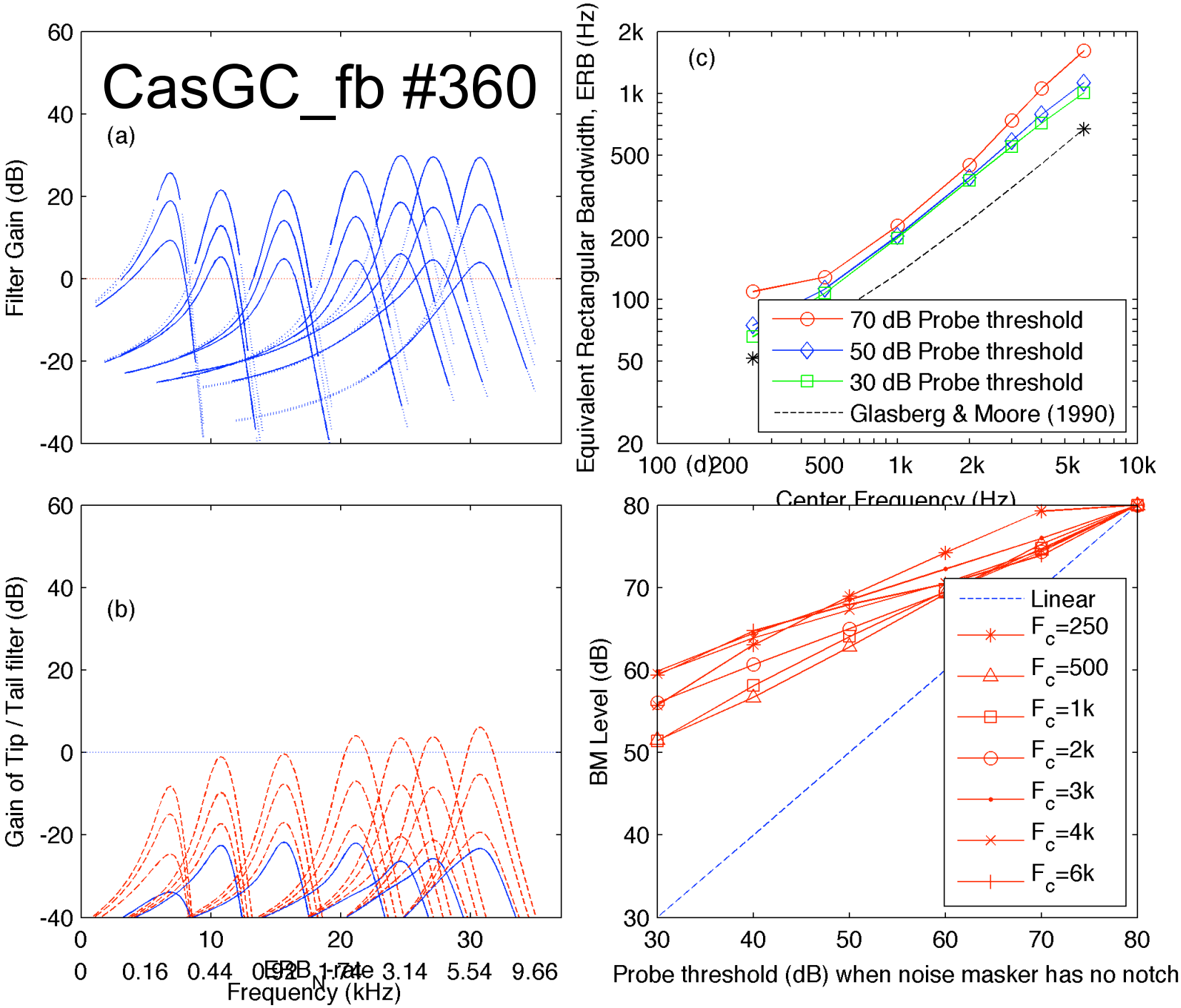
Each fit makes a picture; this is PZFC #516



PZFC Fit#516

```
  case 516  % order 2, 8 params
    FeedbackType = 1;  % enable feedback iteration
    ModelName = 'PZFC'  % pole-zero filter cascade
    ValParam = [ ...
% Final, Nfit = 516, 11-3 parameters, PZFC, cwt 0
      1.73848   0.00000   0.00000 % SumSqrErr=  10226.95
      0.62250  -1.02349   0.94190 % RMSErr   =   2.82994
      0.37208   0.00000   0.00000 % MeanErr  =   0.00000
          Inf   0.00000   0.00000 % RMSCost  =       NaN
      0.00000   0.00000   0.00000
      2.00000   0.00000   0.00000
      1.27403  -0.26291   0.21906
     11.30471   5.33017   0.33995
%    -3.63143  -1.59230   4.68184 % Kv
      ];
    CtrlParam = [ ...  % an 8-parameter fit
      1  0  0  % b1 zero BW relative to ERB
      1  1  1  % B2
      1  0  0  % B21
      0  0  0  % c    one extra zero maybe
      0  0  0  % n1 unused
      0  0  0  % n2 order, stages per nominal ERB
      1  1  1  % frat Fzero:Fpole
      1  1  1  % P0
      ];
```


OZGF_fb #102
(a)
Filter Gain (dB)
60
40
20
0
-20
-40
(b)
Gain of Tip / Tail filter (dB)
0
10
20
30
0
0.16
0.44
3.14
5.54
9.66
Frequency (kHz)
(c)
Equivalent Rectangular Bandwidth, ERB (Hz)
2k
1k
500
200
100
50
20
100
200
500
1k
2k
5k
10k
Center Frequency (Hz)
70 dB Probe threshold
50 dB Probe threshold
30 dB Probe threshold
Glasberg & Moore (1990)
BM Level (dB)
80
70
60
50
40
30
Linear
F c =250
F c =500
F c =1k
F c =2k
F c =3k
F c =4k
F c =6k
Probe threshold (dB) when noise masker has no notch


CasGC_fb #360
(a)
Filter Gain (dB)
60
40
20
0
-20
-40
(b)
Gain of Tip / Tail filter (dB)
0
10
20
30
0
0.16
0.44
0.92
1.74
3.14
5.54
9.66
ERB rate
Frequency (kHz)
(c)
Equivalent Rectangular Bandwidth, ERB (Hz)
2k
1k
500
200
100
50
20
100
200
500
1k
2k
5k
10k
Center Frequency (Hz)
70 dB Probe threshold
50 dB Probe threshold
30 dB Probe threshold
Glasberg & Moore (1990)
BM Level (dB)
80
70
60
50
40
30
Linear
Fc=250
Fc=500
Fc=1k
Fc=2k
Fc=3k
Fc=4k
Fc=6k
Probe threshold (dB) when noise masker has no notch

```
 case 360
    FeedbackType = 1;  % enable feedback iteration
    ModelName = 'CasGC_fb'  % from 12-parameter fit
    ValParam = [ ... %
% Final, Nfit = 360, 14-3 parameters, CasGC_fb, cwt 0
      3.02522   1.15581  -1.72018 % SumSqrErr=  11201.52
     -6.51804   2.64805   0.00000 % RMSErr   =   2.96171
      1.44194  -1.02186   0.00000 % MeanErr  =  -0.00000
      0.01967   0.00292   0.00000 % RMSCost  =       NaN
      1.77270   0.00000   0.00000
      3.81828   0.00000   0.00000
      0.00000   0.00000   0.00000
      0.00000   0.00000   0.00000
      4.00000   0.00000   0.00000
      9.69783   5.26747   3.93392
%    -3.36401  -3.67324   4.80991 % Kv
      ];
    CtrlParam = [ ...  %
      1  1  1  % b1
      1  1  0  % c1
      1  1  0  % Fr
      1  1  0  % Fr1
      1  0  0  % B2
      1  0  0  % C2
      0  0  0  % B21
      0  0  0  % C21
      0  0  0  % n1
      1  1  1  % P0
      ];
```

CasGC_fb Fit#360

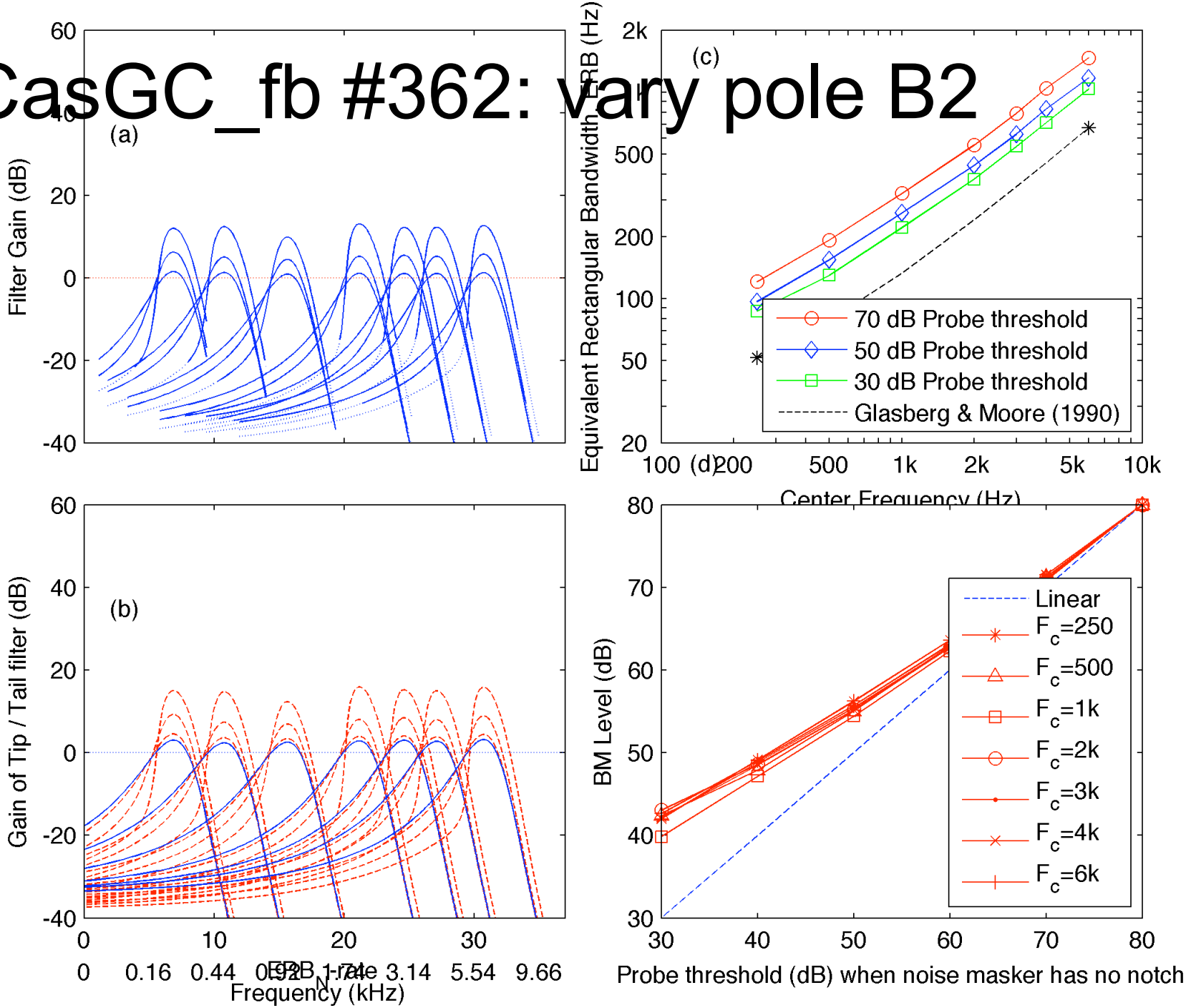
Cascade GC in feedback mode?

- Feedback can work on filter models for which it was not originally planned, sometimes.
- Notice:

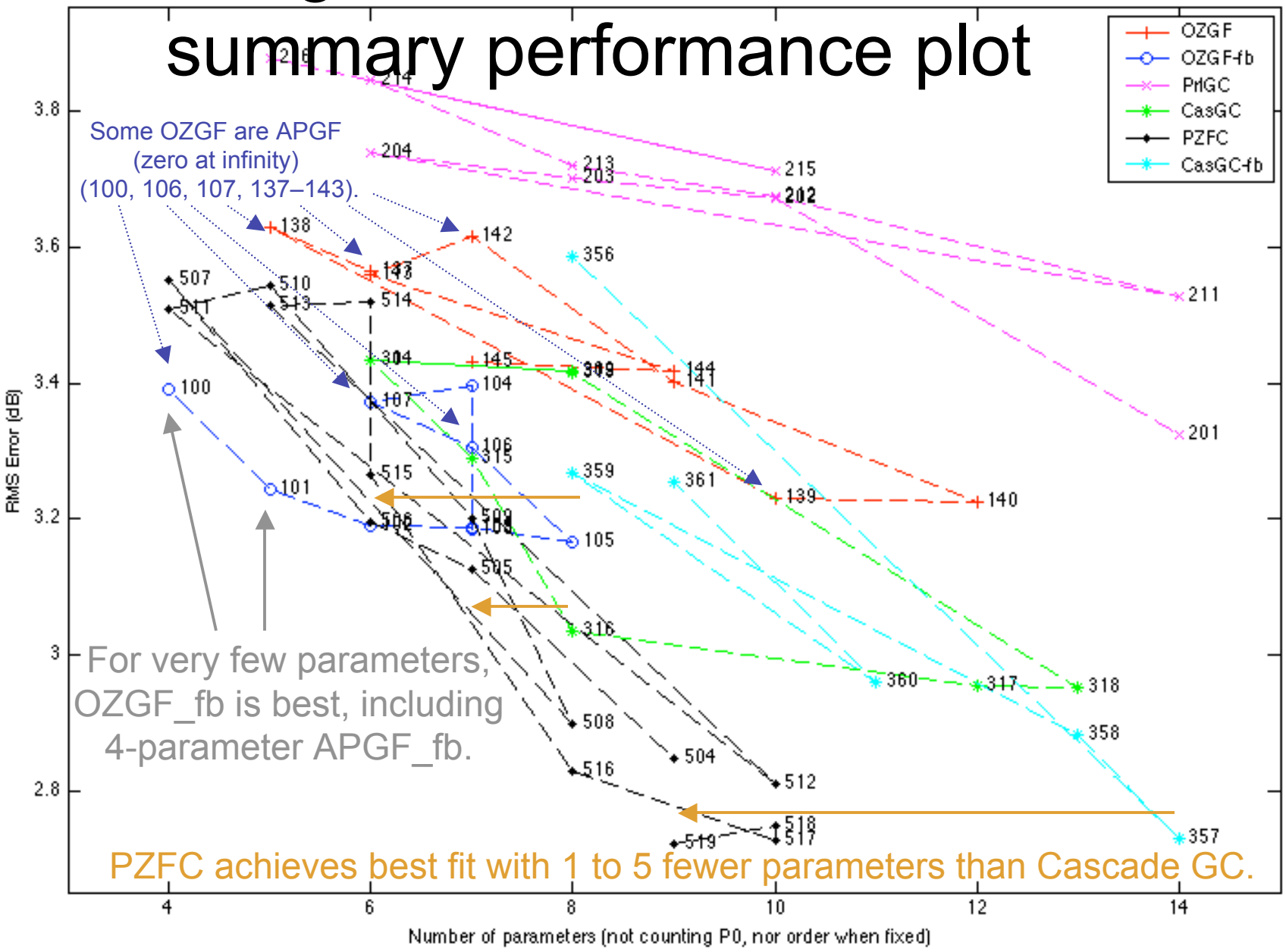
```
case 360
    FeedbackType = 1;  % enable feedback iteration
    ModelName = 'CasGC_fb'  % from 12-parameter fit
```

- This works because the level-dependence affects the gain correctly (Fr: position of high-pass active filter); attempting to use the bandwidth instead, as in CasGC Fit#316, doesn’t work well, since the model equations hold the peak gain fixed independent of bandwidth

CasGC_fb #362: vary pole B2



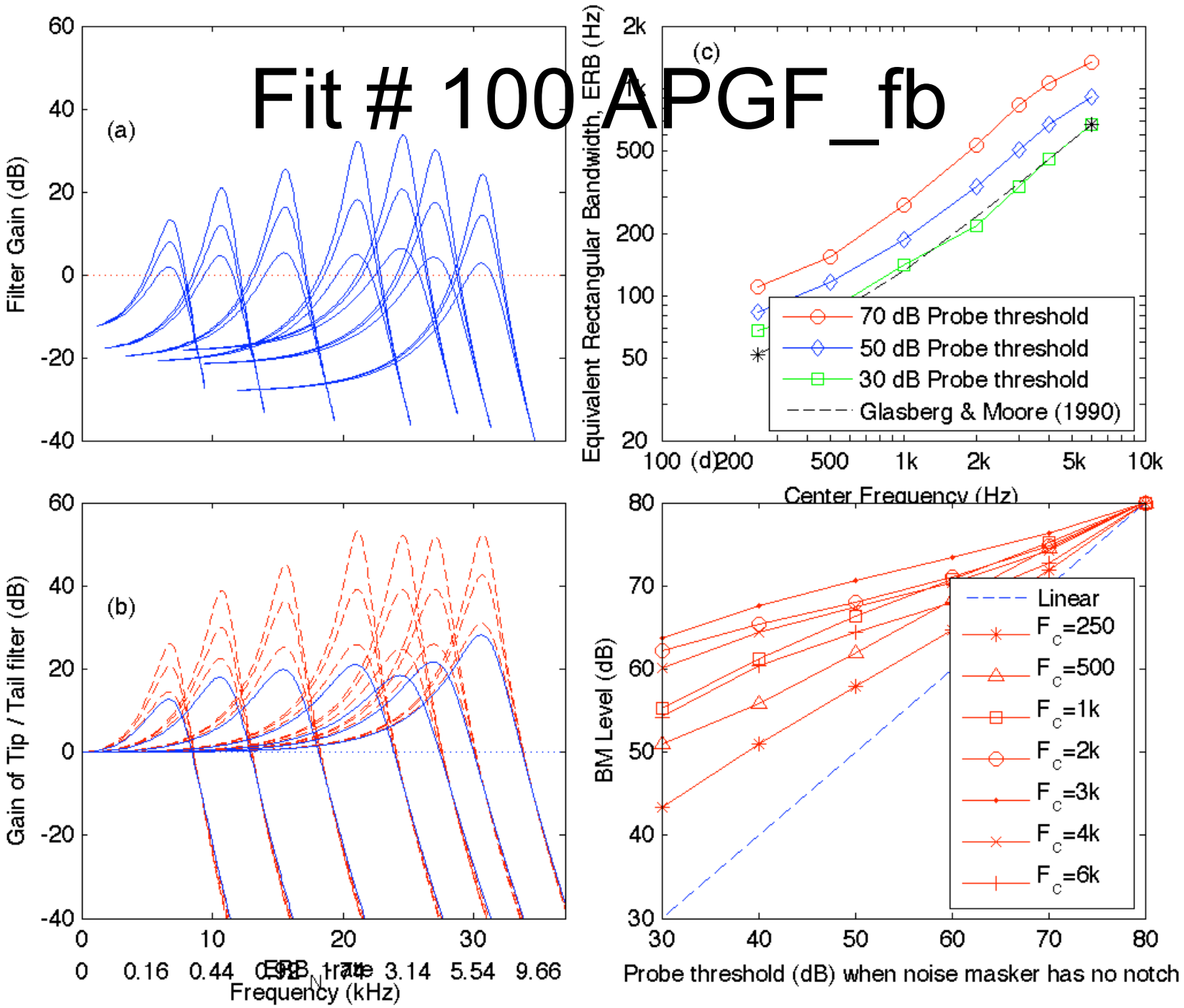
Log files combine into a summary performance plot



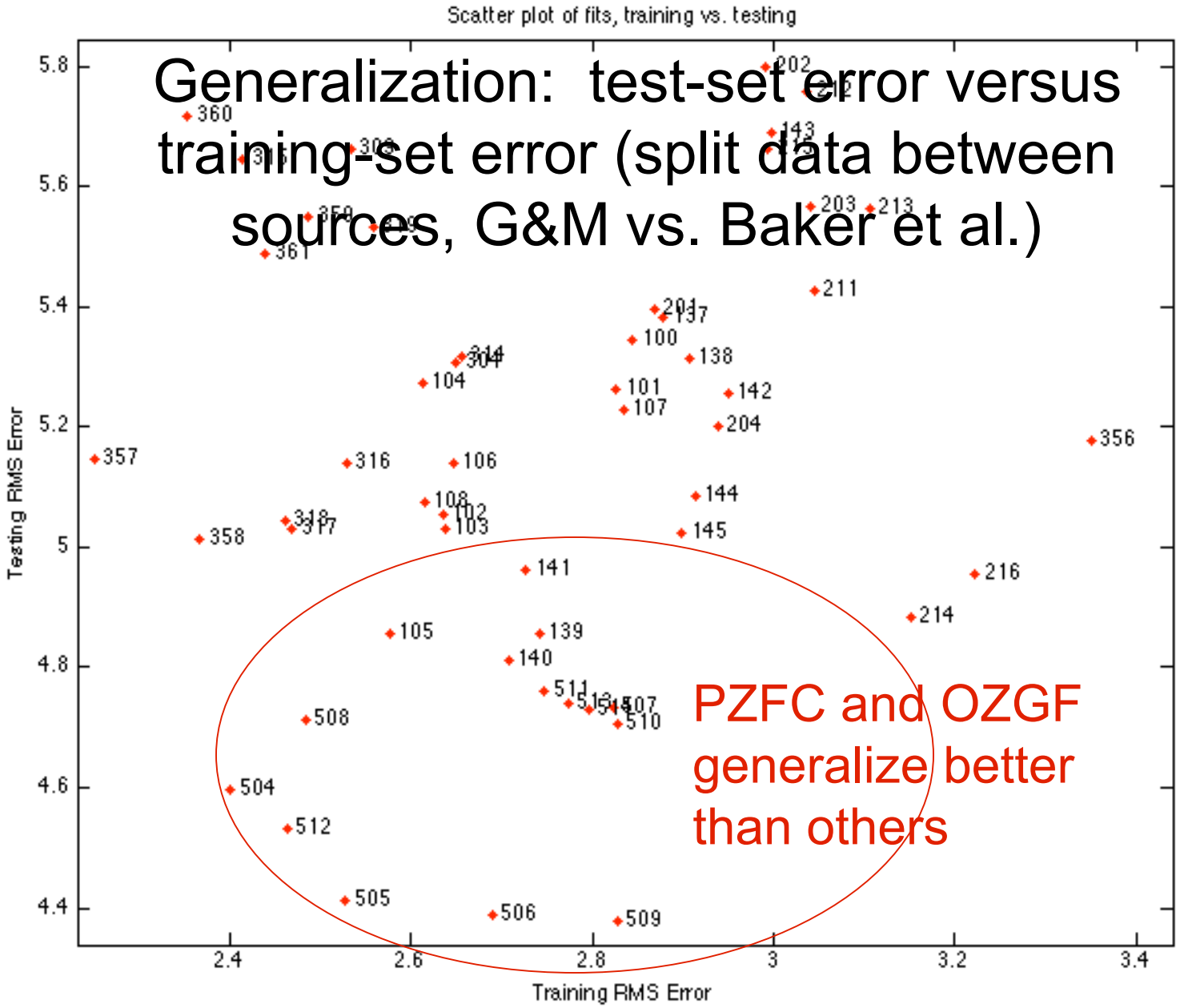
OZGF_fb Fit#100 (APGF)

```
case 100
   FeedbackType = 1;  % enable feedback iteration
   ModelName = 'OZGF_fb'  % APGF
   ValParam = [ ...
     % Final, Nfit = 100, 7-3 parameters, OZGF_fb, cwt 0
     0.00000   0.00000   0.00000 % SumSqrErr=  14684.11
     0.55936  -0.97985   0.89312 % RMSErr   =   3.39101
     0.66293   0.00000   0.00000 % MeanErr  =   0.00000
     Inf       0.00000   0.00000 % RMSCost  =       NaN
     4.00000   0.00000   0.00000
     4.00000   0.00000   0.00000
     0.00000   0.00000   0.00000
     13.59306   8.04349  -1.35988
     %    -3.04024  -0.96252   3.35127 % Kv
     ];
   CtrlParam = [ ...  % a 4-parameter fit
     0  0  0  % b1 unused in feedback version
     1  1  1  % B2
     1  0  0  % B21
     0  0  0  % c
     0  0  0  % n1
     0  0  0  % n2
     0  0  0  % frat unused
     1  1  1  % P0
     ];
```

4-parameter
4th-order APGF,
feedback version


Fit # 100 APGF_fb
(a)
Filter Gain (dB)
60
40
20
0
-20
-40
(b)
Gain of Tip / Tail filter (dB)
0
10
20
30
0
0.16
0.44
3.14
5.54
9.66
Frequency (kHz)
(c)
Equivalent Rectangular Bandwidth, ERB (Hz)
2k
500
200
100
50
20
100
200
500
1k
2k
5k
10k
Center Frequency (Hz)
70 dB Probe threshold
50 dB Probe threshold
30 dB Probe threshold
Glasberg & Moore (1990)
BM Level (dB)
80
70
60
50
40
30
Linear
F c =250
F c =500
F c =1k
F c =2k
F c =3k
F c =4k
F c =6k
Probe threshold (dB) when noise masker has no notch

Generalization: test-set error versus training-set error (split data between sources, G&M vs. Baker et al.)



Conclusion

- The filter fitting framework is a useful tool for seeing how proposed auditory filters relate to others
- The modified framework allows more types of models, including feedback configurations
- Being able to specify different model types in one framework will make this approach more accessible and useful to others
- The APGF, OZGF and PZFC are good auditory filter shapes