Tutorial workshop on methods for large-scale phonetic data analysis

John Coleman
Margaret Renwick*
Ladan Baghai-Ravary

Phonetics Laboratory, University of Oxford

* now at University of Georgia

BAAP, Oxford, 7th April 2014

Schedule

- 9:30 John Coleman: Introduction – Why “big data”?
- 9:40 Ladan Baghai-Ravary: Forced Alignment and Speech Recognition Systems
- 10:10 Margaret Renwick: Steps to Data Mining
- 10:40 Coffee Break (20 mins)
- 11:00 John Coleman and Margaret Renwick: Numerical modelling and statistical methods
- 12:00 Lunch and registration

UNIVERSITY OF OXFORD

Aims

- To give you a flavour of what phonetics is like when the amount of data gets big
- To pass on some lessons we've learned (the hard way) in “big data” phonetics projects, especially work on the Audio BNC
- To inspire you to be ambitious, and boost your confidence
- To “up the game” of the discipline

UNIVERSITY OF OXFORD

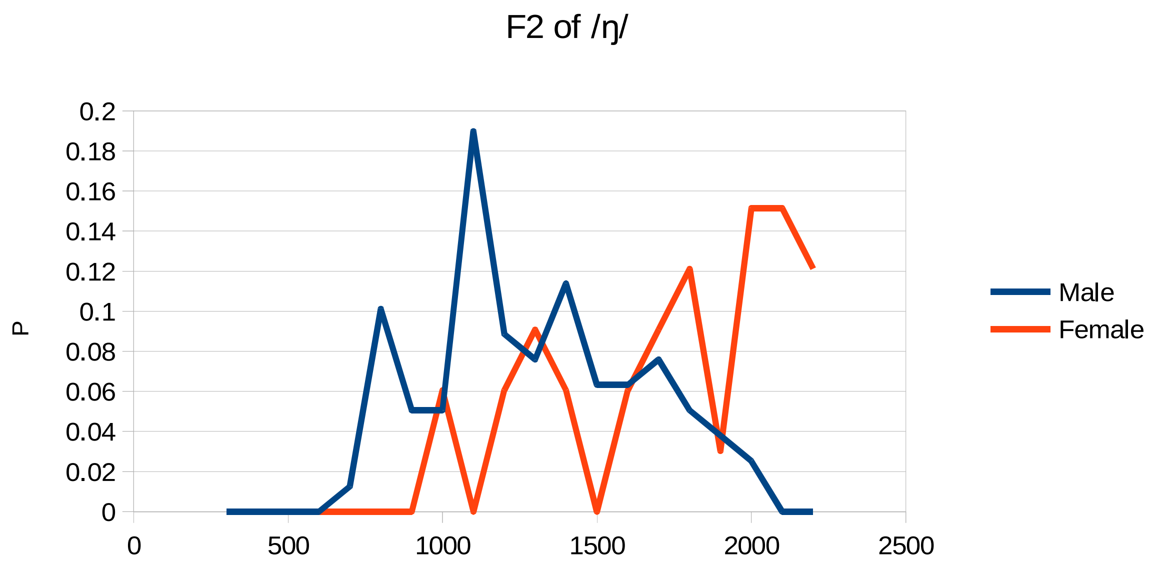
Policy

- We assume little or no prior programming ability, but this is not “programming for beginners”
- Intensive “fire-hose” method of teaching, so don’t expect to understand every step
- If anything is completely baffling, you are welcome to interrupt and ask a question, for clarification
- But: time is limited, so please keep questions on side-issues for the breaks

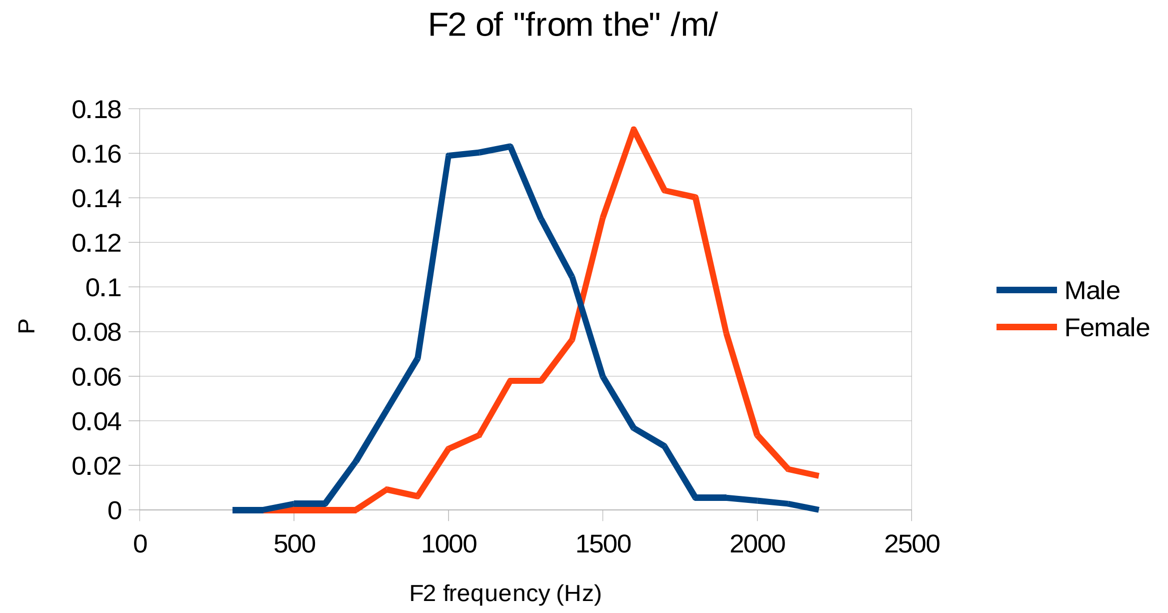
Why is big N good?

- Variation in F2 frequency of /ŋ/ (before /k/ or /g/) *vs*. “from the” /m/

  - – Male, N = 79
  - – Female, N = 33



  - – Male, N = 736
  - – Female, N = 328



UNIVERSITY OF OXFORD

How big should N be?

$$n = \left(\frac{zs}{E} \right)^2$$

where

z = level of confidence (you choose)

- for α = 0.95 ($p < 0.05$) $\rightarrow z = \pm 1.96$
- for α = 0.99 ($p < 0.01$) $\rightarrow z = \pm 2.58$

s = standard deviation in the population

E = maximum allowable error (you decide)

How big should N be?

Example 1: F2 frequency of /ŋ/ before /k/ or /g/; female speakers

for $\alpha = 0.99$ ($p < 0.01$), $z = \pm 2.58$

$s = 371$ Hz

Say $E = 10$ Hz ?

$n = (zs/E)^2 = 9{,}162$. But we only have N = 33!

Say $E = 50$ Hz ? (We used 100 Hz bins) $n = 367$!
Say $E = 100$ Hz and $z = \pm 1.96$? $n = 53$! Still not enough

UNIVERSITY OF OXFORD

How big should N be?

Example 2: F2 frequency of /m/ in “from the”, male speakers

for $\alpha = 0.99$ ($p < 0.01$) $\rightarrow z = \pm 2.58$

$s = 250$ Hz

Say $E = 50$ Hz ?

$n = (zs/E)^2 = 167$. We have N = 736 tokens, so we are OK

We could even be more picky about our measurement accuracy:

$E = 25$ Hz ? $n = 666$

What's the point?

- Typical experiment: N = 1000 data points

- Add 1 second of work per data point
  (measurement, analysis, data management, scratching your head, gazing out of the window ...)

- 1000 s = 17 minutes

What's the point?

- Now scale up to N = 1 million data points

- Add 1 second

- 1,000,000 s = 16,667 minutes
  = 278 hours
  = 11½ days (24 hours), or 35 8-hour days

- Similarly, save 35 days of drudge work by trimming 1 s
  from your work on each data point

- Even if you only have N = 10,000 data points,
  10,000 extra seconds = 2 hours 47 minutes

UNIVERSITY OF OXFORD

How to save 1 million seconds

- Principle of time management courses:
  make time by saving seconds

- Put your computer to work: learn to tell it what to do

- Avoid the graphical user interface if possible: they require *you* to work the mouse etc.

- 1 million extra keystrokes → slippery slope to RSI

- Not necessarily programming, but some command-line interactions or scripts save a lot of time

The memory stick

- 32 GB – cost us (well, cost ESRC) £18.40 each
- Contains 28 GB of data from the Audio BNC
- A valuable asset for you and for your institution
- Given on your agreement that you will not just wipe it
- 4 GB Free Space folder for your own use

- You can make copies on your own/department's computer(s) but refer to Audio_BNC_READ_ME_FIRST.html for terms

- Please visit http://www.phon.ox.ac.uk/AudioBNC and *register*

UNIVERSITY OF OXFORD

What's on the memory stick?

• html folder	- *all 911 Spoken BNC texts transcribed*
• TextGrids folder	- *408 Praat TextGrids, relating to …*
• wavs folder	- *346 selected .wav files*
• Audio_BNC_READ_ME_FIRST.html	- *overview, T's & C's*
• BNC_dict.txt	- *219,000 word variants transcriptions*
• BNCindex.html	- *list of the full Audio BNC contents*
• BNC_licence.pdf	
• BNC_transcription_alphabet.html	– *ASCII codes ↔ IPA*
• PraatSearch.html	- *how to search for words in the audio*
• thephonebook.txt	- *3.98 million segments in this sampler*
• wordtriplets.txt	- *1.13 million words-in-context*

• *In short, a goldmine of speech, so* please don't delete it

Other ways of getting BNC audio etc

- (Most of) the Audio BNC is freely available from
  http://www.phon.ox.ac.uk/AudioBNC
  but it would take an awfully long time to download it

http://bncweb.lancs.ac.uk/ has a search tool that retrieves audio extracts of conversations from our site; you can stream them, listen to them, save them, or link to them. Registration needed.

To find and obtain smaller portions (e.g. words), you'll need to learn how to “mine” the files we have provided ...

or how to use time-interval references to download specific audio segments from the AudioBNC site.

UNIVERSITY OF OXFORD

Typical “big data” workflow

- Design your research question; frame hypothesis H
- Figure out what words or phrases you need to address H
  - *we shall not be addressing these 2 steps*

- Find where in the corpus those items are (and how many there are)

  - “Forced alignment”, the killer app for finding material in large corpora, is essential here. [Ladan]

UNIVERSITY OF OXFORD

Typical “big data” workflow

- “Mine” the corpus, to find and obtain/locate/extract the portions you want to analyse
- Assess the quality of the portions. Are they the correct bits?
- Throw out the rubbish. Have you still got enough?

[Peggy]

UNIVERSITY OF OXFORD

Typical “big data” workflow

- Extract relevant acoustic parameters (using commands, not GUI)
- Check the quality of the extracted parameters
- Model the parameters (numerically) e.g. Functional Data Analysis
- Analyse the *model* parameters, using statistical methods

[Peggy and John]

UNIVERSITY OF OXFORD

Forced alignment and speech recognition

- Ladan

STEPS TO DATA MINING

Steps to data mining

✔ Alignments (Praat TextGrids)

What can the data tell us?

…What data do we have?

UNIVERSITY OF OXFORD

Searching TextGrids

Search the output of the aligner to find

(a) What’s in the recordings?

(b) How many tokens are there?

(c) When are these tokens in the audio?

Benefits of an index

Compilation of all TextGrids in one file

One segment or word (pair, triple) per line

Contains:

   Word(s), start time, end time, filename (or URI)

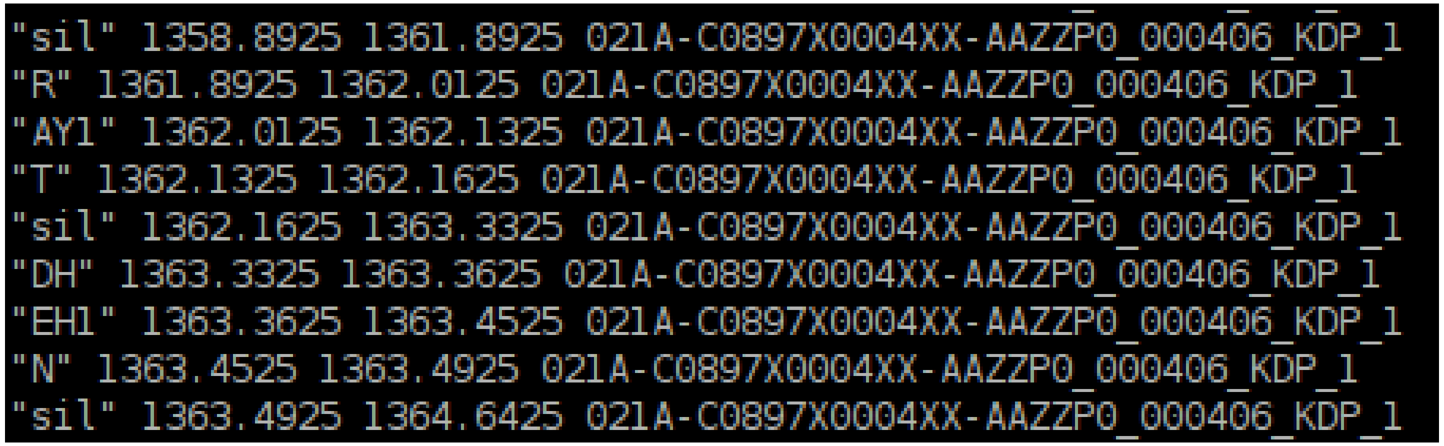
   → Provides direct access to audio information specific to those words

Indexing data

Master indexes

- Permit corpus-wide search
- Include start & end times for words or segments

Example: Entries in “thephonebook.txt”



```
"sil" 1358.8925 1361.8925 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"R" 1361.8925 1362.0125 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"AY1" 1362.0125 1362.1325 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"T" 1362.1325 1362.1625 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"sil" 1362.1625 1363.3325 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"DH" 1363.3325 1363.3625 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"EH1" 1363.3625 1363.4525 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"N" 1363.4525 1363.4925 021A-C0897X0004XX-AAZZP0_000406_KDP_1
"sil" 1363.4925 1364.6425 021A-C0897X0004XX-AAZZP0_000406_KDP_1
```

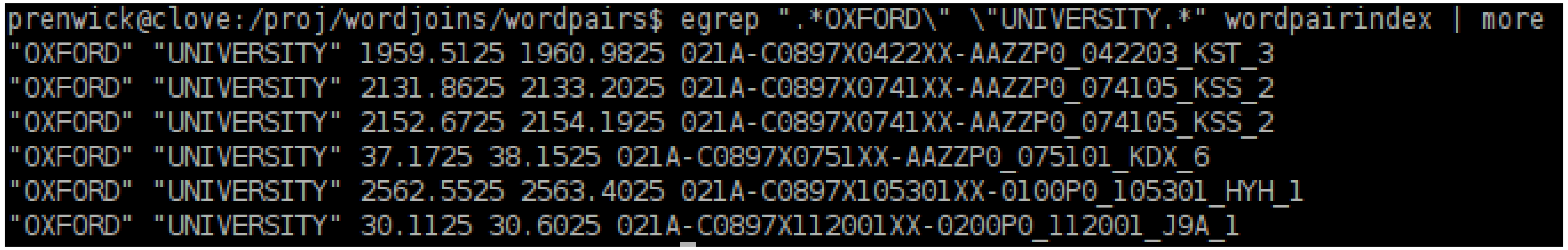
Finding data

Search the index for words, sounds, etc.

- Total word pairs in the Audio BNC: 3,759,845
- Total phone count for the Audio BNC: 20,146,977
  - (count includes silences)

Example:

“Oxford University” appears six times in the Audio BNC.



```
prenwick@clove:/proj/wordjoins/wordpairs$ egrep ".*OXFORD\" \"UNIVERSITY.*" wordpairindex | more
"OXFORD" "UNIVERSITY" 1959.5125 1960.9825 021A-C0897X0422XX-AAZZP0_042203_KST_3
"OXFORD" "UNIVERSITY" 2131.8625 2133.2025 021A-C0897X0741XX-AAZZP0_074105_KSS_2
"OXFORD" "UNIVERSITY" 2152.6725 2154.1925 021A-C0897X0741XX-AAZZP0_074105_KSS_2
"OXFORD" "UNIVERSITY" 37.1725 38.1525 021A-C0897X0751XX-AAZZP0_075101_KDX_6
"OXFORD" "UNIVERSITY" 2562.5525 2563.4025 021A-C0897X105301XX-0100P0_105301_HYH_1
"OXFORD" "UNIVERSITY" 30.1125 30.6025 021A-C0897X112001XX-0200P0_112001_J9A_1
```

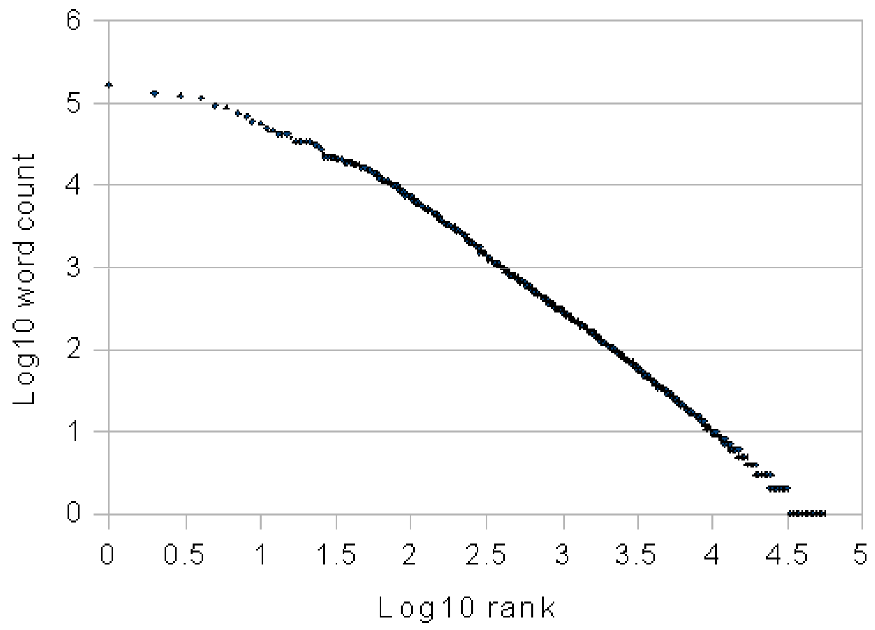
“Only six times?!”

Natural language is unbalanced: linguistic units are not equally distributed.

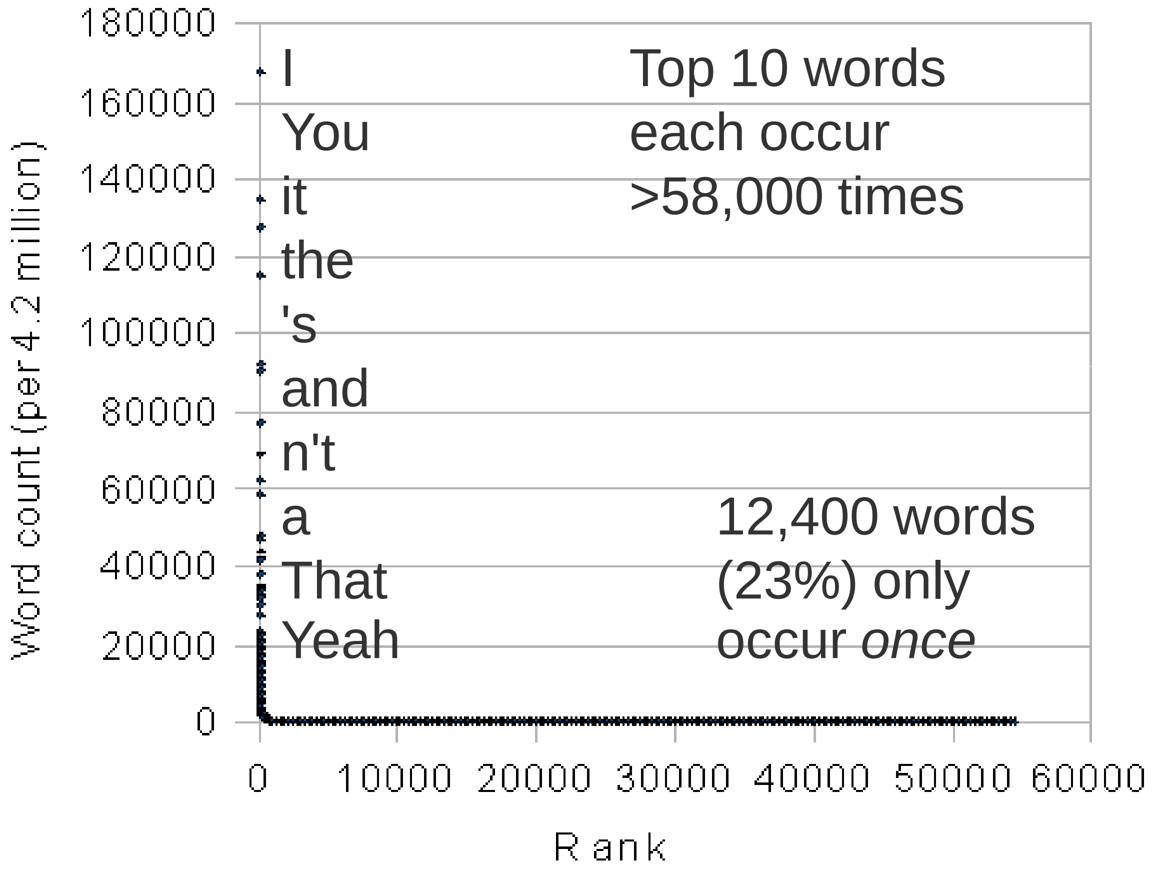
Relative frequency in a corpus

- Some structures are intensely more common than others
- A sufficiently large sample reflects relative frequencies in a language

Zipf’s Law



Relative frequency and the Audio BNC



Just listening and waiting, how long till items show up?

	For the 1st token, listen for	
[ʒ], the least frequent English phoneme (i.e. to get all English phonemes)	13 minutes	
"*twice*" (1000th most frequent word in the Audio BNC)	14 minutes	
"*from the*" (the most frequent word-pair in our nasals study)	17 minutes	
"*railways*" (10,000th most frequent word)	26 hours	
"*getting paid*" (the least frequent word-pair occurring >10 times in our study)	95 hours (4 days)	

UNIVERSITY OF OXFORD

Just listening and waiting, how long till items show up?

	For the 1st token, listen for	**For 10 tokens, listen for**
[ʒ], the least frequent English phoneme (i.e. to get all English phonemes)	13 minutes	5 hours
“*twice*” (1000th most frequent word in the Audio BNC)	14 minutes	44 hours
“*from the*” (the most frequent word-pair in our nasals study)	17 minutes	22 hours
“*railways*” (10,000th most frequent word)	26 hours	41 days without sleep
“*getting paid*” (the least frequent word-pair occurring >10 times in our study)	95 hours (4 days)	37 days

UNIVERSITY OF OXFORD

Example: Searching an index

Step 1: Find a techno-buddy, if needed

Step 2: Open a Terminal/shell window
  OS X: F4 → type Terminal

cd to your workshop directory, e.g.

```
cd /Users/peggy/workshop

cp /Volumes\USB DISK/thephonebook.txt .
```

Type **ls** (and hit Return) to see directory contents

View a file’s beginning:

```
head thephonebook.txt
head -20 thephonebook.txt
```

UNIVERSITY OF OXFORD

Searching for phones

Search for all instances of “M”:

```
grep \"M\" thephonebook.txt
```

Look at only the first few:

```
grep \"M\" thephonebook.txt | head
```

How many are there?

```
grep \"M\" thephonebook.txt | wc -l
```

Comparing phone counts

Search for several phones simultaneously:

```
grep \"[LMNP]\" thephonebook.txt |
  awk '{ print $1 }' |
  sort |
  uniq -c |
  sort -nr
```

Relative frequencies of phones

```
$ grep \"[LMNP]\" thephonebook.txt | awk '{
  print $1 }' | sort | uniq -c | sort -nr
256177 "N"
125880 "L"
110016 "M"
62614 "P"
```

Occurrence of homorganic [mp] limited by frequency of [p]

N is four times more frequent than P.

Can we have a balanced data sample?

Not with spontaneous speech!

Automatic alignment in linguistic research

Automatic alignment is essential

- Much faster than human aligners
- Assigns boundaries even in poor audio signals

…but the alignments are not perfect

- Noise & disfluencies cause alignment errors
- Transcription–audio mismatches throw off alignments
- Poor signal quality makes alignment difficult

UNIVERSITY OF OXFORD

Noise in big data

The Big Data ideal:

Some statistical “noise” from outliers
…but a surfeit of data prevents skewed results

The phonetic reality:

- Subtle phonetic differences *are* disturbed by bad tokens
- Additional high variation across individual speakers

→ Filtering out bad data is essential

UNIVERSITY OF OXFORD

Human quality control

Listen to all tokens of interest

- “Do I hear the words I expect?”
- Eliminate all misaligned tokens

  …*or* search the corpus by hand for more tokens

Accuracy rates

- Word Joins: approx. 67% pairs well-aligned
- BAAP memory stick: higher aligner accuracy

UNIVERSITY OF OXFORD

Big Data tactics for quality control

How do we save seconds of listening?

Use a script:

- Input: List of locations in TextGrids
- Plays the audio
- Do you hear what you want to hear?
- If you do, the script extracts the audio to a .wav file
- Generates a list of which tokens are well-aligned (or not)

Searching & listening

Generate the input:

```
grep "\"LADIES\" \"AND\" \"GENTLEMEN\""
  wordtriplets.txt > ladies_gentlemen.txt
```

Run the listening script:

```
  /Volumes/USB\
  DISK/AudioBNC_sample_for_BAAP/check_wordpair_BAAP.py
  ladies_gentlemen.txt .1
```

Type **y** if you hear “ladies and gentlemen”, **n** if you don’t

Move the results to a subfolder:

```
  mkdir LADIES_AND_GENTLEMEN
  mv LADIES*.wav LADIES_AND_GENTLEMEN
```

Scripted workflows

`awk`: text-based data wrangling

`sox`: command-line audio processing

`wget`: downloads content from a web URL

`Praat`: scriptable software for phonetic analysis

`esps`: command-line phonetic analysis

R: complex data transformations, scripted plotting (also statistical analysis)

…and many more!

UNIVERSITY OF OXFORD

Example: a Praat script

Open Praat

  Praat → Open Praat script…

Select `extract-f0-BAAP.praat` → Open

- Reads in your .wav files
- Creates a pitch object for each
- Gets f0 measurements every 10 ms
- Writes the results to a .csv file

Get directory: in Terminal, **cd** to the `LADIES_AND_GENTLEMEN` directory.
  Type **pwd** and paste the result into Praat’s Directory window.

UNIVERSITY OF OXFORD

Running the Praat script

Get directory: in Terminal, **cd** to the `LADIES_AND_GENTLEMEN` directory. Type **pwd** and copy the result.

In Praat script: Run → Run

location: `/Users/mrenwick/workshop/LADIES_AND_GENTLEMEN/`
paste in your working directory

output: `/Users/mrenwick/workshop/ladies_f0.csv`

UNIVERSITY OF OXFORD

Data mining for phonetics

Automated analysis saves time at every step

Using simple tools & scripts

- Saves valuable seconds, minutes, hours
- Boosts reproducibility
- Allows us to analyze **more data**!
- is fun!

UNIVERSITY OF OXFORD

Coffee Break

Back at 11:00

Analyzing f_0 in “FIFTEEN”

Search for tokens & look at output

In your `workshop` directory:

```
  grep "^\"FIFTEEN\"" wordtriplets.txt |
  head

grep "^\"FIFTEEN\"" wordtriplets.txt |
  wc -l
```

UNIVERSITY OF OXFORD

Analyzing f_0 in “FIFTEEN”

There are 172 tokens of the word “FIFTEEN” on the memory stick

```
grep "^\"FIFTEEN\"" wordtriplets.txt | tr '_' ' ' |
awk '{print "play " $4 ".wav", "trim " $8, $9-$8}'
>play15
```

```
chmod +x play15
```

```
./play15
```

Scripted workflows

Trim and save the audio clips rather than playing them

```
cat play15 |
   awk '{print "sox",$2,"Fifteen/FIFTEEN" NR ".wav",$3,$4,$5}'
   >trim15
```

wget

Search for tokens, write to file for `wget`

```
grep "^\"FIFTEEN\"" wordtriplets.txt |
tr '_' ' ' |
awk '{print "http://bnc.phon.ox.ac.uk/data/"
$4 ".wav?t="$8","$9 }' > wgetlist
```

Download Audio BNC clips

Make directory for `wget` output:

```
mkdir FIFTEEN_wavs
```

Run wget:

```
wget -i wgetlist --directory-prefix=FIFTEEN_wavs/
```

UNIVERSITY OF OXFORD

Rename files

```
cd FIFTEEN_wavs
```

Make a renaming script:

```
ls | awk ’{print "sox", "FIFTEEN_wavs/"$0,
  "FIFTEEN_clips/FIFTEEN"NR".wav"}' > ../rename_wavs
```

```
cd ../
```

```
mkdir FIFTEEN_clips
```

Make script executable: `chmod +x rename_wavs`

Run script: `./rename_wavs`

UNIVERSITY OF OXFORD

Coffee Break

Back at 11:00

Welcome back!

If you didn’t manage to keep up so far, the “good” versions of “FIFTEEN” are in the Good.zip download.

Data modelling

- Case study 1: intonation of FIFTEEN
- Anti-TOBI: modelling the whole contour
- Fitting a function to a contour
- so you can analyse the parameters of the fitted function
- Case study 2: fitting a probability density function to characterise the mean and standard deviation of F2 frequency in /m/

Extracting f_0

Open Praat

  Praat → Open Praat script…

Select `extract-f0-BAAP.praat` → Open

In Praat script: Run → Run

  location: `/Users/mrenwick/workshop/FIFTEEN_clips/`

  output: `/Users/mrenwick/workshop/fifteen_f0.csv`

UNIVERSITY OF OXFORD

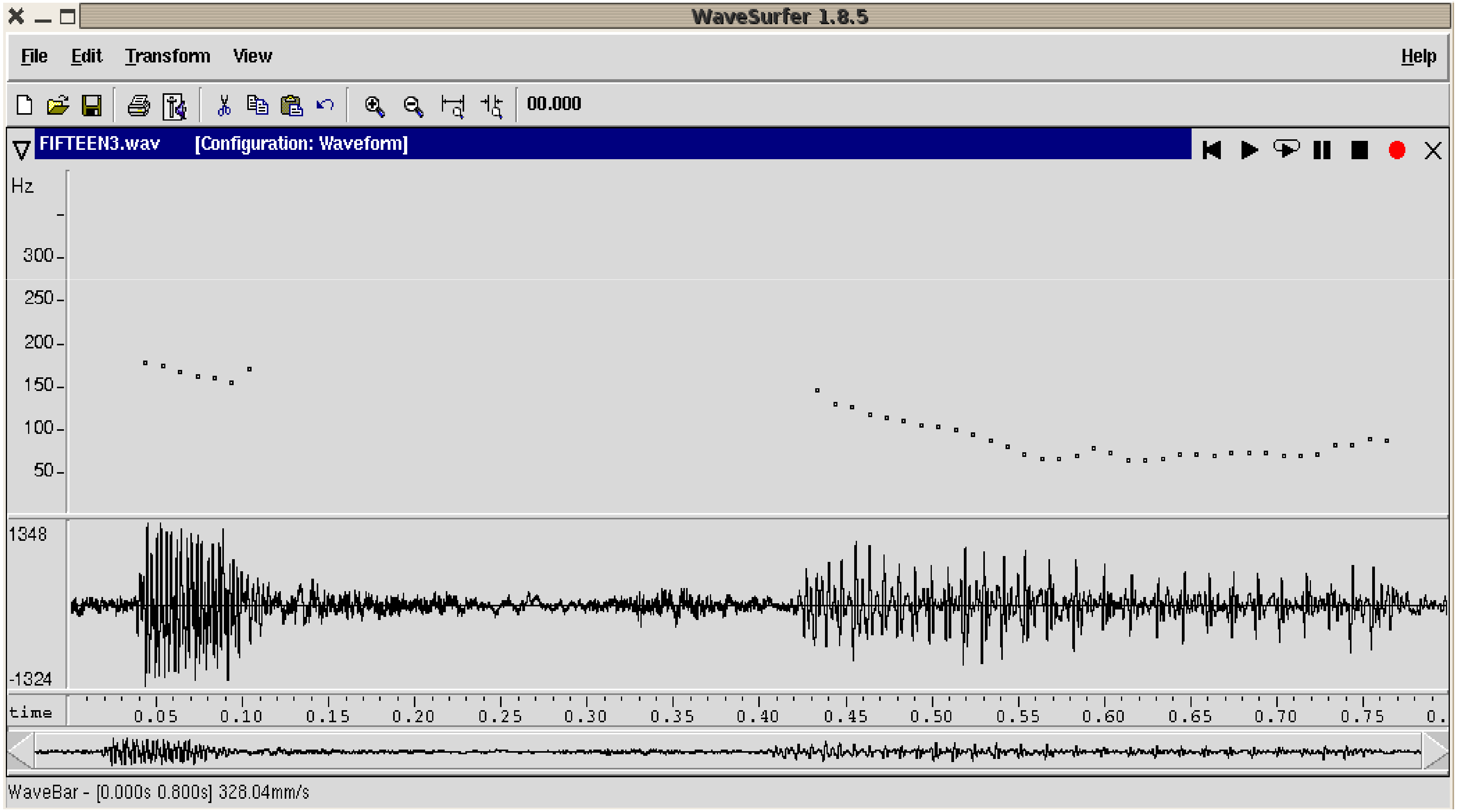
Filter bad data

Run filtering scripts:

```
for i in `awk -F, -v threshold=3 -f data-filter.awk <
  fifteen_f0.csv | awk '{ print $3 }'`;
> do grep $i, fifteen_f0.csv; done |
> awk -F, '{ if ($2!="--undefined--") print $0 }' >
  fifteen_f0_filtered.csv
```

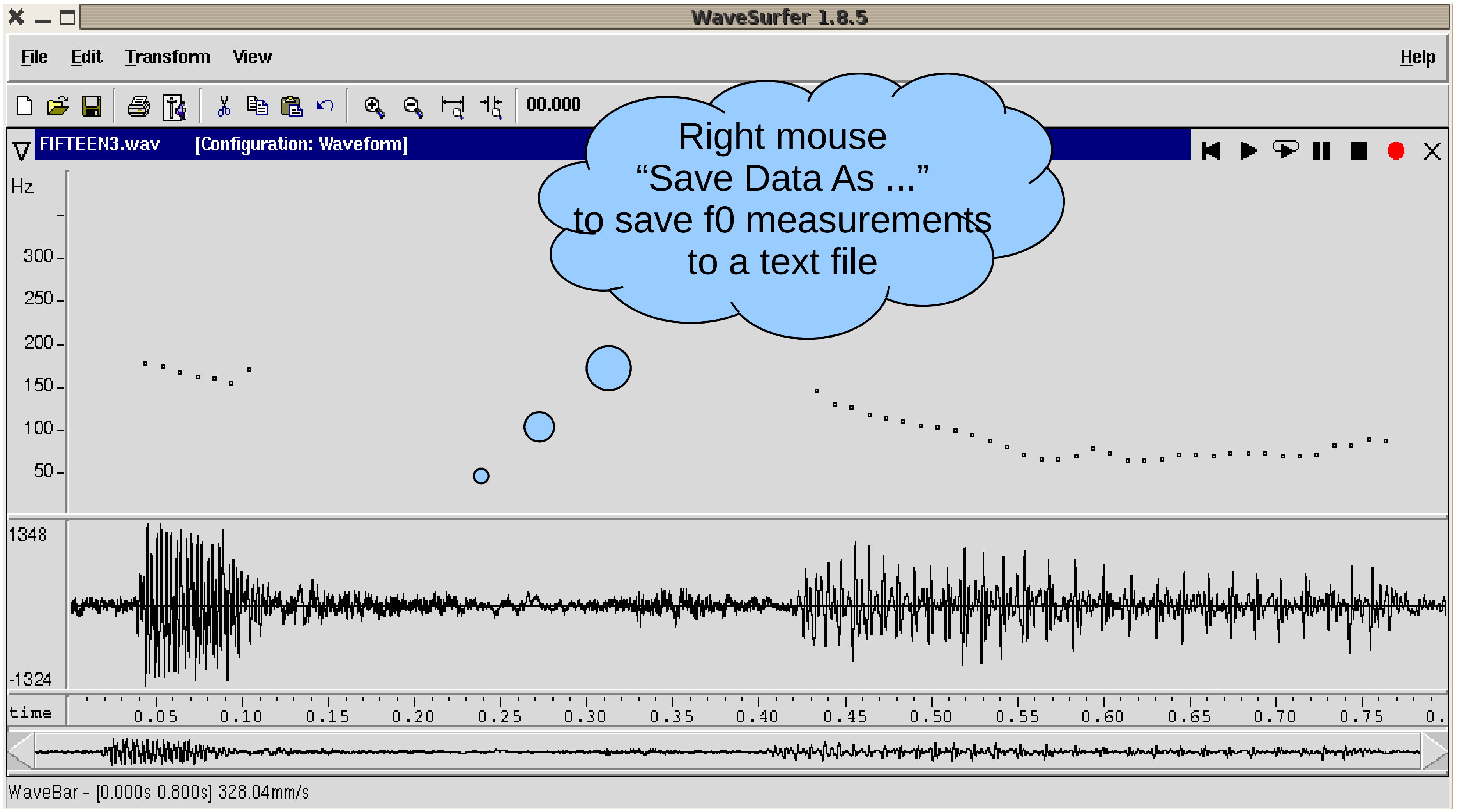
Intonation of FIFTEEN

- Using wavesurfer, ESPS get_f0 to obtain f_0 time-series



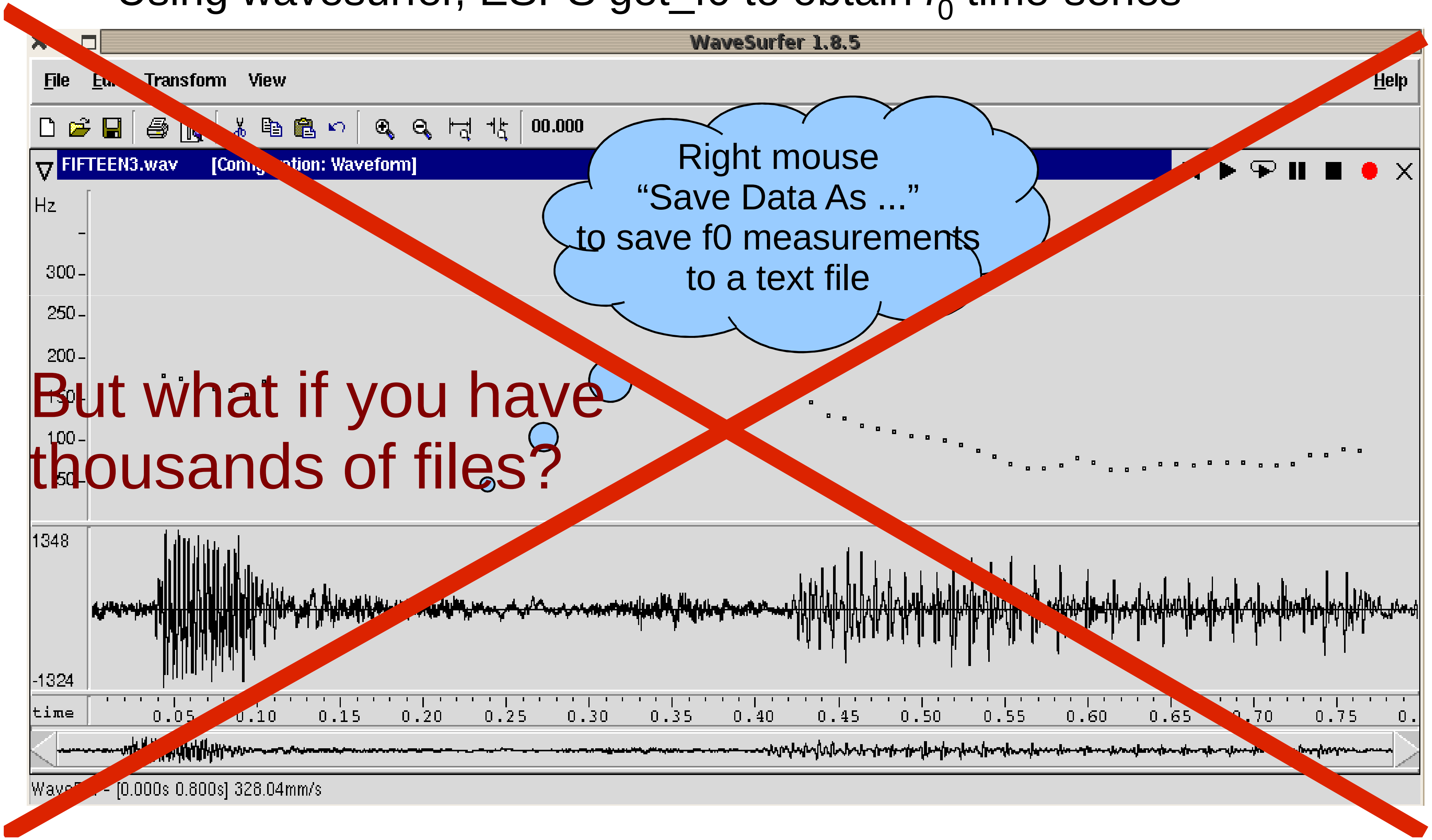
Intonation of FIFTEEN

- Using wavesurfer, ESPS get_f0 to obtain f_0 time-series


WaveSurfer 1.8.5
File Edit Transform View Help
00.000
FIFTEEN3.wav [Configuration: Waveform]
Hz
300
250
200
150
100
50
1348
-1324
time
0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65 0.70 0.75 0.
WaveBar - [0.000s 0.800s] 328.04mm/s
Right mouse “Save Data As ...” to save f0 measurements to a text file

Intonation of FIFTEEN

- Using wavesurfer, ESPS get_f0 to obtain f_0 time-series



Right mouse "Save Data As ..." to save f0 measurements to a text file

But what if you have thousands of files?

Intonation of FIFTEEN

- wavesurfer uses the ESPS get_f0 command to obtain f_0 time-series
- syntax: get_f0 [options] *input_file output_file*

ESPS is a package of UNIX-like commands and programming libraries for speech signal processing.

You can download a recent .deb package for ESPS from http://www.phon.ox.ac.uk/releases

David Talkin's paper on get_f0 is here: http://www.ee.columbia.edu/~dpwe/papers/Talkin95-rapt.pdf

Intonation of FIFTEEN

- Using wavesurfer, ESPS `get_f0` to obtain f_0 time-series
- syntax: `get_f0` [options] *input_file output_file*

```
for i in *.wav
> do get_f0 $i $i.f0
> pplain $i.f0 >$i.f0.csv
> done
```

On one line, that's:

```
for i in *.wav; do get_f0 $i $i.f0; pplain $i.f0 >$i.f0.csv; done
```

UNIVERSITY OF OXFORD

- These .csv files are simple ASCII text files like this: ------------→

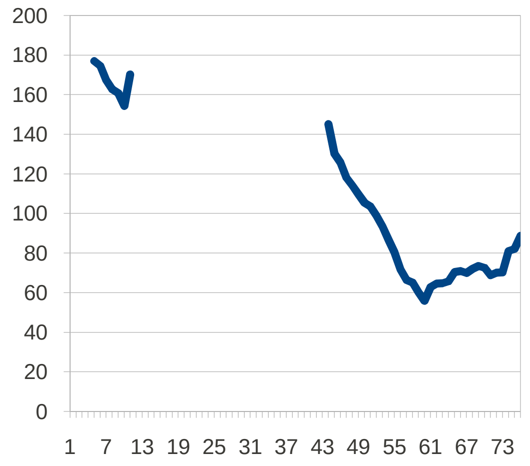
- The first column is f_0, the second is voicing; ignore the other two

```
0 0 0 0.272821
0 0 34.4253 0.551759
0 0 44.9999 0.641592
0 0 242.326 0.515299
176.894 1 401.017 0.553706
174.434 1 399.113 0.931412
167.352 1 378.998 0.951326
162.623 1 358.704 0.927735
160.734 1 356.843 0.931884
154.345 1 250.132 0.617554
170.107 1 159.65 0.843205
0 0 82.2662 0.494668
0 0 92.7429 0.730789
0 0 110.42 0.433576
0 0 71.1711 0.53332
0 0 59.6541 0.419894
0 0 62.9074 0.538716
0 0 53.2908 0.319767
0 0 47.034 0.288603
0 0 38.5281 0.346631
0 0 47.2128 0.452121
0 0 50.4091 0.43822
0 0 44.2441 0.568226
--More--(28%)
```

UNIVERSITY OF OXFORD

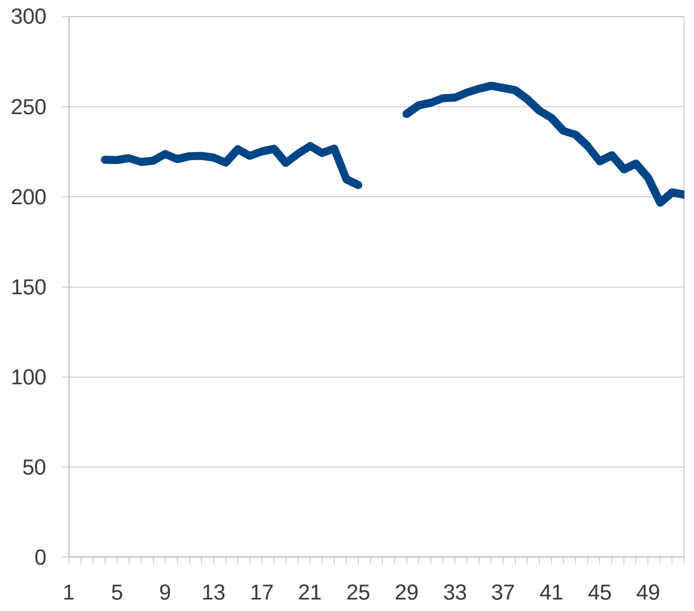
Intonation of FIFTEEN3

- You can open a .csv file in a spreadsheet programme, and plot the data (see, you don't have to have Praat or wavesurfer to draw speech parameters)



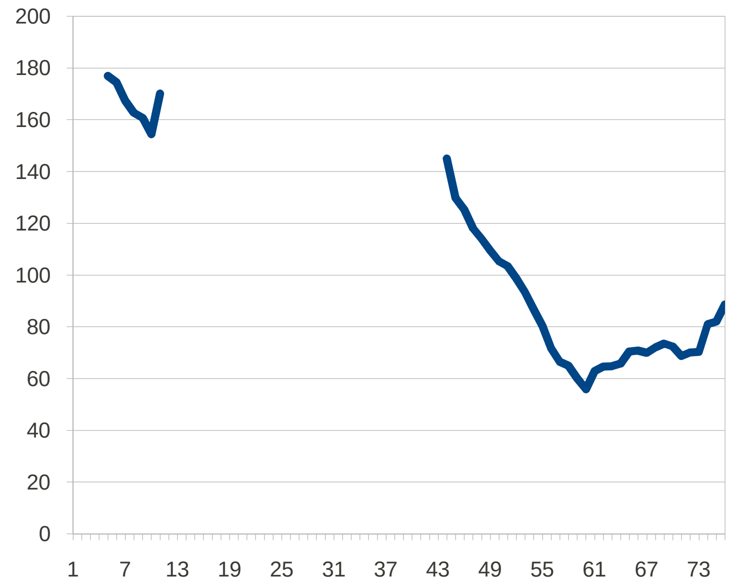
Intonation of FIFTEEN165

- You can open a .csv file in a spreadsheet programme, and plot the data (see, you don't have to have Praat or wavesurfer to draw speech parameters)



How you *could* analyse intonation ...

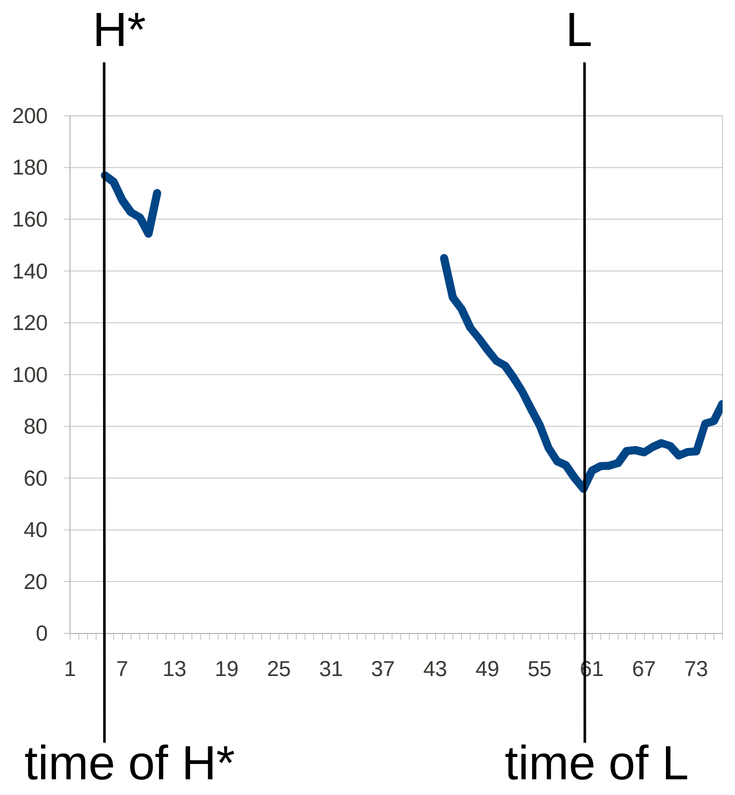
- Many approaches to phonetic analysis focus on particular points of interest in the time series, e.g.



UNIVERSITY OF OXFORD

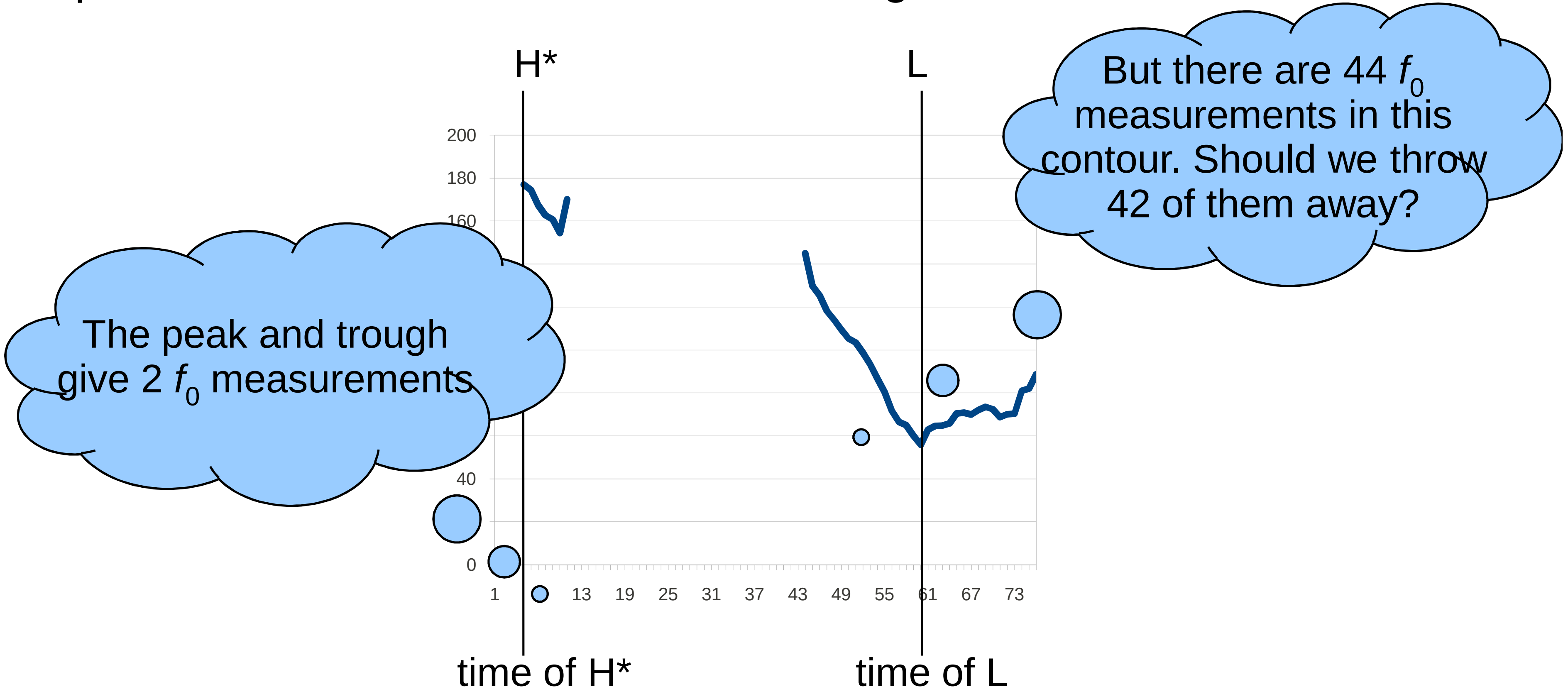
How you *could* analyse intonation ...

- Many approaches to phonetic analysis focus on particular points of interest in the time series, e.g.



How you *could* analyse intonation ...

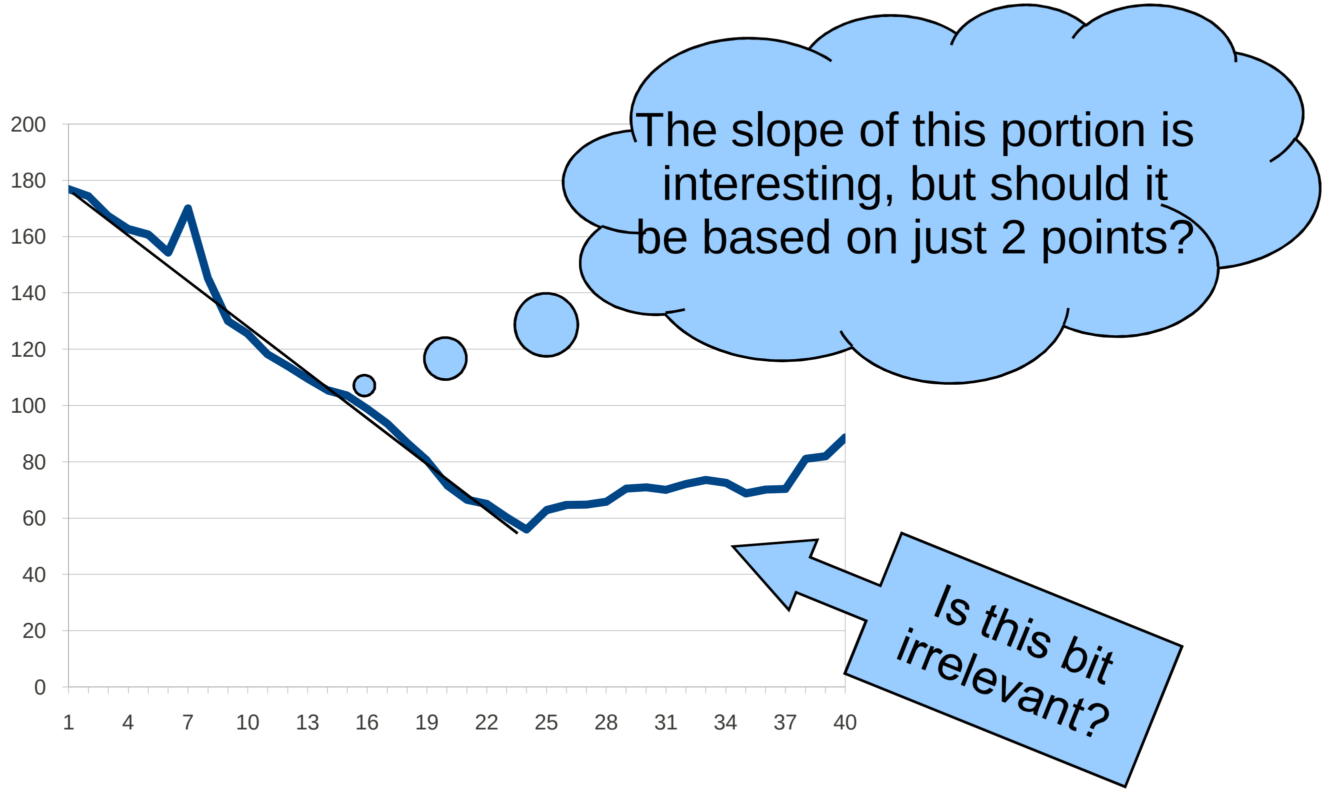
- Many approaches to phonetic analysis focus on particular points of interest in the time series, e.g.



UNIVERSITY OF OXFORD

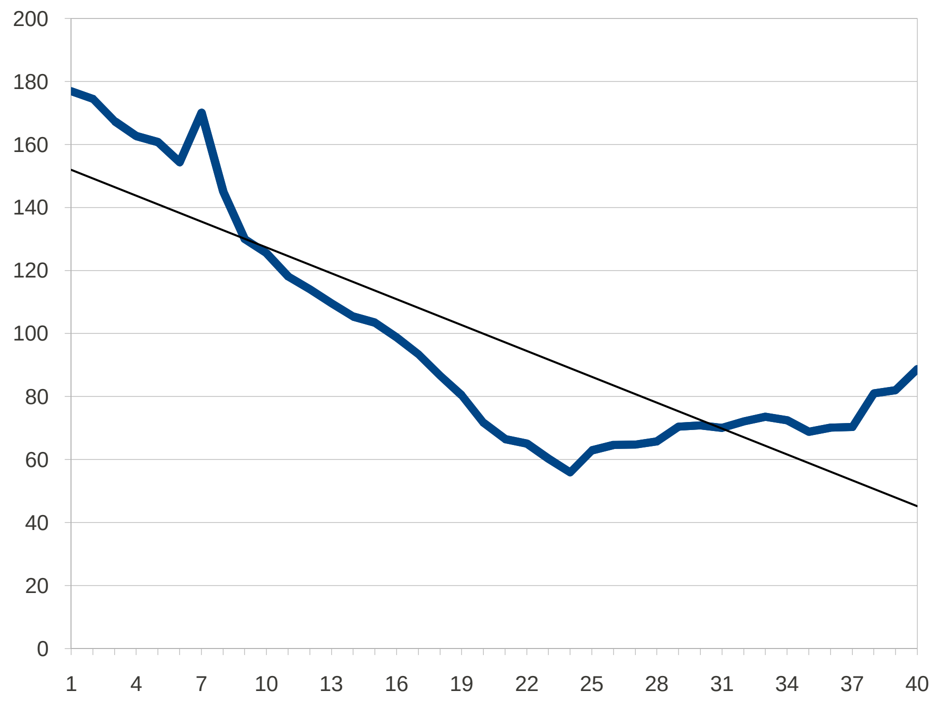
Intonation of FIFTEEN3

- With the discontinuity (voiceless portion) excised:



Intonation of FIFTEEN3

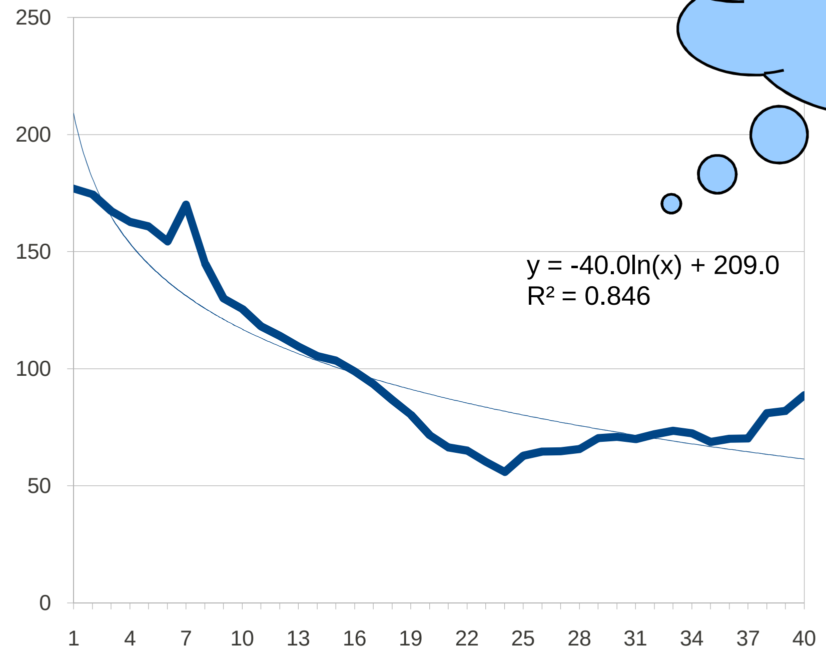
- Cf. a linear regression line; it may not look quite so good, but it's actually a better fit (to the *whole* line)



Intonation of FIFTEEN3

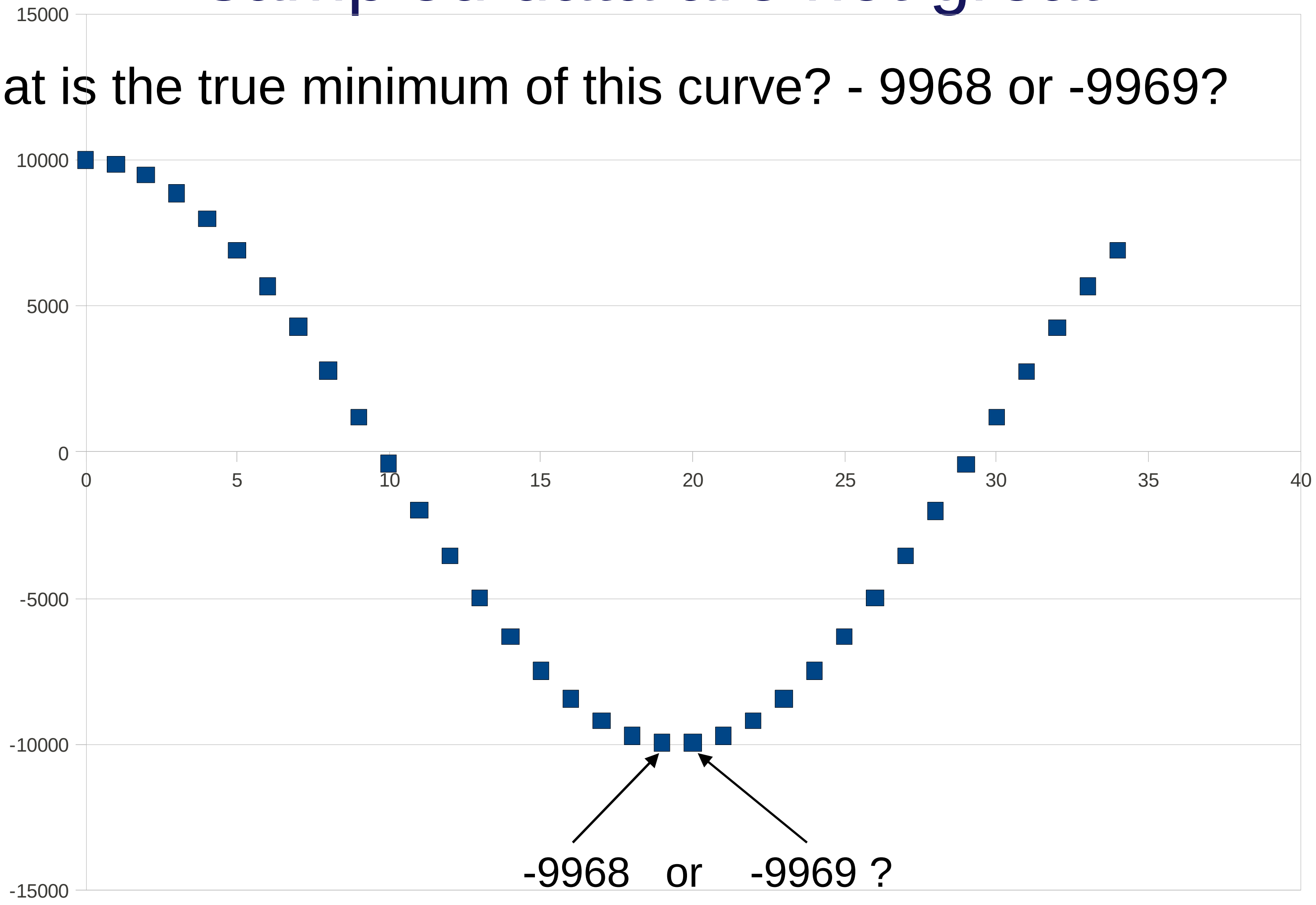
- A logarithmic regression curve fits better

Could we discover anything from the details of this equation?



Why single-point measurements of sampled data are not great

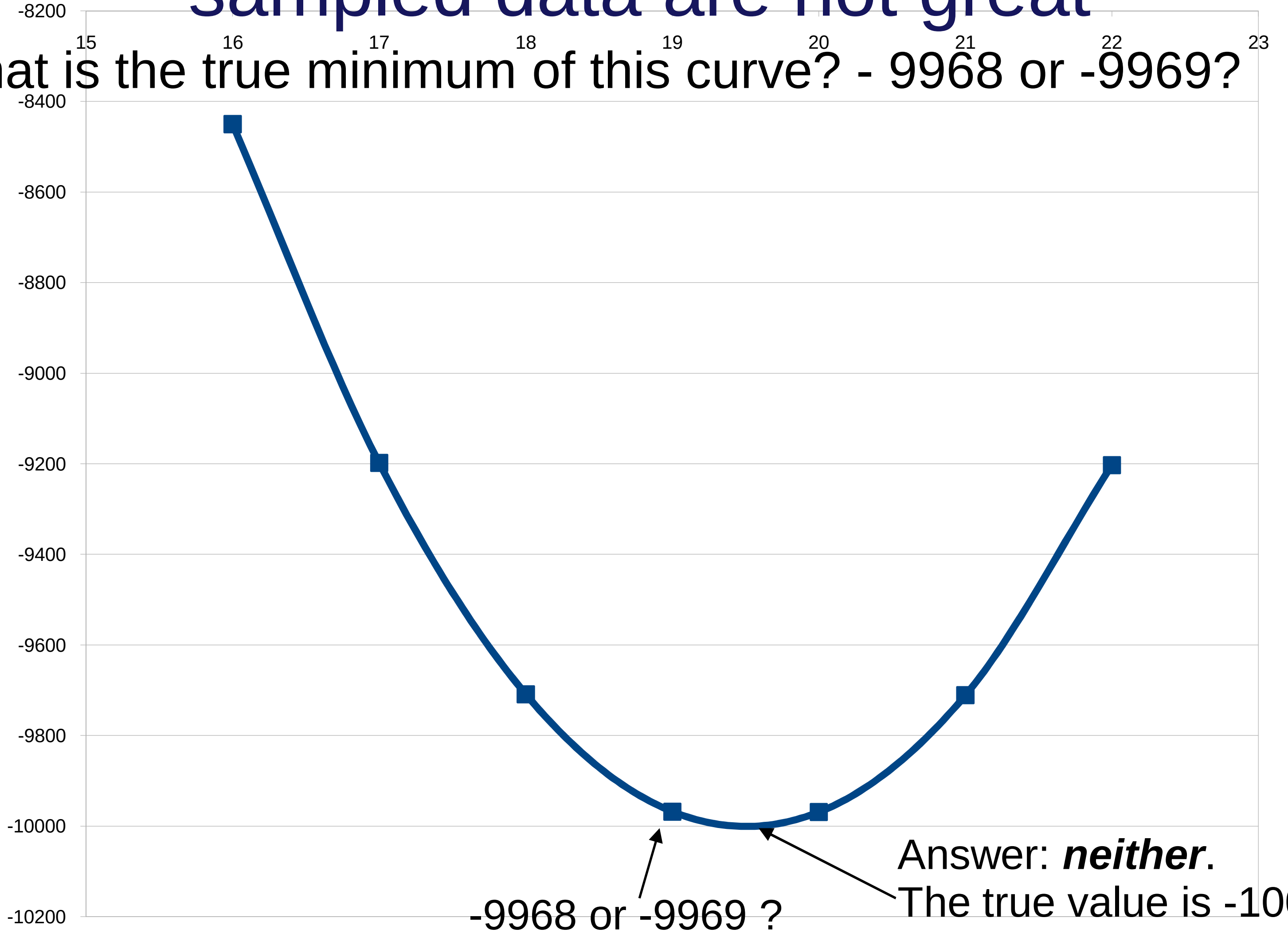
- What is the true minimum of this curve? - 9968 or -9969?



UNIVERSITY OF OXFORD

Why single-point measurements of sampled data are not great

What is the true minimum of this curve? - 9968 or -9969?



-9968 or -9969 ?

Answer: ***neither***.
The true value is -10000; it is a cosine function x10000

Functional Data Analysis

- Modelling sampled data using (continuous) functions

- General approach:

  – Possibly smooth the data a bit, to iron out irrelevant wiggles
  – Possibly normalize the data
  – Registration: some sort of time alignment of the individual tokens (not always necessary)

UNIVERSITY OF OXFORD

Functional Data Analysis

Choose a general kind of (basis) function that looks like your data

- For periodic data: Fourier series
- For nonperiodic data: B-splines
  sometimes: Orthogonal Polynomials (Example 1)
- *others are possible*

- Probability density functions
- E.g., for normally-distributed data: Gaussians (Example 2)

Fit the function to the data
i.e. find the parameters of the function that minimizes the differences between the function and the data

Orthogonal polynomials in Octave/Matlab

- Put numeric data into Matlab's vector notation:

```
f0 = load('FIFTEEN3.wav.f0.csv');
y = f0(:,1);
y = y(y>0);
```

```
y =
[176.894;
174.434;
167.352;
162.623;
160.734;
...
88.6733];
```

- Normalize it: `yn = y/mean(y) -1;`

- Normalize the time dimension to the interval [-1 1], and turn it into a column vector:

```
x = ((1:length(yn))-length(yn)/2)/(length(yn)/2);
x = x';
```

Orthogonal polynomials in Octave/Matlab

• Fit the normalized data to a polynomial (e.g. a cubic)

$$y = a_1x^3 \quad + \quad a_2x^2 \quad + \quad a_3x \quad + \quad a_4$$

```
[a,S] = polyfit(x,yn,3);
```

Output values: a = 0.19321 0.63340 -0.70280 -0.19866

•The fitted function is given by `fit = getfield(S,'yf');`
and restored to the original units (e.g. Hz)

```
ysynth = mean(y)*(fit+1);
```

Orthogonal polynomials in Octave/Matlab

- Fit the normalized data to a polynomial (e.g. a cubic)

$$y = a_1 x^3 + a_2 x^2 + a_3 x + a_4$$

average height

```
[a,S] = polyfit(x,yn,3);
```

Output values: a = 0.19321 0.63340 -0.70280 -0.19866

Orthogonal polynomials in Octave/Matlab

- Fit the normalized data to a polynomial (e.g. a cubic)

$$y = a_1x^3 \quad + \quad a_2x^2 \quad + \quad a_3x \quad + \quad a_4$$

slope (steepness anddirection)

average height

Output values: a = 0.19321 0.63340 -0.70280 -0.19866

Orthogonal polynomials in Octave/Matlab

- Fit the normalized data to a polynomial (e.g. a cubic)

$$y = a_1x^3 + a_2x^2 + a_3x + a_4$$

a_2x^2	a_3x	a_4
breadth of curvature	slope (steepness anddirection)	average height

Output values: a = 0.19321 0.63340 -0.70280 -0.19866

Orthogonal polynomials in Octave/Matlab

• Fit the normalized data to a polynomial (e.g. a cubic)

$$y \;=\; a_1x^3 \quad + \quad a_2x^2 \quad + \quad a_3x \quad + \quad a_4$$

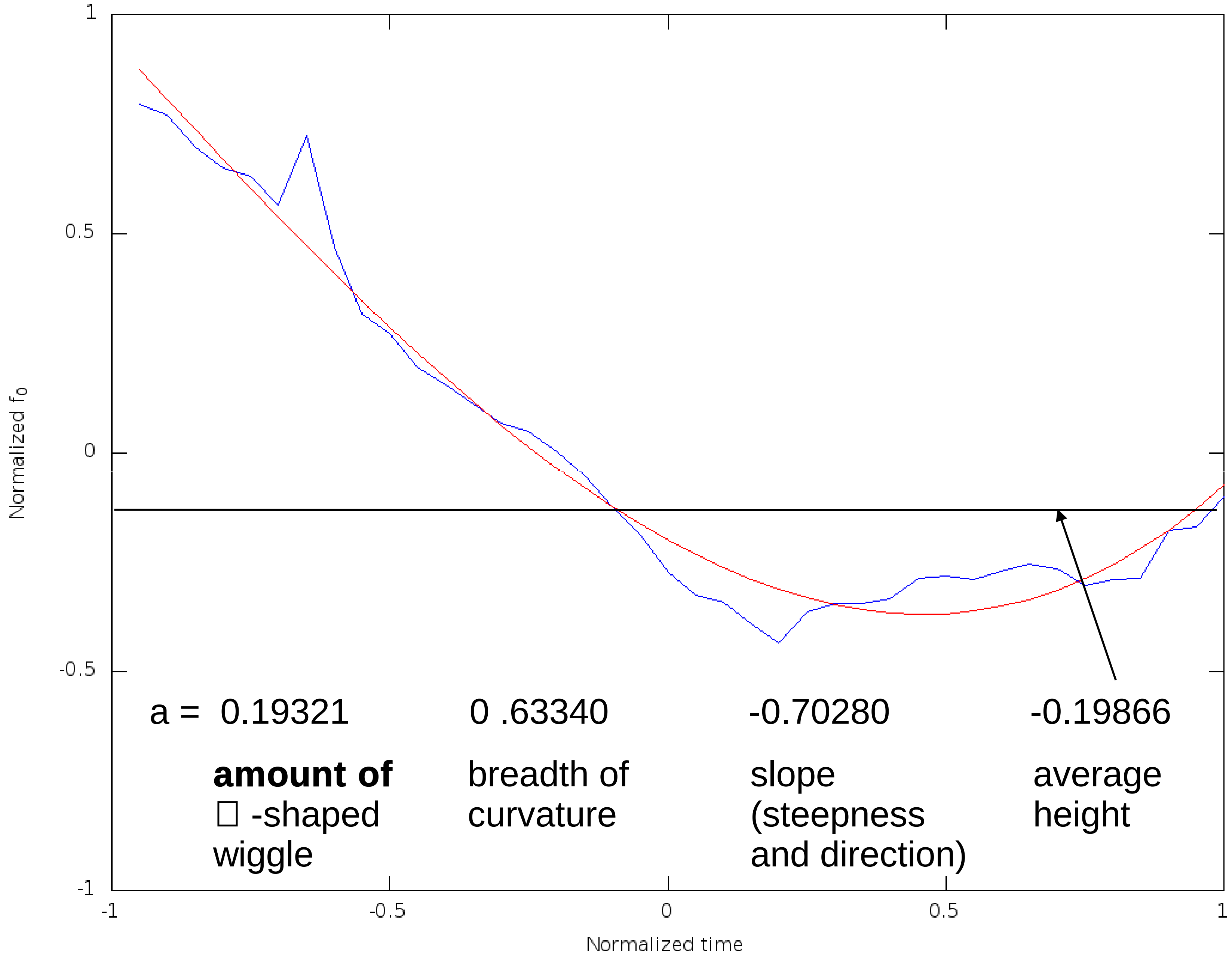
amount of □ -shaped wiggle	breadth of curvature	slope (steepness and direction)	average height

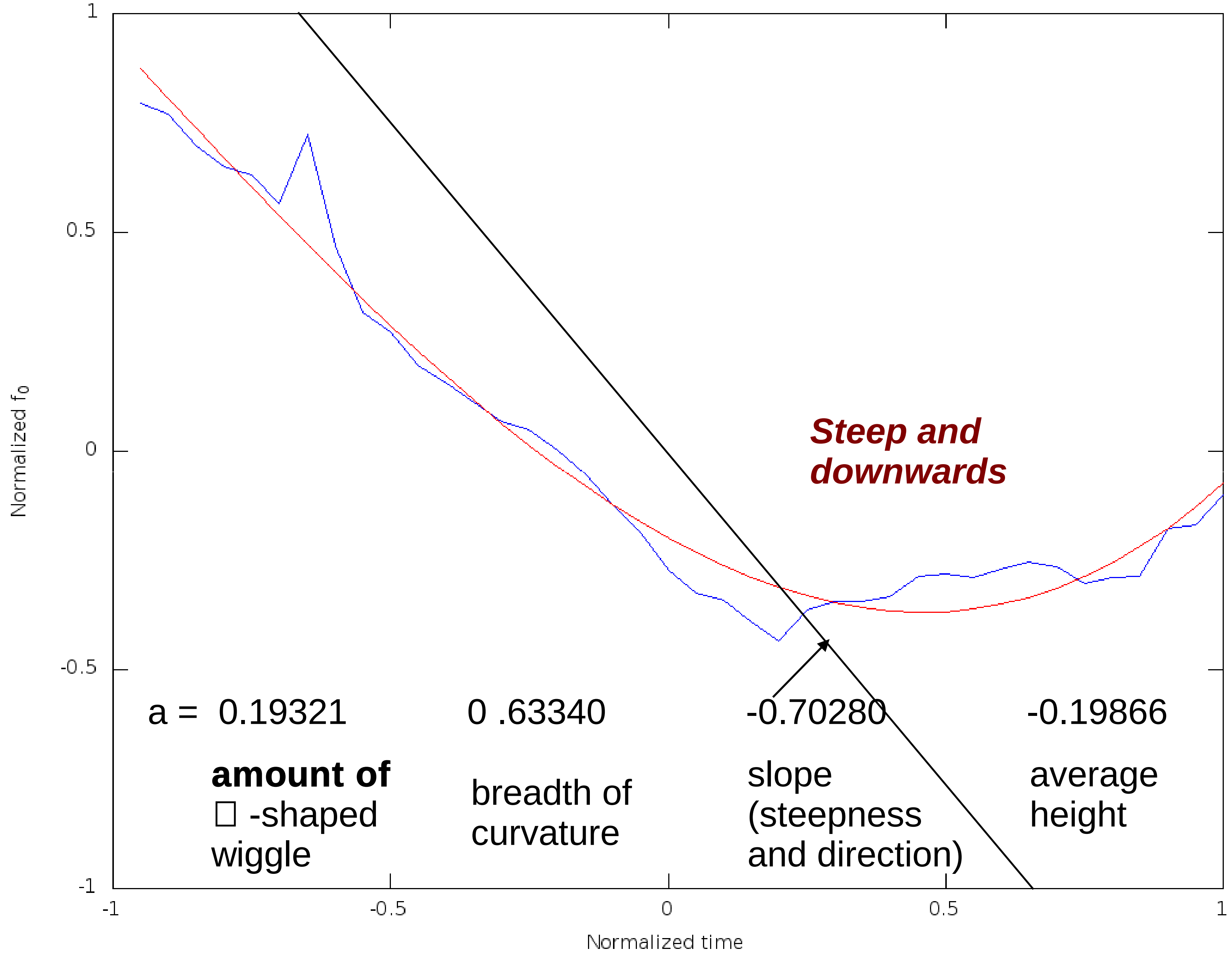
Output values: a = 0.19321 0.63340 -0.70280 -0.19866

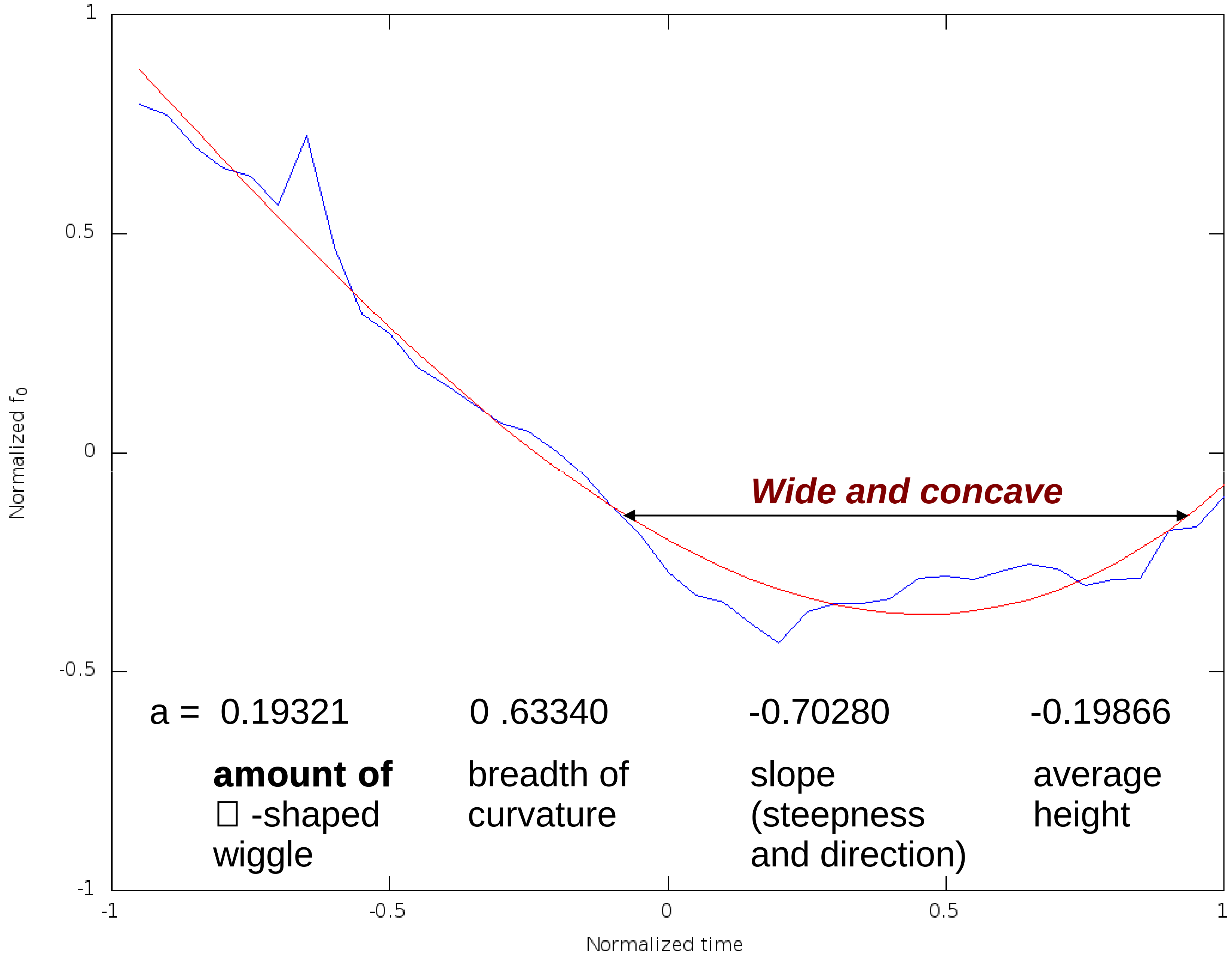
•The fitted function is given by `fit = getfield(S,'yf');`
and can be restored to the original units (e.g. Hz) by

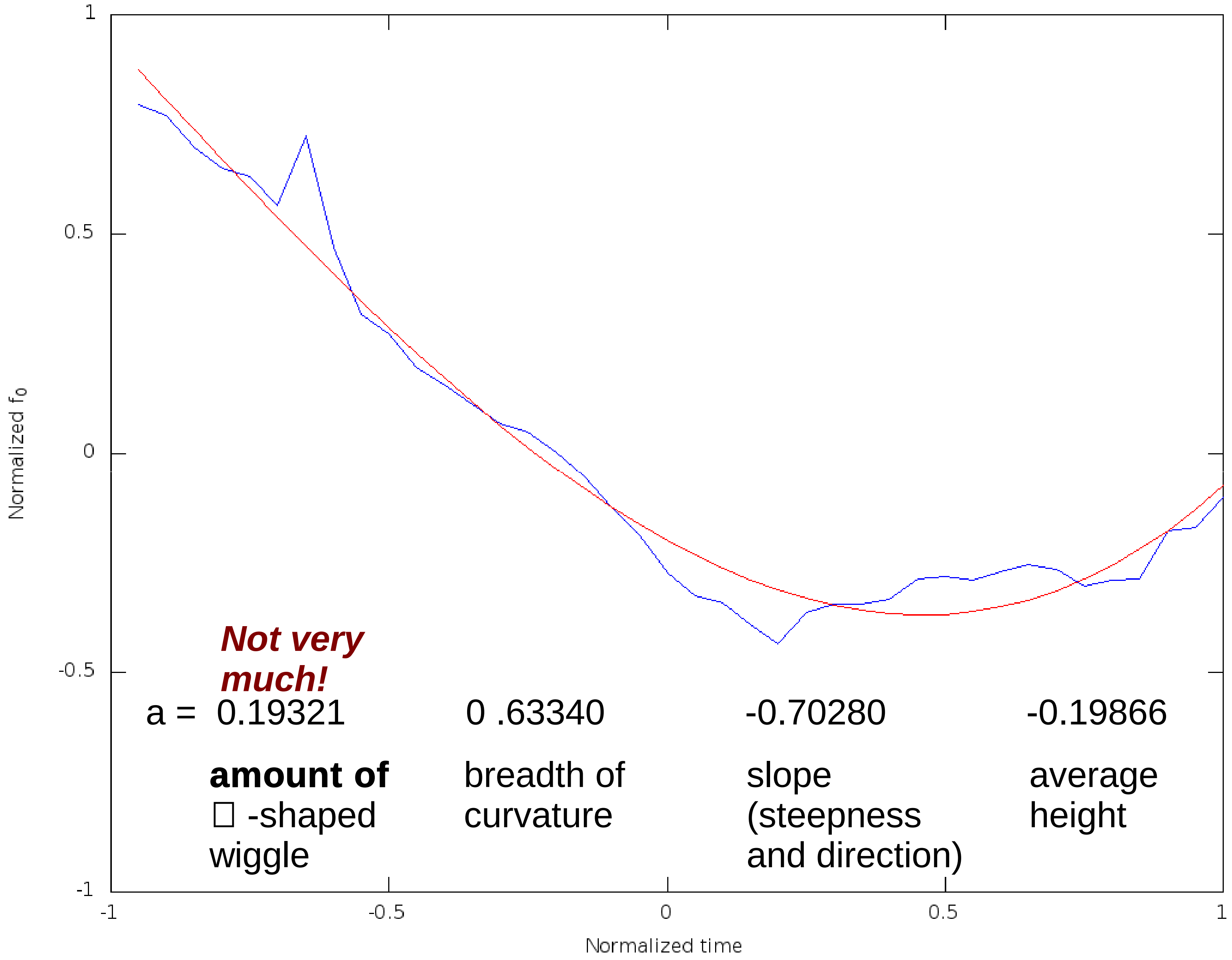
```
ysynth = mean(y)*(fit+1);
```

UNIVERSITY OF OXFORD


a = 0.19321
0 .63340
-0.70280
-0.19866
amount of □ -shaped wiggle
breadth of curvature
slope (steepness and direction)
average height
Normalized f₀
Normalized time
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1
Normalized f0
Normalized time
Steep and downwards
a = 0.19321
0 .63340
-0.70280
-0.19866
amount of □ -shaped wiggle
breadth of curvature
slope (steepness and direction)
average height

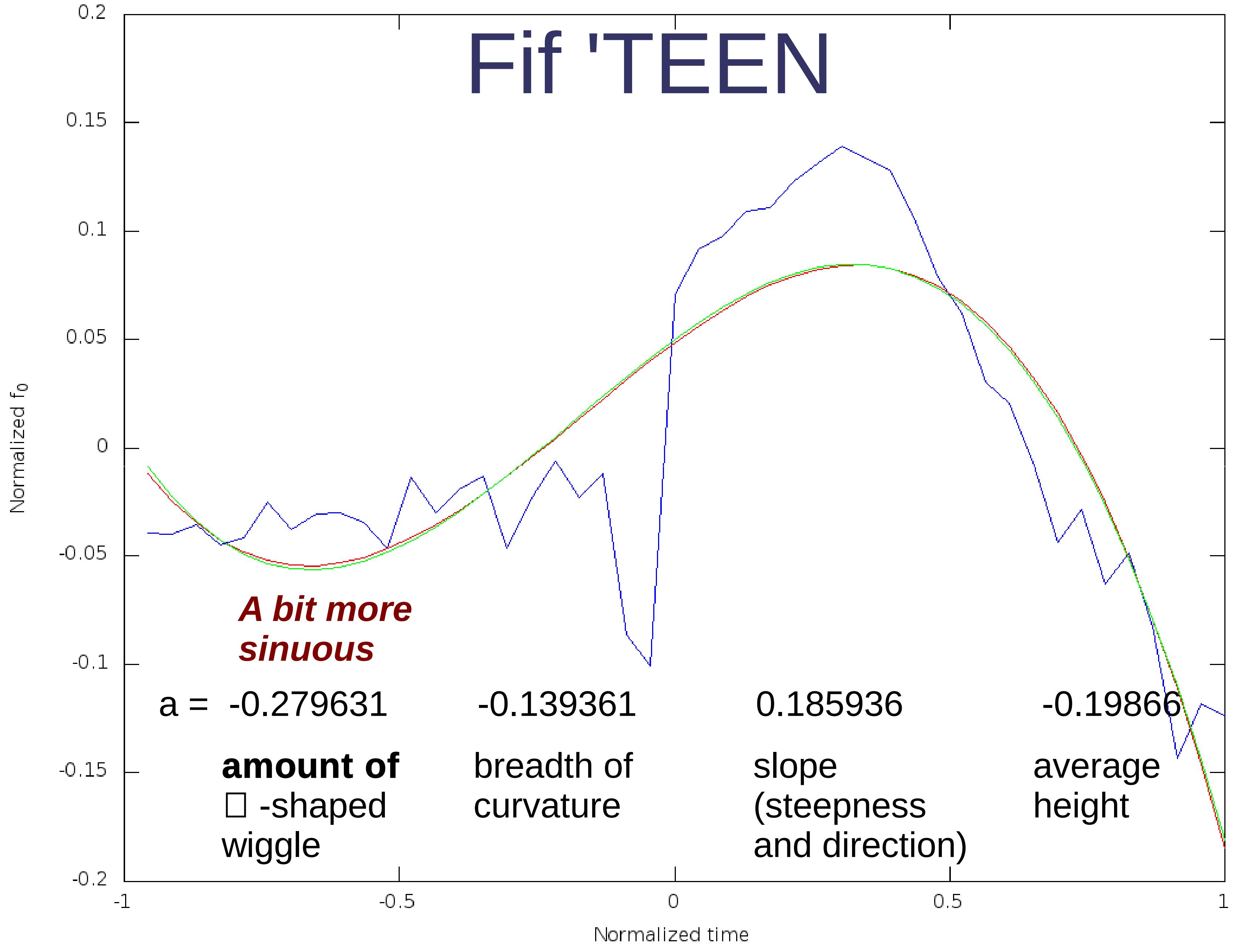

Wide and concave
a = 0.19321
0 .63340
-0.70280
-0.19866
amount of
□ -shaped
wiggle
breadth of
curvature
slope
(steepness
and direction)
average
height
Normalized f_0
Normalized time
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1



Not very much!

a =	0.19321	0 .63340	-0.70280	-0.19866
	amount of □ -shaped wiggle	breadth of curvature	slope (steepness and direction)	average height

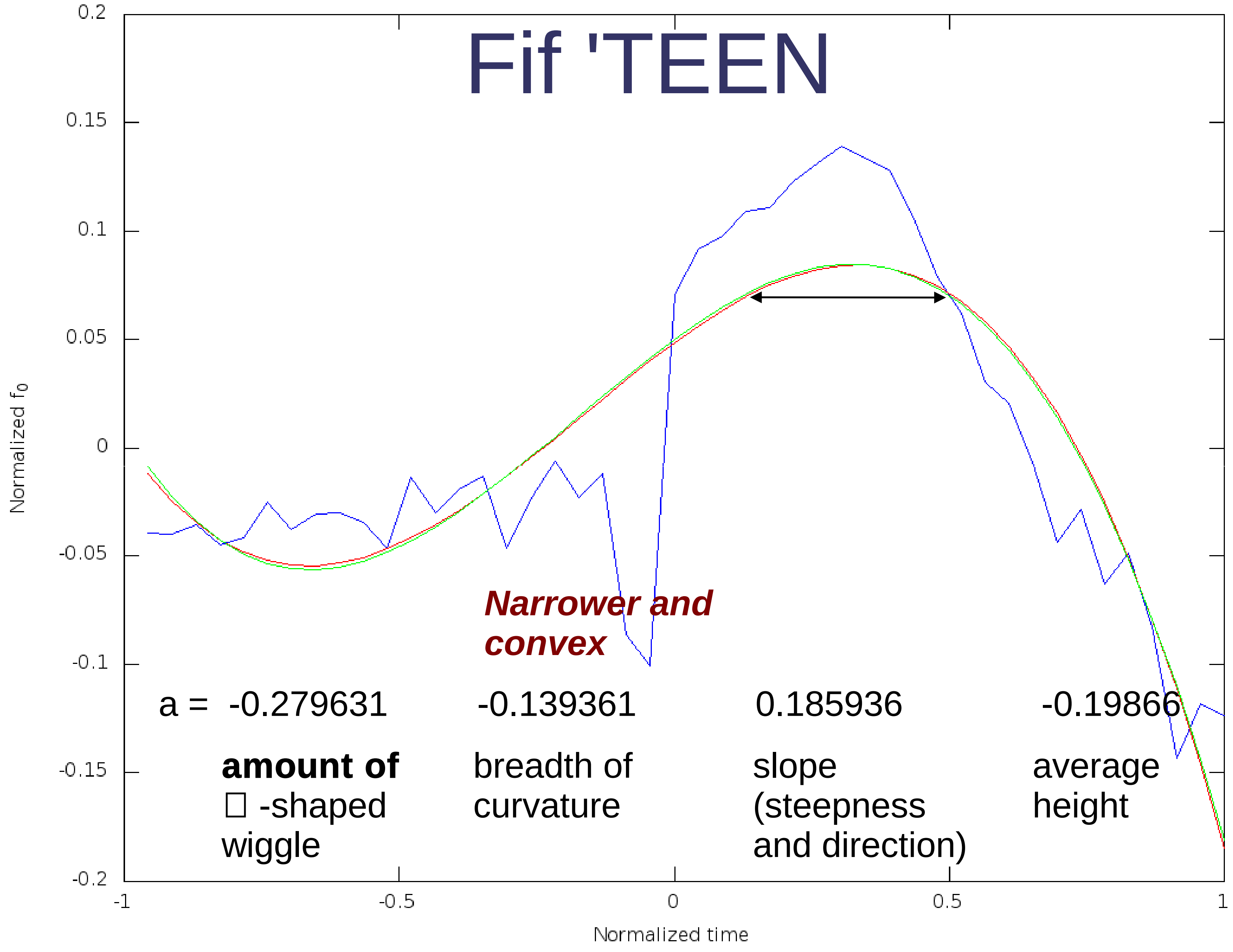
Fif 'TEEN



A bit more sinuous

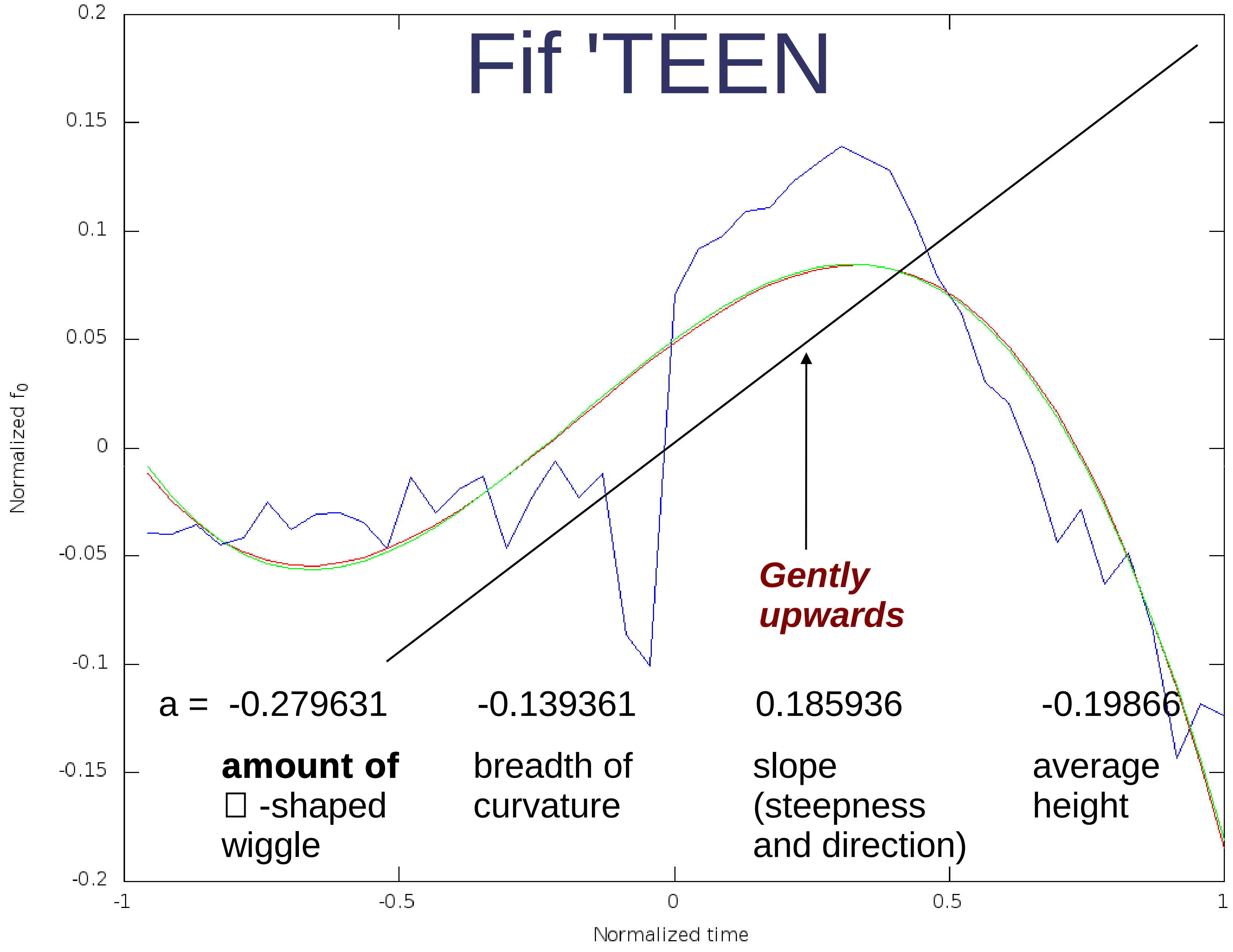
a =	-0.279631	-0.139361	0.185936	-0.19866
	amount of □ -shaped wiggle	breadth of curvature	slope (steepness and direction)	average height

Fif 'TEEN

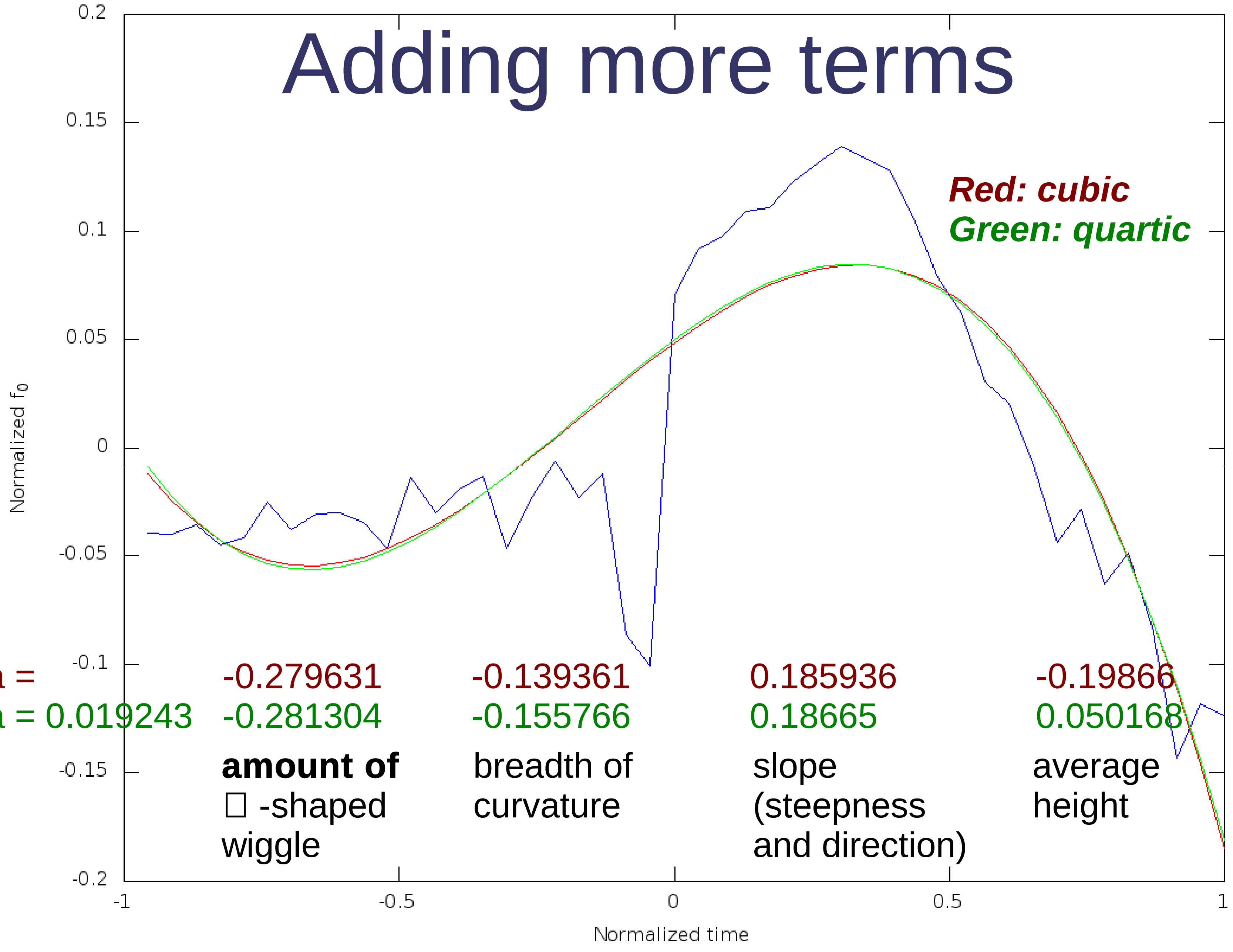


Narrower and convex

a = -0.279631	-0.139361	0.185936	-0.19866
amount of □ -shaped wiggle	breadth of curvature	slope (steepness and direction)	average height


Fif 'TEEN
Gently
upwards
a = -0.279631
-0.139361
0.185936
-0.19866
amount of □ -shaped wiggle
breadth of curvature
slope (steepness and direction)
average height
Normalized f0
Normalized time
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-1
-0.5
0
0.5
1

Adding more terms



Red: cubic

Green: quartic

a =	-0.279631	-0.139361	0.185936	-0.19866
a = 0.019243	-0.281304	-0.155766	0.18665	0.050168
amount of □ -shaped wiggle	breadth of curvature	slope (steepness and direction)	average height	

Orthogonalisation

- Translate polynomial coeffients into *orthogonal* (Legendre) polynomial coeffs:

```
c = [0.4*a(1) 2*a(2)/3 a(3)+6*a(1)/5 a(4)]
```

```
%% a =  0.19321     0.63340    -0.70280    -0.19866
%% c =  0.077284   0.422269  -0.470948  -0.198659
```

UNIVERSITY OF OXFORD

Loop over all the “good” files

```
for i = 2:172
   eval(['fid = fopen(''FIFTEEN',int2str(i),'.wav.f0.csv'');']);
   if (fid ~= -1)                                              %% checks file FIFTEEN i ... exists
      eval(['f0 = load(''FIFTEEN',int2str(i),'.wav.f0.csv'');']);
      y = f0(:,1);
      y = y(y>0);
      yn = y/mean(y)-1;
      x = ((1:length(yn))-length(yn)/2)/(length(yn)/2);
      x = x';
      [a,S] = polyfit(x,yn,3);
      c = [0.4*a(1) 2*a(2)/3 a(3)+6*a(1)/5 a(4)];
      C(i,:) = [i c];
   end
end
save('coeffs.csv','C');
```

Functional data analysis in R

`z-score-f0-poly-BAAP.R`

- Performs z-score transformation of f0 data
- Fits 2nd-order polynomial to each file
    - Intercept
    - Linear (slope) term
    - Quadratic term
    - Adjusted R^2 for curve fit

Script output: `fit2.coeff.f0` object & .csv file

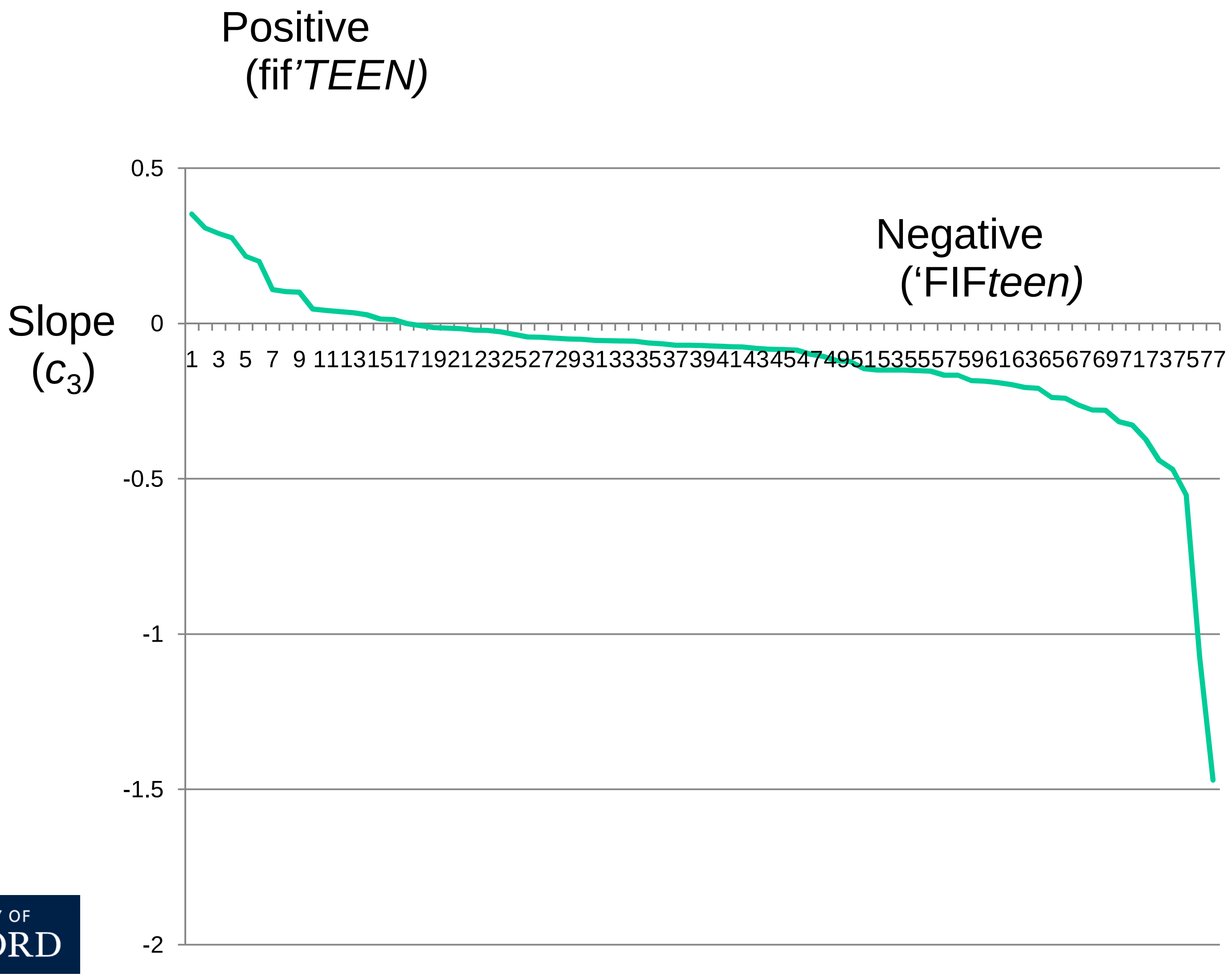
Now do your statistics

- Rather than applying statistical tests to the raw data, examine the means, variances etc of the coefficients of the functions you're using to model the data

coeffs.csv

			Slope	
0	0	0	0	0
2	0.094591528	-0.090229162	-0.022355083	0.0477172
3	0.07728441	0.422268933	-0.470948363	-0.19865862
0	0	0	0	0
5	-0.237378144	-0.065854502	-0.027159549	0.029072195
0	0	0	0	0
7	-0.034872629	-0.072306628	-0.152050826	0.041887784
0	0	0	0	0
9	-0.002626976	-0.181630596	-0.238291819	0.103118291
0	0	0	0	0
11	0.023445549	0.020728149	-0.119549338	-0.00534581
0	0	0	0	0
13	-0.303372913	0.364972835	-0.373086886	-0.172771143
0	0	0	0	0
0	0	0	0	0
16	-0.01144175	-0.002436757	-0.05645396	0.003428771
17	0.110621474	0.167832411	0.289525511	-0.097446139
18	0.184731822	0.707875524	-1.074013742	-0.283804173

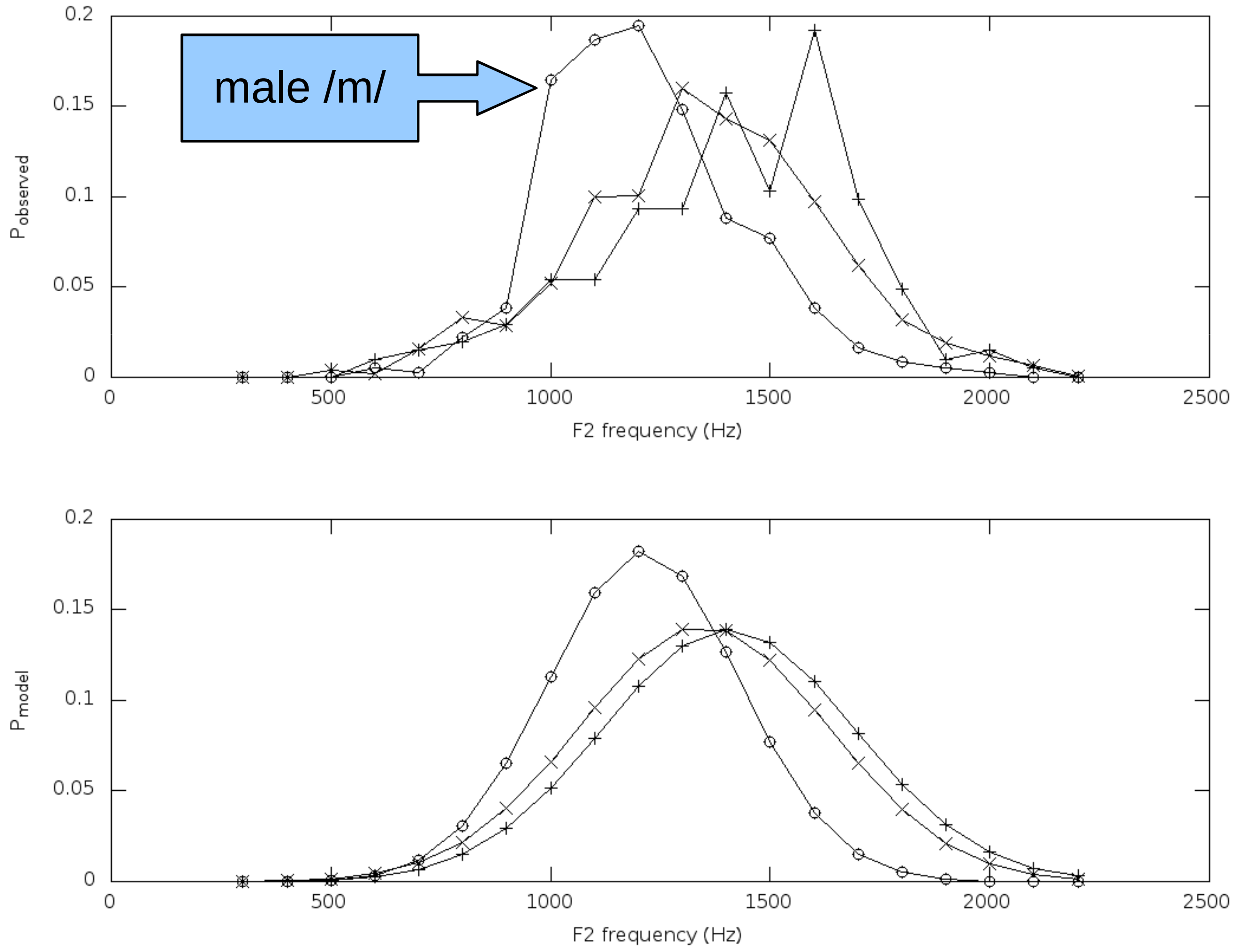
coeffs.csv



UNIVERSITY OF OXFORD

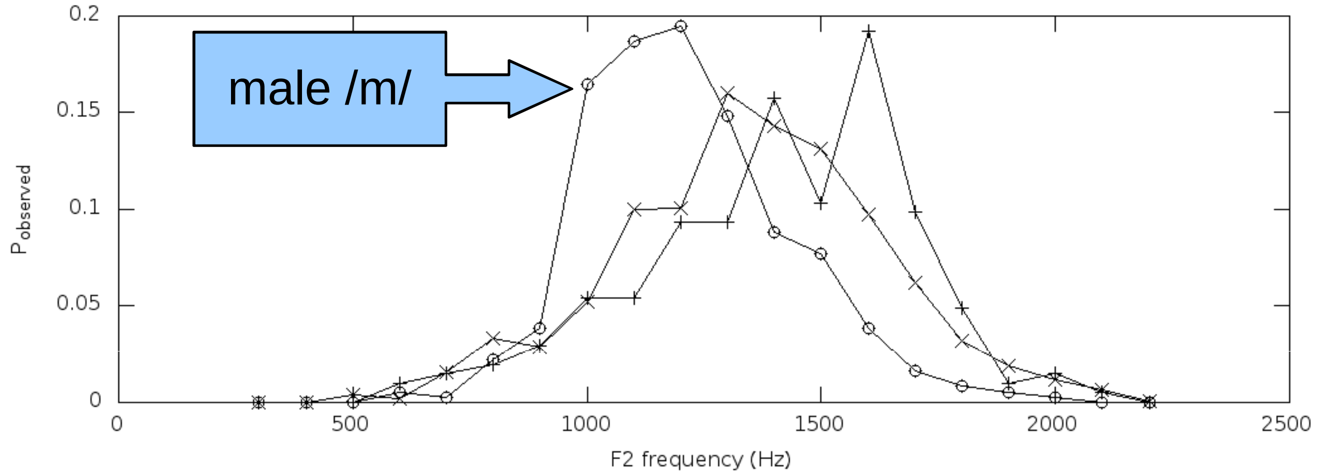
Example 2

- Fitting a probability density functions to histograms



Example 2

- Fitting a probability density functions to histograms

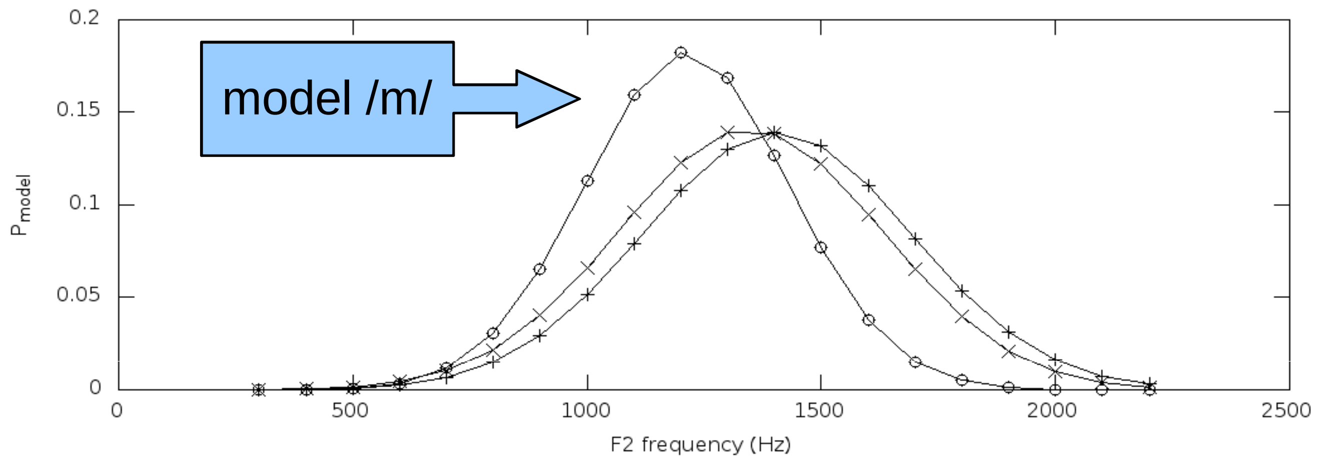


In Matlab (or Octave):

```
f = [300 400 500 600 700 800 900 1000 ... 2200]';
m_hist = [0 0 0 2 1 8 14 60 68 71 54 32 28 14 6 3 2 1 0 0]';
mu_m = sum(m_hist .* f)/sum(m_hist);
sigma_m = sqrt(sum(m_hist .* ((f - mu_m).^2))/(sum(m_hist)-1));
```

Example 2

- Fitting a probability density functions to histograms



In Matlab (or Octave):

```
X_m = [normpdf(f,mu_m,sigma_m)];
a_m = X_m\m_hist;
model_m = a_m*normpdf(f,mu_m,sigma_m);
```

Conclusions

• We need bigger datasets because of (a) Zipf's law and (b) variability in the population

• To scale up for big data, every second saved is a bonus

• To be feasible, automate the analyses as much as possible

• Forced alignment is a “killer app” for finding items in corpora

• Human listening is good for checking whether the automatic selection of data is basically sound

Conclusions

- Otherwise, minimize human processing; learn to script

- Avoid manual selection of segments and painstaking of single points in the data

- Try to analyse properties of the whole curve

- Fit functions to the data, and then do your stats on the coefficients of the fitted functions

UNIVERSITY OF OXFORD