

15-826: Multimedia (Databases) and Data Mining

Lecture #22: DSP tools –
Wavelets
C. Faloutsos



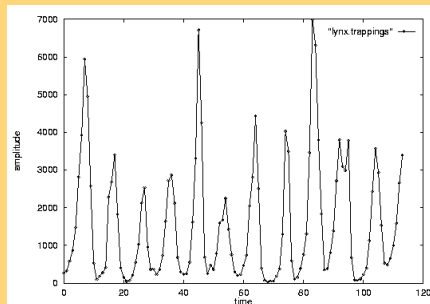
Problem

Goal: given a signal (eg., sales over time and/or space)

Q: Find patterns and/or compress



count



year



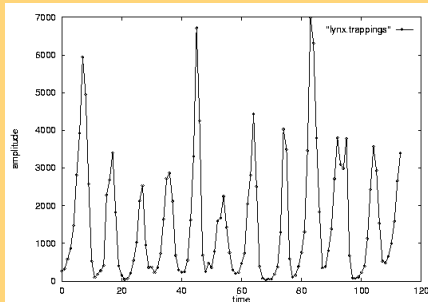
Solutions:

Goal: given a signal (eg., sales over time and/or space)

Q: Find patterns and/or compress

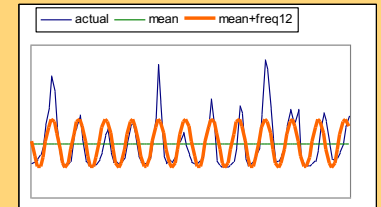


count

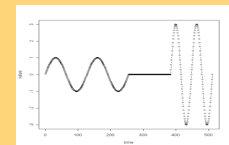
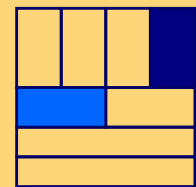


year

A1: Fourier (DFT)



A2: Wavelets (DWT)



Must-read Material

- Wavelets: In [PTVF](#) ch. 13.10; in [MM Textbook](#) Appendix C



Outline

Goal: ‘Find **similar / interesting** things’

- Intro to DB
- ➡ • Indexing - similarity search
- ➡ • Data Mining



Indexing - Detailed outline

- primary key indexing
- ..
- Multimedia –
 - Digital Signal Processing (DSP) tools
 - Discrete Fourier Transform (DFT)
 - Discrete Wavelet Transform (DWT)





DWT - Detailed outline

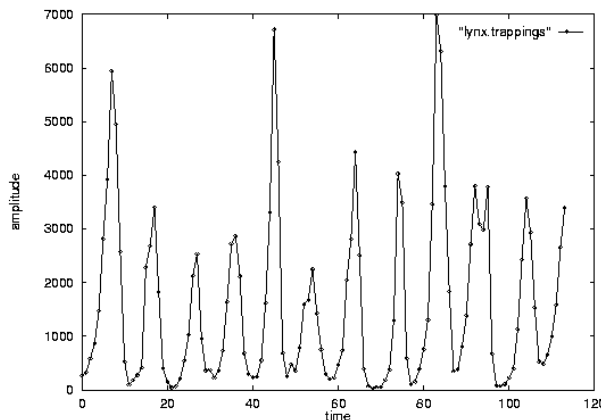
- ➔ • Motivation
- Definition - Intuition
- Examples and drills
- Application: BGPlens
- 2-d DWT

Reminder: Problem:

Goal: given a signal (eg., #packets over time)

Find: patterns, periodicities, and/or compress

count

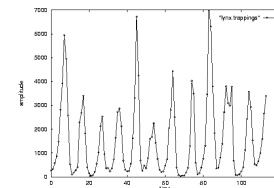


lynx caught per year
(packets per day;
virus infections per month)

year



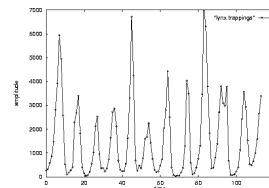
Important observations



Patterns, rules, forecasting and similarity indexing are closely related:

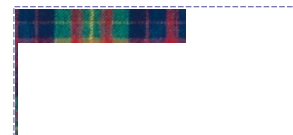


Important observations



Patterns, rules, forecasting and similarity indexing are closely related:

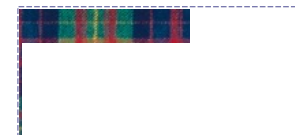
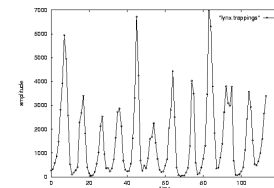
- To do **forecasting**, we need
 - to find **patterns**/rules
 - compress
 - to find **similar** settings in the past
- to find outliers, we need to have forecasts
 - (outlier = too far away from our forecast)





Important observations

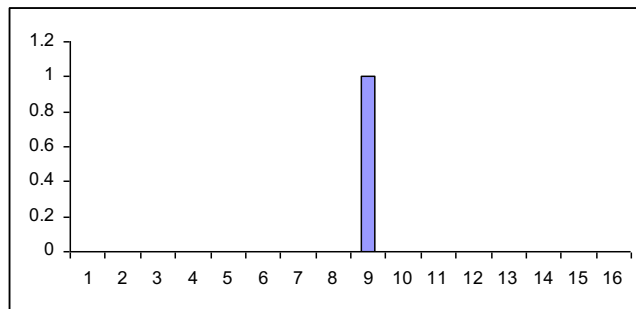
Patterns, rules, forecasting and similarity indexing are closely related:



Wavelets - DWT

- DFT is great - but, how about compressing a spike?

value

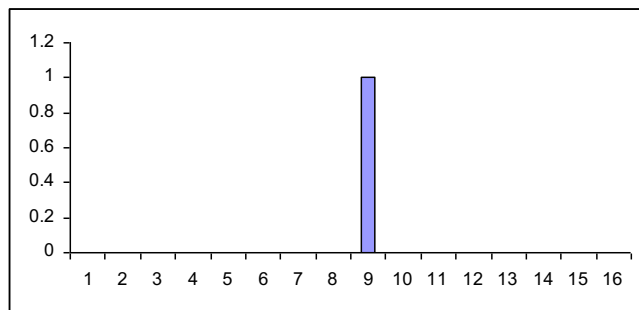


time

Wavelets - DWT

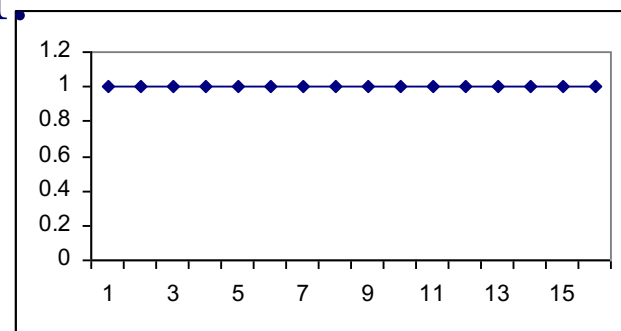
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

value



time

Ampl.

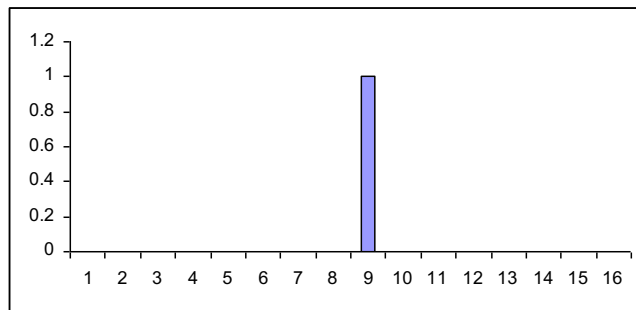


Freq. 13

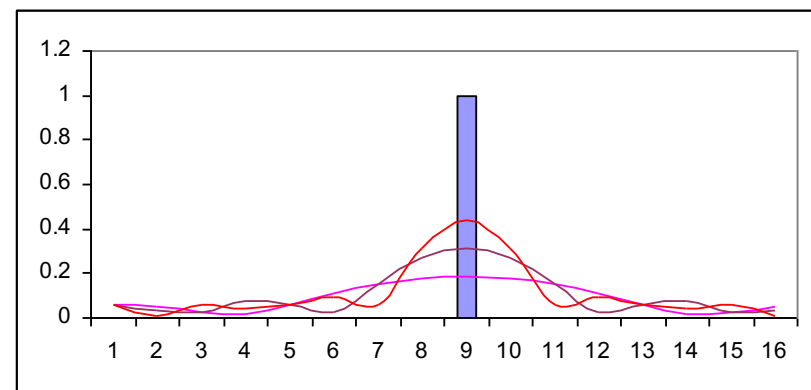
Wavelets - DWT

- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

value



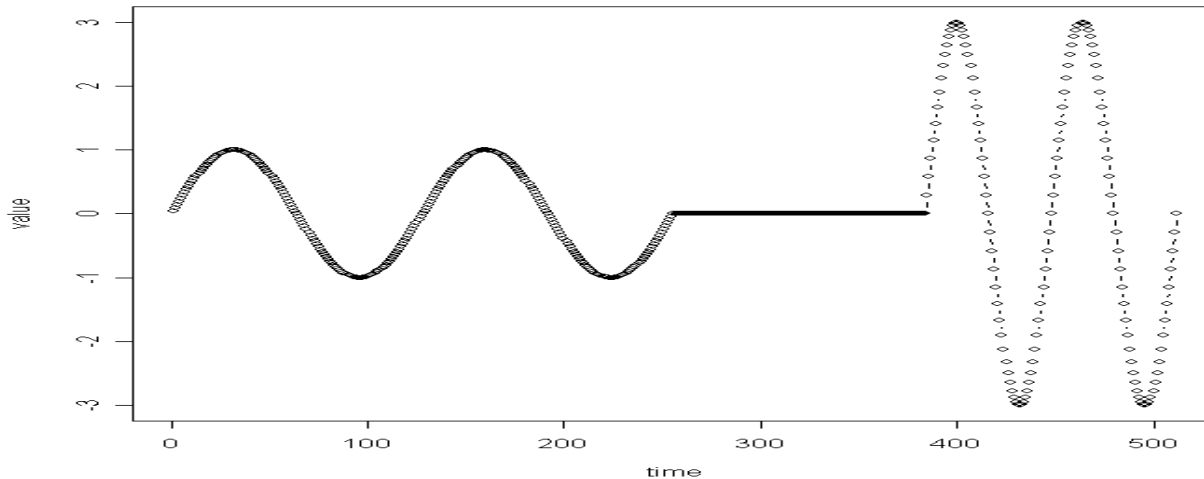
time



Wavelets - DWT

- Similarly, DFT suffers on short-duration waves (eg., baritone, silence, soprano)

value

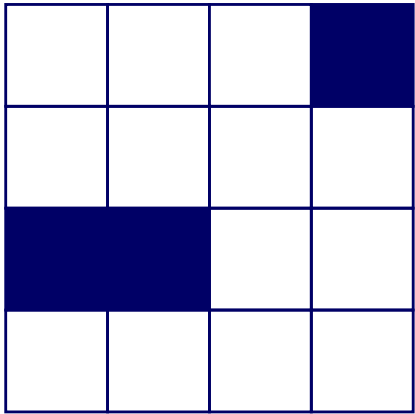
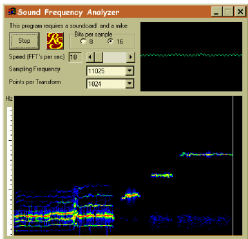


time

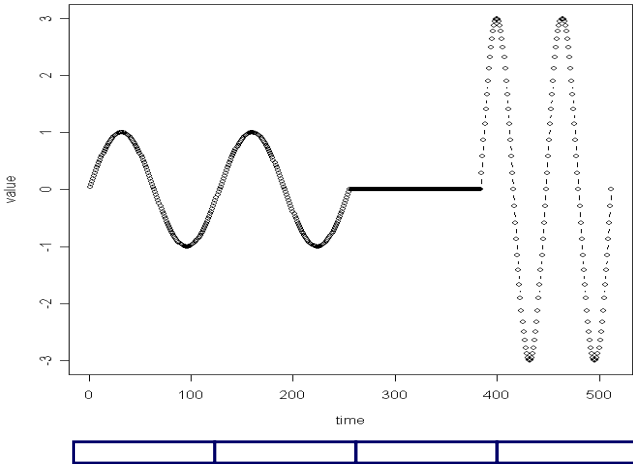
Wavelets - DWT

- Solution#1: Short window Fourier transform (SWFT)
- But: how short should be the window?

freq



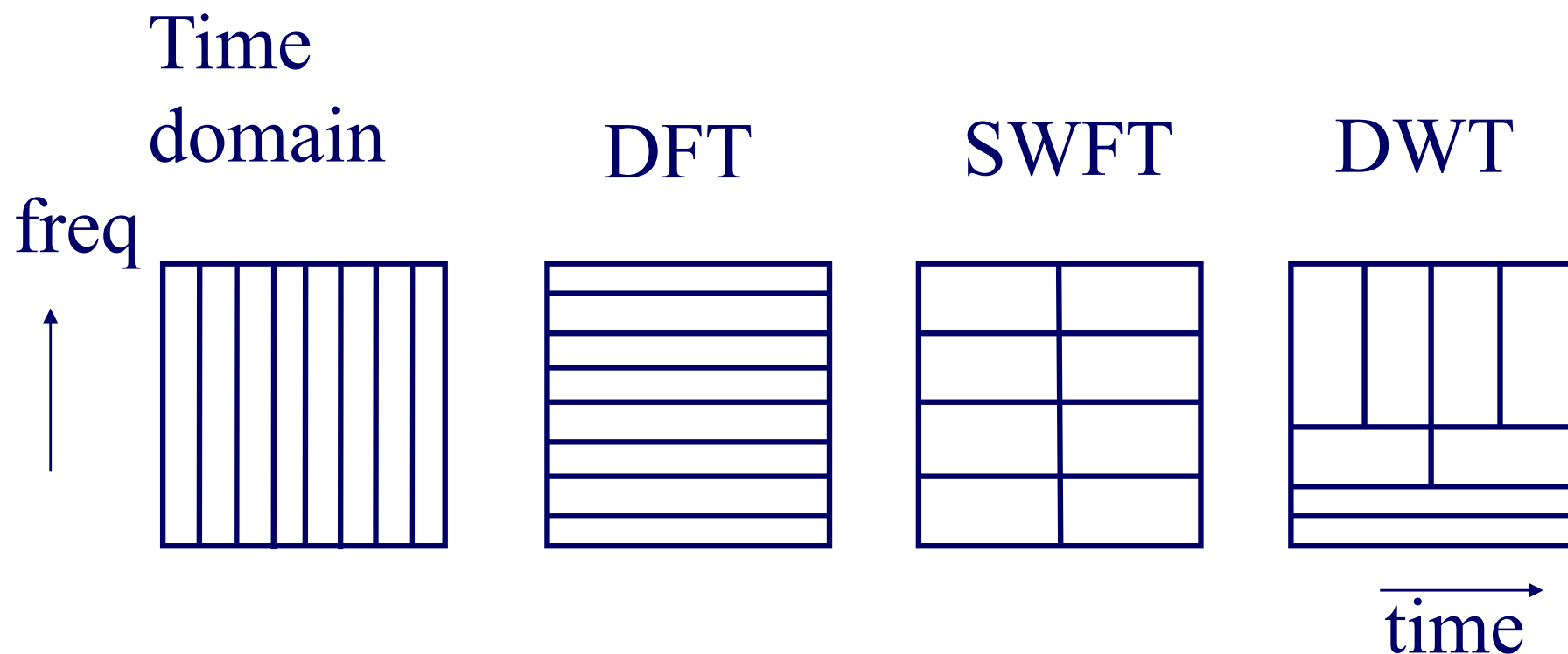
value



time

Wavelets - DWT

- Answer: **multiple** window sizes! -> DWT



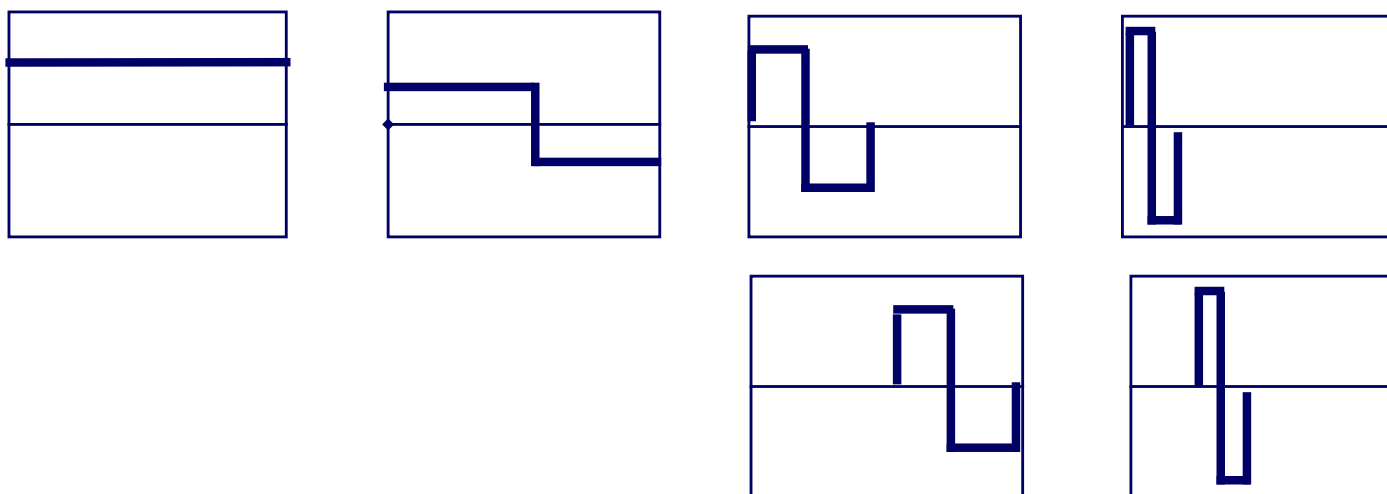


DWT - Detailed outline

- Motivation
- ➔ • Definition - Intuition
- Examples and drills
- Application: BGPlens
- 2-d DWT

Haar Wavelets

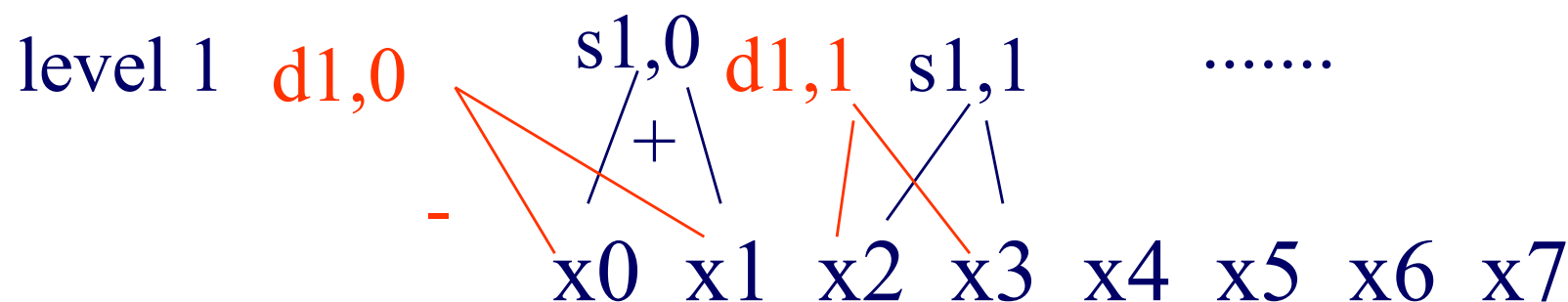
- subtract sum of left half from right half
- repeat recursively for quarters, eight-ths, ...



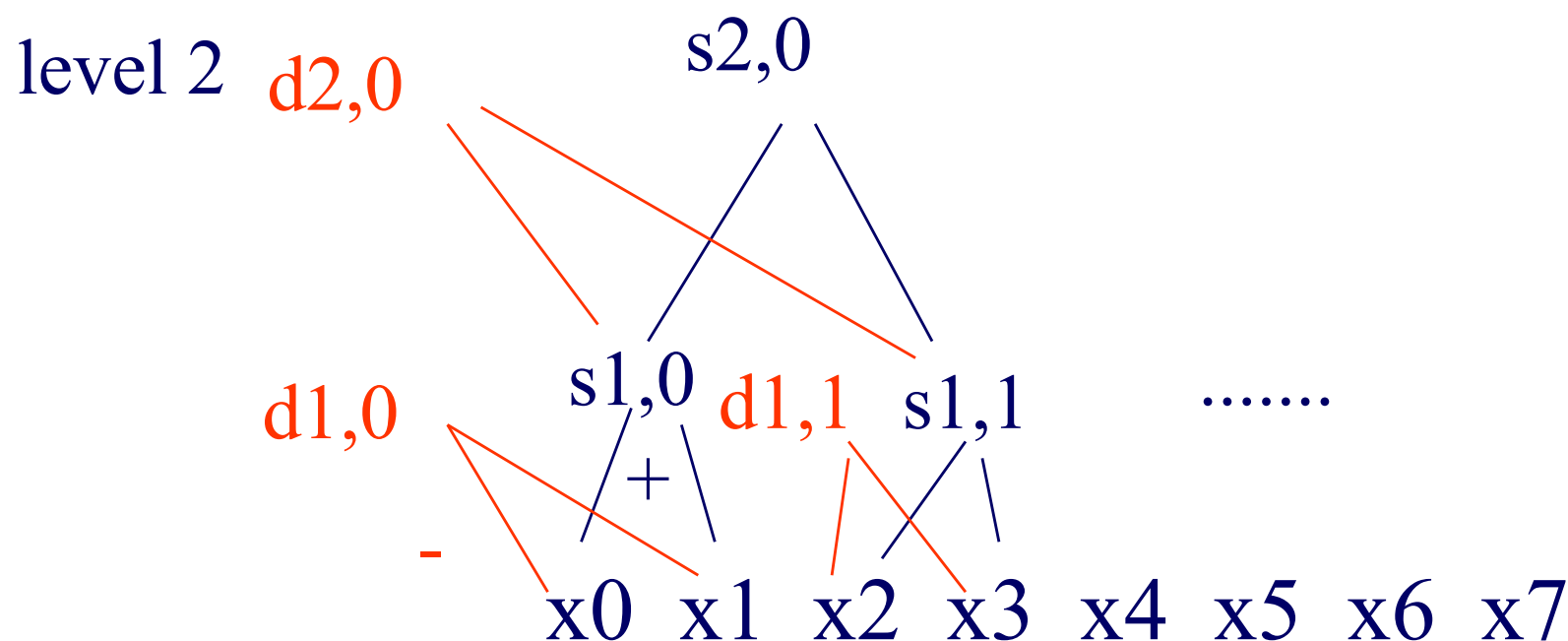
Wavelets - construction

$x_0 \ x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7$

Wavelets - construction

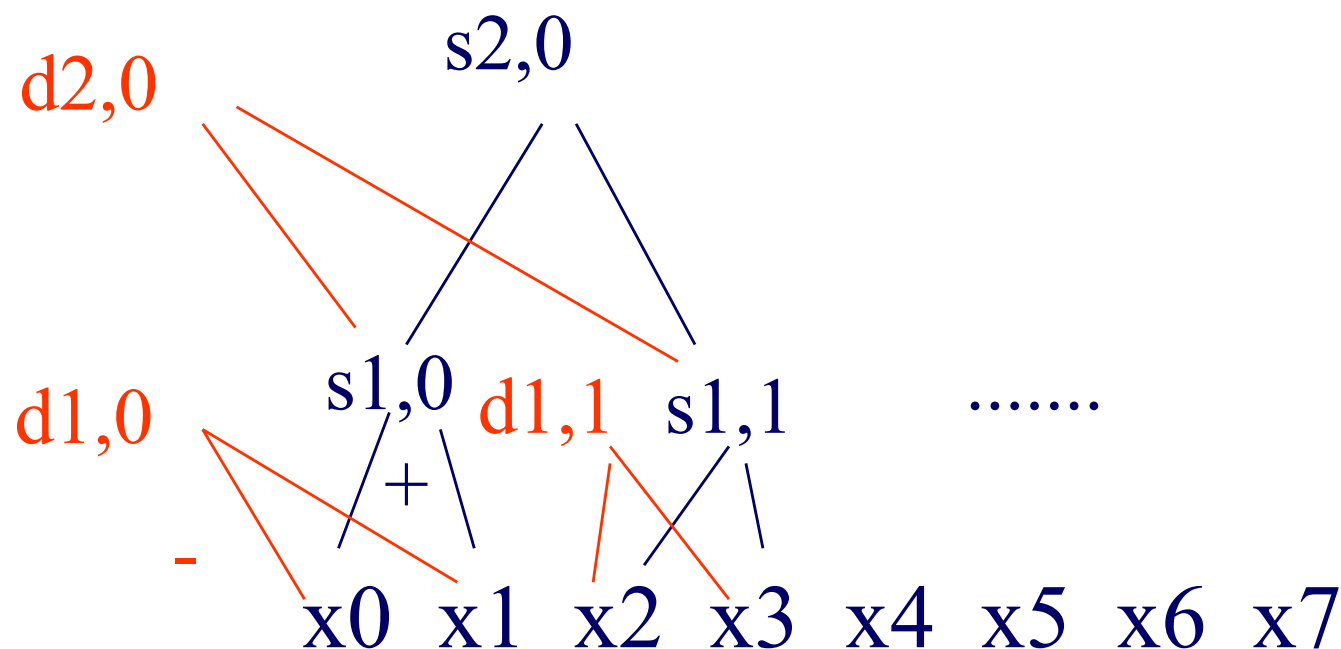


Wavelets - construction



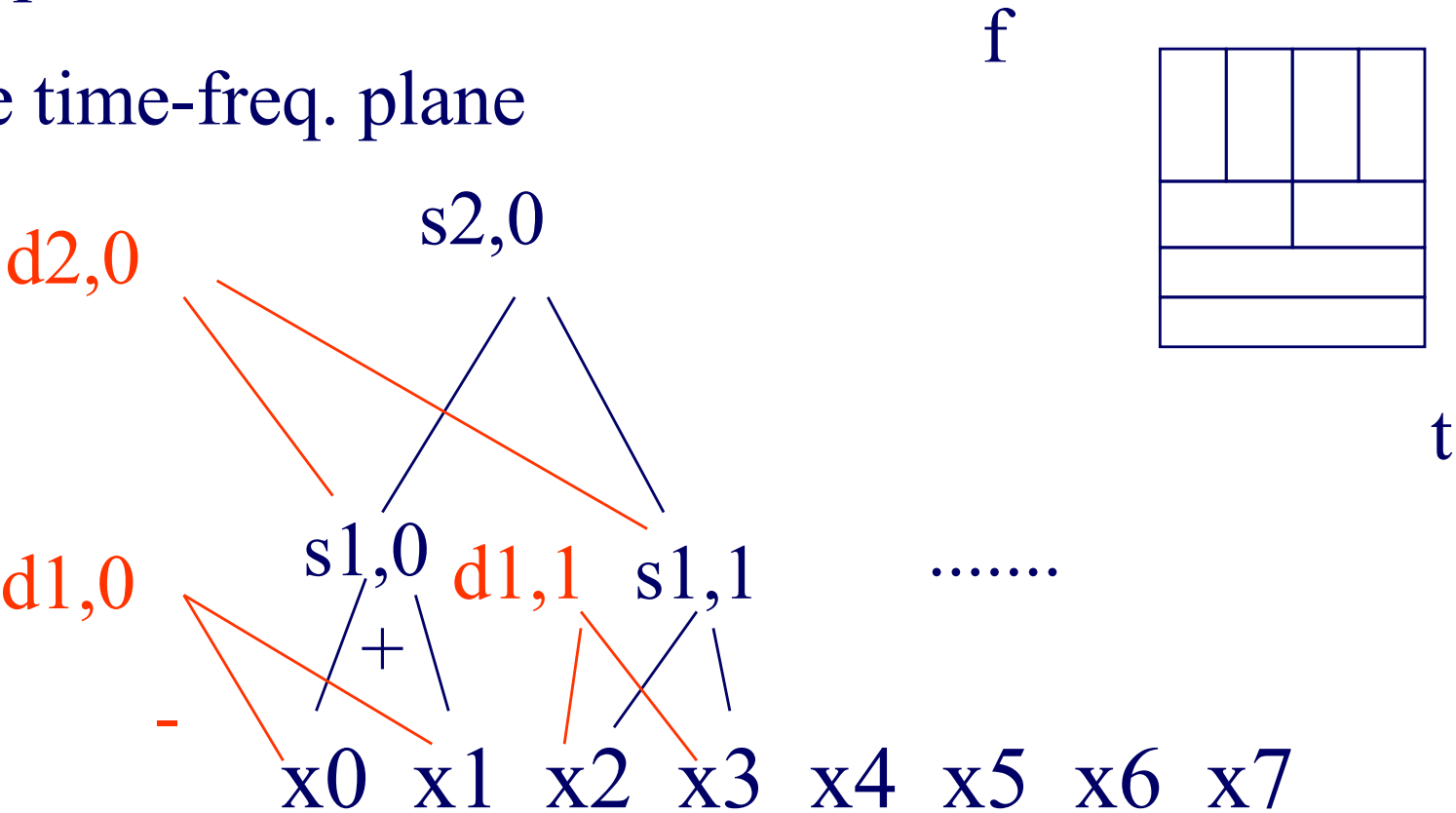
Wavelets - construction

etc ...



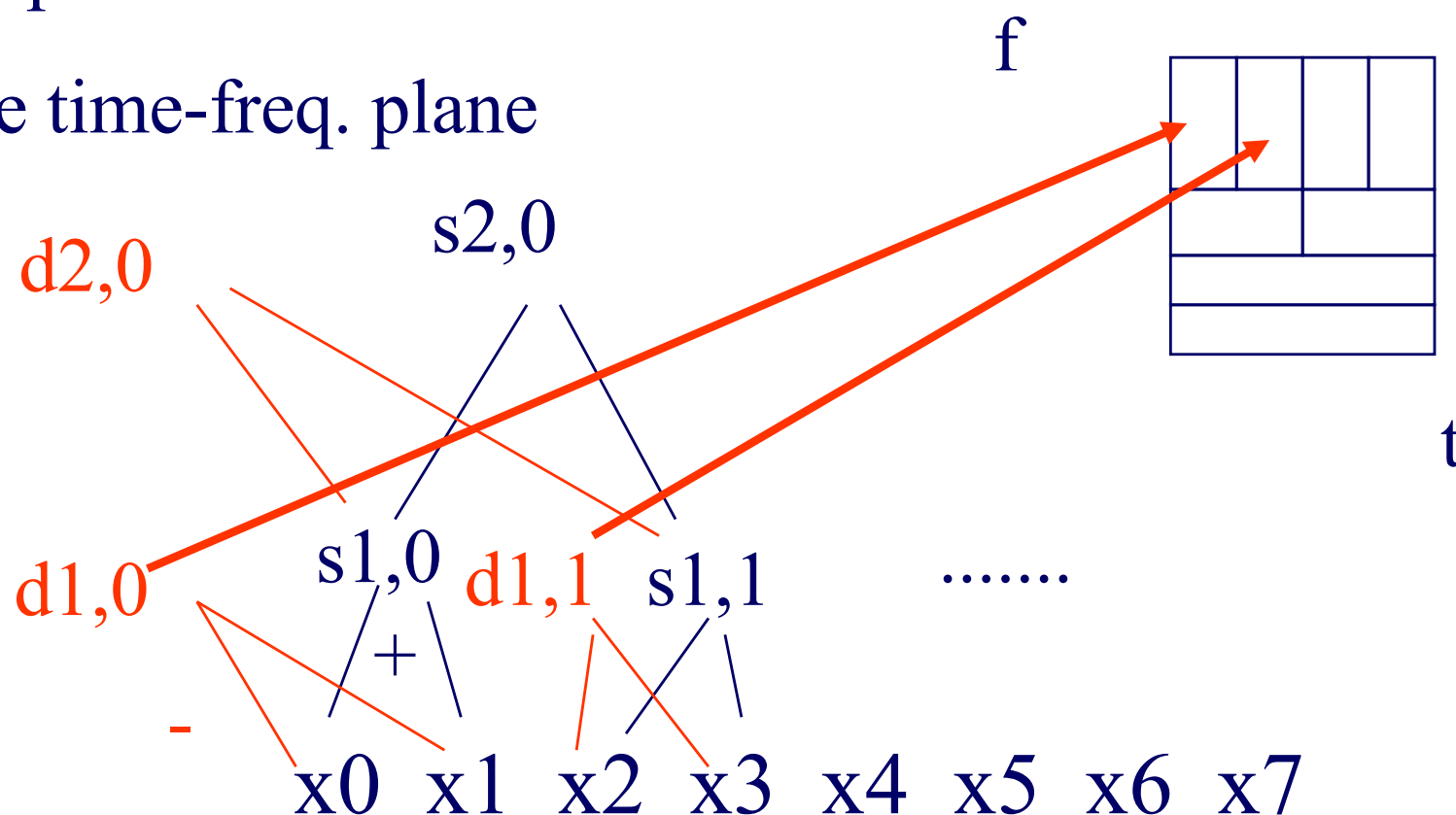
Wavelets - construction

Q: map each coefficient
on the time-freq. plane



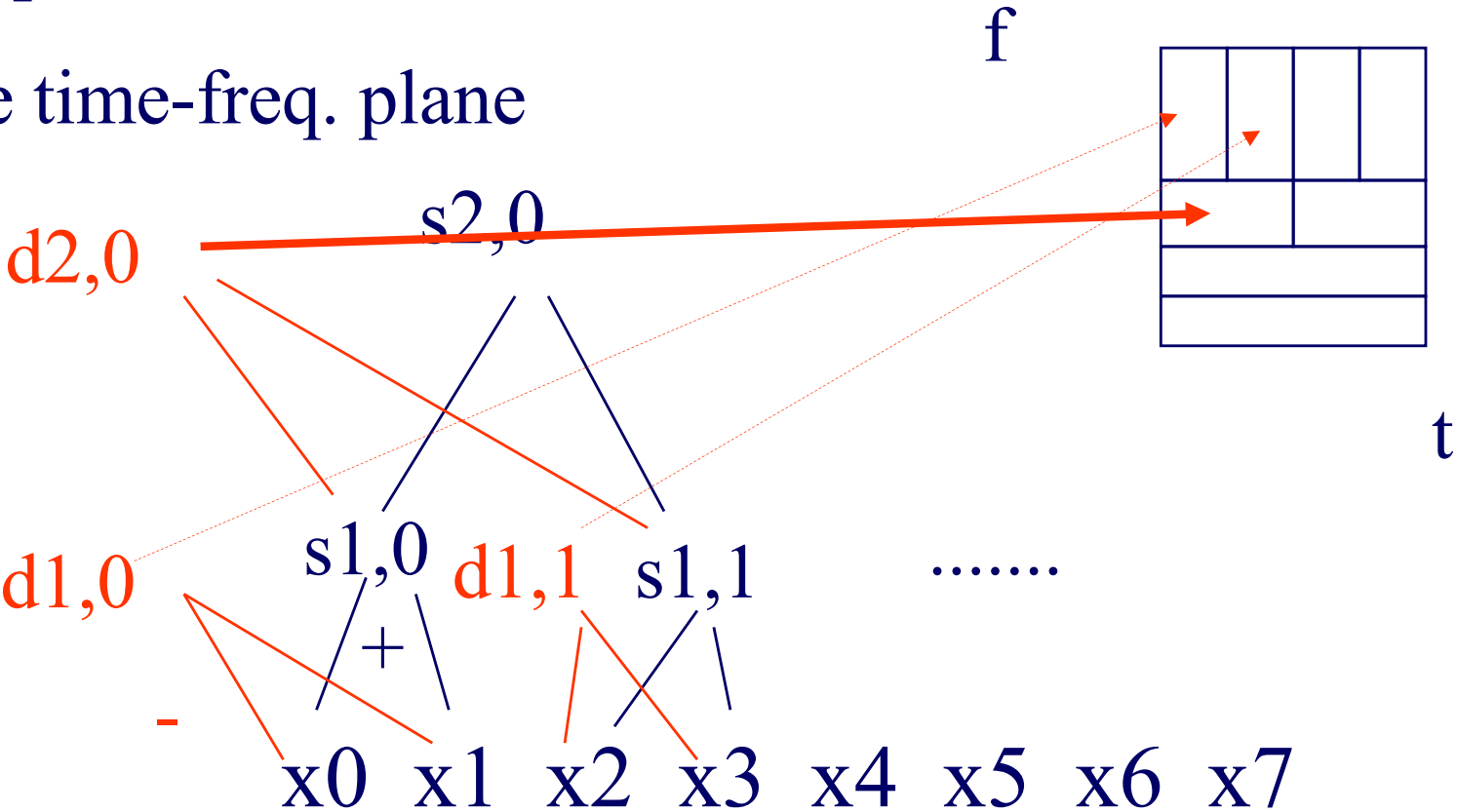
Wavelets - construction

Q: map each coefficient
on the time-freq. plane



Wavelets - construction

Q: map each coefficient
on the time-freq. plane



Haar wavelets - code

```
#!/usr/bin/perl5
# expects a file with numbers
# and prints the dwt transform
# The number of time-ticks should be a power of 2
# USAGE
#  haar.pl <fname>

my @vals=();
my @smooth; # the smooth component of the signal
my @diff; # the high-freq. component

# collect the values into the array @val
while(<>){
    @vals = ( @vals , split );
}
```

```
my $len = scalar(@vals);
my $half = int($len/2);
while($half >= 1 ){
    for(my $i=0; $i< $half; $i++){
        $diff [$i] = ($vals[2*$i] - $vals[2*$i + 1] )/ sqrt(2);
        print "\t", $diff[$i];
        $smooth [$i] = ($vals[2*$i] + $vals[2*$i + 1] )/ sqrt(2);
    }
    print "\n";
    @vals = @smooth;
    $half = int($half/2);
}
print "\t", $vals[0], "\n" ;    # the final, smooth component
```

Also at: www.cs.cmu.edu/~christos/SRC/DWT-Haar-all.tar

Wavelets - construction

Observation1:

‘+’ can be some weighted addition

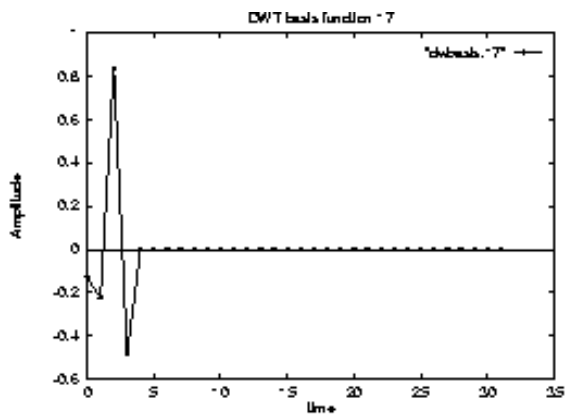
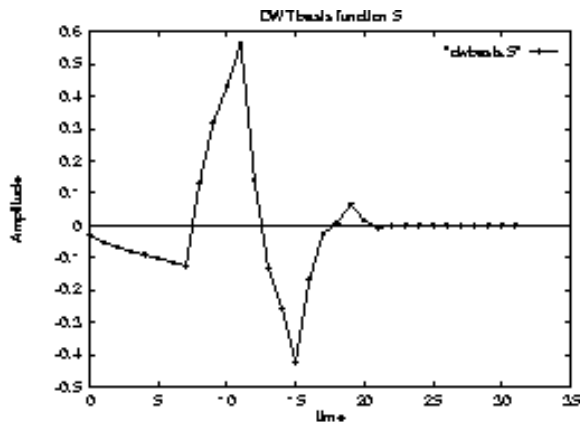
‘-’ is the corresponding weighted difference
(‘Quadrature mirror filters’)

Observation2: unlike DFT/DCT,

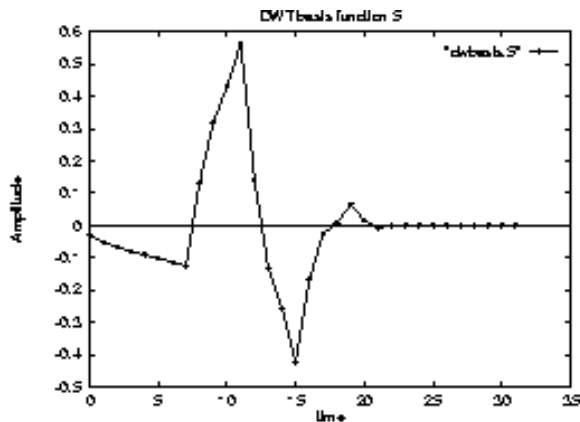
there are **many** wavelet bases: Haar, Daubechies-4, Daubechies-6, Coifman, Morlet, Gabor, ...

Wavelets - how do they look like?

- E.g., Daubechies-4

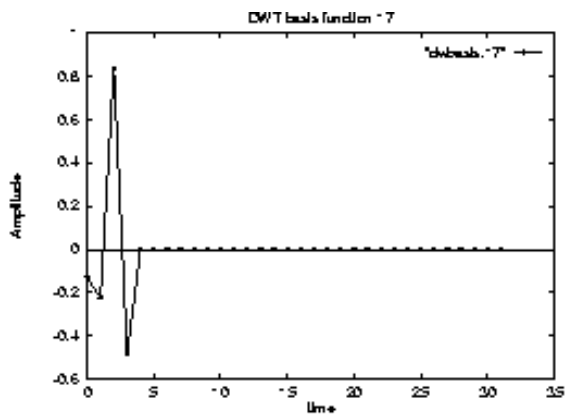


Wavelets - how do they look like?

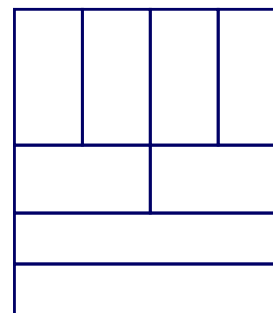


- E.g., Daubechies-4

?

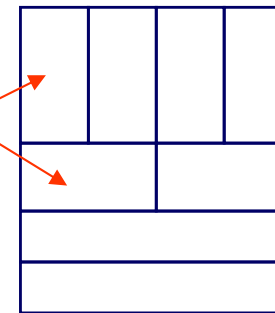
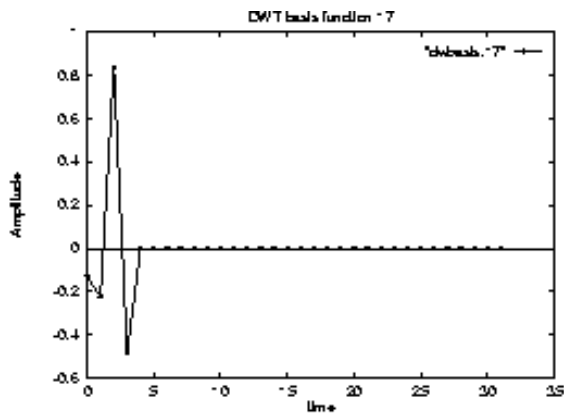
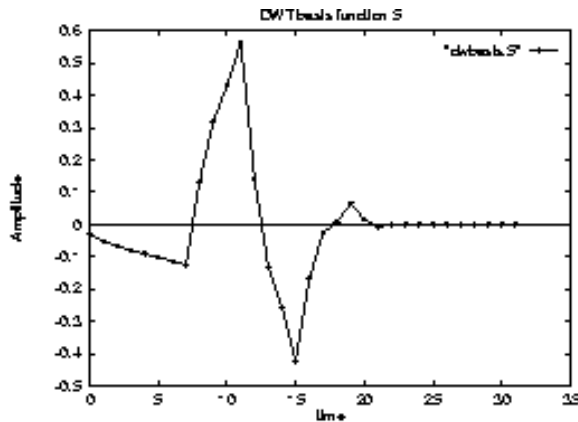


?



Wavelets - how do they look like?

- E.g., Daubechies-4



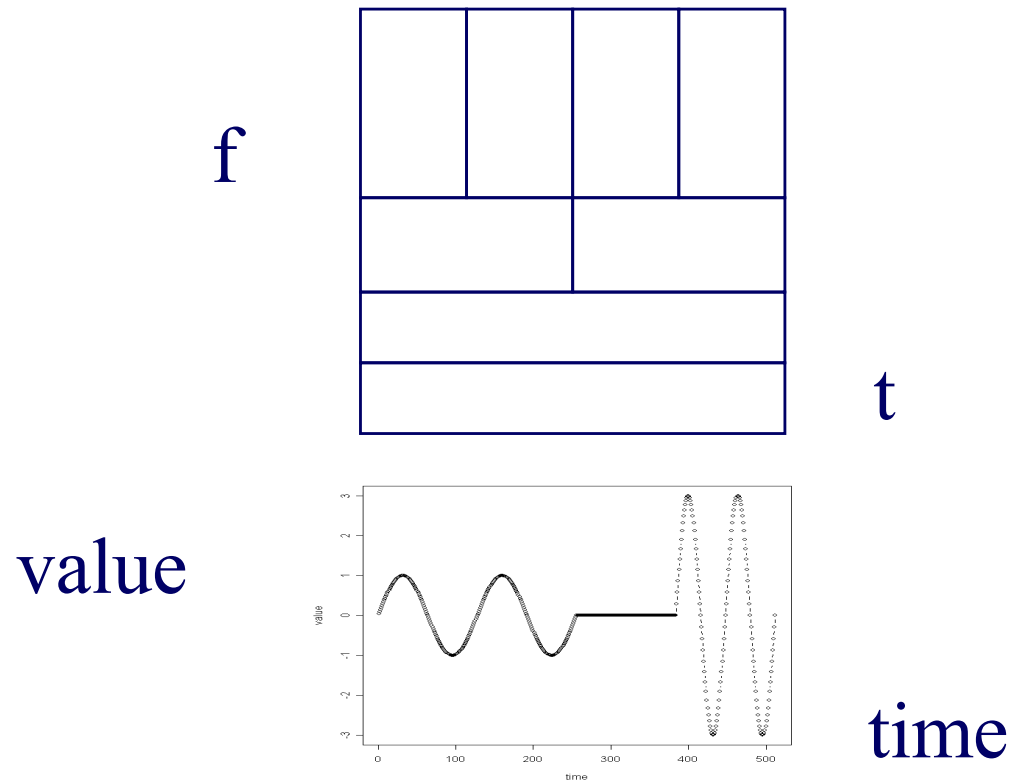


DWT - Detailed outline

- Motivation
- Definition - Intuition
- ➔ • Examples and drills
 - vs DFT
 - Qualitative
 - Quantitative
- Application: BGPlens
- 2-d DWT

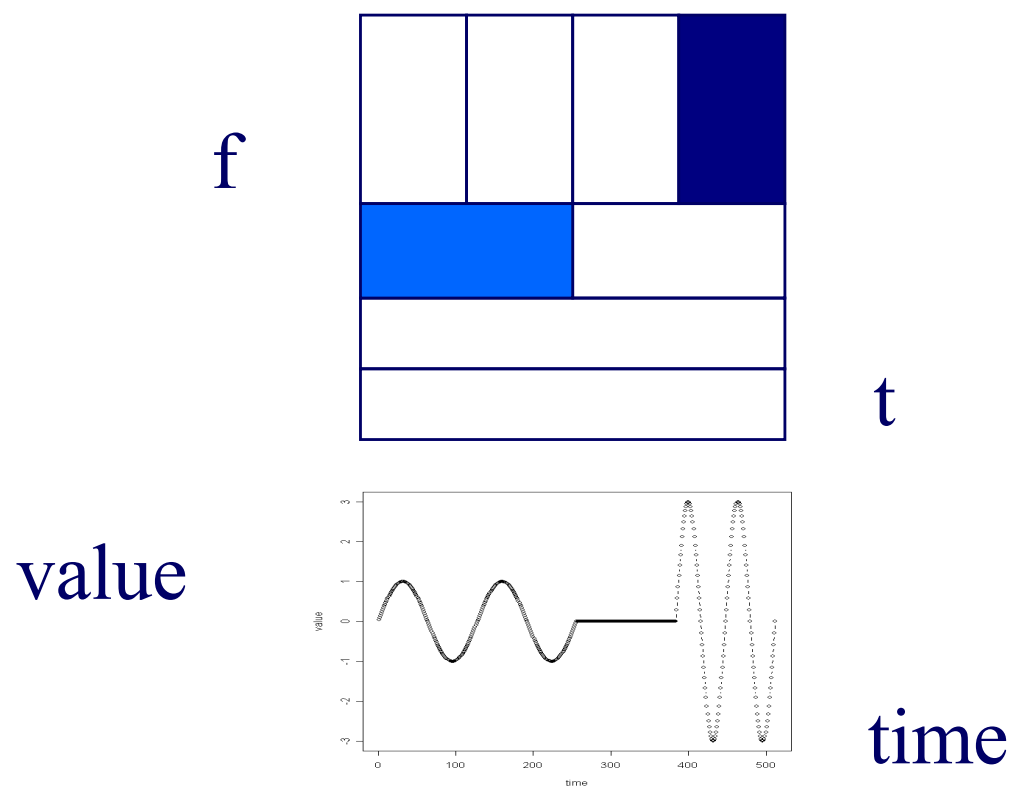
Wavelets - Drill#1:

- Q: baritone/silence/soprano - DWT?



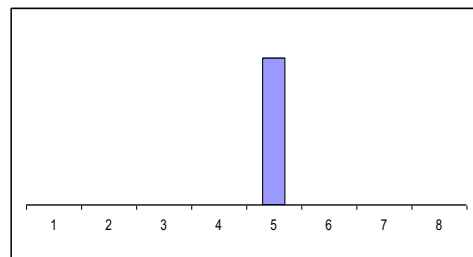
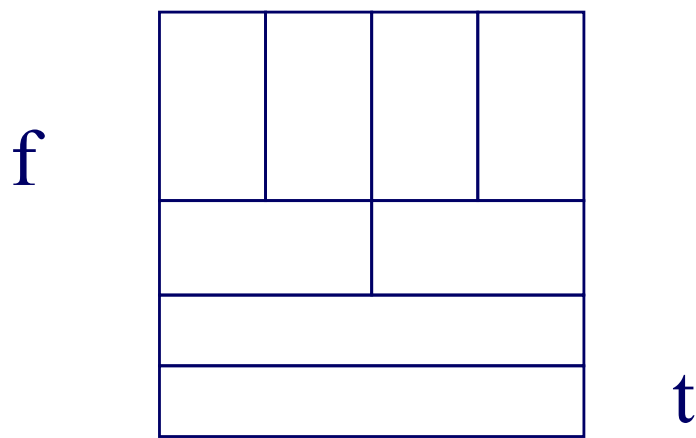
Wavelets - Drill#1:

- Q: baritone/silence/soprano - DWT?



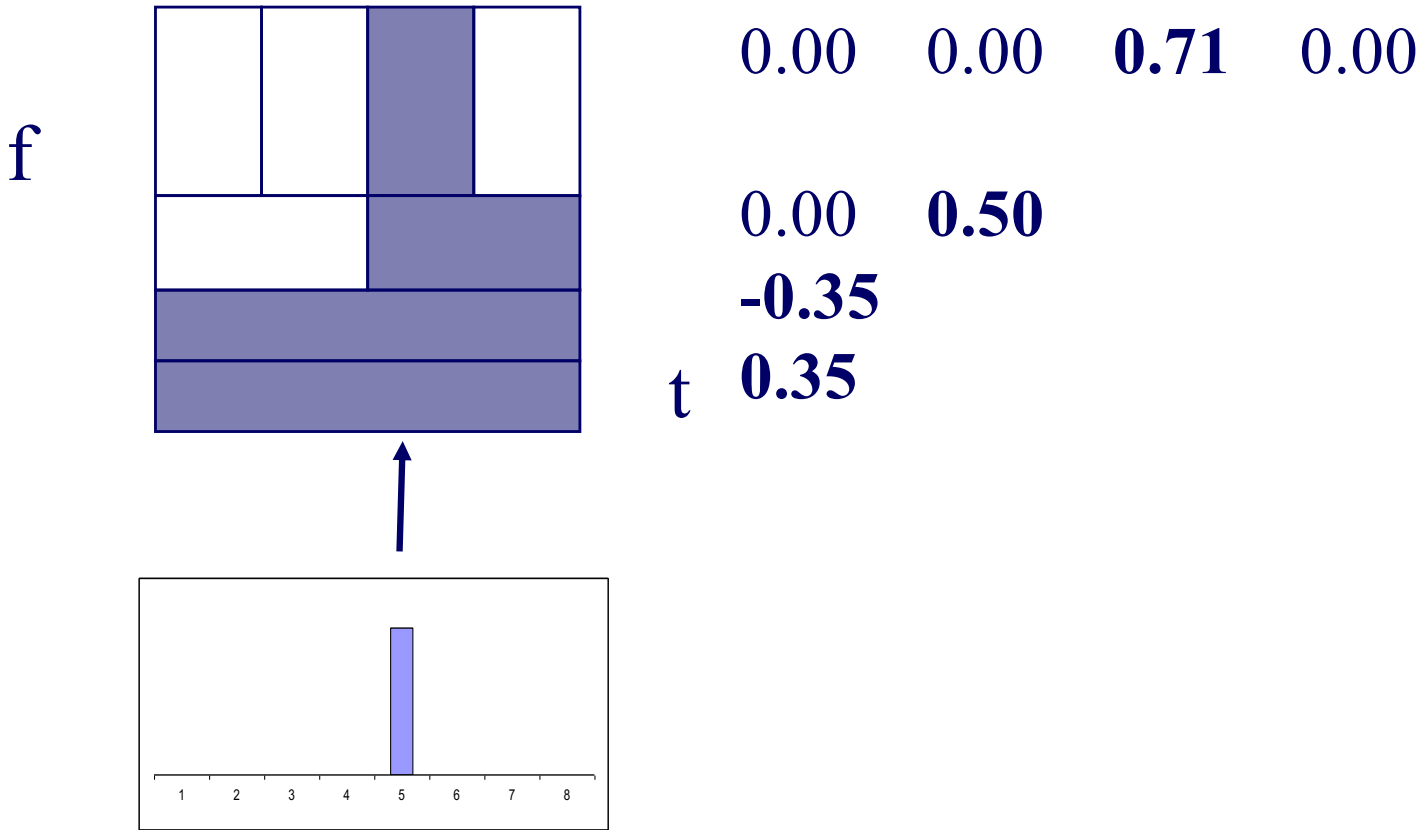
Wavelets - Drill#2:

- Q: spike - DWT?



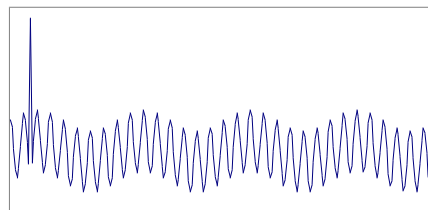
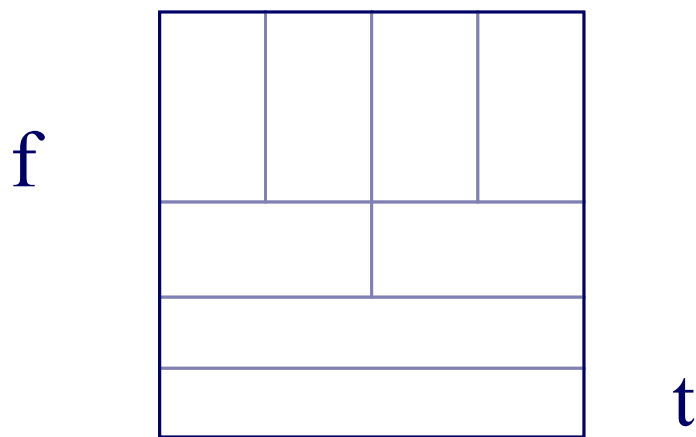
Wavelets - Drill#2:

- Q: spike - DWT?



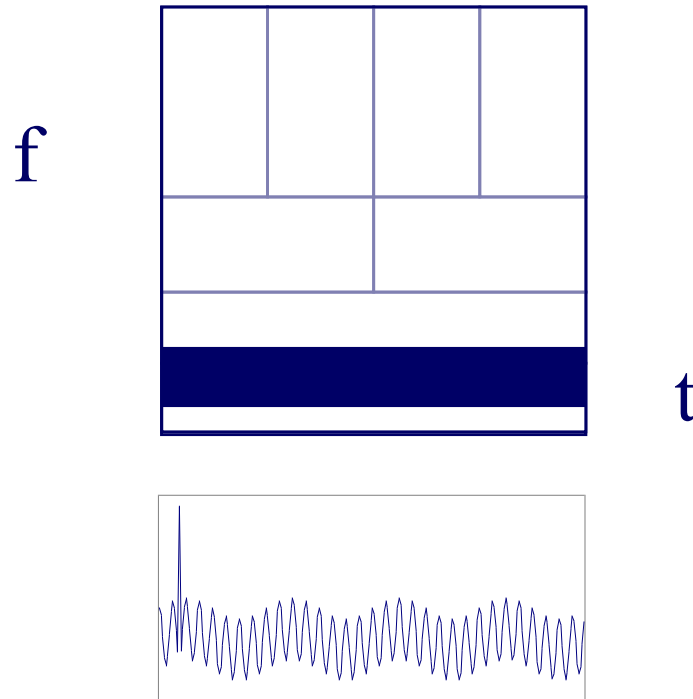
Wavelets - Drill#3:

- Q: weekly + daily periodicity, + spike - DWT?



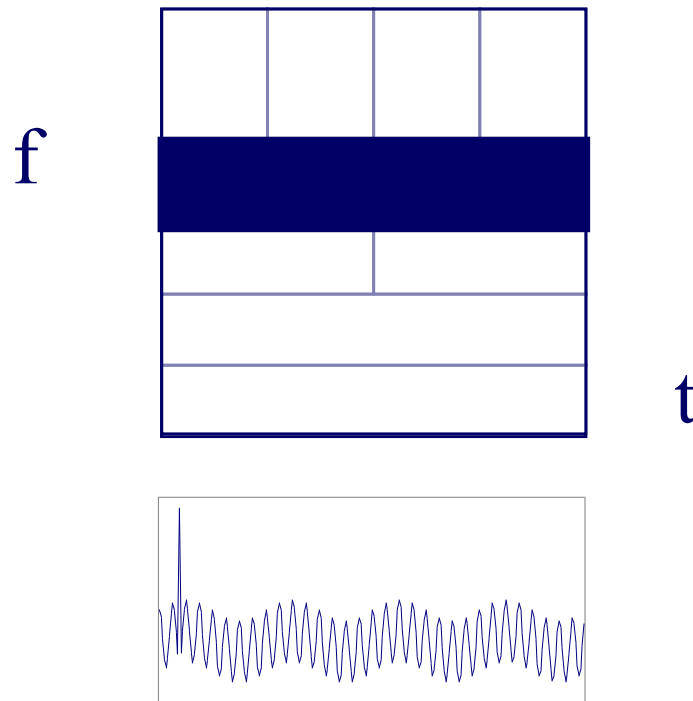
Wavelets - Drill#3:

- Q: **weekly** + daily periodicity, + spike - DWT?



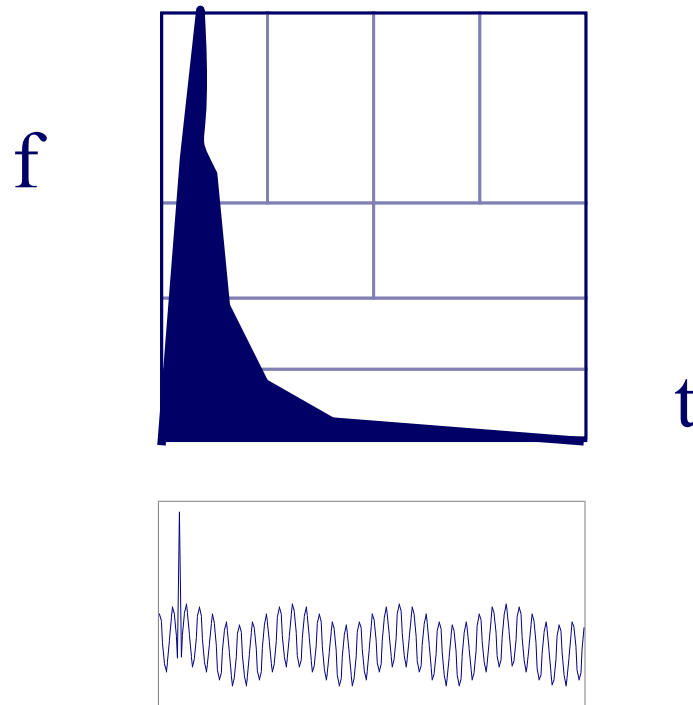
Wavelets - Drill#3:

- Q: weekly + **daily** periodicity, + spike - DWT?



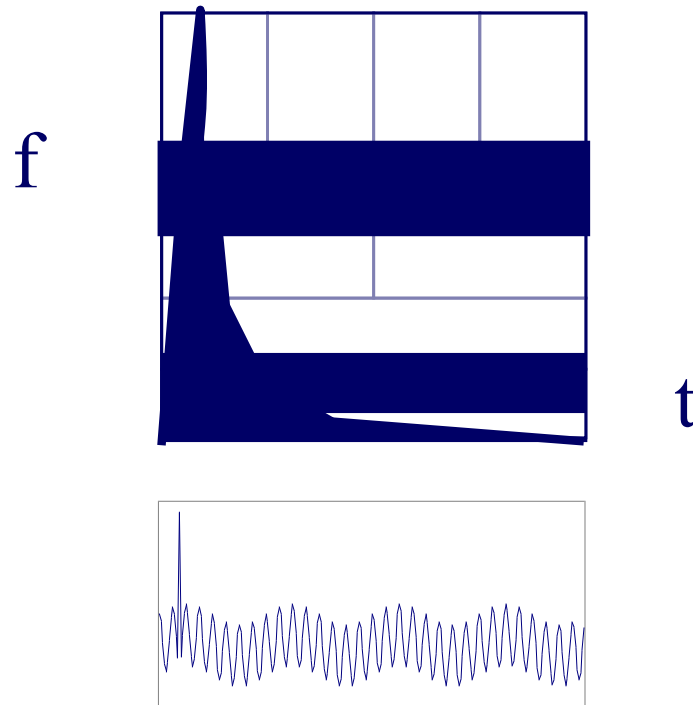
Wavelets - Drill#3:

- Q: weekly + daily periodicity, + **spike** - DWT?



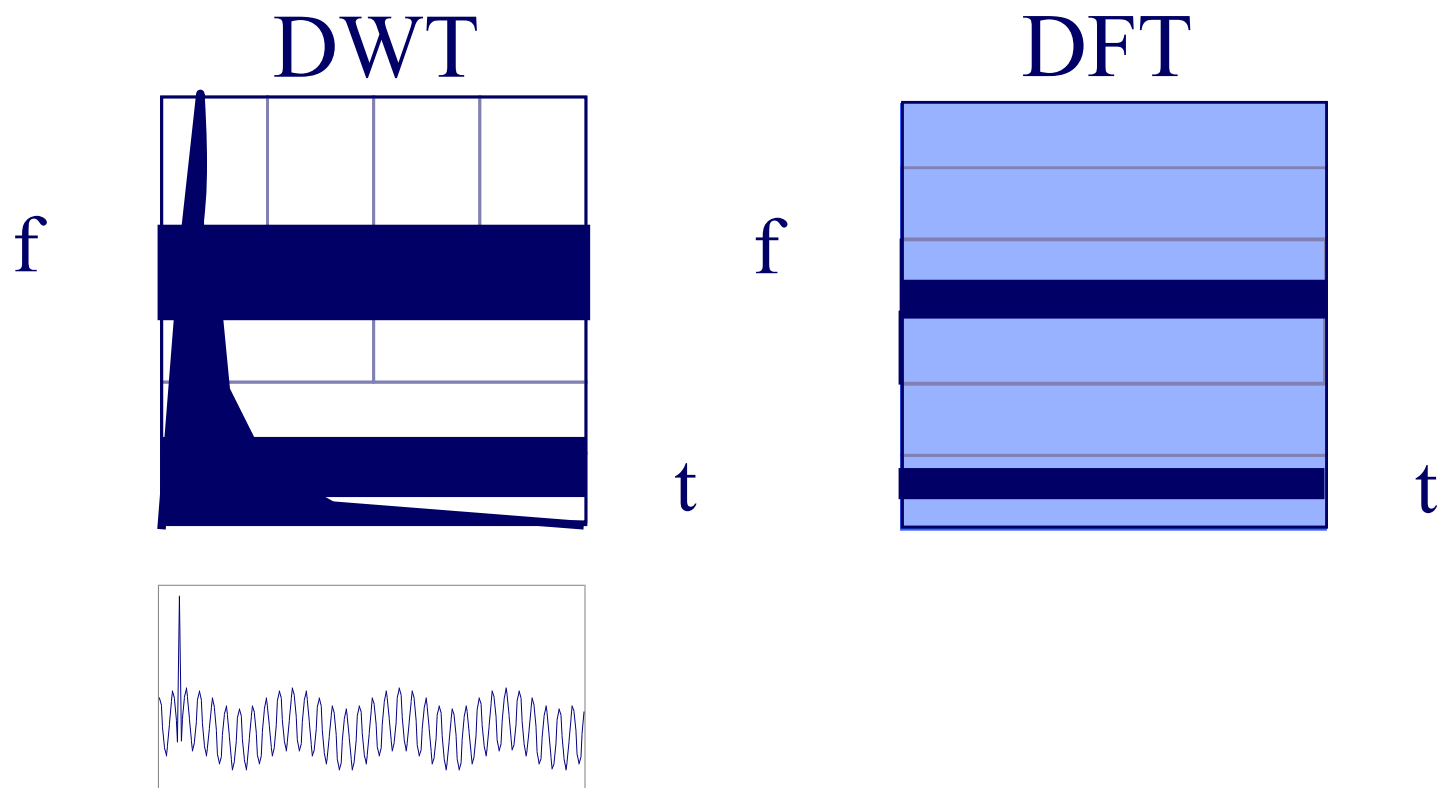
Wavelets - Drill#3:

- Q: weekly + daily periodicity, + spike - DWT?



Wavelets - Drill#3:

- Q: DFT?





DWT - Detailed outline

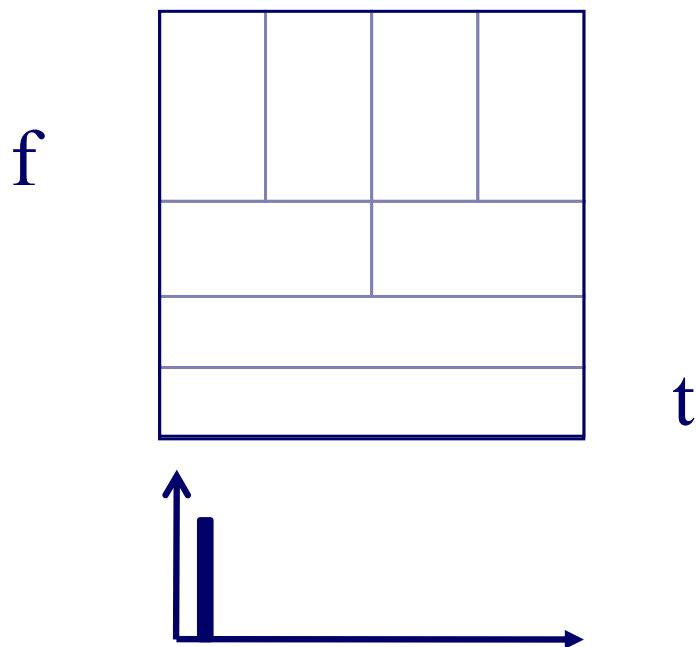
- Motivation
- Definition - Intuition
- Examples and drills
 - vs DFT
 - ➔ – Qualitative
 - Quantitative
- Application: BGPlens
- 2-d DWT

More drills - qualitatively

- Delta == spike
- cosine
- cosine2
- chirp

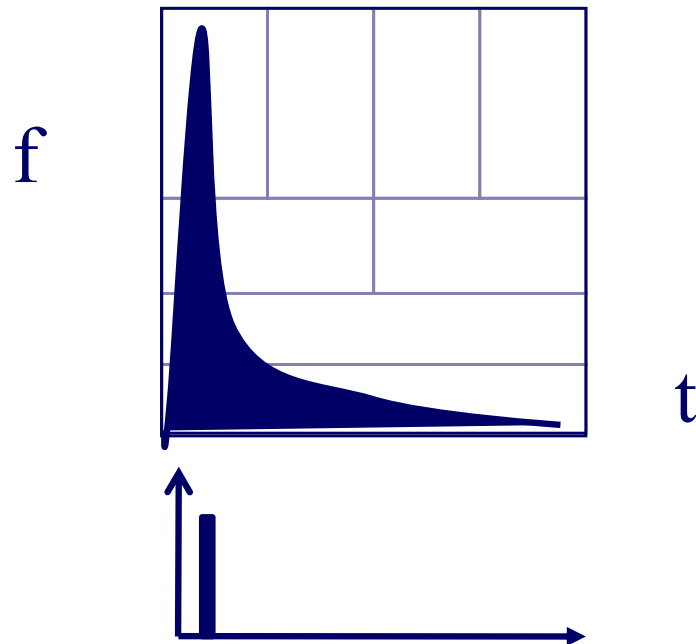
Delta?

$$x(0)=1; \quad x(t)=0 \text{ elsewhere}$$



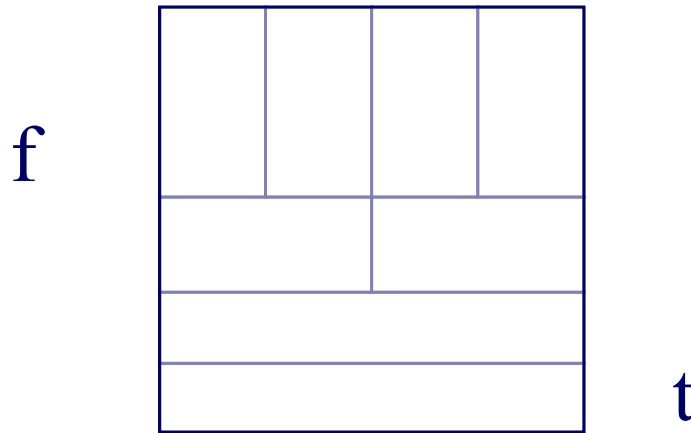
Delta?

$$x(0)=1; \quad x(t)=0 \text{ elsewhere}$$



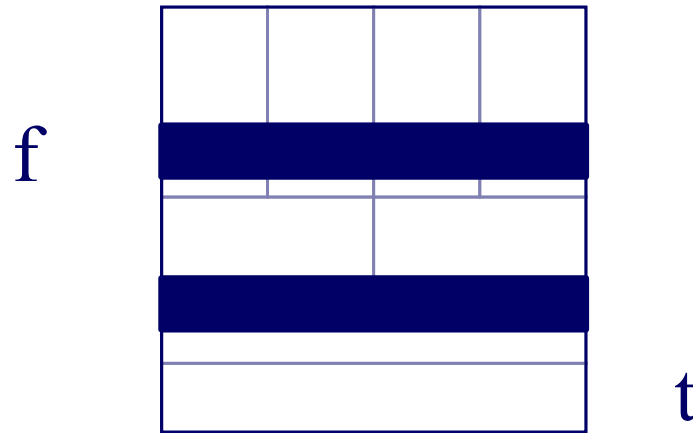
2 cosines?

$$x(t) = \cos(2 * \pi * 4 * t / 1024) + 5 * \cos(2 * \pi * 8 * t / 1024)$$



2 cosines?

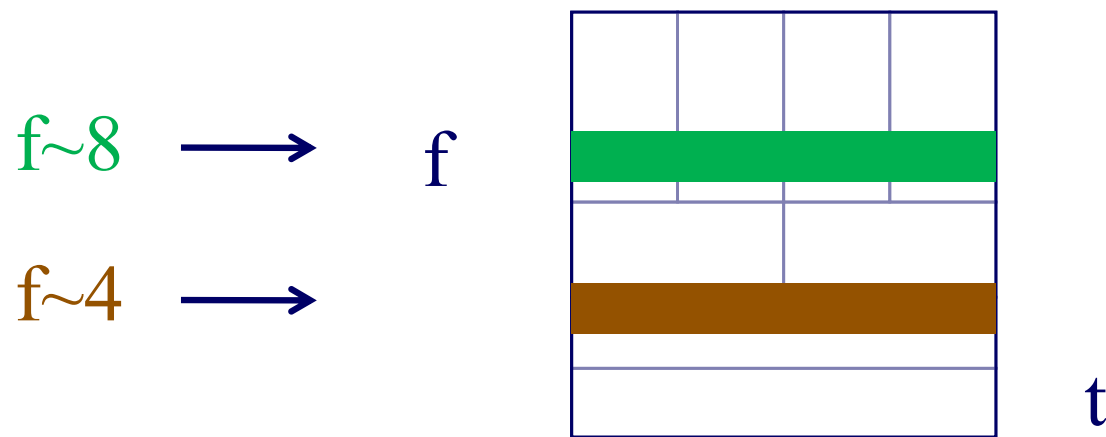
$$x(t) = \cos(2 * \pi * 4 * t / 1024) + 5 * \cos(2 * \pi * 8 * t / 1024)$$



Which one
is for freq.=4?

2 cosines?

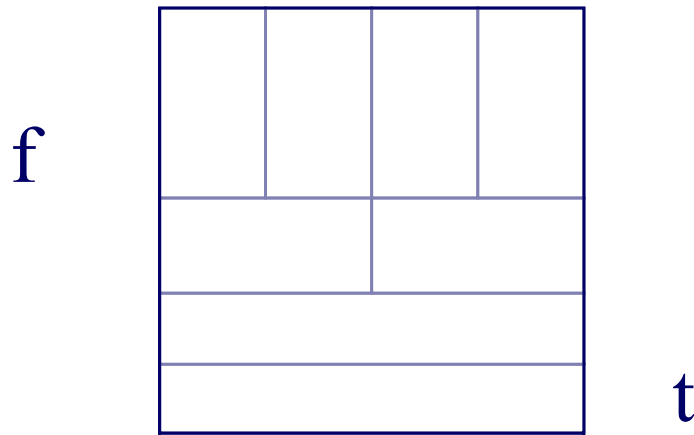
$$x(t) = \cos(2 * \pi * 4 * t / 1024) + 5 * \cos(2 * \pi * 8 * t / 1024)$$



Which one
is for freq.=4?

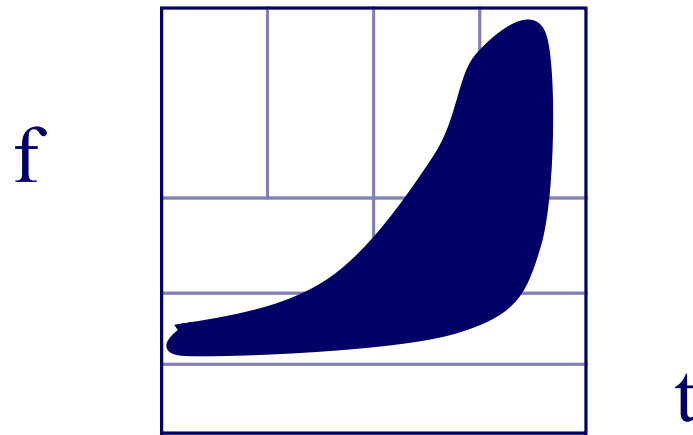
Chirp?

$$x(t) = \cos(2 * \pi * t * t / 1024)$$



Chirp?

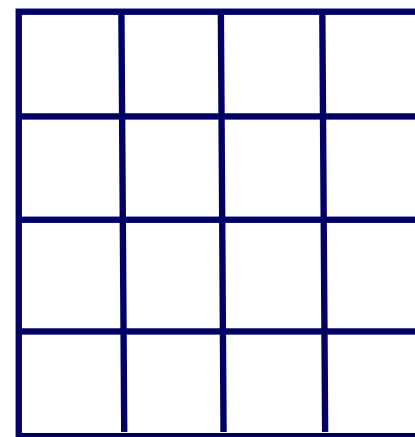
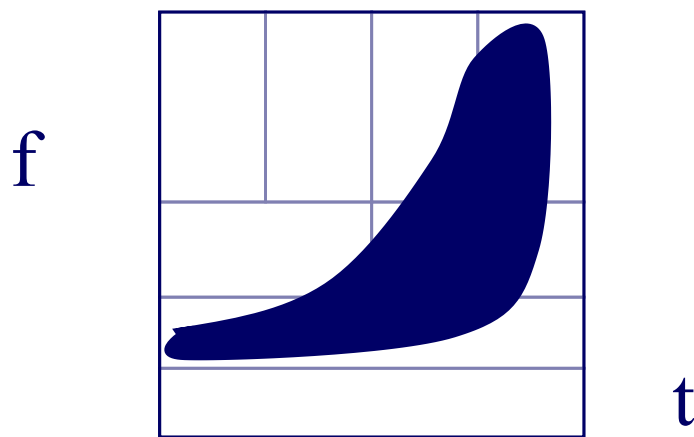
$$x(t) = \cos(2 * \pi * t * t / 1024)$$



Chirp?

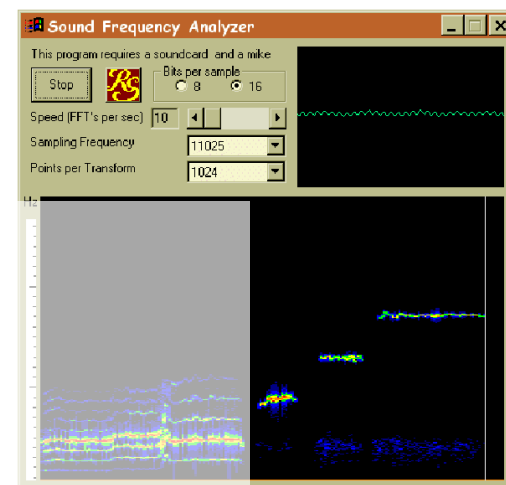
$$x(t) = \cos(2 * \pi * t * t / 1024)$$

SWFT?

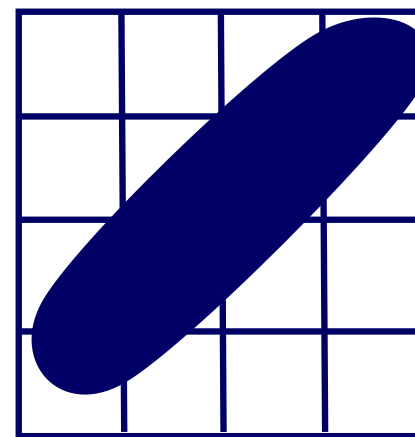
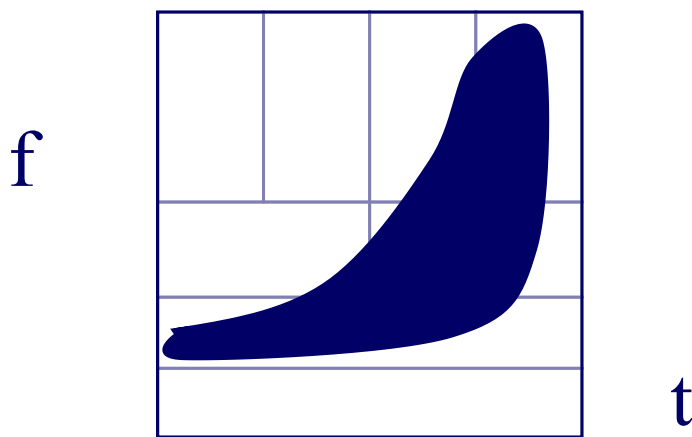


Chirp?

$$x(t) = \cos(2 * \pi * t * t / 1024)$$



SWFT





DWT - Detailed outline

- Motivation
- Definition - Intuition
- Examples and drills
 - vs DFT
 - Qualitative
 - – Quantitative
- Application: BGPlens
- 2-d DWT

Tar file:

https://www.cs.cmu.edu/%7Echristos/SRC/haar_demo_all.tar

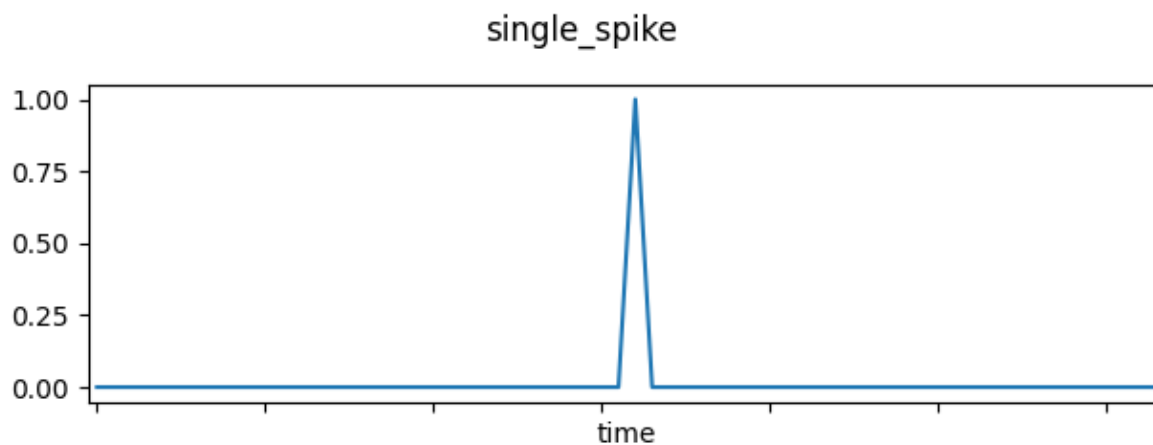
In a **new** python environment (`python -m venv venv`)

```
tar xvf haar_demo_all.tar  
make
```

Drills - quantitatively

- Spike
- Prolonged spike
- High-frequency square wave
- Slow square wave
- Slow sine wave

Spike



freq.

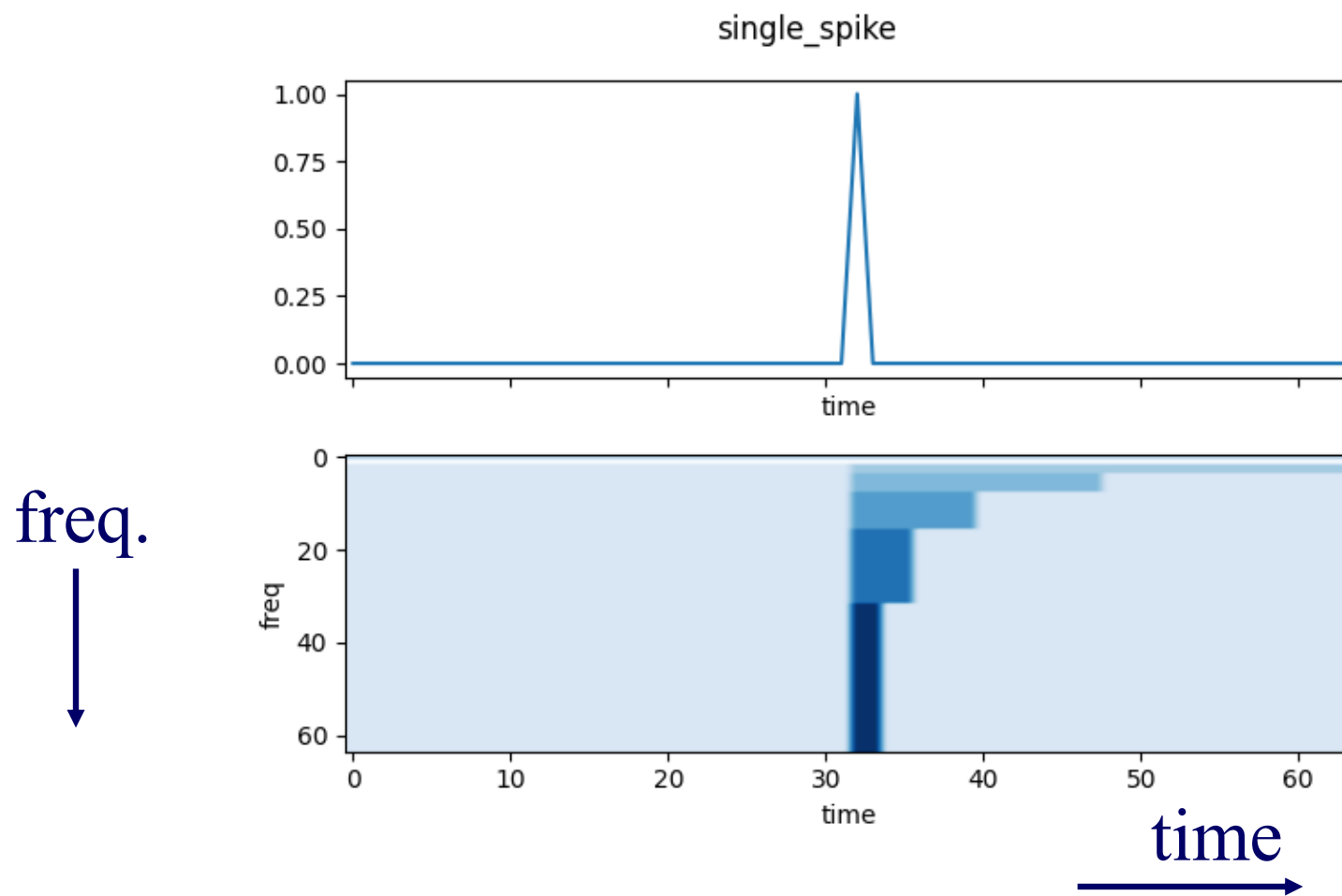


time

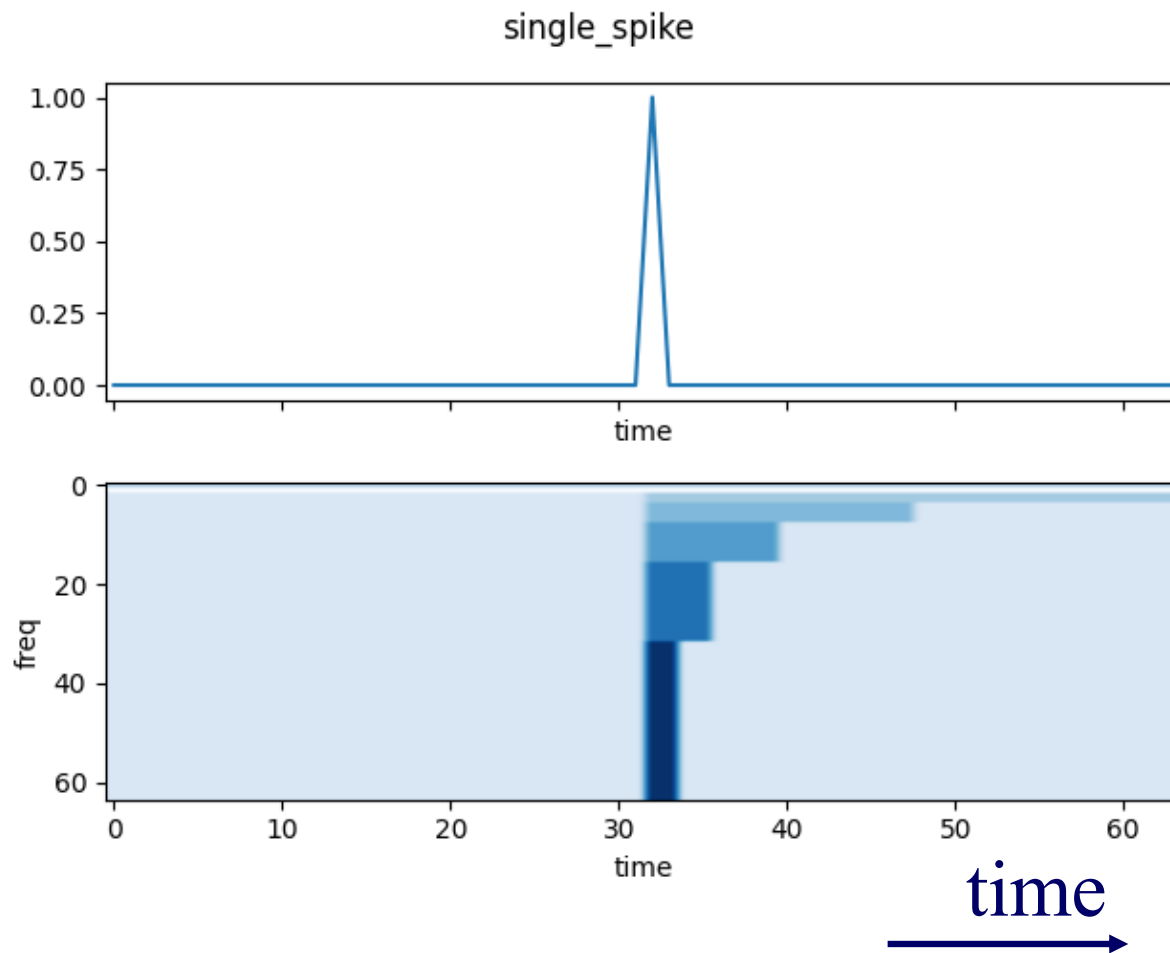
time



Spike



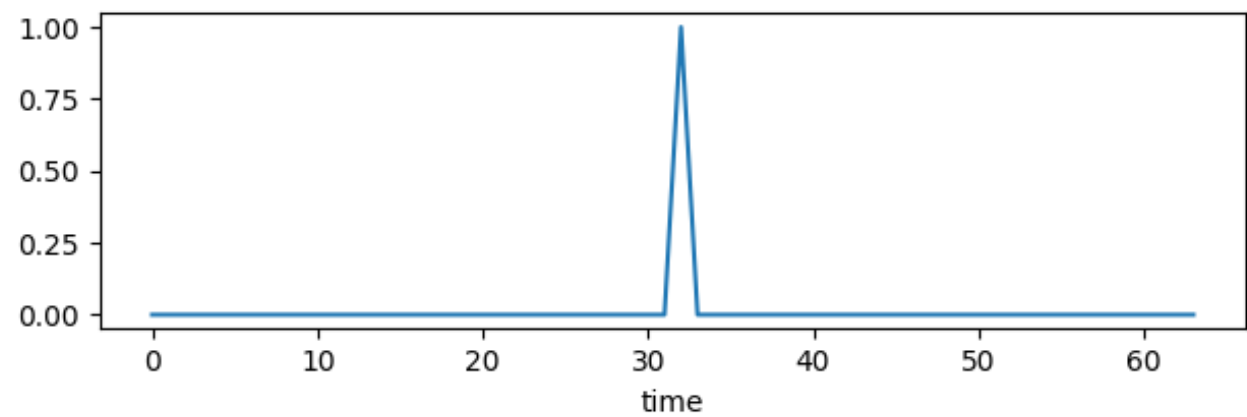
Spike



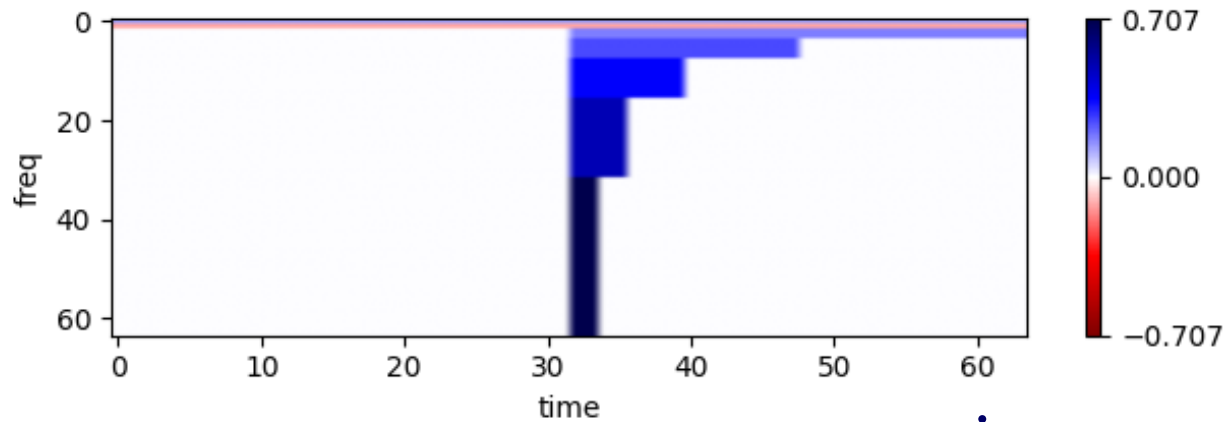
w/ color

Spike

single_spike



freq.
↓

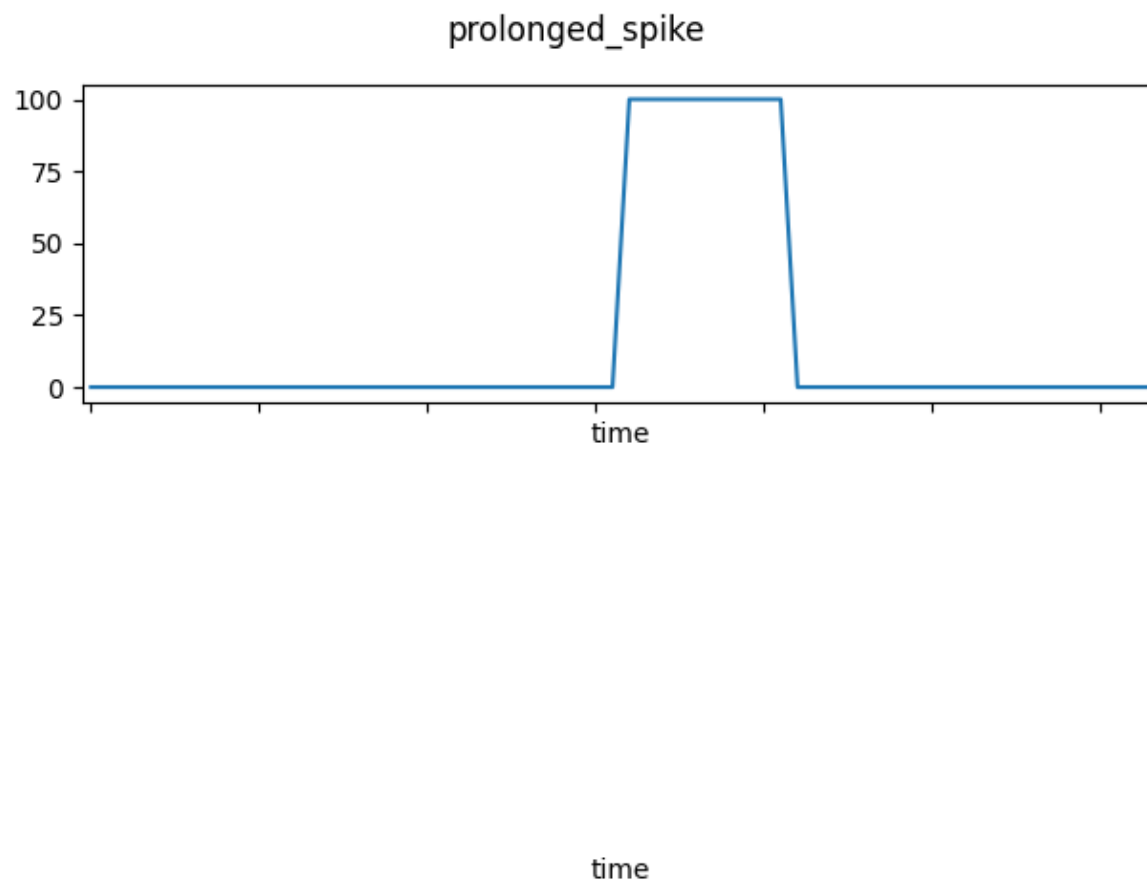


time
→

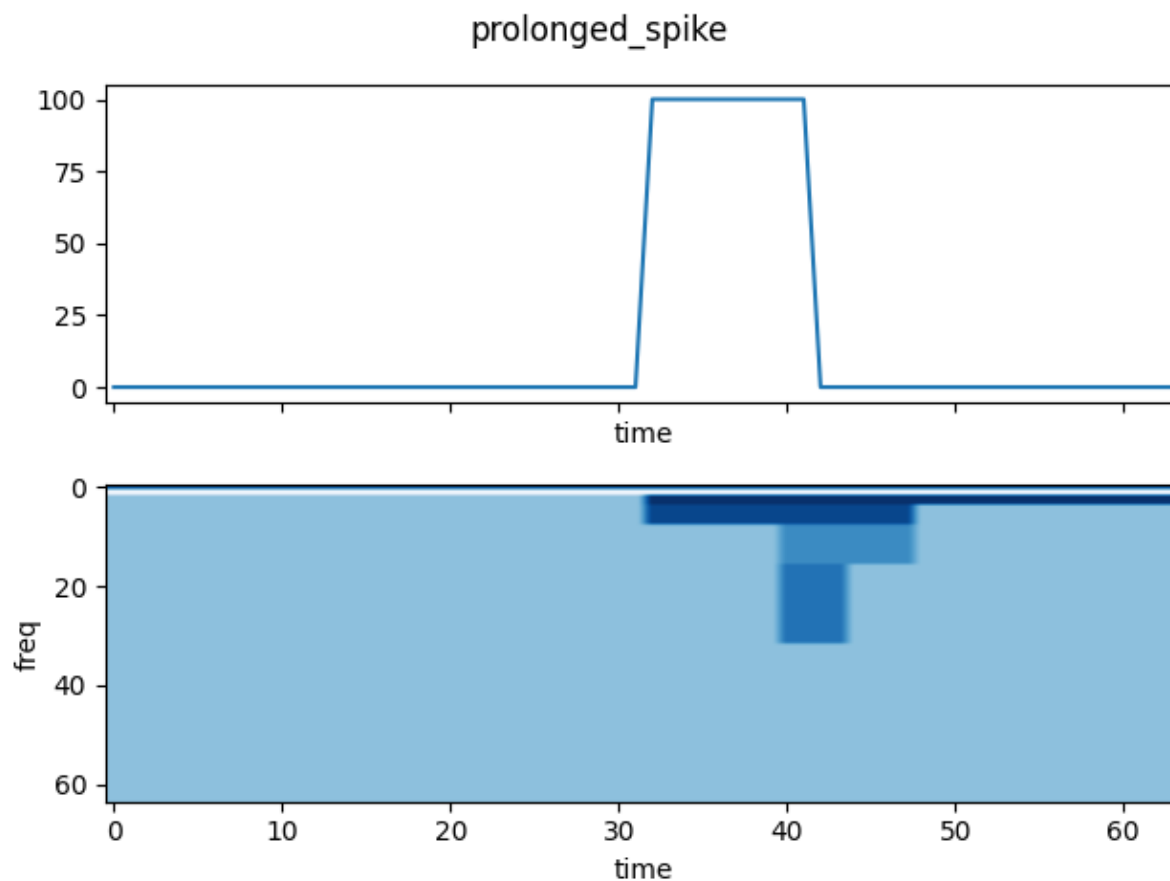
Drills - quantitatively

- ✓ Spike
 - Prolonged spike
 - High-frequency square wave
 - Slow square wave
 - Slow sine wave

Prolonged spike

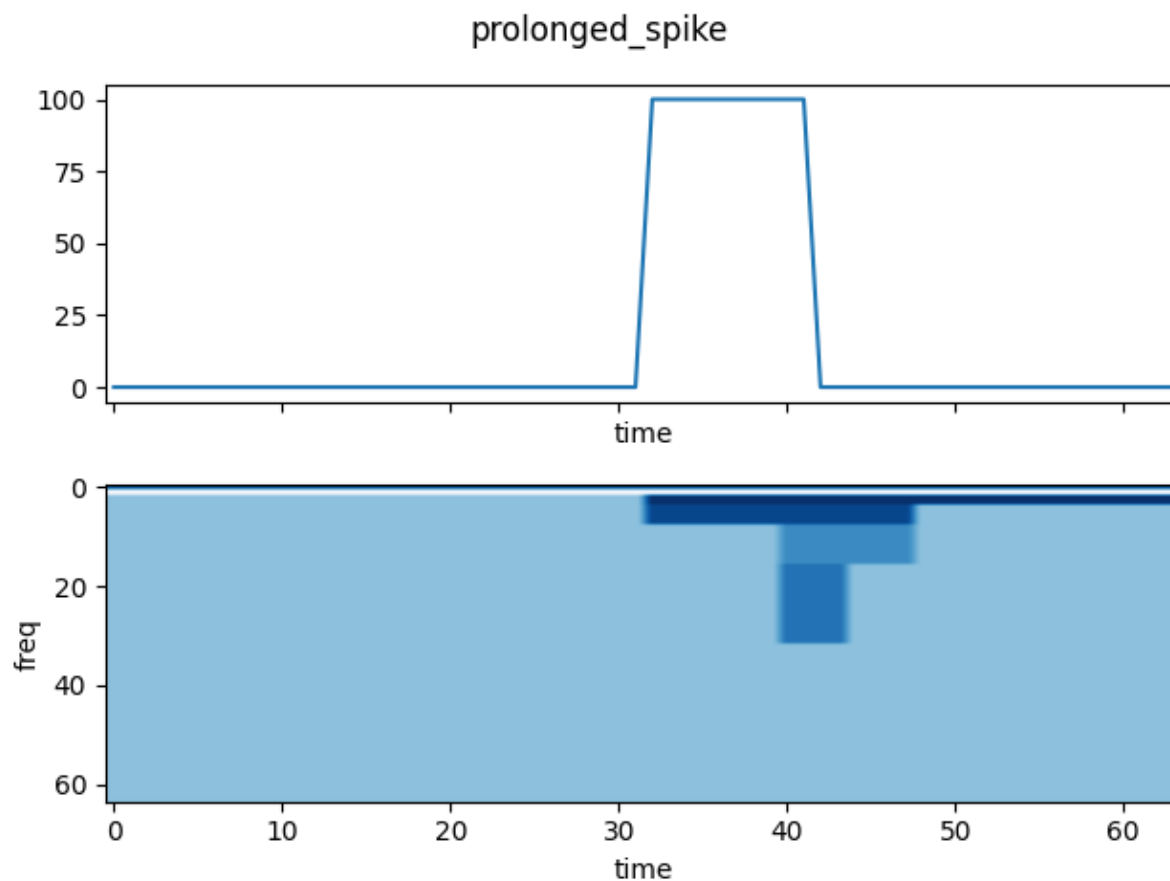


Prolonged spike



Keep it in mind

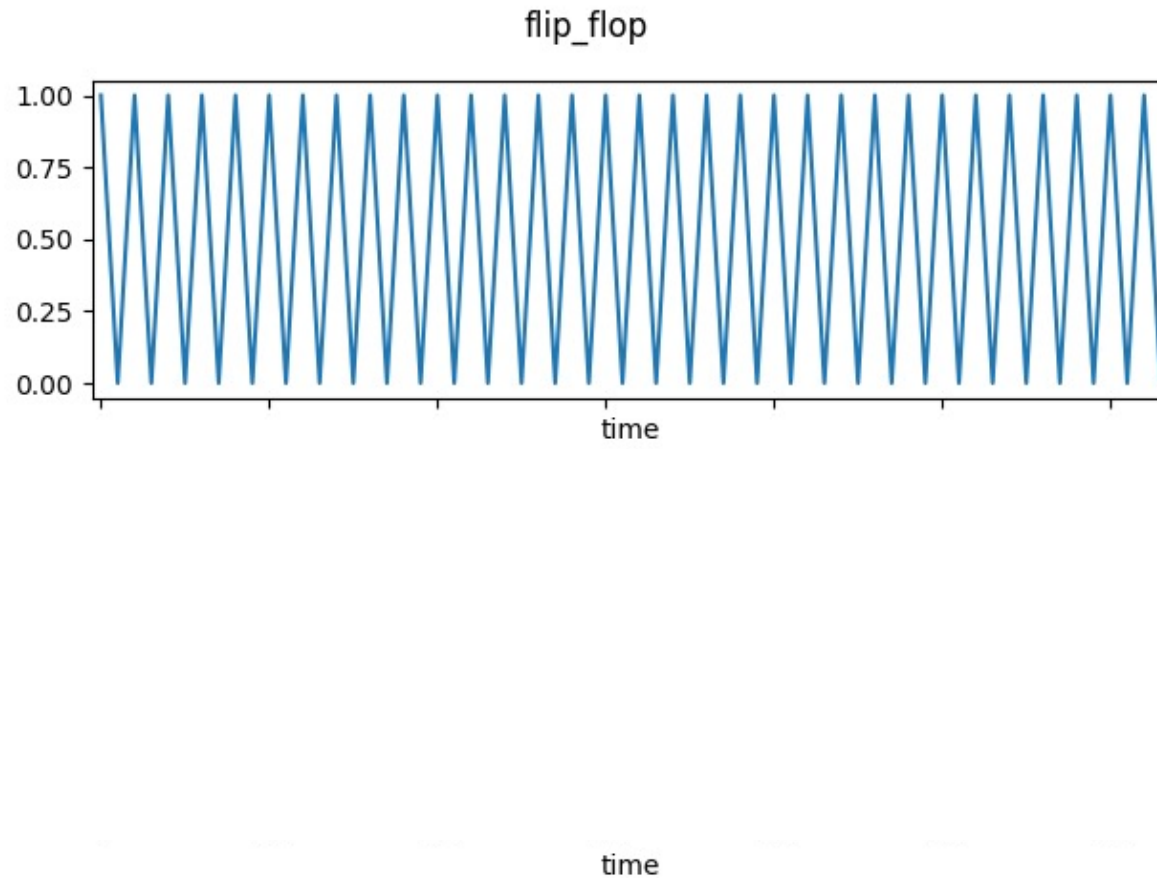
Prolonged spike



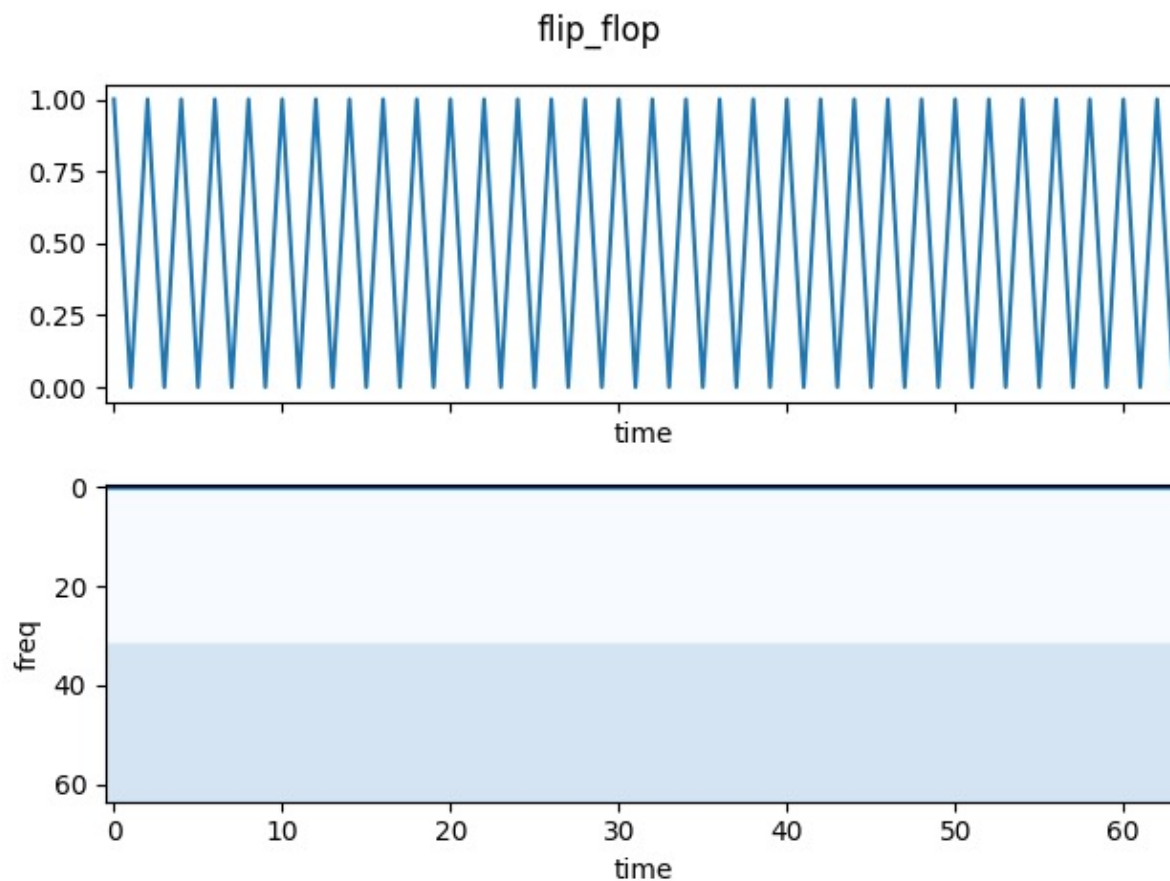
Drills - quantitatively

- ✓ Spike
- ✓ Prolonged spike
 - High-frequency square wave
 - Slow square wave
 - Slow sine wave

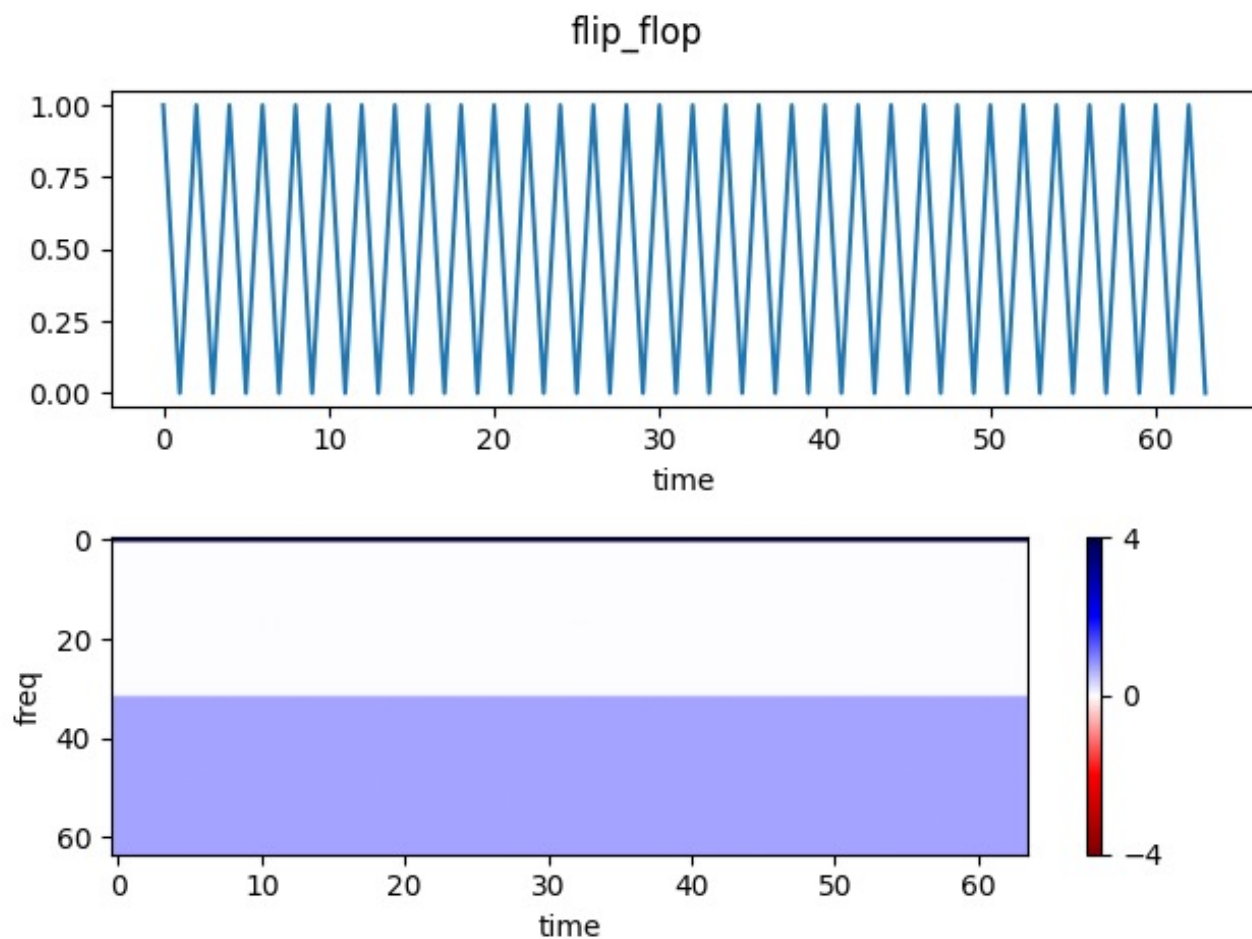
High freq square wave



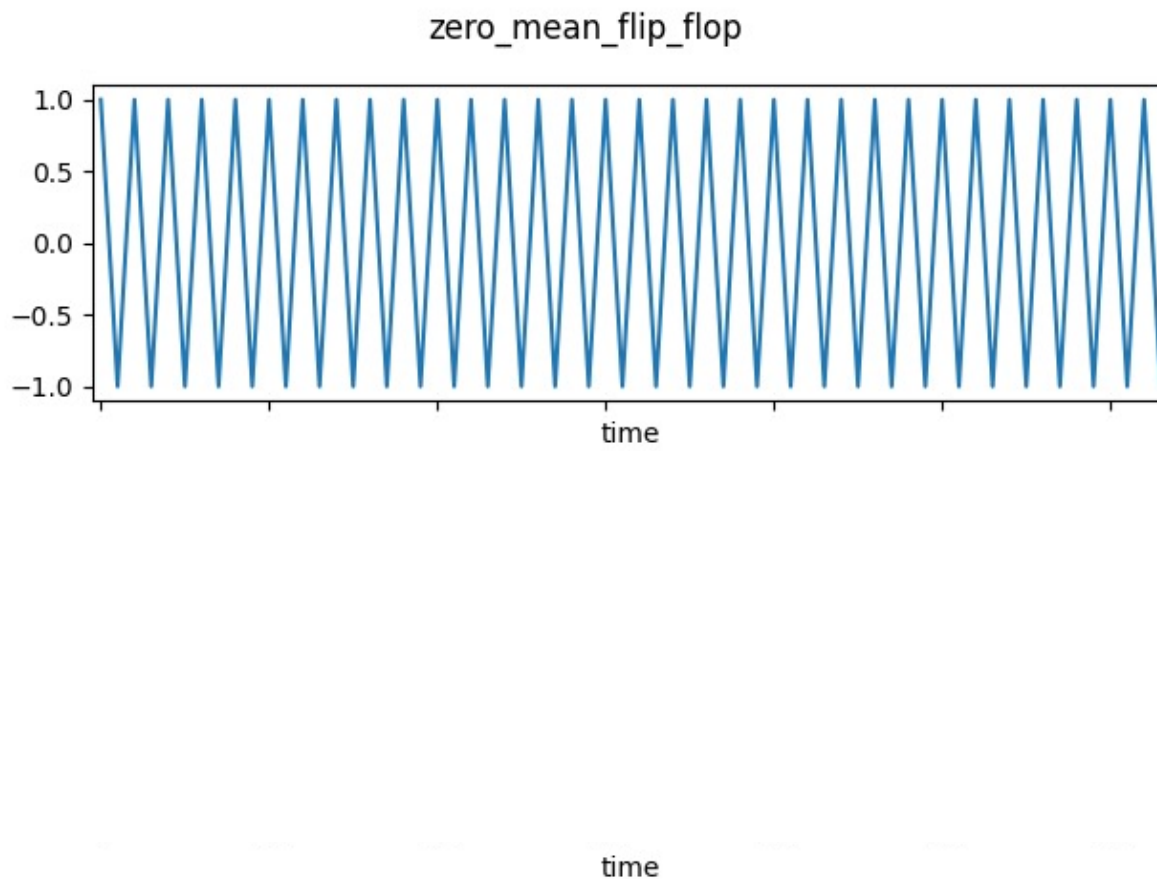
High freq square wave



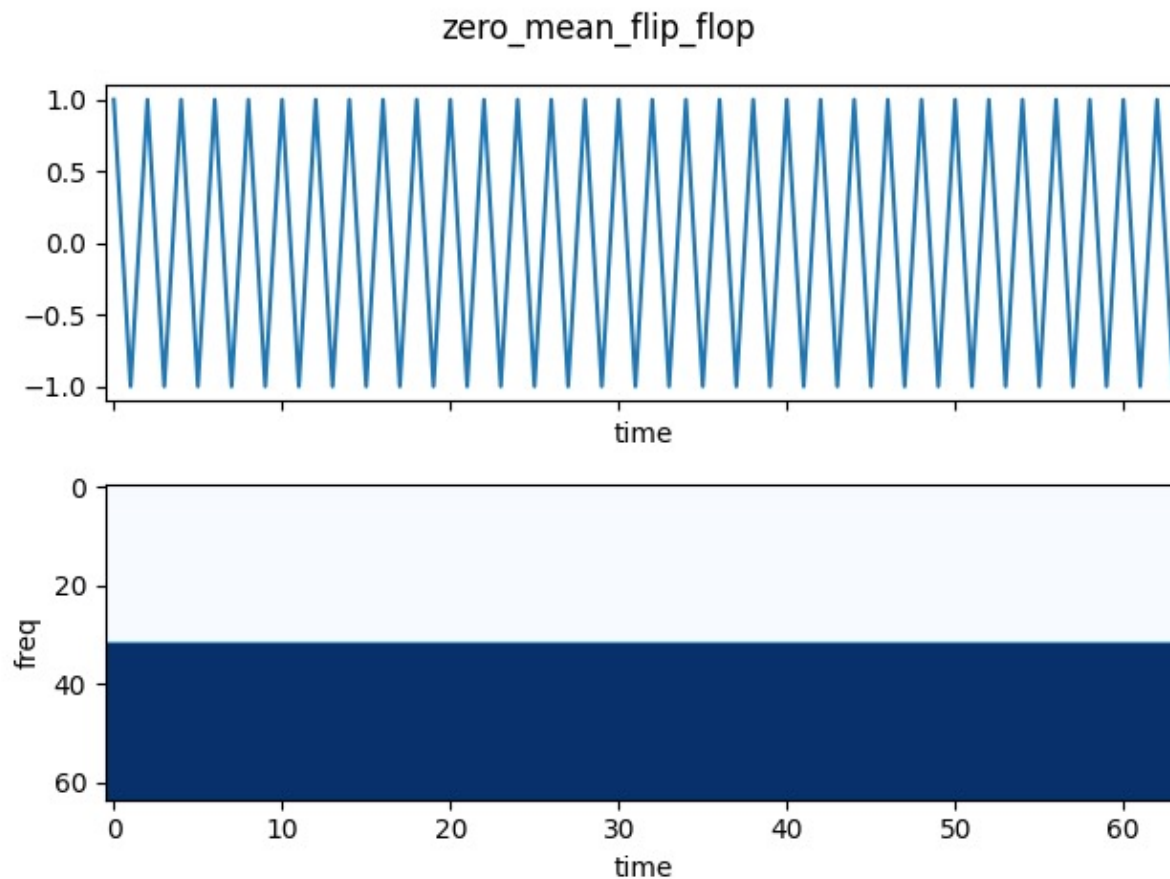
High freq square wave



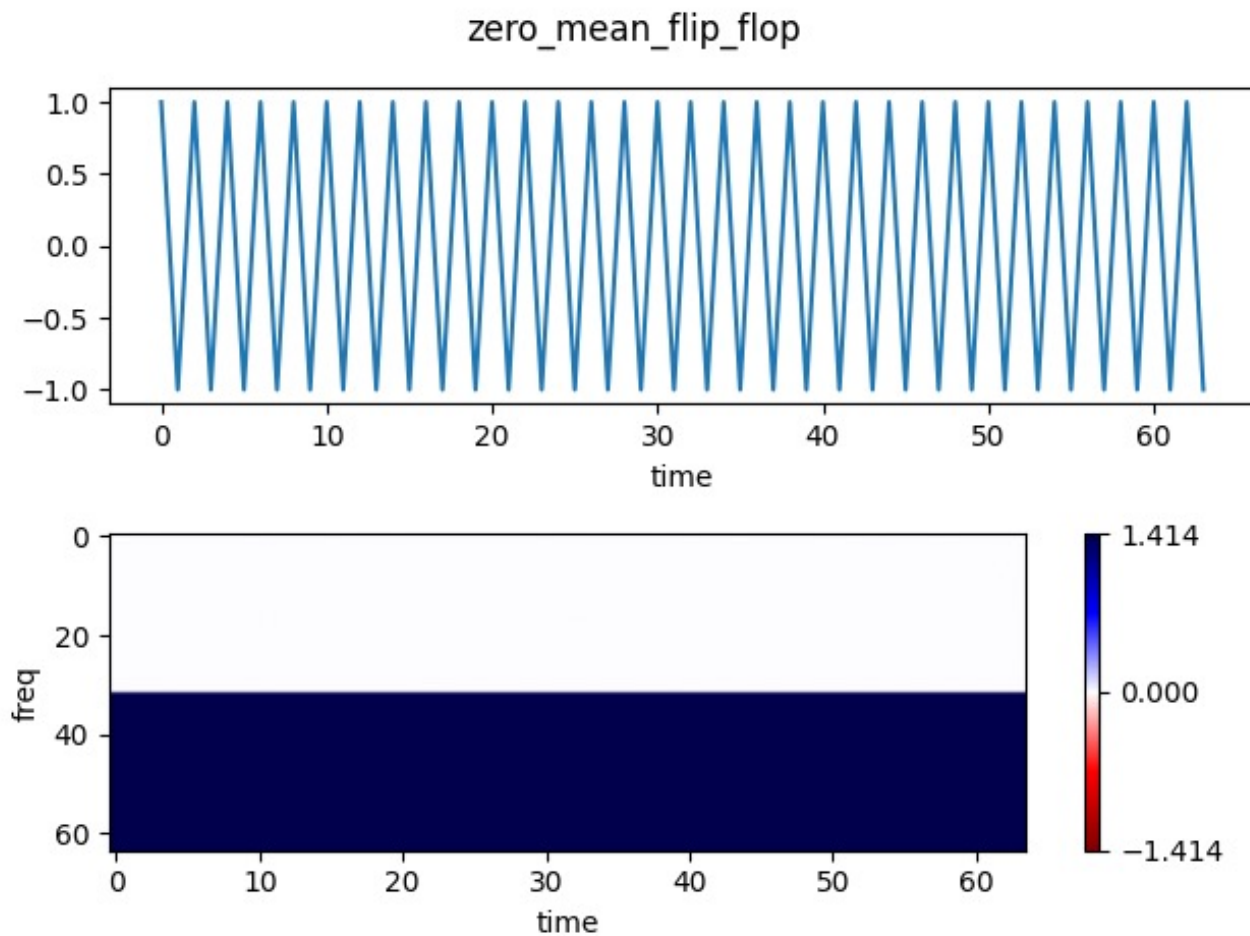
Zero-mean high freq.



Zero-mean high freq.



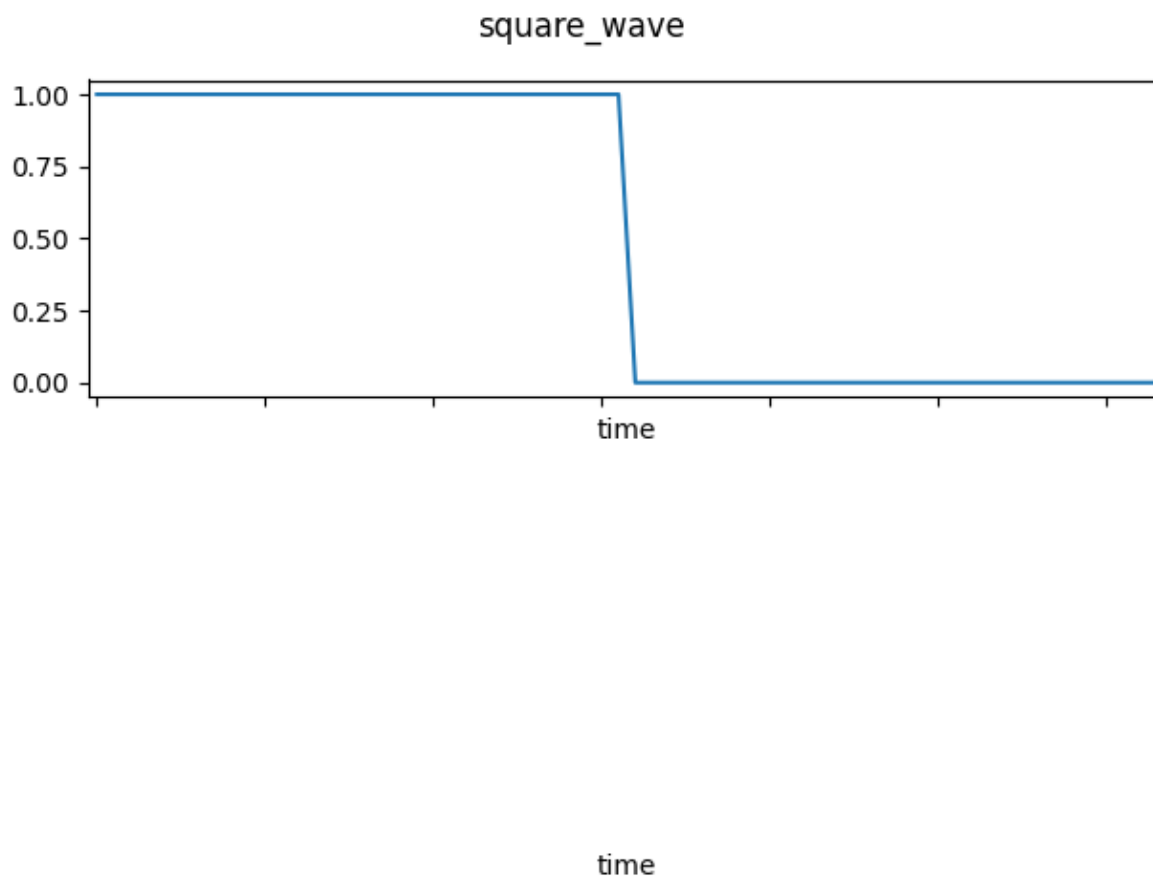
Zero-mean high freq.



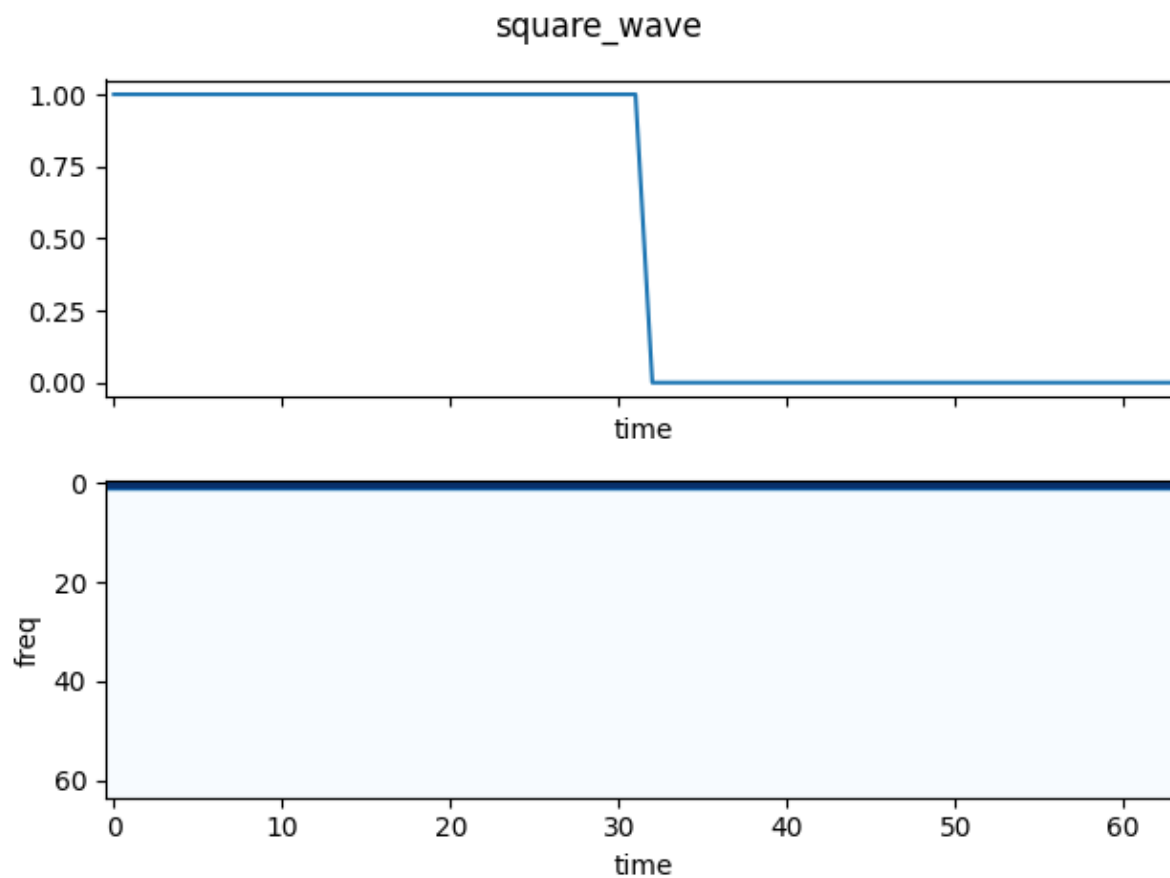
Drills - quantitatively

- ✓ Spike
- ✓ Prolonged spike
- ✓ High-frequency square wave
- Slow square wave
- Slow sine wave

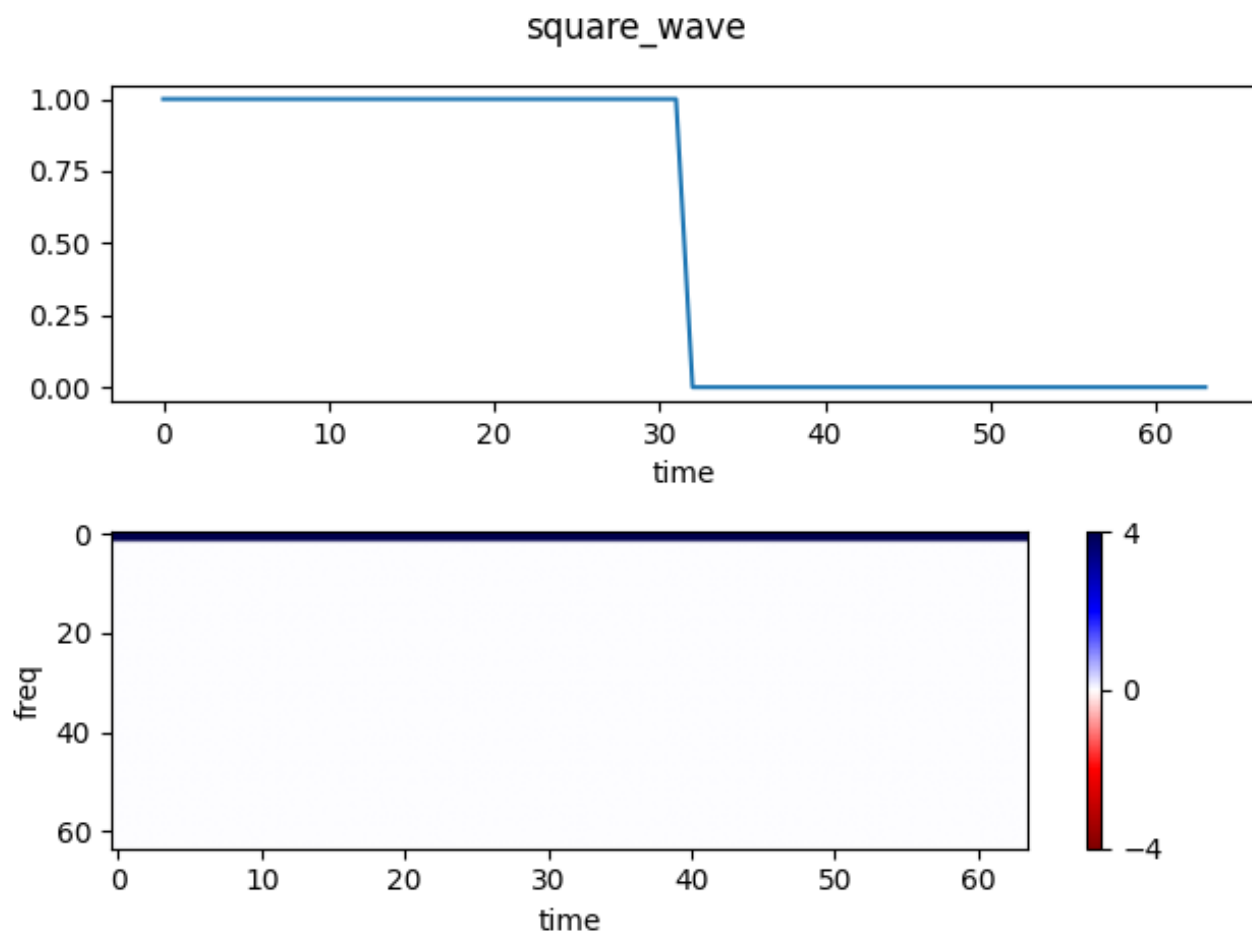
Slow square wave



Slow square wave



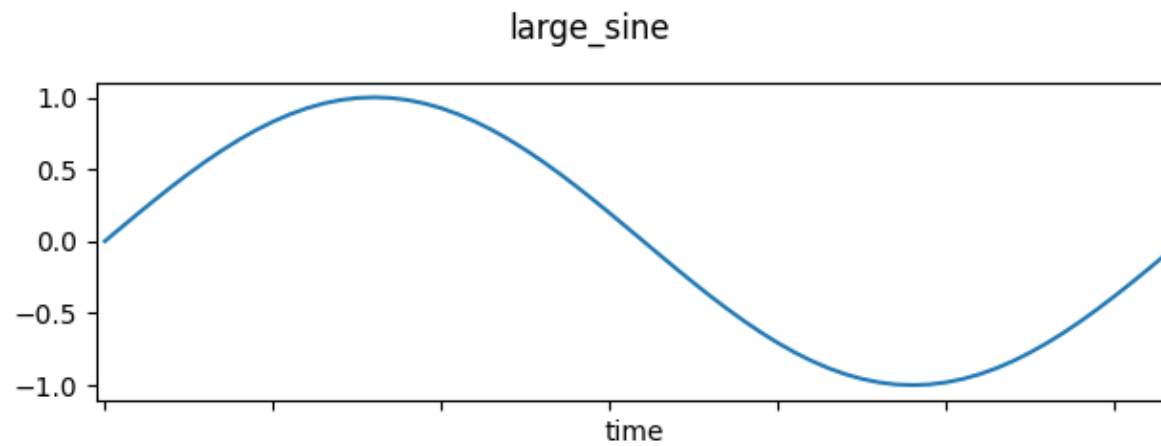
Slow square wave



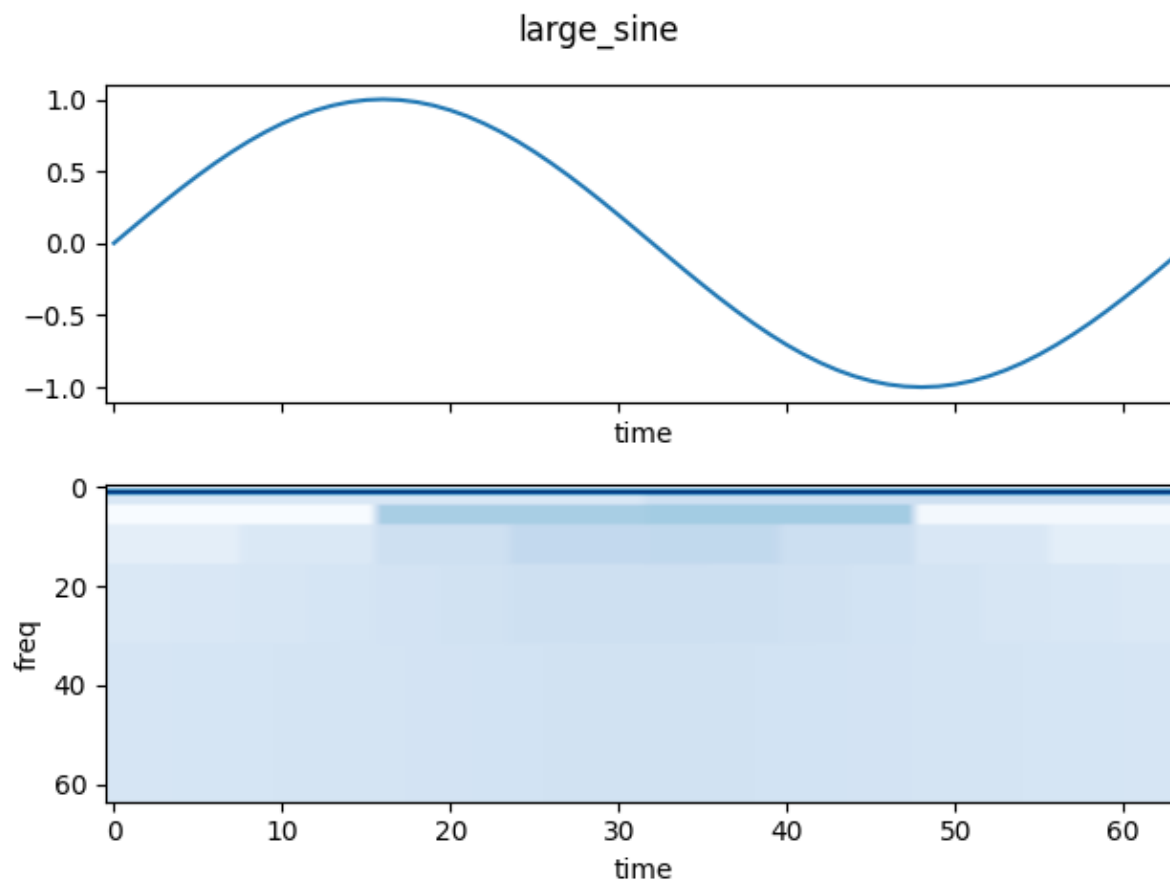
Drills - quantitatively

- ✓ Spike
- ✓ Prolonged spike
- ✓ High-frequency square wave
- ✓ Slow square wave
- Slow sine wave

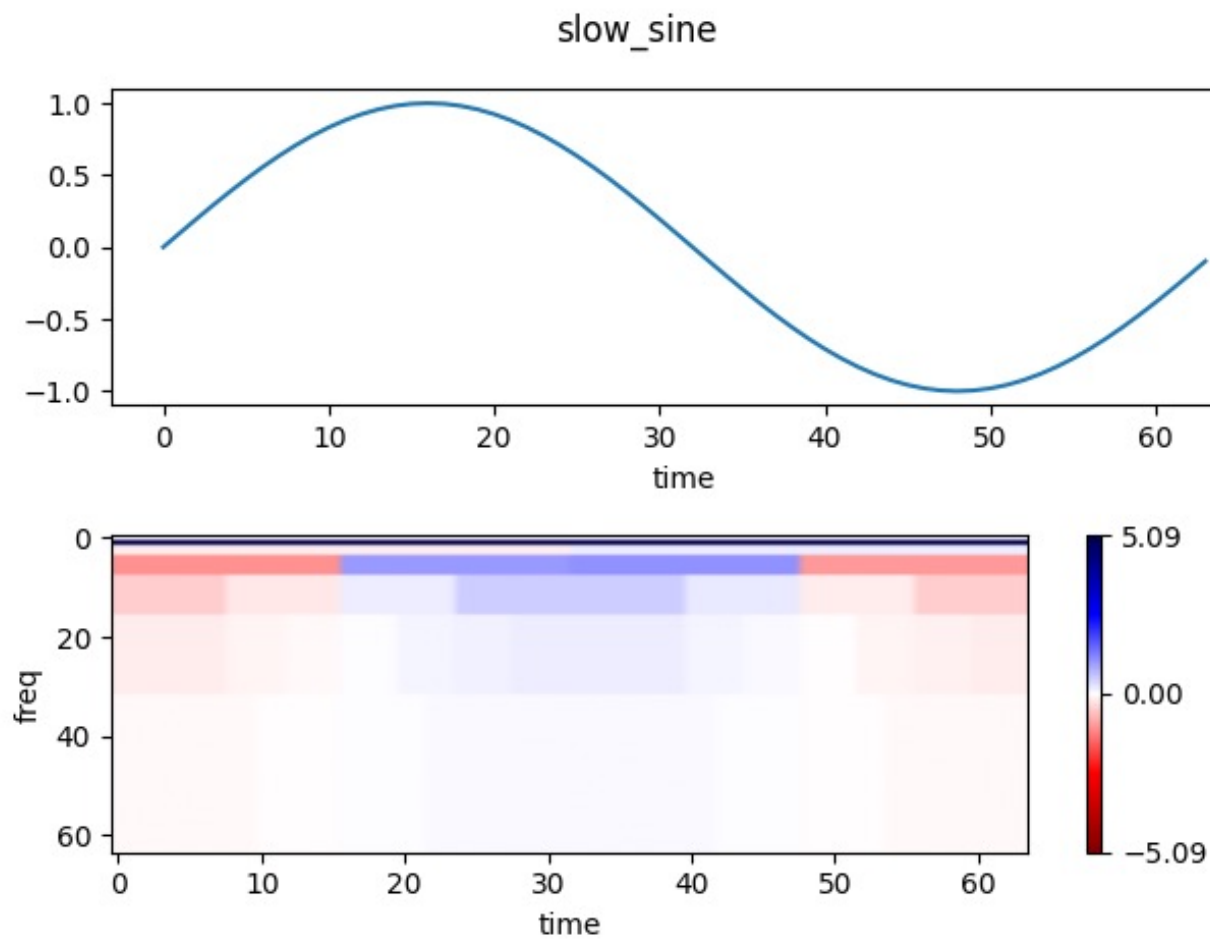
Slow sine



Slow sine



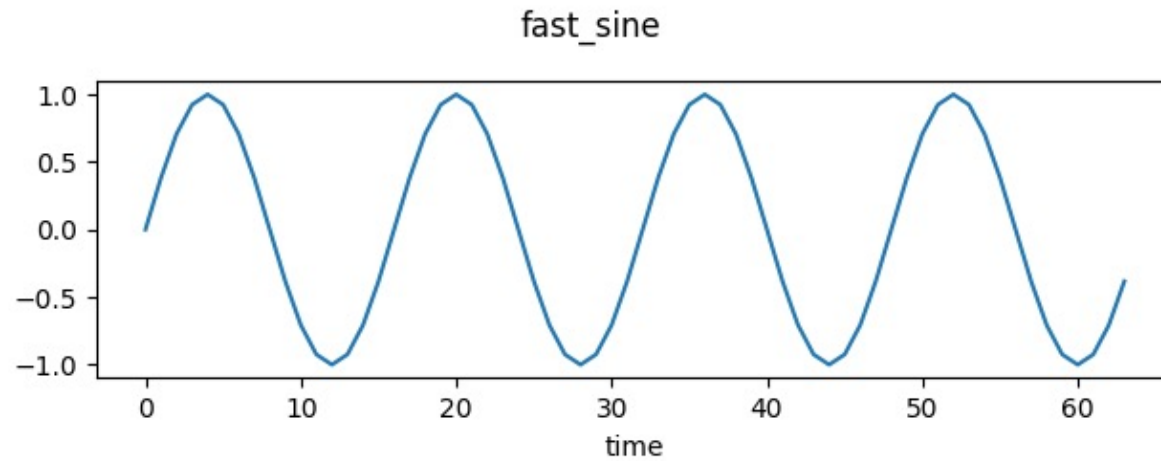
Slow sine



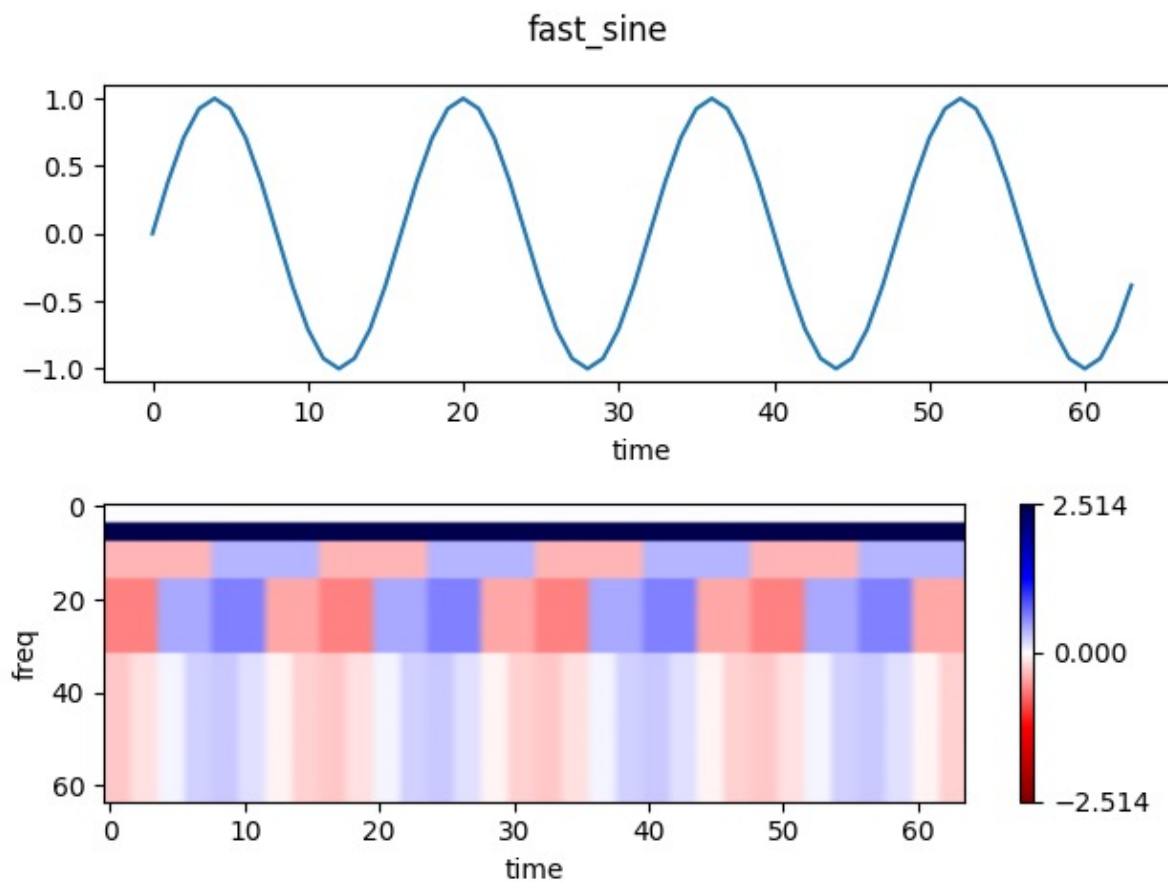
Drills - quantitatively

- ✓ Spike
- ✓ Prolonged spike
- ✓ High-frequency square wave
- ✓ Slow square wave
- ✓ Slow sine wave
- Fast sine wave

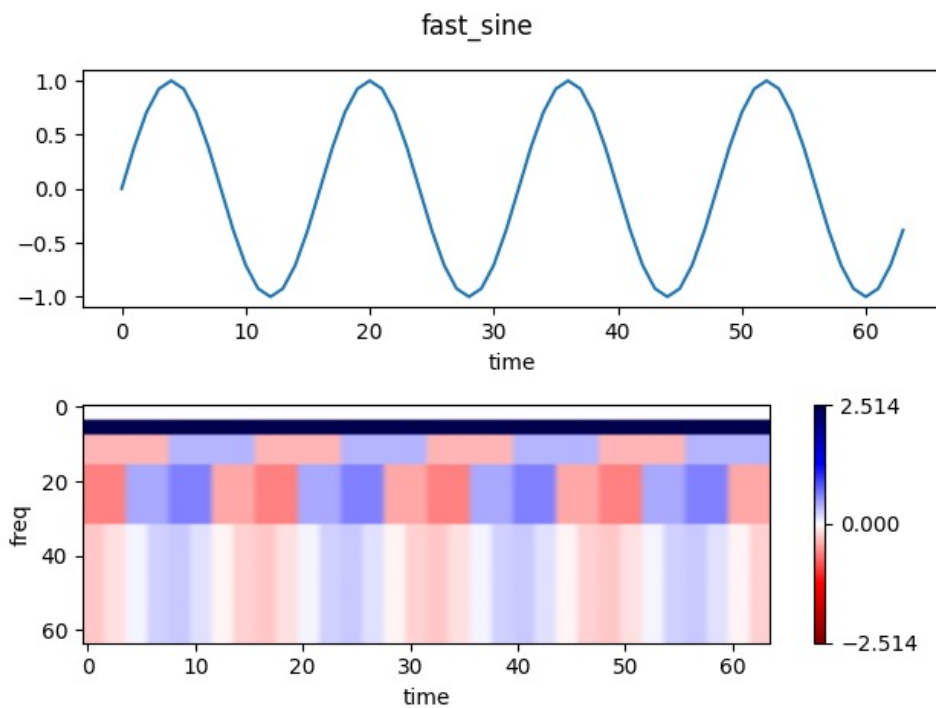
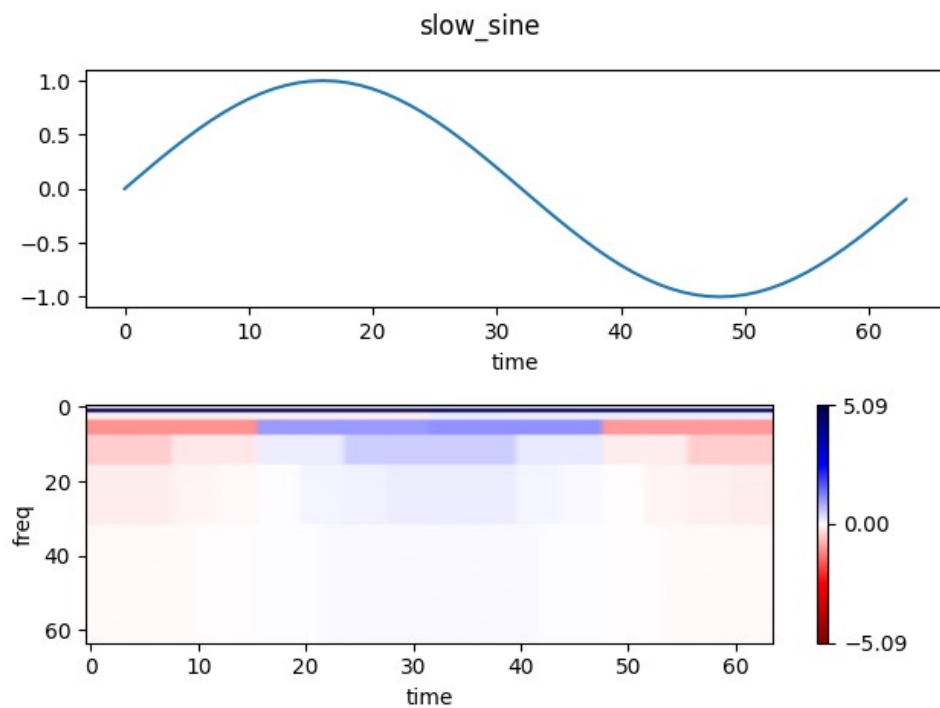
Fast sine



Fast sine



Slow vs Fast sine

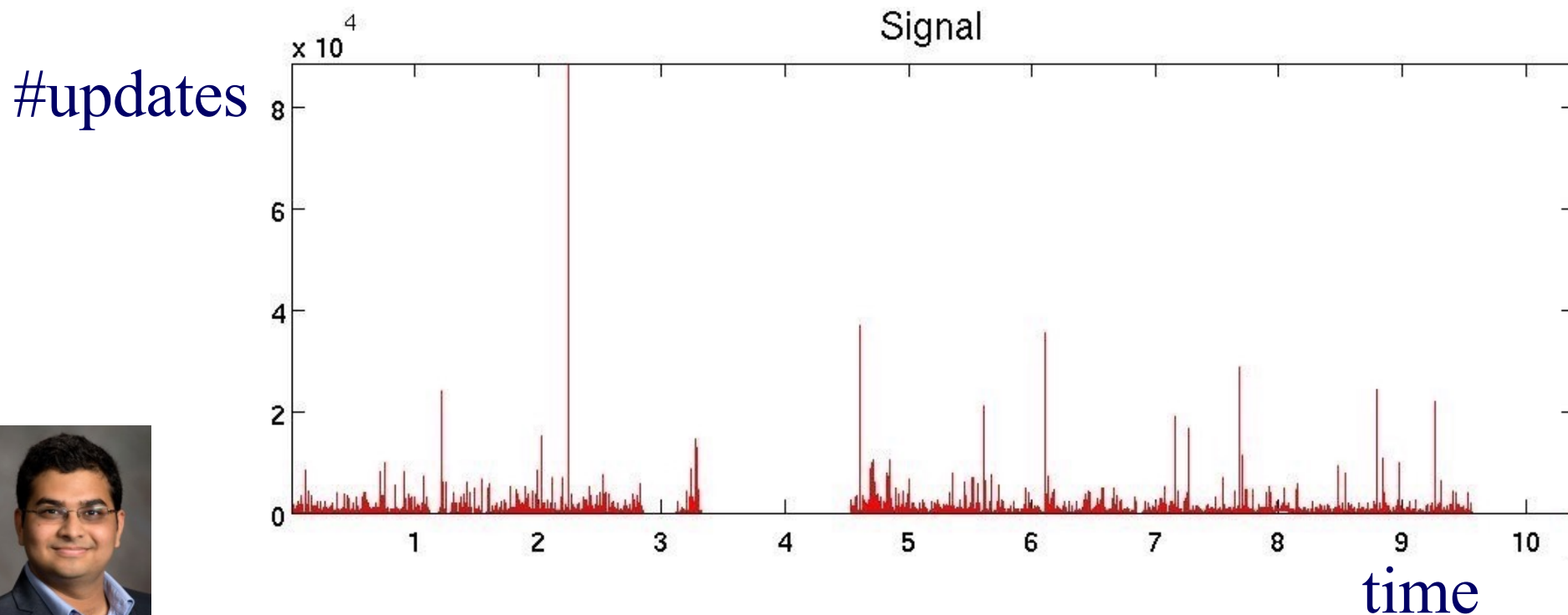




DWT - Detailed outline

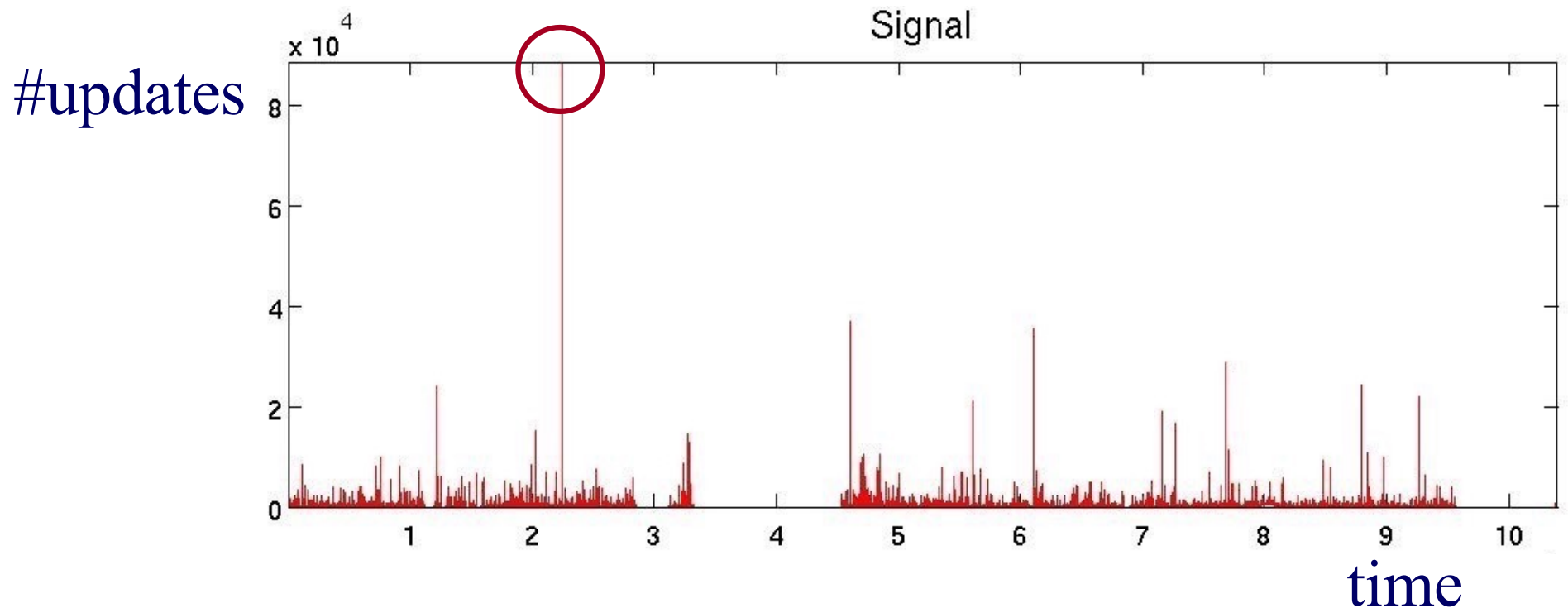
- Motivation
- Definition - Intuition
- Examples and drills
 - vs DFT
 - Qualitative
 - Quantitative
- ➔ • Application: BGP-lens
- 2-d DWT

More examples (BGP updates)



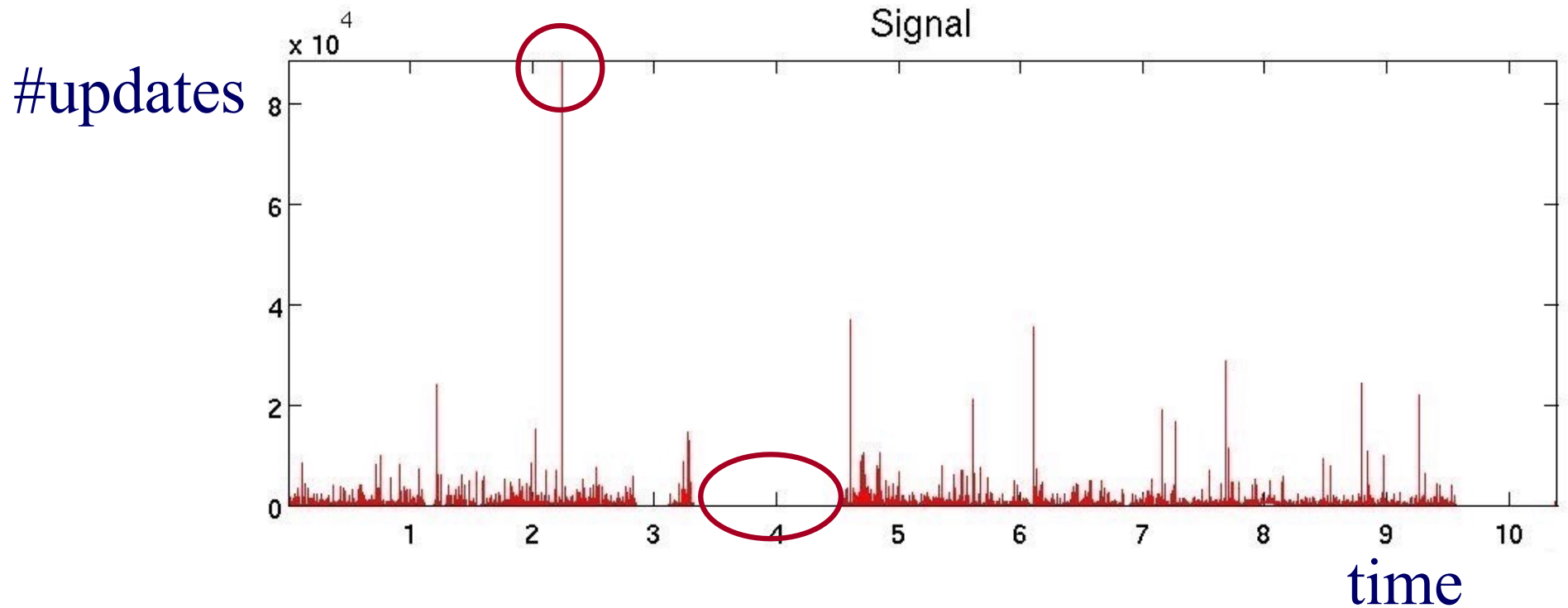
BGP-lens: Patterns and Anomalies in Internet Routing Updates B. Aditya Prakash et al, SIGKDD 2009

More examples (BGP updates)



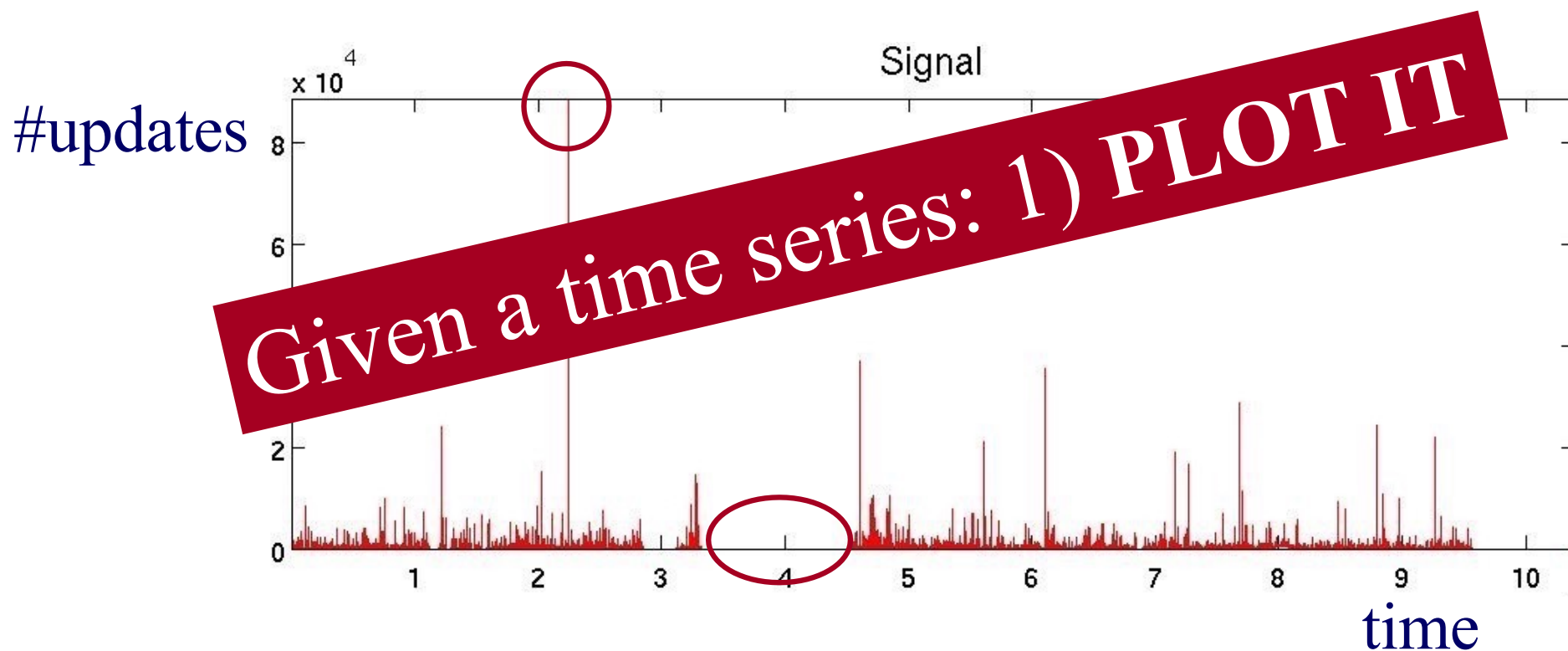
BGP-lens: Patterns and Anomalies in Internet Routing Updates B. Aditya Prakash et al, SIGKDD 2009

More examples (BGP updates)



BGP-lens: Patterns and Anomalies in Internet Routing Updates B. Aditya Prakash et al, SIGKDD 2009

More examples (BGP updates)



BGP-lens: Patterns and Anomalies in Internet Routing Updates B. Aditya Prakash et al, SIGKDD 2009

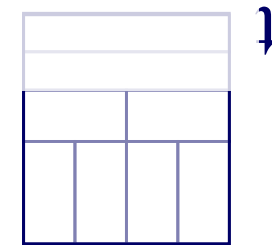
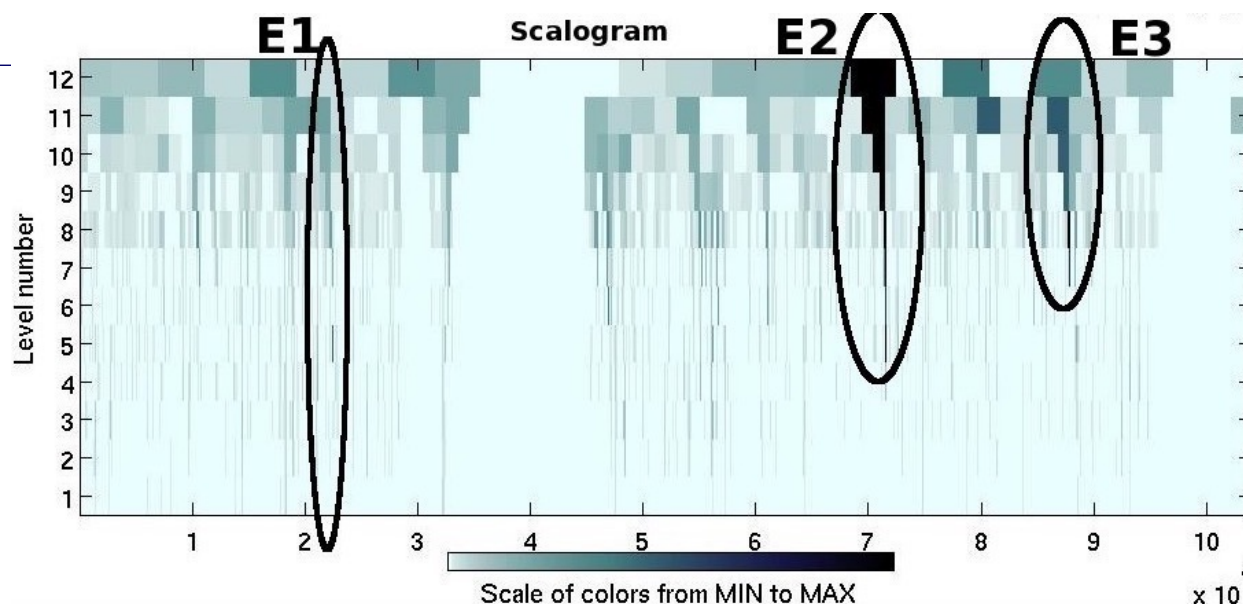
More examples (BGP updates)

Given a time series: 1) PLOT IT
2) DFT / DWT

More examples (BGP updates)

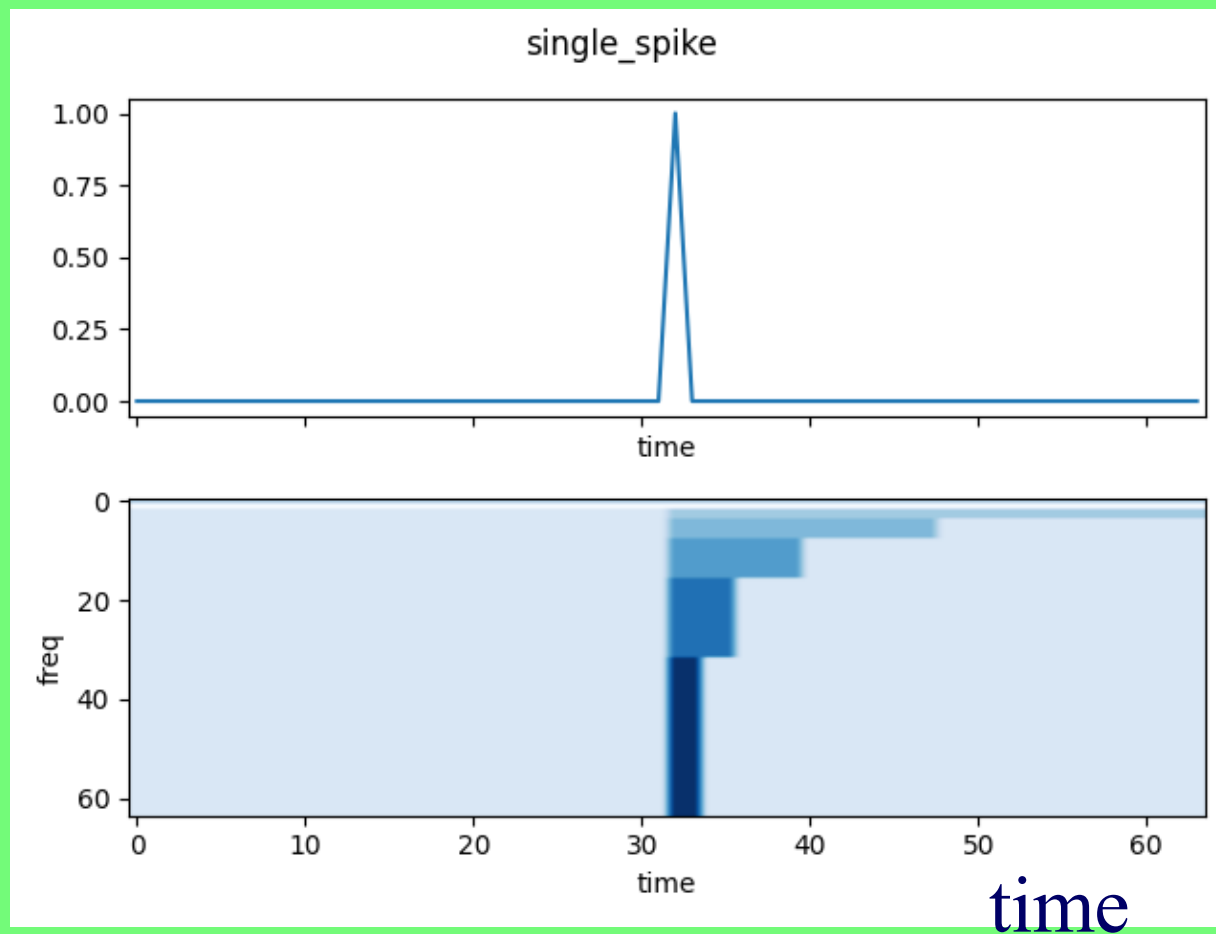
Low freq.:
omitted

freq.





Spike



freq.

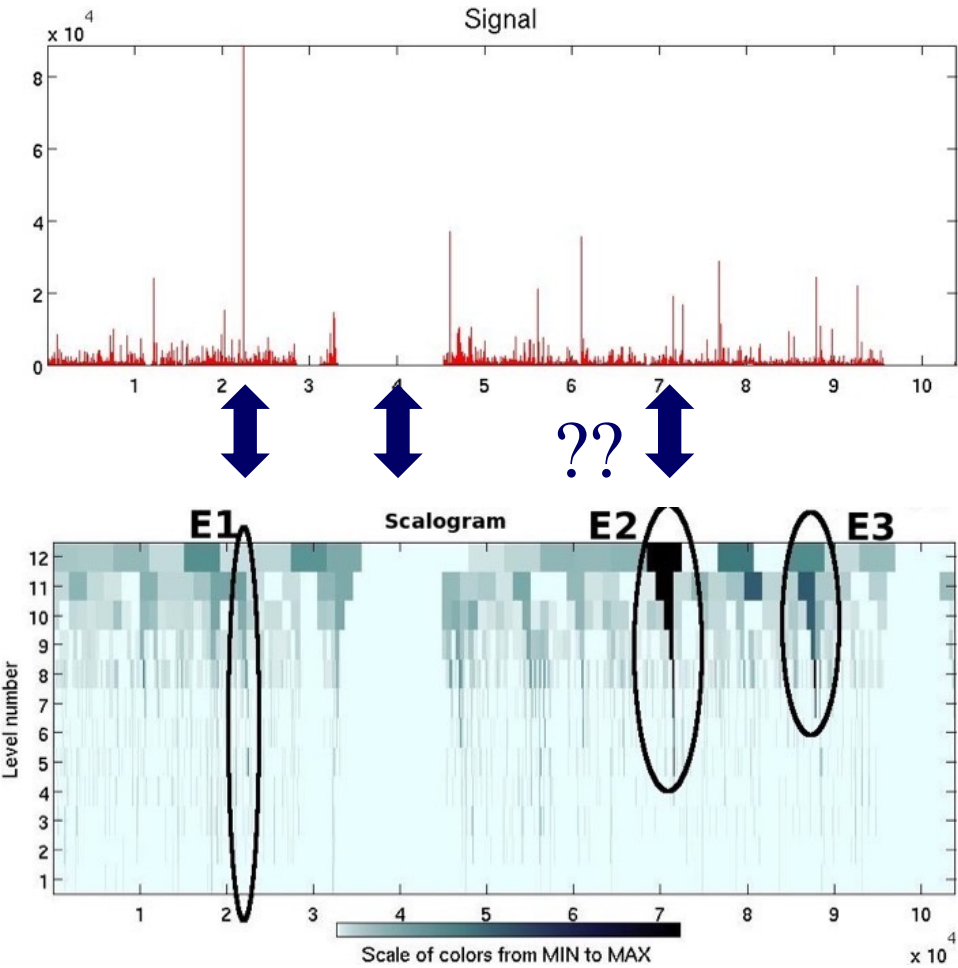


time



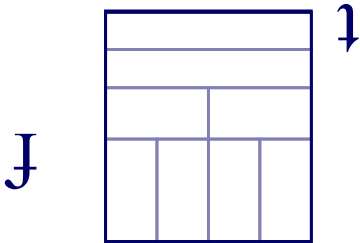
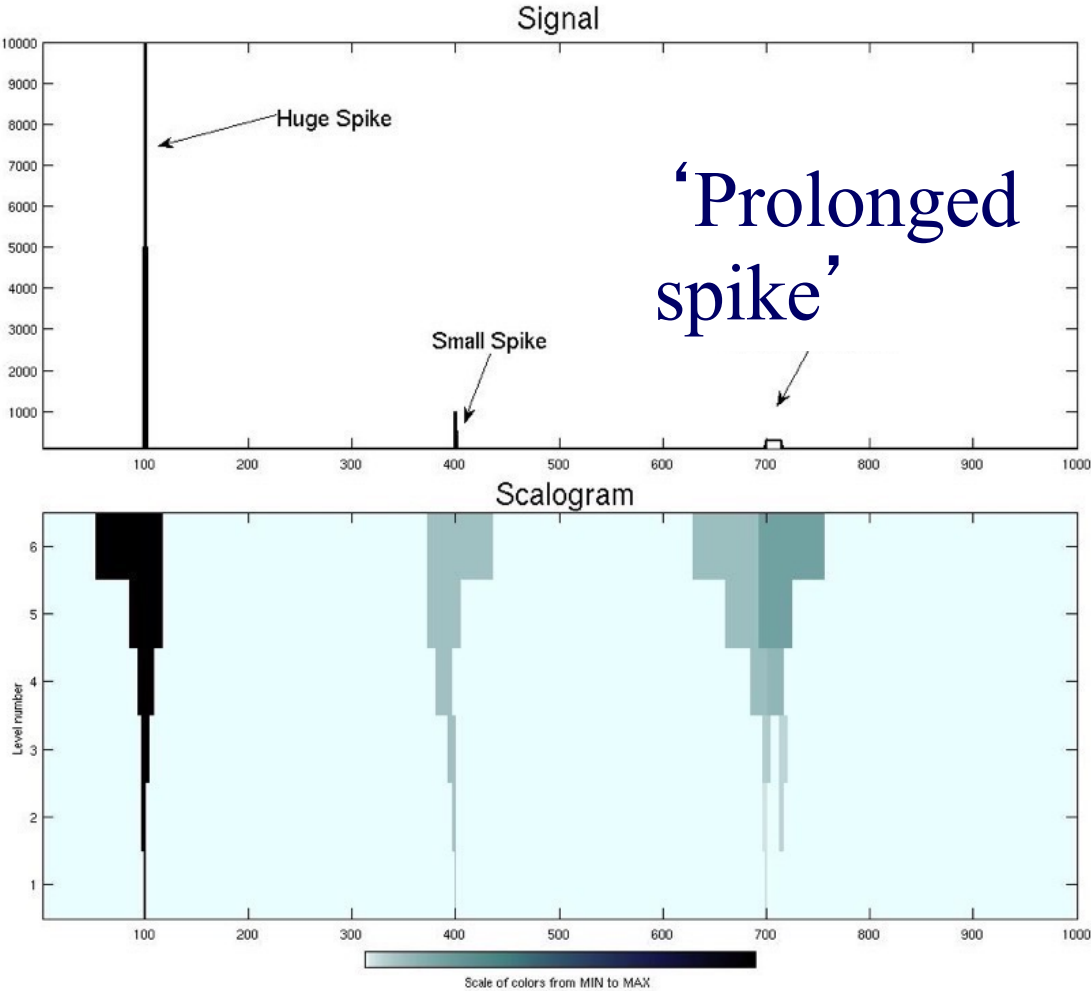
More examples (BGP updates)

freq. ↓



More examples (BGP updates)

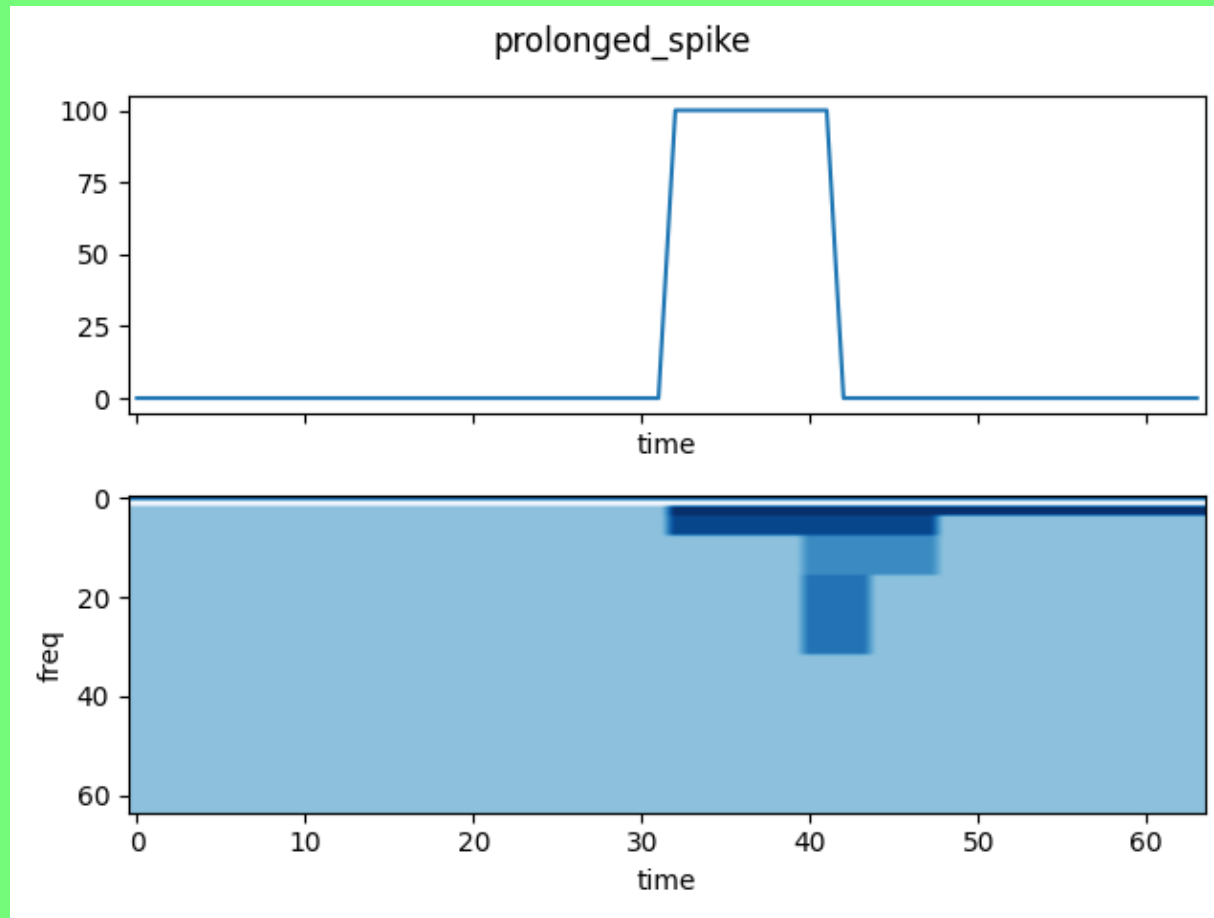
freq. ↓



Keep it in mind

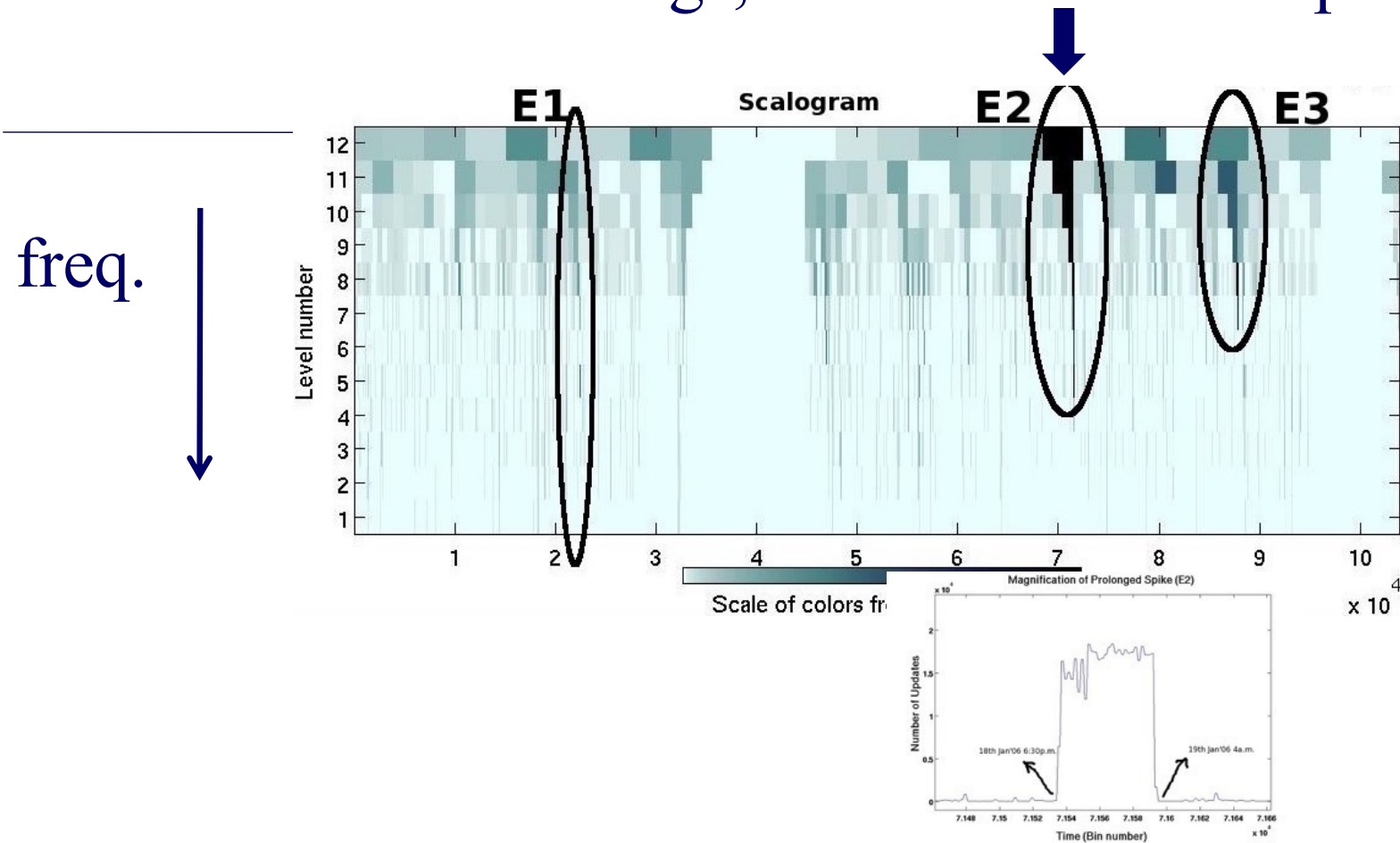


Prolonged spike



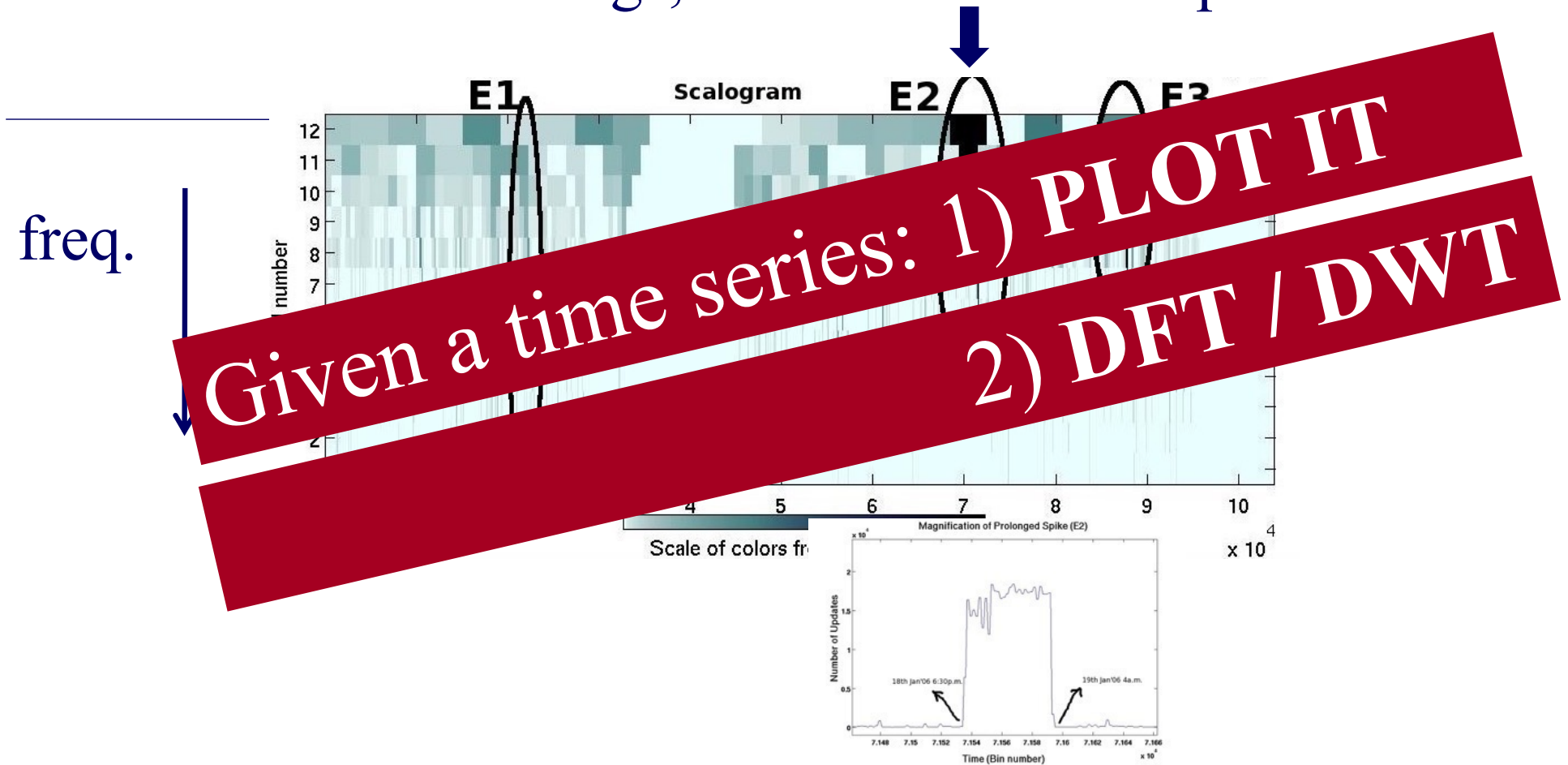
More examples (BGP updates)

15K msgs, for several hours: 6pm-4am



More examples (BGP updates)

15K msgs, for several hours: 6pm-4am



Wavelets - Drill

- Or use 'R', 'octave' or 'matlab' – R:

```
install.packages("wavelets")
```

```
library("wavelets")
```

```
X1<-c(1,2,3,4,5,6,7,8)
```

```
dwt(X1, n.levels=3, filter="d4")
```

```
mra(X1, n.levels=3, filter="d4")
```



DWT - Detailed outline

- Motivation
- Definition - Intuition
- Examples and drills
 - vs DFT
 - Qualitative
 - Quantitative
- Application: BGP-lens
- ➔ • 2-d DWT

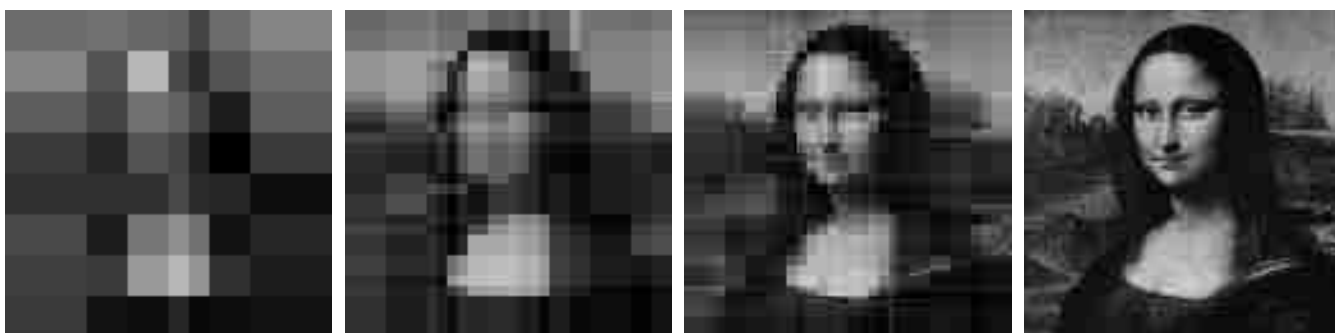
Wavelets - k-dimensions?

- easily defined for any dimensionality (like DFT, DCT)

Wavelets - example

<http://grail.cs.washington.edu/projects/query/>

Wavelets achieve *great* compression:



20

100

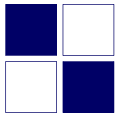
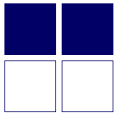
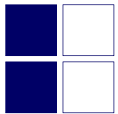
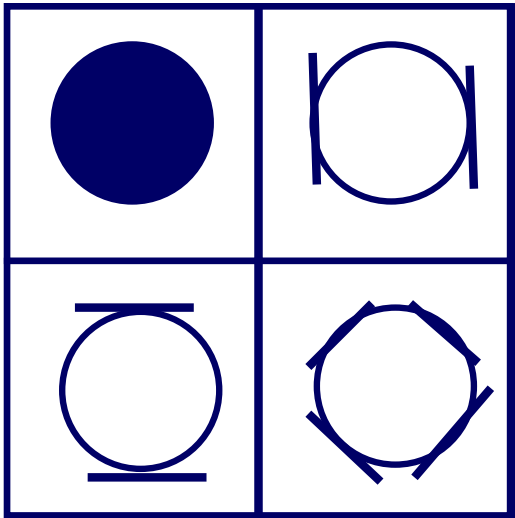
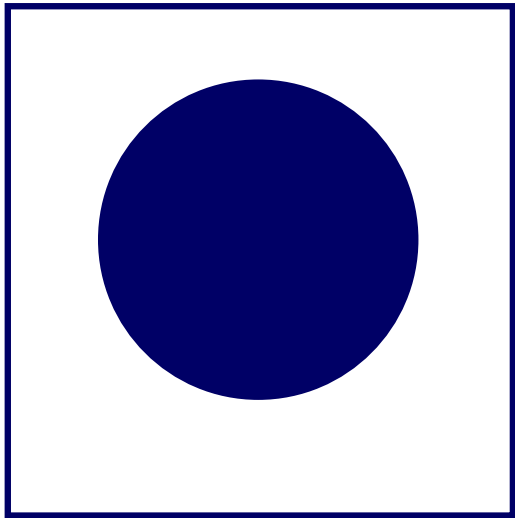
400

16,000

coefficients

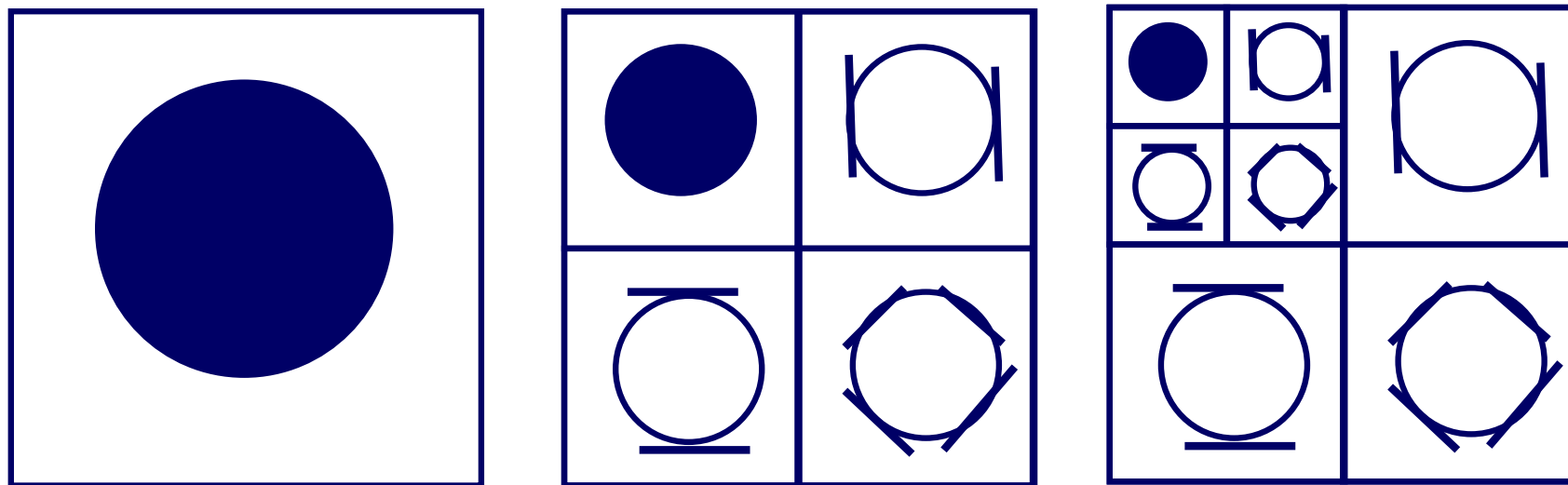
Wavelets - intuition

- Edges (horizontal; vertical; diagonal)



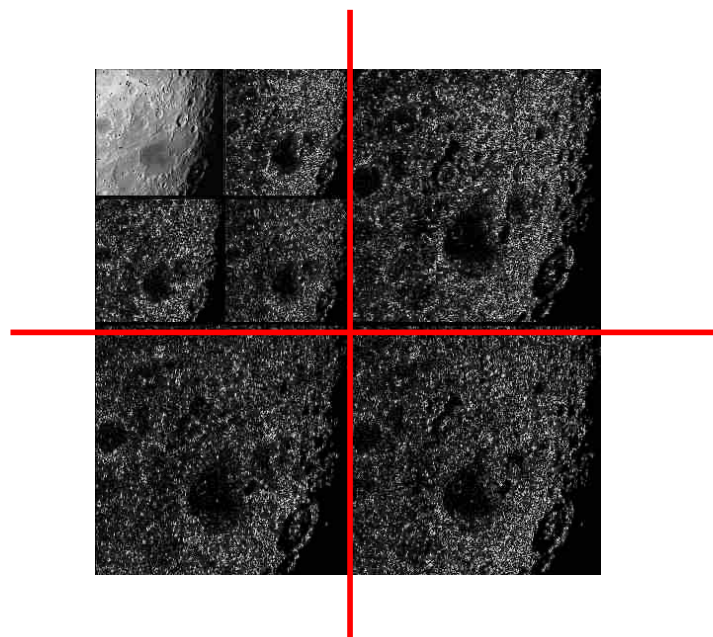
Wavelets - intuition

- Edges (horizontal; vertical; diagonal)
- recurse



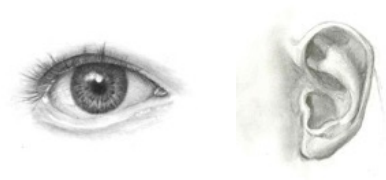
Wavelets - intuition

- Edges (horizontal; vertical; diagonal)
- <http://www331.jpl.nasa.gov/public/wave.html>



Advantages of Wavelets

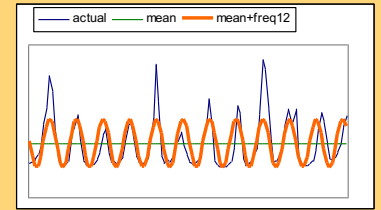
- Better compression (better RMSE with same number of coefficients)
- closely related to the processing of the mammalian eye and ear
- Good for progressive transmission
- handle spikes well
- usually, fast to compute ($O(n)$!)





Overall Conclusions

- DFT spots periodicities
- **DWT** : multi-resolution - matches processing of mammalian ear/eye better
- Both: powerful tools for **compression**, **pattern detection** in real signals
- Both: included in math packages (matlab, R, python)





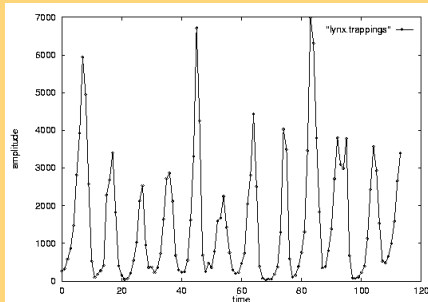
Solutions:

Goal: given a signal (eg., sales over time and/or space)

Q: Find patterns and/or compress

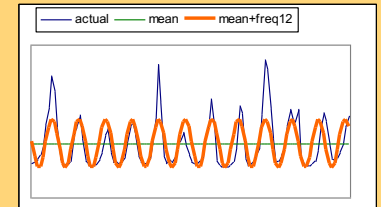


count

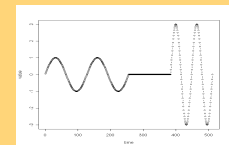
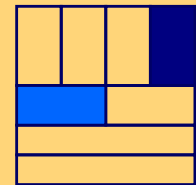


year

✓ A1: Fourier (DFT)



✓ A2: Wavelets (DWT)





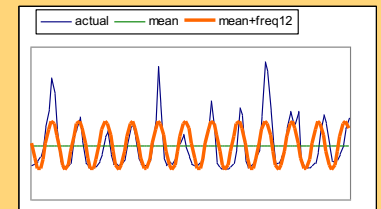
Solutions:

Goal: given a signal (eg., sales over time and/or space)

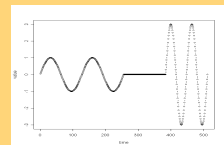
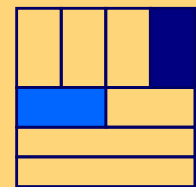
Q: Find patterns and/or compress

A0: PLOT IT

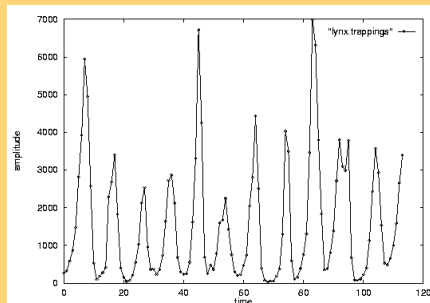
✓ A1: Fourier (DFT)



✓ A2: Wavelets (DWT)



count



year

Resources

- Numerical Recipes in C: great description, intuition and code for all three tools
- *xwpl*: open source wavelet package from Yale, with excellent GUI.

Resources (cont' d)

- [Rice Univ. \(wavelets and other demos\)](#)
- R ('install.packages("wavelets"))
- Python: [PyWavelets](#)
- https://www.cs.cmu.edu/%7Echristos/SRC/haar_demo_all.tar