The Sliding Phase Vocoder

Russell Bradford Richard Dobson John ffitch

Media Research Centre
Department of Computer Science
University of Bath

ICMC Copenhagen, Aug 2007

What Moorer Said

In *Audio in the New Millennium* (JAES May 2000) James Moorer said:

> *Consider for one second what it means if the next 22 years gives anywhere near the same speedup as the previous 22 years has given.*

A few predictions from Moorer:

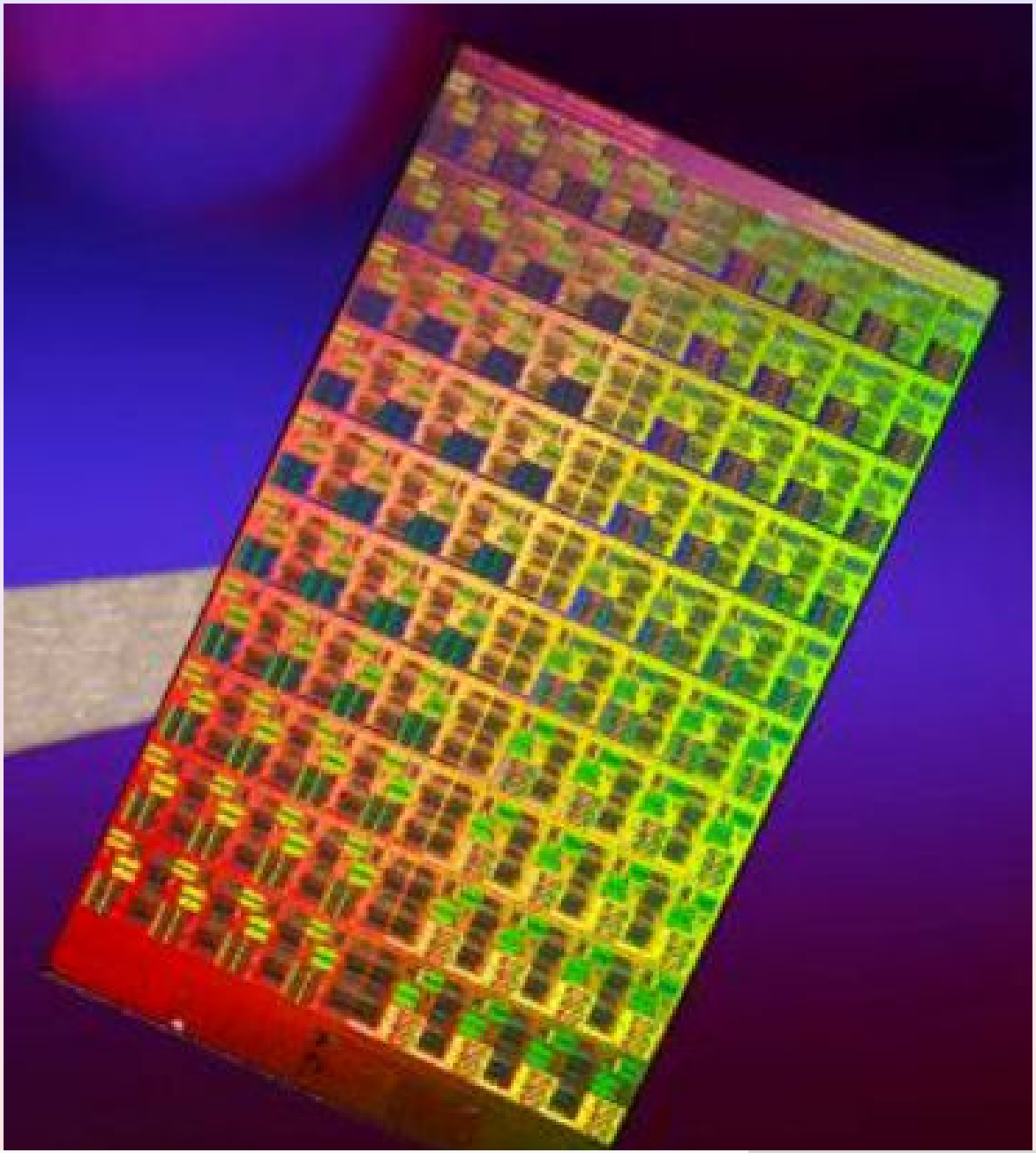
- Process 9000 channels in real time!
- Processes that were considered unfeasible...will become matter-of-fact
- Need for intelligent agents to manage 1M virtual parameters
- In 20 years, loudspeakers and microphones will know where they are.

Where are we today?

- High clock speed = high power consumption
- Multiple CPU $\rightarrow$ multiple cores — may outrun Moore's Law
- Parallel algorithms require parallel hardware
- Intel "Polaris" - 80 dual-floating point cores, 1.1TFlops on one chip ETA 10 years
- Clearspeed "Advance" - 66GFlops (64bit) @ 25 Watts, in use today
- Intel already emphasizing applications in multi-media
- Hardware expensive now, but may become commodity chip for audio acceleration
- Parallel extensions to C - new languages?

UNIVERSITY OF BATH

Intel Polaris 80-core



ClearSpeed Accelerator

Where are we today? (cont)

- Running transforms — the Sliding DFT
- “There is some advantage to be gained by computing the transform at every sample”
- an example ; the “ultimate sampling synthesiser”
- implication: by 2020 processing audio in the frequency domain will become routine
- compute power formidable: analysis rate = sample rate
- our interest is in interactive real-time deployment

The Sliding DFT

See for example Bradford, Dobson and Fitch, ICMC2005 Barcelona.

- Encoding: simple complex rotation - basic SDFT quite economical

$$F_{t+1}(n) = (F_t(n) - f_t + f_{t+N})\, e^{2\pi i \frac{n}{N}}$$

- SDFT highly parallelizable - given the hardware
- analysis rate = sample rate so transformations inflate CPU load
- analysis: frequency-domain windowing by convolution

The Sliding DFT

- resynthesis: sum over all bins = oscillator bank

$$f_t = \frac{1}{N} \sum_{j=0}^{N-1} F_t(j) e^{-2\pi i j t / N}$$

but for a single point $t = 0$ this simplifies to

$$\frac{1}{N} \sum_{j=0}^{N-1} F_0(j)$$

The SPV compared to PVOC

There are significant differences between the classic PVOC and the Sliding Phase Vocoder (SPV):

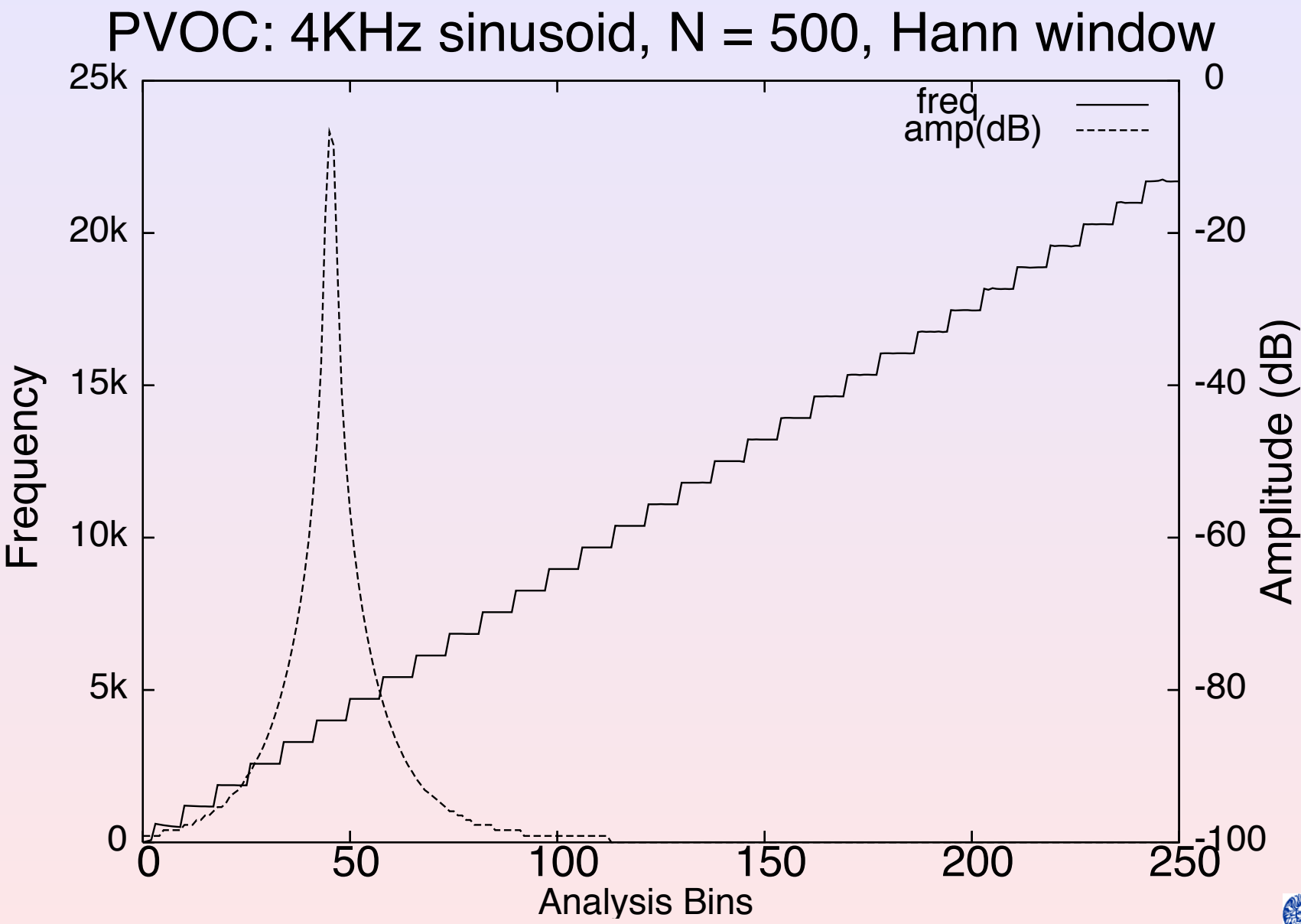
- conversion to amplitude/frequency the same
- (our SPV based on example by Stephan Sprenger (`http://www.dspdimension.org`)
- Bin bandwidth: pvoc = $\pm$ SR/N; SPV = $\pm$ Nyquist
- so in SPV: any frequency in any bin!
- pitch scaling and frequency shifting fiddly in PVOC, trivial in SPV
- double precision floating point required as the phase updates every sample

The SPV analysis frame (amplitude/frequency)

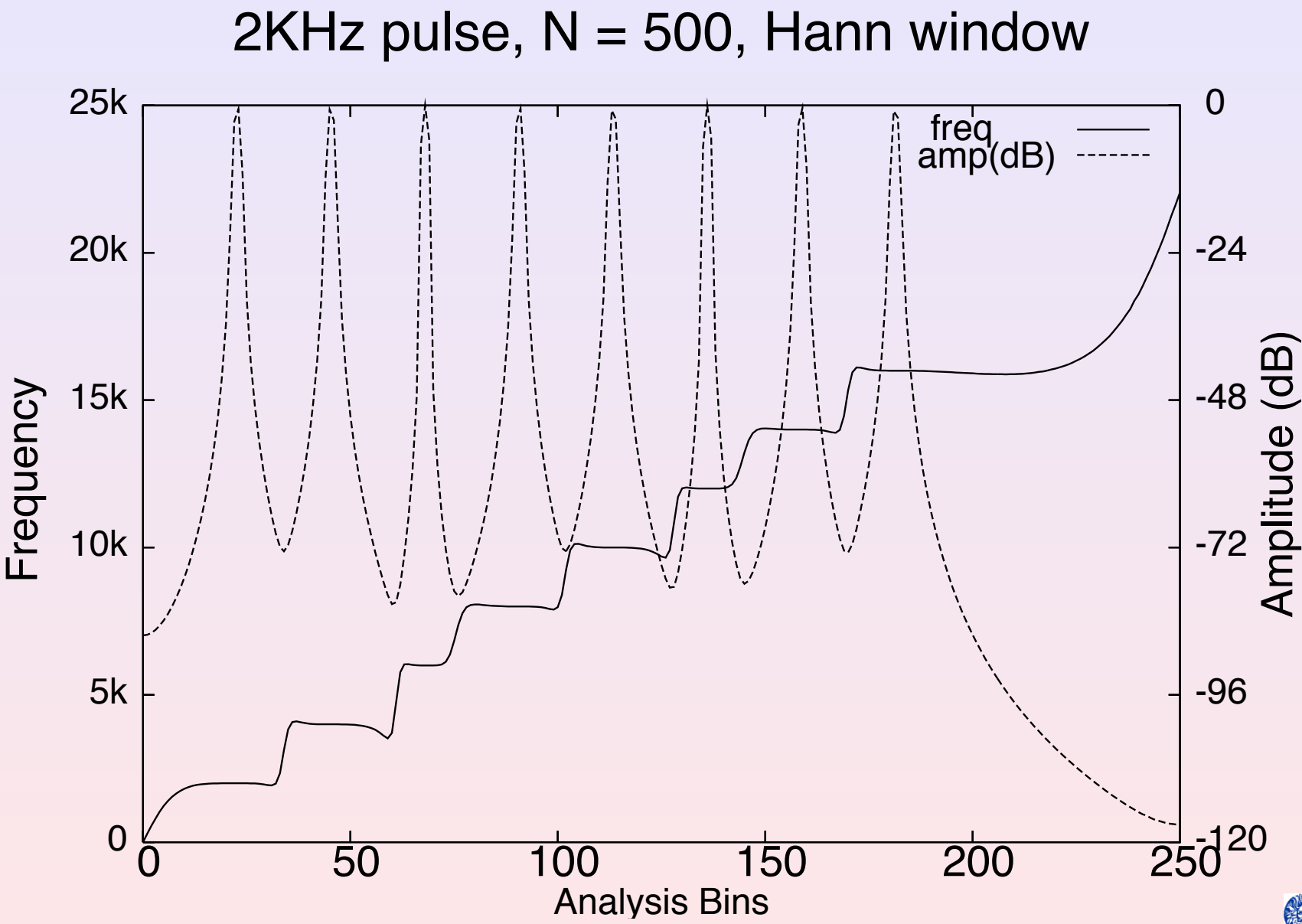
- Basic property: frequency bunching at a peak
- pvoc — constrained by narrow bin bandwidth
- SPV: bins free to converge on peak
- extraordinary bin convergence with single input sinusoid
- bin frequency map gives new view of windowing

UNIVERSITY OF BATH

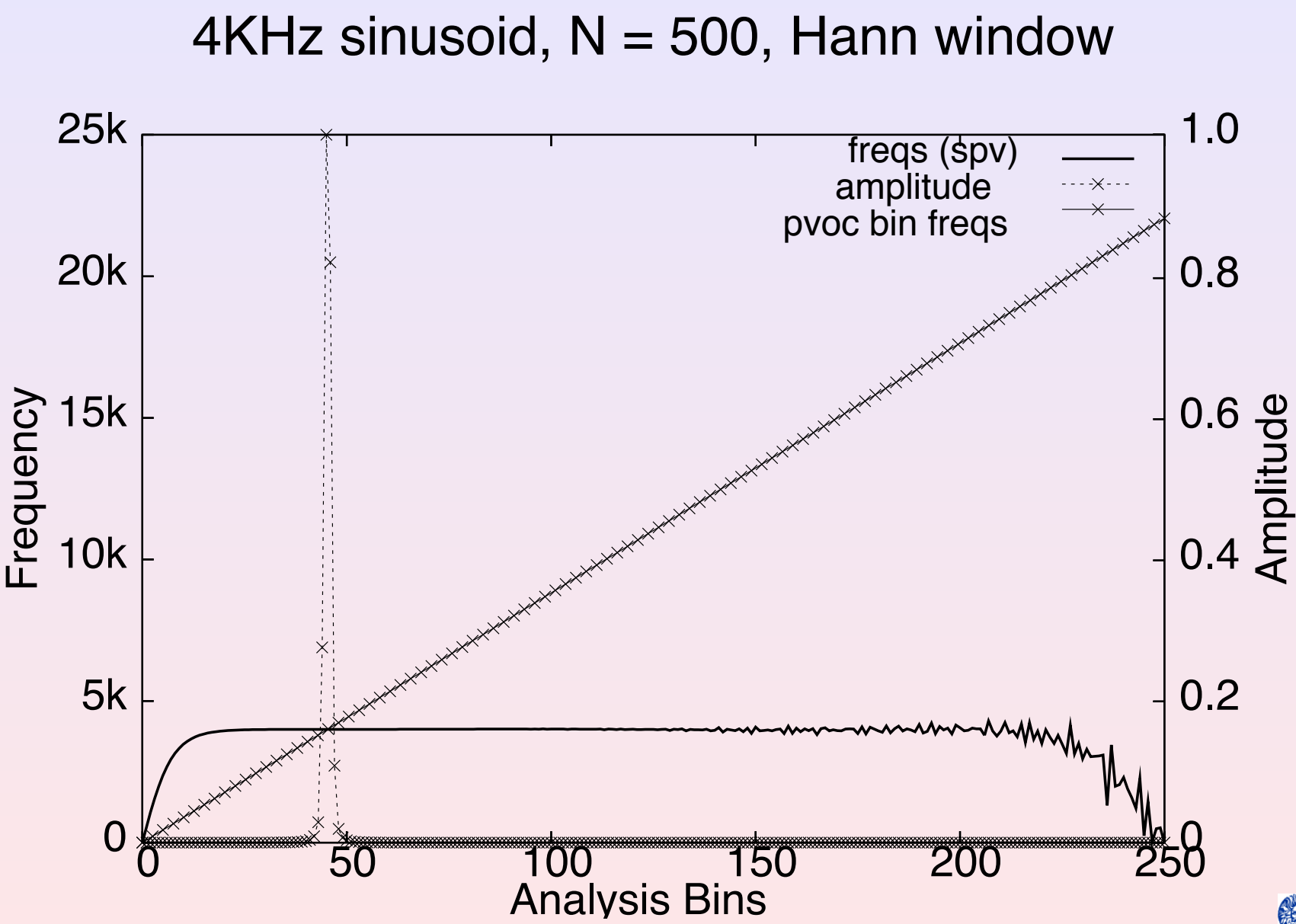
pvoc — constrained by narrow bin bandwidth



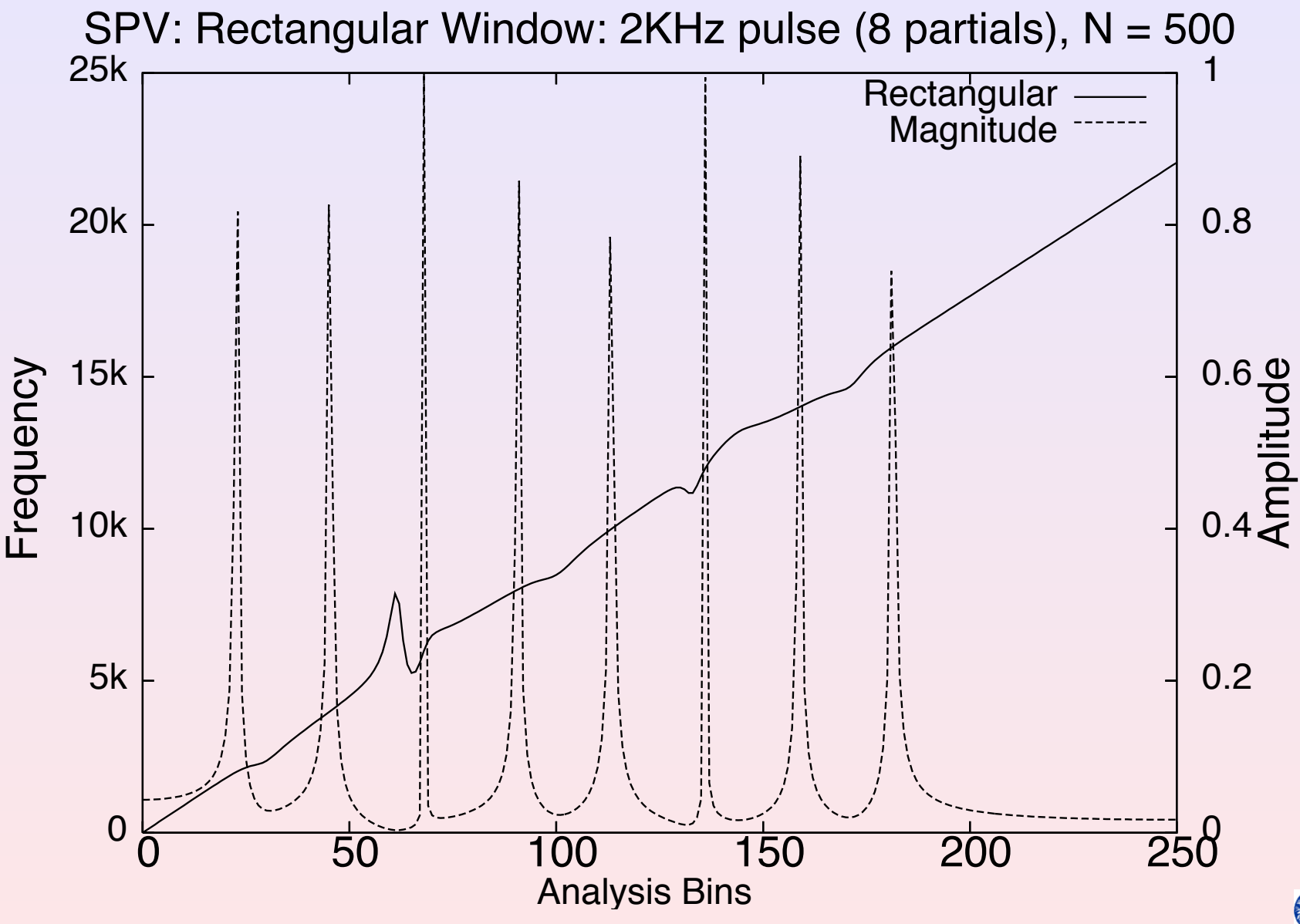
SPV: bins free to converge on peak



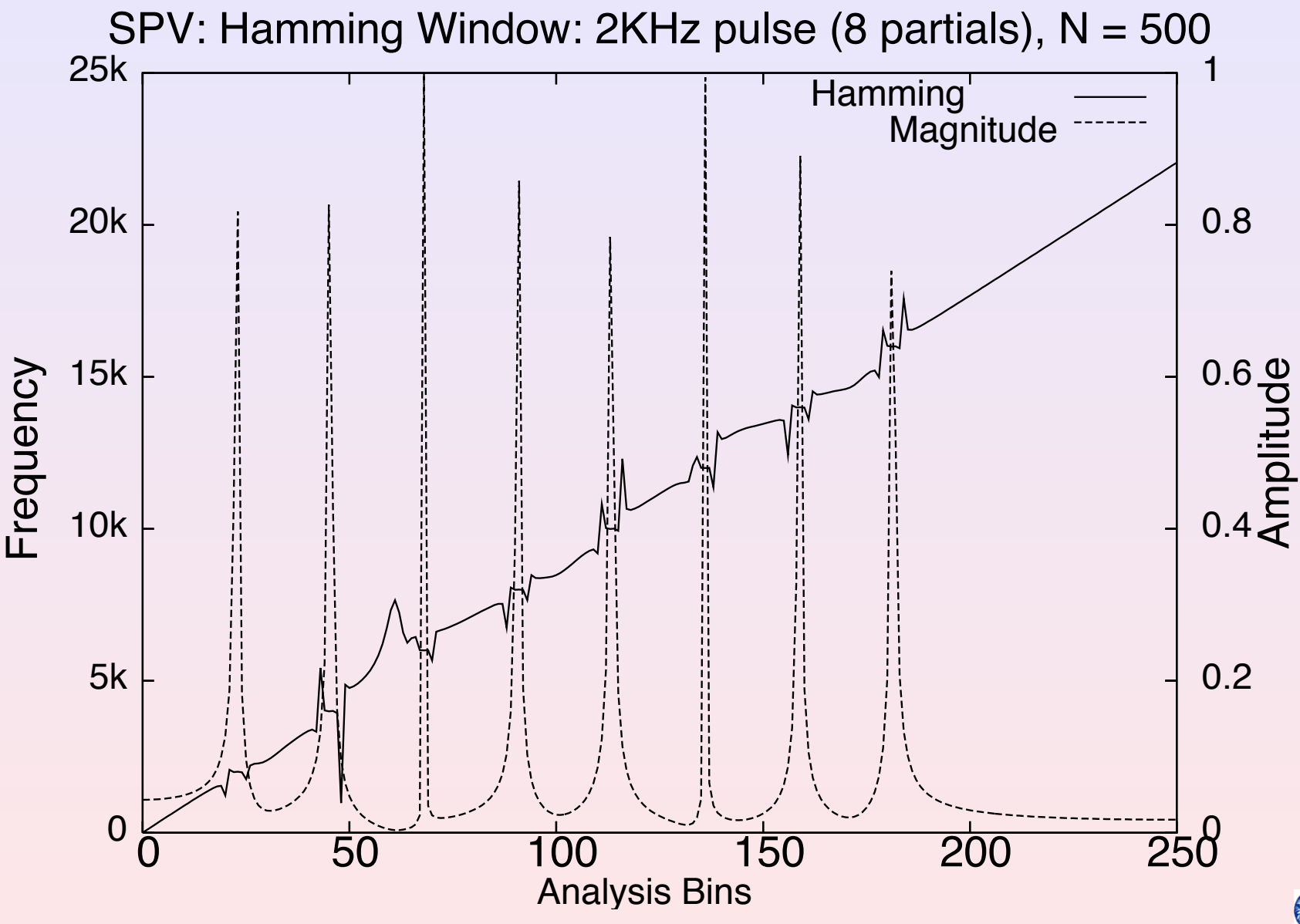
Bin convergence



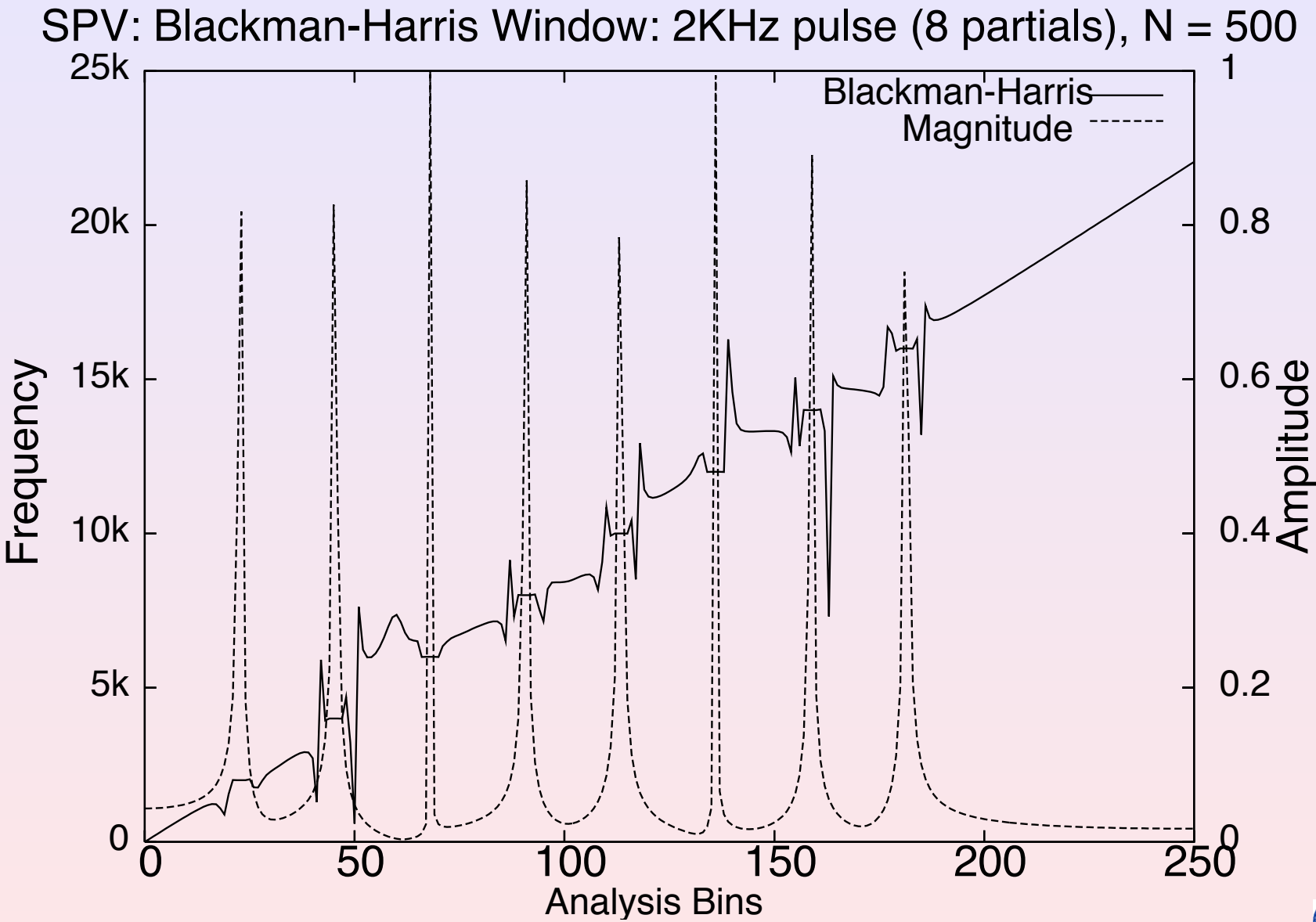
Rectangular Window



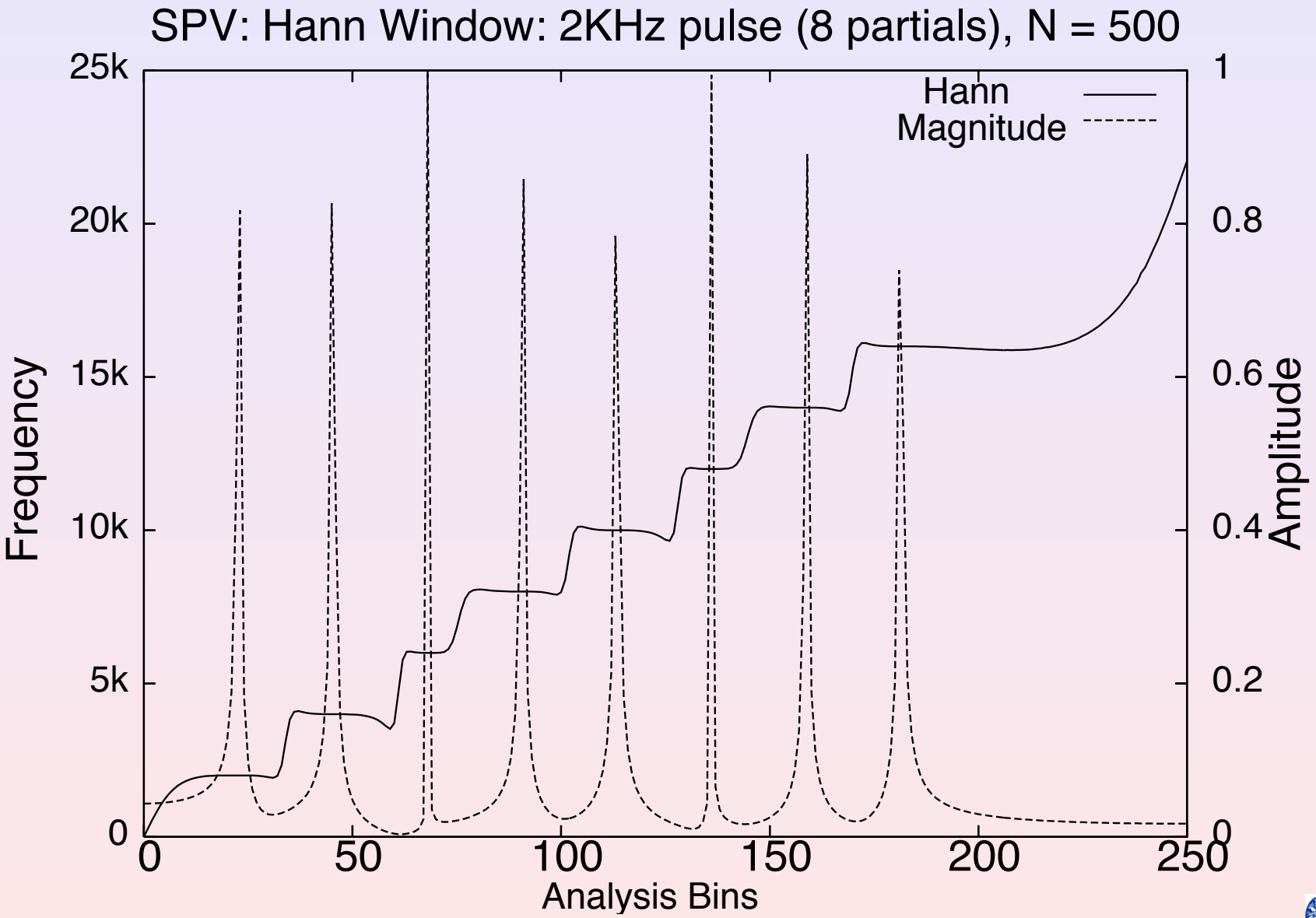
Hamming Window



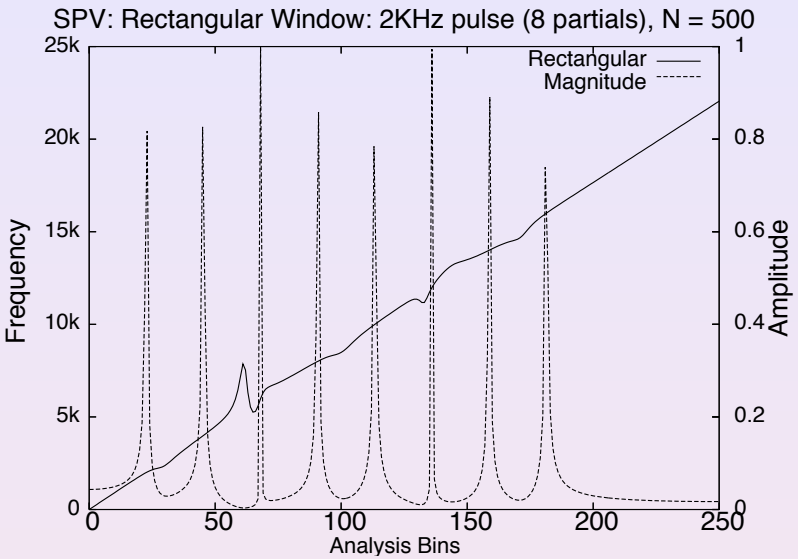
Blackman-Harris Window

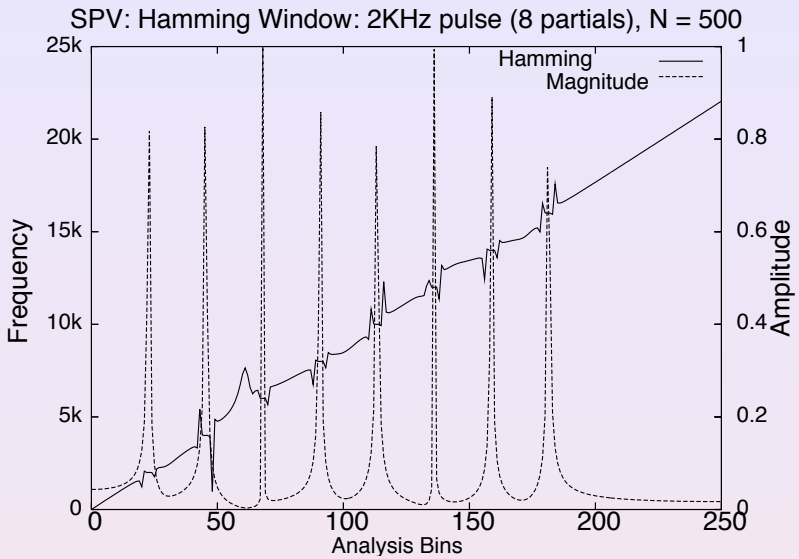


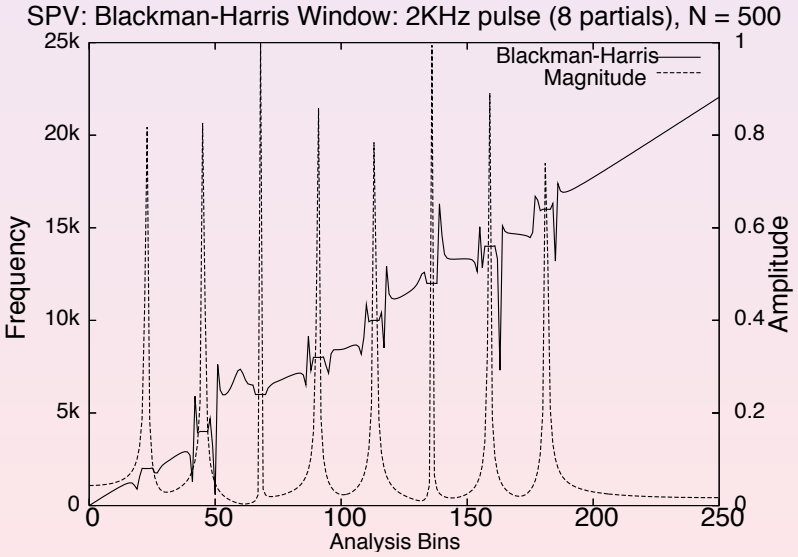
von Hann Window

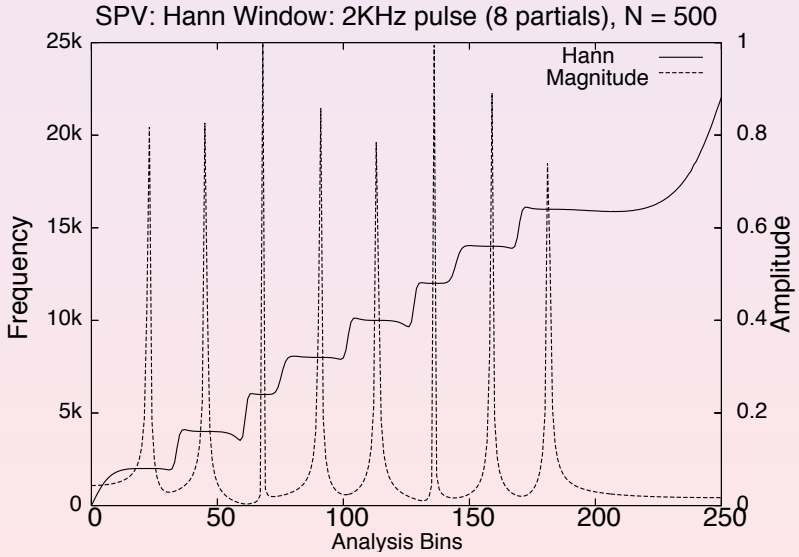


Windowing


SPV: Rectangular Window: 2KHz pulse (8 partials), N = 500
Rectangular
Magnitude
Frequency
Amplitude
Analysis Bins
25k
20k
15k
10k
5k
0
1
0.8
0.6
0.4
0.2
0
50
100
150
200
250


SPV: Hamming Window: 2KHz pulse (8 partials), N = 500
Hamming
Magnitude
Frequency
Amplitude
Analysis Bins
25k
20k
15k
10k
5k
0
1
0.8
0.6
0.4
0.2
0
50
100
150
200
250


SPV: Blackman-Harris Window: 2KHz pulse (8 partials), N = 500
Blackman-Harris
Magnitude
Frequency
Amplitude
Analysis Bins
25k
20k
15k
10k
5k
0
1
0.8
0.6
0.4
0.2
0
50
100
150
200
250


SPV: Hann Window: 2KHz pulse (8 partials), N = 500
Hann
Magnitude
Frequency
Amplitude
Analysis Bins
25k
20k
15k
10k
5k
0
1
0.8
0.6
0.4
0.2
0
50
100
150
200
250

The SPV analysis frame (amplitude/frequency)

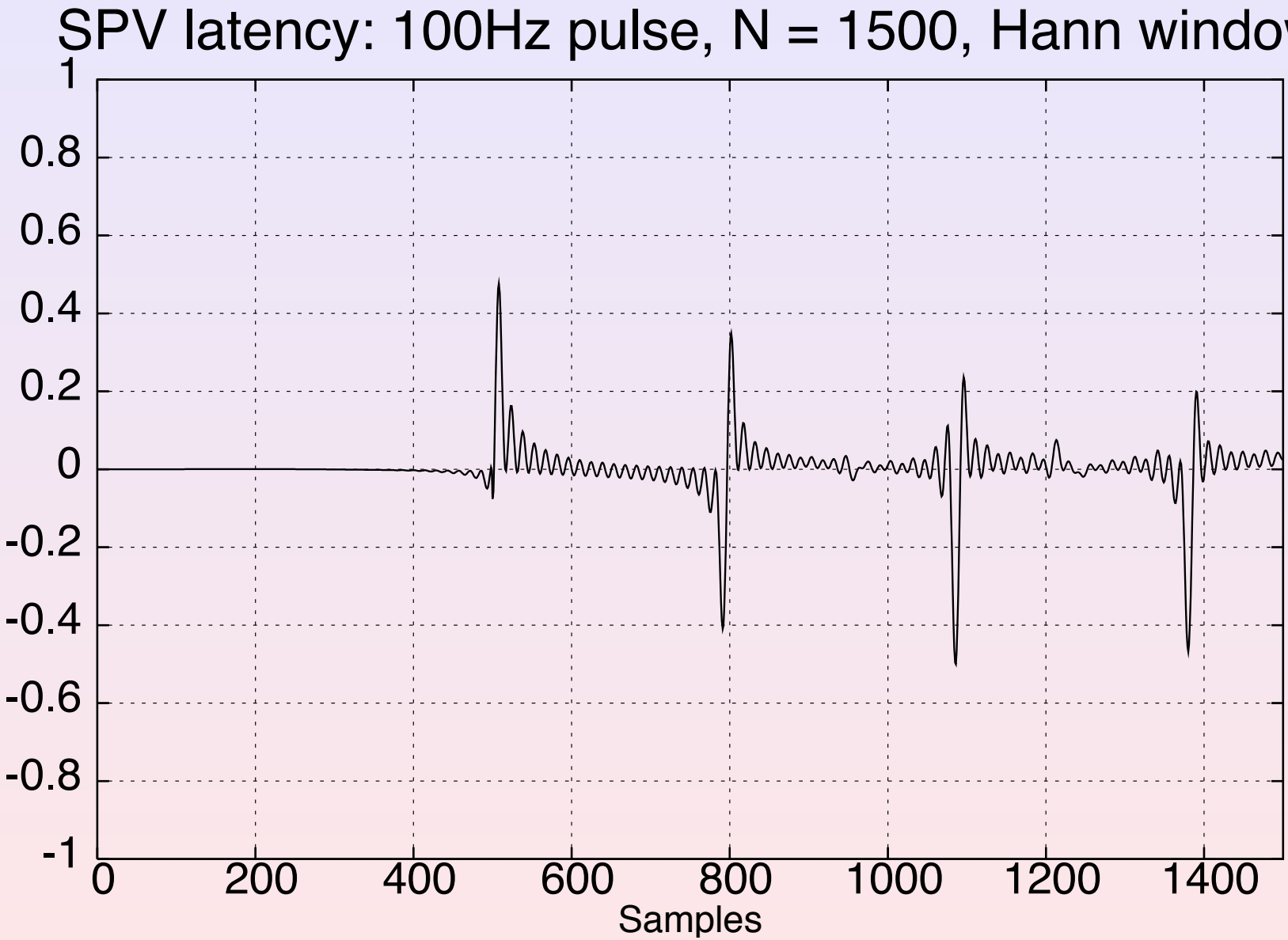
- Basic property: frequency bunching at a peak
- pvoc — constrained by narrow bin bandwidth
- SPV: bins free to converge on peak
- extraordinary bin convergence with single input sinusoid
- bin frequency map gives new view of windowing

UNIVERSITY OF BATH

Advantage — what advantage?

- reduced latency - as much as $2/3^{\text{rds}}$ reduction compared to PVOC
- predictable - avoid artifacts from changing frame overlap
- frequency modifications easy, no bin-remapping required
- drums and transients - subtle differences, no free lunch
- but more sophisticated phase-locking techniques may help SPV to shine
- two ways to pitch shift:
  - simple multiplication
  - peak tracking with additive shifts

Latency


SPV latency: 100Hz pulse, N = 1500, Hann window
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
200
400
600
800
1000
1200
1400
Samples

Drum Attacks

$SR = 44100$, $N = 1024$, von Hann window

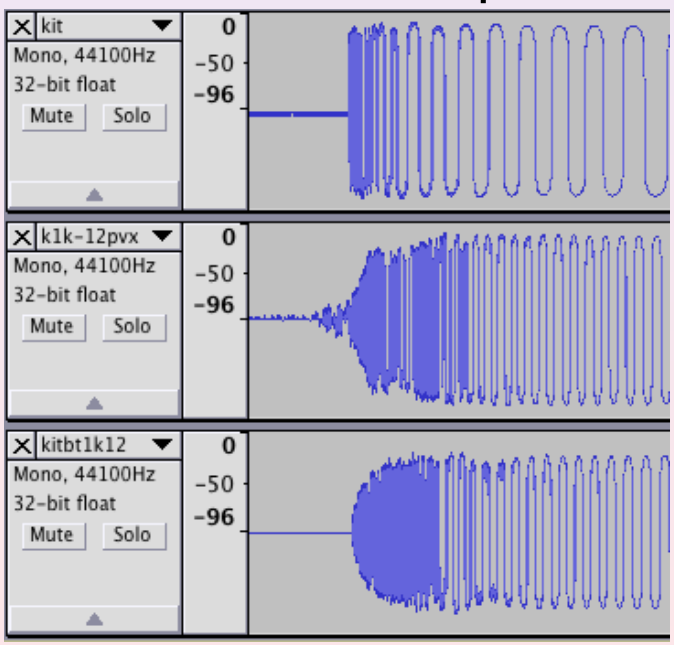
kit = source

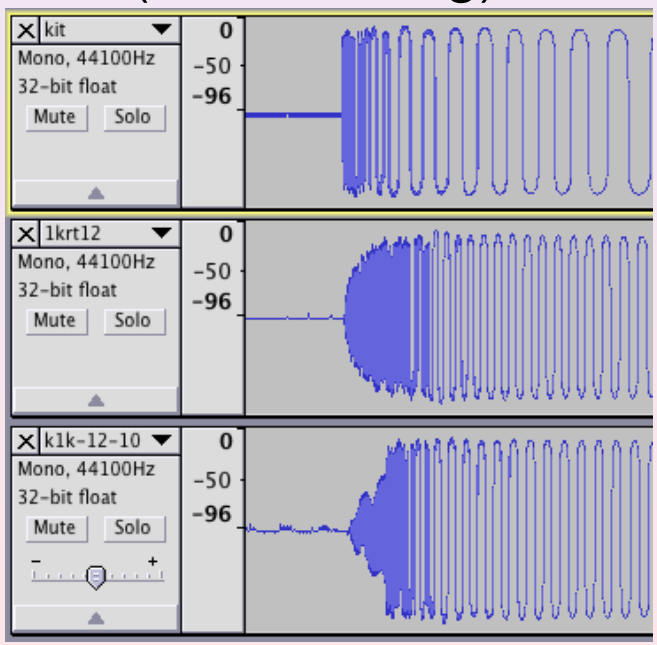
k1k-12pvx = multiplicative (1 octave up) using standard pvoc

kitbt1k12 = multiplicative (1 octave up) using SPV

1krt12 = SPV tracking shift one octave

k1k-12-10 = SPV plain additive (non-tracking) shift by 100 Hz

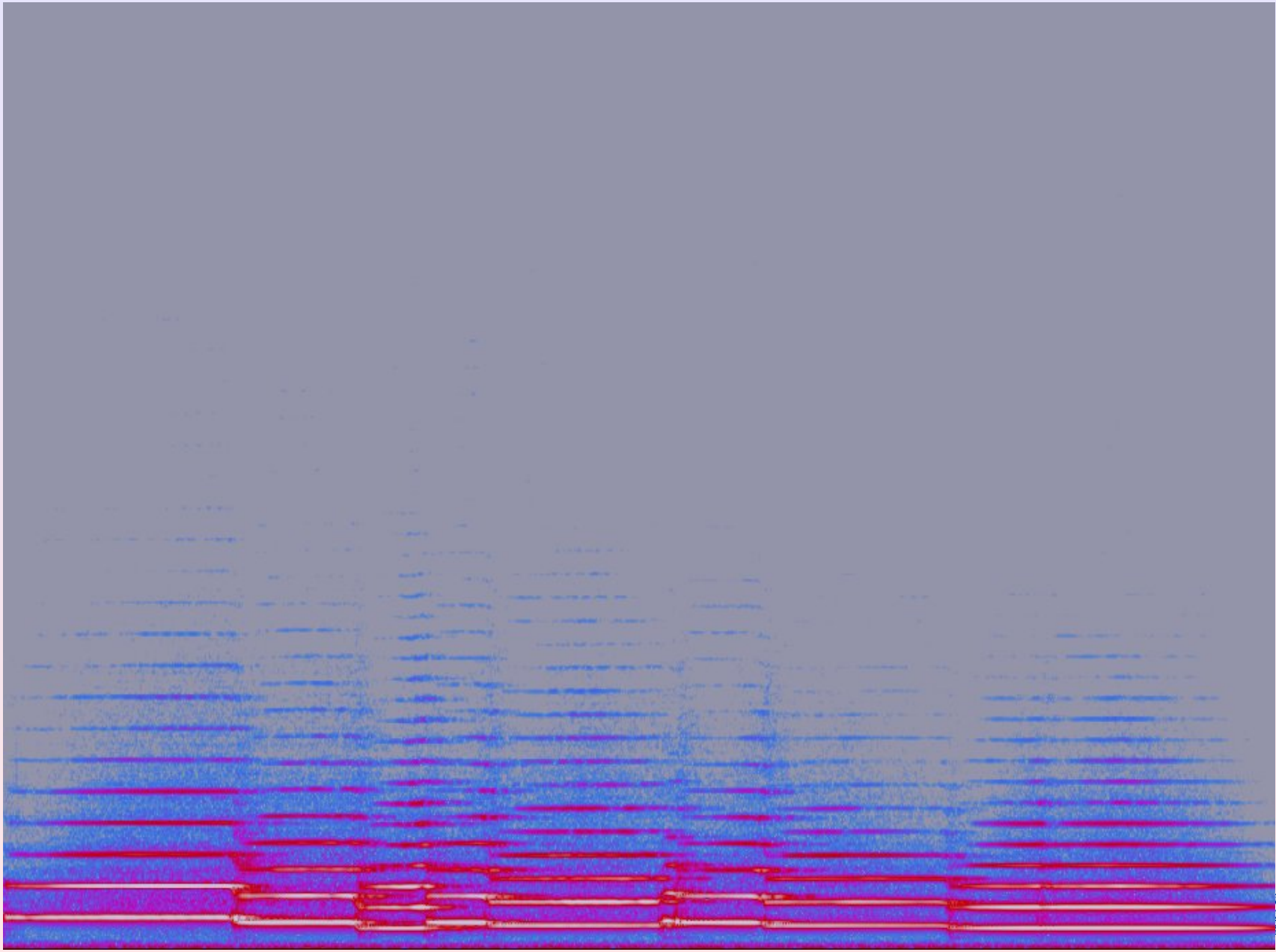




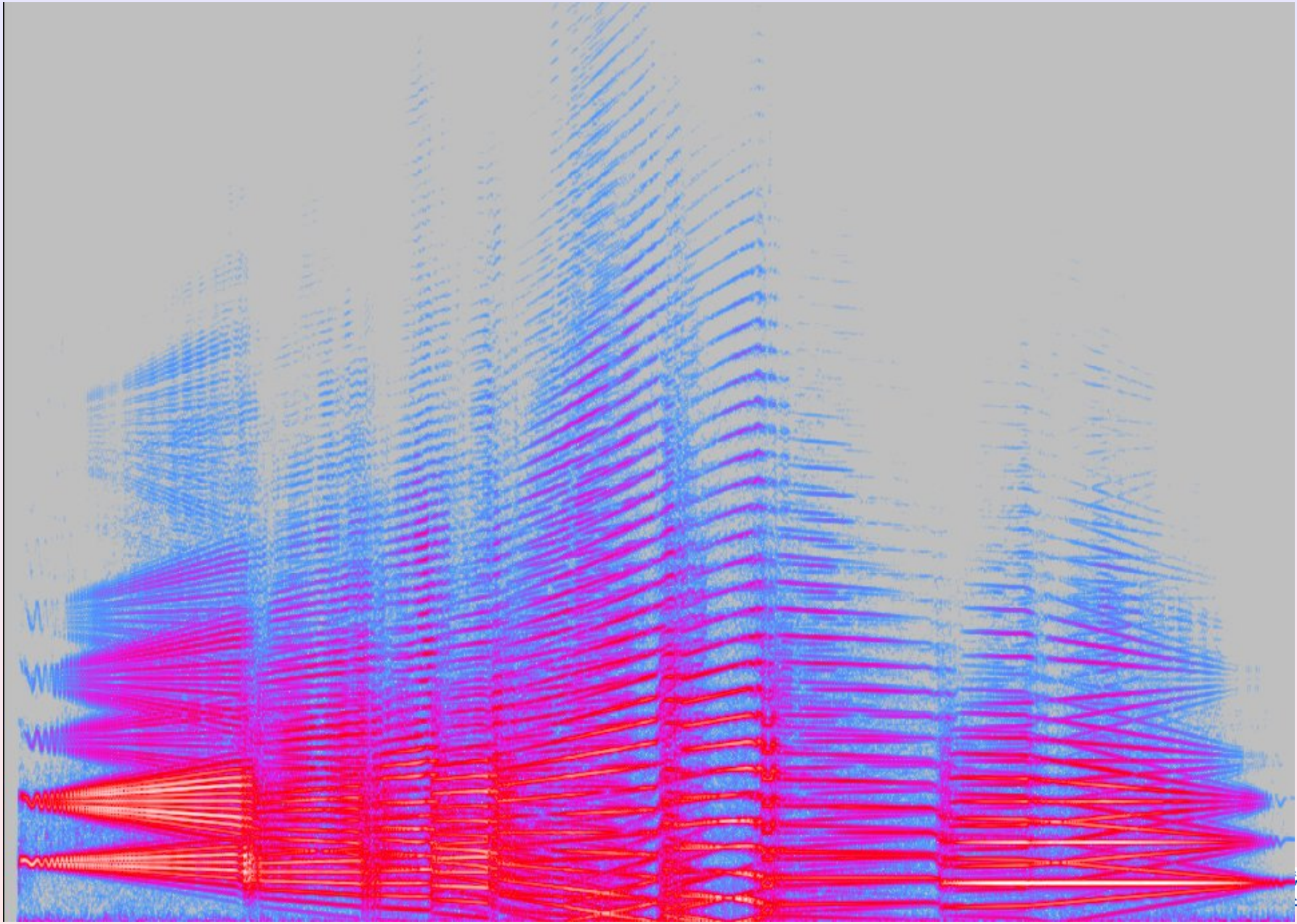
Advantage — what advantage? (cont)

- two ways to pitch shift:
  - simple multiplication
  - peak tracking with additive shifts
- simple additive shift very effective with drums *(sound example)*
- anything you can do with an oscillator bank, you can do with the SPV
- single-sample update enables transformational FM *(sound example)*

Horn Spectrum



Horn Spectrum with FM



Implementation

Our current implementation is on stock Linux and Macintosh computers.

- examples on standard hardware slow!
- expect Advance cards to enable real-time use
- will place code on the net

Looking ahead to High Performance Audio Computing

We are advocating the study of High Performance Audio Computing (HiPAC)

- focus on algorithms that are highly parallelizable (SIMD model)
- SPV is example of “no-compromise” process
- still life in PVOC - multiple channels?
- SPV special case of additive synthesis
- e.g. direct computation of b/l pulse, square, triangle waves
- Holy Grail = constant-Q (with inverse)
- other processes:
  - physical models based on meshes, finite differences
  - instruments, spaces
  - chance to explore no-compromise “ideal” algorithms

UNIVERSITY OF BATH

Looking ahead to High Performance Audio Computing

We finish with a question:

What would you do with 1TFlops?

Acknowlegment

This project has been supported by grant funding from the Arts and Humanities Research Council under their "Speculative Research" theme.