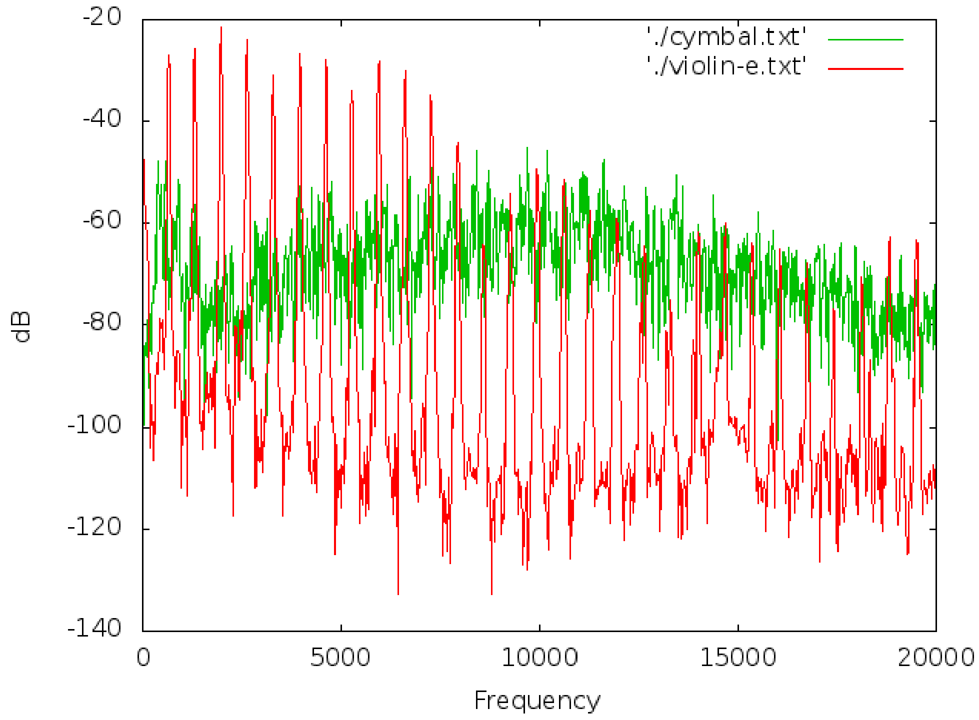
SYNTHESIS BEYOND TWO SINE WAVES

FANG Jiakun
A0123777@u.nus.edu

HARMONIC AND PARTIALS

For harmonic sounds, the first peak is the “fundamental” and the next is the first “harmonic”. Alternately, the peaks are partial 1, partial 2, etc.

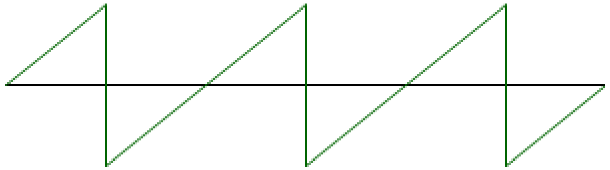


SAWTOOTH WAVE

Energy at all partials, 1/k

Given: amplitude A, frequency f , sampling rate Fs , maximum partial M.

$$x_{\text{sawtooth}}(t) = -\frac{2A}{\pi}\sum_{k=1}^{M}\frac{1}{k}\sin\left(k2\pi\frac{f}{F_s}t\right)$$



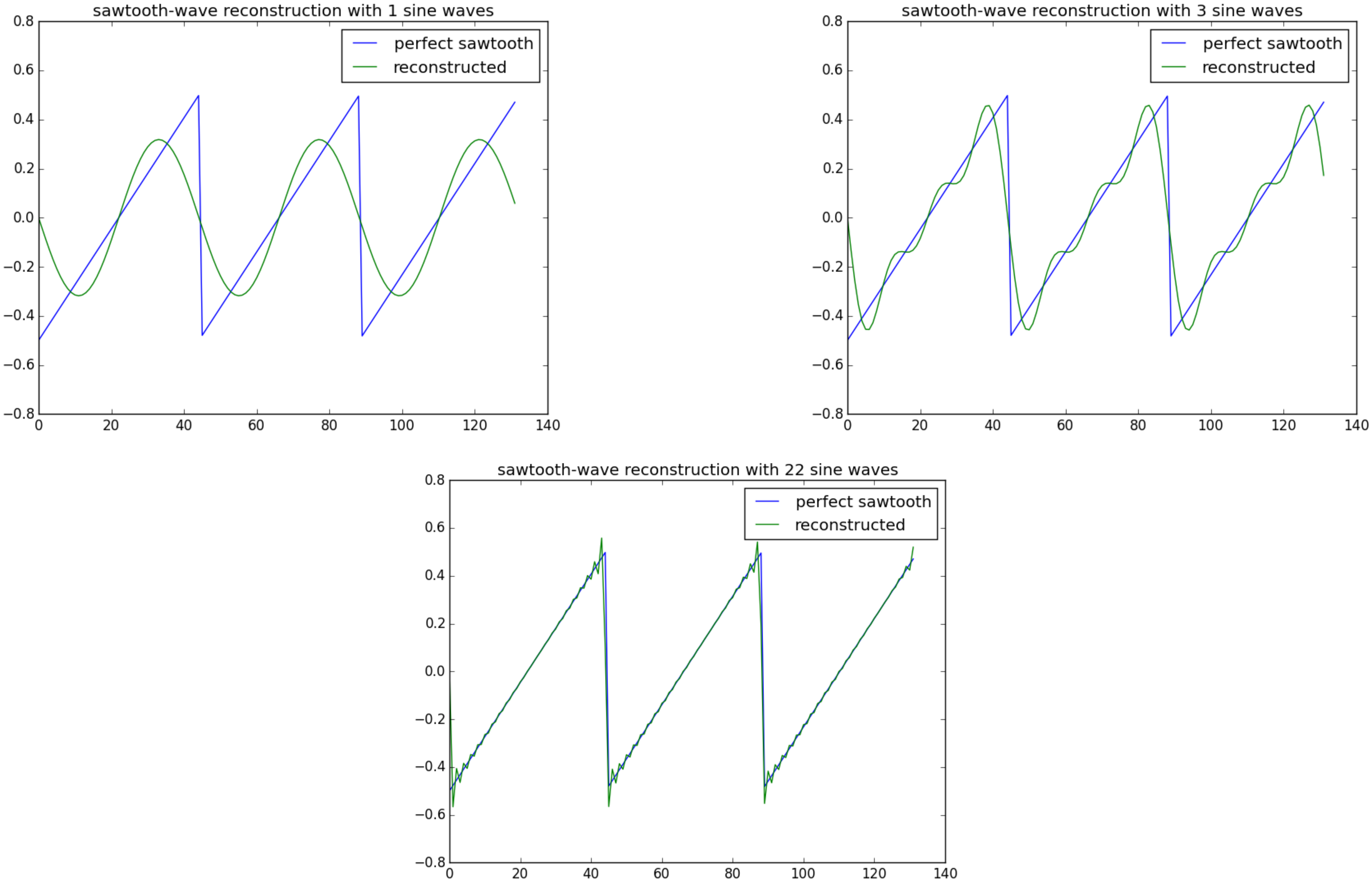
sawtooth = A*scipy.signal.sawtooth(f/FS*2*numpy.pi*t)

BAND-LIMITED

A band-limited signal can be fully reconstructed from its samples, provided that the sampling rate exceeds twice the maximum frequency in the band-limited signal.

$$x_{\text{sawtooth}}(t) = -\frac{2A}{\pi}\sum_{k=1}^{M}\frac{1}{k}\sin\left(k2\pi\frac{f}{F_s}t\right)$$

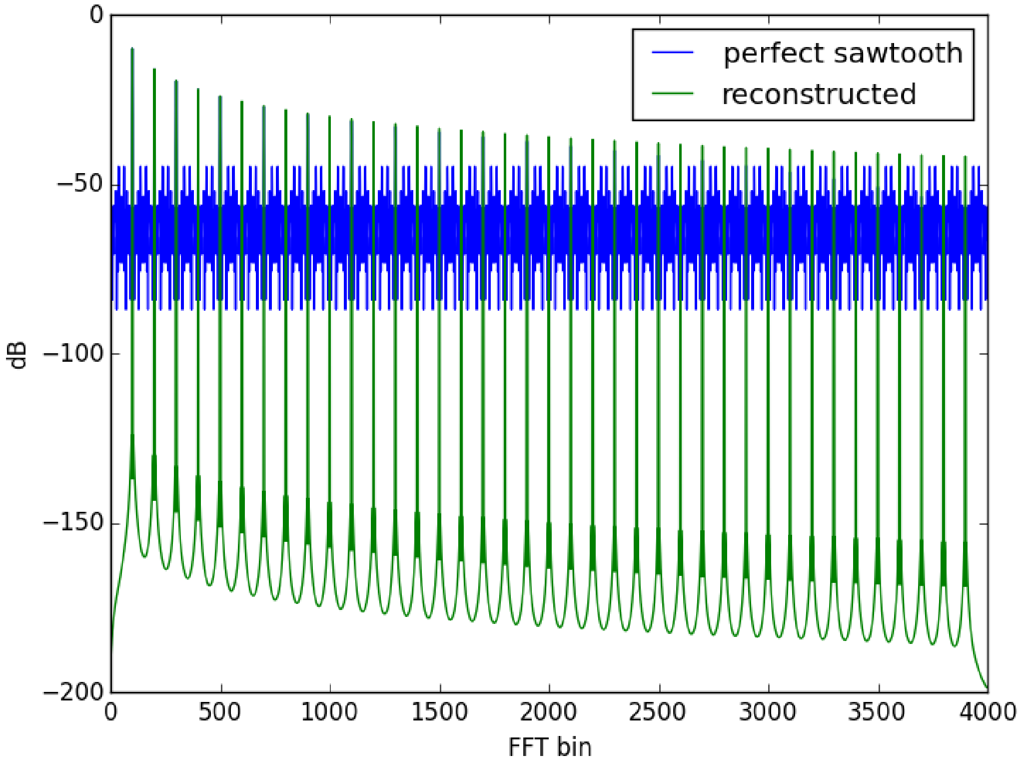
Perfect sawtooth wave is not a band-limited signal.



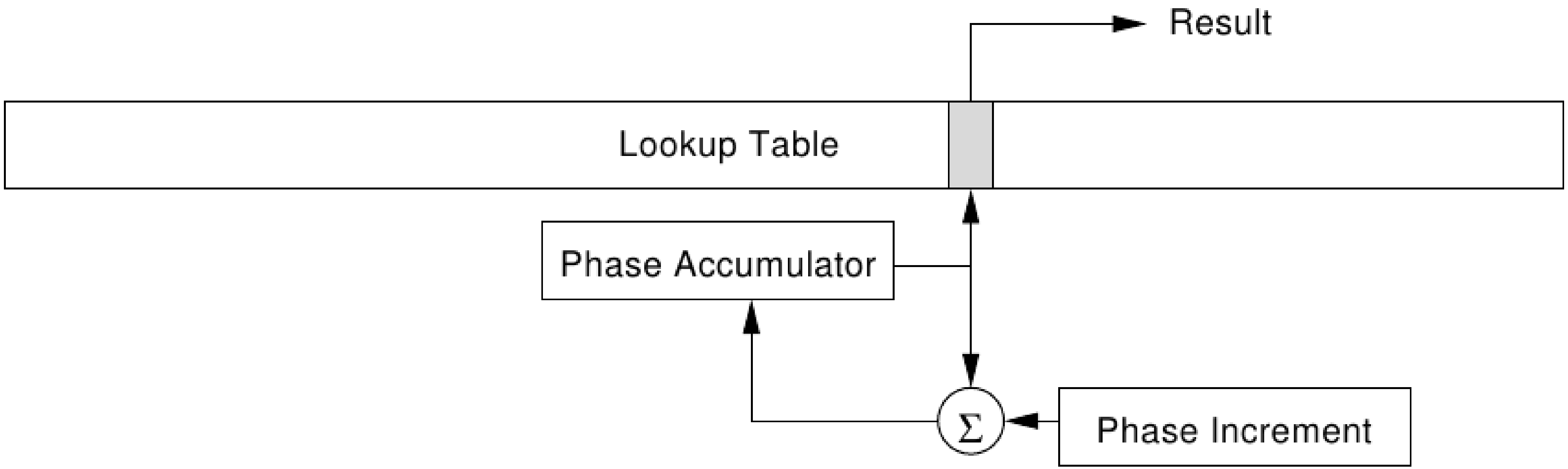
The additive synthesis of a sawtooth wave with an increasing number of harmonics

DB-MAGNITUDE FFT

```
def power_spectrum(x):
    window = numpy.blackman(len(x))
    fft = numpy.fft.fft(x*window)
    # only keep the positive frequencies
    fft = fft[:len(fft)/2+1]
    # magnitude sprectrum, normalize
    magfft = abs(fft) / (numpy.sum(window)/2.0)
    # log-spectrum
    epsilon = 1e-10
    db = 20*numpy.log10(magfft + epsilon)
    return db
```



LOOK-UP TABLE



LOOK-UP TABLE

Synthesize 1 cycle of a sine wave, then alter the way you read it to get any frequency.

- Suppose our table contains NL = 44100 samples
- If the phase increment is 1, we get a sine wave at 1 Hz
- If the phase increment is 100, we get a sine wave at 100 Hz
- When the phase accumulator is larger than NL, MOD NL

To get frequency f , advance by ϕinc samples (can be a float!)

$$\phi_{\text{inc}} = \frac{f}{FS} N_{\text{L}}$$

LINEAR INTERPOLATION

Given (x0, y0) and (x1, y1), find the value at x given x0 ≤ x ≤ x1 and x0 ≠ x1.

$$y = y_0 + (y_1 - y_0)\frac{x - x_0}{x_1 - x_0}$$

y0 = lookup_table[floor(x)]

y1 = lookup_table[floor(x) + 1]

LINEAR INTERPOLATION