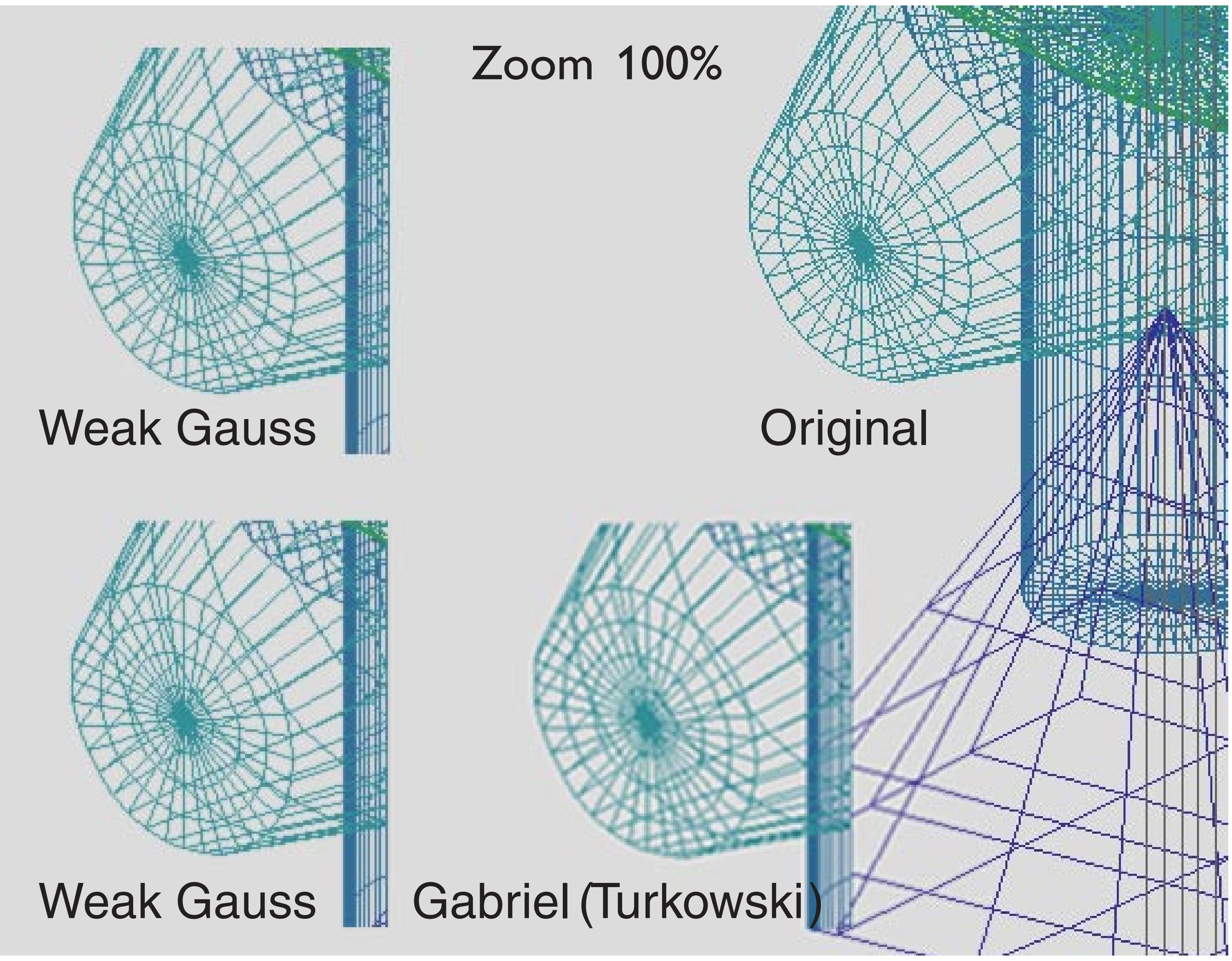
Gernot Hoffmann

Windowed Sinc Interpolation



Contents

Settings for Acrobat

Edit / Preferences / General / Page Display (since version 6)

Custom Resolution 72 dpi and zoom 200%

Edit / Preferences / General / Color Management (full version only)

sRGB
EuroscaleCoated or ISOCoated or SWOP
GrayGamma 2.2

Windowed Sinc Interpolation

1.1 Introduction

Interpolation in images can be done by many methods. Excellent references are Thévenaz et.al. [1] and Meijering [2] .
In actual discussions the Lanczos interpolation is often favoured. Lanczos belongs to the so-called 'Windowed Sinc' methods.
The Sinc function is the ideal reconstruction filter (Whittaker cardinal function) for band-limited data streams of infinite length [3], [4] .
Images are generally not band-limited, because single pixel lines, steps, hairs and grass generate a considerable amount of Nyquist frequencies, which cannot be handled correctly by any digital filter.
Let us assume, an image with alternating vertical black and white lines is scanned in horizontal direction. We get a DC value, and in the AC part the lowest frequency is the Nyquist frequency, which is half the pixel clock frequency.

Turkowsky [5] shows only Bode plots, gain versus frequency. Without step responses, the frequency domain information is often misleading.
The frequency axis notation is not understandable. These curves are symmetrical with respect to the Nyquist frequency.
Therefore the end of the frequency axis should be either the Nyquist frequency at 0.5 or the sampling frequency at 1.0.

In this document we show interpolated step samples and interpolated oscillation samples. The oscillations are band-limited.
Descriptions are in terms of signal processing.

Sample period: T
Effective length of the truncated sinc function: -nT ...nT

Rectangle Window	n = 5	Overshoot, ringing, DC error for steps
Lanczos Window	n=2, n=3	Overshoot, DC error for steps
Von Hann Window	n=2, n=3	Overshoot, DC error for steps

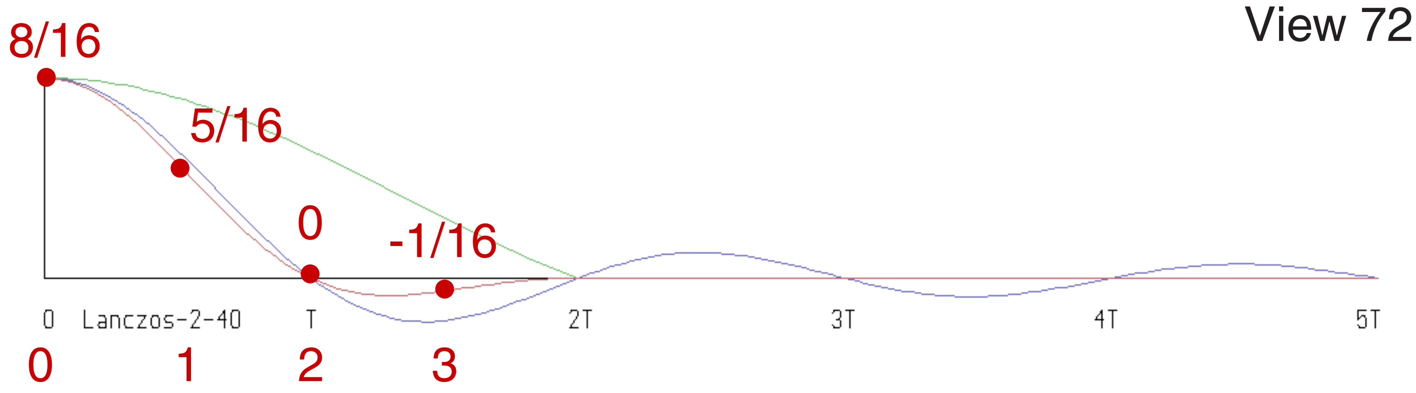
Lanczos and von Hann are very similar. Overshoots will cause halos at sharp edges. They require also clipping.

Windowed Sinc Interpolation

1.2 Introduction

Actually, this paper describes reconstruction. This is not exactly the same as filtering.

View 72 dpi / zoom 200%



The curves are symmetrical. The right half is shown, the left side has to be mirrored.
Reconstruction uses the whole infinite or truncated sinc function (blue) or the finite windowed sinc function (red).

Filtering uses a finite number of weight factors from the windowed curve (red dots) in a Finite Impulse Response filter (FIR).
Often we find some rounding to rational integer values. Here as an example (above) the Gabriel filter [5], which uses a 7x7 kernel with many zeros.

A correctly designed FIR filter does not generate DC errors, opposed to this reconstruction.

Once the sinc function is windowed, then we have exactly one correct reconstruction algorithm.
On the other hand we can build an arbitrary number of FIR filters of different orders, taking more or less values from the windowed sinc function. This is similar to taking values from a Gaussian bell.

These considerations lead to the question, whether an FIR filter, based on windowed sinc, has any close relation to the ideal reconstruction sinc function.

Windowed Sinc Interpolation
2.1 Formulas

$$f_s(t) = \operatorname{sinc}(\pi t/T) = \frac{\sin(\pi t/T)}{\pi t/T}$$

Whittaker reconstruction, valid for $n \to \infty$

$$f_f(t) = \sum_{k=-n}^{n} f_k \cdot \frac{\sin[\pi(t-kT)/T]}{\pi(t-kT)/T}$$

$$f_f(t) = \sum_{k=-n}^{n} f_k \cdot \frac{\sin[\pi|t-kT|/T]}{\pi|t-kT|/T}$$

Rectangle window function, finite n

$$f_w(t) = \begin{cases} 1 & -nT \le t \le +nT \\ 0 & \text{Else} \end{cases}$$

$n = 5$ Rectangle 5

Lanczos window function, finite n

$$f_w(t) = \begin{cases} \sin(\pi t/nT)/(\pi t/nT) & -nT \le t \le +nT \\ 0 & \text{Else} \end{cases}$$

$n = 2$ Lanczos 2
$n = 3$ Lanczos 3

Von Hann window function, finite n

$$f_w(t) = \begin{cases} 0.5[1+\cos(\pi t/nT)] & -nT \le t \le +nT \\ 0 & \text{Else} \end{cases}$$

$n = 2$ Von Hann 2
$n = 3$ Von Hann 3

Blackman window function, finite n

$$f_w(t) = \begin{cases} 0.42 + 0.5\cos(\pi t/nT) + 0.08\cos(2\pi t/nT) & -nT \le t \le +nT \\ 0 & \text{Else} \end{cases}$$

$n = 2$ Blackman 2
$n = 3$ Blackman 3

Whittaker reconstruction, windowed sinc, finite n

$$K = (t \operatorname{div} T - n) \cdot T$$

$$f_f(t) = \sum_{k=-n}^{n} f_u(K) \cdot f_w(t-K) \cdot \frac{\sin[\pi|t-K|/T]}{\pi|t-K|/T}$$

Actual implementation in the procedure DoFilt

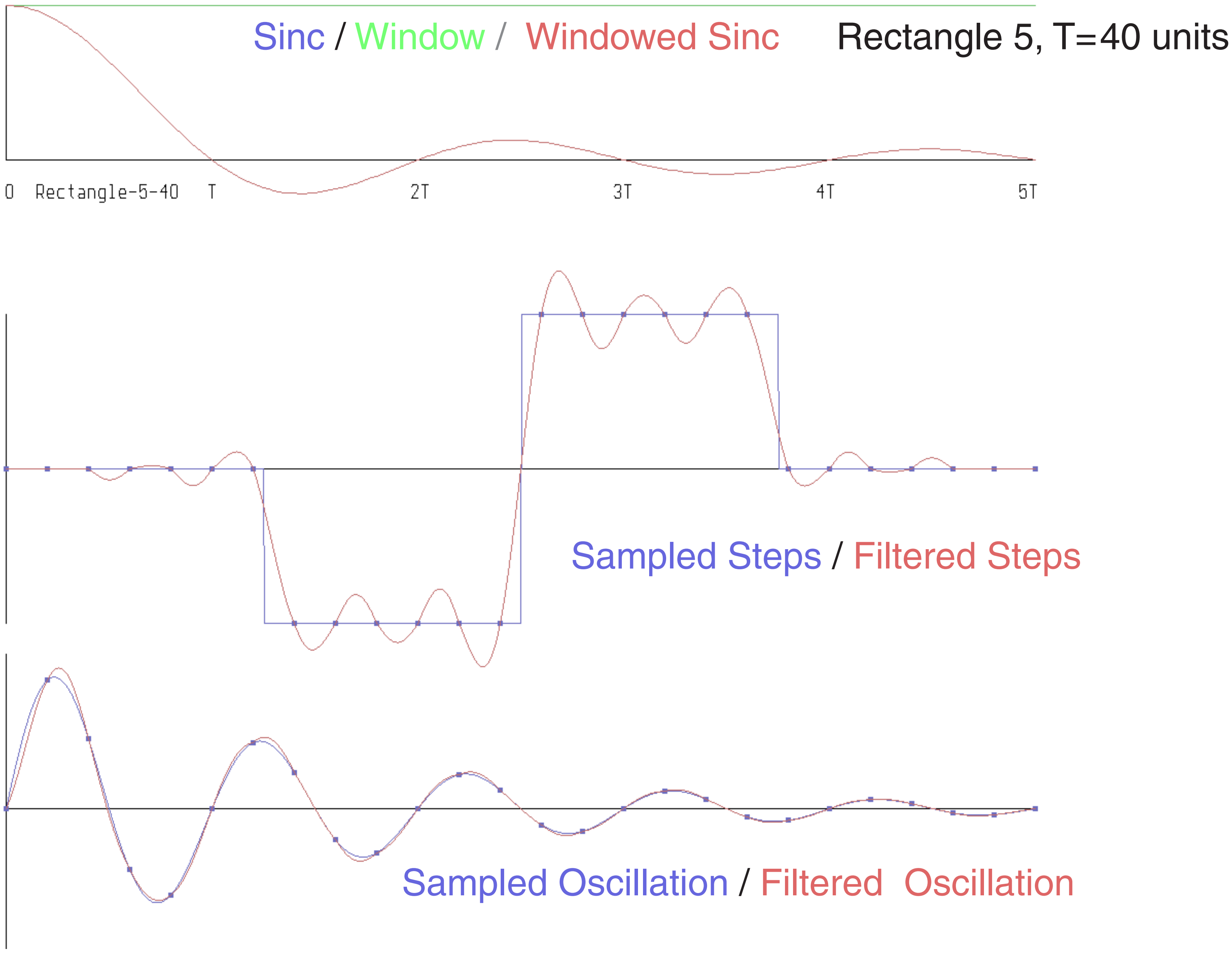
t	Independent variable, time
$f_u(t)$	Function
$f_s(t)$	Sinc function
$f_w(t)$	Window function
$f_v(t)$	Windowed sinc function
f_k	Sampled function $f_k = f_u(t_k)$
$f_f(t)$	Interpolated function
T	Sample period
n	Effective length of filter -nT...nT

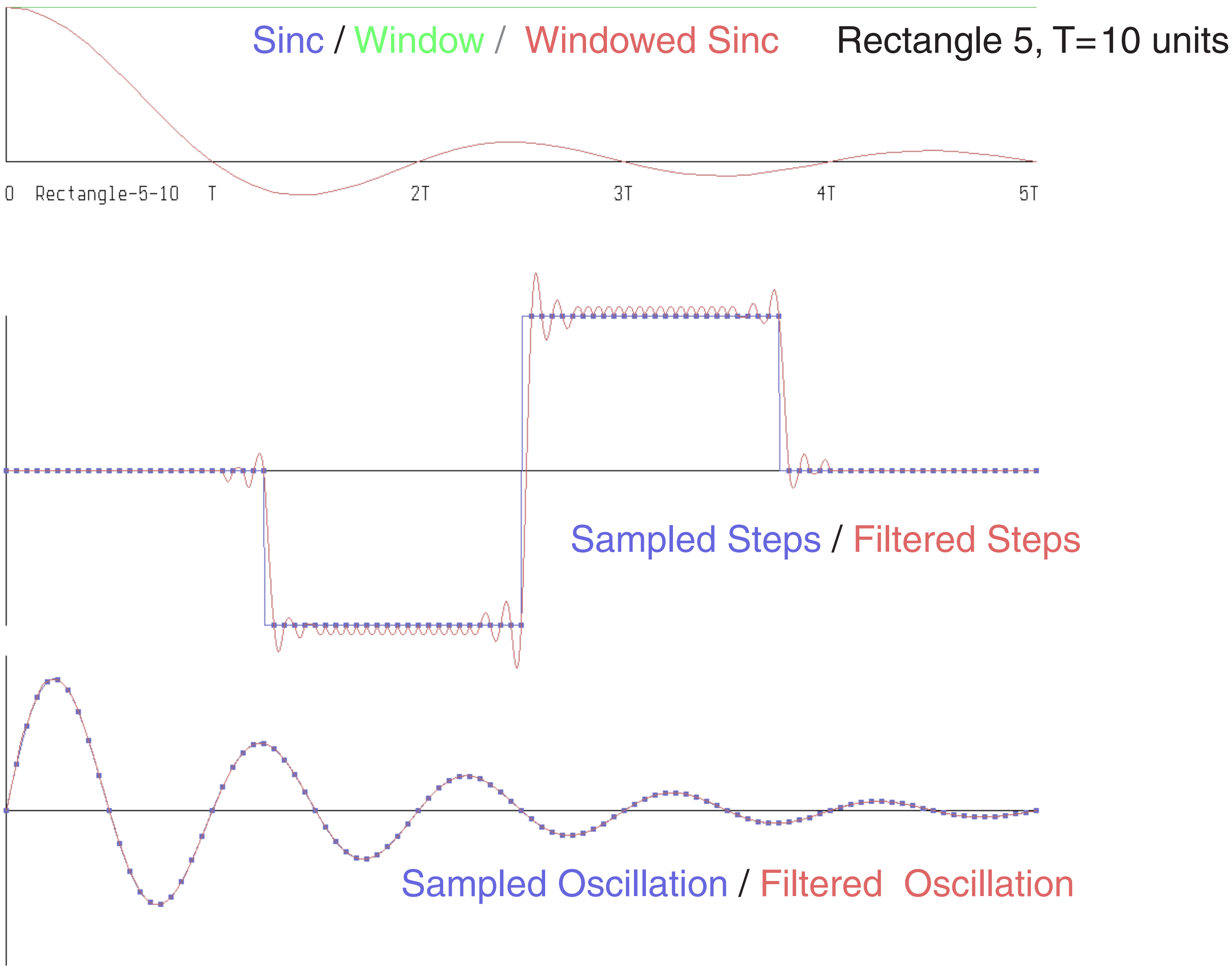
In the examples the sampling time T is given in ‘units’ (10 or 40 units). The period of the damped oscillation has the length 200 units.

The scale for t in the upper diagram (Sinc) is different to the scale in the other diagrams (Steps, Oscillation).

Windowed Sinc Interpolation
3.1 Rectangle Window

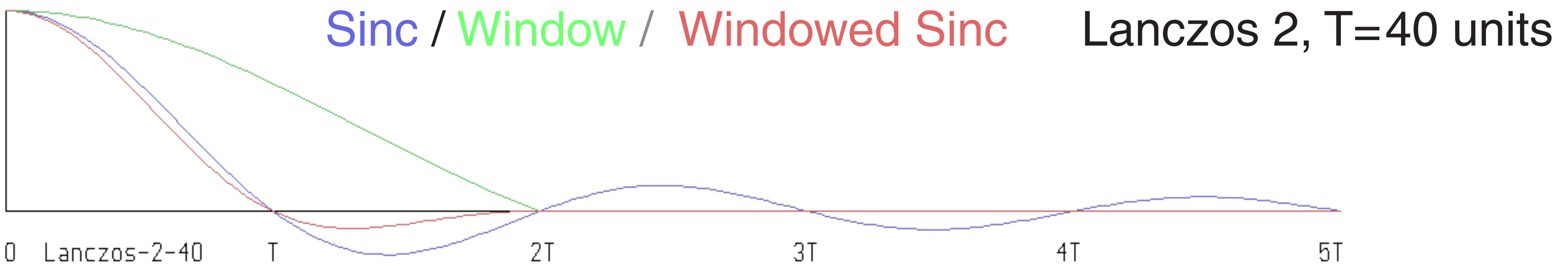
View 72 dpi / zoom 200%



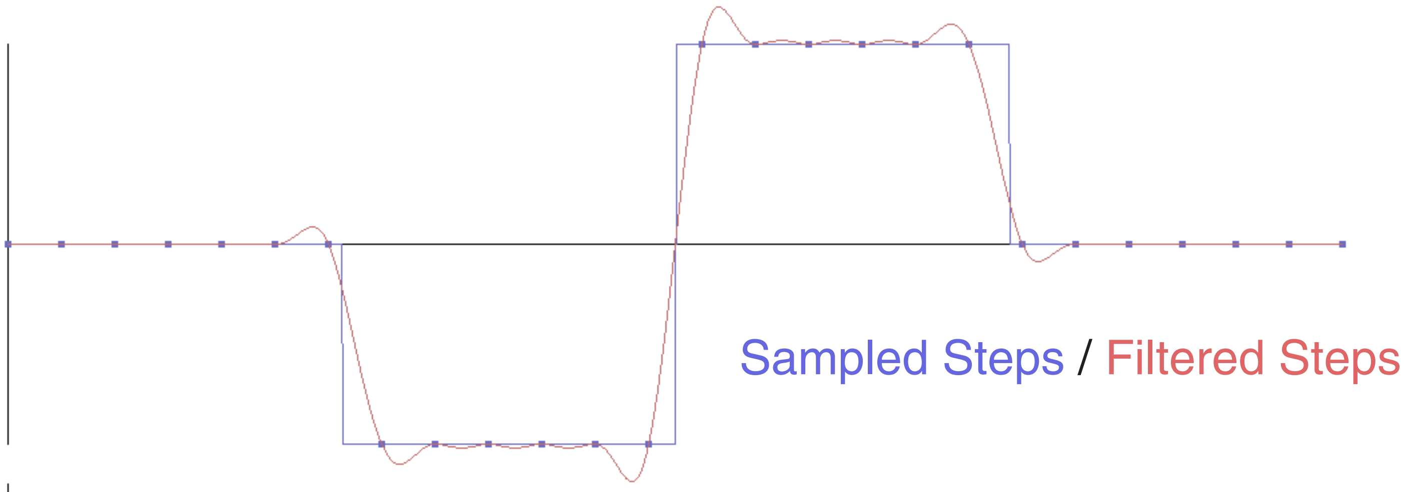


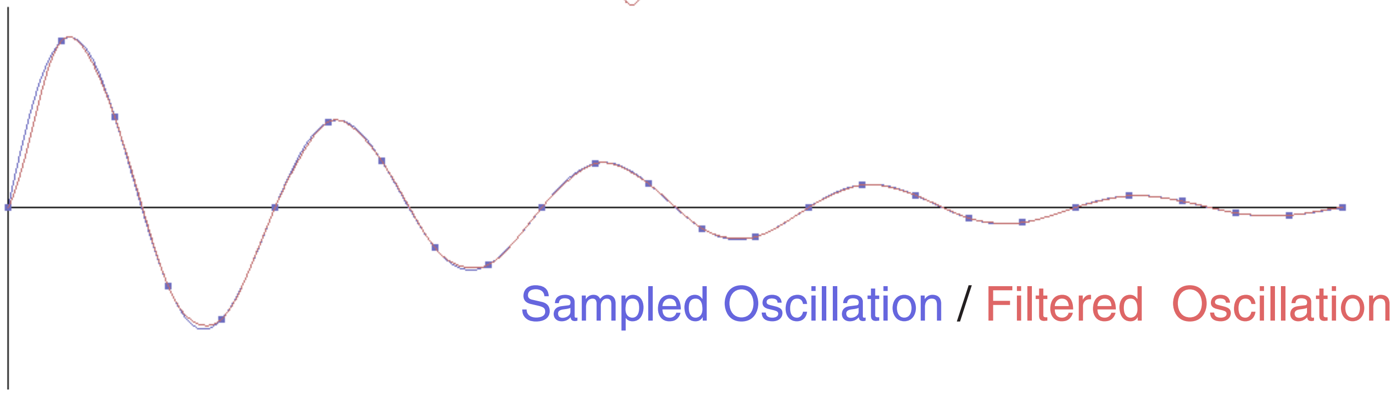
Windowed Sinc Interpolation
4.1 Lanczos Window

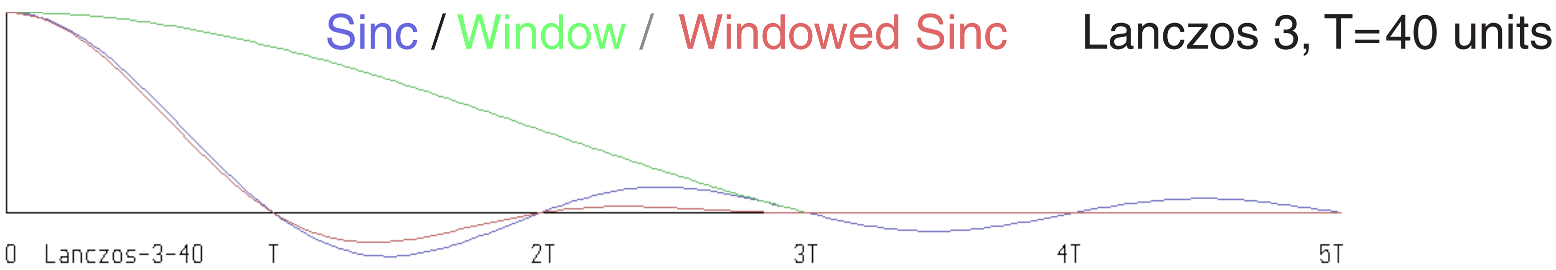
View 72 dpi / zoom 200%

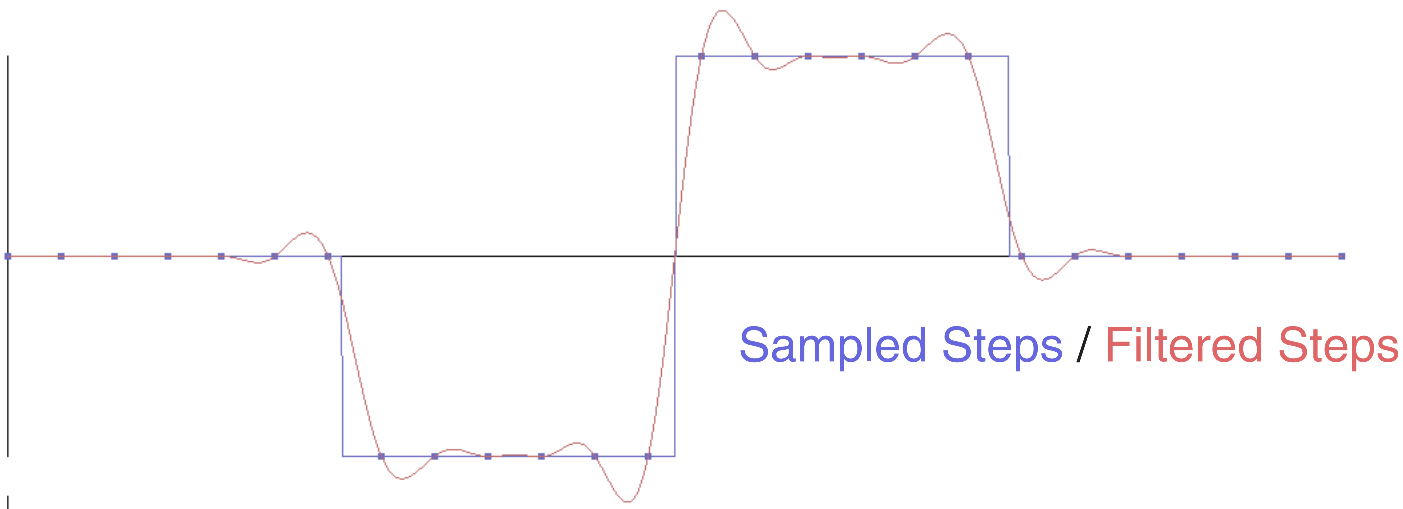


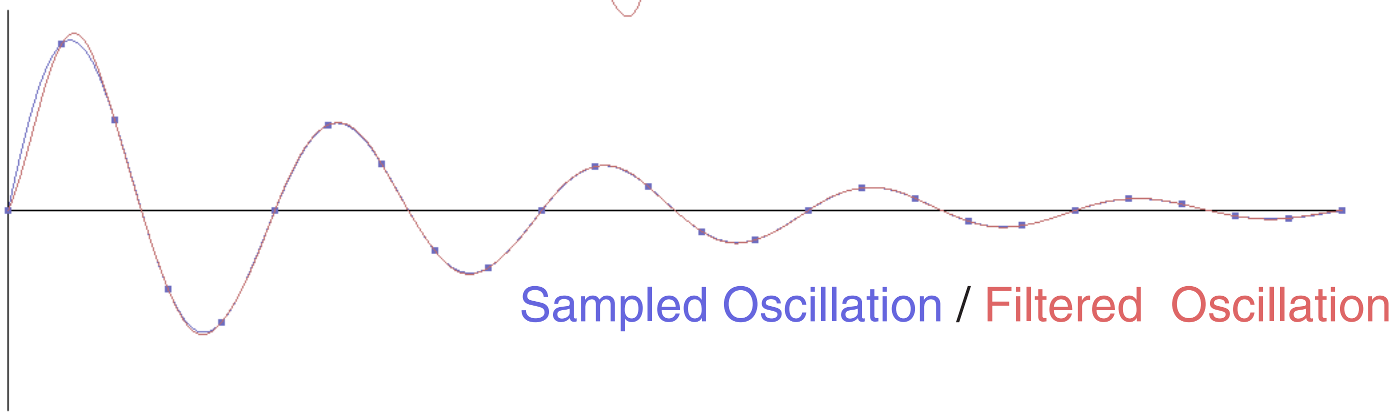
View 72 dpi / zoom 200%







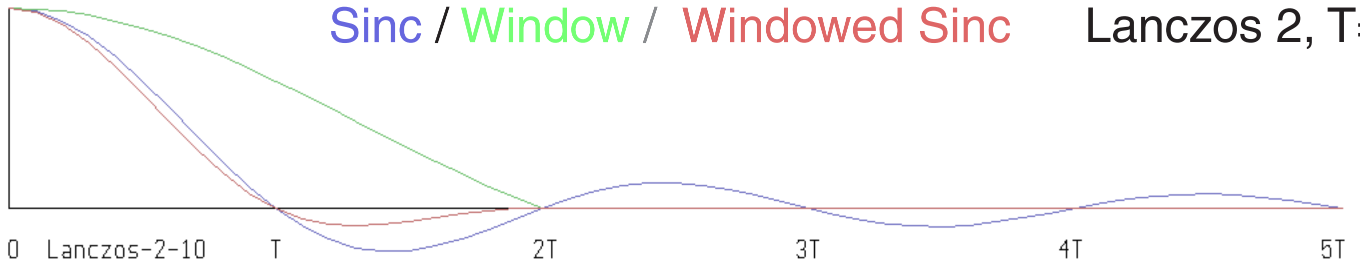


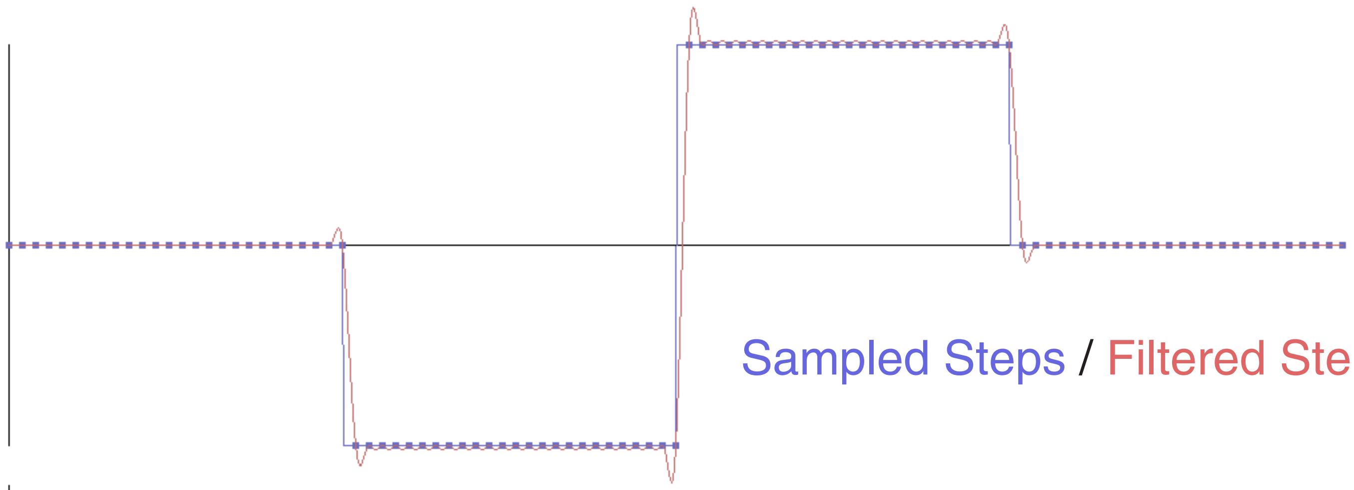


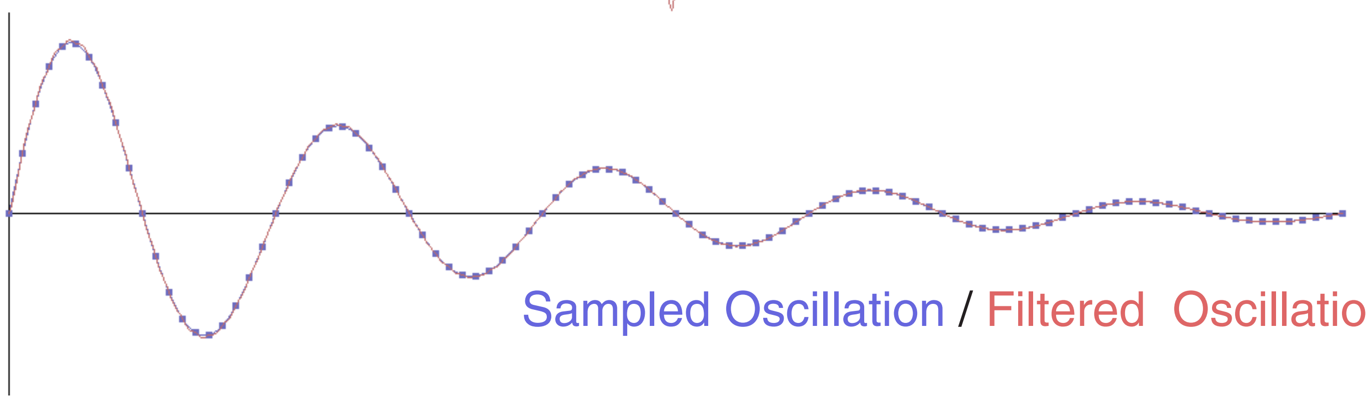
Windowed Sinc Interpolation
4.2 Lanczos Window

View 72 dpi / zoom 200%

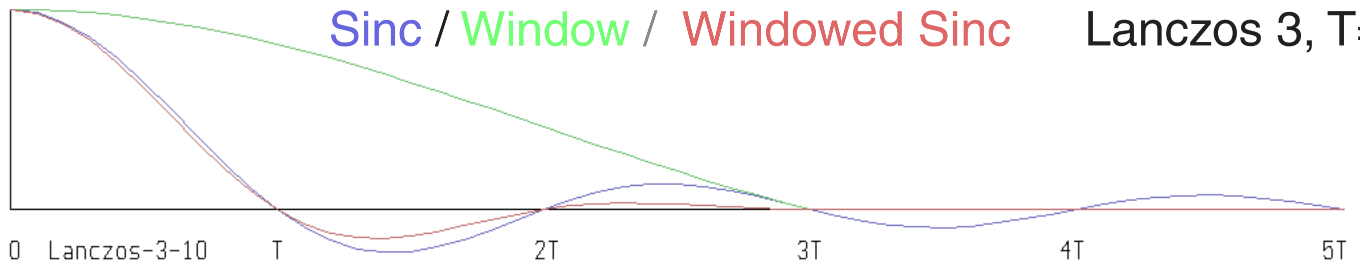
Lanczos 2, T=10 units

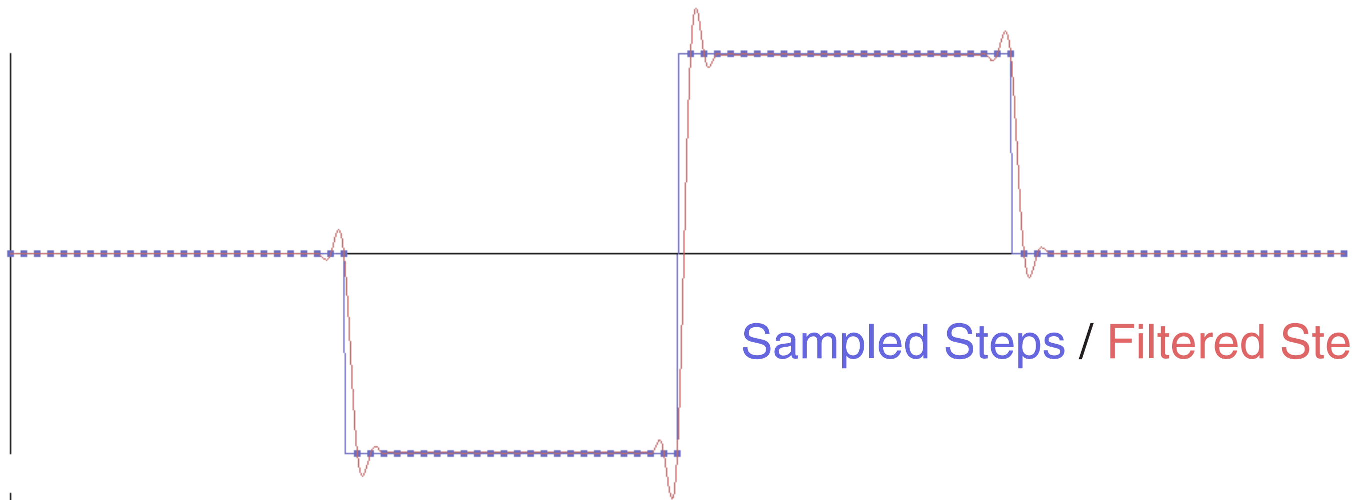


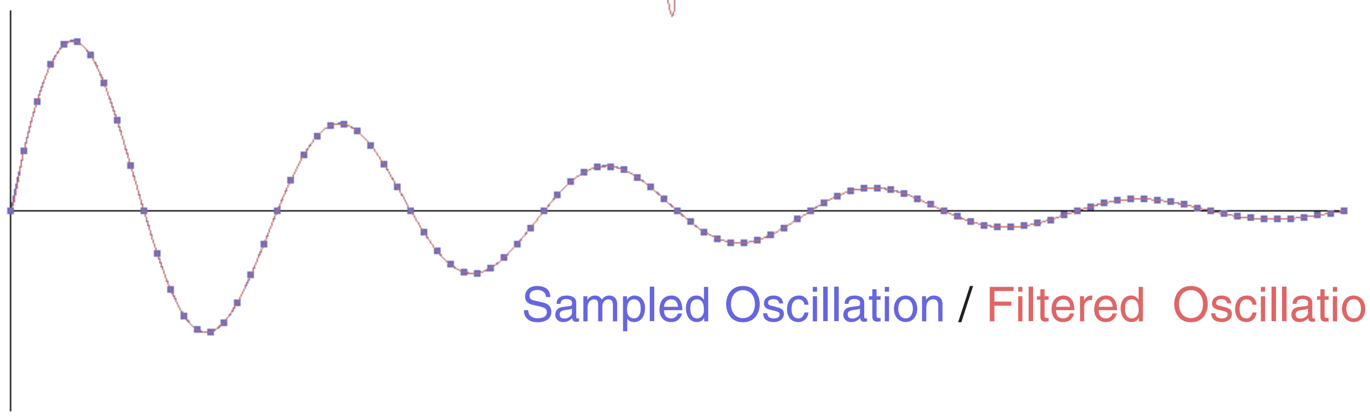




Lanczos 3, T=10 units

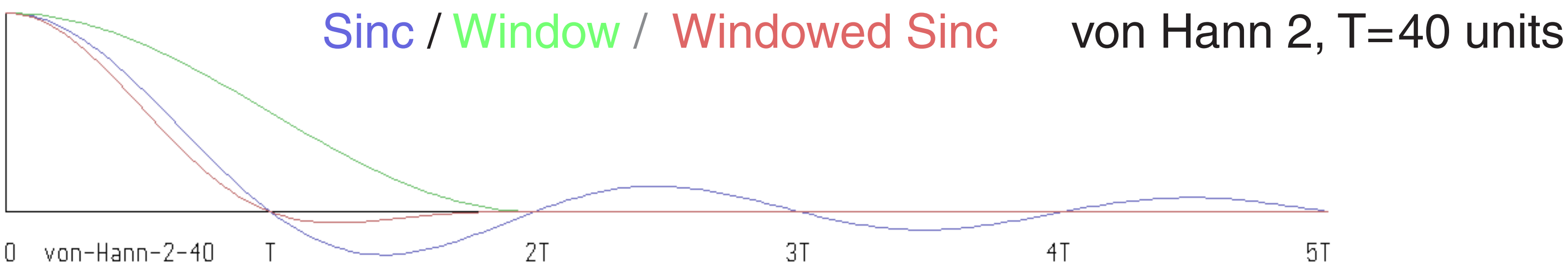


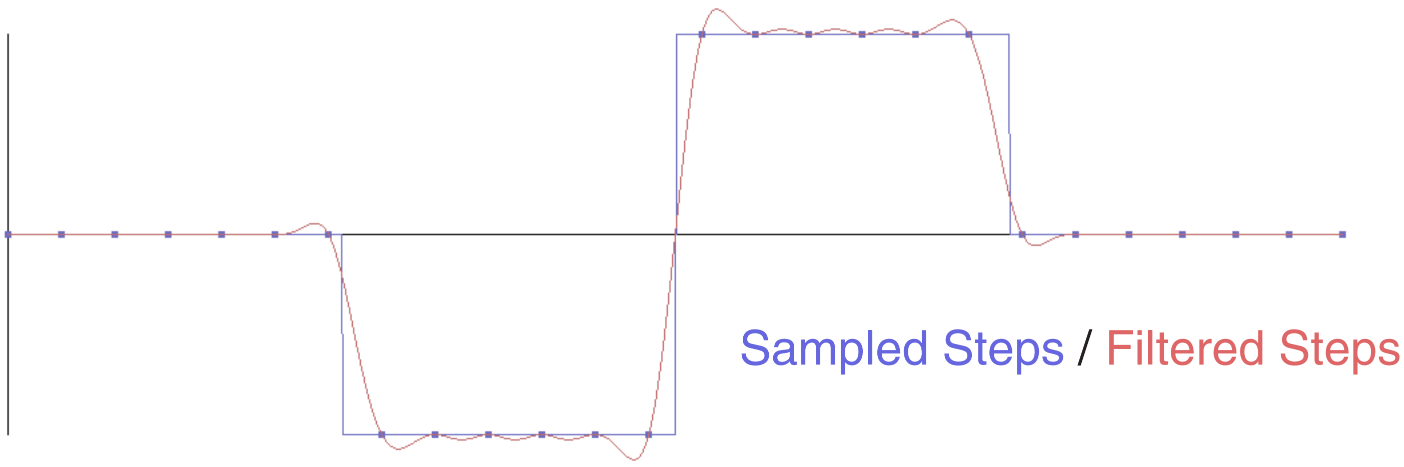


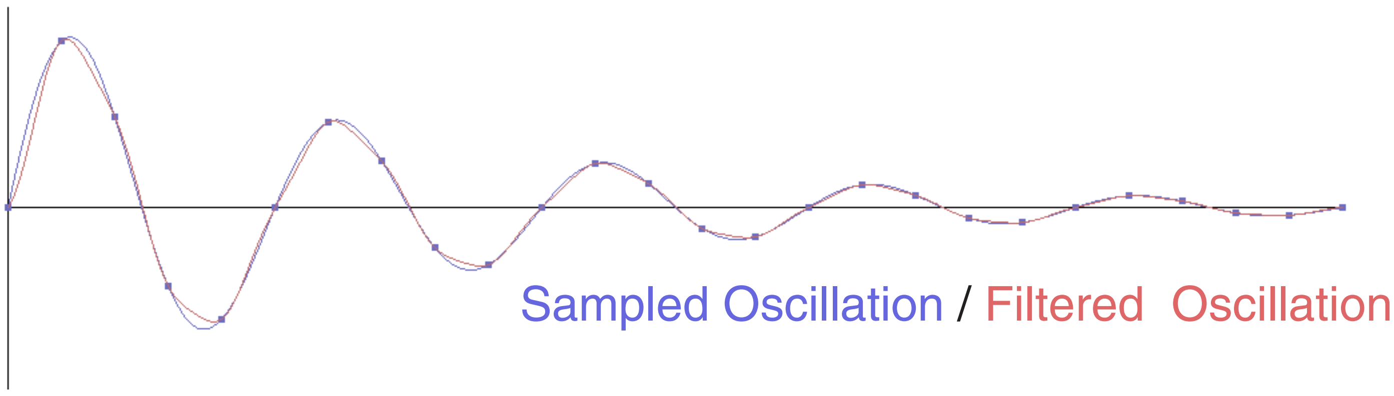


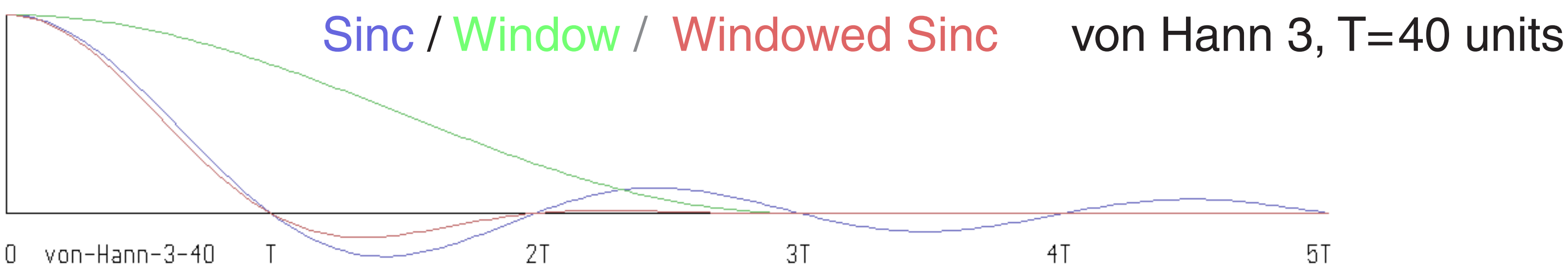
Windowed Sinc Interpolation
5.1 Von Hann Window

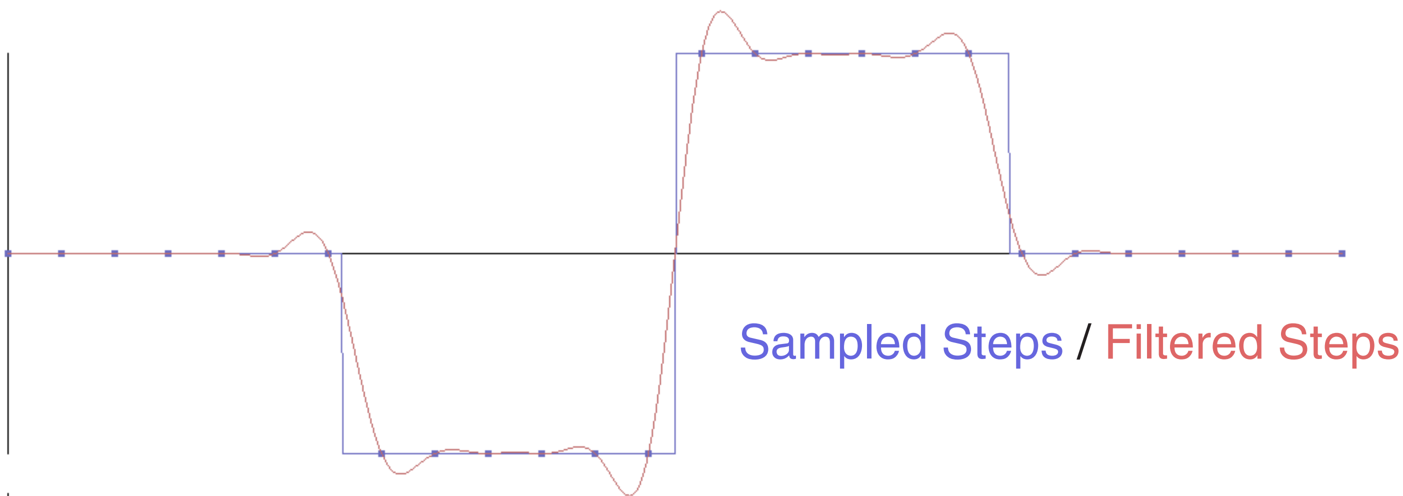
View 72 dpi / zoom 200%

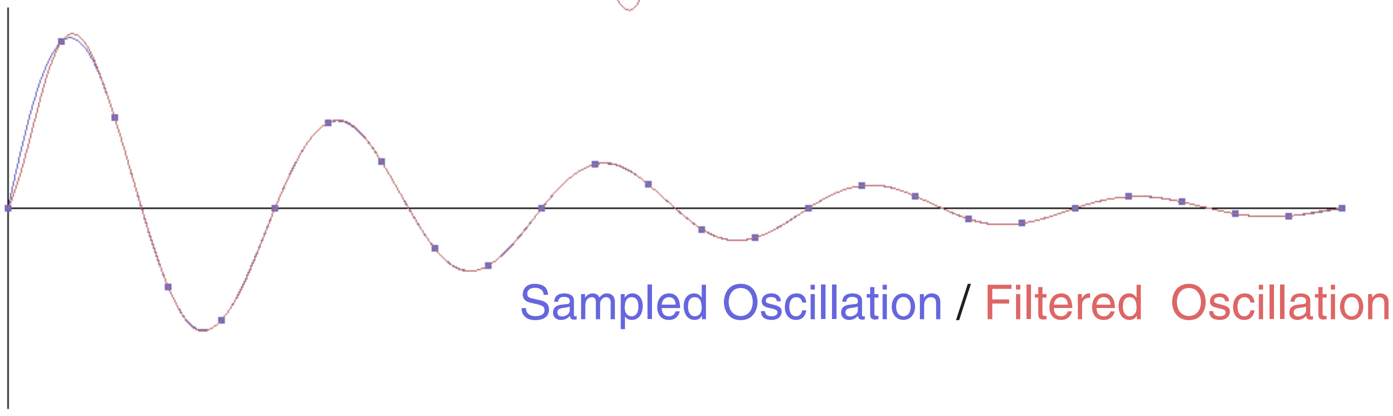






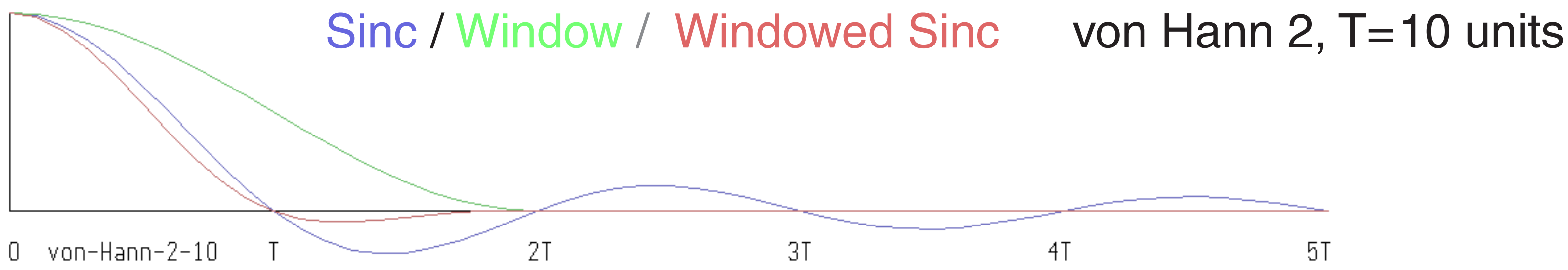


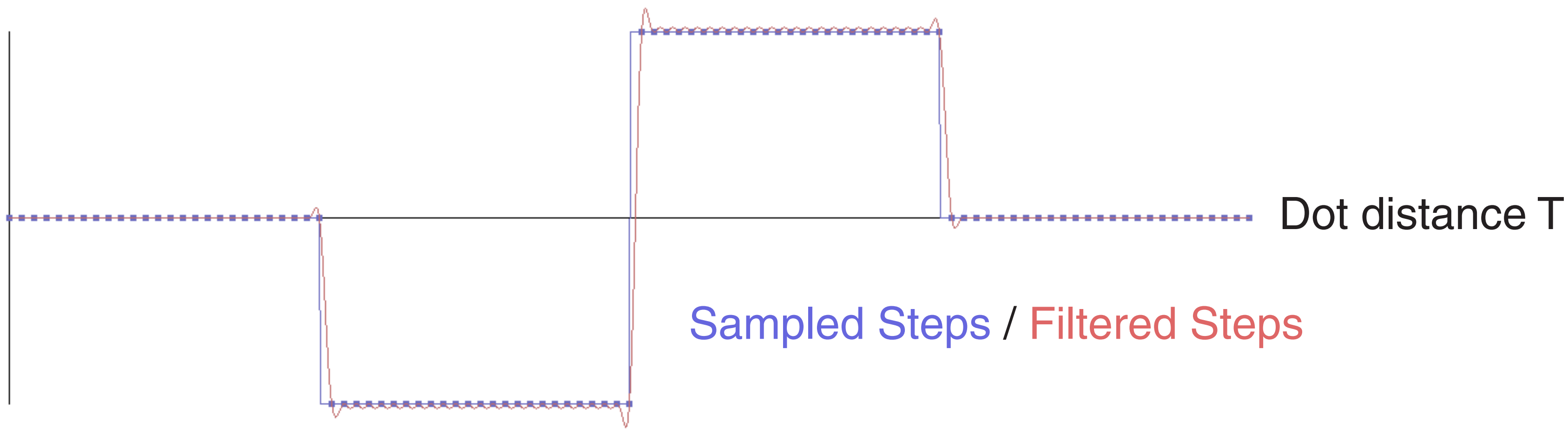


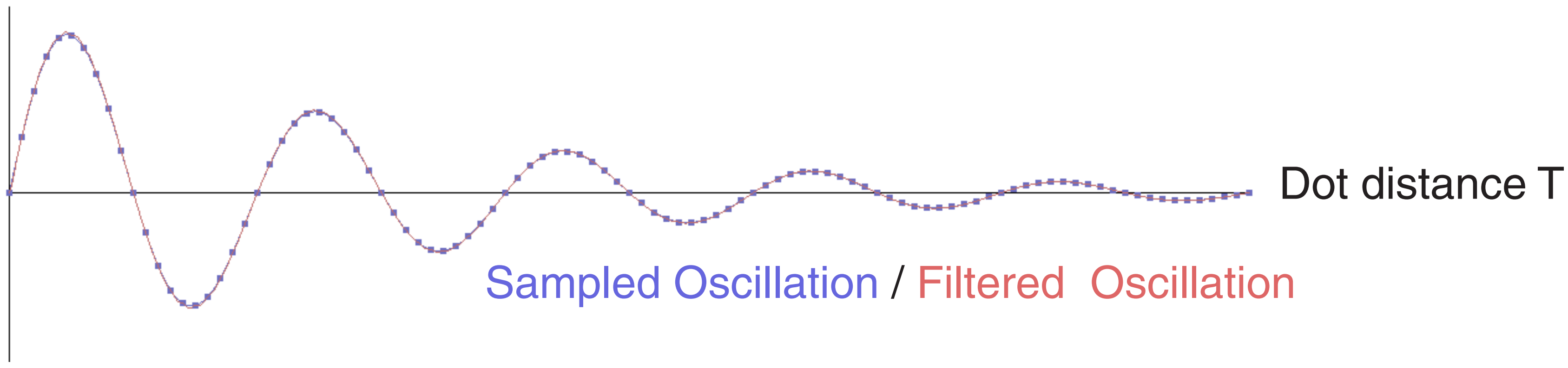


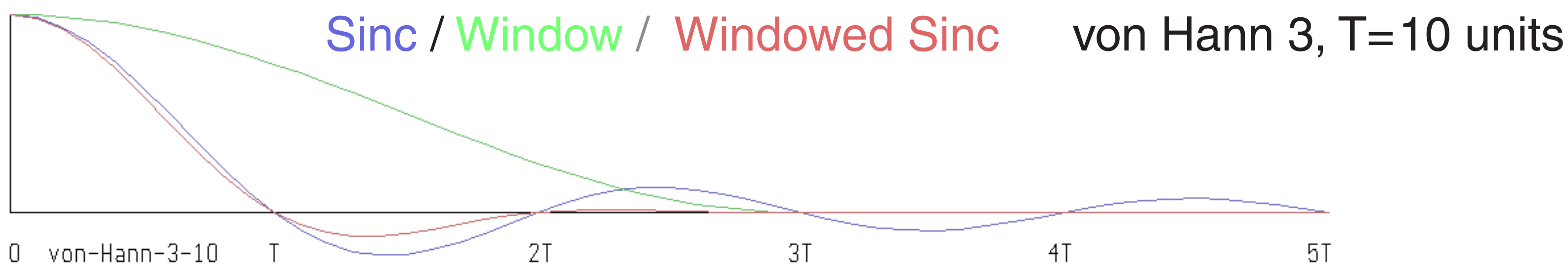
Windowed Sinc Interpolation
5.2 Von Hann Window

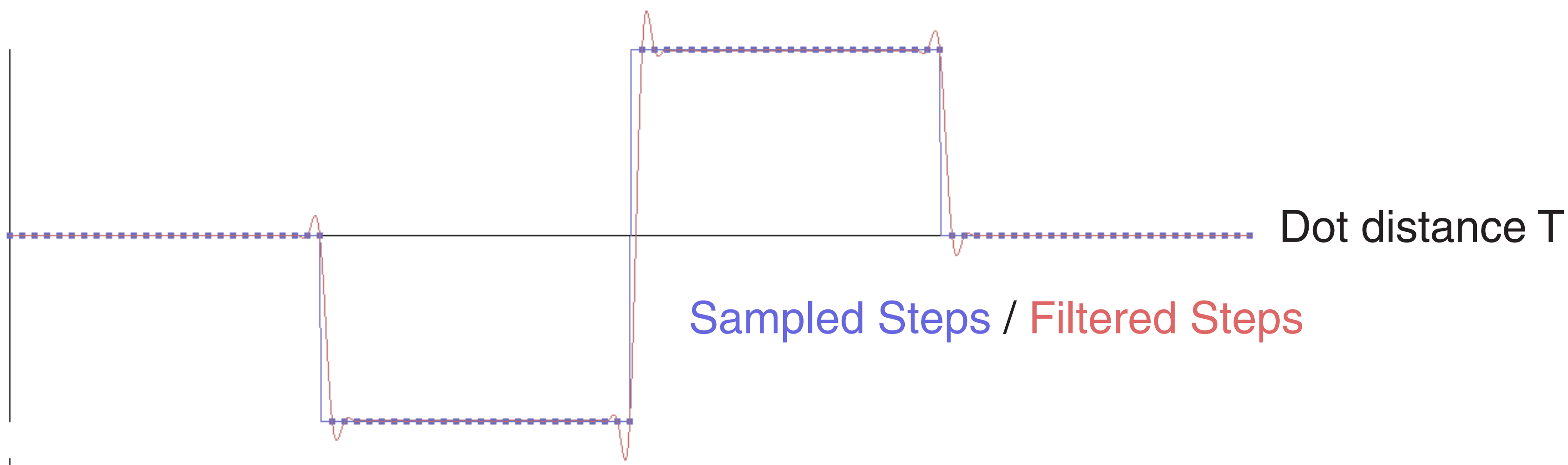
View 72 dpi / zoom 200%

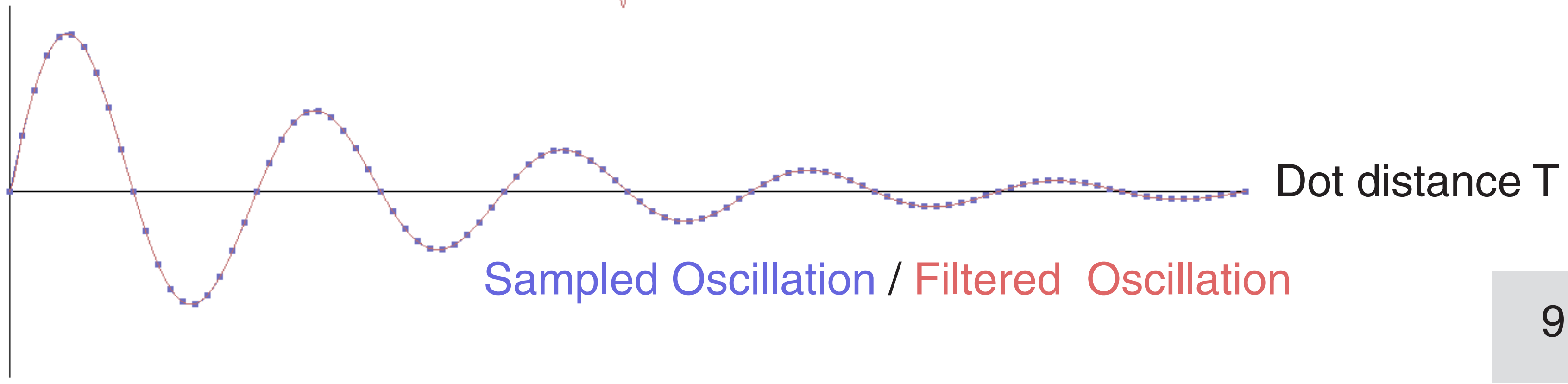






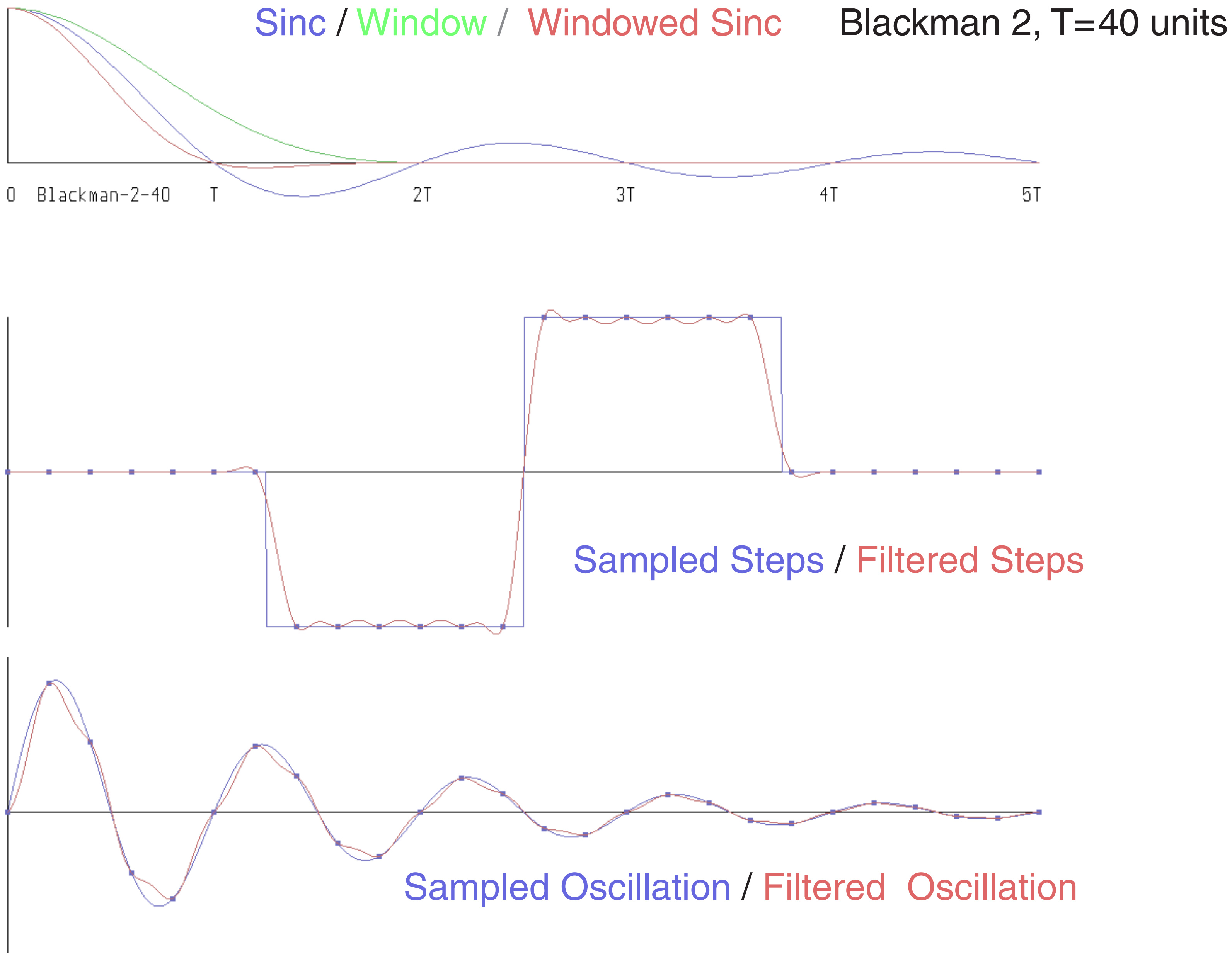


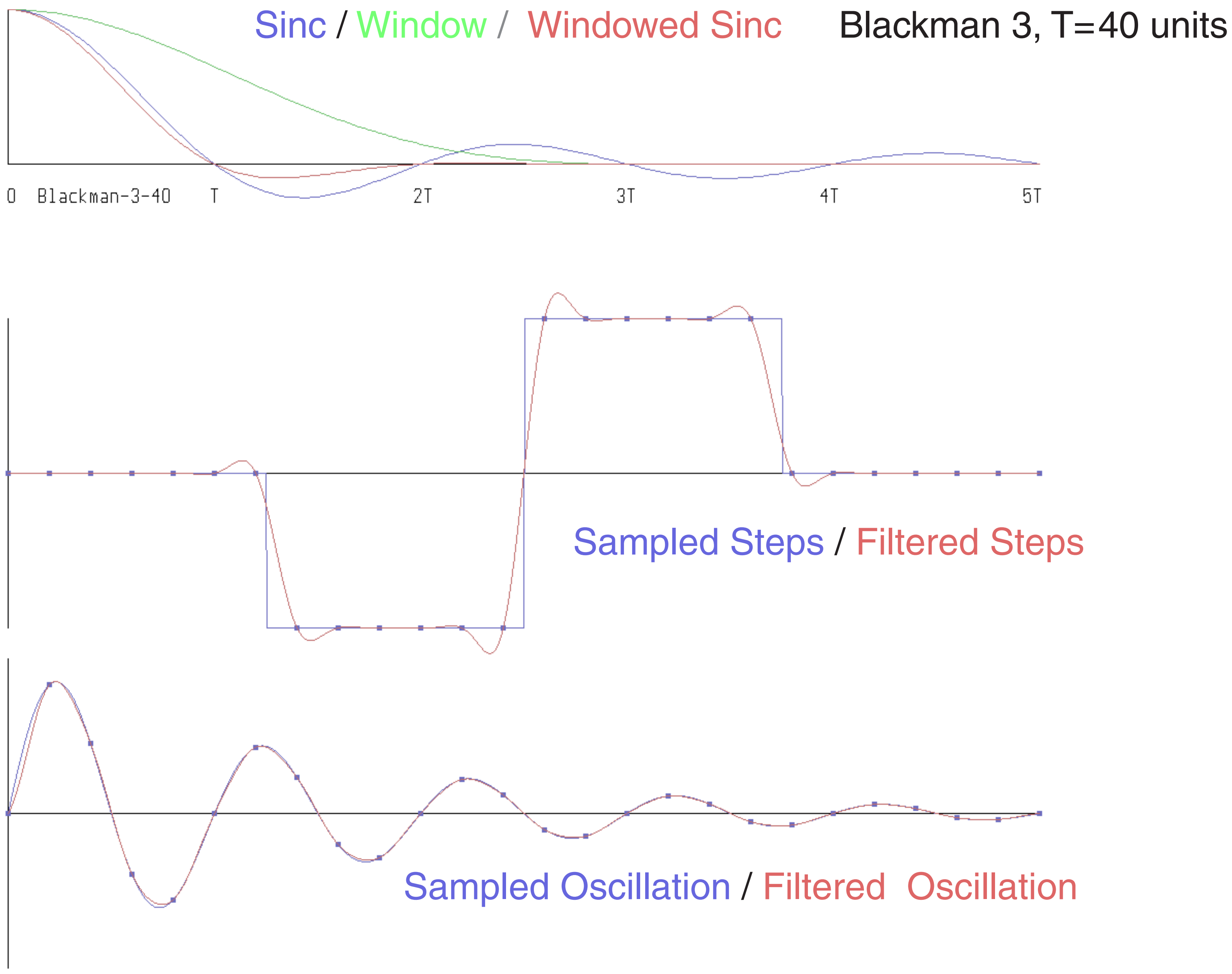




Windowed Sinc Interpolation
6.1 Blackman Window

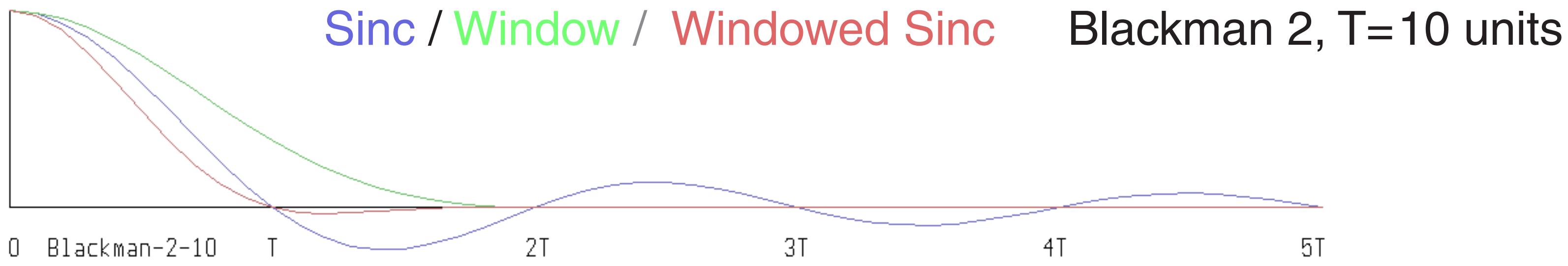
View 72 dpi / zoom 200%

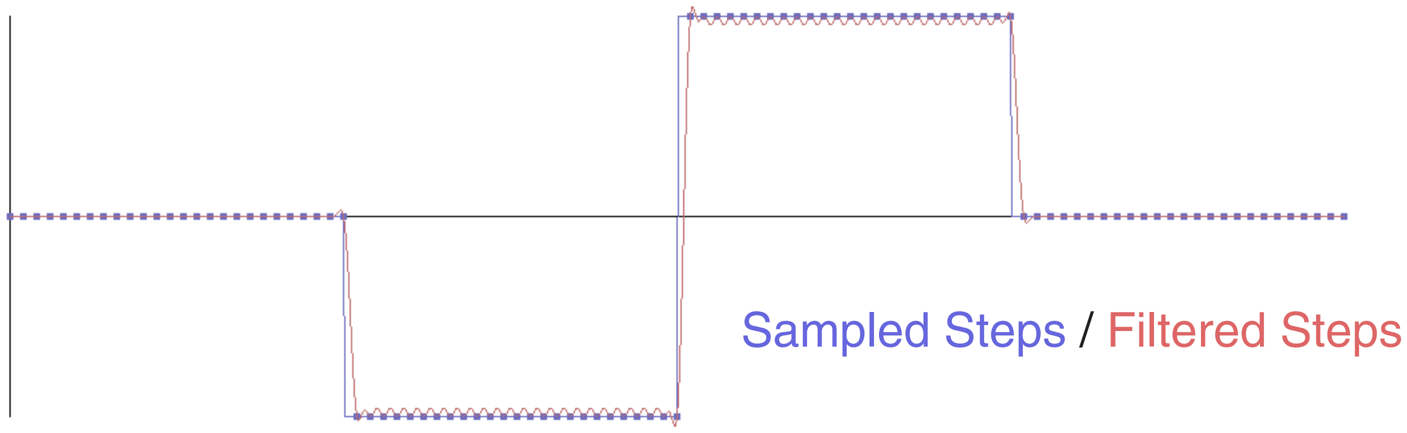


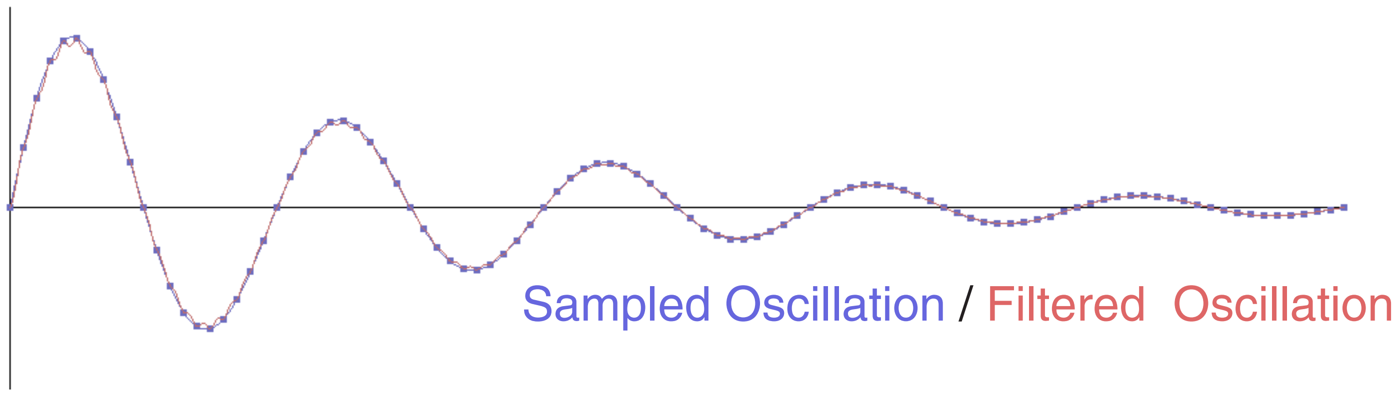


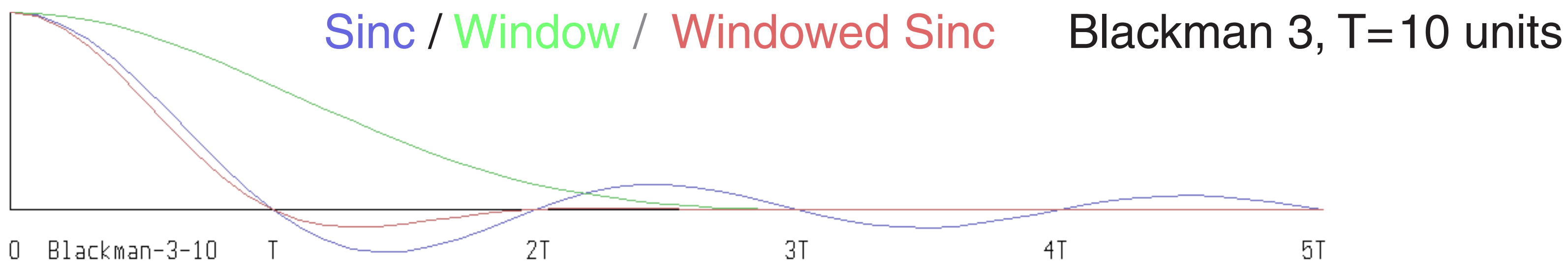
Windowed Sinc Interpolation
6.2 Blackman Window

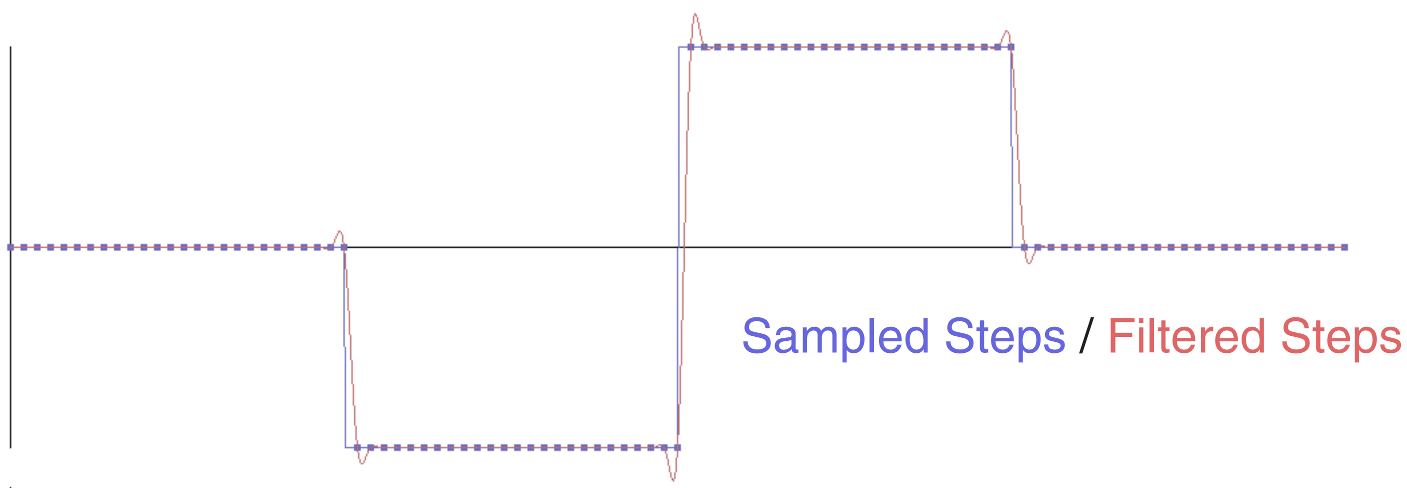
View 72 dpi / zoom 200%

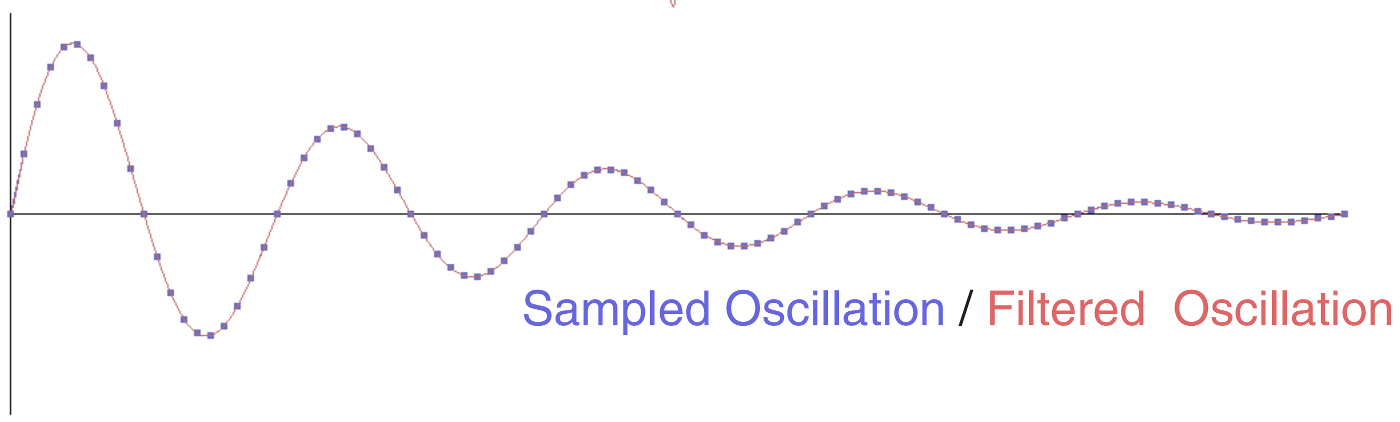












Windowed Sinc Interpolation
7.1 Code

```
Const M  = 1000;                     { For Function, x-pixels      }
      sc = 150;                      { Amplitude     y-pixels      }
      x0 = 100;                      { x-offset                    }
Var   fu,ff: Array[0..M] Of Double; { Function, Filtered Function }
      fs,fv: Array[0..M] Of Double; { Sinc, Windowed Sinc         }
      ti,t1,t2,y0,y1,y2 : Integer;
      T,Tw,N,sel,dt : Integer;
      AutoSave      : Boolean;
      ImagName      : String;
      FName         : String[16];
      pal,col       : Byte;

Procedure FillF (sel: Integer);
Var a: Double; na,nb,nc,k: Integer;
Begin
Case sel of
1: Begin
   na:= M Div 4; nb:= 2*na; nc:= 3*na;
   For ti:=0    to na Do fu[ti]:= 0;
   For ti:=na+1 to nb Do fu[ti]:=-1;
   For ti:=nb+1 to nc Do fu[ti]:=+1;
   For ti:=nc+1 to M  Do fu[ti]:= 0;
   End;
2: Begin
   For ti:=0 to M Do fu[ti]:=exp(-ti/300)*sic(pi*ti/100);
   End;
3: Begin  { Sinc }
   fs[0] :=1;
   For ti:=1 to M Do
    Begin
    a:=pi*ti/T;
    fs[ti]:=sic(a)/a;
    End;
   End;
10:Begin  { Rectangle Windowed Sinc }
   fv[0] :=1;
   For ti:=1 to M  Do fv[ti]:=0;
   For ti:=1 to Tw Do { Tw=5*T }
   Begin
    fv[ti]:=fs[ti];
   End;
   End;
11:Begin  { Lanczos Windowed Sinc }
   fv[0] :=1;
   For ti:=1 to M  Do fv[ti]:=0;
   For ti:=1 to Tw Do
   Begin
    a:=pi*ti/Tw;  { Tw=(2-3)*T }
    fv[ti]:=fs[ti]*sic(a)/a;
   End;
   End;
End;
```

7.2 Code

```
12:Begin  { von Hann Windowed Sinc }
   fv[0] :=1;
   For ti:=1 to M  Do fv[ti]:=0;
   For ti:=1 to Tw Do
   Begin
    a:=pi*ti/Tw;      { Tw=(2-3)*T }
    fv[ti]:=fs[ti]*0.5*(1+coc(a));
   End;
   End;
13:Begin  { Blackman Windowed Sinc }
   fv[0] :=1;
   For ti:=1 to M  Do fv[ti]:=0;
   For ti:=1 to Tw Do
   Begin
    a:=pi*ti/Tw;  { Tw=(2-3)*T }
    fv[ti]:=fs[ti]*(0.42+0.5*coc(a)+0.08*coc(2*a));
   End;
   End;
End; { Case }
End;

Procedure DoFilt;
Var s     : Double;
    k,kT,a: Integer;
Begin
 For ti:=0 to M-1 Do
 Begin
  kT:=((ti Div T)-N)*T;
  s:=0;
  For k:=-N to N Do
  Begin
   If (kT>=0) And (kT<=M) Then
   Begin
   a:=Abs(ti-kT);
   s:=s+fu[kT]*fv[a]; { Function*Windowed Sinc }
   End;
   Inc(kT,T);
  End;
  ff[ti]:=s;
  End;
End;
```

Windowed Sinc Interpolation
7.3 Code

```
Procedure DefFilt(ftype: Integer);
Var txt: String[3];
Begin
Case ftype Of
301: Begin FName:='Rectangle-5-40';T:=40; N:=5; Tw:=N*T; sel:=10;
     End;
303: Begin FName:='Rectangle-5-10';T:=10; N:=5; Tw:=N*T; sel:=10;
     End;
401: Begin FName:='Lanczos-2-40';  T:=40; N:=2; Tw:=N*T; sel:=11;
     End;
402: Begin FName:='Lanczos-3-40';  T:=40; N:=3; Tw:=N*T; sel:=11;
     End;
403: Begin FName:='Lanczos-2-10';  T:=10; N:=2; Tw:=N*T; sel:=11;
     End;
404: Begin FName:='Lanczos-3-10';  T:=10; N:=3; Tw:=N*T; sel:=11;
     End;
501: Begin FName:='von-Hann-2-40'; T:=40; N:=2; Tw:=N*T; sel:=12;
     End;
502: Begin FName:='von-Hann-3-40'; T:=40; N:=3; Tw:=N*T; sel:=12;
     End;
503: Begin FName:='von-Hann-2-10'; T:=10; N:=2; Tw:=N*T; sel:=12;
     End;
504: Begin FName:='von-Hann-3-10'; T:=10; N:=3; Tw:=N*T; sel:=12;
     End;
601: Begin FName:='Blackman-2-40'; T:=40; N:=2; Tw:=N*T; sel:=13;
     End;
602: Begin FName:='Blackman-3-40'; T:=40; N:=3; Tw:=N*T; sel:=13;
     End;
603: Begin FName:='Blackman-2-10'; T:=10; N:=2; Tw:=N*T; sel:=13;
     End;
604: Begin FName:='Blackman-3-10'; T:=10; N:=3; Tw:=N*T; sel:=13;
     End;
End; { case }
End;

Source code does not contain graphics procedures etc..
```

Windowed Sinc Interpolation
8.1 References

[1] PhilippeThévenaz + Thierry Blu + Michael Unser
Interpolaton Revisited
IEEE Transactions on Medical Imaging, Vol.19, 7.July 2000

[2] Erik Meijering
A Chronology of Interpolation: From Ancient Astronomy to Modern Signal and Image Processing
Proceedings of the IEEE, Vol.90, 3.March 2002

[3] Samuel D. Stearns
Digitale Verarbeitung analoger Signale
R.Oldenbourg Verlag München Wien, 1988

[4] Elmar Schrüfer
Signalverarbeitung
Carl Hanser Verlag München Wien, 1990

[5] Ken Turkowsky
Filters for Common Resampling Tasks
10.April 1990
http://www.worldserver.com/turk/computergraphics/ResamplingFilters.pdf

[6] Gernot Hoffmann
Bilinear, Biquadratic, Bicubic and Bicubic Spline Interpolation
http://docs-hoffmann.de/bicubic03042002.pdf

This document
http://docs-hoffmann.de/lanczos07112002.pdf