Discrete Fourier Transform and Windowing

The Fourier transform assumes a periodic function over the range [0, T] in time (or [0, λ] in space). What happens if the function is not actually periodic?

```
In[14]:= Clear["Global`*"]
```

Assume there is a periodic wave with period Tw, and you sample data by taking "npts" data points with a sample interval of Δt, for a total sample time of 'Tmax'. The Fourier transform assumes the function is periodic with period Tmax. If Tmax == (integer*) Tw, then the assumed period matches the actual period, and the Fourier transform works well. What happens if Tmax is not an integer number of cycles of Tw?

a. Periodic Signal

First, construct a case where Tmax == 100 Tw.

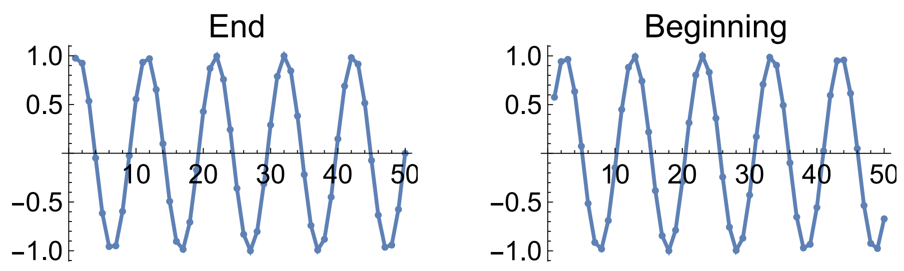
```
In[15]:= Δt = 1.00; (* milliseconds *)
         npts = 1024;
         Tmax = npts * Δt;
         Tw = Tmax / 100.0;
```

```
In[19]:= dataa =  Table[Sin[2 π t / Tw], {t, 1, Tmax, Δt}] (* data from part (a) *);
```

Show the last 50 and first 50 points. Since it is a periodic function, the end of the function maps smoothly onto the beginning.

```
In[25]:= GraphicsRow[{
           ListPlot[Take[dataa, -50], Joined → True, Mesh → All,
            LabelStyle → Larger, PlotStyle → PointSize[0.02], PlotLabel → "End"],
           ListPlot[Take[dataa, 50], Joined → True, Mesh → All,
            LabelStyle → Larger, PlotStyle → PointSize[0.02], PlotLabel → "Beginning"]
          }, ImageSize → Scaled[0.8] ]
```

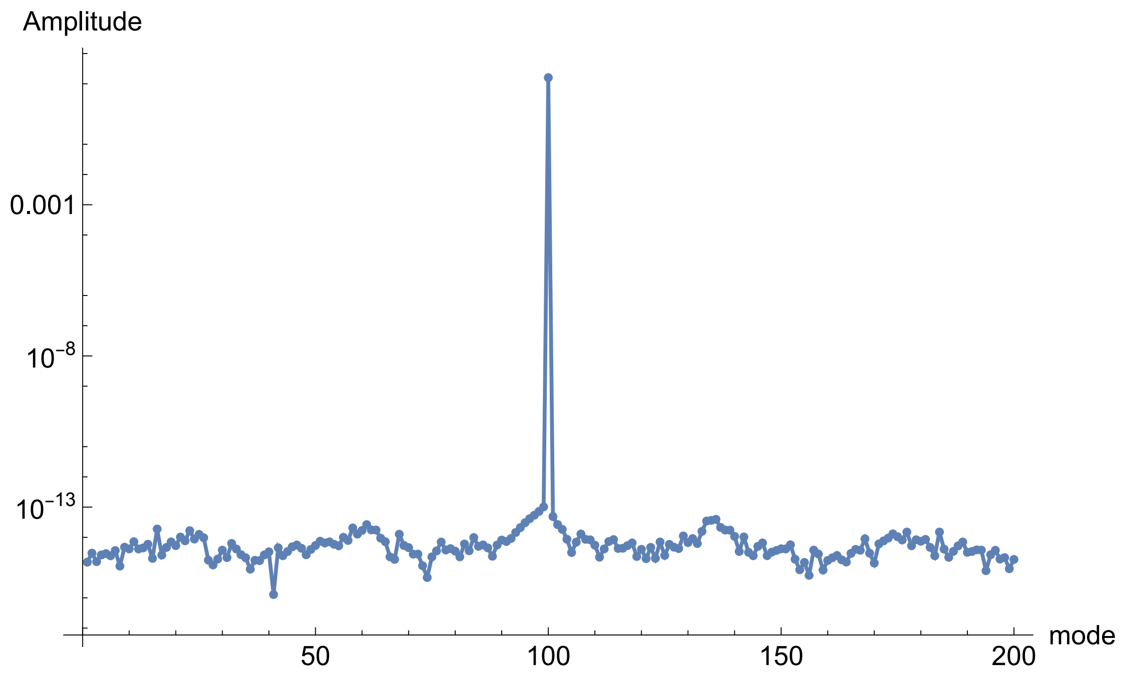
Out[25]=



Use the Fourier[ ] function to do the Fourier transform. Since the Fourier[ ] function returns the element for frequency '0' in slot '1' of the array, we have to shift the frequencies all left by 1, and we really only need the first few hundred in the array. There is a large peak at 100, corresponding to the fact that the period of the signal is 1/100 of the length of the data sample.

```
In[48]:= fta = Abs[Fourier[dataa]];
ftaAmp = RotateLeft[fta, 1]⟦Range[200]⟧;
ListLogPlot[ftaAmp, PlotRange → All, Joined → True, Mesh → All,
 LabelStyle → Larger, AxesLabel → {"mode", "Amplitude"}, ImageSize → Large]
```

Out[50]=



b. Periodic Signal with a non-integer number of periods in the sampling window.

Next, construct a case where Tmax == 100.5 Tw. Now the assumed periodicity of the Fourier analysis does not match the actual periodicity of the underlying function.

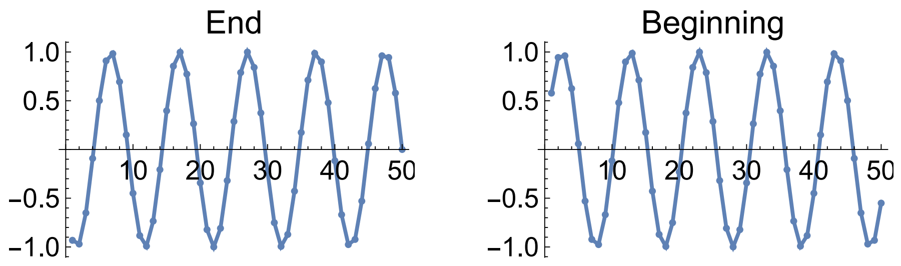
```
In[27]:= Δt = 1.00; (* milliseconds *)
npts = 1024;
Tmax = npts * Δt;
Tw = Tmax / 100.5;
```

```
In[31]:= datab = Table[Sin[2 π t / Tw], {t, 1, Tmax, Δt}] (* data for part (b) *);
```

Since Tmax is not an integer number of periods, the end of one segment does not join smoothly onto the beginning of the next.

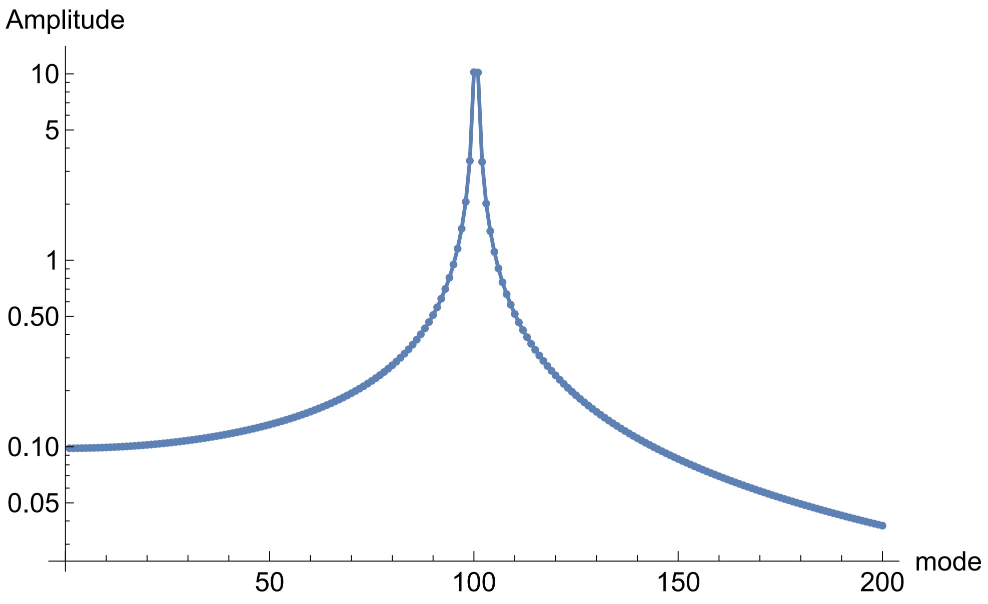
In[32]:=
```
GraphicsRow[{
  ListPlot[Take[datab, -50], Joined → True, Mesh → All,
   LabelStyle → Larger, PlotStyle → PointSize[0.02], PlotLabel → "End"],
  ListPlot[Take[datab, 50], Joined → True, Mesh → All,
   LabelStyle → Larger, PlotStyle → PointSize[0.02], PlotLabel → "Beginning"]
 }, ImageSize → Scaled[0.8]]
```

Out[32]=



In[51]:=
```
ftb = Abs[Fourier[datab]];
ftbAmp = RotateLeft[ftb, 1]⟦Range[200]⟧;
ListLogPlot[ftbAmp, PlotRange → All, Joined → True, Mesh → All,
 LabelStyle → Larger, AxesLabel → {"mode", "Amplitude"}, ImageSize → Scaled[0.8]]
```

Out[52]=



There is clearly a much wider range of frequencies present. Note the amplitude on the vertical scale as well. The amplitude is much higher away from the peak. (The peak height itself is about the same -- the scale on the previous graph covers a much wider range. The dominant frequency is still around 100 (we expect 100.5), but the sharp transition between segments clearly has a wide range of frequencies.

c. Windowing

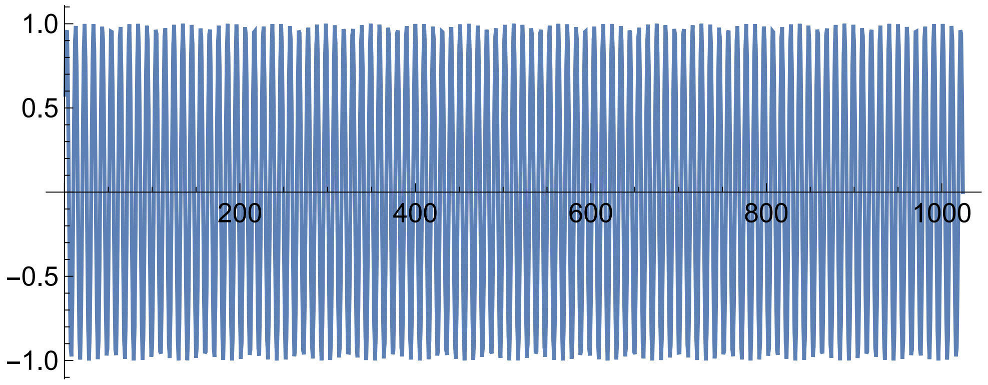
When performing Fourier analysis on physical data, we usually can’t assume Tmax == (integer) Tw. The instrument used to acquire the data sets Δt and the number of data points, and hence Tmax =

(npts) * Δt, while the underlying physical phenomenon being studied sets Tw. To avoid the artificial discontinuity at Tmax, windowing is often applied. There are various approaches, but one of the simplest is to multiply the signal by a very slowly varying envelope (or window) that goes to zero at the end points. This introduces a fake very low frequency component, but smooths out the high frequency glitch.

Here is the original data from part b. Note the discontinuity between the beginning and the end.

```
In[34]:= ListPlot[datab, Joined → True, AspectRatio → 2 / 5,
          LabelStyle → Larger, ImageSize → Scaled[0.8]]
```

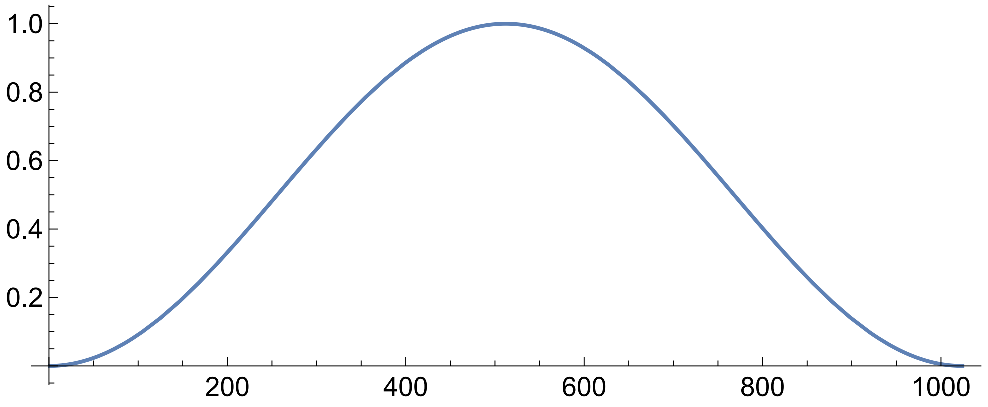
Out[34]=



This is a simple cosine-shaped window. It is zero at the endpoints but rises smoothly to 1 at the center. (It is also sometimes known as a Hanning Window.)

```
In[35]:= win[x_, xmax_] := (1 - Cos[2 π x / xmax]) / 2
         Plot[{win[x, 1024]}, {x, 1, 1024},
          AspectRatio → 2 / 5, LabelStyle → Larger, ImageSize → Scaled[0.8]]
```

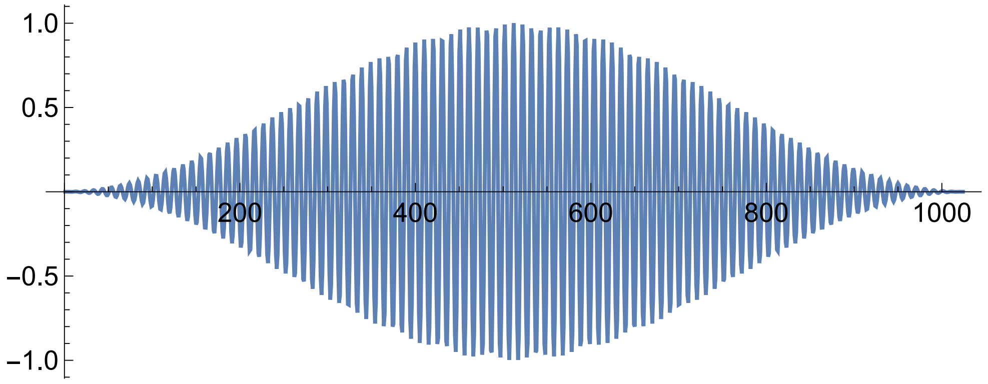
Out[36]=



Multiplying the data from part (b) by the window gives a new set of windowed data.

In[37]:=
```
wdata = Table[datab⟦x⟧ * win[x, npts], {x, 1, npts}] (* Windowed data *);
wdplot = ListPlot[wdata, Joined → True,
  AspectRatio → 2 / 5, LabelStyle → Larger, ImageSize → Scaled[0.8]]
```

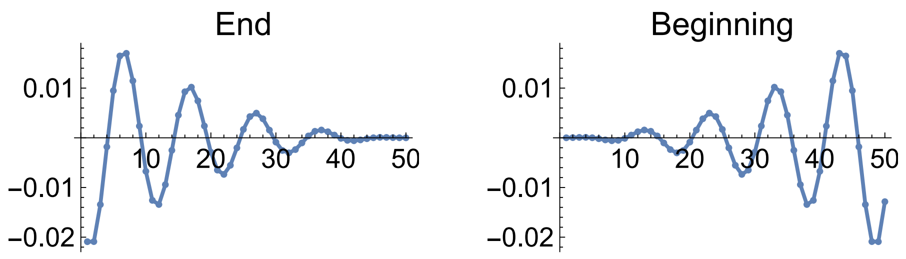
Out[38]=



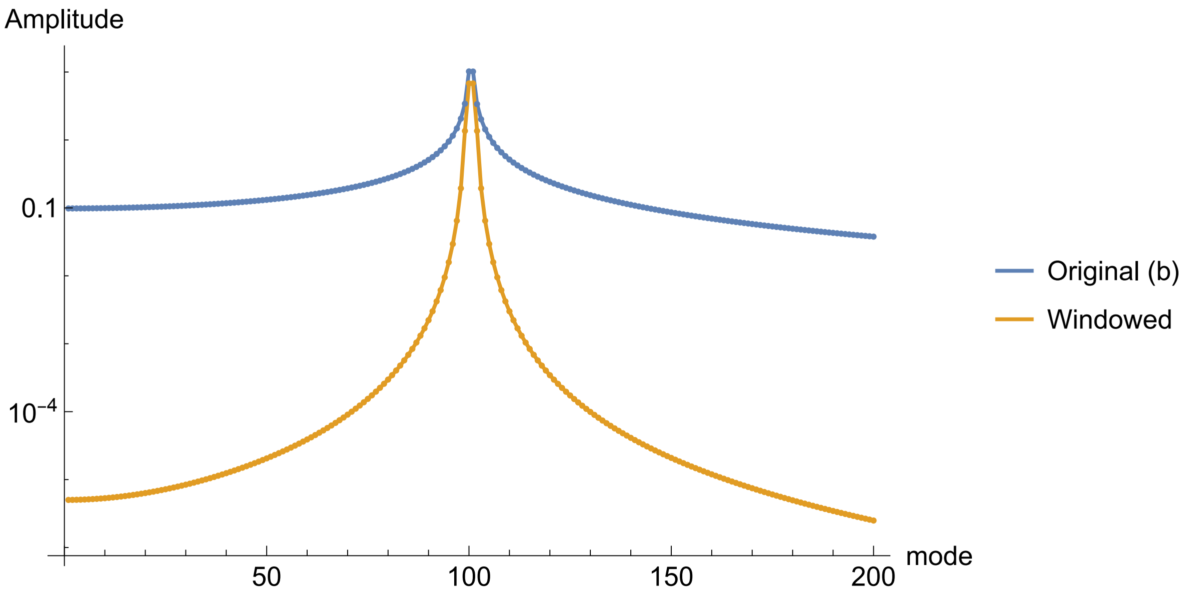
With the windowing, the end of the data maps smoothly onto the beginning, at the expense of introducing some very long wavelength (low frequency) Fourier components.

In[39]:=
```
GraphicsRow[{
  ListPlot[Take[wdata, -50], Joined → True, Mesh → All,
   LabelStyle → Larger, PlotStyle → PointSize[0.02], PlotLabel → "End"],
  ListPlot[Take[wdata, 50], Joined → True, Mesh → All,
   LabelStyle → Larger, PlotStyle → PointSize[0.02], PlotLabel → "Beginning"]
 }, LabelStyle → Larger, ImageSize → Scaled[0.8]]
```

Out[39]=



```
In[45]:= ftw = Abs[Fourier[wdata]];
         ftwAmp = RotateLeft[ftw, 1]⟦Range[200]⟧;
         ListLogPlot[{ftbAmp, ftwAmp}, PlotRange → All, Joined → True,
          Mesh → All, LabelStyle → Larger, AxesLabel → {"mode", "Amplitude"},
          PlotLegends → {"Original (b)", "Windowed"}, ImageSize → Scaled[0.8]]
Out[47]=
```



Comparing just (b) and (b) with a window, the window has clearly helped identify the important frequency in this data set.

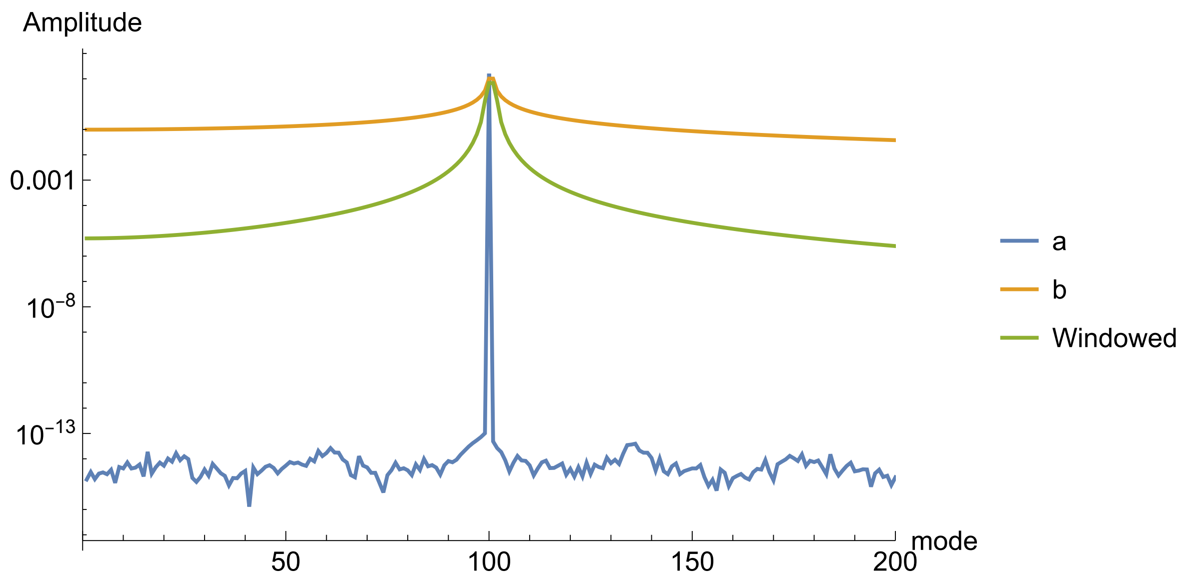
Comparison of data from (a), (b), and windowed (b)

Clearly the smooth data in (a) has the sharpest transform. However, the windowed data does show a slightly sharper peak, and has significantly suppressed the power at frequencies far from 100. (You can also see from the graph that the new dominant mode is a bit more than 100. Of course we expect that since we used Tmax/100.5 in part (b) compared to Tmax/100.0 in part (a).)

Any form of windowing involves tradeoffs, depending on the nature of the incoming signal and on the type of analysis you wish to perform. Commercial signal analyzers typically offer several different window shapes to optimize measuring different pieces of information (such as peak location, peak width, peak height, and background noise level).

In[53]:=
```
ListLogPlot[{ftaAmp, ftbAmp, ftwAmp}, Joined → True,
 PlotRange → {{0, 200}, All}, PlotLegends → {"a", "b", "Windowed"},
 LabelStyle → Larger, AxesLabel → {"mode", "Amplitude"}, ImageSize → Scaled[0.8]]
```

Out[53]=


Amplitude
10^{-8}
10^{-13}
mode
a
b
Windowed