```
Fourier[list]
InverseFourier[list]

N1=256
data = Table[N[Sin[20 2 Pi n/N1]], {n, N1}]; ← pure sin wave
ListLinePlot[data]                                        up aliased negative freq
Data=Fourier[data];
ListLinePlot[Abs[Data],PlotRange->All] ← see results at 20 & 256-20

data = Table[N[Sin[20 2 Pi n/N1] + (RandomReal[] - 1/2)], {n, N1}];
ListLinePlot[data]                          → add random noise
Data=Fourier[data];
ListLinePlot[Abs[Data],PlotRange->All] ← see previous peaks + noise at all frequencies

data = Table[N[Mod[n,20]], {n, N1}]; ← sawtooth wave - has discontinuity
ListLinePlot[data]
Data=Fourier[data];
ListLinePlot[Abs[Data],PlotRange->All] ← lots of strong harmonics due to discontinuity
Data[[1]] ← 0th harmonic is DC signal; to remove it subtract average

md=Mean[data]
data = Table[N[Mod[n,20]-md], {n, N1}];
ListLinePlot[data]
Data=Fourier[data];
ListLinePlot[Abs[Data],PlotRange->All];
Data[[1]] ← now zero

data = Table[Abs[N[Mod[n,40]-20]]-10, {n, N1}]; ← triangle wave; no discontinuity
ListLinePlot[data]
Data=Fourier[data];
ListLinePlot[Abs[Data],PlotRange->All]; ← weaker harmonics; Nyquist looks OK

data = Table[N[Mod[n,20]-md], {n, N1}];
data2=Table[Sum[data[[k]],{k,Max[1,i-5],i}]/5,{i,N1}]; ← running sum of saw tooth
ListLinePlot[data2];
Data2=Fourier[data2];
ListLinePlot[Abs[Data2],PlotRange->All]; ← reduced high freq harmonics

data = Table[N[Sin[20 2 Pi n/N1] + (RandomReal[] - 1/2)], {n, N1}];
data2=Table[Sum[data[[k]],{k,Max[1,i-5],i}]/5,{i,N1}];
ListLinePlot[data2];                          ↑ running sum average     ↑ sin wave + noise
ListLinePlot[data];
Data2=Fourier[data2];
ListLinePlot[Abs[Data2],PlotRange->All]; ← reduced high freq noise
Data=Fourier[data];
ListLinePlot[Abs[Data],PlotRange->All]
```

```
recursion formula uses results from http://www-users.cs.york.ac.uk/~fisher/mkfilter/trad.html
for Butterworth, 5th order, corner=1000, sampling=10000
note: y[[ ]] refers to an entry number in a Table (or array) ...the entry number is a whole nu
mber 1 to max
      yy[ ] is a function of a real (or even complex) number


y=Table[0,{i, 50}]
x[n_]=UnitStep[n-6]
```

step ; y will be filtered response to step

```
Do[
y[[n]] = 1 x[n- 5] + 5 x[n- 4] + 10 x[n- 3] + 10 x[n- 2] + 5 x[n- 1] + 1 x[n- 0] +
 0.1254306222 y[[n- 5]] + -0.8811300754 y[[n- 4]] + 2.5452528683 y[[n- 3]] + -3.8060181193 y[[
n- 2]] + 2.9754221097 y[[n- 1]],
{n,6,50}]
ListPlot[y, Joined->True, PlotRange->All]
```

digital filter

```
w=.5
```

a freq "passed" by this low pass filter

```
x[n_]=Exp[I w n]
y=Table[0,{i, 100}]
Do[
y[[n]] = 1 x[n- 5] + 5 x[n- 4] + 10 x[n- 3] + 10 x[n- 2] + 5 x[n- 1] + 1 x[n- 0] +
 0.1254306222 y[[n- 5]] + -0.8811300754 y[[n- 4]] + 2.5452528683 y[[n- 3]] + -3.8060181193 y[[
n- 2]] + 2.9754221097 y[[n- 1]],
{n,6,100}]
ListPlot[Re[y], Joined->True, PlotRange->All]
```

see amplified ; note "start up"

```
w=1.5
```

a freq "blocked" by this low pass filter

```
x[n_]=Exp[I w n]
y=Table[0,{i, 60}]
Do[
y[[n]] = 1 x[n- 5] + 5 x[n- 4] + 10 x[n- 3] + 10 x[n- 2] + 5 x[n- 1] + 1 x[n- 0] +
 0.1254306222 y[[n- 5]] + -0.8811300754 y[[n- 4]] + 2.5452528683 y[[n- 3]] + -3.8060181193 y[[
n- 2]] + 2.9754221097 y[[n- 1]],
{n,6,60}]
ListPlot[Re[y], Joined->True, PlotRange->All]
```

much surpressed cf w=.5
note "start up"

```
ClearAll[w]
xx[n_]=Exp[I w n]
yy[n_]=Exp[I w n]
```

the amplitude for any freq

```
A[ww_]= (1 xx[- 5] + 5 xx[- 4] + 10 xx[- 3] + 10 xx[- 2] + 5 xx[- 1] + 1 xx[- 0] )/
(1-(0.1254306222 yy[- 5] + -0.8811300754 yy[- 4] + 2.5452528683 yy[- 3] + -3.8060181193 yy[- 2
] + 2.9754221097 yy[- 1]))

Plot[Abs[A[w]],{w,.01,Pi}]

Show[GraphicsArray[{{%9,%14},{%4,%19}}]]
```

passed freq + noise

```
w=.5
ClearAll[x]
x=Table[Exp[I w n]+(RandomReal[] - 1/2),{n, 100}];
x[[1]]=0; x[[2]]=0; x[[3]]=0; x[[4]]=0; x[[5]]=0;
ListPlot[Re[x], Joined->True, PlotRange->All]
y=Table[0,{i, 100}]
Do[
y[[n]] = 1 x[[n- 5]] + 5 x[[n- 4]] + 10 x[[n- 3]] + 10 x[[n- 2]] + 5 x[[n- 1]] + 1 x[[n- 0]] +
 0.1254306222 y[[n- 5]] + -0.8811300754 y[[n- 4]] + 2.5452528683 y[[n- 3]] + -3.8060181193 y[[
n- 2]] + 2.9754221097 y[[n- 1]],
{n,6,100}]
ListPlot[Re[y], Joined->True, PlotRange->All]
```

Digital Filters

Finite Impulse Response (non recursive)

response is for a finite time

$\underbrace{\text{Impulse}}$ → input is zero except at a single time

output $y_k = a_0 x_k + a_1 x_{k-1} + a_2 x_{k-2} + \cdots + a_n x_{k-n}$

eg "running average" where as are constant $= \frac{1}{n+1}$

Note: seek response to a unit step & a unit sinusoidal at some freq ω

Infinite Impulse Response (recursive)

output $y_k = a_0 x_k + a_1 x_{k-1} + \cdots + a_n x_{k-n}$
$\quad - (b_1 y_{k-1} + b_2 y_{k-2} + \cdots b_m y_{k-m})$

Just as a damped driven harmonic oscillator has a homogeneous solution (that decays & depends on initial conditions) and a steady particular solution that does not depend on initial conditions same applies here to digital filters – if $x_k = e^{i\omega k}$ long term $y_k = A e^{i\omega k}$ where A will depend on the frequency.

$$A\left(e^{ik\omega} + b_1 e^{i(k-1)\omega} + b_2 e^{i(k-2)\omega} + \cdots + b_m e^{i(k-m)}\right)$$

$$= a_0 e^{ik\omega} + a_1 e^{i(k-1)\omega} + a_2 e^{i(k-2)\omega} + \cdots + a_n e^{i(k-n)\omega}$$

$$\Rightarrow A = \frac{a_0 + a_1 e^{-i\omega} + \cdots + a_n e^{-in\omega}}{1 + b_1 e^{-i\omega} + b_2 e^{-2i\omega} + b_m e^{-mi\omega}}$$

so can use above to find amplitude & phase of output.