Given an intensity time series of N points, with sampling interval of τ, containing a pulsar signal with period P, write an algorithm for folding this signal, in python.

```python
import numpy as np
import math

# Whats given to us:
# a numpy array named time_series containing all the sample values
# the sampling interval tau
# the period of the pulsar P

N = np.size(time_series)
num_bins = int(P/tau)                                   #for native resolution
bin_interval = P/num_bins                               #time span of each bin

bins = np.zeros((num_bins,math.ceil(N/num_bins)*2))  #this array will contain the
                                                        #bins and their contents. Each
                                                       #row is represented by one bin.

folded = np.zeros(num_bins)                              #this array will contain the
                                                        #average value for each bin

bin_count = np.zeros(num_bins,dtype=int)           #this keep track of how many
                                                      #samples we have added to each bin

#For every sample, we have to decide which bin it goes to. The idea is that the bin
#which contains more than half portion of a sample will be the bin where it goes.

for i in range(len(time_series)):

    bin_index = math.floor(((i * tau) % P) / bin_interval)        #we have to decide
                                                        #whether the sample goes to bin with
                                                        #index bin_index or the previous bin

    f = (((i * tau) % P) - bin_index * bin_interval) / tau  #calculates the
                                                        #fraction of the sample contained in
                                                        #the bin with index bin_index.
                                                        #f could be greater than 1 in case
                                                        #the sample lies completely inside
                                                        #the mentioned bin.

    if f > = 0.5:                                       #if more than half of the sample lies
                                                    #in the bin with index bin_index,
                                                    #then it goes there.
            bins[bin_index,bin_count[bin_index]] = time_series[i]
            bin_count[bin_index] += 1


```

```
46      else:                                           #else the sample should go to the bin
47                                                  #whose index is bin_index-1
48              bins[bin_index-1,bin_count[bin_index-1]] = time_series[i]
49              bin_count[bin_index-1] += 1
50
51  #if bin_index is 0 and f < 0.5 then the sample should go to the last bin. This is
52  #taken care of here since in python the index -1 means the last element of the
53  #array.
54
55  for i in range(num_bins):
56          folded[i] = np.sum(bins[i])/bin_count[i]    #sum all values of each bin and
57                                                       #divide by number of samples added
58                                                        #to each bin
```

How much error can one tolerate in the estimate of (i) P and (ii) τ keeping the smearing of the pulse in the folded profile within some limit. Test these results on gptool by varying P and τ.

(i) The modulo operation (%) in line number 28 of the code above essentially slices up the time series into chunks of length P. When folding is done with an accurate value of P, sample values having the same phase get added to the same bin. But say, if the estimate of period P' is greater than P by an amount Δ, then the chunks are of length P'. This means that the pulses of 2 consecutive chunks will be shifted wrt each other by Δ and summing them up will broaden the pulse by Δ. If our time-series has N pulses, successively adding the left-shifted pulses like this N times would broaden the folded pulse by N$\cdot\Delta$. Δ could be positive or negative so we write $|\Delta|$ from now, instead. Hence the fractional error in the pulse width, say X will be given by $X = \frac{N \cdot |\Delta|}{\rho}$, where ρ is the true pulse width.

So if we want to keep the smearing of the pulse below X fraction of the pulse, then we need to keep the fractional error in the time period ($\frac{|\Delta|}{P}$), below $\frac{\rho \cdot X}{N \cdot P}$ – (i).

This can be tested on gptool by using a value of P which is slightly different from the true value and then measuring the smearing of the pulse as more and more data is folded. The table below shows the amount of fractional smearing of the pulse wrt the amount of data folded, when the value of P is taken to be 382.295240142 ms instead of the accurate value 381.295240142. In the third column are the values of X predicted by the relation (i).

Amount of data folded (seconds)	Measured X	Predicted X
10	1	1.10
20	2.2	2.20
30	3.4	3.31
40	4.4	4.41
50	5.8	5.51
60	7	6.61
70	8.2	7.71
80	9.2	8.82
90	10.2	9.92
100	11.4	11.02

The measured values of the fractional smearing are quite close to the ones predicted by relation (i), especially given the rough method of measuring the smearing that was used.

(ii) Consider the folding scenario with accurate value of τ. Let i be the second sample put in the first bin after sample 1. Thus these two samples have the same phase (approximately). However if the value of τ is less than this, then the value of $(i \cdot \tau)$ `&` P in the code line number `28` will be slightly less than P which will put the sample in one of the last bins instead of the first bin. Same goes for the samples immediately next to the sample i.

This example shows that having an under-estimation of τ results in each cycle having more samples, taking some samples from the next cycle. This results in the pulse getting shifted towards the left wrt the pulse in the previous cycle. Folding in this case would thus broaden the pulse. To calculate the amount of broadening of a 2-folded pulse for an under-estimation of Δ in τ, we find out the number of extra samples that are put in a cycle. This number is given by $(\frac{P}{\tau-\Delta} - \frac{P}{\tau})$, where Δ is positive. And so the amount of broadening after folding N pulses would be $(\frac{P}{\tau-\Delta} - \frac{P}{\tau})\,\tau \cdot N = P \cdot \left(\frac{\Delta}{(\tau-\Delta)\,\tau}\right) \tau \cdot N = \frac{P \cdot \Delta \cdot N}{\tau-\Delta}$. The fractional smearing X will be given by -

$$X = \frac{P \cdot \Delta \cdot N}{(\tau - \Delta) \cdot \rho} \tag{1}$$

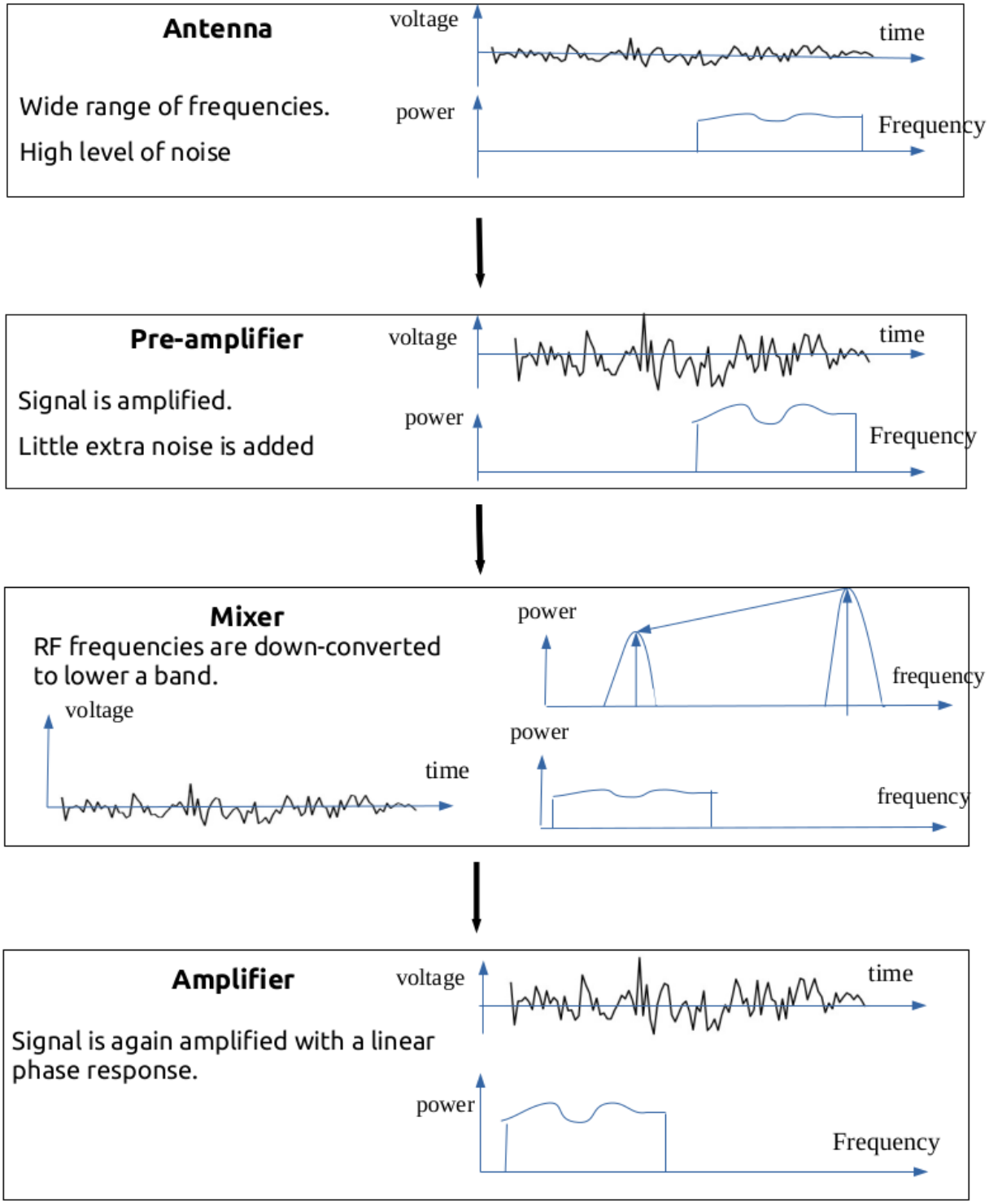
where ρ is the true pulse width. On rearranging this, we find that if we want to keep the smearing of the pulse below X fraction of the pulse, then we need to keep the fractional error in sampling interval $(\frac{|\Delta|}{\tau})$ below $\frac{\rho \cdot X}{\rho \cdot X + N \cdot P}$ – (ii)

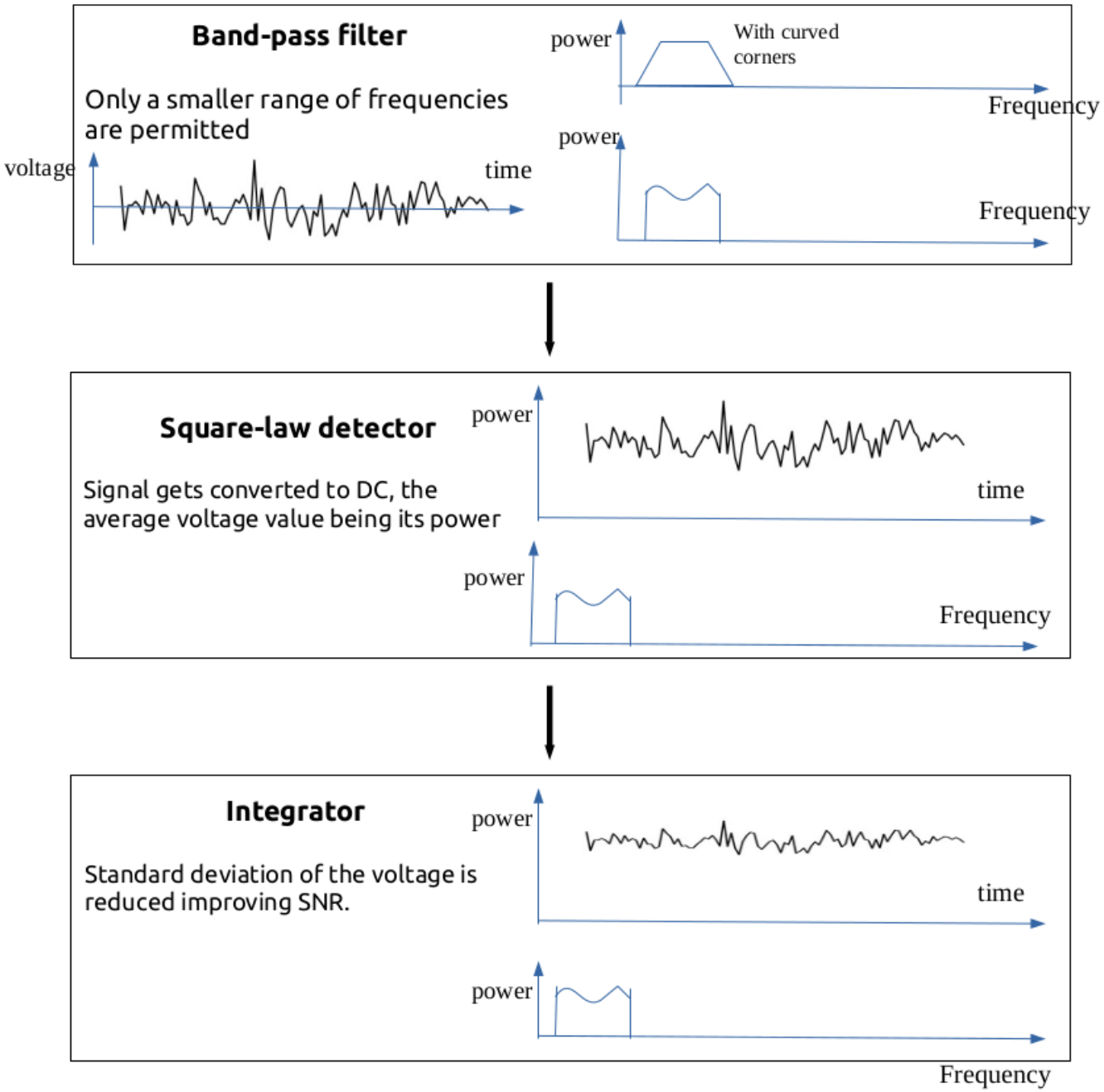
This can be tested on gptool by using a value of τ which is slightly different from the true value and then measuring the smearing of the pulse as more and more data is folded. The table below shows the amount of fractional smearing of the pulse wrt the amount of data folded, when the value of τ is taken to be 0.01022 ms instead of the accurate value 0.01024. In the third column are the values of X predicted by the relation (ii).

Amount of data folded (seconds)	**Measured X**	**Predicted X**
10	1	.81
20	1.8	1.63
30	2.4	2.44
40	3.2	3.25
50	3.8	4.07
60	5.2	4.88
70	6	5.69
80	6.8	6.50
90	8	7.32
100	9	8.13

Again the measured values of the fractional smearing are quite close to the ones predicted by relation (ii), especially given the rough method of measuring the smearing that was used.

Add power spectrum plots to the block diagram of the signal processing line discussed in the meeting.



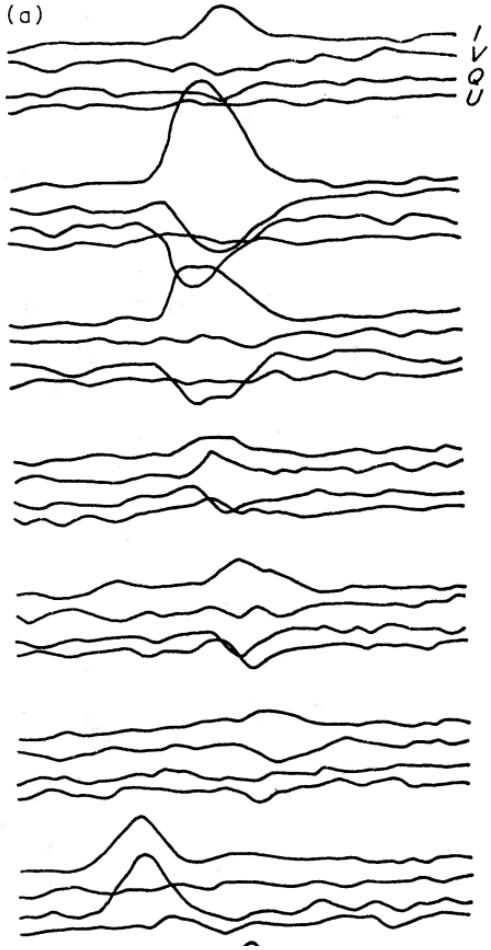


How to derive the mathematical form of the sampled data of a signal whose functional form is known to be, say f(t).

The sampled form of f(t), s(t) at sampling frequency $\frac{1}{\tau}$ is given by the function $s(t) = f(t) \cdot D_\tau(t)$ where D_τ is the function (called 'Dirac Comb') represented by an infinite sequence of dirac delta functions separated by τ. The functional form of s(t) will be such that it will be zero at all values of t except those satisfying $t = k \cdot \tau$ for some integer k. At those points s(t) will be equal to f(t).

To read about the differences between average (folded) profiles and the individual pulses of a pulsar.

Individual pulses of a pulsar show significant variation in their properties, while the average profiles over many many pulses are stable (ignoring effects like nulling). As an example, we have 7 individual pulses of the pulsar PSR 0329 + 54 in the figure on the right (Lyne et al. 1971). For each pulse we have the variation of 4 quantities I, V, Q, U which can be loosely described as total intensity, the circularly polarized component, and the two orthogonal components of the linear polarization respectively.



Considering the quantity I, we can see variation in the phase at which the pulse occurs, in addition to the magnitude of the intensity. There is also variation in the width of the individual pulses. In addition to I, the polarization also shows variation.

In other pulsars, we also see individual pulses being composed of 'sub-pulses', whose width can be many times smaller than the average integrated profile. In some pulsars, the phase of occurence of the subpulses drift systematically across the pulse length. The intensities of these individual subpulses can also vary in a complicated way.