Lab 6: Match filtering in Digital Modulation and Implementation of QPSK Modulation Scheme

Wadhwani Electronics Lab

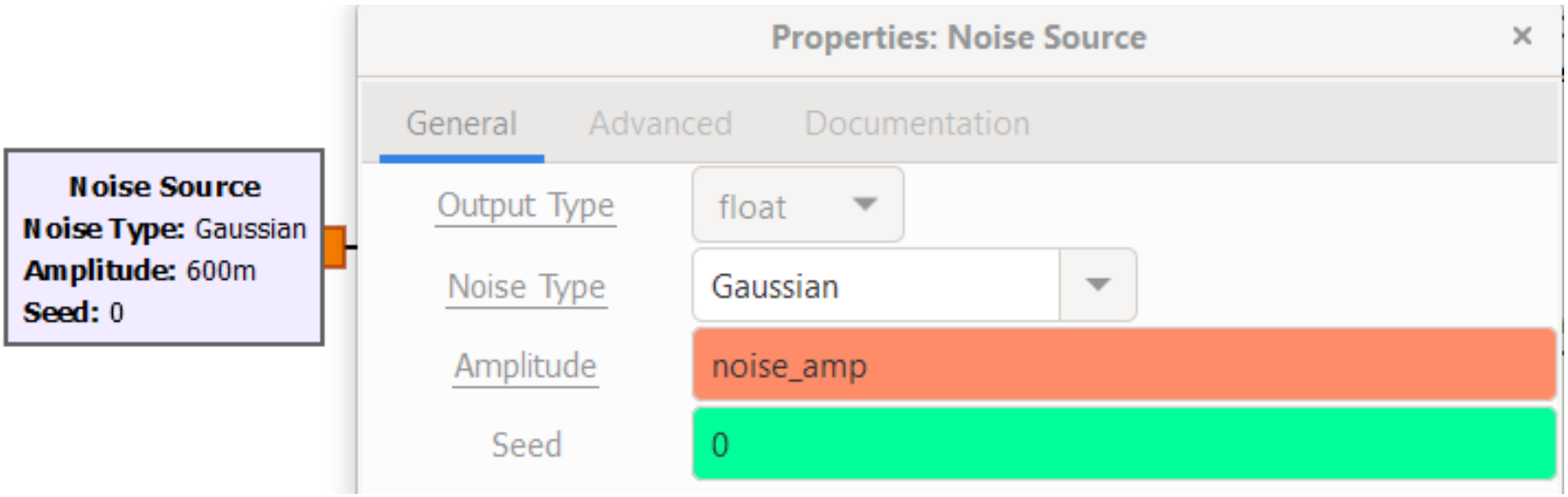
Department of Electrical Engineering
Indian Institute of Technology, Bombay.

Aim of the experiment

1. To study the effect of Noise on digital modulation schemes using the optimal receiver.
2. To understand the use of a matched filter when the signal propagates through a noisy channel.
3. To understand QPSK Modulation and Demodulation scheme

Lab Task 1: Noise impact on digital modulation scheme with the optimal receiver (experiment 5, contd)

- Add Gaussian noise to the digital modulated BPSK signal before demodulation i.e., representing channel noise (experiment 5, contd)



- Demodulate the above signal,observe the output signal, and store the text file using the "File sink" block
- Demonstrate your observation to the respective TA

Note - Vary the noise amplitude in the range of 0 to 1

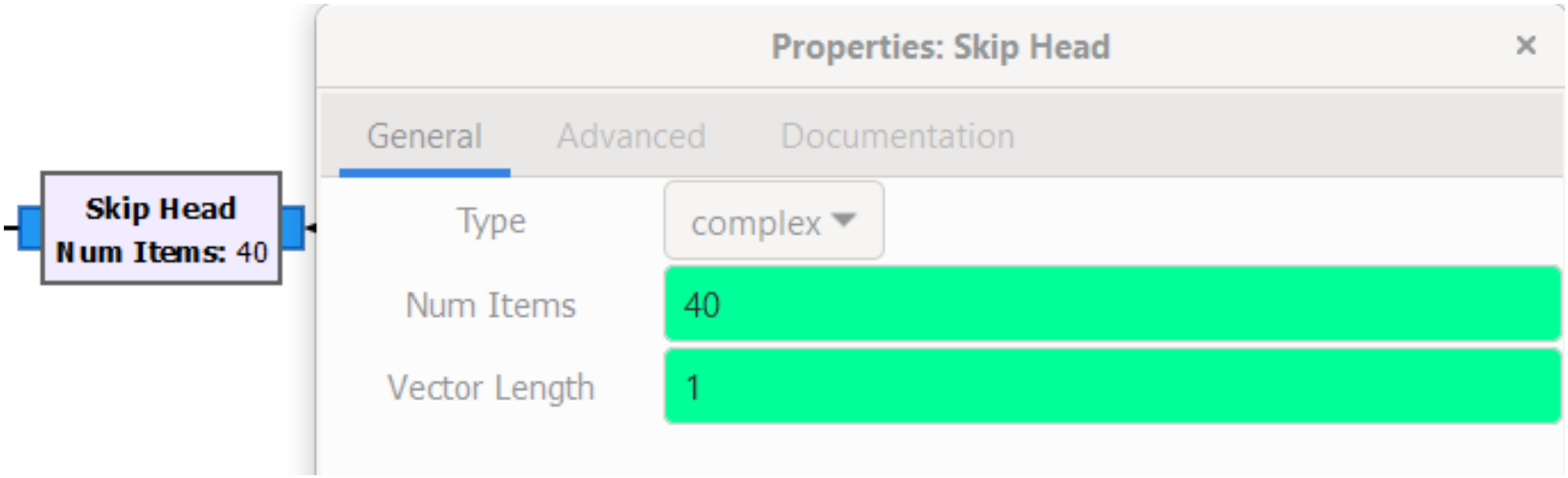
Lab Task 2: Matched Filter Implementation

- Implement BPSK transmitter to transmit the the real message signal with the carrier of 100 KHz (same as in exp. 5)
- Add Noise source (Gaussian) to the modulated BPSK signal
- Demodulate the received data using matched filtering at the receiver end
  - To implement the matched filtering at the BPSK receiver use **RRC** filter after down-converting to the appropriate sampling rate
  - Set the variables of Root-Raised Cosine filter(matched filter) same as that of RRC filter used at transmitter end
  - Use the "skip head" block to remove initial garbage values added by matched filtering followed by "Decimating FIR filter"

  Q:- What happens to SNR after matched filtering?
- Extract ASCII value from received data and store the result using the "File Sink" block to a text file

Skip Head

Skip the first N items, from then on copies items to the output. Useful when there are metadata or junk at the start. The processing in the GNU Radio adds some junk values in the start. If we don't use this then packing bits in byte format will cause wrong packing of ASCII code.



Note : Since the number of junk bits added depends on your flow and the modulation scheme, you need to do iterative trial to get the correct value.

Lab task 3: QPSK Modulation scheme (Transmitter end)

- Implement a QPSK-modulation transmitter to transmit a real message signal using blocks available in GNU Radio
  - For this use the input "Text file" as input using the "File source" block
  - Generate QPSK symbols by first unpacking 8 bits and then packing 2 bits to obtain "00", "01", "10", and "11" bytes followed by a "chunks to symbols " block to map to QPSK symbols (-1-1j, -1+1j, 1+1j, 1-1j)
  - Q. which chunk is mapped to which symbol, demonstrate mapping to your TA using Time Sink?
  - Use Root Raised Cosine(RRC) Filter for pulse shaping with parameters as
    - sps = 8
    - Gain = 1
    - symbol_rate = 10k
    - alpha being varied in range of 0 to 1
  - Observe the constellation of signal and demonstrate it to TA
- The obtained signal is basically your baseband QPSK-modulated signal

Lab task 3 cont'd : Upconversion to a carrier frequency

- Upconvert the baseband signal to passband, by multiplying it with a 100 kHz carrier signal (ensure that the sampling frequency is upconverted as required)
- Observe the obtained signal (which represents a practical QPSK-modulated signal that one can be actually transmit from the antenna).
- Add Gaussian noise to the digital modulated QPSK signal before demodulation

  Note - Vary the noise amplitude in the range of 0 to 1

Lab task 3 cont'd : QPSK Modulation scheme (Receiver)

- Demodulate the received data using matched filtering at the receiver end
  - To implement the matched filtering at the QPSK receiver use **RRC** filter after down-converting to the appropriate sampling rate
  - Set the variables of Root-Raised Cosine filter(matched filter) same as that of RRC filter used at transmitter end
- To extract the symbols from the waveform
  - Use the "skip head" block to remove initial garbage values added by matched filtering followed by "Decimating FIR filter" (to both real and imaginary components of complex demodulated signal individually)
  - Add the "Threshold" block to convert the data(Real & Imaginery component both) to 0, 1 levels and then map to chunks by designing a 2 x 4 decoder
    Note - Please ensure that input data to boolean operator blocks used for decoding are in byte type
- Then extract the ASCII value from the obtained QPSK chunks by "unpacking 2 bits" and packing back to 8 bits and store the result using the "File Sink" block to a text file

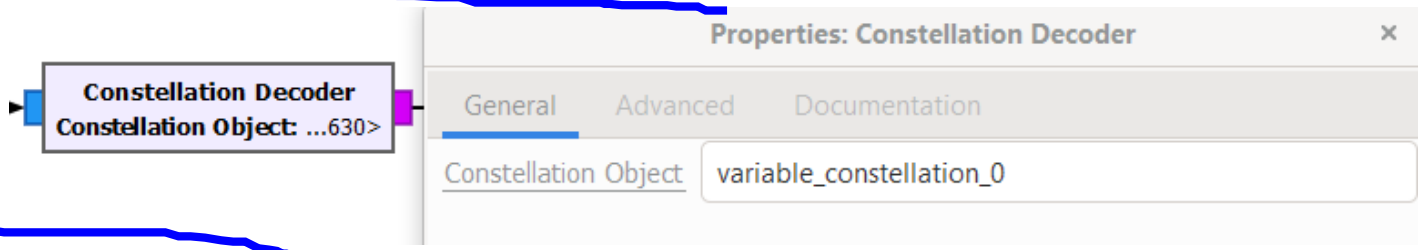
Lab task 4: Constellation Decoder (Bonus)

This part will only be evaluated if task 3 is done correctly

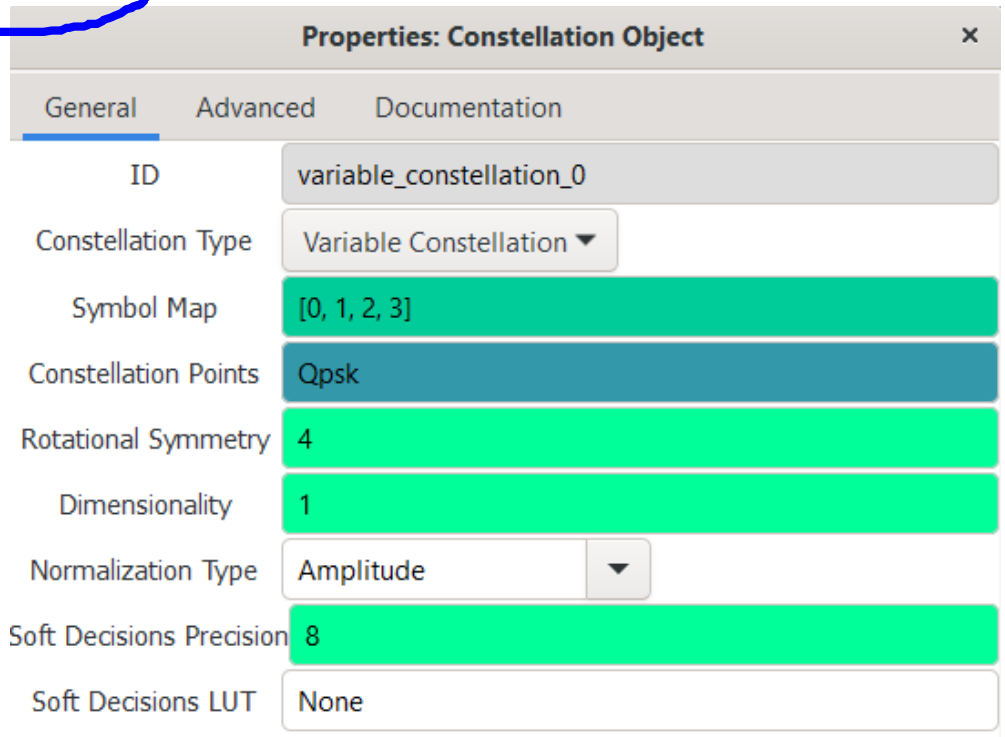
- Demodulate the received data using matched filtering at the receiver end as in task-3
- An Alternative way to decode the symbols is by using a constellation decoder
- To extract the symbols from the waveform
  - Use the "skip head" block to remove initial garbage values added by matched filtering followed by "Decimating FIR filter" (to both real and imaginary components of complex demodulated signal individually)
  - Add the "Threshold" block to get the binary data and convert it to -1, 1 levels and then use the "**constellation decoder**" block for symbol decoding
- Extract ASCII value from the output data of constellation decoder block by "unpacking 2 bits" and pack back to 8 bits and store the result using the "File Sink" block to a text file

Lab task 4(cont'd): Constellation Decoder (Bonus)

- This block is used for symbol decoding



- Constellation decoder block requires a constellation object



- Ensure that constellation points are same as symbol table value given at input (at transmitter)