# Processing of digital samples

Cristian Iñiguez Rodriguez, 1566514@uab.cat

***Abstract*—The main idea is to implement the matched filter of a digital communications receiver and process a received signal to retrieve the data symbols using a square-root-raised-cosinus pulse and the overlap-add algorithm to convolutionate.**

## INTRODUCTION

The goal is to properly process the received signal r(t) with the aim of efficiently retrieving the linearly modulated data symbols that convey the information content of this signal.

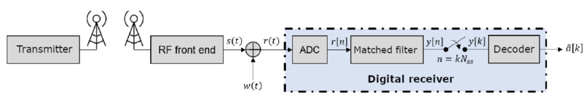


Fig. 1. Basic architecture of a digital communications receiver

Processing the received signal through the matched filter involves the convolution between the received signal and the matched filter impulse response. However, the problem relies in the fact that the transmitted signal is emitted continuously, therefore, received signal is, in principle, of infinite length and the filter response is finite length. Thus it could not be implemented with convecional convolution, here is where overlap-add gives us a solution to such a problem. The main idea of overlap-add is to split the signal in blocks and work with them in frequency domain. Once we have the result in the frequency domain, it is necessary to return to the time domain and then we have to sum the segments to build the convolution output. It almost done, to retrieve the data symbols that convey the information content, is necesary to remove the Doppler frequency component and then decimate the signal to get it, then the constellation plot should be represented.

## I. ANSWER TO QUESTION 1

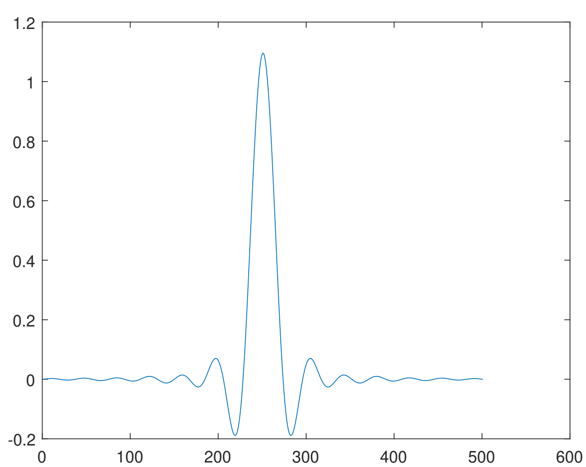


Fig. 2. SQRRC Nss = 25, Ls = 20, roff = 0.35, S = 1

In signal processing, a square-root-raised-cosine, for us SQRRC, is a filter in digital communication to perform the matched filter. SQRRC is the most popular filter that helps to have minimum intersymbol interference, it can be done because the overall response of transmit filter, channel response and receive filter satisfy the Nyquist ISI criterion.

```
waveform_signal = pam_sqrrc(25, 20, 0.35, 1);
plot(waveform_signal);
```

## II. ANSWER TO QUESTION 2

### A. Definition of "matched filter" under AWGN

Under Additive white Gaussian noise, for short AWGN, conditions a matched filter is, for a linear modulation, nothing but a filter whose impulse response is the conjugated and time-reversed version of the shaping pulse being used by the transmitter.

$$p(\tau) = p^*(-\tau)$$

### B. Corresponding filter

As we know the pulse employed by the transmitter is the aforementioned SQRRC pulse, therefore, the corresponding filter would agree with the conjugated and inverted SQRRC over time, but since our SQRRC is real and symmetric, we could use it the way it is.

## III. ANSWER TO QUESTION 3

We compute the energy of the filer as $E = \sum_{n=-\infty}^{\infty} |x[n]|^2$, for our filter we get $E = 24.9996$

```
newWave = fliplr(conj(waveform_signal));
sum(newWave.^2)
```

## IV. ANSWER TO QUESTION 4

The approach behind the Overlap-Add algorithm is to avoid high computational cost and memory resources for long signals, by splitting the signal into shorter segments that are processed individually and then combined, in such a way that the result is equal to what we expect with conventional convolution. In order to efficiently implement the convolution, Overlap-add works by taking into account that

$$F\{r[n] * h[n]\} = R(e^{j\omega}) \cdot H(e^{j\omega})$$

which means that the convolution can be implemented in the frequency domain by means of the product of discrete-time Fourier transforms. In the case under study, such DTFTs are given by $R_m(e^{j\omega}) = F(r_m[n])$ and $H(e^{j\omega}) = F(h[n])$, where they would be the segments of the signals to be added on the output signal.

$$H(e^{j\omega}) = F(h[n])$$

For each $r_m[n]$, do the following

$$R_m(k) = DFT(r_m[n])$$

$$Y_m(k) = R_m(k)H(k)$$

$$y_m(k) = IDFT_N^{-1}\{R_m(k)H(k)\}$$

then to add we have

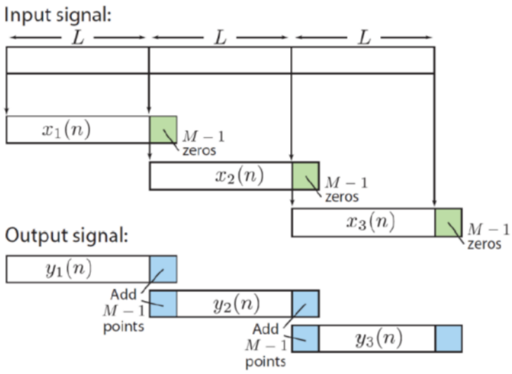


Fig. 3. Overlap-Add method for sequential convolution

The following code is my implementation to overlap and add algorithm.

```
function FilteredSignal = ...
    overlap_add(RxSignal, h)
% length h & input signal
Lh = numel(h);
Lx = numel(RxSignal);
% Optimal N (Efficiency considerations)
N = 2^ceil(log2(2*Lh-1));
% number of blocks
B = ceil(Lx./Lh);
% DTFT (fast fourier transform algorithm)
H = fft(h, N);
% split vector to matrix
rx       = zeros(Lh,B);
rx(1:Lx) = RxSignal;
% aux output
y = zeros(B, N);
for b = 1:B
    % iDTFT ( DTFT ( bloq, N ) x H )
    y(b, :) = ifft(fft(rx(:, b), N) .* H, N);
end
% add row and colum 0
y(B+1, N+1) = 0;
% as the way we save is matrix form we need ...
    two auxiliar vectors to do the summation
y1 = y(:,1:Lh).';
y2 = [zeros(1,Lh); y(1:B,Lh+1:2*Lh)].';
% matrix to vector
y1 = y1(:);
y2 = y2(:);
% sumation vectors
o =  y1 + y2;
%return of the fuction with the expected size
FilteredSignal = o(1:Lh+Lx-1);
end
```

## V. Answer to Question 5

Comparing overlap-add and matlab convolution is as easy as do a substract between the result of both fuctions. Theorical it should be zero, in practical is not zero by some little error.

```
S1 = rand(1024,1);
S2 = rand(4096,1);
c  = conv(S2, S1);
c2 = overlap_add(S2, S1);
abs(c-c2);
```

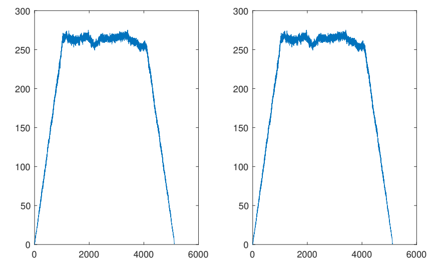

Fig. 4. Convolution and overlap-add

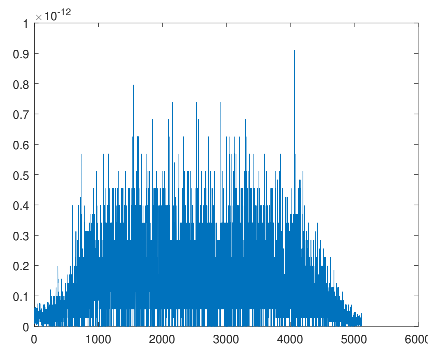


Fig. 5. absolute difference between conv and overlap-add

## VI. Answer to Question 6

Symbol lengh, $N_{ss}$ for short, determine the acquisition bandwidth to successfully perform demodulation. To calculate, we need to know the $T_s$, with it being the sampling period, and the $F_s$, with it being the sampling frequency, when we have that values, calculate the $N_s$ is as easy as $N_{ss} = \frac{F_s}{F}$, where $F = \frac{1}{T_s}$, with values 20 MHz and 1 $\mu s$ we get $N_{ss} = 20$.

```
Nss = @(x,y) x/(1/y);
nss = Nss(20e6, 1e-6);
```

## VII. Answer to Question 7

```
h = fliplr(conj(pam_sqrrc(nss, Ls, roff, 1)));
```

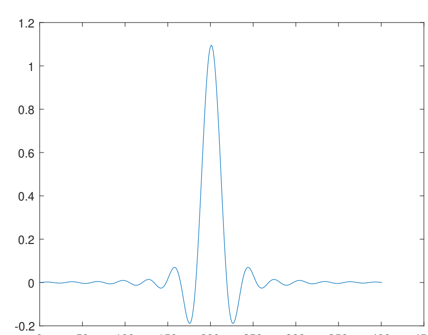

Fig. 6. SQRRC Nss = 20, Ls = 20, roff = 0.35, S = 1

## VIII. Answer to Question 8

```
c  = overlap_add(RxSignal, h);
```

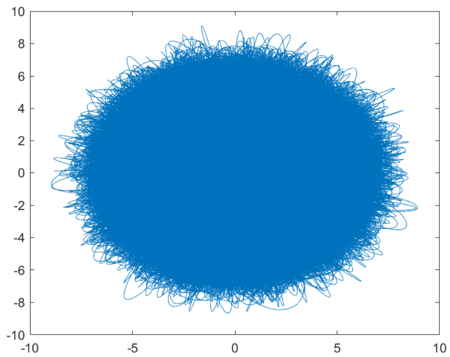


Fig. 7. Overlap-add convolution in complex representation

## IX. Answer to Question 9

Given a received signal $y[n] = \sum_{n=-\infty}^{\infty} a[k]p[n - kN_{ss}]e^{j\omega_d n}$, we can easily remove the Doppler frequency component as long as we know $\omega_d$. The frequency component is removed by multiplying each component of the received signal by $e^{-j\omega_d n}$.

$$y[n] = \sum_{n=-\infty}^{\infty} a[k]p[n - kN_{ss}]e^{j\omega_d n}e^{-j\omega_d n} =$$

$$\sum_{n=-\infty}^{\infty} a[k]p[n - kN_{ss}]$$

If we don't know explicitly $\omega_d$, we can compute it with $F_d$ and $F_s$ and it would be $\omega_d = \frac{2\pi F_d}{F_s}$. Where $F_d$ is the Doppler frequency error and $F_s$ the sampling frequency. In matlab with the following code we would be removing the Doppler frequency component.

```
wd = 2* pi * Fd * (1/Fs);
c  = c .* exp(-1i * wd * (1:numel(c))).';
```

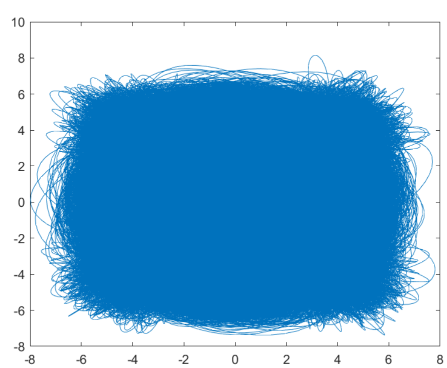


Fig. 8. Removed Doppler component

## X. Answer to Question 10

To proper start the decimation process in order to optimally collect the data-modulation symbols we need to start when the convolution is at its highest, in our case $n$ should be 400. We would have to take each $N_{ss}$ samples to get an appropriate constellation diagram.

## XI. Answer to Question 11

```
Decimated_Signal = c(400:n:end);
```

## XII. Answer to Question 12

The constellation diagram displays the signal as a two-dimensional xy-plane scatter diagram in the complex plane at symbol sampling instants.

In a digital modulation system, information is transmitted as a series of samples, each occupying a uniform time slot were the carrier wave has constant amplitude and phase, restricted to finite number of values . So each sample encodes one of a finite number of "symbols". Each symbol is encoded as a different combination of amplitude and phase of the carrier, so each symbol is represented by a point on the constellation diagram. The constellation diagram shows all the possible symbols that can be transmitted by the system as a collection of points.

For display the constellation diagram of a digital signal by sampling the signal and plotting each received symbol as a point. If the result is a 'ball' or 'cloud' of points surrounding each symbol position, it could mean that the signal is passing through a noisy communication channel and then due to electronic noise or distortion added to the signal, the amplitude and phase received by the demodulator differ from the correct value for the symbol.

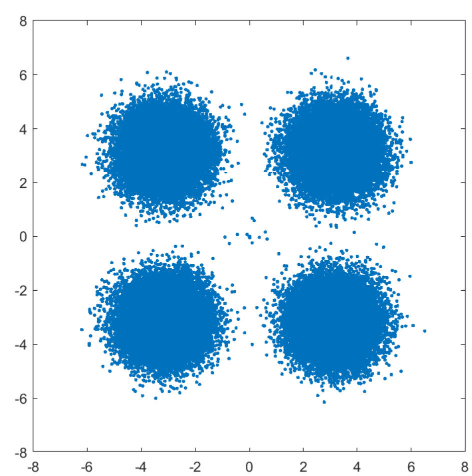


Fig. 9. 4-PAM constellation diagram from r(n)

When it comes to our constellation diagram, as we know, we were working with a PAM pulse, we can say it's a 4-PAM with a small shift showing a characteristic cloud indicating that the canal is noisy.

# Conclusion

In conclusion, we have seen a way to efficiently implement in matlab convolution for continuous signals or long signals that cannot be stored with the overlap-add method. Also we were able to implement a SQRRC filter, known for being a good matched filter, and use it to retrieve the data symbols for a given signal r(n). Once we got the filtered signal, the convolution between the SQRRC pulse and the r(n), we have seen an easy way to remove the Doppler frequency component. And finally we were able to do the constellation diagram of the signal.