5G Signal Processing (EE45GS)

Report on OFDM vs GFDM

Dan Serbanoiu S.N. 200241287

Abstract

The given executable implements a simulation of OFDM and GFDM transmission/reception and was coded and tested using the MATLAB R2020b environment. The program allows the user to change the outcome of the simulation by editing the parameters of "ofdm" and "gfdm" using the structures (.p1) and (.p2), preemptively initialized with a default set of values. The simulation encodes random symbols to OFDM and GFDM signals, sends them through a AWGN channel affected by different SNR values and decodes the received signals, producing the corresponding figures of spectrum, eye-diagrams, constellation diagrams, BER, SER, Q-Factor. The script then saves the figures in a local folder called "figures".

Algorithm

```
close all;
clear;
if not(isfolder("figures"))
    mkdir("figures")
end
%% (.p1) parameter sets for GFDM
gfdm = struct;
gfdm.K = 512; % Number of samples per sub-symbol
gfdm.M = 15; % Number of sub-symbols
gfdm.gfdmM = gfdm.M; % Accounts for OFDM simulation
gfdm.blockLength = gfdm.M*gfdm.K; % Block Length
gfdm.CP = 0.1; % Percentage of cyclic prefix
gfdm.pulse = 'rc'; % Pulse shaping filter ("rc" is raised cosine)
gfdm.a = 0.1; % Roll off factor of the pulse shaping filter
gfdm.mu = 4; % Modulation order of the QAM symbol
gfdm.subcarriers = 1:201; % Allocated subcarriers
gfdm.subsymbols = 2:15; % Allocated sub-symbols
gfdm.blocks = 10; % Number of blocks
%% (.p2) parameter sets for OFDM
ofdm = struct;
ofdm.K = 512;
ofdm.M = 1;
ofdm.gfdmM = gfdm.M;
ofdm.blockLength = ofdm.K;
ofdm.CP = 0.1;
ofdm.pulse = 'rc_td'; % ("rc_td" is a rectangular filter in frequency domain)
ofdm.a = 0.1;
ofdm.mu = 4;
ofdm.subcarriers = 1:201;
ofdm.subsymbols = 1;
ofdm.blocks = gfdm.blocks;
%% (.s1) Simulate GFDM and OFDM transmission
tic
[GFDMS, sym_gfdm_tx, cpx_gfdm_sym_tx, g1] = tx_simul(gfdm);
execution_tx_GFDM = toc;
```

```
tic
[OFDMS, sym_ofdm_tx, cpx_ofdm_sym_tx, g2] = tx_simul(ofdm);
execution_tx_OFDM = toc;
%% (.f1) Plot the PSD of the GFDM/OFDM signals
f = figure("Name", "PSD");
length = min(length(OFDMS), length(GFDMS));
t = linspace(-gfdm.K/2, gfdm.K/2, 2*length+1);
t = t(1:end-1)';
plot(t, mag2db(fftshift(abs(fft(OFDMS(1:length), 2*length))))/2, 'b');
hold on;
plot(t, mag2db(fftshift(abs(fft(GFDMS(1:length), 2*length))))/2, 'r');
ylim([-40, 30]);
xlabel('f/F'); ylabel('PSD [dB]');
legend({'OFDM', 'GFDM'});
print(['figures' filesep 'PSD_gfdm_ofdm'], "-dpng");
set(f, 'visible', 'off');
%% (.s2) Simulate reception with different SNR values
snr = 1:21;
gfdm_ber = []; gfdm_ser = []; ofdm_ber = []; ofdm_ser = [];
execution_rx_GFDM = []; execution_rx_OFDM = [];
[theory_ber, theory_ser] = berawgn(snr, "qam", gfdm.mu^2);
for sii=snr
    tic
    [sym_rx1, cpx_sym_rx1, ber1, ser1] = rx_simul(gfdm, awgn(GFDMS, sii), sym_gfdm_tx, g1);
    execution_rx_GFDM = [execution_rx_GFDM; toc];
    tic
    [sym_rx2, cpx_sym_rx2, ber2, ser2] = rx_simul(ofdm, awgn(OFDMS, sii), sym_ofdm_tx, g2);
    execution_rx_OFDM = [execution_rx_OFDM; toc];
    gfdm_ber = [gfdm_ber; ber1]; gfdm_ser = [gfdm_ser; ser1];
    ofdm_ber = [ofdm_ber; ber2]; ofdm_ser = [ofdm_ser; ser2];
    %% (.f2) Plot the Eye Diagrams/Scatter Plots for different SNR value
    if mod(sii, 3) == 0
        print_eye_scatter("gfdm", gfdm, cpx_sym_rx1, sii);
        print_eye_scatter("ofdm", ofdm, cpx_sym_rx2, sii);
    end
end
%% (.f3) Plot the BER/SER values of the GFDM/OFDM signals
print_ber_ser_q('SER', snr, theory_ser, gfdm_ser, ofdm_ser);
print_ber_ser_q('BER', snr, theory_ber, gfdm_ber, ofdm_ber);
print_ber_ser_q('QFactor', snr, [], 0.5*erfcinv(2*gfdm_ber), 0.5*erfcinv(2*ofdm_ber));
% (.es) Function that can print Eye Diagram, Scatter plot of different formats
function print_eye_scatter(name, opt, sym, sii)
    f1 = eyediagram(sym(1:800), ceil(sqrt(opt.mu)));
    print(strcat('figures', filesep, name, '_eye_snr_', string(sii)), "-dpng");
    f2 = scatterplot(sym, 1, 0, 'r.');
    hold on;
```

```
    scatterplot(qammod(0:opt.mu^2-1, opt.mu^2, 'gray'), 1, 0, 'k*', f2);
    title(strcat('Scatter plot,', {' '}, name, {', '}, 'SNR=', string(sii), 'dB'));
    print(strcat('figures', filesep, name, '_scatter_snr_', string(sii)), "-dpng");
    set(f1, 'visible', 'off');
    set(f2, 'visible', 'off');
end
% (.bsq) Function that can print BER, SER, Q-VALUE of different formats
function print_ber_ser_q(name, snr, theory, gfdm, ofdm)
    f = figure('Name', name);
    if (size(theory) > 0)
        semilogy(snr, theory,'-x');
        hold on;
    end
    semilogy(snr, gfdm, 'o');
    hold on;
    semilogy(snr, ofdm, '*');
    if (size(theory) > 0)
        legend('theory', 'gfdm', 'ofdm');
    else
        legend('gfdm', 'ofdm');
    end
    xlabel('SNR, dB');
    ylabel(name);
    xlim([1, 20]);
    print(['figures' filesep name], "-dpng");
    set(f, 'visible', 'off');
end
% (.t) Simulation of transmitter GFDM/OFDM
function [signal, sym_tx, cpx_sym_tx, g] = tx_simul(opt)
    signal = []; sym_tx = []; cpx_sym_tx = [];
    % (.t1) Generate the transmitter pulse g
    if strcmp(opt.pulse, 'rc')
        t = linspace(-opt.M/2, opt.M/2, opt.M*opt.K + 1);
        t = t(1:end-1);
        t = t';
        g = (sinc(t) .* cos(pi*opt.a*t) ./ (1-4*opt.a*opt.a*t.*t));
        g = fftshift(g);
        g(opt.K+1:opt.K:end) = 0;
        g = g / sqrt(sum(g.*g));
    elseif strcmp(opt.pulse, 'rc_td')
        g = zeros(opt.M*opt.K, 1);
        g(1:opt.K) = 1;
        g = g / sqrt(sum(g.*g));
    end
    for b = 1:opt.blocks
        gfdmM = 1;
```

```
        % (.t2) OFDM case
        if opt.M == 1
            gfdmM = opt.gfdmM;
        end
        for m = 1 : gfdmM
            % (.t3) Generate random data and map to QAM constellation
            sym = randi(2 ^ opt.mu, length(opt.subsymbols) * length(opt.subcarriers), 1) - 1;
            cpx_sym = qammod(sym, 2^opt.mu, 'gray')/ sqrt(2/3 * (2^opt.mu - 1));
            sym_tx = [sym_tx; sym];
            cpx_sym_tx = [cpx_sym_tx; cpx_sym];
            % (.t4) Map data to correct dimensions or pad it with zeros
            if length(opt.subcarriers)== opt.K && length(opt.subsymbols) == opt.M
                D = reshape(cpx_sym, opt.K, opt.M);
            else
                Dm = reshape(cpx_sym, length(opt.subcarriers), length(opt.subsymbols));
                res1 = zeros(opt.K, length(opt.subsymbols));
                res1(opt.subcarriers, :) = Dm;
                res = zeros(opt.K, opt.M);
                res(:, opt.subsymbols) = res1;
                D = res;
            end
            % (.t5) Modulate in time
            DD = repmat(opt.K*ifft(D), opt.M, 1);
            x = zeros(opt.K*opt.M, 1);
            for m=1:opt.M
                symbol = DD(:, m) .* g;
                symbol = circshift(symbol, opt.K*(m-1));
                x = x + symbol;
            end
            % (.t6) Add CP
            cp = ceil(opt.CP*opt.blockLength);
            xcp = [x(end-cp + (1:cp), :); x];
            signal = [signal; xcp];
        end
    end
end
% (.r) Simulation of receiver GFDM/OFDM
function [sym_rx, cpx_sym_rx, ber, ser] = rx_simul(opt, signal, sym_tx, g)
    sym_rx = []; cpx_sym_rx = []; ber = 0; ser = 0;
    for b = 1:opt.blocks
        cp = ceil(opt.CP*opt.blockLength);
        gfdmM = 1;
        dim1 = opt.blockLength + cp;
        dim2 = dim1;
        % (.r1) OFDM case
        if opt.M == 1
```

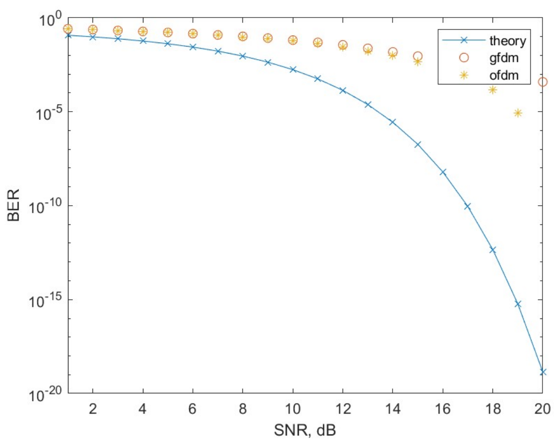
```
            gfdmM = opt.gfdmM;
            dim1 = opt.gfdmM*opt.K + opt.gfdmM*cp;
            dim2 = opt.K + cp;
        end
        for m = 1 : gfdmM
            block = signal((b-1)*dim1 + (m-1)*dim2 + (1:dim2));
            % (.r2) Remove CP
            block = block(cp + 1:end);
            % (.r3) Calculate the transmitter pulse g (Matched Filter)
            g = g(round(1:opt.K/opt.K:end));
            G = conj(fft(g));
            % (.r4) Demodulate in frequency
            L = length(G) / opt.M;
            Xhat = fft(block);
            Dhat = zeros(opt.K, opt.M);
            for k=1:opt.K
                carrier = circshift(Xhat, ceil(L*opt.M/2) - opt.M*(k-1));
                carrier = fftshift(carrier(1:L*opt.M));
                carrierMatched = carrier .* G;
                dhat = ifft(sum(reshape(carrierMatched, opt.M, L), 2)/L);
                Dhat(k,:) = dhat;
            end
            % (.r5) Unmap
            if length(opt.subcarriers) == opt.K && length(subsymbols) == opt.M
                dhat_mf = reshape(Dhat, numel(Dhat), 1);
            else
                Dm = Dhat(opt.subcarriers, opt.subsymbols);
                dhat_mf = reshape(Dm, numel(Dm), 1);
            end
            % (.r6) Map constellation points to data
            dhat_mf = dhat_mf * sqrt(2/3 * (2^opt.mu - 1));
            shm = qamdemod(dhat_mf, 2^opt.mu, 'gray');
            cpx_sym_rx = [cpx_sym_rx; dhat_mf];
            sym_rx = [sym_rx; shm];
        end
    end
    % (.r7) Calculate BER and SER
    [dim1, dim2] = size(sym_tx);
    sym_b_tx = de2bi(sym_tx, opt.mu); sym_b_rx = de2bi(sym_rx, opt.mu); err = [];
    for ii = 1:size(sym_b_tx)
        err = [err; nnz(xor(sym_b_tx(ii, :), sym_b_rx(ii, :)))];
    end
    ser = double(nnz(sym_tx - sym_rx)/dim1);
    ber = sum(err)/(dim1*4);
end
```

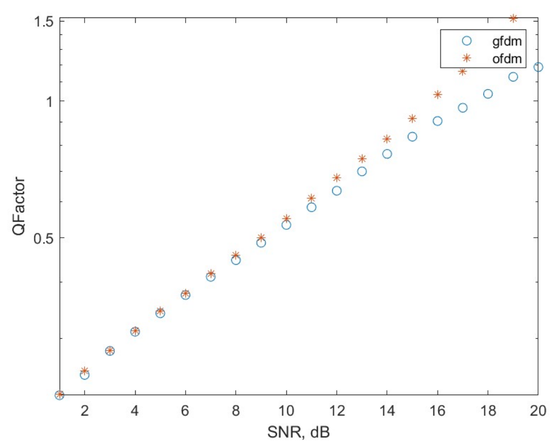
Editable Parameters

Parameter	Description	Realistic Values
K	Samples per sub-symbol	64, 128, 512, 1024
M	Sub-symbols	4, 5, 15 for GFDM, 1 for OFDM
CP	Cyclic Prefix	0.1, 0.2, 0.3, 0.4, 0.5
pulse	Pulse shaping filter	rc = Raised cosine; rc_td = Rectangular filter in frequency domain
a	Roll-off factor of the pulse shaping filter	Between 0 and 1
mu	Modulation of QAM symbol	2, 4, 8, 16
subcarriers	Allocated sub-carriers	1:K
subsymbols	Allocated sub-symbols	1:M for GFDM, 1 for OFDM
blocks	Number of blocks	1, 2, 3, …, ∞

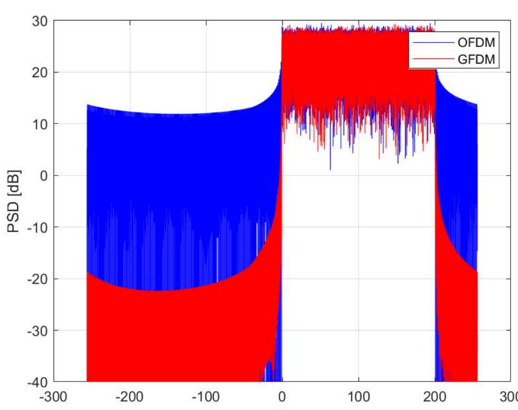
Results of the simulation

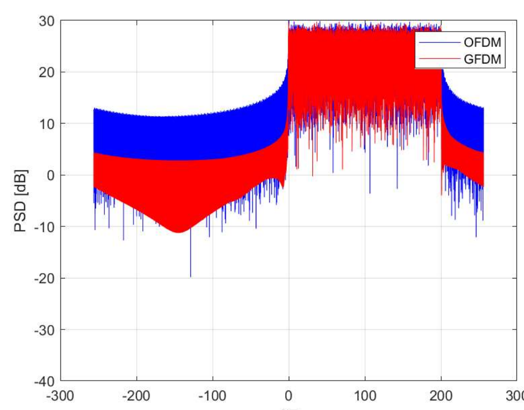
The simulation was executed with the following parameters:
GFDM: K = 512, M = 15, CP = 0.1, pulse = 'rc' (Raised Cosine filter), a = 0.1, mu = 4, blocks = 10, subcarriers = 1:201, subsymbols = 2:15;
OFDM: K = 512, M = 1, CP = 0.1, pulse = 'rc_td' (Rectangular filter in frequency domain), a = 0.1, mu = 4, subcarriers = 1:201, subsymbols = 1, blocks = 10.





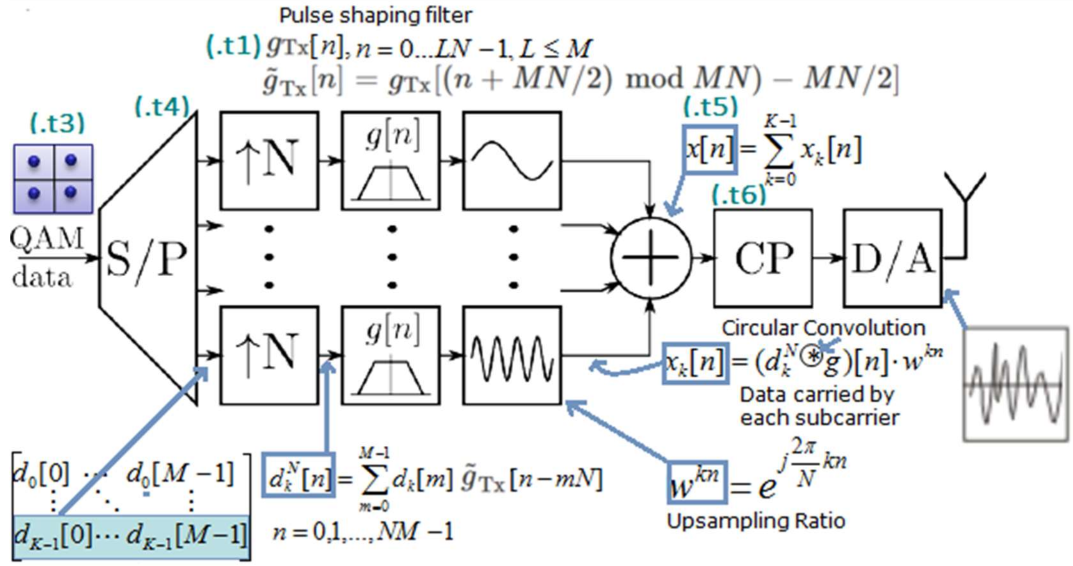
Without CP **With 10% CP**



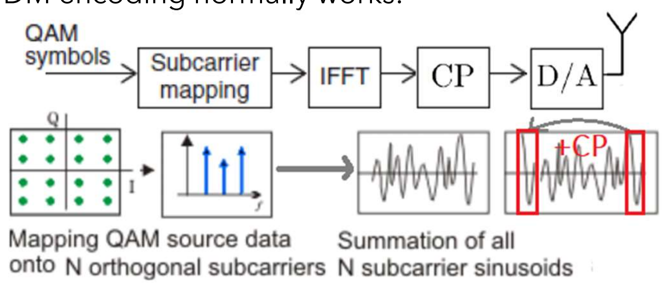


OFDM/GFDM Components

Transmitter



- (.t1): The pulse shaping filter is generated according to the parameters specified in (.p1) and (.p2);
- (.t2): The OFDM encoding is realized by the same function (.t) used for the GFDM encoding. This is because OFDM is a special case of GFDM where M=1, every subcarrier is used and the pulse shaping filter is rectangular in frequency domain. The following figure shows how the OFDM encoding normally works:

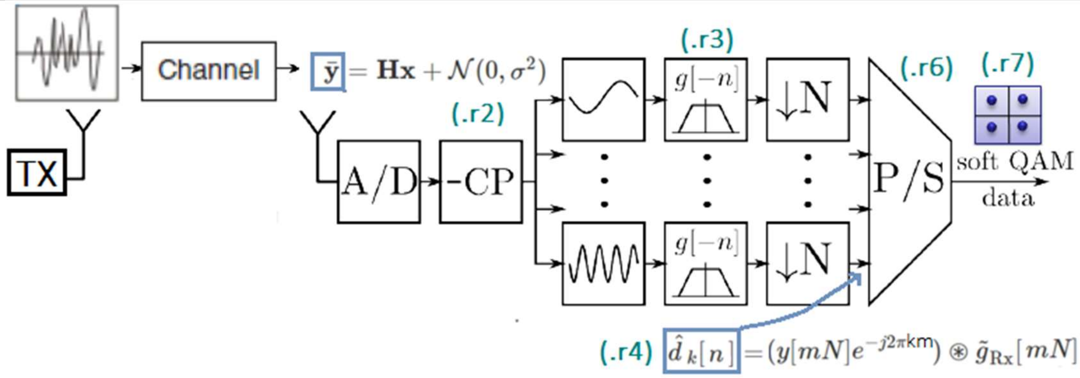


- (.t3): A random set of symbols is generated for every block and mapped to QAM symbols. The number of symbols is the number of chosen sub-carriers times the number of chosen sub-symbols;
- (.t4): Because not all the sub-carriers and sub-symbols are used it is necessary to pad the generated matrix with zeros in order to ensure that the encoding process works properly
- (.t5): A set of sub-carriers K and time slots M are encoded into the x[n] pulse through the following pulse shaping operation:

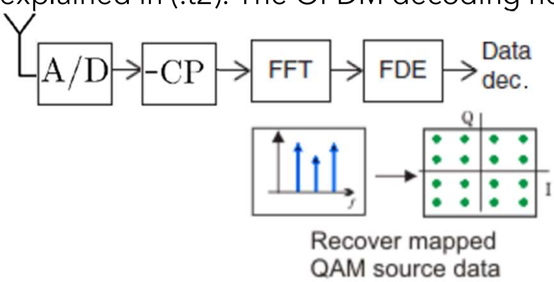
$$x[n] = \sum_{m=0}^{M-1} \sum_{k=0}^{K-1} d_k[m]\, \tilde{g}_{\mathrm{Tx}}[n - mN] e^{j2\pi \frac{kn}{N}},$$

- (.t6): The cyclic prefix is added to the pulse in time domain.

Matched Filter Receiver



- (.r1): This is a list of operations that account for the OFDM decoding, that is done using (.r) for the same reason explained in (.t2). The OFDM decoding normally is done as follows:



- (.r2): The cyclic prefix is removed from the received pulse;
- (.r3): The filter necessary for decoding is calculated from the filter used at the transmitter;
- (.r4): The matched filter operation that recovers the symbols in frequency domain is:

$$\hat{d}_k[n] = \left(y[mN]e^{-j2\pi km}\right) \circledast \tilde{g}_{\mathrm{Rx}}[mN]$$

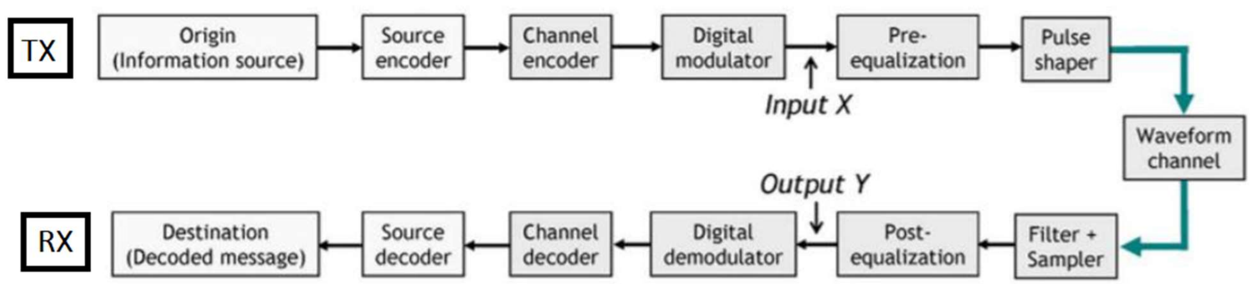
- (.r5): The padding is removed from the resulting matrix and it is converted into serial form;
- (.r6): The guessed constellation points are mapped to symbols according to the appropriate decision boundaries.

OFDM	**GFDM**
Advantages	**Advantages**
○ Subcarriers are orthogonal and their spectral energy does not interfere with the system's ability to recover the original signal. Orthogonality is achieved efficiently using the inverse Fourier transform (IFFT), that maps N symbols to a sum of N sinusoids in time domain that correspond to N orthogonal subcarriers in frequency domain; ○ The receiver can decode the received signal back to frequency domain efficiently using a Fourier transform (FFT); ○ The Fourier transformations ensure the periodicity of the waveform which reduces interference. For example, a pulse shifted in time, corresponds in frequency domain to a frequency proportional phase shift/rotation.	○ Each GFDM block is made of K subcarriers and M time slots. To each carrier and each time slots corresponds one symbol. The spread of the resulting signal in time domain results into a compression in frequency domain achieving better spectral efficiency; ○ Tail biting is employed to remove the need for additional guard intervals and to provide an efficient FFT implementation; ○ The insertion of a cyclic prefix in larger blocks increases further spectral efficiency, while still allowing for efficient channel equalization in the frequency domain; ○ There is flexibility as it is possible to set the block dimensions appropriately and the pulse shaping filter based on the transmission scenario.

Weaknesses	Weaknesses
o The energy of the sidelobes of the components in frequency domain create interference for adjacent subcarriers; o The guard bands or the cyclic prefix can waste a lot of bandwidth; o All the blocks employ the same subcarrier spacing. There is no flexibility for different applications; o Synchronization is problematic.	o It has a worse BER compared to OFDM if a matched filter is used at the receiver; o The transmitter and receiver are much more complex to implement than OFDM and the encoding process is slower.

5G standards, aims and implementation challenges

5G aims to implement network architectures that are programmable, flexible, have lower complexity and more efficient operations and allow for faster development and deployment of services. These networks need to be able to support richer services in many different sectors with different requirements such as gaming, health care, autonomous ride-hailing etc. 5G achieves programmability through VNF (virtualization on network functions), SDN (Software defined networking) and network slicing, allowing operators to manage and orchestrate their networks at different levels (core network, access network and UE). In general, 5G aims to support high data rates, low latency (1ms), mobility, connection density, energy efficiency, spectrum efficiency and area traffic capacity. The main focus of this project is to show some solutions for implementing a communication system at the physical layer while keeping in mind the 5G requirements. A communication system is composed of an efficient transmitter, that encodes messages into appropriate waveforms (such as OFDM, GFDM, FBMC etc.) that can be conveyed through different types of channels affected by different types of distortions such as fading and interference and then decoded correctly and efficiently back to the original data by a matched receiver.



OFDM vs GFDM: Spectral and Energy Efficiency

In 5G energy efficiency is very important because in certain scenarios millimeter waves are used, which are very sensitive to environmental conditions such as obstacles, rain and other waves and can only be used for transmission on relatively short distances. This is why in 5G cells are generally smaller. It is thus very important to ensure that the Power Spectral Density (PSD) of the sidelobes in frequency domain of the encoded signal (OFDM or GFDM) are not causing further interference to adjacent transmitters. The subcarriers or frequencies available for transmission are also limited and need to be reused within adjacent cells. GFDM can pack much more data into larger blocks and is thus better suited for 5G systems. It has both better spectral efficiency and better energy efficiency. OFDM has a better BER than GFDM in general but suffers from very high out-of-band radiation. OFDM also has shorter blocks and wastes much more bandwidth for control information such as the CP. GFDM achieves a better PSD using adaptive filtering and employing one CP for much larger blocks. These conclusions apply both to when the CP is used and when it is not.

OFDM vs GFDM: Complexity
The complexity of OFDM transmitters and receivers is much lower than that of GFDM. From the simulation it can be calculated that an OFDM encoding of 10 blocks of data is double as fast as a GFDM encoding on average. However, it is double as slow to decode the same amount of data. This result shows again that GFDM is better suited for 5G because faster decoding means lower latency.
OFDM vs GFDM: Configurability
GFDM is very configurable. It is possible to use different filters at the transmitter and is possible to use either the same filters or different at receiver according to the requirements of incumbent applications or the type of subcarriers (each subcarrier suffers from different problems). This is very important because one of the requirements of 5G is programmability according to the service request. Several waveforms need to coexist and waveforms need to be considered holistically together with modulation schemes, hardware etc. GFDM allows for scenarios such as introducing sparsity at the signal encoding so that 6 users can be multiplexed together and served with only 4 antennas.
References
http://www.sharetechnote.com/html/5G/5G_Phy_Candidate_GFDM.html https://www.researchgate.net/post/Why-is-IFFT-block-used-in-OFDM-system-at-the-transmission-side https://dsp.stackexchange.com/questions/22473/what-is-difference-between-ofdm-and-gfdm https://ieeexplore.ieee.org/abstract/document/6692619 https://dsp.stackexchange.com/questions/22473/what-is-difference-between-ofdm-and-gfdm