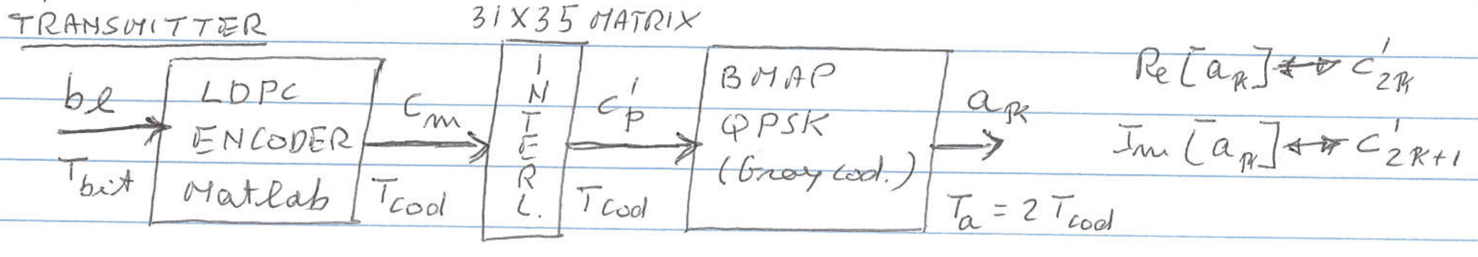
DIGITAL TRANSMISSION — MAY 24, 2016

HW 4 — DUE WED. JUNE 1, 2016 (ROOM Ue @ 10^{30})

Assume that symbols for SC (single carrier) and OFDM systems are generated as follows:

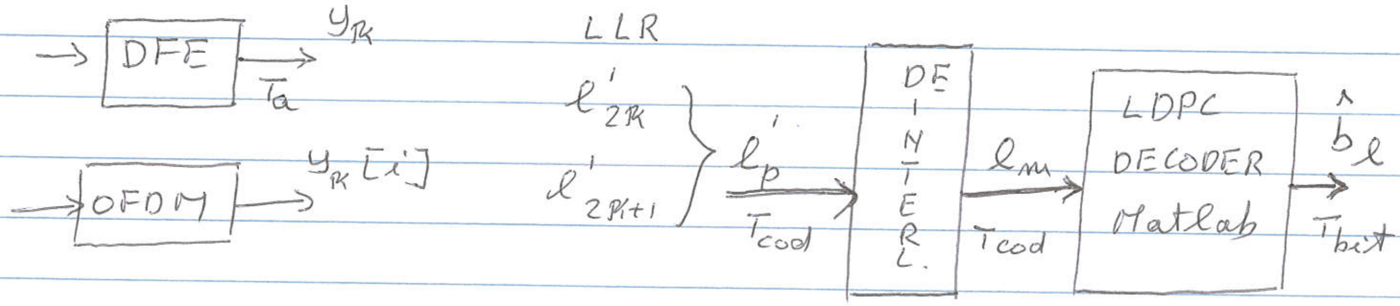
TRANSMITTER



WRITE BY COLUMNS

READ BY ROWS

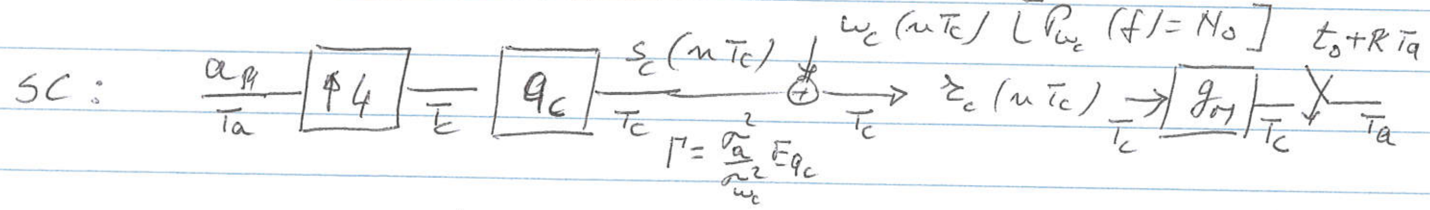
RECEIVER

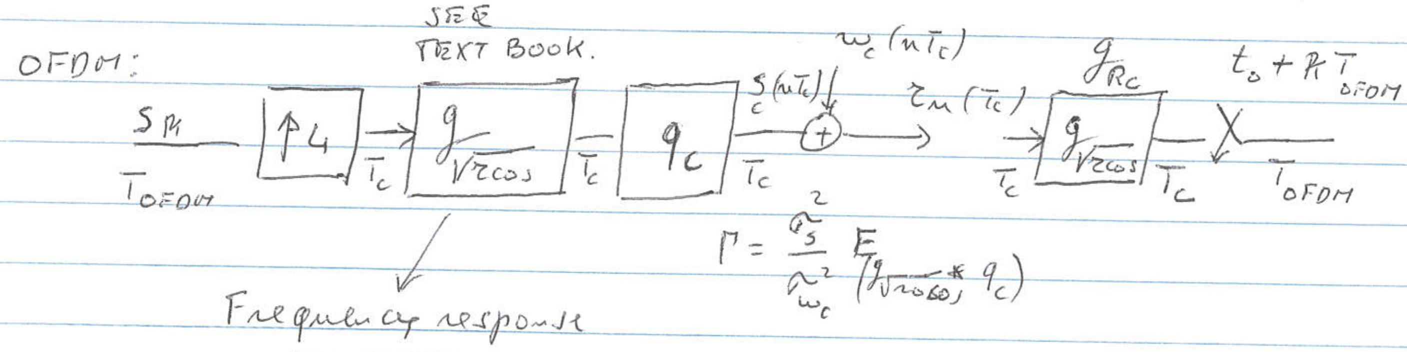


WRITE BY ROWS

READ BY COLUMNS

CHANNEL MODEL (SEE HW3 FOR q_c)





$$\sqrt{rcos\left(\frac{f}{1/T_{OFDM}}, \rho_{OFDM}\right)}, \quad \rho_{OFDM} = 0.0625$$

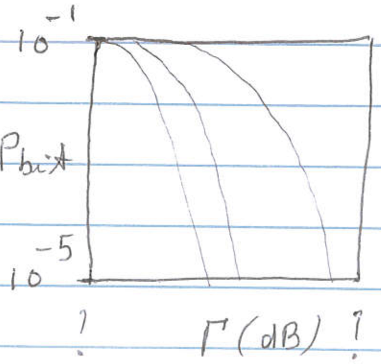
SC system: $\{a_k\}$ are QPSK symbols. Use receiver (b) of HW3 with matched filter and DFE~~@~~T. From y_k, the signal at detection point, after scaling by ψ_D, evaluate $\mathrm{Re}[y_k]$ and $\mathrm{Im}[y_k]$ with corresponding LLR's. Report values of $\bar{t}_0$, M_1, D, M_2 and expressions of LLR's associated to c'_{2k} and c'_{2k+1}. The first four parameter are from HW3.

OFDM system: Choose $\mathcal{M} = 512$ (block size). $T_{OFDM} = \dfrac{T_{block}}{\mathcal{M} + N_{px}}$ where T_{block} is the data 'block period' and N_{px} the prefix length. (In the textbook $T_{block} = T'$)
The receiver filter, operating at T_c, is a $\sqrt{rcos}\left(\frac{f}{1/T_{OFDM}}, \rho_{OFDM}\right)$ with $\rho_{OFDM} = 0.0625$, and its output is sampled at $1/T_{OFDM}$ with a suitable timing phase t_0.
Plot $\{q_c(nT_c)\}$, $\{g_{\sqrt{rcos}}(nT_c)\}$, and $g_c(nT_c) = (g_{\sqrt{rcos}} * q_c * g_{\sqrt{rcos}})$
Plot $\{h(mT_{OFDM}) = g_c(t_0 + mT_{OFDM})\}$, equiv. ch. imp. resp. X OFDM.
Plot $|\mathcal{Q}_c(f)|$ and $|\mathcal{G}_{Rc}(f)|$, in dB, for $f \in [0, 1/2T_c]$.

Determine t_0, i.e. the instant the receiver starts collecting $\mathcal{M} + N_{px}$ samples, in sequence: better, let $t_0 = \bar{t}_0 T_c$, just report $\bar{t}_0$.
Determine suitable values for N_{px} and N_{vir} (the number of virtual subchannels).
Report expression of LLR's.
Plot the DFT of g_c over $\mathcal{M}$ samples: magnitude in dB.

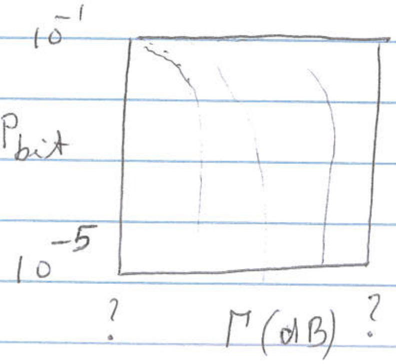
PERFORMANCE: Plot

UNCODED



DFE (blue)
OFDM (red)
AWGN (MF bound) (black)

CODED



DFE (blue)
OFDM (red)
AWGN (black)

(Note: small dynamic)

```
enc = fec.ldpcenc;  % Construct a default LDPC encoder object

% Construct a companion LDPC decoder object
dec = fec.ldpcdec;
dec.DecisionType = 'Hard decision';
dec.OutputFormat = 'Information part';
dec.NumIterations = 50;
% Stop if all parity-checks are satisfied
dec.DoParityChecks = 'Yes';

% Generate and encode a random binary message
msg = randi([0 1],1,enc.NumInfoBits);
codeword = encode(enc,msg);

% Construct a BPSK modulator object
modObj = modem.pskmod('M',2,'InputType','Bit');

% Modulate the signal (map bit 0 to 1 + 0i, bit 1 to -1 + 0i)
modulatedsig = modulate(modObj, codeword);

% Noise parameters
SNRdB = 1;
sigma = sqrt(10^(-SNRdB/10));

% Transmit signal through AWGN channel
receivedsig = awgn(modulatedsig, SNRdB, 0); ...

% Construct a BPSK demodulator object to compute
% log-likelihood ratios
demodObj = modem.pskdemod(modObj,'DecisionType','LLR', ...
     'NoiseVariance',sigma^2);

% Compute log-likelihood ratios (AWGN channel)
llr = demodulate(demodObj, receivedsig);

% Decode received signal
decodedmsg = decode(dec, llr);

% Actual number of iterations executed
disp(['Number of iterations executed = ' ...
     num2str(dec.ActualNumIterations)]);
% Number of parity-checks violated
disp(['Number of parity-checks violated = ' ...
     num2str(sum(dec.FinalParityChecks))]);
% Compare with original message
disp(['Number of bits incorrectly decoded = ' ...
     num2str(nnz(decodedmsg-msg))]);
```