Name:

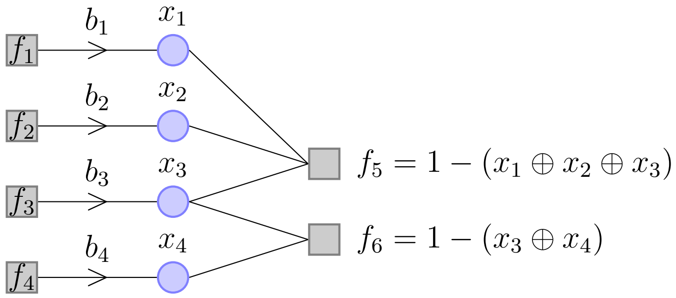
Note:

- You have 2 h 45 min to work at the exam.
- There are six problems that can be solved in any order.
- The exam is closed book (no notes allowed). You are only allowed to use the workstations in the laboratory (not your own laptops). Resources from the internet as well as code written outside this exam are not allowed.
- The code will be evaluated according to the usual criteria, namely correctness, speed, form, and readability. Short comments that allow us to follow what you are doing will improve readability.
- Several problems require writing Matlab code that you will upload on Moodle (as a single archive).
- The Matlab files referenced below are available on Moodle.

Start by downloading from Moodle and unzipping the file with all the data required for the different problems.

Problem 1. 15 p. (LDPC) (This is a “paper and pencil” problem) Consider a binary linear code described by a parity check matrix H with entries in $\{0,1\}$. This means that $\boldsymbol{x} = (x_1, \dots, x_n) \in \{0,1\}^n$ is a codeword iff $H\boldsymbol{x}^T = 0$, where operations are mod 2.

(a) 2 p. Give a parity check matrix H for this code, knowing that the factor graph used by a message passing decoder to compute the a posteriori probability is as in the figure.



(b) 13 p. For a binary code, the messages $V_{n\to m}(x_n)$ send by variable nodes as well as the messages $F_{m\to n}(x_n)$ sent by factor nodes can be represented by a pair of real numbers, one number for each of the two possible values of x_n. In class we have claimed (but not proved) that one can pass the ratio of those two numbers. The purpose of this exercise is to verify this claim by means of an example.

For $i = 1, \dots, 4$, let

$$b_i := \frac{P_{Y_i|X_i}(y_i|0)}{P_{Y_i|X_i}(y_i|1)}$$

be the message sent by factor node i. Your task is to determine the messages needed to decode x_3. (Hence no need to compute all messages.) Each message from a variable node shall be in the form

$$V_{n\to m} = \frac{V_{n\to m}(0)}{V_{n\to m}(1)}, \quad \text{where } V_{n\to m}(x_n) = \prod_{m' \in \mathcal{V}(n)\setminus m} F_{m'\to n}(x_n)$$

and each message from a check node shall be in the form

$$F_{n\to m} = \frac{F_{n\to m}(0)}{F_{n\to m}(1)}, \quad \text{where } F_{m\to n}(x_n) = \sum_{\sim x_n} f_m(\boldsymbol{x}) \prod_{n' \in \mathcal{F}(m)\setminus n} V_{n'\to m}(x'_n).$$

Your answer should be an explicit MAP decoding rule for x_3 that depends only on $b_1, \dots, b_4$.

Solution 1

(a)

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

(b) To simplify the notation, we write V_{ij} instead of $V_{i\to j}$ and similarly for F_{ij}.

$V_{15} = \frac{V_{15}(0)}{V_{15}(1)} = \frac{F_{11}(0)}{F_{11}(1)} = b_1$.

Proceeding similarly,

$V_{25} = b_2$

$V_{46} = b_4$.

Next,

$F_{63} = \frac{V_{46}(0)}{V_{46}(1)} = V_{46} = b_4$

and

$F_{53} = \frac{V_{15}(0)V_{25}(0)+V_{15}(1)V_{25}(1)}{V_{15}(0)V_{25}(1)+V_{15}(1)V_{25}(0)} = \frac{V_{15}V_{25}+1}{V_{15}+V_{25}} = \frac{b_1 b_2+1}{b_1+b_2}$.

We decide that $\hat{x}_3 = 1$ if $F_{33}(0)F_{53}(0)F_{63}(0) < F_{33}(1)F_{53}(1)F_{63}(1)$.

Equivalently, we decide that $\hat{x}_3 = 1$ if $F_{33}F_{53}F_{63} < 1$.

After substituting, we obtain the desired final result:

$\hat{x}_3 = 1$ if $b_3 \frac{b_1 b_2+1}{b_1+b_2} b_4 < 1$.

Problem 2. 8 p. (Miscellaneous Questions) ("paper and pencil")

(a) 2 p. Let $\mathbf{s} = [s_1, \ldots, s_N]$ be a sequence of symbols that we send over a discrete-time AWGN channel of impulse response $h_0, \ldots, h_{L-1}$. So the output is $y_1, y_2, \ldots$ where $y_i = \sum_{l=0}^{L-1} s_{i-l} h_l + z_i$, where the z_i are samples from independent and identically distributed Gaussian random variables. You use $\mathbf{y} = [y_1, \ldots, y_N]$ to estimate the channel as follows. Your channel estimate is the IFFT of $\frac{FFT(\mathbf{y})}{FFT(\mathbf{s})}$, where the division is componentwise. In order for this to work, is there any restriction on the sequence $\mathbf{s}$? Explain.

(b) 2 p. In OFDM, what is the purpose of the cyclic prefix?

(c) 2 p. If we know the position of three satellites and the distances from the receiver to each of them, we can determine the receiver position. Why does the GPS system require four satellites?

(d) 2 p. Consider the parity check matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Which of the following vectors are valid codewords?

$$\boldsymbol{x}_1 = [1 \quad 1 \quad 1 \quad 1 \quad 1]$$
$$\boldsymbol{x}_2 = [1 \quad 0 \quad 1 \quad 1 \quad 0]$$
$$\boldsymbol{x}_3 = [0 \quad 1 \quad 1 \quad 0 \quad 1]$$

Solution 2

(a) We have considered as correct any of the following answers: (i) it has to contain the cyclic prefix; (ii) $FFT(\mathbf{s})$ cannot contain zeros; (iii) the length N must be at least as large as the impulse response.

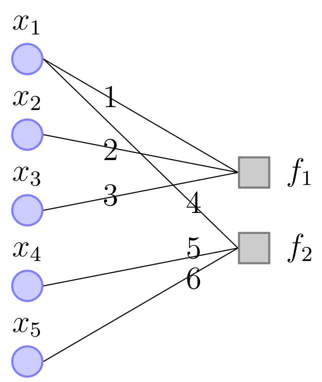
(b) To make the linear convolution of the channel become a cyclic convolution or, equivalently, to make the channel matrix a circularly symmetric.

(c) Because we need to estimate the 3 real numbers that describe the receiver position as well as the clock offset.

(d) $\boldsymbol{x}_2$, because it is the only one that passes the parity check $H\boldsymbol{x}^T = 0$.

Problem 3. 9 p. The figure below shows the factor graph for the binary linear code over $\{0, 1\}$ described by the parity check matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \end{pmatrix}.$$



The edges of the above factor graph have been labeled in the natural order dictated by the check nodes. (In the solution to the LDPC decoder, we implicitly labeled the edges in the natural order dictated by the variable nodes.) Notice that we are allowed to label the socket in any order we desire. While the performance of the code is not affected by the labeling order, the socket matrix as well as M_c and M_v generally do depend on the order in which we label the edges.

(a) 2 p. Write (by hand) the socket matrix for the above graph. (Recall that the kth row of the socket matrix is i, j if the kth edge (socket) has variable node i on the left and check node j on the right.)

(b) 2 p. Write (by hand) the matrix M_v. (Recall that Mv is the $n \times$ (# of sockets) used to determine the messages that go from the variable nodes to the check nodes.)

(c) 5 p. Complete the MATLAB function `p3` that takes H as the input and produces the socket matrix and the sparse matrix M_v. (Use the command `sparse` to produce M_v.)

Solution 3

(a) The socket matrix is

$$\begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ 1 & 2 \\ 4 & 2 \\ 5 & 2 \end{pmatrix}.$$

(b)

$$M_v = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

(c)
```
function [socket,Mv] = p3(H)

% function [SOCKET] = p3(H)
% This function computes the matrix SOCKET
% starting from the parity check matrix H according to the following
% description:
% Input matrix H has dimensions M x N, where M is the number of check
% nodes and N is the number of variable nodes
% SOCKET: matrix of dimensions NEDGES x 2. NEDGES is the total
% number of ones in matrix H. For each edge there is a row in SOCKET:
% the first column is the index of the corresponding variable node,
% and the second one containes the index of the corresponding check node
% MV: sparse matrix of dimensions N x NEDGES, needed to sum all incoming
% edges at variable nodes. It will have value 1 in position (i,j) if
% the j'th socket is connected to variable node i


% Number of variable nodes and check nodes
[~, nvars] = size(H);


% Compute socket (it specifies the edges between variable and check nodes)
[z, k] = find(H');
socket = [z,k];

nedges = length(z); % total number of edges between variable and check nodes

% compute Mv
 Mv = sparse(socket(:,1), 1:nedges, ones(1,nedges), nvars, nedges);
```

Problem 4. 8 p. By running the MATLAB function `p4` you load `innerProducts.mat` which produces a vector `ip` of complex numbers. Process `ip` to obtain a sequence `b` of bits according to the following rule.

Set `b(1)`=0.

For `1< i <= length(ip)`, do the following:

Set `b(i)`=b(i-1) if the phase of `ip(i)` is essentially the same as that of `ip(i-1)`.

Set `b(i)`=1-b(i-1) if the phase of `ip(i)` is essentially the same as that of `ip(i-1)`$+\pi$.

Code efficiency will be taken into consideration.

Solution 4

```
clear all; close all; clc;

% load the inner products
load innerProducts

% determine phase transitions of pi
diff = ip(2:end) .* conj(ip(1:end - 1));
% bit change if diff < 0
diff_b = (real(diff) < 0);

%consider first bit
diff_b=[0, diff_b];

% Integrate to have bits value
b = cumsum(diff_b);
decodedBits = mod(b, 2);
```

Problem 5. 15 p. (A Simple Radar) To determine the position and the (radial) speed of an object, we proceed as follows: at time $t = 0$, we send a pseudorandom (PRN) codesequence, and at time $t = 1$ seconds we send it again. These signals hit the target and come back to the receiver, allowing to estimate the receiver-target distance at two time instants. From the two distances and the corresponding times, we determine the speed, assumed to be constant.

Complete the MATLAB script `p5` as follows. Start by loading the file `signals.m`. This file contains the PRN sequence, the transmitted and the received signals.

(a) 5 p. Determine the travel times (receiver-target-receiver) for the two PRN sequences. If you cannot solve this part, you can use in the following the values of the vector `tof`, expressed in seconds, saved in `signals.mat`.

(b) 2.5 p. Determine the corresponding receiver-target distances.

(c) 2.5 p. The distances you have found correspond to two precise times. What are they.

(d) 2.5 p. Print the target's (radial) speed, with positive sign if the target moves away from the receiver.

(e) 2.5 p. Print the distance at $t = 0$.

Solution 5

```
clear all; close all; clc;

% load the data (code, signalTx, signalRx)
load signals

% sampling period (s)
Ts=1e-6;

% speed of light (m/s)
c=3*1e8;

% time interval between the two pulses (s)
tInt=1; % s

% determine the travel times
c2=abs(xcorr(signalRx,code));
c2=c2(length(signalRx): end);

th=numel(code)/2;
v2=find(c2>th);

% indexing in MATLAB starts from 1!!
tt=Ts*(v2-[1, 1+round(tInt/Ts)])

% determine the distances
distances=c*tt/2;

% determine the times corresponding to these distances
times=[tt(1)/2, tInt+tt(2)/2];

% determine the speed
vEst=(distances(2)-distances(1))/(times(2)-times(1))

%distance at t=0
d0=distances(1)-vEst*times(1)
```

Problem 6. 5 p. ("paper and pencil") We want to determine the position of a parked car using three satellites labeled $1, 2, 3$.

(a) 2 p. Let t_i, $i = 1, 2, 3$, be three arbitrary but fixed times. If $p_r(t, \xi)$ is the car's position at time t in ECEF(ξ), what is the relationship between $p_r(t_1, t_1)$ and $p_r(t_2, t_2)$?

(b) 3 p. At time t_1, the car determines that the time of flight from satellite 1 is τ_1, i.e., a signal from satellite 1 received at time t_1 is emitted at time $t_1 - \tau_1$. Similarly, the car determines that for $i = 2, 3$, a signal emitted by satellite i at time $t_i - \tau_i$ is received at time t_i.

Suppose that for $i = 1, 2, 3$ you know τ_i and the position $p_i(t_i - \tau_i, t_i)$ of satellite i. Which system of 3 equations would you solve to find the position $p_r(t_1, t_1)$ of the car ? (Don't worry if the system has two solutions.)

Solution 6

(a) The parked car is rotating together with the earth. Hence

$$p_r(t_1, t_1) = p_r(t_2, t_2) = p_r(t_3, t_3).$$

(b) The following are three valid equations:

$$\|p_i(t_i - \tau_i, t_i) - p_r(t_i, t_i)\| = c\tau_i, \qquad i = 1, 2, 3.$$

Using Part (a), with $p_r = p_r(t_1, t_1) = p_r(t_2, t_2) = p_r(t_3, t_3)$, we obtain

$$\|p_i(t_i - \tau_i, t_i) - p_r\| = c\tau_i, \qquad i = 1, 2, 3.$$