EECE 580B Modern Coding Theory
WWW: http://dde.binghamton.edu/filler/mct/hw/1

Homework assignment #1
Due date: September 15, 2:50pm

Gaussian elimination in binary arithmetic

Problem description:
You are given a binary matrix $A \in \{0,1\}^{n\times m}$ of size $n \times m$ (n rows and m columns) and a binary column vector of right hand sides (RHS) $b \in \{0,1\}^{n\times 1}$. In order to find the solution of the system of binary linear equations $Ax = b$, with unknown $x \in \{0,1\}^{m\times 1}$, we can use Gaussian elimination done in binary arithmetic (see below). Implement Gaussian elimination algorithm, that for given matrix A and RHS vector b outputs solution x. You are not asked to implement the solver in full details, use the following simplifications:

- Declare error (see below how to declare error in Matlab) if there is no possible solution to the system.
- Declare error if there is more than one solution vector x – (major simplification to the algorithm).

I expect the solution of this assignment in the form of Matlab function:

```
function x = bin_gauss_solve(A, b)
  % implement the body of this function
end
```

Implement the algorithm by yourself; do not use Matlab build-in functions for solving $Ax = b$ in binary arithmetic. Test your code with provided matrices and RHS vectors (see below).

By using the functions `tic()` and `toc()`, measure and report the execution time of your algorithm on all 11 matrices and RHS vectors from file matrices.mat. (Note that some systems of equations with RHS vectors do not have or have more than one solution.)

Binary *arithmetic*: bits ($x, y \in \{0,1\}$) are added as $x + y = XOR(x, y)$, where XOR is binary eXclusive OR. We multiply the bits in the usual way. Study the following example to see the implementation of binary addition in Matlab

```
x = [0 0 1 1]; y = [0 1 0 1];  % set x and y vectors to some values
z = bitxor(x, y)               % use bitxor function on whole vectors
z = mod(x+y, 2)                % calculate x+y mod 2, the result is the same
```

I expect 1 m-file called bin_gauss_solve.m and a list of execution times (or possible error messages if the system has no or more than one solution) for all 11 matrices from matrices.mat.

Hints, warnings and suggestions:

- Make sure that all operations are in binary arithmetic.
- Work on small matrices first. Work out some simple examples of the algorithm on paper and then code the procedure in Matlab.
- Learn how to debug Matlab code.
- Try to avoid for loops as much as possible, use Matlab matrix operations instead.
- Use the following function to declare a user defined error in Matlab.

```
error('Matrix A is rank deficient.');
```

Gaussian elimination algorithm pseudocode: (modified version from Wikipedia).
See http://en.wikipedia.org/wiki/Gaussian_elimination for more information.

```
// A is n by m binary matrix
i := 1 // row and column index

for i := 1 to m do // for every column
  // find non-zero element in column i, starting in row i:
  maxi := i
  for k := i to n do
    if A[k,i] = 1 then maxi := k
  end for
  if A[maxi,i] = 1 then
    swap rows i and maxi in A and b, but do not change the value of i
    Now A[i,i] will contain the old value of A[maxi,i], that is 1
    for u := i+1 to m do
      Add A[u,i] * row i to row u, do this for BOTH, matrix A and RHS vector b
      Now A[u,i] will be 0
    end for
  else
    declare error - more than one solution exist
  end if
end for
if n>m and if you can find zero row in A with nonzero RHS element, then
  declare error - no solution.
end if
// now, matrix A is in upper triangular form and solution can be found
use back substitution to find vector x
```

Test procedure and test data:
I encourage you to test your algorithm on the matrices A1,…,A10 and RHS vectors b1,…,b10 from the following MAT files:

- 'test_matrices_1.mat' – contains 10 matrices and RHS vectors – unique solution exists
- 'test_matrices_2.mat' – contains 10 matrices and RHS vectors – no possible solution
- 'test_matrices_3.mat' – contains 10 matrices and RHS vectors – more than one solution,

which can be obtained from http://dde.binghamton.edu/filler/mct/hw/1 in the following way:

```
clear;                              % clear the workspace
load('test_matrices_1.mat');        % load MAT file with test data

x = bin_gauss_solve(A1, b1);        % use your solver to find x, such that A1*x=b1
sum( b1~=mod(A1*x,2) )              % number of different elements should be 0
...
x = bin_gauss_solve(A10, b10);      % use your solver to find x, such that A10*x=b10
sum( b10~=mod(A10*x,2) )            % number of different elements should be 0
```

Optional questions and extensions: You are encouraged to at least think about the following questions.

1. Can you think of a faster method for solving a system of linear equations? Here, we worked in binary arithmetic, but you do not need to assume that for this question.
2. Can you solve the system of linear equations faster than $\mathcal{O}(n^2)$? If so, how? If not, why?

Purpose of this assignment:
The solver of the system of linear equations is the best possible "decoding" algorithm we can wish for over the binary erasure channel. We will use this decoder for comparison with other decoders we will introduce later.