ELEC97013 Cryptography and Coding Theory Exercise

Wei Dai

October 13, 2023

EEE Department, Imperial College London

Contents

Contents i

I. Teaching Organisation **1**

1. Teaching Organisation **2**
- 1.1. Overview 2
- 1.2. Coursework 2
  - 1.2.1. Schedule 2
  - 1.2.2. Marking (for Coursework 1-3) 3
- 1.3. Software for Coursework 1-3 4
  - 1.3.1. Julia for Programming 4

2. Feedback **5**
- 2.1. You Said We Did 5
- 2.2. Your Feedback 5
- 2.3. Our Feedback to You 5

3. Important Notes **6**
- 3.1. Plagiarism 6
- 3.2. Notes for Coursework Submission 6
  - 3.2.1. Handling Solutions 6

II. Coursework 1 **8**

4. Divisibility **9**

5. Classical Cryptosystems **11**

6. Number Theory **12**

III. Coursework 2 **13**

IV. Coursewor 3 **15**

V. Coursework 4 **17**

7. Learn and Tell Coursework Guideline **19**
- 7.1. Coursework Format 19
  - 7.1.1. Presentation Sessions 19
- 7.2. Marking Scheme 20
  - 7.2.1. Marking Criteria for Presentations 20
- 7.3. Topic Selection 20
  - 7.3.1. Possible Topics 20

Part I.

Teaching Organisation

Teaching Organisation 1.

1.1. Overview

- Lectures
  - Every Monday between 9/10 and 27/11, 16:00-17:50, Room 407A&B.
- Office hours
  - Every Monday between 9/10 and 27/11, 12:00-13:00, Room 503.
- Communications: Should you have any questions or inquiries:
  - Attend the designated office hours.
  - Post them on the Q&A channel.
  - For private matters, please contact us at `weidai.gta.course@gmail.com`.
  - Refrain from using my college email address, as it might result in your email getting overlooked.
- Assessment
  4 coursework: each counts 25%
  - Coursework 1-3 are composed of technical questions.
  - Coursework 4 is a "Learn and tell" presentation.

1.2. Coursework

1.2.1. Schedule

Table 1.1.: Coursework Schedule

	Tasks	Due Date/Time
Coursework 1	Assignment	(Week 3) 16/10
	Due	(Week 5) 30/10 09:59
	Feedback	(Week 6) 6/11
Coursework 2	Assignment	(Week 5) 30/10
	Due	(Week 7) 13/11 09:59
	Feedback	(Week 8) 20/11
Coursework 3	Assignment	(Week 7) 13/11
	Due	(Week 11) 11/12 09:59
	Feedback	Marks in Jan. 2024
Coursework 4	Assignment	(Week 5) 30/10
	Group/Topic decisions	(Week 7) 13/11
	Slides Due	(Week 10) 4/12 08:59
	Presentations	(Week 10) 4-6/12
	Peer marking	(Week 10) 7/12 23:59
	Feedback	(Week 11) 11/12

Note that coursework 4 will be finished before coursework 3 ends.

The procedure for coursework 1-3 is given below.

- Each individual student will receive their own data file from the GTAs.
- Students can form groups in finishing the coursework.
- Before the due time, students need to submit their individual coursework submissions, and specify their group and their contributions to the group. Please note that the relevant files will not be allowed to change after the deadline.
- GTAs will mark individual coursework submission. Let $M_{i,0}$ denote the raw mark received by the student i. Their adjusted mark will be calculated based on the equation (1.1). The adjusted mark will be returned to the student.
- GTAs will choose some coursework questions to discuss. The students who are chosen to lead specific discussions will be given extra marks in order to encourage participation.

The process of coursework 4 is detailed in Chapter 7.

1.2.2. Marking (for Coursework 1-3)

The marking formula for the first three coursework is given by

$$M_i = \min\left(\left(\sum_{k\in\mathscr{G}_l} M_{k,0}\right) * C_i * W_l, M_{i,0}\right), \tag{1.1}$$

where

- M_i is the adjusted mark received by the student i,
- $M_{k,0}$ is the raw mark obtained by the student k,
- $\sum_{k\in\mathscr{G}_l} M_{k,0}$ is the total raw mark obtained by the group $\mathscr{G}_l$,
- C_i is the contribution of the student i in the group,[1]
- W_l is the weighting coefficient for the group $\mathscr{G}_l$, depending on the size of the group,
- and we apply the min function to ensure the adjusted mark does not exceeds the raw mark.

1: Note that

$$\sum_{k\in\mathscr{G}_l} C_k = 1.00 = 100\%.$$

The weighting coefficients for groups are detailed in Table 1.2. There are several considerations behind the design.

- We encourage both individual work and team work.
- The default size of the group is 3 or 4.
- Students from the same group may share the same codes.
- **Cheating offences and plagiarism** are taken very seriously and are dealt with according to the College's Academic Misconduct Policy and Procedure.

Size of the group	1	2	3	4	5	≥ 6
Weighting coefficient W	1.00	0.96	0.92	0.90	0.70	0.50

Table 1.2.: The weighting coefficient W for the group is decided by the size of the group.

Remark 1.2.1 (Extra marks) If

1. a student indicates that they are comfortable to share their

coursework solutions in the feedback/discussion sessions,
2. their coursework solutions are indeed chosen by GTAs and the lecturer for feedback/discussion, and
3. the student participates in explaining the key ideas behind their solutions,

then extra marks will be given the student.
The amount of extra marks given to the student will be decided by the GTAs and the lecturer, based on the quality of the solutions and the explanation of key ideas. The amount of extra marks will be announced to the student.

1.3. Software for Coursework 1-3

The coursework requires Julia programming. The coursework submission files (except coursework 4) include a Jupyter notebook file and a data file in JLD2 format. We recommend VSCode as the default editor.

The software that you need to install/have

- Jupyter Notebook
  Download and install the package management software Anaconda. By default, it will install Python and Jupyter Notebook to your system.
- Julia programming language
  - Download and install the current stable version of Julia.
  - Check whether Julia has been installed properly: Run Julia interactive session (a.k.a. REPL) by following the instructions.
- VSCode
  - Download and install VSCode.
  - We need to install some extensions for VSCode. See here for how to install VSCode extensions.
    - Anaconda should have automatically installed VSCode Extensions `Python` and `Pylance` for you. If not, install them.
    - Install extension `Julia` for Julia programming in VSCode. See here and here for more details.

1.3.1. Julia for Programming

- A collection of tutorials.
- Introduction to Scientific Programming and Machine Learning with Julia.
- A tutorial in pdf format.

Feedback 2.

2.1. You Said We Did

Based on students' feedback in the past several years, we have done the following changes

- Lecture notes
  - We have changed the lecture notes from slides to a book format.
    - Contents are more self-contained.
    - We use `kaobook` LaTeX template so that we have enough space to make notes in the lectures.
  - Less theory and more examples
- Assessment
  - The format has been changed from a final exam to four coursework.
    - Emphasise a lot more on the applications of theory.
    - Help students horn analysis and programming skills.
  - Significantly reduce the weighting of peer marking.
  - Introduce and refine the coursework contents and the marking scheme to encourage both individual efforts and teamworking
    - Each individual student get their own data.
    - Each individual student can specify their own contributions to the group.
    - Individual student's mark depends on the size of the group.

2.2. Your Feedback

Towards the end of the term, please do the **MEQ** (previously known as SOLE) questionaries from the college.

Towards the end of the term, GTAs may contact you for our own questionaries for this module.

2.3. Our Feedback to You

- We target at finishing the coursework marking within 10 working days. We will return the relevant information to you.
- In the lectures, we will discuss some interesting/challenging coursework questions together.

Important Notes | 3.

3.1. Plagiarism

PLAGIARISM/COLLUSION DECLARATION

Coursework submitted for assessment must be the original work of you and your group. Assignments are subjected to regular checks for plagiarism and/or collusion. Plagiarism is the presentation of another person's thoughts or words (those outside your group) as if they were your own. Collusion involves obtaining help from someone outside your group to complete your work. In preparing your coursework, you should not seek help, or copy from any other person or source, including the Internet, without proper and explicit acknowledgement.

There is a procedure in place for you to declare individual contributions within your group for courswork. You must declare the contributions fairly and accurately.

You must not disclose your solutions or insights related to coursework with anyone else, including future students or the Internet.

By acknowledging the the statements above, you are declaring that both this and all subsequent pieces of coursework are, and will remain, the original work of you and your group.

- Submissions will not be accepted without the aforementioned declaration.
- Members of a group are deemed to have collective responsibility for the integrity for work submitted and are liable for any penalty imposed, proportionate to their contributions.

3.2. Notes for Coursework Submission

- Each registered student will get a data file. The data in the data file can be unique.
- Each registered student needs to submit the solutions related to their own data, no matter whether they are in groups or not.

3.2.1. Handling Solutions

In our coursework, we use the following convention.

- If the solution to a question is a unique integer, you need to assign an integer value to your solution variable.
- If the solution to a question does not exist, you need to create a 1-D array of length zero.
- If the solution to a question is not unique, you need to create a 1-D array, of which the length is the number of distinct solutions, and then specify the values in the **ascending** order.

Examples:

```
    x = 3;
    y = Array{Int64,1}(undef,0);
    z = Array{Int64,1}(undef,3);
    z[1] = 3; z[2] = 4; z[3] = 5;
```

Part II.

Coursework 1

Divisibility 4.

Remark 4.0.1 Note that you are not allowed to use any Julia packages except Base and LinearAlgebra.

1. (Congruence equations)
   Solve the following congruence equations in the form of $ax \equiv b \pmod{n}$. Find *all* solutions of x such that $0 < x < n$.
   Use the data in the data file. When there exists no solution,
   a) Find all solutions of x such that $0 < x < n$. [2]
   b) Find all solutions of x such that $0 < x < n$. [2]
   c) Find all solutions of x such that $0 < x < n$. [2]
   d) Find all solutions of x such that $0 < x < n$. [2]
   e) Find all solutions of x such that $0 < x < n$. [2]
   f) Find all solutions of x such that $0 < x < n$. [2]
2. a) The **Fibonacci numbers** $1, 1, 2, 3, 5, 8, \cdots$ are defined using the relationship $F_1 = 1,\ F_2 = 1,\ F_{n+1} = F_n + F_{n-1}$.
   Find the closed form for $\gcd(F_n, F_{n-1})$ for all $n \geq 1$. Prove your result. [3]
   b) Find

$$\gcd\left(\underbrace{11111111}_{F_6 \text{ many}}, \underbrace{11111}_{F_5 \text{ many}}\right).$$

   [1]
   c) Let $a = 111\cdots11$ be formed with F_n many repeated 1's and let $b = 111\cdots11$ be formed with F_{n-1} many repeated 1's.
   Find $\gcd(a, b)$. Prove your answer. [3]
3. (Matrix inverse)
   a) Based on the data in the data file, find all values of $b \pmod{26}$ such that

$$\begin{bmatrix} 1 & a \\ b & 3 \end{bmatrix} \pmod{26}$$

   is invertible. [4]
   b) Find all primes $1 < p \leq 50$ for which

$$\begin{bmatrix} 1 & a \\ b & 3 \end{bmatrix} \pmod{p}$$

   is not invertible. [4]
4. (Matrix inverse) Use the data in the data file to perform the following computations.
   a) Use `LinearAlgebra.det` to compute the determinant of $\boldsymbol{M}$.
   Take the mod m arithmetic. Note the number that you obtained. [1]

b) Use `Base.inv` to compute the inverse matrix of $\boldsymbol{M}$, denoted by $\boldsymbol{M}^{-1}$. [1]

   In the next questions, we will use *cofactors* to compute the determinant and the inverse under modular arithmetic. This webpage details the cofactor expansion of the determinant and the usage of cofactor matrix to compute the inverse of a matrix.

   Also you need to use `big.(M)` to avoid the overflow issue for `Int`.

c) Write a function to use cofactors to compute the determinant. Compute the value $\det(\boldsymbol{M}) \bmod m$. [3]

d) Write a function to use cofactors to compute the inverse matrix $\boldsymbol{M}^{-1} \bmod m$. [4]

   Note that the possible different results between the results in a)-b) and c)-d). This is due to finite precision when using Floating point numbers.

Classical Cryptosystems | 5.

Decrypting a ciphertext usually involves trial and error. Often, this process necessitates writing a specific program or function.

The Julia package `FreqTables.jl` can be helpful.

1. (Shift cipher) The ciphertext in the data file was encrypted by a shift cipher. Decrypt it. [2]
2. (Affine cipher) The ciphertext in the data file was encrypted by an affine cipher. Decrypt it. [4]
3. (Vigenère cipher) The ciphertext in the data file was encrypted by a Vigenère cipher, where the key is of length at most 8.
   a) Write a function to calculate the shifted coincidence frequency for $1 \leq l \leq 8$. [2]
      Use this to estimate the length of the key. [1]
   b) Find the key and hence the plaintext. [6]
4. (Block cipher) We consider a block cipher of the form,

$$\boldsymbol{c} = \boldsymbol{x}\boldsymbol{A} + \boldsymbol{b} \quad \text{mod } 26,$$

   where the matrix $\boldsymbol{A}$ and the vector $\boldsymbol{b}$ are given in the data file.
   a) Write a function to encrypt the plaintext. Find the corresponding ciphertext of the plaintext in the data file. Note that zero-padding is required in this question if the length of plaintext is not a multiple of the block length. [3]
   b) Find the letter frequency of the ciphertext. Observe the difference between the letter frequency of the ciphertext and that of the English language. [2]

Number Theory 6.

1. Suppose $x \equiv a_1 \bmod n_1$, $x \equiv a_2 \bmod n_2$, and $x \equiv a_3 \bmod n_3$. Find the value of $0 < y < n_1 n_2 n_3$ such that $x \equiv y \bmod n_1 n_2 n_3$. [3]
2. Without the help of the computer, divide 2^m by 101. What is the reminder? Briefly show how you get your solution. [2]
3. (Primitive root)[1]
   Note that the prime number p can be very large.

   a) Write a function `is_primitive_root` which can check whether a number a is a primitive root of the prime number p.
      Use your function to check whether the given a is a primitive root of the corresponding given p in the data file. [8]
   b) Write a function `next_primitive_root` of which the input is a number a and a prime number p, and the output is a number x such that $x \geq a$ is the smallest primitive root of p. Use your function to find the next primitive root using the given a and p in the data file. [6]

1: You are allowed to use functions in the Julia package `Primes.jl`

Part III.

Coursework 2

To be announced

Part IV.

Coursewor 3

To be announced

Part V.

Coursework 4

See below for the draft version

Learn and Tell Coursework Guideline 7.

The purpose of this coursework is to encourage students to

- Explore the literature, find interesting or useful stuff to study and present;
- Learn from each other and broaden horizons;
- Practice skills of project management, decision-making, team-playing, leadership, and presentation.

7.1. Coursework Format

Students will work in groups and present a technical topic that they have learned. The topic must be relevant to the module but should go beyond the taught materials. It can be on application or/and theory. Coursework will be marked by GTAs, the instructor, and peer students.

The **milestones** for this coursework are as follows.

1. We will provide detailed information on coursework 4 presentations according to Table 1.1.
2. Students need to form their groups, decide a topic, and choose a time slot for their presentation. The relevant information should be input in the relevant excel file by the due date shown in Table 1.1.
3. Groups submit their slides to the Teams folder by the due time (Table 1.1). Make sure that the key references are listed in your slides. After the due time, the Teams folder will be read-only.
4. Presentation:
    - Each group will have 15 minutes for presentation and 5 minutes for questions.
    - The group is required to attend all the presentations in the session where they present.
    - All the groups in a session are required to arrive 10 minutes before the session starts for the setup.

7.1.1. Presentation Sessions

The presentation sessions will be finalised later. The following list is tentative.

- 4 December, 2023, Room 611
    - 14:00-15:40
    - 16:00-17:40
- 5 December, 2023, Room 909B
    - 9:30-11:10
    - 11:30-13:10
- 6 December, 2023, Room 611
    - 16:00-17:40

7.2. Marking Scheme

$$M_i = (0.4M_{g,p} + 0.6M_{g,a}) * 0.92 + M_{i,p} * 0.08$$

where

- ▶ $M_{g,p}$ is the mark for the group g from peer marking. Each student is supposed to mark certain number of presentations given by other groups. The peer marks will be aggregated.
- ▶ $M_{g,a}$ is the mark for the group g from 'authoritative' marking. 'Authoritative' marks are given by GTAs and the instructor.
- ▶ $M_{i,p}$ is the mark for the individual student i, depending on the quality of the student's peer marks/comments for other groups. Note that every student is required to mark at least 7 other groups.

7.2.1. Marking Criteria for Presentations

- ▶ Technical topic and contents (50%): Timeliness, relevance, depth, and breadth. (Something interesting and useful)
- ▶ Presentation and delivery (35%): Clearness, conciseness, and easiness to follow. Time control. (Audience can learn something)
- ▶ Q&A (15%): Accuracy and conciseness. Time control.

7.3. Topic Selection

- ▶ Every group finds their own topic to present. The topic can be theory, methodology, or applications.
- ▶ The reading materials can be academic papers, book chapters, published articles, etc.
- ▶ Typical questions to be answered when reading a paper include
  - What is the problem under study? Why is the problem relevant/important?
  - What are the approaches/methods to address the problem?
  - What are the pros and cons?

7.3.1. Possible Topics

The following list of topics is only for your references. You can go beyond.

- ▶ Error Correction Codes
  - Convolutional code
  - Fountain codes
  - Golay codes (Binary Golay code)
  - Goppa code and McEliece cryptosystem
  - Low-density parity-check (LDPC) code
  - Network coding
  - Online code
  - Polar code
  - Raptor code

    - Reed-Muller code
    - Tornado code

- Post-quantum cryptography
    - Lattice-based hard problems and cryptography
    - McEliece code based cryptography
    - Shor's algorithm
- Other topics
    - Algebraic integers
    - Arithmetic functions
    - Continued fractions
    - Distribution of primes
    - Elliptic curve crytosystem
    - Factorization
    - Gaussian integers
    - Geometry of numbers
    - Hashing function (birthday attack), hash-based cryptography
    - Homomorphic cryptography
    - Merkle signature scheme
    - Multivariate cryptography
    - Primality test
    - Quadratic reciprocity
    - Zero knowledge games