ELEC70069
Cryptography and Coding Theory Exercise

Wei Dai

October 12, 2024

EEE Department, Imperial College London

Contents

Part I.

Teaching Organisation

1. Teaching Organisation

1.1. Overview

- Lectures
  - Mondays 07/10 - 25/11/2024, 16:00-17:50, Room **407A&B**
  - Mondays 02/12 - 09/12/2024, 16:00-17:50, Room **508A&B**
- Office hours
  - Mondays 07/10 - 02/12/2024, 12:00-12:55, Room 503.
- Communications: Should you have any questions or inquiries:
  - Attend the designated office hours.
  - Post them on the Q&A channel.
  - Refrain from using my college email.
- Assessment
  4 coursework assignments: each counts 25%

1.2. Coursework Assignments

1.2.1. Schedule

Table 1.1.: Coursework Schedule

	Tasks	Due Date/Time
Coursework 1	Assignment	(Week 3) 14/10
	Due	(Week 4) 25/10 11:59
	Feedback	(Week 6) 04/11
Coursework 2	Assignment	(Week 5) 28/10
	Due	(Week 6) 08/11 11:59
	Feedback	(Week 8) 18/11
Coursework 3	Assignment	(Week 7) 11/11
	Due	(Week 8) 22/11 11:59
	Feedback	(Week 10) 02/12
Coursework 4	Assignment	(Week 9) 25/11
	Due	(Week 10) 06/12
	Feedback (Marks only)	(Week 11) 13/12

Coursework Overview:
Individual Data + Team Work + Individual Assessment

The procedure for the coursework assignments is outlined below.

- Each student will receive a unique data file provided by the GTAs.
- Students collaborate in groups while working on the coursework.
  The group information is input in an excel file.

- By the due time, each student must submit their own individual coursework, clearly specifying their contributions to the group effort. Note that submissions cannot be altered after the deadline.
- GTAs will assess individual submissions. Let $M_{i,0}$ denote the raw mark received by the student i. An adjusted mark, calculated using Equation (1.1), will then be provided to the student.
- Some coursework questions will be selected for in-depth discussion. Students chosen to lead these discussions will receive bonus marks to encourage active participation.

1.2.2. Marking

The marking formula for the first three coursework is given by

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

where

- M_i is the adjusted mark received by the student i,
- $M_{k,0}$ is the raw mark obtained by the student k,
- $\sum_{k\in\mathcal{G}_l} M_{k,0}$ is the total raw mark obtained by the group $\mathcal{G}_l$,
- C_i is the contribution of the student i in the group,[1]
- W_l is the weighting coefficient for the group $\mathcal{G}_l$, depending on the size of the group,
- and we apply the min function to ensure the adjusted mark does not exceeds the raw mark. [2]

1: Note that
$$\sum_{k\in\mathcal{G}_l} C_k = 1.00 = 100\%.$$

2: It is typical that $M_i = M_{i,0} \times W_l$, i.e., the adjusted mark is simply the raw mark multiplied by the weighting coefficient.

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 within the group.
- The default size of the group is 2 or 3.
- 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	0.97	0.94	0.86	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 (Bonus marks) GTAs will select students to present their solutions to specific questions in-depth. Students can earn bonus marks by sharing and explaining their coursework solutions during the feedback/discussion sessions.
The bonus marks awarded will be determined by the GTAs and the module leader, based on the quality of the solutions and the clarity of the explanation. The awarded marks will be communicated to the participating student.

1.3. Software for Coursework 1-3

The coursework requires Julia programming. The coursework submission files 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.

Your Feedback Matters 2.

2.1. You Said, We Did

Over the past few years, we've listened carefully to your feedback and implemented substantial changes to enhance your learning experience.

2.1.1. Lecture Notes

- **Transition to Book Format**: We've transformed our lecture notes from slides into a comprehensive book format. This allows for a more cohesive and self-contained learning experience.
- **LaTeX Template**: We've adopted the LaTeX template `kaobook`, which gives you more space for note-taking.
- **Increased Examples**: We've incorporated more examples and illustrations to reinforce key concepts and make the material more engaging.

2.1.2. Assessments

- **Shift to Coursework**: We've transitioned from traditional paper-based exams to coursework assignments. This approach allows us to focus more on the practical application of theoretical knowledge.
- **Skill Development**: Coursework assignments are designed to help you hone your analytical and programming skills.
- **Peer Marking Removed**: We've removed peer marking to ensure that your performance is accurately reflected in your final grade.
- **Individual Recognition**: Our coursework marking process is designed to fairly assess and appreciate your unique contributions and efforts.

2.2. Recent Changes

Based on your feedback from the last academic year, we've implemented the following changes:

- **Reduced Coursework Load**: We've reduced the overall workload of coursework assignments by removing a learn-and-tell presentation coursework and spreading the questions across four assignments instead of three.
- **Coursework submission deadlines**. Coursework deadlines are now on Fridays, providing you with a more relaxed weekend and allowing the department to efficiently handle extension requests.
- **Enhanced Feedback**: We've increased the amount of information provided with your coursework marks. In addition to your individual scores, we'll share statistical information about the class marks.

- **In-Lecture Problem Discussion**: To foster deeper understanding and active engagement, we've incorporated in-lecture problem discussions and solving sessions.

2.3. Your Feedback

Please take the time to complete the **MEQ** questionnaires at the end of the term. Your continued feedback is invaluable in helping us improve the learning experience for this module.

2.4. Our Commitment to You

- **Accessibility and Support**: We're dedicated to providing you with the support you need to succeed. We offer regular **office hours** (Mondays, 12:00-12:55, Room 503) where you can discuss any questions or difficulties you may encounter. Additionally, we maintain a **Q&A channel** on Teams for more immediate assistance.
- **Timely Feedback**: We understand the importance of timely feedback. We aim to complete coursework marking within 10 working days of the submission deadline. You'll receive your marked assignments along with statistical information about the class performance.
- **Engaging Learning Experiences**: We're committed to creating an engaging and supportive learning environment. **In-lecture problem discussions and solving sessions** provide opportunities for you to actively participate, ask questions, and deepen your understanding of the material.

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:

```
# The solution is a unique integer
x = 3;

# The solution is not well-defined (does not exist)
y = Array{Int64,1}(undef,0);

# The solution is not unique
z = Array{Int64,1}(undef,3);
z[1] = 3; z[2] = 4; z[3] = 5;
# Or equivalently
z = [3, 4, 5] ;
```

Part II.

Coursework 1

4. Divisibility

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

1. (Congruence equations)
   Using the data in the data file, solve the following congruence equations in the form of $ax \equiv b \pmod{n}$. Find *all* solutions of x such that $0 < x < n$.
   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 $n \geq 1$. [1]
   Prove your result. [3]
   b) Let $a = 111\cdots11$ be formed with F_n many repeated 1's.
   Let $b = 111\cdots11$ be formed with F_{n-1} many repeated 1's.
   Find $\gcd(a, b)$. [1]
   Prove your answer. [3]
3. (Modular Exponentiation) Using the data in the data file, compute the following exponentiation in modular arithmetic.
   a) Compute $b^p \pmod{n}$. [2]
   b) Compute $b^p \pmod{n}$. [2]
   c) Compute $b^p \pmod{n}$. [2]
   d) Compute $b^p \pmod{n}$. [2]
   e) Compute $b^p \pmod{n}$. [2]
4. (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. [3]
   b) Find all numbers $3 \leq n \leq 50$ for which
   $$\begin{bmatrix} 1 & a \\ b & 3 \end{bmatrix} \pmod{n}$$
   is invertible. [3]
5. (Matrix inverse) In this question, we will use cofactors to compute the inverse of a matrix. You can refer to this webpage for more details on how to use a cofactor matrix to compute the inverse.
   Write a function to compute the inverse of a matrix in modular arithmetic, i.e., $\boldsymbol{A}^{-1} \pmod{n}$, using its cofactor matrix. The entries

in your $\boldsymbol{A}^{-1}$ must be integers between 0 and $n-1$. You may use the functions `LinearAlgebra.det` and `Base.gcdx`.
Using the provided data file and your programmed function, complete the following tasks:

a) Compute $\boldsymbol{A}^{-1} \pmod{n}$. [3]
b) Compute $\boldsymbol{A}^{-1} \pmod{n}$. [3]
c) Compute $\boldsymbol{A}^{-1} \pmod{n}$. [3]

5. Classical Cryptosystems

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,

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

   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]