MATLAB II: Recovering Martian Logs

ENGS102P: Design and Professional Skills 2017/18

Discipline-specific Element 2

MATLAB II: Recovering Martian Logs

For this lab and assignment, we consider that “The Martian” movie and book represent events that actually took place[1]. Once the next NASA mission arrived in Mars, they recovered the computer containing the *codified* log files of the astronaut Mark Watney, as well as recordings of conversations and messages he received from Earth.

Alas, most of the material in the storage units of the recovered computer has been corrupted due to lack of synchronization and cosmic radiation. NASA painstakingly managed to obtain 130 random chunks: `scra_chunk_1` to `scra_chunk_130`. They are available in file `chunks.zip`, found here: https://moodle.ucl.ac.uk/course/view.php?id=43301

You are each assigned a single chunk `scra_chunk_<X>` **and a single key:** `key_<X>.mat` with `X` being the number found next to your name in file `chunk_number_per_student.xlsx`. The keys are available in the file `keys.zip` and can also be found following the above link.

Your task is: to write a Matlab program to recover the log text of your chunk and submit your working code and chunk file for verification of correct operation. The recovered data from your chunk is expected to be English text (in character format), corresponding to the segment of the log file of the astronaut.

1. What is Known About the Data

NASA was able to establish the following facts about the recovered chunks:

1) During various stages of rebooting and recovery, the mission computer wrote each chunk on the storage units as a binary file, i.e., numbers of one of the three types: `double`, `single` or `int32`. Given that the system was malfunctioning, the data type of each chunk is not known and must be found in order to read the data correctly.
2) The data of each chunk is ASCII codes of text. Therefore, when read correctly, each number within each chunk should be an **integer between 0 and 255**.
   (see https://en.wikipedia.org/wiki/ASCII#ASCII_printable_code_chart)
3) NASA inspected the ASCII conversion routine of the mission computer and found that the *codification* routine itself was also corrupted. *The codification algorithm is adding a different integer offset to each character. The integer offset is between 0 and 15, it is randomly generated and is what we will call the key (different for each chunk and given to you together with the chunk).* Specifically, the corruption resulted in the ASCII codes in the chunks being coded *with a delayed (i.e., shifted) version of the expected key, with a shift equal to S.* For instance:

original text	H	e	l	l	o		w	o	r	l	d	!	char
	72	101	108	108	111	32	119	111	114	108	100	33	ASCII code
0≤original *key*≤15	0	12	7	1	14	10	2	3	3	4	1	10	double
original text + original *key*	H	q	s	m	}	*	y	r	u	p	e	+	char
	72	113	115	109	125	42	121	114	117	112	101	43	ASCII code
shifted (*S*) *key*	3	4	1	10	0	12	7	1	14	10	2	3	double
original text + shifted *key*	K	i	m	v	o	,	~	p	⍰	v	f	$	char
	75	105	109	118	111	44	126	112	128	118	102	36	ASCII code **(format)**

[1] From imdb http://www.imdb.com/title/tt3659388/
The Martian (2015) PG-13 | 144 min | Adventure, Drama, Sci-Fi | 2 October 2015 (USA):
During a manned mission to Mars, Astronaut Mark Watney is presumed dead after a fierce storm and left behind by his crew. But Watney has survived and finds himself stranded and alone on the hostile planet. With only meagre supplies, he must draw upon his ingenuity, wit and spirit to subsist and find a way to signal to Earth that he is alive.

Rows in purple are the input data, and coloured in red the unknown information. The ASCII representation is shadowed in blue. Luckily, while the shift *S* of the *key* per chunk is unknown, NASA confirmed that it is fixed per chunk and that:

`0 ≤ S ≤ 31.`

2. Recommendations

In order to succeed, you have to write your own Matlab script that contains commands that achieve the following:

1) *Determine which format your chunk is written in.* Is it binary `double` (double-precision floating point), binary `single` (single-precision floating point), or binary `int32` (signed 32 bit integers)? How can you determine this, given you do not know what the contents are?
2) After you believe you have read your chunk data in the correct format, your next challenge is to find *the shift* `S` *for your codifying key, use it for shifting the expected key and* remove it from the received chunk. Here, you can consider searching for this manually (shifting the *key* with different *S*, removing the shifted *key* from the data and printing it as `char` in MATLAB to see if it is English text). Even better, see if you can automate the search by considering the fact that you know that the recovered text should be in English.
3) Once you have succeeded in the previous two steps, you will need to write code that automatically reads the chunk in the correct way, finds the shift *S* and then provides the English text as output. This will be your submission on Moodle, as described below.

3. Submission Information

You ***SHOULD NOT*** submit a report for this assignment. We ***ONLY*** ask for your MATLAB code, which should be appropriately commented to include your name and student number, as well an explanation for your approach and the associated commands you used (added as comments in the code). Please add at least 5 comments, but ideally more.

This code should also be submitted in a document (e.g., Word document) on Moodle (so we can check for plagiarism). The corresponding MATLAB file and the chunk data **should also be submitted as a ZIP file on Moodle so we can run it**. *In order to receive 100% on this assignment*, your code must execute with your chunk data in MATLAB, and the output should be the recovered text and any other supplementary information you would like to provide. If this is not achieved, your mark will depend on how far you have progressed with steps 1-3 from Section 2.

You must submit your document and ZIP file no later than **10am on Monday 15th January 2018**. Late submissions will incur a penalty.

Please note that you can discuss the assignment with other students, **but each student must write and submit their own code on their own chunk.** No collaboration in the writing of MATLAB code is permitted, and each of you should develop their own solution. Submissions that have been plagiarised will receive a zero mark, and, depending on the level of plagiarism, this will be reported and investigated.

4. Useful Material to Consider

As mentioned in the previous lab, after having covered basic mathematical operations, numerical data handing and plotting in MATLAB, type `helpwin`, and make sure to check (and try) most of the commands in the following toolboxes.

`elfun`,
`elmat`,
`matfun`,
`lang`,
`datatypes` (strings, integers, cell arrays, etc), and
`general`

The commands in these toolboxes will be very useful for this assignment. Specifically, amongst others, the next commands will be useful in your work:

- `fopen`, `fclose`, `fread`, `fwrite`, `fseek`
- `char`, `fprintf`, `disp`, `load`, `circshift`, `length`, `find`, `strfind`, `cell`
- `if`, `while`, `for`, `switch`