

Lab Course

Scientific Computing

Worksheet 1

distributed: 02.11.2017

due to: 12.11.2017, midnight (submitted on the Moodle page)

personal presentation: 14.11.2017 (exact slots will be announced)

In this worksheet, we study some numerical aspects of the classical two-player game *rock-paper-scissors*, using **MATLAB**. In particular, we focus on simple strategies that, in the long run, could benefit both the computer and the human player.

The provided files are:

- `game.fig` contains the graphical user interface and **should not be modified**
- `game.m` contains the initialization of the game and the rest of the logic (e.g. handling events) related to the GUI. It **may be modified**
- `mchoice.m` contains the “brain” of the game. It **may be modified**
- `gen_human_move.m` is a function that generates moves for the human player. It can be used to run a user-defined number of rounds automatically. For the purposes of this worksheet, it **should not be modified**

The GUI contains a button **Save data** that permits you to create a `.mat` file containing the *last transition matrix of the game* and *the results after each round*.

- a) In `mchoice.m`, fill in the implementation of `predict1` such that
- i) it takes two input arguments, j - the previous move of the human player, and $transm$ - the current predicted transition matrix for the human player
 - ii) it prints the input parameter $transm$
 - iii) it uses two global variables $param_a$, $param_b$ representing probabilities satisfying $0 \leq param_a \leq param_b \leq 1$
 - iv) it computes the variable $hnext$, representing the *predicted next move of the human player*, using the following algorithm
 - 1) generate a *uniform* random variable using the built-in function `rand`
 - 2) remembering that $0 \leq param_a \leq param_b \leq 1$, compute $hnext$ as

$$hnext = \begin{cases} \text{mod}(j, 3) + 1 & \text{with probability } param_a \\ \text{mod}(j + 1, 3) + 1 & \text{with probability } param_b - param_a \\ j & \text{with probability } 1 - param_b \end{cases} \quad (1)$$
- using the generated uniform random variable to decide which branch to follow
- v) it returns the variable $next = winchoice(hnext)$
- b) Analyze the three prediction policy functions (the one implemented in **a**) and the two existing ones, found in `mchoice.m`) and fill in the entries of the following table with *yes/no*:

Method \ Property	Deterministic	With memory
predict1		
predict2		
predict3		

- c) Find (experimentally) the convergence limit of the transition matrix $transm$ using the *predict2* policy and the default strategy (implemented in `gen_human_move.m`) for the human player
- Hint:** Perform simulations with an increasing number of rounds

- d) Using the functions `initmchoice`, `updatesamplem`, and `updatetransm`, implement the matrices `samplem2` and `transm2` corresponding to the sample matrix and transition matrix of the computer player, respectively. Print the computed *transm2* matrix at the end of `mchoice`. Additionally, add *transm2* to the list of variables saved to file in `game.m`.

Hint: use the last two moves of the computer player saved in the *history* array as additional parameters of the `mchoice` function.

- e) Similarly to **c)**, find (experimentally) the convergence limit of the transition matrix *transm2* using *predict3* policy and the default strategy provided by the GUI for the human player
- f) Fill in the matrices below with the results from **c)** and **e)**

$$transm \rightarrow \left[\begin{array}{cc} & \\ & \end{array} \right] \quad transm2 \rightarrow \left[\begin{array}{cc} & \\ & \end{array} \right]$$

Questions:

- 1) What is the type of the transition matrix of the game? What properties does it posses?
- 2) Related to exercise **c)**: For which of the three prediction strategies of the computer player can it be (intuitively) guaranteed that, in the long run (i.e. an infinite number of rounds), a winning strategy for the human player can be found?