**DCS 307: Simulation**

**Random Number Generator Streams**

**In-Class Work: 01 Feb 2023**

Name: Kevin Chen

1. What are the first ten service times from the first run?

**[1] 0.6906071 [2] 0.2777054 [3] 0.1499705**

**[4] 0.6604378 [5] 0.5704503 [6] 0.1073709**

**[7] 0.3050448 [8]0.6405310 [9] 1.4994352**

**[10] 0.2054382**

1. What are the first ten service times from the second run?

**[1] 0.6906071 [2] 0.2777054 [3] 0.1499705**

**[4] 0.6604378 [5] 0.3885470 [6] 2.7028233**

**[7] 1.4994352 [8] 0.8472361 [9] 1.8154944**

**[10] 2.7168156**

1. For what customer, and what are the different values, where the first service time diverges between the two runs?

**For the 5th customer. The values of service time for customer 5 in first and second run are 0.5704503 and 0.3885470 respectively.**

1. In the first run, how many arrivals occur **after** the fourth arrival, but before that fourth customer completes service?

**Two arrivals occur after the fourth arrival, but before the fourth customer completes service i.e., 5th and 6th customer.**

1. In the second run, how many arrivals occur **after** the fourth arrival, but before that fourth customer completes service?

**Three arrivals occur after the fourth arrival, but before the fourth customer completes service i.e., 5th customer.**

1. What are the arrival times (rate = 1) generated using **rexp** (no streams)?

**[1] 1.662163**

**[2] 1.223265**

**[3] 0.7673412**

1. What are the service times (rate = 10/9) generated using **rexp** (no streams)?

**[1] 0.3114826**

**[2] 0.2777054**

**[3] 0.6088186**

1. Now what are the arrival times (rate = 1) generated using **rexp** (no streams)?

**[1] 1.662163**

**[2] 0.7673412**

**[3] 0.3085615**

1. Now what are the service times (rate = 10/9) generated using **rexp** (no streams)?

**[1] 1.100939**

**[2] 0.3114826**

**[3] 0.6088186**

1. What are the arrival times (rate = 1) generated using **vexp** (with streams)?

**[1] 0.2334595**

**[2] 1.07828**

**[3] 0.7855611**

1. What are the service times (rate = 10/9) generated using **vexp** (with streams)?

**[1] 0.0305915**

**[2] 0.1224158**

**[3] 1.42499**

1. Now what are the arrival times (rate = 1) generated using **vexp** (with streams)?

**[1] 0.2334595**

**[2] 1.07828**

**[3] 0.7855611**

1. Now what are the service times (rate = 10/9) generated using **vexp** (with streams)?

**[1] 0.0305915**

**[2] 0.1224158**

**[3] 1.42499**

1. What is the result of the following?

**sum(output1$interarrivalTimes - output2$interarrivalTimes)**

**Result: 1.595291**

1. What is the result of the following?

**sum(output1$serviceTimes - output2$serviceTimes)**

**Result: 0**

(complete reflection on next page)

**Reflection / Investigation:**

1. What is the default RNG used by Python? What is its period? List your source(s).

**The default RNG used by Python is Mersenne Twister which has a period of *2^19937-1***

**Source(s):** [random — Generate pseudo-random numbers — Python 3.11.1 documentation](https://docs.python.org/3/library/random.html)

1. What is the default RNG used by R? What is its period? List your source(s).

**The default RNG used by R is Mersenne Twister which has a period of *2^19937-1***

**Source(s):** [R: Random Number Generation (ethz.ch)](https://stat.ethz.ch/R-manual/R-devel/library/base/html/Random.html)

1. What additional RNGs are available in Python, and how do you set up code to use them? List your source(s).

**Additional RNGs available in Python:**

**[1] SystemRandom**

**[2] PCG32/64, Philox, SFC64 (numpy.random module)**

**[3] Xoshiro256, MT19937, PCG32/64 (randomgen and randomstate library)**

**These can be set using the relevant modules and libraries. For example: np.random.PCG64(seed)**

1. What additional RNGs are available in R, and how do you set up code to use them? List your source(s).

**Additional RNGs available in R:**

**[1] Wichmann-Hill**

**[2] Marsaglia-Multicarry**

**[3] Super-Duper**

**[4] L’Ecuyer-CMRG**

**These can be set using the `RNGkind` function. For example: RNGkind(“Super-Duper”)**

**Source(s):** [R: Random Number Generation (ethz.ch)](https://stat.ethz.ch/R-manual/R-devel/library/base/html/Random.html)

1. Given the article posted to Lyceum last class, which of the RNGs you mention in 1-4 above are presented in the article, and how does the article describe and/or rank their relative "goodness"? Include pointers to specific passages in the article.

**From the above stated RNGs, the ones mentioned in the article are:**

**[1] MT19937 (Mersenne Twister)**

**[2] PCG32/64**

**According to the passage *V.2.1 Results of Statistical Tests (pg. 31)* of the article, MT19937-32 is ranked 3rd (first level) and MT19937-64 is ranked 2nd (first level). PCG32 is ranked 2nd (first level).**

**In second level rank, MT19937-32 is placed at 4th and MT19937-64 is placed at 3rd. PCG32 is ranked 4th.**