MATS442 Stochastic simulation — Problems 1, 17.1.2020

The problems marked with **C** have a part which needs to be implemented in a computer programming language. Please return your implementation by email to ⟨santeri.j.karppinen(·at·)jyu.fi⟩.

1. Consider the rain drops example: Assume $(H_n)_{n\geq 1} \overset{\text{i.i.d.}}{\sim}$ Bernoulli($\pi/4$), and suppose that we estimate the value of π by computing Monte Carlo estimates $I_p^{(n)}(f)$ with $f(h) := 4h$.

(a) Compute $\operatorname{Var}_p\big(f(X)\big)$. (Hint: Remember that $\operatorname{Var}(Y) = \mathbb{E}[Y^2] - (\mathbb{E}Y)^2$.)

(b) Determine how big n must be chosen in order to attain an error less than 0.1 with probability at least 95%. Do this both with

i) The asymptotic error result.

ii) The bound from Chebychev's inequality.

(c) How does n change above if the error must be less than 0.01 (keeping the same probability $\geq 95\%$).

(d) (* optional) How does n change if the error must be less than 0.1 but the probability is $\geq 99\%$?

(e) (*) What about if you use the Hoeffding inequality?

2.C Consider the rain drops example as in Problem 1.

(a) Write a function which simulates n independent Bernoulli($\pi/4$) random variables and calculates an asymptotic confidence interval for the average.
Your function should be defined as

```
using Distributions, Statistics
function est_pi(n, pr)
    # Simulate, calculate mean m and confidence interval width d
    (est=m, tol=d)
end
```

where `n` is the number of smples and `pr` is the asymptotic probability of the confidence interval (like `0.95` for a 95% CI). The output `est` should contain your estimator (mean) and `tol` the confidence interval width, so that the CI is `est`±`tol`. (Hint: You may use `rand(Bernoulli(p), n)` to simulate `n` samples from Bernoulli(p), `mean` to calculate the average and `var` for the sample variance (or `std` for the standard deviation). The standard normal quantile function $\Phi(t)$ is `quantile(Normal(0,1), t)`.)

(b) Use your function to calculate the estimator and 95% confidence interval with 100 samples.

(c) Calculate 1000 estimators and related confidence intervals, and check how often the true value was within your confidence interval.

3. Show that if $U_1, U_2 \overset{\text{i.i.d.}}{\sim} \mathcal{U}(0,1)$, then the random vector (R,T) has a density

$$p_{R,T}(r,t) = \begin{cases} \frac{1}{2\pi} r e^{-r^2/2}, & 0 < t < 2\pi, 0 < r < \infty, \\ 0, & \text{otherwise,} \end{cases}$$

where $R = \sqrt{-2\log U_1}$ and $T = 2\pi U_2$.
(Hint: Observe first that R and T are independent.)

4.C Let p be the Cauchy p.d.f., that is, for $x \in \mathbb{R}$,

$$p(x) = \frac{1}{\pi(1+x^2)}.$$

(a) Find an algorithm which generates $X \sim p$ using $U \sim \mathcal{U}(0,1)$.

(b) Implement your method and check that you got it right by inspecting the histogram of 10000 simulated values.
(Hint: If `x` is a vector of simulated values, you may write

```
using Plots # Install by using Pkg; Pkg.add("Plots")
h = histogram(x, normalize=true, bins=LinRange(-4,4,20))
p(t) = 1/(pi*(1+t^2))  # same as function p(t) 1/(pi*(1+t^2)) end
t = LinRange(-4, 4, 500)
plot!(h, t, p.(t))
```

which shows the histogram within $[-4,4]$ normalised so that it matches the density.)

5. Suppose $p : \mathbb{R} \to (0,\infty)$ is a continuous p.d.f., and let $F(x) = \int_{-\infty}^{x} p(y)\mathrm{d}y$ denote the corresponding c.d.f..

Let $U \sim \mathcal{U}(0,1)$ and $-\infty < a < b < \infty$. Find the distribution of

(a) $Y_b := F^{-1}(F(b)U)$,

(b) $Y_{a,b} := F^{-1}\big(F(a)(1-U) + F(b)U\big)$, and

(c) $Y_a := F^{-1}\big(F(a) + (1-F(a))U\big)$.

(Hint: Find first the distribution of $Z_b := F(b)U$, $Z_{a,b} := F(a)(1-U) + F(b)U$ and $Z_a := F(a) + (1-F(a))U$, and draw a picture.)

6.C The early PRNGs were mostly linear congruential generators, that is, based on a recursion of the form

$$Z_n = (aZ_{n-1} + c) \mod M,$$

with some parameters $a, c, M \in \mathbb{N}$ with $a, c < M$ and a seed $Z_0 < M$. Then, $U_n := Z_n/M$ were "random uniform $\mathcal{U}(0,1)$."

Implement the above PRNG with $M = 2^{31}$, $a = 2^{16} + 3$ and $c = 0$ and any $Z_0 \in [0, M)$ you want.

(a) Plot 100 samples produced by your generator (`plot(u)`), and inspect visually whether the produced samples seem i.i.d. $\mathcal{U}(0,1)$.

(b) Produce 3000 uniform samples from a unit cube with your generator, that is, $V_k = (U_{3k+1,3k+2,3k+3})$, and plot them and inspect the randomness as above.
(Hint: Produce 9000 points from your generator to the vector `u` and you can plot them by:

```
using Plots; plotly()
scatter(u[1:3:end], u[2:3:end], u[3:3:end], markersize=1)
```

You can rotate the figure with mouse.)