Aalto University
School of Business

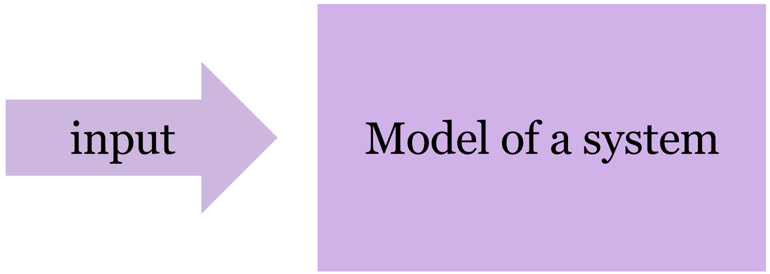
Business Analytics 2 Lecture 3: Monte Carlo Simulation

- *Motivation*
- *Simulation examples with discrete and continuous random variables*
- *Inverse CDF method*

Monte Carlo Simulation - Motivation

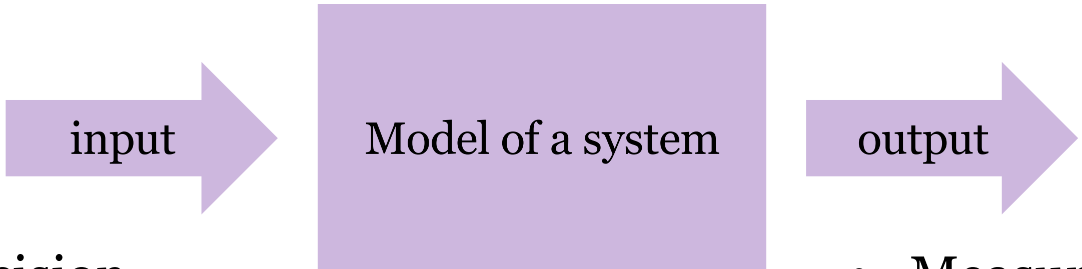
- Probabilistic models are often built to calculate performance measures:
  - Expected values of random variables
    - E.g., expected revenue from trainline ticket sales when demand is not exactly known
  - Event probabilities
    - E.g., probability of revenue below 100,000€
- Closed-form calculations are desirable but
  - can be difficult to obtain even in 'simple' models
  - their presentation may be meaningless for DMs not trained in analytics

- Simulation: use the model to generate 'alternative realities' to evaluate decision outcomes
- Monte Carlo (MC) simulation:
  - Generate samples from a probability model using a computer
  - Use the samples to estimate expected values and event probabilities



To see how the model of a system behaves, we simulate it by feeding inputs to it and observe the outputs

Logical description of relationships
between inputs and outputs



- Decision variables
- Uncontrollable parameters, often contain uncertainty

- Measures of performance or behaviour of the system

Feeding probabilistic inputs to the simulation model provides a systematic way to examine the output

Monte Carlo simulation of a Probability model

Probability model

- Random variable X that follows distribution f_X

$$E[X]$$

$$E[g(X)]$$

$$P(a < X \leq b)$$

Monte Carlo simulation

- Sample of random numbers $(x_1, \ldots, x_n)$ from distribution f_X

$$\frac{\sum_{i=1}^{n} x_i}{n}$$

$$\frac{\sum_{i=1}^{n} g(x_i)}{n}$$

$$\frac{|\{i \in \{1, \ldots n\} | x_i \in (a, b]\}|}{n} \quad (*$$

*) Collect all observations x_i that belong to the interval between *a* and *b*, and compute the size of the resulting set (|*set*| denotes the number of elements in the set)

Inverse CDF Method for Sampling Continuous random variables

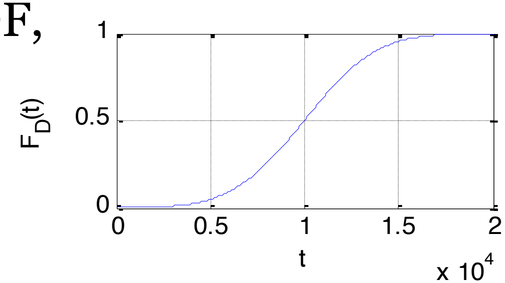
- F_X^{-1} denotes the inverse function of the CDF of r.v. X
- Random variable $Y = F_X^{-1}(U)$, where U~UNI(0,1), follows the same distribution as X.

Why? Because:

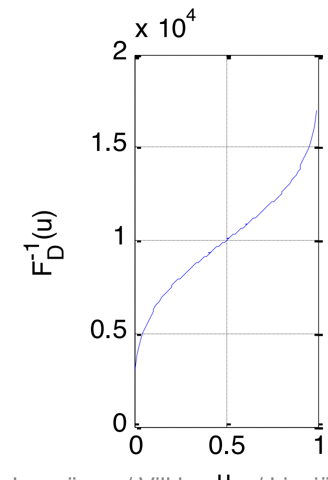
$$F_Y(t) = P(Y \leq t)$$
$$= P(F_X^{-1}(U) \leq t)$$
$$= P(F_X(F_X^{-1}(U)) \leq F_X(t))$$
$$= P(U \leq F_X(t)) = F_X(t)$$

$F_X(F_X^{-1}(U)) = U$ $\qquad$ $P(U \leq b) = b$

- Example: Inverse CDF method for D~N(10000,3000²)
  - $D = F_D^{-1}(U)$, where F_D^{-1} is the inverse of the N(10000,3000²) CDF, follows the same distribution as D
  - Demand is given by D=$F_D^{-1}(U)$

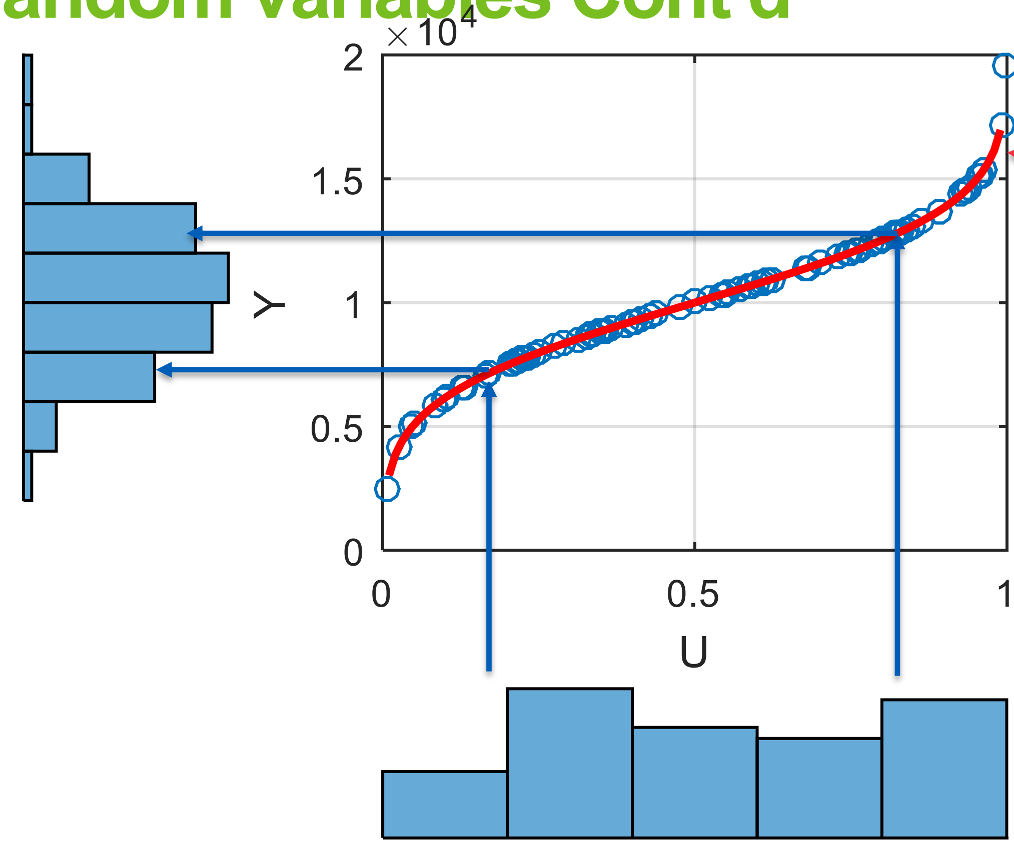


N(10000,3000²) CDF



Inverse of N(10000,3000²) CDF

Inverse CDF Method for Sampling Continuous random variables Cont'd



$$Y = F_D^{-1}(U),$$

F_D^{-1} is the inverse of the N(10000,3000^2) CDF

Case: Risk analysis in product development

A company is producing new products on a make-to-order basis

Known	Not known
• Unit selling price €249 • Admin cost €400,000 • Marketing costs €600,000	• Unit labour costs c_1 • Unit part costs c_2 • Demand D (in units) for the first year on the market

What is the profit function?

What are the best-case and worst-case profits?

	Best case	Worst case
Demand D	28,500 (approx.)	1,500 (approx.)
Unit labour cost c_1	€43	€47
Unit part cost c_2	€80	€100

Aalto University
School of Business

Case: Risk analysis in product development

Assume that we also know:

- Unit labour costs c_1 have the following distribution: $\{€43, 0.1; €44, 0.2; €45, 0.4; €46, 0.2; €47, 0.1\}$
- Unit part costs c_2 can range between €80 and €100 and are uniformly distributed
- Demand D is normally distributed with mean 15,000 units and standard deviation 4,500 units

Case: Risk analysis in product development

What are the inputs, model, and outputs?

Case: Risk analysis in product development

inputs

$$P(D, c_1, c_2) = (249 - c_1 - c_2) \times D - 10\text{^}6$$

outputs

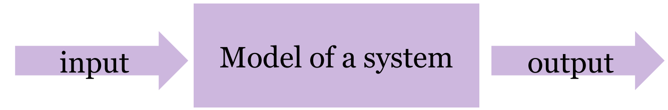
Probabilistic values of D, c_1, c_2 from their probability distributions

Distribution of values for profit P

Case: Risk analysis in product development R / Python model workflow

- Excel is convenient for communicating models, but due to repetitions needed in simulations, a programming language is a better choice
- Two nested parts:
  - First part: calculating one simulation run
  - Second part: repeating the inner loop many times using a for-loop
- Generating random numbers at each iteration:
  - Python: `random` and `NumPy` packages, e.g. `random.choice()` or `numpy.random.normal()`
  - R: using native functions `runif()` or `rnorm(1, mean=150, sd=45)`

Why do we need to repeat the simulation many times?



- Law of Large Numbers: when independent random draws are repeated many times, the distribution of the draws approaches the underlying ‘true’ distribution
- To be more precise, the average of the realised values approaches the expected value of the sampling distribution
- Gambler’s fallacy: believing that a small number of random value draws provides a ‘balanced’ set of values around true expected value

Case: Simulating waiting times in a service operation

- Customers arrive at a service operation so that there are on average 0.125 arrivals per minute
- It takes 5 minutes to serve a customer; What is the probability that a new arriving customer needs to wait in a queue?
- Exponential distr. CDF: $F(x) = 1 - e^{-\lambda x}$, where $\lambda = 0.125$
- Inverse function method to generate interarrival times: find x from the Exponential distr. CDF

$$e^{-\lambda x} = 1 - F(x) \Rightarrow -\lambda x = \ln\big(1 - F(x)\big) \Rightarrow x = -\frac{1}{\lambda}\ln(1 - F(x))$$