Monte Carlo Methods

Logistics and Supply Chain Analytics

Jiahua Wu

382 Business School
j.wu@imperial.ac.uk

Imperial College Business School

Imperial means Intelligent Business

Outline

- Decision making under uncertainty
- Monte Carlo methods
    - Random variable generation
    - Monte Carlo integration

Outline

- Decision making under uncertainty
- Monte Carlo methods
  - Random variable generation
  - Monte Carlo integration

Decision Making under Uncertainty

- In deterministic settings, each particular decision produces a certain, non-random outcome
- In practice, a decision must often be made before all factors that impact the outcome are known
  - The choice of a particular value for the inventory of a product is made before the demand for the product is known
  - The unknown demand D can be modeled as a random variable
- A decision leads to a distribution of costs/profits, rather than a certain, fixed cost/profit value
- For each decision, we must know how to calculate a distribution for any key performance indicator (such as profit, cost and etc.)

An Example: Evaluating a Wireless Data Plan

- A business analytics consultant based in London is considering changing her wireless data plan to accommodate her family's growing use of video streaming services
  - Her current plan "Family Share": £1 for each GB of data her family uses in a given month
  - After doing research on data plans offered by her wireless carrier, she is considering the plan "Superior Share"
    - A fixed fee of £16 for up to 20GB of data per month
    - Any data usage above the threshold will be charged at the rate of £1.5 per GB
    - Unused data under 20GB will not "roll over" to the next month

An Example: Evaluating a Wireless Data Plan

- A decision making problem under uncertainty
  - Input: Monthly usage
  - Output: Monthly payment
  - At the time of her decision to purchase the plan, she does not know exactly what her family's future data usage will be

Monthly Payment under Old Plan

- Based on the analysis of her family's past monthly data usage values, the consultant predicted data usage in any month as a normal random variable with a mean of 23GB and a standard deviation of 5GB
- Then, if the consultant stays with her current data plan, her actual monthly payment is a normal random variable with a mean of £23 and a standard deviation of £5

Monthly Payment under New Plan

- We can calculate the monthly payment value P (in £) for any value of data usage U (in GB)

$$P = 16 + 1.5 \cdot (U - 20)^{+},$$

  where $(y)^{+} = y$ if $y \geq 0$, and 0 otherwise
- What is the expected monthly payment under the new data plan?
  - The expected value of U is 23
  - So, should the expected value of P be

$$16 + 1.5 \times (23 - 20) = 20.5?$$

  - In general, we do not get the correct value for the expected monthly payment that way

Outline

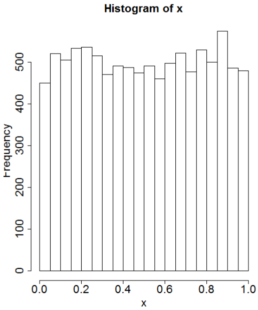
- Decision making under uncertainty
- Monte Carlo methods
  - Random variable generation
  - Monte Carlo integration

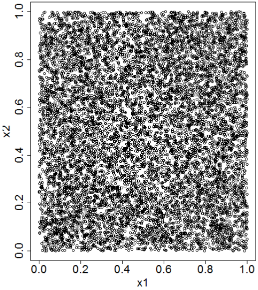
Random Variable Generation

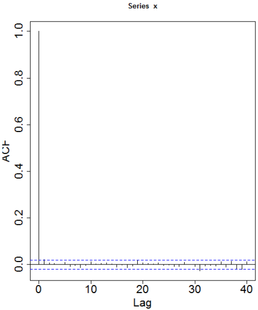
- Monte Carlo methods rely on
  - the possibility of producing a supposedly endless flow of random variables
  - for well-known or new distributions
- `R` has a large number of functions that will generate the standard random variables
  - `runif(100, min=2, max=5)` produces 100 independent generations from a $U(2,5)$ distribution
  - It is therefore counter-productive and inefficient to generate from those standard distributions
  - Rule of thumb: if it is built into R, use it

Some Comments

- A quick check on the properties of this uniform generator is to
  - look at a histogram of the X_i's (left)
  - plot the pairs (X_i, X_{i+1}) (center)
  - look at the estimate autocorrelation function (right)







Some Comments

- Remember: `runif` does not involve randomness per se
- It is a deterministic sequence based on a random starting point
- The R function `set.seed` can produce the same sequence

```
> set.seed(1)
> runif(5)
[1] 0.2655087 0.3721239 0.5728534 0.9082078 0.2016819
> set.seed(1)
> runif(5)
[1] 0.2655087 0.3721239 0.5728534 0.9082078 0.2016819
> set.seed(2)
> runif(5)
[1] 0.1848823 0.7023740 0.5733263 0.1680519 0.9438393
```

- Setting the seed determines all the subsequent values

Generating Non-Standard Random Variables: The Inverse Transform

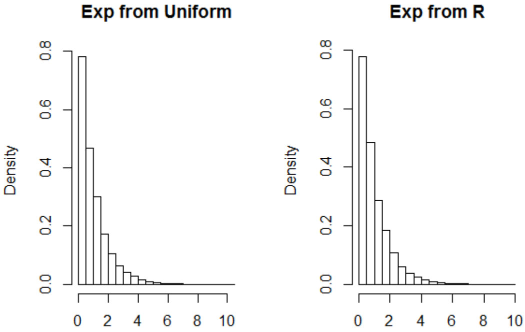
- The probability integral transform allows us to transform a uniform into any random variable
  - If X has density f and cdf F, then we have the relation

$$F(x) = \int_{-\infty}^{x} f(t)\,dt.$$

  - We set $u = F(x)$ and solve for x
    - Eg. If $X \sim Exp(1)$, then $F(x) = 1 - e^{-x}$. Solving for x in $u = 1 - e^{-x}$ gives $x = -\log(1-u)$

Generating Exponentials

```
> Nsim=10^4 #number of random variables
> U=runif(Nsim)
> X=-log(1-U) #transforms of uniforms
> Y=rexp(Nsim) #exponentials from R
> par(mfrow=c(1,2)) #plots
> hist(X,freq=F,main="Exp from Uniform")
> hist(Y,freq=F,main="Exp from R")
```



Generating Non-Standard Random Variables: The Inverse Transform

- The inverse transform
  - Is useful for many probability distributions
  - Requires to compute the inverse of CDFs
  - May be computationally inefficient for those distributions without closed-form CDFs
- For those cases, we must turn to indirect methods
  - Accept-reject method
    - This method is extremely powerful - allows us to simulate virtually any distribution
  - Ziggurat algorithm

Outline

- Decision making under uncertainty
- Monte Carlo methods
  - Random variable generation
  - Monte Carlo integration

Monte Carlo Integration

- The generic problem: Evaluate

$$E_f[h(X)] = \int_{\mathcal{X}} h(x) f(x) \, dx,$$

  - f is a probability density function
  - X takes its values in $\mathcal{X}$
- One potential approach: deterministic numerical integration
  - R functions: `area` and `integrate`
  - Ok in low (one) dimensions

Monte Carlo Integration

- The Monte Carlo Method
  - Generate a sample $(X_1, \ldots, X_n)$ from the density f
  - Approximate the integral with

$$\bar{h}_n = \frac{1}{n}\sum_{j=1}^{n} h(x_j)$$

- The convergence

$$\bar{h}_n = \frac{1}{n}\sum_{j=1}^{n} h(x_j) \to \int_{\mathcal{X}} h(x) f(x)\, dx = E_f[h(X)]$$

is guaranteed by the Strong Law of Large Numbers

Monte Carlo Method: The Example

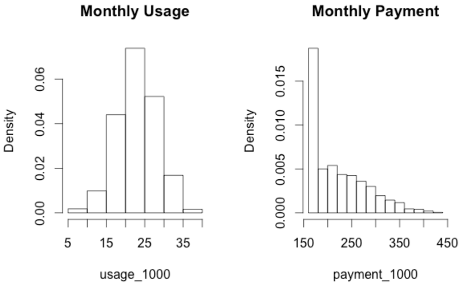
- We first simulate values of monthly data usage
  - Validate the random variable generation by comparing the sample mean and standard deviation with the true values

```
> set.seed(1)
> usage_10 <- rnorm(10, mean=23, sd=5)
> mean(usage_10)
[1] 23.43315
> sd(usage_10)
[1] 5.477069
> usage_1000 <- rnorm(1000, mean=23, sd=5)
> mean(usage_1000)
[1] 22.91002
> sd(usage_1000)
[1] 5.1826
```

- Longer simulations produce more precise estimates
- Random seed value does not matter much when you run a simulation with large number of simulation runs

Simulation Output

- Histograms are often useful for gaining intuition about the inputs and the outputs involved in a simulation
  - In the data plan example, the random input is the data usage U, and the random output is the monthly payment P



Expected monthly payment under the new plan: £21.9

Monte Carlo Methods

- First construct a model connecting inputs to outputs
  - What are the random inputs?
  - What are the outputs of interest?
  - Define mathematical relationship determining outputs as a function of inputs?
- Run the simulation
  - Generate many possible values that random inputs may take
  - For each sequence of events, (compute and) record the outputs
- Analyze the output
  - Distribution of the outputs: average, standard deviation...