Performance Modeling of Computer Systems and Networks

Prof. Vittoria de Nitto Personè

Generating Continuous Random Variates

Università degli studi di Roma Tor Vergata
Department of Civil Engineering and Computer Science Engineering

1

queste sono le variabili random continue con cui avremo a che fare:

Prerequisite

We assume the knowledge of continuous random variables (sect.7.1).

In particular:

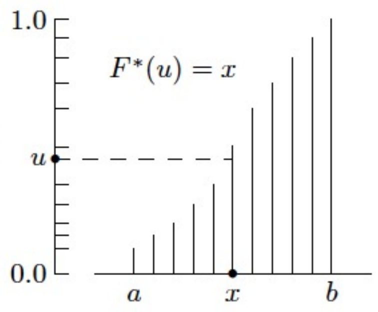
- *Uniform*(a,b)
- *Exponential*(μ)
- *Normal*(μ,σ)
- *Lognormal*(n,b)
- *Erlang*(n,b)
- *Student*(n)

2

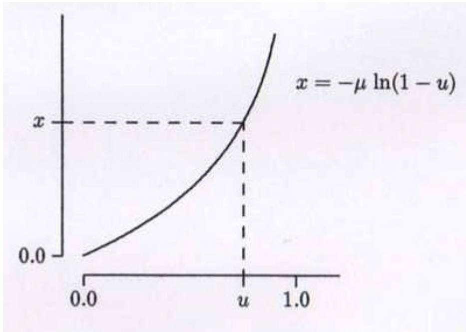
caso continuo piu facile:
nel discreto, abbiamo un 'unico' x valido per un 'insieme'.
nel continuo, non abbiamo questo concetto di 'insieme'.

Discrete Random Variates

$$F^*(u) = \min_x \{x : u < F(x)\}$$



Continuous Random Variates



Prof. Vittoria de Nitto Personè 3

3

qui si parla di inversa vera, concettualmente faccio stessa cosa

Discrete Simulation
Generating Continuous Random Variates

Preliminary Definitions

The *inverse distribution function* (idf) of X is the function $F^{-1} : (0, 1) \rightarrow \chi$, $\forall\, u \in (0, 1)$ as

$$F^{-1}(u) = x$$

where $x \in \chi$ is the unique possible value for $F(x) = u$

There is a one-to-one correspondence between possible values $x \in \chi$ and cdf values $u = F(x) \in (0, 1)$

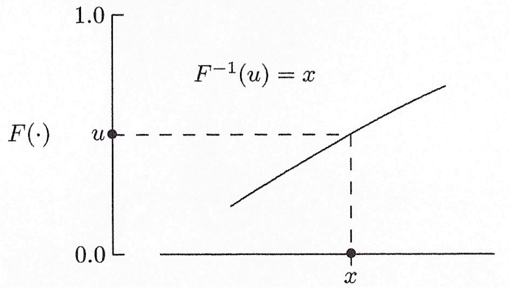
Prof. Vittoria de Nitto Personè 4

4

alcune inverse sono facili (tipo exp), altre meno.

Continuous Random Variable idfs

- Unlike the a discrete random variable, the idf for a continuous random variable is a true inverse



- Can sometimes determine the idf in "closed form" by solving $F(x) = u$ for x

5

queste sono facili, cioè si usano inverse specifiche. altre si valutano in modo algoritmico.

Discrete Simulation
Generating Continuous Random Variates

Examples

- if X is *Uniform(a,b)*, $F(x) = (x-a)/(b-a)$ for $a < x < b$

$$x = F^{-1}(u) = a + (b-a)u \qquad 0 < u < 1$$

- if X is *Exponential(μ)*, $F(x) = 1\text{-exp}(-x/\mu)$ for $x > 0$

$$x = F^{-1}(u) = -\mu \ln(1-u) \qquad 0 < u < 1$$

- if X is a continuous variable with possible value $0 < x < b$ and pdf $f(x) = 2x/b^2$, cdf $F(x) = (x/b)^2$

$$x = F^{-1}(u) = b\sqrt{u} \qquad 0 < u < 1$$

6

vale lo stesso teorema del discreto, ci tranquillizza su quello che possiamo fare.

Random Variate Generation By Inversion

- X is a continuous random variable with idf $F^{-1}(\cdot)$
- Continuous random variable U is *Uniform*(0,1)
- Z is the continuous random variable defined by $Z = F^{-1}(U)$

Theorem
Z and X are identically distributed

Algorithm 1

```
u = Random();
return F-1(u);
```


7

applicazione in pseudocodice di alcune distr.

Inversion examples

- *Uniform*(*a*,*b*) Random Variate

```
u = Random();
return a + (b - a) * u;
```

- *Exponential*(μ) Random Variate

```
u = Random();
return - μ log(1-u);
```


8

alcune caratteristiche che devono avere gli algoritmi:

06/05/21

sincronizzato: una chiamata all'algoritmo corrisponde una chiamata alla random.
nei casi semplici uso tecniche banali, se inversa esplicita non semplice, approssimo oppure la determino numericamente con metodi di analisi (faccio approssimazioni).

Inversion algorithms

- Algorithms in the previous two examples are:
  - portable, exact, robust, efficient, clear, synchronized and monotone
- It is not always possible to solve for a continuous random variable idf explicitly by algebraic techniques
- Two other options may be available:
  1. Use a function that accurately *approximates* $F^{-1}(\cdot)$
  2. Determine the idf by solving $u = F(x)$ *numerically*

  (see section 7.2.2)

9

prendo n variate generate by generatore, n grande, vedo media e varianza.
le confronto con valori teorici, dovrebbero essere vicini.

Testing for correctness

- generate a sample of n random variates where n is large
- evaluate sample mean and standard deviation
- compare them with the theoretical values,
  they should be *reasonably* close !!

This is not enough!! Different distributions can have the same mean and standard deviation !!!

- generate a sample of n random variates and construct a k-bin continuous-data histogram with bin width δ
- f' is the histogram density and $f(x)$ is the pdf

= pdf teorica

$$f' \to f(x) \quad \text{as } n \to \infty \quad \text{and} \quad \delta \to 0$$

- In practice, using a large but finite value of n and a small but non-zero value of δ, perfect agreement between f' and f will not be achieved

Discrete case: natural sampling variability !
Continuous case: variability+binning !!

10

non devo fidarmi completamente, perchè distribuzioni diverse posso avere media e std deviazione uguali. Serve nuovamente istogramma.
nel caso continuo vanno fissati numero bean e ampiezza bean. (se delta bean piccolo, molti bean ma possibile avere rumore, altrimenti delta bean grandi, pochi bean, forse meno precisi). esempio: ho spazio 10, se uso 5 bean, ciascuno ampiezza 2, se uso 2 bean, ampi 5.

il troncamento concettualmente è uguale, in pratica è più semplice perchè ho la vera inversa. ho quindi una trasformazione esatta.

Truncation

- Let X be a continuous random variable with possible values χ and cdf $F(x)=\Pr(X \leq x)$
- Suppose we wish to restrict the possible values of X to $(a, b) \subset \chi$

It is similar to, but simpler than truncation in the discrete-variable context

- X is $\leq a$ with probability $\Pr(X \leq a) = F(a)$
- X is $\geq b$ with probability $\Pr(X \geq b) = 1 - \Pr(X < b) = 1 - F(b)$
- X is between a and b with probability

$$\Pr(a < X < b) = \Pr(X < b) - \Pr(X \leq a) = F(b) - F(a)$$

11

2 cases for truncation

Case 1

if a and b are specified, the cdf of X can be used to determine the left-tail α, right-tail β truncation probabilities

$$\alpha = \Pr(X \leq a) = F(a) \quad \text{and} \quad \beta = \Pr(X > b) = 1\text{-}F(b)$$

Case 2

if α and β are specified, the idf of X can be used to determine left and right truncation points

$$a = F^{-1}(\alpha) \quad \text{and} \quad b = F^{-1}(1\text{-}\ \beta)$$

$F(b) = 1\text{-}\beta$

Both transformations are exact !

12

Library `rvgs`

- Contains 7 continuous random variate generators
  - `double Chisquare(long` *n*`)`
  - `double Erlang(long` *n*`, double` *b*`)`
  - `double Exponential(double` *μ*`)`
  - `double Lognormal(double` *a*`, double` *b*`)`
  - `double Normal(double` *μ*`, double` *σ*`)`
  - `double Student(long` *n*`)`
  - `double Uniform(double` *a*`, double` *b*`)`