Performance Modeling of Computer Systems and Networks

Prof. Vittoria de Nitto Personè

Discrete Random Variates: applications

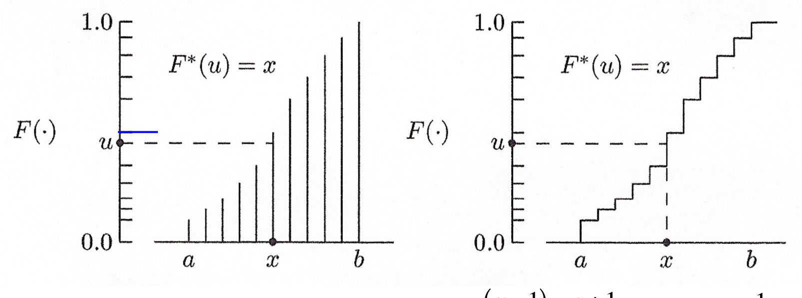
Università degli studi di Roma Tor Vergata
Department of Civil Engineering and Computer Science Engineering

1

Equilikely(a,b)

$$F(x) = \frac{x-a+1}{b-a+1} \qquad x = a, a+1, \ldots, b$$

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



$F(x\text{-}1) \leq u < F(x)$ $\longleftrightarrow$ $\dfrac{(x-1)-a+1}{b-a+1} \leq u < \dfrac{x-a+1}{b-a+1}$

2

$$\frac{(\cancel{x}-1)-a+\cancel{1}}{b-a+1} \le u < \frac{x-a+1}{b-a+1}$$

$$x-a \le (b-a+1)u < x-a+1$$

$$x \le a+(b-a+1)u < x+1$$

$$x = a+\lfloor (b-a+1)u \rfloor$$

$$F^*(u) = a+\lfloor (b-a+1)u \rfloor$$

model 1

Example: Inventory System

in program sis2, the demand per time interval is an *Equilikely*(10,50) random variate

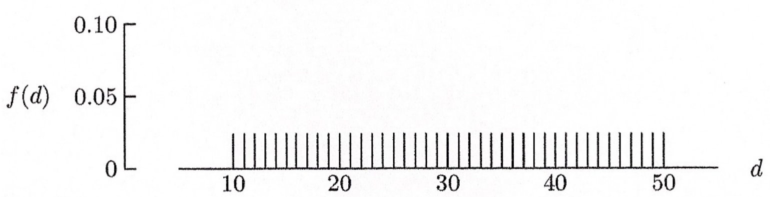
```
long Equilikely(long a, long b)
{  return (a + (long) ((b - a + 1) *
Random()));}

long GetDemand(void)
{
  return (Equilikely(10, 50)); }
…
 while (index < STOP) {
 index++;
 …
 inventory -= demand;}
```

model 1

Discrete Simulation
Discrete RV applications

Example: Inventory System

- μ=30, $\sigma = \sqrt{140} \cong 11.8$, and the demand pdf is flat



- this model is not very realistic

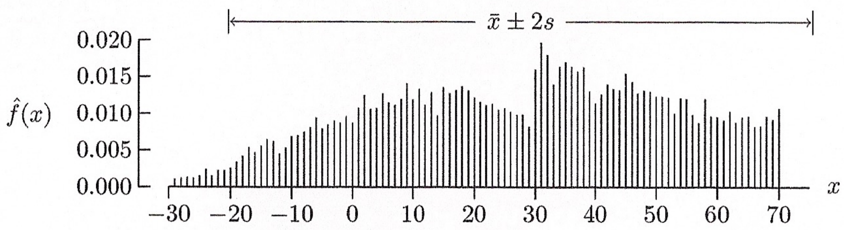
Prof. Vittoria de Nitto Personè 5

5

DE simulation
Discrete–Data Histograms

Example 3.1

- Using sis2mod to generate 10000 weeks of sample data, the inventory level histogram from sis2.out can be constructed (s,S)=(20,80)



- x denotes the inventory level prior to review
- $\bar{x} = 27.63, \quad s = 23.98$
- about 98.5% of the data falls within $\bar{x} \pm 2s$

Equilikely(10, 50) ?

Prof. Vittoria de Nitto Personè 6

6

model 2

Discrete Simulation
Discrete RV applications

Alternative Inventory Demand model

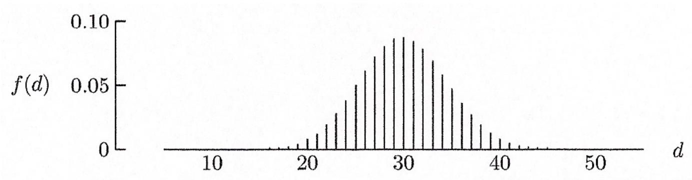
- 100 instances per time interval when demand for 1 unit may occur
- The probability of demand is 0.3 per instance (independently)

  This is *Binomial*(100,0.3) !

- the function `GetDemand` in sis2 becomes:

```
long GetDemand(void) {
        return (Binomial(100,0.3));
}
```

- μ=30, $\sigma = \sqrt{21} \cong 4.6$ and the pdf is:



Prof. Vittoria de Nitto Personè 7

7

model 3

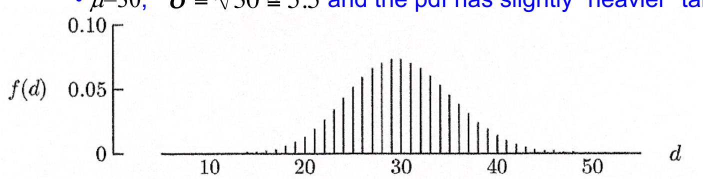
Discrete Simulation
Discrete RV applications

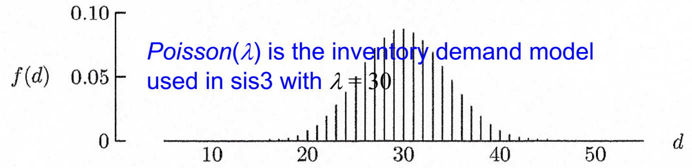
A *Poisson*(30) model

- *Binomial*(*n*,*p*) ≈ *Poisson*(*np*) for large n
- if *Binomial*(100,0.3) is realistic, should also consider *Poisson*(30)
- the function `GetDemand` in sis2 would be:

```
long GetDemand(void) {
        return (Poisson(30));
}
```

- μ=30, $\sigma = \sqrt{30} \cong 5.5$ and the pdf has slightly "heavier" tails:





Poisson(λ) is the inventory demand model used in sis3 with $\lambda = 30$

Prof. Vittoria de Nitto Personè 8

8

model 4

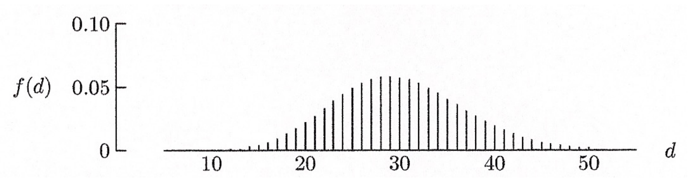
Discrete Simulation
Discrete RV applications

A *Pascal*(50,0.375) model

- 50 instances per time interval
- the demand per instance is *Geometric*(*p*) with p=0.375
- the function `GetDemand` in sis2 would be:

```
long GetDemand(void) {
        return (Pascal(50,0.375));
}
```

- μ=30, $\sigma = \sqrt{48} \cong 6.9$ and the pdf has heavier tails than the *Poisson*(30) pdf:



9

model 5

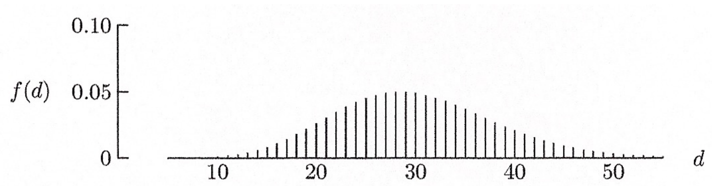
Discrete Simulation
Discrete RV applications

A compound demand model

- the number of demand instances per time interval is *Poisson*(50)
- the demand per instance is *Geometric*(*p*) with p=0.375
- the function `GetDemand` in sis2 becomes:

```
long GetDemand(void) {
        long instances = Poisson(50.0);   //must truncate to avoid 0
        return (Pascal(instances, 0.375));
}
```

- μ=30, $\sigma = \sqrt{66} \cong 8.1$ and the pdf has heavier tails



10

model 5

Discrete Simulation
Discrete RV applications

pdf for the compound demand

• define random variables

D: the demand amount
I: the number of demand instances per time interval

$$f(d) = \Pr(D = d) = \sum_{i=0}^{\infty} \Pr(I = i)\Pr(D = d | I = i) \qquad d = 0,1,2,\ldots$$

• to compute $f(d)$, truncate infinite sum: $0 < a \le i \le b$

```
/* use the library rvms */
double sum = 0.0;
for (i = a; i <= b; i++)
        sum += pdfPoisson(50.0,i) * pdfPascal(i,0.375,d);
return sum;
/* sum is f(d) */
```


Next-Event simulation
InvSys

Comparison of Demand Models

sis2: used an *aggregate* demand for each time interval, generated as an *Equilikely*(10,50) random variate

- Aggregate demand per time interval is random
- Within an interval, time between demand instances is constant
- Example: if aggregate demand is 25, inter-demand time is 1/25=0.04

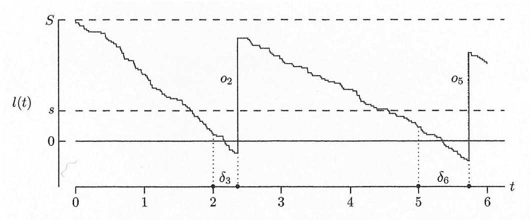
- Now (sis3) using *Exponential*($1/\lambda$) inter-demand times
  - Demand is modeled as an arrival process
  - Average demand per time interval is λ

```
double GetDemand(void)/* --------------------------------------------------- *
                      generate the next demand instance (time) with rate 30 *
                      per time interval and exactly one unit of demand per  *
                      demand instance  * ----------------------------------*/
{ static double time = START;
  SelectStream(0);
  time += Exponential(1.0 / 30.0);
return (time);}
```

model 6

Discrete Simulation
Discrete RV applications

sis3



Program sis4

- Based on sis3 but with a more realistic inventory demand model
- The inter-demand time is an *Exponential*($1/\lambda$) random variate
- Whether or not a demand occurs at demand instances is random with probability p
- To allow for the possibility of more than 1 unit of demand, the demand amount is a *Geometric*(p) random
- Expected demand per time interval is $\dfrac{\lambda p}{(1-p)}$

model 6

Discrete Simulation
Discrete RV applications

The auto dealership

- The inventory demand model for sis4 corresponds to λ customers per week on average
- Each customer will buy
  - 0 autos with prob $(1\text{-}p)$
  - 1 auto with prob $(1\text{-}p)p$
  - 2 autos con prob $(1\text{-}p)p^2$, etc.
- with λ=120 p=0.2, average demand is 30

$$30.0 = \frac{\lambda p}{(1-p)} = \lambda \sum_{x=0}^{\infty} x(1-p)p^x = \underbrace{\lambda(1-p)p}_{19.2} + \underbrace{2\lambda(1-p)p^2}_{7.68} + \underbrace{3\lambda(1-p)p^3}_{2.304} + \ldots$$

- $\lambda(1\text{-}p)$=96.0 customers buy 0 autos
- $\lambda(1\text{-}p)p$=19.2 customers buy 1 auto
- $\lambda(1\text{-}p)p^2$=3.84 customers buy 2 autos
- $\lambda(1\text{-}p)p^3$=0.768 customers buy 3 autos, etc.

model 6 sis4.c

uncoupled processes

```
double GetDemand(long *amount)
{
  static double time = START;
  SelectStream(0);
  time += Exponential(1.0 / 120.0);   /* demand instance */
  SelectStream(2);
  *amount = Geometric(0.2);           /* demand amount   */
  return (time);
}
...
if (t.current == t.demand) { /* process an inventory demand */
    sum.demand += amount;
    inventory  -= amount;
    t.demand    = GetDemand(&amount);
}
if (t.current == t.demand) { /* process an inventory demand */
     sum.demand++;
     inventory--;
     t.demand = GetDemand();
}
```

sis3.c

Truncation: examples

- In the previous example, no bound on number of autos purchased
- Can be made more realistic by <u>truncating</u> possible values
- Start with random variable X with possible values χ={0, 1, 2, …}
  and cdf $F(x)=\Pr(X \leq x)$
- want to restrict X to the finite range $0 \leq a \leq x \leq b < \infty$
- if $a > 0$, $\alpha = \Pr(X < a) = \Pr(X \leq a\text{-}1) = F(a\text{-}1)$
- $\beta = \Pr(X > b) = 1\text{-}\Pr(X \leq b) = 1\text{-}F(b)$
- $\Pr(a \leq X \leq b) = \Pr(X \leq b) - \Pr(X < a) = F(b) - F(a\text{-}1)$

  essentially, always true iff $F(b) \cong 1.0$ and $F(a\text{-}1) \cong 0.0$

The auto dealership

model 5

- the number of demand instances per time interval is *Poisson*(50)
- the demand per instance is *Geometric*(p) with p=0.375

For the *Poisson*(50) random variable I, determine a and b so that

$$\Pr(a \le I \le b) \cong 1.0$$

- use $\alpha = \beta = 10^{-6}$ and `rvms`

```
a = idfPoisson(50.0, α);
b = idfPoisson(50.0,1.0 - β);
```

- results: $a = 20$, $b = 87$
- consistent with the bounds produced by the conversion:

$\Pr(I < 20)$ = cdfPoisson(50.0, 19) $\cong 0.48 \times 10^{-6} < \alpha$
$\Pr(I > 87) = 1.0$ - cdfPoisson(50.0, 87) $\cong 0.75 \times 10^{-6} < \beta$

model 5

Effects of truncation

- truncating *Poisson*(50) to the range $\{20, \ldots, 87\}$ is insignificant: truncated and un-truncated random variables have (essentially) the same distribution

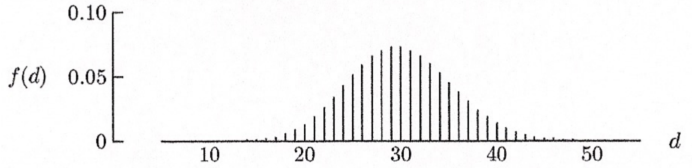
model 3

A *Poisson*(30) model

- *Binomial*(*n*,*p*) ≈ *Poisson*(*np*) for large n
- if *Binomial*(100,0.3) is realistic, should also consider *Poisson*(30)
- the function `GetDemand` in sis2 would be:

```
long GetDemand(void) {
        return (Poisson(30));
}
```

- μ=30, $\sigma = \sqrt{30} \cong 5.5$ and the pdf has slightly "heavier" tails:



19

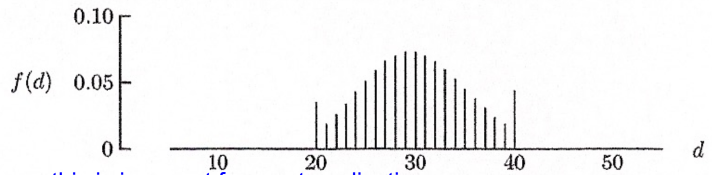
model 3

The auto dealership
incorrect truncation

- use *Poisson*(30) demand model in program sis2
- truncate the demand to the range $20 \le d \le 40$

```
d = Poisson(30.0);
if (d < 20)
        d = 20;
if (d > 40)
        d = 40;
return d;
```

- original left and right tails "grouped together" at 20 and 40



- this is incorrect for most applications

20

model 3

Discrete Simulation
Discrete RV applications

Truncation by cdf modification

- troncate *Poisson*(30) to range $20 \le d \le 40$
- the *Poisson*(30) pdf is:

$$f(d) = \exp(-30)\frac{30^d}{d!} \qquad d = 0,1,2,\ldots$$

$$\Pr(20 \le D \le 40) = F(40) - F(19) = \sum_{d=20}^{40} f(d) \cong 0.945817$$

- compute a new truncated random variable D_t with pdf $f_t(d)$

$$f_t(d) = \frac{f(d)}{F(40) - F(19)} \qquad d = 20,21,\ldots,40$$

model 3

Discrete Simulation
Discrete RV applications

Truncation by cdf modification

- the corresponding truncated cdf is

$$F_t(d) = \sum_{t=20}^{d} f_t(t) = \frac{F(d) - F(19)}{F(40) - F(19)} \qquad d = 20,21,\ldots,40$$

- mean and standard deviation of D_t

$$\mu_t = \sum_{d=20}^{40} d f_t(d) \cong 29.841 \qquad \sigma_t = \sqrt{\sum_{d=20}^{40} (d - \mu_t)^2 f_t(d)} \cong 4.720$$

- mean and standard deviation of *Poisson*(30)

$$\mu = 30.0 \qquad \sigma = \sqrt{30} \cong 5.477$$

model 3

Truncation by cdf modification

- A random variate truncated to $20 \leq d \leq 40$ can be generated by inversion, using the truncated cdf $F_t(\cdot)$ and the Algorithm 2

```
u = Random();
d = 30;
if (Ft(d) <= u)
      while (Ft(d) <= u)
            d++;
else if (Ft(20) <= u)
            while (Ft(d-1) > u)
                  d--;
      else
            d = 20;
return d;
```

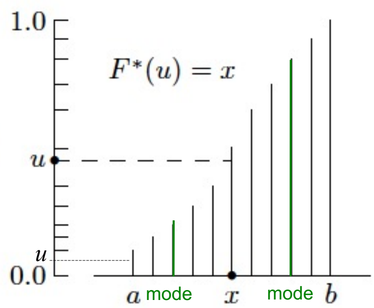
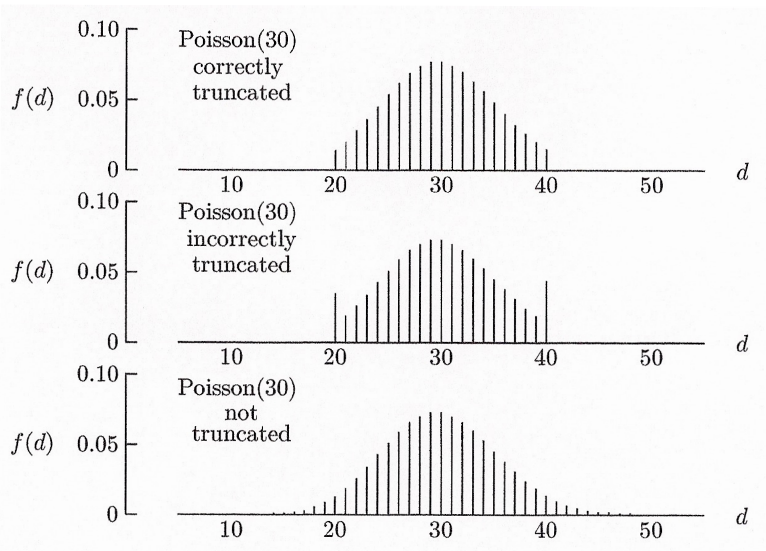


Illustration of pdfs



Truncation - conclusion

see the book for the following

1. the truncation by cdf modification in general
2. a different approach called *truncation by constrained inversion*
3. the simple technique *truncation by acceptance-rejection*

Important points

The modeler should be familiar with

- How these distribution arise
- The support, χ
- The mean, μ
- The variance, σ^2
- The shape of the pdf

- how these distributions relate to one another

Exercises

- Generating Discrete RV, use of libraries: exerc. 6.2.4
- Truncation: exerc. 6.3.1, 6.3.2