

# Specific p.d.f. generation

Marcin Chrzęszcz  
mchrzasz@cern.ch



University of  
Zurich <sup>UZH</sup>

Monte Carlo methods,  
14 April, 2016

There will be no lectures and class on 19<sup>th</sup> of May

# Exponential p.d.f.

⇒ The  $X(\theta, \lambda)$ :

$$\rho_{\theta, \lambda} = \frac{1}{\lambda} e^{-\frac{x-\theta}{\lambda}}$$

⇒ One can transform the variable:

$$x \rightarrow x' = \frac{x - \theta}{\lambda} \Rightarrow E(\theta, \lambda) \rightarrow E(0, 1) : \rho_{0,1} = e^{-x'}, x' \geq 0$$

Reverting the c.d.f.

$$X' = -\ln R, R \in \mathcal{U}(0, 1), \Rightarrow X = \lambda X' + \theta$$

Monolithic series method

1. Generate a sequence:  $U_1, U_2, \dots \in \mathbb{U}(0, 1)$
2. We look at series:  $U_1 \geq U_2 \geq U_3 \dots \geq U_n < U_{n+1}$ , which we then order with numbers:  $0, 1, 2, 3, \dots$
3. First series which length  $n$  is odd we take as integral part of a number. The decimal part is taken as  $R_1$ .

⇒ E7.1 Write the two above generators of  $E(0, 1)$ . Compare c.d.f. and p.d.f.

# Gaussian p.d.f.

⇒ The p.d.f.:

$$\phi_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad -\infty < x < \infty$$

⇒ Now we can always transform the variables:

$$x \rightarrow x' = (x - \mu)/\sigma \Rightarrow N(\mu, \sigma) \rightarrow N(0, 1)$$

⇒ First method of based on Central limit theorem. See Lecture 2.

Bad for the tails.

⇒ Reverting the c.d.f.

- In 1 dim the c.d.f. is not revertible :( One can use an approximation (Odeh, Evans 1974):

$$\Phi^{-1}(u) = \begin{cases} g(u), & 10^{-20} < u < 0.5 \\ -g(1-u) & 0.5 < u < 1 - 10^{-20} \end{cases}$$

$$g(u) = t - \frac{L(t)}{M(t)},$$

$$t = \sqrt{-2 \ln u}$$

$$L(t) = 0.322232431088 + t + 0.342242088547t^2 \\ + 0.0204231210245t^3 + 0.0000453642210148t^4$$

$$M(t) = 0.099348462606 + 0.588581570495t + 0.531103462366t^2 \\ + 0.10353775285t^3 + 0.0038560700634t^4$$

# Gaussian p.d.f.

⇒ Reverting the c.d.f. in 2 dim:

$$\phi(x, y) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2+y^2}{2}}, \quad -\infty < x, y < \infty$$

- We change the coordinates:  $(x, y) = (r \cos \phi, r \sin \phi)$
- And we factorize:  $\widehat{\rho}(\phi, r) = f(\phi)g(r)$ , where  $f(\phi) = \frac{1}{2\pi}$ ,  $g(r) = r e^{-\frac{r^2}{2}}$ .
- The angles is generated flat:  $\mathcal{U}(0, 2\pi)$  and the  $r$  with reverting the c.d.f..

⇒ If  $U_1, U_2 \in \mathcal{U}(0, 1)$ :

$$x = \sqrt{-2 \ln U_1} \cos(2\pi U_2)$$

$$y = \sqrt{-2 \ln U_1} \sin(2\pi U_2)$$

⇒ Accurate and simple to use.

⇒ Time consuming calculations of trigonometrical and logarithm function.

# Gaussian p.d.f.

⇒ The Marsaglia & Bray method (1964):

- If  $U_1, U_2 \in \mathcal{U}(-1, 1)$  are independent random variables, and  $U_1^2 + U_2^2 \leq 1$  then:

$$X_1 = U_1 \sqrt{\frac{-2 \ln(U_1^2 + U_2^2)}{U_1^2 + U_2^2}}, \quad Y_1 = X_1 \frac{U_2}{U_1}$$

have the distribution of  $N(0, 1)$ .

⇒ The algorithm:

- Generate  $R_1, R_2 \in \mathcal{U}(0, 1)$  and calculate the  $U_1 = 2R_1 - 1$ ,  $U_2 = 2R_2 - 1$
- Calculate  $W = U_1^2 + U_2^2$ .
- If  $W > 1$  start over.
- Calculate the  $X = U_1 Z$  and  $Y = U_2 Z$ , where  $Z = \sqrt{\frac{-2 \ln W}{W}}$

⇒ E7.2 Generate  $N(0, 1)$  using c.d.f. reverting and Marsaglia & Bray method.

# Breit-Wigner p.d.f.

⇒ The p.d.f.:

$$f_{\theta, \lambda}(x) = \frac{\lambda}{\pi} \frac{1}{\lambda^2 + (x - \theta)^2}, \quad -\infty < x < \infty$$

⇒ The variable transformation:

$$x \rightarrow x' \Rightarrow C(\theta, \lambda) \rightarrow C(0, 1)$$

⇒ The reverting c.d.f.:

- The c.d.f.

$$F(x) = \frac{1}{\pi} \arctan x + \frac{1}{2}$$
$$\Rightarrow X = \tan \left( \pi \left[ U - \frac{1}{2} \right] \right), \quad U \in \mathcal{U}(0, 1)$$

⇒ A statistical digression: There is no expected value of the Cauchy function. The variance is infinite.

## Breit-Wigner p.d.f.

⇒ One can use a cut-off Cauchy method  $C_u(0, 1)$ :

$$f_u(x) = \begin{cases} \frac{2}{\pi} \frac{1}{1+x^2}, & |x| \leq 1, \\ 0, & |x| > 1, \end{cases}$$

### Theorem:

If a random variable  $X$  has a cut-off Cauchy distribution  $C_u(0, 1)$ , then the new random variable  $Y$ , which is with 50 % equal  $X$  and with 50% equal  $1/X$  has a "normal" Cauchy distribution.

⇒ Prove ( $y \leq -1$ ):

$$\begin{aligned} \mathcal{P}\{Y \leq y\} &= \frac{1}{2} \mathcal{P}\{X \leq y\} + \frac{1}{2} \mathcal{P}\left\{\frac{1}{X} \leq y\right\} = 0 + \frac{1}{2} \mathcal{P}\left\{\frac{1}{y} \leq X < 0\right\} \\ &= \frac{1}{2} \frac{2}{\pi} \int_{1/y}^0 \frac{dr}{1+r^2} = \frac{1}{\pi} \arctan y + \frac{1}{2} \quad \text{c.d.f of } C(0, 1) \end{aligned}$$

⇒ The cut-off Breit-Wigner distribution we generate with elimination method using  $\mathcal{U}(-1, 1)$

⇒ E7.3 Generate the Breit-Wigner distribution with all described methods.



⇒ The p.d.f.:

$$f_1(x) = nx^{n-1}$$

$$f_2(x) = n(1-x)^{n-1}$$

where  $0 \leq x \leq 1$ ,  $n \in \mathbb{N}$  ⇒ Revert the c.d.f.:

$$X = U^{1/n} \longrightarrow f_1, \quad U \in \mathcal{U}(0, 1)$$

$$Y = 1 - U^{1/n} \longrightarrow f_2, \quad U \in \mathcal{U}(0, 1)$$

⇒ Disadvantage: The operation  $U^{1/n}$  is time consuming. ⇒ Second method:

- Generate  $U_1, U_2, \dots, U_n \in \mathcal{U}(0, 1)$ .
- $X = \max\{U_1, U_2, \dots, U_n\}$  has p.d.f. of  $f_1$ .
- $Y = \min\{U_1, U_2, \dots, U_n\}$  has p.d.f. of  $f_2$ .

⇒ E7.4 Generate the  $f_1$  and  $f_2$  p.d.f. with two methods.

# Bernoulli p.d.f.

⇒ The p.d.f.  $b(n, p)$  :

$$\mathcal{P}\{X = m\} = \binom{n}{m} p^m (1 - p)^{n-m}, \quad m = 0, 1, 2, \dots, n.$$

⇒ The interpretation: number of success with the probability  $p$ .

⇒ The algorithm (buffon needle):

```
m = 0;
for (i = 0; i < n; i++) {
    U = GenU(0, 1);
    if (U <= p) m++;
}
return m;
```

⇒ It requires many "trials"

# Bernoulli p.d.f.

⇒ If  $n$  is large, we can use a discrete p.d.f.:

$$p_k = \sum_{i=0}^k \mathcal{P}\{X = i\}$$

and use the algorithm:

```
m = 0;  
U = GenU(0, 1);  
while (U > p[m]) m++;  
return m;
```

## Theory hack:

If  $n$  is big one can write it in a form:  $n = kl$ , where  $l$  is NOT a big number. In this case one can generate  $k$  numbers from distribution  $b(l, p)$  and calculate  $m$  as sum of the generated numbers.

## Theory:

If  $U \in \mathcal{U}(0, 1)$  then:

$$Y = \Theta(p - U) \quad V = \min\left\{\frac{U}{p}, \frac{1 - U}{1 - p}\right\}$$

are independent and  $V \in \mathcal{U}(0, 1)$ .

⇒ This is super nice! We can treat  $Y$  as the indicator of success in the Bernoulli trials. And have a new random variable :)

```
m = 0;
U = GenU(0, 1);
for (i = 0; i < n; i++)
    if (U <= p) { m++; U /= p; }
    else U = (1 - U) / (1 - p);
return m;
```

⇒ E7.5 Please code the above mentioned Bernoulli p.d.f. generation.

## Poisson p.d.f.

⇒ The p.d.f.  $P(\lambda)$ :

$$\mathcal{P}(X = n) = \frac{\lambda^n}{n!} e^{-\lambda}, \quad n = 0, 1, 2, \dots$$

### Theory:

If  $\epsilon_1, \epsilon_2, \epsilon_3, \dots$  are from  $E(0, 1)$  then the random variable:

$$X = \min\{k : \sum_{i=0}^k \epsilon_i > \lambda\}$$

has the distribution of  $P(\lambda)$ .

⇒ The algorithm:

```
X = -1; S = 0;
while (S <= lambda) {
    Y = GenE(0,1);
    S += Y; X++;
}
return X;
```

# Poisson p.d.f.

⇒ The p.d.f.  $P(\lambda)$ :

$$P(X = n) = \frac{\lambda^n}{n!} e^{-\lambda}, \quad n = 0, 1, 2, \dots$$

## Theory:

If  $\epsilon_1, \epsilon_2, \epsilon_3, \dots$  are from  $E(0, 1)$  then the random variable:

$$X = \min\{k : \sum_{i=0}^k \epsilon_i > \lambda\}$$

has the distribution of  $P(\lambda)$ .

⇒ The algorithm 2:

```
X = -1; S = 1; q = exp(-lambda);  
while (S > q) {  
    U = GenU(0,1);  
    S *= U; X++;  
}  
return X;
```

# Poisson p.d.f.

⇒ Reverting the c.d.f.:

```
X = 0;
q = exp(-lambda);
S = P = q;
U = GenU(0,1);
while (U > S) {
    X++;
    P *= lambda/X;
    S += P;
}
return X;
```

⇒ It has problem with large values of  $\lambda$ , at you need many generations which causes numerical instabilities.

⇒ E7.6 Implement the abovementioned ways of generating  $P(\lambda)$ .

# Geometric p.d.f.

⇒ The p.d.f. of  $G(p)$  :

$$\mathcal{P}(X = n) = (1 - p)p^n, \quad n = 0, 1, 2, 3, \dots$$

## Theorem:

If a random variable has a p.d.f. of

$$f_\alpha(x) = \alpha e^{-\alpha x}$$

then  $\lfloor x \rfloor$  has a geometric p.d.f.:

$$G(e^{-\alpha})$$

⇒ Algorithm:

1. Generate a number  $U$  from  $\mathcal{U}(0, 1)$
  2. Calculate  $X = \lfloor \ln U / \ln p \rfloor$
- ⇒ E7.7 Implement the above algorithm.



# Equal division of interval

⇒ The method of equal division of an  $(0, 1)$  interval (the p.d.f.):

$$\mathcal{P}(X = k) = p_k, \quad k = 1, 2, 3, \dots, K$$

⇒ Some times the inverting the c.d.f. might be slow. This happens for large values of  $K$ .

⇒ A more efficient method:

- The interval  $(0, 1)$  we divide in  $K + 1$  bins:  $(\frac{i-1}{K+1}, \frac{i}{K+1})$ , which are equal size and we number them:  $1, 2, \dots, K + 1$ .
- The random variable  $U \in \mathcal{U}(0, 1)$  falls into bin  $\lfloor (K + 1)U \rfloor$ .
- We create a sequence:  $q_j = \sum_{k=0}^j p_k, j = 0, 1, \dots, K$ .
- And a companioning one:  $g_j = \max\{j : q_j < \frac{i}{K+1}\}, i = 0, 1, 2, \dots$

```
U = GenU(0, 1);  
X = g[(int) (K + 1)U + 1] + 1;  
while (q[X-1] > U) X--;  
return X;
```

# Multidimensional generation

- ⇒ Let  $\vec{X}$  be a  $m$  dimensional variable with a p.d.f. of  $f(x_1, x_2, x_3, \dots, x_m)$ .
- ⇒ To generate a p.d.f. like that we use the elimination method.
- ⇒ The problem with this is that for large dimensions we can have problems :(
- ⇒ Example:
  - Generate a flat p.d.f. on the hyper circle  $K_m(0, 1)$  with the accept reject method.
  - The probability of accepting event:

$$p_m = \pi^{m/2} / [2^m \Gamma(m/2 + 1)]$$

$m$	$p_m$	$N_m = 1/p_m$
2	$7.854 \cdot 10^{-1}$	1.27
5	$1.645 \cdot 10^{-1}$	6.08
10	$2.490 \cdot 10^{-3}$	$4.015 \cdot 10^2$
20	$2.461 \cdot 10^{-8}$	$4.063 \cdot 10^7$
50	$1.537 \cdot 10^{-28}$	$6.507 \cdot 10^{28}$

⇒ Good luck simulating  $10^{28}$  points ;)

# Multidimensional generation

⇒ Uniform distribution on a simplex:

**Theorem:**

If  $U_1, U_2, \dots, U_m \in \mathcal{U}(0, 1)$  and  $U_{1:m}, U_{2:m}, \dots, U_{m:m}$ . The a random variable:

$$X_1 = U_{1:m}, X_2 = U_{2:m} - U_{1:m}, \dots, X_m = U_{m:m} - U_{m-1:m}$$

has a uniform distribution on a simplex:

$$W_m = \{(x_1, x_2, \dots, x_m) : \sum_{j=1}^m x_j \leq 1, x_j \geq 0, j = 1, 2, \dots, m\}$$

# Multidimensional generation

⇒ Uniform distribution on a simplex surface:

## Theorem:

If  $U_1, U_2, \dots, U_{m-1} \in \mathcal{U}(0, 1)$  and  $U_{1:m-1}, U_{2:m-1}, \dots, U_{m-1:m-1}$ . The a random variable:

$$X_1 = U_{1:m-1}, X_{m-1} = U_{m-1:m-1} - U_{m-2:m-1}, X_m = 1 - U_{m-1:m-1}$$

has a uniform distribution on a simplex surface:

$$W_m = \{(x_1, x_2, \dots, x_m) : \sum_{j=1}^m x_j = 1, x_j \geq 0, j = 1, 2, \dots, m\}$$

# Backup