Simulation und Bildanalyse mit JAVA

Simulation von Punktprozessen

Konstantin Tjo

1 – Einführung zur Simulation
2 – Poisson-Prozess
3 – Matérn-Cluster-Prozess
4 – Hard-Core-Prozess

1 Einführung zur Simulation

1.1. Erzeugung von auf (0,1) gleichverteilten Pseudo-ZV'en

Algorithmus *(Multiplikativer Kongruenzgenerator)*

INPUT: $a \in \mathbb{N}, m \in \mathbb{N}$ Primzahl, $u_0 \in \mathbb{N}$ *Startwert*.
OUTPUT: Realisierung $\{x_1, \ldots, x_{m-1}\}$.

SIMUNIFORM (a, m, u_0)

1. $a \leftarrow 1000; \qquad m \leftarrow 2001179; \qquad u_0 \leftarrow 3000$
2. **for** $n \leftarrow 1$ **to** $m-1$ **do**
    3. $u_{n+1} \leftarrow au_n \, mod(m); \qquad x_n = u_n/m$
4. **end for**
5. *liefere* $\{x_1, \ldots, x_{m-1}\}$ *zurück.*

- Die Folge X_n ist periodisch mit Periode $m-1$, falls:
  - m ist eine Primzahl,
  - $p = \frac{m-1}{2}$ ist eine Primzahl,
  - $a^p \equiv -1 \quad mod(m)$;
- X_n's sind identisch verteilt mit Mittelwert $1/2$ und empirischen Varianz $\frac{(m-2)}{12m}$.

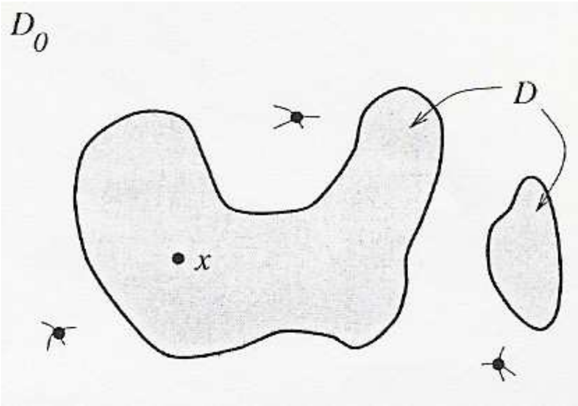
1.2. Erzeugung von "gleichverteilten" Punkten auf einer beschränkten Menge

- Ziel: $x \sim U(D)$
- Sei $D \subset \mathbb{R}^d$ beschränkt, $|D| > 0$,
- und sei $D_0 = \prod_{i=1}^{d} (a_i, a_i + l_i)$, $\qquad a_i, l_i \in \mathbb{R}, i \in \mathbb{N}$

Falls $U_1, \ldots, U_d$ unabhängig und gleichverteilt auf $(0,1)$ sind, so ist $(a_1 + l_1 U_1, \ldots, a_d + l_d U_d)$ gleichverteilt auf D_0.

Algorithmus

1. *Generiere* $x \sim U(D_0)$.
2. **if** $x \in D$ **then**
   3. *liefere* x *zurück.*
4. **else** *gehe zu 1.*



Sei N := Anzahl der Versuche bis zum Erfolg.
Dann gilt:

$$P(N = n) = \frac{|D|}{|D_0|} \left(1 - \frac{|D|}{|D_0|}\right)^{n-1}, \qquad n > 0$$

und

$$\mathbb{E}N = \frac{|D_0|}{|D|}.$$

1.3. Inversionsmethode

- Ziel: $X \overset{d}{\sim} F$ mit $F_X(x) = P(X < x)$
- Definiere $F^{-1}(u) = inf\{x \in \mathbb{R} \mid F(x) \geq u\}\,, \qquad u \in (0,1)$

Algorithmus *(Inversionsmethode)*

1. Generiere $U \sim U(0,1)$.
2. Liefere $F^{-1}(U)$ zurück.

Beweis:

Es gilt: $\quad F^{-1}(u) < x \quad \Leftrightarrow \quad u < F(x)$.
$\Rightarrow P(F^{-1}(U) < x) = P(U < F(x)) = F(x)$
$\Rightarrow F^{-1}(U) \overset{d}{\sim} F$.

Beispiel *(Exponentialverteilung)*

- Sei $X \sim \text{Exp(a)}$, d. h. $F(x) = 1 - exp(-ax)\,, \qquad a > 0.$
- $F(x) = u \qquad \Rightarrow \qquad x = \frac{-ln(1-u)}{a}.$
- Inversionsmethode $\Rightarrow F^{-1}(U) = \frac{-ln(1-U)}{a}$ bzw. $\frac{-lnU}{a}.$

1.4. Poisson-Verteilung

- $X \sim Poi(\lambda)$, falls $P(N = n) = exp(-\lambda)\frac{\lambda^n}{n!}$, $\lambda > 0, n \in \mathbb{N}_0$
- Die Anzahl unabhängiger exponentialverteilter Intervalle mit Parameter 1, die in $[0, \lambda]$ enthalten sind, ist Poisson-verteilt mit Parameter λ.
- Die Länge des i-ten Intervalls kann als $-lnU_i$ ausgedrückt werden.
- Der erzeugte Wert ist der kleinste Index n, der $-\sum_{i=1}^{n+1} lnU_i \geq \lambda$ oder äquivalent $\prod_{i=1}^{n+1} U_i \leq \exp(-\lambda)$ erfüllt.

1.4. Poisson-Verteilung

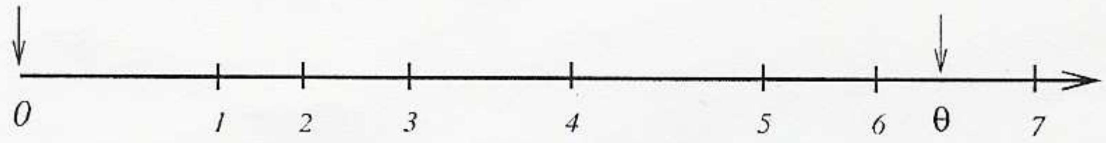


Fig. 7.5. Simulation of a Poisson variable by concatenating intervals with independent exponential lengths. In this example the value generated is 6.

Algorithmus *(Inversionsmethode)*

1. $N \leftarrow 0; \qquad T \leftarrow 1$

2. Generiere $U \sim U(0,1)$ **and** $T \leftarrow UT$*.*

3. **if** $T > e^{-\lambda}$ **then**

4. $N \leftarrow N + 1$ *und gehe zu 2.*

5. **else** *liefere* N *zurück.*

2 Poisson-Prozess

2.1. Homogener stationärer Poisson-Prozess

Definition

Ein Punktprozess X heisst ein homogener (stationärer) Poisson-Prozess mit Intensität $\lambda > 0$, falls folgende Bedingungen erfüllt sind:

- $X(B)$ ist Poisson-verteilt mit Parameter $\lambda|B|$ für alle beschränkte Borelmengen $B \in \mathcal{B}(\mathbb{R}^2)$,
- $X(B_1), \ldots, X(B_n)$ sind unabhängig für jede endliche Folge von paarweise disjunkten Borelmengen $B_1, \ldots, B_n \in \mathcal{B}(\mathbb{R}^2)$.

2.2. Simulation von homogenen Poisson-Prozessen

Algorithmus

INPUT: beschränkte Borelmenge $B \in \mathcal{B}(\mathbb{R}^2)$, Intensität $\lambda > 0$.

OUTPUT: Realisierung $\{x_1, x_2, \ldots\}$ des Poisson-Prozesses in B.

SIMPOISSONPOINT (B, λ)

1. *Generiere* $n \sim Poi(\lambda|B|)$.
2. **for** $i \leftarrow 1$ **to** n **do**
   3. *Generiere Punkt* $x_i \sim U(B)$.
4. **end for**
5. *liefere* $\{x_1, \ldots, x_n\}$ *zurück.*

2.3. Radiale Simulation von homogenen Poisson-Prozessen

Vorteil:

- Beobachtungsfenster hängt nicht von der Realisierung des Prozesses ab
- Punkte sind nach der Entfernung vom Kreismittelpunkt sortiert

Algorithmus

INPUT: Radius $0 < r_{max} < \infty$, Intensität $\lambda > 0$.

OUTPUT: Realisierung $\{x_1, x_2, \ldots\}$ des Poisson-Prozesses in $b(0, r_{max})$ mit $\|x_1\| \leq \|x_2\| \leq \ldots$.

SIMRADPOISSONPOINT (r, λ)

1. $sum \leftarrow 0; \qquad r \leftarrow 0; \qquad n \leftarrow 1$

2. **while** $r \leq r_{max}$ **do**

    3. Generiere $u \sim U(0,1)$*.*

    4. $sum \leftarrow sum + ln(u); \qquad r \leftarrow \sqrt{sum/(-\lambda\pi)}$

    5. **if** $r \leq r_{max}$ **then**

        6. Generiere $\theta \sim U[0, 2\pi)$*.*

        7. $x_n \leftarrow (r \cdot cos\theta, r \cdot sin\theta); \qquad n \leftarrow n+1$

    8. **end if**

9. **end while**

10. liefere $\{x_1, \ldots, x_{n-1}\}$ *zurück.*

3 Matérn-Cluster-Prozess

- Sei $X = \{X_1, X_2, \ldots\}$ ein homogener stationärer Poisson-Prozess mit Intensität $\lambda > 0$.
- Weiterhin sei $\{X^{(1)}, X^{(2)}, \ldots\}$ eine Folge von unabhängigen und identisch verteilten Punktprozessen $X^{(n)}$ mit endlicher erwarteten Anzahl der Punkte, d. h. $\mathbb{E}[X^{(1)}(\mathbb{R}^2)] < \infty$.
- Dann heisst der Punktprozess $Y = \{Y(B) : B \in \mathcal{B}(\mathbb{R}^2)\}$ mit $Y(B) = \sum_{n \geq 1} X^{(n)}(B - X_n)$ für alle $B \in \mathcal{B}(\mathbb{R}^2)$ Poisson-Cluster-Prozess.
- $r_0 = inf\{r > 0 : P(X^{(1)}(b(0,r)) = X^{(1)}(\mathbb{R}^2)) = 1\}$ heisst Clusterradius von Y.

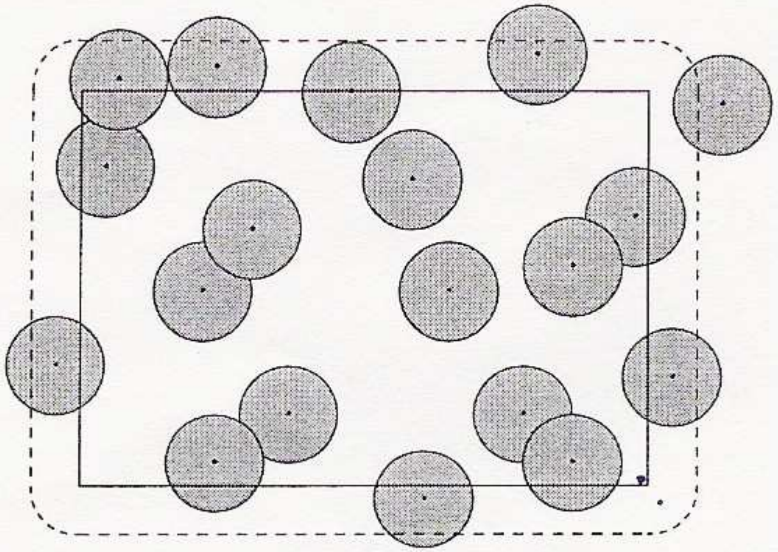


Figure 5.4: Illustration of the Plus Sampling

$$\tilde{B} = \delta_{b(0,r_0)}(B) = \bigcup_{x \in B} (b(0, r_0) + x)$$

3.2. Simulation von Matérn-Cluster-Prozessen

Algorithmus

INPUT: beschränkte Borelmenge $B \in \mathcal{B}(\mathbb{R}^2)$, Intensitäten $\lambda_0, \lambda_1 > 0$ und $r_0 > 0$.

OUTPUT: Realisierung $\{z_1, z_2, \ldots\}$ des Matérn-Cluster-Prozesses in B.

SIMMATÉRNCLUSTER $(B, \lambda_0, \lambda_1, r_0)$

1. $\tilde{B} \leftarrow \delta_{b(0,r_0)}(B)$

2. $\{x_1, \ldots, x_n\} \leftarrow$ *SIMPOISSONPOINT* $(\tilde{B}, \lambda_0)$

3. $k \leftarrow 1$

4. **for** $i \leftarrow 1$ **to** n **do**

   5. $\{y_{i,1}, \ldots, y_{i,m_i}\} \leftarrow$ *SIMRADPOISSONPOINT* (r_0, λ_1)

   6. **for** $j \leftarrow 1$ **to** m_i **do**

      7. **if** $x_i + y_{i,j} \in B$ **then**

         8. $z_k \leftarrow x_i + y_{i,j}; \qquad k \leftarrow k + 1$

      *9.***end if**

   10. **end for**

11. **end for**

12. *liefere* $\{z_1, \ldots, z_{k-1}\}$ *zurück.*

4 Hard-Core-Prozess

Definition

- Sei $X = \{X_1, X_2, \ldots\}$ ein homogener stationärer Poisson-Prozess,
- und sei $0 < r_0 < \infty$ der Hard-Core-Radius.
- Sei $M = \{M_1, M_2, \ldots\}$ eine Folge von unabhängigen und auf $(0,1)$ gleichverteilten Zufallsvariablen. Weiterhin seien X und M unabhängig voneinander.
- Der Punktprozess $Y = \{Y_1, Y_2, \ldots\}$ mit

$$Y = \{X_n : n \geq 1;\ M_n < M_m \text{ für alle } X_m \in b(X_n, r_0),\ m \neq n\}$$

heisst dann Matérn-Hard-Core-Prozess.

4.2. Simulation von Matérn-Hard-Core-Prozessen

Algorithmus

INPUT: beschränkte Borelmenge $B \in \mathcal{B}(\mathbb{R}^2)$, Intensität $\lambda > 0$ und $r_0 > 0$.

OUTPUT: Realisierung $\{y_1, y_2, \ldots\}$ des Matérn-Hard-Core-Prozesses im Beobachtungsfenster B.

SIMMATÉRNHARDCORE (B, λ, r_0)

1. $\tilde{B} \leftarrow \delta_{b(0,r_0)}(B)$
2. $\{x_1, \ldots, x_n\} \leftarrow$ *SIMPOISSONPOINT* $(\tilde{B}, \lambda_0)$
3. **for** $i \leftarrow 1$ **to** n **do**
    4. *Generiere* $m_i \sim U(0,1)$
5. **end for**
6. $k \leftarrow 1$
7. **for** $i \leftarrow 1$ **to** n
    8. **if** $x_i \in B$ **and**
    $\forall j \in \{1, \ldots, n\} : j = i \vee \|x_j - x_i\| > r_0 \vee m_j > m_i$ **then**
        9. $y_k \leftarrow x_i; k \leftarrow k+1$
    10. **end if**
11. **end for**
12. *liefere* $\{y_1, \ldots, y_{k-1}\}$ *zurück.*

Literatur:

C. Lantuéjoul *Geostatistical Simulation Models and Algorithms.* Springer, **2002**.

J. Mayer *On Quality Improvement of Scientific Software: Theory, Methods and Application in the GeoStoch Development.* Ph. D. Thesis, University of Ulm, **2003**.

J. Møller *Statistical Inference and Simulation for Spatial Point Processes.* Chapman and Hall/CRC, Boca Raton, **2003**.

D. Stoyan, H. Stoyan *Fractals, Random Shapes and Point Fields.* J. Wiley & Sons, **1994**.

D. Stoyan, W. S. Kendall, J. Mecke *Stochastic Geometry and its Applications.* J. Wiley & Sons, **1995**.