

Die Monte-Carlo-Methode zur Simulation von Teilchenreaktionen

Michael Rauch | 7. Mai 2010

HAUPTSEMINAR „EXPERIMENTELLE UND THEORETISCHE METHODEN DER COLLIDERPHYSIK“



Monte-Carlo-Methoden

Definition

Numerische Lösung von Problemen mit Hilfe von Zufallszahlen

- Monte-Carlo-Integration
Numerische Integration von mehrdimensionalen Integralen
- Monte-Carlo-Simulation
Simulation von Ereignissen, die mit gewisser Wahrscheinlichkeit auftreten

Ziel: Berechnung des Integrals

$$F(x) = \int_{a_1}^{b_1} dx_1 \int_{a_2}^{b_2} dx_2 \cdots \int_{a_d}^{b_d} dx_d f(x_1, x_2, \dots, x_d)$$

Der Einfachheit halber: d -dimensionaler Einheitswürfel

$$F(x) = \int_0^1 d^d u f(u_1, \dots, u_d)$$

- f überall innerhalb des Einheitswürfels bekannt
- f quadratintegrabel ($\rightarrow$ Konvergenz)
- keine analytische Lösung des (Teil-)Integrals bekannt

Methoden für 1-D Integrale (Rechteckregel, Trapezregel, ...) skalieren schlecht ($\mathcal{O}(N^{-2/d})$)

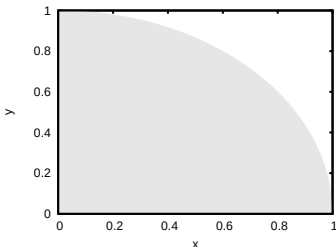
Idee:

Benutze zufällige Punkte statt Gitter

Beispiel: Bestimmung der Kreiszahl π

$$4 \int_0^1 dx \int_0^1 dy \Theta(1 - x^2 - y^2) = \pi$$

- Markiere Viertelkreis im Einheitsrechteck
- Werfe Dartpfeile auf das Rechteck
- Berechne Verhältnis Punkte innerhalb zu Punkte insgesamt
- → Entspricht Viertelkreisfläche zu Fläche gesamt

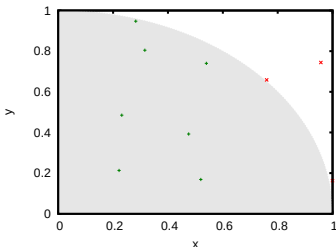


Punkte ges.	Punkte Kreis	Integral
10	7	2.8 ± 0.7
100	83	3.32 ± 0.15
1000	781	3.12 ± 0.05
10000	7862	3.145 ± 0.016

Beispiel: Bestimmung der Kreiszahl π

$$4 \int_0^1 dx \int_0^1 dy \Theta(1 - x^2 - y^2) = \pi$$

- Markiere Viertelkreis im Einheitsrechteck
- Werfe Dartpfeile auf das Rechteck
- Berechne Verhältnis Punkte innerhalb zu Punkte insgesamt
- → Entspricht Viertelkreisfläche zu Fläche gesamt

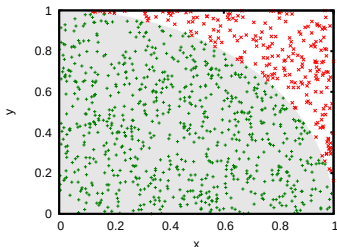


Punkte ges.	Punkte Kreis	Integral
10	7	2.8 ± 0.7
100	83	3.32 ± 0.15
1000	781	3.12 ± 0.05
10000	7862	3.145 ± 0.016

Beispiel: Bestimmung der Kreiszahl π

$$4 \int_0^1 dx \int_0^1 dy \Theta(1 - x^2 - y^2) = \pi$$

- Markiere Viertelkreis im Einheitsrechteck
- Werfe Dartpfeile auf das Rechteck
- Berechne Verhältnis Punkte innerhalb zu Punkte insgesamt
- → Entspricht Viertelkreisfläche zu Fläche gesamt



Punkte ges.	Punkte Kreis	Integral
10	7	2.8 $\pm$ 0.7
100	83	3.32 $\pm$ 0.15
1000	781	3.12 $\pm$ 0.05
10000	7862	3.145 $\pm$ 0.016

- Wahrscheinlichkeitsdichte für diskrete Zufallszahlen r_i
($p_i \geq 0$: Wahrscheinlichkeit für r_i)

$$P(x) = \sum_i p_i \delta(x - r_i) \quad \text{mit} \quad P(x) \geq 0 ; \quad \int dx P(x) = \sum_i p_i = 1$$

- Integral (Erwartungswert)

$$I = \langle f(x) \rangle = \int dx P(x) f(x) ; \quad I_N = \sum_i p_i f(r_i)$$

insbesondere für N gleichverteilte Zufallszahlen

Integral

$$I_N = \frac{1}{N} \sum_i f(r_i)$$

Integral

$$I_N = \frac{1}{N} \sum_i f(r_i)$$

■ Varianz

$$\begin{aligned}\sigma(f(x))^2 &= \langle f(x)^2 \rangle - \langle f(x) \rangle^2 \\ \sigma_N^2 &= \frac{1}{N} \sum_i (f(r_i)^2) - I^2 \simeq \frac{1}{N} \sum_i (f(r_i)^2) - I_N^2\end{aligned}$$

■ Integralfehler (aus zentralem Grenzwertsatz)

$$\Delta I_N = \sqrt{\frac{\sigma_N^2}{N}} \Rightarrow \text{Konvergenz} \propto \frac{1}{\sqrt{N}} \text{ (unabh. von Dimension)}$$

Fehler

$$\Delta I_N = \frac{1}{\sqrt{N}} \sqrt{\frac{1}{N} \sum_i f(r_i)^2 - I_N^2}$$

Integral

$$I_N = \frac{1}{N} \sum_i f(r_i)$$

Fehler

$$\Delta I_N = \frac{1}{\sqrt{N}} \sqrt{\frac{1}{N} \sum_i f(r_i)^2 - I_N^2}$$

- Zuverlässigkeit des Fehlers – aus Varianz der Varianz

$$\sigma\left(\sigma(f(x))\right)^2 = \langle f(x)^4 \rangle - 4 \langle f(x)^3 \rangle \langle f(x) \rangle - \langle f(x)^2 \rangle^2 + 8 \langle f(x)^2 \rangle \langle f(x) \rangle^2 - 4 \langle f(x) \rangle^4$$

Fehlergenauigkeit

$$\Delta(\Delta I_N) = \frac{1}{\sqrt{N^3}} \sqrt{\frac{1}{N} \sum_i f(r_i)^4 - 4 \frac{1}{N} \sum_i f(r_i)^3 I_N - (\Delta I_N)^2 + 8 \Delta I_N I_N^2 - 4 I_N^4}$$

Natürliche Zufallszahlen

- Würfeln, Münze werfen
- radioaktiver Zerfall
- Transistorrauschen
- ...

Nicht die beste Wahl:

- Zufälligkeit
Nachweis notwendig, dass aufeinanderfolgende Zufallszahlen tatsächlich unkorreliert sind
↔ Empirische Tests können nur bis zu einem gewissen Niveau testen
- Geschwindigkeit
Natürliche Prozesse normalerweise zu langsam (typische Zeitskala)
- Reproduzierbarkeit
Notwendig um Programm debuggen zu können

⇒ Erzeuge Zufallszahlen mit Algorithmus

→ Pseudo-Zufallszahlen

Gewünschte Eigenschaften:

- Gute „zufällige“ Verteilung
- Reproduzierbarkeit
- Effizienz
- Lange Periode
- Lange unabhängige Untersequenzen

Algorithmen (z.B. in ROOT)

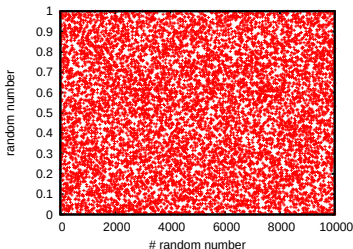
- TRandom – Linear-Kongruenzgenerator

$$y_i = (ay_{i-1} + b) \bmod m \quad (a, b, m \text{ Konstanten, } y_1 \text{ Startwert})$$

- schnell
- kurze Periode $\simeq 10^9$
- niedrige Bits korreliert

$$\text{Beispiel: } I = 2 \int_0^1 dx \int_0^1 dy \int_0^1 dz \sin^2(2\pi(9x - 6y + z)) = 1$$

$$(x, y, z)_n = (y_{3n}/m, y_{3n+1}/m, y_{3n+2}/m)$$



$$a = 65539 \quad b = 0 \quad m = 2^{31}$$

(IBM 1960er Jahre)

- TRandom1 – RANLUX (RCARRY) Algorithmus
- TRandom3 – Mersenne-Twister Algorithmus

Algorithmen (z.B. in ROOT)

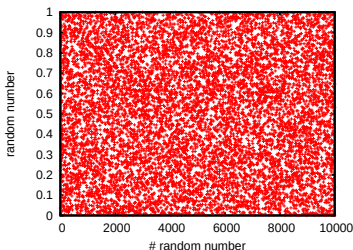
- TRandom – Linear-Kongruenzgenerator

$$y_i = (ay_{i-1} + b) \bmod m \quad (a, b, m \text{ Konstanten, } y_1 \text{ Startwert})$$

- schnell
- kurze Periode $\simeq 10^9$
- niedrige Bits korreliert

$$\text{Beispiel: } I = 2 \int_0^1 dx \int_0^1 dy \int_0^1 dz \sin^2(2\pi(9x - 6y + z)) = 1$$

$$(x, y, z)_n = (y_{3n}/m, y_{3n+1}/m, y_{3n+2}/m)$$



$$a = 65539 \quad b = 0 \quad m = 2^{31}$$

(IBM 1960er Jahre)

$$I = 1.66 \cdot 10^{-30} \pm 0.02 \cdot 10^{-30}$$

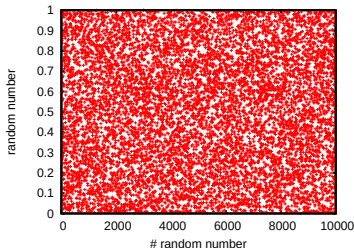
- TRandom1 – RANLUX (RCARRY) Algorithmus
- TRandom3 – Mersenne-Twister Algorithmus

Algorithmen (z.B. in ROOT)

- TRandom – Linear-Kongruenzgenerator
 - $y_i = (ay_{i-1} + b) \bmod m$ (a, b, m Konstanten, y_1 Startwert)
 - schnell
 - kurze Periode $\simeq 10^9$
 - niedrige Bits korreliert

$$\text{Beispiel: } I = 2 \int_0^1 dx \int_0^1 dy \int_0^1 dz \sin^2(2\pi(9x - 6y + z)) = 1$$

$$(x, y, z)_n = (y_{3n}/m, y_{3n+1}/m, y_{3n+2}/m)$$



$$a = 65539 \quad b = 0 \quad m = 2^{31}$$

(IBM 1960er Jahre)

$$I = 1.66 \cdot 10^{-30} \pm 0.02 \cdot 10^{-30}$$

Problem:

$$(9y_{3n} - 6y_{3n+1} + y_{3n+2}) \bmod 2^{31} = 0$$

- TRandom1 – RANLUX (RCARRY) Algorithmus
- TRandom3 – Mersenne-Twister Algorithmus

Algorithmen (z.B. in ROOT)

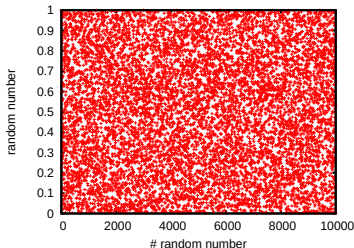
- TRandom – Linear-Kongruenzgenerator

$$y_i = (ay_{i-1} + b) \bmod m \quad (a, b, m \text{ Konstanten, } y_1 \text{ Startwert})$$

- schnell
- kurze Periode $\simeq 10^9$
- niedrige Bits korreliert

$$\text{Beispiel: } I = 2 \int_0^1 dx \int_0^1 dy \int_0^1 dz \sin^2(2\pi(9x - 6y + z)) = 1$$

$$(x, y, z)_n = (y_{3n}/m, y_{3n+1}/m, y_{3n+2}/m)$$



$$a = 65539 \quad b = 0 \quad m = 2^{31}$$

(IBM 1960er Jahre)

$$I = 1.66 \cdot 10^{-30} \pm 0.02 \cdot 10^{-30}$$

Problem:

$$(9y_{3n} - 6y_{3n+1} + y_{3n+2}) \bmod 2^{31} = 0$$

$$a = 69069$$

$$I = 1.008 \pm 0.007$$

- TRandom1 – RANLUX (RCARRY) Algorithmus
- TRandom3 – Mersenne-Twister Algorithmus

Algorithmen (z.B. in ROOT)

- TRandom – Linear-Kongruenzgenerator
- TRandom1 – RANLUX (RCARRY) Algorithmus

$$\Delta n = y_{n-10} - y_{n-24} - c_{n-1}$$

$$y_n = \Delta n, \quad c_n = 0 \quad \text{falls } \Delta n \geq 0$$

$$y_n = \Delta n + 2^{24}, \quad c_n = 1 \quad \text{sonst}$$

Nach jeweils 24 Zahlen verwerfe nächste 199.

- Zufallseigenschaften mathematisch bewiesen
 - lange Periode $\simeq 10^{171}$
 - langsam
- TRandom3 – Mersenne-Twister Algorithmus

Algorithmen (z.B. in ROOT)

- TRandom – Linear-Kongruenzgenerator
- TRandom1 – RANLUX (RCARRY) Algorithmus
- TRandom3 – Mersenne-Twister Algorithmus

$s_1 \dots s_{624}$ Startwerte

$$h = s_{i-624} - s_{i-624} \bmod 2^{31} + s_{i-623} \bmod 2^{31}$$

$$s_i = s_{i-227} \oplus \lfloor h/2 \rfloor \oplus ((h \bmod 2) \cdot 9908b0df_{16})$$

$$a = s_i \oplus \lfloor s_i/2^{11} \rfloor$$

$$b = a \oplus ((a \cdot 2^7) \wedge 9d2c5680_{16})$$

$$c = b \oplus ((b \cdot 2^{15}) \wedge efc60000_{16})$$

$$y_i = c \oplus \lfloor c/2^{18} \rfloor$$

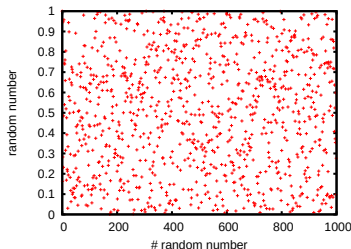
$$\oplus \text{ binäres XOR} \quad \wedge \text{ binäres UND}$$

- schnell
- lange Periode $2^{19937} - 1 \simeq 10^{6000}$

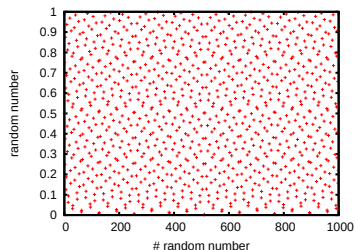
- Für Monte-Carlo-Integration nicht alle Eigenschaften von Zufallszahlen notwendig
- Hohe Varianz eher hinderlich
(Fehler $\sim 1/\sqrt{N}$ vs. $1/N^2$ für z.B. eindimensionale Rechteckregel)
- $\Rightarrow$ Aufgabe der Zufälligkeit um hohe Gleichmäßigkeit zu erreichen
- $\rightarrow$ Fehler $\sim 1/N \ln^p(N)$

Beispiel: 1000 Zufallszahlen (1-dim) (Implementation: GNU Scientific Library)

Mersenne-Twister Pseudozufallszahlen



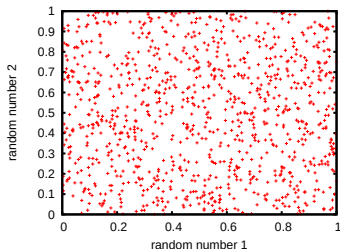
Sobol Quasizufallszahlen



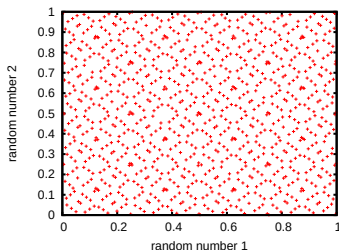
- Für Monte-Carlo-Integration nicht alle Eigenschaften von Zufallszahlen notwendig
- Hohe Varianz eher hinderlich
(Fehler $\sim 1/\sqrt{N}$ vs. $1/N^2$ für z.B. eindimensionale Rechteckregel)
- $\Rightarrow$ Aufgabe der Zufälligkeit um hohe Gleichmäßigkeit zu erreichen
- $\rightarrow$ Fehler $\sim 1/N \ln^p(N)$

Beispiel: 1000 Zufallszahlen (2-dim) (Implementation: GNU Scientific Library)

Mersenne-Twister Pseudozufallszahlen



Sobol Quasizufallszahlen



Algorithmus:

- Wähle Polynom $P = x^g + a_1x^{g-1} + \dots + a_{g-1}x + 1$ mit $a_i = 0$ oder 1
- Berechne $v_i = a_1v_{i-1} \oplus a_2v_{i-2} \oplus \dots \oplus a_{g-1}v_{i-(g-1)} \oplus v_{i-g} \oplus [v_{i-g}/2^g]$
mit $\oplus =$ bitweises XOR
- $x_{n+1} = x_n \oplus v_c$ wobei c die Stelle der am weitesten rechts gelegenen 0
in der Binärdarstellung von n ist

Erste Zufallszahlen

1-D	2-D	
0.5	0.5	0.5
0.75	0.75	0.25
0.25	0.25	0.75
0.375	0.375	0.375
0.875	0.875	0.875
0.625	0.625	0.125
0.125	0.125	0.625
0.1875	0.1875	0.3125
0.6875	0.6875	0.8125
0.9375	0.9375	0.0625

Zufallszahlengeneratoren erzeugen typischerweise gleichmäßige Verteilungen $x \in [0; 1)$

Oft Zufallszahlenverteilung benötigt, die Funktion $g(x)$ folgt

Möglichkeiten:

- Transformationsmethode
- (von-Neumann-)Rückweisungsmethode

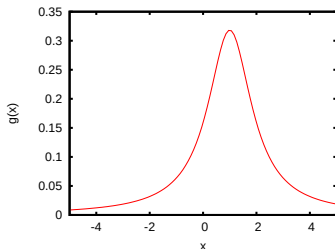
$g(x)$: Funktion für gewünschte Zufallszahlenverteilung (normiert)
 $[x_0; x_1]$: Intervall für Zufallszahlen
 u : uniforme Zufallszahlen in $[0; 1]$

Voraussetzung:

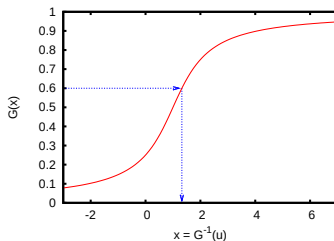
- Kumulative Verteilungsfunktion berechenbar:
 $G(x) = \int_{x_0}^x dt g(t)$
- Kumulative Verteilungsfunktion invertierbar

⇒ Zufallszahlen: $x = G^{-1}(u)$

$$g(x) = \frac{1}{\pi} \frac{1}{(x-1)^2+1}$$



$$G(x) = \frac{1}{\pi} \arctan(x - 1) + 0.5$$



■ Exponentieller Zerfall

$$g(x) = \frac{1}{\tau} \exp\left(-\frac{x}{\tau}\right)$$

$$\Rightarrow x = G^{-1}(u) = -\tau \cdot \ln(1 - u)$$

■ Breit-Wigner-Verteilung (Teilchenresonanz)

$$g(p^2) = \frac{M\Gamma}{\pi} \frac{1}{(p^2 - M^2)^2 + M^2\Gamma^2}$$

$$\Rightarrow p^2 = M\Gamma \tan(\pi u) + M^2$$

■ Gauss-Verteilung direkt:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

$$\Rightarrow x = \mu + \sqrt{2\sigma^2} \operatorname{erf}^{-1}(2u - 1)$$

■ Gauss-Verteilung

mit Trick (Box-Müller-Algorithmus):

Idee: Berechne zwei Gauss-Transformationen gleichzeitig

$$\begin{aligned} 1 &= \frac{1}{2\pi} \left(\int dx \exp\left(-\frac{x^2}{2}\right) \right)^2 \\ &= \frac{1}{2\pi} \int dx dy \exp\left(-\frac{(x^2 + y^2)}{2}\right) \\ &= \int_0^{2\pi} \frac{d\phi}{2\pi} \int_0^\infty ds s \exp\left(-\frac{s^2}{2}\right) \\ &= \int_0^{2\pi} \frac{d\phi}{2\pi} \int_0^\infty dt \exp\left(-\frac{t}{2}\right) \end{aligned}$$

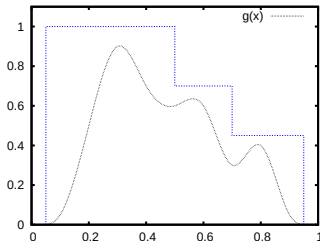
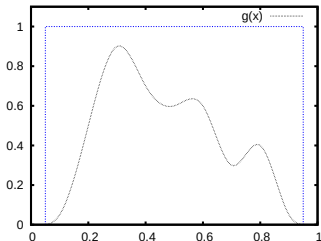
Generiere

$$t \leftarrow -\ln(u_1) \quad \phi \leftarrow 2\pi u_2$$

$$x_1 \leftarrow \sqrt{t} \cos(\phi) \quad x_2 \leftarrow \sqrt{t} \sin(\phi)$$

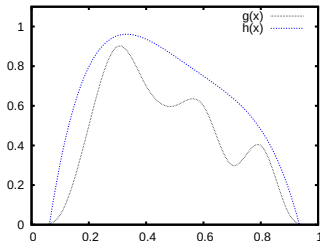
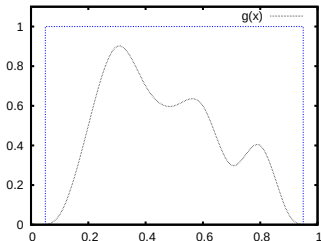
Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$
(oft Rechteckverteilung $h(x) = \text{const.}$)



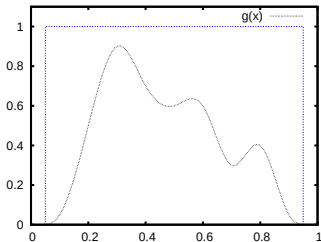
Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$
(oft Rechteckverteilung $h(x) = \text{const.}$)



Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$ (oft Rechteckverteilung $h(x) = \text{const.}$)

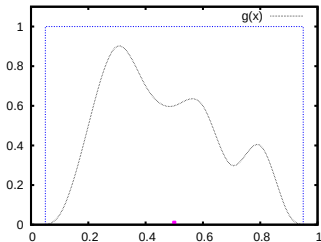


Algorithmus:

- erzeuge x verteilt nach $h(x)$
- erzeuge gleichverteilte Zufallszahl ϱ
- falls $\varrho < g(x)$ akzeptiere x
- ansonsten starte erneut mit neuem x

Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$ (oft Rechteckverteilung $h(x) = \text{const.}$)



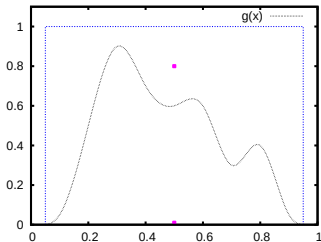
$x = 0.5$

Algorithmus:

- erzeuge x verteilt nach $h(x)$
- erzeuge gleichverteilte Zufallszahl ϱ
- falls $\varrho < g(x)$ akzeptiere x
- ansonsten starte erneut mit neuem x

Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$ (oft Rechteckverteilung $h(x) = \text{const.}$)



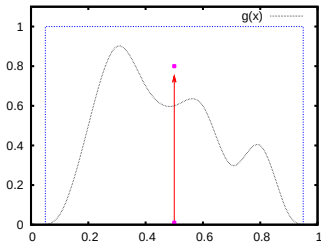
$x = 0.5$ $\varrho = 0.8$

Algorithmus:

- erzeuge x verteilt nach $h(x)$
- erzeuge gleichverteilte Zufallszahl ϱ
- falls $\varrho < g(x)$ akzeptiere x
- ansonsten starte erneut mit neuem x

Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$ (oft Rechteckverteilung $h(x) = \text{const.}$)



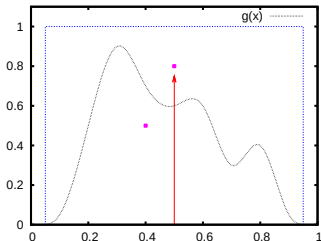
$x = 0.5$ $q = 0.8$

Algorithmus:

- erzeuge x verteilt nach $h(x)$
- erzeuge gleichverteilte Zufallszahl q
- falls $q < g(x)$ akzeptiere x
- ansonsten starte erneut mit neuem x

Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$ (oft Rechteckverteilung $h(x) = \text{const.}$)



$x = 0.5 \quad \varrho = 0.8 \Rightarrow \text{verwerfen}$

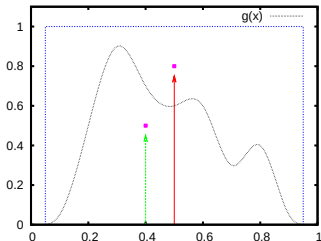
$x = 0.4 \quad \varrho = 0.5$

Algorithmus:

- erzeuge x verteilt nach $h(x)$
- erzeuge gleichverteilte Zufallszahl ϱ
- falls $\varrho < g(x)$ akzeptiere x
- ansonsten starte erneut mit neuem x

Voraussetzung:

- $g(x)$ berechenbar
- $g(x)$ nach oben beschränkt durch einfache (normierte) Verteilung $h(x)$ (oft Rechteckverteilung $h(x) = \text{const.}$)



Algorithmus:

- erzeuge x verteilt nach $h(x)$
- erzeuge gleichverteilte Zufallszahl ϱ
- falls $\varrho < g(x)$ akzeptiere x
- ansonsten starte erneut mit neuem x

$x = 0.5 \quad \varrho = 0.8 \Rightarrow \text{verwerfen}$
 $x = 0.4 \quad \varrho = 0.5 \Rightarrow \text{akzeptieren}$
 $\Rightarrow x = 0.4$

Nützlich, falls genaue Form von $g(x)$ nicht bekannt
Schlechte Wahl von $h(x)$ benötigt viele Zufallszahlen
(Auswertung von $g(x)$ kann viel Rechenleistung erfordern)

Fehler bestimmt durch Varianz

⇒ Kleinere Varianz → kleinerer Fehler

2 Methoden:

- Stratified Sampling (Geschichtetes Zufallsstichprobenverfahren)
hohe Punktedichte in Bereichen hoher Varianz
- Importance Sampling
Finde analytisch berechenbare Näherung $p(x)$ für Integranden

Stratified Sampling

Basierend auf $\int_0^1 dx f(x) = \int_0^a dx f(x) + \int_a^1 dx f(x)$, $0 < a < 1$

Teile Integrationsgebiet in k Unterregionen M_j

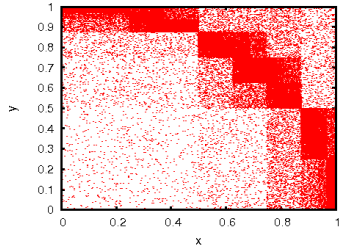
$$I_N = \sum_{j=1}^k \frac{\text{vol}(M_j)}{N_j} \sum_{n=1}^{N_j} f(x_{jn}) \quad \Delta I_N^2 = \sum_{j=1}^k \frac{\text{vol}(M_j)^2}{N_j} \sigma^2(f)|_{M_j}$$

→ für $N_j \propto \sigma(f)|_{M_j}$ minimal

Beispiel (π):

- Berechne Integral und Fehler für 100 Zufallszahlen
- Falls Fehler klein, gib Ergebnis zurück
- Für jede Dimension, teile Integrationsbereich in 2 Hälften und berechne Teilintegral; wähle Dimension mit größtem Teilintegral
- Wiederhole Prozedur für beide Hälften in dieser Dimension

$$I_{\text{strat}} = 3.1422 \pm 0.0003$$
$$I_{\text{std}} = 3.1393 \pm 0.0046$$



Problem: Bereiche mit hoher Varianz, aber kleinem Beitrag zum Integral

Mathematisch: Wechsel der Integrationsvariablen

$$\int dx f(x) = \int \frac{f(x)}{p(x)} p(x) dx = \int \frac{f(x)}{p(x)} dP(x)$$

mit Wahrscheinlichkeitsdichte $p(x) = \frac{\partial^d}{\partial x_1 \dots \partial x_d} P(x) \geq 0 \quad \int dx p(x) = 1$

$$I = \frac{1}{N} \sum_{i=1}^N \frac{f(x_n)}{p(x_n)}$$

$$\Delta I^2 \left(\frac{f}{p} \right) = \frac{1}{N} \left(\frac{1}{N} \sum_{i=1}^N \left(\frac{f(x_n)}{p(x_n)} \right)^2 - I^2 \right)$$

Relevante Größe jetzt $\frac{f(x)}{p(x)}$

⇒ Wähle $p(x)$ so nah wie möglich an $f(x)$

Problem: Falls $p(x) \ll f(x) \rightarrow$ größere Fehler

Oft benutzter Algorithmus: VEGAS (Lepage 1978)

- Ansatz: $p(x_1, \dots, x_d) = p_1(x_1) \cdots p_d(x_d)$
(entspricht Projektion auf die Koordinatenachsen)
- Teile jede Integrationsdimension in K Bereiche
- Wähle Bereichsgrenzen $[b_i, b_{i+1}]$ so, dass $(b_{i+1} - b_i)/l_i = \text{const.}$
(Start: äquidistant)
 $\Rightarrow p_i = b_{i+1} - b_i$
- Integriere $f(x)/p(x)$
- Wiederhole bis gewünschte Genauigkeit erreicht

■ Monte-Carlo-Ereignis-Generatoren

Pythia, Herwig, Sherpa, VBFNLO, MCFM, MC@NLO, ...

- Wirkungsquerschnitt $\rightarrow$ Monte-Carlo-Integration

$$\sigma = \int d\text{PS} \frac{1}{\text{Fluss}} |\mathcal{M}_{fi}|^2 \cdot \Theta_{\text{Schritte}}(f(\text{PS})) \quad (\text{PS} = \text{Phasenraum})$$

- ungewichtete Ereignisse
Gewichtete Ereignisse aus Monte-Carlo-Integration
Entwicklung durch Rückweisungsmethode
- Partonschauer / Hadronisierung
Simulation Elementarteilchen $\rightarrow$ Hadronen (Zeitskala $\lesssim 1 \text{ fm} / c$)

■ Detektorsimulation

z.B. Geant4

- Monte-Carlo-Ereignis-Generatoren

Pythia, Herwig, Sherpa, VBFNLO, MCFM, MC@NLO, ...

- Wirkungsquerschnitt $\rightarrow$ Monte-Carlo-Integration

$$\sigma = \int dPS \frac{1}{\text{Fluss}} |\mathcal{M}_{fi}|^2 \cdot \Theta_{\text{Schnitte}}(f(PS)) \quad (PS = \text{Phasenraum})$$

- ungewichtete Ereignisse
Gewichtete Ereignisse aus Monte-Carlo-Integration
Entwichtung durch Rückweisungsmethode
- Partonschauer / Hadronisierung
Simulation Elementarteilchen $\rightarrow$ Hadronen (Zeitskala $\lesssim 1 \text{ fm} / c$)

- Detektorsimulation

z.B. Geant4

■ Monte-Carlo-Ereignis-Generatoren

Pythia, Herwig, Sherpa, VBFNLO, MCFM, MC@NLO, ...

- Wirkungsquerschnitt $\rightarrow$ Monte-Carlo-Integration

$$\sigma = \int d\text{PS} \frac{1}{\text{Fluss}} |\mathcal{M}_f|^2 \cdot \Theta_{\text{Schnitte}}(f(\text{PS})) \quad (\text{PS} = \text{Phasenraum})$$

Importance Sampling

- Peakstruktur des Integranden ungefähr bekannt
 $\rightarrow$ Variablentransformationen aus Feynman-Diagrammen
z.B. Produktion von (On-shell-)Teilchen $\rightarrow$ Breit-Wigner-Peak
Spinstruktur $\rightarrow$ Winkelabhängigkeiten
- Integration mit VEGAS
- ungewichtete Ereignisse
Gewichtete Ereignisse aus Monte-Carlo-Integration
Entwichtung durch Rückweisungsmethode
- Partonschauer / Hadronisierung
Simulation Elementarteilchen $\rightarrow$ Hadronen (Zeitskala $\lesssim 1 \text{ fm} / c$)

■ Detektorsimulation

z.B. Geant4

■ Monte-Carlo-Ereignis-Generatoren

Pythia, Herwig, Sherpa, VBFNLO, MCFM, MC@NLO, ...

- Wirkungsquerschnitt $\rightarrow$ Monte-Carlo-Integration

$$\sigma = \int d\text{PS} \frac{1}{\text{Fluss}} |\mathcal{M}_f|^2 \cdot \Theta_{\text{Schnitte}}(f(\text{PS})) \quad (\text{PS} = \text{Phasenraum})$$

Importance Sampling

- Peakstruktur des Integranden ungefähr bekannt
 $\rightarrow$ Variablentransformationen aus Feynman-Diagrammen
z.B. Produktion von (On-shell-)Teilchen $\rightarrow$ Breit-Wigner-Peak
Spinstruktur $\rightarrow$ Winkelabhängigkeiten
- Integration mit VEGAS
- ungewichtete Ereignisse
Gewichtete Ereignisse aus Monte-Carlo-Integration
Entwicklung durch Rückweisungsmethode
- Partonschauer / Hadronisierung
Simulation Elementarteilchen $\rightarrow$ Hadronen (Zeitskala $\lesssim 1 \text{ fm} / c$)

■ Detektorsimulation

z.B. Geant4

■ Monte-Carlo-Ereignis-Generatoren

Pythia, Herwig, Sherpa, VBFNLO, MCFM, MC@NLO, ...

- Wirkungsquerschnitt $\rightarrow$ Monte-Carlo-Integration

$$\sigma = \int dPS \frac{1}{\text{Fluss}} |\mathcal{M}_f|^2 \cdot \Theta_{\text{Schnitte}}(f(PS)) \quad (PS = \text{Phasenraum})$$

Importance Sampling

- Peakstruktur des Integranden ungefähr bekannt
 $\rightarrow$ Variablentransformationen aus Feynman-Diagrammen
z.B. Produktion von (On-shell-)Teilchen $\rightarrow$ Breit-Wigner-Peak
Spinstruktur $\rightarrow$ Winkelabhängigkeiten
- Integration mit VEGAS
- ungewichtete Ereignisse
Gewichtete Ereignisse aus Monte-Carlo-Integration
Entwicklung durch Rückweisungsmethode
- Partonschauer / Hadronisierung
Simulation Elementarteilchen $\rightarrow$ Hadronen (Zeitskala $\lesssim 1 \text{ fm} / c$)

■ Detektorsimulation

z.B. Geant4

■ Monte-Carlo-Ereignis-Generatoren

Pythia, Herwig, Sherpa, VBFNLO, MCFM, MC@NLO, ...

- Wirkungsquerschnitt $\rightarrow$ Monte-Carlo-Integration

$$\sigma = \int d\text{PS} \frac{1}{\text{Fluss}} |\mathcal{M}_f|^2 \cdot \Theta_{\text{Schnitte}}(f(\text{PS})) \quad (\text{PS} = \text{Phasenraum})$$

Importance Sampling

- Peakstruktur des Integranden ungefähr bekannt
 - $\rightarrow$ Variablentransformationen aus Feynman-Diagrammen
 - z.B. Produktion von (On-shell-)Teilchen $\rightarrow$ Breit-Wigner-Peak
 - Spinstruktur $\rightarrow$ Winkelabhängigkeiten
- Integration mit VEGAS
- ungewichtete Ereignisse
 - Gewichtete Ereignisse aus Monte-Carlo-Integration
 - Entwicklung durch Rückweisungsmethode
- Partonschauer / Hadronisierung
 - Simulation Elementarteilchen $\rightarrow$ Hadronen (Zeitskala $\lesssim 1 \text{ fm} / c$)

■ Detektorsimulation

z.B. Geant4

- Monte-Carlo-Methoden wichtig in der Teilchenphysik
 - Berechnen von Wirkungsquerschnitten
 - Erzeugen von Ereignissen
 - Detektorsimulation
- Ungefähre numerische Lösung von Problemen mit Hilfe von Zufallszahlen in endlicher Zeit
- Verschiedene Methoden
 - Wahl hängt vom konkreten Problem ab

- R. Kleiss
Monte Carlo Integration
<http://www.hef.kun.nl/~kleiss/mcnotes.pdf>
- Particle Data Group
Review of Particle Physics:
Chapter 33. Monte Carlo Techniques
<http://pdg.lbl.gov/2009/reviews/rpp2009-rev-monte-carlo-techniques.pdf>
- S. Weinzierl
Introduction to Monte Carlo Methods
[arXiv:hep-ph/0006269]
<http://arxiv.org/pdf/hep-ph/0006269v1>
- Implementationen
 - *GNU Scientific Library*
<http://www.gnu.org/software/gsl/>
 - *ROOT*
<http://root.cern.ch/>