

Moderne Methoden in der Auswertung von Experimenten

Alexander Mann

a.mann@lmu.de

(mit Material von Johannes Elmsheuser, Günter Dückeck)



Blockkurs für Bachelor-Studenten

26. April 2017

München

- 1 Monte-Carlo-Methode
- 2 Zufallszahlen
- 3 Beliebig verteilte Zufallszahlen
- 4 Monte-Carlo-Ereignis-Generatoren
- 5 Zusammenfassung

- 1 Monte-Carlo-Methode
- Zufallszahlen
- Beliebig verteilte Zufallszahlen
- Monte-Carlo-Ereignis-Generatoren
- Zusammenfassung

Ein klassisches Beispiel (I)

- Historisches Beispiel zur Berechnung der Zahl π : Buffons Nadel
(Graf G.L.L. von Buffon, 1707 – 1788)
- N Nadeln der Länge l werden auf Fläche mit äquidistanten, parallelen Geraden geworfen (Abstand $d \geq l$).

- Wahrscheinlichkeit für “Geradentreffer” einer einzigen Nadel:

$$p = l_{\text{eff}}/d = l |\cos \varphi|/d$$

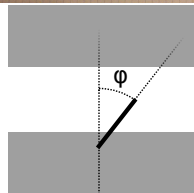
- Integration über gleichverteilte φ :

$$p = \int_0^{2\pi} \frac{l |\cos \varphi|}{d} \frac{d\varphi}{2\pi} = \frac{2l}{\pi d}$$

- Für große N : $N_{\text{Treffer}}/N \rightarrow p$

$$\Rightarrow \frac{2 \cdot N \cdot l}{N_{\text{Treffer}} \cdot d} \rightarrow \pi$$

- **Übung:** Was kommt für π raus (oben rechts)?



Ein klassisches Beispiel (I)

- Historisches Beispiel zur Berechnung der Zahl π : Buffons Nadel
(Graf G.L.L. von Buffon, 1707 – 1788)
- N Nadeln der Länge l werden auf Fläche mit äquidistanten, parallelen Geraden geworfen (Abstand $d \geq l$).

- Wahrscheinlichkeit für “Geradentreffer” einer einzigen Nadel:

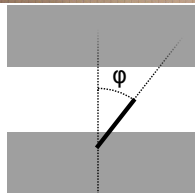
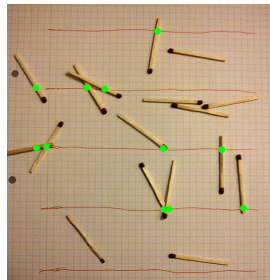
$$p = l_{\text{eff}}/d = l |\cos \varphi|/d$$

- Integration über gleichverteilte φ :

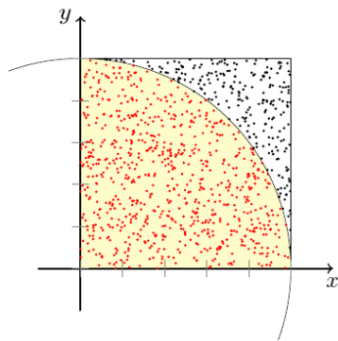
$$p = \int_0^{2\pi} \frac{l |\cos \varphi|}{d} \frac{d\varphi}{2\pi} = \frac{2l}{\pi d}$$

- Für große N : $N_{\text{Treffer}}/N \rightarrow p$
 $\Rightarrow \frac{2 \cdot N \cdot l}{N_{\text{Treffer}} \cdot d} \rightarrow \pi$

- **Übung:** Was kommt für π raus (oben rechts)? $2 \cdot 17/11 \approx 3,1$ ($l \approx d$)



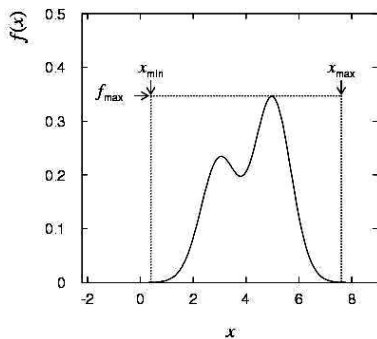
Ein klassisches Beispiel (II)



Näherung der Kreiszahl Pi über das Verhältnis von **“Treffern”** t und der Gesamtzahl der **“Versuche”** n ,

$$\lim_{n \rightarrow \infty} t/n = \pi$$

Integration einer Wahrscheinlichkeitsdichtefunktion *pdf*



Wie im vorhergehenden Beispiel: simuliere die Anzahl der **“Treffer”** und **“Nicht-Treffer”**

Definition “Monte-Carlo-Methode”

Eine numerische Methode zur Berechnung von Wahrscheinlichkeiten und abgeleiteten Größen unter Benutzung von Zufallszahlen

Nutzung

- Verfahren aus der Stochastik
- Anwendung in vielen Bereichen (statistische Physik, Biophysik, Teilchenphysik, Versicherungen, ...)
- löse analytisch nicht oder nur aufwendig lösbare Probleme numerisch
 - auch nützlich nach Bestimmung der analytischen Lösung als unabhängige Gegenprobe
- Basis:
 - sehr häufig durchgeführte Zufallsexperimente
 - Gesetz der großen Zahlen

Definition “Monte-Carlo-Methode”

Eine numerische Methode zur Berechnung von Wahrscheinlichkeiten und abgeleiteten Größen unter Benutzung von Zufallszahlen

Umsetzung

- Erzeuge eine Sequenz von gleichförmig verteilten Zufallszahlen r_1, \dots, r_m im Intervall $[0, 1]$
- Benutze diese Sequenz, um eine andere Sequenz x_1, \dots, x_n zu erzeugen, die einer für uns interessanten Wahrscheinlichkeitsdichtefunktion $f(x)$ folgt
- Benutze die Werte x , um Eigenschaften von $f(x)$ zu bestimmen, z. B. Anzahl von x in $a < x < b$: $\int_a^b f(x)dx$
- \Rightarrow Monte-Carlo-Berechnung $\hat{=}$ Integration
- generierte Monte-Carlo-Werte $\hat{=}$ “simulierte Daten”

- 1 Monte-Carlo-Methode
- 2 Zufallszahlen**
- 3 Beliebige verteilte Zufallszahlen
- 4 Monte-Carlo-Ereignis-Generatoren
- 5 Zusammenfassung

Erzeuge gleichverteilte Zahlen im Intervall $[0, 1]$

- Würfle eine Folge von Zahlen \rightarrow “Zufallszahlengenerator”
- Computeralgorithmen zur Erzeugung der Folge I_1, \dots, I_n
- Computeralgorithmen: nur deterministische Zahlenfolgen, sogenannte “Pseudo-Zufallszahlen”

Anforderungen:

- “zufällig” verteilt mit langer Periode (lang $\gg 10^{\mathcal{O}(100)}$)
- Reproduzierbarkeit ist manchmal erwünscht, d. h. gleiche Zahlenfolge von Zufallszahlen bei gleichen Startbedingungen
- schneller Algorithmus

Beispiele:

- **linear kongruenter Generator:**
 - $I_j = (a \cdot I_{j-1} + c) \bmod m$
 - 3 ganzzahlige Konstanten: Multiplikator a , Summand c , Modul m
 - I_0 : Saatzahl (seed)
 - Zahlenfolge I_1, I_2, \dots zwischen 0 und $m - 1$
 - periodische Folge mit maximaler Periode m
 - gleichförmiges $u_j = I_j/m$ in $[0, 1)$
- **Multiplikativer linear kongruenter Generator:**
 - Setze: $c = 0$

Erzeugung von Zufallszahlen (III)

- Beispiel einer periodischen Folge: $a = 3, m = 7, I_0 = 1$

$$I_0 = 1$$

$$I_1 = (3 \cdot 1) \bmod 7 = 3$$

$$I_2 = (3 \cdot 3) \bmod 7 = 2$$

$$I_3 = (3 \cdot 2) \bmod 7 = 6$$

$$I_4 = (3 \cdot 6) \bmod 7 = 4$$

$$I_5 = (3 \cdot 4) \bmod 7 = 5$$

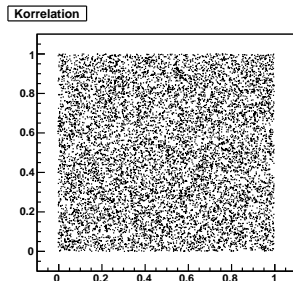
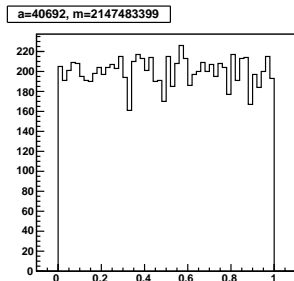
$$I_6 = (3 \cdot 5) \bmod 7 = 1 = I_0$$

$$\Rightarrow I_7 = I_1 \text{ etc.}$$

- \Rightarrow Wähle a, m entsprechend, um lange Periode zu erhalten
- m nahe der größten Integerzahl des Computers
- weniger signifikante Bits sind *weniger* zufällig als Bits höherer Ordnung

Erzeugung von Zufallszahlen (IV)

- u_i sind in $[0, 1)$ aber auch wirklich zufällig?
- Wähle a, m , so dass I_i Zufallszahlentests bestehen (vgl. Blobel / Lohrmann), z. B.
 - gleichförmige Verteilung
 - χ^2 -Test für Unterintervalle von $[0, 1]$
 - Korrelationstest für n -dimensionales Gitter
 - ...
- (Implementation: `demo2.sh`, `ran.py`)



Zufallszahlengeneratoren in ROOT: (Auszug aus Dokumentation)

- TRandom1, based on the RANLUX algorithm, has mathematically proven random proprieties and a period of about 10^{171} . It is however slower than the others.
- TRandom2, is based on the Tausworthe generator of L'Ecuyer, and it has the advantage of being fast and using only 3 words (of 32 bits) for the state. The period is 10^{26} .
- TRandom3, is based on the “Mersenne Twister generator”, and is the recommended one, since it has good random properties (period of about 10^{6000}) and it is fast.

- Monte-Carlo-Methode
- Zufallszahlen
- **3** Beliebige verteilte Zufallszahlen
- Monte-Carlo-Ereignis-Generatoren
- Zusammenfassung

Methoden

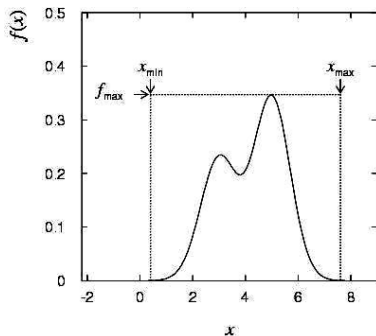
- von Neumannsche Verwerfungsmethode (“acceptance-rejection method”)
- Transformationsmethode
 - u_i in $[0, 1]$ gleichförmig verteilt
 - Suche $x(u)$, so dass $x_I = x(u_i)$ Verteilungsfunktion $f(x)$ folgen
- Spezialfall: gaußverteilte Zufallszahlen

Die Verwerfungsmethode

- Erzeuge eine Zufallszahl x , gleichverteilt in $[x_{\min}, x_{\max}]$, d. h. $x = x_{\min} + r_1(x_{\max} - x_{\min})$, wobei r_1 gleichverteilt in $[0, 1]$.
- Generiere eine zweite, unabhängige Zufallszahl gleichverteilt zwischen 0 und f_{\max} , d. h. $u = r_2 \cdot f_{\max}$ (wobei r_2 gleichverteilt in $[0, 1]$)
- Wenn $u < f(x)$, akzeptiere x als Zufallszahl; falls nicht, wiederhole die Prozedur
- Effizienz: $1/f_{\max}$

entspricht Integration

- Integration einer Wahrscheinlichkeitsdichtefunktion (probability density function, PDF)
- In 2-D: umschlieÙe Funktion mit einer definierten Fläche



Die Verwerfungsmethode – Beispiel

Berechnung der Zahl π

$$f(x) = \sqrt{1 - x^2}$$

im Intervall $[0, 1]$

Akzeptiere Punktepaar, wenn

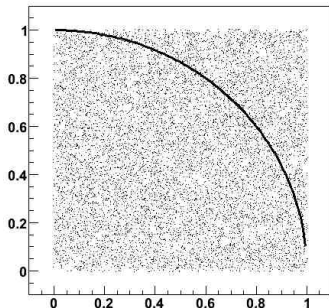
$$x^2 + y^2 \leq 1$$

Bilde Verhältnis von

“Innerhalb/Gesamt”

$$\pi = 4 \cdot \text{Innerhalb/Gesamt}$$

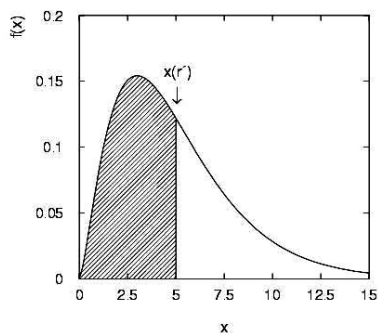
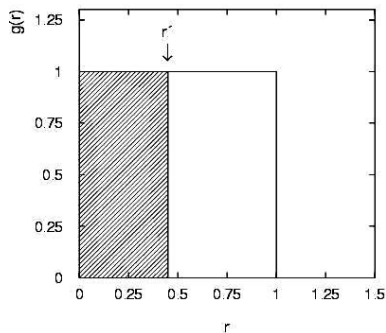
Berechnung von π



⇒ DEMO: `demo3.sh`, `pi.py`

Die Transformationsmethode (I)

- Gegeben u_1, \dots, u_n gleichförmig verteilt in $[0, 1]$
- Finde x_1, \dots, x_n , die $f(x)$ folgen, durch geeignete Transformation $x(u)$

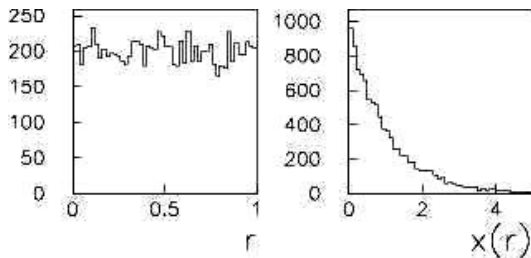


- PDF $f(x) \Rightarrow$ CDF $F(x) : x \rightarrow [0, 1]$
 - kumulative Verteilungsfunktion (CDF) $F(x) \doteq \int_{-\infty}^x f(t)dt$
- Umkehrfunktion $x(u) = F^{-1}(u) : [0, 1] \rightarrow \text{Urbild}(f)$

Die Transformationsmethode (II)

Beispiel:

- Exponentielle Verteilung mit Parameter λ :
PDF $f(x) = \lambda \cdot \exp(-\lambda \cdot x)$ für $x \geq 0$
- $\Rightarrow F(x) = \int_{t_{\min}=0}^x \lambda \cdot \exp(-\lambda \cdot t) dt = 1 - \exp(-\lambda \cdot x)$ (für $x \geq 0$)
- $u(x) \doteq F(x) \Rightarrow x(u) = -\ln(1 - u)/\lambda = -\ln(u)/\lambda$



- Integral $F(x)$ der Verteilung $f(x)$ muss bekannt und invertierbar sein.
- **DEMO:** `inversion.C` ($\lambda = 2$)

Erzeugung gaußverteilter Zufallszahlen

Standardisierte Normalverteilung: $\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$

- Transformationsmethode funktioniert hier nicht
- Einfacher Algorithmus basierend auf Zentralem Grenzwertsatz:

$$x_i = \sum_{j=1}^{12} u_j - 6$$

- Nachteil: ineffizient (12 Zufallszahlen nötig, um 1 zu erzeugen)
- Werte begrenzt auf $[-6, 6]$
- **DO-IT-YOURSELF**: `naive_gauss.C`
- Box-Muller-Verfahren:
 - Erzeuge gleichförmig verteilte u_1, u_2 in $[0, 1]$
 - Berechne: $v_1 = 2u_1 - 1, v_2 = 2u_2 - 1$
(v_1, v_2 gleichförmig verteilt in $[-1, +1]$)
 - Berechne: $r^2 = v_1^2 + v_2^2$.
 - Falls $r^2 > 1$, beginne von vorne.
sonst: $x_1 = v_1 \sqrt{\frac{-2 \ln r^2}{r^2}}$ und $x_2 = v_2 \sqrt{\frac{-2 \ln r^2}{r^2}}$
 - x_1 und x_2 sind unabhängig normal verteilt.
 - (Implementation: `ran3.py` oder `ran3.C`)

Vergleiche Genauigkeit von Monte-Carlo mit anderen Methoden

- Monte-Carlo-Berechnung $\hat{=}$ Integration

Für 1-dimensionales Integral:

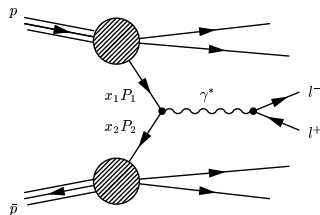
- MC:
 - n : Anzahl der generierten Zufallszahlen
 - Fehler: $\sim 1/\sqrt{n}$
- Numerische Integration mit Trapezregel (oder Simpsonscher Regel)
 - $n \sim$ Anzahl der Intervalle
 - Fehler: $\sim 1/n^2$ ($\sim 1/n^4$)
- Numerische Methoden genauer mit weniger Rechenaufwand

Für N-dimensionales Integral:

- MC: Fehler $\sim 1/\sqrt{n}$, unabhängig von Dimension
- Trapez: Fehler $\sim 1/(\sqrt[n]{n^2})$
- Für $d \geq 4$ MC-Methode besser
 - außerdem: Integrationsgrenzen flexibler, Genauigkeit "erweiterbar"

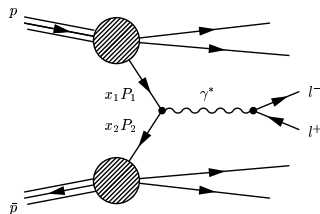
- 1 Monte-Carlo-Methode
- 2 Zufallszahlen
- 3 Beliebige verteilte Zufallszahlen
- 4 Monte-Carlo-Ereignis-Generatoren**
- 5 Zusammenfassung

Einfache Reaktion:
 $p\bar{p} \rightarrow Z/\gamma^* \rightarrow e^+e^-$



- Ereignis-Generatoren werden benutzt, um Teilchenreaktionen zu simulieren
- z. B. PYTHIA, HERWIG++, SHERPA, ALPGEN, POWHEG, ...
- Ausgabe: "Ereignisse", d. h. für jedes Ereignis wird eine Liste von Teilchen generiert zusammen mit den Vierervektoren, etc.

Einfache Reaktion:
 $p\bar{p} \rightarrow Z/\gamma^* \rightarrow e^+e^-$



- Ereignis-Generatoren werden benutzt, um Teilchenreaktionen zu simulieren
- z. B. PYTHIA, HERWIG++, SHERPA, ALPGEN, POWHEG, ...
- Ausgabe: "Ereignisse", d. h. für jedes Ereignis wird eine Liste von Teilchen generiert zusammen mit den Vierervektoren, etc.
- "Ereignis" = 1 Kollisionsereignis bzw. die Menge aller Detektordaten, die einer Kollision von zwei Protonenbündeln zugeordnet werden
- Detektordaten: "echte" Daten oder simulierte Daten

Monte-Carlo-Ereignis-Generatoren

Beispiel

```
----- LHA event information and listing -----  
process =          1  weight = 1.8982e-05  scale = 3.3269e+02 (GeV)  
          alpha_em = 7.8165e-03  alpha_strong = 1.0700e-01  
  
Participating Particles  
no      id stat  mothers  colours  p_x      p_y      p_z      e      m      tau  sp  
1        2  -1    0  0    501  0    0.000  0.000  2832.442  2832.442  0.000  0.000  -1.0  
2        -1 -1    0  0    0  501  0.000  0.000  -60.411  60.411  0.000  0.000  1.0  
3  1000024  1    1  2    0  0    21.435  218.452  526.439  622.751  250.000  0.000  -1.0  
4  1000023  1    1  2    0  0   -21.435 -218.452  2245.592  2270.102  250.000  0.000  -1.0  
  
----- End LHA event information and listing -----
```

- Detektorsimulation erhält Teilchenliste aus Ereignis-Generator als Eingabe
- Detektorsimulation:
 - Simuliert Durchgang der verschiedenen Teilchen durch Detektorkomponenten
 - Coulombstreuung (simuliert Streuwinkel)
 - Teilchenzerfälle (simuliert Lebensdauer)
 - Ionisierungsenergie (simuliert ΔE)
 - Elektromagnetische / hadronische Schauer
 - Elektronische Signale in Detektoren
- Simulierte Ausgabe hat gleiches Format wie echte Daten
 - Einfacher Vergleich zwischen Daten und MC (vorausgesetzt die Effizienzen sind gleich)
- Programmpaket: GEANT4 (*toolkit for the simulation of the passage of particles through matter, initiated 1994*)
 - verwendet von ATLAS, CMS, ALICE, LHCb, ILC, . . . , Astrophysikern, in klinischen Studien, für Simulation von Strahlungsgefahr für Astronauten, . . .

Komplexes Beispiel zur Monte Carlo Methode

Moderne Experimente der Hochenergiephysik bestehen aus sehr vielen einzelnen Detektoren

- L3 am LEP Beschleuniger (CERN) hatte u.a. etwa 11 000 Kristalle zur Energiemessung
- CMS am LHC Beschleuniger wird ca. 15 000 Silizium-Streifendetektoren enthalten mit etwa 10^7 einzelnen Kanälen

Zur Analyse der Daten werden sehr detaillierte MC Simulationen benötigt

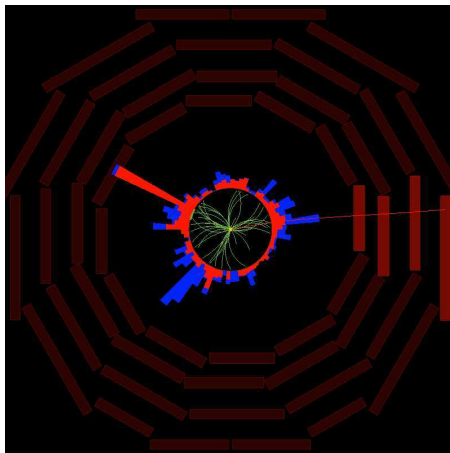
- Simulation der physikalischen Reaktion:
alle entstehenden Teilchen und deren erwartete Energie-, Impuls- und Winkelverteilungen
- Nachweiswahrscheinlichkeit für jedes Detektorelement
- Orts- und Energieauflösung jeder einzelnen Detektorkomponente

Am Ende der Simulation stehen digitalisierte Signale der einzelnen Detektorkomponenten, die sich nicht von echten Daten unterscheiden

Der simulierte Datensatz dient dann zur Optimierung der Selektion und Bestimmung der Akzeptanz

Komplexes Beispiel zur Monte Carlo Methode

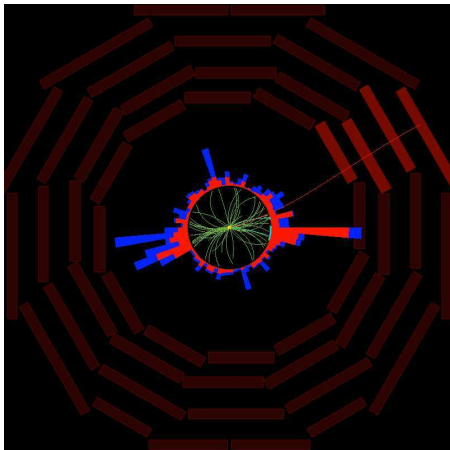
CMS Experiment am LHC Beschleuniger am CERN:
Simulation eines Top-Paar-Ereignisses $pp \rightarrow t\bar{t} + X$



Komplexes Beispiel zur Monte Carlo Methode

CMS Experiment am LHC Beschleuniger am CERN:

Im Vergleich zu einem realen Ereignis in den Daten $pp \rightarrow t\bar{t} + X$



Event generators

	General-Purpose	Specialized
Hard Processes	HERWIG PYTHIA SHERPA 	MadGraph, AlpGen, ...
Resonance Decays		HDECAY, ...
Parton Showers		Ariadne/LDC, VINCIA, ...
Underlying Event		PHOJET/DPMJET
Hadronization		none (?)
Ordinary Decays		TAUOLA, EvtGen

Specialized often best at given task, but need General-Purpose core

Monte-Carlo Event Generation

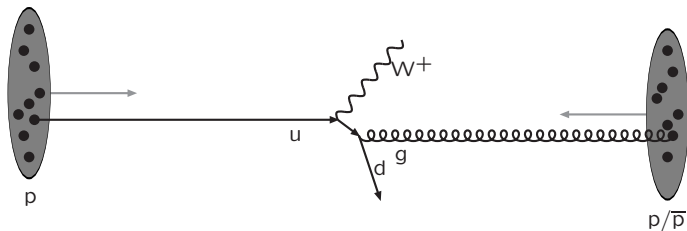
Warning: schematic only, everything simplified, nothing to scale, ...



Incoming beams: parton densities

Slide: Torbjorn Sjostrand

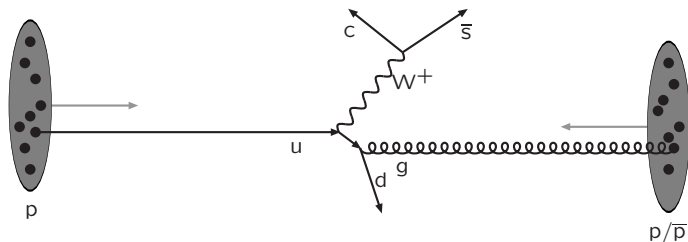
Monte-Carlo Event Generation



Hard subprocess: described by matrix elements

Slide: Torbjorn Sjostrand

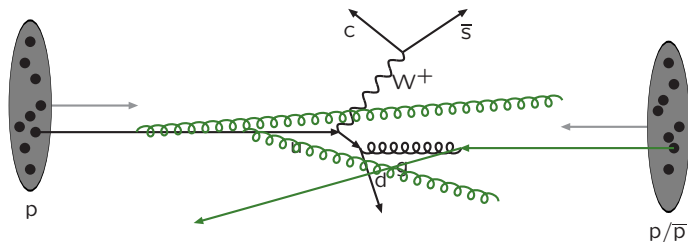
Monte-Carlo Event Generation



Resonance decays: correlated with hard subprocess

Slide: Torbjorn Sjostrand

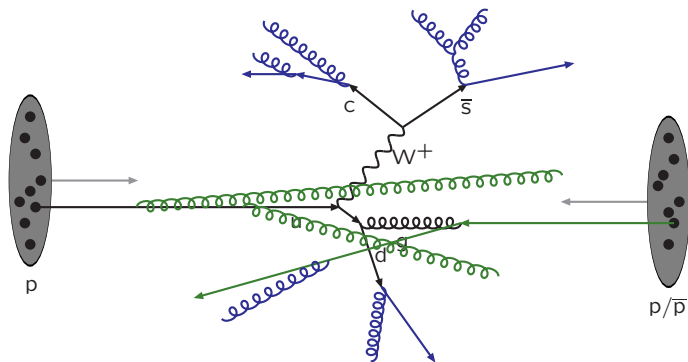
Monte-Carlo Event Generation



Initial-state radiation: spacelike parton showers

Slide: Torbjorn Sjostrand

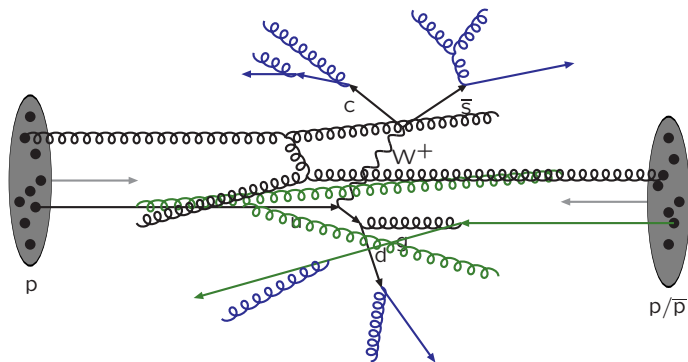
Monte-Carlo Event Generation



Final-state radiation: timelike parton showers

Slide: Torbjorn Sjostrand

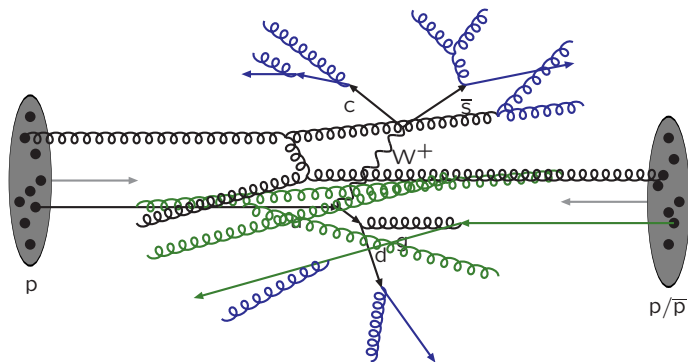
Monte-Carlo Event Generation



Multiple parton-parton interactions ...

Slide: Torbjorn Sjostrand

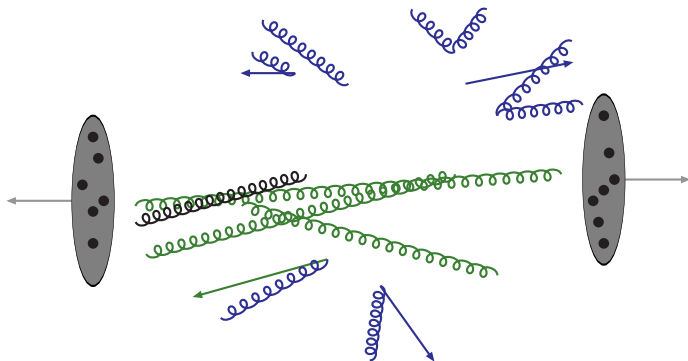
Monte-Carlo Event Generation



... with its **initial-** and **final-**state radiation

Slide: Torbjorn Sjostrand

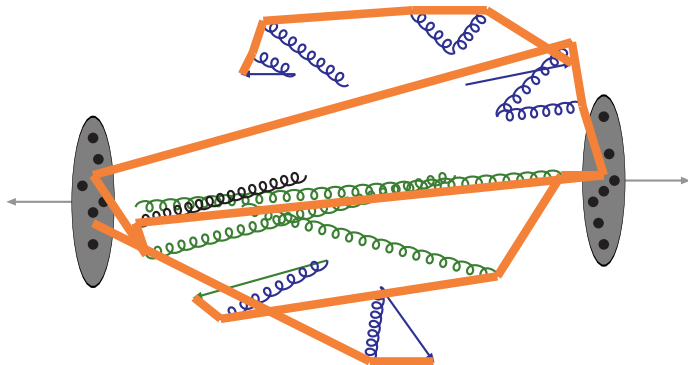
Monte-Carlo Event Generation



Beam remnants and other outgoing partons

Slide: Torbjorn Sjostrand

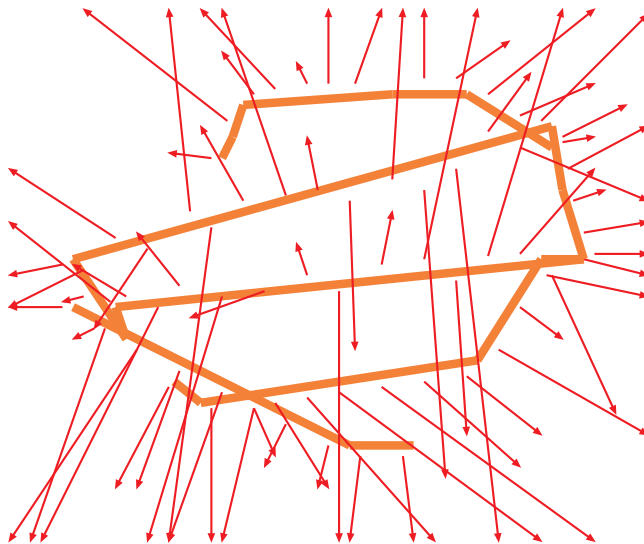
Monte-Carlo Event Generation



Everything is connected by colour confinement strings
Recall! Not to scale: strings are of hadronic widths

Slide: Torbjorn Sjostrand

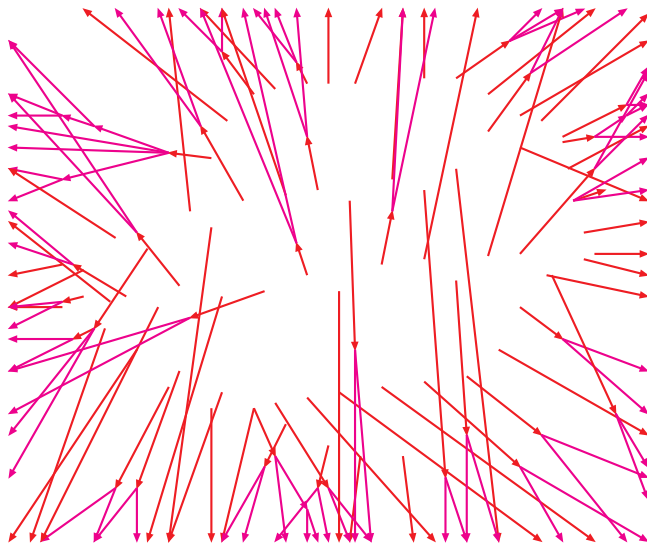
Monte-Carlo Event Generation



The strings fragment to produce primary hadrons

Slide: Torbjorn Sjostrand

Monte-Carlo Event Generation



Many hadrons are unstable and decay further

Slide: Torbjorn Sjostrand

Monte-Carlo-Ereignis-Generatoren

Beispiel

----- PYTHIA Event Listing (complete event) -----

no	id	name	status	mothers	daughters	colours	P_X	P_Y	P_Z	#	W			
0	00	(system)	-12	0	0	0	0.000	0.000	0.000	8000.000	8000.000			
1	2212	(p+)	-12	0	0	0	0.000	0.000	4000.000	4000.000	0.938			
2	2212	(p+)	-12	0	0	0	0.000	0.000	-4000.000	4000.000	0.938			
3	1	(g)	-21	7	5	501	0.000	2832.442	2832.442	0.000	0.000			
4	-1	(dBar)	-21	8	0	5	0.000	0.000	-60.411	60.411	0.000			
5	1000024	(-chi_1+)	-22	3	4	9	0	21.435	218.452	526.439	622.751	250.000		
6	1000023	(-chi_20)	-22	3	13	18	0	-21.435	-219.452	2245.592	2270.102	250.000		
7	2	(u)	-42	41	41	3	501	0	0.000	2832.442	2832.442	0.000		
8	-1	(dBar)	-41	40	40	11	4	0	0.000	-188.076	188.076	0.000		
9	1000024	(-chi_1+)	-44	5	5	43	4	0	25.711	213.290	515.821	612.148	250.000	
10	1000023	(-chi_20)	-44	6	6	44	4	0	-20.347	-219.766	2256.062	2280.576	250.000	
11	21	(g)	-43	8	0	38	39	501	503	-5.364	6.476	-127.517	127.794	0.000
12	21	(g)	-31	19	4	15	505	504	0.000	0.000	20.177	20.177	0.000	
13	21	(g)	-31	20	0	14	15	504	506	0.000	0.000	-9.805	9.805	0.000
14	21	(g)	-33	12	13	16	17	507	508	-3.818	1.424	-9.162	10.028	0.000
15	21	(g)	-33	12	13	18	18	508	507	3.818	-1.424	19.534	19.965	0.000
16	21	(g)	-51	14	0	21	21	508	506	-1.731	3.062	1.424	3.795	0.000
17	21	(g)	-51	14	0	23	23	507	508	-1.632	-1.807	-8.259	8.610	0.000
18	21	(g)	-52	15	15	22	22	505	507	3.363	-1.255	17.207	17.577	0.000
19	21	(g)	-42	25	25	12	12	505	504	-0.000	0.000	20.177	20.177	0.000
20	21	(g)	-41	26	0	24	13	509	506	0.000	-0.000	-41.014	41.014	0.000
21	21	(g)	-44	16	16	27	27	508	506	-1.437	3.004	-1.118	3.513	0.000
22	21	(g)	-44	18	18	28	28	505	507	3.409	-1.264	17.475	17.849	0.000
23	21	(g)	-44	17	17	29	29	507	508	0.461	-2.217	-8.370	8.671	0.000
24	21	(g)	-43	20	0	30	30	509	504	-2.434	0.476	-31.059	31.158	0.000
25	21	(g)	-42	51	0	19	19	505	504	0.000	0.000	20.177	20.177	0.000
26	2	(u)	-41	52	52	31	20	509	0	-0.000	0.000	-646.205	646.205	0.000
27	21	(g)	-44	21	21	37	37	508	506	-1.451	3.037	-1.165	3.562	0.000
28	21	(g)	-44	22	22	34	34	505	507	3.407	-1.259	17.429	17.804	0.000
29	21	(g)	-44	23	23	32	33	507	508	0.362	-1.995	-8.405	8.644	0.000
30	21	(g)	-44	24	24	36	36	508	506	-2.795	0.322	-31.046	31.158	0.000
31	2	(u)	-43	26	0	57	57	506	0	0.476	-1.115	-605.171	605.172	0.330
32	21	(g)	-52	29	0	35	35	508	508	-1.789	3.186	-7.906	8.226	0.000
33	21	(g)	-51	29	0	50	50	507	510	-0.922	-0.223	0.020	0.949	0.000
34	21	(g)	-52	28	28	48	49	505	507	3.396	-1.221	16.910	17.273	0.000
35	21	(g)	-52	25	25	55	55	511	505	0.422	-11.705	-7.406	7.611	0.000
36	21	(g)	-51	32	0	59	59	511	508	0.832	0.181	-0.394	0.938	0.000
37	21	(g)	-52	27	27	53	53	508	506	-1.319	2.761	1.059	3.238	0.000
38	21	(g)	-51	11	0	45	45	505	503	-1.455	0.455	-15.992	16.035	0.000
39	21	(g)	-51	11	0	46	46	512	503	-4.273	6.022	-116.692	116.926	0.000
40	-1	(dBar)	-53	42	0	8	8	0	503	-0.000	0.000	-193.243	193.243	0.000
41	2	(u)	-42	62	0	7	7	0	503	0.000	0.000	2832.442	2832.442	0.000
42	-1	(dBar)	-41	63	63	47	40	0	513	0.000	-0.000	-217.113	217.113	0.000
43	1000024	(-chi_1+)	-44	9	9	64	64	0	0	25.812	213.204	515.658	611.985	250.000
44	1000023	(-chi_20)	-44	10	10	65	65	0	0	-20.321	-219.766	2256.233	2280.747	250.000
45	21	(g)	-44	38	38	66	66	501	512	-1.057	0.426	-15.993	16.034	0.000
46	21	(g)	-44	39	39	67	67	512	503	-4.627	5.814	-116.702	116.916	0.000
47	21	(g)	-43	48	0	51	51	503	513	-2.478	0.343	-23.868	23.874	0.000
48	21	(g)	-51	34	0	54	54	505	514	3.119	-1.352	16.843	17.183	0.000
49	21	(g)	-51	34	0	60	60	514	507	0.109	0.112	0.069	0.171	0.000
50	21	(g)	-51	38	0	58	58	507	510	-0.844	-0.204	0.018	0.865	0.000
51	21	(g)	-41	69	69	61	25	505	515	0.000	0.000	22.541	22.541	0.000
52	2	(u)	-42	70	70	26	26	509	0	0.000	0.000	-646.205	646.205	0.000
53	21	(g)	-44	37	37	71	71	508	506	-1.110	2.804	-0.937	3.261	0.000
54	21	(g)	-44	48	48	72	72	505	514	3.189	-1.040	16.848	17.179	0.000
55	21	(g)	-44	35	35	73	73	510	511	0.423	-1.703	-7.391	7.596	0.000
56	21	(g)	-62	43	43	83	84	0	504	-2.785	3.324	-31.052	31.064	0.000
57	2	(u)	-44	31	31	75	75	506	0	0.476	-1.115	-605.145	605.147	0.330
58	21	(g)	-44	50	50	76	76	507	510	-0.842	0.196	0.022	0.864	0.000
59	21	(g)	-44	36	36	77	77	511	508	0.833	-0.186	-0.398	0.942	0.000
60	21	(g)	-44	49	49	78	78	514	507	0.110	0.114	0.067	0.172	0.000
61	21	(g)	-43	51	0	79	79	504	515	-0.684	-0.369	-2.348	2.378	0.000
62	2	(u)	-41	70	0	80	80	0	504	0.000	0.000	2832.440	2832.440	0.000
63	-1	(dBar)	-61	2	0	42	42	0	504	0.466	-1.387	-217.111	217.116	0.000
64	1000024	(-chi_1+)	-62	43	43	83	84	0	504	-2.785	3.324	-31.052	31.064	0.000
65	1000023	(-chi_20)	-62	44	44	102	103	0	0	-20.769	-219.221	2256.941	2281.397	250.000
66	21	(g)	-62	45	45	129	129	501	512	-1.022	0.324	-15.996	16.031	0.000
67	21	(g)	-62	46	46	138	138	502	503	-1.776	3.068	-116.725	116.996	0.000
68	21	(g)	-62	47	47	137	137	503	504	-0.357	0.191	-23.869	23.872	0.000

Monte-Carlo-Ereignis-Generatoren

Beispiel

150	111	(pi0)	-83	129	138	241	242	0	0	0	0.016	-0.348	-1.427	1.475	0.135
151	-211	pi-	83	129	138	0	0	0	0	0	-0.713	0.092	-8.936	8.966	0.140
152	111	(pi0)	-83	129	138	243	244	0	0	0	-0.730	0.018	-6.471	6.514	0.135
153	2212	p+	83	129	138	0	0	0	0	0	0.146	-0.262	-3.643	3.774	0.938
154	-2212	pbar-	83	129	138	0	0	0	0	0	0.002	-0.378	-3.757	3.890	0.938
155	211	pi+	84	129	138	0	0	0	0	0	0.096	0.236	0.254	0.386	0.140
156	-211	pi-	84	129	138	0	0	0	0	0	0.022	0.034	-0.651	0.667	0.140
157	111	(pi0)	-84	129	138	245	246	0	0	0	0.473	0.554	-0.572	0.936	0.135
158	111	(pi0)	-84	129	138	247	248	0	0	0	0.076	0.007	0.592	0.612	0.135
159	211	pi+	84	129	138	0	0	0	0	0	-0.945	0.649	-0.623	1.312	0.140
160	-211	pi-	84	129	138	0	0	0	0	0	0.400	-0.003	0.444	0.613	0.140
161	111	(pi0)	-84	129	138	249	250	0	0	0	-0.544	0.692	0.279	0.934	0.135
162	221	(eta)	-84	129	138	251	253	0	0	0	0.132	0.622	0.674	1.077	0.548
163	213	(rho+)	-84	129	138	205	206	0	0	0	-0.021	-0.167	1.093	1.321	0.723
164	221	(eta)	-84	129	138	254	256	0	0	0	0.021	-0.503	3.688	3.851	0.548
165	111	(pi0)	-84	129	138	257	258	0	0	0	0.302	0.326	0.641	0.792	0.135
166	-211	pi-	84	129	138	0	0	0	0	0	0.093	-0.494	4.701	4.730	0.140
167	111	(pi0)	-84	129	138	259	260	0	0	0	1.988	-0.556	4.496	4.949	0.135
168	211	pi+	84	129	138	0	0	0	0	0	-0.301	0.076	0.851	0.916	0.140
169	-211	pi-	84	129	138	0	0	0	0	0	-0.097	-0.038	0.198	0.264	0.140
170	213	(rho+)	-84	129	138	207	208	0	0	0	0.494	0.095	2.126	2.313	0.762
171	223	(omega)	-84	129	138	261	263	0	0	0	-0.074	-0.733	-0.931	1.411	0.762
172	2	(u)	-71	75	75	174	187	506	0	0	1.129	0.455	-605.178	605.179	0.330
173	2103	(uddL)	-71	80	80	174	187	0	506	0	-0.672	-0.594	1145.015	1145.016	0.771
174	213	(rho+)	-83	172	173	209	210	0	0	0	0.819	0.624	-324.280	324.283	0.827
175	111	(pi0)	-83	172	173	264	265	0	0	0	0.143	-0.327	-166.885	166.885	0.135
176	111	(pi0)	-83	172	173	266	267	0	0	0	0.638	-0.266	95.690	95.693	0.135
177	111	(pi0)	-83	172	173	268	269	0	0	0	-0.227	0.597	-10.469	10.489	0.135
178	-211	pi-	84	172	173	0	0	0	0	0	-0.175	-0.348	-0.982	1.066	0.140
179	213	(rho+)	-84	172	173	211	212	0	0	0	0.931	-0.002	-3.682	3.750	0.710
180	-213	(rho-)	-84	172	173	213	214	0	0	0	-0.506	0.187	-2.946	3.097	0.786
181	213	(rho+)	-84	172	173	215	216	0	0	0	0.474	-0.011	3.125	3.235	0.691
182	111	(pi0)	-84	172	173	270	271	0	0	0	-0.054	-0.655	9.355	9.379	0.135
183	-211	pi-	84	172	173	0	0	0	0	0	0.044	0.546	11.291	11.305	0.140
184	211	pi+	84	172	173	0	0	0	0	0	-0.146	-0.116	3.783	3.790	0.140
185	-211	pi-	84	172	173	0	0	0	0	0	0.136	0.647	60.209	60.212	0.140
186	2224	(Deltat++)	-84	172	173	217	218	0	0	0	-0.253	-1.201	595.259	595.261	1.141
187	-211	pi-	84	172	173	0	0	0	0	0	-0.467	0.186	461.750	461.750	0.140
188	211	pi+	91	118	0	0	0	0	0	0	15.695	41.859	132.836	140.157	0.140
189	111	(pi0)	-91	118	0	272	273	0	0	0	38.554	99.810	317.062	334.629	0.135
190	-211	pi-	91	119	0	0	0	0	0	0	0.098	0.326	1.809	1.846	0.140
191	111	(pi0)	-91	119	0	274	275	0	0	0	-0.054	4.865	15.271	19.129	0.135
192	2212	p+	91	122	0	0	0	0	0	0	0.013	0.394	2.019	2.261	0.938
193	211	pi+	91	122	0	0	0	0	0	0	0.016	0.216	0.862	0.910	0.140
194	211	pi+	91	123	0	0	0	0	0	0	0.328	1.484	4.553	4.802	0.140
195	-211	pi-	91	123	0	0	0	0	0	0	0.252	0.085	0.991	1.036	0.140
196	310	K_S0	91	125	125	0	0	0	0	0	0.282	-0.250	0.413	0.749	0.938

Monte-Carlo-Ereignis-Generatoren

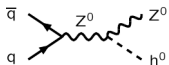
Beispiel

264	22	gamm	91	175	0	0	0	0	0	0.004	-0.108	-30.867	30.868	0.000
265	22	gamm	91	175	0	0	0	0	0	-0.219	0.019	-136.017	136.018	0.000
266	22	gamm	91	176	0	0	0	0	0	0.247	-0.106	-46.600	46.601	0.000
267	22	gamm	91	176	0	0	0	0	0	0.391	-0.160	-49.090	49.092	0.000
268	22	gamm	91	177	0	0	0	0	0	-0.080	0.072	-1.607	1.610	0.000
269	22	gamm	91	177	0	0	0	0	0	-0.147	0.525	-8.662	8.679	0.000
270	22	gamm	91	182	0	0	0	0	0	-0.067	-0.169	2.172	2.179	0.000
271	22	gamm	91	182	0	0	0	0	0	0.313	-0.486	-7.184	7.200	0.000
272	22	gamm	91	189	0	0	0	0	0	31.604	81.710	259.501	273.891	0.000
273	22	gamm	91	189	0	0	0	0	0	6.950	18.099	57.561	60.738	0.000
274	22	gamm	91	191	0	0	0	0	0	1.398	3.466	11.072	11.674	0.000
275	22	gamm	91	191	0	0	0	0	0	0.493	1.402	4.199	4.454	0.000
276	22	gamm	91	200	0	0	0	0	0	0.200	0.244	-1.796	1.824	0.000
277	22	gamm	91	200	0	0	0	0	0	0.184	0.016	-0.727	0.738	0.000
278	22	gamm	91	204	0	0	0	0	0	-0.185	0.244	-5.078	5.088	0.000
279	22	gamm	91	204	0	0	0	0	0	-0.216	0.552	-11.493	11.508	0.000
280	22	gamm	91	206	0	0	0	0	0	-0.375	0.141	0.469	0.486	0.000
281	22	gamm	91	206	0	0	0	0	0	-0.036	0.096	0.090	0.136	0.000
282	22	gamm	91	208	0	0	0	0	0	0.146	0.024	0.439	0.462	0.000
283	22	gamm	91	208	0	0	0	0	0	0.499	-0.106	0.888	1.025	0.000
284	22	gamm	91	210	0	0	0	0	0	0.293	0.232	-166.316	166.317	0.000
285	22	gamm	91	210	0	0	0	0	0	0.220	0.286	-122.073	122.074	0.000
286	22	gamm	91	212	0	0	0	0	0	0.119	-0.015	-2.768	2.770	0.000
287	22	gamm	91	212	0	0	0	0	0	0.041	0.060	-0.632	0.636	0.000
288	22	gamm	91	214	0	0	0	0	0	-0.106	0.244	-1.883	1.902	0.000
289	22	gamm	91	214	0	0	0	0	0	-0.023	0.098	-0.321	0.336	0.000
290	22	gamm	91	216	0	0	0	0	0	0.104	-0.041	0.759	0.767	0.000
291	22	gamm	91	216	0	0	0	0	0	0.668	-0.211	1.419	1.436	0.000
292	22	gamm	91	220	0	0	0	0	0	-0.174	-0.564	3.400	3.450	0.000
293	22	gamm	91	220	0	0	0	0	0	-3.225	-10.397	59.384	60.374	0.000
294	22	gamm	91	221	0	0	0	0	0	-2.664	-7.819	47.018	47.739	0.000
295	22	gamm	91	221	0	0	0	0	0	-0.791	-2.198	13.646	13.844	0.000
296	22	gamm	91	230	0	0	0	0	0	-0.072	0.196	-2.758	2.766	0.000
297	22	gamm	91	230	0	0	0	0	0	0.003	0.139	0.923	0.931	0.000
298	22	gamm	91	231	0	0	0	0	0	0.041	0.070	-2.433	2.435	0.000
299	22	gamm	91	231	0	0	0	0	0	0.015	0.045	-0.214	0.219	0.000
300	22	gamm	91	232	0	0	0	0	0	0.008	0.033	-0.075	0.082	0.000
301	22	gamm	91	232	0	0	0	0	0	0.089	0.005	-1.321	1.324	0.000
302	22	gamm	91	237	0	0	0	0	0	-1.449	0.969	-38.188	38.227	0.000
303	22	gamm	91	237	0	0	0	0	0	-0.015	-0.814	0.814	0.814	0.000
304	22	gamm	91	251	0	0	0	0	0	0.017	0.190	0.065	0.201	0.000
305	22	gamm	91	251	0	0	0	0	0	0.030	0.209	0.247	0.325	0.000
306	22	gamm	91	252	0	0	0	0	0	0.026	-0.171	-0.049	0.180	0.000
307	22	gamm	91	252	0	0	0	0	0	0.006	0.049	0.122	0.132	0.000
308	22	gamm	91	253	0	0	0	0	0	-0.035	-0.032	0.043	0.064	0.000
309	22	gamm	91	253	0	0	0	0	0	0.087	0.036	-0.147	0.175	0.000
310	22	gamm	91	256	0	0	0	0	0	0.043	0.037	0.274	0.280	0.000
311	22	gamm	91	256	0	0	0	0	0	0.120	-0.088	0.322	0.355	0.000
312	22	gamm	91	263	0	0	0	0	0	0.015	-0.171	-0.139	0.223	0.000

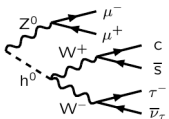
Monte Carlo generation

Matrix elements (ME):

- 1) Hard subprocess:
 $|\mathcal{M}|^2$, Breit-Wigners,
parton densities.

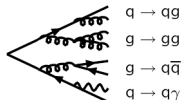


- 2) Resonance decays:
includes correlations.

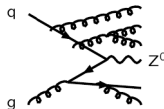


Parton Showers (PS):

- 3) Final-state parton showers.

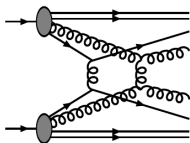


- 4) Initial-state parton showers.

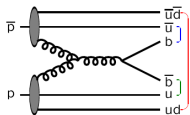


Monte Carlo generation

5) Multiple parton-parton interactions.

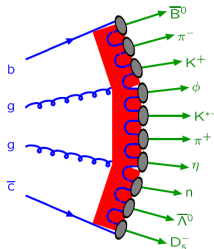


6) Beam remnants, with colour connections.

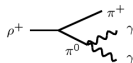


5) + 6) = Underlying Event

7) Hadronization



8) Ordinary decays: hadronic, τ , charm, ...



Slide: Else Lytken et al.

- Monte-Carlo-Methode
- Zufallszahlen
- Beliebig verteilte Zufallszahlen
- Monte-Carlo-Ereignis-Generatoren
- 5** ● Zusammenfassung

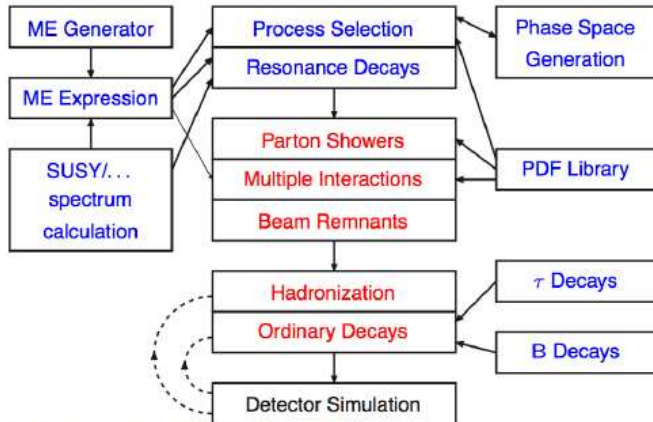
Was haben wir gelernt:

- Monte-Carlo-Methoden: Definition und Beispiele
- Erzeugung von (Pseudo-) Zufallszahlen am Computer
- Erzeugung von beliebig verteilter Zufallszahlen
- Anwendung von Monte-Carlo-Methoden in der Teilchenphysik

Extra

- Seine $f(x)$, $g(y)$ Wahrscheinlichkeitsdichtefunktionen
- zusammenhängend über eine Transformationsfunktion $y = y(x)$
- \Rightarrow allgemein $g(y) = f(x(y)) \left| \frac{dx}{dy} \right|$
- Speziell für $f(x)$ gleichverteilt zwischen 0 und 1: $g(y) = \left| \frac{dx}{dy} \right|$
- Damit ergibt sich (z. B.) mit analoger Rechnung wie oben:
 - (... um die Transformation einer Gleichverteilung in die Exponentialverteilung zu finden)
- $\left| \frac{dx}{dy} \right| = \lambda \cdot \exp(-\lambda \cdot y) \quad (y \geq 0)$
- $\int dx = \int \lambda \cdot \exp(-\lambda \cdot y) dy$
- $x = 1 - \exp(-\lambda \cdot y)$
- $y = -\ln(1 - x)/\lambda$

What they do



Several standardized interfaces!