Monte-Carlo-Methoden in der Teilchenphysik

Alexander Mann

a.mann@lmu.de

(mit Material von Johannes Elmsheuser, Günter Duckeck u. a.)

Blockkurs “Datenauswertung in der Teilchenphysik”

22. April 2020

München

Inhalt

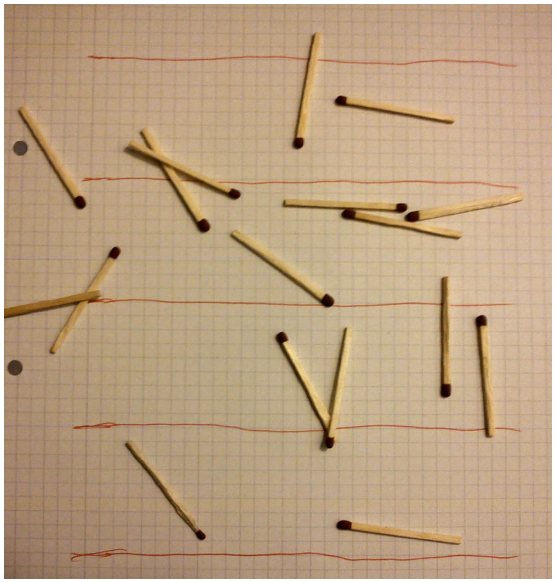
1. Monte-Carlo-Methode
2. Zufallszahlen
3. Beliebig verteilte Zufallszahlen
4. Monte-Carlo-Ereignis-Generatoren
5. Detektorsimulation
6. Zusammenfassung

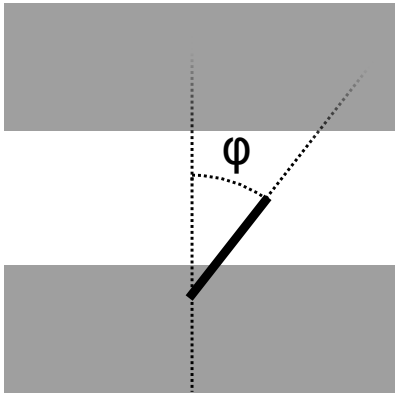
Übersicht

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

Ein klassisches Beispiel (I)

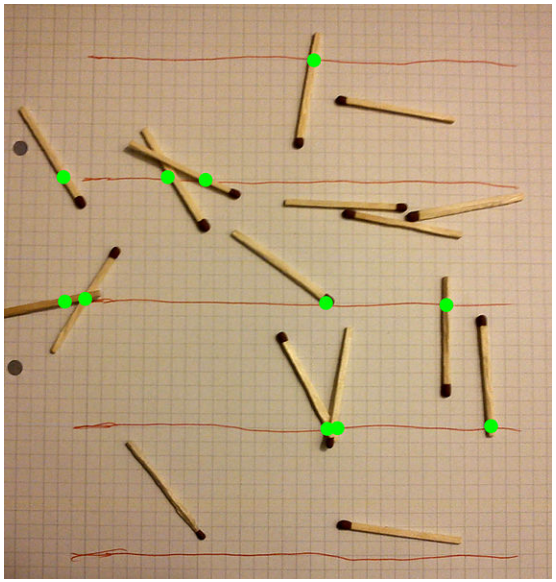
- Historisches Beispiel zur Berechnung der Zahl π: Buffons Nadel
  (Graf G.L.L. von Buffon, 1707 – 1788)
- N Nadeln der Länge l werden auf Fläche mit äquidistanten, parallelen Geraden geworfen (Abstand $d \geq l$).
- Wahrscheinlichkeit für "Geradentreffer" einer einzigen Nadel:
$$p = l_{\text{eff}}/d = l\,|\cos\varphi|\,/d$$
- Integration über gleichverteilte φ:
$$p = \int_0^{2\pi} \frac{l|\cos\varphi|}{d}\frac{\mathrm{d}\varphi}{2\pi} = \frac{2l}{\pi d}$$
- von Mises: $N_{\text{Treffer}}/N \to p$ für $N \to \infty$
$$\Rightarrow \frac{2\cdot N\cdot l}{N_{\text{Treffer}}\cdot d} \to \pi$$
- Frage: Was kommt oben rechts für π raus?

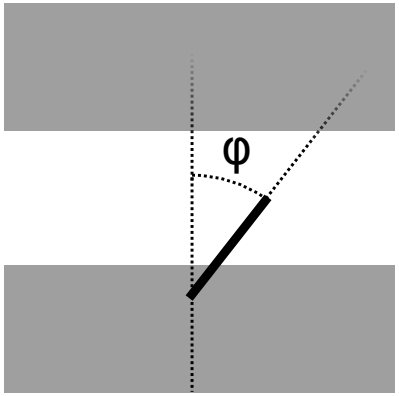




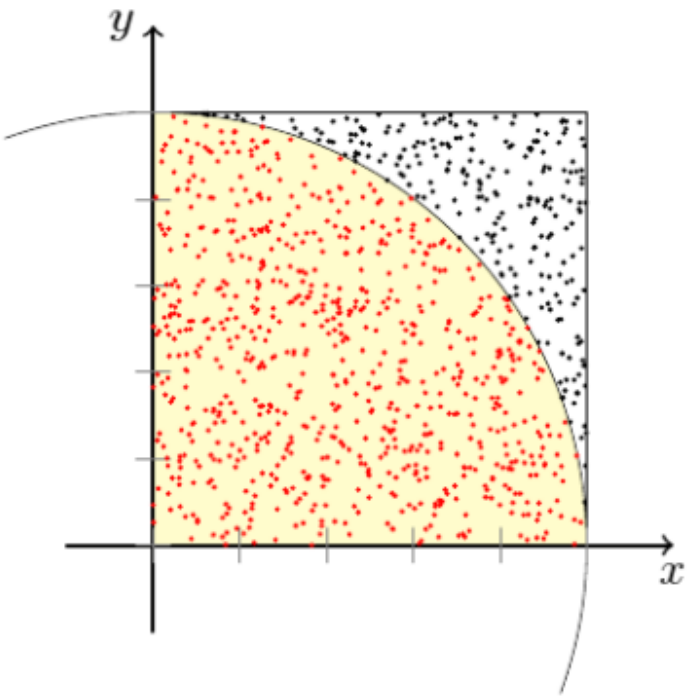
Ein klassisches Beispiel (I)

- 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:
$$p = \int_0^{2\pi} \frac{l|\cos\varphi|}{d}\frac{\mathrm{d}\varphi}{2\pi} = \frac{2l}{\pi d}$$
- von Mises: $N_{\text{Treffer}}/N \to p$ für $N \to \infty$
$$\Rightarrow \frac{2\cdot N\cdot l}{N_{\text{Treffer}}\cdot d} \to \pi$$
- Frage: Was kommt oben rechts für π raus?
  mit $l \approx d$: $\pi \approx 2 \cdot 17/11 \approx 3{,}\overline{09}$





Ein klassisches Beispiel (II)



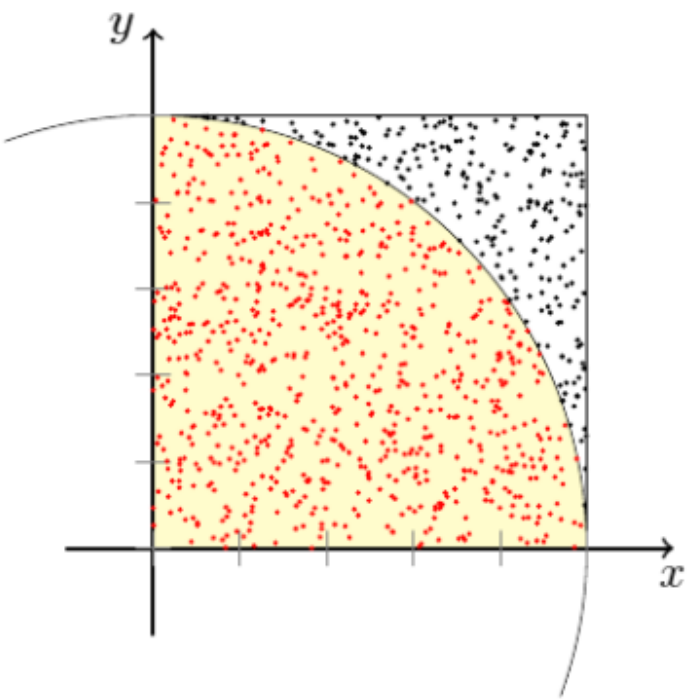
Ähnlich: Näherung der Kreiszahl π
über das Verhältnis von "Treffern" t
und der Gesamtzahl der "Versuche" n,

$$\lim_{n\to\infty} t/n = \pi/4$$

(weil $A_{\triangle} = \pi r^2/4$ und $A_{\square} = r^2$)

(Implementation: `animate_pi_circle.py`)

Ein klassisches Beispiel (II)



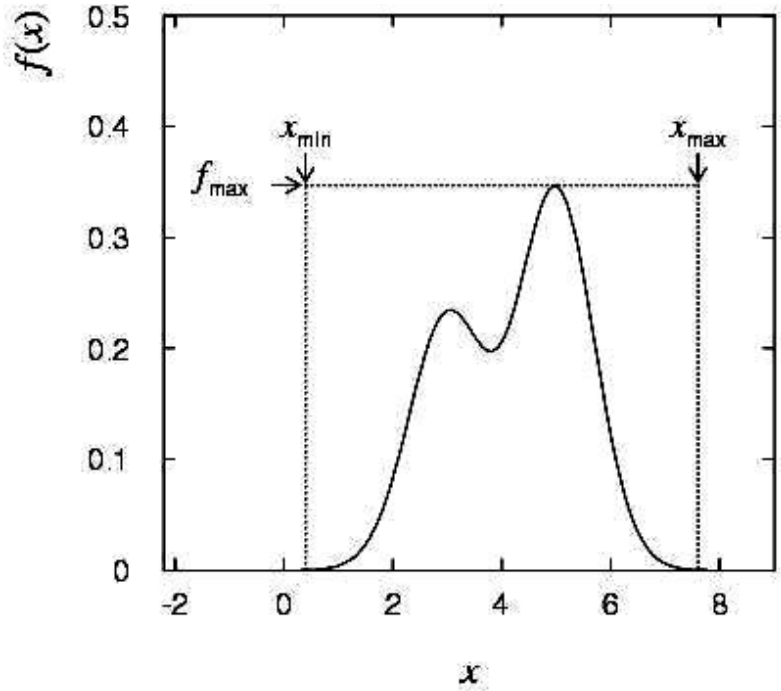
Ähnlich: Näherung der Kreiszahl π über das Verhältnis von "Treffern" t und der Gesamtzahl der "Versuche" n,

$$\lim_{n\to\infty} t/n = \pi/4$$

(weil $A_{\triangle} = \pi r^2/4$ und $A_{\square} = r^2$)

(Implementation: `animate_pi_circle.py`)

Analog: Integration einer Wahrscheinlichkeitsdichtefunktion



wie im vorhergehenden Beispiel: simuliere die Anzahl der "Treffer" und "Nicht-Treffer"

Monte-Carlo-Methode

Definition “Monte-Carlo-Methode”

Wikipedia: Monte-Carlo-Algorithmen sind randomisierte Algorithmen, die mit einer nichttrivial nach oben beschränkten Wahrscheinlichkeit ein falsches Ergebnis liefern dürfen.

Monte-Carlo-Methode

Definition “Monte-Carlo-Methode”

Weniger abstrakt: *numerische* Methode zur Berechnung von Wahrscheinlichkeiten und abgeleiteten Größen unter Benutzung von *Zufallszahlen*

Monte-Carlo-Methode

Definition “Monte-Carlo-Methode”

Weniger abstrakt: *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 zur Bestätigung einer analytischen Lösung
    als unabhängige (und ggf. schnell implementierte) Gegenprobe
- Grundlage:
  - sehr häufig durchgeführte Zufallsexperimente
  - Gesetz der großen Zahlen

Monte-Carlo-Methode

Definition “Monte-Carlo-Methode”

Weniger abstrakt: *numerische* Methode zur Berechnung von Wahrscheinlichkeiten und abgeleiteten Größen unter Benutzung von *Zufallszahlen*

Umsetzung

- Erzeuge eine Sequenz von gleichförmig verteilten Zufallszahlen u_i
- Benutze diese Sequenz, um eine andere Sequenz $x_1, \ldots, x_n$ 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_i in $a < x < b \approx \int_a^b f(x)\mathrm{d}x$
- in Teilchenphysik oft Monte-Carlo-Berechnung $\widehat{=}$ Phasenraum-Integration $I = \int_a^b g(x)\mathrm{d}x \approx I_{\mathrm{MC}} = \frac{b-a}{n} \sum_{i=1}^{n} g(x_i)$

$\rightarrow$ Woher kriegen wir die Zufallszahlen?

Übersicht

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

Erzeugung von Zufallszahlen (I)

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

- Würfle eine Folge von Zahlen $\rightarrow$ “Zufallszahlengenerator”
- Computeralgorithmen zur Erzeugung der Folge $u_1, \ldots, u_n$
- Computeralgorithmen: nur deterministische Zahlenfolgen, sogenannte “Pseudo-Zufallszahlen”
- (es gibt spezielle Hardware für echte Zufallszahlengenerierung)

Anforderungen:

- “zufällig” verteilt
  - Gleichförmigkeit, Unabhängigkeit
- lange Periode (lang $\gg 10^{\mathcal{O}(100)}$)
- schnell zu berechnen, geringer Speicherplatzbedarf
- Reproduzierbarkeit ist manchmal erwünscht, d. h. gleiche Zahlenfolge von Zufallszahlen bei gleichen Startbedingungen

Erzeugung von Zufallszahlen (II)

Beispiele:

- linear kongruenter Generator:
  - $I_j = (a \cdot I_{j-1} + c) \mod m$
  - 3 ganzzahlige Konstanten: Multiplikator a, Summand c, Modul m
  - I_0: Initialwert *(random seed)*
  - Zahlenfolge $I_1, I_2, ...$ zwischen 0 und $m - 1$
    - $\Rightarrow I_j$: periodische Folge mit maximaler Periode m
    - $\Rightarrow$ Zufallszahlen $u_j = I_j/m$ in $[0, 1)$
- Multiplikativer linear kongruenter Generator:
  - Spezialfall mit $c = 0$ (dann muss $I_0 > 0$)

Erzeugung von Zufallszahlen (III)

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

$$
\begin{aligned}
I_0 &= 1 \\
I_1 &= (3 \cdot 1) \mod 7 = 3 \\
I_2 &= (3 \cdot 3) \mod 7 = 2 \\
I_3 &= (3 \cdot 2) \mod 7 = 6 \\
I_4 &= (3 \cdot 6) \mod 7 = 4 \\
I_5 &= (3 \cdot 4) \mod 7 = 5 \\
I_6 &= (3 \cdot 5) \mod 7 = 1 = I_0 \\
\Rightarrow I_7 &= I_1 \text{ etc.}
\end{aligned}
$$

- "Wirkt zufällig": $1, 3, 2, 6, 4, 5, \ldots$
- $\Rightarrow$ 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 (III)

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

$$
\begin{aligned}
I_0 &= 1 \\
I_1 &= (3 \cdot 1) \mod 7 = 3 \\
I_2 &= (3 \cdot 3) \mod 7 = 2 \\
I_3 &= (3 \cdot 2) \mod 7 = 6 \\
I_4 &= (3 \cdot 6) \mod 7 = 4 \\
I_5 &= (3 \cdot 4) \mod 7 = 5 \\
I_6 &= (3 \cdot 5) \mod 7 = 1 = I_0 \\
\Rightarrow I_7 &= I_1 \text{ etc.}
\end{aligned}
$$

- "Wirkt zufällig": $1, 3, 2, 6, 4, 5, \ldots$
- $\Rightarrow$ 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 (III)

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

$$
\begin{aligned}
I_0 &= 1 \\
I_1 &= (3 \cdot 1) \mod 7 = 3 \\
I_2 &= (3 \cdot 3) \mod 7 = 2 \\
I_3 &= (3 \cdot 2) \mod 7 = 6 \\
I_4 &= (3 \cdot 6) \mod 7 = 4 \\
I_5 &= (3 \cdot 4) \mod 7 = 5 \\
I_6 &= (3 \cdot 5) \mod 7 = 1 = I_0 \\
\Rightarrow I_7 &= I_1 \text{ etc.}
\end{aligned}
$$

- "Wirkt zufällig": $1, 3, 2, 6, 4, 5, \ldots$
- $\Rightarrow$ 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 (III)

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

$$
\begin{aligned}
I_0 &= 1 \\
I_1 &= (3 \cdot 1) \mod 7 = 3 \\
I_2 &= (3 \cdot 3) \mod 7 = 2 \\
I_3 &= (3 \cdot 2) \mod 7 = 6 \\
I_4 &= (3 \cdot 6) \mod 7 = 4 \\
I_5 &= (3 \cdot 4) \mod 7 = 5 \\
I_6 &= (3 \cdot 5) \mod 7 = 1 = I_0 \\
\Rightarrow I_7 &= I_1 \text{ etc.}
\end{aligned}
$$

- “Wirkt zufällig”: $1, 3, 2, 6, 4, 5, \ldots$
- $\Rightarrow$ 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 (III)

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

$$
\begin{aligned}
I_0 &= 1 \\
I_1 &= (3 \cdot 1) \mod 7 = 3 \\
I_2 &= (3 \cdot 3) \mod 7 = 2 \\
I_3 &= (3 \cdot 2) \mod 7 = 6 \\
I_4 &= (3 \cdot 6) \mod 7 = 4 \\
I_5 &= (3 \cdot 4) \mod 7 = 5 \\
I_6 &= (3 \cdot 5) \mod 7 = 1 = I_0 \\
\Rightarrow I_7 &= I_1 \text{ etc.}
\end{aligned}
$$

- "Wirkt zufällig": $1, 3, 2, 6, 4, 5, \ldots$
- $\Rightarrow$ 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 (III)

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

$$
\begin{aligned}
I_0 &= 1 \\
I_1 &= (3 \cdot 1) \mod 7 = 3 \\
I_2 &= (3 \cdot 3) \mod 7 = 2 \\
I_3 &= (3 \cdot 2) \mod 7 = 6 \\
I_4 &= (3 \cdot 6) \mod 7 = 4 \\
I_5 &= (3 \cdot 4) \mod 7 = 5 \\
I_6 &= (3 \cdot 5) \mod 7 = 1 = I_0 \\
\Rightarrow I_7 &= I_1 \text{ etc.}
\end{aligned}
$$

- "Wirkt zufällig": $1, 3, 2, 6, 4, 5, \ldots$
- $\Rightarrow$ 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 (III)

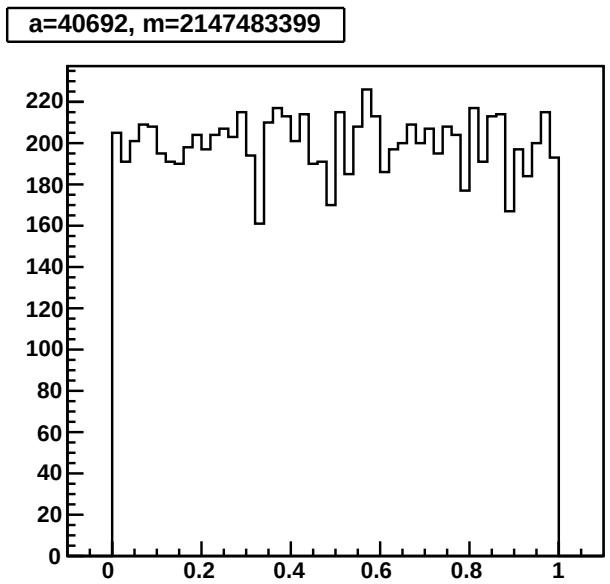
- Beispiel einer periodischen Folge: $a = 3, m = 7, I_0 = 1$

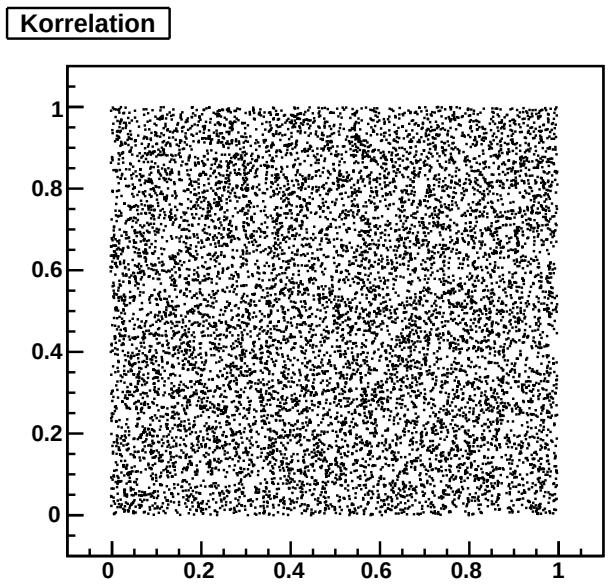
$$
\begin{aligned}
I_0 &= 1 \\
I_1 &= (3 \cdot 1) \mod 7 = 3 \\
I_2 &= (3 \cdot 3) \mod 7 = 2 \\
I_3 &= (3 \cdot 2) \mod 7 = 6 \\
I_4 &= (3 \cdot 6) \mod 7 = 4 \\
I_5 &= (3 \cdot 4) \mod 7 = 5 \\
I_6 &= (3 \cdot 5) \mod 7 = 1 = I_0 \\
\Rightarrow I_7 &= I_1 \text{ etc.}
\end{aligned}
$$

- “Wirkt zufällig”: $1, 3, 2, 6, 4, 5, \ldots$
- $\Rightarrow$ 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)

- $\Rightarrow u_i = I_j/m \in [0, 1)$ *per constructionem* — aber auch wirklich zufällig?
  - Korrelationen würden MC-Berechnungen verfälschen
- Wähle a, m, so dass u_i "Zufallszahlentests" bestehen, zum Beispiel (vgl. Blobel / Lohrmann):
  - gleichförmige Verteilung
    - χ^2-Test für Unterintervalle von $[0, 1]$
  - Korrelationstest für n-dimensionales Gitter
  - ...
- (Implementation: `demo2.sh, demo2b.sh, demo2c.sh`)





Erzeugung von Zufallszahlen (IV)

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 very slow.
- `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. Drawbacks: generates only 32 random bits and fails some random-generator tests
- `TRandomMixMax`, recent improvement on `TRandom3`, with 61 random bits, passing all "TestU01" random tests, fast

Aufruf:

- `gRandom->Uniform()`
- erzeugt unabhängige, gleichverteilte Zufallszahlen $\in [0, 1)$

Übersicht

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

Erzeugung beliebig verteilter Zufallszahlen

Methoden

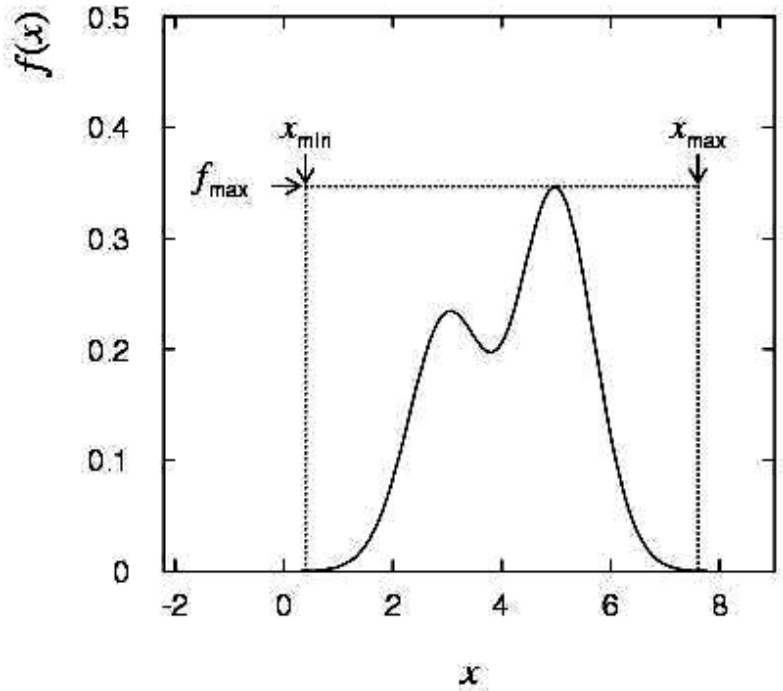
- Gesucht: Zufallszahlen mit beliebiger Verteilung $f(x)$
  - von Neumannsche Verwerfungsmethode (“acceptance-rejection method”)
  - Transformationsmethode
- Spezialfall: z. B. gaußverteilte Zufallszahlen

Die Verwerfungsmethode

- Gesucht: Zufallszahlen mit Verteilung $f(x)$
- Erzeuge Zufallszahl x, gleichverteilt in $[x_{\min}, x_{\max}]$, d. h. $x = x_{\min} + u_1(x_{\max} - x_{\min})$, wobei u_1 gleichverteilt in $[0, 1]$.
- Generiere zweite, unabhängige Zufallszahl gleichverteilt zwischen 0 und $f_{\max}$, d. h. $y = u_2 \cdot f_{\max}$ (wobei u_2 gleichverteilt in $[0, 1]$)
- Wenn $y < f(x)$, akzeptiere x als Zufallszahl; falls nicht, wiederhole
  - niedrige Effizienz, vor allem wenn f schlecht durch Rechteck angenähert

entspricht Integration

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

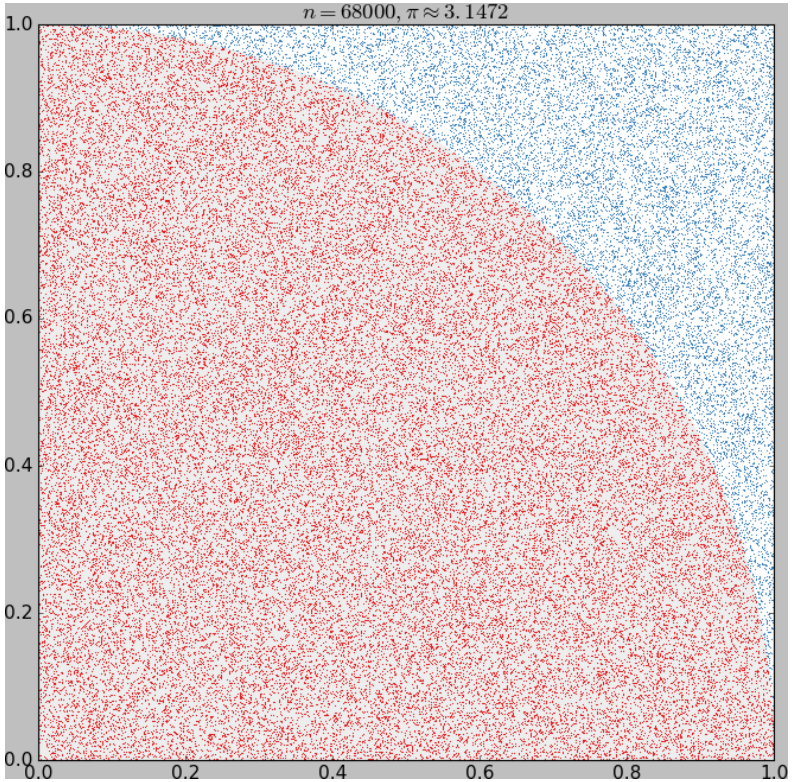


Die Verwerfungsmethode – Beispiel

Beispiel von oben:
Berechnung der Zahl π

Integration von $f(x) = \sqrt{1 - x^2}$ im Intervall $[0, 1]$

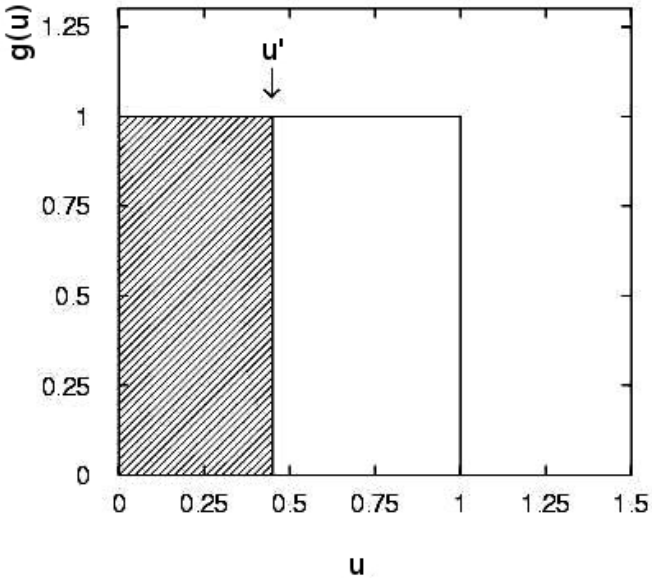
Akzeptiere Punktepaar, wenn
$x^2 + y^2 <= 1$
Bilde Verhältnis von
“Innerhalb/Gesamt”
$\pi = 4 \cdot$ Innerhalb/Gesamt

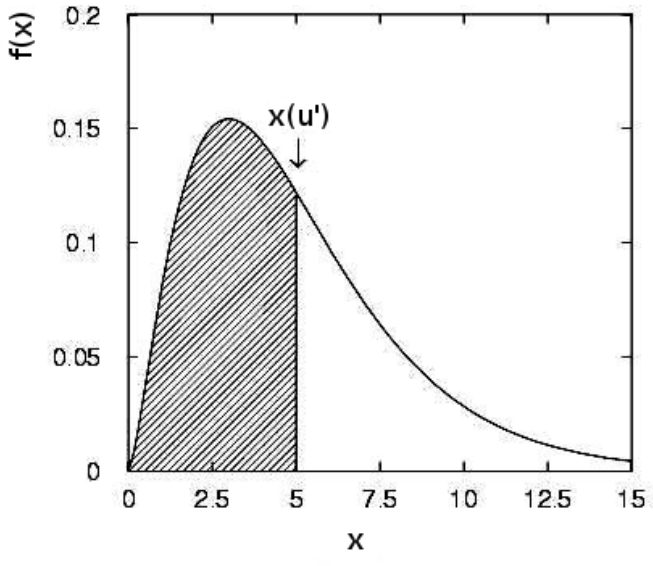


Die Transformationsmethode (I)

- Gegeben: $u_1, ..., u_n$ gleichförmig verteilt in $[0, 1]$
- Gesucht:

  Zufallszahlen $x(u)$, die Wahrscheinlichkeitsdichtefunktion $f(x)$ folgen



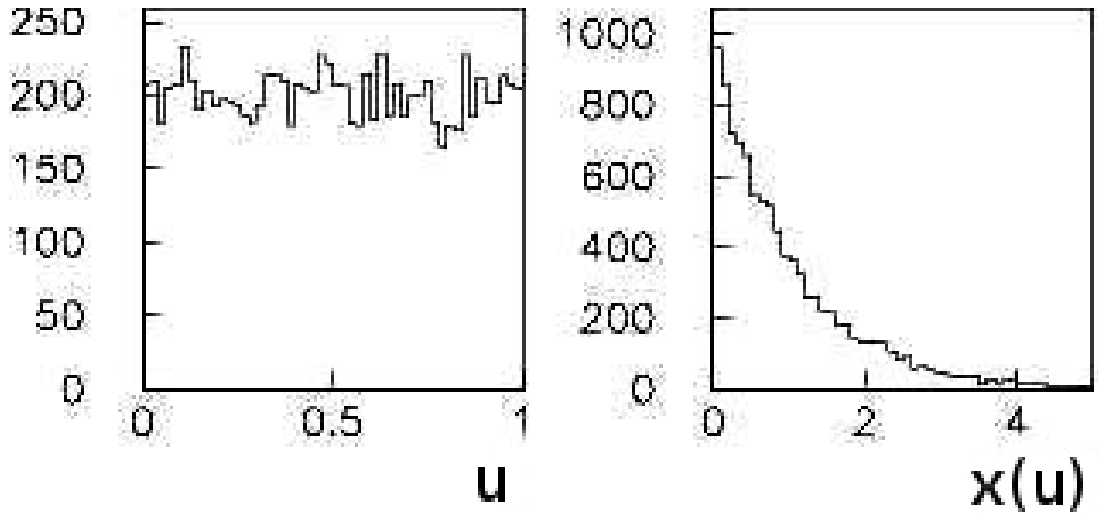


- PDF $f(x) \Rightarrow$ CDF $F(x) : x \rightarrow [0, 1]$
  - mit kumulativer Verteilungsfunktion (CDF) $F(x) \doteq \int_{-\infty}^{x} f(t)\mathrm{d}t$
- Umkehrfunktion $x(u) = F^{-1}(u) : [0, 1] \rightarrow$ Definitionsmenge(f)
  - Zufallszahlen $x_j = F^{-1}(u_i)$ folgen konstruktionsgemäß der PDF $f(x)$

Die Transformationsmethode (II)

Beispiel:

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



- DEMO: `inversion.C` ($\lambda = 2$)
- beachte: große x entsprechen kleinen u
  - $\Rightarrow$ Auflösung von u_j limitiert x_j

Allgemeine Vss. für Anwendbarkeit der Transformationsmethode:

- Integral $F(x)$ der Verteilung $f(x)$ muss bekannt und invertierbar sein.

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

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:
  $x_i = \sum_{j=1}^{12} u_j - 6$
  - Nachteile:
    - 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`)

Spezialfälle

- nicht immer sinnvoll, von gleichförmiger Verteilung auszugehen
- gaußverteilte Zufallszahlen sind ein Beispiel
- gibt spezialisierte Algorithmen für viele weitere Fälle wie z. B.
  - gaußverteilte Zufallszahlen in n Dimensionen, ggf. korreliert
  - Poissonverteilung, χ^2-Verteilung, Cauchy-Verteilung, …
  - zufällige Winkelverteilungen
- Vorteile: effizienter durch
  - Vermeidung aufwendiger Rechenoperationen
  - sinnvolle Näherungen

Effizienz von Monte-Carlo-Berechnungen

Vergleiche Genauigkeit von Monte-Carlo mit anderen Methoden

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

Für 1-dimensionales Integral:

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

Für N-dimensionales Integral:

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

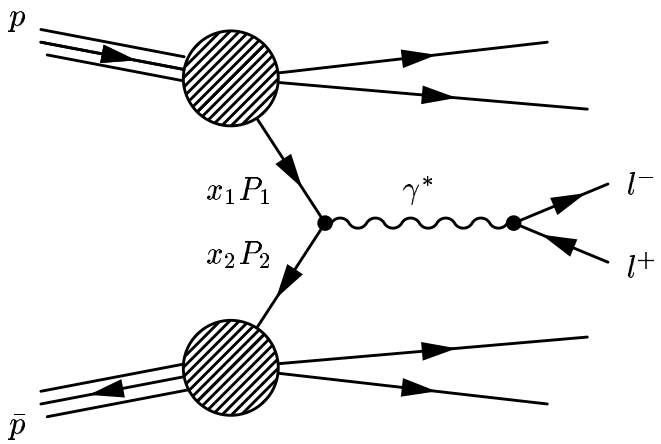
Übersicht

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

Monte-Carlo-Ereignis-Generatoren

Einfache Reaktion:

$$p\bar{p} \to Z/\gamma^* \to e^+e^-$$

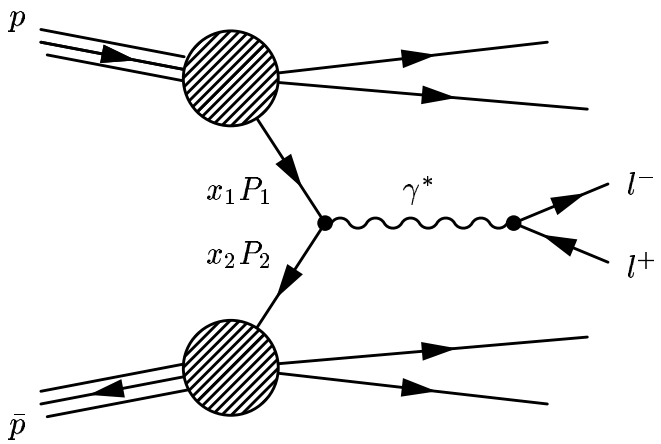


- Ereignis-Generatoren werden benutzt, um Teilchenreaktionen zu simulieren
- z. B. PYTHIA, HERWIG++, SHERPA, ALPGEN, POWHEG, ...
- Ausgabe: “Kollisionsereignisse”, d. h. für jedes Ereignis wird eine Liste von Teilchen generiert zusammen mit den Vierervektoren, etc.

Monte-Carlo-Ereignis-Generatoren

Einfache Reaktion:

$$p\bar{p} \to Z/\gamma^* \to e^+e^-$$



- Ereignis-Generatoren werden benutzt, um Teilchenreaktionen zu simulieren
- z. B. Pythia, Herwig++, Sherpa, Alpgen, PowHeg, ...
- Ausgabe: “Kollisionsereignisse”, d. h. für jedes Ereignis wird eine Liste von Teilchen generiert zusammen mit den Vierervektoren, etc.
- allg. kurz “Ereignis” (event) = Menge aller Detektordaten, die einer Kollision von zwei Protonenbündeln zugeordnet werden
- Detektordaten: entweder “echte” Daten oder (wie hier) 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        t
    1          2   -1     0     0   501     0      0.000      0.000   2832.442   2832.442      0.000    0.0
    2         -1   -1     0     0     0   501      0.000      0.000    -60.411     60.411      0.000    0.0
    3    1000024    1     1     2     0     0     21.435    218.452    526.439    622.751    250.000    0.0
    4    1000023    1     1     2     0     0    -21.435   -218.452   2245.592   2270.102    250.000    0.0

--------  End LHA event information and listing  ------------------------------------------------------
```

$$u + \bar{d} \to \widetilde{\chi}_1^+ + \widetilde{\chi}_2^0$$

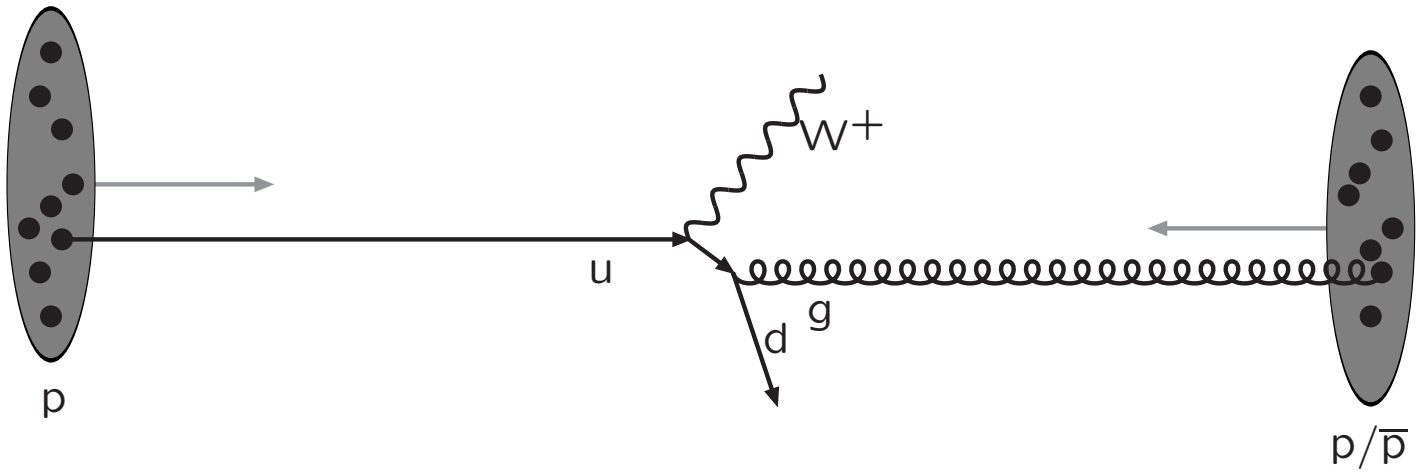
Monte-Carlo Event Generation: Steps

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

Incoming beams: parton densities

Slide: Torbjorn Sjostrand

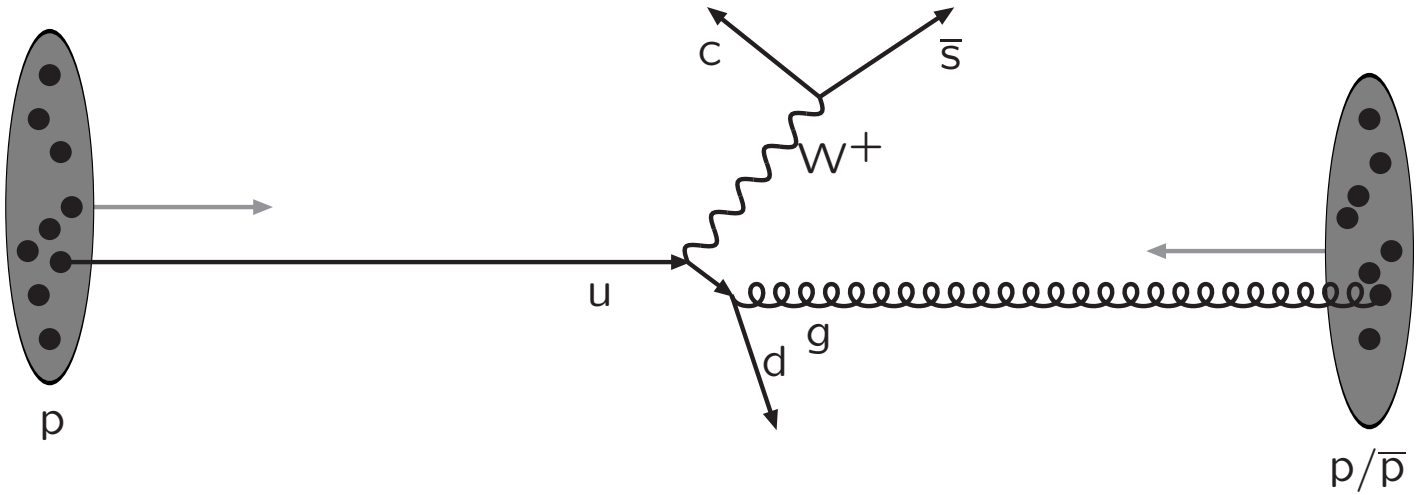
Monte-Carlo Event Generation: Steps



Hard subprocess: described by matrix elements

Slide: Torbjorn Sjostrand

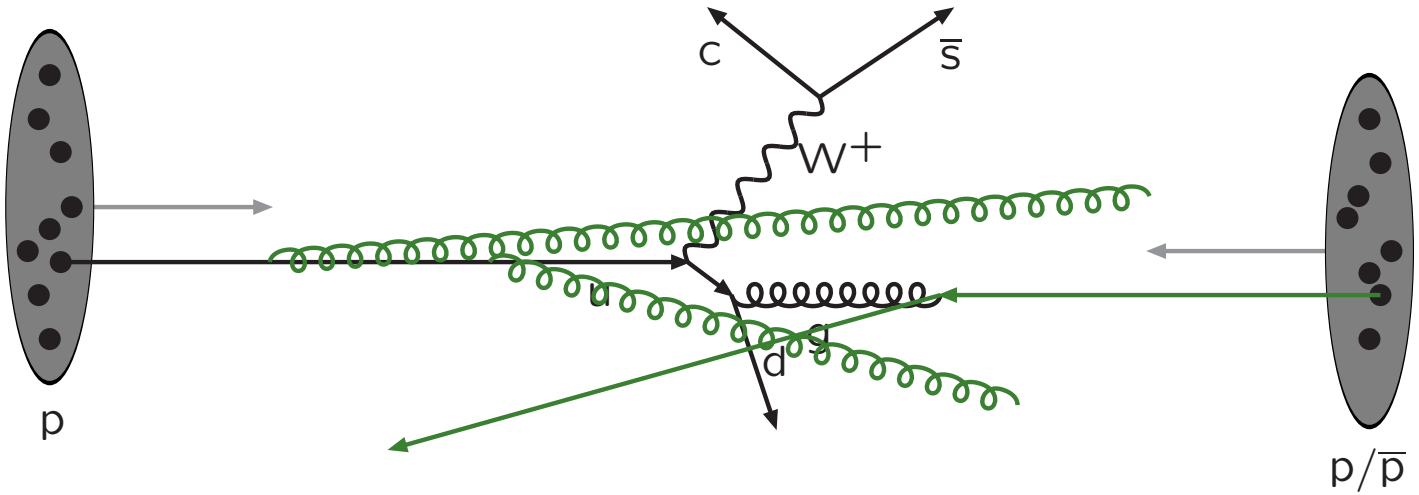
Monte-Carlo Event Generation: Steps



Resonance decays: correlated with hard subprocess

Slide: Torbjorn Sjostrand

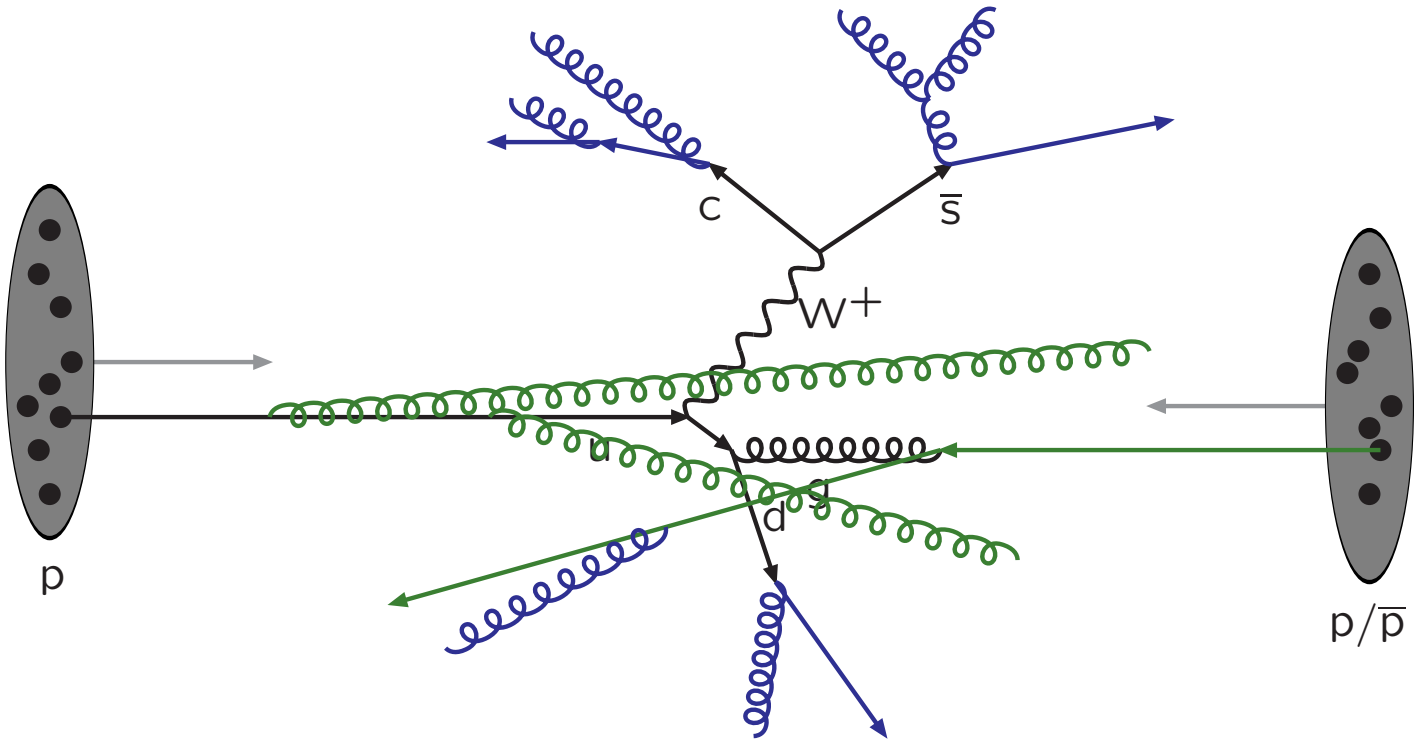
Monte-Carlo Event Generation: Steps



Initial-state radiation: spacelike parton showers

Slide: Torbjorn Sjostrand

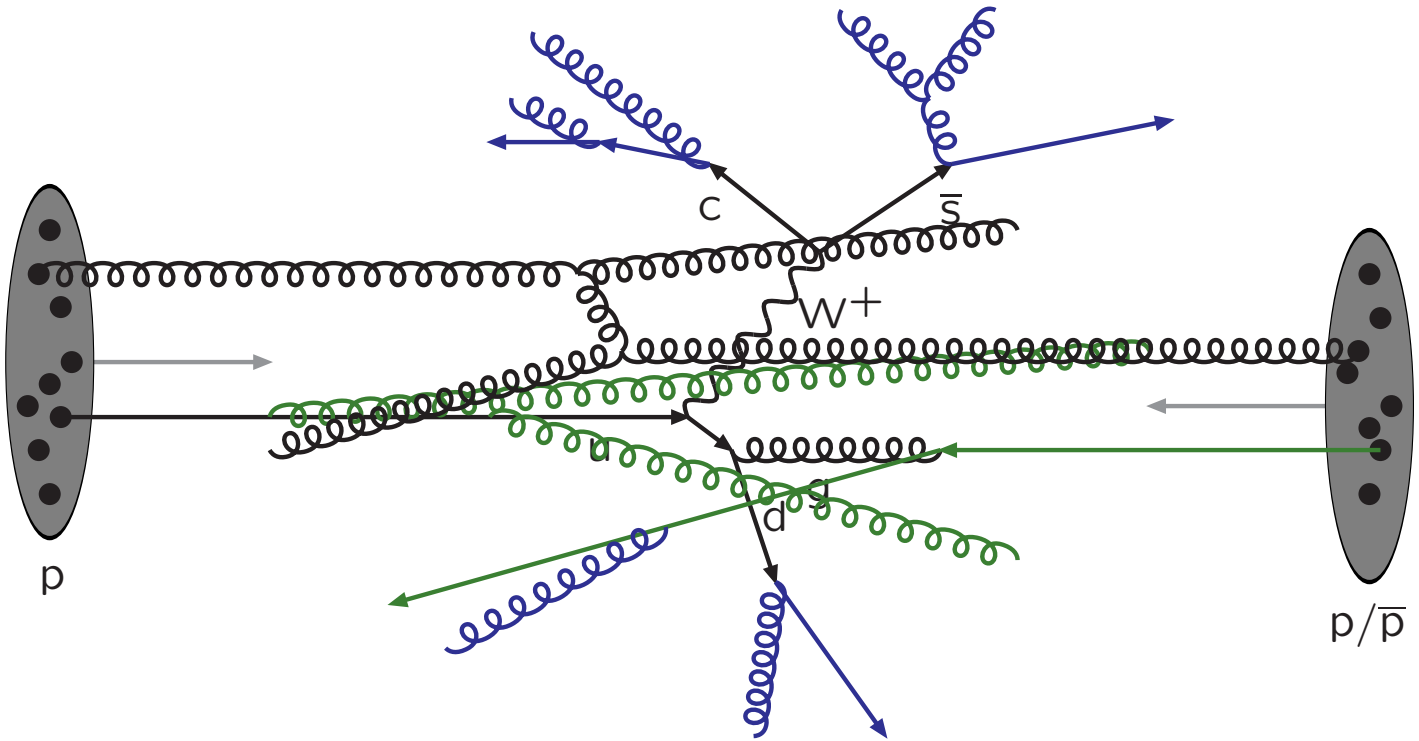
Monte-Carlo Event Generation: Steps



Final-state radiation: timelike parton showers

Slide: Torbjorn Sjostrand

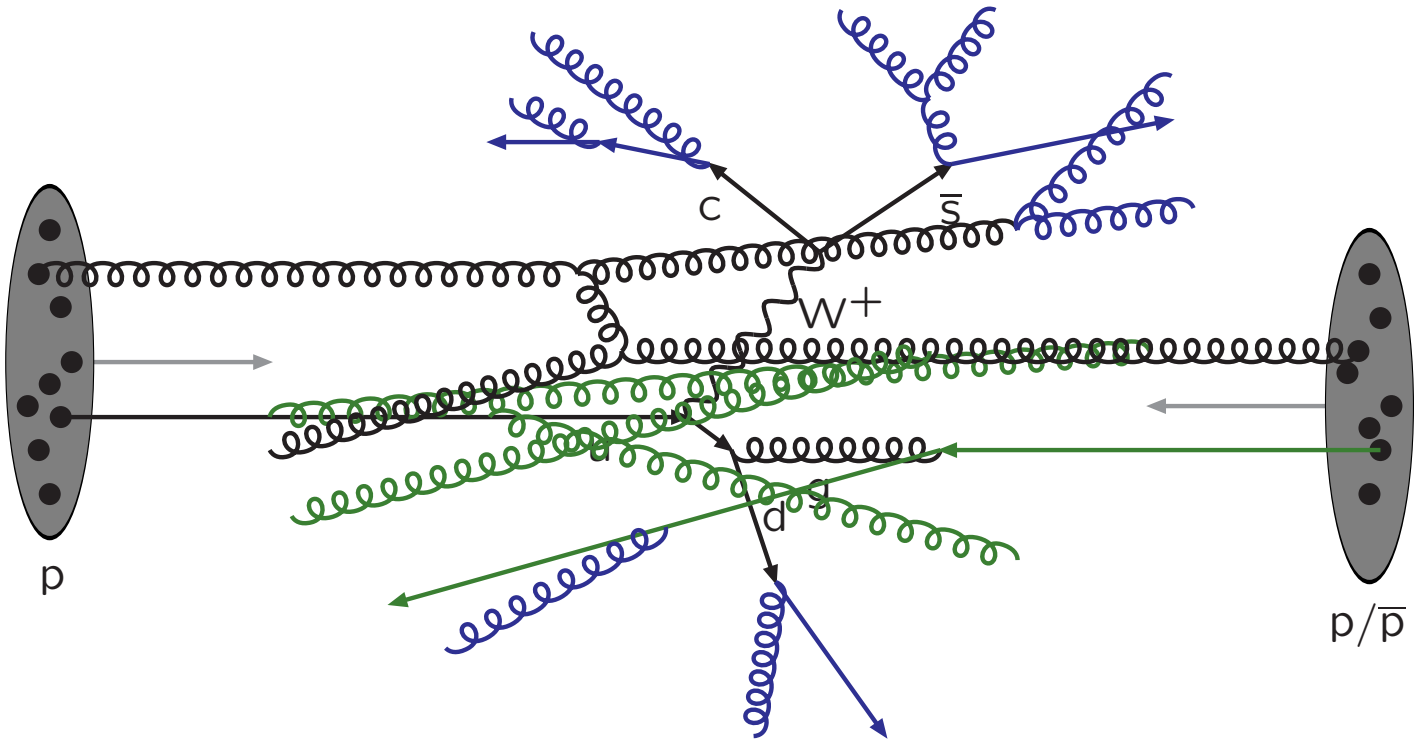
Monte-Carlo Event Generation: Steps



Multiple parton–parton interactions . . .

Slide: Torbjorn Sjostrand

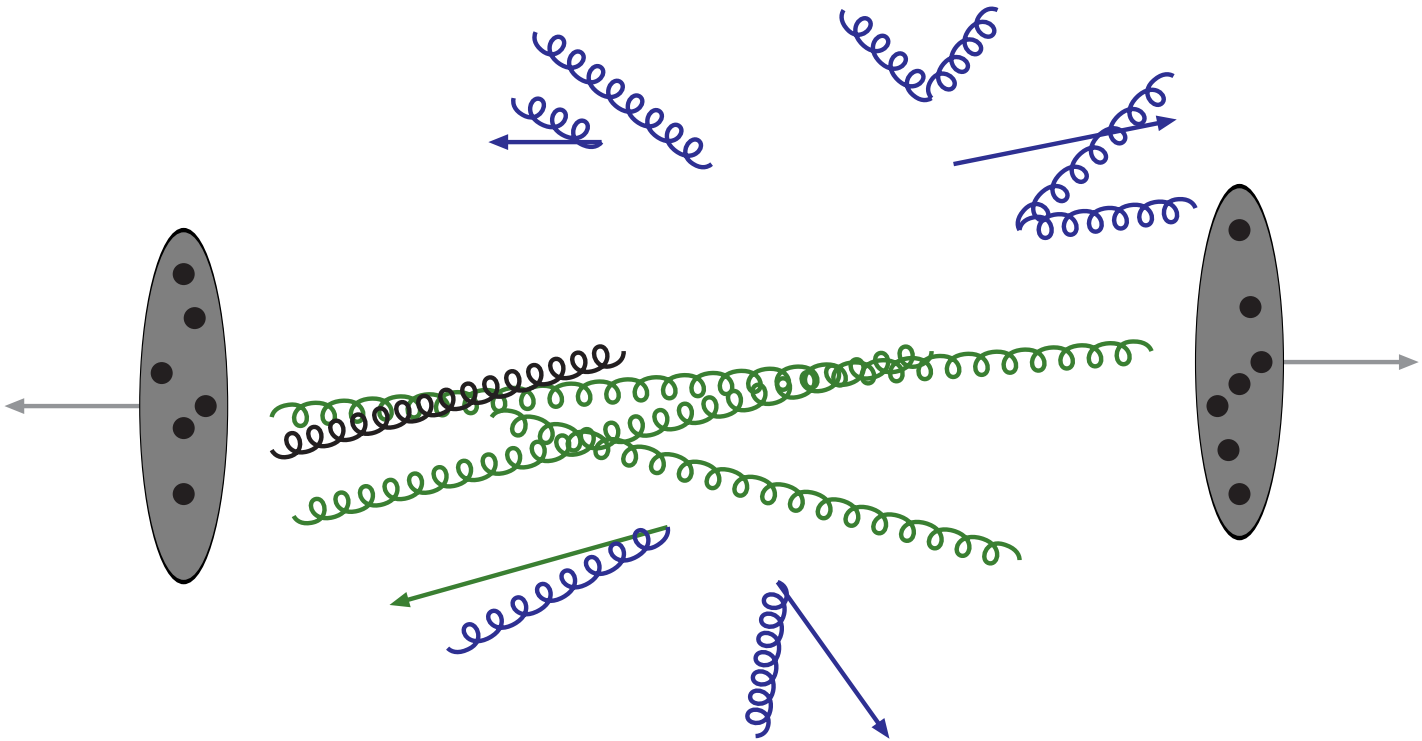
Monte-Carlo Event Generation: Steps



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

Slide: Torbjorn Sjostrand

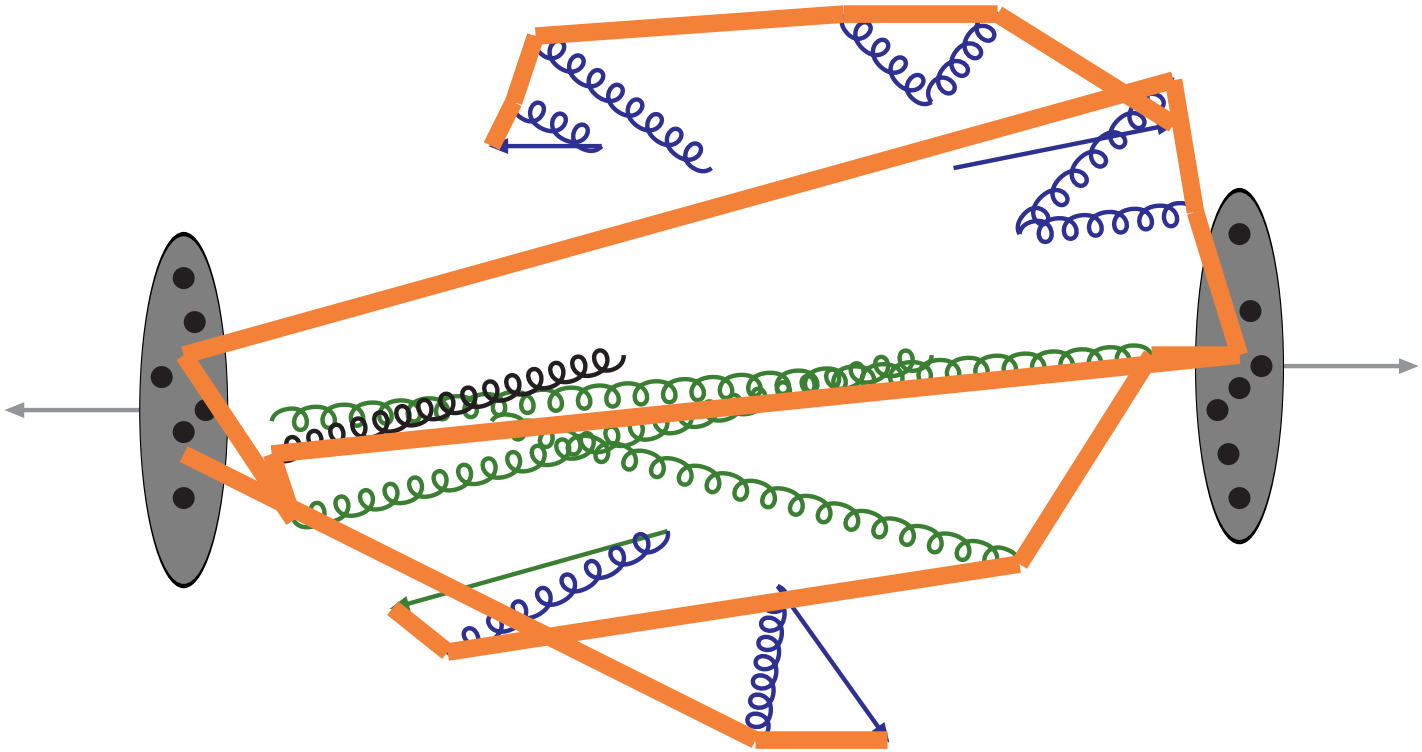
Monte-Carlo Event Generation: Steps



Beam remnants and other outgoing partons

Slide: Torbjorn Sjostrand

Monte-Carlo Event Generation: Steps

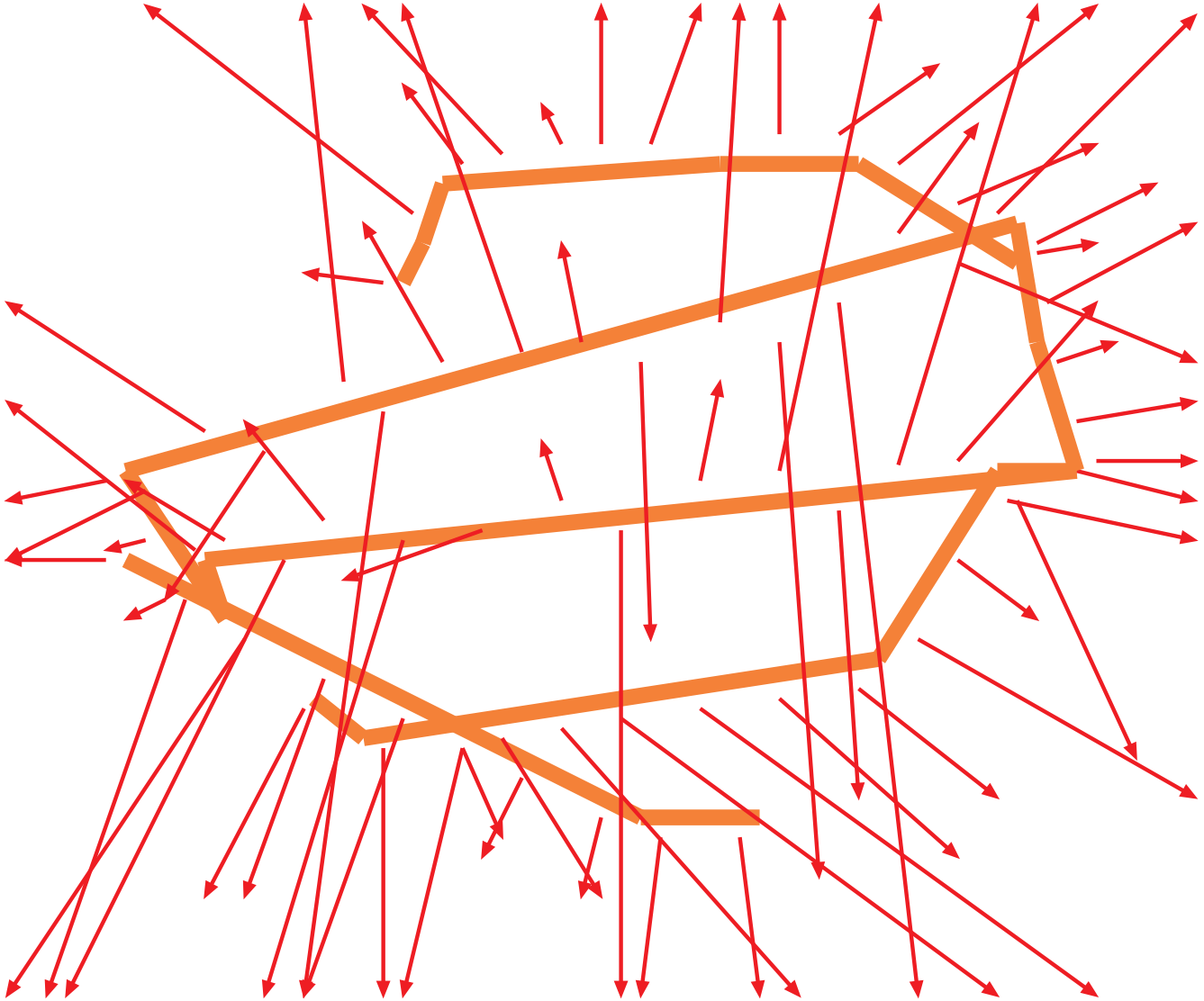


Everything is connected by colour confinement strings

Recall! Not to scale: strings are of hadronic widths

Slide: Torbjorn Sjostrand

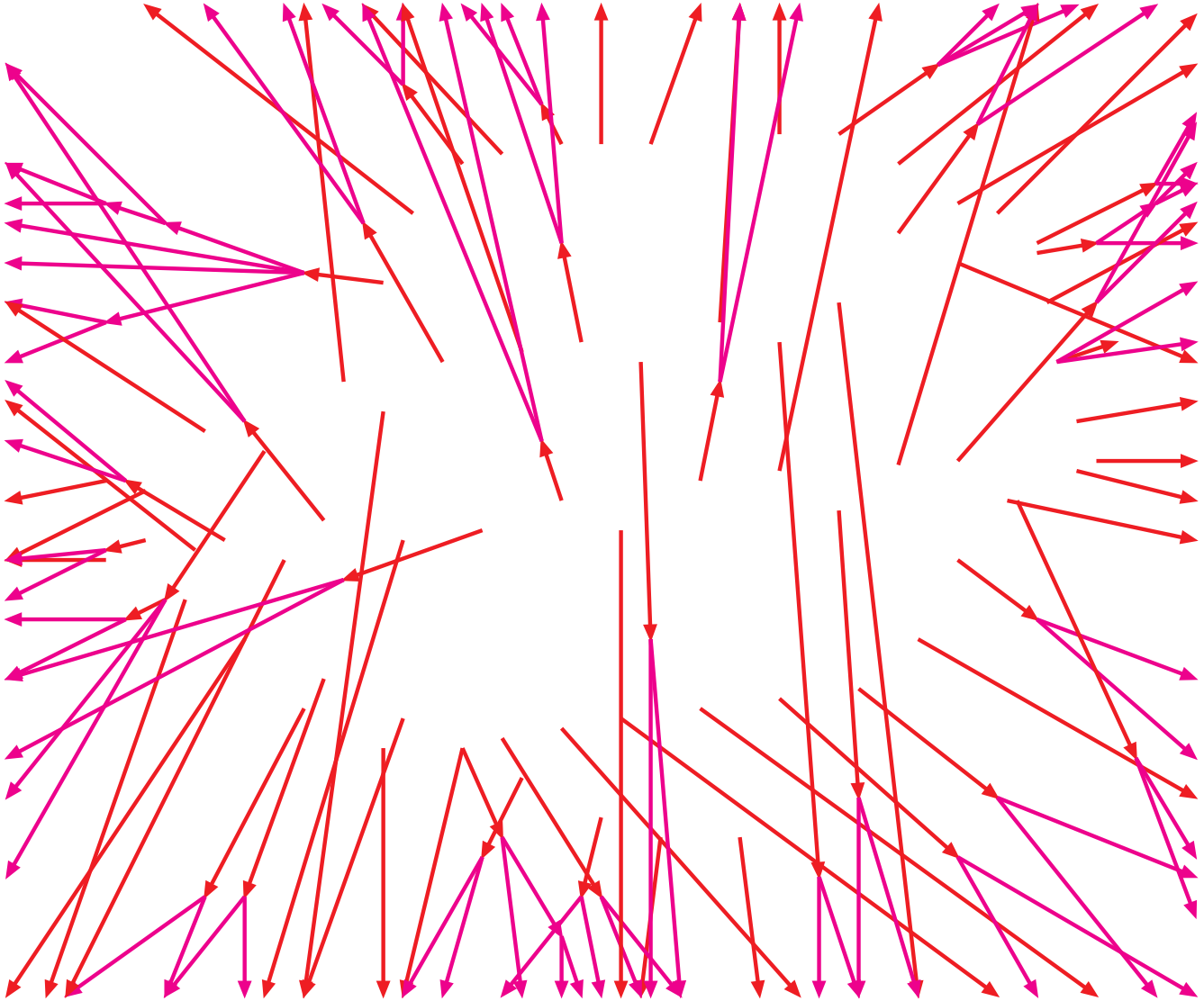
Monte-Carlo Event Generation: Steps



The strings fragment to produce primary hadrons

Slide: Torbjorn Sjostrand

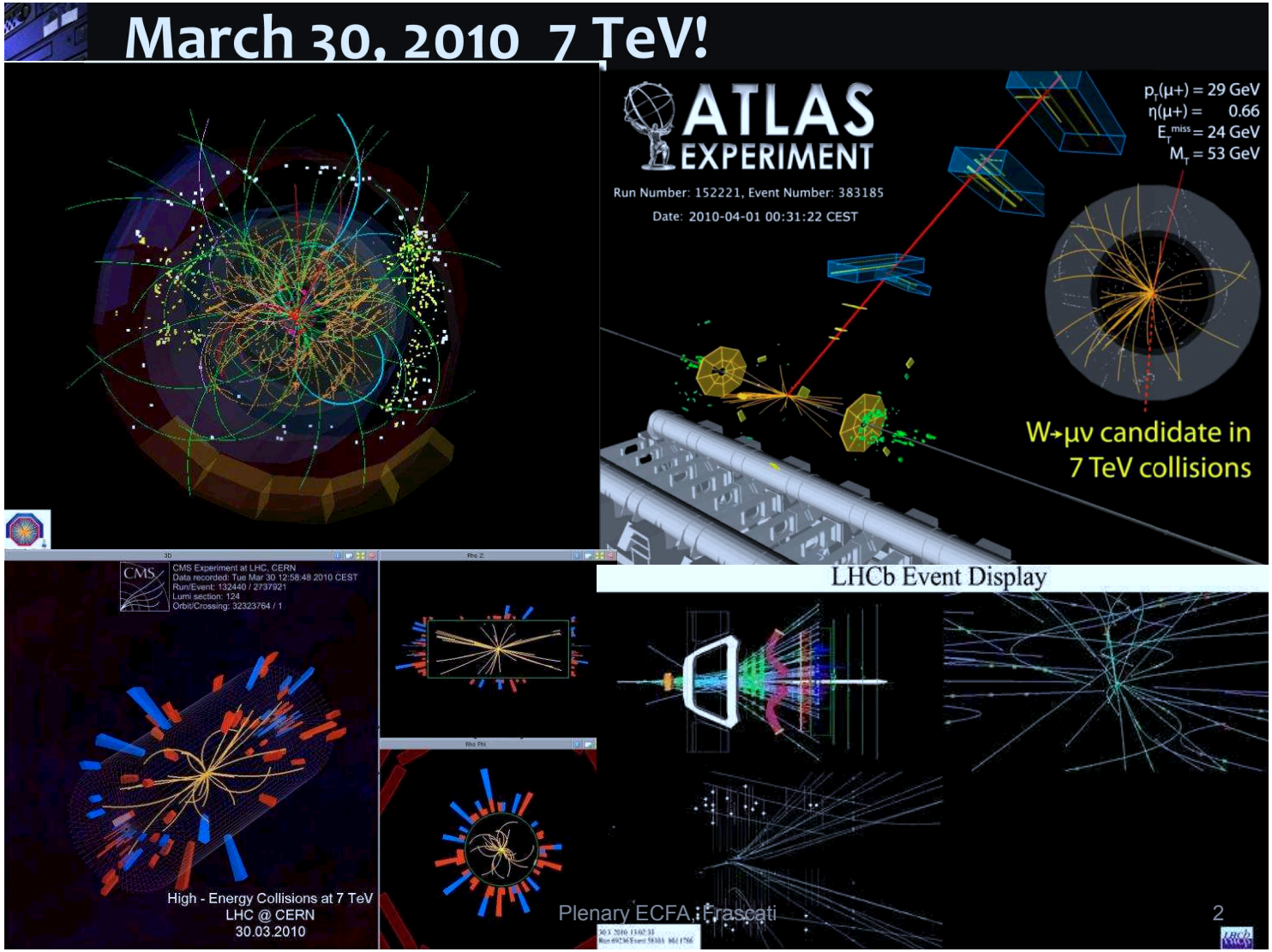
Monte-Carlo Event Generation: Steps



Many hadrons are unstable and decay further

Slide: Torbjorn Sjostrand

Monte-Carlo Event Generation: Steps



These are the particles that hit the detector

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        e          m
     0        90   (system)          -11     0     0     0     0     0     0      0.000      0.000      0.000   8000.000   8000.000
     1      2212   (p+)              -12     0     0    62     0     0     0      0.000      0.000   4000.000   4000.000      0.938
     2      2212   (p+)              -12     0     0    63     0     0     0      0.000      0.000  -4000.000   4000.000      0.938
     3         2   (u)               -21     7     7     5     6   501     0      0.000      0.000   2832.442   2832.442      0.000
     4        -1   (dbar)            -21     8     0     5     6     0   501      0.000      0.000    -60.411     60.411      0.000
     5   1000024   (~chi_1+)         -22     3     4     9     9     0     0     21.435    218.452    526.439    622.751    250.000
     6   1000023   (~chi_20)         -22     3     4    10    10     0     0    -21.435   -218.452   2245.592   2270.102    250.000
     7         2   (u)               -42    41    41     3     3   501     0     -0.000      0.000   2832.442   2832.442      0.000
     8        -1   (dbar)            -41    40    40    11     4     0   503      0.000     -0.000   -188.076    188.076      0.000
     9   1000024   (~chi_1+)         -44     5     5    43    43     0     0     25.711    213.290    515.821    612.148    250.000
    10   1000023   (~chi_20)         -44     6     6    44    44     0     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    19    14    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   506     -3.818      1.424     -9.162     10.028      0.000
    15        21   (g)               -33    12    13    18    18   505   507      3.818     -1.424     19.534     19.955      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     -1.985     -8.405      8.644      0.000
    30        21   (g)               -44    24    24    56    56   509   504     -2.795      1.322    -31.046     31.200      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)               -51    29     0    35    36   510   508      1.386     -1.799     -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      3.306     -1.221     16.910     17.273      0.000
    35        21   (g)               -51    32     0    55    55   510   511      0.422     -1.705     -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   501   512     -1.091      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    62     7     7   501     0     -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   -219.788   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     -4.027      5.814   -116.702    116.916      0.000
    47        21   (g)               -43    42     0    68    68   503   513     -0.408      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)               -52    33    33    58    58   507   510     -0.844     -0.204      0.018      0.869      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     -1.310      2.800      1.037      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)               -44    30    30    74    74   509   504     -2.795      1.324    -31.052     31.206      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      0.865      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     -0.084     -0.369      2.348      2.378      0.000
    62         2   (u)               -61     1     0    41    41   504     0     -0.592      0.805   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     0     25.798    213.056    514.977    611.359    250.000
    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   512   503     -3.776      5.068   -116.725    116.896      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

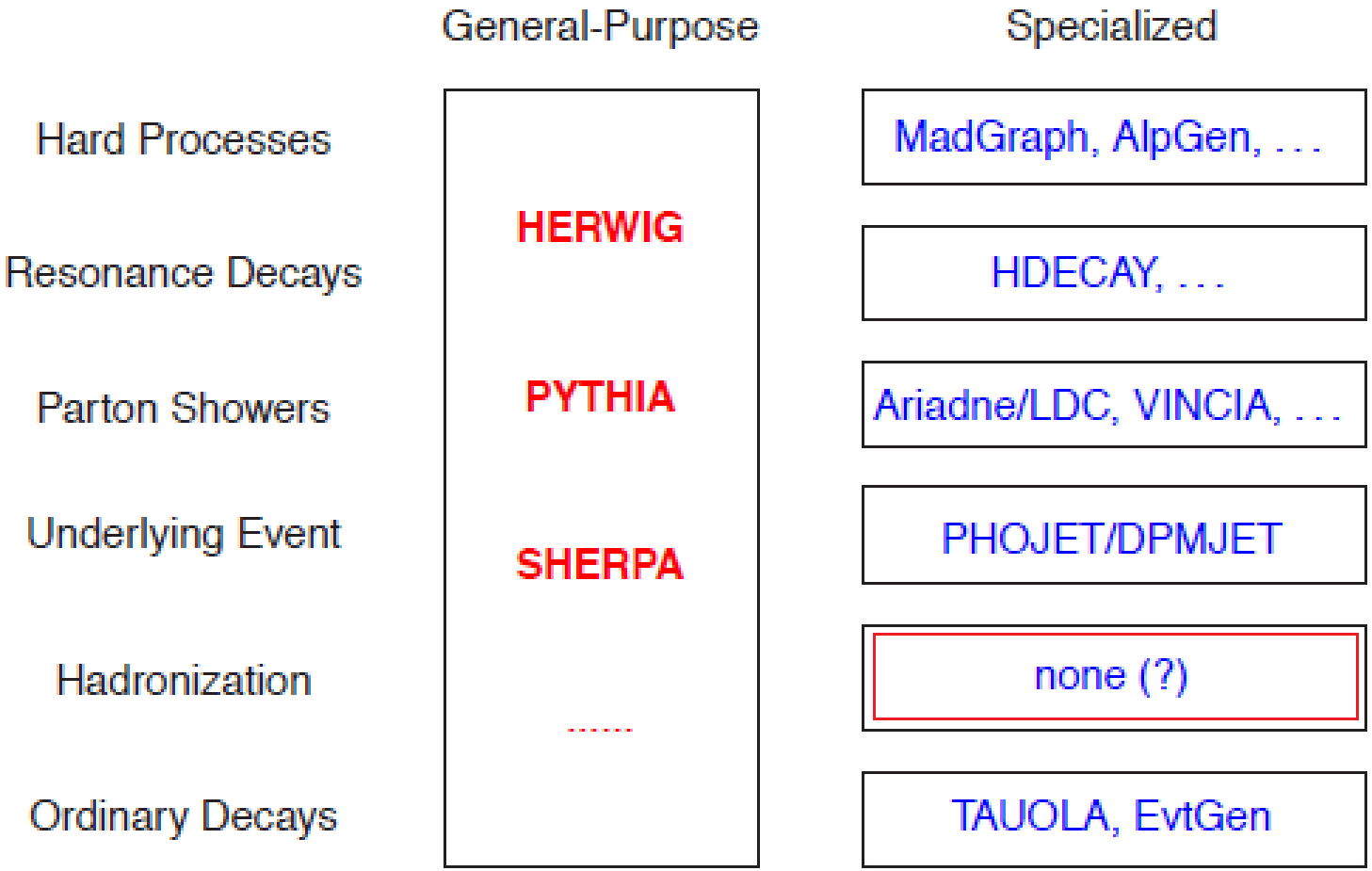
150	111	(pi0)	-83	129	138	241	242	0	0	0.016	-0.348	-1.427	1.475	0.135
151	-211	pi-	83	129	138	0	0	0	0	-0.713	0.092	-8.936	8.966	0.140
152	111	(pi0)	-83	129	138	243	244	0	0	-0.730	0.018	-6.471	6.514	0.135
153	2212	p+	83	129	138	0	0	0	0	0.146	-0.262	-3.643	3.774	0.938
154	-2212	pbar-	83	129	138	0	0	0	0	0.002	-0.378	-3.757	3.890	0.938
155	211	pi+	84	129	138	0	0	0	0	0.096	0.236	0.254	0.386	0.140
156	-211	pi-	84	129	138	0	0	0	0	0.022	0.034	-0.651	0.667	0.140
157	111	(pi0)	-84	129	138	245	246	0	0	0.473	0.554	-0.572	0.936	0.135
158	111	(pi0)	-84	129	138	247	248	0	0	0.076	0.007	0.592	0.612	0.135
159	211	pi+	84	129	138	0	0	0	0	-0.945	0.649	-0.623	1.312	0.140
160	-211	pi-	84	129	138	0	0	0	0	0.400	-0.003	0.444	0.613	0.140
161	111	(pi0)	-84	129	138	249	250	0	0	-0.544	0.692	0.279	0.934	0.135
162	221	(eta)	-84	129	138	251	253	0	0	0.132	0.622	0.674	1.077	0.548
163	213	(rho+)	-84	129	138	205	206	0	0	-0.021	-0.167	1.093	1.321	0.723
164	221	(eta)	-84	129	138	254	256	0	0	0.821	-0.503	3.688	3.851	0.548
165	111	(pi0)	-84	129	138	257	258	0	0	0.302	0.326	0.641	0.792	0.135
166	-211	pi-	84	129	138	0	0	0	0	0.093	-0.494	4.701	4.730	0.140
167	111	(pi0)	-84	129	138	259	260	0	0	1.988	-0.556	4.496	4.949	0.135
168	211	pi+	84	129	138	0	0	0	0	-0.301	0.076	0.851	0.916	0.140
169	-211	pi-	84	129	138	0	0	0	0	-0.097	-0.038	0.198	0.264	0.140
170	213	(rho+)	-84	129	138	207	208	0	0	0.494	0.095	2.126	2.313	0.762
171	223	(omega)	-84	129	138	261	263	0	0	-0.074	-0.733	-0.931	1.411	0.762
172	2	(u)	-71	75	75	174	187	506	0	1.129	0.455	-605.178	605.179	0.330
173	2103	(ud_1)	-71	80	80	174	187	0	506	-0.672	-0.594	1145.015	1145.016	0.771
174	213	(rho+)	-83	172	173	209	210	0	0	0.819	0.624	-324.280	324.283	0.827
175	111	(pi0)	-83	172	173	264	265	0	0	0.143	-0.327	-166.885	166.885	0.135
176	111	(pi0)	-83	172	173	266	267	0	0	0.638	-0.266	-95.690	95.693	0.135
177	111	(pi0)	-83	172	173	268	269	0	0	-0.227	0.597	-10.469	10.489	0.135
178	-211	pi-	84	172	173	0	0	0	0	-0.175	-0.348	-0.982	1.066	0.140
179	213	(rho+)	-84	172	173	211	212	0	0	0.031	-0.002	-3.682	3.750	0.710
180	-213	(rho-)	-84	172	173	213	214	0	0	-0.506	0.187	-2.946	3.097	0.786
181	213	(rho+)	-84	172	173	215	216	0	0	0.474	-0.011	3.125	3.235	0.691
182	111	(pi0)	-84	172	173	270	271	0	0	-0.054	-0.655	9.355	9.379	0.135
183	-211	pi-	84	172	173	0	0	0	0	0.044	0.546	11.291	11.305	0.140
184	211	pi+	84	172	173	0	0	0	0	-0.146	-0.116	3.783	3.790	0.140
185	-211	pi-	84	172	173	0	0	0	0	0.136	0.647	60.209	60.212	0.140
186	2224	(Delta++)	-84	172	173	217	218	0	0	-0.253	-1.201	595.259	595.261	1.141
187	-211	pi-	84	172	173	0	0	0	0	-0.467	0.186	461.750	461.750	0.140
188	211	pi+	91	118	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	38.554	99.810	317.062	334.629	0.135
190	-211	pi-	91	119	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	1.792	4.868	15.271	16.129	0.135
192	2212	p+	91	122	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	0.231	0.216	0.862	0.929	0.140
194	211	pi+	91	123	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	0.252	0.085	0.991	1.036	0.140
196	310	K_S0	91	125	125	0	0	0	0	0.282	-0.250	0.413	0.749	0.498

Monte-Carlo-Ereignis-Generatoren

Beispiel

264	22	gamma	91	175	0	0	0	0	0	0.004	-0.108	-30.867	30.868	0.000
265	22	gamma	91	175	0	0	0	0	0	0.139	-0.219	-136.017	136.018	0.000
266	22	gamma	91	176	0	0	0	0	0	0.247	-0.106	-46.600	46.601	0.000
267	22	gamma	91	176	0	0	0	0	0	0.391	-0.160	-49.090	49.092	0.000
268	22	gamma	91	177	0	0	0	0	0	-0.080	0.072	-1.807	1.810	0.000
269	22	gamma	91	177	0	0	0	0	0	-0.147	0.525	-8.662	8.679	0.000
270	22	gamma	91	182	0	0	0	0	0	-0.067	-0.169	2.172	2.179	0.000
271	22	gamma	91	182	0	0	0	0	0	0.013	-0.486	7.184	7.200	0.000
272	22	gamma	91	189	0	0	0	0	0	31.604	81.710	259.501	273.891	0.000
273	22	gamma	91	189	0	0	0	0	0	6.950	18.099	57.561	60.738	0.000
274	22	gamma	91	191	0	0	0	0	0	1.298	3.466	11.072	11.674	0.000
275	22	gamma	91	191	0	0	0	0	0	0.493	1.402	4.199	4.454	0.000
276	22	gamma	91	200	0	0	0	0	0	0.200	0.244	-1.796	1.824	0.000
277	22	gamma	91	200	0	0	0	0	0	0.104	0.016	-0.727	0.734	0.000
278	22	gamma	91	204	0	0	0	0	0	-0.185	0.244	-5.078	5.088	0.000
279	22	gamma	91	204	0	0	0	0	0	-0.216	0.552	-11.493	11.508	0.000
280	22	gamma	91	206	0	0	0	0	0	-0.075	0.141	0.469	0.496	0.000
281	22	gamma	91	206	0	0	0	0	0	-0.036	0.096	0.090	0.136	0.000
282	22	gamma	91	208	0	0	0	0	0	0.146	0.024	0.439	0.463	0.000
283	22	gamma	91	208	0	0	0	0	0	0.499	0.106	0.888	1.025	0.000
284	22	gamma	91	210	0	0	0	0	0	0.293	0.232	-166.316	166.317	0.000
285	22	gamma	91	210	0	0	0	0	0	0.220	0.286	-122.073	122.074	0.000
286	22	gamma	91	212	0	0	0	0	0	0.119	-0.015	-2.768	2.770	0.000
287	22	gamma	91	212	0	0	0	0	0	0.041	0.060	-0.632	0.636	0.000
288	22	gamma	91	214	0	0	0	0	0	-0.106	0.244	-1.883	1.902	0.000
289	22	gamma	91	214	0	0	0	0	0	-0.023	0.098	-0.321	0.336	0.000
290	22	gamma	91	216	0	0	0	0	0	0.104	-0.041	0.759	0.767	0.000
291	22	gamma	91	216	0	0	0	0	0	0.068	-0.211	1.419	1.436	0.000
292	22	gamma	91	220	0	0	0	0	0	-0.174	-0.564	3.400	3.450	0.000
293	22	gamma	91	220	0	0	0	0	0	-3.225	-10.397	59.384	60.374	0.000
294	22	gamma	91	221	0	0	0	0	0	-2.664	-7.819	47.018	47.739	0.000
295	22	gamma	91	221	0	0	0	0	0	-0.791	-2.198	13.646	13.844	0.000
296	22	gamma	91	230	0	0	0	0	0	-0.072	0.196	-2.758	2.766	0.000
297	22	gamma	91	230	0	0	0	0	0	0.003	0.139	-0.921	0.931	0.000
298	22	gamma	91	231	0	0	0	0	0	0.041	0.070	-2.433	2.435	0.000
299	22	gamma	91	231	0	0	0	0	0	0.015	0.045	-0.214	0.219	0.000
300	22	gamma	91	232	0	0	0	0	0	0.008	0.033	-0.075	0.082	0.000
301	22	gamma	91	232	0	0	0	0	0	0.089	0.005	-1.321	1.324	0.000
302	22	gamma	91	237	0	0	0	0	0	-1.449	0.969	-38.188	38.227	0.000
303	22	gamma	91	237	0	0	0	0	0	-0.015	0.009	-0.814	0.814	0.000
304	22	gamma	91	251	0	0	0	0	0	0.017	0.190	0.065	0.201	0.000
305	22	gamma	91	251	0	0	0	0	0	0.030	0.209	0.247	0.325	0.000
306	22	gamma	91	252	0	0	0	0	0	0.026	0.171	0.049	0.180	0.000
307	22	gamma	91	252	0	0	0	0	0	0.006	0.049	0.122	0.132	0.000
308	22	gamma	91	253	0	0	0	0	0	-0.035	-0.032	0.043	0.064	0.000
309	22	gamma	91	253	0	0	0	0	0	0.087	0.036	0.147	0.175	0.000
310	22	gamma	91	256	0	0	0	0	0	0.043	0.037	0.274	0.280	0.000
311	22	gamma	91	256	0	0	0	0	0	0.120	-0.088	0.322	0.355	0.000
312	22	gamma	91	263	0	0	0	0	0	0.015	-0.171	-0.139	0.221	0.000

Event generators



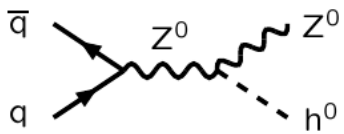
Specialized often best at given task, but need General-Purpose core

Slide: Else Lytken et al.

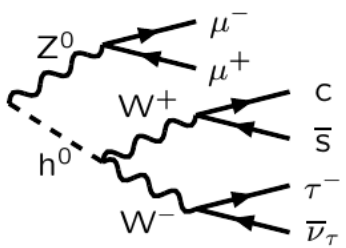
Monte Carlo generation

Matrix elements (ME):

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

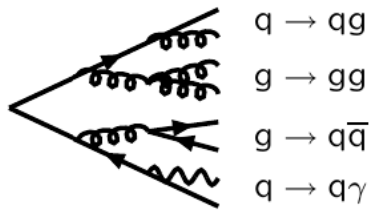


2) Resonance decays: includes correlations.

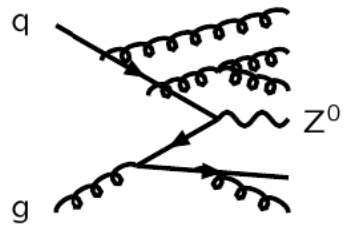


Parton Showers (PS):

3) Final-state parton showers.



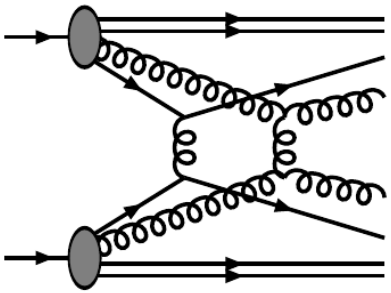
4) Initial-state parton showers.



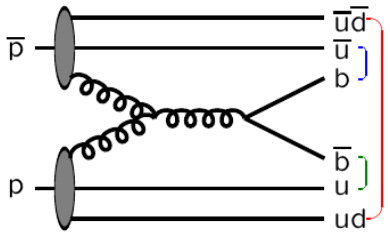
Slide: Else Lytken et al.

Monte Carlo generation

5) Multiple parton–parton interactions.

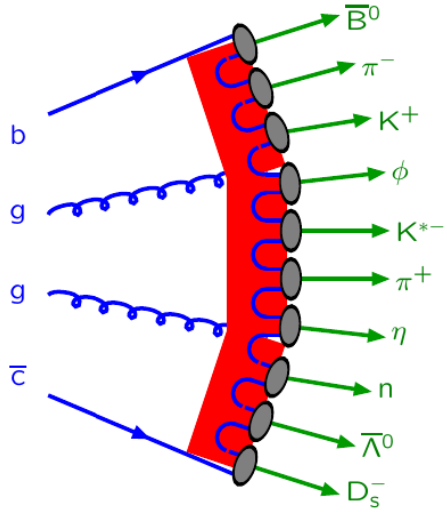


6) Beam remnants, with colour connections.

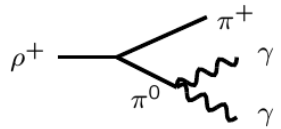


5) + 6) = Underlying Event

7) Hadronization



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



Slide: Else Lytken et al.

Monte-Carlo Event Generation

Generatoren bei Belle 2

- $e^+e^- \rightarrow Y(4S) \rightarrow B\overline{B}$
  Anfangszustand genau bekannt
  → Generation startet mit Y(4S)-Teilchen
  → Zerfallskette simuliert mit EvtGen-Software
  (Verzweigungsverhältnisse und Zerfallsmodelle)

- $e^+e^- \rightarrow q\overline{q}$
  Simulation von Fragmentation
  ähnlich wie bei ATLAS

- $e^+e^- \rightarrow \ell^+\ell^-$
  Spezielle Generatoren
  basierend auf QED-Rechnungen

```
300553 (Upsilon(4S))
    521 (B+)
      443 (J/psi)
        211 (pi+)
       -211 (pi-)
        211 (pi+)
       -213 (rho-)
         -211 (pi-)
          111 (pi0)
             22 (gamma)
             22 (gamma)
      323 (K*+)
        311 (K0)
          310 (K_S0)
            211 (pi+)
           -211 (pi-)
        211 (pi+)
   -521 (B-)
      423 (D*0)
        421 (D0)
         -321 (K-)
          211 (pi+)
         22 (gamma)
       15 (tau-)
         13 (mu-)
        -14 (anti-nu_mu)
         16 (nu_tau)
      -16 (anti-nu_tau)
```

Slide: Thomas Kuhr

Übersicht

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

Monte-Carlo-Detektorsimulation

- Detektorsimulation:
  - erhält Teilchenliste aus Ereignis-Generator als Eingabe
  - simuliert Durchgang aller Teilchen durch Detektorkomponenten
    - Coulombstreuung (simuliert Streuwinkel)
    - Teilchenzerfälle (simuliert Lebensdauer)
    - Ionisierungsenergie (simuliert ΔE)
    - Elektromagnetische / hadronische Schauer
  - schlussendlich: Signale in Detektorausleseelektronik
- 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* using MC methods, initiated 1994, CERN)
  - verwendet von ATLAS, CMS, ALICE, LHCb, ILC, . . . , Astrophysikern, für klinische Studien, für Simulation von Strahlungsgefahr für Astronauten, in der Mikroelektronik, . . .

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 detailierte MC Simulationen benötigt

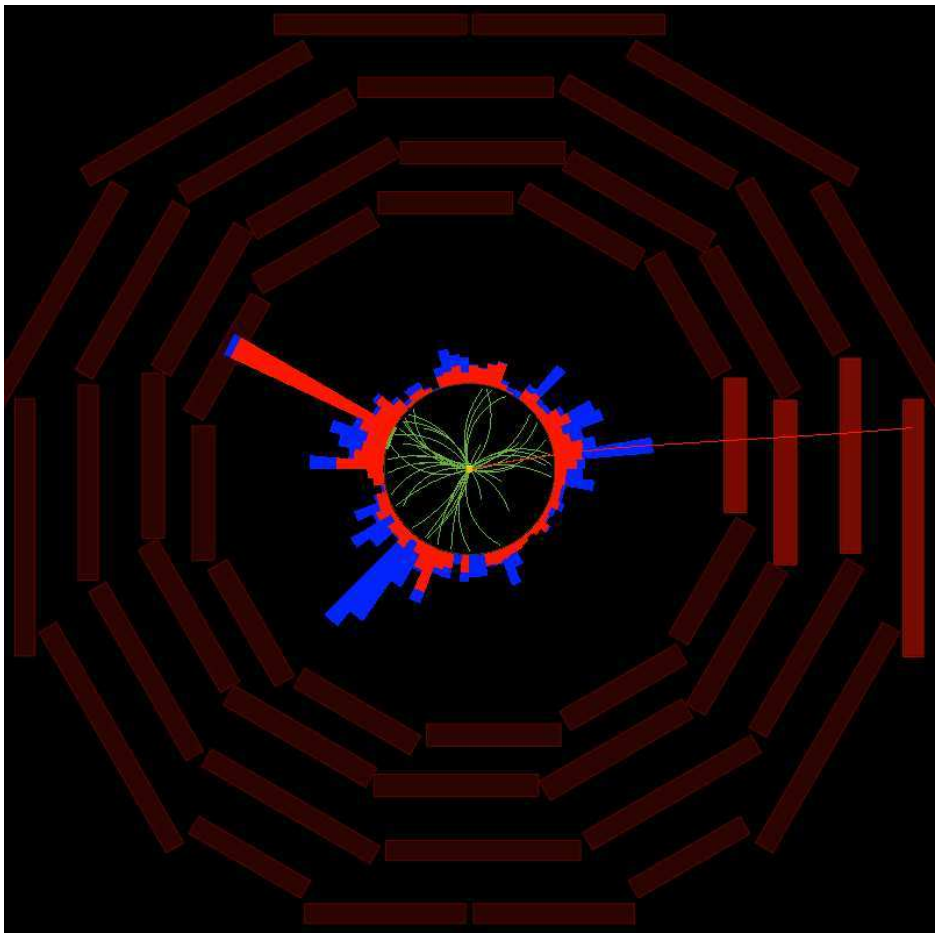
- Simulation der physikalischen Reaktion:
  alle enstehenden Teilchen und deren erwartete Energie-, Impuls- und Winkelverteilungen
- Nachweiswahrscheinlichkeit für jedes Detektorelement
- Orts- und Energieauflösung jeder einzelnen Detektorkomponente

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

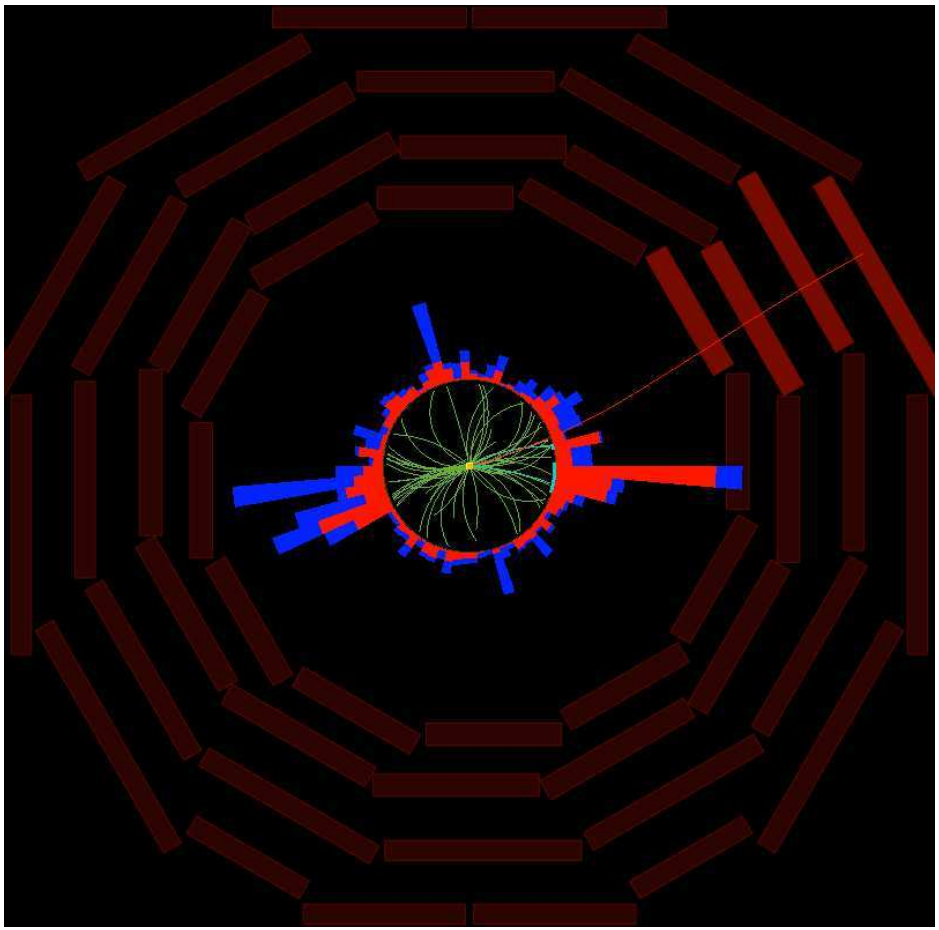
Komplexes Beispiel zur Monte Carlo Methode

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

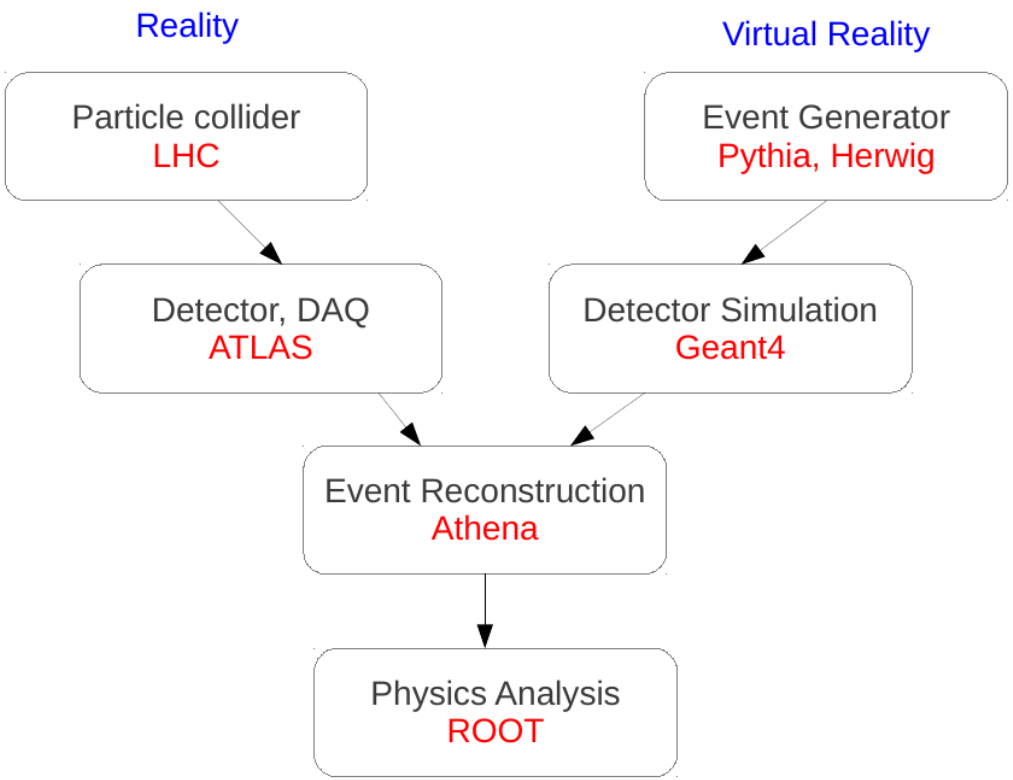


Komplexes Beispiel zur Monte Carlo Methode

CMS Experiment am LHC Beschleuniger am CERN:
Im Vergleich zu einem realen Ereignisse in den Daten $\mathrm{pp} \to \mathrm{t\bar{t}} + X$



Monte Carlo (MC) - What is MC?

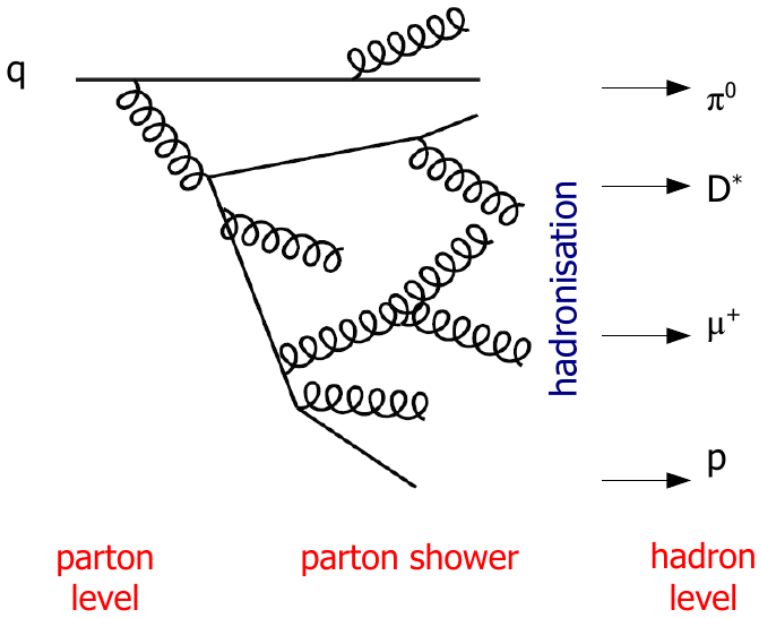


- MC simulates what happens at the *LHC* and *ATLAS*
- Many different programmes can be used at each stage

Slide: J. Morris

Monte-Carlo Event Generation: "Truth Record"

MC Generation



- MC Generator stops with set of "stable" final state particles
- Complete 4-vector info is known about every particle
- All parent-daughter relations are known and stored
- High energy parton state known as parton level
- Stable particle state known as hadron level

- This level of information is often called the truth record
- This is the pure event before it interacts with any apparatus

Slide: J. Morris

Monte-Carlo Event Generation: “Reconstruction”

Reconstruction

Going from electronic pulses to analysis objects

- Data and MC pass through the same reconstruction algorithms
- Raw electronic pulses reconstructed into:
  - Tracks
  - Calorimeter deposits
- Which are then reconstructed into:
  - Jets, electron, muons, taus,
  - Photons, tracks, missing E_{T}

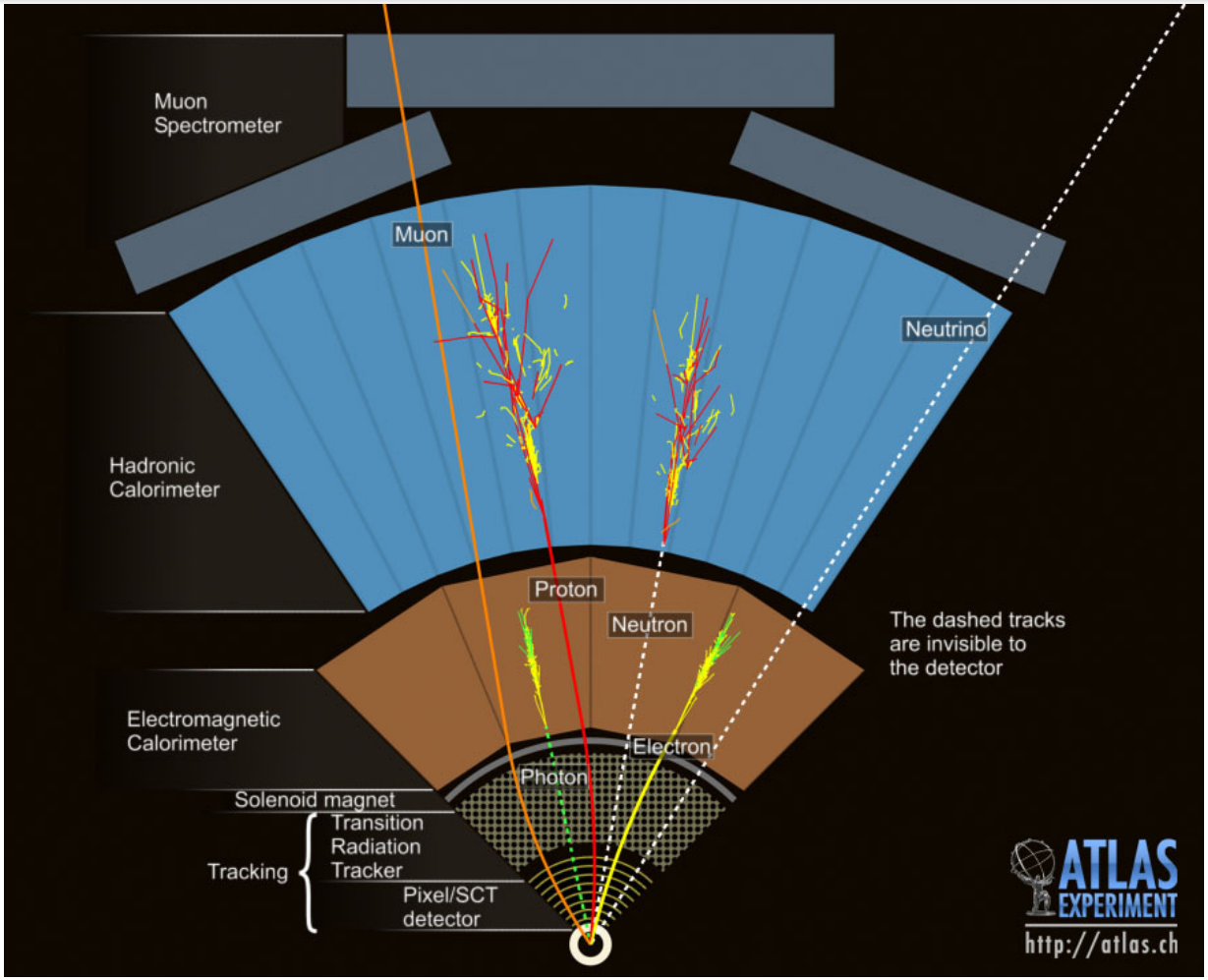
Real life issues need to be reflected in the MC

- Some parts of the detector become faulty over time
- e.g. - A section of the calorimeter readout dies and cannot be repaired until the detector is opened up in a shutdown
- Lets say that this affects x% of the data luminosity
- Need to generate MC with this problem in x% of the MC
  - Cannot know x until end of year
  - $\Rightarrow$ Need to reprocess the MC at the end of the year
- Some MC bugs do not become apparent for some time

Slide: J. Morris

Teilchenidentifikation in ATLAS

- Schematische Darstellung von Teilchenidentifikation in ATLAS
  (und ähnlich aufgebauten Detektoren)
- Tatsächliche Implementation = komplexe Algorithmen, oftmals mit ML



Übersicht

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

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
- MC-Simulation: Ereignisgenerator, Detektorsimulation, Rekonstruktion

Extra

Transformationsmethode

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

Transformationsmethode

Herleitung

- Generate a continuous random variable $X \sim F$ as follows :

> 1. Generate a uniform random variable U
> 2. Set $X = F^{-1}(U)$

- Proof: Have to show that the CDF of the samples produced by this method is precisely $F(x)$.

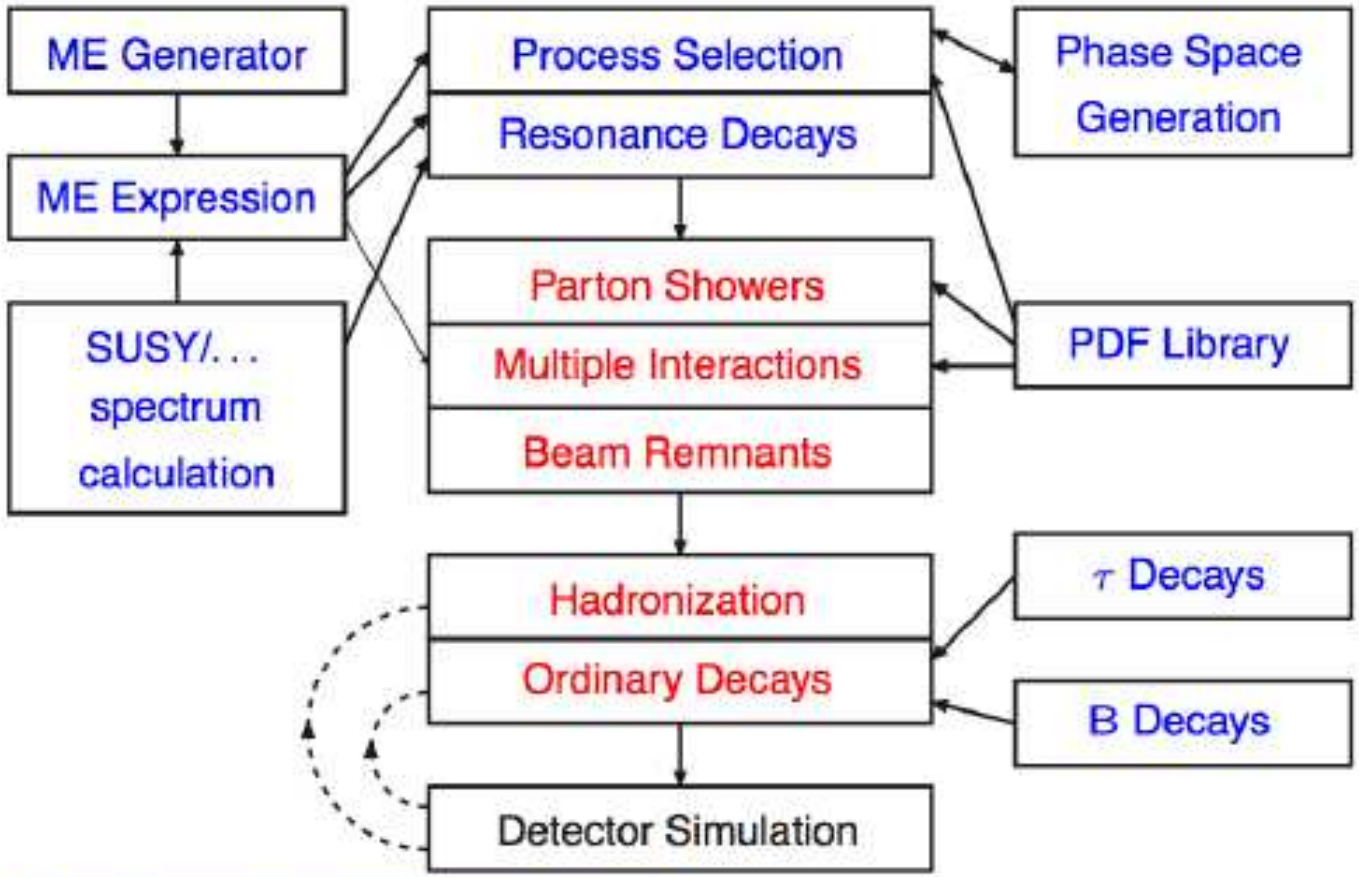
$$\begin{aligned} \mathbf{P}(X \leq x) &= \mathbf{P}(F^{-1}(U) \leq x) \\ &= \mathbf{P}(U \leq F(x)) && (1) \\ &= F(x) && (2) \end{aligned}$$

where

  - (1) follows by the fact that F is an increasing function
  - (2) follows from the fact $0 \leq F(x) \leq 1$ and the CDF of a uniform $F_U(y) = y$ for all $y \in [0, 1]$

Quelle

What they do



Several standardized interfaces!

Slide: Else Lytken et al.