Karlsruhe Institute of Technology

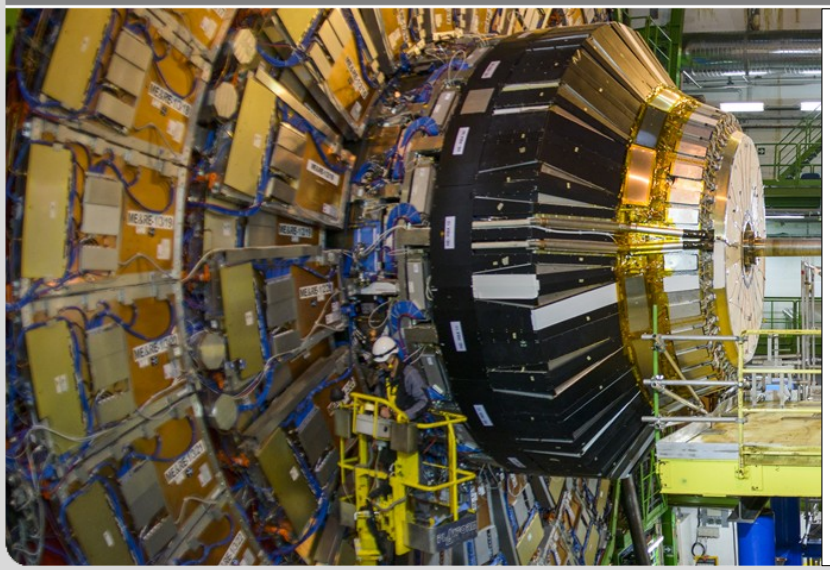
Rechnernutzung in der Physik

Teil 3 – Statistische Methoden in der Datenanalyse

Roger Wolf

8. Dezember 2015

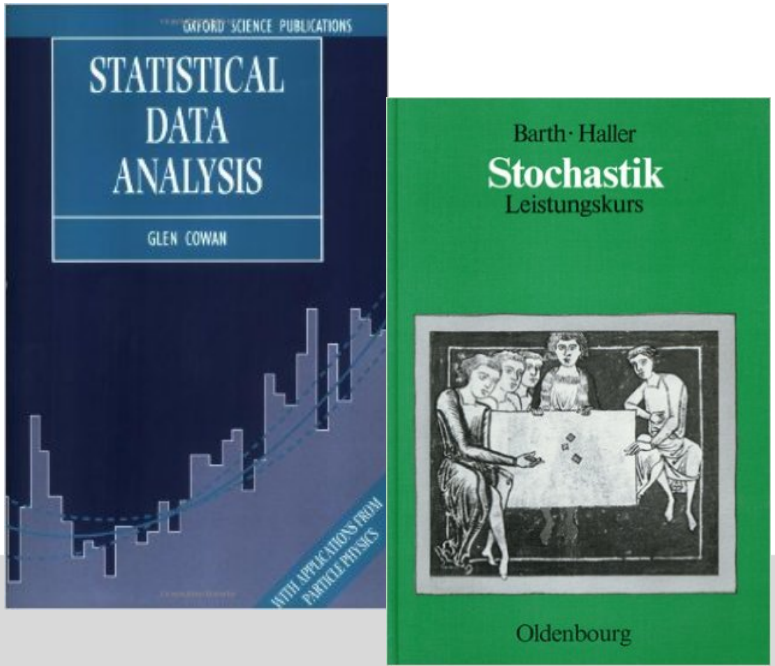
INSTITUTE OF EXPERIMENTAL PARTICLE PHYSICS (IEKP) – PHYSICS FACULTY



- Grundlagen der Wahrscheinlichkeitstheorie, Werkzeuge zur statistischen Datenanalyse
- Gängige Wahrscheinlichkeitsverteilungen
- Monte-Carlo Methoden
- Parameterschätzung
- Hypothesentests

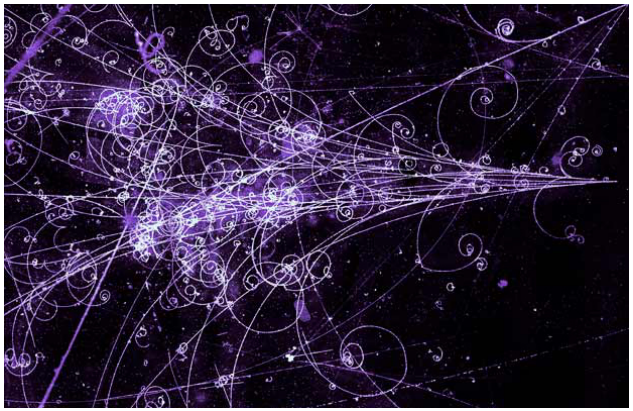
Literatur

- Glen Cowan: Statistical Data Analysis, Oxford (1997).
- G. Bohm, G. Zech: Einführung in die Statistik und Messwertanalyse für Physiker, DESY E-Buch (2006).
- V. Blobel, E. Lohrmann: Statistische und numerische Methoden der Datenanalyse, DESY E-Buch (2012).
- F. Barth, R. Haller: Stochastik, Oldenbourg/Ehrenwirth (1996).
- ...



Statistisk in der Teilchenphysik

- Im Experiment:
  - Alle Gesetzmäßigkeiten beruhen auf expliziten Ratenmessungen.
  - Z.B. am LHC zeichnen wir Billionen von Teilchenkollisionen auf.
  - All diese Kollisionen sind perfekt statistisch unabhängig!



- In der Theorie:
  - Wellenfunktionen in der Quantenmechanik werden explizit als Wahrscheinlichkeitsdichtefunktionen interpretiert.
  - Das Matrixelement $\mathcal{S}_{fi}$ gibt die Wahrscheinlichkeit den Endzustand f aus dem Anfangszustand i vorzufinden.
  - Eine Messung setzt sich aus gut verstandenen, unabhängigen statistischen Prozessen zusammen: $pdf \rightarrow ME \rightarrow hadronization \rightarrow energy\ loss\ in\ material \rightarrow digitization.$
  - Erlaubt *event-by-event* Simulation teilchenphysikalischer Vorgänge.

Perfektes Anwendungsgebiet statistischer Methoden!

Kapiel 3.1:

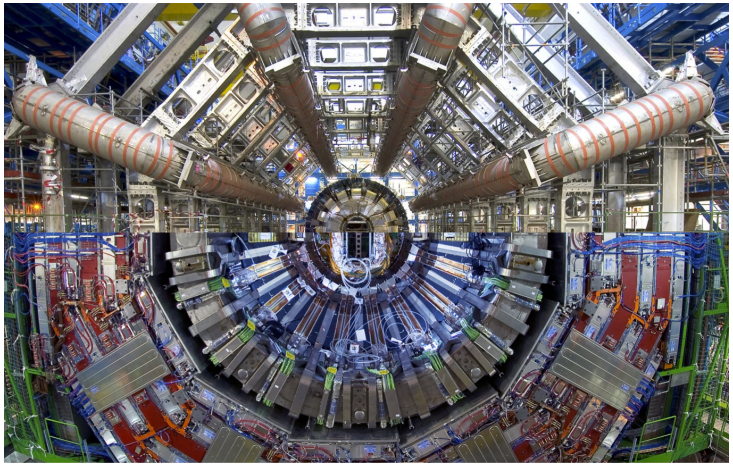
Grundlagen der Wahrscheinlichkeits-theorie

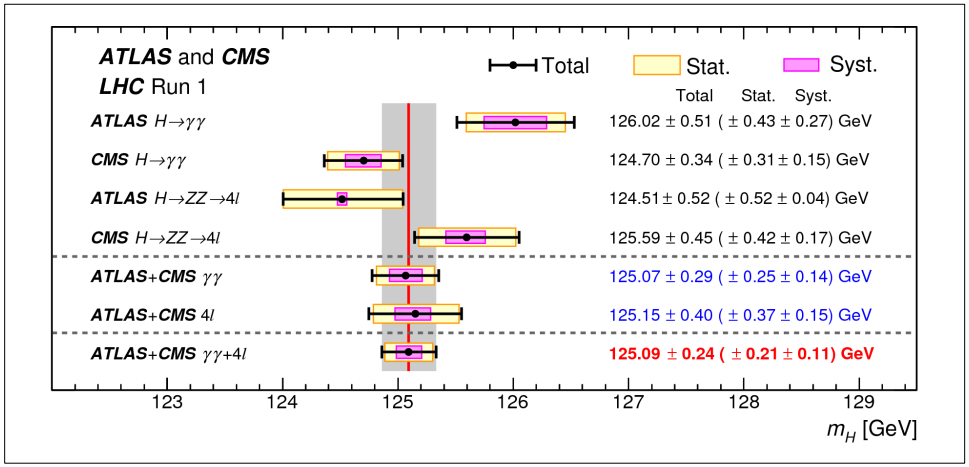
Meßunsicherheiten in der Physik?

- In der Naturwissenschaft werden Erkenntnisse durch Experimente gewonnen und daraus gezogene Schlußfolgerungen durch Experimente überprüft.[(1)]

- Zentral bei der Auswertung der Experimente ist das Konzept der (Meß-)Unsicherheit.

Prominentes Beispiel aus der Teilchenphysik: Messung der Masse der Higgs Bosons.





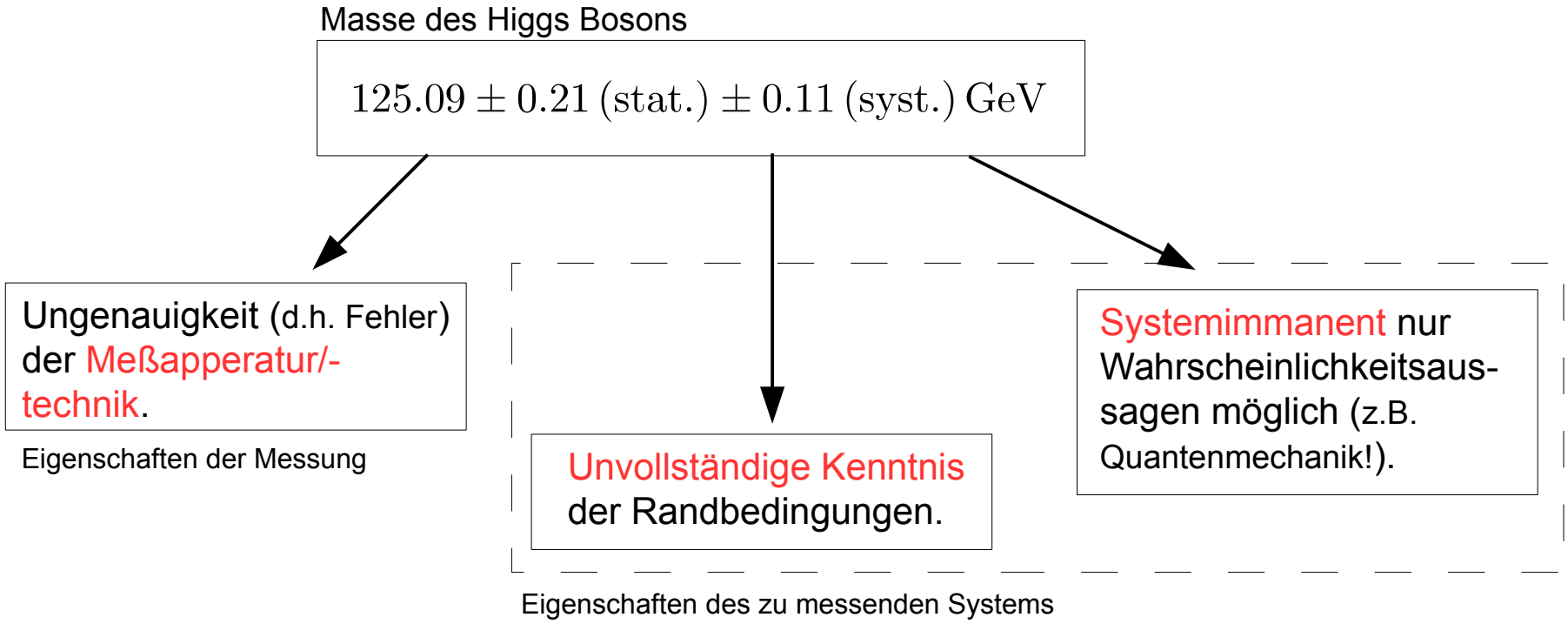
$$125.09 \pm 0.21\,(\text{stat.}) \pm 0.11\,(\text{syst.})\,\text{GeV}$$

→ Keine Messung ohne Unsicherheit!

(1) Barth/Haller: Stochastik (1996).

Quellen von Unsicherheiten & Rolle des Zufalls

- Unsicherheiten (d.h. Fehler) einer Messung können verschiedene Ursachen haben:



- Wir nennen eine Eigenschaft eines zu vermessenden Systems **zufallsverteilt** wenn sie nicht bekannt ist oder nicht mit exakter Genauigkeit vorhergesagt werden kann.

Quantifizierung des Zufalls

- Das Maß an Zufall läßt sich durch das Konzept der Wahrscheinlichkeit quantifizieren.
- Die Anfänge der mathematischen Wahrscheinlichkeitstheorie gehen zurück bis in die Renaissance im 15 Jhrd.[(2)]
- Der moderne Wahrscheinlichkeitsbegriff begründet sich auf die Mengentheoretische Formulierung von Kolmogorov von 1933.[(3)]

(2)
Luca Pacioli: *Summa de Arithmetica Geometrica et Proportionality* (1491).

(3)
Andrei Nikolajewitsch Kolmogorov (1903 – 1987).

Ergebnis- und Ereignisraum

Eine Menge $\Omega = \{\omega_1, \omega_2, \ldots \omega_n\}$ heißt **Ergebnisraum** eines Zufsallsexperiments, wenn jedem Versuchsausgang höchstens ein Element $\omega_i \in \Omega$ zugeordnet ist. Die ω_i heißen dann die Ergebnisse des Zufallsexperiments.

- Beispiele:
  - Münzwurf $\Omega = \{\mathrm{Kopf}, \mathrm{Zahl}\}$.
  - Würfelwurf $\Omega = \{1, 2, 3, 4, 5, 6\}$.
  - Bestimmung der Fallbeschleunigung (wie lautet der Ergebnisraum?).
  - Bestimmung der Anzahl radioaktiver Atome nach einer Zeitspanne Δt (wie lautet der Ergebnisraum?).
  - Zerfall eines τ-Leptons in der Teilchenphysik.

Jede Teilmenge $A \subset \Omega$ heißt **Ereignis**. A tritt genau dann ein, wenn sich ein Ergebnis ω_i einstellt, das in A enthalten ist. Die Menge aller Ereignisse heißt **Ereignisraum** $\mathfrak{P}(\Omega)$.(4)

(4) Beachte: Interpretation des Ereignisraums noch offen!

Wahrscheinlichkeitsverteilung (nach Kolmogorov)

Eine auf dem Ereignisraum $\mathfrak{P}(\Omega)$ definierte Funktion

$$\mathcal{P} : \mathfrak{P}(\Omega) \to \mathbb{R} \qquad ; \qquad A \to \mathcal{P}(A),$$

heißt **Wahrscheinlichkeitsverteilung über dem Ergebnisraum** Ω [5], wenn sie die folgenden Eigenschaften erfüllt:

- Für jedes Ereignis $A \in \mathfrak{P}(\Omega)$ gilt $\mathcal{P}(A) \geq 0$ (Nichtnegativität).
- Für die Wahrscheinlichkeit zweier disjunkter Ereignisse A und B ($A \cap B = \emptyset$) gilt: $\mathcal{P}(A \cup B) = \mathcal{P}(A) + \mathcal{P}(B)$ (Linearität).
- Die Wahrscheinlichkeit $\mathcal{P}(\mathfrak{P}(\Omega)) = 1$ (Normierungsbedingung).

(5)
Beachte: Interpretation der Wahrscheinlichkeitsverteilung!

Wahrscheinlichkeitsverteilung (nach Kolmogorov)

- Folgerungen (ohne Beweis, aber einfach zu beweisen):

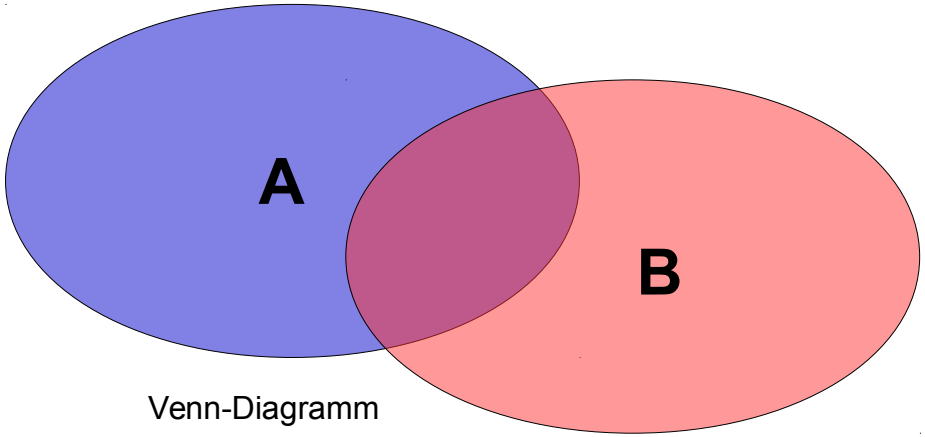
  $\mathcal{P}(\bar{A}) = 1 - \mathcal{P}(A)$ (komplementäres Ereignis).

  $\mathcal{P}(\emptyset) = 0$ (unmögliches Ereignis).

  $\mathcal{P}(A \cup B) = \mathcal{P}(A) + \mathcal{P}(B) - \mathcal{P}(A \cap B)$.

  $A \subset B$ impliziert $\mathcal{P}(A) \leq \mathcal{P}(B)$.

  $\mathcal{P}(A) \in [0; 1]$.



Venn-Diagramm

Bedingte Wahrscheinlichkeit

Sei $\mathcal{P}$ eine Wahrscheinlichkeitsverteilung über dem Ergebnisraum Ω, B ein Ereignis mit $\mathcal{P}(B) > 0$ und A ein beliebiges Ereignis. Dann heißt

$$\mathcal{P}_B(A) = \mathcal{P}(A|B) = \frac{\mathcal{P}(A \cap B)}{\mathcal{P}(B)}$$

die (bedingte) **Wahrscheinlichkeit von A unter Bedingung B**.

- Anmerkungen:
  - $\mathcal{P}_B(A)$ ist selbst eine Wahrscheinlichkeitsverteilung über Ω (die den Axiomen von Kolmogorov genügt).
  - $\mathcal{P}(A)$ kann selbst als bedingte Wahrscheinlichkeit $\mathcal{P}_S(A)$ betrachtet werden für $S = \mathfrak{P}(\Omega)$.
  - $\mathcal{P}(A \cap B) = \mathcal{P}_B(A) \cdot \mathcal{P}(B)$.

Satz von Bayes

Sei $\mathcal{P}$ eine Wahrscheinlichkeitsverteilung über dem Ergebnisraum Ω, A und B Ereignisse mit $\mathcal{P}(A) > 0$ und $\mathcal{P}(B) > 0$. Dann erhält man $\mathcal{P}_A(B)$ aus $\mathcal{P}_B(A)$ durch die Umformung[6]:

$$\mathcal{P}_A(B) = \frac{\mathcal{P}_B(A) \cdot \mathcal{P}(B)}{\mathcal{P}(A)}$$

$$\mathcal{P}_B(A) = \frac{\mathcal{P}(A \cap B)}{\mathcal{P}(B)} \qquad \mathcal{P}_A(B) = \frac{\mathcal{P}(B \cap A)}{\mathcal{P}(A)}$$

$$\mathcal{P}(A \cap B) = \mathcal{P}(B \cap A)$$

$$\mathcal{P}_B(A) \cdot \mathcal{P}(B) = \mathcal{P}_A(B) \cdot \mathcal{P}(A)$$

$$\mathcal{P}_A(B) = \frac{\mathcal{P}_B(A) \cdot \mathcal{P}(B)}{\mathcal{P}(A)} \quad \text{qed}$$

Thomas Bayes (1702 – 1761)

(6)
Geht zurück auf: *An Essay towards solving a problem in the Doctrine of Chances* (1763 posthum)

Satz von der totalen Wahrscheinlichkeit

Bilden die Ereignisse $A_1, A_2, \ldots A_n$ mit $\mathcal{P}(A_i) > 0$ für alle i eine Zerlegung des Ergebnisraums Ω (mit den Eigenschaften $\cup_i A_i = \Omega$ und $A_i \cap A_j = \emptyset, i \neq j$) dann gilt für die Wahrscheinlichkeit eines beliebigen Ereignisses B:

$$\mathcal{P}(B) = \sum_{i=1}^{n} \mathcal{P}_{A_i}(B) \cdot \mathcal{P}(A_i)$$

- Oft sieht man den Satz von der totalen Wahrscheinlichkeit auch in einer Formulierung in Kombination mit dem Satz von Bayes:

$$\mathcal{P}_B(A_j) = \frac{\mathcal{P}_{A_j}(B) \cdot \mathcal{P}(A_j)}{\sum_i \mathcal{P}_{A_i}(B) \cdot \mathcal{P}(A_i)}$$

- Oder etwas eingängiger für den Spezialfall $A_1 = A$ und $A_2 = \bar{A}$:

$$\mathcal{P}_B(A) = \frac{\mathcal{P}_A(B) \cdot \mathcal{P}(A)}{\mathcal{P}_A(B) \cdot \mathcal{P}(A) + \mathcal{P}_{\bar{A}}(B) \cdot \mathcal{P}(\bar{A})}$$

Stochastische Unabhängigkeit

Zwei Ereignisse A und B heißen stochastisch **unabhängig**, wenn gilt

$$\mathcal{P}(A \cap B) = \mathcal{P}(A) \cdot \mathcal{P}(B)$$

und stochastisch abhängig sonst.

- Im Rahmen bedingter Wahrscheinlichkeiten bekommt stochastische Unabhängigkeit eine sehr anschauliche Bedeutung:

  $\mathcal{P}_A(B) = \mathcal{P}(B)$ (und umgekehrt)

  D.h. die Wahrscheinlichkeit für das Eintreten von Ereignis B is unabhängig vom Eintreten des Ereignisses A.

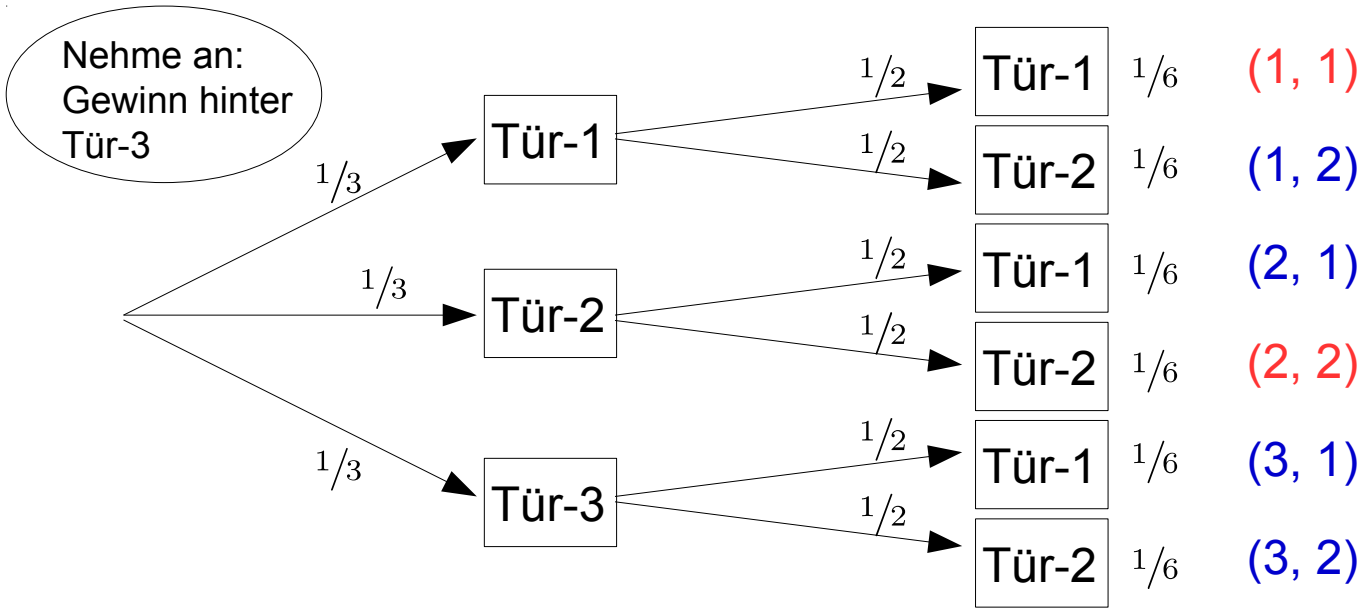
Geh auf's Ganze... (aka Ziegenproblem)

- Beispiel aus einer grausamen Fernsehshow der 80er Jahre:

- Spielidee:
  - Drei verschlossene Türen. Hinter einer befindet sich der Gewinn.
  - Nachdem der Kandidat eine Tür gewählt hat öffnet der Moderator eine der nicht gewählten Türen.
  - Wecheln oder nicht wechseln?

Geh auf's Ganze... (aka Ziegenproblem)

- Beispiel aus einer grausamen Fernsehshow der 80er Jahre:



- $\mathcal{P}(\text{Wechsel}) = 2/3$, $\mathcal{P}(\neg\text{Wechsel}) = 1/3$ $\rightarrow$ Wechsel lohnt sich (NB: nicht 1:1 auf die Politik übertragbar...).
- Wie ist Ihre Empfehlung und das Ergebnis für $n \to \infty$ viele Türen?

Geh auf's Ganze... (mal anders)

Bedingte Wahrscheinlichkeiten in der Medizin

- Stellen Sie sich eine Krankheit vor, die zu $0,1\%$ in der Bevölkerung vorkommt.
- A priori Wahrscheinlichkeit die Krankheit zu haben:

  $\mathcal{P}(\text{krank}) = 0,1\%$, $\mathcal{P}(\text{gesund}) = 99,9\%$
- Stellen Sie sich weiter einen Test vor der die Krankheit zu 98% erkennt, wenn sie denn vorliegt, aber zu 3% falsche Positivergebnisse liefert:

  $\mathcal{P}_{\text{krank}}(+) = 98\%$ $\mathcal{P}_{\text{krank}}(-) = 2\%$ $\mathcal{P}_{\text{gesund}}(+) = 3\%$ $\mathcal{P}_{\text{gesund}}(-) = 97\%$
- Sie haben einen positiven Test. Sind sie krank?

Bedingte Wahrscheinlichkeiten in der Medizin

- Stellen Sie sich eine Krankheit vor, die zu $0,1\%$ in der Bevölkerung vorkommt.
- A priori Wahrscheinlichkeit die Krankheit zu haben:

  $\mathcal{P}(\text{krank}) = 0,1\%$, $\mathcal{P}(\text{gesund}) = 99,9\%$
- Stellen Sie sich weiter einen Test vor der die Krankheit zu 98% erkennt, wenn sie denn vorliegt, aber zu 3% falsche Positivergebnisse liefert:

  $\mathcal{P}_{\text{krank}}(+) = 98\%$ $\mathcal{P}_{\text{krank}}(-) = 2\%$ $\mathcal{P}_{\text{gesund}}(+) = 3\%$ $\mathcal{P}_{\text{gesund}}(-) = 97\%$
- Sie haben einen positiven Test. Sind sie krank?

$$\mathcal{P}_{+}(\text{krank}) = \frac{\mathcal{P}_{\text{krank}}(+)\cdot\mathcal{P}(\text{krank})}{\mathcal{P}_{\text{krank}}(+)\cdot\mathcal{P}(\text{krank})+\mathcal{P}_{\text{gesund}}(+)\cdot\mathcal{P}(\text{gesund})+}$$

$$= \frac{0.98\cdot 0.001}{0.98\cdot 0.001+0.03\cdot 0.999} = 0.032$$

Bedingte Wahrscheinlichkeiten in der Medizin

- Stellen Sie sich eine Krankheit vor, die zu $0,1\%$ in der Bevölkerung vorkommt.
- A priori Wahrscheinlichkeit die Krankheit zu haben:

  $\mathcal{P}(\text{krank}) = 0,1\%$, $\mathcal{P}(\text{gesund}) = 99,9\%$
- Stellen Sie sich weiter einen Test vor der die Krankheit zu 98% erkennt, wenn sie denn vorliegt, aber zu 3% falsche Positivergebnisse liefert:

  $\mathcal{P}_{\text{krank}}(+) = 98\%$ $\mathcal{P}_{\text{krank}}(-) = 2\%$ $\mathcal{P}_{\text{gesund}}(+) = 3\%$ $\mathcal{P}_{\text{gesund}}(-) = 97\%$
- Sie haben einen positiven Test. Sind sie krank?

$$\mathcal{P}_{+}(\text{krank}) = \frac{\mathcal{P}_{\text{krank}}(+)\cdot\mathcal{P}(\text{krank})}{\mathcal{P}_{\text{krank}}(+)\cdot\mathcal{P}(\text{krank})+\mathcal{P}_{\text{gesund}}(+)\cdot\mathcal{P}(\text{gesund})+}$$

$$= \frac{0.98\cdot 0.001}{0.98\cdot 0.001+0.03\cdot 0.999} = 0.032$$

- Wie verändert sich diese Wahrscheinlichkeit, wenn sie den Test nochmal machen und er wieder positiv ausfällt?

Bedingte Wahrscheinlichkeiten in der Medizin

- Stellen Sie sich eine Krankheit vor, die zu $0,1\%$ in der Bevölkerung vorkommt.
- A priori Wahrscheinlichkeit die Krankheit

  $\mathcal{P}(\text{krank}) = 0,1\%$, $\mathcal{P}($
- Stellen Sie s kennt, wenn sie denn vorli

  $\mathcal{P}_{\text{krank}}(+) = 9$, $\mathcal{P}_{\text{gesund}}(-) = 97\%$
- Sie haben einen krank?

$$\mathcal{P}_+(\text{krank}) = \frac{\mathcal{P}_{\text{krank}}(+)\cdot\mathcal{P}(\text{krank})}{\mathcal{P}_{\text{krank}}(+)\cdot\mathcal{P}(\text{krank})+\mathcal{P}_{\text{gesund}}(+)\cdot\mathcal{P}(\text{gesund})+}$$

$$= \frac{0.98\cdot 0.001}{0.98\cdot 0.001+0.03\cdot 0.999} = 0.032 \quad (\to 0.519)$$

- Wie verändert sich diese Wahrscheinlichkeit, wenn sie den Test nochmal machen und er wieder positiv ausfällt?

Wiederholen lohnt sich! Bei unerfreulichem Ausgang machen Sie den Test ein zweites mal.

Überweisungen vom Allgemeinmediziner zum Facharzt machen die Diagnose des Facharztes besser!

Interpretation von Wahrscheinlichkeiten

- Viele Funktionen genügen den Axiomen von Kolmogorov. Es bleibt jedoch dem Anwender überlassen Ω und $\mathcal{P}$ geeignet zu interpretieren!

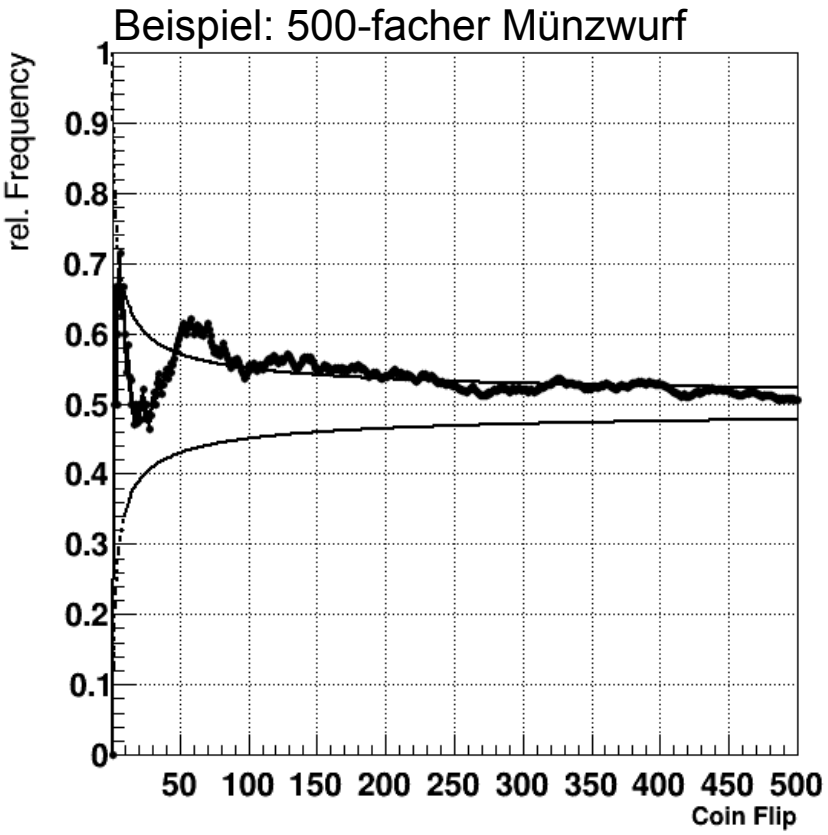
- Zwei (Haupt-)Schulen:

Frequentistische (klassische) Wahrscheinlichkeitsinterpretation

  - Annahme: ein wahrer Wert existiert, der im Grenzwert einer erschöpfenden Stichprobe ermittelt werden kann.

$$\mathcal{P}(A) = \lim_{N \to |\mathfrak{P}(\Omega)|} \frac{A}{N}$$

  - Hauptaufgabe der beurteilenden Statistik $\mathcal{P}(A)$ aus kleineren Stichproben zu bestimmen.



Interpretation von Wahrscheinlichkeiten

- Hier werden die Elemente des Ergebnisraums oft als Hypothesen bezeichnet, der Ereignisraum $\mathfrak{P}(\Omega)$ heißt Hypothesenraum.

- $\mathcal{P}(A)$: subjektives Maß des Fürwahrhaltens der Hypothese A.

$$\mathcal{P}_{\text{Daten}}(\text{Modell}) \propto \mathcal{P}_{\text{Modell}}(\text{Daten}) \cdot \mathcal{P}(\text{Modell})$$

Posteriori Wahrscheinlichkeit

Likelihood

A priori Wahrscheinlichkeit (*prior*)

- Herausforderung: Konstruktion des Hypothesenraums, so daß er Wahrscheinlichkeitsdefinition erfüllt.

Subjektive (**Bayesianische**) Wahrscheinlichkeitsinterpretation(7)

- Keine Aussage über *prior*. Wenn prior vorgegeben Aussage darüber, wie *prior* im Lichte ausgewerteter Daten zu bewerten ist.

- Für ein Beispiel aus dem Leben, siehe Folie 21.

(7) Beinhaltet frequentistische Interpretation (als möglichen Hypothenraum für eine Problemstellung).

Stochastisches Modell

- Zuordnung von $\mathcal{P}$ zu jedem Ergebnis in Ω abhängig von Interpretation & Geschick des Experimentators. Der Raum $(\Omega, \mathcal{P})$ **heißt Wahrscheinlichkeitsraum**.

Realität:

Mit welcher Wahrscheinlichkeit tritt ein reales Ereignis A_r ein?

Überprüfung z.B. durch relative Häufigkeit oder Hypothesentest.

$\mathcal{P}(A)$ als Wahrscheinlichkeit des realen Ereignisses A_r.

Konstruktion des Modells.

Übertragung des mathematischen Resultats in die Realität.

Modell:

Wahrscheinlichkeitsraum $(\Omega, \mathcal{P})$, Modellereignis A.

Rechnung.

Mathematisches Resultat $\mathcal{P}(A)$.

Kapiel 3.1:

Grundlagen der Wahrscheinlichkeits-theorie

- Rolle der Statistik in der modernen Physik.
- Ergebnisraum, Ereignisraum, Wahrscheinlichkeitsverteilung.
- Bedingte Wahrscheinlichkeit, Satz von der totalen Wahrscheinlichkeit, Unabhägigkeit zweier Ereignisse.
- Interpretation von (Zufalls-)Experimenten und stochastische Modelle.

Kapiel 3.2:

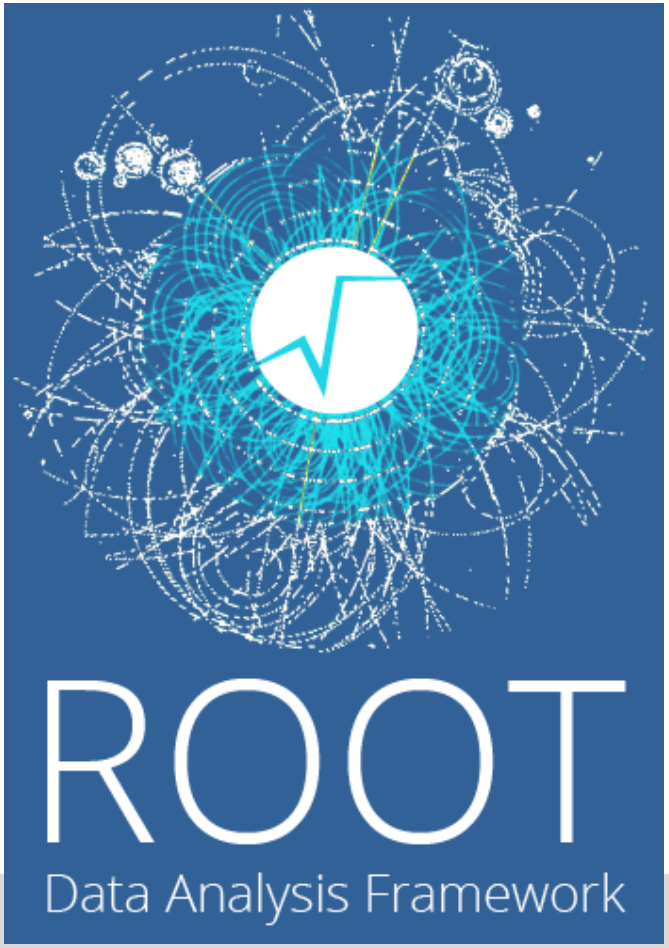
Werkzeuge zur statistischen Datenanalyse

Motivation und Übersicht?

- Um große Datenmengen analysieren und stochastiche Modelle überprüfen/validieren zu können benötigen Sie Werkzeuge, die:
  - Große Datenmengen übersichtlich darstellen können.
  - Vieldimensionale Wahrscheinlichkeitsräume „ausleuchten können“.
- Solche Werkzeuge existieren in großer Zahl, mit variierender Qualität, z.B:
  - Origin, MATLAB (nicht MS Excel).
  - gnuplot.
  - Python basiert: NumPy, SciPy, matplotlib, ...
  - **ROOT**.

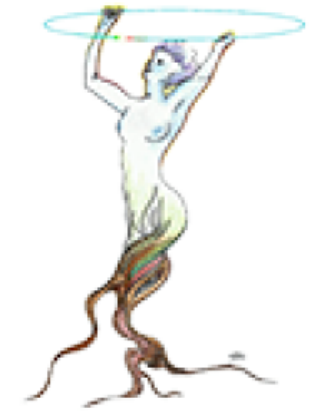
Was ist ROOT?

- Open-Source framework zur Datenanalyse (C++-Klassenbobliothek).
- Entwickelt hauptsätzlich am CERN, Standardwerkzeug in der Teilchenphysik.
- Anwendungen:
  - Verwaltung großer Datenmengen
  - Darstellung mathematischer Funktionen & Daten, Verwaltung von Unsicherheiten.
  - Definition und Anpassungen von Modellen and Meßdaten.
  - Unterstützt Verwendung von Monte-Carlo Methoden.
  - Grafische Oberfläche.
  - Entwicklungsumgebung für komplexere Datenanalysepakete (z.B. bei CMS am LHC).



ROOT Dokumentation

- Hauptwebseite *root.cern.ch*.
- Benutzerhandbuch.
- Klassenreferenz (für root 5.34).
- Einführung für KIT Studenten.
- **NB:**
  keine noch so gute Einführung taugt was, wenn Sie sich nicht selbst mutig an die s/w heranwagen. Sie sind Physiker im Herzen: probieren Sie die s/w aus, spielen Sie damit. Übungsblatt 9 wird Ihnen dabei eine Anleitung geben.



http://root.cern.ch

A ROOT Guide For Students

"Diving Into ROOT"

Abstract:
ROOT is an object-oriented framework for data analysis. Among its prominent features are an advanced graphical user interface for visualization and interactive data analysis and an interpreter for the C++ programming language, which allows rapid prototyping of analysis code based on the C++ classes provided by ROOT. Access to ROOT classes is also possible from the very versatile and popular scripting language PYTHON.
This introductory guide shows the main features applicable to typical problems of data analysis in student labs: input and plotting of data from measurements and comparison with and fitting of analytical functions. Although appearing to be quite a heavy gun for some of the simpler problems, getting used to a tool like ROOT at this stage is an optimal preparation for the demanding tasks in state-of-the art, scientific data analysis.

Erste Schritte

- Starten von ROOT:

```
prompt> root
   ******************************************
   *                                        *
   *        W E L C O M E  to  R O O T      *
   *                                        *
   *   Version   5.34/23   7 November 2014  *
   *                                        *
   *  You are welcome to visit our Web site *
   *          http://root.cern.ch           *
   *                                        *
   ******************************************

ROOT 5.34/23 (heads/v5-34-00-patches@v5-34-22-106-g4a0dea3,

CINT/ROOT C/C++ Interpreter version 5.18.00, July 2, 2010
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0]
```

Beenden:

```
root [0] .q
```

- NB: benötigt Umgebungsvariable $ROOTSYS:
  - $ROOTSYS/bin in $PATH,
  - $ROOTSYS/lib in $LD_LIBRARY_PATH.

Benutzung von ROOT

- Interaktiv, aus der Kommandozeile (C++ interpreter, nicht empfohlen).

- Makros (=kurze C++(-artige) Funktionen in text files):

Text file:

```
void macro_01()
{
  cout << "Die ersten 10 Quadratzahlen:" << endl;
  for( int i = 1; i <= 10; ++i ) {
    cout << i*i << endl;
  }
}
```

Ausführung in ROOT:

```
root[0] .x macro_01.cxx
```

Alternativ:

```
root[0] .L macro_01.cxx
root[1] macro_01()
```

- Kompilierte Makros (=echte C++ Funktionen in text files):

Text file:

```
#include <iostream>
using namespace std;

void macro_01()
{
  cout << "Die ersten 10 Quadratzahlen:" << endl;
  for( int i = 1; i <= 10; ++i ) {
    cout << i*i << endl;
  }
}
```

Ausführung in ROOT:

```
root[0] .x macro_01.cxx+
```

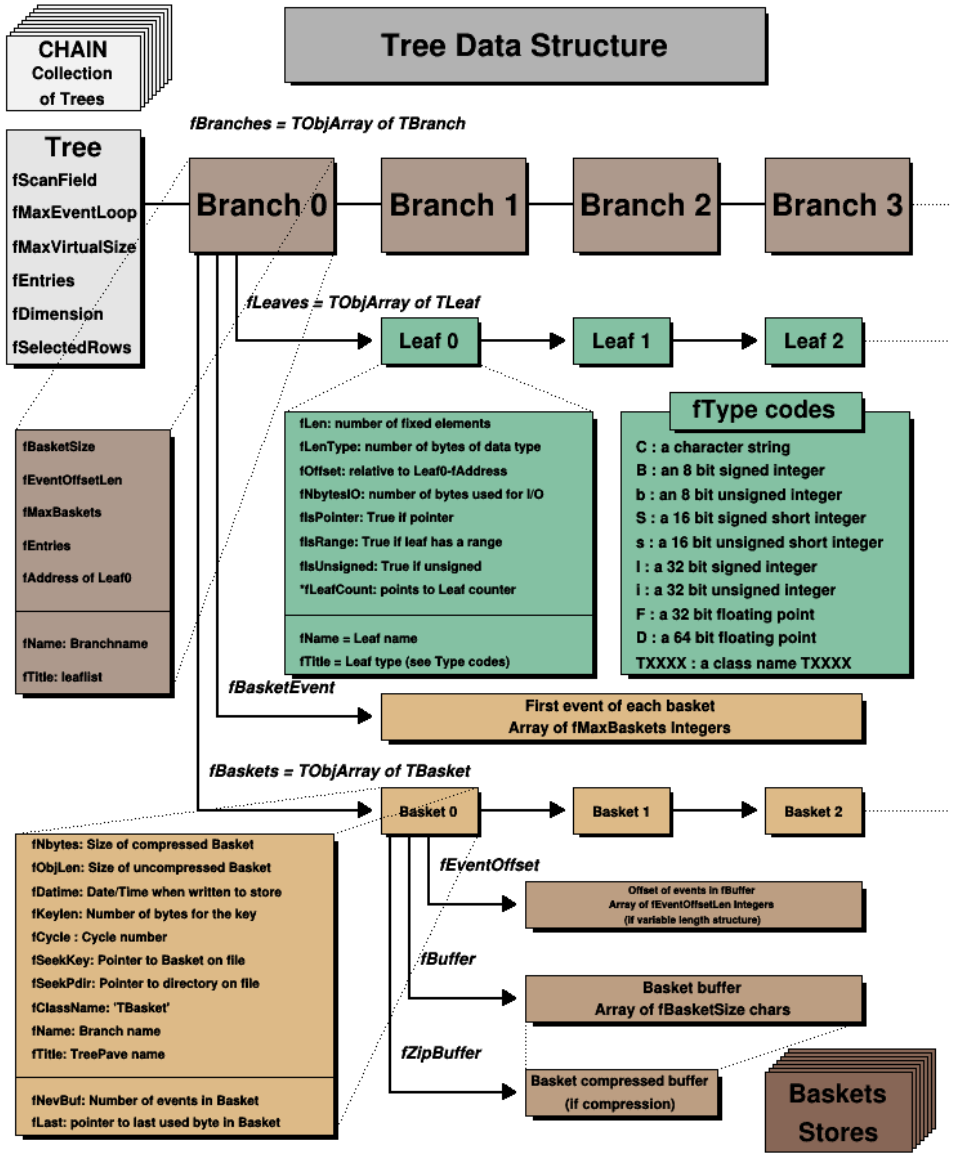
Alternativ:

```
root[0] .L macro_01.cxx+
root[1] macro_01()
```

ROOT Datenmodell: Trees

- ROOT-Tree:
  - Datenspeicherung in Klassen (z.B. alle Objekte einer Teilchenkollision in einem **Ereignis**).
  - Hierarchie: Tree – Branch – Leaf.
- Einfachster Tree: n-Tuple:
  - Tree aus identischen Variablentypen (z.B. *float*, Viererimpulse, ...).

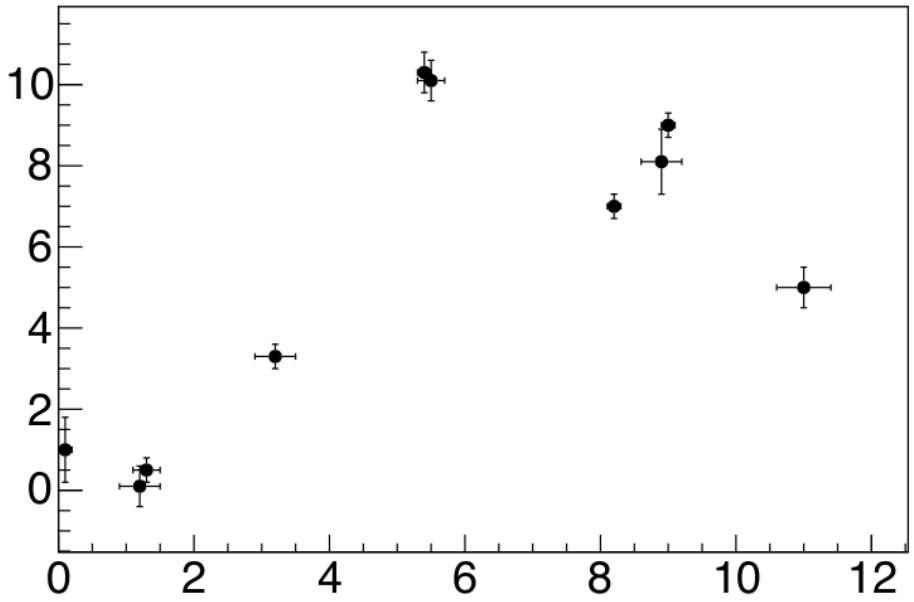
```
root[0] TNtuple* nt = new TNtuple( "nt",
"Mein n-Tupel", "E:px:py:pz" )
root[1] nt->Fill( 10.0, 9.0, 0.5, 0 )
root[2] nt->Fill( 25.3, -5.0, 20.0, -0.5 )
...
```



Darstellung von Daten: Graphen

- Darstellung in Form von Datenpunkten (x, y) mit (gegebenenfalls asymmetrischen) Unsicherheiten/Fehlerbalken in x- und/oder y-Richtung:

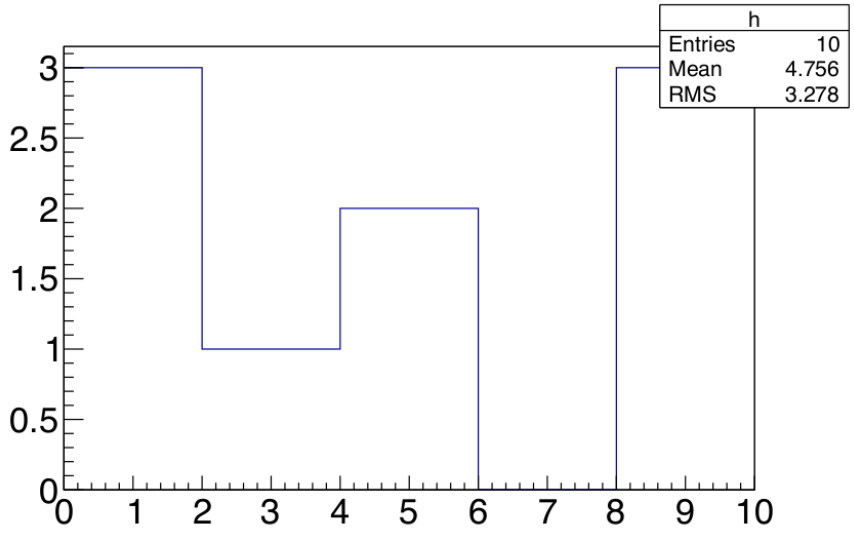
```
void graph()
{
  const int N = 10;
  float x[N] = { 1.3, 5.4, 3.2, 5.5, 8.2,
                 8.6, 9.0, 1.2, 0.1, 11 };
  float value[N] = { 0.5, 10.3, 3.3, 10.1, 7.0,
                 8.1, 9.0, 0.1, 1.0, 5.0 };
  float ex[N] = { 0.2, 0.1, 0.3, 0.2, 0.1,
                 0.3, 0.1, 0.3, 0.1, 0.4 };
  float ey[N] = { 0.3, 0.5, 0.3, 0.5, 0.3,
                 0.8, 0.3, 0.5, 0.8, 0.5 };

  TGraphErrors* g =
      new TGraphErrors( N, x, value, ex, ey );
  g->SetMarkerStyle( 20 );
  g->SetMarkerSize( 1.0 );
  g->Draw( "ap" );
}
```



Darstellung von Daten: Histogramme

- Histogramm = Häufigkeitsverteilung:
  - Kompakte Darstellung großer Datenmengen (→ Informationsverlußt).
  - Einteilung eines Intervalls in Teilintervalle (=bins) gleicher oder variabler Breite.
  - Inhalt des bins: Anzahl der Ereignisse.
  - Auch in >1 Dimensionen.
  - Es gibt auch underflow/overflow bins.

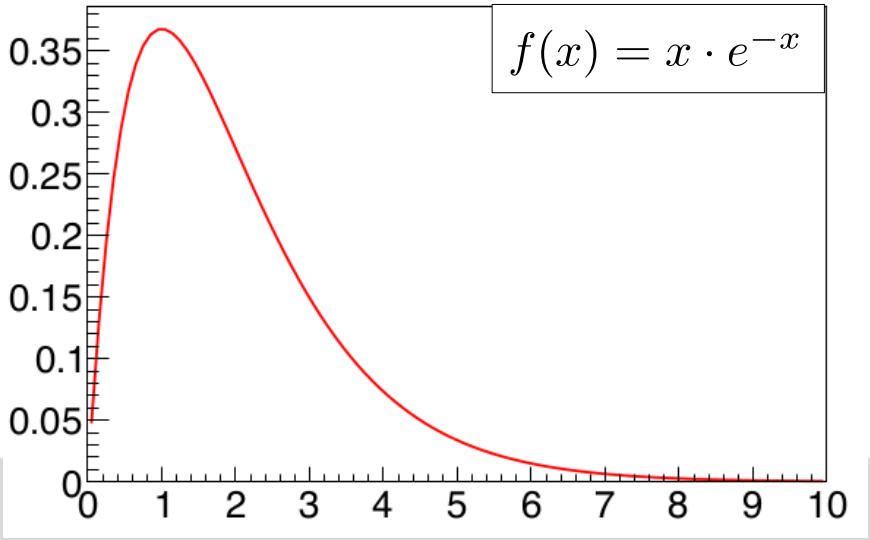
```
void histogramm()
{
  TH1F* h = new TH1F( "h", "Histogramm", 5, 0, 10 );
  float data[10] = { 1.3, 5.4, 3.2, 5.5, 8.2,
                     8.9, 9.0, 1.2, 0.1, 11.0 };
  for( int i = 0; i < 10; ++i ) {
    h->Fill( data[i] );
  }
  h->Draw();
}
```

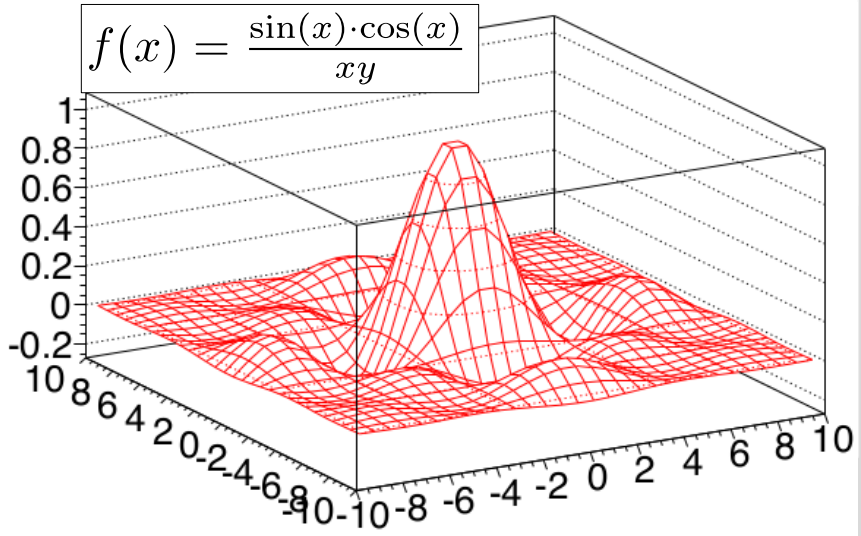


Darstellung von Funktionen

- Standardfunktionen vordefiniert.
- Einfachere Funktionen über *strings* definierbar.
- (Beliebig) Kompliziertere Funktionen über C++-Funktionen mit festem Interface.

```
void funktion()
{
  TF1* f1 = new TF1( "f1", "exp(-x)*x", 0, 10 );
  f1->Draw();

  TF2* f2 =
    new TF2( "f2", func2, -10, 10, -10, 10, 0 );
  f2->Draw( "surf" );
}

double func2( double* val, double* par )
{
  double x = val[0];
  double y = val[1];
  return sin(x)*sin(y)/(x*y);
}
```

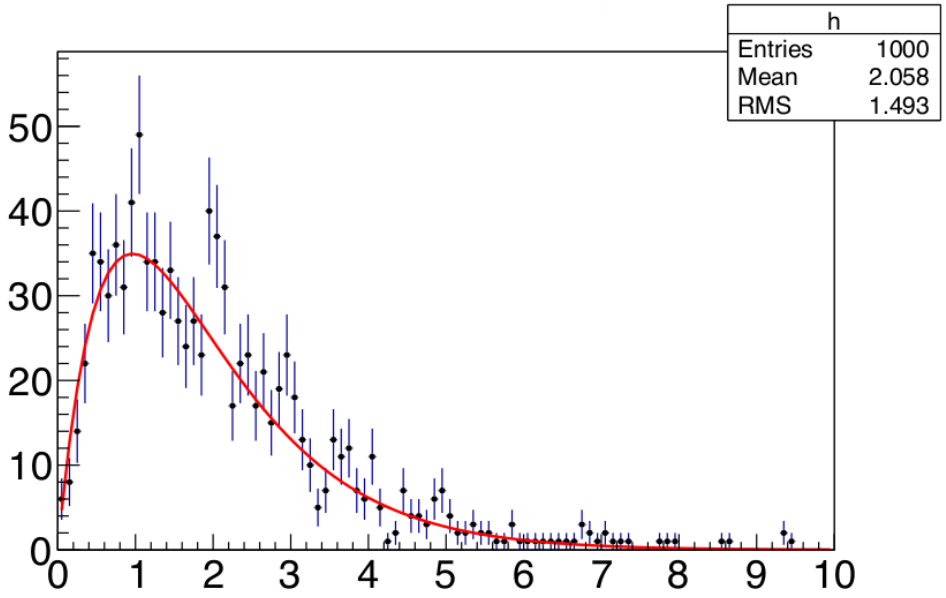




Anpassung von Funktionen

- **Beispiel:** Erzeugung von Zufallsverteilung mit Monte-Carlo Methoden (blaue Pseudo-Datenpunkte; cf. nächste Vorlesung).

- Anschließend: Anpassung mit parametrisierter Funktion (rote Kurve).




```
void anpassung()
{
  TH1* h =
    new TH1F( "h", "Histogramm", 100, 0, 10 );
  TF1* f1 =
    new TF1( "f1", "exp(-x)*x", 0, 10 );

  h->FillRandom( "f1", 1000 );
  h->SetMarkerStyle( 20 );
  h->SetMarkerSize( 0.5 );

  TF1* ffit =
    new TF1( "ffit", "[0]*exp(-[1]*x)*x", 0, 10 );
  ffit->SetParameters( 10, 1 );
  h->Fit("ffit","","e");
}
```

Ausblick: ROOT-basierte Analysepakete

- ROOT stellt auch die Grundlage für weiterführende komplexere Analysepakete:
- **RooFit**: mächtigeres Interface für komplizierte Anpassungen an Daten und Monte-Carlo basierte Modelltests (z.B. Anpassung der Kopplungen des Higgs Bosons, mit ~4250 nicht trivial korrelierten Störparametern).
- **RooStat**: statistische Datenanalyse.
- **TMVA**: multivariate Klassifikation und Regression von Daten (Likelihood Methoden, Neuronale Netze, Boosted Decision Tree's, ...)

Python Interface zu ROOT

- ROOT C++-Klassen haben auch eine python Sprachanbindung (pyroot):
  - Quasi 1:1 Abbildung der C++-Klassen.
  - Alle Vorteile einer Skriptsprache, wie python (e.g. dynamische Typisierung).
  - In der Praxis oft heterogener Ansatz: zeit-/speicherkritische Anwendungen in C++, Steuerung und einfache Aufgaben (=Menschkritische Anwendungen) in python.
- Vergleich Anpassungsrechnung von zuvor:

C++

```
void anpassung()
{
  TH1* h =
    new TH1F( "h", "Histogramm", 100, 0, 10 );
  TF1* f1 =
    new TF1( "f1", "exp(-x)*x", 0, 10 );

  h->FillRandom( "f1", 1000 );
  h->SetMarkerStyle( 20 );
  h->SetMarkerSize( 0.5 );

  TF1* ffit =
    new TF1( "ffit", "[0]*exp(-[1]*x)*x", 0, 10 );
  ffit->SetParameters( 10, 1 );
  h->Fit("ffit","","e");
}
```

Python

```
from ROOT import TH1F, TF1

def anpassung():
    h  = TH1F( "h", "Histogramm", 100, 0, 10 );
    f1 = TF1( "f1", "exp(-x)*x", 0, 10 );

    h.FillRandom( "f1", 1000 );
    h.SetMarkerStyle( 20 );
    h.SetMarkerSize( 0.5 );

    ffit = TF1( "ffit", "[0]*exp(-[1]*x)*x", 0, 10 );
    ffit.SetParameters( 10, 1 );
    h.Fit("ffit","","e");

if '__main__':
    anpassung()
```

Ernste Alternativen zu ROOT

- Als Alternative bieten sich einige spezielle python Bibliotheken an:
  - **NumPy**: Grundfunktionen, z.B. Arrays, Matrixrechnung, ...
  - **SciPy**: Sammlung von numerischen Algorithmen, z.B. zur Signalprozessierung, Optimierung, statistischen Datenanalyse.
  - **matplotlib**: Darstellung von Daten in Histogrammen, Graphen, ...
- ROOT? Pyroot? NumPy/SciPy? - Qual der Wahl?
  - Die Übungen dieser Vorlesungen lassen sich auch mit plain python bewältigen.
  - Im Masterstudium (insb. in der experimentellen (Astro-) Teilchenphysik) werden sie dem beschriebenen Hybridansatz begegnen: ROOT (in C++) mit Steuerelementen in python.

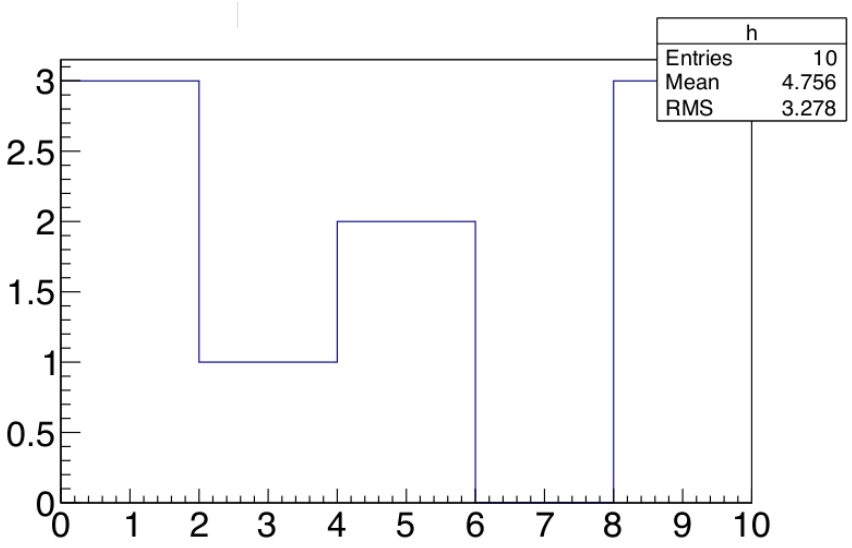
Vergleich: ROOT vs Python

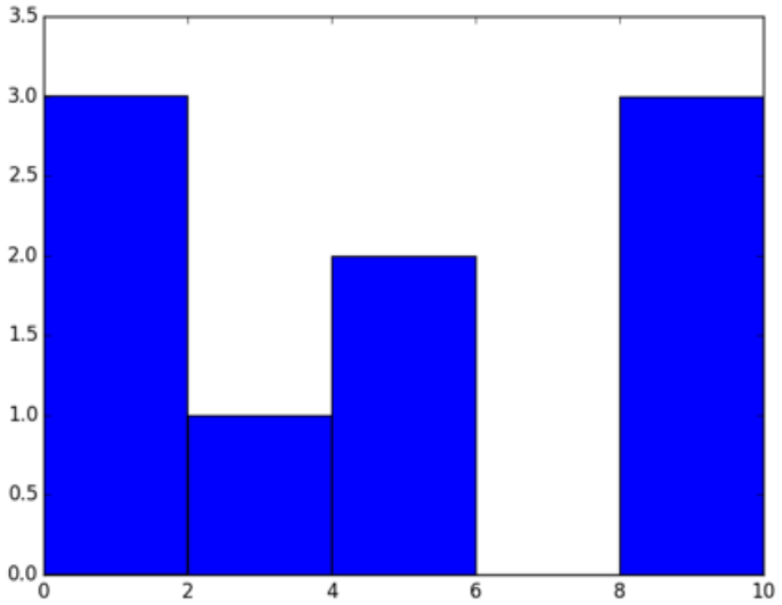
C++

```
void histogramm()
{
  TH1F* h = new TH1F( "h", "Histogramm", 5, 0, 10 );
  float data[10] = { 1.3, 5.4, 3.2, 5.5, 8.2,
                     8.9, 9.0, 1.2, 0.1, 11.0 };
  for( int i = 0; i < 10; ++i ) {
    h->Fill( data[i] );
  }
  h->Draw();
}
```

Python

```
import numpy as np
import matplotlib.pyplot as plt

def histogramm():
    data = np.array( [ 1.3, 5.4, 3.2, 5.5, 8.2,
                       8.9, 9.0, 1.2, 0.1, 11.0 ] )
    plt.hist( data, bins=5, range=(0,10) )
    plt.axis( [0, 10, 0, 3.5 ])
    plt.show()

if '__main__':
    histogramm()
```





Kapiel 3.2:

Werkzeuge zur statistischen Datenanalyse

- ROOT: C++-Framework zur Datenanalyse:
  - Erste Schritte/Dokumentation.
  - (Software-)Modell zur Datenspeicherung (ROOT-Tree).
  - Darstellung von Daten: Graphen, Histogramme.
  - Funktionen und Anpassung von Funktionen.
  - ROOT-basierte weiterführende Analysepakete.
- Python Bibliotheken als Alternative zu ROOT.