Modeliranje 1-D porazdelitve: razpadi Higgsovega bozona

1. naloga: Praktikum strojnega učenja v fiziki

predavatelj: prof. dr. Borut Paul Kerševan
asistent: Jan Gavranovič

Institut Jožef Stefan (F9)

6. oktober 2023

Institut "Jožef Stefan" Ljubljana, Slovenija

Univerza v Ljubljani
Fakulteta za matematiko in fiziko

1. vaje

Pregled 1. vaje

1. Fizikalni uvod v nalogo.
2. Priprava delovnega okolja (dostop do FMF strežnika, koda, Python, urejevalniki, ssh, ...).
3. Podrobnejša informacije in usmeritve za reševanje naloge:
   - branje podatkov,
   - generiranje histogramov,
   - napake na histogramih,
   - funkcije za gladko fitanje ozadja,
   - primer preprostega fitanja s knjižnico `scipy`,
   - parametrizacija signala.
4. Navodila za pripravo in oddajo poročila.

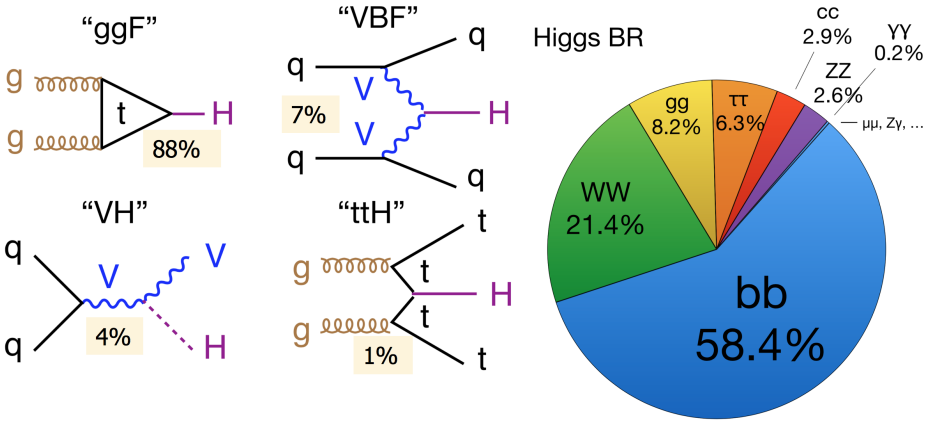
Kratek fizikalni uvod

Fizikalni uvod

- Kolaboracije na LHC se posvečajo iskanju nove fizike in natačnim meritvam napovedi Standardnega Modela (SM).
- Za primerjavo z meritvami je potrebno poznati oblike napovedanih porazdelitev.
- Primer: porazdelitev invariantne mase dveh mionov, ki nastaneta pri razpadu Higgsovega bozona (naša vaja).
- Modelska (teoretična/napovedana) porazdelitev je pridobljena z MC simulacijami.
- Primerjava modelske (MC) in izmerjene (data) porazdelitve $\Rightarrow$ odstopanje med obema lahko pomeni nekaj novega (npr. odkritje Higgsovega bozona).
- V MC lahko ločimo signal (zanimivi, a redki procesi) in ozadje (znane in izmerjene interakcije).
- Pri končnem opisu procesov si pomagamo z regresijo (*fitom*, parametrizacijo) kinematičnih porazdelitev, ki nas zanimajo (popravimo/zgladimo simulacijo za najboljše ujemanje).

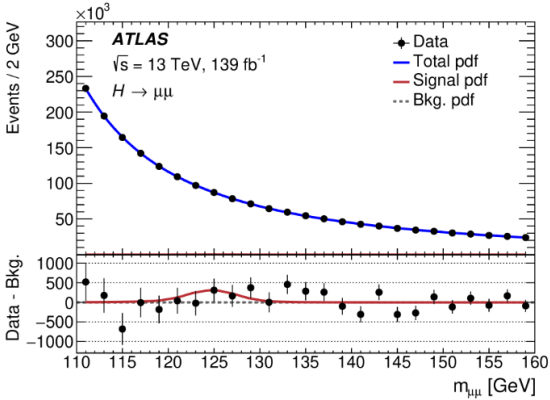
Nastanek in razpad Higgsovega bozona na LHC-ju

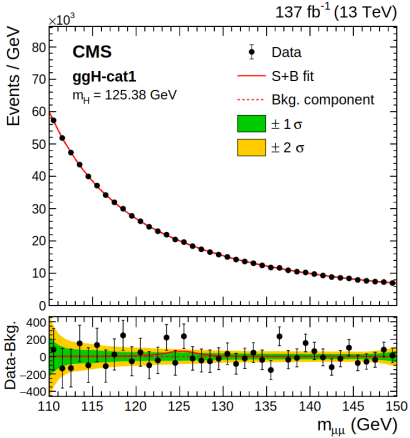
- Nastanek in razpad pri $m_H = 126$ GeV:



$m_{\mu\mu}$ porazdelitev na detektorjih ATLAS in CMS

- Iskanje signala v ogromnih količinah podatkov je velik izziv.
- Uporaba najnaprednejših računskih metod za ločevanje procesov signala od ozadja.





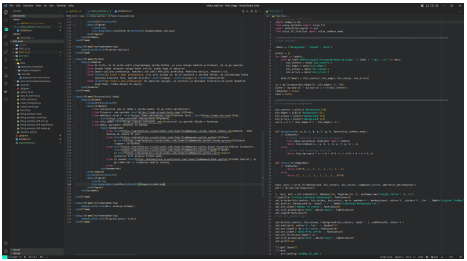
Priprava okolja

Python

- Za tiste, ki še niste vešči programskega jezika Python, je prva naloga idealna priložnost, da se ga naučite.
- Seveda lahko rešujete nalogo kakor želite, glede tega ni omejitev.
- Imate različno predznanje, nekateri ste imeli Mafijski praktikum, Modelsko analizo, nekateri nič.
- **Za tiste z manj predznanja**: cilj prve naloge je, da se spoznate z jezikom Python, do naslednjega tedna spoznate klasične fite, uporabo knjižnic `numpy`, `scipy` in `matplotlib`.
- **Za tiste z več predznanja**: Ni omejitev navzgor, na učilnici je dostopna literatura do parih dodatnih algoritmov, lahko delate že naprej.

Urejevalniki kode

- Urejevalnik vam je lahko v veliko pomoč, če ga znate uporabljati!
- Priporočam uporabo VSCode.
- Nekateri ostali: Sublime Text, Vim, PyCharm, ...
- Zelo dober tutorial za uporabo VSCode s Pythonom.
- Nekaj uporabnih dodatkov za VSCode:
  1. Remote - SSH (dostop do remote PC-jev),
  2. Python in Pylance (support za Python),
  3. Black formatter, flake8 in isort (pravilen stil pisanja Python kode),
  4. in seveda GitHub Copilot, ki ga lahko kot študentje dobite zastonj,
  5. ...



Virtualno okolje in knjižnice

- Ideja virtualnega okolja je, da se izolira okolje, v katerem se izvajajo programi, od okolja, ki je na računalniku.
- Tako se lahko v tem "virtualnem" okolju namesti samo tiste knjižnice, ki so potrebne za izvajanje določenih programov.
- Knjižnice za to nalogo:
  - `numpy` (za numerične operacije),
  - `scipy` (znanstveno in tehnično računanje),
  - `matplotlib` (za risanje grafov),
  - `pandas` (za delo s podatki),
  - `scikit-learn` (za strojno učenje).
- Včasih pride prav tudi `jupyter notebook` (predvsem za risanje grafov).
- Podrobnejša navodila so v `README.md` na repozitoriju.

Računanje na daljavo

- Ni nujno, lahko pa vam pride prav!
- Računalnik MARVIN na FMF, na katerem lahko poganjate vaše domače naloge.
- Dostopen preko ssh: `ssh <username>@marvin.fmf.uni-lj.si`.
- Uporabniki Windows-ov: najlažje kar z Windows Subsystem for Linux.
- Na spletni učilnici si ustvarite račun, geslo dobite po mailu.
- Na strežniku lahko zaganajte zahtevnejše izračune v sistemu Linux, tako vam sploh ni treba imeti kode lokalno.
- Virtualno okolje za vse možne ML pakete v python okolju:
  `source /data/virtualenvs/virtenv-py310-PSUF/bin/activate`.
- Imate dostop do materiala domačih nalog: `/data/PSUF_naloge/1-naloga/`.
- Skripte se poganjajo z:
  `(virtenv-py310-PSUF)<username>@marvin: /PSUF$ python <skripta>.py`

Kako do podatkov in kode?

- Predprocesirane podatke/histograme za reševanje naloge dobite:
  - Spletna učilnica,
  - CERNBox ($\sim$ 7 GB):
    1. `mc_bkg_new.h5`,
    2. `mc_sig.h5`,
    3. `data.h5`.
  - Marvin: `/data/PSUF_naloge/1-naloga/DATA/`,
  - Naloži vam jih lahko tudi skripta `create_histograms`.
- Generacija svojih histogramov ni nujna (je pa dobrodošla) - imate že nekaj narejenih histogramov v območjih:
  1. celotno kinematično območje,
  2. Z vrh,
  3. interval okrog Higgsovega bozona.
- Prav tako je že pripravljena koda, ki večinoma sama po sebi deluje, dostopna na:
  - Spletna učilnica,
  - GitHub z ukazom: `git clone https://github.com/j-gavran/PSUF_Hmumu.git`.

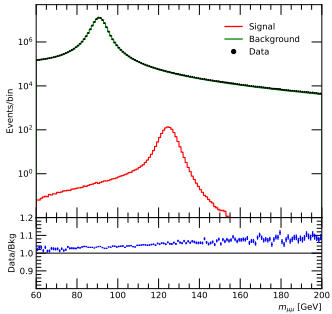
Reševanje naloge

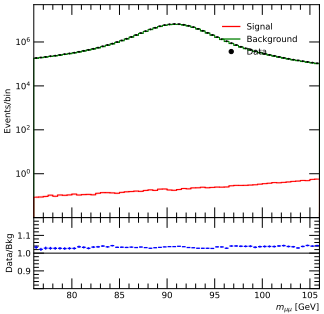
Histogrami in napake

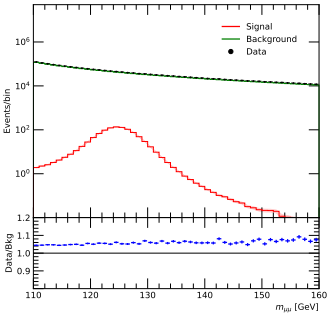
- Podatki so v formatu `.h5`, ki ga lahko berete s knjižnico `pandas`.
- Skripta `create_histograms` zgenerira `.npz` datoteke histogramov, ki jih lahko berete z `numpy`-jem (pripravljenih je že nekaj histogramov za različne $m_{\mu\mu}$ intervale).
- Potrebovali boste:
  1. `bin_edges`: robne x vrednosti binov,
  2. `bin_centers`: centralne x vrednosti binov,
  3. `bin_values`: število dogodkov za posamični bin,
  4. `bin_errors`: pravilne napake za posamični bin.
- Zaželjeno je, da se igrate s to skripto in generirate histograme z različnim binning-om oz. intervalom (lahko tudi z bini različnih širin).
- Napaka na binih za simulirano ozadje in signal je že določena: $\sigma_k^2 = \sum_{x_i \in k} w_i^2$.
- Napaka na binih podatkov je Poissonska: $\sigma_k^2 = \sqrt{N_k}$.
- Pri delu se namenoma zmoti, preveri učinek Poissonske in konstantne napake na simuliranem ozadju in signalu.

Risanje histogramov

- Dovolj je preprosta vizualizacija, da vam bo lažje pa imate pripravljeno skripto `visualize_data`.
- **Zgoraj**: simuliran signal, simulirano ozadje in zajete podake.
- **Spodaj**: razmerje med zajetimi podatki in simuliranim ozadjem.
- Za stil risanja grafov lahko uporabite predloge iz mplhep.







Modeliranje ozadja

- Razmerje signal/ozadje je $\sim 0.2\%$ v območju $m_{\mu\mu} \in [120, 130]$ GeV $\rightarrow$ potrebujemo dober opis ozadja za razločevanje med procesi.
- Glavni proces ozadja je razpad $Z \rightarrow \mu^+\mu^-$ (Drell-Yan), ki prispeva resonančno porazdelitev z vrhom pri $m_Z = 91.2$ GeV.
- Še veliko drugih procesov ozadja (npr. tvorba in razpad ZZ, WW in WZ, $t\bar{t}$ nastanek, ...)
- Na voljo imamo več metod za ustrezno določitev simulacije ozadja:
  - uteževanje s polinomi nizke stopnje,
  - teoretično motivirani nastavki za modeliranje oblike porazdelitve,
  - uporaba SVM metod z različnimi jedri in regularizatorji,
  - uporaba metod strojnega učenja (npr. GPR),
  - dekompozicija na ortogonalne polinome.

Glajenje ozadja s funkcijskimi nastavki

- Glajenja ozadja (smoothing) na intervalu 110 GeV naprej, preveri različne funkcije:
  1. polinomi: $f(m) = a_1 + a_2 m + a_3 m^2 + \ldots$,
  2. potenčne funkcije: $f(m) = a m^b$,
  3. exponentne funkcije: $f(m) = a_1 e^{-b_1 m} + a_2 e^{-b_2 m} + \ldots$,
  4. dijet funkcija: $f(m) = a(1 - m^d)^c m^{b_1 + b_2 \log m + b_3 \log m^2 + \ldots}$,
  5. eksponent polinoma: $f(m) = a e^{b_1 m + b_2 m^2 + \ldots}$.
- Primer uporabe funkcije `curve_fit` (preberi dokumentacijo!) iz knjižnice `scipy` za naivno fitanje ozadja:

```
1 import numpy as np
2 from scipy.optimize import curve_fit
3
4 poly3 = lambda m, a, b, c, d: a + b*m + c*m**2 + d*m**3
5 popt, pcov = curve_fit(poly3, bin_centers, bin_values, sigma=bin_errors, p0=np.ones(4))
6
7 std = np.sqrt(np.diag(pcov))
8 fit_values = poly3(bin_centers, *popt)
9
```

Teoretično motivirani nastavki

- Uporaba različnih teoretično podkrepljenih nastavkov.
- Kolaboracija CMS uporablja nastavek (modificiran Breit-Wigner):

$$mBW(m_{\mu\mu}|m_Z, \Gamma_Z, a_1, a_2, a_3) = \frac{e^{a_2 m_{\mu\mu} + a_3 m_{\mu\mu}^2}}{(m_{\mu\mu} - m_Z)^{a_1} + (\Gamma_Z/2)^{a_1}} \,.$$

- Kolaboracija ATLAS uporabi bolj komplicirano funkcijo, ki je v skripti `atlas_fit_function`.
- ATLAS funkcije podajajo zgolj obliko - ne pozabite dodati prostega parametra ali poljubne funkcije (npr. utež s polinomom) za normalizacijo!
- Uporabimo jo lahko na naslednji način:

```
1 from atlas_fit_function import atlas_invMass_mumu_core
2
3 atlas_fit = lambda m, K: K * atlas_invMass_mumu_core(x)
4
```

Modeliranje signala

- V SM je $H \to \mu\mu$ signal ozka resonanca širine 4.1 MeV pri $m_H = 125.09$ GeV.
- Signal je vsota vseh načinov nastanka Higgsovega bozona (ggF, VBF, VH, $t\bar{t}H$).
- Za parametrizacijo signala se uporabi Crystal Ball (CB) funkcijo, ki je kombinacija Gaussovskega jedra in debelejših potenčnih repov:

$$\mathrm{CB}\left(m_{\mu\mu} \mid \alpha_{\mathrm{L,R}}, n_{\mathrm{L,R}}, m_{\mathrm{CB}}, \sigma_{\mathrm{CB}}\right) = \begin{cases} \mathrm{e}^{-(m_{\mu\mu}-m_{\mathrm{CB}})^2/2\sigma_{\mathrm{CB}}^2}, & -\alpha_{\mathrm{L}} < \frac{m_{\mu\mu}-m_{\mathrm{CB}}}{\sigma_{\mathrm{CB}}} < \alpha_{\mathrm{R}} \\ \left(\frac{n_{\mathrm{L}}}{|\alpha_{\mathrm{L}}|}\right)^{n_{\mathrm{L}}} \mathrm{e}^{-\alpha_{\mathrm{L}}^2/2} \left(\frac{n_{\mathrm{L}}}{|\alpha_{\mathrm{L}}|} - |\alpha_{\mathrm{L}}| - \frac{m_{\mu\mu}-m_{\mathrm{CB}}}{\sigma_{\mathrm{CB}}}\right)^{-n_{\mathrm{L}}}, & \frac{m_{\mu\mu}-m_{\mathrm{CB}}}{\sigma_{\mathrm{CB}}} \leq -\alpha_{\mathrm{L}} \\ \left(\frac{n_{\mathrm{R}}}{|\alpha_{\mathrm{R}}|}\right)^{n_{\mathrm{R}}} \mathrm{e}^{-\alpha_{\mathrm{R}}^2/2} \left(\frac{n_{\mathrm{R}}}{|\alpha_{\mathrm{R}}|} - |\alpha_{\mathrm{R}}| + \frac{m_{\mu\mu}-m_{\mathrm{CB}}}{\sigma_{\mathrm{CB}}}\right)^{-n_{\mathrm{R}}}, & \frac{m_{\mu\mu}-m_{\mathrm{CB}}}{\sigma_{\mathrm{CB}}} \geq \alpha_{\mathrm{R}} \end{cases}$$

- Na končen rezultat fitamo

$$\alpha \cdot \mathrm{CB}\left(m_{\mu\mu} \mid \alpha_{\mathrm{L,R}}, n_{\mathrm{L,R}}, m_{\mathrm{CB}}, \sigma_{\mathrm{CB}}\right),$$

kjer je α normalizacijski faktor.

Parametrizacija signala s CB funkcijo

- Končni cilj je, da od podatkov odštejete fit ozadja in dobite signal.
- Da ocenite vaš rezultat, pofitate CB na signal - primer v `fitting_example_CB`.
- Poskusite najprej parametrizirati/pofitati simuliran signal iz histogramov.
- Da pofitate Crystall Ball funkcijo, jo morate obvezno pomnožiti s prostim (*scale*) parametrom!
- Zgolj s funkcijo `curve_fit` in 7 prostimi parametri boste težko prišli do dobrega fita.
- Ker so prosti parametr $\alpha_{L,R}$ in $n_{L,R}$ visoko korelirani, je za hitro konvergenco smiselno kombinirati samo minimizacijo s pregledom (*scanning*):
  1. spreminjaj $n_{L,R}$ po neki mreži smiselnih vrednosti in poišči $\alpha_{L,R}$ z minimizacijo,
  2. iterativno obrni postopek: spreminjaj vrednosti $\alpha_{L,R}$ in minimiziraj $n_{L,R}$.
- Cilj: faktor za napoved α SM čim bližje vrednosti 1 (ne pozabi na napako!).

Priprava poročila

Kako naj izgleda poročilo

- Podobno kot modelska analiza in mafijski praktikum.
- Na nekaj straneh opišeš potek vaje in predstaviš rezultate, zanimive probleme in rešitve.
- Okvirno se držite navodil (točke 1-9).
- Cilj naloge je, da ste sami čim bolj inovativni:
  1. preizkušate čim več načinov kako priti do rezultata,
  2. ni tako pomembno, kakšni so rezultati,
  3. štejejo vztrajnost in dobre ideje.
- Možnosti za analizo je ogromno, ocenite kaj se vam zdi smiselno preveriti in vključiti.
- Baseline ocena je 8 (rešiš vse naloge iz navodil, večina stvari dela že out of the box), vse dodatno (razmisleki, ideje, inovativni grafi, še kakšna metoda, ...) je 9 in 10.
- **Rok za oddajo**: petek 20.10.2023 do 10:00 zjutraj.
- **Obvezno**: PDF format.
- **Ime datoteke**: `psuf_naloga1_Ime_Priimek.pdf`.
- **Zadeva**: PSUF naloga 1 Ime Priimek na moj mail: `jan.gavranovic@ijs.si`.

Zaključek 1. vaj

- Splošna vprašanja → učilnica (mogoče ima še kdo enake težave) - spodbujamo pomaganje!
- Za ostala vprašanja me dobite na `jan.gavranovic@ijs.si` ali pa v pisarni C82 na IJS.
- 2. del vaj prihodnji teden (v petek) → začnemo z naprednejšimi metodami.
- Do takrat: preberite navodila, postavite si Python okolje, preizkusite kako stvari delujejo, spoznajte se s podatki in kodo, narišite histograme, naredite fite za signal in ozadje.