# Determination of *pp*-collision time with TOF detector

Semyon Yurchenko

Saint Petersburg State University

*sem2600@mail.ru*
*Work done during START program in JINR*

August 16, 2022

# Task and initial conditions

Using information about particles trajectories and hits from TOF detector determine time of *pp*-collision.

1. In this work only tracks with momentum below 500 MeV will not be considered.
2. Resolution of TOF detector $\sigma_t = 70\ ps$.
3. Momentum resolution: $\frac{\sigma_p}{p} = 5\%$
4. TOF radius is 1 $m$ and length of 3.772 $m$.

# Plan of simulation

1. Get tracks of charged particles with momentum over 500 MeV from Pythia8 events with $\sqrt{s} = 27\ GeV$.
2. Calculate intersection point with TOF detector($t_i$).
3. Calculate arc length of trajectory($L_i$).
4. Smear $t_i$ with $N(t_i, 70\ ps)$ and $p_i$ with $N(p_i, 0.05 \cdot p_i)$.
5. Using information about arc lengths of trajectories, TOF hits and particle momentum determine time of *pp*-collision($t_0$).

# Plan of simulation

1. Get tracks of charged particles with momentum over 500 MeV from Pythia8 events with $\sqrt{s} = 27\ GeV$.
2. Calculate intersection point with TOF detector($t_i$).
3. Calculate arc length of trajectory($L_i$).
4. Smear $t_i$ with $N(t_i, 70\ ps)$ and $p_i$ with $N(p_i, 0.05 \cdot p_i)$.
5. **Using information about arc lengths of trajectories, TOF hits and particle momentum determine time of *pp*-collision($t_0$).**

# Brute force

How brute force works:

1. Choose particles types to make tof hypotheses$\rightarrow [\pi^{\pm}, K^{\pm}, p^{\pm}]$.

$$tof_{ik} = \frac{L_i}{c}\sqrt{1 + \frac{m_k^2}{p_i^2}} \quad (1)$$

2. For every event check all tracks hypotheses combinations - $3^N$ variants.
3. On every step calculate $t_0$ and $\chi^2$ and find $\chi^2_{min}$.

$$\chi^2 = \sum^N \frac{(t_0 + tof_{ik} - t_i)^2}{\sigma_t^2}, \quad t_0 = \sum^N \frac{t_i - tof_{ik}}{N} \quad (2)$$

4. Time complexity $O(N \cdot 3^N)$.

# Brute force

How brute force works:

1. Choose particles types to make tof hypotheses $\rightarrow [\pi^{\pm}, K^{\pm}, p^{\pm}]$.

$$tof_{ik} = \frac{L_i}{c}\sqrt{1 + \frac{m_k^2}{p_i^2}} \tag{1}$$

2. For every event check all tracks hypotheses combinations - $3^N$ variants.
3. On every step calculate $t_0$ and $\chi^2$ and find $\chi^2_{min}$.

$$\chi^2 = \sum^{N} \frac{(t_0 + tof_{ik} - t_i)^2}{\sigma_t^2}, \quad t_0 = \sum^{N} \frac{t_i - tof_{ik}}{N} \tag{2}$$

4. Time complexity $O(N \cdot 3^N)$ - **very slow!!!**

# Genetic algorithm

How genetic algorithm works:

1. Create population of random candidate solutions - $v([m_i]_k)$.
2. Create mutant vector from random candidates in population(DE-inspired):

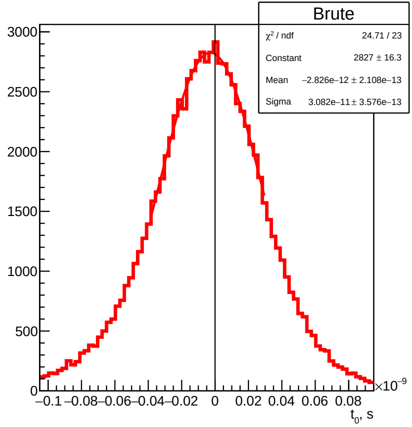
$$v_{mut} = v_r + F \cdot (v_p - v_q) \tag{3}$$

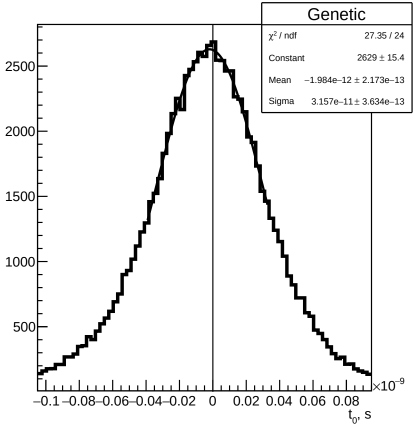
3. Check if $\chi^2_{mut} < \chi^2_r$ then replace $v_r$ with $v_{mut}$. If not, population remains unchanged - **Darwinian selection**.
4. Repeat
5. After some number of steps stop and choose $\chi^2_{min}$ as an answer.
6. Time complexity - $O(N \cdot N_{population} \cdot N_{steps})$, $800 < N_{steps} < 1000$.

# Genetic algorithm vs Brute force

- Brute force gives solution with minimal $\chi^2$, but very slowly.
- Genetic algorithm has less accuracy, but much faster.

Comparison of 2 algorithms was done on events with number of tracks $4 < N < 15$.



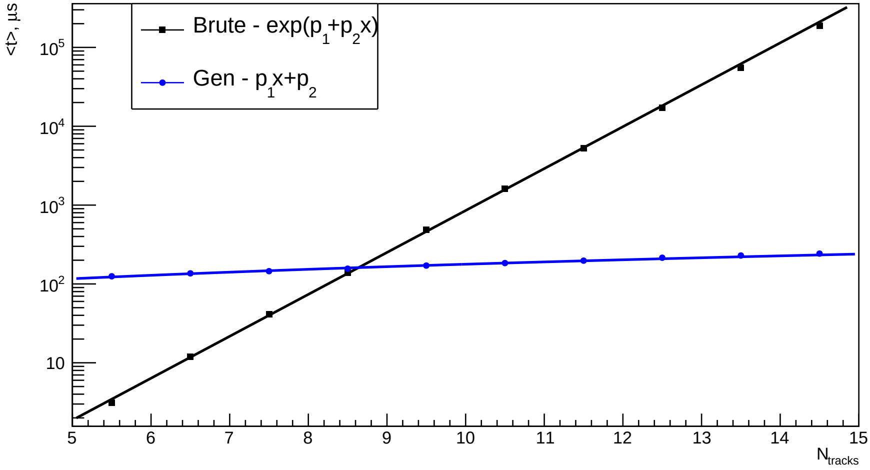
# Genetic algorithm vs Brute force

1. Brute Force resolution of $t_0$ is 34 ps and Genetic Algorithm is 37 ps.
2. PID event efficiency for Brute Force is 35.0% and for Genetic algorithm it is the same.
3. PID track efficiency for Brute Force is 86.2% and for Genetic algorithm is 85.8%.

PID event efficiency - percentage of events where all tracks were guessed correctly.

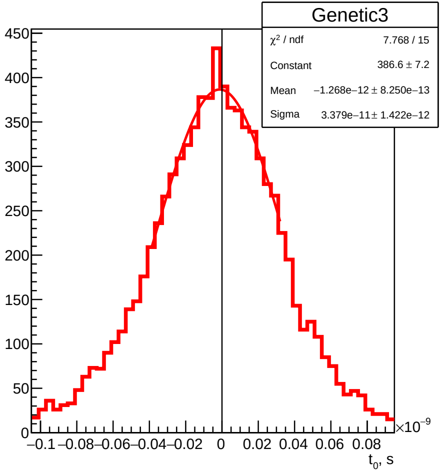
PID track efficiency - percentage of tracks that were guessed correctly.

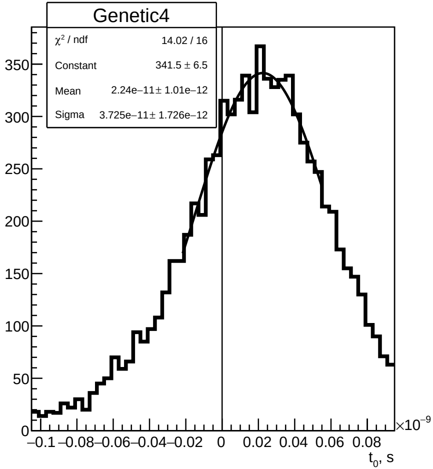
# Time complexity



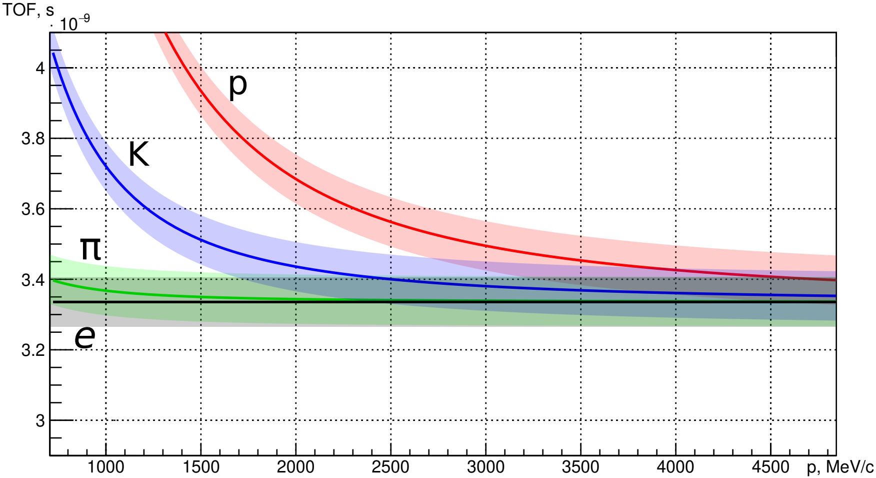
# Different hypotheses

Genetic3 hypotheses is $[\pi^{\pm}, K^{\pm}, p^{\pm}]$
Genetic4 hypotheses is $[e^{\pm}, \pi^{\pm}, K^{\pm}, p^{\pm}]$



# TOFs


TOF, s
· 10^-9
4
3.8
3.6
3.4
3.2
3
1000
1500
2000
2500
3000
3500
4000
4500
p, MeV/c
p
K
π
e

# Summary

1. In events with low multiplicity($4 < N < 15$) Brute Force on average spend 5 ms per event, and Genetic algorithm 0.160 ms.
2. Run time grows slowly as function of multiplicity for Genetic algorithm.
3. Brute Force resolution of $t_0$ is 34 ps and Genetic Algorithm is 37 ps.
4. PID event efficiency for Brute Force is 35.0% and for Genetic algorithm it is the same.
5. PID track efficiency for Brute Force is 86.2% and for Genetic algorithm is 85.8%.

# Outlook

1. Compare DE-inspired algorithm with other types of Genetic algorithms.
2. Optimise Genetic algorithm to decrease run time so it can be used online.

# Thank you for your attention!