JuliaHEP 2023 Workshop
08.11.2023

Cumulants & Co.

Piotr Kalaczyński on behalf of Krzysztof Domino, Piotr Gawron, Łukasz Pawela, Dariusz Kurzyk, Tony Kelman, Alex Arslan and Kristoffer Carlsson

ASTROCENT
Particle Astrophysics Science and Technology Centre
International Research Agenda

Nicolaus Copernicus Astronomical Center
Polish Academy of Sciences

IITiS

European Funds Smart Growth

Republic of Poland

FNP Foundation for Polish Science

European Union European Regional Development Fund

Introduction

Code presented here was developed at IITiS in Gliwice, Poland

Based on: arXiv:1301.7744v3
Exploiting Symmetry in Tensors for High Performance: Multiplication with Symmetric Tensors
(M. D. Schatz, T. M. Low, R. A. van de Geijn, T. G. Kolda)

By:
- Krzysztof Domino
- Piotr Gawron
- Łukasz Pawela
- Tony Kelman
- Alex Arslan
- Kristoffer Carlsson

NOT by me!

Repos:
- github.com/iitis/Cumulants.jl → arXiv:1701.05420v4
- github.com/iitis/CumulantsFeatures.jl → doi/10.1016/j.physa.2020.124995, arXiv:1808.03513v1
- github.com/iitis/CumulantsUpdates.jl → doi/10.2478/amcs-2019-0015

Me:

Just a humble technician in the Scientific Computing & IT Group at ASTROCENT in Warsaw, Poland

Particle Astrophysics Science and Technology Centre
International Research Agenda

current focus: maintenance of python & C++ simulation codes,
excited to dive deeper into Julia ☺

Cumulants

What are cumulants anyway?

quantitative measures of the shape of the distribution

(an alternative to the moments)

Can be generated from a Taylor expansion:

$$H(t) = \sum_{d=1}^{\infty} \kappa_d \frac{(it)^d}{d!} = \underset{\kappa_1}{\underset{\|}{\mu}} it - \underset{\kappa_2}{\underset{\|}{\sigma^2}} \frac{t^2}{2} + \cdots$$

d – order of the cumulant

which must be differentiated and evaluated at $t = 0$ to extract the cumulants κ_d

(hello Julia ☺)

The symbols may seem familiar for a good reason:

$\kappa_1 = \mu$: mean

$\kappa_2 = \sigma^2$: variance

$\kappa_3 \sim$ skewness (after rescaling)

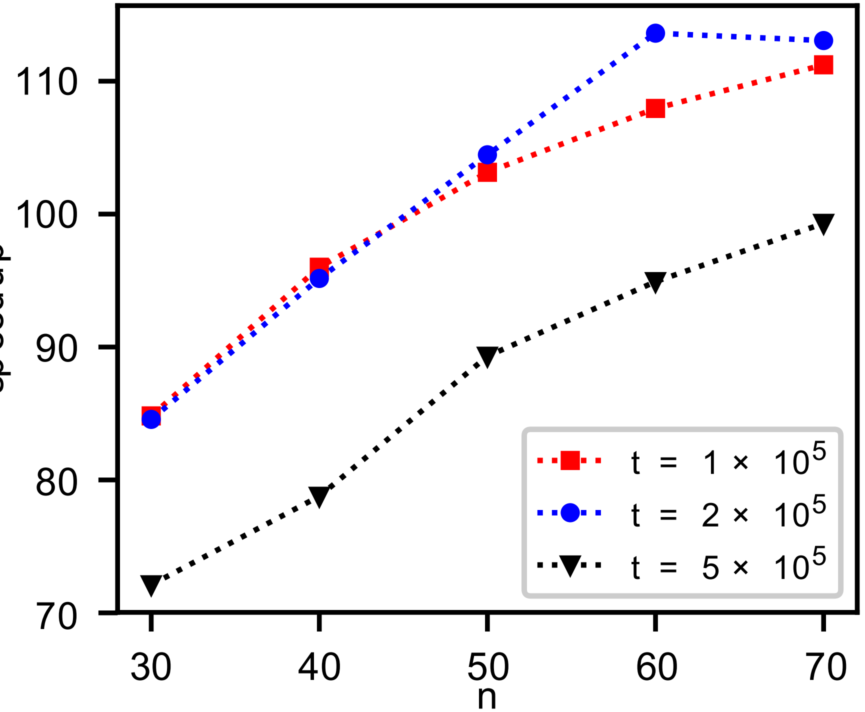
$\kappa_4 \sim$ kurtosis (after rescaling)

no special names for higher orders (that I know of …)

from arXiv:1701.05420v4

Storage of symmetric tensors is optimized wrt. the naïve storage scheme (storing all elements)

Expected computational complexity reduction: by a factor $d!$



(wrt. to naïve)

number of marginal variables

Here :

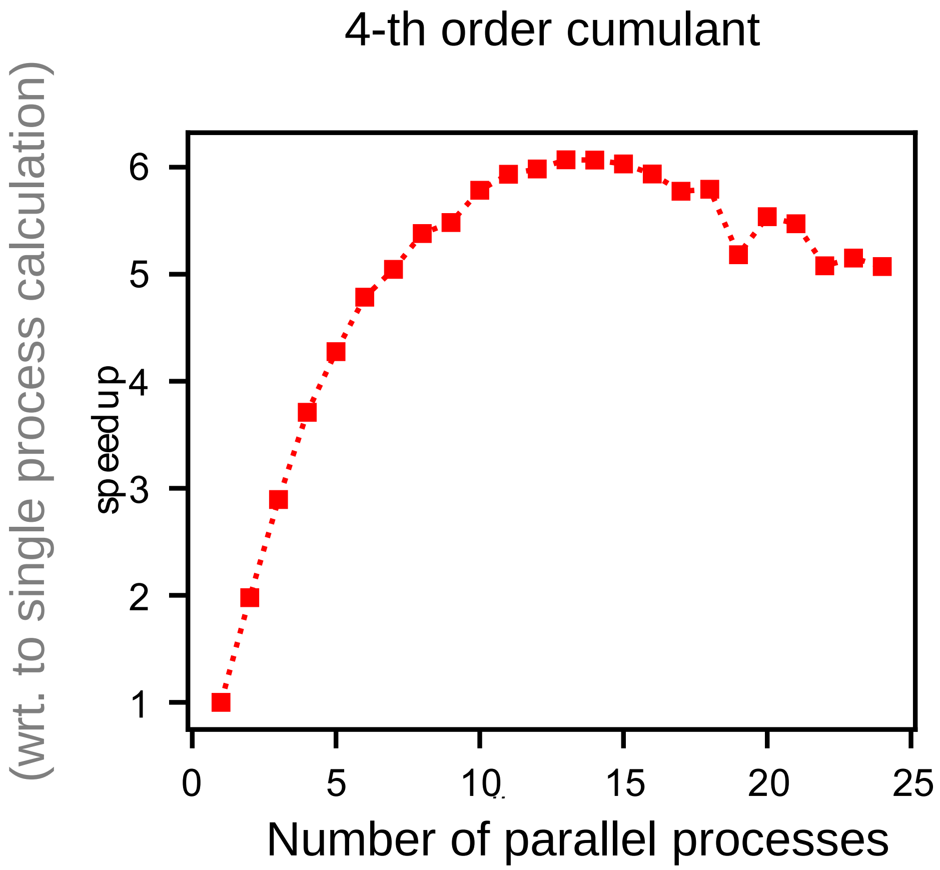
- Computation of the 4th cumulant (κ_4) tensor
- Actual speedup exceeds $d! = 4! = 24$

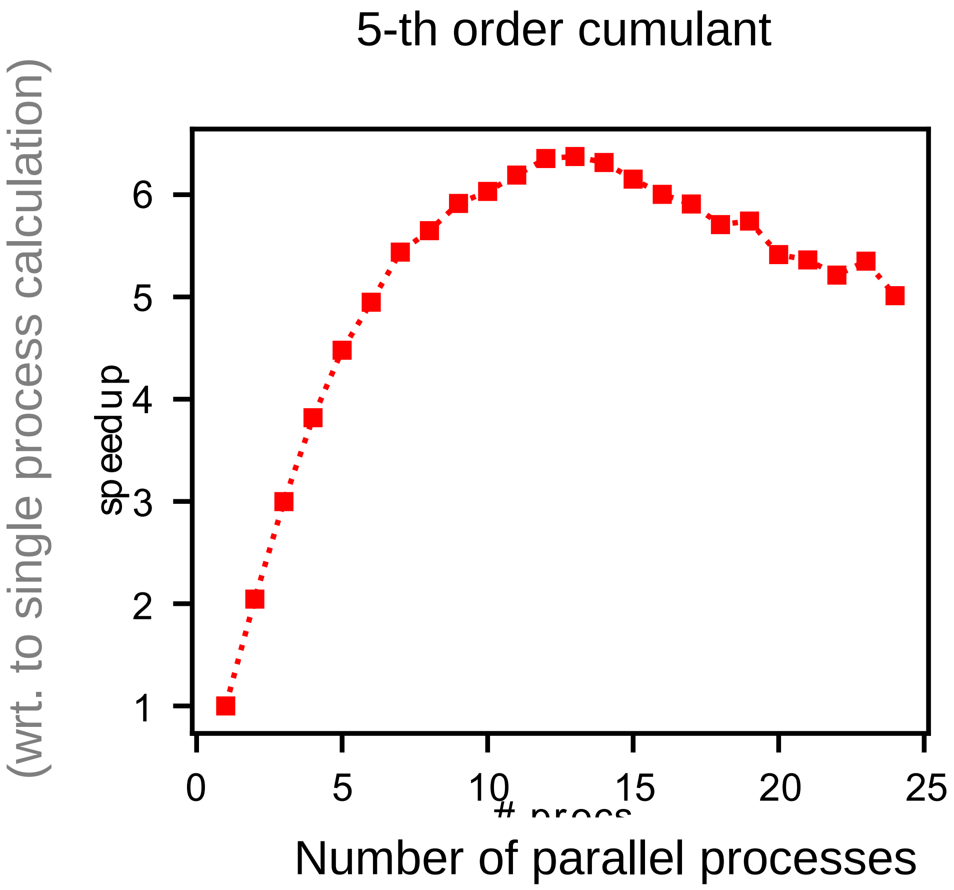
t – number of data samples

from [arXiv:1701.05420v4]

The code is not completely parallelisable, but can be accelerated by running multiple processes

speedup: with respect to single process calculation of cumulants





Performance: Julia vs R implementation

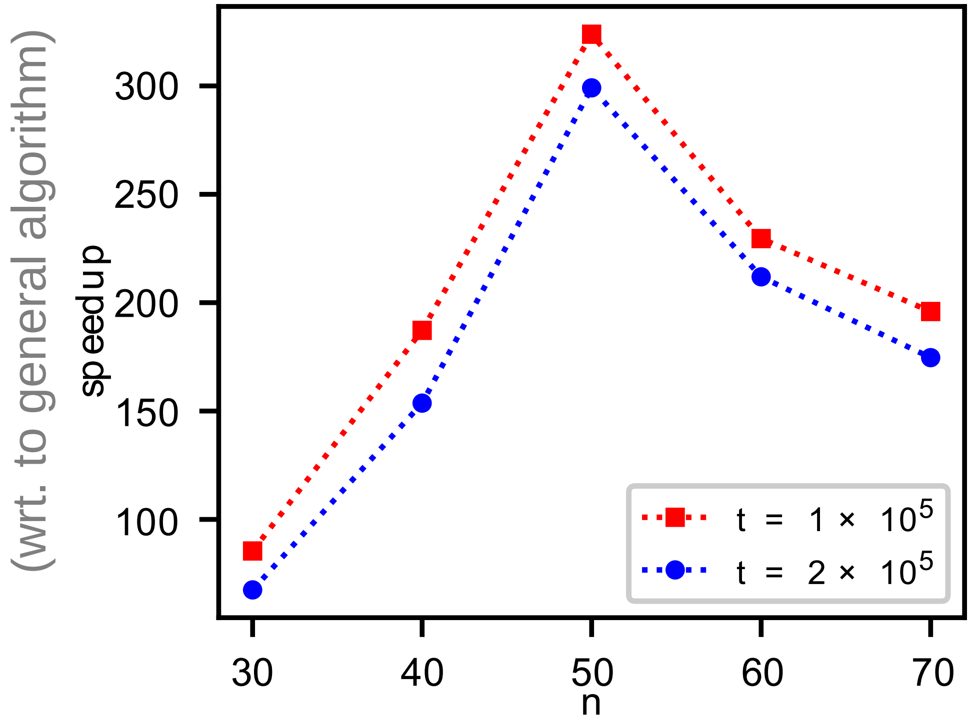
from arXiv:1701.05420v4

Speedup wrt. state of the art:

- Cumulants.jl vs the general (arbitrary cumulant order) algorithm from here
- 4th cumulant tensor

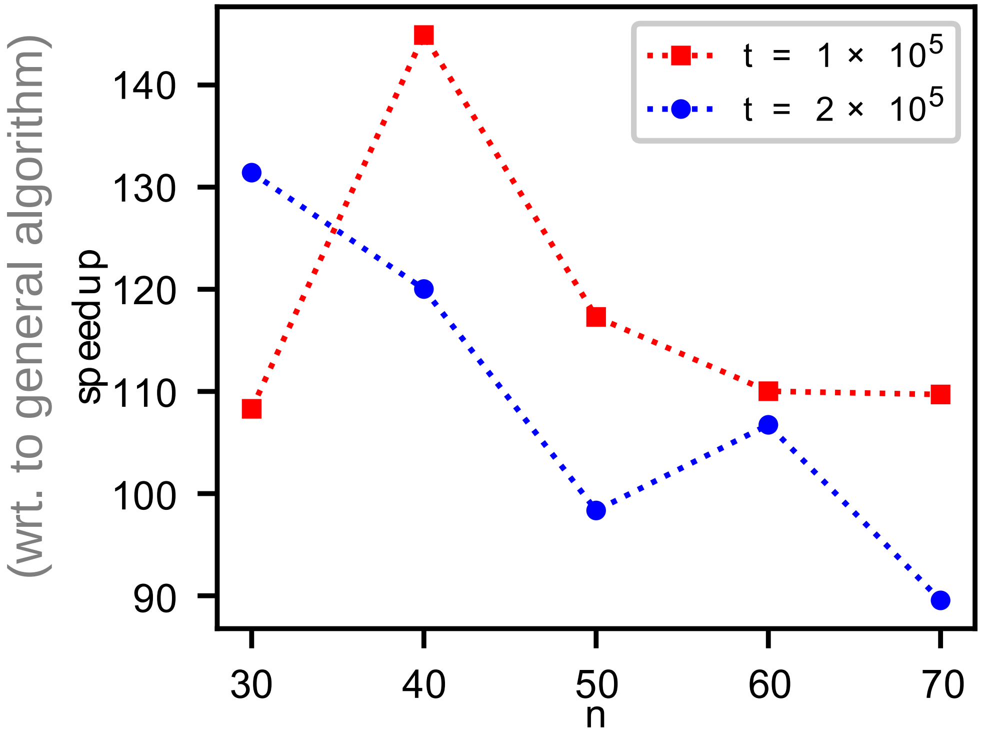
t – number of data samples

Comparing with julia reimplementation



number of marginal variables

Comparing with R reimplementation

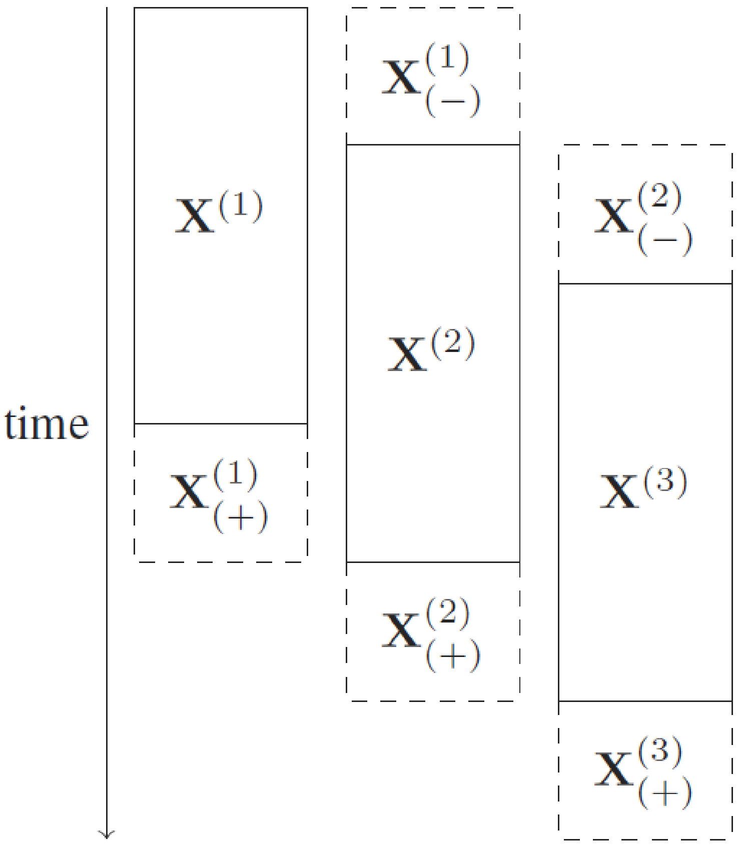


number of marginal variables

from doi/10.2478/amcs-2019-0015

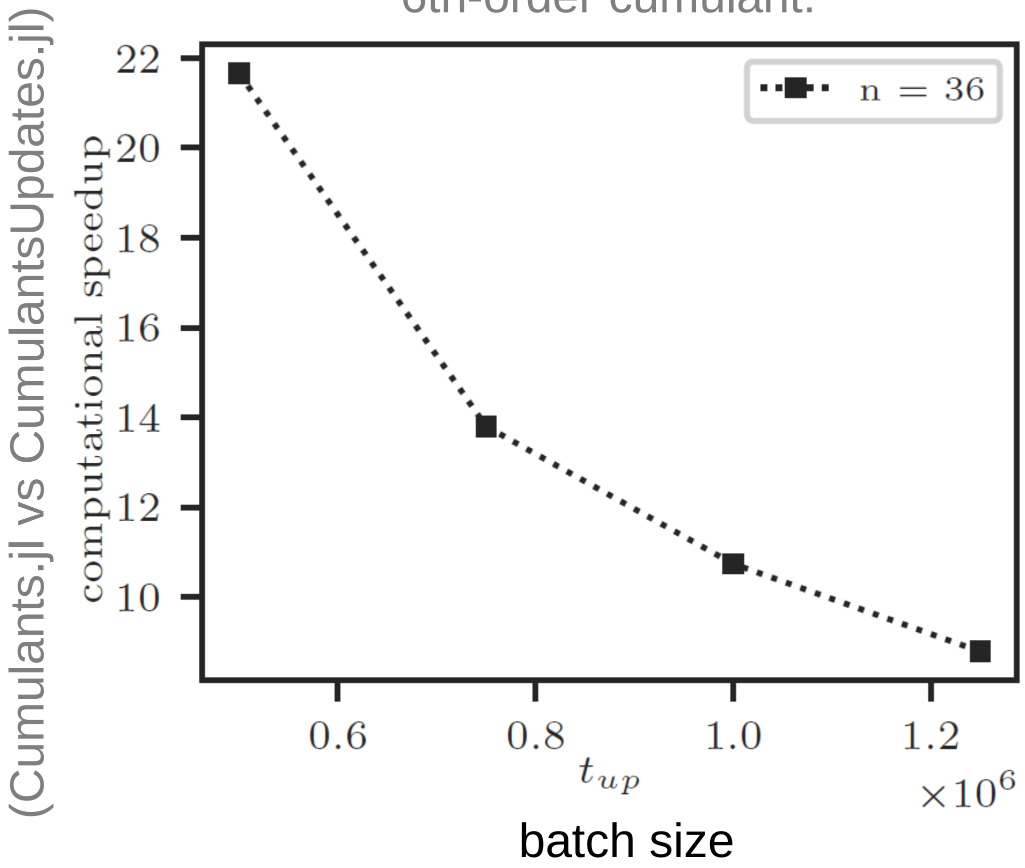
Cumulants can be computed online, in a sliding window of datastreams:

(CumulantsUpdates.jl)



… and this turns out to be even faster than Cumulants.jl:

6th-order cumulant:



$t = 2.5 \cdot 10^7$ – number of data samples

n – numer of marginal variables

Applications

Okay, where in HEP can this be used? (just some examples)

some analyses use cumulants up to 8th order

↓

specialised algorithm (orders 1-4) NOT enough!

- At colliders:
  - arXiv:1109.0593v2
    Error Estimation for Moments Analysis in Heavy-Ion Collision Experiments
    (Xiaofeng Luo)
  - arXiv:2305.13874v2
    Holographic study of higher-order baryon number susceptibilities at finite temperature and density
    (Z. Li, J. Liang, S. He, L. Li)
  - arXiv:2303.13414v1
    Higher-order correlations between different moments of two flow amplitudes in Pb−Pb collisions at $\sqrt{s_{\mathrm{NN}}} = 5.02$ *TeV*
    (ALICE Collaboration)
  - arXiv:2209.11940v2
    Higher-Order Cumulants and Correlation Functions of Proton Multiplicity Distributions in $\sqrt{s_{\mathrm{NN}}} = 3$ *GeV Au+Au Collisions at the RHIC STAR Experiment*
    (STAR Collaboration)
- In astrophysics:
  - arXiv:2204.05305v3
    Galaxy and halo angular clustering in LCDM and Modified Gravity cosmologies
    (P. Drozda, W. A. Hellwing, M. Bilicki)
  - arXiv:2209.14810v2
    Magnetic helicity fluxes from triple correlators
    (K. Gopalakrishnan, K. Subramanian)
- Lattice QCD:
  - arXiv:2305.10916v2
    Microscopic Encoding of Macroscopic Universality: Scaling Properties of Dirac Eigenspectra near QCD Chiral Phase Transition
    (H. Ding, W. Huang, S. Mukherjee, P. Petreczky)

Other projects

There is more if you're into …

Quantum computing: QuantumInformation.jl

from arXiv:1806.11464v3

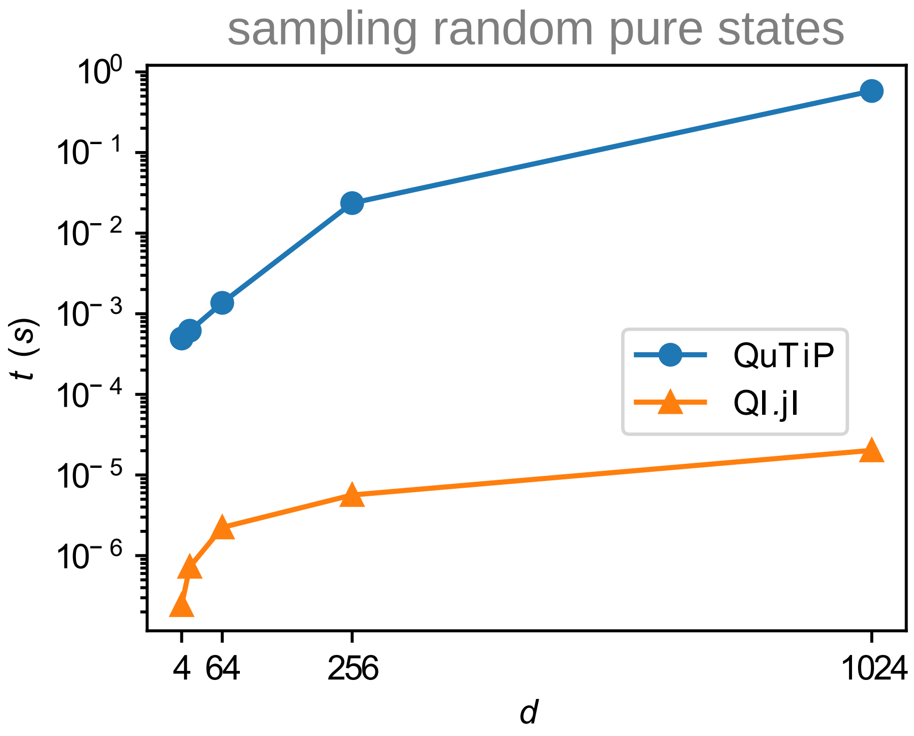
❖ natively works with bra-ket notation!

```
julia> ket(1,2)
2-element Array{Complex{Float64},1}:
 1.0 + 0.0im
 0.0 + 0.0im

julia> (1/sqrt(2)) * (ket(1,2) + ket(2,2))
2-element Array{Complex{Float64},1}:
 0.7071067811865475 + 0.0im
 0.7071067811865475 + 0.0im
```

❖ comes with many convenient implementations of states, transformations, etc.

❖ faster than QuTiP (python code)

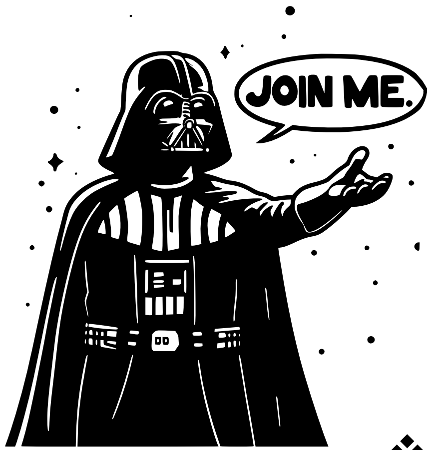


Random matrix sampling on GPU: MatrixEnsembles.jl

Summary

To sum up:

- Need to compute high-order cumulants or just do it fast? Use Cumulants.jl etc. ☺
- Want to learn more? See:
  arxiv/1701.05420
  doi/10.2478/amcs-2019-0015



My ideas for new projects:

- CORSIKA alternative in Julia:
  - I'm aware of C++ rework (C8)
  - but … maybe it can be more efficient? (looped differentiation and matrix operations under the hood …)
  - and just easier (MUCH friendlier syntax)
  - healthy competition never hurts ;-)
- Simulation of acoustic signal from HE particles:
  - thermo-acoustic mechanism: known & measured
  - simulated, but nothing really open-source and general purpose, like CORSIKA (at least that I am aware of …)
  - combined simulation of light & sound? (could allow for better event reco)

Thank you for your attention!

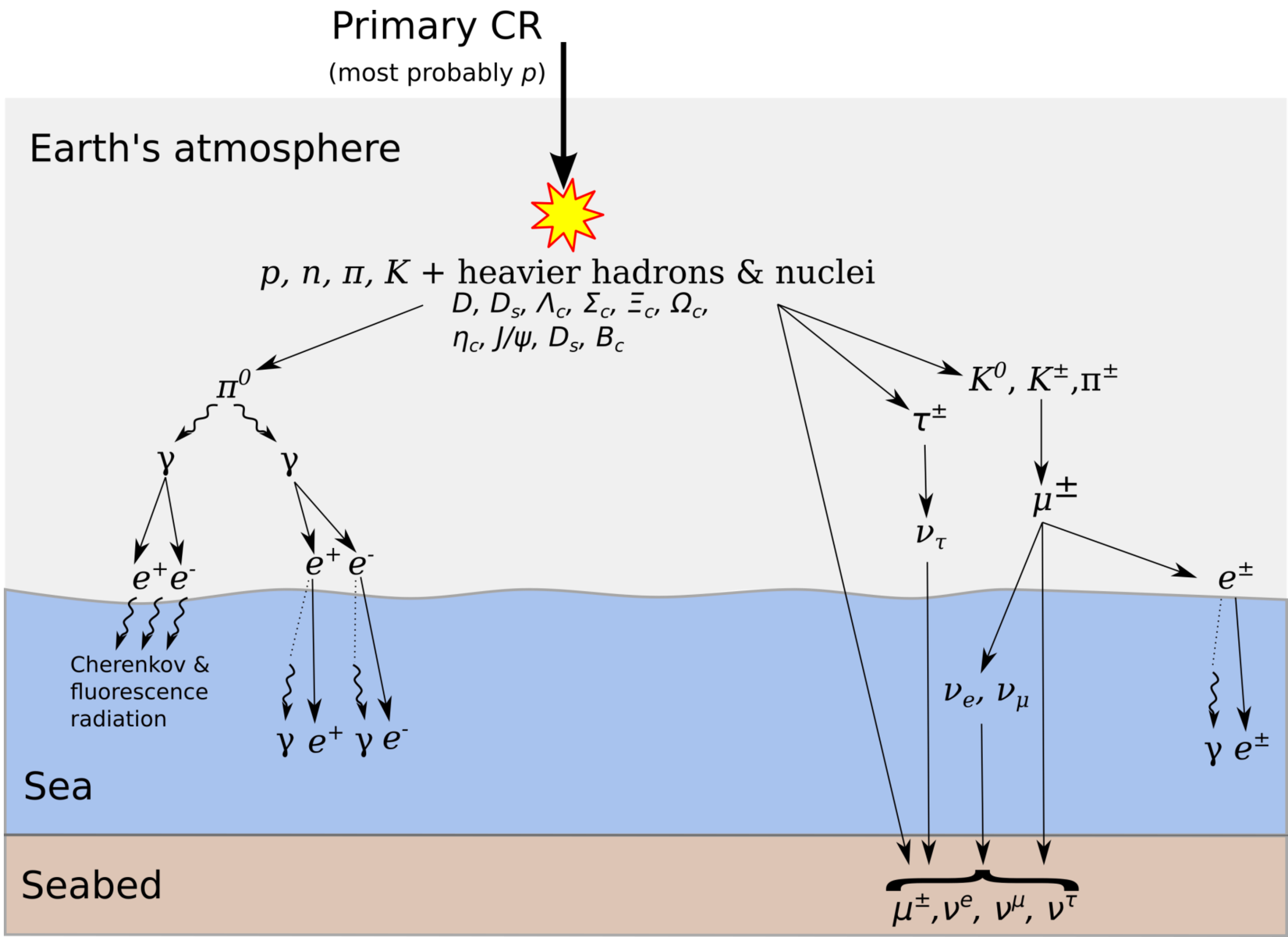
European Funds Smart Growth
Republic of Poland
FNP Foundation for Polish Science
European Union European Regional Development Fund

Backup

Extensive Air Showers

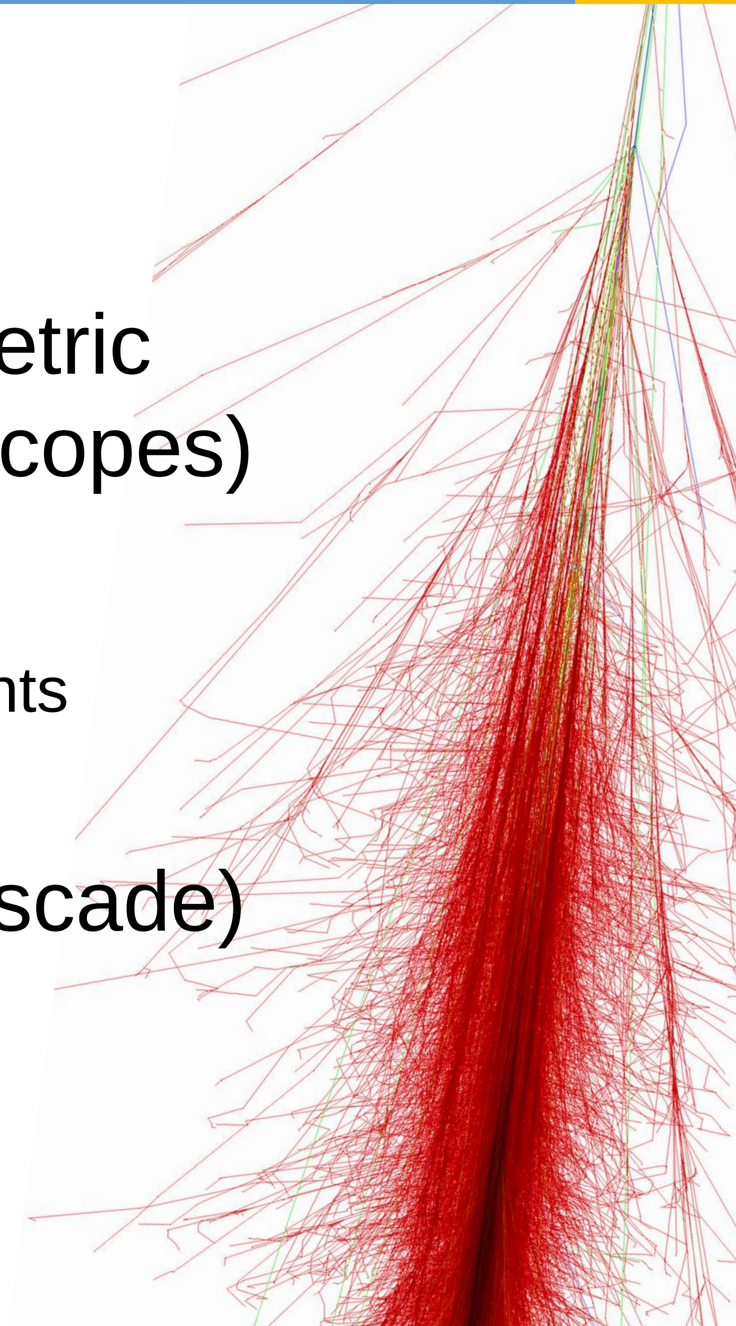
EAS:

- Caused by primary CR
- Typically start at $h{\sim}30-40\text{km}$
- 3 main components:
  - electromagnetic (EM)
  - hadronic
  - muonic
- Simulated with CORSIKA



We have 2 options:

1. MUPAGE (atmospheric **MU**ons from **PA**rametric formulas: a fast **GE**nerator for neutrino telescopes)
   - developed for ANTARES
   - fast muon MC generator
   - based on parametric formulas and MACRO measurements
   - parameters can be freely tuned
2. CORSIKA (**CO**smic **R**ay **SI**mulations for **KA**scade)
   - developed for KASCADE
   - full simulation of air showers
   - customizable (models, primaries, etc.)



git.km3net.de/opensource/gseagen

gSeaGen

Code for propagating muons and/or neutrinos to neutrino telescopes. Developed for KM3NeT, but applicable to other experiments

My work in this project:

- Implement processing of CORSIKA showers
- Speed, memory & storage optimization
- Rework of the geometry: no more flat Earth!
- Code maintenance

A paper by me under internal review …

Tech stack:

- C++
- ROOT
- PERL
- PROPOSAL
  (github.com/tudo-astroparticlephysics/PROPOSAL)

Current devs:

- Carla Distefano
- Alfonso Andres Garcia Soto
- Piotr Kalaczyński
- Johannes Schumann
- Rodrigo Garcia
- Andrey Romanov