# Towards full electromagnetic physics vectorisation in the GeantV transport framework

Marilena Bandieramonte
(marilena.bandieramonte@cern.ch)
on behalf of the GeantV development team

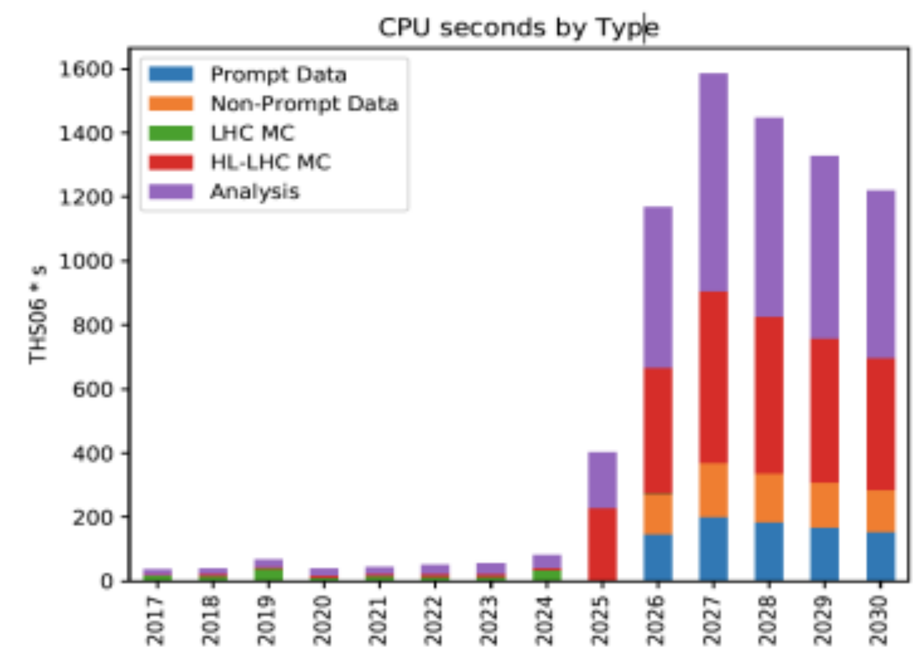
**CHEP2018 - 10th July 2018, Sofia**

# INTRODUCTION


marilena.bandieramonte@cern.ch

# INTRODUCTION

- Event simulation is one of the **most time consuming** parts of the workflow, in the HEP sw ecosystem
  - For high-luminosity LHC phase (HL-LHC), the upgraded experiments expect to collect **150 times more data** than in Run 1
- The **GeantV R&D project** was launched in 2013, aiming at exploring emerging computer technologies in order to significantly increase run-time performance of detector simulation

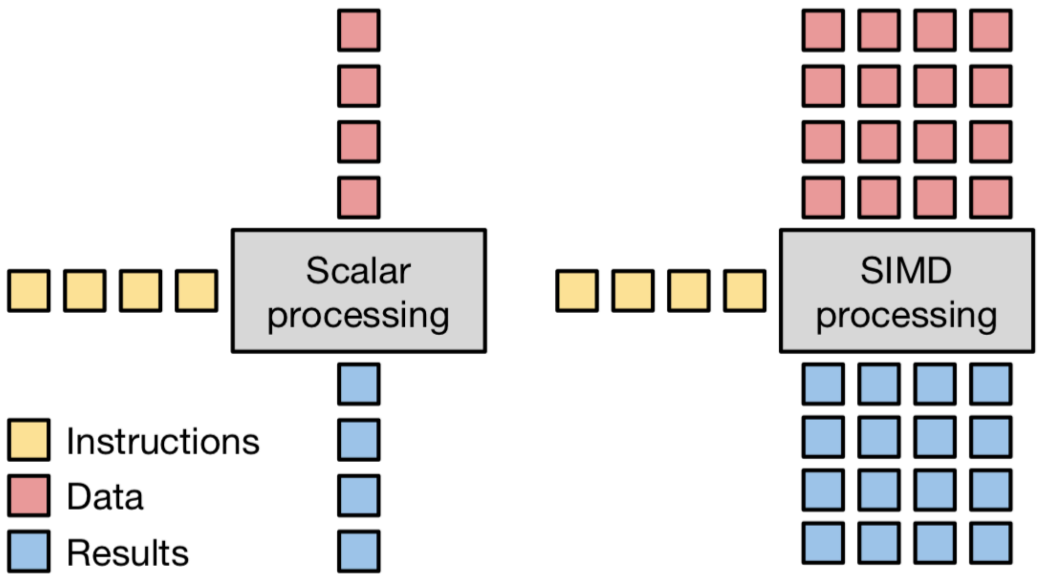



**CMS and Atlas estimated CPU needs for HL-LHC (source: CWP)**


marilena.bandieramonte@cern.ch

# INTRODUCTION

- Event simulation is one of the **most time consuming** parts of the workflow, in the HEP sw ecosystem
  - For high-luminosity LHC phase (HL-LHC), the upgraded experiments expect to collect **150 times more data** than in Run 1
- The **GeantV R&D project** was launched in 2013, aiming at exploring emerging computer technologies in order to significantly increase run-time performance of detector simulation
- The project studies performance gains when changing the classic particle transport approach, propagating **multiple tracks from multiple events in parallel**
  - improving code and data locality in the process
  - enabling SIMD/SIMT execution models: Vectorization+Multithreading
- **Vectorization of physics library** is important as key part of the algorithmic chain





marilena.bandieramonte@cern.ch

# WHEN CAN WE PROFIT FROM VECTORIZATION

- Functions with many **math computations**
  - Such as +, *, /, sqrt, sin, cos, exp, log (ordered according to approximate computation complexity)
- Functions with **minimal branching**
  - Branching **may** require to evaluate both branches for vectorized code

**Scalar code**

```
if (cond > rndArray[0]) {
  eps  = Math::Exp(-al1 * rndArray[1]);
  eps2 = eps * eps;
} else {
  eps2 = eps02 + (1. - eps02) * rndArray[1];
  eps  = std::sqrt(eps2);
}
```

**Vector code**

```
MaskD_v cond1 = cond > rnd1;
if (!MaskEmpty(cond1)) {
  vecCore::MaskedAssign(eps, cond1, Math::Exp(-al1 * rnd2));
  vecCore::MaskedAssign(eps2, cond1, eps * eps);
}
if (!MaskEmpty(!cond1)) {
  vecCore::MaskedAssign(eps2, !cond1, eps02 + (1.0 - eps02) * rnd2);
  vecCore::MaskedAssign(eps, !cond1, Math::Sqrt(eps2));
}
```

- Functions not bounded by **memory access**
  - Load 4 doubles into SIMD register is one instruction but it is not faster than loading values one by one


marilena.bandieramonte@cern.ch

# GEANTV EM PHYSICS LIBRARY

**Current State**

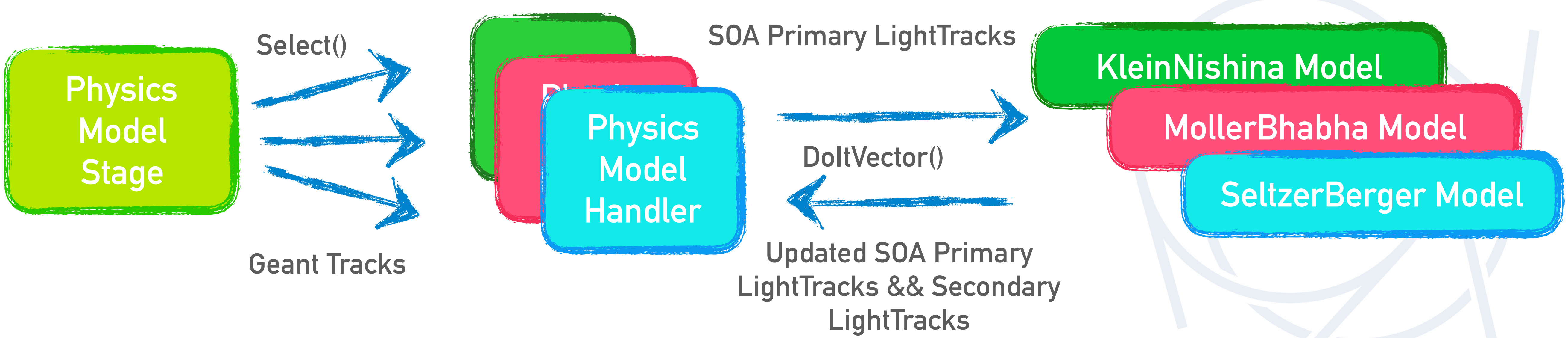
| particle | processes | model(s) GeantV | model(s) Geant4 |
|---|---|---|---|
| $e^-$ | ionisation | Møller [100eV-100TeV] | Møller [100eV-100TeV] |
| | bremsstrahlung | Seltzer-Berger [1keV-1GeV] | Seltzer-Berger [1keV-1GeV] |
| | | Tsai (Bethe-Heitler) w. LPM. [1GeV-100TeV] | Tsai (Bethe-Heitler) w. LPM. [1GeV-100TeV] |
| | Coulomb sc. | GS MSC model [100eV-100TeV] | Urban MSC model [100eV-100MeV] |
| | | | Mixed model [100MeV-100TeV] |
| $e^+$ | ionisation | Bhabha [100eV-100TeV] | Bhabha [100eV-100TeV] |
| | bremsstrahlung | Seltzer-Berger [1keV-1GeV] | Seltzer-Berger [1keV-1GeV] |
| | | Tsai (Bethe-Heitler) w. LPM. [1GeV-100TeV] | Tsai (Bethe-Heitler) w. LPM. [1GeV-100TeV] |
| | Coulomb sc. | GS MSC model [100eV-100TeV] | Urban MSC model [100eV-100MeV] |
| | | | Mixed model [100MeV-100TeV] |
| | annihilation | Heitler ($2\gamma$) [0-100TeV] | Heitler ($2\gamma$) [0-100TeV] |
| $\gamma$ | photoelectric | Sauter-Gavrila + EPICS2014 [1eV-100TeV] | Sauter-Gavrila + EPICS2014 [1eV-100TeV] |
| | incoherent sc. | Klein-Nishina$^+$ [100eV-100TeV] | Klein-Nishina$^+$ [100eV-100TeV] |
| | $e^-e^+$pair production | Bethe-Heitler$^+$ [100eV-80GeV] | Bethe-Heitler$^+$ [100eV-80GeV] |
| | | Bethe-Heitler$^+$ w. LPM [80GeV-100TeV] | Bethe-Heitler$^+$ w. LPM [80GeV-100TeV] |
| | coherent sc. | - | Livermore |
| + | energy loss fluct. | - | Urban |

- Every model is **tested and verified against the corresponding Geant4** model (cross section per atom, cross section per volume, and kinematic of primary and secondary particles)
- **EM showers** in GeantV can be **fully simulated** in real applications (i.e. FullCMS, TestEM3, TestEM5, FullLHCb) and the results are verified against the corresponding Geant4 simulation


marilena.bandieramonte@cern.ch

# ELECTROMAGNETIC PHYSICS – FINAL STATE GENERATION

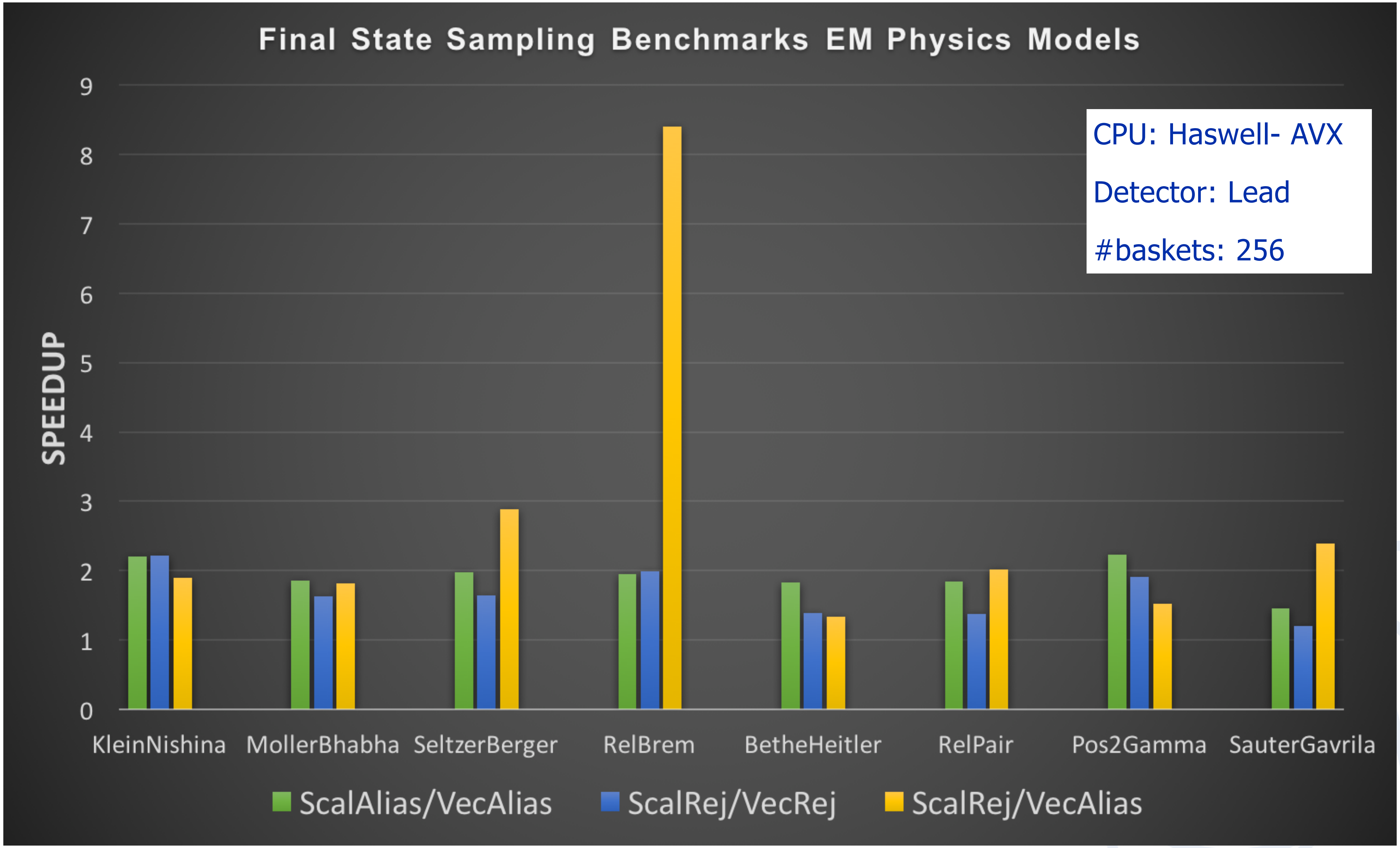
- Once the particle undergoes a physics process, the **final state generation stage** occurs:
  - Differential cross sections are used to update the **kinematic properties** of the primary particle and to **produce secondary particles** (if necessary)
  - Sampling with rejection / Sampling with Alias tables + approximations
- The final state generation involves **two main** parts in GeantV:
  - **Framework** part: filtering of tracks (according to particles type, physics process), gathering relevant track information into SOA form (LightTracks), call the physics model
  - **Model** part: Specific code for the selected physics model: update primaries, create secondaries..





marilena.bandieramonte@cern.ch

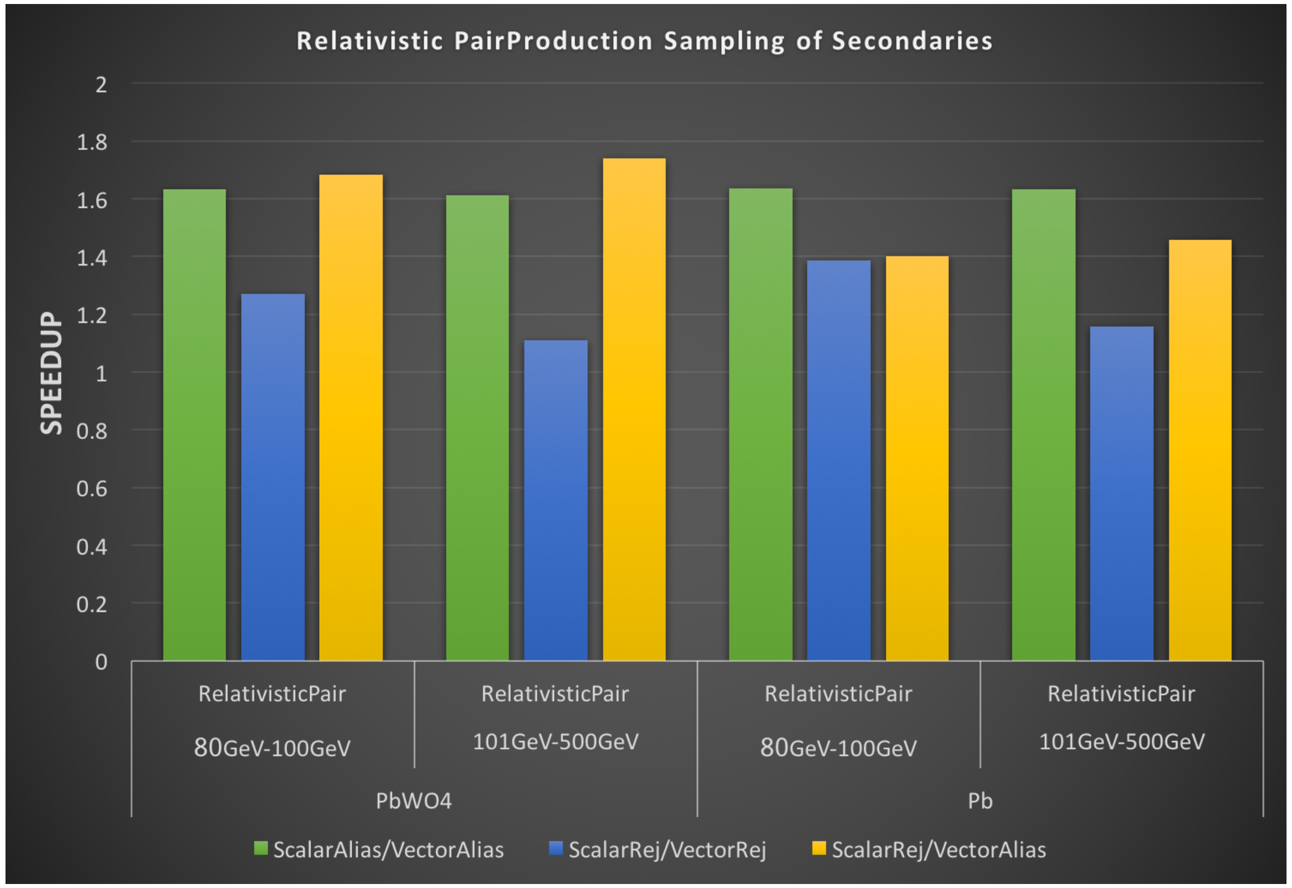
# MODEL LEVEL TEST BENCHMARKS

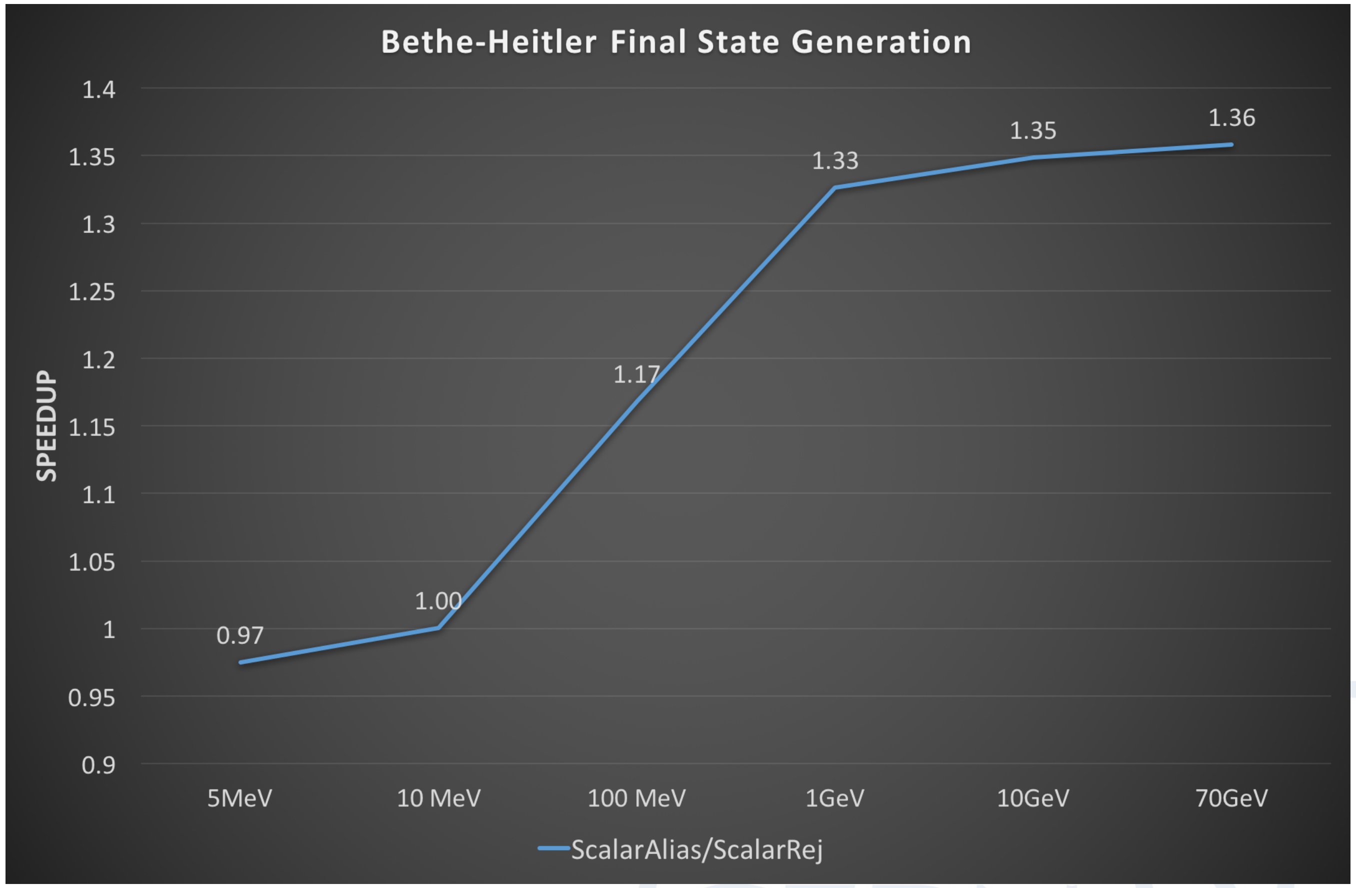



See next talk, from A. Gheata for benchmarks on the full simulation chain


marilena.bandieramonte@cern.ch

# TUNING THE SIMULATION THROUGH MODELS




CPU: Haswell - AVX
Detector: Pb/PbWO4
Model for Energy Range [80GeV-100TeV]
#baskets: 256

CPU: Haswell - AVX
Detector: Pb
Model for Energy Range [2 $e^-{}_m c^2$-80GeV]
Scalar execution


marilena.bandieramonte@cern.ch

# SUMMARY

- Physics model integrated in the GeantV framework and validated against corresponding Geant4 simulation
- Most of the **EM physics library** models are now **vectorized**
  - Work in progress on Multiple scattering
  - Optimization of the current vectorized implementations
- SpeedUp
  - At the level of final-state EM Model: between 1.5-3 on Haswell, 2-4 on Skylake with AVX2
  - See A. Gheata talk for the impact on realistic EM showers in calorimeters and fullCMS applications

# WORK IN PROGRESS

- Work on **other parts** of the physics framework
  - ComputeIntLenght(), generation and insertion of secondary particles, etc..
- VecMath library and Vectorized pRNG (handling reproducibility issues)
- Study the possibility to substitute double precision computations with single one, in some parts of the physics library (i.e. transport in magnetic field)
- Add **AVX512** support (UME::SIMD)
  - currently there is no way to test consistently AVX512

**GeantV project is hosted at: https://gitlab.cern.ch/GeantV/geant**
**GeantV website: http://geant.cern.ch**


marilena.bandieramonte@cern.ch

# THANKS FOR THE ATTENTION!

# QUESTIONS?

marilena.bandieramonte@cern.ch

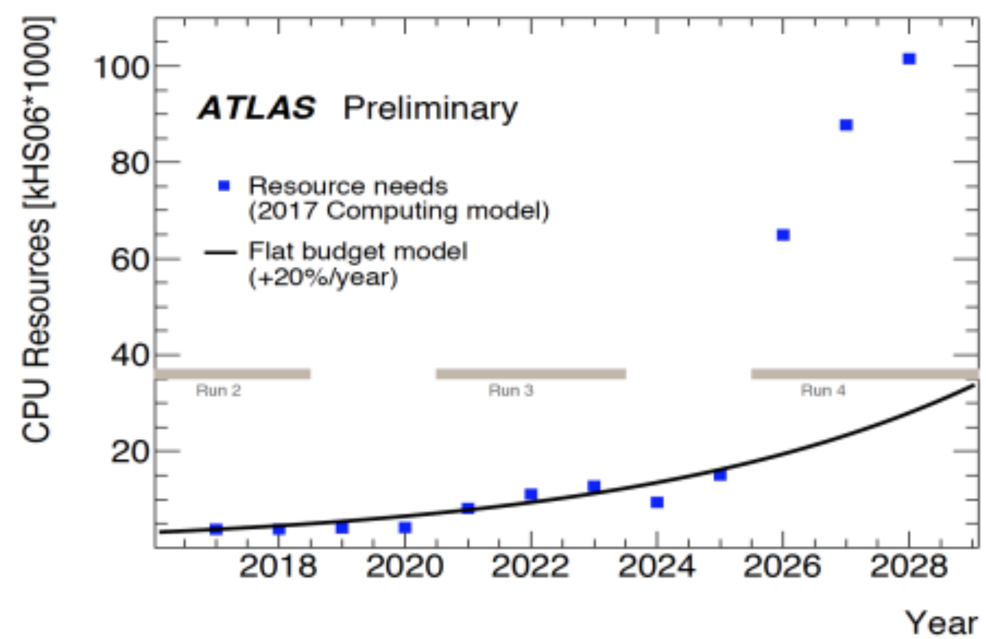
# BACKUP

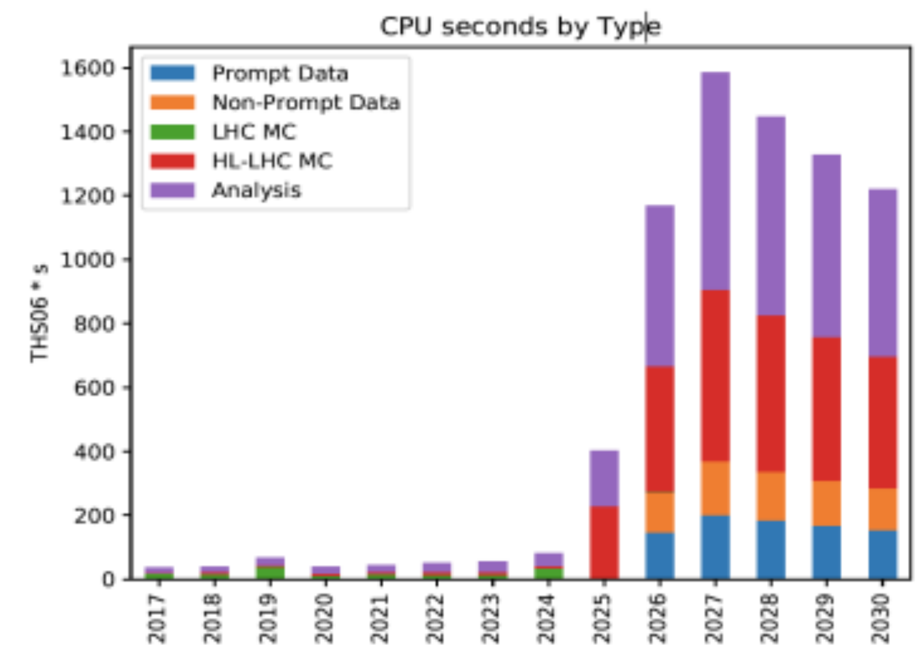
marilena.bandieramonte@cern.ch

# NEED FOR FASTER SIMULATION CODE FOR HEP COMMUNITY

- During the first two runs, the LHC experiments produced, reconstructed, stored, transferred, and analysed **tens of billions** of simulated events
- As part of the high-luminosity LHC physics program (HL-LHC), the upgraded experiments expect to collect **150 times more data** than in Run 1
- More than **50%** of WLCG power used for simulations
- **GeantV**: path towards a faster toolkit **2-5 x Geant4**

Estimated ~10x CPU needs for the HL-LHC era

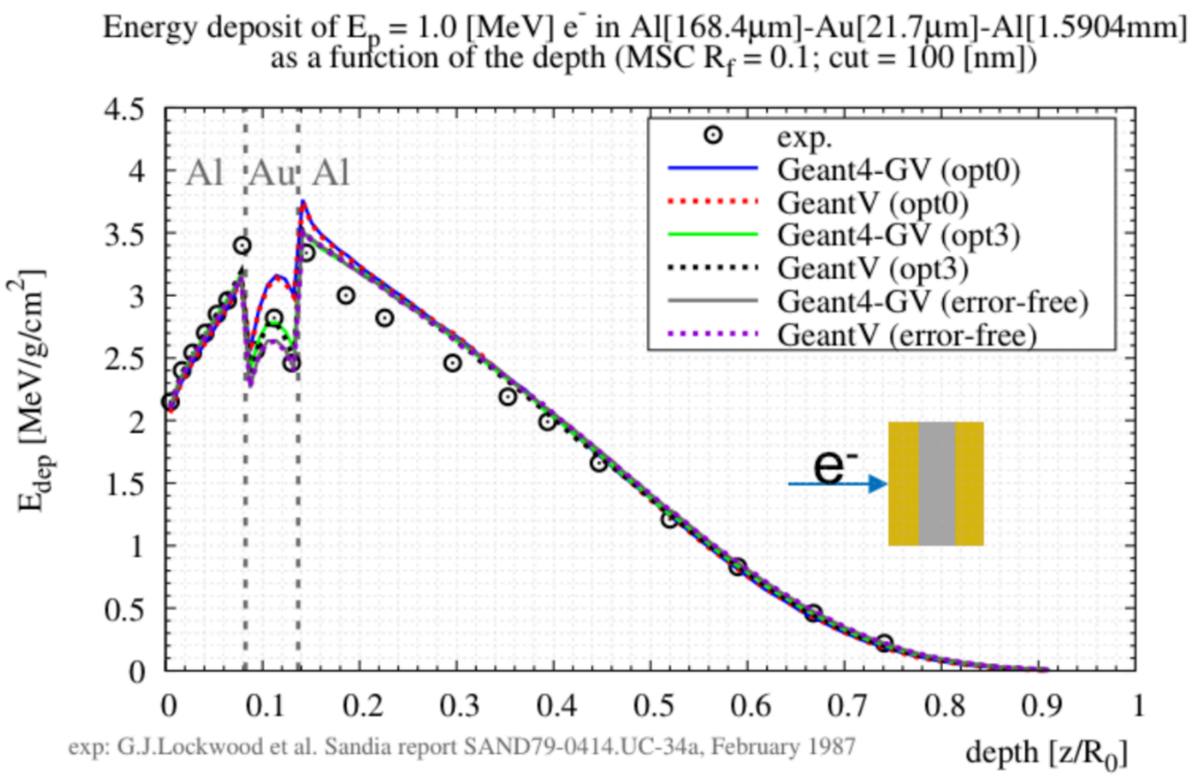



**CMS and Atlas estimated CPU needs for HL-LHC (source: CWP)**


marilena.bandieramonte@cern.ch

# EM PHYSICS MODELS VALIDATION

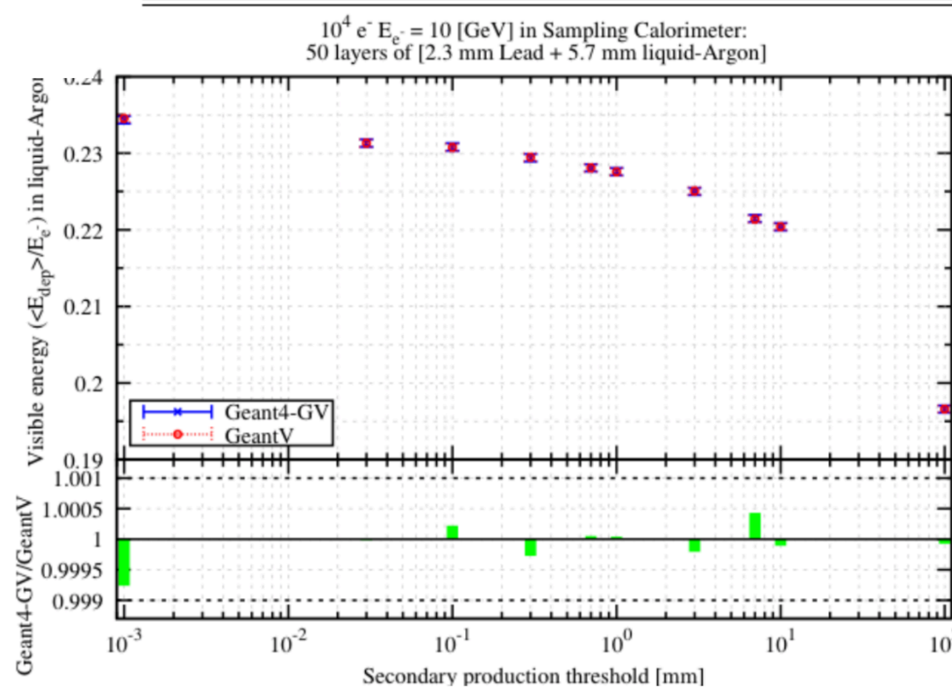



Multi-layered target

Scalar EM models revisited in a vectorization friendly way (e.g. vectorizable sampling) and validated against Geant4 version.

$10^5$ 1 [GeV] e- in ATLAS bar. simpl. cal. : 50 layers of [2.3 mm Pb + 5.7 mm lAr]; p.cut = 0.7 [mm]

| | $e^-/e^+$: ionisation, bremsstrahlung, msc; $\gamma$: Compton, conversion | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | GeantV | | | | Geant4 | | | |
| material | $E_d$[GeV] | rms [MeV ] | tr.l. [m] | rms [cm] | $E_d$[GeV] | rms [MeV ] | tr.l. [m] | rms [cm] |
| Pb | 0.69450 | 15.198 | 51.015 | 1.189 | 0.69448 | 15.234 | 51.016 | 1.192 |
| lAr | 0.22792 | 14.675 | 106.11 | 7.592 | 0.22796 | 14.656 | 106.13 | 7.582 |




Mean number of :

| | | |
|---|---|---|
| gamma | 405.87 | 406.15 |
| electron | 9411.49 | 9419.44 |
| positron | 53.77 | 53.71 |
| charged steps | 11470 | 11476 |
| neutral steps | 49177 | 49222 |

*credit: M. Novak*

Work in progress on vectorization of all the EM physics - expected to be included in the beta release!

ATLAS simplified sampling calorimeter


marilena.bandieramonte@cern.ch

# MAXIMUM SPEEDUP ACHIEVABLE

- Depends on the **vector width** but..
- Generally **is less than the vector register width**
  - some operations are **slower** for vector registers

Reciprocal Throughput* for Division DP (SandyBridge)

| | |
|---|---|
| Scalar | 10-20 cycles |
| Vector | 20-44 cycles |

    - Maximum speedup for division will be ~ 2 for this CPU
  - **Overhead** payed to gather data into SIMD vectors
- Another important factor is the **number of execution units** for particular instructions = number of instructions that can be executed simultaneously.

*The average number of core clock cycles per instruction for a series of independent instructions of the same kind in the same thread.

**


marilena.bandieramonte@cern.ch

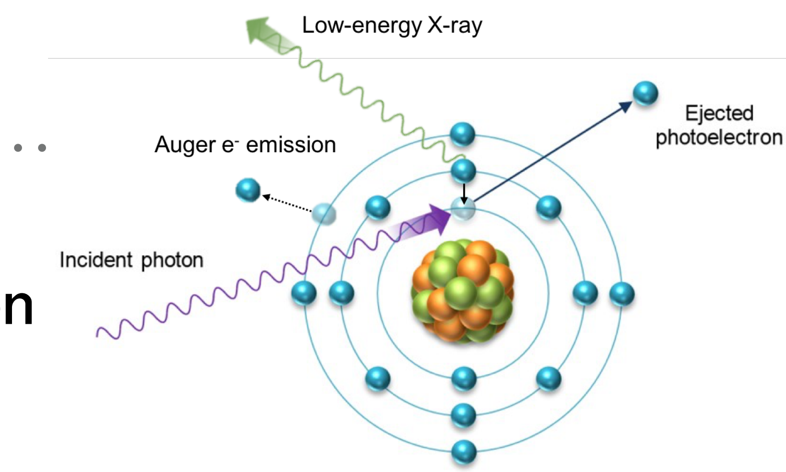
# RESULTS: MODEL LEVEL TEST BENCHMARKS

| Model | Haswell (avx) | |
|---|---|---|
| | Scalar Time [ms] | SpeedUp |
| Klein-Nishina alias | 56.4 | 2.2 |
| Klein-Nishina rej | 48.37 | 2.21 |
| Moller-Bhabba alias | 51.32 | 1.85 |
| Moller-Bhabba rej | 50.21 | 1.62 |
| Seltzer-Berger brems alias | 73.19 | 1.98 |
| Seltzer-Berger brems rej | 106.63 | 1.64 |
| Relativistic brems alias | 76.96 | 2 |
| Relativistic brems rej | 330.57 | 2 |
| Bethe-Heitler pair alias | 86.53 | 1.82 |
| Bethe-Heitler pair rej | 62.98 | 1.39 |
| Relativistic pair alias | 91.66 | 1.37 |
| Relativistic pair rej | 83.42 | 1.83 |
| Positron2Gamma alias | 60.78 | 2.23 |
| Positron2Gamma rej | 41.34 | 1.91 |
| Sauter-Gavrila alias | 66.4 | 1.45 |
| Sauter-Gavrila rej | 108.89 | 1.2 |

Test with Lead, #baskets: 256

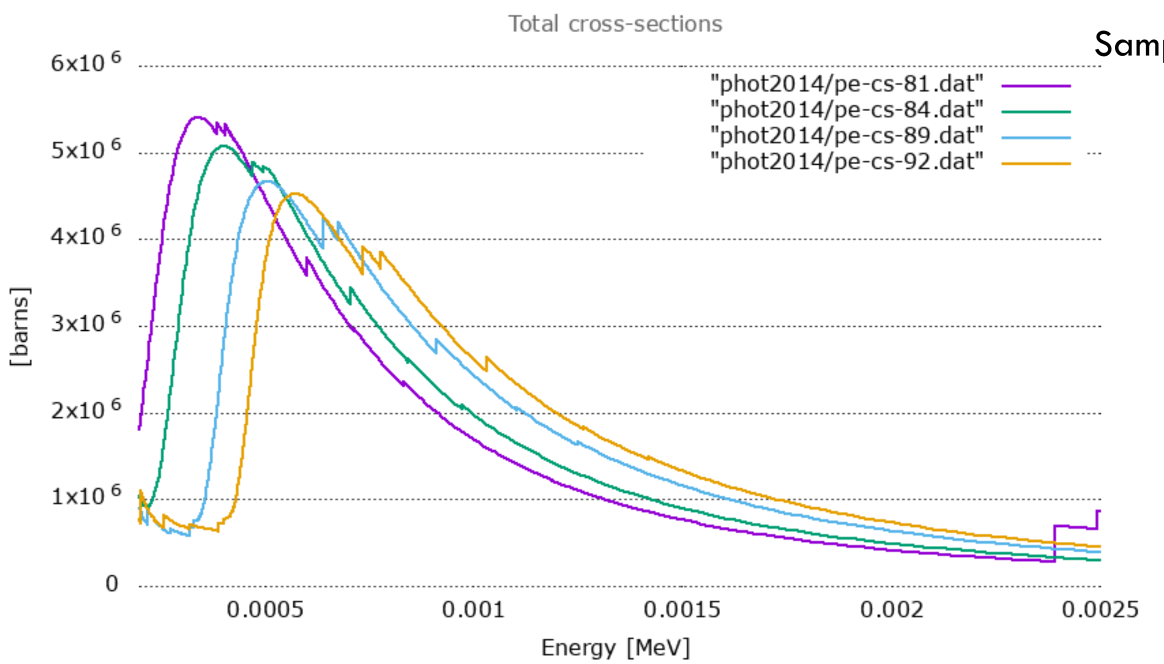

marilena.bandieramonte@cern.ch

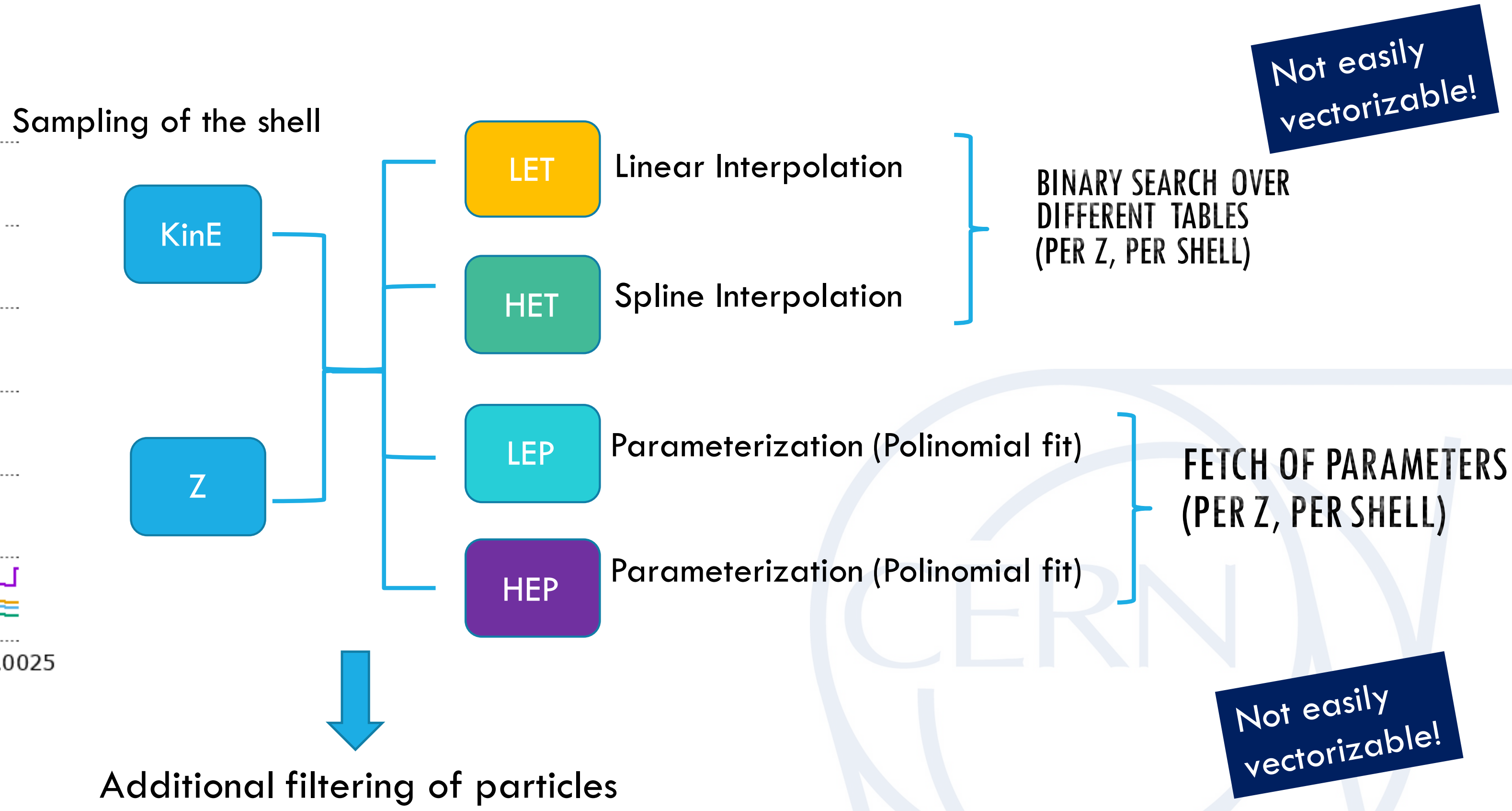
# EXAMPLE: PE EFFECT




## - Photoelectric effect total cross-section is not an easy function

- Fit in two different energy ranges, but not below k-shell binding energy
  - Tabulated cross-sections left for low energies
- For the final state sampling one need to sample
  - the angle: described by the SauterGavrila differential cross-section
  - the subshell: This is going through a binary search algorithm (not vectorizable) + linear or spline interpolation

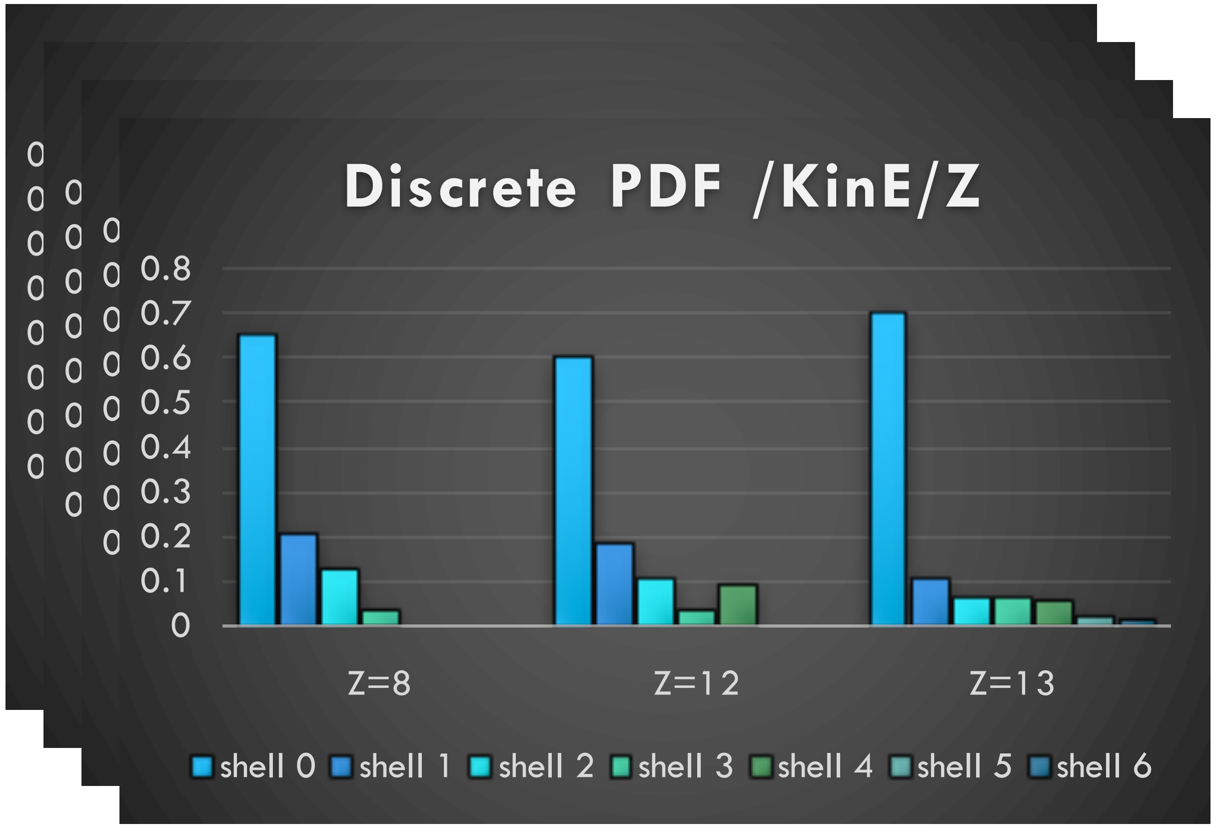


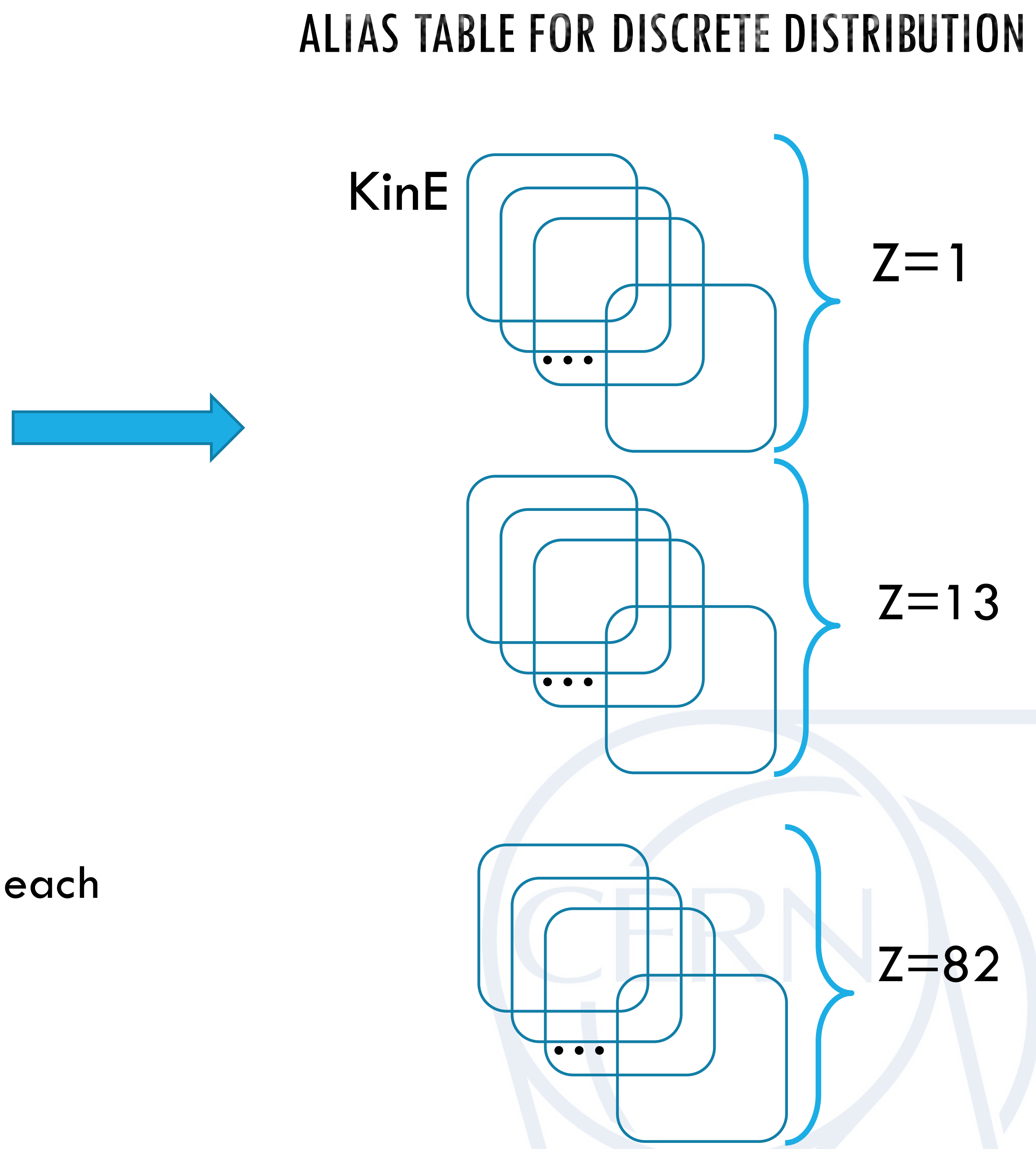





marilena.bandieramonte@cern.ch

# VECTORIZATION WITH DISCRETE ALIAS TABLES




- We generated a denser ss-cs dataset
  - to build equally spaced (in energy) discrete PDFs for each element (linearly interpolated)
  - From them we can build Alias Table
    - PRO: sampling of shells with only one case
    - CONS: Gathering operations


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING


marilena.bandieramonte@cern.ch

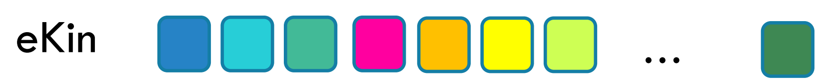
# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

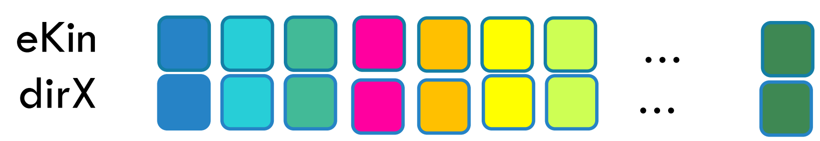





marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1. Prepare values that are needed for sampling, in form of arrays

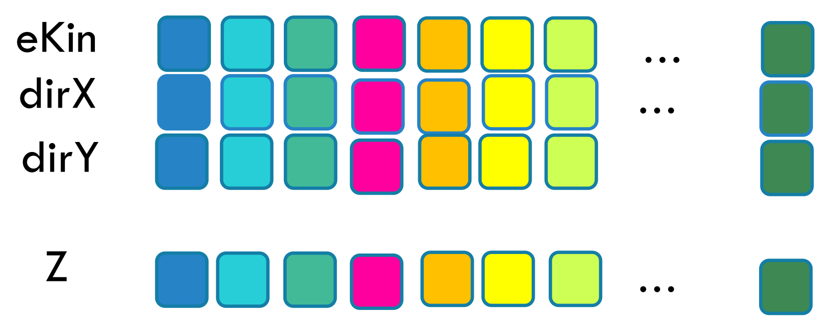




marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1. Prepare values that are needed for sampling, in form of arrays

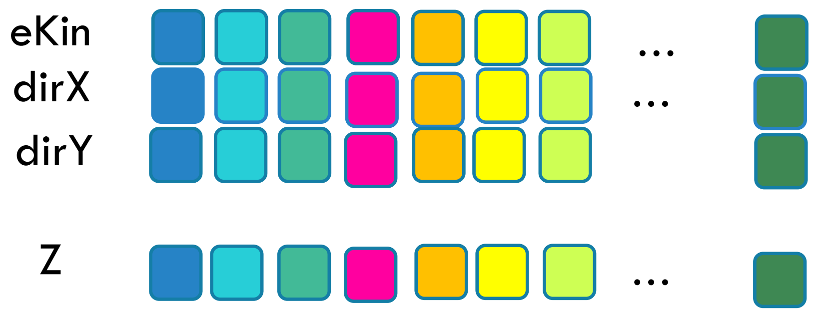




marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

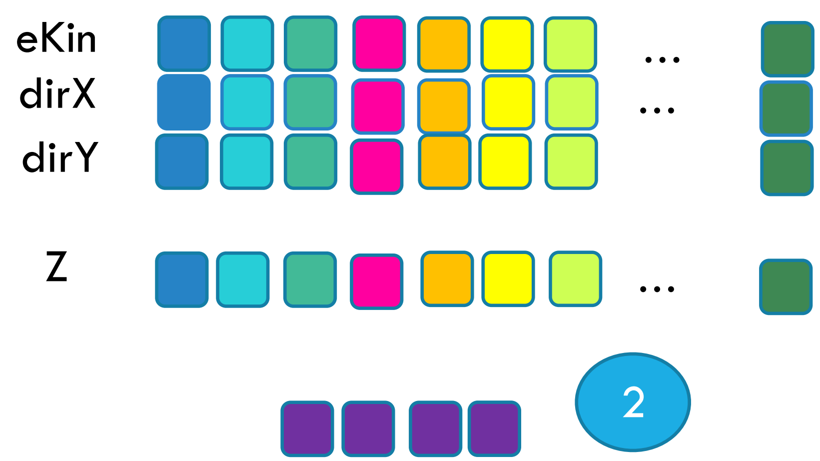



2 Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

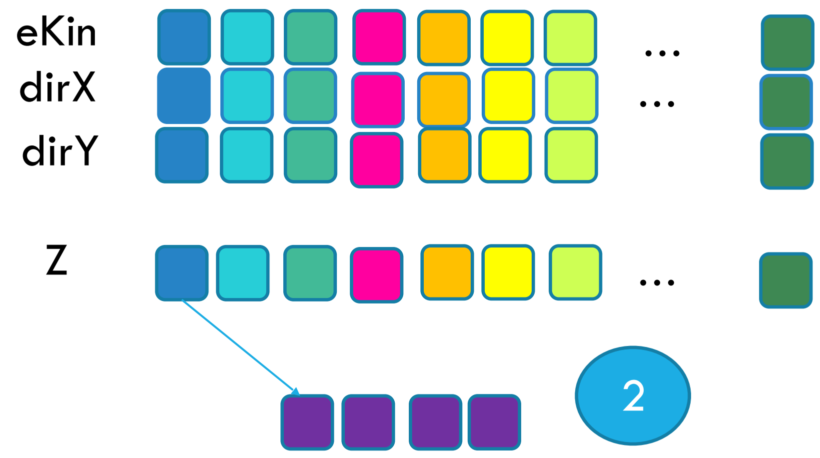



Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

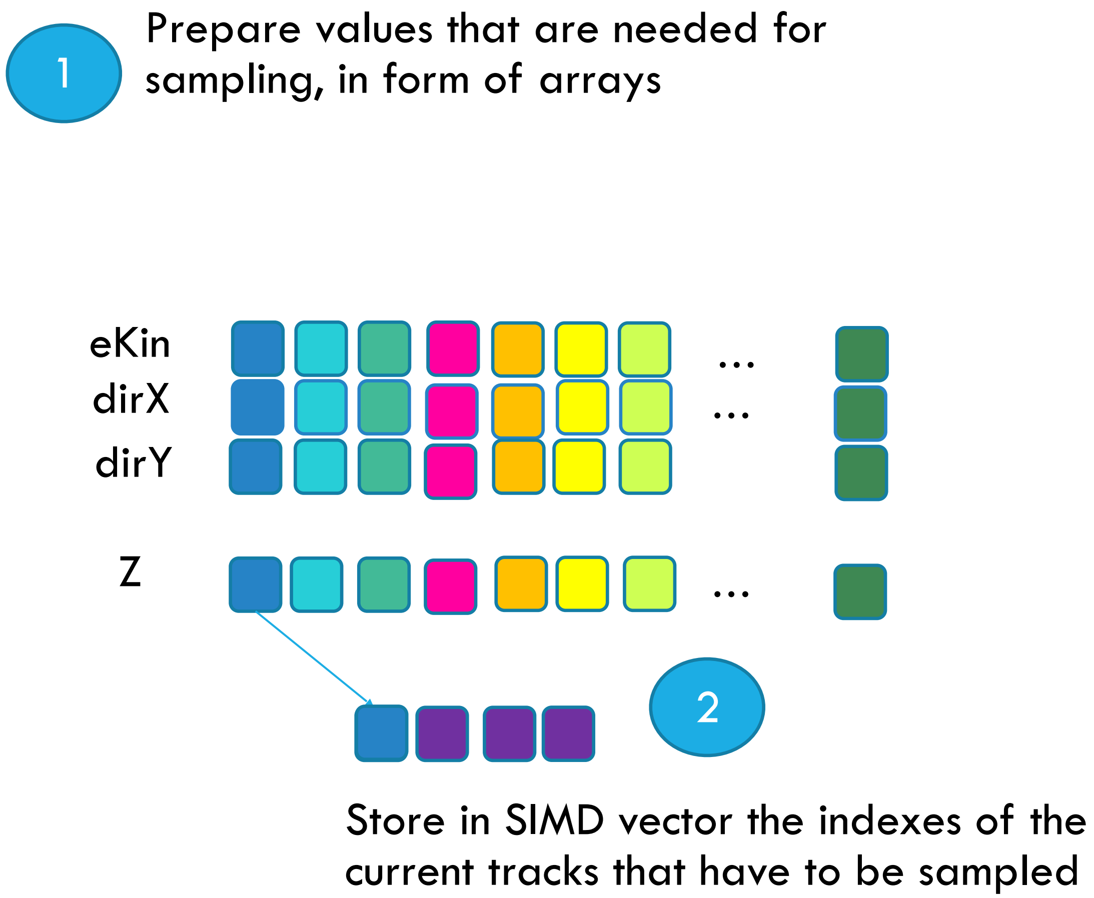
1 Prepare values that are needed for sampling, in form of arrays




Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

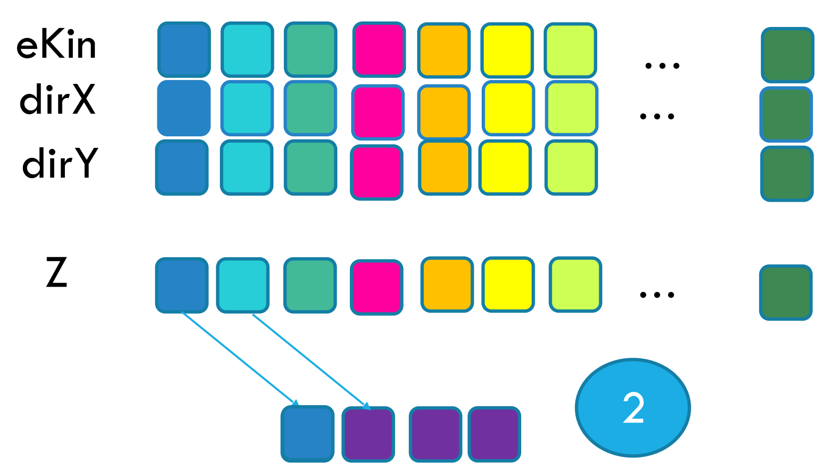




marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

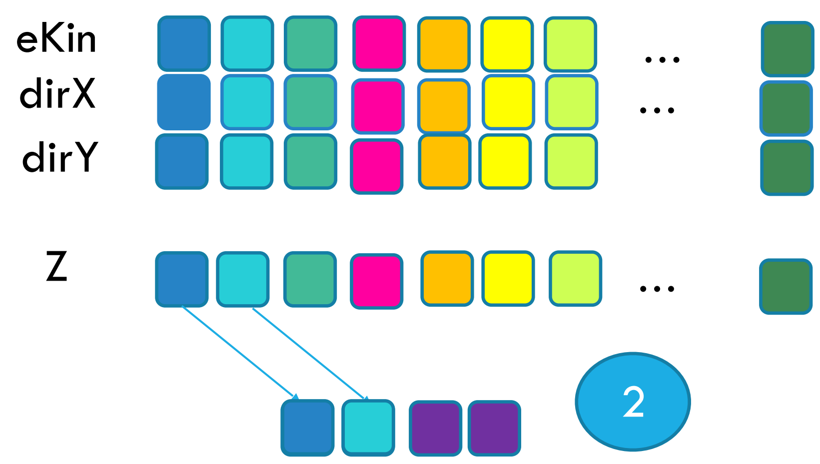



Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

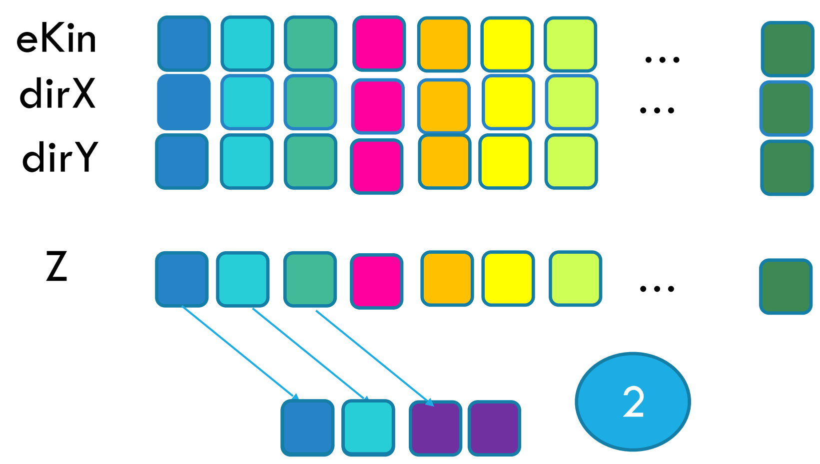



Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

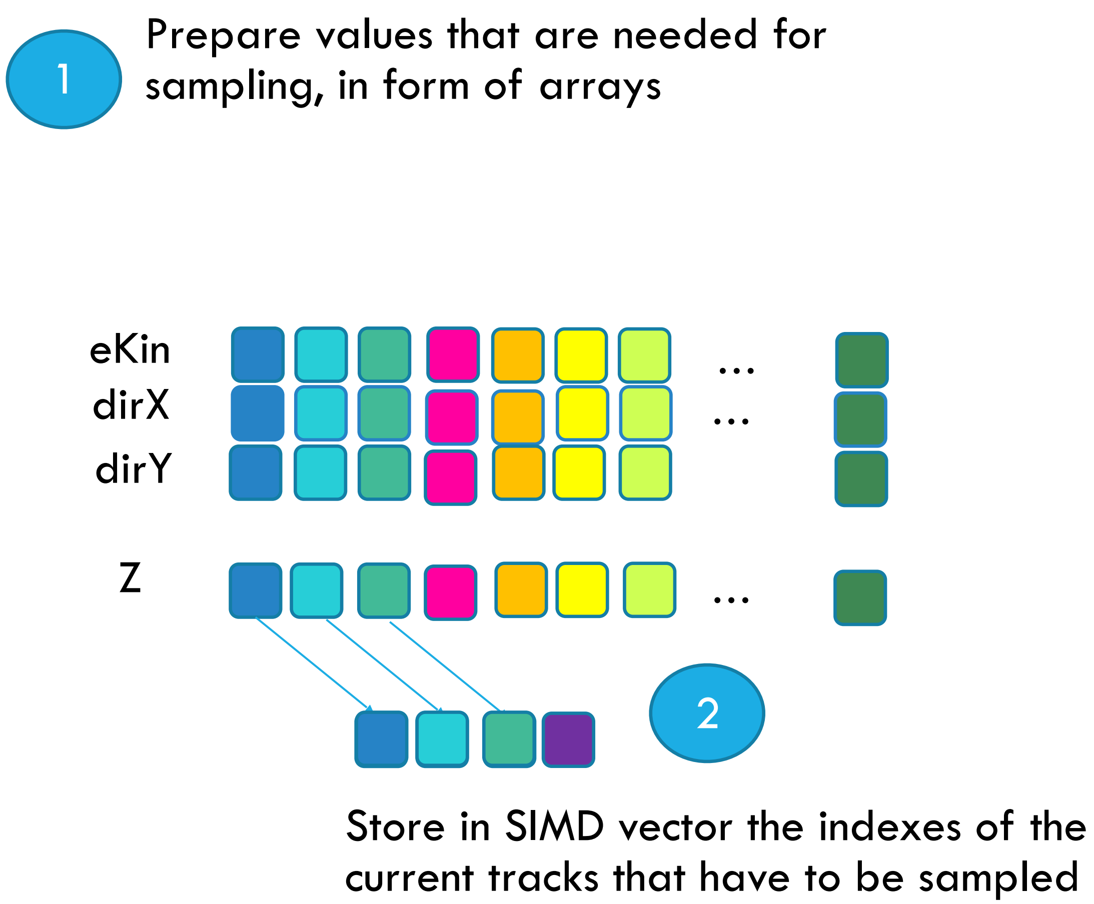
1 Prepare values that are needed for sampling, in form of arrays




Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

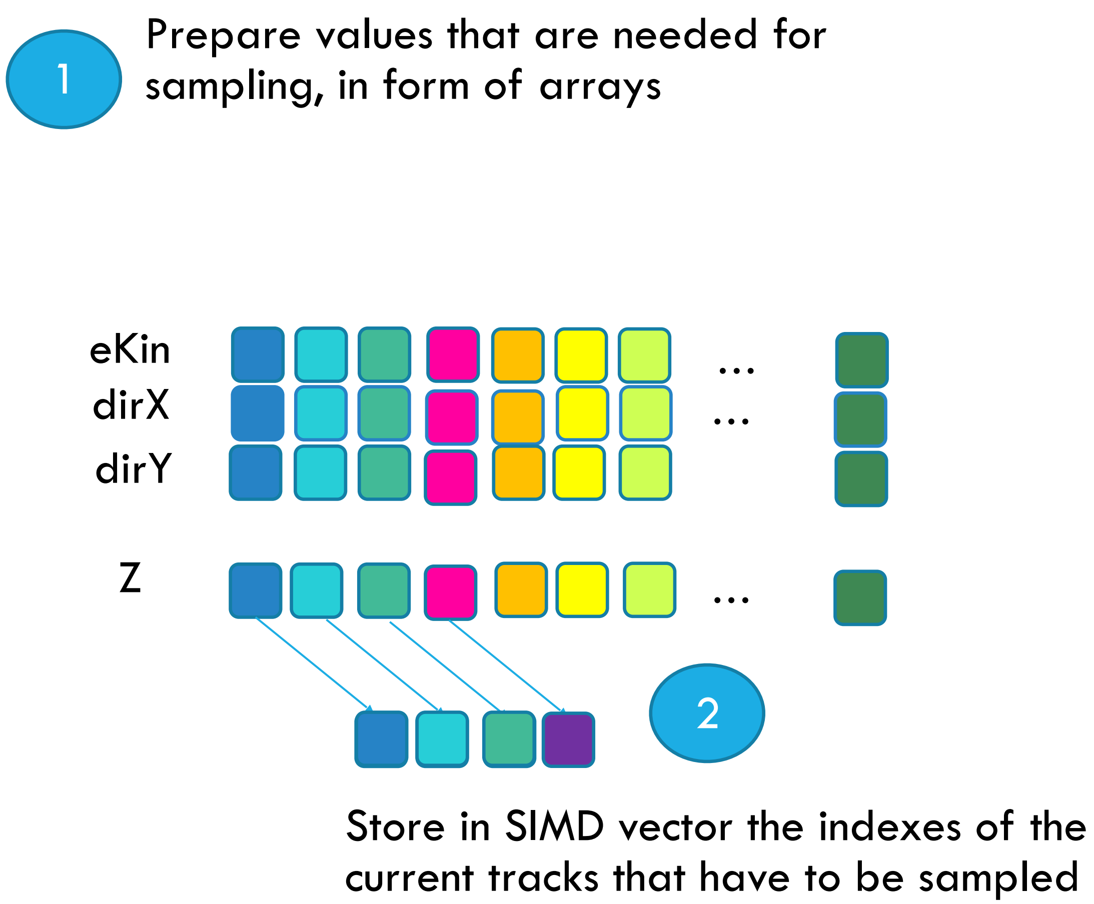
# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

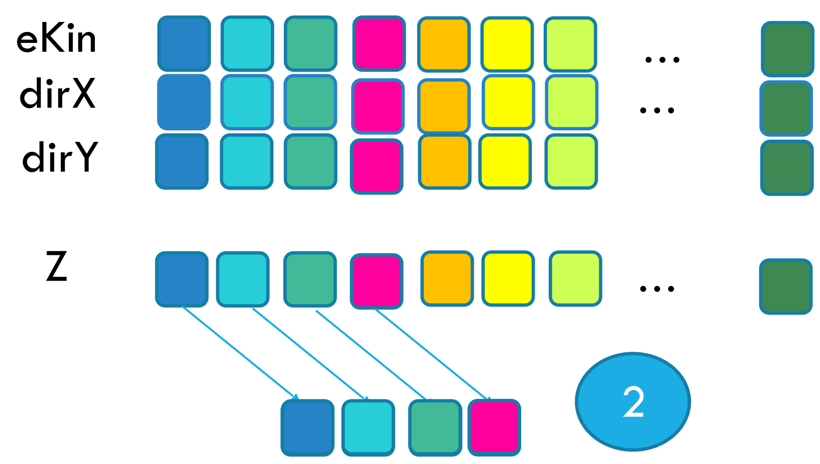




marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

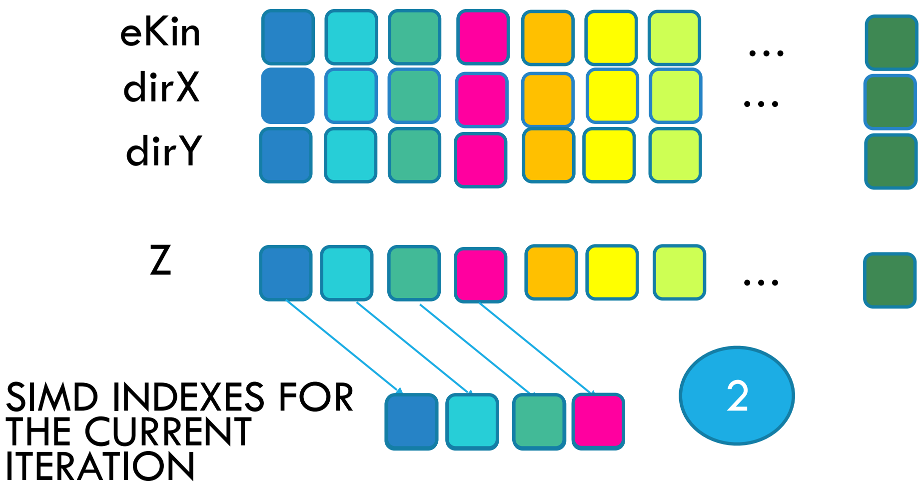



Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

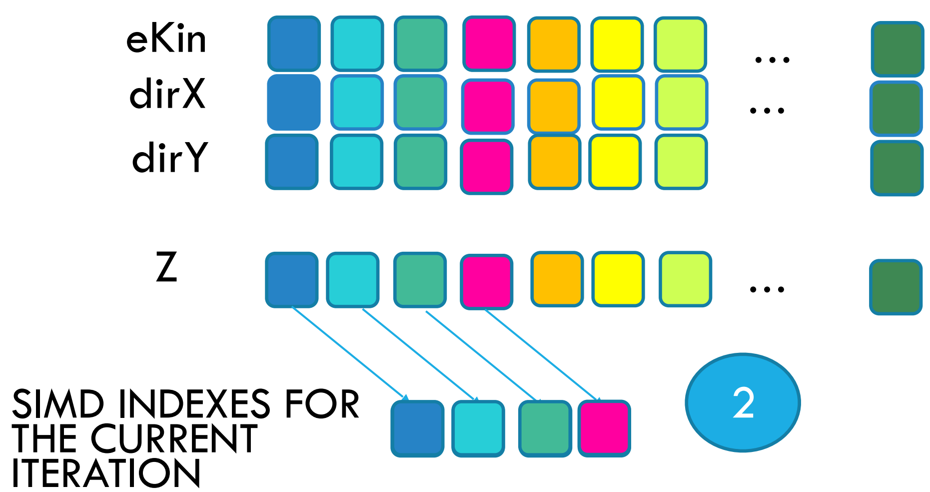



Store in SIMD vector the indexes of the current tracks that have to be sampled


marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays



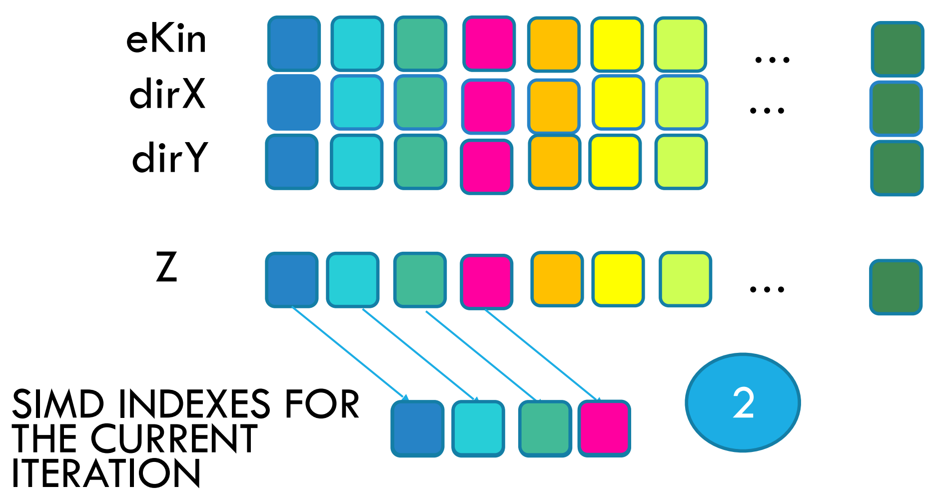

Store in SIMD vector the indexes of the current tracks that have to be sampled

3 Gather values from array using this indexes


marilena.bandieramonte@cern.ch

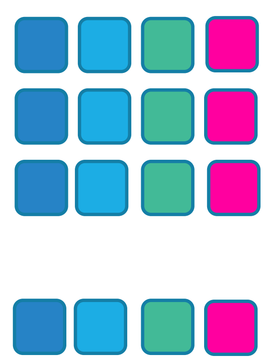
# VECTORIZATION OF REJECTION SAMPLING

1. Prepare values that are needed for sampling, in form of arrays




2. Store in SIMD vector the indexes of the current tracks that have to be sampled

3. Gather values from array using this indexes

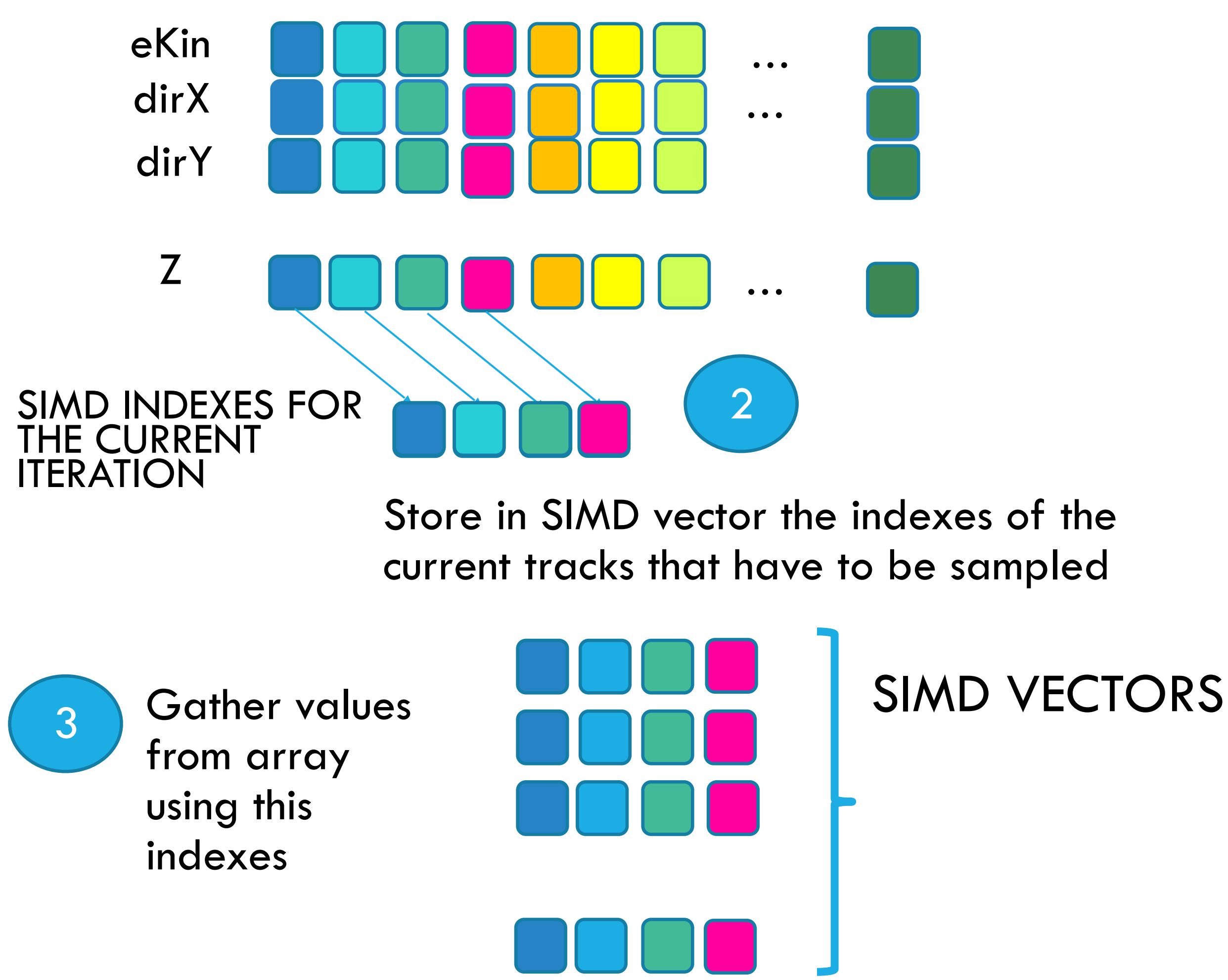




marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1. Prepare values that are needed for sampling, in form of arrays

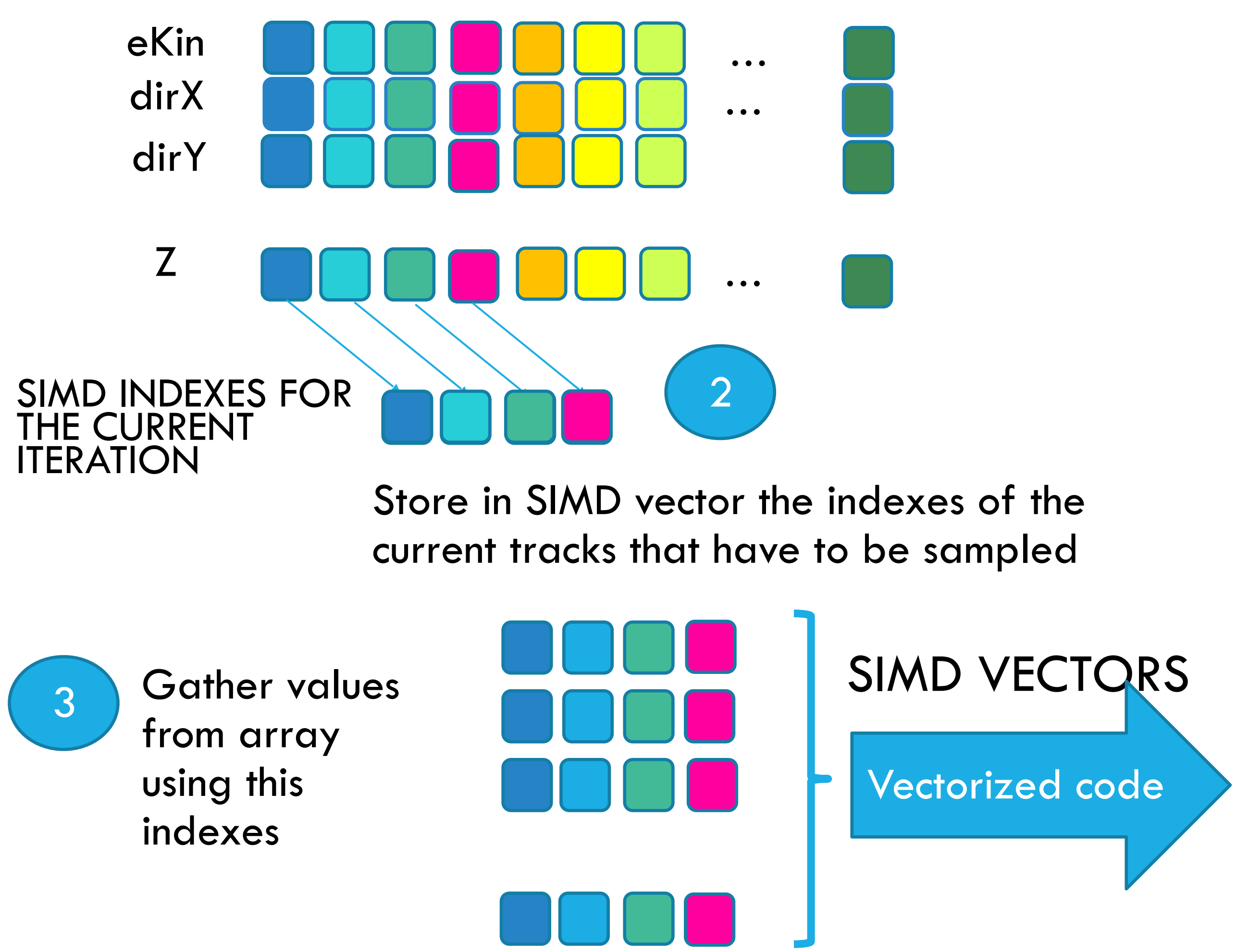




marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays

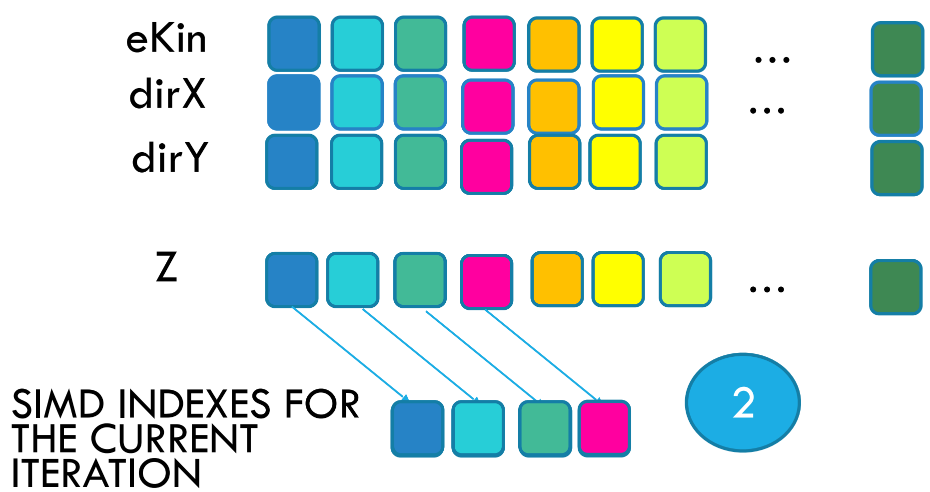




marilena.bandieramonte@cern.ch

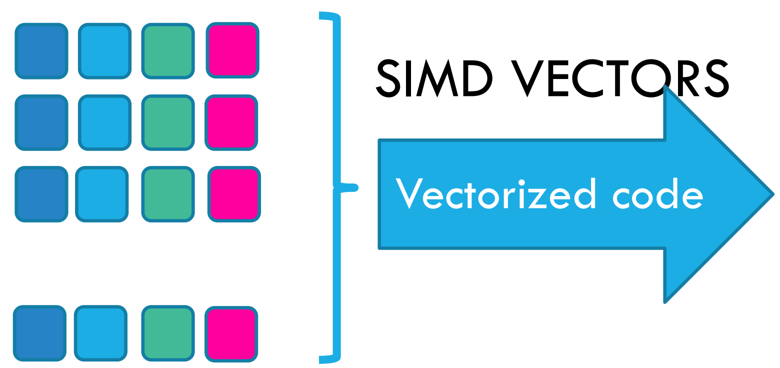
# VECTORIZATION OF REJECTION SAMPLING

1 Prepare values that are needed for sampling, in form of arrays




Store in SIMD vector the indexes of the current tracks that have to be sampled

3 Gather values from array using this indexes

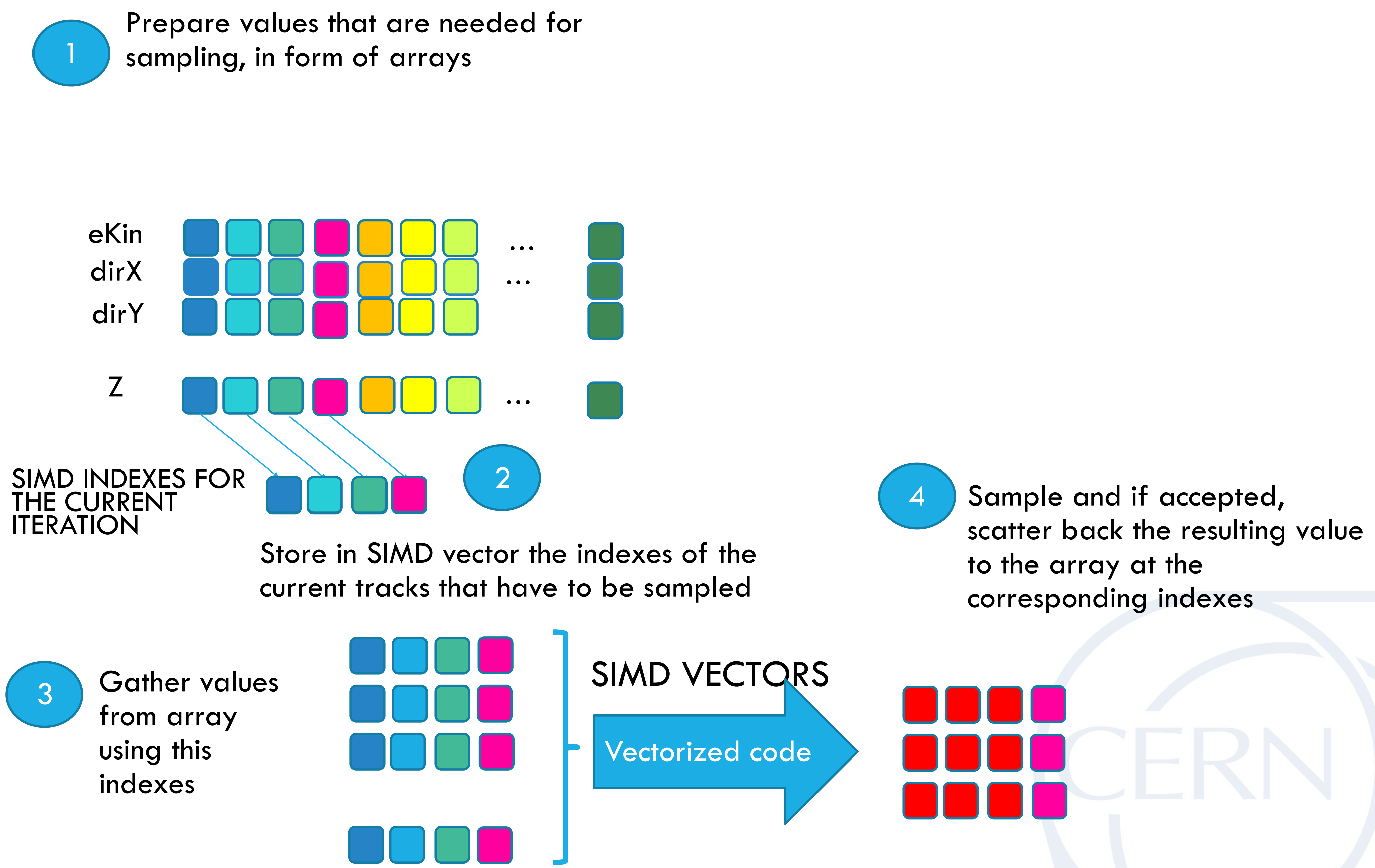



4 Sample and if accepted, scatter back the resulting value to the array at the corresponding indexes



marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

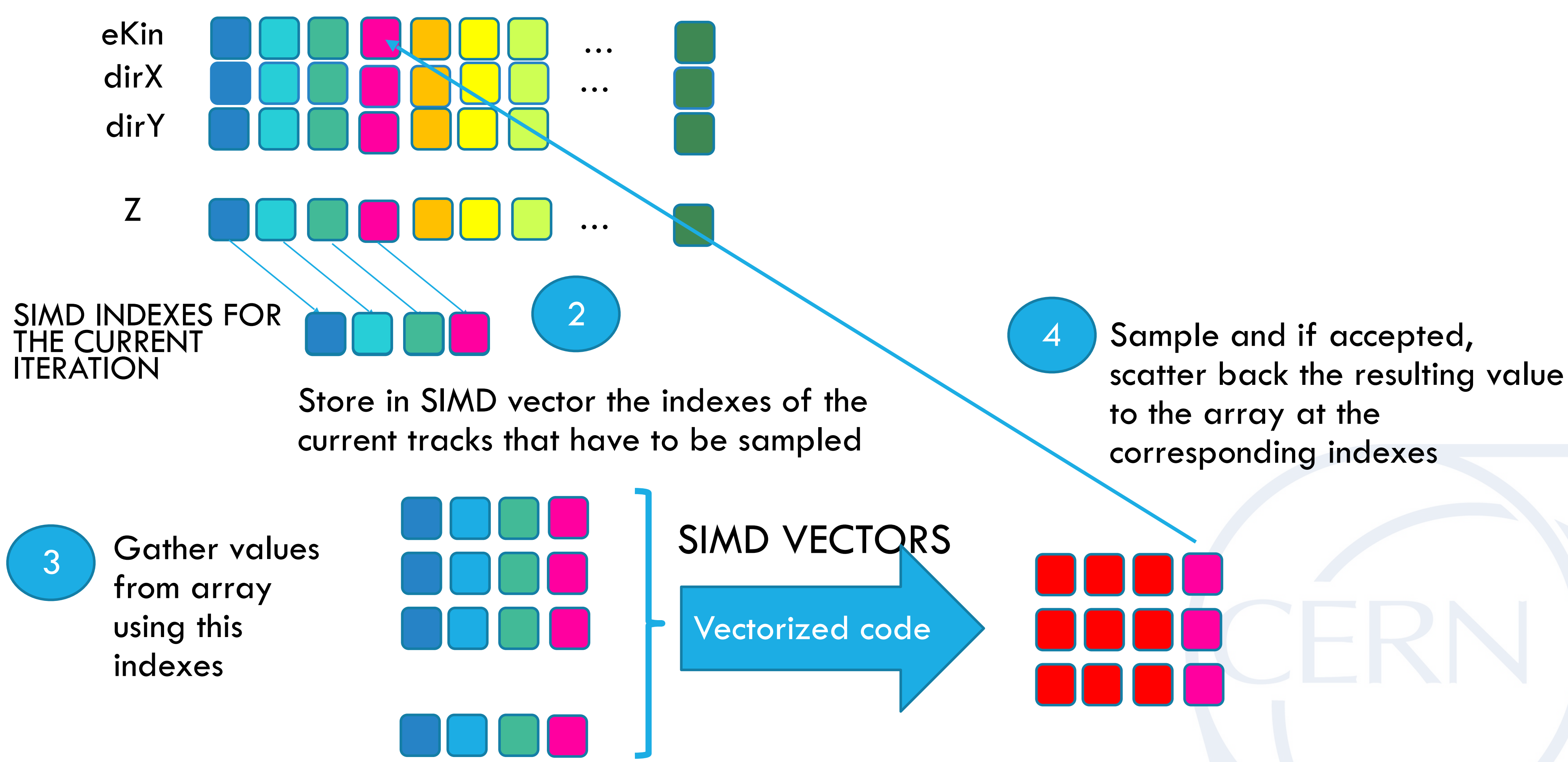





marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING

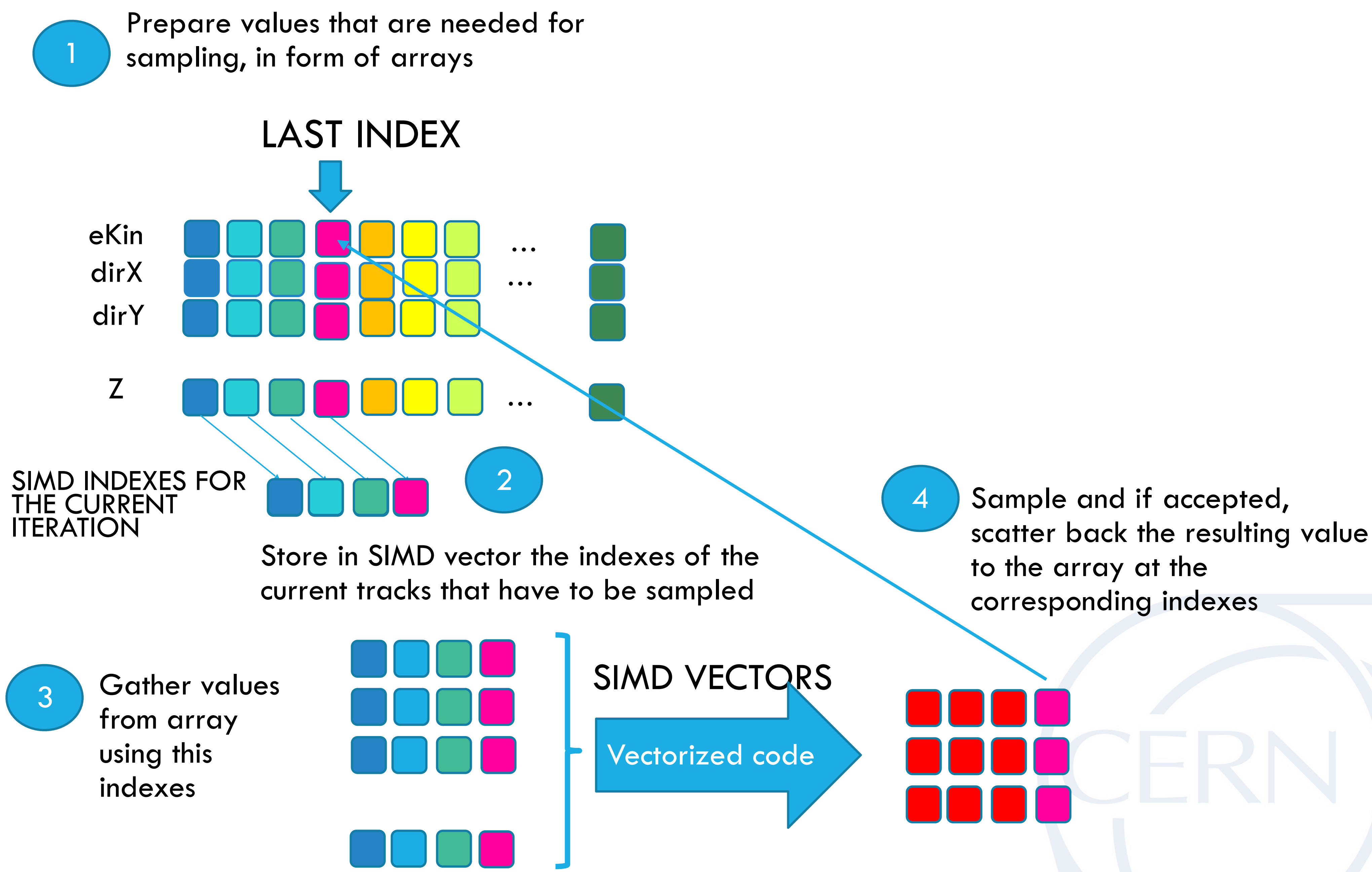
1 Prepare values that are needed for sampling, in form of arrays





marilena.bandieramonte@cern.ch

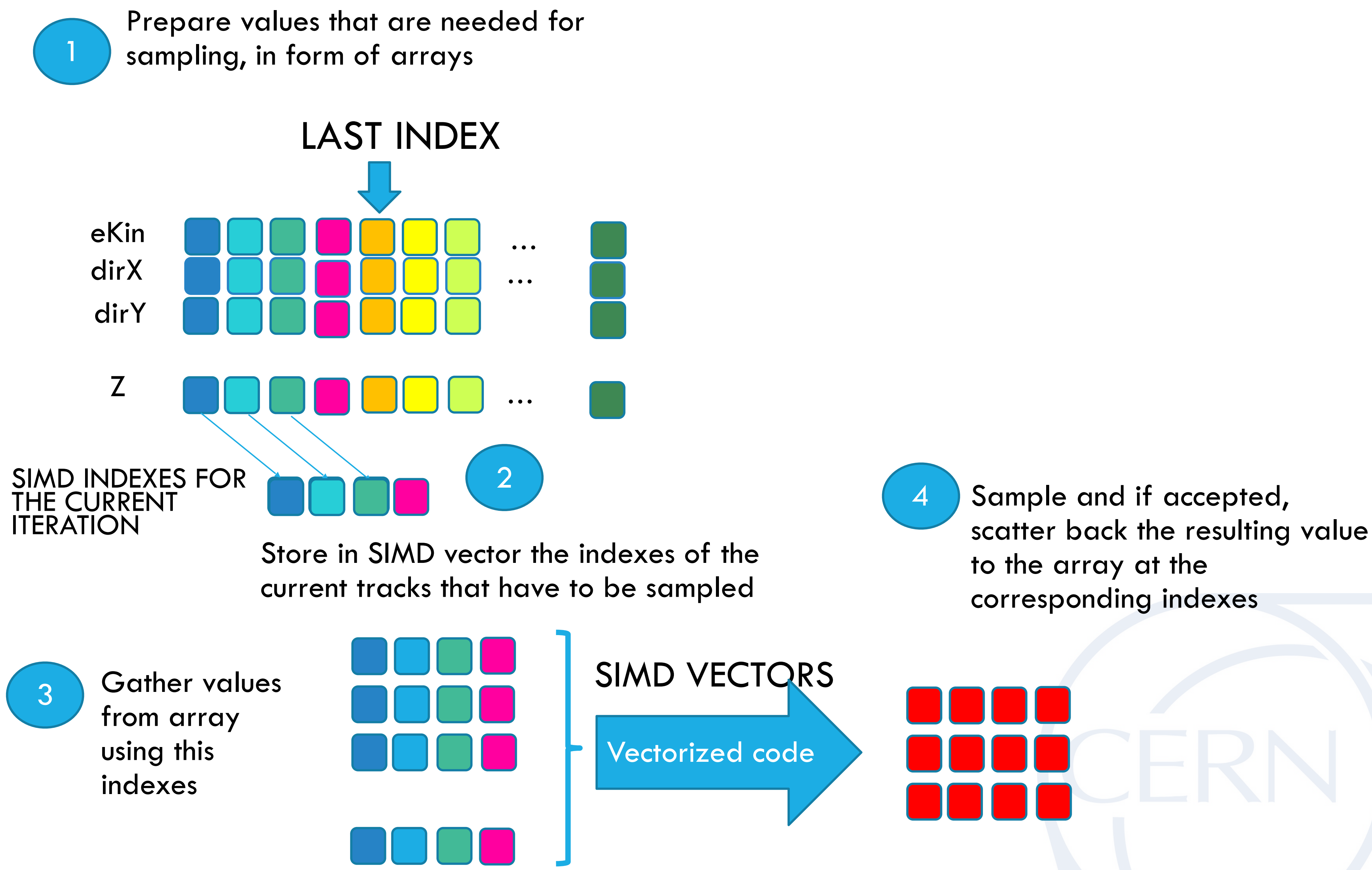
# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch

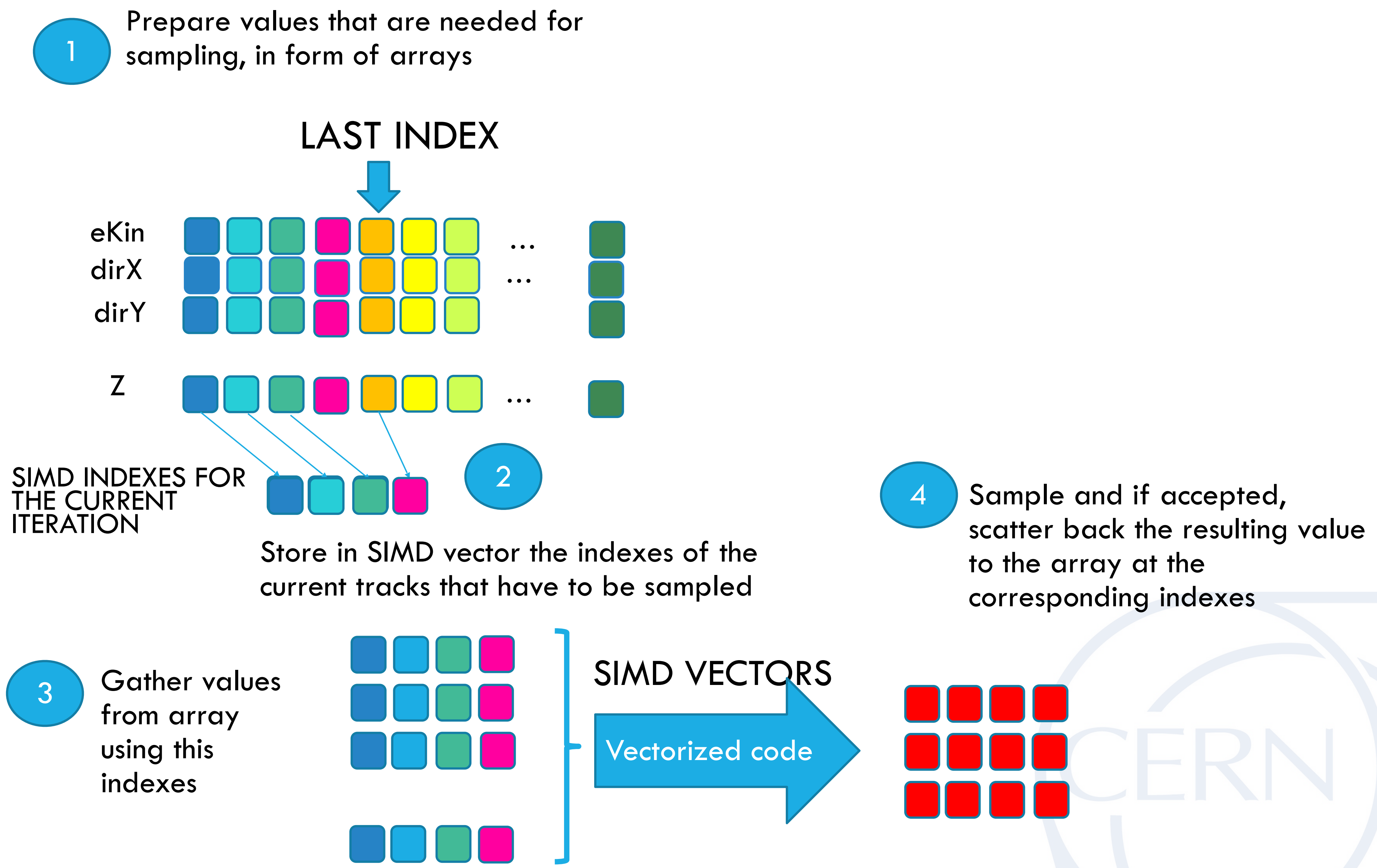
# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch

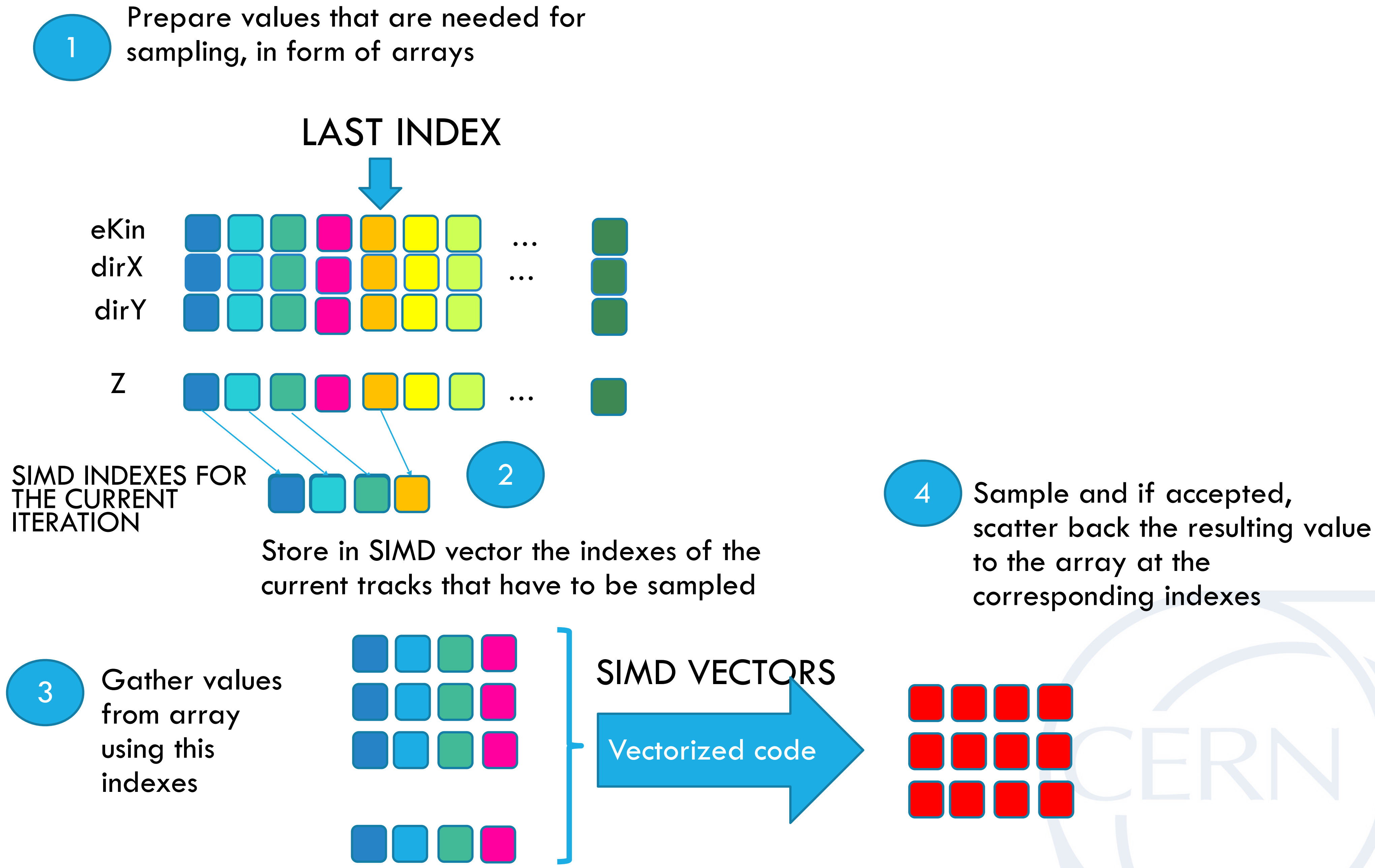
# VECTORIZATION OF REJECTION SAMPLING






marilena.bandieramonte@cern.ch

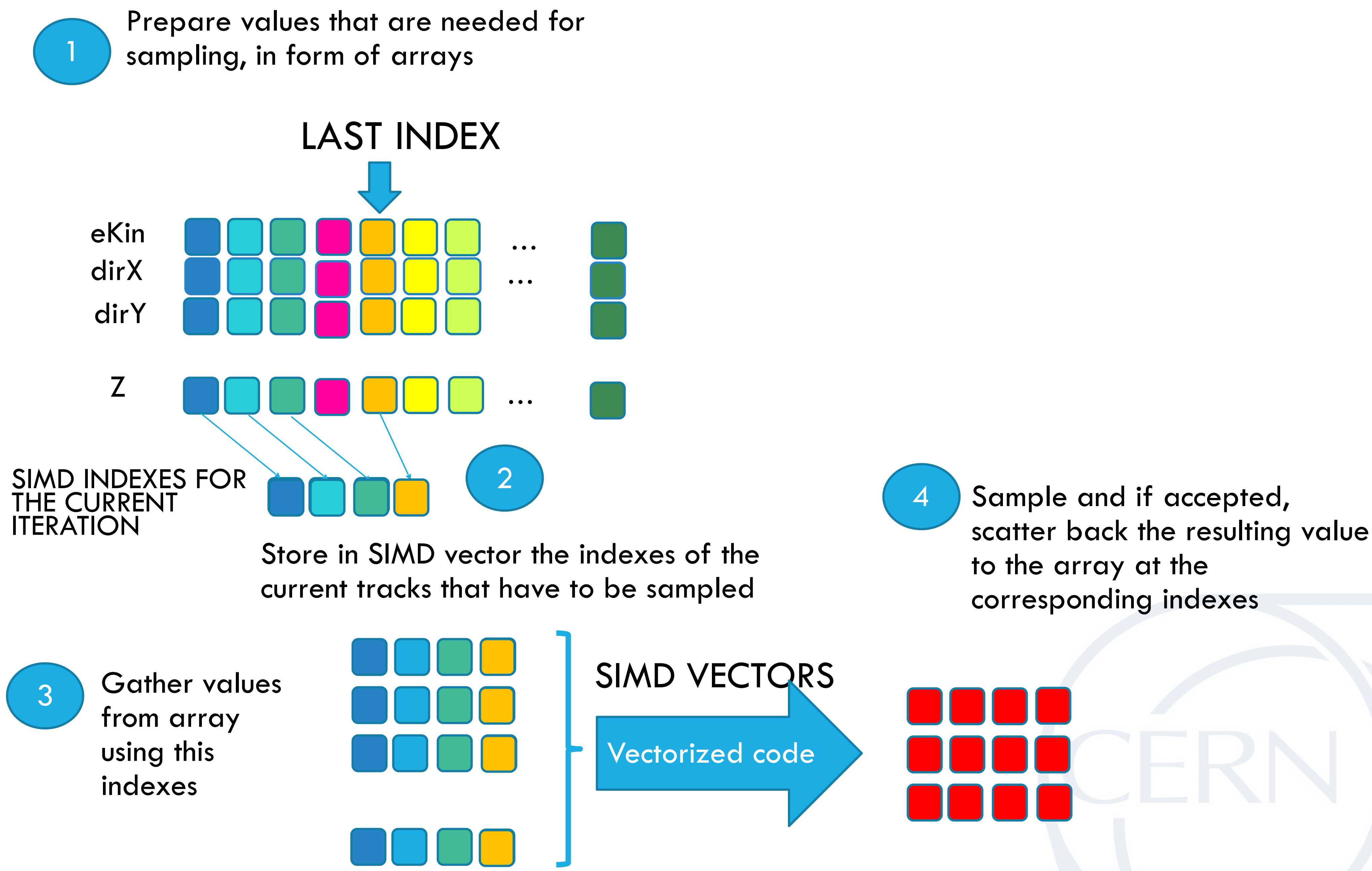
# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch

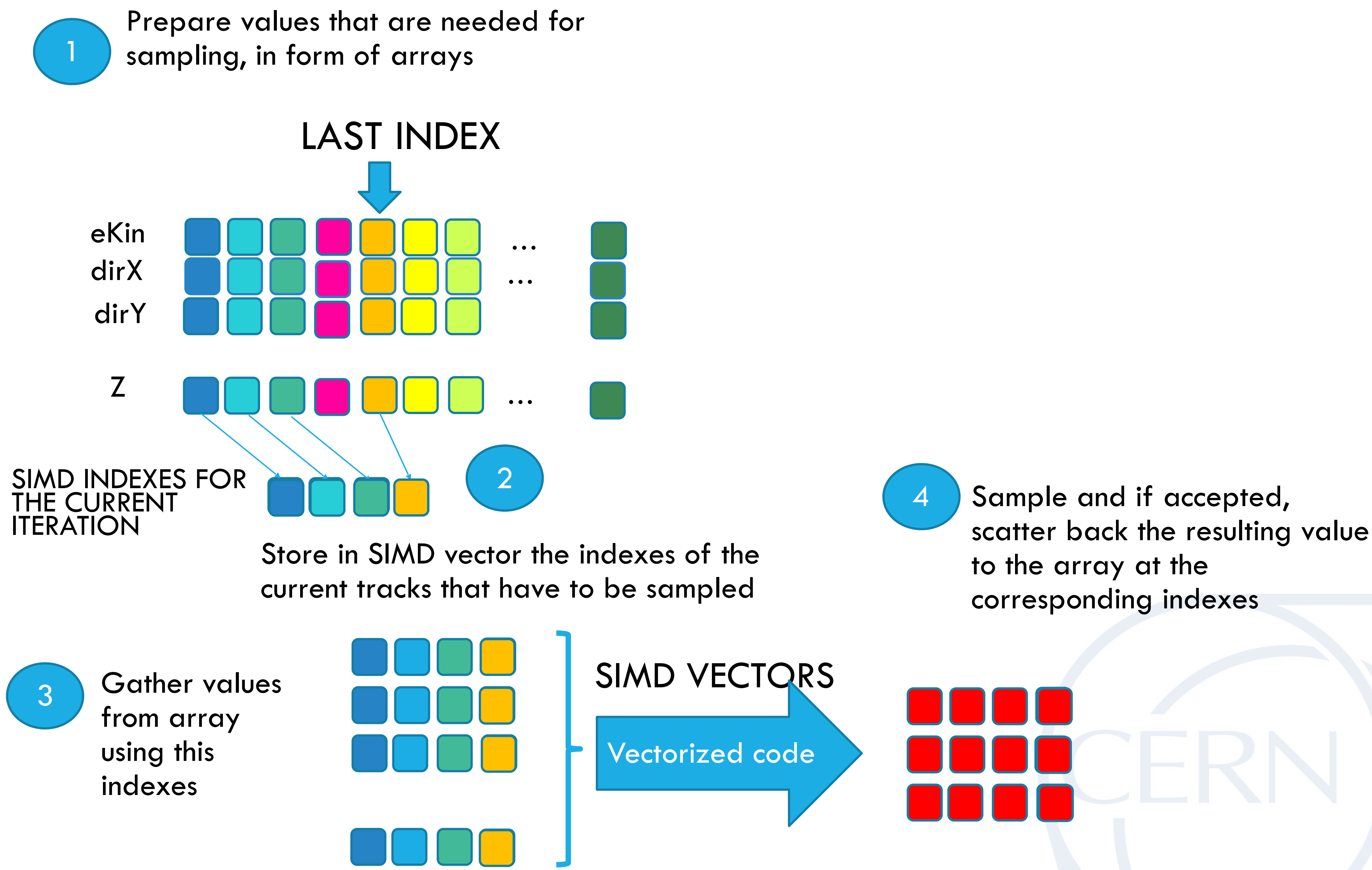
# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch

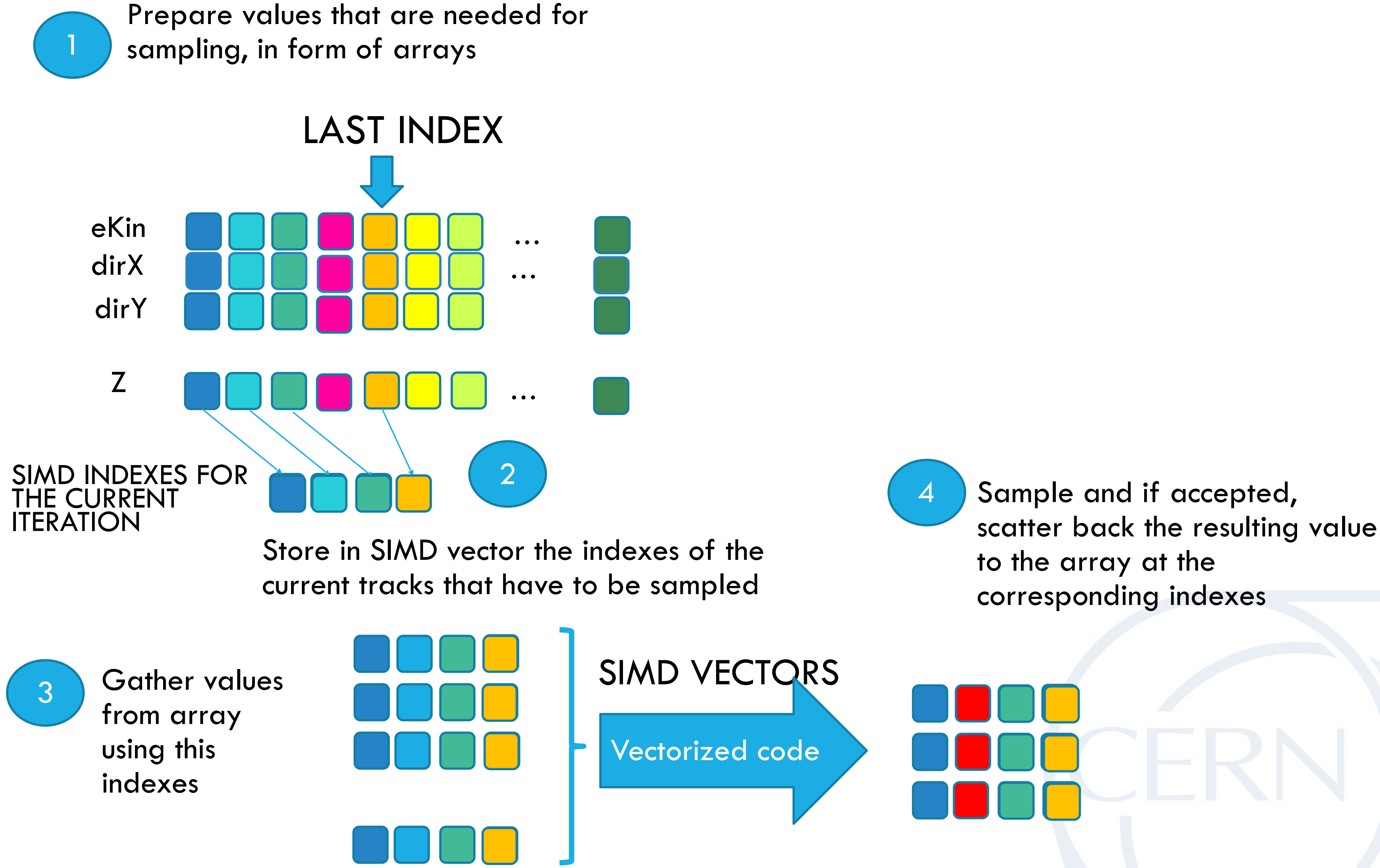
# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch

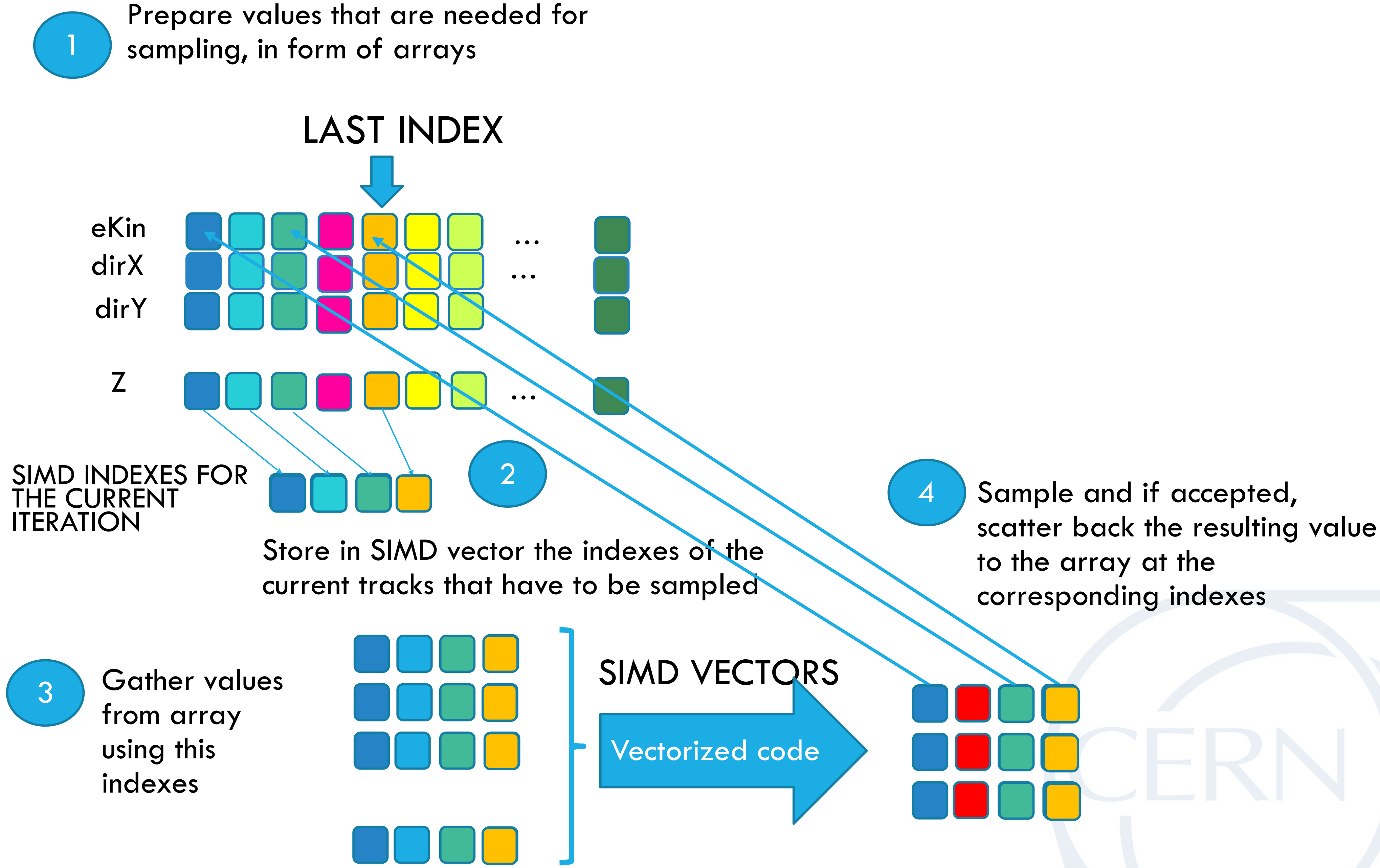
# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch

# VECTORIZATION OF REJECTION SAMPLING





marilena.bandieramonte@cern.ch