GeantV R&D: meeting the challenges of future HEP simulation software

Marilena Bandieramonte

(marilena.bandieramonte@cern.ch)

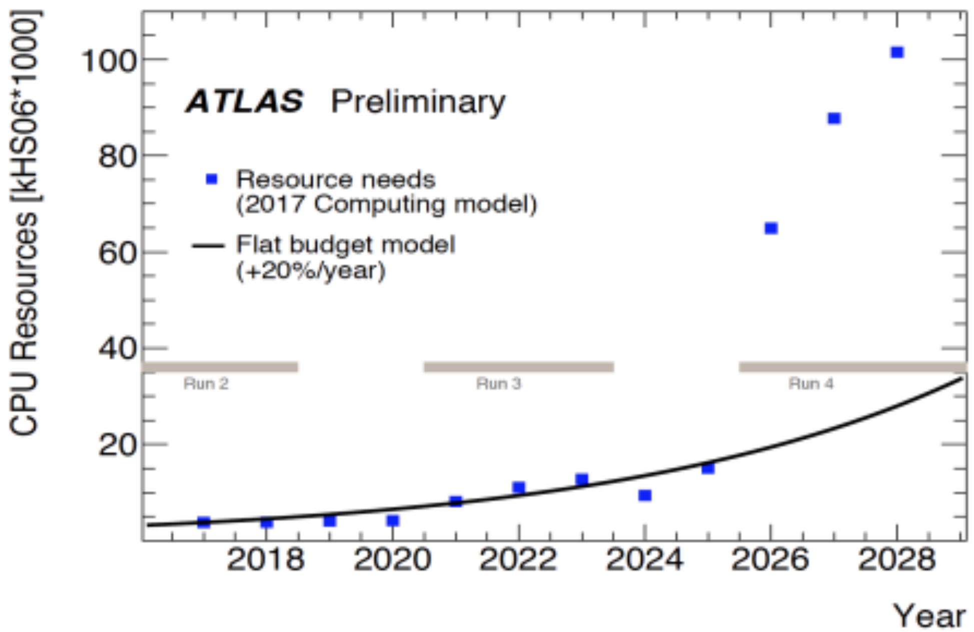
on behalf of the GeantV development team

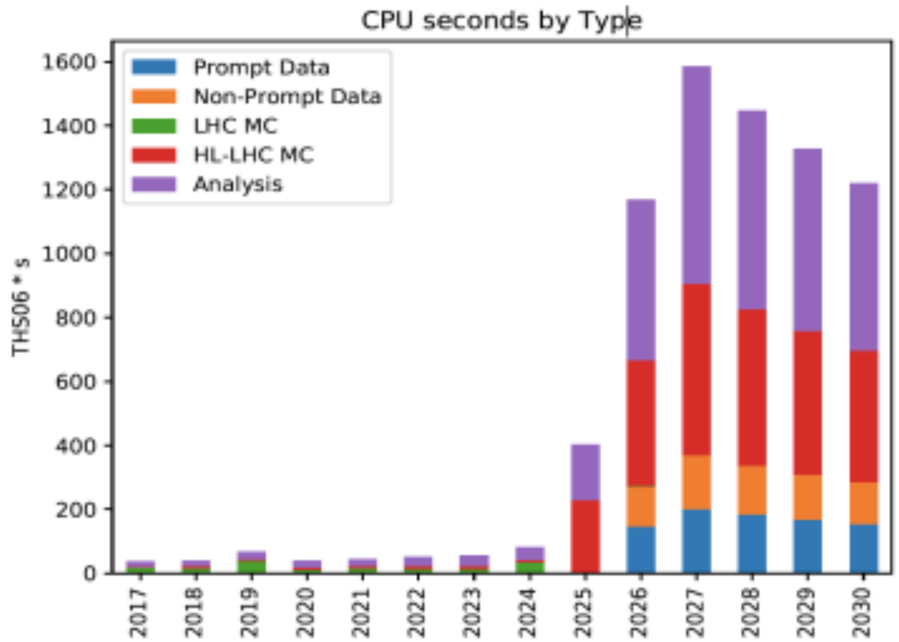
30th May 2018, Alghero

NEED FOR FASTER SIMULATION CODE FOR HEP COMMUNITY

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

- 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**



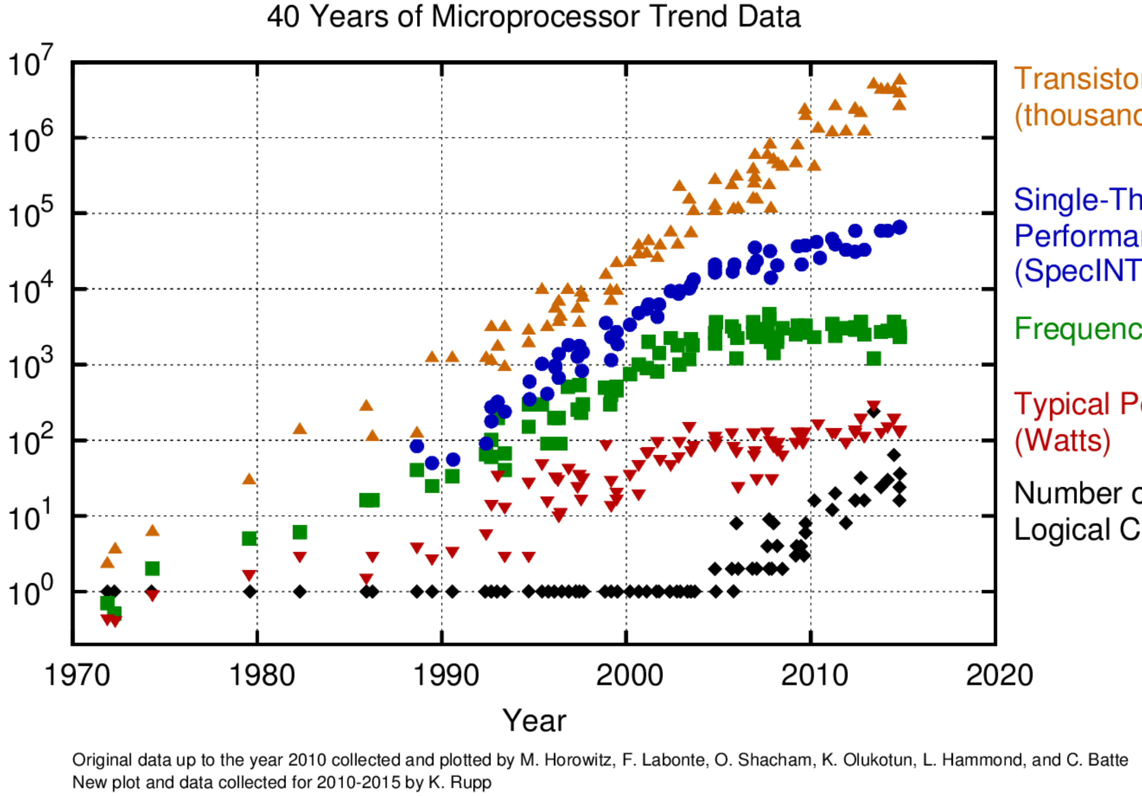


CMS and Atlas estimated CPU needs for HL-LHC (source: CWP)

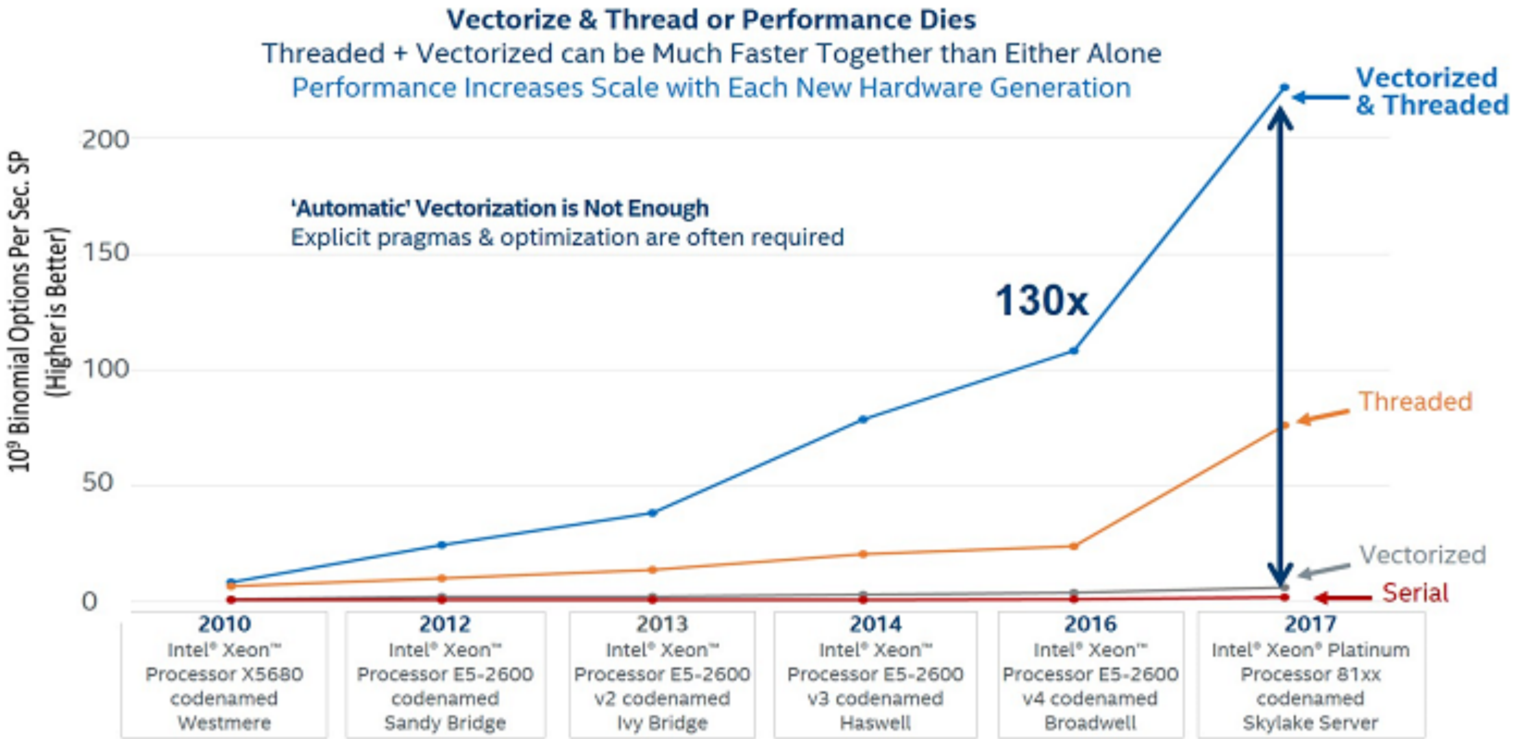
marilena.bandieramonte@cern.ch

SOFTWARE PERFORMANCE: MUCH TO GAIN

History of Intel chip introductions by clock speed and number of transistors



- Evolution trend of faster single-threaded CPU performance broken **more than 10 years ago.**
- Increase of **CPU cores** and more execution units to overcome stagnation in CPU Clock Speed



- Most applications need **significant redesign** in order to benefit from modern CPU architectures

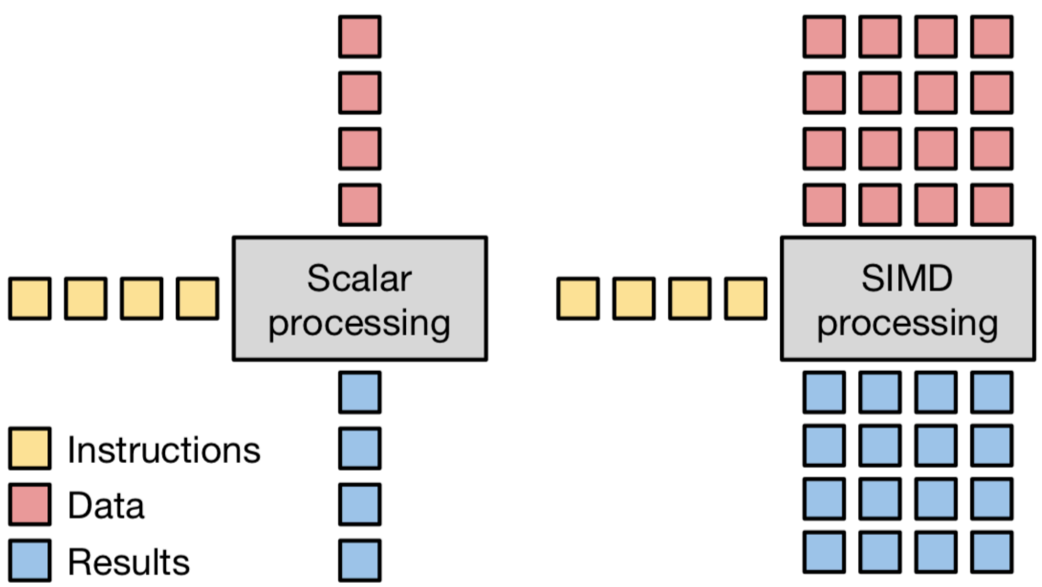
http://www.gotw.ca/publications/concurrency-ddj.htm

marilena.bandieramonte@cern.ch

VECTORIZATION: ENABLING SIMD INSTRUCTIONS

Instruction-level parallelism

SIMD → Single Instruction, Multiple Data



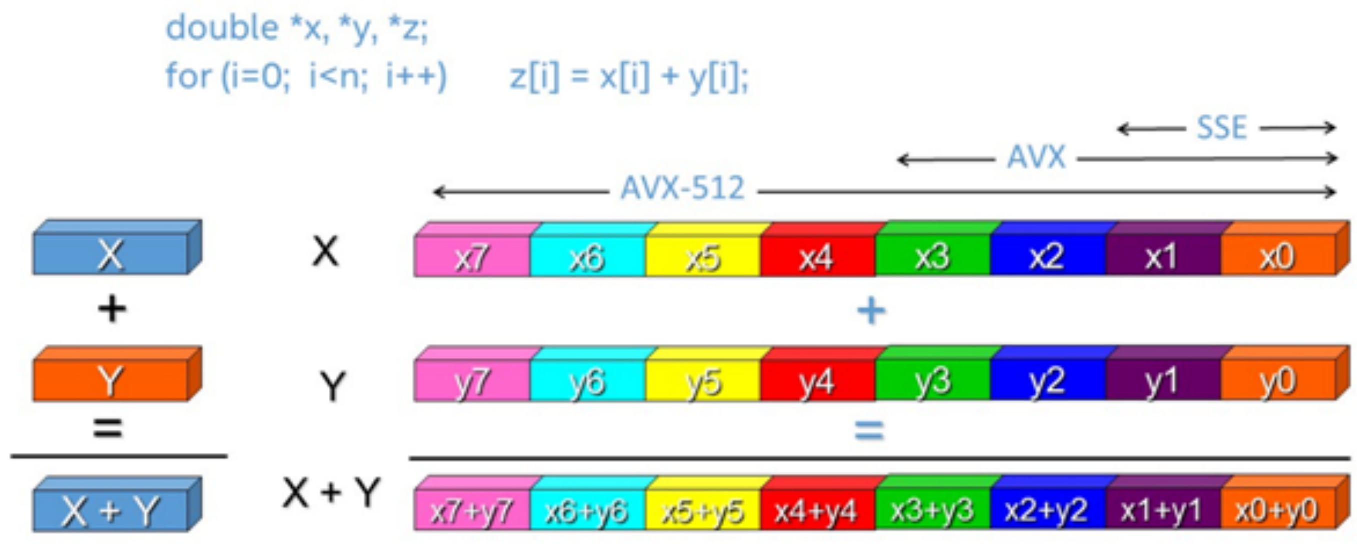


Figure 1 Scalar and vectorized loop versions with Intel® SSE, AVX and AVX-512.

Making the most of fine grain parallelism through vectorization allows the performance of software applications to scale with the number of cores in the processor enabling multithreading and multitasking.

marilena.bandieramonte@cern.ch

SIMD PROGRAMMING MODELS

How do we get it?

- Auto-vectorization
  - Compiler optimization converting repetitive scalar instructions (loops) to SIMD code
- Compiler pragmas
  - Code annotations persuading the compiler into vectorizing
  - OpenMP, CilkPlus
  - May not preserve exact scalar behavior
- SIMD libraries
  - VCL, Vc, UME::SIMD, VecCore
  - Explicit programming using specific vector types and operations
- Compiler intrinsics
  - Built-in inline compiler functions accessing architecture-specific vector instructions
- Assembly
  - The really low-level stuff on top of HW implementation

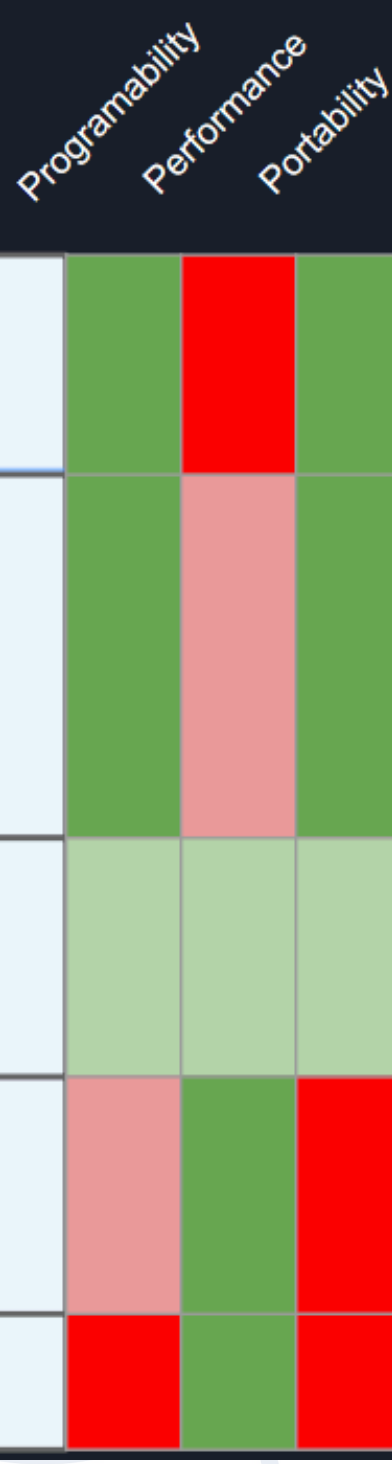
```
float a[N], b[N], c[N];

for (int i = 0; i < N; i++)
  a[i] = b[i] * c[i];
```

```
float a[N], b[N], c[N];

#pragma omp simd
#pragma ivdep
for (int i = 0; i < N; i++)
  a[i] = b[i] * c[i];
```

```
#include <VecCore/VecCore>
using Float_v =
   backend::VcVector::Float_v;
Float_v a = b * c;
```

```
#include <x86intrin.h>
__m256 a, b, c;

a = _mm256_mul_ps(b, c);
```

```
asm volatile(“vmulps %ymm1,
%ymm0”);
```



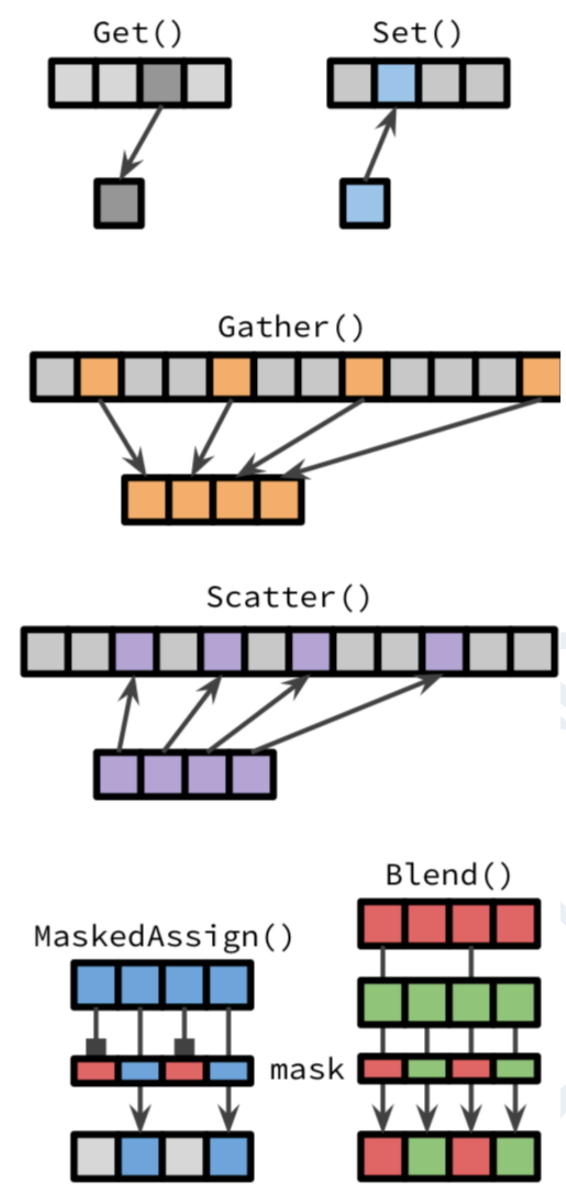
A. Gheata @WLCG and HSF workshop 2018 - Naples

marilena.bandieramonte@cern.ch

VECCORE: EXPLICIT SIMD VECTORIZATION LIBRARY

VecCore is a library providing a coherent set of abstract vector types and operations to enable **SIMD** vectorization on all platforms. This abstraction layer is built on top of the libraries Vc and UME::SIMD

```
namespace vecCore {

template <typename T> struct TypeTraits;
template <typename T> using Mask   = typename TypeTraits<T>::MaskType;
template <typename T> using Index  = typename TypeTraits<T>::IndexType;
template <typename T> using Scalar = typename TypeTraits<T>::ScalarType;

// Vector Size
template <typename T> constexpr size_t VectorSize();

// Get/Set
template <typename T> Scalar<T> Get(const T &v, size_t i);
template <typename T> void Set(T &v, size_t i, Scalar<T> const val);

// Load/Store
template <typename T> void Load(T &v, Scalar<T> const *ptr);
template <typename T> void Store(T const &v, Scalar<T> *ptr);

// Gather/Scatter
template <typename T, typename S = Scalar<T>>
T Gather(S const *ptr, Index<T> const &idx);

template <typename T, typename S = Scalar<T>>
void Scatter(T const &v, S *ptr, Index<T> const &idx);

// Masking/Blending
template <typename M> bool MaskFull(M const &mask);
template <typename M> bool MaskEmpty(M const &mask);

template <typename T> void MaskedAssign(T &dst, const Mask<T> &mask, const T &src);
template <typename T> T Blend(const Mask<T> &mask, const T &src1, const T &src2);

} // namespace vecCore
```

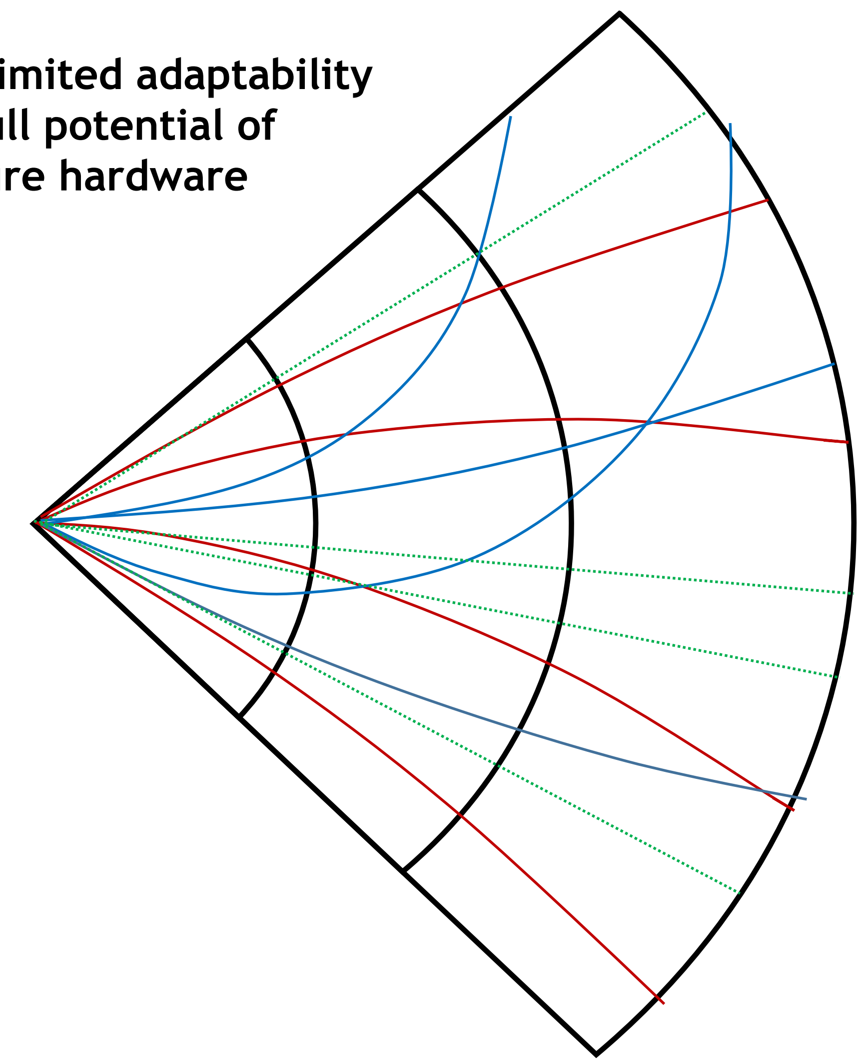


G. Amadio @HEP sw community meeting on GeantV RnD

marilena.bandieramonte@cern.ch

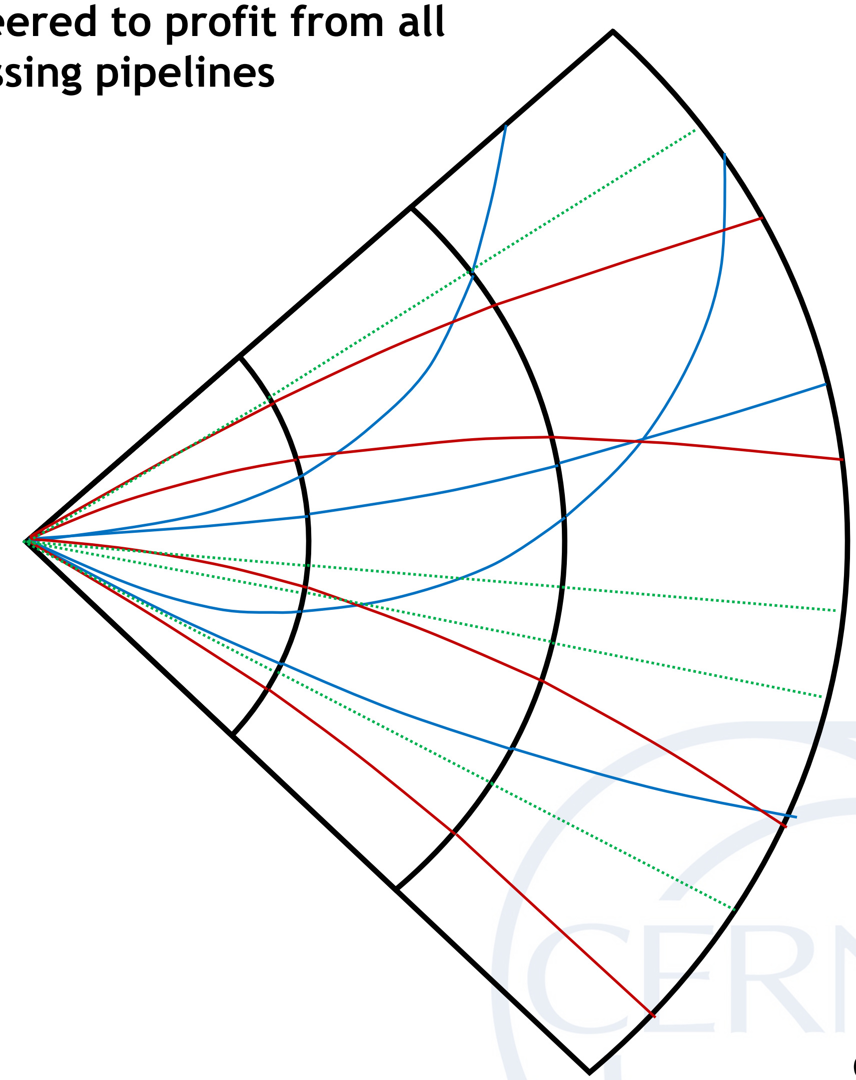
GEANTV – THE CONCEPT

Classical simulation (G3, G4 and others)
Flexible, but limited adaptability towards the full potential of current & future hardware



- One track at a time - Stack approach
- Single event transport
- Embarrassingly parallel
- Cache coherency – low
- Vectorization – low (scalar auto-vectorization)

GeantV simulation
Engineered to profit from all processing pipelines

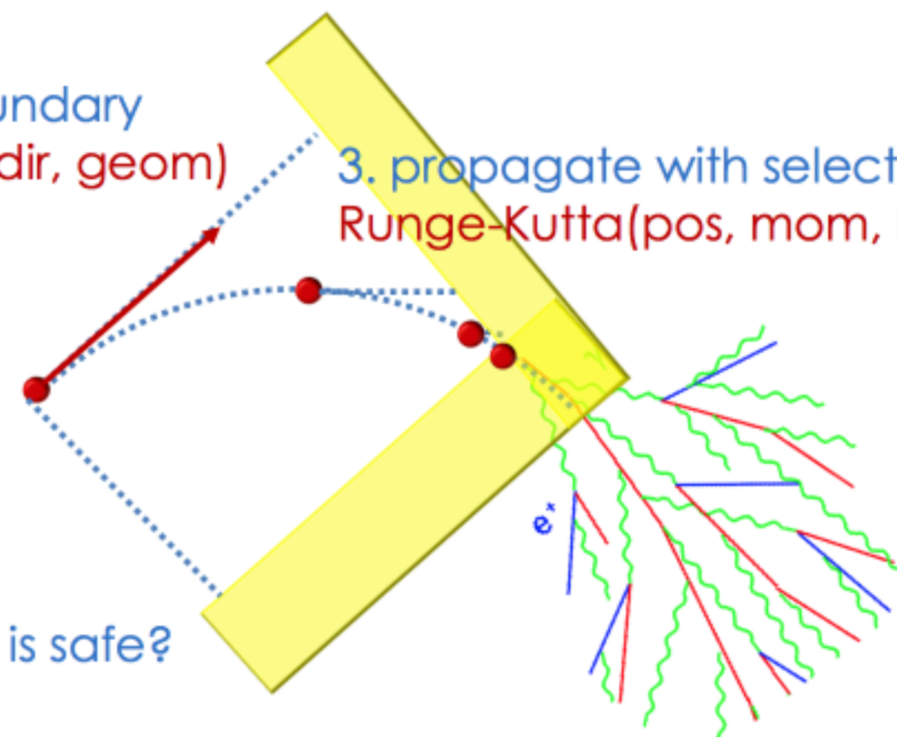


credit A. Gheata

- Basket approach
- Multi event transport
- Fine-grain parallelism + threads
- Cache coherency – high
- Vectorization – high (explicit multi-particle interfaces)

marilena.bandieramonte@cern.ch

PARTICLE TRANSPORT STAGES

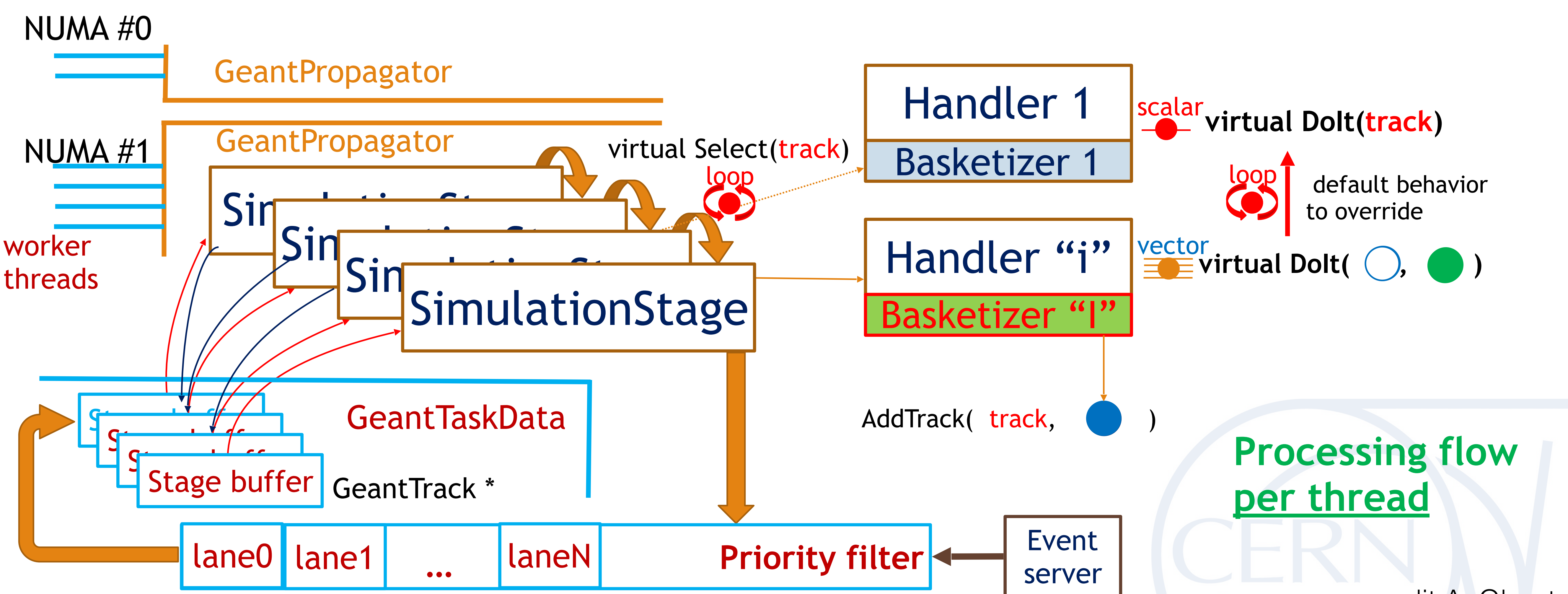


Do **stages** 1 to 6 for every track step – not very efficient….

Instead, accumulate particles and execute a stage when having a full "basket"
Do this for every stage in a concurrent environment -> **framework**

marilena.bandieramonte@cern.ch

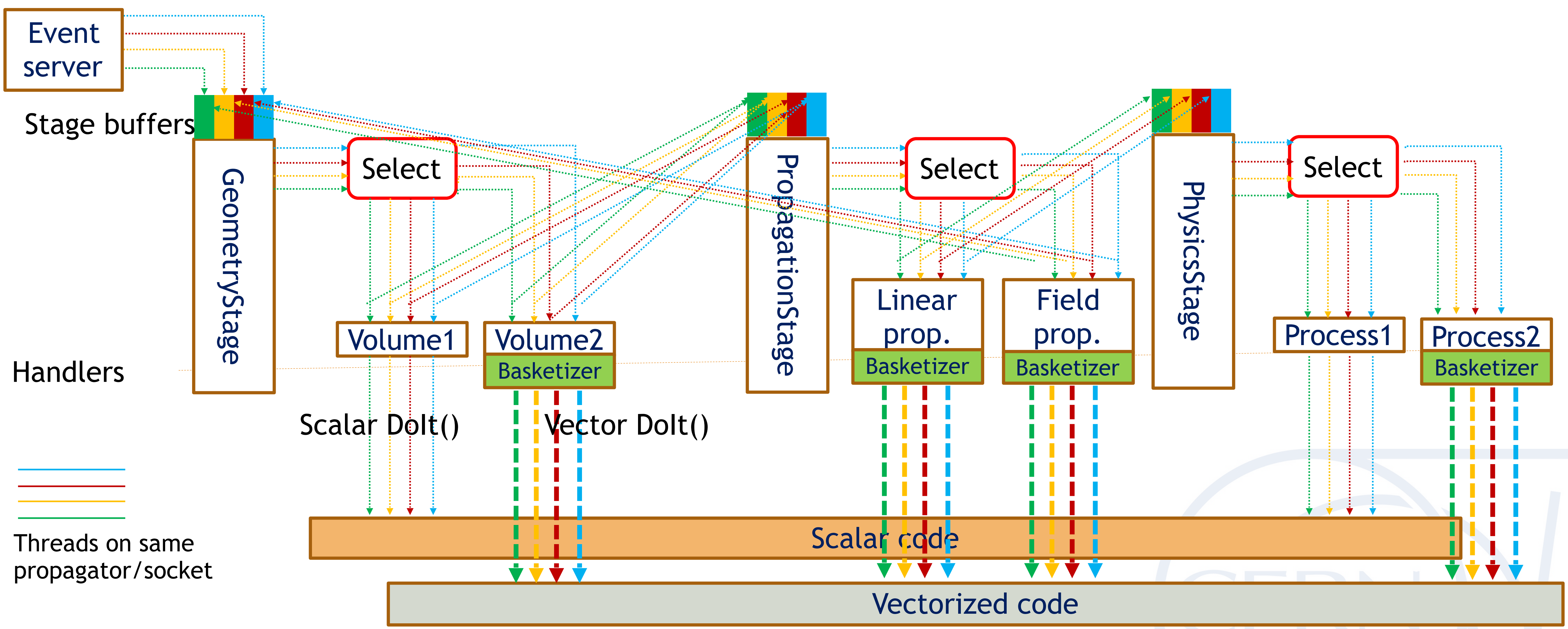
A GENERIC VECTOR FLOW APPROACH



credit A. Gheata

marilena.bandieramonte@cern.ch

SCALAR AND VECTOR PROCESSING COMBINED

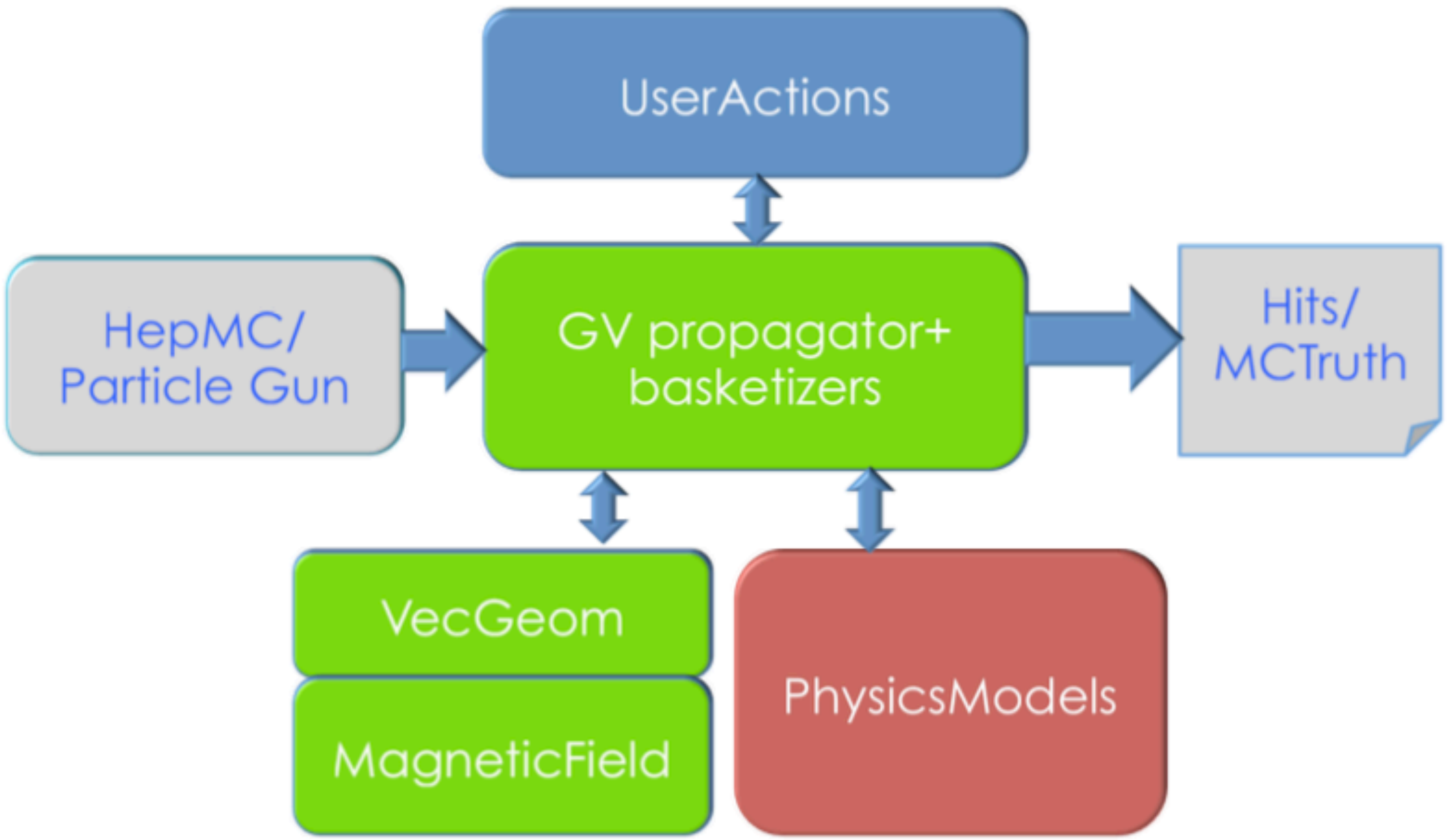


credit A. Gheata

marilena.bandieramonte@cern.ch

CURRENT OVERALL GEANTV STATUS

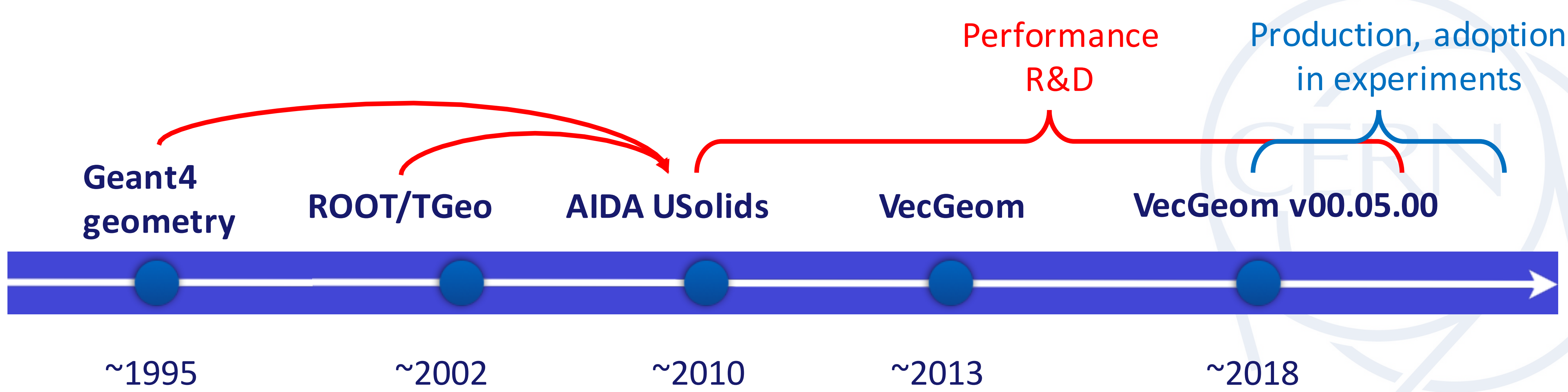
- alpha tag
  - complete version of scheduling, basketization
  - vectorized geometry
  - first vectorized magnetic field
  - full EM physics (scalar)
  - ‘user actions’
  - examples
    - realistic simulations
    - validation against Geant4 and data



W. Pokorski @WLCG and HSF workshop 2018 - Naples

VECGEOM EVOLUTION

- Geometry takes **30-40% CPU time** of typical Geant4 HEP Simulation
- AIDA project aiming initially to unify and modernize geometry algorithms
- Scope extended to encompass vectorization and multi-architecture/multi-platform support -> VecGeom
- 2018 marks phasing out the initial Usolids implementation while entering the production phase for VecGeom
  - Licence Apache 2.0 established
- Next: integration in ROOT & Geant4 as complete alternative to native navigation

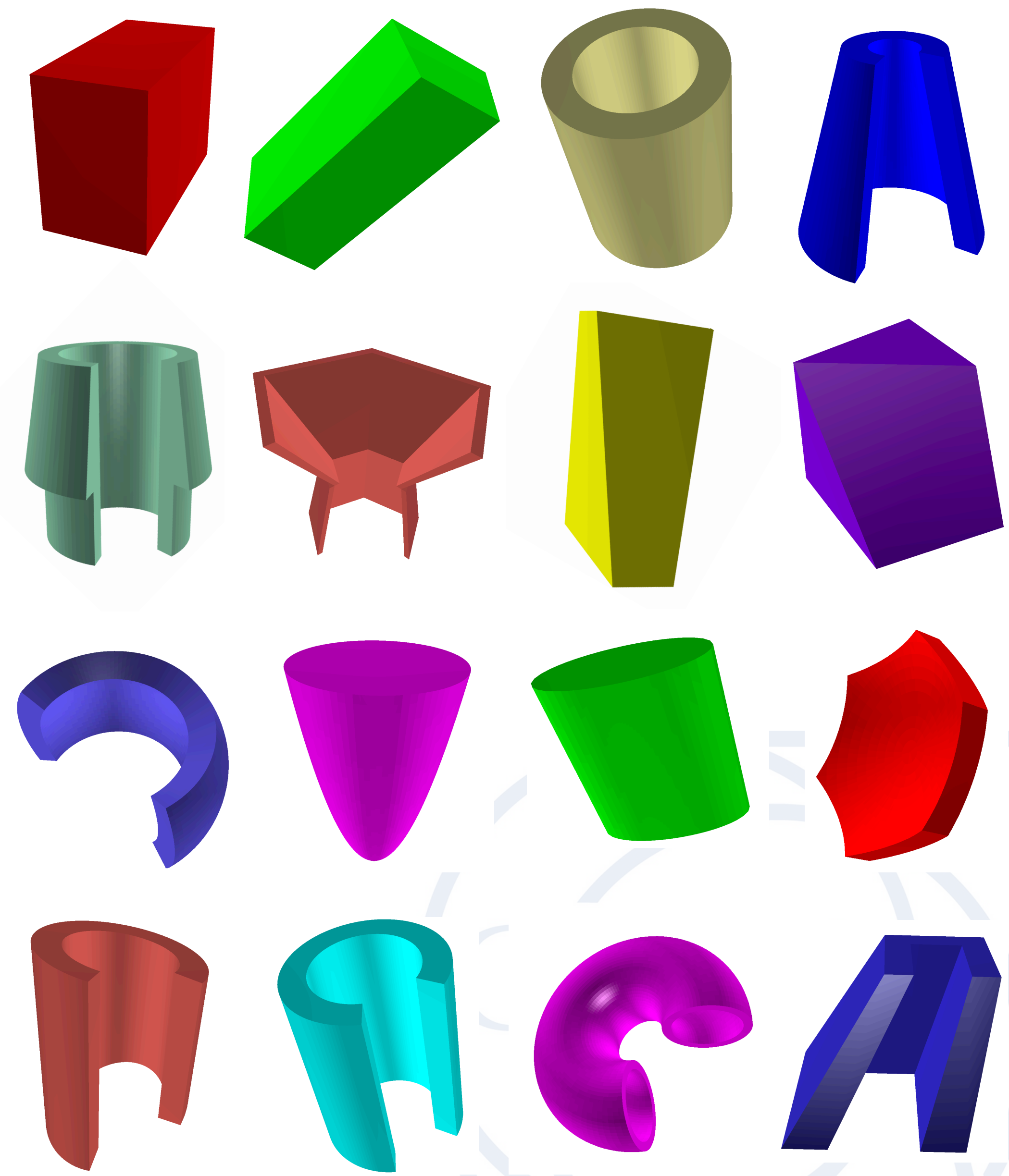


marilena.bandieramonte@cern.ch

VECGEOM: SHAPES IMPLEMENTATION STATUS

- **VecGeom**: A library of vectorised geometry algorithms to take maximum advantage of SIMD architectures

- Implementation **status**:
  - Implemented shapes:
    - Box, Orb, Trapezoid (Trap), Simple Trapezoid (Trd), Sphere (+ sphere section), Tube (+ cylindrical section) , Cone (+ conical section), Generic Trapezoid (Arb8), Polycone, Polyhedron, Paraboloid, Parallelepiped (Para), Hyperboloid, Ellipsoid, Torus (+ torus section), Scaled Solid, Boolean (addition, subtraction, intersection), Cut Tube, Simple SExtru, Extruded solid(expressed as specialization of tessellated-solid), Tessellated Solid, Multi-Union (WIP),
  - Missing:
    - Tetrahedron (Tet -> GenericTrapezoid)
    - Generic Polycone
    - Elliptical Cone, Elliptical Tube (example done)
      - can be composed through scaling
    - Half-spaces/planes
    - Twisted shapes (box, trap, tube)
      - complex and infrequent use

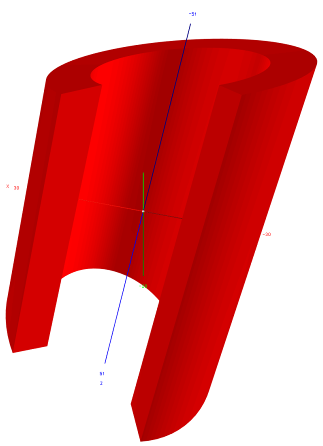


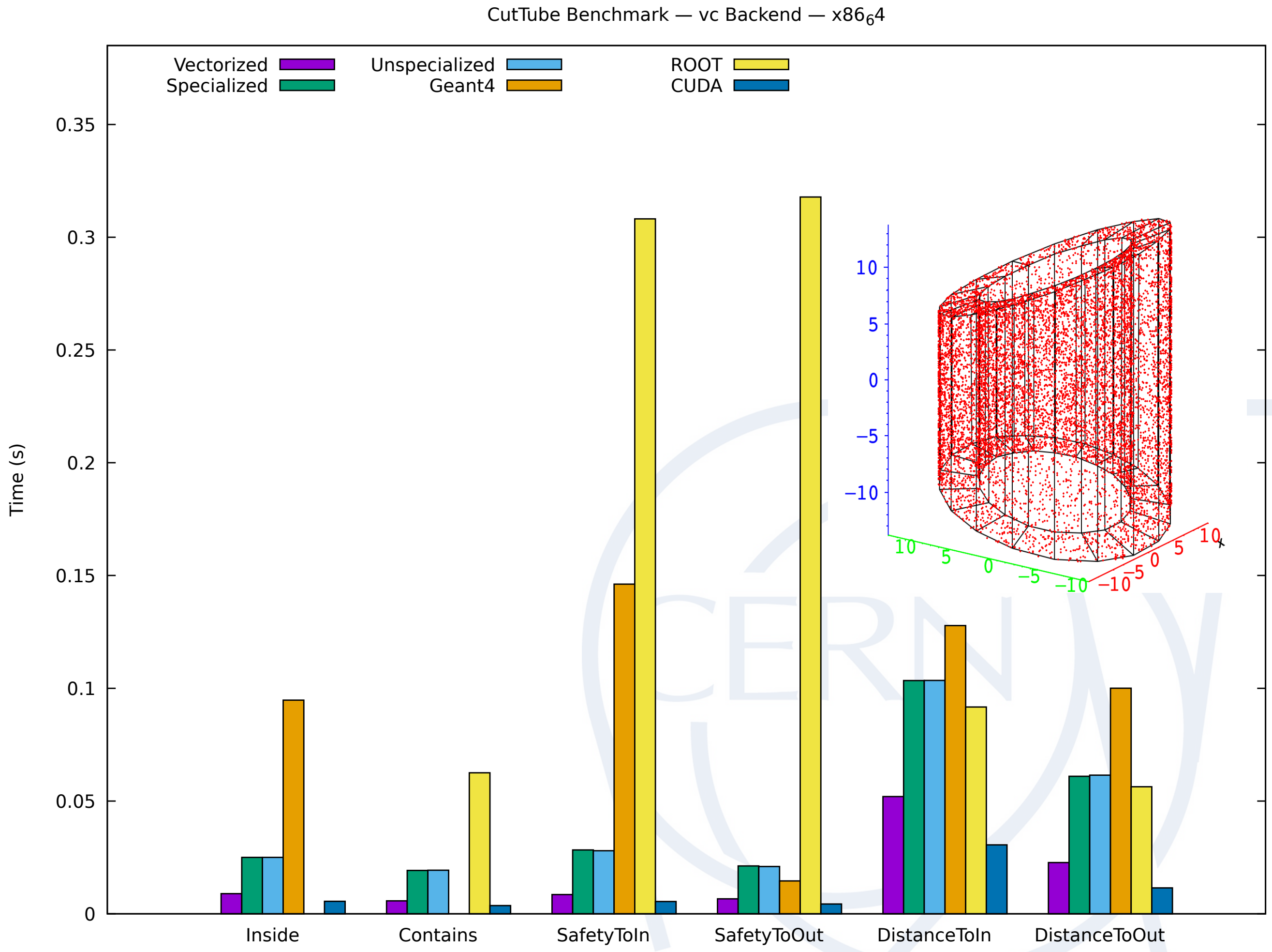
marilena.bandieramonte@cern.ch

EXAMPLE: CUT TUBE

- A **tube cut** by two planes
  - Planes cutting Z axis at +/- Z, defined by normal vectors oriented outwards the solid
- Available both in **ROOT** and **Geant4** geometry packages
- Existing in ALICE and new CMS geometry
- Implemented using as primitives the tube and plane implementations
  - Plane implementation to be reused for half-space primitive, used for defining Boolean solids in ROOT

- Overall scalar performance comparable to ROOT/Geant4
  - Much better for Inside, Contains and SafetyToIn





M. Gheata @AIDA-2020 Annual Meeting - April 2017

marilena.bandieramonte@cern.ch

GEANTV EM PHYSICS STATUS

- **Electromagnetic** (EM) transport simulation is challenging as it occupies a large part of the computing resources used in full detector simulation

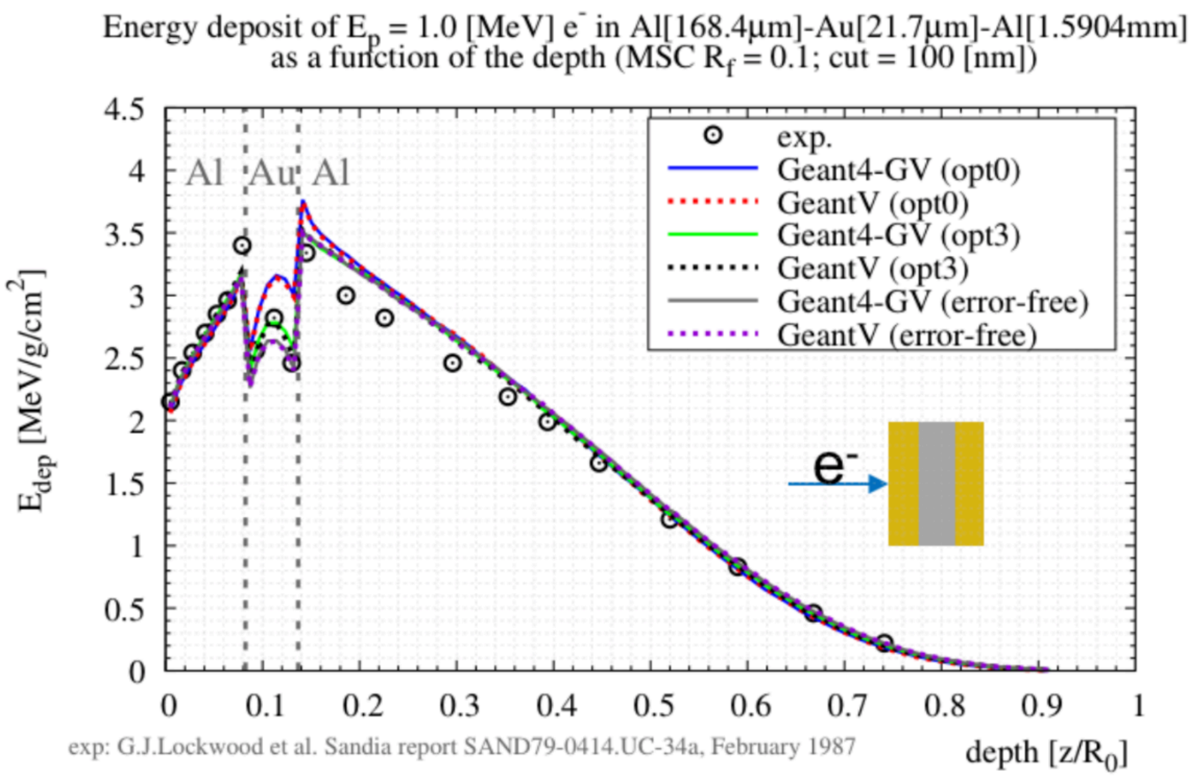
Current State

particle	processes	model(s)	
		GeantV	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 (2γ) [0-100TeV]	Heitler (2γ) [0-100TeV]
γ	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

- **EM showers** in GeantV can be **fully simulated in scalar mode** (models not vectorised) in single and multithreaded mode
- Every model **tested and verified against the corresponding Geant4** model (cross section per atom, cross section per volume, and kinematic of primary and secondary particles)

marilena.bandieramonte@cern.ch

REVISITED EM PHYSICS MODELS: VALIDATION



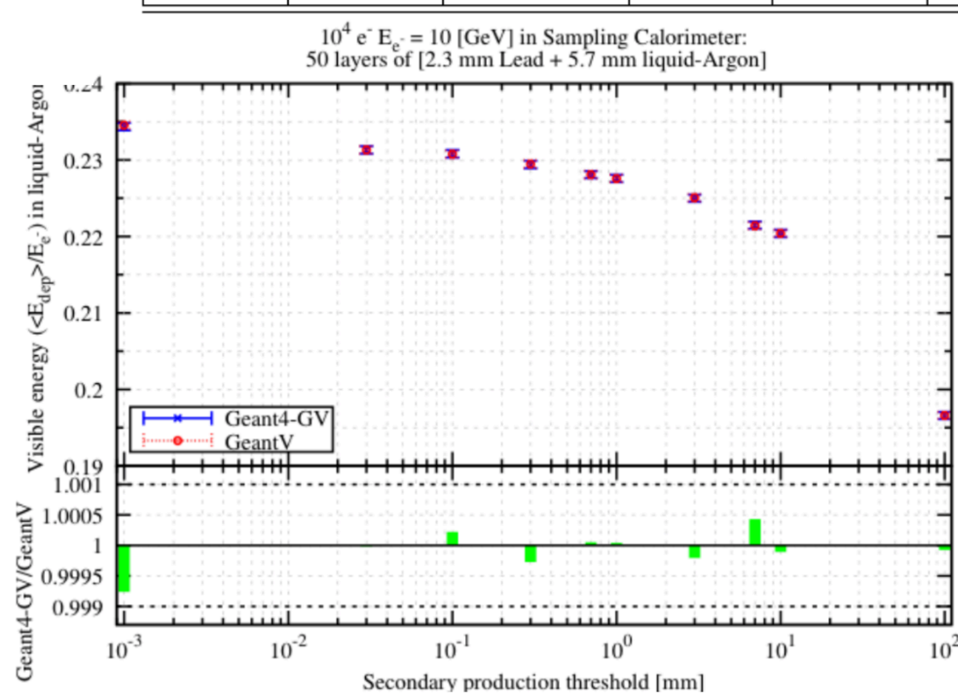
Multi-layered target

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; γ: 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

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



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

ATLAS simplified sampling calorimeter

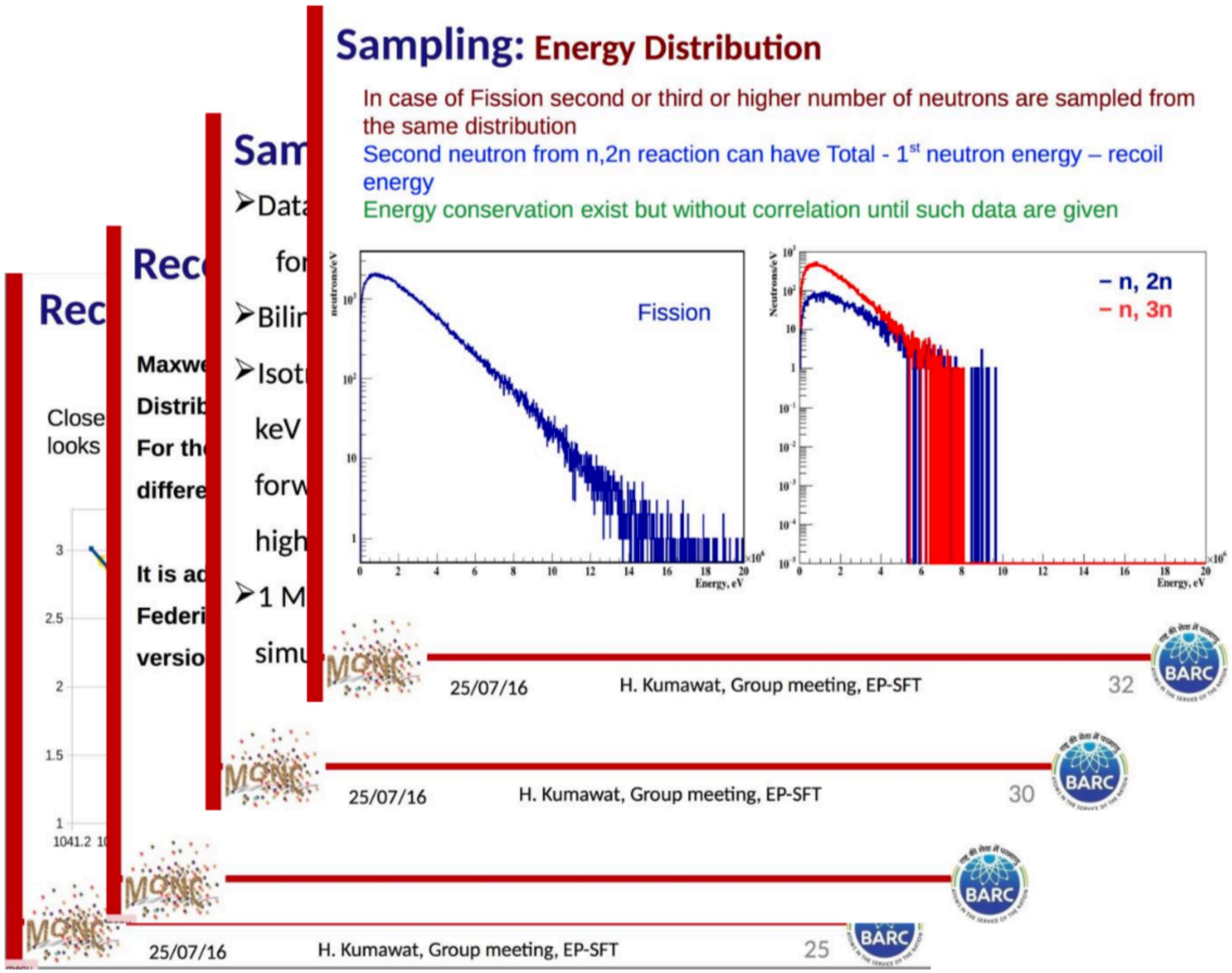
marilena.bandieramonte@cern.ch

HADRONIC PHYSICS

- Bertini is self contained enough to be interfaced to GEANTV as an external package **(WIP)**
  - With some surgery & mostly for testing purposes
- Further work after the beta and the proof of concept with vectorized EM shower:
  - EPOS (T.Pierog, Karlsruhe Institute of Technology, KIT) could be interfaced both to GEANT4 and to GEANTV as an external library

marilena.bandieramonte@cern.ch

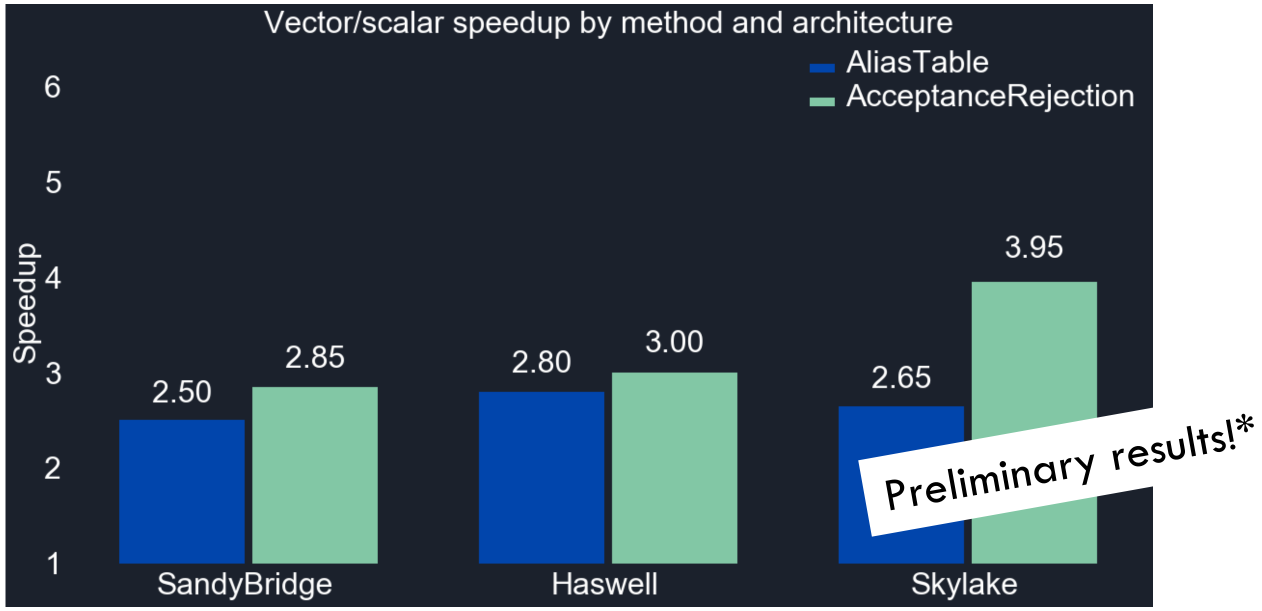
NEUTRONS



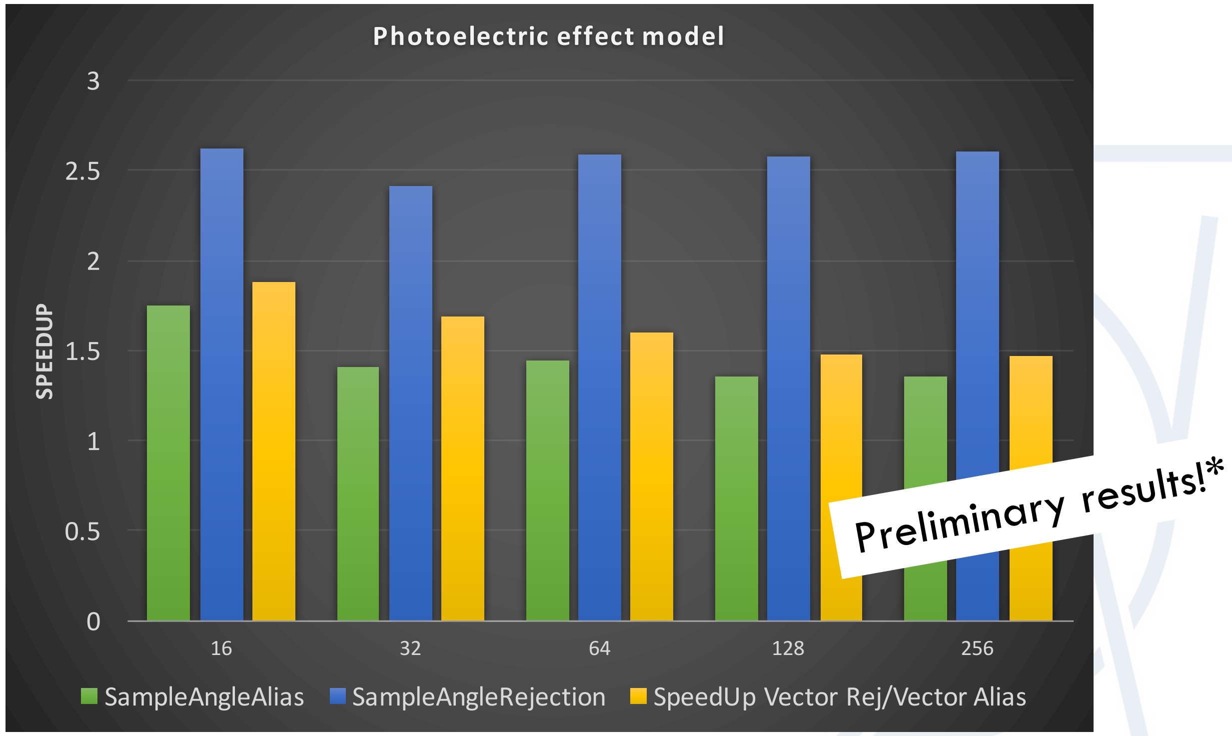
- Neutron transport could be a sweet-spot for vectorization
- We are preparing two "normalized tests":
  - The TARC experiment
  - The "Troitsk" experiment (ADS proposed setup)
- These exist in GEANT4 and we will reproduce them in GEANTV

PHYSICS VECTORISATION – WORK IN PROGRESS

- Perform **deep profiling** of the methods to highlight major (sometimes unexpected) **hotspots**
- **Re-think** and **re-design** some data structures and algorithms
  - Improve **cache locality** and **minimise** the need for **Gather** and **Scatter** operations (mainly scalar)
    - **Alias Table stored as AOS** instead of SOA with some precomputed values
  - Reduce the **conditional branches** and **unroll the loops** that kill vectorization
- Implementation and test of the **Filtering/Basketizing** mechanism for physics
  - Handlers per physics processes/model
- Vectorization of the **main functions/blocks** of code used in the physics library:
  - Alias/Rejection with shuffling Sampling
  - RotationToLabFrame
  - Generation of secondaries



Klein-Nishina sampling of final state performance

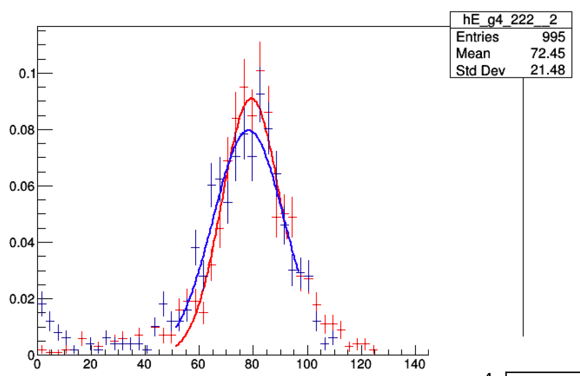


Photoelectric sampling of scattering angle performance

marilena.bandieramonte@cern.ch

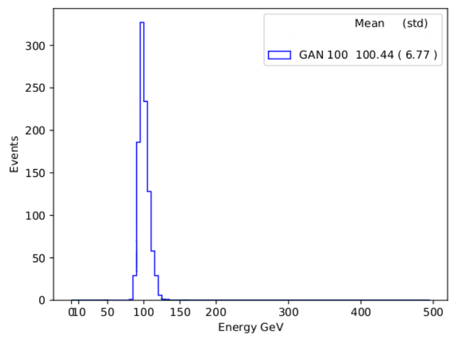
R&D ACTIVITY FOCUSED ON MACHINE LEARNING – FAST SIM

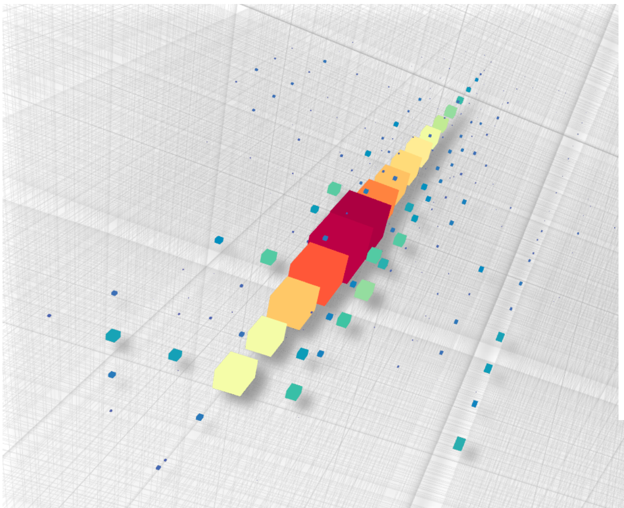
- For **LHC-Run 3**, the simulation demands will require an order of magnitude speed-up of the simulation applications
  - This number will only increase for future accelerators like HL-LHC and in particular the FCC and HE-LHC studies.
    —-> FAST SIMULATION IS NEEDED
- Fast simulation "**hooks**" à la G4 are being designed according to v3 flow
- There are different **ongoing RnD activities** on machine learning
  - Replacing the classical transport with trained neural networks to generate detector response
- **Example**: First ML prototype for simulation of high granularity calorimeters
  - Complete GAN based model for the simulation of particle showers in calorimeter (including particle type, energy, and trajectory)

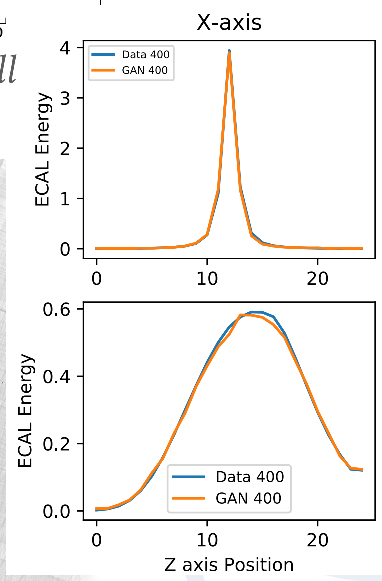


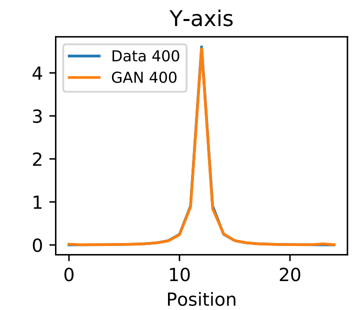
energy deposit - single cell

energy of particles to generate

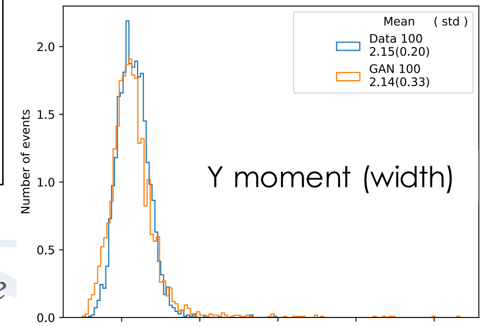








avg energy shower shape



2nd moment along y-axis

S. Vallecorsa@CERN openlab Technical Workshop, January 2018

marilena.bandieramonte@cern.ch

USER APPLICATION: EXAMPLES

- Main **GeantV** applications:
  - **TestEm5**: simulation of particle transmission through a simple slab
  - **TestEm3**: general simplified sampling calorimeter simulation to study EM shower simulation
    - optional MCtruth handling
  - **fullCMS**: general simulation
  - **fullLHCb**: general simulation + hits handling
  - **cmsToyGV**: example of interfacing of GeantV with a simplified multi-threading version of CMSSW
- The project is hosted at: https://gitlab.cern.ch/GeantV/geant
- GeantV website: http://geant.cern.ch

marilena.bandieramonte@cern.ch

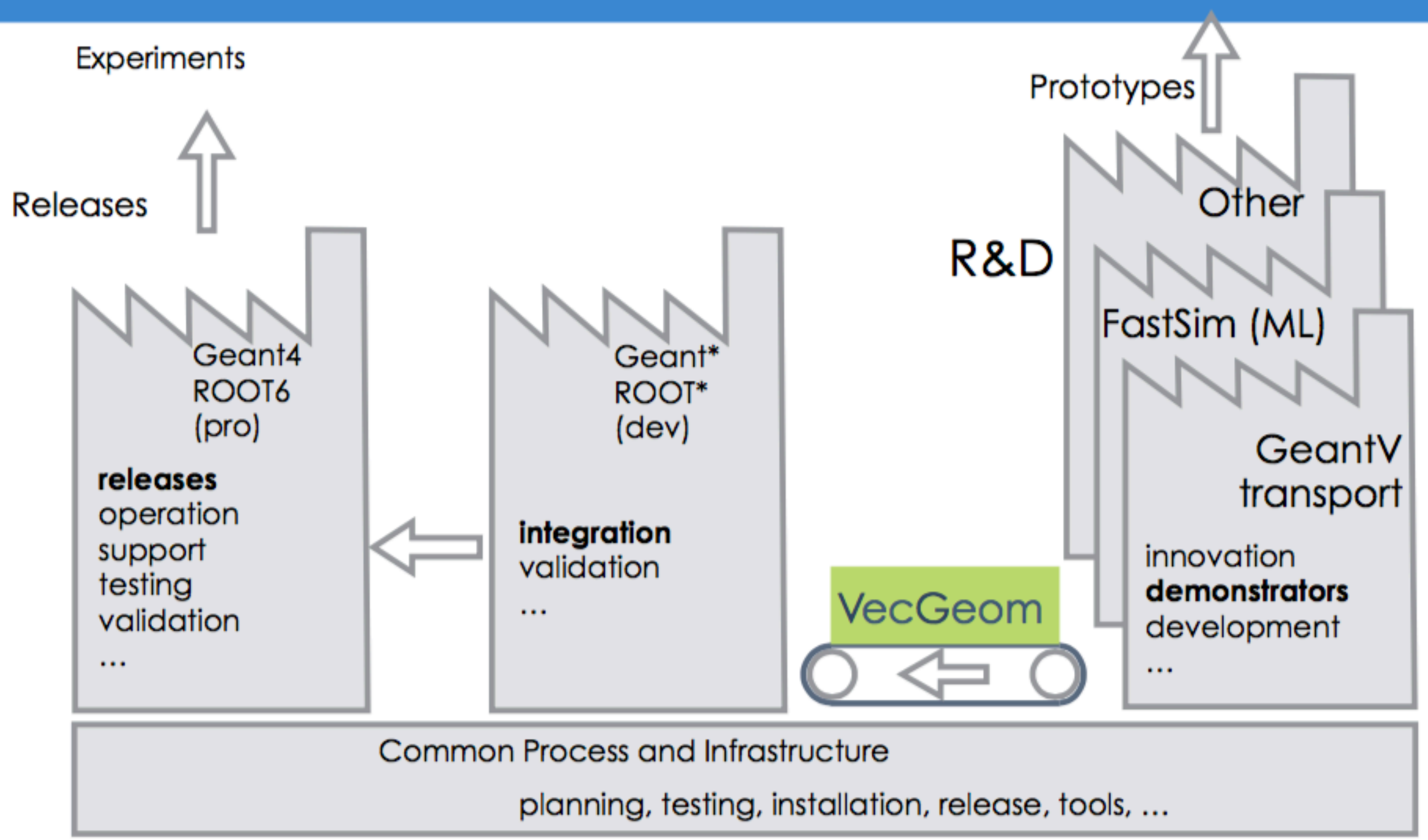
SUMMARY

- **GeantV R&D** aims at demonstrating benefits of vectorised particle transport+multithreading
  - Valuable components already delivered to the community via Geant4/ROOT: VecGeom/ VecCore, improved photoelectric model, MSC, pair-production
  - Aiming at complete EM shower simulation in a vector flow
- Status:
  - **Alpha** tag: fullEM transport, vectorized geometry/magnetic field, scalar physics, user interfaces, examples of different complexity
- **Plan for 2018:**
  - **Beta** tag: demonstrate achievable speedup in real applications:
    - EM shower transport fully vectorized
    - Examples including MC truth processing and I/O
    - Integration of fastsim hooks
      - ML based examples

marilena.bandieramonte@cern.ch

BACKUP SLIDES

Simulation software R&D

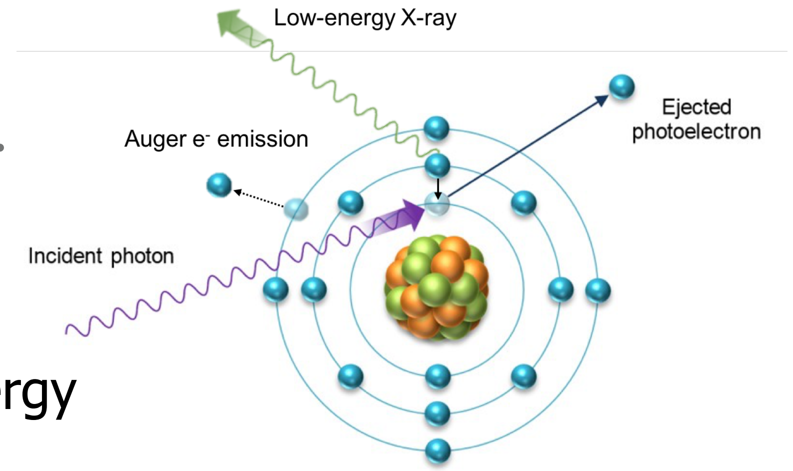


WHAT MAY PREVENT VECTORIZATION

- **Compiler must prove that the operation leads to same result -> Not always easy**
  - Loop data dependencies, pointer aliasing, non-inline function calls, early returns, nested control flow, conditional recursions, …
  - HEP code has lots of those…
- **Compilers may add runtime checks (tails, loop size), vectorize parts of the loop, or just refuse to vectorize**
  - Performance can vary wildly among compilers (https://godbolt.org)
  - And doing small changes in a large code base may silently disable auto-vectorization
- **In many cases there is no loop, or at too high level (events, tracks)**

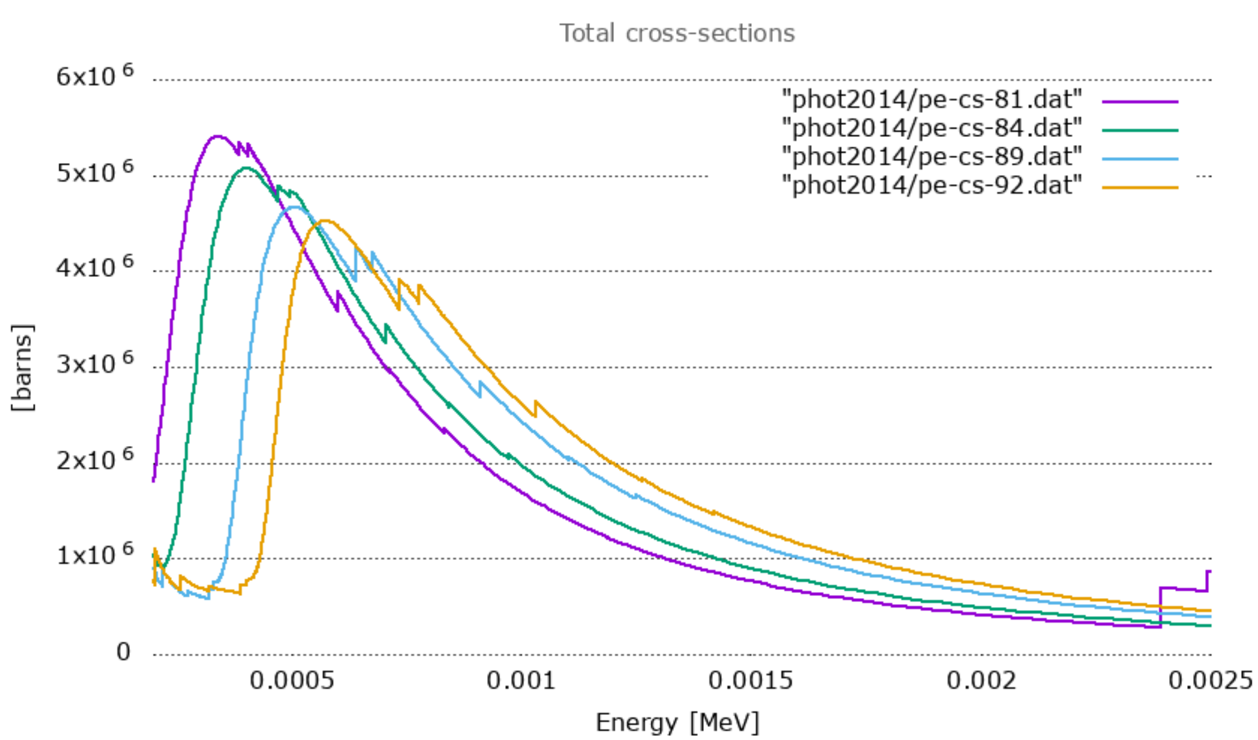
marilena.bandieramonte@cern.ch

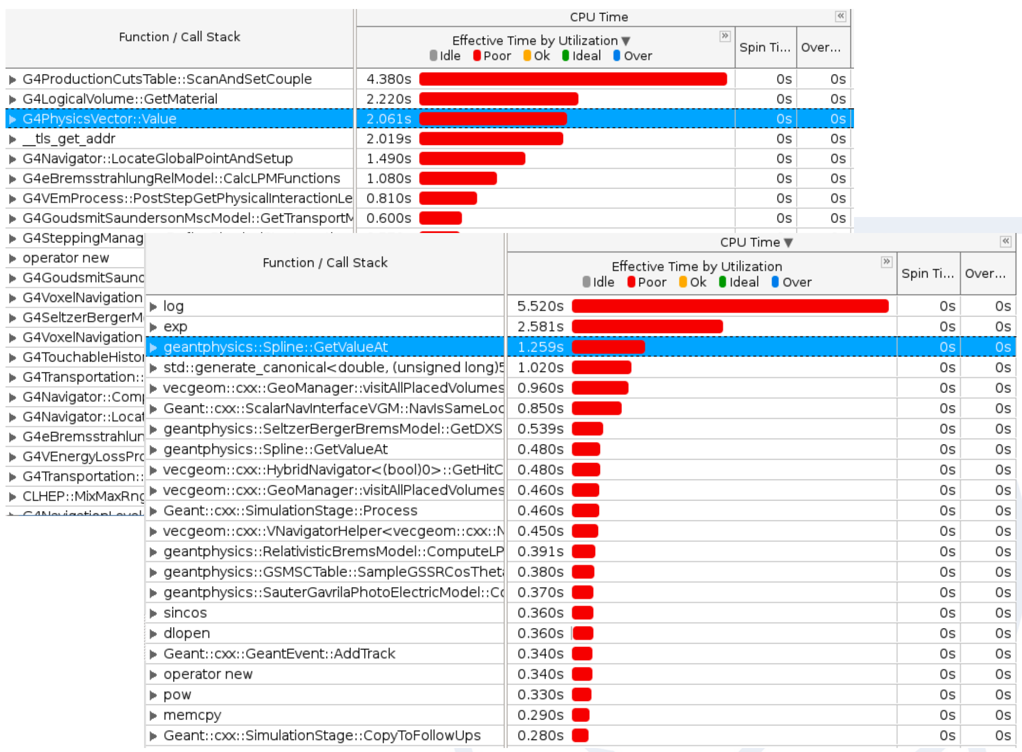
EXAMPLE: PHOTOELECTRIC EFFECT



- 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 subshell: This is going through a binary search algorithm (not vectorizable) + linear or spline interpolation
  - the angle: described by the SauterGavrila differential cross-section





Function / Call Stack	CPU Time: Effective Time by Utilization (Idle, Poor, Ok, Ideal, Over)	Spin Ti...	Over...
G4ProductionCutsTable::ScanAndSetCouple	4.380s	0s	0s
G4LogicalVolume::GetMaterial	2.220s	0s	0s
G4PhysicsVector::Value	2.061s	0s	0s
__tls_get_addr	2.019s	0s	0s
G4Navigator::LocateGlobalPointAndSetup	1.490s	0s	0s
G4eBremsstrahlungRelModel::CalcLPMFunctions	1.080s	0s	0s
G4VEmProcess::PostStepGetPhysicalInteractionLe	0.810s	0s	0s
G4GoudsmitSaundersonMscModel::GetTransportM	0.600s	0s	0s
G4SteppingManag			
operator new			
G4GoudsmitSaunc			
G4VoxelNavigation			
G4SeltzerBergerM			
G4VoxelNavigation			
G4TouchableHistor			
G4Transportation::			
G4Navigator::Comp			
G4Navigator::Locat			
G4eBremsstrahlun			
G4VEnergyLossPrc			
G4Transportation::			
CLHEP::MixMaxRng			

Function / Call Stack	CPU Time ▼: Effective Time by Utilization (Idle, Poor, Ok, Ideal, Over)	Spin Ti...	Over...
log	5.520s	0s	0s
exp	2.581s	0s	0s
geantphysics::Spline::GetValueAt	1.259s	0s	0s
std::generate_canonical<double, (unsigned long)5	1.020s	0s	0s
vecgeom::cxx::GeoManager::visitAllPlacedVolumes	0.960s	0s	0s
Geant::cxx::ScalarNavInterfaceVGM::NavIsSameLoc	0.850s	0s	0s
geantphysics::SeltzerBergerBremsModel::GetDXS	0.539s	0s	0s
geantphysics::Spline::GetValueAt	0.480s	0s	0s
vecgeom::cxx::HybridNavigator<(bool)0>::GetHitC	0.480s	0s	0s
vecgeom::cxx::GeoManager::visitAllPlacedVolumes	0.460s	0s	0s
Geant::cxx::SimulationStage::Process	0.460s	0s	0s
vecgeom::cxx::VNavigatorHelper<vecgeom::cxx::N	0.450s	0s	0s
geantphysics::RelativisticBremsModel::ComputeLP	0.391s	0s	0s
geantphysics::GSMSCTable::SampleGSSRCosThet	0.380s	0s	0s
geantphysics::SauterGavrilaPhotoElectricModel::C	0.370s	0s	0s
sincos	0.360s	0s	0s
dlopen	0.360s	0s	0s
Geant::cxx::GeantEvent::AddTrack	0.340s	0s	0s
operator new	0.340s	0s	0s
pow	0.330s	0s	0s
memcpy	0.290s	0s	0s
Geant::cxx::SimulationStage::CopyToFollowUps	0.280s	0s	0s

Profiling of a Geant4/GeantV application, revealing their major hotspots

marilena.bandieramonte@cern.ch

EXAMPLE: PHOTOELECTRIC EFFECT

Sampling of the shell

KinE

Z

LET — Linear Interpolation

HET — Spline Interpolation

BINARY SEARCH OVER DIFFERENT TABLES (PER Z, PER SHELL)

Not easily vectorizable!

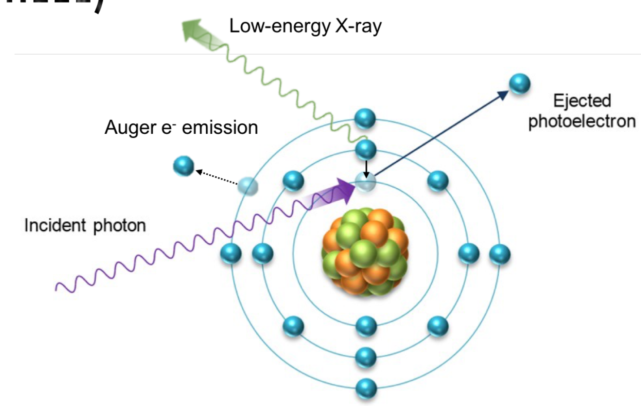
LEP — Parameterization (Polinomial fit)

HEP — Parameterization (Polinomial fit)

FETCH OF PARAMETERS (PER Z, PER SHELL)

Not easily vectorizable!

Additional filtering of particles



- 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: See next slide

marilena.bandieramonte@cern.ch

User interaction status

- user application, detector construction, physics list, magnetic field description very similar to Geant4
- Notifications during transport like for the other simulation packages
  - Very close to Geant4 style: Begin/End run, event, primary, …
- ‘User actions’ are more complex, due to:
  - Concurrency: scoring data has to be handled per thread -> thread safety
  - Event mixing: data has to be allocated separately per event slot!
    - GeantV provides services to deal with the extra complexity, with several examples

marilena.bandieramonte@cern.ch

HITS I/O AND MCTRUTH STATUS

- Infrastructure for hits-handling implemented
  - example(FullLHCb) implemented simulating full LHCb detector and saving (concurrently) hits
- Implemented necessary hooks for users to store particles history
  - Simple example implemented based on HepMC3 event record

marilena.bandieramonte@cern.ch