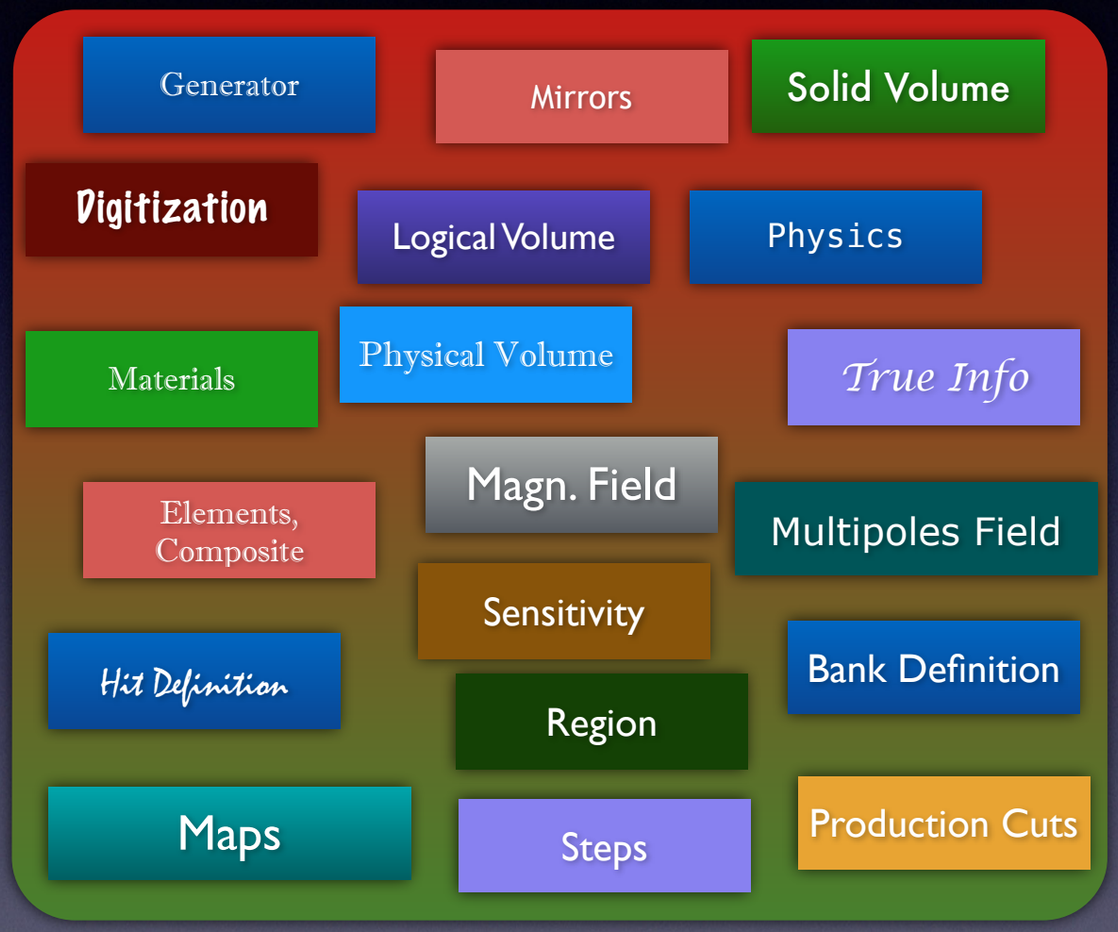
GEant4 Monte Carlo

Status of main components
What is working
What is not working
Outlook

Maurizio Ungaro, EIC Software Meeting
Jefferson Lab Sept 24, 2015

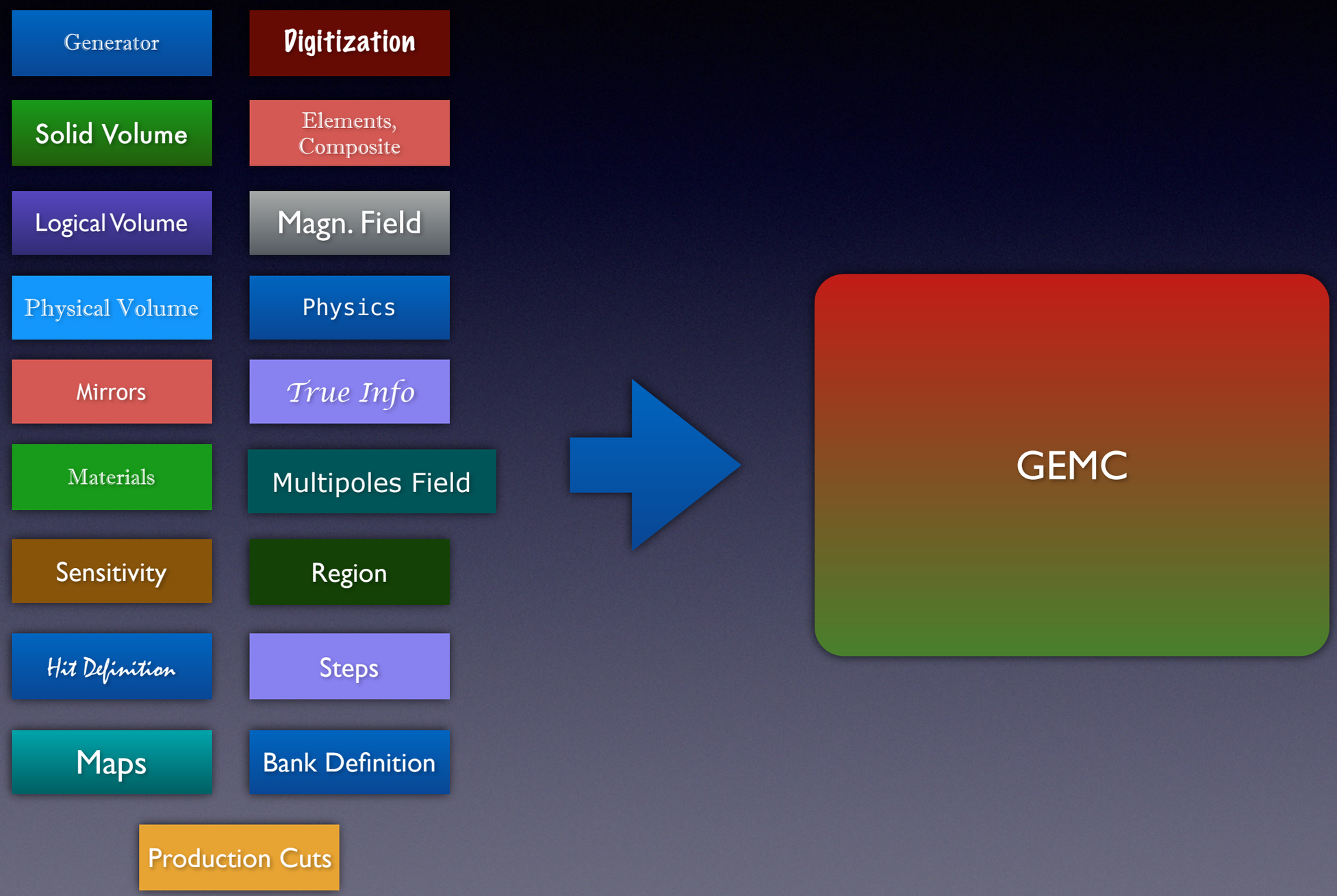
gemc 2 portfolio



Ingredients to cook a geant4 simulation

gemc 2 portfolio

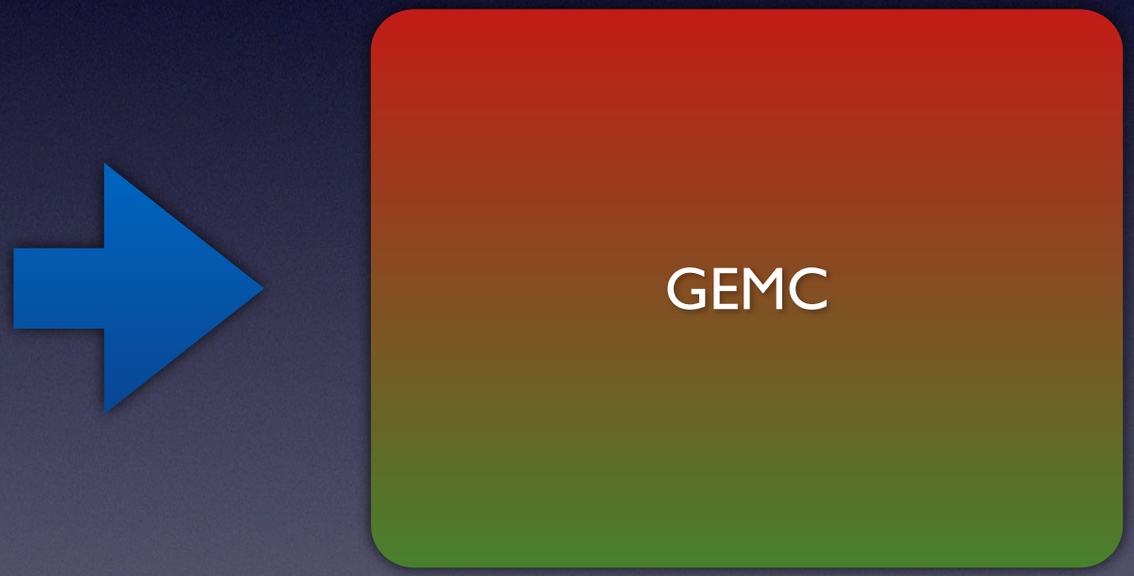
abstraction of "detector" to table of parameters



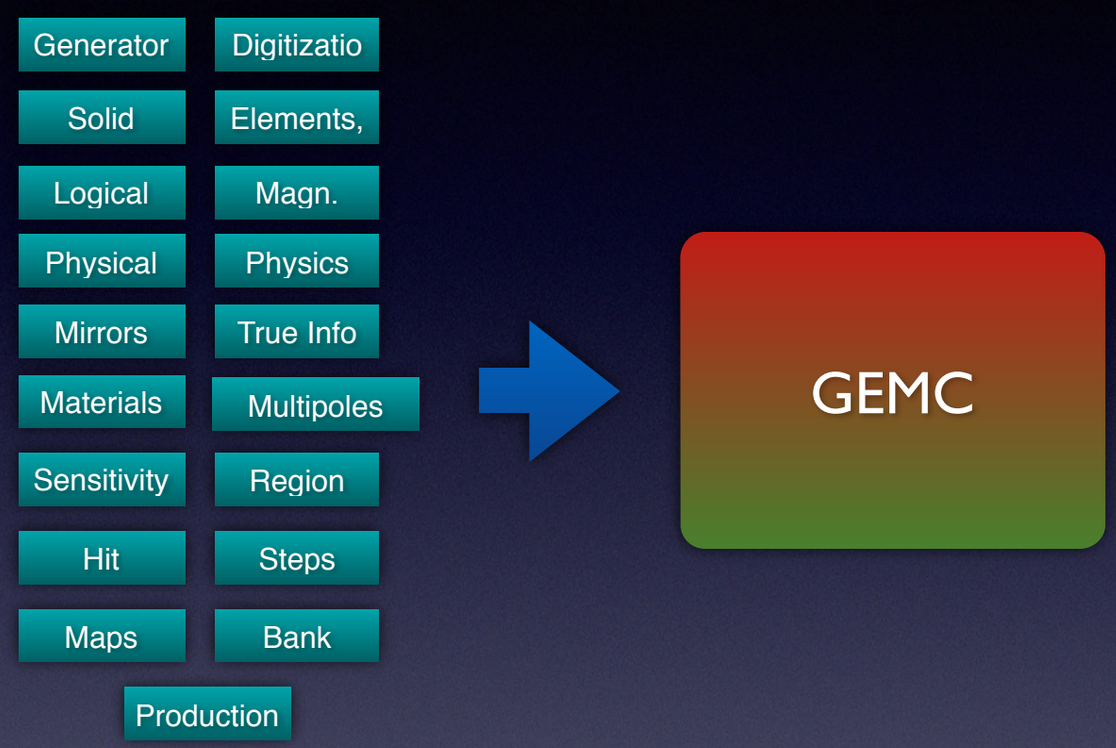
gemc 2 portfolio

standardized api for all components

- Generator
- Digitization
- Solid Volume
- Elements, Composite
- Logical Volume
- Magn. Field
- Physical Volume
- Physics
- Mirrors
- True Info
- Materials
- Multipoles Field
- Sensitivity
- Region
- Hit Definition
- Steps
- Maps
- Bank Definition
- Production Cuts



gemc API status



- Complete description of a detector outside code.
- detector is a c++ "object"
- current APIs:
  - perl
  - python (not completed)
  - java (in the works)
  - wanted: root?

Database: TEXT and MYSQL factories.
Both support "variation"
MYSQL supports "run number"

gemc API status

```
# Configuration file for ctof

# Detector name and variation
detector_name: ctof
variation:     original

# Factory can be MYQL or TEXT
factory:       TEXT

# run ranges and variation will apply to geometry,
materials and parameters
rmin:       1
rmax:       10000
```

Configuration file selects system name, factory, variation, run ranges

```
sub build_mother
{
        my %detector = init_det();

        $detector{"name"}        = "ctof";
        $detector{"mother"}      = "root";
        $detector{"description"} = "Central TOF Mother Volume";
        $detector{"pos"}         = "0*cm 0.0*cm 0*mm";
        $detector{"rotation"}    = "0*deg 0*deg 0*deg";
        $detector{"color"}       = "000000";
        $detector{"type"}        = "Tube";

        # allowing a gap to contain the paddles
        my $Rin        = $R - $dz - 0.1;
        my $Rout       = $R + $dz + 0.1;
        my $DZ         = $dy;

        $detector{"dimensions"}  = "$Rin*cm $Rout*cm $DZ*cm 0*deg 360*deg";
        $detector{"material"}    = "G4_AIR";

        print_det(\%configuration, \%detector);

}
```

api will read conf. file, build detector

gemc API status

Working?

- External database: YES
- API: perl works but perl is old. python or java are modern languages. Both not completed.
- ROOT api wanted.

generators

Internal: up to 3 particles
One primary two "beam"

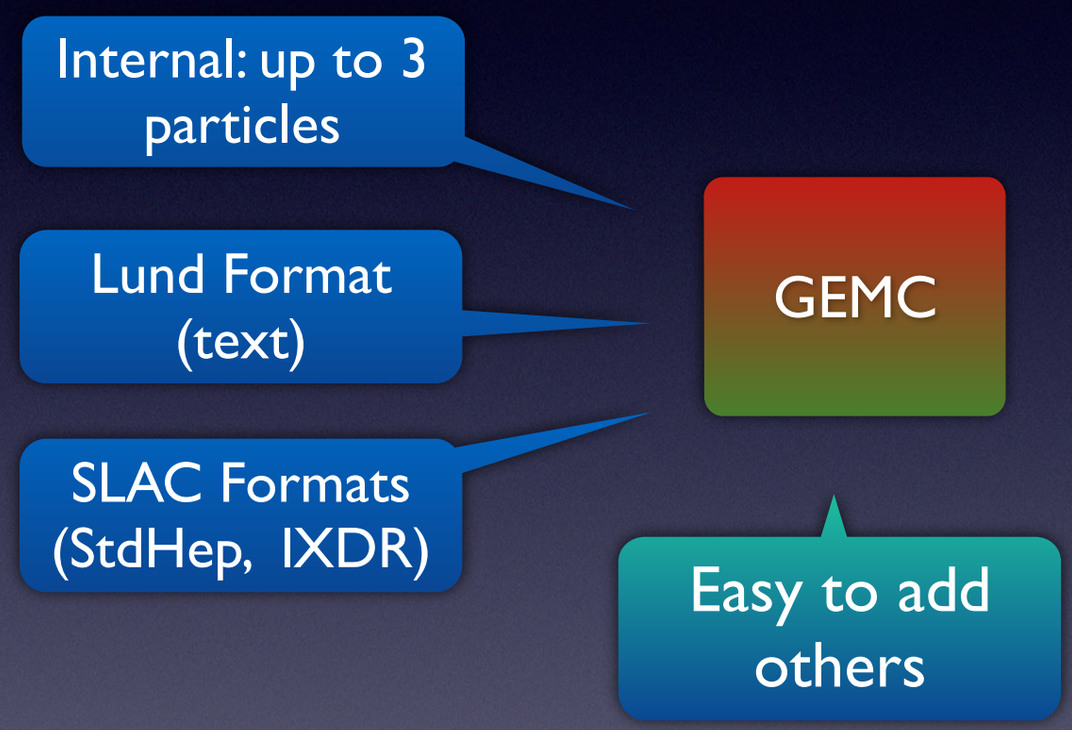
Lund Format (text)

GEMC

SLAC Formats
(StdHep, IXDR)

Easy to add others

generators



Working? yes, but not elegant.

Needed: factory mechanism.

Wanted: general geant4 MPrimaryGenerator plugin that could be loaded by gemc, fairroot, etc.

luminosity

Generating Background

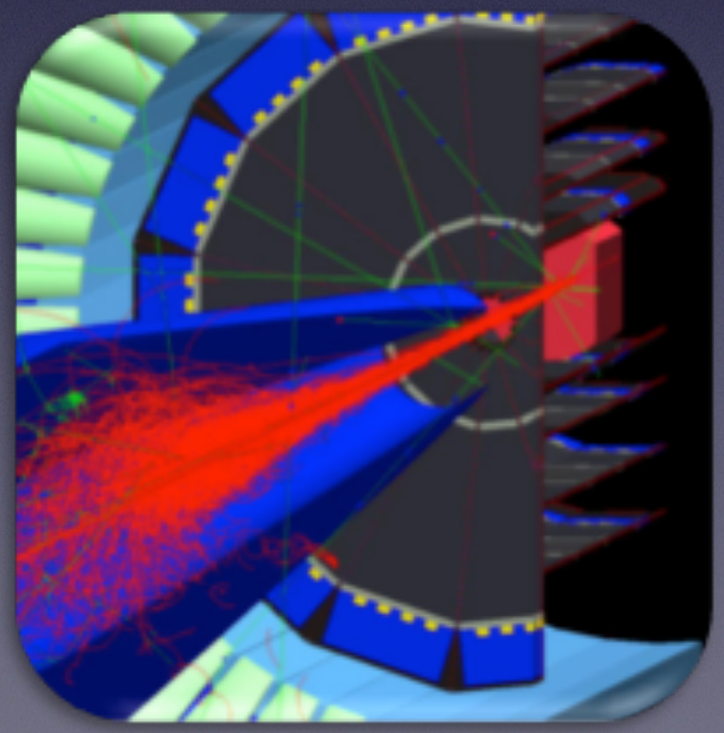
To add background coming from the beam the following quantities must be defined

1. a time window: the total time of one event
2. the number of beam particles for each event
3. the number of beam bunches

These quantities are defined with the **LUMI_EVENT** option. For example for clas12 10^{35} luminosity on 5cm LH2 target:

```
<option name="LUMI_EVENT"     value="124000, 250*ns, 2*ns" />
<option name="LUMI_P"         value="e-, 11*GeV, 0*deg, 0*deg" />
<option name="LUMI_V"         value="(0.,0.,-10.)cm" />
<option name="LUMI_SPREAD_V"  value="(0.01, 0.01)cm" />
```

Adds 124000 e- in 250 ns time window, grouped in 2 ns bunches. That would produce 125 bunches with 992 particles each bunch. The beam is 100 micron wide and starts 10 cm upstream of the center of the target.



Working?

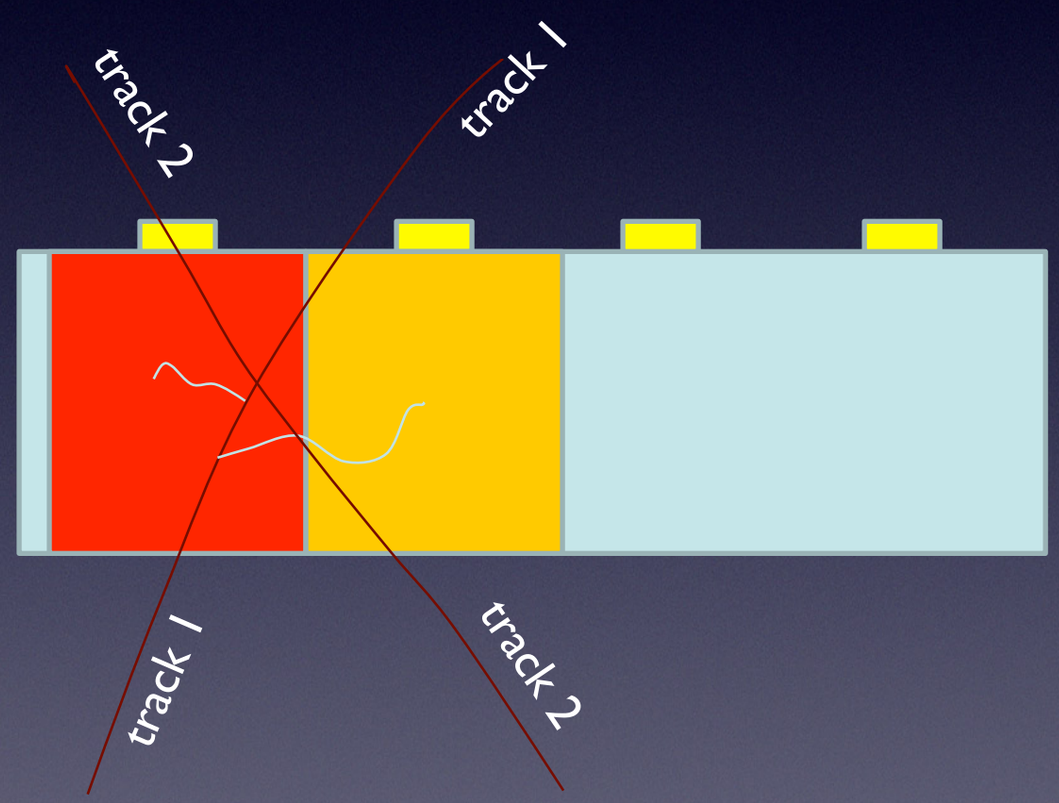
Yes, with one or 2 beams.

Wanted:

- gemc optimisation
- geant4 multithreading

electronic time window

Purpose: detector response is same as ADC/TDC/FADC signal.



All steps in the same time window, same detector cell form a hit

Working: awesome.

advanced digitization

"advanced digitization":

- Digitization should be AGARA: As Good As Reasonably Achievable
- Calibration Constant, mechanism should not slow down or complicate GEMC. Same DB.

advanced digitization

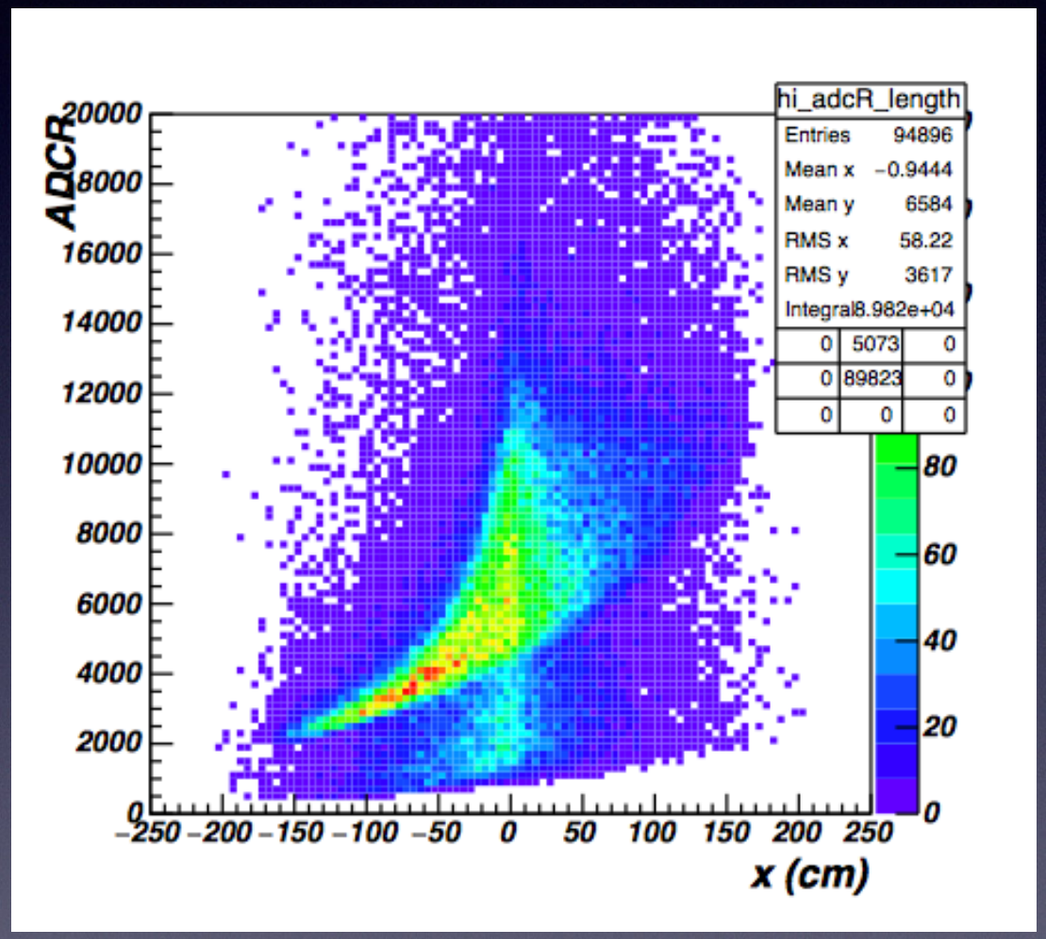
ADC:

- Attenuation according to exponential law
- Conversion from energy to ADC based on MIP signal (dEdxMIP=2 MeV/cm, countsForAMinimumIonizing=2000)

TDC:
- Delay due to light propagation in the paddle (effective velocity)
- Parameterized Time Walk
- Gaussian time spread based on parameters that will be matched to data (σ2=σ02+σ12/√EPMT Conversion from time to TDC (time2tdc=20ns-1)

Output: both “smeared” and “unsmeared” TDCs

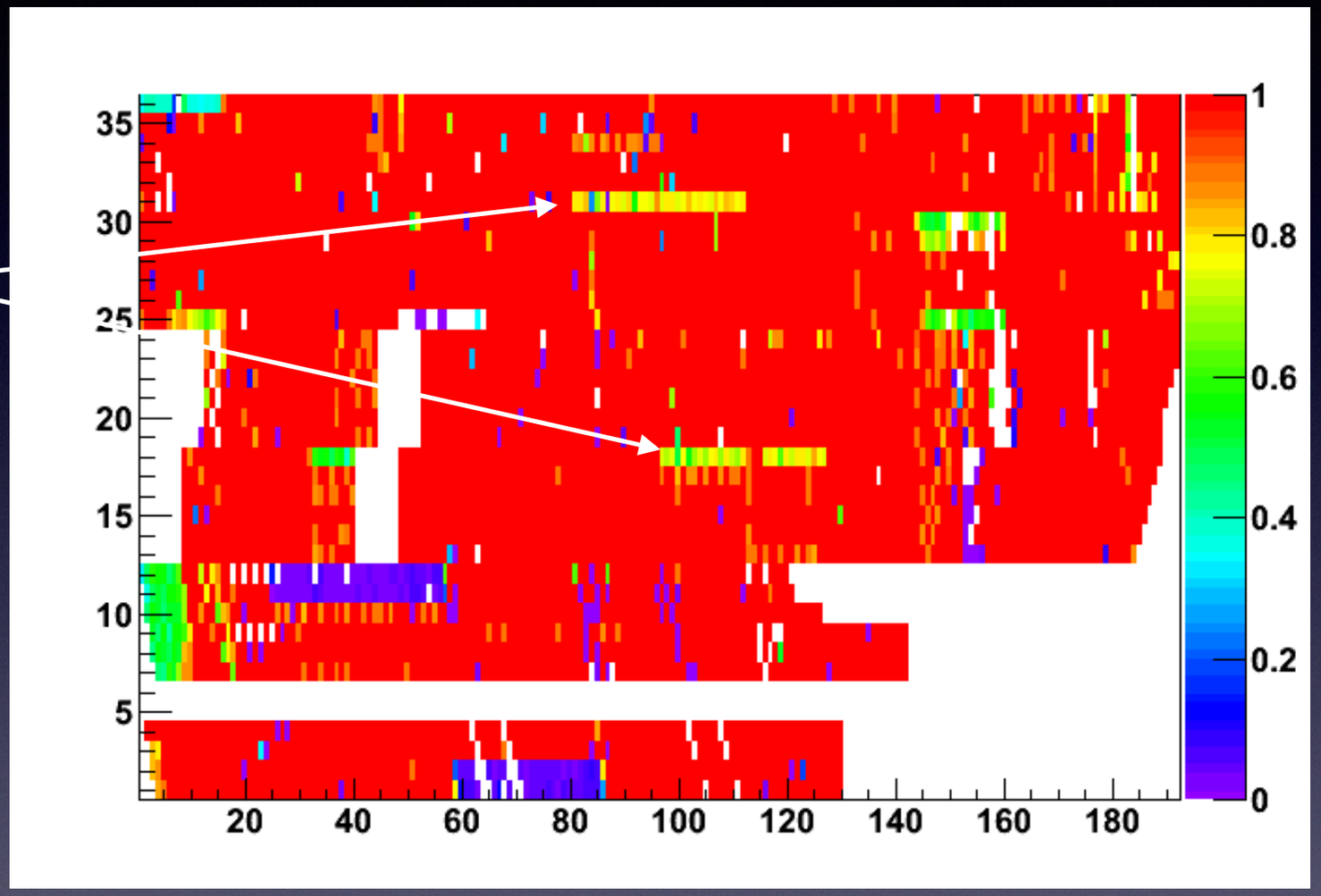
Status:
0 - fully functioning
1 - noADC
2 - noTDC
3 - noADC, noTDC(PMT is dead)
5 - any other reconstruction problem



advanced digitization

DC Efficiency for g12
runs contributed proportionally to their FC Q

dead 50% of the time, correlated!



The right thing to do:
Kill dead channels based on individual runs
(logistic nightmare)

advanced digitization

-RUN_WEIGHTS= “runs.txt”

2	0.1
13	0.6
22	0.2
30	0.1

-N=100,000 generates:

10,043 with constants from run 2
59,901 with constants from run 13
20,034 with constants from run 22
10,022 with constants from run 30

Run number put in header bank

(events are also ordered by run)

advanced digitization

TEST: FTOF paddle

-RUN_WEIGHTS="runs.txt" -N=100000

```
> Run weights table loaded:
   - run: 2     weight: 0.1      n. events: 10043
   - run: 13    weight: 0.6      n. events: 59901
   - run: 22    weight: 0.2      n. events: 20034
   - run: 30    weight: 0.1      n. events: 10022
```

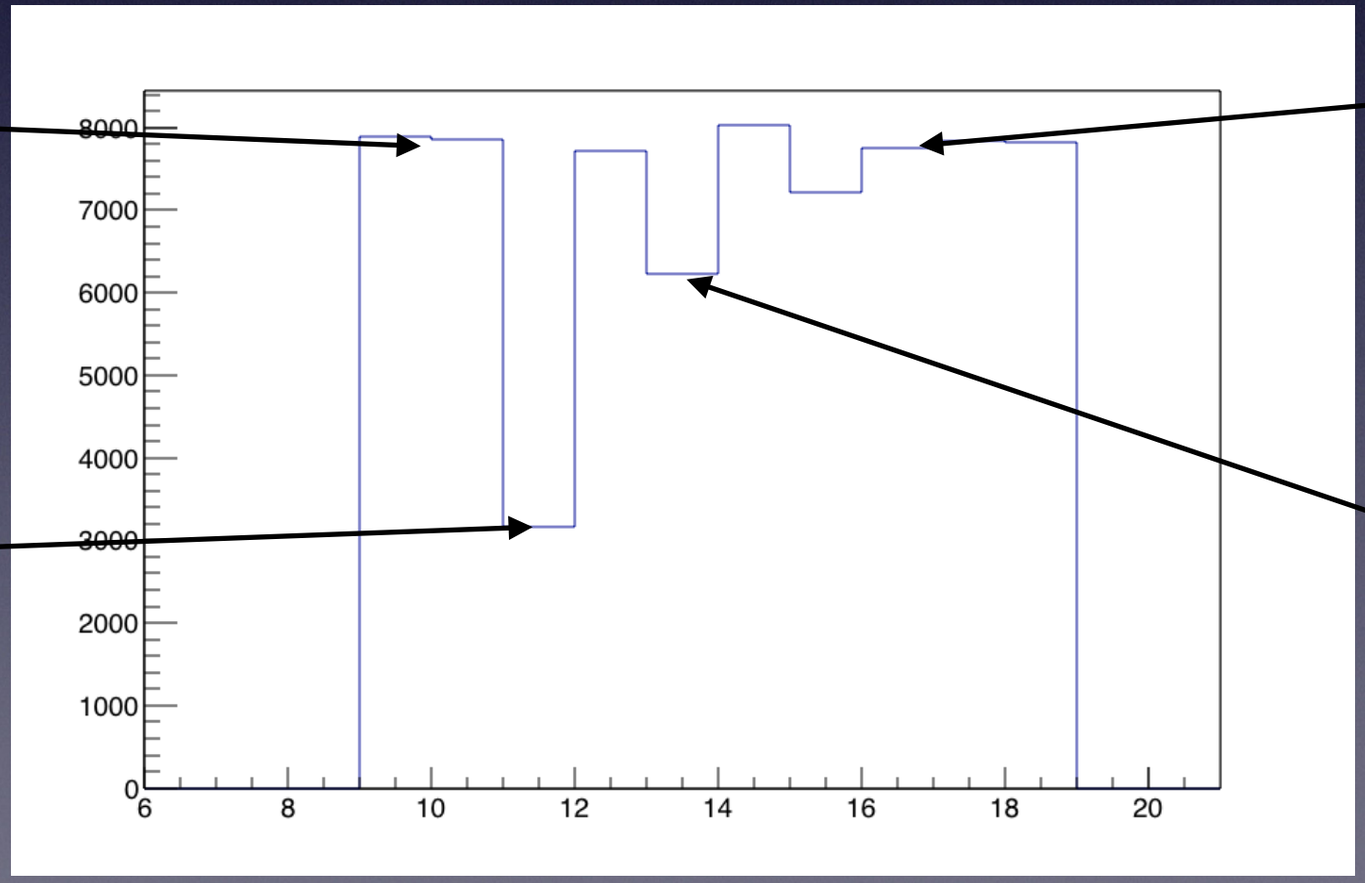
DB status “3”

for paddle 11 in run 13
for paddle 13 in run 22
for paddle 15 in run 30

7.8K

3156

(40.4%)



7208

(92.4%)

6226

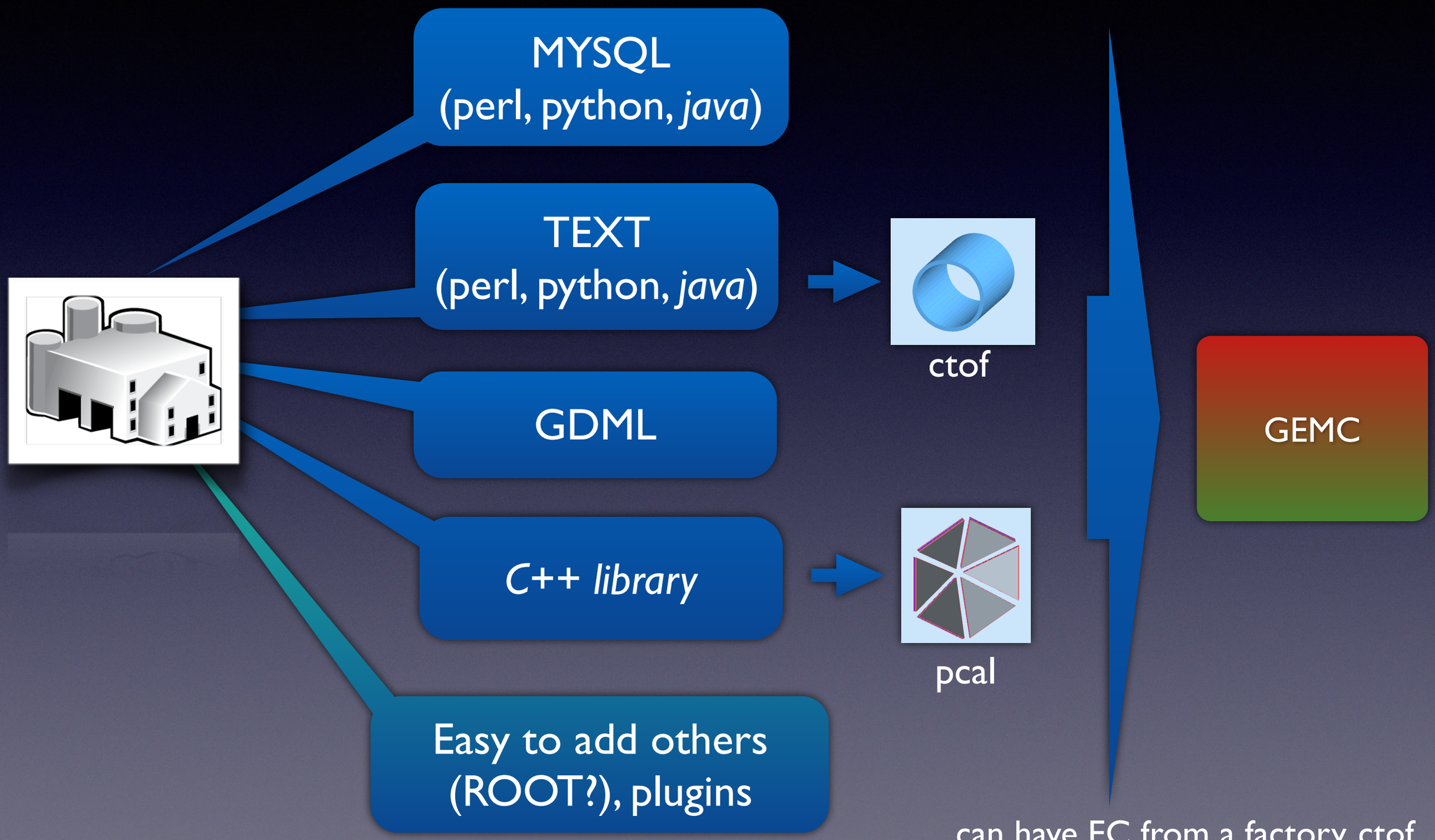
(79.8%)

advanced digitization

Working: prototype stage. Awesome

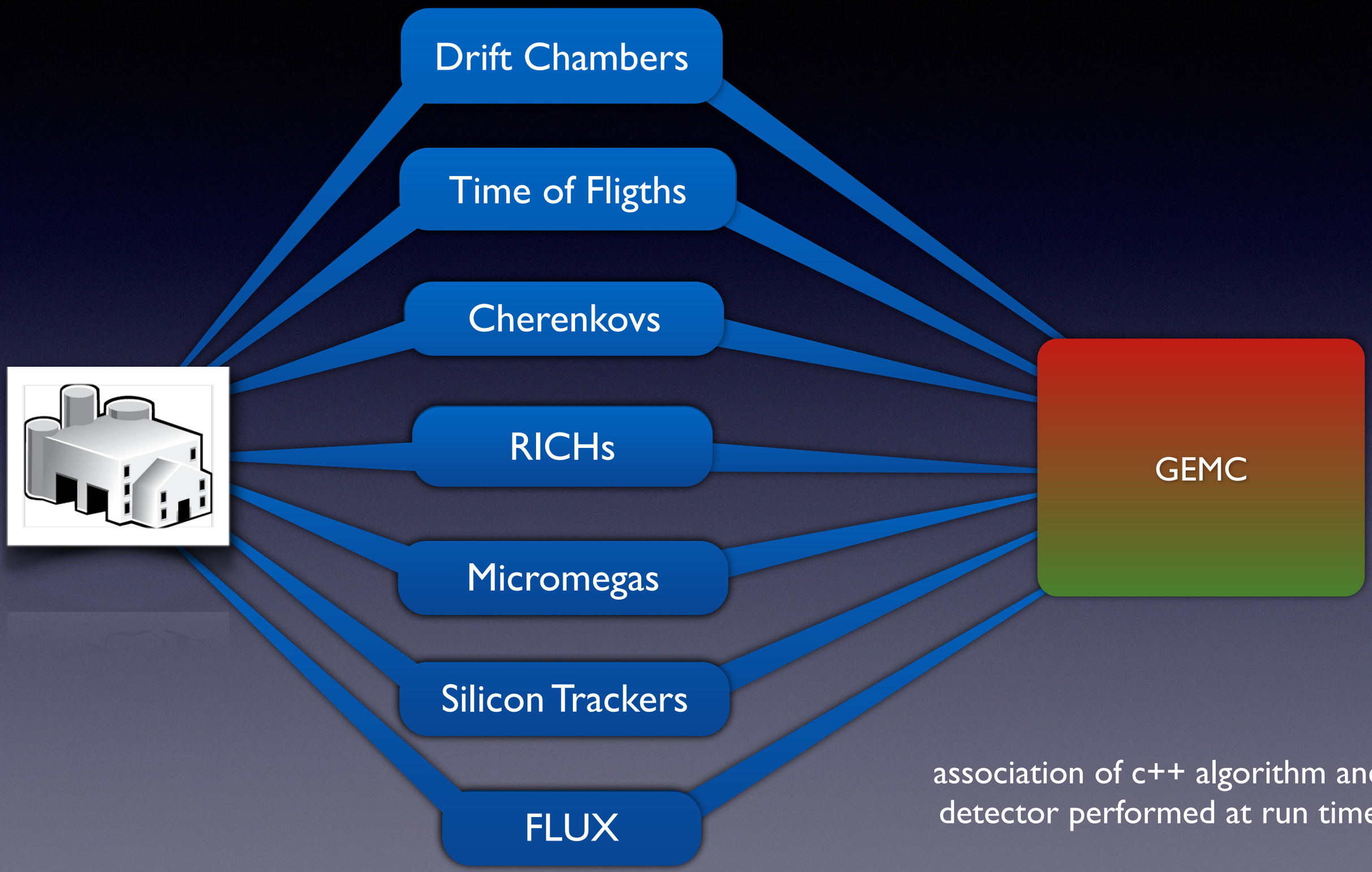
- parameters are the same as used in reconstruction, calibration (same DB)
- can “label” instead of “kill”
- both smeared and unsmeared output
- can always do digitization at a second stage by external software

detector factories



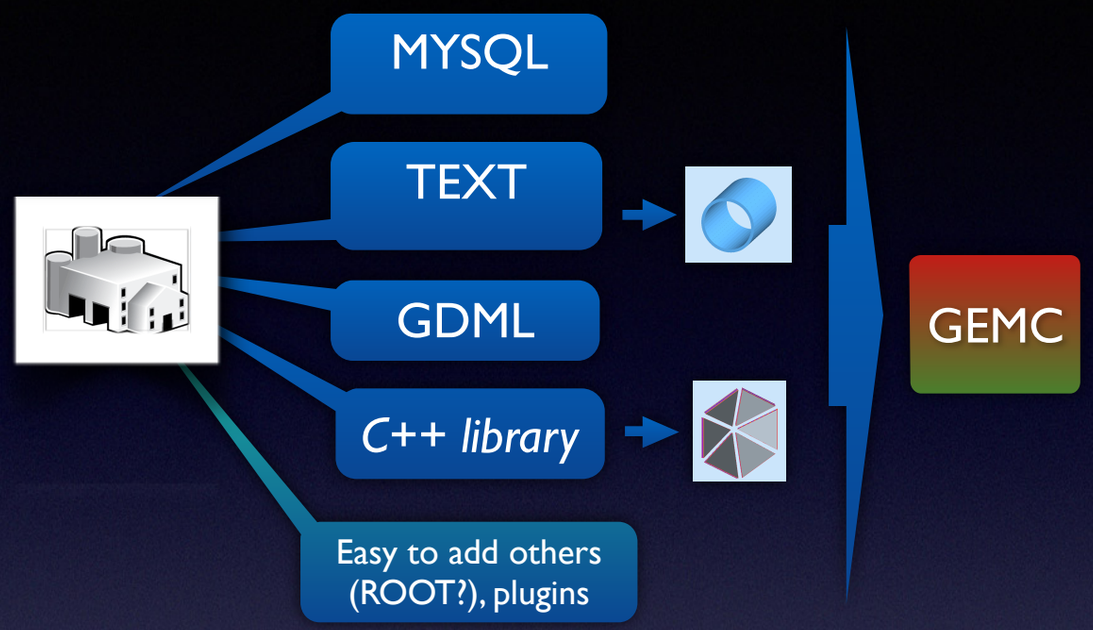
can have EC from a factory, ctof from another.

hit processes factories



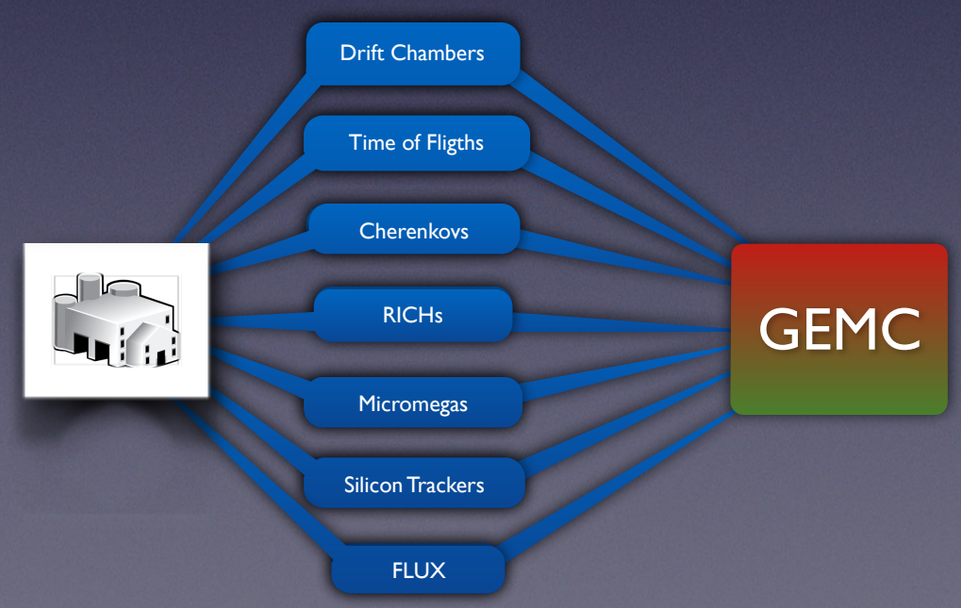
association of c++ algorithm and detector performed at run time

gemc factories



Working?

- MYSQL, TEXT, C++: <u>awesome</u>
- GDML: beta
- Wanted: ROOT



Working?

- Mechanism: yes but can be optimized (string comparison maybe expensive)
- Plugin: NOT CODED YET. Routines still compiled with gemc core (ouch).

output, bank structure

```
> BST  100, 0

  > True Step by Step infos   (101, 0)
    - Edep                    (101, 1)
    - Pid                     (101, 2)
    - positions               (101, 3)

  > Dgtz Step by Step infos   (102, 0)
    - ADCL                    (102, 1)
    - ADCR                    (102, 2)

  > True Integrated infos     (103, 0)
    - Edep                    (103, 1)
    - Pid                     (103, 2)
    - positions               (103, 3)

  > Dgtz Integrated infos     (104, 0)
    - ADCL                    (104, 1)
    - ADCR                    (104, 2)

  > Voltage as a function of time (105, 0)
    - Identifier                (105, 1)
    - Time                      (105, 2)
    - Voltage                   (105, 3)

> Trigger Bank                (106, 0)
    - Identifier              (106, 1)
    - Time                    (106, 2)
    - Voltage                 (106, 3)
```

```
"pid",     "Ri", "ID of the first particle entering the sensitive volume");
"mpid",    "Ri", "ID of the mother of the first particle entering the sensitive
"tid",     "Ri", "Track ID of the first particle entering the sensitive volume"
"mtid",    "Ri", "Track ID of the mother of the first particle entering the sen
"otid",    "Ri", "Track ID of the original track that generated the first parti
"trackE",  "Rd", "Energy of the track");
"totEdep", "Rd", "Total Energy Deposited");
"avg_x",   "Rd", "Average X position in global reference system");
"avg_y",   "Rd", "Average Y position in global reference system");
"avg_z",   "Rd", "Average Z position in global reference system");
"avg_lx",  "Rd", "Average X position in local reference system");
"avg_ly",  "Rd", "Average Y position in local reference system");
"avg_lz",  "Rd", "Average Z position in local reference system");
"px",      "Rd", "x component of momentum of the particle entering the sensitiv
"py",      "Rd", "y component of momentum of the particle entering the sensitiv
"pz",      "Rd", "z component of momentum of the particle entering the sensitiv
“vx",      "Rd", "x component of primary vertex of the particle entering the se
"vy",      "Rd", "y component of primary vertex of the particle entering the se
"vz",      "Rd", "z component of primary vertex of the particle entering the se
"mvx",     "Rd", "x component of primary vertex of the mother of the particle e
"mvy",     "Rd", "y component of primary vertex of the mother of the particle e
"mvz",     "Rd", "z component of primary vertex of the mother of the particle e
"avg_t",   "Rd", "Average time");
"hitn",    "Ri", "Hit Number");
```

sector
SuperLayer
Layer
wire
LR
Doca
SDoca
time
Stime

output, bank structure

```
> BST  100, 0

  > True Step by Step infos  (101, 0)
    - Edep                   (101, 1)
    - Pid                    (101, 2)
    - positions              (101, 3)

  > Dgtz Step by Step infos  (102, 0)
    - ADCL                   (102, 1)
    - ADCR                   (102, 2)

  > True Integrated infos    (103, 0)
    - Edep                   (103, 1)
    - Pid                    (103, 2)
    - positions              (103, 3)

  > Dgtz Integrated infos    (104, 0)
    - ADCL                   (104, 1)
    - ADCR                   (104, 2)

  > Voltage as a function of time (105, 0)
    - Identifier                 (105, 1)
    - Time                       (105, 2)
    - Voltage                    (105, 3)

> Trigger Bank               (106, 0)
    - Identifier             (106, 1)
    - Time                   (106, 2)
    - Voltage                (106, 3)
```

```
> Header  10, 0

    - Event number
    - Timestap
    - Event Type
    […]
```

```
> Generated Particles

    - pid
    - p
    - v
    - vector<Det. Summary>
      ▸ EC,  22 hits, 3 GeV
      ▸ DC,  29 hits, 0.6 MeV
      ▸ HTCC, 2 hits, 23 nphe
```

Fast MC Mode

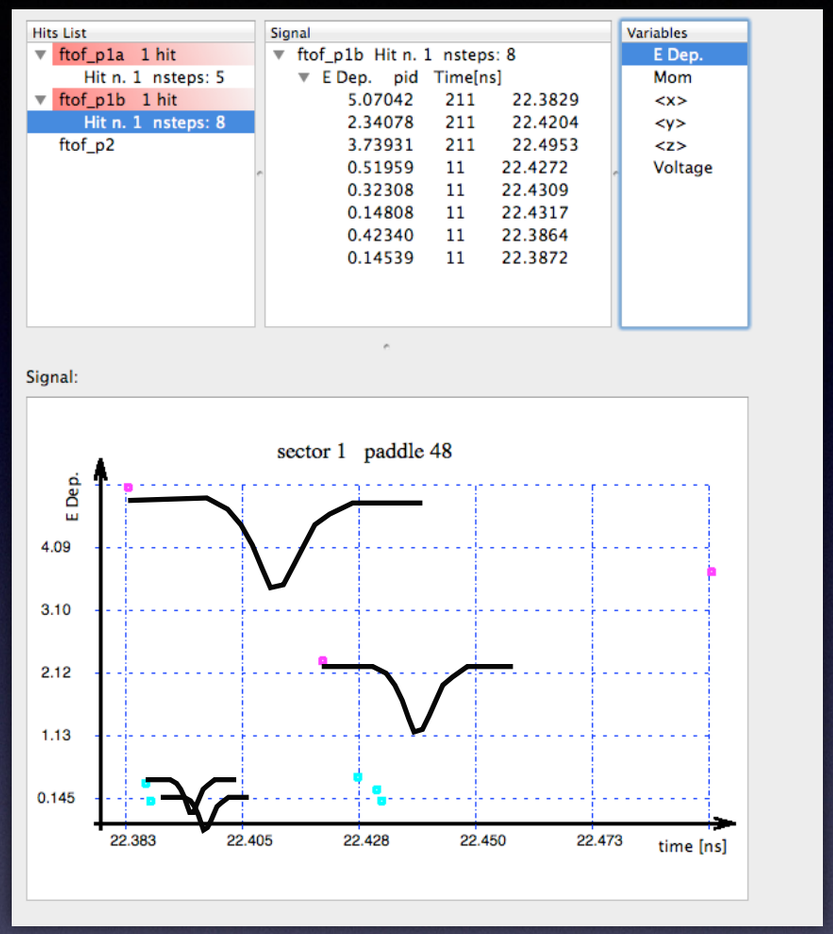
- ENERGY_CUT
- “fast MC routines”

output, bank structure

Working?

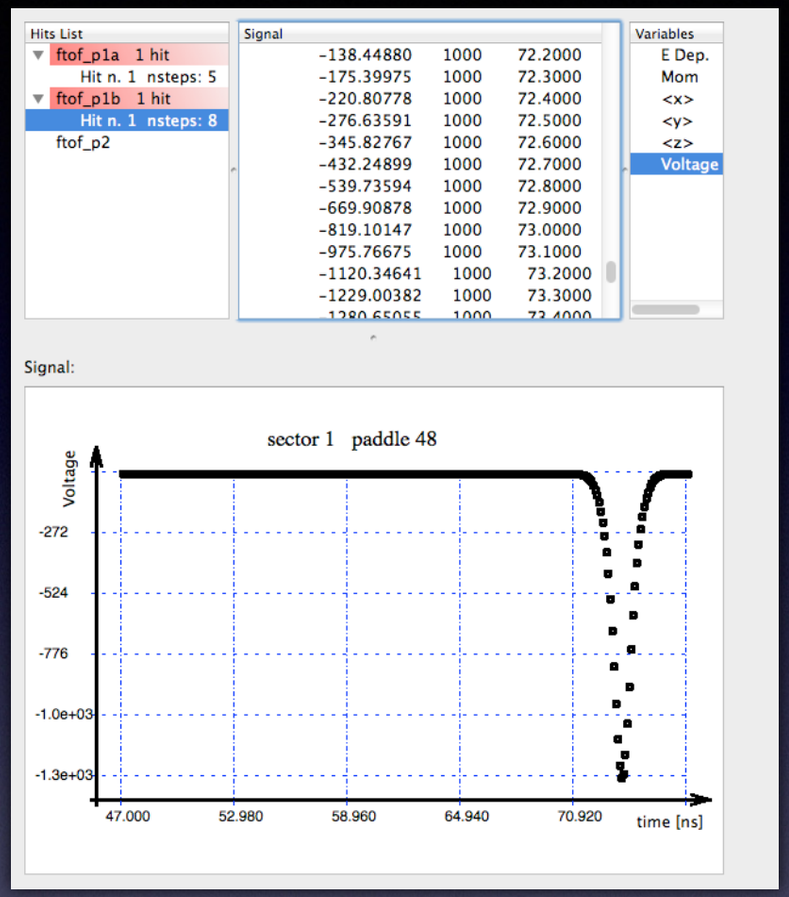
- awesome. Allows for integrated over electronic time window (pile ups) or step by step output for external digitization
- Plugin: not coded yet but not crucial.
- Wanted: ROOT output plugin.

Voltage Signal



Rise time: 1ns
Fall time: 2ns
Delay: 50ns
1 MeV = 100 mV

Total Signal is integral (over electronic time window) of all the step-signals.
Signal processing time is small.



Working: YES, but parameters are common for all channels. May need to introduce channel-by-channel pars

Magnetic Fields

```
<mfield>
        <description name="clas12-torus" factory="ASCII" comment="clas12 superconducting torus"/>
        <symmetry type="phi-segmented" format="map" integration="RungeKutta" minStep="1*mm"/>
        <map>
                <coordinate>
                        <first  name="azimuthal"     npoints="61"   min="0"   max="30"  units="deg"/>
                        <second name="transverse"    npoints="126"  min="0"   max="500" units="cm"/>
                        <third  name="longitudinal"  npoints="126"  min="100" max="600" units="cm"/>
                </coordinate>
        <field unit="kilogauss"/>
        <interpolation type="none"/>
        </map>
</mfield>
0       0       100     0       0       0
0       0       104     0       0       0
```

mapped, constant fields

```
<mfield>
      <description name="meic_ele_downstream_quadrupole4" factory="ASCII"
        comment="meic_det1_ele_downstream_quadrupole4"/>

      <symmetry type="multipole" format="simple" integration="RungeKutta" minStep="1*mm"/>

      <dimension Npole="4" scale="23.9743818" Bunit="T" x="0" y="0" z="-4.30"
        XYZunit="m" rot="0" ROTunit="rad" ROTaxis="Y"/>

</mfield>
```

Dipole, Quadrupoles, Multipoles

- Working: yes
- Needed: superimposing fields
- Wanted: better, more general description (useful for reconstruction as well)

Optical Properties

- **surface**
- **type**
- **optical properties:**
  - photonEnergy
  - indexOfRefraction
  - reflectivity
  - efficiency
  - specularlobe
  - specularspike
  - backscatter

```
            polished:  smooth perfectly polished surface
polishedfrontpainted:  smooth top-layer (front) paint
 polishedbackpainted:  same is 'polished' but with a back-paint

              ground:  rough surface
  groundfrontpainted:  rough top-layer (front) paint
   groundbackpainted:  same as 'ground' but with a back-paint

 polishedlumirrorair:  mechanically polished surface, with lumirror
polishedlumirrorglue:  mechanically polished surface, with lumirror & meltmount
         polishedair:  mechanically polished surface
   polishedteflonair:  mechanically polished surface, with teflon
      polishedtioair:  mechanically polished surface, with tio paint
    polishedtyvekair:  mechanically polished surface, with tyvek
   polishedvm2000air:  mechanically polished surface, with esr film
  polishedvm2000glue:  mechanically polished surface, with esr film & meltmount

   etchedlumirrorair:  chemically etched surface, with lumirror
  etchedlumirrorglue:  chemically etched surface, with lumirror & meltmount
           etchedair:  chemically etched surface
     etchedteflonair:  chemically etched surface, with teflon
        etchedtioair:  chemically etched surface, with tio paint
      etchedtyvekair:  chemically etched surface, with tyvek
     etchedvm2000air:  chemically etched surface, with esr film
    etchedvm2000glue:  chemically etched surface, with esr film & meltmount

   groundlumirrorair:  rough-cut surface, with lumirror
  groundlumirrorglue:  rough-cut surface, with lumirror & meltmount
           groundair:  rough-cut surface
     groundteflonair:  rough-cut surface, with teflon
        groundtioair:  rough-cut surface, with tio paint
      groundtyvekair:  rough-cut surface, with tyvek
     groundvm2000air:  rough-cut surface, with esr film
    groundvm2000glue:  rough-cut surface, with esr film & meltmount
```

Optical Properties

- surface
- type
- optical properties:
  - photonEnergy
  - indexOfRefraction
  - reflectivity
  - efficiency
  - specularlobe
  - specularspike
  - backscatter

```
     dielectric_metal:  dielectric-metal interface
dielectric_dielectric:  dielectric-dielectric interface
       dielectric_LUT:  dielectric-Look-Up-Table interface
```

Optical Properties

- **surface**
- **type**
- **optical properties:**
  - photonEnergy
  - indexOfRefraction
  - reflectivity
  - efficiency
  - specularlobe
  - specularspike
  - backscatter

```
Table of optical photon energies (wavelengths) from 190-650 nm:

my $penergy =
"  1.9074494*eV  1.9372533*eV  1.9680033*eV  1.9997453*eV  2.0325280*eV " .
"  2.0664035*eV  2.1014273*eV  2.1376588*eV  2.1751616*eV  2.2140038*eV " .
"  2.2542584*eV  2.2960039*eV  2.3393247*eV  2.3843117*eV  2.4310630*eV " .
"  2.4796842*eV  2.5302900*eV  2.5830044*eV  2.6379619*eV  2.6953089*eV " .
"  2.7552047*eV  2.8178230*eV  2.8833537*eV  2.9520050*eV  3.0240051*eV " .
"  3.0996053*eV  3.1790823*eV  3.2627424*eV  3.3509246*eV  3.4440059*eV " .
"  3.5424060*eV  3.6465944*eV  3.7570973*eV  3.8745066*eV  3.9994907*eV " .
"  4.1328070*eV  4.2753176*eV  4.4280075*eV  4.5920078*eV  4.7686235*eV " .
"  4.9593684*eV  5.1660088*eV  5.3906179*eV  5.6356459*eV  5.9040100*eV " .
"  6.1992105*eV  ";

Reflectivity of AlMgF2 coated on thermally shaped acrylic sheets, measured by AJRP,
10/01/2012:
my $reflectivity =
"  0.8331038     0.8309071     0.8279127     0.8280742     0.8322623 " .
"  0.837572      0.8396875     0.8481834     0.8660284     0.8611336 " .
"  0.8566167     0.8667431     0.86955       0.8722481     0.8728122 " .
"  0.8771635     0.879907      0.879761      0.8831943     0.8894673 " .
"  0.8984234     0.9009531     0.8910166     0.8887382     0.8869093 " .
"  0.8941976     0.8948479     0.8877356     0.9026919     0.8999685 " .
"  0.9101617     0.8983005     0.8991694     0.8990987     0.9000493 " .
"  0.9065833     0.9028855     0.8985184     0.9009736     0.9086968 " .
"  0.9015145     0.8914838     0.8816829     0.8666895     0.8496298 " .
"  0.9042583 ";
```

Optical Properties

Working?

- awesome.
- geant4 propagation of optical photons not perfect

gemc collaboration (gemc.github.io)

How to contribute

Feel free to ask

So let's say that you have an idea for a great feature. It's a good idea to open an issue describing the feature and its implementation and ask the code author's opinion. If they agree, go for it! They might even have some good suggestions for changes or additions to the feature as well.

If it's a bug you found, occasionally it can be ok to just create a **pull request** (PR), as long as it's clearly a bug with a straightforward fix, but it's also not a bad idea to file the bug as an issue first.

Finally, if you want to contribute but are not sure where, you can ask the author if they need help with anything – it could be as simple as helping improve the documentation.

Forking the repo

Ok, so we have a great feature idea (or we found a bug), we opened an issue to check with the author, and they signed off on it. Whoo! Time to get to coding. First thing you do is create a fork, that is a copy of the main repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Forking a repository is a simple two steps process:

1. On GitHub, navigate to the gemc repository.
2. In the top-right corner of the page, click Fork.

You now have a copy of the repo you just forked, available in your GitHub account; its **fork-url-address** can be found on the right menu.

You can create a pull request based on this fork. If you are working on several new features at once, you can create a branch for each feature.

Code Standards

When writing both commits and code, it's important to do so in harmony with a project's existing style. If the project uses camelCase variable naming, this is how you should name your variables as well. If the project has a test suite, you should be writing tests for any changes you make.

Even if you don't agree with some of the author's stylistic decisions, you should adhere to them in your PR. If you have a solid reason why they should be changed, open up an issue and discuss it there. Never ever change an author's existing code style to something you prefer, this is in extremely poor taste.

Create a Pull Request

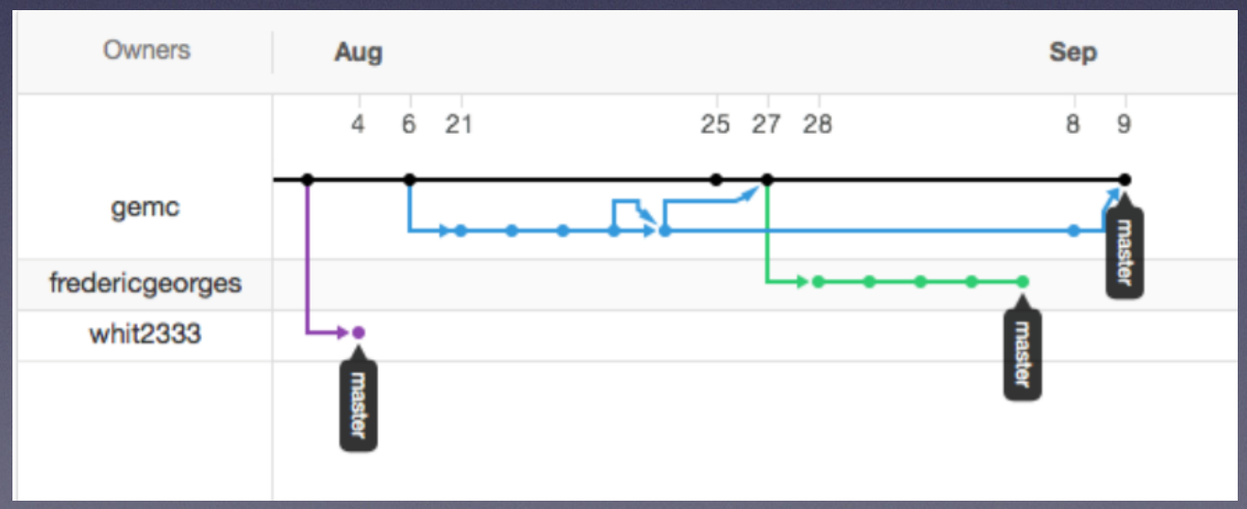
To create the pull request, navigate in github to your fork, and click on the PR button:

You will be presented with a page with a summary of your changes. Once you're ready, go ahead and press the PR button to provide additional informations:

- Make sure you selected the correct branch name ("master" if it's the main fork)
- Make sure the title and description are clear and concise
- If the change is visual, make sure to include a screenshot or gif
- If the PR closes an issue, make sure to put Closes #X at the end of the

description on a newline

1. Create an "issue"
2. Fork
3. Modify
4. Pull request



source: https://github.com/gemc

Outlook

Projects:

Overlapping Fields

Multichannel ADC/TDC

C++11

G4 Multithreading

Code optimisation

ROOT detector factory

ROOT output plugin

Improvements:

API for detector

Common generator library

Hit process should be plugins

Field superposition