Progress on generic biasing

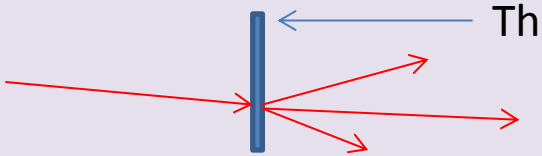
Marc Verderi
LLR – Ecole polytechnique

Geant4 Collaboration Meeting
Seville
September 2013

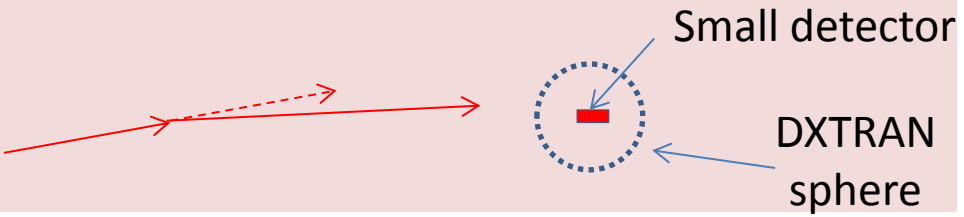
INTRODUCTION

Introduction / reminder (1/2)

- Geant4 proposes biasing options
  - Geometrical importance sampling, Leading particle biasing, Radioactive decay biasing, G4WrapperProcess, Reverse MC
- But misses others
  - Exponential transform: $p(\ell) = \sigma \cdot e^{-\sigma\ell} \rightarrow p'(\ell) = \sigma' \cdot e^{-\sigma'\ell}$
    - Change total cross-section
    - Make change direction dependent
  - forced interaction:
    - Force interaction in thin volume



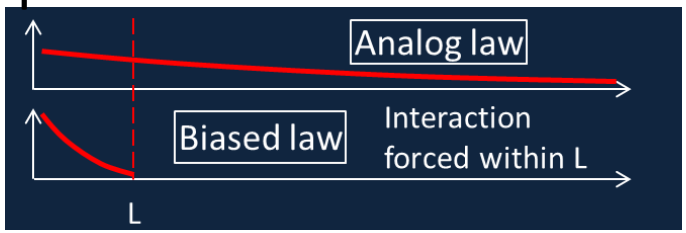
  - forced flight (towards detector)
    - So called DXTRAN
    - Force scattering towards detector



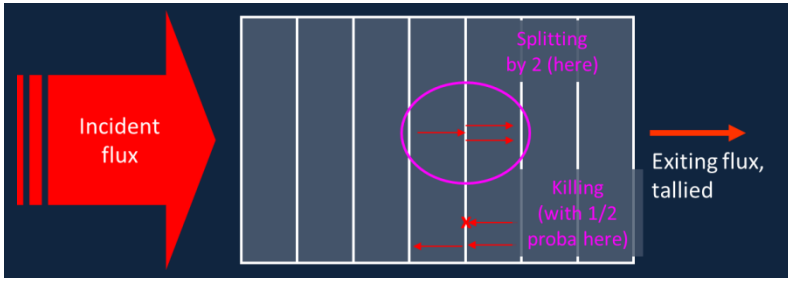
  - etc.
- These options implies ***changing the behavior*** of either
  - the physics process interaction probability (PostStep GPIL)
  - the physics process generation of the final state (PostStep DoIt)
  - Or both

Reminder (2/2)

- We rely on the two main biasing techniques:
  - **Importance sampling**
    - Change of (analog) probabilities (ie : GPIL, Final State)



  - **Splitting (killing)**
    - Split(kill) particles when moving towards the (un)desired direction of phase space.



- For biasing of GPIL (importance sampling) we reviewed the formalism, inferred how to fit it in Geant4, and came to:
  - The sampling of the interaction law should be made according to the biased law in the PostStepGPIL, while remembering the analog cross-section for further weight calculation
  - At each step, and for each biased process, a non-interaction weight has to be applied.
  - When a step ends with an interaction produced by one process, an interaction weight has to be applied too.

DESIGN APPROACH

Approach of design

- FLUKA and MCNPX control biasing through “data cards”
  - Advantage of being robust
  - But a new functionality requires a new development of the software

- Try to follow a “toolkit” approach, taking advantage of OO technology
  - Try to model the problem in term of a few abstract classes
    - that provide the interfaces to the Geant4 kernel
  - And let the concrete cases be implemented inheriting from these abstract classes
    - Eg : splitting, killing, forced collision, brem. splitting, etc.
  - Then Geant4 can provide some usual implementations of concrete cases
    - That can be controlled with “command line”, in a “data cards”- style
  - But users are free to extend the toolkit with their own dedicated biasing techniques
    - If none of the options provided respond to their need
    - Without the need (hopefully) of modifying the Geant4 kernel

- Some thinking on a real case…

A real example, from MCNP

Neutron shielding problem:

1) Neutron source
   - Force emission direction
2) Cross concrete slabs
   - Force penetration
3) Travel up to top
4) Make tally in cell flux (F4)
   - Force interaction for scoring
5) Tally in detector off-axis (F5)
   - Force scattering towards detector

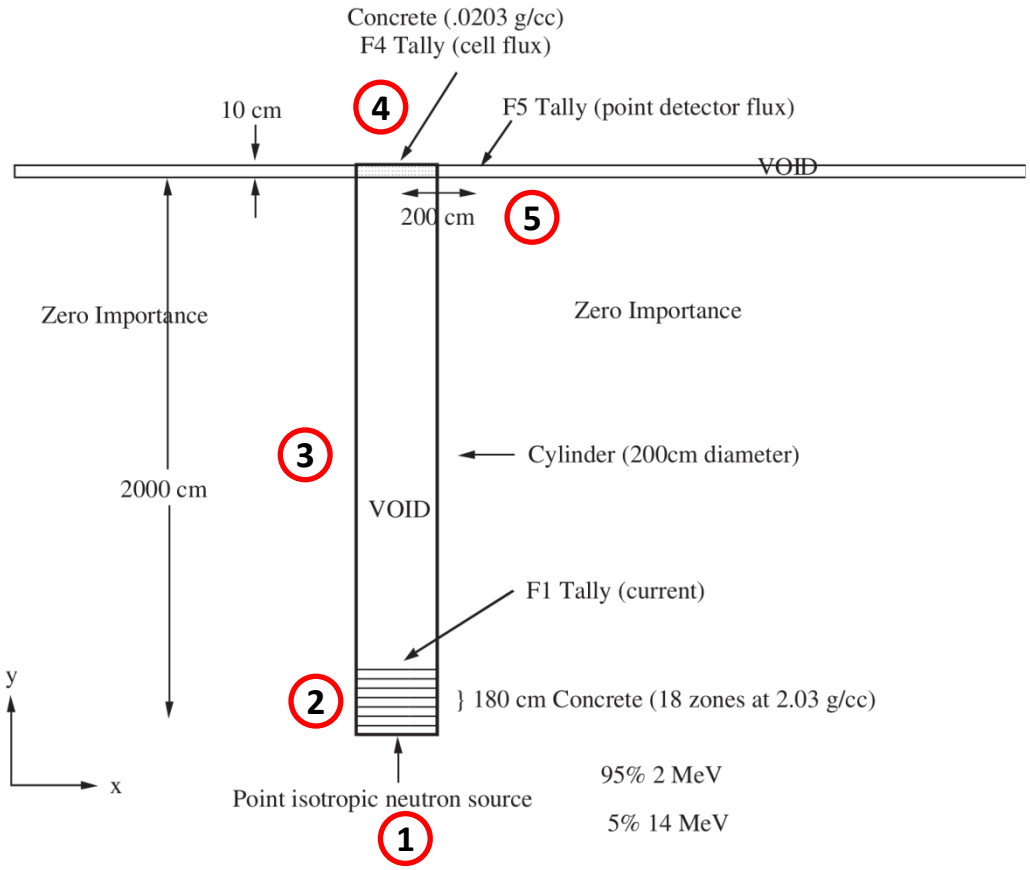


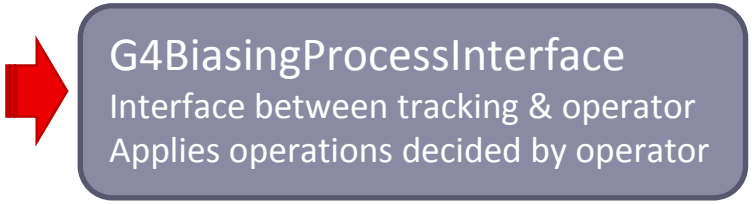
Figure 10.1 Transport problem.

Observations

- Many techniques…
- Several techniques are used, and are used in non-trivial sequences
  - Decision making for applying these techniques needs lot of freedom
  - Decision to be made on volume, particle energy, position, etc. on a step by step basis
- Physics processes have to behave differently depending on volume or set of volumes
  - Change in their interaction probability
  - Change in their final state production
- Non-physics (splitting / killing) applies as well
- What do we identify ?
  - Biasing operations:
    - Splitting or killing
    - or change in interaction probability
    - or a change in final state production
  - A decision making entity, to “pilot” the biasing operations
    - A “biasing operator”, consulted step by step for decision
  - Need to control the behavior of physics processes
    - In their interaction probability
    - In their final state production

Observations

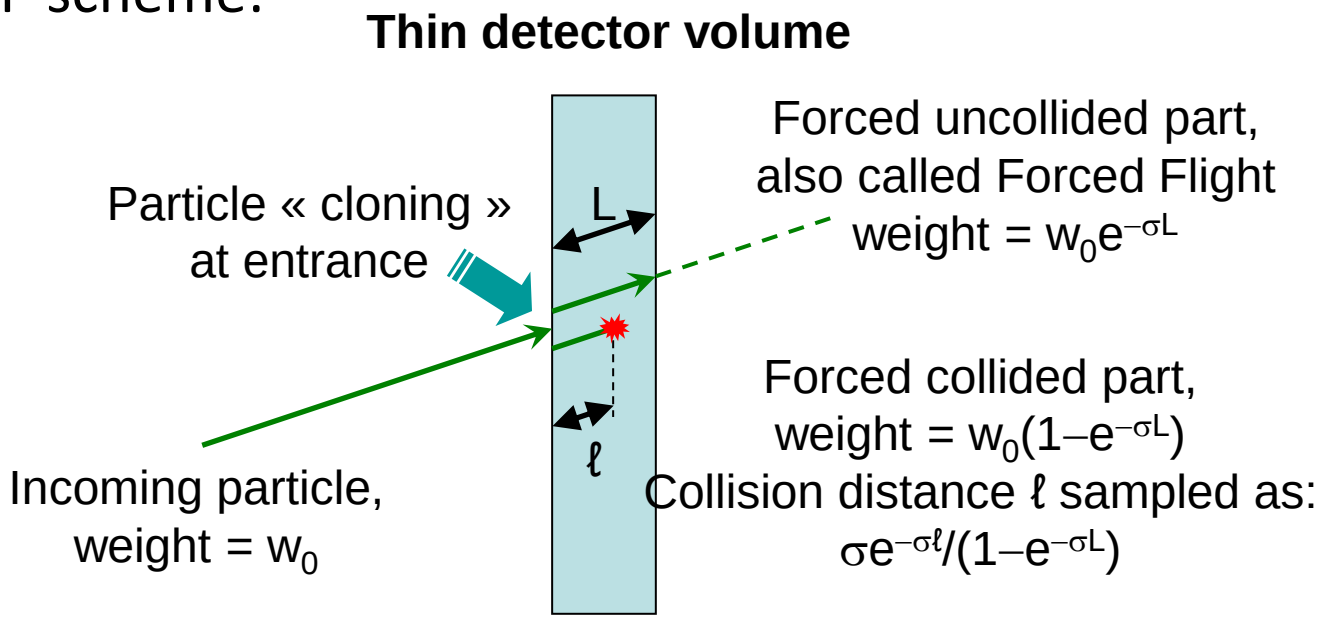
- Many techniques…
- Several techniques are used, and are used in non-trivial sequences
  - Decision making for applying these techniques needs lot of freedom
  - Decision to be made on volume, particle energy, position, etc. on a step by step basis
- Physics processes have to behave differently depending on volume or set of volumes
  - Change in their interaction probability
  - Change in their final state production
- Non-physics (splitting / killing) applies as well
- What do we identify ?
  - Biasing operations:
    - Splitting or killing
    - or change in interaction probability
    - or a change in final state production
  - A decision making entity, to “pilot” the biasing operations
    - A “biasing operator”, consulted step by step for decision
  - Need to control the behavior of physics processes
    - In their interaction probability
    - In their final state production



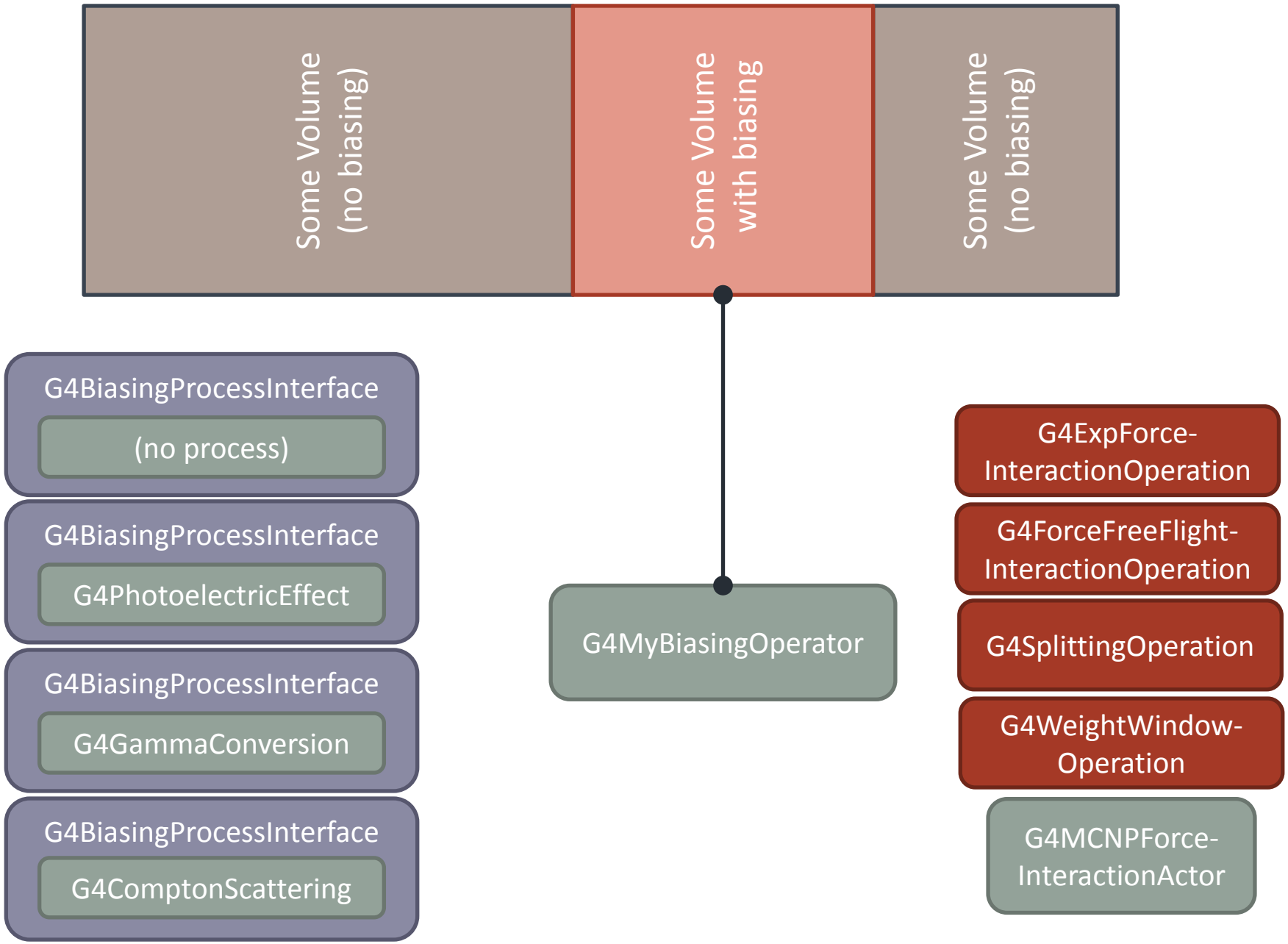
IMPLEMENTATION OF AN MCNP-LIKE FORCE COLLISION SCHEME

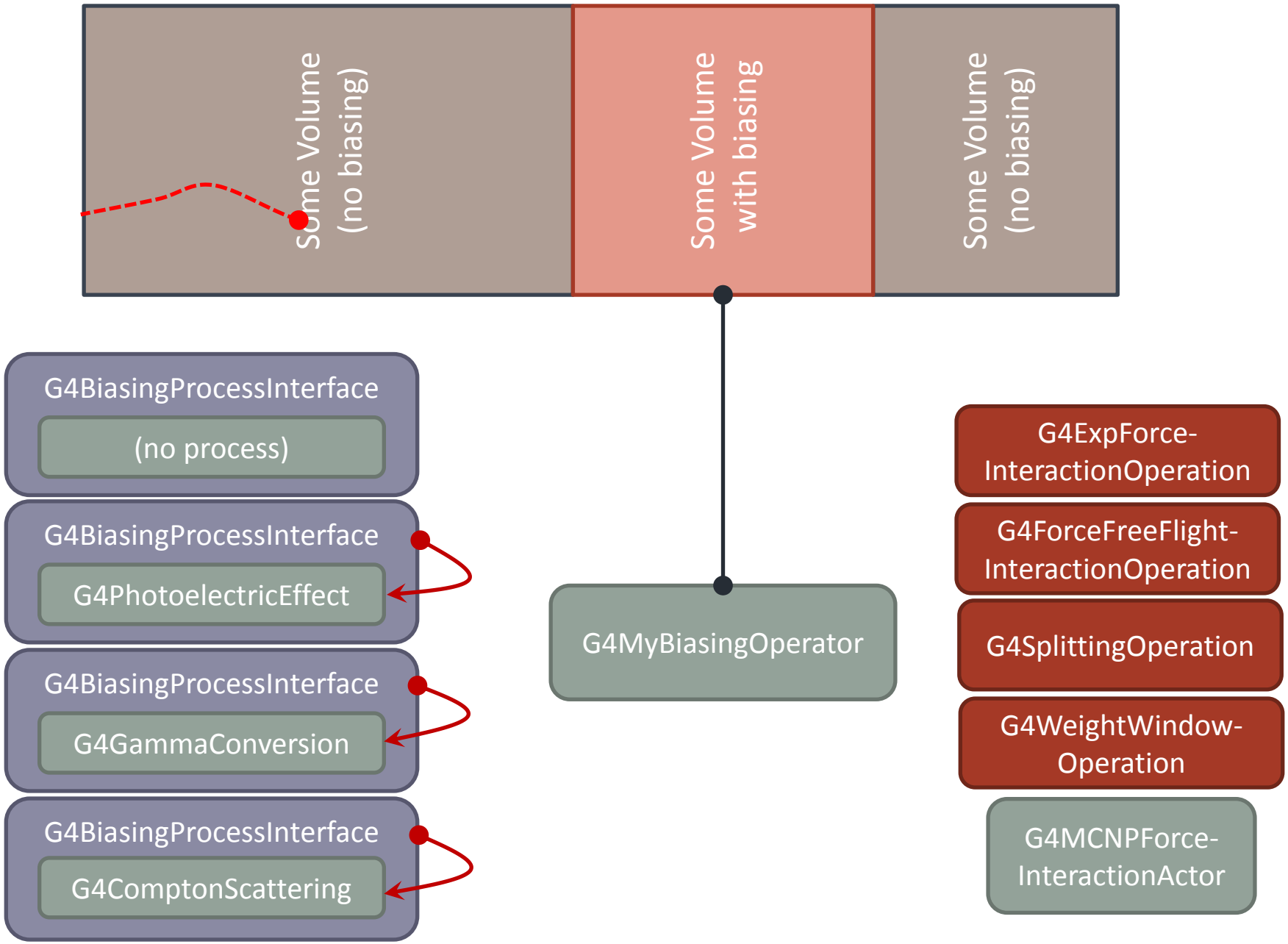
Emulation of an MCNP-like “force collision”

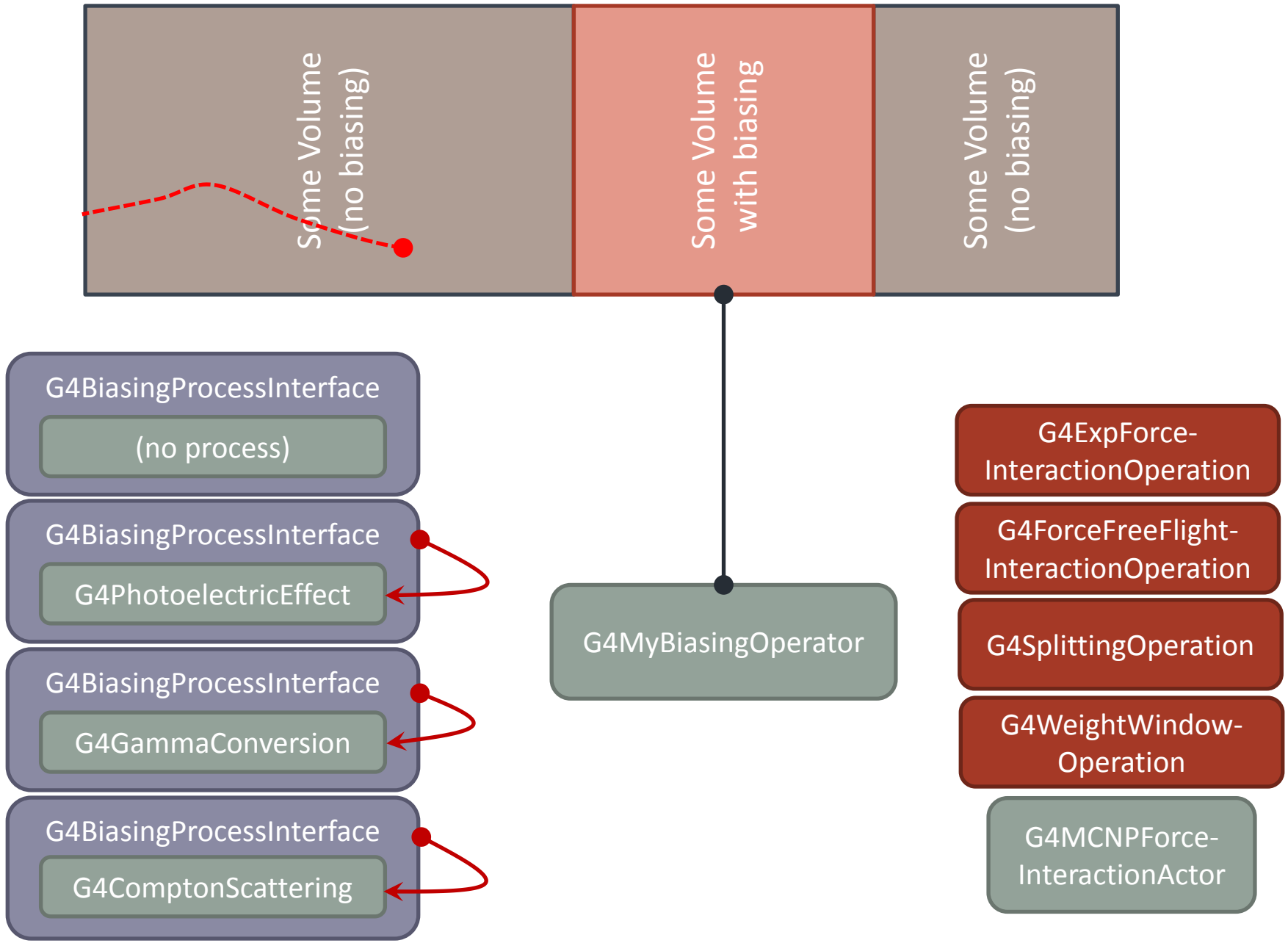
- “Exercise” to challenge proposed design
  - Can it satisfy this existing / popular option ?
- MCNP scheme:

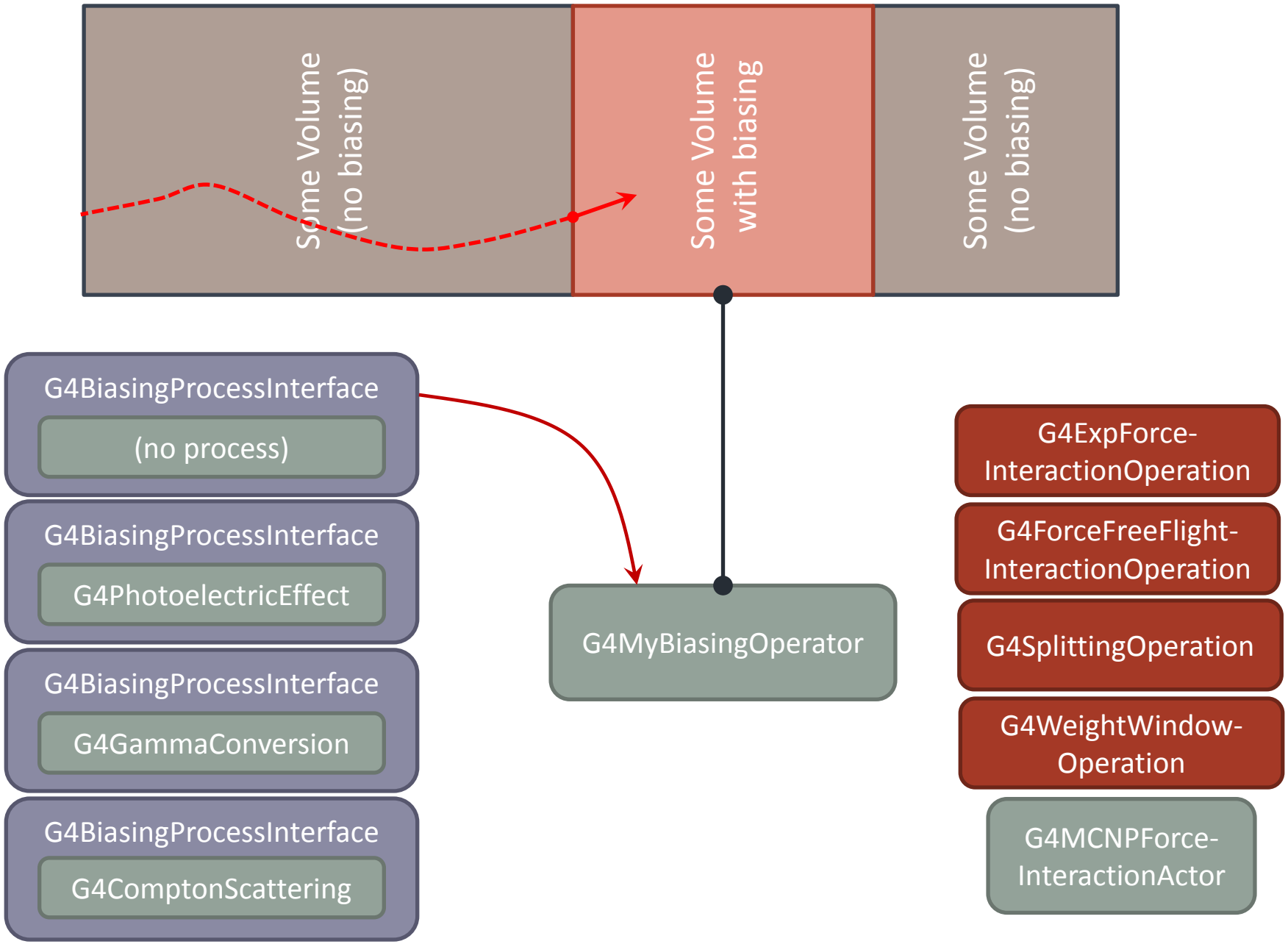


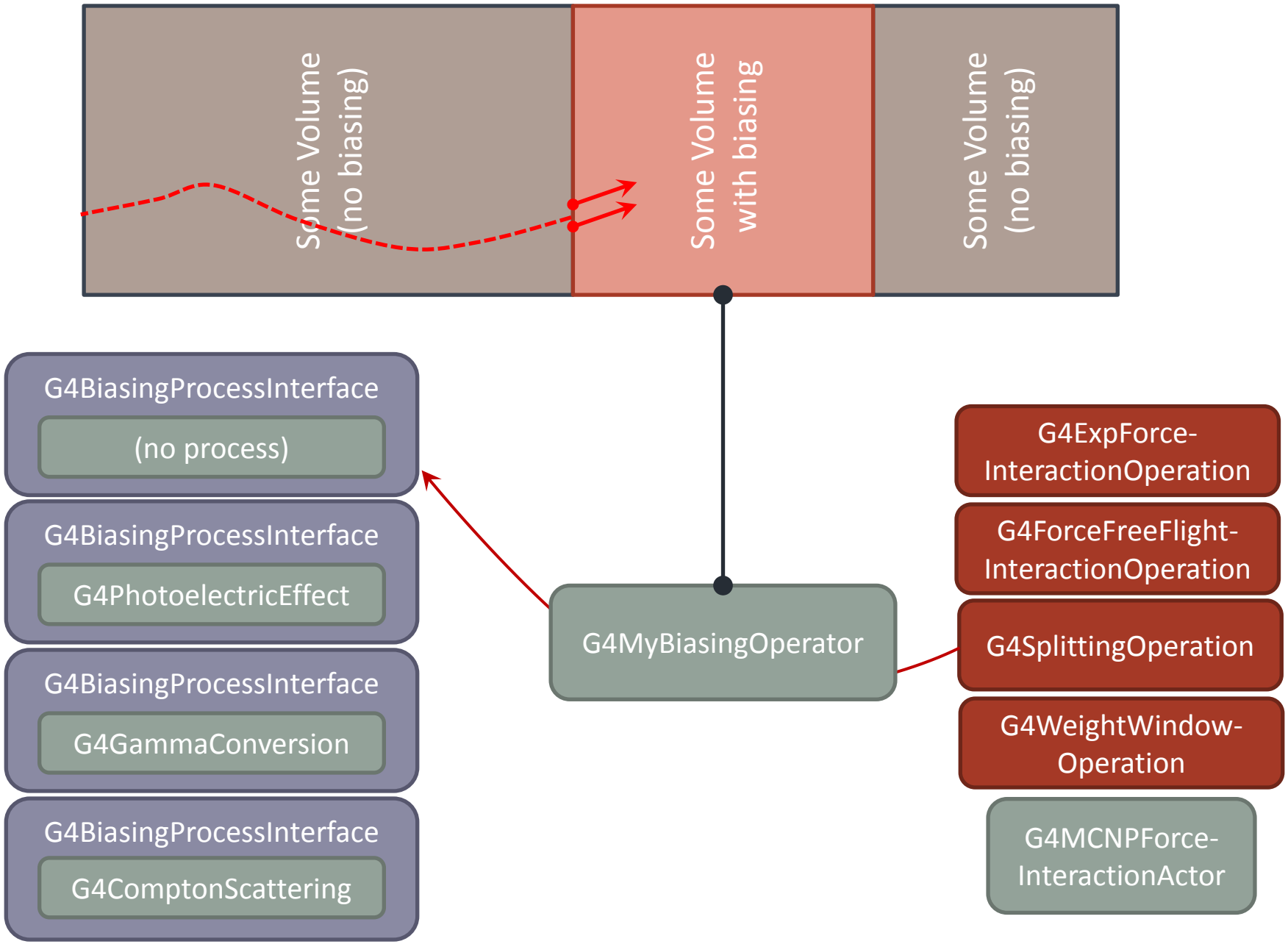
  - Involves : forcing a free flight, forcing an interaction – both with related weight computations, relying on *our* formalism- , doing that using the *total* cross-section –ie making operations to cooperate- : a pretty challenging exercise.
- Coming cartoon illustrates the way it works

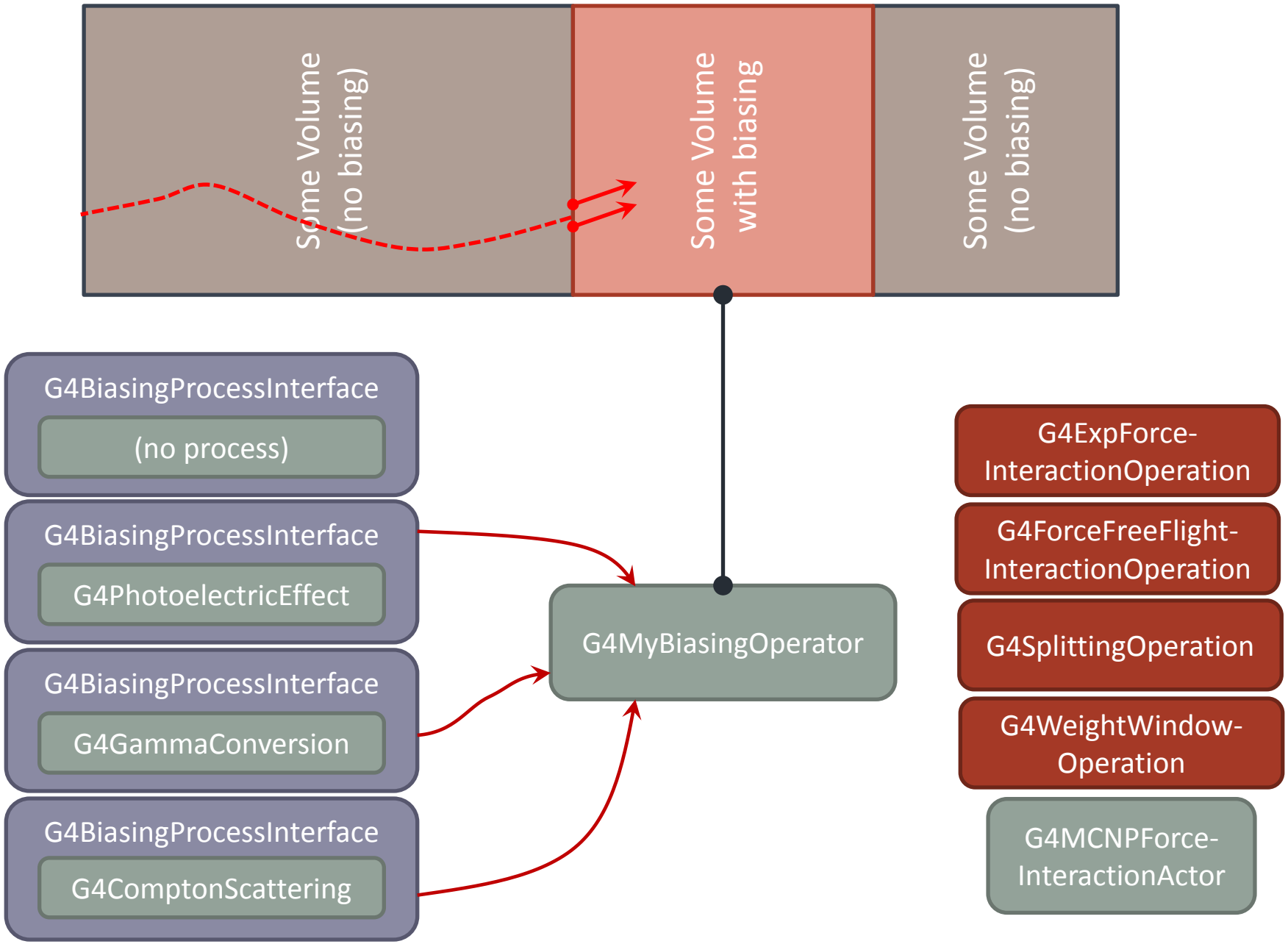

Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor

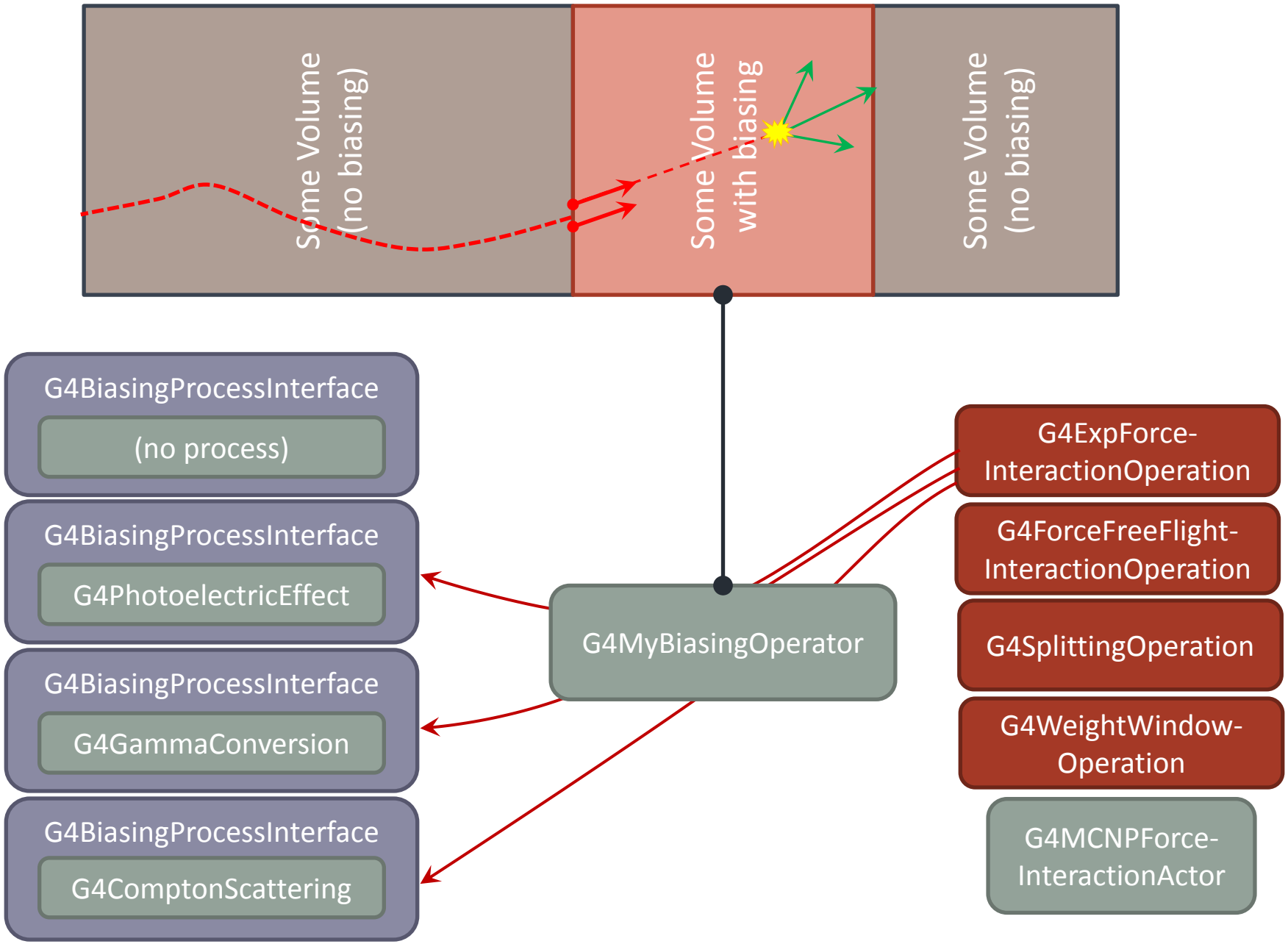

Some Volume
(no biasing)
Some Volume
with biasing
Some Volume
(no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-
InteractionOperation
G4ForceFreeFlight-
InteractionOperation
G4SplittingOperation
G4WeightWindow-
Operation
G4MCNPForce-
InteractionActor

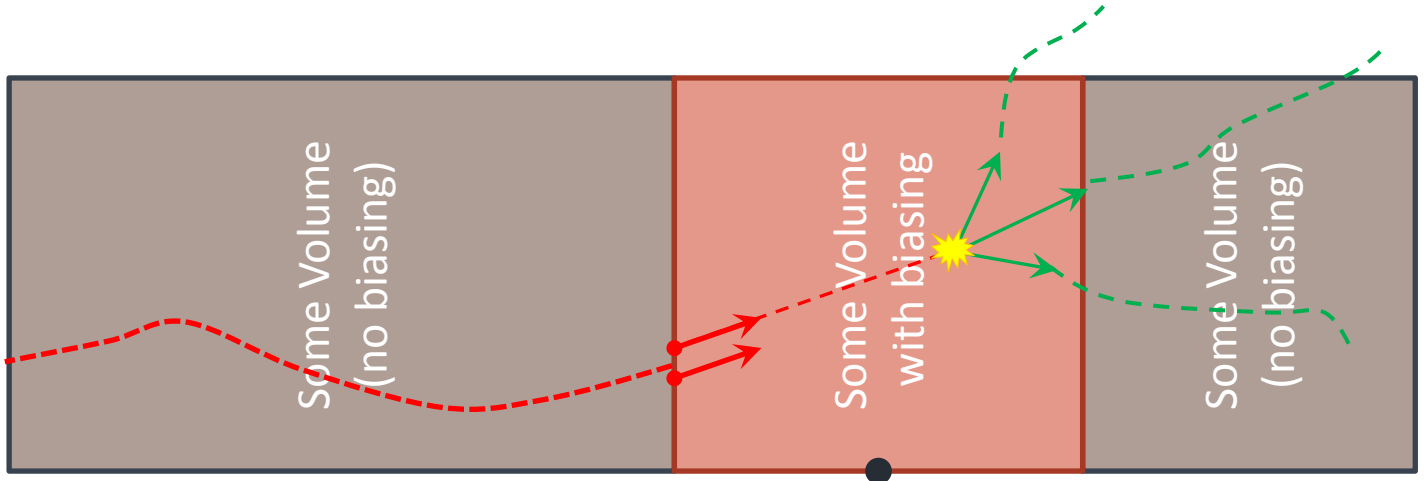

Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor


Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor


Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor


Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor


Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor



G4BiasingProcessInterface
(no process)

G4BiasingProcessInterface
G4PhotoelectricEffect

G4BiasingProcessInterface
G4GammaConversion

G4BiasingProcessInterface
G4ComptonScattering

G4MyBiasingOperator

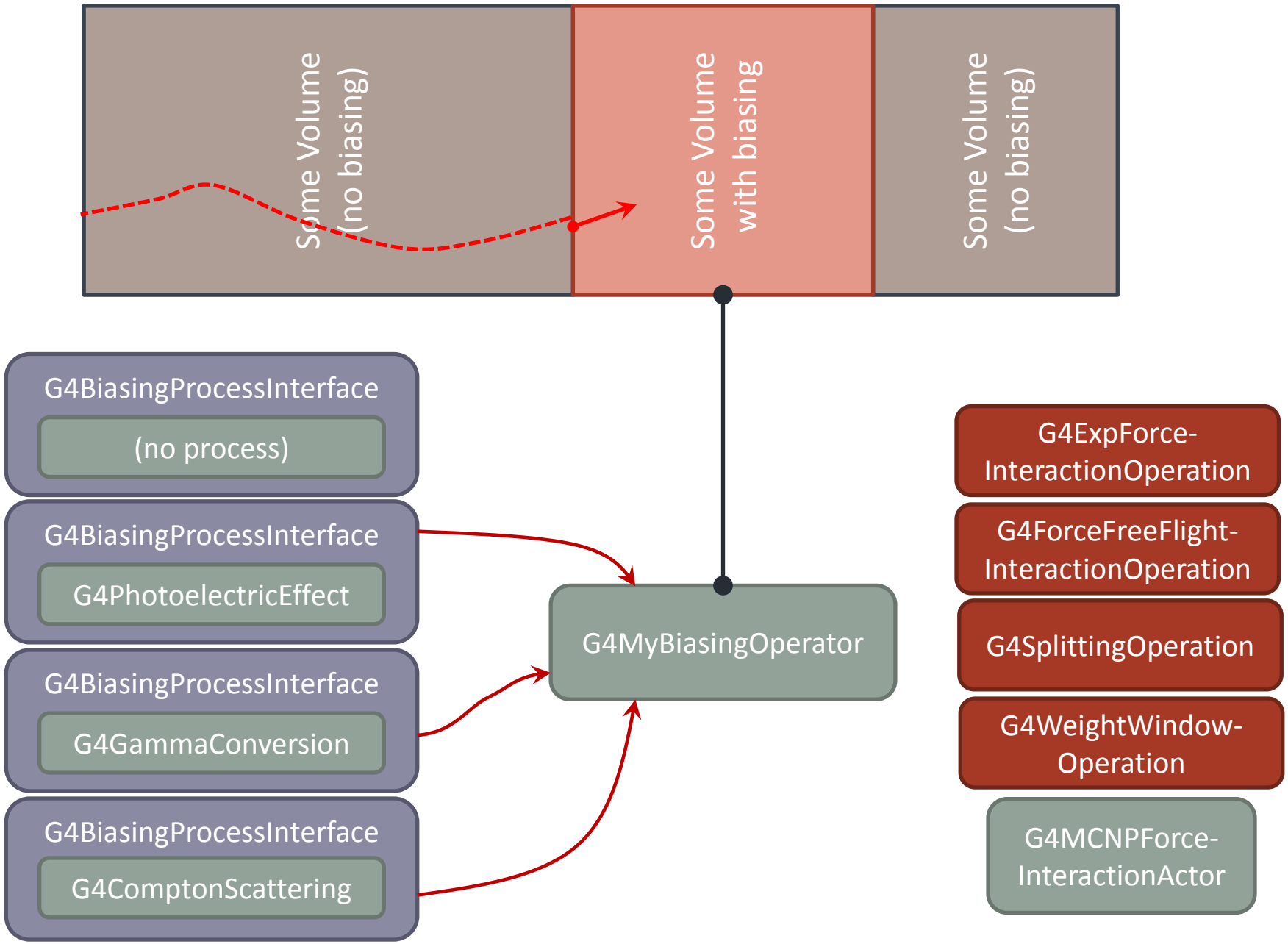
G4ExpForce-
InteractionOperation

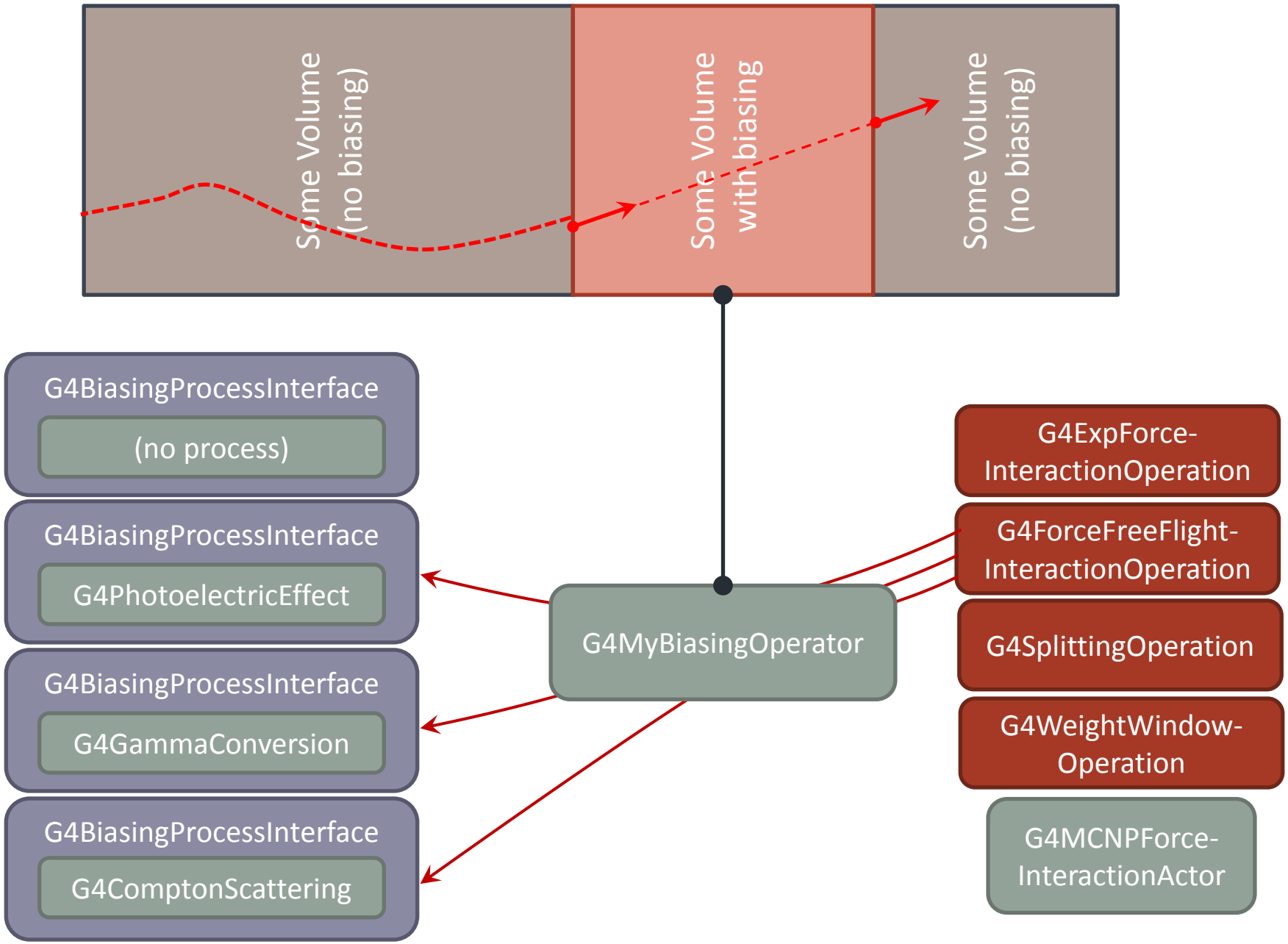
G4ForceFreeFlight-
InteractionOperation

G4SplittingOperation

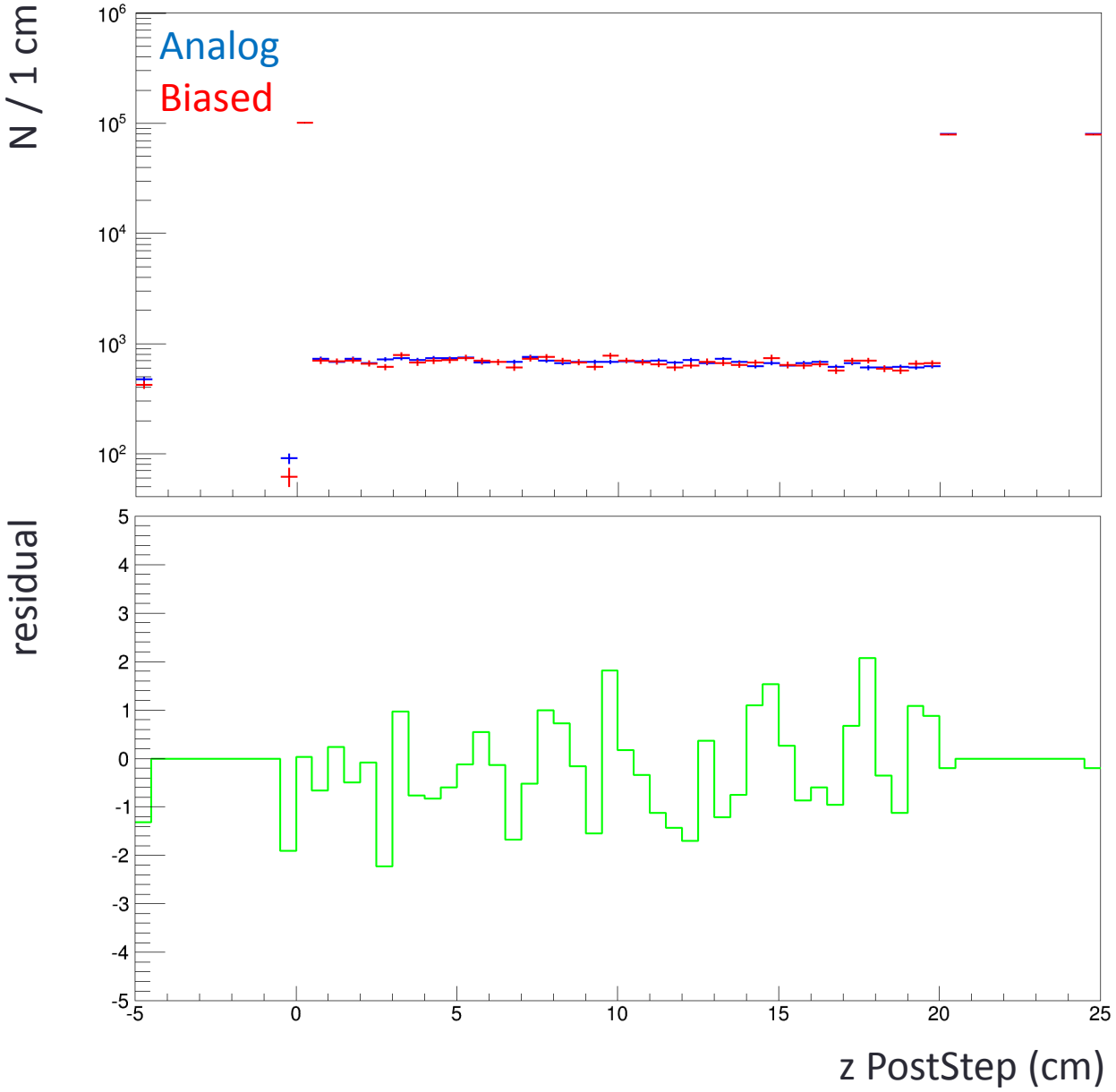
G4WeightWindow-
Operation

G4MCNPForce-
InteractionActor

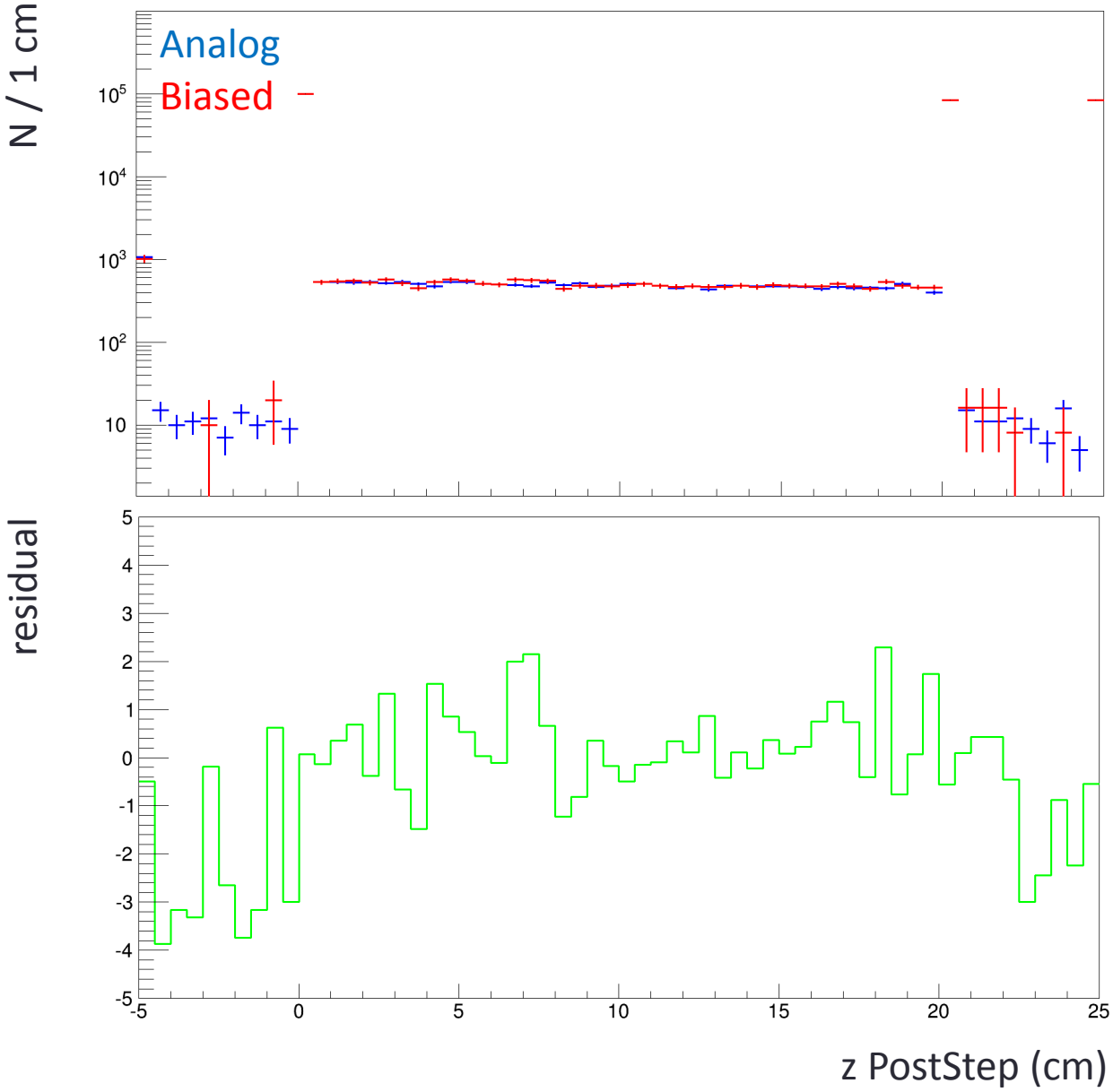

Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor


Some Volume (no biasing)
Some Volume with biasing
Some Volume (no biasing)
G4BiasingProcessInterface
(no process)
G4BiasingProcessInterface
G4PhotoelectricEffect
G4BiasingProcessInterface
G4GammaConversion
G4BiasingProcessInterface
G4ComptonScattering
G4MyBiasingOperator
G4ExpForce-InteractionOperation
G4ForceFreeFlight-InteractionOperation
G4SplittingOperation
G4WeightWindow-Operation
G4MCNPForce-InteractionActor

z PostStep for 100 MeV γ in a 20cm long liquid N_2



z PostStep for 1 GeV K^0_L in a 20cm long liquid N_2



HELPER TOOLS

Tools to help configuring physics lists (1/2)

- Approach is intrusive to physics lists
  - Have to make simple the modifications needed

- **Granular level: G4BiasingHelper**
  - Insert needed processes in the G4ProcessManager of a particle
  - For a physics process: substitute the process with a wrapped version
    - Adding or wrapping the AlongStep methods
    - Ie a PostStep process becomes a PostStep + AlongStep process
    - Caring about process ordering

  - Methods:

```
static  G4bool      ActivatePhysicsBiasing(G4ProcessManager* pmanager,  G4String processToBias,
                                           G4String wrappedName = "");
static  void    ActivateNonPhysicsBiasing(G4ProcessManager* pmanager,  G4String nonPhysProcessName = "");
```

  - Usage:
    - After a process manager has been setup, helper methods can be invoked:

```
G4BiasingHelper::ActivatePhysicsBiasing(pmanager, "compt");
G4BiasingHelper::ActivatePhysicsBiasing(pmanager, "conv");
G4BiasingHelper::ActivateNonPhysicsBiasing(pmanager);
```

Tools to help configuring physics lists (2/2)

- **Global level: G4BiasingPhysics**
  - A physics constructor to be used to modify an existing physics list
    - Makes use of the G4BiasingHelper
  - Example of methods:

```
void PhysicsBias(const G4String& particleName);
void NonPhysicsBias(const G4String& particleName);
void Bias(const G4String& particleName);
```

  - Usage:

```
FTFP_BERT* physList = new FTFP_BERT;
G4BiasingPhysics* biasPhys = new G4BiasingPhysics();
biasPhys->Bias("gamma");
biasPhys->Bias("neutron");
biasPhys->Bias("kaon0L");
physList->RegisterPhysics(biasPhys);
runManager->SetUserInitialization(physList);
```

WHAT’S NEXT

What is missing / needs discussions

- Have to decide where to release this code in the distribution
  - Not much a problem for what has been presented here
  - An issue to be discussed for what concerns final state biasing (see below [*])
  - And provide examples, tests, documentation
- Final state biasing not prototyped yet:
  - Brem. splitting and leading particle should be “easy”
    - And could be made rather generic
  - [*] More difficult is final state biasing in term of biasing distributions:
    - Biasing of energy/angular/secondaries (…) distributions
    - No generic “differential cross-section” concept
      - » Needed to compute the probability of a given topology
    - Will have to make direct calls to specific facilities in the physics packages
    - Make biasing dependent on other phys. Packages ? Or the reverse, putting concrete biasing classes in the physics packages ?
- Facilities for “histories”:
  - Ie trajectory-like quantities, recording weight evolution, very useful when chasing for big weights
- Facilities to monitor convergence

Conclusion

- What has been done:
  - Review of principles (importance sampling, splitting)
  - Review of formalism for GPIL biasing, and extension compared to existing packages
    - Per process modification interaction length looks doable and consistent
  - Prototype implementation of G4VBiasingOperation, G4VBiasingOperator, G4BiasingProcessInterface
  - Prototype implementation for:
    - Force interaction operation (exp. And flat laws)
    - Forced free flight operation
    - Exponential transform operation
    - (and related simple operators to allow activating these operations for the tracking)
  - Tools to help configuring the physics list simply
- What is missing:
  - Have to release this code somewhere in the distribution
  - Final state biasing not prototyped yet:
    - Some biasing operations should be “easy”
    - Mode difficult is final state biasing in term of biasing distributions:
      - Not generic “differential cross-section” concept
      - Would explicit calls to the physics packages facilities
  - Facilities for “histories”:
  - Facilities to monitor convergence

BACKUP

COMPARISON WITH FLUKA AND MCNP ?

Tentative comparison with FLUKA and MCNP functionalities

- Compare the existing FLUKA and MCNPX functionalities with the existing and *planned* Geant4 ones, provided by previous presented development.
  - Ie : comparison is not exactly fair with FLUKA and MCNPX
  - Need to say the above to be fair ;)
- But also, to repeat, as based on abstract classes, a non-implemented functionality can be provided by (well-advertised) user.
- Coming comparison is also an exercise to check if proposed design is able to cover FLUKA and MCNPX functionalities
  - Even if, in a first stage, all functionalities may not be concretely provided.

FLUKA / Geant4 biasing functionalities

Biasing options in FLUKA from http://www.fluka.org/content/manuals/fluka2011.manual	**Options in Geant4,** present or future
Leading particle biasing for electrons and photons: region dependent, below user-defined energy threshold and for selected physical effects. ?	Will be done, and will be shared with other "leading".
Russian Roulette and splitting at boundary crossing based on region relative importance.	Existing. Will be "re-provided" with new design, prototyped.
Region-dependent multiplicity tuning in high energy nuclear interactions.	Can be done and more general.
Region-dependent biased downscattering and non-analogue absorption of low-energy neutrons.	Can be done.
Biased decay length for increased daughter production.	Will be done. Prototyped.
Biased inelastic nuclear interaction length.	Will be done. Prototyped.
Biased interaction lengths for electron and photon electromagnetic interactions.	Will be done. Prototyped.
Biased angular distribution of decay secondary particles.	Can de done.
Region-dependent weight window in three energy ranges (and energy group dependent for low energy neutrons).	Existing. Can be "re-provided" and more general.
Bias setting according to a user-defined logics.	(need more info in FLUKA, but is actual purpose of this dev.)
User-defined neutrino direction biasing.	Can be done, easily.
User-defined step by step importance biasing.	Can be done, easily.

Simple for neutral. More difficult for charged but understood.

MCNPX / Geant4 biasing functionalities

Biasing options in MCNPX From LA-UR-03-1987, MCNP5 manual	**Options in Geant4,** present or future
Energy Cutoff & Time Cutoff	Existing (not considered as biasing)
Geometry Splitting with Russian Roulette	Existing. Will be "re-provided" with new design, prototyped.
Energy Splitting/Roulette and Time Splitting/Roulette	Can be done easily.
Weight Cutoff	Existing (in some way). Easy.
Weight Window	Existing. Will be "re-provided" with new design. Can be made more general.
Exponential Transform	Will be done. Prototyped.
Implicit Capture (or "Implicit capture," "survival biasing," and "absorption by weight reduction")	Can be done.
Forced Collisions	Will be done. Prototyped.
Source Variable Biasing	Existing.
Point Detector Tally (?)	(not biasing ?)
DXTRAN	Planned, need more work. Doable.
Correlated Sampling	Not planned for now "à la MCNP". But doable with user's invest.