The SGV fast detector simulation program

Mikael Berggren[1]

[1] DESY, Hamburg

ILD general meeting, Online, Feb 13, 20 24

CLUSTER OF EXCELLENCE
QUANTUM UNIVERSE

PHYSICS AT THE TERA SCALE
Helmholtz Alliance

DESY.

FTX

Outline


NEED FOR SPEED™
SPEEDHUNTERS
6666
Rocket Bunny
Rocket Bunny

Full SM simulation

Total cross-section @250 GeV for $e^+e^- \rightarrow$ 2f and 4f: $\mathcal{O}(1)$ nb (Whizard).

- $\int \mathcal{L}dt \sim = 5$ ab$^{-1} \rightarrow 5 \star 10^9$ events are expected.
- $\sim$ 1-20 ms to generate one event.
- $\sim$ 2 ms to fastsim (SGV) one event.

$\sim 5 \times 10^7$ s of CPU time is needed, ie around 1 year. But:This goes to 2000 years with full simulation. And this is just to simulate as many events as the data...

This does not include all bhabha's and $\gamma\gamma$ events $\sim$ an order of magnitude more $\Rightarrow$ without FastSim skimming, many of our results risk to be systematics dominated by lack of MC statistics!

Full SM simulation

Total cross-section @250 GeV for $e^+e^- \rightarrow$ 2f and 4f: $\mathcal{O}(1)$ nb (Whizard).

- $\int \mathcal{L}dt \sim = 5$ ab$^{-1} \rightarrow 5 \star 10^9$ events are expected.
- $\sim$ 1-20 ms to generate one event.
- $\sim$ 2 ms to fastsim (SGV) one event.

$\sim 5 \times 10^7$ s of CPU time is needed, ie around 1 year. But:This goes to 2000 years with full simulation. And this is just to simulate as many events as the data...

This does not include all bhabha's and $\gamma\gamma$ events $\sim$ an order of magnitude more $\Rightarrow$ without FastSim skimming, many of our results risk to be systematics dominated by lack of MC statistics!

SUSY parameter scans

Simple example:

- MSUGRA: 4 parameters + sign of μ
- Scan each in eg. 20 steps
- Eg. 5000 events per point (modest requirement: in sps1a' almost 1 million SUSY events are expected for 500 fb^{-1} !)
- = $20^4 \times 2 \times 5000 = 1.6 \times 10^9$ events to generate...

Slower to generate and simulate than $\gamma\gamma$ events

Also here: CPU millenniums with full simulation

SUSY parameter scans

Simple example:

- MSUGRA: 4 parameters + sign of μ
- Scan each in eg. 20 steps
- Eg. 5000 events per point (modest requirement: in sps1a' almost 1 million SUSY events are expected for 500 fb^{-1} !)
- = $20^4 \times 2 \times 5000 = 1.6 \times 10^9$ events to generate...

Slower to generate and simulate than $\gamma\gamma$ events

Also here: CPU millenniums with full simulation

Fast simulation types, and the choice for ILC

Different types, with increasing level of sophistication:

- 4-vector smearing.
- Parametric, needing input from FullSim: Traditional Delphes
  - Hard to treat correlations, eg. between p measurement and ip:s.
  - Hard to handle confusion in high granularity calorimeters.
  - No dE/dx, secondary vertices, effect of hit level inefficiencies ....
  - But very fast, and very condensed output.
  - By theorists, for theorists.
- Covariance matrix machines, not needing input from FullSim:SGV
  - Full covariance matrix available for each track-helix.
  - Individual shower shape and position generated ⇒ can approximate confusion.
  - Hit patterns known ⇒ dE/dX and hit-level efficiencies doable.
  - Covariance matrices available ⇒ vertex fitting.
  - Anything up to DST-level detail can be output.
  - As fast as Delphes.
  - By experimetalists, for both exprimetalists and theorists.

Fast simulation types, and the choice for ILC

Different types, with increasing level of sophistication:

- 4-vector smearing.
- Parametric, needing input from FullSim: Traditional Delphes
  - Hard to treat correlations, eg. between p measurement and ip:s.
  - Hard to handle confusion in high granularity calorimeters.
  - No dE/dx, secondary vertices, effect of hit level inefficiencies ....
  - But very fast, and very condensed output.
  - By theorists, for theorists.
- Covariance matrix machines, not needing input from FullSim:SGV
  - Full covariance matrix available for each track-helix.
  - Individual shower shape and position generated ⇒ can approximate confusion.
  - Hit patterns known ⇒ dE/dX and hit-level efficiencies doable.
  - Covariance matrices available ⇒ vertex fitting.
  - Anything up to DST-level detail can be output.
  - As fast as Delphes.
  - By experimetalists, for both exprimetalists and theorists.

Fast simulation types, and the choice for ILC

Different types, with increasing level of sophistication:

- 4-vector smearing.
- Parametric, needing input from FullSim: Traditional Delphes
  - Hard to treat correlations, eg. between p measurement and ip:s.
  - Hard to handle confusion in high granularity calorimeters.
  - No dE/dx, secondary vertices, effect of hit level inefficiencies ....
  - But very fast, and very condensed output.
  - By theorists, for theorists.
- Covariance matrix machines, not needing input from FullSim:SGV
  - Full covariance matrix available for each track-helix.
  - Individual shower shape and position generated ⇒ can approximate confusion.
  - Hit patterns known ⇒ dE/dX and hit-level efficiencies doable.
  - Covariance matrices available ⇒ vertex fitting.
  - Anything up to DST-level detail can be output.
  - As fast as Delphes.
  - By experimetalists, for both exprimetalists and theorists.

Fast simulation types, and the choice for ILC

Different types, with increasing level of sophistication:

- 4-vector smearing.
- Parametric, needing input from FullSim: Traditional Delphes
  - Hard to treat correlations, eg. between p measurement and ip:s.

For ILC:

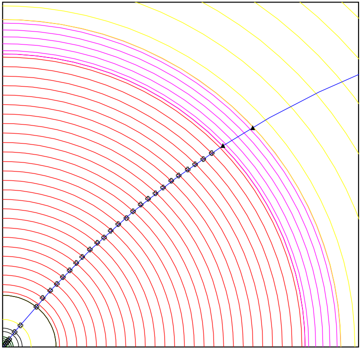
Only Covariance matrix machines have sufficient detail. Here, I'll cover "la Simulation à Grande Vitesse", SGV.

- Covariance matrix machines, not needing input from FullSim:SGV
  - Full covariance matrix available for each track-helix.
  - Individual shower shape and position generated ⇒ can approximate confusion.
  - Hit patterns known ⇒ dE/dX and hit-level efficiencies doable.
  - Covariance matrices available ⇒ vertex fitting.
  - Anything up to DST-level detail can be output.
  - As fast as Delphes.
  - By experimetalists, for both exprimetalists and theorists.

SGV: How tracking works

SGV is a machine to calculate covariance matrices

Tracking: Follow track-helix through the detector.

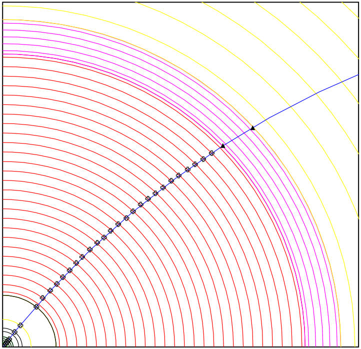


- Calculate cov. mat. at perigee, including material, measurement errors and extrapolation. NB: this is exactly what Your Kalman filter does!
- Smear perigee parameters (Choleski decomposition: takes all correlations into account)
- Helix *parameters* exactly calculated, *errors* with one approximation: helix moved to (0,0,0) for this.

SGV: How tracking works

SGV is a machine to calculate covariance matrices

Tracking: Follow track-helix through the detector.

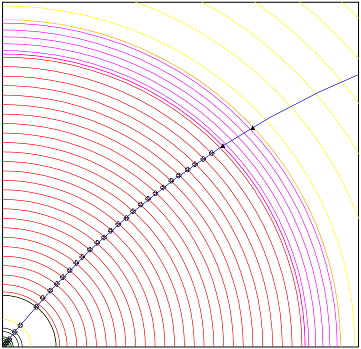


- Calculate cov. mat. at perigee, including material, measurement errors and extrapolation. NB: this is exactly what Your Kalman filter does!
- Smear perigee parameters (Choleski decomposition: takes all correlations into account)
- Helix *parameters* exactly calculated, *errors* with one approximation: helix moved to (0,0,0) for this.

SGV: How tracking works

SGV is a machine to calculate covariance matrices

Tracking: Follow track-helix through the detector.

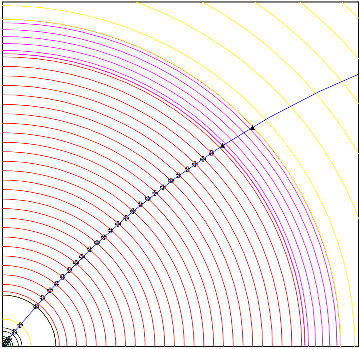


- Calculate cov. mat. at perigee, including material, measurement errors and extrapolation. NB: this is exactly what Your Kalman filter does!
- Smear perigee parameters (Choleski decomposition: takes all correlations into account)
- Helix *parameters* exactly calculated, *errors* with one approximation: helix moved to (0,0,0) for this.

SGV: How tracking works

SGV is a machine to calculate covariance matrices

Tracking: Follow track-helix through the detector.

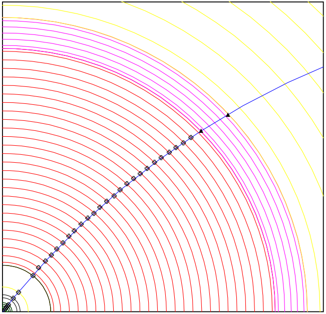


- Calculate cov. mat. at perigee, including material, measurement errors and extrapolation. NB: this is exactly what Your Kalman filter does!
- Smear perigee parameters (Choleski decomposition: takes all correlations into account)
- Helix *parameters* exactly calculated, *errors* with one approximation: helix moved to (0,0,0) for this.

SGV: How the rest works

SGV is a machine to calculate covariance matrices

Calorimeters: Follow particle to intersection with calorimeters.



- Response type: MIP, EM or hadronic shower, below threshold, etc.
- Simulate single particle response from parameters.
- Easy to plug in more sophisticated shower-simulation. More in a minute...

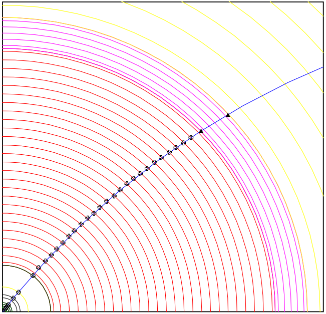
Other stuff:

- EM-interactions in detector material simulated
- Plug-ins for particle identification, track-finding efficiencies,...
- Information on hit-patterns accessible to analysis.

SGV: How the rest works

SGV is a machine to calculate covariance matrices

Calorimeters: Follow particle to intersection with calorimeters.



- Response type: MIP, EM or hadronic shower, below threshold, etc.
- Simulate single particle response from parameters.
- Easy to plug in more sophisticated shower-simulation. More in a minute...

Other stuff:

- EM-interactions in detector material simulated
- Plug-ins for particle identification, track-finding efficiencies,...
- Information on hit-patterns accessible to analysis.

SGV: How the rest works

- User data, delivered in Module-global arrays:
  - Extended 4-vectors .
  - Track helix parameters with correlations.
  - Calorimetric clusters.
  - When relevant: true values.
  - Auxiliary information on particle history, detector-elements used etc.
  - Event-global variables.
- User Analysis tasks :
  - Jet-finding.
  - Event-shapes.
  - Primary and secondary vertex fitting.
  - Impact parameters.

  Can be calculated by routines, included in SGV. Access routines give an easy interface to the detector geometry.

SGV: How the rest works

- User data, delivered in Module-global arrays:
  - Extended 4-vectors .
  - Track helix parameters with correlations.
  - Calorimetric clusters.
  - When relevant: true values.
  - Auxiliary information on particle history, detector-elements used etc.
  - Event-global variables.
- User Analysis tasks :
  - Jet-finding.
  - Event-shapes.
  - Primary and secondary vertex fitting.
  - Impact parameters.

Can be calculated by routines, included in SGV. Access routines give an easy interface to the detector geometry.

Calorimeter simulation: SGV strategy

- Concentrate on what really matters:
  - True charged particles splitting off (a part of) their shower: double-counting.
  - True neutral particles merging (a part of) their shower with charged particles: enetgy loss.
- Don't care about neutral-neutral or charged-charged merging.
- Nor about multiple splitting/merging.
- Then: identify the most relevant variables available in fast simulation:
  - Cluster energy.
  - Distance to nearest particle of "the other type"
  - EM or hadron.
  - Barrel or end-cap.

Calorimeter simulation: SGV strategy

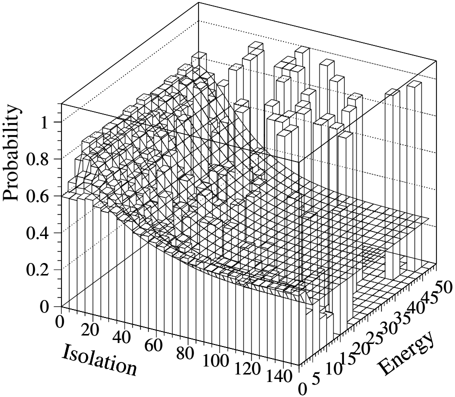
- Concentrate on what really matters:
  - True charged particles splitting off (a part of) their shower: double-counting.
  - True neutral particles merging (a part of) their shower with charged particles: enetgy loss.
- Don't care about neutral-neutral or charged-charged merging.
- Nor about multiple splitting/merging.
- Then: identify the most relevant variables available in fast simulation:
  - Cluster energy.
  - Distance to nearest particle of "the other type"
  - EM or hadron.
  - Barrel or end-cap.

Calorimeter simulation: SGV strategy

- Concentrate on what really matters:
  - True charged particles splitting off (a part of) their shower: double-counting.
  - True neutral particles merging (a part of) their shower with charged particles: enetgy loss.
- Don't care about neutral-neutral or charged-charged merging.
- Nor about multiple splitting/merging.
- Then: identify the most relevant variables available in fast simulation:
  - Cluster energy.
  - Distance to nearest particle of "the other type"
  - EM or hadron.
  - Barrel or end-cap.

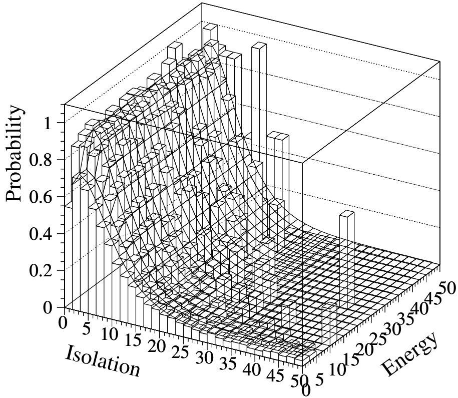
Observed distributions

- Probability to split (charged had or γ)
- Fraction the energy vs distance
- ... and vs E
- Fit of the Distribution of the fraction
- Average fraction vs. E and distance.



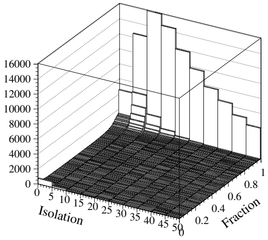
Observed distributions

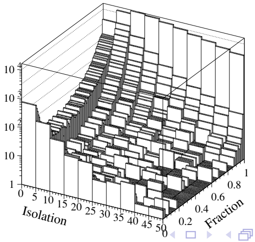
- Probability to split (charged had or γ)
- Fraction the energy vs distance
- ... and vs E
- Fit of the Distribution of the fraction
- Average fraction vs. E and distance.



Observed distributions

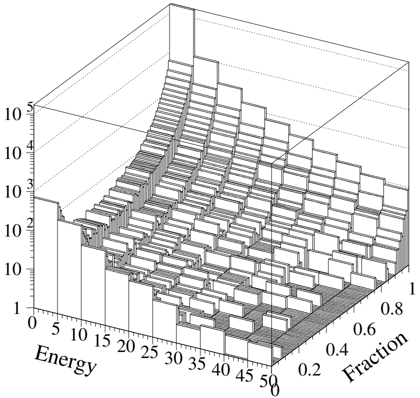
- Probability to split (charged had or γ)
- Fraction the energy vs distance
- ... and vs E
- Fit of the Distribution of the fraction
- Average fraction vs. E and distance.





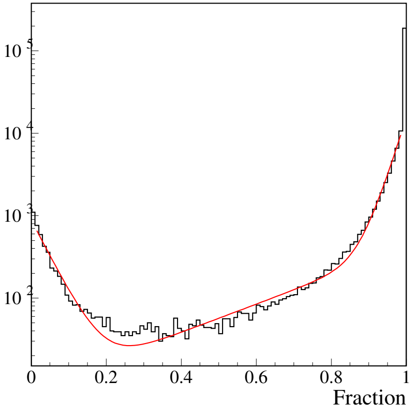
Observed distributions

- Probability to split (charged had or γ)
- Fraction the energy vs distance
- ... and vs E
- Fit of the Distribution of the fraction
- Average fraction vs. E and distance.



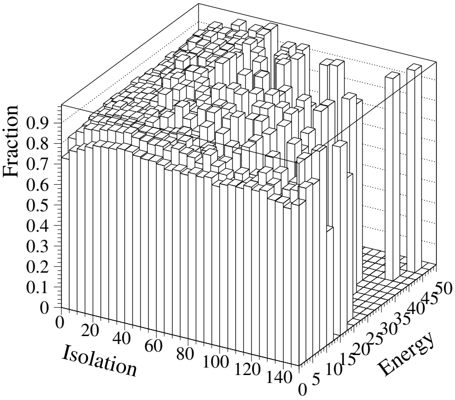
Observed distributions

- Probability to split (charged had or γ)
- Fraction the energy vs distance
- ... and vs E
- Fit of the Distribution of the fraction
- Average fraction vs. E and distance.



Observed distributions

- Probability to split (charged had or γ)
- Fraction the energy vs distance
- ... and vs E
- Fit of the Distribution of the fraction
- Average fraction vs. E and distance.



Observations

- Identify and factorise:
  1. Probability to split
  2. If split, probability to split off/merge the entire cluster.
  3. If split, but not 100 %: Form of the p.d.f. of the fraction split off.
- Observations:
  1. Depnds on the isolation - strongly for merging, slightly for splitting - but can be treated in two energy bins with no energy dependence in the bin. %5 over-all dependence on barrel/endcap.
  2. Depends only on energy. Is small for splitting, important for merging at low E.
  3. Depends on both energy and isolation (very little for splitting), but only via the average.
- All cases (EM/had - split/merge - Barrel/endcap) can be described by the same functional shapes.
- Functions are combinations of exponentials and lines.
- 28 parameters × 4 cases (em/had × double-counting/loss)

Observations

- Identify and factorise:
  1. Probability to split
  2. If split, probability to split off/merge the entire cluster.
  3. If split, but not 100 %: Form of the p.d.f. of the fraction split off.
- Observations:
  1. Depnds on the isolation - strongly for merging, slightly for splitting - but can be treated in two energy bins with no energy dependence in the bin. %5 over-all dependence on barrel/endcap.
  2. Depends only on energy. Is small for splitting, important for merging at low E.
  3. Depends on both energy and isolation (very little for splitting), but only via the average.
- All cases (EM/had - split/merge - Barrel/endcap) can be described by the same functional shapes.
- Functions are combinations of exponentials and lines.
- 28 parameters × 4 cases (em/had × double-counting/loss)

Observations

- Identify and factorise:
  1. Probability to split
  2. If split, probability to split off/merge the entire cluster.
  3. If split, but not 100 %: Form of the p.d.f. of the fraction split off.
- Observations:
  1. Depnds on the isolation - strongly for merging, slightly for splitting - but can be treated in two energy bins with no energy dependence in the bin. %5 over-all dependence on barrel/endcap.
  2. Depends only on energy. Is small for splitting, important for merging at low E.
  3. Depends on both energy and isolation (very little for splitting), but only via the average.
- All cases (EM/had - split/merge - Barrel/endcap) can be described by the same functional shapes.
- Functions are combinations of exponentials and lines.
- 28 parameters × 4 cases (em/had × double-counting/loss)

Observations

- Identify and factorise:
  1. Probability to split
  2. If split, probability to split off/merge the entire cluster.
  3. If split, but not 100 %: Form of the p.d.f. of the fraction split off.
- Observations:
  1. Depnds on the isolation - strongly for merging, slightly for splitting - but can be treated in two energy bins with no energy dependence in the bin. %5 over-all dependence on barrel/endcap.
  2. Depends only on energy. Is small for splitting, important for merging at low E.
  3. Depends on both energy and isolation (very little for splitting), but only via the average.
- All cases (EM/had - split/merge - Barrel/endcap) can be described by the same functional shapes.
- Functions are combinations of exponentials and lines.
- 28 parameters × 4 cases (em/had × double-counting/loss)

Observations

- Identify and factorise:
  1. Probability to split
  2. If split, probability to split off/merge the entire cluster.
  3. If split, but not 100 %: Form of the p.d.f. of the fraction split off.
- Observations:
  1. Depnds on the isolation - strongly for merging, slightly for splitting - but can be treated in two energy bins with no energy dependence in the bin. %5 over-all dependence on barrel/endcap.
  2. Depends only on energy. Is small for splitting, important for merging at low E.
  3. Depends on both energy and isolation (very little for splitting), but only via the average.
- All cases (EM/had - split/merge - Barrel/endcap) can be described by the same functional shapes.
- Functions are combinations of exponentials and lines.
- 28 parameters $\times$ 4 cases (em/had $\times$ double-counting/loss)

Steering SGV

Two steering files...

- Program steering:
  - Single file, with sections for general, generator, detector and analysis steering.
  - Many examples included.
  - Extensive comments in these.
- Geometry description:
  - Described by cylinders and planes.
  - Attach material properties (rad. length, material, int. length, ...)
  - Attach measurement capabilities (quantities measured, dependence on local angles, ...)
  - Several commented examples included.
  - For all details: Chapter 6 in sgv_ug.pdf (created during install)

Steering SGV

Two steering files...

- Program steering:
  - Single file, with sections for general, generator, detector and analysis steering.
  - Many examples included.
  - Extensive comments in these.
- Geometry description:
  - Described by cylinders and planes.
  - Attach material properties (rad. length, material, int. length, ...)
  - Attach measurement capabilities (quantities measured, dependence on local angles, ...)
  - Several commented examples included.
  - For all details: Chapter 6 in sgv_ug.pdf (created during install)

Steering SGV

Two steering files...

- Program steering:
  - Single file, with sections for general, generator, detector and analysis steering.
  - Many examples included.
  - Extensive comments in these.
- Geometry description:
  - Described by cylinders and planes.
  - Attach material properties (rad. length, material, int. length, ...)
  - Attach measurement capabilities (quantities measured, dependence on local angles, ...)
  - Several commented examples included.
  - For all details: Chapter 6 in sgv_ug.pdf (created during install)

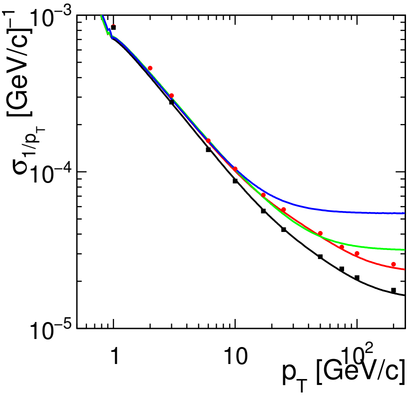
Example Geometry:TPC simplified and full

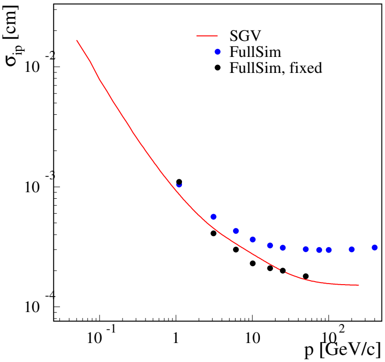
```
BF: BFIELD=3.5

TYPE:  BARREL_TRACKING
         .
         .

      LAYER : TPC
         REPEAT : TIMES=26, DELTA_R=5.448

            GEOMETRY:  R=39.5  , ZMIN=0.0, ZMAX=230.25
            MATERIAL:    X0=0.00047
            MEASUREMENT: CODE=1, SIG_RPHI=0.00301, SIG_Z=0.09,
                          SIG_RPHI_SLOPE=-0.0000058
         ENDREPEAT :
         .
         .
TYPE:  FORWARD_TRACKING
         .
         .

TYPE: BARREL_CALORIMETRY
         .
         .

TYPE: FORWARD_CALORIMETRY
         .
         .
```

Example Geometry:TPC simplified and full

```
BF: BFIELD=3.5

TYPE:  BARREL_TRACKING
         .
         .

      LAYER : TPC
         REPEAT : TIMES=225, DELTA_R=0.6

            GEOMETRY:  R=39.5  , ZMIN=0.0, ZMAX=230.25
            MATERIAL:    X0=0.000052222
            MEASUREMENT: CODE=1,SIG_RPHI=0.0050,SIG_Z=0.04,
                          SIG_RPHI_BETA=0.090,
                          DIFFUSION=0.0053,
                          MOBILITY=3.0
         ENDREPEAT :
         .
         .

TYPE:  FORWARD_TRACKING
         .
         .

TYPE: BARREL_CALORIMETRY
         .
         .

TYPE: FORWARD_CALORIMETRY
         .
         .
```

SGV and FullSim ILD: Tracking

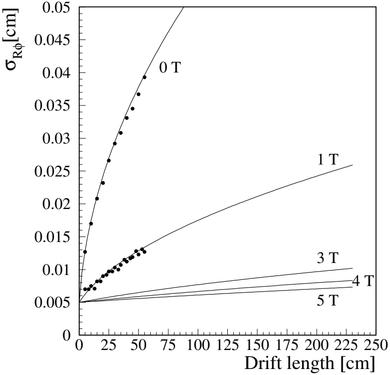




Lines: SGV, dots: Mokka+Marlin

TPC point-resolution vs. Z and B in SGV and test-beam

- Points: Prototype measurements (from DBD/DBD SVN)



- Lines: Formula used in SGV: $\sigma^2_{R\phi} = \sigma^2_0 + (\beta_{slope} \sin\phi)^2 + \frac{D^2_{0T}}{1+(\mu B)^2} \sin\theta Z$, $\Rightarrow$ complicated relation, but gets better with shorter drift-length and higher B.

SGV and FullSim ILD: Jets and events

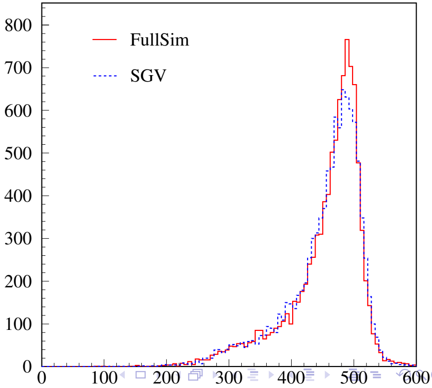
Feed exactly the same physics events through FullSim or SGV.

- Overall:
  - Total seen energy
- $e^+e^- \to ZZ \to$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag

SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

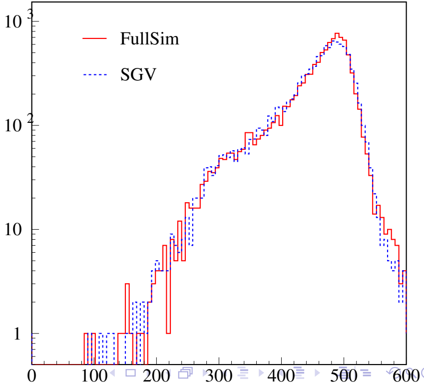
- Overall:
  - Total seen energy
- $e^+e^- \to ZZ \to$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

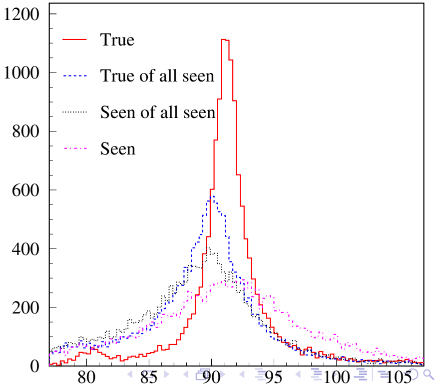
- Overall:
  - Total seen energy
- $e^+e^- \rightarrow ZZ \rightarrow$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

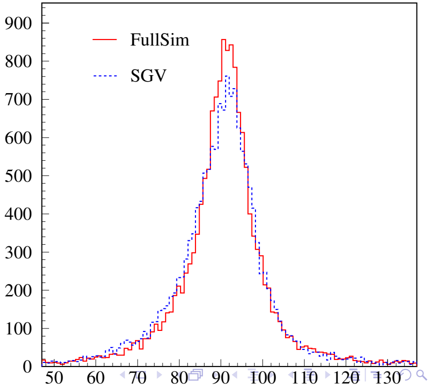
- Overall:
  - Total seen energy
- $e^+e^- \rightarrow ZZ \rightarrow$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

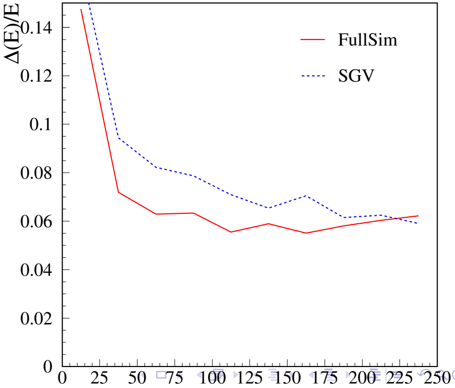
- Overall:
  - Total seen energy
- $e^+e^- \rightarrow ZZ \rightarrow$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

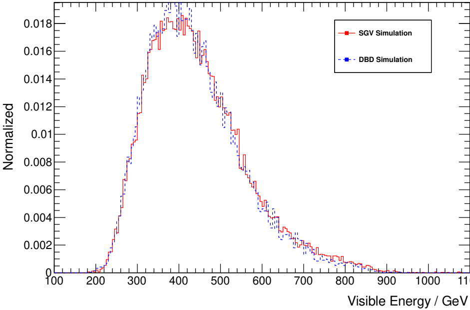
- Overall:
  - Total seen energy
- $e^+e^- \rightarrow ZZ \rightarrow$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

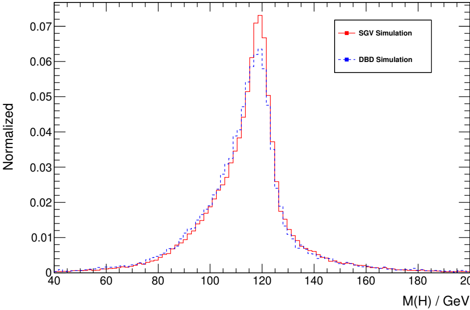
- Overall:
  - Total seen energy
- $e^+e^- \rightarrow ZZ \rightarrow$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

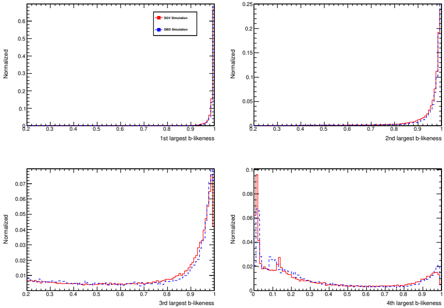
- Overall:
  - Total seen energy
- $e^+e^- \rightarrow ZZ \rightarrow$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties $\Rightarrow$ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



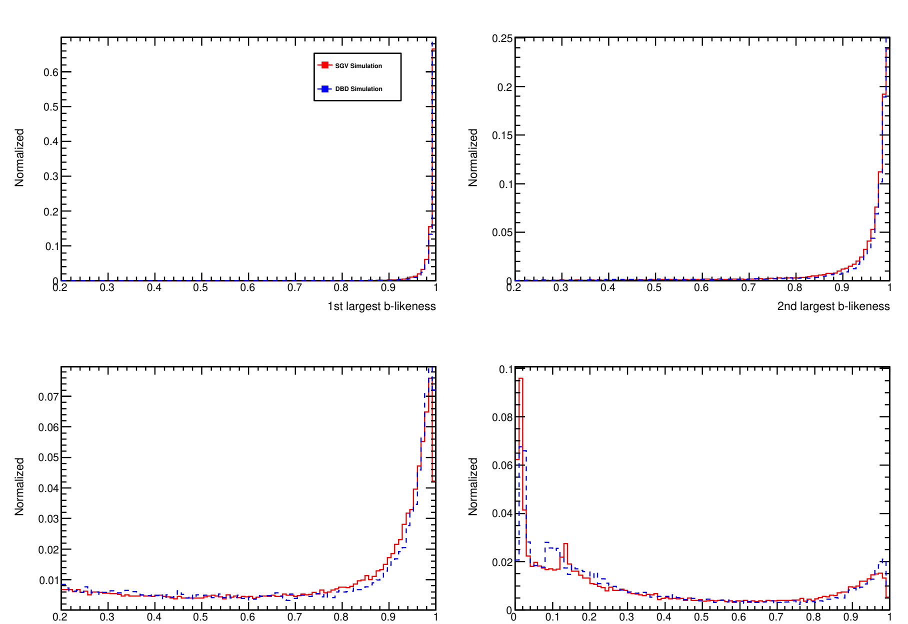
SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

- Overall:
  - Total seen energy
- $e^+e^- \to ZZ \to$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties ⇒ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



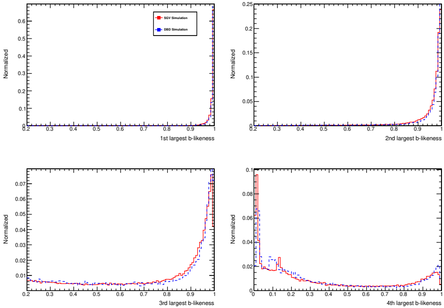
SGV and FullSim ILD: Jets and events



SGV and FullSim ILD: Jets and events

Feed exactly the same physics events through FullSim or SGV.

- Overall:
  - Total seen energy
- $e^+e^- \rightarrow ZZ \rightarrow$ four jets:
  - Reconstructed M_Z at different stages in FullSim.
  - Seen Reconstructed M_Z, FullSim and SGV.
  - Jet-Energy resolution (NB: r.m.s., including jet-finding uncertainties $\Rightarrow$ not the standard plot for JER!)
- *Zhh* at 1 TeV:
  - Visible E
  - Higgs Mass
  - b-tag



SGV for Detector optimisation: Outcome

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways, in total 300 different models. Details in backup and AWLC-KEK 2015.

Construct metrics on performance
- both low level and physics output
- and on potential savings w.r.t. the baseline detector.

- Only at least 5 % savings, best $\sim$ 5 performances in each group of mitigation strategies.
- For $\sim$ all strategies, the same sizes remain.
- performance and savings iso-curves

SGV for Detector optimisation: Outcome

For the “IDR” detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways, in total 300 different models. Details in backup and AWLC-KEK 2015.

Construct metrics on performance
- both low level and physics output
- and on potential savings w.r.t. the baseline detector.

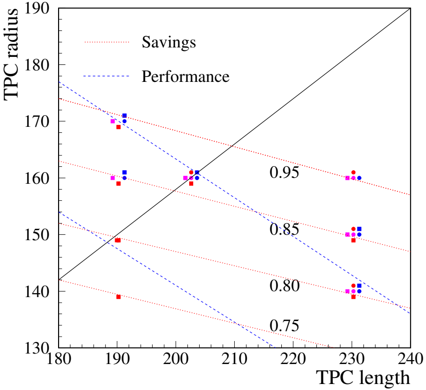
- Only at least 5 % savings, best $\sim$ 5 performances in each group of mitigation strategies.
- For $\sim$ all strategies, the same sizes remain.
- performance and savings iso-curves

SGV for Detector optimisation: Outcome

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways, in total 300 different models. Details in backup and AWLC-KEK 2015.

Construct metrics on performance
- both low level and physics output
- and on potential savings w.r.t. the baseline detector.

- Only at least 5 % savings, best ∼ 5 performances in each group of mitigation strategies.
- For ∼ all strategies, the same sizes remain.
- performance and savings iso-curves



Technicalities

- Written in Fortran 08, a re-write of the Fortran77-based SGV2 series.Managed in git.Install script included.
- Requires:
  - Fortran compiler, e.g. gfortran - any version between 4.7 and 10 ($\Rightarrow$ future-proof!).
  - Standard Linux math: `blas` and `lapack`.
  - PYTHIA vers 6 (even if not needed for event-generation: SGV uses many PYTHIA6 extras).
  - To produce the doc's during installation: `TexLive` and `pandoc`
- Features:
  - On-demand memory allocation.
  - Callable PYTHIA, Whizard 1.x, ...
  - Input from Hepevt/PYJETS, stdhep, slcio, HepMC2/3, GuineaPig.
  - Output of selected generated events to PYJETS, stdhep, slcio, or HepMC2/3.
  - No imposed format of reconstructed events. Code to make LCIO-DSTs or ROOT trees supplied.
- Typical generation+simulation+reconstruction time $\mathcal{O}(1)$ ms.

Technicalities

- Written in Fortran 08, a re-write of the Fortran77-based SGV2 series.Managed in git.Install script included.
- Requires:
  - Fortran compiler, e.g. gfortran - any version between 4.7 and 10 ($\Rightarrow$ future-proof!).
  - Standard Linux math: `blas` and `lapack`.
  - PYTHIA vers 6 (even if not needed for event-generation: SGV uses many PYTHIA6 extras).
  - To produce the doc's during installation: `TexLive` and `pandoc`
- Features:
  - On-demand memory allocation.
  - Callable PYTHIA, Whizard 1.x, ...
  - Input from Hepevt/PYJETS, stdhep, slcio, HepMC2/3, GuineaPig.
  - Output of selected generated events to PYJETS, stdhep, slcio, or HepMC2/3.
  - No imposed format of reconstructed events. Code to make LCIO-DSTs or ROOT trees supplied.
- Typical generation+simulation+reconstruction time $\mathcal{O}(1)$ ms.

Technicalities

- Written in Fortran 08, a re-write of the Fortran77-based SGV2 series.Managed in git.Install script included.
- Requires:
  - Fortran compiler, e.g. gfortran - any version between 4.7 and 10 ($\Rightarrow$ future-proof!).
  - Standard Linux math: `blas` and `lapack`.
  - PYTHIA vers 6 (even if not needed for event-generation: SGV uses many PYTHIA6 extras).
  - To produce the doc's during installation: `TexLive` and `pandoc`
- Features:
  - On-demand memory allocation.
  - Callable PYTHIA, Whizard 1.x, ...
  - Input from Hepevt/PYJETS, stdhep, slcio, HepMC2/3, GuineaPig.
  - Output of selected generated events to PYJETS, stdhep, slcio, or HepMC2/3.
  - No imposed format of reconstructed events. Code to make LCIO-DSTs or ROOT trees supplied.
- Typical generation+simulation+reconstruction time $\mathcal{O}(1)$ ms.

Technicalities

- Written in Fortran 08, a re-write of the Fortran77-based SGV2 series.Managed in git.Install script included.
- Requires:
  - Fortran compiler, e.g. gfortran - any version between 4.7 and 10 ($\Rightarrow$ future-proof!).
  - Standard Linux math: `blas` and `lapack`.
  - PYTHIA vers 6 (even if not needed for event-generation: SGV uses many PYTHIA6 extras).
  - To produce the doc's during installation: `TexLive` and `pandoc`
- Features:
  - On-demand memory allocation.
  - Callable PYTHIA, Whizard 1.x, ...
  - Input from Hepevt/PYJETS, stdhep, slcio, HepMC2/3, GuineaPig.
  - Output of selected generated events to PYJETS, stdhep, slcio, or HepMC2/3.
  - No imposed format of reconstructed events. Code to make LCIO-DSTs or ROOT trees supplied.
- Typical generation+simulation+reconstruction time $\mathcal{O}(1)$ ms.

Getting started with SGV

Do

git clone https://berggren@gitlab.desy.de/mikael.berggren/sgv.git sgv/

Check that `lapack` and `blas` installed. If PYTHIA6 is not installed, check in `README` how to get it from HepForge.

Then

cd sgv ; . ./install

This will take you about 30 seconds ...

- Study README do get the first test job done (another 30 seconds)
- Look README in the samples sub-directory, to enhance the capabilities, eg.:
  - Get ROOT interface set up, and produce you first ROOT tree.
  - Get the LCIO, HepMC, and/or STDHEP i/o set up.
  - Make you first LCIO DST.

Getting started with SGV

Do

git clone https://berggren@gitlab.desy.de/mikael.berggren/sgv.git sgv/

Check that `lapack` and `blas` installed. If PYTHIA6 is not installed, check in `README` how to get it from HepForge.

Then

cd sgv ; . ./install

This will take you about 30 seconds ...

- Study README do get the first test job done (another 30 seconds)
- Look README in the samples sub-directory, to enhance the capabilities, eg.:
  - Get ROOT interface set up, and produce you first ROOT tree.
  - Get the LCIO, HepMC, and/or STDHEP i/o set up.
  - Make you first LCIO DST.

Getting started with SGV

Do

git clone https://berggren@gitlab.desy.de/mikael.berggren/sgv.git sgv/

Check that `lapack` and `blas` installed. If PYTHIA6 is not installed, check in `README` how to get it from HepForge.

Then

cd sgv ; . ./install

This will take you about 30 seconds ...

- Study README do get the first test job done (another 30 seconds)
- Look README in the samples sub-directory, to enhance the capabilities, eg.:
  - Get ROOT interface set up, and produce you first ROOT tree.
  - Get the LCIO, HepMC, and/or STDHEP i/o set up.
  - Make you first LCIO DST.

Getting started with SGV

Do

git clone https://berggren@gitlab.desy.de/mikael.berggren/sgv.git sgv/

Check that `lapack` and `blas` installed. If PYTHIA6 is not installed, check in `README` how to get it from HepForge.

To learn more about SGV:

Follow the ▸ tutorial! (There is an introduction first - The actual tutorial starts a 20:50.)

Then

cd sgv ; . ./inst

This will take you about 30 seconds ...

- Study README do get the first test job done (another 30 seconds)
- Look README in the samples sub-directory, to enhance the capabilities, eg.:
  - Get ROOT interface set up, and produce you first ROOT tree.
  - Get the LCIO, HepMC, and/or STDHEP i/o set up.
  - Make you first LCIO DST.

Summary and out-look

- The SGV FastSim program for ILC physics simulation was presented, and (I hope) was shown to be up to the job, both in physics and computing performance and stability (millions of events produced)
- SGV is a full-blown fast detector simulation, not just a parameterised four-vector smearer
  - Comparisons to FullSim was shown to be quite good, also for complicated features like h.f. tagging.
  - A pre-existing full simulation is not needed to get realistic results. Descriptions of e^+e^- detectors available after installation.
  - Still: SGV is as fast as eg. `Delphes`.
- Many input-methods (internal and external). No output format imposed on the user, plugins for LCIO and Root output available.

Summary and out-look

- The SGV FastSim program for ILC physics simulation was presented, and (I hope) was shown to be up to the job, both in physics and computing performance and stability (millions of events produced)
- SGV is a full-blown fast detector simulation, not just a parameterised four-vector smearer
  - Comparisons to FullSim was shown to be quite good, also for complicated features like h.f. tagging.
  - A pre-existing full simulation is not needed to get realistic results. Descriptions of e^+e^- detectors available after installation.
  - Still: SGV is as fast as eg. `Delphes`.
- Many input-methods (internal and external). No output format imposed on the user, plugins for LCIO and Root output available.

Summary and out-look

- The SGV FastSim program for ILC physics simulation was presented, and (I hope) was shown to be up to the job, both in physics and computing performance and stability (millions of events produced)
- SGV is a full-blown fast detector simulation, not just a parameterised four-vector smearer
  - Comparisons to FullSim was shown to be quite good, also for complicated features like h.f. tagging.
  - A pre-existing full simulation is not needed to get realistic results. Descriptions of e^+e^- detectors available after installation.
  - Still: SGV is as fast as eg. `Delphes`.
- Many input-methods (internal and external). No output format imposed on the user, plugins for LCIO and Root output available.

Summary and out-look

- The SGV FastSim program for ILC physics simulation was presented, and (I hope) was shown to be up to the job, both in physics and computing performance and stability (millions of events produced)
- SGV is a full-blown fast detector simulation, not just a parameterised four-vector smearer
  - Comparisons to FullSim was shown to be quite good, also for complicated features like h.f. tagging.
  - A pre-existing full simulation is not needed to get realistic results. Descriptions of e^+e^- detectors available after installation.
  - Still: SGV is as fast as eg. `Delphes`.
- Many input-methods (internal and external). No output format imposed on the user, plugins for LCIO and Root output available.

Summary and out-look

- New features to come:
  - EDM4HEP I/O
  - Currently, the detector is assumed to be mirror-symmetric. In a current development version, this restriction is removed (HALHF !)
  - Whizard 3.x has re-introduced Whizard-as-a-subroutine (was there in V1, but not in V2). So callable Whizard v3 will be implemented in SGV.
  - Add uncertainty in energy-loss into the track covariance matrix.
- A covariance-machine like SGV is needed not to be cornered between the systematic errors of a parameterised fast-sim, and the statistical errors of FullSim

Summary and out-look

- New features to come:
  - EDM4HEP I/O
  - Currently, the detector is assumed to be mirror-symmetric. In a current development version, this restriction is removed (HALHF !)
  - Whizard 3.x has re-introduced Whizard-as-a-subroutine (was there in V1, but not in V2). So callable Whizard v3 will be implemented in SGV.
  - Add uncertainty in energy-loss into the track covariance matrix.
- A covariance-machine like SGV is needed not to be cornered between the systematic errors of a parameterised fast-sim, and the statistical errors of FullSim

Summary and out-look

- New features to come:
  - EDM4HEP I/O
  - Currently, the detector is assumed to be mirror-symmetric. In a current development version, this restriction is removed (HALHF !)
  - Whizard 3.x has re-introduced Whizard-as-a-subroutine (was there in V1, but not in V2). So callable Whizard v3 will be implemented in SGV.
  - Add uncertainty in energy-loss into the track covariance matrix.
- A covariance-machine like SGV is needed not to be cornered between the systematic errors of a parameterised fast-sim, and the statistical errors of FullSim

... a tool for rapid LC studies?

Peer-reviewed papers using SGV

- Phys. Rev D101 (2020) 7, 075053
- ILD-PHYS-2019-001 (Accepted by Phys. ReV. D)
- Eur.Phys.J.C 76 (2016) 4,183
- Eur.Phys.J.C 75 (2015) 12, 617
- Phys. Rev D 91 (2015) 113007
- Phys. Rev D 90 (2014) 114029
- Phys. Rev D 89 (2014) 11, 113006
- Eur.Phys.J.C 73 (2013) 12,2660
- Eur.Phys.J.C 72 (2012) 2213
- Phys. Rev. D 82 (2010) 055016
- NIM A 579 (2007) 750
- Eur. Phys. J. C 31 (2003) 421
- Eur. Phys, J. direct (2000) 1

+ innumerable theses, reports, arXiv submissions and conference proceedings. Including the Tesla TDR, LoI, TDR, the IDR and ILC/ILD inputs to EPPSU and Snowmass 2013.

... a tool for rapid LC studies ?

SGV was used for

- Defining the forward tracking geometry of LDC:
  - Vienna 2005. ▸ LDC and ▸ tracking
- The utility (or not) of the silicon envelope
  - ▸ Valencia 2006
- Merge of LDC and GLD into ILD
  - ▸ Cambrige 2008
- Define the options for the IDR
  - ▸ KEK 2015
- Also: SGV is part of the FullSim machinery: It is used to select which part of the pairs-background to overlay.

... a tool for rapid LC studies ?

SGV was used for

- Defining the forward tracking geometry of LDC:
  - Vienna 2005. ▸ LDC and ▸ tracking
- The utility (or not) of the silicon envelope
  - ▸ Valencia 2006
- Merge of LDC and GLD into ILD
  - ▸ Cambrige 2008
- Define the options for the IDR
  - ▸ KEK 2015
- Also: SGV is part of the FullSim machinery: It is used to select which part of the pairs-background to overlay.

... a tool for rapid LC studies !

SGV was used for

- Defining the forward tracking geometry of LDC:
  - Vienna 2005. ▸ LDC and ▸ tracking
- The utility (or not) of the silicon envelope
  - ▸ Valencia 2006
- Merge of LDC and GLD into ILD
  - ▸ Cambrige 2008
- Define the options for the IDR
  - ▸ KEK 2015
- Also: SGV is part of the FullSim machinery: It is used to select which part of the pairs-background to overlay.

Thank You !

Thank You !

Backup

BACKUP SLIDES

LCIO Collections with DST output

- Added sensible values to all collections that will (probably) be there on the DST from the fullSim production.
  - BuildUpVertex
  - BuildUpVertex_RP
  - MarlinTrkTracks
  - PandoraClusters
  - PandoraPFOs
  - PrimaryVertex
  - RecoMCTruthLink
  - MCParticlesSkimmed
  - V0Vertices
  - V0RecoParticles
  - BCALParticles
  - BCALClusters
  - BCALMCTruthLink
  - PrimaryVertex_RP
- Also added more relation links:
  - MCTruthRecoLink
  - ClusterMCTruthLink
  - MCTruthClusterLink
  - MCTruthTrackLink
  - TrackMCTruthLink
  - MCTruthBcalLink

Comments

Secondary vertices (as before):

- Use true information to find all secondary vertices.
- For all vertices with ≥ 2 seen charged tracks: do vertex fit.
- Concequence:
  - Vertex *finding* is too good.
  - Vertex *quality* should be comparable to FullSim.

In addition: Decide from parent pdg-code if it goes into BuildUpVertex or V0Vertices !

MCParticle :

- There might be some issues with history codes in the earlier part of the event (initial beam-particles, 94-objects, ...)

Comments

Clusters:

- Are done with the Pandora confusion parametrisation on.
- Expect $\sim$ correct dispersion of jet energy, but a few % to high central value.
- See my talk three weeks ago.
- Warning: Clusters are always only in one detector , so don't use E_{had}/E_{EM} for e/π: It will be $\equiv$ 100 % efficient !

Navigators

- All the navigators that the TruthLinker processor makes when all flags are switched on are created:
  - Both Seen to True and True to Seen (weights are different !)
  - Seen is both PFOs, tracks and clusters.
  - The standard RecoMCTruthLink collection is as it would be from FullSim ie. weights between 0 and 1.

SGV for Detector optimisation

For the “IDR” detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

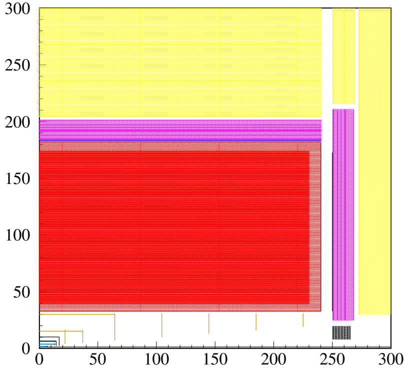
1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

back

SGV for Detector optimisation

For the “IDR” detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

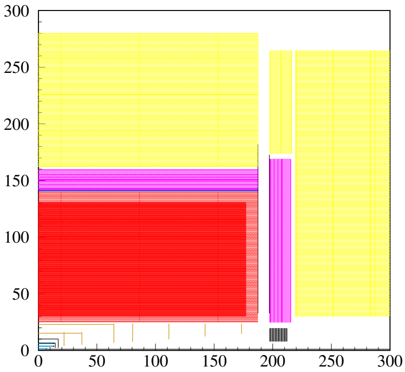


back

SGV for Detector optimisation

For the “IDR” detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

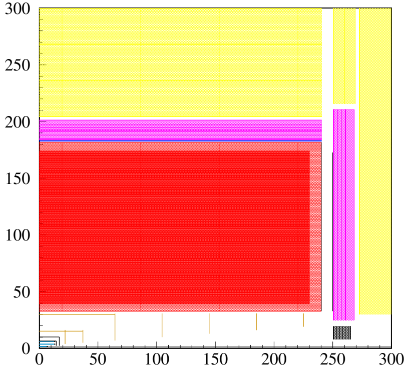


back

SGV for Detector optimisation

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

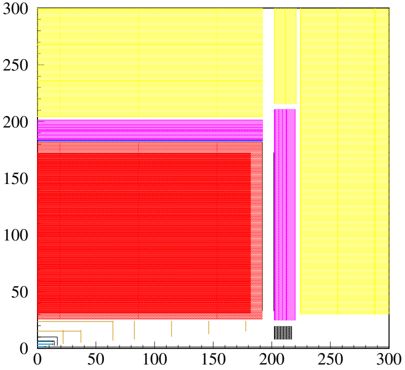


back

SGV for Detector optimisation

For the “IDR” detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

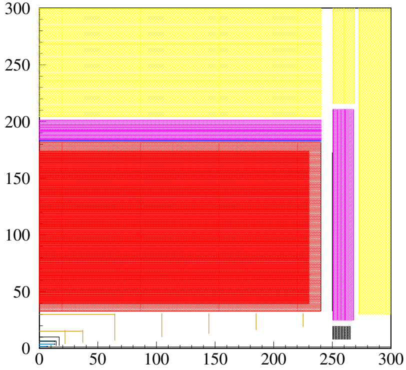


back

SGV for Detector optimisation

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

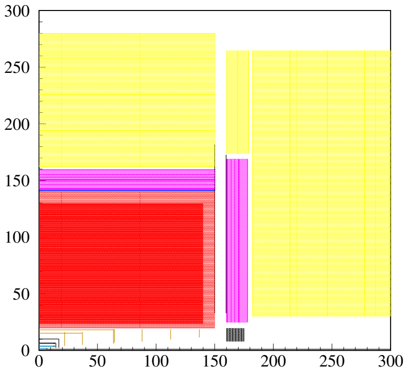


back

SGV for Detector optimisation

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

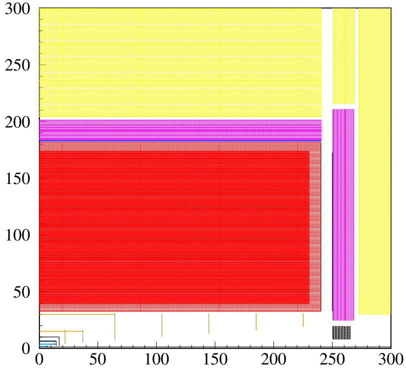


back

SGV for Detector optimisation

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

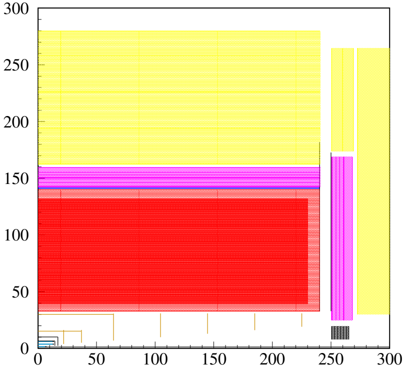


back

SGV for Detector optimisation

For the “IDR” detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

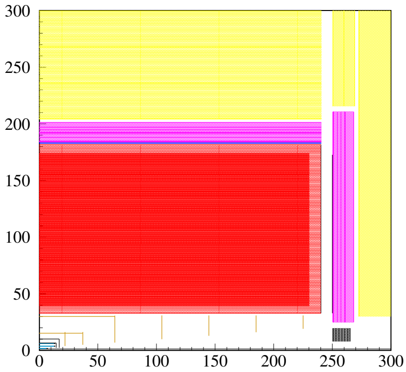


back

SGV for Detector optimisation

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.

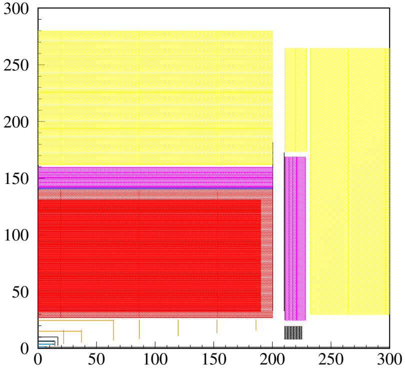


back

SGV for Detector optimisation

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep baseline radius.
3. Keep aspect ratio = 1
4. Keep baseline length.
5. Keep length = baseline-40 cm.



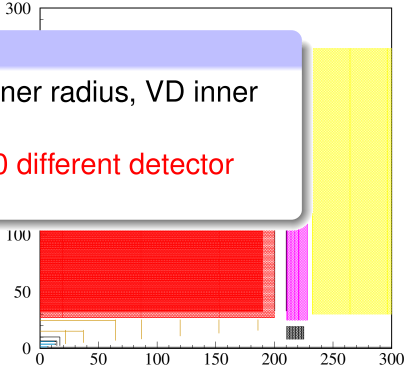
back

SGV for Detector optimisation

For the "IDR" detector optimisation effort, SGV was used to change the ILD baseline geometry in many ways. For each of these modifications in 5 steps were done:

1. Keep baseline aspect ratio.
2. Keep bas
3. Keep asp
4. Keep bas
5. Keep lengt = baseline-40 cm.

back



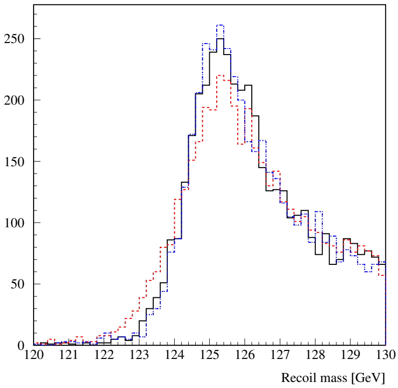
In addition:

Vary B-field, TPC inner radius, VD inner radius, ...

$2 \times 5 \times 5 \times 6 = 300$ different detector models !

Detector optimisation: The good, the bad, the ugly

- Higgs recoil-mass @350 GeV for the nominal ILD (black), the worst case (red) and the best case (blue)
- Simple b-tagger with two observables:
  - $\ln(-\ln(P(\text{All are primary})))$
  - Sum of 2:d to 5:th largest ip/σ_{ip}.
  - Optimise the cuts (on- the-fly) for best $S/\sqrt{S+B}$.
- $S/\sqrt{S+B}$ vs radius of inner layer of the VD



back

Detector optimisation: The good, the bad, the ugly

- Higgs recoil-mass @350 GeV for the nominal ILD (black), the worst case (red) and the best case (blue)
- Simple b-tagger with two observables:
  - $\ln(-\ln(P(\text{All are primary})))$
  - Sum of 2:d to 5:th largest ip/σ_{ip}.
  - Optimise the cuts (on- the-fly) for best $S/\sqrt{S+B}$.
- $S/\sqrt{S+B}$ vs radius of inner layer of the VD