Positron Track Reconstruction for LUXE using a Quantum Computer

Arianna Crippa[1], Lena Funcke[2], Tobias Hartung[3,6], Beate Heinemann[1,4], Karl Jansen[1,5],
Annabel Kropf [1,4], Stefan Kühn[6], Federico Meloni[1], David Spataro[1,4], Cenk Tüysüz[1,5], Yee Chinn Yap[1]

[1]Deutsches Elektron-Synchrotron DESY [2]Massachusetts Institute of Technology, MIT [3]University of Bath [4]Albert-Ludwigs-Universität Freiburg
[5]Humboldt-Universität zu Berlin [6]CaSToRC, The Cyprus Institute

Introduction.

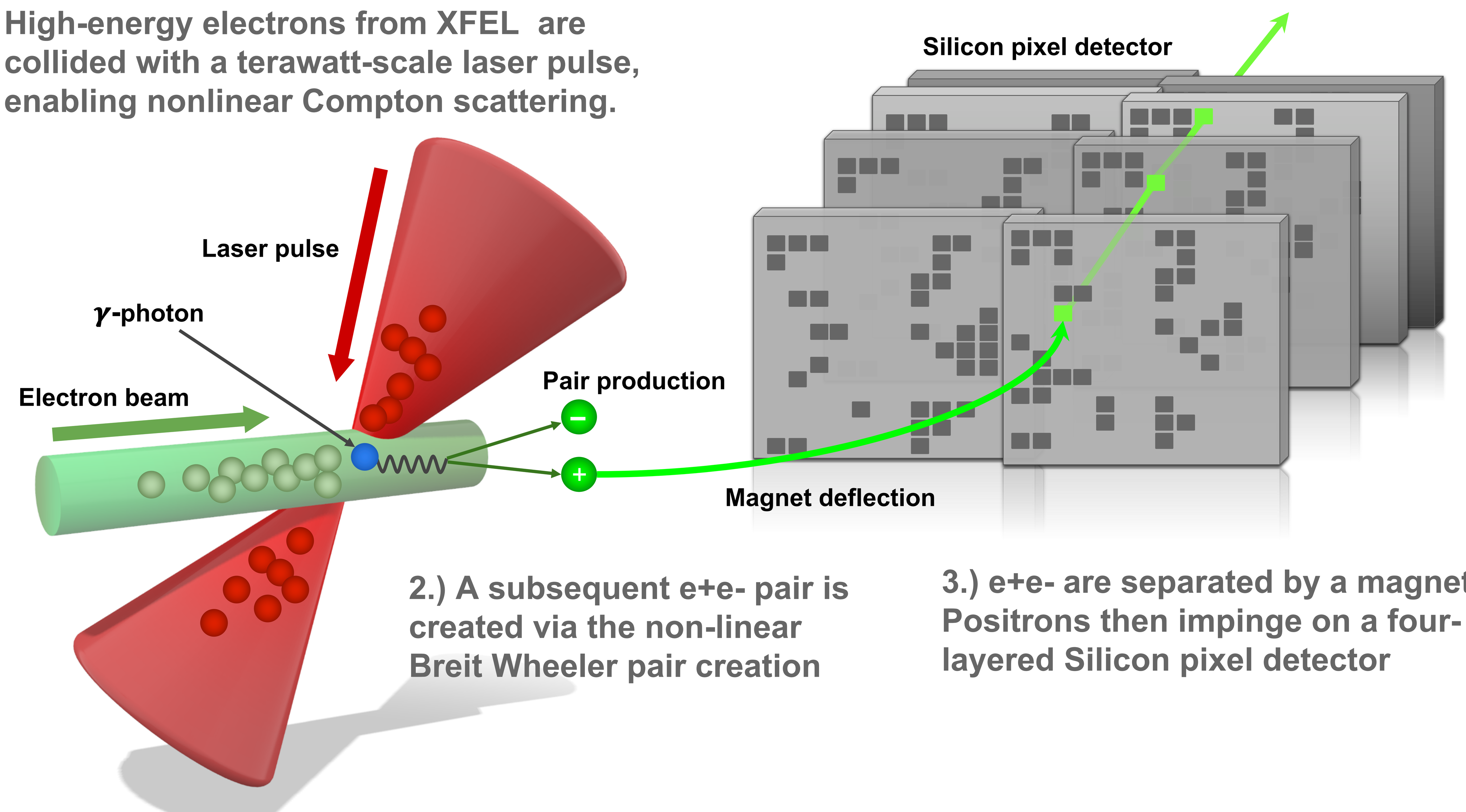
- LUXE (Laser Und XFEL Experiment) is a proposed experiment at DESY.
- The experiment's primary aim is to investigate the transition from the well-probed *perturbative into the non-perturbative regime* of QED that occurs at very high energies.
- One of the main goals is to measure the positron rate as a function of the laser intensity parameter ξ, defined as

$$\xi = \frac{m_e \epsilon_L}{\omega_L \epsilon_{cr}}$$

m_e : electron mass
ω_L : laser frequency
$\epsilon_{L,cr}$: laser/critical field strength

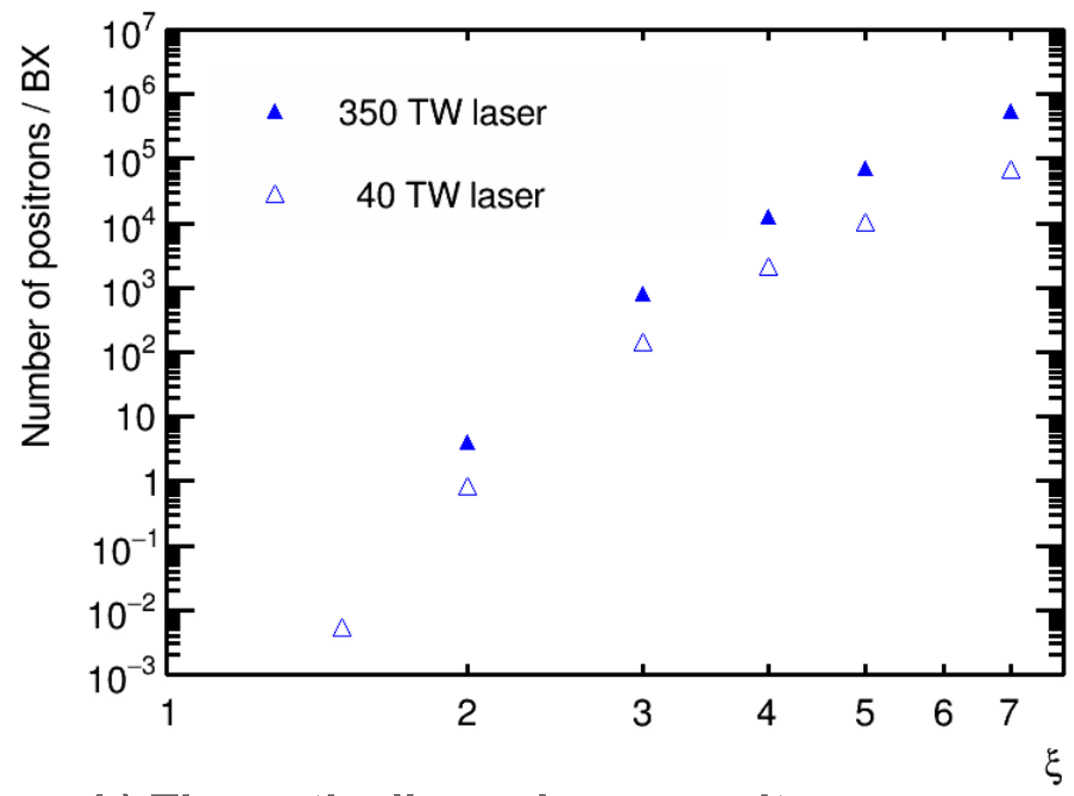
- The tracking problem can be formulated as a quadratic unconstrained binary optimization (QUBO), allowing the algorithm to be mapped onto a *quantum computer.*

1. High-energy electrons from XFEL are collided with a terawatt-scale laser pulse, enabling nonlinear Compton scattering.



2.) A subsequent e+e- pair is created via the non-linear Breit Wheeler pair creation

3.) e+e- are separated by a magnet. Positrons then impinge on a four-layered Silicon pixel detector



4.) Theoretically, a slower positron production rate is expected after the the critical field is reached.

Challenge. maintain good linearity up to high multiplicities, keep a low background rate below 10−3 per BX at low ξ

Goal. benchmark performance against classical methods using Graph Neural Network or a Combinatorial Kalman Filter.

Sample. Monte Carlo simulated event samples and a custom detector simulation

Minimizing the QUBO with a quantum algorithms returns the best set of triplets.

$$O(a,b,T) = \sum_{i=1}^{N} a_i T_i + \sum_{i}^{N} \sum_{j<i}^{N} b_{ij} T_i T_j \qquad T_i, T_j \in \{0,1\}$$

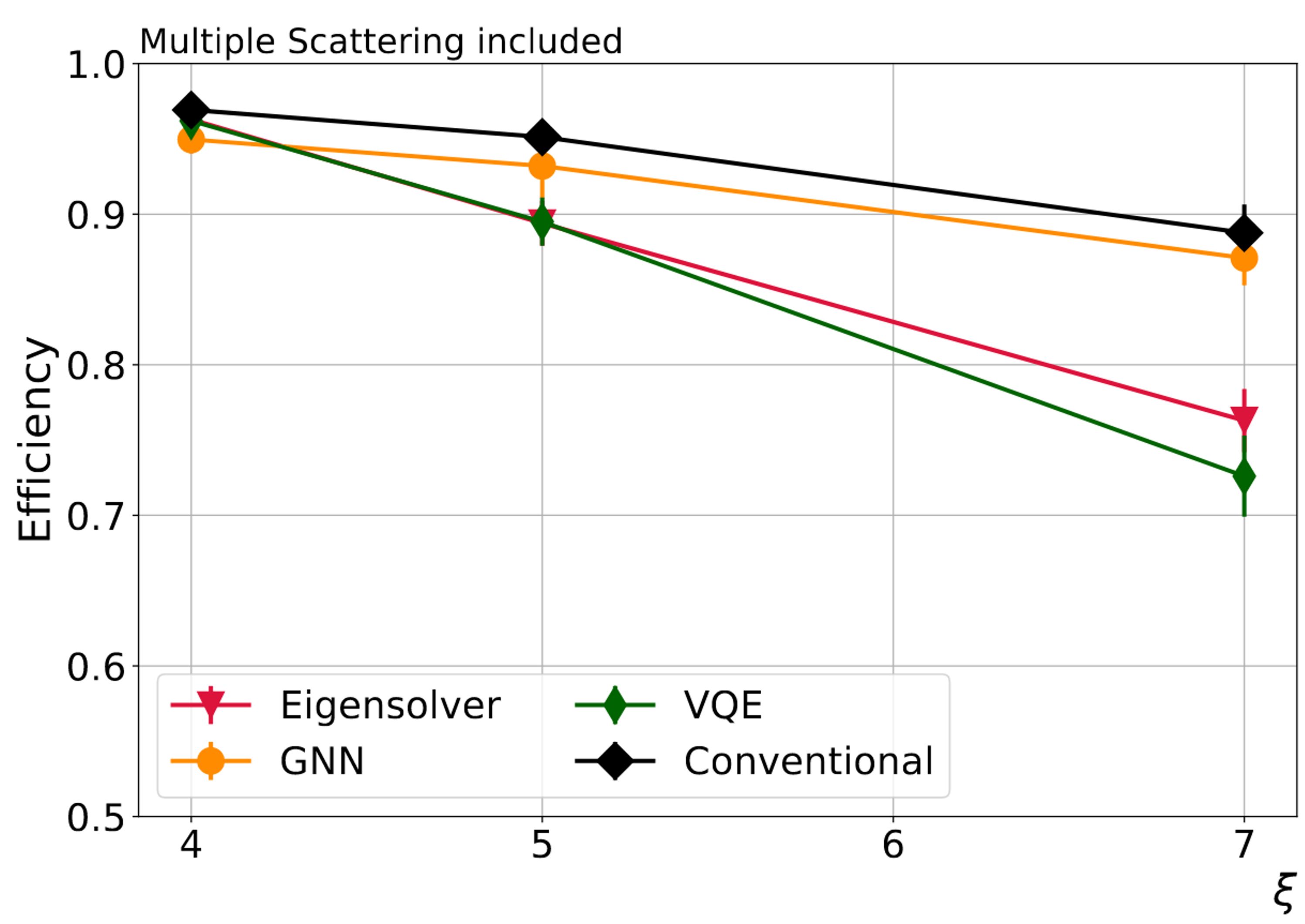
Weighting triplet T_i with quality a_i

Compatibility b_ij between two triplets

$$b_{ij} = \begin{cases} -S(T_i, T_j), & \text{if } (T_i, T_j) \text{ form a quadruplet,} \\ \zeta & \text{if } (T_i, T_j) \text{ are in conflict,} \\ 0 & \text{otherwise.} \end{cases}$$

Key questions.

- How does the performance depend on ξ?
- How does quantum noise affect the results?
- What quantum algorithm is optimal?
- What are the quantum computer requirements to run efficiently?
- How does the choice of quantum computer affect the results?



Max. fake rate:
VQE: 0.18 Eigensolver: 0.16 GNN: 0.07 Conventional: 0.03

Kinematic Fitting at Future e^+e^- Higgs Factories

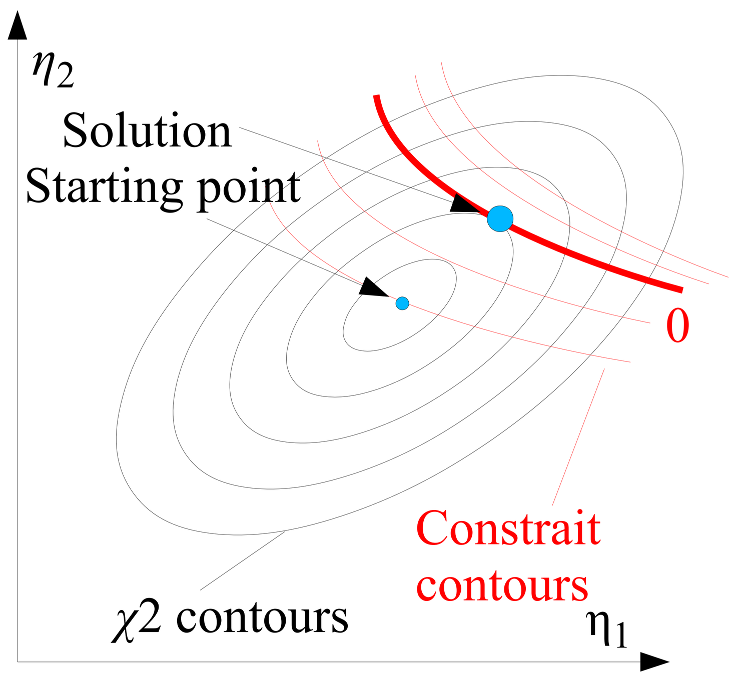
Benno List[1,2], Jenny List[1,2]
[1]Deutsches Elektron-Synchrotron DESY [2]Currently at CERN

Kinematically Constrained Fitting.

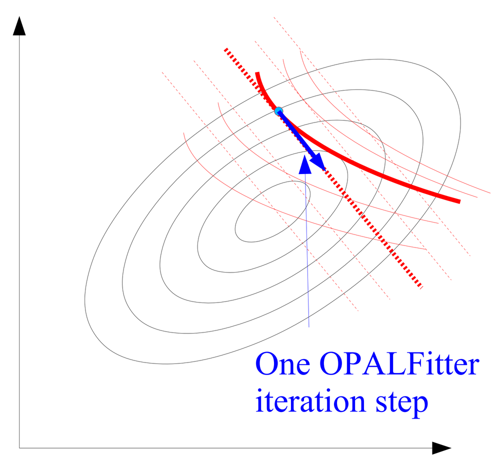
Lot of knowledge in e+e- events beyond the raw measurements:

- known four-momentum of the initial state, e.g. Σpy = 0 → hard constraint
- masses of intermediate particles, e.g. $M(jj) = M_H$ or M_Z → hard or soft constraint
- know which quantities are very well measured and which less so → error parametrisation

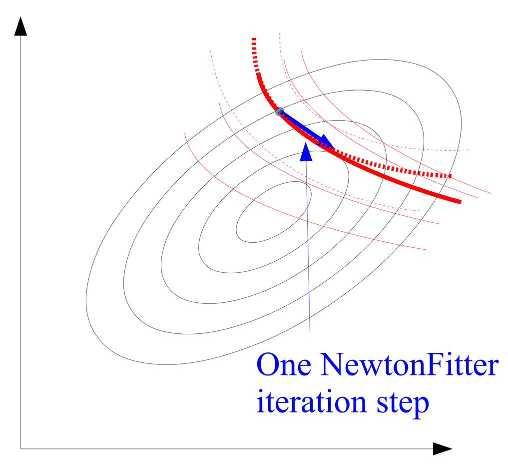
=> formulate hypothesis under which to interpret the event
=> test hypothesis by minimizing $\chi 2$ underconstraints by adjusting particle momenta



OPALFitter:
Approximates constraint by tangential plane



NewtonFitter:
Approximates constraint by tangential paraboloid



Method of Lagrange Multipliers

$$\chi^2_T(\vec{\eta},\vec{\xi},\vec{\lambda}) = (\vec{y}-\vec{\eta})^T \cdot V^{-1} \cdot (\vec{y}-\vec{\eta}) + 2\vec{\lambda}^T \cdot \vec{f}(\vec{\eta},\vec{\xi})$$

$$\begin{aligned} \nabla_\eta \chi^2_T &= -2V^{-1}\cdot(\vec{y}-\vec{\eta}) + 2\vec{F}^T_\eta \cdot \vec{\lambda} = \vec{0}, \\ \nabla_\xi \chi^2_T &= \vec{F}^T_\xi \cdot \vec{\lambda} = \vec{0}, \\ \nabla_\lambda \chi^2_T &= 2\vec{f}(\vec{\eta},\vec{\xi}) = \vec{0}, \end{aligned}$$

Exploit this to

- improve precision on observables, e.g. invariant masses
- determine unmeasured quantities (e.g. neutrino momentum)
- find best jet pairing
- select / reject events which match / don't hypothesis

Including ISR & Co.

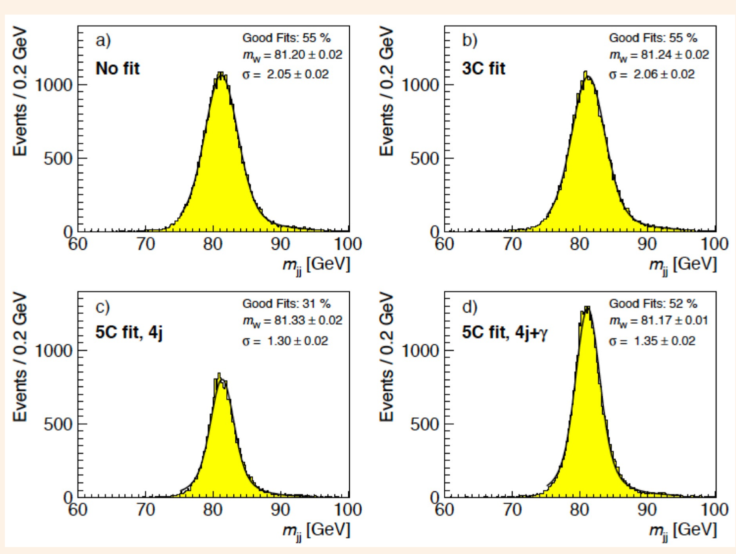
Additional FitObject with p_z as pseudo-measured parameter:

- "Measured" value = p_z balance
- "Error": σ of ISR spectrum transformed into a Gaussian

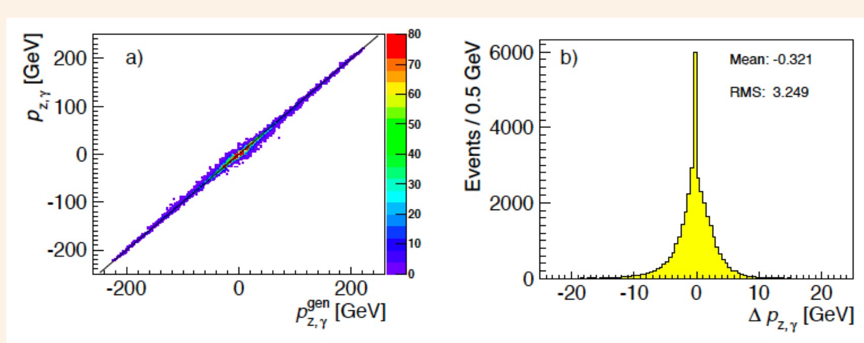
$$\mathcal{P}(p_{z,\gamma}) = \frac{\beta}{2E_{\max}} \cdot \left|\frac{p_{z,\gamma}}{E_{\max}}\right|^{\beta-1}$$

$$z = \mathrm{sign}(p_{z,\gamma}) \left(\frac{|p_{z,\gamma}|}{E_{\max}}\right)^\beta$$

$$\eta = \sqrt{2}\cdot \mathrm{erf}^{-1}(z)$$

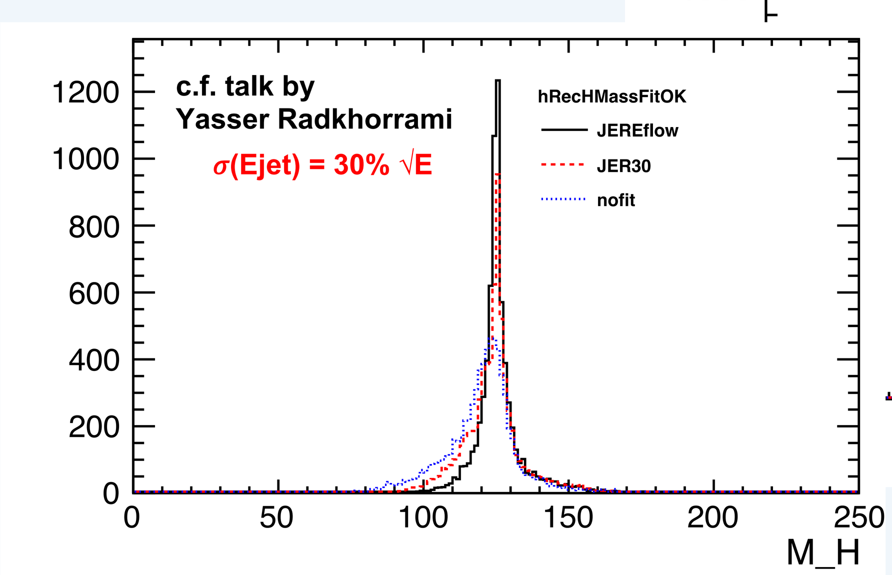


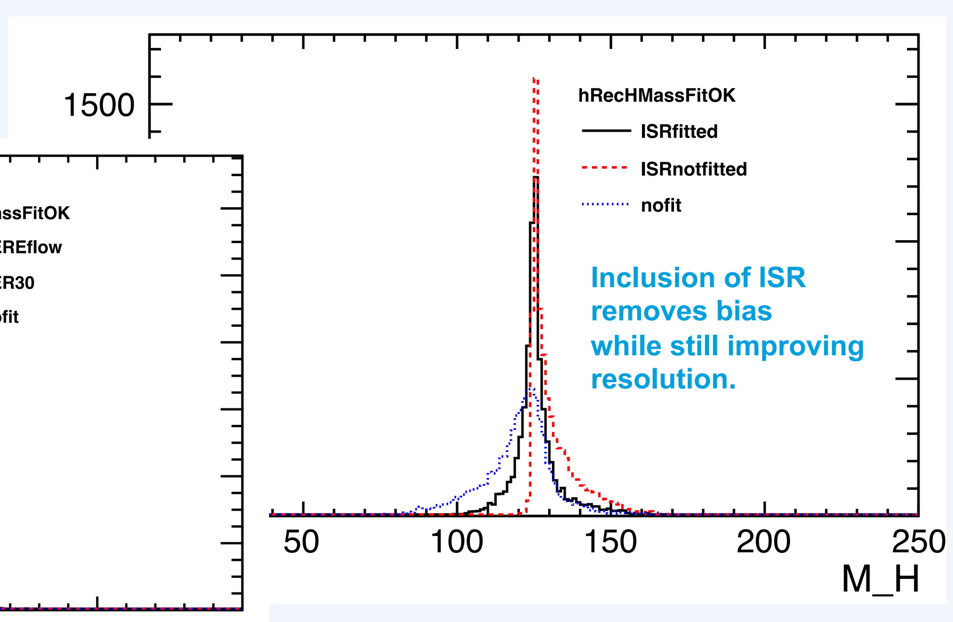
Quality of fitted photon p_z in WW->4j @ 500 GeV



Impact on Higgs reconstruction.

In ee → ZH → $\mu\mu$bb at 250 GeV





Software Implementation.

FitObject. Encapsulates all details of the parametrization, calculates its own contributions to global $\chi 2$ and its derivatives, calculates derivatives of 4-vector components wrt parameters.

Constraint. Calculates its value from 4-vectors of FitObjects and its derivatives wrt the 4-vector components of the FitObjects.

Fitter. Sets up and solves the system of equations, administers list of FitObjects and Constraints.

From FitObjects

$$\begin{pmatrix} V^{-1}\cdot\vec{y} \\ \vec{0} \\ -\vec{f}^\nu + F^\nu_\eta \vec{\eta}^\nu + F^\nu_\xi \cdot \vec{\xi}^\nu \end{pmatrix} = \begin{pmatrix} V^{-1} & 0 & (F^\nu_\eta)^T \\ 0 & 0 & (F^\nu_\xi)^T \\ F^\nu_\eta & F^\nu_\xi & 0 \end{pmatrix} \cdot \begin{pmatrix} \eta^{\nu+1} \\ \vec{\xi}^{\nu+1} \\ \vec{\lambda}^{\nu+1} \end{pmatrix}$$

From ConstraintObjects

Create FitObjects (2 jets)

Create Constraints: $\Sigma p_x = 0$, $\Sigma p_y = 0$, Invariant mass = 90GeV

Tell constraints over which FitObjects they should sum

Create the Fitter Engine

Tell the Fitter which Objects are to be fitted, and which Constraints are to be observed

Perform the Fit

```
//                     E    theta phi    dE dtheta dphi mass
JetFitObject jet1 (44., 1.2, 0.087, 5.0, 0.2,  0.1, 0.);
JetFitObject jet2 (46., 1.8, 3.120, 5.0, 0.2,  0.1, 0.);

// Constraint 0*sum(E) + 1*sum(px) + 0*sum(py) + 0*sum(pz) = 0
MomentumConstraint pxconstraint (0, 1, 0, 0, 0);
pxconstraint.addToFOList (jet1);
pxconstraint.addToFOList (jet2);

// Constraint 0*sum(E) + 0*sum(px) + 1*sum(py) + 0*sum(pz) = 0
MomentumConstraint pyconstraint (0, 0, 1, 0, 0);
pyconstraint.addToFOList (jet1);
pyconstraint.addToFOList (jet2);

// Constraint total mass = 90
MassConstraint mconstraint (90);
mconstraint.addToFOList (jet1);
mconstraint.addToFOList (jet2);

OPALFitter fitter;

fitter.addFitObject (jet1);
fitter.addFitObject (jet2);

fitter.addConstraint (pxconstraint);
fitter.addConstraint (pyconstraint);
fitter.addConstraint (mconstraint);

fitter.initialize();
double prob = fitter.fit();
```

MarlinKinfit. https://github.com/iLCSoft/MarlinKinfit
Example processors. https://github.com/iLCSoft/MarlinKinfitProcessors
Tutorial. https://github.com/ILDAnaSoft/MarlinKinfitTutorial

Next Steps.

- Transmit full ErrorFlow covariance matrix to FitObjects
- Implement correlations between FitObjects, e.g. to model jet clustering errors
- Optimisation of step length choice in NewtonFitter
- Fundamentally new minimizer, e.g. ML-based?
- Application to multi-jet analyses, e.g. ee → ZH, WW, tt, ZHH, …

contact:
jenny.list@desy.de

Learn more:

- M. Beckmann, B. List, J. List, Nucl.Instrum.Meth.A 624 (2010) 184-191, https://doi.org/10.1016/j.nima.2010.08.107
- B. List, J. List, LC-TOOL-2009-001, https://bib-pubdb1.desy.de/record/88030
- B. List, Constrained Fits, in Data Analysis in High Energy Physics: A Practical Guide to Statistical Methods, Wiley-VCH, ISBN 978-3527410583

HELMHOLTZ
RESEARCH FOR GRAND CHALLENGES