Core: Status of the Global χ^2 Fitter (GX2F)

Alexander J. Pfleger, University of Graz

9^{th} of November 2023

Contents

Timeline of the Development

Why use the GX2F?

Theory

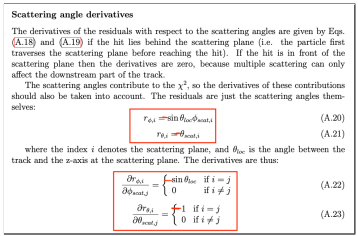
Implementation

Status & Future

Timeline of the GX2F

Previous work

- Implementation in ATLAS
- Thijs Gerrit Cornelissen, Track Fitting in the ATLAS Experiment (2006).
- Attempt to include in Acts



Scattering angle derivatives

The derivatives of the residuals with respect to the scattering angles are given by Eqs. (A.18) and (A.19) if the hit lies behind the scattering plane (i.e. the particle first traverses the scattering plane before reaching the hit). If the hit is in front of the scattering plane then the derivatives are zero, because multiple scattering can only affect the downstream part of the track.

The scattering angles contribute to the χ^2, so the derivatives of these contributions should also be taken into account. The residuals are just the scattering angles themselves:

$$r_{\phi,i} = \sin\theta_{\text{loc}}\phi_{\text{scat},i} \tag{A.20}$$

$$r_{\theta,i} = \theta_{\text{scat},i} \tag{A.21}$$

where the index i denotes the scattering plane, and θ_{loc} is the angle between the track and the z-axis at the scattering plane. The derivatives are thus:

$$\frac{\partial r_{\phi,i}}{\partial \phi_{\text{scat},j}} = \begin{cases} \sin\theta_{\text{loc}} & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.22}$$

$$\frac{\partial r_{\theta,i}}{\partial \theta_{\text{scat},j}} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.23}$$

Timeline of the GX2F

Previous work

- Implementation in ATLAS
- Thijs Gerrit Cornelissen, Track Fitting in the ATLAS Experiment (2006).
- Attempt to include in Acts

Phase 1

- Analytic solution up to 2D, straight line
- Deeper understanding of multiple scattering
- Python toy detector
- Ideal pull distributions

Scattering angle derivatives

The derivatives of the residuals with respect to the scattering angles are given by Eqs. (A.18) and (A.19) if the hit lies behind the scattering plane (i.e. the particle first traverses the scattering plane before reaching the hit). If the hit is in front of the scattering plane then the derivatives are zero, because multiple scattering can only affect the downstream part of the track.

The scattering angles contribute to the χ^2, so the derivatives of these contributions should also be taken into account. The residuals are just the scattering angles themselves:

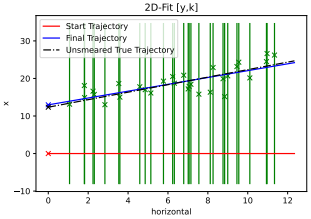
$$r_{\phi,i} = \sin\theta_{\text{loc}}\phi_{\text{scat},i} \tag{A.20}$$

$$r_{\theta,i} = \theta_{\text{scat},i} \tag{A.21}$$

where the index i denotes the scattering plane, and θ_{loc} is the angle between the track and the z-axis at the scattering plane. The derivatives are thus:

$$\frac{\partial r_{\phi,i}}{\partial \phi_{\text{scat},j}} = \begin{cases} \sin\theta_{\text{loc}} & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.22}$$

$$\frac{\partial r_{\theta,i}}{\partial \theta_{\text{scat},j}} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.23}$$



Timeline of the GX2F

Previous work

- Implementation in ATLAS
- Thijs Gerrit Cornelissen, Track Fitting in the ATLAS Experiment (2006).
- Attempt to include in Acts

Scattering angle derivatives

The derivatives of the residuals with respect to the scattering angles are given by Eqs. (A.18) and (A.19) if the hit lies behind the scattering plane (i.e. the particle first traverses the scattering plane before reaching the hit). If the hit is in front of the scattering plane then the derivatives are zero, because multiple scattering can only affect the downstream part of the track.

The scattering angles contribute to the χ^2, so the derivatives of these contributions should also be taken into account. The residuals are just the scattering angles themselves:

$$r_{\phi,i} = \sin\theta_{\text{loc}}\phi_{\text{scat},i} \tag{A.20}$$

$$r_{\theta,i} = \theta_{\text{scat},i} \tag{A.21}$$

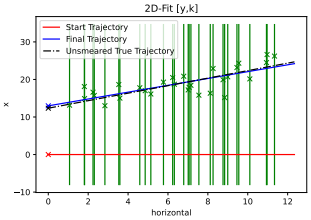
where the index i denotes the scattering plane, and θ_{loc} is the angle between the track and the z-axis at the scattering plane. The derivatives are thus:

$$\frac{\partial r_{\phi,i}}{\partial \phi_{\text{scat},j}} = \begin{cases} \sin\theta_{\text{loc}} & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.22}$$

$$\frac{\partial r_{\theta,i}}{\partial \theta_{\text{scat},j}} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.23}$$

Phase 1

- Analytic solution up to 2D, straight line
- Deeper understanding of multiple scattering
- Python toy detector
- Ideal pull distributions



Phase 2

- Framework in Acts for test driven development
- n-dimensional measurements
- Python bindings
- Magnetic field
- Test on ITk geometry

Timeline of the GX2F

Previous work

- Implementation in ATLAS
- Thijs Gerrit Cornelissen, Track Fitting in the ATLAS Experiment (2006).
- Attempt to include in Acts

Scattering angle derivatives

The derivatives of the residuals with respect to the scattering angles are given by Eqs. (A.18) and (A.19) if the hit lies behind the scattering plane (i.e. the particle first traverses the scattering plane before reaching the hit). If the hit is in front of the scattering plane then the derivatives are zero, because multiple scattering can only affect the downstream part of the track.

The scattering angles contribute to the χ^2, so the derivatives of these contributions should also be taken into account. The residuals are just the scattering angles themselves:

$$r_{\phi,i} = \sin\theta_{\text{loc}}\phi_{\text{scat},i} \tag{A.20}$$

$$r_{\theta,i} = \theta_{\text{scat},i} \tag{A.21}$$

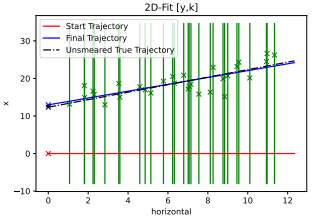
where the index i denotes the scattering plane, and θ_{loc} is the angle between the track and the z-axis at the scattering plane. The derivatives are thus:

$$\frac{\partial r_{\phi,i}}{\partial \phi_{\text{scat},j}} = \begin{cases} -\sin\theta_{\text{loc}} & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.22}$$

$$\frac{\partial r_{\theta,i}}{\partial \theta_{\text{scat},j}} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.23}$$

Phase 1

- Analytic solution up to 2D, straight line
- Deeper understanding of multiple scattering
- Python toy detector
- Ideal pull distributions



Phase 2

- Framework in Acts for test driven development
- n-dimensional measurements
- Python bindings
- Magnetic field
- Test on ITk geometry

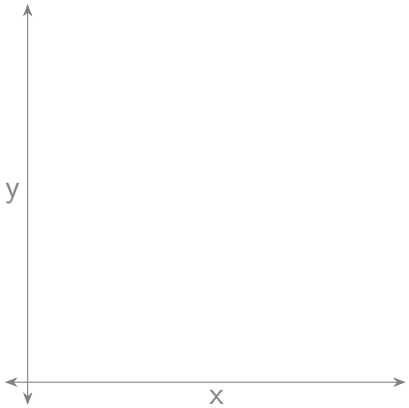
Phase 3 (future)

- Add material effects
- Athena integration
- Improve performance

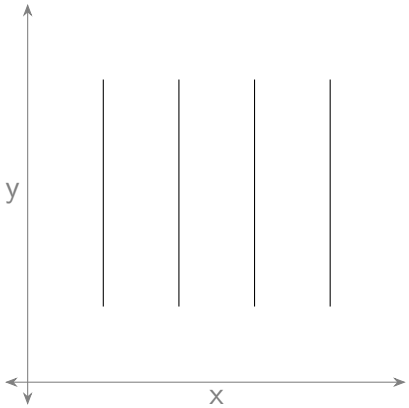
Why use the GX2F?

- Global fit has it easier with left-right-ambiguous measurements
  - Multiwire drift chambers – global fit could be stable
  - Alignment: sequential fits have trouble
  - ATLAS Muons (straws)
- Support TRT (Transition Radiation Tracker – straw tubes) in run-2/3
- Cross check different fitters

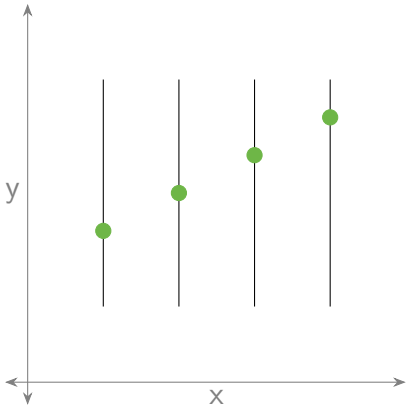
Theory – Getting Started



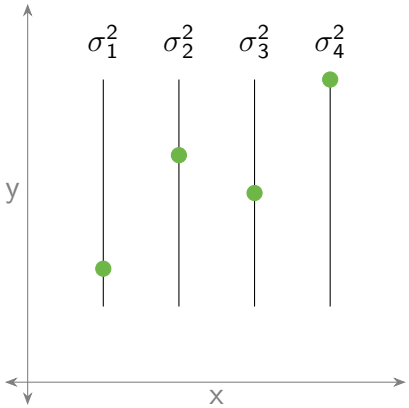
Theory – Getting Started



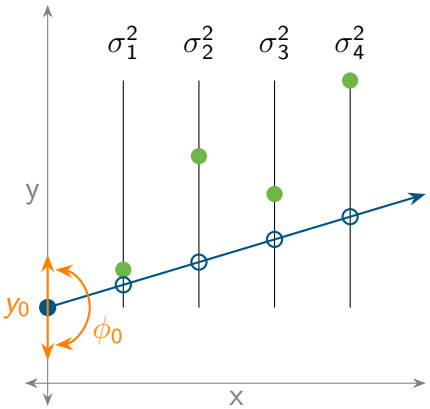
Theory – Getting Started



Theory – Getting Started



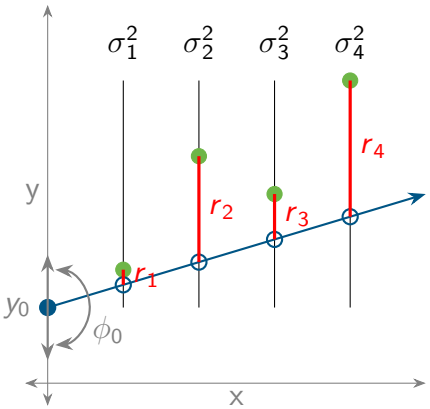
Theory – Getting Started



parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \end{pmatrix}$

propagation function: $f(\vec{\alpha}) = \begin{pmatrix} y_0 + \tan(\phi_0) \cdot x \\ \phi_0 \end{pmatrix}$

Theory – Getting Started

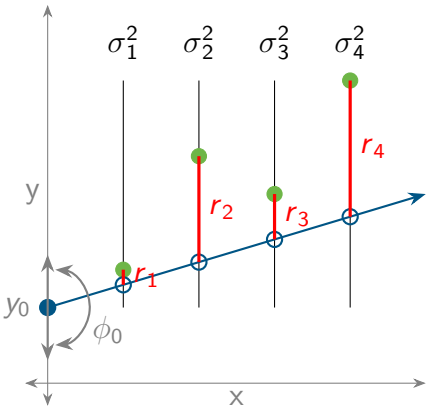


parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \end{pmatrix}$

propagation function: $f(\vec{\alpha}) = \begin{pmatrix} y_0 + \tan(\phi_0) \cdot x \\ \phi_0 \end{pmatrix}$

residuals: $r_i = m_i - f_i^m(\vec{\alpha})$

Theory – Getting Started

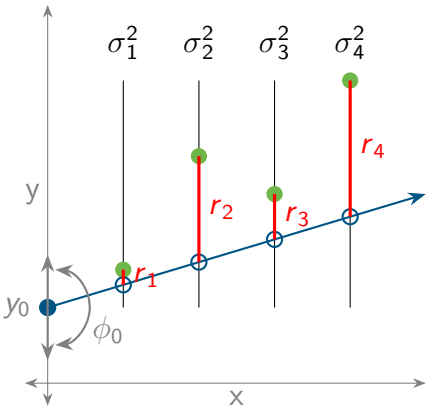


parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \end{pmatrix}$

propagation function: $f(\vec{\alpha}) = \begin{pmatrix} y_0 + \tan(\phi_0) \cdot x \\ \phi_0 \end{pmatrix}$

residuals: $r_i = m_i - f_i^m(\vec{\alpha}) = m_i - (y_0 + \tan(\phi_0) \cdot x)$

Theory – Getting Started



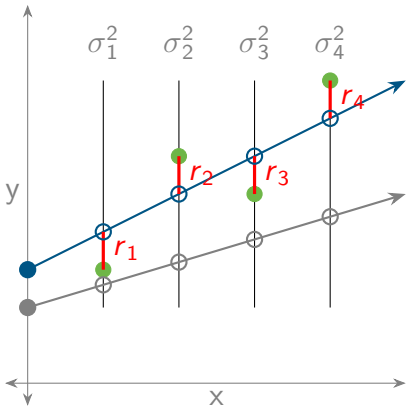
parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \end{pmatrix}$

propagation function: $f(\vec{\alpha}) = \begin{pmatrix} y_0 + \tan(\phi_0) \cdot x \\ \phi_0 \end{pmatrix}$

residuals: $r_i = m_i - f_i^m(\vec{\alpha}) = m_i - (y_0 + \tan(\phi_0) \cdot x)$

$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2}$$

Theory – Getting Started



parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \end{pmatrix}$

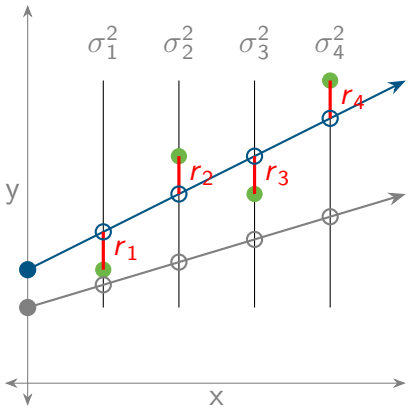
propagation function: $f(\vec{\alpha}) = \begin{pmatrix} y_0 + \tan(\phi_0) \cdot x \\ \phi_0 \end{pmatrix}$

residuals: $r_i = m_i - f_i^m(\vec{\alpha}) = m_i - (y_0 + \tan(\phi_0) \cdot x)$

$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2}$$

update: $\vec{\alpha}_{n+i} = \vec{\alpha}_n + \vec{\delta\alpha}_n$

Theory – Getting Started



parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \end{pmatrix}$

propagation function: $f(\vec{\alpha}) = \begin{pmatrix} y_0 + \tan(\phi_0) \cdot x \\ \phi_0 \end{pmatrix}$

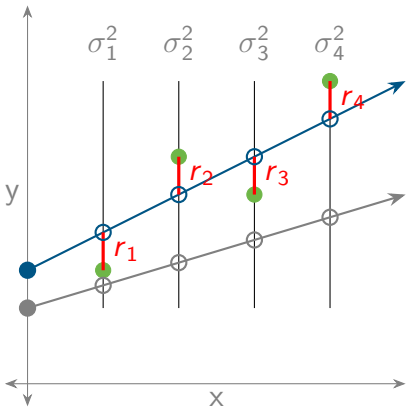
residuals: $r_i = m_i - f_i^m(\vec{\alpha}) = m_i - (y_0 + \tan(\phi_0) \cdot x)$

$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2}$$

update: $\vec{\alpha}_{n+i} = \vec{\alpha}_n + \delta\vec{\alpha}_n$

$$[a_{kl}]\delta\vec{\alpha}_n = \vec{b}$$

Theory – Update

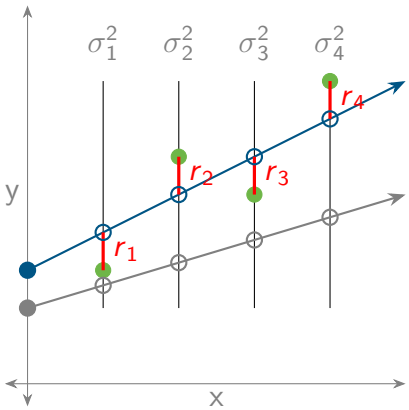


$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2}$$

update: $\vec{\alpha}_{n+i} = \vec{\alpha}_n + \vec{\delta\alpha}_n$

$$[a_{kl}]\vec{\delta\alpha}_n = \vec{b}$$

Theory – Update



$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2}$$

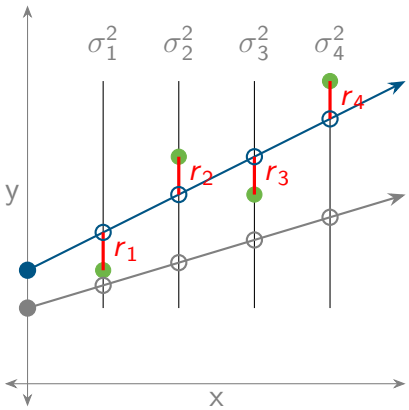
update: $\vec{\alpha}_{n+i} = \vec{\alpha}_n + \vec{\delta\alpha}_n$

$$[a_{kl}]\vec{\delta\alpha}_n = \vec{b}$$

$$a_{kl} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_k} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_l}$$

$$b_k = \sum_{i=1}^{N} \frac{r_i}{\sigma_i^2} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_k}$$

Theory – Update



$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2}$$

update: $\vec{\alpha}_{n+i} = \vec{\alpha}_n + \delta\vec{\alpha}_n$

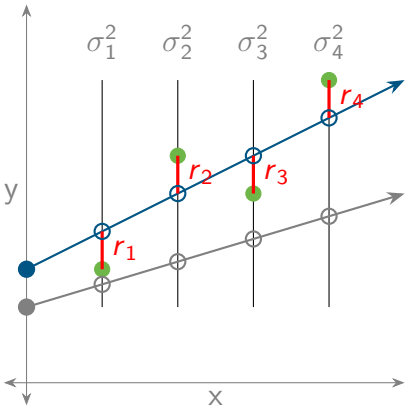
$$[a_{kl}]\delta\vec{\alpha}_n = \vec{b}$$

$$a_{kl} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_k} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_l}$$

$$\mathbf{J} = \begin{pmatrix} \cdot & \cdots & \cdot \\ \vdots & \frac{\partial f^m(\vec{\alpha})}{\partial \alpha_k} & \vdots \\ \cdot & \cdots & \cdot \end{pmatrix}$$

$$b_k = \sum_{i=1}^{N} \frac{r_i}{\sigma_i^2} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_k}$$

Theory – Update



$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2}$$

update: $\vec{\alpha}_{n+i} = \vec{\alpha}_n + \delta\vec{\alpha}_n$

$$[a_{kl}]\delta\vec{\alpha}_n = \vec{b}$$

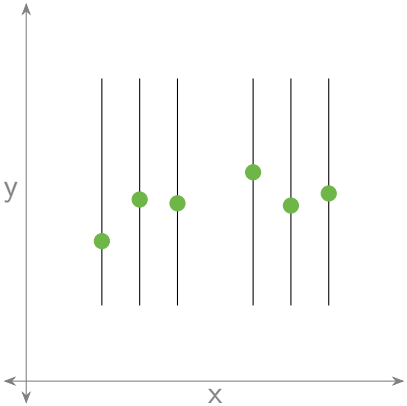
$$a_{kl} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_k} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_l}$$

$$\mathbf{J} = \begin{pmatrix} \cdot & \cdots & \cdot \\ \vdots & \frac{\partial f^m(\vec{\alpha})}{\partial \alpha_k} & \vdots \\ \cdot & \cdots & \cdot \end{pmatrix}$$

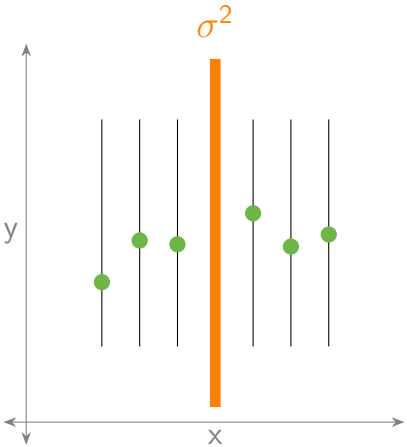
$$b_k = \sum_{i=1}^{N} \frac{r_i}{\sigma_i^2} \frac{\partial f_i^m(\vec{\alpha})}{\partial \alpha_k}$$

$$cov_{\vec{\alpha}} = [a_{kl}]^{-1}$$

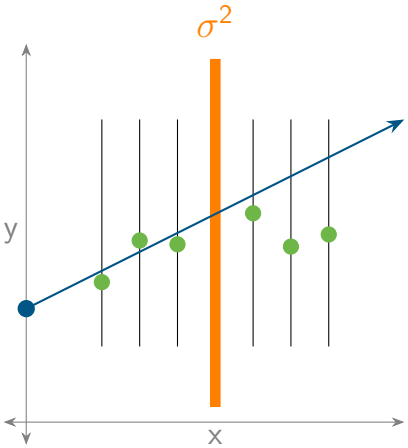
Theory – Scattering



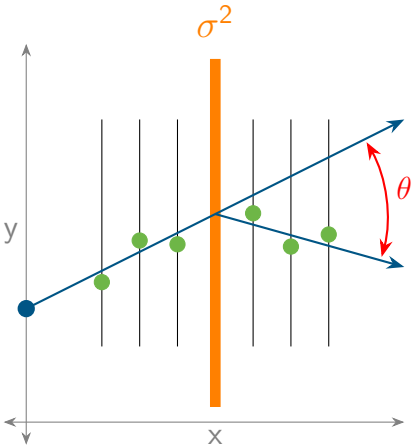
Theory – Scattering



Theory – Scattering

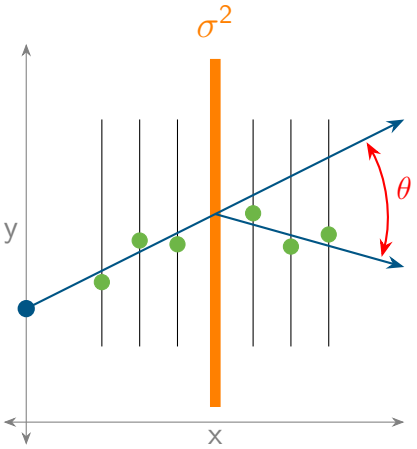


Theory – Scattering



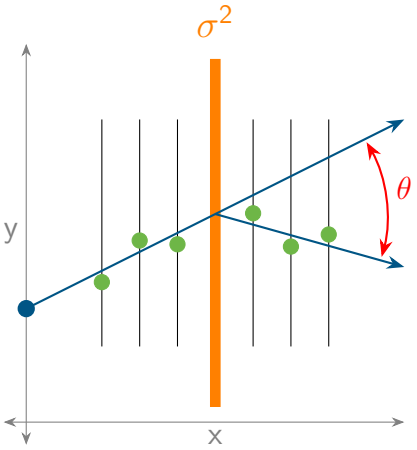
parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \\ \theta_{s1} \\ \\ \vdots \end{pmatrix}$

Theory – Scattering



parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \\ \theta_{s1} \\ \phi_{s1} \\ \vdots \end{pmatrix}$

$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2} + \sum_{s}^{S} \left(\frac{\theta_s^2}{\sigma_s^2} + \underbrace{\frac{\sin^2(\theta_{loc})\phi_s^2}{\sigma_s^2}}_{3D} \right)$$



parameters: $\vec{\alpha} = \begin{pmatrix} y_0 \\ \phi_0 \\ \theta_{s1} \\ \phi_{s1} \\ \vdots \end{pmatrix}$

$$\chi^2 = \sum_{i=1}^{N} \frac{r_i^2}{\sigma_i^2} + \sum_{s}^{S} \left(\frac{\theta_s^2}{\sigma_s^2} + \underbrace{\frac{\sin^2(\theta_{loc})\phi_s^2}{\sigma_s^2}}_{3D} \right)$$

residuals: $r_s = \begin{cases} 0 - \theta_s(\vec{\alpha}) \\ 0 - \sin(\theta_{loc})\phi_s(\vec{\alpha}) \end{cases}$

Implementation – Configuration

```cpp
template <typename traj_t>
struct Gx2FitterOptions {
  Gx2FitterOptions( ... ) : ... {}

  Gx2FitterOptions() = delete;

  ... //common options: geoContext, magFieldContext, calibrationContext,
    extensions, propagatorPlainOptions, referenceSurface, multipleScattering,
    energyLoss, freeToBoundCorrection

  /// Max number of iterations during the fit (abort condition)
  size_t nUpdateMax = 5;

  /// Disables the QoP fit in case of missing B-field
  bool zeroField = false;

  /// Check for convergence (abort condition). Set to 0 to skip.
  double relChi2changeCutOff = 1e-7;
};
```

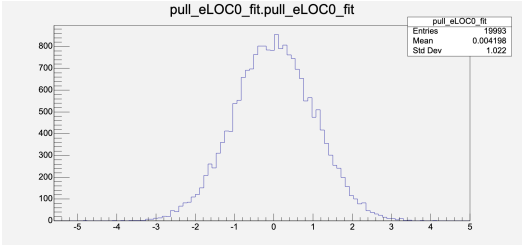
Implementation – Abort Conditions

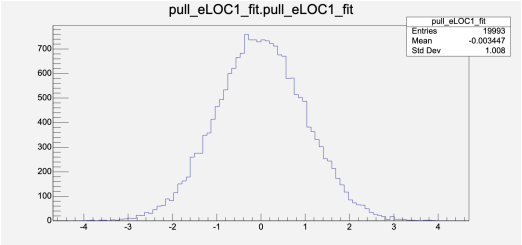
```
for (size_t nUpdate = 0; nUpdate < nUpdateMax; nUpdate++) {
  params.parameters() += deltaParams; // update parameters
  ... // set up propagator, set options -> propagate

  for (size_t iMeas = 0; iMeas < result.collectorResiduals.size(); iMeas++) {
    ... // get ri, covi, projectedJacobian
    ... // calculate chi2meas, aMatrixMeas, bVectorMeas
    ... // add to sums chi2sum, aMatrix, bVector
  }

  deltaParams = ... // [a] * delta = b

  if ((relChi2changeCutOff != 0) && (nUpdate > 0) &&
      (std::abs(chi2sum / oldChi2sum - 1) < relChi2changeCutOff)) {
    break;
  }
...
}
```

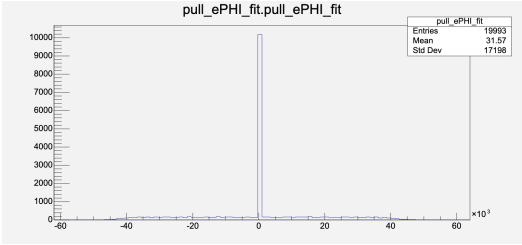
Implementation – Measurement Collection

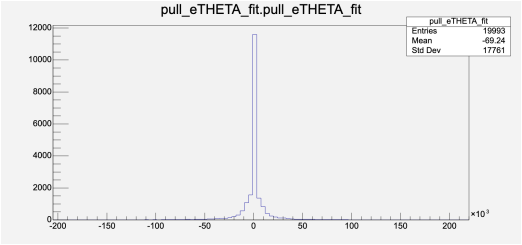
```cpp
template <size_t measDim, typename traj_t>
void collector(typename traj_t::TrackStateProxy& trackStateProxy,
               Gx2FitterResult<traj_t>& result, const Logger& logger) {
  auto predicted = trackStateProxy.predicted();
  auto measurement = trackStateProxy.template calibrated<measDim>();
  auto covarianceMeasurement =
      trackStateProxy.template calibratedCovariance<measDim>();
  // Project Jacobian and predicted measurements into the measurement dims
  auto projJacobian = ...; // projector()<measDim, 6>() * jacobianFromStart
  auto projPredicted = ...; // projector()<measDim, 6>() * predicted

  // Collect residuals, covariances, and projected jacobians
  for (size_t i = 0; i < measDim; i++) {
    result.collectorResiduals.push_back(measurement[i] - projPredicted[i]);
    result.collectorCovariances.push_back(covarianceMeasurement(i, i));
    result.collectorProjectedJacobians.push_back(projJacobian.row(i));
  }
}
```

Status & Future – Generic Detector – Pulls









Status & Future

- Add material effects
- Improve actor (holes, performance, ...)
- ODD
- Athena integration
- Improve overall performance
- Energy loss
- n-dimensional measurements
- Weights for $\vec{\delta\alpha}$ for faster convergence