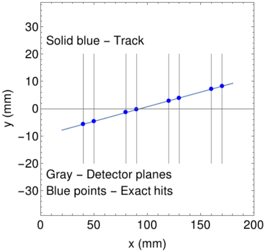
CLAS12, Track-Based, SVT Alignment with Millepede

G.P. Gilfoyle

Outline: The problem
A toy model
Basic idea
Status

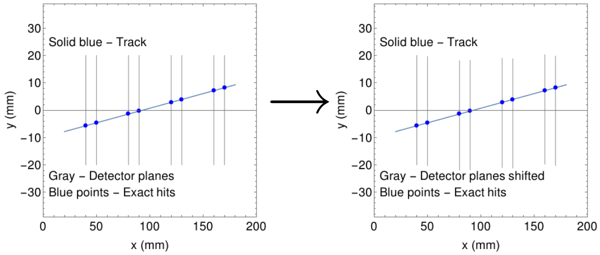
The Problem



Toy model:

- Straight tracks.
- Planar detectors.
- Shift detectors only in y.

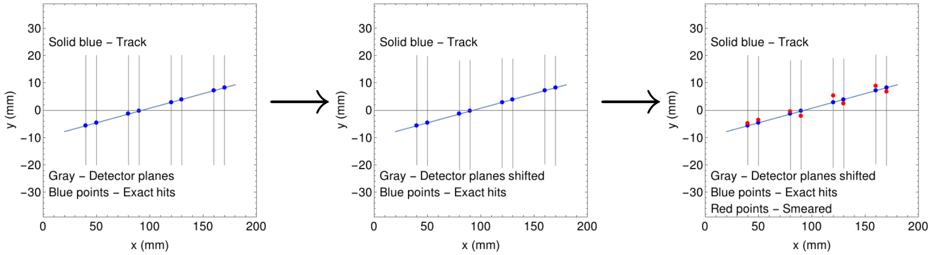
The Problem



Toy model:

- Straight tracks.
- Planar detectors.
- Shift detectors only in y.

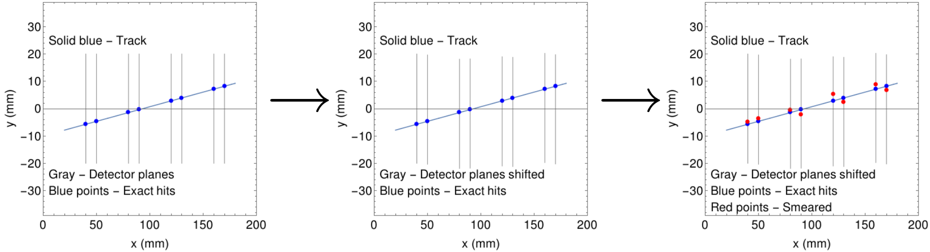
The Problem



Toy model:

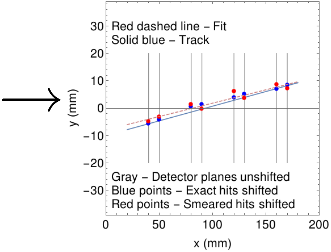
- Straight tracks.
- Planar detectors.
- Shift detectors only in y.

The Problem



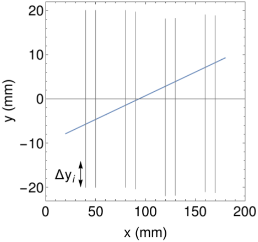
Toy model:

- Straight tracks.
- Planar detectors.
- Shift detectors only in y.



millepede: Linear Least Squares with Many Parameters

- In some least squares fit problems with many parameters those parameters can be divided into two classes.
  - Global - *i.e.* geometry.
  - Local - only present in subsets of the data, *i.e.* slope of a track.
- The code uses methods to solve the linear least square problem, irrespective of the number of local parameters.
- Up to ten thousand global parameters can be fitted.
- A simple test case:



Typically we fit the track with $y(x) = a + bx$.

In `millepede` use

$$y_{fit} = f(x, \vec{q}, \vec{p}) = \underbrace{\Delta y_1 + \Delta y_2 + ... + \Delta y_8}_{\text{global}, \ \vec{p}} + \underbrace{a + bx}_{\text{local}, \ \vec{q}} \quad .$$

Assume the initial fit with $\Delta y_i = 0$ is close to the final one so you can use the partial derivatives.

$$\frac{\partial z}{\partial \Delta y_i} = 1 \quad \frac{\partial z}{\partial a} = 1 \quad \frac{\partial z}{\partial b} = x$$

And use the residual $z = y_{meas} - f(x, \vec{q}, \vec{p})$.

CLAS12 `millepede`: Status

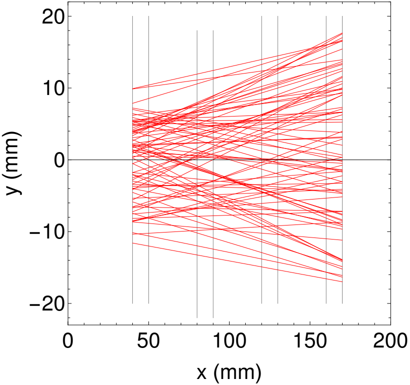
- Use the toy model described above as a tutorial.
- The code is running on the farm thanks to Mike Staib (CMU).
- Being used for HPS (Pelle Hansson and Alessandre Filippe) and GlueX (Mike Staib).
- A `millepede` event for the toy model.

Label	Measurement (mm)	Uncertainty (mm)	local derivatives		global derivatives
1	0.4378	1.4250	1.0	60.0	1.0
		...			
i	z_i	σ_i	1.0	x_i	1.0

- `myMille`
  - Running `millepede` requires two stages - (1) prepare a binary file with the data and (2) run the code that does the fitting called `pede`.
  - A C^{++} code to create the input binary called `myMille` has been written and tested with local tools.
- The code `pede` runs, reads the binary input file, and with the 'proper' constraints appears to work - thanks to Mike Staib, Alessandre Filippe, and Pele Hansson.

CLAS12 millepede: First Tests with millepede

- Generate a sample of straight tracks in our 2D, toy detector model.
- Limit the sample to events with hits in all eight detector planes.
- Generated over 26,000 events that satisfy this criteria.



CLASS12 `millepede`: Results

- Test 1
  - Set $\Delta y_i = 0.0$ for all detector planes, simulate 26,000 straight tracks.
  - Fix $\Delta y_i = 0$ for planes 1-2 and 5-8 and let Δy_i vary for planes 3-4.
  - Simulation Input: `millepede` Output:

$$\Delta y_3 = 0.0\ mm \quad \longrightarrow \quad \Delta y_3 = -0.0041 \pm 0.0067\ mm$$

$$\Delta y_4 = 0.0\ mm \qquad\qquad \Delta y_4 = -0.0018 \pm 0.0066\ mm$$

CLAS12 millepede: Results

- Test 1
  - Set $\Delta y_i = 0.0$ for all detector planes, simulate 26,000 straight tracks.
  - Fix $\Delta y_i = 0$ for planes 1-2 and 5-8 and let Δy_i vary for planes 3-4.
  - Simulation Input: → millepede Output:

$$\Delta y_3 = 0.0\ mm \longrightarrow \Delta y_3 = -0.0041 \pm 0.0067\ mm$$

$$\Delta y_4 = 0.0\ mm \qquad \Delta y_4 = -0.0018 \pm 0.0066\ mm$$

- Test 2
  - Set $\Delta y_i = 0.0$ for planes 1-2, 5-8, $\Delta y_i = -2.0\ mm$ for planes 3-4, and simulate 26,000 straight tracks.
  - Fix $\Delta y_i = 0$ for planes 1-2 and 5-8 and let Δy_i vary for planes 3-4.
  - Simulation Input: → millepede Output:

$$\Delta y_3 = -2.0\ mm \longrightarrow \Delta y_3 = -1.999 \pm 0.007\ mm$$

$$\Delta y_4 = -2.0\ mm \qquad \Delta y_4 = -2.012 \pm 0.007\ mm$$

CLAS12 `millepede`: Results

- Test 1
  - Set $\Delta y_i = 0.0$ for all detector planes, simulate 26,000 straight tracks.
  - Fix $\Delta y_i = 0$ for planes 1-2 and 5-8 and let Δy_i vary for planes 3-4.
  - Simulation Input: `millepede` Output:

$$\Delta y_3 = 0.0\ mm \quad \longrightarrow \quad \Delta y_3 = -0.0041 \pm 0.0067\ mm$$

$$\Delta y_4 = 0.0\ mm \qquad\qquad \Delta y_4 = -0.0018 \pm 0.0066\ mm$$

- Test 2
  - Set $\Delta y_i = 0.0$ for planes 1-2, 5-8, $\Delta y_i = -2.0\ mm$ for planes 3-4, and simulate 26,000 straight tracks.
  - Fix $\Delta y_i = 0$ for planes 1-2 and 5-8 and let Δy_i vary for planes 3-4.
  - Simulation Input: `millepede` Output:

$$\Delta y_3 = -2.0\ mm \quad \longrightarrow \quad \Delta y_3 = -1.999 \pm 0.007\ mm$$

$$\Delta y_4 = -2.0\ mm \qquad\qquad \Delta y_4 = -2.012 \pm 0.007\ mm$$

- Issues: Effect of number of constraints, rank defect, deciphering `millepede` output...