Alignment of the Silicon Vertex Tracker (SVT)

- Alignment of the SVT requires finding small shifts in sensor position and orientation.
  - Number of parameters = $N_{regions}$ x N_{layers} x N_{trans} x N_{rot} = 66 x 2 x 3 x 2 = 792 parameters. BIG!
  - Other experiments face daunting challenges – CMS silicon tracker at LHC has about 50,000 parameters.
- Program **millepede** (yes, that's how it's spelled) does linear least squares fitting with many parameters.
  - Use matrix formulation of the least squares method and divide the matrix elements into two classes.
    - Global parameters – the geometry. 'Same' values in all the events.
    - Local – track fit parameters. Change from event-to-event.
  - Calculate partial derivatives of the fit residuals with respect to local (i.e. fit) parameters and global ones (geometry) – C++ program **mille**.
  - Isolate the global parameters (geometry) and invert – Fortran code **pede**.
- A simple example.
  - Two-dimensional 'detector' with eight planes (Fig. 1).
  - The fit equation is

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

  - Excellent fit results (Fig. 2) if the constraints are well measured (i.e. survey data).
  - Results sensitive to accuracy of constraints.

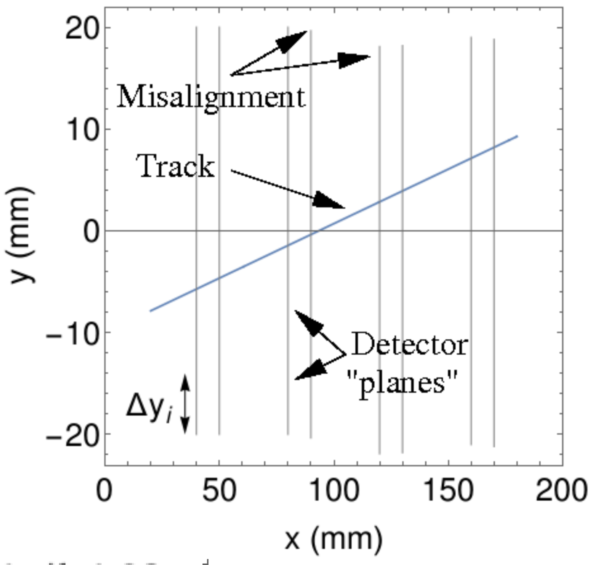

Figure 1

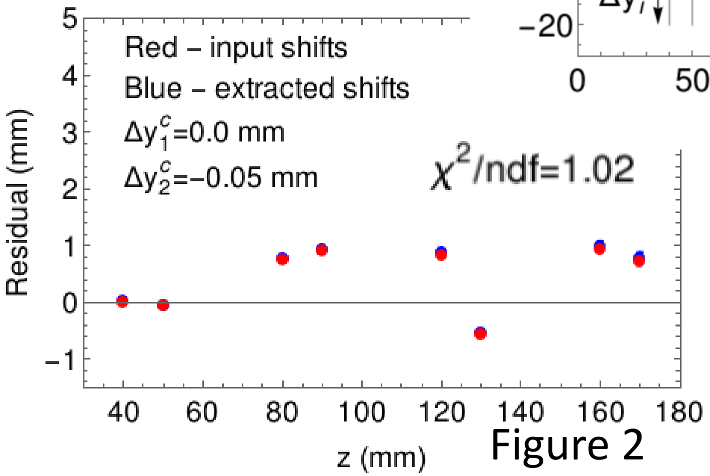

Figure 2

Alignment of the Silicon Vertex Tracker (SVT)

- Status of **millepede** at JLab.
  - Program built and running on the JLab farm by Mike Staib (CMU).
  - Used for alignment of GlueX CDC (Staib) and HPS tracker (Phillipe and Hansson).
- Introduction to millipede using 2D model above – 'easy' to understand, interpret results.
- Developed simple 3D model for testing.
- **mille** code for Type 1 tracks in SVT.
  - Code written, now testing.
  - Use Java common tools to extract alignment data for millipede - track, layer, sector, strip, DOCA, geometry.
  - C++ code extracts DOCA from geometry.
    - ➢ Consistent with coatjava results.
    - ➢ Geometry test – new ced option.
  - First derivative calculations done.
  - Code will produce input data for **pede**.

$$\frac{\partial DOCA_x}{\partial xshiftL} =$$

$$\begin{aligned}
&-1 - \Bigg((-x0w + x1w) \Bigg((x0w - x1w) \left(1 + \frac{(-x0t + x1t)^2}{z1t^2} + \frac{(-y0t + y1t)^2}{z1t^2}\right) + \\
&\qquad \frac{(-x0t + x1t) \left(-z0w + \frac{(-x0t+x1t)\,(-x0w+x1w)}{z1t} + \frac{(-y0t+y1t)\,(-y0w+y1w)}{z1t} + z1w\right)}{z1t} \Bigg)\Bigg) \Bigg/ \\
&\Bigg(-\left(-z0w + \frac{(-x0t + x1t)\,(-x0w + x1w)}{z1t} + \frac{(-y0t + y1t)\,(-y0w + y1w)}{z1t} + z1w\right)^2 + \\
&\qquad \left(1 + \frac{(-x0t + x1t)^2}{z1t^2} + \frac{(-y0t + y1t)^2}{z1t^2}\right) \left((-x0w + x1w)^2 + (-y0w + y1w)^2 + (-z0w + z1w)^2\right) \Bigg) + \\
&\Bigg((-x0t + x1t) \Bigg((x0w - x1w) \Bigg(-z0w + \frac{(-x0t + x1t)\,(-x0w + x1w)}{z1t} + \frac{(-y0t + y1t)\,(-y0w + y1w)}{z1t} + \\
&\qquad z1w \Bigg) + \frac{(-x0t + x1t)\,\left((-x0w + x1w)^2 + (-y0w + y1w)^2 + (-z0w + z1w)^2\right)}{z1t} \Bigg)\Bigg) \Bigg/ \\
&\Bigg(z1t \Bigg(-\left(-z0w + \frac{(-x0t + x1t)\,(-x0w + x1w)}{z1t} + \frac{(-y0t + y1t)\,(-y0w + y1w)}{z1t} + z1w\right)^2 + \\
&\qquad \left(1 + \frac{(-x0t + x1t)^2}{z1t^2} + \frac{(-y0t + y1t)^2}{z1t^2}\right) \left((-x0w + x1w)^2 + (-y0w + y1w)^2 + (-z0w + z1w)^2\right) \Bigg)\Bigg)
\end{aligned}$$

Strip end points: $(x0w, y0w, z0w)$, $(x1w, y1x, z1w)$
Track fit end points: $(x0t, y0t, z0t)$, $(x1t, y1t, z1t)$
Geometry parameters: $xshiftL$