Merlin

Nick Walker
DESY
LC02 5th February, 2002
Introduction

• Briefly explain what Merlin is and what it can do
• Make clear the tracking algorithms used, so comparisons with other codes can be made
• Show some pretty results
• Explain where we want to go
Merlin in a Nutshell

- Merlin is a C++ class library for doing charged particle accelerator simulations
- Current library has >300 classes
- Has a long (and dubious) heritage (APTkit, CLASSIC)
- Originally designed to study ground motion effects in BDS
- Now been extended to model, well, (almost) everything 😊
- More info: http://www.desy.de/~Merlin

Don’t Panic – this is not a talk about C++
The Accelerator Model

• Supported Standard Components
  – Drifts, Dipoles, Quads, Sextupoles, Octupoles
  – BPM, Profile Monitor (Wire scanner)
  – Solenoid
  – RF acceleration (SW and TW structures)
  – X and Y corrector dipoles
  – X-Y corrector windings
    (can be added to any multipole magnet)

• Non-Beamline Components
  – Magnet movers, Magnet Supports, Girders

The Component Library is always growing…
The Accelerator Model

• All Accelerator Components have:
  – An E-M field (Tesla, volts/meter)
  – A physical aperture
    [circular and rectangular currently supported]
  – An accelerator geometry
    [responsible for alignment, coordinate frame transformations etc.]
  – Most support ‘channels’ [see later]
Particle Tracking Module

• 6-d particle tracking (ray tracing):

\[ x_i \in \{x, x', y, y', ct, \delta = \Delta p/p_0\} \]

• Particles assumed relativistic (\(\beta = 1\))

• Tracking uses 6-d second-order TRANSPORT maps up to sextupole:

\[ x_i = R_{ij} x_j + T_{ijk} x_j x_k \]

• Higher-order multipoles modelled as chromatic thin-lens kicks at centre of element
Beam Energy and Tracking

- **B** fields stored (not \( K_n \))
- Particle bunch carries its own *reference* momentum (\( P_{\text{ref}} \))
- Particle \( \delta_i \) referenced to \( P_{\text{ref}} \)
- \( P_{\text{ref}} \) used to calculate map \(( R + T )\)

Note: \( \langle P_i \rangle = P_{\text{ref}} (1 + \langle \delta \rangle ) \)
Beam Energy and Tracking

- Special case: **Sector Bend**
- $P_0$ for $(R + T)$ taken from bend curvature and field: $P_0 = \frac{e c B}{h}$
- $\delta_i$ are scaled accordingly:

$$\delta_i \rightarrow \frac{P_{\text{ref}}}{P_0} (1 + \delta_i) - 1$$

- Fixed geometry ($h = \text{const}$)
- Changing $B$ or $P_{\text{ref}}$ changes orbit

Note: full second-order map for mixed function magnet plus pole face rotation and curvature included
Beam Energy and Tracking

- Sector Bend map expanded around 'matched' momentum for given $B$ field
- All other magnet maps are expanded about the bunch reference momentum $P_{\text{ref}}$
Beam Energy and Tracking

Small difference between adjusting $P_{\text{ref}}$ and $\Delta p/p$ for FF systems (probably FD)
How acceleration is Modelled

• By default, cavities modelled by linear map in the transverse plane:
  – TRANSPORT matrix + end field for TW
  – ‘Chambers’ matrix for SW
• Matrix calculated for $P_{ref}$ (no chromatic effects)
• Alternative: use matrix calculated for each particle (i.e. $P_{ref} (1+\delta)$)
  – More accurate, but slow!
  – No significant difference seen (so far!)
How acceleration is Modelled

- Longitudinal Phase Space
  - Two Methods:

\[
\delta_i \rightarrow \delta_i + \overline{V} \left[ \cos(\phi_0 - k z_i) - \cos(\phi_0) \right] / \left[ 1 + \overline{V} \cos(\phi_0) \right]
\]

<table>
<thead>
<tr>
<th>(P_{\text{ref}})</th>
<th>(P_{\text{ref}}) + (V \cos(\phi_0))</th>
</tr>
</thead>
<tbody>
<tr>
<td>No acceleration</td>
<td>Full acceleration</td>
</tr>
</tbody>
</table>

Use for linac studies
Used for storage rings

\[
\overline{V} = \frac{V}{P_{\text{ref, initial}}}
\]
Wakefields

- $W_\parallel$ and $W_\perp$ modelled as impulse approximation
- Applied at exit of every cavity
- Longitudinal charge distribution estimated by binning particles (within $\pm3\sigma_z$)
- Particles are re-binned after bends (when needed)
- All particles in a bin (‘slice’) receive same kick (no interpolation)
- For transverse wake, $\langle x \rangle$ and $\langle y \rangle$ of each bin is statistically calculated for each impulse
% error in loss parameter (K) as a function of bin number

Based on NLC S-band wake with $\sigma_z = 600\mu m$ gaussian bunch (10$^5$ particles)
Alignment

- Full 3-d alignment \((x, y, z, \theta_x, \theta_y, \theta_z)\)
- For \(\theta_x, \theta_y\) small angle approximation assumed
- Bunch is transformed into local component frame for tracking

Note: for tilted cavities, transverse RF kick and cross-talk between \(W_{\parallel}\) correctly model (I think!)
Alignment: nest frames
Alignment: nest frames

- BPM frame
- QUAD frame
- YCOR frame
- MOVER frame
- SUPPORT frame
Ground motion: Girders and Supports

- Ground motion applied to ‘Support Structures’
- ATL currently only spectrum supported
Tuning: the Channel concept

- Tuning ‘knobs’ and algorithms all work via *channels*
- Channels mimic the control system
- Channels are ‘generic’; algorithms can be easily re-used with other devices
Tuning: the Channel concept

- Linear Feedback
- Signals
- BPM
- Actuators
- YCOR
Tuning: the Channel concept

- Linear Feedback
- Actuators
- Signals
- BPM
- MOVER
TESLA examples

- TESLA linac with coherent betatron oscillation
- Once linear $\langle y \delta \rangle$ correlation removed, $\Delta \varepsilon / \varepsilon < 1\%$
- No filamentation

![Graph showing the effect of initial energy spread on energy divergence]

- With initial energy spread (~3%)
- No initial energy spread

$\langle y \delta \rangle$ removed
TESLA Examples: DR→IP

- X-Y scatter plots at IP
- 35\(\mu\)m random ‘vibration’ applied to all magnets
- Centroid jitter removed
TESLA Examples: DR→IP

- X-Y scatter plots at IP
- Adjusting bunch compressor RF phase by ±2.5°
NLC examples: DR → IP

\[ E_{\text{beam}} = 247.53 \text{ GeV}, \quad \sigma_{x,y} = 236, 3.76 \text{ nm}, \quad \delta_{\text{RMS}} = 0.46\% \]
Storage rings too!

• Thanks to Andy Wolski (LBL)
• Code to support DR studies:
  – Closed orbit, tunes etc.
  – Emittance calculations (Chao’s method)
  – Dynamic aperture studies
  – Realistic wiggler maps
    (AW Merlin extensions, not in core library – yet!)
What’s next?

- More benchmarking with other codes
  See next two talks 😊
- Resolve NLC results
- Repeat for CLIC
- Studies of static and dynamic errors with tuning
  - SLAC ground motion models (spectrums)
  - Implement BBA modules
    DF steering written but not tested
  - Implement tuning knobs (trivial)
- Start modelling ‘machine from start-up’