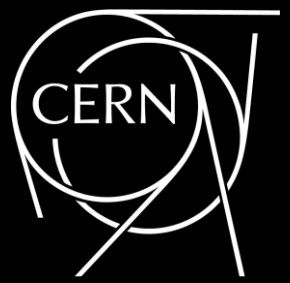
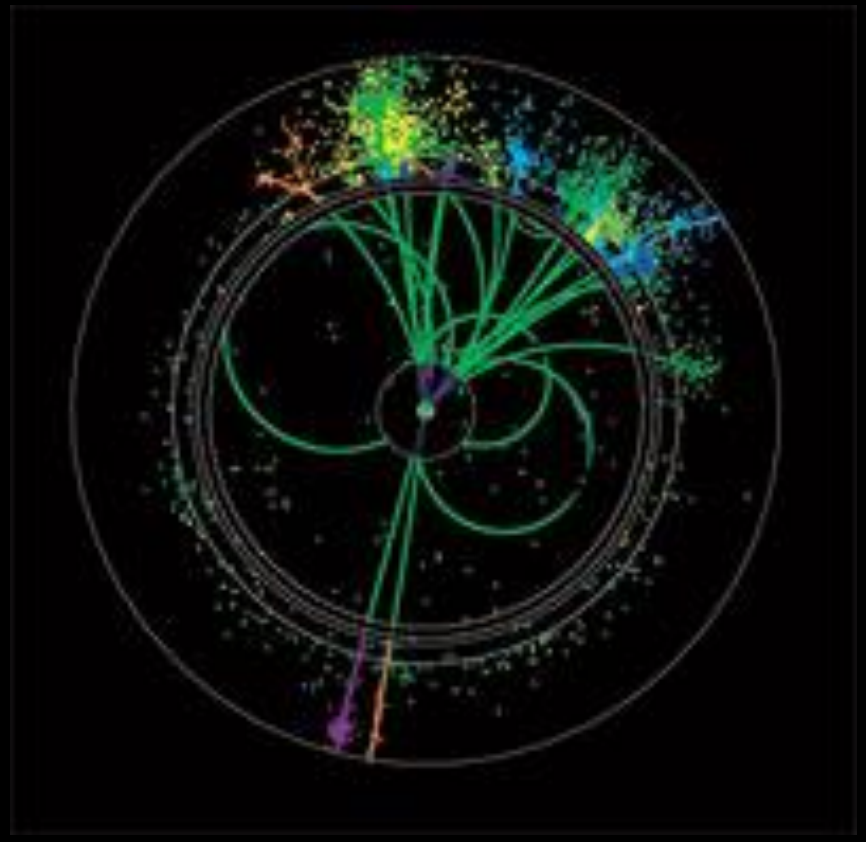
Interpolation based approach to charged particle tracking in non-uniform electromagnetic fields



Outline

- Problem statement
- Baseline tracking algorithm
- Proposed tracking algorithm
- Implementation details
- Benchmarks



Problem statement

Initial value problem

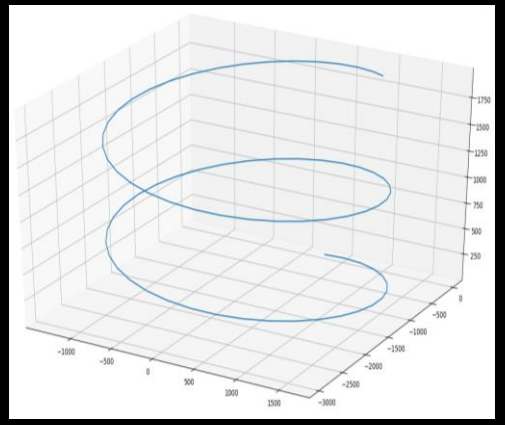
- $y' = f(x, y), y(x_0) = y_0$

Numerical solution

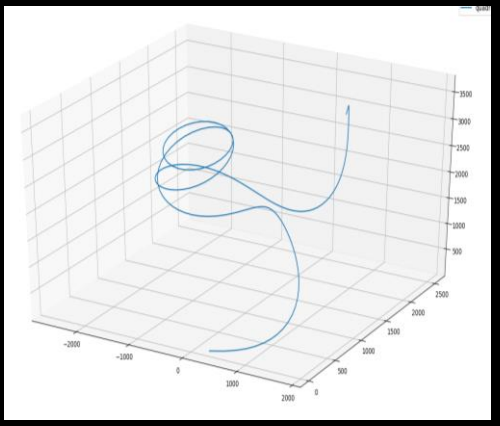
- Runge-Kutta method $(x_0, y_0) \rightarrow (x_t, y_t)$ + interpolation

Step size selection

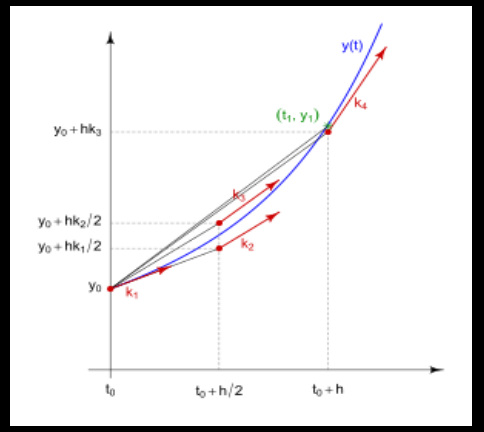
- Adaptive control



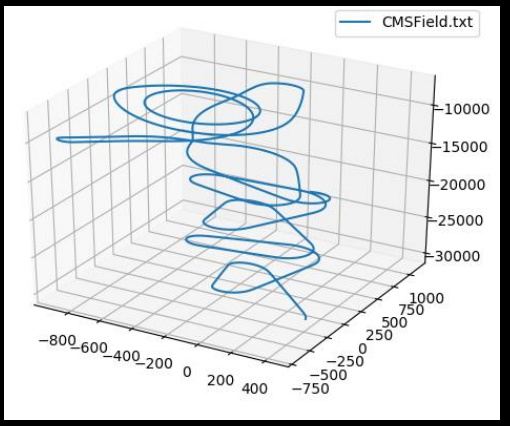
Uniform B-field



Quadrupole B-field



Runge-Kutta method



CMS field

Problem statement

Initial value problem

- $y' = f(x, y), y(x_0) = y_0$

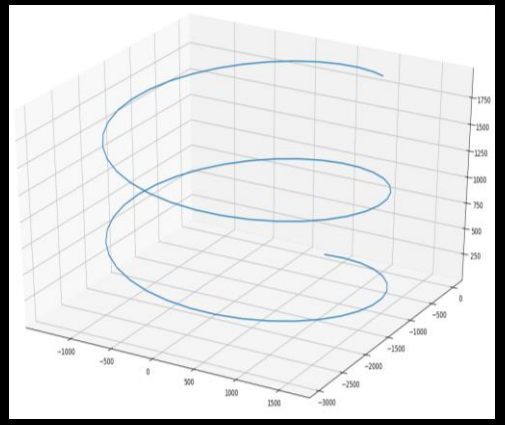
Numerical solution

- Runge-Kutta method $(x_0, y_0) \rightarrow (x_t, y_t)$ + interpolation

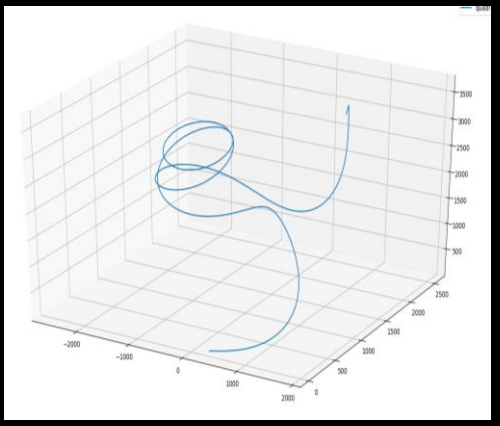
Step size selection

- Adaptive control

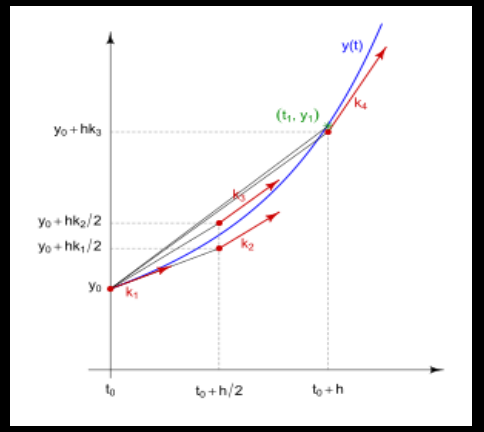
How to find geometry intersection?



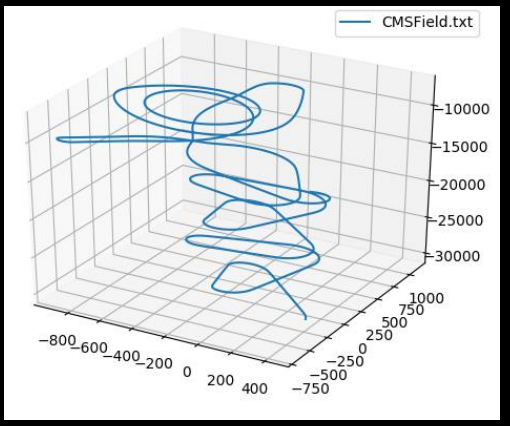
Uniform B-field



Quadrupole B-field



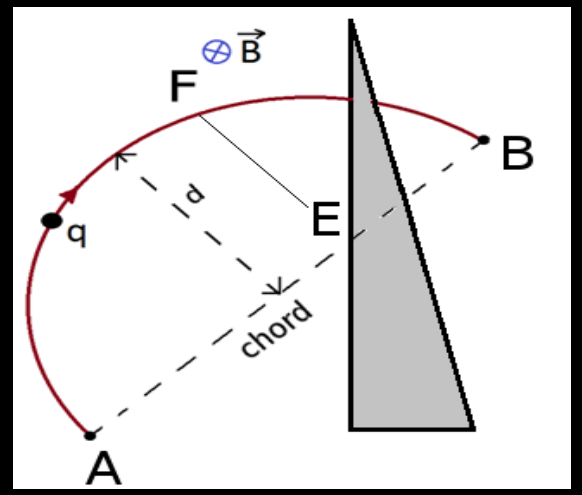
Runge-Kutta method



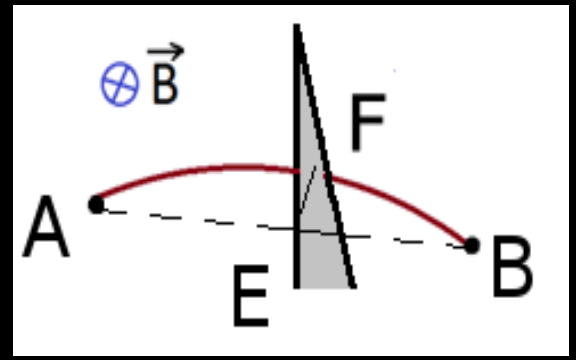
CMS field

Baseline tracking algorithm

- Make series of steps without error control to predict the step size (d < deltaChord)
- Make a step with error control to improve the accuracy if needed (ΔB < deltaOneStep)
- If the chord intersects:
  - make a substeps with error control to locate the intersection point



Step 1

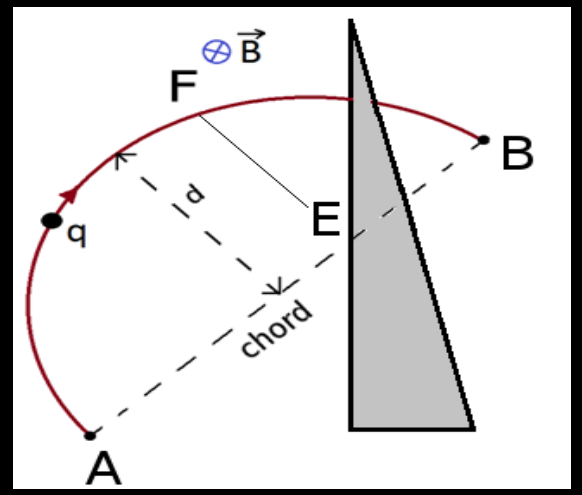


Step 2

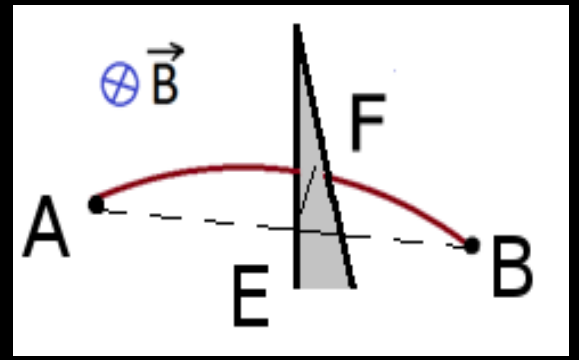
Baseline tracking algorithm

- Make series of steps without error control to predict the step size (d < deltaChord)
- Make a step with error control to improve the accuracy if needed (ΔB < deltaOneStep)
- If the chord intersects:
  - make a substeps with error control to locate the intersection point

Can we use interpolation for intersection and step size estimation?



Step 1



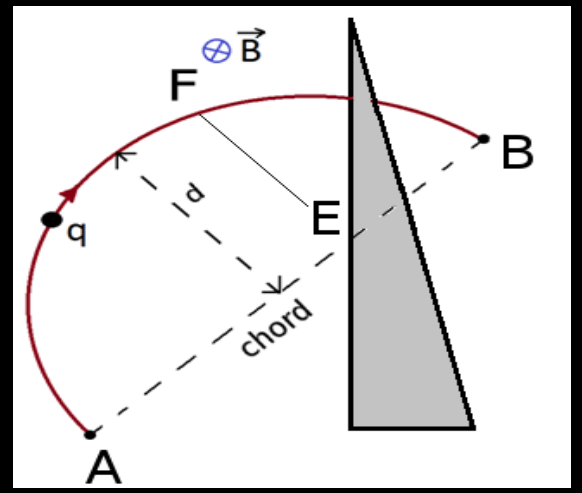
Step 2

Proposed tracking algorithm

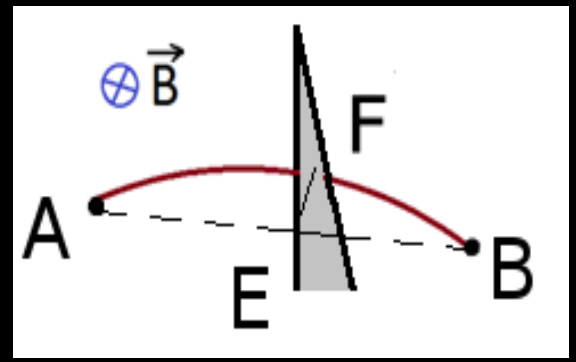
- Make a series of steps with error control until d < deltaChord
- Use interpolation to find point where d < deltaChord
- If the chord intersects:
  - Use interpolation to locate the intersection point

Pros: *A lot fewer* field evaluations required for large steps

Default in 10.5-ref-08



Step 1



Step 2

Main Geant4 classes

- *Field:* magnetic, electric, gravity
- *Equation: uses Field to calculate right hand side of ODE*
- *Stepper:* RK method (DormandPrince, CachKarp, BogackiShampine)
- *Driver:* uses stepper to perform step with error < errorMax
- *ChordFinder:* uses Driver to perform step with d < deltaChord
- *Intersection locator:* uses Driver to locate intersection point

Implementation details

- InterpolationDriver owns std::vector<Stepper> for each substep

- Given step size *s*
  - advance with stepper_i until $\Sigma s_i = s$ or $d > \text{deltaChord}$
  - use interpolation to find y: $d < \text{deltaChord}$
  - keep the whole integrated interval
- Given intermediate point s_i
  - use binary search to find stepper_i
  - use interpolation to obtain y_i

How to use it?

Specify stepper and driver classes manually

```
auto field = new MyScalarRZMagFieldFromMap("cmsmagfield2015.txt");
auto equation = new G4Mag_UsualEqRhs(field);
auto stepper = new G4DormandPrince745(equation);
auto driver = new G4InterpolationDriver<G4DormandPrince745>(minStep, stepper);
auto chordFinder = new G4ChordFinder(driver);
```

Or just use the defaults

```
auto field = new MyScalarRZMagFieldFromMap("cmsmagfield2015.txt");
auto chordFinder = new G4ChordFinder(field);
```

Benchmarks: Test NTST

BaBar silicon vertex tracker and 40-layer drift chamber

	Range cut	Looper cut	Energy cut
run2a.mac	2 mm	200 MeV	1 MeV
run2b.mac	2 mm	200 MeV	Not applied
run2c.mac	2 mm	Not applied	Not applied

Parameters

	Run2a.mac	Run2b.mac	Run2c.mac
Proposed algorithm	0.0135	0.0355	0.203
Baseline algorithm	0.0152	0.0424	0.227

Event user time, s

Number of field calls

	run2a.mac	run2b.mac	run2c.mac
Proposed algorithm	29‘259‘483	198’250’781	1’017’187’737
Baseline algorithm	124’803’759	528’455’931	1’977’952’244

Benchmarks: Full CMS

Particle: 10 GeV pion

	Number of field calls	Time (s)
Baseline algorithm	118’672’255	278
Proposed algorithm	331’081’282	291

Thank you for your attention!