Field Propagation

John Apostolakis (CERN) for the GeantV development team

Outline

- Propagation in field: what, why, ..
- Improved Runge-Kutta algorithms
- Vectorization
- First results
- Status of integration

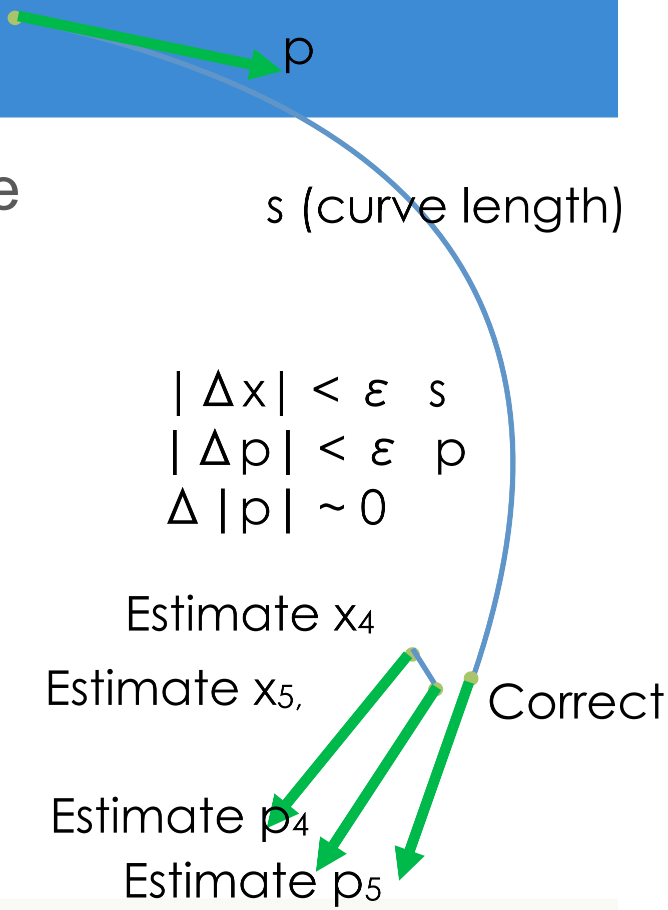
Components

- Field evaluation
- Evaluation of the ‘force’ dP/ds
- Integration of equation of motion – alternative methods
- Driver – ensuring integration accuracy & performance
- Intersection with volume boundaries

- Text

Integrating efficiently

- Given a detector's field B(x,y,z) [or B+E] we need to integrate the trajectory of each track, taking care
  - to stay within a relative accuracy ε
  - to be fast - using as few calls as possible to the field method
- Typically choice is Runge-Kutta method
  - No 'history' + ability to adjust step size



Text

Embedded Runge-Kutta methods

- "Integrate" $dy/dx = F(x, y)$ **interval** x_0 to x_0+h
- Uses **evaluations** of $F(x, y)$
  - $f_i = F(\ x_0 + a_i h\ ,\ \ y_0 + h\ \Sigma_{j<i}\ b_{ij}\ f_j\ )$
  - $y_{estim}(x_0 + h) = \Sigma_i\ c_i\ f_i$
- Each method has its 'tableau' made up by a_i, b_{ij}, c_i
- Key Parameters of an RK method:
  - Number of 'stages' = number of evaluations of f()
  - 'Order' N = the expected scaling of the error ~ h^{N+1}
  - Embedded method = 2nd 'line' to estimate error →

$f_1 = F(x_0\ ,\ \ y_0\ )$
$f_2 = F(\ x_0 + a_2\ h\ ,\ \ y_0 + h\ b_{21}\ f_j\ )$
$f_3 = F(\ x_0 + a_3\ h\ ,\ \ y_0 + h\ b_{31}\ f_1 + h\ b_{32}\ f_2)$

a_i	b_{ij}		
0			
$\frac{1}{2}$	$\frac{1}{2}$		
$\frac{3}{4}$	0	$\frac{3}{4}$	
1	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$

c_j

$$y_{RBS3} = 2f_1/9 + f_2/3 + 4f_3/9$$

c'_j

$\frac{7}{24}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{8}$

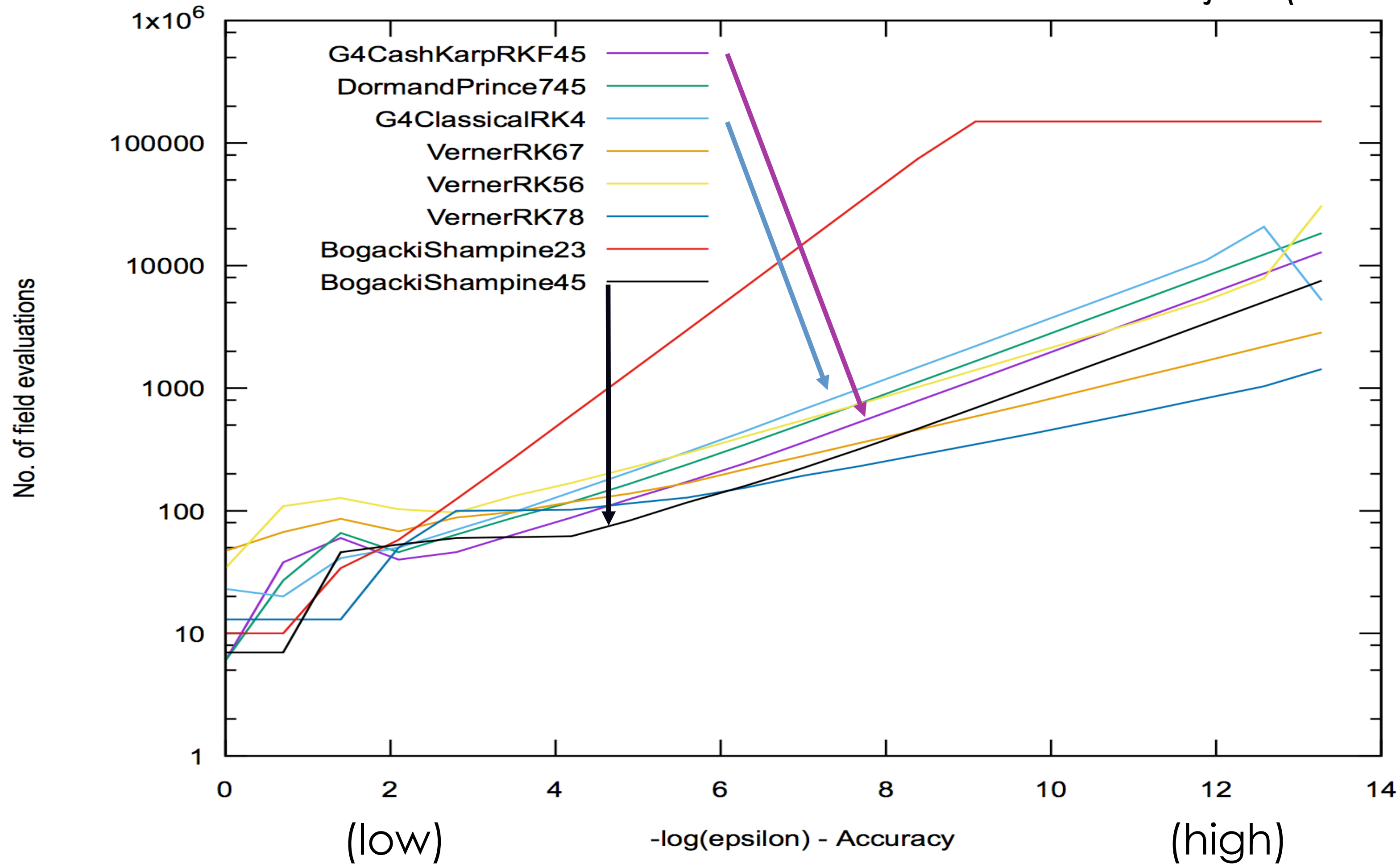
$$y_{est}(x_0 + h) = \Sigma_i\ c_i\ f_i$$

$$\Delta y = \Sigma_i\ (c'_i - c_i)\ f_i$$

Improved methods

- Seek to go far with fewer field evaluations
- RK method order and choice ('tableau' values) determine number of steps to achieve a certain accuracy
- Introduced new, more efficient methods
- New features
  - First Same As Last = Estimates derivative at end-point
  - Interpolation = ability to evaluate (m)any intermediate point(s), with fixed number of extra derivative/field evaluations

Somanth Banerjee (GSoC 2015)



- Text

New methods

- Several newer 5th order RK methods, established & recent
  - State of the art: Dormand-Prince (DoPri5), Bogacki-Shampine
  - New Tsitouras 2008 (obtained ‘without simplifying constraints’)
- Higher order methods - potential use for high accuracy or complex, smooth fields
  - cross-over point depends on complexity of dy/dx i.e. field
- Most have “first same as last” property
  - evaluate/ & use field at endpoint of interval, so it is available ‘no-cost’ for next interval
- Integrated and released in Geant4 10.3-beta (June 2016)

Text

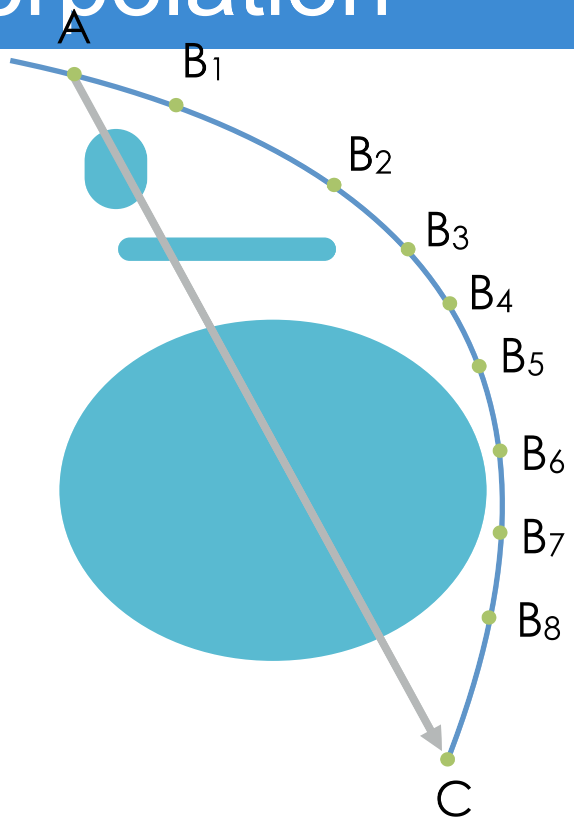
SELECTED new methods

Name / Authors	Order	Stages	Error Estim.	#Evaluations Failed	#Evaluations Good	FSAL	Interpolation (Order)	Extra evaluations
Classical	4	4	N	11	11	No	No	-
CashKarp	5	6	Y	5	5	No	No	-
Dormant-Prince 5 "DoPri5"	5	7	Y	6	5	Yes	Yes - 2 ways (4/5)	0/2
Bogacki-Shampine45	5	8	Y	7	6	Yes	Yes	2
Dormand-Prince8	8	13	Y	12	11	No	No	
Verner78 'efficent'	8	13	Y	12	12	No	Yes - 2 (7 / 8)	4/8

- Text

New RK methods - Interpolation

- Selected RK methods offer capability of estimating any intermediate point given its ‘distance’ along the curve
  - One-time cost of a few extra field evaluations
  - Reduced cost of evaluating intermediate points (vs new integration)
  - Will enable faster location of intersection point with surface boundary



Text

SELECTED new methods

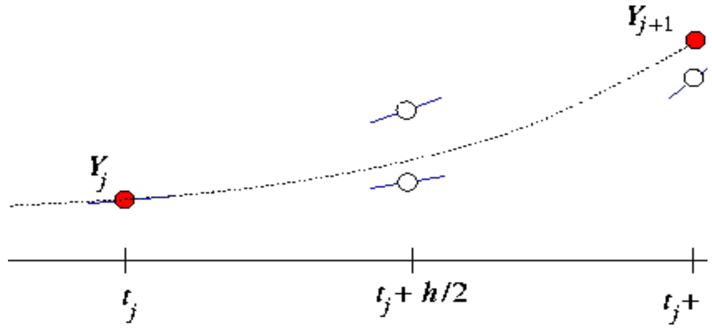
Name / Authors	Order	Stages	Error Estim.	#Evaluations Failed	#Evaluations Good	FSAL	Interpolation (Order)	Extra evaluations
Classical	4	4	N	11	11	No	No	-
CashKarp	5	6	Y	5	5	No	No	-
Dormant-Prince 5 "DoPri5"	5	7	Y	6	5	Yes	Yes - 2 ways (4/5)	0/2
Bogacki-Shampine45	5	8	Y	7	6	Yes	Yes	2
Dormand-Prince8	8	13	Y	12	11	No	No	
Verner78 'efficent'	8	13	Y	12	12	No	Yes - 2 (7 / 8)	4/8

Integrating motion

		Vecotrizable
Magnetic Field	$\vec{B}$	✓
Force	$\vec{F} = \frac{1}{m}\vec{p} \times \vec{B}$	✓
Equation of motion (ODE)	$\frac{d\vec{\mathbf{x}}}{ds} = \frac{\vec{\mathbf{p}}}{\lvert\vec{\mathbf{p}}\rvert}$ $\frac{d\vec{\mathbf{p}}}{ds} = \frac{1}{\mathbf{p}}\overrightarrow{\mathbf{F}(v)}$	✓

Vectorizable

Runge Kutta

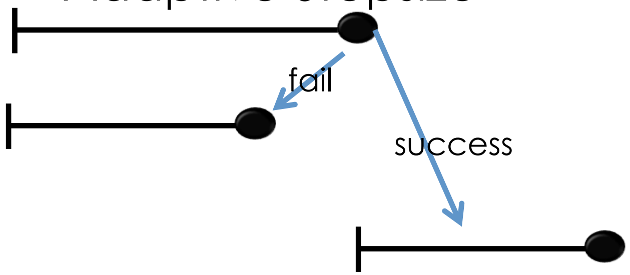


Input : y_o
Output: y_1, Δy_1

✓

Driver

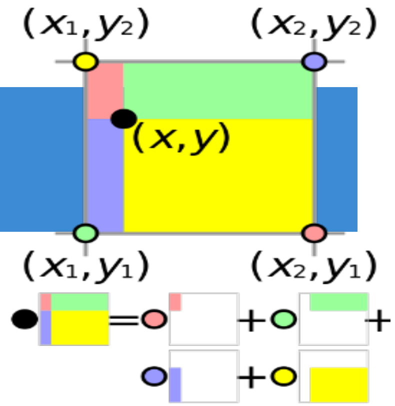
Error Control
Adaptive Stepsize



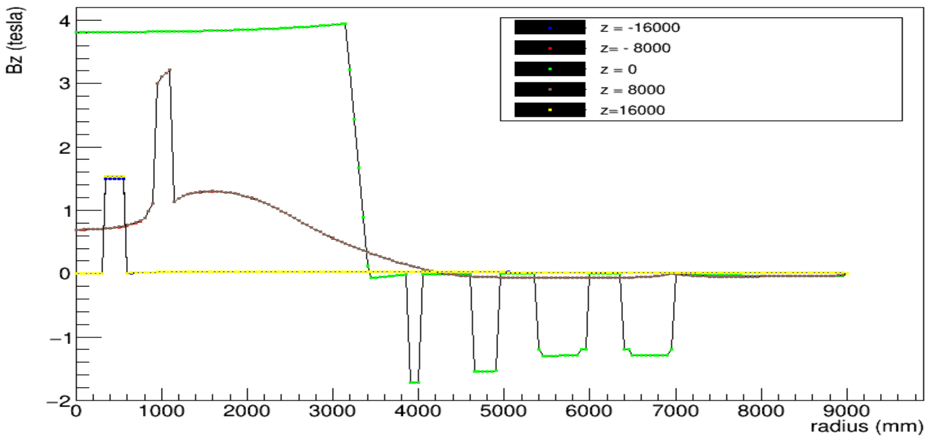
✗

(Not Naively vectorizable)

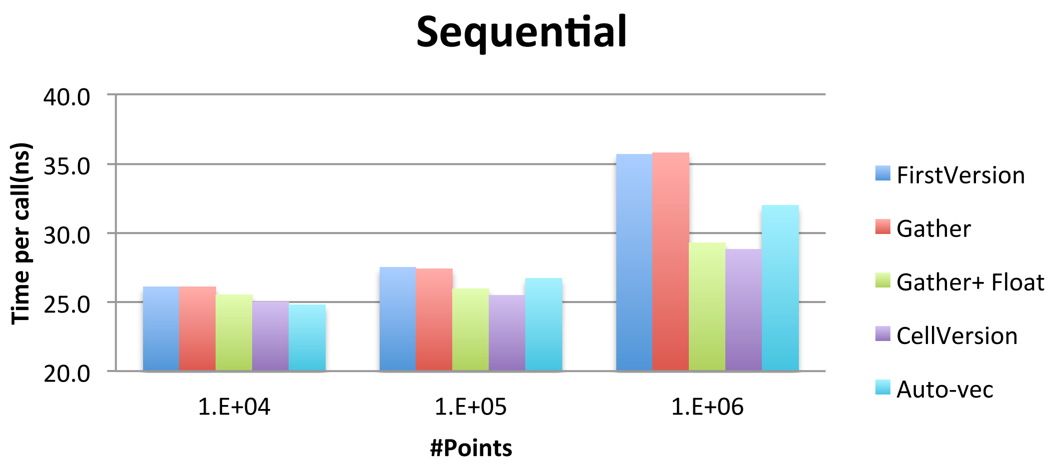
Bilinear field interpolation

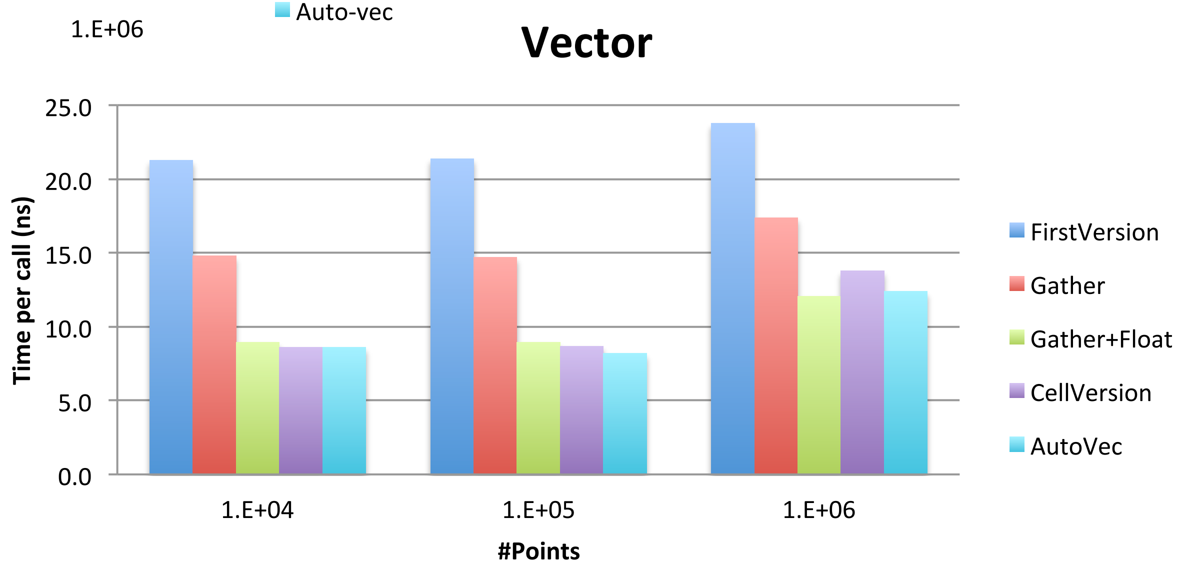


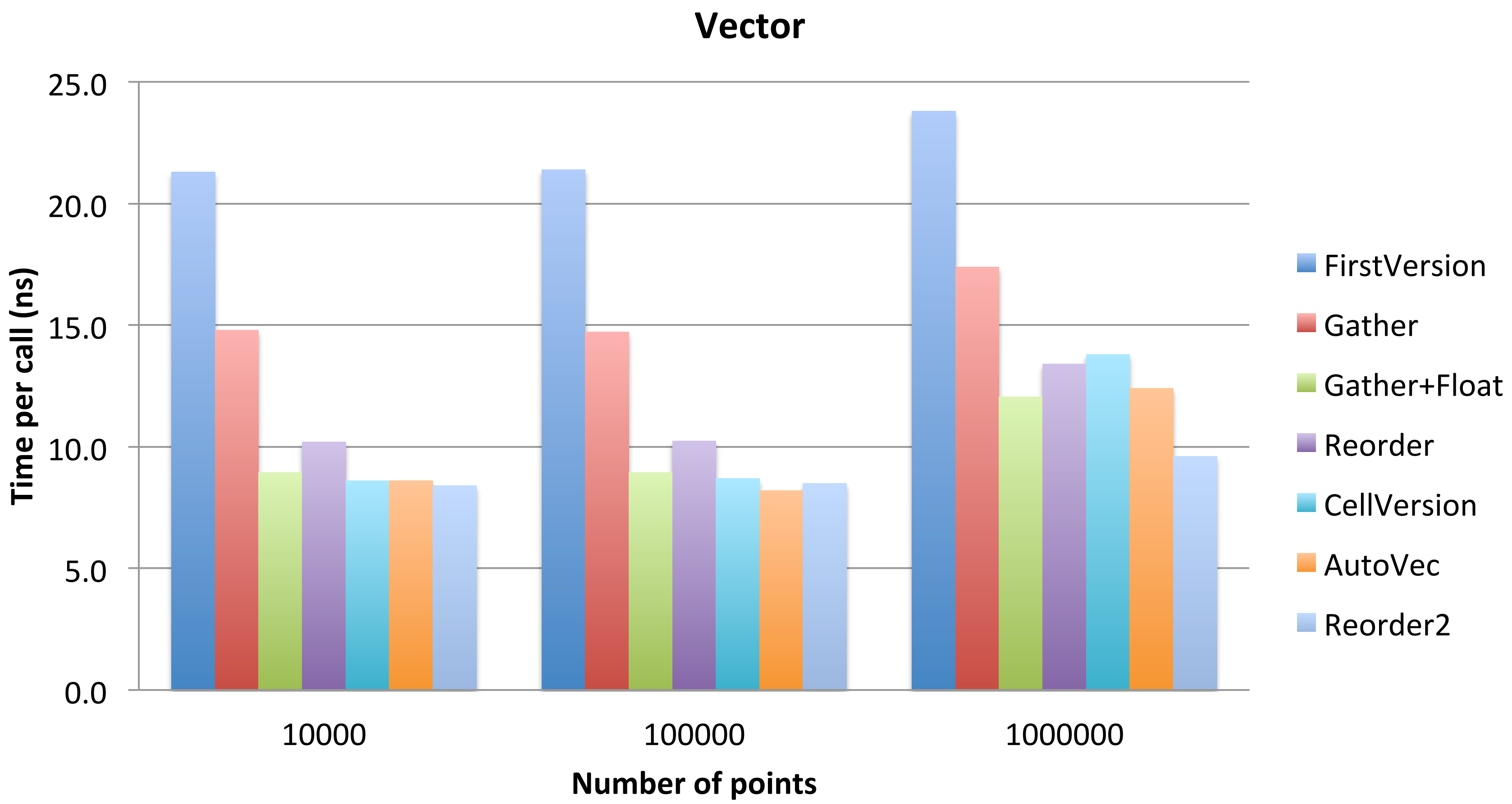
- Start with sample values of 2D CMS field.
- Assume phi-symmetric field.
- Find magnetic field given a point in 3D space.

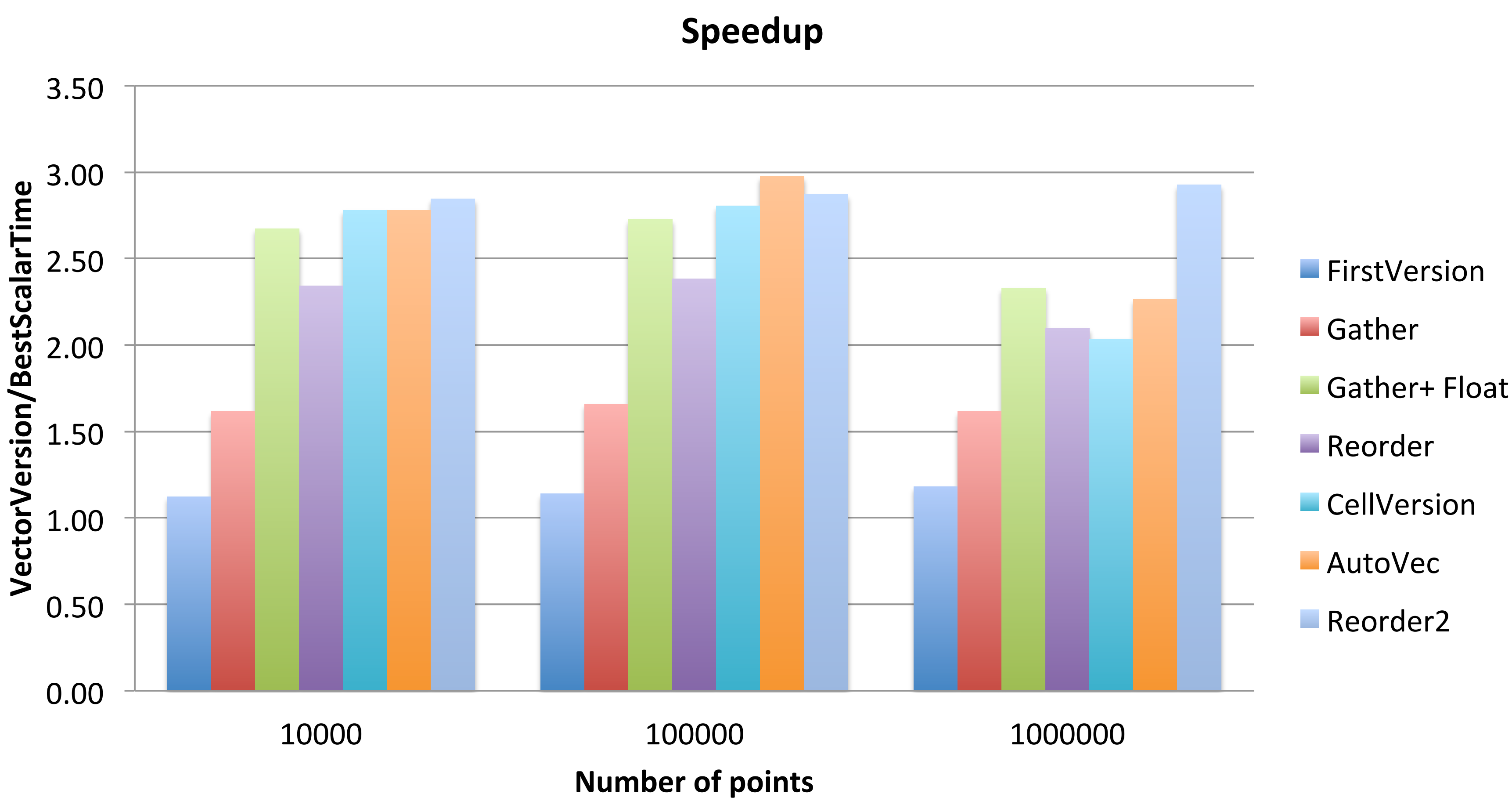


B_Z vs r (for boundaries along Z)


Sequential
40.0
35.0
30.0
25.0
20.0
Time per call(ns)
1.E+04
1.E+05
1.E+06
#Points
FirstVersion
Gather
Gather+ Float
CellVersion
Auto-vec


Vector
25.0
20.0
15.0
10.0
5.0
0.0
Time per call (ns)
1.E+04
1.E+05
1.E+06
#Points
FirstVersion
Gather
Gather+Float
CellVersion
AutoVec


Vector
25.0
20.0
15.0
10.0
5.0
0.0
Time per call (ns)
10000
100000
1000000
Number of points
FirstVersion
Gather
Gather+Float
Reorder
CellVersion
AutoVec
Reorder2


Speedup
VectorVersion/BestScalarTime
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
10000
100000
1000000
Number of points
FirstVersion
Gather
Gather+ Float
Reorder
CellVersion
AutoVec
Reorder2

Field evaluation: observations

- Speedup factor of ~3
- Semi-realistic benchmark:
  - Half the points are new; the other half are ‘moved’ near to previous values.
  - Exponential random distribution.
  - Time reduced by ~5%. Likely effect is from cache.
- Difference in performance from changing doubles to floats:
  - 3-20% for sequential
  - 30-40% for vector version
- Difference in performance from changing order of memory operations:
  - 5-7% for sequential
  - 5-20% for vector version

Vectorization of integration driver

- Takes a buffer stream of 16 particles/tracks.
- Starts working with 4 in Vc vector.
- As soon as integration is over for one track, insert a new track in its place.

Preliminary results (100 steps)		
	Sequential	Vectorized
#CashKarp Calls	435	172
#OneGoodStep Calls	324	94

Status of integration

- Currently dispatches single tracks for integration of motion - ‘baskets’ are under development
- Using Runge-Kutta for every step is costly >40% CPU time for complicated field (bilinear model of CMS field)
- Added choice between helix and Runge-Kutta on per track basis – arbitrarily $0.05 < \theta < 10$ (radians) 4% CPU time
  - Must improve condition to depend on variability of field within the volume (or the step)
- Ready for gluing of first ‘vector’ implementation of integration

Backup slides

Vtune analysis

		Elapsed Time	Instructions Retired	CPI Rate	Back-end Bound	Memory Bound	Core Bound	Port Utilization
Sequential (nRep = 200)	Reorder2 (Haswell Xeon)	8.166	31.7B	0.947	0.786	0.282	0.505	0.327
Vector (nRep = 500)	Reorder2 (Haswell Xeon)	7.665	33.8B	0.776	0.697	0.143	0.555	0.463