FiTQun Computing Times with different Minimizers

JULIAN

Minuit Minimizers

- Fits (Prefit, One Ring, Multiring, etc.) done using Minuit minimizers
- Each algorithm uses a specified minimizer
- Three main Minuit minimizers: "**Simplex**", "**Migrad**", "**Minimize**"
- **Simplex** algorithm is a stepping method, doesn't produce covaraince matrix and gives no information about parameter correlation. Itself cannot really judge how good the fit was.
- **Migrad** algorithm minimizes with the function and its first derivatives using a variable metric method, produces covariance matrix. (According to Minuit Documentation the best and fastest choice.)
- **Minimize** starts by using Migrad. If that fails, it switches to Simplex. After Simplex completes, it switches back to Migrad.

- FiTQun uses **Simplex** for every fit. Why?

Testing the Minimizers

1. Simulate events in HyperK using WCSim
   - E.g. Particle gun with 10 electron events at 100MeV from (0,0,0) in x-direction
2. Implement a counter into FiTQun that counts every interation of a minimizer.
3. Run FiTQun multiple times with different minimizers and extract time and number of iterations
   - For computing times, FiTQun parameters like DoSubEvent or DoMRFit are also very important
   - The CC-IN2P3 cluster is rather slow, so the ratio of computing times is what counts!
4. Extract fitting information from .root file and compare performance

Test Case

- 10 e^- @ 100MeV
- Only single ring fit
- No multi-ring, no multi-segment
- DoSubEvent on
- Only test for electron hypothesis

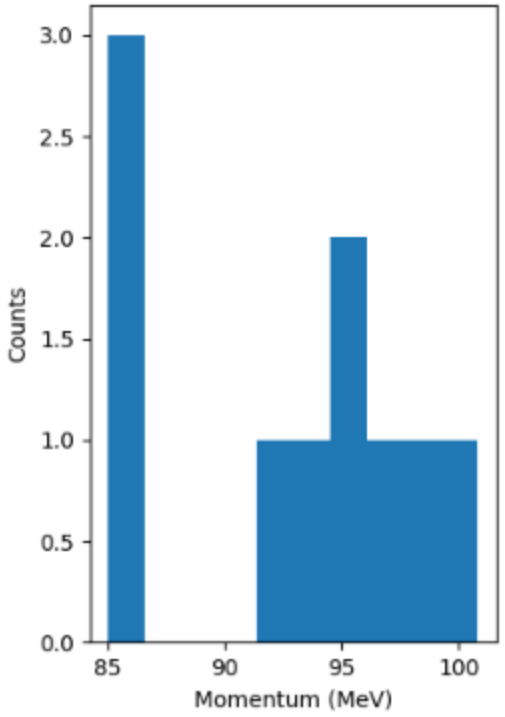
```
Attempt 30 Iteration 20084
Attempt 30 Iteration 20085
Attempt 30 Iteration 20086
Attempt 30 Iteration 20087
Attempt 30 Iteration 20088
Attempt 30 Iteration 20089
Attempt 30 Iteration 20090
Attempt 30 Iteration 20091
Attempt 30 Iteration 20092
Attempt 30 Iteration 20093
Attempt 30 Iteration 20094
MINUIT did not converge!!! n_Iter=1381
```

Minimizer	SIMPLEX	MIGRAD	MINIMIZE
Computing Time	516 s	1136 s	1728 s
N(iterations)	4324	11692	20094
N(attempts)	30	30	30

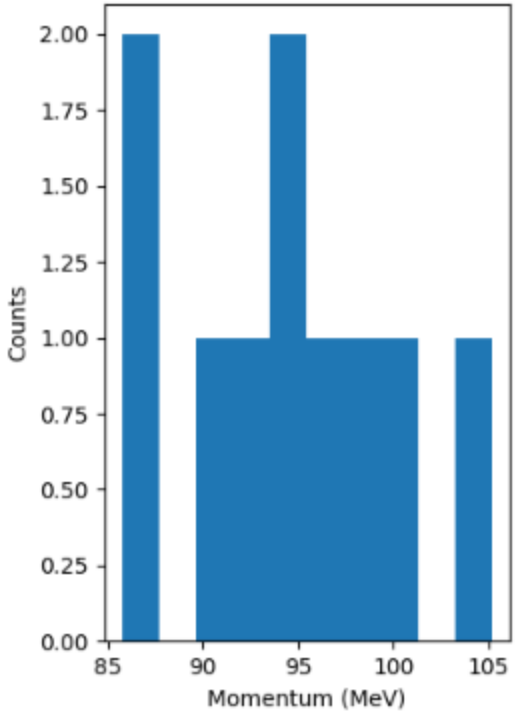
- N(Iterations) corresponds to number of executions of OneRingWrapper function
- N(attempts) corresponds to number of executions of OneRingFit function
- Simplex fastest by far (in every tested example)
- Always 3 attempts per event
- Iteration steps take same amout of time
- Migrad appears to fail

Performance - Momentum

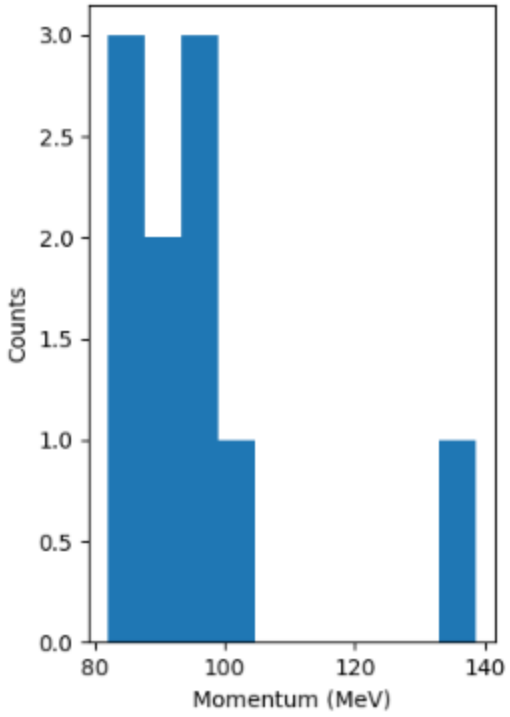
Simplex



Migrad

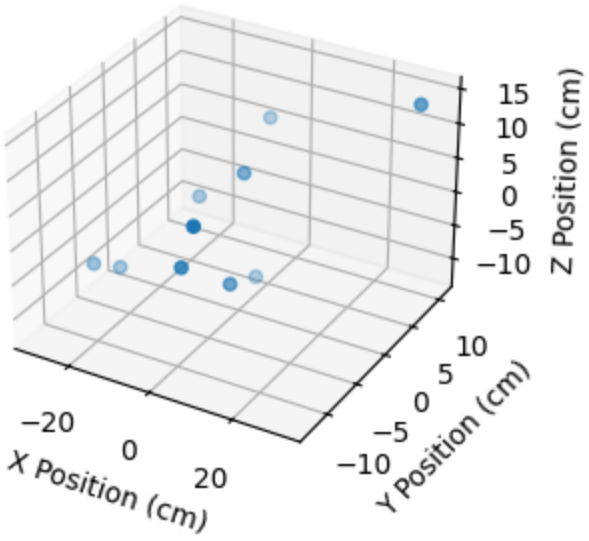


Minimize

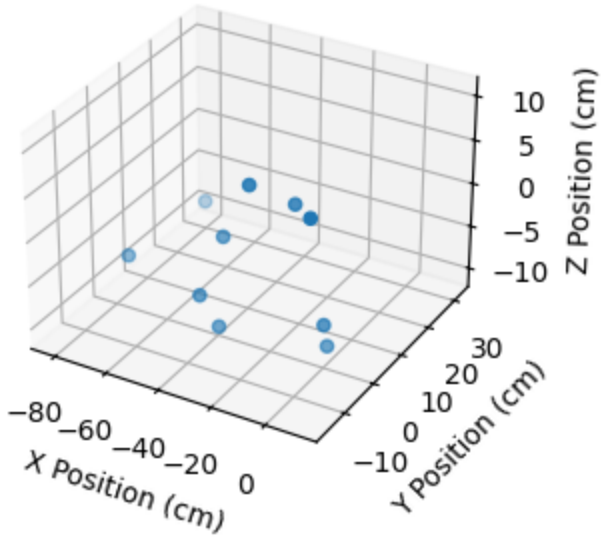


Performance - Position

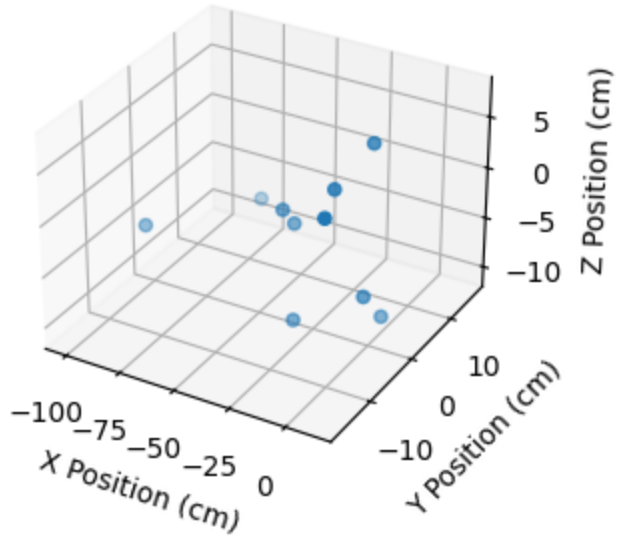
Simplex



Migrad

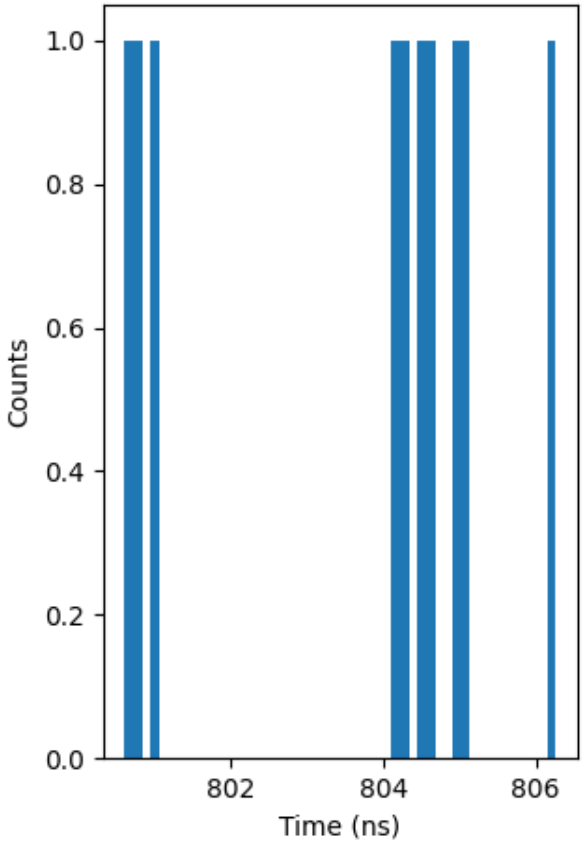


Minimize

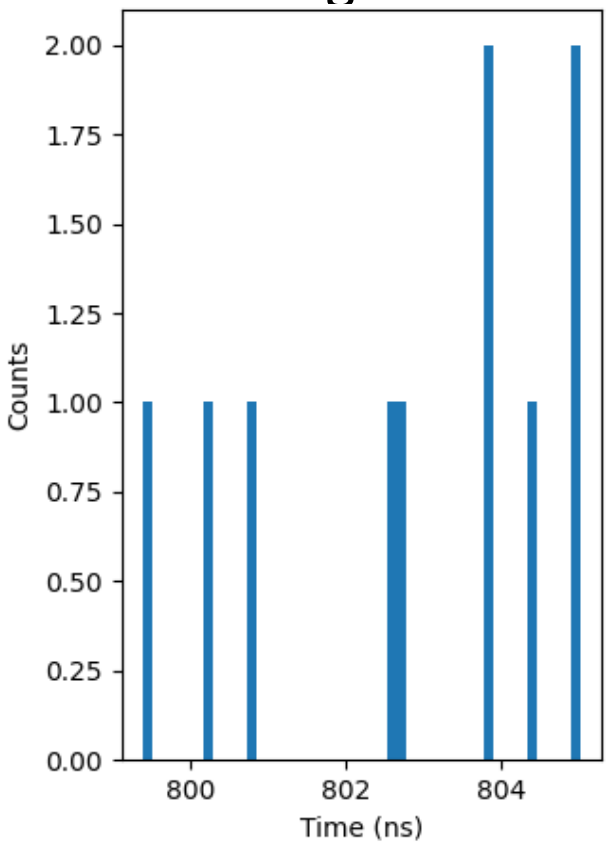


Performance - Time

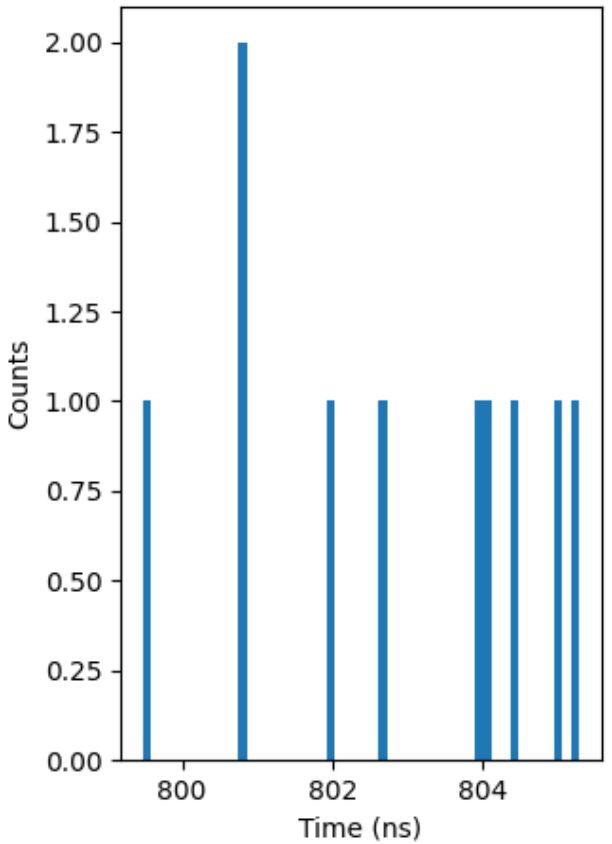
Simplex



Migrad



Minimize



Conclusions

- Computing time is governed by number of iterations --> should be minimized
- Simplex is the fastest minimizer, doesn't concede performance
- Migrad struggles / "fails" to converge
- Minimize is slowest and doesn't improve performance
- All three have similar performance in momentum and time reconstruction
- Simplex more accurate in position reconstruction

To Do:

- Improve statistics
- Compare further performance parameters (E.g. extract lifetime from muon events, nll comparison of electron vs. muon hypothesis, etc.
- Script to track changes in performance and computing time