Summer Student Update 4

An (even more) In Depth Look at the RDataFrame Macro

Konrad Helms

FTX Software Meeting

01.09.22

What happened so far:

- wrote many macros (with different sub-versions) for Higgs recoil mass in $e^+e^- \to Z^* \to ZH \to H\ell^+\ell^-$
- last week: mainly focused on RDataFrame macro in python

Language	Macro	Avg. runtime	Min. runtime	Max. runtime
ROOT	c++ api	17.97 s	17.54 s	18.08 s
Python	Conventional evt. loop	20.20 s	19.86 *s*	20.56 s
	Uproot (lazy)	12.44 s	11.91 s	12.95 s
	Uproot (concatenate)	59.49 s	59.42 s	59.57 s
	RDataFrame*	**15.60 s**	**15.11 s**	**16.55 s**

*: RDataFrame without `ROOT.EnableImplicitMT(...)` $\implies$ ROOT chooses how many threads are taken

The RDF Macro

- assumption: we are I/O bound
- used `root-readspeed` (GitHub) tool to measure the expected optimal throughput from ROOT for a given dataset

```
Thread pool size:0
Real time: 1.57872 s
CPU time: 0.87 s
Uncompressed data read: 68572 bytes
Compressed data read: 33959 bytes
Uncompressed throughput: 0.041423 MB/s
0.041423 MB/s/thread for 1 threads
Compressed throughput: 0.020514 MB/s
0.020514 MB/s/thread for 1 threads
```

RDF avg. runtime: 15.60 s
↓
real time for read: ca. 1.6 s
↓
not I/O bound?

The Hunt for the Long Runtimes

- overhead times:
  - `from os import listdir`: 0.2 s
  - `import ROOT` +1.8 s
  - `from ROOT import RDataFrame` +0.8 s
  - ...

  $\Longrightarrow$ importing all modules: $\simeq$ 2.5 s to 3 s
- use: `verbosity = ROOT.Experimental.RLogScopedVerbosity(ROOT.Detail.RDF.RDFLogChannel(),ROOT.Experimental.ELogLevel.kInfo)` in python macro

The Hunt for the Long Runtimes - Signal (?!)

```
Run()>: Starting event loop number 0.

Jit()>: Just-in-time compilation phase
completed in 3.573096 seconds.

RunTreeReader()>: Processing trees in
files signal: entry range [0,17142],
using slot 0 in thread 139735803369280.


Run()>: Finished event loop number 0
(0.03s CPU, 0.028059s elapsed).
```

```python
def main():

    print("processing signal...")
    sigDir = "../../data/signal"
    sig_files = getFiles(sigDir)
    sig_events = RDataFrame("events",sig_files)
    sig_hist = doEvtLoop(sig_events,"signal")
    print("...done")

    print("processing bkg...")
    bkgDir = "../../data/bkg"
    bkg_files = getFiles(bkgDir)
    bkg_events = RDataFrame("events",bkg_files)
    bkg_hist = doEvtLoop(bkg_events,"bkg")
    print("...done")
```

```python
def doEvtLoop(df,histtitle):
    df = applyCut(df,2)

    df = makeLorentzVector(df)
    #df.Display(["idx","px"]).Print()

    df = calcRecoilMass(df)
    #df.Display(["idx","px","py","pz","e","Einit","recoilMass"]).Print()
    print("Stats for "+ histtitle + " events")
    df.Report().Print()
    hist = df.Histo1D((histtitle, "; recoil mass [GeV]; ; ",40,50.,250.),"recoilMass")

    return hist
```

The Hunt for the Long Runtimes - Bkg. (?!)

```
Run()>: Starting event loop number 0.

Jit()>: Just-in-time compilation phase
completed in 1.389356 seconds.

RunTreeReader()>: Processing trees in
files bkg: entry range [0,292805],
using slot 0 in thread 139735803369280.


Run()>: Finished event loop number 0
(0.31s CPU, 0.308021s elapsed).
```

```python
def main():

    print("processing signal...")
    sigDir = "../../data/signal"
    sig_files = getFiles(sigDir)
    sig_events = RDataFrame("events",sig_files)
    sig_hist = doEvtLoop(sig_events,"signal")
    print("...done")

    print("processing bkg...")
    bkgDir = "../../data/bkg"
    bkg_files = getFiles(bkgDir)
    bkg_events = RDataFrame("events",bkg_files)
    bkg_hist = doEvtLoop(bkg_events,"bkg")
    print("...done")
```

```python
def doEvtLoop(df,histtitle):
    df = applyCut(df,2)

    df = makeLorentzVector(df)
    #df.Display(["idx","px"]).Print()

    df = calcRecoilMass(df)
    #df.Display(["idx","px","py","pz","e","Einit","recoilMass"]).Print()
    print("Stats for "+ histtitle + " events")
    df.Report().Print()
    hist = df.Histo1D((histtitle, "; recoil mass [GeV]; ; ",40,50.,250.),"recoilMass")

    return hist
```

The Hunt for the Long Runtimes - Signal (???)

```
Run()>: Starting event loop number 1.

Jit()>: Just-in-time compilation phase
completed in 0.252379 seconds.

RunTreeReader()>: Processing trees in
files signal: entry range [0,17142],
using slot 0 in thread 139735803369280.


Run()>: Finished event loop number 1
(0.22s CPU, 0.215331s elapsed).
```

```python
def main():

    print("processing signal...")
    sigDir = "../../data/signal"
    sig_files = getFiles(sigDir)
    sig_events = RDataFrame("events",sig_files)
    sig_hist = doEvtLoop(sig_events,"signal")
    print("...done")

    print("processing bkg...")
    bkgDir = "../../data/bkg"
    bkg_files = getFiles(bkgDir)
    bkg_events = RDataFrame("events",bkg_files)
    bkg_hist = doEvtLoop(bkg_events,"bkg")
    print("...done")
```

```python
def doEvtLoop(df,histtitle):
    df = applyCut(df,2)

    df = makeLorentzVector(df)
    #df.Display(["idx","px"]).Print()

    df = calcRecoilMass(df)
    #df.Display(["idx","px","py","pz","e","Einit","recoilMass"]).Print()
    print("Stats for "+ histtitle + " events")
    df.Report().Print()
    hist = df.Histo1D((histtitle, "; recoil mass [GeV]; ; ",40,50.,250.),"recoilMass")

    return hist
```

The Hunt for the Long Runtimes - Bkg. (???)

```
Run()>: Starting event loop number 1.

Jit()>: Nothing to jit and execute.

RunTreeReader()>: Processing trees in
files bkg: entry range [0,292805],
using slot 0 in thread 139735803369280.


Run()>: Finished event loop number 1
(2.51s CPU, 2.50069s elapsed).
```

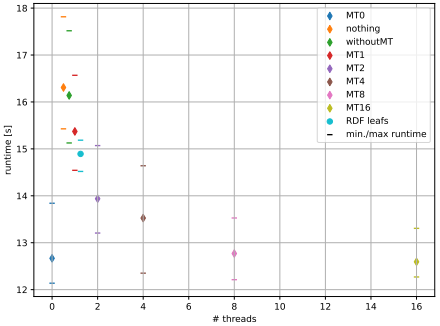
```python
def main():

    print("processing signal...")
    sigDir = "../../data/signal"
    sig_files = getFiles(sigDir)
    sig_events = RDataFrame("events",sig_files)
    sig_hist = doEvtLoop(sig_events,"signal")
    print("...done")

    print("processing bkg...")
    bkgDir = "../../data/bkg"
    bkg_files = getFiles(bkgDir)
    bkg_events = RDataFrame("events",bkg_files)
    bkg_hist = doEvtLoop(bkg_events,"bkg")
    print("...done")
```

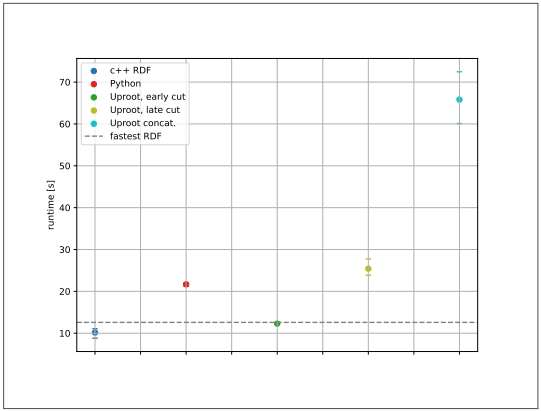
```python
def doEvtLoop(df,histtitle):
    df = applyCut(df,2)

    df = makeLorentzVector(df)
    #df.Display(["idx","px"]).Print()

    df = calcRecoilMass(df)
    #df.Display(["idx","px","py","pz","e","Einit","recoilMass"]).Print()
    print("Stats for "+ histtitle + " events")
    df.Report().Print()
    hist = df.Histo1D((histtitle, "; recoil mass [GeV]; ; ",40,50.,250.),"recoilMass")

    return hist
```

Additionally

- RDF macro in c++ (shows the same strange behaviour)
- started writing the report and presentation for Monday
- timed every macro (next slide), but for better comparison do it again and switch to other system instead of `naf-ilc11`

Plots


MT0
nothing
withoutMT
MT1
MT2
MT4
MT8
MT16
RDF leafs
min./max runtime
runtime [s]
threads


c++ RDF
Python
Uproot, early cut
Uproot, late cut
Uproot concat.
fastest RDF
runtime [s]