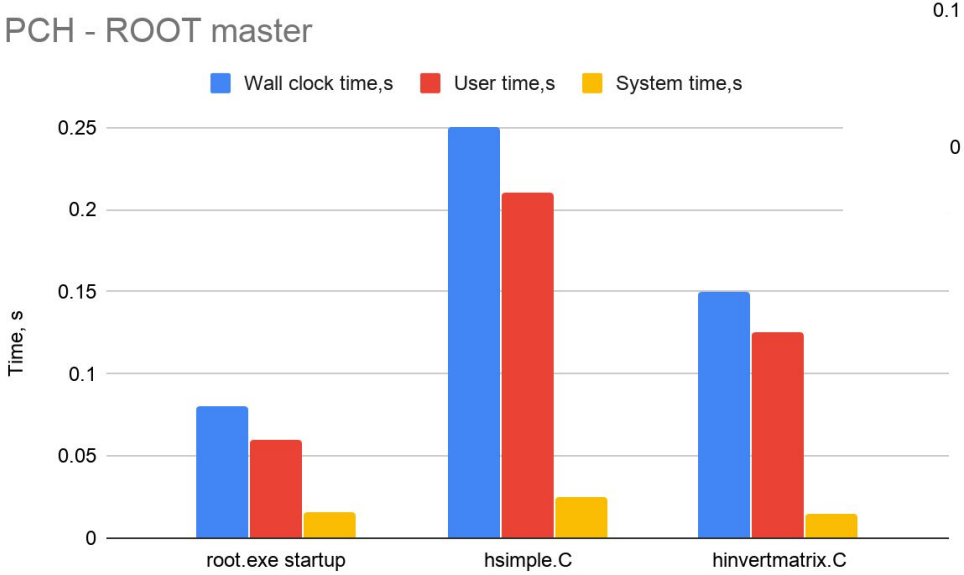
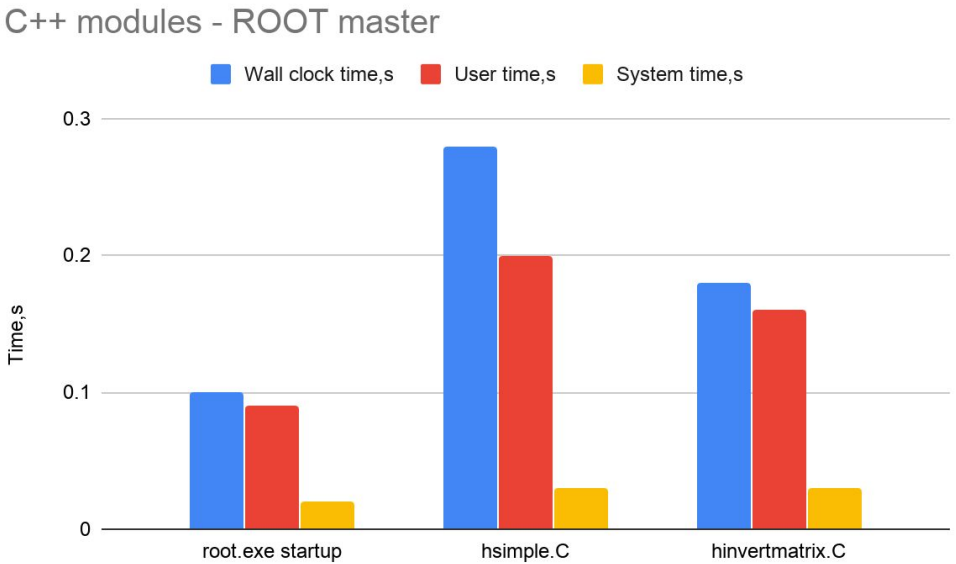
Preloading of C++ Modules

- Loads all PCMs in LD_LIBRARY_PATH at startup
- Very efficient but not a no-op: memory use from deserialized module preamble - memory footprint scales with number of preloaded modules. Various efficiency improvements possible in clang, such as:
  - Lazy loading of modules based on global module indices. Modules are not preloaded but loaded when an identifier is unknown.
  - Mixed mode: preloading essential modules and lazy-load non-essential ones.

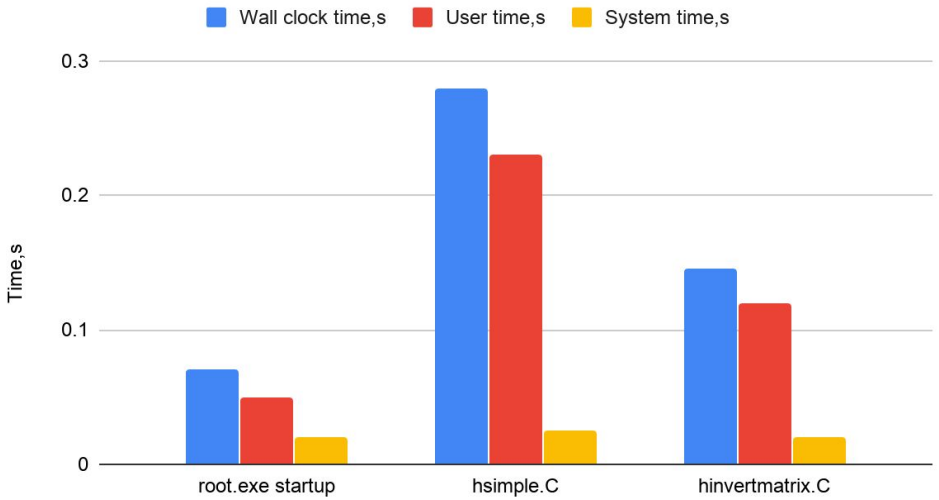
Lazy Loading of C++ Modules

- Work in progress: Global Module Index: PR4016

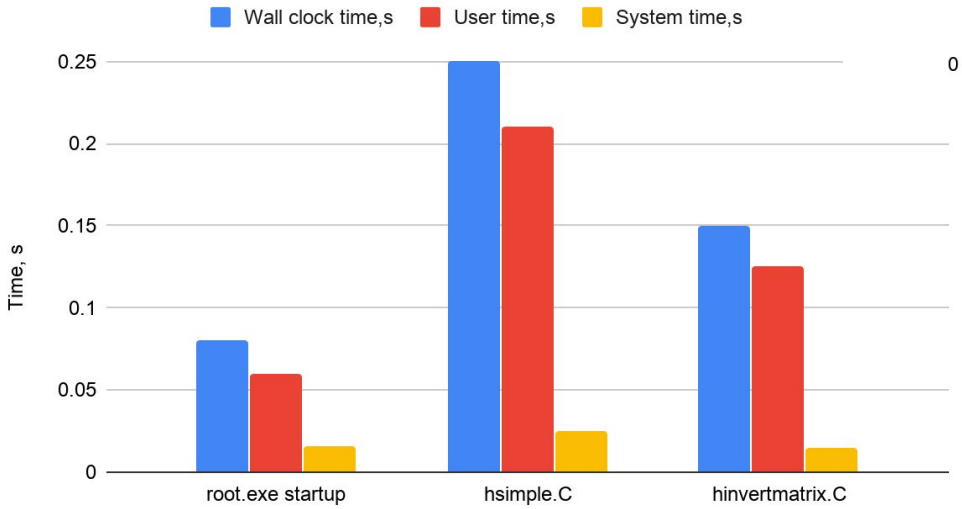

PCH - ROOT master
Wall clock time,s
User time,s
System time,s
0.25
0.2
0.15
0.1
0.05
0
Time, s
root.exe startup
hsimple.C
hinvertmatrix.C


C++ modules - ROOT master
Wall clock time,s
User time,s
System time,s
0.3
0.2
0.1
0
Time,s
root.exe startup
hsimple.C
hinvertmatrix.C

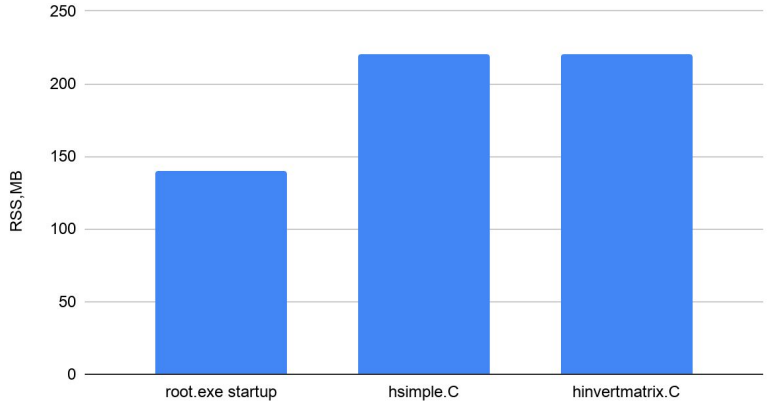
GMI - ROOT master



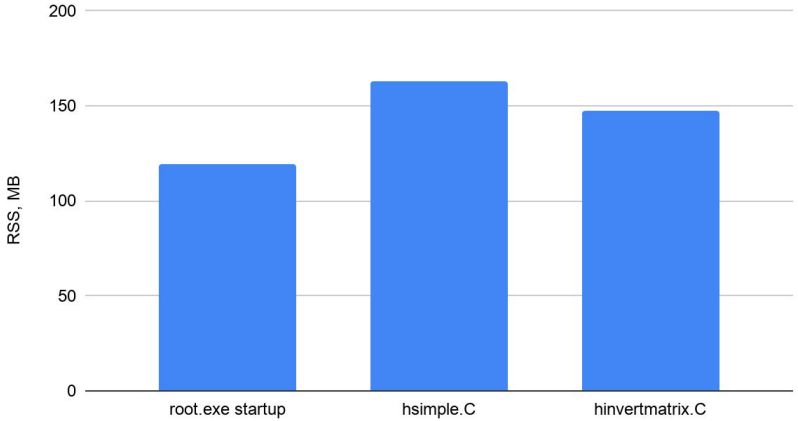
PCH - ROOT master



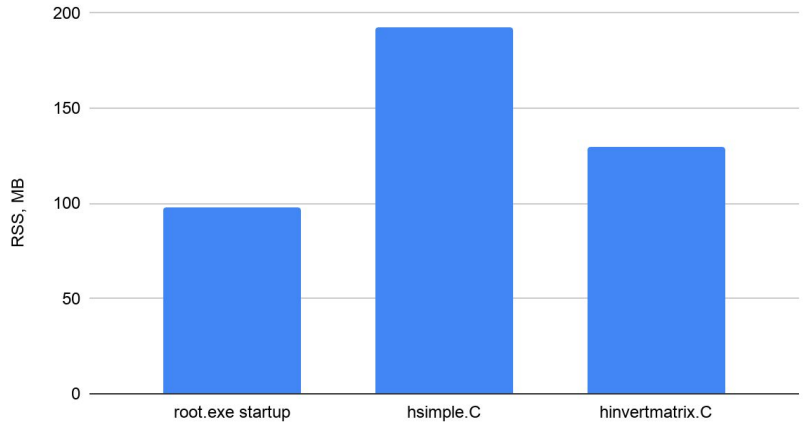
C+ modules - ROOT master



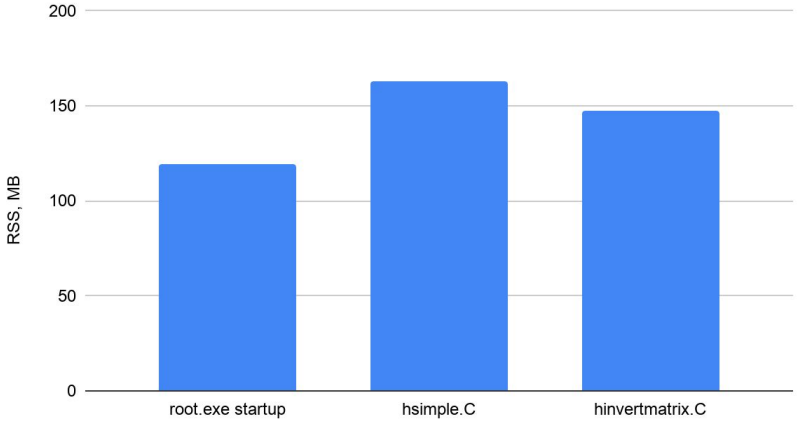
PCH Master



GMI - ROOT master



PCH Master



Lazy Loading of C++ Modules: Preliminary performance results

- We can reduce significantly the loaded modules (for the hsimple we currently load 50 modules rather than 20)

- The excess in rt is due to the many virtual calls to resolve identifier from module. Our current understanding is that it can be optimized a lot.

CMSSW

(measurements done before ROOT-10514)