WS 25/26 Numerik 0 Notes

Igor Dimitrov

2025-10-10

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

1 Intro

1.1 HDNum

1.1.1 add the hdnum repo as a submodule:

```
git submodule add https://parcomp-git.iwr.uni-heidelberg.de/Teaching/hdnum.git hdnum
git commit -m "Add hdnum as submodule"
git push
```

1.1.2 clone a repo that already has submodules

- if you clone for the first time:

```
git clone --recurse-submodules https://parcomp-git.iwr.uni-heidelberg.de/Teaching/hdnum.git
```

- if you clone first without recurse

```
git clone <your-repo-url>
cd num-notes-ws25
git submodule update --init --recursive
```

1.1.3 pulling the updaets in the submodules on another machine

```
git pull
git submodule update --init --recursive
```

1.1.4 CMAKE

1.1.4.1 1) Repository layout

```
num-notes-ws25/
  _quarto.yml
  code_examples/
    hdnum/                    # submodule (header-only library)
    my_solutions/             # your code
      CMakeLists.txt          # top-level for examples
      src/
        ch1/
          CMakeLists.txt
          ch1_ode_demo.cpp
          ch1_newton_demo.cpp
        ch2/
          CMakeLists.txt
        ...
  .github/workflows/publish.yml
```

1.1.4.2 2) Top-level CMake for examples (`code_examples/my_solutions/CMakeLists.txt`)

- Defines a shared **INTERFACE** target `hdnum_common` (provides include paths, optional GMP).
- Adds each chapter as a subdirectory if it exists.

```
cmake_minimum_required(VERSION 3.16)
project(hdnum_examples CXX)
set(CMAKE_CXX_STANDARD 20)

### (Optional) put all executables under a single bin/ root
set(CMAKE_RUNTIME_OUTPUT_DIRECTORY "${CMAKE_BINARY_DIR}/bin")

### hdnum include dir (submodule)
set(HDNUM_DIR "${CMAKE_CURRENT_LIST_DIR}/../hdnum")

### Common config shared by all examples
add_library(hdnum_common INTERFACE)
target_include_directories(hdnum_common INTERFACE "${HDNUM_DIR}")

### Optional: enable GMP support in hdnum
```

```
option(HDNUM_USE_GMP "Enable GMP in hdnum" OFF)
if(HDNUM_USE_GMP)
  target_compile_definitions(hdnum_common INTERFACE HDNUM_HAS_GMP=1)
  find_library(GMPXX gmpxx)
  find_library(GMP   gmp)
  if(GMPXX AND GMP)
    target_link_libraries(hdnum_common INTERFACE ${GMPXX} ${GMP})
  else()
    message(FATAL_ERROR "GMP not found; set paths or disable HDNUM_USE_GMP")
  endif()
endif()

### Chapters to include (add/remove as needed)
set(HDNUM_CHAPTERS ch1 ch2 ch3 ch4 ch5 ch6 ch7 ch8 ch9 ch10)
foreach(ch IN LISTS HDNUM_CHAPTERS)
  if(EXISTS "${CMAKE_CURRENT_LIST_DIR}/src/${ch}/CMakeLists.txt")
    add_subdirectory("src/${ch}")
  endif()
endforeach()
```

1.1.4.3 3) Per-chapter CMake with per-chapter output folders

- Each chapter decides which executables to build.
- **Per-chapter runtime output** goes to `build/bin/<chapter>/` (and per-config subfolders on multi-config generators).

```
### code_examples/my_solutions/src/ch1/CMakeLists.txt
set(CHAPTER ch1)

### Per-chapter output folder
set(OUT_DIR "${CMAKE_BINARY_DIR}/bin/${CHAPTER}")
set(CMAKE_RUNTIME_OUTPUT_DIRECTORY "${OUT_DIR}")
foreach(cfg IN ITEMS Debug Release RelWithDebInfo MinSizeRel)
  set(CMAKE_RUNTIME_OUTPUT_DIRECTORY_${cfg} "${OUT_DIR}/${cfg}")
endforeach()

### Helper to add ch1 examples
function(add_ch_example name)
  add_executable(${name} "${name}.cpp")
  target_link_libraries(${name} PRIVATE hdnum_common)
  # Optional: group in IDEs
  set_target_properties(${name} PROPERTIES FOLDER "${CHAPTER}")
endfunction()

### List executables in this chapter
add_ch_example(ch1_ode_demo)
add_ch_example(ch1_newton_demo)
```

Repeat a similar `CMakeLists.txt` in `src/ch2`, `src/ch3`, … adding that chapter's `.cpp` files.

1.1.4.4 4) Build commands (configure once, then build)

Single-config (Linux/Mint, Makefiles/Ninja):

```
cd code_examples/my_solutions
### configure (once per build dir or when options/CMakeLists change)
cmake -S . -B build -DCMAKE_BUILD_TYPE=Release               # + -DHDNUM_USE_GMP=ON if neede
### build (repeat as you edit sources)
cmake --build build -j
```

Multi-config (MSVC / Ninja Multi-Config):

```
cmake -S . -B build
cmake --build build --config Release
```

Outputs

```
build/bin/ch1/ch1_ode_demo
build/bin/ch1/ch1_newton_demo
build/bin/ch2/...
```

(or `build/bin/ch1/Release/...` on multi-config generators)

1.1.4.5 5) Using GMP (high precision) without editing submodule

- Install dev package: `sudo apt install -y libgmp-dev`
- Enable once at configure time (persists in the build cache):

```
cmake -S . -B build -DCMAKE_BUILD_TYPE=Release -DHDNUM_USE_GMP=ON
```

- CMake finds and links `gmpxx/gmp`, and defines `HDNUM_HAS_GMP=1` for all examples via `hdnum_common`.

  Tip: keep **two build dirs** if you switch often:

  ```
  cmake -S . -B build          -DCMAKE_BUILD_TYPE=Release -DHDNUM_USE_GMP=OFF
  cmake -S . -B build-gmp      -DCMAKE_BUILD_TYPE=Release -DHDNUM_USE_GMP=ON
  cmake --build build
  cmake --build build-gmp
  ```

1.1.4.6 6) Troubleshooting quickies

- `fatal error: hdnum.hh: No such file or directory` → `HDNUM_DIR` wrong; ensure submodule is initialized; include path points at the folder that **contains** `hdnum.hh`.
- `GMP not found` → install `libgmp-dev`; reconfigure; or pass custom `-DCMAKE_LIBRARY_PATH=/path` `-DCMAKE_INCLUDE_PATH=/path`.
- Changing `HDNUM_USE_GMP` or editing `CMakeLists.txt` → reconfigure (rerun the `cmake -S . -B build ...` step). Otherwise just `cmake --build build -j`.