# Krahasim i Strukturës "Production-style" për Shembullin e Oshilatorit Harmonik

## Python vs C++ vs Julia

Klaudio Peqini
Kursi: Modellimi në Fizikë

Departamenti i Fizikës

05 Mars, 2026

# Motivimi pedagogjik

- **Notebook-u** është laborator didaktik: i shpejtë për eksplorim, vizualizim dhe shpjegim.
- **Kodi "production"** kërkon: modularitet, parametrizim, testim, dokumentim dhe ekzekutim të riprodhueshëm.
- Ky shembull demonstron të njëjtin model fizik (oshilatori harmonik) në tre ekosisteme, me të njëjtën ide:

$$\textit{Modeli} \rightarrow \textit{Integratorët} \rightarrow \textit{Diagnostika} \rightarrow \textit{CLI/Rezultate}\ (\text{CSV})$$

# Çfarë ruajmë nga modeli?

Për oshilatorin harmonik (pa fërkim):

$$\dot{x} = v, \qquad \dot{v} = -\omega^2 x, \qquad E = \frac{1}{2}v^2 + \frac{1}{2}\omega^2 x^2.$$

- Krahasojmë integratorët: Euler, RK4, Velocity-Verlet (simetrik).
- Një kriter i thjeshtë cilësor: **drift-i relativ i energjisë** pas një kohe të gjatë.

# Tabela krahasuese (strukturë dhe ekzekutim)

| Aspekti | Python | C++ | Julia |
|---|---|---|---|
| Qëllimi didaktik | Kod i lexueshëm + testim i shpejtë | Performancë + disiplinë tipizimi | Sintaksë e thjeshtë + performancë |
| Struktura tipike | `src/`, `scripts/`, `tests/` | `include/`, `src/`, `build/` | `src/`, `scripts/`, `Project.toml` |
| Paketim / build | `pyproject.toml` (`pip -e .`) | `CMakeLists.txt` (`cmake`) | `Project.toml` (`-project=.`) |
| CLI (parametrizim) | ✓ (`argparse`) | ✓ (parser minimal) | ✓ (`ArgParse.jl`) |
| Output standard | CSV (`t,x,v`) | CSV (`t,x,v`) | CSV (`t,x,v`) |
| Vizualizim | ✓ (`matplotlib`) | ✗ (qëllimisht jashtë) | ✓ (`Plots.jl`) |
| Testim minimal | ✓ (`pytest`) | ✗ (mund të shtohet) | ✗ (mund të shtohet) |
| Tipizim statik | ✗ (opsional) | ✓ | ✓ (me tipe) |
| Shpejtësi (në përgjithësi) | Mesatare | Shumë e lartë | Shumë e lartë |

# Tabela krahasuese (praktika "production")

| **Praktikë** | **Python** | **C++** | **Julia** |
|---|---|---|---|
| Ndarje përgjegjësish (model/integrator/diagnostikë) | ✓ | ✓ | ✓ |
| Parametrizim pa ndryshuar kodin | ✓ | ✓ | ✓ |
| Riprodhueshmëri (komanda të qarta) | ✓ | ✓ | ✓ |
| Integrim me repo të kursit | Shumë i lehtë | I lehtë (me CMake) | I lehtë (me `Project.toml`) |
| Kthim i shpejtë pedagogjik (feedback i menjëhershëm) | Shumë i lartë | Mesatar | I lartë |
| Përshtatje për HPC | Mirë (me vektorizim/numba) | Shumë mirë | Shumë mirë |

# Komanda tipike (për demonstrim në klasë)

**Python**

- `python scripts/run_oscillator.py -omega 2.0 -dt 0.05 -t-end 50 -method verlet -out out.csv`
- `python scripts/plot_results.py -csv out.csv -omega 2.0`

**C++**

- `mkdir -p build && cd build && cmake .. && cmake -build . -j`
- `./oscillator -omega 2.0 -dt 0.05 -t_end 50 -method verlet -out out.csv`

**Julia**

- `julia -project=. scripts/run_oscillator.jl -omega 2.0 -dt 0.05 -t_end 50 -method verlet -out out.csv`
- `julia -project=. scripts/plot_results.jl -csv out.csv -omega 2.0`

# Mesazh për studentët

- **Notebook-u** na ndihmon të kuptojmë idetë shpejt dhe të ndërtojmë intuitë.
- **Kodi "production"** na mëson disiplinën: strukturë, riprodhueshmëri, testim dhe komunikim shkencor.
- E njëjta fizikë, por tre "kultura" inxhinierike: zgjidhni mjetin sipas qëllimit.

# Slide për repo-n

**Sugjerim për vendosje në monorepo (në `bundle_1/production_demos/`)**

- `python/` (`oscillator_prod_python`)
- `cpp/` (`oscillator_prod_cpp`)
- `julia/` (`oscillator_prod_julia`)

**Shënim:** Për të gjeneruar PDF në GitHub Actions (opsionale), mund të shtohet një workflow i thjeshtë LaTeX.