

# Why High-performance Computing?

(Lectures on High-performance Computing for Economists I)

---

Jesús Fernández-Villaverde<sup>1</sup> and Pablo Guerrón<sup>2</sup>

August 27, 2024

<sup>1</sup>University of Pennsylvania

<sup>2</sup>Boston College

# Computation in economics I

- Computing has become a central tool in economics:
  1. Macro → solution and estimation of dynamic equilibrium models, policy evaluation and forecast, ...
  2. Micro → computation of games, labor/life-cycle models, models of industry dynamics, study of networks, bounded rationality and agent-based models, ...
  3. Econometrics → machine learning, non-standard estimators, simulation-based estimators, large datasets, ...
  4. International/spatial economics → models with heterogeneous firms and countries, dynamic models of international trade, spatial models, economic consequences of climate change and environmental policies, ...
  5. Finance → asset pricing, non-arbitrage conditions, VaR, ...
  6. Economic history → processing of large sets of non-standard information, library data, historical counterfactuals, ...

- Computation helps, complements, and extends economic and econometric theory. [Judd \(1997\)](#).
- Economics is not different from other scientific and engineering fields (if anything, economics has been slow to embrace computation).
- Widespread movement: [On Computing by Paul S. Rosenbloom](#).
- Nowadays, computation in economics is also becoming key in:
  1. Policy making institutions.
  2. Regulatory agencies.
  3. Industry.

# Past, present, and future

- Move towards computation started already in the 1950s (estimation of simultaneous equations models, simple static GE models, ...).
- But it gathered speed after the 1980s (RBC research program, first-generation simulation estimators, structural estimation, interest rate models, ...).
- Most likely, the computational trend will increase over time:
  1. Drop in computing costs.
  2. Big data.
  3. Machine learning and AI.
  4. New hardware.
  5. Change in composition of the profession.

# Consequences for students

- This means that you will spend a substantial share of your professional career:
  1. Coding.
  2. Dealing with coauthors and research assistants that code.
  3. Reading and evaluating computational papers.
  4. Supervising/regulating people using computational methods.
- You want to lay solid foundations for the future: concrete tools will change, but fundamental ideas will not.

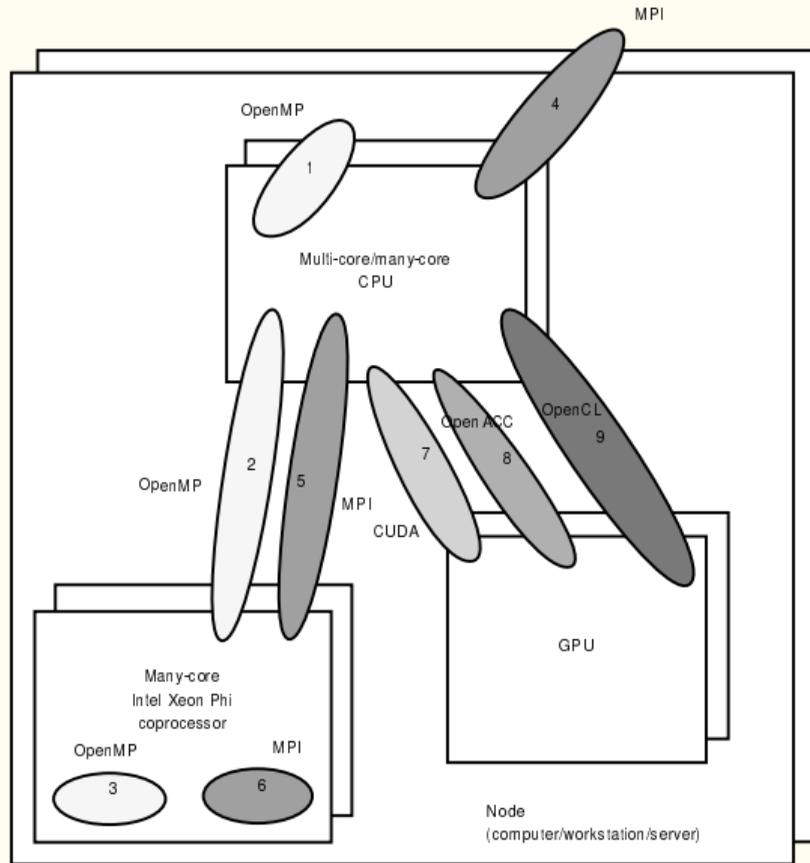
# High-performance computing

- High-performance computing (HPC) deals with scientific problems that require substantial computational power.
- Even simple problems in economics generate HPC challenges:
  1. Dynamic programming with several state variables.
  2. Highly non-linear DSGE models with many shocks.
  3. Problems with occasionally binding constraints.
  4. Complex asset pricing.
  5. Structural estimation.
  6. Frontier estimators without close-forms formulae.
  7. Handling large datasets.

# Parallel processing

- Usually, but not always, HPC involves the use of several processors:
  1. Multi-core/many-core CPUs (in a single machine or networked).
  2. Many-core coprocessors.
  3. GPUs (graphics processing units).
  4. TPUs (tensor processing units).
  5. FPGAs (field-programmable gate arrays).
- Most of these machines are available to all researchers at low prices.
- Nevertheless, we will also think about how to produce efficient serial code (although, following the most recent developments, we will not emphasize much vectorization).

# Parallel paradigms

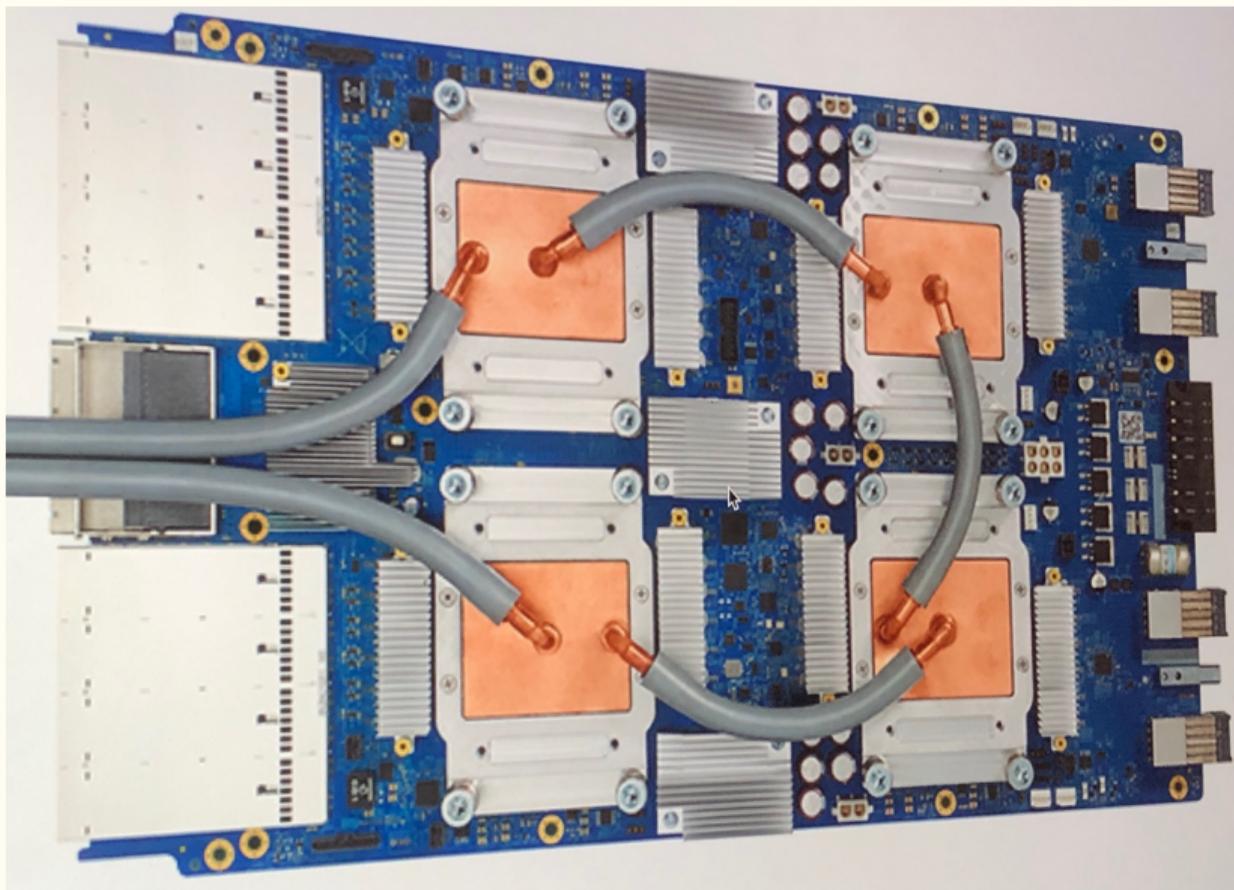




# Coprocessor



# TPUs





# Total time

- Often, HPC is framed regarding running time.
- In practice, coding and debugging time is likely more relevant than running time.
- We will spend considerable effort in discussing proper coding.
- Savings in development time are often first-order. Savings in running time are most times second-order.

**Adapted from Gen. Robert H. Barrow, USMC (27th Commandant of the US Marine Corps)**

Amateurs talk about the speed of their processors, but professionals study coding techniques.

## Some resources

- HPC carpentry: <https://hpc-carpentry.github.io/>.
- Livermore documentation and tutorials: <https://hpc.llnl.gov/training/>.
- HPC Wire: <https://www.hpcwire.com/>.
- Inside HPC: <https://insidehpc.com/>.
- *Introduction to High Performance Computing for Scientists and Engineers* by Georg Hager and Gerhard Wellein.
- *Parallel and High Performance Computing* by Robert Robey and Yuliana Zamora.
- *High Performance Computing: Modern Systems and Practices* by Thomas Sterling, Matthew Anderson, and Maciej Brodowicz.

- This set of notes does NOT cover:
  1. Theory of computation and complexity theory.
  2. Automata theory.
  3. Computer architecture.
  4. Computer arithmetic.
  5. Numerical analysis.
  6. Solution, estimation, and parallelization algorithms applied to economics.
  7.  $\text{\LaTeX}$  and  $\text{\BIBTeX}$ .