

BS1029: Computational Finance with C++

Lecture 0: Introduction and Course Structure

Panos Parpas
Imperial College London
p.parpas@imperial.ac.uk

General Information

- **Lecturer: Panos Parpas** (Huxley Building, Room 357, email: p.parpas@imperial.ac.uk)
 - **Office Hours:** generally available after the lecture, please let me know if you are coming to my office.
- **Teaching Assistants:** ()
 - ♦ **Buse Korkmaz** (buse.korkmaz18@imperial.ac.uk)
 - ♦ **Office Hours:** Please arrange via email
- Lecture slides, code, and exercises are available in Dropbox directory
- Weekly lectures, lab sessions: **Active participation is strongly encouraged**
- **One assessed coursework** (Week 5, **tbc**), counts for 50% of your mark for the course.
- **Exam** at the end of the course

Recommended Background

- Introduction to C++ (or similar)
- Derivative/Arbitrage pricing
- Risk management of derivatives (Greeks)
- Basic Stochastic Calculus
- Basic knowledge of Partial Differential Equations (as used in pricing models)
- Basic Linear Algebra

Aims of the Course

- Introduce the **three main areas** of numerical techniques used in computational finance:
 1. **Optimisation Methods**: Used for portfolio selection, model calibration, machine learning applications.
 2. **Monte Carlo Methods**: Used for derivatives pricing, risk management.
 3. **Numerical Methods for PDEs**: Used for derivatives pricing.
- **Introduce C++** through these numerical methods.

After this course:

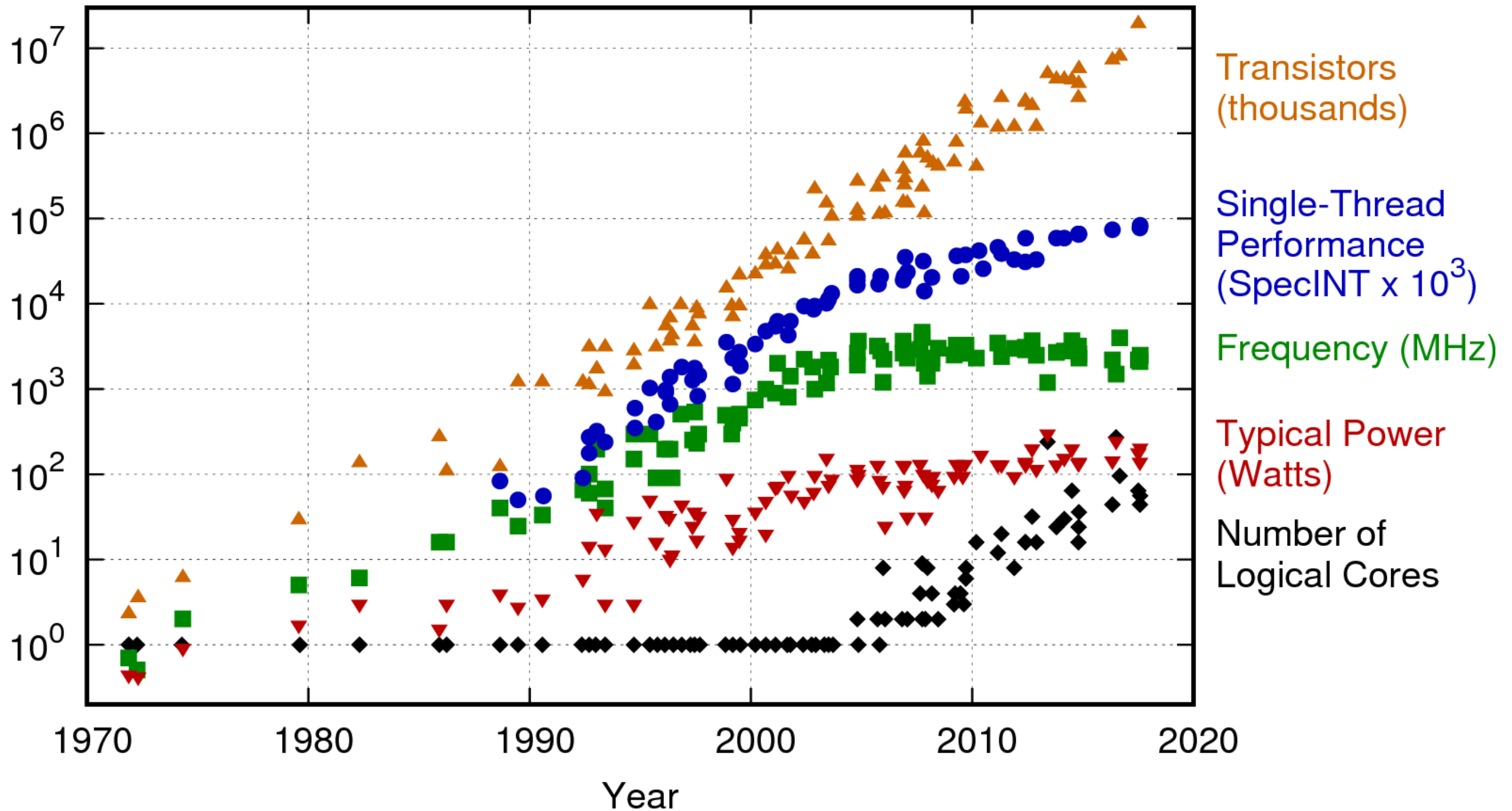
1. Be able to **understand the concepts** behind the main numerical methods in finance.
2. Be able to **formulate** portfolio optimisation, pricing problems, and risk management problems as mathematical models and then **develop efficient code to solve** them.
3. Be able to **design and understand object oriented design**
4. Be able to **read technical literature and undertake independent study** in numerical methods and C++.

Why Should you learn C++?

- Many tasks/jobs in finance industry require sophisticated mathematical and programming skills (quant analysts, developer, traders etc).
- C++ is one of the most widely used programming languages in finance (mainly for historical reasons)
- Concepts in C++ carry over in other languages too
- If you know C++ can pick up other (easier) languages more easily

Why Should you learn C++?

42 Years of Microprocessor Trend Data



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

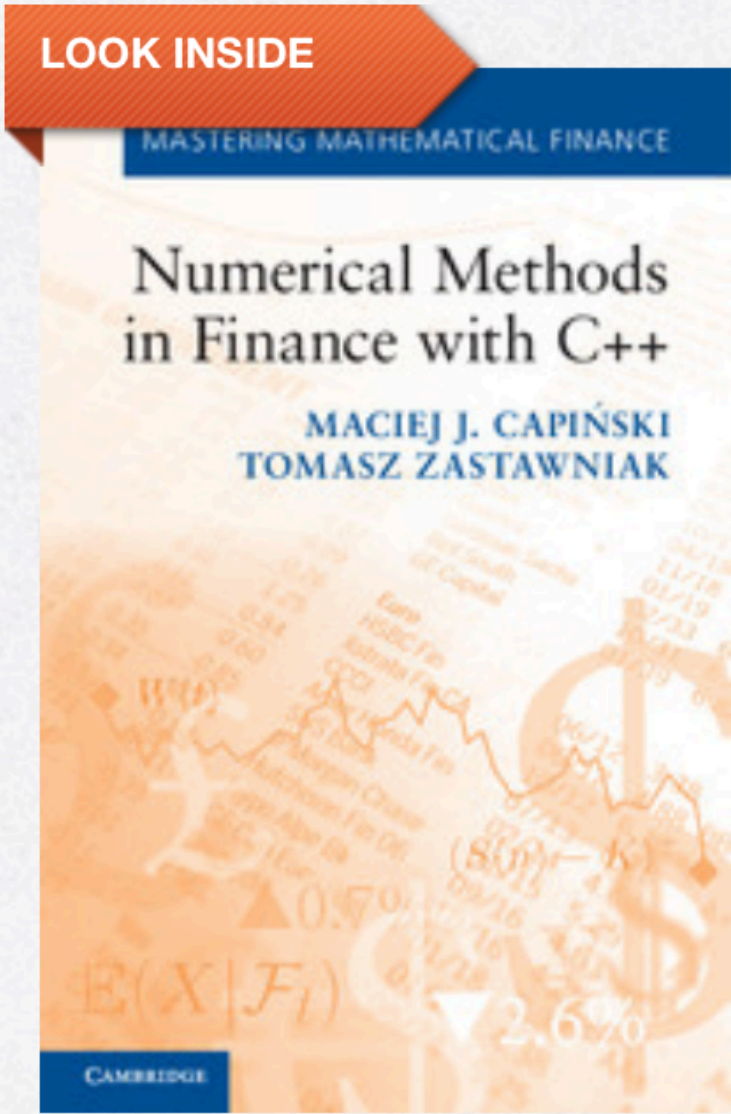
Why Should you learn C++?

Total					
	Energy		Time		Mb
(c) C	1.00	(c) C	1.00	(c) Pascal	1.00
(c) Rust	1.03	(c) Rust	1.04	(c) Go	1.05
(c) C++	1.34	(c) C++	1.56	(c) C	1.17
(c) Ada	1.70	(c) Ada	1.85	(c) Fortran	1.24
(v) Java	1.98	(v) Java	1.89	(c) C++	1.34
(c) Pascal	2.14	(c) Chapel	2.14	(c) Ada	1.47
(c) Chapel	2.18	(c) Go	2.83	(c) Rust	1.54
(v) Lisp	2.27	(c) Pascal	3.02	(v) Lisp	1.92
(c) Ocaml	2.40	(c) Ocaml	3.09	(c) Haskell	2.45
(c) Fortran	2.52	(v) C#	3.14	(i) PHP	2.57
(c) Swift	2.79	(v) Lisp	3.40	(c) Swift	2.71
(c) Haskell	3.10	(c) Haskell	3.55	(i) Python	2.80
(v) C#	3.14	(c) Swift	4.20	(c) Ocaml	2.82
(c) Go	3.23	(c) Fortran	4.20	(v) C#	2.85
(i) Dart	3.83	(v) F#	6.30	(i) Hack	3.34
(v) F#	4.13	(i) JavaScript	6.52	(v) Racket	3.52
(i) JavaScript	4.45	(i) Dart	6.67	(i) Ruby	3.97
(v) Racket	7.91	(v) Racket	11.27	(c) Chapel	4.00
(i) TypeScript	21.50	(i) Hack	26.99	(v) F#	4.25
(i) Hack	24.02	(i) PHP	27.64	(i) JavaScript	4.59
(i) PHP	29.30	(v) Erlang	36.71	(i) TypeScript	4.69
(v) Erlang	42.23	(i) Jruby	43.44	(v) Java	6.01
(i) Lua	45.98	(i) TypeScript	46.20	(i) Perl	6.62
(i) Jruby	46.54	(i) Ruby	59.34	(i) Lua	6.72
(i) Ruby	69.91	(i) Perl	65.79	(v) Erlang	7.20
(i) Python	75.88	(i) Python	71.90	(i) Dart	8.64
(i) Perl	79.58	(i) Lua	82.91	(i) Jruby	19.84

General Remarks

- This course combines three fields together: C++, numerical methods and finance.
- You need to have some understanding of all three to get the most out of this course.
- Can be achieved by hard work, active participation and problem solving.
- Use office hours, labs and lectures to ask questions.

Recommended Book



LOOK INSIDE

MASTERING MATHEMATICAL FINANCE

**Numerical Methods
in Finance with C++**

MACIEJ J. CAPIŃSKI
TOMASZ ZASTAWNIAK

Cambridge

Numerical Methods in Finance with C++

Part of [Mastering Mathematical Finance](#)

AUTHORS:
[Maciej J. Capiński](#), AGH University of Science and Technology, Krakow
[Tomasz Zastawniak](#), University of York

DATE PUBLISHED: August 2012

AVAILABILITY: In stock

FORMAT: Hardback

ISBN: 9781107003712

[Rate & review](#)

Available on-line from library

Source code available from book website & dropbox directory

Plan (Tentative)

- **Week 1:** Introduction & Binomial Pricing Model
- **Week 2:** Revisiting the binomial pricer: C++ classes, inheritance.
- **Week 3:** American Options: Advanced inheritance concepts
- **Week 4:** Nonlinear Optimisation (Theory)
- **Week 5:** Coursework discussion and lab
- **Week 6:** Nonlinear Optimisation (Implied Volatility, Function pointers, Function templates)
- **Week 7:** Monte Carlo Methods (Theory) & C++ Implementation
- **Week 8+9:** Finite Difference Methods (Theory & C++ implementation)

Check your programming Environment: `Main01.cpp`

- Use `make` tool to compile (worry about this later!)
- `#include <iostream>` part of the standard library to handle I/O in C++
- `using namespace std;` Prevent name clashes, try to remove it to see what happens. You will need to use `std::cout`, `std::cin`, `std::endl`
- `int main()`: entry point for program
- `return 0;` return a negative value if the program failed for some reason.