Programming for Business Tasks

Programming optimisation and operations research algorithms with Julia

Prof. Dr. Xavier Gandibleux

Nantes Université
Département Informatique – Faculté des Sciences et Techniques
France

May-June 2022

Short bio


NANTES
PARIS
Caen
Lisieux
Evreux
Dreux
Chartres
Versailles
St-Denis
Meaux
Senlis
Melun
Evry
Fontainebleau
Alençon
Le Mans
Laval
Rennes
Fougères
Dinan
Saint-Brieuc
Morlaix
Lannion
Brest
Quimper
Lorient
Vannes
Saint-Nazaire
Angers
Saumur
Tours
Blois
Orléans
Cholet
La Roche-sur-Yon
Poitiers
Châtellerault
Châteauroux
Niort
Guéret

Short bio

1. Teaching areas
   - Optimization: operations research; discrete and combinatorial optimization; multi-objective optimization.
   - Artificial intelligence: metaheuristics; multi-objective metaheuristics.
   - Computer science: Introduction to computer concepts; Julia programming language.
2. Research areas
   - Optimization: operations research; multi-objective optimization.
   - Artificial intelligence: metaheuristics.
   - Scientific computing: MIP solvers.
   - Applications: railway transportation, autonomous vehicles.

Short bio

1. Teaching areas
   - Optimization: operations research; discrete and combinatorial optimization; multi-objective optimization.
   - Artificial intelligence: metaheuristics; multi-objective metaheuristics.
   - Computer science: Introduction to computer concepts; Julia programming language.
2. Research areas
   - Optimization: operations research; multi-objective optimization.
   - Artificial intelligence: metaheuristics.
   - Scientific computing: MIP solvers.
   - Applications: railway transportation, autonomous vehicles.

Short bio

1. Teaching areas
   - Optimization: operations research; discrete and combinatorial optimization; multi-objective optimization.
   - Artificial intelligence: metaheuristics; multi-objective metaheuristics.
   - Computer science: Introduction to computer concepts; Julia programming language.
2. Research areas
   - Optimization: operations research; multi-objective optimization.
   - Artificial intelligence: metaheuristics.
   - Scientific computing: MIP solvers.
   - Applications: railway transportation, autonomous vehicles.

My timeline on generalist programming languages

1. ’80
   - Assembler (6502, Z80)
   - Basic, Pascal, Fortran 77, Cobol, C
2. ’90
   - CommonLisp
   - Ada95
3. later
   - (python, javascript, processing, Java, C++)
   - Julia

In my research I work with Julia and C

My timeline on generalist programming languages

1. ’80
   - Assembler (6502, Z80)
   - Basic, Pascal, Fortran 77, Cobol, C
2. ’90
   - CommonLisp
   - Ada95
3. later
   - (python, javascript, processing, Java, C++)
   - Julia

In my research I work with Julia and C

My timeline on generalist programming languages

1. ’80
   - Assembler (6502, Z80)
   - Basic, Pascal, Fortran 77, Cobol, C
2. ’90
   - CommonLisp
   - Ada95
3. later
   - (python, javascript, processing, Java, C++)
   - Julia

In my research I work with Julia and C

My timeline on generalist programming languages

Example: how to get the column sums of a matrix?

Assembly

```
main:
        push    rbp
        mov     rbp, r
sp
        push    rbx
        sub     rsp, 1
04
        lea     rax,
[rbp-80]
        mov     rdi, r
ax
        call    std::v
ector<std::vector<int,
std::allocator<int> >,
std::allocator<std::ve
ctor<int, std::allocat
or<int> > > >::vector
() [complete object co
nstructor]
        mov     QWORD
PTR [rbp-48], 2
        mov     QWORD
PTR [rbp-56], 2
```

C++

```
vector <int> result;
result.resize(cols);
for(size_t j = 0; j <
cols; ++j)
{
    int sum = 0;
    for(size_t i = 0;
i < rows; ++i)
    {
        sum += A[i]
[j];
    }
    result[j] = sum;
}
```

Julia

```
sum(A; dims=1)
```

(example from Fons van der Plas)

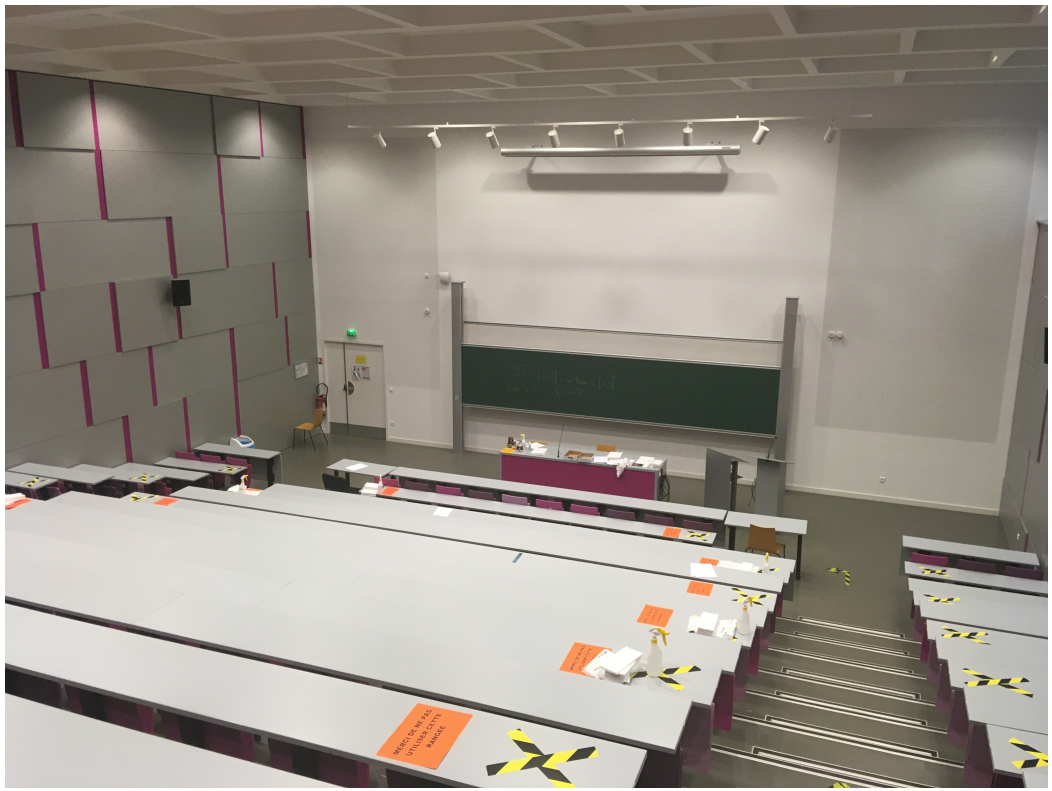
Profile of the public for this course

Level of fluency with

- a programming language..... ★ ★ ☆ ☆ ☆
- data structures..... ★ ☆ ☆ ☆ ☆
- LP/IP/MIP optimisation..... ★ ★ ★ ☆ ☆
- an algebraic modeling language... ★ ★ ☆ ☆ ☆

Programming with Julia

Overview, planning, organisation



Overview

Content of the course:

- Week 1 (23 to 25 of May): Learning Julia
  Toward implementing optimization models and optimisation algorithms in Julia.
- Week 2 (08 to 10 of June): Practicing Julia
  Project on solving an optimization problem.

Organisation

Lectures are at the following slots:

- week 1 (face-to-face): AM: 09h15 → 11h45 PM: 13h45 + 17h00
- week 2 (online): AM: 09h15 → 11h00 PM: 14h30 + 16h15
- Join through this Zoom link:
  `https://univ-nantes-fr.zoom.us/j/98374752334?pwd=eTQ5ckN4ZmR6UHgvTHJ0dEx1TGRPQT09`

Planning:

- Week 1 (23 to 25 of May): Learning Julia

  Introduction, software environment, REPL, packages, working with Julia. Values, variables, types, constants, print, input, comments, basic maths. Explicit optimization models with JuMP. Arrays (vectors, matrices, lists), tuples, dictionaries, sets, characters, strings. Control flow (conditionals, loops), functions (single line, anonymous, general), usual functions. Implicit and advanced optimization models with JuMP. Special topics (random numbers, search and sort, plotting). Data structures (composite). Multi-objective optimisation models with vOptGeneric. Case study.

- Week 2 (08 to 10 of June): Practicing Julia

Works and how to get a grade

No homework

No midterm/final examination

Proposed exercices to be due during the lectures

Grades will be computed as follow:

- 25%: case study (to be due on June 01)
- 75%: project (to be due on June 17)

Exercises, the case study and the project will be uploaded on moodle.

The case study and the project can be done in groups of 2 ~ 3 students.