StochDynamicProgramming.jl : a Julia package for multistage stochastic optimization.

V. Leclère, H. Gerard, F. Pacaud, T. Rigaut

July 6, 2016

Contents

Programming Language for Numerical Mathematics

We often tends to believe that making numerical code should be done in a two step process:

1. writing the algorithm in a high-level script like language (matlab, scilab, python...) to test the idea and variations on small to medium sized problem.
2. re-writing the algorithm in a low level language (C, C++, Fortran...) for numerical efficiency applicable on large scale problem.

Which means writing the algorithm twice.

Could we do it only once ?

Programming Language for Numerical Mathematics

We often tends to believe that making numerical code should be done in a two step process:

1. writing the algorithm in a high-level script like language (matlab, scilab, python...) to test the idea and variations on small to medium sized problem.
2. re-writing the algorithm in a low level language (C, C++, Fortran...) for numerical efficiency applicable on large scale problem.

Which means writing the algorithm twice.

Could we do it only once ?

Programming Language for Numerical Mathematics

We often tends to believe that making numerical code should be done in a two step process:

1. writing the algorithm in a high-level script like language (matlab, scilab, python...) to test the idea and variations on small to medium sized problem.
2. re-writing the algorithm in a low level language (C, C++, Fortran...) for numerical efficiency applicable on large scale problem.

Which means writing the algorithm twice.

Could we do it only once ?

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A “high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments”
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

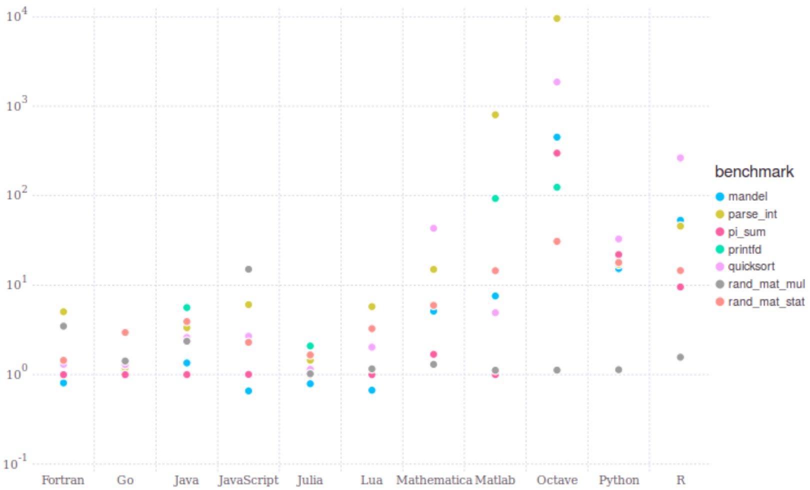
What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

What is Julia ?

- A beautiful woman name
- A "high-level, high-performance language for technical computing, with syntax that is familiar to users of other technical computing environments"
- Summary of best features:
  - Supports Windows, OSX, and Linux
  - Just in Time compiler
  - Automatic generation of efficient, specialized code for different argument types
  - Designed for parallelism and distributed computation
  - Powerful shell-like capabilities for managing other processes
  - Call python and C function
  - Built-in Package manager
  - MIT licensed: free and open source
  - all on github
  - some sugar-candy notational capacities

Is Julia really faster than alternatives ?



What is JuliaOpt ?

- JuliaOpt is an organization that brings together packages written in Julia that are related to optimization.
- Two modeller:
  - JuMP
  - Convex
- Interfaced with a number of solver (CPLEX, Gurobi, Mosek, Clp, GLPK, Knitro, Cbc, ECOS, Ipopt, NLopt, SCS).
- Optim contains the basic non-linear algorithms.

Contents

Problem statement

We consider the optimization problem

$$\begin{aligned} \min_{\pi} \quad & \mathbb{E}\left(\sum_{t=0}^{T-1} L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_t) + K(\mathbf{X}_T)\right) \\ s.t. \quad & \mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_t) \\ & \mathbf{U}_t = \pi_t(\mathbf{X}_t, \mathbf{W}_t) \end{aligned}$$

under the crucial assumption that $(\mathbf{W}_t)_{t\in\{1,\cdots,T\}}$ is a white noise

Stochastic Dynamic Programming

By the white noise assumption, this problem can be solved by dynamic programming, where the Bellman functions satisfy

$$\begin{cases} V_T(x) & = & K(x) \\ \hat{V}_t(x, w) & = & \min_{u_t \in \mathbb{U}} L_t(x, u_t, w) + V_{t+1} \circ f_t(x, u_t, w) \\ V_t(x) & = & \mathbb{E}\left(\hat{V}_t(x, \mathbf{W}_t)\right) \end{cases}$$

Indeed, an optimal policy for this problem is given by

$$\pi_t(x, w) \in \arg\min_{u_t \in \mathbb{U}} \left\{ L_t(x, u_t, w) + V_{t+1} \circ f_t(x, u_t, w) \right\}$$

SDDP : At the beginning of step k

At the beginning of step k, we suppose that we have, for each time step t, an approximation V_t^k of V_t satisfying

- $V_t^k \leq V_t$
- $V_T^k = K$
- V_t^k is convex

SDDP: Forward path: define a trajectory

- Randomly select a scenario $(\xi_0, \dots, \xi_{T-1}) \in \mathbb{W}^T$
- Define a trajectory $(x_t^{(k)})_{t=0,\dots,T}$ by

$$x_{t+1}^{(k)} = f_t(x_t^{(k)}, u_t^{(k)}, \xi_t)$$

where $u_t^{(k)}$ is an optimal solution of

$$\min_{u \in \mathbb{U}} L_t\left(x_t^{(k)}, u, \xi_t\right) + V_{t+1}^{(k)} \circ f_t\left(x_t^{(k)}, u, \xi_t\right)$$

- This trajectory is given by the optimal policy where V_t is replaced by $V_t^{(k)}$

SDDP: Backward path: add cuts

- For any t we want to add a cut to the approximation $V_t^{(k)}$ of V_t
- At time t solve, for any possible w,

$$\hat{\beta}_t^{(k+1)}(w) = \min_{x,u} \quad L_t(x,u,w) + V_{t+1}^{(k+1)} \circ f_t(x,u,w),$$
$$s.t \quad x = x_t^{(k)} \qquad [\hat{\lambda}_t^{(k+1)}(w)]$$

- Compute $\lambda_t^{(k+1)} = \mathbb{E}\left(\lambda_t^{(k+1)}(\mathbf{W}_t)\right)$ and $\beta_t^{(k+1)} = \mathbb{E}\left(\beta_t^{(k+1)}(\mathbf{W}_t)\right)$
- Add a cut

$$V_t^{(k+1)}(x) = \max\left\{V_t^{(k)}(x), \beta_t^{(k+1)} + \left\langle \lambda_t^{(k+1)}, x - x_t^{(k)} \right\rangle\right\}$$

- Go one step back in time: $t \leftarrow t-1$. Upon reaching $t=0$, we have completed step k of the algorithm

Contents

What is StochDynamicProgramming.jl ?

- A package written in Julia, part of JuliaOpt, aimed at solving multistage stochastic optimization.
- The focus is on Dynamic Programming like approaches.
- The idea is to define the SP model:
    - define the dynamic of the system f_t
    - define the costs of the problem L_t
    - define the constraints
    - define noise laws (assumed discrete and time-independent)
- Then it can be solved in three ways:
    - with an extensive formulation approach and calling a solver
    - with a DP approach (defining the discretization of state and controls)
    - with a SDDP approach

What is StochDynamicProgramming.jl ?

- A package written in Julia, part of JuliaOpt, aimed at solving multistage stochastic optimization.
- The focus is on Dynamic Programming like approaches.
- The idea is to define the SP model:
  - define the dynamic of the system f_t
  - define the costs of the problem L_t
  - define the constraints
  - define noise laws (assumed discrete and time-independent)
- Then it can be solved in three ways:
  - with an extensive formulation approach and calling a solver
  - with a DP approach (defining the discretization of state and controls)
  - with a SDDP approach

What is our philosophy ?

- Enable user to easily define the SP model.
- Allow to compare different approaches to tackle a given problem.
- Allow to benchmark variants of SDDP on a given problem.
- Invite academic contributions...

Wait, but is it difficult to install and use ?

Short answer: no.

Long answer:

1. Install Julia (see: http://julialang.org/downloads/).
2. launch julia.
   - Pkg.update()
   - Pkg.add(StochDynamicProgramming)
3. You are all set to go. I suggest looking at stock-example.jl as a tutorial. (notebooks are on the way).

Wait, but is it difficult to install and use ?

Short answer: no.

Long answer:

1. Install Julia (see: http://julialang.org/downloads/).
2. launch julia.
   - Pkg.update()
   - Pkg.add(StochDynamicProgramming)
3. You are all set to go. I suggest looking at stock-example.jl as a tutorial. (notebooks are on the way).

Conclusion and warning

- This is a free and open-source package for simple implementation of multistage stochastic optimization problem (with an MDP approach).
- It implements three way of solving the problem, with a focus on SDDP.
- It is easy to install and test.
- It is still an early version with lots of evolution to come in the next year...
- But what the package lacks in documentation or stability is compensated by a dedicated support team for our first users ;-)
- Don't hesitate to contact us for help, feedback or contributions !

Conclusion and warning

- This is a free and open-source package for simple implementation of multistage stochastic optimization problem (with an MDP approach).
- It implements three way of solving the problem, with a focus on SDDP.
- It is easy to install and test.
- It is still an early version with lots of evolution to come in the next year...
- But what the package lacks in documentation or stability is compensated by a dedicated support team for our first users ;-)
- Don't hesitate to contact us for help, feedback or contributions !

Conclusion and warning

- This is a free and open-source package for simple implementation of multistage stochastic optimization problem (with an MDP approach).
- It implements three way of solving the problem, with a focus on SDDP.
- It is easy to install and test.
- It is still an early version with lots of evolution to come in the next year...
- But what the package lacks in documentation or stability is compensated by a dedicated support team for our first users ;-)
- Don't hesitate to contact us for help, feedback or contributions !

Conclusion and warning

- This is a free and open-source package for simple implementation of multistage stochastic optimization problem (with an MDP approach).
- It implements three way of solving the problem, with a focus on SDDP.
- It is easy to install and test.
- It is still an early version with lots of evolution to come in the next year...
- But what the package lacks in documentation or stability is compensated by a dedicated support team for our first users ;-)
- Don't hesitate to contact us for help, feedback or contributions !

Conclusion and warning

- This is a free and open-source package for simple implementation of multistage stochastic optimization problem (with an MDP approach).
- It implements three way of solving the problem, with a focus on SDDP.
- It is easy to install and test.
- It is still an early version with lots of evolution to come in the next year...
- But what the package lacks in documentation or stability is compensated by a dedicated support team for our first users ;-)
- Don't hesitate to contact us for help, feedback or contributions !

Conclusion and warning

- This is a free and open-source package for simple implementation of multistage stochastic optimization problem (with an MDP approach).
- It implements three way of solving the problem, with a focus on SDDP.
- It is easy to install and test.
- It is still an early version with lots of evolution to come in the next year...
- But what the package lacks in documentation or stability is compensated by a dedicated support team for our first users ;-)
- Don't hesitate to contact us for help, feedback or contributions !