Predictive Analysis in Julia

Philip Thomas
SQuInT Breakout Session
21 February 2015

Brief overview of predictive analysis in Julia using the JuMP package for optimization, with a focus on Physics applications

Agenda

1. Introduction
2. Optimization and Operations Research
3. Julia and JuMP
4. Applications

Introduction

Resources for this talk

- Github.com/StaffJoy/jump-examples
- Vagrant is the easiest way to run the examples
- Follow-up at Blog.StaffJoy.com
- Shoot us a tweet: @StaffJoy

About Me

- WUSTL 2013 - BS in Systems Engineering, Physics
- Data telemetry and analysis for network security
- StaffJoy application makes teams more efficient by automating shift scheduling and management.
- We use JuMP extensively (10-50 models per workforce)

StaffJoy

Optimization and Operations Research

Optimization

Minimize or maximize an **objective function** subject to **constraints** by varying **decision variables**.

Decision variables are typically:

- Binary
- Integral
- Unconstrained

Example - Carrying Change

What is the lightest way to carry 99 cents in US coins?

$$\begin{array}{ll} \text{Min} & 2.5p + 5n + 2.268d + 5.670*q \\ \text{s.t.} & p + 5n + 10d + 25q \geq 99 \\ & p,\ n,\ d,\ q \geq 0 \\ & p,\ n,\ d,\ q \in Z \end{array}$$

Problem Classification

Type	Example objective function	Example Algorithm
Linear Programming	$x + y \quad \forall\ x, y \in R$	Simplex
Integer Programming	$x + y \quad \forall\ x, y \in Z$	Branch and bound
Convex Programming	$\sqrt{}(x + y) \quad \forall\ x, y \in R$	Interior point method
General Nonlinear Programming	$x*y \quad \forall\ x, y \in R$	Evolutionary algorithm

StaffJoy

Classic OR Optimization Applications

- Knapsack problem
- Routing problems
- Traveling salesman problem
- Scheduling

Physics Applications

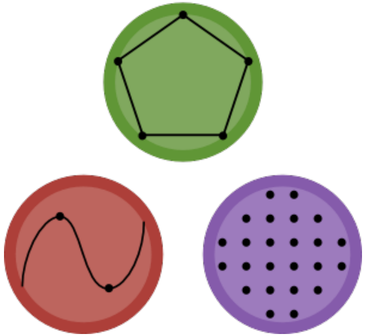
- Variational calculus (Power series)
- Lagrangian mechanics
- Ground energy states, e.g. repellant particles
- Power flow

Julia and JuMP

Julia

- Open source scientific computing language
- JIT compiler with dynamic Dispatch
- Package Manager
- Parallel and Distributed
- learnxinyminutes.com/docs/julia/

JuMP - Optimization in Julia



- **Ju**lia for **M**athematical **P**rogramming
- JuliaOpt.org
- Wrapper for low-level solvers
- Provides an extensible optimization metalanguage
- Currently supports LP, IP, NLP and more

StaffJoy

Why JuMP

- “Rosetta stone” for optimization
- High-level
- Extensible
- Supports a variety of low-level solvers, including commercial and open-source

Brief Intro to JuMP

Import	`using JuMP`
Model	`m = Model(solver=CbcSolver())`
Variables	`@defVar(m, x <= 0, Int)`
	`@defVar(m, y >= -4, Int)`
Constraints	`@addConstraint(m, x - 2y == -2)`
Objective	`@setObjective(m, Min, x-y)`
Solve	`solve(m)`

Applications

Example - Carrying Change

What is the lightest way to carry 99 cents in US coins?

$$\begin{array}{ll} \text{Min} & 2.5p + 5n + 2.268d + 5.670*q \\ \text{s.t.} & p + 5n + 10d + 25q \geq 99 \\ & p,\ n,\ d,\ q \geq 0 \\ & p,\ n,\ d,\ q \in Z \end{array}$$

macklemore.jl

```
using JuMP, Cbc

m = Model(solver=CbcSolver())

@defVar(m, pennies >= 0, Int)
@defVar(m, nickels >= 0, Int)
@defVar(m, dimes >= 0, Int)
@defVar(m, quarters >= 0, Int)

@addConstraint(m, 1 * pennies + 5 * nickels + 10 * dimes + 25 * quarters
>= 99)

@setObjective(m, Min,
    2.5 * pennies + 5 * nickels + 2.268 * dimes + 5.670 * quarters)

solve(m)
```

StaffJoy

macklemore.jl

```
Minimum mass: 22.68 grams
using:
    0 pennies
    0 nickels
    10 dimes
    0 quarters
```

Catenary

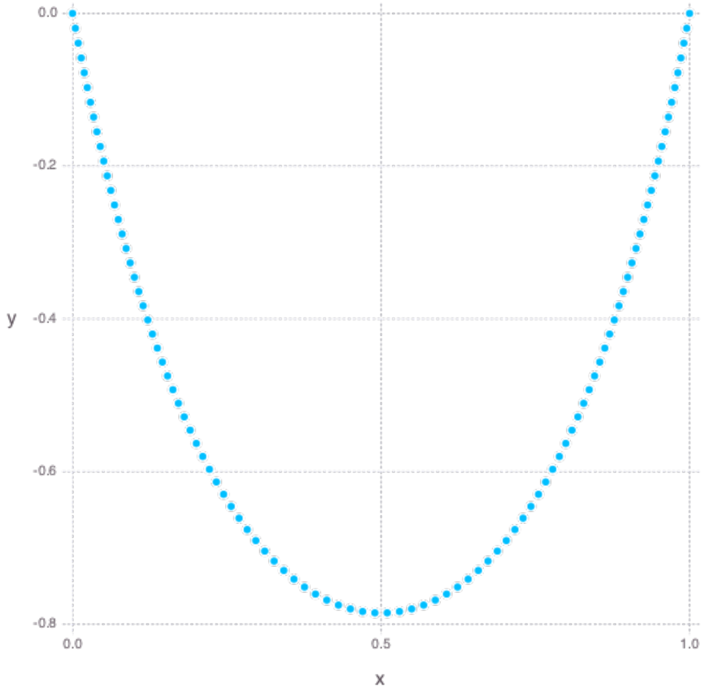
StaffJoy

catenary.jl

Code on Github

StaffJoy

catenary.jl



StaffJoy

Agenda

1. Introduction
2. Optimization and Operations Research
3. Julia and JuMP
4. Applications

StaffJoy

StaffJoy

Philip Thomas
Philip@StaffJoy.com