1. Introduction

- Logistics
- Optimization examples
- About the course
- Detailed example with code
- Assignment #0

Logistics

- Meeting times: Tue and Thu 1:00pm–2:15pm, Ingraham Hall, Room B10.
- **Instructor:** Laurent Lessard (laurent.lessard@wisc.edu)
  Office hours: Tue 3:00pm–4:00pm, Room TBA

  TA: Jui-Yang Chang (jchang38@wisc.edu)
  Office Hours: Mon and Thu, 9:45am–10:45am, Room TBA

  TA: Emad Sadeghi (ssadeghi@wisc.edu)
  Office Hours: Wed and Thu, 4:00–5:00pm, Room TBA

  All office hours start next week

More logistics

- Course website (notes, code samples, assignments):
  www.laurentlessard.com/teaching/524-intro-to-optimization/
  The website will be updated frequently.
- We will make extensive use of the online discussion forum at www.piazza.com. This is a great resource so take advantage of it! There will be homework hints and tips too.
- All assignments must be submitted **electronically**. We will use Gradescope for all submissions and grading. I will sign you up and you should receive a confirmation email.

Prerequisites

- Exposure to a bit of linear algebra and calculus (Math 320, Math 340, or equivalent). We will review all relevant concepts as we need them.
- Comfortable with programming (CS 302 or equivalent). It doesn't have to be Java, but if you've never written code in your life, this class will be very difficult for you.
- Very helpful (but not required): some experience with numerical computing (e.g. Matlab, Python, Julia, R)

For this course, we will exclusively use **Julia**. Julia is a new programming language for scientific computing similar to Matlab and Python. It's open-source and super fast!

Coursework

- Homework (45%): weekly assignments, a mix of theory questions and practical problem-solving (with coding in Julia). Roughly 10 assignments total.
- Midterm Exam (25%): Thur Mar 22, 7:15pm–9:15pm Mostly theory questions, covering the first 2/3 of the course (up until spring break).
- Final project (30%): You will work in groups of 3 or 4. More details to come...

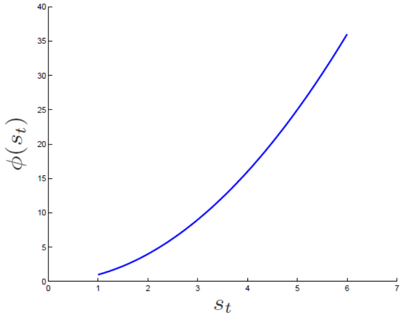
Textbook

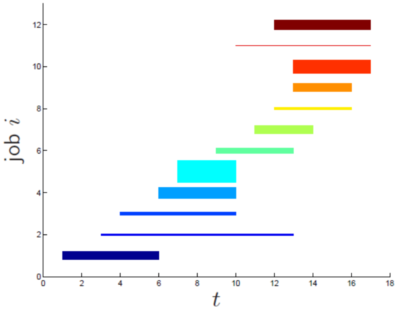
There is no required texbook for the course. However, there are plenty of good references. Here are a few:

- S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. The book is available for free here: http://stanford.edu/∼boyd/cvxbook/.
- H.P. Williams. *Model Building in Mathematical Programming*, 5th Edition. Wiley, 2013.
- R.L. Rardin. *Optimization in Operations Research*. Prentice Hall, 1998.

Ex. 1: Processor scheduling

- processor adjusts its speed s_t over time
- jobs must be scheduled. Each job has: arrival time, deadline, total work required.
- energy consumed $\phi(s_t)$ depends on speed

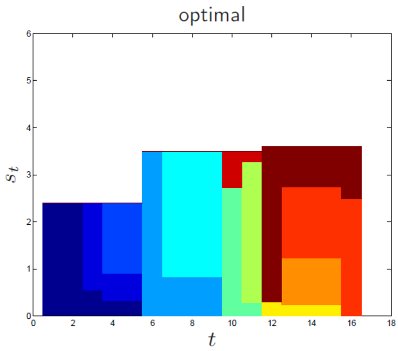


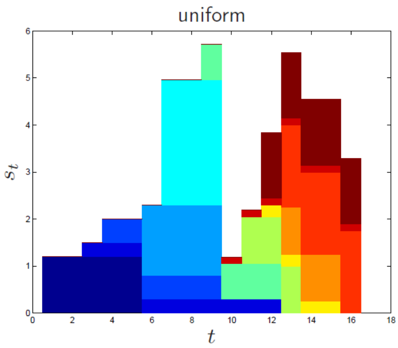


Source: S. Boyd (Stanford)

Ex. 1: Processor scheduling

- uniform schedule: requires 204.3 energy.
- optimal schedule: requires 167.1 energy.
- similar to other types of scheduling problems!
  e.g. employees, power networks,...

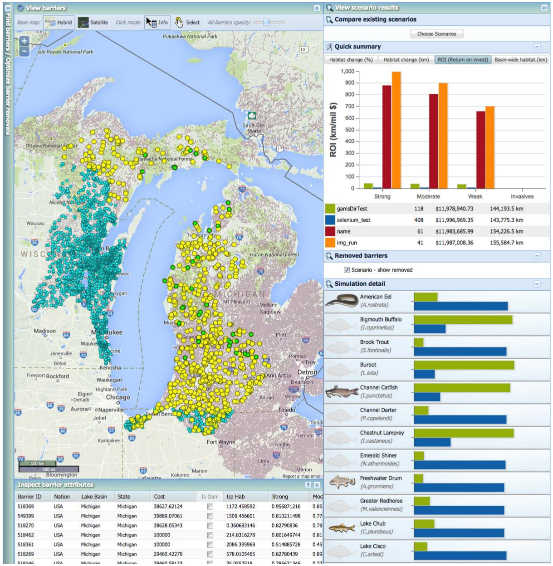




Source: S. Boyd (Stanford)

Ex. 2: Fish and Wildlife Service

- Great lakes basin data visualization
- 250k interdependent barriers on the network of rivers
- Complex optimization for budget constraints, specific fish guilds, invasive species.



View barriers
Base map: Hybrid Satellite
Click mode: Info Select
All-Barriers opacity:
Find barriers / Optimize barrier removals
View scenario results
Compare existing scenarios
Choose Scenarios
Quick summary
Habitat change (%)
Habitat change (km)
ROI (Return on invest)
Basin-wide habitat (km)
ROI (km/mil $)
Strong
Moderate
Weak
Invasives
Removed barriers
Scenario - show removed
Simulation detail
American Eel (A.rostrata)
Bigmouth Buffalo (I.cyprinellus)
Brook Trout (S.fontinalis)
Burbot (L.lota)
Channel Catfish (I.punctatus)
Channel Darter (P.copelandi)
Chestnut Lamprey (I.castaneus)
Emerald Shiner (N.atherinoides)
Freshwater Drum (A.grunniens)
Greater Redhorse (M.valenciennesi)
Lake Chub (C.plumbeus)
Lake Cisco (C.artedi)
Inspect barrier attributes

🟩 gamsDirTest	118	$11,978,940.73	144,193.5 km
🟦 selenium_test	408	$11,996,969.35	143,775.3 km
🟥 name	61	$11,983,685.99	154,226.5 km
🟧 img_run	41	$11,987,008.36	155,584.7 km

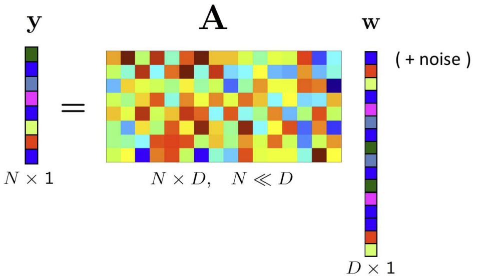
☑ Scenario - show removed

Barrier ID	Nation	Lake Basin	State	Cost	Is Dam	Up Hab	Strong	Mod
518369	USA	Michigan	Michigan	38627.62124	☐	1172.458592	0.956871216	0.85
549399	USA	Michigan	Michigan	39889.07061	☐	1509.466601	0.810211498	0.77
518270	USA	Michigan	Michigan	38628.05343	☐	0.360683146	0.82790836	0.78
518462	USA	Michigan	Michigan	100000	☐	214.8316278	0.801649744	0.81
518361	USA	Michigan	Michigan	100000	☐	2086.395968	0.514885728	0.45
518269	USA	Michigan	Michigan	29460.42279	☐	578.0105465	0.82780439	0.89
518146	USA	Michigan	Michigan	29460.58133	☐	35.0557518	0.786621346	0.77

Source: M. Ferris (UW–Madison), Fishwerks.

Ex. 3: Compressive sensing

- *N* outputs (observations) caused by *D* inputs (features)
  - Disease prediction: $\mathbf{y}$ are patients, $\mathbf{A}$ are gene markers, or
  - Imaging: $\mathbf{y}$ are image benchmarks, $\mathbf{A}$ are MRI modalities.

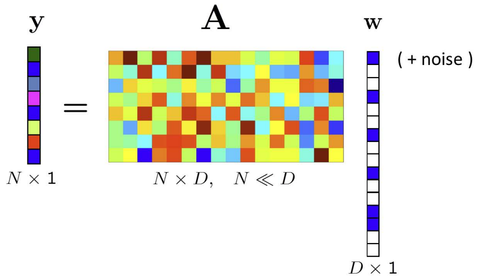


System is under-determined!

Source: R. Baranyuk (Rice) et.al.

Ex. 3: Compressive sensing

- N outputs (observations) caused by D inputs (features)
  - Disease prediction: $\mathbf{y}$ are patients, $\mathbf{A}$ are gene markers, or
  - Imaging: $\mathbf{y}$ are image benchmarks, $\mathbf{A}$ are MRI modalities.



System is over-determined! (Look for a sparse solution)

Source: R. Baranyuk (Rice) et.al.

Course objectives

1. Write down an algebraic formulation of a optimization problem that captures key elements of the problem. We call this an **optimization model**.
2. Perform standard mathematical manipulations and categorize the problem you're dealing with.
3. Develop an intuition for different types of models, their solutions, and the implications and trade-offs involved.
4. Write code in Julia to solve these problems.
5. Learn to interpret solutions, validate, perform sensitivity analysis, explore trade-offs, etc.

Course topics (by model type)

1. Linear programs (LP)
2. Least squares and quadratic programs (QP)
3. Second-order cone programs (SOCP)

   (exam and spring break)
4. Integer programs (IP) and mixed-integer programs (MIP)
5. Nonlinear programs (NLP and MINLP)

   (final project due)

Categories of optimization models

We can categorize the model types from the previous slide:

- Linear *vs.* Nonlinear
- Convex *vs.* Nonconvex
- Discrete *vs.* Continuous
- Deterministic *vs.* Stochastic

These categorizations have a **significant** impact on the computational tractability of a problem instance.

Gateway to more advanced courses

- Linear programming (CS 525, 526)
- Convex analysis (CS 727)
- Stochastic/Dynamic programming (CS 719, 723)
- Integer Optimization (CS 728)
- Nonlinear optimization (CS 726, 730)

Selected applied topics:

- Machine learning (CS 760, 761, 762)
- Optimal control (ECE 719, 819, 821)
- Robot motion planning (ME 739, 780)

Top Brass example

Top Brass Trophy Company makes large championship trophies for youth athletic leagues. At the moment, they are planning production for fall sports: football and soccer. Each football trophy has a wood base, an engraved plaque, a large brass football on top, and returns $12 in profit. Soccer trophies are similar except that a brass soccer ball is on top, and the unit profit is only $9. Since the football has an asymmetric shape, its base requires 4 board feet of wood; the soccer base requires only 2 board feet. At the moment there are 1000 brass footballs in stock, 1500 soccer balls, 1750 plaques, and 4800 board feet of wood. What trophies should be produced from these supplies to maximize total profit assuming that all that are made can be sold?

football soccer both

Top Brass data

Recipe for building each trophy

	wood	plaques	footballs	soccer balls	profit
football	4 ft	1	1	0	\$12
soccer	2 ft	1	0	1	\$9

Quantity of each ingredient in stock

	wood	plaques	footballs	soccer balls
in stock	4800 ft	1750	1000	1500

Top Brass model components

1. **Decision variables**
   - f: number of football trophies built
   - s: number of soccer trophies built
2. **Constraints**
   - $4f + 2s \leq 4800$ (wood budget)
   - $f + s \leq 1750$ (plaque budget)
   - $0 \leq f \leq 1000$ (football budget)
   - $0 \leq s \leq 1500$ (soccer ball budget)
3. **Objective**
   - Maximize $12f + 9s$ (profit)

Top Brass model (optimization form)

$$\begin{aligned} \underset{f,\,s}{\text{maximize}} \quad & 12f + 9s \\ \text{subject to:} \quad & 4f + 2s \le 4800 \\ & f + s \le 1750 \\ & 0 \le f \le 1000 \\ & 0 \le s \le 1500 \end{aligned}$$

- This is an instance of a **linear program** (LP), which is a type of optimization model.
- We have decision variables and parameters.

Top Brass model (generic)

$$\begin{aligned}
\underset{f,s}{\text{maximize}} \quad & c_1 f + c_2 s \\
\text{subject to:} \quad & a_{11} f + a_{12} s \le b_1 \\
& a_{21} f + a_{22} s \le b_2 \\
& \ell_1 \le f \le u_1 \\
& \ell_2 \le s \le u_2
\end{aligned}$$

- By changing the parameters, we create different *instance*.
- It's good practice to separate parameters (data) from the algebraic structure (model).

Top Brass code (IJulia notebook)

Top Brass.ipynb

Note: we did *not* separate the data from the model in this example! Next class, we will see how we can improve the code.

Assignment #0 (not graded)

1. Get up and running with Julia + IJulia + JuMP.
2. Load the file Top Brass.ipynb in IJulia and confirm that you can reproduce the results shown in class.
3. Experiment! Try changing the parameters and seeing if the solution still makes sense.

Getting started with Julia

Two options:

1. Install JuliaPro (personal version) for free here: https://juliacomputing.com/products/juliapro.html. This is the most reliable way to run Julia. Once up and running, run `Pkg.add("Clp")` to install the solver.

2. Run Julia inside your browser: https://www.juliabox.com/ Log in with any google account and create a new file using "Julia 0.6.2". Once you're in, use the package manager to install the `Clp` package and build it.

Julia tutorials

- Noteworthy differences between Julia and other languages:
  http://docs.julialang.org/en/stable/manual/noteworthy-differences/
- Useful tutorial:
  https://learnxinyminutes.com/docs/julia/
- Official documentation:
  http://docs.julialang.org/en/stable/
- JuMP documentation:
  http://www.juliaopt.org/JuMP.jl/0.18/

TO DO

- Bookmark the class website, take a look:
  www.laurentlessard.com/teaching/524-intro-to-optimization/
- Sign up on Piazza:
  https://piazza.com/wisc/spring2018/cseceisye524
- Get Julia up and running (in-class tutorial next Tuesday)