Disciplined Convex Optimization with CVXR

Anqi Fu Bala Narasimhan Stephen Boyd

EE & Statistics Departments

Stanford University

useR! Conference 2018

Convex Optimization

CVXR

Examples

Future Work

Outline

Convex Optimization

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, M \\ & Ax = b \end{array}$$

with variable $x \in \mathbf{R}^n$

- Objective and inequality constraints $f_0, \ldots, f_M$ are convex
- Equality constraints are linear

Convex Optimization

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, M \\ & Ax = b \end{array}$$

with variable $x \in \mathbf{R}^n$

- Objective and inequality constraints $f_0, \ldots, f_M$ are convex
- Equality constraints are linear

Why?

- We can solve convex optimization problems
- There are many applications in many fields, including machine learning and statistics

Convex Problems in Statistics

- Least squares, nonnegative least squares
- Ridge and lasso regression
- Isotonic regression
- Huber (robust) regression
- Logistic regression
- Support vector machine
- Sparse inverse covariance
- Maximum entropy and related problems
- ... and new methods being invented every year!

Domain Specific Languages for Convex Optimization

- Special languages/packages for general convex optimization
- CVX, CVXPY, YALMIP, Convex.jl
- Slower than custom code, but extremely flexible and enables fast prototyping

Domain Specific Languages for Convex Optimization

- Special languages/packages for general convex optimization
- CVX, CVXPY, YALMIP, Convex.jl
- Slower than custom code, but extremely flexible and enables fast prototyping

```
from cvxpy import *
beta = Variable(n)
cost = norm(X * beta - y)
prob = Problem(Minimize(cost))
prob.solve()
beta.value
```

Outline

Convex Optimization

CVXR

Examples

Future Work

CVXR

A modeling language in R for convex optimization

- Connects to many open source solvers
- Uses disciplined convex programming to verify convexity
- Mixes easily with general R code and other libraries

Outline

Ordinary Least Squares (OLS)

- minimize $||X\beta - y||_2^2$
- $\beta \in \mathbf{R}^n$ is variable, $X \in \mathbf{R}^{m \times n}$ and $y \in \mathbf{R}^m$ are constants

Ordinary Least Squares (OLS)

- minimize $||X\beta - y||_2^2$
- $\beta \in \mathbf{R}^n$ is variable, $X \in \mathbf{R}^{m \times n}$ and $y \in \mathbf{R}^m$ are constants

```
library(CVXR)
beta <- Variable(n)
obj <- sum_squares(y - X %*% beta)
prob <- Problem(Minimize(obj))
result <- solve(prob)
solution$value
solution$getValue(beta)
```

- `X` and `y` are constants; `beta`, `obj`, and `prob` are S4 objects
- `solve` method returns a list that includes optimal `beta` and objective `value`

Non-Negative Least Squares (NNLS)

- minimize $||X\beta - y||_2^2$ subject to $\beta \geq 0$

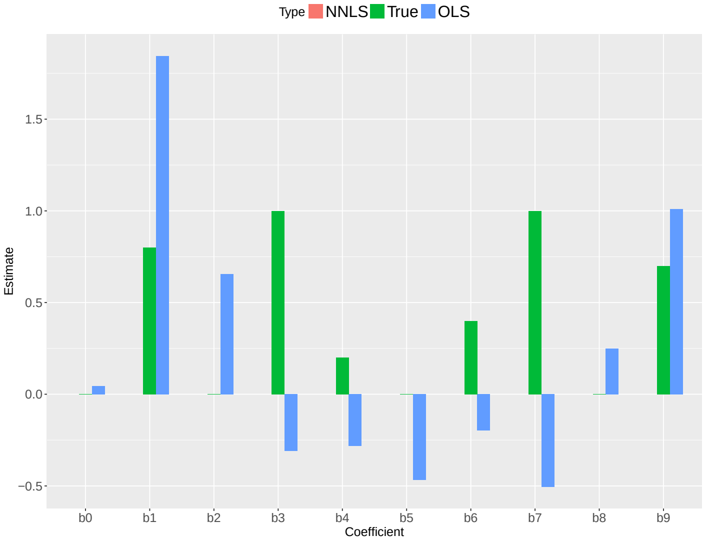
Non-Negative Least Squares (NNLS)

- minimize $||X\beta - y||_2^2$ subject to $\beta \geq 0$

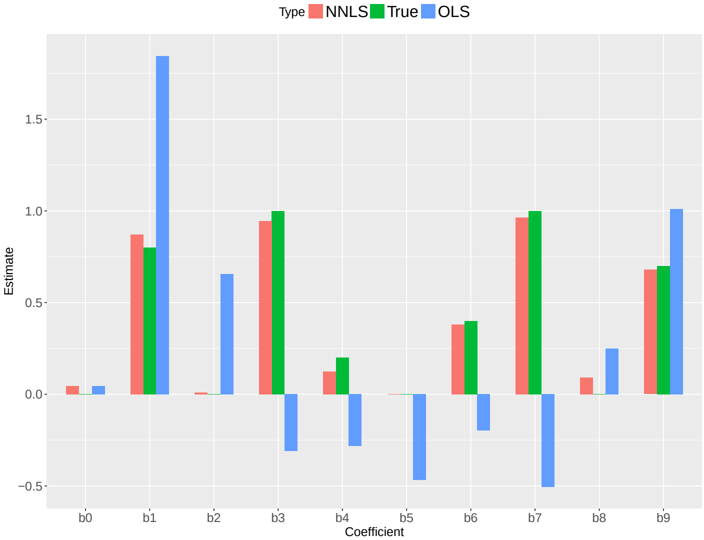
```
constr <- list(beta >= 0)
prob2 <- Problem(Minimize(obj), constr)
result2 <- solve(prob2)
result2$value
result2$getValue(beta)
```

- Construct new problem with list `constr` of constraints formed from constants and variables
- Variables, parameters, expressions, and constraints exist outside of any problem

True vs. Estimated Coefficients



True vs. Estimated Coefficients



Fastest Mixing Markov Chain

- Connected graph with vertices $\mathcal{V} = \{1, \ldots, n\}$ and edges $\mathcal{E}$
- Each vertex has a self-loop, all edges bidirected
- Find Markov chain with fastest mixing rate

Fastest Mixing Markov Chain

- Connected graph with vertices $\mathcal{V} = \{1, \ldots, n\}$ and edges $\mathcal{E}$
- Each vertex has a self-loop, all edges bidirected
- Find Markov chain with fastest mixing rate
- Mixing rate is **monotone** in second largest eigenvalue modulus (SLEM) of transition matrix $P \in \mathbf{R}^{n \times n}$
- Choose P to minimize SLEM $\mu(P)$, satisfy connectivity

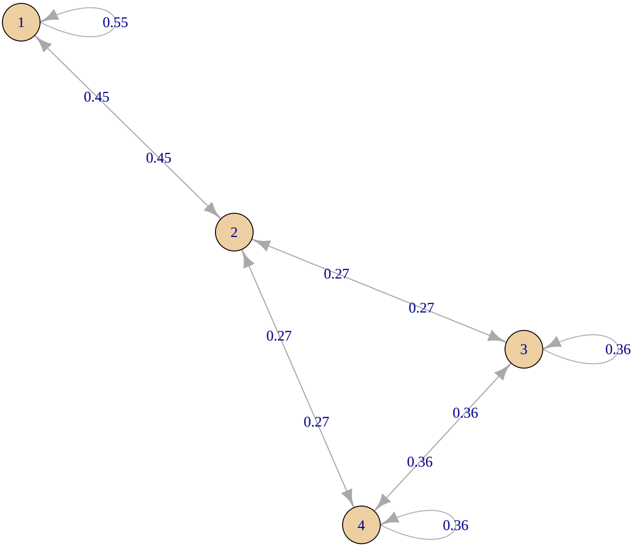
$$
\begin{array}{ll}
\text{minimize} & \mu(P) = \lambda_{\max}(P - \frac{1}{n}\mathbf{1}\mathbf{1}^\top) \\
\text{subject to} & P \geq 0, \quad P\mathbf{1} = \mathbf{1}, \quad P = P^\top \\
& P_{ij} = 0, \quad (i,j) \notin \mathcal{E}
\end{array}
$$

Fastest Mixing Markov Chain

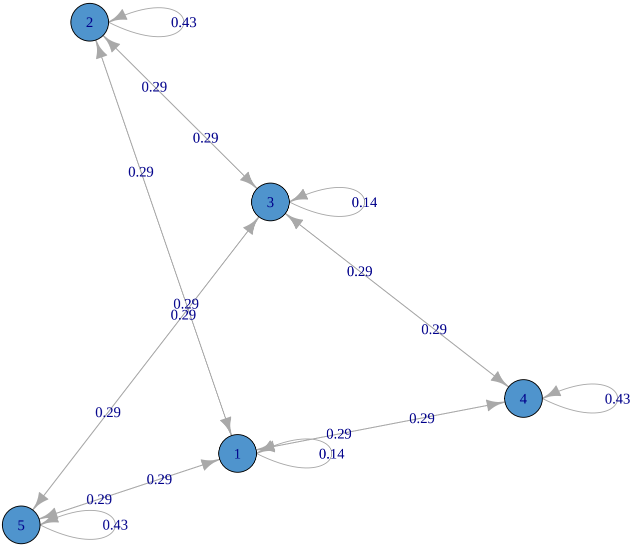
```
P <- Variable(n,n)
obj <- Minimize(lambda_max(P - 1/n))
constr <- list(P >= 0, P %*% ones == ones, P == t(P),
+    P[idxs] == 0)
prob <- Problem(obj, constr)
result <- solve(prob)
result$getValue(P)
```

- `lambda_max` is the maximum eigenvalue function (can also use spectral norm)
- `idxs` is matrix containing all unconnected vertices (i, j)

Triangle + 1 Edge



Bipartite 2 + 3



Outline

Convex Optimization

CVXR

Examples

Future Work

Future Work

- Flesh out convex functions in library
- Develop more applications and examples
- Add warm start support
- Further speed improvements

Official site: `cvxr.rbind.io`
CRAN page: `CRAN.R-project.org/package=CVXR`