ORF 522: Lecture 8

Linear Programming: Chapter 11
Game Theory

Robert J. Vanderbei

October 9, 2012

Slides last edited at 11:51am on Tuesday 9th October, 2012

Operations Research and Financial Engineering, Princeton University
http://www.princeton.edu/∼rvdb

Rock-Paper-Scissors

A two person game.

Rules: At the count of three declare one of:

Rock Paper Scissors

Winner Selection. Identical selection is a draw. Otherwise:

- Rock beats Scissors
- Paper beats Rock
- Scissors beats Paper

Check out Sam Kass' version: Rock, Paper, Scissors, Lizard, Spock

It was featured recently on The Big Bang Theory.

Payoff Matrix

Payoffs are *from* row player *to* column player:

$$A = \begin{array}{c} \\ R \\ P \\ S \end{array} \begin{array}{c} \begin{array}{ccc} R & P & S \end{array} \\ \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix} \end{array}$$

Note: Any *deterministic* strategy employed by either player can be defeated systematically by the other player.

Two-Person Zero-Sum Games

Given: $m \times n$ matrix A.

- *Row player* (rowboy) selects a *strategy* $i \in \{1, ..., m\}$.
- *Col player* (colgirl) selects a *strategy* $j \in \{1, ..., n\}$.
- Rowboy pays colgirl a_{ij} dollars.

Note: The rows of A represent deterministic strategies for rowboy, while columns of A represent deterministic strategies for colgirl.

Deterministic strategies are usually bad.

Randomized Strategies.

- Suppose rowboy picks i with probability y_i.
- Suppose colgirl picks j with probability x_j.

Throughout, $x = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$ and $y = \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix}^T$ will denote *stochastic vectors*:

$$\begin{aligned} x_j &\geq 0, \qquad j = 1, 2, \ldots, n \\ \sum_j x_j &= 1. \end{aligned}$$

If rowboy uses random strategy y and colgirl uses x, then *expected payoff* from rowboy to colgirl is

$$\sum_i \sum_j y_i a_{ij} x_j = y^T A x$$

Colgirl's Analysis

Suppose colgirl were to adopt strategy x.

Then, rowboy's best defense is to use y that minimizes the expected payment:

$$\min_y y^T A x$$

And so colgirl should choose that x which maximizes these possibilities:

$$\max_x \min_y y^T A x$$

Solving Max-Min Problems as LPs

Inner optimization is easy:

$$\min_y y^T A x = \min_i e_i^T A x$$

(e_i denotes the vector that's all zeros except for a one in the i-th position—that is, deterministic strategy i).

Note: Reduced a minimization over a *continuum* to one over a *finite set*.

We have:

$$\max \left(\min_i \; e_i^T A x\right)$$

$$\begin{aligned} \sum_j x_j &= 1, \\ x_j &\geq 0, \qquad j = 1, 2, \ldots, n. \end{aligned}$$

Reduction to a Linear Programming Problem

Introduce a scalar variable v representing the value of the inner minimization:

$$\max v$$

$$\begin{aligned} v &\le e_i^T A x, \qquad i = 1, 2, \ldots, m, \\ \sum_j x_j &= 1, \\ x_j &\ge 0, \qquad j = 1, 2, \ldots, n. \end{aligned}$$

Writing in pure matrix-vector notation:

$$\max v$$

$$\begin{aligned} ve - Ax &\le 0 \\ e^T x &= 1 \\ x &\ge 0 \end{aligned}$$

(e denotes the vector of all ones).

Finally, in Block Matrix Form

$$\max \begin{bmatrix} 0 \\ 1 \end{bmatrix}^T \begin{bmatrix} x \\ v \end{bmatrix}$$

$$\begin{bmatrix} -A & e \\ e^T & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} \begin{matrix} \leq \\ = \end{matrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$x \geq 0$$

$$v \ \text{free}$$

Rowboy's Perspective

Similarly, rowboy seeks y^* attaining:

$$\min_y \max_x \; y^T A x$$

which is equivalent to:

$$\begin{aligned} \min u & \\ ue - A^T y &\geq 0 \\ e^T y &= 1 \\ y &\geq 0 \end{aligned}$$

Rowboy's Problem in Block-Matrix Form

$$\min \begin{bmatrix} 0 \\ 1 \end{bmatrix}^T \begin{bmatrix} y \\ u \end{bmatrix}$$

$$\begin{bmatrix} -A^T & e \\ e^T & 0 \end{bmatrix} \begin{bmatrix} y \\ u \end{bmatrix} \begin{matrix} \geq \\ = \end{matrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$y \geq 0$$

$$u \ \text{free}$$

Note: Rowboy's problem is dual to colgirl's.

MiniMax Theorem

Let x^* denote colgirl's solution to her max–min problem.
Let y^* denote rowboy's solution to his min–max problem.
Then

$$\max_x \; {y^*}^T A x = \min_y \; y^T A x^*.$$

Proof. From *Strong Duality Theorem*, we have

$$u^* = v^*.$$

Also,

$$\begin{aligned} v^* &= \min_i e_i^T A x^* = \min_y y^T A x^* \\ u^* &= \max_j {y^*}^T A e_j = \max_x {y^*}^T A x \end{aligned}$$

QED
"As far as I can see, there could be no theory of games...without that theorem...I thought there was nothing worth publishing until the Minimax Theorem was proved" – John von Neumann

AMPL Model

```
set ROWS;
set COLS;
param A {ROWS,COLS} default 0;

var x{COLS} >= 0;
var v;

maximize zot: v;

subject to ineqs {i in ROWS}:
    sum{j in COLS} -A[i,j] * x[j] + v <= 0;

subject to equal:
    sum{j in COLS} x[j] = 1;
```

AMPL Data

```
data;
set ROWS := P S R;
set COLS := P S R;
param A: P  S  R:=
      P  0  1 -2
      S -3  0  4
      R  5 -6  0
    ;

solve;
printf {j in COLS}: "    %3s %10.7f \n", j, 102*x[j];
printf {i in ROWS}: "    %3s %10.7f \n", i, 102*ineqs[i];
printf: "Value = %10.7f \n", 102*v;
```

AMPL Output

```
ampl gamethy.mod
LOQO: optimal solution (12 iterations)
primal objective -0.1568627451
  dual objective -0.1568627451
        P 40.0000000
        S 36.0000000
        R 26.0000000
        P 62.0000000
        S 27.0000000
        R 13.0000000
Value = -16.0000000
```

Dual of Problems in General Form

Consider:

$$\begin{aligned} \max c^T x & \\ Ax &= b \\ x &\geq 0 \end{aligned}$$

Rewrite equality constraints as pairs of inequalities:

$$\begin{aligned} \max c^T x & \\ Ax &\leq b \\ -Ax &\leq -b \\ x &\geq 0 \end{aligned}$$

Put into block-matrix form:

$$\begin{aligned} \max c^T x & \\ \begin{bmatrix} A \\ -A \end{bmatrix} x & \begin{matrix} \leq \\ \leq \end{matrix} \begin{bmatrix} b \\ -b \end{bmatrix} \\ x &\geq 0 \end{aligned}$$

Dual is:

$$\begin{aligned} \min \begin{bmatrix} b \\ -b \end{bmatrix}^T \begin{bmatrix} y^+ \\ y^- \end{bmatrix} & \\ \begin{bmatrix} A^T & -A^T \end{bmatrix} \begin{bmatrix} y^+ \\ y^- \end{bmatrix} &\geq c \\ y^+, y^- &\geq 0 \end{aligned}$$

Which is equivalent to:

$$\begin{aligned} \min b^T(y^+ - y^-) & \\ A^T(y^+ - y^-) &\geq c \\ y^+, y^- &\geq 0 \end{aligned}$$

Finally, letting $y = y^+ - y^-$, we get

$$\begin{aligned} \min b^T y & \\ A^T y &\geq c \\ y & \quad \text{free.} \end{aligned}$$

Moral

- Equality constraints $\Longrightarrow$ free variables in dual.
- Inequality constraints $\Longrightarrow$ nonnegative variables in dual.

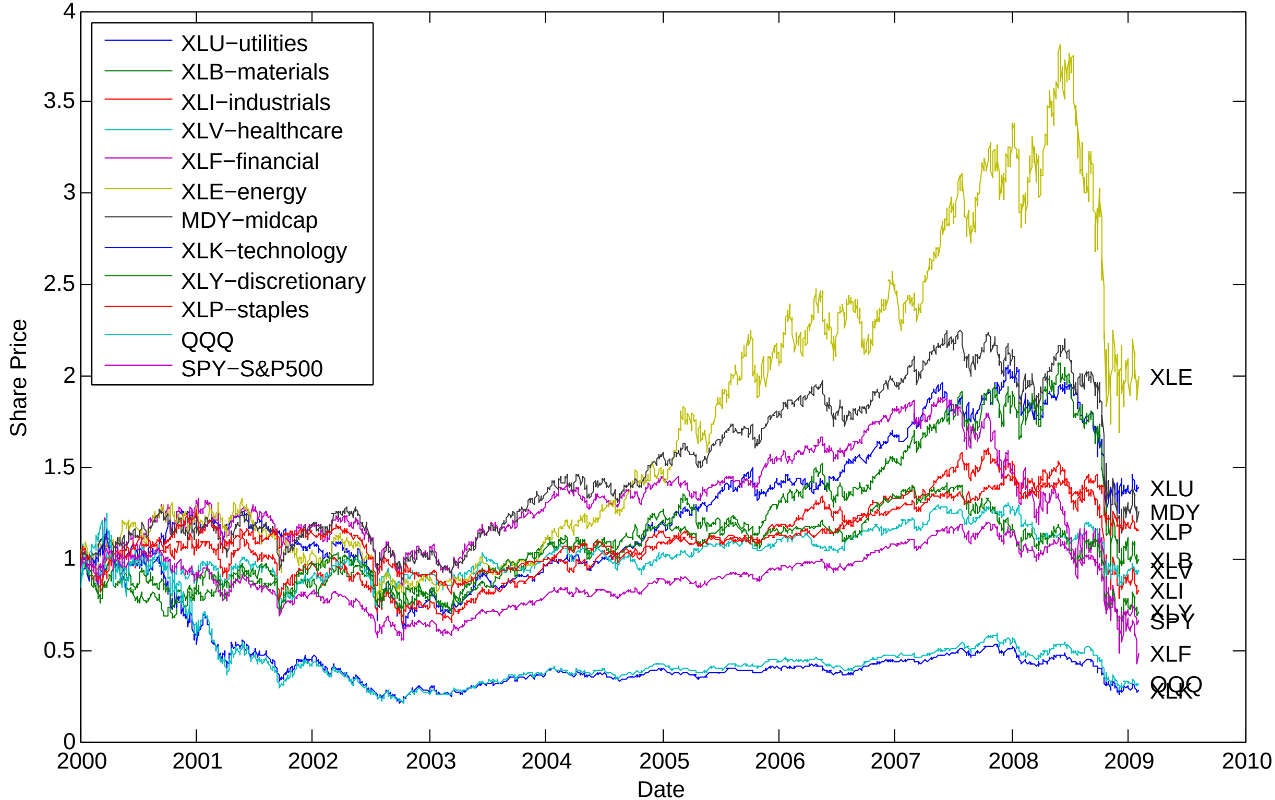
Corollary:

- Free variables $\Longrightarrow$ equality constraints in dual.
- Nonnegative variables $\Longrightarrow$ inequality constraints in dual.

A Real-World Example

The Ultra-Conservative Investor

Consider again the historical investment data ($S_j(t)$):



We can let let $R_{j,t} = S_j(t)/S_j(t-1)$ and view R as a payoff matrix in a game between *Fate* and the *Investor*.

Fate's Conspiracy

The columns represent pure strategies for our conservative investor.
The rows represent how history might repeat itself.
Of course, for tomorrow, Fate won't just repeat a previous year but, rather, will present some mixture of these previous years.
Likewise, the investor won't put all of her money into one asset. Instead she will put a certain fraction into each.
Using this data in the game-theory AMPL model, we get the following mixed-strategy percentages for Fate and for the investor.

Investor's Optimal Asset Mix:

XLP	90.7
QQQQ	9.3

Mean, old Fate's Mix:

2008-10-08	37.6
2008-11-28	62.4

The value of the game is the investor's expected return, 94.3%, which is actually a loss of 5.7%.

AMPL Model

```
set ROWS;
set COLS;
param A {ROWS,COLS} default 0;

var x{COLS} >= 0;
var v;

maximize zot: v;

subject to ineqs {i in ROWS}: sum{j in COLS} -A[i,j] * x[j] + v <= 0;

subject to equal: sum{j in COLS} x[j] = 1;

data;

set COLS := xlu xlb xli xlv xlf xle mdy xlk xly xlp qqqq spy;
set ROWS := include 'dates.out';

param A: xlu xlb xli xlv xlf xle mdy xlk xly xlp  qqqq spy:=
include 'amplreturn3.data' ;

solve;

printf "Investor's strategy\n";
printf {j in COLS: x[j]>0.0005}: "    %40s %5.1f \n", j, 100*x[j];
printf "\n";
printf "God's strategy\n";
printf {i in ROWS: ineqs[i]>0.0005}: "    %40s %5.1f \n", i, 100*ineqs[i];
```