ESTM 60203: Introduction to Operations Research

Jeffrey Kantor
Univ. of Notre Dame

November 5, 2009
Preface

These are preliminary notes for a 10 lecture module entitled “Introduction to Operations Research” to be offered Fall, 2009, for an audience of first year graduate students in the ESTEEM program at Notre Dame. An early and longer version of these notes were prepared for a full semester course offered in the Fall of 2008. This is a work in progress, suggestions and corrections are welcome.
Mission Statement

This module introduces students to the modeling of typical process operations, optimal solution of those models using analytical and computational techniques, and strategies for planning and operation under uncertainty. Students will gain the modeling and computational skills useful for real-world applications, and understand the role of modeling, information, and computation in range of process applications.
ESTM 60203: Introduction to Operations Research
Lecture 1: Course Overview & Introduction to Linear Optimization

Jeffrey Kantor
University of Notre Dame

November 6, 2009
Agenda

- Overview
  - What do mean by Operations Research?
  - Organization
    - Class List/Roster/Office Hours
    - Topics
    - Lecture Schedule

- Introductory Example - Giapetto’s Workshop
  - Formulation of a Linear Programming Problem
  - Graphical Analysis
  - Computational Approaches
    - Excel and Solver
    - Matlab
    - Algebraic Modeling Languages
Operations – What are They?

Examples of Process Operations

- Purchasing and delivering fuels to satisfy needs of a complex campus utility. [http://www.fuelsmanager.com/assets/SupplyChain_0506_web.jpg](http://www.fuelsmanager.com/assets/SupplyChain_0506_web.jpg)
- Managing assets of a complex logistics company

Operations – any set of transactions or activities intended to produce value. May range from simple (e.g., managing simple inventories) to complex (a vertically integrated commodity chemicals company).
Optimization – What is it?

Optimization (also known as Mathematical Programming) solves problems such as

1. What blend of raw materials are needed to meet production requirements?
2. How should a process be designed and operated to maximize expected profit?
3. How should I schedule the production of goods from several plants to meet contractual requirements while minimizing cost?
4. For a given level of business risk, what mix of investments will maximize expected return?
5. How do I plan and operate a process with uncertainties in process behavior, prices, and product demand?
Topics

The course provides an introduction to computational methods for operation research with application to process systems. Topics include:

1. Introduction to Optimization
   - Linear and Quadratic Programming
   - Computational Systems including Matlab, Spreadsheets, and Algebraic Modeling Languages (AMPL/GMPL)
   - Blending Problems
   - Network flow & Transportation Problems

2. Operations
   - Job Shops and Flowshops
   - Scheduling and Assignment

3. Managing Uncertainty
   - Diversification and Portfolio management
   - Multistage decision making
   - Planning with Stochastic Programming
What you should learn

- Mathematical programming techniques, including mixed integer programming, dynamic programming, and stochastic programming
  - Use of modeling languages and state of the art solvers

- Applications
  - How to formulate and solve optimization problems arising in operations

- New Insights
  - Role of models and analysis in operations
  - Role of information and optimization in decision support
  - Managing uncertainty and the value of information
Learning Activities

- Lectures and Demonstrations
- Directed readings from textbooks and primary sources
- Homework exercises
- Projects
- Examination
## Evaluation / Assessment

<table>
<thead>
<tr>
<th>Activity</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lectures &amp; Demos</td>
<td>Participation (10%)</td>
</tr>
<tr>
<td>Directed Readings</td>
<td></td>
</tr>
<tr>
<td>Homework</td>
<td>15%</td>
</tr>
<tr>
<td>Semester Project</td>
<td>25%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>25%</td>
</tr>
</tbody>
</table>
Academic Integrity

From the *Faculty Guide to the Academic Code of Honor* (www.nd.edu/~hnrcode):

**Honor Code Pledge**

“As a member of the Notre Dame community, I will not participate in or tolerate academic dishonesty”
Academic Integrity (cont.)

Homework  Discuss homework problems with one another, but the work you turn under your name must be your own. Computer programs, problem solutions, written work, for example, must be individually prepared by you.

Exams  Examinations are scheduled at the start of the semester. During an examination any communications or questions should be directed to me or my designate. You may not give or receive aid from others in the classroom during an examination.

Professionalism  Academic integrity includes responsible and professional behavior, regular class attendance, common courtesy, timely performance and respect for due dates, scheduled exams, and appointments.
## Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Nov 6 Linear Optimization</td>
</tr>
<tr>
<td>M</td>
<td>Nov 16 Blending &amp; Mixture Problems</td>
</tr>
<tr>
<td>W</td>
<td>Nov 18 Network Flow and Transportation</td>
</tr>
<tr>
<td>F</td>
<td>Nov 20 Project and Process Management</td>
</tr>
<tr>
<td>M</td>
<td>Nov 23 Discrete Optimization</td>
</tr>
<tr>
<td>W</td>
<td>Nov 25 Job and Flow Shops</td>
</tr>
<tr>
<td>M</td>
<td>Nov 30 Scheduling and Assignment</td>
</tr>
<tr>
<td>W</td>
<td>Dec 2 Uncertainty, Risk, and Diversification</td>
</tr>
<tr>
<td>M</td>
<td>Dec 7 Inventory Management with Uncertain Demand</td>
</tr>
<tr>
<td>W</td>
<td>Dec 9 Optimization under Uncertainty</td>
</tr>
</tbody>
</table>
Part I: Linear Optimization
Outline for Linear Optimization

Part 1: Lecture: Formulation and Modeling of Linear Programs
- Example problem - Giapetto’s Workshop
- What we seek from analysis and computation
- Sensitivity Analysis

Part 2: Appendix: Mathematics of Linear Programming
- Characterizing the optimum
- Simplex Method - Active Set Method

Part 3: Lectures: Applications
- Blending, Mixture, and Materials Problems
- Refinery Operations
- Network Flow and Transportation
- Project Management
Giapetto’s Woodcarving

Giapetto’s Workshop produces two types of wooden toys:

- **Soldiers** - Each sells for $27, and requires
  - $10 of raw materials, and $14 of labor and overhead
  - 2 hours of finishing labor, and 1 hour of carpentry labor

- **Train** - Each sells for $21, and requires
  - $9 of raw materials, and $10 of labor and overhead
  - 1 hour of finishing labor, and 1 hour of carpentry labor

**Constraints**

- 100 finishing hours, 80 carpentry hours available weekly.
- At most 40 toy soldiers will be sold each week.

What is the maximum achievable weekly profit?

---

1Example from Winston, *Operations Research*
Decision Variables

What are the decision variables?

$x_1$: Weekly production of toy soldiers.

$x_2$: Weekly production of toy trains.
Objective Function

What quantity are we trying to maximize?

\[
\text{Revenue} = 27x_1 + 21x_2 \\
\text{Cost} = (10 + 14)x_1 + (9 + 10)x_2
\]

Maximize

\[
\text{Profit} = \text{Revenue} - \text{Cost} \\
= (27 - 10 - 14)x_1 + (21 - 9 - 10)x_2 \\
= 3x_1 + 2x_2
\]
How are decisions constrained?

Production must be a non-negative (integer):

\[ x_1, x_2 \geq 0 \]

Production cannot exceed demand (if there is no storage for unsold goods):

\[ x_1 \leq 40 \]

Resources are limited:

- Limit of 100 hours/wk of finishing labor:
  \[ x_1 + x_2 \leq 100 \]

- Limit of 80 hours/wk of carpentry labor:
  \[ 2x_1 + x_2 \leq 100 \]
Problem Summary

The optimization problem is to maximize profit by selecting 2 decision variables subject to 5 constraints:

Maximize

\[ \text{Profit} = 3x_1 + 2x_2 \]

Subject to

\[ x_1 \geq 0 \quad \text{Non-negative production} \]
\[ x_2 \geq 0 \quad \text{Non-negative production} \]
\[ x_1 \leq 40 \quad \text{Maximum Demand for Soldiers} \]
\[ 2x_1 + x_2 \leq 100 \quad \text{Finishing Constraint} \]
\[ x_1 + x_2 \leq 80 \quad \text{Carpentry Constraint} \]
Single Product Strategies

Soldiers Only Strategy \((x_2 = 0)\)
Maximize \(3x_1\) subject to

\[
\begin{align*}
x_1 & \geq 0 \\
x_1 & \leq 40 \\
2x_1 & \leq 100 \\
x_1 & \leq 80
\end{align*}
\]

Solution: \(x_1 = 40\) for a profit of $120/wk

Trains Only Strategy \((x_1 = 0)\)
Maximize \(2x_2\) subject to

\[
\begin{align*}
x_2 & \geq 0 \\
x_2 & \leq 100 \\
x_2 & \leq 80
\end{align*}
\]

Solution: \(x_2 = 80\) for a profit of $160/wk

Would a mixed strategy yield a higher profit?
Graphical Analysis

- Solutions will be found in the positive quadrant.
- Constraints determine the feasible region.
The limited demand for toy soldiers \((x_1 \leq 40)\) limits solutions to the left of the line \(x_1 = 40\).
Finishing Labor Constraint

The region satisfying the constraint on finishing labor \((2x_1 + x_2 \leq 100)\) lies 'below' the line \(2x_1 + x_2 = 100\).
Carpentry Labor Constraint

Solutions satisfying the carpentry labor constraint lie below the line $x_1 + x_2 = 80$. 
Feasible Region

The *feasible region* is the set of decision variable satisfying all constraints, which, for an LP, is

- Convex
- If finite, a polyhedron, or
- If infinite, a cone.

An LP is *feasible* if the feasible region is not empty. Otherwise it is *infeasible.*
Profit Contours

Contours of constant profit $p$ satisfy $3x_1 + 2x_2 = p$. 
Extrema are found at vertices representing the intersection of active constraints.
What did we learn from this Example?

- Formulation of Decision variables, linear constraints, and linear objective function.
- Review 2-D analysis of toy Linear Programs.
- Key concepts in the solution of Linear Programs: Feasibility, active constraints, polyhedral feasibility set, vertices.
Solving Linear Programs
Historical Perspectives

1947 George Dantzig publishes Simplex Method for Linear Programming. Works well in practice, but later found to have Exponential complexity in the worst case.


1984 New York Times announces Karmarkar’s interior point method

- "Startling theoretical breakthrough" - Bell Labs
- "It’s big dollars" - American Airlines
- "It’s important when conditions are changing rapidly, for example, the price of crude oil" - Exxon
Computational Solutions

What’s the right tool for the job?

- Spreadsheet “Solvers”
  - Easy to use, nearly universal access
  - Integrated with standard desktop software
  - Not always robust, may need to buy additional solvers

- Algebraic Modeling Languages
  - A natural language for describing optimization problems
  - Usable with multiple solvers
  - Easy to modify parameters and data
  - “Industrial Stength”

- Matlab, Optimization Toolbox, and other software libraries
Spreadsheet “Solvers”

The “Solver” function of modern spreadsheets offer access to a remarkable range of optimization capabilities and probably the most widely used tools for optimization ever developed.

**Excel**  Microsoft Excel has offered linear programming since 1990 through the “Solver” function. Frontline Systems [http://www.solver.com/pressinfo.htm](http://www.solver.com/pressinfo.htm) markets tools to extend functionality.

**Gnumeric**  Gnumeric (the GNU spreadsheet) has a “Solver” similar to Excel based on the open source lp_solve and glpk linear programming packages.

**OpenOffice**  Solvers for the OpenOffice spreadsheet are undergoing rapid development. There is a plug-in tool for access to CVXOPT [http://abel.ee.ucla.edu/cvxopt](http://abel.ee.ucla.edu/cvxopt) for convex programming.
Toy Problem - Giapetto’s Woodcarving²

Giapetto’s Workshop produces two types of wooden toys:

- **Soldiers** - Each sells for $27, and requires
  - $10 of raw materials, and $14 of labor and overhead
  - 2 hours of finishing labor, and 1 hour of carpentry labor

- **Train** - each sells for $21, and requires
  - $9 of raw materials, and $10 of labor and overhead
  - 1 hour of finishing labor, and 1 hour of carpentry labor

- **Constraints**
  - 100 finishing hours, 80 carpentry hours available weekly.
  - At most 40 toy soldiers will be sold each week.

What is the maximum achievable weekly profit?²

²Example from Winston, *Operations Research*
Decision Variables

Decision variables are quantities which, once specified, determine the state of the system under consideration.

Exercise

Before going further, take a few minutes to think about this example. Write down a list of plausible decision variables, a short description of including units and upper and lower bounds. Then compare your response to the table on the next slide.
### Decision Variables

<table>
<thead>
<tr>
<th>Var</th>
<th>Description</th>
<th>Units</th>
<th>LB</th>
<th>UB</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>Production of toy soldiers</td>
<td>[units/week]</td>
<td>$\geq 0$</td>
<td>$\leq 40$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>Production of toy trains</td>
<td>[units/week]</td>
<td>$\geq 0$</td>
<td>$\leq 40$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>Raw materials used</td>
<td>[$/week]</td>
<td>$\geq 0$</td>
<td>$\leq 100$</td>
</tr>
<tr>
<td>$x_4$</td>
<td>Finishing labor used</td>
<td>[hours/week]</td>
<td>$\geq 0$</td>
<td>$\leq 80$</td>
</tr>
<tr>
<td>$x_5$</td>
<td>Carpentry labor used</td>
<td>[hours/week]</td>
<td>$\geq 0$</td>
<td>$\leq 80$</td>
</tr>
<tr>
<td>$x_6$</td>
<td>Revenue</td>
<td>[$/week]</td>
<td>$\geq 0$</td>
<td>$\leq 80$</td>
</tr>
<tr>
<td>$x_7$</td>
<td>Expense</td>
<td>[$/week]</td>
<td>$\geq 0$</td>
<td>$\leq 80$</td>
</tr>
<tr>
<td>$x_8$</td>
<td>Profit</td>
<td>[$/week]</td>
<td>$\geq 0$</td>
<td>$\leq 80$</td>
</tr>
</tbody>
</table>
Model Formulation

Objective:

\[ \text{max } x_8 \]

Equality Constraints:

\[
\begin{align*}
  x_3 &= 10x_1 + 9x_2 \\
  x_4 &= 2x_1 + x_2 \\
  x_5 &= x_1 + x_2 \\
  x_6 &= 27x_1 + 21x_2 \\
  x_7 &= x_3 + 4x_4 + 6x_5 \\
  x_8 &= x_6 - x_7
\end{align*}
\]
How are decisions constrained?

In addition to non-negativity and model constraints, decisions are subject to:

Demand Constraint:

\[ x_1 \leq 40 \]

Resource Constraints:

\[ x_4 \leq 100 \]
\[ x_5 \leq 80 \]
Demonstrating Excel Solver

- Microsoft includes a sophisticated solver for LP’s in modern releases of Excel, however it is not typically not installed by default. Thus you may need to carry out the following one-time tasks if "Solver" doesn’t appear under the Tools menu:
  1. In Excel, select menu option Tools>Add-Ins…
  2. From the dialog box, select the Solver Add-In

- The "Giapetto’s Workshop" example is available for download from the course web site.
# Excel Solver

## Giapetto's Workshop

CBE - Applied Optimization and Process Operations

<table>
<thead>
<tr>
<th><strong>Objective Function</strong></th>
<th>Value</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profit</td>
<td>180</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Decision Variables</strong></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Soldiers Produced</td>
<td>20</td>
<td>0</td>
<td>40</td>
</tr>
<tr>
<td>Trains Produced</td>
<td>60</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Raw Materials</td>
<td>740</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Finishing Labor</td>
<td>100</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>Carpentry Labor</td>
<td>80</td>
<td>0</td>
<td>80</td>
</tr>
<tr>
<td>Revenue</td>
<td>1800</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expense</td>
<td>1620</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Profit</td>
<td>180</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Equality Constraints</strong></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Material Cost</td>
<td>740</td>
<td>740</td>
<td></td>
</tr>
<tr>
<td>Finishing Labor Hours</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Carpentry Labor Hours</td>
<td>80</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>Revenue</td>
<td>1800</td>
<td>1800</td>
<td></td>
</tr>
<tr>
<td>Expense</td>
<td>1620</td>
<td>1620</td>
<td></td>
</tr>
<tr>
<td>Profit</td>
<td>180</td>
<td>180</td>
<td></td>
</tr>
</tbody>
</table>
Matlab w/Optimization Toolbox

The Matlab Optimization Toolbox provides a solver for linear programs. You provide matrices with the problem data, then call linprog to compute a solution:

```matlab
%% Giapetto’s Workshop (w/Optimization Toolbox)

A = [1 0; 2 1; 1 1];
b = [40;100;80];
f = [-3 2];
lb = [0;0];
ub = [];

x = linprog(f,A,b,[],[],lb,ub);
profit = -f*x;
disp(profit);
```
Algebraic Modeling Languages

What is an "Algebraic Modeling Language"?
Why do I need an Algebraic Modeling Language?
What are some examples of Algebraic Modeling Languages?

- Proven, used in Industry
  - GAMS
  - AMPL (and GMPL, ZIMPL open-source variants)
  - AIMMS
  - Mosel (part of Xpress-IVE package)
  - LINGO, MPL, OPL
  - FortC++

- Experimental, Object Oriented Codes
  - YAMILP, CVX, SOS
Comparison of Three Modeling Languages

The complete Giapetto’s Workshop example can be downloaded from the course web site. Here we show the problem modeled using three different systems:

- Mosel
- AMPL/GMPL
- Matlab with CVX
model "Giapetto’s Workshop" uses "mmxprs";
dclarations
  x1,x2,x8: mpvar       ! Declare Decision Vars
dend-declarations

x1 >= 0
x2 >= 0
demand := x1 <= 40       ! Demand
flabor := 2*x1 + x2 <= 100 ! Finishing Labor
clabor := x1 + x2 <= 80  ! Carpentry Labor
x8 = 3*x1 + 2*x2

minimize(-x8)

writeln(" Objective: ", getobjval)
writeln(" Soldiers: ", getsol(x1))
writeln(" Trains: ", getsol(x2))
writeln(" Profit: ", getsol(x8))
dend-model
An AMPL/GMPL Model

# Giapetto’s Workshop
# Find the optimal solution for maximizing Giapetto’s profit.
# Command Line: glpsol -m Giapetto1.mod -o Giapetto.sol

/* decision variables */
var Soldiers >=0;
var Trains >=0;

/* objective function */
maximize Profit: 3*Soldiers + 2*Trains;

/* Constraints */
s.t. Finishing : 2*Soldiers + Trains <= 100;
s.t. Carpentry : Soldiers + Trains <= 80;
s.t. Demand : Soldiers <= 40;

end;
### AMPL/GMPL Output

**Problem:** Giapetto  
**Rows:** 4  
**Columns:** 2  
**Non-zeros:** 7  
**Status:** OPTIMAL  
**Objective:** Profit = 180 (MAXimum)

<table>
<thead>
<tr>
<th>No.</th>
<th>Row name</th>
<th>St</th>
<th>Activity</th>
<th>Lower bound</th>
<th>Upper bound</th>
<th>Marginal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Profit</td>
<td>B</td>
<td>180</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Finishing</td>
<td>NU</td>
<td>100</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Carpentry</td>
<td>NU</td>
<td>80</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Demand</td>
<td>B</td>
<td>20</td>
<td>40</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No.</th>
<th>Column name</th>
<th>St</th>
<th>Activity</th>
<th>Lower bound</th>
<th>Upper bound</th>
<th>Marginal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Soldiers</td>
<td>B</td>
<td>20</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Trains</td>
<td>B</td>
<td>60</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Matlab/CVX Model - Simple Version

CVX uses Matlab’s objected programming capabilities to provide a ‘language’ for describing convex optimization problems.

%% Giapetto’s Workshop (w/CVX)

cvx_quiet(true);
cvx_begin
    variable x(2);
x(1) >= 0;
x(2) >= 0;
x(1) <= 40;
2*x(1) + x(2) <= 100;
x(1) + x(2) <= 80;
profit = 3*x(1) + 2*x(2);
maximize(profit);
cvx_end

disp(profit);
%% Giapetto’s Workshop (w/CVX)

A = [1 0; 2 1; 1 1];
b = [40;100;80];
f = [3 2];

cvx_quiet(true);
cvx_begin
    variable x(2);
    x >= 0;
    A*x <= b;
    profit = f*x;
    maximize(profit);

CVX_end

disp(profit);
Which would you prefer?

Discuss criteria you might use to decide which of these solvers is appropriate for a particular application:

- Excel with Solver
- Matlab with Optimization Toolbox
- Mosel
- AMPL/GMPL
- Matlab with CVX
Demonstrations

- Unboundedness
- Infeasibility
- Sensitivity Analysis
What did we learn?

Today you how to go about modeling simple linear programming problems, and translating that model for solution using either Excel/Solver, Matlab, or modeling languages. We emphasized the importance of complete and descriptive models.

- Formulation of Simple Problems
- Terminology: Decision Variables, Objective, Active Contraints, Slack Variables, Sensitivity, Slack, Unboundedness, Infeasibility
- Demonstration of Excel & Matlab, and several modeling languages
- Sensitivity Reports
Appendix: Mathematics of Linear Programming

(Background Only – Not for Lecture)
Agenda

Provide a mathematical foundation for discussing Linear Programming problems and solution algorithms.

1. Introduce “Standard Formulations” of Linear Programming problems.
2. Establish Criteria for Optimality
3. Strong and Weak Duality Theorems
Mathematical Elements

Up to this point we have been emphasizing the formulation of simple linear models that involve

- A set of decision variables
- A linear objective
- Possible lower and upper bounds on the decision variables
- Linear equality constraints expressing the relationships among variables.
- Linear inequality constraints expressing resource or other operational bounds.
Linear Programs in “Standard Form”

The phrase "Standard Form" might suggest that there is a single, commonly accepted mathematical description. If only that were true! What follows is a sampling of "Standard Forms".
Example: Giapetto’s Revisited

Maximize

\[ f = 3x_1 + 2x_2 \]

Subject to

\[ x_1 \geq 0 \quad \text{Non-negative production} \]
\[ x_2 \geq 0 \quad \text{Non-negative production} \]
\[ x_1 \leq 40 \quad \text{Maximum Demand for Soldiers} \]
\[ 2x_1 + x_2 \leq 100 \quad \text{Finishing Constraint} \]
\[ x_1 + x_2 \leq 80 \quad \text{Carpentry Constraint} \]
Example: Giapetto’s Revisited

Changing to a minimization, and changing the inequalities so they are consistent –

Minimize

\[ f = \begin{bmatrix} -3 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \]

Subject to

\[ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ -2 & -1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ -40 \\ -100 \\ -80 \end{bmatrix} \]
Standard Form: What we’ll be using ..

This leads to a very simple format we’ll be use to implement a solution algorithm

$$\min_{x} f = c^T x$$

Subject to:

$$A x \geq b$$

We’ll frequently use the equivalent notation

$$a_i^T x \geq b_i; \quad i = 1, 2, \ldots, m$$

where $a_i^T$ are the rows of $A$, i.e.,

$$A = \begin{bmatrix}
    a_1^T \\
    \vdots \\
    a_m^T
\end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix}
    b_1 \\
    \vdots \\
    b_m
\end{bmatrix}$$
## Example: Giapetto’s Second Model

<table>
<thead>
<tr>
<th>Var</th>
<th>Description</th>
<th>Units</th>
<th>LB</th>
<th>UB</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>Production of toy soldiers</td>
<td>[units/week]</td>
<td>$\geq 0$</td>
<td>$\leq 40$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>Production of toy trains</td>
<td>[units/week]</td>
<td>$\geq 0$</td>
<td></td>
</tr>
<tr>
<td>$x_3$</td>
<td>Raw materials used</td>
<td>[$/week]</td>
<td>$\geq 0$</td>
<td></td>
</tr>
<tr>
<td>$x_4$</td>
<td>Finishing labor used</td>
<td>[hours/week]</td>
<td>$\geq 0$</td>
<td>$\leq 100$</td>
</tr>
<tr>
<td>$x_5$</td>
<td>Carpentry labor used</td>
<td>[hours/week]</td>
<td>$\geq 0$</td>
<td>$\leq 80$</td>
</tr>
<tr>
<td>$x_6$</td>
<td>Revenue</td>
<td>[$/week]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_7$</td>
<td>Expense</td>
<td>[$/week]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_8$</td>
<td>Profit</td>
<td>[$/week]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Formulation

Objective:

\[
\text{max } x_8
\]

Equality Constraints:

\[
\begin{align*}
x_3 &= 10x_1 + 9x_2 \\
x_4 &= 2x_1 + x_2 \\
x_5 &= x_1 + x_2 \\
x_6 &= 27x_1 + 21x_2 \\
x_7 &= x_3 + 4x_4 + 6x_5 \\
x_8 &= x_6 - x_7
\end{align*}
\]

Bounds:

\[
\begin{align*}
0 &\leq x_1 \leq 40 \\
0 &\leq x_2 \leq \infty \\
0 &\leq x_3 \leq \infty \\
0 &\leq x_4 \leq 100 \\
0 &\leq x_5 \leq 80 \\
-\infty &\leq x_6 \leq \infty \\
-\infty &\leq x_7 \leq \infty \\
-\infty &\leq x_8 \leq \infty
\end{align*}
\]
Standard Form: Second Version

A given problem can be written in many forms, with varying definition and number of decision variables, constraints, and even objective functions. Is there a "Standard Form" for linear programs?

Edgar, Himmelblau, and Lasdon:

\[
\begin{align*}
\max f &= \sum_{j=1}^{n} c_j x_j \\
\text{Subject to: } &
\sum_{j=1}^{n} a_{ij} x_j = b_i; \ i = 1, 2, \ldots, m \\
&l_j \leq x_j \leq u_j; \ j = 1, 2, \ldots, n
\end{align*}
\]
Matrix form

Minimize:

\[ f = cx \]

Subject to:

\[ Ax = b \]
\[ l \leq x \leq u \]

where \( A \) is an \( m \times n \) matrix, \( b \) is an \( m \times 1 \) vector of constraints, \( c \) is a \( 1 \times n \) vector of objective coefficients, and \( l \) and \( u \) are \( n \times 1 \) vectors of bounds on the decision variables.
Standard Form: Third Version

Similar to the last with equality constraints $Ax = b$, but with only non-negativity constraints on $x$.

$$\min_x f = c^T x$$

Subject to:

$$Ax = b$$

$$x \geq 0$$

Leads to elegant presentations of theory (Fletcher; Boyd; Many others ....)
Standard Form: Fourth Version

No equality constraints, only inequality and non-negativity constraints.

\[
\min_{\mathbf{x}} f = c^T \mathbf{x}
\]

Subject to:

\[
A \mathbf{x} \leq b
\]

\[
\mathbf{x} \geq 0
\]

A common model used, for example, by Vanderbei.
Standard Form: Diwekar Version

No equality constraints, only inequality and non-negativity constraints.

\[
\max_{x} f = c^T x
\]

Subject to:

\[
Ax \leq b
\]

\[
x \geq 0
\]

A common model used, for example, by Diwekar.
Summary of Techniques

1. The sense of an inequality or the optimization can be changed by multiplying through by $-1$.

2. An inequality can be converted to an equality by introducing a slack variable, e.g. $x_j \leq b_j$ can be replaced with $x_j + s = b_j$ with $s \geq 0$.

3. A decision variable that may have both positive and negative values can be represented by
   \[
   x = x^+ - x^-
   \]
   where $x^+, x^- \geq 0$.

4. Equalities can be rewritten as two inequalities.

5. Redundant inequality constraints can be removed by projection onto a reduced space of decision variables.
Converting Among Standard Forms

It’s a straightforward matter to translate problems from one standard form to another, though the translation may change the number of decision variables or constraints.

For example, we’ll demonstrate the conversion

\[
\begin{align*}
\min_x f &= c^T x \\
Ax &= b \\
x &\geq 0
\end{align*}
\Rightarrow
\begin{align*}
\min_{x'} f &= c'^T x' \\
A'x' &\geq b'
\end{align*}
\]
Cont.

Given the equality constraints $Ax = b$, matrix $A$ is partitioned as

$$A = [A_1 | A_2]$$

where $A_1$ is $m \times m$ square and invertible. (Can this always be done? What conditions need to placed on $A$?) The constraints

$$A_1x_1 + A_2x_2 = b$$

are then solved

$$x_1 = A_1^{-1}b - A_1^{-1}A_2x_2$$
Partitioning $c^T$ conformably

$$c^T = \begin{bmatrix} c_1^T \\ c_2^T \end{bmatrix}$$

and plugging back into the objective function and non-negativity constraint

$$\min_x f = \left[ c_2^T - c_1^T A_1^{-1} A_2 \right] x_2 + c_1^T A_1^{-1} b$$

$$\begin{bmatrix} -A_1^{-1} A_2 \\ l_{m-n} \end{bmatrix} x_2 \geq \begin{bmatrix} -A_1^{-1} b \\ 0 \end{bmatrix}$$
To complete the demonstration, define

\[ f' = f - c_1^T A_1^{-1} b \]
\[ c' = c_2^T - c_1^T A_1^{-1} A_2 \]
\[ A' = \begin{bmatrix} -A_1^{-1} A_2 \\ I_{m-n} \end{bmatrix} \]
\[ b' = \begin{bmatrix} -A_1^{-1} b \\ 0 \end{bmatrix} \]

With \( n' = n - m \) decision variables. The equality constraints were used to eliminate \( m \) decision variables from the original problem.

Suggested Exercise: Show how to transform the other standard forms to the format we’re using.
Review of Gradients

Gradients are key to many types of engineering analysis. Consider a function $f(x, y) = xe^{-x^2 - y^2}$.
Review - Gradients

A Contour plot of the function \( f(x, y) = xe^{-x^2-y^2} \)
Lecture 1: Overview

Review - Gradients

Consider a change in $\delta f$:

$$\delta f \approx \frac{\partial f}{\partial x} \delta x + \frac{\partial f}{\partial y} \delta y$$

$$= \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix}$$

Two observations:

1. Gradient is vector.
2. The gradient is orthogonal to the function contours. (Why?)
Review - Gradients

\[ f(x,y) = x e^{-x^2} - y^2 \]
Gradient of Linear Objective and Constraints

For $f(x) = c^T x$:

$$\nabla_x f = \begin{bmatrix} c_1 & c_2 & \ldots & c_n \end{bmatrix} = c^T$$

For $g(x) = a_i^T x$:

$$\nabla_x g = a_i^T$$
Characterizing an Optimum

The constraints define the feasible region of solutions. If the region is empty, then the problem is infeasible.
Characterizing an Optimum

The *Active Constraints* at the optimal vertex define a linear system of equations.

\[ a_i^T = b; \quad i \in \mathcal{A} \]

How to characterize?
Gradients

Consider the gradients of the constraints and objective at the optimal vertex.
Key Insight

A vertex is optimal if the gradient of the objective is a positive sum of the gradients of the active constraints.

Giapetto’s Workshop Example

- Demand
- Finishing
- Carpentry

$x_1$: Soldiers Produced
$x_2$: Trains Produced
Characterizing an Optimum

Active Constraints at the optimal vertex define a linear system of equations.

\[ a_i^T x = b_i \quad i \in \mathcal{A} \]

\( \mathcal{A} \) is the Active Set. How to characterize?
The Active Constraints and Active Set

Suppose constraints 4 & 5 are active (i.e., $A = \triangle, \nabla$)

\[
\begin{bmatrix}
1 & 0 \\
0 & 1 \\
-1 & 0 \\
-2 & -1 \\
-1 & -1
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} \geq
\begin{bmatrix}
0 \\
0 \\
-40 \\
-100 \\
-80
\end{bmatrix}
\]

We can use Slack Variables $s_i \geq 0$ to keep track of which constraints are active. Constraint $i$ is active if $s_i = 0$:

\[a_i^T x - s_i = b_i\]

for example, if 4 & 5 are active, then

$s_1, s_2, s_3 > 0$

$s_4, s_5 = 0$
Consider the gradients of the Active Constraints and the Objective at the optimal vertex.

\[
c^T = [-3 - 2]
\]
\[
a_4^T = [-2 - 1]
\]
\[
a_5^T = [-1 - 1]
\]
The Key Insight

A vertex is optimal if the gradient of the objective is in the cone of the gradients formed by the active constraints. That is, if we find $n$ non-negative $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ so that

$$c^T = \lambda_1 a_1^T + \lambda_2 a_2^T + \lambda_3 a_3^T + \lambda_4 a_4^T + \lambda_5 a_5^T$$
The Key Insight

A vertex is optimal if the gradient of the objective is in the *cone* of the gradients formed by the active constraints. That is, if we find $n$ non-negative $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ so that

$$c^T = \lambda_1 a_1^T + \lambda_2 a_2^T + \lambda_3 a_3^T + \lambda_4 a_4^T + \lambda_5 a_5^T$$

For our example, at the optimum,

$$c^T = [-3 - 2] = [-2 - 1] + [-1 - 1]$$

$$= \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} -2 & -1 \\ -1 & -1 \end{bmatrix}$$

So that

$$c^T = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ -2 & -1 \\ -1 & -1 \end{bmatrix}$$
So to find an optimum, we’re looking for a $\lambda_i \geq 0$ where $\lambda_i \geq 0$ for the active constraints, and

\[
c^T = \sum_{i=1}^{m} a_i^T \lambda_i = \lambda^T A
\]

or, equivalently,

\[
A^T \lambda = c \quad \text{where} \quad \lambda \geq 0
\]
Complementarity Condition:
The relationship between $s_i$ and $\lambda_i$

The final trick is to force $\lambda_i = 0$ for the inactive constraints. Using the slack variables $s_i \geq 0$, $i = 1, \ldots, m$ so that

$$a_i^T x - s_i = b_i$$

Then $s_i = 0$ if constraint $i$ is active\(^3\).

The constraint we need is

$$s_i \lambda_i = 0 \quad i = 1, 2, \ldots, m$$

This is the \textit{Complementarity Condition}

\(^3\)Later we’ll talk about degeneracy
Necessary Conditions for an Optimal Solution

To summarize, necessary conditions for a solution to the Linear Program is the existence of \( x, s, \) and \( \lambda \) satisfying

\[
\begin{align*}
Ax - s &= b \\
A^T \lambda &= c \\
s &\geq 0 \\
\lambda &\geq 0 \\
s^T \lambda &= 0
\end{align*}
\]
What else can we say?

Suppose we can find a solution $x^*, s^*$, and $\lambda^*$ to the necessary conditions. Then consider

$$\lambda^T (Ax^* - s^* = b)$$

$$x^T (A^T \lambda^* = c)$$

If there is a solution to the necessary condition, then the solution satisfies

$$c^T x^* = b^T \lambda^*$$

More generally

$$c^T x \geq b^T \lambda$$

for any feasible $x$, $s$, and $\lambda$ are feasible solutions.
Duality

The duality gap is the difference $b^T \lambda - c^T x$ which is zero only for a solution to our linear programming problem.

<table>
<thead>
<tr>
<th>Primal</th>
<th>Dual</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\min_x f = c^T x$</td>
<td>$\max_x g = b^T \lambda$</td>
</tr>
<tr>
<td>$Ax - s = b$</td>
<td>$A^T \lambda = c$</td>
</tr>
<tr>
<td>$s \geq 0$</td>
<td>$\lambda \geq 0$</td>
</tr>
</tbody>
</table>

This is the key to many approaches linear programming, including Simplex Methods, Interior Point Methods, and generalizable to nonlinear and global optimization methods.
Weak Duality

Theorem

[Weak Duality Theorem] If $x$ is feasible for the Primal problem, and $\lambda$ is feasible for the Dual problem, then

$$c^T x \geq b^T \lambda$$
Strong Duality

Theorem

**[Strong Duality Theorem]** If the Primal problem has a finite solution $x^*$, then the Dual problem has a finite solution $\lambda^*$ such that

$$c^T x^* = b^T \lambda^*$$
What did we learn?

- Formulation of Linear Programs
- Standard form for a linear program
- Techniques to convert among standard forms
- Characterization of the Optimum to a Linear Program
- Dual of a Linear Program
- Nomenclature including bounds, slacks, Lagrange multipliers, dual
Appendix: Active Set Algorithm for Linear Programming

(Background Only – Not for Lecture)
Agenda

1 Development of an Algorithm
   - Simplex ("Active Set") Method
   - Diagnosing Problems
     - Infeasibility
     - Unbounded solutions
     - Degenerate Constraints

2 Another look at the Necessary conditions for Optimality
   - Characterizing Optimality
   - Duality, and Basic Results

3 Barebones Matlab Implementation
How to Solve a Linear Program?

Given a Linear Program:

$$\min_x f = c^T x$$
$$a_i^T x \geq b_i \quad i = 1, 2, \ldots, m$$

How does one compute numerical solutions?
What does it mean to Solve?

Given a Linear Program in $n$ decision variables:

$$\min_{x} f = c^T x$$
$$a_i^T x \geq b_i \quad i = 1, 2, \ldots, m$$

What does it mean to solve a Linear Program?

1. Find an $\hat{x}$ minimizing $f$ and satisfying all of the constraints, or show no such value exists. Possible outcomes
   - Infeasible if no value exists satisfying all constraints.
   - Unbounded if the minimum value is $-\infty$.
   - Possibly unique or non-unique solutions.

2. If a solution unique solution exists, find $n$ constraints (Active Set $A$) which determine the solution
   $$a_i^T \hat{x} = b_i \quad i \in A$$
Active Set

Given \( m \geq n \) constraints, a subset \( A \) of \( n \) constraints defines a potential solution

\[
a_i^T x = b_i \quad i \in A
\]

We’ll call \( A \) the Active Set. We want to find the active set associated with an optimal solution.

\[
A_A = \begin{bmatrix} a_i^T \end{bmatrix} \quad i \in A
\]

\[
b_A = \begin{bmatrix} b_i \end{bmatrix} \quad i \in A
\]
Active Set Example

Suppose constraints 4 & 5 are active (active set \{4, 5\})

\[
\begin{bmatrix}
1 & 0 \\
0 & 1 \\
-1 & 0 \\
-2 & -1 \\
-1 & -1
\end{bmatrix}
\begin{bmatrix}
 x_1 \\
 x_2
\end{bmatrix}
\geq
\begin{bmatrix}
0 \\
0 \\
-40 \\
-100 \\
-80
\end{bmatrix}
\]

\#4 \rightarrow
\[
\begin{bmatrix}
1 & 0 \\
0 & 1 \\
-1 & 0 \\
-2 & -1 \\
-1 & -1
\end{bmatrix}
\begin{bmatrix}
 x_1 \\
 x_2
\end{bmatrix}
\geq
\begin{bmatrix}
0 \\
0 \\
-40 \\
-100 \\
-80
\end{bmatrix}
\]

\#5 \rightarrow

Slack Variables \( s_i \geq 0 \) track which constraints are active.

\[ a_i^T x - s_i = b_i \]

where \( s_1, s_2, s_3 > 0 \) and \( s_4, s_5 = 0 \). The solution we’re after is the solution to

\[
\begin{bmatrix}
-2 & -1 \\
-1 & -1
\end{bmatrix}
\begin{bmatrix}
 x_1 \\
 x_2
\end{bmatrix}
= \begin{bmatrix}
-100 \\
-80
\end{bmatrix}
\]
Brute Force?

For \( m \) constraints and \( n \) decision variables, there are up to

\[
\frac{m!}{n!(m - n)!}
\]

possible vertices. This is a big number.

- For example, for \( m = 50 \) and \( n = 10 \) – a modest problem – up to \( 1.0272 \times 10^{10} \) vertices.
- Double problem size \((m = 100 \text{ and } n = 20)\) then there are \( 5.36 \times 10^{20} \) vertices.

Brute force is not practical!
A Basic Framework

Given a Linear Program:

\[
\begin{align*}
\min_x f &= c^T x \\
\mathbf{a}_i^T x &\geq b_i, \quad i = 1, 2, \ldots, m
\end{align*}
\]

1. Find an initial feasible point \( x^0 \) (equiv., active set) satisfying all constraints. If none, then problem is infeasible and we’re done.

2. While not Optimal:
   - Find an "Improved Solution" \( S(x^k) \rightarrow x^{k+1} \)
   - If Unbounded then done.
   - Watch out for Degeneracy or other numerical instabilities


Challenge is to develop tests for Optimality, Infeasibility, Unboundedness, and a Convergent algorithm.
Active Set Method

Given:
An initial \textit{feasible} active set $A$ yields an initial \textit{feasible} candidate solution for $x_A$ and $f_A$.

While not optimal:
\begin{enumerate}
  \item Perturb the active constraints. Find which one most rapidly reduces the objective function. Remove that one from the active set.
  \item From the currently inactive constraints, find the 'most constraining'. Add this to the active set.
  \item Update problem data.
\end{enumerate}

Need to watch out for numerical instabilities, unbounded solutions, and degenerate constraints.
Initial Feasible Solution

Given an initial active set $\mathcal{A}$, we solve for an initial solution

$$x_\mathcal{A} = A^{-1}_\mathcal{A} b_\mathcal{A}$$

$$f_\mathcal{A} = c^T A^{-1}_\mathcal{A} b_\mathcal{A}$$

Feasibility is verified by checking

$$A x_\mathcal{A} \geq b$$
Example

Consider the toy problem

\[
\min_x f = \begin{bmatrix} -3 & -2 \end{bmatrix} \begin{bmatrix} c^T \\ x \end{bmatrix}
\]

Subject to:

\[
\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ -2 & -1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ -40 \\ -100 \\ -80 \end{bmatrix}
\]
Example

Choosing the active set $\mathcal{A} = \{1, 2\}$ yields

$$
\begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0
\end{bmatrix}
$$

$A_{\mathcal{A}}$ $x_{\mathcal{A}}$ $b_{\mathcal{A}}$
Example (Cont.)

Solution for $x_A$:

$$x_A = A^{-1}_A b_A$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solution for $f_A$:

$$f_A = c^T x_A$$

$$= \begin{bmatrix} -3 & -2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} = 0$$

It’s easy to check that the other 3 constraints are satisfied.
Perturbations

What happens if we make a 'feasible' perturbation to the active constraints?

\[ A_A x'_A = b_A + s_A \]

The perturbation is feasible for the active constraints if \( s_A \geq 0 \).

\[ x'_A = \underbrace{A_A^{-1} b_A + A_A^{-1} s_A}_{x_A} \]

\[ f'_A = \underbrace{c^T A_A^{-1} b_A + c^T A_A^{-1} s_A}_{f_A} \underbrace{\lambda^T_A}_{\lambda} \]

(Notice how the slack variables \( s \) and sensitivity variables \( \lambda \) made a natural entrance to our problem?)
Example (Cont.)

The perturbed solution:

\[
\begin{align*}
\mathbf{x}_A' &= \mathbf{x}_A + A_A^{-1} \mathbf{s}_A \\
&= \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}
\end{align*}
\]

The perturbed objective:

\[
\begin{align*}
\mathbf{f}_A' &= \mathbf{f}_A + c^T A_A^{-1} \mathbf{s}_A \\
&= \begin{bmatrix} 0 \\ f_A \end{bmatrix} + \begin{bmatrix} -3 & -2 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}
\end{align*}
\]
Removing a Constraint from the Active Set

The values in

$$\lambda^T_A = c^T A_A^{-1}$$

represent the sensitivity of the objective function to perturbations in the active constraints. Two possibilities:

1. If $\lambda^T_A \geq 0$ then at an optimum. Problem finished.
2. If any $\lambda^T_A < 0$, then there is an opportunity to further reduce the objective function with a feasible perturbation.

Simple heuristic: Select the constraint corresponding to the most negative value in $\lambda^T_A$ for removal from the Active Set.
Example (Cont.)

In our example,

\[ \lambda^T_A = c^T A^{-1}_A \]
\[ = \begin{bmatrix} -3 & -2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \]
\[ = \begin{bmatrix} -3 & -2 \end{bmatrix} \]

Observations:
- Not optimal since at least one of the \( \lambda \)'s is less than zero.
- The most negative entry is \(-3\) corresponding to the first constraint in the Active Set.
Which constraint to add?

Removing the $p^{th}$ constraint of the active set means that $x'_A$ is shifting in the direction of the $p^{th}$ column of $A^{-1}_A$.

$$x'_A = x_A + A^{-1}_A s_A$$

Call the direction $d_p$ be the $p^{th}$ column of $A^{-1}_A$. Then

$$x'_A = x_A + \alpha \ d_p$$

How far can we move before we run into another constraint?
Finding the **Most Constraining Constraint**

For all of the remaining constraints $a_i^T x' \geq b_i$, so

$$a_i^T (x_A + \alpha \ d_p) \geq b_i$$

$$\alpha \ a_i^T \ d_p \geq b_i - a_i^T x_A$$

There are two cases:

1. If $a_i^T \ d_p \geq 0$ then there is no upper bound on $\alpha$
2. If $a_i^T \ d_p < 0$, then making $\alpha$ large enough will eventually lead to constraint violation.

The *most constraining constraint* is the one for which

$$\alpha = \min_{i \notin A, a_i^T \ d_p < 0} \frac{b_i - a_i^T \ x_A}{a_i^T \ d_p}$$
Example (Cont.)

In our example we decided to remove constraint #1 from the Active Set. For the inactive constraints:

Constraint 3: \( \frac{b_3 - a_3^T x_A}{a_3^T d_p} = \frac{-40 - 0}{-1} = 40 \)

Constraint 4: \( \frac{b_4 - a_4^T x_A}{a_4^T d_p} = \frac{-100 - 0}{-2} = 50 \)

Constraint 5: \( \frac{b_5 - a_5^T x_A}{a_5^T d_p} = \frac{-80 - 0}{-1} = 80 \)

The smallest of these is 40, so Constraint 3 will be added to the Active Set.
Summary: Active Set Algorithm

Given a problem in standard form, and an initial feasible active set $A$.

1. Calculate $A^{-1}_A$, $x_A = A^{-1}_A b_A$, $\lambda^T_A = c^T A^{-1}_A$

2. While any $\lambda^T_A < 0$, repeat

   1. Select the constraint $p$ corresponding to the most negative value in $\lambda^T_A$ for removal from the Active Set.
   2. Set $d_p$ to the $p^{th}$ column of $A^{-1}_A$. Add the most constraining constraint is the one for which

   $$\alpha = \min_{i \not\in A} \frac{b_i - a^T_i x_A}{a^T_i d_p}$$

   to the active set.

3. Update $A$, $A^{-1}_A$, $x_A$, $\lambda^T_A$
Necessary Conditions for an Optimal Solution

Let \( x \) comprise \( n \) decision variables (i.e., \( x \in \mathbb{R}^n \)) subject to \( m \) constraints (i.e., \( A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m \)):

\[
\min_x f = c^T x \\
Ax \geq b
\]

Necessary conditions for a solution are the existence of \( x \in \mathbb{R}^n, s \in \mathbb{R}^m, \) and \( \lambda \in \mathbb{R}^m \) satisfying

\[
Ax - s = b \\
A^T \lambda = c \\
s, \lambda \geq 0 \\
s^T \lambda = 0
\]