Optimization

Why Optimization?

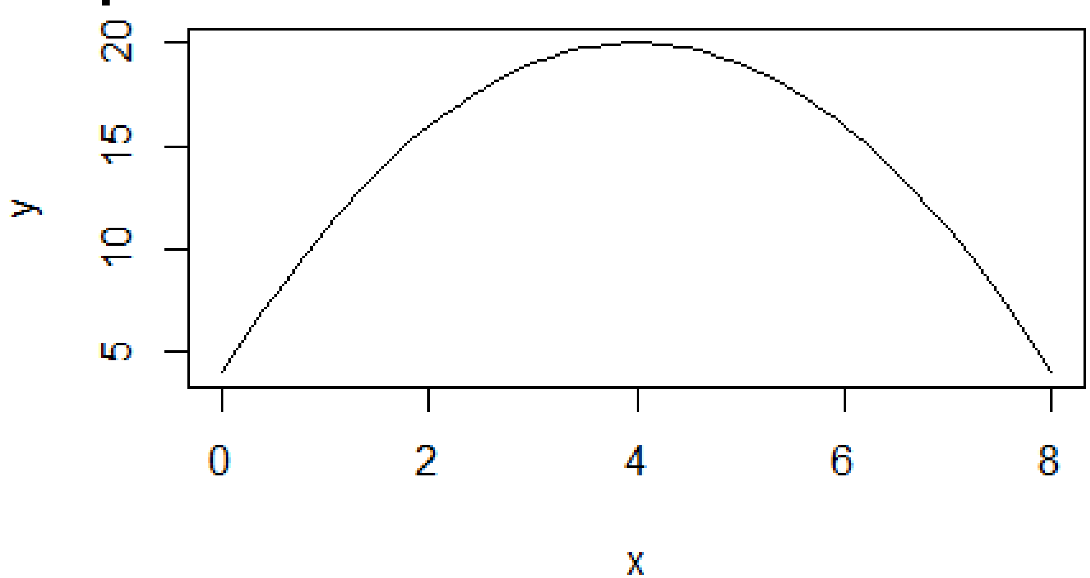
- Optimization is important in many businesses to find the best set of decisions for a particular performance. For example…
- Optimize operational efficiency: capital, personnel, equipment, vehicles, facilities.
- Create measurable return on investment: Optimize costs, earnings and service.
- There are applications of optimization in most industries including: manufacturing, transportation, logistics, financial services, utilities, energy, telecommunications, government, defense and retail.

What is optimization in the mathematical sciences?

- Optimizing a function is finding the maximum (or minimum) a function can take.

What is optimization in the mathematical sciences?

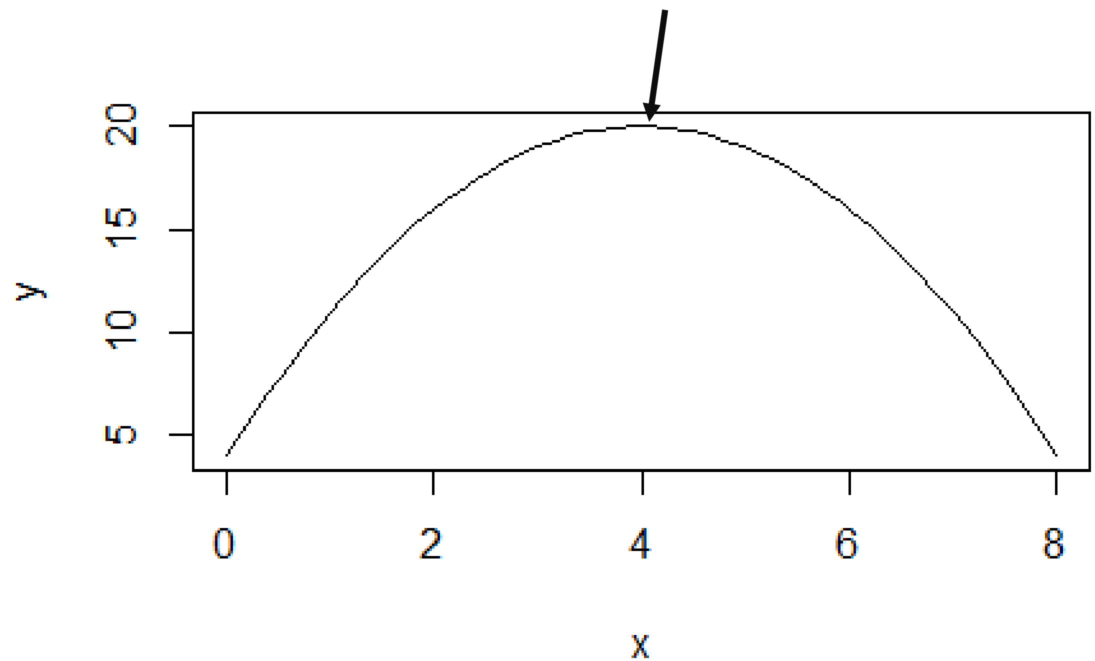
- Optimizing a function is finding the maximum (or minimum) a function can take.
- Goal is find out what values of the "decision variables" (or input variables) optimize this function



What is optimization in the mathematical sciences?

In example below, the x-variable is the "decision variable" and the y is the function we are trying to optimize…

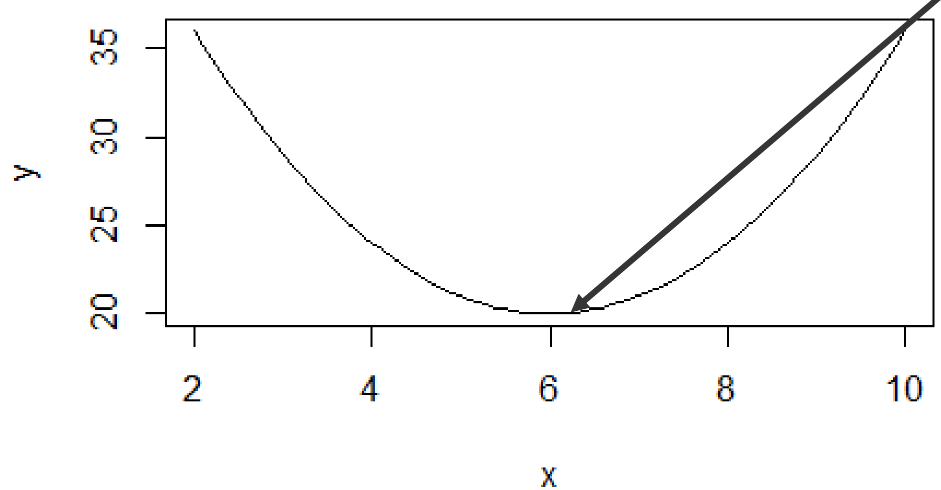
Maximum of function



The maximum occurs when the decision or "input" variable is 4 (the optimal value of the "output" is 20).

Minimize a function

Minimum occurs at "input" of 6 and "output" at this value is 20.



Terminology

- The "input" variables are the variables in which we can change to optimize a function. These variables are referred to as the **decision variables**.
- The **objective function** is the function in which we are trying to optimize (either maximize or minimize). This function is a function of the decision variables. We are trying to find the best values of the decision variables that optimize this function.
- The **constraints** are functions of the decision variables that define the constraints of the problem.
- **Parameters** are values inherent in the problem that we are not able to control

Optimization layout

- **Decision variables**: $x_1, x_2, \ldots x_k$
- **Objective function**:

$$\Sigma a_i x_i = a_1 x_1 + a_2 x_2 + a_3 x_3 + \ldots a_k x_k$$

$a_1, a_2, \ldots a_k$ are the coefficients in the objective function

- **Constraints**:

$$\Sigma b_{1i} x_i \leq c_1$$
$$\Sigma b_{2i} x_i \geq c_2$$
$$\Sigma b_{3i} x_i = c_3$$
$$\ldots$$

b_{ji} are the constraint coefficients; $c_1, c_2 \ldots$ are **parameters** defined in the model

Outputs from an optimization

- No optimization exists
- More than one solution exists
- There exists one unique solution to the problem

4 main types of optimization problems

- Linear programming – objective function and constraints are linear (LP)
- Integer linear programming – objective function and constraints are linear, but decision variables MUST be integers (ILP)
- Mixed integer linear programming – same as ILP with only some decision variables restricted to integers
- Non-linear programming – objective function and constraints are continuous, but not all are linear

Linear Programming

- The **feasible region** is the region defined by the constraints (need to find the optimal solution to the objective function over this region)
- One of the easiest solutions for solving a linear programming problem is the Simplex method
  - Strategy is to move along the edges of the feasible region until the optimal value of the objective function is found

Example

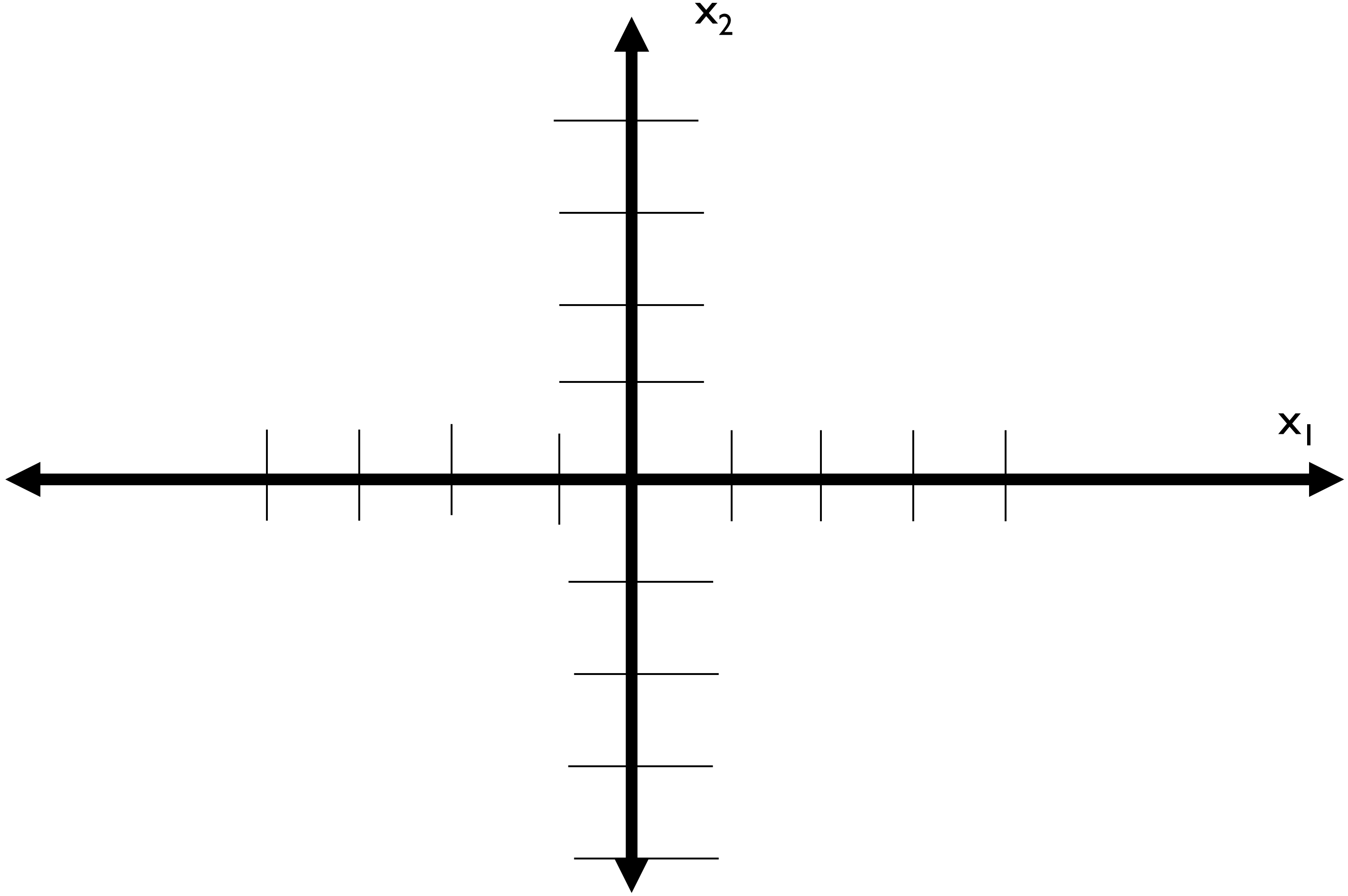
- Find maximum of $2x_1 + 3x_2$

- Constraints:

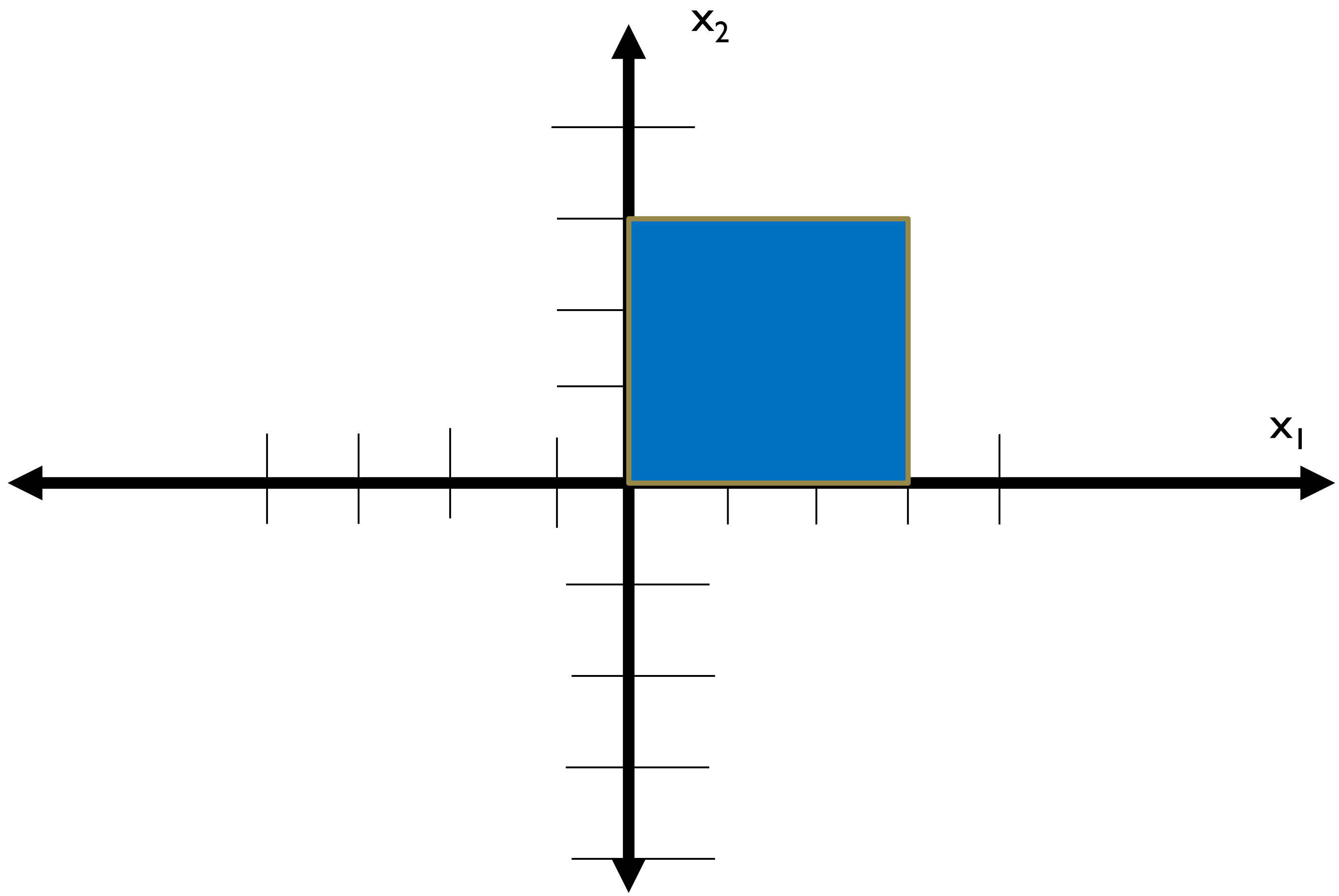
  $0 \leq x_1 \leq 3$

  $0 \leq x_2 \leq 3$

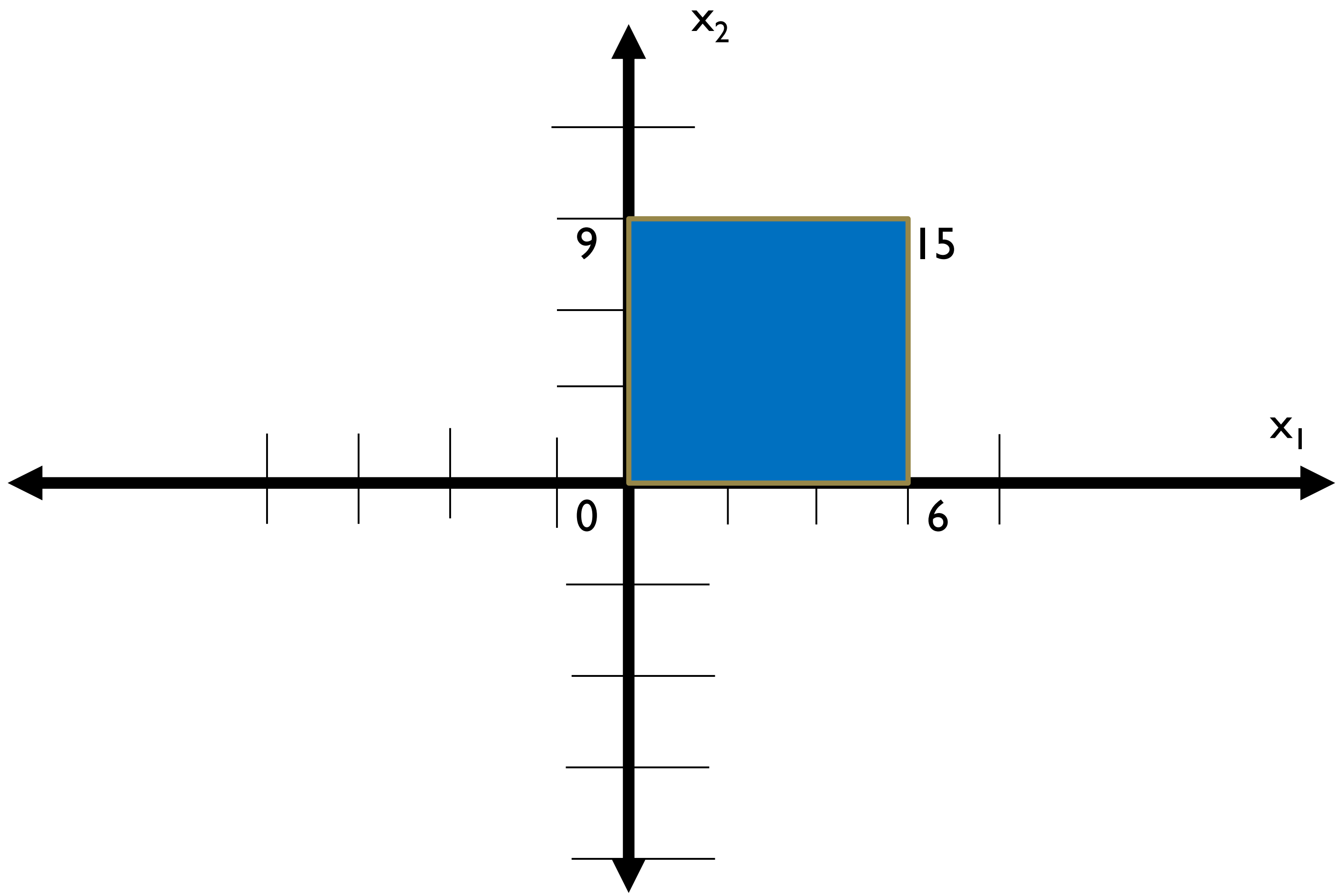
Feasible Region?



Feasible Region?



Feasible Region?



Example of Simplex algorithm

MAXIMIZE: $15C + 24D$

CONSTRAINTS:

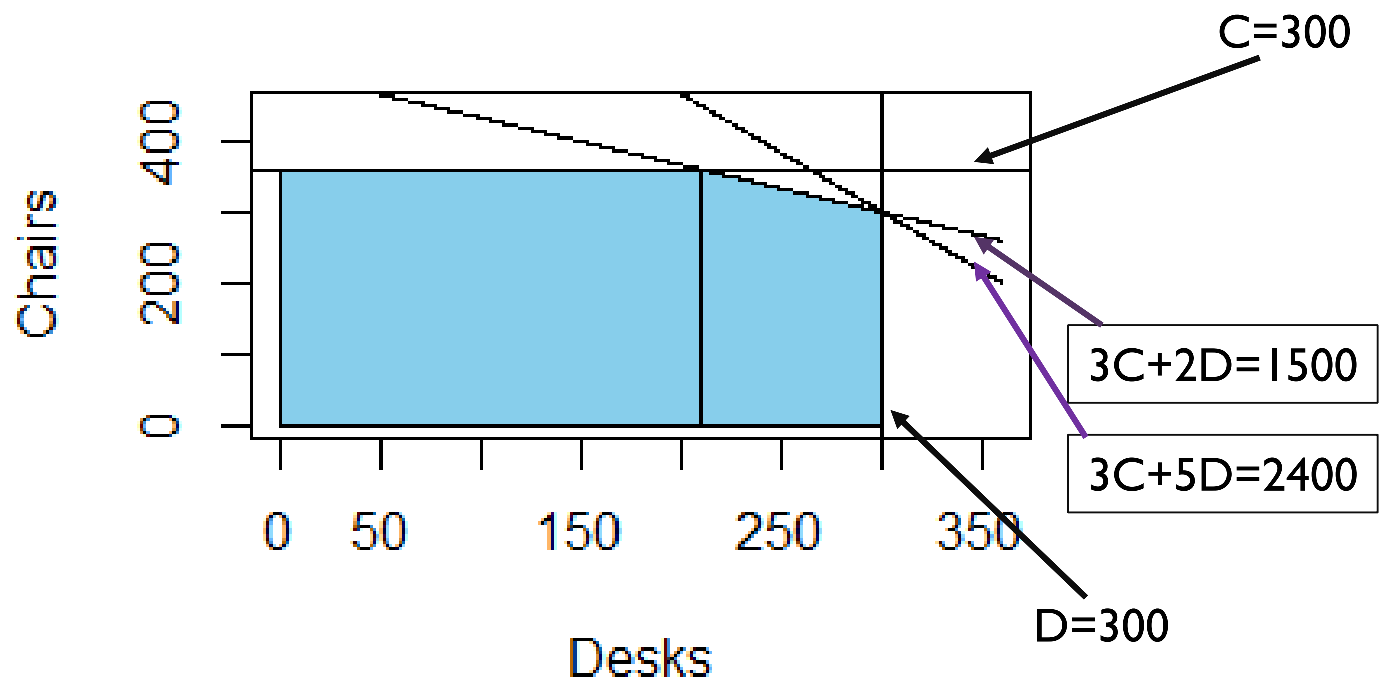
$3C + 5D \leq 2400$

$3C + 2D \leq 1500$

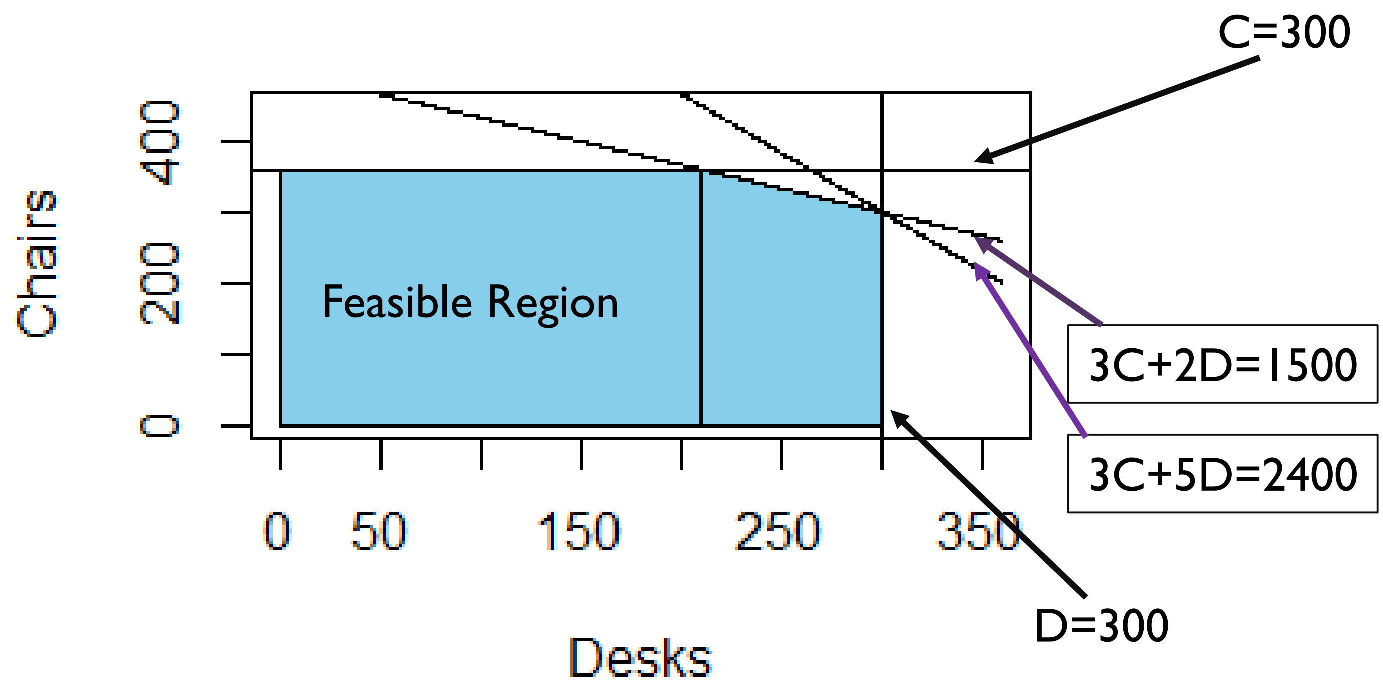
$C \leq 360$

$D \leq 300$

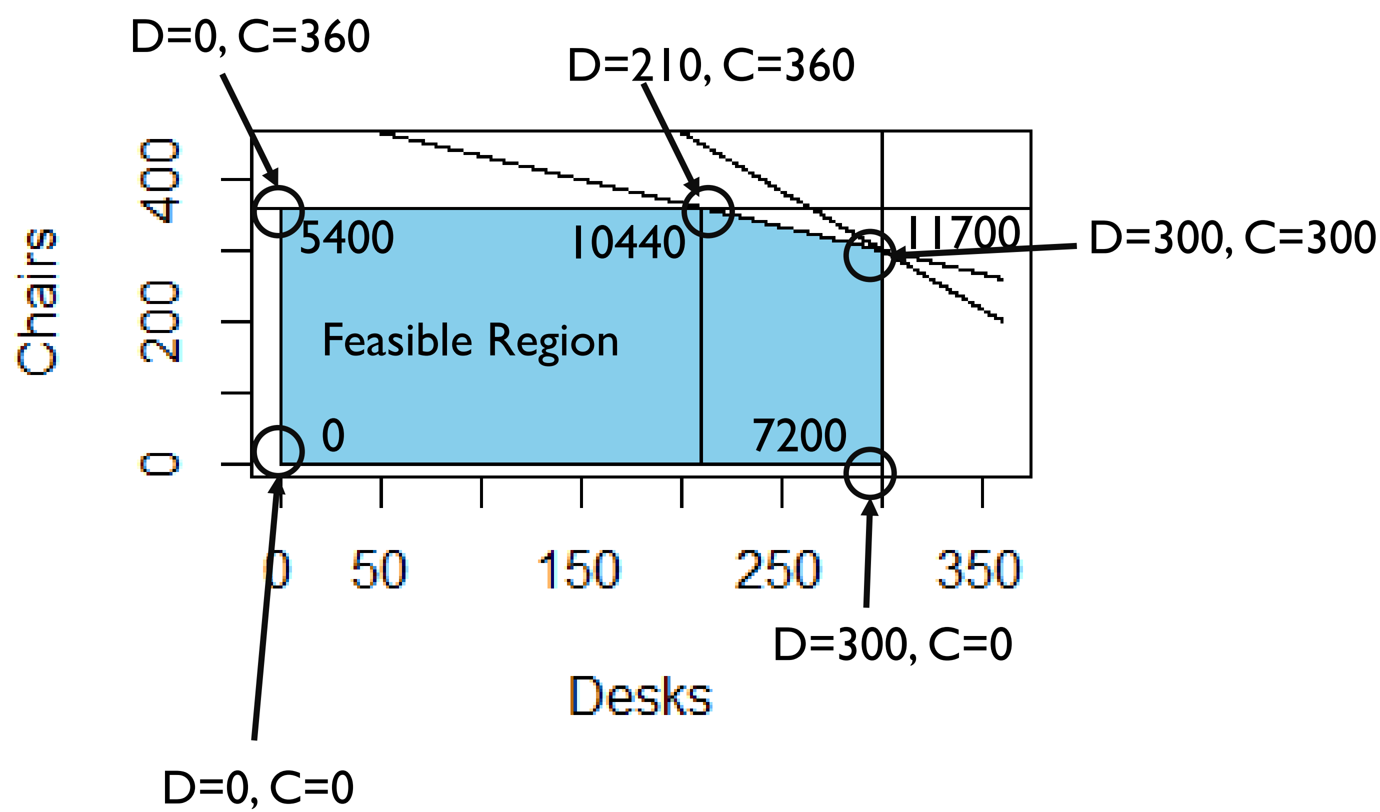
Feasible region



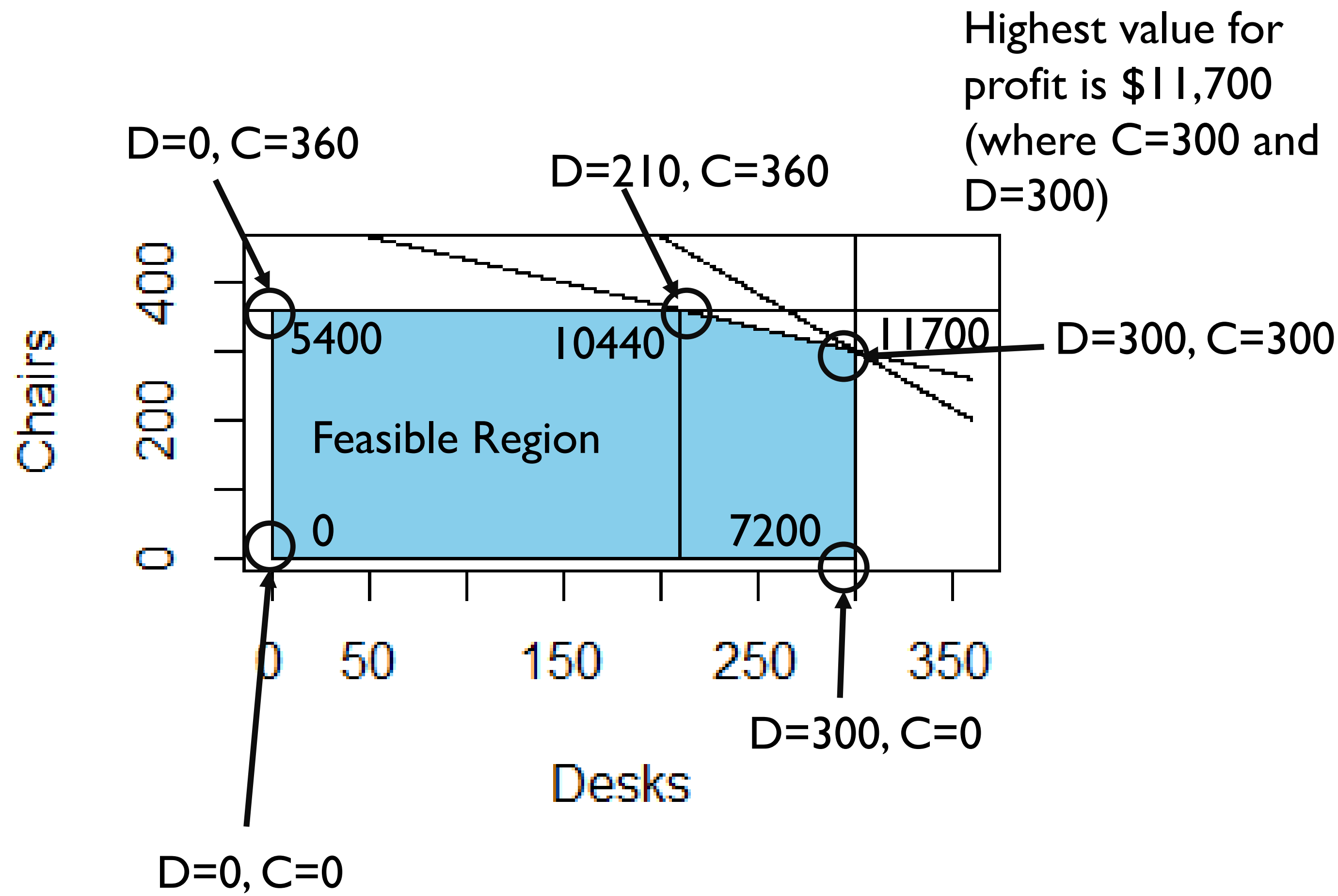
Feasible region



Simplex method



Simplex method



4 types of linear programming

- Allocation model (goal is to maximize objective function; constraints are stated in terms of less than)
- Covering model (goal is to minimize objective function; constraints are stated in terms of greater than)
- Blending model
- Network model (will come later)

Example

- Veerman Furniture Company makes chairs, desks and tables with the following information

	Hours per unit			
Department	Chairs	Desks	Tables	Hours Avail
Fabrication	4	6	2	1850
Assembly	3	5	7	2400
Shipping	3	2	4	1500
Demand potential	360	300	100	
Profit	$15	$24	$18	

Want to maximize profit

Example:

- Objective function:
  - MAXIMIZE: $15C + 24D + 18T$
- Constraints:
  - Fabri: $4C + 6D + 2T \leq 1850$
  - Assem: $3C + 5D + 7T \leq 2400$
  - Shipp: $3C + 2D + 4T \leq 1500$
  - $C \leq 360$
  - $D \leq 300$
  - $T \leq 100$
  - (Implicit: $C \geq 0$, $D \geq 0$, $T \geq 0$)

SAS Code

```
proc optmodel;
/* declare variables */
var Chairs>=0, Desks>=0, Tables>=0;
/* maximize objective function (profit) */
max Profit = 15*Chairs + 24*Desks + 18*Tables;
/* subject to constraints */
con Assembly: 3*Chairs + 5*Desks +7*Tables<=2400;
con Shipping: 3*Chairs + 2*Desks + 4*Tables<=1500;
con Fabrication: 4*Chairs+6*Desks+2*Tables<=1850;
con DemandC: Chairs <=360;
con DemandD: Desks<=300;
con DemandT: Tables<=100;
solve with lp;
/* display solution */
print Chairs Desks Tables;
quit;
```

Output

The SAS System

The OPTMODEL Procedure

Problem Summary	
Objective Sense	Maximization
Objective Function	Profit
Objective Type	Linear
Number of Variables	3
Bounded Above	0
Bounded Below	3
Bounded Below and Above	0
Free	0
Fixed	0
Number of Constraints	6
Linear LE (<=)	6

Output cont

Linear EQ (=)	0
Linear GE (>=)	0
Linear Range	0
Constraint Coefficients	12

Performance Information	
Execution Mode	Single-Machine
Number of Threads	1

The OPTMODEL Procedure

Solution Summary	
Solver	LP
Algorithm	Dual Simplex
Objective Function	Profit
Solution Status	Optimal
Objective Value	8400
Primal Infeasibility	0
Dual Infeasibility	0
Bound Infeasibility	0
Iterations	2
Presolve Time	0.00
Solution Time	0.00

Chairs	Desks	Tables
0	275	100

In R

```
###Need to install package lpSolve
library(lpSolve)
###Coefficients for the objective function
f.obj=c(15,24,18)
### Constraints equations added as a matrix
f.con=matrix(c(4,6,2,3,5,7,3,2,4,1,0,0,0,1,0,0,0,1),nrow=6,byrow=T)
##### Need to let R know direction of the constraints (nrows in R should equal # of inequalities/equalities)
f.dir=c("<=","<=","<=","<=","<=","<=")
###Parameter values for constraints
f.rhs = c(1850,2400,1500,360,300,100)
### The call to optimize the function
lp.model=lp ("max", f.obj, f.con, f.dir, f.rhs,compute.sens = 1)
#####Get status of convergence
lp.model$status
f.names=c('Chairs','Desks','Tables')  ##In case you want to make pretty
names(lp.model$solution)=f.names  ## making it pretty
lp.model$solution  ## Getting the solution
lp.model$objval  ### Getting the value of the objective function
```

R output

```
> lp.model$status
[1] 0
> lp.model$solution
Chairs   Desks Tables
     0     275     100
> lp.model$objval
[1] 8400
```

Gurobi in R

```
library(gurobi)
library(prioritizr)
###Need to put everything in a list (similar to R code above)
model <- list()
 model$obj        <- c(15,24,18)  ##objective function coefficients
 model$modelsense <- "max“  ###is it a max or min problem
 model$rhs        <- c(1850,2400,1500,360,300,100)  ###parameters
 model$sense       <- c("<=","<=","<=","<=","<=","<=") #### constraints inequalities
 model$vtype       <- "C“  ##### C means continuous
##### matrix of constraint coefficients
 model$A          <- matrix(c(4,6,2,3,5,7,3,2,4,1,0,0,0,1,0,0,0,1),nrow=6,byrow=T)
##### Now run it
 result <- gurobi(model, list())
 print(result$status)
 f.names=c('Chairs','Desks','Tables')
 names(result$x)=f.names
 print(result$x)
 print(result$objval)
```

Output from Gurobi (R)

```
print(result$status)
[1] "OPTIMAL"
> f.names=c('Chairs','Desks','Tables')
> names(result$x)=f.names
> print(result$x)
Chairs Desks Tables 0 275 100
> print(result$objval) [1] 8400
```