# rgurobi: Solve optimisation problems using Gurobi

*Jeffrey O. Hanson*[1]

[1]*School of Biological Sciences, The University of Queensland, Brisbane, QLD, Australia*

*Correspondance should be addressed to jeffrey.hanson@uqconnect.edu.au*

*01 March 2016*

## Contents

# Introduction

# Case-study

Here, we will use a case-study to illustrate the use of the `rgurobi R` package. Specifically, we will tackle the uncapacitated facility location problem (Cornuéjols *et al.* 1990). This problem involves a product. This product is made at facilities. Products are transported to stores and sold to consumers. Each store has a different level of demand. The cost of transporting the product from a facility to a store is associated with a cost. Building a facility in a location to minimise transportation costs is also associated with a cost. The goal of this problem is to find out where facilities should be built to minimise the combined building and transportation costs.

We can formulate this as an optimisation problem. Let $I$ denote the set of stores where the product is sold (indexed by $i$). Let $J$ denote the set of locations were facilities could be built (indexed by $j$). Let $f_j$ denote the cost of building a facility in location $j$. Let the cost of transporting the product from location $j$ to store $i$ be $c_{ij}$. Each store is assigned to receive products from a single constructed facility ($Y_{ij}$). The decision variables in the problem are the location $X_j$ variables and the assignment variables $Y_{ij}$:

$$X_j = \begin{cases} 1, & \text{if a facility is constructed at location } j \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij} = \begin{cases} 1, & \text{if store } i \text{ receives products from location } j \\ 0, & \text{otherwise} \end{cases}$$

Specficially, the problem is formulated as an integer problem (IP):

$$\begin{aligned}
\text{(UFLP)} \qquad & \text{Min} \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} d_{ij} Y_{ij} + \sum_{i=0}^{I-1} f_i X_i && && \text{(1a)} \\
& \text{s.t.} \sum_{i=1}^{I-1} Y_{ij} = 1 && \forall 0 \leq i \leq I-1 && \text{(1b)} \\
& Y_{ij} \leq X_i && \forall 0 \leq i \leq I-1, \; 0 \leq j \leq J-1 && \text{(1c)} \\
& X_i \in 0, 1 && \forall 0 \leq i \leq I-1 && \text{(1d)}
\end{aligned}$$

The objective function is the combined cost of building facilities in selected locations and the transportation costs from these facilities to the stores (1a). Constraints (1b) ensure that each store recevies products from a facility. Constraints (1c) ensure that stores are only assigned to receive products from locations where facilities are to be built. Contraints (1d) ensure that the decison variables are binary.

### Data and dependencies

We will use data collected by Daskin (1995). This dataset is stored in the `city1990` object and contains data for 88 cities obtained from the 1990 Population and Housing Census. Each row corresponds to a different city. The cities present in the dataset are the 50 most populous cities in the lower 48 states in addition to the state capitals. The 'first.demand' column contains the 1990 population of each city. The 'second.demand' column contains the number of households in the city in 1990. The 'fixed.cost' is the 1990 median home values in the city.

We will use this data construct a hypothetical facility location problem. In this exercise, we have a product that needs to be manufactured and transported to each of the 88 cities. We want to identify which cities we should build the facilities to maximise profits. In other words, we want to identify which cities we should build cities in that will minimise the transportation costs and the initial costs of building facilities in those locations. We will use the population in each city ('first.demand' column) to represent the demand for our product. We will also use the median home values in each city ('fixed.cost' column) to represent the cost of building a facility in the city. The cost of transporting the product from one city to another is \$0.00001 per mile per unit of demand. Note that in this particular instance of the problem, the demand points are in the same location as the locations where we could be facilities.

```
# load data
data(city1990)

# show first 20 rows of data
head(city1990)
```

```
##           city state  longitude latitude fixed.cost first.demand
## 1     New York    NY  -73.94548 40.67054     189600      7322564
## 2  Los Angeles    CA -118.41120 34.11210     244500      3485398
## 3      Chicago    IL  -87.68497 41.83705      78700      2783726
## 4      Houston    TX  -95.38673 29.76870      58000      1630553
## 5 Philadelphia    PA  -75.13468 40.00682      49400      1585577
## 6    San Diego    CA -117.13577 32.81495     189400      1110549
##   second.demand
## 1       2819401
## 2       1217405
## 3       1025174
## 4        616877
## 5        603075
## 6        406096
```

Let's visualise the problem.

```
# load packages
library(dplyr)
library(tidyr)
library(fields)
library(sp)
```

```
library(rworldxtra)
library(ggplot2)
```

```
# load map data
data(countriesHigh)
countries.FPLY <- countriesHigh[countriesHigh$ADMIN ==
    'United States of America',]

# make plot
ggplot() +
    geom_polygon(data=countries.FPLY, aes(x=long, y=lat, group=group),
        fill='grey85', color='grey70') +
    geom_point(data=city1990, aes(x=longitude, y=latitude, color=first.demand),
        size=2, alpha=0.6) +
    theme_classic() +
    theme(axis.ticks=element_blank(), axis.text=element_blank(),
        plot.margin=unit(c(0,0,0,0),'cm'), axis.line=element_blank(),
        strip.background = element_rect(fill='grey20'),
        strip.text = element_text(color='white'),
        axis.title=element_blank(), legend.title=element_blank()) +
    coord_cartesian(
        xlim=range(city1990$longitude),
        ylim=range(city1990$latitude)) +
    scale_color_distiller(name='Demand', palette='YlGnBu')
```

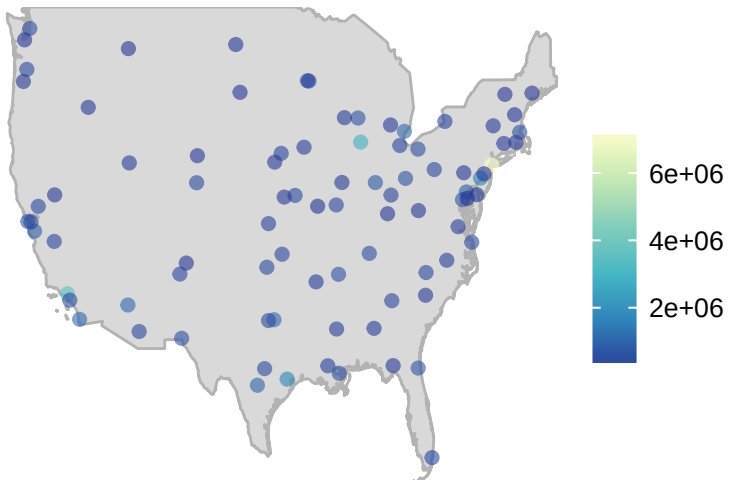


**Figure 1** Points denote the location of cities. The color of each point indicates the relative demand for the product in the city.

### Prepare data for Gurobi

To solve the problem, we need to reformat the data into a `list` containing the problem formulation.

```
## intialization
n <- nrow(city1990)
model <- list(modelsense='min')

## preliminary processing
# create distance matrix
cities.dists <- rdist.earth(select(city1990, longitude, latitude))
transport.costs <- cities.dists * 0.0025 *
    matrix(rep(city1990$first.demand, each=n), byrow=TRUE, ncol=n, nrow=n)

## main processing
# create variable dictionary
tmp <- expand.grid(seq_len(n), seq_len(n))
var.dict <- c(paste0('X_',seq_len(n)),
    paste0('Y_',tmp[[1]], '_', tmp[[2]]))
var.dict <- structure(seq_along(var.dict), .Names=var.dict)

# store model objective function (eqn 1a)
model$obj <- c(city1990$fixed.cost)
for (i in seq_len(n))
    for (j in seq_len(n))
        model$obj <- c(model$obj, transport.costs[i,j])

# preallocate model constraints
counter <- 0
model$A_rows <- c()
model$A_cols <- c()
model$A_vals <- c()
model$rhs <- c()
model$sense <- c()

# constraint (eqn 1b)
for (i in seq_len(n)) {
    counter <- counter + 1
    model$A_rows <- c(model$A_rows, rep(counter, n))
    model$A_cols <- c(model$A_cols,
        unname(var.dict[paste0('Y_',i,'_',seq_len(n))]))
    model$A_vals <- c(model$A_vals, rep(1, n))
}
model$rhs <- c(model$rhs, rep(1, n))
model$sense <- c(model$sense, rep('=', n))

# constraint (eqn 1d)
for (i in seq_len(n)) {
    model$A_rows <- c(model$A_rows, counter+seq_len(n))
    model$A_cols <- c(model$A_cols, rep(i, n))
    model$A_vals <- c(model$A_vals, rep(1, n))
```

```
    model$A_rows <- c(model$A_rows, counter+seq_len(n))
    model$A_cols <- c(model$A_cols,
        unname(var.dict[paste0('Y_',seq_len(n),'_',i)]))
    model$A_vals <- c(model$A_vals, rep(-1, n))

    model$rhs <- c(model$rhs, rep(0, n))
    model$sense <- c(model$sense, rep('>', n))

    counter <- counter + n
}

# construct model matrix
model$A <- sparseMatrix(i=model$A_rows, j=model$A_cols, x=model$A_vals)

# constraint (eqn 1e)
model$vtype <- rep('B', length(var.dict))
model$ub <- rep(1, length(var.dict))
model$lb <- rep(0, length(var.dict))
```

### Solving the problem

Now we can solve the problem.

```
# solve the problem
result <- gurobi(model, params=list(Presolve=0), NumberSolutions=2)

# print total cost of best solution
print(result$obj[1])
```

```
## [1] 1728234
```

```
# extract selected facility locations
solution.locations <- result$x[,seq_len(n),drop=FALSE]

# print number of facilities in best solution
print(sum(solution.locations[1,]))
```

```
## [1] 4
```

### Visualise solutions

Let's take a look at the best solution.

```
# prepare data
city1990$Solution <- c('Not selected','Selected')[solution.locations[1,]+1]
city1990$Frequency <- colMeans(solution.locations)
```

```
connections <- data.frame(
    connection=grep('^Y\\_.*$', names(var.dict), value=TRUE),
    status=result$x[1,grep('^Y\\_.*$', names(var.dict), value=FALSE)])
connections <- connections %>%
    filter(status==1) %>%
    mutate(
        i=as.numeric(sapply(strsplit(as.character(connection), '\\_'), `[[`, 2)),
        j=as.numeric(sapply(strsplit(as.character(connection), '\\_'), `[[`, 3)),
        i.longitude=city1990$longitude[i],
        i.latitude=city1990$latitude[i],
        j.longitude=city1990$longitude[j],
        j.latitude=city1990$latitude[j]
    )
connections <- cbind(
        gather(connections, point, longitude, i.longitude, j.longitude),
        select(gather(connections, point, latitude, i.latitude, j.latitude),
            latitude)
    ) %>%
    select(connection,i,j,longitude,latitude) %>%
    arrange(connection)

# visualise best solution
ggplot() +
    geom_polygon(data=countries.FPLY, aes(x=long, y=lat, group=group),
        fill='grey85', color='grey70') +
    geom_line(data=connections, aes(x=longitude, y=latitude, group=connection),
        color='grey10') +
    geom_point(data=city1990, aes(x=longitude, y=latitude,
        color=Solution)) +
    theme_classic() +
    theme(axis.ticks=element_blank(), axis.text=element_blank(),
        plot.margin=unit(c(0,0,0,0),'cm'), axis.line=element_blank(),
        strip.background = element_rect(fill='grey20'),
        strip.text = element_text(color='white'),
        axis.title=element_blank()) +
    coord_cartesian(
        xlim=range(city1990$longitude),
        ylim=range(city1990$latitude))
```

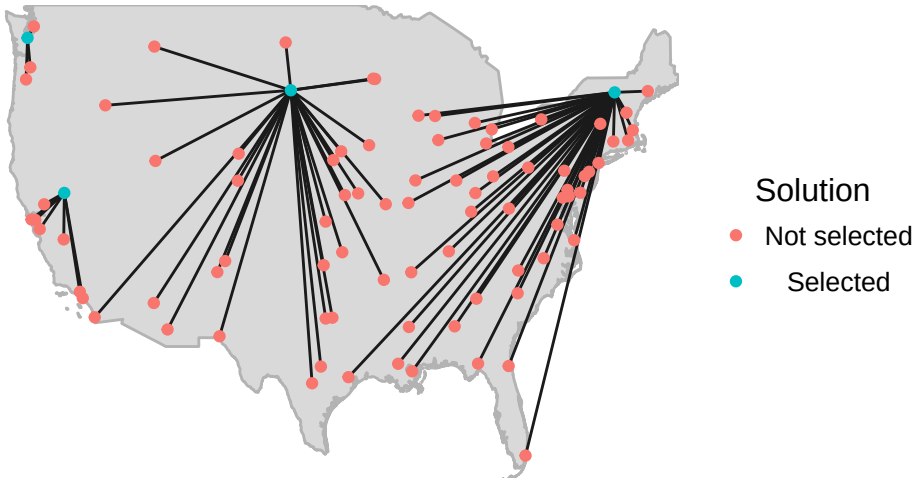


**Figure 2** Points denote cities. Cities that have been selected to host a facility are denoted in blue. Lines indicate transportation routes.

Let's also plot the selection frequency of facilities in the solution pool.

```
# make plot
ggplot() +
    geom_polygon(data=countries.FPLY, aes(x=long, y=lat, group=group),
        fill='grey85', color='grey70') +
    geom_point(data=city1990, aes(x=longitude, y=latitude, size=Frequency)) +
    theme_classic() +
    theme(axis.ticks=element_blank(), axis.text=element_blank(),
        plot.margin=unit(c(0,0,0,0),'cm'), axis.line=element_blank(),
        strip.background = element_rect(fill='grey20'),
        strip.text = element_text(color='white'), axis.title=element_blank()) +
    coord_cartesian(
        xlim=range(city1990$longitude),
        ylim=range(city1990$latitude))
```

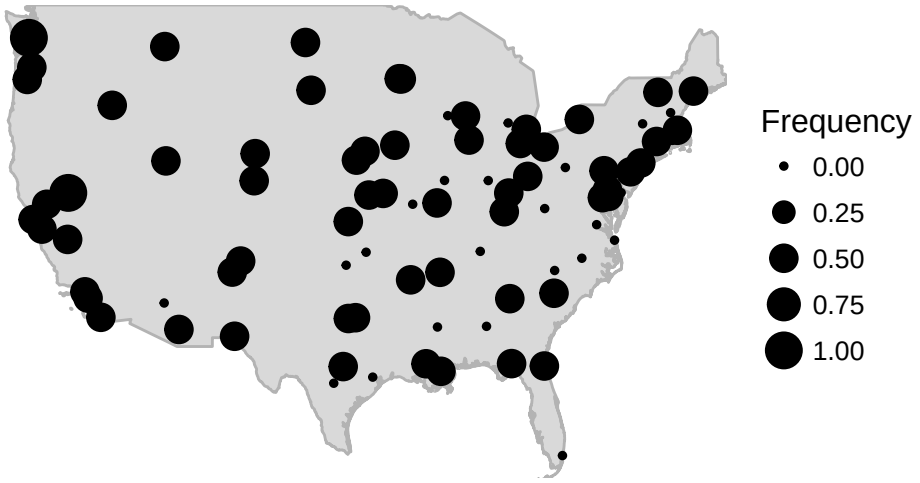


**Figure 3** Points indicate cities. The size of each point indicates the number of times that the corresponding city has been selected as a place to build a facility.

## References

Cornuéjols, G., Nemhauser, G. L., Wolsey, L. A. (1990) The uncapacitated facility location problem. In: *Discrete Location Theory* (eds P. B. Mirchandani & R. L. Francis) pp. 119–171 Wiley, New York.

Daskin, M. S. (1995) Appendix G: Longitudes, latitudes, demands, and fixed costs for CITY1990.GRT: An 88-node problem defined on the Continental United States. In: *Network and Discrete Location* pp. 476–479 Wiley, New York.