Exploring the Weber Problem Numerically

Philip A. Viton

January 17, 2014

Introduction

- The appendix to this handout shows how to create a simple R function to plot our results, and then how to incorporate it into our `weber` function.
- We’ll use this to visualize factory location in the 2D Weber problem.
- Since we’ll be plotting the result, we’ll present just the data matrix summarizing our problem, and the Solution block of the results print-out, rather than the complete problem setting.

Base Setup

```
> dat
   scoord tcoord theta kcost
[1,]  2   0  1.0  1
[2,]  0   2  1.5  1
[3,]  3   3  2.0  1
```

Initial Optimal Location

```
> dat
   scoord tcoord theta kcost
[1,]  2   0  1.0  1
[2,]  0   2  1.5  1
[3,]  3   3  2.0  1
```

Solution (see plot window):
Factory located at: 2.2065 2.3697
Total trans cost: 7.7613

Both inputs are weight-losing, so locate away from the market. $Z_2$ looses more, so locate closer to $Z_1$. 
**Equal Input Proportions**

What if $\theta_1 = \theta_2 = 2$?

<table>
<thead>
<tr>
<th>scoord</th>
<th>tcoord</th>
<th>theta</th>
<th>kcost</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>[2,]</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>[3,]</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

Solution (see plot window):
Factory located at: 1.3999 2.0458
Total trans cost: 8.6593

Inputs are equally weight-losing, so locate away from the market. The optimal location would be equidistant between the inputs, but there is a correction because the market is off-center.

**Equal But Smaller Input Proportions**

What if $\theta_1 = \theta_2 = 1$?

<table>
<thead>
<tr>
<th>scoord</th>
<th>tcoord</th>
<th>theta</th>
<th>kcost</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>[2,]</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>[3,]</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

Solution (see plot window):
Factory located at: 1.5772 1.5775
Total trans cost: 5.2779

Inputs are weight-losing, but less than before, so locate closer to the market.

**Non-Weight-Losing Inputs**

Take $\theta_1 + \theta_2 = 1$ ; $\theta_1 = 0.75$

<table>
<thead>
<tr>
<th>scoord</th>
<th>tcoord</th>
<th>theta</th>
<th>kcost</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>2</td>
<td>0</td>
<td>1.00</td>
</tr>
<tr>
<td>[2,]</td>
<td>0</td>
<td>2</td>
<td>0.75</td>
</tr>
<tr>
<td>[3,]</td>
<td>3</td>
<td>3</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Solution (see plot window):
Factory located at: 2.0000 0.0000
Total trans cost: 2.9119

We need to ship 1T of final product, and less of each input: locate the factory at the market. Same result for the reverse case, when $\theta_2 = 0.75$.

**Locate at $Z_1$**

Take $\theta_1 = 1.5 , \theta_2 = 0.5$

<table>
<thead>
<tr>
<th>scoord</th>
<th>tcoord</th>
<th>theta</th>
<th>kcost</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>2</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>[2,]</td>
<td>0</td>
<td>2</td>
<td>1.5</td>
</tr>
<tr>
<td>[3,]</td>
<td>3</td>
<td>3</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Solution (see plot window):
Factory located at: 0.0000 2.0000
Total trans cost: 4.4096

Here we need to ship 1.5 T of $Z_1$ versus 0.5 T of $Z_2$ per ton of final product. Locate the factory at $Z_1$. 
Adding a Plot

This appendix shows how to create and use a function that plots our Weber results. We will want to plot both the Weber Triangle and optimal factory location.

R has extremely good plotting capabilities, and it is worth getting to know them. Many of the online intros contains discussions of plotting, and there is also a book in the UseR! series on the plotting system called Lattice Graphics.

However, we'll opt for simplicity rather than sophistication. One limitation is that our function will not plot a convex hull: that is, it will be correct only for the 2-input Weber problem. You might want to explore how you could extend the function to handle the general case. (Hint: your first step should be to search R’s help system for, say “convex hull”).

Adding a Plot (I)

Our plot function will take two arguments: our standard data matrix (which will be supplied from the call to `weber`), and an optional second argument, which, if present will be a results structure from an `optim` call.

The reason for doing it this way is that if we omit the second argument, we can plot just the basic Weber Triangle for our problem.

To accomplish this, we provide a default value (here, `NULL`) for the second (results structure) argument, in our function header. If the second argument is omitted when we call our plot function, then it will be assumed to have this default value.

Adding a Plot (II)

To plot the Weber Triangle we use a line plot, but we need to connect the first and last points, which is not done by default. To achieve this, we extract the locations into a temporary matrix `m` and then append a copy of row 1 to `m` using `rbind` (ie “row-bind”).

For the Weber Triangle we want to see both the points and the connecting lines. To achieve this we we use `type="o"` : this overlays the points and the lines.

If the second argument (the optimization structure) is not `NULL` (which we check via `if (!is.null(res))`) we use the `points` function to add a point (the factory location, the solution of our optimization) to the existing plot (ie the Weber Triangle). Note that we need to add the `transpose (t())` of the solution.
Adding a Plot (III)

We also make two changes in the `weber` function from the last handout:

- We make a global copy of `optim`’s solution structure, in case we want to examine it separately: `weber_res <<- z` right after `z <<- optim`. The symbol `<<-` is R’s global assignment operator.
- Note that this will be overlaid each time we run the `weber` function: if you want to keep it for longer than that, you should assign `weber_res` to another variable after running `optim`.
- Then we call our plotting function: `wplot(data,z)`, where `z` is the result of `optim`. Of course, we could also have called it as `wplot(data,weber_res)` given that we saved the solution.

Philip A. Viton

CRP 4110 — Exploring Weber

January 17, 2014 13 / 14

Adding a Plot (IV)

Here’s our plotting function. As you can see, it’s very simple.

```r
wplot<-function(data,res=NULL){
m<-data[,c("scoord","tcoord")]
m<-rbind(m,m[1,])
plot(m,type="o")
if (!is.null(res)){points(t(res$par),pch=19)}
}
```

Here, `pch=19` selects a filled circle as the plot symbol.

Philip A. Viton

CRP 4110 — Exploring Weber

January 17, 2014 14 / 14