ESCUELA UNIVERSITARIA POLITÉCNICA
DE LA ALMUNIA DE DOÑA GODINA (ZARAGOZA)

ANEXOS

Optimización de una red de aprovisionamiento

Optimization of a supply network

425.16.108

Autor: **Alejandro Sarabia Hernando**
Director: **Luis Mariano Esteban Escaño**
Fecha: **29 de junio de 2016**

ÍNDICE DE CONTENIDO

INDICE DE ILUSTRACIONES

Autor: **Alejandro Sarabia Hernando**
Nº TFG: **425.16.108**

INDICE DE TABLAS

1. CÓDIGOS COMPLETOS

1.1. SERVER.R

```
library(shiny)
library(TSP)
library(rJava)
library(xlsx)
library(lpSolve)
shinyServer(function(input, output) {
Distancias<matrix(rbind(c(0,168,354,368,503,511,639,486,507,621,235,200,
370,142,604,360,281,595,469,305,851,597,473,549,759,253,502,442,144,760,705,
499,611,872,473,693,666,862,351,501,449,419,221,247,187,450,598,513,392),
c(168,0,291,534,688,524,787,652,673,640,248,385,555,308,617,351,447,692,561,
407,1017,762,486,654,925,419,472,626,82,926,871,665,662,1038,639,748,832,
1028,517,595,551,432,319,413,166,616,744,679,484),
c(354,291,0,658,610,796,942,789,664,436,520,392,353,496,889,168,601,497,834,
225,1141,886,758,868,1049,552,201,590,218,1049,995,789,934,1162,763,996,984,
1151,640,401,769,704,591,499,438,739,901,803,756),
c(368,534,658,0,328,731,416,263,238,615,531,250,504,276,809,526,168,580,493,
438,519,264,571,390,427,108,636,308,511,427,373,187,467,540,109,470,382,529,
66,498,253,655,412,131,465,120,374,179,424),
c(503,688,610,328,0,1026,691,538,93,328,726,311,265,547,1104,464,464,292,788,
376,655,498,866,665,660,404,421,66,670,567,607,462,743,523,292,745,657,591,
390,210,632,949,706,367,661,414,650,358,718),
c(511,524,796,731,1026,0,610,606,920,1116,280,695,865,541,103,888,564,1090,
272,799,1088,784,163,477,995,624,997,961,589,1057,890,691,485,1170,845,571,
708,1104,663,996,467,100,428,692,351,728,567,825,313),
c(639,787,942,416,691,610,0,158,601,978,611,598,788,552,687,811,399,943,372,
723,543,336,450,136,484,402,920,671,783,609,282,243,155,640,397,101,100,560,
352,861,222,533,473,468,611,280,62,377,302),
c(486,652,789,263,538,606,158,0,449,826,516,445,636,399,685,659,247,790,369,
570,487,183,447,134,395,249,768,519,631,456,296,90,211,569,244,213,181,504,
```

199,708,142,531,378,315,589,127,118,225,300),
c(507,673,664,238,93,920,601,449,0,382,694,272,265,440,997,518,357,347,681,
376,662,408,759,575,570,297,475,75,651,526,516,372,652,573,201,655,567,628,
300,265,525,842,600,260,629,324,559,268,611),
c(621,640,436,615,328,1116,978,826,382,0,839,448,260,684,1208,291,708,210,
1032,330,1041,786,1110,953,948,648,234,309,566,905,895,750,1031,862,580,1033
,945,929,678,122,876,1023,782,611,774,702,938,646,962),
c(235,248,520,531,726,280,611,516,694,839,0,417,587,263,372,610,383,812,386,
521,1014,759,241,479,921,419,719,658,311,922,868,555,487,1035,636,573,710,
1024,513,718,377,187,144,433,73,612,569,675,309),
c(200,385,392,250,311,695,598,445,272,448,417,0,192,267,786,259,255,417,579,
170,796,541,657,523,703,206,341,243,361,704,650,444,585,817,352,651,639,806,
295,323,423,601,361,118,352,394,555,458,510),
c(370,555,353,504,265,865,788,636,265,260,587,192,0,431,955,202,446,233,769,
120,986,731,847,713,894,397,158,245,476,841,840,634,775,797,515,841,829,865,
485,140,613,770,530,344,521,584,745,582,700),
c(142,308,496,276,547,541,552,399,440,684,263,267,431,0,633,455,135,657,367,
367,760,505,441,369,668,165,564,480,286,668,614,400,432,781,382,517,578,770,
259,564,270,448,147,179,199,358,510,422,289),
c(604,617,889,809,1104,103,687,685,997,1208,372,786,955,633,0,980,641,1182,
340,891,1165,862,231,555,1073,702,1089,1038,681,1135,967,768,563,1247,922,
638,786,1182,741,1088,545,192,520,770,443,805,645,903,390),
c(360,351,168,526,464,888,811,659,518,291,610,259,202,455,980,0,468,344,792,
92,1009,754,870,736,916,419,124,438,277,917,863,656,798,1030,631,864,852,
1019,508,248,636,793,553,366,498,607,768,670,722),
c(281,447,601,168,464,564,399,247,357,708,383,255,446,135,641,468,0,672,327,
382,651,396,405,270,559,62,580,398,425,559,505,248,333,672,273,418,426,661,
150,590,171,488,245,129,333,249,358,313,257),
c(595,692,497,580,292,1090,943,790,347,210,812,417,233,657,1182,344,672,0,997
,348,943,751,1075,918,914,613,303,274,621,855,860,715,996,812,545,998,911,879
,643,94,841,998,758,576,749,667,903,611,928),
c(469,561,834,493,788,272,372,369,681,1032,386,579,769,367,340,792,327,997,0,
706,849,546,112,239,756,385,903,722,625,818,651,452,165,931,606,251,469,866,
424,914,228,212,249,453,387,489,329,587,74),
c(305,407,225,438,376,799,723,570,376,330,521,170,120,367,891,92,382,348,706,

Autor: **Alejandro Sarabia Hernando**
Nº TFG: **425.16.108**

```
0,922,667,783,649,830,333,202,350,335,830,776,570,712,943,544,777,765,932,421
,247,550,706,466,280,457,520,682,584,636),
c(851,1017,1141,519,655,1088,543,487,662,1041,1014,796,986,760,1165,1009,651
,943,849,922,0,315,928,614,97,591,1119,732,994,173,286,440,692,133,460,639,
454,74,527,922,619,1011,895,658,948,439,599,395,781),
c(597,762,886,264,498,784,336,183,408,786,759,541,731,505,862,754,396,751,546
,667,315,0,619,305,223,337,865,479,741,296,125,131,383,409,206,385,264,332,
274,668,309,702,545,404,695,136,289,142,472),
c(473,486,758,571,866,163,450,447,759,1110,241,657,847,441,231,870,405,1075,
112,783,928,619,0,316,834,463,981,800,550,896,729,530,324,1009,684,410,547,
943,502,992,306,100,323,531,313,567,406,664,152),
c(549,654,868,390,665,477,136,134,575,953,479,523,713,369,555,736,270,918,239
,649,614,305,316,0,521,329,847,646,692,583,416,217,85,696,371,167,234,630,326,
835,101,401,341,397,479,254,93,351,171),
c(759,925,1049,427,660,995,484,395,570,948,921,703,894,668,1073,916,559,914,
756,830,97,223,834,521,0,498,1026,640,902,94,227,348,599,195,367,602,395,115,
435,829,526,919,802,565,856,346,506,303,688),
c(253,419,552,108,404,624,402,249,297,648,419,206,397,165,702,419,62,613,385,
333,591,337,463,329,498,0,529,342,400,501,446,240,389,613,215,452,424,603,92,
534,227,545,302,72,357,191,356,254,314),
c(502,472,201,636,421,997,920,768,475,234,719,341,158,564,1089,124,580,303,
903,202,1119,865,981,847,1026,529,0,395,400,991,974,768,910,948,666,975,964,
1015,620,206,745,904,664,478,620,719,880,733,834),
c(442,626,590,308,66,961,671,519,75,309,658,243,245,480,1038,438,398,274,722,
350,732,479,800,646,640,342,395,0,604,597,587,443,723,584,272,726,638,699,412
,192,566,883,640,301,595,395,630,339,652),
c(144,82,218,511,670,589,783,631,651,566,311,361,476,286,681,277,425,621,625,
335,994,741,550,692,902,400,400,604,0,904,850,644,725,1017,618,838,810,1006,
495,523,593,495,382,391,228,594,742,658,546),
c(760,926,1049,427,567,1057,609,456,526,905,922,704,841,668,1135,917,559,855,
818,830,173,296,896,583,94,501,991,597,904,0,335,343,661,117,324,664,542,106,
435,786,550,981,803,566,856,346,568,269,750),
c(705,871,995,373,607,890,282,296,516,895,868,650,840,614,967,863,505,860,651
,776,286,125,729,416,227,446,974,587,850,335,0,250,436,384,324,664,542,106,
435,786,550,814,665,514,804,255,343,251,583),
```

```
c(499,665,789,187,462,691,243,90,372,750,555,444,634,400,768,656,248,715,452,
570,440,131,530,217,348,240,768,443,644,343,250,0,296,458,168,299,201,447,165
,632,204,616,419,326,591,51,203,149,385),
c(611,662,934,467,743,485,155,211,652,1031,487,585,775,432,563,798,333,996,
165,712,692,383,324,85,599,389,910,723,725,661,436,296,0,774,449,83,253,709,
405,914,178,408,348,459,486,332,99,439,177),
c(872,1038,1162,540,523,1170,640,569,573,862,1035,817,797,781,1247,1030,672,
812,931,943,133,409,1009,696,195,613,948,584,1017,117,384,458,774,0,438,736,
550,70,549,745,664,1095,917,680,970,461,682,374,865),
c(473,639,763,109,292,845,397,244,201,580,636,352,515,382,922,631,273,545,606
,544,460,206,684,371,367,215,666,272,618,324,324,168,449,438,0,451,363,426,
172,462,323,768,518,237,571,120,356,66,538),
c(693,748,996,470,745,571,101,213,655,1033,573,651,841,517,638,864,418,998,
251,777,639,385,410,167,602,452,975,726,838,664,664,299,83,736,451,0,196,656,
407,916,263,493,493,523,571,335,101,443,263),
c(666,832,984,382,657,708,100,181,567,945,710,639,829,578,786,852,426,911,469
,765,454,264,547,234,395,424,964,638,810,542,542,201,253,550,363,196,0,470,
360,828,321,632,572,522,710,247,161,344,401),
c(862,1028,1151,529,591,1104,560,504,628,929,1024,806,865,770,1182,1019,661,
879,866,932,74,332,943,630,115,603,1015,699,1006,106,106,447,709,70,426,656,
470,0,536,806,636,1029,904,667,957,448,616,361,798),
c(351,517,640,66,390,663,352,199,300,678,513,295,485,259,741,508,150,643,424,
421,527,274,502,326,435,92,620,412,495,435,435,165,405,549,172,407,360,536,0,
559,192,589,395,158,448,116,315,190,358),
c(501,595,401,498,210,996,861,708,265,122,718,323,140,564,1088,248,590,94,914
,247,922,668,992,835,829,534,206,192,523,786,786,632,914,745,462,916,828,806,
559,0,759,903,663,477,654,585,820,529,845),
c(449,551,769,253,632,467,222,142,525,876,377,423,613,270,545,636,171,841,228
,550,619,309,306,101,526,227,745,566,593,550,550,204,178,664,323,263,321,636,
192,759,0,390,239,299,377,208,187,305,159),
c(419,432,704,655,949,100,533,531,842,1023,187,601,770,448,192,793,488,998,
212,706,1011,702,100,401,919,545,904,883,495,981,814,616,408,1095,768,493,
632,1029,589,903,390,0,336,618,259,651,491,749,236),
c(221,319,591,412,706,428,473,378,600,782,144,361,530,147,520,553,245,758,249
,466,895,545,323,341,802,302,664,640,382,803,665,419,348,917,518,433,572,904,
```

```
395,663,239,336,0,326,144,422,431,558,171),
c(247,413,499,131,367,692,468,315,260,611,433,118,344,179,770,366,129,576,453
,280,658,404,531,397,565,72,478,301,391,566,514,326,459,680,237,523,522,667,
158,477,299,618,326,0,371,258,428,321,384),
c(187,166,438,465,661,351,611,589,629,774,73,352,521,199,443,498,333,749,387,
457,948,695,313,479,856,357,620,595,228,856,804,591,486,970,571,571,710,957,
448,654,377,259,144,371,0,548,569,612,309),
c(450,616,739,120,414,728,280,127,324,702,612,394,584,358,805,607,249,667,489
,520,439,136,567,254,346,191,719,395,594,346,255,51,332,461,120,335,247,448,
116,585,208,651,422,258,548,0,238,101,420),
c(598,744,901,374,650,567,62,118,559,938,569,555,745,510,645,768,358,903,329,
682,599,289,406,93,506,356,880,630,742,568,343,203,99,682,356,101,161,616,315
,820,187,491,431,428,569,238,0,337,261),
c(513,679,803,179,358,825,377,225,268,646,675,458,582,422,903,670,313,611,587
,584,395,142,664,351,303,254,733,339,658,269,251,149,430,374,66,443,344,361,
190,529,305,749,558,321,612,101,337,0,519),
c(392,484,756,424,718,313,302,300,611,962,309,510,700,289,390,722,257,928,74,
636,781,472,152,171,688,314,834,652,546,750,583,385,177,865,
      538,263,401,798,  358,845,159,236,171,384,309,420,261,519,0)),ncol=49)
      attributes(Distancias)$dimnames[[1]]<c("ALBACETE","ALICANTE","ALMERIA",
"AVILA","BADAJOZ","BARCELONA","BILBAO","BURGOS","CACERES","CADIZ",
"CASTELLON","CIUDAD.REAL","CORDOBA","CUENCA”,"GERONA”,"GRANADA",
"GUADALAJARA","HUELVA","HUESA","JAEN","LA.CORUNA","LEON","LERIDA",
"LOGRONO","LUGO",”MADRID","MALAGA","MERIDA","MURCIA","ORENSE","OVIEDO",
"PALENCIA",PAMPLONA","PONTEVEDRA","SALAMANCA","SAN.SEBASTIAN",
"SANTANDER","SANTIAGO.DE.COMPOSTELA","SEGOVIA","SEVILLA”,”SORIA”
"TARRAGONA","TERUEL","TOLEDO","VALENCIA","VALLADOLID",”VITORIA","ZAMORA",
"ZARAGOZA")
      attributes(Distancias)$dimnames[[2]]<c("ALBACETE","ALICANTE","ALMERIA",
"AVILA","BADAJOZ","BARCELONA","BILBAO","BURGOS","CACERES","CADIZ",
"CASTELLON","CIUDAD.REAL","CORDOBA","CUENCA”,"GERONA”,"GRANADA",
"GUADALAJARA","HUELVA","HUESA","JAEN","LA.CORUNA","LEON","LERIDA",
"LOGRONO","LUGO",”MADRID","MALAGA","MERIDA","MURCIA","ORENSE","OVIEDO",
"PALENCIA",PAMPLONA","PONTEVEDRA","SALAMANCA","SAN.SEBASTIAN",
"SANTANDER","SANTIAGO.DE.COMPOSTELA","SEGOVIA","SEVILLA”,”SORIA”
```

```
"TARRAGONA","TERUEL","TOLEDO","VALENCIA","VALLADOLID",”VITORIA","ZAMORA",
"ZARAGOZA")
        dataset=reactive({
         infile=input$file1
        if(is.null(infile))
        return(NULL)
        read.xlsx(infile$datapath,1,header=FALSE,colClasses=rep("numeric",10))})
        RutaCompra<-reactive({
        Matrizkm<-matrix(rep(Distancias[,input$Origen],10),nrow=49)
        A<-dataset()
        FuncionCerof<-function(x,y){if(is.na(A[x,y])) 10000 else A[x,y]}
        B<-as.data.frame(matrix(mapply(FuncionCerof,rep(seq(1,49),rep(10,49)),
rep (seq(1,10),49)),byrow=T,ncol=10))
        MM<-as.matrix(input$costekm*Matrizkm+B)
        objective.in<-as.numeric(c(MM[1,1],MM[2,1],MM[3,1],MM[4,1],MM[5,1],
MM[6,1],MM[7,1],MM[8,1],MM[9,1],MM[10,1],MM[11,1],MM[12,1],MM[13,1],
MM[14,1],MM[15,1],MM[16,1],MM[17,1],MM[18,1],MM[19,1],MM[20,1],MM[21,1],
MM[22,1],MM[23,1],MM[24,1],MM[25,1],MM[26,1],MM[27,1],MM[28,1],MM[29,1],
MM[30,1],MM[31,1],MM[32,1],MM[33,1],MM[34,1],MM[35,1],MM[36,1],MM[37,1],
MM[38,1],MM[39,1],MM[40,1],MM[41,1],MM[42,1],MM[43,1],MM[44,1],MM[45,1],
MM[46,1],MM[47,1],MM[48,1],MM[49,1],MM[1,2],MM[2,2],MM[3,2],MM[4,2],MM[5,2],
MM[6,2],MM[7,2],MM[8,2],MM[9,2],MM[10,2],MM[11,2],MM[12,2],MM[13,2],
MM[14,2],MM[15,2],MM[16,2],MM[17,2],MM[18,2],MM[19,2],MM[20,2],MM[21,2],
MM[22,2],MM[23,2],MM[24,2],MM[25,2],MM[26,2],MM[27,2],MM[28,2],MM[29,2],
MM[30,2],MM[31,2],MM[32,2],MM[33,2],MM[34,2],MM[35,2],MM[36,2],MM[37,2],
MM[38,2],MM[39,2],MM[40,2],MM[41,2],MM[42,2],MM[43,2],MM[44,2],MM[45,2],
MM[46,2],MM[47,2],MM[48,2],MM[49,2],MM[1,3],MM[2,3],MM[3,3],MM[4,3],MM[5,3],
MM[6,3],MM[7,3],MM[8,3],MM[9,3],MM[10,3],MM[11,3],MM[12,3],MM[13,3],
MM[14,3],MM[15,3],MM[16,3],MM[17,3],MM[18,3],MM[19,3],MM[20,3],MM[21,3],
MM[22,3],MM[23,3],MM[24,3],MM[25,3],MM[26,3],MM[27,3],MM[28,3],MM[29,3],
MM[30,3],MM[31,3],MM[32,3],MM[33,3],MM[34,3],MM[35,3],MM[36,3],MM[37,3],
MM[38,3],MM[39,3],MM[40,3],MM[41,3],MM[42,3],MM[43,3],MM[44,3],MM[45,3],
MM[46,3],MM[47,3],MM[48,3],MM[49,3],MM[1,4],MM[2,4],MM[3,4],MM[4,4],
MM[5,4],MM[6,4],MM[7,4],MM[8,4],MM[9,4],MM[10,4],MM[11,4],MM[12,4],MM[13,4],
MM[14,4],MM[15,4],MM[16,4],MM[17,4],MM[18,4],MM[19,4],MM[20,4],MM[21,4],
```


Autor: **Alejandro Sarabia Hernando**
Nº TFG: **425.16.108**

MM[22,4],MM[23,4],MM[24,4],MM[25,4],MM[26,4],MM[27,4],MM[28,4],MM[29,4],
MM[30,4],MM[31,4],MM[32,4],MM[33,4],MM[34,4],MM[35,4],MM[36,4],MM[37,4],
MM[38,4],MM[39,4],MM[40,4],MM[41,4],MM[42,4],MM[43,4],MM[44,4],MM[45,4],
MM[46,4],MM[47,4],MM[48,4],MM[49,4],MM[1,5],MM[2,5],MM[3,5],MM[4,5],
MM[5,5],MM[6,5],MM[7,5],MM[8,5],MM[9,5],MM[10,5],MM[11,5],MM[12,5],
MM[13,5],MM[14,5],MM[15,5],MM[16,5],MM[17,5],MM[18,5],MM[19,5],MM[20,5],
MM[21,5],MM[22,5],MM[23,5],MM[24,5],MM[25,5],MM[26,5],MM[27,5],MM[28,5],
MM[29,5],MM[30,5],MM[31,5],MM[32,5],MM[33,5],MM[34,5],MM[35,5],MM[36,5],
MM[37,5],MM[38,5],MM[39,5],MM[40,5],MM[41,5],MM[42,5],MM[43,5],MM[44,5],
MM[45,5],MM[46,5],MM[47,5],MM[48,5],MM[49,5],MM[1,6],MM[2,6],MM[3,6],
MM[4,6],MM[5,6],MM[6,6],MM[7,6],MM[8,6],MM[9,6],MM[10,6],MM[11,6],MM[12,6],
MM[13,6],MM[14,6],MM[15,6],MM[16,6],MM[17,6],MM[18,6],MM[19,6],MM[20,6],
MM[21,6],MM[22,6],MM[23,6],MM[24,6],MM[25,6],MM[26,6],MM[27,6],MM[28,6],
MM[29,6],MM[30,6],MM[31,6],MM[32,6],MM[33,6],MM[34,6],MM[35,6],MM[36,6],
MM[37,6],MM[38,6],MM[39,6],MM[40,6],MM[41,6],MM[42,6],MM[43,6],MM[44,6],
MM[45,6],MM[46,6],MM[47,6],MM[48,6],MM[49,6],MM[1,7],MM[2,7],MM[3,7],
MM[4,7],MM[5,7],MM[6,7],MM[7,7],MM[8,7],MM[9,7],MM[10,7],MM[11,7],MM[12,7],
MM[13,7],MM[14,7],MM[15,7],MM[16,7],MM[17,7],MM[18,7],MM[19,7],MM[20,7],
MM[21,7],MM[22,7],MM[23,7],MM[24,7],MM[25,7],MM[26,7],MM[27,7],MM[28,7],
MM[29,7],MM[30,7],MM[31,7],MM[32,7],MM[33,7],MM[34,7],MM[35,7],MM[36,7],
MM[37,7],MM[38,7],MM[39,7],MM[40,7],MM[41,7],MM[42,7],MM[43,7],MM[44,7],
MM[45,7],MM[46,7],MM[47,7],MM[48,7],MM[49,7],MM[1,8],MM[2,8],MM[3,8],
MM[4,8],MM[5,8],MM[6,8],MM[7,8],MM[8,8],MM[9,8],MM[10,8],MM[11,8],MM[12,8],
MM[13,8],MM[14,8],MM[15,8],MM[16,8],MM[17,8],MM[18,8],MM[19,8],MM[20,8],
MM[21,8],MM[22,8],MM[23,8],MM[24,8],MM[25,8],MM[26,8],MM[27,8],MM[28,8],
MM[29,8],MM[30,8],MM[31,8],MM[32,8],MM[33,8],MM[34,8],MM[35,8],MM[36,8],
MM[37,8],MM[38,8],MM[39,8],MM[40,8],MM[41,8],MM[42,8],MM[43,8],MM[44,8],
MM[45,8],MM[46,8],MM[47,8],MM[48,8],MM[49,8],MM[1,9],MM[2,9],MM[3,9],
MM[4,9],MM[5,9],MM[6,9],MM[7,9],MM[8,9],MM[9,9],MM[10,9],MM[11,9],MM[12,9],
MM[13,9],MM[14,9],MM[15,9],MM[16,9],MM[17,9],MM[18,9],MM[19,9],MM[20,9],
MM[21,9],MM[22,9],MM[23,9],MM[24,9],MM[25,9],MM[26,9],MM[27,9],MM[28,9],
MM[29,9],MM[30,9],MM[31,9],MM[32,9],MM[33,9],MM[34,9],MM[35,9],MM[36,9],
MM[37,9],MM[38,9],MM[39,9],MM[40,9],MM[41,9],MM[42,9],MM[43,9],MM[44,9],
MM[45,9],MM[46,9],MM[47,9],MM[48,9],MM[49,9],MM[1,10],MM[2,10],MM[3,10],
MM[4,10],MM[5,10],MM[6,10],MM[7,10],MM[8,10],MM[9,10],MM[10,10],MM[11,10],

```
MM[12,10],MM[13,10],MM[14,10],MM[15,10],MM[16,10],MM[17,10],MM[18,10],
MM[19,10],MM[20,10],MM[21,10],MM[22,10],MM[23,10],MM[24,10],MM[25,10],
MM[26,10],MM[27,10],MM[28,10],MM[29,10],MM[30,10],MM[31,10],MM[32,10],
MM[33,10],MM[34,10],MM[35,10],MM[36,10],MM[37,10],MM[38,10],MM[39,10],
MM[40,10],MM[41,10],MM[42,10],MM[43,10],MM[44,10],MM[45,10],MM[46,10],
MM[47,10], MM[48,10], MM[49,10]) )
      const.mat<-rbind(rep(c(1,0),c(49,49*9)),c(rep(0,49),rep(1,49),rep(0,49*8)),
c(rep(0,49*2),rep(1,49),rep(0,49*7)),c(rep(0,49*3),rep(1,49),rep(0,49*6)),
c(rep(0,49*4),rep(1,49),rep(0,49*5)),c(rep(0,49*5),rep(1,49),rep(0,49*4)),
c(rep(0,49*6),rep(1,49),rep(0,49*3)),c(rep(0,49*7),rep(1,49),rep(0,49*2)),
c(rep(0,49*8),rep(1,49),rep(0,49*1)), c(rep(0,49*9),rep(1,49),rep(0,49*0)))
      const.dir<-c("=","=","=","=","=","=","=","=","=","=")
      const.rhs<-c(1,1,1,1,1,1,1,1,1,1)
      sol<-lp (direction = "min", objective.in=objective.in,const.mat=const.mat,
const.dir=const.dir, const.rhs=const.rhs)
      CiudadesReparto1<-which(sol$solution=="1")
      CiudadesReparto<-which(sol$solution=="1")c(0,49,49*2,49*3,49*4,49*5,
49*6,49*7,49*8,49*9)
      CiudadesrutaP<-attributes(Distancias)$dimnames[[2]][CiudadesReparto]
      Ciudadesruta<-as.vector(labels(table(CiudadesrutaP))$CiudadesrutaP)
      if(input$Origen%in%Ciudadesruta==TRUE){
      Matriz<-Distancias[Ciudadesruta,Ciudadesruta]}
      else{Matriz<-Distancias[c(input$Origen,Ciudadesruta),c(input$Origen,
Ciudadesruta)]}
       if (length(Ciudadesruta)<2)  paste(input$Origen)
      else { CiudadesRepartoTSP<-TSP(Matriz, labels = NULL)
      initialtour<-as.integer(which(labels(Matriz)[[1]]==input$Origen[1]))
      method1<-input$Method
      CRtsp <- insert_dummy(CiudadesRepartoTSP, label = "cut")
      results1<-solve_TSP((CRtsp),method1,control=list(start=initialtour))
      labels(results1)[labels(results1)!="cut"]}})
      LongitudRuta<-reactive({
       Matrizkm<-matrix(rep(Distancias[,input$Origen],10),nrow=49)
      A<-dataset()
      FuncionCerof<-function(x,y){if(is.na(A[x,y])) 10000 else A[x,y]}
```

```
B<-as.data.frame(matrix(mapply(FuncionCerof,rep(seq(1,49),rep(10,49)),
rep (seq(1,10),49)),byrow=T,ncol=10))
MM<-as.matrix(input$costekm*Matrizkm+B)
objective.in<-as.numeric(c(MM[1,1],MM[2,1],MM[3,1],MM[4,1],MM[5,1],
MM[6,1],MM[7,1],MM[8,1],MM[9,1],MM[10,1],MM[11,1],MM[12,1],MM[13,1],
MM[14,1],MM[15,1],MM[16,1],MM[17,1],MM[18,1],MM[19,1],MM[20,1],MM[21,1],
MM[22,1],MM[23,1],MM[24,1],MM[25,1],MM[26,1],MM[27,1],MM[28,1],MM[29,1],
MM[30,1],MM[31,1],MM[32,1],MM[33,1],MM[34,1],MM[35,1],MM[36,1],MM[37,1],
MM[38,1],MM[39,1],MM[40,1],MM[41,1],MM[42,1],MM[43,1],MM[44,1],MM[45,1],
MM[46,1],MM[47,1],MM[48,1],MM[49,1],MM[1,2],MM[2,2],MM[3,2],MM[4,2],MM[5,2],
MM[6,2],MM[7,2],MM[8,2],MM[9,2],MM[10,2],MM[11,2],MM[12,2],MM[13,2],
MM[14,2],MM[15,2],MM[16,2],MM[17,2],MM[18,2],MM[19,2],MM[20,2],MM[21,2],
MM[22,2],MM[23,2],MM[24,2],MM[25,2],MM[26,2],MM[27,2],MM[28,2],MM[29,2],
MM[30,2],MM[31,2],MM[32,2],MM[33,2],MM[34,2],MM[35,2],MM[36,2],MM[37,2],
MM[38,2],MM[39,2],MM[40,2],MM[41,2],MM[42,2],MM[43,2],MM[44,2],MM[45,2],
MM[46,2],MM[47,2],MM[48,2],MM[49,2],MM[1,3],MM[2,3],MM[3,3],MM[4,3],MM[5,3],
MM[6,3],MM[7,3],MM[8,3],MM[9,3],MM[10,3],MM[11,3],MM[12,3],MM[13,3],
MM[14,3],MM[15,3],MM[16,3],MM[17,3],MM[18,3],MM[19,3],MM[20,3],MM[21,3],
MM[22,3],MM[23,3],MM[24,3],MM[25,3],MM[26,3],MM[27,3],MM[28,3],MM[29,3],
MM[30,3],MM[31,3],MM[32,3],MM[33,3],MM[34,3],MM[35,3],MM[36,3],MM[37,3],
MM[38,3],MM[39,3],MM[40,3],MM[41,3],MM[42,3],MM[43,3],MM[44,3],MM[45,3],
MM[46,3],MM[47,3],MM[48,3],MM[49,3],MM[1,4],MM[2,4],MM[3,4],MM[4,4],
MM[5,4],MM[6,4],MM[7,4],MM[8,4],MM[9,4],MM[10,4],MM[11,4],MM[12,4],MM[13,4],
MM[14,4],MM[15,4],MM[16,4],MM[17,4],MM[18,4],MM[19,4],MM[20,4],MM[21,4],
MM[22,4],MM[23,4],MM[24,4],MM[25,4],MM[26,4],MM[27,4],MM[28,4],MM[29,4],
MM[30,4],MM[31,4],MM[32,4],MM[33,4],MM[34,4],MM[35,4],MM[36,4],MM[37,4],
MM[38,4],MM[39,4],MM[40,4],MM[41,4],MM[42,4],MM[43,4],MM[44,4],MM[45,4],
MM[46,4],MM[47,4],MM[48,4],MM[49,4],MM[1,5],MM[2,5],MM[3,5],MM[4,5],
MM[5,5],MM[6,5],MM[7,5],MM[8,5],MM[9,5],MM[10,5],MM[11,5],MM[12,5],
MM[13,5],MM[14,5],MM[15,5],MM[16,5],MM[17,5],MM[18,5],MM[19,5],MM[20,5],
MM[21,5],MM[22,5],MM[23,5],MM[24,5],MM[25,5],MM[26,5],MM[27,5],MM[28,5],
MM[29,5],MM[30,5],MM[31,5],MM[32,5],MM[33,5],MM[34,5],MM[35,5],MM[36,5],
MM[37,5],MM[38,5],MM[39,5],MM[40,5],MM[41,5],MM[42,5],MM[43,5],MM[44,5],
MM[45,5],MM[46,5],MM[47,5],MM[48,5],MM[49,5],MM[1,6],MM[2,6],MM[3,6],
MM[4,6],MM[5,6],MM[6,6],MM[7,6],MM[8,6],MM[9,6],MM[10,6],MM[11,6],MM[12,6],
```

MM[13,6],MM[14,6],MM[15,6],MM[16,6],MM[17,6],MM[18,6],MM[19,6],MM[20,6],
MM[21,6],MM[22,6],MM[23,6],MM[24,6],MM[25,6],MM[26,6],MM[27,6],MM[28,6],
MM[29,6],MM[30,6],MM[31,6],MM[32,6],MM[33,6],MM[34,6],MM[35,6],MM[36,6],
MM[37,6],MM[38,6],MM[39,6],MM[40,6],MM[41,6],MM[42,6],MM[43,6],MM[44,6],
MM[45,6],MM[46,6],MM[47,6],MM[48,6],MM[49,6],MM[1,7],MM[2,7],MM[3,7],
MM[4,7],MM[5,7],MM[6,7],MM[7,7],MM[8,7],MM[9,7],MM[10,7],MM[11,7],MM[12,7],
MM[13,7],MM[14,7],MM[15,7],MM[16,7],MM[17,7],MM[18,7],MM[19,7],MM[20,7],
MM[21,7],MM[22,7],MM[23,7],MM[24,7],MM[25,7],MM[26,7],MM[27,7],MM[28,7],
MM[29,7],MM[30,7],MM[31,7],MM[32,7],MM[33,7],MM[34,7],MM[35,7],MM[36,7],
MM[37,7],MM[38,7],MM[39,7],MM[40,7],MM[41,7],MM[42,7],MM[43,7],MM[44,7],
MM[45,7],MM[46,7],MM[47,7],MM[48,7],MM[49,7],MM[1,8],MM[2,8],MM[3,8],
MM[4,8],MM[5,8],MM[6,8],MM[7,8],MM[8,8],MM[9,8],MM[10,8],MM[11,8],MM[12,8],
MM[13,8],MM[14,8],MM[15,8],MM[16,8],MM[17,8],MM[18,8],MM[19,8],MM[20,8],
MM[21,8],MM[22,8],MM[23,8],MM[24,8],MM[25,8],MM[26,8],MM[27,8],MM[28,8],
MM[29,8],MM[30,8],MM[31,8],MM[32,8],MM[33,8],MM[34,8],MM[35,8],MM[36,8],
MM[37,8],MM[38,8],MM[39,8],MM[40,8],MM[41,8],MM[42,8],MM[43,8],MM[44,8],
MM[45,8],MM[46,8],MM[47,8],MM[48,8],MM[49,8],MM[1,9],MM[2,9],MM[3,9],
MM[4,9],MM[5,9],MM[6,9],MM[7,9],MM[8,9],MM[9,9],MM[10,9],MM[11,9],MM[12,9],
MM[13,9],MM[14,9],MM[15,9],MM[16,9],MM[17,9],MM[18,9],MM[19,9],MM[20,9],
MM[21,9],MM[22,9],MM[23,9],MM[24,9],MM[25,9],MM[26,9],MM[27,9],MM[28,9],
MM[29,9],MM[30,9],MM[31,9],MM[32,9],MM[33,9],MM[34,9],MM[35,9],MM[36,9],
MM[37,9],MM[38,9],MM[39,9],MM[40,9],MM[41,9],MM[42,9],MM[43,9],MM[44,9],
MM[45,9],MM[46,9],MM[47,9],MM[48,9],MM[49,9],MM[1,10],MM[2,10],MM[3,10],
MM[4,10],MM[5,10],MM[6,10],MM[7,10],MM[8,10],MM[9,10],MM[10,10],MM[11,10],
MM[12,10],MM[13,10],MM[14,10],MM[15,10],MM[16,10],MM[17,10],MM[18,10],
MM[19,10],MM[20,10],MM[21,10],MM[22,10],MM[23,10],MM[24,10],MM[25,10],
MM[26,10],MM[27,10],MM[28,10],MM[29,10],MM[30,10],MM[31,10],MM[32,10],
MM[33,10],MM[34,10],MM[35,10],MM[36,10],MM[37,10],MM[38,10],MM[39,10],
MM[40,10],MM[41,10],MM[42,10],MM[43,10],MM[44,10],MM[45,10],MM[46,10],
MM[47,10], MM[48,10], MM[49,10]))

const.mat<-rbind(rep(c(1,0),c(49,49*9)),c(rep(0,49),rep(1,49),rep(0,49*8)),
c(rep(0,49*2),rep(1,49),rep(0,49*7)),c(rep(0,49*3),rep(1,49),rep(0,49*6)),
c(rep(0,49*4),rep(1,49),rep(0,49*5)),c(rep(0,49*5),rep(1,49),rep(0,49*4)),
c(rep(0,49*6),rep(1,49),rep(0,49*3)),c(rep(0,49*7),rep(1,49),rep(0,49*2)),
c(rep(0,49*8),rep(1,49),rep(0,49*1)), c(rep(0,49*9),rep(1,49),rep(0,49*0)))

```
const.dir<-c("=","=","=","=","=","=","=","=","=","=")
const.rhs<-c(1,1,1,1,1,1,1,1,1,1)
sol<-lp (direction = "min", objective.in=objective.in,const.mat=const.mat,
const.dir=const.dir, const.rhs=const.rhs)
CiudadesReparto1<-which(sol$solution=="1")
CiudadesReparto<-which(sol$solution=="1")c(0,49,49*2,49*3,49*4,49*5,
49*6,49*7,49*8,49*9)
CiudadesrutaP<-attributes(Distancias)$dimnames[[2]][CiudadesReparto]
Ciudadesruta<-as.vector(labels(table(CiudadesrutaP))$CiudadesrutaP)
if(input$Origen%in%Ciudadesruta==TRUE){
Matriz<-Distancias[Ciudadesruta,Ciudadesruta]}
else{Matriz<-Distancias[c(input$Origen,Ciudadesruta),c(input$Origen,
Ciudadesruta)]}
if(length(Ciudadesruta)<2) (0)
else {CiudadesRepartoTSP<-TSP(Matriz, labels = NULL)
CRtsp <- insert_dummy(CiudadesRepartoTSP, label = "cut")
method1<-input$Method
initialtour<-as.integer(which(labels(Matriz)[[1]]==input$Origen[1]))
results1<-solve_TSP((CRtsp),method1,control=list(start=initialtour))
V<-as.integer(results1)
tour_length(CRtsp, V)}})
CiudadesProducto<-reactive({
Matrizkm<-matrix(rep(Distancias[,input$Origen],10),nrow=49)
A<-dataset()
FuncionCerof<-function(x,y){if(is.na(A[x,y])) 10000 else A[x,y]}
B<-as.data.frame(matrix(mapply(FuncionCerof,rep(seq(1,49),rep(10,49)),
rep (seq(1,10),49)),byrow=T,ncol=10))
MM<-as.matrix(input$costekm*Matrizkm+B)
objective.in<-as.numeric(c(MM[1,1],MM[2,1],MM[3,1],MM[4,1],MM[5,1],
MM[6,1],MM[7,1],MM[8,1],MM[9,1],MM[10,1],MM[11,1],MM[12,1],MM[13,1],
MM[14,1],MM[15,1],MM[16,1],MM[17,1],MM[18,1],MM[19,1],MM[20,1],MM[21,1],
MM[22,1],MM[23,1],MM[24,1],MM[25,1],MM[26,1],MM[27,1],MM[28,1],MM[29,1],
MM[30,1],MM[31,1],MM[32,1],MM[33,1],MM[34,1],MM[35,1],MM[36,1],MM[37,1],
MM[38,1],MM[39,1],MM[40,1],MM[41,1],MM[42,1],MM[43,1],MM[44,1],MM[45,1],
MM[46,1],MM[47,1],MM[48,1],MM[49,1],MM[1,2],MM[2,2],MM[3,2],MM[4,2],MM[5,2],
```

MM[6,2],MM[7,2],MM[8,2],MM[9,2],MM[10,2],MM[11,2],MM[12,2],MM[13,2],
MM[14,2],MM[15,2],MM[16,2],MM[17,2],MM[18,2],MM[19,2],MM[20,2],MM[21,2],
MM[22,2],MM[23,2],MM[24,2],MM[25,2],MM[26,2],MM[27,2],MM[28,2],MM[29,2],
MM[30,2],MM[31,2],MM[32,2],MM[33,2],MM[34,2],MM[35,2],MM[36,2],MM[37,2],
MM[38,2],MM[39,2],MM[40,2],MM[41,2],MM[42,2],MM[43,2],MM[44,2],MM[45,2],
MM[46,2],MM[47,2],MM[48,2],MM[49,2],MM[1,3],MM[2,3],MM[3,3],MM[4,3],MM[5,3],
MM[6,3],MM[7,3],MM[8,3],MM[9,3],MM[10,3],MM[11,3],MM[12,3],MM[13,3],
MM[14,3],MM[15,3],MM[16,3],MM[17,3],MM[18,3],MM[19,3],MM[20,3],MM[21,3],
MM[22,3],MM[23,3],MM[24,3],MM[25,3],MM[26,3],MM[27,3],MM[28,3],MM[29,3],
MM[30,3],MM[31,3],MM[32,3],MM[33,3],MM[34,3],MM[35,3],MM[36,3],MM[37,3],
MM[38,3],MM[39,3],MM[40,3],MM[41,3],MM[42,3],MM[43,3],MM[44,3],MM[45,3],
MM[46,3],MM[47,3],MM[48,3],MM[49,3],MM[1,4],MM[2,4],MM[3,4],MM[4,4],
MM[5,4],MM[6,4],MM[7,4],MM[8,4],MM[9,4],MM[10,4],MM[11,4],MM[12,4],MM[13,4],
MM[14,4],MM[15,4],MM[16,4],MM[17,4],MM[18,4],MM[19,4],MM[20,4],MM[21,4],
MM[22,4],MM[23,4],MM[24,4],MM[25,4],MM[26,4],MM[27,4],MM[28,4],MM[29,4],
MM[30,4],MM[31,4],MM[32,4],MM[33,4],MM[34,4],MM[35,4],MM[36,4],MM[37,4],
MM[38,4],MM[39,4],MM[40,4],MM[41,4],MM[42,4],MM[43,4],MM[44,4],MM[45,4],
MM[46,4],MM[47,4],MM[48,4],MM[49,4],MM[1,5],MM[2,5],MM[3,5],MM[4,5],
MM[5,5],MM[6,5],MM[7,5],MM[8,5],MM[9,5],MM[10,5],MM[11,5],MM[12,5],
MM[13,5],MM[14,5],MM[15,5],MM[16,5],MM[17,5],MM[18,5],MM[19,5],MM[20,5],
MM[21,5],MM[22,5],MM[23,5],MM[24,5],MM[25,5],MM[26,5],MM[27,5],MM[28,5],
MM[29,5],MM[30,5],MM[31,5],MM[32,5],MM[33,5],MM[34,5],MM[35,5],MM[36,5],
MM[37,5],MM[38,5],MM[39,5],MM[40,5],MM[41,5],MM[42,5],MM[43,5],MM[44,5],
MM[45,5],MM[46,5],MM[47,5],MM[48,5],MM[49,5],MM[1,6],MM[2,6],MM[3,6],
MM[4,6],MM[5,6],MM[6,6],MM[7,6],MM[8,6],MM[9,6],MM[10,6],MM[11,6],MM[12,6],
MM[13,6],MM[14,6],MM[15,6],MM[16,6],MM[17,6],MM[18,6],MM[19,6],MM[20,6],
MM[21,6],MM[22,6],MM[23,6],MM[24,6],MM[25,6],MM[26,6],MM[27,6],MM[28,6],
MM[29,6],MM[30,6],MM[31,6],MM[32,6],MM[33,6],MM[34,6],MM[35,6],MM[36,6],
MM[37,6],MM[38,6],MM[39,6],MM[40,6],MM[41,6],MM[42,6],MM[43,6],MM[44,6],
MM[45,6],MM[46,6],MM[47,6],MM[48,6],MM[49,6],MM[1,7],MM[2,7],MM[3,7],
MM[4,7],MM[5,7],MM[6,7],MM[7,7],MM[8,7],MM[9,7],MM[10,7],MM[11,7],MM[12,7],
MM[13,7],MM[14,7],MM[15,7],MM[16,7],MM[17,7],MM[18,7],MM[19,7],MM[20,7],
MM[21,7],MM[22,7],MM[23,7],MM[24,7],MM[25,7],MM[26,7],MM[27,7],MM[28,7],
MM[29,7],MM[30,7],MM[31,7],MM[32,7],MM[33,7],MM[34,7],MM[35,7],MM[36,7],
MM[37,7],MM[38,7],MM[39,7],MM[40,7],MM[41,7],MM[42,7],MM[43,7],MM[44,7],

```
MM[45,7],MM[46,7],MM[47,7],MM[48,7],MM[49,7],MM[1,8],MM[2,8],MM[3,8],
MM[4,8],MM[5,8],MM[6,8],MM[7,8],MM[8,8],MM[9,8],MM[10,8],MM[11,8],MM[12,8],
MM[13,8],MM[14,8],MM[15,8],MM[16,8],MM[17,8],MM[18,8],MM[19,8],MM[20,8],
MM[21,8],MM[22,8],MM[23,8],MM[24,8],MM[25,8],MM[26,8],MM[27,8],MM[28,8],
MM[29,8],MM[30,8],MM[31,8],MM[32,8],MM[33,8],MM[34,8],MM[35,8],MM[36,8],
MM[37,8],MM[38,8],MM[39,8],MM[40,8],MM[41,8],MM[42,8],MM[43,8],MM[44,8],
MM[45,8],MM[46,8],MM[47,8],MM[48,8],MM[49,8],MM[1,9],MM[2,9],MM[3,9],
MM[4,9],MM[5,9],MM[6,9],MM[7,9],MM[8,9],MM[9,9],MM[10,9],MM[11,9],MM[12,9],
MM[13,9],MM[14,9],MM[15,9],MM[16,9],MM[17,9],MM[18,9],MM[19,9],MM[20,9],
MM[21,9],MM[22,9],MM[23,9],MM[24,9],MM[25,9],MM[26,9],MM[27,9],MM[28,9],
MM[29,9],MM[30,9],MM[31,9],MM[32,9],MM[33,9],MM[34,9],MM[35,9],MM[36,9],
MM[37,9],MM[38,9],MM[39,9],MM[40,9],MM[41,9],MM[42,9],MM[43,9],MM[44,9],
MM[45,9],MM[46,9],MM[47,9],MM[48,9],MM[49,9],MM[1,10],MM[2,10],MM[3,10],
MM[4,10],MM[5,10],MM[6,10],MM[7,10],MM[8,10],MM[9,10],MM[10,10],MM[11,10],
MM[12,10],MM[13,10],MM[14,10],MM[15,10],MM[16,10],MM[17,10],MM[18,10],
MM[19,10],MM[20,10],MM[21,10],MM[22,10],MM[23,10],MM[24,10],MM[25,10],
MM[26,10],MM[27,10],MM[28,10],MM[29,10],MM[30,10],MM[31,10],MM[32,10],
MM[33,10],MM[34,10],MM[35,10],MM[36,10],MM[37,10],MM[38,10],MM[39,10],
MM[40,10],MM[41,10],MM[42,10],MM[43,10],MM[44,10],MM[45,10],MM[46,10],
MM[47,10], MM[48,10], MM[49,10]) )
    const.mat<-rbind(rep(c(1,0),c(49,49*9)),c(rep(0,49),rep(1,49),rep(0,49*8)),
c(rep(0,49*2),rep(1,49),rep(0,49*7)),c(rep(0,49*3),rep(1,49),rep(0,49*6)),
c(rep(0,49*4),rep(1,49),rep(0,49*5)),c(rep(0,49*5),rep(1,49),rep(0,49*4)),
c(rep(0,49*6),rep(1,49),rep(0,49*3)),c(rep(0,49*7),rep(1,49),rep(0,49*2)),
c(rep(0,49*8),rep(1,49),rep(0,49*1)), c(rep(0,49*9),rep(1,49),rep(0,49*0)))
    const.dir<-c("=","=","=","=","=","=","=","=","=","=")
    const.rhs<-c(1,1,1,1,1,1,1,1,1,1)
    sol<-lp (direction = "min", objective.in=objective.in,const.mat=const.mat,
const.dir=const.dir, const.rhs=const.rhs)
    CiudadesReparto1<-which(sol$solution=="1")
    CiudadesReparto<-which(sol$solution=="1")-c(0,49,49*2,49*3,49*4,49*5,
49*6,49*7,49*8,49*9)
    CiudadesrutaP<-attributes(Distancias)$dimnames[[2]][CiudadesReparto]
    Ciudades<-attributes(Distancias)$dimnames[[2]][CiudadesReparto]
    Ciudades})
```

```
output$DatosCostes<-renderTable({TablaCostes<-dataset()
Costes<c("ALBACETE","ALICANTE","ALMERIA","AVILA","BADAJOZ",
"BARCELONA","BILBAO","BURGOS","CACERES","CADIZ","CASTELLON",
"CIUDAD.REAL","CORDOBA","CUENCA","GERONA","GRANADA",
"GUADALAJARA","HUELVA","HUESCA","JAEN","LA.CORUNA","LEON",
"LERIDA","LOGRONO","LUGO","MADRID","MALAGA","MERIDA","MURCIA",
"ORENSE","OVIEDO","PALENCIA","PAMPLONA","PONTEVEDRA","SALAMANCA",
"SAN.SEBASTIAN","SANTANDER","SANTIAGO.DE.COMPOSTELA","SEGOVIA",
"SEVILLA","SORIA","TARRAGONA","TERUEL","TOLEDO","VALENCIA",
"VALLADOLID","VITORIA","ZAMORA","ZARAGOZA")
dimnames(TablaCostes)[[1]]<-Costes
TablaCostes})
output$CiudadesConProducto<-renderTable({rbind(CiudadesProducto())})
output$RutaDeCompra<-renderTable({Tabla<-data.frame(RutaCompra())
names(Tabla)<-"Rutadecompra"
Tabla})
output$LongitudDeLaRuta<- renderPrint({LongitudRuta()})})
```

1.2. UI.R

```
library(shiny)
shinyUI(fluidPage(
titlePanel(img(src="SY.png")),
sidebarLayout(
 inputPanel(
fileInput('file1',"Introducir archivos de precios",accept=c('sheetName','header'),
multiple=FALSE),
numericInput("costekm","Coste transporte por km", min=0, max=100, value =
0.05 ),
selectInput("Origen",label="OrigenRuta,choices=c("ALBACETE","ALICANTE",
"ALMERÍA"="ALMERIA","ÁVILA"="AVILA","BADAJOZ","BARCELONA","BILBAO",
"BURGOS","CÁCERES"="CACERES","CÁDIZ"="CADIZ","CASTELLÓN"="CASTELLON",
"CIUDA DREAL"="CIUDAD.REAL","CÓRDOBA"="CORDOBA","CUENCA","GERONA",
"GRANADA","GUADALAJARA","HUELVA","HUESCA","JAÉN"="JAEN",
"LACORUÑA"="LA.CORUNA","LEÓN"="LEON","LÉRIDA"="LERIDA",
```


Autor: **Alejandro Sarabia Hernando**
Nº TFG: **425.16.108**

```
"LOGROÑO"="LOGRONO","LUGO","MADRID","MÁLAGA"="MALAGA",
"MÉRIDA"="MERIDA","MURCIA","ORENSE","OVIEDO","PALENCIA","PAMPLONA",
"PONTEVEDRA","SALAMANCA",SAN  SEBASTIÁN"="SAN.SEBASTIAN",  "SANTANDER",
"SANTIAGODE COMPOSTELA"="SANTIAGO.DE.COMPOSTELA", "SEGOVIA", "SEVILLA",
"SORIA","TARRAGONA","TERUEL","TOLEDO","VALENCIA","VALLADOLID","VITORIA",
"ZAMORA","ZARAGOZA"),selected="MADRID"),
      selectInput("Method",label="Algoritmo",choices=c("nearest_insertion",
"farthest_insertion","cheapest_insertion","arbitrary_insertion"),selected=
"nearest_insertion"),
      width=12),
      mainPanel(tabsetPanel(tabPanel(strong(p(span("Introducción"))),
h3(strong(span("Bienvenido a Purchaser App",style= "color:blue"))),
h4(span("Esta aplicación te permite obtener, de forma rápida y simple, la mejor ruta
para comprar los productos que necesitas.")),
h4(span("Solo tienes que seguir estos sencillos pasos:")),
h5(span("- Cargar el fichero xls con los precios de los productos. La tabla debe ser
49x10(CiudadesxProductos), todos datos deben ser númericos y en caso de no haber
producto en alguna ciudad dejar un espacio en blanco. ")),
h5(span("- Introducir el coste por kilometro del transporte utilizado.")),
h5(span("-Seleccionar la ciudad origen de la ruta.")),
h5(span("- Elegir el algoritmo con el que se desea obtener la ruta óptima. ")),
h6(em("**nearest_insertion. "),"En cada paso elige la ciudad más cercana a una
ciudad de la ruta. "),
h6(em("**farthest_insertion."),"En cada paso elige la ciudad más lejana a una ciudad
de la ruta. "),
h6(em("**cheapest_insertion. "),"En cada paso elige la ciudad que hace minimo el
incremento de la ruta. "),
h6(em("**arbitrary_insertion. "),"En cada paso elige la ciudad arbitrariamente entre
las que quedan por incluir en la ruta."),
h4(span("Una vez eligidos los campos anteriores observar la  ruta  y su longitud en  la
pestaña",strong(" Ruta de compra",style= "color:blue")))),
      tabPanel(strong(p(span("Matriz de costes"))),tableOutput("DatosCostes")),
      tabPanel(strong(p("Rutadecompra")),
h5(strong(span("Ciudades de compra del producto"))),
tableOutput("CiudadesConProducto"),
```

```
h5(strong(span("Longitud de la ruta (Km)"))),
verbatimTextOutput("LongitudDeLaRuta"),
tableOutput("RutaDeCompra"),
h5(strong(span("Mapa de España"))),
      img(src="RA.png")))))))
```


Autor: **Alejandro Sarabia Hernando**
Nº TFG: **425.16.108**

2. MANUALES DE LA APLICACIÓN

2.1. MANUAL DE INSTALACIÓN EN EQUIPO PROPIO

Para realizar la instalación de la aplicación "Purchaser App" en otro equipo basta con seguir estos sencillos pasos:

1. Cumplir los requisitos previos

Estos requisitos hacen referencia a tener instalado el software RStudio, el cual se puede conseguir en su página web oficial *https://www.rstudio.com/*.

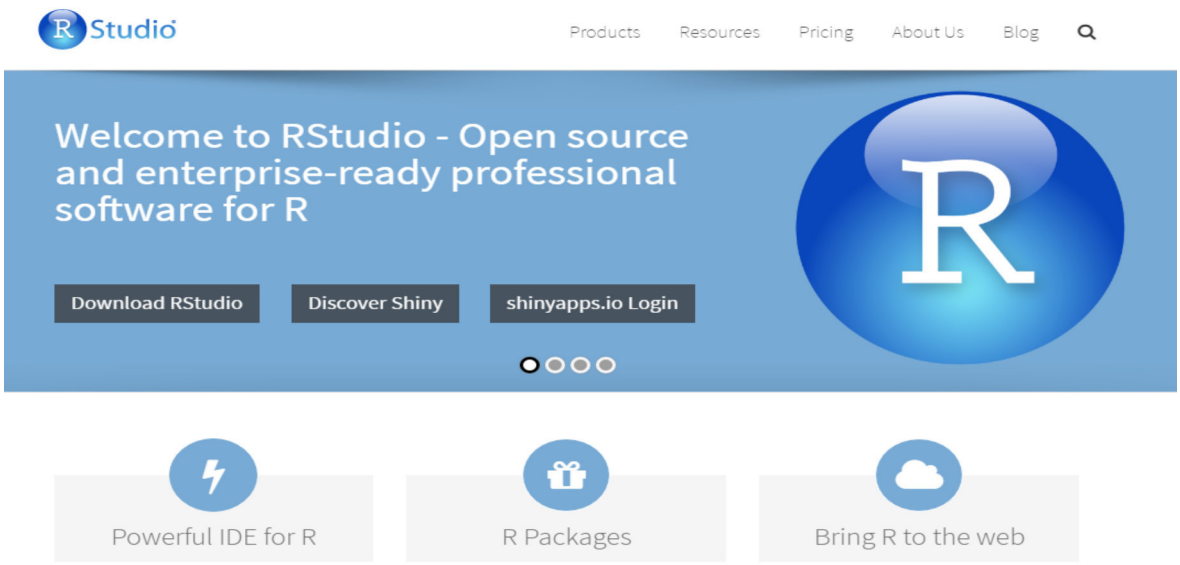

Ilustración 1: www.rstudio.com

2. Copiar la carpeta "AppPurchaser" en el equipo

En esta carpeta se encuentran los archivos necesarios para el funcionamiento de la aplicación. Estos son:

- server.R
- ui.R
- Carpeta *www*. En ella se encuentran las imágenes de la aplicación.

Esta carpeta se puede encontrar en el CD entregado junto al TFG.

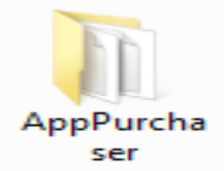

Ilustración 2: Carpeta AppPurchaser

En caso de no poder acceder a él, mandar un e-mail a la dirección de correo *asarabiahernado@hotmail.com* especificando la razón por la cual se quiere obtener la aplicación. Si la razón se considera adecuada se mandara la carpeta a la dirección de correo que ha efectuado la solicitud.

3. Abrir RStudio e instalar los paquetes necesarios

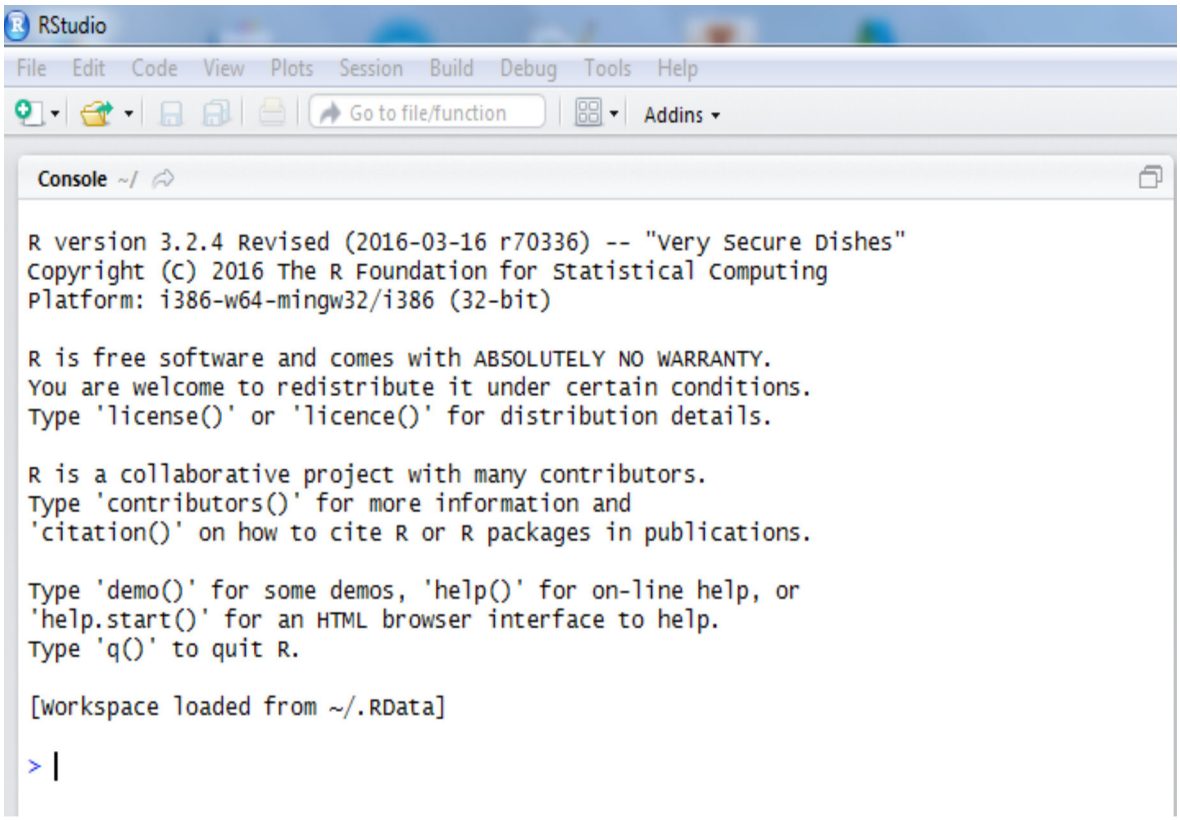

Ilustración 3: Consola RStudio

Los paquetes necesarios que se tienen que preinstalar para luego ser cargados son:

- shiny
- TSP
- rJava
- xlsx
- lpSolve

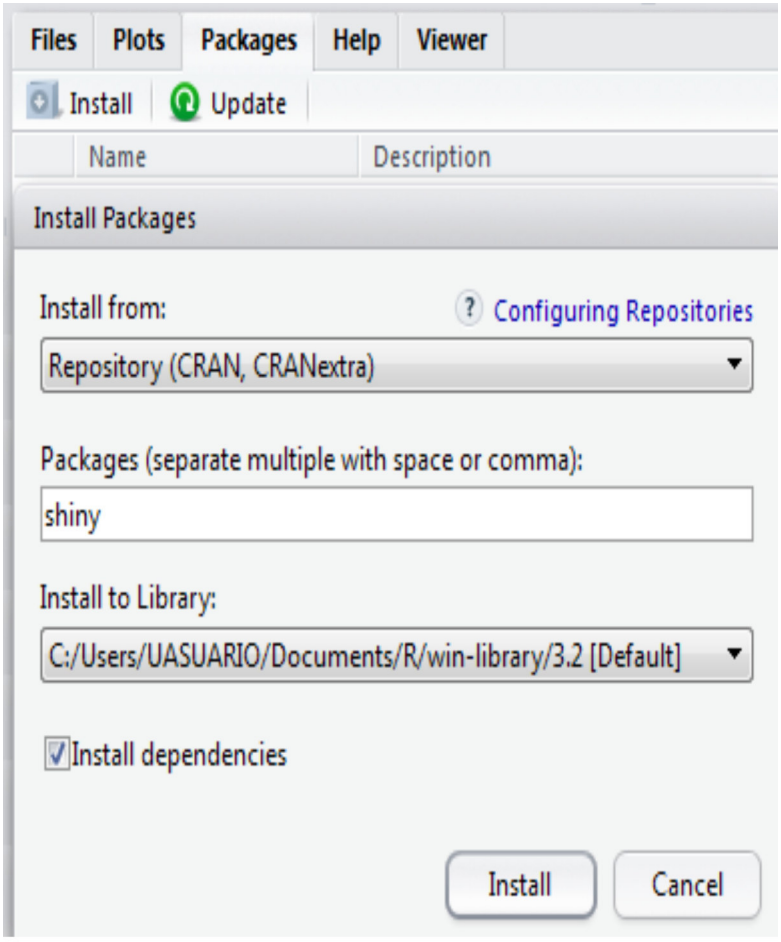

Ilustración 4: Instalación paquetes

4. Cargar los archivos server.R y ui.R

Se realiza con la secuencia: File ——>Open File.

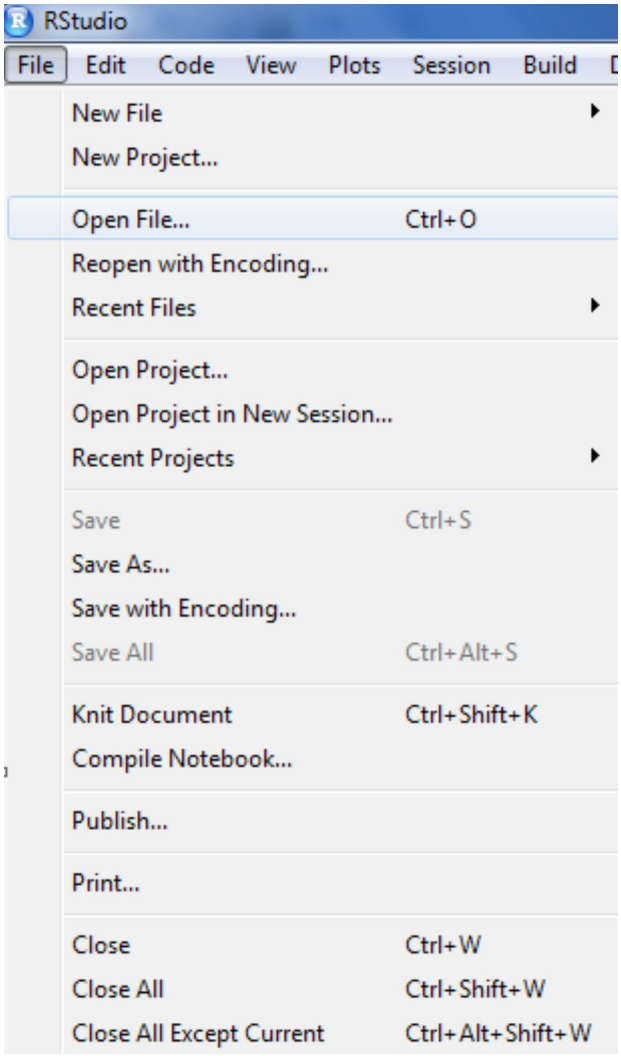

Ilustración 5: Cargar archivos

5. Correr el código

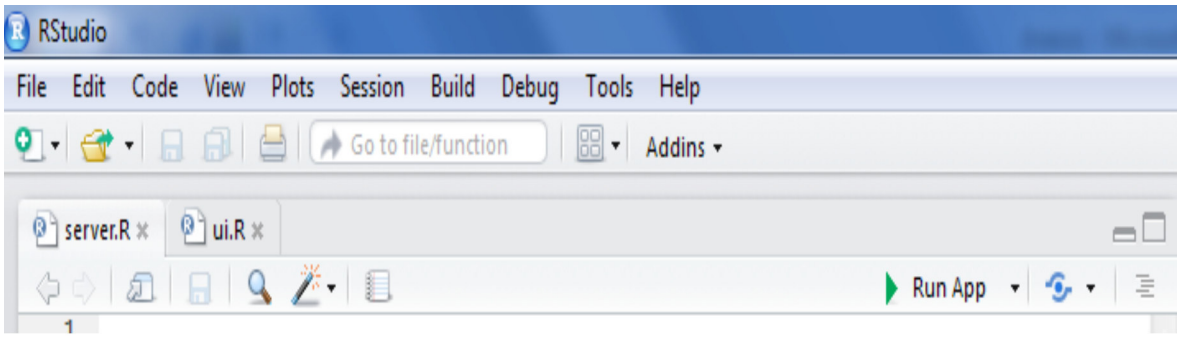

Ilustración 6: Correr código

2.2. MANUAL DE UTILIZACIÓN

El manejo de esta aplicación es fácil e intuitivo gracias a su interfaz sencilla y atractiva a la vista. A continuación se dan algunas instrucciones:

- Introducir archivos de precios

  - Cargar el fichero xls con los precios de los productos
  - La tabla debe ser 49x10 (CiudadesxProductos)
  - Todos datos deben ser numéricos.
  - En caso de no haber producto en alguna ciudad dejar un espacio en blanco
  - El fichero cargado puede observarse en la pestaña "Matriz de costes"

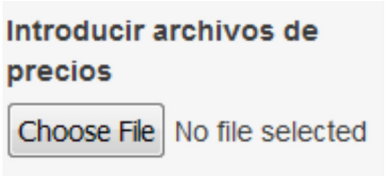

Ilustración 7: Introducir archivo de precios

- Coste tranporte por kilometro

  - Introducir el coste por kilometro del transporte utilizado

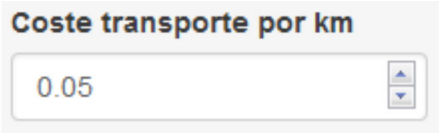

Ilustración 8: Coste transporte por km

- Origen ruta

  - Seleccionar la ciudad origen de la ruta
  - Siempre aparecerá la primera en la ruta de compra
  - Aparecen en orden alfabético

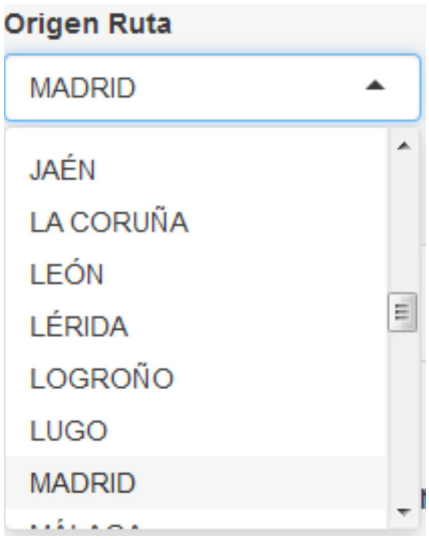

Ilustración 9: Origen ruta

- Algoritmo

  - Elegir el algoritmo con el que se desea obtener la ruta optima
  - Los algoritmos disponibles son:
    - nearest_insertion: En cada paso elige la ciudad más cercana a una ciudad de la ruta
    - farthest_insertion: En cada paso elige la ciudad más lejana a una ciudad de la ruta
    - cheapest_insertion: En cada paso elige la ciudad que hace mínimo el incremento de la ruta
    - arbitrary_insertion: En cada paso elige la ciudad arbitrariamente entre las que quedan por incluir en la ruta
  - Se aconseja probar todos los algoritmos para constatar cual da la menos distancia

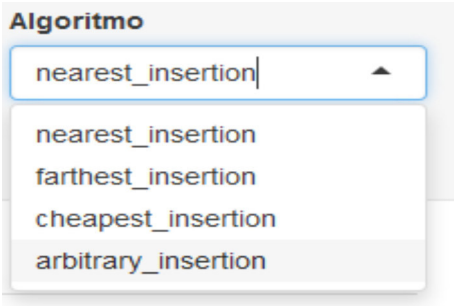

Ilustración 10: Algoritmo

- Resultados

  - En la pestaña *Ruta de compra* se pueden visualizar:
    - Ciudades de compra del producto
    - Longitud de la ruta (Km)
    - Ruta de compra
  - Las soluciones se actualizan automáticamente cuando se realiza algún cambio en la elección de datos

Ciudades de compra del producto

	1	2	3	4	5	6	7	8	9	10
1	ALMERIA	GUADALAJARA	MALAGA	TERUEL	BARCELONA	SALAMANCA	LOGRONO	SEGOVIA	CUENCA	MURCIA

Longitud de la ruta (Km)

```
[1] 2257
```

	Ruta de compra
1	MADRID
2	GUADALAJARA
3	CUENCA
4	TERUEL
5	LOGRONO
6	BARCELONA
7	MURCIA
8	ALMERIA
9	MALAGA
10	SALAMANCA
11	SEGOVIA

Ilustración 11: Resultados

3. LISTA DE PRODUCTOS

En la aplicación se necesita que la matriz contega la distancias entre todas las ciudades. Esto se puede obtener con la función shortest.paths().

	Producto 1	Producto 2	Producto 3	Producto 4	Producto 5
Albacete	2049	2047	2011	1997	2037
Alicante	2082	2021	2030	1980	2023
Almería	1932	1961	2070	2003	1979
Ávila	2016	1966	2014	1959	1927
Badajoz	1979	1987	1981	1949	2101
Barcelona	2108	2069	2021	1946	1899
Bilbao	1962	1994	2011	1942	1989
Burgos	1965	1965	2028	2012	1991
Cáceres	2039	1994	1976	1962	2043
Cádiz	1932	1984	2021	2091	2020
Castellón	1962	2067	2029	2002	2058
Ciudad Real	2045	2026	1995	2051	2021
Córdoba	2004	2016	2015		2004
Cuenca	2053	2049	2077	2061	
Gerona	1935	2007	1962	2009	2025
Granada	2034	1967	1952	2097	1947
Guadalajara	1963	1911	2034	2040	2051
Huelva	2022	2029	1937	1989	2017
Huesca	1973	1983	2001	2006	1957
Jaén	2018	2014	2065	2037	2098
La Coruña	1949	2007	2007	2021	2044
León	1992	1990	1933	2005	1951

Autor: **Alejandro Sarabia Hernando**
Nº TFG: **425.16.108**

Lérida	2014	2035	1959	1971	1981
Logroño	1993	2014	2017	1996	1933
Lugo	1973	1961	2019	2016	1934
Madrid	2039	2068	1955	1996	2030
Málaga	2026	2038	1890	1939	1976
Mérida	2049	1927	2028	2005	1984
Murcia	2081	1991	2047	2057	2048
Orense	2027	2009	1956	1932	2006
Oviedo	1987	2008	2022	2118	2051
Palencia	1999	1963	1940	2001	2034
Pamplona	2001	1989	2000	1940	2030
Pontevedra	2016	1967	1999	2025	1999
Salamanca	1975	2075	2002	2015	2042
S.Sebastian	1968	2003	2016	2043	1983
Santander	2045	1980	2014	2060	2014
S.Compostela	2088	1930	2066	2008	1960
Segovia	2075	2045	1934	2047	2056
Sevilla	1987	1938	1924	2067	2020
Soria	2008	1998	1940	2012	2039
Tarragona	2066	1986	1962	1999	1969
Teruel	2001	1993	1992	1916	1972
Toledo	1957	2004	1977	2102	2002
Valencia	1997	1952	2038	2032	1934
Valladolid	1974	2026	2023	1957	1991
Vitoria	2017	2051	1992	1983	1963
Zamora	1989	1977	2037	2080	2019
Zaragoza	2000	1998	1957	1978	2038

Tabla 1: Lista de precios

	Producto 6	Producto 7	Producto 8	Producto 9	Producto 10
Albacete	1960	1952	2020	1953	2003
Alicante	1987	2029	2014	2031	2021
Almería	1997	1950	2036	1949	1984
Ávila	1988	2067	2004	1966	2015
Badajoz	1965	1927	2054	2053	1966
Barcelona	2028	2007	2046	1947	2073
Bilbao	2061	1971	1961	1970	2018
Burgos	1970	2047	2055	2029	2061
Cáceres	1962	1955	2040	1950	1936
Cádiz	2014	2050	1898	2007	2010
Castellón	1962	2023	1958	1947	2036
Ciudad Real	1994	1968	2049	1967	1968
Córdoba	2043	1987	1939	2018	1970
Cuenca	2049	1989	2046	1919	2017
Gerona	1938	2025	1988	1953	2020
Granada	2143	2003	2083	2060	1957
Guadalajara	2080	1997	2018	2050	2039
Huelva	2014	1942	1978	1944	1980
Huesca	1957	2053	1954	1993	2036
Jaén	2070		1984	1913	2021
La Coruña	2023	1936	2000	1918	2067
León	1985	1983	1970	1983	2054
Lérida	2023	2024	1995	2003	1995
Logroño	2015	1921	1963	2031	2054
Lugo	2030	1990	1948	1986	2028
Madrid	2045	1979	2009	1999	1997
Málaga	1992	2016	1997	2009	2011

Autor: **Alejandro Sarabia Hernando**
Nº TFG: **425.16.108**

Mérida	2005	2067		2012	2065
Murcia	2029	1992	1961	2123	1879
Orense	2076	2044	2034	1912	1954
Oviedo	2067	2072	1947	2008	2052
Palencia	2025	1975	1984	2033	2073
Pamplona	1987	1950	2009	2100	2030
Pontevedra	1973	2072	2034	1986	2040
Salamanca	1932	2031	1951	2003	2039
S.Sebastian	2003	2010	2019	2002	2019
Santander	2027	1978	2042	2058	2015
S.Compostela	1978	1999	1962	1986	1999
Segovia	1980	2042	1925	2067	2027
Sevilla	2000	2060	1958	1932	1972
Soria	1956	1995	1990	1959	1960
Tarragona	1941	1959	1949	2016	1976
Teruel	1998	2029	2012	2071	1978
Toledo	2005	1988	1951	2029	2025
Valencia	1944	1982	1964	1957	2052
Valladolid	2049	2036	1965	1993	1981
Vitoria	1956	1948	2115	1960	1985
Zamora	2041	2044	1999	2033	1906
Zaragoza	1933	1941	2064	2023	1968

Tabla 2: Lista de precios

Relación de documentos

(_) Memoria .....80 páginas
(X) Anexos .....27 páginas

La Almunia, a 29 de Junio de 2016

Firmado: Alejandro Sarabia Hernando