Chapter 4

Conclusion and suggestion

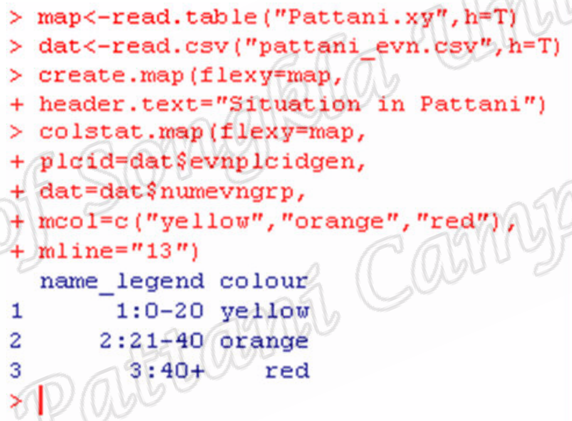
4.1 Conclusion

In this study we created 10 functions. We can divide them to three groups. The first group being functions to manage regions, namely *create.map*(), *setcol.map*(), *setcol.cmap*(), *setnme.map*() and *combine.map*(). The second group contains functions to show statistics data for each region. These are *colstat.map*() and *piestat.map*(). The third group contains functions to compute area, perimeter and center of regions, and these are *area.map*(), *perimeter.map*() and *center.map*().

The *create.map*() function is the main function to use to create a map. The *setcol.map*() function is a function for specifying color of each region. The *setcol.cmap*() function is a function for specifying color for a complex region. The *setnme.map*() function displays name on each region. The *combine.map*() function allows users to combine different regions into one region. The *colstat.map*() and *piestat.map*() function display statistical data on a map, *colstat.map*() displays color shade on a map to represent the statistical data and *piestat.map*() function shows the circle sign. The *area.map*() function computes the area of each region. The *perimeter.map*() function computes the perimeter of each region and the *center.map*() function computes the center of each region

4.2 An example of application of functions

Figure 4.1, this example uses social unrest data from the four southern-most provinces of Thailand, namely Pattani, Yala, Narathiwas and Songkhla. The regions include all districts of Pattani, Yala and Narathiwas province plus four districts of Songkhla. These districts have been subjected to continuing social unrest over recent years. It shows the distribution of violent events in Pattani province. Three contrasting colors were used to display the level of unrest in the area, based on the number of events in each district.



```
> map<-read.table("Pattani.xy",h=T)
> dat<-read.csv("pattani_evn.csv",h=T)
> create.map(flexy=map,
+ header.text="Situation in Pattani")
> colstat.map(flexy=map,
+ plcid=dat$evnplcidgen,
+ dat=dat$numevngrp,
+ mcol=c("yellow","orange","red"),
+ mline="13")
  name_legend colour
1      1:0-20 yellow
2     2:21-40 orange
3       3:40+    red
> 
```

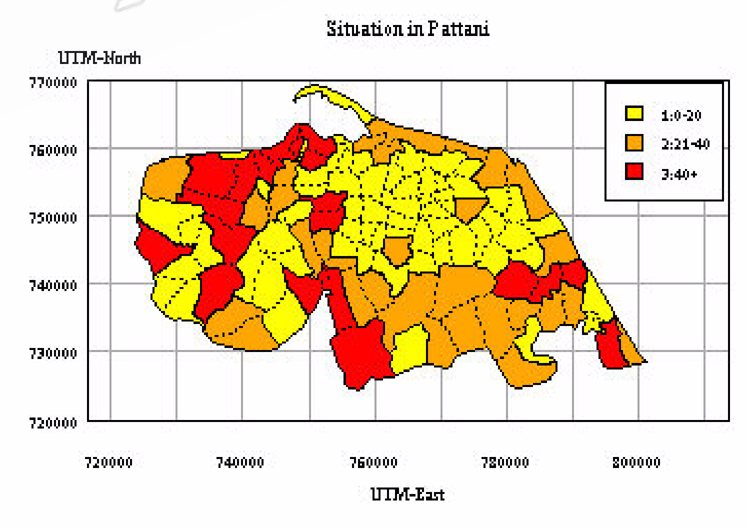
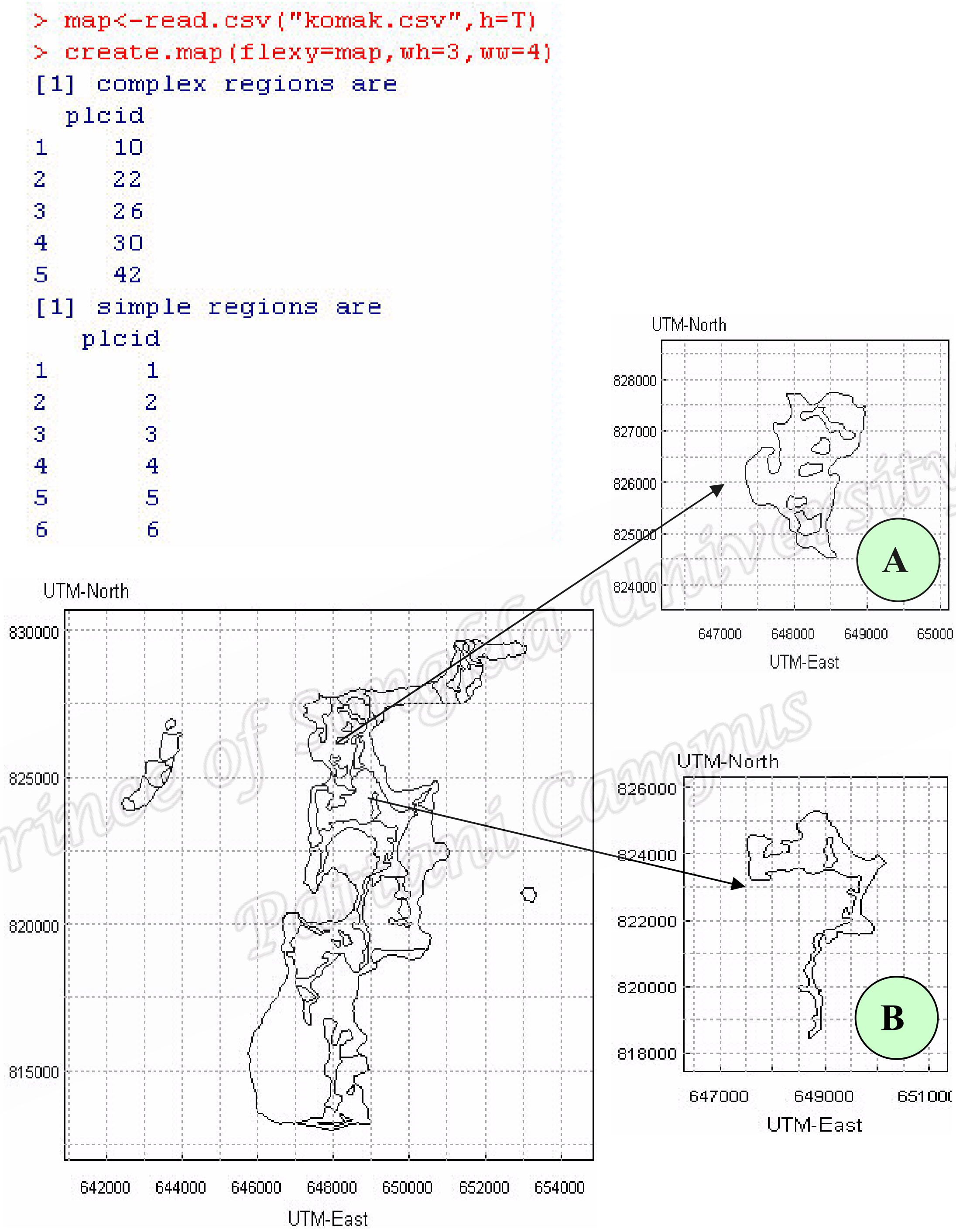


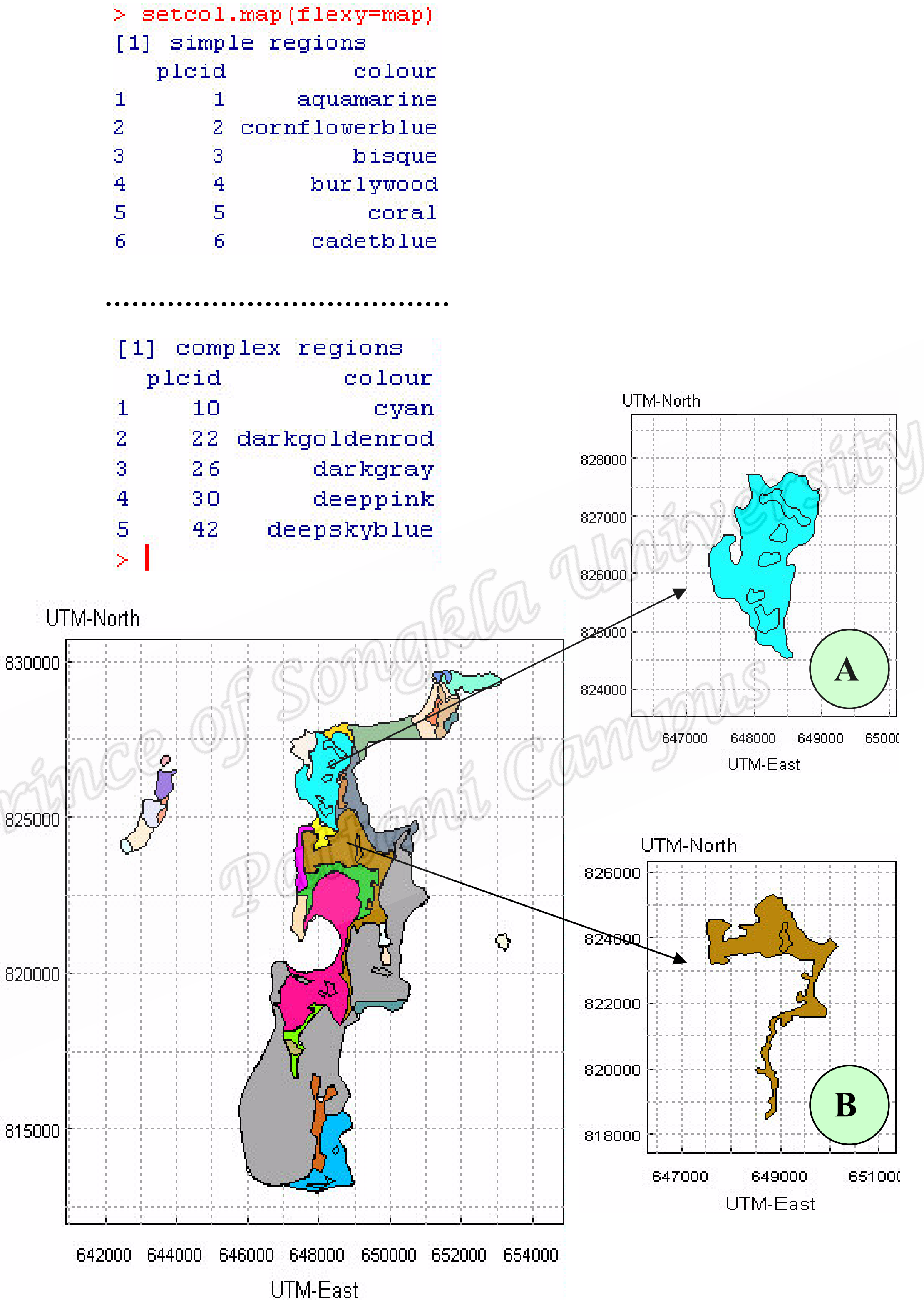
Figure 4.1: Map showing terrorist events in Pattani province

If a data set has complex regions and simple regions the user can create a map using the *create.map*() function. An example of the result is shown in figure 4.2. It is Ko Mak sub-district, Pak Phayun district, Phatthalung Province. This map has five complex regions. To see more clearly, we zoom two regions, which we call A and B. Region A has four regions and region B has twoegions. When the user uses *setcol.map*() function and specifies different color for each region, the outcome is the same color in each region of a complex region. Such a result is shown in figure 4.3. A is blue color and B is brown color. If user needs to specify different color in a complex region, user can do it using *setcol.cmap*() function. Such a result is shown in figure 4.4. If user uses the *area.map*() function to compute the area of each region, the results on R Console will show the area of each region for simple regions and each region of each complex region. An example is shown in figure 4.5.



```
> map<-read.csv("komak.csv",h=T)
> create.map(flexy=map,wh=3,ww=4)
[1] complex regions are
  plcid
1     10
2     22
3     26
4     30
5     42
[1] simple regions are
   plcid
1      1
2      2
3      3
4      4
5      5
6      6
```

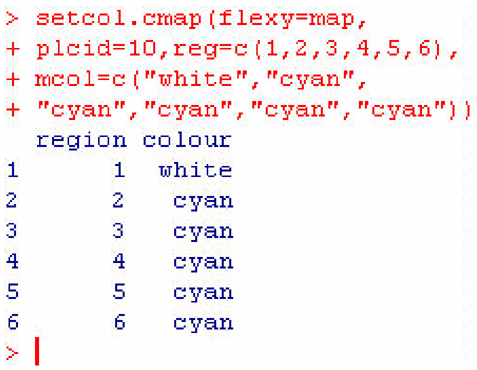
Figure 4.2: A map having a simple region and a complex region



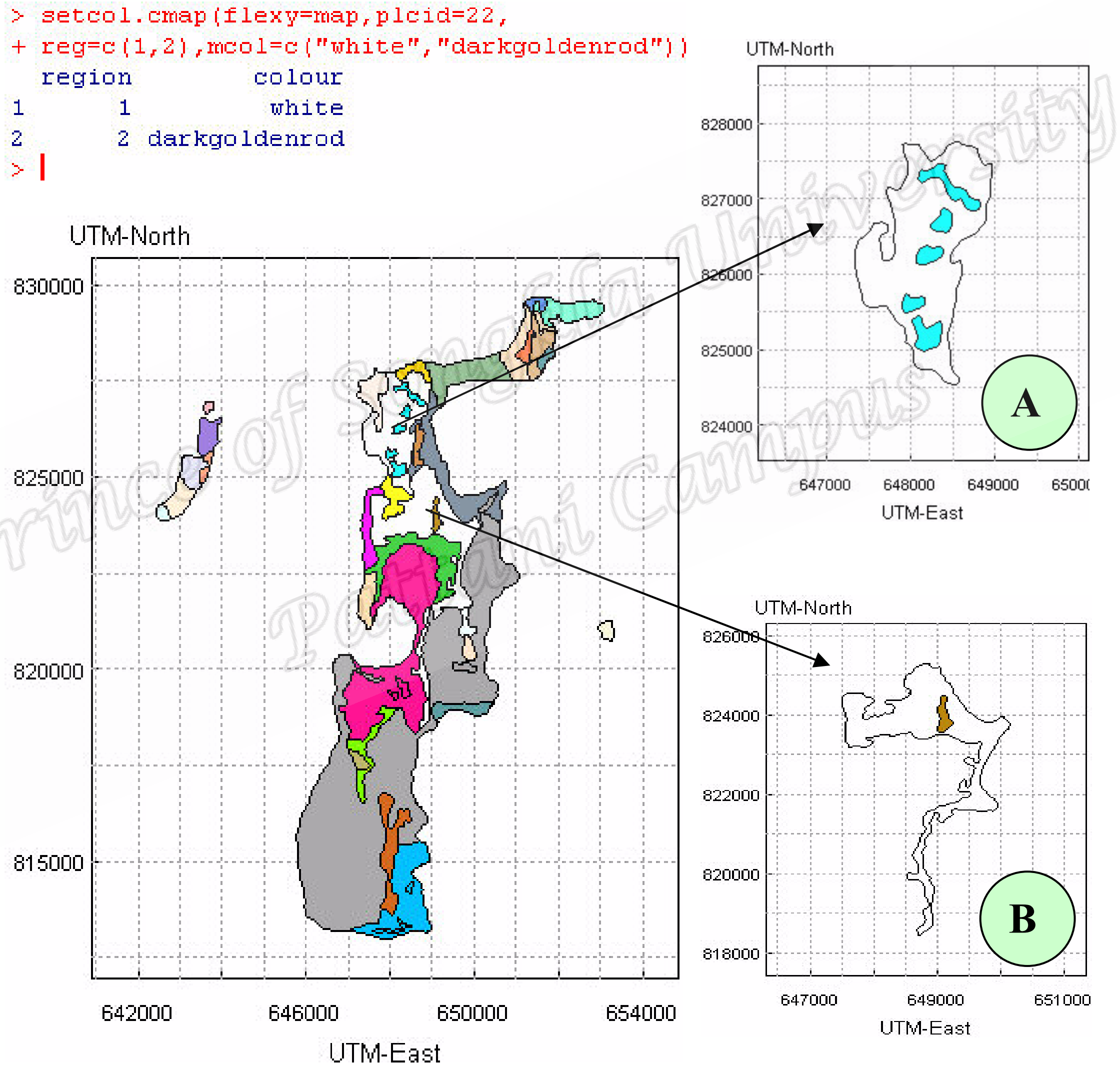
```
> setcol.map(flexy=map)
[1] simple regions
   plcid           colour
1      1       aquamarine
2      2   cornflowerblue
3      3           bisque
4      4        burlywood
5      5            coral
6      6        cadetblue
```

```
[1] complex regions
   plcid          colour
1     10            cyan
2     22  darkgoldenrod
3     26        darkgray
4     30        deeppink
5     42     deepskyblue
> |
```

Figure 4.3: A map with different color specified for each region

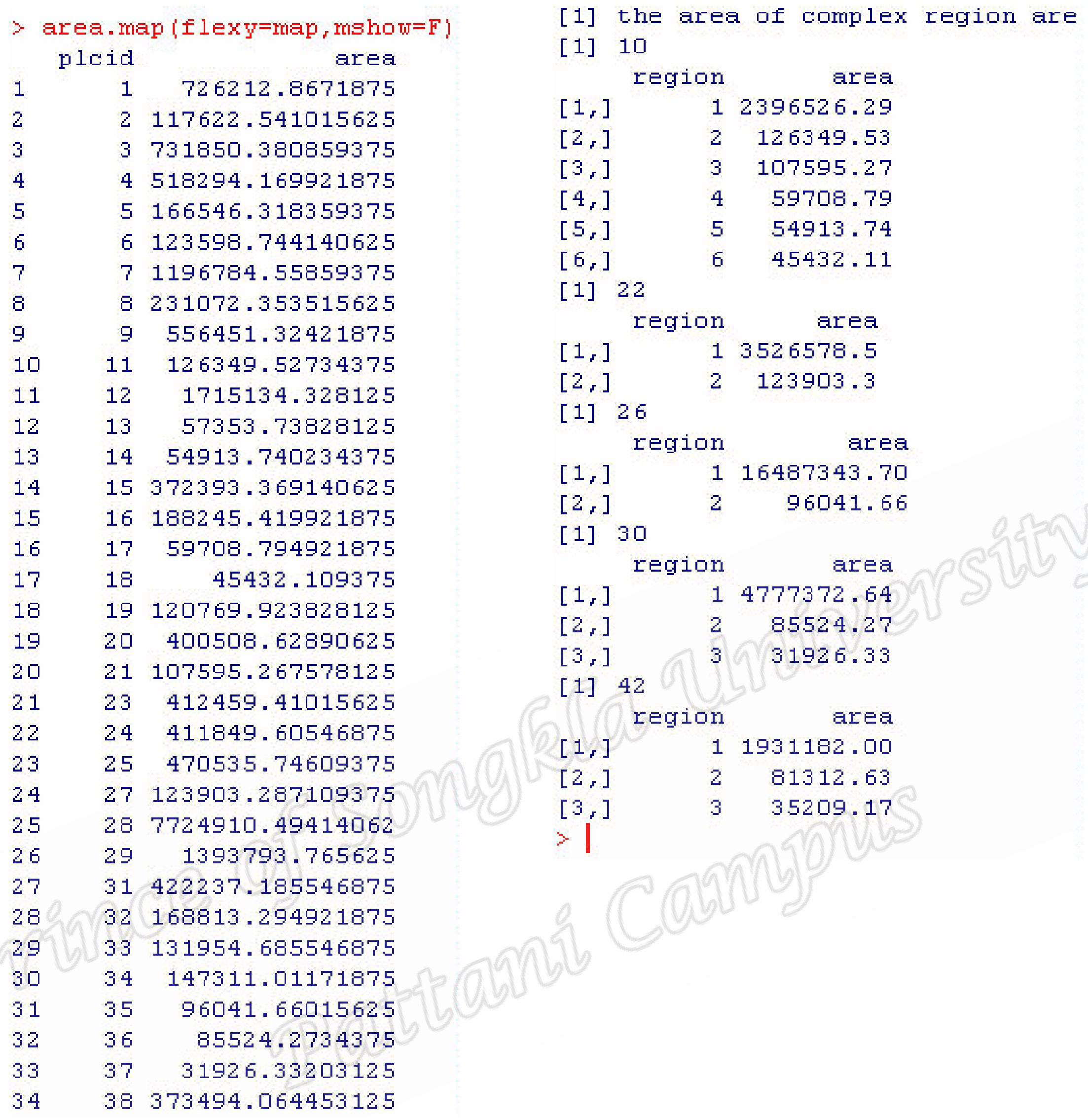


```
> setcol.cmap(flexy=map,
+ plcid=10,reg=c(1,2,3,4,5,6),
+ mcol=c("white","cyan",
+ "cyan","cyan","cyan","cyan"))
  region colour
1      1  white
2      2   cyan
3      3   cyan
4      4   cyan
5      5   cyan
6      6   cyan
> 
```



```
> setcol.cmap(flexy=map,plcid=22,
+ reg=c(1,2),mcol=c("white","darkgoldenrod"))
  region        colour
1      1         white
2      2 darkgoldenrod
> 
```

Figure 4.4: Specified color for a complex region



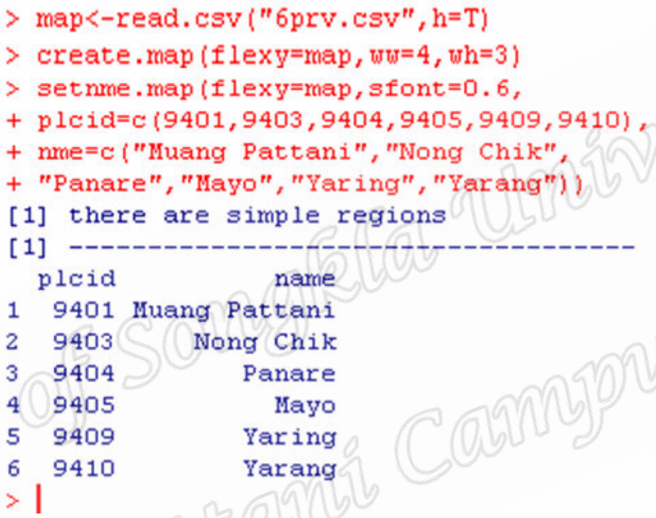
```
> area.map(flexy=map,mshow=F)
   plcid             area
1      1    726212.8671875
2      2  117622.541015625
3      3  731850.380859375
4      4  518294.169921875
5      5  166546.318359375
6      6  123598.744140625
7      7  1196784.55859375
8      8  231072.353515625
9      9   556451.32421875
10    11   126349.52734375
11    12    1715134.328125
12    13    57353.73828125
13    14   54913.740234375
14    15  372393.369140625
15    16  188245.419921875
16    17   59708.794921875
17    18      45432.109375
18    19  120769.923828125
19    20   400508.62890625
20    21  107595.267578125
21    23   412459.41015625
22    24   411849.60546875
23    25   470535.74609375
24    27  123903.287109375
25    28  7724910.49414062
26    29    1393793.765625
27    31  422237.185546875
28    32  168813.294921875
29    33  131954.685546875
30    34   147311.01171875
31    35    96041.66015625
32    36     85524.2734375
33    37    31926.33203125
34    38  373494.064453125
```

```
[1] the area of complex region are
[1] 10
     region       area
[1,]      1 2396526.29
[2,]      2  126349.53
[3,]      3  107595.27
[4,]      4   59708.79
[5,]      5   54913.74
[6,]      6   45432.11
[1] 22
     region      area
[1,]      1 3526578.5
[2,]      2  123903.3
[1] 26
     region        area
[1,]      1 16487343.70
[2,]      2    96041.66
[1] 30
     region       area
[1,]      1 4777372.64
[2,]      2   85524.27
[3,]      3   31926.33
[1] 42
     region       area
[1,]      1 1931182.00
[2,]      2   81312.63
[3,]      3   35209.17
> |
```

Figure 4.5: The results from *area.map*() function

4.3 Suggestion

We can compare longitude and latitude system with the Cartesian coordinate system when using *x*-, *y*- coordinates. For example, figure 4.6 showssix districts in Pattani province which are Mueang Pattani, Nong Chik, Yaring, Yarang, Panare and Mayo. A map is showthat is from a Cartesian coordinate system and a map is shown in the figure 4.7 which is from a longitude and latitude system. Clearly some map creation functions can work with both systems, but when we computed the area using *area.map*() function, the areas of the same place are different for the two systems, as shown in the example in figure 4.8. This study found thatt he longitude and latitude system can use the functions *create.map*(), *setcol.map*(), *setcol.cmap*(), *setnme.map*(), *combine.map*(), colstat.*map*() and *piestat.map*() but it does not work when using *area.map*(), *perimeter.map*() and *center.map*().



```
> map<-read.csv("6prv.csv",h=T)
> create.map(flexy=map,ww=4,wh=3)
> setnme.map(flexy=map,sfont=0.6,
+ plcid=c(9401,9403,9404,9405,9409,9410),
+ nme=c("Muang Pattani","Nong Chik",
+ "Panare","Mayo","Yaring","Yarang"))
[1] there are simple regions
[1] ------------------------------------
  plcid          name
1  9401 Muang Pattani
2  9403     Nong Chik
3  9404        Panare
4  9405          Mayo
5  9409        Yaring
6  9410        Yarang
> 
```

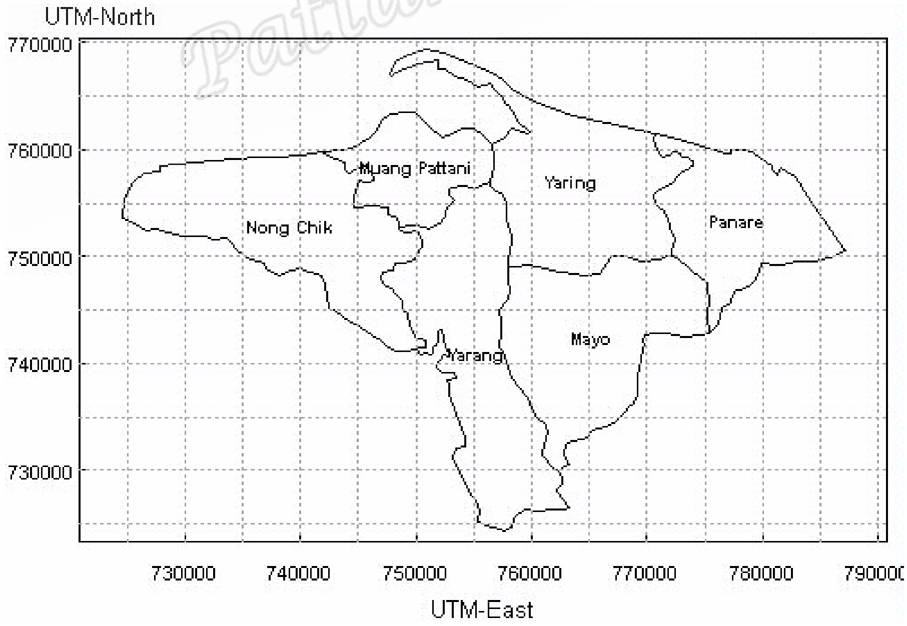


Figure 4.6: A map is from a Cartesian coordinate system

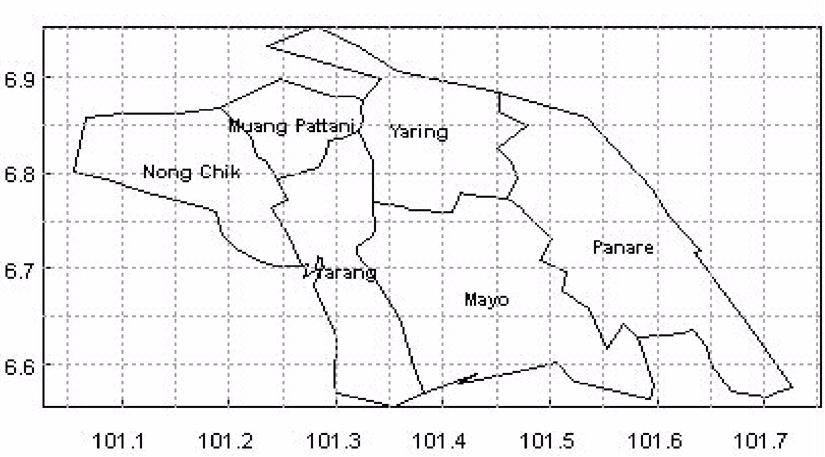
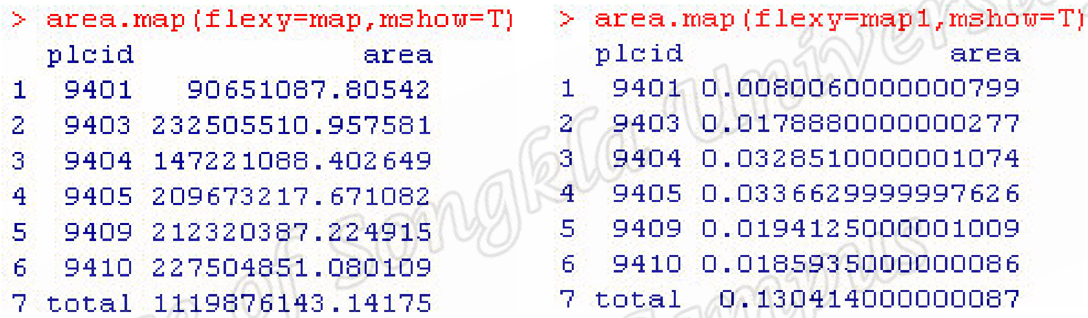


Figure 4.7: A map is from a longitude and latitude system



```
> area.map(flexy=map,mshow=T)
  plcid              area
1  9401    90651087.80542
2  9403  232505510.957581
3  9404  147221088.402649
4  9405  209673217.671082
5  9409  212320387.224915
6  9410  227504851.080109
7 total  1119876143.14175
```

```
> area.map(flexy=map1,mshow=T)
  plcid                area
1  9401 0.0080060000000799
2  9403 0.0178880000000277
3  9404 0.0328510000001074
4  9405 0.0336629999997626
5  9409 0.0194125000001009
6  9410 0.0185935000000086
7 total   0.130414000000087
```

Figure 4.8: The result from *area.map*() function on R Console

4.4 Ongoing work

Ongoing work is to develop a package in R of the 10 functions, then place it on the R website for others to use. Suggestions will be invited, for improvement and further development of this package. The next step will be to develop other functions such as zoom the map, count the neighbors of polygon, create the street, shows the river, make the contour, etc. We hope to develop the Graphic User Interface (GUI) in R, to make it easier to use the functions.