AptaKan: Analysis of the data of fluorescent bioassays

Navneet Phogat and Matthias Kohl
Institute of Precision Medicine
Hochschule Furtwangen University, Germany

June 29, 2017

Contents

1 Introduction

The package "AptaKan" has the functions as well as in - built shinyapps based Graphical User Interfaces (GUIs). The package "AptaKan" divided in three parts on the basis of the functions: First, Reading the fluorescent data and Combining the replicates. Second, Computation of confidence intervals, Analysis of data on the basis of different statistical models and Computation of the concentration. Third, Computation of the dissociation constant.

User may load the "AptaKan" package.

```
> library(AptaKan)
```

2 S4 Class aptakandt

The S4 Class `aptakandt` is defined to contain the data to control the flow of the data in package "AptaKan". All the slots of Class `aptakandt` contains the data in the form of an object of Class `data.frame`. The Class `aptakandt` has six slots, which includes `initialData` to contain the initial data of fluorescence except concentration, `crepData` to contain the results of the function `CombineReps`, `concentration` to contain the concentration, `modelData` to contain the results of implementation of different statistical models of analysis, `confData` to contain the results of confidence intervals and `KdData` to contain the results of computation of the dissociation constant.

3 Reading the fluorescent data

The `ReadAptakan` function is based on the functions `read.csv` and `read.table` of the package "`utils`". The function `ReadAptakan` reads in .csv as well as .txt (Tab delimited) files of the fluorescent data, generated through the fluorescent bioaasays. The format of both types of files is provided in the folder exData, which is present in the folder inst of the package "AptaKan": Aptakandata.csv, Aptakandata.txt, aptakdnew.csv and aptakdnew.txt. The file aptakdnew.csv and aptakdnew.txt are to be implemented, while computing the dissociation constant (Kd), which requires the data till saturation point. The final data will be stored in the new S4 Class aptakandt. The following is the example to read in the .csv file:

```
> #library(AptaKan) # load the Aptakan package
> ##To read the .csv file
> file.csv <- system.file("exData", "Aptakandata.csv", package = "AptaKan")
> read.datacsv <- ReadAptakan(file.csv)
> ## To express all the data
> read.datacsv
> ## To express the initial data (all the data read in, except concentration)
> slot(read.datacsv,"initialData")
> ## To express the concentration data (from the data read in)
> slot(read.datacsv,"concentration")
```

The following is the example to read in the .txt file:

```
> ## To read the .txt file
> file.txt <- system.file("exData", "Aptakandata.txt", package = "AptaKan")
> read.datatxt <- ReadAptakan(file.txt, type = ".txt")
> read.datatxt ## to express all the data
> ## To express the initial data (except concentration)
> slot(read.datatxt, "initialData")
> ## To express the concentration data
> slot(read.datatxt, "concentration")
```

4 Combine the replicates

The function CombineReps combine the technical replicates (on the basis of mean or median) as well as compute the standard deviation of an object of the Class aptakandt, which are produced as an output of the function ReadAptakan. The examples are shown for the data read in the .csv file format. But, in a similar way, can also be implemented on the data, read in from the .txt file. Following is the example to combine the technical replicates on the basis of mean:

```
> ###2. To combine the technical replicates, on the basis of mean
> cr.mean <- CombineReps(read.datacsv)
> cr.mean ## To express all the data

> slot(cr.mean,"crepData")  ##To express the combined replicates

          CA        CB        CC        CD        CE        CF        CG
R1  630.0000       NaN       NaN 1035.5000 1214.5000 1317.5000       NaN
```

```
R2   442.0000      NaN      NaN  449.7500  630.5000  654.2500  471.2500
R3   411.0000      NaN      NaN  523.0000  554.5000  645.5000  855.0000
R4  1249.0000      NaN      NaN  908.0000  849.2500 1859.7500 1986.0000
R5   474.7500      NaN      NaN  926.2500  821.0000  939.7500 1491.5000
R6  1019.0000      NaN      NaN  831.6667  744.6667 1357.3333 1525.0000
R7   521.3333      NaN      NaN  561.0000  780.0000  921.0000 1157.6667
R8   387.3333      NaN      NaN  498.6667  479.6667  885.3333  877.3333
R9   420.6667      NaN      NaN  950.6667  729.6667 1204.0000 2037.6667
R10  771.0000 924.0000 834.6667  823.0000  981.6667 1222.3333 1789.0000
R11  617.3333 564.6667      NaN 1241.3333       NaN 1241.3333       NaN
           CH       CI       CJ   CK
R1  2091.5000      NaN 3309.750  NaN
R2   967.5000  921.750 1956.000  NaN
R3   930.5000 1142.750 2713.000  NaN
R4  1454.2500 1878.000 2467.500  NaN
R5  1795.0000 2166.750 2793.250  NaN
R6  1229.6667 2165.000 2182.000  NaN
R7   966.6667 1475.667 1973.667  NaN
R8  1096.3333 1164.333 1572.333  NaN
R9  1555.6667      NaN 2559.667 2805
R10       NaN      NaN      NaN  NaN
R11       NaN      NaN      NaN  NaN
```

Following is the example to combine the technical replicates on the basis of the median:

```
> ##To combine the replicates on the basis of median
> cr.median <- CombineReps(read.datacsv, calc = "Median")
> cr.median ##To express the overall result
> slot(cr.median,"crepData") ##Toexpress the combined replicates
```

Following is the example to compute the standard deviation within the replicates:

```
> ##To compute the standard deviation
> cr.sd <- CombineReps(read.datacsv, calc = "sd")
> cr.sd
```

```
> ## To visualise the standard deviation
> slot(cr.sd,"crepData")
```

```
             CA        CB       CC        CD        CE         CF         CG
R1    31.864296        NA       NA 702.96159 251.43654  389.71229         NA
R2   196.073116        NA       NA 187.53733 435.01456  505.20912   46.25563
R3     1.632993        NA       NA  24.42676  19.12241   26.28688   17.56891
```

```
R4  1636.805425        NA       NA 493.44503 224.52821 1928.30433 1663.35344
R5    63.646812        NA       NA 402.32936 164.33097  316.58951  427.03044
R6   885.961060        NA       NA  86.89265 177.84919  632.80434  721.46171
R7    98.591751        NA       NA  16.52271  74.10128   73.32803  287.38882
R8    51.733290        NA       NA  72.70030  77.02164  383.28623  163.23705
R9    16.502525        NA       NA 816.11049  61.00273  178.96648  992.33328
R10  247.774091 264.96037 140.4184 131.57127 130.69940   70.43673   77.78817
R11   64.856251  37.52777       NA  26.57693        NA   26.57693         NA
           CH        CI        CJ       CK
R1  251.16860        NA 481.06852       NA
R2  515.16308  68.69437 241.22327       NA
R3   41.23510 103.47101  24.37212       NA
R4  170.67782 601.68153  87.79711       NA
R5  522.59034 337.64614  49.90908       NA
R6  140.59279 388.26280 151.33737       NA
R7   56.69509 151.01766 142.57045       NA
R8  153.15787 227.51117 179.52530       NA
R9  155.87281        NA 580.08304 58.10336
R10        NA        NA        NA       NA
R11        NA        NA        NA       NA
```

5 Compute the confidence intervals

The function `apta.conf` computes the confidence intervals of the combined replicates. The function `apta.conf` can be implemented on an object of Class `aptakandt`, which is produced as an output from the function `CombineReps`. Following is the example to compute the confidence interval by implementing the function `apta.conf`:

```
> ###3. 95% Confidence intervals
> conf.mean <- apta.conf(cr.mean)
> conf.mean ##To visualise all the data

> ## To express the confidence intervals
> slot(conf.mean,"confData")
```

```
  min.value max.value
1  812.1544  2387.429
2  454.3513  1168.899
3  445.9460  1497.866
4 1182.3185  1980.619
5  870.8205  1981.242
```

```
6   990.9884  1772.595
7   700.1992  1389.051
8   582.9839  1157.349
9   920.0820  2145.668
10  778.2105  1320.551
11  540.0814  1292.252
```

6 To implement the different statistical models to analyse the data

The function `aptamodelall` allows the user to analyse the data on the basis of different models, including linear model, log linear model, robust linear model, log robust linear model and generalised linear model. It provides the different values of r-square, adj.rsquare, intercept and slope in case of linear, log linear, robust linear and log robust linear models as well as the values of AIC, intercept and slope in case of generalised linear model. The function `aptamodelall` is based on the function `aptamodel` and can be implemented on an object of class `aptakandt`, which is produced as an output from the function `CombineReps`. The function `aptamodel` is described in the section auxilliary functions. The examples are shown to implement the function `aptamodelall` on combined replicates (combined on the basis of mean), but, in similar way, the function `aptamodelall` can also be implemented on the combined replicates, which are combined on the basis of meadian. Following is the example to analyse the data on the basis of linear (lm) model:

```
> ##To analyse the data on the basis of linear model
> lm.result.all <- aptamodelall(cr.mean)
> ##To express all the resulting data
> lm.result.all

> ##To express the modelData, which contains the final result
> slot(lm.result.all,"modelData")

    rsquare adj.rsquare intercept    slope
1  0.9935984   0.9919980  630.7055 5.408853
2  0.8037950   0.7710942  242.6149 2.801280
3  0.8412174   0.8147536  114.9606 4.218809
4  0.6468693   0.5880142 1011.1851 2.807550
5  0.9742233   0.9699272  452.5371 4.792586
6  0.7904061   0.7554738  764.5864 3.038549
7  0.9493631   0.9409236  448.4715 2.934910
8  0.9469877   0.9381523  373.7152 2.444069
9  0.7556901   0.7149718  875.5950 2.311314
```

```
10 0.8062091   0.7674509  657.7745 4.568743
11 0.7441172   0.6161758  609.6190 4.904762
```

Following is the example to analyse the data on the basis of the log linear model:

```
> ## To analyse the data on the basis of log linear model
> loglm.result.all <- aptamodelall(cr.mean, method = "loglm")
> ## To express all the resulting data
> loglm.result.all

> ## To express the modelData, which contains the resulting data
> slot(loglm.result.all,"modelData")
```

```
     rsquare adj.rsquare intercept       slope
1  0.9263121   0.9078902  6.679548 0.003113411
2  0.8349929   0.8074918  5.986563 0.002875405
3  0.9740831   0.9697636  5.966803 0.003609641
4  0.5663320   0.4940540  6.941774 0.001787535
5  0.9035840   0.8875146  6.422273 0.003438848
6  0.7273003   0.6818503  6.731163 0.002116515
7  0.9199034   0.9065539  6.313003 0.002694998
8  0.8750171   0.8541866  6.082789 0.002870646
9  0.5994701   0.5327151  6.725767 0.001553790
10 0.8440551   0.8128662  6.584979 0.003847866
11 0.7339350   0.6009025  6.403824 0.005576932
```

Following is the example to analyse the data on the basis of the robust linear model:

```
> ## To analyse the data on the basis of robust linear model
> roblm.result.all <- aptamodelall(cr.mean, method = "roblm")
> ## To express all the resulting data
> roblm.result.all

> ##To express the modelData, which contains the resulting data
> slot(roblm.result.all,"modelData")
```

```
     rsquare adj.rsquare intercept    slope
1  0.9930504   0.9913129  632.9277 5.401507
2  0.7756117   0.7382136  261.7501 2.741670
3  0.9766436   0.9727509  368.7501 2.205993
4  0.6107409   0.5458643 1011.8725 2.798277
5  0.9726742   0.9681199  461.8260 4.774248
6  0.7603207   0.7203742  768.4279 3.021407
```

```
7  0.9571696   0.9500312  458.4045 2.953287
8  0.9410876   0.9312689  373.6865 2.439215
9  0.8807684   0.8608964  538.1951 4.242416
10 0.7649131   0.7178957  664.7818 4.436413
11 0.7129235   0.5693853  608.0447 4.883622
```

Following is the example to analyse the data on the basis of the log robust linear model:

```
> ## To analyse the data on the basis of log robust linear model
> logroblm.result.all <- aptamodelall(cr.mean, method = "logroblm")
> ## To express all the resulting data
> logroblm.result.all

> ## To express the modelData, which contains the resulting data
> slot(logroblm.result.all,"modelData")
```

```
    rsquare adj.rsquare intercept       slope
1  0.9146573   0.8933216  6.692205 0.003067456
2  0.8381961   0.8112288  6.003890 0.002876940
3  0.9759675   0.9719620  6.034086 0.003055241
4  0.5286079   0.4500425  6.947213 0.001766382
5  0.8862764   0.8673224  6.425123 0.003430724
6  0.6915022   0.6400859  6.738317 0.002094914
7  0.9092118   0.8940804  6.314081 0.002686879
8  0.8584628   0.8348732  6.078295 0.002876430
9  0.5589306   0.4854190  6.740606 0.001523788
10 0.8201326   0.7841592  6.585532 0.003834783
11 0.7012505   0.5518758  6.402648 0.005550777
```

Following is the example to analyse the data on the basis of the glm model:

```
> ## To analyse the data on the basis of generalised linear model
> glm.result.all <- aptamodelall(cr.mean, method = "glm", result = "glm")
> ## To express all the resulting data
> glm.result.all

> ## To express the modelData, which contains the resulting data
> slot(glm.result.all,"modelData")
```

```
        AIC intercept       slope
1   86.38816  6.690912 0.003105469
2  107.05517  6.001303 0.002888937
3   97.03284  5.965516 0.003635055
```

```
4  122.46801  6.988348 0.001693923
5  114.21087  6.437768 0.003432721
6  117.05683  6.756887 0.002082112
7  104.63007  6.322500 0.002683977
8  106.64621  6.098345 0.002861765
9  128.01228  6.811377 0.001504048
10  91.53906  6.590341 0.003853456
11  58.34376  6.420716 0.005612819
```

7 To generate the plots of the analysis of the data

The function `aptakanplotall` can be implemented togenerate the plots of the analysis of the data, on the basis of the different statistical models, including linear model, log linear model, robust linear model and log linear robust model. The function `aptakanplotall` is based on the function `aptakanplot`, which is an auxilliary function to the function `aptakanplotall`. The function `aptakanplotall` can generate all the plots, on the basis of one model, of all the combined replicates in a single step. The plots will be generated in pdf format. Following is the code to generate all the plots, using the function `aptakanplotall`.

```
> ## Plots on the basis of the linear model
> aptakanplotall(cr.mean)
> ## Plots on the basis of log linear model
> aptakanplotall(cr.mean, method = "loglm")
> ## Plots on the basis of robust linear model
> aptakanplotall(cr.mean, method = "roblm")
> ## Plots on the basis of log robust linear model
> aptakanplotall(cr.mean, method = "logroblm")
```

8 To plot the diagnostic plots on the basis of linear model and log linear model, using ggplot2

The function `apta.dplotall` can be implemented to plot the diagnostic plots of analysis of data, based on the different statistical method, including, linear model and loglinear model. The function `apta.dplotall` is based on the function `apta.dplot`. The function `apta.dplotall` acts on the object of Class `aptakandt`, produced as a result after implementing the function `CombineReps`. Following is the example to plot the diagnostic plots:

```
> ##To get all the first plot
> apta.dplotall(cr.mean)  ##linear model
> apta.dplotall(cr.mean, method = "loglm")
> ## To get all the second plot
> apta.dplotall(cr.mean, plottype = "QQ")
> apta.dplotall(cr.mean, method = "loglm", plottype = "QQ")
> ## To get all the third plot
> apta.dplotall(cr.mean, plottype = "SL")
> apta.dplotall(cr.mean, method = "loglm", plottype = "SL")
> ## To get all the fourth plot
> apta.dplotall(cr.mean, plottype = "Cook")
> apta.dplotall(cr.mean, method = "loglm", plottype = "Cook")
> ## To get all the fifth plot
> apta.dplotall(cr.mean, plottype = "RL")
> apta.dplotall(cr.mean, method = "loglm", plottype = "RL")
> ## To get all the sixth plot
> apta.dplotall(cr.mean, plottype = "CL")
> apta.dplotall(cr.mean, method = "loglm", plottype = "CL")
```

9 To plot the diagnostic plots on the basis of different statistical models, using plot

The function apta.diagplot can be implemented to plot the diagnostic plots of the analysis of data, based on different statistical methods, including, linear model, log linear model, robust linear model, robust log linear model and generalised linear model. The function apta.diagplot can be implemented on the object of Class aptakandt, produced as a result of implementation of the function CombineReps. Following is the example to implement the function apta.diagplot:

```
> ## To plot on the basis of linear model
> apta.diagplot(cr.mean, which = 1:6)
> ## To plot on the basis of log linear model
> apta.diagplot(cr.mean, method = "loglm", which = 1:6)
> ## To plot on the basis of robust linear model
> apta.diagplot(cr.mean, method = "roblm", which = 1:5)
> ## To plot on the basis of log robust linear model
> apta.diagplot(cr.mean, method = "logroblm", which = 1:5)
> ## To plot on the basis of generalised linear model
> apta.diagplot(cr.mean, method = "glm", which = 1:5)
```

One can define the argument rowno, on the basis of the row no. of the combined replicates, which one wants to plot. The default is 1, that is the first row will be considered.

For example, if I want to plot for the 2nd row of the combined replicates, then rowno will be equal to 2. Following is the example to clarify the concept of rowno:

```
> ## To plot on the basis of linear model for row no. 2
> apta.diagplot(cr.mean, rowno =2, which= 1:6)
> ## To plot on the basis of the log linear model for row no. 3
> apta.diagplot(cr.mean, rowno = 3, method = "loglm", which = 1:6)
> ##To plot on the basis of robust linear model for row no. 4
> apta.diagplot(cr.mean, rowno = 4, method = "roblm", which = 1:5)
> ## To plot on the basis of log robust linear model for row no. 4
> apta.diagplot(cr.mean, rowno = 4, method = "logroblm", which = 1:5)
> ## To plot on the basis of glm model for row no. 2
> apta.diagplot(cr.mean, rowno =2, method = "glm", which = 1:5)
```

10 Compute the concentration from the fluorescence value

The function `aptaconc` computes the concentration of the combined replicates which are produced as a result after implementing the function `aptamodelall`. One needs to provide the fluorescence for which the concentration is to be calculated. Following is the example to compute the concentration at fluorescence value of 1000, using the lm model:

```
> apta.conc.lm <- aptaconc(lm.result.all,fluo = 1000)
> ## To express the result
> apta.conc.lm
```

```
[1] 68.27593
```

Following is the example to compute the concentration at fluorescence value of 1000, using the lmrob model:

```
> apta.conc.lmrob <- aptaconc(roblm.result.all,fluo = 1000)
> # To express the result
> apta.conc.lmrob
```

```
[1] 67.95738
```

Following is the example to compute the concentration at the fluorescence value of 1000, using the log linear model:

```
> apta.conc.loglm <- aptaconc(loglm.result.all,fluo = 1000, model = "loglm")
> ## To express the result
> apta.conc.loglm
```

```
[1] 260.6774
```

Following is the example to compute the concentration at the fluorescence value of 1000, using the log robust linear model:

```
> apta.conc.loglmrob <- aptaconc(logroblm.result.all,fluo = 1000, model = "loglm")
> ## To express the result
> apta.conc.loglmrob
```

```
[1] 285.9576
```

11 Compute the Dissociation constant (Kd)

The function apta.Kdall computes the dissociation constant of the combined replicates, which are produced as a result after implementing the function CombineReps. The function also plots the fitting curve of dissociation constant. The function apta.Kdall is based on the function apta.Kd. The description of the function apta.Kd is provided in the section auxilliary functions. The format of the files .txt Tab delimited and .csv are provided in the folder exData of folder inst in the package "AptaKan". Generally, the dissociation constant is computed by two different methods: 1. Sigmoidal model, which includes the suqare of the substrate concentration (Fluorescence vs square of substrate concentration) and 2. Non-sigmoidal Model, which includes the Fluorescence vs substrate concentration. The function computes the dissociation constant by two different methods (sigmoidal model and non-sigmoidal model) and correlation between the original and predicted values as well as plots the fitting curve of dissociation constant. The following is the example to compute the dissociation constant and correlation and plot the simulation (fitting) plots, on the basis of the sigmoidal model and non-sigmoidal model:

```
> ##To read the .csv file
> file.kd <- system.file("exData", "aptakdnew.csv", package = "AptaKan")
> read.datacsv <- ReadAptakan(file.kd)
> ## To combine the replicates
> kd.rep.mean <- CombineReps(read.datacsv)
> ## To express the combined replicates
> kd.rep.mean
```

To compute the dissociation constant by sigmoidal model:

```
> ## To compute the dissociation constant by sigmoidal model
> kd <- apta.Kdall(kd.rep.mean, kdstart = 1)
> ## To express all the resulting data
> kd
> ##To express the dissociation constant and correlation
> slot(kd,"KdData")

       Kd Correlation
1 286.112   0.9463834
2 286.112   0.9463834
3 286.112   0.9463834
4 286.112   0.9463834
```

To plot the fitting results of sigmoidal model and generate the pdf file of plots:

```
> ## To plot the fitting results
> apta.Kdall(kd.rep.mean, plot = "yes", kdstart = 1)
```

To compute the dissociation constant by non-sigmoidal model:

```
> ## To compute the dissociation constant by non-sigmoidal model
> kd.nonsig <- apta.Kdall(kd.rep.mean, method = "non-sig", kdstart = 1)
> ## To express all the resulting data
> kd.nonsig

> ## To visualise the dissociation constant and correlation
> slot(kd.nonsig,"KdData")

        Kd Correlation
1 288.0339   0.9418284
2 288.0339   0.9418284
3 288.0339   0.9418284
4 288.0339   0.9418284
```

To plot the fitting results of the non-sigmoidal model and generate the pdf file of plots:

```
> ## To plot the fitting results
> apta.Kdall(kd.rep.mean, plot = "yes", method = "non-sig", kdstart = 1)
```

12 Auxilliary Functions

12.1 aptamodel

The function `aptamodel` is an auxilliary function to the function `aptamodelall`. We recommned to implement the function `aptamodelall` directly. The function `aptamodel` can be implemeted for a single compbined replicate. While, the function `aptamodelall` can be implemented for all the combined replicates, whether the combined replicates are one or more to do the computation in a single step. Following is the example to implement the function `aptamodel`:

```
> ###aptamodel
> ##To analyse the data on the basis of linear model
> lm.result <- aptamodel(cr.mean)
> lm.result ##To express the result
> ## To analyse the data on the basis of log linear model
> loglm.result <- aptamodel(cr.mean, method = "loglm")
> loglm.result
> ## To analyse the data on the basis of robust linear model
> roblm.result <- aptamodel(cr.mean, method = "roblm")
> roblm.result
> ## To analyse the data on the basis of log robust linear model
> logroblm.result <- aptamodel(cr.mean, method = "logroblm")
> logroblm.result
> ## To analyse the data on the basis of generalised linear model
> glm.result <- aptamodel(cr.mean, method = "glm", result = "glm")
> glm.result
```

12.2 aptakanplot

The function `aptakanplot` is an auxilliary function to the function `aptakanplotall`. We recommend to implement the function `aptakanplotall`, instead of the function `aptakanplot`, because the function `aptakanplotall` can be implemented on one or more combined replicates to generate the results in a single step. The function `aptakanplot` also acts on the object of the Class `aptakandt`, produced as an output of the function `CombineReps`. Following code can be used to implement the function `aptakanplot`:

```
> ### To plot the analysis of data
> ## On the basis of the linear model
> lm.plot <- aptakanplot(cr.mean)
> lm.plot
> ## On the basis of the log linear model
> loglm.plot <- aptakanplot(cr.mean, method = "loglm", main = "lm(log(y)~x)")
> loglm.plot
> ##on the basis of robust linear model
> roblm.plot <- aptakanplot(cr.mean, method = "roblm", main = "lmrob(y~x)")
> roblm.plot
> ## On the basis of the log robust linear model
> logroblm.plot <- aptakanplot(cr.mean, method = "logroblm", main = "lmrob(log(y)~x)")
> logroblm.plot
```

12.3 apta.Kd

The function `apta.Kd` is an auxilliary function to the function `apta.Kdall`. The function `apta.Kd` can be directly implemented through the function `apta.Kdall`. Therefore, we recommend to implement the function `apta.Kdall`instead of the function `apta.Kd`, because the function `apta.Kdall` is designed to compute the results of all the combined replicates in a single step, whether the combined replicates are one or more. Following is the example to implement the function `apta.Kd`:

```
> ## To implement the apta.Kd
> file1 <- system.file("exData", "aptakdnew.csv", package = "AptaKan")
> read.datacsv <- ReadAptakan(file1)
> ## To combine the replicates on the basis of mean
> ## (replicates can also be combined on the basis of the median, but we
> ## recommend to combine the replicates on the basis of the mean, because that's
> ## the standard way to compute the dissociation constant)
> kd.rep1.mean <- CombineReps(read.datacsv)
> kd.rep1.mean ## To express all the data
> slot(kd.rep1.mean,"crepData")  ##To express the combined replicates
> kd.rep.median <- CombineReps(read.datacsv)
> kd.rep.median ## To express all the data
> slot(kd.rep.median,"crepData") ## To express the combined replicates
> ## To compute the dissociation constant by sigmoidal model
> kd <- apta.Kd(kd.rep1.mean, kdstart = 1)
> kd ## To express the results
> kd.median <- apta.Kd(kd.rep.median, kdstart = 1)
> kd.median ## To express the results
> ## To plot the fitting results of the sigmoidal model
> apta.Kd(kd.rep1.mean, plot = "yes", kdstart = 1)
> apta.Kd(kd.rep.median, plot = "yes", kdstart = 1)
> ## To compute the dissociation constant by non-sigmoidal model
> kd.nonsig <- apta.Kd(kd.rep1.mean, method = "non-sig", kdstart = 1)
> ## To express the final results
> kd.nonsig
> kd.nonsig.median <- apta.Kd(kd.rep.median, method = "non-sig", kdstart = 1)
> ## To express the final results
> kd.nonsig.median
> ## To plot the fitting results of the sigmoidal model
> apta.Kd(kd.rep.median, plot = "yes", method = "non-sig")
> apta.Kd(kd.rep.median, plot = "yes", method = "non-sig")
```

12.4 apta.dplot

The function `apta.dplot` is an auxilliary function to the function `apta.dplotall`.In a similar way, like other functions `aptamodelall` and `apta.Kdall`, we recommend to implement the function `apta.dplotall`, instead of the function `apta.dplot`. Following is the code to implement the function `apta.dplot`:

```
> ##To get first plot
> apta.dplot(cr.mean)  ##linear model
> apta.dplot(cr.mean, method = "loglm")
> ## To get second plot
> apta.dplot(cr.mean, plottype = "QQ")
> apta.dplot(cr.mean, method = "loglm", plottype = "QQ")
> ## To get third plot
> apta.dplot(cr.mean, plottype = "SL")
> apta.dplot(cr.mean, method = "loglm", plottype = "SL")
> ## To get fourth plot
> apta.dplot(cr.mean, plottype = "Cook")
> apta.dplot(cr.mean, method = "loglm", plottype = "Cook")
> ## To get fifth plot
> apta.dplot(cr.mean, plottype = "RL")
> apta.dplot(cr.mean, method = "loglm", plottype = "RL")
> ## To get sixth plot
> apta.dplot(cr.mean, plottype = "CL")
> apta.dplot(cr.mean, method = "loglm", plottype = "CL")
```

13 Graphical User Interfaces (GUIs)

Graphical user interfaces are designed, through shiny app,to implement the functions in a user friendly way and to generate the dynamic reports of the results. This section can be divided into two parts, which are as follows:

13.1 GUI for analysis, computation of confidence interval and concentration

This graphical user interface is designed for the purpose of the analysis of the data through different statistical models and to compute the confidence interval and concentration from the fluorescence. One can download the dynamic report in html format after implementation, which will be automatically saved in the folder analysis.gui in folder inst of the package aptakan. The graphical user interface can be launched in an easy way through the function `analysis.gui`. Following is the example to launch the graphical user interface:

```
> ## To launch the graphical user interface
> analysis.gui()
```

13.2 GUI for computation of dissociation constant and correlation

This graphical user interface is designed to implement the concepts of the computation of the dissociation constant and correlation between the original and predicted values. Here, also, the user can download the dynamic report in html format, which will be saved in the folder kd.gui in folder inst of the package aptakan. The graphical user interface can be launched in an easy way through the function `Kd.gui`. Following is the example to launch the grapphical user interface:

```
> ## To launch the graphical user interface
> Kd.gui()
```

14 Report generation of the plots

The .Rmd files are provided in the folder Plotreport of the package `"AptaKan"`. It has two .Rmd files, one is Modelplot.Rmd. Here, the example is shown with the file aptakdnew.csv, which is present in the folder exData of folder inst of package `"AptaKan"`. To generate the pdf report, one can use knit to pdf, while to generate the html report, one can use knit to html. The resulting examples are shown in the files Modelplot in two formats of pdf and html. The another file is Kdplot.Rmd, which is also present in the same folder Plotreport. This is designed to generate the plots of dissociation constant. The resulting examples are shown in the files, Kdplot in the formats of pdf and html.