ReadqPCR: Functions to load RT-qPCR data into R

James Perkins
University College London

Matthias Kohl
Furtwangen University

April 30, 2018

Contents

1 Introduction

The package "ReadqPCR" contains different functions for reading qPCR data into R.

As well as the functions to read in the data, "ReadqPCR" contains the qPCRBatch and CyclesSet class definition. The data output by these RT-qPCR systems is in the form of fluorescence values or quantification cycle (Cq values), which represents the number of cycles of amplification needed in order to detect the expression of a given gene from a sample.

"ReadqPCR" is designed to be complementary to another R package, "NormqPCR", which is intended for the normalisation of qPCR data. It must be installed before the other module.

We load the "ReadqPCR" package.

```
> library(ReadqPCR)
```

2 read.LC480

With function `read.LC480` raw fluoresence data of Roche LightCycler 480 may be read in. The result is saved in an object of class `CyclesSet`. At the moment the function works only with the data exported to a txt-file, but we plan to add support also for the xml-export.

```
> path <- system.file("exData", package = "ReadqPCR")
> LC480.example <- file.path(path, "LC480_Example.txt")
> cycData <- read.LC480(file = LC480.example)
> ## Fluorescence data
> head(exprs(cycData))
```

```
     A1    A2    A3    A4    A5    A6    A7    A8    A9   A10      A11   A12
1 15.58 16.93 27.32 16.60 16.88 27.93 15.37 17.27 27.26 15.91 41594.00 25.74
2 15.53 16.96 27.69 16.68 16.91 28.23 15.47 17.35 27.59 15.87    16.17 25.76
3 15.49 16.93 27.56 16.67 16.95 28.27 15.41 17.32 27.60 15.83    16.23 25.77
4 15.47 16.97 27.62 16.63 16.87 28.22 15.42 17.28 27.62 15.83    16.17 25.70
5 15.47 16.88 27.56 16.65 16.92 28.22 15.38 17.31 27.52 15.74    16.18 25.64
6 15.41 16.90 27.55 16.65 16.87 28.20 15.41 17.27 27.57 15.76    16.23 25.70
     B1    B2    B3    B4    B5    B6    B7       B8    B9      B10   B11
1 16.61 17.74 27.15 17.57 17.67 29.30 41322 41625.00 26.84    16.93 16.64
2 16.67 17.80 27.42 17.61 17.71 29.76 41322    17.19 27.58 41625.00 16.68
3 16.57 17.78 27.47 17.64 17.73 29.74 41322    17.16 27.58 41350.00 16.64
4 16.56 17.72 27.49 17.57 17.64 29.87 41442    17.22 27.61    17.14 16.57
5 16.52 17.68 27.44 17.59 17.62 29.78    17 41625.00 27.56 41472.00 16.57
6 16.59 17.69 27.50 17.54 17.62 29.90 41564    17.22 27.61    17.14 16.56
       B12       C1    C2    C3       C4    C5       C6    C7       C8       C9
1 41512.00 41322.00 16.60 27.13 41292.00 16.77    26.88 17.43    16.96    27.86
2    26.35    16.89 16.55 27.43    18.14 16.77    27.19 17.50 41442.00    28.19
3    26.21    16.86 16.54 27.46 41473.00 16.81 41574.00 17.50    16.99    28.22
4    26.27    16.81 16.51 27.38 41504.00 16.83    27.17 17.43 41564.00 41606.00
5    26.22    16.79 16.43 27.48 41443.00 16.87    27.16 17.47 41322.00    28.15
6    26.15    16.84 16.44 27.47 41626.00 16.86    27.14 17.51 41291.00    28.17
       C10   C11      C12       D1    D2    D3    D4    D5    D6    D7    D8
1 41443.00 16.49    25.92    17.22 16.80 27.25 17.66 17.18 27.41 17.42 17.22
2 41535.00 16.53    26.22    17.28 16.76 27.62 17.69 17.26 27.59 17.49 17.14
3    18.00 16.57    26.30 41564.00 16.72 27.51 17.71 17.26 27.55 17.40 17.26
4 41443.00 16.55    26.24 41595.00 16.76 27.61 17.62 17.30 27.72 17.50 17.27
5    18.00 16.54 41390.00 41534.00 16.72 27.45 17.73 17.27 27.66 17.49 17.27
6    17.98 16.49 41451.00 41291.00 16.71 27.47 17.64 17.29 27.69 17.45 17.20
     D9   D10   D11      D12    E1    E2    E3       E4    E5       E6       E7
1 27.28 17.47 16.63 41332.00 27.52 17.80 26.51    28.69 17.62    27.48    27.64
```

```
2 27.74 17.46 16.70    27.38 27.78 17.79 26.84    29.00 17.67    27.79    28.00
3 27.74 17.40 16.66    27.38 27.69 17.75 26.88    28.99 17.71    27.89    27.99
4 27.91 17.51 16.76    27.42 27.61 17.71 26.90 41423.00 17.72    27.93 41361.00
5 27.82 17.43 16.69    27.33 27.63 17.79 26.82 41393.00 17.74    27.95 41483.00
6 27.87 17.46 16.69    27.41 27.59 17.76 26.84 41362.00 17.77 41302.00    28.14
       E8       E9   E10   E11   E12       F1       F2    F3    F4       F5
1    16.97 41361.00 27.17 16.69 26.82 41605.00 41473.00 28.28 28.24 41442.00
2 41442.00    28.53 27.41 16.88 27.20    27.25    18.15 28.65 28.65 41472.00
3 41442.00    28.51 27.33 16.83 27.19    26.99 41535.00 28.61 28.64 41381.00
4 41534.00    28.54 27.37 16.85 27.19 41544.00    18.19 28.61 28.60    17.16
5 41411.00    28.52 27.33 16.81 27.19    26.99 41412.00 28.53 28.61    17.18
6 41350.00    28.58 27.41 16.80 27.18    27.00 41412.00 28.48 28.66    17.18
     F6    F7    F8       F9      F10   F11   F12       G1    G2    G3      G4
1 27.99 27.50 16.67    27.65 41422.00 16.68 26.61    27.91 44743 15189    27.7
2 28.29 27.79 16.74    27.95    28.39 16.83 26.97 41453.00 45108 17380 41392.0
3 28.25 27.84 16.74    27.98    28.35 16.78 26.95    27.99 46204 16285    28.0
4 28.30 27.88 16.84 41333.00    28.38 16.82 26.99    27.89 44743 15554 41453.0
5 28.33 27.94 16.92 41392.00    28.31 16.86 26.96    27.84 44743 15554 41422.0
6 28.28 27.95 16.82 41361.00    28.39 16.86 26.97    27.74 41456 12997 41514.0
     G5    G6       G7    G8    G9   G10   G11   G12    H1       H2    H3    H4
1 31929 22433    27.79 34121 29738 28.32 20668 32295 15.97 41410.00 45108 15158
2 33756 23894 41361.00 35582 31929 28.58 24685 35582 15.85    15.97 46204 17715
3 33025 22433 41422.00 34851 30834 28.46 23590 33756 15.76    15.88 46569 16984
4 32660 24259 41514.00 35217 31199 28.42 22859 34121 15.72    15.88 45474 15523
5 33025 21702 41392.00 33756 28642 28.38 20302 33025 15.75    15.84 44378 16619
6 32660 20972    27.97 32295 29738 28.41 21398 33756 15.75    15.82 42552 13697
       H5       H6    H7    H8    H9   H10   H11   H12
1    26.70 41391.00 14427 21002 24624 24624 41371 17746
2 41391.00    27.28 18810 24289 28277 26816 41554 21763
3    26.90    27.17 17349 22098 25720 24259 41312 19572
4    26.85    27.21 15158 21732 26816 25355 41432 19572
5    26.82    27.21 14427 21002 26451 23894 41312 17746
6    26.80    27.26 14793 22098 25720 23163 36312 15554

> ## Pheno data
> head(pData(cycData))

   Sample position Sample name Program number Segment number
A1              A1    Sample_1              2              3
A2              A2    Sample_2              2              3
A3              A3    Sample_2              2              3
```

```
A4              A4    Sample_3              2              3
A5              A5    Sample_4              2              3
A6              A6    Sample_4              2              3
```

```
> ## Feature data
> head(fData(cycData))
```

```
  Cycle number Acquisition time Acquisition temperature
1            1           691600                   71.79
2            2           770133                   71.64
3            3           848716                   71.64
4            4           927233                   71.67
5            5          1005816                   71.67
6            6          1084316                   71.67
```

The class `CyclesSet` is an extension of class `eSet`.

```
> cycData
```

```
CyclesSet (storageMode: lockedEnvironment)
assayData: 45 features, 96 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: A1 A2 ... H12 (96 total)
  varLabels: Sample position Sample name Program number Segment number
  varMetadata: labelDescription
featureData
  featureNames: 1 2 ... 45 (45 total)
  fvarLabels: Cycle number Acquisition time Acquisition temperature
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

```
> ## Information about class CyclesSet
> getClass("CyclesSet")
```

```
Class "CyclesSet" [package "ReadqPCR"]

Slots:

Name:           assayData          phenoData        featureData
```

```
Class:          AssayData AnnotatedDataFrame AnnotatedDataFrame

Name:      experimentData         annotation       protocolData
Class:              MIAxE          character AnnotatedDataFrame

Name:   .__classVersion__
Class:           Versions

Extends:
Class "eSet", directly
Class "VersionedBiobase", by class "eSet", distance 2
Class "Versioned", by class "eSet", distance 3
```

The additional sample information file can be read in with function `read.LC480SampleInfo` and is saved in an object of class `AnnotatedDataFrame`.

```
> LC480.SamInfo <- file.path(path, "LC480_Example_SampleInfo.txt")
> samInfo <- read.LC480SampleInfo(file = LC480.SamInfo)
> ## Additional sample information
> head(pData(samInfo))
```

```
  Sample position Sample name Replicate of Filter combination       Target name
1              A1    Sample_1                         465-510 negativ Kontrolle
2              A2    Sample_2                         465-510               goi
3              A3    Sample_2                         465-510                hk
4              A4    Sample_3                         465-510 negativ Kontrolle
5              A5    Sample_4                         465-510               goi
6              A6    Sample_4                         465-510                hk
  Sample Pref color Efficiency Combined sample and target type
1         $00FF8000          2                 Target Negative
2             clRed          2                  Target Unknown
3         $0030D700          2                     Ref Unknown
4         clFuchsia          2                 Target Negative
5            clGray          2                  Target Unknown
6         $0012D7FA          2                     Ref Unknown
```

For adding this additional information to the `CyclesSet` we provide a method for function `merge`.

```
> cycData1 <- merge(cycData, samInfo)
> ## Extended pheno data
> phenoData(cycData1)
```

```
An object of class 'AnnotatedDataFrame'
  sampleNames: 1 2 ... 96 (96 total)
  varLabels: Sample position Sample name ... Combined sample and target
    type (10 total)
  varMetadata: labelDescription
```

```
> head(pData(cycData1))
```

```
  Sample position Sample name Program number Segment number Replicate of
1              A1    Sample_1              2              3
2              A2    Sample_2              2              3
3              A3    Sample_2              2              3
4              A4    Sample_3              2              3
5              A5    Sample_4              2              3
6              A6    Sample_4              2              3
  Filter combination       Target name Sample Pref color Efficiency
1            465-510 negativ Kontrolle         $00FF8000          2
2            465-510               goi             clRed          2
3            465-510                hk         $0030D700          2
4            465-510 negativ Kontrolle         clFuchsia          2
5            465-510               goi            clGray          2
6            465-510                hk         $0012D7FA          2
  Combined sample and target type
1                 Target Negative
2                  Target Unknown
3                     Ref Unknown
4                 Target Negative
5                  Target Unknown
6                     Ref Unknown
```

```
> varMetadata(phenoData(cycData1))
```

```
                                              labelDescription
Sample position                                Sample position
Sample name                                        Sample name
Program number                                  Program number
Segment number                                  Segment number
Replicate of                                      Replicate of
Filter combination                           Filter combination
Target name                                         Target name
Sample Pref color                             Sample Pref color
Efficiency                                           Efficiency
Combined sample and target type Combined sample and target type
```

3 read.qPCR

`read.qPCR` allows the user to read in qPCR data and populate a `qPCRBatch` R object (see section `qPCRBatch`) using their own data matrix. The format of the data file should be tab delimited and have the following columns, the first two of which are optional (although they should either be provided together, or not at all):

Well Optional, this represents the position of the detector on a plate. This information, if given, will be used to check the plates are of the same size and will also be used in order to plot a representation of the card to look for spatial effects and other potential problems. Both Well number and Plate ID must be present to enable a plate to be plotted.

Plate Optional, this is an identifier for the plate on which an experiment was performed. It is not possible to have duplicate plate IDs with the same Well number. Neither is it possible to have Plate Ids without Well numbers. Both Well number and Plate ID must be present to enable a plate to be plotted.

Sample The sample being analysed. Each sample must contain the same detectors in order to combine and compare samples effectively and to form a valid expression set matrix.

Detector This is the identifier for the gene being investigated. The Detectors must be identical for each sample.

Cq This is the quantification cycle value for a given detector in the corresponding sample.

The generic function `read.qPCR` is called to read in the qPCR file. It is similar to the `read.affybatch` function of the `"affy"` package, in that it reads a file and automatically populates an R object, `qPCRBatch` described below. However it is different in that the file is user formatted. In addition, unlike `read.affybatch`, and also unlike the `read.taqman` function detailed below, only one file may be read in at a time.

If `Well` and `Plate ID` information are given, then these are used to populate the `exprs.well.order`, a new `assayData` slot introduced in the `qPCRBatch` object, as detailed below in section `qPCRBatch`.

So for the `qPCR.example.txt` file, in directory `exData` of this library, which contains `Well` and `Plate ID` information, as well as the mandatory `Sample`, `Detector` and `Cq` information, we can read in the data as follows.

```
> path <- system.file("exData", package = "ReadqPCR")
> qPCR.example <- file.path(path, "qPCR.example.txt")
> qPCRBatch.qPCR <- read.qPCR(qPCR.example)
```

qPCRBatch.qPCR will be a qPCRBatch object with an exprs and exprs.well.order, as well as a phenoData slot which gets automatically populated in the same way as when using read.affybatch. More detail is given in the qPCRBatch section below.

read.qPCR can deal with technical replicates. If the same detector and sample identifier occurs more than once, the suffix _TechReps.n is concatenated to the detector name, where n in $\{1, 2...N\}$ is the number of the replication in the total number of replicates, N, based on order of appearence in the qPCR data file. So for a qPCR file with 2 technical replicates and 8 detectors per replicate, with one replicate per plate, the detector names would be amended as follows:

```
> qPCR.example.techReps <- file.path(path, "qPCR.techReps.txt")
> qPCRBatch.qPCR.techReps <- read.qPCR(qPCR.example.techReps)
> rownames(exprs(qPCRBatch.qPCR.techReps))[1:8]

[1] "gene_aj_TechReps.1" "gene_aj_TechReps.2" "gene_al_TechReps.1"
[4] "gene_al_TechReps.2" "gene_ax_TechReps.1" "gene_ax_TechReps.2"
[7] "gene_bo_TechReps.1" "gene_bo_TechReps.2"
```

The reason for appending the suffix when technical replicates are encountered is in order to populate the exprs and exprs.well.order slots correctly and keep them to the assayData format. It also allows the decisions on how to deal with the analysis and combination of technical replicates to be controlled by the user, either using the "NormqPCR" package, or potentially some other function that takes assayData format R objects as input.

4 read.taqman

read.taqman allows the user to read in the data output by the Sequence Detection Systems (SDS) software which is the software used to analyse the Taqman Low Density Arrays. This data consists of the header section, which gives some general information about the experiment, run date etc., followed by the raw Cq values detected by the software, followed by summary data about the experiment. read.taqman is a generic function, and is called in a way similar to the read.affybatch function of the "affy" package.

```
> taqman.example <- file.path(path, "example.txt")
> qPCRBatch.taq <- read.taqman(taqman.example)
```

Currently the SDS software only allows up to 10 plates to be output onto one file. read.taqman allows any number of SDS output files to be combined to make a single qPCRBatch, as long as they have matching detector identifiers.

```
> path <- system.file("exData", package = "ReadqPCR")
> taqman.example <- file.path(path, "example.txt")
```

```
> taqman.example.second.file <- file.path(path, "example2.txt")
> qPCRBatch.taq.two.files <- read.taqman(taqman.example,
+                                taqman.example.second.file)
```

SDS output will not necessarily contain plate identifiers, in which case a numeric identifier will be generated, which will increment for each plate, depending on the order of the plates within the SDS files. This is important for filling the `exprs.well.order` slot of the `qPCRBatch`.

read.taqman can also deal with technical replicates. If the same detector and sample identifier occurs more than once, the suffix `_TechRep.n` will be concatenated to the detector name, where n in $\{1, 2...N\}$ is the number of the replication in the total number of replicates N, based on the order of occurence in the taqman data file. So for a taqman file with 4 technical replicates of 96 detectors per sample, with one sample per plate, the detector names would be amended as follows:

```
> taqman.example.tech.reps <- file.path(path, "exampleTechReps.txt")
> qPCRBatch.taq.tech.reps <- read.taqman(taqman.example.tech.reps)
> rownames(exprs(qPCRBatch.taq.tech.reps))[1:8]

[1] "ACE.Hs00174179_m1_TechReps.1"
[2] "ACE.Hs00174179_m1_TechReps.2"
[3] "ACE.Hs00174179_m1_TechReps.3"
[4] "ACE.Hs00174179_m1_TechReps.4"
[5] "AT1R.AGTR1..Hs00241341_m1_TechReps.1"
[6] "AT1R.AGTR1..Hs00241341_m1_TechReps.2"
[7] "AT1R.AGTR1..Hs00241341_m1_TechReps.3"
[8] "AT1R.AGTR1..Hs00241341_m1_TechReps.4"
```

As with read.qPCR, the motivation for appending the suffix when technical replicates are encountered is in order to populate the `exprs` and `exprs.well.order` slots correctly and keep them to the `assayData` format. Again it allows the decisions on how to deal with the analysis of technical replicates to be controlled by the user, either using the `"NormqPCR"` package, or otherwise.

5 qPCRBatch

`qPCRBatch` is an S4 class, designed to store information on the raw Cq values which represents the relative gene expression for a given sample, phenotypic information on the different samples which enable the user to compare expression accross different conditions or cell lines, and information on the spatial location of the different detectors used to measure Cq. This is achieved by making `qPCRBatch` an an extension of eSet, which means we

can recycle slots such as exprs and pData, and by introducing a new `assyData` slot. Here is an example of what a `qPCRBatch` looks like. note the similarity to `eSet`:

```
> qPCRBatch.taq
```

```
qPCRBatch (storageMode: lockedEnvironment)
assayData: 96 features, 8 samples
  element names: exprs, exprs.well.order
protocolData: none
phenoData
  sampleNames: fp1.day3.v fp2.day3.v ... fp.8.day.3.mia (8 total)
  varLabels: sample
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
Annotation:
```

pData will be filled automatically if no data is given, in a way analagous to read.affybatch:

```
> pData(qPCRBatch.taq)
```

```
               sample
fp1.day3.v          1
fp2.day3.v          2
fp5.day3.mia        3
fp6.day3.mia        4
fp.3.day.3.v        5
fp.4.day.3.v        6
fp.7.day.3.mia      7
fp.8.day.3.mia      8
```

In addition there is a new slot, `exprs.well.order` which extends the `assayData` slot used for `exprs()`. It has the same dimensions as `exprs` (as every instance of `assayData` must). The cells contain further details on the position on the arrays where the different meaurements were taken.

The data provided by this slot can be used in order to identify certain problems with arrays, perhaps due to spatial effects and other problems with the microfluidics technology that is used by many of these systems.

This is conceptually similar to the cdf file information being stored in the `AffyBatch` class, which contains information on the spatial layout of features on an affy chip. However it differs since it allows for different arrays within the same `affyBatch` object to have different layouts to each other. This information can be viewed using the `exprs.well.order()` function.

When using `read.taqman`, if the input file includes identifiers for the different arrays in the experiment, the identifiers will be of the format `<plate.id>-<plate.position>`. However if no names are given for the different plates, `"ReadqPCR"` will assign them a numeric identifier, which increments depending on the order of plates in the original file. When several input files are given, as in the case of SDS files, the order in which they are supplied as arguments to the `read.taqman` function will be mirrored in the order of the numeric identifiers for the different plates. However, to minimise confusion, we recommend the useR giving the plates their own unique identifiers where possible.

Without plate names:

```
> head(exprs.well.order(qPCRBatch.taq))
```

```
                    fp1.day3.v fp2.day3.v fp5.day3.mia fp6.day3.mia
Actb.Rn00667869_m1   "1-38"     "1-134"    "1-230"      "1-326"
Adipoq.Rn00595250_m1 "1-61"     "1-157"    "1-253"      "1-349"
Adrbk1.Rn00562822_m1 "1-66"     "1-162"    "1-258"      "1-354"
Agtrl1.Rn00580252_s1 "1-91"     "1-187"    "1-283"      "1-379"
Alpl.Rn00564931_m1   "1-10"     "1-106"    "1-202"      "1-298"
B2m.Rn00560865_m1    "1-37"     "1-133"    "1-229"      "1-325"
                    fp.3.day.3.v fp.4.day.3.v fp.7.day.3.mia fp.8.day.3.mia
Actb.Rn00667869_m1   "2-38"       "2-134"      "2-230"        "2-326"
Adipoq.Rn00595250_m1 "2-61"       "2-157"      "2-253"        "2-349"
Adrbk1.Rn00562822_m1 "2-66"       "2-162"      "2-258"        "2-354"
Agtrl1.Rn00580252_s1 "2-91"       "2-187"      "2-283"        "2-379"
Alpl.Rn00564931_m1   "2-10"       "2-106"      "2-202"        "2-298"
B2m.Rn00560865_m1    "2-37"       "2-133"      "2-229"        "2-325"
```

With plate names:

```
> taqman.example.plateNames <- file.path(path, "exampleWithPlateNames.txt")
> qPCRBatch.taq.plateNames <- read.taqman(taqman.example.plateNames)
> head(exprs.well.order(qPCRBatch.taq.plateNames))
```

```
                    gp.1.day.3.v gp.2.day.3.v gp.5.day.3.mia gp.6.day.3.mia
Actb.Rn00667869_m1   "PlateA-38"  "PlateA-134" "PlateA-230"   "PlateA-326"
Adipoq.Rn00595250_m1 "PlateA-61"  "PlateA-157" "PlateA-253"   "PlateA-349"
Adrbk1.Rn00562822_m1 "PlateA-66"  "PlateA-162" "PlateA-258"   "PlateA-354"
Agtrl1.Rn00580252_s1 "PlateA-91"  "PlateA-187" "PlateA-283"   "PlateA-379"
Alpl.Rn00564931_m1   "PlateA-10"  "PlateA-106" "PlateA-202"   "PlateA-298"
B2m.Rn00560865_m1    "PlateA-37"  "PlateA-133" "PlateA-229"   "PlateA-325"
                    gp.3.day.3.v gp.4.day.3.v gp.7.day.3.mia gp.8.day.3.mia
Actb.Rn00667869_m1   "PlateB-38"  "PlateB-134" "PlateB-230"   "PlateB-326"
```

```
Adipoq.Rn00595250_m1 "PlateB-61"  "PlateB-157" "PlateB-253"   "PlateB-349"
Adrbk1.Rn00562822_m1 "PlateB-66"  "PlateB-162" "PlateB-258"   "PlateB-354"
Agtrl1.Rn00580252_s1 "PlateB-91"  "PlateB-187" "PlateB-283"   "PlateB-379"
Alpl.Rn00564931_m1   "PlateB-10"  "PlateB-106" "PlateB-202"   "PlateB-298"
B2m.Rn00560865_m1    "PlateB-37"  "PlateB-133" "PlateB-229"   "PlateB-325"
```

In addition, a mixture of files with and without plate identifiers is possible.

```
> taqman.example <- file.path(path, "example.txt")
> taqman.example.plateNames <- file.path(path, "exampleWithPlateNames.txt")
> qPCRBatch.taq.mixedPlateNames <- read.taqman(taqman.example,
+                                     taqman.example.plateNames)
> head(exprs.well.order(qPCRBatch.taq.mixedPlateNames))
```

```
                    fp1.day3.v fp2.day3.v fp5.day3.mia fp6.day3.mia
Actb.Rn00667869_m1   "1-38"     "1-134"    "1-230"      "1-326"
Adipoq.Rn00595250_m1 "1-61"     "1-157"    "1-253"      "1-349"
Adrbk1.Rn00562822_m1 "1-66"     "1-162"    "1-258"      "1-354"
Agtrl1.Rn00580252_s1 "1-91"     "1-187"    "1-283"      "1-379"
Alpl.Rn00564931_m1   "1-10"     "1-106"    "1-202"      "1-298"
B2m.Rn00560865_m1    "1-37"     "1-133"    "1-229"      "1-325"
                    fp.3.day.3.v fp.4.day.3.v fp.7.day.3.mia fp.8.day.3.mia
Actb.Rn00667869_m1   "2-38"       "2-134"      "2-230"        "2-326"
Adipoq.Rn00595250_m1 "2-61"       "2-157"      "2-253"        "2-349"
Adrbk1.Rn00562822_m1 "2-66"       "2-162"      "2-258"        "2-354"
Agtrl1.Rn00580252_s1 "2-91"       "2-187"      "2-283"        "2-379"
Alpl.Rn00564931_m1   "2-10"       "2-106"      "2-202"        "2-298"
B2m.Rn00560865_m1    "2-37"       "2-133"      "2-229"        "2-325"
                    gp.1.day.3.v gp.2.day.3.v gp.5.day.3.mia gp.6.day.3.mia
Actb.Rn00667869_m1   "PlateA-38"  "PlateA-134" "PlateA-230"   "PlateA-326"
Adipoq.Rn00595250_m1 "PlateA-61"  "PlateA-157" "PlateA-253"   "PlateA-349"
Adrbk1.Rn00562822_m1 "PlateA-66"  "PlateA-162" "PlateA-258"   "PlateA-354"
Agtrl1.Rn00580252_s1 "PlateA-91"  "PlateA-187" "PlateA-283"   "PlateA-379"
Alpl.Rn00564931_m1   "PlateA-10"  "PlateA-106" "PlateA-202"   "PlateA-298"
B2m.Rn00560865_m1    "PlateA-37"  "PlateA-133" "PlateA-229"   "PlateA-325"
                    gp.3.day.3.v gp.4.day.3.v gp.7.day.3.mia gp.8.day.3.mia
Actb.Rn00667869_m1   "PlateB-38"  "PlateB-134" "PlateB-230"   "PlateB-326"
Adipoq.Rn00595250_m1 "PlateB-61"  "PlateB-157" "PlateB-253"   "PlateB-349"
Adrbk1.Rn00562822_m1 "PlateB-66"  "PlateB-162" "PlateB-258"   "PlateB-354"
Agtrl1.Rn00580252_s1 "PlateB-91"  "PlateB-187" "PlateB-283"   "PlateB-379"
Alpl.Rn00564931_m1   "PlateB-10"  "PlateB-106" "PlateB-202"   "PlateB-298"
B2m.Rn00560865_m1    "PlateB-37"  "PlateB-133" "PlateB-229"   "PlateB-325"
```

If the files to be combined do not have matching detector names, or if duplicate sample or plate names are given, read.taqman will stop and give an error message.
When reading in `qPCR` files with `read.qPCR`, `exprs.well.order` will be populated as long as `Well` and `Plate ID` columns are given in the input file, otherwise the `exprs.well.order` slot will be `NULL`.

So when plate ID and Well data are given:

```
> head(exprs.well.order(qPCRBatch.qPCR))
```

```
                   caseA caseB controlA controlB
gene_ai_TechReps.1  A-18  A-43     B-18     B-43
gene_ai_TechReps.2  C-18  C-43     D-18     D-43
gene_az_TechReps.1  A-23  A-48     B-23     B-48
gene_az_TechReps.2  C-23  C-48     D-23     D-48
gene_bc_TechReps.1   A-6  A-31      B-6     B-31
gene_bc_TechReps.2   C-6  C-31      D-6     D-31
```

And when they are not:

```
> qPCR.example.noPlateOrWell <- file.path(path, "qPCR.noPlateOrWell.txt")
> qPCRBatch.qPCR.noPlateOrWell <- read.qPCR(qPCR.example.noPlateOrWell)
> exprs.well.order(qPCRBatch.qPCR.noPlateOrWell)
```

```
NULL
```

Once a qPCRBatch has been populated it is theoretically possible to use any tool which takes as it's input an `exprs` set matrix. However it is important to bear in mind the values are not raw expression values but Cq values, and a lower Cq will indicate a higher expression level for a given transcript in the sample. Also it is important to note that when normalising, the amount is relative and is intended to be compared to another condition or tissue type in order to look for differential expression between condition; the technology is not designed to give absolute quantification.