Manual

DIAMONDS

Affymetrix & Illumina Normalization tool

1.0

BSc. Bio-informatics Student: Job van Riet

22-4-2014

Contents

Abstract

This is the manual to the complete web-front-/back-end application used in the normalization of Affymetrix and Illumina technology-based gene-expression data. In future, Agilent microarray technology will also be supported.

This tool can be used to perform background correction, variance stabilization, normalization and quality assessment. There are multiple versions available; a local version of the Illumina normalization pipeline and a web-based version with front-/back-end support.

The original R scripts for the background correction, normalization, variance stabilization, quality assessment and statistics of Affymetrix gene-array and Illumina Beadchip expression data are provided by the Department of Bioinformatics - BiGCaT Bioinformatics and Systems Biology Research Group Maastricht University as a module in the ArrayAnalysis package.

These R scripts have been modified under the Apache v2.0 license.

Both technical and usage information is described in this document.

For an overview of all functionalities of this pipeline, see chapter 2. For an overview of the functionalities of the interface, see chapter 3. For an overview of the technical aspects, see chapter 4.

1. Background

Background correction and normalization are generally performed to reduce noise caused by technical and biological variance and transforming the distribution of expressions into a Gaussian (normal) distribution to allow for standardized statistics.

This is a standard procedure when analyzing gene expression arrays and an existing online-available pipeline was already used. This pipeline was originally developed by the Department of Bioinformatics - BiGCaT Bioinformatics and Systems Biology Research Group Maastricht University as a module in the ArrayAnalysis package. It was however deemed to be more effective to locally maintain and run this pipeline to allow for easier editing and improving/addition of functionalities. This pipeline has also been redesigned to allow for stand-alone functionality and integration within the DIAMONDs platform and OpenCPU.

The improvements and added functionality upon the original pipeline are the use of a relational database to store all the data, logging, subset of samples to use in normalizing and statistics, command line argument parser (allowing for easier manipulation of variables related to used methodology), web-interface to alter/view data and run the pipeline with a selection of parameters, improved code-commentary and cleaner code.

For Illumina technologies, this pipeline uses the initial output files from Illumina GenomeStudio Software[1,2] after annotating and analyzing the probes and corresponding gene expression profiles, defined as the Sample Probe Profile and Gene Probe Profile.

For Affymetrix technologies, this pipeline uses .CEL files. Version 3 files were generated by the MAS software while version 4 files are generated by the GCOS software.

The user can then select the methods of background correcting, normalization, variance stabilization and which statistics should be run, optionally choosing to only use a subset of samples in the statistics. The samples (and information about these samples such as which compound is used, noel, noAel, concentration compound and all other defined information), statistics and the corresponding normalized gene expressions are stored in a relational database for use in future analysis. The possible statistics are a boxplot of amplitude of expression levels per sample, density plot of intensity of expressions per sample, density plot of coefficient of variance from expressions per sample, clustering of all samples based on expression, PCA per sample based on expressions, array correlation plot of all samples on array.

This pipeline outputs normalized data and statistics relating to the quality assessment of the data and analysis of samples based on user selection of methods and samples.

For an overview of all functionalities of this pipeline, see chapter 2. For an overview of the functionalities of the interface, see chapter 3. For an overview of the technical aspects, see chapter 4.

2. Pipeline description and manual

Each technology (Illumina, Affymetrix, Agilent) has an individual pipeline instead of a combined pipeline as this greatly increases the simplicity of adding and testing additional functionalities.

There are multiple versions of the pipeline available, there is a local pipeline for Illumina technology only, an DIAMONDS-integrated pipeline for all three technologies and an openCPU version for performing the resource-heavy calculations on an external calculation server by performing an R call.

2.1. Summary of the methodology used in the Illumina normalization pipeline

The main flow of the Illumima pipeline is shown in figure 1 “Global flowchart Illumina normalization pipeline”**.** All versions of the Illumina pipeline (stand-alone, OpenCPU and DIAMONDs-integrated) have a similar approach with only a slightly differing method of data input and output and small segments of internal code. These pipelines can be run using rscript using:

```
Rscript runIlluminaNormalization.R -i <inputDir> -o <outputDir> -s <sampleProfile.txt> -c <controlProfile.txt> -d <descriptionFile.txt> etc.
```

and in RStudio by commenting out the following line:

```
userParameters <- getArguments(commandArgs(trailingOnly = TRUE))
```

and replacing it with:

```
userParameters <- getArguments(c("-h", "--rawDataQC" ,FALSE etc ))
```

The outline of the flow is as such:

1. Start count of the runtime of the script
2. Read the config.R file containing the configuration options such as the user capable of interacting with the database and the main folder of the application on the user’s server.
3. Read the command-line arguments supplied to optParser.
    a. Correct the file paths of the folders by adding/removing characters where needed.
    b. Check if the combination of species, array type and array annotation file is valid.
    c. Start a log-file and sink all output to this file if `--createLog TRUE`, Log file is kept in the output directory.
4. Load the functions of the other R scripts for this pipeline.
    a. functions_loadpackages.R
    b. functions_makeImages.R
    c. functions_myDB.R
    d. functions_qualityControl.R
5. Load additional required R packages.
    a. Automatically installs any missing packages if executing user has permissions to write to R library folder.
6. Read in the descriptionFile containing the samples that should be used in the normalization and the coloring/grouping of samples in statistics, this file also contains the sampleNames as they should be shown in the plots and the link between the designated name of the assay.

    a. Create generic sampleNames, strips symbols which could cause problems and replace with X.
7. Read in the Sample Probe Profile containing the raw intensities of the gene-expressions, this returns a LumiBatch R Object containing the raw expression values.
    a. Create generic sampleNames, strips symbols which could cause problems and replace with X.
8. Make a subset of samples to use during normalization based on the samples in the description file if --normSubset TRUE.
9. Check if all samples in the descriptionFile are also present in the rawData LumiBatch Object.
    a. If rawData LumiBatch Object has more samples than descriptionFile; stop the script.
        i. This is not the case if --normSubset TRUE as the rawData only contains the samples defined in the descriptionFile.txt
    b. Check if all sampleNames are unique.
    c. Reorder the rawData samples based on the occurrence in descriptionFile.
10. If --perGroup TRUE, reorder the samples based on the group in which they should be shown in the statistics.
11. If --bgSub FALSE, perform background correction.
    a. Load the Control Probe Profile to the rawData LumiBatch object.
    b. Perform background correction based on the –bgcorrect.m <method>
12. Perform normalization on rawData Lumibatch Object
    a. Normalization method based on –normalization.m <method>
    b. Perform variance stabilization based on –variance.m <method> if –variance.stabilize TRUE
    c. Return a normData LumiBatch Object
13. Create eSets expression matrix of rawData and normData. (Container for high-throughput assays and experimental metadata.)
14. Create summary files containing the mean, SD, distance to mean and detection rate (0,01) for each sample based on –rawSummary TRUE & –normSummary TRUE
15. Save the LumiBatch R object of rawData and normData in the outputDirectory, based on –save.rawData TRUE & –save.normData TRUE
16. Perform statistics if –performStatistics TRUE
    a. Perform the calculations and make the plots for the raw data if –rawDataQC TRUE
        i. Make a subset of rawData on which to make the quality assessment if –statSubsetTRUE, based on the samples found in the –statFile <path>.
        ii. Generate the information needed to make the plots such as a vector of symbols and colors etc.
        iii. Each plot is made based on a different parameter.
            1. Plots are stored in the appropriate /statistics/ folder based on the–idStatistics <number>
    b. Perform the calculations and make the plots for the norm data if –normDataQC TRUE
        i. Make a subset of normData on which to make the quality assessment if –statSubsetTRUE, based on the samples found in the –statFile <path>.

ii. Generate the information needed to make the plots such as a vector of symbols and colors etc.
iii. Each plot is made based on a different parameter.
    1. Plots are stored in the appropriate /statistics/ folder based on the –idStatistics <number>

17. Load in old normalized data if only statistics should be run and the normalized R Lumibatch object is already available if –loadOldNorm TRUE
18. Perform filtering if –filtering TRUE, this will create a filtered.normData LumiBatch object in which the low/no expressed probes are filtered based on the threshold defined in –filter.Th TRUE and the min. number of beads in –filter.dp TRUE
19. Create tab-delimited annotation files which contain the expression values of all LumiBatch Objects if –createAnno TRUE, these will be stored in the output directory of the normalized data.
20. Save the expressions to the database if –saveToDB TRUE, this will save the normalized expressions to the relational database. This can take a **long** time as the big O is $O(nSamples \cdot nProbes)$.
21. Close the log file and set the job as completed and append the running time of the pipeline to script.

A job is kept in the database for the normalization/statistics run using --idJob <number> which keeps track of the process and any errors found in the pipeline. All files will also be stored in the database in order to show the user which files belong to which study and where they are located. A user to connect to the relational database housing the ERD shown in chapter 4. is required for the DIAMONDS-integrated and OpenCPU version and configured in the config.R script located in the /R/ folder of the application.

2.1.1. Parameters Illumina Pipeline

The pipeline is highly adjustable using many parameters, some of the parameters are needed in order to use the pipeline. These required parameters are:

```
-i INPUTDIR, --inputDir=INPUTDIR
                    Path to folder where the Control_Probe_Profile, Sample_Probe_Profile and Description

              -o OUTPUTDIR, --outputDir=OUTPUTDIR
                    Path to folder where the output files will be stored
default = [/var/www/normdb//expressionData]

-n STUDYNAME, --studyName=STUDYNAME
                    Used to be called ns, Name of the study (Used in naming the output files)
default = [2014-04-22_14.02.30]

              -j IDJOB, --idJob=IDJOB
                    Job ID for updating the job status if failed and done.
default = [1]

              -x IDSTUDY, --idStudy=IDSTUDY
                    idStudy, keeps track of what study this is.
default = [1]
```

```
        -s SAMPLEPROBEPROFILEPATH, --sampleProbeProfilePath=SAMPLEPROBEPROFILEPATH
                Used to be called expFile, contains the expression values of the samples
default = [Sample_Probe_Profile.txt]

        -c CONTROLPROBEPROFILEPATH, --controlProbeProfilePath=CONTROLPROBEPROFILEPATH
                Used to be called bgFile, contains the expression values of the control probes
default = [Control_Probe_Profile.txt]

        -d DESCFILE, --descFile=DESCFILE
                Tab-delimited file containing: arrayNames | sampleNames | sampleGroup
default = [descriptionFile.txt]
```

If –normalize TRUE

```
        -y IDNORM, --idNorm=IDNORM
                idNorm, keeps track of what normalization run this is.
default = [1]
```

If –normalize FALSE

```
        -f LOADOLDNORM, --loadOldNorm=LOADOLDNORM
                Whether to load the old normalized data (given with -m/--normData).
default = [FALSE]
```

If –performStatistics TRUE

```
        -O STATISTICSDIR, --statisticsDir=STATISTICSDIR
                Path to folder where the output statistics files will be stored
default = [/var/www/normdb//statistics]
```

If –statSubSet TRUE

```
        -B STATSUBSET, --statSubset=STATSUBSET
                Whether statistics should be done only on a subset. (defined in --statFile)
default = [FALSE]
```

These are all parameters of the Illumina pipeline, not all of these parameters need to be supplied as each also has a default value:

```
Options:
        -i INPUTDIR, --inputDir=INPUTDIR
                Path to folder where the Control_Probe_Profile, Sample_Probe_Profile and Description
file are found
default = [/var/www/normdb//data]

        -o OUTPUTDIR, --outputDir=OUTPUTDIR
                Path to folder where the output files will be stored
default = [/var/www/normdb//expressionData]

        -O STATISTICSDIR, --statisticsDir=STATISTICSDIR
                Path to folder where the output statistics files will be stored
default = [/var/www/normdb//statistics]
```

--scriptDir=SCRIPTDIR
Path to folder where the scripts are stored.
default = [/var/www/normdb//R/illuminaNorm]

--species=SPECIES
Species used for samples (E.g. Human, Mouse or Rat etc)
default = [Human]

--arrayType=ARRAYTYPE
Main type of array that was used (E.g. HumanHT-12 or HumanWG-6)
default = [HumanHT-12]

--annoType=ANNOTYPE
What annotation file should be used to check Genes+Probes etc (E.g. humanHT-12_V4_0_R2_15002873_B)
default = [HumanHT-12_V4_0_R2_15002873_B]

-n STUDYNAME, --studyName=STUDYNAME
Used to be called ns, Name of the study (Used in naming the output files)
default = [2014-04-22_14.02.30]

--createAnno=CREATEANNO
Create an annotation file containing lumiID, probeID etc.
default = [TRUE]

--saveHistory=SAVEHISTORY
Whether to save the R history to the output directory.
default = [TRUE]

-l CREATELOG, --createLog=CREATELOG
Create a log file in the output directory. Captures ALL messages from stdout and stderr.
default = [FALSE]

-j IDJOB, --idJob=IDJOB
Job ID for updating the job status if failed and done.
default = [1]

-y IDNORM, --idNorm=IDNORM
idNorm, keeps track of what normalization run this is.
default = [1]

-x IDSTUDY, --idStudy=IDSTUDY
idStudy, keeps track of what study this is.
default = [1]

-D SAVETODB, --saveToDB=SAVETODB
Whether to save data to DB.
default = [TRUE]

-S IDSTATISTICS, --idStatistics=IDSTATISTICS
IdStatistics on which to save the files.
default = [1]

-B STATSUBSET, --statSubset=STATSUBSET
Whether statistics should be done only on a subset. (defined in --statFile)
default = [FALSE]

-X NORMSUBSET, --normSubset=NORMSUBSET
Whether normalization should be done on a subset of samples. (defined in --description File)
default = [FALSE]

-s SAMPLEPROBEPROFILEPATH, --sampleProbeProfilePath=SAMPLEPROBEPROFILEPATH
Used to be called expFile, contains the expression values of the samples
default = [Sample_Probe_Profile.txt]

-c CONTROLPROBEPROFILEPATH, --controlProbeProfilePath=CONTROLPROBEPROFILEPATH
Used to be called bgFile, contains the expression values of the control probes
default = [Control_Probe_Profile.txt]

-d DESCFILE, --descFile=DESCFILE
Tab-delimited file containing: arrayNames | sampleNames | sampleGroup
default = [descriptionFile.txt]

-F STATFILE, --statFile=STATFILE
File containing a single column with the sampleNames or ArrayNames on which statistics should be performed. If none given, statistics is performed on all samples in the description file.
default = [statSubsetFile.txt]

-m NORMDATA, --normData=NORMDATA
R Lumibatch object containing the normalized expression values.
default = [normData.Rdata]

-f LOADOLDNORM, --loadOldNorm=LOADOLDNORM
Whether to load the old normalized data (given with -m/--normData).
default = [FALSE]

-u BGSUB, --bgSub=BGSUB
Whether background correction has been done already. (E.g. in Illumina GenomeStudio) If FALSE -> perform bg correction using controlProbeProfileFile. If TRUE -> Skip bg correction
default = [FALSE]

--detectionTh=DETECTIONTH
The p-value threshold of determining detectability of the expression.
default = [0.01]

--convertNuID=CONVERTNUID
Determine whether to convert the probe identifier as nuID
default = [TRUE]

--dec=DEC
The character used in the files to indicate decimal values (E.g. '.' or ',' etc.)
default = [.]

--parseColumnName=PARSECOLUMNNAME
Determine whether to parse the column names and retrieve the sample information (Assume the sample information is separated by a tab)
default = [FALSE]

--checkDupId=CHECKDUPID
Determine whether to check duplicated TargetIDs or ProbeIds. The duplicated ones will be averaged.
default = [TRUE]

--save.rawData=SAVE.RAWDATA
Whether to save lumi.batch of raw data as R object in WORK.DIR
default = [TRUE]

--normType=NORMTYPE
Type of normalization to do (lumi or neqc)
default = [lumi]

--bgcorrect.m=BGCORRECT.M
List of the parameters for lumiExpresso{lumiB}, method for background correction.
Possible parameters: c('none', 'bgAdjust', 'forcePositive', 'bgAdjust.affy')
default = [bgAdjust]

--variance.stabilize=VARIANCE.STABILIZE
Whether do to variance stabilization
default = [TRUE]

--variance.m=VARIANCE.M
Method of variance stabilization for lumiExpresso{lumiT} package.
Possible parameters are: c('vst', 'log2', 'cubicRoot')
default = [log2]

--normalize=NORMALIZE
Whether to normalize or not
default = [TRUE]

--normalization.m=NORMALIZATION.M
List of parameters for lumiExpresso{lumiB}, method of normalization.
parameters are: c(quantile, rsn, ssn, loess, vsn)
default = [quantile]

--save.normData=SAVE.NORMDATA
Whether to save lumi.batch of normalized data as R object in WORK.DIR
default = [TRUE]

--filtering=FILTERING
Whether filtering of probes with a low/no expression should be performed
default = [TRUE]

--filter.Th=FILTER.TH
Threshold for probe filtering.
default = [0.01]

--filter.dp=FILTER.DP
Threshold for the count of the same probe in multiple samples
default = [0]

-p PERFORMSTATISTICS, --performStatistics=PERFORMSTATISTICS
Should clustering and PCA be done alongside the normalization of the data? (Individual options are below)
default = [TRUE]

--rawDataQC=RAWDATAQC
Determine whether to do QC assessment for the raw data; if false no summary can be computed.
default = [TRUE]

--normDataQC=NORMDATAQC
Determine whether to do QC assessment for the normed data; if false no summary can be computed.
default = [TRUE]

--rawSummary=RAWSUMMARY
Whether to create a summary table in WORK.DIR of the raw data
default = [TRUE]

--normSummary=NORMSUMMARY
Whether to create a summary table in WORK.DIR of the normalized data
default = [TRUE]

--perGroup=PERGROUP
Reorder rawData lumibatch file FIRST on Group and THEN ON sampleNames (Used for the order of visualization in plots)
default = [TRUE]

--raw.boxplot=RAW.BOXPLOT
Should a boxplot be made for the raw data.
default = [TRUE]

--raw.density=RAW.DENSITY
Should a density plot be made for the raw data.
default = [TRUE]

--raw.cv=RAW.CV
Should a cv plot be made for the raw data.
default = [TRUE]

--raw.sampleRelation=RAW.SAMPLERELATION
Should a sample relation plot be made for the raw data.
default = [TRUE]

--raw.pca=RAW.PCA
Should a PCA be made for the raw data.
default = [TRUE]

--raw.correl=RAW.CORREL
Should a correlation plot be made for the raw data.
default = [TRUE]

--norm.boxplot=NORM.BOXPLOT
Should a boxplot be made for the normalized data.
default = [TRUE]

--norm.density=NORM.DENSITY
Should a density plot be made for the normalized data.
default = [TRUE]

--norm.cv=NORM.CV
Should a cv plot be made for the normalized data.
default = [TRUE]

--norm.sampleRelation=NORM.SAMPLERELATION
Should a sample relation plot be made for the normalized data.

default = [TRUE]

--norm.pca=NORM.PCA
Should a PCA be made for the normalized data.
default = [TRUE]

--norm.correl=NORM.CORREL
Should a correlation plot be made for the normalized data.
default = [TRUE]

--clusterOption1=CLUSTEROPTION1
Distance calculation method.
Possible parameters are c('Pearson', 'Spearman', 'Euclidean')
default = [Pearson]

--clusterOption2=CLUSTEROPTION2
Method of clustering.
Possible parameters are c('Ward', 'McQuitty', 'average', 'median', 'single', 'complete', 'centroid')
default = [average]

--img.width=IMG.WIDTH
The max. width of the plots.
default = [1920]

--img.heigth=IMG.HEIGTH
The max. heigth of the plots.
default = [1080]

--img.pointSize=IMG.POINTSIZE
The size of the points on plots.
default = [24]

--img.maxArray=IMG.MAXARRAY
The maximum datapoint on each plot per page.
default = [41]

-h, --help
Show this help message and exit

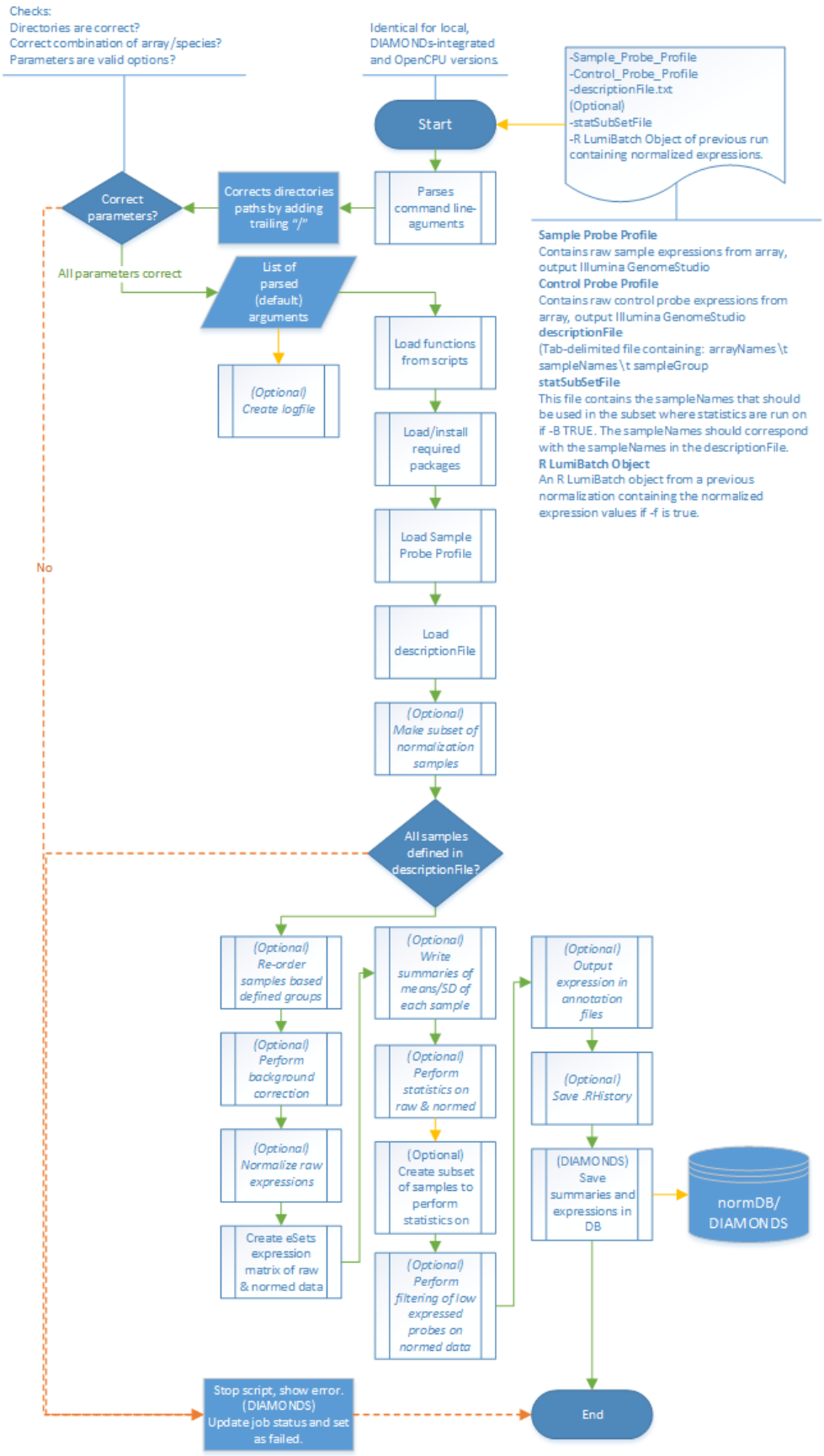

Figure 1, global flowchart of the Illumina normalization pipeline. All versions (local, DIAMONDs-integrated and OpenCPU) have an identical flow with only the input/output directories and small segments of internal code differing.

2.2. Summary of the methodology used in the Affymetrix normalization pipeline

The main flow of the Affymetrix pipeline is shown in figure 2 “Global flowchart Affymetrix normalization pipeline”**.** The Affymetrix pipeline does not have a local version as this was implemented directly into the DIAMONDS-integrated version. This pipelines can be run using rscript using:

```
Rscript runAffymetrixNormalization.R -i \<inputDir\> -o \<outputDir\> -s \<sampleProfile.txt\> -c \<controlProfile.txt\> -d \<descriptionFile.txt\> etc.
```

and in RStudio by commenting out the following line:

```
userParameters <- getArguments(commandArgs(trailingOnly = TRUE))
```

and replacing it with:

```
userParameters <- getArguments(c("-h", "--outputDirectory" , “/var/www/output/”, etc ))
```

The outline of the flow is as such:

1. Start count of the runtime of the script.
2. Read the config.R file containing the configuration options such as the user capable of interacting with the database and the main folder of the application on the user’s server.
3. Read the command-line arguments supplied to optParser.
    a. Correct the file paths of the folders by adding/removing characters where needed.
    b. Check if the combination of species, array type and array annotation file is valid.
    c. Start a log-file and sink all output to this file if `--createLog TRUE`, Log file is kept in the output directory.
4. Load the functions of the other R scripts for this pipeline.
    a. functions_loadpackages.R
    b. functions_makeImages.R
    c. functions_myDB.R
    d. functions_qualityControl.R
5. Load additional required R packages.
    a. Automatically installs any missing packages if executing user has permissions to write to R library folder.
6. Read in the descriptionFile containing the samples that should be used in the normalization and the coloring/grouping of samples in statistics, this file also contains the sampleNames as they should be shown in the plots and the link between the designated name of the assay.
    a. Create generic sampleNames, strips symbols which could cause problems and replace with X.
7. Read in the Sample Probe Profile containing the raw intensities of the gene-expressions, this returns a LumiBatch R Object containing the raw expression values.
    a. Create generic sampleNames, strips symbols which could cause problems and replace with X.
8. Make a subset of samples to use during normalization based on the samples in the description file if `--normSubset TRUE`.
9. Check if all samples in the descriptionFile are also present in the rawData LumiBatch Object.

   a. If rawData LumiBatch Object has more samples than descriptionFile; stop the script.
      i. This is not the case if --normSubset TRUE as the rawData only contains the samples defined in the descriptionFile.txt
   b. Check if all sampleNames are unique.
   c. Reorder the rawData samples based on the occurrence in descriptionFile.
10. If --perGroup TRUE, reorder the samples based on the group in which they should be shown in the statistics.
11. Begin the creation of the statistics report.
   a. Perform statistics on the raw data based on the supplied logical parameters.
12. Normalize the data if --normalize TRUE based on the method defined in --normMeth TRUE
   a. Use a custom annotation file if --useCustomAnnotation TRUE, the file containing the custom annotation is supplied in --CDFtype <path>
13. Load in old normalized data if only statistics should be run and the normalized R Lumibatch object is already available if –loadOldNorm TRUE
14. Perform statistics on the normalized data if –normDataQC <path>
   a. Plots are stored in the appropriate /statistics/ folder based on the–idStatistics <number>
15. Save all normalized data and plots.
16. Save the expressions to the database if –saveToDB TRUE, this will save the normalized expressions to the relational database. This can take a **long** time as the big O is $O(nSamples \cdot nProbes)$.
17. Close the log file and set the job as completed and append the running time of the pipeline to script.

A job is kept in the database for the normalization/statistics run using --idJob <number> which keeps track of the process and any errors found in the pipeline. All files will also be stored in the database in order to show the user which files belong to which study and where they are located. A user to connect to the relational database housing the ERD shown in chapter 4. is required for the DIAMONDS-integrated and OpenCPU version and configured in the config.R script located in the /R/ folder of the application.

2.2.1. Parameters Affymetrix pipeline

```
Options:
        -j IDJOB, --idJob=IDJOB
                Job ID for updating the job status if failed and done.
default = [1]

        -y IDNORM, --idNorm=IDNORM
                idNorm, keeps track of what normalization run this is.
default = [1]

        -x IDSTUDY, --idStudy=IDSTUDY
                idStudy, keeps track of what study this is.
default = [1]

        -D SAVETODB, --saveToDB=SAVETODB
                Whether to save data to DB.
default = [TRUE]
```

```
        -S IDSTATISTICS, --idStatistics=IDSTATISTICS
                IdStatistics on which to save the files.
default = [1]

        -n STUDYNAME, --studyName=STUDYNAME
                Used to be called ns, Name of the study (Used in naming the output files)
default = [2014-04-22_14.30.30]

        -i INPUTDIR, --inputDir=INPUTDIR
                Path to folder where the .CEL files are found
default = [/var/www/normdb//data]

        -o OUTPUTDIR, --outputDir=OUTPUTDIR
                Path to folder where the output files will be stored
default = [/var/www/normdb//expressionData]

        -O STATISTICSDIR, --statisticsDir=STATISTICSDIR
                Path to folder where the output statistics files will be stored
default = [/var/www/normdb//statistics]

        --scriptDir=SCRIPTDIR
                Path to folder where the scripts are stored.
default = [/var/www/normdb//R/affymetrixNorm]

        -d DESCFILE, --descFile=DESCFILE
                Tab-delimited file containing: arrayNames | sampleNames | sampleGroup
default = [descriptionFile.txt]

        -m NORMDATA, --normData=NORMDATA
                R Lumibatch object containing the normalized expression values. (If normalization has b
een run before)
default = [normData.Rdata]

        -F STATFILE, --statFile=STATFILE
                File containing a single column with the sampleNames or ArrayNames on which statistic
s should be performed. If none given, statistics is performed on all samples in the description file.
default = [statSubsetFile.txt]

        -a USECUSTOMANNOTATION, --useCustomAnnotation=USECUSTOMANNOTATION
                Whether to use a custom annotation file, given in -A
default = [FALSE]

        -A CUSTOMANNOTATION, --customAnnotation=CUSTOMANNOTATION
                File containing the custom annotation of the array
default = []

        --saveHistory=SAVEHISTORY
                Whether to save the R history to the output directory.
default = [TRUE]

        -X NORMSUBSET, --normSubset=NORMSUBSET
                Whether normalization should be done on a subset of samples. (defined in --description
File)
default = [FALSE]

        --normalize=NORMALIZE
```

Whether to normalize or not
default = [TRUE]

--save.normData=SAVE.NORMDATA
Whether to save lumi.batch of normalized data as R object in WORK.DIR
default = [TRUE]

--save.rawData=SAVE.RAWDATA
Whether to save lumi.batch of raw data as R object in WORK.DIR
default = [TRUE]

-f LOADOLDNORM, --loadOldNorm=LOADOLDNORM
Whether to load the old normalized data (given with -m/--normData).
default = [FALSE]

-z NORMMETH, --normMeth=NORMMETH
possible values for Data pre-processing: (RMA, GCRMA, PLIER, none)
default = [RMA]

-J NORMOPTION1, --normOption1=NORMOPTION1
two possible values: (group, dataset)
default = [dataset]

-L CDFTYPE, --CDFtype=CDFTYPE
annotation format (default: ENSG), possibilities: (ENTREZG, REFSEQ, ENSG, ENSE, ENST, VEGAG, VEGAE, VEGAT, TAIRG, TAIRT, UG, MIRBASEF, MIRBASEG)
default = [ENSG]

--species=SPECIES
It is required when customCDF is called. Possibilities: abbreviations: (Ag, At, Bt, Ce, Cf, Dr, Dm, Gg, Hs, MAmu, Mm, Os, Rn, Sc, Sp, Ss or full names: Anopheles gambiae, Arabidopsis thaliana, Bos taurus, Caenorhabditis elegans, Canis familiaris, Danio rerio, Drosophila melanogaster, Gallus gallus, Homo sapiens, Macaca mulatta, Mus musculus, Oryza sativa, Rattus norvegicus, Saccharomyces cerevisiae, Schizosaccharomyces pombe, Sus scrofa)
default = []

-B STATSUBSET, --statSubset=STATSUBSET
Whether statistics should be done only on a subset. (defined in --statFile)
default = [FALSE]

-G PERGROUP, --perGroup=PERGROUP
boolean for whether the arrays have to be ordered per group in the plots FALSE keeps the order of the description file, TRUE reorders per group
default = [TRUE]

-p PERFORMSTATISTICS, --performStatistics=PERFORMSTATISTICS
Should clustering and PCA be done alongside the normalization of the data? (Individual options are below)
default = [TRUE]

--rawDataQC=RAWDATAQC
Determine whether to do QC assessment for the raw data; if false no summary can be computed.
default = [TRUE]

--normDataQC=NORMDATAQC

Determine whether to do QC assessment for the normed data; if false no summary can be computed.
default = [TRUE]

--layoutPlot=LAYOUTPLOT
boolean for plot of the array layout
default = [TRUE]

--controlPlot=CONTROLPLOT
boolean for plots of the AFFX controls on the arrays
default = [TRUE]

--samplePrep=SAMPLEPREP
boolean for Sample prep controls
default = [TRUE]

--ratioPlot=RATIOPLOT
boolean for 3?/5? for b-actin and GAPDH
default = [TRUE]

--degPlot=DEGPLOT
boolean for DNA degration plot
default = [TRUE]

--hybridPlot=HYBRIDPLOT
boolean for Spike-in controls
default = [TRUE]

--percPres=PERCPRES
boolean for Percent present
default = [TRUE]

--posnegDistrib=POSNEGDISTRIB
boolean for +and - controls distribution
default = [TRUE]

--bgPlot=BGPLOT
boolean for Background intensity
default = [TRUE]

--scaleFact=SCALEFACT
boolean for Scale factor
default = [TRUE]

--boxplotRaw=BOXPLOTRAW
boolean for Raw boxplot of log-intensity
default = [TRUE]

--boxplotNorm=BOXPLOTNORM
boolean for Norm boxplot of log-intensity
default = [TRUE]

--densityRaw=DENSITYRAW
boolean for Raw density histrogram
default = [TRUE]

--densityNorm=DENSITYNORM

boolean for Norm density histrogram
default = [TRUE]

--MARaw=MARAW
boolean for Raw MA-plot
default = [TRUE]

--MANorm=MANORM
boolean for Norm MA-plot
default = [TRUE]

--MAOption1=MAOPTION1
two possible values: group or dataset
default = [dataset]

--spatialImage=SPATIALIMAGE
boolean for 2D images
default = [TRUE]

--PLMimage=PLMIMAGE
boolean for 2D PLM plots
default = [TRUE]

--posnegCOI=POSNEGCOI
boolean for + and ? controls COI plot
default = [TRUE]

--Nuse=NUSE
boolean for NUSE
default = [TRUE]

--Rle=RLE
boolean for RLE
default = [TRUE]

--correlRaw=CORRELRAW
boolean for Raw correlation plot
default = [TRUE]

--correlNorm=CORRELNORM
boolean for Norm correlation plot
default = [TRUE]

--clusterRaw=CLUSTERRAW
boolean for Raw hierarchical clustering
default = [TRUE]

--clusterNorm=CLUSTERNORM
boolean for Norm hierarchical clustering
default = [TRUE]

--clusterOption1=CLUSTEROPTION1
possible values for Distance: (Spearman, Pearson, Euclidian)
default = [Spearman]

--clusterOption2=CLUSTEROPTION2
possible values for Tree: (ward, singlecomplete, average, mcquitty, median, centroid)

default = [ward]

--PCARaw=PCARAW
boolean for PCA analysis of raw data
default = [TRUE]

--PCANorm=PCANORM
boolean for PCA analysis of normalized data
default = [TRUE]

--PMAcalls=PMACALLS
boolean for Present/Marginal/Absent calls using MAS5
default = [FALSE]

--img.width=IMG.WIDTH
The max. width of the plots.
default = [1920]

--img.heigth=IMG.HEIGTH
The max. heigth of the plots.
default = [1080]

--img.pointSize=IMG.POINTSIZE
The size of the points on plots.
default = [24]

--img.maxArray=IMG.MAXARRAY
The maximum datapoint on each plot per page.
default = [41]

-h, --help
Show this help message and exit

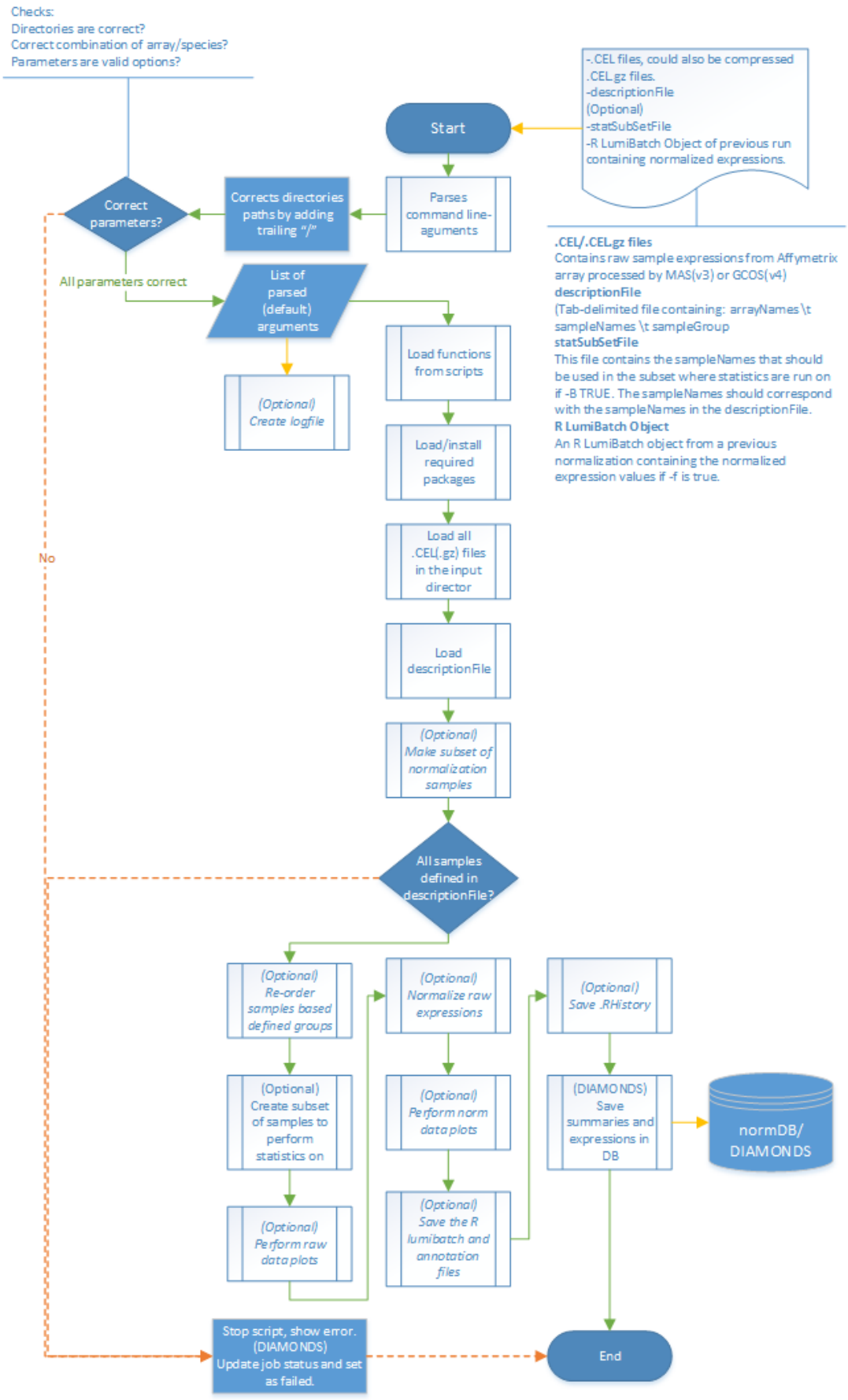

Figure 2, global flowchart of the Affymetrix normalization pipeline.

3. Interface description and manual

A web-interface has been designed to allow for easier implementations of the Illumina normalization pipeline and the subsequent storing and viewing of data. This has been designed in conjunction with a MySQL database.

The general process-flow of this interface is designed as thus:

The user has to create a new “study” or select an exist “study”. This study is used to keep track of the samples included in the study and all the operations performed on this study, alongside all the files that have been uploaded/outputted.

When the user has selected a study, samples can be added to this specific study. The samples are defined as the experimental samples performed on the (gene-expression) microarray. These samples can be supplied with additional information such as which compound is used, type of sample (positive control / negative control / sample etc.) and all other relevant information to the study such as the NOEL / NOAEL / LD50, RNA concentration etc. This is a modular design and has no limit to the amount of information is supplied per sample. The uploaded samples can be viewed and modified using the web-interface.

When samples have been added, omics data can also be added. For Illumina normalization, the required files are the Sample Probe Profile and Control Probe Profile containing the fluorescent values signifying expressional relativity of the samples and control probes.

For Affymetrix normalization, the required files are .CEL files containing the intensity of gen-expressions. Whether each sample is stored into one large .CEL file or in separate .CEL files makes no difference. It can also read compressed .CEL.gz files.

An extra tab-delimited file containing the names of the samples as defined on the array and the names of the samples as defined by the user should also be uploaded if this has not been done when uploading the samples. This is used to match the correct samples to the microarray.

When both samples and omics data has been uploaded to the study, a normalization can be run using user-selected methods and options for the pipeline such as, amongst others, the method of normalization, whether to perform filtering and which statistics should be run. The user can also specify a subset of samples upon which the statistics should be run, this is useful when it is interesting to look at any a specific group of samples.

When normalization has been run, statistics can be run on this normalized data again, also with a subset of samples, without performing normalization.

All plots, files, jobs, samples and additional information from a specific study can be viewed, downloaded and/or altered.

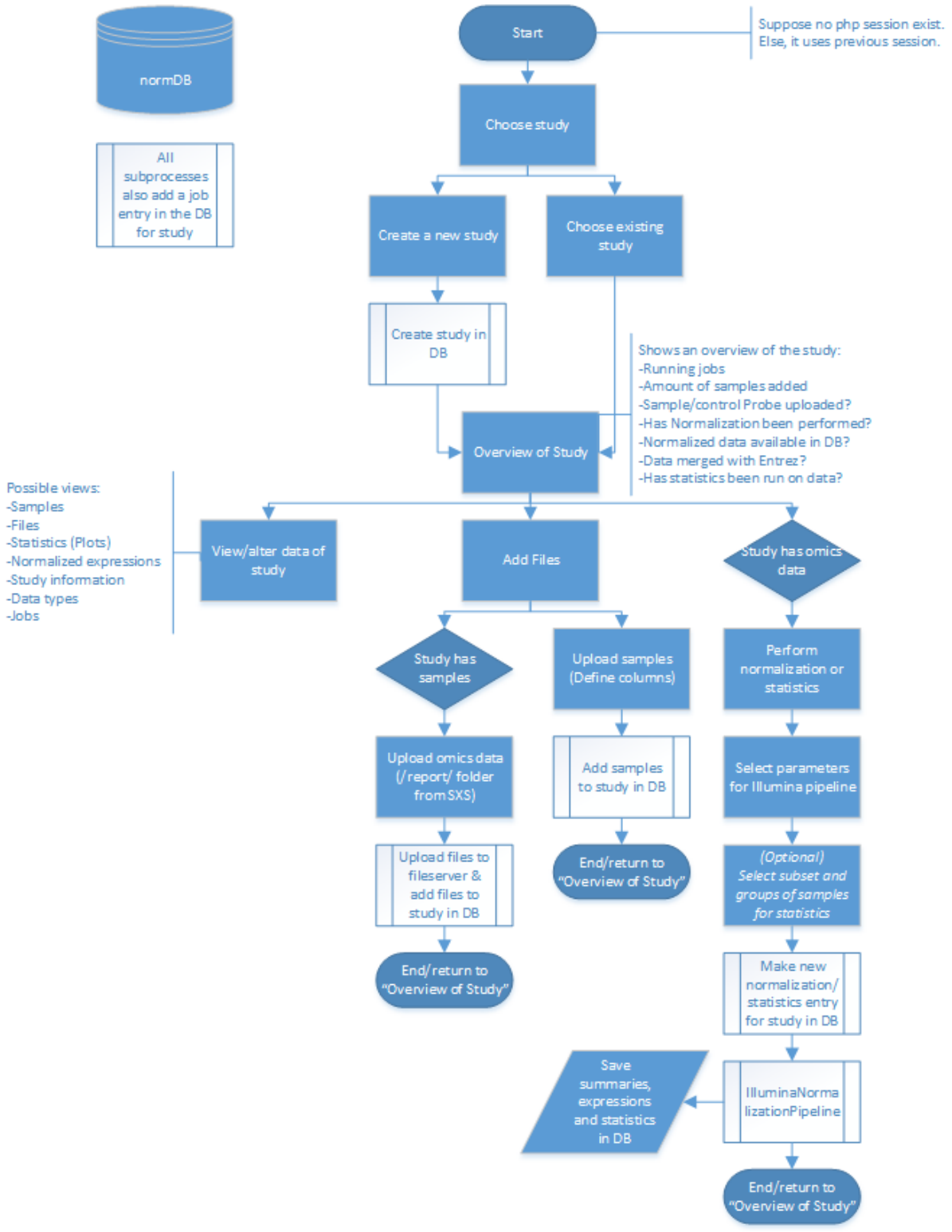

Figure 3, flowchart of the global flow of the front-end interface.

3.1 Manual interface

3.1.1. Page – Navigation Menu

The navigation menu is found on top and is used to navigate the pages, it shows a (2-dimensional) dropdown menu if the mouse hovers above each element.

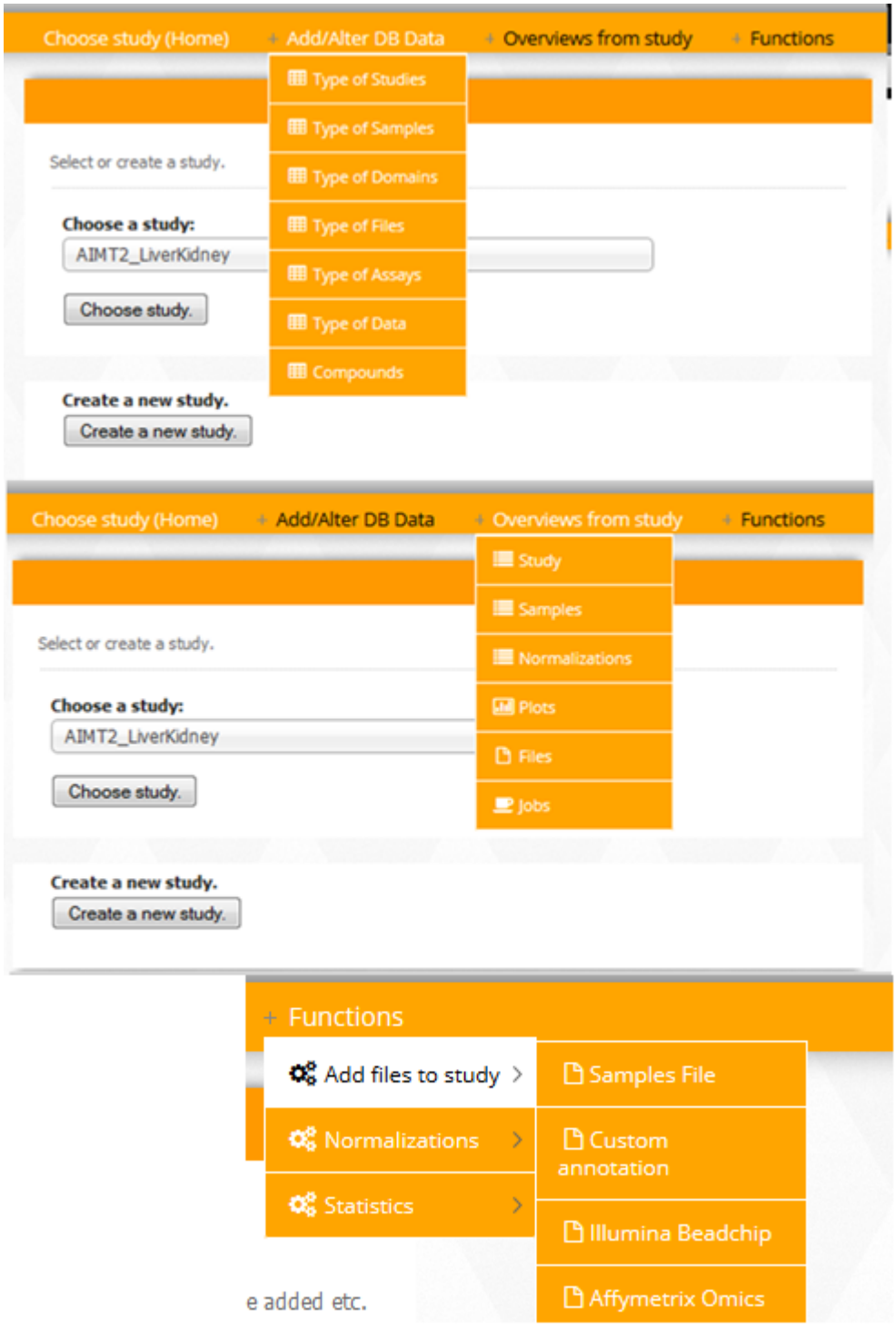

Figure 4, impression of the navigation menu.

3.1.2. Page - chooseStudy

If no study has been made or selected, the user will be presented by the "**chooseStudy**" page. This page can be used to select an existing study on which to perform the operations or the create a new study by opening the "**createStudy**" page. This is also the page which will be presented if no cookie is present in which study information is kept.

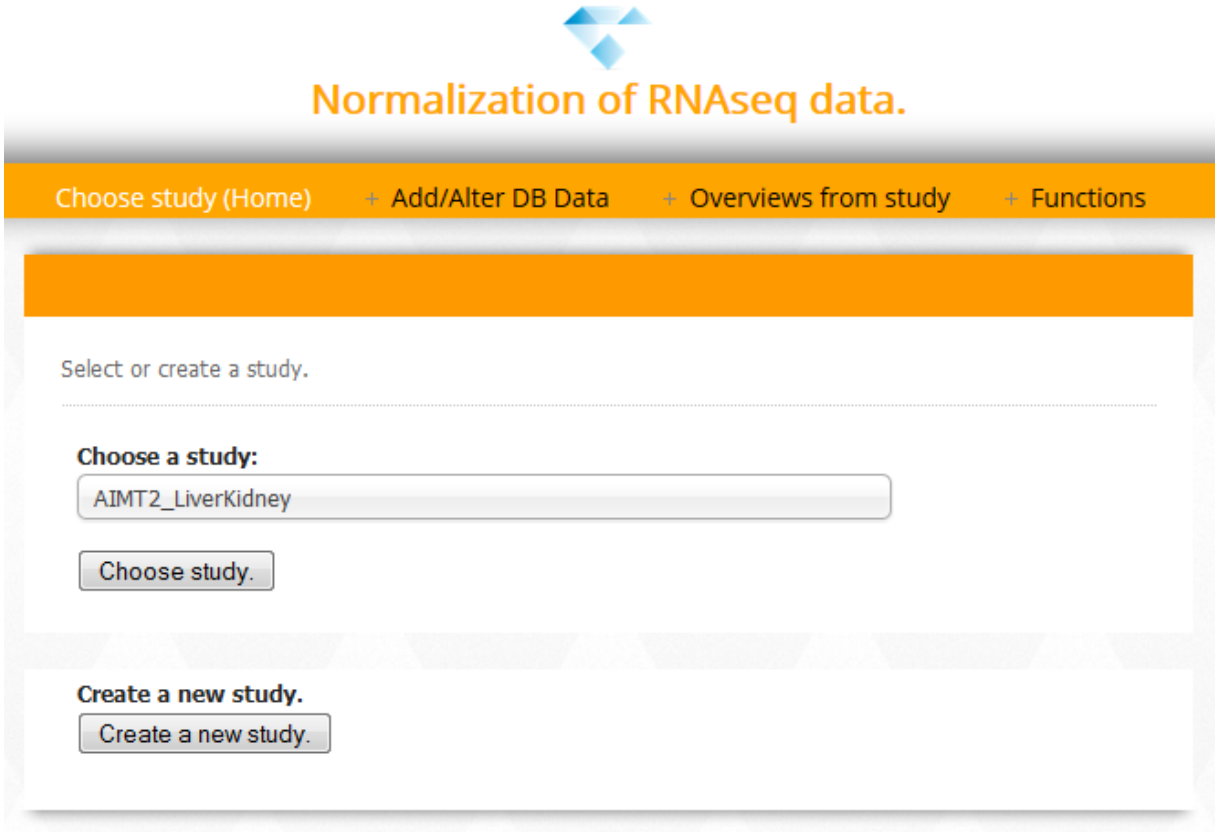

Figure 5, impression of the chooseStudy.php page.

3.1.2. Page – createStudy

This form can be used to create a new study in the database. The dropdown menus can be searched through by simply typing characters that are found in your to-be-searched item. By pressing the "Submit new Study" button, the form is submitted to getForm.php and further functions will store the study in the DB.

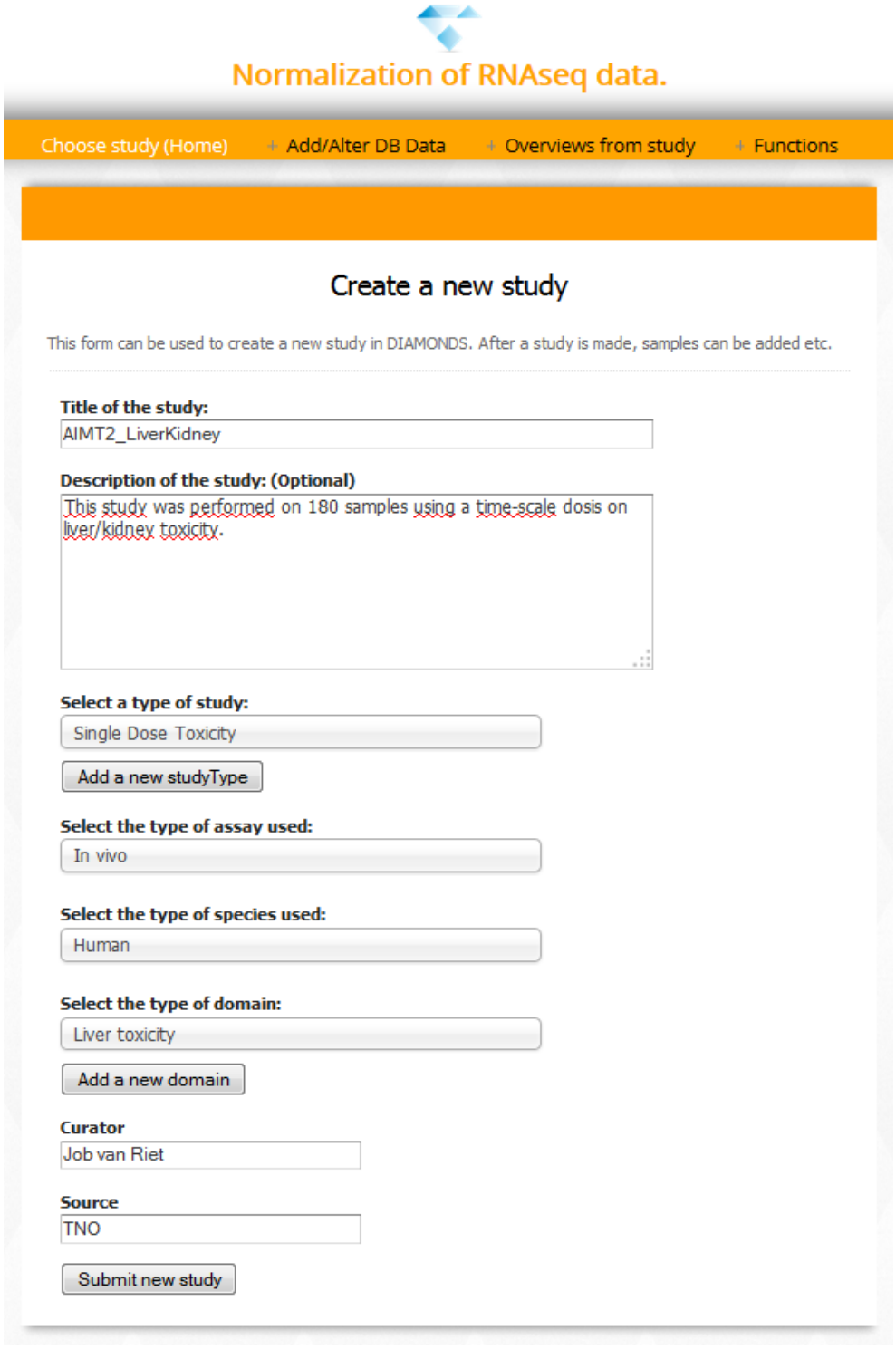

Figure 6, impression of the createStudy.php page.

3.1.3. Page – studyOverview

This page is used to present an overview of the chosen study. It shows if a the study has any running jobs such as normalization of data or statistics and also shows the count of jobs performed so far. It also shows the amount of samples designated to the study and if these samples already have an assayName assigned. The arrayNames are used to find the samples on the microarray. It further shows if a normalization has been run and if the normalized data (R LumiBatch/AffyBatch objects) are present in the database and fileserver.

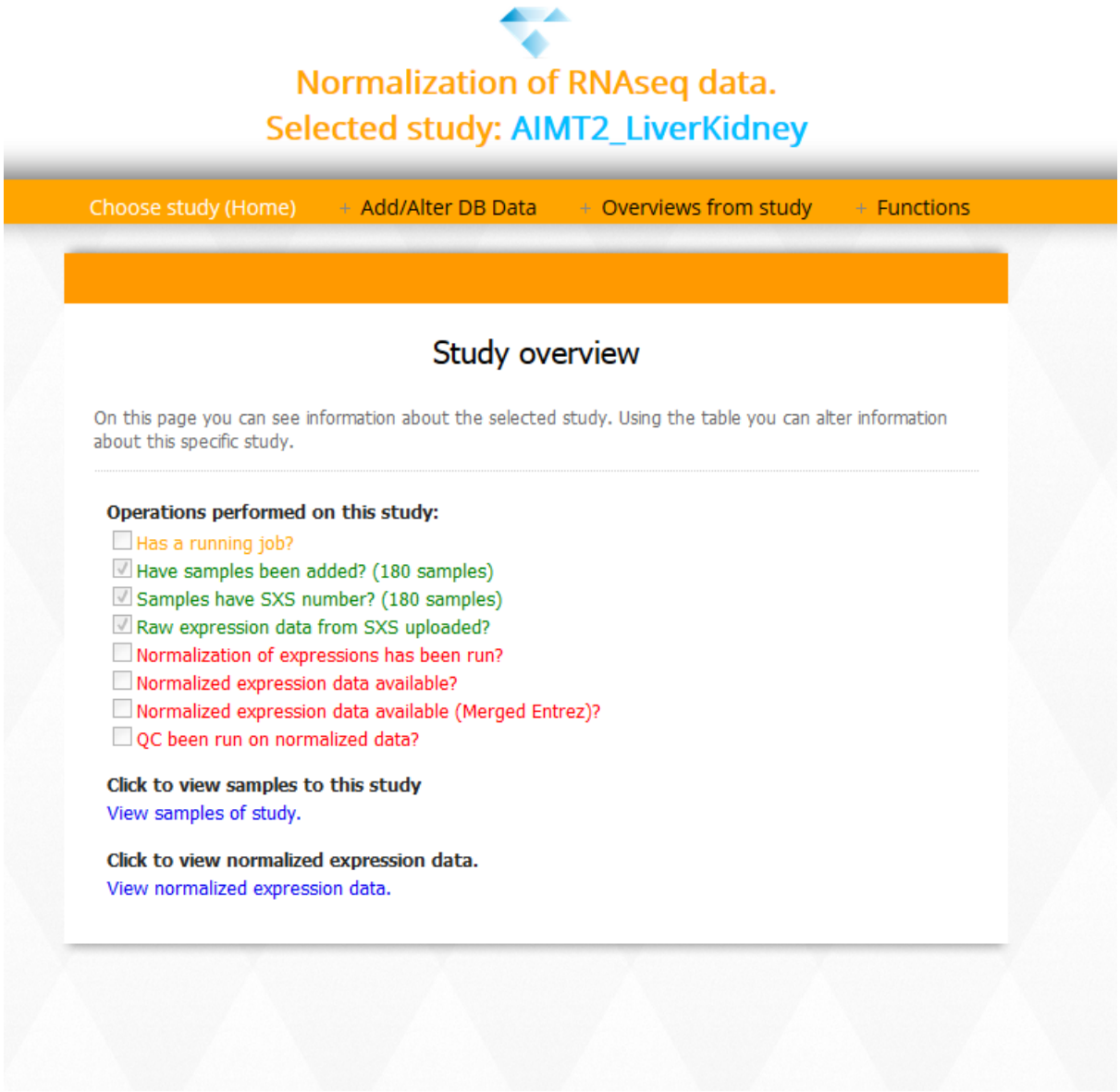

Figure 7, impression of the studyOverview.php page.

3.1.4. Page – dataOverviews

This is an impression for the various pages which allow for viewing and altering data. This page houses a responsive CRUD table which directly, with security in mind, is capable of performing SQL actions on the database. The table uses the JQuery JTable library.

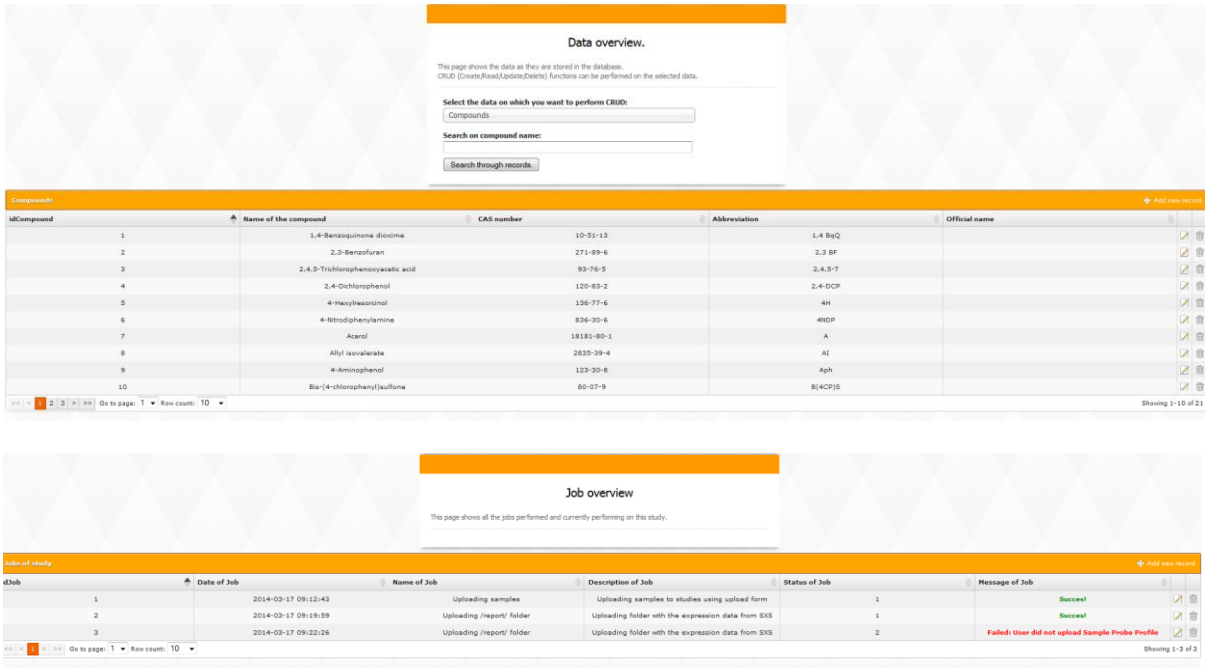

Figure 8, impression of the Job and compounds data table. This table shows the various jobs performed by the selected study. It uses coloring (Green/Red) to quickly show the status of the jobs. If a job has failed, it is shown in red. On success it is shown in green.

3.1.5. Page – uploadFiles

This is an impression for the pages used to upload files. This specific page is used to upload a tab delimited file with samples and information about these samples. The columns can be modular and dynamically be defined by selecting the “dataType” for each column. These “dataTypes” correspond to entries in the database with information about the “dataTypes”. This allows for flexible formats to be uploaded in a modular fashion.

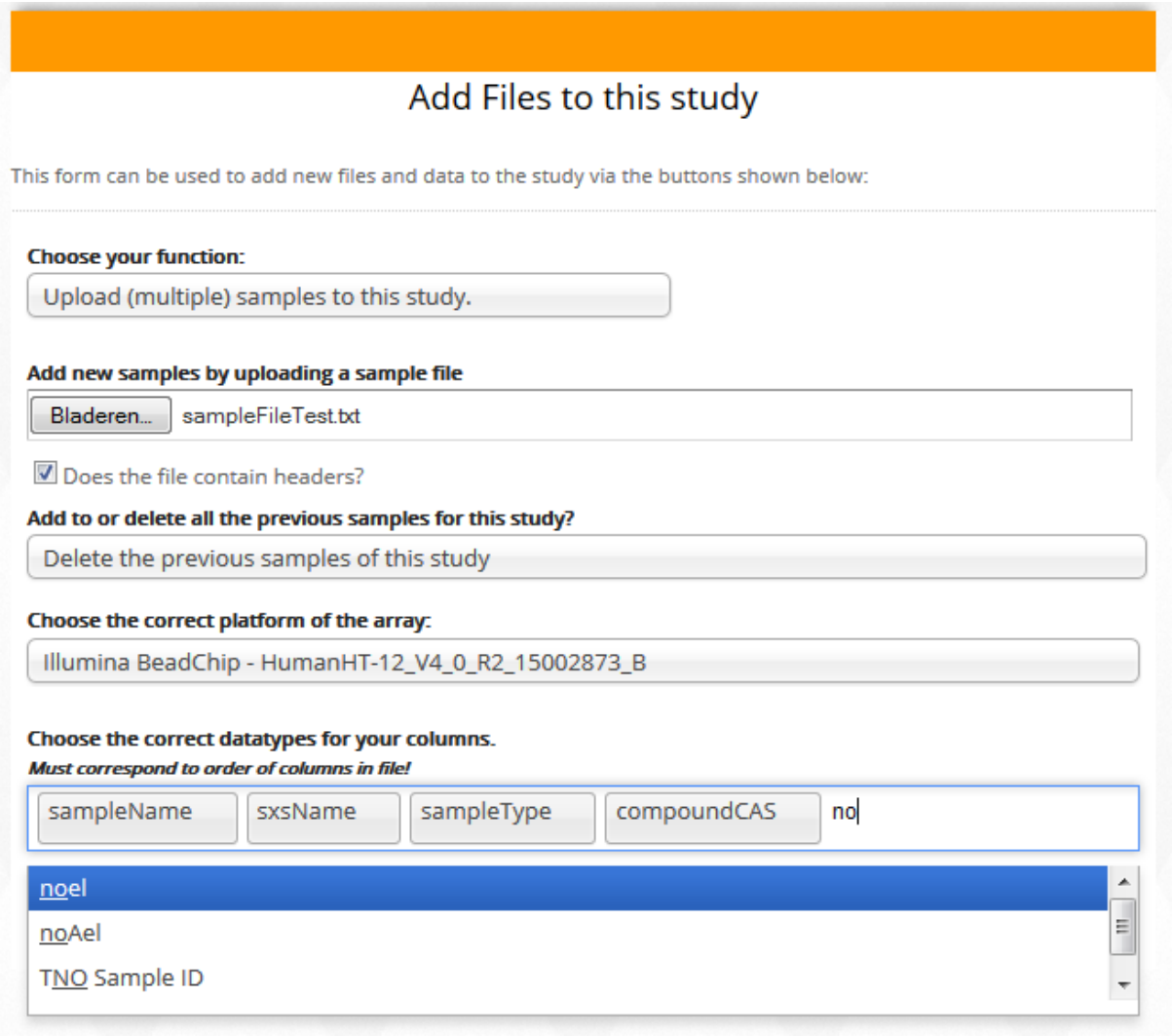

Figure 9, impression of page to upload samples to the DB. This design is also used to upload other files.

3.1.6. Page – Illumina Normalization

This is an impression of the page used to perform normalization of the uploaded Illumina omics files for the selected study. The user adjust the methods and steps used in the Illumina Normalization pipeline using the dropdown menus. A subset can also be selected to select samples on which the normalization is run.

The user can also choose the use the checkboxes to disable options, this also hides that option in the page. The user can also select a subset of samples to perform the statistics on. The user can also chose to set groups during statistics, these clusters will be groups will be used to cluster similar samples together during statistics. The coloring of the tables has been copied from the original ArrayAnalysis module.

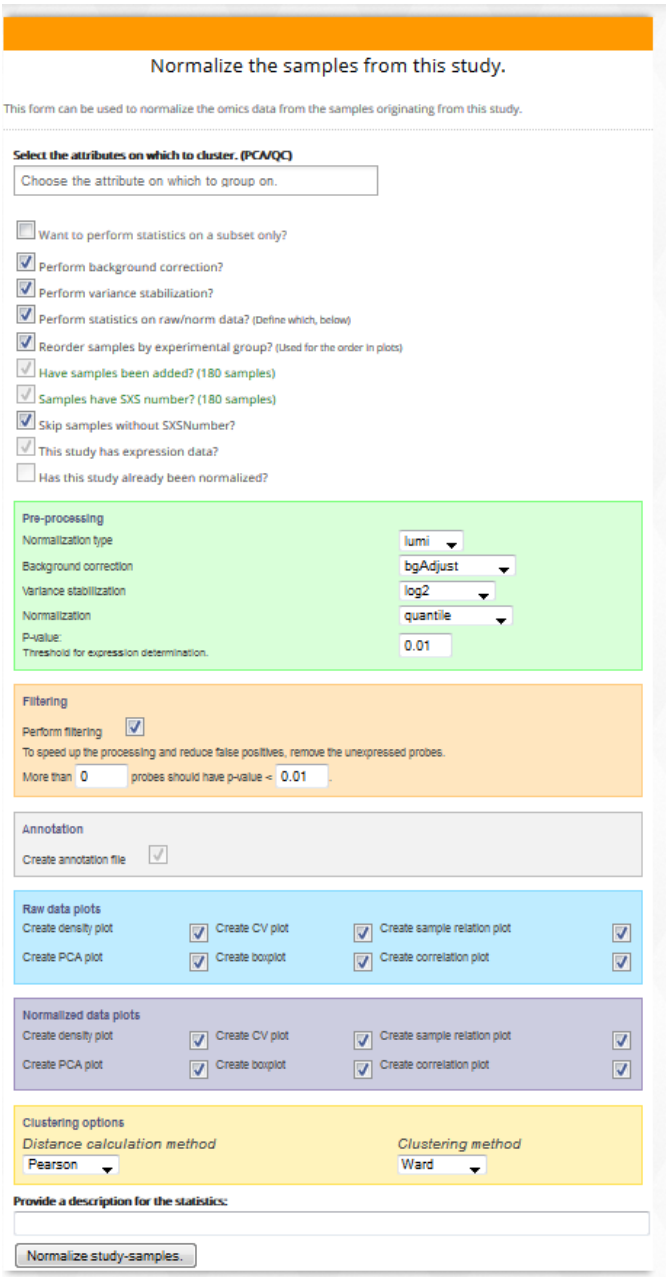

Figure 10, impression of the Illumina normalization options.

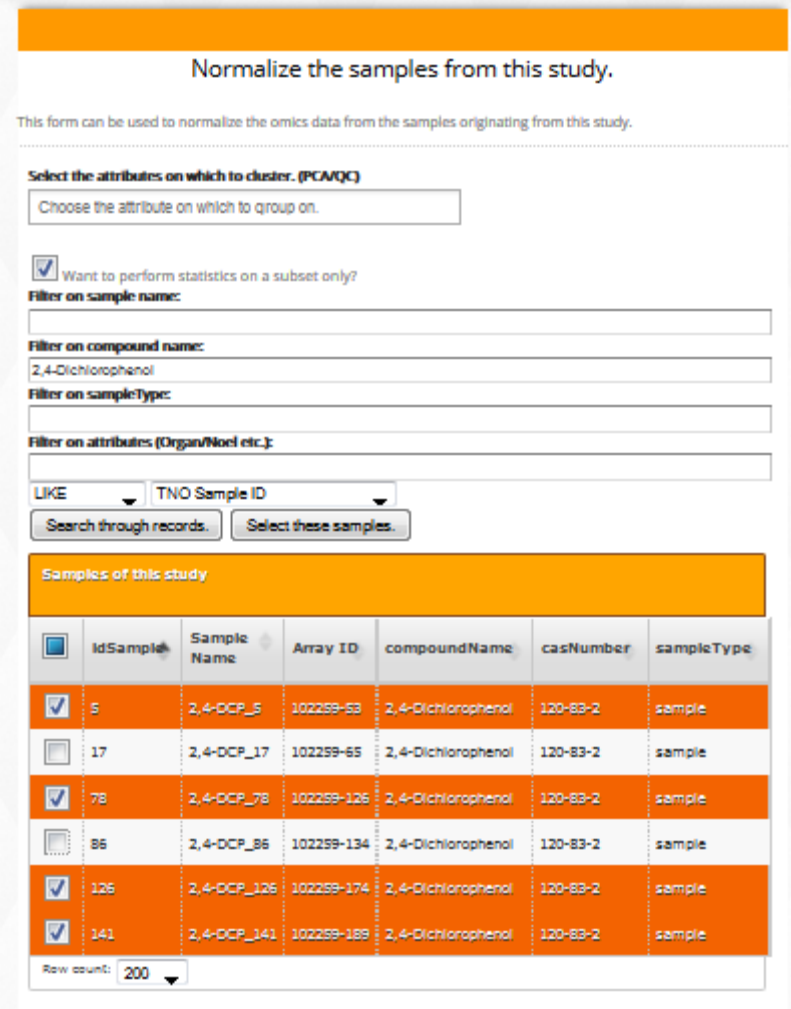

Figure 11, impression of the subset creation of samples.

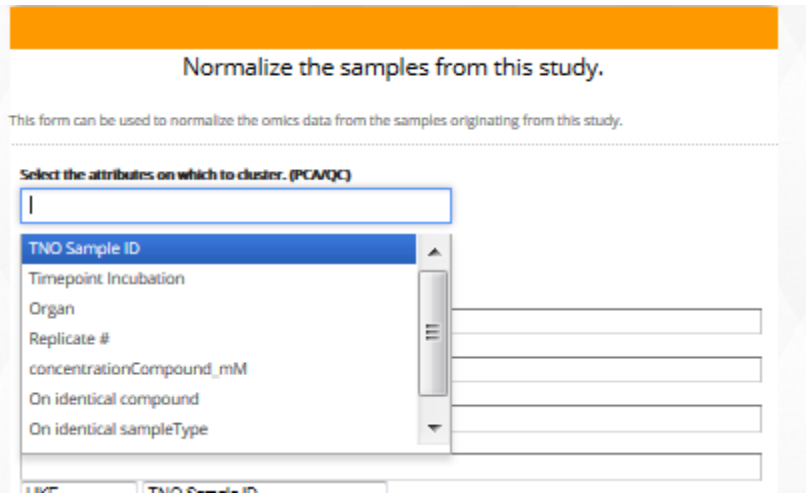

Figure 12, impression of the method of selecting which clusters should be made the samples during statistics. E.g. if "Organ" is chosen, all compounds tested on a specific organ (kidney/liver etc.) will be clustered together during statistics.

4. Technical documentation

The Illumina pipeline uses the following packages: limma, ALL, bioDist, gplots, annotate, arrayQualityMetrics, lumi, org.Hs.eg.db, org.Mm.eg.db, org.Rn.eg.db, RMySQL, optparser.

The Affymetrix pipeline uses the following packages: affy, affycomp, affyPLM, affypdnn, bioDist, simpleaffy, affyQCReport, plier, gdata, gplots, yaqcaffy, RMySQLm, optparser.

The SQL language was used to provide back-end MySQL 5.1.73-0ubuntu0.10.04.1 database support in combination with RMySQL 0.9-3 for the querying of this database from R.

OpenCPU 1.2.2. is used to provide a web API for interfacing with the R pipeline.

The front-end of the web-interface has been created in HTML5, CSS with added functionality provided by JavaScript and the following JavaScript libraries: jquery-1.11.0.js, jquery-ui.js, chosen.jquery.js, chosen.order.js, jquery.jtable.js, jquery.tablesorter.js, jquery.tablesorter.pager.js. (All JavaScript/JQuery libraries where the most current version as of date of writing)

The back-end of the web-interface has been created using PHP 5.4.26 and HTTP is handled by Apache 2.4.7.

All scripts and needed materials can be downloaded and viewed in the appropriate GitHub repositories: localIlluminaNormalization[3], webIllu-AffyNormalization[4] and OpenCPUIlluminaNormalization[5].

4.1. Installation

- Upload the scripts from GitHub, using the same folder structure, on a web-server running:
  - Apache *(Version >= 2.4 + HTML5 support)*
  - PHP5
  - MySQL *(Version >= 5.0)*
  - R *(Version >= 2.0, install optparse and RMySQL manually)*
- *(Optional) Set correct permissions on files and folders.*
- Create the database using the SQL code provided in the /sql/ folder. (Change the username/password)
- *(Optional) Use the provided sample data to fill the database with options from the /sql/sampleData.sql file.*
- Change the security usernames/password and other logins and paths in the /logic/config.php file.
- Change the MySQL user/password for the RMySQL functions and other configuration in the /R/config.R file.

4.2. ERD relation database

This is the ERD for the database that houses all data from the Illumina normalization pipeline and user uploaded samples and files for use in this pipeline. This ERD has been designed in MySQL workbench 6.0. The SQL code is available on GitHub[4].

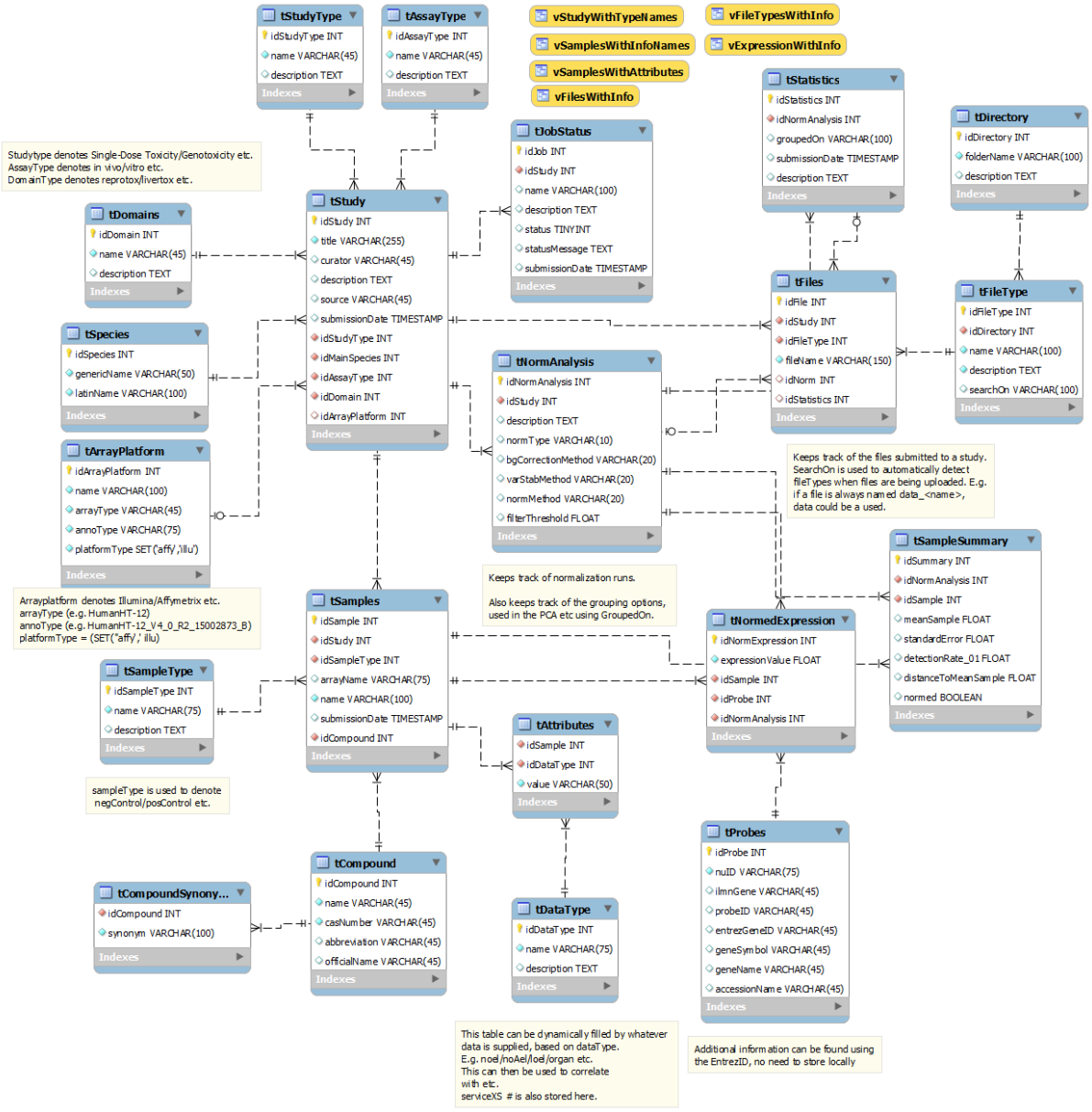

Figure 13, ERD for the storage of data from the normalization pipeline and user uploaded samples and information.

4.3. Files & formats

The web-interface generates multiple files required for additional functions during the normalization of the gene-expressions. Some file are also required to be uploaded by the user such as a linker-file containing the names of the samples as inserted into the DB and the name of the samples as they are designated on the assays. The format of these files can be found in this chapter.

4.3.1. descriptionFile.txt

If normalization (and statistics) is performed, this file will be created in the output directory of the files containing the normalized files and objects, e.g. /var/www/normdb/data/<idStudy>_<studyName>/expressions/normed/<idNorm>/

If normalization has already been run and statistics are done on the existing normalized R Lumi/Affybatch Object, the description file will be stored in the output directory of the statistics run, e.g. /var/www/normdb/data/<idStudy>_<studyName>/statistics/<idStat>/

This file contains the samples that will be used in the normalization process, if this file contains less samples than the raw data containing the expressions, a subset will be made of the raw data which only contains the samples provided in the description file. The groups of the

The format of this file is as such:

- arrayName
  - This is the name of the sample as designated on the microarray.
- sampleName
  - This is the name of the sample as the user has named it, this name will be shown during statistics.
- Group
  - This is the grouping of the samples during statistics, similarly named groups will be ordered together.

Table 1, example data for the descriptionFile.txt used for identifying the samples to the array.

arrayName	sampleName	Group
102259-49	DMSO_1	Liver
102259-50	DP_2	kidney
102259-51	DP_3	Liver
102259-52	T(PG)BE_4	Liver
102259-53	2,4-DCP_5	Liver
102259-54	2,3 BF_6	kidney
102259-55	DMSO_7	Liver
102259-56	1,4 BqQ_8	kidney

4.3.2. Logfile

A logfile will be kept in the output directory of the normalized files, this logfile stored all the STDOUT and STDERR messages of the R pipeline using sink(). The creation of the logfile can be enabled/disabled by using the `--createLog` parameter.

Excerpt log output:

Required packages successfully loaded.

Reading the description
file:/var/www/normdb//data/1_AIMT2/expressionData/normed/1/descriptionFile.txt

Description file loaded successfully.

Loading sample probe
profile:/var/www/normdb//data/1_AIMT2/expressionData/raw//Sample_Probe_Profile_102259-2.txt
Perform Quality Control assessment of the LumiBatch object ...
Directly converting probe sequence to nuIDs ...

Successfully loaded the Sample Probe Profile.

Checking if description data is valid for the given sample probe profile.

Description data is valid.

Re-ordering raw Sample Probe Profile per group defined in the description file.

Re-ordering successful.

Performing background correction (bgAdjust) on the Sample Probe Profile using the Control Probe Profile: /var/www/normdb//data/1_AIMT2/expressionData/raw//Control_Probe_Profile_102259-2.txt

Loading Control Probe Profile:Control_Probe_Profile_102259-2.txt

Combining Control data with Sample data.
Inputting the data ...

Normaling (lumi) the raw Sample Probe Profile using background
correction:Control_Probe_Profile_102259-2.txt
Background Correction: bgAdjust
Variance Stabilizing Transform method: log2
Normalization method: quantile

4.3.1. statSubset.txt

If statistics should be run on a subset of the samples used in the normalization process, the statSubSet file will be created in the output directory of the statistics run, e.g. /var/www/normdb/data/<idStudy>_<studyName>/statistics/<idStat>/

This file will contain the samples that will be extracted from the rawData/normData Lumi/AffyBatch objects and be used in the plotting of various quality assessments.

The format of this file is as such:

- sampleName

- This is the name of the sample as the user has named it, this name will be shown during statistics. If normalization has already been run, this is also the name that is used in the Lumi/AffyBatch Object.

- Group
  - This is the grouping of the samples during statistics, similarly named groups will be ordered together.

Table 2, example data for the descriptionFile.txt used for identifying the samples to the array.

sampleName	Group
DMSO_1	Liver
DP_2	kidney
DP_3	Liver
T(PG)BE_4	Liver
2,4-DCP_5	Liver
2,3 BF_6	kidney
DMSO_7	Liver
1,4 BqQ_8	kidney

4.3.4. File containing the samples

The file that can be used to upload samples to the selected study does not to adhere to a specific format as the user needs to manually select the semantic of each column, however there are few columns that are required in order to successfully upload samples. The required fields, in no specific order, in this file are:

- Name of the sample
- CAS number of the compound
- Name of the compound

Optionally, the name of the sample on the array can also be given and which will subsequently be stored in the appropriate field in the database. (arrayName)

4.3.5. File to convert array names to sample names

If the user did not specify the designation of the sample on the microarray, it can be done when uploading the raw gene-expressions files. This file needs to contain the two (2) columns in this exact order:

Table 3, example input of the file to convert the sample names into arraynames.

sampleName	arrayName
DMSO_1	102259-49
DP_2	102259-50
DP_3	102259-51
T(PG)BE_4	102259-52
2,4-DCP_5	102259-53

The sample name needs to correspond with the name of the samples given when uploading the samples.

4.3.6. Folder Structure

Order	Folder				Function
1.	/data/				Main /data/ folder
1.1.		<idStudy>_<studyTitle>/			Folder for all the contents of a study
1.1.1.			sampleAnnotation/		Stores the file(s) used to upload the samples
1.1.2.			expressionData/		Stores the gene expression data
1.1.2.1.				/raw/	Stores the raw expression files
1.1.2.2.				/normed/	Stores the normed expression files (From pipeline)
1.1.3.			statistics/		Stores the statistical plots
1.1.3.1.				<idStatistics>/	Stores the statistical plots per study
1.1.4.			Unknown/		Stores all the files not recognized on filename
1.1.4.1.				<idStudy>/	Stores all the files not recognized on filename per study

References

[1] Commonly attributed to: Illumina. GenomeStudio Software. Unknown. Web. Accessed 18th March 2014. SSRN: http://www.illumina.com/informatics/sequencing-microarray-data-analysis/genomestudio.ilmn

[2] Commonly attributed to: Illumina. GenomeStudio TM Gene Expression Module v1.0 November 2008. Web. Accessed 18th March 2014.
SSRN: http://supportres.illumina.com/documents/myillumina/c94519f7-9348-4308-a32f-b66ff3959e99/genomestudio_gx_module_v1.0_ug_11319121_reva.pdf

[3] van Riet, J. 2014 "Repository for the local version of Illumina Normalization Pipeline" https://github.com/J0bbie/localIlluminaNormalization

[4] van Riet, J. 2014 "Repository for the DIAMONDs integrated version of Illumina Normalization Pipeline" https://github.com/J0bbie/webIllu-AffyNormalization

[5] van Riet, J. 2014 "Repository for the OpenCPU version of Illumina Normalization Pipeline" https://github.com/J0bbie/OpenCPUIlluminaNormalization