Statistical analysis of *in vitro* screening for inhibitors of viral infection

Exploration of the results

Touret et al. (2020).

2020-04-13

Contents

Introduction

Data

Supplementary data tables

```
#### Directories ####
message("Directories and files")

dir <- c(data = "../data",
         results = "../results",
         figures = "figures")
dir.create(dir["results"], showWarnings = FALSE, recursive = TRUE)

## Data file
supTableFile <- file.path(dir["data"], "suppl-table_Touret-2020.xlsx")
```

```
#### Load data from Excel workbook ####
message("Loading data from excel workbook.")
supTable <- read.xlsx(file = supTableFile, sheetIndex = 1)
#supTable <- read_xlsx(path = supTableFile, sheet = 1, col_names = TRUE)
# dim(supTable)
# View(supTable)
# names(supTable)

colNames <- colnames(supTable)
colNames[1] <- "ID"
colnames(supTable) <- colNames

## Suppress the last row (NA)
supTable <- supTable[!is.na(supTable$ID), ]
# dim(supTable)

## Assign row names for convenience
# View(supTable)


## Extract plate number
supTable$plateNumber <- as.numeric(substr(supTable[, 1], start = 1, stop = 2))
# table(supTable$plateNumber)
plateNumbers <- unique(supTable$plateNumber)

## Assign a color to each molecule according to its plate number
plateColor <- rainbow(n = length(plateNumbers))
names(plateColor) <- unique(supTable$plateNumber)

supTable$color <- plateColor[supTable$plateNumber]
message("\tLoaded main table with ", nrow(supTable), " rows ")
```

The supplementary table downloaded from bioRxiv contains 1520 molecules.

Raw viability measurements

```
#### Load raw data ####
message("Loading raw data")
nbPlates <- 19
rowsPerPlate <- 8
columnsPerPlate <- 12

dataPerPlate <- list()

## Control 1: uninfected cells
cellControl <- data.frame(matrix(ncol = 8, nrow = nbPlates))
colnames(cellControl) <- LETTERS[1:rowsPerPlate]

## Control 2: untreated infected cells
virusControl <- data.frame(matrix(ncol = 6, nrow = nbPlates))
colnames(virusControl) <- LETTERS[3:rowsPerPlate]

## Prepare a table to store the raw data
inhibTable <- data.frame(matrix(ncol = 8, nrow = nbPlates*rowsPerPlate * columnsPerPlate))
colnames(inhibTable) <- c("ID",
                          "Plate",
                          "Row",
                          "Column",
                          "viability",
                          "cellControl",
                          "virusControl",
                          "chemicalName")

i <- 2 ## for quick test
for (i in 1:nbPlates) {
  message("\tLoading data from plate ", i)
  sheetName <- paste0("Plate", i)

  ## Raw measures
  # rawMeasures <- read.xlsx(file = supTableFile,
  #                          sheetName = sheetName,
  #                          rowIndex = 30:37,
  #                          colIndex = 2:13, header = FALSE)
  rawMeasures <- read_xlsx(path = supTableFile, col_names = FALSE,
                           sheet = sheetName,
                           range = "B30:M37", progress = FALSE)
  rawMeasures <- as.data.frame(rawMeasures)
  rownames(rawMeasures) <- LETTERS[1:nrow(rawMeasures)]
  colnames(rawMeasures) <- 1:ncol(rawMeasures)
  # dim(rawMeasures)
  # View(rawMeasures)

  ## Extract control values
  cellControl[i, ] <- as.vector(rawMeasures[,1])
  virusControl[i, ] <- as.vector(rawMeasures[3:8,12])
  plateVC <- mean(unlist(virusControl[i, ]))
  plateCC <- mean(unlist(cellControl[i, ]))
```

```
  ## Extract all values
  r <- 1
  for (r in 1:rowsPerPlate) {
    currentRowName <- LETTERS[r]
    currentValues <- unlist(rawMeasures[currentRowName,])
    id <- paste0(sprintf("%02d",i),
                 currentRowName,
                 sprintf("%02d",1:columnsPerPlate))

    ## Compute the start index for the data table
    startIndex <- (i - 1) * (rowsPerPlate * columnsPerPlate) + (r - 1) * columnsPerPlate + 1
    # message(cat("\t\tIDs\t",  startIndex, id))
    indices <- startIndex:(startIndex + columnsPerPlate - 1)
    # length(indices)
    inhibTable[indices, "ID"] <- id
    inhibTable[indices, "Plate"] <- i
    inhibTable[indices, "Row"] <- currentRowName
    inhibTable[indices, "Column"] <- 1:columnsPerPlate
    inhibTable[indices, "viability"] <- currentValues
    inhibTable[indices, "virusControl"] <- plateVC
    inhibTable[indices, "cellControl"] <- plateCC
  }

  dataPerPlate[[i]] <- list()
  dataPerPlate[[i]][["rawMeasures"]] <- rawMeasures
}

# dim(inhibTable)
# names(inhibTable)
# View(inhibTable)
# View(dataPerPlate)
# View(dataPerPlate[[1]][["rawMeasures"]])
# table(inhibTable$Row, inhibTable$Column) ## Check that there are 19 entries for each plate position

## Use the plate well ID as rowname
rownames(inhibTable) <- inhibTable$ID

## Check consistency between IDs in supplementary Touret Table 1
##  and those created here
touretIDs <- unlist(supTable$ID)
# length(touretIDs)
inhibIDs <- inhibTable$ID
# length(inhibIDs)

## Retrieve the molecule names from Table 1 of the bioRxiv workbook
inhibTable$chemicalName <- NA
inhibTable[inhibTable$ID %in% touretIDs, "chemicalName"] <-
  as.vector(supTable$Chemical.name)

## Cell control: uninfected cells
cellControlIndices <- inhibTable$Column == 1
inhibTable[cellControlIndices, "chemicalName"] <- "uninfected"
```

```
## Virus control: infected cells, no treatment
virusControlIndices <- (inhibTable$Column == 12) & (inhibTable$Row %in% LETTERS[3:8])
# table(virusControlIndices)
inhibTable[virusControlIndices, "chemicalName"] <- "infected no treatment"

## Define the treatment type
wellType <- NA
wellType[cellControlIndices] <- "cellCtl"
wellType[virusControlIndices] <- "virusCtl"

## All the other ones are treated with a given molecule
wellType[!(virusControlIndices | cellControlIndices)] <- "treated"

inhibTable[, "wellType"] <- wellType


kable(t(table(inhibTable$wellType)), caption = "Well types. cellCtl: no infection; virusCtl: infection
```

Table 1: Well types. cellCtl: no infection; virusCtl: infection without treatment; treated: inhected and treated with one molecule

cellCtl	treated	virusCtl
152	1558	114

The raw data contains 19 plates with 8 rows (indiced A to H) and 12 columns (indiced from 1 to 12.)

The raw data consists of viability measurements in cell cultures.

Distribution of raw viability values

```
#### Distribution of raw measurements ####
classInterval <- 500
# xmin <- floor(min(inhibTable$viability)/classInterval) * classInterval
xmin <- 0 ## Intently start the scale at0 to show the remnant viability
xmax <- ceiling(max(inhibTable$viability)/classInterval) * classInterval
breaks = seq(from = xmin, to = xmax, by = classInterval)
# range(inhibTable$viability)

par(mfrow = c(3, 1))
par(mar = c(2,5,3,1))
hist(inhibTable[wellType == "cellCtl", "viability"],
     main = "Uninfected (cell control)",
     xlab = "Viability",
     ylab = "Number of plate wells",
     las = 1,
     breaks = breaks, col = "palegreen", border = "palegreen")

hist(inhibTable[wellType == "virusCtl", "viability"],
     main = "Infected, no treatment (virus control))",
     xlab = "Viability",
     ylab = "Number of plate wells",
     las = 1,
```

```
     breaks = breaks, col = "orange", border = "orange")

hist(inhibTable[wellType == "treated", "viability"],
     main = "Treated cells",
     xlab = "Viability",
     ylab = "Number of plate wells",
     las = 1,
     breaks = breaks, col = "#AACCFF", border = "#AACCFF")
```

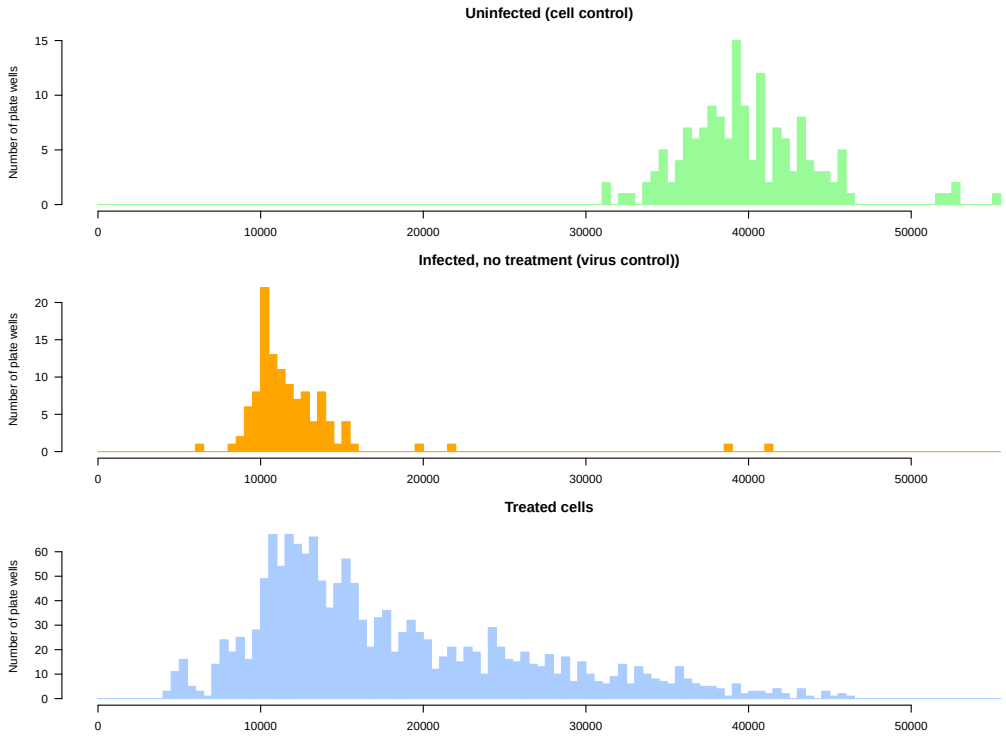

Figure 1: Distributions of the raw viability measures. Top: uninfected cells. Middle: infected cells without treatment. Bottom: infected cells treated with a specific molecule.

```
par(par.ori)
```

- ***Cell control***.
  - The top panel (green) shows the distribution of viability measurements in controls cultures, where the cells were neither infected by the virus nor treated with a drug.
  - Each plate contains 8 wells with uninfected cells (total = 152).
- ***Virus control***.
  - The middle panel (orange) shows the distribution of viability measurements in infected cells without drug treatment
  - The virus control was performed on 6 wells per plate, total = 114).
- ***Treated cells***.
  - The bottom distribution (pale blue) shows the viability values for cells infected and treated with a given drug.
  - Note that each drug was tested on a single well (no replicates). Indeed, in order to face the COVID-19 emergency, the study attempted to test as fast as possible a wide range of molecules.

This first screening was thus performed without replicates. This has to be taken into account for the normalisation, which should be done with no estimation of the variance for the individual drugs.

– The distribution is strongly asymmetrical, and seems bi- or multi-modal. This distribution can be considered as a mixture between different distributions;

  * all the drugs that have no inhibitory effect (and are thus expected to have a viability similar to the virus control);
  * various drugs that inhibit the action of the virus, each one with its specific level of inhibition. This probably corresponds to the widely dispersed values above the bulk of distribution (and above the virus control distribution)

Arbidol treatment

A treatment with 10µM Arbidol – a broad-spectrum antiviral – was used as control, with duplicate test in 2 wells per plate.

```
#### Select arbidol duplicates as plate-wise milestones ####
arbidolWells <- (inhibTable$Column == 12) & (inhibTable$Row %in% c("A", "B"))
inhibTable[arbidolWells, "chemicalName"] <- "Arbidol"
# inhibTable[arbidolWells, c("ID", "chemicalName")]
```

Raw viability boxplots

```
#### Plate-wise colors ####

## Assign a color to each plate
## A trick: we alternate the colors of the rainbow in order
## to see the contrast between successive plates
platePalette <- rainbow(n = length(plateNumbers))
plateColor <- vector(length = nbPlates)
oddIndices <- seq(1, nbPlates, 2)
evenIndices <- seq(2, nbPlates, 2)
plateColor[oddIndices] <- platePalette[1:length(oddIndices)]
plateColor[evenIndices] <- platePalette[(length(oddIndices) + 1):nbPlates]
names(plateColor) <- 1:nbPlates

## Assign a color to each result according to its plate
inhibTable$color <- plateColor[inhibTable$Plate]
# table(inhibTable$color) ## Check that each plate has 96 wells
```

```
#### Extract raw viability measures per plate for arbidol ####

arbidoltV <- inhibTable[arbidolWells, c("Plate", "viability")]
```

```
#### Boxplots per plate  ####

rawBoxPlot <- boxplot(viability ~ Plate + wellType,
        main = "Raw viability per plate",
        data = inhibTable,
        las = 1, col = plateColor,
        xlab = "Viability (raw)",
        cex.axis = 0.5, cex = 0.5,
        horizontal = TRUE
        )
```

```
abline(v = seq(from = 0, to = max(inhibTable$viability), by = 2000), col = "#EEEEEE", lty = "dashed")
abline(v = seq(from = 0, to = max(inhibTable$viability), by = 10000), col = "grey")
# View(rawBoxPlot)

## Add points to denote the arbidol controls
stripchart(viability ~ Plate, vertical = FALSE, data = inhibTable[arbidolWells, ],
    method = "jitter", add = TRUE, pch = 20, cex = 0.7, col = 'orange')


legend("topright", legend = names(plateColor),
       title = "Plate", fill = plateColor, cex = 0.8)
```

The boxplot of the raw viability measurements highlights a plate effect, for the treated cells (middle barplots) but also for the untreated virus control (top boxplots) and uninfected cell control (bottom boxplots).

- **Treated:**
  - The medians and interquartile ranges show strong variations between plates.
  - In particular, plate 1 (in red) has a the smallest median and a remarkably compact interquartile range. There are many statistical outliers (empty circles) in this plate, which might correspond to the molecules having a significant inhibitory effect.
  - In contrast, plates 11 to 15 show a high median and a wide dispersion of the viability measures, and there is not a single statistical outlier.
- **Virus control**
  - Not surprisingly, untreated infected cells generally gave a very small viability, with small variations (very compact interquartile rectangles)
  - There is an obvious problem for plate 17, which shows a broad range of values, with a third quartile falling in the range of the uninfected cells. This suggests a problem with at least 2 of the 6 replicates (missed infection ?). Noticeably, the median falls in the same range as for the virus control of the other plates.
- **Cell control**
  - The cell control performed as expected in all the plates, with high viability values.
  - Note however that the viability measurements show inter-plate variations, with median values ranging from ~37,000 to ~53,000.

Importantly, there is a consistency between the inter-plate differences observed for virus control, treated cells and cell control, respectively. For example, plate 2 whose consistently higher value than the other plates for the three types of wells. This highlights the importances to perform a plate-wise standardisation.

Log2 viability boxplots

We performed a log2 transformation of the raw viability measures in order to normalise them.

```
#### Boxplots of log2_transformed viability per plate ####
inhibTable$log2V <- log2(inhibTable$viability)

## Box plot per plate and well type
boxplot(log2V ~ Plate + wellType,
        main = "log2-transformed viability",
        data = inhibTable,
        las = 1, col = plateColor,
        xlab = "log2(V)",
        cex.axis = 0.5, cex = 0.5,
        horizontal = TRUE)
```

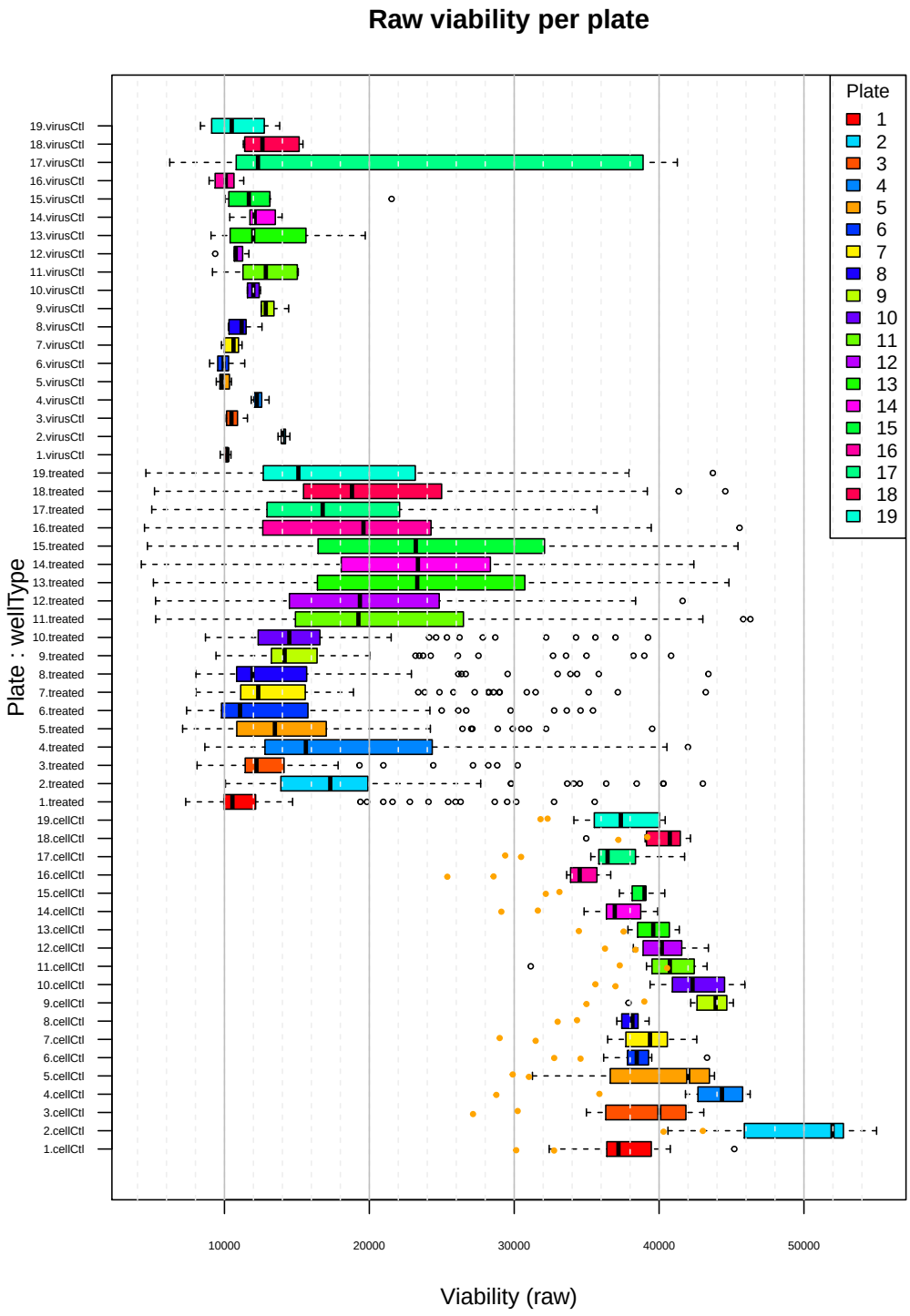

Figure 2: Distribution of raw viability values per plate. Top virus control (untreated infected cells); middle: treated cells; bottom: cell control (uninfected). Orange dots denote the arbidol control duplicates.

```
abline(v = seq(from = 12, to = max(inhibTable$log2V), by = 0.2), col = "#EEEEEE", lty = "dashed")
abline(v = seq(from = 12, to = max(inhibTable$log2V), by = 1), col = "grey")
legend("topright", legend = names(plateColor),
       title = "Plate", fill = plateColor, cex = 0.8)

## Add points to denote the arbidol controls
stripchart(log2V ~ Plate, vertical = FALSE, data = inhibTable[arbidolWells, ],
    method = "jitter", add = TRUE, pch = 20, cex = 0.7, col = 'orange')
```

```
par(par.ori)
```

The barplots of log2-transformed viability measures show yet another potential plate bias: in plates 11 to 19, several molecules are associated to a much smaller viability than in any of the untreated cells. This might reflect a cytotoxic effect of the drug that would enforce the viral infection, but there is a priori no reason to expect such effects to be concentrated on the last plates.

Note: I (JvH) think this should be reported to the company that produces these plates

Plate-wise standardisation

Two-points scaling: defining a relative viability (v_r)

Taking into account the above-reported results, we apply the following procedure to standardise the individual viability measures.

1. For each plate, we define two standard values:
   - V_c is the median viability of the 8 cell controls (uninfected cells)
   - V_v is the median viability of the 6 virus controls (infected untreated cells)

   These two values are deliberately estimated with the median measurement of the control replicate, in order to avoid the effect of outliers as denoted for the virus control of plate 17.

2. A **relative viability index** v_r is computed for each treatment as follows.

$$v_r = \frac{v - V_c}{V_v - V_c} \cdot 100$$

   - v_r provides a viability measurement on a scale where 0 corresponds to the median of infected untreated cells, and 100 to the median of uninfected cells.
   - Note that V_c values lower than 0 denote treatments with a lower viability than the untreated virus infection. This might result from a cytotoxic effect of the drug, or from a plate bias.
   - V_c values can in principle also take values higher than 100, denoting a highly efficient treatment.

```
#### Compute relative viability ####

## Plate-wise virus control
vcMed <- as.vector(by(data = inhibTable[wellType == "virusCtl", "viability"],
                      INDICES = inhibTable[wellType == "virusCtl", "Plate"],
                      FUN = median))
names(vcMed) <- 1:nbPlates

## Plate-wise cell control
ccMed <- as.vector(by(data = inhibTable[wellType == "cellCtl", "viability"],
                      INDICES = inhibTable[wellType == "cellCtl", "Plate"],
                      FUN = median))
names(ccMed) <- 1:nbPlates
```

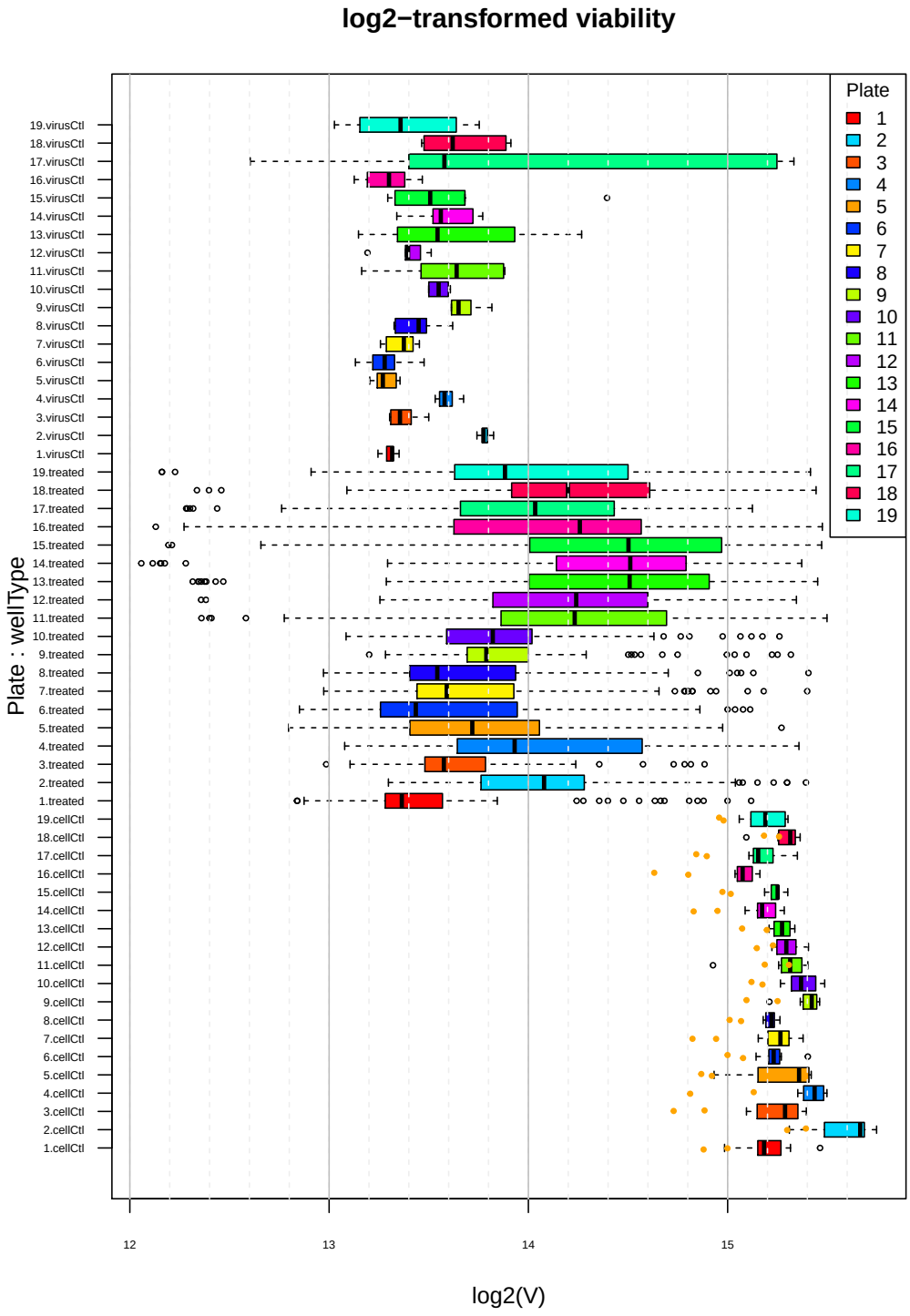

Figure 3: Distribution of log2-transformed viability values per plate. Top virus control (untreated infected cells); middle: treated cells; bottom: cell control (uninfected). Orange dots denote the arbidol control duplicates.

```
## Relative viability
inhibTable$vcMed <- vcMed[inhibTable$Plate]
inhibTable$ccMed <- ccMed[inhibTable$Plate]
inhibTable$Vr <- 100 * (inhibTable$viability - inhibTable$vcMed) /
  (inhibTable$ccMed - inhibTable$vcMed)
```

Relative viability boxplots

```
#### Boxplots of relative viability per plate ####

## Box plot per plate and well type
boxplot(Vr ~ Plate + wellType,
        main = "Relative viability",
        data = inhibTable,
        las = 1, col = plateColor,
        xlab = "Vr",
        cex.axis = 0.5, cex = 0.5,
        horizontal = TRUE)
abline(v = seq(from = 0, to = max(inhibTable$Vr), by = 5), col = "#EEEEEE", lty = "dashed")
abline(v = seq(from = 0, to = max(inhibTable$Vr), by = 25), col = "grey")
legend("topright", legend = names(plateColor),
       title = "Plate", fill = plateColor, cex = 0.8)

## Add points to denote the arbidol controls
stripchart(Vr ~ Plate, vertical = FALSE, data = inhibTable[arbidolWells, ],
    method = "jitter", add = TRUE, pch = 20, cex = 0.7, col = 'orange')
```

```
par(par.ori)
```

The box plots show that the relative viability already has a normalizing effect by positioning each treatment between the virus and cell control of its own plate.

- The virus controls are well regrouped in the range of smaller v_r values (the third quartiles all fall below 15, except for plate).
- The cell controls occupy the high range (their first quartile is higher than 80) and are quite compactly grouped around 100.

However, we still observe a systematic plate effect in plates 11 to 19:

- their median is mich higher than for the plates 1 to 10;
- they also show a much wider inter-quartile rectangle, denoting a very high variance between all the viability measurements on this plate;
- this higher variance is even visible in the virus control (untreated infected cells), and it is thus unlikely that it results from the particular molecules sampled on this second half of the plates.

We thus need a way to perform a between-plates standardisation of the variance.

Relative viability plots

```
yRange <- c(floor(min(inhibTable$Vr)), ceiling(max(inhibTable$Vr)))

par(mfrow = c(4,1))
plot(inhibTable[wellType == "virusCtl", "Vr"],
     main = "Virus control (infected, untreated)",
     xlab = "Replicates, sorted per plate",
```

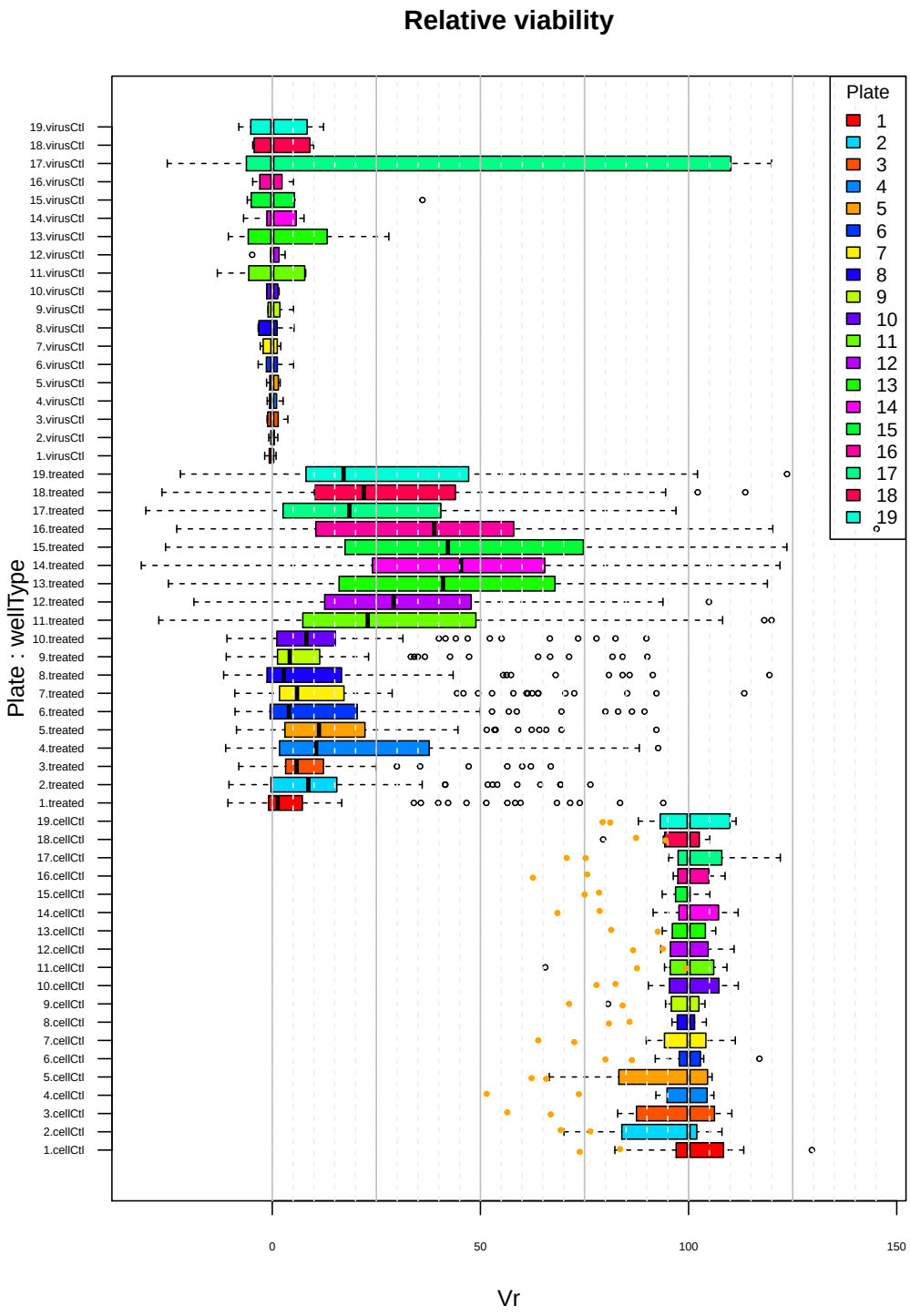

Figure 4: Distribution of relative viability (Vr) values per plate. Top virus control (untreated infected cells); middle: treated cells; bottom: cell control (uninfected). Orange dots denote the arbidol control duplicates.

```
     ylab = "relative viability",
     col = inhibTable[wellType == "virusCtl", "color"],
     ylim = yRange,
     xlim = c(0, (nbPlates * 6 * 1.05)),
     panel.first = c(abline(h = 0),
                     abline(h = 100),
                     abline(h = seq(10,90, 20), lty = "dotted"),
                     abline(v = (1:19) * 6, col = "grey")
                     ),
     las = 1,
     cex = 0.5
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)
plot(inhibTable[wellType == "treated", "Vr"],
     main = "Relative viability (Vr)",
     xlab = "Molecules",
     ylab = "relative viability",
     col = inhibTable[wellType == "treated", "color"],
     ylim = yRange,
     xlim = c(0, (nbPlates * 80 * 1.05)),
     panel.first = c(abline(h = 0),
                     abline(h = 100),
                     abline(h = seq(10,90, 20), lty = "dotted"),
                     abline(v = (1:19) * 80, col = "grey")
                     ),
     las = 1,
     cex = 0.5
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)
plot(inhibTable[wellType == "cellCtl", "Vr"],
     main = "Cell control (uninfected)",
     xlab = "Replicates, sorted per plate",
     ylab = "relative viability",
     col = inhibTable[wellType == "cellCtl", "color"],
     ylim = yRange,
     xlim = c(0, (nbPlates * 8 * 1.05)),
     panel.first = c(abline(h = 0),
                     abline(h = 100),
                     abline(h = seq(10,90, 20), lty = "dotted"),
                     abline(v = (1:19) * 8, col = "grey")
                     ),
     las = 1,
     cex = 0.5
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
```

```
       cex = 0.7)
# names(supTable)
VrRank <- order(inhibTable$Vr, decreasing = TRUE)
plot(inhibTable[VrRank, "Vr"],
     main = "Ranked relative viability values",
     xlab = "Molecules (ranked by relative viability)",
     ylab = "relative viability",
     col = inhibTable[VrRank, "color"],
     cex = 0.5,
     panel.first = grid(),
     xlim = c(0, length(VrRank) * 1.05)
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.4)
```

```
par(mfrow = c(1,1))
```

IQR-based standardisation

Plate-wise IQR-standardised viability

We perform a plate-wise normalisation with a centering on the median and a scaling to compensate for the plate-wise variance effect denoted above. To this purpose, we compute z-scores from the original value.

- centering: substract an estimator of the plate-wise mean ;
- scaling: divide by an estimator of the plate-wise standard deviation ($\hat{\sigma}_i$)

$$z_{c,i} = \frac{V_c - \hat{\mu}_i}{\hat{\sigma}_i}$$

where

- V_c is the viability for molecule c;
- $\hat{\mu}_i$ is the estimate for the mean viability of all the treated cells in plate i;
- $\hat{\sigma}_i$ is the estimate for the standard deviation of all the treated cells in plate i;

In classical statistics, the estimators of centrality and dispersion are derived from the sample mean and standard deviation, respectively:

- the population mean is used as maximum likelihood estimator of the population: $\hat{\mu} = \bar{x}$
- the population standard deviation (σ) is estimated with the sample standard deviation, corrected for the systematic bias: $\hat{\sigma} = \sqrt{n/(n-1)} \cdot s$)

However, we must be careful because each plate supposedly contain a mixture of inactive (no inhibitory effect) and active (inhibitory) molecules. The previous histograms and box plots show that these inhibitory molecules appear as statistiacal outliers (with very high viability values) and would thus strongly bias the estimation of the background variance (the variance due to fluctuations in absence of treatment).

One possibility would be to use the standard deviation of the virus control to this purpose, but this would leat to instable estimators, since they would be based on 6 points per plate. In addition, the boxplots show that the variance among treated cells is higher than the virus control, suggesting some generic effect of the treatments.

Another strategy is to consider that the variance (and standard deviation) can be estimated from the bulk of treated cell viability measures themselves, and to use **robust estimators** of the central tendency (i.e. the plate-wise median) and dispersion (i.e. the plate-wise interquartile range).

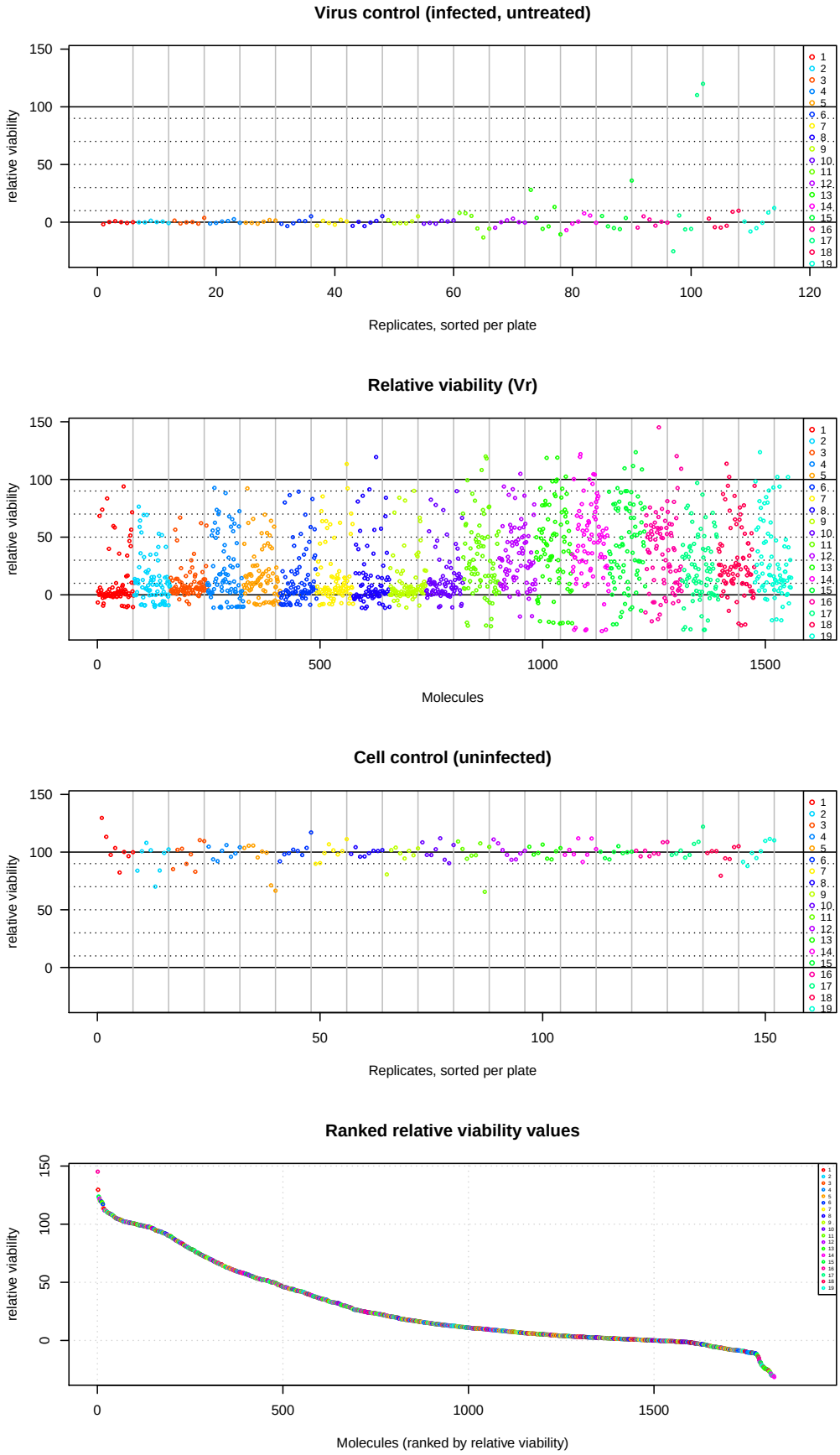

Figure 5: Values of the plate-wise relative viability index for all the tested molecules. Molecules are colored according to the plate number. A: virus control (infected untreated); B: treated cells; C: cell control (untreated cells); D: ranked values (all types)

This approach relies on the assumption that, *in each plate*, the number of active molecule (statistical outliers). Since teach plate contains tests of 80 molecules, there are 19 molecules above the third quartile ($Q3$). However, it has to be noted that the plates were manufactured with some grouping of molecules of the same structural family. It might thus happen that some plates contain more than 19 molecules having an inhibitory effect. Such a situation would result in a loss of sensitivity, since the presence of active molecules in the inter-quartile range would lead to over-estimate the dispersion.

An alternative is to estimate the dispersion based on the range between the first quartile ($Q1$) and the median ($\tilde{x}$) of each plate.

In summary, we compute robust estimators, in order to avoid the effect of outliers (in this case, the suspected outliers are the molecules having an actual inhibitory effect). To this purpose, we use:

- plate-wise median viability ($\tilde{v}$) to estimate the mean
- plate-wise standardised inter-quantile range (IQR) to estimate the standard deviation

```
#### Copute plate-wise statistics ####
statPerPlate <- data.frame(
  Plate = plateNumbers,
  TrMean = as.vector(by(
    data = inhibTable[wellType == "treated", "viability"],
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = mean)),
  TrSD = as.vector(by(
    data = inhibTable[wellType == "treated", "viability"],
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = sd)),
  TrMedian = as.vector(by(
    data = inhibTable[wellType == "treated", "viability"],
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = median)),
  VCMedian = as.vector(by(
    data = inhibTable[wellType == "virusCtl", "viability"],
    INDICES = inhibTable[wellType == "virusCtl", "Plate"],
    FUN = median)),
  CCMedian = as.vector(by(
    data = inhibTable[wellType == "cellCtl", "viability"],
    INDICES = inhibTable[wellType == "cellCtl", "Plate"],
    FUN = median)),
  TrMin = as.vector(by(
    data = inhibTable[wellType == "treated", "viability"],
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = min)),
  TrMax = as.vector(by(
    data = inhibTable[wellType == "treated", "viability"],
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = max)),
  TrQ1 = as.vector(by(
    data = inhibTable[wellType == "treated", "viability"],
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = quantile, probs = 0.25)),
  TrQ3 = as.vector(by(
    data = inhibTable[wellType == "treated", "viability"],
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = quantile, probs = 0.75)),
  TrIQR = as.vector(by(    data = inhibTable[wellType == "treated", "viability"],
```

```
    INDICES = inhibTable[wellType == "treated", "Plate"],
    FUN = IQR))
)
rownames(statPerPlate) <- statPerPlate$Plate
```

We define a plate-wise scaling factor from the interquantile range, standardized by the inter-quartile range of a Gaussian distribution.

$$S_i = \frac{Q3_N - Q1_N}{Q3_i - Q3_i} = \frac{1.349}{Q3_i - Q3_i}$$

Where $Q1$ and $Q3$ denote the first and third quartile, N the Normal distribution, and i is the plate number.

The viability measures of each plate are then multiplied by the corresponding scaling factor to obtain plate-wise standardized values (z), which will have the same inter-quartile range as the normal distribution.

$$z_{c,i} = \frac{v_c - \hat{\mu}_i}{\hat{\sigma}_i} = (v_c - \tilde{v_i})\frac{Q3_N - Q1_N}{Q3_i - Q3_i} = (v_c - \tilde{v_i}) \cdot S_i$$

The table below indicates the plate-wise statistics and scaling factor.

```
## Compute the scaling factor as the difference between median and Q1
## standardised relative to the same difference in a normal distribution.
statPerPlate$scaling <- (qnorm(p = 0.5) - qnorm(p = 0.25)) / (statPerPlate$TrMedian - statPerPlate$TrQ1)

## Compute scaling factor based on the standardized inter-quartile range.
statPerPlate$scaling <- (qnorm(p = 0.75) - qnorm(p = 0.25)) / (statPerPlate$TrQ3 - statPerPlate$TrQ1)

kable(statPerPlate, caption = "Plate-wise statistics of treated cells. Column prefixes: Tr = treated ce
```

Table 2: Plate-wise statistics of treated cells. Column prefixes: Tr = treated cells; CC = cell control (uninfected cells); VC = virus control (infected, untreated cells).

Plate	TrMean	TrSD	TrMedian	VCMedian	CCMedian	TrMin	TrMax	TrQ1	TrQ3	TrIQR
1	13078.98	6327.944	10549.5	10202.0	37197.5	7326	35557	9969.50	12096.00	2126.50
2	18894.41	7774.932	17303.5	14018.0	51971.0	10069	43022	13913.00	19892.75	5979.75
3	13606.28	4252.179	12210.5	10482.0	40016.0	8110	30240	11448.50	14054.00	2605.50
4	18639.57	8795.201	15619.5	12241.0	44342.0	8648	41994	12806.00	23904.75	11098.75
5	15143.20	6913.707	13480.5	9876.5	42012.0	7114	39526	10887.75	16908.00	6020.25
6	13807.12	6808.213	11075.5	9942.0	38458.5	7389	35439	9820.00	15667.25	5847.25
7	15326.20	7533.419	12333.0	10626.0	39375.0	8040	43221	11145.50	15479.75	4334.25
8	14881.32	7091.909	11934.5	11185.5	38161.0	8033	43402	10849.50	15586.50	4737.00
9	16608.39	6631.827	14154.0	12852.5	43904.5	9418	40831	13261.75	16285.50	3023.75
10	16253.40	6447.321	14478.0	11993.0	42310.5	8690	39237	12384.00	16514.25	4130.25
11	21207.99	8911.929	19248.0	12856.0	40742.5	5255	46301	15098.00	26419.50	11321.50
12	20477.00	7715.348	19350.5	10781.5	40187.5	5248	41624	14512.75	24710.50	10197.75
13	23317.91	10141.963	23308.5	11987.0	39596.5	5097	44821	16580.00	30565.00	13985.00
14	23068.06	8605.551	23353.5	12083.0	36942.0	4263	42400	18064.25	28255.75	10191.50
15	23456.87	9728.914	23208.5	11686.0	38995.5	4685	45445	16517.25	32070.50	15553.25
16	19153.77	8060.416	19596.0	10090.0	34521.5	4483	45555	12661.00	24257.75	11596.75
17	17726.85	6808.917	16780.5	12314.5	36445.0	4983	35715	12940.00	22049.00	9109.00
18	20831.10	8404.778	18803.0	12623.0	40740.5	5173	44573	15462.00	24899.00	9437.00
19	18404.23	8714.324	15103.5	10505.0	37360.5	4578	43712	12725.75	23102.50	10376.75

```
#### Compute plate-wise IQR-standardised viability ####

## Centering: substract the median
## Scaling: divide by IQR
## Standardise: multiply by IQR of the normal distribution
plate <- as.vector(inhibTable$Plate)
inhibTable$iqNormV <- (inhibTable$viability - statPerPlate[plate, "TrMedian"]) * statPerPlate[plate, "s
# IQR(inhibTable$iqNormV)
# IQR(rnorm(n = 1000000))

# normIQR <- 2 * (qnorm(p = 0.5) - qnorm(p = 0.25))
# normII <- normII * normIQR
# sd(normII)
# IQR(normII)
```

```
### Descriptive statistics on the IQR-standardised viability ####
iqNormVstat <- data.frame(
  mean = mean(inhibTable$iqNormV),
  sd = sd(inhibTable$iqNormV),
  IQR = IQR(inhibTable$iqNormV),
  var = var(inhibTable$iqNormV),
  min = min(inhibTable$iqNormV),
  Q1 = as.vector(quantile(inhibTable$iqNormV, probs = 0.25)),
  median = median(inhibTable$iqNormV),
  Q3 = as.vector(quantile(inhibTable$iqNormV, probs = 0.75)),
  max = max(inhibTable$iqNormV)
)

kable(t(iqNormVstat),
      col.names = "Stat",
      caption = "Statistics of the plate-wise IQR-standardised viability")
```

Table 3: Statistics of the plate-wise IQR-standardised viability

	Stat
mean	0.8456739
sd	2.6832047
IQR	1.6171691
var	7.1995876
min	-2.5268796
Q1	-0.4963435
median	0.0209851
Q3	1.1208256
max	21.9772962

Histograms of inter-quartile standardised viability

The histogram of plate-wise IQR-standardised viability shows a clear improvement : the median is much closer to the mode than with the raw or log-transformed II values.

```
#### Histograms of IQR-standardised viability ####
histBreaks = seq(from = floor(min(inhibTable$iqNormV)),
                 to = ceiling(max(inhibTable$iqNormV)), by = 0.1)
```

```
par(mfrow = c(3,1))
hist(inhibTable[wellType == "virusCtl", "iqNormV"],
     main = "Virus control - IQR standardised viability",
     breaks = histBreaks, xlab = "IQR-standardised viability",
     col = "orange", border = "orange")
hist(inhibTable[wellType == "treated", "iqNormV"],
     main = "Treated cells - IQR standardised viability",
     breaks = histBreaks, xlab = "IQR-standardised viability",
     col = "grey", border = "grey")
abline(v = mean(inhibTable[wellType == "treated", "iqNormV"]), col = "blue")
abline(v = median(inhibTable[wellType == "treated", "iqNormV"]), col = "darkgreen")
legend("topright", legend = c("mean", "median"),
       col = c("blue", "darkgreen"),
       lwd = 2)
hist(inhibTable[wellType == "cellCtl", "iqNormV"],
     main = "Cell control (untreated) - IQR standardised viability",
     breaks = histBreaks, xlab = "IQR-standardised viability",
     col = "palegreen", border = "palegreen")
```

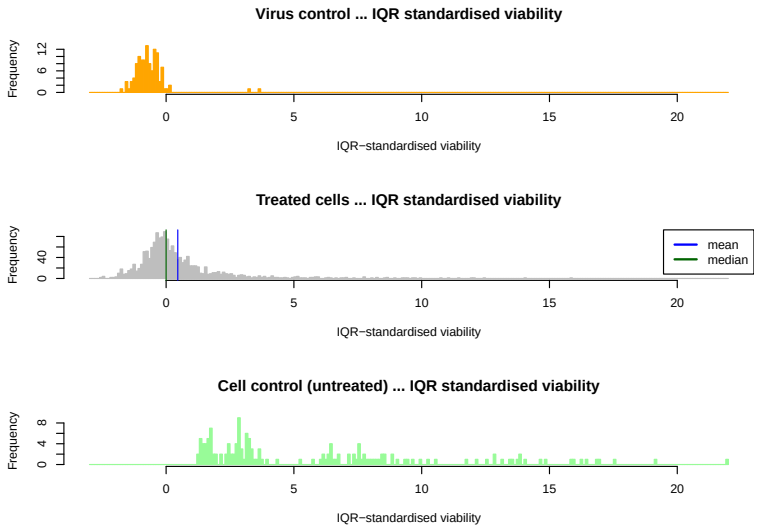

Figure 6: Distribution of the plate-wise IQR-standardised viability. Top: virus control (infected, untreated); middle: treated; bottom: cell control (untreated)

```
par(par.ori)
```

Boxplots: inter-quartile standardised viability

```
#### Boxplots of relative viability per plate ####

## Box plot per plate and well type
boxplot(iqNormV ~ Plate + wellType,
        main = "Inter-quartile standardized viability",
        data = inhibTable,
        las = 1, col = plateColor,
        xlab = "iqNormV",
        cex.axis = 0.5, cex = 0.5,
        horizontal = TRUE)
```

```
abline(v = seq(from = 0, to = max(inhibTable$iqNormV), by = 1), col = "#EEEEEE", lty = "dashed")
abline(v = seq(from = 0, to = max(inhibTable$iqNormV), by = 5), col = "grey")
legend("topright", legend = names(plateColor),
       title = "Plate", fill = plateColor, cex = 0.8)

## Add points to denote the arbidol controls
stripchart(iqNormV ~ Plate, vertical = FALSE, data = inhibTable[arbidolWells, ],
    method = "jitter", add = TRUE, pch = 20, cex = 0.7, col = 'orange')
```

```
par(par.ori)
```

The above boxplots show that the inter-quartile standardization efficiently corrects for the over-dispersion of the plates 11 to 19. However we may expect a lot of sensitivity for these plates. There are however still some weaknesses.

- The vurus control show good properties: in absence of treatment, infected cells have sligtly negative values, except for 2 outliers in plate 17.
- The cell control box plots show wide variation in their medians and dispersion:
  - Uninfected cells (cell control) have much lower values in some plates (plates 4 and 11-19) than in other ones. This is not expected, since these cells should by definition have the same viability values.
  - Even for the plates where the cell controls have a high viability after IQR-based standardisation, there are strong between-plates variations.

This standardisation seems thus efficient to correct the apparent over-dispersion of plates 11 to 19, and thereby reduce the rate of likely false positives, but the wide between-plate variability of the untreated cells suggest that the resulting z-scores should not be interpreted as indicators of viability.

Dot plots: inter-quartile standardised viability

```
yRange <- c(floor(min(inhibTable$iqNormV)), ceiling(max(inhibTable$iqNormV)))

par(mfrow = c(4,1))
plot(inhibTable[wellType == "virusCtl", "iqNormV"],
     panel.first = grid(),
     main = "Virus control (infected, untreated)",
     xlab = "Replicates, sorted per plate",
     ylab = "iqNormV",
     ylim = yRange,
     col = inhibTable[wellType == "virusCtl", "color"],
     cex = 0.5, las = 1
     )
abline(h = 0)
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)
plot(inhibTable[wellType == "treated", "iqNormV"],
     panel.first = grid(),
     main = "IQR-standardised viability (iqNormV)",
     xlab = "Molecules",
     ylab = "iqNormV",
     ylim = yRange,
```

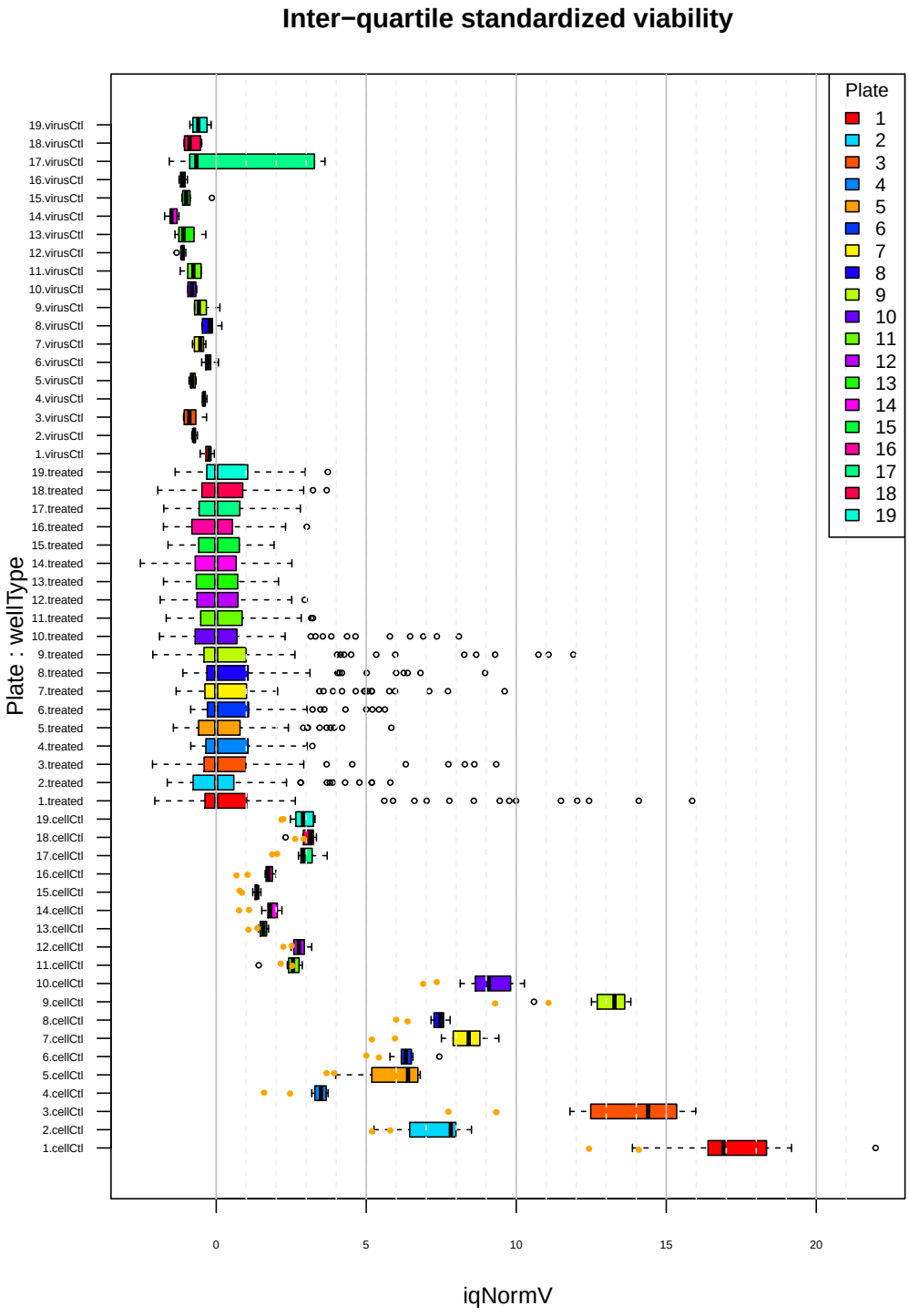

Figure 7: Distribution of inter-quartile standardised viability (iqNormV) values per plate. Top virus control (untreated infected cells); middle: treated cells; bottom: cell control (uninfected). Orange dots denote the arbidol control duplicates.

```
     col = inhibTable[wellType == "treated", "color"],
     cex = 0.5, las = 1,
     xlim = c(0, length(inhibTable[wellType == "treated", "iqNormV"])*1.05)
     )
abline(h = 0)
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)
plot(inhibTable[wellType == "cellCtl", "iqNormV"],
     panel.first = grid(),
     main = "Cell control (uninfected)",
     xlab = "Replicates, sorted per plate",
     ylab = "iqNormV",
     ylim = yRange,
     col = inhibTable[wellType == "cellCtl", "color"],
     cex = 0.5, las = 1
     )
abline(h = 0)
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)
# names(supTable)
iqNormVrank <- order(inhibTable$iqNormV, decreasing = TRUE)
plot(inhibTable[iqNormVrank, "iqNormV"],
     main = "Ranked IQR-standardised viability values",
     xlab = "Molecules (ranked by iqNormV index)",
     ylab = "iqNormV",
     col = inhibTable[iqNormVrank, "color"],
     cex = 0.5, las = 1,
     panel.first = grid(),
     xlim = c(0, length(iqNormVrank) * 1.05)
     )
abline(h = 0)
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.4)
```

```
par(mfrow = c(1,1))
```

The plot of IQR-standardised viability values (top panel) clearly shows that the plate-wise normalisation suppressed the background bias. However it denotes a new problem: the cell controls show striking differences depending on the plates. Noticeably,they show very high values in plates 1 and 3, and very low values in plates 11 to 19, as well as in plate 4.

P-value computation

We compute the p-value as the upper tail of the normal distribution (rigth-side test) in order to detect significantly high values of the plate-wise IQR-standardised index.

```
#### Compute P-value for the inhibition index ####
inhibTable$p.value <- pnorm(inhibTable$iqNormV, mean = 0, sd = 1, lower.tail = FALSE)
inhibTable$log10Pval <- log10(inhibTable$p.value)
```

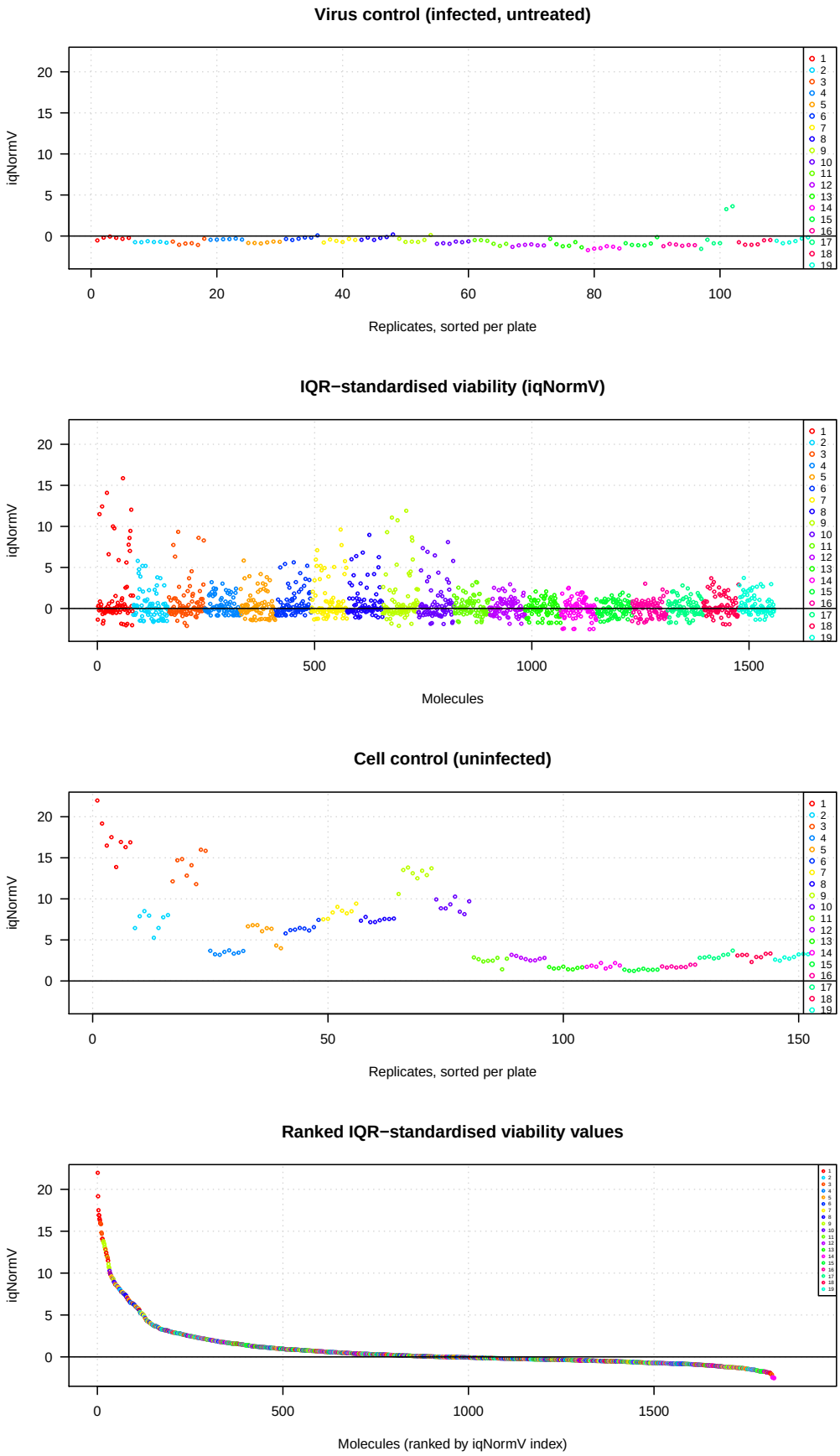

Figure 8: Values of the plate-wise IQR-standardised viability index for all the tested molecules. Molecules are colored according to the plate number. A: virus control (infected untreated); B: treated cells; C: cell control (untreated cells); D: ranked values (all types)

```
inhibTable$e.value <- inhibTable$p.value * sum(wellType == "treated")
inhibTable$padj <- NA
inhibTable[wellType == "treated", "padj"] <-
  p.adjust(inhibTable[wellType == "treated", "p.value"], method = "fdr")
inhibTable$log10Padj <- log10(inhibTable$padj)
```

```
hist(inhibTable[wellType == "treated", "p.value"],
     breaks = 20,
     col = "grey",
     main = "P-value histogram after plate-wise normalisation",
     xlab = "Nominal P-value (unadjusted)",
     ylab = "Frequency")
```

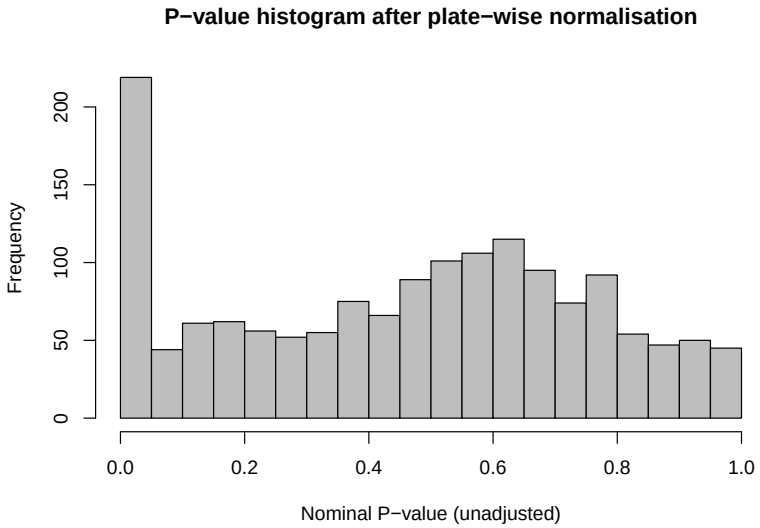

Figure 9: Histogram of the nominal (unadjusted) p-values derived from the plate-wise IQR-standardised viability index.

```
## Estimate the proportion of tests under H0 and H1
## following the method proposed by Storey-Tibshirani (2003)
# lambda <- 0.40
# table(inhibTable[wellType == "treated", "p.value"] > lambda)
# m0 <- sum(inhibTable[wellType == "treated", "p.value"] > lambda) / (1 - lambda)
# m1 <- sum(wellType == "treated") - m0
# print(m0)
# print(m1)
#
```

```
#### Manhattan plot ####
plot(x = -inhibTable[wellType == "treated", "log10Padj"],
     col = inhibTable[wellType == "treated", "color"],
     main = "Significance plot",
     xlab = "Relative viability",
     ylab = "Significance = -log10(Padj)",
     panel.first = c(
       abline(h = 0),
       abline(h = seq(0, -min(inhibTable[wellType == "treated", "log10Padj"]), 20), lty = "dotted", col
       abline(v = (1:19) * 80, col = "grey")
     ),
```

```
     las = 1,
     cex = 0.5)
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)
```

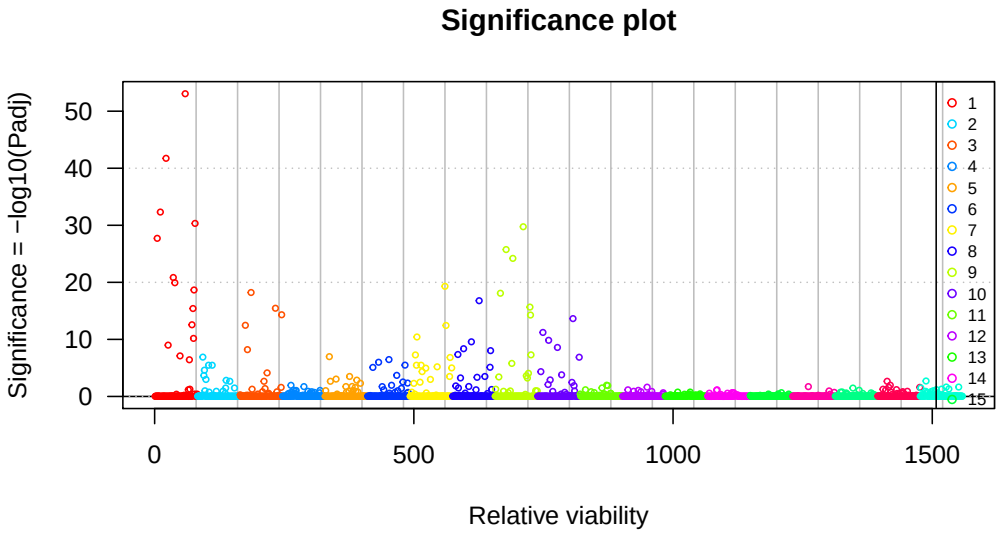

Figure 10: Volcano plot.

```
#### Volcano plot ####
plot(x = inhibTable[wellType == "treated", "Vr"],
     y = -inhibTable[wellType == "treated", "log10Padj"],
     col = inhibTable[wellType == "treated", "color"],
     main = "Volcano plot",
     xlab = "Relative viability",
     ylab = "Significance = -log10(Padj)")
grid()
```

Three-points normalisation with arbidol

```
#### Viability indices for the arbidol contron ####
# sum(arbidolWells)
kable(inhibTable[arbidolWells, c("ID", "Vr", "iqNormV", "log10Padj")], caption = "Viability indices for
```

Table 4: Viability indices for the arbidol controls (2 replicates per plate)

	ID	Vr	iqNormV	log10Padj
01A12	01A12	73.85675	12.4275490	-32.3178798
01B12	01B12	83.53614	14.0851476	-41.7386174
02A12	02A12	76.42084	5.8018695	-6.8937539
02B12	02B12	69.27516	5.1900660	-5.4705606
03A12	03A12	56.46374	7.7389641	-12.4691925
03B12	03B12	66.89917	9.3346482	-18.2166799
04A12	04A12	51.48438	1.5981130	-0.4177311

	ID	Vr	iqNormV	log10Padj
04B12	04B12	73.64568	2.4627726	-1.0961060
05A12	05A12	62.30337	3.6787294	-2.6575853
05B12	05B12	65.78861	3.9296920	-3.0420868
06A12	06A12	80.01683	5.0026843	-5.0885199
06B12	06B12	86.39560	5.4223335	-5.9921058
07A12	07A12	72.54165	5.9595684	-7.2737070
07B12	07B12	63.90483	5.1867667	-5.4705606
08A12	08A12	80.86412	5.9986444	-7.3537397
08B12	08B12	85.82788	6.3799583	-8.3653050
09A12	09A12	71.29170	9.2955227	-18.0887986
09B12	09B12	84.15078	11.0769118	-25.7430361
10A12	10A12	82.45403	7.3529630	-11.2189130
10B12	10B12	77.88241	6.9002820	-9.8404583
11A12	11A12	99.33480	2.5390102	-1.1699219
11B12	11B12	87.60153	2.1491448	-0.8285687
12A12	12A12	93.84649	2.5169971	-1.1591100
12B12	12B12	86.67109	2.2378820	-0.9018416
13A12	13A12	92.60218	1.3741071	-0.2871946
13B12	13B12	81.41401	1.0761451	-0.1553551
14A12	14A12	68.51040	0.7624782	-0.0762164
14B12	14B12	78.64355	1.0959011	-0.1634674
15A12	15A12	78.50016	0.8600007	-0.0957624
15B12	15B12	75.04348	0.7781248	-0.0791430
16A12	16A12	75.66461	1.0445889	-0.1460609
16B12	16B12	62.60361	0.6733992	-0.0552944
17A12	17A12	75.26367	2.0282082	-0.7396688
17B12	17B12	70.72170	1.8658983	-0.6123467
18A12	18A12	87.37619	2.6284873	-1.2650548
18B12	18B12	94.48208	2.9140930	-1.5917678
19A12	19A12	81.17518	2.2361983	-0.9018416
19B12	19B12	79.35432	2.1726282	-0.8489690

Selection of candidate molecules

```
#### Select significant normalised II values ####
alpha <- 0.05

kable(table(inhibTable$padj < alpha))
```

Var1	Freq
FALSE	1439
TRUE	119

```
# table(inhibTable$padj < alpha)
selected <- subset(inhibTable, inhibTable$padj < alpha)

## Sort by decreasing adjusted p-value
selected <- selected[order(selected$padj, decreasing = FALSE), ]
# kable(names(selected), row.names=TRUE)
```

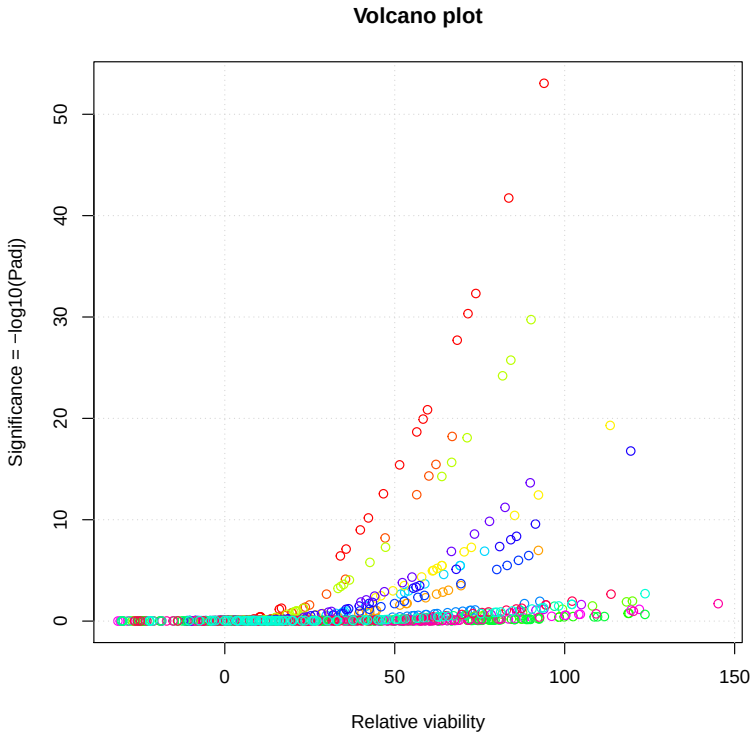

Figure 11: Volcano plot.

```
## Print selected molecules
kable(selected[ , c(1:3, 5:7, 10, 12, 15)],
      row.names = FALSE,
      digits = 4,
      caption = "Candidate moecules sorted by significance after plate-wise normalisation. ")
```

Table 6: Candidate moecules sorted by significance after plate-wise normalisation.

ID	Plate	Row	viability	cellControl	virusControl	color	vcMed	iqNormV
01F08	1	F	35557	37964.25	10135.667	#FF0000FF	10202.0	15.8639
01B12	1	B	32753	37964.25	10135.667	#FF0000FF	10202.0	14.0851
01A12	1	A	30140	37964.25	10135.667	#FF0000FF	10202.0	12.4275
01H07	1	H	29517	37964.25	10135.667	#FF0000FF	10202.0	12.0323
09F04	9	F	40831	43177.12	13104.833	#BCFF00FF	12852.5	11.9014
01A06	1	A	28661	37964.25	10135.667	#FF0000FF	10202.0	11.4893
09B12	9	B	38983	43177.12	13104.833	#BCFF00FF	12852.5	11.0769
09D04	9	D	38231	43177.12	13104.833	#BCFF00FF	12852.5	10.7414
01D05	1	D	26302	37964.25	10135.667	#FF0000FF	10202.0	9.9929
01D08	1	D	25951	37964.25	10135.667	#FF0000FF	10202.0	9.7702
07G07	7	G	43221	39295.12	10537.833	#FFF200FF	10626.0	9.6135
01H05	1	H	25447	37964.25	10135.667	#FF0000FF	10202.0	9.4505
03B12	3	B	30240	39305.25	10619.000	#FF5100FF	10482.0	9.3346
09A12	9	A	34990	43177.12	13104.833	#BCFF00FF	12852.5	9.2955
08E11	8	E	43402	38082.62	11173.500	#1B00FFFF	11185.5	8.9612
09G07	9	G	33578	43177.12	13104.833	#BCFF00FF	12852.5	8.6656
03G08	3	G	28831	39305.25	10619.000	#FF5100FF	10482.0	8.6051

ID	Plate	Row	viability	cellControl	virusControl	color	vcMed	iqNormV
01H03	1	H	24085	37964.25	10135.667	#FF0000FF	10202.0	8.5865
03H10	3	H	28214	39305.25	10619.000	#FF5100FF	10482.0	8.2857
09G08	9	G	32682	43177.12	13104.833	#BCFF00FF	12852.5	8.2659
10G08	10	G	39237	42600.38	12007.833	#6B00FFFF	11993.0	8.0865
01G11	1	G	22800	37964.25	10135.667	#FF0000FF	10202.0	7.7713
03A12	3	A	27158	39305.25	10619.000	#FF5100FF	10482.0	7.7390
07G09	7	G	37155	39295.12	10537.833	#FFF200FF	10626.0	7.7255
10A12	10	A	36991	42600.38	12007.833	#6B00FFFF	11993.0	7.3530
07B04	7	B	35140	39295.12	10537.833	#FFF200FF	10626.0	7.0984
01H04	1	H	21606	37964.25	10135.667	#FF0000FF	10202.0	7.0139
10B12	10	B	35605	42600.38	12007.833	#6B00FFFF	11993.0	6.9003
08D06	8	D	35840	38082.62	11173.500	#1B00FFFF	11185.5	6.8077
01C05	1	C	20965	37964.25	10135.667	#FF0000FF	10202.0	6.6072
10D08	10	D	34265	42600.38	12007.833	#6B00FFFF	11993.0	6.4626
08B12	8	B	34338	38082.62	11173.500	#1B00FFFF	11185.5	6.3800
03B05	3	B	24413	39305.25	10619.000	#FF5100FF	10482.0	6.3178
08H03	8	H	33883	38082.62	11173.500	#1B00FFFF	11185.5	6.2504
08A12	8	A	32999	38082.62	11173.500	#1B00FFFF	11185.5	5.9986
09G09	9	G	27535	43177.12	13104.833	#BCFF00FF	12852.5	5.9696
07A12	7	A	31481	39295.12	10537.833	#FFF200FF	10626.0	5.9596
01E08	1	E	19837	37964.25	10135.667	#FF0000FF	10202.0	5.8917
05A10	5	A	39526	39914.62	9950.833	#FFA100FF	9876.5	5.8361
02A12	2	A	43022	49597.88	14070.500	#00D7FFFF	14018.0	5.8019
10H10	10	H	32205	42600.38	12007.833	#6B00FFFF	11993.0	5.7898
07H07	7	H	30870	39295.12	10537.833	#FFF200FF	10626.0	5.7694
06D11	6	D	35439	38823.25	10013.833	#0036FFFF	9942.0	5.6207
01G06	1	G	19381	37964.25	10135.667	#FF0000FF	10202.0	5.6024
06B12	6	B	34579	38823.25	10013.833	#0036FFFF	9942.0	5.4223
09D02	9	D	26112	43177.12	13104.833	#BCFF00FF	12852.5	5.3348
06H02	6	H	33631	38823.25	10013.833	#0036FFFF	9942.0	5.2036
02B12	2	B	40310	49597.88	14070.500	#00D7FFFF	14018.0	5.1901
02C08	2	C	40283	49597.88	14070.500	#00D7FFFF	14018.0	5.1840
07B12	7	B	28998	39295.12	10537.833	#FFF200FF	10626.0	5.1868
07B03	7	B	28977	39295.12	10537.833	#FFF200FF	10626.0	5.1802
07F02	7	F	28578	39295.12	10537.833	#FFF200FF	10626.0	5.0560
08H02	8	H	29549	38082.62	11173.500	#1B00FFFF	11185.5	5.0162
06A12	6	A	32760	38823.25	10013.833	#0036FFFF	9942.0	5.0027
07H10	7	H	28271	39295.12	10537.833	#FFF200FF	10626.0	4.9605
07C10	7	C	28189	39295.12	10537.833	#FFF200FF	10626.0	4.9350
02B04	2	B	38458	49597.88	14070.500	#00D7FFFF	14018.0	4.7723
07C03	7	C	27275	39295.12	10537.833	#FFF200FF	10626.0	4.6505
10A08	10	A	28693	42600.38	12007.833	#6B00FFFF	11993.0	4.6428
03F02	3	F	20971	39305.25	10619.000	#FF5100FF	10482.0	4.5357
09G04	9	G	24235	43177.12	13104.833	#BCFF00FF	12852.5	4.4974
10E06	10	E	27835	42600.38	12007.833	#6B00FFFF	11993.0	4.3625
06F06	6	F	29752	38823.25	10013.833	#0036FFFF	9942.0	4.3087
02B03	2	B	36351	49597.88	14070.500	#00D7FFFF	14018.0	4.2970
09F10	9	F	23712	43177.12	13104.833	#BCFF00FF	12852.5	4.2641
05E07	5	E	32203	39914.62	9950.833	#FFA100FF	9876.5	4.1952
07H06	7	H	25806	39295.12	10537.833	#FFF200FF	10626.0	4.1933
08G02	8	G	26652	38082.62	11173.500	#1B00FFFF	11185.5	4.1912
09A09	9	A	23449	43177.12	13104.833	#BCFF00FF	12852.5	4.1468

ID	Plate	Row	viability	cellControl	virusControl	color	vcMed	iqNormV
08E07	8	E	26367	38082.62	11173.500	#1B00FFFF	11185.5	4.1100
08B06	8	B	26146	38082.62	11173.500	#1B00FFFF	11185.5	4.0471
09G02	9	G	23192	43177.12	13104.833	#BCFF00FF	12852.5	4.0321
05B12	5	B	31018	39914.62	9950.833	#FFA100FF	9876.5	3.9297
07D09	7	D	24842	39295.12	10537.833	#FFF200FF	10626.0	3.8933
02B07	2	B	34533	49597.88	14070.500	#00D7FFFF	14018.0	3.8868
10C04	10	C	26226	42600.38	12007.833	#6B00FFFF	11993.0	3.8370
05F11	5	F	30496	39914.62	9950.833	#FFA100FF	9876.5	3.8127
02F05	2	F	34108	49597.88	14070.500	#00D7FFFF	14018.0	3.7909
19B02	19	B	43712	37547.62	10838.667	#00FFD7FF	10505.0	3.7191
02G02	2	G	33661	49597.88	14070.500	#00D7FFFF	14018.0	3.6901
03E06	3	E	19319	39305.25	10619.000	#FF5100FF	10482.0	3.6804
05A12	5	A	29898	39914.62	9950.833	#FFA100FF	9876.5	3.6787
18B09	18	B	44573	39976.62	13087.000	#FF0051FF	12623.0	3.6837
06G08	6	G	26690	38823.25	10013.833	#0036FFFF	9942.0	3.6023
07B10	7	B	23824	39295.12	10537.833	#FFF200FF	10626.0	3.5764
10G06	10	G	25357	42600.38	12007.833	#6B00FFFF	11993.0	3.5532
06H07	6	H	26141	38823.25	10013.833	#0036FFFF	9942.0	3.4757
05G08	5	G	28867	39914.62	9950.833	#FFA100FF	9876.5	3.4477
07A09	7	A	23381	39295.12	10537.833	#FFF200FF	10626.0	3.4385
10B11	10	B	24609	42600.38	12007.833	#6B00FFFF	11993.0	3.3089
11E11	11	E	46301	39977.00	12715.167	#6BFF00FF	12856.0	3.2234
18C04	18	C	41359	39976.62	13087.000	#FF0051FF	12623.0	3.2243
06E06	6	E	24996	38823.25	10013.833	#0036FFFF	9942.0	3.2115
04B07	4	B	41994	44212.12	12335.000	#0086FFFF	12241.0	3.2056
11F03	11	F	45815	39977.00	12715.167	#6BFF00FF	12856.0	3.1655
10G10	10	G	24128	42600.38	12007.833	#6B00FFFF	11993.0	3.1518
08A08	8	A	22905	38082.62	11173.500	#1B00FFFF	11185.5	3.1241
05F02	5	F	27115	39914.62	9950.833	#FFA100FF	9876.5	3.0551
08C11	8	C	22641	38082.62	11173.500	#1B00FFFF	11185.5	3.0489
05D10	5	D	27037	39914.62	9950.833	#FFA100FF	9876.5	3.0377
04D11	4	D	40541	44212.12	12335.000	#0086FFFF	12241.0	3.0290
06C08	6	C	24183	38823.25	10013.833	#0036FFFF	9942.0	3.0239
16C10	16	C	45555	34816.88	10077.500	#FF00A1FF	10090.0	3.0197
19E10	19	E	37933	37547.62	10838.667	#00FFD7FF	10505.0	2.9678
19H04	19	H	37918	37547.62	10838.667	#00FFD7FF	10505.0	2.9659
12E07	12	E	41624	40366.50	10758.167	#BC00FFFF	10781.5	2.9464
03H09	3	H	17844	39305.25	10619.000	#FF5100FF	10482.0	2.9167
18B12	18	B	39189	39976.62	13087.000	#FF0051FF	12623.0	2.9141
18H11	18	H	39180	39976.62	13087.000	#FF0051FF	12623.0	2.9128
05F08	5	F	26431	39914.62	9950.833	#FFA100FF	9876.5	2.9019
11E02	11	E	43021	39977.00	12715.167	#6BFF00FF	12856.0	2.8326
19A03	19	A	36884	37547.62	10838.667	#00FFD7FF	10505.0	2.8315
08B02	8	B	21857	38082.62	11173.500	#1B00FFFF	11185.5	2.8257
02G11	2	G	29796	49597.88	14070.500	#00D7FFFF	14018.0	2.8182
02F04	2	F	29770	49597.88	14070.500	#00D7FFFF	14018.0	2.8123
17D04	17	D	35715	37294.88	20304.000	#00FF86FF	12314.5	2.8041
06G05	6	G	23041	38823.25	10013.833	#0036FFFF	9942.0	2.7605
03E08	3	E	17478	39305.25	10619.000	#FF5100FF	10482.0	2.7272
19F05	19	F	35692	37547.62	10838.667	#00FFD7FF	10505.0	2.6765

Analysis of Touret's original Inhibition Index (II)

Inhibition index

The inhibition index is derived from the raw viability measurement in the following way.

Descriptive stats

```
ii <- supTable$Inhibition.Index
iiStat <- list(
  mean = mean(ii),
  sd = sd(ii),
  var = var(ii),
  min = min(ii),
  Q1 = as.vector(quantile(ii, probs = 0.25)),
  median = median(ii),
  Q3 = as.vector(quantile(ii, probs = 0.75)),
  max = max(ii)
)

kable(t(as.data.frame.list(iiStat)), col.names = "Stat")
```

	Stat
mean	0.2865268
sd	0.3758726
var	0.1412802
min	-0.4444311
Q1	0.0279599
median	0.1665643
Q3	0.4917926
max	2.0988878

Distribution

The distribution of inhibition index values is trongly asymmetrical. The mode is much lower than the mean and the median (robust estimator of central tendency). A normal fit will tuhs give a poor estimate of the p-values.

```
hist(supTable$Inhibition.Index, breaks = 100, col = "grey", border = "grey")
abline(v = iiStat$mean, col = "blue")
abline(v = iiStat$median, col = "darkgreen")

legend("topright", legend = c("mean", "median"),
       col = c("blue", "darkgreen"),
       lwd = 2)
```

Normalisation

Log transform

A classical method for normalisation is to take the log of the values. We first had to shift the data in order for all of them to take positive values.

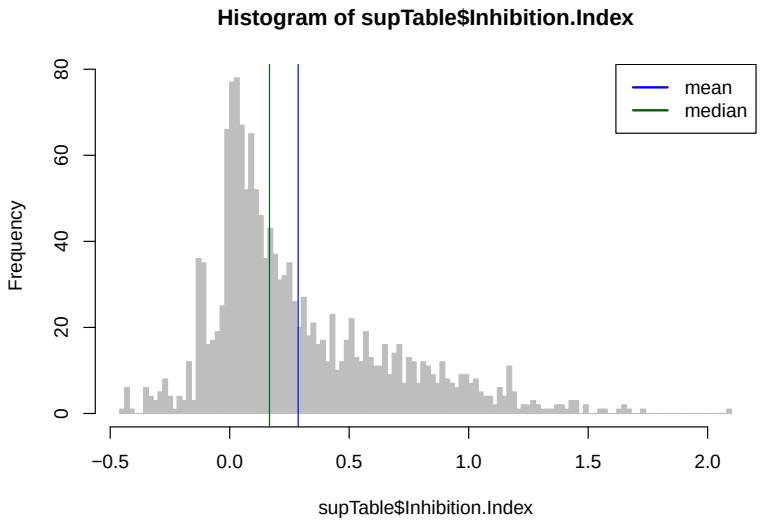

Figure 12: Distribution of the inhibition index

```
# fit.gamma <- fitdist(data = ii - iiStat$min + 1, distr = "gamma", method = "mge")
# summary(fit.gamma)
#
# plot(fit.gamma)

#### Compute a normalised distribution from inhibition indices ####
iiPositive <- ii - iiStat$min + 1 ## shift the distrib to achieve non-negative numbers
logII <- log(iiPositive)

logIIStat <- list(
  mean = mean(logII),
  sd = sd(logII),
  var = var(logII),
  min = min(logII),
  Q1 = as.vector(quantile(logII, probs = 0.25)),
  median = median(logII),
  Q3 = as.vector(quantile(logII, probs = 0.75)),
  max = max(logII)
)

kable(t(as.data.frame.list(logIIStat)), col.names = "Stat", caption = "Parameters of the log-normalised
```

Table 8: Parameters of the log-normalised inhibition index distribution

	Stat
mean	0.5272798
sd	0.2027703
var	0.0411158
min	0.0000000
Q1	0.3868877
median	0.4768523
Q3	0.6607395

	Stat
max	1.2650638

However, even after log transformation the distribution remains highly asymmetrical, with a mode much smaller than the median and mean.

```
#### Histogram of log-normalised values ####
hist(logII, breaks = 100, col = "grey", border = "grey")
abline(v = mean(logII), col = "blue")
abline(v = median(logII), col = "darkgreen")

legend("topright", legend = c("mean", "median"),
       col = c("blue", "darkgreen"),
       lwd = 2)
```

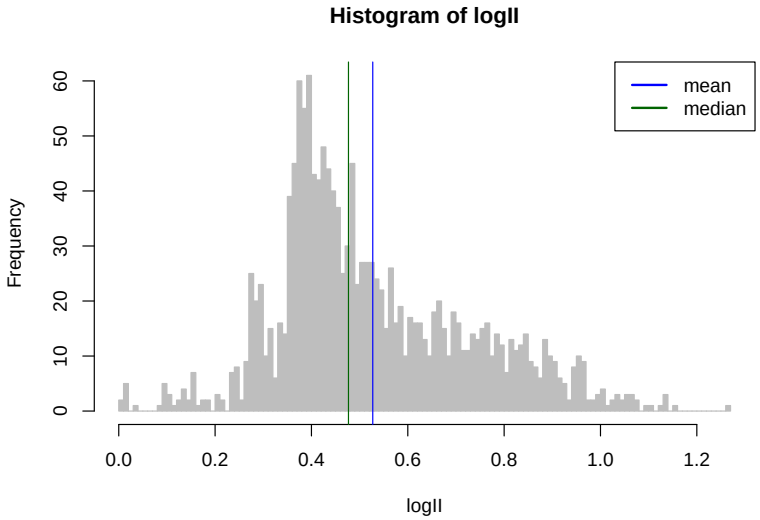

Figure 13: Distribution of the inhibition index

Evidence of a plate bias

Ranked values

We plot the inhibition index values ordered by plate and position number (top) or ranked by decreasing value (bottom). In both cases, the color denotes the plate number.

```
par(mfrow = c(2,1))
plot(ii,
     panel.first = grid(),
     main = "Inhibition index values",
     xlab = "Molecules (ranked by inhibition index)",
     ylab = "Inhibition index",
     col = supTable$color,
     cex = 0.5,
     xlim = c(0, length(ii)*1.05)
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
```

```
       cex = 0.7)

sortedTable <- supTable[order(supTable$Inhibition.Index, decreasing = TRUE), ]
plot(sortedTable$Inhibition.Index,
     panel.first = grid(),
     main = "Ranked inhibition index values",
     xlab = "Molecules (ranked by inhibition index)",
     ylab = "Inhibition index",
     col = sortedTable$color,
     cex = 0.5,
     xlim = c(0, length(ii)*1.05)
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)
```

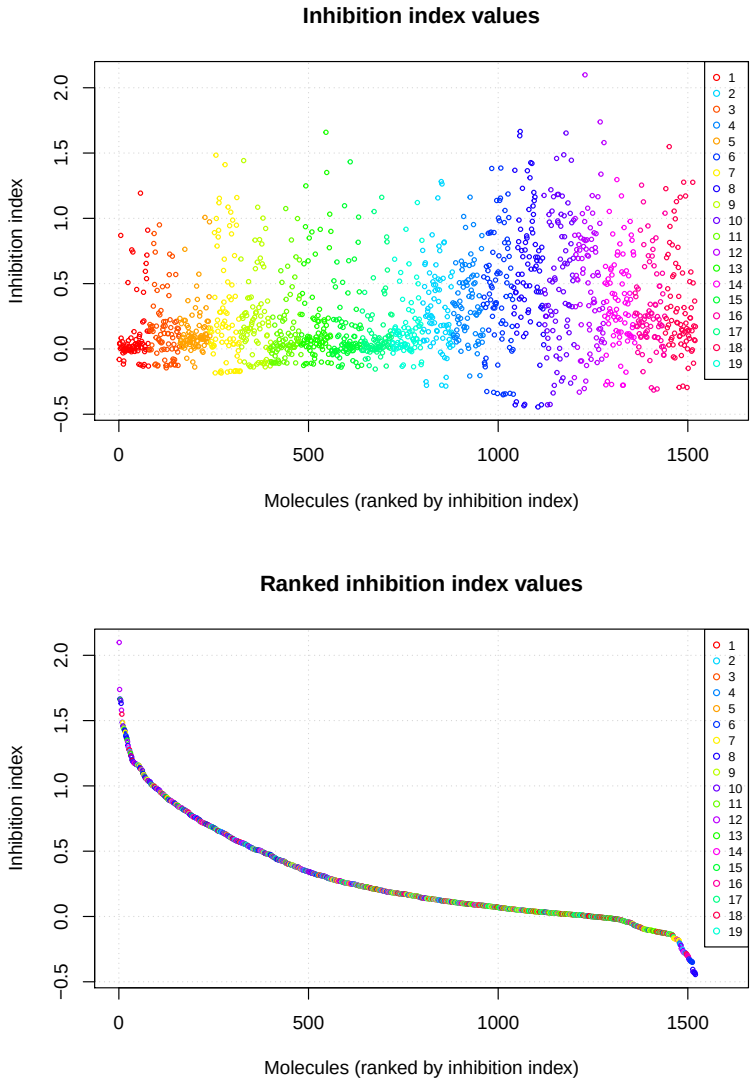

Figure 14: Values of the inhibition index for all the tested molecules. Molecules are colored according to the plate number.

```
par(mfrow = c(1, 1))
```

The molecule-wised colored plots of inhibition index suggest a plate-wise effect.

Plate-wise normalisation

We perform a plate-wise normalisation using robust estimators, in order to avoid the effect of outliers (in this case, the suspected outliers are the molecules having an actual inhibitory effect).

To this purpose, we use: - plate-wise median to estimate the mean - plate-wise standardised inter-quantile range (IQR) to estimate the standard deviation

```
#### Copute plate-wise statistics ####
plateStat <- data.frame(
  plate = plateNumbers,
  mean = as.vector(by(data = supTable$Inhibition.Index, INDICES = supTable$plateNumber, FUN = mean)),
  sd = as.vector(by(data = supTable$Inhibition.Index, INDICES = supTable$plateNumber, FUN = sd)),
  median = as.vector(by(data = supTable$Inhibition.Index, INDICES = supTable$plateNumber, FUN = median)),
  min = as.vector(by(data = supTable$Inhibition.Index, INDICES = supTable$plateNumber, FUN = min)),
  max = as.vector(by(data = supTable$Inhibition.Index, INDICES = supTable$plateNumber, FUN = max)),
  IQR = as.vector(by(data = supTable$Inhibition.Index, INDICES = supTable$plateNumber, FUN = IQR))
)
rownames(plateStat) <- plateStat$plate

kable(plateStat, caption = "Plate-wise statistics")
```

Table 9: Plate-wise statistics

plate	mean	sd	median	min	max	IQR
1	0.1165661	0.2665169	0.0184570	-0.1318422	1.1928831	0.0914910
2	0.1541782	0.2523084	0.1133156	-0.1450055	0.9498831	0.2156783
3	0.1443563	0.1960778	0.0863385	-0.1387721	1.0073009	0.1349143
4	0.2982710	0.4307195	0.1601301	-0.1844422	1.4836918	0.4739120
5	0.2345274	0.3194481	0.1628049	-0.1383338	1.4421869	0.2814504
6	0.1393632	0.2580634	0.0412022	-0.1109600	1.0748024	0.2271337
7	0.2241196	0.3673457	0.0873107	-0.1267828	1.6589036	0.2073556
8	0.1439492	0.2894590	0.0331407	-0.1396088	1.4326962	0.1976995
9	0.1253725	0.2449408	0.0420476	-0.1543792	1.1609812	0.1107754
10	0.1541551	0.2335481	0.0948601	-0.1365916	1.1209955	0.1617424
11	0.3071834	0.3265338	0.2478965	-0.2846758	1.2816167	0.4140013
12	0.3499997	0.2757037	0.3159377	-0.2074233	1.1619058	0.3761552
13	0.4314013	0.4397504	0.4146992	-0.3511397	1.3851376	0.5813343
14	0.5857602	0.4775020	0.6038263	-0.4444311	1.6653052	0.5380693
15	0.5187760	0.4972562	0.4842358	-0.4279441	1.6531873	0.8015369
16	0.5253860	0.4768292	0.5615571	-0.3309767	2.0988878	0.6858842
17	0.3590275	0.3386716	0.3050932	-0.2784932	1.2965981	0.4539432
18	0.2912052	0.3201378	0.2263674	-0.3152611	1.2542724	0.3426682
19	0.3404116	0.4026624	0.1982078	-0.2950106	1.5490336	0.4724498

```
## Centering: substract the median
## Scaling: divide by IQR
## Standardise: multiply by IQR of the normal distribution
normII <- (supTable$Inhibition.Index - plateStat[supTable$plateNumber, "median"]) / plateStat[supTable$
# IQR(normII)
```

```
# IQR(rnorm(n = 1000000))

normIQR <- qnorm(p = 0.75) - qnorm(p = 0.25)
normII <- normII * normIQR
# sd(normII)
# IQR(normII)

supTable$normInhibIndex <- normII
```

```
### Descriptive statistics on the normalised Inhibition Index ####
normIIStat <- list(
  mean = mean(normII),
  sd = sd(normII),
  IQR = IQR(normII),
  var = var(normII),
  min = min(normII),
  Q1 = as.vector(quantile(normII, probs = 0.25)),
  median = median(normII),
  Q3 = as.vector(quantile(normII, probs = 0.75)),
  max = max(normII)
)

kable(t(as.data.frame.list(normIIStat)), col.names = "Stat", caption = "Statistics of the plate-wise no
```

Table 10: Statistics of the plate-wise normalised inhibition index

	Stat
mean	0.4144692
sd	1.8129741
IQR	1.3183994
var	3.2868752
min	-2.6280587
Q1	-0.5094508
median	0.0000000
Q3	0.8089486
max	17.3161990

The histogram of plate-wise normalised values shows a clear improvement : the median is much closer to the mode than with the raw or log-transformed II values.

```
hist(normII, breaks = 100, col = "grey", border = "grey")
abline(v = mean(normII), col = "blue")
abline(v = median(normII), col = "darkgreen")

legend("topright", legend = c("mean", "median"),
       col = c("blue", "darkgreen"),
       lwd = 2)
```

Normalised II plots

The plot of normalised II values (top panel) clearly shows that the plate-wise normalisation suppressed the background bias.

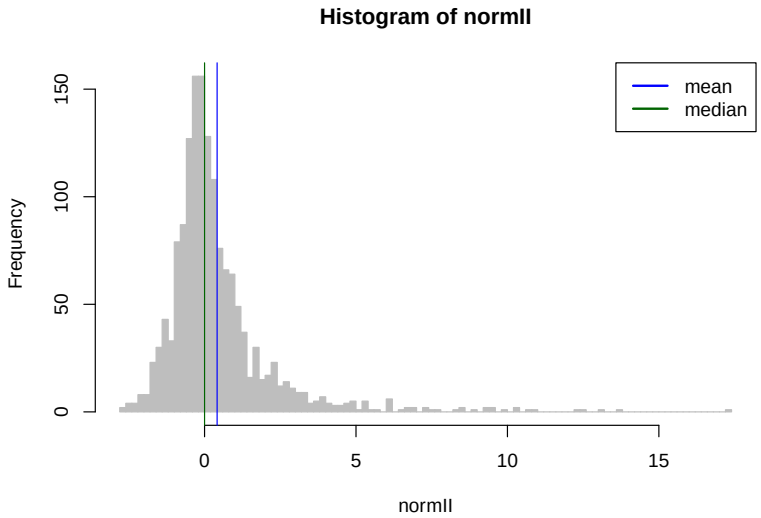

Figure 15: Distribution of the plate-wise normalised inhibition index

```
par(mfrow = c(2,1))
plot(normII,
     panel.first = grid(),
     main = "Inhibition index values",
     xlab = "Molecules (ranked by inhibition index)",
     ylab = "Inhibition index",
     col = supTable$color,
     cex = 0.5,
     xlim = c(0, length(normII)*1.05)
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.7)

# names(supTable)
normIIrank <- order(supTable$normInhibIndex, decreasing = TRUE)
plot(supTable[normIIrank, "normInhibIndex"],
     panel.first = grid(),
     main = "Ranked inhibition index values",
     xlab = "Molecules (ranked by inhibition index)",
     ylab = "Inhibition index",
     col = supTable[normIIrank, "color"],
     cex = 0.5,
     xlim = c(0, length(normII)*1.05)
     )
legend("topright",
       legend = names(plateColor),
       col = plateColor, pch = 1,
       cex = 0.4)
```

```
par(mfrow = c(1,1))
```

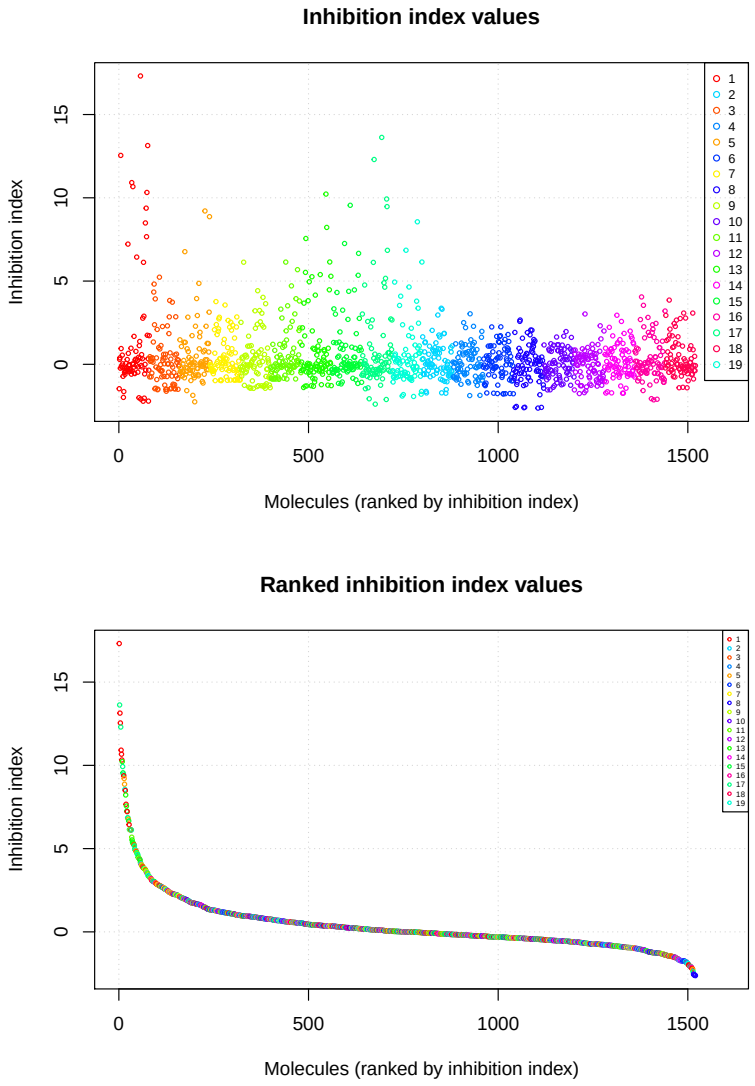

Figure 16: Values of the plate-wise normalised inhibition index for all the tested molecules. Molecules are colored according to the plate number.

P-value computation

We compute the p-value as the upper tail of the normal distribution (rigth-side test) in order to detect significantly high values of the plate-wise normalised index.

```
#### Compute P-value for the inhibition index ####
supTable$p.value <- pnorm(normII, mean = 0, sd = 1, lower.tail = FALSE)
supTable$log10Pval <- log10(supTable$p.value)
supTable$e.value <- supTable$p.value * length(normII)
supTable$padj <- p.adjust(supTable$p.value, method = "fdr")
supTable$log10Padj <- log10(supTable$padj)
```

```
hist(supTable$p.value, breaks = 20,
     col = "grey",
     main = "P-value histogram after plate-wise normalisation",
     xlab = "Nominal P-value (unadjusted)",
     ylab = "Frequency")
```

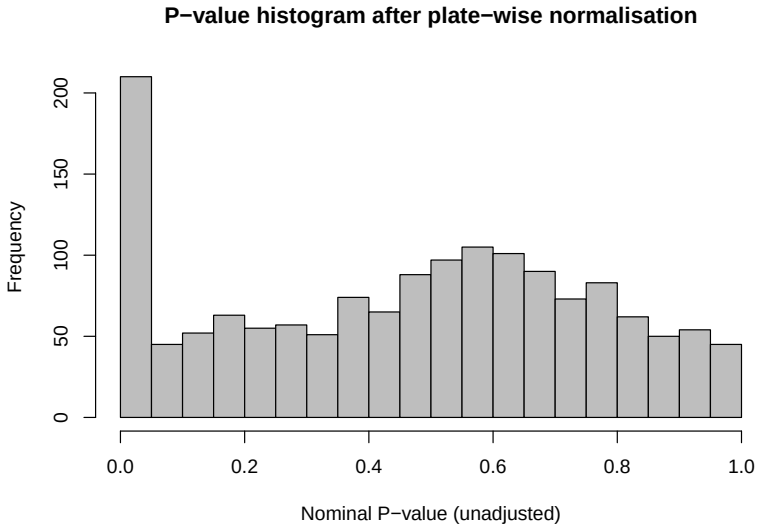

Figure 17: Histogram of the nominal (unadjusted) p-values derived from the plate-wise normalised inhibition index.

```
#### Volcano plot ####

plot(x = supTable$normInhibIndex,
     y = -supTable$log10Padj,
     col = supTable$color,
     main = "Volcano plot",
     xlab = "Normalised inhibition index",
     ylab = "Significance = -log10(Padj)")
grid()
```

Selection of candidate molecules

```
#### Select significant normalised II values ####
alpha <- 0.05
# table(supTable$padj < alpha)
selected <- subset(supTable, supTable$padj < alpha)
```

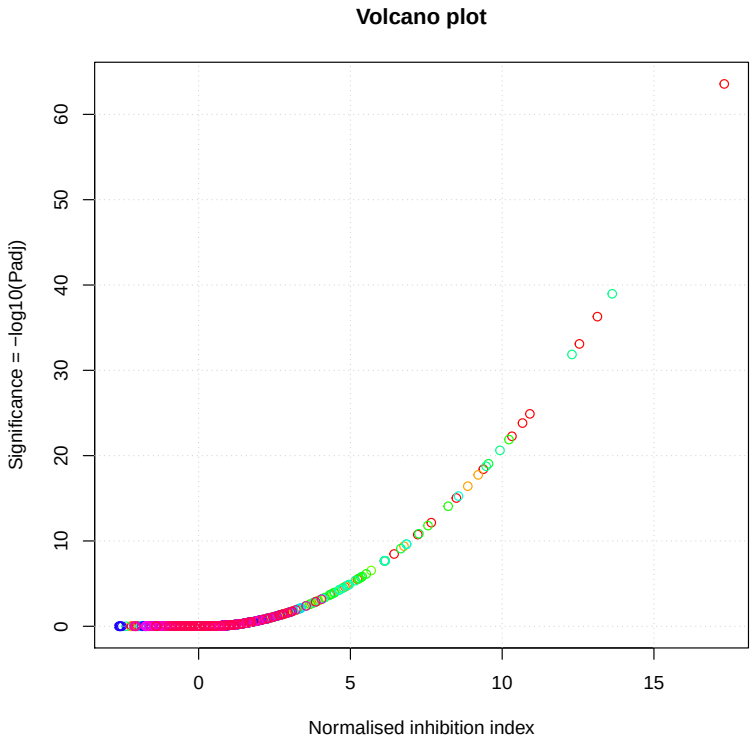

Figure 18: Volcano plot.

```
## Sort by decreasing adjusted p-value
selected <- selected[order(selected$padj, decreasing = FALSE), ]
# kable(names(selected), row.names=TRUE)

## Print selected molecules
kable(selected[ , c(1:3, 5:7, 10, 12, 15)],
      row.names = FALSE,
      digits = 4,
      caption = "Candidate moecules sorted by significance after plate-wise normalisation. ")
```

Table 11: Cand
normalisation.

ID	Prestw.number	Chemical.name	Broad.Therapeutic.class
01F08	Prestw-57	Benoxinate hydrochloride	Neuromuscular
09F04	Prestw-693	Promazine hydrochloride	Central Nervous System
01H07	Prestw-76	Dibucaine	Neuromuscular
01A06	Prestw-5	Atracurium besylate	Neuromuscular
09D04	Prestw-1456	Opipramol dihydrochloride	Central Nervous System
01D05	Prestw-34	Triamterene	Metabolism
01D08	Prestw-37	Pyrimethamine	Infectiology
01H05	Prestw-74	Amitryptiline hydrochloride	Central Nervous System
07G07	Prestw-546	Pregnenolone	Endocrinology
09G07	Prestw-706	Chlorcyclizine hydrochloride	Allergology 'Central Nervous System
08E11	Prestw-1284	Hydroxychloroquine sulfate	Metabolism
09G08	Prestw-707	Diphenylpyraline hydrochloride	Allergology 'Central Nervous System
01H03	Prestw-72	Imipramine hydrochloride	Central Nervous System

ID	Prestw.number	Chemical.name	Broad.Therapeutic.class
03G08	Prestw-227	Clemizole hydrochloride	Allergology 'Dermatology 'Infectiology
03H10	Prestw-239	Orphenadrine hydrochloride	Allergology 'Central Nervous System
10G08	Prestw-787	Merbromin disodium salt	Infectiology
01G11	Prestw-70	Tolnaftate	Infectiology
07G09	Prestw-548	Chloroquine diphosphate	Metabolism
01H04	Prestw-73	Sulindac	Central Nervous System
07B04	Prestw-493	Omeprazole	Gastroenterology
08D06	Prestw-1410	Exemestane	Endocrinology
01C05	Prestw-24	Norethynodrel	Endocrinology
09G09	Prestw-708	Benzethonium chloride	Infectiology
10D08	Prestw-757	Chlorotrianisene	Endocrinology
03B05	Prestw-174	Alverine citrate salt	Neuromuscular
08H03	Prestw-632	Dipivefrin hydrochloride	Ophthalmology
01E08	Prestw-47	Ticlopidine hydrochloride	Hematology
06D11	Prestw-440	Epiandrosterone	Endocrinology
07H07	Prestw-1144	Mirtazapine	Central Nervous System
10H10	Prestw-799	Pridinol methanesulfonate salt	Central Nervous System
05A10	Prestw-329	Tacrine hydrochloride	Central Nervous System
01G06	Prestw-65	Diphenhydramine hydrochloride	Allergology 'Central Nervous System
09D02	Prestw-671	Dydrogesterone	Endocrinology
06H02	Prestw-1358	Vatalanib	Oncology
07B03	Prestw-492	Nitrofural	Infectiology
07F02	Prestw-531	Pirenperone	Central Nervous System
08H02	Prestw-1210	Alendronate sodium	Metabolism
07H10	Prestw-1817	Tazarotene	Dermatology
07C10	Prestw-509	Bromperidol	Central Nervous System
02C08	Prestw-1314	Pioglitazone	Endocrinology
09G04	Prestw-703	Famprofazone	Central Nervous System 'Metabolism
07C03	Prestw-502	Biperiden hydrochloride	Central Nervous System
10A08	Prestw-1140	Liranaftate	Infectiology
09F10	Prestw-699	Hexestrol	Endocrinology
03F02	Prestw-211	Piroxicam	Central Nervous System 'Hematology 'Metabol
02B04	Prestw-93	Azacyclonol	Central Nervous System
09A09	Prestw-1154	Nilvadipine	Cardiovascular
06F06	Prestw-455	Mebhydroline 1,5-naphtalenedisulfonate	Allergology
10E06	Prestw-765	Ethoxyquin	Metabolism
09G02	Prestw-701	Trihexyphenidyl-D,L hydrochloride	Central Nervous System
07H06	Prestw-555	Nifuroxazide	Infectiology 'Metabolism
08G02	Prestw-1506	Mizolastine	Allergology
05E07	Prestw-366	Ambroxol hydrochloride	Respiratory
08E07	Prestw-1331	Rimantadine hydrochloride	Infectiology
02B03	Prestw-92	Zimelidine dihydrochloride monohydrate	Central Nervous System
08B06	Prestw-1351	Tenatoprazole	Metabolism
07D09	Prestw-518	Budesonide	Endocrinology
10C04	Prestw-743	Medrysone	Metabolism
18B09	Prestw-1951	Eperisone HCl	Neuromuscular
05F11	Prestw-380	Clebopride maleate	Central Nervous System
03E06	Prestw-205	Tolfenamic acid	Central Nervous System 'Metabolism
06G08	Prestw-1157	Rifapentine	Infectiology
02B07	Prestw-96	Guanabenz acetate	Central Nervous System
19B02	Prestw-1996	Budralazine	Cardiovascular
02F05	Prestw-134	Diltiazem hydrochloride	Cardiovascular 'Hematology 'Metabolism

ID	Prestw.number	Chemical.name	Broad.Therapeutic.class
07B10	Prestw-499	Propafenone hydrochloride	Cardiovascular
06H07	Prestw-476	Primaquine diphosphate	Infectiology
10G06	Prestw-785	Dicumarol	Hematology
04B07	Prestw-256	Isotretinoin	Dermatology
02G02	Prestw-141	Verapamil hydrochloride	Cardiovascular
07A09	Prestw-488	Dosulepin hydrochloride	Central Nervous System
05G08	Prestw-387	Carbetapentane citrate	Central Nervous System 'Neuromuscular
04D11	Prestw-280	Quinidine hydrochloride monohydrate	Cardiovascular 'Infectiology
18C04	Prestw-1961	Methandrostenolone	Endocrinology
10B11	Prestw-1820	Amprenavir	Infectiology
06E06	Prestw-445	Cyclobenzaprine hydrochloride	Neuromuscular
10G10	Prestw-789	Drofenine hydrochloride	Neuromuscular
11E11	Prestw-850	Equilin	Endocrinology
08A08	Prestw-1139	Itraconazole	Infectiology 'Metabolism
06C08	Prestw-1393	Dibenzepine hydrochloride	Central Nervous System
11F03	Prestw-1454	Nylidrin	Cardiovascular
08C11	Prestw-1409	Etretinate	Dermatology
05F02	Prestw-371	Ketotifen fumarate	Allergology
05D10	Prestw-359	Dextromethorphan hydrobromide monohydrate	Central Nervous System
18H11	Prestw-2043	Artenimol	Infectiology
03H09	Prestw-238	Lomefloxacin hydrochloride	Infectiology
19E10	Prestw-2052	Dilevalol	Cardiovascular
19H04	Prestw-1940	Acetyl spiramycin	Infectiology
05F08	Prestw-377	Nafronyl oxalate	Cardiovascular 'Neuromuscular
12E07	Prestw-926	Idazoxan hydrochloride	Central Nervous System
06G05	Prestw-1323	Quetiapine hemifumarate	Central Nervous System
09A03	Prestw-1270	Gefitinib	Oncology
16C10	Prestw-1710	Ethoxzolamide	Ophthalmology 'Gastroenterology 'Central Ner
08B02	Prestw-571	Tetracaïne hydrochloride	Neuromuscular
11E02	Prestw-1455	Olanzapine	Central Nervous System
17D04	Prestw-1857	Oxiglutatione	Ophthalmology
19A03	Prestw-2045	Eletriptan	Central Nervous System
03E08	Prestw-1181	Tibolone	Endocrinology
01G07	Prestw-66	Minaprine dihydrochloride	Central Nervous System
02G11	Prestw-150	Dihydroergotamine tartrate	Central Nervous System
02F04	Prestw-133	Hydroxyzine dihydrochloride	Allergology 'Central Nervous System
04H02	Prestw-311	Ifenprodil tartrate	Cardiovascular
03C03	Prestw-182	Levamisole hydrochloride	Immunology 'Infectiology
04C06	Prestw-265	Dimenhydrinate	Allergology 'Central Nervous System
18D06	Prestw-2008	Azaribine	Oncology 'Dermatology
06C11	Prestw-430	Cisapride	Gastroenterology
19F05	Prestw-2019	Vonoprazan	Gastroenterology
01G04	Prestw-63	Nifedipine	Cardiovascular
08H05	Prestw-1463	Tomoxetine hydrochloride	Central Nervous System
19D11	Prestw-2067	Cyclofenil	Endocrinology
04C05	Prestw-264	Dyclonine hydrochloride	Neuromuscular
09H08	Prestw-717	Finasteride	Endocrinology
08E05	Prestw-1252	Butenafine hydrochloride	Infectiology 'Metabolism
18B06	Prestw-1945	Exifone	Central Nervous System

Conclusions

I strongly recommend to use the plate-wise normalised inhibition index in order to select the candidate molecules.

With an adjusted p-value of 0.05, 114 molecules are declared significant and could be considered as candidate for further characterisation.

Experimental design

- la manip consiste à faire un test colorimétrique
  - sur certains puits tout est négatif, sur d'autres tout est positiv
- mortalité des cellules
  - infectées non-traitées ("virus control")
  - 6 puits
- viabilité des cellules -> 100%
  - cellules non-infectées et non-traitées ("cell control")
  - 8 puits
- duplicat de traitement à l'arabidol 10µM
  - seuil par plaque
  - contrôle interne de la protection des cellules contre la mortalité viro-induite
- Valeurs de départ
  - mort cellulaire (moyenne sur 6 puits)
  - viabilité (moyenne sur 8 puits)
  - valeur du composé (monoplicat) : même concentration de 10µM par composé

Indice d'inhibition:

$$V = A/(D + A)$$

$$II =$$

On cherche des molécules inhibitrices avec l'EC50 aux alentours de 5µM -> la concentration standard de 10µM on espère se trouver au début du plateau.

A faire

- comparer les listes de composés avant / après normalisation par plaque
- boxplot ou violin par plaque avant / après
- standardisation des valeurs de viabilité par la médiane
- distribution de valeurs plaque par plaque –> vérifier si certaines plaques ont l'air d'avoir plus de 20 cibles (percentile75)
- Contrôles intra-plaques
  - utiliser médiane plutôt que moyenne ?
- Valeurs négatives:
  - fluctuations expérimentales ?

  - mortalité induite par le virus + la molécule ?
- Marquer sur les graphiques les valeurs de l'arbidol (contrôle interne en duplicat)
- boxplot des contrôles par plaque:
  - cell viability
  - viral