Mapping of Signalling Networks through Synthetic Genetic Interaction Analysis by RNAi

Bernd Fischer

November 3, 2022

Contents

1 Introduction

The package contains the data from the RNAi interaction screen reported in

> Thomas Horn, Thomas Sandmann, Bernd Fischer, Elin Axelsson, Wolfgang Huber, and Michael Boutros (2011): *Mapping of Signalling Networks through Synthetic Genetic Interaction Analysis by RNAi*, Nature Methods 8(4): 341-346.

This vignette shows the R code for ploting all figures in the paper.

2 Installation of the RNAinteractMAPK package

To install the package RNAinteractMAPK, you need a running version of R (www.r-project.org, version $\geq$ 2.13.0). After installing R you can run the following commands from the R command shell to install RNAinteractMAPK and all required packages.

```
> if (!requireNamespace("BiocManager", quietly=TRUE))
+     install.packages("BiocManager")
> BiocManager::install("RNAinteractMAPK")
```

3 Loading data and creating an RNAinteract object

The package RNAinteractMAPK is loaded by the following command.

```
> library("RNAinteractMAPK")
```

The package contains the data and the source code for reproducing all figures from the paper *Mapping of Signalling Networks through Synthetic Genetic Interaction Analysis by RNAi* by Horn, Sandmann, Fischer, Huber, Boutros, Mapping of Signalling Networks through Synthetic Genetic Interaction Analysis by RNAi, Nature Methods, 2011. Most of the underlying analysis is implemented in the software package RNAinteract. Please refer to the RNAinteract vignette for further details.
The main interaction matrix `PI` can be loaded by the following by the following code.

```
> data("Dmel2PPMAPK", package="RNAinteractMAPK")
> PI <- getData(Dmel2PPMAPK, type="pi", format="targetMatrix", screen="mean", withoutgroups = c("pos
```

If you want to reproduce a single figure you can now immediately jump to the respective section and run the code within one section. The source code shown in this PDF document shows the essential part of the analysis. Sometimes, some further formatting is needed to produce the plots as shown here; the code for that is not displayed in the PDF, but can be accessed in the Sweave file `RNAinteractMAPK.Rnw`. The R code is extracted from the Sweave file and written to a file `RNAinteractMAPK.R` by

```
> Stangle(system.file("doc", "RNAinteractMAPK.Rnw", package="RNAinteractMAPK"))
```

It is an R-script that reproduces all figures including any formatting steps. In addition it is easier to copy the code out of the R script to the R console when going through this manuscript.

4 Main analysis of the synthetic genetic interaction screen

4.1 Creation of an RNAinteract object from flat files

The raw data of the screen and the plate configuration can be found in the following directory

```
> inputpath = system.file("extdata", "Dmel2PPMAPK",package="RNAinteractMAPK")
> inputpath
```

The directory `inputpath` contains five text files:
`Platelist.txt`, `Targets.txt`, `Reagents.txt`, `TemplateDesign.txt`, `QueryDesign.txt`.
See the vignette of RNAinteract for further details on the file formats. The data and annotation are loaded and an *RNAinteract* object is created with the following command.

```
> Dmel2PPMAPK = createRNAinteractFromFiles(name="Pairwise PPMAPK screen", path = inputpath)
```

The object `Dmel2PPMAPK` contains two replicate screens and three readout channels.

```
> getScreenNames(Dmel2PPMAPK)

[1] "1" "2"

> getChannelNames(Dmel2PPMAPK)

[1] "nrCells"   "area"      "intensity"
```

4.2 Single RNAi effects and pairwise interactions

First, the single perturbation effects (called main effects) are estimated from the screen. For each template position and for each query reagent a main effect is estimated.

```
> Dmel2PPMAPK <- estimateMainEffect(Dmel2PPMAPK)
> Dmel2PPMAPK <- normalizeMainEffectQuery(Dmel2PPMAPK,batch=rep(1:4,each=48))
> Dmel2PPMAPK <- normalizeMainEffectTemplate(Dmel2PPMAPK, channel="intensity")
```

In our data, the main effect contained apparent time or plate dependent trends. We adjusted and removed them. The normalization of the main effects does not influence the subsequent estimation of the pairwise interaction, but it makes the main effects better comparable between replicates and different screens. Next, the pairwise interaction term was estimated.

```
> Dmel2PPMAPK <- computePI(Dmel2PPMAPK)
```

The object `Dmel2PPMAPK` contains two replicate screens. We summarized these two screens by taking the mean value for each measurement and added the mean screen as a new screen to the original screen.

```
> Dmel2PPMAPKmean <- summarizeScreens(Dmel2PPMAPK, screens=c("1","2"))
> Dmel2PPMAPK <- bindscreens(Dmel2PPMAPK, Dmel2PPMAPKmean)
```

Three different p-values (t-test with pooled variance morderation, limma, and multivariate Hotelling T^2 test) were computed by

```
> library(qvalue)
> p.adj.fct <- function(x) {
+   I = which(is.finite(x))
+   qjob = qvalue(x[I])
+   q.value = rep(NA,length(x))
+   q.value[I] = qjob$qvalues
+ }
> Dmel2PPMAPK <- computePValues(Dmel2PPMAPK, p.adjust.function = p.adj.fct, verbose = FALSE)
> Dmel2PPMAPKT2 <- computePValues(Dmel2PPMAPK, method="HotellingT2", p.adjust.function = p.adj.fct,
> Dmel2PPMAPKlimma <- computePValues(Dmel2PPMAPK, method="limma", p.adjust.function = p.adj.fct, ver
```

independently for each screen and each channel. The p-values are corrected for multiple testing by the method of Storey, 2003. If the p.adjust.function is not specified, the method of Benjamini-Hochberg is applied.

5 Main figures

Figure 1 abc: Genetic interaction surfaces

The code to generate these is the same as that for Suppl. Figure S5, shown later in this vignette.

Figure 2: Heatmap of genetic interactions

See Suppl. Figures S6, S7, and S8.

Figure 4 b: Node degree distribution

See Suppl. Figure S11.

Figure 4 cd, Fig. 5 a: double RNAi plots for CG10376, Gap1, Cka

Double RNAi plots are plotted for three different genes.

```
> plotDoublePerturbation(Dmel2PPMAPK, screen="mean", channel="nrCells", target="CG10376",
+                        main="CG10376", range=c(-1, 0.05),
+                        axisOnOrigin=FALSE, drawBox=FALSE, show.labels="q.value",
+                        xlab="rel. nuclear count [log2]", ylab="rel. nuclear count [log2]")

> plotDoublePerturbation(Dmel2PPMAPK, screen="mean", channel="nrCells", target="Gap1",
+                        main="Gap1 (CG6721)", range=c(-2.3, 1.0),
+                        axisOnOrigin=FALSE, drawBox=FALSE, show.labels="q.value",
+                        xlab="rel. nuclear count [log2]", ylab="rel. nuclear count [log2]")
```

```
> plotDoublePerturbation(Dmel2PPMAPK, screen="mean", channel="nrCells", target="Cka",
+                        main="Cka (CG7392)", range=c(-2.3, 1.0),
+                        axisOnOrigin=FALSE, drawBox=FALSE, show.labels="q.value",
+                        xlab="rel. nuclear count [log2]", ylab="rel. nuclear count [log2]")
```

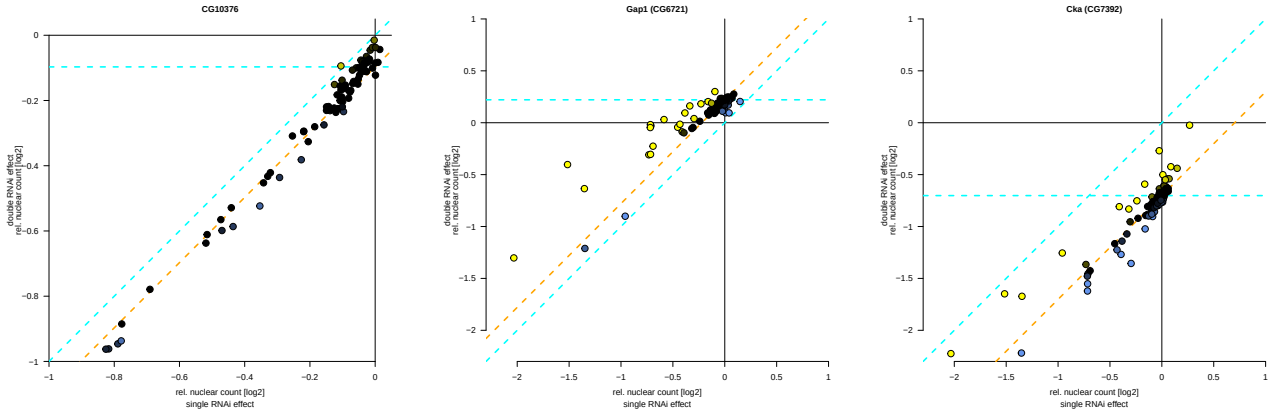


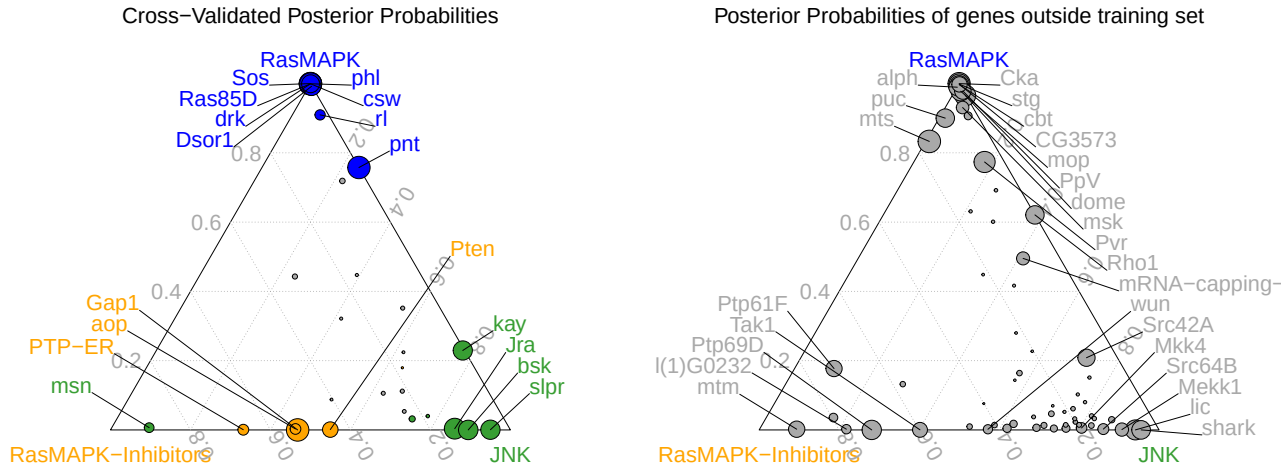
Figure 4 ef: Classification

To train a classifier a training set is defined. A sparse linear discriminant analysis is trained individually for each channel and each screen. By leave-one-out cross validation the cross validated posterior probability for each gene in the training set is computed. This is an estimate of the predictive power of each classifier. The classifiers are combined by averaging their posterior probabilities.

```
> traingroups = list(RasMapK = c("csw","drk","Sos","Ras85D","phl","Dsor1","rl","pnt"),
+   RasMapKInh = c("Gap1","PTP-ER","Mkp3","aop","Pten"),
+   JNK = c("Gadd45","Btk29A","msn","slpr","bsk","Jra","kay"),
+   others = c("CG10376","CG42327","CG13197","CG12746","CG3530","CG17029","CG17598",
+     "CG9391","CG9784","CG10089"))
> groupcol = c(RasMapK = "blue", RasMapKInh = "orange", JNK = "green", others = "white")
> sgi <- sgisubset(Dmel2PPMAPK, screen=c("1","2"))
> CV <- MAPK.cv.classifier(sgi, traingroups)
```

The cross-validated posterior probabilities are depicted by a ternary plot. In a ternary plot one can only display probabilities for three classes. Since a fourth doubt class is included in the training set, the assignment probability to the *doubt* class is depicted as the circle diameter. The smaller the circle diameter, the more likely the gene belongs to the doubt class. The distance to the apexes shows the assignment probability of a gene to one of the three classes given that the gene does not belong to the doubt class. In the same manner the posterior probabilities of the classifier for *new* genes outside the training set are shown in an extra plot.

```
> MAPK.plot.classification(CV$CVposterior, y=CV$y,
+                          classes = c("RasMapKInh", "JNK", "RasMapK"),
+                          classnames = c("RasMAPK-Inhibitors","JNK","RasMAPK"),
+                          classcol = c("orange", "#389e33", "blue"),
+                          main = "Cross-Validated Posterior Probabilities",
+                          textToLeft = c("Ras85D","Sos","Dsor1")
+                      )

> prediction <- MAPK.predict.classification(sgi, traingroups)
> MAPK.plot.classification(prediction$posteriornew,
+                          classes = c("RasMapKInh", "JNK", "RasMapK"),
+                          classnames = c("RasMAPK-Inhibitors","JNK","RasMAPK"),
+                          classcol = c("orange", "#389e33", "blue"),
+                          main = "Posterior Probabilities of genes outside training set",
+                          textToRight = c("stg","Cka"),
+                          threshText = 0.4)
```

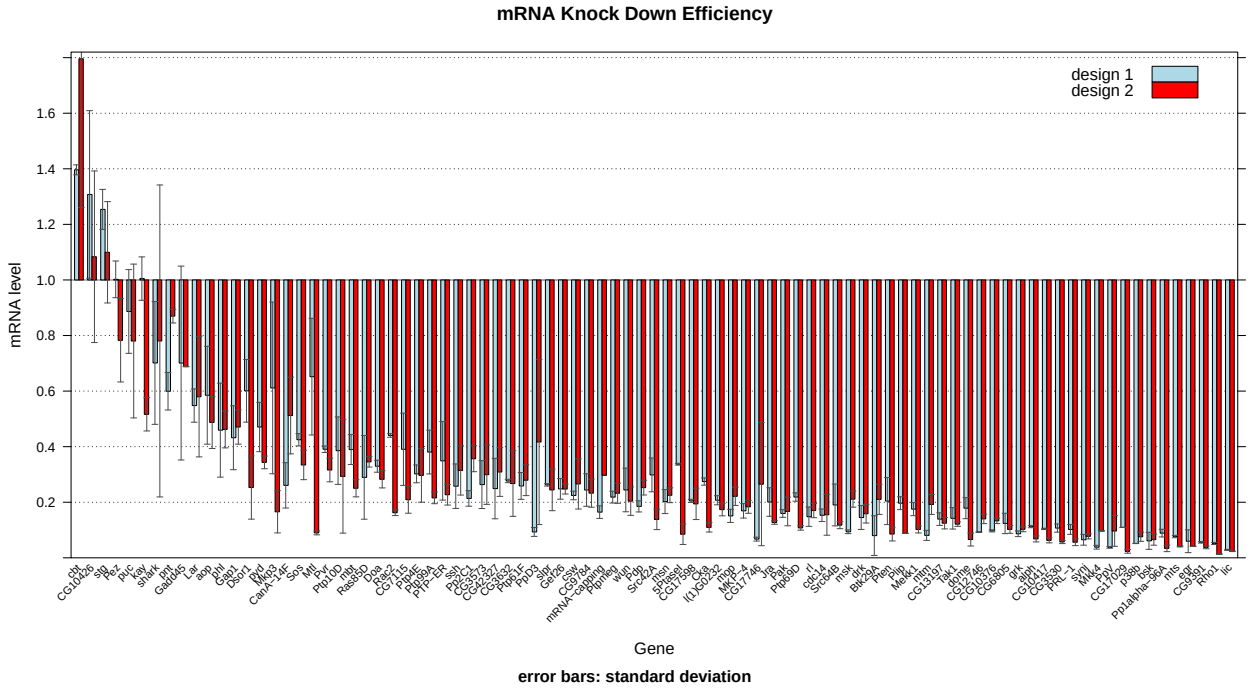


6 Supplementary Figures

Suppl. Figure S1: Validation of mRNA levels for single RNAi knockdowns

For each gene there were two independent dsRNA designs. The mRNA level of the targeted genes was measured by qRT-PCR 5 days after infection. The table `T` contains mean and stderror over replicates of qRT-PCR experiments for each RNAi reagent. Each row contains the mRNA levels for two RNAi designs.

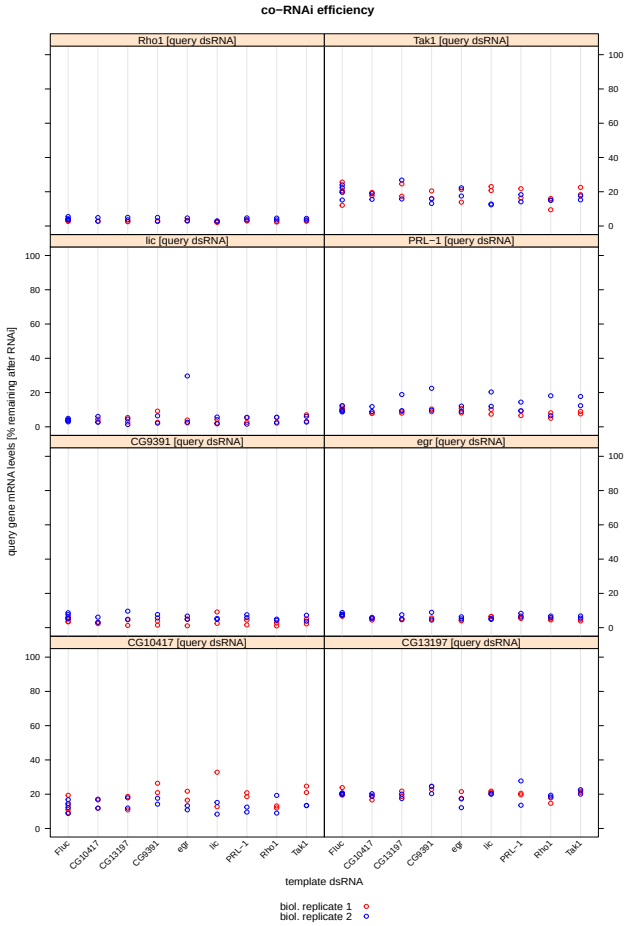
```
> data("mRNAsingleKDefficiency", package = "RNAinteractMAPK")
> hh <- apply(as.matrix(mRNAsingleKDefficiency[,c("MeanDesign1","MeanDesign2")]),1,mean)
> I <- order(-hh)
> D <- data.frame(mRNAlevel=c(mRNAsingleKDefficiency[,"MeanDesign1"],mRNAsingleKDefficiency[,"MeanDe
+                 sd=c(mRNAsingleKDefficiency[,"StderrDesign1"],mRNAsingleKDefficiency[,"StderrDesig
+                 Gene=factor(c(mRNAsingleKDefficiency[,"Symbol"],mRNAsingleKDefficiency[,"Symbol"])
+                   levels=mRNAsingleKDefficiency[I,"Symbol"]),
+                 design=factor(rep(c("design 1","design 2"),each=89),
+                   levels=c("design 1","design 2")))
> library(lattice)
> bc <- barchart(mRNAlevel ~ Gene, data = D,
+          layout = c(1,1), origin = 1, stack = FALSE,
+          group = D$design, col = c("lightblue", "red"),
+          auto.key = list(points=FALSE, rectangles = TRUE, corner = c(0.97,0.97)),
+          par.settings = list(superpose.polygon = list(col=c("lightblue", "red"))),
+          xlab="Gene", ylab = "mRNA level",
+          sub = "error bars: standard deviation", col.sub = "gray30",
+          scales = list(x = list(cex=0.7,rot = 45), y=list(tick.number=9)),
+          ylim=c(0,1.82), main = "mRNA Knock Down Efficiency",
+          panel=function(...) {
+            panel.abline(h=seq(0.2, 1.9, by=0.2),lty="dotted")
+            panel.abline(h=1)
+            panel.barchart(...)
+            x = (as.integer(D$Gene) + 0.375 * (as.integer(D$design) - (2 + 1)/2))
+            y1 = D$mRNAlevel - D$sd
+            y2 = D$mRNAlevel + D$sd
+            panel.arrows(x,y1,x,y2,code=3,angle=90,length=0.03,col="gray30")
+          } )
>
```



Suppl. Figure S2: Validation of mRNA levels for double RNAi knockdowns

The mRNA level of 8 genes was measured by qRT-PCR after RNAi knockdown of these genes in the presence of a second RNAi knockdown. The experiment was repeated twice (passage 4, passage 42).

```
> data("mRNAdoubleKDefficiency", package="RNAinteractMAPK")

> dp <- dotplot(100-(mRNAdoubleKDefficiency$RNAi) ~ mRNAdoubleKDefficiency$template |
+   mRNAdoubleKDefficiency$query, groups = mRNAdoubleKDefficiency$passage,  ylim=c(-5,105),
+               main="co-RNAi efficiency", layout = c(2,4),
+               xlab="template dsRNA", ylab="query gene mRNA levels [% remaining after RNAi]",
+               scales=list(x=list(rot=45)),
+               key = simpleKey(c("biol. replicate 1", "biol. replicate 2"),space = "bottom"))
```



Suppl. Figure S3: Single RNAi phenotypes

Load the data and the plate annotation. The number-of-cells and area features are extracted from the feature set and features are log-transformed.

```
> data("singleKDphenotype", package="RNAinteractMAPK")
> singleKDphenotype[,"nrcells"] <- log2(singleKDphenotype[,"nrcells"])
> singleKDphenotype[,"area"] <- log2(singleKDphenotype[,"area"])
> # inputfile <- system.file("extdata","singleKnockDownPhenotype/annoSingleKnockDownEffect.csv",
> #                          package="RNAinteractMAPK")
> # Anno <- read.csv(inputfile,sep=";",stringsAsFactors = FALSE)
> # Anno$Symbol <- substr(Anno$Symbol,1,12)
> # exclude csw, PTP-ER, and puc, because they are measured twice
> # F <- factor(Anno$Symbol[!(Anno$Symbol %in% c("csw_exp2","PTP-ER_exp2","puc_exp2"))])
> # F = factor(Anno$Symbol, levels = unique(Anno$Symbol[!(Anno$Symbol %in% c("csw_exp2","PTP-ER_exp2
```

The experiment consisted of 4 plates. Within each plate 4 neighboring wells contained the same RNAi-reagent. Plates 1 and 2 contained the first RNAi design, plates 3 and 4 the second RNAi design for each gene. The data was first reorganized in a matrix with wells in the rows and the 4 plates in columns. The knockdown csw, PTP-ER, and puc were measured twice in the experiment. The second experiment was removed. The data was normalized by subtracting the shorth on each plate. For each gene on each plate the mean value was computed for the four replicates.

```
> Mean <- list()
> DRho1 <- list()
> Dpnt <- list()
> for (ds in colnames(singleKDphenotype)) {
+   D <- matrix(singleKDphenotype[,ds],nr=384,nc=4)
+   Mean[[ds]] <- matrix(0.0,nr=nlevels(singleKDphenotypeAnno$Symbol),nc=4)
```

```
+   row.names(Mean[[ds]]) = levels(singleKDphenotypeAnno$Symbol)
+   DRho1[[ds]] <- D[which(singleKDphenotypeAnno$Symbol == "Rho1"),]
+   Dpnt[[ds]] <- D[which(singleKDphenotypeAnno$Symbol == "pnt"),]
+   for (i in 1:4) {
+     mm <- genefilter:::shorth(D[,i],tie.limit=0.5)
+     DRho1[[ds]][,i] <- DRho1[[ds]][,i] - mm
+     Dpnt[[ds]][,i] <- Dpnt[[ds]][,i] - mm
+     Mean[[ds]][,i] <- tapply(D[,i]-mm,singleKDphenotypeAnno$Symbol,mean)
+   }
+ }
```

Rho1 shows different phenotype than control for nrCells feature.

```
> t.test(as.vector(DRho1[["nrcells"]]))

        One Sample t-test

data:  as.vector(DRho1[["nrcells"]])
t = -105.55, df = 15, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -1.209427 -1.161549
sample estimates:
mean of x
-1.185488
```

Rho1 shows different phenotype than control for area feature.

```
> t.test(as.vector(DRho1[["area"]]))

        One Sample t-test

data:  as.vector(DRho1[["area"]])
t = 65.777, df = 15, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 0.5261616 0.5614030
sample estimates:
mean of x
0.5437823
```

pnt shows different phenotype than control for nrCells feature.

```
> t.test(as.vector(Dpnt[["nrcells"]]))

        One Sample t-test

data:  as.vector(Dpnt[["nrcells"]])
t = -45.665, df = 15, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -1.598643 -1.456062
sample estimates:
mean of x
-1.527352
```

pnt shows different phenotype than control for area feature.

```
> t.test(as.vector(Dpnt[["area"]]))
```

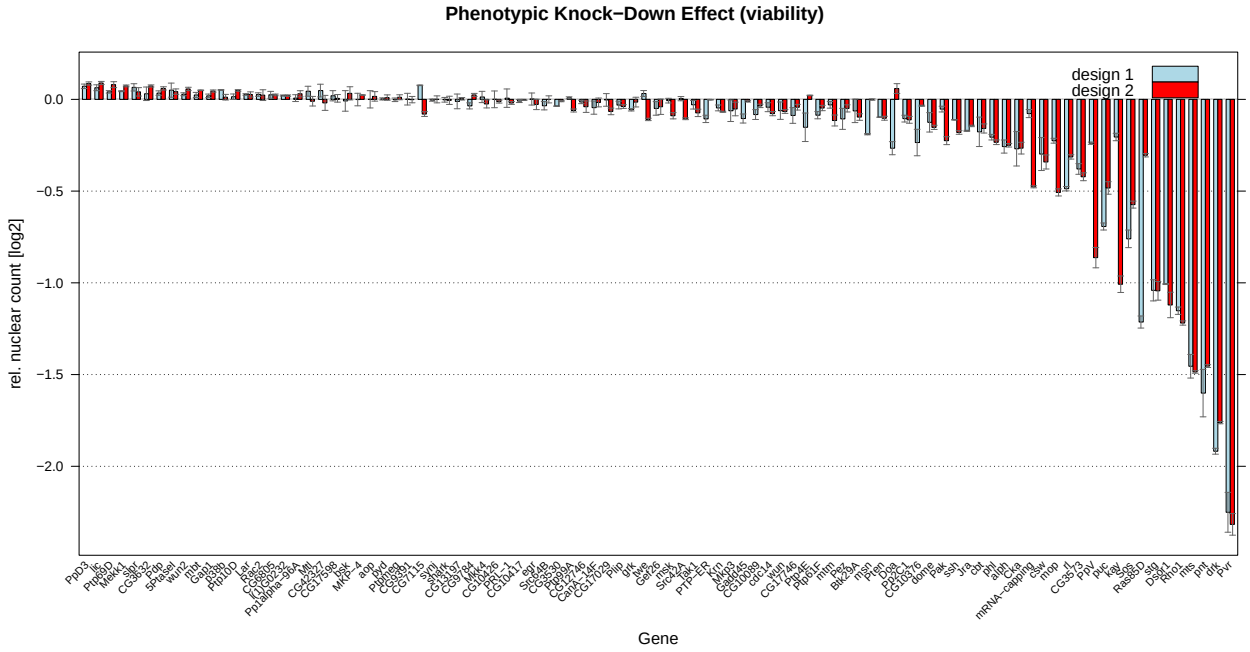
```
        One Sample t-test

data:  as.vector(Dpnt[["area"]])
t = -40.463, df = 15, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -0.4314376 -0.3882589
sample estimates:
 mean of x
-0.4098483
```

Mean and standard deviation were computed for each RNAi design over two replicates.

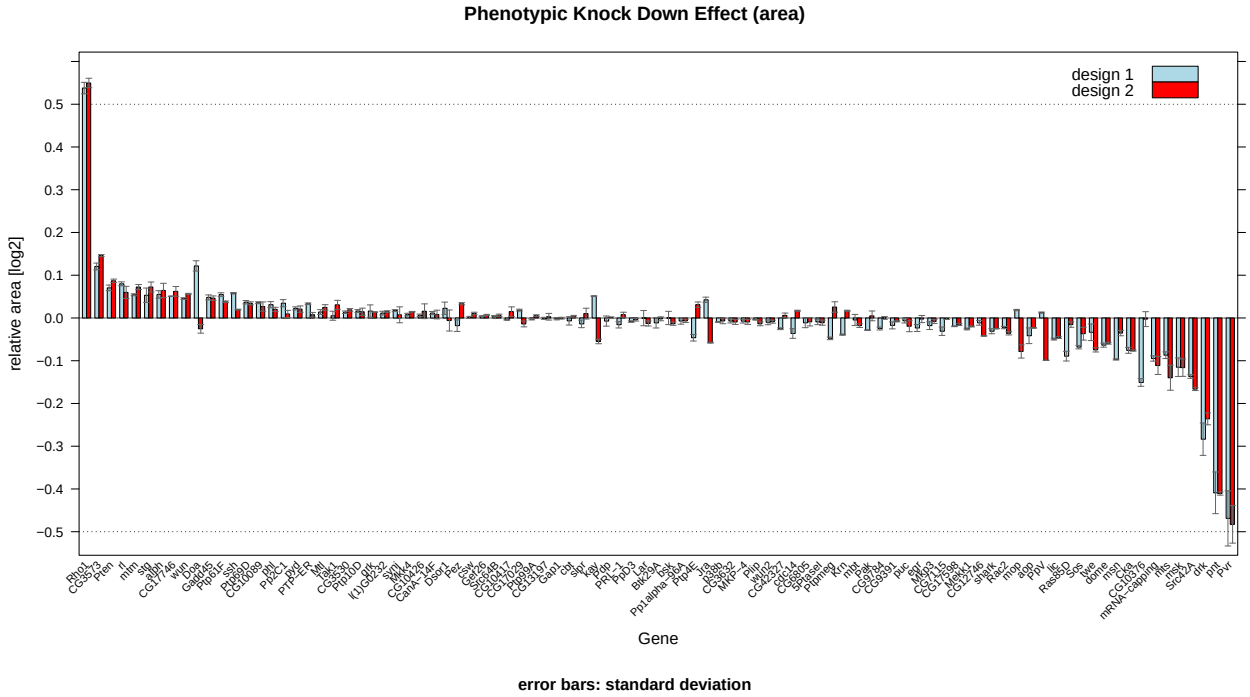
```
> M2 <- list()
> SD2 <- list()
> M3 <- list()
> SD3 <- list()
> for (ds in colnames(singleKDphenotype)) {
+   M2[[ds]] <- matrix(NA,nr=nrow(Mean[[ds]]),nc=2)
+   SD2[[ds]] <- matrix(NA,nr=nrow(Mean[[ds]]),nc=2)
+   M2[[ds]][,1] <- apply(Mean[[ds]][,1:2],1,mean)
+   M2[[ds]][,2] <- apply(Mean[[ds]][,3:4],1,mean)
+   SD2[[ds]][,1] <- apply(Mean[[ds]][,1:2],1,sd)
+   SD2[[ds]][,2] <- apply(Mean[[ds]][,3:4],1,sd)
+
+   M3[[ds]] <- apply(M2[[ds]],1,mean)
+   SD3[[ds]] <- apply(M2[[ds]],1,sd)
+ }
```

A barplot was plotted for both features.

```
> ds <- "nrcells"
> I <- order(-M3[[ds]])
> D <- data.frame(level=c(M2[[ds]][,1],M2[[ds]][,2]),sd=c(SD2[[ds]][,1],SD2[[ds]][,2]),
+                 Gene=factor(rep(row.names(Mean[[ds]]),2), levels=row.names(Mean[[ds]])[I]),
+                 design=factor(rep(c("design 1","design 2"),each=nrow(Mean[[ds]])),
+                   levels=c("design 1","design 2")))
> bc <- barchart(level ~ Gene, data = D,
+                layout = c(1,1), origin=0, stack = FALSE,
+                groups=D$design, col = c("lightblue", "red"),
+                auto.key = list(points = FALSE, rectangles = TRUE, corner=c(0.97,0.97)),
+                par.settings = list(superpose.polygon=list(col=c("lightblue", "red"))),
+                ylab =  "rel. nuclear count [log2]",
+                scales = list(x = list(cex=0.7,rot = 45), y=list(tick.number=9)),
+                xlab="Gene",
+                sub = "error bars: standard deviation", col.sub = "gray40",
+                main = "Phenotypic Knock-Down Effect (viability)",
+                panel=function(...) {
+                  panel.abline(h=c(-2,-1.5,-1,-0.5,0,0.5),lty="dotted")
+                  panel.abline(h=0)
+                  pbc = panel.barchart(...)
+                  x = (as.integer(D$Gene) + 0.375 * (as.integer(D$design) - (2 + 1)/2))
+                  y1 = D$level - D$sd
+                  y2 = D$level + D$sd
+                  panel.arrows(x,y1,x,y2,code=3,angle=90,length=0.03,col="gray40")
+                } )
```



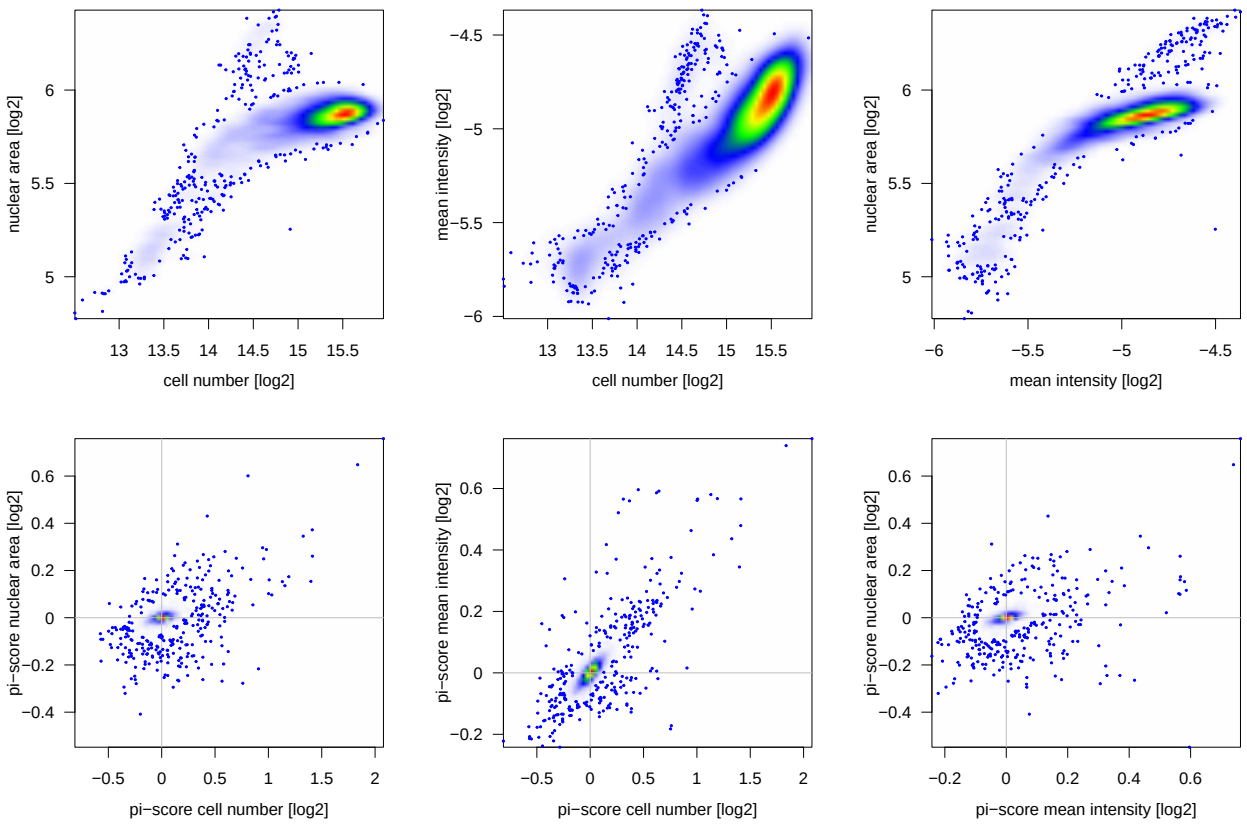

```
> ds <- "area"
> I <- order(-M3[[ds]])
> D <- data.frame(level=c(M2[[ds]][,1],M2[[ds]][,2]),sd=c(SD2[[ds]][,1],SD2[[ds]][,2]),
+                 Gene=factor(rep(row.names(Mean[[ds]]),2), levels=row.names(Mean[[ds]])[I]),
+                 design=factor(rep(c("design 1","design 2"),each=nrow(Mean[[ds]])),
+                   levels=c("design 1","design 2")))
> bc <- barchart(level ~ Gene, data = D,
+                layout = c(1,1), origin=0,stack = FALSE,
+                groups=D$design, col = c("lightblue", "red"),
+                auto.key = list(points = FALSE, rectangles = TRUE, corner=c(0.97,0.97)),
+                par.settings = list(superpose.polygon=list(col=c("lightblue", "red"))),
+                ylab = "relative area [log2]",
+                scales = list(x = list(cex=0.7,rot = 45), y=list(tick.number=9)),
+                xlab="Gene",
+                sub = "error bars: standard deviation", col.sub = "gray40",
+                main = "Phenotypic Knock Down Effect (area)",
+                panel=function(...) {
+                  panel.abline(h=c(-2,-1.5,-1,-0.5,0,0.5),lty="dotted")
+                  panel.abline(h=0)
+                  pbc = panel.barchart(...)
+                  x = (as.integer(D$Gene) + 0.375 * (as.integer(D$design) - (2 + 1)/2))
+                  y1 = D$level - D$sd
+                  y2 = D$level + D$sd
+                  panel.arrows(x,y1,x,y2,code=3,angle=90,length=0.03,col="gray40")
+          } )
>
```



Suppl. Figure S4: Correlation of different features

The readout `D` normalized by the time-effect and the pairwise interaction score `PI` is extracted from the `Dmel2PPMAPK`-object. To speed up drawing, only 5000 randomly selected gene pairs are plotted.

```
> D <- getData(Dmel2PPMAPK,type="data",screen="mean")[,1,c("nrCells","area","intensity")]
> PI <- getData(Dmel2PPMAPK,type="pi",screen="mean")[,1,c("nrCells","area","intensity")]
> set.seed(491127)
> I <- sample(1:nrow(D),5000)

> MAPK.smooth.scatter(D[I,"nrCells"], D[I,"area"], respect=FALSE, nrpoints=300,
+                 xlab="cell number [log2]", ylab="nuclear area [log2]")

> MAPK.smooth.scatter(D[I,"nrCells"], D[I,"intensity"], respect=FALSE, nrpoints=300,
+                 xlab="cell number [log2]", ylab="mean intensity [log2]")

> MAPK.smooth.scatter(D[I,"intensity"], D[I,"area"], respect=FALSE, nrpoints=300,
+                 xlab="mean intensity [log2]", ylab="nuclear area [log2]")

> MAPK.smooth.scatter(PI[I,"nrCells"], PI[I,"area"], respect=FALSE, nrpoints=300,
+                 xlab="pi-score cell number [log2]", ylab="pi-score nuclear area [log2]")

> MAPK.smooth.scatter(PI[I,"nrCells"], PI[I,"intensity"], respect=FALSE, nrpoints=300,
+                 xlab="pi-score cell number [log2]", ylab="pi-score mean intensity [log2]")

> MAPK.smooth.scatter(PI[I,"intensity"], PI[I,"area"], respect=FALSE, nrpoints=300,
+                 xlab="pi-score mean intensity [log2]", ylab="pi-score nuclear area [log2]")
```



Suppl. Figure S5:

For 6×6 gene pairs a dilution series was done. For each gene pair, 8×8 different pairs of dsRNA concentration (0 ng,10 ng,20 ng,40 ng,80 ng,100 ng,120 ng,140 ng) were tested. The readout is first reshaped into a 5-dimensional array `A` (features×gene1×gene2×concentration1×concentration2).

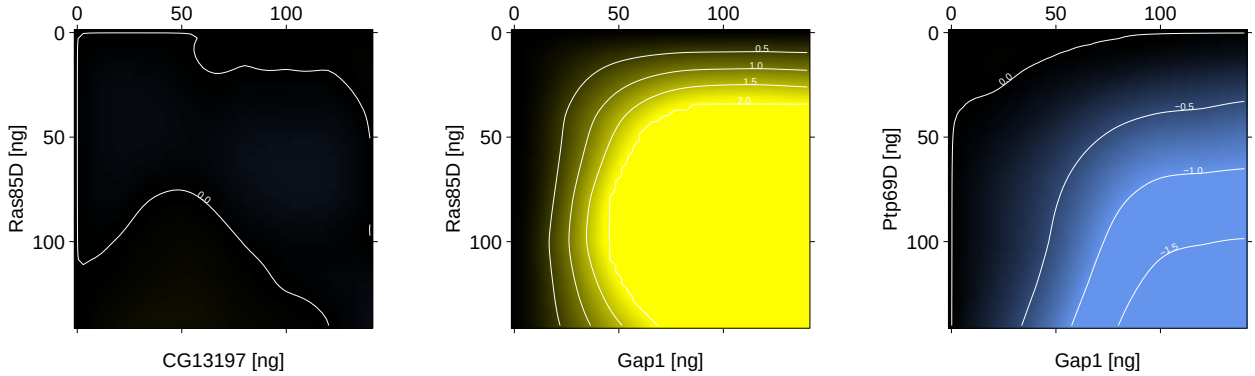
```
> data("dsRNAiDilutionSeries")
> dsRNAiDilutionSeries[,"nrCells"] <- log2(dsRNAiDilutionSeries[,"nrCells",drop=FALSE])
> A <- MAPK.screen.as.array(dsRNAiDilutionSeries, dsRNAiDilutionSeriesAnno)
```

To reduce measurement noise the 8×8 concentration dependent feature surfaces are smoothed by thin plate splines. Cross validation is performed to estimate the degree of freedom for each feature surface. Since this process is quite time consuming you don't need to run the next code chunk, but can load a precomputed tables as is shown in the subsequent code chunk.

```
> set.seed(491127)
> # warning: Very time consuming. Go on with next code chunk and load precomputed values.
> dsRNAiDilutionSeriesDF <- MAPK.cv.TPS(A)
> write.table(dsRNAiDilutionSeriesDF, file="Figure-S06-resCV.txt", sep="\t", quote=FALSE)
```

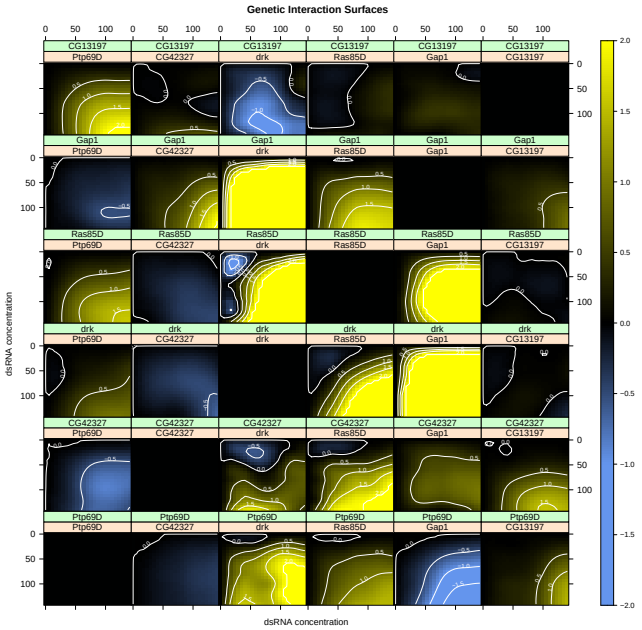
Together with the data a matrix `dsRNAiDilutionSeriesDF` with precomputed degrees of freedom is already loaded.
For the high resolution images in Figure 1, a thin plate spline model was fitted in the interaction surface. The interactions were estimated and the surface was screend on a 50×50 grid.

```
> TPSmodel <- MAPK.estimate.TPS(A, DF=dsRNAiDilutionSeriesDF, n.out=50)

> print(MAPK.plot.TPS.single(gene1="Ras85D", gene2="CG13197", TPSmodel=TPSmodel, range=c(-2,2)))

> print(MAPK.plot.TPS.single(gene1="Ras85D", gene2="Gap1", TPSmodel=TPSmodel, range=c(-2,2)))

> print(MAPK.plot.TPS.single(gene1="Ptp69D", gene2="Gap1", TPSmodel=TPSmodel, range=c(-2,2)))
```



To show a table with all 6×6 genetic interaction surfaces, we screen the estimated interaction surfaces with a lower rate (25×25 grid).

```
> print(MAPK.plot.TPS.all(TPSmodel=TPSmodel, range=c(-2,2)))
```

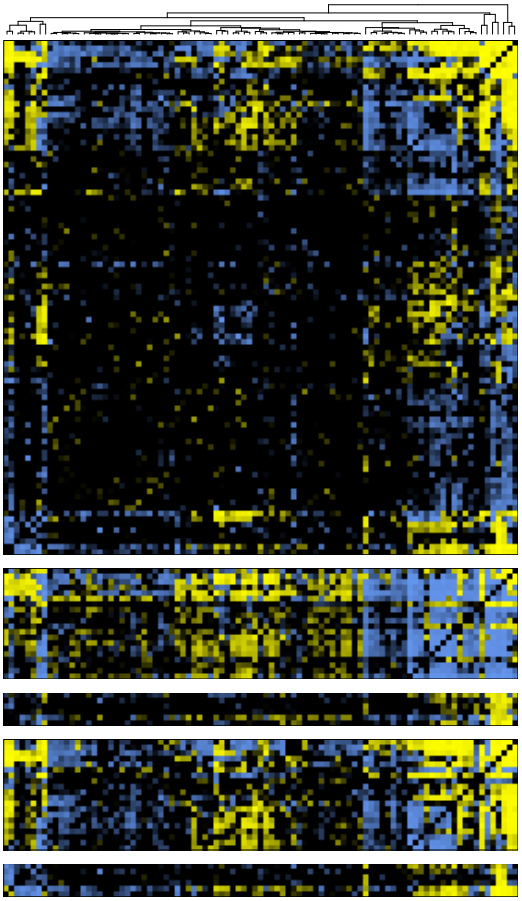


Suppl. Figure S6, S7, S8:

All heatmaps show the same ordering of genes which is derived from the clustering of the genetic interaction matrix for the cell number feature. In Figure 2, the number-of-cells interaction matrix is plotted completely. From the area and intensity matrix only the rows containing the RasMAPK and JNK pathway members are shown.

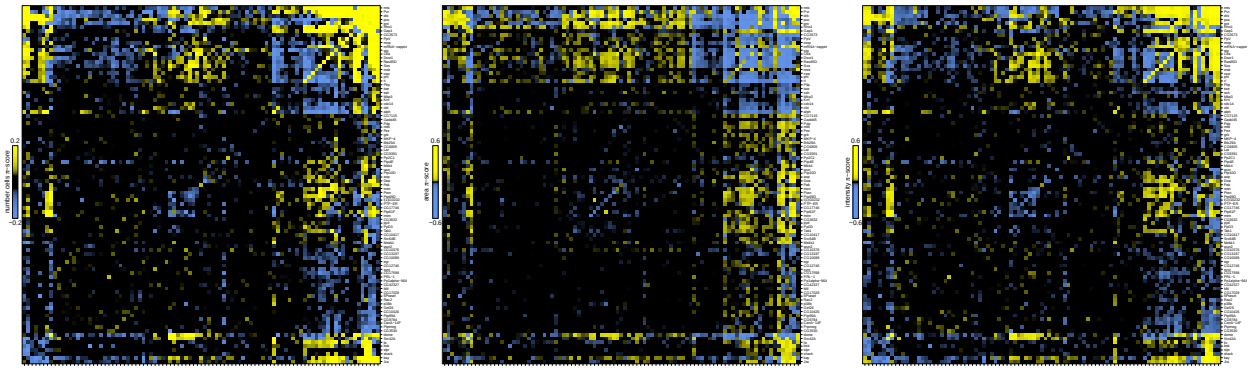
```
> PInrcells <- getData(Dmel2PPMAPK,type="pi",format="targetMatrix",screen="mean",
+                      channel="nrCells",withoutgroups=c("pos","neg"))
> PIarea <- getData(Dmel2PPMAPK,type="pi",format="targetMatrix",screen="mean",
+                   channel="area",withoutgroups=c("pos","neg"))
> PIintensity <- getData(Dmel2PPMAPK,type="pi",format="targetMatrix",screen="mean",
+                        channel="intensity",withoutgroups=c("pos","neg"))
> PC = embedPCA(Dmel2PPMAPK, screen="mean", channel="nrCells", dim=4, withoutgroups=c("pos","neg"))
> hc = hclust(dist(PC))
> hc <- RNAinteract:::swaptree(hc, 92)
> subset1 <- row.names(PInrcells)[hc$order[1:6]]
> subset2 <- row.names(PInrcells)[hc$order[74:93]]
> allgenes <- row.names(PInrcells)[hc$order]
```

```
> RNAinteract:::grid.sgiDendrogram(hc = hc)

> MAPK.plot.heatmap.raster(PInrcells, subset=allgenes, hc.row = hc, hc.col=hc, pi.max=0.05)

> MAPK.plot.heatmap.raster(PIarea, subset=subset2, hc.row = hc, hc.col=hc, pi.max=0.02)

> MAPK.plot.heatmap.raster(PIarea, subset=subset1, hc.row = hc, hc.col=hc, pi.max=0.02)

> MAPK.plot.heatmap.raster(PIintensity, subset=subset2, hc.row = hc, hc.col=hc, pi.max=0.02)

> MAPK.plot.heatmap.raster(PIintensity, subset=subset1, hc.row = hc, hc.col=hc, pi.max=0.02)
```



The supplemental figures S6, S7, and S8 show the complete interaction matrices for the three features nrCells, area, and intensity. The ordering of genes is the same in all three heatmaps. They are ordered according to a clustering of the nrCells interaction map.

```
> grid.sgiHeatmap(PInrcells, pi.max=0.2,
+                 main=expression(paste("number cells ", pi,"-score")), hc.row = hc, hc.col = hc)

> grid.sgiHeatmap(PIarea, pi.max=0.5,
+                 main=expression(paste("area ", pi,"-score")), hc.row = hc, hc.col = hc)

> grid.sgiHeatmap(PIintensity, pi.max=0.5,
+                 main=expression(paste("intensity ", pi,"-score")), hc.row = hc, hc.col = hc)
```

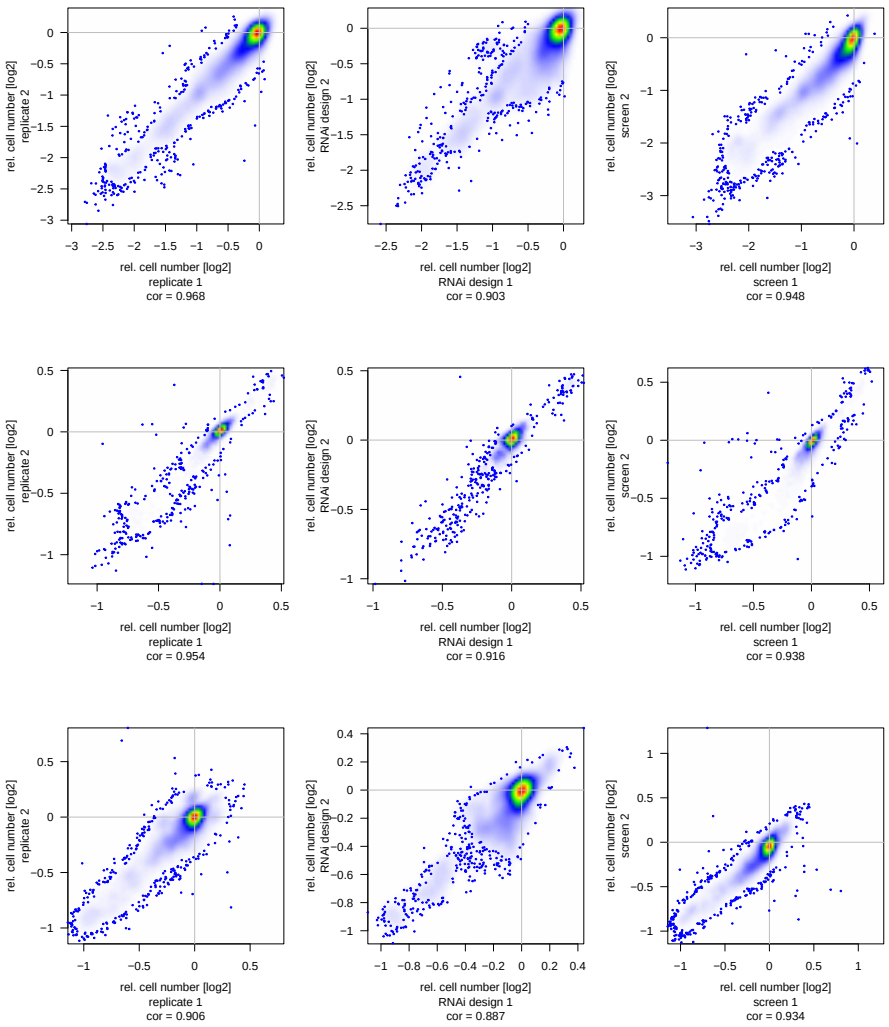


Suppl. Figure S9: Correlation of features across replicates

Scatter plots of read-out for number-of-cells feature. A scatter plot is shown for within-screen replicates, between independent dsRNA designs, and for between-screen replicates.

```
> D <- getData(Dmel2PPMAPK, normalized = TRUE)
> Main <- getMainNeg(Dmel2PPMAPK)
> RepData <- getReplicateData(Dmel2PPMAPK, screen="1", channel="nrCells", type="data",
+                             normalized = TRUE)
> IndDesignData <- getIndDesignData(Dmel2PPMAPK, screen="1", channel="nrCells", type="data",
+                                   normalized = TRUE)

> MAPK.smooth.scatter(RepData$x-Main["1","nrCells"], RepData$y-Main["1","nrCells"],
+                     nrpoints=300,
+                   xlab=c("rel. cell number [log2]", "replicate 1",
+                     sprintf("cor = %0.3f", cor(RepData$x,RepData$y))),
+                   ylab=c("rel. cell number [log2]", "replicate 2"),respect=TRUE)

> MAPK.smooth.scatter(IndDesignData$x-Main["1","nrCells"], IndDesignData$y-Main["1","nrCells"],
+                     nrpoints=300,
+                   xlab=c("rel. cell number [log2]", "RNAi design 1",
+                     sprintf("cor = %0.3f", cor(IndDesignData$x,IndDesignData$y))),
+                   ylab=c("rel. cell number [log2]", "RNAi design 2"),respect=TRUE)

> MAPK.smooth.scatter(D[,"1","nrCells"]-Main["1","nrCells"], D[,"2","nrCells"]-Main["2","nrCells"],
+                     nrpoints=300,
+                   xlab=c("rel. cell number [log2]", "screen 1",
+                     sprintf("cor = %0.3f", cor(D[,"1","nrCells"],D[,"2","nrCells"]))),
+                   ylab=c("rel. cell number [log2]", "screen 2"),respect=TRUE)
```

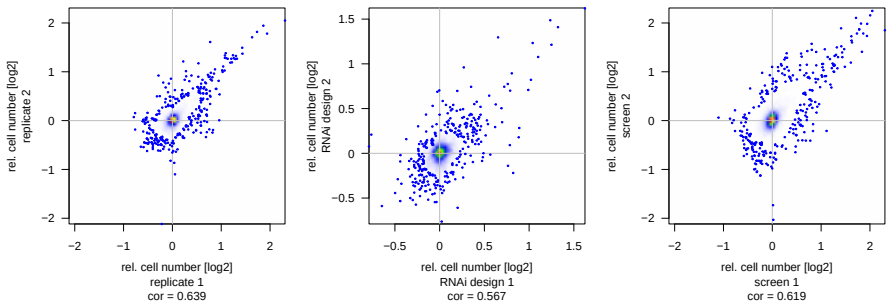
In the same way the scatter plots for area and intensity features were generated.



Suppl. Figure S10: Correlation of interaction scores across replicates

Same as Figure S9, but for interaction scores.

```
> D <- getData(Dmel2PPMAPK, type="pi", normalized = TRUE)
> RepData <- getReplicateData(Dmel2PPMAPK, screen="1", channel="nrCells", type="pi",
+                             normalized = TRUE)
> IndDesignData <- getIndDesignData(Dmel2PPMAPK, screen="1", channel="nrCells", type="pi",
+                                   normalized = TRUE)

> MAPK.smooth.scatter(RepData$x, RepData$y,nrpoints=300,
+                 xlab=c("rel. cell number [log2]", "replicate 1",
+                   sprintf("cor = %0.3f", cor(RepData$x, RepData$y))),
+                 ylab=c("rel. cell number [log2]", "replicate 2"),respect=TRUE)

> MAPK.smooth.scatter(IndDesignData$x, IndDesignData$y,nrpoints=300,
+                 xlab=c("rel. cell number [log2]", "RNAi design 1",
+                   sprintf("cor = %0.3f", cor(IndDesignData$x, IndDesignData$y))),
+                 ylab=c("rel. cell number [log2]", "RNAi design 2"),respect=TRUE)
>

> MAPK.smooth.scatter(D[,"1","nrCells"], D[,"2","nrCells"], nrpoints=300,
+                 xlab=c("rel. cell number [log2]", "screen 1",
+                   sprintf("cor = %0.3f", cor(D[,"1","nrCells"], D[,"2","nrCells"]))),
+                 ylab=c("rel. cell number [log2]", "screen 2"),respect=TRUE)
```

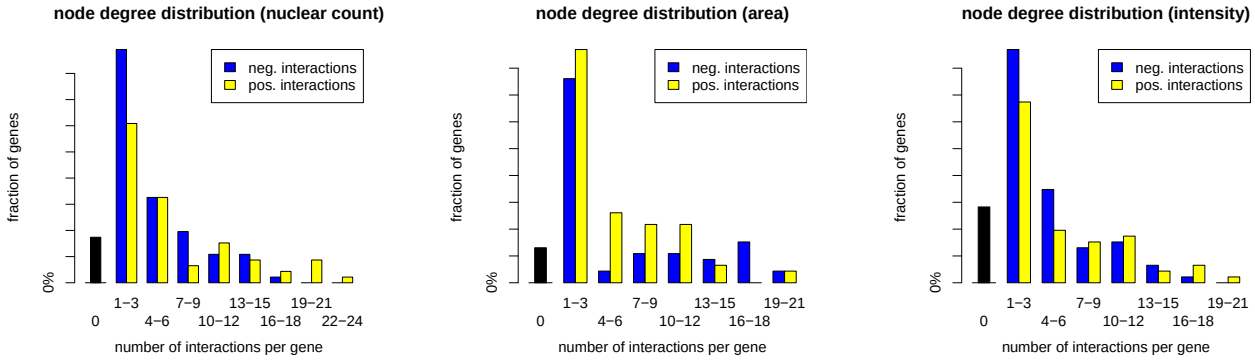


Suppl. Figure S11, Figure 4 b: Node degree distributions

For each gene the number of positively and negatively interacting genes (on a global 5% FDR) is computed and summarized in a histogram.

```
> screen = "mean"
> for (channel in c("nrCells","area","intensity")) {
+   filename <- switch(channel,
+                      nrCells="Figure-4-b-nodeDegreeNrCells.pdf",
+                      area="Figure-S11-a-nodeDegreeArea.pdf",
+                      intensity="Figure-S11-b-nodeDegreeIntensity.pdf")
+   main <- sprintf("node degree distribution (%s)",
+                   switch(channel, nrCells = "nuclear count", area = "area", intensity = "intensity
+
+   q = 0.05
+   by = 3
+   col = c("blue","yellow")
+
+   QV = getData(Dmel2PPMAPK, type="q.value", format="targetMatrix", screen=screen, channel=channel,
+     withoutgroups=c("pos","neg"))
+   PI = getData(Dmel2PPMAPK, type="pi",      format="targetMatrix", screen=screen, channel=channel,
+     withoutgroups=c("pos","neg"))
+   M = getMain(Dmel2PPMAPK,type="main",design="template",summary="target",
+     withoutgroups=c("pos","neg"), screen=screen,channel=channel)
+
+   A.pos = (QV <= q) & (PI > 0)
+   A.neg = (QV <= q) & (PI < 0)
+
+   diag(A.pos) = FALSE
+   diag(A.neg) = FALSE
+
+   a.pos = apply(A.pos,1,sum)
+   a.neg = apply(A.neg,1,sum)
+   A = A.pos | A.neg
+   a = apply(A,1,sum)
+
+   pdf(width=4.5,height=4.5,file=filename)
+   mv = max(c(a.pos,a.neg))
+   breaks = c(-0.5,seq(0.5,mv-0.5,by=3),mv+0.5)
+   h=0
+   T = matrix(c(0,1),nr=1,nc=2)
+   z=1
+   if (by > 1) {
+     names.arg = c("0")
+     for (i in 3:length(breaks)) {
+       names.arg = c(names.arg,sprintf("%d-%d",h+1,h+by))
```

```
+       z=z+1
+       for (j in 1:by) {
+         T = rbind(T, c(h+j,z))
+       }
+       h = h + by
+     }
+   }
+   h.a.pos = hist(a.pos,breaks=breaks,plot=FALSE)
+   h.a.neg = hist(a.neg,breaks=breaks,plot=FALSE)
+   df = data.frame(pos = h.a.pos$counts/92, neg = h.a.neg$counts/92, names=names.arg)
+   df[1,1] = df[1,2] = 0
+   bp=barplot(t(as.matrix(df[,c(2,1)])),beside=TRUE,col=col,main=main,
+     xlab="number of interactions per gene", ylab="fraction of genes",
+     cex.axis=1,cex.names=1,yaxt="n")
+   axis(2,at=c(0,0.05,0.1,0.15,0.2,0.25,0.3,0.35,0.4),
+        labels=c("0%","","10%","","20%","","30%","",""))
+   rect(1.5,0,2.5,sum(a==0)/92,col="black")
+   mtext(names.arg,side=1,at=apply(bp,2,mean),cex=1,line=c(1.5,0.5))
+   legend("topright",c("neg. interactions", "pos. interactions"), fill=col)
+   dev.off()
+ }
```



Suppl. Figure S12: Comparison to known interactions from DroID

The `data.frame Networks` contains the list of interactions reported in DroID (version 2010_10) resticted to the gene considered in this paper. `reportNetworks` computes p-values with the exact Fisher test on the set of interactions with FDR 5%.

```
> data("Networks", package="RNAinteractMAPK")
> reportNetworks(sgisubset(Dmel2PPMAPK, screen="mean"), Networks = Networks)
```

The p-values are written to a text file.

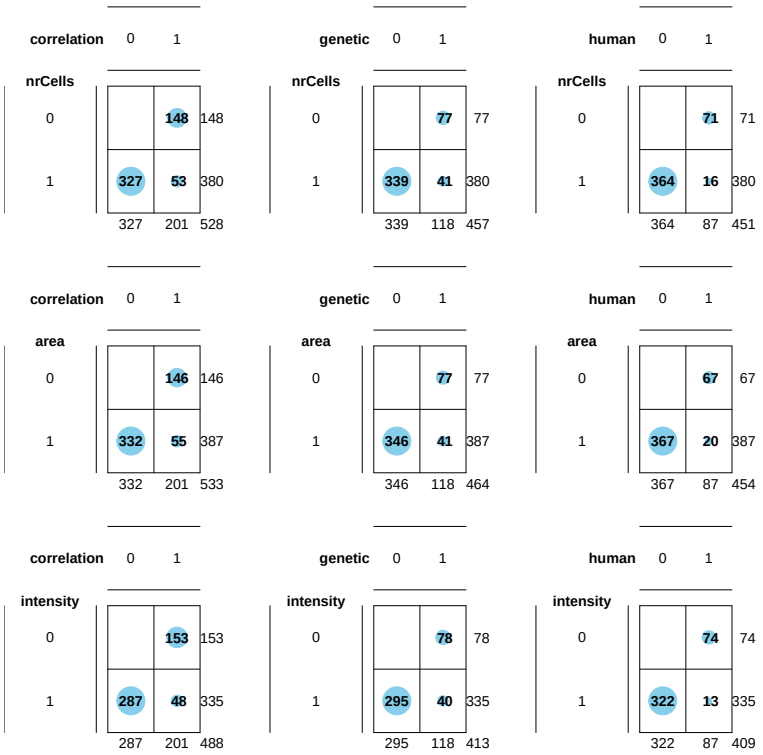
```
> print(read.table("networks/networks-pv-mean-nrCells.txt", sep="\t", header=TRUE))
```

```
                p.value odds.ratio
correlation 5.826518e-14   4.104737
genetic     1.766395e-15   5.996728
human       3.591339e-03   2.368796
```

```
> print(read.table("networks/networks-pv-mean-area.txt", sep="\t", header=TRUE))
```

```
                p.value odds.ratio
correlation 7.228662e-15   4.247135
genetic     3.400385e-15   5.864882
human       6.566888e-05   3.109088
```

```
> print(read.table("networks/networks-pv-mean-intensity.txt", sep="\t", header=TRUE))

                p.value odds.ratio
correlation 4.392413e-13   4.140770
genetic     1.211680e-16   6.712831
human       1.674841e-02   2.110339
```



Suppl. Figure S13: Known protein-protein interactions are overrepresented

The genetic interactions (FDR 5%) are compared to known physical interactions between the genes. Since the DroID database only reports 3 physical interactions among the considered genes, all direct links on the RasMAPK and JNK pathways are regarded as known physical interactions. Since, in the genetic interaction screen, there are much more interactions within the two pathways than between phosphatases, only a subset of the complete genetic interaction matrix is compared. We selected only those gene pairs within one of the two pathways and compared the genetic interactions among these with the physical interactions (direct links on pathway).
At first a complete interaction matrix `INT` extracted from the screeen with the entries $-1, 0, 1$ for negative interaction, no interaction, and positive interaction.

```
> PI <- getData(Dmel2PPMAPK, type="pi", format="targetMatrix",
+               screen="mean", channel="nrCells", withoutgroups=c("pos","neg"))
> QV <- getData(Dmel2PPMAPK, type="q.value", format="targetMatrix",
+               screen="mean", channel="nrCells", withoutgroups=c("pos","neg"))
> INT <- matrix(0, nr=93, nc=93)
> INT[QV <= 0.05] = 1
> INT[(QV <= 0.05) & (PI < 0)] = -1
> diag(INT) <- NA
```

The matrices `G1` and `G2` contain the information, which genes are part of the RasMAPK or JNK pathway.

```
> data("pathwayMembership", package="RNAinteractMAPK")
> G1 <- matrix(pathwayMembership[row.names(PI)], nr=93, nc=93)
> G2 <- t(G1)
> diag(G1) <- diag(G2) <- NA
```

The physical interactions (direct links in the pathways) are loaded and stored in a matrix with 0 for no physical interaction and 1 for interaction.

```
> data("PhysicalInteractions", package="RNAinteractMAPK")
> isPP <- matrix(0, nr=93, nc=93, dimnames = list(row.names(PI),row.names(PI)))
> for (i in 1:nrow(PhysicalInteractions)) {
+   isPP[PhysicalInteractions[i,1], PhysicalInteractions[i,2]] <- 1
+   isPP[PhysicalInteractions[i,2], PhysicalInteractions[i,1]] <- 1
+ }
> diag(isPP) <- NA
```

The matrices are stored as a data.frame where each row represents one gene pair. The gene pairs within the RasMAPK and JNK pathway are selected.

```
> df <- data.frame(int = as.vector(INT), G1 = as.vector(G1), G2 = as.vector(G2),
+                  isPP = as.vector(isPP))
> df <- df[!is.na(df$G1),]
> df <- df[((df$G1 == "RASMAPK" & df$G2 == "RASMAPK") | (df$G1 == "JNK" & df$G2 == "JNK")),]
```

The contingency table of genetic and physical interactions and Fisher's exact test comparing the two sets of interactions.

```
> t <- table(df[,c(1,4)])
> print(t)
```

```
    isPP
int    0   1
  -1  42  12
  0  374  24
  1   56  16
```

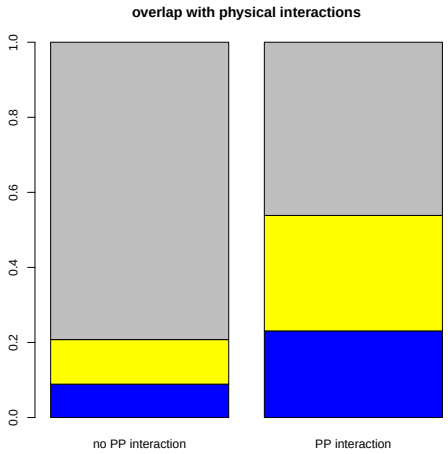
```
> t3 <- t[2:3,]
> t3[2,] <- t3[2,] + t[1,]
> t3 <- t3 / 2 # the matrix is symetric and contains each gene pair twice
> print(t3)
```

```
   isPP
int   0   1
  0 187  12
  1  49  14
```

```
> fisher.test(t3, alternative="greater")

        Fisher's Exact Test for Count Data

data:  t3
p-value = 0.0005198
alternative hypothesis: true odds ratio is greater than 1
95 percent confidence interval:
 2.025466      Inf
sample estimates:
odds ratio
  4.419974
```

A barplot of the distribution of genetic interactions within the set of gene that physically interact and within the set of genes were we do not have evidence of a physical interactions.

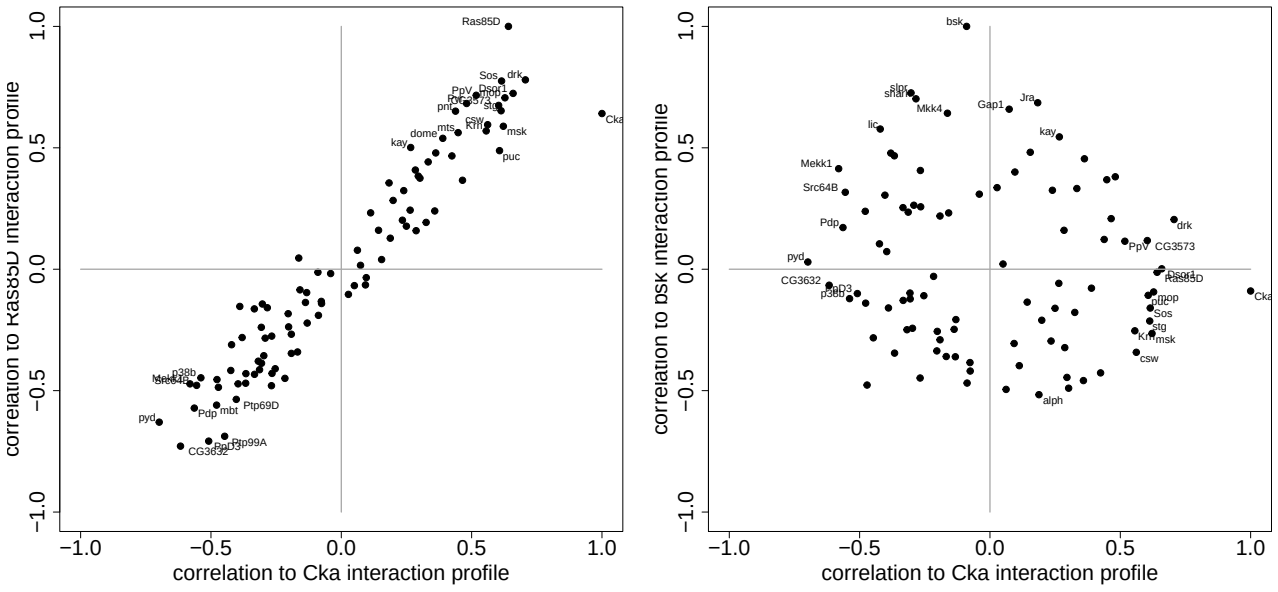
```
> t2 <- t
> t2[,1] <- t2[,1] / sum(t2[,1])
> t2[,2] <- t2[,2] / sum(t2[,2])
> t2 <- t2[c(1,3,2),]
> colnames(t2) <- c("no PP interaction", "PP interaction")
> barplot(t2, col=c("blue", "yellow", "gray"), main="overlap with physical interactions")
```



Suppl. Figure S14: Correlation profiles of Cka, Ras85 and bsk

The Pearson correlation matrix is computed from the interaction scores. The correlation profile of bsk (JNK pathway) and Ras85D (RasMAPK pathway) are plotted against the correlation profile of Cka.

```
> PI <- getData(Dmel2PPMAPK, type="pi", format="targetMatrix", screen="mean", channel="nrCells",
+               withoutgroups = c("pos", "neg"))
> C = cor(PI)

> plot(C["Cka",], C["Ras85D",], pch=20, cex=2, xlim=c(-1,1), ylim=c(-1,1),
+      cex.axis=2, cex.lab=2,
+      xlab="correlation to Cka interaction profile",
+      ylab="correlation to Ras85D interaction profile")

> plot(C["Cka",], C["bsk",], pch=20, cex=2, xlim=c(-1,1), ylim=c(-1,1),
+      cex.axis=2, cex.lab=2,
+      xlab="correlation to Cka interaction profile",
+      ylab="correlation to bsk interaction profile")
```



Suppl. Figure S15: Comparison between Acumen and CellTiterGlo pi-scores

The pi-score derived in the main screen were compared to the pi-score derived by a CellTiterGlo experiment. The CellTiterGlo validation screen were performed for a small set of gene pairs for this purpose. The CellTiterGlo data were loaded. The screen was performed in triplicates. Each replicate was located on a separate plate. Each gene pair was repeated multiple times within one column. To eliminate plate effects, a median polish was applied. The log-transformed estimated column effects were used for the further analysis.

```
> data("cellTiterGlo", package="RNAinteractMAPK")
> M1 <- t(matrix(cellTiterGlo$plate1,nr=24,nc=16))
> M2 <- t(matrix(cellTiterGlo$plate2,nr=24,nc=16))
> M3 <- t(matrix(cellTiterGlo$plate3,nr=24,nc=16))
> MP1 <- medpolish(M1)
> MP2 <- medpolish(M2)
> MP3 <- medpolish(M3)
> Anno <- cellTiterGlo[1:24,2:3]
> data <- log2(cbind(MP1$col+MP1$overall, MP2$col+MP2$overall, MP3$col+MP3$overall))
```

The respective values from the main screen were loaded.

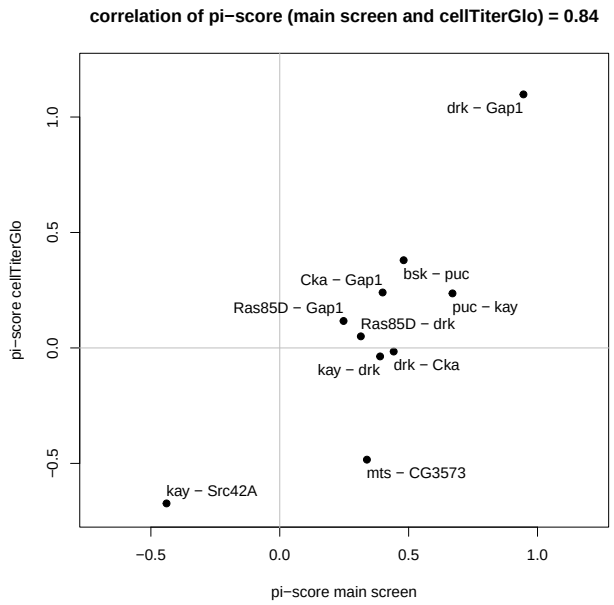
```
> PIscreen <- Anno[(Anno$dsRNA_1 != "Fluc") & (Anno$dsRNA_2 != "Fluc"),]
> PIscreen$pi.screen <- log2(getData(Dmel2PPMAPK, type="pi", format="targetMatrix",
+ channel="nrCells", screen="mean", do.inv.trafo=TRUE)[as.matrix(PIscreen)])
```

Main effects and pairwise interaction scores are estimated from the CellTiterGlo data and added to the `PIscreen` data.frame.

```
> result <- data.frame(PIscreen, pi.CTG = 0.0, main1.CTG = 0.0, main2.CTG = 0.0, p.value.CTG = 1.0)
> for (i in 1:nrow(PIscreen)) {
+   N <- which((Anno$dsRNA_1 == "Fluc") & (Anno$dsRNA_2 == "Fluc"))
+   doubleRNAi <- which(((Anno$dsRNA_1 == PIscreen[i,1]) & (Anno$dsRNA_2 == PIscreen[i,2]))
+                       | ((Anno$dsRNA_1 == PIscreen[i,2]) & (Anno$dsRNA_2 == PIscreen[i,1])))
+   singleRNAi1 <- which(((Anno$dsRNA_1 == PIscreen[i,1]) & (Anno$dsRNA_2 == "Fluc"))
+                       | ((Anno$dsRNA_1 == "Fluc") & (Anno$dsRNA_2 == PIscreen[i,1])))
+   singleRNAi2 <- which(((Anno$dsRNA_1 == "Fluc") & (Anno$dsRNA_2 == PIscreen[i,2]))
+                       | ((Anno$dsRNA_1 == PIscreen[i,2]) & (Anno$dsRNA_2 == "Fluc")))
+
+   neg <- apply(data[N,],2,mean)
+   main1 <- data[singleRNAi1,] - neg
+   main2 <- data[singleRNAi2,] - neg
+   nimodel <- neg + main1 + main2
+   piCTG <- data[doubleRNAi,] - nimodel
+   pvCTG <- t.test(piCTG)
+   result[i,"pi.CTG"] <- mean(piCTG)
+   result[i,"main1.CTG"] <- mean(main1)
+   result[i,"main2.CTG"] <- mean(main2)
+   result[i,"p.value.CTG"] <- t.test(piCTG)$p.value
+ }
```

Finally the π-score from the main screen is plotted against the π-score of the CellTitderGlo experiment.

```
> plot(result[,"pi.screen"], result[,"pi.CTG"], xlim=c(-0.7,1.2), ylim=c(-0.7, 1.2), pch=19,
+      xlab="pi-score main screen", ylab="pi-score cellTiterGlo",
+      main=sprintf("correlation of pi-score (main screen and cellTiterGlo) = %0.2f",
+        cor(result[,"pi.screen"], result[,"pi.CTG"])))
```



7 Tables

Suppl. Table 3:

The function `reportGeneListsPaper` is similar to the `reportGeneLists` function in the package RNAinteract. It writes tab separated values for each of the three features (nrCells, area, intensity) as well as one text file containing Hotellings T^2 p-value.

```
> Dmel2PPMAPKT2 <- computePValues(Dmel2PPMAPK, method="HotellingT2", verbose = FALSE)
> Dmel2PPMAPKlimma <- computePValues(Dmel2PPMAPK, method="limma",verbose = FALSE)
> MAPK.report.gene.lists.paper(Dmel2PPMAPK, Dmel2PPMAPKlimma, Dmel2PPMAPKT2)
> head(read.table("Tab3_nrCells.txt", sep="\t", header=TRUE))

  gene1 gene2 q.value.ttest p.value.ttest q.value.limma p.value.limma     main1
1   drk   mts  3.407287e-07  1.056825e-10  8.685126e-12  1.754571e-15 0.3426912
2  Rho1   mts  2.938592e-06  1.846088e-09  1.115070e-09  9.010667e-13 0.5267007
3   Pvr   mts  2.938592e-06  2.734355e-09  1.053069e-09  4.254825e-13 0.2542159
4   kay   puc  4.111501e-06  5.100995e-09  1.115070e-09  7.129572e-13 0.7695607
5   Jra   puc  6.922132e-06  1.073506e-08  8.652531e-09  1.223590e-11 0.9096439
6   drk   Pvr  9.292358e-06  1.729305e-08  6.031712e-09  6.598052e-12 0.3426912
      main2      neg        NI Measured       pi       FBgn1       FBgn2
1 0.3892580 48225.76  6713.295 15088.88 2.246533 FBgn0004638 FBgn0004177
2 0.3892580 48225.76 10374.652 17829.81 1.718926 FBgn0014020 FBgn0004177
3 0.3892580 48225.76  4923.544 13037.12 2.645926 FBgn0032006 FBgn0004177
4 0.5394751 48225.76 21756.898 34910.44 1.591260 FBgn0001297 FBgn0243512
5 0.5394751 48225.76 24240.402 37124.00 1.518502 FBgn0001291 FBgn0243512
6 0.2542159 48225.76  4191.323  9143.50 2.136130 FBgn0004638 FBgn0032006
      CG1    CG2                          Name1
1  CG6033 CG7109   downstream of receptor kinase
2  CG8416 CG7109                           Rho1
3  CG8222 CG7109 PDGF- and VEGF-receptor related
4 CG33956 CG7850                          kayak
5  CG2275 CG7850             Jun-related antigen
6  CG6033 CG8222   downstream of receptor kinase
                        Name2
1               microtubule star
2               microtubule star
```

```
3                microtubule star
4                        puckered
5                        puckered
6 PDGF- and VEGF-receptor related
```

```
> sessionInfo()
```

```
R version 4.2.1 (2022-06-23)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 20.04.5 LTS

Matrix products: default
BLAS:   /home/biocbuild/bbs-3.16-bioc/R/lib/libRblas.so
LAPACK: /home/biocbuild/bbs-3.16-bioc/R/lib/libRlapack.so

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_GB              LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] lattice_0.20-45        qvalue_2.30.0          RNAinteractMAPK_1.36.0
[4] RNAinteract_1.46.0     sparseLDA_0.1-9

loaded via a namespace (and not attached):
  [1] Category_2.64.0        bitops_1.0-7           bit64_4.0.5
  [4] RColorBrewer_1.1-3     httr_1.4.4             GenomeInfoDb_1.34.0
  [7] tools_4.2.1            elasticnet_1.3         utf8_1.2.2
 [10] R6_2.5.1               affyio_1.68.0          KernSmooth_2.23-20
 [13] cellHTS2_2.62.0        DBI_1.1.3              BiocGenerics_0.44.0
 [16] colorspace_2.0-3       gridExtra_2.3          tidyselect_1.2.0
 [19] bit_4.0.4              compiler_4.2.1         preprocessCore_1.60.0
 [22] mda_0.5-3              graph_1.76.0           cli_3.4.1
 [25] Biobase_2.58.0         caTools_1.18.2         scales_1.2.1
 [28] mvtnorm_1.1-3          genefilter_1.80.0      affy_1.76.0
 [31] RBGL_1.74.0            stringr_1.4.1          splots_1.64.0
 [34] XVector_0.38.0         jpeg_0.1-9             pkgconfig_2.0.3
 [37] maps_3.4.1             fastmap_1.1.0          limma_3.54.0
 [40] rlang_1.0.6            RSQLite_2.2.18         generics_0.1.3
 [43] hwriter_1.3.2.1        gtools_3.9.3           dplyr_1.0.10
 [46] RCurl_1.98-1.9         magrittr_2.0.3         GenomeInfoDbData_1.2.9
 [49] dotCall64_1.0-2        lars_1.3               interp_1.1-3
 [52] Matrix_1.5-1           Rcpp_1.0.9             munsell_0.5.0
 [55] S4Vectors_0.36.0       fansi_1.0.3            viridis_0.6.2
 [58] abind_1.4-5            lifecycle_1.0.3        vsn_3.66.0
 [61] stringi_1.7.8          MASS_7.3-58.1          zlibbioc_1.44.0
 [64] plyr_1.8.7             ICS_1.3-1              gplots_3.1.3
 [67] grid_4.2.1             blob_1.2.3             gdata_2.18.0.1
 [70] crayon_1.5.2           deldir_1.0-6           Biostrings_2.66.0
 [73] splines_4.2.1          annotate_1.76.0        KEGGREST_1.38.0
 [76] locfit_1.5-9.6         pillar_1.8.1           reshape2_1.4.4
 [79] geneplotter_1.76.0     stats4_4.2.1           XML_3.99-0.12
 [82] glue_1.6.2             mitools_2.4            latticeExtra_0.6-30
```

```
 [85] BiocManager_1.30.19    spam_2.9-1             png_0.1-7
 [88] vctrs_0.5.0            gtable_0.3.1           assertthat_0.2.1
 [91] cachem_1.0.6           ggplot2_3.3.6          ICSNP_1.1-1
 [94] xtable_1.8-4           survey_4.1-1           viridisLite_0.4.1
 [97] class_7.3-20           survival_3.4-0         tibble_3.1.8
[100] AnnotationDbi_1.60.0   memoise_2.0.1          IRanges_2.32.0
[103] fields_14.1            GSEABase_1.60.0
```