R Code of the 'Martins et al' manuscript

D.-L. Couturier / I. de Santiago / F.C. Martins

Last modified: 16 May 2022

This R markdown document presents the R code used to generate statistical results - like estimates, tests and p-values - presented in the manuscript *Somatic chromosomal number alterations affecting driver genes inform in-vitro and clinical drug response in high-grade serous ovarian cancer* of **Martins et al**.

```
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
##
## attached base packages:
## [1] grid      stats     graphics  utils     datasets  grDevices methods   base
##
## other attached packages:
##  [1] BiocParallel_1.28.3 vcdExtra_0.8-0      gnm_1.1-2           vcd_1.4-9
##  [5] pheatmap_1.0.12     fgsea_1.20.0        limma_3.50.3        ggcorrplot_0.1.3
##  [9] ggplot2_3.3.6       RColorBrewer_1.1-3  multcomp_1.4-19     TH.data_1.1-1
## [13] mvtnorm_1.1-3       survivalMPL_0.2-1   survival_3.3-1      psych_2.2.3
## [17] MASS_7.3-57         robustlmm_2.5-1     lme4_1.1-29         Matrix_1.4-1
## [21] robustbase_0.95-0   nlme_3.1-157        effectsize_0.6.0.1  clinfun_1.1.0
## [25] ordinal_2019.12-10  pander_0.6.5        knitr_1.39          setwidth_1.0-4
## [29] lattice_0.20-45     colorout_1.2-2
##
## loaded via a namespace (and not attached):
##  [1] insight_0.17.0      numDeriv_2016.8-1.1 tools_4.1.2         utf8_1.2.2
##  [5] R6_2.5.1            DBI_1.1.2           colorspace_2.0-3    nnet_7.3-17
##  [9] withr_2.5.0         tidyselect_1.1.2    gridExtra_2.3       mnormt_2.0.2
## [13] emmeans_1.7.3       compiler_4.1.2      performance_0.9.0   cli_3.3.0
## [17] sandwich_3.0-1      bayestestR_0.12.1   scales_1.2.0        DEoptimR_1.0-11
## [21] lmtest_0.9-40       stringr_1.4.0       digest_0.6.29       relimp_1.0-5
## [25] minqa_1.2.4         rmarkdown_2.14      ca_0.71.1           pkgconfig_2.0.3
## [29] htmltools_0.5.2     fastmap_1.1.0       highr_0.9           rlang_1.0.2
## [33] generics_0.1.2      zoo_1.8-10          dplyr_1.0.9         magrittr_2.0.3
```

```
## [37] qvcalc_1.0.2        parameters_0.17.0   Rcpp_1.0.8.3        munsell_0.5.0
## [41] fansi_1.0.3         lifecycle_1.0.1     fastGHQuad_1.0.1    ucminf_1.1-4
## [45] stringi_1.7.6       yaml_2.3.5          parallel_4.1.2      crayon_1.5.1
## [49] splines_4.1.2       tmvnsim_1.0-2       pillar_1.7.0        boot_1.3-28
## [53] estimability_1.3    codetools_0.2-18    fastmatch_1.1-3     glue_1.6.2
## [57] evaluate_0.15       data.table_1.14.2   vctrs_0.4.1         nloptr_2.0.0
## [61] gtable_0.3.0        purrr_0.3.4         datawizard_0.4.0    xfun_0.30
## [65] xtable_1.8-4        coda_0.19-4         tibble_3.1.7        ellipsis_0.3.2
```

1 Figures 1B, 1C, 1D and S1A

We present here the code allowing to reproduce the p-values related to Figures 1B, 1C, 1D and S1A

1.1 Input

These analyses are based on the input file 'input/figure_1b1c1d.csv' which, for every gene available in the TCGA ovarian cancer cohort (rows), reports the following information (columns):

- **gene.name**: gene name,
- **groups**: a four-level factor with levels
  - *driverNo_CancerGeneNo*,
  - *driverYes_CancerGeneNo*,
  - *driverNo_CancerGeneYes*,
  - *driverYes_CancerGeneYes*,
- **pearson.coef** and **spearmen.coef**: the Pearson's and Sperman's correlation coefficient estimates (function stats::cor with method argument respectively set to **pearson** and **spearman**) between
  - the normalised gene expression on the linear scale [downloaded by means of the function TCGAbiolinks::GDCquery by specyfing **"TCGA-OV"** as 'project' argument and **Gene expression** as data.category argument] and normalised by means of the function edgeR::cpm,
  - the copy number variation [downloaded by means of the function TCGAbiolinks::GDCquery by specyfing **"TCGA-OV"** as 'project' argument and *Copy Number Variation** as data.category argument] for each gene of interest,

- **met.meanofmean** and **met.meanofmedian**: the mean of the mean (function stats::mean) and median (function stats::median) of the beta methylation values [downloaded by means of the function TCGAbiolinks::GDCquery by specyfing **"TCGA-OV"** as 'project' argument and **DNA Methylation** as data.category argument] of the cpgs corresponding to each gene of interest. Genes without cpgs have missing values.
- **freq**: highest value of the homozygous deletion and amplification frequencies per gene
- **cancer.gene**: a dichotomous variable indicating if the gene is a cancer gene (Yes) or not (No) according to the "Ovarian Serous Cystadenocarcinoma (TCGA, PanCancer Atlas)" CBioportal dataset.

The following Table shows a subset of the data.

```
# import:
figure_1b1c1d = read.csv("input/figure_1b1c1ds1a.csv")
kable(figure_1b1c1d[1:10,1:7],format="simple")
```

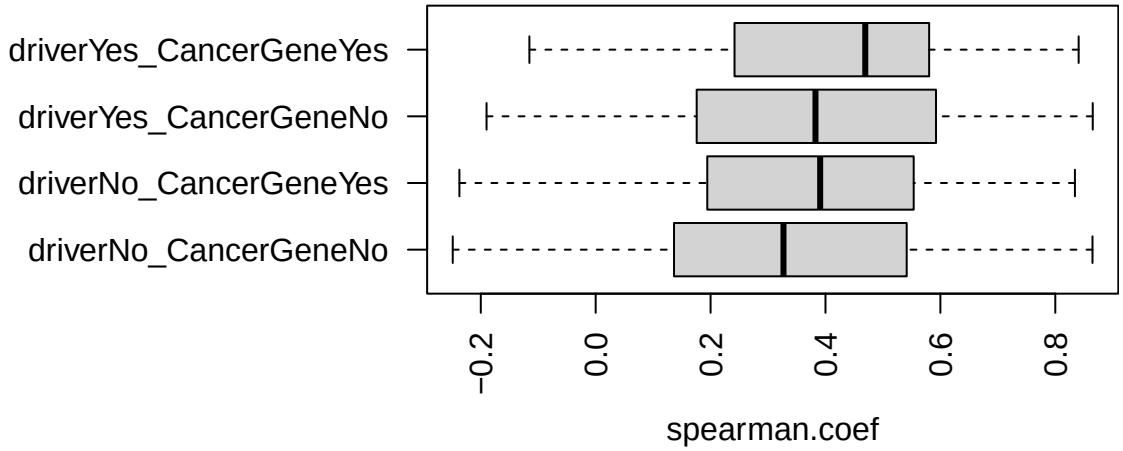
gene.name	groups	pearson.coef	spearman.coef	met.meanofmedian	met.meanofmean	freq
A1BG	driverNo_CancerGeneNo	0.3655632	0.3902986	0.9793272	0.9534876	0.017
A1CF	driverNo_CancerGeneNo	0.1291051	0.1252074	0.5180796	0.5165415	0.002
A2M	driverNo_CancerGeneNo	-0.0832420	-0.0746510	0.6147637	0.5884950	0.047
A2ML1	driverYes_CancerGeneNo	0.2396249	0.2280388	0.5840633	0.5630131	0.052
A3GALT2	driverNo_CancerGeneNo	0.3449309	0.3136347	NA	NA	0.031
A4GALT	driverNo_CancerGeneNo	0.3286870	0.3000023	0.2617974	0.2679463	0.038
A4GNT	driverNo_CancerGeneNo	0.0546836	0.0904996	NA	NA	0.047
AAAS	driverNo_CancerGeneNo	0.6194282	0.5843482	0.0591291	0.0782812	0.007
AACS	driverNo_CancerGeneNo	0.5673087	0.5572815	NA	NA	0.010
AADAC	driverYes_CancerGeneNo	0.1733435	0.1781697	NA	NA	0.080

1.2 Figure 1B

The following code allows to reproduce the p-values of Figure 1B:

- The column **wilcox.pvalue** shows the p-values of the Wilcoxon's tests comparing the Spearman correlation coefficients for the three comparisons of interest [main analysis],
- The column **welchtest.pvalue** shows the p-values of the Welch's tests comparing the Pearson correlation coefficients for the three comparisons of interest [sensitivity analysis].

```
# boxplot:
par(mfrow=c(1,1),mar=c(4,12,1,0))
boxplot(spearman.coef~groups,data=figure_1b1c1d,
        las=2,ylab="",cex.axis=1,horizontal=TRUE)
```




```
# comparisons:
n.test  = 3
id.test = data.frame(id=1:n.test,
                     group1=rep("driverYes_CancerGeneYes",n.test),
                     group2=unique(figure_1b1c1d$groups[
                                   figure_1b1c1d$groups!="driverYes_CancerGeneYes"]),
                     wilcox.pvalue=NA,
                     welchtest.pvalue=NA)
for(tw in 1:n.test){
    pos = figure_1b1c1d$groups==id.test$group1[tw]|
          figure_1b1c1d$groups==id.test$group2[tw]
    id.test$wilcox.pvalue[tw] = wilcox.test(spearman.coef~groups,
                                    data=figure_1b1c1d[pos,])$p.value
    id.test$welchtest.pvalue[tw] = t.test(pearson.coef~groups,
                                     data=figure_1b1c1d[pos,])$p.value
}
kable(id.test,format="simple")
```

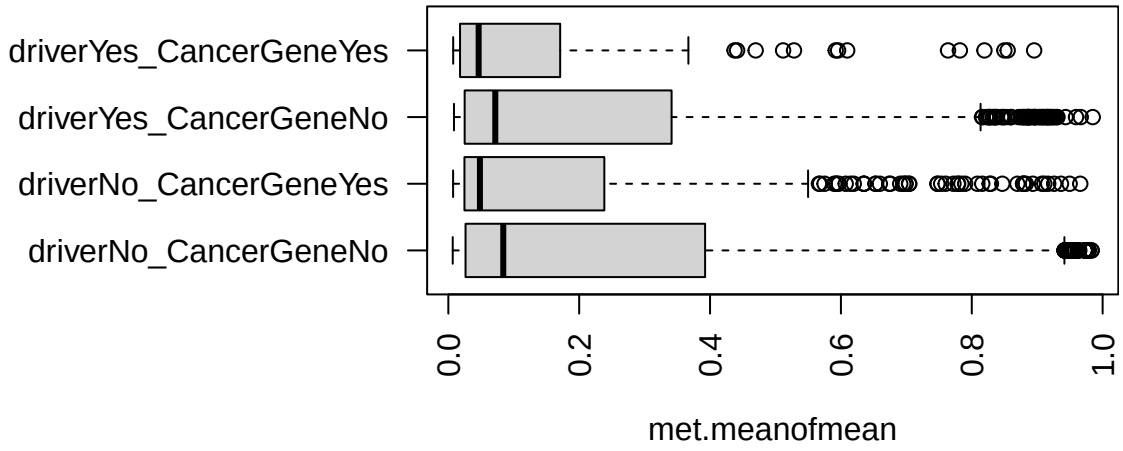
id	group1	group2	wilcox.pvalue	welchtest.pvalue
1	driverYes_CancerGeneYes	driverNo_CancerGeneNo	0.0000079	0.0000001
2	driverYes_CancerGeneYes	driverYes_CancerGeneNo	0.0403309	0.0096056
3	driverYes_CancerGeneYes	driverNo_CancerGeneYes	0.0094840	0.0006881

1.3 Figure 1C

The following code allows to reproduce the p-values of Figure 1C:

- The column **wilcox.pvalue** shows the p-values of the Wilcoxon's tests comparing the mean of the mean of the beta methylation values per CpG for the genes of interest for the three comparisons of interest [main analysis],
- The column **welchtest.pvalue** shows the p-values of the Welch's tests comparing the mean of the median of the beta methylation values per CpG for the genes of interest for the three comparisons of interest [sensitivity analysis].

```
# boxplot:
par(mfrow=c(1,1),mar=c(4,12,1,0))
boxplot(met.meanofmean~groups,data=figure_1b1c1d,
        las=2,ylab="",cex.axis=1,horizontal=TRUE)
```




```
# comparisons:
n.test  = 3
id.test = data.frame(id=1:n.test,
                     group1=rep("driverYes_CancerGeneYes",n.test),
                     group2=unique(figure_1b1c1d$groups[
                                   figure_1b1c1d$groups!="driverYes_CancerGeneYes"]),
                     wilcox.pvalue=NA,
                     welchtest.pvalue=NA)
for(tw in 1:n.test){
    pos = figure_1b1c1d$groups==id.test$group1[tw]|
          figure_1b1c1d$groups==id.test$group2[tw]
    id.test$wilcox.pvalue[tw] = wilcox.test(met.meanofmean~groups,
                                            data=figure_1b1c1d[pos,])$p.value
    id.test$welchtest.pvalue[tw] = t.test(met.meanofmedian~groups,
                                          data=figure_1b1c1d[pos,])$p.value
}
kable(id.test,format="simple")
```

id	group1	group2	wilcox.pvalue	welchtest.pvalue
1	driverYes_CancerGeneYes	driverNo_CancerGeneNo	0.0002546	0.0000547
2	driverYes_CancerGeneYes	driverYes_CancerGeneNo	0.0052981	0.0024735
3	driverYes_CancerGeneYes	driverNo_CancerGeneYes	0.1399423	0.4021579

1.4 Figure 1D

The following code allows to reproduce the p-values of Figure 1D:

- The column **pearson.pvalue** shows the p-values of the Pearsons's correlation tests between the normalised methylation and gene expression data for the four groups of interest [main analysis],
- The column **spearman.pvalue** shows the p-values of the Spearman's correlation tests between the normalised methylation and gene expression data for the four groups of interest [sensitivity analysis].

```
# comparisons:
n.test  = length(unique(figure_1b1c1d$groups))
id.test = data.frame(id=1:n.test,
                     group=unique(figure_1b1c1d$groups),
                     pearson.pvalue=NA,
                     spearman.pvalue=NA,
                     n = NA)
# normalisation
x0   = figure_1b1c1d$met.meanofmean
y0   = figure_1b1c1d$spearman.coef
grp0 = figure_1b1c1d$groups
x1   = x0[!is.na(x0)]
y1   = y0[!is.na(x0)]
n    = length(x1)
x2   = qnorm(ppoints(1:n)[rank(x1)])
y2   = qnorm(ppoints(1:n)[rank(y1)])
grp2 = id.cna_tcga$groups[!is.na(x0)]
# tests
for(tw in 1:n.test){
    x2.g = x2[grp2==id.test$group[tw]]
    y2.g = y2[grp2==id.test$group[tw]]
    id.test$n[tw] = length(x2.g)
    id.test$pearson.pvalue[tw] = cor.test(x2.g,y2.g,method="pearson")$p.value
    id.test$spearman.pvalue[tw] = cor.test(x2.g,y2.g,method="spearman")$p.value

}
kable(id.test,format="simple")
```

id	group	pearson.pvalue	spearman.pvalue	n
1	driverNo_CancerGeneNo	0.0000000	0.0000000	8720
2	driverYes_CancerGeneNo	0.0000000	0.0000000	1330
3	driverNo_CancerGeneYes	0.0000000	0.0000000	590
4	driverYes_CancerGeneYes	0.0042216	0.0008408	115

1.5 Figure S1A

The following code allows to reproduce the p-values of Figure S1A

We first define relevant quantities for 100 threshold values for prevalence of altered versus non-altered cancer genes over the range of interest (between 0 and 0.1) like

- the Number of cancer genes above the prevalence threshold,
- the difference in median Spearman correlation levels between altered versus non-altered cancer genes,
- the p-value of the Wilcoxon's test

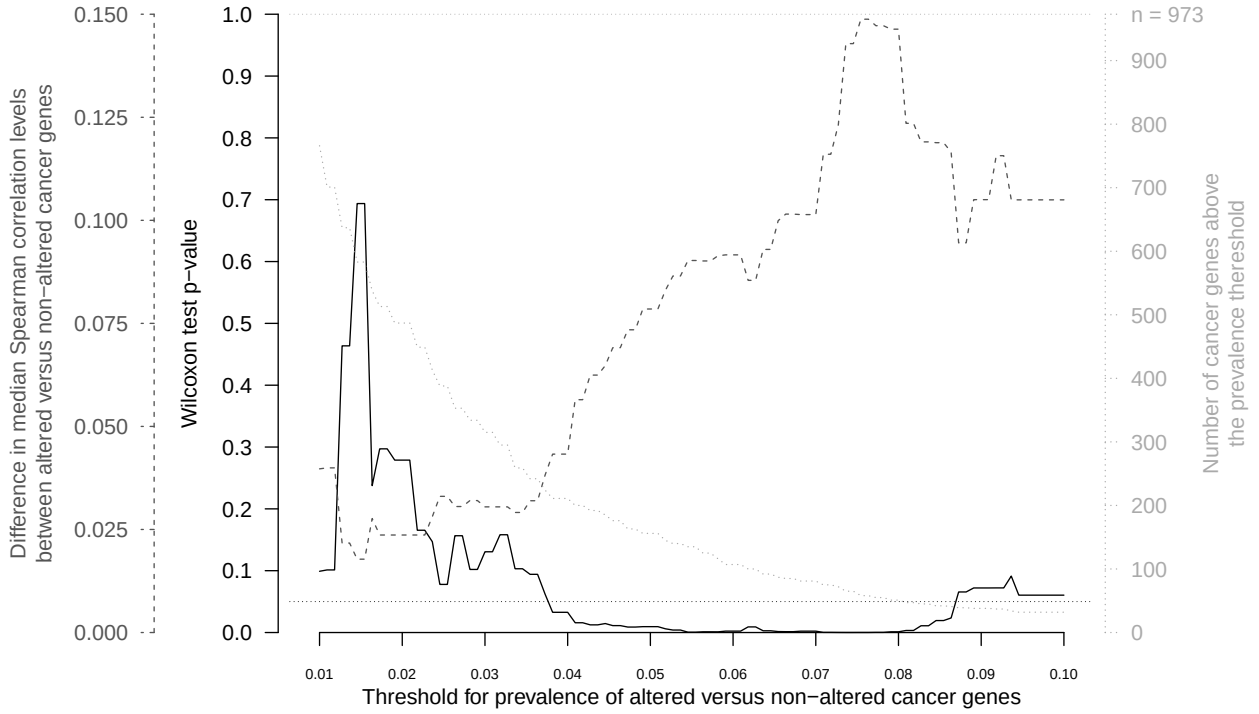
and then plot them.

```
##
## information
##
n.prob  = 100
id.prob = data.frame(pos=1:n.prob,
                     id=as.character(seq(0.01,0.1,length=n.prob)),
                     value=seq(0.01,0.1,length=n.prob))
mx.inf.p4 = matrix(NA,ncol=4,nrow=n.prob,
                   dimnames=list(id.prob$id,c("nN","nY","diff","p.value")))
for(pw in 1:n.prob){
    figure_1b1c1d$freqw = unlist(sapply(split(figure_1b1c1d$freq,figure_1b1c1d$gene.name),
        function(x,prob){
        if(any(x>=prob)){rep(TRUE,length(x))}else{rep(FALSE,length(x))}},
        prob=id.prob$value[pw]))
    # keep only 1 info per gene
    figure_1b1c1d$groupw = paste0("driver",c("No","Yes")[figure_1b1c1d$freqw+1],
                                  "_CancerGene",figure_1b1c1d$cancer.gene)
    figure_1b1c1d$groupw = factor(figure_1b1c1d$groupw,
                                  levels=c("driverNo_CancerGeneNo","driverYes_CancerGeneNo",
                                           "driverNo_CancerGeneYes","driverYes_CancerGeneYes"))
    dataw = figure_1b1c1d[figure_1b1c1d$groupw=="driverYes_CancerGeneYes"|
                          figure_1b1c1d$groupw=="driverNo_CancerGeneYes",]
    dataw$groupw = droplevels(dataw$groupw)
    # sample size
    mx.inf.p4[pw,c("nN","nY")] = table( dataw$groupw)
    # diff in mean
    median_groupw = tapply(dataw$spearman.coef,dataw$groupw,median)
    mx.inf.p4[pw,c("diff")] = median_groupw[2]-median_groupw[1]
    # pval
    mx.inf.p4[pw,c("p.value")] = wilcox.test(spearman.coef~groupw,data=dataw)$p.value
    #
    .cat(pw,n.prob)
}
```

```
## ..................................................50 Mon May 16 09:59:14 2022
## ..................................................100 Mon May 16 09:59:17 2022
```

```
##
## plot
##
par(mar=c(3.25,12,1,7))
plot(1,1,pch="",xlim=range(id.prob$value),ylim=c(0,1),axes=FALSE,
     xlab="",ylab="")
axis(1,seq(0.01,0.1,.01),pos=0,cex.axis=.7)
#
lines(id.prob$value,mx.inf.p4[,"p.value"],col=gray(0))
lines(id.prob$value,mx.inf.p4[,"diff"]/.15,col=gray(1/3),lty=2)
lines(id.prob$value,mx.inf.p4[,"nY"]/973,col=gray(2/3),lty=3)
axis(1,at=mean(range(id.prob$value)),
"Threshold for prevalence of altered versus non-altered cancer genes",
tick=FALSE)
#
axis(2,seq(0,1,.1),pos=0.005,col.axis=gray(0),col=gray(0),las=2)
```

```
abline(h=0.05,col=gray(0),lty=3,lwd=.75)
axis(2,at=0.5,"Wilcoxon test p-value",
     col.axis=gray(0),col=gray(0),tick=FALSE,padj=-4)
#
axis(2,seq(0,.15,.025)/.15,format(seq(0,.15,.025)),pos=-0.01,
     col.axis=gray(1/3),col=gray(1/3),las=2,lty=2)
axis(2,at=0.5,paste0("Difference in median Spearman correlation levels",
     "\nbetween altered versus non-altered cancer genes"),
     col.axis=gray(1/3),col=gray(1/3),tick=FALSE,padj=-5)
#
axis(4,c(seq(0,973,100),973)/973,c(seq(0,973,100),"n = 973"),
     pos=.105,col.axis=gray(2/3),col=gray(2/3),las=2,lty=3)
abline(h=max(apply(mx.inf.p4[,c("nN","nY")],1,sum))/973,col=gray(2/3),lty=3,lwd=.75)
axis(4,at=0.5,"Number of cancer genes above\nthe prevalence thereshold",
     col.axis=gray(2/3),col=gray(2/3),tick=FALSE,padj=2.5)
```



2 Figures 2B, 2D, 2H, S4C and S4F

We present here the code used to analyse the relationship between drug response (AUC defined as described in Section 13) and the adjusted copy number (ACN) and relative copy number (RCN) of different drivers:

- 2B: response to Paclitaxel as a function of MYC ACN/RCN,
- 2D: response to AZ0156 as a function of CCNE1 ACN/RCN,
- 2H: response to AZ2014 as a function of MYC ACN/RCN,
- S4C: response to Doxorubicin as a function of KRAS ACN/RCN,
- S4F: response to AZ2014 as a function of TERT ACN/RCN.

2.1 Input

This analysis is based on the input file 'input/figure_2b2d2hs4cs4f.csv' which, for each sample (rows), reports the following information (columns):

- **patient**: the patient ID (to have independent data, the second sample of patient 409 was discarded),
- **MYC/KRAS/CCNE1**: drivers relative copy numbers (on the log2 scale),
- **Paclitaxel/Doxorubicin/AZD0156/AZD2014**: lab drug response (measured by means of AUC),

The following Table shows a subset of the data.

```
# import:
figure_2b2d2hs4cs4f = read.csv("input/figure_2b2d2hs4cs4f.csv")
kable(figure_2b2d2hs4cs4f[1:5,1:8],format="simple")
```

patient	MYC	CCNE1	KRAS	TERT	Paclitaxel	AZD2014	Oxaliplatin
294	0.907147	0.1426132	0.4397947	-0.0330413	0.5015600	0.5757075	0.6031470
333	1.765129	0.1985729	-0.1339190	0.2715367	0.6448216	0.6004840	0.9794704
364	1.321792	-0.1334255	0.6892377	0.4679765	0.8472468	0.4932141	0.7459932
409	0.964717	1.0813844	0.1963729	1.1219111	0.4600522	0.3835962	0.7183650
413	1.004909	0.1493322	-0.0796468	-0.0699756	0.2889143	0.5045604	0.8973087

2.2 Analyses

```
# define drug/driver combinations of interest per figure:
n.figure  = 5
id.figure = data.frame(pos    = 1:n.figure,
                       id     = c("2B","2D","2H","S4C","S4F"),
                       drug   = c("Paclitaxel","AZD0156","AZD2014","Doxorubicin","AZD2014"),
                       driver = c("MYC","CCNE1","MYC","KRAS","TERT"))
# useful
CCNE1.keep = figure_2b2d2hs4cs4f[,"CCNE1"]<log2(6)-1
#
par(mfrow=c(5,2),mar=c(3.5,5,1,0),omi=c(0,0,0,0))
ylimw  = c(0,max(figure_2b2d2hs4cs4f[,id.figure$drug])*1.05)
xlimw  = range(figure_2b2d2hs4cs4f[,id.figure$driver])
effectsize_figure = as.list(rep(NA,n.figure))
names(effectsize_figure) = id.figure$driver
#
for(fw in 1:n.figure){# fw=5
    # data
```

```
auc  = figure_2b2d2hs4cs4f[,id.figure$drug[fw]]
rcn  = figure_2b2d2hs4cs4f[,id.figure$driver[fw]]
acn  = unlist(round(2*2^rcn))
acn3 = ordered(c(NA,2,3,rep(4,100))[acn],levels=c(2:4),labels=c("2","3","4+"))
# scatterplot
plot(1,1,pch="",ylim=ylimw,xlim=range(rcn),
     axes=FALSE,xlab="",ylab="",
     main=paste0("Figure ",id.figure$id[fw]," (left)"))
abline(h=seq(0,1.2,.2),col="light gray",lty=3)
points(rcn,auc,
       pch=if(fw!=3){19}else{c(15,19)[.an(CCNE1.keep)+1]},
       col=paste0(gray(0.5),99),
       cex=2)
axis(1,seq(-1,5,.5),pos=0)
axis(1,mean(range(rcn)),paste0(id.figure$driver[fw]," RCN (log2 scale)"),
     tick=FALSE,cex.axis=1.25,padj=.75)
axis(2,seq(0,1.2,.2),las=2)
axis(2,1.2/2,paste0(id.figure$drug[fw]," drug response"),
     tick=FALSE,cex.axis=1.25,padj=-3.5)
fit = lm(auc~rcn)
abline(fit,col="black",lwd=2)
effectsize_figure[[fw]] = effectsize(fit)
# additional fit removing samples with high CCNE1 CN for AZD2014/MYC
if(id.figure$drug[fw]=="AZD2014"&id.figure$driver[fw]=="MYC"){
   fit2 = lm(auc[CCNE1.keep]~rcn[CCNE1.keep])
   abline(fit2,col="black",lwd=1,lty=2)
   effectsize_figure[[fw]] = effectsize(fit)
   }
# legends (including p-values)
if(!(id.figure$drug[fw]=="AZD2014"&id.figure$driver[fw]=="MYC")){
    legend("top",legend=.p("p-val. trend = ",.pval(coef(summary(fit))[2,4]/2)),
           cex=1,text.col="#00B6ED",box.lwd=NA)
}else{
    legend("top",legend=c(.p("p-val. trend = ",.pval(coef(summary(fit))[2,4]/2)),
                          .p("p-val. trend = ",.pval(coef(summary(fit2))[2,4]/2),
                             "\n(excl. CCNE1 ACN > 6 )")),
           cex=1,text.col="#00B6ED",box.lwd=NA,
           col="#00B6ED",lty=c(1,2))
    legend(0,0.4,title="CCNE1 ACN:",
           legend=c("<=6",">6"),
           pch=c(19,15),text.col="black",col="black",
           box.lwd=NA,bg=paste0(gray(0.5),40))
}
# boxplot
plot(1,1,pch="",ylim=ylimw,xlim=c(.75,2.25),axes=FALSE,
     xlab="",#paste0(id.figure$driver[fw]," RCN (log2 scale)"),
     ylab=paste0(id.figure$drug[fw]," drug response"),
     main=paste0("Figure ",id.figure$id[fw]," (right)"))
abline(h=seq(0,1.2,.2),col="light gray",lty=3)
boxplot(auc~acn3,
     xlab="",ylab="",main="",
     boxwex =.25,at = c(1.2,1.5,1.8),
     col=paste0(gray(0.5),99),ylim=ylimw,axes=FALSE,add=TRUE)
```

```
    axis(2,seq(0,1.2,.2),las=2)
    axis(1,c(1.2,1.5,1.8),c("2","3","4+"),pos=0)
    axis(1,1.5,.p(id.figure$driver[fw]," ACN"),
         tick=FALSE,cex.axis=1.25,padj=.75)
    # one-sided trend test
    testw = try(jonckheere.test(auc,acn3))
    if(class(testw)!="try-error"){
        if(!is.na(testw$p.value/2)){
        legend("top",legend=.p("p-val. trend = ",.pval(testw$p.value/2)),
                cex=1,text.col="#00B6ED",box.lwd=NA)
        }else{
        legend("top",legend=.p("p-val. = ",
                .pval(t.test(auc[.an(acn3)<3]~acn3[.an(acn3)<3])$p.value/2)),
                cex=1,text.col="#00B6ED",box.lwd=NA)
        }
        }
    # one-sided t.test
    if(id.figure$id[fw]=="2H"){
        segments(1.15,1.1,
                 1.85,1.1,col="#00B6ED",lwd=2)
        segments(1.15,1.1,
                 1.15,1.09,col="#00B6ED",lwd=2)
        segments(1.85,1.1,
                 1.85,1.09,col="#00B6ED",lwd=2)

        text(1.5,1.1,pos=3,col="#00B6ED",
             .p("p-val. = ",.pval(t.test(auc[acn3!="3"]~acn3[acn3!="3"],
                var.equal=TRUE)$p.value/2)),cex=1)
        }
}
```

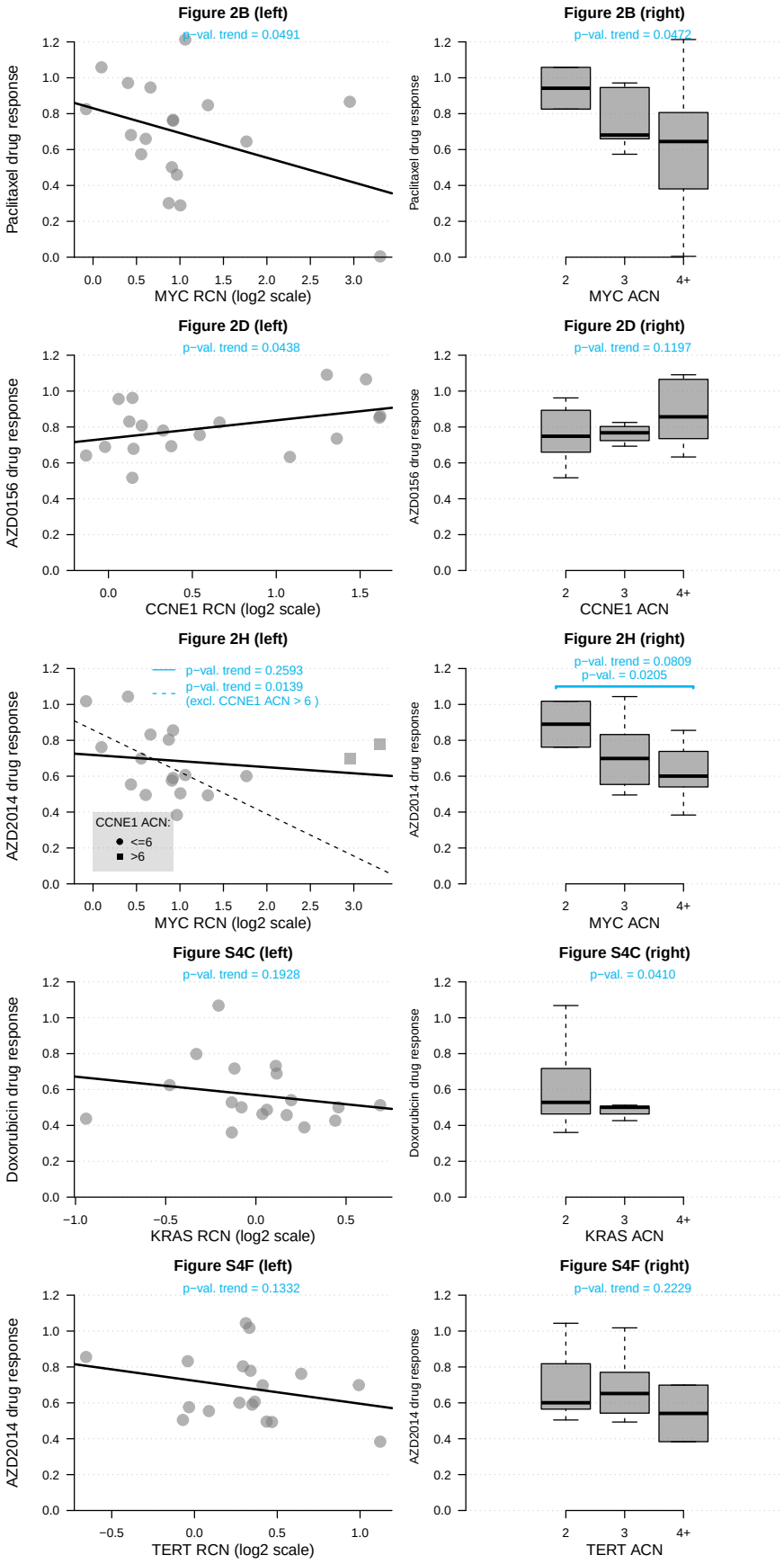


Figure 2B (left)
p−val. trend = 0.0491
Paclitaxel drug response
MYC RCN (log2 scale)
Figure 2B (right)
p−val. trend = 0.0472
Paclitaxel drug response
MYC ACN
Figure 2D (left)
p−val. trend = 0.0438
AZD0156 drug response
CCNE1 RCN (log2 scale)
Figure 2D (right)
p−val. trend = 0.1197
AZD0156 drug response
CCNE1 ACN
Figure 2H (left)
p−val. trend = 0.2593
p−val. trend = 0.0139
(excl. CCNE1 ACN > 6)
CCNE1 ACN:
<=6
>6
AZD2014 drug response
MYC RCN (log2 scale)
Figure 2H (right)
p−val. trend = 0.0809
p−val. = 0.0205
AZD2014 drug response
MYC ACN
Figure S4C (left)
p−val. trend = 0.1928
Doxorubicin drug response
KRAS RCN (log2 scale)
Figure S4C (right)
p−val. = 0.0410
Doxorubicin drug response
KRAS ACN
Figure S4F (left)
p−val. trend = 0.1332
AZD2014 drug response
TERT RCN (log2 scale)
Figure S4F (right)
p−val. trend = 0.2229
AZD2014 drug response
TERT ACN

We finally report here the effect sizes corresponding to the different fits of interest:

```
print(effectsize_figure)
```

```
## $MYC
## # Standardization method: refit
##
## Parameter   | Coefficient (std.) |        95% CI
## ------------------------------------------------
## (Intercept) |           2.08e-17 | [-0.47, 0.47]
## rcn         |              -0.40 | [-0.89, 0.08]
##
## $CCNE1
## # Standardization method: refit
##
## Parameter   | Coefficient (std.) |        95% CI
## ------------------------------------------------
## (Intercept) |           5.34e-16 | [-0.47, 0.47]
## rcn         |               0.41 | [-0.07, 0.90]
##
## $MYC
## # Standardization method: refit
##
## Parameter   | Coefficient (std.) |        95% CI
## ------------------------------------------------
## (Intercept) |          -9.13e-17 | [-0.51, 0.51]
## rcn         |              -0.16 | [-0.69, 0.36]
##
## $KRAS
## # Standardization method: refit
##
## Parameter   | Coefficient (std.) |        95% CI
## ------------------------------------------------
## (Intercept) |           2.09e-16 | [-0.50, 0.50]
## rcn         |              -0.22 | [-0.73, 0.30]
##
## $TERT
## # Standardization method: refit
##
## Parameter   | Coefficient (std.) |        95% CI
## ------------------------------------------------
## (Intercept) |          -7.13e-17 | [-0.49, 0.49]
## rcn         |              -0.28 | [-0.79, 0.23]
```

3 Figure 2E

We present here the code allowing to reproduce the cluster analysis presented in Figure 2E.

3.1 Input

This analysis is based on the input file 'input/figure_2e.csv' which, for each sample (rows), reports the standardised [(x-mean(x))/sqrt(var(x))] drug response (area under the curve estimates; refer to Section 13 for detail) for each drug of interest (column).

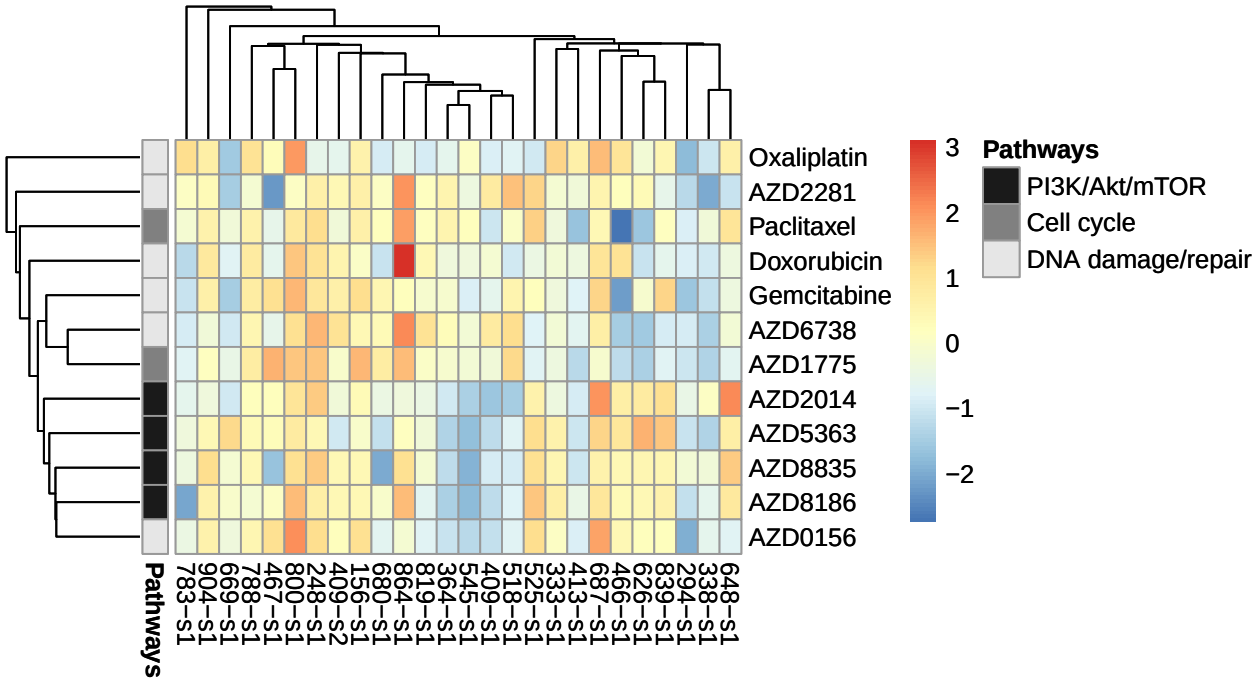
The following Table shows a subset of the data.

```
# import:
figure_2e = read.csv("input/figure_2e.csv",row.names=1)
kable(figure_2e[1:2,1:7],format="simple")
```

	AZD8186	AZD2014	AZD5363	AZD2281	AZD0156	AZD6738	AZD1775
156-s1	0.4140057	0.3379225	-0.0381315	0.6076683	1.083319	0.4169115	1.619257
248-s1	0.6527483	1.4120392	0.4067076	0.6370872	1.160650	1.6150815	1.434186

3.2 Cluster analysis

```
# pathway
pathway = data.frame(Pathways=rep("DNA damage/repair",ncol(figure_2e)),
                     row.names=colnames(figure_2e))
pathway$Pathways[!is.na(match(colnames(figure_2e),
        c("AZD2014","AZD5363","AZD8186","AZD8835")))] =
        'PI3K/Akt/mTOR'
pathway$Pathways[!is.na(match(colnames(figure_2e),
        c("Paclitaxel","AZD1775")))] =
        'Cell cycle'
col2 = list(Pathways = c('PI3K/Akt/mTOR' = gray(.1),
                         'Cell cycle' = gray(.5),
                         'DNA damage/repair' = gray(.9)))
# clustering
clust2 = pheatmap::pheatmap(t(figure_2e),
             annotation_row=pathway,
             clustering_distance_rows = "correlation",
             clustering_distance_cols = "correlation",
             clustering_method = "single",
             annotation_names_row = TRUE,
             labels_row = colnames(figure_2e),
             annotation_colors = col2,annotation_legend=TRUE,
             show_rownames=TRUE, scale="none")
clust2
```


Oxaliplatin
AZD2281
Paclitaxel
Doxorubicin
Gemcitabine
AZD6738
AZD1775
AZD2014
AZD5363
AZD8835
AZD8186
AZD0156
Pathways
783–s1
904–s1
669–s1
788–s1
467–s1
800–s1
248–s1
409–s2
156–s1
680–s1
864–s1
819–s1
364–s1
545–s1
409–s1
518–s1
525–s1
333–s1
413–s1
687–s1
466–s1
626–s1
839–s1
294–s1
338–s1
648–s1
3
2
1
0
−1
−2
Pathways
PI3K/Akt/mTOR
Cell cycle
DNA damage/repair

4 Figures 2F

We present here the code allowing to reproduce the cluster analysis and correlation test corresponding to Figure 2F.

4.1 Input

This analysis is based on the input file 'input/figure_2f.csv' which, for each sample (rows, to have independent data, the second sample of patient 409 was discarded), reports the drug response (area under the curve estimates, refer to Section 13) for each drug (column).

The following Table shows a subset of the data.

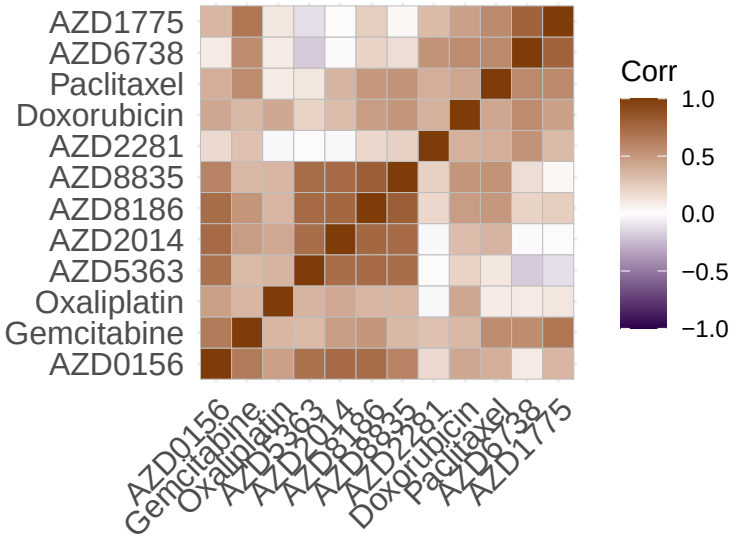
```
# import:
figure_2f = read.csv("input/figure_2f.csv")
kable(figure_2f[1:3,1:7],format="simple")
```

AZD8186	AZD2014	AZD5363	AZD2281	AZD0156	AZD6738	AZD1775
0.9374681	0.7228402	0.8629120	0.9985092	0.9513489	0.8206759	0.9999955
0.9649787	0.9132249	0.9202300	1.0028731	0.9624234	0.9969480	0.9634500
0.7572422	0.5757075	0.7320789	0.7238694	0.5168565	0.6269203	0.4756609

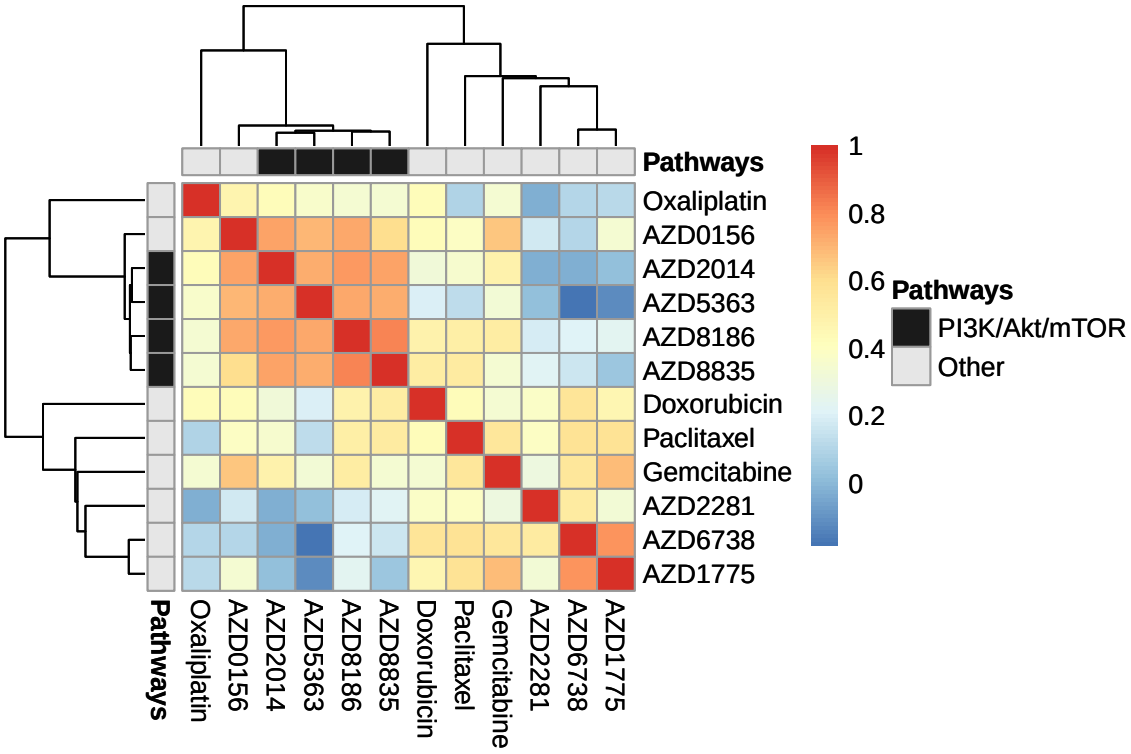
4.2 Analysis

We start with two cluster analyses of the (spearman) correlation matrix of the drug responses. Both clustering show that the drugs related to the PI3K-pathway ("AZD2014","AZD5363","AZD8186","AZD8835") are clustered together:

```
# first clustering
colw  = RColorBrewer:::brewer.pal(11,"PuOr")[11:1]
clust1 = ggcorrplot::ggcorrplot(cor(figure_2f,method="spearman"),
            hc.order = TRUE,hc.method="complete",
            colors =c(colw[1], "white",colw[11]))
clust1
```



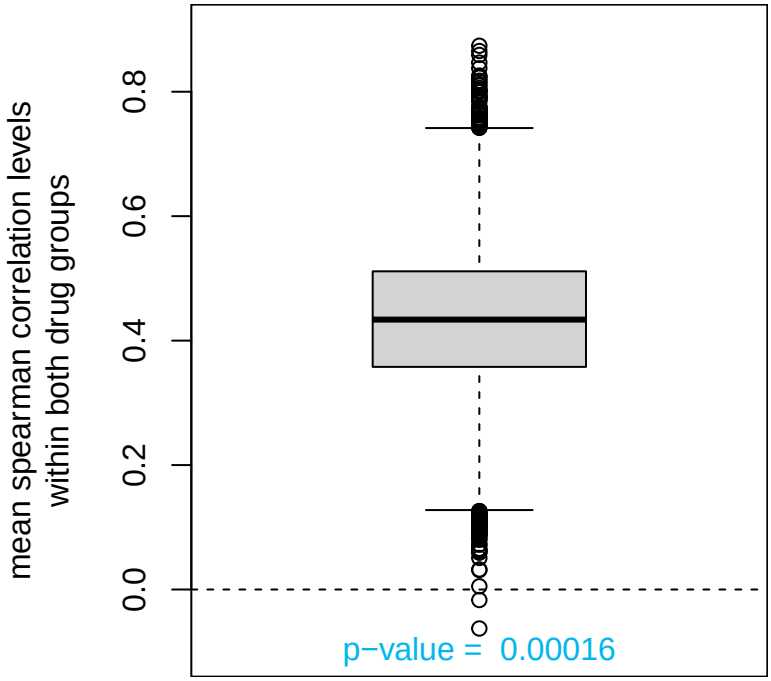
```
# second clustering
pathway = data.frame(Pathways=rep("Other",ncol(figure_2f)),
                     row.names=colnames(figure_2f))
pathway$Pathways[!is.na(match(colnames(figure_2f),
        c("AZD2014","AZD5363","AZD8186","AZD8835")))] =
        'PI3K/Akt/mTOR'
col2 = list(Pathways = c('PI3K/Akt/mTOR' = gray(.1),
                         'Other' = gray(.9)))
clust2 = pheatmap::pheatmap(cor(figure_2f,method="spearman"),
    annotation_row=pathway,
    annotation_col=pathway,
    clustering_distance_rows = "correlation",
    clustering_distance_cols = "correlation",
    clustering_method = "single",
    annotation_names_row = TRUE,
    labels_row = colnames(figure_2f),
    annotation_colors = col2,annotation_legend=TRUE,
    show_colnames=TRUE,
    show_rownames=TRUE, scale="none")
clust2
```



We continue with the test of equality of means of the correlation between the AUC of drugs belonging (group 1) or not (group 2) to the 'PI3K/Akt/mTOR' pathway by means of a non-parametric bootstrap test.

```
# inference
id.drug = c('AZD8186','AZD5363','AZD2014','AZD8835',
            'AZD0156','Oxaliplatin','Paclitaxel','AZD1775',
            'Gemcitabine','Doxorubicin','AZD2281','AZD6738')
mx.auc.dd = figure_2f[,id.drug]
mx.cor_val.dd = cor(mx.auc.dd,method="spearman")
```

```r
#
set.seed(69526)# seed set to the last author phone extension
n.R = 2.5e+4    # number of boostrap samples
pos.k_group = list(g1=c(2:4,15:16,28),g2=c(5:12,17:24,29:36,41:48))
mx.pos.Rn = t(sapply(1:n.R,function(x)sample(1:nrow(figure_2f),nrow(figure_2f),replace=TRUE)))
diff.R = apply(mx.pos.Rn,1,function(x){# x=mx.pos.Rn[1,]
    mx.cor.dd = cor(mx.auc.dd[x,],method="spearman")
    mean(mx.cor.dd[pos.k_group[[1]]])-mean(mx.cor.dd[pos.k_group[[2]]])
    })
par(mfrow=c(1,1),mar=c(0,5,0,0))
boxplot(diff.R,ylab=paste("Difference between the\nmean spearman",
        "correlation levels\nwithin both drug groups"),
        ylim=c(-.1,.9))
abline(h=0,col=1,lty=2)
text(1,-.1,paste("p-value = ",mean(diff.R<0)*2),col="#00B6ED")
```



5 Figure 3A

We present here the code allowing to reproduce analysis presented in Figure 3A.

5.1 Input

This analysis is based on the input file 'input/figure_3a.csv' which, for each gene of the TCGA OV cohort (rows), reports

- the lab response to different drugs, measured by the area under the curve (AUC, refer to Section 13 for detail),
- the relative copy number (RCN) for different genes of interest.

The following Table shows a subset of the data (10 genes and 2 TCGA OV samples).

```
# import:
figure_3a = as.matrix(read.csv("input/figure_3a.csv",row.names=1))
kable(figure_3a[1:10,1:2],format="simple")
```

	TCGA.5X.AA5U.01A.11R.A406.31	TCGA.25.2400.01A.01R.1569.13
TSPAN6|7105	5.819914	7.226783
TNMD|64102	-5.099694	-2.283508
DPM1|8813	5.232342	5.492996
SCYL3|57147	4.845457	3.754218
C1orf112|55732	4.592747	3.590604
FGR|2268	1.597968	2.190109
CFH|3075	3.835176	3.486401
FUCA2|2519	4.832520	6.614863
GCLC|2729	5.097277	6.494283
NFYA|4800	6.590041	5.072130

5.2 Test

We first estimate the Pearson correlation between the expression of each gene and the expression of MYC.

```
# correlation estimates
idx       = which(rownames(figure_3a) == "MYC|4609")
genes     = rownames(figure_3a)
genes     = genes[-idx]
n.genes   = length(genes)
genesCorr = sapply(genes, function(x) cor.test(figure_3a[x ,],
                  figure_3a[ idx,], use="complete")[3:4])
# re-organise and add MTOR pathway indicator
bioplanet_mtor <- c("EIF4B|1975","EIF4E|1977","EIF4EBP1|1978","EIF4G1|1981",
                    "MTOR|2475","RPS6|6194","RPS6KB1|6198","RHEB|6009",
                    "EEF2K|29904","RPTOR|57521","MLST8|64223")
genesCorr = data.frame(pval = as.numeric(genesCorr[1, ]),
                       cor  = as.numeric(genesCorr[ 2, ]),
                       mtor = (names(genesCorr[ 2, ]) %in% bioplanet_mtor),
                       row.names=genes)
```

and finally compare the correlation levels of genes belonging or not to the MTOR pathway

```r
# tests (including sensitivity analyses)
wilcox.test(cor~mtor,data=genesCorr)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  cor by mtor
## W = 59016, p-value = 0.02451
## alternative hypothesis: true location shift is not equal to 0
```

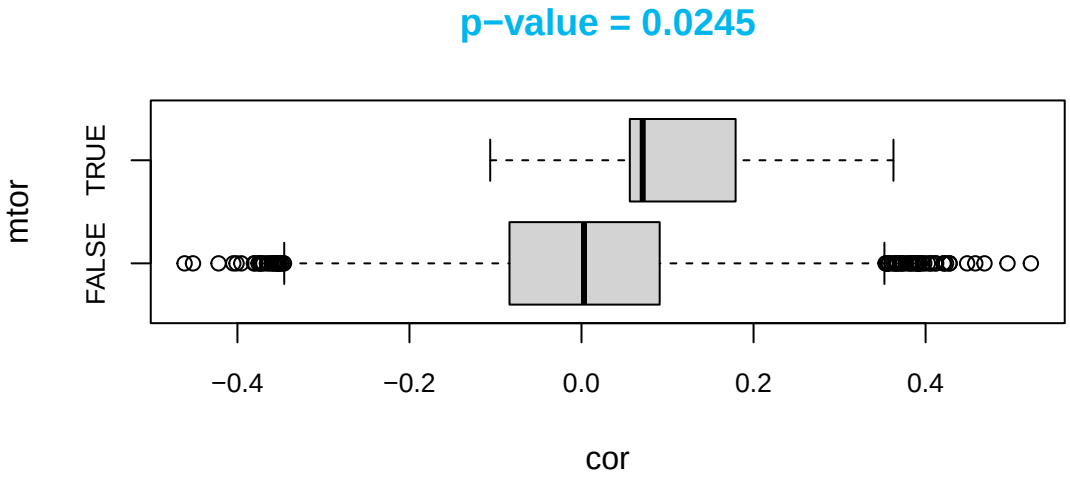
```r
t.test(cor~mtor,data=genesCorr)
```

```
##
##  Welch Two Sample t-test
##
## data:  cor by mtor
## t = -2.3422, df = 10.01, p-value = 0.04116
## alternative hypothesis: true difference in means between group FALSE and group TRUE is not equal to 0
## 95 percent confidence interval:
##  -0.196138545 -0.004909809
## sample estimates:
## mean in group FALSE  mean in group TRUE
##         0.003825099         0.104349276
```

```r
t.test(cor~mtor,data=genesCorr,var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  cor by mtor
## t = -2.6181, df = 17647, p-value = 0.00885
## alternative hypothesis: true difference in means between group FALSE and group TRUE is not equal to 0
## 95 percent confidence interval:
##  -0.17578477 -0.02526358
## sample estimates:
## mean in group FALSE  mean in group TRUE
##         0.003825099         0.104349276
```

```r
# boxplot
boxplot(cor~mtor,data=genesCorr,
        main=paste0("p-value = ",
        round(wilcox.test(cor~mtor,data=genesCorr)$p.value,4)),
        col.main="#00B6ED",horizontal=TRUE,cex.axis=.8)
```


p−value = 0.0245
mtor
TRUE
FALSE
−0.4
−0.2
0.0
0.2
0.4
cor

6 Figure 3B

We present here the code allowing to reproduce analysis presented in Figure 3B.

6.1 Input

This analysis is based on the input file 'input/figure_3b.csv' which, for each gene of the TCGA OV cohort (rows), reports

- the lab response to different drugs, measured by the area under the curve (AUC, refer to Section 13 for detail),
- the relative copy number (RCN) for different genes of interest.

The following Table shows a subset of the data.

```
# import:
figure_3b = read.csv("input/figure_3b.csv",row.names=1)
kable(figure_3b[1:6,],format="simple")
```

	logFC	t.value	p.value
MYC|4609	2.7285458	29.161669	0
MAP4|4134	0.7330158	9.231323	0
VPS9D1-AS1|100128881	1.3467736	9.056031	0
CGREF1|10669	1.6013730	8.768168	0
VGLL4|9686	0.8164458	8.390927	0
AR|367	2.0042059	8.197872	0

6.2 GSEA

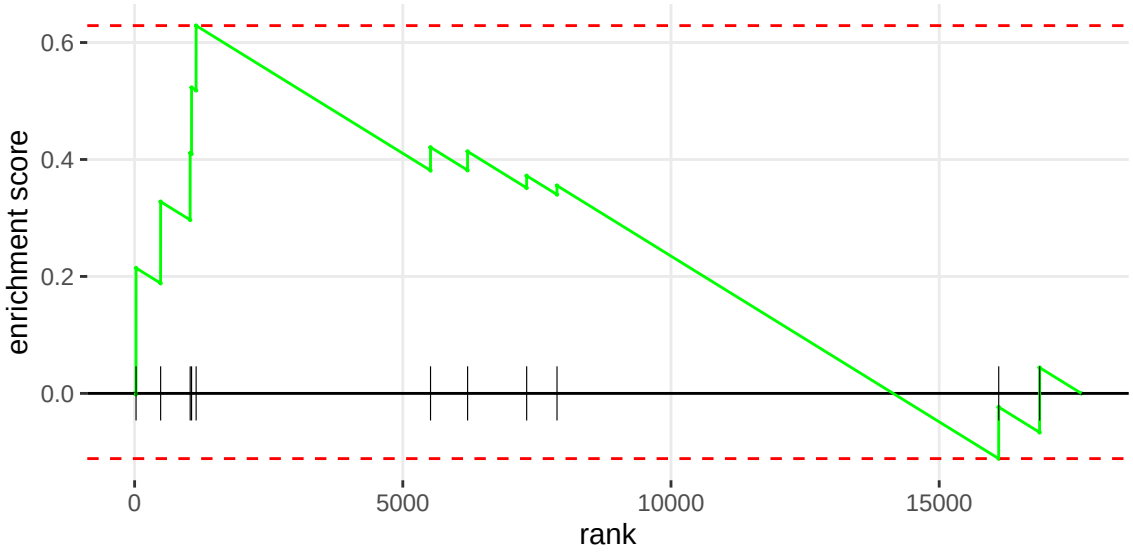
We first run the GSEA test of interest

```
# list of mTOR pathway genes
bioplanet_mtor <- c("EIF4B|1975","EIF4E|1977","EIF4EBP1|1978","EIF4G1|1981",
                    "MTOR|2475","RPS6|6194","RPS6KB1|6198","RHEB|6009",
                    "EEF2K|29904","RPTOR|57521","MLST8|64223")
# GSEA test
set.seed(123)
tval        = figure_3b$t.value
names(tval) = rownames(figure_3b)
gsea.test   = fgsea::fgsea(pathways = list("mtor"=bioplanet_mtor),
                           stats = tval,
                           nperm=10000,
                           BPPARAM=BiocParallel::MulticoreParam(workers = 1))
# organise results
as.data.frame(gsea.test)[,1:7]
```

pathway	pval	padj	ES	NES	nMoreExtreme	size
mtor	0.0250965	0.0250965	0.6288991	1.623818	142	11

and finally generate the corresponding GSEA enrichment plot.

```
fgsea::plotEnrichment(bioplanet_mtor, tval)
```



7 Figure 3C, 4A, 4B and 4C

We present here the code allowing to reproduce the results of the permutation tests presented in Figures 3C, 4A, 4B and 4C.

7.1 Input

These analyses are based on the following input files

- 'input/figure_3c4a4b4c.csv': for each cancer gene (refer to reference 49 of the manuscript for detail) not belonging to 8Q at the exception of MYC, reports the following information (columns):
  - **id**: gene name,
  - **pathway**: a logical vector, coded 'TRUE' for cancer genes belonging to the to the PI3K or RAS pathways based on the Reactome definition,
  - **type**: a two-level factor indicating if the gene of interest is an oncogene (og) or a tumour suppressor (tsg),
- 'input/figure3c.csv': for each cancer gene (rows), reports the estimated CNA (with values -2 [homozygous loss] to 2 [amplification]) of all samples of the **TCGA Ovarian Cancer cohort** (column),
- 'input/figure4a.csv': for each cancer gene (rows), reports the estimated CNA (with values -2 [homozygous loss] to 2 [amplification]) of all samples of the **TCGA Breast Cancer cohort** (column),
- 'input/figure4b.csv': for each cancer gene (rows), reports the estimated CNA (with values -2 [homozygous loss] to 2 [amplification]) of all samples of the **METABRIC Breast Cancer cohort** (column),
- 'input/figure4c.csv': for each cancer gene (rows), reports the estimated CNA (with values -2 [homozygous loss] to 2 [amplification]) of all samples of the **TCGA Lung SSC cancer cohort** (column).

```
# import:
figure_3c4a4b4c = read.csv("input/figure_3c4a4b4c.csv",row.names=1)
figure_3c = read.csv("input/figure_3c.csv",row.names=1)
figure_4a = read.csv("input/figure_4a.csv",row.names=1)
figure_4b = read.csv("input/figure_4b.csv",row.names=1)
figure_4c = read.csv("input/figure_4c.csv",row.names=1)
```

The following Table shows a subset of the gene file *figure_3c4a4b4c*.

```
figure_3c4a4b4c[1:10,]
```

	type	pathway
MYC	og	TRUE
RPL22	tsg	FALSE
HES3	tsg	FALSE
HES2	tsg	FALSE
ERRFI1	tsg	FALSE
PIK3CD1	og	TRUE
MTOR	og	TRUE
SPEN	tsg	FALSE
EPHA2	tsg	FALSE
SDHB	tsg	FALSE

The following Table shows a subset of the CNA file *figure6c* (TCGA Ovarian Cancer cohort)

```
figure_4a[1:10,1:4]
```

	TCGA.A1.A0SI.01	TCGA.A1.A0SP.01	TCGA.A2.A04P.01	TCGA.A2.A04T.01
MYC	0	1	2	2
RPL22	0	1	1	0
HES3	0	1	1	0
HES2	0	1	1	0
ERRFI1	0	1	1	0
PIK3CD	0	1	1	0
MTOR	0	1	-1	0
SPEN	0	1	-1	0
EPHA2	0	1	-1	0
SDHB	0	1	-1	0

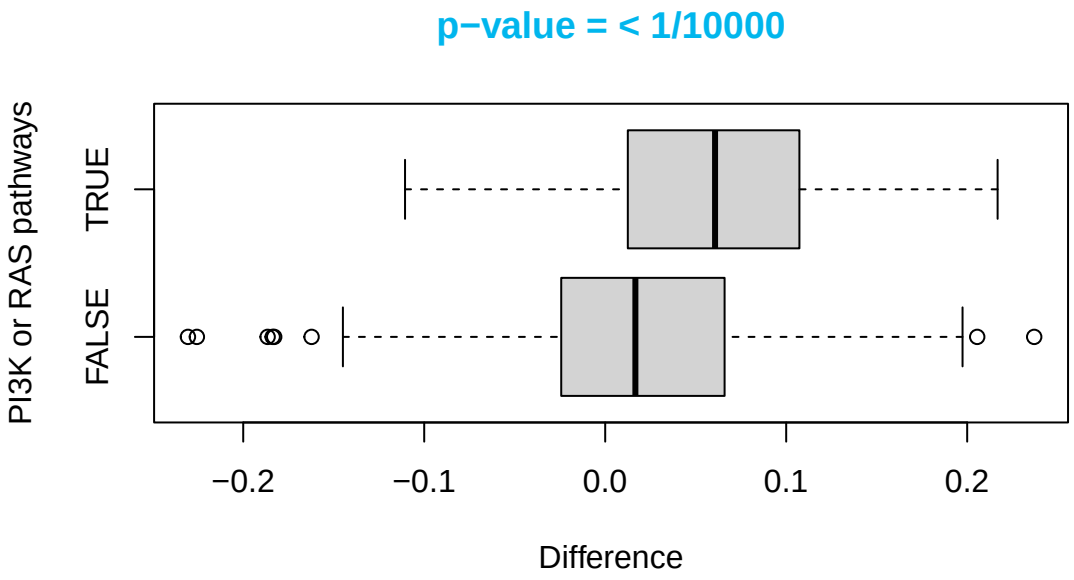
7.2 Permutation tests

In this Section, we present the code of the permutation tests for each cohort of interest. (Takes a few minutes).

7.2.1 TCGA Ovarian Cancer cohort (3C)

```
# reverse scale for tsg genes
figure_3c[figure_3c4a4b4c$type=="tsg",] = -figure_3c[figure_3c4a4b4c$type=="tsg",]
# identify samples with myc gain or amplification
myc_ampl.i = figure_3c["MYC",]>0
# useful
diff.fun = function(x,g){
        mean(x[g]>0,na.rm=TRUE)-mean(x[!g]>0,na.rm=TRUE)
        }
# estimates
figure_3c4a4b4c$diff.g = apply(figure_3c,1,diff.fun,g=myc_ampl.i)
boxplot(diff.g~pathway,data=figure_3c4a4b4c[-1,],
        ylab="PI3K or RAS pathways",xlab="Difference",horizontal=TRUE)
# mean difference of interest
diff0 = mean((figure_3c4a4b4c$diff.g[-1])[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean((figure_3c4a4b4c$diff.g[-1])[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
# randomise order of participants to both conditions
n.R = 1e+4
n.i = ncol(figure_3c)
set.seed(1)
mx.order.Ri = t(apply(matrix(runif(n.i*n.R),ncol=n.R),2,order))
if(!any(dir("input")=="figure_3c_diffstar")){
    diffstar = apply(mx.order.Ri,1,function(x){
        diff.g = apply(figure_3c[,x],1,diff.fun,g=myc_ampl.i)[-1]
        mean(diff.g[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean(diff.g[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
        })
    save(diffstar,file="input/figure_3c_diffstar")
}else{
    load("input/figure_3c_diffstar")
```

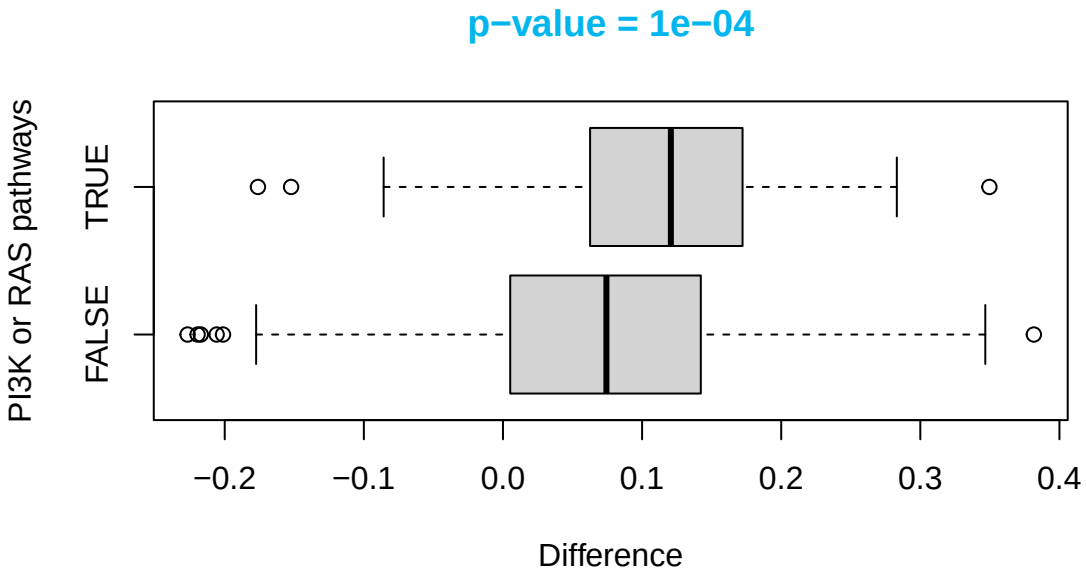
```
    }
# one-sided test result
pval = round(mean(diffstar>diff0),4)
title(paste0("p-value = ",ifelse(pval==0,"< 1/10000",pval)),
      col.main="#00B6ED")
```



7.2.2 TCGA Breast Cancer cohort (4A)

```
# reverse scale for tsg genes
figure_4a[figure_3c4a4b4c$type=="tsg",] = -figure_4a[figure_3c4a4b4c$type=="tsg",]
# identify samples with myc gain or amplification
myc_ampl.i = figure_4a["MYC",]>0
# useful
diff.fun = function(x,g){
        mean(x[g]>0,na.rm=TRUE)-mean(x[!g]>0,na.rm=TRUE)
        }
# estimates
figure_3c4a4b4c$diff.g = apply(figure_4a,1,diff.fun,g=myc_ampl.i)
boxplot(diff.g~pathway,data=figure_3c4a4b4c[-1,],
        ylab="PI3K or RAS pathways",xlab="Difference",horizontal=TRUE)
# mean difference of interest
diff0 = mean((figure_3c4a4b4c$diff.g[-1])[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean((figure_3c4a4b4c$diff.g[-1])[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
# randomise order of participants to both conditions
n.R = 1e+4
n.i = ncol(figure_4a)
set.seed(1)
mx.order.Ri = t(apply(matrix(runif(n.i*n.R),ncol=n.R),2,order))
if(!any(dir("input")=="figure_4a_diffstar")){
    diffstar = apply(mx.order.Ri,1,function(x){
        diff.g = apply(figure_4a[,x],1,diff.fun,g=myc_ampl.i)[-1]
        mean(diff.g[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean(diff.g[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
        })
```

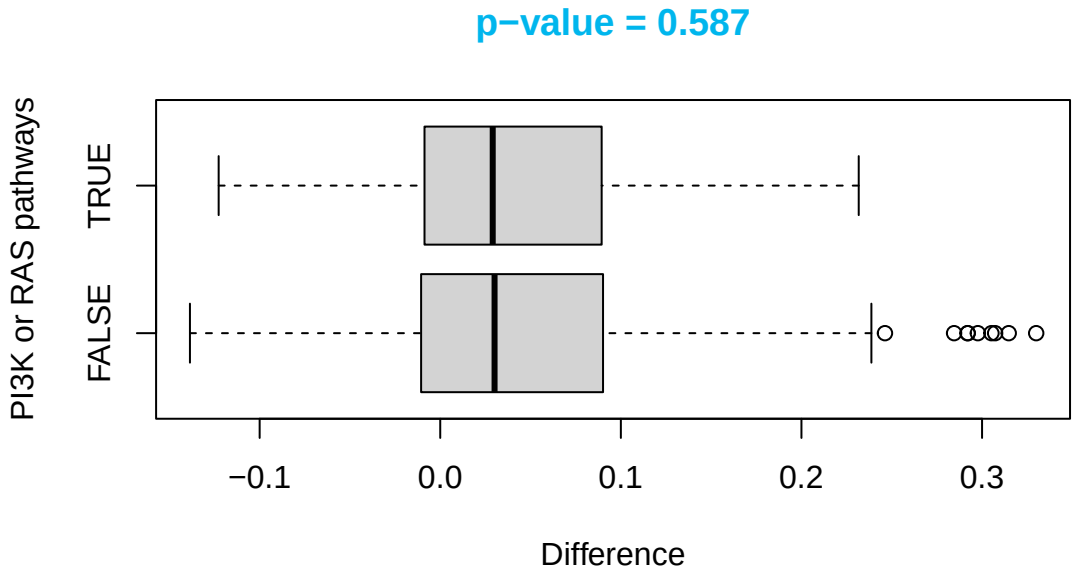
```
    save(diffstar,file="input/figure_4a_diffstar")
}else{
    load("input/figure_4a_diffstar")
    }
# one-sided test result
pval = round(mean(diffstar>diff0),4)
title(paste0("p-value = ",ifelse(pval==0,"< 1/10000",pval)),
      col.main="#00B6ED")
```



7.2.3 METABRIC Breast Cancer cohort (4B)

```
# reverse scale for tsg genes
figure_4b[figure_3c4a4b4c$type=="tsg",] = -figure_4b[figure_3c4a4b4c$type=="tsg",]
# identify samples with myc gain or amplification
myc_ampl.i = figure_4b["MYC",]>0
# useful
diff.fun = function(x,g){
        mean(x[g]>0,na.rm=TRUE)-mean(x[!g]>0,na.rm=TRUE)
        }
# estimates
figure_3c4a4b4c$diff.g = apply(figure_4b,1,diff.fun,g=myc_ampl.i)
boxplot(diff.g~pathway,data=figure_3c4a4b4c[-1,],
        ylab="PI3K or RAS pathways",xlab="Difference",horizontal=TRUE)
# mean difference of interest
diff0 = mean((figure_3c4a4b4c$diff.g[-1])[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean((figure_3c4a4b4c$diff.g[-1])[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
# randomise order of participants to both conditions
n.R = 1e+4
n.i = ncol(figure_4b)
set.seed(1)
mx.order.Ri = t(apply(matrix(runif(n.i*n.R),ncol=n.R),2,order))
if(!any(dir("input")=="figure_4b_diffstar")){
    diffstar = apply(mx.order.Ri,1,function(x){
        diff.g = apply(figure_4b[,x],1,diff.fun,g=myc_ampl.i)[-1]
```

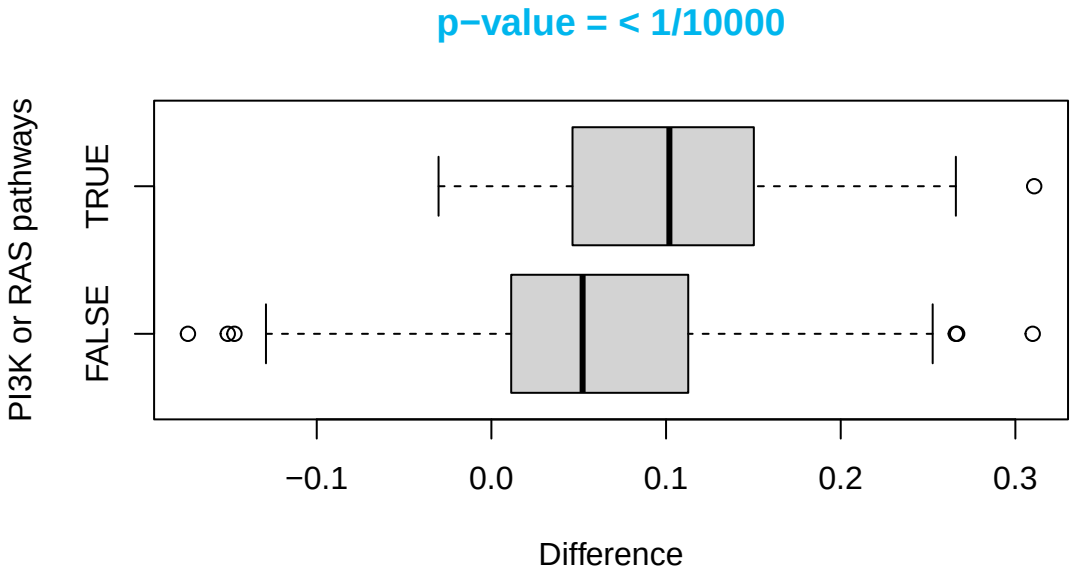
```
        mean(diff.g[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean(diff.g[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
        })
    save(diffstar,file="input/figure_4b_diffstar")
}else{
    load("input/figure_4b_diffstar")
    }
# one-sided test result
pval = round(mean(diffstar>diff0),4)
title(paste0("p-value = ",ifelse(pval==0,"< 1/10000",pval)),
      col.main="#00B6ED")
```



7.2.4 TCGA Lung Cancer cohort (4C)

```
# reverse scale for tsg genes
figure_4c[figure_3c4a4b4c$type=="tsg",] = -figure_4c[figure_3c4a4b4c$type=="tsg",]
# identify samples with myc gain or amplification
myc_ampl.i = figure_4c["MYC",]>0
# useful
diff.fun = function(x,g){
        mean(x[g]>0,na.rm=TRUE)-mean(x[!g]>0,na.rm=TRUE)
        }
# estimates
figure_3c4a4b4c$diff.g = apply(figure_4c,1,diff.fun,g=myc_ampl.i)
boxplot(diff.g~pathway,data=figure_3c4a4b4c[-1,],
        ylab="PI3K or RAS pathways",xlab="Difference",horizontal=TRUE)
# mean difference of interest
diff0 = mean((figure_3c4a4b4c$diff.g[-1])[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean((figure_3c4a4b4c$diff.g[-1])[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
# randomise order of participants to both conditions
n.R = 1e+4
n.i = ncol(figure_4c)
set.seed(1)
mx.order.Ri = t(apply(matrix(runif(n.i*n.R),ncol=n.R),2,order))
```

```r
if(!any(dir("input")=="figure_4c_diffstar")){
    diffstar = apply(mx.order.Ri,1,function(x){
        diff.g = apply(figure_4c[,x],1,diff.fun,g=myc_ampl.i)[-1]
        mean(diff.g[ figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)-
        mean(diff.g[!figure_3c4a4b4c$pathway[-1]],na.rm=TRUE)
        })
    save(diffstar,file="input/figure_4c_diffstar")
}else{
    load("input/figure_4c_diffstar")
    }
# one-sided test result
pval = round(mean(diffstar>diff0),4)
title(paste0("p-value = ",ifelse(pval==0,"< 1/10000",pval)),
      col.main="#00B6ED")
```



8 Figure 3D

We present here the code allowing to reproduce the results of the association tests presented in Figure 3D.

8.1 Input

This analysis is based on the input file 'input/figure_3d.csv' which, for each sample of the TCGA ovarian cohort (rows, n=579), reports the estimated CNA for 11 HGSOC drivers genes associated with the PI3K pathway (columns). Information is coded as follows:

- -2: homozygous deletion,
- -1: loss,
- 0: normal,
- +1: gain,
- +2: amplification.

The following Table shows a subset of the data.

```
# import:
figure_3d = read.csv("input/figure_3d.csv",row.names=1)
kable(figure_3d[1:6,],format="simple")
```

	MYC	PTEN	GAB2	PTK6	IGF1R	PIK3CA	NF1	KRAS	AKT1	AKT2	AKT3
TCGA.04.1331.01	0	-2	-1	2	2	2	-1	0	0	0	2
TCGA.04.1332.01	1	0	0	1	0	1	-1	1	-1	2	0
TCGA.04.1335.01	1	-1	0	0	0	1	-1	0	0	-1	0
TCGA.04.1336.01	2	0	0	0	1	2	-2	-1	0	0	0
TCGA.04.1337.01	0	0	0	1	0	1	-1	-1	0	0	0
TCGA.04.1338.01	2	1	-1	2	0	2	-1	1	1	-1	1

8.2 Analyses

For each gene of interest, we report

- **oncogene**: a logical vector, coded 'TRUE' for oncogenes and 'FALSE' for tumour suppressors,
- **pc_myc=2**: the % of cases with amplification (+2) of the oncogene of interest or with homozygous loss (-2) of the tumour suppressor of interest among samples with MYC amplification (2),
- **pc_myc<2**: the % of cases without amplification (-2, -1, 0, +1) of the oncogene of interest or without homozygous loss (-1, 0, +1, +2) of the tumour suppressor of interest among samples without MYC amplification (-2, -1, 0, +1),
- **pval_chi**: the p-value of a Chi-square test of association of two categorical 3-level factors,
- **pval_cmh**: the p-value of a Generalised Cochran-Mantel-Haenszel test of association of two ordered 3-level factors.

```
# object
res = as.data.frame(matrix(nrow=ncol(figure_3d)-1,ncol=5))
colnames(res) = c("oncogene","pc_myc=2","pc_myc<2","pval_chi","pval_cmh")
rownames(res) = (colnames(figure_3d)[-1])[order(colnames(figure_3d)[-1])]
# driver gene type
res$oncogene = TRUE
res[c("NF1","PTEN"),"oncogene"] = FALSE
```

```
# useful
pval.fun = function(pval,digit=4){
    out = format(c(.an(.p("0.",.p(rep(1,digit),collapse=""))),round(pval,digits=digit)))[-1]
    out[is.na(pval)] = ""
    out[!is.na(match(out,c("0.00000","0.0000","0.000","0.00","0.0","0")))] =
        .p("<0.",.p(rep(0,digit-1),collapse=""),"1")
    out
    }
# loop for each gene
for(gw in 1:nrow(res)){
    data = table(factor(figure_3d[,rownames(res)[gw]],levels=-2:2),
                 factor(figure_3d[,"MYC"],levels=-2:2))
    # % of interest
    lw = ifelse(res$oncogene[gw],"2","-2")
    res$'pc_myc=2'[gw] = data[lw,"2"]/sum(data[,"2"])
    res$'pc_myc<2'[gw] = sum(data[lw,colnames(data)!="2"])/
                         sum(data[,colnames(data)!="2"])
    # recode 5-level factors into 3-level factors
    levelw = if(res$oncogene[gw]){c(1,1,1,2,3)}else{c(3,2,1,1,1)}
    tablew = table(levelw[as.numeric(factor(figure_3d[,rownames(res)[gw]],
                  levels=-2:2))],
                  c(1,1,1,2,3)[as.numeric(factor(figure_3d[,"MYC"],
                  levels=-2:2))])
    # chi-square and CMH tests
    res$'pval_chi'[gw] = pval.fun(chisq.test(tablew)$p.value)
    res$'pval_cmh'[gw] = pval.fun(vcdExtra::CMHtest(tablew)[[1]]["cor","Prob"])
    }
# print
res
```

	oncogene	pc_myc=2	pc_myc<2	pval_chi	pval_cmh
AKT1	TRUE	0.0617284	0.0446429	0.0367	0.0035
AKT2	TRUE	0.0534979	0.0982143	0.0107	0.1085
AKT3	TRUE	0.1275720	0.0714286	0.0007	0.4418
GAB2	TRUE	0.1769547	0.0952381	0.0002	0.0009
IGF1R	TRUE	0.1152263	0.0476190	0.0013	0.0012
KRAS	TRUE	0.1358025	0.1309524	0.0001	0.9569
NF1	FALSE	0.0781893	0.0625000	0.0038	0.0065
PIK3CA	TRUE	0.3868313	0.2172619	<0.0001	0.0003
PTEN	FALSE	0.0617284	0.0327381	0.0015	0.0334
PTK6	TRUE	0.1975309	0.1160714	0.0004	0.2929

9 Figure S2C

We present here the code allowing to reproduce Figure S2C.

9.1 Input

This analysis is based on the input file 'input/figure_s2c.csv' which, for each gene (rows), reports the relative copy number (on the log2 scale) for each individual (3-digit ids) genomic segment (rows) in tumour samples ('_tumour' columns) and in matched spheroid ('_spheroid' columns).

The following Table shows a subset of the data.

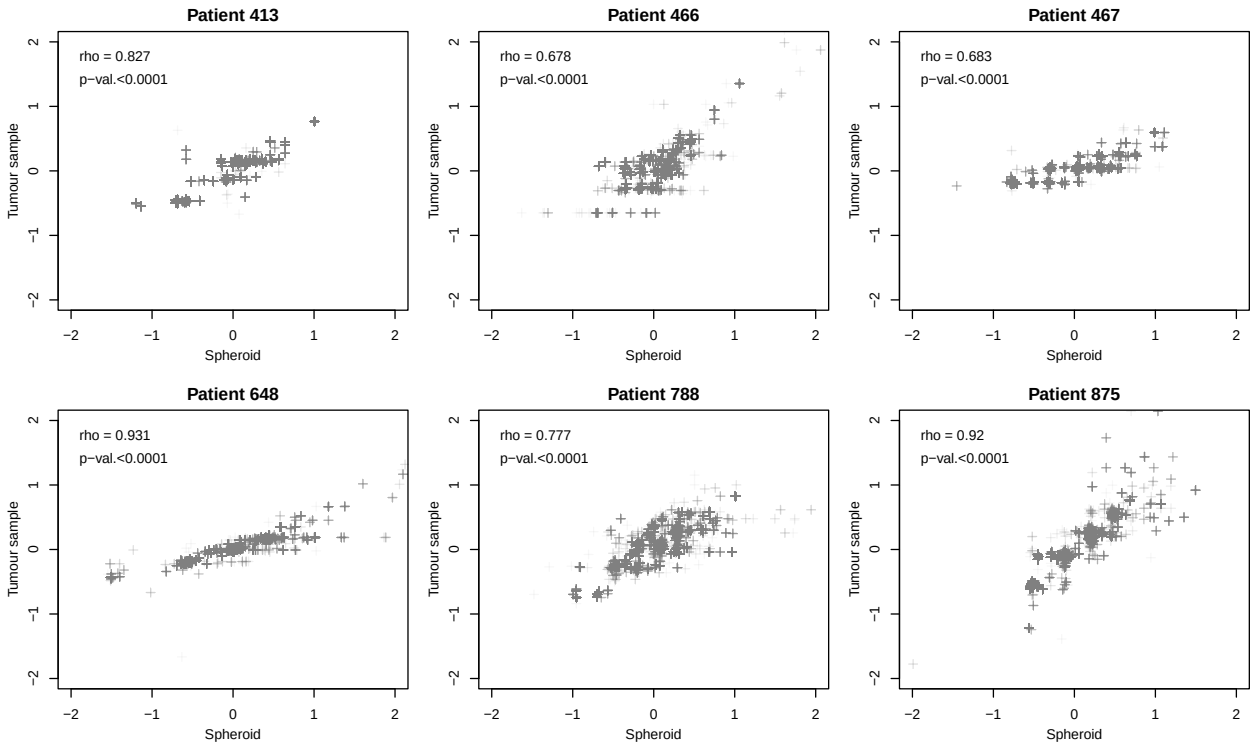
```
# import:
figure_s2c = read.csv("input/figure_s2c.csv",row.names=1)
kable(figure_s2c[1:10,c(1,7,2,8)],format="simple")
```

	X413_spheroid	X413_tumour	X466_spheroid	X466_tumour
A1BG	0.1493322	0.0800524	-0.0323971	-0.2517887
NAT2	-1.1951130	-0.4974336	-0.2853170	-0.6509205
ADA	0.3552261	0.1866726	0.4564536	0.2391914
CDH2	-0.7081651	NA	-0.3509781	-0.2859235
AKT3	0.0951364	0.1613873	0.2412423	0.3309614
ZBTB11-AS1	0.1680628	-0.1452935	0.3565230	0.4470678
MED6	-0.1487642	0.1773827	-0.3528885	-0.2672544
NR2E3	0.0287285	0.1769396	-0.1608073	-0.2548752
NAALAD2	-0.1334255	0.0699524	0.1723468	0.1879973
SNORD116-1	-0.0225750	0.1354289	-0.1616039	-0.2548752

9.2 Plot

This code produce the relative copy number comparison of spheroid and primary tumour for each patient of interest and reports the spearman correlation coefficient (and corresponding p-value).

```
par(mfrow=c(2,3),mar=c(4,4,2,0.25))
patient_overlap = unique(substr(colnames(figure_s2c),1,4))
for(pw in 1:length(patient_overlap)){
plot(figure_s2c[,paste0(patient_overlap[pw],"_spheroid")],
     figure_s2c[,paste0(patient_overlap[pw],"_tumour")],
     xlim=c(-2,2),ylim=c(-2,2),pch=3,col=paste0(gray(.5),"10"),
     main = .p("Patient ",substr(patient_overlap[pw],2,4)),
     xlab = "", ylab = "")
     axis(1,0,"Spheroid",padj=2,tick=FALSE)
     axis(2,0,"Tumour sample",padj=-2,tick=FALSE)
     corw = cor.test(mx.rcn.Gnw_sph[,pw],mx.rcn.Gnw_ovo4[,pw],
                     method="spearman",use="pairwise.complete.obs")
     text(-2,1.75,.p("rho = ",round(corw$estimate,3)),col=1,pos=4)
     text(-2,1.4,.p("p-val.",.pval(corw$p.value)),col=1,pos=4)
}
```


Patient 413
rho = 0.827
p−val.<0.0001
Patient 466
rho = 0.678
p−val.<0.0001
Patient 467
rho = 0.683
p−val.<0.0001
Patient 648
rho = 0.931
p−val.<0.0001
Patient 788
rho = 0.777
p−val.<0.0001
Patient 875
rho = 0.92
p−val.<0.0001
Tumour sample
Spheroid

10 Figure S4E

We present here the code allowing to reproduce the cluster analysis presented in Figure S4E.

10.1 Input

This analysis is based on the input file 'input/figure_s4e.csv' which, for each sample (rows), reports

- the lab response to different drugs, measured by the area under the curve (AUC, refer to Section 13 for detail),
- the relative copy number (RCN) for different genes of interest.

The following Table shows a subset of the data.

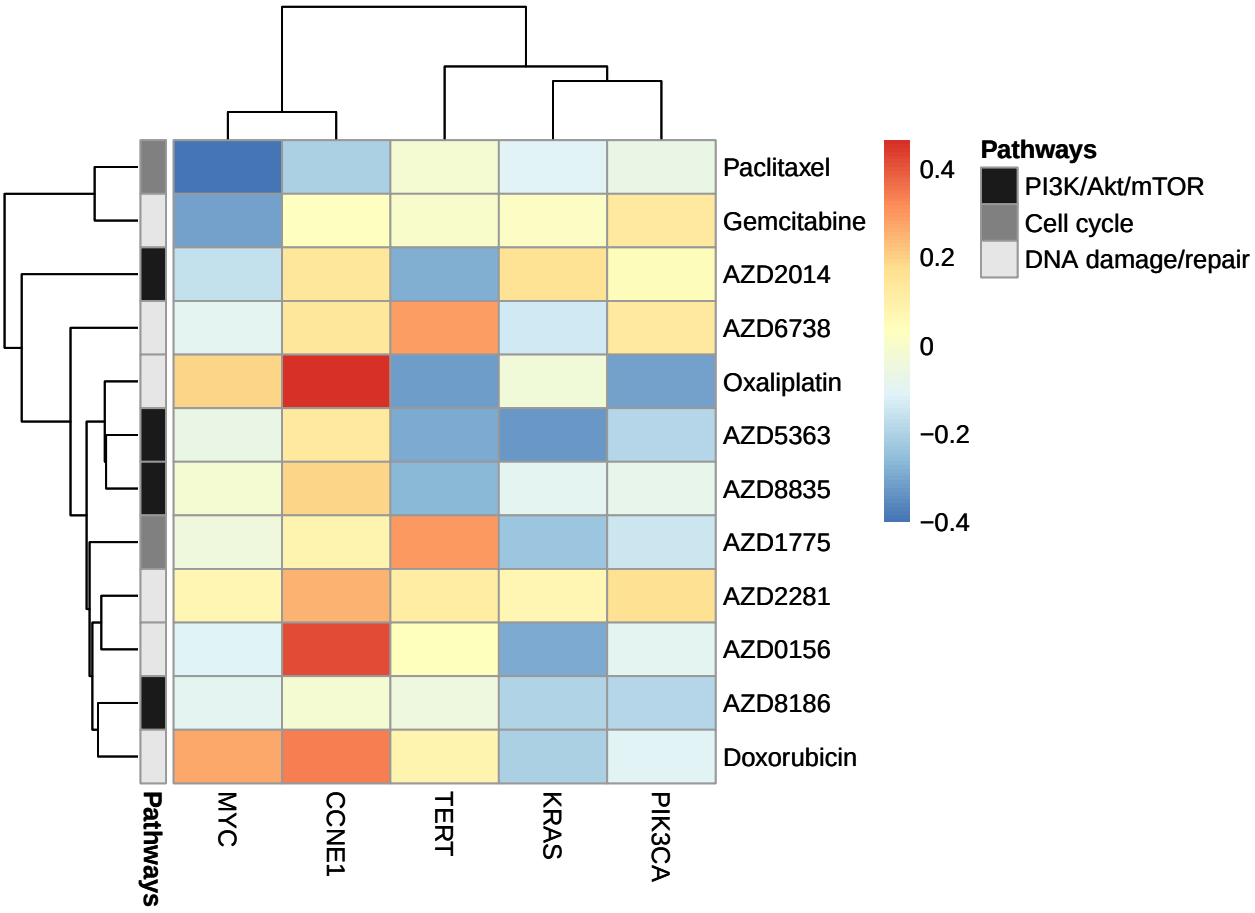
```
# import:
figure_s4e = read.csv("input/figure_s4e.csv",row.names=1)
kable(figure_s4e[1:6,1:7],format="simple")
```

	KRAS	MYC	PIK3CA	TERT	CCNE1	AZD8186	AZD2014
294-s1	0.4397947	0.9071470	0.5876543	-0.0330413	0.1426132	0.7572422	0.5757075
333-s1	-0.1339190	1.7651285	-0.1826986	0.2715367	0.1985729	0.9699401	0.6004840
364-s1	0.6892377	1.3217915	0.7621503	0.4679765	-0.1334255	0.7232977	0.4932141
409-s1	0.1963729	0.9647170	1.1596707	1.1219111	1.0813844	0.7509781	0.3835962
409-s2	0.1924085	0.9208771	1.0383084	1.0155328	1.1049335	0.9366704	0.6186606
413-s1	-0.0796468	1.0049088	0.5641125	-0.0699756	0.1493322	0.8303464	0.5045604

10.2 Cluster analysis

```
# list of drivers and drugs
id.driver = colnames(figure_s4e)[1:5]
n.driver  = length(id.driver)
id.drug   = colnames(figure_s4e)[-c(1:5)]
n.drug    = length(id.drug)
# pathway
pathway = data.frame(Pathways=rep("DNA damage/repair",n.drug),
                     row.names=id.drug)
pathway$Pathways[!is.na(match(id.drug,
        c("AZD2014","AZD5363","AZD8186","AZD8835")))] =
        'PI3K/Akt/mTOR'
pathway$Pathways[!is.na(match(id.drug,
        c("Paclitaxel","AZD1775")))] =
        'Cell cycle'
col2 = list(Pathways = c('PI3K/Akt/mTOR' = gray(.1),
                         'Cell cycle' = gray(.5),
                         'DNA damage/repair' = gray(.9)))
# estimates
mx.cor.dc = matrix(NA,nrow=n.drug,ncol=n.driver,
                   dimnames=list(id.drug,id.driver))
for(dw in 1:n.drug){
    for(cw in 1:n.driver){
        mx.cor.dc[dw,cw] = cor(figure_s4e[,id.drug[dw]],figure_s4e[,id.driver[cw]],
                               method="pearson")
```

```
        }
    }
# clustering
clust2 = pheatmap::pheatmap(mx.cor.dc,
             annotation_row=pathway,
             clustering_distance_rows = "correlation",
             clustering_distance_cols = "correlation",
             clustering_method = "single",
             annotation_names_row = TRUE,
             labels_row = id.drug,
             annotation_colors = col2,annotation_legend=TRUE,
             show_rownames=TRUE, scale="none")
clust2
```



11 Figure S4D

We present here the code used to analyse the relationship between

- clinical response to Paclitaxel (often combined with Carboplatin), measured by the difference of CA125 levels on the log2 scale after and before treatment,
- MYC relative copy number (RCN).

As the same patient may have several lines of treatment with Paclitaxel, linear mixd models were used to take the potential within-patient dependence into account.

11.1 Input

This analysis is based on the input file 'input/figure_s4d.csv' which, for each chemotherapy line (rows), reports the following information (columns):

- **patient**: the patient ID,
- **ca125_diff**: clinical response (difference of CA125 levels on the log2 scale before and after treatment),
- **MYC**: relative copy numbers,
- **purity**: tumour purity estimate (TP53 allele frequency).

The following Table shows a subset of the data.

```
# import:
figure_s4d = read.csv("input/figure_s4d.csv")
kable(figure_s4d[1:6,],format="simple")
```

patient	line	ca125_diff	MYC	purity
22	1	1.4365369	1.050686	0.1225
75	1	-3.3489274	1.248346	0.4865
75	2	-2.0614005	1.248346	0.4865
88	1	-3.2016339	1.802701	0.3100
96	1	-2.0334707	2.046803	0.3745
189	1	-0.7259072	1.829991	0.7245

11.2 Analyses

Random intercept linear mixed model modelling clinical response as a function of MYC RCN on the log scale (fixed effect) and patient (random effect).

```
summary(lme(ca125_diff~log(MYC),random=~1|patient,data=figure_s4d))
```

```
## Linear mixed-effects model fit by REML
##   Data: figure_s4d
##        AIC      BIC    logLik
##   184.8004 191.4547 -88.40021
##
## Random effects:
##  Formula: ~1 | patient
##         (Intercept) Residual
## StdDev:    1.561854 1.527085
##
## Fixed effects:  ca125_diff ~ log(MYC)
##                 Value Std.Error DF   t-value p-value
```

```
## (Intercept) -2.231297 0.4532457 36 -4.922931  0.0000
## log(MYC)    -1.318578 0.6492268 36 -2.030997  0.0497
##  Correlation:
##          (Intr)
## log(MYC) -0.636
##
## Standardized Within-Group Residuals:
##        Min         Q1        Med         Q3        Max
## -1.2252030 -0.7158970  0.2559393  0.5606308  1.1947609
##
## Number of Observations: 41
## Number of Groups: 38
```

Random intercept linear mixed model modelling clinical response as a function of MYC RCN on the log scale and tumour purity (fixed effects) and patient (random effect).

```
summary(lme(ca125_diff~log(MYC)+purity,random=~1|patient,data=figure_s4d))
```

```
## Linear mixed-effects model fit by REML
##   Data: figure_s4d
##        AIC      BIC    logLik
##   179.3047 187.4927 -84.65237
##
## Random effects:
##  Formula: ~1 | patient
##          (Intercept) Residual
## StdDev: 0.0003811168 2.050489
##
## Fixed effects:  ca125_diff ~ log(MYC) + purity
##                 Value Std.Error DF   t-value p-value
## (Intercept) -3.286602 0.6085116 35 -5.401051  0.0000
## log(MYC)    -2.090303 0.6993959 35 -2.988727  0.0051
## purity       3.226542 1.4223193 35  2.268508  0.0296
##  Correlation:
##          (Intr) l(MYC)
## log(MYC)  0.006
## purity   -0.737 -0.506
##
## Standardized Within-Group Residuals:
##        Min         Q1        Med         Q3        Max
## -2.0832026 -0.8413232  0.1543666  0.7248313  2.1610652
##
## Number of Observations: 41
## Number of Groups: 38
```

12 Figures S5A and S5B

We present here the code used to analyse the relationship between

- different measures of changes in CA125 levels during a time interval,
- the CT scan assessment at the end of the same time interval.

12.1 Input

This analysis is based on the input file 'input/figure_s5as5b.csv' which reports, for each CT scan segment of less than 100 days (rows), the following information (columns):

- **patient**: the patient id corresponding to each CT scan segment,
- **ct.end**: the CT 4-level ordinal assessment of the tumour evolution at the end of the time segment of interest, with 1 = 'complete response', 2 = 'partial response', 3 = 'stable disease' and 4 = 'progressive disease',
- **TDL to MRO**: 18 different CA125 measures related to the segment of interest, where
  - the 1st letter refers to the method to estimate CA125 measure between two time points: **T** for the triangular method, **M** for an M-spline approximation.
  - the 2nd letter refers to the CA125 function used between the 2 segment boundaries: **D** for the difference, **S** for the slope, **R** for ratio
  - the 3rd letter refers to the CA125 scale: **L** for the log2 scale, **C** for the cube root scale, **O** for the linear/original scale.

The following Table shows a subset of the data.

```
# import:
figure_s5as5b = read.csv("input/figure_s5as5b.csv")
kable(figure_s5as5b[1:6,1:7],format="simple")
```

patient	ct.end	TDL	TDC	TDO	TSL	TSC
47	4	0.4906847	0.4146030	16.767031	0.0080440	0.0067968
47	3	0.1422932	0.1258215	5.404762	0.0033091	0.0029261
47	2	-1.9333926	-1.9027688	-108.824017	-0.0227458	-0.0223855
47	3	-0.0809173	-0.0624948	-2.095238	-0.0013265	-0.0010245
47	4	0.5122935	0.4168418	15.571429	0.0056296	0.0045807
156	3	-0.3219281	-0.2344773	-7.000000	-0.0041809	-0.0030452

12.2 Figure S5A

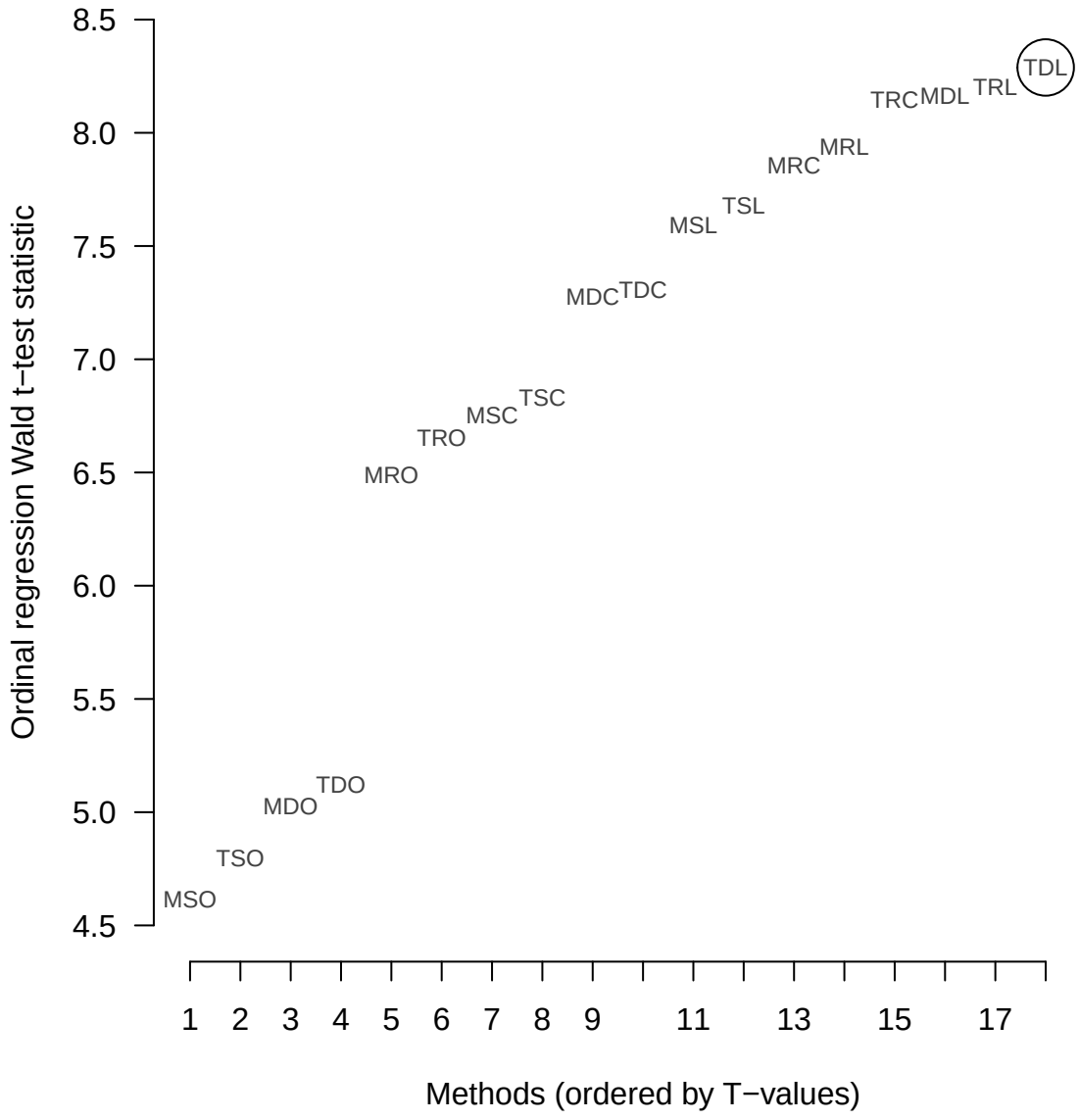
We first perform 18 ordinal regressions with the 4-level CT assessement as ordinal response and each of the different CA125 selected measures as predictor.

We then record two measures of strength of the relationship and goodness of fit:

- the Wald t-statistic,
- the weighted kappa between the observed and model predicted CT 3-level assessment.

We finally display the ordered the Wald t-statistic for each ordered selected CA125 measure:

```
#
par(mfcol=c(1,1),mar=c(4,4,0,0))
mx.hat.m2 = mx.hat.m2[order(mx.hat.m2[,1]),]
plot(mx.hat.m2[,1],xlab="Methods (ordered by T-values)",ylab="Ordinal regression Wald t-test statistic"
     xlim= c(1,nrow(mx.hat.m2)+1),axes=FALSE,pch="",ylim=c(4.5,8.5))
axis(1,1:nrow(mx.hat.m2))
axis(2,seq(4.5,8.5,.5),las=2)
text(1:nrow(mx.hat.m2),mx.hat.m2[,1],rownames(mx.hat.m2),
     col=gray(.25),cex=.75)
points(n.measure,mx.hat.m2[n.measure,"t-value"],pch=1,cex=4)
```

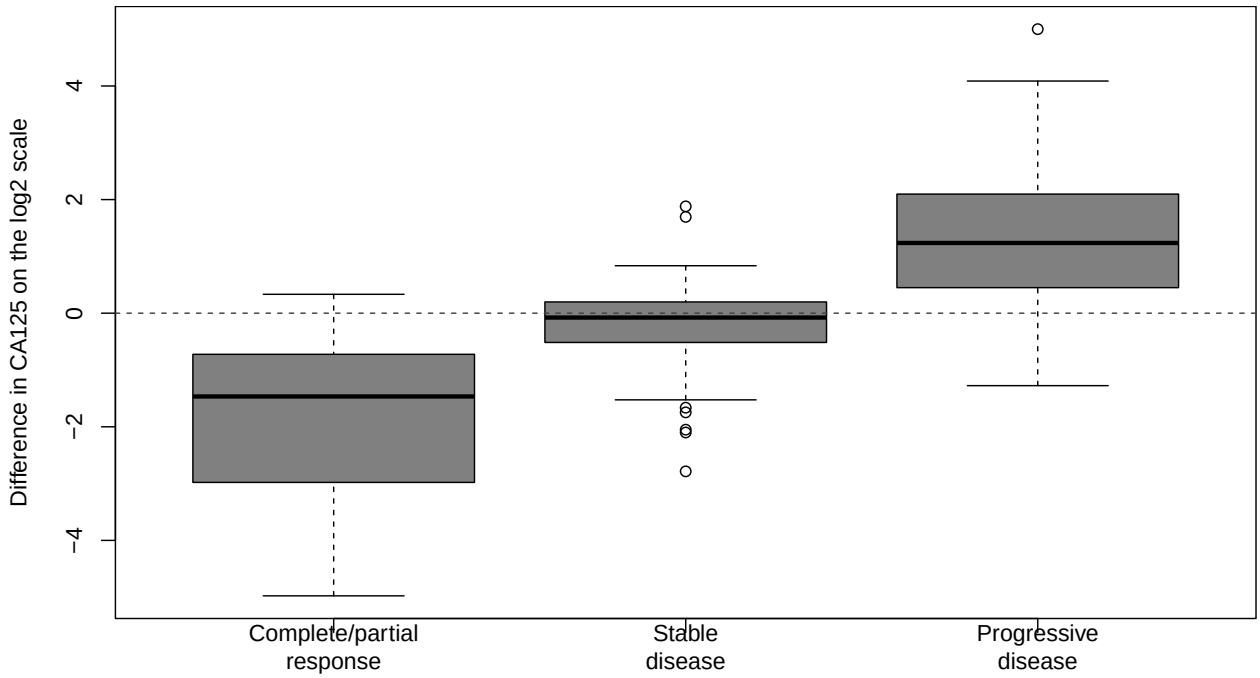


12.3 Figure S5B

Here, we (i) transform the 4-level CT scan assessment into a 3-level ordinal factor with "Complete/partial response", "Stable disease","Progressive disease" as level, (ii) display the CA125 measures on the scale of interest for each CT scan assessment level, (iii) report the p-value of the corresponding proportional odds model:

```
## ordinal response
figure_s5as5b$ct.end = ordered(c(1,1,2,3)[figure_s5as5b[,"ct.end"]],
                               levels = 1:3,
                               labels=c("Complete/partial\nresponse","Stable\ndisease",
                                        "Progressive\ndisease"
                                        )
                               )
## list of CA125 measure
par(mfcol=c(1,1),mar=c(3,4,0,0))
boxplot(TDL~ct.end,data=figure_s5as5b,col=gray(.5),
```

```
        ylab="Difference in CA125 on the log2 scale")
abline(h=0,col=gray(0.25),lty=2)
```




```
#
pval = coef(summary(ordinal::clm(ct.end~TDL,data=figure_s5as5b)))["TDL","Pr(>|z|)"]

id.measure = colnames(figure_s5as5b)[-(1:2)]
n.measure  = length(id.measure)
```

13 Figures S5C and S5D

We present here the code allowing to obtain AUC (area under the curve) and IC50 estimate for each drug and sample of interest.

13.1 Input

These analyses are based on the input files 'input/figure_s5c.csv' and 'input/figure_s5d.csv'.

The file 'input/figure_s5c.csv' reports, for each well corresponding to the spheroids of patient 364 (rows), the following information (columns):

- **drug**: the name of both drugs of interest,
- **concentration.level**: the drug level as a factor, from 0 for control wells to 8,
- **concentration**: the drug concentration,
- **response**: the drug response.

The following Table shows a random subset of the data.

```
# import:
set.seed(123)
figure_s5c = read.csv("input/figure_s5c.csv")
figure_s5c$concentration.level = factor(figure_s5c$concentration.level)
kable(figure_s5c[sample(1:nrow(figure_s5c),6),],format="simple")
```

	drug	concentration.level	concentration	response
31	Oxaliplatin	Dose-5	9.5	164640
79	Doxorubicin	Dose-8	30.0	13147
51	Doxorubicin	Dose-0	0.0	244500
14	Oxaliplatin	Dose-0	0.0	257481
67	Doxorubicin	Dose-4	0.3	154551
42	Doxorubicin	Dose-0	0.0	203636

The file 'input/figure_s5d.csv' reports, for each sample of each patient (rows), the following information (columns):

- **sample**: the sample ID (the first 3 digits correspond to the patient. Patient 409 has two samples),
- **auc.DDD**-columns: the AUC drug response estimates (as defined in the previous Section) for drug DDD (where DDD denotes one of the drug of interest),
- **ic50.DDD**-columns: the IC50 drug response estimates (as defined in the previous Section) for drug DDD (where DDD denotes one of the drug of interest). Missing values correspond to cases in which a standardised drug response of 0.5 was not achieved.

The following Table shows a subset of the data.

```
# import:
figure_s5d = read.csv("input/figure_s5d.csv")
kable(figure_s5d[1:6,c(1:3,14:15)],format="simple")
```

sample	auc.AZD8186	auc.AZD2014	ic50.AZD8186	ic50.AZD2014
156-s1	0.9374681	0.7228402	NA	-5.159440
248-s1	0.9649787	0.9132249	NA	NA
294-s1	0.7572422	0.5757075	NA	-7.235251
333-s1	0.9699401	0.6004840	NA	-6.739911

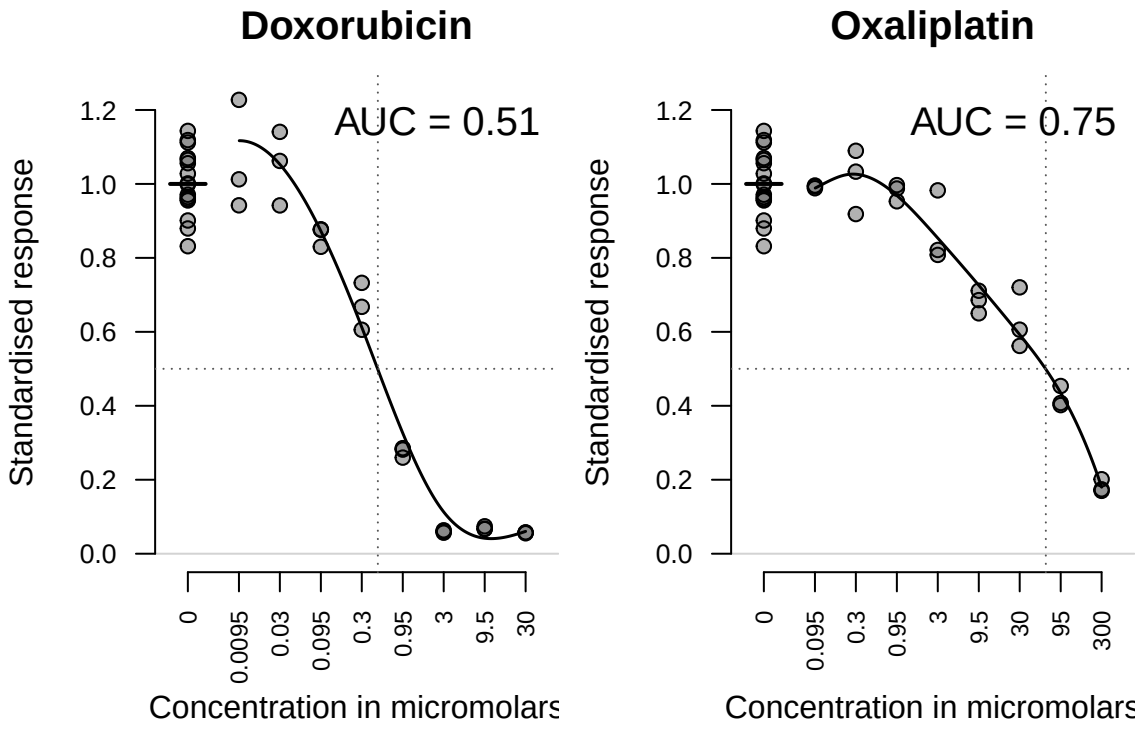
sample	auc.AZD8186	auc.AZD2014	ic50.AZD8186	ic50.AZD2014
338-s1	0.8176349	0.6807075	NA	-5.271731
364-s1	0.7232977	0.4932141	-4.982083	-8.760100

13.2 Figure S5C

This code allows to reproduce the estimation of the AUC and IC50 for 2 drugs for the spheroids of patient 364. The relationship between drug response and drug concentration (on the log scale) is modelled by means of M-splines on points not found to be outliers when fitting the data by means of a 4th order polynomial robust regression. I-splines are then used to estimate AUC over the concentration range of interest.

```
#
id.labdrug = names(table(figure_s5c$drug))
n.labdrug  = length(id.labdrug)
par(mfrow=c(1,n.labdrug),mar=c(4.5,4.5,2.5,0),omi=c(0,0,0,0))
for(dw in 1:n.labdrug){# dw=2
    # data
    data = figure_s5c[figure_s5c$drug == id.labdrug[dw],]
    data$pos = 1:nrow(data)
    # standardise by median response in control group
    median.ctrl = median(data$response[data$concentration.level=="Dose-0"])
    data$stand.response = data$response/median.ctrl
    data$logconc = log(data$concentration/(1000*10^(dw-1)))
    data$logconc[is.infinite(data$logconc)] = -13 # fictive position of control
    # diplay points
    xlim1 = range(data$logconc)
    xlim2 = range(data$logconc)+c(-.5,.5)
    ylimw = c(0,1.25)#max(data$value,na.rm=TRUE))
    plot(data$logconc,data$stand.response,
         xlim=xlim2,ylim=ylimw,
         pch = 19,col=paste0(gray(0.5),"99"),
         axes=FALSE,xlab="Concentration in micromolars",
         ylab="Standardised response",
         main=id.labdrug[dw],cex.main=1.25)
    axis(1,at=unique(data$logconc),
          tapply(data$concentration,data$concentration.level,function(x)x[1]),
          las=2,cex.axis=.75)
    abline(h=0,col="light gray")
    axis(2,las=2,cex.axis=.8)
    # 4rth order polynomial robust regression
    newdata = data.frame(stand.response=NA,logconc=seq(xlim1[1],xlim1[2],
                         length=1000))
    fit   = lmrob(I(stand.response-1)~logconc+I(logconc^2)+
                  I(logconc^3)+I(logconc^4)-1,
                  data=rbind(data,data[data$concentration==max(data$concentration),]))
    data$keep = (fit$rweights>0.4)[1:nrow(data)]
    points(data$logconc,data$stand.response,pch=1)
    # splines
    Alpha    = c(seq(xlim1[1],xlim1[2],length=5))
    mx.f.xu = survivalMPL:::basis_mpl(data$logconc[data$keep],
                         knots=list(Alpha=Alpha),
                         basis="msplines",
```

```
                        order=4,which=1)
m        = ncol(mx.f.xu)
theta    = abs(optim(rep(1,m),function(theta,psi,true){
                sum(abs(psi%*%abs(theta)-true)^2)},
                psi=mx.f.xu,
                true=data$stand.response[data$keep],
                control=list(maxit=100000),method="BFGS")$par)
mx.f.xu  = survivalMPL:::basis_mpl(newdata$logconc,
                        knots=list(Alpha=Alpha),
                        basis="msplines",
                        order=4,which=1)
f.x  = mx.f.xu%*%theta
posw = newdata$logconc>min(data$logconc[data$concentration>0])
lines(newdata$logconc[posw],f.x[posw],col=1,lwd=1.5)
# ic50
abline(h=0.5,col=gray(1/3),lty=3)
if(any(f.x<0.5)){
   posw = which(f.x<0.5&
                newdata$logconc>min(data$logconc[data$concentration==0]))
   if(length(posw)>0){
       IC50 = newdata$logconc[posw[1]]
       abline(v=IC50,col=gray(1/3),lty=3)
       }
}
# control mean
medw = median(data$stand.response[data$concentration==0])
segments(-13.5,medw,-12.5,medw,col=1,lwd=2)
# auc via isplines
rangew   = range(data$logconc[data$concentration>0])
mx.F.xu  = survivalMPL:::basis_mpl(rangew,
                        knots=list(Alpha=Alpha),
                        basis="msplines",
                        order=4,which=2)
int.lim = mx.F.xu%*%theta
AUC = (int.lim[2]-int.lim[1])/(rangew[2]-rangew[1])
legend("topright",legend=.p("AUC = ",round(AUC,2)),
       text.col=1,box.lwd=NA,cex=1.25)
}# end drug
```

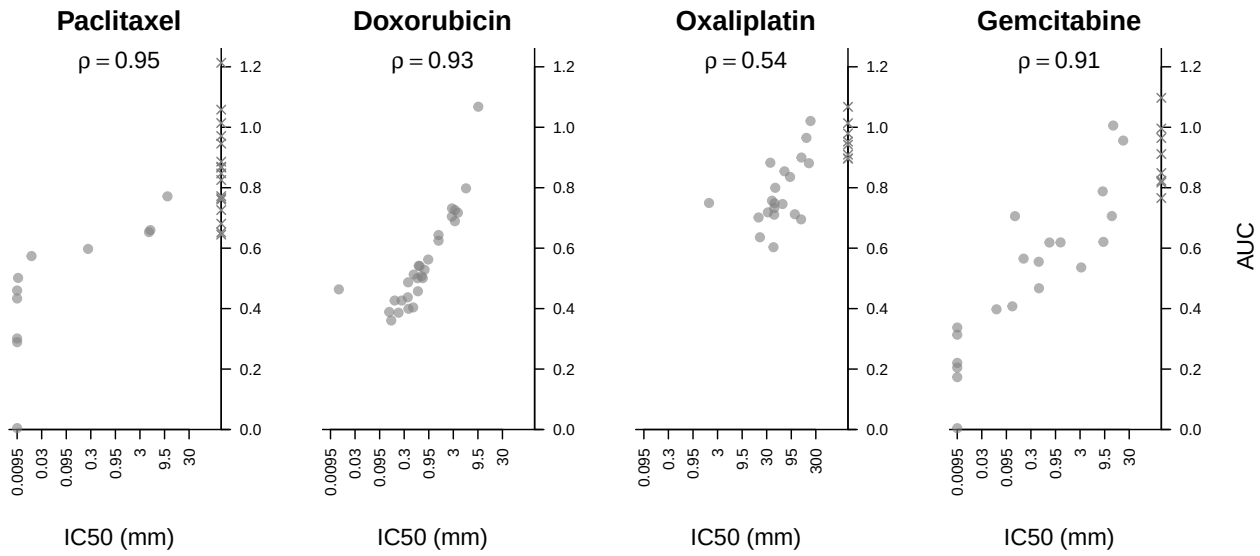


13.3 Figure S5D

Here, we display the relationship between the IC50 (x-axis) and AUC (y-axis) estimates, obtained as described in the previous section, for 4 drugs of interest (plots) and estimate the Spearman's correlation coefficient based on samples with non-missing IC50 estimates (The AUC estimates of samples with missing IC50 information appear on the y-axes by means of crosses).

```
## select drug of interest
id.drug = c("Paclitaxel","Doxorubicin",
            "Oxaliplatin","Gemcitabine")
n.drug  = length(id.drug)
id.dose = rep(c(0.0095,0.0300),4)*10^rep(0:3,each=2)
n.dose  = length(id.dose)
## plot and correlation estimates
par(mfrow=c(1,n.drug),mar=c(6,1,3,4),omi=c(0,0,0,.1))
ylimw = c(0,range(figure_s5d[,paste0("auc.",id.drug)],na.rm=TRUE)[2])
xlimw = c(min(figure_s5d[,paste0("ic50.",id.drug)],na.rm=TRUE),-2)
for(dw in 1:n.drug){# dw=1
    aucw  = .p("auc.",id.drug[dw])
    ic50w = .p("ic50.",id.drug[dw])
    plot(1,1,pch="",main=id.drug[dw],cex.main=1.75,
         xlim=xlimw,ylim=ylimw,axes=FALSE,xlab="",ylab="")
    y = id.file[,aucw]
    x = id.file[,ic50w]
    axis(1,at=log(id.dose/1000),
          id.dose*10^(id.drug[dw]=="Oxaliplatin"),
          las=2,pos=0)
    axis(1,at=mean(xlimw),
         "IC50 (mm)",tick=FALSE,padj=4,cex.axis=1.5)
    axis(4,pos=xlimw[2],las=2,seq(0,1.2,.2))
```

```
    if(dw==n.drug){
        axis(4,at=mean(ylimw),
             "AUC",tick=FALSE,padj=3,cex.axis=1.5)
        }
    abline(h=0)
    segments(xlimw[2],0,xlimw[2],100,col="black")
    points(id.file[,ic50w],id.file[,aucw],
           pch=19,col=paste0(gray(.5),99),cex=1.25)
    points(rep(xlimw[2],sum(is.na(x))),y[is.na(x)],
           pch=4,col=gray(.5),cex=1.25)
    #
    rho = round(cor(id.file[,ic50w],id.file[,aucw],use="pairwise.complete",method="spearman"),2)
    text(mean(xlimw),ylimw[2],labels=bquote(rho == .(rho)),
         col=1,cex=1.5)
    }
```



14 Figures S5E, S5F and S5G

We present here the code allowing to reproduce the analyses presented in Figures S5E, S5F and S5G

14.1 Input

These analyses are based on the input file 'input/figure_s5es5fs5g.csv'.

The file 'input/figure_s5es5fs5g.csv' reports, for each chemotherapy line of each patient (rows), the following information (columns):

- **patient**: the patient ID,
- **line**: the chemotherapy line (1 for the first, 2 for the second, aso.)
- **TDL**: the change in CA125 value (refer to Section 12),
- **TDL_surgery**: the change in CA125 value considering
  - either the changes in CA125 from one month post-surgery,
  - or the pre-surgery changes in CA125, whichever the longest (number of days),
- **auc**: the drug response (refer to Section 13),
- **week**: the length of the chemotherapy treatment (in weeks),
- **week_surgery**: the length of the chemotherapy treatment (in weeks) over the period considered to define **TDL_surgery**,
- **drug1, drug2, drug3, drugcomb.name, drugcomb.name2**: different ways of describing the chemotherapy treatment

The following Table shows the subset of the data corresponding to patient 333 who had 5 chemotherapy lines.

```
# import:
figure_s5es5fs5g = read.csv("input/figure_s5es5fs5g.csv")
figure_s5es5fs5g$patient = as.factor(figure_s5es5fs5g$patient)
figure_s5es5fs5g$drugcomb.name2 = as.factor(figure_s5es5fs5g$drugcomb.name2)
kable(figure_s5es5fs5g[6:9,c(1:7,11)],format="simple")
```

	patient	line	TDL	TDL_surgery	auc	week	week_surgery	drugcomb.name
6	333	1	-4.520990	-2.367796	0.6315837	19	10	CARBOPLATIN+PACLITAXEL
7	333	4	-1.102247	-1.102247	0.5176181	20	20	CARBOPLATIN+DOXORUBICIN
8	333	5	-3.059320	-3.059320	0.6448216	15	15	PACLITAXEL
9	333	6	0.165620	0.165620	0.5436191	17	17	CARBOPLATIN+GEMCITABINE

14.2 Figure S5E

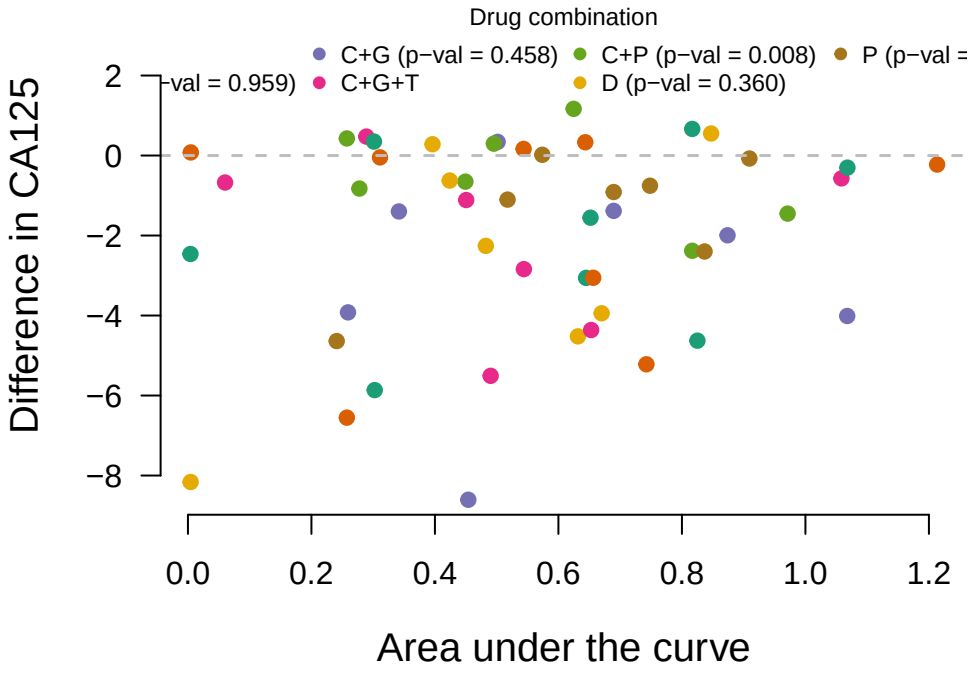
Here we analyse the relationship between in vivo (difference in CA125) and in vitro (AUC) drug responses per drug combinations by means of

- linear mixed models with patients as random intercepts in case of (potential) within-patient dependence,
- linear models in case of absence of within-patient dependence,

and in vivo and in vitro drug responses as outcome and predictor respectively. We then report the p-value corresponding to the test investigating if the AUC fixed effect parameter is different from 0.

```
#
x = figure_s5es5fs5g$auc
y = figure_s5es5fs5g[,"TDL"]
```

```
# legend (with p-value)
legendw = levels(figure_s5es5fs5g$drugcomb.name2)
for(lw in 1:length(legendw)){# lw=1
    dataw = figure_s5es5fs5g[figure_s5es5fs5g$drugcomb.name2==legendw[lw],]
    dataw$patient = droplevels(dataw$patient)
    # independence
    if(all(table(dataw$patient)==1)){
        fitw = try(lm(TDL~auc,data=dataw))
        if(class(fitw)!="try-error"){
            if(all(!is.na(coef(fitw)))){
                pval = coef(summary(fitw))["auc","Pr(>|t|)"]
                if(!is.na(pval)){
                legendw[lw] = .p(legendw[lw]," (p-val = ",.pval(pval,digit=3),")")
                }
            }
            }
    # dependence
    }else{
        fitw = try(lme(TDL~auc,random=~1|patient,data=dataw))
        if(class(fitw)!="try-error"){
            pval = coef(summary(fitw))["auc","p-value"]
            legendw[lw] = .p(legendw[lw]," (p-val = ",.pval(pval,digit=3),")")
            }
    }
    }
# plot
colw = brewer.pal(nlevels(figure_s5es5fs5g$drugcomb.name2),"Dark2")
par(mfrow=c(1,1),mar=c(4,4,0,0))
plot(x,y,col=colw,main="",ylim=c(-8.5,3.5),axes=FALSE,
     xlab="Area under the curve",#\n(low value means high sensitivity to drugs)",
     ylab="Difference in CA125",pch=19,cex.lab=1.25)
abline(h=0,col="gray",lty=2,lwd=1.5)
axis(1)
axis(2,las=2)
#
legend("top",ncol=4,box.lwd=NA,
       col=colw,pch=19,
       legend=legendw,
       title="Drug combination",cex=.75)
```



14.3 Figure S5F

Here we analyse the relationship between in vivo (difference in CA125) and in vitro (AUC) drug responses per chemotherapy line as a 3-level factor ('1st line', '2nd line' and '3 lines and more') by means of linear mixed models with

- patients as random intercepts (to take the potential within-patient dependence into account),
- in vivo and in vitro drug responses as outcome and predictor respectively,
- the residual variance being a function of the chemotherapy duration (in weeks).

We then report the multiplicity adjusted p-values corresponding to different contrasts of interest.

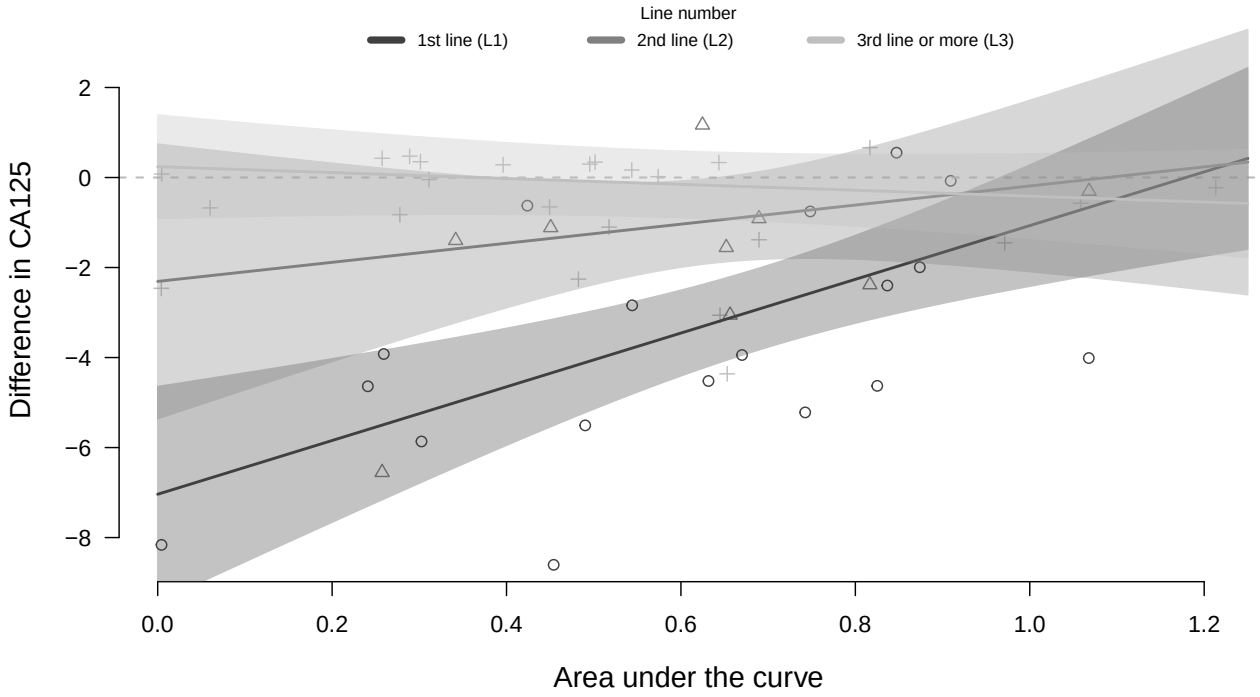
```
## line as a 3-level factor
figure_s5es5fs5g$line.factor = factor(
    c(1,2,rep(3,100))[figure_s5es5fs5g$line],
    levels=1:3,
    labels=c("1st line","2nd line",">2nd line"))
## heteroscedastic random intercept model
fit.lme = lme(TDL~auc*line.factor,
          random=~1|patient,
          data=figure_s5es5fs5g,
          weights = varPower(form = ~ week),
          method="REML"
          )
## contrasts
K = rbind(l1_0  = c( 0, 1, 0, 0, 0, 0),
          l2_0  = c( 0, 1, 0, 0, 1, 0),
          l3_0  = c( 0, 1, 0, 0, 0, 1),
          l1_l2 = c( 0, 0, 0, 0, 1, 0),
          l1_l3 = c( 0, 0, 0, 0, 0, 1),
          l2_l3 = c( 0, 0, 0, 0, 1,-1)
          )
summary(glht(fit.lme,linfct=K))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lme.formula(fixed = TDL ~ auc * line.factor, data = figure_s5es5fs5g,
##     random = ~1 | patient, weights = varPower(form = ~week),
##     method = "REML")
##
## Linear Hypotheses:
##             Estimate Std. Error z value Pr(>|z|)
## l1_0 == 0    5.9686     1.6377   3.644  0.00138 **
## l2_0 == 0    2.1203     2.3245   0.912  0.77483
## l3_0 == 0   -0.6544     0.7468  -0.876  0.79518
## l1_l2 == 0  -3.8483     2.9302  -1.313  0.51835
## l1_l3 == 0  -6.6230     1.7044  -3.886  < 0.001 ***
## l2_l3 == 0   2.7747     2.3594   1.176  0.60863
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

We finally display the fit per chemotherapy line.

```
colw = gray(c(.25,.5,.75))
par(mfrow=c(1,1),mar=c(4,4,0,0))
x = figure_s5es5fs5g$auc
y = figure_s5es5fs5g$TDL
plot(x,y,col=colw[as.numeric(figure_s5es5fs5g$line.factor)],
     main="",ylim=c(-8.5,3.5),axes=FALSE,
     xlab="Area under the curve",
     ylab="Difference in CA125",
     pch=as.numeric(figure_s5es5fs5g$line.factor),
     cex.lab=1.25)
abline(h=0,col="gray",lty=2,lwd=1.5)
axis(1)
axis(2,las=2)
#
legend("top",ncol=3,box.lwd=NA,
       col=colw,lty=1,lwd=5,
       legend=.p(c("1st line (L1)","2nd line (L2)","3rd line or more (L3)")),
       title="Line number",cex=.75)
# estimates and se
axe.x  = seq(0,1.25,length=1000)
newdat = data.frame(auc=rep(axe.x,nlevels(figure_s5es5fs5g$line.factor)),
                    line.factor=factor(rep(levels(figure_s5es5fs5g$line.factor),each=length(axe.x)),
                                       levels=levels(figure_s5es5fs5g$line.factor)),
                    week = 0
                    )
newdat$pred   = predict(fit.lme, newdat, level = 0)
design.matrix = model.matrix(eval(eval(fit.lme$call$fixed)[-2]), newdat)
predvar   <- diag(design.matrix %*% fit.lme$varFix %*% t(design.matrix))
newdat$SE <- sqrt(predvar)
for(lw in 1:nlevels(figure_s5es5fs5g$line.factor)){# lw=1
    posw = .an(newdat$line.factor) == lw
    auc  = newdat$auc[posw]
    mid  = newdat$pred[posw]
    low  = mid-qnorm(.975)*newdat$SE[posw]
```

```
    high = mid+qnorm(.975)*newdat$SE[posw]
    lines(auc,mid,col=colw[lw],lwd=2)
    x = c(auc,auc[length(auc):1])
    y = c(high,low[length(auc):1])
    polygon(x,y,col=.p(colw[lw],50),border = .p(colw[lw],50))

    }
```



14.4 Figure S5G

Here, we repeat the analysis presented in Figure S5F to the data ignoring CA125 changes around surgery time.

```
## heteroscedastic random intercept model
fit.lme = lme(TDL_surgery~auc*line.factor,
          random=~1|patient,
          data=figure_s5es5fs5g,
          weights = varPower(form = ~ week_surgery),
          method="REML"
          )
## contrasts
K = rbind(l1_0  = c( 0, 1, 0, 0, 0, 0),
          l2_0  = c( 0, 1, 0, 0, 1, 0),
          l3_0  = c( 0, 1, 0, 0, 0, 1),
          l1_l2 = c( 0, 0, 0, 0, 1, 0),
          l1_l3 = c( 0, 0, 0, 0, 0, 1),
          l2_l3 = c( 0, 0, 0, 0, 1,-1)
          )
summary(glht(fit.lme,linfct=K))
```

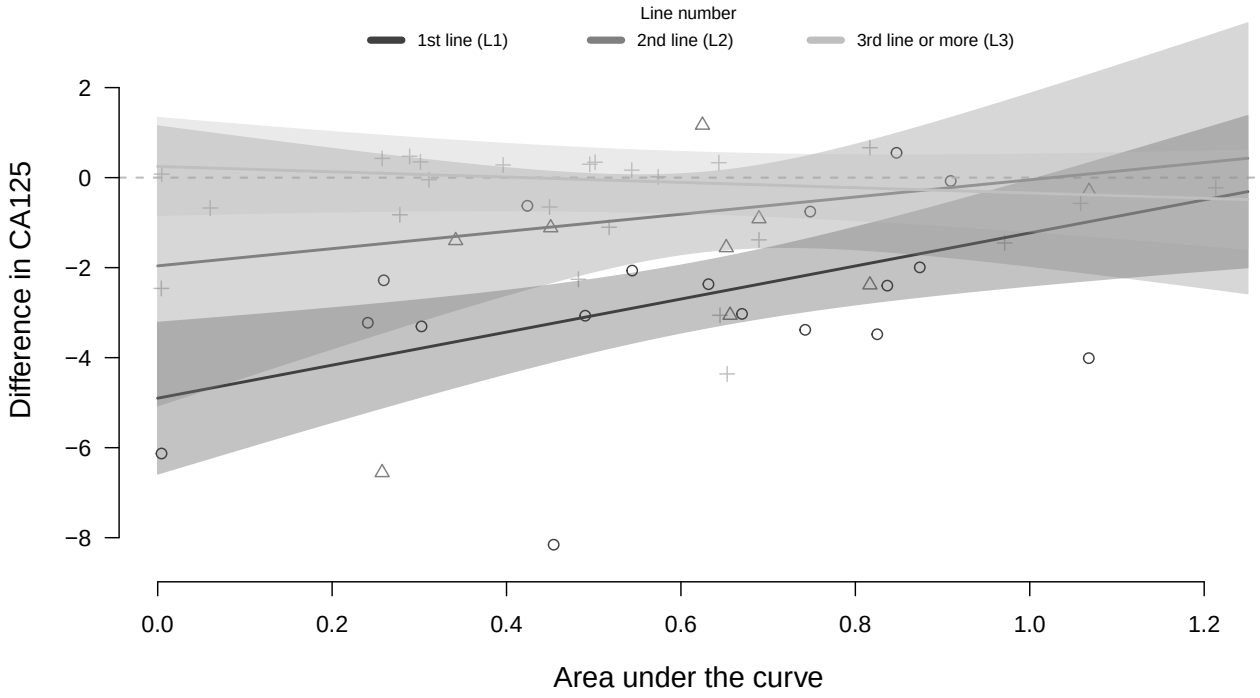
```
##
```

```
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lme.formula(fixed = TDL_surgery ~ auc * line.factor, data = figure_s5es5fs5g,
##     random = ~1 | patient, weights = varPower(form = ~week_surgery),
##     method = "REML")
##
## Linear Hypotheses:
##            Estimate Std. Error z value Pr(>|z|)
## l1_0 == 0    3.6719     1.2316   2.982  0.01210 *
## l2_0 == 0    1.9121     2.3901   0.800  0.83510
## l3_0 == 0   -0.5900     0.6976  -0.846  0.81100
## l1_l2 == 0  -1.7598     2.8049  -0.627  0.91203
## l1_l3 == 0  -4.2619     1.3075  -3.260  0.00498 **
## l2_l3 == 0   2.5021     2.4271   1.031  0.70130
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

We finally display the fit per chemotherapy line.

```
colw = gray(c(.25,.5,.75))
par(mfrow=c(1,1),mar=c(4,4,0,0))
x = figure_s5es5fs5g$auc
y = figure_s5es5fs5g$TDL_surgery
plot(x,y,col=colw[as.numeric(figure_s5es5fs5g$line.factor)],
     main="",ylim=c(-8.5,3.5),axes=FALSE,
     xlab="Area under the curve",
     ylab="Difference in CA125",
     pch=as.numeric(figure_s5es5fs5g$line.factor),
     cex.lab=1.25)
abline(h=0,col="gray",lty=2,lwd=1.5)
axis(1)
axis(2,las=2)
#
legend("top",ncol=3,box.lwd=NA,
       col=colw,lty=1,lwd=5,
       legend=.p(c("1st line (L1)","2nd line (L2)","3rd line or more (L3)")),
       title="Line number",cex=.75)
# estimates and se
axe.x  = seq(0,1.25,length=1000)
newdat = data.frame(auc=rep(axe.x,nlevels(figure_s5es5fs5g$line.factor)),
                    line.factor=factor(rep(levels(figure_s5es5fs5g$line.factor),each=length(axe.x)),
                                       levels=levels(figure_s5es5fs5g$line.factor)),
                    week = 0
                    )
newdat$pred   = predict(fit.lme, newdat, level = 0)
design.matrix = model.matrix(eval(eval(fit.lme$call$fixed)[-2]), newdat)
predvar   <- diag(design.matrix %*% fit.lme$varFix %*% t(design.matrix))
newdat$SE <- sqrt(predvar)
for(lw in 1:nlevels(figure_s5es5fs5g$line.factor)){# lw=1
    posw = .an(newdat$line.factor) == lw
    auc  = newdat$auc[posw]
    mid  = newdat$pred[posw]
    low  = mid-qnorm(.975)*newdat$SE[posw]
    high = mid+qnorm(.975)*newdat$SE[posw]
```

```
    lines(auc,mid,col=colw[lw],lwd=2)
    x = c(auc,auc[length(auc):1])
    y = c(high,low[length(auc):1])
    polygon(x,y,col=.p(colw[lw],50),border = .p(colw[lw],50))

    }
```



15 Figures S6A and S6B

We present here the code allowing to reproduce the analyses presented in Figures S6A and S6B

15.1 Input

These analyses are based on the input file 'input/figure_2b2d2hs4cs4f.csv' and 'input/figure_s6as6b.csv'.

The content of file was already described previsously. The file 'input/figure_s6as6b.csv' reports, for each patient (columns), the ACN of all genes (rows):

```
# import:
figure_s6as6b = read.csv("input/figure_s6as6b.csv",row.names=1)
colnames(figure_s6as6b) = substr(colnames(figure_s6as6b),2,4)
kable(figure_s6as6b[1:5,1:6],format="simple")
```

	294	333	364	409	413	466
A1BG	0.1426132	-1.1986800	0.2852278	-0.3220469	0.1493322	-0.0323971
NAT2	-0.0569707	-0.2838431	-0.2175449	-1.4081209	-1.1951130	-0.2853170
ADA	1.0664460	0.5567405	1.0930217	0.4333143	0.3552261	0.4564536
CDH2	-0.0675629	-0.2013319	-0.3661076	0.0713317	-0.7081651	-0.3509781
AKT3	-0.0126196	-0.2430614	0.6033702	0.5427205	0.0951364	0.2412423

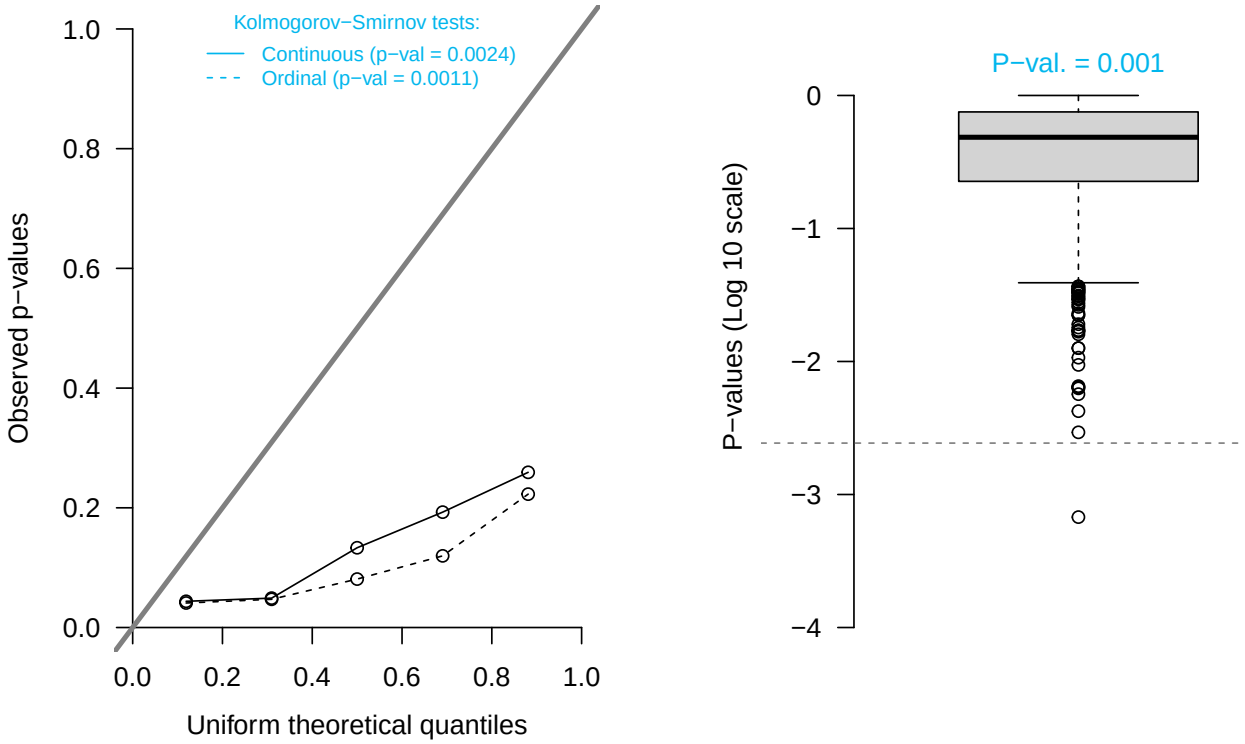
15.2 Figures S6A

- left plot: we reproduce the p-values of Figures 2B, 2D, 2H, S4C and S4F (see above) and compare their empirical distribution function with the uniform probability distribution by means of Kolmogorov–Smirnov tests,
- right plot: we repeat this exercise for 1000 sets of 5 randomly selected genes (excluding the ones with missing data or part of the primary list of interest) and display the 1000 p-values (on the log10 scale) of the Kolmogorov–Smirnov tests obatined when considering gene copy-number as a continuous predictor by means of a boxplot. The horizontal dashed line corresponds to log10(0.0024), the result observed when considering the pre-specified list of genes.

```
##
## relevant elements of Figures 2B, 2D, 2H, S4C and S4F
##
n.assoc  = 5
id.assoc = data.frame(pos    = 1:n.assoc,
                      id     = c("2B","2D","2H","S4C","S4F"),
                      drug   = c("Paclitaxel","AZD0156","AZD2014","Doxorubicin","AZD2014"),
                      gene   = c("MYC","CCNE1","MYC","KRAS","TERT"))
id.assoc$alternative  = c("less")
id.assoc$alternative[id.assoc$gene=="CCNE1"]  = c("greater")
id.assoc$alternative2  = c("decreasing")
id.assoc$alternative2[id.assoc$gene=="CCNE1"]  = c("increasing")
id.assoc0 = id.assoc
#
figure_2b2d2hs4cs4f = read.csv("input/figure_2b2d2hs4cs4f.csv",row.names=1)
#
id.assoc$pval.cont = NA
id.assoc$pval.ord = NA
n.R = 1001
#
if(!any(dir("input")=="FS6A")){
    ar.pval.ra2 = array(dim=c(n.R,n.assoc,2),
                        dimnames=list(1:n.R,.p(id.assoc$id,"+",id.assoc$gene),
                            c("continuous","ordinal")))

    set.seed(1)
    for(rw in 1:n.R){# rw=1

        # randomly select genes:
        genew = rownames(figure_s6as6b)[is.na(match(rownames(figure_s6as6b),
            c("KRAS","MYC","PIK3CA","TERT","CCNE1")))&
            apply(is.na(figure_s6as6b),1,sum)==0]
        genew = genew[order(runif(length(genew)))][1:n.assoc]
        # position 2 to 1001 correspond to random assoc
        if(rw > 1){
            mx.rcn_random.sg = t(figure_s6as6b[genew,])
        # position 1 corresponds to real assoc
        }else{
            mx.rcn_random.sg = t(figure_s6as6b[c("KRAS","MYC","PIK3CA","TERT","CCNE1"),])
        }
        colnames(mx.rcn_random.sg) = c("KRAS","MYC","PIK3CA","TERT","CCNE1")

        ## pval
```

```
        for(aw in 1:n.assoc){# aw=1
            # data
            auc  = figure_2b2d2hs4cs4f[,id.assoc$drug[aw]]
            rcn  = mx.rcn_random.sg[,id.assoc$gene[aw]]
            acn  = unlist(round(2*2^rcn))
            acn3 = ordered(c(NA,2,3,rep(4,100))[acn],levels=c(2:4),labels=c("2","3","4+"))
            # continuous trend
            fit = lm(auc~rcn)
            glhtw = summary(glht(fit,linfct=matrix(c(0,1),ncol=2),
                alternative=id.assoc$alternative[aw]))[[9]]$pvalues
            ar.pval.ra2[rw,aw,1] = glhtw
            # trend test
            if(rw==1){
               testw = try(jonckheere.test(auc,acn3,
                   alternative=id.assoc$alternative2[aw]))
               if(class(testw)!="try-error"){
                    if(!is.na(testw$p.value)){
                        ar.pval.ra2[rw,aw,2] = testw$p.value
                    }else{
                        # reverse order for altrenative to be meaningful:
                        acn3 = c(2:1)[as.numeric(droplevels(acn3))]
                        ar.pval.ra2[rw,aw,2] = t.test(auc~acn3,
                            alternative=id.assoc$alternative[aw])$p.value
                    }
                    }
                }
           }# end dw
        .cat(rw,n.R)
        }# end rw
    save(ar.pval.ra2,file=.p("input/FS6A"))
}else{
    load(.p("input/FS6A"))
}
#
par(mfrow=c(1,2),mar=c(4,5,1,0))
##
## left plot
##
id.assoc$pval.cont = ar.pval.ra2[1,,1]
id.assoc$pval.ord  = ar.pval.ra2[1,,2]
plot(1,1,pch="",xlim=c(0,1),ylim=c(0,1),xlab="",ylab="",axes=FALSE)
axis(1,pos=0)
axis(1,at=0.5,tick=FALSE,pad=1.25,"Uniform theoretical quantiles")
axis(2,pos=0,las=2)
axis(2,at=0.5,tick=FALSE,pad=-2.5,"Observed p-values")
#
abline(0,1,col=gray(0.5),lwd=3)
#
pvalw = id.assoc$pval.cont[order(id.assoc$pval.cont)]
points(ppoints(pvalw),pvalw,col=1)
lines(ppoints(pvalw),pvalw,col=1,lty=1)
pvalw = id.assoc$pval.ord[order(id.assoc$pval.ord)]
points(ppoints(pvalw),pvalw,col=1)
```

```
lines(ppoints(pvalw),pvalw,col=1,lty=2)
#
legend("top",
       title="Kolmogorov-Smirnov tests:",
       legend = c(.p("Continuous (p-val = ",
                     .pval(ks.test(id.assoc$pval.cont,"punif")$p.value),")"),
                  .p("Ordinal (p-val = ",
                     .pval(ks.test(id.assoc$pval.ord,"punif")$p.value),")")),
       col="#00B6ED", lty=1:2, cex=.75,text.col="#00B6ED",box.lwd=NA)
##
## right plot
##
tmp = apply(ar.pval.ra2[-1,,1],1,function(x){
    ks.test(x,"punif")$p.value})
boxplot(log10(tmp),axes=FALSE,
        xlim=c(.5,1.25),ylim=c(-4,0.5))#c(min(log10(tmp)),0.5))
axis(2,seq(-4,0,1),las=2,pos=0.625)
axis(2,at=mean(range(log10(tmp))),"P-values (Log 10 scale)",tick=FALSE,padj=1)
abline(h=log10(ks.test(id.assoc$pval.cont,"punif")$p.value),col=gray(.5),lty=2)
text(1,0.25,.p("P-val. = ",mean(tmp<ks.test(id.assoc$pval.cont,"punif")$p.value)),col="#00B6ED")
```



If there was no relationship between drug response and copy number, the p-values would be close to the unit line (dark blue) on the left plot, and the dotted horizontal line would not correspond to an outlier on the right plot (only one random gene copy number - drug association over a 1000 had a smaller Kolmogorov-Smirnov p-value). We can note that this is far from being the case, thus showing a strong support for our hypothesis that copy number changes can predict drug response.

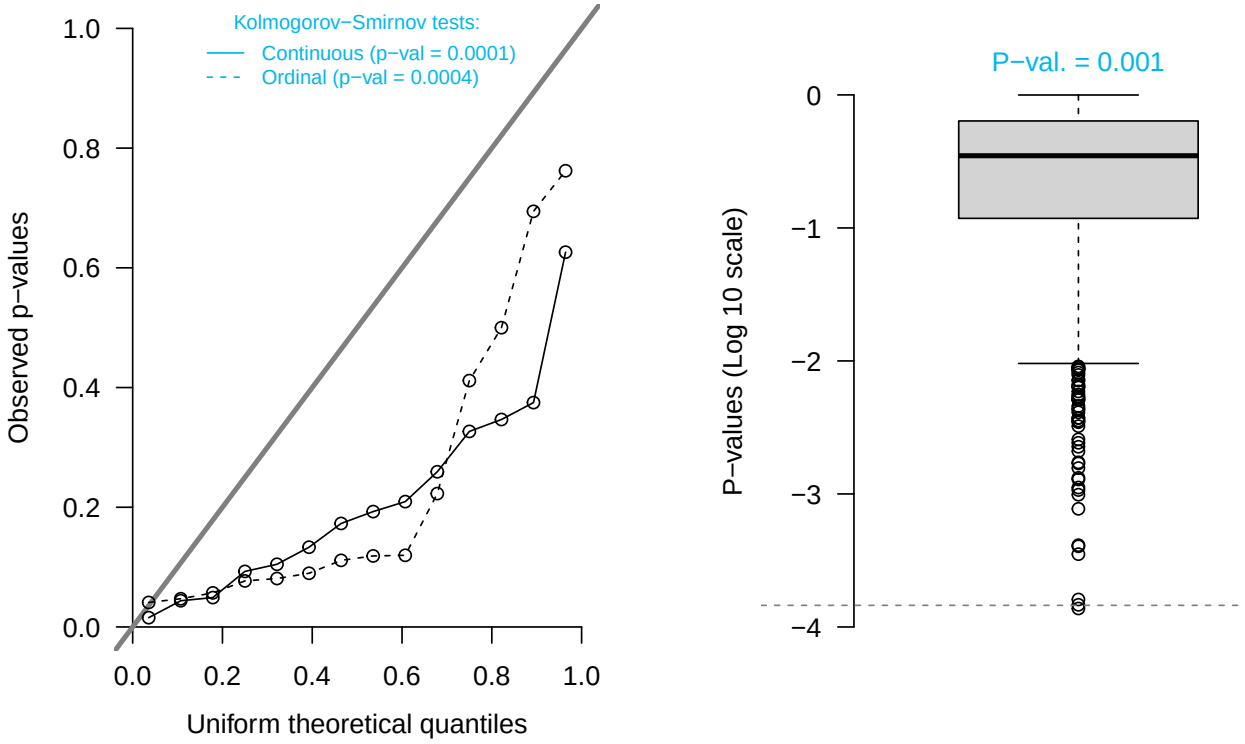
15.3 Figures S6B

We repeat here the analysis performed in the previous section when considering an extended list of associations (5 original ones + 9 additional ones).

```
##
## relevant elements of Figures 2B, 2D, 2H, S4C and S4F
##
n.assoc  = 14
id.assoc = data.frame(pos    = 1:n.assoc,
                      id     = 1:n.assoc,
                      drug   = c('Paclitaxel','AZD2014','AZD2014','AZD2014',
                      'Oxaliplatin','AZD5363','AZD5363','AZD5363','AZD5363',
                      'AZD8835','AZD8835','AZD2281','AZD0156','Doxorubicin'),
                      gene   = c('MYC','MYC','TERT','PIK3CA','CCNE1',
                      'MYC','TERT','KRAS','PIK3CA','TERT','PIK3CA',
                      'CCNE1','CCNE1','KRAS'))
id.assoc$alternative  = c("less")
id.assoc$alternative[id.assoc$gene=="CCNE1"]  = c("greater")
id.assoc$alternative2  = c("decreasing")
id.assoc$alternative2[id.assoc$gene=="CCNE1"]  = c("increasing")
#
figure_2b2d2hs4cs4f = read.csv("input/figure_2b2d2hs4cs4f.csv",row.names=1)
#
id.assoc$pval.cont = NA
id.assoc$pval.ord = NA
n.R = 1001
#
if(!any(dir("input")=="FS6B")){
    ar.pval.ra2 = array(dim=c(n.R,n.assoc,2),
                        dimnames=list(1:n.R,.p(id.assoc$id,"+",id.assoc$gene),
                            c("continuous","ordinal")))
    set.seed(1)
    for(rw in 1:n.R){# rw=1

        # keep 5 randomly select genes for the 14 random associations
        # to keep the dependence between these associations as in the original set
        genew = rownames(figure_s6as6b)[is.na(match(rownames(figure_s6as6b),
            c("KRAS","MYC","PIK3CA","TERT","CCNE1")))&
            apply(is.na(figure_s6as6b),1,sum)==0]
        genew = genew[order(runif(length(genew)))][1:5]
        # position 2 to 1001 correspond to random assoc
        if(rw > 1){
            mx.rcn_random.sg = t(figure_s6as6b[genew,])
        # position 1 corresponds to real assoc
        }else{
            mx.rcn_random.sg = t(figure_s6as6b[c("KRAS","MYC","PIK3CA","TERT","CCNE1"),])
        }
        colnames(mx.rcn_random.sg) = c("KRAS","MYC","PIK3CA","TERT","CCNE1")

        ## pval
        for(aw in 1:n.assoc){# aw=1
            # data
            auc  = figure_2b2d2hs4cs4f[,id.assoc$drug[aw]]
```

```
            rcn  = mx.rcn_random.sg[,id.assoc$gene[aw]]
            acn  = unlist(round(2*2^rcn))
            acn3 = ordered(c(NA,2,3,rep(4,100))[acn],levels=c(2:4),labels=c("2","3","4+"))
            # continuous trend
            fit = lm(auc~rcn)
            glhtw = summary(glht(fit,linfct=matrix(c(0,1),ncol=2),
                alternative=id.assoc$alternative[aw]))[[9]]$pvalues
            ar.pval.ra2[rw,aw,1] = glhtw
            # trend test
            if(rw==1){
               testw = try(jonckheere.test(auc,acn3,
                   alternative=id.assoc$alternative2[aw]))
               if(class(testw)!="try-error"){
                    if(!is.na(testw$p.value)){
                        ar.pval.ra2[rw,aw,2] = testw$p.value
                    }else{
                        # reverse order for altrenative to be meaningful:
                        acn3 = c(2:1)[as.numeric(droplevels(acn3))]
                        ar.pval.ra2[rw,aw,2] = t.test(auc~acn3,
                            alternative=id.assoc$alternative[aw])$p.value
                    }
                    }
                }
           }# end dw
        .cat(rw,n.R)
        }# end rw
    save(ar.pval.ra2,file=.p("input/FS6B"))
}else{
    load(.p("input/FS6B"))
}
#
par(mfrow=c(1,2),mar=c(4,5,1,0))
##
## left plot
##
id.assoc$pval.cont = ar.pval.ra2[1,,1]
id.assoc$pval.ord  = ar.pval.ra2[1,,2]
plot(1,1,pch="",xlim=c(0,1),ylim=c(0,1),xlab="",ylab="",axes=FALSE)
axis(1,pos=0)
axis(1,at=0.5,tick=FALSE,pad=1.25,"Uniform theoretical quantiles")
axis(2,pos=0,las=2)
axis(2,at=0.5,tick=FALSE,pad=-2.5,"Observed p-values")
#
abline(0,1,col=gray(0.5),lwd=3)
#
pvalw = id.assoc$pval.cont[order(id.assoc$pval.cont)]
points(ppoints(pvalw),pvalw,col=1)
lines(ppoints(pvalw),pvalw,col=1,lty=1)
pvalw = id.assoc$pval.ord[order(id.assoc$pval.ord)]
points(ppoints(pvalw),pvalw,col=1)
lines(ppoints(pvalw),pvalw,col=1,lty=2)
#
legend("top",
```

```
       title="Kolmogorov-Smirnov tests:",
       legend = c(.p("Continuous (p-val = ",
                     .pval(ks.test(id.assoc$pval.cont,"punif")$p.value),")"),
                  .p("Ordinal (p-val = ",
                     .pval(ks.test(id.assoc$pval.ord,"punif")$p.value),")")),
       col="#00B6ED", lty=1:2, cex=.75,text.col="#00B6ED",box.lwd=NA)
##
## right plot
##
tmp = apply(ar.pval.ra2[-1,,1],1,function(x){
    ks.test(x,"punif")$p.value})
boxplot(log10(tmp),axes=FALSE,
        xlim=c(.5,1.25),ylim=c(-4,0.5))#c(min(log10(tmp)),0.5))
axis(2,seq(-4,0,1),las=2,pos=0.625)
axis(2,at=mean(range(log10(tmp))),"P-values (Log 10 scale)",tick=FALSE,padj=1)
abline(h=log10(ks.test(id.assoc$pval.cont,"punif")$p.value),col=gray(.5),lty=2)
text(1,0.25,.p("P-val. = ",mean(tmp<ks.test(id.assoc$pval.cont,"punif")$p.value)),col="#00B6ED")
```



The same conclusions are obtained when considering the extended list of associations.

Finally, we report in the following table the p-values corresponding to the 14 associations of interest when considering CN as a continuous (pvalue.cont) or ordinal (pvalue.ord) predictor. Associations are ranked according to *pvalue.cont* in increasing order. *primary* indicates which associations were part of the primary list of pre-specified *gene CN - drug* associations of interest.

```
id.assoc$primary = !is.na(match(paste0(id.assoc$drug,id.assoc$gene,sep="+"),
                                paste0(id.assoc0$drug,id.assoc0$gene,sep="+")))
id.assoc = id.assoc[order(id.assoc$pval.cont),
                    c("drug","gene","alternative","pval.cont","pval.ord","primary")]
row.names(id.assoc) = NULL
```

```
kable(id.assoc,row.names=NA)
```

drug	gene	alternative	pval.cont	pval.ord	primary
Oxaliplatin	CCNE1	greater	0.0155460	0.0769197	FALSE
AZD0156	CCNE1	greater	0.0438077	0.1197166	TRUE
Paclitaxel	MYC	less	0.0490961	0.0471612	TRUE
AZD8835	TERT	less	0.0927974	0.1112791	FALSE
AZD5363	KRAS	less	0.1047561	0.1186667	FALSE
AZD2014	TERT	less	0.1332333	0.2228541	TRUE
AZD2281	CCNE1	greater	0.1729189	0.0897183	FALSE
Doxorubicin	KRAS	less	0.1927776	0.0409783	TRUE
AZD5363	TERT	less	0.2093984	0.4116230	FALSE
AZD2014	MYC	less	0.2592770	0.0809497	TRUE
AZD8835	PIK3CA	less	0.3266590	0.5000000	FALSE
AZD5363	PIK3CA	less	0.3465778	0.7621732	FALSE
AZD5363	MYC	less	0.3748809	0.0569516	FALSE
AZD2014	PIK3CA	less	0.6261027	0.6943627	FALSE