

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

**Florian Heigwer, German Cancer Research Center (DKFZ),
Heidelberg, Germany and f.heigwer @ dkfz.de**

2018-07-16

Abstract

Context-dependent changes in genetic vulnerabilities are important to understand the wiring of cellular pathways and variations in different environmental conditions. However, methodological frameworks to investigate the plasticity of genetic networks over time or in response to external stresses are lacking. To analyze the plasticity of genetic interactions, we performed an arrayed combinatorial RNAi screen in Drosophila cells at multiple time points and after pharmacological inhibition of Ras signaling activity. Using an image-based morphology assay to capture a broad range of phenotypes, we assessed the effect of 12768 pairwise RNAi perturbations in six different conditions. We found that genetic interactions form in different trajectories and developed an algorithm, termed MODIFI, to analyze how genetic interactions rewire over time. Using this framework, we identified more statistically significant interactions compared to end-points assays and further observed several examples of context-dependent crosstalk between signaling pathways such as an interaction between Ras and Rel which is dependent on MEK activity.

Contents

About	4
1 Figure 1: A genome wide chemo-genetic screen.	4
2 Data normalization	6
3 Normalization for positional biases	7
4 Calculation of distance and dsRNA wise quality metrics.	7
5 Figure 1 B: Plot correlations of screens	13
6 Figure 1 C: Plot correlations of Ras pathway genes	14
7 Figure 1 D&E: Plot heatmap of Ras pathway	16

8	Supplementary Figure S1B: Correlation of biological and chemical perturbation of Ras signaling	17
9	Supplementary Figure S1D-G: Compound characterization on Dmel2 cells	18
10	Supplementary Figure S2: Control separation capacity of our morphological assay	22
11	Next we analyzed our time resolved genetic interaction screen	24
12	Analyze feature correlation and filter out non-redundant features	30
13	Supplemental Figure 3A: Featurewise biological reproductability analysis	30
14	Supplemental Figure 3B: Pairwise feature correlation analysis and redundancy filter	32
15	Supplemental Figure 3C: Scatterplot of uncorrelating features after redundancy checks	33
16	Supplemental Figure 3D: Check if features enrich for non-redundant information content of correlated residuals	34
17	Figure 2C: Biological replicate correlation	36
18	Figure 2D: dsRNA design replicate correlation	36
19	Figure 2E: Biological control separation	37
20	Figure 2F: Example interaction heatmap	39
21	Supplementary Figure 5A-C: Feature comparison to Fischer et al. 2015	41
22	Supplementary Figure 5D-G: Feature comparison to Horn et al. 2011	44
23	Supplementary Figure 6: Quality control metrics of gene-gene combinatorial screening	49
24	Comparison of different statistical measures	54
25	Figure 3B: comparison of different statistical methods with regards to their power	57
26	Figure 3C: Differential interaction scores per pathway	59

27	Figure 4A-D: Example stable and differential positive and negative interactions	62
28	Figure 4E: Proof that the linear model does predict equal amounts of interactions for every feature independently	63
29	Figure 4F-F': Test if all model residuals can be explained by measurement variance.	65
30	Figure 4G-H: pathway specific stable and differential genetic interactions	67
31	Figure 4I-J: gene specific stable and differential genetic interactions	70
32	Figure 5A-D: gene specific stable and differential genetic interactions	72
33	Figure 6A: Small multiple networks	76
34	Figure 6B: Example correlations of differential interactions based on cell eccentricity	78
35	Figure 6C: A correlation network of differential genetic interactions	82
36	Supplementary Figure S7: A differential correlation analysis according to Billmann et al.	83
37	Supplementary Figure S8: dsRNA correlations	85
38	Figure 7C,C': Rel-pnt differential interactions	86
39	Figure 7D,E: Loss of Rel rescues loss of Ras signaling activity by Pvf2 upregulation	87
40	Supplementary Figure S10: Short term expression changes after Rel/Ras signaling co-perturbations	93
41	Session info	96

About

This vignette summarizes all source code needed to reproduce all data containing main and supplementary figures. The networks shown in figure 6 can be reproduced using the Cytoscape Session file in the supplements.

1 Figure 1: A genome wide chemo-genetic screen.

Data used in this analysis were per-well feature data extracted using the pipeline in “Feature_extraction_genome_wide_screens.Rmd” and the genome-wide HD3 dsRNA library annotation.

Let's define functions that do a B-Score normalization and reduce the chessboard like positional analysis bias.

```
conditional_mean=function(x,y){  
  if(!any(is.na(x))){  
    if(!any(is.na(y))){  
      return((x+y)/2)  
    }else{  
      return((x))  
    }  
  }else{  
    if(!any(is.na(y))){  
      return((y))  
    }else{  
      return(NA)  
    }  
  }  
}  
  
b_score_norm=function(x){  
  if(any(is.numeric(x))){  
    testm=matrix(0,16,24)  
    dimnames(testm)=list(LETTERS[1:16],1:24)  
    for(i in LETTERS[1:16]){  
      for(j in 1:24){  
        testm[i,j]=x[paste0(i,j)]  
      }  
    }  
    x1=LETTERS[1:16][seq(1,16,by=2)]  
    x2=LETTERS[1:16][seq(2,16,by=2)]  
    y1=c(1:24)[seq(1,24,by=2)]  
    y2=c(1:24)[seq(2,24,by=2)]  
    a=mean(c(testm[x1,y1],testm[x2,y2]),na.rm=T)  
    b=mean(c(testm[x2,y1],testm[x1,y2]),na.rm=T)  
    if(!any(is.na(c(a,b)))){  
      if(a>b){  
        testm[x1,y1]=testm[x1,y1]-abs(b-a)
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
testm[x2,y2]=testm[x2,y2]-abs(b-a)
}else{
  testm[x2,y1]=testm[x2,y1]-abs(b-a)
  testm[x1,y2]=testm[x1,y2]-abs(b-a)
}
}
medpol=medpolish(testm,na.rm = T,trace.iter =F)$residuals
medpol=medpol/(mad(medpol,na.rm = T))
y=x
for(i in LETTERS[1:16]){
  for(j in 1:24){
    y[paste0(i,j)]=medpol[i,j]
  }
}
return(y)
}else{
  return(x)
}

## load DMSO and MEK Screens (Geldanamycin and Diap1 co-perturbation screens are also loaded but only included
data('genome_wide_gene_drug_feature_data_raw', package='MODIFIdata')

genome_wide_gene_drug_feature_data=genome_wide_gene_drug_feature_data[
  intersect(
    grep("^\d{10}|\d{20}*",
         genome_wide_gene_drug_feature_data$plate),
    grep("^\$S18|\$S19|\$S17|\$S16|\$S21|\$S20|\$S30|\$S22|\$S23",
         genome_wide_gene_drug_feature_data$screenID)
  )
,]

screenIDs=c("S16","S17","S18","S19","S20","S21","S30","S22","S23")
plates = as.character(c(1001:1047,2001:2041))
wells = sort(unique(genome_wide_gene_drug_feature_data$well))

feature.list=as.list(names(genome_wide_gene_drug_feature_data)[!(names(genome_wide_gene_drug_feature_data) %in%
  names(feature.list)=names(genome_wide_gene_drug_feature_data)[!(names(genome_wide_gene_drug_feature_data) %in%
  names(feature.list)

arraylist = lapply(
  feature.list,
  function(feature){
    acast( genome_wide_gene_drug_feature_data, well ~ plate ~ screenID, value.var =feature, drop=TRUE)
  }
)

myarray=abind(arraylist,along=0)
names(dimnames(myarray))=c("feature","well","plate","screen.id")
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
myarray=myarray[-grep("dark|fussel|cx|cy|theta",dimnames(myarray)$feature),,]
```

create a list of control names with associated wells

```
well_array = list(1:8, 1:2, 1:2, 1:2, 1:2, 1:2)
names(well_array) = c("gfp", "rluc", "drk", "dmek", "pten", "gap1")
well_array$gfp = c("A1","C2","E2","G2","I2", "K2", "M2", "O2")
well_array$rluc = c("A2","B1")
well_array$drk = c("B2","J2")
well_array$dmek = c("D2","L2")
well_array$pten = c("F2","N2")
well_array$gap1 = c("H2","P2")

#creates a list of screen.id names
screen_array = names(myarray[1,1,1,])

#two plates entirely failed as assessed by manual inspection -> probable contamination
myarray[,"1013","S30"]=NA
myarray[,"2028","S19"]=NA
feature_choice=c("actin.b.mad.sd","actin.b.mad.tmean","actin.b.mean.sd","actin.b.mean.tmean","actin.h.cor.s1",
                 , "actin.s.perimeter.sd","actin.s.perimeter.tmean","actin.s.radius.mean.sd","actin.s.ra
                 , "DNA.b.mad.tmean","DNA.b.mean.tmean","DNA.h.ent.s1.sd","DNA.h.ent.s1.tmean","DNA.h.v
                 , "DNA.s.radius.sd.sd","DNA.s.radius.sd.tmean","nuclear.displacement.tmean","tubulin.b.
```

2 Data normalization

```
#new_features_data is the data of the reduced number of clusters to be used in further analysis
wnorm_array <- myarray[feature_choice,,,]
#Add the minimum of each feature to the respective feature, to reduce problems with log()
for(feature in dimnames(wnorm_array)$feature){
  wnorm_array[feature,,,] <- wnorm_array[feature,,,] + abs(min(wnorm_array[feature,,,],na.rm=TRUE))
}

#Transform the data with log(x+1)
log_array <- log1p(wnorm_array) #the actual data for later usage

#dividing all well by the plates median, setting the plate to 0, if the median is 0 and do nothing if the me
new_features_data <- apply(wnorm_array, c(1,3,4), b_score_norm)

#rearranges the array to the same format as myarray
new_features_data <- filtered_wells <- aperm(new_features_data, c(2,1,3,4))

#Rename the manipulated dimension
names(dimnames(new_features_data))[2] <- "well"
```

3 Normalization for positional biases

```
#we filter out technically failed wells
bad_wells=which(is.na(apply(new_features_data,c(2,3,4),function(x){if(max(x,na.rm = T)>50){return(NA)}else{return(x)}}))
for(i in 1:nrow(bad_wells)){
  filtered_wells[,bad_wells[i,"well"],bad_wells[i,"plate"],bad_wells[i,"screen.id"]]=NA
}

#we calculate average data for each treatment thread is Diap1 knockdown, hsp90 is Geldanamycin treatment, controls are S16, S18, S20, S22
mean_data <- filtered_wells[,,c("S16", "S18", "S20","S22")]
dimnames(mean_data)$screen.id <- c("control", "mek", "hsp90","thread")

for(well in dimnames(filtered_wells)$well){
  for(plate in dimnames(filtered_wells)$plate){
    mean_data[,well,plate,"control"]=conditional_mean(filtered_wells[,well,plate,"S16"],filtered_wells[,well,plate,"S18"])
    mean_data[,well,plate,"mek"]=conditional_mean(filtered_wells[,well,plate,"S18"],filtered_wells[,well,plate,"S20"])
    mean_data[,well,plate,"hsp90"]=conditional_mean(filtered_wells[,well,plate,"S20"],filtered_wells[,well,plate,"S22"])
    mean_data[,well,plate,"thread"]=conditional_mean(filtered_wells[,well,plate,"S22"],filtered_wells[,well,plate,"S16"])
  }
}
```

4 Calculation of distance and dsRNA wise quality metrics

Do not execute if you are in a hurry. Takes about 60-90 min to run.

```
#read in the annotations
data('HD3_annotations', package='MODIFIdata')
data('fbgn_knowledgesum', package='MODIFIdata')
data('RNA_expression_values', package='MODIFIdata')

row.names(go)=go[,1]

#creating the list of all controls to be removed from the distance calculation
well_to_gene_mapping=list()
annotation_list=list()
for(i in 1:nrow(annotation)){
  well_to_gene_mapping[[as.character(annotation[i,1])]][[annotation[i,2]]]=annotation[i,"gene"]
  annotation_list[[paste(as.character(annotation[i,1]),annotation[i,2],annotation[i,"gene"], sep = ".")]]= annotation[i,"gene"]
}

distances_to_gfp=list()
strongest_distances_to_gfp=list()
#Distance between each screen group and the gfp control
for(perturbation in dimnames(mean_data)$screen.id){
  #generating the mean gfp feature vector for the perturbation
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
mean_gfp_control <- list()
for(feature in dimnames(new_features_data)$feature){
  mean_gfp_control[[feature]] <- mean(mean_data[feature,well_array$gfp,,perturbation], na.rm = TRUE)
}
mean_gfp_control=unlist(mean_gfp_control)
#calculating the distances of each well in the first screen group to the gfp feature vector:
dist_control <- apply(mean_data[,,perturbation],c(2,3),function(x){sqrt(sum((x-mean_gfp_control)^2))})
strongest_dist_control <- apply(mean_data[,,perturbation],c(2,3),function(x){names(sort((x-mean_gfp_control)^2))})
#extracting the features with the biggest distance to the gfp controls
for(i in rownames(dist_control)){
  for(j in colnames(dist_control)){
    distances_to_gfp[[perturbation]][[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= dist_control[i,j]
    strongest_distances_to_gfp[[perturbation]][[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]=paste(strongest_dist_control[i,j])
  }
}

#Calculating the distances between the control screens and the mek screens
dist_mek <- list()

distmek=apply(mean_data,c(2,3),function(x){sqrt(sum((x[, "control"]-x[, "mek"])^2)))})
for(i in rownames(distmek)){
  for(j in colnames(distmek)){
    dist_mek[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= distmek[i,j]
  }
}
#Calculating the distances between the control screens and the hsp90 screens
dist_hsp <- list()

disthsp=apply(mean_data,c(2,3),function(x){sqrt(sum((x[, "control"]-x[, "hsp90"])^2)))})
for(i in rownames(disthsp)){
  for(j in colnames(disthsp)){
    dist_hsp[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= disthsp[i,j]
  }
}
#Calculating the distances between the control screens and the thread screens
dist_thread <- list()

distthread=apply(mean_data,c(2,3),function(x){sqrt(sum((x[, "control"]-x[, "thread"])^2)))})
for(i in rownames(distthread)){
  for(j in colnames(distthread)){
    dist_thread[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= distthread[i,j]
  }
}

#Checking for reproducability between 1k and 2k plates via correlation
fbgn=list()

for(i in 1:nrow(annotation)){
  if(is.null( fbgns[[annotation[i,"fbgn"]]][[[1]]])){
    fbgns[[annotation[i,"fbgn"]]][[[1]]]=annotation[i,c("well","plate","gene")]
  }
}
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

}else{
  fbgn[[annotation[i,"fbgn"]]][[2]]=annotation[i,c("well","plate","gene")]
}
}
for(i in names(fbgn)){
  if(!is.na(i)){
    if(length(fbgn[[i]])<2){
      fbgn[[i]][[2]]=c(NA,NA,NA)
    }
  }
}
designs=list()

for(i in names(fbgn)){
  if(!is.na(i)){
    designs[[paste(fbgn[[i]][[1]][2],fbgn[[i]][[1]][1],fbgn[[i]][[1]][3],sep=".")]]=paste(fbgn[[i]][[2]][2],fbgn[[i]][[2]][1],fbgn[[i]][[2]][3],sep="."))
    designs[[paste(fbgn[[i]][[2]][2],fbgn[[i]][[2]][1],fbgn[[i]][[2]][3],sep=".")]]=paste(fbgn[[i]][[1]][2],fbgn[[i]][[1]][1],fbgn[[i]][[1]][3],sep="."))
  }
}
x=list()
design1=new_features_data[c("cell_number","actin.s.perimeter.tmean","actin.m.eccentricity.tmean","DNA.s.perimeter.tmean")]
for(wellplategene in names(designs)){
  if(wellplategene != "NA.NA.NA" & designs[[wellplategene]] != "NA.NA.NA"){
    x[[["S30"]][[wellplategene]] <- cor(design1[,unlist(strsplit(wellplategene,split = "\\."))][2],unlist(strsplit(designs[[wellplategene]],split = "\\.")))[2],use = "pairwise.complete.obs")
  }
}
design1=new_features_data[c("cell_number","actin.s.perimeter.tmean","actin.m.eccentricity.tmean","DNA.s.perimeter.tmean")]
for(wellplategene in names(designs)){
  if(wellplategene != "NA.NA.NA" & designs[[wellplategene]] != "NA.NA.NA"){
    x[[["S17"]][[wellplategene]] <- cor(design1[,unlist(strsplit(wellplategene,split = "\\."))][2],unlist(strsplit(designs[[wellplategene]],split = "\\.")))[2],use = "pairwise.complete.obs")
  }
}
#Correlation between screen 16 and 17
#Calculating the distances between the control screens and the hsp90 screens
z16_17 <- list()

z1617=apply(new_features_data,c(2,3),function(x){cor(x[, "S16"],x[, "S17"],use="pairwise.complete.obs")})
for(i in rownames(disthsp)){
  for(j in colnames(disthsp)){
    z16_17[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= z1617[i,j]
  }
}

z18_19 <- list()

z1819=apply(new_features_data,c(2,3),function(x){cor(x[, "S18"],x[, "S19"],use="pairwise.complete.obs")})
for(i in rownames(disthsp)){

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
for(j in colnames(disthsp)){
  z18_19[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= z1819[i,j]
}
}

z20_21 <- list()

z2021=apply(new_features_data,c(2,3),function(x){cor(x[, "S20"],x[, "S21"],use="pairwise.complete.obs")})
for(i in rownames(disthsp)){
  for(j in colnames(disthsp)){
    z20_21[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= z2021[i,j]
  }
}

z22_23 <- list()

z2223=apply(new_features_data,c(2,3),function(x){cor(x[, "S22"],x[, "S23"],use="pairwise.complete.obs")})
for(i in rownames(disthsp)){
  for(j in colnames(disthsp)){
    z22_23[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= z2223[i,j]
  }
}

z18_30 <- list()

z1830=apply(new_features_data,c(2,3),function(x){cor(x[, "S18"],x[, "S30"],use="pairwise.complete.obs")})
for(i in rownames(disthsp)){
  for(j in colnames(disthsp)){
    z18_30[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= z1830[i,j]
  }
}

z19_30 <- list()

z1930=apply(new_features_data,c(2,3),function(x){cor(x[, "S19"],x[, "S30"],use="pairwise.complete.obs")})
for(i in rownames(disthsp)){
  for(j in colnames(disthsp)){
    z19_30[[paste(j,i,well_to_gene_mapping[[j]][[i]], sep = ".")]]= z1930[i,j]
  }
}

res=list()

for(wellplategene in names(dist_mek)){
  temp=c()
  temp=c(temp,annotation_list[[wellplategene]])
  for(pert in names(distances_to_gfp)){
    temp=c(temp,distances_to_gfp[[pert]][[wellplategene]])
  }
  for(pert in names(distances_to_gfp)){
    temp=c(temp,strongest_distances_to_gfp[[pert]][[wellplategene]])
  }
}
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
}

if(wellplategene %in% names(x$S17)){
  temp=c(temp,
    dist_mek[[wellplategene]],
    dist_hsp[[wellplategene]],
    dist_thread[[wellplategene]],
    x$S17[[wellplategene]],
    x$S30[[wellplategene]],
    z16_17[[wellplategene]],
    z18_19[[wellplategene]],
    z20_21[[wellplategene]],
    z22_23[[wellplategene]],
    z18_30[[wellplategene]],
    z19_30[[wellplategene]],
    designs[[wellplategene]])

}else{
  temp=c(temp,
    dist_mek[[wellplategene]],
    dist_hsp[[wellplategene]],
    dist_thread[[wellplategene]],
    NA,
    NA,
    z16_17[[wellplategene]],
    z18_19[[wellplategene]],
    z20_21[[wellplategene]],
    z22_23[[wellplategene]],
    z18_30[[wellplategene]],
    z19_30[[wellplategene]],
    NA)
}

temp=c(temp,expression_values[as.character(annotation_list[[wellplategene]]["fbgn"]),1])
temp=c(temp,go[as.character(annotation_list[[wellplategene]]["fbgn"]),2])
#print(length(temp))
res[[wellplategene]]=temp
}

res=do.call("rbind.data.frame",res)
names(res)=c("plate","well","id","content","query_id","fbgn","gene",paste(names(distances_to_gfp),"dist_to_gfp"))

#creating the lists for the shiny app / hit identification
#gene_feature contains all genes/wells on all plates with all screens
columns <- list()
i <- 1
setup <- melt(new_features_data[1,,], value.name = feature, varnames = c("well", "plate", "screen"),as.is = TRUE)
for(feature in names(new_features_data[,1,1,1])){
  columns[[feature]] <- melt(new_features_data[feature,,])[,4]
  i <- i + 1
}
columns <- append(columns, setup[1:3],0)
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

columns[["merger"]] <- paste(columns[["well"]],columns[["plate"]],sep=".")
annotation[["merger"]] <- paste(annotation[["well"]],annotation[["plate"]],sep=".")

gene_feature <- merge(as.data.frame(columns),annotation[,c("gene","merger")], by=c("merger"))
gene_feature <- gene_feature[-1]
gene_feature=cbind(gene_feature[,"gene"],gene_feature)
gene_feature <- gene_feature[-ncol(gene_feature)]
names(gene_feature)[1]="gene"

for(i in names(res)){res[[i]]=as.character(res[[i]])}
for(i in c("control_dist_to_gfp_Q1" ,
  "mek_dist_to_gfp_Q1" ,
  "hsp90_dist_to_gfp_Q1" ,
  "dist_to_mek_Q3" ,
  "dist_to_hsp_Q3" ,
  "design_corr_S17_Q4" ,
  "design_corr_S30_Q4" ,
  "biol_corr_DMS01_DMS02_Q5",
  "biol_corr_MEK1_MEK2_Q5" ,
  "biol_corr_HSP1_HSP2_Q5" ,
  "biol_corr_MEK1_MEK3_Q5" ,
  "biol_corr_MEK2_MEK3_Q5" ,
  "log2_expression_Q6" ,
  "go_knowledge_sum_Q7",
  "dist_to_thread_Q3",
  "biol_corr_THREAD1_THREAD2_Q5" ,
  "thread_dist_to_gfp_Q1"
)) {
  res[[i]]=as.numeric(res[[i]])
}
res$diffences_in_phenotype_strength_ctrl_mek_Q10=res$mek_dist_to_gfp_Q1-res$control_dist_to_gfp_Q1
for(i in c("control_dist_to_gfp_Q1" ,
  "mek_dist_to_gfp_Q1" ,
  "hsp90_dist_to_gfp_Q1" ,
  "dist_to_mek_Q3" ,
  "dist_to_hsp_Q3" ,
  "dist_to_thread_Q3",
  "thread_dist_to_gfp_Q1",
  "diffences_in_phenotype_strength_ctrl_mek_Q10"
)){
  res[[i]]=(res[[i]]-mean(res[[i]],na.rm = T))/(sd(res[[i]],na.rm = T))
}

res$biol_corr_max_Q8=apply(res[,c("biol_corr_DMS01_DMS02_Q5",
  "biol_corr_MEK1_MEK2_Q5" ,
  "biol_corr_HSP1_HSP2_Q5" ,
  "biol_corr_MEK1_MEK3_Q5" ,
  "biol_corr_MEK2_MEK3_Q5")],1,max,na.rm=T)
res$design_corr_max_Q9=apply(res[,c(
  "design_corr_S17_Q4" ,
  "design_corr_S30_Q4" ,
  "biol_corr_THREAD1_THREAD2_Q5" ,
  "log2_expression_Q6" ,
  "dist_to_thread_Q3" ,
  "dist_to_mek_Q3" ,
  "dist_to_hsp_Q3" ,
  "mek_dist_to_gfp_Q1" ,
  "hsp90_dist_to_gfp_Q1" ,
  "control_dist_to_gfp_Q1" ,
  "go_knowledge_sum_Q7" ,
  "biol_corr_DMS01_DMS02_Q5" ,
  "biol_corr_MEK1_MEK2_Q5" ,
  "biol_corr_HSP1_HSP2_Q5" ,
  "biol_corr_MEK1_MEK3_Q5" ,
  "biol_corr_MEK2_MEK3_Q5" ,
  "log2_expression_Q6" ,
  "dist_to_thread_Q3" ,
  "thread_dist_to_gfp_Q1" ,
  "diffences_in_phenotype_strength_ctrl_mek_Q10")]
,1,max,na.rm=T)

```

```
"design_corr_S30_Q4"      ),1,max,na.rm=T)

for(i in names(gene_feature)){gene_feature[[i]]=as.character(gene_feature[[i]])}
for(i in names(gene_feature)[5:length(gene_feature)]){
  gene_feature[[i]]=as.numeric(gene_feature[[i]])
}
```

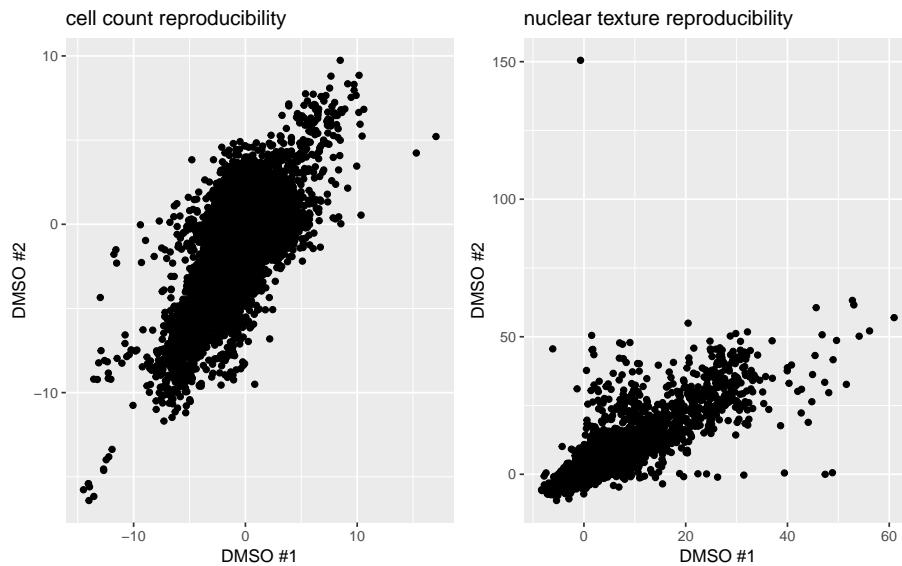
5 Figure 1 B: Plot correlations of screens

Screening sets 16 & S17 are DMSO treated genome wide screens. Screening sets 18 & S19 are MEKi treated genome wide screens.

```
a<-gene_feature %>%
 tbl_df() %>%
  filter(screen %in% c("S18","S19","S16","S17")) %>%
  gather(feature,value,-gene,-well,-plate,-screen) %>%
  spread(screen,value) %>%
  filter(feature=="cell_number") %>%
  ggplot(aes(x=S16,y=S17)) +
  labs(x="DMSO #1",y="DMSO #2") +
  ggtitle("cell count reproducibility") +
  geom_point()

b<-gene_feature %>%
 tbl_df() %>%
  filter(screen %in% c("S18","S19","S16","S17")) %>%
  gather(feature,value,-gene,-well,-plate,-screen) %>%
  spread(screen,value) %>%
  filter(feature=="DNA.h.var.s1.tmean") %>%
  ggplot(aes(x=S16,y=S17)) +
  labs(x="DMSO #1",y="DMSO #2") +
  ggtitle("nuclear texture reproducibility") +
  geom_point()

a+b
```



6 Figure 1 C: Plot correlations of Ras pathway genes

```

tocor<-gene_feature %>%
 tbl_df() %>%
  filter(screen %in% c("S18"), gene %in% c("Dsor1", "drk")) %>%
  group_by(gene) %>%
  summarise_if(is.numeric, mean, na.rm=T) %>%
  gather(feature,value,-gene) %>%
  spread(gene,value) %>%
  dplyr::select(-feature) %>%
  drop_na() %>%
  as.data.frame()

a <- tocor %>%
  ggplot(aes(x=.[[1]],y=.[[2]])) +
  geom_point() +
  geom_smooth(method = "rlm") +
  labs(x="drk",y="Dsor1") +
  ggtitle(paste0("r= ", rcorr(x=tocor[[1]],y=tocor[[2]]),type = "pearson")$r[2]," p~ ", rcorr(x=tocor[[1]],y=tocor[[2]]),type = "pearson")$p[2])

tocor<-gene_feature %>%
  tbl_df() %>%
  filter(screen %in% c("S18"), gene %in% c("Dsor1", "RasGAP1")) %>%
  group_by(gene) %>%
  summarise_if(is.numeric, mean, na.rm=T) %>%
  gather(feature,value,-gene) %>%
  spread(gene,value) %>%
  dplyr::select(-feature) %>%
  drop_na() %>%
  as.data.frame()

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

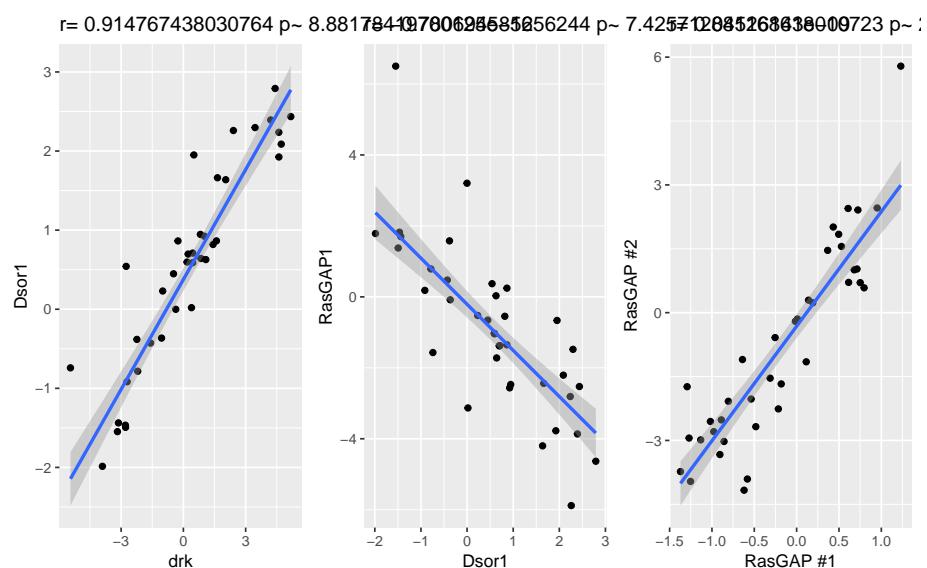
b <- tocor %>%
  ggplot(aes(x=.[[1]],y=.[[2]])) +
  geom_point() +
  geom_smooth(method = "rlm") +
  labs(x="Dsor1",y="RasGAP1") +
  ggtitle(paste0("r= ", rcorr(x=tocor[[1]],y=tocor[[2]]),type = "pearson")$r[2]," p~ ", rcorr(x=tocor[[1]],y=tocor[[2]]),type = "pearson")$p[2])

tocor <- gene_feature %>%
 tbl_df() %>%
  filter(screen %in% c("S18","S19"),gene %in% c("RasGAP1")) %>%
  group_by(well,screen) %>%
  filter(n()==2) %>%
  ungroup() %>%
  dplyr::select(-gene,-well) %>%
  gather(feature,value,-plate,-screen) %>%
  group_by(plate,feature) %>%
  summarise(value=mean(value,na.rm=T)) %>%
  spread(plate,value) %>%
  drop_na() %>%
  dplyr::select(-feature)

c <- tocor %>%
  ggplot(aes(x=.[[1]],y=.[[2]])) +
  geom_point() +
  geom_smooth(method = "rlm") +
  labs(x="RasGAP #1",y="RasGAP #2") +
  ggtitle(paste0("r= ", rcorr(x=tocor[[1]],y=tocor[[2]]),type = "spearman")$r[2]," p~ ", rcorr(x=tocor[[1]],y=tocor[[2]]),type = "spearman")$p[2])

a + b + c

```

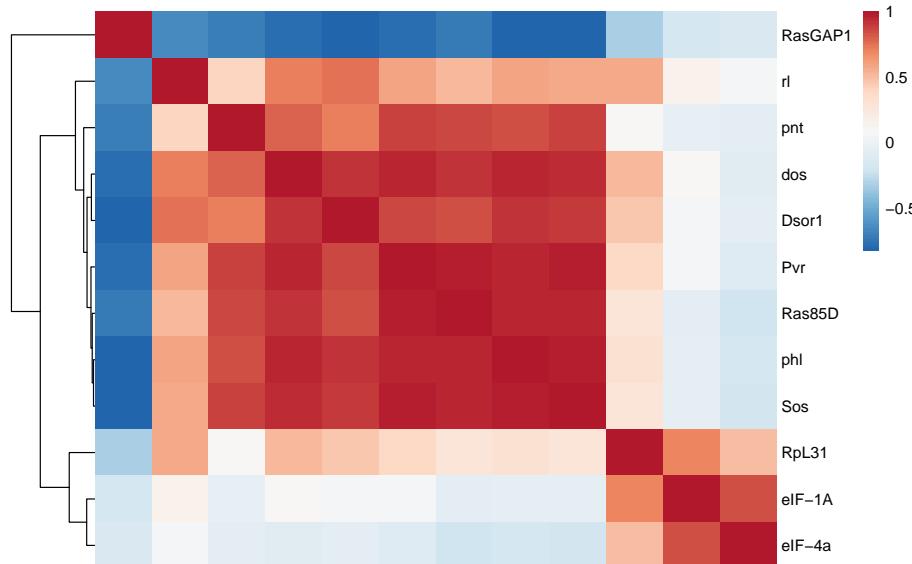


7 Figure 1 D&E: Plot heatmap of Ras pathway

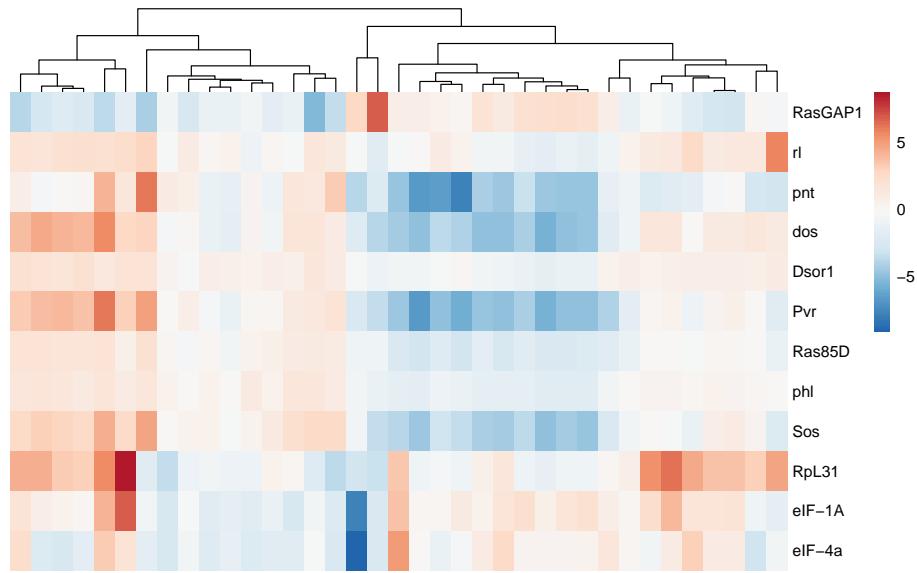
```
toplot<-gene_feature %>%
 tbl_df() %>%
  filter(screen %in% c("S18","S19"), gene %in% c("Ras85D", "phl", "rl", "pnt", "Sos", "dos", "Pvr", "RasGAP1", "Dsor1"))
gather(feature,value,-gene,-well,-plate,-screen) %>%
filter(feature!="actin.h.cor.s1.sd") %>%
group_by(gene,feature) %>%
summarise(value=mean(value,na.rm=T)) %>%
spread(gene,value) %>%
as.data.frame()

row.names(toplot)=toplot$feature

ordering <- cor(toplot[,-1]) %>% pheatmap::pheatmap( color = colorRampPalette(rev(brewer.pal(n = 7, name =
"RdBu")))(100),border_color = NA,treeheight_col = 0,show_colnames = F)
```



```
feature_heatmap<-pheatmap::pheatmap(t(toplot[,-1])[ordering$tree_row$order,],#
cluster_rows = F,show_colnames = F,color = colorRampPalette(rev(brewer.pal(n = 7, name = "RdBu")))(100))
```



8 Supplementary Figure S1B: Correlation of biological and chemical perturbation of Ras signaling

```

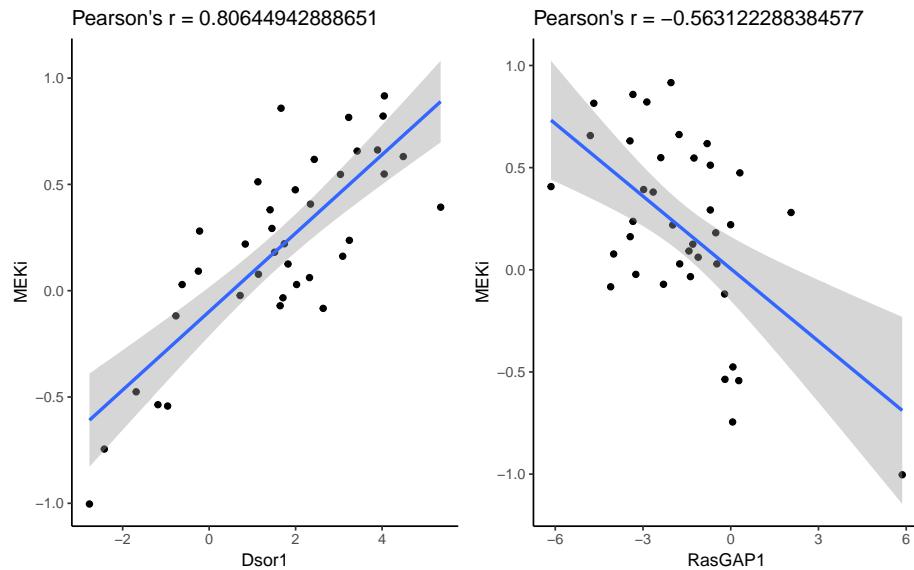
gene_feature %>>%tbl_df()%>%filter(screen %in% c("S16", "S17", "S18", "S19"))
a<-gene_feature %>%filter(well %in% c("B1", "A2"), screen %in% c("S18"), grepl("10*", plate)) %>%gather(feature,
b<-gene_feature %>%filter(well %in% c("D2", "L2"), screen %in% c("S16", "S17"), grepl("10*", plate)) %>%gather(feature)

c<-left_join(a,b) %>% ggplot(aes(x=Dsor1,y=MEKi)) + geom_point() +geom_smooth(method="lm") + ggtitle(paste0("Dsor1 vs MEKi", screen))
b<-gene_feature %>%filter(well %in% c("H2", "P2"), screen %in% c("S16", "S17"), grepl("10*", plate)) %>%gather(feature)

d<-left_join(a,b) %>% ggplot(aes(x=RasGAP1,y=MEKi)) + geom_point() +geom_smooth(method="lm") + ggtitle(paste0("RasGAP1 vs MEKi", screen))

c+d

```



9 Supplementary Figure S1D-G: Compound characterization on Dmel2 cells

```

data('ED50_feature_data', package='MODIFIdata')

par(mfrow=c(3,1))
#####
# plot dose-response of PD-0325901 treated cells; all concentrations are nM; GFP dsRNA is a non-targeting control
#####

feature="cells"
x=100000
feature_vector=c()
for(i in c(2:23)){
  x=x/2
  wells=grep(paste0("^\$EC50_GFP..",i,"\$"),
             paste0(measured_data$mean_per_field_mapping$plate, measured_data$mean_per_field_mapping$well))
  tobind=cbind(rep(x,
                    times=length(measured_data$mean_per_field_data[wells,feature])),
  measured_data$mean_per_field_data[wells,feature])
)
  row.names(tobind)=paste0(measured_data$mean_per_field_mapping$plate, measured_data$mean_per_field_mapping$well)
  feature_vector=rbind(feature_vector,tobind)
}

feature_vector=as.data.frame(feature_vector)
print(feature)
## [1] "cells"

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
colnames(feature_vector)=c("conc",as.character(feature))
mdl=drm(cells ~ conc, data = as.data.frame(feature_vector), fct = LL.4(names=c("Slope","Lower Limit","Upper
ed50=ED(mdl,50,interval="delta")
##
## Estimated effective doses
##
##      Estimate Std. Error   Lower   Upper
## e:1:50  1.45961    0.10887  1.24586  1.67335
mean=aggregate.data.frame(feature_vector,by = list(feature_vector$conc),mean)[,c(1,3)]
names(mean)=c("conc","mean")
plot(mdl,col="blue",type="none",main=paste("GFP_rec","ED50 = ",round(ed50[1],digits = 3),"+-",round(ed50[2],
points(mean,pch=19,cex=0.8,col="red",main=feature)

sem=aggregate.data.frame(feature_vector,by = list(feature_vector$conc),function(x){sd(x)/sqrt(length(x))})
names(sem)=c("conc","sem")
arrows(sem$conc, mean$mean-sem$sem, sem$conc, mean$mean+sem$sem, length=0.05, angle=90, code=3,"red")
abline(v = ed50[1],col="blue",lwd=2)

#####
# plot dose-response of fresh medium on compound treated cells after compound washout
#####

feature="cells"
x=100000
feature_vector=c()
for(i in c(2:23)){
  x=x/2
  wells=grep(paste0("^EC50_rec.",i,"$"),
             paste0(measured_data$mean_per_field_mapping$plate,measured_data$mean_per_field_mapping$well))
  tobind=cbind(rep(x,
                    times=length(measured_data$mean_per_field_data[wells,feature]))
  ),
  measured_data$mean_per_field_data[wells,feature]
)
  row.names(tobind)=paste0(measured_data$mean_per_field_mapping$plate,measured_data$mean_per_field_mapping$well)
  feature_vector=rbind(feature_vector,tobind)
}

feature_vector=as.data.frame(feature_vector)
print(feature)
## [1] "cells"
colnames(feature_vector)=c("conc",as.character(feature))
mdl=drm(cells ~ conc, data = as.data.frame(feature_vector), fct = LL.4(names=c("Slope","Lower Limit","Upper
ed50=ED(mdl,50,interval="delta")
##
## Estimated effective doses
##
##      Estimate Std. Error   Lower   Upper
## e:1:50  6.90663    0.77694  5.37855  8.43471
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

mean=aggregate.data.frame(feature_vector,by = list(feature_vector$conc),mean)[,c(1,3)]
names(mean)=c("conc","mean")
plot(mdl,col="blue",type="none",main=paste("GFP_rec","ED50 = ",round(ed50[1],digits = 3),"+-",round(ed50[2],2)))
points(mean,pch=19,cex=0.8,col="red",main=feature)#,log="x",plot.new=F

sem=aggregate.data.frame(feature_vector,by = list(feature_vector$conc),function(x){sd(x)/sqrt(length(x))})
names(sem)=c("conc","sem")
arrows(sem$conc, mean$mean-sem$sem, sem$conc, mean$mean+sem$sem, length=0.05, angle=90, code=3,"red")
abline(v = ed50[1],col="blue",lwd=2)

#####
# plot dose-response of used compound on fresh cells
#####

feature="cells"
x=100000
feature_vector=c()
for(i in c(2:23)){
  x=x/2
  wells=grep(paste0("^EC50_stab_..",i,"$"),
             paste0(measured_data$mean_per_field_mapping$plate,measured_data$mean_per_field_mapping$well))
  tobind=cbind(rep(x,
                    times=length(measured_data$mean_per_field_data[wells,feature]))
  ),
  measured_data$mean_per_field_data[wells,feature]
)
  row.names(tobind)=paste0(measured_data$mean_per_field_mapping$plate,measured_data$mean_per_field_mapping$well)
  feature_vector=rbind(feature_vector,tobind)
}

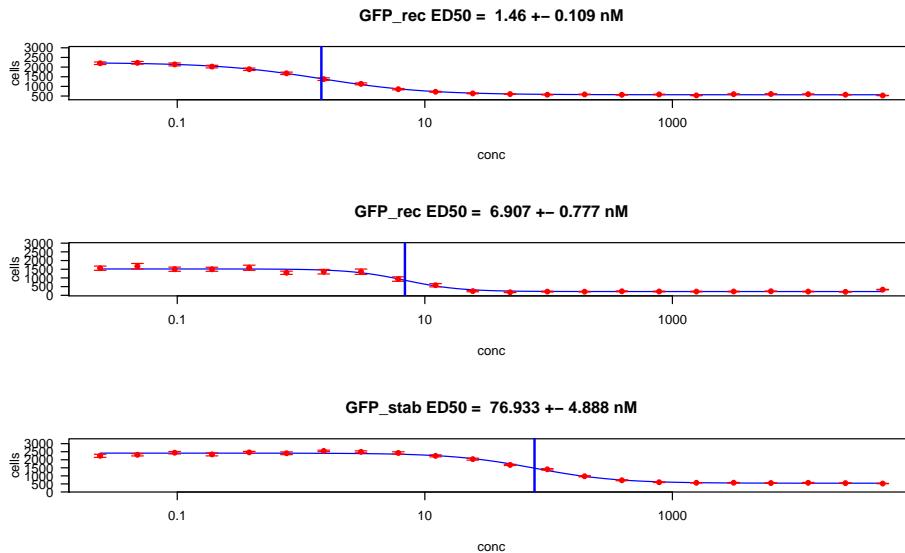
feature_vector=as.data.frame(feature_vector)
print(feature)
## [1] "cells"
colnames(feature_vector)=c("conc",as.character(feature))
mdl=drm(cells ~ conc, data = as.data.frame(feature_vector), fct = LL.4(names=c("Slope","Lower Limit","Upper Limit")))
ed50=ED(mdl,50,interval="delta")
##
## Estimated effective doses
##
##      Estimate Std. Error   Lower   Upper
## e:1:50  76.9334    4.8876 67.3373 86.5296
mean=aggregate.data.frame(feature_vector,by = list(feature_vector$conc),mean)[,c(1,3)]
names(mean)=c("conc","mean")
plot(mdl,col="blue",type="none",main=paste("GFP_stab","ED50 = ",round(ed50[1],digits = 3),"+-",round(ed50[2],2)))
points(mean,pch=19,cex=0.8,col="red",main=feature)#,log="x",plot.new=F

sem=aggregate.data.frame(feature_vector,by = list(feature_vector$conc),function(x){sd(x)/sqrt(length(x))})

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
names(sem)=c("conc", "sem")
arrows(sem$conc, mean$mean-sem$sem, sem$conc, mean$mean+sem$sem, length=0.05, angle=90, code=3, "red")
abline(v = ed50[1], col="blue", lwd=2)
```



```
#####
# plot dose-response of different cellular features
#####

par(mfrow=c(1,4))

for(feature in c("cells", "tubulin.b.mad.median", "DNA.b.mean.median", "DNA.s.radius.mean.median")){
  x=100000
  feature_vector=c()
  for(i in c(3:23)){
    x=x/2
    wells=grep(paste0("^EC50_stab_..", i, "$"),
               paste0(measured_data$mean_per_field_mapping$plate, measured_data$mean_per_field_mapping$well))
    tobind=cbind(rep(x,
                      times=length(measured_data$mean_per_field_data[wells, feature])),
                 measured_data$mean_per_field_data[wells, feature])
    )
    row.names(tobind)=paste0(measured_data$mean_per_field_mapping$plate, measured_data$mean_per_field_mapping$well)
    feature_vector=rbind(feature_vector, tobind)
  }

  feature_vector=as.data.frame(feature_vector)
  print(feature)
  colnames(feature_vector)=c("conc", as.character(feature))
  feature_vector$cells=(feature_vector[[feature]]-min(feature_vector[[feature]]))/(max(feature_vector[[feature]])-min(feature_vector[[feature]]))
  mean=aggregate.data.frame(feature_vector, by = list(feature_vector$conc), mean)[,c(1,3)]
  names(mean)=c("conc", "mean")
```

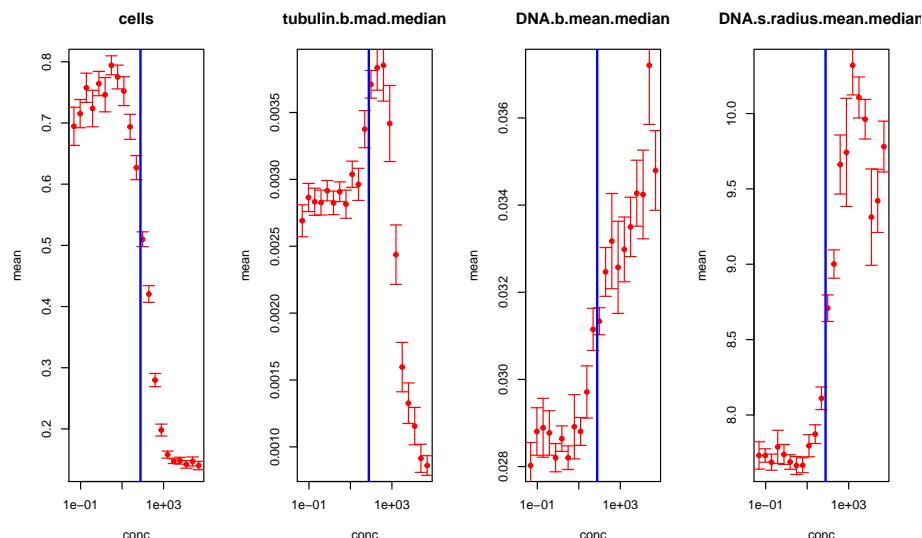
```

plot(mean,pch=19,cex=0.8,col="red",log="x",main=feature)

sem=aggregate.data.frame(feature_vector,by = list(feature_vector$conc),function(x){sd(x)/sqrt(length(x))})
names(sem)=c("conc","sem")
arrows(sem$conc, mean-mean-sem, sem$conc, mean+mean+sem$sem, length=0.05, angle=90, code=3,"red")
abline( v = 77,
        col="blue",lwd=2)
}

## [1] "cells"
## [1] "tubulin.b.mad.median"
## [1] "DNA.b.mean.median"
## [1] "DNA.s.radius.mean.median"

```



10 Supplementary Figure S2: Control separation capacity of our morphological assay

```

myarray.norm<-new_features_data

z_kernel=function(x,sigma=1){
  res=0
  for(i in 1:length(x)){
    for(j in 1:length(x)){
      res=res+exp(-((x[i]-x[j])^2/(2*sigma^2)))
    }
  }
  return(res/(length(x)^2))
}

z_factor=function(pos,neg){1-3*(sd(pos)+sd(neg))/(mean(pos)-mean(neg))}


```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
feature<- "cell_number"
plates_that_correlate.idx=c(1:44)
screen<- "S16"

plates=list()
for(i in plates_that_correlate.idx){
  plates[[i]]=t(myarray.norm[,c("B2","J2"),i,screen])
}
cur_neg=do.call("rbind.data.frame",plates)
cur_neg$class="neg"

plates=list()
for(i in plates_that_correlate.idx){
  plates[[i]]=t(myarray.norm[,c("H2","P2"),i,screen])
}
cur_pos=do.call("rbind.data.frame",plates)
cur_pos$class="pos"

dataset=rbind(cur_neg,cur_pos)
lda_anal=lda(class~,data = dataset)

plates=list()
for(i in plates_that_correlate.idx){
  cur_pos=t(myarray.norm[,c("H2","P2"),i,screen])
  cur_neg=t(myarray.norm[,c("B2","J2"),i,screen])
  neg_P=apply(cur_neg,1,function(x){sum(x*lda_anal$scaling)})
  pos_P=apply(cur_pos,1,function(x){sum(x*lda_anal$scaling)})
  plates[[i]]=cbind(i,pos_P,neg_P,z_factor(pos_P,neg_P))
}
multi=cur_z=do.call("rbind.data.frame",plates)

pos=cur_z[,2]
neg=cur_z[,3]
par(mfrow = c(2,3))
plot(cur_z[,1],cur_z[,2],pch=16,col="green",ylim=c(-10,10),ylab="weighted sum of features",xlab="plate-index")
points(cur_z[,1],cur_z[,3],pch=16,col="red")

plot(2,2,xlim=c(-10,10),ylim=c(0,1),sub=paste("Z'-factor =",round(mean(cur_z[,4],na.rm = T),digits=3)),ylab="")
lines(density(cur_z[,2],na.rm = T),col="green",lwd=2)
lines(density(cur_z[,3],na.rm = T),col="red",lwd=2)

plot(cur_z[,1],cur_z[,4],pch=16,col="black",ylim=c(-1,1),main=paste("Multifactorial-Z' =",round(mean(cur_z[,4],na.rm = T),digits=3)))
abline(h=mean(cur_z[,4],na.rm = T))

#####
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

plates=list()
for(i in plates_that_correlate.idx){
  pos_P=cur_pos=myarray.norm["cell_number",c("H2","P2"),i,screen]
  neg_P=cur_neg=myarray.norm["cell_number",c("B2","J2"),i,screen]
  plates[[i]]=cbind(i,pos_P,neg_P,z_factor(pos_P,neg_P))
}
uni=cur_z=do.call("rbind.data.frame",plates)

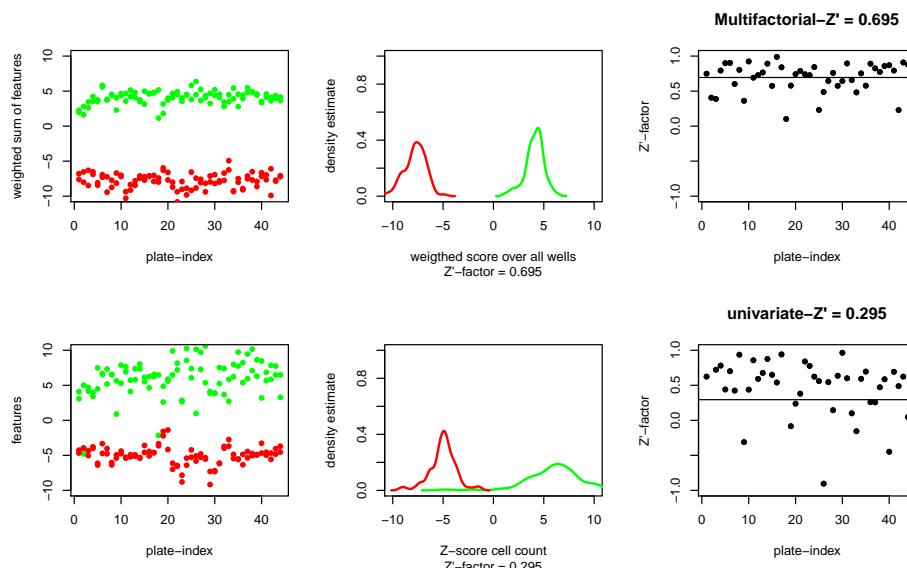
pos=cur_z[,2]
neg=cur_z[,3]

plot(cur_z[,1],cur_z[,2],pch=16,col="green",ylim=c(-10,10),ylab="features",xlab="plate-index")
points(cur_z[,1],cur_z[,3],pch=16,col="red")

plot(2,2,xlim=c(-10,10),ylim=c(0,1),sub=paste("Z'-factor =",round(mean(cur_z[,4],na.rm = T),digits=3)),ylab=
lines(density(cur_z[,2],na.rm = T),col="green",lwd=2)
lines(density(cur_z[,3],na.rm = T),col="red",lwd=2)

plot(cur_z[,1],cur_z[,4],pch=16,col="black",ylim=c(-1,1),main=paste("univariate-Z' =",round(mean(cur_z[,4],na.
abline(h=mean(cur_z[,4],na.rm = T)))

```



```
rm(list = ls())
```

11 Next we analyzed our time resolved genetic interaction screen

First we collect data, normalize by per-plate negative controls, glog transform each feature's data, scale the data, annotate the wells and reformat to an array with dimensions according to experimental conditions.

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

## load raw per-well level data in the form of a four dimensional array as well-set-plate-feature
data('raw_data_array_wCTRL', package='MODIFIdata')

##normalize to negative controls that did not see any query gene
neg=c("A12","A13","C12","C13",
      "E12","E13","G12","G13",
      "I12","I13","K12","K13",
      "M12","M13","O12","O13")

my.data.array.norm=apply(my.data.array,
                        c(2,3,4),
                        function(x){
                          m=median(x[neg],na.rm = T)
                          if(m==0){
                            return(x/1)
                          }else{
                            return(x/m)
                          }
                        }
                      )

##glog transform all features and center and scale them by robust Z-Score
for(i in dimnames(my.data.array)[[4]]){
  c=quantile(my.data.array.norm[,,,i],probs = 0.03,na.rm = T)
  my.data.array.norm[,,,i]=glog(x = my.data.array.norm[,,,i] , a = c)
  my.data.array.norm[,,,i]=(my.data.array.norm[,,,i]-median(my.data.array.norm[,,,i],na.rm = T))/mad(my.data.array.norm[,,,i])
}
my.data.array.norm_zprime<-my.data.array.norm

#extract the cell count data
my.data.array.norm.qc=my.data.array.norm[,,,'cell_number']

#calculate the Z-prime-factor for each assay plate between drk and RasGAP1 knockdown
my.data.array.zprime=apply(my.data.array.norm.qc,
                           c(2,3),
                           function(x){
                             negs=x[c("F12","F13")]
                             poss=x[c("H12","H13","J12","J13")]
                             imageHTS::zprime(a = negs,b = poss,method = "maha")
                           }
                         )

bad_S1S7=which(diag(cor(my.data.array.norm[, "S01",,1],my.data.array.norm[, "S07",,1], use = "p" ))<=0.6)
bad_S2S8=which(diag(cor(my.data.array.norm[, "S02",,1],my.data.array.norm[, "S08",,1], use = "p" ))<=0.6)
bad_S3S9=which(diag(cor(my.data.array.norm[, "S03",,1],my.data.array.norm[, "S09",,1], use = "p" ))<=0.6)
bad_S4S10=which(diag(cor(my.data.array.norm[, "S04",,1],my.data.array.norm[, "S10",,1], use = "p" ))<=0.6)
bad_S5S11=which(diag(cor(my.data.array.norm[, "S05",,1],my.data.array.norm[, "S11",,1], use = "p" ))<=0.6)
bad_S6S12=which(diag(cor(my.data.array.norm[, "S06",,1],my.data.array.norm[, "S12",,1], use = "p" ))<=0.6)

#only the S1S7 and S2S8 had plates which do not correlate <0.6
my.data.array.norm[, "S07",bad_S1S7,]=NA_real_

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

my.data.array.norm[, "S08", bad_S2S8, ]=NA_real_
bad_zprime=which(my.data.array.zprime<0.3, arr.ind = T)

#mask all low quality data from downstream analysis
for(i in 1:nrow(bad_zprime)){
  my.data.array.norm[,bad_zprime[i,1],bad_zprime[i,2],]=NA_real_
}

#load the annotations for each well and each plate and their mapping to HD3-IDs
data('query_mapping', package='MODIFIdata')
data('target_mapping', package='MODIFIdata')

dimnames(my.data.array.norm)[[1]][grep("B02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_1"
dimnames(my.data.array.norm)[[1]][grep("D02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_2"
dimnames(my.data.array.norm)[[1]][grep("F02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_3"
dimnames(my.data.array.norm)[[1]][grep("H02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_4"
dimnames(my.data.array.norm)[[1]][grep("J02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_5"
dimnames(my.data.array.norm)[[1]][grep("L02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_6"
dimnames(my.data.array.norm)[[1]][grep("N02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_7"
dimnames(my.data.array.norm)[[1]][grep("P02$", dimnames(my.data.array.norm)[[1]])]="HD31RLUC_8"
dimnames(my.data.array.norm)[[1]][grep("B23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_1"
dimnames(my.data.array.norm)[[1]][grep("D23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_2"
dimnames(my.data.array.norm)[[1]][grep("F23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_3"
dimnames(my.data.array.norm)[[1]][grep("H23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_4"
dimnames(my.data.array.norm)[[1]][grep("J23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_5"
dimnames(my.data.array.norm)[[1]][grep("L23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_6"
dimnames(my.data.array.norm)[[1]][grep("N23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_7"
dimnames(my.data.array.norm)[[1]][grep("P23$", dimnames(my.data.array.norm)[[1]])]="HD32RLUC_8"

for(i in 1:length(dimnames(my.data.array.norm)[[3]])){
  if(length(grep(paste0("^", dimnames(my.data.array.norm)[[3]][i], "$"), query_mappin$V2))!=0){
    dimnames(my.data.array.norm)[[3]][i]=query_mappin$V3[grep(paste0("^", dimnames(my.data.array.norm)[[3]][i], "$"), query_mappin$V2)]
  }
}

for(i in 1:length(dimnames(my.data.array.norm)[[1]])){
  if(length(grep(paste0("^", dimnames(my.data.array.norm)[[1]][i], "$"), target_mappin$V2))!=0){
    dimnames(my.data.array.norm)[[1]][i]=target_mappin$V1[grep(paste0("^", dimnames(my.data.array.norm)[[1]][i], "$"), target_mappin$V2)]
  }
}

my.data.array.norm_repl1.1=my.data.array.norm[grep("^HD3", dimnames(my.data.array.norm)[[1]]), c("S01", "S02", "S03")]
my.data.array.norm_repl1.2=my.data.array.norm[grep("^HD3", dimnames(my.data.array.norm)[[1]]), c("S07", "S08", "S09")]

my.data.array.norm=abind(my.data.array.norm_repl1.1, my.data.array.norm_repl1.2, rev.along = 0)
dimnames(my.data.array.norm)[[5]]=c("repl1", "repl2")

rm(my.data.array.norm_repl1.1, my.data.array.norm_repl1.2)
my.data.array.norm_DMSO_48=my.data.array.norm[, "S07", , , ]
my.data.array.norm_DMSO_72=my.data.array.norm[, "S08", , , ]
my.data.array.norm_DMSO_96=my.data.array.norm[, "S09", , , ]

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
my.data.array.norm_DMSO=abind(my.data.array.norm_DMSO_48,my.data.array.norm_DMSO_72,my.data.array.norm_DMSO_96)
dimnames(my.data.array.norm_DMSO)[[5]]=c("48h","72h","96h")
my.data.array.norm_MEKi_48=my.data.array.norm[, "S10", , ,]
my.data.array.norm_MEKi_72=my.data.array.norm[, "S11", , ,]
my.data.array.norm_MEKi_96=my.data.array.norm[, "S12", , ,]
my.data.array.norm_MEKi=abind(my.data.array.norm_MEKi_48,my.data.array.norm_MEKi_72,my.data.array.norm_MEKi_96)
dimnames(my.data.array.norm_MEKi)[[5]]=c("48h","72h","96h")
my.data.array.norm=abind(my.data.array.norm_DMSO,my.data.array.norm_MEKi,rev.along = 0)
dimnames(my.data.array.norm)[[6]]=c("DMSO","MEKi")
names(dimnames(my.data.array.norm))=c("target gene","query gene","feature","replicate","time point","perturbation")

rm(my.data.array.norm_DMSO_48,
  my.data.array.norm_DMSO_72,
  my.data.array.norm_DMSO_96,
  my.data.array.norm_MEKi_48,
  my.data.array.norm_MEKi_72,
  my.data.array.norm_MEKi_96,
  my.data.array.norm_DMSO,
  my.data.array.norm_MEKi,
  my.data.array)
```

Next we apply the med-polish to call statistic genetic interactions according to methods established by Bernd Fischer. Input is normed and quality controlled data. Output are genetic interactions as pi-scores, maineffects and reformatted measurements and p-value associated to interactions in each individual condition.

```
data('id_to_gene_mapping', package='MODIFIdata')
data('query_mapping', package='MODIFIdata')
data('target_mapping', package='MODIFIdata')

id_to_gene=list()
for(i in 1:nrow(id_to_gene_mapping)){
  id_to_gene[[id_to_gene_mapping[i,1]]]=id_to_gene_mapping[i,3]
  id_to_gene[[id_to_gene_mapping[i,2]]]=id_to_gene_mapping[i,3]
}

pb <- txtProgressBar(style = 3,max = 310,initial = 0,min = 1)
counter=1
interaction_scores=my.data.array.norm
mainEffects = list(target = interaction_scores[,1,,,,],
                    query = interaction_scores[1,,,,],
                    overall = interaction_scores[1,1,,,,])
#interactions are called using the HD2013SGImaineffects function implemented in the R-package whcih supplement this manuscript
# we called interactions indepently for each dsRNA, biological replicates, feature, time point and treatment
for(i in 1:155){
  for(j in 1:2){
    for(k in 1:3){
      for(l in 1:2){
        MP=HD2013SGI::HD2013SGImaineffects(x = my.data.array.norm[,,i,j,k,l],
                                                TargetNeg = c("HD31RLUC_1","HD32RLUC_1","HD31RLUC_2"))
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

"HD31RLUC_3", "HD32RLUC_3", "HD31RLUC_4",
"HD31RLUC_5", "HD32RLUC_5", "HD31RLUC_6",
"HD31RLUC_7", "HD32RLUC_7", "HD31RLUC_8"
na.rm=T
)
interaction_scores[,,i,j,k,l]=MP$pi
mainEffects$query[,i,j,k,l]=MP$queryMainEffect
mainEffects$target[,i,j,k,l]=MP$targetMainEffect
mainEffects$overall[i,j,k,l]=MP$neg
}
}
setTxtProgressBar(pb, counter)
counter=counter+1
}
}

# annotate genes from dsRNA identifiers
interaction_scores.des1=interaction_scores[grep("HD31",dimnames(interaction_scores)[[1]]),,,,]
interaction_scores.des2=interaction_scores[grep("HD32",dimnames(interaction_scores)[[1]]),,,,]

for(i in 1:length(dimnames(interaction_scores.des1)[[1]])){
  if(dimnames(interaction_scores.des1)[[1]][i] %in% names(id_to_gene)){
    dimnames(interaction_scores.des1)[[1]][i]=id_to_gene[[dimnames(interaction_scores.des1)[[1]][i]]]
  }else{
    dimnames(interaction_scores.des1)[[1]][i]=gsub("HD31(.+)", "\\\1", dimnames(interaction_scores.des1)[[1]][i])
  }
}

for(i in 1:length(dimnames(interaction_scores.des2)[[1]])){
  if(dimnames(interaction_scores.des2)[[1]][i] %in% names(id_to_gene)){
    dimnames(interaction_scores.des2)[[1]][i]=id_to_gene[[dimnames(interaction_scores.des2)[[1]][i]]]
  }else{
    dimnames(interaction_scores.des2)[[1]][i]=gsub("HD32(.+)", "\\\1", dimnames(interaction_scores.des2)[[1]][i])
  }
}

for(i in 1:length(dimnames(mainEffects$target)[[1]])){
  if(dimnames(mainEffects$target)[[1]][i] %in% names(id_to_gene)){
    dimnames(mainEffects$target)[[1]][i]=id_to_gene[[dimnames(mainEffects$target)[[1]][i]]]
  }
}

# resort interaction, maineffect and measured effect array such that dsRNA designs and biological replicates

# main effects, interactions and measured data, do now have the dimensions: target.gene-query.gene-feature-t
interaction_scores.des2=interaction_scores.des2[dimnames(interaction_scores.des1)[[1]],,,,]
interaction_scores.des1.1=interaction_scores.des1[,,,1,,]
interaction_scores.des1.2=interaction_scores.des1[,,,2,,]
interaction_scores.des2.1=interaction_scores.des2[,,,1,,]
interaction_scores.des2.2=interaction_scores.des2[,,,2,,]
interaction_scores=abind(interaction_scores.des1.1,interaction_scores.des1.2,interaction_scores.des2.1,interaction_scores.des2.2,沿)

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

fit = limma::eBayes(limma::lmFit(Data[K,]))
padj[K] = p.adjust(fit$p.value, method="BH")
PADJ[,,,i] = padj
cat(sprintf("i=%2d",i),
      " nr int (1%) = ", sum(padj <= 0.01, na.rm=TRUE)/nrow(Data),
      " nr int (3%) = ", sum(padj <= 0.03, na.rm=TRUE)/nrow(Data), "\n")
}

D=(D[,,,,,1]+D[,,,,,2]+D[,,,,,3]+D[,,,,,4])/4
for(i in 1:dim(D)[5]){
  D[,,,i]=D[,,,i]/s[i]
}
Interactions = list(piscore = D,
                     scale = s,
                     padj = PADJ
)
inter_df=melt(Interactions$piscore,varnames = c("target","query","time","drug","feature"),value.name = "piscore")
sig_df=melt(Interactions$padj,varnames = c("target","query","time","drug","feature"),value.name = "fdr")
data_df=tbl_df(full_join(inter_df,sig_df))

my.data.array.norm.reshaped_df<-
  melt(my.data.array.norm.reshaped,varnames=c("target","query","feature","time","drug","replicate"),value.name = "value")
  rm(sig_df,
     Interactions,
     inter_df,
     my.data.array.norm.reshaped,
     my.data.array.norm,
     D,
     s,
     PADJ)

```

12 Analyze feature correlation and filter out non-redundant features

Filter non-correlating features

13 Supplemetal Figure 3A: Featurewise biological reproductability analysis

```

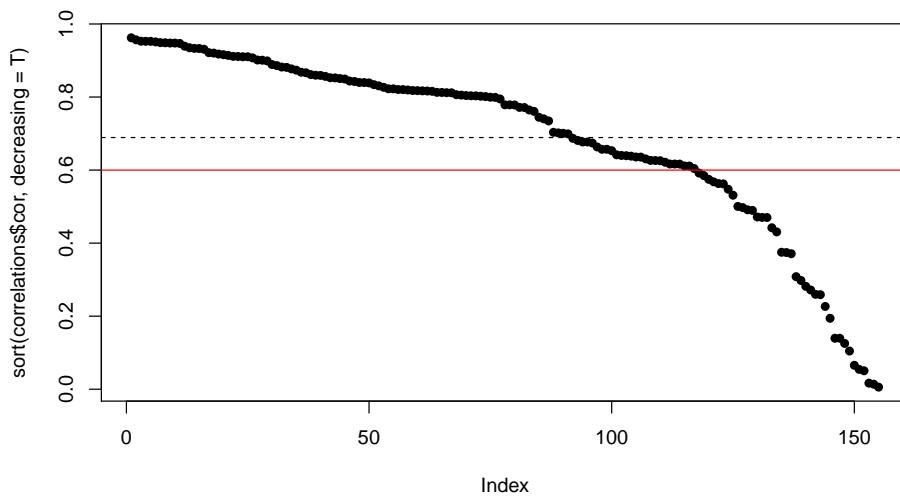
correlations=my.data.array.norm.reshaped_df %>%
  filter(time=="96h",drug=="MEKi") %>%
  extract(replicate,c("design","bio"),regex="(des\\d)(rep\\d)") %>%

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
spread(bio,value) %>%
group_by(feature) %>%
summarise(cor=cor(repl1,repl2,use = "p",method = "p"))

plot(sort(correlations$cor,decreasing = T),pch=16)
abline(h=0.6, col="red")
abline(h=mean(correlations$cor), col="black",lty=2)
```



```
mean(correlations$cor)
## [1] 0.6891315

#Filter features without any reproducible interactions in any condition

goodfeature<-
  correlations %>%
  mutate(feature=factor(feature,levels=feature[order(cor,decreasing=T)])) %>%
  dplyr::filter(cor>=0.6) %>%
  arrange(desc(cor)) %>%
  pull(feature) %>%
  as.character()

allfeature<-
  correlations %>%
  mutate(feature=factor(feature,levels=feature[order(cor,decreasing=T)])) %>%
  arrange(desc(cor)) %>%
  pull(feature) %>%
  as.character()
```

14 Supplemental Figure 3B: Pairwise feature correlation analysis and redundancy filter

```

require(RColorBrewer)
goodfeature=data_df %>%
  group_by(feature) %>%
  filter(fdr<=0.05 & feature %in% goodfeature) %>%
  mutate(sigIntfeat=length(fdr)) %>%
  pull(feature) %>%
  unique() %>%
  as.character()

#####
# compute features wise correlations or load them from previous session (takes a long time! do not run)
#####
# subdf=data_df %>% filter(time=="96h",drug=="MEKi",feature %in% goodfeature)
# tmp=combn(as.character(goodfeature),2,simplify = F)
# fun<-function(x){
#   c(cor(filter(subdf,feature %in% x[1])$piscore,
#         filter(subdf,feature %in% x[2])$piscore,
#         method = "p",use = "p"),x)
# }
# multicoreParam <- MulticoreParam(workers = 20,progressbar=TRUE)
# correllations=bpplapply(tmp,fun, BPPARAM = multicoreParam)
# cors=do.call("rbind.data.frame",correllations)
# names(cors) = c("corr","feat1","feat2")
# cors %>>%tbl_df() %>%mutate(corr=as.numeric(as.character(corr))) %>%arrange(desc(corr))

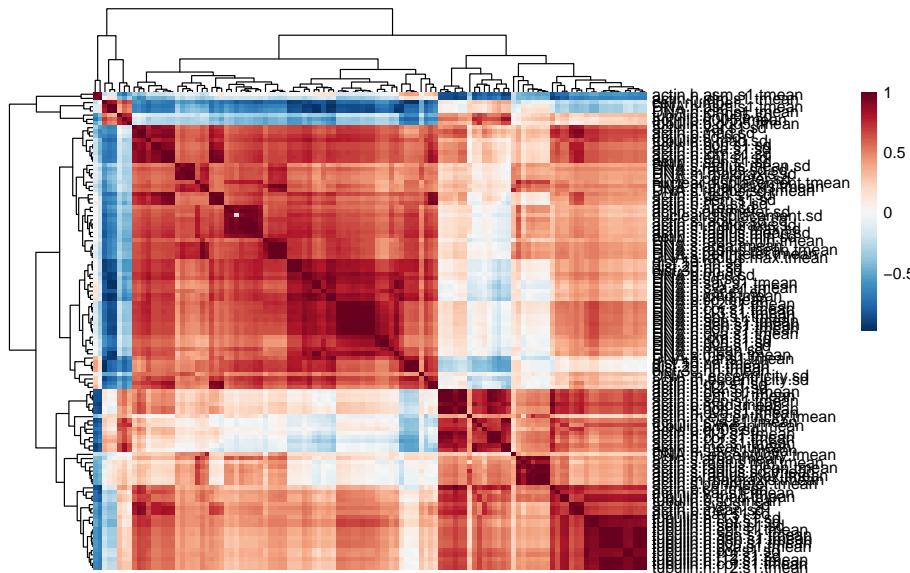
data('feature_correlations_interactions', package='MODIFIdata')
upperiag=cors
lowerdiag=upperiag %>% dplyr::select(corr,feat1=feat2,feat2=feat1)
diag=cbind.data.frame("corr"=1,"feat1"=levels(upperiag$feat1),"feat2"=levels(upperiag$feat1))

feature_corrs=bind_rows(upperiag,lowerdiag,diag) %>% spread(feat2,corr) %>% dplyr::select(-feat1) %>% t()

x=pheatmap::pheatmap(feature_corrs,color = colorRampPalette(rev(brewer.pal(n = 11, name ="RdBu")))(100))

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



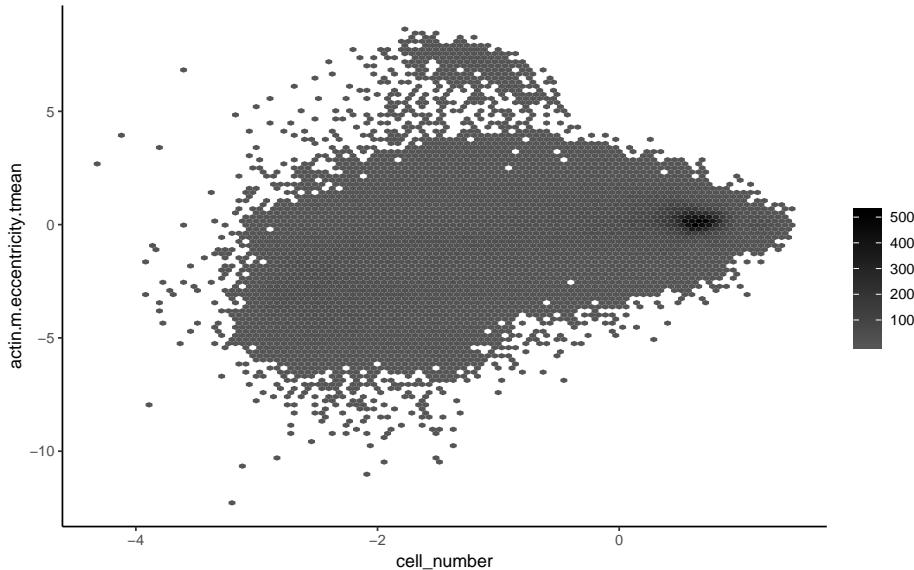
```
#####
# filter features such that highest reproducible features are chosen first
# second each feature that shows a pearson correlation higher than 0.7 with this feature is disregarded
# iteratively test that for all remaining features
# fix cell_number and tubulin.b.mad.mean
#####

for(i in allfeature){
  if(i %in% goodfeature){
    corrf=c(as.character(filter(cors,feat1==i & abs(corr)>0.7)$feat2),
           as.character(filter(cors,feat2==i & abs(corr)>0.7)$feat1))
    goodfeature=goodfeature[!goodfeature %in% corrf]
  }
}
goodfeature=c("cell_number","tubulin.b.mad.tmean",goodfeature) %>% unique()
```

15 Supplemetal Figure 3C: Scatterplot of uncorrelating features after redundancy checks

```
#show that they are independent
colfunc <- colorRampPalette(c("#555555", "black"))
colfunc(30)
## [1] "#555555" "#525252" "#4F4F4F" "#4C4C4C" "#494949" "#464646" "#434343"
## [8] "#404040" "#3D3D3D" "#3A3A3A" "#373737" "#343434" "#313131" "#2E2E2E"
## [15] "#2B2B2B" "#292929" "#262626" "#232323" "#202020" "#1D1D1D" "#1A1A1A"
## [22] "#171717" "#141414" "#111111" "#0E0E0E" "#0B0B0B" "#080808" "#050505"
## [29] "#020202" "#000000"
```

```
my.data.array.norm.reshaped_df %>%
  filter(time=="96h",drug=="MEKi",feature %in% c("cell_number","actin.m.eccentricity.tmean")) %>%
  extract(replicate,c("design","bio"),regex="(des\\d)(rep\\d)") %>%
  spread(feature,value) %>%
  ggplot() +
  geom_hex(aes(cell_number, actin.m.eccentricity.tmean), bins = 100) +
  #geom_point(aes(x=cell_number, y=actin.m.eccentricity.tmean)) +
  scale_fill_gradient("", colours = colfunc(100)) +
  theme_classic()
```



16 Supplemetal Figure 3D: Check if features enrich for non-redundant information content of correlated residuals

```
data=my.data.array.norm.reshaped_df %>%
  filter(time=="96h",drug=="MEKi",feature %in% c(goodfeature)) %>%
  extract(replicate,c("design","bio"),regex="(des\\d)(rep\\d)") %>%
  group_by(bio,feature,target,query) %>%
  summarise(value=mean(value,na.rm=T))

#####
# reformat data to fit the bernd scheme of target ~ query ~ feature ~ replicate
#####

test1 = drop_na(data)

test1= acast(data,target ~ query ~ feature ~ bio)
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

ordered_feat=allfeature[allfeature %in% goodfeature]

test1=test1[,ordered_feat,]

# !remove nas from the data!
test1[which(is.na(test1))]=0

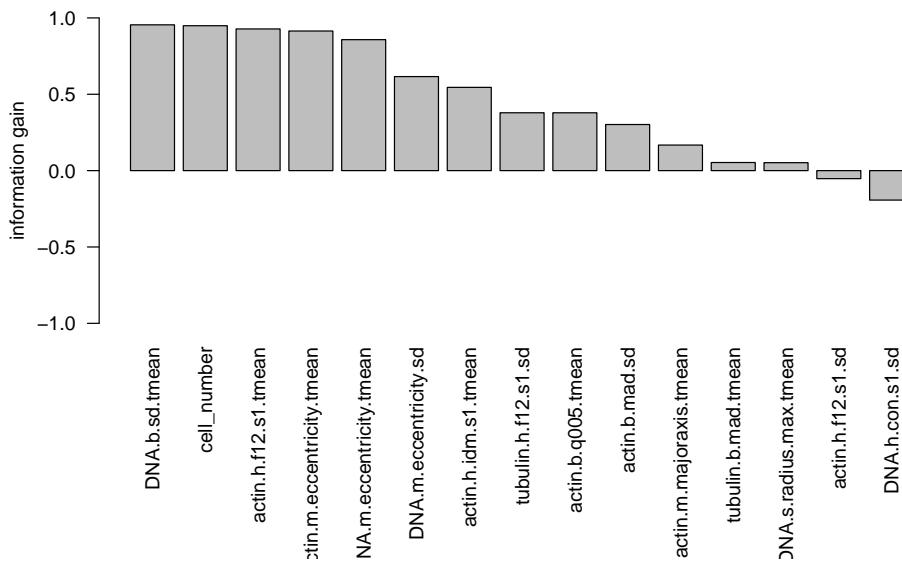
# set new dimensions to the data frame
test=test1
dim(test) = c(prod(dim(test)[1:2]),dim(test)[3:4])

test=aperm(test,c(1,3,2))

#sub sample the data for quicker evaluation
set.seed(5830458)
sample = sample(seq_len(dim(test)[1]), 6000)
subSampleForStabilitySelection = list(D = test[sample,,],
                                      sample = sample,
                                      phenotype = dimnames(test1)[[3]])
#calculate the information gain per selected feature
stabilitySelection = HD2013SGI::HD2013SGIselectByStability(subSampleForStabilitySelection,verbose = T,presel

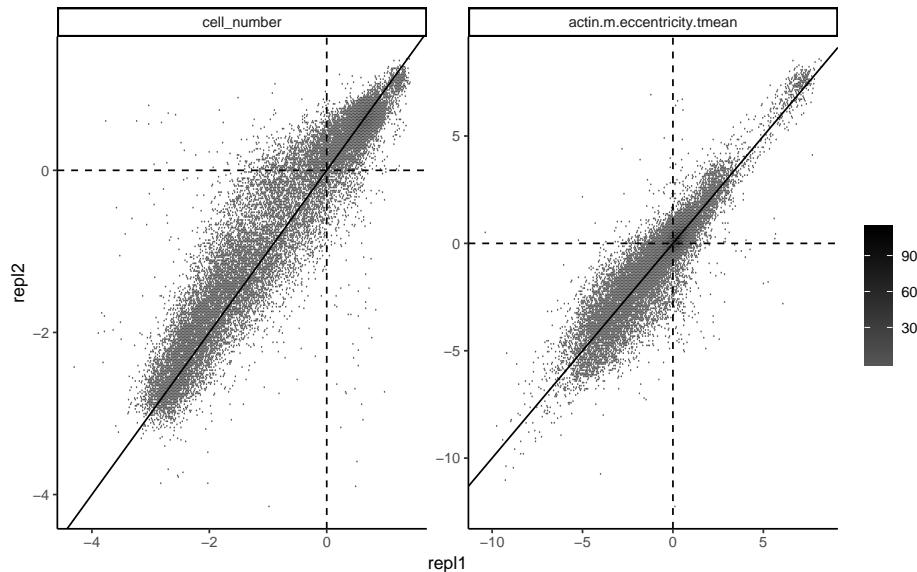
#plot the information gain
par(mar=c(10, 4, 2, 0))
barplot(stabilitySelection$correlation[order(stabilitySelection$correlation,decreasing = T)],
        names.arg=stabilitySelection$selected[order(stabilitySelection$correlation,decreasing = T)],
        ylim=c(-1,1), las=2, ylab="information gain")

```



17 Figure 2C: Biological replicate correlation

```
my.data.array.norm.reshaped_df %>%
  filter(time=="96h", drug=="MEKi", feature %in% c("cell_number", "actin.m.eccentricity.tmean")) %>%
  extract(replicate, c("design", "bio"), regex="(des\\d)(repl\\d)") %>%
  spread(bio, value) %>%
  ggplot(aes(x=repl1, y=repl2)) +
  geom_hex(aes(repl1, repl2), bins = 250) +
  geom_abline(slope = 1, intercept = c(0, 0)) +
  geom_vline(xintercept = 0, lty=2) +
  geom_hline(yintercept = 0, lty=2) +
  scale_fill_gradientn("", colours = colfunc(150)) +
  facet_wrap("feature", scales = "free") +
  theme_classic()
```



18 Figure 2D: dsRNA design replicate correlation

```
data=my.data.array.norm.reshaped_df %>%
  dplyr::filter(time=="96h", drug=="MEKi", feature %in% c(goodfeature)) %>%
  tidyverse::extract(replicate, c("design", "bio"), regex="(des\\d)(repl\\d)") %>%
  tidyverse::spread(design,value) %>%
  group_by(target, query, feature) %>%
  summarise(des1=mean(des1), des2=mean(des2))

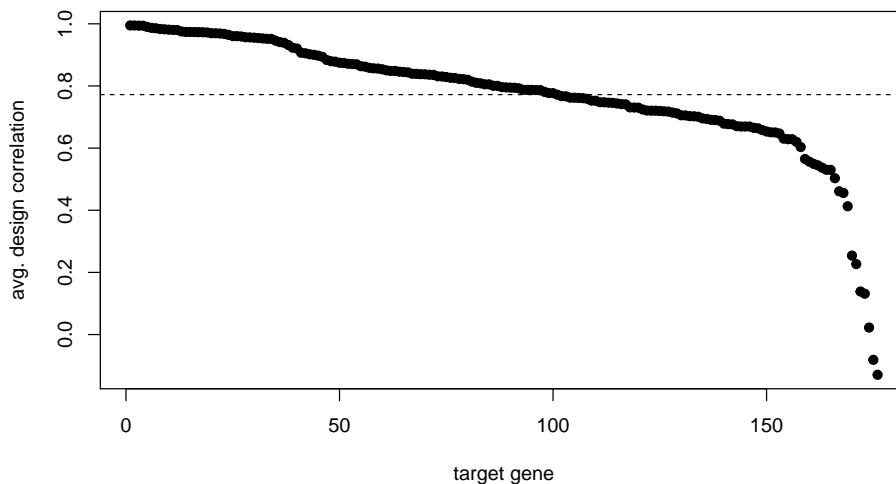
#calculate the correlation between design replicates for all good features separated by queries such that qu
# the target gene design correlations are then averaged over all queries
c_avg_q_all_f <-
  data %>%
  group_by(target, query) %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
summarise(correlation=cor(des1,des2,method="p",use="p")) %>%
group_by(target) %>%
summarise(correlation2=mean(correlation,na.rm=T)) %>%
mutate(target=factor(target,levels=target[order(correlation2,decreasing=T)]))

plot(sort(c_avg_q_all_f$correlation2,decreasing = T),ylab="avg. design correlation",xlab="target gene",pch=19)
abline(h=mean(c_avg_q_all_f$correlation2),col="black",lty=2)
title(main=mean(c_avg_q_all_f$correlation2))
```

0.772156710097175



19 Figure 2E: Biological control separation

```
myarray.norm<-my.data.array.norm_zprime[,,goodfeature]

z_kernel=function(x,sigma=1){
  res=0
  for(i in 1:length(x)){
    for(j in 1:length(x)){
      res=res+exp(-(x[i]-x[j])^2/(2*sigma^2))
    }
  }
  return(res/(length(x)^2))
}

z_factor=function(pos,neg){1-3*(sd(pos)+sd(neg))/(mean(pos)-mean(neg))}

feature<-"cell_number"
plates_that_correlate.idx=c(1:80)
screen<-"S06"

par(mfrow=c(1,2))
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

plates=list()
for(i in plates_that_correlate.idx){
  plates[[i]]=myarray.norm[c("P12","P13"),screen,i,]
}
cur_neg=do.call("rbind.data.frame",plates)
cur_neg$class="neg"

plates=list()
for(i in plates_that_correlate.idx){
  plates[[i]]=myarray.norm[c("H12","H13"),screen,i,]
}
cur_pos=do.call("rbind.data.frame",plates)
cur_pos$class="pos"

dataset=rbind(cur_neg,cur_pos) %>% drop_na()
lda_anal=lda(class~,data = dataset)

plates=list()
for(i in plates_that_correlate.idx){
  cur_pos=myarray.norm[c("H12","H13"),screen,i,]
  cur_neg=myarray.norm[c("P12","P13"),screen,i,]
  neg_P=apply(cur_neg,1,function(x){sum(x*lda_anal$scaling)})
  pos_P=apply(cur_pos,1,function(x){sum(x*lda_anal$scaling)})
  plates[[i]]=cbind(i,pos_P,neg_P,z_factor(pos_P,neg_P))
}
multi=cur_z=do.call("rbind.data.frame",plates)

pos=cur_z[,2]
neg=cur_z[,3]
par(mfrow = c(2,3))
plot(cur_z[,1],cur_z[,2],pch=16,col="green",ylim=c(-30,15),ylab="weighted sum of features",xlab="plate-index")
points(cur_z[,1],cur_z[,3],pch=16,col="red")

plot(2,2,xlim=c(-30,15),ylim=c(0,1),sub=paste("Z'-factor =",round(mean(cur_z[,4]),na.rm = T),digits=3)),ylab=
lines(density(cur_z[,2],na.rm = T),col="green",lwd=2)
lines(density(cur_z[,3],na.rm = T),col="red",lwd=2)

plot(cur_z[,1],cur_z[,4],pch=16,col="black",ylim=c(-1,1),main=paste("Multifactorial-Z' =",round(mean(cur_z[,4]),na.rm = T)))
abline(h=mean(cur_z[,4]),na.rm = T))

#####
plates=list()
for(i in plates_that_correlate.idx){
  pos_P=cur_pos=myarray.norm[c("H12","H13"),screen,i,"cell_number"]
  neg_P=cur_neg=myarray.norm[c("P12","P13"),screen,i,"cell_number"]
  plates[[i]]=cbind(i,pos_P,neg_P,z_factor(pos_P,neg_P))
}
uni=cur_z=do.call("rbind.data.frame",plates)

```

```

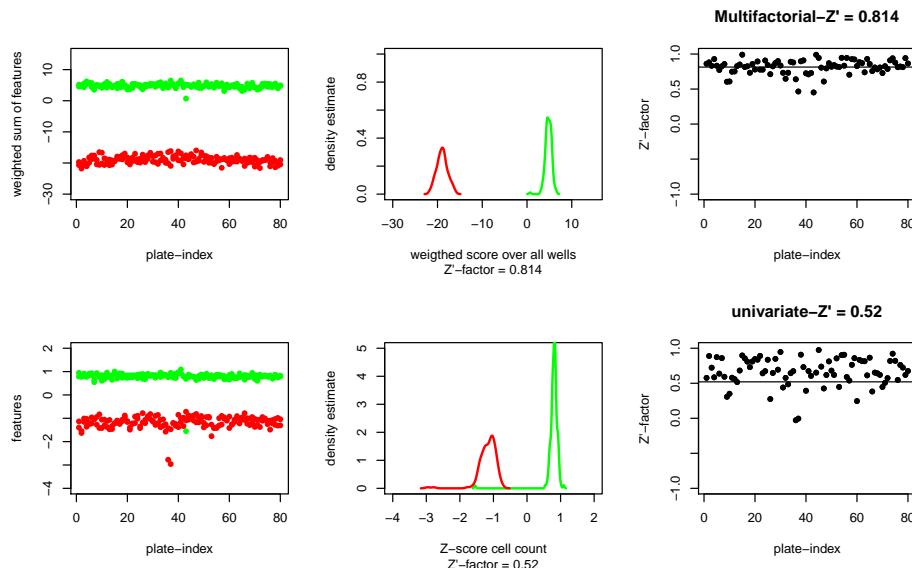
pos=cur_z[,2]
neg=cur_z[,3]

plot(cur_z[,1],cur_z[,2],pch=16,col="green",ylim=c(-4,2),ylab="features",xlab="plate-index")
points(cur_z[,1],cur_z[,3],pch=16,col="red")

plot(2,2,xlim=c(-4,2),ylim=c(0,5),sub=paste("Z'-factor =",round(mean(cur_z[,4],na.rm = T),digits=3)),ylab="density estimate")
lines(density(cur_z[,2],na.rm = T),col="green",lwd=2)
lines(density(cur_z[,3],na.rm = T),col="red",lwd=2)

plot(cur_z[,1],cur_z[,4],pch=16,col="black",ylim=c(-1,1),main=paste("univariate-Z' =",round(mean(cur_z[,4],na.rm = T),digits=3)))
abline(h=mean(cur_z[,4],na.rm = T))

```



20 Figure 2F: Example interaction heatmap

```

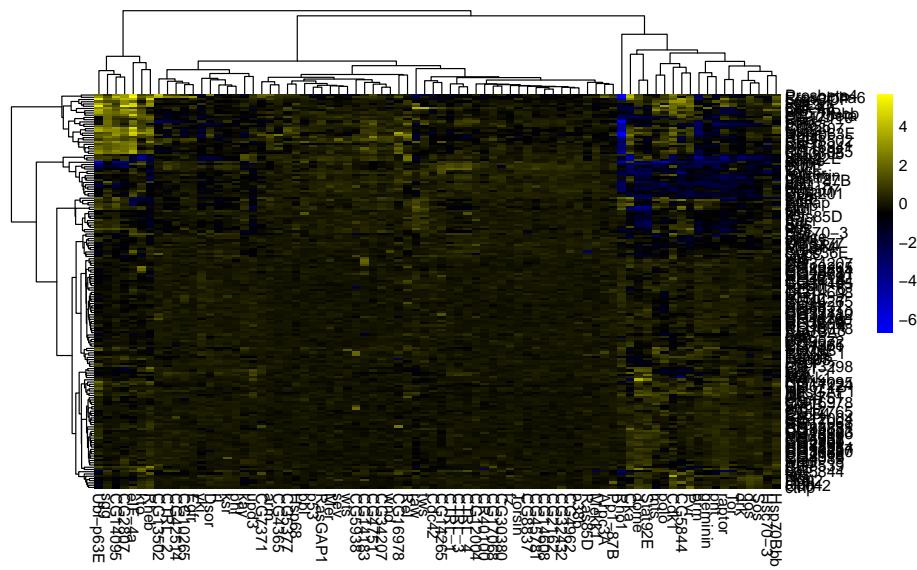
test<-melt(interaction_scores) %>%tbl_df()
colnames(test) <- c("target","query","feature","time","drug","replicate","piscore")
test_sub <- test %>%
  filter(feature %in% goodfeature) %>%
  group_by(target,query,feature,time,drug) %>%
  summarise(piscore=mean(piscore,na.rm=T)) %>%
  ungroup()

eccentr_DMSO_96 <- test_sub %>%
  filter(time=="96h",
        feature=="actin.m.eccentricity.tmean",
        drug=="DMSO"
      ) %>%
  dplyr::select(target,query,piscore) %>%

```

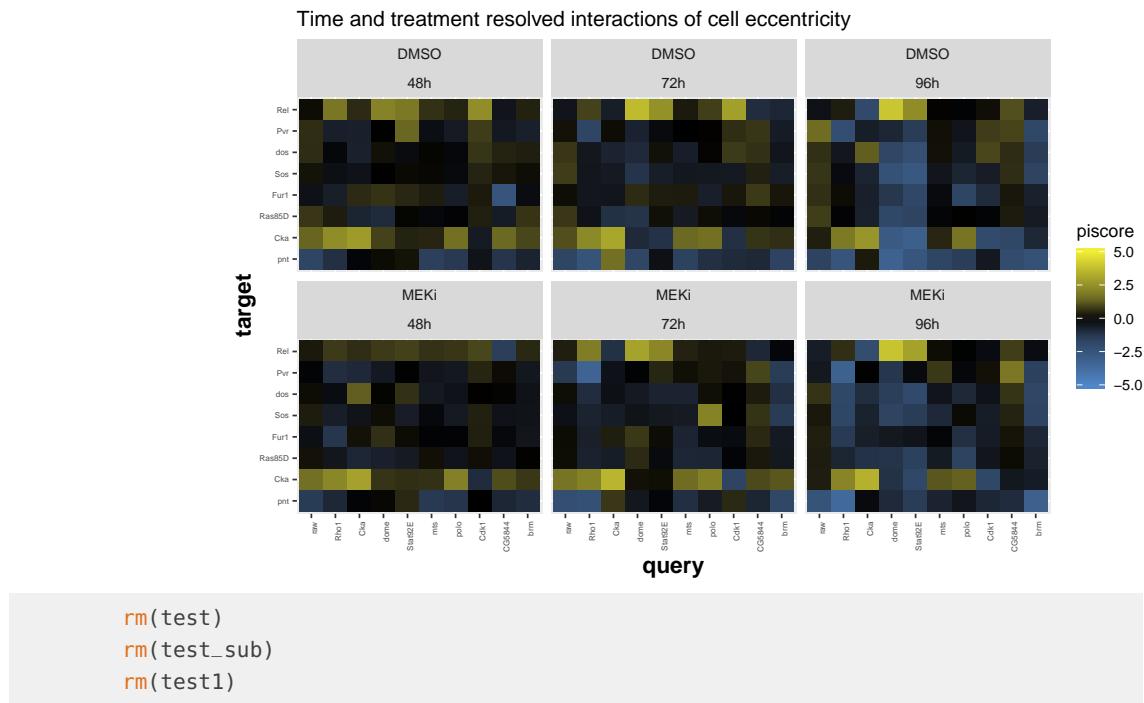
Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
drop_na() %>%
  spread(query,piscore) %>%
  `row.names<-`(.$.target) %>%
  dplyr::select_if(is.numeric) %>%
  heatmap(cluster_rows=TRUE, cluster_cols=TRUE,color = colorRampPalette(c("blue", "black", "yellow"))(1000))
```



```
test_sub %>%
  filter(
    feature %in% c("actin.m.eccentricity.tmean"),
    !grepl("CTRL",query),
    !grepl("RLUC",target)
  ) %>%
  dplyr::select(target,query,piscore,drug,time) %>%
  mutate(target=factor(target,levels = eccentr_DMSO_96$tree_row$labels[eccentr_DMSO_96$tree_row$order])) %>%
  mutate(query=factor(query,levels = eccentr_DMSO_96$tree_col$labels[eccentr_DMSO_96$tree_col$order])) %>%
  filter(query %in% c("raw","Rhol","Cka","dome","Stat92E","mts","polo","Cdk1","CG5844","brm"),target %in% c(
  ggplot(aes(x=query,y=target,fill=piscore)) +
  facet_wrap(~drug+time) +
  geom_raster() +
  scale_fill_gradientn(na.value = 'black',colours = c("#5087C8","black","#F2EE35"),
                       guide = "colorbar",values = rescale(c(-5,-1.5,0,1.5,5)), limits=c(-5,5)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1) ) +
  theme(axis.text=element_text(size=5),axis.title=element_text(size=14,face="bold"))+
  ggttitle("Time and treatment resolved interactions of cell eccentricity")
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



21 Supplementary Figure 5A-C: Feature comparison to Fischer et al. 2015

```

#load pi score matrix from Fischer et al., Elife 2015
data("pimatrix", package="DmelSGI")
fisheretal=melt(pimatrix$D) %>%tbl_df()

#loadpi-score matrix from Heigwer et al.
heigweretal=melt(interaction_scores) %>%tbl_df()
names(heigweretal)=c("target", "query", "feature", "time", "treatment", "replicate", "value")

#identify common query genes
commonqueries=levels(heigweretal$query)[levels(heigweretal$query) %in% levels(fisheretal$query)]
#it is actually only RasGAP1

#identify common target genes
commontargets=levels(heigweretal$target)[levels(heigweretal$target) %in% levels(fisheretal$target)]

#filter the data set for the common data points to Fischer et al
heigwercommon=heigweretal %>%
  filter(treatment=="DMSO", time=="72h", query=="RasGAP1", target %in% commontargets) %>% #, feature=="DNA.m.major"
  dplyr::select(target, query, feature, heigwer=value) %>%
  group_by(target, query, feature) %>%
  summarise(heigwer=mean(heigwer, na.rm=T)) %>%

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
ungroup()

#filter the data set for the common data points to Heigwer et al
fishercommon=fisheretal %>% filter(query=="RasGAP1",target %in% commontargets) %>% #phenotype=="10x.meanNonmitotic.cell.0.s.area"
  dplyr::select(target,query,phenotype,fisher=value) %>%
  group_by(target,query,phenotype) %>%
  summarise(fisher=mean(fisher,na.rm=T))%>%
  ungroup()

#join the commonalities
common=left_join(heigwercommon,fishercommon)

#analyze linear dependency between two features of the different screens by linear modeling and anova
#one needs to note that these interdependencies are actually in interaction score space! meaning the a significant_dependencies<-
  common %>%
  group_by(feature,phenotype) %>%
  do(co=lm(heigwer~fisher,data=.)) %>%
  tidy(co) %>%
  ungroup() %>%
  filter(term=="fisher") %>%
  mutate(fdr=p.adjust(p.value)) %>%
  filter(fdr<0.05) %>%
  arrange(fdr)

#we found that the following features can be translated from heigwer et al to fisher et al

"DNA.s.perimeter.tmean" = "10x.meanNonmitotic.cell.0.s.area"
"DNA.s.radius.sd.tmean" = "4x.areaNucAll"
"DNA.m.eccentricity.tmean" = "10x.meanNonmitotic.nucleus.DAPI.m.majoraxis"
"DNA.b.mad.tmean" = "4x.areapH3All"
"DNA.h.asm.s1.tmean" = "4x.count"

#as control we plot these pairwise dependencies
a<-common %>%
  filter(feature=="DNA.m.eccentricity.tmean",phenotype=="10x.meanNonmitotic.nucleus.DAPI.m.majoraxis") %>%
  ggplot(aes(x=heigwer,y=fisher)) +
  geom_point() +
  geom_smooth(method="rlm",se=F)+
  theme_classic() +
  xlab("Heigwer: DNA.m.eccentricity.tmean") +
  ylab("Fischer: 10x.meanNonmitotic.nucleus.DAPI.m.majoraxis") +
  ggtitle(paste0('FDR = ',significant_dependencies %>% filter(feature=="DNA.m.eccentricity.tmean",phenotype==

b<-common %>%
  filter(feature=="DNA.b.mad.tmean",phenotype=="4x.areapH3All") %>%
  ggplot(aes(x=heigwer,y=fisher)) +
  geom_point() +
  geom_smooth(method="rlm",se=F)+
  theme_classic() +
  xlab("Heigwer: DNA.b.mad.tmean") +
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

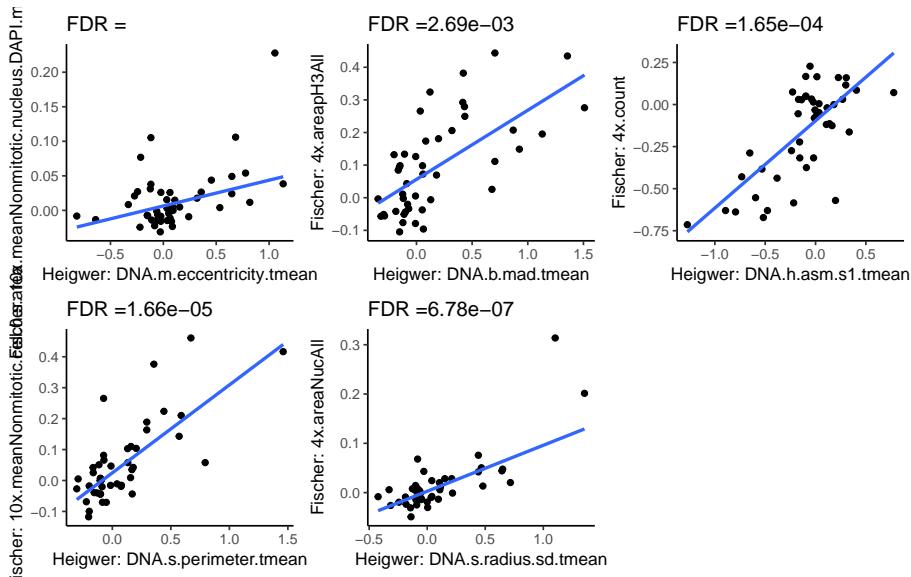
```
ylab("Fischer: 4x.areapH3All") +
ggtitle(paste0('FDR =',significant_dependencies %>% filter(feature=="DNA.b.mad.tmean",phenotype=="4x.areapH3All")))

c<-common %>%
filter(feature=="DNA.h.asm.s1.tmean",phenotype=="4x.count") %>%
ggplot(aes(x=heigwer,y=fisher)) +
geom_point() +
geom_smooth(method="rlm",se=F) +
theme_classic() +
xlab("Heigwer: DNA.h.asm.s1.tmean") +
ylab("Fischer: 4x.count") +
ggtitle(paste0('FDR =',significant_dependencies %>% filter(feature=="DNA.h.asm.s1.tmean",phenotype=="4x.count")))

d<-common %>%
filter(feature=="DNA.s.perimeter.tmean",phenotype=="10x.meanNonmitotic.cell.0.s.area") %>%
ggplot(aes(x=heigwer,y=fisher)) +
geom_point() +
geom_smooth(method="rlm",se=F) +
theme_classic() +
xlab("Heigwer: DNA.s.perimeter.tmean") +
ylab("Fischer: 10x.meanNonmitotic.cell.0.s.area") +
ggtitle(paste0('FDR =',significant_dependencies %>% filter(feature=="DNA.s.perimeter.tmean",phenotype=="10x.meanNonmitotic.cell.0.s.area")))

e<-common %>%
filter(feature=="DNA.s.radius.sd.tmean",phenotype=="4x.areaNucAll") %>%
ggplot(aes(x=heigwer,y=fisher)) +
geom_point() +
geom_smooth(method="rlm",se=F) +
theme_classic() +
xlab("Heigwer: DNA.s.radius.sd.tmean") +
ylab("Fischer: 4x.areaNucAll") +
ggtitle(paste0('FDR =',significant_dependencies %>% filter(feature=="DNA.s.radius.sd.tmean",phenotype=="4x.areaNucAll")))

a+b+c+d+e
```



22 Supplementary Figure 5D-G: Feature comparison to Horn et al. 2011

```
#load the pi-score data produced by Horn et al. 2011
#library(RNAinteractMAPK)
data("Dmel2PPMAPK", package="RNAinteractMAPK")
PI <- RNAinteract::getData(Dmel2PPMAPK, type="pi", format="targetMatrix", screen="mean", withoutgroups = c(""))

hornetal<-melt(PI)
names(hornetal)=c("target","query","repl","phenotype","horn")

hornetal %>%tbl_df()%>%dplyr::select(-repl)

#identify common query genes
commonqueries=levels(heigweretal$query)[levels(heigweretal$query) %in% levels(hornetal$query)]
#it is actually only RasGAP1

#identify common target genes
commontargets=levels(heigweretal$target)[levels(heigweretal$target) %in% levels(hornetal$target)]

#filter the data set for the common data points to Horn et al
heigwercommon=heigweretal %>%
  filter(treatment=="DMSO", time=="72h", query %in% commonqueries, target %in% commontargets) %>% #, feature=="DMSO"
  dplyr::select(target, query, feature, heigwer=value) %>%
  group_by(target, query, feature) %>%
  summarise(heigwer=mean(heigwer, na.rm=T)) %>%
  ungroup()

#filter the data set for the common data points to Heigwer et al
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
horncommon=hornetal %>% filter(query %in% commonqueries,target %in% commontargets) %>% #phenotype=="10x.mean"
  dplyr::select(target,query,phenotype,horn) %>%
  group_by(target,query,phenotype) %>%
  summarise(horn=mean(horn,na.rm=T))%>%
  ungroup()

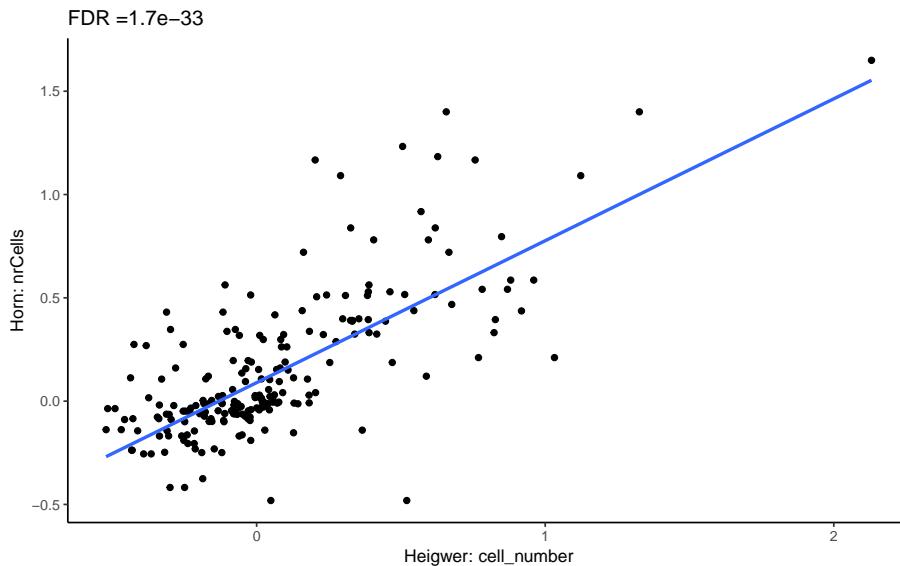
#join the commonalities
common=left_join(heigwercommon,horncommon)

#analyze linear dependency between two features of the different screens by linear modeling and anova
#one needs to note that these interdependencies are actually in interaction score space! meaning the a significant_dependencies<-
  common %>%
  group_by(feature,phenotype) %>%
  do(co=lm(heigwer~horn,data=.)) %>%
  tidy(co) %>%
  ungroup() %>%
  filter(term=="horn") %>%
  mutate(fdr=p.adjust(p.value)) %>%
  filter(fdr<0.05) %>%
  arrange(fdr)

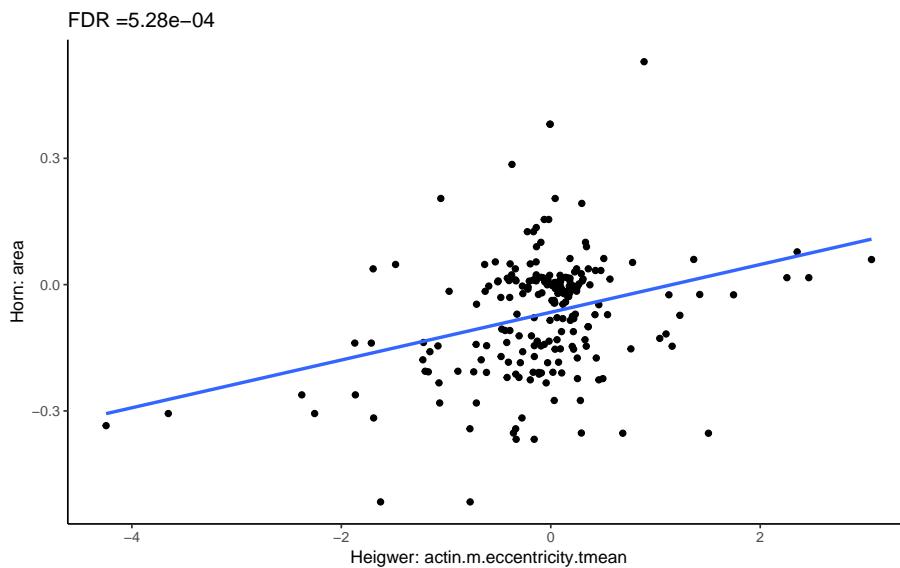
"cell_number"="nrCells"
"actin.m.eccentricity.tmean" = "area"
"DNA.b.mean.tmean" = "intensity"

#plot the feature mappings as scatterplot
common %>%
  filter(feature=="cell_number",phenotype=="nrCells") %>%
  ggplot(aes(x=heigwer,y=horn)) +
  geom_point() +
  geom_smooth(method="rlm",se=F)+ 
  theme_classic() +
  xlab("Heigwer: cell_number") +
  ylab("Horn: nrCells") +
  ggtitle(paste0('FDR =',significant_dependencies %>% filter(feature=="cell_number",phenotype=="nrCells") %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



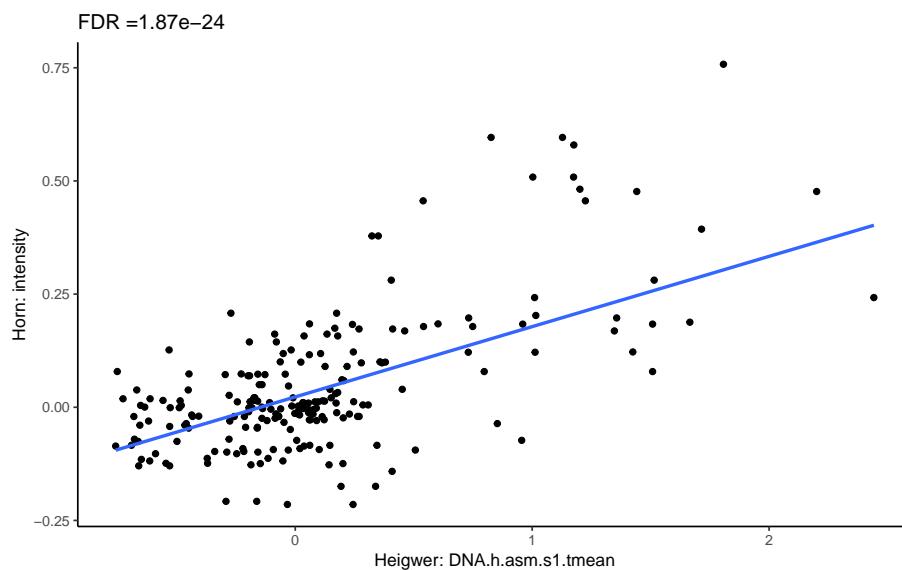
```
common %>%
  filter(feature=="actin.m.eccentricity.tmean",phenotype=="area") %>%
  ggplot(aes(x=heigwer,y=horn)) +
  geom_point() +
  geom_smooth(method="rlm",se=F) +
  theme_classic() +
  xlab("Heigwer: actin.m.eccentricity.tmean") +
  ylab("Horn: area") +
  ggtitle(paste0('FDR =',significant_dependencies %>% filter(feature=="actin.m.eccentricity.tmean",phenotype==
```



```
common %>%
  filter(feature=="DNA.h.asm.s1.tmean",phenotype=="intensity") %>%
  ggplot(aes(x=heigwer,y=horn)) +
  geom_point() +
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
geom_smooth(method="rlm",se=F)+  
theme_classic() +  
xlab("Heigwer: DNA.h.asm.s1.tmean") +  
ylab("Horn: intensity") +  
ggtitle(paste0('FDR =',significant_dependencies %>% filter(feature=="DNA.h.asm.s1.tmean",phenotype=="intensity")))
```

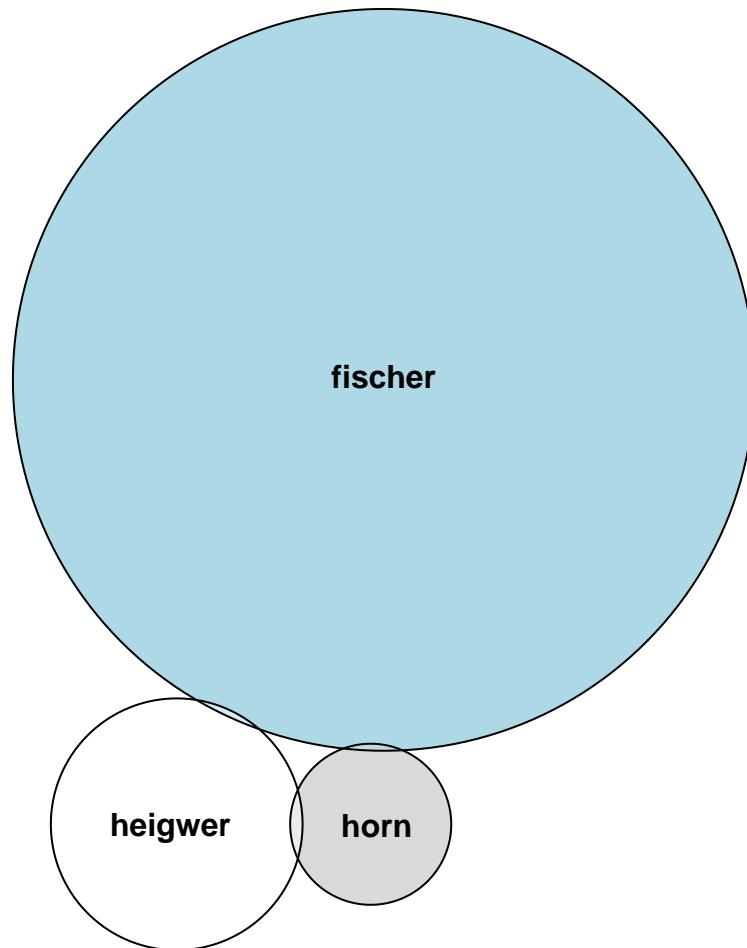


```
#get all interactions screened by Heigwer et al.
```

```
heigwer_combis<-heigweretal %>%  
dplyr::select(target,query) %>%  
dplyr::distinct() %>%  
filter(!grepl("CTRL",query)) %>%  
mutate(target=as.character(target),query=as.character(query)) %>%  
group_by(target,query) %>%  
filter(target != query) %>%  
mutate(tmp=paste(sort(c(target,query)),collapse="_")) %>%  
.tmp %>%  
unique()  
  
horn_combis<-hornetal %>%  
dplyr::select(target,query) %>%  
dplyr::distinct() %>%  
filter(!grepl("CTRL",query)) %>%  
mutate(query=gsub(x = query,pattern="Gap1",replacement="RasGAP1"),target=gsub(x = target,pattern="Gap1",replacem  
mutate(target=as.character(target),query=as.character(query)) %>%  
group_by(target,query) %>%  
filter(target != query) %>%  
mutate(tmp=paste(sort(c(target,query)),collapse="_")) %>%  
.tmp %>%  
unique()  
  
fischer_combis<-fisheretal %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
dplyr::select(target,query) %>%  
dplyr::distinct() %>%  
filter(!grepl("CTRL",query)) %>%  
mutate(target=as.character(target),query=as.character(query)) %>%  
group_by(target,query) %>%  
filter(target != query) %>%  
mutate(tmp=paste(sort(c(target,query)),collapse="_")) %>%  
. $tmp %>%  
unique()  
  
fit1 <- euler(c("heigwer" = 10273, "horn" = 4107, "fischer" = 90636,  
"heigwer&horn" = 114, "heigwer&fischer" = 57,  
"horn&fischer" = 51, "heigwer&horn&fischer" = 6))  
  
plot(fit1,counts = T)
```



```
rm(heigweretal)
```

23 Supplementary Figure 6: Quality control metrics of gene-gene combinatorial screening

```

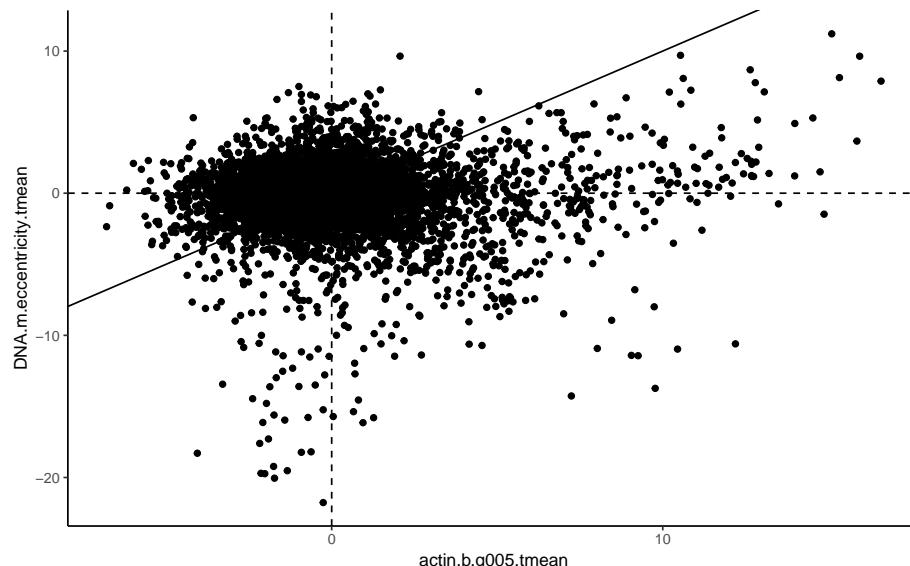
rm(Interactions)
#plot uncorrelating features pi-scores

colfunc <- colorRampPalette(c("#555555", "black"))

data = data_df %>%
  filter(feature %in% goodfeature, time=="96h", drug=="MEKi") %>%
  dplyr::select(-fdr) %>%
  spread(feature,piscore) %>%
  drop_na()

data %>%
  ggplot(aes(x=actin.b.q005.tmean, y=DNA.m.eccentricity.tmean)) +
  geom_point() +
  geom_abline(slope = 1,intercept = c(0,0)) +
  geom_vline(xintercept = 0,lty=2) +
  geom_hline(yintercept = 0,lty=2) +
  theme_classic()

```



```

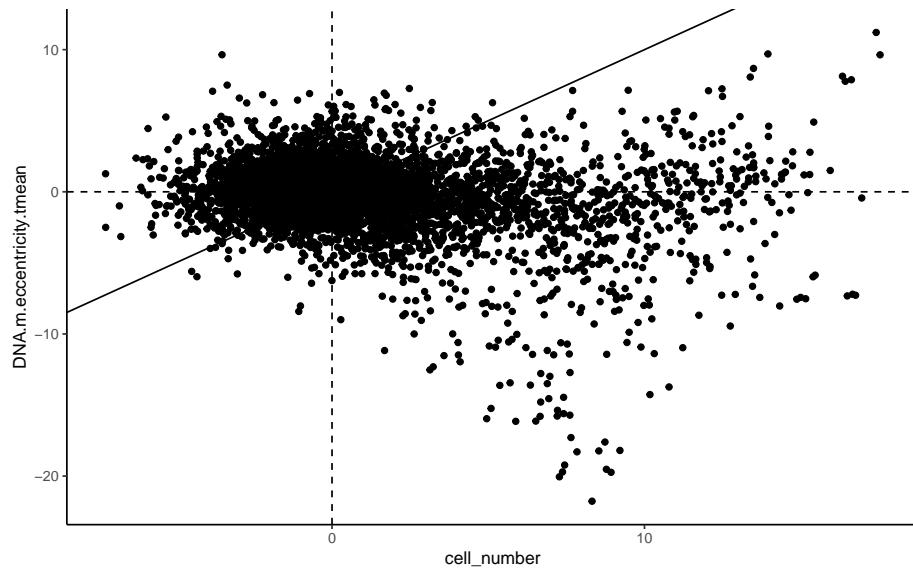
cor(data$actin.b.q005.tmean,data$DNA.m.eccentricity.tmean,use="p",method="p")
## [1] 0.0360771

#plot uncorrelating features pi-scores
#Cell number vs nuclear eccentricity
data %>%
  ggplot(aes(x=cell_number, y=DNA.m.eccentricity.tmean)) +
  geom_abline(slope = 1,intercept = c(0,0)) +

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
geom_vline(xintercept = 0,lty=2) +
geom_hline(yintercept = 0,lty=2) +
geom_point() +
theme_classic()
```



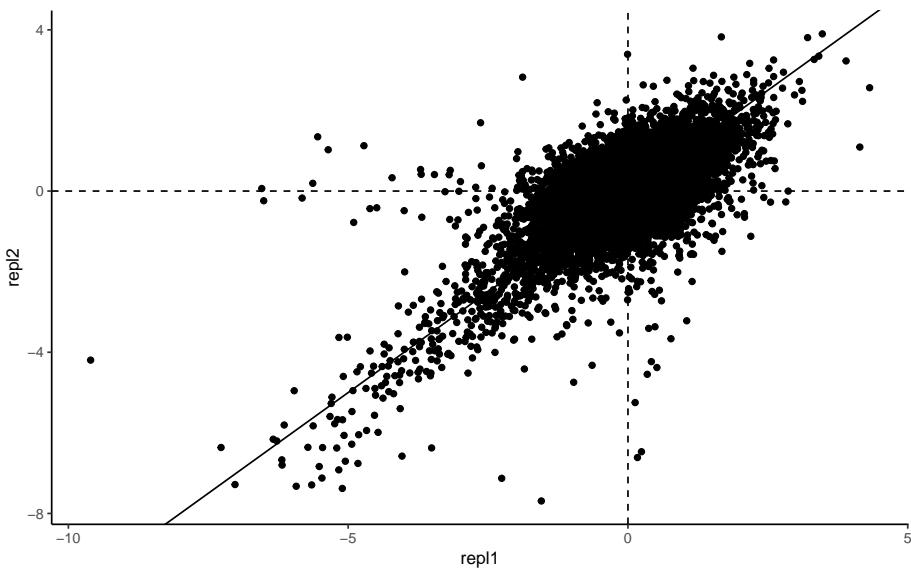
```
cor(data$cell_number,data$DNA.m.eccentricity.tmean,use="p",method="p")
## [1] -0.2182899

#plot correlating features pi-scores

interaction_scores_df<-melt(interaction_scores,varnames=c("target","query","feature","time","drug","replicate"))

interaction_scores_df %>%
  filter(feature == "DNA.m.eccentricity.tmean", time=="96h",drug=="MEKi") %>%
  extract(replicate,c("design","bio"),regex="(des\\d)(repl\\d)") %>%
  spread(bio,value) %>%
  ggplot(aes(x=repl1,y=repl2))+
  geom_abline(slope = 1,intercept = c(0,0)) +
  geom_vline(xintercept = 0,lty=2) +
  geom_hline(yintercept = 0,lty=2) +
  geom_point() +
  theme_classic()
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



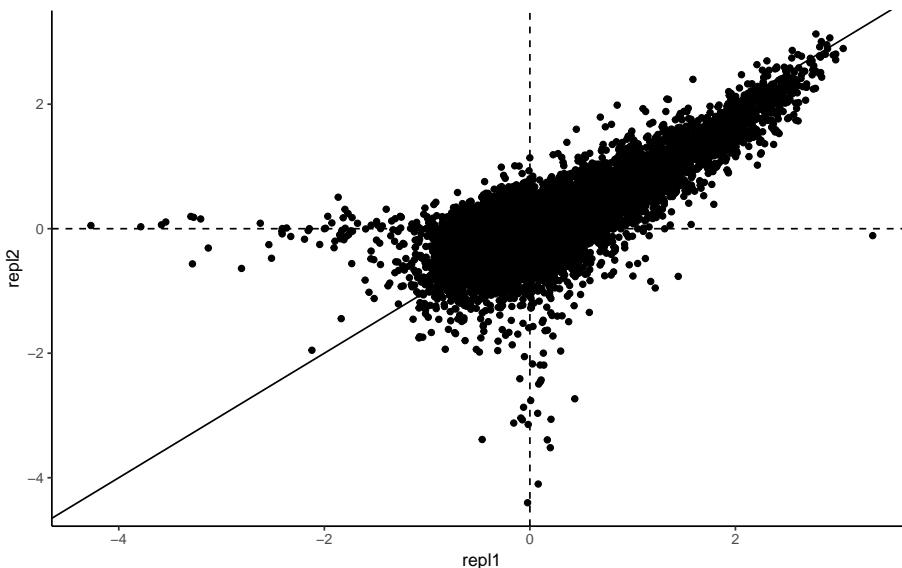
```
d=interaction_scores_df %>%
  filter(feature == "DNA.m.eccentricity.tmean", time=="96h",drug=="MEKi") %>%
  extract(replicate,c("design","bio"),regex="(des\\d)(rep\\d)") %>%
  spread(bio,value)

cor(d$repl1,d$repl2,use="p",method="p")
## [1] 0.6212475

#plot correlating features pi-scores

interaction_scores_df %>%
  filter(feature == "cell_number", time=="96h",drug=="MEKi") %>%
  extract(replicate,c("design","bio"),regex="(des\\d)(rep\\d)") %>%
  spread(bio,value) %>%
  ggplot(aes(x=repl1,y=repl2))+
  geom_abline(slope = 1,intercept = c(0,0)) +
  geom_vline(xintercept = 0,lty=2) +
  geom_hline(yintercept = 0,lty=2) +
  geom_point() +
  theme_classic()
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



```
d=interaction_scores_df %>%
  filter(feature == "cell_number", time=="96h",drug=="MEKi") %>%
  extract(replicate,c("design","bio"),regex="(des\\d)(rep\\d)") %>%
  spread(bio,value)

cor(d$repl1,d$repl2,use="p",method="p")
## [1] 0.7603566

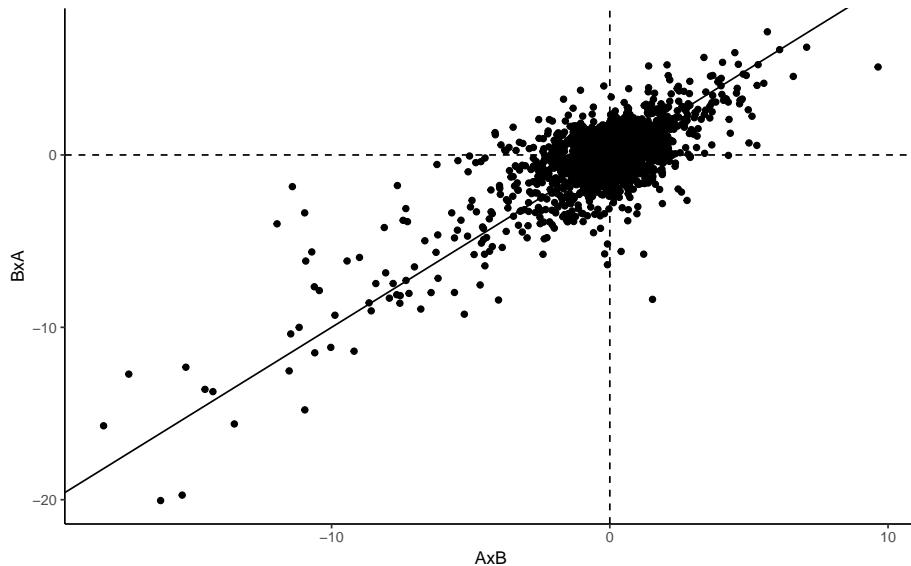
#plot reversibility of interactions DNA.m.eccentricity.tmean

data=data_df %>%
  dplyr::filter(time=='96h',drug=='MEKi',feature=="DNA.m.eccentricity.tmean") %>%
  dplyr::select(target,query,piscore) %>%
  filter(target %in% unique(.query),query %in% unique(.target)) %>%
  arrange(target,query) %>%
  mutate(target=as.character(target),query=as.character(query)) %>%
  group_by(target,query) %>%
  filter(target != query) %>%
  mutate(tmp=paste(sort(c(target,query)),collapse="_")) %>%
  group_by(tmp) %>%
  mutate(rep=c("AxB","BxA")) %>%
  dplyr::select(piscore,tmp,rep) %>%
  spread(rep,piscore) %>%
  ungroup()

data %>%
  ggplot(aes(x=AxB,y=BxA))+
  geom_point() +
  geom_abline(slope = 1,intercept = c(0,0)) +
  geom_vline(xintercept = 0,lty=2) +
  geom_hline(yintercept = 0,lty=2) +
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
theme_classic()
```

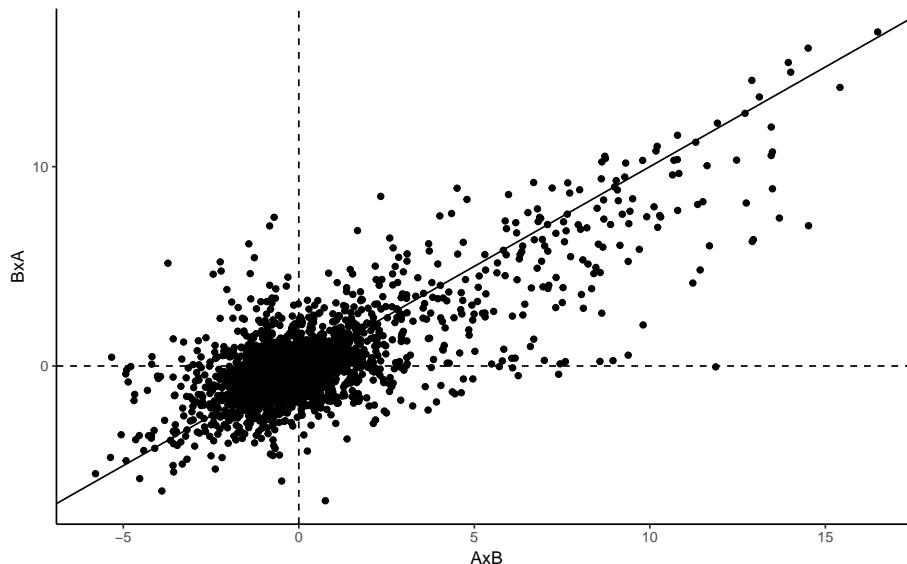


```
cor(data$AxB,data$BxA,use = "p")
## [1] 0.7479145

#plot reversibility of interactions cell number

data=data_df %>%
  dplyr::filter(time=='96h',drug=='MEKi',feature=="cell_number") %>%
  dplyr::select(target,query,piscore) %>%
  filter(target %in% unique(.query),query %in% unique(.target)) %>%
  arrange(target,query) %>%
  mutate(target=as.character(target),query=as.character(query)) %>%
  group_by(target,query) %>%
  filter(target != query) %>%
  mutate(tmp=paste(sort(c(target,query)),collapse="_")) %>%
  group_by(tmp) %>%
  mutate(rep=c("AxB","BxA")) %>%
  dplyr::select(piscore,tmp,rep) %>%
  spread(rep,piscore) %>%
  ungroup()

data %>%
  ggplot(aes(x=AxB,y=BxA))+
  geom_point() +
  geom_abline(slope = 1,intercept = c(0,0)) +
  geom_vline(xintercept = 0,lty=2) +
  geom_hline(yintercept = 0,lty=2) +
  theme_classic()
```



```
cor(data$AxB,data$BxA,use = "p")
## [1] 0.7643091
```

24 Comparison of different statistical measures

First we calculate and save differential interaction estimates for all the different statistical methods.

```
sample_data=interaction_scores[,goodfeature,,,] %>% melt() %>%tbl_df()
names(sample_data)=c("target","query","feature","time","treatment","replicate","piscore")

sample_data <- sample_data %>% filter(!grepl("CTRL",query),!grepl("RLUC",target))

my_rlm<-function(x){
  mdl=rlm(data=x,piscore~time+treatment)
  resi=mdl$residuals %>% sum(.^2)
  vari=x %>% group_by(time,treatment) %>% summarise(v=var(piscore,na.rm=T)) %>% .$v %>% sum(na.rm=T)
  timesig=f.robttest(mdl,var = "time")$p.value
  treatsig=f.robttest(mdl,var = "treatmentMEKi")$p.value
  res=cbind(c(timesig,treatsig),mdl$coefficients[-1],resi,vari)
  rownames(res)=c("time","treatmentMEKi")
  colnames(res)=c("pval","estimate","residuals","sumofvariance")
  return(res)
}

analyzed=sample_data %>%
  group_by(feature,treatment,target,query) %>%
  do(mutate(.,time=sample(time))) %>%
  ungroup() %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
group_by(target,query,feature) %>%
summarise(pi_mean=mean(piscore,na.rm=T)) %>%
ungroup()

analyzed2=sample_data %>%
  group_by(feature,treatment,target,query) %>%
  do(mutate(.,time=sample(time))) %>%
  ungroup() %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  group_by(target,query,feature) %>%
  do(mdl=my_rlm(.)) %>%
  tidy(mdl) %>%
  ungroup() %>%
  rename(term=.rownames`)

analyzed %<>% left_join(analyzed2) %>% group_by(feature) %>% mutate(fdr=p.adjust(pval,method = "BH")) %>% ungroup()

analyzed=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  group_by(target,query,feature) %>%
  summarise(pi_mean=mean(piscore,na.rm=T)) %>%
  ungroup()

analyzed2=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  group_by(target,query,feature) %>%
  do(mdl=my_rlm(.)) %>%
  tidy(mdl) %>%
  ungroup() %>%
  rename(term=.rownames`)

analyzed %<>% left_join(analyzed2) %>% group_by(feature) %>% mutate(fdr=p.adjust(pval,method = "BH")) %>% ungroup()

my_rlm_wotime<-function(x){
  mdl=r lm(data=x,piscore~treatment)
  resi=mdl$residuals %>% sum(.^2)
  vari=x %>% group_by(treatment) %>% summarise(v=var(piscore,na.rm=T)) %>% .$v %>% sum(na.rm=T)
  treatsig=f.robusttest(mdl,var = "treatmentMEKi")$p.value
  res=cbind(c(treatsig),mdl$coefficients[-1],resi,vari)
  rownames(res)=c("treatmentMEKi")
  colnames(res)=c("pval","estimate","residuals","sumofvariance")
  return(res)
}

analyzed=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  filter(time==96) %>%
  group_by(target,query,feature) %>%
  summarise(pi_mean=mean(piscore,na.rm=T)) %>%
  ungroup()
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
analyzed2=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  filter(time==96) %>%
  group_by(target,query,feature) %>%
  do(mdl=my_rlm_wotime(.)) %>%
  tidy(mdl) %>%
  ungroup() %>%
  rename(term=`.rownames`)

analyzed %<-% left_join(analyzed2) %>% group_by(feature) %>% mutate(fdr=p.adjust(pval,method = "BH")) %>% ungroup()

analyzed=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  group_by(target,query,feature) %>%
  summarise(pi_mean=mean(piscore,na.rm=T)) %>%
  ungroup()

analyzed2=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  group_by(target,query,feature) %>%
  do(mdl=my_rlm_wotime(.)) %>%
  tidy(mdl) %>%
  ungroup() %>%
  rename(term=`.rownames`)

analyzed %<-% left_join(analyzed2) %>% group_by(feature) %>% mutate(fdr=p.adjust(pval,method = "BH")) %>% ungroup()

analyzed2=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  filter(time==96) %>%
  group_by(target,query,feature) %>%
  do(mdl=t.test(piscore~treatment,data = .)) %>%
  tidy(mdl) %>%
  ungroup()

ttested_96h_differentialints<-analyzed2 %>% group_by(feature) %>% mutate(fdr=p.adjust(p.value,method = "BH"))

analyzed=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  group_by(target,query,feature) %>%
  summarise(pi_mean=mean(piscore,na.rm=T)) %>%
  ungroup()

analyzed2=sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  group_by(target,query,feature) %>%
  do(
    model=lm(data = .,piscore~time+treatment)
  ) %>%
  tidy(model) %>%
  ungroup()
```

```

analyzed=left_join(analyzed,analyzed2)

analyzed <- analyzed %>% group_by(feature) %>% mutate(fdr=p.adjust(p.value,method = "BH"))

analyzed <- analyzed %>% filter(!grepl("CTRL",query),!grepl("RLUC",target)) %>% ungroup()

ints_from_limma=list()

for(feat in sample_data %>% pull(feature) %>% unique){

  cell_count_ints_for_limma<-
    sample_data %>%
    filter(time=="96h",feature==feat) %>%
    unite(contrast,treatment,replicate) %>%
    spread(contrast,piscore) %>%
    dplyr::select(-feature,-time)

  design <- cbind(Untreated=c(1,1,1,1,0,0,0,0),Treated=c(0,0,0,0,1,1,1,1))
  fit <- limma::lmFit(cell_count_ints_for_limma[,3:10], design)
  cont.matrix <- limma::makeContrasts(TreatedvsUntreated=Treated-Uncreated, levels=design)
  fit2 <- limma::contrasts.fit(fit, cont.matrix)
  fit2 <- limma::eBayes(fit2)
  cell_count_ints_for_limma$fdr=p.adjust(fit2$p.value,method="BH")

  ints_from_limma[[feat]]=cell_count_ints_for_limma[,c(1,2,11)]

}

analyzed <- bind_rows(ints_from_limma,.id = "feature")

```

25 Figure 3B: comparison of different statistical methods with regards to their power

```

#load various gene annotation like pathway associations or signaling function, plus the grouping of different
data('query_annotation', package='MODIFIdata')
data('target_annotation', package='MODIFIdata')
data('grouping', package='MODIFIdata')

fdrt=0.1

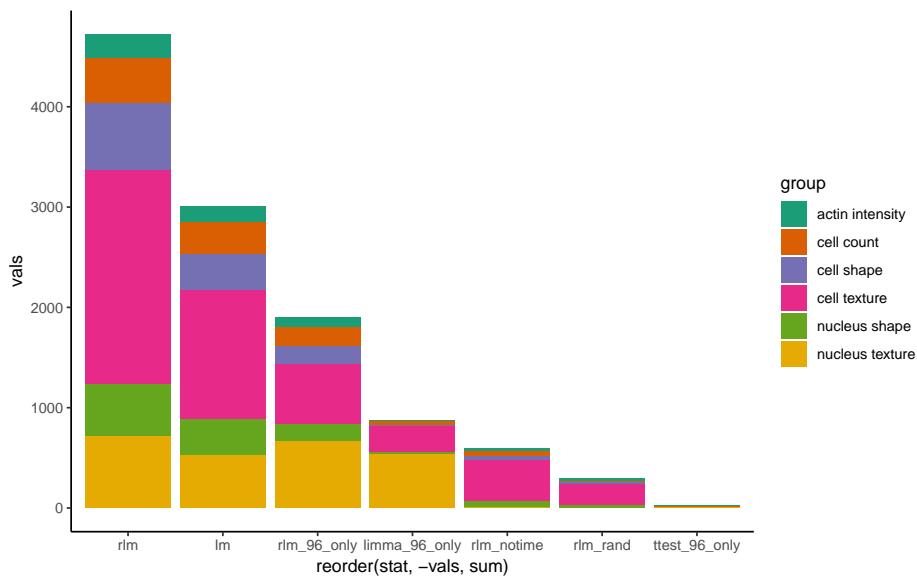
data('time_treatment_rlm_model_analyzed_interactions', package='MODIFIdata')
rlm<-analyzed %>% filter(term=="treatmentMEKi",fdr<fdrt) %>% group_by(feature) %>% summarise(n()) %>% dplyr::
total<-analyzed %>% filter(term=="treatmentMEKi",fdr<1) %>% group_by(feature) %>% summarise(n()) %>% dplyr:::

data('time_treatment_rlm_model_analyzed_interactions_shuffled', package='MODIFIdata')

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
rlm_rand<-rlm_shuffled %>% filter(term=="treatmentMEKi",fdr<fdrt) %>% group_by(feature) %>% summarise(n()) %>%  
  
data('time_treatment_lm_model_analyzed_notscaled!_interactions', package='MODIFIdata')  
lm<-lm_analyzed %>% filter(term=="treatmentMEKi",fdr<fdrt) %>% group_by(feature) %>% summarise(n()) %>% dplyr::  
  
data('treatment_rlm_wotime_model_analyzed_time487296_interactions', package='MODIFIdata')  
rlm_notime<-rlm_notime %>% filter(term=="treatmentMEKi",fdr<fdrt) %>% group_by(feature) %>% summarise(n()) %>%  
  
data('treatment_rlm_wotime_model_analyzed_time96_interactions', package='MODIFIdata')  
rlm_96_only<-rlm96 %>% filter(term=="treatmentMEKi",fdr<fdrt) %>% group_by(feature) %>% summarise(n()) %>%  
  
data('treatment_t_test_analyzed_time96_interactions', package='MODIFIdata')  
ttest_96_only<-ttested_96h_differentialints %>% filter(fdr<fdrt) %>% group_by(feature) %>% summarise(n()) %>%  
  
data('time_treatment_limma_96h_interactions', package='MODIFIdata')  
limma_96_only<-limma96h %>% filter(fdr<fdrt) %>% group_by(feature) %>% summarise(n()) %>% dplyr::rename(limma_96_only=joined)  
  
joined <- rlm %>% full_join(rlm_rand) %>% full_join(rlm_notime) %>% full_join(lm) %>% full_join(rlm_96_only)  
## Joining, by = "feature"  
## Warning: Column `feature` joining factor and character vector, coercing into  
## character vector  
  
joined[is.na(joined)]=0  
  
joined %>%  
  gather("stat",value,-feature) %>%  
  left_join(grouping) %>%  
  group_by(group,stat) %>%  
  summarise(vals=sum(value)) %>%  
  ggplot(aes(x=reorder(stat,-vals,sum),y=vals,fill=group)) +  
    geom_bar(stat = "identity") +  
    scale_fill_brewer(palette = "Dark2") +  
    theme_classic()  
## Joining, by = "feature"
```



26 Figure 3C: Differential interaction scores per pathway

```

data('time_treatment_rlm_model_analyzed_interactions', package='MODIFIdata')

dat <- analyzed %>%
  left_join(target_annotation) %>%
  left_join(query_annotation) %>%
  dplyr::select(target, query, feature, term, fdr, targetpathway, querypathway, estimate)
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector
## Joining, by = "query"
## Warning: Column `query` joining factor and character vector, coercing into
## character vector

dat_to_plot <- dat %>%
  filter(feature=="cell_number", fdr<0.1, term=="treatmentMEKi") %>%
  gather(pw, pathway, -estimate, -fdr, -term, -feature, -target, -query) %>%
  dplyr::select(-pw) %>%
  gather(partner, gene, -feature, -term, -fdr, -estimate, -pathway) %>%
  dplyr::select(-partner) %>%
  mutate(delta=estimate) %>%
  bind_rows(mutate(., pathway="all")) %>%
  filter(pathway %in% c("Ras/Map", "jak/stat", "immunity", "proteasome", "all", "translation")) %>%
  mutate(pathway=factor(pathway, levels = c("all", "Ras/Map", "jak/stat", "immunity", "proteasome", "translation")))

f<-function(x){
  bind_rows(x, dat_to_plot %>% filter(pathway=="all")) %>% mutate(spec=if_else(pathway=="all", "false", "true"))
}

```

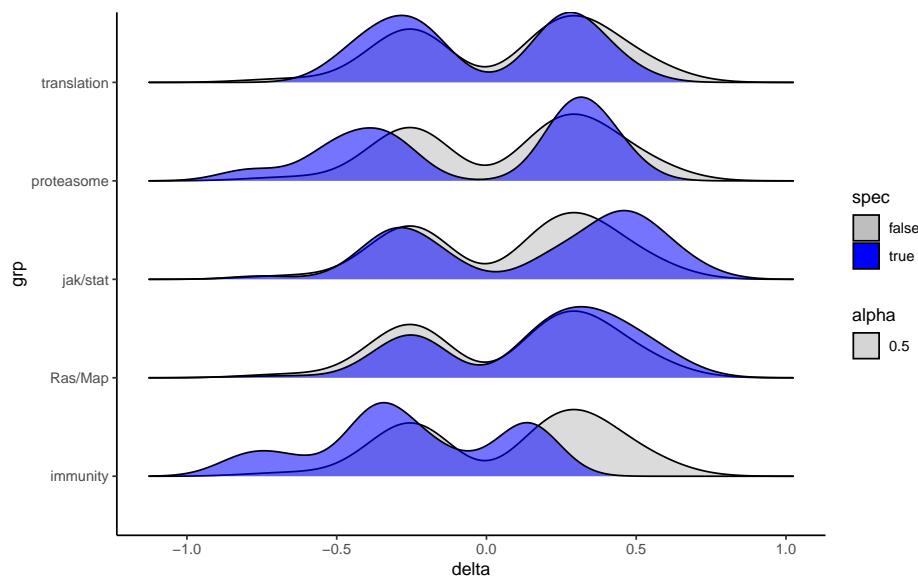
Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

}

dat_to_plot %>%
  group_by(pathway) %>%
  do(
    f(..)
  ) %>%
  ungroup() %>%
  filter(grp!="all") %>%
  mutate(grp=factor(grp,levels = c("immunity","Ras/Map","jak/stat","proteasome","translation"))) %>%
  ggplot(aes(delta, y=grp,fill=spec,height=..density..,alpha=0.5)) +
  geom_joy(scale=0.85) +
  scale_fill_manual(values = c("grey", "blue")) +
  theme_classic()
## Picking joint bandwidth of 0.101

```



```

dat_to_plot %>%
  group_by(pathway) %>%
  do(
    f(..)
  ) %>%
  filter(grp!="all") %>%
  group_by(grp) %>%
  do(
    mdl=dplyr::select(.,val=delta,grouping=spec) %>% ks.test(.val[.$grouping=="true"],.val[.$grouping=="fa")
  ) %>%
  tidy(mdl) %>%
  ungroup() %>%
  mutate(fdr=p.adjust(p.value,method="BH"))
## Warning in ks.test(.val[.$grouping == "true"], .val[.$grouping ==
## "false"], : p-value will be approximate in the presence of ties

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
## Warning in ks.test(.val[.$grouping == "true"], .val[.$grouping ==
## "false"], : p-value will be approximate in the presence of ties

## Warning in ks.test(.val[.$grouping == "true"], .val[.$grouping ==
## "false"], : p-value will be approximate in the presence of ties

## Warning in ks.test(.val[.$grouping == "true"], .val[.$grouping ==
## "false"], : p-value will be approximate in the presence of ties

## Warning in ks.test(.val[.$grouping == "true"], .val[.$grouping ==
## "false"], : p-value will be approximate in the presence of ties
## # A tibble: 5 x 6
##   grp      statistic    p.value method alternative     fdr
##   <fct>        <dbl>     <dbl> <fct>       <fct>     <dbl>
## 1 Ras/Map      0.118 0.000112 Two-sample Kolmog~ two-sided 1.87e-4
## 2 jak/stat     0.282 0.0000000849 Two-sample Kolmog~ two-sided 2.12e-7
## 3 immunity     0.519 0.0000000308 Two-sample Kolmog~ two-sided 1.54e-7
## 4 proteasome    0.280 0.0107   Two-sample Kolmog~ two-sided 1.33e-2
## 5 translation   0.177 0.122    Two-sample Kolmog~ two-sided 1.22e-1

dat_to_plot %>%
  dplyr::select(val=delta,grouping=pathway) %>%
  ks.test(.val[.$grouping=="Ras/Map"],.val[.$grouping=="jak/stat"],alternative = "two.sided",data=.)
## Warning in ks.test(.val[.$grouping == "Ras/Map"], .val[.$grouping == "jak/
## stat"], : p-value will be approximate in the presence of ties
##
## Two-sample Kolmogorov-Smirnov test
##
## data: .val[.$grouping == "Ras/Map"] and .val[.$grouping == "jak/stat"]
## D = 0.19731, p-value = 0.001747
## alternative hypothesis: two-sided

dat_to_plot %>%
  dplyr::select(val=delta,grouping=pathway) %>%
  ks.test(.val[.$grouping=="translation"],.val[.$grouping=="proteasome"],alternative = "two.sided",data=.)
## Warning in ks.test(.val[.$grouping == "translation"], .val[.$grouping == :
## cannot compute exact p-value with ties
##
## Two-sample Kolmogorov-Smirnov test
##
## data: .val[.$grouping == "translation"] and .val[.$grouping == "proteasome"]
## D = 0.22506, p-value = 0.2753
## alternative hypothesis: two-sided
```

27 Figure 4A-D: Example stable and differential positive and negative interactions

```

sample_data=interaction_scores[,,goodfeature,,,] %>% melt() %>%tbl_df()
names(sample_data)=c("target","query","feature","time","treatment","replicate","piscore")

sample_data <- sample_data %>% filter(!grepl("CTRL",query),!grepl("RLUC",target))

# to illustrate different types of behaviours that can be observed in such an analysis we plot representative
#condition stable buffering lethal interaction between Proteasome sub unit 4 (Prosbeta4/Prosbeta4) and mikro
a<-sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  filter(target=="Prosbeta4",query=="mts",feature=="cell_number") %>%
  ggplot(aes(x=time,y=piscore,col=treatment)) +
  geom_smooth(formula = y~x,method = "lm") +
  geom_point() +
  geom_hline(yintercept = 0,lty=2,lwd=1) +
  theme_classic()

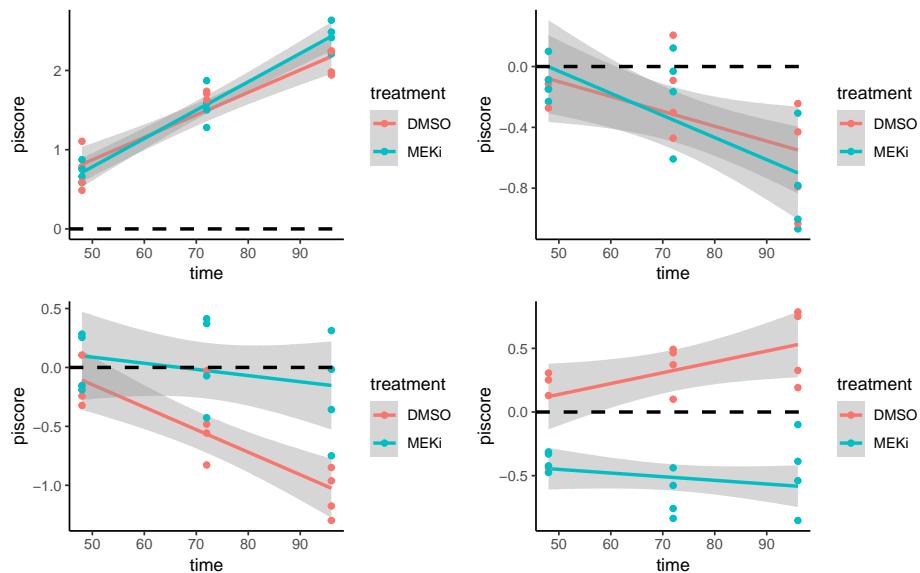
#condition stable synthetic lethal interaction between kinase supressor of ras (ksr) and rolled (rl/ERK)
b<-sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  filter(target=="ksr",query=="rl",feature=="cell_number") %>%
  ggplot(aes(x=time,y=piscore,col=treatment)) +
  geom_smooth(formula = y~x,method = "lm") +
  geom_point() +
  geom_hline(yintercept = 0,lty=2,lwd=1) +
  theme_classic()

#negative dynamic synthetic lethal interaction between skuld (skd/MED13) and Dstat (Stat92E/STAT5B)
c<-sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  filter(target=="skd",query=="Stat92E",feature=="cell_number") %>%
  ggplot(aes(x=time,y=piscore,col=treatment)) +
  geom_smooth(formula = y~x,method = "lm") +
  geom_point() +
  geom_hline(yintercept = 0,lty=2,lwd=1) +
  theme_classic()

#positive dynamic synthetic lethal interaction between Relish (Rel/NFKB) and pointed (pnt/ETS1)
d<-sample_data %>%
  mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
  filter(target=="Rel",query=="pnt",feature=="cell_number") %>%
  ggplot(aes(x=time,y=piscore,col=treatment)) +
  geom_smooth(formula = y~x,method = "rlm") +
  geom_point() +
  geom_hline(yintercept = 0,lty=2,lwd=1) +
  theme_classic()

```

a+b+c+d



28 Figure 4E: Proof that the linear model does predict equal amounts of interactions for every feature independently

```
p1 <- analyzed %>%
  filter(fdr<0.1, term=="time", estimate>0) %>%
  left_join(grouping) %>%
  group_by(group) %>%
  summarise(count=n(), norm=n[1], val=count/norm) %>%
  ggplot(aes(x="", y=val, fill=group))+
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y") +
  scale_fill_brewer(palette = "Dark2") +
  ggttitle("alleviating stable interactions")
## Joining, by = "feature"
## Warning: Column `feature` joining factor and character vector, coercing into
## character vector

analyzed %>%
  filter(fdr<0.1, term=="time", estimate>0) %>%
  n_distinct()
## [1] 8522

p2 <- analyzed %>%
  filter(fdr<0.1, term=="time", estimate<0) %>%
  left_join(grouping) %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
group_by(group) %>%
  summarise(count=n(),norm=n[1],val=count/norm) %>%
  ggplot(aes(x="", y=val, fill=group))+ 
    geom_bar(width = 1, stat = "identity") +
    coord_polar("y") +
    scale_fill_brewer(palette = "Dark2") +
    ggtitle("aggravating stable interactions")
## Joining, by = "feature"
## Warning: Column `feature` joining factor and character vector, coercing into
## character vector

analyzed %>%
  filter(fdr<0.1,term=="time",estimate<0) %>%
  n_distinct()
## [1] 9946

p3 <- analyzed %>%
  filter(fdr<0.1,term=="treatmentMEKi",estimate>0) %>%
  left_join(grouping) %>%
  group_by(group) %>%
  summarise(count=n(),norm=n[1],val=count/norm) %>%
  ggplot(aes(x="", y=val, fill=group))+ 
    geom_bar(width = 1, stat = "identity") +
    coord_polar("y") +
    scale_fill_brewer(palette = "Dark2") +
    ggtitle("positive differential interactions")
## Joining, by = "feature"
## Warning: Column `feature` joining factor and character vector, coercing into
## character vector

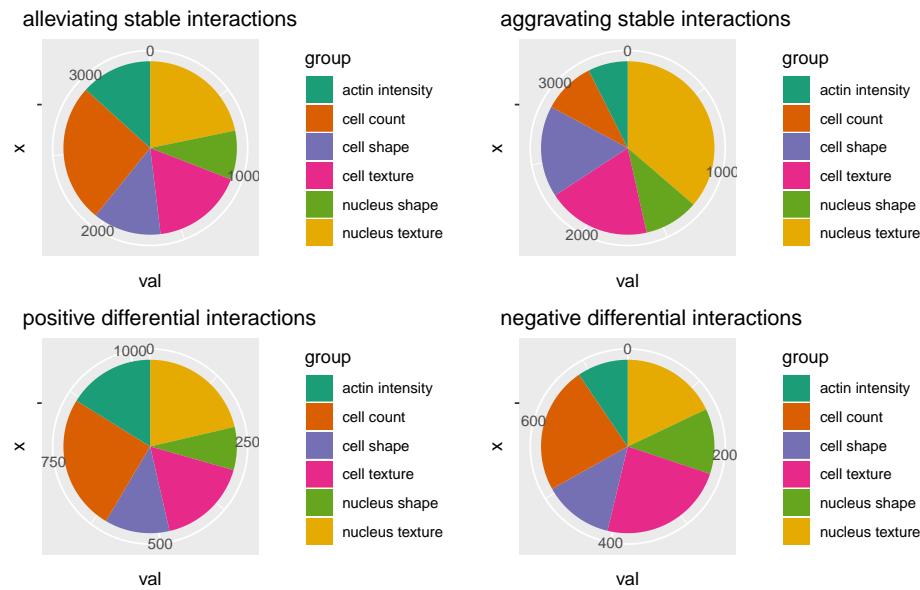
analyzed %>%
  filter(fdr<0.1,term=="treatmentMEKi",estimate>0) %>%
  n_distinct()
## [1] 2551

p4 <- analyzed %>%
  filter(fdr<0.1,term=="treatmentMEKi",estimate<0) %>%
  left_join(grouping) %>%
  group_by(group) %>%
  summarise(count=n(),norm=n[1],val=count/norm) %>%
  ggplot(aes(x="", y=val, fill=group))+ 
    geom_bar(width = 1, stat = "identity") +
    coord_polar("y") +
    scale_fill_brewer(palette = "Dark2") +
    ggtitle("negative differential interactions")
## Joining, by = "feature"
## Warning: Column `feature` joining factor and character vector, coercing into
## character vector

analyzed %>%
  filter(fdr<0.1,term=="treatmentMEKi",estimate<0) %>%
```

```
n_distinct()
## [1] 2172

p1 + p2 + p3 + p4 + plot_layout(ncol = 2,nrow = 2)
```



29 Figure 4F-F': Test if all model residuals can be explained by measurement variance

This analysis tests if the residuals of the linear model fit can be explained by the variance in the data. If the variance scales smaller than the residuals it is indicative that the model does not explain the data well. If variance and residuals scale equally we can conclude that the model fits the data well. And if the model fails it is because the data has too high variance rather than the wrong model was chosen.

```
colfunc <- colorRampPalette(c("#555555", "black"))

a<-analyzed %>%
  filter(term=="time",feature=="cell_number") %>%
  ggplot(aes(x=residuals,y=sumofvariance)) +
  geom_hex(aes(residuals, sumofvariance), bins = 250) +
  scale_x_log10() +
  scale_y_log10() +
  geom_smooth(method = "rlm",col="black") +
  scale_fill_gradient("", colours = colfunc(150)) +
  theme_classic()

analyzed %>%
  filter(term=="time",feature=="cell_number") %>% dplyr::select(residuals,sumofvariance) %>% cor(use="p",meth
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

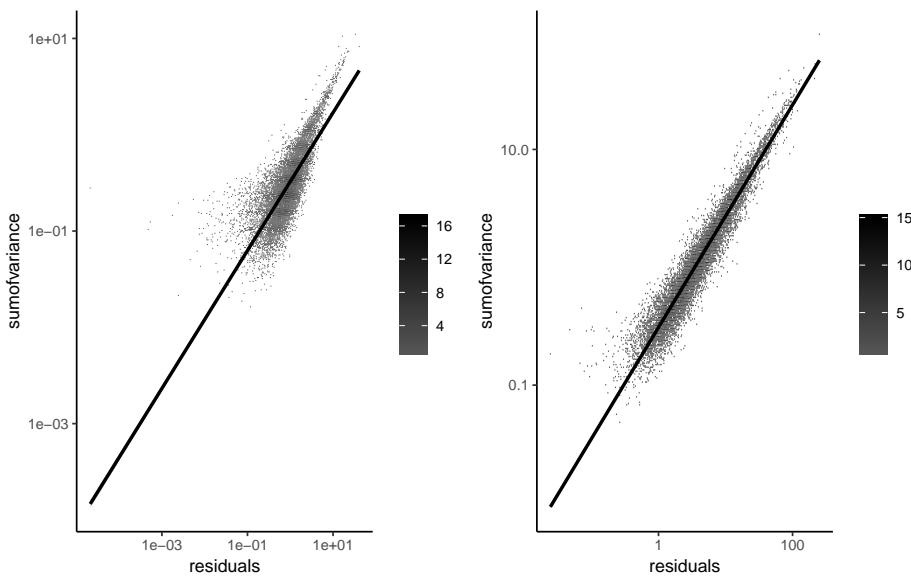
```
##             residuals sumofvariance
## residuals      1.0000000   0.9280765
## sumofvariance  0.9280765   1.0000000

b<-analyzed %>%
  filter(term=="time",feature=="actin.m.eccentricity.tmean") %>%
  ggplot(aes(x=residuals,y=sumofvariance)) +
  geom_hex(aes(residuals, sumofvariance), bins = 250) +
  scale_x_log10() +
  scale_y_log10() +
  geom_smooth(method = "rlm",col="black") +
  scale_fill_gradientn("", colours = colfunc(150)) +
  theme_classic()

analyzed %>%
  filter(term=="time",feature=="actin.m.eccentricity.tmean") %>% dplyr::select(residuals,sumofvariance) %>% c

##             residuals sumofvariance
## residuals      1.0000000   0.9633149
## sumofvariance  0.9633149   1.0000000

a+b
## Warning in self$trans$transform(x): NaNs produced
## Warning: Transformation introduced infinite values in continuous x-axis
## Warning in self$trans$transform(x): NaNs produced
## Warning: Transformation introduced infinite values in continuous x-axis
## Warning: Removed 143 rows containing non-finite values (stat_binhex).
## Warning: Removed 143 rows containing non-finite values (stat_smooth).
## Warning in self$trans$transform(x): NaNs produced
## Warning: Transformation introduced infinite values in continuous x-axis
## Warning in self$trans$transform(x): NaNs produced
## Warning: Transformation introduced infinite values in continuous x-axis
## Warning: Removed 3 rows containing non-finite values (stat_binhex).
## Warning: Removed 3 rows containing non-finite values (stat_smooth).
```



30 Figure 4G-H: pathway specific stable and differential genetic interactions

```
#merge annotations and data
pathwaycoverage <- target_annotation %>% group_by(targetpathway) %>% summarise(n=n())

toplot <- analyzed %>%
  filter(term=="time", fdr<=0.3, feature=="cell_number") %>% #
  left_join(target_annotation) %>%
  left_join(query_annotation) %>%
  mutate(direction=ifelse(estimate>0,"alleviating","aggravating")) %>%
  dplyr::select(target,query,targetpathway,querypathway,direction)
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector
## Joining, by = "query"
## Warning: Column `query` joining factor and character vector, coercing into
## character vector

names=toplot %>%
  rowwise() %>%
  do(con=paste(sort(c(as.character(.$target),as.character(.$query))),collapse="_")) %>%
  unnest()

toplot<-
  cbind.data.frame(toplot,names) %>%
 tbl_df() %>%
  dplyr::select(-target,-query) %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

dplyr::distinct() %>%
dplyr::select(targetpathway,querypathway,direction) %>%
gather(obsolete,pathway,-direction) %>%
dplyr::select(-obsolete,targetpathway=pathway) %>%
group_by(targetpathway,direction) %>%
summarise(count=n()) %>%
complete(targetpathway,direction,fill = list(0)) %>%
left_join(pathwaycoverage) %>%
mutate(n_count=count/n)
## Joining, by = "targetpathway"

sums <- toplot %>% group_by(direction) %>% summarise_if(is.numeric,sum)
tempo <-left_join(toplot,sums %>% dplyr::select(countsum=count,nsum=n,-n_count,direction))
## Joining, by = "direction"
tempo %>>%
  group_by(targetpathway,direction) %>%
  mutate(p=binom.test(count,countsum,p = n/nsum,alternative = "g")$p.value) %>%
  ungroup() %>%
  mutate(fdr=p.adjust(p,method = "BH"),sig=ifelse(fdr<=0.1,'good','bad'))

tempo_stable <- tempo %>% dplyr::select(pathway=targetpathway,n_count_stable=n_count,sig_stable=sig,direction)

toplot <- analyzed %>%
  filter(term=="treatmentMEKi",fdr<=0.3,feature=="cell_number") %>% #, feature=="cell_number"
  left_join(target_annotation) %>%
  left_join(query_annotation) %>%
  mutate(direction=ifelse(estimate>0,"positive","negative")) %>%
  dplyr::select(target,query,targetpathway,querypathway,direction)
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector
## Joining, by = "query"
## Warning: Column `query` joining factor and character vector, coercing into
## character vector

names=toplot %>%
  rowwise() %>%
  do(con=paste(sort(c(as.character(.$target),as.character(.$query))),collapse="_")) %>%
  unnest()

toplot<-
  cbind.data.frame(toplot,names) %>%
 tbl_df() %>%
  dplyr::select(-target,-query) %>%
  dplyr::distinct() %>%
  dplyr::select(targetpathway,querypathway,direction) %>%
gather(obsolete,pathway,-direction) %>%
dplyr::select(-obsolete,targetpathway=pathway) %>%
group_by(targetpathway,direction) %>%
summarise(count=n())

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
complete(targetpathway,direction,fill = list(0)) %>%
  left_join(pathwaycoverage) %>%
  mutate(n_count=count/n)
## Joining, by = "targetpathway"

sums <- toplot %>% group_by(direction) %>% summarise_if(is.numeric,sum)
tempo <-left_join(toplot,sums %>% dplyr::select(countsum=count,nsum=n,-n_count,direction))
## Joining, by = "direction"
tempo %>>%
  group_by(targetpathway,direction) %>%
  mutate(p=binom.test(count,countsum,p = n/nsum,alternative = "g")$p.value) %>%
  ungroup() %>%
  mutate(fdr=p.adjust(p,method = "BH"),sig=ifelse(fdr<=0.1,'good','bad'))

tempo_diff <- tempo %>% dplyr::select(pathway=targetpathway,n_count_diff=n_count,sig_diff=sig,direction_diff=direction)

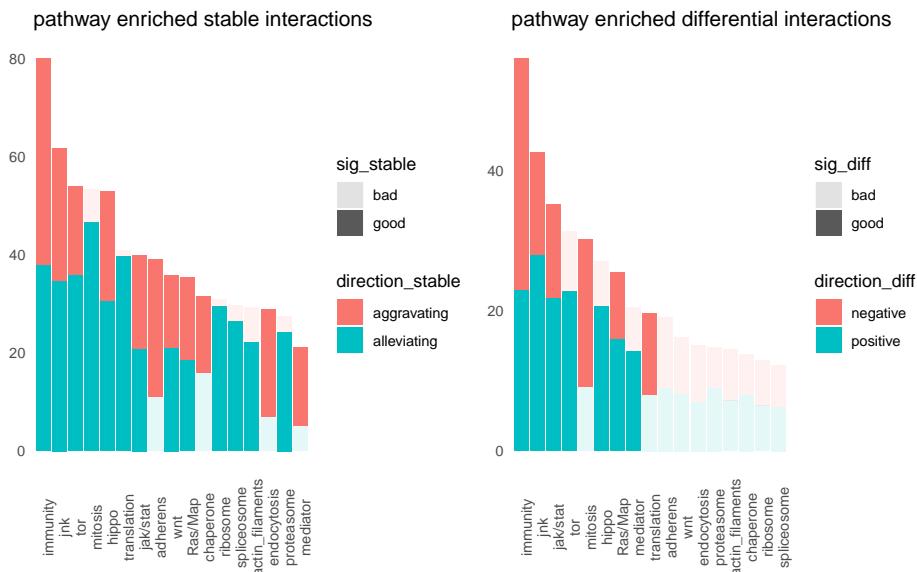
tempo_complete <- full_join(tempo_stable,tempo_diff) %>% complete(pathway,direction_stable,direction_diff) %>%
## Joining, by = "pathway"

a<-tempo_complete %>%
  dplyr::select(pathway,n_count_stable,direction_stable,sig_stable) %>%
  dplyr::distinct() %>%
  ggplot(aes(x=reorder(pathway,-n_count_stable,sum),y=n_count_stable,fill=direction_stable,alpha=sig_stable)) +
  geom_bar(stat="identity") +
  theme(axis.title = element_blank(),panel.background = element_blank(),axis.ticks.x = element_blank(),axis.title.y=element_blank())+
  ggtitle("pathway enriched stable interactions")

b<-tempo_complete %>%
  dplyr::select(pathway,n_count_diff,direction_diff,sig_diff) %>%
  dplyr::distinct() %>%
  ggplot(aes(x=reorder(pathway,-n_count_diff,sum,ma.rn=T),y=n_count_diff,fill=direction_diff,alpha=sig_diff)) +
  geom_bar(stat="identity") +
  theme(axis.title = element_blank(),panel.background = element_blank(),axis.ticks.x = element_blank(),axis.title.y=element_blank())+
  ggtitle("pathway enriched differential interactions")

a+b
## Warning: Using alpha for a discrete variable is not advised.
## Warning: Using alpha for a discrete variable is not advised.
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



31 Figure 4I-J: gene specific stable and differential genetic interactions

```
#Plot a diagram, that shows which genes have few or a lot of differential genetic itneractions
x <- analyzed %>%
  filter(term=="treatmentMEKi", fdr<0.3, feature=="cell_number") %>%
  mutate(sign=ifelse(estimate>0,"positive","negative")) %>%
  dplyr::select(target,query,sign) %>%
  mutate(target=as.character(target),query=as.character(query))

y <- x %>% rowwise() %>%
  do(con=paste(sort(c(.target,.query)),collapse="_")) %>%
  unnest() %>%
  separate(con,into=c('target','query'),sep = "_")

x$target=y$target
x$query=y$query

x1 <- x %>% dplyr::distinct() %>% group_by(target,sign) %>% summarise(n=n()) %>% ungroup()
x2 <- x %>% dplyr::distinct() %>% group_by(query,sign) %>% summarise(n=n()) %>% ungroup() %>% dplyr::select(-n)

a<- rbind(x1,x2) %>%
  group_by(target,sign) %>%
  summarise(n=sum(n)) %>%
  spread(sign,n,fill = 0) %>%
  left_join(target_annotation) %>%
  mutate(targetpathway=ifelse(targetpathway %in% c("Ras/Map","jak/stat","immunity","tor"),targetpathway,NA))
ggplot(aes(x=positive,y=negative,col=targetpathway,label=target)) + geom_point() +
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
geom_text(aes(label=ifelse(positive>15|negative>15,as.character(target),'')),hjust=1,vjust=1) +
  theme_classic()
## Joining, by = "target"

differential <- analyzed %>%
  filter(term=="treatmentMEKi",fdr<0.3,feature=="cell_number") %>%
  mutate(sign=ifelse(estimate>0,"positive","negative")) %>%
  dplyr::select(target,query,sign) %>%
  mutate(target=as.character(target),query=as.character(query))

y <- differential %>%  rowwise() %>%
  do(con=paste(sort(c(.target,.query)),collapse="_")) %>%
  unnest() %>%
  separate(con,into=c('target','query'),sep = "_")

differential$target=y$target
differential$query=y$query
differential=differential %>%
  dplyr::distinct() %>%
  dplyr::select(-sign) %>%
  gather(type,gene) %>%
  dplyr::select(-type,target=gene) %>%
  group_by(target) %>%
  summarise(diff=n())

dynamic <- analyzed %>%
  filter(term=="time",fdr<0.3,feature=="cell_number") %>%
  mutate(sign=ifelse(estimate>0,"positive","negative")) %>%
  dplyr::select(target,query,sign) %>%
  mutate(target=as.character(target),query=as.character(query))

y <- dynamic %>%  rowwise() %>%
  do(con=paste(sort(c(.target,.query)),collapse="_")) %>%
  unnest() %>%
  separate(con,into=c('target','query'),sep = "_")

dynamic$target=y$target
dynamic$query=y$query
dynamic=dynamic %>%
  dplyr::distinct() %>%
  dplyr::select(-sign) %>%
  gather(type,gene) %>%
  dplyr::select(-type,target=gene) %>%
  group_by(target) %>%
  summarise(dyn=n())

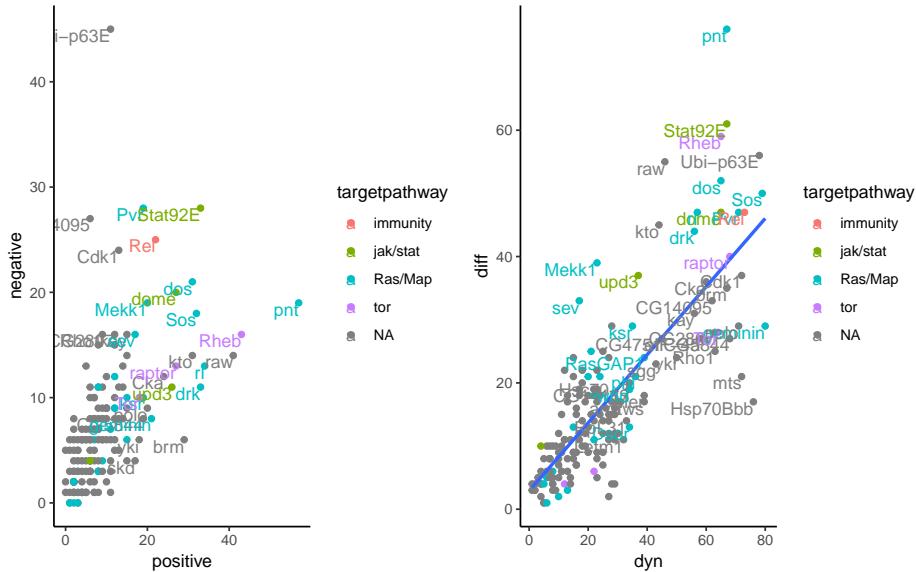
b<-dynamic %>%
  left_join(differential) %>%
  left_join(target_annotation) %>%
```

```

  mutate(diff=ifelse(is.na(diff),0,diff)) %>%
  mutate(targetpathway=ifelse(targetpathway %in% c("Ras/Map","jak/stat","immunity","tor"),targetpathway,NA))
  ggplot(aes(x=dyn,y=diff,col=targetpathway,label=target)) +
  geom_point() +
  geom_smooth(aes(x=dyn,y=diff),method="lm",se=F,inherit.aes = F) +
#  geom_abline(slope = 1,intercept = 0) +
  geom_text(aes(label=ifelse(dyn>30|diff>30,as.character(target),'')),hjust=1,vjust=1) +
  theme_classic()
## Joining, by = "target"
## Joining, by = "target"

```

a+b



32 Figure 5A-D: gene specific stable and differential genetic interactions

```

#Figure 5A: plot the time dependence of interactions among different core pathways
dat <- analyzed %>%
  left_join(target_annotation) %>%
  left_join(query_annotation) %>%
  dplyr::select(target,query,feature,term,fdr,targetpathway,querypathway,estimate)
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector
## Joining, by = "query"
## Warning: Column `query` joining factor and character vector, coercing into
## character vector

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

a<-dat %>%
  filter(feature=="cell_number", fdr<0.1, term=="time") %>% #
  gather(pw, pathway, -estimate, -fdr, -term, -feature, -target, -query) %>%
  dplyr::select(-pw) %>%
  mutate(speed=(abs(estimate)-min(abs(estimate)))/(max(abs(estimate))-min(abs(estimate)))) %>%
  bind_rows(mutate(., pathway="all")) %>%
  filter(pathway %in% c("all", "Ras/Map", "jak/stat", "immunity", "proteasome", "translation")) %>%
  mutate(pathway=factor(pathway, levels=c("all", "immunity", "jak/stat", "Ras/Map", "proteasome", "translation")))
  ggplot(aes(x=pathway, y=speed)) +
  geom_boxplot() +
  theme_classic() +
  geom_signif(comparisons = list(c("immunity", "jak/stat"),
                                 c("jak/stat", "Ras/Map"),
                                 c("proteasome", "Ras/Map"),
                                 c("proteasome", "translation"))
  ),
  step_increase=0.1,
  map_signif_level=c("***=0.001, **=0.01, *=0.05),
  test = 'wilcox.test',
  test.args = list(alternative="two.sided")
)

#Figure 5B: plot the time dependence of interaction differences among different core pathways

#first filter significant differential interactions:
set<-
  analyzed %>%
  filter(feature %in% goodfeature, term=="treatmentMEKi", fdr<0.1) %>%
  unite(chosen, target, query, feature, sep = "&") %>%
  pull(chosen) %>%
  unique()

interaction_scores_df<-
  melt(interaction_scores) %>%
 tbl_df() %>%
  dplyr::select(target=Var1, query=Var2, feature=Var3, time=Var4, treatment=Var5, replicate=Var6, pi=value) %>%
  unite(chosen, target, query, feature, sep = "&", remove = F)

candidates_an <- interaction_scores_df %>%
  filter(chosen %in% set) %>%
  left_join(target_annotation) %>%
  left_join(query_annotation)
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector
## Joining, by = "query"
## Warning: Column `query` joining factor and character vector, coercing into
## character vector

#plot parallel coordinates of the mean differences over time
b<-candidates_an %>%

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

filter(chosen %in% set
       ,time!="72h"
       ) %>%
dplyr::select(target,query,targetpathway,querypathway,pi,treatment,time) %>%
gather(pw,pathway,-target,-query,-pi,-treatment,-time) %>%
dplyr::select(-pw) %>%
filter( pathway %in% c("Ras/Map","jak/stat","immunity","proteasome","translation")) %>%
group_by(target,query,pathway,time) %>%
summarise(diff=abs(mean(pi[treatment=="DMSO"],na.rm=T)-mean(pi[treatment=="MEKi"],na.rm=T))) %>%
group_by(pathway,time) %>%
summarise(diff=median(diff,na.rm=T)) %>%
ggplot(aes(y=diff, x=time,group = pathway,label = pathway)) +
  geom_path(aes(color=pathway),alpha = 1,lineend = 'round', linejoin = 'round',lwd=2.3) +
  geom_point(aes(color=pathway),alpha = 1,cex=2.3) +
  geom_vline(xintercept =c(1,2),lty=2) +
  #geom_label_repel() +
  theme_classic()

#Figure 5B': plot the time dependence of interaction differences among different core pathways, when normaliz

c<-candidates_an %>%
filter(chosen %in% set
       ,time!="72h"
       ) %>%
dplyr::select(target,query,targetpathway,querypathway,pi,treatment,time) %>%
gather(pw,pathway,-target,-query,-pi,-treatment,-time) %>%
dplyr::select(-pw) %>%
filter( pathway %in% c("Ras/Map","jak/stat","immunity","proteasome","translation")) %>%
group_by(target,query,pathway,time) %>%
summarise(diff=abs(mean(pi[treatment=="DMSO"],na.rm=T)-mean(pi[treatment=="MEKi"],na.rm=T))) %>%
group_by(pathway,time) %>%
summarise(diff=median(diff,na.rm=T)) %>%
group_by(pathway) %>%
mutate(diff=diff-diff[time=="48h"]) %>%
ggplot(aes(y=diff, x=time,group = pathway,label = pathway)) +
  geom_path(aes(color=pathway),alpha = 1,lineend = 'round', linejoin = 'round',lwd=2.3) +
  geom_vline(xintercept =c(1,2),lty=2) +
  geom_point(aes(color=pathway),alpha = 1,cex=2.3) +
  #geom_label_repel() +
  theme_classic()

#Figure 5C: initial interaction differences among different example features

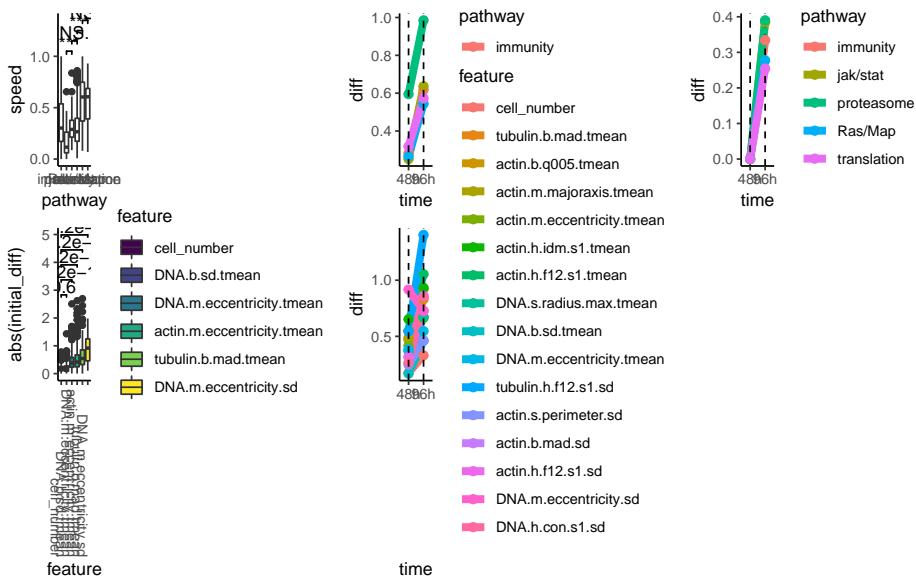
#plot examples as ordered boxes
d<- interaction_scores_df %>%
filter(chosen %in% set,time=="48h") %>%
group_by(target,query,feature) %>%
summarise(initial_diff=mean(pi[treatment=="DMSO"],na.rm=T)-mean(pi[treatment=="MEKi"],na.rm=T)) %>%
ungroup() %>%
filter(feature %in% c("cell_number","actin.m.eccentricity.tmean","DNA.m.eccentricity.tmean","DNA.m.eccentr
mutate(feature=factor(feature,levels = c("cell_number","DNA.b.sd.tmean","DNA.m.eccentricity.tmean","actin.m.eccen

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
ggplot(aes(y=abs(initial_diff), x=feature, fill=feature)) +  
  geom_boxplot() +  
  theme_classic() +  
  geom_signif(  
    comparisons = list(  
      c("cell_number","DNA.b.sd.tmean"),  
      c("cell_number","DNA.m.eccentricity.tmean"),  
      c("cell_number","actin.m.eccentricity.tmean"),  
      c("cell_number","tubulin.b.mad.tmean"),  
      c("cell_number","DNA.m.eccentricity.sd")  
    ),  
    test = "t.test",  
    test.args = list(alternative="two.sided"),  
    step_increase = 0.2  
  ) +  
  theme(axis.text.x=element_text(angle=-90,hjust=1))  
  
#Figure 5D: plot parallel coordinates of the mean differences over time  
#clearly shows that texture feature mark differential interaction features the most time dependent  
#again highlights the different initial interaction differences that can be observed depending on the pathway  
#shows that DNA exxentricity variation across cells does vary the most initially by become suppressed in the  
#most feature show a similar rate of differential interaction development  
  
e<-interaction_scores_df %>%  
  filter(chosen %in% set  
    ,time!="72h"  
    ) %>%  
  group_by(target,query,feature,time) %>%  
  summarise(diff=abs(mean(pi[treatment=="DMSO"],na.rm=T)-mean(pi[treatment=="MEKi"],na.rm=T))) %>%  
  group_by(feature,time) %>%  
  summarise(diff=median(diff,na.rm=T)) %>%  
  ggplot(aes(y=diff, x=time,group = feature,label = feature)) +  
    geom_path(aes(color=feature),alpha = 1,lineend = 'round', linejoin = 'round',lwd=2.3) +  
    geom_point(aes(color=feature),alpha = 1,cex=2.3) +  
    geom_vline(xintercept =c(1,2),lty=2) +  
    theme_classic()  
  
a+b+c+d+e
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



33 Figure 6A: Small multiple networks

```

nodes_of_interest <- c('hoip','CG2807','eIF-2beta','CG3605','CG11985','skd','kto','RasGAP1','dome','Rheb','S')

#compo= read_csv("./default_node_table.csv") %>%
#  #      dplyr::select(target=name,targetpathway)

data("core_genes_and_pathways", package="MODIFIdata")

data_sub <-
  data_df %>%
  filter(feature %in% goodfeature, target %in% nodes_of_interest, query %in% nodes_of_interest)

data_sub %>%
  filter(fdr<0.1, feature=="actin.m.eccentricity.tmean", time=="48h", drug=="DMSO") %>%
  mutate(sign=sign(piscore)) %>%
  dplyr::select(target,query,piscore,sign) %>%
  left_join(compo) %>%
  write_delim(path = "DMSO_48.tab", delim = "\t")
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector

data_sub %>%
  filter(fdr<0.1, feature=="actin.m.eccentricity.tmean", time=="72h", drug=="DMSO") %>%
  mutate(sign=sign(piscore)) %>%
  dplyr::select(target,query,piscore,sign) %>%
  left_join(compo) %>%
  write_delim(path = "DMSO_72.tab", delim = "\t")

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector

data_sub %>%
  filter(fdr<0.1,feature=="actin.m.eccentricity.tmean",time=="96h",drug=="DMSO") %>%
  mutate(sign=sign(piscore)) %>%
  dplyr::select(target,query,piscore,sign) %>%
  left_join(compo) %>%
  write_delim(path = "DMSO_96.tab",delim = "\t")
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector

data_sub %>%
  filter(fdr<0.1,feature=="actin.m.eccentricity.tmean",time=="48h",drug=="MEKi") %>%
  mutate(sign=sign(piscore)) %>%
  dplyr::select(target,query,piscore,sign) %>%
  left_join(compo) %>%
  write_delim(path = "MEKi_48.tab",delim = "\t")
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector

data_sub %>%
  filter(fdr<0.1,feature=="actin.m.eccentricity.tmean",time=="72h",drug=="MEKi") %>%
  mutate(sign=sign(piscore)) %>%
  dplyr::select(target,query,piscore,sign) %>%
  left_join(compo) %>%
  write_delim(path = "MEKi_72.tab",delim = "\t")
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector

data_sub %>%
  filter(fdr<0.1,feature=="actin.m.eccentricity.tmean",time=="96h",drug=="MEKi") %>%
  mutate(sign=sign(piscore)) %>%
  dplyr::select(target,query,piscore,sign) %>%
  left_join(compo) %>%
  write_delim(path = "MEKi_96.tab",delim = "\t")
## Joining, by = "target"
## Warning: Column `target` joining factor and character vector, coercing into
## character vector

#the node layout was then set in cytoscape as of the correlation network in 6C
#go on to cytoscape using the textfiles that were created
```

34 Figure 6B: Example correlations of differential interactions based on cell eccentricity

```

a<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("dome","Stat92E"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  mutate(bla=factor(bla,levels = bla[order(dome)])) %>%
  gather(target,estimate,-bla) %>%
  ggplot(aes(x=bla,y=target,fill=estimate)) +
  geom_raster() +
  scale_fill_gradient(low="#5087C8" ,high ="#F2EE35" ) +
  theme_classic()

tocor<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("dome","Stat92E"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  dplyr::select(-bla)# %>%

rcorr(x=tocor[[1]],y=tocor[[2]],type = "pearson")
##      x     y
## x  1.00 0.73
## y  0.73 1.00
##
## n= 76
##
##
## P
##   x   y
## x   0
## y   0

b<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("Pvr","Sos"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  mutate(bla=factor(bla,levels = bla[order(Sos)])) %>%
  gather(target,estimate,-bla) %>%
  ggplot(aes(x=bla,y=target,fill=estimate)) +
  geom_raster() +
  scale_fill_gradient(low="#5087C8" ,high ="#F2EE35" ) +
  theme_classic()

tocor<-analyzed %>%

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
filter(term=="treatmentMEKi",target %in% c("Pvr","Sos"),feature=="actin.m.eccentricity.tmean") %>%
dplyr::select(target,query,feature,estimate) %>%
spread(target,estimate) %>%
unite(bla,query,feature) %>%
dplyr::select(-bla)# %>%

rcorr(x=tocor[[1]],y=tocor[[2]],type = "pearson")
##      x      y
## x  1.00  0.51
## y  0.51  1.00
##
## n= 76
##
##
## P
##      x      y
## x  0
## y  0

c<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("pnt","Rel"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  mutate(bla=factor(bla,levels = bla[order(pnt)])) %>%
  gather(target,estimate,-bla) %>%
  ggplot(aes(x=bla,y=target,fill=estimate)) +
  geom_raster() +
  scale_fill_gradient(low="#5087C8" ,high ="#F2EE35" ) +
  theme_classic()

tocor<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("pnt","Rel"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  dplyr::select(-bla)# %>%

rcorr(x=tocor[[1]],y=tocor[[2]],type = "pearson")
##      x      y
## x  1.00 -0.37
## y -0.37  1.00
##
## n= 76
##
##
## P
##      x      y
## x  0.001
## y  0.001
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
d<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("RasGAP1","Rel"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  mutate(bla=factor(bla,levels = bla[order(RasGAP1)])) %>%
  gather(target,estimate,-bla) %>%
  ggplot(aes(x=bla,y=target,fill=estimate)) +
  geom_raster() +
  scale_fill_gradient(low="#5087C8" ,high ="#F2EE35" ) +
  theme_classic()

tocor<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("RasGAP1","Rel"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  dplyr::select(-bla)# %>%

rcorr(x=tocor[[1]],y=tocor[[2]],type = "pearson")
##      x      y
## x  1.00  0.38
## y  0.38  1.00
##
## n= 76
##
##
## P
##      x      y
## x          8e-04
## y  8e-04

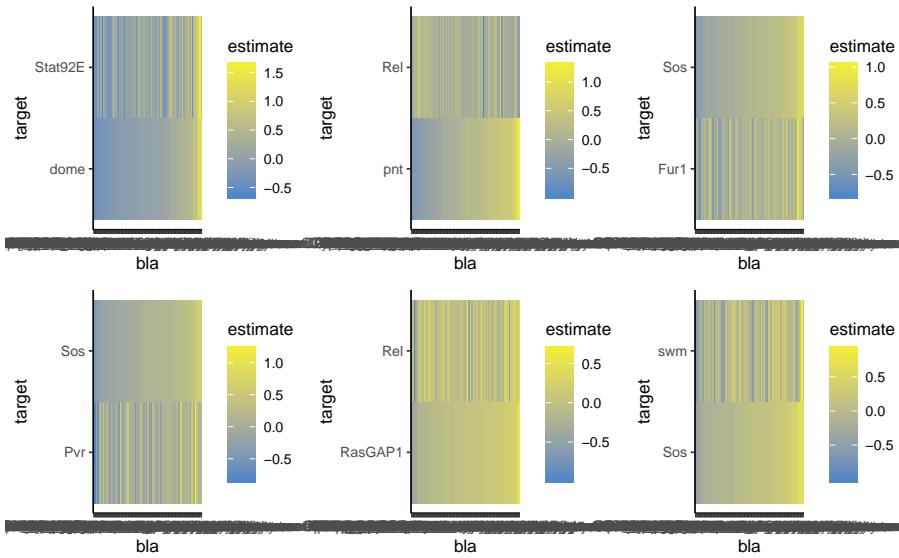
e<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("Sos","swm"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
  mutate(bla=factor(bla,levels = bla[order(Sos)])) %>%
  gather(target,estimate,-bla) %>%
  ggplot(aes(x=bla,y=target,fill=estimate)) +
  geom_raster() +
  scale_fill_gradient(low="#5087C8" ,high ="#F2EE35" ) +
  theme_classic()

tocor<-analyzed %>%
  filter(term=="treatmentMEKi",target %in% c("Sos","swm"),feature=="actin.m.eccentricity.tmean") %>%
  dplyr::select(target,query,feature,estimate) %>%
  spread(target,estimate) %>%
  unite(bla,query,feature) %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
dplyr::select(-bla) # %>%  
  
rcorr(x=tocor[[1]],y=tocor[[2]],type = "pearson")  
##      x     y  
## x  1.0  0.5  
## y  0.5  1.0  
##  
## n= 76  
##  
##  
## P  
##   x   y  
## x    0  
## y   0  
  
f<-analyzed %>%  
  filter(term=="treatmentMEKi",target %in% c("Sos","Fur1"),feature=="actin.m.eccentricity.tmean") %>%  
  dplyr::select(target,query,feature,estimate) %>%  
  spread(target,estimate) %>%  
  unite(bla,query,feature) %>%  
  mutate(bla=factor(bla,levels = bla[order(Sos)])) %>%  
  gather(target,estimate,-bla) %>%  
  ggplot(aes(x=bla,y=target,fill=estimate)) +  
  geom_raster() +  
  scale_fill_gradient(low="#5087C8" ,high ="#F2EE35" ) +  
  theme_classic()  
  
tocor<-analyzed %>%  
  filter(term=="treatmentMEKi",target %in% c("Sos","Fur1"),feature=="actin.m.eccentricity.tmean") %>%  
  dplyr::select(target,query,feature,estimate) %>%  
  spread(target,estimate) %>%  
  unite(bla,query,feature) %>%  
  dplyr::select(-bla) # %>%  
  
rcorr(x=tocor[[1]],y=tocor[[2]],type = "pearson")  
##      x     y  
## x  1.00 0.57  
## y  0.57 1.00  
##  
## n= 76  
##  
##  
## P  
##   x   y  
## x    0  
## y   0  
  
a+c+f+b+d+e
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



35 Figure 6C: A correlation network of differential genetic interactions

```
#all differential interaction estimates are being used in this analysis
#all pairwise complete pearson correlation coefficients are calculated
#every positive correlation higher than 0.5 draws an edge

data("target_annotation", package = "MODIFIdata")
data("query_annotation", package = "MODIFIdata")
data("time_treatment_rlm_model_analyzed_interactions", package = "MODIFIdata")

x=analyzed %>%
  filter(term=="treatmentMEKi") %>%
  dplyr::select(target, query, feature, estimate) %>%
  unite(feat, feature, query) %>%
  spread(target, estimate, fill = 0) %>%
  dplyr::select(-feat) %>%
  cor(use = "p", method = "p") %>%
  melt() %>%
  filter(value!=1) %>%
  dplyr::select(Var1, Var2, Diffcor=value)

names=x %>%
  filter(Diffcor>0.5) %>%
  rowwise() %>%
  do(con=paste(sort(c(as.character(.\$Var1), as.character(.\$Var2))), collapse="_")) %>%
  unnest() %>%
  separate(con, into=c('target', 'query'), sep = "_")
```

```

df=x %>%
  filter(Diffcor>0.5)

df=cbind.data.frame(names,df) %>%
  dplyr::select(-Var1,-Var2) %>%
  dplyr::distinct() %>%
  left_join(target_annotation) %>%
  left_join(query_annotation) %>%
  dplyr::select(target,query,Diffcor,targetpathway,querypathway)
## Joining, by = "target"
## Joining, by = "query"

write_delim(df,"correlations_of_differential_interactions_v5.tab",delim = "\t")

#go on in cytoscape

```

36 Supplementary Figure S7: A differential correlation analysis according to Billmann et al.

```

x=data_df %>%
  filter(fdr<0.1,!grepl("*RLUC*",target), !grepl("*CTRL*",query)) %>%
  unite(feat,query,feature,time) %>%
  unite(targ,target,drug) %>%
  dplyr::select(-fdr) %>%
  spread(targ,piscore,fill=0) %>%
  dplyr::select(-feat)

corrs=cor(x,use = "p",method = "p")

corrs[upper.tri(corrs,diag = T)]=NA

corr_df=melt(corrs) %>% separate(Var1,c("gene1","cond1"),sep = "_") %>% separate(Var2,c("gene2","cond2"),sep = "_")

final_intra_inner_corr <- corr_df %>%
  mutate(comp=if_else(cond1==cond2,"intra","inter")) %>%
  dplyr::select(-cond2) %>%
  spread(comp,value) %>%
  drop_na() %>%
  dplyr::select(-cond1) %>%
  group_by(gene1,gene2) %>%
  summarise(inter=mean(inter,na.rm=T),intra=mean(intra,na.rm=T))

a<-final_intra_inner_corr %>%
  dplyr::select(target=gene1,query=gene2,inter,intra) %>%
  left_join(target_annotation %>%
  dplyr::select(target,targetpathway)) %>%
  mutate(targetpathway=ifelse(targetpathway %in% c("Ras/Map","jak/stat","tor"),targetpathway,"other")) %>%

```

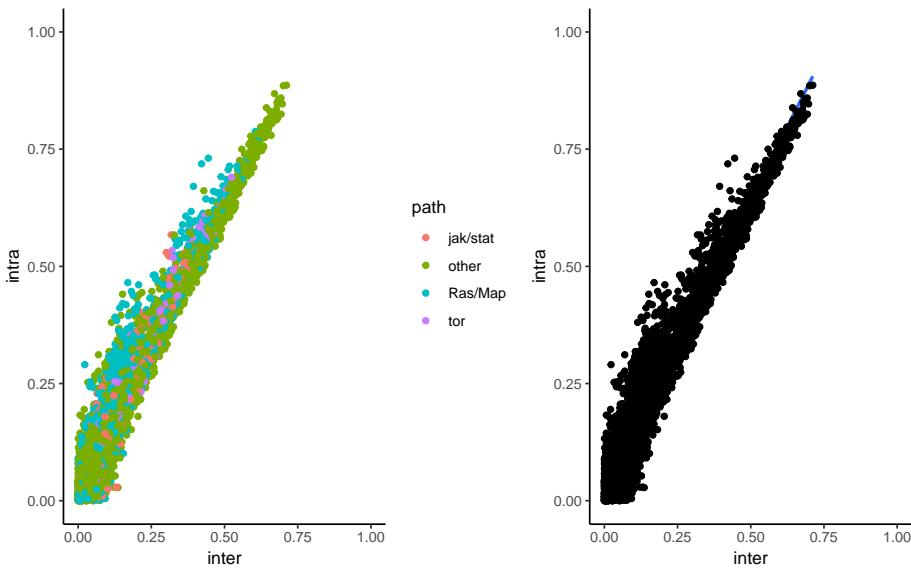
Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
left_join(query_annotation %>%
  dplyr::select(query,querypathway)) %>%
  mutate(querypathway=ifelse(querypathway %in% c("Ras/Map","jak/stat","tor"),querypathway,"other")) %>%
  mutate(path=ifelse(querypathway %in% c("Ras/Map","jak/stat","tor"),querypathway,ifelse(targetpathway %in%
  ungroup() %>%
  ggplot(aes(x=inter,y=intra,text=paste0(target,query))) +
  geom_point(aes(col=path)) +
  # geom_text_repel() +
  xlim(0,1) +
  ylim(0,1) +
  theme_classic()
## Joining, by = "target"
## Joining, by = "query"

b<-final_intra_inner_corr %>%
  dplyr::select(target=gene1,query=gene2,inter,intra) %>%
  left_join(target_annotation %>%
  dplyr::select(target,targetpathway)) %>%
  mutate(targetpathway=ifelse(targetpathway %in% c("Ras/Map","jak/stat","tor"),targetpathway,"other")) %>%
  left_join(query_annotation %>%
  dplyr::select(query,querypathway)) %>%
  mutate(querypathway=ifelse(querypathway %in% c("Ras/Map","jak/stat","tor"),querypathway,"other")) %>%
  mutate(path=ifelse(querypathway %in% c("Ras/Map","jak/stat","tor"),querypathway,ifelse(targetpathway %in%
  ungroup() %>%
  ggplot(aes(x=inter,y=intra)) +
  geom_smooth(mapping = aes(x=inter,y=intra),method="lm") +
  geom_point() +
  xlim(0,1) +
  ylim(0,1) +
  theme_classic()
## Joining, by = "target"
## Joining, by = "query"

a+b
## Warning: Removed 7656 rows containing missing values (geom_point).
## Warning: Removed 7656 rows containing non-finite values (stat_smooth).
## Warning: Removed 7656 rows containing missing values (geom_point).
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways



37 Supplementary Figure S8: dsRNA correlations

```

sub_df<-my.data.array.norm.reshaped_df %>% filter(target=="Rel",query %in% c("CTRL_1","CTRL_2","CTRL_3","CTRL_4"))

a<-sub_df %>%
  extract(replicate,c("design","rep"),regex = "(des\\d)(rep\\d)") %>%
  group_by(target,feature,design) %>%
  summarise(value=mean(value,na.rm=T)) %>%
  spread(design,value) %>%
  ungroup() %>%
  ggplot(aes(x=des1,y=des2)) +
  geom_point() +
  geom_smooth(method="lm")+
  theme_classic()+
  ggtitle("Rel design correlation")

tocor<-sub_df %>%
  extract(replicate,c("design","rep"),regex = "(des\\d)(rep\\d)") %>%
  group_by(target,feature,design) %>%
  summarise(value=mean(value,na.rm=T)) %>%
  spread(design,value) %>%
  ungroup() %>%
  dplyr::select(x=des1,y=des2)

x<-rcorr(tocor$x,tocor$y,type="p")

sub_df<-my.data.array.norm.reshaped_df %>% filter(target=="pnt", query %in% c("CTRL_1","CTRL_2","CTRL_3","CTRL_4"))

b<-sub_df %>%
  extract(replicate,c("design","rep"),regex = "(des\\d)(rep\\d)") %>%
  group_by(target,feature,design) %>%
  summarise(value=mean(value,na.rm=T)) %>%
  spread(design,value) %>%
  ungroup() %>%
  dplyr::select(x=des1,y=des2)
  
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

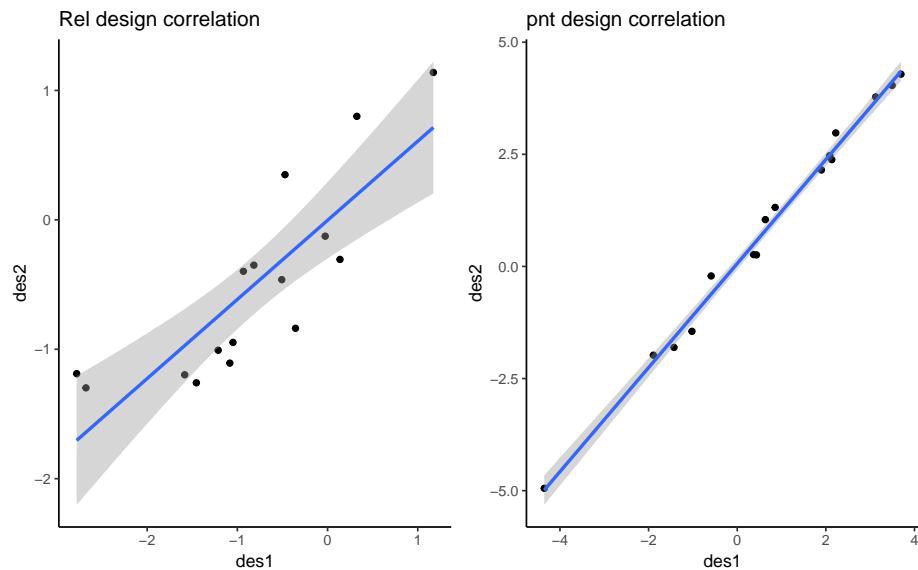
group_by(target,feature,design) %>%
summarise(value=mean(value,na.rm=T)) %>%
spread(design,value) %>%
ungroup() %>%
ggplot(aes(x=des1,y=des2)) +
geom_point() +
geom_smooth(method="lm") +
theme_classic()+
ggtitle("pnt design correlation")

tocor<-sub_df %>%
extract(replicate,c("design","rep"),regex = "(des\\d)(rep\\d)") %>%
group_by(target,feature,design) %>%
summarise(value=mean(value,na.rm=T)) %>%
spread(design,value) %>%
ungroup() %>%
dplyr::select(x=des1,y=des2)

y<-rcorr(tocor$x,tocor$y,type="p")

a+b

```



38 Figure 7C,C': Rel-pnt differential interactions

```

a<-sample_data %>%
mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
filter(target=="Rel",query=="pnt",feature=="cell_number") %>%
ggplot(aes(x=time,y=piscore,colour=treatment)) +
geom_smooth(formula = y~x,method = "rlm",aes(fill=treatment)) +
geom_point() +

```

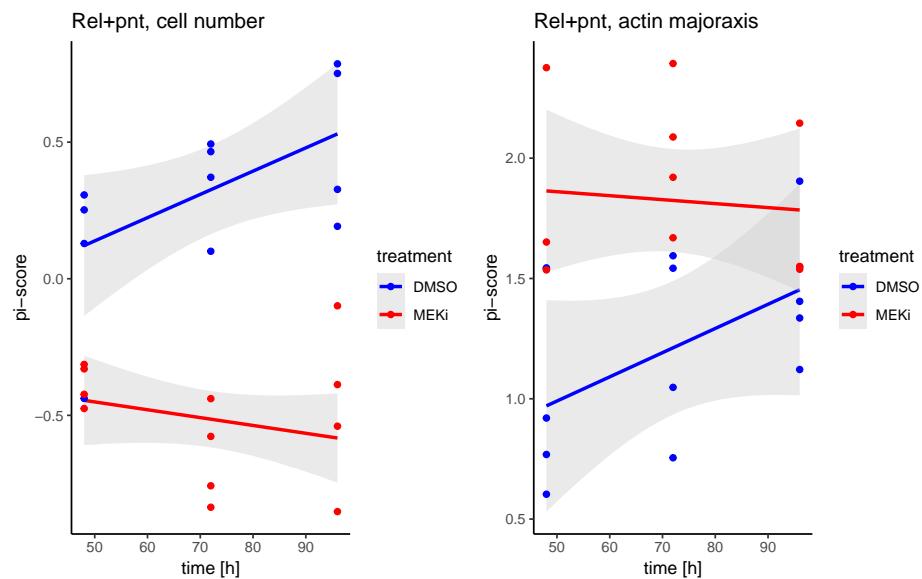
```

theme_classic() +
scale_fill_manual(values = c("DMSO"=rgb(204,204,204,0.5,maxColorValue = 255),"MEKi"=rgb(204,204,204,0.5,maxColorValue = 255))
scale_color_manual(values = c("DMSO"="blue","MEKi"="red")) +
labs(x = "time [h]" , y ="pi-score", title="Rel+pnt, cell number")

b<-sample_data %>%
mutate(time=ifelse(time=="48h",48,ifelse(time=="72h",72,96))) %>%
filter(target=="Rel",query=="pnt",feature=="actin.m.majoraxis.tmean") %>%
ggplot(aes(x=time,y=piScore,colour=treatment)) +
geom_smooth(formula = y~x,method = "rlm",aes(fill=treatment)) +
geom_point() +
theme_classic() +
scale_fill_manual(values = c("DMSO"=rgb(204,204,204,0.5,maxColorValue = 255),"MEKi"=rgb(204,204,204,0.5,maxColorValue = 255))
scale_color_manual(values = c("DMSO"="blue","MEKi"="red")) +
labs(x = "time [h]" , y ="pi-score", title="Rel+pnt, actin majoraxis")

```

a+b



39 Figure 7D,E: Loss of Rel rescues loss of Ras signaling activity by Pvf2 upregulation

```

#read in raw Cp values
data('figure7_qPCR', package='MODIFIdata')

#reformat Cp values
result<-raw_dat %>%
dplyr::select(Cp,Pos,Name) %>%
drop_na() %>%

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

extract(Pos,c("row","col"),regex = "(\w)(\d+)",remove = F) %>%
mutate(sample=
  if_else(col %in% c(1,2,3),"RLUC",
         if_else(col %in% c(4,5,6),"Rel",
                 if_else(col %in% c(7,8,9),"pnt",
                         if_else(col %in% c(10,11,12),"Rel_pnt",
                                 if_else(col %in% c(13,14,15),"Pvr",
                                         if_else(col %in% c(16,17,18),"RasGAP1","none"
                                               )
                                         )
                                     )
                                 )
                             )
                         )
                     )
) %>%
mutate(replicate=
  if_else(col %in% c(1,4,7,10,13,16,19,20,21),1,
         if_else(col %in% c(2,5,8,11,14,17,22,23,24),2,
                 if_else(col %in% c(3,6,9,12,15,18),3,0
                       )
                     )
                   )
) %>%
mutate(primer=
  if_else(row %in% c("A","B","C"),"rps",
         if_else(row %in% c("D","E","F"),"Pvf2",
                 if_else(row %in% c("G","H","I"),"sty",
                         if_else(row %in% c("J","K","L"),"Rel",
                                 if_else(row %in% c("M","N"),"pnt",
                                         if_else(row %in% c("O","P"),"RasGAP1","none"
                                               )
                                             )
                                           )
                                         )
                                       )
                                     )
                                   )
                                 )
) %>%
mutate(Cp=as.numeric(Cp)) %>%
filter(primer != "none",replicate!=0,sample!="none") %>%
dplyr::select(-Pos,-row,-col,-Name) %>%
group_by(sample,primer,replicate) %>%
summarise(Cp=mean(Cp)) %>%
ungroup() %>%
spread(primer,Cp) %>%
mutate(pnt_expression=2^(pnt-rps),
       Rel_expression=2^(Rel-rps),
       Pvf2_expression=2^(Pvf2-rps),
       sty_expression=2^(sty-rps),
       RasGAP1_expression=2^(RasGAP1-rps))

#plot the data in bargraphs for the figure

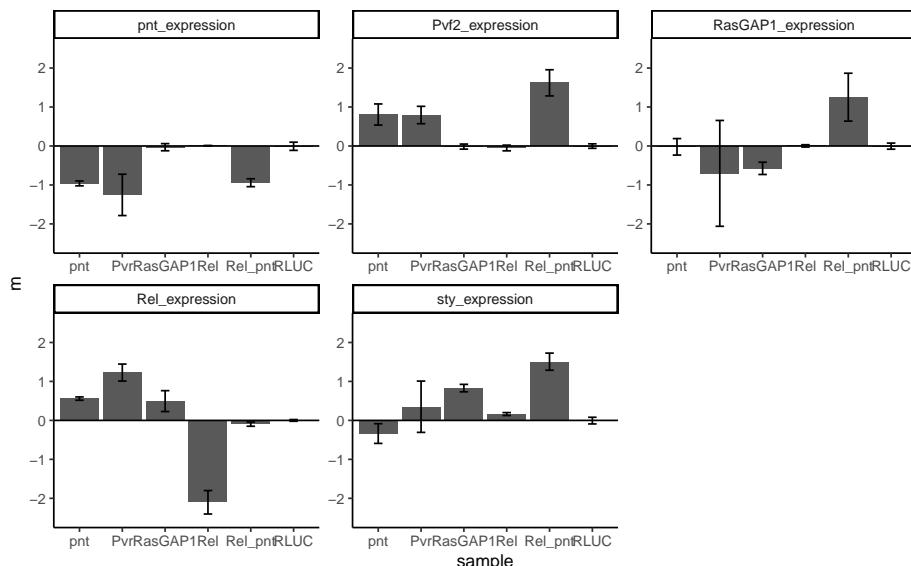
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli
  filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%
  group_by(gene) %>%
  mutate(value=value/mean(value[sample=="RLUC"])) %>%
  group_by(sample,gene) %>%
  mutate(value=log2(value)) %>%
  summarise(m=mean(value),s=sd(value)/sqrt(length(value)),l=m-s,u=m+s) %>%
  ungroup() %>%
  mutate(sample=factor(c(sample),levels=c("pnt","Pvr","RasGAP1","Rel","Rel_pnt","RLUC"))) %>%
  ggplot(aes(x=sample,y=m)) +
  facet_wrap(~gene,scales = "free") +
  geom_bar(stat="identity",position=position_dodge()) +
  geom_errorbar(aes(ymax=u, ymin=l),
                width=.2, # Width of the error bars
                position=position_dodge(.9)) +
  ylim(c(-2.5,2.5)) +
  geom_hline(yintercept = 0) +
  theme_classic()

```



```

#test the statisticss by a two sided t.test
result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli
  filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%
  group_by(gene) %>%
  mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%
  group_by(sample,gene) %>%
  mutate(value=log2(value)) %>%
  filter(gene=="Rel_expression",sample %in% c("RLUC","Rel")) %>%
  t.test(data=.,value~sample)

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
##  
## Welch Two Sample t-test  
##  
## data: value by sample  
## t = -6.9639, df = 2.0217, p-value = 0.01944  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -3.3860539 -0.8161683  
## sample estimates:  
## mean in group Rel mean in group RLUC  
## -2.101450688 -0.000339577  
  
result %>%  
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli  
  filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%  
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%  
  group_by(gene) %>%  
  mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%  
  group_by(sample,gene) %>%  
  mutate(value=log2(value)) %>%  
  filter(gene=="Rel_expression",sample %in% c("RLUC","Pvr")) %>%  
  t.test(data=.,value~sample)  
##  
## Welch Two Sample t-test  
##  
## data: value by sample  
## t = 5.6005, df = 2.0412, p-value = 0.02907  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## 0.3025354 2.1530202  
## sample estimates:  
## mean in group Pvr mean in group RLUC  
## 1.227438201 -0.000339577  
  
result %>%  
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli  
  filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%  
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%  
  group_by(gene) %>%  
  mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%  
  group_by(sample,gene) %>%  
  mutate(value=log2(value)) %>%  
  filter(gene=="Rel_expression",sample %in% c("RLUC","pnt")) %>%  
  t.test(data=.,value~sample)  
##  
## Welch Two Sample t-test  
##  
## data: value by sample  
## t = 11.708, df = 3.0152, p-value = 0.001307  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
##  0.4082181 0.7117819
## sample estimates:
## mean in group pnt mean in group RLUC
##          0.559660423      -0.000339577

result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli
  filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%
  group_by(gene) %>%
  mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%
  group_by(sample,gene) %>%
  mutate(value=log2(value)) %>%
  filter(gene=="pnt_expression",sample %in% c("RLUC","Rel_pnt")) %>%
  t.test(data=.,value~sample)
##
## Welch Two Sample t-test
##
## data: value by sample
## t = -6.3916, df = 3.9959, p-value = 0.003087
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -1.3413207 -0.5286793
## sample estimates:
## mean in group Rel_pnt      mean in group RLUC
##          -0.942706259      -0.007706259

result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli
  filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%
  group_by(gene) %>%
  mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%
  group_by(sample,gene) %>%
  mutate(value=log2(value)) %>%
  filter(gene=="pnt_expression",sample %in% c("RLUC","pnt")) %>%
  t.test(data=.,value~sample)
##
## Welch Two Sample t-test
##
## data: value by sample
## t = -7.7898, df = 3.2603, p-value = 0.003255
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -1.3250231 -0.5805325
## sample estimates:
## mean in group pnt mean in group RLUC
##          -0.960484037      -0.007706259

result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
ungroup() %>% gather(gene,value,-replicate,-sample) %>%
group_by(gene) %>%
mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%
group_by(sample,gene) %>%
mutate(value=log2(value)) %>%
filter(gene=="sty_expression",sample %in% c("RLUC","Rel_pnt")) %>%
t.test(data=.,value~sample)

##
## Welch Two Sample t-test
##
## data: value by sample
## t = 6.4091, df = 2.5933, p-value = 0.01179
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## 0.6901057 2.3343388
## sample estimates:
## mean in group Rel_pnt      mean in group RLUC
##           1.507242184          -0.004980038

result %>%
dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,replicates)
filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
ungroup() %>% gather(gene,value,-replicate,-sample) %>%
group_by(gene) %>%
mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%
group_by(sample,gene) %>%
mutate(value=log2(value)) %>%
filter(gene=="sty_expression",sample %in% c("RLUC","RasGAP1")) %>%
t.test(data=.,value~sample)

##
## Welch Two Sample t-test
##
## data: value by sample
## t = 6.511, df = 3.9548, p-value = 0.002987
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## 0.4770095 1.1918794
## sample estimates:
## mean in group RasGAP1      mean in group RLUC
##           0.829464407          -0.004980038

result %>%
dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,replicates)
filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
ungroup() %>% gather(gene,value,-replicate,-sample) %>%
group_by(gene) %>%
mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%
group_by(sample,gene) %>%
mutate(value=log2(value)) %>%
filter(gene=="RasGAP1_expression",sample %in% c("RLUC","RasGAP1")) %>%
```

```

t.test(data=.,value~sample)
##
## Welch Two Sample t-test
##
## data: value by sample
## t = -3.2414, df = 2.9373, p-value = 0.04925
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -1.136449819 -0.003550181
## sample estimates:
## mean in group RasGAP1      mean in group RLUC
##           -0.574196605          -0.004196605

result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,Pvf2_expression,sty_expression,RasGAP1_expression,repli
  filter(sample %in% c("Rel","pnt","Rel_pnt","Pvr","RasGAP1","RLUC")) %>%
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%
  group_by(gene) %>%
  mutate(value=value/mean(value[sample=="RLUC"],na.rm = T)) %>%
  group_by(sample,gene) %>%
  mutate(value=log2(value)) %>%
  filter(gene=="Pvf2_expression",sample %in% c("RLUC","Rel_pnt")) %>%
  t.test(data=.,value~sample)
##
## Welch Two Sample t-test
##
## data: value by sample
## t = 4.7679, df = 2.118, p-value = 0.03687
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## 0.2338778 3.0127889
## sample estimates:
## mean in group Rel_pnt      mean in group RLUC
##           1.621051716          -0.002281617

```

40 Supplementary Figure S10: Short term expression changes after Rel/Ras signaling co-perturbations

```

#read in raw Cp values
data('figureS10_qPCR', package='MODIFIdata')

#reformat Cp values
result<-raw_data_48h_qpcr %>%
  dplyr::select(Cp,Pos,Name) %>%
  drop_na() %>%
  extract(Pos,c("row","col"),regex = "(\w)(\d+)",remove = F) %>%
  mutate(sample=

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```

if_else(col %in% c(1,2),"RLUC",
       if_else(col %in% c(3,4),"Rel",
              if_else(col %in% c(5,6),"pnt",
                     if_else(col %in% c(7,8),"Rel_pnt",
                            if_else(col %in% c(9,10),"Sos",
                                   if_else(col %in% c(11,12),"Rel_Sos",
                                          if_else(col %in% c(13,14),"rl",
                                                 if_else(col %in% c(15,16),"Rel_rl","none"
                                                       )
                                                     )
                                                   )
                                                 )
                                               )
                                             )
                                           )
                                         )
                                       )
                                     )
                                   )
                                 )
                               )
                             )
                           )
                         )
                       )
                     )
                   )
                 )
               )
             )
           )
         )
       )
     )
   )
 ) %>%
mutate(replicate=
  if_else(col %in% c(1,3,5,7,9,11,13,15),1,
         if_else(col %in% c(2,4,6,8,10,12,14,16),2,0
               )
             )
       )
 ) %>%
mutate(primer=
  if_else(row %in% c("A","B","C"),"rps",
         if_else(row %in% c("D","E","F"),"Rel",
                if_else(row %in% c("G","H","I"),"pnt",
                       if_else(row %in% c("J","K","L"),"rl","none"
                           )
                             )
                           )
                         )
                       )
                     )
                   )
                 )
               )
             )
           )
         )
       )
     )
   )
 ) %>%
mutate(triplicate=
  if_else(row %in% c("A","D","G","J"),1,
         if_else(row %in% c("B","E","H","K"),2,
                if_else(row %in% c("C","F","I","L"),3,0
                      )
                        )
                      )
                    )
                  )
                )
              )
            )
          )
        )
      )
    )
  )
) %>%
mutate(Cp=as.numeric(Cp)) %>%
filter(primer != "none",replicate!=0,sample!="none") %>%
dplyr::select(-Pos,-row,-col,-Name) %>%
group_by(sample,primer,replicate) %>%
summarise(Cp=mean(Cp)) %>%
ungroup() %>%
spread(primer,Cp) %>%
mutate(pnt_expression=2^(pnt-rps),
       Rel_expression=2^(Rel-rps),
       rl_expression=2^(rl-rps)) %>%

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
drop_na()

# Plot relative expression foldchanges normed to non-targeting control and housekeeping

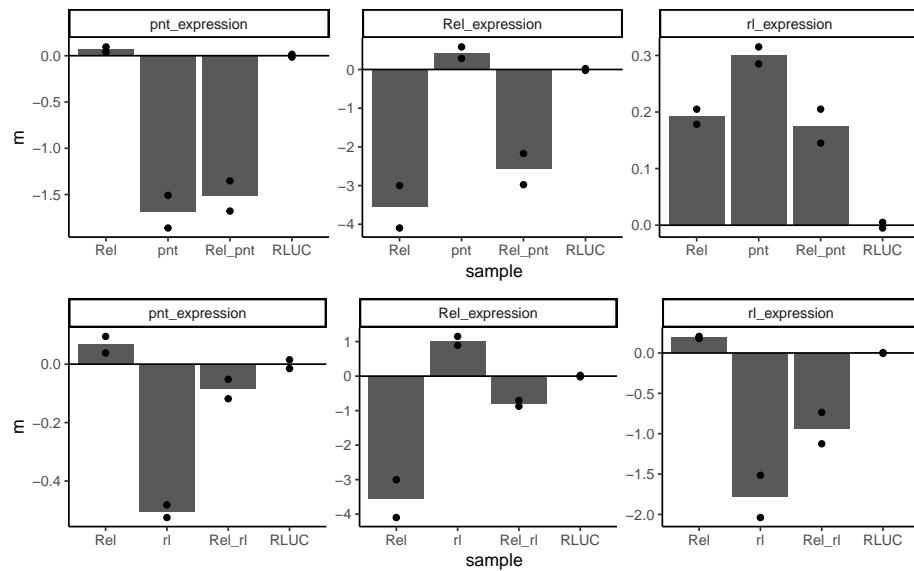
a<-result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,rl_expression,replicate) %>%
  filter(sample %in% c("Rel","pnt","Rel_pnt","RLUC")) %>%
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%
  group_by(gene) %>%
  mutate(value=value/mean(value[sample=="RLUC"])) %>%
  group_by(sample,gene) %>%
  mutate(value=log2(value)) %>%
  summarise(m=mean(value),s=sd(value)/sqrt(length(value)),l=m-s,u=m+s) %>%
  ungroup() %>%
  mutate(sample=factor(c(sample),levels=c("Rel","pnt","Rel_pnt","RLUC"))) %>%
  ggplot(aes(x=sample,y=m)) +
  facet_wrap("gene",scales = "free") +
  geom_bar(stat="identity",position=position_dodge()) +
  geom_point(mapping=aes(sample,value),
             data=result %>%
               dplyr::select(sample,pnt_expression,Rel_expression,rl_expression,replicate) %>%
               filter(sample %in% c("Rel","pnt","Rel_pnt","RLUC")) %>%
               ungroup() %>% gather(gene,value,-replicate,-sample) %>%
               group_by(gene) %>%
               mutate(value=value/mean(value[sample=="RLUC"])) %>%
               group_by(sample,gene) %>%
               mutate(value=log2(value))
            ) +
  geom_hline(yintercept = 0) +
  theme_classic()

b<-result %>%
  dplyr::select(sample,pnt_expression,Rel_expression,rl_expression,replicate) %>%
  filter(sample %in% c("Rel","rl","Rel_rl","RLUC")) %>%
  ungroup() %>% gather(gene,value,-replicate,-sample) %>%
  group_by(gene) %>%
  mutate(value=value/mean(value[sample=="RLUC"])) %>%
  group_by(sample,gene) %>%
  mutate(value=log2(value)) %>%
  summarise(m=mean(value),s=sd(value)/sqrt(length(value)),l=m-s,u=m+s) %>%
  ungroup() %>%
  mutate(sample=factor(c(sample),levels=c("Rel","rl","Rel_rl","RLUC"))) %>%
  ggplot(aes(x=sample,y=m)) +
  facet_wrap("gene",scales = "free") +
  geom_bar(stat="identity",position=position_dodge()) +
  geom_point(mapping=aes(sample,value),
             data=result %>%
               dplyr::select(sample,pnt_expression,Rel_expression,rl_expression,replicate) %>%
               filter(sample %in% c("Rel","rl","Rel_rl","RLUC")) %>%
               ungroup() %>% gather(gene,value,-replicate,-sample) %>%
               group_by(gene) %>%
```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
    mutate(value=value/mean(value[sample=="RLUC"])) %>%
    group_by(sample,gene) %>%
    mutate(value=log2(value))
  ) +
geom_hline(yintercept = 0) +
theme_classic()

a+b+plot_layout(ncol = 1)
```



41 Session info

```
sessionInfo()
## R version 3.4.4 (2018-03-15)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS High Sierra 10.13.6
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] grid      stats     graphics   grDevices  utils      datasets   methods
## [8] base
##
## other attached packages:
## [1] hexbin_1.27.2      bindrcpp_0.2.2      ggjoy_0.4.1
```

```

## [4] ggridges_0.5.0      RColorBrewer_1.1-2 Modifi_0.1.0
## [7] drc_3.0-1           ggrepel_0.8.0     eulerr_4.1.0
## [10] scales_0.5.0.9000   FitAR_1.94       bestglm_0.37
## [13] ltsa_1.4.6         leaps_3.0        patchwork_0.0.1
## [16] pheatmap_1.0.10    taRifx_1.0.6    abind_1.4-5
## [19] Hmisc_4.1-1         Formula_1.2-3   survival_2.42-4
## [22] lattice_0.20-35   reshape2_1.4.3  ggsignif_0.4.0
## [25] sfsmisc_1.1-2     MASS_7.3-50     broom_0.4.5
## [28]forcats_0.3.0      stringr_1.3.1   dplyr_0.7.6
## [31] purrr_0.2.5        readr_1.1.1     tidyverse_1.2.1
## [34] tibble_1.4.2       ggplot2_3.0.0   tidyverse_1.2.1
## [37] BiocStyle_2.6.1
##
## loaded via a namespace (and not attached):
## [1] utf8_1.1.4          tidyselect_0.2.4   RSQLite_2.1.1
## [4] AnnotationDbi_1.40.0 htmlwidgets_1.2      munsell_0.5.0
## [7] codetools_0.2-15    preprocessCore_1.40.0 withr_2.1.2
## [10] colorspace_1.3-2    BiocInstaller_1.28.0 Biobase_2.38.0
## [13] Category_2.44.0    knitr_1.20       rstudioapi_0.7
## [16] stats4_3.4.4       robustbase_0.93-1  splots_1.44.0
## [19] labeling_0.3        polyclip_1.9-0    mnormt_1.5-5
## [22] hwriter_1.3.2     bit64_0.9-7    rprojroot_1.3-2
## [25] TH.data_1.0-8      xfun_0.3        R6_2.2.2
## [28] ICSNP_1.1-1       locfit_1.5-9.1  bitops_1.0-6
## [31] assertthat_0.2.0    multcomp_1.4-8   nnet_7.3-12
## [34] RNAinteract_1.26.0 gtable_0.2.0    affy_1.56.0
## [37] cellHTS2_2.42.0   sandwich_2.4-0  rlang_0.2.1
## [40] genefilter_1.60.0   splines_3.4.4   lazyeval_0.2.1
## [43] acepack_1.4.1     checkmate_1.8.5 yaml_2.1.19
## [46] modelr_0.1.2      backports_1.1.2 RBGL_1.54.0
## [49] tools_3.4.4       bookdown_0.7   psych_1.8.4
## [52] gplots_3.0.1      affyio_1.48.0  HD2013SGI_1.18.0
## [55] BiocGenerics_0.24.0 Rcpp_0.12.17   plyr_1.8.4
## [58] base64enc_0.1-3    zlibbioc_1.24.0 imageHTS_1.28.1
## [61] RCurl_1.95-4.10   rpart_4.1-13   S4Vectors_0.16.0
## [64] zoo_1.8-2          haven_1.1.2    cluster_2.0.7-1
## [67] survey_3.33-2     magrittr_1.5    data.table_1.11.4
## [70] ICS_1.3-1          openxlsx_4.1.0  mvtnorm_1.0-8
## [73] hms_0.4.2          evaluate_0.10.1 fftwtools_0.9-8
## [76] xtable_1.8-2       XML_3.98-1.11  rio_0.5.10
## [79] jpeg_0.1-8          readxl_1.1.0   IRanges_2.12.0
## [82] gridExtra_2.3       compiler_3.4.4 KernSmooth_2.23-15
## [85] crayon_1.3.4       htmltools_0.3.6 pcaPP_1.9-73
## [88] tiff_0.1-5          geneplotter_1.56.0 rrcov_1.4-4
## [91] lubridate_1.7.4    DBI_1.0.0     Matrix_1.2-14
## [94] car_3.0-0          cli_1.0.0     vsn_3.46.0
## [97] gdata_2.18.0        parallel_3.4.4 bindr_0.1.1
## [100] pkgconfig_2.0.1    foreign_0.8-70  xml2_1.2.0
## [103] foreach_1.4.4      annotate_1.56.2 prada_1.54.0
## [106] rvest_0.3.2        digest_0.6.15  pls_2.6-0
## [109] graph_1.56.0       grpreg_3.1-4   rmarkdown_1.10

```

Time-resolved mapping of genetic interactions to model rewiring of signaling pathways

```
## [112] cellranger_1.1.0      htmlTable_1.12      GSEABase_1.40.1
## [115] curl_3.2                 EBImage_4.20.1     gtools_3.8.1
## [118] nlme_3.1-137              jsonlite_1.5       carData_3.0-1
## [121] viridisLite_0.3.0        limma_3.34.9       pillar_1.2.3
## [124] httr_1.3.1                plotrix_3.7-2     DEoptimR_1.0-8
## [127] glue_1.2.0                zip_1.0.0          png_0.1-7
## [130] iterators_1.0.9           glmnet_2.0-16     bit_1.1-14
## [133] class_7.3-14              stringi_1.2.3     blob_1.1.1
## [136] caTools_1.17.1            latticeExtra_0.6-28 memoise_1.1.0
## [139] e1071_1.6-8
```