

Statistical analysis for BDL mice

BDL Analysis

This script contains the complete statistical analysis for the publication Pathobiochemical signatures of cholestatic liver disease in bile duct ligated mice (BMC Systems Biology). Main steps in the analysis are # The analysis consists of regression analysis of the various measured factors based # on different measures. # - Pearson Correlation # - Spearman Correlation # - YS1 and YR1 (time course correlation)

```
baseLoc <- system.file(package="BDLanalysis")
extPath <- file.path(baseLoc, "extdata")
resultsPath <- "/home/mkoenig/git/bdl-analysis/results"
```

Read BDL data

The BDL data, BDL sample definition and the Fluidigm probe definitions are loaded via

```
library("calibrate")
```

Loading required package: MASS

```
library('BDLanalysis')
data(BDLdata)
data(BDLsamples)
data(BDLprobes)
```

Calculation of mean timecourse for factors averaged over the 5 repeats.

```
BDLmean <- bdl_mean_data(BDLdata, BDLsamples)
BDLmean.time <- as.numeric(levels(as.factor(BDLsamples$time)))
```

Create overview plots for all individual factors. These are stored in the resultsPath folder.

```
# Single factor visualization
plot_single_factor(name=colnames(BDLdata)[2])

# Creates plots of all factors in BDLdata
plot_all_factors(path=resultsPath)
```

Actb controls

Actb was measured on all Fluidigm chips and serves as quality control of the measurement/correlation analysis. The pairwise correlation between all Actb measurements should result in high correlation values

```

# Actb control figure
png(filename=file.path(resultsPath, "Actb_control.png"), width=1600, height=600, res=200)
par(mfrow=c(1,3))
plot_cor_pair("Actb", "Actb.x", single_plots=FALSE)
plot_cor_pair("Actb", "Actb.y", single_plots=FALSE)
plot_cor_pair("Actb.x", "Actb.y", single_plots=FALSE)
par(mfrow=c(1,1))
dev.off()

```

```

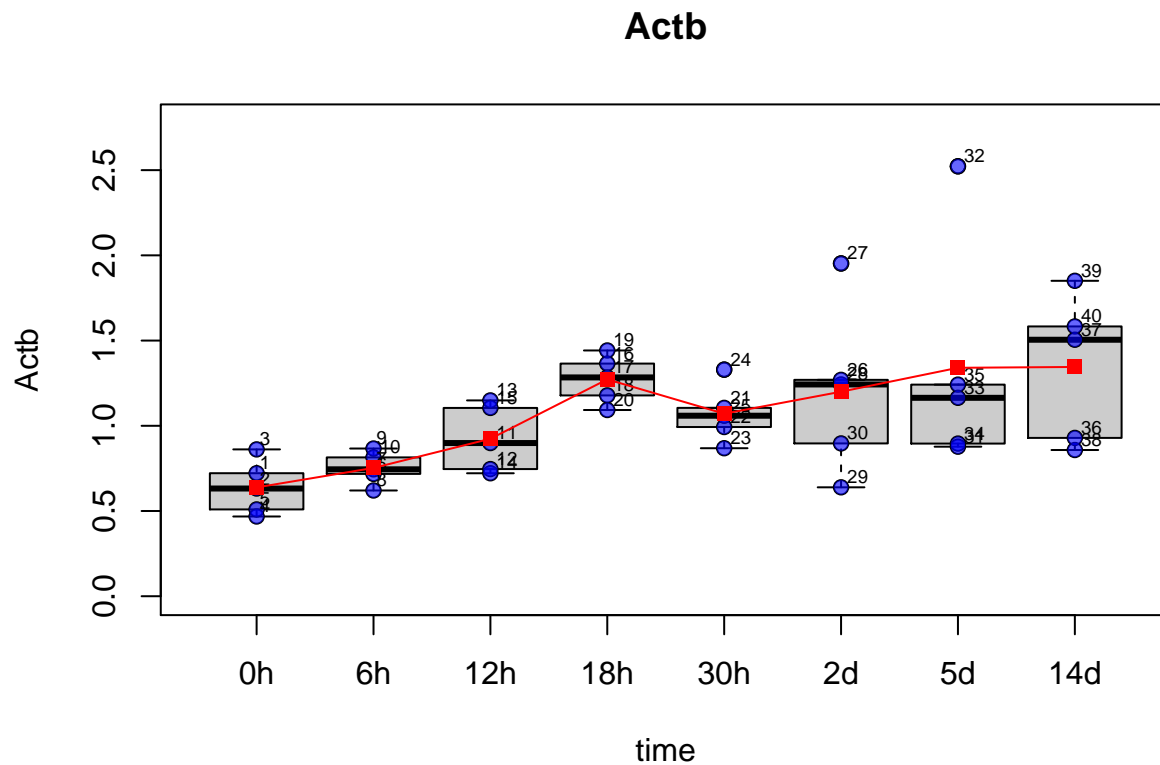
## pdf
## 2

```

```

# calculate the correlations
plot_single("Actb")

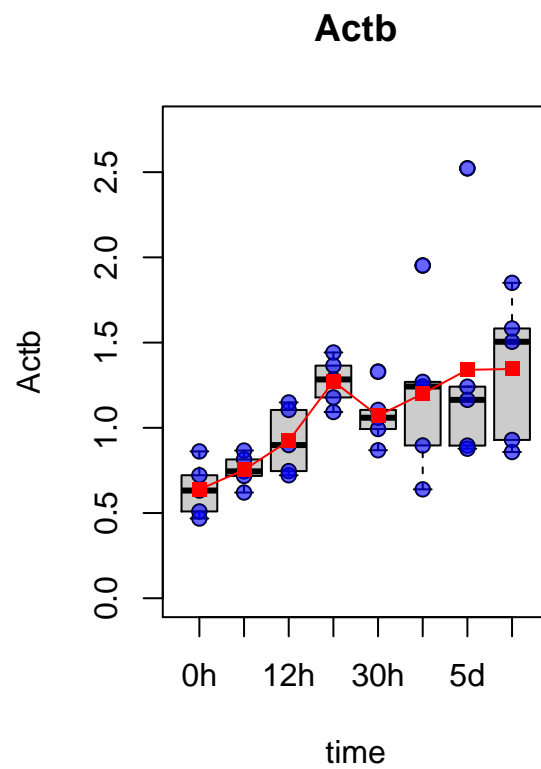
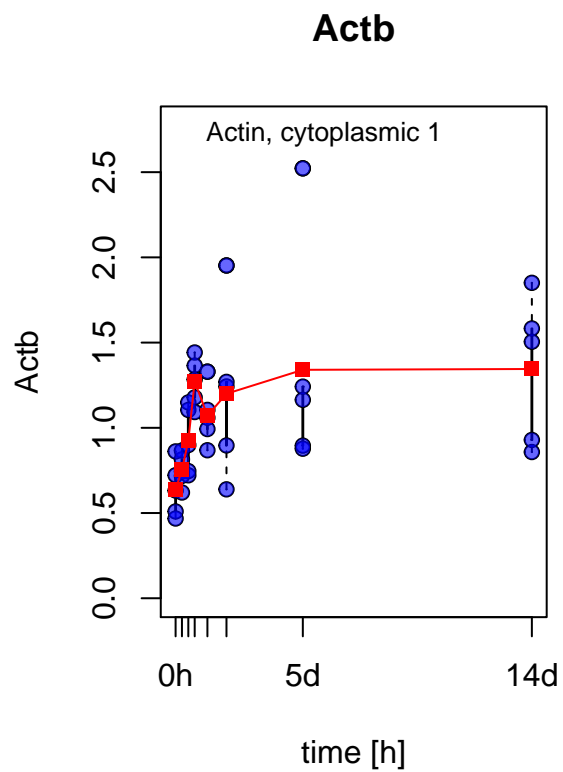
```



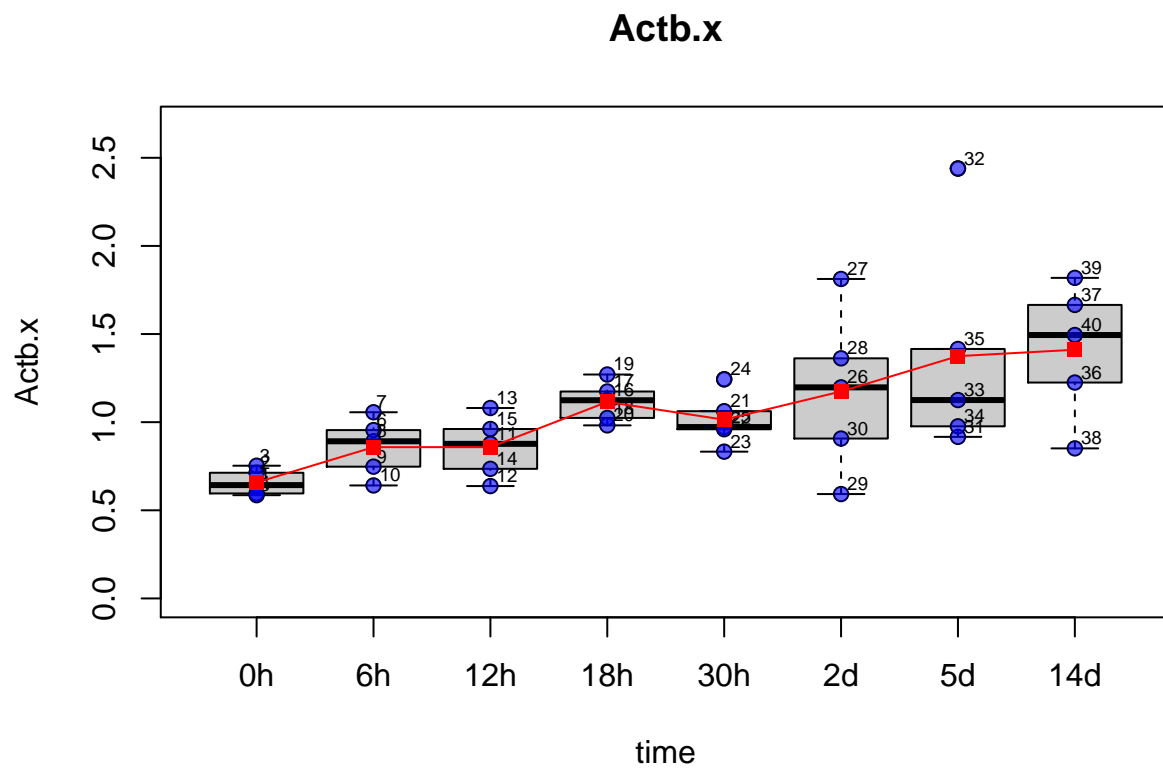
```

plot_single_factor("Actb", path=NULL)

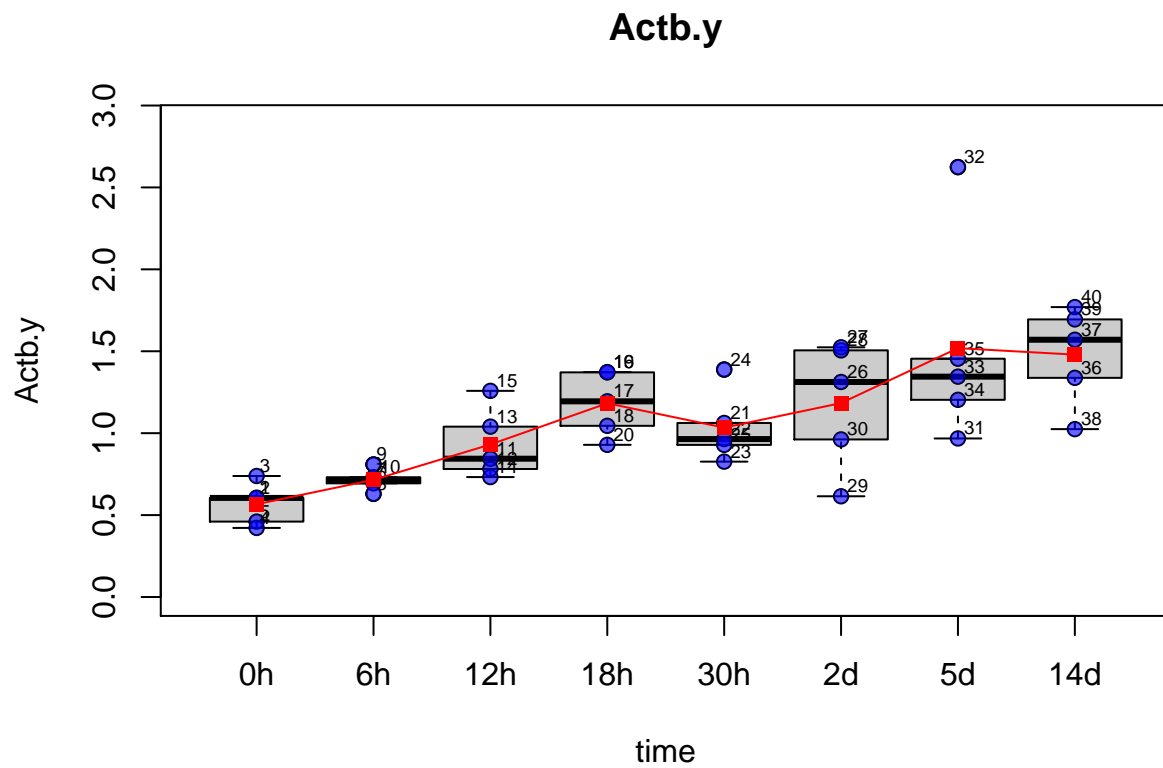
```



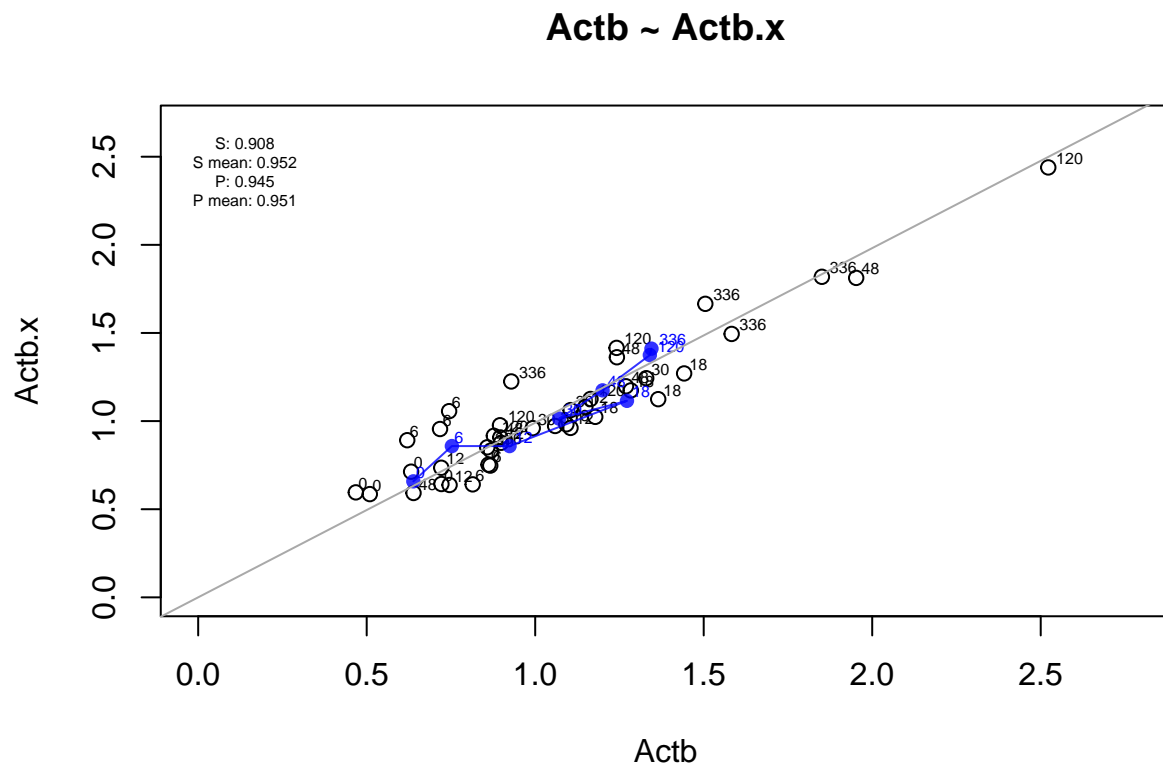
```
plot_single("Actb.x")
```



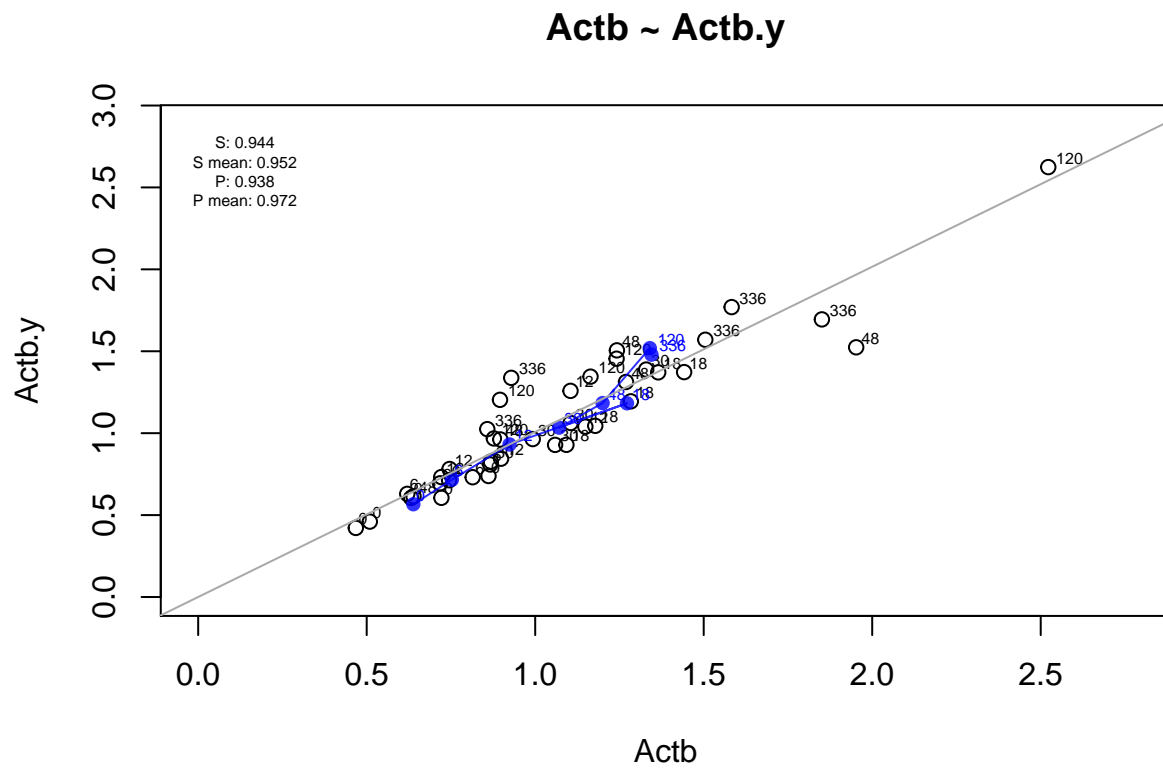
```
plot_single("Actb.y")
```



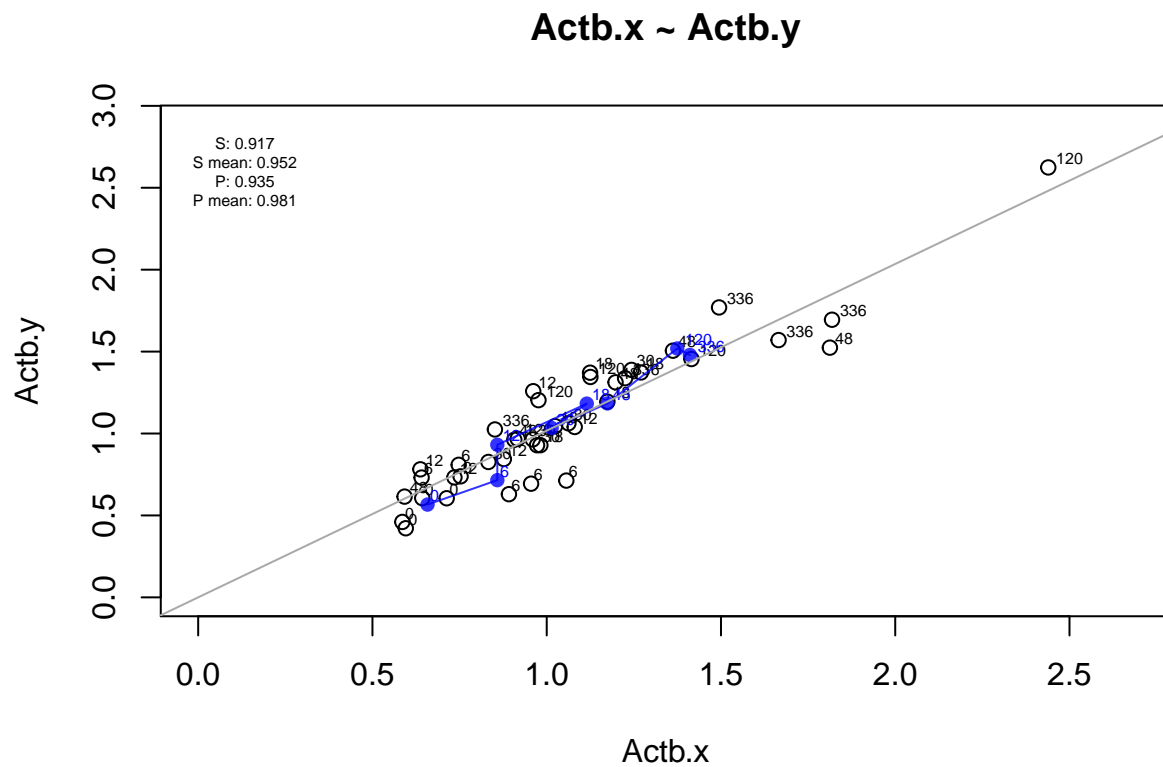
```
plot_cor_pair("Actb", "Actb.x", single_plots=FALSE)
```



```
plot_cor_pair("Actb", "Actb.y", single_plots=FALSE)
```



```
plot_cor_pair("Actb.x", "Actb.y", single_plots=FALSE)
```



```
# calculate Spearman and Pearson correlation coefficients on mean and individual
# data
```

```
actb.spearman <- cor(data.frame(Actb=BDLdata$Actb,
                                Actb.x=BDLdata$Actb.x,
                                Actb.y=BDLdata$Actb.y), method="spearman")

actb.spearman.mean <- cor(data.frame(Actb=BDLmean$Actb,
                                      Actb.x=BDLmean$Actb.x,
                                      Actb.y=BDLmean$Actb.y), method="spearman")

actb.pearson <- cor(data.frame(Actb=BDLdata$Actb,
                                Actb.x=BDLdata$Actb.x,
                                Actb.y=BDLdata$Actb.y), method="pearson")

actb.pearson.mean <- cor(data.frame(Actb=BDLmean$Actb,
                                      Actb.x=BDLmean$Actb.x,
                                      Actb.y=BDLmean$Actb.y), method="pearson")

print(actb.spearman)
```

```
##           Actb    Actb.x    Actb.y
## Actb    1.0000000 0.9076923 0.9435272
## Actb.x  0.9076923 1.0000000 0.9170732
## Actb.y  0.9435272 0.9170732 1.0000000
```

```
print(actb.spearman.mean)
```

```
##           Actb    Actb.x    Actb.y
## Actb    1.0000000 0.952381 0.952381
## Actb.x  0.952381 1.0000000 0.952381
## Actb.y  0.952381 0.952381 1.0000000
```

```
print(actb.pearson)
```

```
##           Actb    Actb.x    Actb.y
## Actb    1.0000000 0.9445285 0.9377102
## Actb.x  0.9445285 1.0000000 0.9345367
## Actb.y  0.9377102 0.9345367 1.0000000
```

```
print(actb.pearson.mean)
```

```
##           Actb    Actb.x    Actb.y
## Actb    1.0000000 0.9511591 0.9722055
## Actb.x  0.9511591 1.0000000 0.9810189
## Actb.y  0.9722055 0.9810189 1.0000000
```

Heatmap of the full data set provides an overview over the data quality

```
library("gplots")
```

```
colors <- HeatmapColors()
heatmap.2(t(as.matrix(BDLdata)), col=colors(100), scale="row", Rowv=NULL, Colv=NULL,
          key=TRUE, trace="none", cexRow=0.5, keysize=0.8)
```



Read BDL data

The BDLdata set of all timecourses for the factors is reshaped into matrix form for the ANOVA calculation

```
BDLmatrices <- bdl_matrix_data(BDLdata, BDLsamples)
print(BDLmatrices[[1]])
```

```
##           R1           R2           R3           R4           R5
## 0h  1.4215683 2.0292143 1.8044804 2.5907449 1.5619353
## 6h  2.0224053 1.4065564 1.2302870 1.1687865 1.4883115
## 12h 0.8251309 1.1925449 1.1471248 1.5450861 0.9355473
## 18h 0.9570258 0.7432429 1.0102834 0.8211681 0.9952273
## 30h 0.9151403 0.9153196 0.6373844 1.0636297 1.1712628
## 2d  0.5448800 0.4740537 0.7814212 0.7423229 0.9027864
## 5d  0.6008619 0.6509884 0.5077800 0.8952370 1.2653993
## 14d 1.0690760 1.1632259 0.9085463 0.7040999 0.5053402
```

Dimension reduction via ANOVA

A one-way analysis of variance (ANOVA) was applied to isolate factors showing significant ($p_{adjusted} < 0.05$) up- or down-regulation during the time course, using the Holm's procedure to correct for any artificial p-value inflation. In its simplest form, ANOVA provides a statistical test of whether or not the means of several groups are equal, and therefore generalizes the t-test to more than two groups, with the groups being the sampled time points. For every of the individual factors in the BDL data set an ANOVA was calculated. Dimension reduction of the BDL data set was then performed by filtering out factors which did not significantly change over time.

An ANOVA is calculated for all factors (N=40). A data.frame with the p-values of the ANOVA is returned which is used for filtering the BDL data set. Significance codes are added for visual inspection.

Adjust p-values for multiple testing

A multitude of tests were performed, namely an ANOVA for every single factor. Consequently, the reported p-values of the ANOVA have to be adjusted via multiple testing procedures. Using the p.adjust function which given a set of p-values, returns p-values adjusted using one of several methods. The Bonferroni, Holm, Hochberg, Hommel are designed to give strong control of the family-wise error rate. There seems no reason to use the unmodified Bonferroni correction because it is dominated by Holm's method, which is also valid under arbitrary assumptions.

The correction is performed using Holm

Holm, S. (1979). A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics 6, 65-70.

TODO: Add the code for single factor ANOVA

The ANOVA is now calculated for every single factor


```

df.anova <- all_factor_anova()
df.anova$sig <- sapply(df.anova$p.value, significant_code) # add significant codes

df.anova$p.holm <- p.adjust(df.anova$p.value, method="holm" , n = length(df.anova$p.value))
df.anova$sig.holm <- sapply(df.anova$p.holm, significant_code)

# order the results py the adjusted p-values
df.anova.ordered <- df.anova[with(df.anova, order(p.holm)), ]
df.anova.ordered

```

##	factors	p.value	sig	p.holm	sig.holm
## 8	Cyp1a2	1.913423e-16	***	2.946671e-14	***
## 144	bilirubin	3.273960e-14	***	5.009158e-12	***
## 71	Il10rb	7.639824e-14	***	1.161253e-11	***
## 60	Tgfb1	2.197248e-13	***	3.317844e-11	***
## 48	Cc12	2.320015e-13	***	3.480023e-11	***
## 50	Cd86	4.433981e-13	***	6.606632e-11	***
## 79	Ccr2	4.720799e-13	***	6.986782e-11	***
## 85	Mrc1	4.762017e-13	***	7.000165e-11	***
## 67	Tnfrsf1b	5.443030e-13	***	7.946824e-11	***
## 56	Cxcl5	4.345645e-12	***	6.301185e-10	***
## 153	CTGF	5.466353e-12	***	7.871548e-10	***
## 77	Il10ra	1.154771e-11	***	1.651322e-09	***
## 17	Gstm1	6.513806e-11	***	9.249604e-09	***
## 68	Cc17	2.430883e-10	***	3.427546e-08	***
## 86	Ccr5	3.139262e-10	***	4.394967e-08	***
## 81	Hgf	4.203344e-10	***	5.842648e-08	***
## 59	Osmr	7.389499e-10	***	1.019751e-07	***
## 62	Cc14	7.670907e-10	***	1.050914e-07	***
## 38	Nr0b2	9.608293e-10	***	1.306728e-07	***
## 104	Tgfbr2	1.276990e-09	***	1.723937e-07	***
## 149	BrdU_HSC	1.569640e-09	***	2.103318e-07	***
## 63	Cc15	2.104076e-09	***	2.798421e-07	***
## 99	Col1a1	3.356632e-09	***	4.430755e-07	***
## 58	Ifnar1	5.933095e-09	***	7.772354e-07	***
## 154	S100A4	7.122988e-09	***	9.259885e-07	***
## 98	Sparc	8.383632e-09	***	1.081489e-06	***
## 138	Cyp2e1	1.181064e-08	***	1.511761e-06	***
## 74	Cxcr2	1.390313e-08	***	1.765697e-06	***
## 64	Ccr3	1.524189e-08	***	1.920479e-06	***
## 70	Cd69	2.200276e-08	***	2.750345e-06	***
## 47	Cyp2c29	2.384128e-08	***	2.956319e-06	***
## 23	Gsta2	3.180374e-08	***	3.911860e-06	***
## 76	Tnf	3.602497e-08	***	4.395046e-06	***
## 117	Gdf2	4.958043e-08	***	5.999232e-06	***
## 54	Il1b	6.128729e-08	***	7.354474e-06	***
## 61	Ifng	6.367290e-08	***	7.575986e-06	***
## 69	Osm	6.366375e-08	***	7.575986e-06	***
## 87	Cc13	7.354920e-08	***	8.605256e-06	***
## 66	Il13	8.081570e-08	***	9.374621e-06	***
## 57	Cxcr1	9.412831e-08	***	1.082476e-05	***
## 35	Cyp2c37	9.650268e-08	***	1.100131e-05	***
## 73	Cd14	1.028168e-07	***	1.161830e-05	***

## 137	Col3a1	1.816561e-07	***	2.034548e-05	***
## 53	Tnfrsf1a	3.028259e-07	***	3.361367e-05	***
## 72	Il2	4.437114e-07	***	4.880825e-05	***
## 49	Ifnb1	4.504193e-07	***	4.909571e-05	***
## 80	Egf	4.556800e-07	***	4.921344e-05	***
## 90	Il4	4.601996e-07	***	4.924136e-05	***
## 52	Il28b	4.645752e-07	***	4.924497e-05	***
## 78	Il10	4.666371e-07	***	4.924497e-05	***
## 22	Slc10a1	5.102225e-07	***	5.306314e-05	***
## 114	Timp2	6.116144e-07	***	6.299628e-05	***
## 93	Cxcl3	6.750161e-07	***	6.885164e-05	***
## 92	Ccl8	1.232797e-06	***	1.245125e-04	***
## 119	Ctgf	1.336225e-06	***	1.336225e-04	***
## 11	Gstp1	1.409173e-06	***	1.395082e-04	***
## 1	Ppara	1.683106e-06	***	1.649444e-04	***
## 83	Ifnar2	1.898212e-06	***	1.841266e-04	***
## 88	Il6	2.352096e-06	***	2.258012e-04	***
## 55	Il17a	2.580378e-06	***	2.451359e-04	***
## 103	Bad	4.125226e-06	***	3.877712e-04	***
## 107	Timp1	4.678453e-06	***	4.350961e-04	***
## 113	Cdh1	4.919990e-06	***	4.526391e-04	***
## 13	Cebpa	5.473439e-06	***	4.980829e-04	***
## 152	alpha.SMA	5.638004e-06	***	5.074204e-04	***
## 147	BrdU_NHC	6.005016e-06	***	5.344464e-04	***
## 123	Cdh2	6.179147e-06	***	5.437650e-04	***
## 150	BrdU_SirirusRed	8.819839e-06	***	7.673260e-04	***
## 110	Pdgfb	9.032454e-06	***	7.767911e-04	***
## 94	Il6st	1.037076e-05	***	8.815149e-04	***
## 125	Fn1	1.457762e-05	***	1.224520e-03	**
## 127	Mki67	1.774623e-05	***	1.472937e-03	**
## 82	Ifna1	1.805029e-05	***	1.480124e-03	**
## 91	Egfr	1.988312e-05	***	1.610533e-03	**
## 148	BrdU_Kupffer	2.470612e-05	***	1.976489e-03	**
## 121	Tnc	2.611215e-05	***	2.062860e-03	**
## 30	Ugt1a1	3.570685e-05	***	2.785135e-03	**
## 16	Sult1a1	3.922274e-05	***	3.020151e-03	**
## 142	GLDH	5.711856e-05	***	4.341011e-03	**
## 120	Notch1	6.016402e-05	***	4.512302e-03	**
## 51	Met	6.803468e-05	***	5.034566e-03	**
## 29	Cyp7a1	1.365100e-04	***	9.965231e-03	**
## 3	Cyp24a1	1.391831e-04	***	1.002118e-02	*
## 133	Tgfb2	1.542364e-04	***	1.095078e-02	*
## 97	Birc5	2.454637e-04	***	1.718246e-02	*
## 89	Actb.y	2.806640e-04	***	1.936582e-02	*
## 116	Bak1	4.077548e-04	***	2.772733e-02	*
## 122	Bax	4.371327e-04	***	2.928789e-02	*
## 105	Rarres1	5.491381e-04	***	3.624311e-02	*
## 151	bileInfarcts	5.904178e-04	***	3.837715e-02	*
## 2	Cyp3a11	8.410363e-04	***	5.382632e-02	.
## 10	Sult1b1	9.257764e-04	***	5.832392e-02	.
## 18	Cyp4a10	1.124446e-03	**	6.903882e-02	.
## 95	Pparg	1.113529e-03	**	6.903882e-02	.
## 44	Hk2	1.271173e-03	**	7.627035e-02	.
## 143	ALT	1.515972e-03	**	8.944232e-02	.

## 140	Smad6	1.665654e-03	** 9.660796e-02
## 39	Cyp2b10	2.093404e-03	** 1.193240e-01
## 146	BrdU_HC	2.310442e-03	** 1.293848e-01
## 65	Il1rn	2.507326e-03	** 1.379030e-01
## 108	Nes	2.590456e-03	** 1.398846e-01
## 5	Nfkb1a	3.024538e-03	** 1.603005e-01
## 139	Rps18	3.292204e-03	** 1.711946e-01
## 12	Cxcl15	4.356699e-03	** 2.221916e-01
## 25	Socs3	4.981022e-03	** 2.475551e-01
## 26	Vdr	4.951101e-03	** 2.475551e-01
## 96	Edn1	5.022042e-03	** 2.475551e-01
## 40	Nr2f1	5.717984e-03	** 2.687452e-01
## 32	Abcg2	6.283587e-03	** 2.883192e-01
## 46	Nr1i3	6.267808e-03	** 2.883192e-01
## 43	Hnf4a	6.700157e-03	** 2.948069e-01
## 134	Mmp10	7.347455e-03	** 3.159406e-01
## 45	Ptgs2	8.168008e-03	** 3.430563e-01
## 109	Bcl2l11	8.338430e-03	** 3.430563e-01
## 31	Socs1	8.654611e-03	** 3.461844e-01
## 132	Pten	9.397086e-03	** 3.664864e-01
## 42	Actb.x	1.025240e-02	* 3.895911e-01
## 84	Cxcl2	1.098554e-02	* 4.064650e-01
## 111	Xiap	1.124710e-02	* 4.064650e-01
## 118	Pde4a	1.263533e-02	* 4.422367e-01
## 24	Dpyd	1.337053e-02	* 4.432805e-01
## 75	Cxcl1	1.303766e-02	* 4.432805e-01
## 106	Lama1	1.428488e-02	* 4.571161e-01
## 100	Col8a1	1.634422e-02	* 5.066710e-01
## 128	Prom1	1.870355e-02	* 5.543027e-01
## 136	Actb	1.847676e-02	* 5.543027e-01
## 20	Ahr	1.989103e-02	* 5.569489e-01
## 27	Nr2f2	2.048348e-02	* 5.569489e-01
## 4	Nos2	2.237430e-02	* 5.817317e-01
## 112	Notch3	2.905904e-02	* 7.264759e-01
## 21	Cebpd	3.036439e-02	* 7.287453e-01
## 19	Hmox1	3.343725e-02	* 7.690567e-01
## 41	Cyp2d22	3.800488e-02	* 8.361075e-01
## 101	Igf1	4.658924e-02	* 9.783741e-01
## 102	Fas1	4.854110e-02	* 9.783741e-01
## 6	Cyp2c39	1.057596e-01	1.000000e+00
## 7	Nr3c1	1.299624e-01	1.000000e+00
## 9	Abcb1a	2.850937e-01	1.000000e+00
## 14	Nr1h3	6.326127e-01	1.000000e+00
## 15	Sod2	1.208096e-01	1.000000e+00
## 28	Rxra	8.615924e-02	. 1.000000e+00
## 33	Abcc2	5.803779e-02	. 1.000000e+00
## 34	Por	4.054924e-01	1.000000e+00
## 36	Nr1i2	6.889241e-02	. 1.000000e+00
## 37	Cebpb	1.605796e-01	1.000000e+00
## 115	Wisp1	1.651084e-01	1.000000e+00
## 124	Pde4b	2.641945e-01	1.000000e+00
## 126	Smad7	1.170115e-01	1.000000e+00
## 129	Ch25h	1.296172e-01	1.000000e+00
## 130	Pde4d	1.668398e-01	1.000000e+00

```
## 131      Acta2 2.365621e-01      1.000000e+00
## 135      Col4a3 6.313442e-02      1.000000e+00
## 141      Col6a6 1.370452e-01      1.000000e+00
## 145      albumin 3.546133e-01      1.000000e+00
```

```
# save the results
write.table(df.anova.ordered, file=file.path(resultsPath, 'BDLanova.csv'), sep="\t", quote=FALSE)
BDLanova <- df.anova
save(df.anova, file=file.path(resultsPath, "BDLanova.Rdata"))
```

Filter factors

The factors are filtered based on acceptance level, with the cutoff for the adjusted p-value being p_{accept} . I.e. all factors with a ANOVA with $p_{adjusted} \geq p_{accept}$ are filtered out. The filtered data sets are generated.

```
p.accept = 0.05 # acceptance level
idx.accept = (df.anova$p.holm < p.accept) # accepted subset

# accepted
table(df.anova$p.holm<p.accept) # 64 rejected / 90 accepted (adjusted)
```

```
##
## FALSE  TRUE
##      64    90
```

```
table(df.anova$p.value<p.accept) # 19 rejected / 135 accepted (unadjusted)
```

```
##
## FALSE  TRUE
##      19   135
```

```
# subset of filtered data
BDLdata.fil <- BDLdata[, idx.accept]
BDLmean.fil <- BDLdata[, idx.accept]
```

Heatmap of the filtered BDL data.
 TODO: add the view of the Chips and histopathology

```
library('ALL')
```

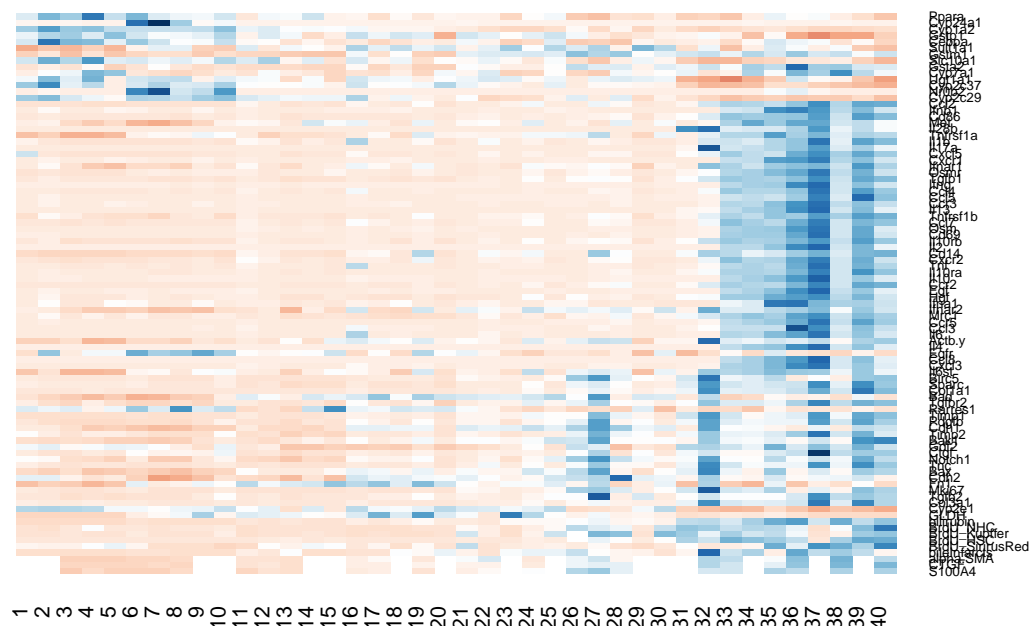
```
## Loading required package: Biobase
## Loading required package: BiocGenerics
## Loading required package: parallel
##
## Attaching package: 'BiocGenerics'
##
## The following objects are masked from 'package:parallel':
##
##      clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##      clusterExport, clusterMap, parApply, parCapply, parLapply,
```

```
##      parLapplyLB, parRapply, parSapply, parSapplyLB
##
## The following object is masked from 'package:stats':
##
##      xtabs
##
## The following objects are masked from 'package:base':
##
##      anyDuplicated, append, as.data.frame, as.vector, cbind,
##      colnames, do.call, duplicated, eval, evalq, Filter, Find, get,
##      intersect, is.unsorted, lapply, Map, mapply, match, mget,
##      order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##      rbind, Reduce, rep.int, rownames, sapply, setdiff, sort,
##      table, tapply, union, unique, unlist, unsplit
##
## Welcome to Bioconductor
##
##      Vignettes contain introductory material; view with
##      'browseVignettes()'. To cite Bioconductor, see
##      'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
# plot of the data subset which is used for the correlation analysis
col2 <- colorRampPalette(c("#67001F", "#B2182B", "#D6604D", "#F4A582", "#FDDBC7",
                           "#FFFFFF", "#D1E5F0", "#92C5DE", "#4393C3", "#2166AC", "#053061"))
heatmap.2(t(as.matrix(BDLdata.fil)), col=col2(100), scale="row", Rowv=FALSE, Colv=FALSE,
          key=TRUE, trace="none", cexRow=0.5, keysize=0.8)
```

```
## Warning in heatmap.2(t(as.matrix(BDLdata.fil)), col = col2(100), scale =
## "row", : Discrepancy: Rowv is FALSE, while dendrogram is `none`. Omitting
## row dendrogram.
```

```
## Error in plot.new(): figure margins too large
```



```
png(filename=file.path(resultsPath, "BDLdata.fil.png"), width=1600, height=1600, res=200)
heatmap.2(t(as.matrix(BDLdata.fil)), col=col2(100), scale="row", Rowv=NULL, Colv=NULL,
          key=TRUE, trace="none", cexRow=0.5, keysize=0.8)
```

```
## Warning in heatmap.2(t(as.matrix(BDLdata.fil)), col = col2(100), scale =
## "row", : Discrepancy: Rowv is FALSE, while dendrogram is `none`. Omitting
## row dendrogram.
```

```
dev.off()
```

```
## pdf
## 2
```

Correlation analysis

The correlation and cluster analysis uses a correlation measure for time series data in combination with Complete-Linkage hierarchical clustering, which provided the best enrichments on gene-expression time-series in a recent comparisons of methods {Jaskowiak2014, Jaskowiak2013}. The time-course experiment includes $n = 8$ time points denoted by t_1, \dots, t_n consisting of a total $p = 154$ factors, with $N = 5$ repeats per time point. Correlation analysis between factors i and j ($i, j = 1, \dots, p$) was performed using a modified correlation coefficient based similarity measure developed for clustering of time-course data ($Y_{i,j}^{S2}$ and $Y_{i,j}^{R2}$) {Son2008}. $Y_{i,j}^{S2}$ and $Y_{i,j}^{R2}$ are linear combinations of a classical correlation part $R_{i,j}$ (Pearson) or $S_{i,j}$ (Spearman), a component $A_{i,j}$ accounting for the similarity in changes between the time courses and a component $M_{i,j}$ comparing the location of minimum and maximum $Y_{i,j}^{S2} = 1S_{i,j} + 1A_{i,j} + 2M_{i,j}$, $Y_{i,j}^{R2} = 1R_{i,j} + 1A_{i,j} + 2M_{i,j}$, with $R_{i,j}$ and $S_{i,j}$ being calculated on the individual data for factor i and j , $A_{i,j}$ and $M_{i,j}$ on the mean time courses, averaged over the N replicates. $Y_{i,j}^{S2}$ and $Y_{i,j}^{R2}$ were extended in a simple manner to account for the non-equidistant time points $t_1, \dots, t_n = 0h, 6h, 12h, 18h, 30h, 2d, 5d, 14d$ in the study design $Y_{i,j}^{S3} = 1S_{i,j} + 1A_{i,j} + 2M_{i,j}$, $Y_{i,j}^{R3} = 1R_{i,j} + 1A_{i,j} + 2M_{i,j}$. Herein, $A_{i,j}$ calculates the correlation of slopes analogue to the correlation in distances in $A_{i,j}^*$ between factors i and j $A_{i,j}^* = (\text{Pearson correlation}(s_i, s_j) + 1)/2$ with $s_i = (s_{i1}, s_{i2}, \dots, s_{i(n-1)})$ being the vector of slopes $s_{ik} = s(i, t_k, t_{k+1}) = x_{i,t_k} - x_{i,t_{k+1}} - t_k M_{i,j}^*$ calculates the absolute distance in maximum and minimum times instead of the distance of indices in $M_{i,j}^* M_{i,j}^{**} = 1 - |t_{i\min} - t_{j\min}| + |t_{i\max} - t_{j\max}|^2 / (t_n - t_0)$. Throughout the analysis the weights were For comparison Pearson, Spearman and $Y_{i,j}^{S2}$ and $Y_{i,j}^{R2}$ correlation coefficients were calculated. See Supporting Information for details. All computations were performed in R with all source code and data provided in the supplement.

Correlation matrices are calculated using a modified correlation score for the analysis of time course data. Standard Pearson and Spearman scores are calculated as reference values and for comparison.

Based on the correlation scores hierarchical clustering is performed using complete linkage (hclust).

Spearman & Pearson correlation matrices are plotted either in normal order or based on the reordering via hierarchical clustering

```
# correlation matrix
require(corrplot)
```

```
## Loading required package: corrplot
```

```
cor.pearson <- cor(BDLdata.fil, method="pearson", use="pairwise.complete.obs")
cor.spearman <- cor(BDLdata.fil, method="spearman", use="pairwise.complete.obs")
```

```
# Helper function for creating correlation plot and saving to results folder.
```

```
f_corrplot <- function(name, data, order, folder="./results",
                      width=1600, height=1600, res=200){
  fname <- sprintf("%s_%.png", name, order)
  col2 <- HeatmapColors()
  if (!is.null(folder)){
    png(filename=file.path(folder, "correlation", fname), width=width, height=height, res=res)
  }
  corplot(data, order=order, hclust.method="complete", method="color", type="full",
          tl.cex=0.3, tl.col="black", col=col2(10))
  if (!is.null(folder)){
    invisible(dev.off())
  }
}
```

Spearman

```
f_corrplot("cor.spearman", data=cor.spearman, order="original", folder=resultsPath)
```

```
f_corrplot("cor.spearman", data=cor.spearman, order="hclust", folder=resultsPath)
```

Pearson

```
f_corrplot("cor.pearson", data=cor.pearson, order="original", folder=resultsPath)
```

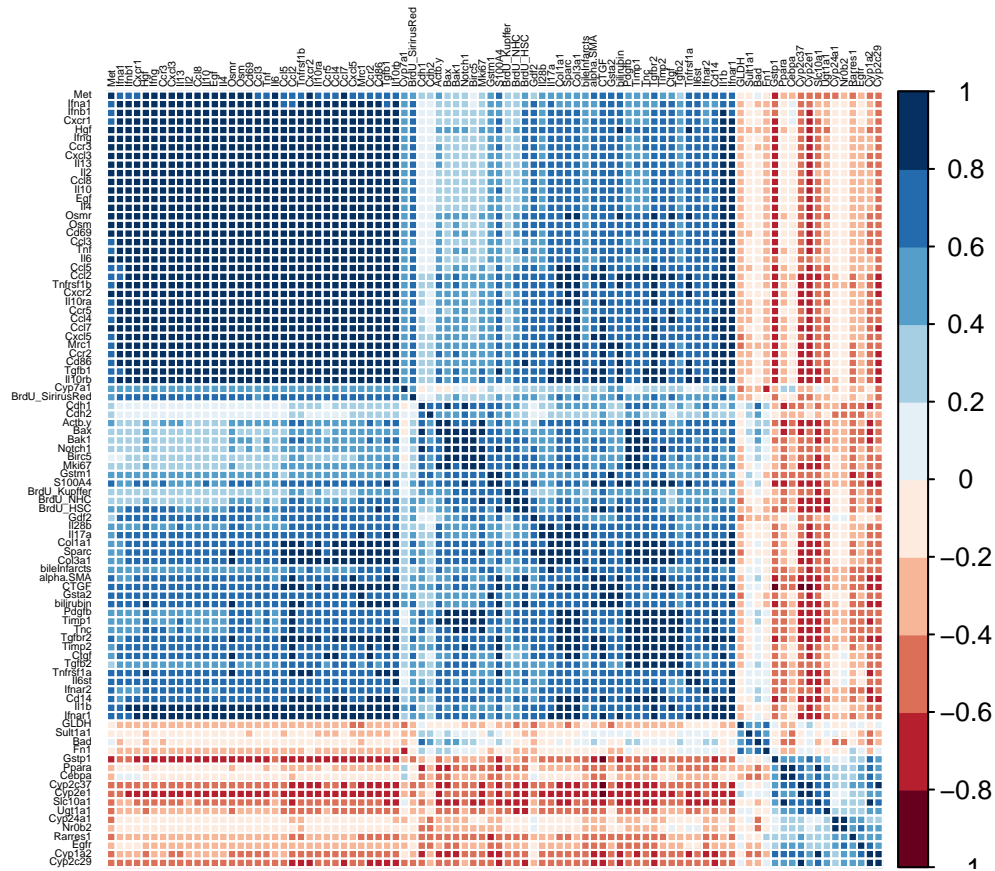
```
f_corrplot("cor.pearson", data=cor.pearson, order="hclust", folder=resultsPath)
```

Spearman correlation with hierarchical clustering

```
f_corrplot("cor.spearman", data=cor.spearman, order="hclust", folder=NULL)
```



```
# Pearson correlation with hierarchical clustering
f_corrplot("cor.pearson", data=cor.pearson, order="hclust", folder=NULL)
```



YS and YR correlation

Calculation of time-course based correlation measurements, namely ys1, ys2, ys3, yr1, yr2, yr3 In ys1, ys2, yr1, yr2 all calculations are performed on the mean time course data. The classical correlation components are replaced with the correlations calculated on the individual data points.

```
# calculate ys1, yr1 on mean data, i.e. correlation part (S*), slope part (A) and min/max part (M) are
w <- list(w1=0.5, w2=0.25, w3=0.25)
ys1.mean <- ys1.df(BDLmean.fil, BDLmean.time, w1=w$w1, w2=w$w2, w3=w$w3, use="pairwise.complete.obs")
ys2.mean <- ys2.df(BDLmean.fil, BDLmean.time, w1=w$w1, w2=w$w2, w3=w$w3, use="pairwise.complete.obs")

# scaling to interval [-1, 1]
# The dimensions correspond to the filtered dataset
cor.ys1.mean <- 2*(ys1.mean$value - 0.5)
cor.ys2.mean <- 2*(ys2.mean$value - 0.5)

# Pearson & spearman correlation on full dataset as replacement for the
# mean Pearson/Spearman in ys1, ys2, yr1, yr2
# Now calculate the scores with full correlation
cor.S_star <- ( cor(BDLmean.fil, method="spearman", use="pairwise.complete.obs") + 1 )/2
```



```

cor.R_star <- ( cor(BDLmean.fil, method="pearson", use="pairwise.complete.obs") + 1 )/2

# Calculate the ys(1,2,3) and yr(1,2,3) with single time point correlation for
# Spearman, respectively Pearson (instead of mean)
# Takes the individual correlation, slope and min/max components for the respective
# score parts and weights with the provided weighting factors (w1,w2,w3)
cor.y1.raw <- w$w1*cor.S_star + w$w2*ys1.mean$A + w$w3*ys1.mean$M
cor.y2.raw <- w$w1*cor.S_star + w$w2*ys2.mean$A_star + w$w3*ys2.mean$M_star

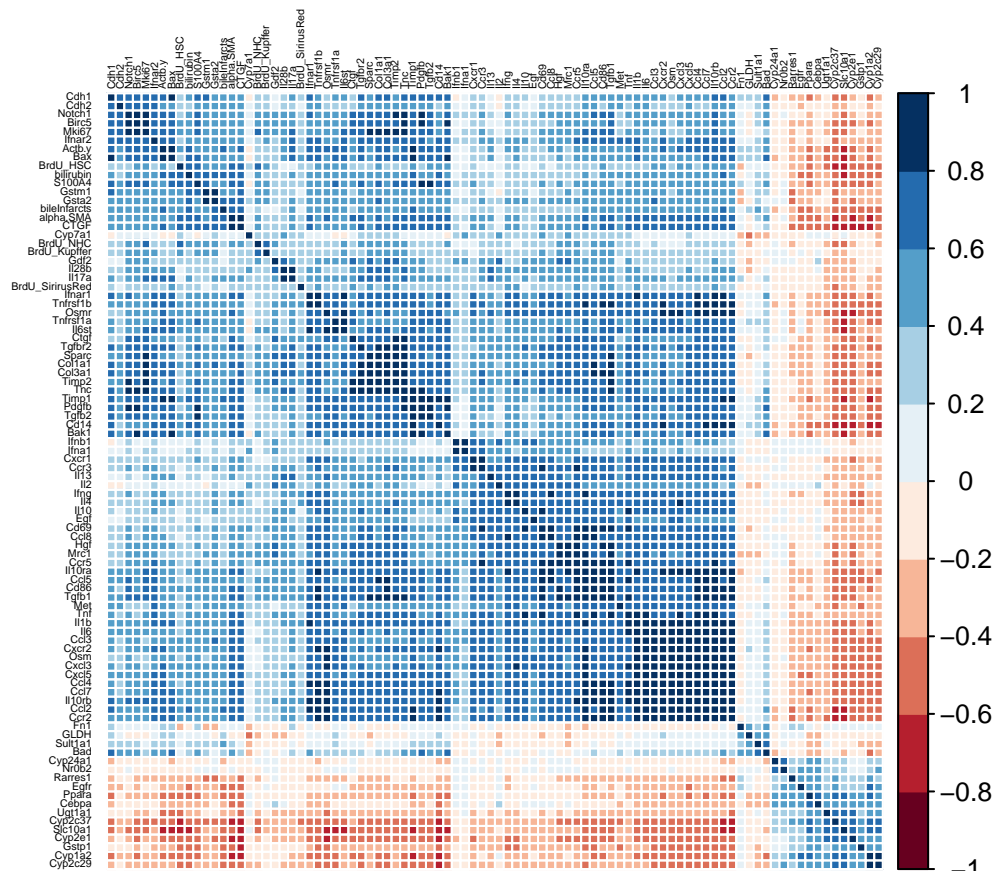
# considering slope and time difference
cor.y3.raw <- w$w1*cor.S_star + w$w2*ys2.mean$A_star2 + w$w3*ys2.mean$M_star2

cor.yr1.raw <- w$w1*cor.R_star + w$w2*ys1.mean$A + w$w3*ys1.mean$M
cor.yr2.raw <- w$w1*cor.R_star + w$w2*ys2.mean$A_star + w$w3*ys2.mean$M_star
# considering slope and time difference
cor.yr3.raw <- w$w1*cor.R_star + w$w2*ys2.mean$A_star2 + w$w3*ys2.mean$M_star2

# scaling of correlation coefficient in interval [-1,1]
cor.y1 <- 2*(cor.y1.raw-0.5)
cor.y2 <- 2*(cor.y2.raw-0.5)
cor.y3 <- 2*(cor.y3.raw-0.5)
cor.yr1 <- 2*(cor.yr1.raw-0.5)
cor.yr2 <- 2*(cor.yr2.raw-0.5)
cor.yr3 <- 2*(cor.yr3.raw-0.5)

# plot subset for further analysis
f_corrplot("cor.y2", data=cor.y2, order="hclust", folder=NULL)

```



```
# and create all files on disk
f_corrplot("cor.ys1", data=cor.ys1, order="hclust", folder=resultsPath)
f_corrplot("cor.ys2", data=cor.ys2, order="hclust", folder=resultsPath)
f_corrplot("cor.ys3", data=cor.ys3, order="hclust", folder=resultsPath)
f_corrplot("cor.yr1", data=cor.yr1, order="hclust", folder=resultsPath)
f_corrplot("cor.yr2", data=cor.yr2, order="hclust", folder=resultsPath)
f_corrplot("cor.yr3", data=cor.yr3, order="hclust", folder=resultsPath)

# extreme example where the difference between pearson and spearman matters
# plot_cor_pair("Nos2", "Cxcl15")
# plot_cor_pair("albumin", "Cyp2b10")
```

Hierarchical clustering

The next step of dimension reduction is clustering of the correlation matrix. This groups the factors into sets with the correlation within sets being larger than between sets. This effectively finds groups of factors which have similar time courses.

The `hclust` function in R was used for clustering with complete linkage method for hierarchical clustering. This particular clustering method defines the cluster distance between two clusters to be the maximum distance between their individual components.

An overview of the clusters is given in the `results/cluster` folder.

```
# apply hierarchical clustering based on selected correlation measure
```

```
method = "ys2"  
if (identical(method, "ys1")){  
  cor.cluster <- cor.ys1  
}else if (identical(method, "ys2")){  
  cor.cluster <- cor.ys2  
}else if (identical(method, "ys3")){  
  cor.cluster <- cor.ys3  
}else if (identical(method, "yr1")){  
  cor.cluster <- cor.yr1  
}else if (identical(method, "yr2")){  
  cor.cluster <- cor.yr2  
}else if (identical(method, "yr3")){  
  cor.cluster <- cor.yr3  
}
```

```
# perform hierarchical clustering and cut into Ngroups clusters.
```

```
hc <- hclust(dist(cor.cluster))  
Ngroups <- 5  
groups <- cutree(hc, k=Ngroups)  
groups.hc.order <- groups[hc$order]
```

```
# Plot the clusters
```

```
require('matrixStats')
```

```
## Loading required package: matrixStats
```

```
## matrixStats v0.14.2 (2015-06-23) successfully loaded. See ?matrixStats for help.
```

```
##
```

```
## Attaching package: 'matrixStats'
```

```
##
```

```
## The following objects are masked from 'package:Biobase':
```

```
##
```

```
##      anyMissing, rowMedians
```

```
# mean plots for clusters
```

```
f_normalize_centering <- function(a){  
  a.norm <- (a - mean(a))/(max(a, na.rm=TRUE) - min(a, na.rm=TRUE))  
  return(a.norm)  
}
```

```
# plot of mean clusters
```

```
plot_clusters <- function(folder=NULL){  
  if (!is.null(folder)){  
    fname <- sprintf("%s_cluster_overview.png", method)  
    png(filename=sprintf("../results/cluster/%s", fname), width=1600, height=1600, res=200)  
  }  
  # par(mfrow=c(ceiling(sqrt(Ngroups)),ceiling(sqrt(Ngroups))))  
  par(mfrow=c(2,3))
```

```
steps <- 1:8
```

```
for (k in 1:Ngroups){  
  g <- groups.hc.order[groups.hc.order==k]  
  N <- ceiling(sqrt(length(g)))
```

```

dgroup <- BDLmean[names(g)]

# centralize and normalize columns, i.e. the individual factors for comparison
dgroup.norm <- apply(dgroup, 2, f_normalize_centering)

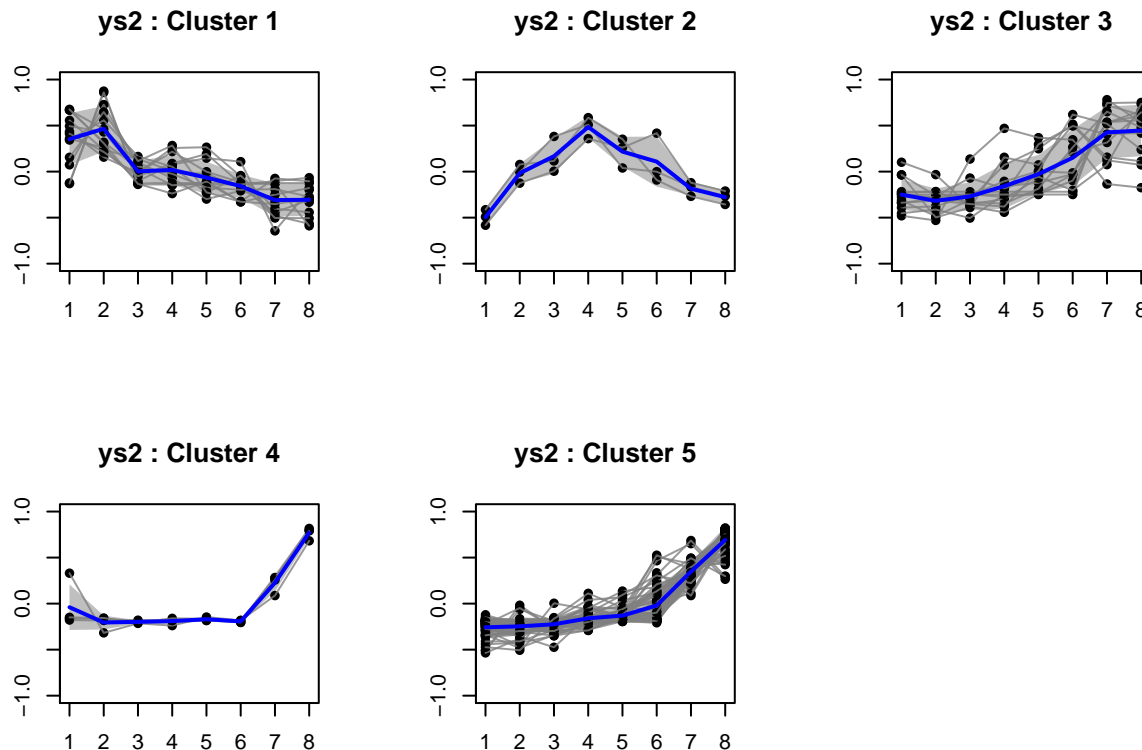
# mean and sd for timepoints
g.mean <- rowMeans(dgroup.norm)
g.sd <- rowSds(dgroup.norm) # apply(dgroup.norm, 2, sd)

# plot sd range
plot(1, type="n", xlab="", ylab="", xlim=c(1, 8), ylim=c(-1, 1), main=sprintf("%s : Cluster %s", name, cluster),
     polygon(c(steps, rev(steps)), c(g.mean+g.sd, rev(g.mean-g.sd)),
            col = rgb(0.5,0.5,0.5,0.5), border = NA)

# individual data
for (name in names(g)){
  points(steps, dgroup.norm[, name], pch=16, col="black")
  lines(steps, dgroup.norm[, name], col=rgb(0.5,0.5,0.5, 0.8), lwd=1)
}
# mean over factors in cluster
lines(steps, g.mean, col="blue", lwd=2)
}
par(mfrow=c(1,1))
if (!is.null(folder)){
  invisible(dev.off())
}
}

plot_clusters(folder=NULL)

```



Overview of the individual time courses in the clusters

```
# Plot individual time courses in cluster
plot_clusters_items <- function(folder=NULL){
  for (k in 1:Ngroups){
    if (!is.null(folder)){
      fname <- sprintf("%s_cluster_%s.png", method, k)
      png(filename=sprintf("../results/cluster/%s", fname), width=3000, height=3000, res=200)
    }
    g <- groups.hc.order[groups.hc.order==k]
    N <- ceiling(sqrt(length(g)))
    par(mfrow=c(N,N))
    for (name in names(g)){
      plot_single(name_A=name)
    }
    par(mfrow=c(1,1))
    if (!is.null(folder)){
      invisible(dev.off())
    }
  }
}

plot_clusters_items(folder=NULL)
```