A bendamustine resistance gene signature in Diffuse Large B-cell Lymphoma and Multiple Myeloma: Supplementary text 1

Issa Ismail Issa, Rasmus Froberg Brøndum,
Hanne Due, Linnéa Schmidt, Martin Bøgsted & Karen Dybkær

31/08/2020

Contents

DoseR

Load DBF files and process with DoseR (Falgreen et al. 2014).

```
A.data.file <- file.path("../GeneratedData/bendamustine_doseR.RData")

if(!file.exists(A.data.file)){
  set.seed(10000)

  A.data <- createMetaData(dbf.path = "../ExternalData/DBF files/",
                           protocol.path  = "../ExternalData/Protocols/",
```

```
                          correctionname = "Correction",
                          date.format = "%d%m%y",
                          additional.metadata = "../ExternalData/Cell_line_data.xlsx",
                          data.file = "../GeneratedData/Bendamustine_DoseR_meta")

A.data <- readDBFData(A.data = A.data,
                      discard.lines = c(1, 2),
                      dosevar       = "Concentration",
                      additivevar   = "Additive",
                      controlval    = "Control",
                      backgroundval = "Background",
                      mistakeval    = "X",
                      progressbar   = "window",
                      update = TRUE)

A.data <- bgModel(A.data = A.data,
                  weights = "fitted",
                  fitted.a = FALSE,
                  outlier.test = 3,
                  outlier.iter = 2,
                  parametrisation = "unrestricted",
                  progressbar = "window",
                  update = TRUE)


A.data <- bootstrap(A.data,
                    n.samples = 50,
                    type = "parametric",
                    verbose = FALSE)

A.data <- doseResponseModel(A.data = A.data,
                            models = c("G","R", "D"),
                            t = 48,
                            AUC.q = 0,
                            dose.scale = "ug/ml",
                            dose.logfun.to = "log10",
                            parametrisation = "restricted",
                            progressbar = "window")

## Plots of background corrected data
plotbgModel(A.data,
           pdfit = TRUE,
           col = rev(colours[1:4]),
           pointsize = 7,
           figure.output = "../output/Background")

plot.growthModel(A.data,
                 time.points.used = "all" ,
                 bootstrap.conf = TRUE,
                 line.col = colours[2],
                 pointsize = 8,
                 pdfit = TRUE,
                 figure.output = "../output/GrowthModel",
```

```
                    plotgrid = TRUE, #nrows=4, ncols=5,
                    col.by.identifier = FALSE)

  save(A.data, file = A.data.file)
} else{
  load(A.data.file)
}
```

Print table with summary statistics

```
kable(tableCI(A.data,
      dose.scale = "ug/ml"),
      row.names = F,
      col.names = c("Cellline", "T0","GI50", "TGI", "LC48", "AUC0"))
```

Cellline	T0	GI50	TGI	LC48	AUC0
AMO-1	30 (29;31)	1.84 (1.83;1.84)	1.91 (1.91;1.92)	2.01 (2.00;2.02)	350.77 (344.73;357.66)
DB	40 (39;41)	1.83 (1.83;1.84)	1.96 (1.95;1.97)	2.10 (2.10;2.10)	344.57 (338.43;347.85)
FARAGE_M	48 (46;50)	0.99 (0.96;1.01)	1.37 (1.34;1.39)	1.58 (1.57;1.59)	269.60 (258.59;273.48)
HBL-1_M	32 (31;32)	1.46 (1.44;1.47)	1.63 (1.62;1.64)	1.81 (1.80;1.82)	313.61 (308.36;316.95)
KMS-11	47 (45;49)	1.82 (1.81;1.82)	1.85 (1.85;1.85)	1.91 (1.91;1.92)	340.53 (330.90;348.95)
KMS-12-BM	38 (37;40)	1.30 (1.28;1.33)	1.53 (1.52;1.53)	1.62 (1.61;1.63)	287.42 (279.99;292.87)
KMS-12-PE	45 (43;47)	1.80 (1.80;1.80)	1.81 (1.81;1.82)	1.83 (1.82;1.84)	346.87 (336.46;354.13)
LP-1	37 (35;38)	2.07 (2.03;2.10)	2.12 (2.10;2.13)	2.16 (2.12;2.17)	351.59 (334.09;361.07)
MC-116	59 (55;62)	1.56 (1.52;1.57)	1.75 (1.71;1.78)	1.81 (1.81;1.81)	288.97 (274.71;294.63)
MOLP-2	129 (119;141)	0.99 (0.94;1.01)	1.20 (1.16;1.22)	1.80 (1.77;1.80)	250.41 (237.85;261.82)
MOLP-8	32 (31;33)	1.27 (1.25;1.28)	1.50 (1.47;1.52)	1.75 (1.72;1.78)	297.54 (293.42;302.04)
NCI-H929	43 (42;45)	1.80 (1.80;1.80)	1.83 (1.80;1.83)	1.88 (1.81;1.89)	340.65 (326.74;346.13)
NU-DHL-1	34 (33;35)	0.96 (0.93;0.99)	1.28 (1.27;1.29)	1.52 (1.51;1.52)	267.75 (261.68;269.21)
NU-DUL-1	43 (42;44)	0.60 (0.60;0.61)	0.73 (0.72;0.74)	0.96 (0.95;0.97)	228.13 (226.01;235.19)
OCI-Ly3_M	26 (25;26)	1.35 (1.33;1.36)	1.61 (1.57;1.64)	1.88 (1.87;1.89)	310.70 (304.25;314.61)
OCI-Ly7_M	44 (42;46)	1.54 (1.53;1.55)	1.69 (1.67;1.71)	1.83 (1.82;1.84)	269.17 (262.78;278.12)
OCI-Ly8	23 (23;24)	1.52 (1.49;1.54)	1.83 (1.82;1.83)	1.92 (1.91;1.93)	315.94 (309.48;320.47)
OPM-2	69 (66;73)	1.82 (1.81;1.83)	1.88 (1.87;1.89)	2.06 (2.05;2.07)	342.42 (329.41;350.07)

Cellline	T0	GI50	TGI	LC48	AUC0
RIVA_M	40 (38;41)	1.34 (1.32;1.38)	1.56 (1.56;1.57)	1.73 (1.72;1.74)	286.65 (275.47;294.00)
RPMI-8226	43 (41;46)	2.10 (2.10;2.11)	2.10 (2.10;2.14)	2.11 (2.11;2.21)	378.98 (369.63;382.48)
SU-DHL-10_M	31 (30;32)	1.83 (1.83;1.83)	1.87 (1.87;1.88)	1.93 (1.92;1.93)	351.16 (343.86;357.56)
SU-DHL-5	35 (34;37)	0.44 (0.43;0.45)	0.62 (0.62;0.63)	0.78 (0.78;0.79)	219.79 (213.34;223.48)
SU-DHL-8_M	58 (56;61)	1.05 (1.03;1.07)	1.24 (1.22;1.26)	1.54 (1.52;1.56)	281.88 (269.61;283.06)
U266	58 (56;60)	1.80 (1.75;1.80)	1.88 (1.88;1.89)	2.09 (2.08;2.10)	333.48 (326.50;337.13)
U2932_M	46 (43;47)	1.16 (1.13;1.19)	1.37 (1.35;1.39)	1.68 (1.67;1.70)	288.84 (280.70;293.88)

Dose response figures

Get plotting order

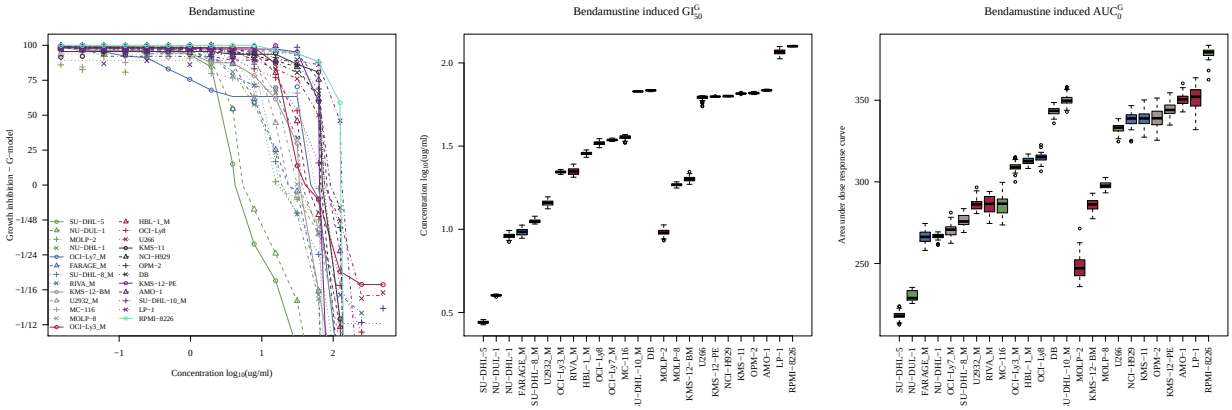
```
plotOrder <- unique(A.data$data$iso.fits$Bendamustine$G[,c("Cellline", "Disease")])
plotOrder$meanGI50 <- colMedians(A.data$summary$Bendamustine$G$GI50)
plotOrder$meanAUC  <- colMedians(A.data$summary$Bendamustine$G$AUC)
plotOrderGI50      <- plotOrder$Cellline[order(plotOrder$Disease, plotOrder$meanGI50)]
plotOrderAUC       <- plotOrder$Cellline[order(plotOrder$Disease, plotOrder$meanAUC)]
```

```
par(mfrow = c(1,3), oma = c(1.2,0,0,0))

plot.DRdata(A.data,
            model      = "G",
            col.scheme =  colours,
            n.columns  = 2,
            plot.data  = TRUE,
            legend     = TRUE,
            font.main  = 1,
            legend.cex = 0.8,
            dose.scale = "ug/ml",
            ylim = c(-1/12,100))
```

```
## [1] "all"
```

```
DRdataBoxplot(A.data,
              model       = "G",
              type        = "GI50",
              dose.scale  = "ug/ml",
              dose.logfun = "log10",
              col.all = rep(colours[1:4], each = 2),
              plot.order = plotOrderGI50)

DRdataBoxplot(A.data,
              model       = "G",
```

```
                type        = "AUC",
                dose.scale  = "ug/ml",
                dose.logfun = "log10",
                col.all = rep(colours[1:4], each = 2),
                plot.order = plotOrderAUC)
```




```
## [1] "all"
```

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

Correlation Analysis

Load old AUC data from original REGS (Falgreen et al. 2015)

```
if(!file.exists("../GeneratedData/auc_old.RData")){
  load(paste0("K:/FORSK-Projekt/Projekter/Scientific Projects/007_PhD_Steffen_2010/",
              "PhD Study/Analysis/Study III - REGS/ExternalData/A.data.Rdata"))

  b <- list()
  b[["cyclophosfamide"]] <- old$summary$Cyclophosphamide$G$AUC["BC",]
  b[["doxorubicin"]] <- old$summary$Doxorubicin$G$AUC["BC",]
  b[["vincristine"]] <- old$summary$Vincristine$G$AUC["BC",]
  b[["melphalan"]]   <- old$summary$Melphalan$G$AUC["BC",]

  AUC.old <- array(NA, dim=c(length(unique(unlist(sapply(b,names)))),
                             length(b)))
  row.names(AUC.old) <- unique(unlist(sapply(b,names)))
  AUC.old[names(b[["cyclophosfamide"]]), 1] <- b[["cyclophosfamide"]]
  AUC.old[names(b[["doxorubicin"]]), 2] <- b[["doxorubicin"]]
  AUC.old[names(b[["vincristine"]]), 3] <- b[["vincristine"]]
  AUC.old[names(b[["melphalan"]]), 4] <- b[["melphalan"]]

  row.names(AUC.old) <- recode(row.names(AUC.old),
                               "U-266" = "U266",
                               "FARAGE" = "FARAGE_M",
                               "KMS-12" = "KMS-12_M",
```

```
                                "OCI-Ly3" = "OCI-Ly3_M",
                                "OCI-Ly7" = "OCI-Ly7_M",
                                "RIVA" = "RIVA_M",
                                "SU-DHL-10" = "SU-DHL-10_M",
                                "SU-DHL-8" = "SU-DHL-8_M",
                                "U2932" = "U2932_M")

  colnames(AUC.old) <- c("cyclophosfamide", "doxorubicin", "vincristine", "melphalan")
  save(AUC.old, file = "../GeneratedData/auc_old.RData")
} else{
  load("../GeneratedData/auc_old.RData")
}
```

Extract AUC from current study and merge

```
bendamustine_AUC <- data.frame("bendamustine" = A.data$summary[[1]]$G$AUC["BC", ])
AUC_combined <- merge(bendamustine_AUC,AUC.old, by = 0, all.x = T, all.y = T)
names(AUC_combined)[3] <- "cyclophosphamide"
```

Add cellline meta data

```
cellline_meta <- read_xlsx("../ExternalData/Cell_line_data.xlsx")
disease <- cellline_meta[,c("Cellline","Disease")]
AUC_combined <- merge(AUC_combined, disease, by.x = "Row.names", by.y = "Cellline")
row.names(AUC_combined) <- AUC_combined$Row.names
AUC_combined$Disease <- factor(AUC_combined$Disease, levels = c("MM", "DLBCL"))
colnames(AUC_combined) <- str_to_title(colnames(AUC_combined))
```

List AUC data

```
kable(AUC_combined[,-1])
```

	Bendamustine	Cyclophosphamide	Doxorubicin	Vincristine	Melphalan	Disease
AMO-1	350.7658	394.5053	269.2963	115.95004	377.0481	MM
DB	344.5708	345.6162	275.5403	130.75344	320.5391	DLBCL
FARAGE_M	269.6047	311.0123	179.5583	56.11407	NA	DLBCL
HBL-1_M	313.6084	NA	NA	NA	NA	DLBCL
KMM-1	NA	295.2193	345.9903	132.58739	319.5699	MM
KMS-11	340.5267	317.9442	355.9492	122.95652	351.5188	MM
KMS-12-BM	287.4202	327.5494	330.5789	112.35735	387.2146	MM
KMS-12-PE	346.8659	308.7462	340.0362	121.19976	398.6227	MM
LP-1	351.5947	308.2082	315.5337	186.70591	384.6651	MM
MC-116	288.9678	NA	271.7750	61.97078	NA	DLBCL
MM1S	NA	241.8550	227.4044	89.68633	313.4711	MM
MOLP-2	250.4064	NA	NA	NA	NA	MM
MOLP-8	297.5354	248.5153	252.5796	149.29363	301.8613	MM
NCI-H929	340.6480	321.8616	291.2106	149.46391	319.4915	MM
NU-DHL-1	267.7498	171.7444	200.6328	NA	274.5045	DLBCL
NU-DUL-1	228.1279	214.1146	223.8800	90.25121	NA	DLBCL
OCI-Ly19	NA	202.1319	166.8577	53.99711	NA	DLBCL
OCI-Ly3_M	310.6959	261.6746	253.3673	74.82913	NA	DLBCL

	Bendamustine	Cyclophosphamide	Doxorubicin	Vincristine	Melphalan	Disease
OCI-Ly7_M	269.1699	257.8940	324.8760	114.48730	362.6347	DLBCL
OCI-Ly8	315.9425	NA	NA	NA	NA	DLBCL
OPM-2	342.4212	302.8723	329.4768	95.03467	363.3948	MM
RIVA_M	286.6516	295.9854	326.6613	108.99317	NA	DLBCL
RPMI-8226	378.9778	349.6584	297.0493	110.73944	415.6642	MM
RPMI-8226 LR5	NA	NA	NA	NA	429.5230	MM
SU-DHL-10_M	351.1579	NA	NA	NA	NA	DLBCL
SU-DHL-4	NA	NA	288.7893	NA	NA	DLBCL
SU-DHL-5	219.7875	164.5473	201.5278	57.86447	NA	DLBCL
SU-DHL-8_M	281.8773	270.6052	222.0442	126.06404	NA	DLBCL
U266	333.4787	375.8063	325.3712	90.25150	NA	MM
U2932_M	288.8447	329.8053	294.7770	85.33884	NA	DLBCL

Remove DLBCL for melphalan before investigating cross-resistance

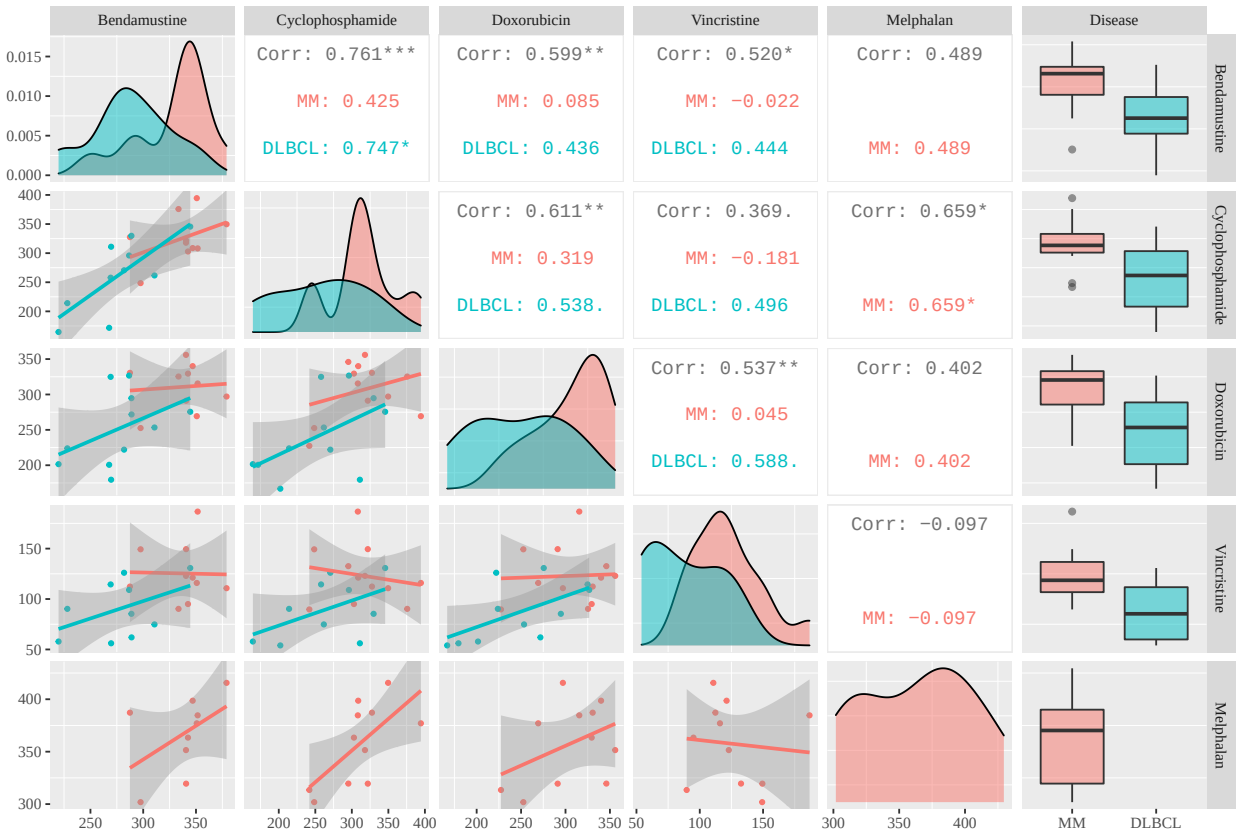
```
AUC_combined$Melphalan[AUC_combined$Disease == "DLBCL"] <- NA
```

Cross Resistance plot

```
ggpairs(subset(AUC_combined, disease!="ALL")[,-1], aes(colour = Disease))
```

```
full_plot <- ggpairs(AUC_combined[,-1],
                title = "Correlation of drug response",
                mapping = ggplot2::aes(colour=Disease),
                lower = list(continuous = wrap("smooth", alpha = 1, size=1),
                             discrete = "blank", combo="blank"),
                diag = list(discrete="barDiag",
                            continuous = wrap("densityDiag", alpha=0.5 )),
                upper = list(combo = wrap("box_no_facet", alpha=0.5),
                             continuous = wrap("cor", size=4))) +
                theme(panel.grid.major = element_blank())    # remove gridlines

plots = list()
for (i in 1:5){
  plots <- c(plots, lapply(1:full_plot$ncol, function(j) getPlot(full_plot, i = i, j = j)))
}
ggmatrix(plots,
         nrow = 5,
         byrow = T,
         ncol=full_plot$ncol,
         xAxisLabels = full_plot$xAxisLabels,
         yAxisLabels = full_plot$yAxisLabels[1:5],
         title="Correlation of drug response")
```

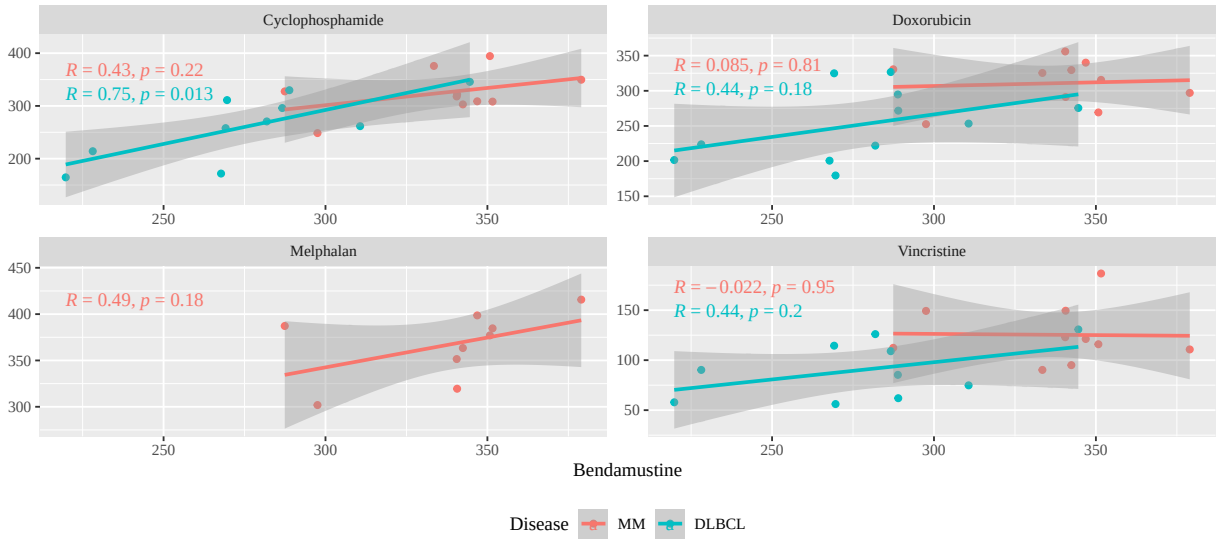
Correlation of drug response




```
cor(AUC_combined[,2:6], use = "pairwise.complete.obs")
```

```
##                 Bendamustine Cyclophosphamide Doxorubicin Vincristine   Melphalan
## Bendamustine       1.0000000        0.7614691   0.5986615  0.52027036  0.48941982
## Cyclophosphamide   0.7614691        1.0000000   0.6114696  0.36855480  0.65861475
## Doxorubicin        0.5986615        0.6114696   1.0000000  0.53651581  0.40225149
## Vincristine        0.5202704        0.3685548   0.5365158  1.00000000 -0.09695655
## Melphalan          0.4894198        0.6586148   0.4022515 -0.09695655  1.00000000
```

```
plotData <- pivot_longer(AUC_combined, cols = c("Cyclophosphamide", "Doxorubicin",
                                                "Vincristine","Melphalan"))
ggplot(plotData, aes(x = Bendamustine, y = value, col = Disease)) +
         geom_point() + theme(legend.position = "bottom") +
         geom_smooth(method='lm', formula= y~x) +
         stat_cor() + ylab("") +
  facet_wrap(~name, ncol = 2, scales = "free")
```



Classifier training

Load cellline data Gene expression data. Data has been normalized using fRMA.

```
load("../ExternalData/cl.gep.file.Rdata")
cellline_meta <- read_xlsx("../ExternalData/Cell_line_data.xlsx")
rownames(cellline_meta) <- cellline_meta$Cellline
```

```
## Warning: Setting row names on a tibble is deprecated.
```

BAGS Classify cell line data

```
cl.gep.sc <- microarrayScale(cl.gep)
cellline_meta2 <- cellline_meta[colnames(cl.gep.sc),]

cl.gep.sc.mm <- cl.gep.sc[, cellline_meta2$Cellline[cellline_meta2$Disease == "MM"]]
row.names(cl.gep.sc.mm) <- sub("_at", "", row.names(cl.gep.sc.mm))

load(paste0("K:/FORSK-Projekt/Projekter/Scientific Projects/053_BAGS_MM/GeneratedData/",
            "BAGSClassifierTools.RData"))
BAGSclassifier(new.data = cl.gep.sc.mm, BAGS.coef = BAGS.coef, percent.classified = 100)
```

```
## $class
##      AMO-1     KMS-11  KMS-12-BM  KMS-12-PE       LP-1     MOLP-2     MOLP-8   NCI-H929      OPM-2
## Plasmacell Plasmacell Plasmacell     Pre-BI Plasmacell Plasmacell Plasmacell Plasmacell Plasmacell
##  RPMI-8226       U266
## Plasmacell Plasmacell
## Levels: Pre-BI Pre-BII Immature Naive Memory Plasmacell Unclassified
##
## $prob.mat
##                Pre-BI      Pre-BII     Immature        Naive      Memory Plasmacell
## AMO-1     0.015452136 0.0797462709 0.0790573711 0.020693708 0.260914845  0.5441357
```

```
## KMS-11    0.107458980 0.0693306811 0.0352418000 0.085978425 0.078093149  0.6238970
## KMS-12-BM 0.067540222 0.0146864746 0.0121790983 0.047855200 0.172128047  0.6856110
## KMS-12-PE 0.466776967 0.1234398548 0.0465249891 0.031733709 0.045299633  0.2862248
## LP-1      0.031459023 0.0105495479 0.0030491836 0.043386791 0.068901602  0.8426539
## MOLP-2    0.002936654 0.0003983842 0.0191860838 0.184368722 0.032661969  0.7604482
## MOLP-8    0.010410688 0.0028270508 0.0020155486 0.001661190 0.001550252  0.9815353
## NCI-H929  0.001339503 0.0007862209 0.0005294767 0.004138004 0.007821035  0.9853858
## OPM-2     0.064779894 0.0261184744 0.0041104385 0.006266404 0.012539434  0.8861854
## RPMI-8226 0.180406914 0.0029834672 0.0121545971 0.303832900 0.120435728  0.3801864
## U266      0.024311183 0.0861225637 0.0955057071 0.148866607 0.188403117  0.4567908
##
## $prob
##     AMO-1    KMS-11 KMS-12-BM KMS-12-PE      LP-1    MOLP-2    MOLP-8  NCI-H929     OPM-2 RPMI-8226
## 0.5441357 0.6238970 0.6856110 0.4667770 0.8426539 0.7604482 0.9815353 0.9853858 0.8861854 0.3801864
##      U266
## 0.4567908
##
## $cut
##        0%
## 0.3801864
```

Functions for classifier training

```
"%w/o%" <- function(x, value) x[!x %in% value]

cutFun <- function(x) {
  cut(x,
      breaks=c(-Inf, quantile(x, c(1/3, 2/3), na.rm = TRUE), Inf),
      labels = c("Sensitive", "Intermediate","Resistant"))
}

trainClassifier <- function(drug, train.gep = cl.gep, alphas=seq(0.1, 1, by =  0.05)){
  AUC.drug <- subset(AUC_combined, select=c(drug, "Disease"),
                     is.na(AUC_combined[[drug]])==FALSE)
  int1 <- intersect(rownames(AUC.drug), colnames(train.gep))
  AUC.drug <- AUC.drug[int1,]

  if(length(unique(AUC.drug$Disease)) > 1){
    # Create AUC classes within diseases
    AUC.drug$AUC.class <- factor("Intermediate",
                                 levels= c("Sensitive", "Intermediate", "Resistant"))
    AUC.drug$AUC.class[AUC.drug$Disease == "DLBCL"] <-
      cutFun(AUC.drug[[drug]][AUC.drug$Disease == "DLBCL"])
    AUC.drug$AUC.class[AUC.drug$Disease == "MM"] <-
      cutFun(AUC.drug[[drug]][AUC.drug$Disease == "MM"])
  } else{
    AUC.drug$AUC.class <- cutFun(AUC.drug[[drug]])
  }

  AUC.drug$AUC.class <- as.character(AUC.drug$AUC.class)
  AUC.drug <- subset(AUC.drug, AUC.class %in% c("Resistant","Sensitive"))
  y <- as.factor(AUC.drug$AUC.class)

  # The GEP data is aligned with the dose response
```

```
int2 <- intersect(rownames(AUC.drug), colnames(train.gep))
train.set <- microarrayScale(train.gep[,int2], center = "median", scale = FALSE)
train.set <- train.set[, order(colnames(train.set))]

## Matrix to hold results from CV
mat <- matrix(NaN, ncol = 4, nrow = length(alphas))
colnames(mat) <- paste(drug, c("", ".lo", ".up", ".n"), sep = "")
row.names(mat) <- alphas

## plot all the cross validation plots for later inspection
pdf(paste0("../output/",drug,"_cvplots_classification.pdf"))

class.drug.fit <- list()
for(alpha in alphas){
  cat(drug, alpha, "\n")

  fit <-
    cv.glmnet(x = t(train.set),
              alpha        = alpha,
              grouped      = FALSE,
              type.measure = "class",
              family       = "multinomial",
              nfolds       = length(y),
              keep         = TRUE,
              y            = y,
              lambda       = exp(seq(-6,3, length.out = 600)),
              standardize  = FALSE,
              nlambda      = 600)



  plot(fit, main = paste(drug, alpha))

  wh.min <- length(fit$cvm) - which.min(rev(fit$cvm))
  mat[paste(alpha), drug] <- min(fit$cvm)
  mat[paste(alpha), paste(drug, ".lo", sep = "")] <-
    fit$cvlo[wh.min]
  mat[paste(alpha), paste(drug, ".up", sep = "")] <-
    fit$cvup[wh.min]
  mat[paste(alpha), paste(drug, ".n", sep = "")] <-
    fit$nzero[wh.min]

  class.drug.fit[[paste(alpha)]] <- fit
}
dev.off()

# Find best fit
best.drug.fit <- bestFit(class.drug.fit, mat)

# Extract scaled coefficients for classifier
best.drug.coef <- bestFitCoef(best.drug.fit, train.set)

# Define output
```

```
  out <- list()
  out[[paste0(drug, ".cv.list")]]  <- class.drug.fit
  out[[paste0(drug, ".cv.matrix")]]  <- mat
  out[[paste0(drug, ".cv.best.fit")]]   <- best.drug.fit
  out[[paste0(drug, ".resistance.coef")]] <- best.drug.coef

  return(out)
}

bestFit <- function(class.drug.fit, mat){
  fit <- class.drug.fit[[names(which.min((mat[, 1])))]]
  fit$lambda.min <- rev(fit$lambda)[which.min(rev(fit$cvm))]
  return(fit)
}

bestFitCoef <- function(drugClassifier.cv, train.set){
  COEF <- as.matrix(coef(drugClassifier.cv, s = "lambda.min")$Resistant)
  resistance.coef <- COEF[COEF != 0, ]
  probes <- names(resistance.coef)[-1]
  sd.drug <- apply(train.set[probes, , drop=FALSE], 1, function(x) sd(x, na.rm = TRUE))
  resistance.class.coef <- resistance.coef * c(1, sd.drug)
}

source("cv_plots.R")
```

Combined Classifier

```
file.drug.class <-file.path("../GeneratedData/bendamustine.EN.cv.RData")
if(file.exists(file.drug.class)){
    load(file.drug.class)
}else{
  set.seed(42)
  bendamustine.EN.cv <- trainClassifier(drug = "bendamustine", train.gep = cl.gep)
  save(bendamustine.EN.cv, file = file.drug.class)
}
```

Get minimum error from CV

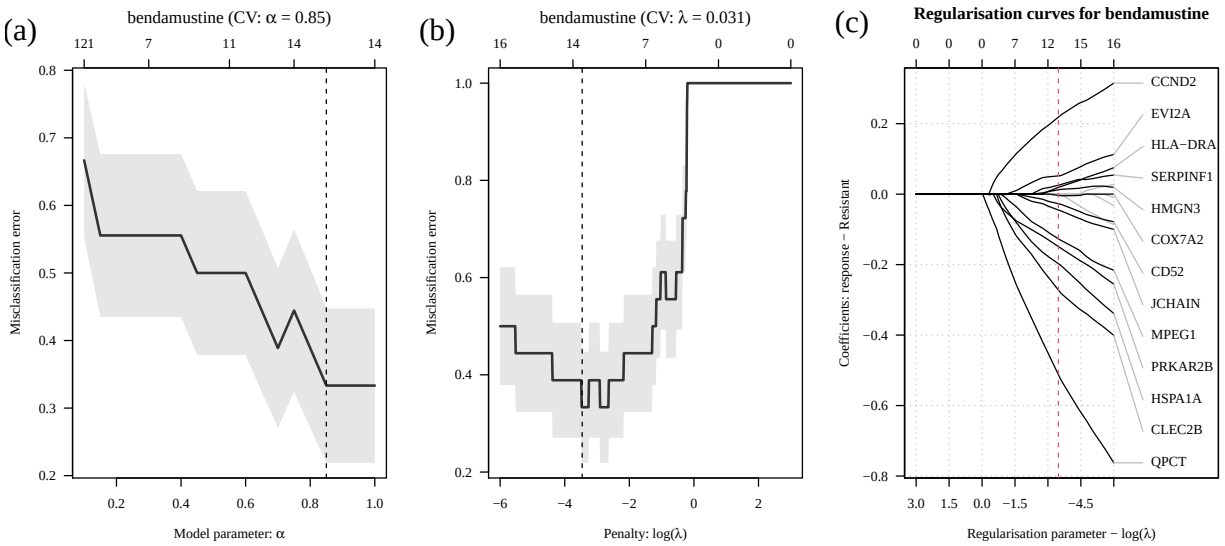
```
min(bendamustine.EN.cv$bendamustine.cv.best.fit$cvm)
```

```
## [1] 0.3333333
```

The result of the cross validation is plotted below.

```
drug <- "bendamustine"

# Extract values for plots
mat <- get(paste0(drug, ".EN.cv"))[[paste0(drug, ".cv.matrix")]]
drugClassifier.cv <- get(paste0(drug, ".EN.cv"))[[paste0(drug, ".cv.best.fit")]]
```

```r
# Plot output
par(mfrow = c(1,3))
plot.cv.alpha(x  = as.numeric(row.names(mat)),
              y  = mat[, drug],
              lo = mat[, paste(drug, ".lo", sep = "")],
              up = mat[, paste(drug, ".up", sep = "")],
              nzero = mat[,paste(drug, ".n", sep = "")],
              main1 = drug,
              las   = 1,
              ylab = "Misclassification error")

mtext("(a)", 3,
      line=1.5, adj=-0.25, cex=9/pointsize)

plot.cv.lambda(x  = log(drugClassifier.cv$lambda),
               y  = drugClassifier.cv$cvm,
               lo = drugClassifier.cv$cvlo,
               up = drugClassifier.cv$cvup,
               nzero = drugClassifier.cv$nzero,
               main1 = drug,
               las   = 1,
               ylab = "Misclassification error")

mtext("(b)", 3,
      line=1.5, adj=-0.25, cex=9/pointsize)

plot.coef(drugClassifier.cv,
          label.pct=32,
          main = paste("Regularisation curves for", drug),
          max.names = 20)

mtext("(c)", 3,
      line=2, adj=-0.25, cex=9/pointsize)
```



```
rm(mat, drugClassifier.cv)
```

Build Classifier

```
bendamustineClassifier.file <- "../GeneratedData/bendamustineClassifier.RData"

if(!file.exists(bendamustineClassifier.file)){
  ## Extract scaled coefficients from CV object
  bendamustine.resistance.class.coef <- bendamustine.EN.cv[["bendamustine.resistance.coef"]]

  # Build classifier
  bendamustineClassifier <- function(newx, coef = bendamustine.resistance.class.coef){
    ## If any probes are missing in GEP data, set to zero (remove)
    missProbe <- which(!names(coef) %in% rownames(newx))[-1]
    if(length(missProbe) > 0) coef <- coef[-missProbe]

    x <- rbind(1, newx[names(coef)[-1],,drop= FALSE])
    prob <- t(x) %*% as.matrix((coef)  )
    prob <- exp(prob) / ( exp(prob) + exp(-prob))
    colnames(prob) <- "Resistant"
    prob
  }
  save(bendamustineClassifier, bendamustine.resistance.class.coef,
       file = bendamustineClassifier.file)
} else{
  load(bendamustineClassifier.file)
}
```

Print Coefficients for classifier

Table 3: Genes included in the bendamustine classifier. Coefficients are scaled by standard deviation of genes in training data.

ensembl_gene_id	hgnc_symbol	Coefficient
ENSG00000005249	PRKAR2B	-0.393
ENSG00000110852	CLEC2B	-0.774
ENSG00000112695	COX7A2	-0.010
ENSG00000115828	QPCT	-1.560
ENSG00000118418	HMGN3	0.031
ENSG00000118971	CCND2	0.677
ENSG00000126860	EVI2A	0.118
ENSG00000132386	SERPINF1	0.056
ENSG00000132465	JCHAIN	-0.182
ENSG00000169442	CD52	-0.095
ENSG00000197629	MPEG1	-0.370
ENSG00000204287	HLA-DRA	0.069
ENSG00000204389	HSPA1A	-0.350

Classifier Validation

DLBCL

Load Validation data. GEP data has previously been cohort RMA normalized for the analysis in (Michaelsen et al. 2018).

```
## Load GEP data
gepALL      <- readRDS("../ExternalData/DLBCLdata/GEPall_ENSG.sc")
gepALL.affy <- readRDS("../ExternalData/DLBCLdata/GEPall_AFFY.sc")

## Load Metadata and subset to samples with GEP
metaALL <- readRDS("../ExternalData/DLBCLdata/metadataCombinedAll")
metaALL <- subset(metaALL, CEL %in% colnames(gepALL))

## Order GEP according to metadata
gepALL <- gepALL[, metaALL$CEL]
gepALL.affy <- gepALL.affy[, metaALL$CEL]
```

Classify samples with bendamustine REGS and BAGS (Dybkaer et al. 2015)

```
metaALL$Bendamustine.DLBCL.res.prob <- bendamustineClassifier(exprs(gepALL))
metaALL$Bendamustine.DLBCL.res.class <- cutFun(metaALL$Bendamustine.DLBCL.res.prob)

metaALL$BAGS <- BAGS(exprs(gepALL.affy))$class
```

Tabulate data

```
tablePct <- function(x, y){
  count <- as.matrix(table(x,y))
  count <- addmargins(count)
  pct   <- count / count[, ncol(count)]#as.matrix(table(x,y)) / rowSums(as.matrix(table(x,y)))
  for(i in 1:nrow(count)){
    for(j in 1:ncol(count)){
      count[i,j] = paste0(count[i,j], " (", round(100*pct[i,j]),")")
    }
  }
  return(count)
}

BagsTable <- tablePct(metaALL$Data, metaALL$BAGS)
ABCGCBTable  <- tablePct(metaALL$Data, metaALL$WrightClass)
BagsABCtable <- tablePct(metaALL$Data[metaALL$WrightClass == "ABC"],
                         metaALL$BAGS[metaALL$WrightClass == "ABC"])
BagsGCBtable <- tablePct(metaALL$Data[metaALL$WrightClass == "GCB"],
                         metaALL$BAGS[metaALL$WrightClass == "GCB"])

kable(BagsTable, caption = "BAGS classes by dataset")
```

Table 4: BAGS classes by dataset

	Naive	Centroblast	Centrocyte	Memory	Plasmablast	Unclassified	Sum
IDRC	12 (3)	87 (19)	186 (40)	33 (7)	70 (15)	79 (17)	467 (100)
LLMPPCHOP	10 (6)	39 (22)	59 (33)	24 (13)	20 (11)	29 (16)	181 (100)
LLMPPRCHOP	16 (7)	43 (18)	90 (39)	23 (10)	34 (15)	27 (12)	233 (100)
MDFCI	9 (10)	20 (22)	32 (36)	6 (7)	12 (13)	11 (12)	90 (100)
Sum	47 (5)	189 (19)	367 (38)	86 (9)	136 (14)	146 (15)	971 (100)

```
kable(ABCGCBTable, caption = "ABC / GCB classes by dataset")
```

Table 5: ABC / GCB classes by dataset

	ABC	GCB	Unclassified	Sum
IDRC	198 (42)	225 (48)	44 (9)	467 (100)
LLMPPCHOP	74 (41)	76 (42)	31 (17)	181 (100)
LLMPPRCHOP	93 (40)	107 (46)	33 (14)	233 (100)
MDFCI	42 (48)	30 (34)	15 (17)	87 (100)
Sum	407 (42)	438 (45)	123 (13)	968 (100)

```
kable(BagsABCtable, caption = "BAGS(ABC) classes by dataset")
```

Table 6: BAGS(ABC) classes by dataset

	Naive	Centroblast	Centrocyte	Memory	Plasmablast	Unclassified	Sum
IDRC	7 (4)	20 (10)	51 (26)	26 (13)	44 (22)	50 (25)	198 (100)
LLMPPCHOP	7 (9)	11 (15)	16 (22)	15 (20)	10 (14)	15 (20)	74 (100)
LLMPPRCHOP	6 (6)	10 (11)	25 (27)	19 (20)	15 (16)	18 (19)	93 (100)
MDFCI	6 (14)	3 (7)	9 (21)	5 (12)	10 (24)	9 (21)	42 (100)
Sum	26 (6)	44 (11)	101 (25)	65 (16)	79 (19)	92 (23)	407 (100)

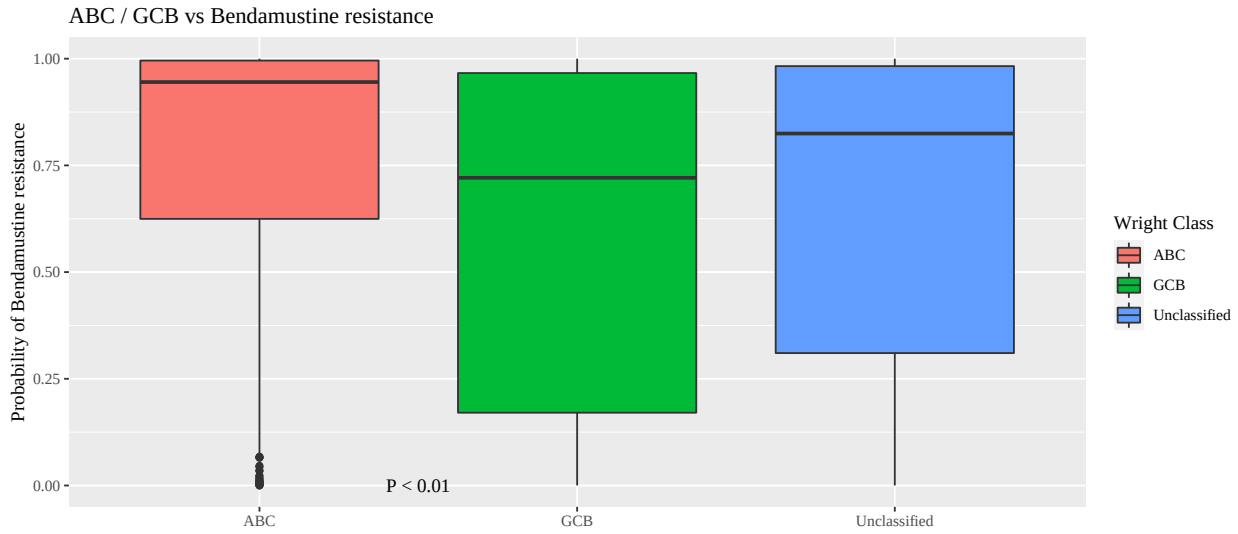
```
kable(BagsGCBtable, caption = "BAGS(GCB) classes by dataset")
```

Table 7: BAGS(GCB) classes by dataset

	Naive	Centroblast	Centrocyte	Memory	Plasmablast	Unclassified	Sum
IDRC	5 (2)	60 (27)	124 (55)	3 (1)	14 (6)	19 (8)	225 (100)
LLMPPCHOP	3 (4)	26 (34)	33 (43)	3 (4)	3 (4)	8 (11)	76 (100)
LLMPPRCHOP	5 (5)	27 (25)	58 (54)	2 (2)	10 (9)	5 (5)	107 (100)
MDFCI	1 (3)	13 (43)	14 (47)	0 (0)	1 (3)	1 (3)	30 (100)
Sum	14 (3)	126 (29)	229 (52)	8 (2)	28 (6)	33 (8)	438 (100)

ABC /GCB vs Bendamustine resistance

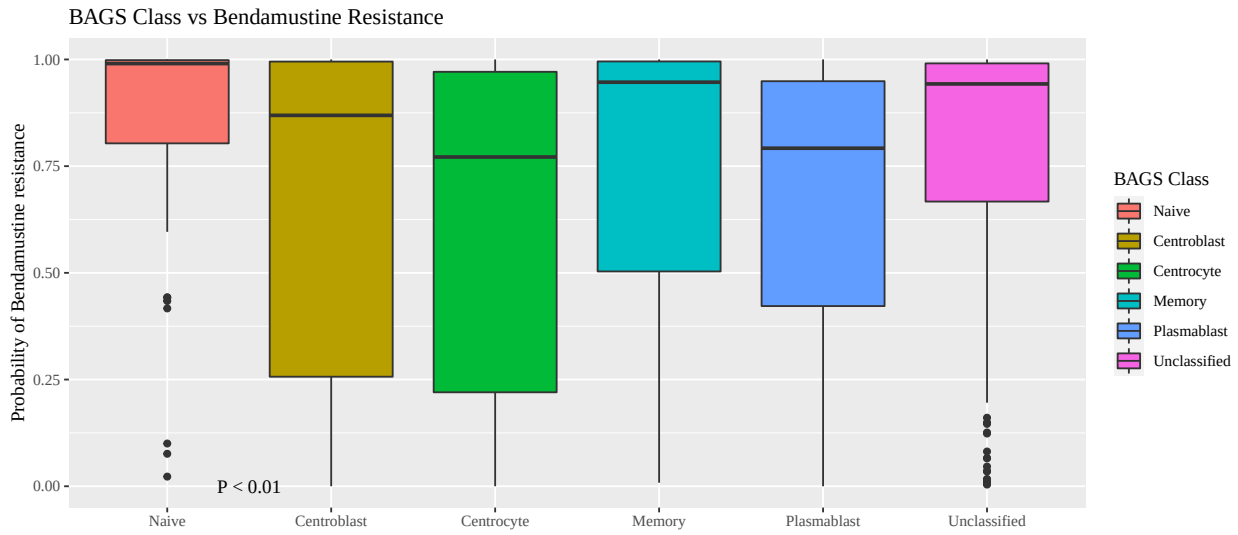
```
WrighTAKtest.lm <- lm(Bendamustine.DLBCL.res.prob~WrightClass,
                     data = subset(metaALL, !is.na(WrightClass)))
WrighTAKtest <- anova(WrighTAKtest.lm)
WrighTAKtest.kw <- kruskal.test(Bendamustine.DLBCL.res.prob~WrightClass,
                                data = subset(metaALL, !is.na(WrightClass)))

P = changeP(WrighTAKtest.kw$p.value)

dlbcl_abcgcb_plot <- ggplot(subset(metaALL, !is.na(WrightClass)),
                            aes(WrightClass, Bendamustine.DLBCL.res.prob,
                                fill = WrightClass)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "Wright Class") +
  ggtitle("ABC / GCB vs Bendamustine resistance") +
  annotate("text", x = 1.5, y = 0,
           label =paste(P))

dlbcl_abcgcb_plot
```



BAGS vs Bendamustine resistance

```
BAGSTAKtest.kw <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS, data = metaALL)

P = changeP(BAGSTAKtest.kw$p.value)

dlbcl_bags_plot <- ggplot(metaALL, aes(BAGS, Bendamustine.DLBCL.res.prob, fill = BAGS)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "BAGS Class") +
  ggtitle("BAGS Class vs Bendamustine Resistance") +
  annotate("text", x = 1.5, y = 0,
           label =paste(P))
```

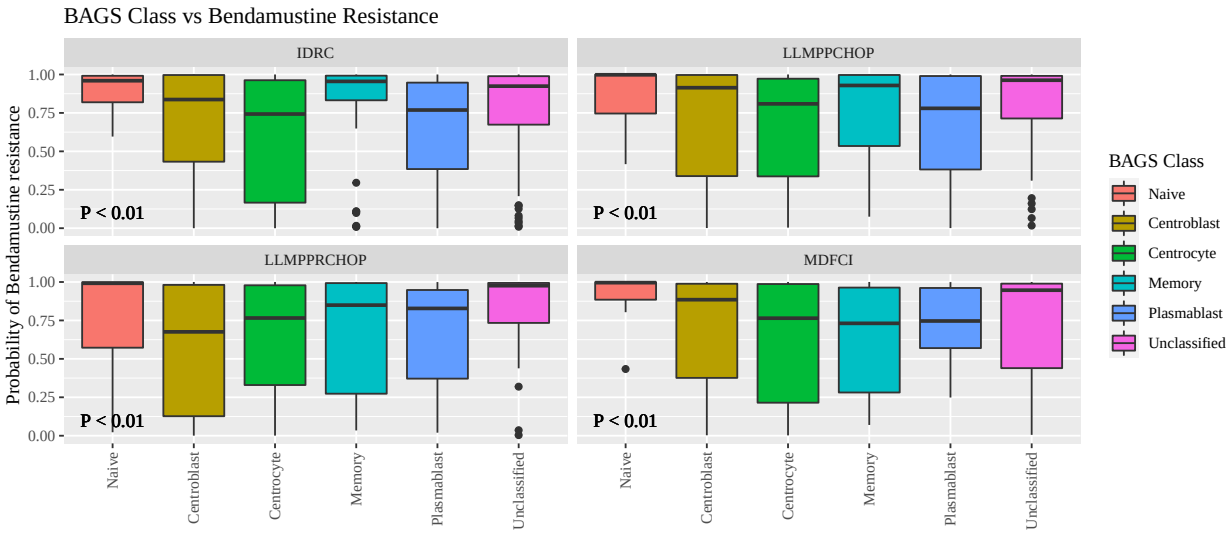
```
dlbcl_bags_plot
```



BAGS vs bendamustine resistance for individual datasets

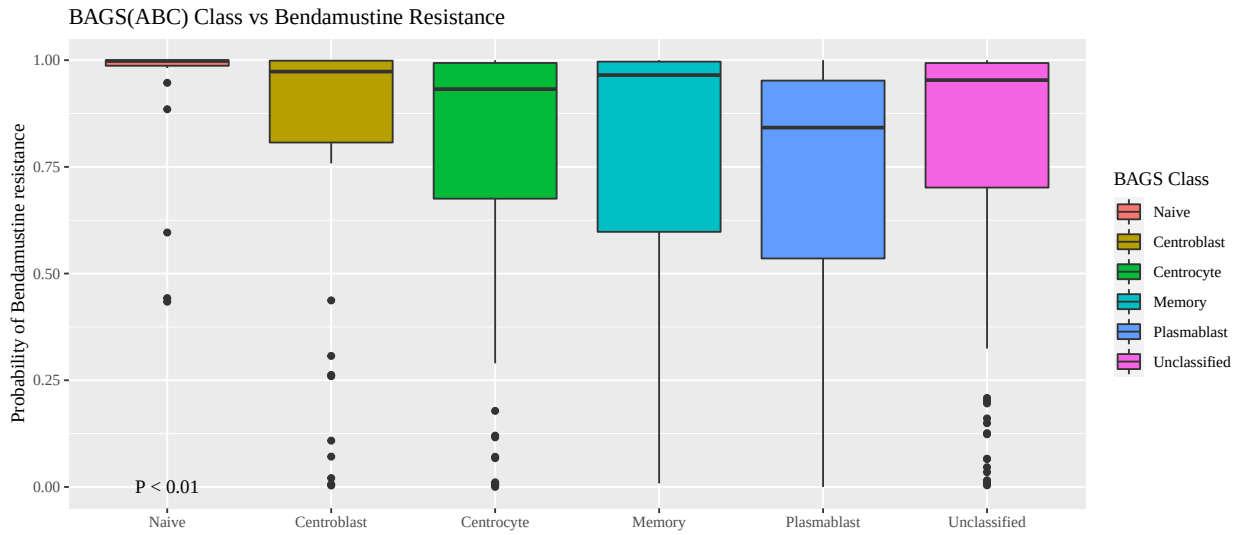
```
BAGSTAKtest.kw.i <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                 data = subset(metaALL, Data == "IDRC"))
BAGSTAKtest.kw.l1 <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                  data = subset(metaALL, Data == "LLMPPCHOP"))
BAGSTAKtest.kw.l2 <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                  data = subset(metaALL, Data == "LLMPPRCHOP"))
BAGSTAKtest.kw.m <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                 data = subset(metaALL, Data == "MDFCI"))

BAGSTAKtest.kw.i <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                 subset(metaALL, Data == "IDRC"))
BAGSTAKtest.kw.l1 <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                  subset(metaALL, Data == "LLMPPCHOP"))
BAGSTAKtest.kw.l2 <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                  subset(metaALL, Data == "LLMPPRCHOP"))
BAGSTAKtest.kw.m <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                                 subset(metaALL, Data == "MDFCI"))

ann_text <- data.frame("dataset" = c("IDRC", "LLMPPCHOP", "LLMPPRCHOP", "MDFCI"),
                       "x" = 1,
                       "y" = 0.1,
                       "lab" = c(changeP(BAGSTAKtest.kw.i$p.value),
                                 changeP(BAGSTAKtest.kw.l1$.value),
                                 changeP(BAGSTAKtest.kw.l2$.value),
                                 changeP(BAGSTAKtest.kw.m$value))
)

dlbcl_bags_plot_dataset <- ggplot(metaALL, aes(BAGS, Bendamustine.DLBCL.res.prob,
                                               fill = BAGS)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
```

```
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "BAGS Class") +
  ggtitle("BAGS Class vs Bendamustine Resistance") +
  facet_wrap(~Data) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  geom_text(data = ann_text, aes(x = x, y = y, label = lab), inherit.aes = F)
dlbcl_bags_plot_dataset
```



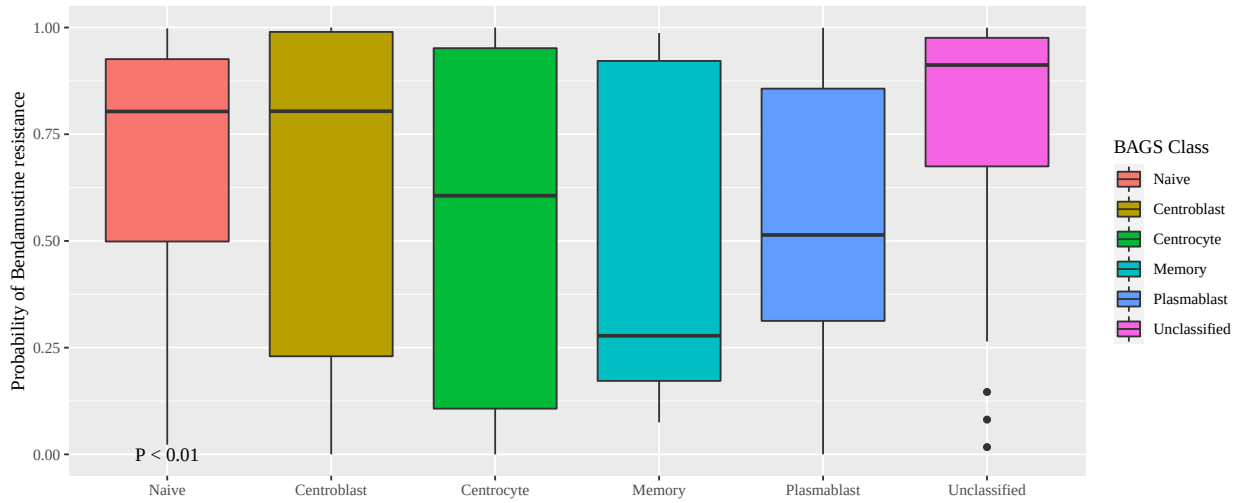
BAGS (ABC) vs Bendamustine resistance

```
BAGSTAKtest.kw <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                               data = subset(metaALL, WrightClass == "ABC"))
P = changeP(BAGSTAKtest.kw$p.value)

dlbcl_bags_abc_plot <-
  ggplot(subset(metaALL, WrightClass == "ABC"), aes(BAGS, Bendamustine.DLBCL.res.prob,
                                                    fill = BAGS)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "BAGS Class") +
  ggtitle("BAGS(ABC) Class vs Bendamustine Resistance") +
  annotate("text", x = 1, y = 0,
           label =paste(P))

dlbcl_bags_abc_plot
```

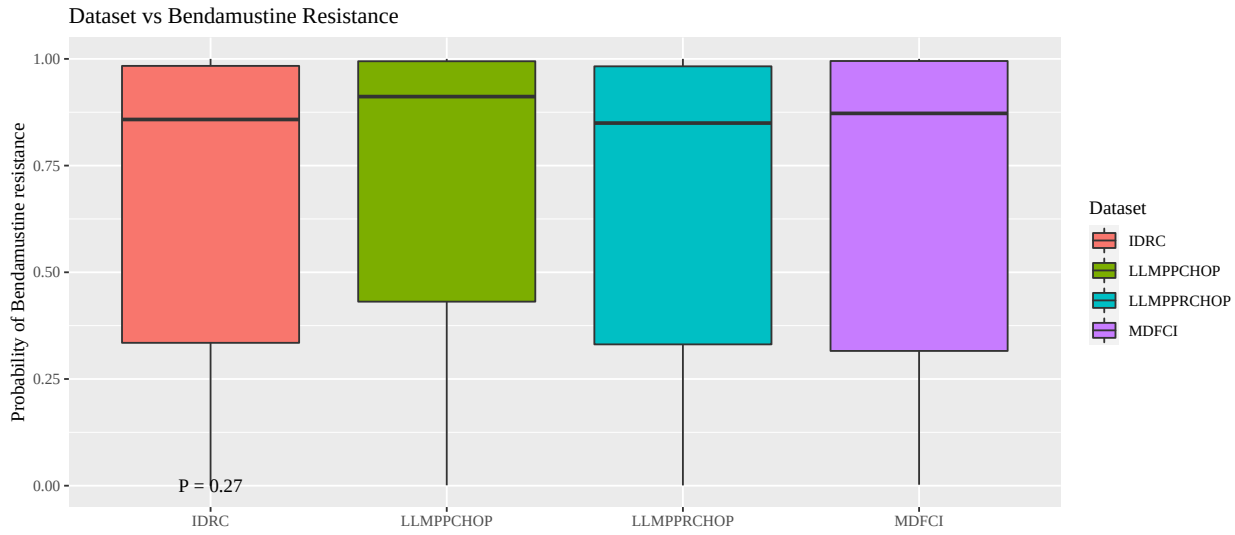


BAGS (GCB) vs Bendamustine resistance

```
BAGSTAKtest.kw <- kruskal.test(Bendamustine.DLBCL.res.prob~BAGS,
                               data = subset(metaALL, WrightClass == "GCB"))
P = changeP(BAGSTAKtest.kw$p.value)

dlbcl_bags_gcb_plot <-
  ggplot(subset(metaALL, WrightClass == "GCB"), aes(BAGS, Bendamustine.DLBCL.res.prob,
                                                    fill = BAGS)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "BAGS Class") +
  ggtitle("BAGS(GCB) Class vs Bendamustine Resistance") +
  annotate("text", x = 1, y = 0,
           label = paste(P))
dlbcl_bags_gcb_plot
```



```
DataTAKtest.kw <- kruskal.test(Bendamustine.DLBCL.res.prob~Data, data = metaALL)
P = changeP(DataTAKtest.kw$p.value)

dlbcl_dataset_plot <- ggplot(metaALL, aes(Data, Bendamustine.DLBCL.res.prob,
                                          fill = Data)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "Dataset") +
  ggtitle("Dataset vs Bendamustine Resistance") +
  annotate("text", x = 1, y = 0,
           label =paste(P))

dlbcl_dataset_plot
```



MM

Load Validation data. GEP data has previously been cohort RMA normalized for the analysis in (Bødker et al. 2018).

```
## UAMS
load("../ExternalData/UAMS/uams.RData")
load("../ExternalData/UAMS/metadataUAMS.RData")
uams.sc <- microarrayScale(uams)

## HOVON65
load("../ExternalData/Hovon65/hovon65.RData")
load("../ExternalData/Hovon65/metadataHovon65.RData")
hovon65.sc <- microarrayScale(hovon65)

## Myeloma IX
load("../ExternalData/MyelomaIX/myelomaIX.RData")
load("../ExternalData/MyelomaIX/metadataMyelomaIX.RData")
myelomaIX.sc <- microarrayScale(myelomaIX)
```

BAGS-MM Classify (Bødker et al. 2018)

```
load("../ExternalData/BAGSClassifierTools.RData")
metadataUAMS$BAGS <- BAGSclassifier(uams.sc, BAGS.coef = BAGS.coef)$class
metadataHovon65$BAGS <- BAGSclassifier(hovon65.sc, BAGS.coef = BAGS.coef)$class
metadataMyelomaIX$BAGS <- BAGSclassifier(myelomaIX.sc, BAGS.coef = BAGS.coef)$class
```

Function for TC Classification (Bergsagel et al. 2005).

```
maxRatio <- function(x) {
  # Ratio between each element of vector x and max of x
  x/max(x)
}
TCClassify <- function(x) {
  # Adjustment Bergsagel et al.'s TC classifier to remapped CDFs
  # x should be a matrix
  # Hardcode the symbol to ensembl_gene_id mappings
  #mart <- useMart(biomart = "ensembl", dataset = "hsapiens_gene_ensembl")
  ensIDs <-
    c("FGFR3" = "ENSG00000068078", "WHSC1" = "ENSG00000109685",
      "MAF" = "ENSG00000178573", "ITGB7" = "ENSG00000139626",
      "CX3CR1" = "ENSG00000168329", "CCND3" = "ENSG00000112576",
      "CCND1" = "ENSG00000110092", "CCND2" = "ENSG00000118971")
  classes <- c("4p16", "MAF", "6p21", "11q13", "D1", "D1plusD2", "D2", "Unclassified")
  if (!all(ensIDs %in% rownames(x))) {
    cat("All probesets not present in data.\n")
  }
  t1 <- maxRatio(2^x[ensIDs["FGFR3"],]) > 0.30 | maxRatio(2^x[ensIDs["WHSC1"],]) > 0.10
  t2 <- maxRatio(2^x[ensIDs["ITGB7"],]) > 0.18 & maxRatio(2^x[ensIDs["CX3CR1"],]) > 0.02
  t3 <- maxRatio(2^x[ensIDs["CCND3"], ]) > 0.5
  t4 <- maxRatio(2^x[ensIDs["CCND1"], ]) > 0.25
  t5 <- maxRatio(2^x[ensIDs["CCND2"], ]) < 0.09 & maxRatio(2^x[ensIDs["CCND1"], ]) > 0.024
  t6 <- maxRatio(2^x[ensIDs["CCND2"], ]) > 0.09 & maxRatio(2^x[ensIDs["CCND1"], ]) > 0.024
  t7 <- maxRatio(2^x[ensIDs["CCND2"], ]) > 0.09 & maxRatio(2^x[ensIDs["CCND1"], ]) < 0.024
  class <- rep("Unclassified", ncol(x))
  class[ t1] <- classes[1]
  class[!t1 & t2] <- classes[2]
  class[!t1 & !t2 & t3] <- classes[3]
  class[!t1 & !t2 & !t3 & t4] <- classes[4]
  class[!t1 & !t2 & !t3 & !t4 & t5] <- classes[5]
  class[!t1 & !t2 & !t3 & !t4 & !t5 & t6] <- classes[6]
  class[!t1 & !t2 & !t3 & !t4 & !t5 & !t6 & t7] <- classes[7]
  names(class) <- colnames(x)
  return(factor(class, levels = classes))
}
```

TC classify

```
metadataUAMS$TCClass      <- TCClassify(uams)
metadataHovon65$TCClass   <- TCClassify(hovon65)
metadataMyelomaIX$TCClass <- TCClassify(myelomaIX)
```

Bendamustine REGS classify

```
## Rename coefs to match names in clinical cohorts
b.mm.coef <- bendamustine.resistance.class.coef
names(b.mm.coef) <- sub("_at", "", names(b.mm.coef))

metadataUAMS$Bendamustine.MM.res.prob <- bendamustineClassifier(uams.sc,
                                                                coef = b.mm.coef)
metadataUAMS$Bendamustine.MM.res.class <- cutFun(metadataUAMS$Bendamustine.MM.res.prob)


metadataHovon65$Bendamustine.MM.res.prob <- bendamustineClassifier(hovon65.sc,
                                                                   coef = b.mm.coef)
metadataHovon65$Bendamustine.MM.res.class <- cutFun(metadataHovon65$Bendamustine.MM.res.prob)

metadataMyelomaIX$Bendamustine.MM.res.prob <- bendamustineClassifier(myelomaIX.sc,
                                                                     coef = b.mm.coef)
metadataMyelomaIX$Bendamustine.MM.res.class <- cutFun(metadataMyelomaIX$Bendamustine.MM.res.prob)
```

Create clinical MM metadataset

```
vars <- c("BAGS", "TCClass","Bendamustine.MM.res.prob",
          "Bendamustine.MM.res.class", "EFS", "OS")
MM.meta <- rbind(metadataUAMS[,vars],
                 metadataHovon65[,vars],
                 metadataMyelomaIX[,vars])
MM.meta$dataset <- c(rep("UAMS", nrow(metadataUAMS)),
                     rep("HOVON-65", nrow(metadataHovon65)),
                     rep("MyelomaIX", nrow(metadataMyelomaIX)))
```

Tabulate data

```
BAGSMMtable <- tablePct(MM.meta$dataset, MM.meta$BAGS)
TCclasstable <- tablePct(MM.meta$dataset, MM.meta$TCClass)

kable(BAGSMMtable, caption = "BAGS-MM class by dataset")
```

Table 8: BAGS-MM class by dataset

	Pre-BI	Pre-BII	Immature	Naive	Memory	Plasmacell	Unclassified	Sum
HOVON-65	2 (1)	19 (6)	45 (14)	61 (19)	134 (42)	11 (3)	48 (15)	320 (100)
MyelomaIX	1 (0)	14 (6)	23 (9)	59 (24)	105 (43)	8 (3)	37 (15)	247 (100)
UAMS	3 (1)	29 (5)	58 (10)	143 (26)	219 (39)	23 (4)	84 (15)	559 (100)
Sum	6 (1)	62 (6)	126 (11)	263 (23)	458 (41)	42 (4)	169 (15)	1126 (100)

```
kable(TCclasstable, caption = "TC class by dataset")
```

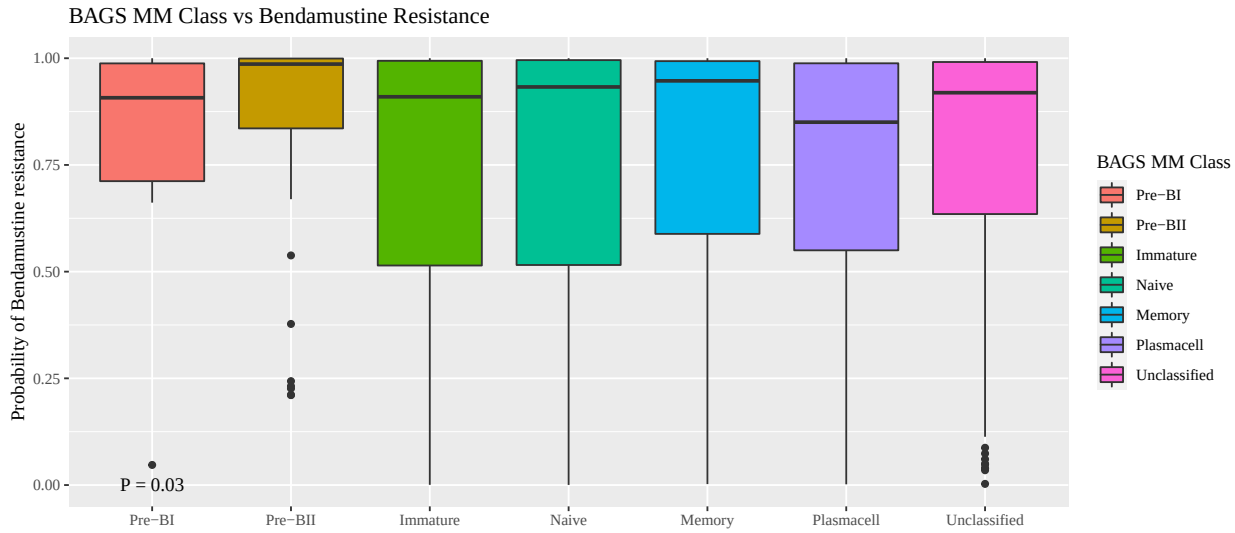
Table 9: TC class by dataset

	4p16	MAF	6p21	11q13	D1	D1plusD2	D2	Unclassified	Sum
HOVON-65	39 (12)	13 (4)	3 (1)	42 (13)	105 (33)	4 (1)	40 (12)	74 (23)	320 (100)

	4p16	MAF	6p21	11q13	D1	D1plusD2	D2	Unclassified	Sum
MyelomaIX	33 (13)	6 (2)	4 (2)	24 (10)	77 (31)	1 (0)	31 (13)	71 (29)	247 (100)
UAMS	78 (14)	37 (7)	8 (1)	85 (15)	210 (38)	15 (3)	45 (8)	81 (14)	559 (100)
Sum	150 (13)	56 (5)	15 (1)	151 (13)	392 (35)	20 (2)	116 (10)	226 (20)	1126 (100)

BAGS-MM vs bendamustine resistance

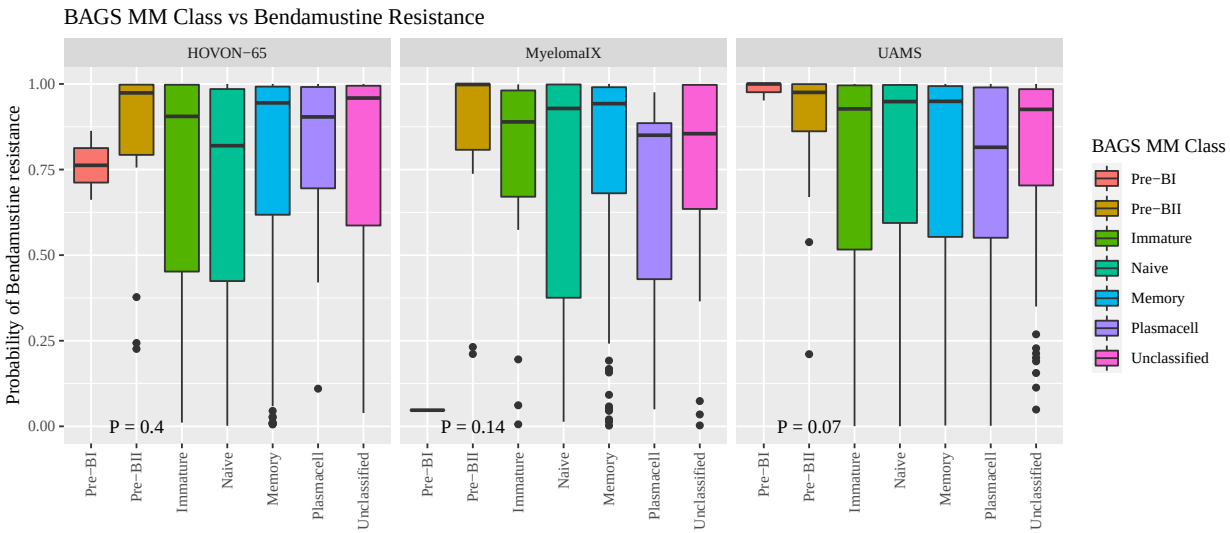
```
BAGSTAKtest.kw <- kruskal.test(Bendamustine.MM.res.prob~BAGS, data = MM.meta)
P = changeP(BAGSTAKtest.kw$p.value)

mm_bags_plot <- ggplot(MM.meta, aes(BAGS, Bendamustine.MM.res.prob,
                                    fill = BAGS)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "BAGS MM Class") +
  ggtitle("BAGS MM Class vs Bendamustine Resistance") +
  annotate("text", x = 1, y = 0,
           label =paste(P))
mm_bags_plot
```



BAGS-MM vs bendamustine resistance for individual datasets

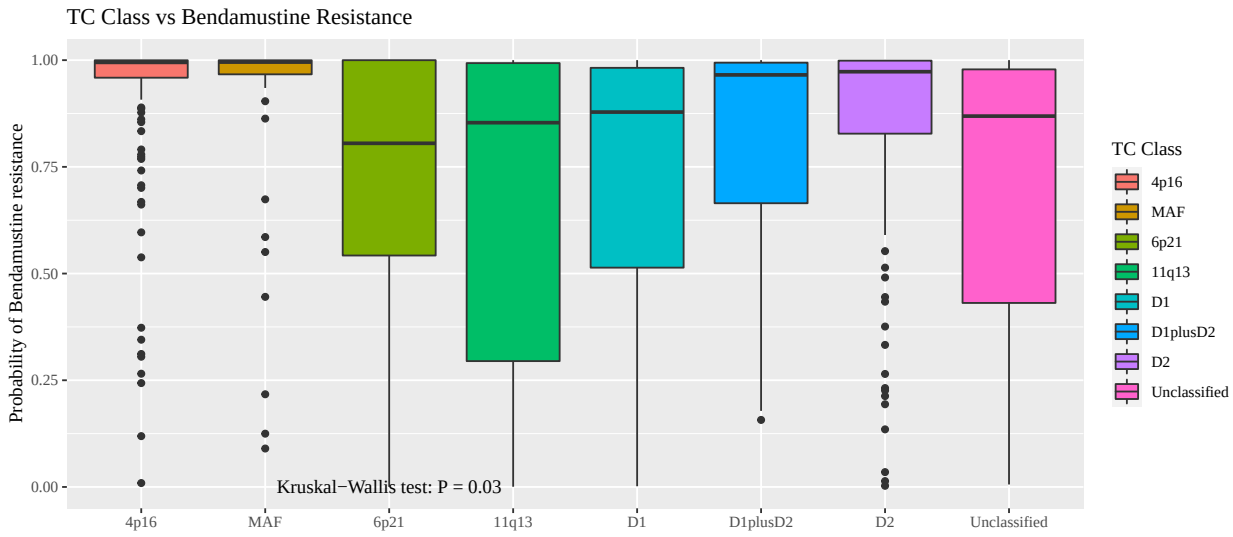
```
BAGSTAKtest.kw.h <- kruskal.test(Bendamustine.MM.res.prob~BAGS,
                                 data = subset(MM.meta, dataset == "HOVON-65"))
BAGSTAKtest.kw.m <- kruskal.test(Bendamustine.MM.res.prob~BAGS,
                                 data = subset(MM.meta, dataset == "MyelomaIX"))
BAGSTAKtest.kw.u <- kruskal.test(Bendamustine.MM.res.prob~BAGS,
                                 data = subset(MM.meta, dataset == "UAMS"))

ann_text <- data.frame("dataset" = c("HOVON-65", "MyelomaIX", "UAMS"),
                       "x" = 2,
                       "y" = 0.0,
                       "lab" = c(changeP(BAGSTAKtest.kw.h$p.value),
```

```
                                changeP(BAGSTAKtest.kw.m$p.value),
                                changeP(BAGSTAKtest.kw.u$p.value))
)

mm_bags_plot_dataset <- ggplot(MM.meta, aes(BAGS, Bendamustine.MM.res.prob,
                                            fill = BAGS)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "BAGS MM Class") +
  ggtitle("BAGS MM Class vs Bendamustine Resistance") +
  facet_wrap(~dataset) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  geom_text(data = ann_text, aes(x = x, y = y, label = lab), inherit.aes = F)
mm_bags_plot_dataset
```



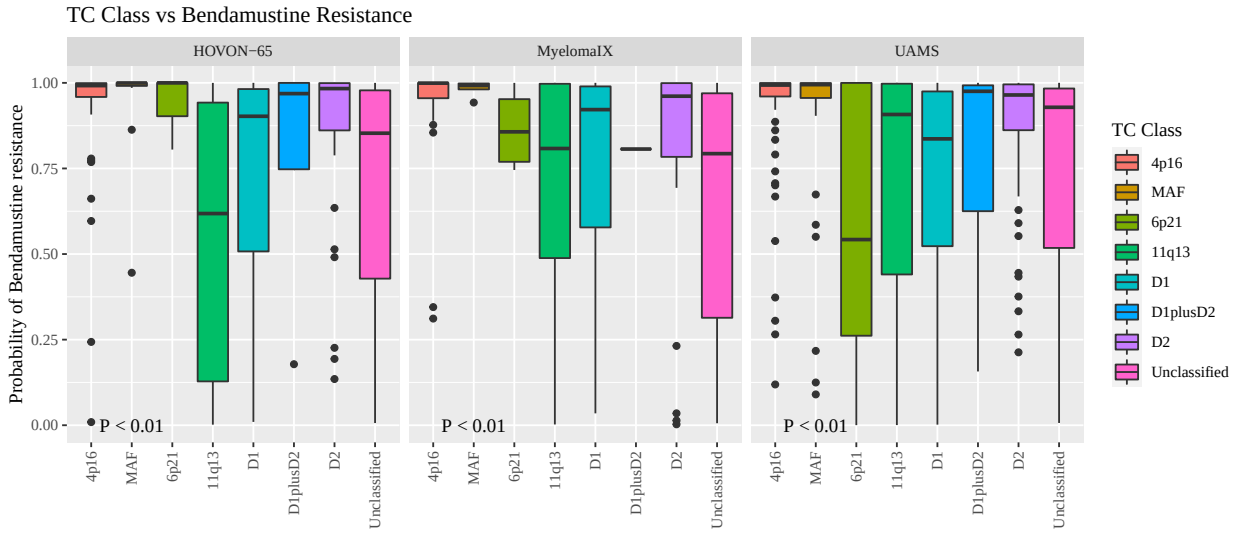
BAGS-MM vs bendamustine resistance

```
TCtest.kw <- kruskal.test(Bendamustine.MM.res.prob~TCClass, data = MM.meta)
P = changeP(BAGSTAKtest.kw$p.value)

mm_TC_plot <- ggplot(MM.meta, aes(TCClass, Bendamustine.MM.res.prob,
                                  fill = TCClass)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "TC Class") +
  ggtitle("TC Class vs Bendamustine Resistance") +
  annotate("text", x = 3, y = 0,
           label =paste("Kruskal-Wallis test:", P))
mm_TC_plot
```



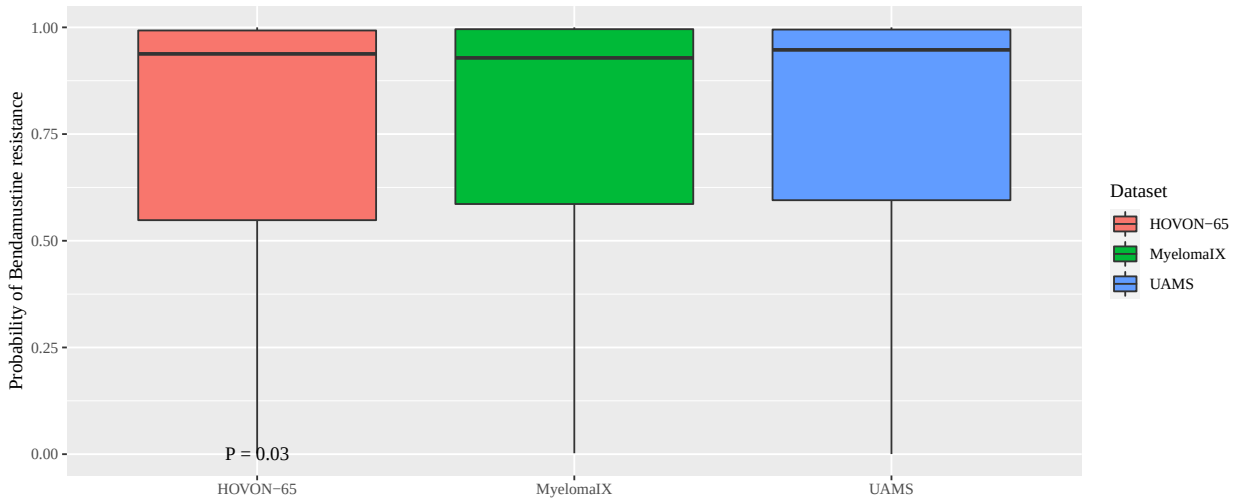
TC Class vs bendamustine resistance for individual datasets

```
TCtest.kw.h <- kruskal.test(Bendamustine.MM.res.prob~TCClass,
                            data = subset(MM.meta, dataset == "HOVON-65"))
TCtest.kw.m <- kruskal.test(Bendamustine.MM.res.prob~TCClass,
                            data = subset(MM.meta, dataset == "MyelomaIX"))
TCtest.kw.u <- kruskal.test(Bendamustine.MM.res.prob~TCClass,
                            data = subset(MM.meta, dataset == "UAMS"))

ann_text <- data.frame("dataset" = c("HOVON-65", "MyelomaIX", "UAMS"),
                       "x" = 2,
                       "y" = 0.0,
                       "lab" = c(changeP(TCtest.kw.h$p.value),
                                 changeP(TCtest.kw.m$p.value),
                                 changeP(TCtest.kw.u$p.value))
)

mm_TC_plot_dataset <- ggplot(MM.meta, aes(TCClass, Bendamustine.MM.res.prob,
                                          fill = TCClass)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "TC Class") +
  ggtitle("TC Class vs Bendamustine Resistance") +
  facet_wrap(~dataset) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  geom_text(data = ann_text, aes(x = x, y = y, label = lab), inherit.aes = F)
mm_TC_plot_dataset
```



Dataset vs bendamustine resistance

```
DatasetTAKtest.kw <- kruskal.test(Bendamustine.MM.res.prob~dataset, data = MM.meta)
P = changeP(BAGSTAKtest.kw$p.value)

mm_dataset_plot <- ggplot(MM.meta, aes(dataset, Bendamustine.MM.res.prob,
                                       fill = dataset)) +
  geom_boxplot() +
  scale_x_discrete(name = "") +
  scale_y_continuous(name = "Probability of Bendamustine resistance") +
  scale_fill_discrete(name = "Dataset") +
  ggtitle("Dataset vs Bendamustine Resistance") +
  annotate("text", x = 1, y = 0.0,
           label =paste(P))

mm_dataset_plot
```



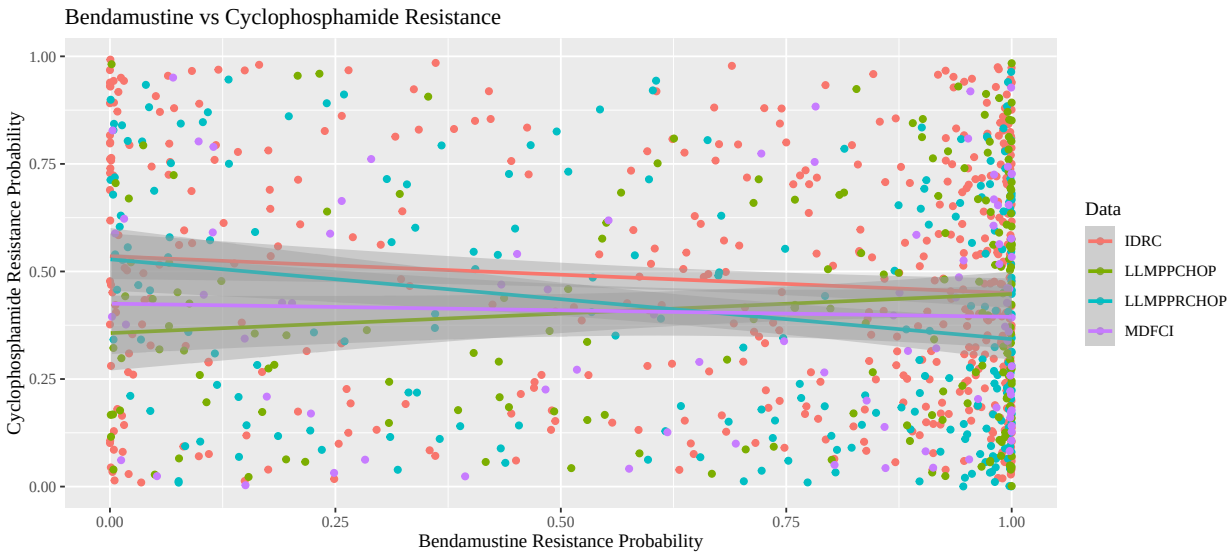
Survival analysis

Plot cyclophosphamide Resistance vs Bendamustine resistance at patient level. First we predict the cyclophosphamide resistance for DLBCL patients using the REGS classifier from (Falgreen et al. 2015)

```
metaALL$Cyclophosphamide.res.prob <- CyclophosphamideClassifier(exprs(gepALL.affy))
```

Then we plot the predicted resistance of cyclophosphamide vs bendamustine.

```
ggplot(metaALL, aes(x = Bendamustine.DLBCL.res.prob, y = Cyclophosphamide.res.prob, color = Data)) +
  geom_point() +
  ggtitle("Bendamustine vs Cyclophosphamide Resistance") +
  xlab("Bendamustine Resistance Probability") +
  ylab("Cyclophosphamide Resistance Probability") +
  geom_smooth(method='lm', formula= y~x)
```




```
ggsave(filename = "../output/Figures/Fig_S3.tiff",width = 7,
       height = 5, dpi = 600, compression = "lzw")
```

```
## Restructure surv objects
metaALL2 <- metaALL %>% filter(Data != "MDFCI")

metaALL2$OS <- Surv(metaALL2$OS[,1], metaALL2$OS[,2])
metaALL2$PFS <- Surv(metaALL2$PFS[,1], metaALL2$PFS[,2])
```

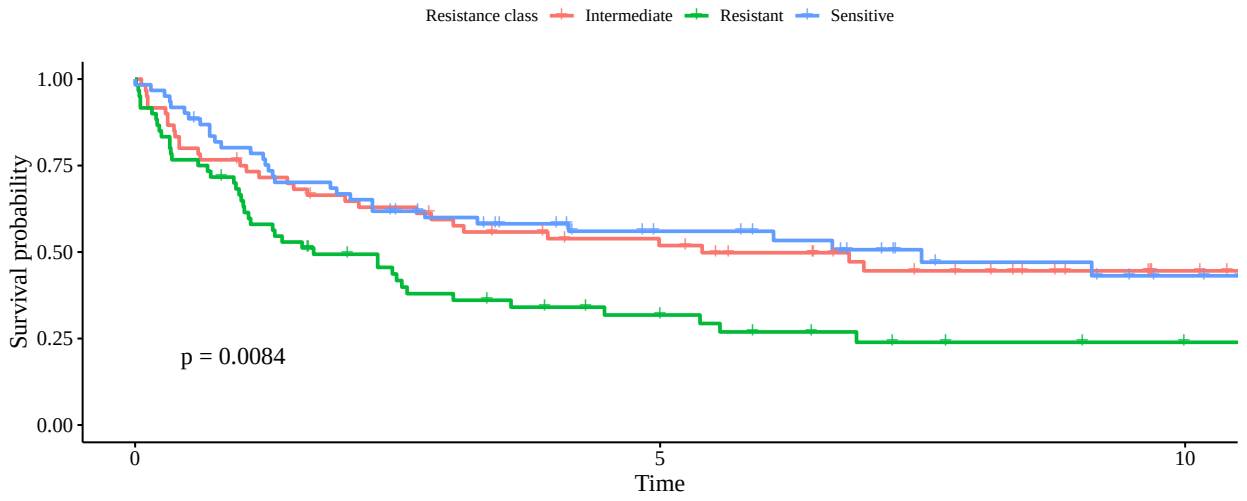
OS

Plot Overall survival for individual datasets

```
os.plot.list <- list()
for(dataset in c("LLMPPRCHOP","LLMPPCHOP" )){
  meta.list <- metaALL2 %>%
    filter(Data == dataset) %>%
```

```
    mutate(Bendamustine.DLBCL.res.class = cutFun(Bendamustine.DLBCL.res.prob))

  sf.os <- survfit(meta.list$OS ~ meta.list$Bendamustine.DLBCL.res.class)
  os.plot.list[[dataset]] <-
    ggsurvplot(sf.os,
               data = meta.list,
               title = paste0(dataset,": Bendamustine resistance class vs OS"),
               legend.labs = c("Intermediate", "Resistant", "Sensitive"),
               legend.title = "Resistance class",
               xlim = c(0,10),
               pval = T)
}
```

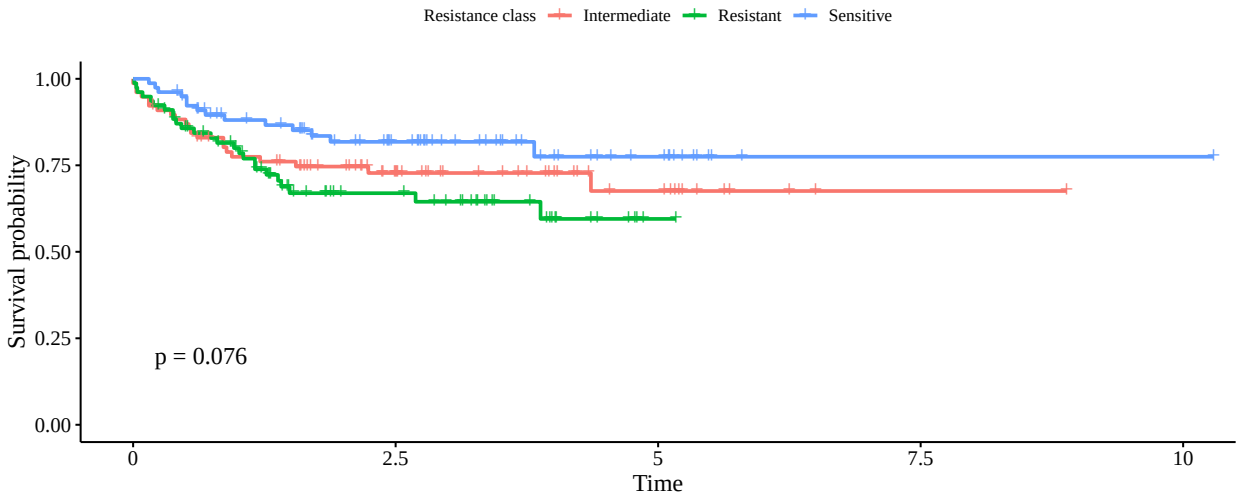
```
os.plot.list[["LLMPPCHOP"]]
```




```
ggsave(print(os.plot.list[["LLMPPCHOP"]]), filename = "../output/Figures/Fig_S2A.tiff",width = 7,
       height = 5, dpi = 600, compression = "lzw")
```

```
os.plot.list[["LLMPPRCHOP"]]
```

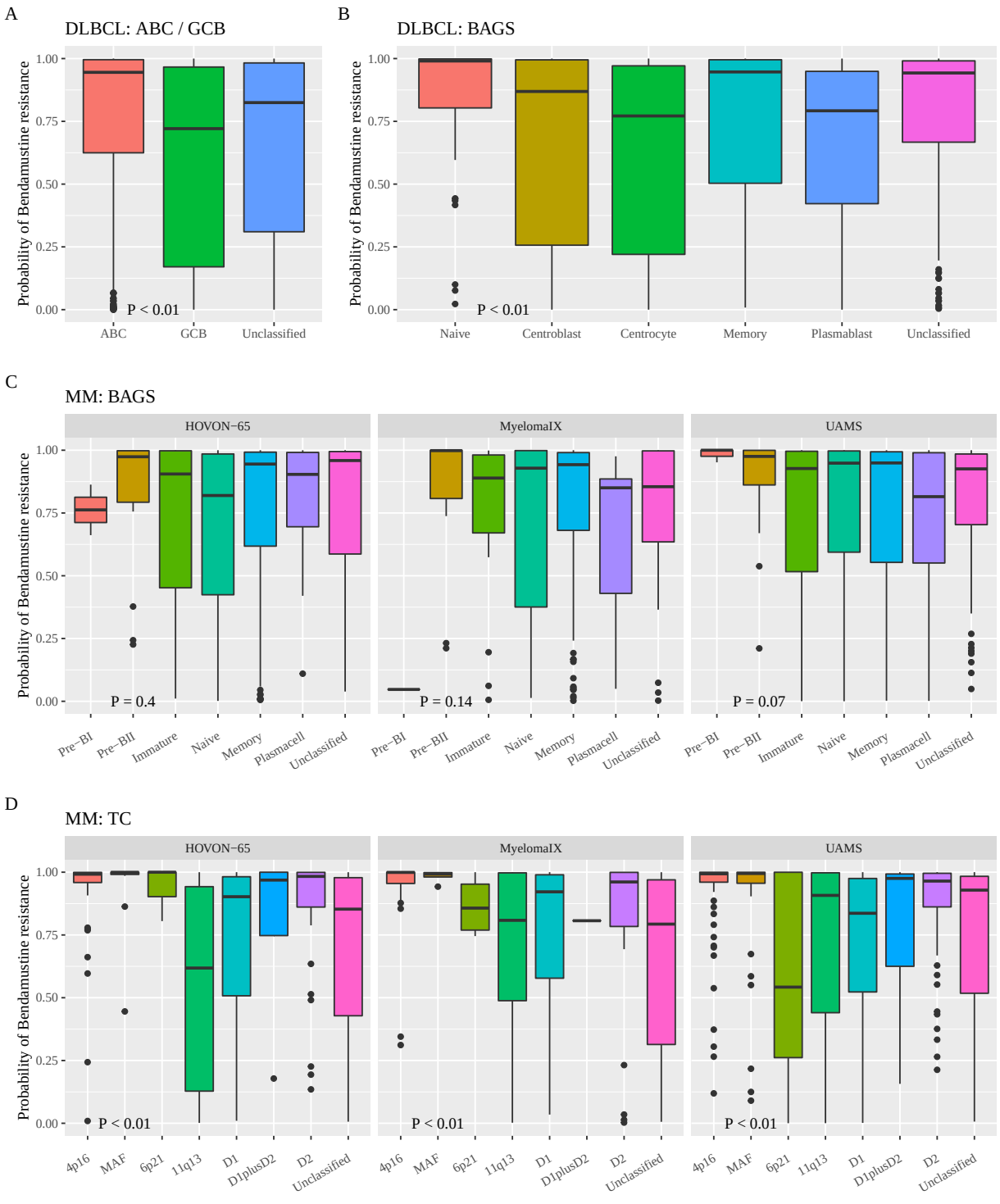
LLMPPRCHOP: Bendamustine resistance class vs OS




```
ggsave(print(os.plot.list[["LLMPPRCHOP"]]), filename = "../output/Figures/Fig_S2B.tiff",width = 7,
       height = 5, dpi = 600, compression = "lzw")
```

Combine Plots for publication

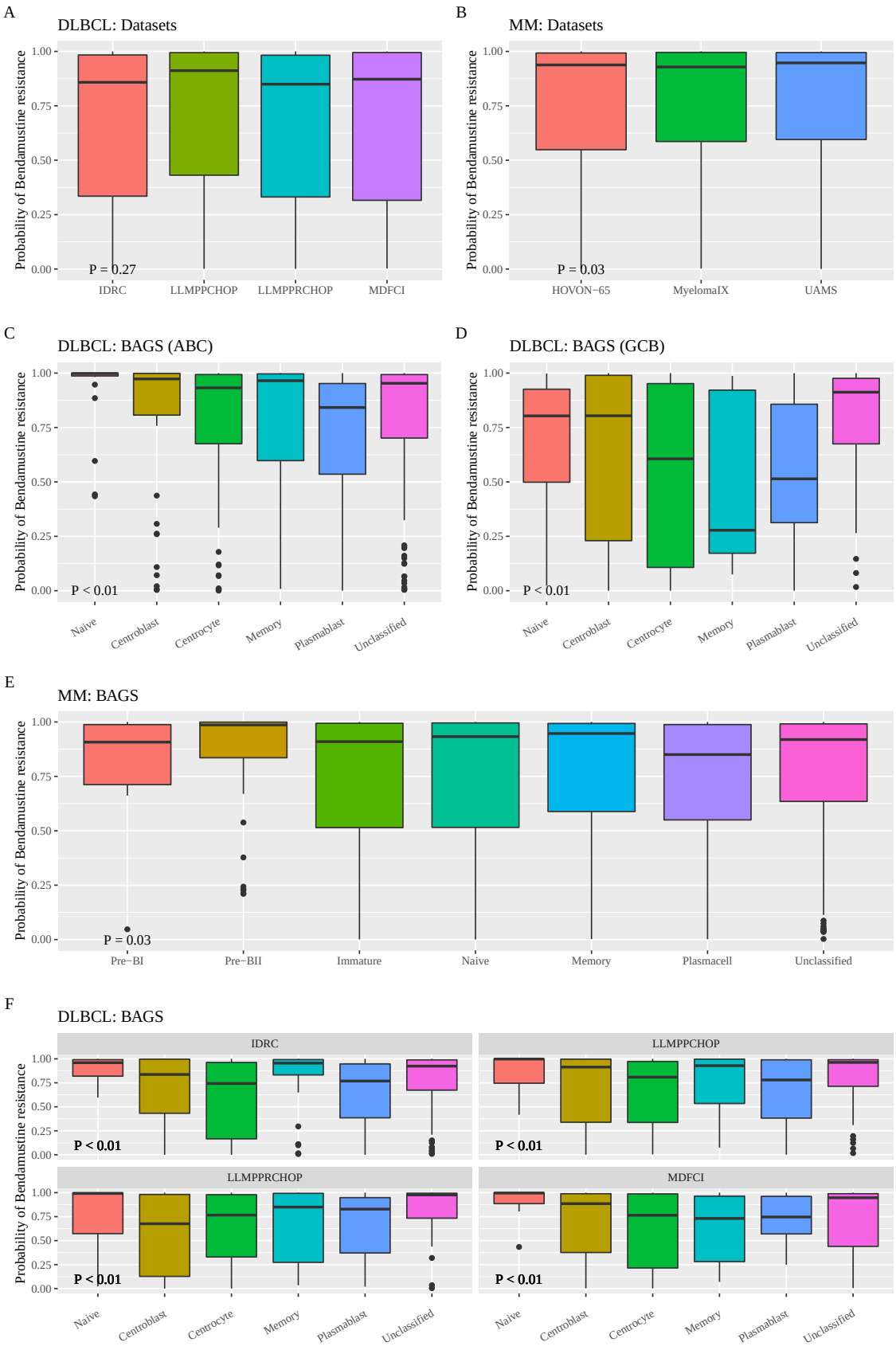
```
p1 <- dlbcl_abcgcb_plot + theme(legend.position = "none") +
  ggtitle("DLBCL: ABC / GCB")
p2 <- dlbcl_bags_plot + theme(legend.position = "none") +
  ggtitle("DLBCL: BAGS")
p3 <- mm_bags_plot_dataset + theme(legend.position = "none") +
  ggtitle("MM: BAGS") +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))
p4 <- mm_TC_plot_dataset + theme(legend.position = "none") +
  ggtitle("MM: TC") +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))

(p1 + p2 + plot_layout(widths = c(0.3, 0.7))) / p3 / p4 +
  plot_annotation(tag_levels = "A")
```




```
ggsave(filename = "../output/Figures/Fig_3.tiff",width = 10,
       height = 12, dpi = 600, compression = "lzw")
```

```
p1 <- dlbcl_dataset_plot + theme(legend.position = "none") + ggtitle("DLBCL: Datasets")
  #theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))
```

```
p2 <- mm_dataset_plot + theme(legend.position = "none") + ggtitle("MM: Datasets")
  #theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))

p3 <- dlbcl_bags_abc_plot + theme(legend.position = "none") + ggtitle("DLBCL: BAGS (ABC)") +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))

p4 <- dlbcl_bags_gcb_plot + theme(legend.position = "none") + ggtitle("DLBCL: BAGS (GCB)") +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))

p5 <- mm_bags_plot + theme(legend.position = "none") + ggtitle("MM: BAGS") #+
  #theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))

p6 <- dlbcl_bags_plot_dataset + theme(legend.position = "none") + ggtitle("DLBCL: BAGS") +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.7, hjust=1))

(p1 + p2 ) / (p3 + p4) / p5 / p6 +
    plot_annotation(tag_levels = "A")
```


A
DLBCL: Datasets
Probability of Bendamustine resistance
1.00
0.75
0.50
0.25
0.00
P = 0.27
IDRC
LLMPPCHOP
LLMPPRCHOP
MDFCI
B
MM: Datasets
Probability of Bendamustine resistance
P = 0.03
HOVON−65
MyelomaIX
UAMS
C
DLBCL: BAGS (ABC)
Probability of Bendamustine resistance
P < 0.01
Naive
Centroblast
Centrocyte
Memory
Plasmablast
Unclassified
D
DLBCL: BAGS (GCB)
Probability of Bendamustine resistance
P < 0.01
Naive
Centroblast
Centrocyte
Memory
Plasmablast
Unclassified
E
MM: BAGS
Probability of Bendamustine resistance
P = 0.03
Pre−BI
Pre−BII
Immature
Naive
Memory
Plasmacell
Unclassified
F
DLBCL: BAGS
IDRC
LLMPPCHOP
LLMPPRCHOP
MDFCI
Probability of Bendamustine resistance
P < 0.01
Naive
Centroblast
Centrocyte
Memory
Plasmablast
Unclassified

```
ggsave(filename = "../output/Figures/Fig_S1.tiff",width = 10,
       height = 15, dpi = 600, compression = "lzw")
```

Session info

```
devtools::session_info()
```

```
## - Session info -------------------------------------------------------------------------------------
##  setting  value
##  version  R version 4.0.3 (2020-10-10)
##  os       Windows 10 x64
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  Danish_Denmark.1252
##  ctype    Danish_Denmark.1252
##  tz       Europe/Paris
##  date     2020-10-30
##
## - Packages -----------------------------------------------------------------------------------------
##  package        * version     date       lib source
##  abind            1.4-5       2016-07-21 [1] CRAN (R 4.0.0)
##  affy           * 1.66.0      2020-04-27 [1] Bioconductor
##  affyio           1.58.0      2020-04-27 [1] Bioconductor
##  animation      * 2.6         2018-12-11 [1] CRAN (R 4.0.0)
##  AnnotationDbi  * 1.50.3      2020-07-25 [1] Bioconductor
##  askpass          1.1         2019-01-13 [1] CRAN (R 4.0.0)
##  assertthat       0.2.1       2019-03-21 [1] CRAN (R 4.0.0)
##  backports        1.1.10      2020-09-15 [1] CRAN (R 4.0.2)
##  base64enc        0.1-3       2015-07-28 [1] CRAN (R 4.0.0)
##  Biobase        * 2.48.0      2020-04-27 [1] Bioconductor
##  BiocFileCache    1.12.1      2020-08-04 [1] Bioconductor
##  BiocGenerics   * 0.34.0      2020-04-27 [1] Bioconductor
##  BiocManager      1.30.10     2019-11-16 [1] CRAN (R 4.0.0)
##  biomaRt        * 2.44.1      2020-06-17 [1] Bioconductor
##  bit              4.0.4       2020-08-04 [1] CRAN (R 4.0.2)
##  bit64            4.0.5       2020-08-30 [1] CRAN (R 4.0.2)
##  bitops           1.0-6       2013-08-17 [1] CRAN (R 4.0.0)
##  blob             1.2.1       2020-01-20 [1] CRAN (R 4.0.0)
##  broom            0.7.1       2020-10-02 [1] CRAN (R 4.0.2)
##  callr            3.5.1       2020-10-13 [1] CRAN (R 4.0.3)
##  car              3.0-10      2020-09-29 [1] CRAN (R 4.0.2)
##  carData          3.0-4       2020-05-22 [1] CRAN (R 4.0.0)
##  cellranger       1.1.0       2016-07-27 [1] CRAN (R 4.0.0)
##  checkmate        2.0.0       2020-02-06 [1] CRAN (R 4.0.0)
##  cli              2.1.0       2020-10-12 [1] CRAN (R 4.0.3)
##  cluster          2.1.0       2019-06-19 [2] CRAN (R 4.0.3)
##  codetools        0.2-16      2018-12-24 [2] CRAN (R 4.0.3)
##  colorspace       1.4-1       2019-03-18 [1] CRAN (R 4.0.0)
##  crayon           1.3.4       2017-09-16 [1] CRAN (R 4.0.0)
```

```
##  curl            4.3         2019-12-02 [1] CRAN (R 4.0.0)
##  data.table      1.13.0      2020-07-24 [1] CRAN (R 4.0.2)
##  DBI             1.1.0       2019-12-15 [1] CRAN (R 4.0.0)
##  dbplyr          1.4.4       2020-05-27 [1] CRAN (R 4.0.0)
##  DEoptimR        1.0-8       2016-11-19 [1] CRAN (R 4.0.0)
##  desc            1.2.0       2018-05-01 [1] CRAN (R 4.0.0)
##  devtools        2.3.2       2020-09-18 [1] CRAN (R 4.0.2)
##  digest          0.6.25      2020-02-23 [1] CRAN (R 4.0.3)
##  DoseR         * 0.1         2020-06-12 [1] Github (HaemAalborg/DoseR@706369f)
##  dplyr         * 1.0.2       2020-08-18 [1] CRAN (R 4.0.2)
##  ellipsis        0.3.1       2020-05-15 [1] CRAN (R 4.0.0)
##  evaluate        0.14        2019-05-28 [1] CRAN (R 4.0.0)
##  fansi           0.4.1       2020-01-08 [1] CRAN (R 4.0.0)
##  farver          2.0.3       2020-01-16 [1] CRAN (R 4.0.0)
##  fastmap         1.0.1       2019-10-08 [1] CRAN (R 4.0.0)
##  forcats         0.5.0       2020-03-01 [1] CRAN (R 4.0.0)
##  foreach         1.5.1       2020-10-15 [1] CRAN (R 4.0.3)
##  foreign       * 0.8-80      2020-05-24 [1] CRAN (R 4.0.0)
##  Formula       * 1.2-4       2020-10-16 [1] CRAN (R 4.0.3)
##  fs              1.5.0       2020-07-31 [1] CRAN (R 4.0.2)
##  gdata         * 2.18.0      2017-06-06 [1] CRAN (R 4.0.0)
##  generics        0.0.2       2018-11-29 [1] CRAN (R 4.0.0)
##  GGally        * 2.0.0       2020-06-06 [1] CRAN (R 4.0.2)
##  ggplot2       * 3.3.2       2020-06-19 [1] CRAN (R 4.0.2)
##  ggpubr        * 0.4.0       2020-06-27 [1] CRAN (R 4.0.2)
##  ggsignif        0.6.0       2019-08-08 [1] CRAN (R 4.0.2)
##  glmnet        * 4.0-2       2020-06-16 [1] CRAN (R 4.0.2)
##  glue            1.4.2       2020-08-27 [1] CRAN (R 4.0.2)
##  GO.db         * 3.11.4      2020-08-24 [1] Bioconductor
##  googleVis       0.6.6       2020-07-08 [1] CRAN (R 4.0.2)
##  graph         * 1.66.0      2020-04-27 [1] Bioconductor
##  gridExtra       2.3         2017-09-09 [1] CRAN (R 4.0.0)
##  gtable          0.3.0       2019-03-25 [1] CRAN (R 4.0.0)
##  gtools          3.8.2       2020-03-31 [1] CRAN (R 4.0.0)
##  haven           2.3.1       2020-06-01 [1] CRAN (R 4.0.0)
##  hemaClass     * 1.3.2       2020-07-07 [1] Github (HaemAalborg/hemaClass@63bd1c4)
##  highr           0.8         2019-03-20 [1] CRAN (R 4.0.0)
##  Hmisc         * 4.4-1       2020-08-10 [1] CRAN (R 4.0.2)
##  hms             0.5.3       2020-01-08 [1] CRAN (R 4.0.0)
##  htmlTable       2.1.0       2020-09-16 [1] CRAN (R 4.0.2)
##  htmltools       0.5.0       2020-06-16 [1] CRAN (R 4.0.2)
##  htmlwidgets     1.5.2       2020-10-03 [1] CRAN (R 4.0.2)
##  httpuv          1.5.4       2020-06-06 [1] CRAN (R 4.0.0)
##  httr            1.4.2       2020-07-20 [1] CRAN (R 4.0.2)
##  IRanges       * 2.22.2      2020-05-21 [1] Bioconductor
##  iterators       1.0.13      2020-10-15 [1] CRAN (R 4.0.3)
##  jpeg            0.1-8.1     2019-10-24 [1] CRAN (R 4.0.0)
##  jsonlite        1.7.1       2020-09-07 [1] CRAN (R 4.0.2)
##  km.ci           0.5-2       2009-08-30 [1] CRAN (R 4.0.2)
##  KMsurv          0.1-5       2012-12-03 [1] CRAN (R 4.0.0)
##  knitr         * 1.30        2020-09-22 [1] CRAN (R 4.0.2)
##  labeling        0.3         2014-08-23 [1] CRAN (R 4.0.0)
##  later           1.1.0.1     2020-06-05 [1] CRAN (R 4.0.0)
##  lattice       * 0.20-41     2020-04-02 [2] CRAN (R 4.0.3)
```

```
##  latticeExtra     0.6-29      2019-12-19 [1] CRAN (R 4.0.0)
##  lifecycle        0.2.0       2020-03-06 [1] CRAN (R 4.0.0)
##  Luminescence     0.9.7       2020-01-08 [1] CRAN (R 4.0.2)
##  magrittr         1.5         2014-11-22 [1] CRAN (R 4.0.0)
##  MASS             7.3-53      2020-09-09 [2] CRAN (R 4.0.3)
##  Matrix         * 1.2-18      2019-11-27 [2] CRAN (R 4.0.3)
##  matrixStats      0.57.0      2020-09-25 [1] CRAN (R 4.0.2)
##  memoise          1.1.0       2017-04-21 [1] CRAN (R 4.0.0)
##  mgcv             1.8-33      2020-08-27 [2] CRAN (R 4.0.3)
##  mime             0.9         2020-02-04 [1] CRAN (R 4.0.0)
##  munsell          0.5.0       2018-06-12 [1] CRAN (R 4.0.0)
##  mvtnorm          1.1-1       2020-06-09 [1] CRAN (R 4.0.0)
##  nlme           * 3.1-149     2020-08-23 [1] CRAN (R 4.0.2)
##  nnet             7.3-14      2020-04-26 [2] CRAN (R 4.0.3)
##  openssl          1.4.3       2020-09-18 [1] CRAN (R 4.0.2)
##  openxlsx         4.2.2       2020-09-17 [1] CRAN (R 4.0.2)
##  patchwork      * 1.0.1       2020-06-22 [1] CRAN (R 4.0.2)
##  pcaPP            1.9-73      2018-01-14 [1] CRAN (R 4.0.0)
##  pillar           1.4.6       2020-07-10 [1] CRAN (R 4.0.2)
##  pkgbuild         1.1.0       2020-07-13 [1] CRAN (R 4.0.2)
##  pkgconfig        2.0.3       2019-09-22 [1] CRAN (R 4.0.0)
##  pkgload          1.1.0       2020-05-29 [1] CRAN (R 4.0.0)
##  plyr           * 1.8.6       2020-03-03 [1] CRAN (R 4.0.0)
##  png              0.1-7       2013-12-03 [1] CRAN (R 4.0.0)
##  prada          * 1.63.0      2019-11-08 [1] Bioconductor
##  preprocessCore   1.50.0      2020-04-27 [1] Bioconductor
##  prettyunits      1.1.1       2020-01-24 [1] CRAN (R 4.0.0)
##  processx         3.4.4       2020-09-03 [1] CRAN (R 4.0.2)
##  progress         1.2.2       2019-05-16 [1] CRAN (R 4.0.0)
##  promises         1.1.1       2020-06-09 [1] CRAN (R 4.0.0)
##  ps               1.4.0       2020-10-07 [1] CRAN (R 4.0.3)
##  purrr            0.3.4       2020-04-17 [1] CRAN (R 4.0.0)
##  R6               2.4.1       2019-11-12 [1] CRAN (R 4.0.0)
##  rappdirs         0.3.1       2016-03-28 [1] CRAN (R 4.0.2)
##  raster           3.3-13      2020-07-17 [1] CRAN (R 4.0.2)
##  RColorBrewer   * 1.1-2       2014-12-07 [1] CRAN (R 4.0.0)
##  Rcpp             1.0.5       2020-07-06 [1] CRAN (R 4.0.1)
##  RcppArmadillo    0.9.900.3.0 2020-09-03 [1] CRAN (R 4.0.2)
##  RCurl          * 1.98-1.2    2020-04-18 [1] CRAN (R 4.0.0)
##  readxl         * 1.3.1       2019-03-13 [1] CRAN (R 4.0.0)
##  remotes          2.2.0       2020-07-21 [1] CRAN (R 4.0.2)
##  reshape          0.8.8       2018-10-23 [1] CRAN (R 4.0.2)
##  rhandsontable    0.3.7       2018-11-20 [1] CRAN (R 4.0.2)
##  rio              0.5.16      2018-11-26 [1] CRAN (R 4.0.2)
##  RJSONIO        * 1.3-1.4     2020-01-15 [1] CRAN (R 4.0.0)
##  rlang            0.4.8       2020-10-08 [1] CRAN (R 4.0.3)
##  RLumShiny        0.2.2       2019-01-11 [1] CRAN (R 4.0.2)
##  rmarkdown        2.4         2020-09-30 [1] CRAN (R 4.0.2)
##  robustbase     * 0.93-6      2020-03-23 [1] CRAN (R 4.0.0)
##  rpart            4.1-15      2019-04-12 [2] CRAN (R 4.0.3)
##  rprojroot        1.3-2       2018-01-03 [1] CRAN (R 4.0.0)
##  rrcov          * 1.5-5       2020-08-03 [1] CRAN (R 4.0.2)
##  RSQLite          2.2.1       2020-09-30 [1] CRAN (R 4.0.2)
##  rstatix          0.6.0       2020-06-18 [1] CRAN (R 4.0.2)
```

```
##  rstudioapi       0.11        2020-02-07 [1] CRAN (R 4.0.0)
##  S4Vectors      * 0.26.1      2020-05-16 [1] Bioconductor
##  scales           1.1.1       2020-05-11 [1] CRAN (R 4.0.0)
##  sessioninfo      1.1.1       2018-11-05 [1] CRAN (R 4.0.0)
##  shape            1.4.5       2020-09-13 [1] CRAN (R 4.0.2)
##  shiny          * 1.5.0       2020-06-23 [1] CRAN (R 4.0.2)
##  shinydashboard   0.7.1       2018-10-17 [1] CRAN (R 4.0.2)
##  shinyIncubator * 0.2.2       2020-06-12 [1] Github (rstudio/shiny-incubator@b53511d)
##  shinysky         0.1.3       2020-07-07 [1] Github (AnalytixWare/ShinySky@695b4b9)
##  sp               1.4-4       2020-10-07 [1] CRAN (R 4.0.3)
##  SparseM        * 1.78        2019-12-13 [1] CRAN (R 4.0.0)
##  stringi          1.5.3       2020-09-09 [1] CRAN (R 4.0.2)
##  stringr        * 1.4.0       2019-02-10 [1] CRAN (R 4.0.0)
##  survival       * 3.2-7       2020-09-28 [2] CRAN (R 4.0.3)
##  survminer      * 0.4.8       2020-07-25 [1] CRAN (R 4.0.2)
##  survMisc         0.5.5       2018-07-05 [1] CRAN (R 4.0.2)
##  testthat         2.3.2       2020-03-02 [1] CRAN (R 4.0.2)
##  tibble           3.0.4       2020-10-12 [1] CRAN (R 4.0.3)
##  tidyr          * 1.1.2       2020-08-27 [1] CRAN (R 4.0.2)
##  tidyselect       1.1.0       2020-05-11 [1] CRAN (R 4.0.0)
##  topGO          * 2.40.0      2020-04-27 [1] Bioconductor
##  usethis          1.6.3       2020-09-17 [1] CRAN (R 4.0.2)
##  vctrs            0.3.4       2020-08-29 [1] CRAN (R 4.0.2)
##  withr            2.3.0       2020-09-22 [1] CRAN (R 4.0.2)
##  WriteXLS       * 6.0.0       2020-10-17 [1] CRAN (R 4.0.3)
##  xfun             0.18        2020-09-29 [1] CRAN (R 4.0.3)
##  XML            * 3.99-0.5    2020-07-23 [1] CRAN (R 4.0.2)
##  xtable           1.8-4       2019-04-21 [1] CRAN (R 4.0.0)
##  yaml             2.2.1       2020-02-01 [1] CRAN (R 4.0.0)
##  zip              2.1.1       2020-08-27 [1] CRAN (R 4.0.2)
##  zlibbioc         1.34.0      2020-04-27 [1] Bioconductor
##  zoo              1.8-8       2020-05-02 [1] CRAN (R 4.0.2)
##
## [1] C:/Users/y12c/Documents/R/win-library/4.0
## [2] C:/Program Files/R/R-4.0.3/library
```

References

Bergsagel, P Leif, W Michael Kuehl, Fenghuang Zhan, Jeffrey Sawyer, Bart Barlogie, and John Shaughnessy. 2005. "Cyclin D dysregulation: an early and unifying pathogenic event in multiple myeloma." *Blood* 106 (1): 296–303. https://doi.org/10.1182/blood-2005-01-0034.

Bødker, Julie Støve, Rasmus Froberg Brøndum, Alexander Schmitz, Anna Amanda Schönherz, Ditte Starberg Jespersen, Mads Sønderkær, Charles Vesteghem, et al. 2018. "A multiple myeloma classification system that associates normal B-cell subset phenotypes with prognosis." *Blood Advances* 2 (18): 2400–2411. https://doi.org/10.1182/bloodadvances.2018018564.

Dybkaer, K., M. Bogsted, S. Falgreen, J. S. Bodker, M. K. Kjeldsen, A. Schmitz, a. E. Bilgrau, et al. 2015. "Diffuse Large B-Cell Lymphoma Classification System That Associates Normal B-Cell Subset Phenotypes with Prognosis." *Journal of Clinical Oncology* 33 (12): 1379–88. https://doi.org/10.1200/JCO.2014.57.7080.

Falgreen, Steffen, Karen Dybkær, Ken H Young, Zijun Y Xu-Monette, Tarec C El-Galaly, Maria Bach Laursen, Julie S Bødker, et al. 2015. "Predicting response to multidrug regimens in cancer patients using

cell line experiments and regularised regression models." *BMC Cancer* 15 (1): 235. https://doi.org/10.1186/s12885-015-1237-6.

Falgreen, Steffen, Maria Bach Laursen, Julie Støve Bødker, Malene Krag Kjeldsen, Alexander Schmitz, Mette Nyegaard, Hans Erik Johnsen, Karen Dybkær, and Martin Bøgsted. 2014. "Exposure time independent summary statistics for assessment of drug dependent cell line growth inhibition." *BMC Bioinformatics* 15 (1): 168. https://doi.org/10.1186/1471-2105-15-168.

Michaelsen, Thomas Yssing, Julia Richter, Rasmus Froberg Brøndum, Wolfram Klapper, Hans Erik Johnsen, Mads Albertsen, Karen Dybkær, and Martin Bøgsted. 2018. "A B-cell-associated gene signature classification of diffuse large B-cell lymphoma by NanoString technology." *Blood Advances* 2 (13): 1542–6. https://doi.org/10.1182/bloodadvances.2018017988.