Preprocessing of the kidney data

Wolfgang Huber

June 2, 2004

Contents

1 Reading the data

The **input** of this step are the files `SampleData(Kidney2).txt` (Holgers table with patient and experiment annotation), `Table1-General.RData` an extended table with patient annotations that we curated in the course of the CGH analysis project, and 175 Arrayvision files in directory `Echip`.

The **output** of this step are the files `pat.rda`, `hybanno.rda`, and `qua.rda`.

```
> datdir = "/home/whuber/Kidney3"
> patientfile1 = "SampleData(Kidney2).txt"
> patientfile2 = "Table1-General.RData"
> hybdir = file.path(datdir, "Echip")
```

Read patient data file

```
> pat <- read.table(file.path(hybdir, patientfile1), header = TRUE,
+     skip = 4, sep = "\t", fill = FALSE, colClasses = c("numeric",
+         "character", "numeric", "numeric", "numeric", "numeric",
+         "character", "character", "character", "character", "numeric"))
> colnames(pat) = c("lfdnr", "patientid", "RNA.c.", "ţg.RNA", "ţl.RNA",
+     "ţl.H2O", "Dye", "type", "clonal.net.changes", "grading",
+     "clinical.stage")
> cat("Read", nrow(pat), "rows with", ncol(pat), "columns from",
+     patientfile1, "\n")
```

```
Read 87 rows with 11 columns from SampleData(Kidney2).txt
```

```
> pat$patientid = gsub(" ", "", pat$patientid)
```

Some corrections of typos in the E-Nr.!

```
> pat$patientid[which(pat$patientid == "98-U09256")] = "98-09256"
> pat$patientid[which(pat$patientid == "98-U04607")] = "98-04607"
> pat$patientid[which(pat$patientid == "98-U04349")] = "98-04349"
> pat$patientid[which(pat$patientid == "98-U00100")] = "98-00100"
> pat$patientid[which(pat$patientid == "97-09688")] = "97-U09688"
> rownames(pat) = pat$patientid
```

Plausibility checks and cleaning up of the coding convention for `grading`.

```
> stopifnot(all(pat$lfdnr == 1:nrow(pat)))
> stopifnot(all(!duplicated(pat$patientid)))
> stopifnot(all(pat$Dye == "Cy3"))
> stopifnot(all(sort(unique(pat$type)) == c("ccRCC", "chRCC", "pRCC")))
> if (!is.numeric(pat$grading)) {
+     pat$grading <- gsub(" ", "", pat$grading)
+     pat$grading <- gsub("G", "", pat$grading)
+     pat$grading[which(pat$grading == "1-2")] <- "2"
+     pat$grading[which(pat$grading == "2-3")] <- "3"
+     pat$grading[which(pat$grading == "")] <- NA
+     irregular.grade = which(!(pat$grading %in% c("1", "2", "3",
+         NA)))
+     cat("Irregular entries for 'grade' that were replaced by NA:",
+         pat$grading[irregular.grade], "\n")
+     pat$grading[irregular.grade] = NA
+ }
```

```
Irregular entries for 'grade' that were replaced by NA: ?
```

```
> pat$grading <- as.numeric(pat$grading)
> stopifnot(all(pat$grading %in% c(1:3) | is.na(pat$grading)))
> stopifnot(all(pat$clinical.stage %in% 1:4 | is.na(pat$clinical.stage)))
> pat$clonal.net.changes[which(pat$clonal.net.changes == "?")] = NA
> pat = cbind(pat, rep(F, nrow(pat)))
> colnames(pat)[ncol(pat)] = "in.Table1"
```

Add information from the General-table. In case of conflicting entries, those from the General-table have preference (they were less contaminated by Excel).

```
> load(file.path(datdir, patientfile2))
> x[grep("klarz", x$histo), "histo"] = "ccRCC"
> x[grep("chromo", x$histo), "histo"] = "chRCC"
> x[grep("pap", x$histo), "histo"] = "pRCC"
> xpat = character(0)
> for (p in 1:nrow(pat)) {
+     if (any(rownames(x) == rownames(pat)[p])) {
+         pat$in.Table1[p] = T
+         newrow = rownames(pat)[p]
+         stopifnot(length(newrow) == 1)
+         xpat = c(xpat, newrow)
+         if (!identical(pat$type[p], x[newrow, "histo"])) {
+             cat(rownames(pat)[p], ": subytpe:", x[newrow, "histo"],
+                 "instead of", pat$type[p], "\n")
+             pat$type[p] = x[newrow, "histo"]
+         }
+         if (!identical(pat$clinical.stage[p], x[newrow, "clinical.stage"])) {
+             cat(rownames(pat)[p], ": clinical stage:", x[newrow,
+                 "clinical.stage"], "instead of", pat$clinical.stage[p],
+                 "\n")
+             pat$clinical.stage[p] = x[newrow, "clinical.stage"]
+         }
+         if (!identical(pat$grading[p], x[newrow, "grading"])) {
+             cat(rownames(pat)[p], ": grade:", x[newrow, "grading"],
+                 "instead of", pat$grading[p], "\n")
+             pat$grading[p] = x[newrow, "grading"]
+         }
+         if (is.na(x[newrow, "histo"])) {
+             cat(rownames(pat)[p], ": type NA instead of", pat$type[p],
+                 "\n")
+             pat$type[p] = NA
+         }
+         pat$clonal.net.changes[p] = gsub(" ", "", pat$clonal.net.changes[p])
+         pat$clonal.net.changes[p] = gsub("\\\"", "", pat$clonal.net.changes[p])
+         if (!identical(pat$clonal.net.changes[p], x[newrow, "clonal.netto.changes"]) &
+             !(x[newrow, "clonal.netto.changes"] == "" & is.na(pat$clonal.net.changes[p])))
+             cat(pat$patientid[p], ": clonal net changes:", as.character(x[newrow,
+                 "clonal.netto.changes"]), "instead of", pat$clonal.net.changes[p],
+                  "\n")
+              pat$clonal.net.changes[p] = as.character(x[newrow,
+                  "clonal.netto.changes"])
+          }
+      }
+ }
```

```
99-U03541 : grade: 3 instead of 2
98-13205 : subytpe: pRCC instead of chRCC
97-08910 : clinical stage: 1 instead of 3
97-08910 : clonal net changes: hyperdiploidy,-3,-14,+18,+21 instead of pseudodiploidy,der(3
97-U08539 : clinical stage: 4 instead of 3
01-U05109 : clonal net changes: hypodiploidy,+X,-2,-3p,-3p,+3q,-5p,+7,-11q,+12q,+12q,-14q,-
01-U04984 : clinical stage: 3 instead of 1
01-U03888 : clinical stage: 4 instead of 3
00-U08004 : clinical stage: 4 instead of 3
00-U07981 : clinical stage: 1 instead of NA
00-U06423 : clinical stage: 4 instead of 3
00-U03544 : clinical stage: 4 instead of 2
00-U00786 : clinical stage: 1 instead of NA
```

Add additional variables

```
> columns = c("organ", "t", "n", "m", "progress", "rf.survival",
+     "died", "survival.time.in.months", "age", "sex", "tumor.size.cm",
+     "no.of.net.changes", "id.neu")
> pat = cbind(pat, matrix(NA, nrow = nrow(pat), ncol = length(columns)))
> colnames(pat)[(ncol(pat) - length(columns) + 1):ncol(pat)] = columns
> for (column in columns) pat[xpat, columns] = x[xpat, columns]
> colnames(pat)[which(colnames(pat) == "survival.time.in.months")] = "survival.time"
```

Translate German to English

```
> pat$sex[pat$sex == "w"] = "f"
> stopifnot(all(pat$sex[!is.na(pat$sex)] %in% c("m", "f")))
> pat$organ[pat$organ == "Niere"] = "kidney"
> pat$organ[pat$organ == "Metastase (Lunge)"] = "metastasis (lung)"
```

Chromosomal aberrations from sidelines (copy number data):

```
> tmp = matrix(NA, nrow = nrow(pat), ncol = ncol(patcnarmside))
> colnames(tmp) = colnames(patcnarmside)
> for (pt in 1:nrow(pat)) {
+     if (any(rownames(patcnarmside) == rownames(pat)[pt]))
+         tmp[pt, ] = patcnarmside[rownames(pat)[pt], ]
+ }
> pat = cbind(pat, tmp)
```

Construct hybanno table with patient names and filenames

```
> hybfiles = sort(dir(hybdir, pattern = "[0-9][0-9]*.txt$"))
> sts = strsplit(hybfiles, "_")
> hybanno = data.frame(filename = I(hybfiles), patientid = I(sapply(sts,
+     function(x) return(x[1]))), slideid = I(sub(".txt", "", sapply(sts,
+     function(x) return(x[2])))))
> hybanno$patientid[which(hybanno$patientid == "98-U09256")] = "98-09256"
> hybanno$patientid[which(hybanno$patientid == "98-U04607")] = "98-04607"
> hybanno$patientid[which(hybanno$patientid == "98-U04349")] = "98-04349"
> hybanno$patientid[which(hybanno$patientid == "98-U00100")] = "98-00100"
> hybanno$patientid[which(hybanno$patientid == "97-09688")] = "97-U09688"
> cat("table(table(hybanno$patientid))\n")

table(table(hybanno$patientid))

> print(table(table(hybanno$patientid)))

 1  2  3  4
 9 69  8  1
```

Remove the patient IDs

```
> mt <- match(hybanno$patientid, pat$patientid)
> stopifnot(!any(is.na(mt)))
> patientid <- pat[, c("patientid", "lfdnr")]
> save(patientid, file = "patientid.rda")
> pat$patientid <- paste(pat$lfdnr)
> hybanno$patientid <- pat$patientid[mt]
> save(pat, file = "pat.rda")
> save(hybanno, file = "hybanno.rda")
```

Read intensity files

```
> col.spotanno = c("Spot.labels")
> col.qua = c("VOL...Levels.x.mm2.1", "Bkgd.1", "VOL...Levels.x.mm2",
+     "Bkgd")
```

```
> col.diff = "Diff..sVOL....Ctrl...Data"
> col.numeric = c(col.qua, col.diff)
> for (h in 1:nrow(hybanno)) {
+     fn = file.path(hybdir, hybanno$filename[h])
+     dat = read.table(fn, sep = "\t", header = TRUE, skip = 1,
+         as.is = TRUE)
+     stopifnot(all(col.numeric %in% colnames(dat)))
+     if (h == 1) {
+         qua = array(NA, dim = c(nrow(dat), 4, nrow(hybanno)))
+         spotanno = dat[, col.spotanno]
+     }
+     else {
+         stopifnot(all(dat[, col.spotanno] == spotanno))
+     }
+     stopifnot(all(sapply(dat[, col.numeric], is.numeric)))
+     for (i in 1:length(col.qua)) {
+         qua[, i, h] = dat[, col.qua[i]]
+     }
+     ch1 = qua[, 1, h] - qua[, 2, h]
+     ch2 = qua[, 3, h] - qua[, 4, h]
+     ch1[ch1 < 0] = 0
+     ch2[ch2 < 0] = 0
+     stopifnot(all(abs(ch2 - ch1 - dat[, col.diff]) < 0.01))
+ }
> dimnames(qua) = list(1:8704, c("fg.green", "bg.green", "fg.red",
+     "bg.red"), 1:175)
> save(qua, file = "qua.rda")
```

2 Normalization

Normalization with vsn. Use only the ”good” hybs selected by Holger. (See also below the section on quality control. However, Holger selected the set of 74 arrays (one per patient) manually according to subjective criteria.

Three different methods were considered for background correction: ”subtbg”, ”fgonly”, and ”bgfilt”. For the following analyses, we will use ”bgfilt”, i.e. subtraction of a smoothed version of Genepix's local background.

The **input** of this step are the files `qua.rda` and `selected_chiphybs.txt`. The **output** is `ny74.bgfilt.rda`

The following is the content of the file `runvsn74.R`.

```
library(vsn)
```

```
if(!exists("qua"))      load("qua.rda")
if(!exists("spotanno")) load("spotanno.rda")
if(!exists("hybanno"))  load("hybanno.rda")
## subtbg: use background as is
if(FALSE) {
  bqua  = cbind(qua[,"fg.green",]-qua[,"bg.green",], qua[,"fg.red",]-qua[,"bg.red",])
  ny175.subtbg = vsn(bqua)
  save(ny175.subtbg, file="ny175.rda")
}

## fgonly: do not use background
if(FALSE) {
  ny175.fgonly = vsn(cbind(qua[,"fg.green",], qua[,"fg.red",]))
  save(ny175.fgonly, file="ny175.fgonly.rda")
}

## bgfilt: use running-median-filtered background
nrcol = 17
nrrow = 16
stopifnot(all(sort(unique(spotanno$Column)) == 1:nrcol))
stopifnot(all(sort(unique(spotanno$Row))    == 1:nrrow))

## physical 2d coordinates on the slide
nrbl = 4
spotanno$x = spotanno$Column + ((spotanno$Block-1)  %% nrbl) * nrcol
spotanno$y = spotanno$Row    + ((spotanno$Block-1) %/% nrbl) * nrrow

## make sure x and y make sense
tmp = read.table("Echip/00-P09206_E44-1.txt", sep='\t', header=T, skip=1, as.is=T)
graphics.off(); x11(width=7, height=7); par(mfrow=c(2,2))
plot(tmp$Pos.X...mm,  spotanno$x, pch=".")
plot(tmp$Pos.Y...mm,  spotanno$y, pch=".")
plot(tmp$Pos.X...mm2, spotanno$x, pch=".")
plot(tmp$Pos.Y...mm3, spotanno$y, pch=".")

epsilon = 1.5^2  ## neighboorhood size
nrslide = dim(qua)[3]
bqua    = matrix(NA, nrow=nrow(spotanno), ncol=2*nrslide)
for (k in 1:nrow(spotanno)) {
  nb <- (spotanno$x - spotanno$x[k])^2 + (spotanno$y - spotanno$y[k])^2 <= epsilon
  #cat(sprintf("%5d:%2d ", as.integer(k), as.integer(length(which(nb)))))
  for(h in 1:dim(qua)[3]) {
    bqua[k, h        ] = qua[k, "fg.green", h] - median(qua[nb, "bg.green", h])
    bqua[k, h+nrslide] = qua[k, "fg.red",   h] - median(qua[nb, "bg.red",   h])
  }
}

hyb.sel = rep(F, nrslide)
holger.tab = read.table("selected_chiphybs.txt", header=T, sep="\t")
hyb.sel[holger.tab[,1]] = T
hyb.sel[which(hybanno$patientid=="92-26315")] = F #unknown subtype
hyb.sel[which(hybanno$patientid=="01-U04275")] = F #is a metastasis
ny74.bgfilt = vsn(bqua[, c(which(hyb.sel), which(hyb.sel)+nrslide)])
save(ny74.bgfilt, hyb.sel, file="ny74.bgfilt.rda")
```

3 Assemble the exprSet

1. average over duplicate spots
2. build expression set

The **input** of this step are `ny74.bgfilt.rda`, `spotanno.rda`, and `pat.rda`. The **output** is `eset.rda`.

```
> library(Biobase)
> subtractred = TRUE
> make.cloneandspotanno = TRUE
> make.eset = TRUE
> load(file.path(datdir, "ny74.bgfilt.rda"))
> dat = exprs(ny74.bgfilt)
> nslides = ncol(dat)/2
> for (f in c("pat", "hybanno")) load(paste(f, "rda", sep = "."))
> nprobe = 4224
> if (make.cloneandspotanno) {
+     spotanno = read.table(file.path(hybdir, "Kidney-2-E(GM2).txt"),
+         sep = "\t", skip = 0, header = TRUE, as.is = TRUE)
+     colnames(spotanno)[9] = "vendor"
+     colnames(spotanno)[which(colnames(spotanno) == "Name")] = "description"
+     spotanno = cbind(spotanno, rep(NA, nrow(spotanno)))
+     int.index = numeric(nrow(spotanno))
+     for (spot in 1:nrow(spotanno)) int.index[spot] = (spot -
+         1)%/%(17 * 16 * 4) * (17 * 16 * 4) + (spotanno$Row[spot] -
+         1) * 17 * 4 + (spotanno$Block[spot] - 1)%%4 * 17 + spotanno$Column[spot]
+     new = numeric(nrow(spotanno))
+     new[int.index] = 1:8704
+     spotanno = spotanno[new, ]
+     colnames(spotanno)[ncol(spotanno)] = "probe"
+     cloneanno = data.frame(plate = numeric(nprobe), SrcRow = numeric(nprobe),
+         SrcCol = numeric(nprobe), imageid = numeric(nprobe),
+         AccNumber = I(character(nprobe)), spot1 = numeric(nprobe),
+         spot2 = numeric(nprobe), description = I(character(nprobe)),
+         vendor = I(character(nprobe)))
+     z = 0
+     for (j in unique(spotanno$SrcPlt)) {
+         for (k in unique(spotanno$SrcRow)) {
+             for (l in unique(spotanno$SrcCol)) {
+                 spots = which(spotanno$SrcPlt == j & spotanno$SrcRow ==
+                   k & spotanno$SrcCol == l)
+                 stopifnot(length(spots) %in% c(0, 2) | spotanno$SrcPlt[spots[1]] ==
+                   0)
+                 if (length(spots) == 2) {
+                   z = z + 1
+                   spotanno$probe[spots] = z
+                   cloneanno$plate[z] = spotanno$SrcPlt[spots[1]]
+                   cloneanno$SrcRow[z] = spotanno$SrcRow[spots[1]]
+                   cloneanno$SrcCol[z] = spotanno$SrcCol[spots[1]]
+                   cloneanno$imageid[z] = spotanno$ImageID[spots[1]]
+                   cloneanno$AccNumber[z] = spotanno$AccNumber[spots[1]]
+                   cloneanno$description[z] = spotanno$description[spots[1]]
+                   cloneanno$vendor[z] = spotanno$vendor[spots[1]]
+                   cloneanno$spot1[z] = spots[1]
+                   cloneanno$spot2[z] = spots[2]
+                 }
+             }
+         }
+     }
+     save(spotanno, file = "spotanno.rda")
+     save(cloneanno, file = "cloneanno.rda")
+ } else {
+     load("spotanno.rda")
+     load("cloneanno.rda")
+ }
> if (subtractred) {
+     kdat = dat[, 1:nslides] - dat[, (nslides + 1):ncol(dat)]
+     kdat[which(kdat > 4)] = 4
+     kdat[which(kdat < -4)] = -4
+ } else {
+     kdat = dat[, 1:nslides]
+     cat("Not subtracting R from G\n")
+ }
> kdat = (kdat[cloneanno$spot1, ] + kdat[cloneanno$spot2, ])/2
> if (make.eset) {
+     cat("using", nslides, "hybs ")
+     pat.sel = which(pat$patientid %in% hybanno$patientid[which(hyb.sel)])
+     npat = length(pat.sel)
+     cat("from", length(pat.sel), "patients\n")
+     exprs = matrix(nrow = nprobe, ncol = npat)
+     for (pt in 1:npat) {
+         hybs = which(hyb.sel & hybanno$patientid == pat$patientid[pat.sel[pt]])
+         if (length(hybs) == 1) {
+             exprs[, pt] = kdat[, which(which(hyb.sel) %in% hybs)]
+         }
+         else {
+             error("hhu")
+         }
+     }
+     rownames(pat) = pat$patientid
+     eset = new("exprSet", exprs = exprs, phenoData = new("phenoData",
+         pData = pat[pat.sel, ], varLabels = as.list(colnames(pat))))
+     save(eset, file = "eset.rda")
+ }
```

```
using 74 hybs from 74 patients
```

4 Quality Control

The **input** of this step is `ny175.rda`, the **output** are the files `subtbg/cv.rda`, `fgonly/cv.rda`, and `bgfilt/cv.rda` (for three different background correction methods).

We compute some measures for quality control.

- cv: gives a measure of correlation between any pair of hybs (MAD of the difference of Δh)

- width: the MAD of Δh for each single hyb
- mediancv: the median of the cv's with all other hybs
- cvpw: mean of cv's among replicates (but the number of hybs per pat. ranges from 1 to 4)
- intdupcor: correlation of normalized intensities of duplicate spots
- deltahdupcor: same for Δh values

The following is the content of the file `qc.R`

```
load("spotanno.rda")
load("hybanno.rda")

graphics.off()
INTERACTIVE = FALSE

## colormap
n    = 256
cols = rgb((0:n)/n,(0:n)/n, (n:0)/n)

##---------------------------------------------------
## propneg
##---------------------------------------------------
if(!exists("qua")) load("qua.rda")
propneg = propneggreen = rep(NA, dim(qua)[3])
for (j in 1:dim(qua)[3])  {
  propneg[j] = length(which(qua[,c("fg.green","fg.red"),j]
                           -qua[,c("bg.green","bg.red"),j] < 0)) / (2*dim(qua)[1])
  propneggreen[j] = length(which(qua[,"fg.green",j]
                                -qua[,"bg.green",j]        < 0)) / dim(qua)[1]
}

figw = 12
figh = 6
x11(width=figw, height=figh)
par(mfrow=c(1,2))

hist(propneg,      breaks=seq(0,1,by=0.05), main="propneg", col="blue")
hist(propneggreen, breaks=seq(0,1,by=0.05), main="propneggreen", col="darkgreen")
```

```
dev.copy(pdf, width=figw, height=figh,
         file=file.path("Figures", "propneg.pdf"))
dev.off()

##-----------------------------------------------
## Loop over the different background methods
##-----------------------------------------------
for (bgmethod in c("subtbg", "fgonly", "bgfilt")) {

infile  = paste("ny175", bgmethod, "rda", sep=".")
outfile = paste("qc",    bgmethod, "rda", sep=".")
cat("Loading", infile, "\n")
load(infile)

dat  = get(paste("ny175", bgmethod, sep="."))@h

nrslides = ncol(dat)/2

##-----------------------------------------------
## intdupcor
##-----------------------------------------------
intdupcor = intdupcorgreen = numeric(nrslides)

if (INTERACTIVE) {
  figw = 12
  figh = 6
  x11(width=figw, height=figh)
  par(mfrow=c(1,2))
}
for (slide in 1:nrslides) {
  x1g = dat[   1:4352, slide]
  x2g = dat[4353:8704, slide]
  intdupcorgreen[slide] = cor(x1g, x2g)
  x1b = c(dat[   1:4352, slide], dat[   1:4352, nrslides+slide])
  x2b = c(dat[4353:8704, slide], dat[4353:8704, nrslides+slide])
  intdupcor[slide] = cor(x1b, x2b)

  if (INTERACTIVE & min(intdupcor[slide], intdupcorgreen[slide]) < 0.7) {
    plot(x1g, x2g, pch=".",
      main=paste("hyb", slide, ", intdupcorgreen=", signif(intdupcorgreen[slide], 3)))
    lines(10*(-1:1),10*(-1:1),col="red")
    plot(x1b, x2b, pch=".",
      main=paste("hyb", slide, ", intdupcor =", signif(intdupcor[slide], 3)))
    lines(10*(-1:1),10*(-1:1),col="red")
    readLines(n=1)
  }
}

figw = 12
figh = 4
x11(width=figw, height=figh)
par(mfrow=c(1,3))

hist(intdupcor,      breaks=seq(0,1,by=0.05), col="blue",      main=paste("intdupcor",
hist(intdupcorgreen, breaks=seq(0,1,by=0.05), col="darkgreen", main=paste("intdupcorgreen",
plot(intdupcor, intdupcorgreen, main=paste("intdupcorcomp", bgmethod))
lines(0:1,0:1,col="red")

dev.copy(pdf, width=figw, height=figh,
         file=file.path("Figures", paste("intdupcor", bgmethod, "pdf", sep=".")))
dev.off()

##--------------------------------------------------
## width
##--------------------------------------------------
width = numeric(nrslides)
for (i in 1:nrslides)
  width[i] = mad(dat[,i] - dat[,i + nrslides])

wide = (width>0.715)

figw = 6
figh = 6
x11(width=figw, height=figh)

hist(width, breaks=15, main=paste("width", bgmethod), col="black")
dev.copy(pdf, width=figw, height=figh,
         file=file.path("Figures", paste("width", bgmethod, "pdf", sep=".")))
dev.off()

##--------------------------------------------------
## cv
```

```
##--------------------------------------------------
cv = cvgreen = matrix(0, nrow=nrslides, ncol=nrslides)

deltahy = dat[, 1:nrslides] - dat[, nrslides+(1:nrslides)]
for(i in 2:nrslides) {
  cat(i,"")
  hyi  = dat[,i]
  dhyi = deltahy[,i]
  for(j in 1:(i-1)) {
    cv[i,j]      = mad(dhyi - deltahy[,j])
    cvgreen[i,j] = mad( hyi - dat[, j])
  }
}
cv      = cv      + t(cv)
cvgreen = cvgreen + t(cvgreen)

##---------- cv versus width ----------
figw = 10
figh = 10/8*6
x11(width=figw, height=figh)

layout(matrix(1:3, ncol=3, nrow=1), widths=c(1,6,1), heights=1)
image(x=1, y=1:nrslides, matrix(width, nrow=1,ncol=nrslides), col=cols, main="width")
image(x=1:nrslides, y=1:nrslides, cv, col=cols, main=paste("cv", bgmethod))
   ## text(1:length(hyb.sel), 1, hyb.sel, col="red")
   ## colorbar:
image(x=1, y=seq(min(cv, na.rm=T), max(cv, na.rm=T), length=length(cols)),
               matrix(1:length(cols), nrow=1, ncol=length(cols)), col=cols, xaxt="n", xlab=

dev.copy(pdf, width=figw, height=figh,
         file=file.path("Figures", paste("cv", bgmethod, "pdf", sep=".")))
dev.off()

##---------- cvgreen vs. intdupcorgreen ----------
x11(width=figw, height=figh)

layout(matrix(1:3, ncol=3, nrow=1), widths=c(1,6,1), heights=1)
image(x=1, y=1:nrslides, matrix(intdupcorgreen, nrow=1,ncol=nrslides), col=cols, main="intd
image(x=1:nrslides, y=1:nrslides, cvgreen, col=cols, main=paste("cvgreen", bgmethod))
   ## text(1:length(hyb.sel), 1, hyb.sel, col="red")
   ## colorbar:
```

```
image(x=1, y=seq(min(cvgreen, na.rm=T), max(cvgreen, na.rm=T), length=length(cols)),
               matrix(1:length(cols), nrow=1, ncol=length(cols)), col=cols, xaxt="n", xlab=

dev.copy(pdf, width=figw, height=figh,
         file=file.path("Figures", paste("cvgreen", bgmethod, "pdf", sep=".")))
dev.off()

mediancv = mediancvgreen = numeric(nrslides)
for (i in 1:ncol(cv)) {
  mediancv[i] = median(cv[i,], na.rm=T)
  mediancvgreen[i] = median(cvgreen[i,], na.rm=T)
}

figw = 12
figh = 4
x11(width=figw, height=figh)
par(mfrow=c(1,3))

rg  = range(c(mediancv, mediancvgreen))
brk = seq(rg[1], rg[2], len=15)
hist(mediancv,      breaks=brk, main=paste("mediancv", bgmethod),      col="blue")
hist(mediancvgreen, breaks=brk, main=paste("mediancvgreen", bgmethod), col="darkgreen")
plot(mediancv, mediancvgreen, main=paste("mediancvcomp", bgmethod))

dev.copy(pdf, width=figw, height=figh,
         file=file.path("Figures", paste("mediancv", bgmethod, "pdf", sep=".")))
dev.off()

library("stepfun")
npat = length(unique(hybanno$patientid)) #$
cvpw = numeric(npat)
for (i in 1:npat) {
  hybs = which(hybanno$patientid == unique(hybanno$patientid)[i]) #$
  ## hybs = which(!wide & hybanno$patientid == unique(hybanno$patientid)[i]) #$
  cvpw[i] = mean(cv[hybs, hybs], na.rm=T)
  if (!is.na(cvpw[i]) & cvpw[i]>0.65 | any(mediancv[hybs]>0.75)) {
    cat(hybs, "\t", cvpw[i], "\t", mediancv[hybs], "\n")
  }
}

figw = 8
```

```
figh = 8
x11(width=figw, height=figh)
plot(ecdf(cvpw), main=paste("ecdf", bgmethod))
dev.copy(pdf, width=figw, height=figh,
         file=file.path("Figures", paste("ecdf", bgmethod, "pdf", sep=".")))
dev.off()

if (0) {
  x11()
  for (i in which(mediancv>0.75)) {
    plot(dat[,i]+dat[,nrslides + i], dat[,i]-dat[,nrslides + i], pch=".",
       main=paste("hyb", i, ", width(mad) =", format(width[i])))
    readLines(n=1)
  }
}

if (0) {
  x11()
  for (i in which(mediancv<0.7)) {
    plot(dat[,i]+dat[,nrslides + i], dat[,i]-dat[,nrslides + i],
        pch=".", main=paste("hyb", i, ", mad =", format(width[i]), "mediancv=", format(medi
        col=spotanno$SrcPlt, ylim=c(-4, 4)) #$
    points(dat[grep("empty", spotanno$Name),i]+dat[grep("empty", spotanno$Name),nrslides +
           dat[grep("empty", spotanno$Name),i]-dat[grep("empty", spotanno$Name),nrslides +
    points(dat[grep("Blank", spotanno$Name),i]+dat[grep("Blank", spotanno$Name),nrslides +
           dat[grep("Blank", spotanno$Name),i]-dat[grep("Blank", spotanno$Name),nrslides +
    readLines(n=1)
  }
}

save(mediancv, mediancvgreen, cv, width, cvpw, propneg, propneggreen,
     intdupcor, intdupcorgreen, file=outfile)

} ## end of for loop


test.R
==========
perform statistical tests for each gene,
estimate FDR
Results go into the directory Results2/
```

```
(newest version, after updating sample annotations in 11-03)

Output:
Results2/cctest.results.rda
Results2/subtype.results.rda
Results2/cox.results.RData
Results2/test.FDR.results*.RData

Genelists generated by the function write.clonelist:

Results2/*.html
Results2/*_resultat.txt
```