MVisAGe Vignette

Vonn Walter

Last updated August 22, 2017

1 Introduction

High-profile studies such as those conducted by The Cancer Genome Atlas (TCGA) have provided biomedical researchers unprecedented access to high-quality genomic datasets. Unfortunately, even basic analyses can be difficult for researchers without specialized bioinformatics skills. MVisAGe was designed with this audience in mind, and the package allows users to easily perform bivariate analyses involving mRNA expression and another quantitative genomic variable (e.g. DNA copy number data). Although numerous approaches have been developed for bivariate analyses of specific types of genomic data (Huang et al. (2012) and Lahti et al. (2013)), Pearson and Spearman correlation coefficients continue to be widely used, particularly for exploratory analyses. MVisAGe uses these correlation coefficients because they can be computed efficiently with matrix-based approaches. As currently implemented, hypothesis testing is not the primary function of MVisAGe, although the package does include some basic functionality for assessing the statistical significance of the Pearson and Spearman correlation coefficients it computes. Instead, MVisAGe allows users to compute and visualize the correlation coefficients on a regional or genomewide basis with the goal of assessing the effect of an underlying genomic alteration (e.g. DNA copy number change) on gene expression.

A typical MVisAGe analysis starts with two matrices of quantitative genomic data, one of which contains gene expression data (exp.mat). The second matrix (called cn.mat here) contains data from another genomic variable that may be associated with gene expression, say DNA copy number data. Although MVisAGe may be applied if cn.mat is a matrix of gene-level DNA methylation data, in this vignette we will focus exclusively on DNA copy number data. The rows of both exp.mat and cn.mat are indexed by genes, and the columns are indexed by samples. It is important to note that MVisAGe does not perform any type of preprocessing or normalization. Thus expression measurements should be quantified using RPKM, RSEM, etc., and DNA copy number measurments should be $\log_2$ ratios.

Users can apply MVisAGe to their own data or data that has been downloaded from public repositories. In this vignette we use two TCGA head and neck squamous cell carcinoma (HNSC) datasets that were downloaded from the Broad Institute's Firehose GDAC (https://gdac.broadinstitute.org/). The R data object MVisAGe.RData that is included in the package contains gene expression and DNA copy number data that was produced as follows:

- The gene expression file
  HNSC.rnaseqv2__illuminahiseq_rnaseqv2__unc_edu__Level_3__RSEM_genes_normalized__data.data.txt
  and the copy number file all_data_by_genes.txt were downloaded,

- We restricted both datasets to rows corresponding to genes in chr11 and chr12,
- We restricted both datasets to the samples in the first 100 columns.

Now we install the package and load the data.

```
> library(MVisAGe)
> data(MVisAGe)
> ls()

[1] "cn.mat"       "exp.mat"      "gene.annot"   "sample.annot"

> exp.mat[1:5, 1:5]

             TCGA.4P.AA8J.01A.11R.A39I.07 TCGA.BA.4074.01A.01R.1436.07
gene_id      "normalized_count"           "normalized_count"
A2ML1|144568 "2258.5118"                  "237.0097"
A2M|2        "8160.1109"                  "1567.7058"
AAAS|8086    "1064.8590"                  "574.6041"
AACS|65985   "1416.7309"                  "580.7702"
             TCGA.BA.4075.01A.01R.1436.07 TCGA.BA.4076.01A.01R.1436.07
gene_id      "normalized_count"           "normalized_count"
A2ML1|144568 "44.4861"                    "6187.5902"
A2M|2        "993.1288"                   "1194.8052"
AAAS|8086    "580.9996"                   "575.6264"
AACS|65985   "1157.7114"                  "2010.7569"
             TCGA.BA.4077.01B.01R.1436.07
gene_id      "normalized_count"
A2ML1|144568 "2346.0058"
A2M|2        "1963.4216"
AAAS|8086    "860.9240"
AACS|65985   "1442.2522"

> cn.mat[1:5, 1:5]

      Locus.ID    Cytoband   TCGA.4P.AA8J.01A.11D.A390.01
A2M   "        2" "12p13.31" " 0.184"
A2ML1 "   144568" "12p13.31" " 0.184"
A2MP1 "    -3926" "12p13.31" " 0.184"
AAAS  "     8086" "12q13.13" " 0.022"
AACS  "    65985" "12q24.31" " 0.022"
      TCGA.BA.4074.01A.01D.1432.01 TCGA.BA.4075.01A.01D.1432.01
A2M   " 0.001"                     "-0.063"
A2ML1 " 0.001"                     "-0.063"
A2MP1 " 0.001"                     "-0.063"
AAAS  " 0.013"                     "-0.140"
AACS  " 0.002"                     "-0.234"
```

The above output shows that exp.mat and cn.mat are matrices of character values that contain extraneous rows and columns. Moreover, the gene names are not the same in the two matrices, and neither are the sample identifiers. The section below introduces two 'helper functions' that are specifically designed to reformat TCGA data downloaded from The Broad Institute's Furehose GDAC. Users with comparable gene names and sample identifiers may choose to skip ahead to the following section.

2 Reformatting TCGA Data

The tcga.exp.convert() and tcga.cn.convert() functions can be applied to reformat exp.mat and cn.mat, respectively. exp.mat = tcga.exp.convert(exp.mat) removes the extraneous top row, shortens the TCGA barcodes in the column names to the form TCGA.XX.XXXX.XXX, rewrites the row names of the form GENE_SYMBOL|GENE_ID_number as GENE_SYMBOL, removes any rows corresponding to duplicate gene symbols, and creates a numeric matrix.

```
> exp.mat = tcga.exp.convert(exp.mat)
> exp.mat[1:5, 1:5]
```

```
         TCGA.4P.AA8J.01A TCGA.BA.4074.01A TCGA.BA.4075.01A TCGA.BA.4076.01A
A2ML1          2258.5118         237.0097          44.4861        6187.5902
A2M            8160.1109        1567.7058         993.1288        1194.8052
AAAS           1064.8590         574.6041         580.9996         575.6264
AACS           1416.7309         580.7702        1157.7114        2010.7569
AASDHPPT        518.8723        1070.5634        1872.1694         584.5468
         TCGA.BA.4077.01B
A2ML1          2346.0058
A2M            1963.4216
AAAS            860.9240
AACS           1442.2522
AASDHPPT        452.8393
```

cn.mat = tcga.cn.convert(cn.mat) removes extraneous columns, shortens TCGA barcodes in the column names to the form TCGA.XX.XXXX.XXX, and creates a numeric matrix.

```
> cn.mat = tcga.cn.convert(cn.mat)
> cn.mat[1:5, 1:5]
```

```
      TCGA.4P.AA8J.01A TCGA.BA.4074.01A TCGA.BA.4075.01A TCGA.BA.4076.01A
A2M              0.184            0.001           -0.063            0.000
A2ML1            0.184            0.001           -0.063            0.000
A2MP1            0.184            0.001           -0.063            0.000
AAAS             0.022            0.013           -0.140            0.000
AACS             0.022            0.002           -0.234           -0.032
```

```
      TCGA.BA.4077.01B
A2M             -0.708
A2ML1           -0.262
A2MP1           -0.708
AAAS            -0.011
AACS            -0.016
```

At this point both exp.mat and cn.mat are numeric matrices whose row names are gene symbols and whose column names are TCGA barcodes.

3 Preparing Data for Analysis

A gene annotation file (gene.annot) containing the chromosome number, genomic position, and cytoband information is needed in order to plot gene-level correlation coefficients. One such file is included here based on HGNC gene symbols and hg38 gene positions, but users can easily create their own gene.annot files if they have different gene identifiers or different genome builds. Gene annotation files should contain the following three columns: column 1 = chromosome number written as "chr1", column 2 = genomic position, column 3 = cytoband. The row names of gene.annot should be the same type of gene names that are used in exp.mat and cn.mat. An example is shown below:

```
> head(gene.annot)
```

```
          chr     pos         cytoband
A1BG      "chr19" " 58350152" "q13.43"
A1BG-AS1  "chr19" " 58353576" "q13.43"
A1CF      "chr10" " 50842541" "q11.23"
A2M       "chr12" "  9091834" "p13.31"
A2M-AS1   "chr12" "  9066615" "p13.31"
A2ML1     "chr12" "  8855259" "p13.31"
```

Optional sample annotation files (sample.annot) are two column files that contain categorical sample annotation data about the samples, e.g. disease status (tumor/normal) or vital status (alive/dead). The first column contains sample ids that must have the same form as the column names of exp.mat and cn.mat, while the second column contains the categorical sample annotation data. The example below shows human papillomavirus (HPV) infection status (HPV+/HPV-) and tumor site information (Oral Cavity, Hypopharynx, Larynx, Oropharynx) from the TCGA HNSC manuscript (Nature, 2015). In this vignette we will restrict attention to HPV status.

```
> head(sample.annot)
```

```
     Barcode            New.HPV.Status Site
[1,] "TCGA.BA.4074.01A" "HPV-"         "Oral Cavity"
```

```
[2,] "TCGA.BA.4076.01A" "HPV-"         "Larynx"
[3,] "TCGA.BA.4077.01A" "HPV+"         "Oropharynx"
[4,] "TCGA.BA.4078.01A" "HPV-"         "Larynx"
[5,] "TCGA.BA.5149.01A" "HPV-"         "Oral Cavity"
[6,] "TCGA.BA.5151.01A" "HPV-"         "Oral Cavity"
```

```
> sample.annot = sample.annot[,c(1, 2)]
> head(sample.annot)
```

```
     Barcode            New.HPV.Status
[1,] "TCGA.BA.4074.01A" "HPV-"
[2,] "TCGA.BA.4076.01A" "HPV-"
[3,] "TCGA.BA.4077.01A" "HPV+"
[4,] "TCGA.BA.4078.01A" "HPV-"
[5,] "TCGA.BA.5149.01A" "HPV-"
[6,] "TCGA.BA.5151.01A" "HPV-"
```

It is often the case that different genes appear in gene.annot, exp.mat, and cn.mat. Moreover, the same sample identifiers may not be present in exp.mat and cn.mat. For this reason MVisAGe includes an additional helper function that is useful when preparing data for analysis. prepped.data = data.prep(exp.mat, cn.mat, gene.annot, sample.annot, log.exp = F) produces a list containing four elements, and the names are "exp", "cn", "gene.annot", and "sample.annot". Gene names and sample ids now appear in the same order across the appropriate matrices. The default option sample.annot = NULL should be used if no sample annotation data is available. The next argument, log.exp, specifies whether the expression data has been log transformed. The default is FALSE, and in this case a $\log_2(x+1)$ transformation is applied to exp.mat. The argument gene.list is used to restrict to a set of genes of interest, e.g. genes identified by GISTIC (Beroukim et al. (2007), Mermel et al. (2011)) as having recurrent DNA copy number gains or losses. All genes are used if gene.list = NULL, the default value.

```
> prepped.data = data.prep(
+   exp.mat,
+   cn.mat,
+   gene.annot,
+   sample.annot,
+   log.exp = F,
+   gene.list = NULL)
```

```
[1] "Checking gene names"
[1] "Gene names in expression vs. CN"
[1] TRUE
[1] "Gene names in expression vs. gene annotation"
[1] TRUE
[1] "Checking sample names"
```

```
[1] "Sample names in expression vs. CN"
[1] TRUE
[1] "Sample names in expression sample annotation"
[1] TRUE
```

```
> prepped.data[["exp"]][1:5, 1:5]
```

```
         TCGA.BA.4074.01A TCGA.BA.4076.01A TCGA.BA.4078.01A TCGA.BA.5149.01A
A2M             10.615359        10.223767        12.156678        11.497423
A2ML1            7.894877        12.595395         9.854848         7.521612
AAAS             9.168933         9.171493        10.005123        10.033737
AACS             9.184306        10.974240        10.504708         9.861332
AASDHPPT        10.065501         9.193641         9.012680         8.307791
         TCGA.BA.5151.01A
A2M             12.837524
A2ML1           13.287780
AAAS             8.865353
AACS            10.649798
AASDHPPT        10.203820
```

```
> prepped.data[["cn"]][1:5, 1:5]
```

```
         TCGA.BA.4074.01A TCGA.BA.4076.01A TCGA.BA.4078.01A TCGA.BA.5149.01A
A2M                 0.001            0.000            0.063           -0.351
A2ML1               0.001            0.000            0.063           -0.351
AAAS                0.013            0.000            0.070            0.014
AACS                0.002           -0.032            0.043            0.014
AASDHPPT           -0.481           -0.806           -0.758           -0.752
         TCGA.BA.5151.01A
A2M                -0.002
A2ML1              -0.002
AAAS               -0.002
AACS               -0.003
AASDHPPT            0.000
```

```
> head(prepped.data[["gene.annot"]])
```

```
         chr  pos         cytoband
A2M      "12" "  9091834" "p13.31"
A2ML1    "12" "  8855259" "p13.31"
AAAS     "12" " 53314541" "q13.13"
AACS     "12" "125104351" "q24.31"
AASDHPPT "11" "106088128" "q22.3"
ABCB9    "12" "122946499" "q24.31"
```

```
> head(prepped.data[["sample.annot"]])
```

```
     Barcode            New.HPV.Status
[1,] "TCGA.BA.4074.01A" "HPV-"
[2,] "TCGA.BA.4076.01A" "HPV-"
[3,] "TCGA.BA.4078.01A" "HPV-"
[4,] "TCGA.BA.5149.01A" "HPV-"
[5,] "TCGA.BA.5151.01A" "HPV-"
[6,] "TCGA.BA.5152.01A" "HPV-"
```

4 Computing correlation coefficients

The corr.compute() function computes gene-level correlation coefficients based on input matrices of gene expression and DNA copy number data with matched genes and sample ids. The output of corr.compute() is an eight-column matrix whose rows are indexed by genes: column 1 = chromosome number; column 2 = genomic position (in base pairs); column 3 = cytoband; column 4 = the gene-level correlation coefficient ρ computed from the expression and copy number data (method = "pearson" computes Pearson correlation coefficients by default, whereas method = "spearman" computes Spearman correlatio coefficients); column 5 = the squared correlation coefficient; column 6 = the t statistic $\rho\sqrt{(n-2)/(1-\rho^2)}$, where n is the sample size and $n-2$ is the degrees of freedom; column 7 = the one-sided p-value corresponding to the t statistic in column 6 (the default alternative = "greater" corresponds to $\rho > 0$, which is appropriate for gene expression and DNA copy number data, whereas alternative = "less" is appropriate for gene expression and DNA methylation data); and column 8 = the Bonferroni-Hochberg FDR q-value based on the p-value in column 7. The digits parameter (digits = 5 is the default) is used in conjection with the signif() function to control the number of significant digits in columns 4 - 8. The corr.list.compute() function allows users to apply corr.compute() separately to groups of samples defined using sample annotation. The output of corr.list.compute(exp.mat, cn.mat, gene.annot, sample.annot) is a list, and its length is the number of distinct groups defined by the categorical sample annotation data (the length is 1 if sample.annot = NULL). Each list member is a eight-column matrix produced by corr.compute(). In the example below the correlation coefficients are shown for the samples in the HPV- samples in the TCGA HNSC cohort.

```
> output.list = corr.list.compute(
+   prepped.data[["exp"]],
+   prepped.data[["cn"]],
+   prepped.data[["gene.annot"]],
+   prepped.data[["sample.annot"]],
+   method = "pearson",
+   digits = 5,
+   alternative = "greater")
> names(output.list)
```

```
[1] "HPV-" "HPV+"
```

```
> head(output.list[["HPV-"]])
```

```
         chr  pos         cytoband R           R^2          tStat
A2M      "12" "  9091834" "p13.31" "0.087135"  "0.0075924"  "0.62464"
A2ML1    "12" "  8855259" "p13.31" "-0.010413" "0.00010842" "-0.074365"
AAAS     "12" " 53314541" "q13.13" "0.54434"   "0.29631"    "4.6341"
AACS     "12" "125104351" "q24.31" "0.57309"   "0.32843"    "4.9941"
AASDHPPT "11" "106088128" "q22.3"  "0.90148"   "0.81267"    "14.874"
ABCB9    "12" "122946499" "q24.31" "0.52961"   "0.28049"    "4.4589"
         pValue       qValue
A2M      "0.2675"     "0.38339"
A2ML1    "0.52949"    "0.63071"
AAAS     "1.2579e-05" "7.2015e-05"
AACS     "3.6489e-06" "2.395e-05"
AASDHPPT "0"          "0"
ABCB9    "2.2725e-05" "0.00011979"
```

5 Permutation-based statistical significance

As noted above, the output of corr.compute() and corr.list.compute() contains p- and q-values derived from the test statistics $t = \rho\sqrt{((n-2)/(1-\rho^2))}$. The perm.significance() function uses a permutation-based approach to assess significance that is similar to the method introduced by Salari et al. (2010). The significance of an observed correlation ρ_g for gene g is assessed using an empirical null distribution obtained by randomly permuting the samples for g. This is done by randomly permuting the columns of cn.mat, so the same permutation is used for each gene. The output of perm.significance() is a seven-column matrix. The first five columns are identical to the corresponding columns produced by corr.compute(), while column 6 contains the one-side permutation-based p-value for the observed value of ρ and column 7 contains the Benjamini-Hochberg q-value associated with the p-value in column 6. The arguments perm.significance are the same as the arguments of corr.compute, with the exception of num.perms, which controls the number of permutations used to compute each gene-specific null distribution (num.perms = 1e3 is the default). Like corr.list.compute(), the function perm.significance.list.compute() can be applied when multiple groups are defined by sample annotation data.

```
> perm.results = perm.significance(
+   prepped.data[["exp"]],
+   prepped.data[["cn"]],
+   prepped.data[["gene.annot"]],
+   method = "pearson",
+   digits = 5,
+   num.perms = 1e3,
```

```
+   random.seed = NULL,
+   alternative = "greater"
+   )
> head(perm.results)
```

```
         chr  pos         cytoband R           R^2          perm_pValue
A2M      "12" "  9091834" "p13.31" "0.10614"   "0.011267"   "0.233"
A2ML1    "12" "  8855259" "p13.31" "-0.010946" "0.00011981" "0.604"
AAAS     "12" " 53314541" "q13.13" "0.61651"   "0.38009"    "0.001"
AACS     "12" "125104351" "q24.31" "0.59198"   "0.35045"    "0.001"
AASDHPPT "11" "106088128" "q22.3"  "0.90622"   "0.82124"    "0.001"
ABCB9    "12" "122946499" "q24.31" "0.55573"   "0.30883"    "0.001"
         perm_qValue
A2M      "0.33224"
A2ML1    "0.69311"
AAAS     "0.0031301"
AACS     "0.0031301"
AASDHPPT "0.0031301"
ABCB9    "0.0031301"
```

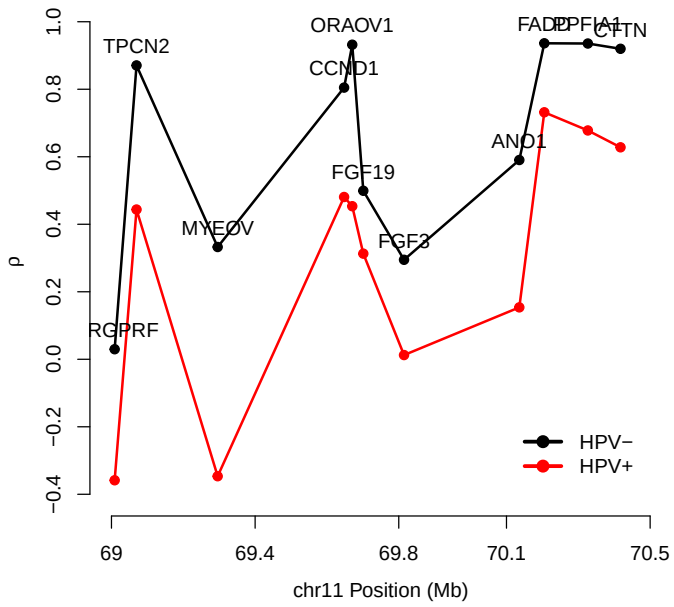
6 Plotting correlation coefficients

As noted in the introduction, MVisAGe can produce graphical output to visualize the correlation coefficients produced by corr.list.compute() in a given genomic region. Three different functions are available depending on the size of the genomic region of interest. In each case the input is a list produced by corr.list.compute(), and there are a variety of graphical parameters that can be defined by the user.

As name implies, unsmooth.region.plot() plots raw correlation coefficients in a region defined by the user. Because of the inherent noise, this function is best suited to regions containing a relatively small number of genes.

```
> unsmooth.region.plot(
+         plot.list = output.list,
+         plot.chr = 11,
+         plot.start = 69e6,
+         plot.stop = 70.5e6,
+         plot.column = "R",
+         plot.points = T,
+         plot.lines = T,
+         gene.names = T,
+         annot.colors = c("black", "red", "green", "blue", "cyan"),
+         vert.pad = .05,
+         num.ticks = 5,
+         ylim.low = NULL,
+         ylim.high = NULL,
```

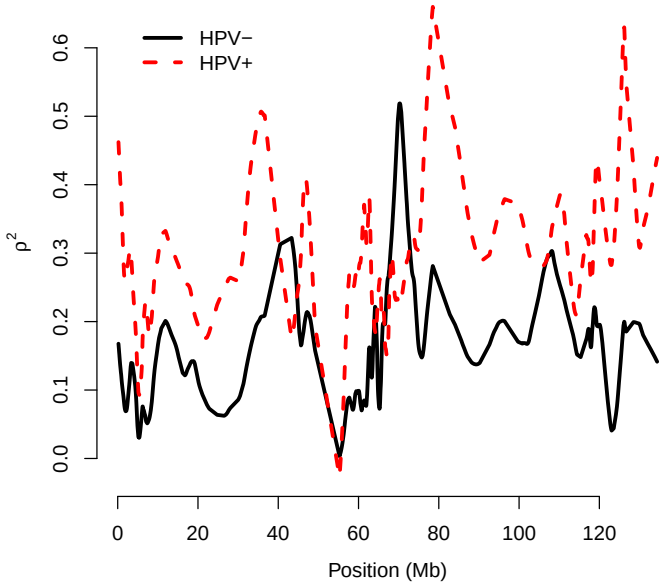
```
+         pch.vec = c(19, 19),
+         lty.vec = NULL,
+         lwd.vec = c(2, 2),
+         plot.legend = T,
+         legend.loc = "bottomright"
+         )
```



The above figure shows that HPV- samples have markedly higher Pearson correlation coefficients in chr11q13.3, a region that is frequently amplified in HNSC and contains the known driver genes *CCND1, FADD*, and *CTTN*.

The smooth.region.plot() function applies loess smoothing to the correlation coefficients so users can visualize regional trends in the correlation coefficients over larger chromosomal regions. This is done because the noise present in the gene-level correlation coefficients can make it difficult to observe regional trends that may be of interest. The level of smoothing is controlled by the loess.span parameter, which is measured in number of genes. In particular, loess.span is used to define the *span* parameter in the loess function using the formula $span = \frac{\text{loess.span}}{n+(2*\text{expand.size})}$, where n is the number of genes in the region and expand.size is another parameter (also measured in number of genes) used to lessen the effects of smoothing at the end of the region. Additional information about the *span* parameter can be found in help menu for the loess() function, while details about expand.size are available in the forthcoming manuscript with Walter et al.

```
> smooth.region.plot(
+         plot.list = output.list,
+         plot.chr = 11,
+         plot.start = 0e6,
+         plot.stop = 135e6,
+         plot.column = "R^2",
+         annot.colors = c("black", "red", "green", "blue", "cyan"),
+         vert.pad = 0,
+         ylim.low = NULL,
+         ylim.high = NULL,
+         lty.vec = c(1, 2),
+         lwd.vec = c(3, 3),
+         plot.legend = T,
+         legend.loc = "topleft",
+         loess.span = 50,
+         expand.size = 3,
+         xaxis.label = "Position (Mb)",
+         yaxis.label = expression(rho^2),
+         main.label = NULL,
+         axis.cex = 1,
+         label.cex = 1,
+         xaxis.line = 2.5,
+         yaxis.line = 2.5,
+         main.line = 0
+         )
```

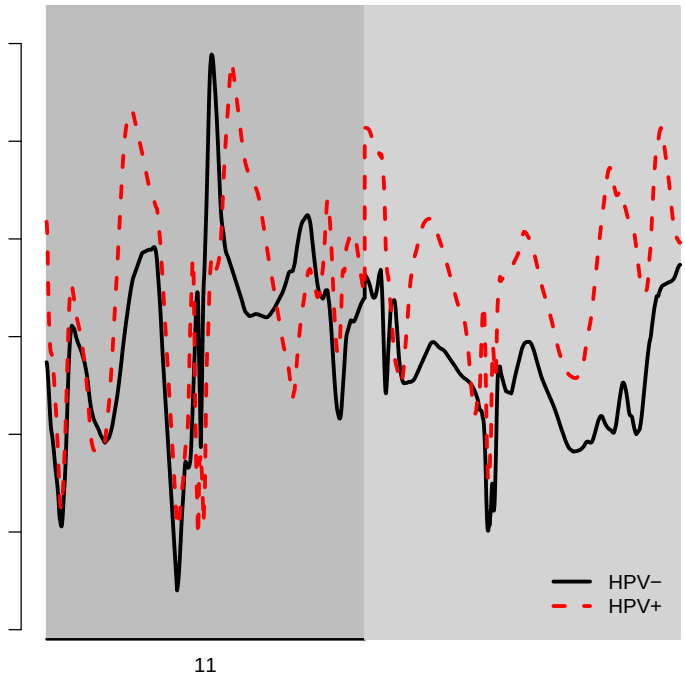


The above figure shows that HPV- samples exhibit a pronounced peak near 70Mb that is effectively absent in the HPV+ samples, which agrees with the output from unsmooth.region.plot() shown above.

Although the smooth.region.plot() function is useful for plotting smoothed correlation coefficients in a region of a given chromosome, smooth.genome.plot() should be used to make similar figures for multiple chromosomes or across the genome. The figure produced in the following example is similar to the one produced above by smooth.region.plot(), only now we show both chr11 and chr12.

```
> smooth.genome.plot(plot.list = output.list,
+         plot.column = "R",
+         annot.colors = c("black", "red", "green", "blue", "cyan"),
+         vert.pad = 0.05,
+         ylim.low = NULL,
+         ylim.high = NULL,
+         plot.legend = T,
+         legend.loc = "bottomright",
+         lwd.vec = c(3, 3),
+         lty.vec = c(1, 2),
+         loess.span = 100,
+         expand.size = 100,
+         rect.colors = c("light gray", "gray"),
```

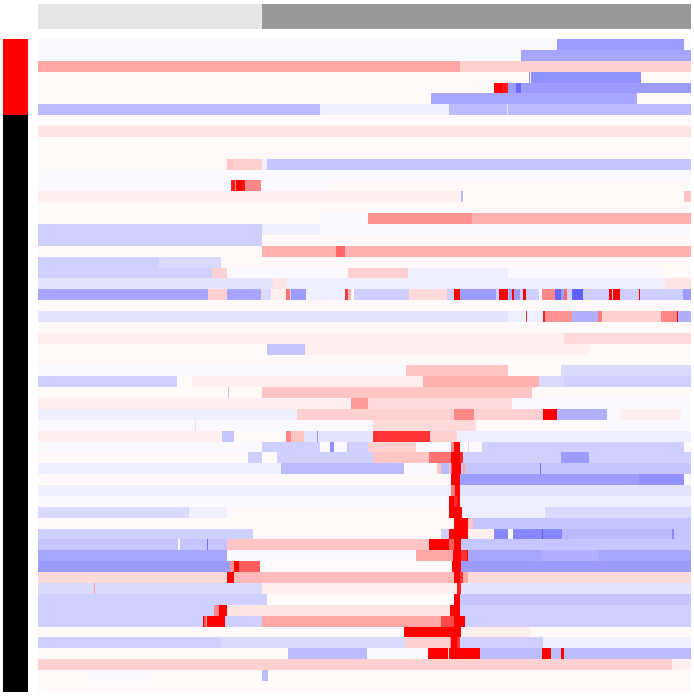
```
+         chr.label = T,
+         xaxis.label = "Chromosome",
+         yaxis.label = expression(rho),
+         main.label = NULL,
+         axis.cex = 1,
+         label.cex = 1,
+         xaxis.line = 1.5,
+         yaxis.line = 2.5,
+         main.line = 0)
```



7 Copy number heatmaps

Recurrent DNA copy number changes are of interest because they may be associated with changes in expression of regional genes. Although plots of the correlation coefficients can be used to visualize the magnitude of the association in a given region, copy number heatmaps are useful because they can be used to visualize the DNA copy number changes. Therefore the copy number heatmaps produced by cn.region.heatmap() can be used in conjunction with plots produced by smooth.region.plot() to explore copy number changes and the resulting effect on gene expression in a given chromosomal region.

```
> cn.region.heatmap(cn.mat = prepped.data[["cn"]],
+         gene.annot = prepped.data[["gene.annot"]],
+         plot.chr = 11,
+         plot.start = 0e6,
+         plot.stop = 135e6,
+         sample.annot = prepped.data[["sample.annot"]],
+         sample.cluster = T,
+         low.thresh = -2,
+         high.thresh = 2,
+         num.cols = 50,
+         collist = c("blue", "white", "red"),
+         annot.colors = c("black", "red"),
+         plot.list = output.list,
+         plot.sample.annot = T,
+         cytoband.colors = c("gray90", "gray60")
+         )
```



The above heatmap illustrates the presence of recurrent copy number gains at chr11q13.3 in the HPV- samples (black annotation bars) that are largely absent in the HPV+ samples (red annotation bars). The output from unsmooth.region.plot() and smooth.region.plot() shown above suggests that these amplifications lead to increased expression of regional genes, thereby illustrating their importance.

8 References

Huang N, Shah PK, Li C (2012). Lessons from a decade of integrating cancer copy number alterations with gene expression profiles. Brief. Bioinform. 13(3): 305 - 316.

Lahti L, Schafer M, Klein H-U, Bicciato S, Dugas M (2013). Cancer gene prioritization by integrative analysis of mRNA expression and DNA copy number data: a comparative review. Brief. Bioinform. 14(1): 27 - 35.

The Cancer Genome Atlas Research Network (2015). Comprehensive genomic characterization of head and neck squamous cell carcinomas. Nature 517(7536): 576 - 582.

Beroukhim R, Getz G, Nghiemphu L, Barretina J, Hsueh T, Linhart D, et al. (2007). Assessing the significance of chromosomal aberrations in cancer: methodology and application to glima. Proc. Nat. Acad. Sci. 104(50) 20007 - 20012.

Mermel CH, Schumacher SE, Hill B, Meyerson ML, Beroukhim R, Getz G (2011). GISTIC2.0 facilitates sensitive and confidence localization of targets of focal somatic copy-number alteration in human cancers. Genome Biol. 12:R41 doi 10.1186/gb-2011-12-4-r41.

Salari K, Tibshirani R, Pollack JR (2010). DR-Integrator: a new analytic tool for intergrating DNA copy number and gene expression data. Bioinformatics 26(3): 414 - 416.