Vignette for *DeSousa2013*: Poor prognosis colon cancer is defined by a molecularly distinct subtype and precursor lesion

Xin Wang

January 10, 2013

Contents

1 Introduction **2**

2 Package installation **2**

3 Overview **2**

4 Subtype identification **3**
4.1 Microarray data preprocessing 3
4.2 Computing GAP statistic 4
4.3 Consensus clustering 6
4.4 Collapsing probesets to unique genes 9
4.5 Filtering patient samples 9
4.6 Feature selection 10
4.7 Build PAM classifier 12

5 Subtype characterization **14**
5.1 Survival analysis 14

6 Session info **17**

7 References **18**

1 Introduction

The vignette helps the user to reproduce main results and figures using the AMC-AJCCII-90 data set to discover and characterize three molecular distinct subtypes of colon cancer. Please refer to DeSousa et al. [1] for more details about the biological background, experimental design as well as bioinformatic analyses.

2 Package installation

Please run all analyses in this vignette under version 2.15 of R. The following packages are employed by the *DeSousa2013* package: *frma*, *hgu133plus2frmavecs* (for microarray data preprocessing), *cluster* (for computing GAP statistics), *sva* (for correcting for non-biological batch effects), *ConsensusClusterPlus* (for consensus clustering), *siggenes* (for differential gene identification), *pamr* (for PAM classification), *survival* (for survival analysis and generating KM plots). These packages should be automatically installed when installing *DeSousa2013* from bioconductor:

```
> if (!requireNamespace("BiocManager", quietly=TRUE))
+     install.packages("BiocManager")
> BiocManager::install("DeSousa2013")
```

3 Overview

De Sousa et al. interrogated the heterogeneity of colon cancer using an unsupervised classification strategy over 1100 patients. The main bioinformatic pipeline is focused on how to identify three main molecularly distinct subtypes based on the microarray data for 90 stage II colon cancer patients.

To begin, we need to load the package:

```
> library(DeSousa2013)
```

All analyses included in this package are wrapped in a pipeline function *CRCPipeLine*:

```
> data(AMC)
> CRCPipeLine(celpath=".", AMC_sample_head, AMC_CRC_clinical,
```

```
+ preprocess=FALSE, gap.ntops = c(2, 4, 8, 12, 16, 20)*1000,
+ gap.K.max = 6, gap.nboot = 100, MADth=0.5, conClust.maxK=12,
+ conClust.reps=1000, diffG.pvalth=0.01, diffG.aucth=0.9,
+ savepath=".")
```

Although the pipeline function can also reproduce the preprocessing of microarray data, it is not recommended as it can take a long time and the results may change slightly due to different versions of annotation packages and analysis packages. Therefore, the user can choose to skip the preprocessing step and run the following analyses by setting the argument *preprocess* to *TRUE*. When the function finishes, all figures will be saved in a given directory *savepath*.

4 Subtype identification

Now we introduce step by step the computational workflow to perform microarray data preprocessing, computing GAP statistics, consensus clustering, sample and gene filtering, building a PAM classifier, classification, etc. using the AMC-AJCCII-90 data set. Finally, we will characterize the prognosis of three subtypes by survival analysis.

4.1 Microarray data preprocessing

The AMC-AJCCII-90 data set includes 90 stage II colon cancer patients (GSE33113), 13 adenomas and 6 normal samples. Microarrays of all samples were normalized and summarized using frozen robust multiarray analysis (fRMA)[2]. Probe-specific effects and variances are precomputed and frozen in fRMA, which facilitates batches analysis. Gene expression presence/absence was detected using the barcode algorithm[3] and genes that were not present in at least one sample were filtered out. Non-biological batch effects between cancer samples + normals and adenomas were corrected using *ComBat* [4].

The whole preprocessing analysis can be reproduced by function *geneExpPre*:

```
> data(AMC)
> ge.pre <- geneExpPre(celpath, AMC_sample_head)
> ge.all <- ge.pre[["ge.all"]]
```

```
> selPbs <- ge.pre[["selPbs"]]
> ge.CRC <- ge.all[selPbs, ]
```

where *celpath* is the path to the directory including '.CEL' files of all microarrays and *AMC_sample_head* is a data frame including mapping information between microarray ids and sample ids. *AMC_sample_head* is included in this package and can be loaded using function *data*.

Since we focus on subtype identification in this vignette, the preprocessed data *ge.all* returned by function *geneExpPre* only contains the 90 cancer samples, and `selPbs` is a vector of probesets that are expressed in at least one sample:

```
> length(selPbs)

[1] 37007

> dim(ge.CRC)

[1] 37007    90
```

4.2 Computing GAP statistic

GAP statistic[5] is a popular method to estimate the number of clusters in a set of data by comparing the change in observed and expected within-cluster dispersion. To identify the optimal number of clusters, GAP statistic was computed for k=1 to 6 for selected top variable genes. The function *compGapStats* can be used to reproduce this step:

```
> gaps <- compGapStats(ge.CRC, ntops=c(2, 4, 8, 12, 16, 20)*1000,
+ K.max=6, nboot=100)
> gapsmat <- gaps[["gapsmat"]]
> gapsSE <- gaps[["gapsSE"]]
```

The function *figGAP* can be used to compare the GAP scores across different numbers of clusters:

```
> figGAP(gapsmat, gapsSE)
```

As shown in Figure 1, a peak was consistently found at k=3, irrespective of the gene set size employed, indicating that three subtypes is ideal to explain the inherent data structure of the AMC-AJCCII-90 set.

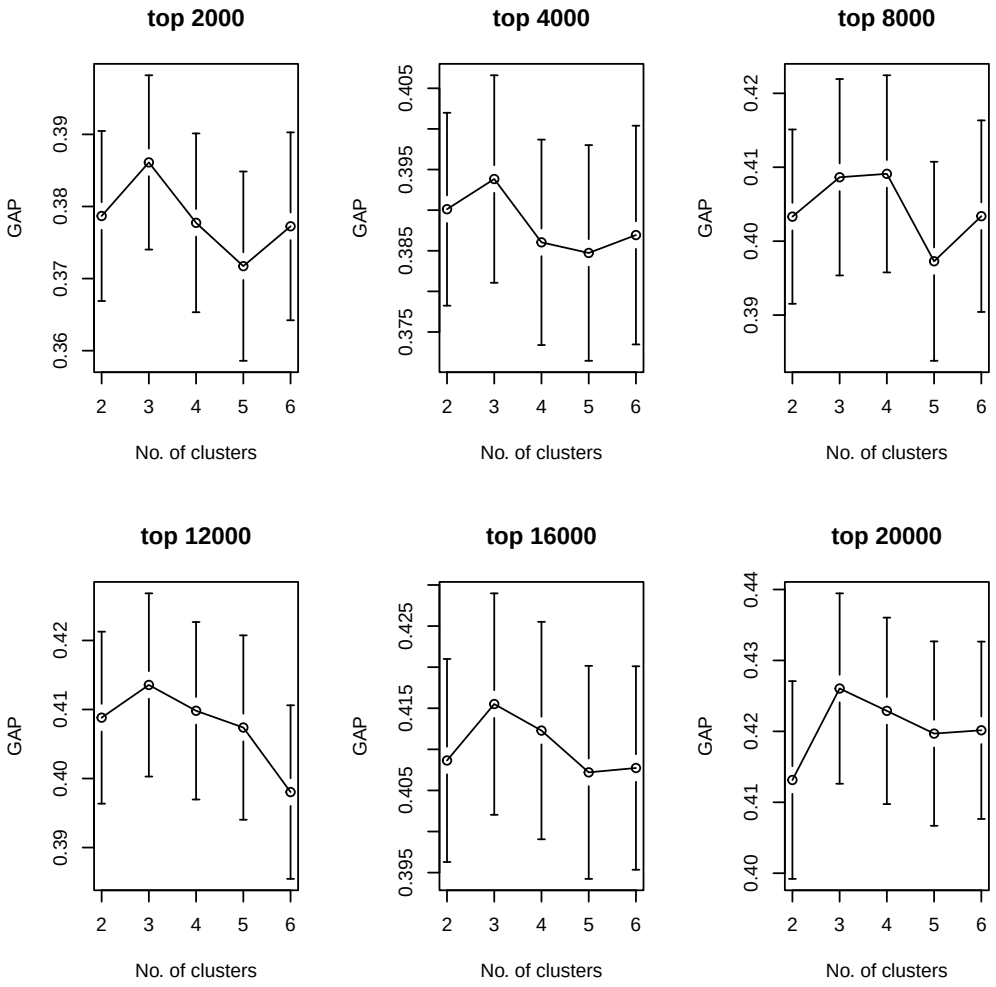


Figure 1: **Graph depicts the GAP statistic for a range of 2000-20000 probesets.** This analysis suggests optimal clustering at 3 clusters in the AMC-AJCCII-90 set, independent on the number of probesets used. Error bars indicate SEM.

4.3 Consensus clustering

Using the following function, probesets with most variability (Median absolute deviation >0.5) across samples were retained and median centred:

```
> sdat <- selTopVarGenes(ge.CRC, MADth=0.5)
```

```
> dim(sdat)
```

```
[1] 7846   90
```

Using the 7846 most variable probesets, we performed hierarchical clustering with agglomerative average linkage to cluster these samples. Consensus clustering[6] was employed, with 1000 iterations and 0.98 subsampling ratio, to assess the clustering stability. The function *conClust* reproduces this step:

```
> clus <- conClust(sdat, maxK=12, reps=1000)
```

```
> clus
```

```
col004 col003 col008 col009 col012 col013 col016 col020 col021 col024
     2      2      2      2      3      2      2      2      2      2
col026 col035 col036 col040 col052 col053 col054 col058 col059 col060
     3      2      3      2      3      3      3      2      3      2
col061 col064 col066 col069 col073 col075 col079 col081 col082 col084
     3      3      3      3      3      3      3      2      3      2
col085 col091 col093 col096 col099 col100 col045 col037 col014 col062
     2      2      2      2      2      2      3      3      3      3
col072 col092 col098 col050 col080 col034 col086 col010 col068 col063
     3      3      3      3      1      1      1      1      1      1
col002 col001 col005 col006 col007 col011 col015 col017 col018 col019
     1      1      1      1      1      1      1      1      1      1
col022 col023 col033 col038 col039 col046 col047 col048 col049 col051
     1      1      1      1      1      1      1      1      1      1
col055 col057 col065 col070 col074 col076 col078 col083 col090 col094
     1      1      1      1      1      1      1      1      1      1
col095 col097 col101 col102 col056 col041 col077 col089 col067 col044
     1      1      1      1      1      1      1      1      1      1
```

Figure 2 shows the consensus clustering results for k=2, 3, 4 and 5. As indicated in the cumulative density curves (Figure 3), a significant increase in clustering stability was observed from k=2 to 3, but not for k>3.

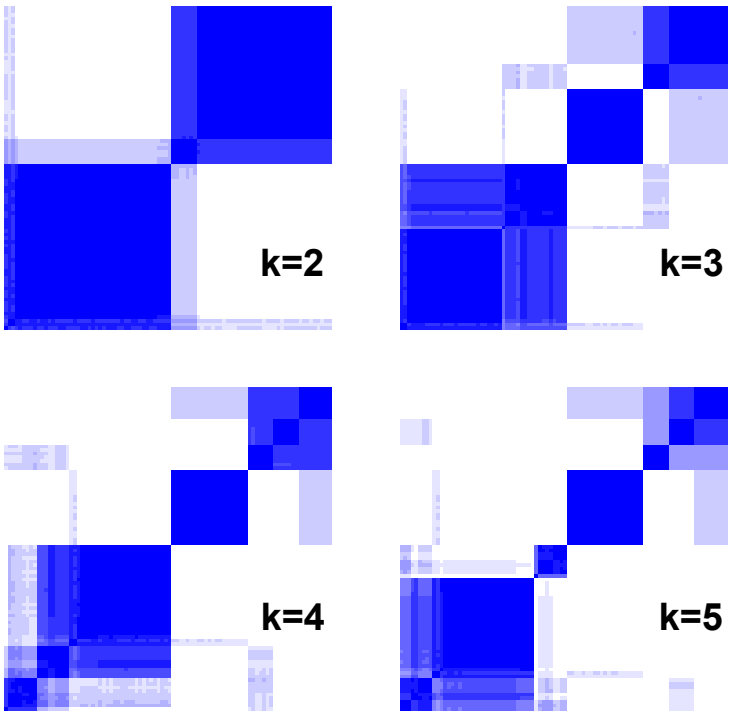


Figure 2: **Consensus clustering for 2, 3, 4 and 5 clusters.** The color is scaled to the frequency that two samples are in the same cluster.

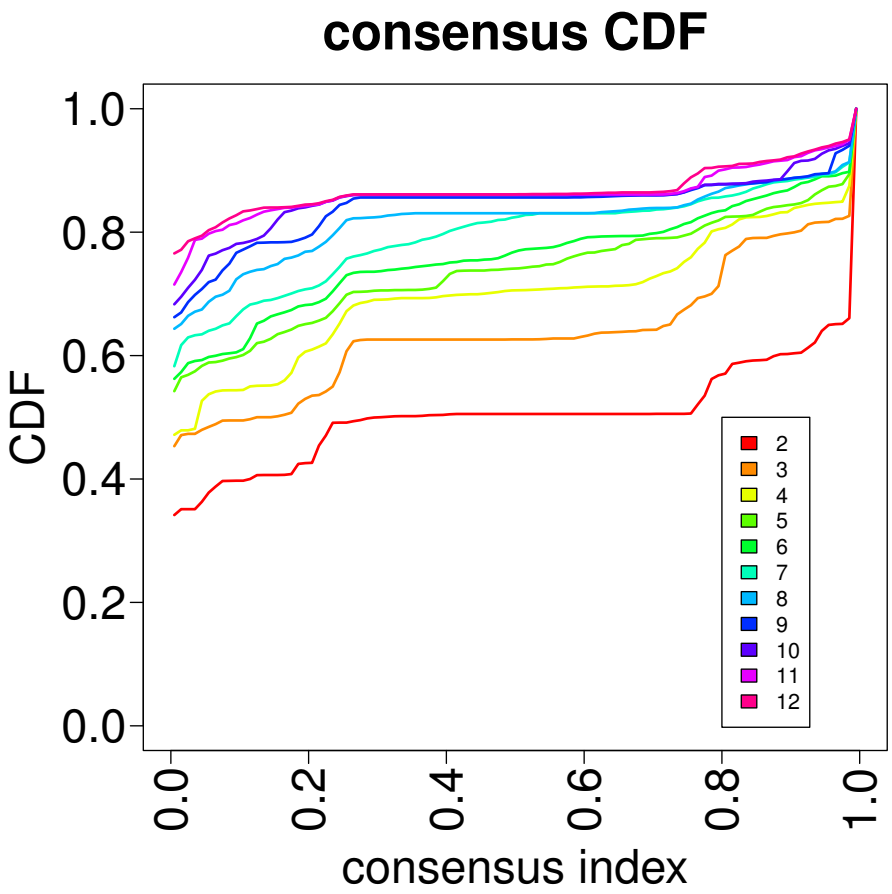


Figure 3: **Empirical cumulative distribution (CDF) of consensus clustering for 2 to 12 clusters.**

4.4 Collapsing probesets to unique genes

To facilitate the use of the classifier on other platforms, we collapsed the expression levels for probesets to genes. This was performed before selection of most representative genes (SAM and AUC) and generation of the classifier (PAM), for each gene the probeset with highest overall expression was selected. The following function can reproduce the annotation file to map from probesets to unique genes:

```
> uniGenes <- pbs2unigenes(ge.CRC, sdat)

> length(uniGenes)

[1] 4762

> print(uniGenes[1:20])

   213418_at 1555759_a_at    204600_at 1552256_a_at   1552281_at
     "HSPA6"       "CCL5"      "EPHB3"     "SCARB1"    "SLC39A5"
 221646_s_at 1552309_a_at 1552312_a_at 1552316_a_at    227444_at
   "ZDHHC11"       "NEXN"      "MFAP3"     "GIMAP1"     "ARMCX4"
 218250_s_at  225240_s_at   1552365_at   1552370_at    212805_at
     "CNOT7"       "MSI2"       "SCIN"    "C4orf33"     "PRUNE2"
1552470_a_at 1552477_a_at    213050_at 1554897_s_at    222379_at
    "ABHD11"       "IRF6"       "COBL"     "RHBDL2"      "KCNE4"
```

4.5 Filtering patient samples

Subsequently Silhouette width[7] was computed to identify the most representative samples within each cluster. Samples with positive silhouette width (n=85) were retained to build the classifier (Figure 4). The function *filterSamples* computes Silhouette width, based on which to select most representative samples:

```
> samp.f <- filterSamples(sdat, uniGenes, clus)
> silh <- samp.f[["silh"]]
> sdat.f <- samp.f[["sdat.f"]]
> clus.f <- samp.f[["clus.f"]]

> dim(sdat.f)
```

```
[1] 4762   85
```

```
> rownames(sdat.f)[1:20]
```

```
   213418_at 1555759_a_at    204600_at 1552256_a_at   1552281_at
     "HSPA6"       "CCL5"      "EPHB3"     "SCARB1"    "SLC39A5"
 221646_s_at 1552309_a_at 1552312_a_at 1552316_a_at    227444_at
   "ZDHHC11"       "NEXN"      "MFAP3"     "GIMAP1"     "ARMCX4"
 218250_s_at  225240_s_at   1552365_at   1552370_at    212805_at
     "CNOT7"       "MSI2"       "SCIN"    "C4orf33"     "PRUNE2"
1552470_a_at 1552477_a_at    213050_at 1554897_s_at    222379_at
    "ABHD11"       "IRF6"       "COBL"     "RHBDL2"      "KCNE4"
```

As we see in this filtering step, probesets have been converted to genes. The function *figSilh* generates the Silhouette width figure:

```
> figSilh(silh)
```

4.6 Feature selection

To build the CCS classifier, we applied two filtering steps to select the most representative and predictive genes. First, we used Significance Analysis of Microarrays (SAM)[8] (R package *siggenes*) to identify genes significantly differentially expressed (FDR$<$0.01) between each subtype and the other two. The function *findDiffGenes* performs SAM analysis to search for top differential genes between three subtypes:

```
> diffGenes <- findDiffGenes(sdat.f, clus.f, pvalth=0.01)
```

```
> length(diffGenes)
```

```
[1] 2747
```

As we see, 2747 unique genes are highly differentially expressed between subtypes.

Second, we calculated AUC (area under ROC curve, R package *ROCR*) to assess each gene's ability to separate one subtype from the other two. The function *filterDiffGenes* filters differential genes based on AUC scores:

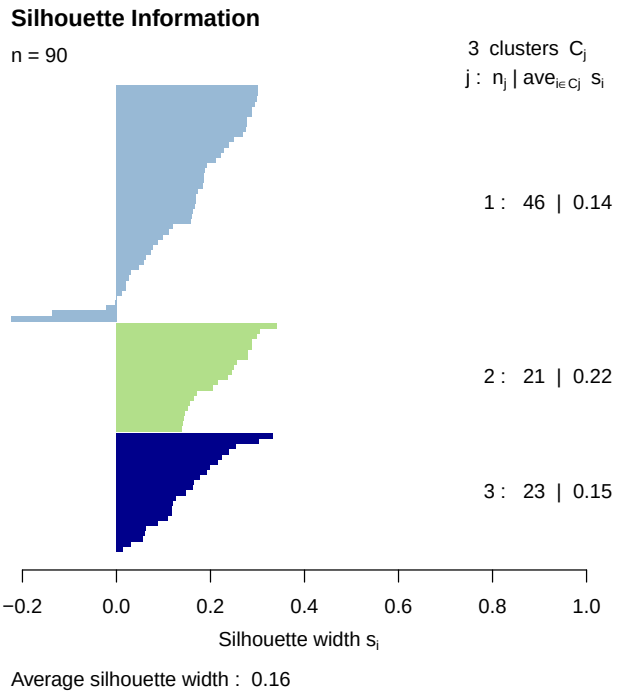


Figure 4: **Silhouette widths of colon cancer samples.** The figure shows Silhouette widths of samples in each cluster. Samples with positive Silhouette values were selected as core samples to build the classifier.

```
> diffGenes.f <- filterDiffGenes(sdat.f, clus.f, diffGenes, aucth=0.9)

> length(diffGenes.f)

[1] 323
```

After this step, we retained 323 unique genes that are most predictive.

4.7 Build PAM classifier

The retained 329 genes with AUC $>$ 0.9 were trained by PAM[9] to build a robust classifier. To select the optimal threshold for centroid shrinkage, we performed 10-fold cross-validation over a range of thresholds for 1000 iterations, and selected the one yielding a good performance (error rate $<$ 2%) with the least number of genes (Figure 5). Of note, the gene filtering steps do not have any significant influence on the selection of signature genes, as observed from PAM classification using various cut-offs on SAM FDR and AUC (data not shown).

The function *buildClassifier* builds the PAM classifier:

```
> sigMat <- sdat.f[diffGenes.f, names(clus.f)]
> classifier <- buildClassifier(sigMat, clus.f, nfold=10, nboot=100)
> signature <- classifier[["signature"]]
> pam.rslt <- classifier[["pam.rslt"]]
> thresh <- classifier[["thresh"]]
> err <- classifier[["err"]]
```

The function *figPAMCV* reproduces the PAM cross validation figure:

```
> figPAMCV(err)
```

Using this strategy, we built a classifier of 146 unique genes and used it to classify the CRC samples (Figure 6).

The classifier is then applied to classify the 90 colon cancer samples. A posterior probability $>$0.5 for one of the subtypes was regarded as being indicative of association with that group. The following functions are used to reproduce the classification and figure:

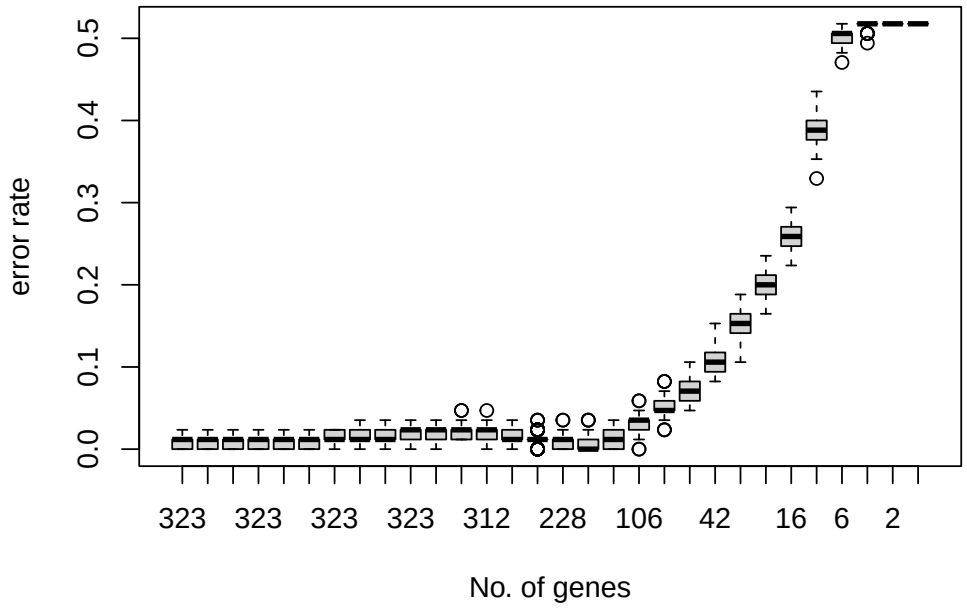


Figure 5: **PAM classification with cross validations.** The figure shows the cross validation error rate as a function of level of PAM shrinkage (number of genes left).

```
> datsel <- sdat[names(uniGenes), ]
> rownames(datsel) <- uniGenes
> datsel <- datsel[diffGenes.f, ]
> pamcl <- pamClassify(datsel, signature, pam.rslt, thresh, postRth=1)
> sdat.sig <- pamcl[["sdat.sig"]]
> pred <- pamcl[["pred"]]
> clu.pred <- pamcl[["clu.pred"]]
> nam.ord <- pamcl[["nam.ord"]]
> gclu.f <- pamcl[["gclu.f"]]
```

```
> figClassify(AMC_CRC_clinical, pred, clu.pred, sdat.sig, gclu.f, nam.ord)
```

5 Subtype characterization

5.1 Survival analysis

The clinical relevance of this classification becomes evident when analysing the disease-free survival, which revealed a significantly poorer prognosis for CCS3 patients compared to CCS1-CIN and CCS2-MSI patients (Figure 7).

To reproduce the progression free survival analysis, we developed function *progAMC*:

```
> prog <- progAMC(AMC_CRC_clinical, AMC_sample_head, clu.pred)
```

```
Call:
survdiff(formula = Surv(time, status) ~ x, data = data4surv)

     N Observed Expected (O-E)^2/E (O-E)^2/V
x=1 44        3    10.22     5.104    11.107
x=2 22        4     4.83     0.143     0.191
x=3 24       12     3.95    16.436    21.049

 Chisq= 22  on 2 degrees of freedom, p= 2e-05
```

```
> surv <- prog[["surv"]]
> survstats <- prog[["survstats"]]
```

The function *figKM* can be used to reproduce the KM plot:

```
> figKM(surv, survstats)
```

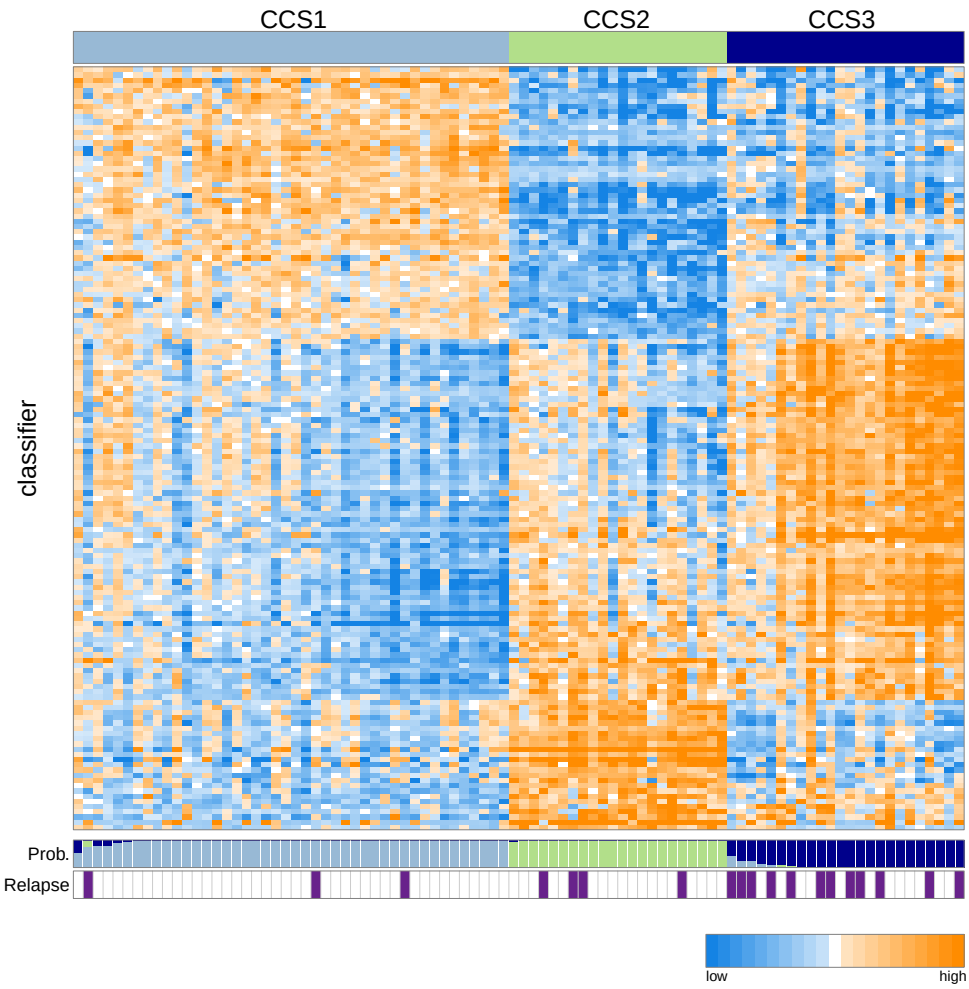


Figure 6: **The AMC-AJCCII-90 set is classified in three subtypes according to the classifier.** The top bar indicates the subtypes; light blue; CCS1, green; CCS2, dark blue; CCS3. In the heatmap, rows indicate genes from the classifier and columns represent patients. The heatmap is color-coded based on median centred log2 gene expression levels (orange, high expression; blue, low expression). The lower bar indicates the posterior probability of belonging to each respective subtype.

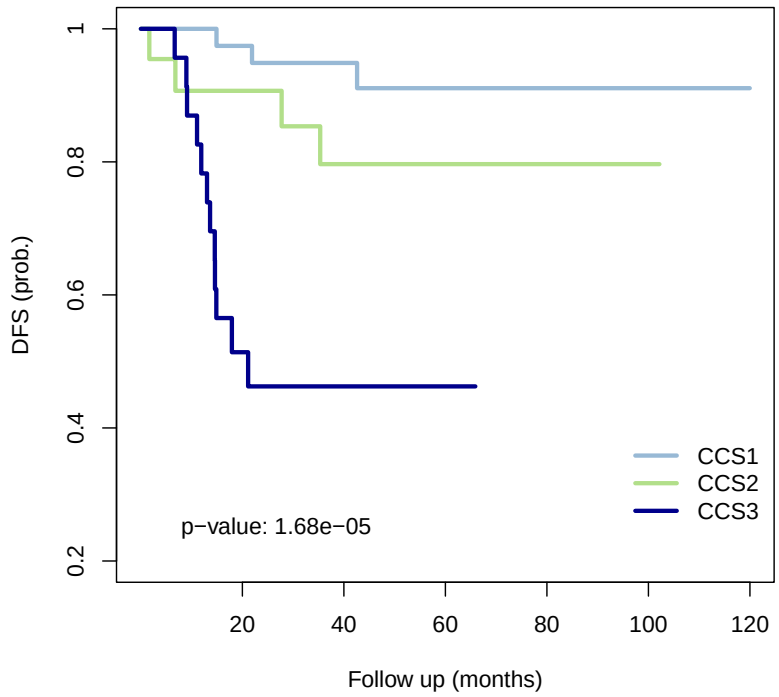


Figure 7: **Kaplan-Meier graphs depicting Disease-free survival (DFS) within the AMC-AJCCII-90 set stratified by the CCS classification.**

6 Session info

This document was produced using:

```
> toLatex(sessionInfo())
```

- R version 4.6.1 (2026-06-24), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=en_US.UTF-8`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Time zone: `Etc/UTC`
- TZcode source: `system (glibc)`
- Running under: `Ubuntu 26.04.1 LTS`
- Matrix products: default
- BLAS: `/usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3`
- LAPACK: `/usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.32.so` ; LAPACK version3.12.0
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: DeSousa2013 1.49.0
- Loaded via a namespace (and not attached): abind 1.4-8, affxparser 1.85.1, affy 1.91.0, affyio 1.83.0, annotate 1.91.0, AnnotationDbi 1.75.2, Biobase 2.73.2, BiocBaseUtils 1.15.1, BiocGenerics 0.59.12, BiocManager 1.30.27, BiocParallel 1.47.0, Biostrings 2.81.9, bit 4.6.0, bit64 4.8.6, bitops 1.1-0, blob 1.3.0, buildtools 1.0.0, cachem 1.1.0, caTools 1.18.4, cli 3.6.6, cluster 2.1.8.3,

codetools 0.2-20, compiler 4.6.1, ConsensusClusterPlus 1.77.0, crayon 1.5.3, DBI 1.3.0, DelayedArray 0.39.8, edgeR 4.99.10, evaluate 1.0.5, fastmap 1.2.0, ff 4.5.3, foreach 1.5.2, frma 1.65.0, genefilter 1.95.0, generics 0.1.4, GenomicRanges 1.65.4, gplots 3.3.0, grid 4.6.1, gtools 3.9.5, hgu133plus2.db 3.13.0, httr 1.4.9, IRanges 2.47.5, iterators 1.0.14, KEGGREST 1.53.6, KernSmooth 2.23-27, knitr 1.52, lattice 0.23-1, limma 3.99.2, locfit 1.5-9.12, maketools 1.3.2, MASS 7.3-66, Matrix 1.7-6, MatrixGenerics 1.25.0, matrixStats 1.5.0, memoise 2.0.1, mgcv 1.9-4, multtest 2.69.0, nlme 3.1-171, oligo 1.77.0, oligoClasses 1.75.0, pamr 1.57, parallel 4.6.1, pkgconfig 2.0.3, png 0.1-9, preprocessCore 1.75.1, R6 2.6.1, rlang 1.3.0, ROCR 1.0-12, RSQLite 3.53.3, S4Arrays 1.13.2, S4Vectors 0.51.10, scrime 1.3.7, Seqinfo 1.3.2, siggenes 1.87.0, SparseArray 1.13.4, splines 4.6.1, statmod 1.5.2, stats4 4.6.1, SummarizedExperiment 1.43.0, survival 3.8-12, sva 3.61.0, sys 3.4.3, tools 4.6.1, vctrs 0.7.3, xfun 0.62, XML 3.99-0.25, xtable 1.8-8, XVector 0.53.0

7 References

[1] De Sousa E Melo, F. and Wang, X. and Jansen, M. et al. Poor prognosis colon cancer is defined by a molecularly distinct subtype and precursor lesion. *accepted* 2

[2] McCall, Matthew N and Bolstad, Benjamin M and Irizarry, Rafael A (2010). Frozen robust multiarray analysis (fRMA). *Biostatistics*, **11**(2), 242–253. 3

[3] McCall, Matthew N and Uppal, Karan and Jaffee, Harris A and Zilliox, Michael J and Irizarry, Rafael A (2011). The Gene Expression Barcode: leveraging public data repositories to begin cataloging the human and murine transcriptomes. *Nucleic acids research*, **39**(suppl 1), D1011–D1015. 3

[4] Johnson, W Evan and Li, Cheng and Rabinovic, Ariel (2007). Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, **8**(1), 118–127. 3

[5] Tibshirani, Robert and Walther, Guenther and Hastie, Trevor (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **63**(2), 411–423. 4

[6] Monti, Stefano and Tamayo, Pablo and Mesirov, Jill and Golub, Todd (2003). Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine learning*, **52**(1), 91–118. 6

[7] Rousseeuw, Peter J (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis *Journal of computational and applied mathematics*, **20**, 53–65. 9

[8] Tusher, Virginia Goss and Tibshirani, Robert and Chu, Gilbert (2001). Significance analysis of microarrays applied to the ionizing radiation response. *PNAS*, **98**(9), 5116–5121. 10

[9] Tibshirani, Robert and Hastie, Trevor and Narasimhan, Balasubramanian and Chu, Gilbert (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression. *PNAS*, **99**(10), 6567–6572. 12