Cancer Analysis on NCI60 Data

Statistic Final Project By- Sagar Jain (sj735) & Aayush Mandhyan (am2447)

In [515]:
```
#installing Packages
install.packages("ggdendro")
install.packages("ISLR")
library(ISLR)

library(ISLR)
library(ggplot2)
library(ggdendro)

library(grid)
library(gridExtra)
library(lattice)
```

T test and False Discovery rate Functions

In [467]:
```
mytfunc <- function(x)
{
    xbar <- mean(x)
    sd0 <- sd(x)
    n <- length(x)
    tstat <- xbar/(sd0/sqrt(n))
    p0 <- pt(tstat, n-1)
    c(p0, 1-p0)
}
```

In [468]:
```
fdr <- function(v1, Q)
{
    o1 <- order(v1)
    pvec <- v1[o1]
    m <- length(v1)
    qline <- Q * c(1:m)/m
    plot(c(c(1:m), c(1:m)), c(qline, pvec), type='n', xlab='ordering', ylab='
    lines(c(1:m), qline)
    points(c(1:m), pvec)
    dv <- pvec - qline
    I1 <- (dv < 0)
    pmax <- max(pvec[I1], na.rm = T)
    I2 <- pvec <= pmax
    points(c(1:m)[I2], pvec[I2], col='red')
    d <- o1[I2]
    d <- d[!is.na(d)]
    return(d)
}
```

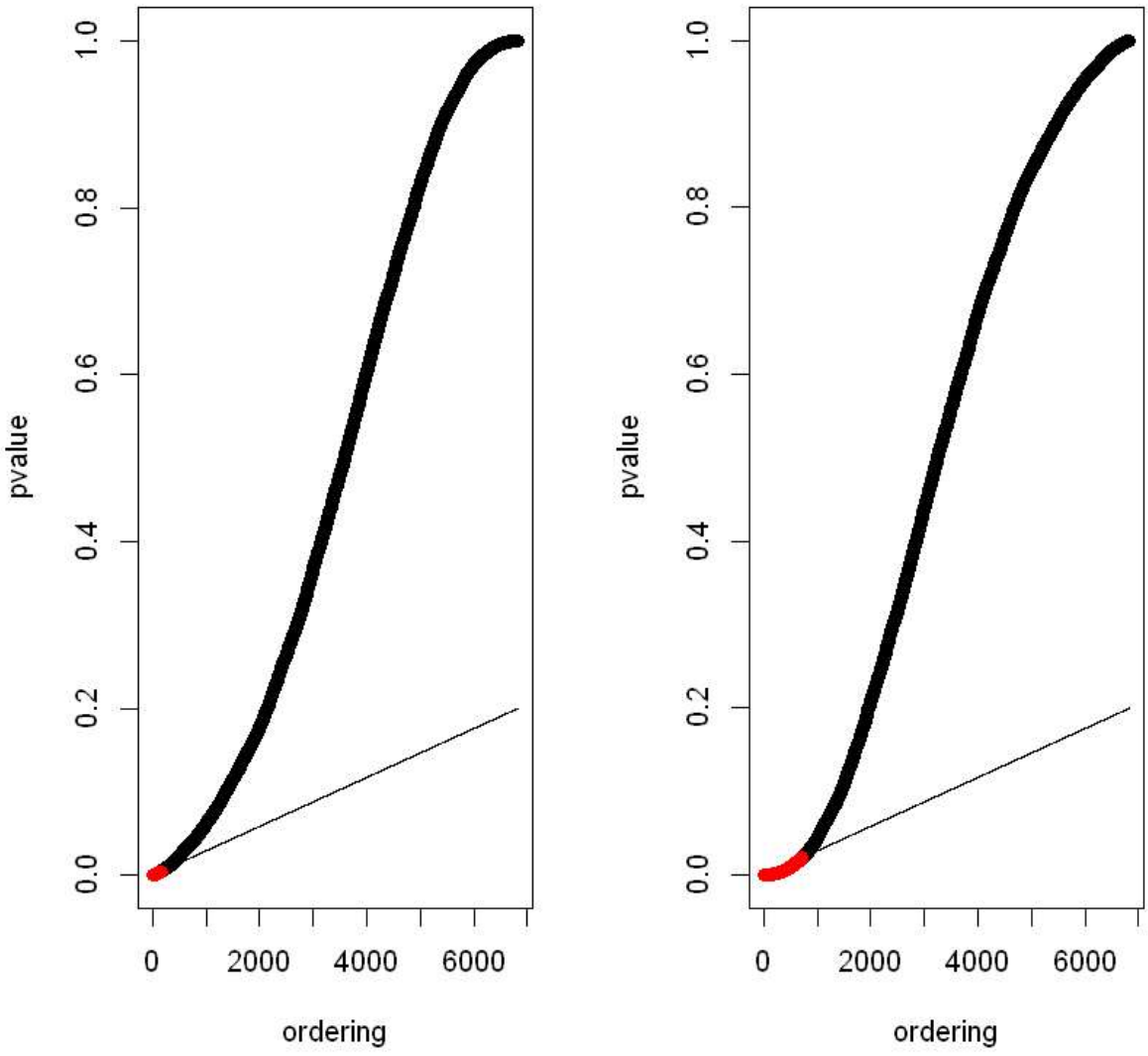
In [469]:
```
1 nci_labs <- NCI60$labs
2 nci_data <- NCI60$data
3 table(nci_labs)
```

```
nci_labs
     BREAST         CNS       COLON K562A-repro K562B-repro    LEUKEMIA 
          7           5           7           1           1           6 
MCF7A-repro MCF7D-repro    MELANOMA       NSCLC     OVARIAN    PROSTATE 
          1           1           8           9           6           2 
      RENAL     UNKNOWN 
          9           1 
```

Cancer Type : Renal

In [470]:

```
I1 <- NCI60$labs == 'RENAL'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'RENAL',]

par(mfrow = c(1,2))

#overexpress
overexpress_renal <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)



#underexpress
underexpress_renal <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)


#Prenal <- apply(m1, 2, mytfunc)

```

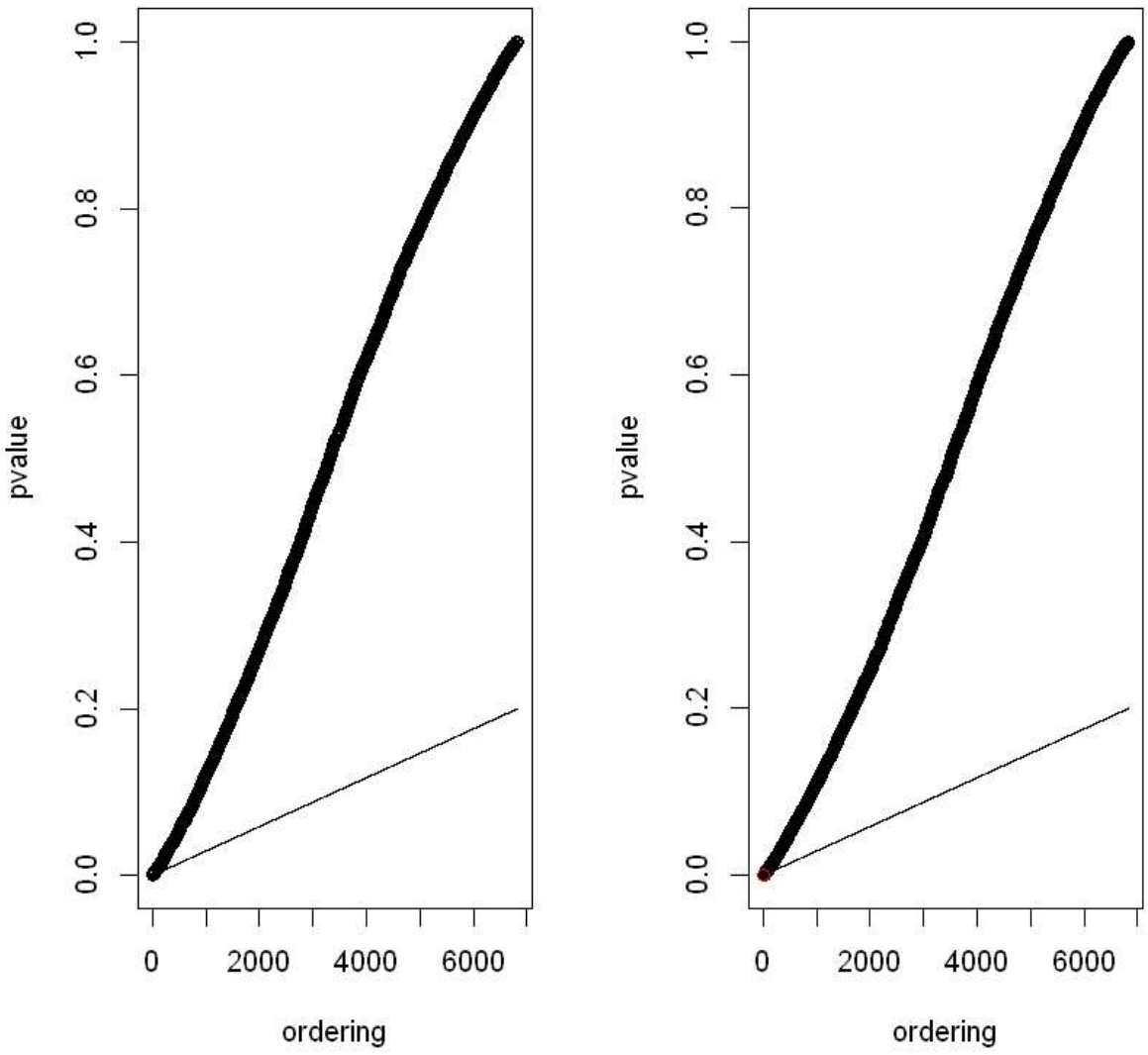


Cancer Type : BREAST

In [471]:

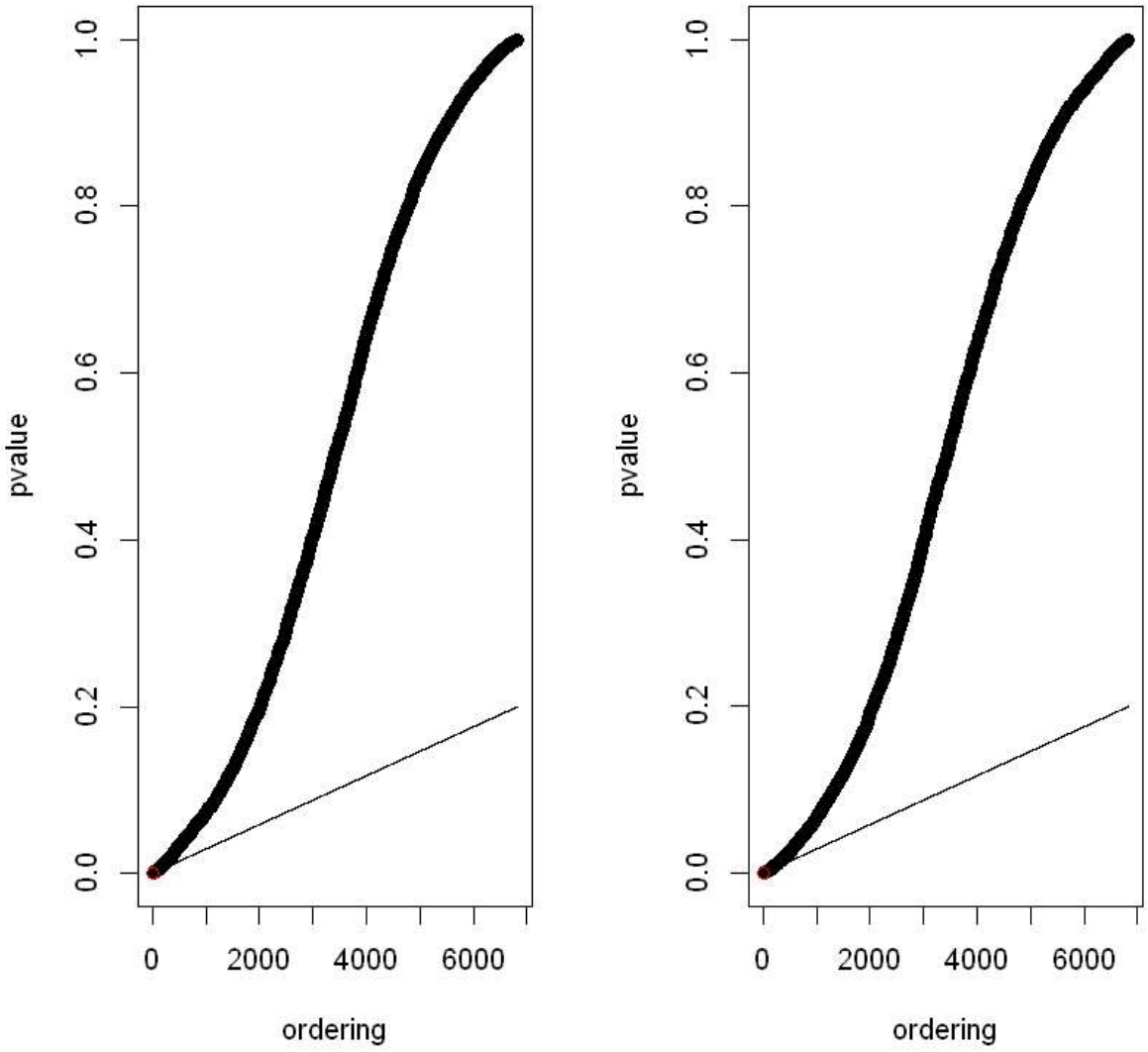
```
I1 <- NCI60$labs == 'BREAST'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'BREAST',]

par(mfrow = c(1,2))

#overexpress
overexpress_breast <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_breast <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)

#Prenal <- apply(m1, 2, mytfunc)

```

```
Warning message in max(pvec[I1], na.rm = T):
"no non-missing arguments to max; returning -Inf"
```



Cancer Type : CNS

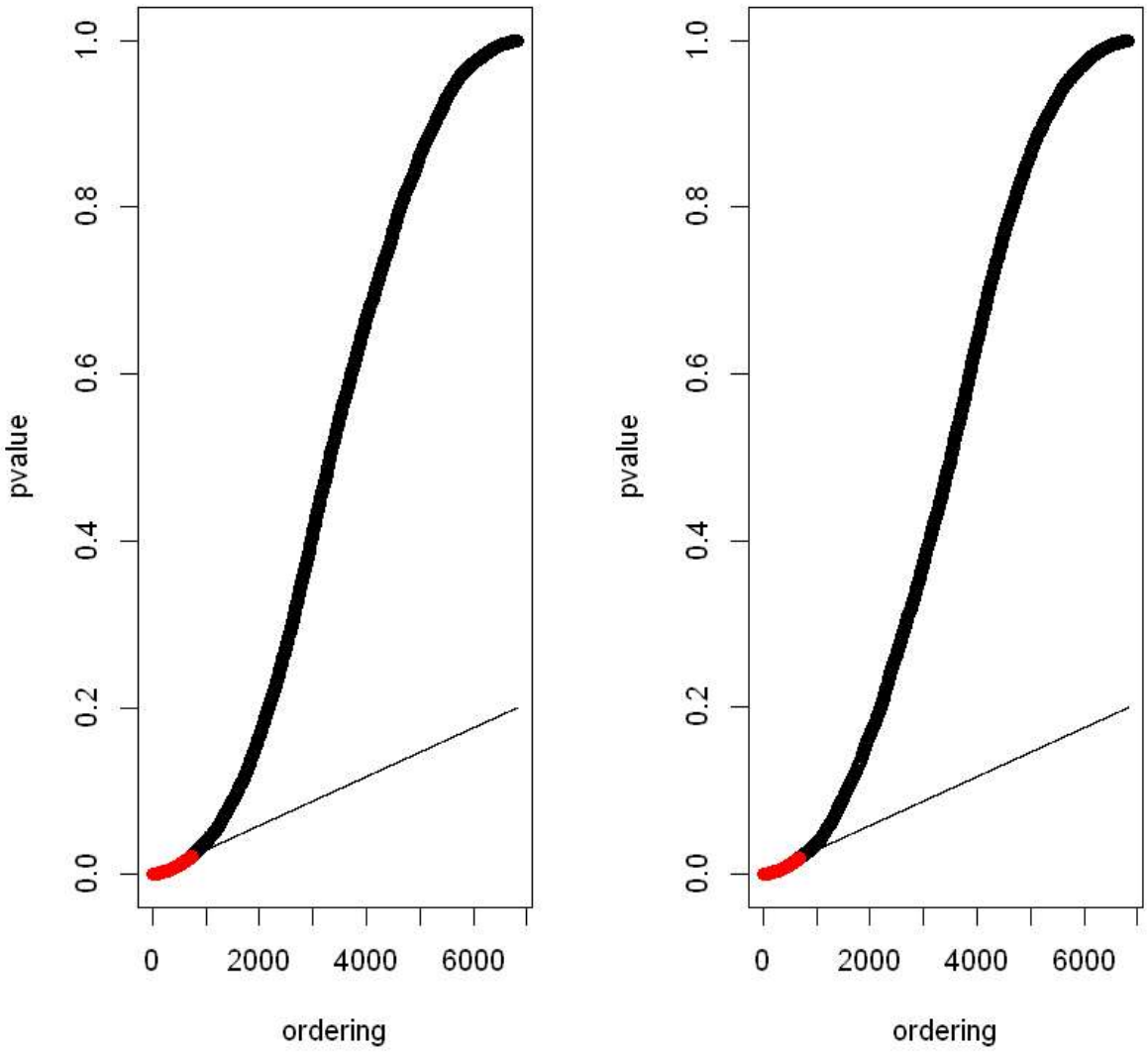
In [472]:
```
I1 <- NCI60$labs == 'CNS'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'CNS',]
par(mfrow = c(1,2))

#overexpress
overexpress_CNS <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_CNS <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)

#Prenal <- apply(m1, 2, mytfunc)

```



Cancer Type : COLON

In [473]:

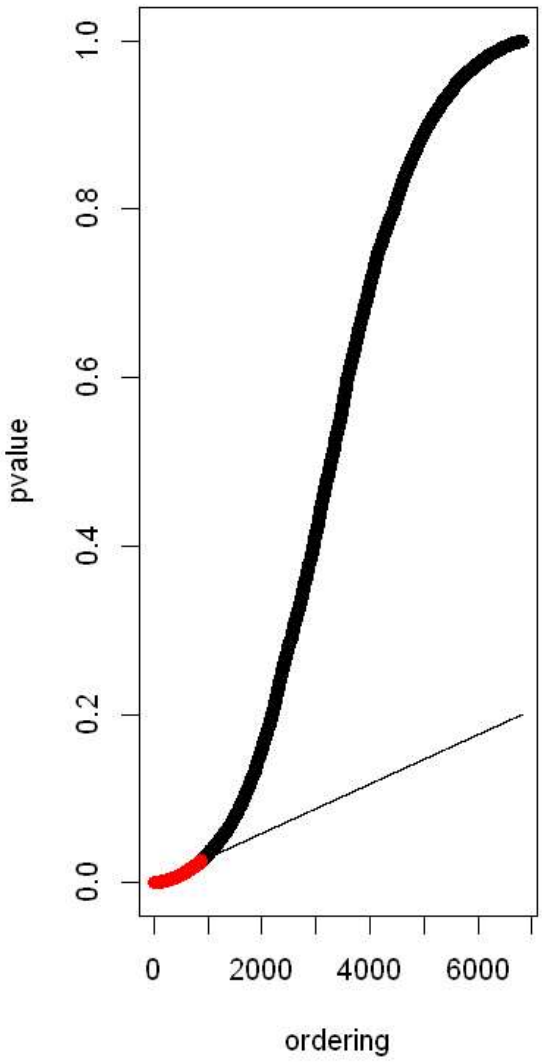
```
I1 <- NCI60$labs == 'COLON'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'COLON',]
par(mfrow = c(1,2))

#overexpress
overexpress_colon <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_colon <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)


#Prenal <- apply(m1, 2, mytfunc)

```

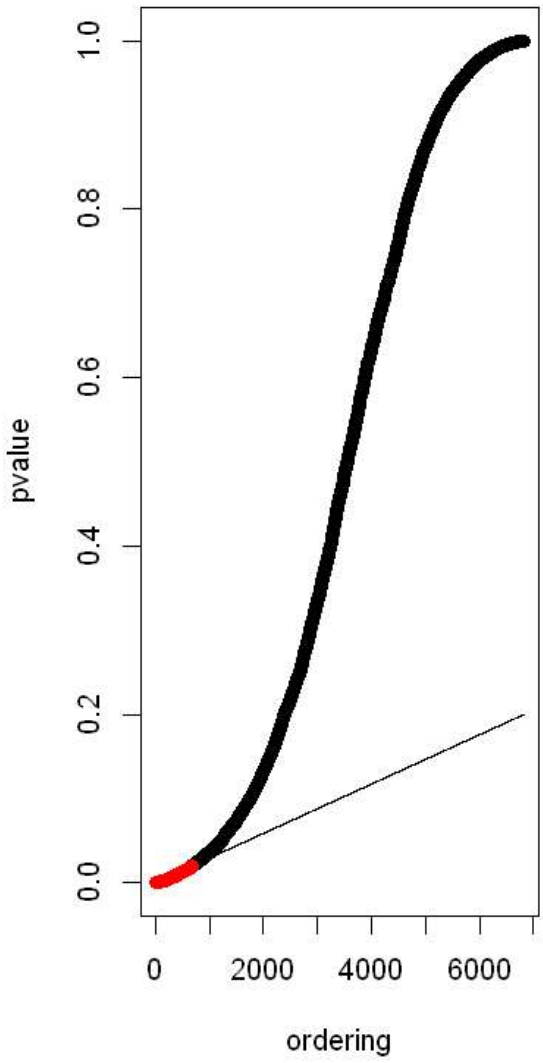


Cancer Type : LEUKEMIA

In [474]:

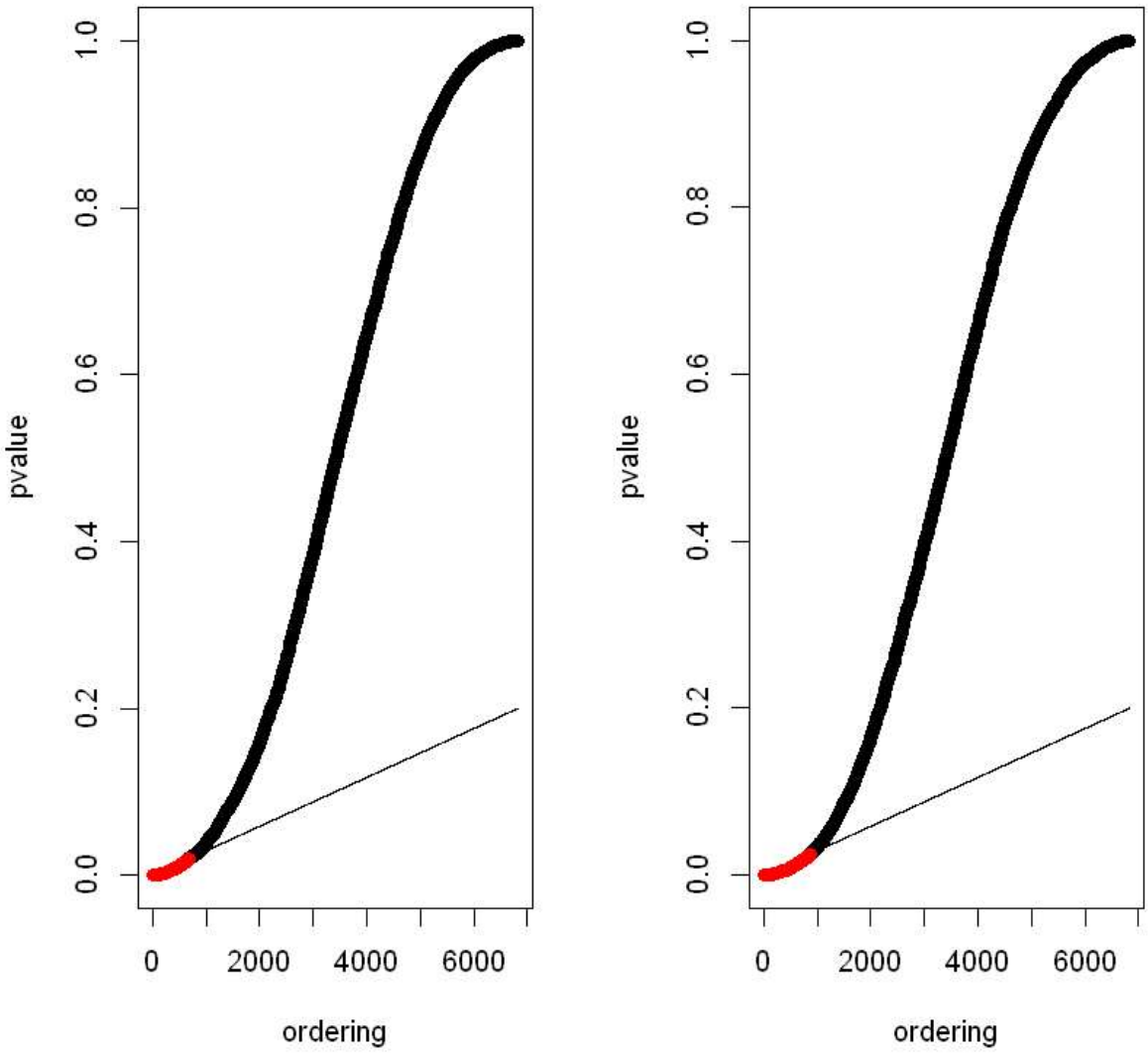
```
I1 <- NCI60$labs == 'LEUKEMIA'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'LEUKEMIA',]
par(mfrow = c(1,2))

#overexpress
overexpress_leukemia <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_leukemia <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)


#Prenal <- apply(m1, 2, mytfunc)

```





Cancer Type : MELANOMA

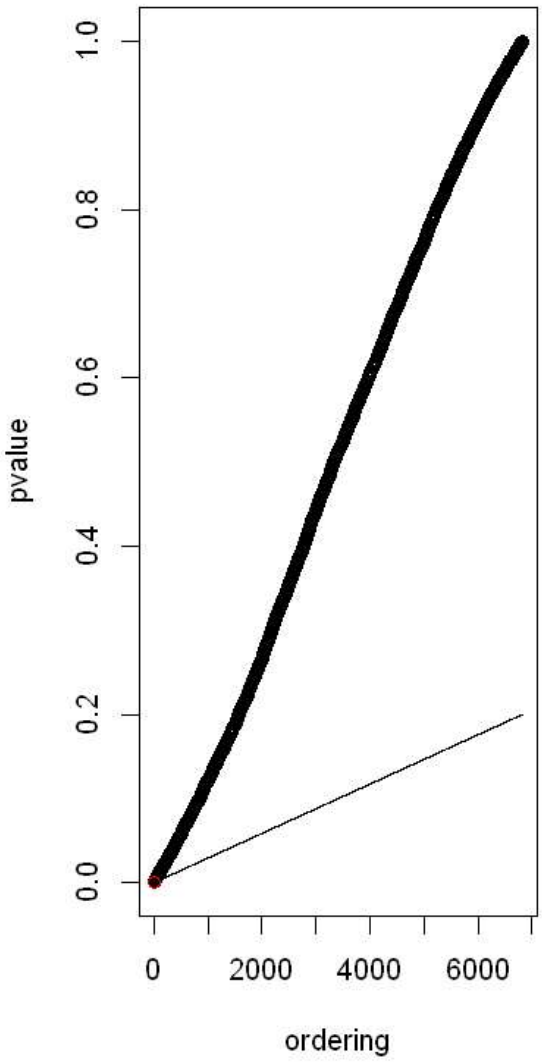
In [475]:
```
I1 <- NCI60$labs == 'MELANOMA'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'MELANOMA',]
par(mfrow = c(1,2))

#overexpress
overexpress_MELANOMA <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_MELANOMA <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)


#Prenal <- apply(m1, 2, mytfunc)

```

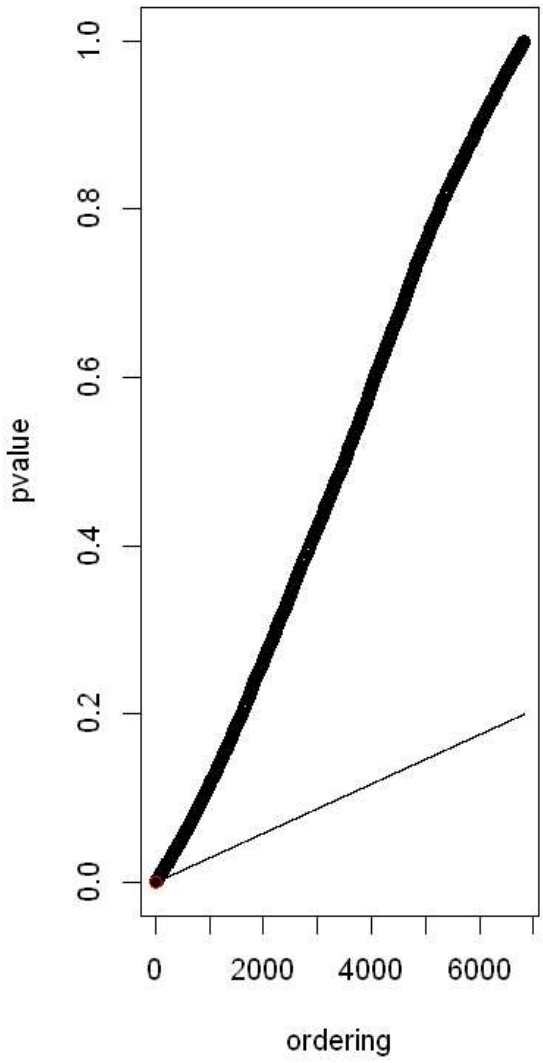


Cancer Type : NSCLC

In [476]:

```
I1 <- NCI60$labs == 'NSCLC'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'NSCLC',]
par(mfrow = c(1,2))

#overexpress
overexpress_NSCLC <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_NSCLC <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)


#Prenal <- apply(m1, 2, mytfunc)

```



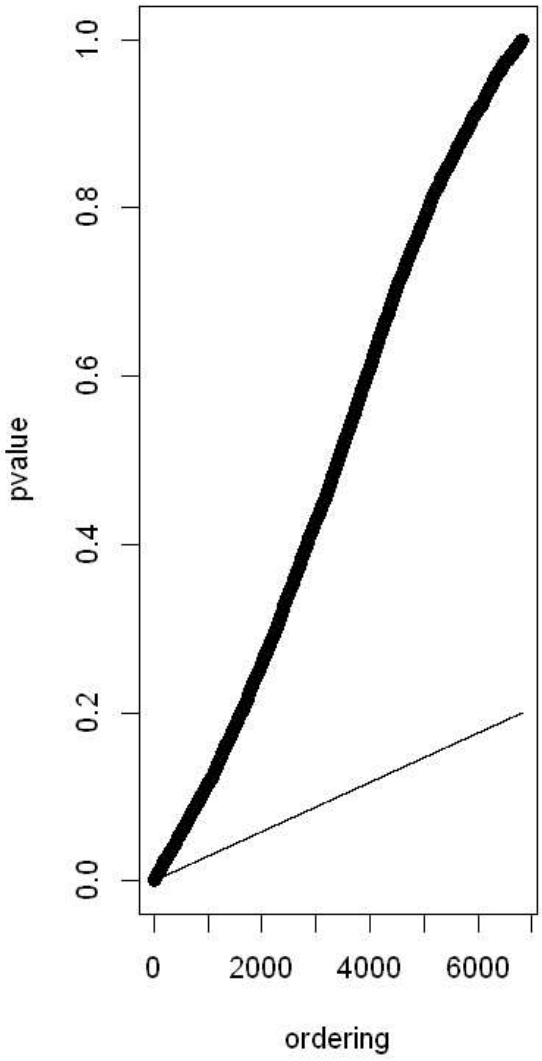


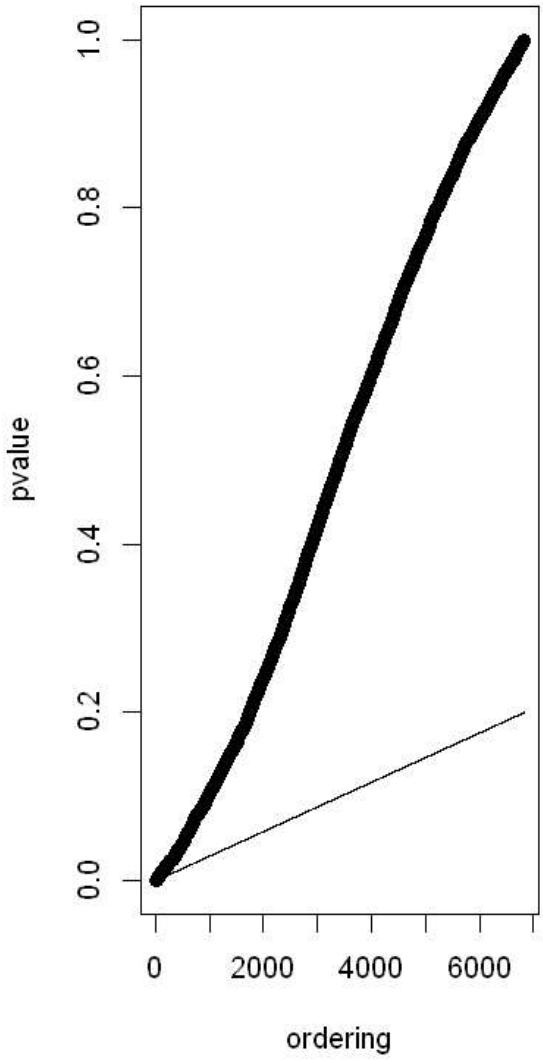
Cancer Type : OVARIAN

In [477]:

```
I1 <- NCI60$labs == 'OVARIAN'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'OVARIAN',]
par(mfrow = c(1,2))

#overexpress
overexpress_OVARIAN <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_OVARIAN <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)


#Prenal <- apply(m1, 2, mytfunc)

```

```
Warning message in max(pvec[I1], na.rm = T):
"no non-missing arguments to max; returning -Inf"Warning message in max(pvec[I
1], na.rm = T):
"no non-missing arguments to max; returning -Inf"
```

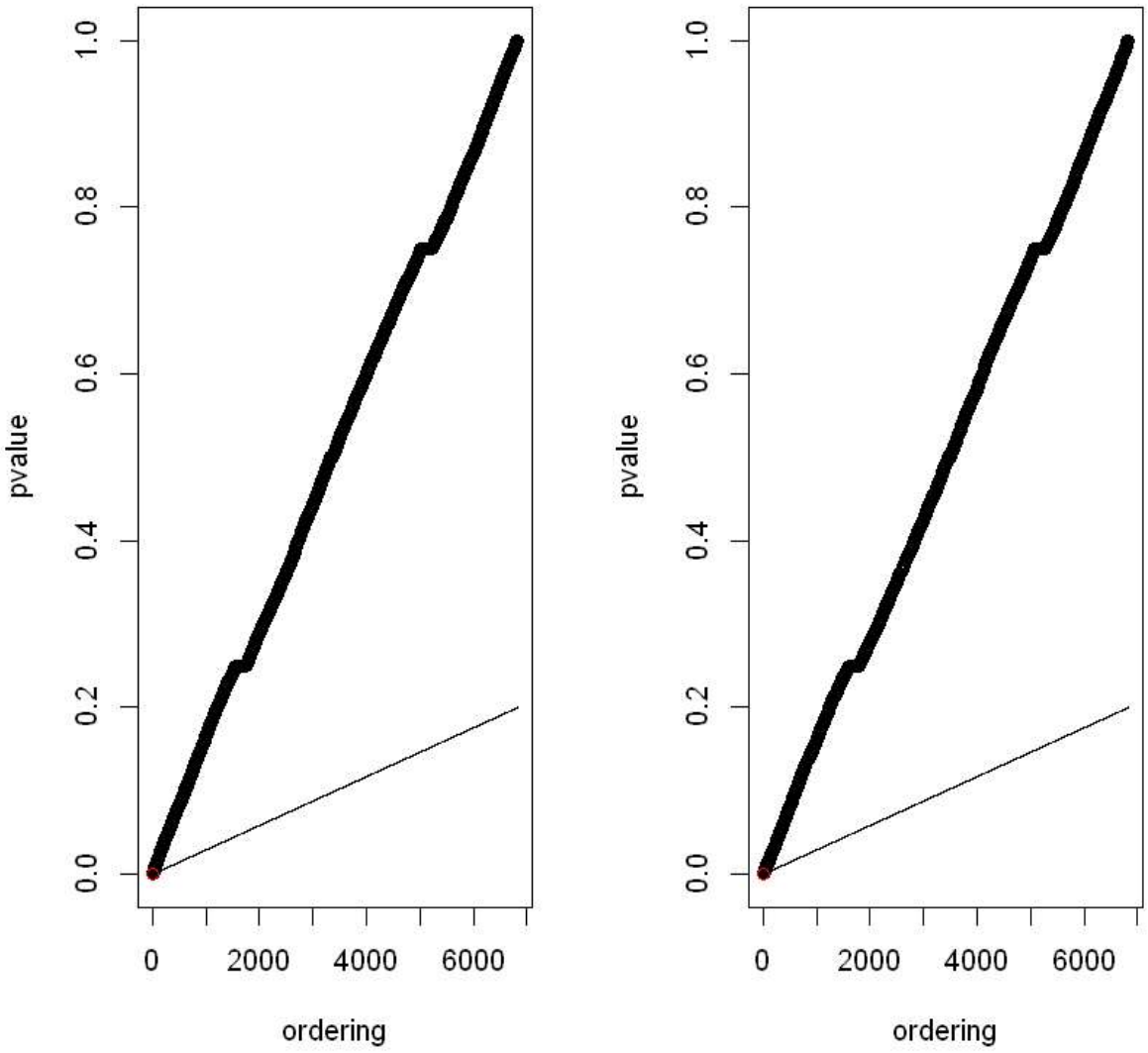




FDR test does not extract any significant gene for OVARIAN so we will not include it in our analysis since it has no genes.

Cancer Type : PROSTATE

In [478]:
```
I1 <- NCI60$labs == 'PROSTATE'
m1 <- NCI60$data[I1,]

duh_mat <- NCI60$data[NCI60$labs == 'PROSTATE',]
par(mfrow = c(1,2))

#overexpress
overexpress_PROSTATE <- fdr(apply(duh_mat, 2, mytfunc)[1,], .2)


#underexpress
underexpress_PROSTATE <- fdr(apply(duh_mat, 2, mytfunc)[2,], .2)


#Prenal <- apply(m1, 2, mytfunc)

```



Comparing OverExpress and UnderExpress

Values of Different types of cancer

Once we identify, the significant underexpressed and overexpressed genes in different cancers, we need to see which ones are common between the cancer types. This helps in creating medicines which can tackle multiple cancer types at once and is of significant value to the pharmaceutical companies.

Leukemia v/s Melanoma overexpress

In [479]:
```
Reduce(intersect, list(overexpress_leukemia,overexpress_MELANOMA))
print(length(Reduce(intersect, list(overexpress_leukemia,overexpress_MELANOMA
```

```
5930  6080  329  5931  6081  6604  5965  6369  5559  5816  5924  190  5556  6391
332  251  245  278  5584  6627  5589  252  6380  214  6041  5557  196  5841  5837
5558  5673  5742  5932  6370  6390  167  5940  6416  5586  6392  6414  143  6399
6275  287  5136  6384  5929  5746  5439  281  5744  336  283  6312  5567  5971
246  165  5463  325  321  6039  307  5899  6388  5437  5755  5970  5104  5452
351  6415  5964  327  5450  5972  6351  159  5762  4971  181  6382  5839  3233
322  5761  5453

[1] 88
```

Leukemia v/s Melanoma underexpress

In [480]:
```
Reduce(intersect, list(underexpress_leukemia,underexpress_MELANOMA))
print(length(Reduce(intersect, list(underexpress_leukemia,underexpress_MELANC
```

```
2163  2822  2215  2347  5356  2247  2087  2724  2723  2246  3520  2270  2394
2159  2221  4195  2716  2061  4191  2257  4413  2348  4193  2717  2126  2395
2164  2607  4205  4586  3370  2236  4591  3761  4192  2918  1297  1430  2346
2233  814  1107  2732  2158  2211  2224  1462  4587  4109  2345  2235  4234  2733
438  2726  2606  4185  1442  1629

[1] 59
```

Colon v/s MELANOMA overexpress

In [481]:
```
Reduce(intersect, list(overexpress_colon,overexpress_MELANOMA))
print(length(Reduce(intersect, list(overexpress_colon,overexpress_MELANOMA))))
```

```
2888  6110  6111  3356  5502  6114  5731  5899  5972  5836  5971  6322  6109
5514  6260  5794  5964  5839  5965  5513  936  6382  5730  5729  5100  5715  5970
5834  6404  5837  5958  5294  5631  6655  926  5734  6414  5835  6367  4907  5519
687  5293  5957  5757  5976  2333  5795  6355  6358

[1] 50
```

Colon v/s MELANOMA underexpress

In [482]:
```
Reduce(intersect, list(underexpress_colon,underexpress_MELANOMA))
print(length(Reduce(intersect, list(underexpress_colon,underexpress_MELANOMA))))
```

```
4550  3956  2228  1060  3957  3961  3304  4549  762  4057  3973  4245  4242  4635
4243  4595  3954  401  3286  3955  4058  3959  4634  4244  3977  4633  2895  4660

[1] 28
```

Leukemia v/s Colon overexpress

In [483]:
```
Reduce(intersect, list(overexpress_leukemia,overexpress_colon))
print(length(Reduce(intersect, list(overexpress_leukemia,overexpress_colon))))
```

```
5872  5868  5878  5934  6023  6431  5869  5953  5903  5886  4289  5915  5887
6249  6141  6072  6137  5965  6073  5939  5866  5933  5956  4353  4288  6143
6087  5810  5961  5891  6417  5892  5803  5890  6051  6065  5815  5875  6277
5898  4373  5809  6172  5937  6064  6031  5534  5902  5913  6142  5901  5814
5871  6245  6136  5805  5962  4286  5946  5910  4005  6015  6140  5535  5813
5960  4105  5952  5837  6130  5840  3936  6218  4254  4846  5947  6099  5914
4330  5802  5951  4095  5804  5935  5938  5686  5825  5251  4331  6097  6006
6085  5429  4011  6414  1  5829  6135  5864  5874  5109  5885  6104  5676  4290
5113  5636  6098  6035  5909  5718  5936  5984  5425  6581  6044  5081  5950
4009  6092  5537  6254  6133  4488  4257  5430  5943  5855  6091  6063  5411
4267  4168  5859  5955  5971  4164  5912  481  6000  6695  5899  6278  5684  4106
4156  5101  4163  5911  6094  6054  6134  5948  4158  6123  5852  5873  6100
6203  4352  5970  6053  5856  6082  5870  5614  5830  6084  4099  6101  5964
6055  4423  4103  5687  4341  5941  5884  4395  5972  5896  5982  6382  5839
6001  6290  4359  5760  5824  5862

[1] 190
```

We see that there are 190 common genes which are underexpressed between COLON and LEUKEMIA. There are 99 common genes which are overexpressed between the 2 cancers. This means that there are higher chances of creating a medicine for the underexpressed genes and we could use the same medicine to tackle both the cancer types.

Leukemia v/s Colon underexpress

In [484]:
```
Reduce(intersect, list(underexpress_leukemia,underexpress_colon))
print(length(Reduce(intersect, list(underexpress_leukemia,underexpress_colon)
```

```
1915  1691  1346  2038  1347  2142  1904  1815  2090  1777  1903  1695  1697
1539  1453  1871  1893  2091  2128  1894  2298  1451  1870  1968  2025  1694
2151  1936  2129  3607  1956  2287  1301  2024  3406  1696  2147  2245  1887
2423  3295  2049  2148  1560  3296  1929  1914  1450  2130  6774  1792  1762
1452  2047  2171  1769  1864  1835  2424  2282  2059  2008  1519  392  2131  2187
1021  468  1982  3595  1818  232  1902  1572  1850  2943  3294  1728  1768  1927
2283  1849  1868  1563  1313  3297  2385  1916  1922  2042  2135  1901  1847  391
2127  739  1918  1538  1841

[1] 99
```

Leukemia v/s Renal overexpress

In [485]:
```
Reduce(intersect, list(overexpress_leukemia,overexpress_renal))
print(length(Reduce(intersect, list(overexpress_leukemia,overexpress_renal))))
```

```
4216  4214  4213  3963  4215

[1] 5
```

Leukemia v/s Renal underexpress

In [486]:
```
Reduce(intersect, list(underexpress_leukemia,underexpress_renal))
print(length(Reduce(intersect, list(underexpress_leukemia,underexpress_renal)
```

```
3380  2944  3381  3445  3459  2939  5632

[1] 7
```

Renal v/s Colon overexpress

In [487]:
```
Reduce(intersect, list(overexpress_renal,overexpress_colon))
print(length(Reduce(intersect, list(overexpress_renal,overexpress_colon))))
```

```
2813  2777  2632

[1] 3
```

We see that in Colon and RENAL, there are very few common genes between the 2, so its harder to create a common medicine to tackle these 2 types of cancer

Renal v/s Colon underexpress

In [488]:
```
Reduce(intersect, list(underexpress_renal,underexpress_colon))
print(length(Reduce(intersect, list(underexpress_renal,underexpress_colon))))
```

282 98 100 252 355 151 152 5350 5338 175 354 6535 11 4251 470

[1] 15

Breast v/s Colon underexpress

In [489]:
```
Reduce(intersect, list(underexpress_breast,underexpress_colon))
print(length(Reduce(intersect, list(underexpress_breast,underexpress_colon))))
```

[1] 0

Breast v/s Colon overexpress

In [490]:
```
Reduce(intersect, list(overexpress_breast,overexpress_colon))
print(length(Reduce(intersect, list(overexpress_breast,overexpress_colon))))
```

[1] 0

As we see, there are no common overexpressed and underexpressed genes between the BREAST and Colon. Thus a cancer company cannot create common medicines to tackle both simultaneously.

MELANOMA v/s Renal overexpress

In [491]:
```
Reduce(intersect, list(overexpress_MELANOMA,overexpress_renal))
print(length(Reduce(intersect, list(overexpress_MELANOMA,overexpress_renal))))
```

3034 1694 3148 3147 1695 341 1841 1696 2182

[1] 9

MELANOMA v/s Renal underexpress

In [492]:
```
Reduce(intersect, list(underexpress_MELANOMA,underexpress_renal))
print(length(Reduce(intersect, list(underexpress_MELANOMA,underexpress_renal)
```

```
4349  4256  4348  5984  4288  4332  4255  4289  4277  4350  5804  4286  4362
5773  4285  4278  4259  6151  6150  5887  4098  5774  6152  6156  5888  6153
5659  5805  4352  6154  4011  6157  6061  5775  5619  4643  5886  4642  4373
3895  6148  6149  5380  5884  6203  4292  4260  5316  6006  5808  6155  4374
6247  6158  3562  6147  3510  4701  5315  3841  4342  4247  4700  5807  5508
5863  6306  6433  6184  3842  5806  5810  5986  5360  5876  5661  5802  6015
5809  6304  6524  5687  4699  6031  4390  6161  4106  5678  6051  6159  6550
5803  4698  6106  3893

[1] 95
```

Leukemia v/s Colon v/s Renal overexpress

In [493]:
```
Reduce(intersect, list(overexpress_colon,overexpress_renal,overexpress_leukem
print(length(Reduce(intersect, list(overexpress_colon,overexpress_renal,overe
```

```
[1] 0
```

Leukemia v/s Colon v/s Renal underexpress

In [494]:
```
Reduce(intersect, list(underexpress_colon,underexpress_renal,underexpress_leu
print(length(Reduce(intersect, list(underexpress_colon,underexpress_renal,und
```

```
[1] 0
```

We see that there are no overexpressed or underexpressed genes between the 3 cancer types and so no medicine can tackle the 3 of them simultaneously. Similarly, we can check for more combinations.

Additional Analysis on Cancer Cells DATA

In [495]:
```
nci.labs=NCI60$labs
nci.data=NCI60$data
#The data has 64 rows and 6,830 columns.
dim(nci_data)
```

```
64  6830
```

In [496]:
```
1 table(nci_labs)
```

```
nci_labs
     BREAST         CNS       COLON K562A-repro K562B-repro    LEUKEMIA 
          7           5           7           1           1           6 
MCF7A-repro MCF7D-repro    MELANOMA       NSCLC     OVARIAN    PROSTATE 
          1           1           8           9           6           2 
      RENAL     UNKNOWN 
          9           1 
```

PCA on the NCI60 Data

We first perform PCA on the data after scaling the variables (genes) to have standard deviation one, although one could reasonably argue that it is better not to scale the genes.

In [497]:
```
1 pr.out =prcomp (nci.data , scale=TRUE)
```

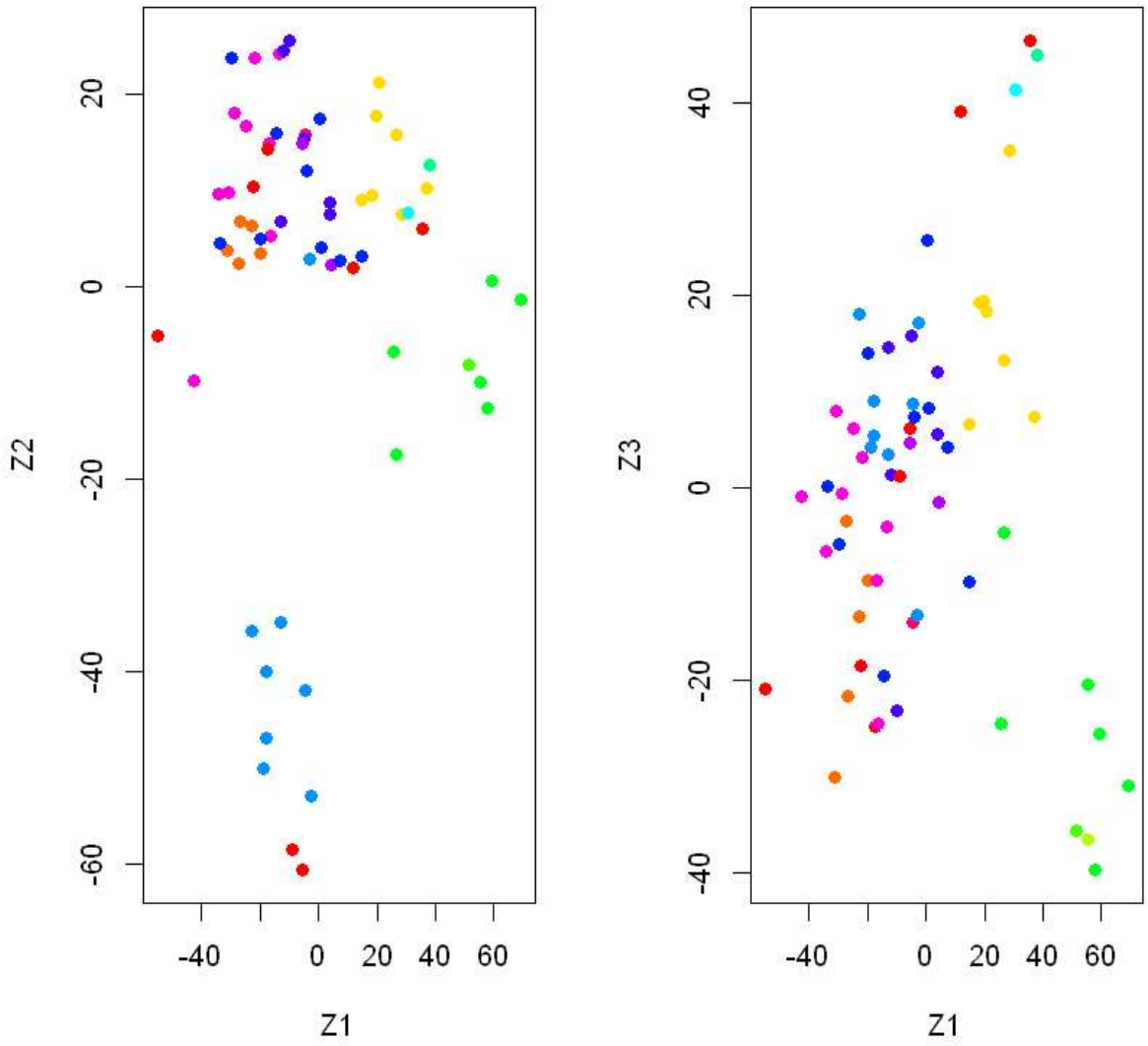
We now plot the first few principal component score vectors, in order to visualize the data. The observations (cell lines) corresponding to a given cancer type will be plotted in the same color, so that we can see to what extent the observations within a cancer type are similar to each other We first create a simple function that assigns a distinct color to each element of a numeric vector. The function will be used to assign a color to each of the 64 cell lines, based on the cancer type to which it corresponds.

#Note that the rainbow() function takes as its argument a positive integer, rainbow() and returns a vector containing that number of distinct colors.

In [498]:
```
1 Cols <- function(vec) {
2   cols <- rainbow(length(unique(vec)))
3   return(cols[as.numeric(as.factor(vec))])
4 }
```

We now can plot the principal component score vectors.

In [499]:
```
par(mfrow = c(1, 2))
par(mfrow =c(1,2))
plot(pr.out$x[,1:2], col=Cols(nci.labs), pch =19,
xlab ="Z1",ylab="Z2")
plot(pr.out$x[,c(1,3) ], col=Cols(nci.labs), pch =19,
xlab ="Z1",ylab="Z3")
```



On the whole, cell lines corresponding to a single cancer type do tend to have similar values on the first few principal component score vectors. This indicates that cell lines from the same cancer type tend to have pretty similar gene expression levels.

We can obtain a summary of the proportion of variance explained (PVE) of the first few principal components using the summary() method for a prcomp object :

In [500]:
```
#PVE
summary (pr.out)
```

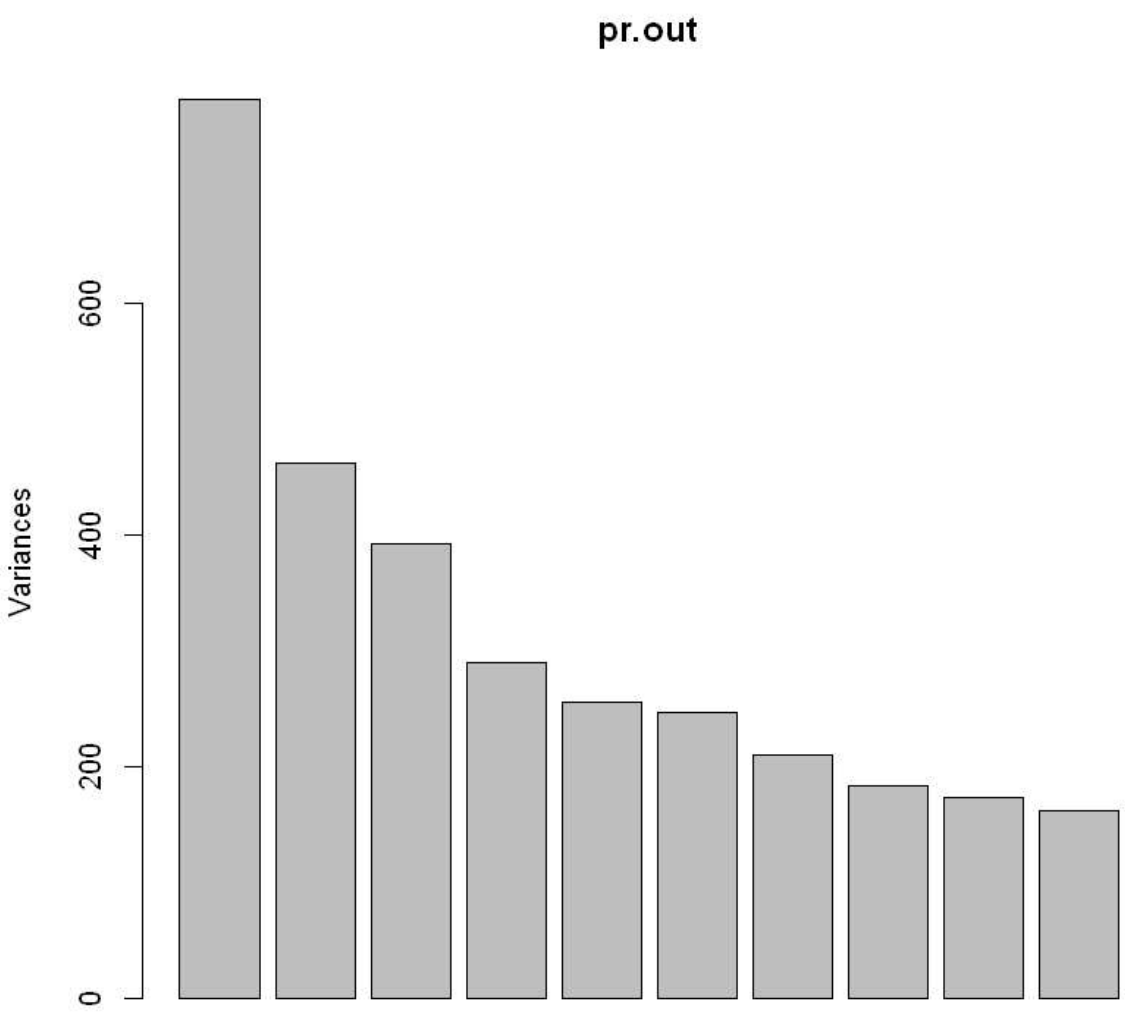
```
Importance of components:
                           PC1      PC2      PC3      PC4      PC5      PC6
Standard deviation     27.8535 21.48136 19.82046 17.03256 15.97181 15.72108
Proportion of Variance  0.1136  0.06756  0.05752  0.04248  0.03735  0.03619
Cumulative Proportion   0.1136  0.18115  0.23867  0.28115  0.31850  0.35468
                           PC7      PC8      PC9     PC10     PC11     PC12
Standard deviation     14.47145 13.54427 13.14400 12.73860 12.68672 12.15769
Proportion of Variance  0.03066  0.02686  0.02529  0.02376  0.02357  0.02164
Cumulative Proportion   0.38534  0.41220  0.43750  0.46126  0.48482  0.50646
                          PC13     PC14     PC15     PC16     PC17     PC18
Standard deviation     11.83019 11.62554 11.43779 11.00051 10.65666 10.48880
Proportion of Variance  0.02049  0.01979  0.01915  0.01772  0.01663  0.01611
Cumulative Proportion   0.52695  0.54674  0.56590  0.58361  0.60024  0.61635
                          PC19    PC20     PC21    PC22    PC23    PC24
Standard deviation     10.43518 10.3219 10.14608 10.0544 9.90265 9.64766
Proportion of Variance  0.01594  0.0156  0.01507  0.0148 0.01436 0.01363
Cumulative Proportion   0.63229  0.6479  0.66296  0.6778 0.69212 0.70575
                          PC25    PC26    PC27   PC28    PC29    PC30    PC31
Standard deviation     9.50764 9.33253 9.27320 9.0900 8.98117 8.75003 8.59962
Proportion of Variance 0.01324 0.01275 0.01259 0.0121 0.01181 0.01121 0.01083
Cumulative Proportion  0.71899 0.73174 0.74433 0.7564 0.76824 0.77945 0.79027
                          PC32    PC33    PC34    PC35    PC36    PC37    PC38
Standard deviation     8.44738 8.37305 8.21579 8.15731 7.97465 7.90446 7.82127
Proportion of Variance 0.01045 0.01026 0.00988 0.00974 0.00931 0.00915 0.00896
Cumulative Proportion  0.80072 0.81099 0.82087 0.83061 0.83992 0.84907 0.85803
                          PC39    PC40    PC41   PC42    PC43   PC44    PC45
Standard deviation     7.72156 7.58603 7.45619 7.3444 7.10449 7.0131 6.95839
Proportion of Variance 0.00873 0.00843 0.00814 0.0079 0.00739 0.0072 0.00709
Cumulative Proportion  0.86676 0.87518 0.88332 0.8912 0.89861 0.9058 0.91290
                          PC46    PC47    PC48    PC49    PC50    PC51    PC52
Standard deviation     6.8663 6.80744 6.64763 6.61607 6.40793 6.21984 6.20326
Proportion of Variance 0.0069 0.00678 0.00647 0.00641 0.00601 0.00566 0.00563
Cumulative Proportion  0.9198 0.92659 0.93306 0.93947 0.94548 0.95114 0.95678
                          PC53    PC54    PC55    PC56    PC57   PC58    PC59
Standard deviation     6.06706 5.91805 5.91233 5.73539 5.47261 5.2921 5.02117
Proportion of Variance 0.00539 0.00513 0.00512 0.00482 0.00438 0.0041 0.00369
Cumulative Proportion  0.96216 0.96729 0.97241 0.97723 0.98161 0.9857 0.98940
                          PC60    PC61    PC62    PC63      PC64
Standard deviation     4.68398 4.17567 4.08212 4.04124 1.237e-14
Proportion of Variance 0.00321 0.00255 0.00244 0.00239 0.000e+00
Cumulative Proportion  0.99262 0.99517 0.99761 1.00000 1.000e+00
```

In [501]:
```
1 summary(pr_out)
```

```
Importance of components:
                           PC1      PC2      PC3      PC4      PC5      PC6
Standard deviation     27.8535 21.48136 19.82046 17.03256 15.97181 15.72108
Proportion of Variance  0.1136  0.06756  0.05752  0.04248  0.03735  0.03619
Cumulative Proportion   0.1136  0.18115  0.23867  0.28115  0.31850  0.35468
                           PC7      PC8      PC9     PC10     PC11     PC12
Standard deviation     14.47145 13.54427 13.14400 12.73860 12.68672 12.15769
Proportion of Variance  0.03066  0.02686  0.02529  0.02376  0.02357  0.02164
Cumulative Proportion   0.38534  0.41220  0.43750  0.46126  0.48482  0.50646
                          PC13     PC14     PC15     PC16     PC17     PC18
Standard deviation     11.83019 11.62554 11.43779 11.00051 10.65666 10.48880
Proportion of Variance  0.02049  0.01979  0.01915  0.01772  0.01663  0.01611
Cumulative Proportion   0.52695  0.54674  0.56590  0.58361  0.60024  0.61635
                          PC19    PC20     PC21    PC22    PC23    PC24
Standard deviation     10.43518 10.3219 10.14608 10.0544 9.90265 9.64766
Proportion of Variance  0.01594  0.0156  0.01507  0.0148 0.01436 0.01363
Cumulative Proportion   0.63229  0.6479  0.66296  0.6778 0.69212 0.70575
                          PC25    PC26    PC27   PC28    PC29    PC30    PC31
Standard deviation     9.50764 9.33253 9.27320 9.0900 8.98117 8.75003 8.59962
Proportion of Variance 0.01324 0.01275 0.01259 0.0121 0.01181 0.01121 0.01083
Cumulative Proportion  0.71899 0.73174 0.74433 0.7564 0.76824 0.77945 0.79027
                          PC32    PC33    PC34    PC35    PC36    PC37    PC38
Standard deviation     8.44738 8.37305 8.21579 8.15731 7.97465 7.90446 7.82127
Proportion of Variance 0.01045 0.01026 0.00988 0.00974 0.00931 0.00915 0.00896
Cumulative Proportion  0.80072 0.81099 0.82087 0.83061 0.83992 0.84907 0.85803
                          PC39    PC40    PC41   PC42    PC43   PC44    PC45
Standard deviation     7.72156 7.58603 7.45619 7.3444 7.10449 7.0131 6.95839
Proportion of Variance 0.00873 0.00843 0.00814 0.0079 0.00739 0.0072 0.00709
Cumulative Proportion  0.86676 0.87518 0.88332 0.8912 0.89861 0.9058 0.91290
                          PC46    PC47    PC48    PC49    PC50    PC51    PC52
Standard deviation     6.8663 6.80744 6.64763 6.61607 6.40793 6.21984 6.20326
Proportion of Variance 0.0069 0.00678 0.00647 0.00641 0.00601 0.00566 0.00563
Cumulative Proportion  0.9198 0.92659 0.93306 0.93947 0.94548 0.95114 0.95678
                          PC53    PC54    PC55    PC56    PC57   PC58    PC59
Standard deviation     6.06706 5.91805 5.91233 5.73539 5.47261 5.2921 5.02117
Proportion of Variance 0.00539 0.00513 0.00512 0.00482 0.00438 0.0041 0.00369
Cumulative Proportion  0.96216 0.96729 0.97241 0.97723 0.98161 0.9857 0.98940
                          PC60    PC61    PC62    PC63      PC64
Standard deviation     4.68398 4.17567 4.08212 4.04124 1.237e-14
Proportion of Variance 0.00321 0.00255 0.00244 0.00239 0.000e+00
Cumulative Proportion  0.99262 0.99517 0.99761 1.00000 1.000e+00
```

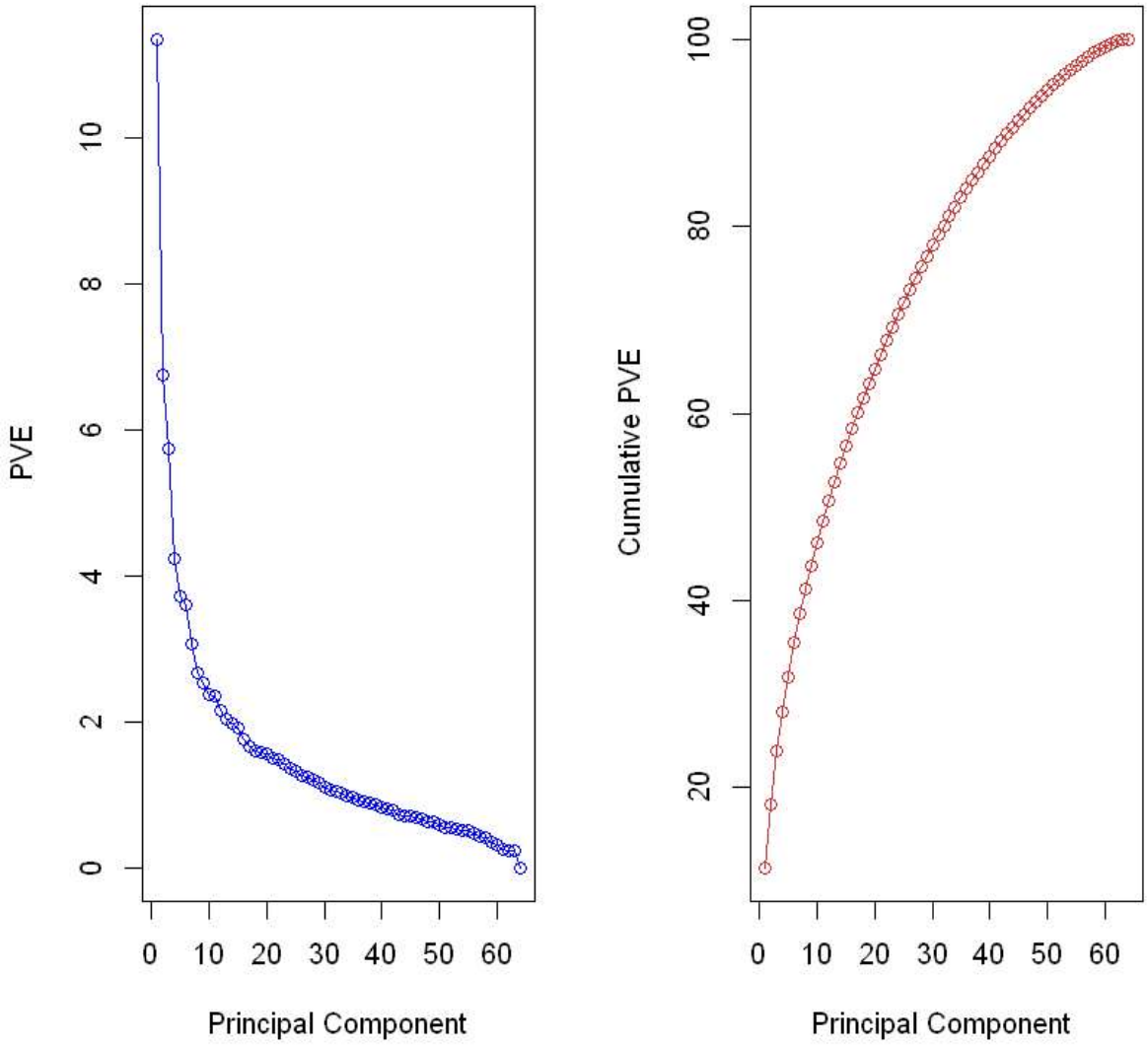
Using the plot() function, we can also plot the variance explained by the first few principal components.

In [502]:
```
1 plot(pr.out)
```



Note that the height of each bar in the bar plot is given by squaring the corresponding element of pr.out$sdev. It is more informative plot the PVE of each principal component (i.e. a scree plot) and the cumulative PVE of each principal component

In [503]:
```
pve <- 100 * pr_out$sdev^2/sum(pr_out$sdev^2)
par(mfrow = c(1, 2))
plot(pve, type = "o", ylab = "PVE", xlab = "Principal Component", col = " blu
plot(cumsum(pve), type = "o", ylab = "Cumulative PVE", xlab = "Principal Comp
```



We see that together, the first seven principal components explain around 40% of the variance in the data. This is not a huge amount of the variance. However, looking at the scree plot, we see that while each of the first seven principal components explain a substantial amount of variance, there is a marked decrease in the variance explained by further principal components. That is, there is an elbow in the plot after approximately the seventh principal component. This suggests that there may be little benefit to examining more than seven or so principal components (though even examining seven principal components may be difficult).

In [504]:
```
summary(pr.out)$importance[2,]
```

PC1	0.11359
PC2	0.06756
PC3	0.05752
PC4	0.04248
PC5	0.03735
PC6	0.03619
PC7	0.03066
PC8	0.02686
PC9	0.02529
PC10	0.02376
PC11	0.02357
PC12	0.02164
PC13	0.02049
PC14	0.01979
PC15	0.01915
PC16	0.01772
PC17	0.01663
PC18	0.01611
PC19	0.01594
PC20	0.0156
PC21	0.01507
PC22	0.0148
PC23	0.01436
PC24	0.01363
PC25	0.01324
PC26	0.01275
PC27	0.01259
PC28	0.0121
PC29	0.01181
PC30	0.01121
PC31	0.01083
PC32	0.01045
PC33	0.01026
PC34	0.00988
PC35	0.00974
PC36	0.00931
PC37	0.00915
PC38	0.00896
PC39	0.00873
PC40	0.00843
PC41	0.00814
PC42	0.0079
PC43	0.00739
PC44	0.0072
PC45	0.00709
PC46	0.0069

PC47	0.00678
PC48	0.00647
PC49	0.00641
PC50	0.00601
PC51	0.00566
PC52	0.00563
PC53	0.00539
PC54	0.00513
PC55	0.00512
PC56	0.00482
PC57	0.00438
PC58	0.0041
PC59	0.00369
PC60	0.00321
PC61	0.00255
PC62	0.00244
PC63	0.00239
PC64	0

The elements of cumsum(pve)

In [505]:
```
summary(pr.out)$importance[3,]
```

PC1	0.11359
PC2	0.18115
PC3	0.23867
PC4	0.28115
PC5	0.3185
PC6	0.35468
PC7	0.38534
PC8	0.4122
PC9	0.4375
PC10	0.46126
PC11	0.48482
PC12	0.50646
PC13	0.52695
PC14	0.54674
PC15	0.5659
PC16	0.58361
PC17	0.60024
PC18	0.61635
PC19	0.63229
PC20	0.64789
PC21	0.66296
PC22	0.67776
PC23	0.69212
PC24	0.70575
PC25	0.71899
PC26	0.73174
PC27	0.74433
PC28	0.75643
PC29	0.76824
PC30	0.77945
PC31	0.79027
PC32	0.80072
PC33	0.81099
PC34	0.82087
PC35	0.83061
PC36	0.83992
PC37	0.84907
PC38	0.85803
PC39	0.86676
PC40	0.87518
PC41	0.88332
PC42	0.89122
PC43	0.89861
PC44	0.90581
PC45	0.9129
PC46	0.9198

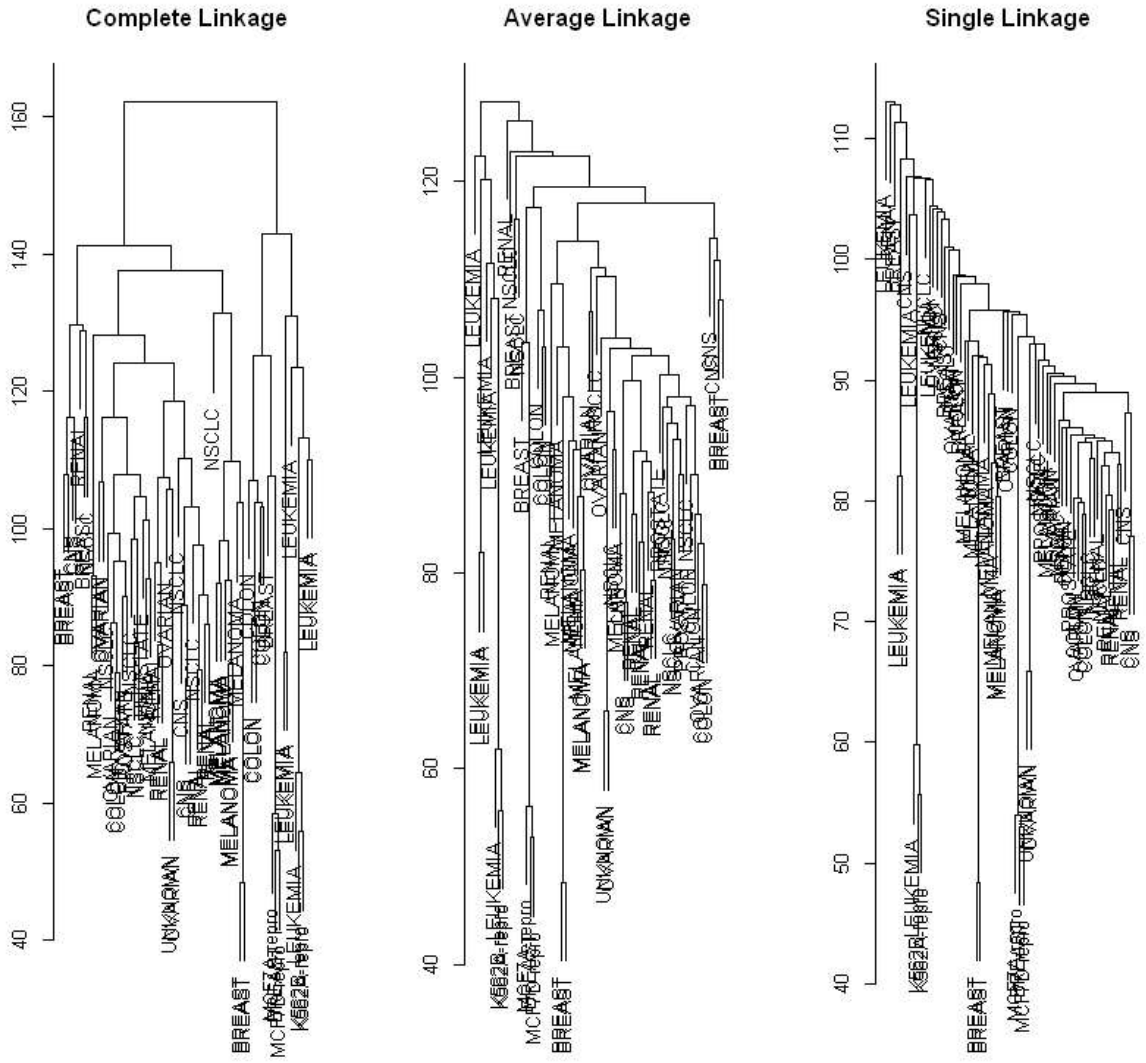
PC47	0.92659
PC48	0.93306
PC49	0.93947
PC50	0.94548
PC51	0.95114
PC52	0.95678
PC53	0.96216
PC54	0.96729
PC55	0.97241
PC56	0.97723
PC57	0.98161
PC58	0.98571
PC59	0.9894
PC60	0.99262
PC61	0.99517
PC62	0.99761
PC63	1
PC64	1

Clustering the Observations of the NCI60 Data

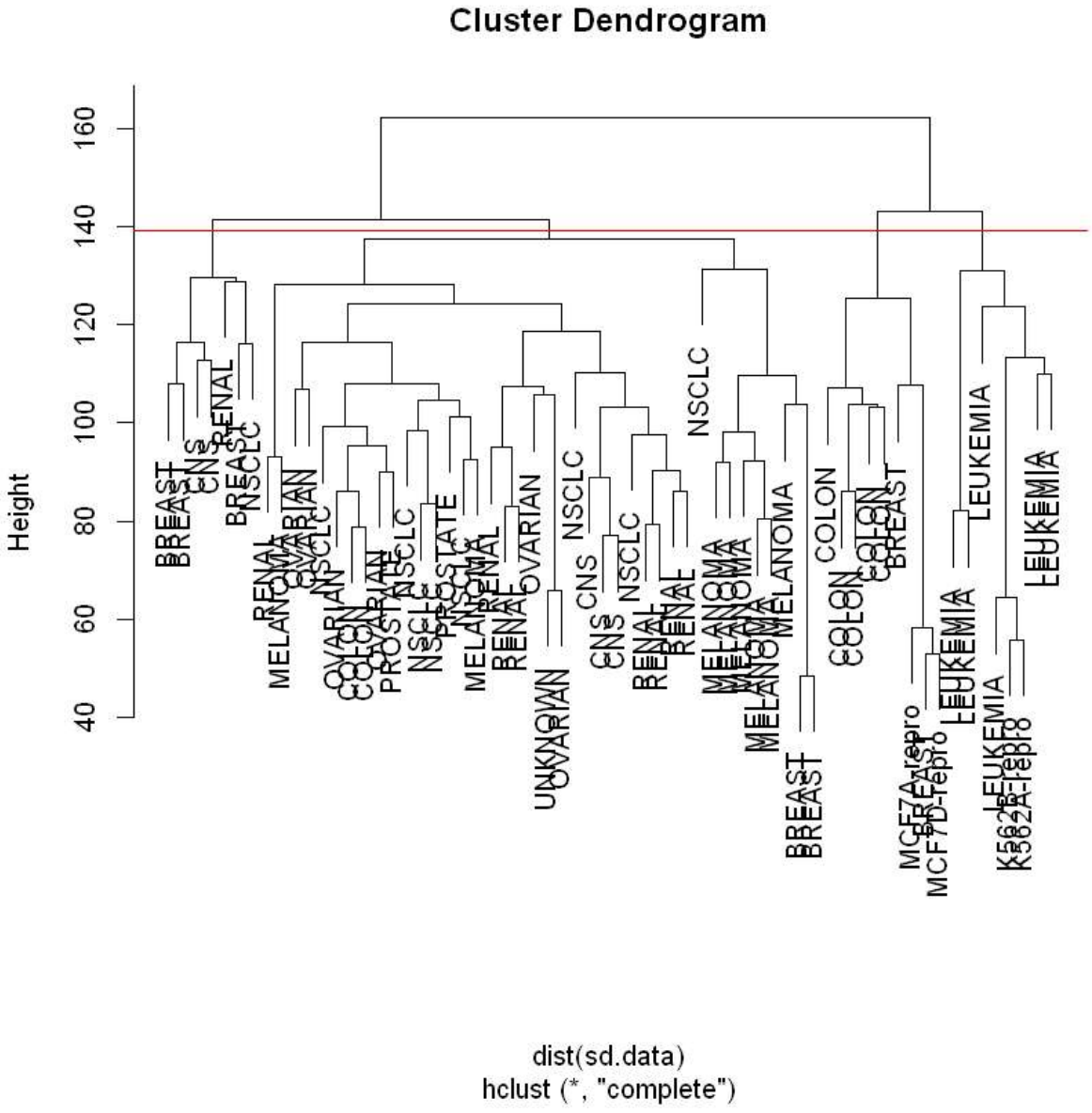
We now proceed to hierarchically cluster the cell lines in the NCI60 data, with the goal of finding out whether or not the observations cluster into distinct types of cancer.

We now perform hierarchical clustering of the observations using complete, single, and average linkage. Euclidean distance is used as the dissimilarity measure. Complete and average linkage are generally preferred to single linkage.

In [506]:
```
sd.data <- scale(nci_data)
par(mfrow = c(1, 3))
data_dist <- dist(sd.data)
plot(hclust(data_dist), labels = nci_labs, main = "Complete Linkage", xlab =
#Average
plot(hclust(data_dist, method = "average"), labels = nci_labs, main = "Averag
#Single
plot(hclust(data_dist, method = "single"), labels = nci_labs, main = "Single
```



In [507]:
```
hc.out =hclust(dist(sd.data))
par(mfrow = c(1, 1))
plot(hc.out, labels = nci_labs)
abline(h = 139, col = "red")
```



Extended Observations of the NCI60 Data

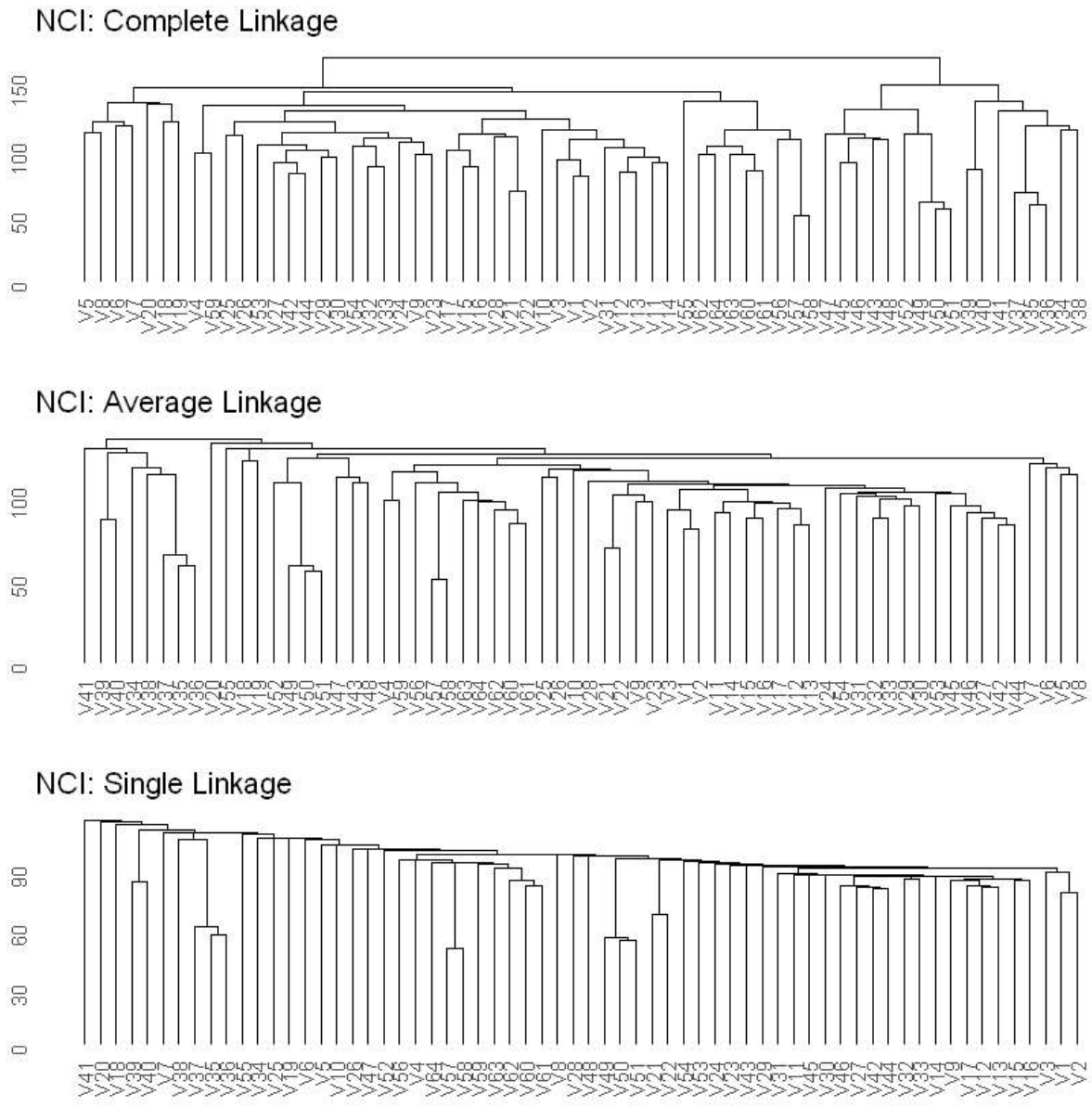
We now proceed to hierarchically cluster the cell lines in the NCI60 data, with the goal of finding out whether or not the observations cluster into distinct types of cancer. To begin, we standardize the variables to have mean zero and standard deviation one. This step is optional, and need only be performed if we want each gene to be on the same scale:

In [508]:
```
nci_labels = NCI60$labs
nci_data = NCI60$data
dim(nci_data)
scaled_nci_data = scale(nci_data)
```

64 6830

We now perform hierarchical clustering of the observations using complete, single, and average linkage. We'll use standard Euclidean distance as the dissimilarity measure:

In [509]:
```
nci_hc_complete = hclust(dist(scaled_nci_data), method = "complete")
nci_hc_average = hclust(dist(scaled_nci_data), method = "average")
nci_hc_single = hclust(dist(scaled_nci_data), method = "single")


plot_complete_nci = ggdendrogram(nci_hc_complete, rotate = FALSE, size = 2) +
plot_average_nci = ggdendrogram(nci_hc_average, rotate = FALSE, size = 2) + l
plot_single_nci = ggdendrogram(nci_hc_single, rotate = FALSE, size = 2) + lab

grid.arrange(plot_complete_nci, plot_average_nci, plot_single_nci)
```



We see that the choice of linkage certainly does affect the results obtained. Typically, single linkage will tend to yield trailing clusters: very large clusters onto which individual observations attach one-by-one. On the other hand, complete and average linkage tend to yield more balanced, attractive clusters. For this reason, complete and average linkage are generally preferred to single linkage. Clearly cell lines within a single cancer type do tend to cluster together, although the clustering is not perfect.

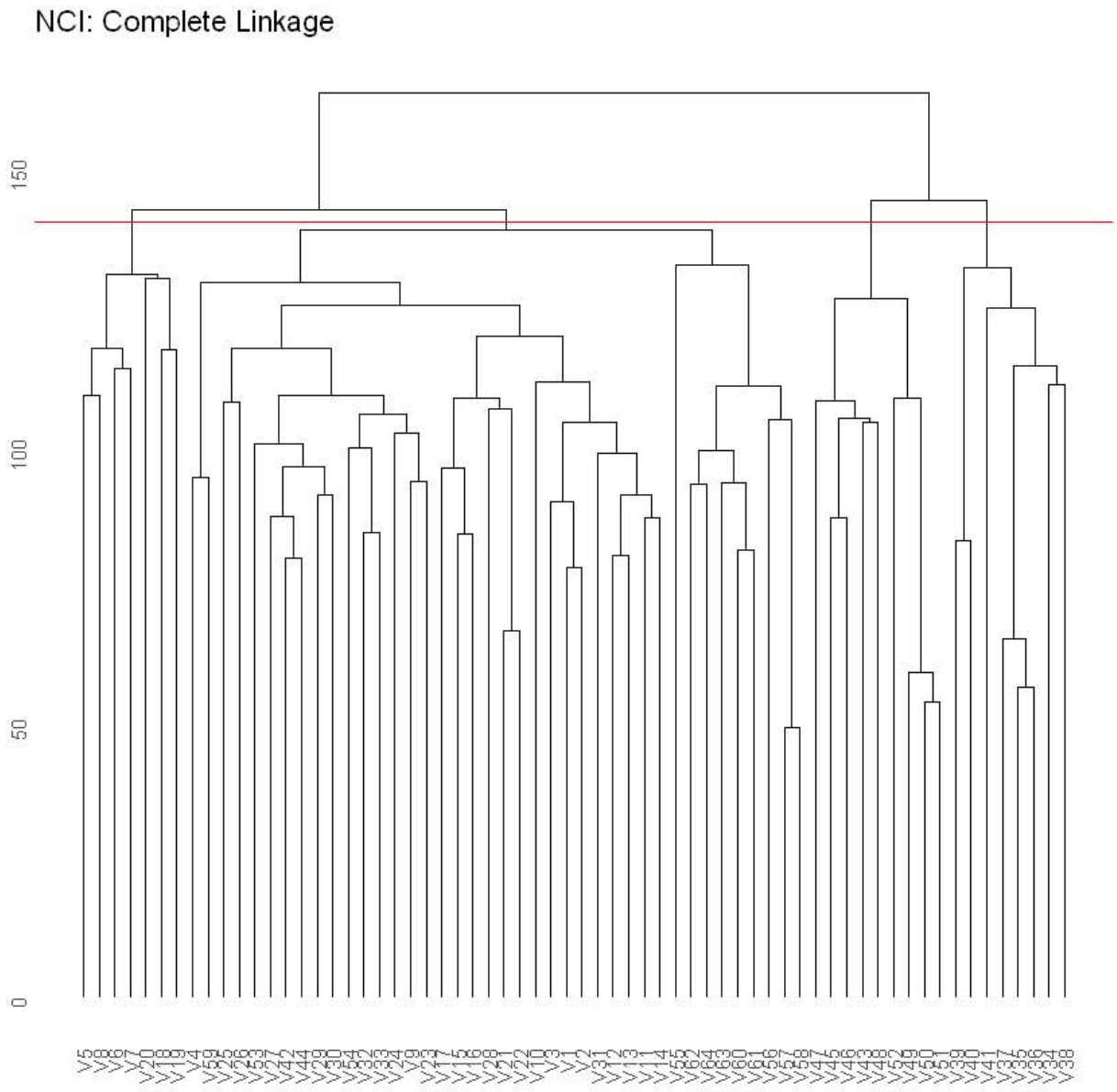
Let's use our complete linkage hierarchical clustering for the analysis. We can cut the dendrogram at the height that will yield a particular number of clusters, say 4:

In [510]:
```
hc_clusters = cutree(nci_hc_complete, 4)
table(nci_labels, hc_clusters)
```

```
             hc_clusters
nci_labels    1 2 3 4
  BREAST      2 3 0 2
  CNS         3 2 0 0
  COLON       2 0 0 5
  K562A-repro 0 0 1 0
  K562B-repro 0 0 1 0
  LEUKEMIA    0 0 6 0
  MCF7A-repro 0 0 0 1
  MCF7D-repro 0 0 0 1
  MELANOMA    8 0 0 0
  NSCLC       8 1 0 0
  OVARIAN     6 0 0 0
  PROSTATE    2 0 0 0
  RENAL       8 1 0 0
  UNKNOWN     1 0 0 0
```

There are some clear patterns. All the leukemia cell lines fall in cluster 3, while the breast cancer cell lines are spread out over three different clusters. We can plot the cut on the dendrogram that produces these four clusters by adding a geom_hline(), which draws a horizontal line on top of our plot:

In [511]:
```
ggdendrogram(nci_hc_complete, rotate = FALSE, size = 2) +
    labs(title = "NCI: Complete Linkage") +
    geom_hline(yintercept = 139, color = "red")

```



Printing the output of hclust gives a useful brief summary of the object:

In [512]:
```
nci_hc_complete
```

```
Call:
hclust(d = dist(scaled_nci_data), method = "complete")

Cluster method   : complete
Distance         : euclidean
Number of objects: 64
```

We claimed earlier that K-means clustering and hierarchical clustering with the dendrogram cut to

obtain the same number of clusters can yield very different results. How do these NCI60 hierarchical clustering results compare to what we get if we perform K-means clustering with K = 4?

In [513]:
```
set.seed(2)
km_out = kmeans(scaled_nci_data, 4, nstart = 20)
km_clusters = km_out$cluster
```

We can use a confusion matrix to compare the differences in how the two methods assigned observations to clusters:

In [514]:
```
table(km_clusters, hc_clusters)
```

```
           hc_clusters
km_clusters  1  2  3  4
          1 11  0  0  9
          2  0  0  8  0
          3  9  0  0  0
          4 20  7  0  0
```

We see that the four clusters obtained using hierarchical clustering and Kmeans clustering are somewhat different. Cluster 2 in K-means clustering is identical to cluster 3 in hierarchical clustering. However, the other clusters differ: for instance, cluster 4 in K-means clustering contains a portion of the observations assigned to cluster 1 by hierarchical clustering, as well as all of the observations assigned to cluster 2 by hierarchical clustering.