Raman Spectra of Chondrocytes in Cartilage: hyperSpec's `chondro` data set

Claudia Beleites <cbeleites@units.it>
CENMAT, DMRN, University of Trieste

January 26, 2011

Reproducing this vignette

The data set and source file of this vignette are available as a zipped archive at *hyperSpec*'s homepage, http://hyperspec.r-forge.r-project.org/chondrocytes.zip (ca. 9 MB). The original data file (ca. 31 MB) is far too large to be included in the package.
In order to reproduce the examples by typing in the commands, have a look at the definitions of color palettes used in this document are defined in `vignettes.defs`. Also, the package *hyperSpec* needs to be loaded first via `library (hyperSpec)`.

1 Introduction

This vignette describes the `chondro` data set. It shows a complete data analysis work flow on a Raman map demonstrating frequently needed preprocessing methods

- baseline correction
- normalization
- smoothing / interpolating spectra

and other basic work techniques

- plotting spectra
- plotting false color maps
- cutting the spectral range,
- selecting (extracting) or deleting spectra, and
- *aggregating* spectra (e.g. calculating cluster mean spectra).

The chemometric methods used are

- Principal Component Analysis (PCA) and
- hierarchical cluster analysis,

showing how to use data analysis procedures provided by R and other packages.

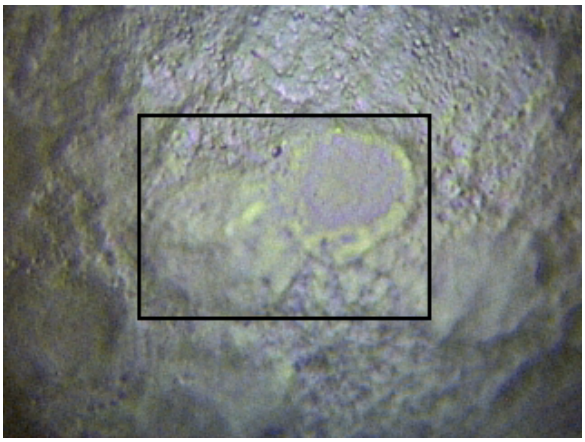


Figure 1: Microphotograph of the cartilage section. The frame indicates the measurement area (35 × 21 µm).

2 The Data Set

Raman spectra of a cartilage section were measured on each point of a grid, resulting in a so-called *Raman map*. Figure 1 shows a microscope picture of the measured area and its surroundings.

The measurement parameters were:

Excitation wavelength: 633 nm

Exposure time: 10 s per spectrum

Objective: 100×, NA 0.85

Measurement grid: 35 ×21 µm, 1 µm step size

Spectrometer: Renishaw InVia

3 Data Import

Renishaw provides a converter to export their proprietary data in a so-called long format ASCII file. Raman maps are exported having four columns, *y*, *x*, *raman shift*, and *intensity*. *hyperSpec* comes with a function to import such files, `scan.txt.Renishaw`. The function assumes a map as default, but can also handle single spectra (`data = "spc"`), time series (`data = "ts"`), and depth profiles (data = "depth"). In addition, large files may be processed in chunks. In order to speed up the reading `scan.txt.Renishaw` does not allow missing values, but it does work with `NA`.

```
> chondro <- scan.txt.Renishaw ("rawdata/chondro.txt", data = "xyspc")
> chondro

hyperSpec object
   875 spectra
   3 data columns
   1272 data points / spectrum
wavelength: Delta * tilde(nu)/cm^-1 [numeric] 601.62 602.66 ... 1802.2
data:  (875 rows x 3 columns)
   1. y: y/(mu * m) [numeric] -4.77 -4.77 ... 19.23
   2. x: x/(mu * m) [numeric] -11.55 -10.55 ... 22.45
   3. spc: I / a.u. [matrix1272] 501.72 518.53 ... 151.92 + NA
```

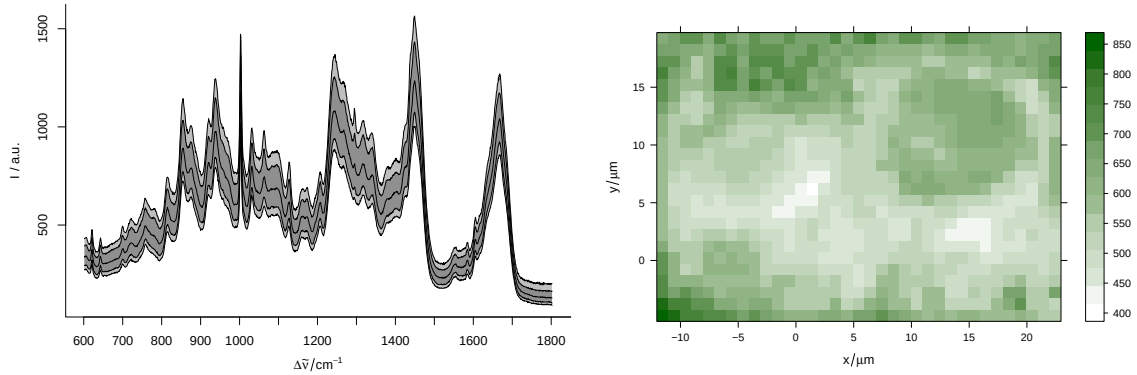
To get an overview of the spectra (figure 2a):

```
> plot (chondro, "spcprctl5")
```

A mean intensity map (figure 2b) is produced by:

```
> plotmap (chondro, func.args = list (na.rm = TRUE), col.regions = seq.palette (20))
```

`plotmap` applies a function to squeeze all spectral intensities into a summary characteristic for the whole spectrum. This function defaults to the `mean`. Further arguments that should be handed to this function can be given in list *func.args*. As the raw data contains `NA`s due to deleting cosmic ray spikes, this argument is needed here.



(a) The raw spectra: median, 16^{th} and 84^{th}, and 5^{th} and 95^{th} percentile spectra.

(b) [The sum intensity of the raw spectra.

Figure 2: The raw data.

4 Preprocessing

As usual in Raman spectroscopy of biological tissues, the spectra need some preprocessing.

4.1 Spectral Smoothing

As the overview print shows that the spectra contain `NA`s (from cosmic spike removal that was done previously), the first step is to remove these. Also, the wavelength axis of the raw spectra is not evenly spaced (the data points are between 0.85 and 1 cm^{-1} apart from each other). Furthermore, it would be good to trade some spectral resolution for higher signal to noise ratio. All three of these issues are tackled by interpolating and smoothing of the wavelength axis by `spc.loess`. The resolution is to be reduced to 8 cm^{-1}, or 4 cm^{-1} data point spacing.

```
> chondro <- spc.loess (chondro, seq (602, 1800, 4))
> chondro
```

```
hyperSpec object
   875 spectra
   3 data columns
   300 data points / spectrum
wavelength: Delta * tilde(nu)/cm^-1 [numeric] 602 606 ... 1798
data:  (875 rows x 3 columns)
   1. y: y/(mu * m) [numeric] -4.77 -4.77 ... 19.23
   2. x: x/(mu * m) [numeric] -11.55 -10.55 ... 22.45
   3. spc: I / a.u. [matrix300] 517.03 499.77 ... 168.04
```

The spectra are now the same as in the data set `chondro`. However, the data set also contains the clustering results (see at the very end of this document). They are stored for saving as the distributed demo data:

```
> spectra.to.save <- chondro
```

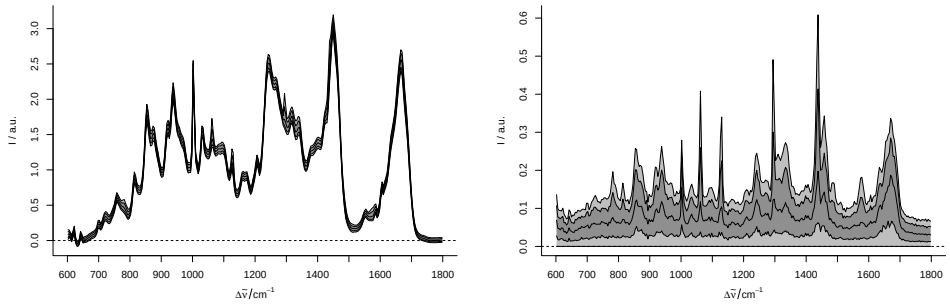
4.2 Baseline Correction

The next step is a linear baseline correction. `spc.fit.poly.below` tries to automatically find appropriate support points for polynomial baselines. The default is a linear baseline, which is appropriate in our case:

```
> baselines <- spc.fit.poly.below (chondro)

Fitting with npts.min =  15

> chondro <- chondro - baselines
```



(a) The spectra after smoothing, baseline correction, and normalization.

(b) The spectra after subtracting the 5^{th} percentile spectrum.

Figure 3: The preprocessed spectra.

4.3 Normalization

As the spectra are quite similar, area normalization should work well:.

```
> chondro <- sweep (chondro, 1, apply (chondro, 1, mean), "/")
> plot (chondro, "spcprctl5")
```

For the results of these preprocessing steps, see figure 3a.

4.4 Subtracting the Overall Composition

The spectra are very homogeneous, but I'm interested in the differences between the different regions of the sample. Subtracting the minimum spectrum cancels out the matrix composition that is common to all spectra. But the minimum spectrum also picks up a lot of noise. So instead, the 5^{th} percentile spectrum is subtracted:

```
> chondro <- sweep (chondro, 2, apply (chondro, 2, quantile, 0.05), "-")
> plot (chondro, "spcprctl5")
```

The resulting data set is shown in figure 3b. Some interesting differences start to show up: there are distinct lipid bands in some but not all of the spectra.

4.5 Outlier Removal by Principal Component Analysis (PCA)

PCA is a technique that decomposes the data into scores and loadings (virtual spectra). It is known to be quite sensitive to outliers. Thus, I use it for outlier detection. The resulting scores and loadings are put again into *hyperSpec* objects by `decomposition`:

```
> pca <- prcomp (~ spc, data = chondro$., center = TRUE)
> scores <- decomposition (chondro, pca$x, label.wavelength = "PC", label.spc = "score / a.u.")
> loadings <- decomposition (chondro, t(pca$rotation), scores = FALSE, label.spc = "loading I / a.u.")
```

Plotting the scores of each PC against all other gives a good idea where to look for outliers.

```
> pairs (scores [[,,1:20]], pch = 19, cex = 0.5)
```

Now the spectra can be found either by plotting two scores against each other (by `plot`) and identifying with `identify`, or they can be identified in the score map by `map.identify`. There is also a function to identify spectra in a spectra plot, `spc.identify`, but this is not helpful here.

```
> out <- map.identify (scores [,,5])
> out <- c (out, map.identify (scores [,,6]))
> out <- c (out, map.identify (scores [,,7]))

> out

[1] 105 140 216 289  75  69

> outcols <- c ("red", "blue", "#800080", "orange", "magenta", "brown")
> cols <- rep ("black", nrow(chondro))
> cols [out] <- outcols
```

We can check our findings by comparing the spectra to the bulk of spectra (figure):

```
> plot(chondro[1], plot.args = list (ylim = c (1, length (out) + .7)), lines.args = list(  type = "n"))
> for (i in seq (along = out)){
+    plot(chondro, "spcprctl5", yoffset = i, add = TRUE, col = "gray")
+    plot (chondro [out[i]], yoffset = i, col = outcols[i] , add = TRUE, lines.args = list (lwd = 2))
+    text (600, i + .33, out [i])  }
```

and also by looking where these spectra appear in the scores `pairs` plot (figure):

```
> png ("fig/fig-pca-pairs2.png", width = 1000, height = 1000)
> pairs (scores [[,,1:7]], pch = 19, cex = 1, col = cols)
> dev.off ()
```

Finally, the outliers are removed:

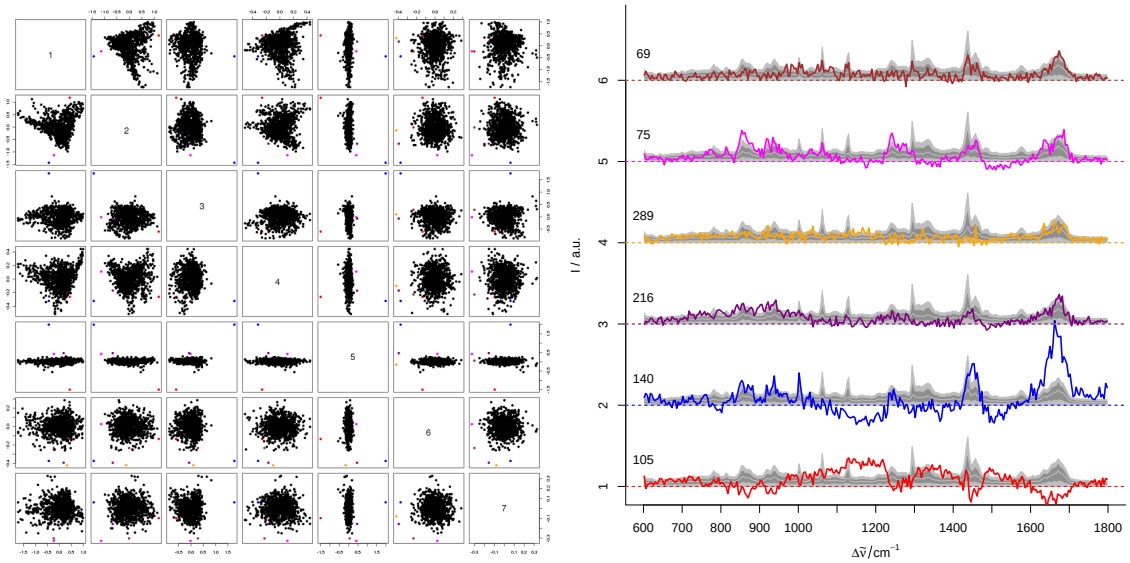
```
> chondro <- chondro [- out]
```

5 Hierarchical Cluster Analysis (HCA)

HCA fuses objects according to their (dis)similarity. The result is a dendrogram, a graph stating at which level two objects are similar and thus grouped together.

The first step in HCA is the choice of the distance. The R function `dist` offers a variety of distance measures to be computed. The so-called PEARSON distance $D^2_{Pearson} = \frac{1-COR(X)}{2}$ is popular in data analysis of vibrational spectra and is provided by *hyperSpec*.

Also for computing the dendrogram, a number of choices are available. I choose WARD's method, and, as it uses EUCLIDean distance for calculating the dendrogram, EUCLIDean distance also for the distance matrix :



(a) `pairs` plot of the first 7 scores.

(b) The suspected outlier spectra.

Figure 4: Outlier removal by PCA

```
> dist <- dist (chondro [[]])
> dendrogram <- hclust (dist, method = "ward")

> plot (dendrogram)
```

In order to get clusters, the dendrogram is cut at a level specified either by height or by the number of clusters.

```
> chondro$clusters <- as.factor (cutree (dendrogram, k = 3))
> cols <- c ("dark blue", "orange", "#C02020")
```

The result for k= 3 clusters is plotted as a map (figure 5b). If the color-coded variate (left hand side of the formula) is a factor, the legend bar does not show intermediate colors, and *hyperSpec*'s `levelplot` method uses the levels of the factor for the legend.

Thus meaningful names are assigned

```
> levels (chondro$clusters) <- c ("matrix", "lacuna", "cell")
```

and the cluster membership map is plotted:

```
> print (plotmap (chondro, clusters ~ x * y, col.regions = cols))
```

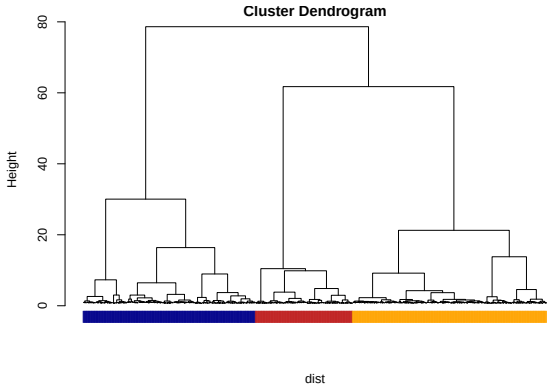
The cluster membership can also be marked in the dendrogram:

```
> plot (dendrogram, labels = FALSE, hang = 0)
> points (seq_along (dendrogram$order), rep (-3, length (dendrogram$order)),
+         col = cols [chondro$clusters [dendrogram$order]], pch = "|")
```

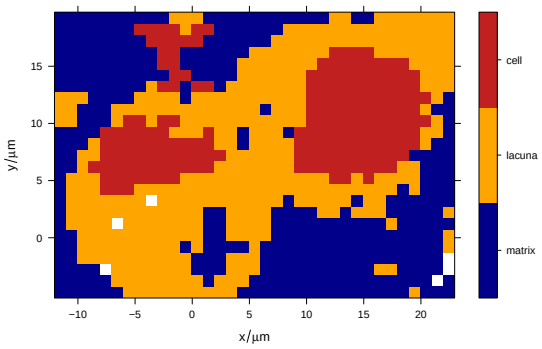
Figure 5a shows the dendrogram and 5b the resulting cluster map. The three clusters correspond to the cartilage matrix, the lacuna and the cells. The left cell is destroyed and its contents are leaking into the matrix, while the right cells looks intact.

We can calculate the cluster mean spectra using `aggregate`. However, we can do even better and plot the cluster mean spectra ± 1 standard deviation (see figure 6a):

```
> cluster.means <- aggregate (chondro, chondro$clusters, mean_pm_sd)
> plot(cluster.means, stacked = ".aggregate", fill = ".aggregate", col = cols)
```

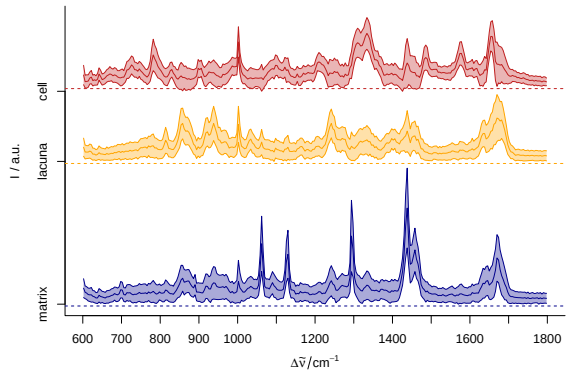


(a) The dendrogram.

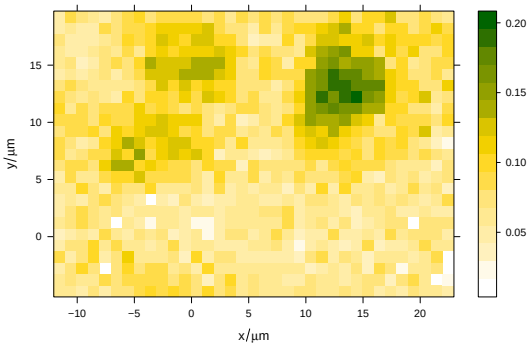


(b) The cluster map for $k = 3$ clusters.

Figure 5: Hierarchical cluster analysis.



(a) The cluster mean ± 1 standard deviation spectra. The blue cluster shows distinct lipid bands, the green cluster collagen, and the red cluster proteins and nucleic acids.



(b) False colour map of the DNA band intensities.

Figure 6

6 Plotting a False-Colour Map of Certain Spectral Regions

hyperSpec comes with a sophisticated interface for specifying spectral ranges. Expressing things like 1000 cm^{-1} ± 1 data points are easily possible. Thus, we can have a fast look at the nucleic acid distribution, using the DNA bands at 728, 782, 1098, 1240, 1482, and 1577 cm^{-1}:

```
> plotmap (chondro[, , c( 728, 782, 1098, 1240, 1482, 1577)],
+         col.regions = colorRampPalette (c("white", "gold", "dark green"), space = "Lab") (20))
```

The result is shown in figure 6b. While the nucleus of the right cell shows up nicely, nothing is detected in the remainders of the left cell.

7 Saving the data set

Finally, the example data set is put together and saved:

```
> spectra.to.save$clusters <- factor (NA, levels = levels (chondro$clusters))
> spectra.to.save$clusters[- out] <- chondro$clusters
> chondro <- spectra.to.save
> save (chondro, file = "chondro.rda")
```

This is the file distributed with *hyperSpec* as example data set.

Session Info

```
R version 2.12.1 (2010-12-16)
Platform: x86_64-pc-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.utf8       LC_NUMERIC=C              LC_TIME=en_US.utf8
 [4] LC_COLLATE=en_US.utf8     LC_MONETARY=C             LC_MESSAGES=en_US.utf8
 [7] LC_PAPER=en_US.utf8       LC_NAME=C                 LC_ADDRESS=C
[10] LC_TELEPHONE=C            LC_MEASUREMENT=en_US.utf8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] hyperSpec_0.96-20101125 lattice_0.19-17

loaded via a namespace (and not attached):
[1] grid_2.12.1
```