Mapping databases of X-ray powder patterns

by Ron Wehrens and Egon Willighagen

With the advent of high-throughput analysis methods, the number of large databases containing information on chemical compounds has increased rapidly. They are being used for many different purposes. In the pharmaceutical industry, for example, databases containing hundreds of thousands of drug-like compounds are being used to help identify promising new drug candidates. Obviously, the need for efficient data mining tools is increasing as well. It is especially difficult to assess how the separate objects in the database relate. Visualisation of the contents is of prime importance but not trivial; R offers a rich environment for this kind of exploratory analysis.

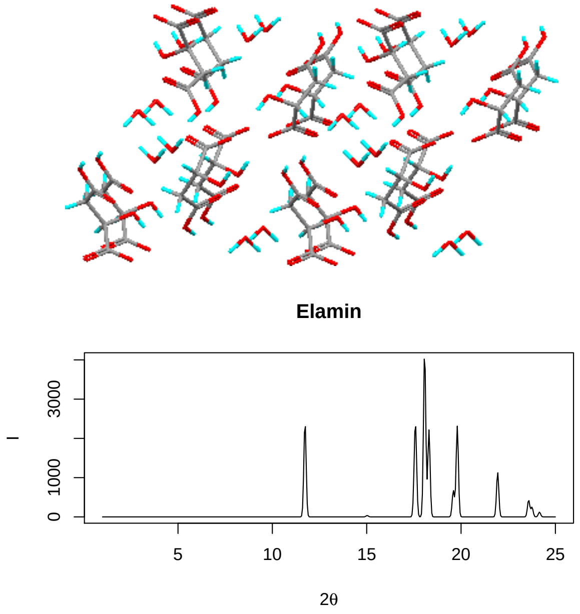

Figure 1: Crystal structure of ELAMIN (top) and associated powder pattern (bottom).

We show an example (Wehrens et al., 2005) of mapping part of the Cambridge Structural Database (CSD) (Allen, 2002) to a Kohonen Self-Organising Map (Kohonen, 2001). This database currently contains nearly 400,000 crystal structures. To assess the similarity of the crystal packing of these structures, many options are open. One can, e.g., compare the unit cell parameters (i.e. lengths of, and angles between, the three main axes that define the smallest replicated unit – the unit cell). Although this is easy and quick, there are several disadvantages to this approach. Apart from the fact that there is no unique definition of the unit cell, this representation does not capture information about the positions of the atoms within the cell. Therefore, we use X-ray powder diffractograms, which contain all packing information. In Figure 1, the crystal structure of compound ELAMIN is shown, together with the powder pattern. The shape of the unit cell determines peak positions, and the electron density within the unit cell determines peak heights.

Crucial is the similarity measure used to compare powder patterns: it must above all capture similarities in peak position, where it should be noted that the number of peaks in patterns of different compounds may be quite different. This is achieved by a measure called the weighted cross-correlation (WCC) (de Gelder et al., 2001). It takes one parameter, the triangle width, the maximal shift that will still give a positive contribution to the similarity when comparing peaks.

A self-organising map consists of a regular grid of units, where each unit has a "codebook vector", initially a random selection of patterns from the data set. During the training of the self-organising map, patterns are presented in random order. Each time, the unit that is most similar to the presented pattern is determined. The codebook vectors of this unit and of all other units within a certain neighbourhood are updated to become more similar to the presented pattern. Both the neighbourhood and the learning rate are decreased during training. At the end of the training, only the winning unit is updated; this, in fact, is a k-means clustering. The codebook vectors then can be interpreted as cluster centers. New objects can easily be projected in the map by comparing their patterns to all codebook vectors. Such a two-dimensional map is ideally suited for visual inspection and may show relations that otherwise may have gone unnoticed.

All this is implemented in package **wccsom**, available from CRAN. It is based on the `SOM` function by B.D. Ripley in package **class**, but has a number of extra features, most notably plotting functions, and the use of the WCC similarity function. The data used in this paper are proprietary (the CSD is a commercial database) but the package contains two smaller sets of powder patterns, provided by René de Gelder (Crystallography Department, Radboud University Nijmegen).

Example

For illustration purposes, we use a small dataset of 205 simulated powder patterns. Each pattern consists of 481 variables (2θ values). The set is constructured by searching the CSD for the compounds that are very similar to a set of twelve seed compounds; each group of compounds is considered to be a separate class. Powder patterns for three of the twelve classes are shown in Figure 2. ELAMIN is the seed compound of class 12. Obviously, not all classes are equally well defined: the spread in class three is much larger than in the other two.

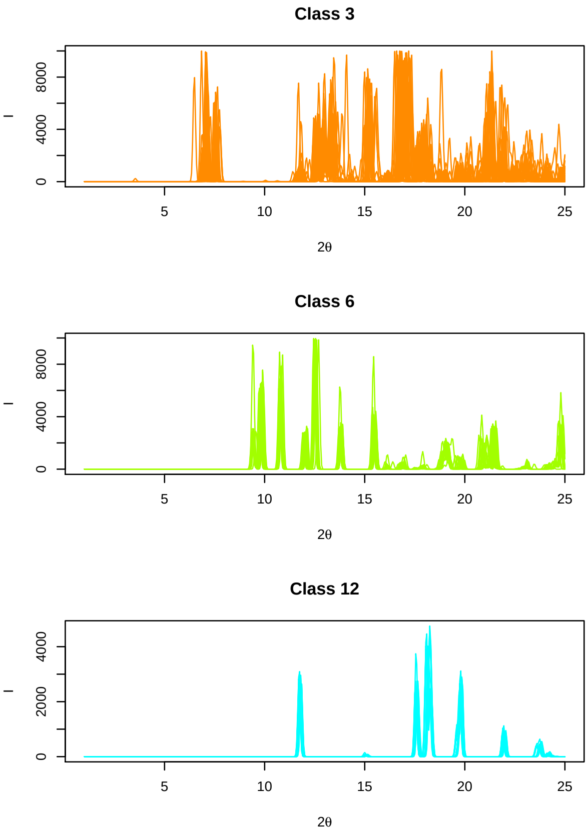

Figure 2: Powder patterns of classes 3, 6, and 12 (top to bottom).

The patterns are stored in a 205 x 481 matrix `testX`. They are mapped to a six-by-six Kohonen map by the following piece of code:

```
> library(wccsom)
> set.seed(7)
> somnet <- WCCSOM(testX,
+   somgrid(6, 6, "hexagonal"),
+   trwidth=20)
```

The triangle width for the WCC similarity function in this case is twenty points, corresponding to one degree 2θ. The convergence of the network can be assessed after training by plotting the mean change in similarity of the mapped object to the winning codebook vectors for each epoch (i.e. a complete presentation of the training set).

```
> plot(somnet, type="changes")
```

This is shown in Figure 3. In this case, the changes at the end of the training are very small; note that this is also the result of the decrease in learning parameter α. The training is followed by a k-means clustering, which essentially performs a fine-tuning of the network.

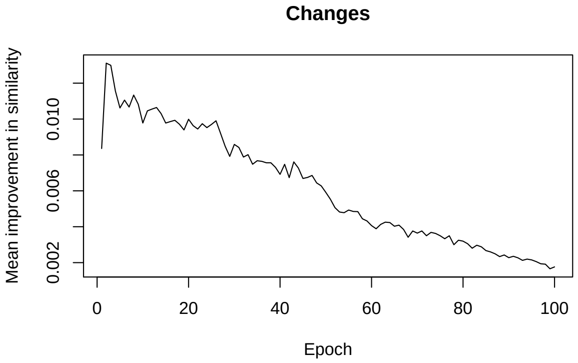

Figure 3: Convergence of network training: the plot shows the increase in similarity between the winning unit codebook vectors and the mapped objects during training.

Several other plots are available:

```
> mycols <- rainbow(12, start=0, end=.7)
> par(mfrow=c(2,2))
> plot(somnet, type="counts", main="Counts")
> plot(somnet, type="quality",
               main="Quality")
> plot(somnet, type="mapping",
               main="Mapping",
       labels=classes, col=mycols[classes])
> plot(somnet, type="codes", main="Codes")
```

This leads to the plots in Figure 4. The figure in the top left shows how many objects are mapped to each unit: gray indicates an empty unit. In the figure labelled "Quality", the mean similarity of objects to the corresponding codebook vector is indicated in colour; the blue lines give an indication of the standard deviations. A line pointing downwards indicates the minimal variation in this set; a line pointing upward the maximal. Units containing only one object are usually very similar to that object; no variation is indicated in such a case. Which type of object is mapped to what unit is shown in the figure at the bottom left: `classes` is just a class vector of length 205. With the exception of class three, which is structurally the most diverse class, all objects of one class are mapped to one unit or one group of adjacent units. Finally, the unit codes are shown in the bottom right plot. As one would expect, they look very much like powder patterns.

Since the classes are chosen to be well separated, this is not a difficult example. However, using Euclidean distances or correlations as (dis)similarity measures does not lead to a good mapping: the twelve classes are mixed up completely.

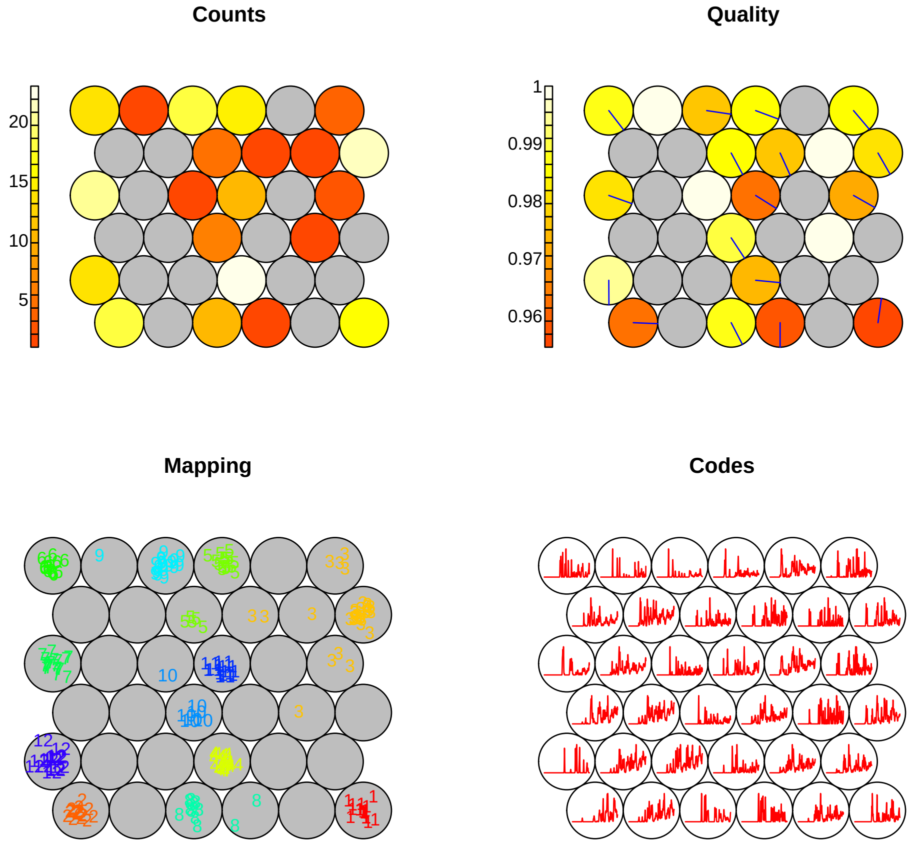

Figure 4: Several plotting types available for WCCSOM objects.

A more interesting case is presented by mapping all 2303 steroid-like structures present in the CSD (November 2004 release) to a 16x16 hexagonal grid. Apart from the plots shown earlier, the `summary` function can be used to get more information from the mapping. In particular, it is possible to get information on what objects are mapped to what unit, and how smooth the map is, i.e. how different the codebook vectors of neighbouring units are. Moreover, it is possible to summarize other information that may be present, such as reduced cell parameters. The seed structure for class 3 in the previous example, ECARAB, is also a steroid. It is mapped to unit 241 in the steroid map. We can check what other steroids are mapped there, and can compare their reduced cell parameters.

```
> set.seed(1)
> steroid.net <- WCCSOM(steroidX,
+   gr = somgrid(16, 16, "hexagonal"),
+   trwidth=20)
> options(digits=2)
> summary(steroid.net, type="unit",
+   nr = 241,
+   properties = steroid.props[,1:6])
Agreement with codebook vector of
25 objects mapped to cell 241:
         wccs   a   b   c alpha beta gamma
bifqai01 0.98 8.2 14 14   114   90    90
ecarab   0.99 8.1 14 14   114   90    90
eripuq   0.97 8.3 14 14   114   90    90
eripuq01 0.98 8.3 14 14   114   90    90
...
zuzdon   0.99 8.1 14 14   114   90    90
zuzdut   0.97 8.1 14 14   116   90    90
zuzfab   0.95 8.1 14 14   116   90    90
```

All compounds mapped to this unit have very similar reduced cell parameters and very high similarities to the codebook vector. Class 5 of the test data set, which is also a steroid class but with a different symmetry (axis lengths of 7.2, 13.6, and 25.6 Å, respectively, and all angles α, β and γ equal to 90 degrees), is mapped in a totally different area of the map.

Although the compounds from the small test dataset are not all steroids, they still can be mapped to the steroid map:

```
> classif <- wccassign(steroid.net, testX)
> par(mfrow=c(2,1))
> plot(steroid.net, "mapping",
+      classif=classif,
+      labels=classes, col=mycols[classes])
> plot(steroid.net, "quality",
+      classif=classif)
```

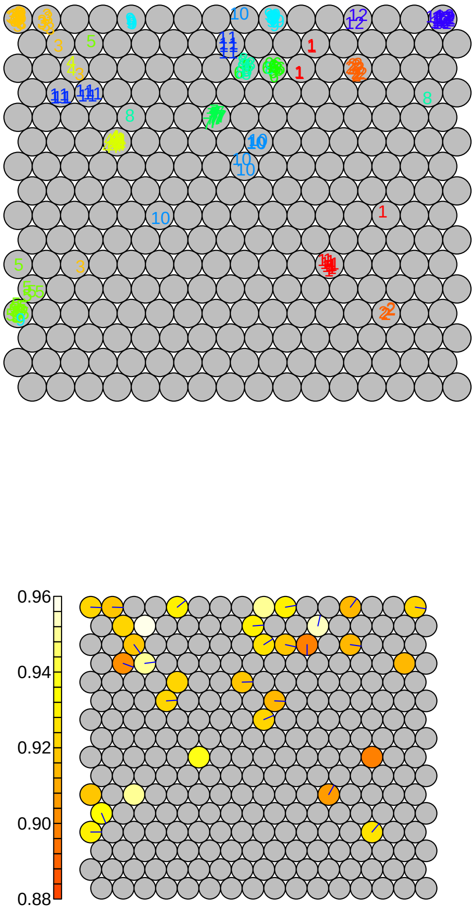

Figure 5: Mapping the test data set to the steroid map. Bottom figure: quality plot.

This leads to the plot in Figure 5. There is considerably more spread, mainly because of the much larger variability in powder patterns in the steroid data set. Note that the mapping quality is also lower.

Upscaling

Training the steroid data set takes roughly 45 minutes on a 2.0 GHz AMD Opteron processor. Obviously, training a network for all structures in the CSD presents problems regarding memory as well as calculation time. However, one of the advantages of self-organising maps is that one can train in steps: first use a subset of the complete database for initial training, and then gradually add new patterns until all have been presented to the net several times. An added advantage is that one can easily update the map when new releases of the database become available. Care must be taken, however, not to destroy other information in the network.

Still, it is a challenging task, and several tricks should be applied to increase the speed of training. One obvious improvement is to use a growing network instead of a decreasing neighbourhood; another is to do the bulk of the training with fewer variables. For this, functions `expand.som` and `bucket/debucket` are available in package **wccsom**. Note that the triangle width for the WCC function should be adapted accordingly. In the following example, we first decrease the resolution of the powder patterns by a factor of five, and train a small four-by-four network. This network is expanded twice so that a similar-sized network as in the above example is obtained.

```
> steroid.Xsm <- bucket(steroidX, 5)
> somnet <- WCCSOM(steroid.Xsm,
+   gr=somgrid(4, 4, "hexagonal"),
+   rlen=20, radius=4, trwidth=4)
> for (i in 1:2) {
+   map.exp <- expand.som(somnet)
+   somnet <- WCCSOM(X, gr=map.exp$grid,
+     rlen=20, radius=4,
+     nhbrdist = map.exp$nhbrdist,
+     init = t(map.exp$codes))
+ }
```

Finally, the original resolution is restored (function `debucket` uses linear interpolation), and a few training epochs are used to fine-tune the codebook vectors. The results are comparable to the original map.

```
> sternet.codes <- debucket(t(somnet$codes),
+                           ncol(steroidX))
> somnet.final <- WCCSOM(steroidX,
+   gr = somnet$grid, rlen=20,
+   radius=4, nhbrdist = map.exp$nhbrdist,
+   init = sternet.codes)
> options(digits = 2)
> ecarabindex <-
+   which(dimnames(steroidX)[[1]] ==
+         "ecarab")
> summary(somnet.final, type="object",
+   nr = ecarabindex,
+   properties = steroid.props[,1:6])
Agreement of object ecarab with
23 other objects mapped to cell 6:
         wccs   a  b  c alpha beta gamma
bifqai01 0.98 8.2 14 14   114   90    90
eripuq   0.96 8.3 14 14   114   90    90
eripuq01 0.97 8.3 14 14   114   90    90
...
zuzdon   0.99 8.2 14 14   114   90    90
zuzdut   0.97 8.1 14 14   116   90    90
zuzfab   0.95 8.1 14 14   116   90    90
```

Again, all compounds mapped to this unit share the same reduced cell parameters. Indeed, the set of compounds mapped to this unit is almost identical to the one found earlier. By this procedure, training time could be reduced by fifty percent. For larger data sets and, especially, larger nets, speed improvements are even bigger.

One further potential improvement that we have investigated but so far have been less succesful with, is the use of stick patterns, rather than semi-continuous patterns consisting of equidistant points. Not only will this use less memory (on average 40 – 100 peaks are present in one pattern), but the calculation of the similarity function is faster, too. Moreover, it allows for easy selection of the most prominent features only, again decreasing computational requirements. However, the crucial updating step during training so far has proved difficult to define.

Applications

The trained map may be used to visualize the contents of the database in a variety of ways. For individual compounds, one can show areas that contain similar crystal packings. Pair-wise comparisons need not to be performed for the whole database, which is computationally very expensive: one can concentrate on the codebook vectors of the map. All compounds that map to similar units are candidates for further investigation.

A particular advantage of using semi-continuous powder patterns is that experimental patterns can directly be mapped. No peak picking is required. One can even assign chemical meaning to some characteristics of the unit vectors. If many peaks are present in the low 2θ ranges, it means that the cell volume is quite large. For other forms of spectroscopy, some peaks may even be interpreted directly. All this may aid in the elicudation of structure parameters of unknown compounds.

One can also use the map as a means of stratified sampling: a small set of compounds (e.g. one for each unit) can be selected that covers the complete chemical space. In polymorph prediction, this feature can be used to reduce the number of structures before (expensive) energy minimisation.

As a final example, trained maps can be used for database quality control. Although many steps in the sequence from measuring data to storing the results in a database can be automated and in fact are, there is still more than enough opportunity for error. Compounds with either very unrealistic powder patterns or properties very dissimilar to compounds mapped to the same units are candidates for futher investigation.

In conclusion, the examples in this paper show how the versatility of the R environment can contribute to a better understanding of the connections in large databases of molecules. The necessary speed is obtained by using compiled code; the trained maps are stored as R objects which can easily be interrogated, even by inexperienced R users, and on which new objects can easily be mapped. One could even envisage a web-based application similar to examples mentioned in (Kohonen, 2001).

Acknowledgements

René de Gelder is acknowledged for his crystallographic input; Willem Melssen for providing expertise on self-organising maps.

Bibliography

F. H. Allen. The Cambridge Structural Database: a quarter of a million crystal structures and rising. *Acta Crystallogr.*, B58:380–388, 2002. 24

R. de Gelder, R. Wehrens, and J. A. Hageman. A generalized expression for the similarity spectra: application to powder diffraction pattern classification. *J. Comput. Chem.*, 22(3):273–289, 2001. 24

T. Kohonen. *Self-Organizing Maps*. Number 30 in Springer Series in Information Sciences. Springer, Berlin, 3 edition, 2001. 24, 28

R. Wehrens, W.J. Melssen, L. Buydens, and R. de Gelder. Representing structural databases in a self-organizing map. *Acta Cryst.*, B61:548–557, 2005. 24

Institute for Molecules and Materials
Analytical Chemistry
The Netherlands
R.Wehrens,e.willighagen@science.ru.nl

Generating, Using and Visualizing Molecular Information in R

by Rajarshi Guha

Introduction

R, as a statistical computing environment, has a wide variety of functionality that makes it suitable for modeling purposes in a number of fields. In the case of cheminformatics we are often presented with a large amount of information, from which chemical meaning and insight must be extracted, using computational tools. In many cases, problems in chem-