

Analysing place cells

Balazs B Ujfalussy

20/11/2017

This is a demo for illustrating place cells. We will analyse a 30 min recording session from a rat running back and forth in a 1.6 m long linear track, from the laboratory of Gyuri Buzsáki. There are about 120 pyramidal cells recorded simultaneously, half of them are place cells. There are also 20 interneurons.

There are two exercises related to the dataset:

- analysing the coding properties of place cells - how reliable they are? How much information they contain?
- decoding the activity of place cells - can you use the techniques learned in the course to predict the position of the rat from the spike counts?

The source of the data: Groszmark, A.D., Long J. and Buzsáki, G (2016). Recordings from hippocampal area CA1, PRE, during and POST novel spatial learning. CRCNS.org <http://dx.doi.org/10.6080/K0862DC5>

Paper related to the dataset: Groszmark, A.D., and Buzsáki, G. (2016). Diversity in neural firing dynamics supports both rigid and learned hippocampal sequences. Science 351, 1440–1443.

Load the dataset and observe place cells

We will read a preprocessed data file ‘Achilles.RData’ - Achilles is the name of the rat. It is a list storing different variables describing the experiment.

```
load('./Achilles.RData')
```

The `summary()` function tells you what variables are encoded into the list `rat`:

```
summary(rat)
```

```
##           Length Class  Mode
## pos          161524 -none- numeric
## spt          1731934 -none- numeric
## iruns.up           84 -none- numeric
## iruns.down        84 -none- numeric
## PyrIDs           120 -none- numeric
## IntIDs            17 -none- numeric
## MazeRange         2 -none- numeric
```

- The main variables encoded are the position (`pos`), spike times (`spt`). The position is a matrix of two columns: time in seconds and smoothed 1D position of the animal along the linear track. The `spt` is also a two column matrix, time in seconds and the ID of the cell that emitted the spike. (Cell IDs refer to the electrodes the cell was recorded from, so they do not start from 1...)
- Before and after each run the animal stays at the end for a while to consume reward. Up and down runs (left and right, sorry :-)) are associated with different neuronal activity, so they are treated differently. The variable `iruns.down` and `iruns.up` stores the start and the end of the individual runs indexing the rows of the matrix `pos`.
- The variable `PyrIDs` and `IntIDs` stores the name of the (putative) pyramidal cells and interneurons.
- `MazeRange` is the x coordinates associated with the start and the end of each run.

The individual variables in the list can be referred by the \$ sign. For example the dimensionality of the position variable can be prompted as `dim(rat$pos)` which returns 80762, 2 meaning that this matrix has 80762 rows and 2 columns.

Now we will load a function that will analyse this data to return the spike counts for each cell on each runs in the function of the (discretised) position.

```
source("PlaceCellFunctions.R")
require(viridis)
```

```
## Loading required package: viridis
```

```
## Loading required package: viridisLite
```

```
pos <- rat$pos
spt <- rat$spt
```

We define the spatial discretization in 5 cm.

```
dx <- 0.05 # cm, resolution
x.breaks <- seq(rat$MazeRange[1], rat$MazeRange[2], by=dx)
x.mids <- round(x.breaks[-1] - dx/2, 3)
```

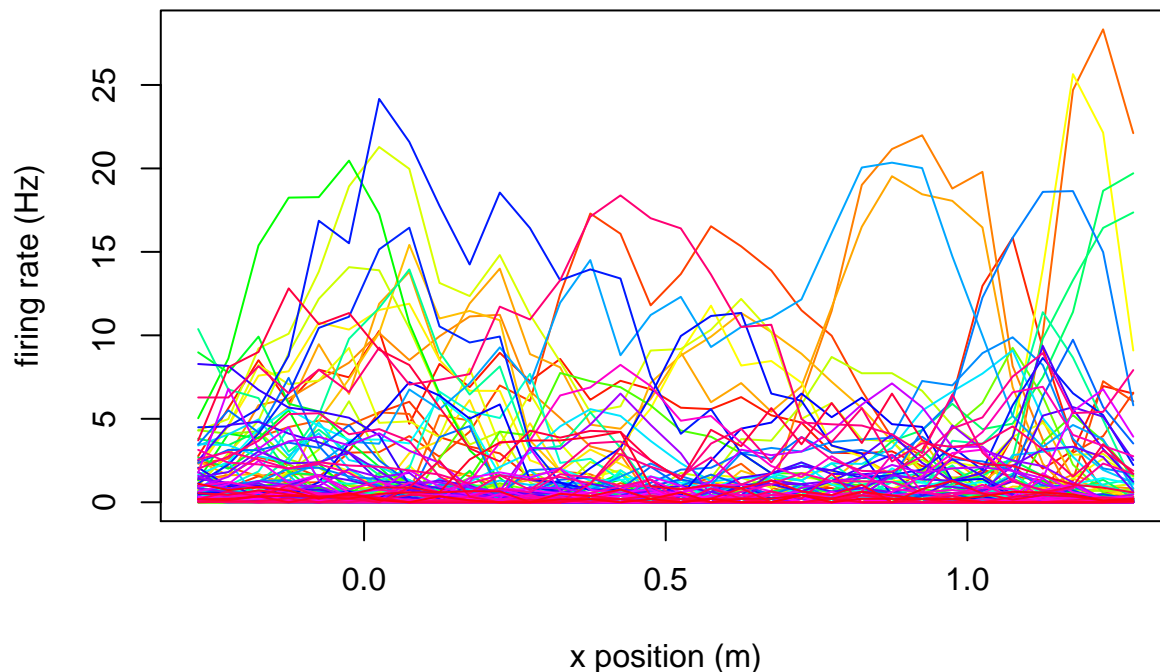
`act.runs` is an array with the spikes of each cell on each trial with spatial resolution `dx`. Its dimensions are {number of cells} x {distance} x {trials}. The last neuron is not a true neuron, but stores the time (in seconds) the rat spent at each location at each trial. We only analyse the up runs here.

```
act.runs <- cell.act.runs(spt, pos, i.runs=rat$iruns.up, dx=0.05, MazeRange=rat$MazeRange, cell.IDs=rat$cell.IDs)
```

We divide spike count with the occupancy time to get firing rates, and plot the firing rate of all cells in the function of distance.

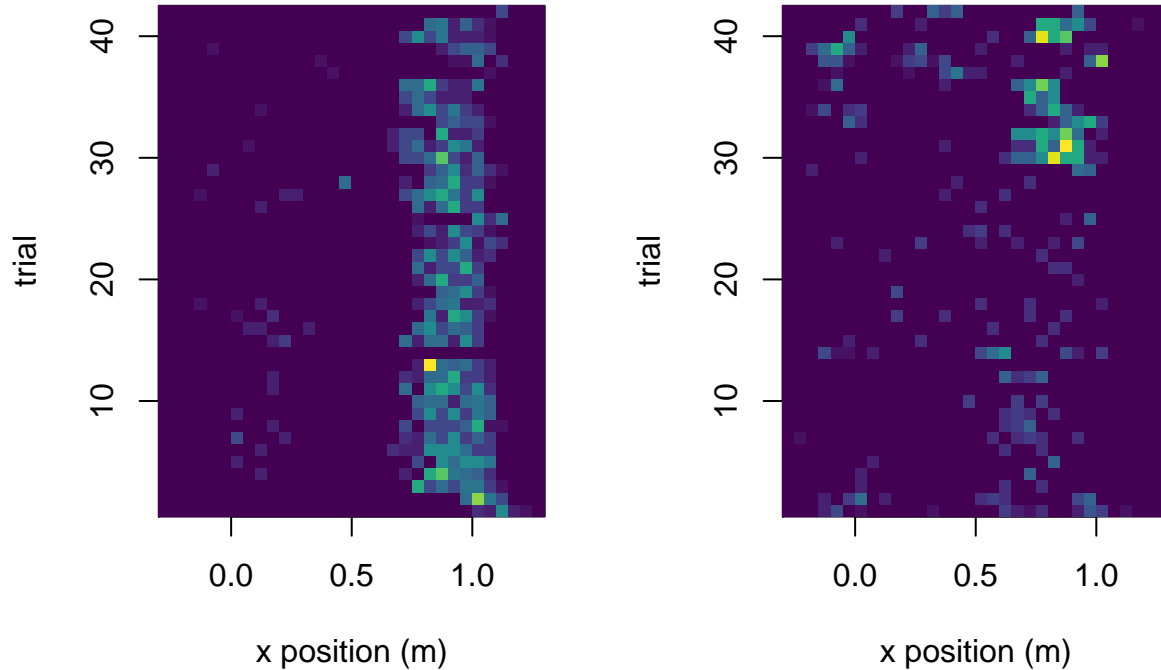
```
ratemaps.t <- apply(act.runs[1:120,,], c(1,2), sum)
Tmap <- apply(act.runs[121,,], 1, sum)
```

```
ratemaps.all <- t(ratemaps.t) / Tmap
matplot(x.mids, ratemaps.all, t='l', lty=1, col=rainbow(120), xlab='x position (m)', ylab='firing rate (Hz)')
```



Next, plot firing rates of two example neurons estimated on individual runs - we still need to divide with occupancy time!

```
par(mfcol=c(1,2))
image(x.mids, 1:42, act.runs[11,,] / act.runs[121,,], col=viridis(24), xlab='x position (m)', ylab='trial')
image(x.mids, 1:42, act.runs[26,,] / act.runs[121,,], col=viridis(24), xlab='x position (m)', ylab='trial')
```

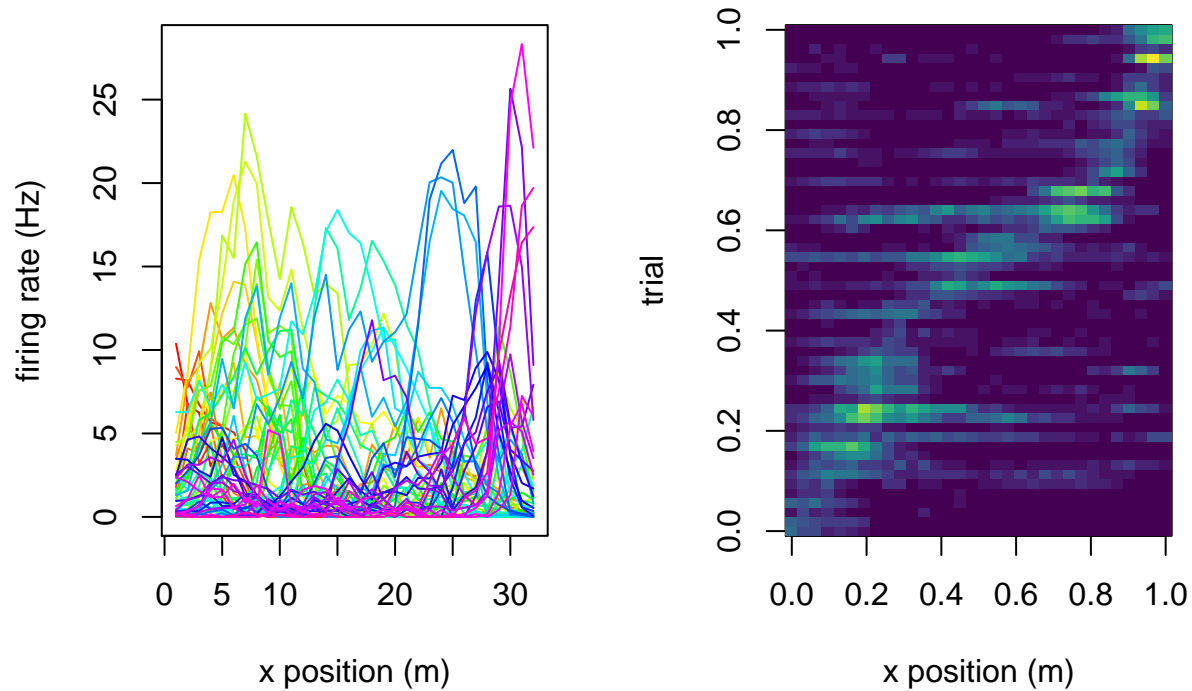


The

left cell looks like a classical place cell, the right starts to fire only after the 30th run.

Finally plot the ratemap of all active cells - sorted according to the position of the peak.

```
i.cells.active <- which(apply(ratemaps.all, 2, max) > 5)
ratemaps <- ratemaps.all[,i.cells.active]
ii.maxs <- apply(ratemaps, 2, which.max)
sort.peaks <- sort(ii.maxs, ind=T)$ix
par(mfcol=c(1,2))
matplot(ratemaps[,sort.peaks], t='l', lty=1, col=rainbow(60), xlab='x position (m)', ylab='firing rate')
image(ratemaps[,sort.peaks], col=viridis(24), xlab='x position (m)', ylab='trial')
```



Now we will do static decoding of place cell activity. We prepare two datasets, a training and a test dataset, each containing \mathbf{s} spike counts of the 54 active cells and the location of the animal x . We can learn the relationship between \mathbf{s} and x using the training data. Then we can predict the position x from the spikes \mathbf{s} .

Homework 1 - decoding

Predict the position x from the spikes \mathbf{s} in the test set - `test.spikes`.

Homework 2 - reliability

Calculate the information rate (bits/spike) for all place cells. Is this a useful measure? Does it reflect the reliability of the cells? What else would you use to discriminate proper place cells and cells that fired only a couple of times at a given location?