Analysing place cells

Balazs B Ujfalussy

20/11/2017

This is a demo for illustrating the activity of place cells. We will analyse a 30 min recording session from a rat running back and forth in a 1.6 m ling linear track, from the laboratory of Gyuri Buzsaki. There are about 120 pyramidal cells recorded simulataneously, half of them have place field in this environment. There are also 20 interneurons.

There are two exercices 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 predict the position of the rat from the spike counts of the cells?

The source of the data: Grosmark, 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: Grosmark, 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 contains a list named `rat` that stores different variables describing the experiment.

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

The `summary()` function tells you what variables are encoded int 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 columnn matrix, time in seconds and the ID of the cell that emitted the spike. (Cell id's refer to the elctrodes 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. The activity of the place cells can depend on the direction of the movement, so up and down runs (left and right, sorry :-)) are treated differently. The variable `iruns.down` and `iruns.up` stores the index of the start and the end of the individual down or up runs indexing the rows of the matrix `pos`.
- The variable `PyrIDs` and `IntIDs` stores the name of the (putative) pyramidal cells and interneurons.
- `MazeRange` stores the x coordinates associated with the start and the end of the runs.

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")
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)
```

Next we prepare an array named `act.runs` that contains 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, but one of the homeworks is to compare the place fields in the two directions. The function `cell.maps.runs` extracts the relevant data for us.

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

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
```

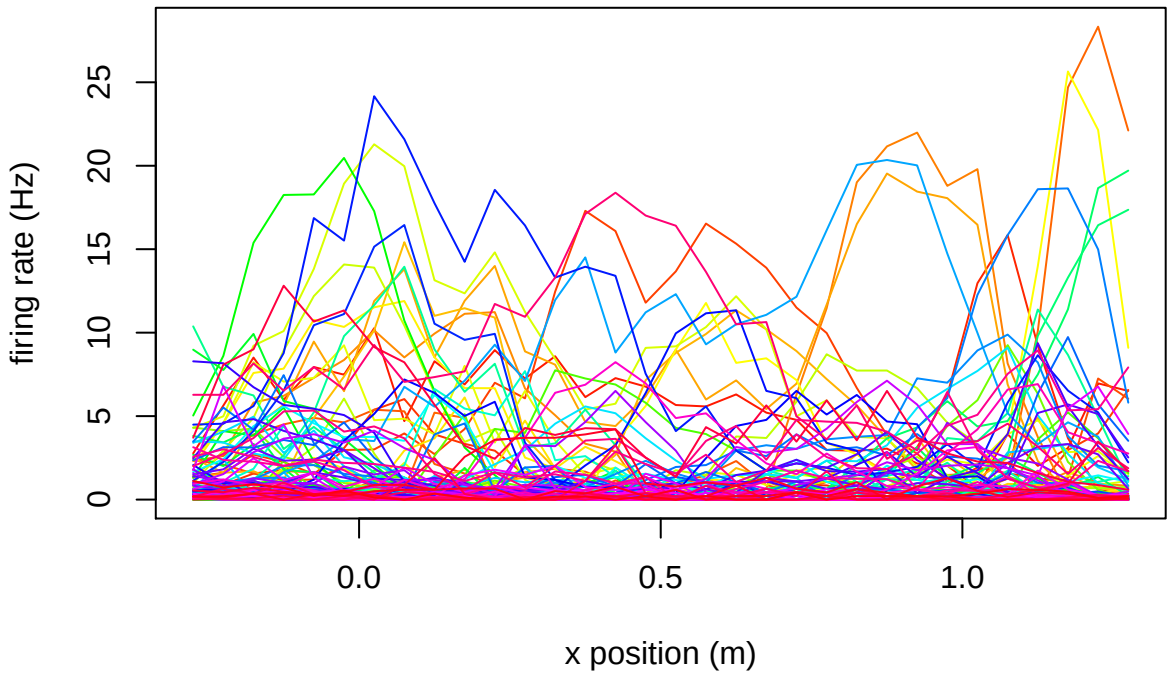

Figure 1: Ratemap of all pyramida cells.

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)); par(mar=c(4,4,4,4))
image(x.mids, 1:42, act.runs[70,,] / act.runs[121,,], col=topo.colors(24), xlab='x position (m)', ylab=
lines(x.mids, ratemaps.all[,70], col=heat.colors(3)[2], lwd=2)
axis(4, c(0, 5, 10), c(0, 5, 10))
mtext('firing rate (Hz)', 4, 2, adj=0)
image(x.mids, 1:42, act.runs[97,,] / act.runs[121,,], col=topo.colors(24), xlab='x position (m)', ylab=
lines(x.mids, ratemaps.all[,97], col=heat.colors(3)[2], lwd=2)
axis(4, c(0, 5, 10), c(0, 5, 10))
mtext('firing rate (Hz)', 4, 2, adj=0)
```

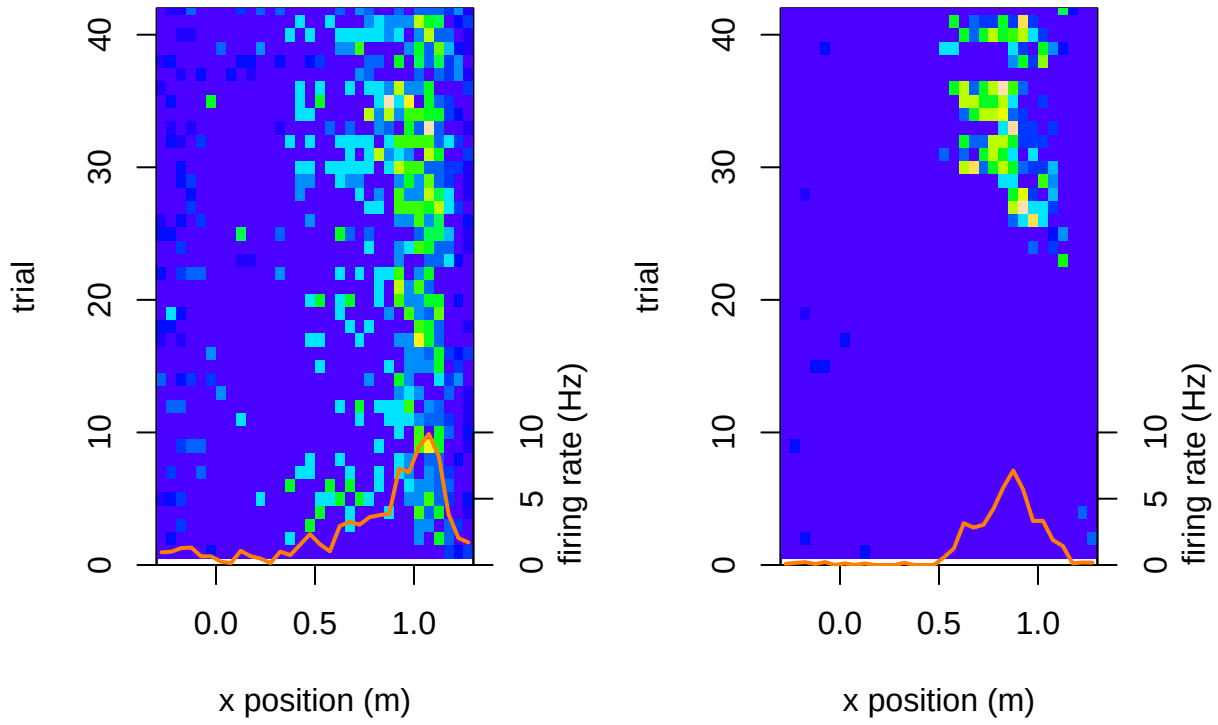

Figure 2: Firing rates on individual trials for two neurons. The left cell looks like a classical place cell, the right starts to fire only after the 30th run.

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)
N.cells.active <- length(i.cells.active)
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(x.mids, ratemaps[,sort.peaks], t='l', lty=1, col=rainbow(60), xlab='x position (m)', ylab='firi
image(x.mids, 1:N.cells.active, ratemaps[,sort.peaks], col=topo.colors(24), xlab='x position (m)', ylab=
```

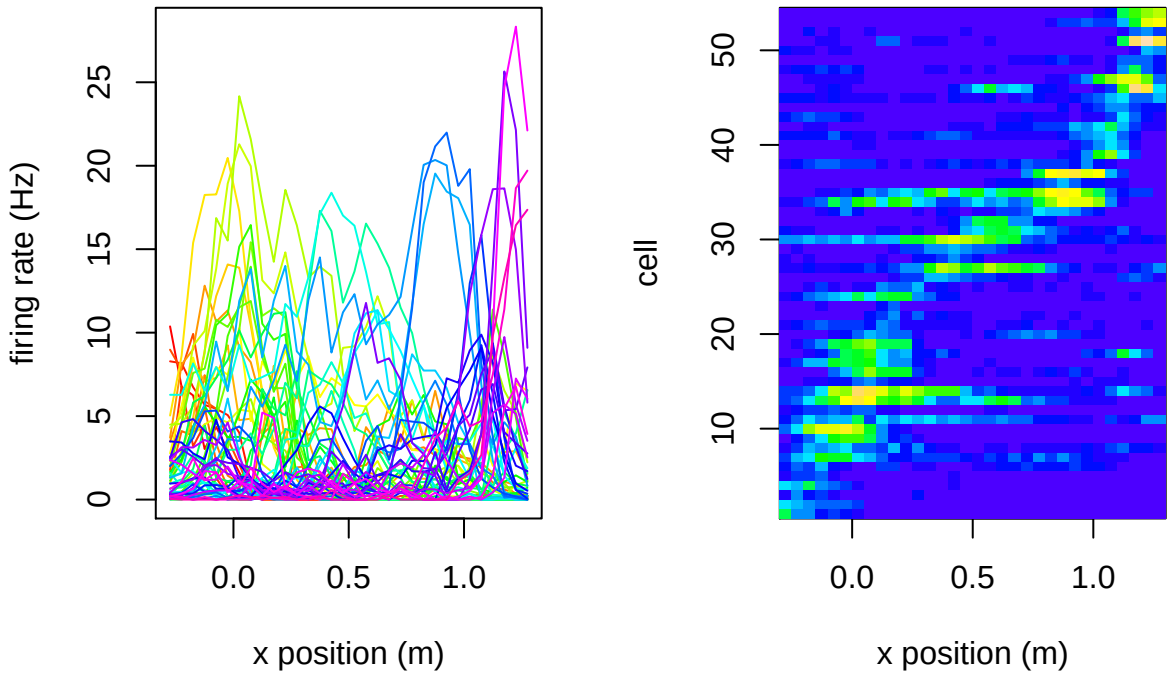

Figure 3: Place cells sorted according to their peak firing rate

Homeworks

1. Reliability of the place cells. A place cell is useful if it always works, i.e., it fires action potential whenever the animal is at a certain location. If it only signals at half of the times, it seems to be less informative. Look at the place cells of Figure 2. Which one is a more useful as a place cell? [2p]
2. Look at the activity of other place cells on individual runs. Are they reliable indicator of the animal's location? Can you find examples of reliable and unreliable cells? Can you identify other classes? [4p]
3. Can you come up with a simple metric that measures the reliability of a place cell? Show how it works on two example place cells! [6p]
4. Calculate the reliability metric for the entire population of place cells and investigate whether it correlates with our intuitive sense of reliability! You can get the full points even if the metric does not work, but try to understand why it fails! [4p]
5. Calculate the place fields of the same place cells on the opposite runs. Compare the location of the place fields in the two direction. Can you find cells with the activity truely depending on the position and independent of the running direction? Can you find cells that are sensitive to the running direction? [4p]