Predicting Movement from Nueron Spikes

Andrew Resnikoff

Introduction

Brains are composed of neurons, which send electrical signals to communicate with other cells and control how an organism behaves. These signals are called "spikes."

Through this investigation, we aim to better understand the relationship between the number of spikes that occur in a short time period, a specific action and the neurons associated with that action.

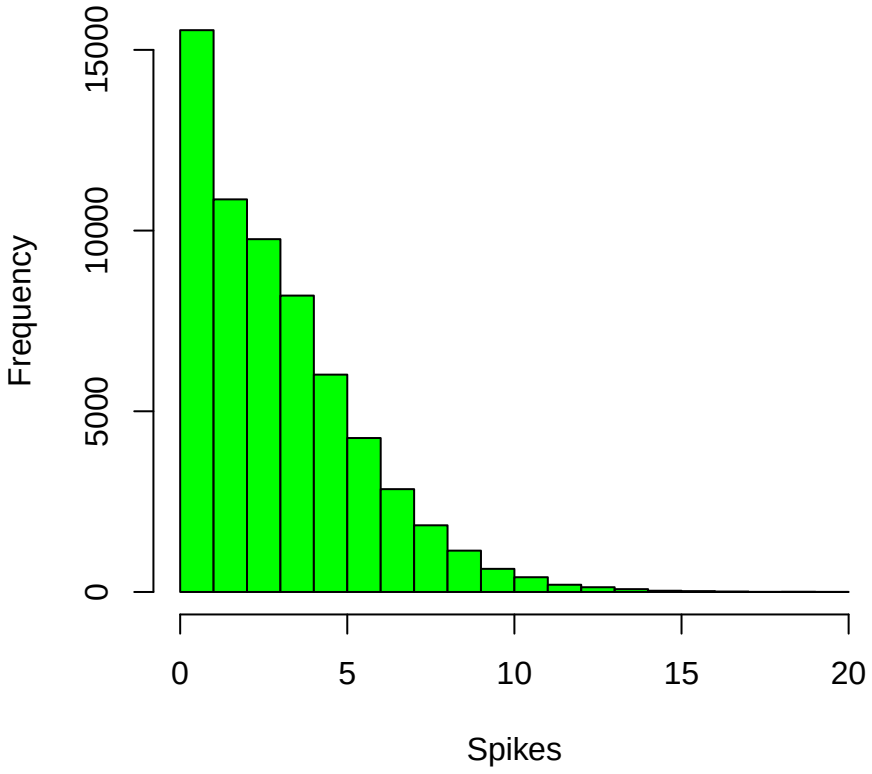
Data

We are given a data set of 646 observations from an experiment. Each observation corresponds to a 100ms time period and is a row in the data set. Each row has 96 different entries, each corresponding to a neuron. Each entry is the number of spikes in that time period for that specific neuron.

There are no missing entries in the data. The summary statistics for the data are printed below.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   3.000   3.409   5.000  20.000
```

Histograms of the number of spikes is shown below.



The distribution of the number of spikes (in general) is skewed right with a center around 3. This distribution ranges from 0 to 20.

If we look at the distribution of the number of spikes for random neurons or random observations, we see that the spike distribution can vary.

Data Analysis

We are given that the average number of spikes over a short interval is given by

$$\vec{b} \cdot \vec{v} + a + \text{noise}$$

This model is related to a factor model with 2 factors, where each factor is one of the x and y component of hand movement velocity vector.

$\vec{b}$ is the preferred direction vector of a specific neuron. We can place this vector for each neuron together in a matrix, called B. Since there are 2 possible directions of movement, the preferred direction vector should have 2 entries. Thus, this matrix would be either 2×96 or 96×2.

$\vec{v}$ is the direction a monkey intends to move its hand. There are 2 components of the velocity and in each observation the monkey is moving its hand. If we put together a matrix V of the monkey's hand movement vector for each 100ms observation, we would have either a 646×2 or 2×646 matrix.

a is a constant baseline number of spikes for each neuron. This number is added to $\vec{b} \cdot \vec{v}$, which would be the number of spikes generated from movement to get the total number of spikes in the 100ms time period. We can put the a value in a length 96 vector representing the baseline for all of the neurons and then create a matrix where there is a row for each observation and each row is the vector of baseline values. Call this matrix A.

Notice we can now put together a full model for our data. Let's call our response X and our noise matrix ϵ.

$$X = VB + A + \epsilon$$

We say this model is related to a factor model because of the A matrix. Since this is just a constant matrix, we can subtract this from our response and fit a factor model to the result. Then, we can simply add the A matrix back in. Letting $X' = X - A$, we have

$$X' = VB + \epsilon$$

V is the matrix of factor scores, as the velocity the monkey moves its hands are a hidden random variable.

B is the matrix of factor loadings, as the preferred direction vector will determine how the velocity will affect each neuron.

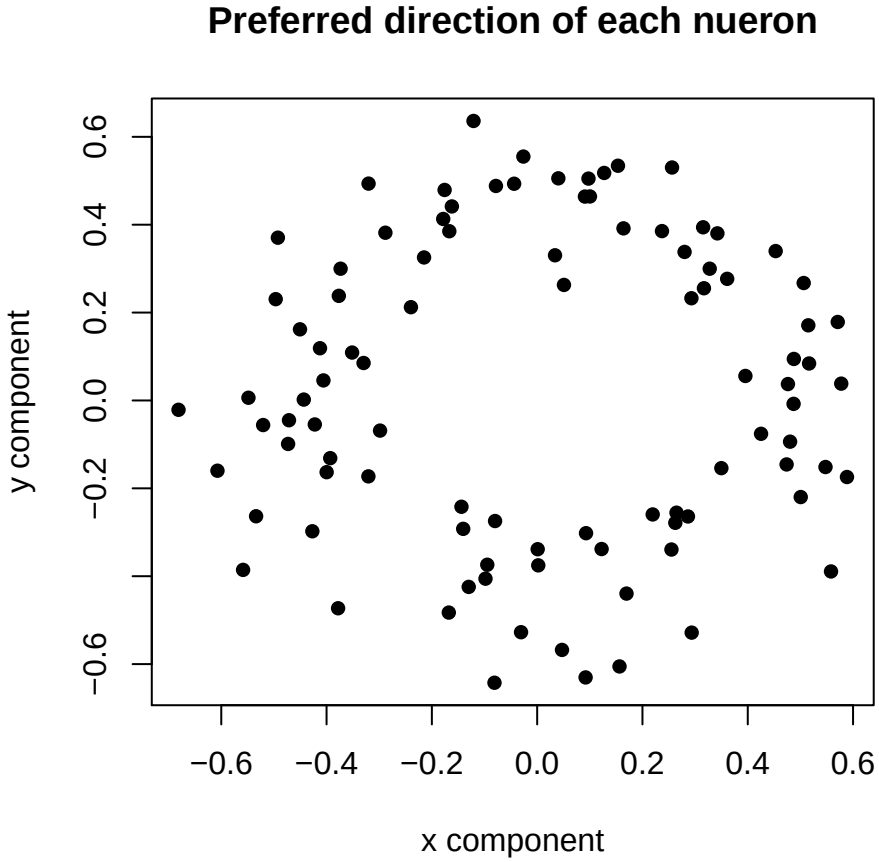
Notice that the 2 factors are represented in the 2 columns of B and in the 2 rows of V as the direction in which the monkey's hand moves.

2 Factor Model

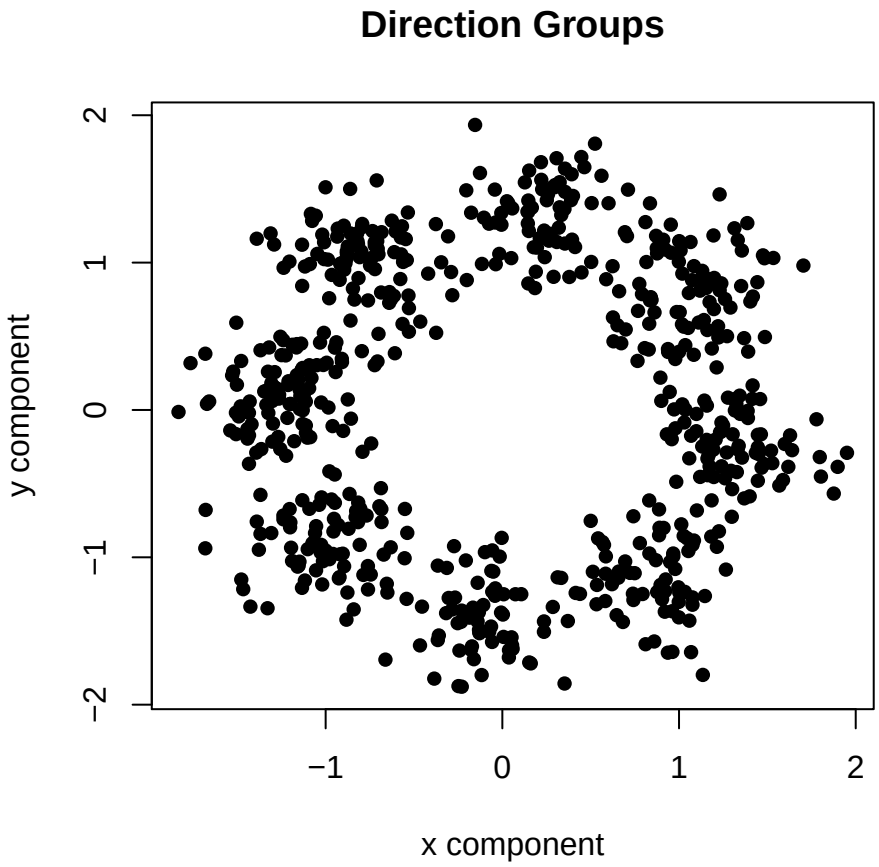
We can fit a 2-factor model to the data as follows.

```
fm<-factanal(x = neur,factors = 2,scores = "regression")
```

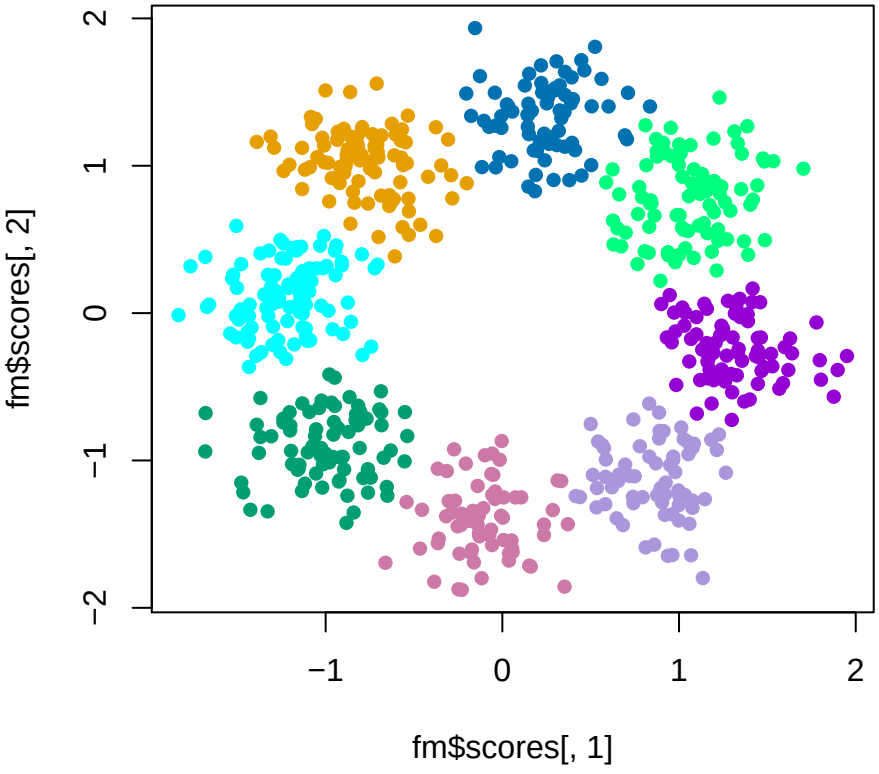
The plot below shows the preferred direction for each neuron, using the estimated factor loadings from the factor model.



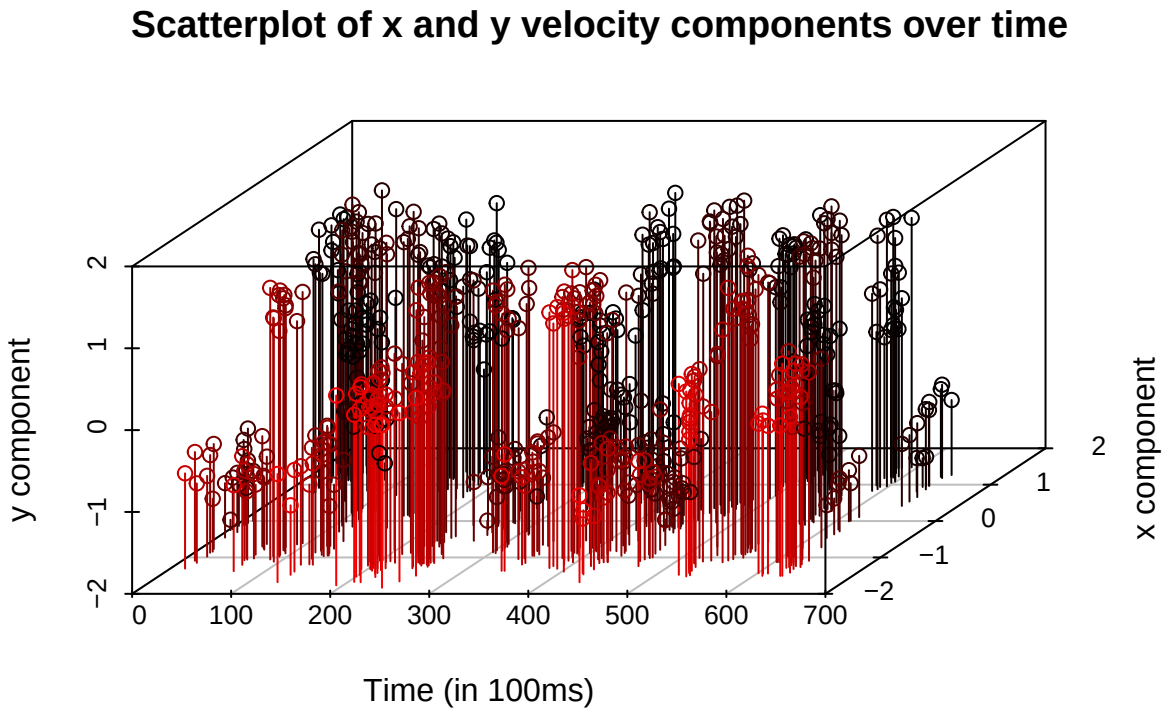
The factor scores from our factor model estimate the intended direction of motion



Using the k-means clustering algorithm, we can actually generate a plot that color codes the probable directions that the monkey was moving in.



Using 3d scatterplot above shows the x and y components over time. Smaller x components are highlighted in red where larger ones are highlighted in black. From this plot, it seems that the monkey changed direction about every 10 seconds (100 * 100ms).



A rotation matrix is an orthogonal matrix that rotates around a given angle α. In the 2 dimensional space we are currently working with, a rotation matrix r would be of the form

$$r_\alpha = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}$$

Since we wish to rotate 30 degrees, we first convert degrees to radians. 30 degrees is equal to $\frac{\pi}{6}$ radians.

$$r_{\frac{\pi}{6}} = \begin{bmatrix} \cos\frac{\pi}{6} & -\sin\frac{\pi}{6} \\ \sin\frac{\pi}{6} & \cos\frac{\pi}{6} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}$$

This has the affect of rotating the B matrix (containing the preferred direction vector of each neuron) 30 degrees as well. This difference in coordinate systems will have no other effects on the factor analysis. This is because when we change the coordinate system, we are only changing how the factor scores space relates to the factor loadings space.

3 Factor Model

```
mix3<-npEM(as.matrix(neur),mu0=3,maxiter=100,eps=1e-4,verb=FALSE,)
```

A 3 cluster model is appropriate because we can write the model in terms of the sum of three different components. The first component is VB, the second is A and the third is ϵ.

The relative weights for the 3 components are

```
signif(mix3$lambdahat,3)
```

```
## [1] 0.319 0.316 0.365
```

8 Factor Mixture Model

```
mix8<-npEM(as.matrix(neur),mu0=8,maxiter=100,eps=1e-4,verb=FALSE)
```

An 8 Cluster Mixture Model is appropriate because each of the 8 clusters will represent one of the 8 directions that the monkey is moving its hand.

The relative weights for the 8 components are

```
signif(mix8$lambdahat,3)
```

```
## [1] 0.104 0.144 0.128 0.136 0.130 0.108 0.133 0.116
```

```
cv.ll<-function(data,factors,nfolds=5){
  n <- nrow(data)
  fold.labels <- sample(rep(1:nfolds, length.out=n))
  ll<-rep(NA,nfolds)
  for (fold in 1:nfolds) {
    test.rows <- which(fold.labels == fold)
    train <- data[-test.rows,]
    test <- data[test.rows,]
    fm<-factanal(train,factors = factors,scores = "regression")
    ll[fold]<-charles(fm,test)
  }
  return(mean(ll))
}
```

```
cv.mix<-function(data,nfolds=5,k){
  n<-nrow(data)
  fold.labels <- sample(rep(1:nfolds, length.out=n))
  loglikes <- rep(NA,nfolds)
  for (fold in 1:nfolds){
    test.rows <- which(fold.labels == fold)
    train <- data[-test.rows,]
    test <- data[test.rows,]
    mixture <- npEM(train,mu0 = k,maxiter = 400,eps=1e-2,verb=FALSE)
    ll<-as.numeric(logmixlik.np(npmix = mixture,data = test)[1])
    loglikes[fold] <- ll
  }
  return(mean(data.matrix(loglikes)))
}
```

The cross validated log likelihood for a mixture model with 3 components is

```
cv.mix(data = neur,k=3)
```

```
## [1] -20474.63
```

The cross validated log likelihood for a mixture model with 3 components is

```
cv.mix(data = neur,k=8)
```

```
## [1] -21555.99
```

The cross validated log likelihood for a factor model with 2 factors is

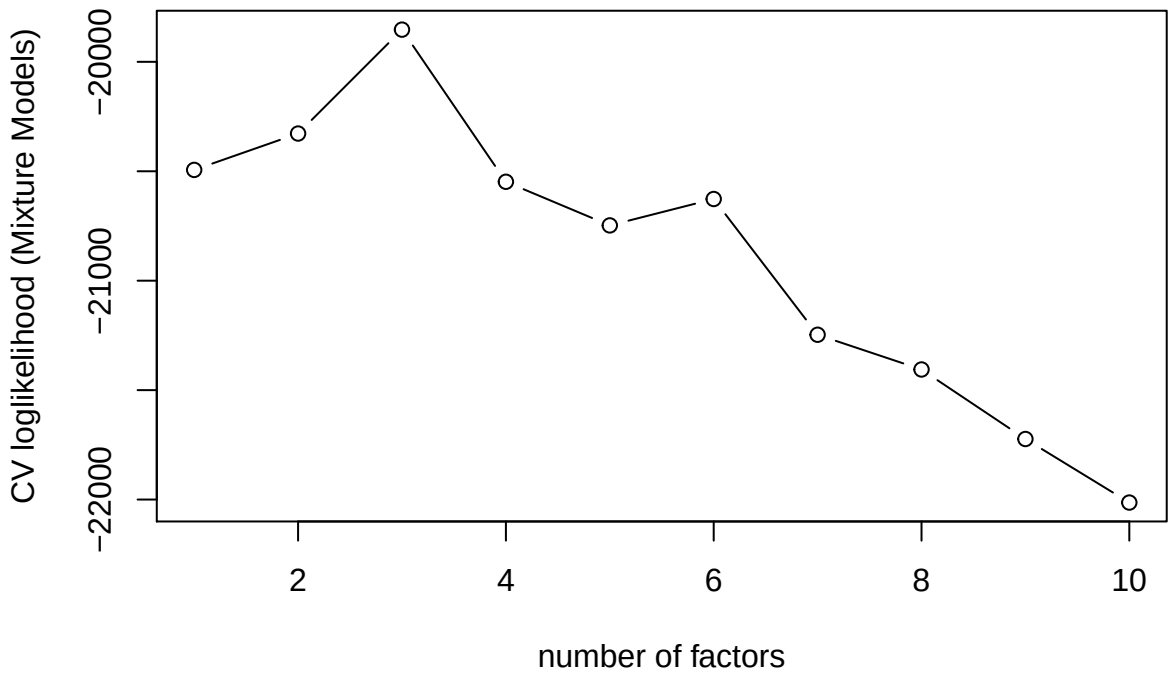
```
cv.ll(data = neur,factors = 2)
```

```
## [1] -16222.41
```

We can check the cross validated log likelihood for many different models.

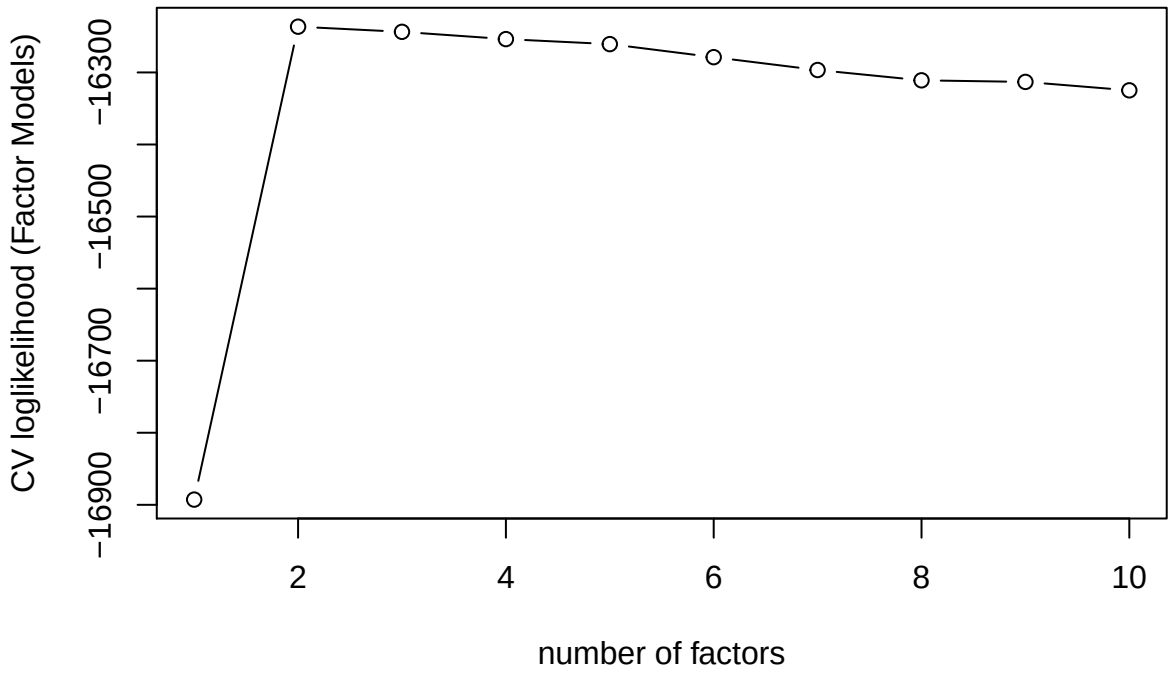
The cross validated log likelihood for variable mixture models are shown and plotted below.

```
cv.mix.val<-sapply(1:10,cv.mix,data=neur,nfolds=5)

plot(1:10, cv.mix.val,
xlab="number of factors", type="b", ylab="CV loglikelihood (Mixture Models)")
```



The cross validated log likelihood for variable factor models are shown and plotted below.

```
cv.ll.val<-sapply(1:10,cv.ll,data=neur,nfolds=5)

plot(1:10, cv.ll.val,
xlab="number of factors", type="b", ylab="CV loglikelihood (Factor Models)")
```



Conclusions

Between the mixture models, the plot and values show that the three component mixture model has the highest log likelihood and is therefore the best model.

Between the factor models, the two factor model has the highest log likelihood and is therefore the best model.

When we compare the two models, we see that the log likelihood is much greater for factor models, and the two factor model has the highest log likelihood of any other model. This implies that (of these models), the two factor model is the best overall.

This is as expected, because (as explained in more depth in **Models** *2 Factor Model*) this fits the model that states the expected number of spikes in a short time period is $a + \vec{b} \cdot \vec{v}$.