SpatIS – on calculating p-values and performing power analysis

Bernardo B. Niebuhr

14 de junho de 2018

Introduction

We are interested here in understanding whether individuals in a population are more or less specialized in their space use - i.e., how much different individuals in a population differ in their foraging areas and space use patterns. Our questions then are whether SpatIS - the spatial individual specialization index - is greater than what it would be expected if all individuals used the same areas, and if our sampling size is enough to infer that.

Below we describe how we are calculating the *p-value* and the statistical power of these analyses. We started with the following steps:

1) Calculate individual SpatIS for the data (the overlap between the individual and the population utilization distributions - UD) and save this values in an array called `observed`, with the 10 observed SpatIS values (we have n = 10 individuals sampled).
2) Randomize individuals' locations, to remove the use of specific foraging areas by different individuals, but keeping the place where they rest, the nest. Then, calculate individual SpatIS using data with randomized locations (step 1).
3) Repeat (2) 99 times (in our example here). We put that in a list of arrays called `expected`, with 99 arrays of 10 individual SpatIS values each (one array for each randomization).

Therefore, now we are working with the comparison of distributions of individual SpatIS values, not a comparison between the means (the population SpatIS values) as we were doing before.

Let's take a look at the `observed` and `expected` objects:

```
observed <- SpatIS_indiv[[1]]
expected <- SpatIS_indiv[2:naleat]
expected.polled <- unlist(expected) # all values pooled as a single array/distribution

# Look at observed values
observed
```

```
##        01        02        03        04        05        06        07
## 0.6516243 0.6419090 0.7071718 0.4643688 0.6831533 0.8865875 0.9913265
##        08        09        10
## 0.8046149 0.5369594 0.5967313
```

```
# Look at expected values
head(expected, n = 2)
```

```
## [[1]]
##        01        02        03        04        05        06        07
## 0.1827090 0.2014979 0.4456843 0.1761024 0.1651270 0.3127077 0.2657389
##        08        09        10
## 0.1956515 0.2256709 0.1812234
##
## [[2]]
```

```
##        01        02        03        04        05        06        07
## 0.2399795 0.2046650 0.2461841 0.2489311 0.1732306 0.2238741 0.2713958
##        08        09        10
## 0.2189411 0.1776490 0.1539547
```

```
# Loog at expected values pooled in a single array
head(expected.polled, n = 20)
```

```
##        01        02        03        04        05        06        07
## 0.1827090 0.2014979 0.4456843 0.1761024 0.1651270 0.3127077 0.2657389
##        08        09        10        01        02        03        04
## 0.1956515 0.2256709 0.1812234 0.2399795 0.2046650 0.2461841 0.2489311
##        05        06        07        08        09        10
## 0.1732306 0.2238741 0.2713958 0.2189411 0.1776490 0.1539547
```

Significance

To calculate a p-value, I followed Marco's suggestion and compared the observed and the expected distributions (the latter pooled for all randomizations).

I did that in two ways:

- using a two-sample t-test
- using a z test

It seems to me that both are equivalent, (the value of the statistic is exactly the same!) but I am not sure about the technical details. Take a look here at the implementation and results.

```
# p-value using a t.test
t.test(observed, expected.polled, alternative = 'greater', mu = 0,
       var.equal = FALSE, conf.level = 0.95)
```

```
##
##  Welch Two Sample t-test
##
## data:  observed and expected.polled
## t = 9.0765, df = 9.0228, p-value = 3.915e-06
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.3658898       Inf
## sample estimates:
## mean of x mean of y
## 0.6964447 0.2379910
```

```
# p-value using a z.test
# install.packages('BSDA', dependencies = T)
library(BSDA)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'BSDA'
```

```
## The following object is masked from 'package:datasets':
##
##     Orange
```

```r
z.test(observed, expected.polled, alternative = 'greater', mu = 0,
       sigma.x = sd(observed), sd(expected.polled), conf.level = 0.95)
```

```
##
##  Two-sample z-Test
##
## data:  observed and expected.polled
## z = 9.0765, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.3753721        NA
## sample estimates:
## mean of x mean of y
## 0.6964447 0.2379910
```

Power analysis

To calculate power, I first built a function called `t.power`, in which we compare each randomized array of 10 SpatIS values with the observed one, using a t test. As a result, we have 99 results of the t test. We then calculate power as the proportion of times we got the t.test p-value smaller than 0.05. I am not so sure about this implementation of power analysis, but I just followed what I found here: https://statistics.berkeley.edu/computing/r-t-tests.

```r
# power
t.power = function(obs, exp){
  ts = lapply(X = exp, FUN = t.test, y = obs, alternative = 'less', mu = 0,
              var.equal = FALSE, conf.level = 0.95)

  pval <- function(x) x$p.value
  tps <- unlist(lapply(ts, pval))

  sum(tps < .05) / length(tps)
}
#t.power(obs = observed, exp = expected)
```

Finally, I built a function in which we randomly select the individual SpatIS value for n individuals, from the observed and expected distributions, and calculate the power value for this subsample. We repeat that 100 times per value of n and calculate power as the average value among all 100 samples. We did that from n = 2 to n = 10, to check which sample size we need to reach sample sufficiency, in our test.

```r
t.power.vs.n <- function(n, observed, expected, n.repeat = 100) {
  power.val <- c()
  for(i in 1:n.repeat) {
    samp.obs <- sample(observed, n)
    samp.exp <- lapply(expected, sample, n)
    power.val <- c(power.val, t.power(samp.obs, samp.exp))
  }
  # list(power.val, mean(power.val))
  mean(power.val)
}
# Example for t = 3
t.power.vs.n(3, observed = observed, expected = expected)
```

```
## [1] 0.9755102
```

```r
# For all values
power.values <- unlist(lapply(2:10, t.power.vs.n, observed = observed, expected = expected))
data.frame(n = 2:10, power = power.values)
```

```
##    n     power
## 1  2 0.4709184
## 2  3 0.9513265
## 3  4 1.0000000
## 4  5 1.0000000
## 5  6 1.0000000
## 6  7 1.0000000
## 7  8 1.0000000
## 8  9 1.0000000
## 9 10 1.0000000
```

Basically, it seems that we would need only 3 individuals to check for individual specialization in space use, based our data. I am not sure if this makes sense, but maybe it is related to the difficulties in building null models in spatial and movement ecology.