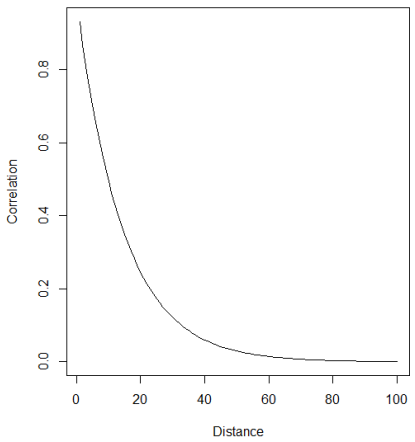
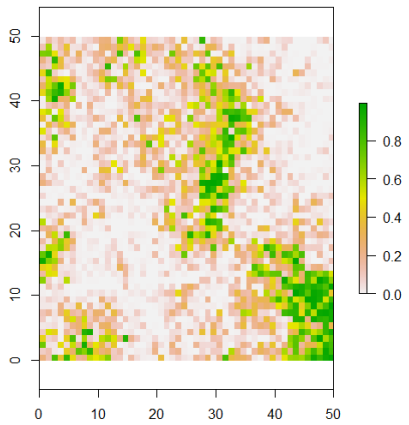
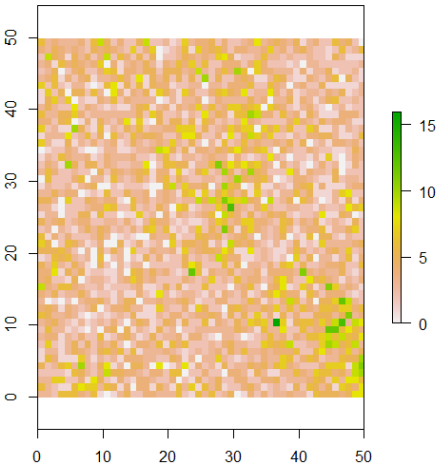
```
> library(pscl)
> library("raster")
> library("rgeos")
> library("rgdal")
> library("MASS")
> # organism: honey bee population
> # environmental predictor variable: floral diversity (sweet clover coverage)
> # we investigate the population in 5km * 5km area with 100 m^2 pixel size.
> # More flower diversity means higher sweet clover coverage at a site and we expect more honey bees there.
> # Therefore, we see high correlation between flower diversity and sweet clover abundance.
> # Moreover, my environmental variable is more spatially auto-correlated because I
> # assumed that pollination happens close to the flowering plants and sweet clover also grows with patchy
> # distribution, then I choose phi=0.07.
>
> # define function to draw random samples from a multivariate normal distribution
>
> set.seed(23456)
>
> rmvn <- function(n, mu = 0, V = matrix(1)) {
+   p <- length(mu)
+   if (any(is.na(match(dim(V), p))))
+     stop("Dimension problem!")
+   D <- chol(V)
+   t(matrix(rnorm(n * p), ncol = p) %*% D + rep(mu, rep(n, p)))
+ }
>
> # set up a square lattice region
> simgrid <- expand.grid(1:50, 1:50)
> n <- nrow(simgrid)
>
> # set up distance matrix
> distance <- as.matrix(dist(simgrid))
> # Generate random variable
>
> phi = 0.07 #phi determines scale of distance variation
> plot(1:100, exp(-phi * 1:100), type = "l", xlab = "Distance", ylab = "Correlation")
>
```



```
> y_true <- rmvn(1, rep(0, n), exp(-phi * distance))
> X_beta <- rbeta(n, shape1 = exp(y_true), shape2 = exp(1-y_true))
> Xraster_beta <- rasterFromXYZ(cbind(simgrid[, 1:2] - 0.5, X_beta))
>
> plot(Xraster_beta)
```

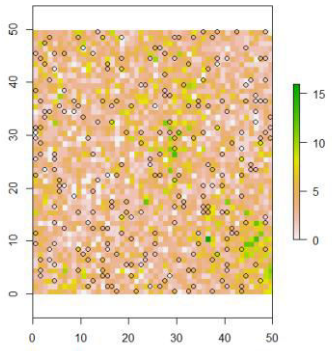



```
> # I changed my predictor variable to produce count data for giving to the neighbor
> # but my actual data is continues because I'm working on coverage in percent.
> X <- rpois(n, lambda=exp(1+rbeta(n, shape1 = exp(y_true), shape2 = exp(1-y_true))))
>
> # visualize results
> Xraster <- rasterFromXYZ(cbind(simgrid[, 1:2] - 0.5, X))
>
> plot(Xraster)
>
```

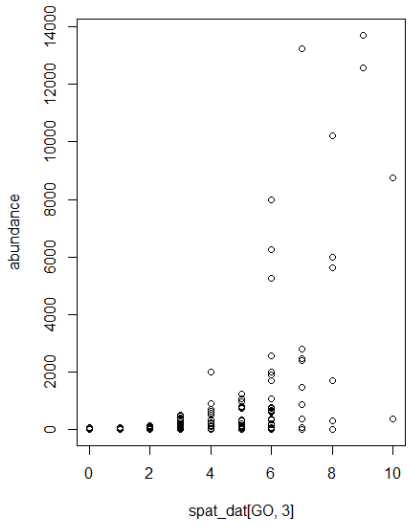



```
> writeRaster(Xraster,file="Xraster.tif",format="GeoTiff")
> # Converting raster to a data frame
> spat_dat=rasterToPoints(Xraster)
```

```
> # The maximum point to sample equals my population size which is 2500. I chose 10% of my population as
> # sample size.
> sample_size <- 250
> GO=sample(x=c(1:nrow(spat_dat)),size=sample_size)
> points(spat_dat[GO,c(1:2)])
>
```




```
> presence_intercept=0.5
> presence_slope=1.3
>
> PA=rbinom(sample_size,plogis(presence_intercept+spat_dat[GO,3]*presence_slope),
+           size=1)
>
> count_intercept=2
> count_slope=0.8
> over_dispersion=0.5
>
> # by multiplying the binomial and negative binomial variable we assume that these 2 distributions are
> # independent. As we see in PA and abundance matrix, in some cases the presence is 1 but the abundance is 0.
> # In some samples we may have high floral diversity but no sweet clover.
> # high floral coverage does not mean we will have high amount of sweet clover and then honey bee. High
> # spatial autocorrelation shows the patchy nature of flowers (high coverage in some areas)
> abundance=PA*rnbinom(sample_size,mu=
+                        exp(count_intercept+count_slope*spat_dat[GO,3]),
+                      size=over_dispersion)
>
> plot(abundance~spat_dat[GO,3])
>
```




```
> write.csv(data.frame(abundance,spat_dat[GO,3]),"file.csv")
> # I'm reading Monica's data which is how many pika are in the region and the number of pink flowers at each
site.
>
> data <- read.csv("Pika_PinkFlowers.csv")
> head(data)
  X abundance spat_dat.GO..3.
1 1        27              10
2 2         6               6
3 3         1               6
4 4         9               6
5 5        11               1
6 6        10               7
> plot(data$abundance~data$spat_dat.GO..3.)
>
> # as there is no 0 value in the data, it means that there is no need for applying binomial (because her PA
> # matrix has just 1 in it and we may not be able to retrieve presence/absence intercept and slope). I also
> # don't see over dispersion in the data too (negative binomial is not reasonable for that too). However, I
> # used Poisson and negative binomial to see which one gives the better results for her data.
> m1 <- glm(data$abundance~data$spat_dat.GO..3.,family="poisson")
> coef_m1<-coef(m1)
> slope_poisson <- exp(coef_m1[2])
> intercept_poisson <- exp(coef_m1[1])
> confint(m1)
Waiting for profiling to be done...
                          2.5 %   97.5 %
(Intercept)          0.87762234 1.853110
data$spat_dat.GO..3. 0.08515651 0.232434
>
> abundance2 <- data$abundance[which(data$abundance>0)]
> m1_nb <- glm.nb(abundance2~data$spat_dat.GO..3.)
> coef_m1_nb<-coef(m1_nb)
> slope_nb <- exp(coef_m1_nb[2])
> intercept_nb <- exp(coef_m1_nb[1])
> confint(m1_nb)
Waiting for profiling to be done...
```

```
                           2.5 %    97.5 %
(Intercept)          0.807648175 2.3221336
data$spat_dat.GO..3. 0.007901195 0.2505494
> cbind(slope_poisson, intercept_poisson, slope_nb, intercept_nb)
                     slope_poisson intercept_poisson slope_nb intercept_nb
data$spat_dat.GO..3.      1.171217          3.980419 1.136427     4.752601

> # It seems that negative binomial regression works better for slope estimation than Poisson. Because her
> # true value for slope is 0.1. The Poisson regression's slope is also close to the real values. If I did not
> # know about her real values, I would definitely choose Poisson's results because, there is no over
> # dispersion in her data and the confidence interval is smaller for Poisson regression rather than negative
> # binomial. If her data was in hurdle model, I could use hurdle function to retrieve the intercept and slope
> # she used. below is the line for doing that:
> #mod.hurdle <- hurdle(abundance ~ ., data = data[,2:3], dist = "negbin", zero.dist = "binomial")
> #mod.hurdle
```