Supplemental File: Integrating point process models, evolutionary ecology, and traditional knowledge improves landscape archaeology: A case from Southwest Madagascar

Dylan S. Davis

6/8/2020

Introduction

This R Markdown shows the development of a point process modeling procedure to evaluate archaeological predictive models. This code reflects the methods discussed in the text of the manuscript.

Load Libraries

```
library(spatstat)

## spatstat 1.63-0       (nickname: 'Space camouflage')
## For an introduction to spatstat, type 'beginner'

library(maptools)

library(raster)

library(rgdal)

## rgdal: version: 1.4-8, (SVN revision 845)


library(rgeos)

## rgeos version: 0.5-2, (SVN revision 621)
##  GEOS runtime version: 3.6.1-CAPI-1.10.1
##  Linking to sp version: 1.3-2


library(sp)
library(MuMIn)
library(MASS)
library(here)

##
## Attaching package: 'MASS'

## The following objects are masked from 'package:raster':
##
##     area, select
## The following object is masked from 'package:spatstat':
##
##     area

library(graphics)
```

Setup workspace for spatial analysis

```
setwd(here())

#Bounding window data
win_ext <- readOGR(dsn="AOI_boundary.shp")
```

Load Archaeological Points and Environmental Datasets

```
Arch <- readOGR(dsn = "TOTAL_POINT_artifWGS_2019.shp")

veg_index <- mask(raster("SAVI_35_dist_p5_9.tif"), win_ext)

dunes <- mask(raster("dune_dist_p5_9.tif"), win_ext)

water <- mask(raster("water_dist_p5_9.tif"), win_ext)

coral <- mask(raster("coral_dist_p5_9.tif"), win_ext)

islands <- mask(raster("island_dist_p5_9.tif"), win_ext)

rocks <- mask(raster("rock_dist_5_91.tif"), win_ext)

bedrock <- mask(raster("bedrk_120dst.tif"), win_ext)
```

Convert data to PPM format for spatstat analysis

```
b_win <- as.owin(win_ext)

#creates coordinate matrix

arch_pp <- matrix(NA, nrow = nrow(Arch), ncol = 2)
arch_pp[,1] <- Arch$coords.x1 #Long
arch_pp[,2] <- Arch$coords.x2 #Lat

#converts CM to PPP (spatstat format)

pparch_points <- ppp(x=arch_pp[,1], y=arch_pp[,2],
                    window = owin(bbox(arch_pp)[1,],
                                  bbox(arch_pp)[2,]))
```

```
## Warning: data contain duplicated points #there are no duplicated points,
but some points are too close together for spatstat to recognize. We use
rjitter to correct this.

ppparch <- rjitter(ppparch_points, 0.001) #to rectify duplicated point issue


SAVI_pp <- as.im(veg_index)

pppwater <- as.im(water)

pppdunes <- as.im(dunes)

pppcoral <- as.im(coral)

pppisland <- as.im(islands)

ppprocshor <- as.im(rocks)

ppp_bedrock <- as.im(bedrock)
```

Exploratory Analysis

```
#First-order Trends using rho-hat estimation with 95% confidence bands

#Unweighted Rho-hat tests


SAVI_rh_nw <- rhohat(ppparch, SAVI_pp)


dune_rh_nw <- rhohat(ppparch, pppdunes)


water_rh_nw <- rhohat(ppparch, pppwater)


coral_rh_nw <- rhohat(ppparch, pppcoral)


island_rh_nw <- rhohat(ppparch, pppisland)


rock_rh_nw <- rhohat(ppparch, ppprocshor)

bedrock_rh_nw <- rhohat(ppparch, ppp_bedrock)

bedrock_rh_nw

#The following code creates Figure 4.
```

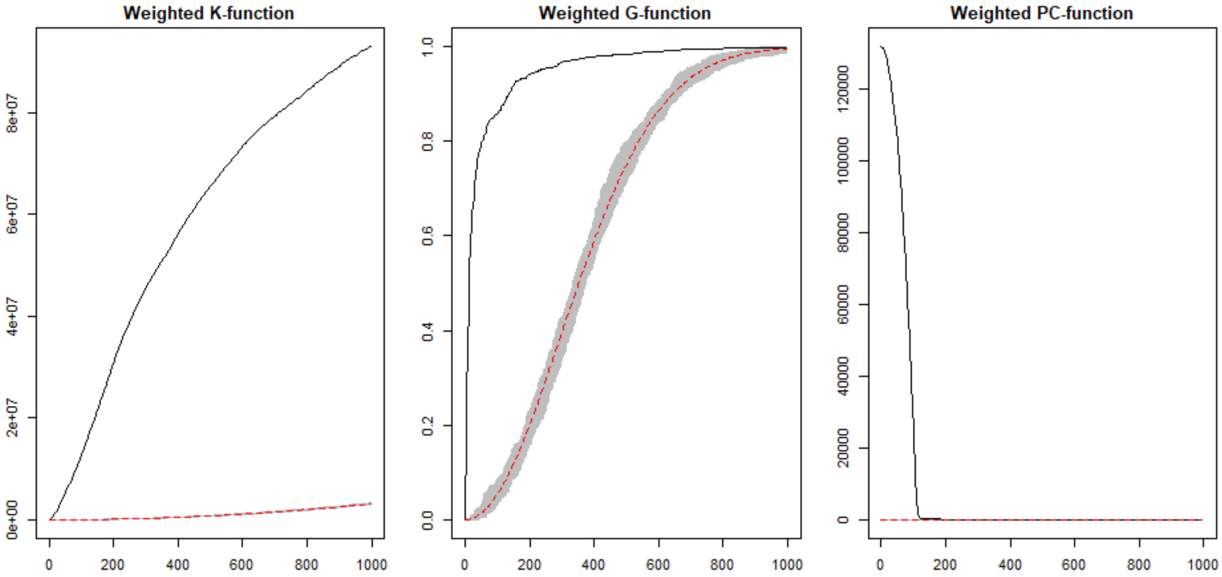
```r
par(mfrow=c(3,3))
par(mar=c(2,2,1.5,1.5))

plot(bedrock_rh_nw, main= "Bedrock", legend=F, ylim=c(0,0.00002),
xlab="Distance (m)", ylab="Absolute Intensity")# highest y-limit (0.00009)
plot(island_rh_nw, main= "Islands", legend=F, ylim=c(0,0.00002),
xlab="Distance (m)", ylab="Absolute Intensity") #very high y-limit (0.1)
plot(rock_rh_nw, main= "Rock Outcrops", legend=F, ylim=c(0,0.00002),
xlab="Distance (m)", ylab="Absolute Intensity") #high limit (0.00004)
plot(coral_rh_nw, main= "Coral", legend=F, ylim=c(0,0.00002), xlab="Distance
(m)", ylab="Absolute Intensity")
plot(water_rh_nw, main= "Ocean", legend=F, ylim=c(0,0.00002), xlab="Distance
(m)", ylab="Absolute Intensity")
plot(dune_rh_nw, main= "Paleodunes", legend=F, ylim=c(0,0.00002),
xlab="Distance (m)", ylab="Absolute Intensity")
plot(SAVI_rh_nw, main= "Vegetation", legend=F, ylim=c(0,0.00002),
xlab="Distance (m)", ylab="Absolute Intensity")

dev.off()

par(mfrow=c(1,1))

#First-order rho-hat intensity tests using weights (by artifact count)

SAVI_rh <- rhohat(ppparch, SAVI_pp, weights = Arch$Total_Mate)
dune_rh <- rhohat(ppparch, pppdunes, weights = Arch$Total_Mate)


water_rh <- rhohat(ppparch, pppwater, weights = Arch$Total_Mate)


coral_rh <- rhohat(ppparch, pppcoral, weights = Arch$Total_Mate)


island_rh <- rhohat(ppparch, pppisland, weights = Arch$Total_Mate)


rock_rh <- rhohat(ppparch, ppprocshor, weights = Arch$Total_Mate)

bedrock_rh <- rhohat(ppparch, ppp_bedrock, weights = Arch$Total_Mate)

#The following code creates Figure 5.

par(mfrow=c(3,3))
par(mar=c(2,2,1.5,1.5))

plot(bedrock_rh, main="Bedrock", legend=F, ylim=c(0,0.0002), xlab="Distance
(m)", ylab="Absolute Intensity")#highest y-limit (0.002)
plot(island_rh, main="Islands", legend=F, ylim=c(0,0.0002), xlab="Distance
(m)", ylab="Absolute Intensity") #very low y-limit (0.00002)
plot(rock_rh, main="Rock Outcrops", legend=F, ylim=c(0,0.0002),
xlab="Distance (m)", ylab="Absolute Intensity")#third highest y-limit,
everything else visible from this point
plot(coral_rh, main="Corals", legend=F, ylim=c(0,0.0002), xlab="Distance
(m)", ylab="Absolute Intensity")
plot(water_rh, main="Ocean", legend=F, ylim=c(0,0.0002), xlab="Distance (m)",
ylab="Absolute Intensity")
plot(dune_rh, main="Paleodunes", legend=F, ylim=c(0,0.0002), xlab="Distance
(m)", ylab="Absolute Intensity")
plot(SAVI_rh, main="Vegetation", legend=F, ylim=c(0,0.0002), xlab="Distance
(m)", ylab="Absolute Intensity")

par(mfrow=c(1,1))
dev.off()
```

Second-Order Tests: Summary Distribution Functions

```
#Unweighted
K_test <- envelope(ppparch, fun=Kest, nsim=39, fix.n=T,
correction="translation", global=F)

G_test <- envelope(ppparch, fun=Gest, nsim=39, fix.n=T, correction="best",
global=F)

PCFtest <- envelope(ppparch, fun=pcf, nsim=39, fix.n=T,
correction="translation", global=F, divisor="d")

par("mar") #check dimensions of image plots

#The following code creates Figure 6

par(mfrow=c(1,3))
par(mar=c(2,2,2,2))
plot(K_test, main="K-function",xlim=c(0,1000), legend=F, xlab="Distance (m)")
plot(G_test, main="G-function",xlim=c(0,1000),  legend=F, xlab="Distance
(m)")
plot(PCFtest, main="PC-function",xlim=c(0,1000), legend=F, xlab="Distance
(m)")
```

```
#Weighted model functions (by artifact count)
K_test_w <- envelope(ppparch, fun=Kest, nsim=39, fix.n=T,
wght=Arch$Total_Mate, correction="translation", global=F)

G_test_w <- envelope(ppparch, fun=Gest, nsim=39, fix.n=T,
wght=Arch$Total_Mate, correction="best", global=F)

PCFtest_w <- envelope(ppparch, fun=pcf, nsim=39, fix.n=T,
wght=Arch$Total_Mate, correction="translation", global=F, divisor="d")

#Creates figure below (weighted summary distribution functions)
```

```
par(mfrow=c(2,2))
par(mar=c(2,2,2,2))
plot(K_test_w, main="Weighted K-function",xlim=c(0,1000), legend=F)
plot(G_test_w, main="Weighted G-function",xlim=c(0,1000),  legend=F)
plot(PCFtest_w, main="Weighted PC-function",xlim=c(0,1000), legend=F)

par(mfrow=c(1,1))
```



Point Process Modeling of First-Order Properties

```
#PPM_0 - null model using complete spatial randomness (CSR)

ppm0 <- ppm(ppparch, ~1, correction="translation")


#Point Process model of Davis et al. (2020) predictive algorithm

ppm1 <- ppm(ppparch, ~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland,
correction = "translation")

#New algorithm including all environmental covariates
ppm2 <- ppm(ppparch,
~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland+ppprocshor+ppp_bedrock,
correction = "translation")

ppm2
```

```
## Nonstationary Poisson process
##
## Log intensity:  ~SAVI_pp + pppdunes + pppcoral + pppwater + pppisland +
## ppprocshor + ppp_bedrock
##
## Fitted trend coefficients:
##   (Intercept)       SAVI_pp      pppdunes      pppcoral      pppwater
## -1.023702e+01 -1.300368e-03  5.448134e-04 -1.149442e-02  1.092304e-02
##     pppisland    ppprocshor   ppp_bedrock
##  3.847663e-03 -3.899362e-04 -2.019069e-02
##
##                   Estimate         S.E.       CI95.lo       CI95.hi Ztest
## (Intercept) -1.023702e+01 2.560448e-01 -1.073886e+01 -9.7351829007   ***
## SAVI_pp     -1.300368e-03 8.236907e-04 -2.914772e-03  0.0003140365
## pppdunes     5.448134e-04 5.105335e-04 -4.558138e-04  0.0015454407
## pppcoral    -1.149442e-02 1.335140e-03 -1.411124e-02 -0.0088775922   ***
## pppwater     1.092304e-02 1.360987e-03  8.255550e-03  0.0135905213   ***
## pppisland    3.847663e-03 9.748846e-04  1.936924e-03  0.0057584013   ***
## ppprocshor  -3.899362e-04 6.091072e-05 -5.093190e-04 -0.0002705534   ***
## ppp_bedrock -2.019069e-02 1.094142e-03 -2.233517e-02 -0.0180462133   ***
##                   Zval
## (Intercept) -39.981372
## SAVI_pp      -1.578709
## pppdunes      1.067145
## pppcoral     -8.609151
## pppwater      8.025818
## pppisland     3.946788
## ppprocshor   -6.401767
## ppp_bedrock -18.453449
```

Covariate Model Selection

```
#Model Selection between CSR, the original Davis et al. (2020) model, and
model with new covariates
MS_AIC <- model.sel(ppm0, ppm1, ppm2, rank = AIC)
MS_AIC
```

```
## Model selection table
##                               trend df    logLik     AIC  delta weight
## ppm2 S_pp+pppd+pppc+pppw+ppps+pppr+  8  -8394.182 16804.4    0.0      1
## ppm1       S_pp+pppd+pppc+pppw+ppps  6 -10139.231 20290.5 3486.1      0
## ppm0                                 1 -11438.133 22878.3 6073.9      0
## Abbreviations:
## trend:  = '~1',
##        S_pp+pppd+pppc+pppw+ppps =
'~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland',
##        S_pp+pppd+pppc+pppw+ppps+pppr+ =
'~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland+ppprocshor+'
## Models ranked by AIC(x)
```

```r
MS_BIC <- model.sel(ppm0, ppm1, ppm2, rank = BIC)
MS_BIC
```

```
## Model selection table
##                               trend df     logLik     BIC   delta weight
## ppm2 S_pp+pppd+pppc+pppw+ppps+pppr+  8  -8394.182 16841.9    0.00      1
## ppm1       S_pp+pppd+pppc+pppw+ppps  6 -10139.231 20318.6 3476.72      0
## ppm0                                 1 -11438.133 22883.0 6041.08      0
## Abbreviations:
## trend:  = '~1',
##        S_pp+pppd+pppc+pppw+ppps = 
'~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland',
##        S_pp+pppd+pppc+pppw+ppps+pppr+ = 
'~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland+ppprocshor+'
## Models ranked by BIC(x)
```

```r
#Assess best fitting model (PPM2) using stepwise selection
stepAIC(ppm2)
```

```
## Start:  AIC=16804.36
## ~SAVI_pp + pppdunes + pppcoral + pppwater + pppisland + ppprocshor + 
##     ppp_bedrock

##               Df   AIC
## - pppdunes     1 16804
## <none>           16804
## - SAVI_pp      1 16805
## - pppisland    1 16821
## - ppprocshor   1 16856
## - pppwater     1 16882
## - pppcoral     1 16898
## - ppp_bedrock  1 19366


## Step:  AIC=16803.49
## ~SAVI_pp + pppcoral + pppwater + pppisland + ppprocshor + ppp_bedrock

##               Df   AIC
## <none>           16804
## - SAVI_pp      1 16805
## - pppisland    1 16822
## - ppprocshor   1 16854
## - pppwater     1 16882
## - pppcoral     1 16898
## - ppp_bedrock  1 19504

## Nonstationary Poisson process
## 
## Log intensity:  ~SAVI_pp + pppcoral + pppwater + pppisland + ppprocshor + 
## ppp_bedrock
## 
```

```
## Fitted trend coefficients:
##   (Intercept)       SAVI_pp      pppcoral      pppwater     pppisland 
## -1.023407e+01 -1.410127e-03 -1.149096e-02  1.092652e-02  3.993073e-03 
##    ppprocshor   ppp_bedrock 
## -3.835392e-04 -2.009802e-02 
## 
##                  Estimate         S.E.       CI95.lo       CI95.hi Ztest
## (Intercept) -1.023407e+01 2.557837e-01 -1.073540e+01 -9.7327431970   ***
## SAVI_pp     -1.410127e-03 8.191544e-04 -3.015640e-03  0.0001953863      
## pppcoral    -1.149096e-02 1.326391e-03 -1.409064e-02 -0.0088912848   ***
## pppwater     1.092652e-02 1.353405e-03  8.273893e-03  0.0135791415   ***
## pppisland    3.993073e-03 9.637460e-04  2.104166e-03  0.0058819805   ***
## ppprocshor  -3.835392e-04 6.042613e-05 -5.019722e-04 -0.0002651061   ***
## ppp_bedrock -2.009802e-02 1.093286e-03 -2.224082e-02 -0.0179552200   ***
##                   Zval
## (Intercept) -40.010644
## SAVI_pp      -1.721442
## pppcoral     -8.663329
## pppwater      8.073355
## pppisland     4.143284
## ppprocshor   -6.347240
## ppp_bedrock -18.383129
## Problem:
```

```
stepBIC <- stepAIC(ppm2, k=log(length(Arch)))
```

```
## Start:  AIC=16841.87
## ~SAVI_pp + pppdunes + pppcoral + pppwater + pppisland + ppprocshor + 
##     ppp_bedrock

##                Df   AIC
## - pppdunes      1 16836
## - SAVI_pp       1 16838
## <none>            16842
## - pppisland     1 16854
## - ppprocshor    1 16889
## - pppwater      1 16914
## - pppcoral      1 16930
## - ppp_bedrock   1 19399

## 
## Step:  AIC=16836.31
## ~SAVI_pp + pppcoral + pppwater + pppisland + ppprocshor + ppp_bedrock

##                Df   AIC
## - SAVI_pp       1 16833
## <none>            16836
## - pppisland     1 16850
## - ppprocshor    1 16882
## - pppwater      1 16910
## - pppcoral     1 16926
## - ppp_bedrock  1 19532

##
## Step:  AIC=16832.75
## ~pppcoral + pppwater + pppisland + ppprocshor + ppp_bedrock

##                Df   AIC
## <none>            16833
## - pppisland    1 16851
## - ppprocshor   1 16878
## - pppwater     1 16910
## - pppcoral     1 16928
## - ppp_bedrock  1 20076

#New Best fitting PPM based on model selection
ppm3 <- ppm(ppparch, ~pppcoral+pppwater+pppisland+ppprocshor+ppp_bedrock,
correction = "translation") #Best fitting based on BIC

ppm4 <- ppm(ppparch,
~SAVI_pp+pppcoral+pppwater+pppisland+ppprocshor+ppp_bedrock, correction =
"translation") #Best fitting based on AIC

#Following code produces data in Table 2

MS_AIC2 <- model.sel(ppm0, ppm1, ppm2, ppm3, ppm4, rank = AIC)

## Model selection table
##                                trend
## ppm4 S_pp+pppc+pppw+ppps+pppr+ppp_b
## ppm2 S_pp+pppd+pppc+pppw+ppps+pppr+
## ppm3      pppc+pppw+ppps+pppr+ppp_b
## ppm1       S_pp+pppd+pppc+pppw+ppps
## ppm0
##      df     logLik     AIC
## ppm4  7  -8394.744 16803.5
## ppm2  8  -8394.182 16804.4
## ppm3  6  -8396.311 16804.6
## ppm1  6 -10139.231 20290.5
## ppm0  1 -11438.133 22878.3
##        delta weight
## ppm4    0.00  0.452
## ppm2    0.88  0.292
## ppm3    1.13  0.256
## ppm1 3486.97  0.000
## ppm0 6074.78  0.000
## Abbreviations:
## trend:  = ‘~1’,
##        pppc+pppw+ppps+pppr+ppp_b =
## ‘~pppcoral+pppwater+pppisland+ppprocshor+ppp_bedrock’,
##        S_pp+pppc+pppw+ppps+pppr+ppp_b =
## ‘~SAVI_pp+pppcoral+pppwater+pppisland+ppprocshor+ppp_bedrock’,
##         S_pp+pppd+pppc+pppw+ppps =
## ‘~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland’,
##         S_pp+pppd+pppc+pppw+ppps+pppr+ =
## ‘~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland+ppprocshor+’
## Models ranked by AIC(x)
```

```
MS_BIC2 <- model.sel(ppm0, ppm1, ppm2, ppm3, ppm4, rank = BIC)
```

```
## Model selection table
##                                trend
## ppm3      pppc+pppw+ppps+pppr+ppp_b
## ppm4 S_pp+pppc+pppw+ppps+pppr+ppp_b
## ppm2 S_pp+pppd+pppc+pppw+ppps+pppr+
## ppm1       S_pp+pppd+pppc+pppw+ppps
## ppm0                               
##      df     logLik     BIC
## ppm3  6  -8396.311 16832.8
## ppm4  7  -8394.744 16836.3
## ppm2  8  -8394.182 16841.9
## ppm1  6 -10139.231 20318.6
## ppm0  1 -11438.133 22883.0
##         delta weight
## ppm3     0.00  0.848
## ppm4     3.55  0.143
## ppm2     9.12  0.009
## ppm1  3485.84  0.000
## ppm0  6050.20  0.000
## Abbreviations:
## trend:  = ‘~1’,
##         pppc+pppw+ppps+pppr+ppp_b =
## ‘~pppcoral+pppwater+pppisland+ppprocshor+ppp_bedrock’,
##         S_pp+pppc+pppw+ppps+pppr+ppp_b =
## ‘~SAVI_pp+pppcoral+pppwater+pppisland+ppprocshor+ppp_bedrock’,
##         S_pp+pppd+pppc+pppw+ppps =
## ‘~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland’,
##         S_pp+pppd+pppc+pppw+ppps+pppr+ =
## ‘~SAVI_pp+pppdunes+pppcoral+pppwater+pppisland+ppprocshor+’
## Models ranked by BIC(x)
```

Assess Residual Values

```
#Evaluate Residual Values for Best Fitting Model
RES_PPM3 <- residuals.ppm(ppm3, drop=T)

RES_PPM4 <- residuals.ppm(ppm4, drop=T)
```

```r
RES_PPM1 <- (residuals.ppm(ppm1, drop=T))

#Following code produces Figure 7.

par(mfrow=c(3,2))
par(mar=c(1,3,1,3)) #reset image dimensions to 1x1

plot.msr(RES_PPM3, main="PPM3 Raw Residuals", pch=16, cex=0.25)

diagnose.ppm(ppm3, main="PPM3 Smoothed Residuals",type = "pearson",
which="smooth", cumulative = T)

plot.msr(RES_PPM4, main="PPM4 Raw Residuals", pch=16, cex=0.25)

diagnose.ppm(ppm4, main="PPM4 Smoothed Residuals",type = "pearson",
which="smooth", cumulative = T)

plot.msr(RES_PPM1, main="PPM1 Raw Residuals", pch=16, cex=0.25)

diagnose.ppm(ppm1, main="PPM1 Smoothed Residuals", type = "pearson",
which="smooth", cumulative = T)

par(mfrow=c(1,1))

#Assess fit of models with second-order properties using Residual K- and G-
tests

K_sim1_ppm0 <- envelope(ppm0, Kres, nsim=39, fix.n=T,
correction="translation", global=F, divisor="d")

K_sim1_ppm1 <- envelope(ppm1, Kres, nsim=39, fix.n=T,
correction="translation", global=F, divisor="d")

K_sim1_ppm3 <- envelope(ppm3, Kres, nsim=39, fix.n=T,
correction="translation", global=F, divisor="d")

K_sim1_ppm4 <- envelope(ppm4, Kres, nsim=39, fix.n=T,
correction="translation", global=F, divisor="d")

G_sim1_ppm0 <- envelope(ppm0, Gres, nsim=39, fix.n=T, correction="best",
global=F)

G_sim1_ppm1 <- envelope(ppm1, Gres, nsim=39, fix.n=T, correction="best",
global=F)

G_sim1_ppm3 <- envelope(ppm3, Gres, nsim=39, fix.n=T, correction="best",
global=F)

G_sim1_ppm4 <- envelope(ppm4, Gres, nsim=39, fix.n=T, correction="best",
global=F)
```

```
#Following Code Produces Figure 7

par(mfrow=c(2,3))
par(mar=c(2,2,2,2))
plot(K_sim1_ppm3, legend=F, main="Residual K-Function\n PPM3", xlab="Distance (m)")
plot(K_sim1_ppm4, legend=F, main="Residual K-Function\n PPM4", xlab="Distance (m)")
plot(K_sim1_ppm1, legend=F, main="Residual K-Function\n PPM1", xlab="Distance (m)")
plot(G_sim1_ppm3, legend=F, main="Residual G-Function\n PPM3", xlab="Distance (m)")
plot(G_sim1_ppm4, legend=F, main="Residual G-Function\n PPM4", xlab="Distance (m)")
plot(G_sim1_ppm1, legend=F, main="Residual G-Function\n PPM1", xlab="Distance (m)")



#Coefficient Estimates of Best-fitting model (PPM3)

ppm3

## Nonstationary Poisson process
##
## Log intensity:  ~pppcoral + pppwater + pppisland + ppprocshor +
## ppp_bedrock
##
## Fitted trend coefficients:
##   (Intercept)      pppcoral      pppwater     pppisland
## -10.351284181  -0.011757903   0.011161231   0.004294965
##    ppprocshor   ppp_bedrock
##  -0.000384706  -0.020488528

##                  Estimate         S.E.       CI95.lo
## (Intercept) -10.351284181 2.479784e-01 -1.083731e+01
## pppcoral     -0.011757903 1.326579e-03 -1.435795e-02
## pppwater      0.011161231 1.353968e-03  8.507502e-03
## pppisland     0.004294965 9.516529e-04  2.429760e-03
## ppprocshor   -0.000384706 6.089174e-05 -5.040516e-04
## ppp_bedrock  -0.020488528 1.075191e-03 -2.259586e-02
##                   CI95.hi Ztest       Zval
## (Intercept) -9.8652554468   *** -41.742685
## pppcoral    -0.0091578568   ***  -8.863329
## pppwater     0.0138149591   ***   8.243349
## pppisland    0.0061601708   ***   4.513164
## ppprocshor  -0.0002653604   ***  -6.317868
## ppp_bedrock -0.0183811931   *** -19.055715
```

GIS Analysis

From the results of exploratory rho-hat tests and coefficient estimates from the PPMs, we take the above information into consideration to develop a series of probability rasters in ArcGIS. The Raster Calculator tool is used.

```
## Code Creates Unweighted predictive raster using Raster Calculator Tool,
following PPM2

Unweighted_Model =
(1/"bdrck_dpt_10m")+(1/"coral_dist_p5_9.tif")+(1/"island_dist_p5_9.tif")+(1/"
rock_dist_5_91.tif")+(1/"water_dist_p5_9.tif")

## Code Creates Weighted Model 1 using Raster Calculator Tool, following PPM
results

Weighted_Model1 =
((2.5*1/"bdrck_dpt_10m"))+(1.75*(1/"coral_dist_p5_9.tif"))+(1/"dune_dist_p5_9
.tif")+(1.75*(1/"island_dist_p5_9.tif"))+(2*(1/"rock_dist_5_91.tif"))+(1/"wat
er_dist_p5_9.tif")+(2*(1/"SAVI_35_dist_p5_9.tif"))

## Code Creates Weighted Model 2 using Raster Calculator Tool, following PPM
results

Weighted_Model2 =
((3*1/"bdrck_dpt_10m"))+(2*(1/"coral_dist_p5_9.tif"))+(1/"dune_dist_p5_9.tif"
)+(1.75*(1/"island_dist_p5_9.tif"))+(2.5*(1/"rock_dist_5_91.tif"))+(1/"water_
dist_p5_9.tif")+(2*(1/"SAVI_35_dist_p5_9.tif"))

## Code Creates Weighted Model 3 using Raster Calculator Tool, following PPM
results

Weighted_Model3 =
((2.5*1/"bdrck_dpt_10m"))+(1.5*(1/"coral_dist_p5_9.tif"))+(1/"dune_dist_p5_9.
tif")+(1.5*(1/"island_dist_p5_9.tif"))+(2*(1/"rock_dist_5_91.tif"))+(1/"water
_dist_p5_9.tif")+(1.75*(1/"SAVI_35_dist_p5_9.tif"))

## Code Creates Weighted Model 4 using Raster Calculator Tool, following PPM
results

Weighted_Model4 =
((2.5*1/"bdrck_dpt_10m"))+(1.5*(1/"coral_dist_p5_9.tif"))+(1.75*(1/"dune_dist
_p5_9.tif"))+(1.5*(1/"island_dist_p5_9.tif"))+(2*(1/"rock_dist_5_91.tif"))+(1
/"water_dist_p5_9.tif")+(1.75*(1/"SAVI_35_dist_p5_9.tif"))
```

These rasters are assessed against one another in their ability to positively identify known areas with archaeological material. The best performing probability raster is then tested using the PPM procedure detailed above to quantitatively assess its performance against the unweighted model and the Davis et al. (2020) model (PPM1).

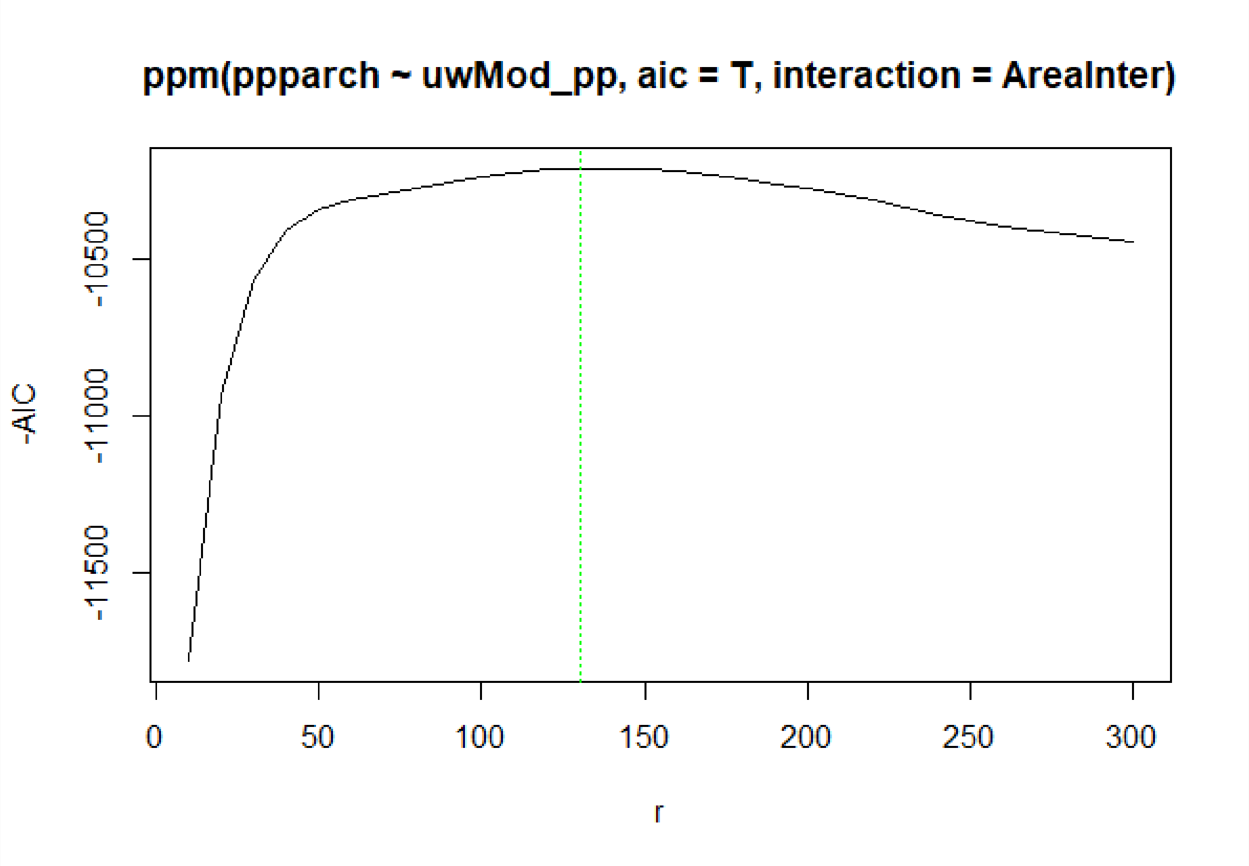
Assess best predictive raster against Weighted Raster

```
#Weighted Raster
w_model <- mask(raster("PPA_Datasets/Weight_Mod5_WGS.tif"), win_ext)
```

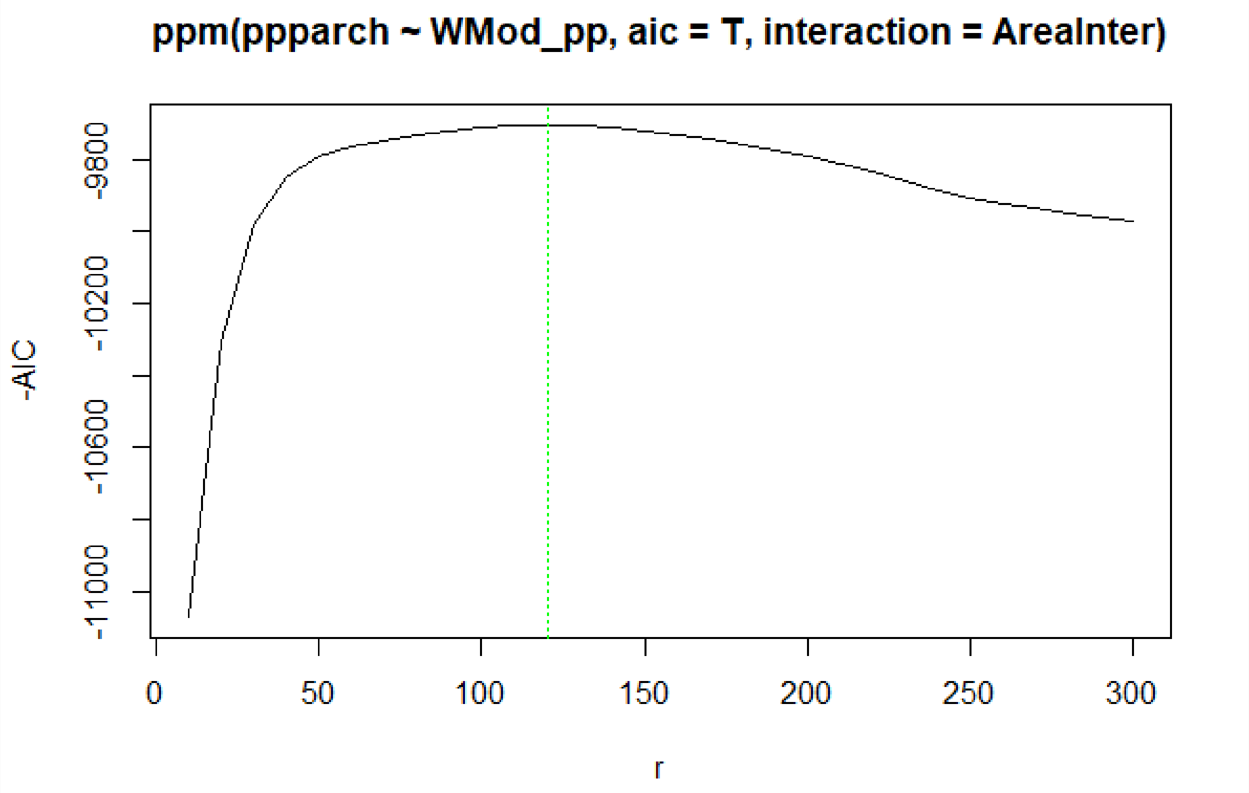
```
WMod_pp <- as.im(w_model)

ppm5 <- ppm(ppparch, ~WMod_pp, correction = "translation")

#Unweighted raster (same as PPM3)
uw_model <- mask(raster("PPA_Datasets/unweight_BIC_WGS1.tif"), win_ext)
uwMod_pp <- as.im(uw_model)
ppm6 <- ppm(ppparch, ~uwMod_pp, correction = "translation")
```

Point Process Modeling of Second-Order Properties

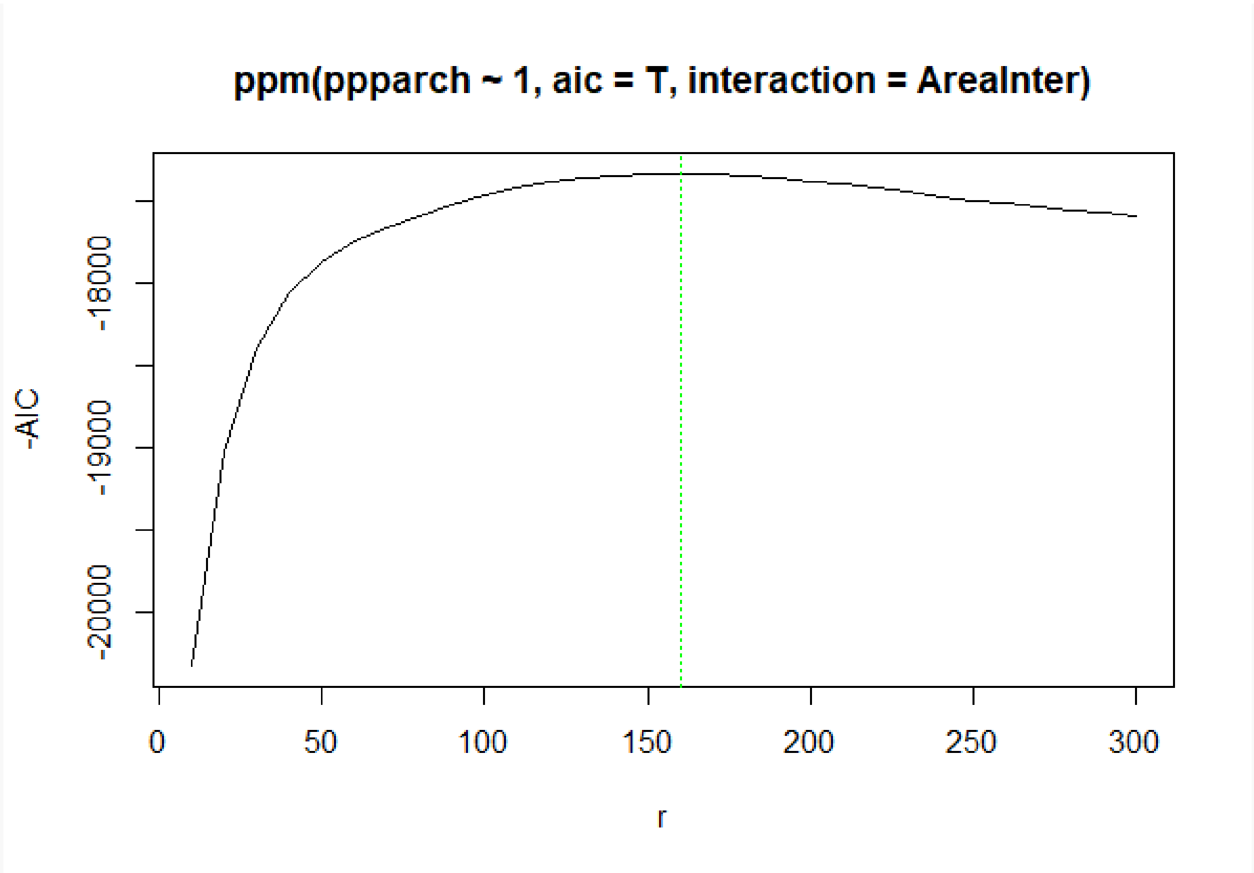
```
##Evaluate Clustering/Dispersion as Model for Settlement Distribution

#Unweighted Model with Area Interaction
area_int2 <- data.frame(r=seq(10, 300, by=10))
p1 <- profilepl(area_int2, AreaInter, ppparch~uwMod_pp, aic=T)

p1

## profile log pseudolikelihood
## for model:  ppm(ppparch ~ uwMod_pp,  aic = T,  interaction = AreaInter)
## fitted with rbord = 600
## interaction: Area-interaction process
## irregular parameter: r in [10, 300]
## optimum value of irregular parameter:  r = 130

plot(p2)
```



```
ppm7 <- as.ppm(p1)

#Weighted Model with Area Interaction
area_int3 <- data.frame(r=seq(10, 300, by=10))
p2 <- profilepl(area_int3, AreaInter, ppparch~WMod_pp, aic=T)

p2

## profile log pseudolikelihood
## for model:  ppm(ppparch ~ WMod_pp,  aic = T,  interaction = AreaInter)
## fitted with rbord = 600
## interaction: Area-interaction process
## irregular parameter: r in [10, 300]
## optimum value of irregular parameter:  r = 120(

plot(p2)
```



```
ppm8 <- as.ppm(p2)


#CSR with Area Interaction

area_int3 <- data.frame(r=seq(10, 300, by=10))
p3 <- profilepl(area_int3, AreaInter, ppparch~1, aic=T)

p3

## profile log pseudolikelihood
## for model:  ppm(ppparch ~ WMod_pp,  aic = T,  interaction = AreaInter)
## fitted with rbord = 600
## interaction: Area-interaction process
## irregular parameter: r in [10, 300]
## optimum value of irregular parameter:  r = 160

Ppm9 <- as.ppm(p3)

plot(p3)
```



```
##Evaluate Fit of Area Interaction Process Models

K_sim1_ppm7 <- envelope(ppm7, Kres, nsim=39, fix.n=T,
correction="translation", global=T)

K_sim1_ppm8 <- envelope(ppm8, Kres, nsim=39, fix.n=T,
correction="translation", global=T)

#Following code produces figure 9

par(mfrow=c(2,1))

plot(K_sim1_ppm7, legend=F, main="K-Function\n Unweighted + Area
Interaction", xlab="Distance (m)")

plot(K_sim1_ppm8, legend=F, main="K-Function\n Weighted + Area Interaction",
xlab="Distance (m)")

par(mfrow=c(1,1))

#Model Selection of second-order PPMs

#Following code produces data in Table 6

MS_AIC2 <- model.sel(ppm0, ppm5, ppm6, ppm7, ppm8, ppm9, rank = AIC)
```

```
MS_AIC2

> MS_AIC2

## Model selection table
##             Q trend correction        interaction rbord df
## ppm8 ppp~W_pp                  l(Ar-int,AI,l(inf,   600  3
## ppm7 ppp~u_pp                  l(Ar-int,AI,l(inf,   600  3
## ppm5      ppp WM_pp translatin                           2
## ppm6      ppp uM_pp translatin                           2
## ppm9    ppp~1                  l(Ar-int,AI,l(inf,   600  2
## ppm0      ppp       translatin                           1
##          logLik     AIC    delta weight
## ppm8  -4849.373  9703.7     0.00      1
## ppm7  -5103.432 10211.7   507.96      0
## ppm5  -6038.332 12080.7  2376.92      0
## ppm6  -6465.358 12934.7  3230.97      0
## ppm9  -8666.598 17336.0  7632.29      0
## ppm0 -11438.133 22878.3 13174.52      0
## Abbreviations:
## Q: ppp = ‘ppparch’, ppp~1 = ‘ppparch~1’,
##    ppp~u_pp = ‘ppparch~uwMod_pp’,
##    ppp~W_pp = ‘ppparch~WMod_pp’
## trend:  = ‘~1’, uM_pp = ‘~uwMod_pp’, WM_pp = ‘~WMod_pp’
## correction: translatin = ‘translation’
## interaction: l(Ar-int,AI,l(inf, = ‘list(Area-
## interactionprocess,AreaInter,list(inforder,’
## Models ranked by AIC(x)


MS_BIC2 <- model.sel(ppm0, ppm5, ppm6, ppm7, ppm8, ppm9, rank = BIC)

MS_BIC2

## Model selection table
##             Q trend correction        interaction rbord df
## ppm8 ppp~W_pp                  l(Ar-int,AI,l(inf,   600  3
## ppm7 ppp~u_pp                  l(Ar-int,AI,l(inf,   600  3
## ppm5      ppp WM_pp translatin                           2
## ppm6      ppp uM_pp translatin                           2
## ppm9    ppp~1                  l(Ar-int,AI,l(inf,   600  2
## ppm0      ppp       translatin                           1
##          logLik     BIC    delta weight
## ppm8  -4849.373  9718.8     0.00      1
## ppm7  -5103.432 10226.9   508.12      0
## ppm5  -6038.332 12090.0  2371.23      0
## ppm6  -6465.358 12944.1  3225.28      0
## ppm9  -8666.598 17346.6  7627.76      0
## ppm0 -11438.133 22883.0 13164.14      0
## Abbreviations:
## Q: ppp = ‘ppparch’, ppp~1 = ‘ppparch~1’,
##    ppp~u_pp = ‘ppparch~uwMod_pp’,
 ##   ppp~W_pp = ‘ppparch~WMod_pp’
## trend:  = ‘~1’, uM_pp = ‘~uwMod_pp’, WM_pp = ‘~WMod_pp’
## correction: translatin = ‘translation’
## interaction: l(Ar-int,AI,l(inf, = ‘list(Area-
## interactionprocess,AreaInter,list(inforder,’
## Models ranked by BIC(x)
```