Simulating scenarios: comparing the the influence of the nearest feature with the cumulative distance

Bernardo Niebuhr, Bram van Moorter

19 January, 2022

Contents

1 Introduction

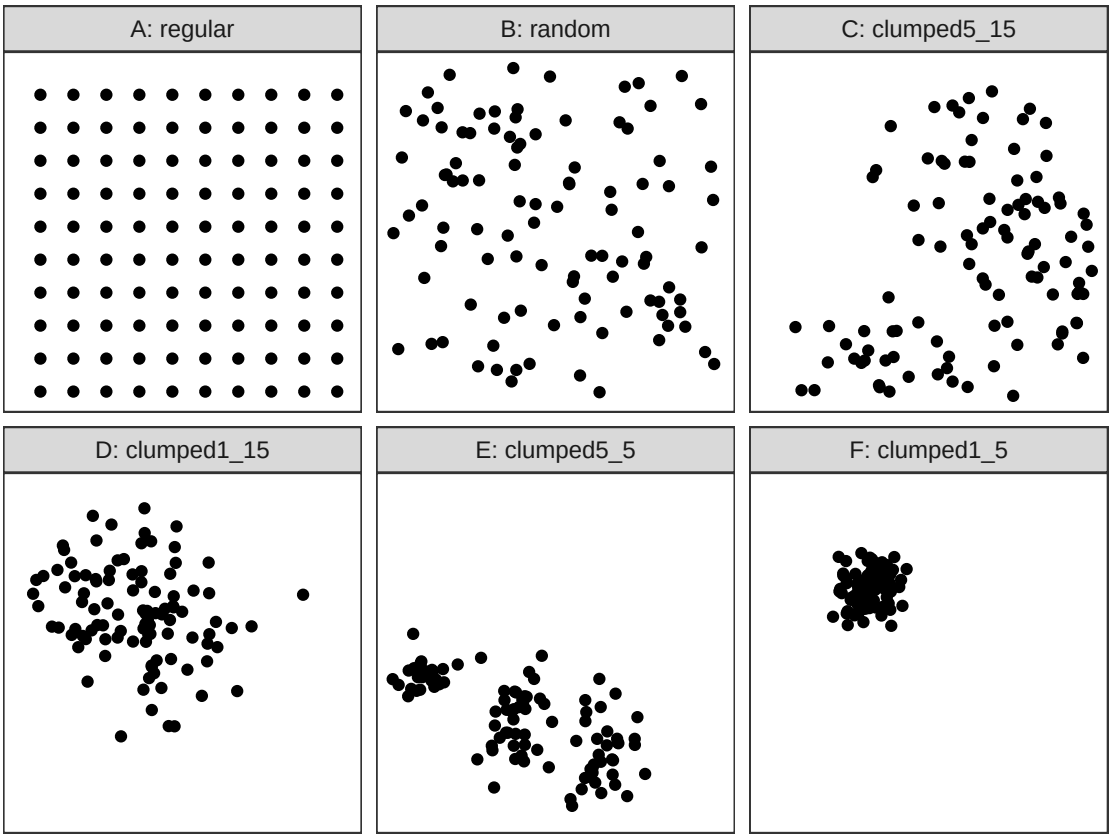
Influence: Bartlett

```
# Load packages

# data manipulation
library(dplyr)
library(purrr)
library(ggplot2) # for plot
library(ggpubr) # for plot
library(lsr) # for correlations

# spatial packages
library(mobsim)
library(terra)
library(landscapetools)
library(sf)
library(spatstat)

# oneimpact package
library(oneimpact)
```

```
# Set parameters for the simulation
set.seed(12)

# random, gradient, single center, multiple centers
methods <- c("regular", "random", "mobsim")
name <- c("regular", "random", "clumped5_15", "clumped1_15", "clumped5_5", "clumped1_5")
nfeat <- c(100) # number of features
res <- 10 # resolution
ext <- 30000 # extent of the landscape
nc <- c(5, 1) # number of centers or clusters for clumped
wd <- c(0.15, 0.05) * ext # width of the "clusters"

# ZoI values
zoi_vals <- c(20, 50, 100, 250, 500, 750, 1000, 1250, 1500, 2000, 3000, 4000, 5000, 6000, 8000, 10000, 120

# parameters
parms_df1 <- expand.grid(
  method = methods, n_features = nfeat,
  centers = nc[1], width = wd
```

```
) # first 3 scenarios
parms_df2 <- expand.grid(
  method = methods[3], n_features = nfeat,
  centers = nc[2], width = wd
) # parameters for clumped1
parms_df <- dplyr::bind_rows(parms_df1, parms_df2) %>%
  dplyr::arrange(desc(width), n_features, method) %>%
  dplyr::slice(1:4, 7:8)
```

```
# simulate points
pts_10 <- parms_df %>%
  purrr::pmap(set_points,
    res = res,
    extent_x = c(0, ext),
    extent_y = c(0, ext),
    buffer_around = 12000
  )

# coordinates
pts_coords <- pts_10 %>%
  purrr::map(first)

# raster with points
pts_all_rast_10 <- pts_10 %>%
  purrr::map(~ .[[2]])

pts_all_rast_100 <- pts_all_rast_10 %>%
  purrr::map(terra::aggregate, fact = 10, fun = "max", na.rm = TRUE)

# vector with points, for plotting
pts_vect_list <- pts_coords %>%
  purrr::map(sf::st_as_sf, coords = c(1:2), crs = terra::crs(pts_all_rast_10[[1]]))
for (i in 1:length(pts_vect_list)) pts_vect_list[[i]]$spatial_dist <- name[i]
pts_vect <- dplyr::bind_rows(pts_vect_list) %>%
  dplyr::mutate(
    spatial_dist = factor(spatial_dist, levels = name),
    labels = paste(LETTERS[spatial_dist], spatial_dist, sep = ": ")
  )

# landscape delimitation
landscape <- st_bbox(c(xmin = 0, xmax = ext, ymax = 0, ymin = ext), crs = terra::crs(pts_all_rast_10[[1]]))
  st_as_sfc()

# plot
pts_vect %>%
  ggplot() +
  # geom_sf(data = landscape) +
  geom_sf() +
  facet_wrap(~labels) +
  theme_bw() +
  theme(
    panel.grid.minor = element_blank(), panel.grid.major = element_blank(),
    axis.text = element_blank(), axis.ticks = element_blank()
  )
```



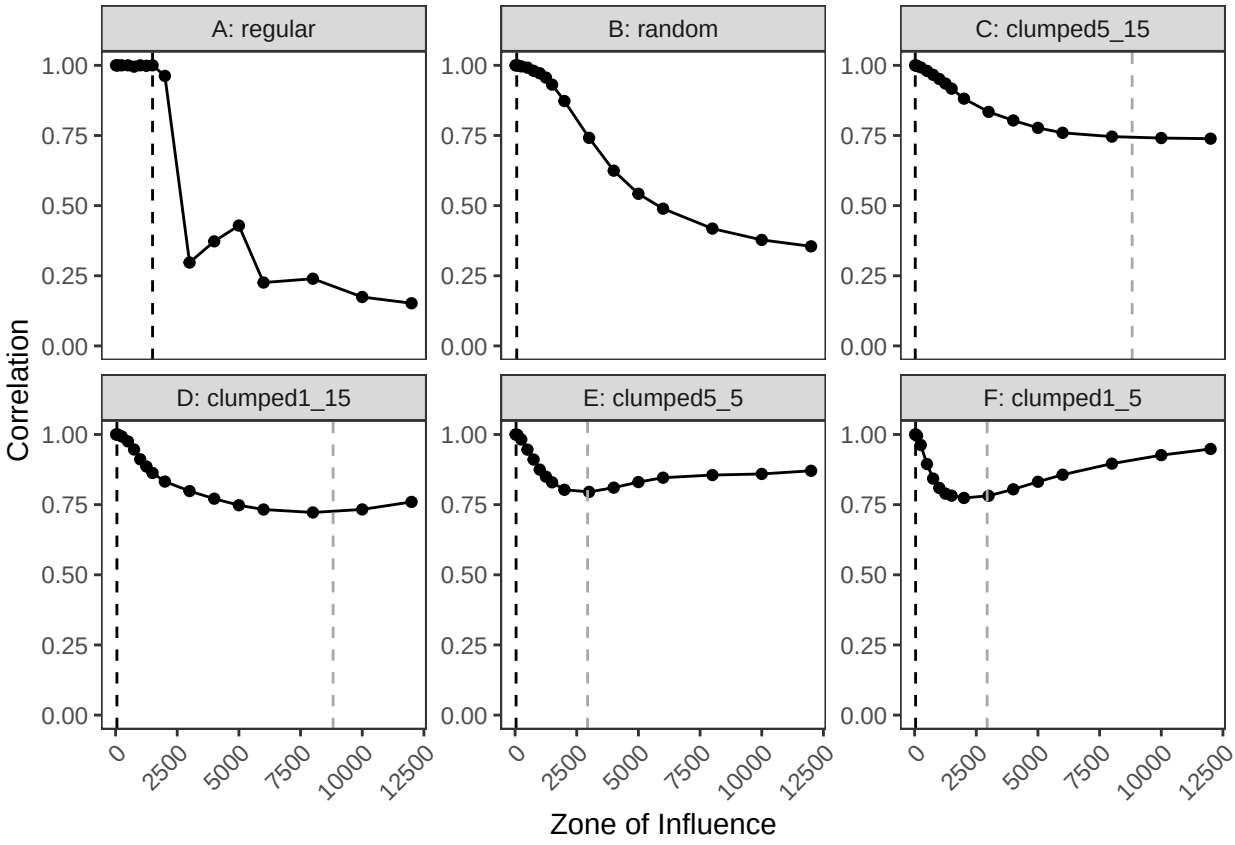

```
# crate scenarios, separating each point in a different
# raster, to calculate distances
create_scenarios <- tibble::tibble(
  spatial_dist = name,
  n_features = rep(nfeat, length(name)),
  # mean isolation
  mean_isolation = map_dbl(pts_coords, mean_isolation, n_rand = 150, ext = c(0, ext)),
  # mean distance between points
  mean_dist = map_dbl(pts_coords, function(x) mean(dist(x))),
  # mean nearest neighbor distance
  mean_nndist = map_dbl(pts_coords, function(x) mean(spatstat.geom::nndist(x))),
  # points coordinates, all together
  points_10 = pts_all_rast_10,
  points_100 = pts_all_rast_100
)

# simplify tibble with scenarios and distances
scenarios_dist <- create_scenarios %>%
  dplyr::mutate(
    parms = purrr::map(spatial_dist, function(x) {
      tibble::tibble(zoi = zoi_vals)
    })
  ) %>%
  tidyr::unnest(parms) %>%
  dplyr::mutate(points_rast = ifelse(zoi > 500, points_100, points_10)) %>%
  dplyr::select(-c(points_10, points_100))

# map(scenarios_dist$points_rast, terra::res)
```

```r
filter_type <- "bartlett"

# create filter matrices
parms_filter <- expand.grid(method = filter_type, zoi = zoi_vals, scenario = name) %>%
  dplyr::select(-scenario) %>%
  dplyr::mutate(res = map(scenarios_dist$points_rast, function(x) terra::res(x)[1])) %>%
  tidyr::unnest(res)
scenarios_dist$filter_matrices <- parms_filter %>% pmap(create_filter)

# calculate cumulative influence
maps_cuminf <- purrr::map2(scenarios_dist$points_rast, scenarios_dist$filter_matrices, function(x, y) {
  print("running...")
  calc_influence_cumulative(x = x, type = "mfilter", zoi = y)
})

maps_cuminf <- map(maps_cuminf, terra::crop, y = terra::ext(c(c(0, ext), c(0, ext))))
```

```r
# calculate cumulative influence
maps_nearestinf <- purrr::map2(scenarios_dist$points_rast, scenarios_dist$zoi, function(x, y) {
  print("running...")
  calc_influence_nearest(x = x, transform = filter_type, zoi = y)
})

maps_nearestinf <- map(maps_nearestinf, terra::crop, y = terra::ext(c(c(0, ext), c(0, ext))))
```

```r
scenarios_dist <- scenarios_dist %>%
  dplyr::mutate(
    rast_nearest = maps_nearestinf,
    rast_cuminf = maps_cuminf,
    corr = purrr::map2(rast_nearest, rast_cuminf, function(x, y) {
      cor(terra::values(x), terra::values(y))
    })
  ) %>%
  tidyr::unnest(corr) %>%
  dplyr::mutate(spatial_dist = factor(spatial_dist,
    levels = name[c(1, 2, 3, 4, 5, 6)],
    labels = paste(LETTERS[1:length(name)], name[c(1, 2, 3, 4, 5, 6)], sep = ": ")
  ))

# plot
dat_isol <- scenarios_dist %>%
  dplyr::group_by(n_features, spatial_dist) %>%
  dplyr::summarise(
    mean_iso = unique(mean_isolation),
    mean_dist = unique(mean_dist),
    mean_nndist = unique(mean_nndist)
  ) %>%
  dplyr::mutate(
    min_dist = map_dbl(pts_coords, function(x) min(dist(x))),
    max_dist = map_dbl(pts_coords, function(x) max(dist(x))),
    min_nndist = map_dbl(pts_coords, function(x) min(spatstat.geom::nndist(x))),
    iso = map_dbl(pts_coords, mean_isolation, n_rand = 150, ext = c(0, ext)),
    size_cluster = c(NA, NA, wd[1], wd[1], wd[2], wd[2])
  )

scenarios_dist %>%
```

```
# dplyr::filter(spatial_dist == "A: regular") %>%
ggplot(aes(zoi, abs(corr))) +
geom_point() +
geom_line() +
geom_vline(data = dat_isol, aes(xintercept = min_dist / 2), linetype = 2) +
geom_vline(data = dat_isol, aes(xintercept = size_cluster * 1.96), linetype = 2, color = "darkgrey") +
ylim(00, 1) +
facet_wrap(~spatial_dist, scales = "free_y") +
labs(
  x = "Zone of Influence",
  y = "Correlation",
  color = "Spatial distribution",
  linetype = "Mean n.n. distance"
) +
theme_bw() +
theme(
  axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1),
  panel.grid.minor = element_blank(), panel.grid.major = element_blank()
) #+
```



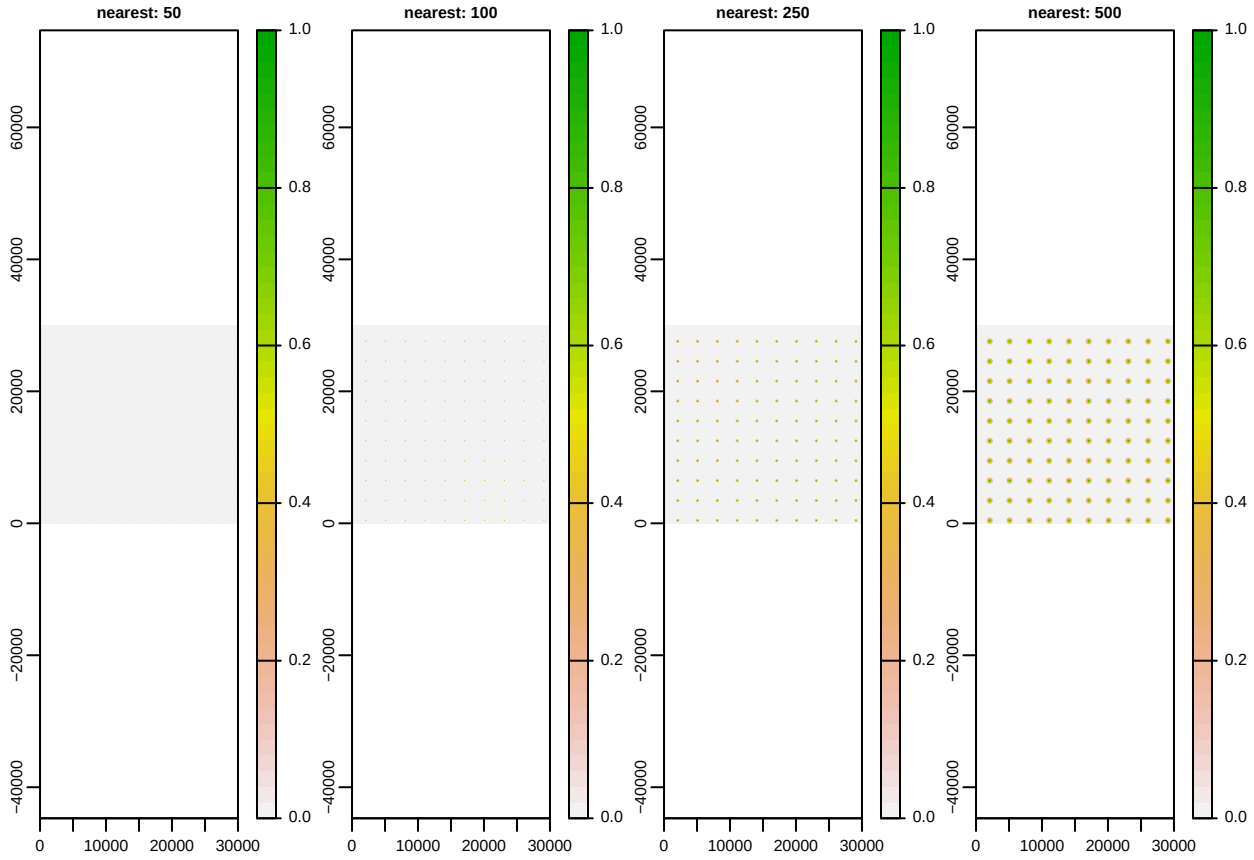

```
scale_x_continuous(trans = "log10")
```

```
## <ScaleContinuousPosition>
##  Range:
##  Limits:    0 --    1
```

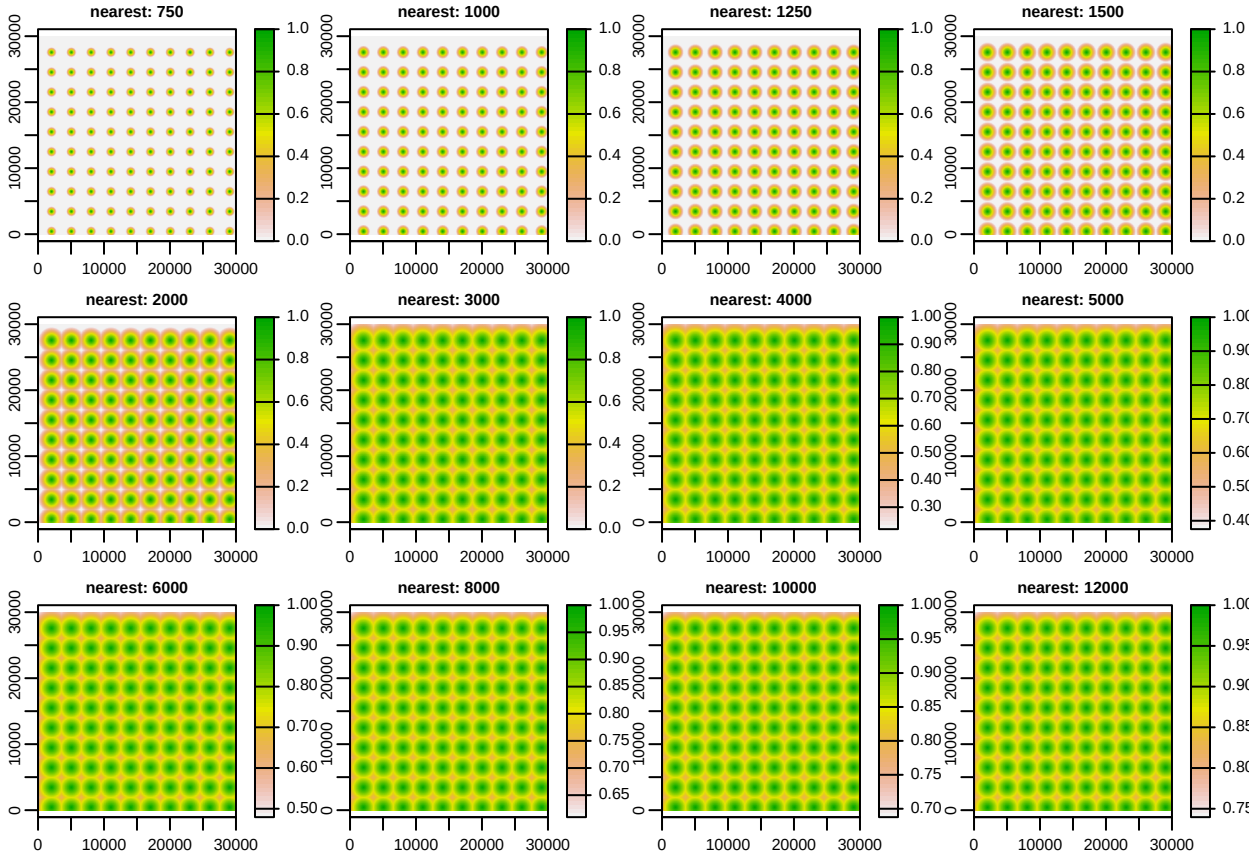
The point of inflection is related to the size of the clusters - when the ZoI gets to this value, the correlation between nearest and cumulative influence is minimum. Beyond that, the correlation starts to grow again, because the influence of the cluster of features starts to increase far from the cluster in a way that is similar between the two

influence measures. However, when there are several clusters (panel C and E, instead of only one cluster), there might be some variation in this value, and the curve might increase in a non-linear way, depending on the distance between clusters (e.g. distance between villages or urban centers).

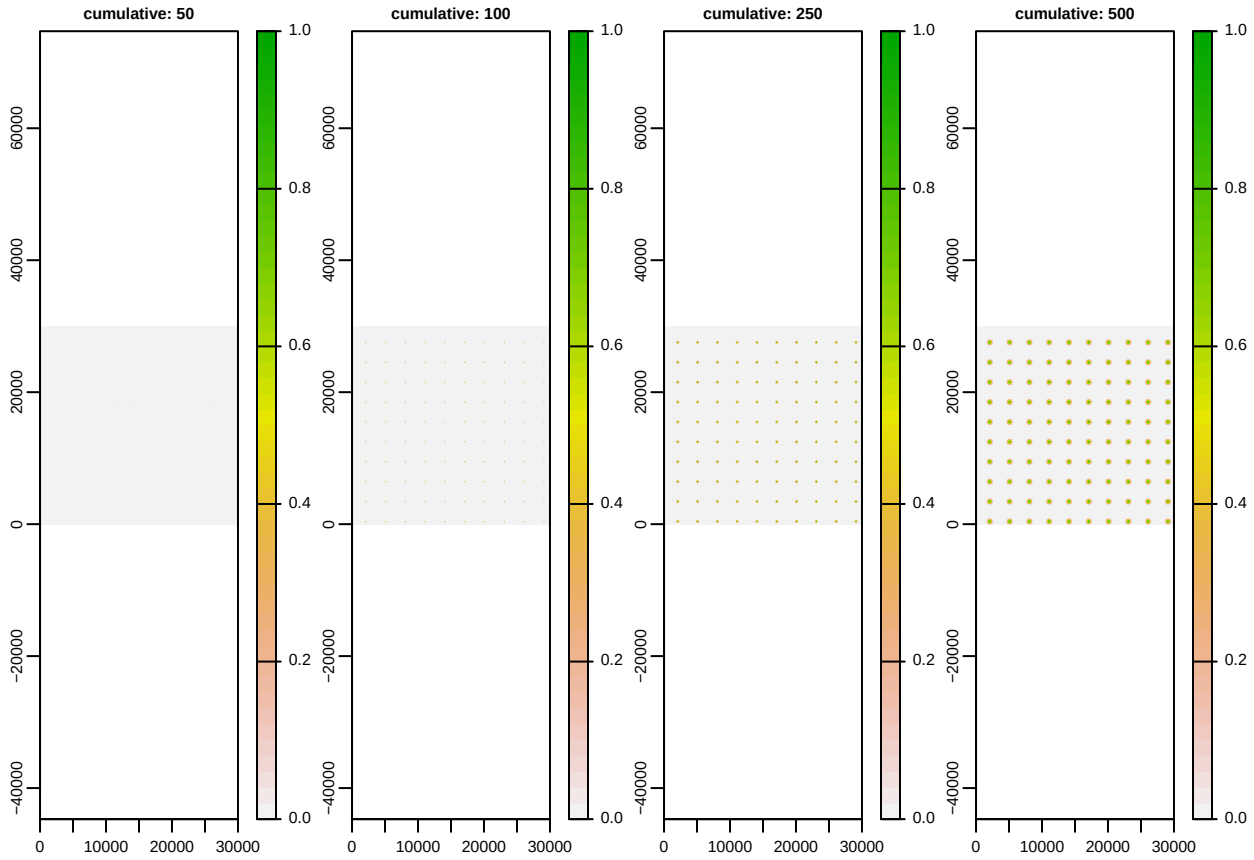
```
which_scenarios <- c(0)
indices1 <- which_scenarios * length(zoi_vals) + 2:5
indices2 <- which_scenarios * length(zoi_vals) + 6:length(zoi_vals)
to_viz1 <- do.call(c, scenarios_dist$rast_nearest[indices1])
names(to_viz1) <- paste(rep("nearest", nlyr(to_viz1)), zoi_vals[2:5],
  sep = ": "
)
plot(to_viz1, nc = 4)
```



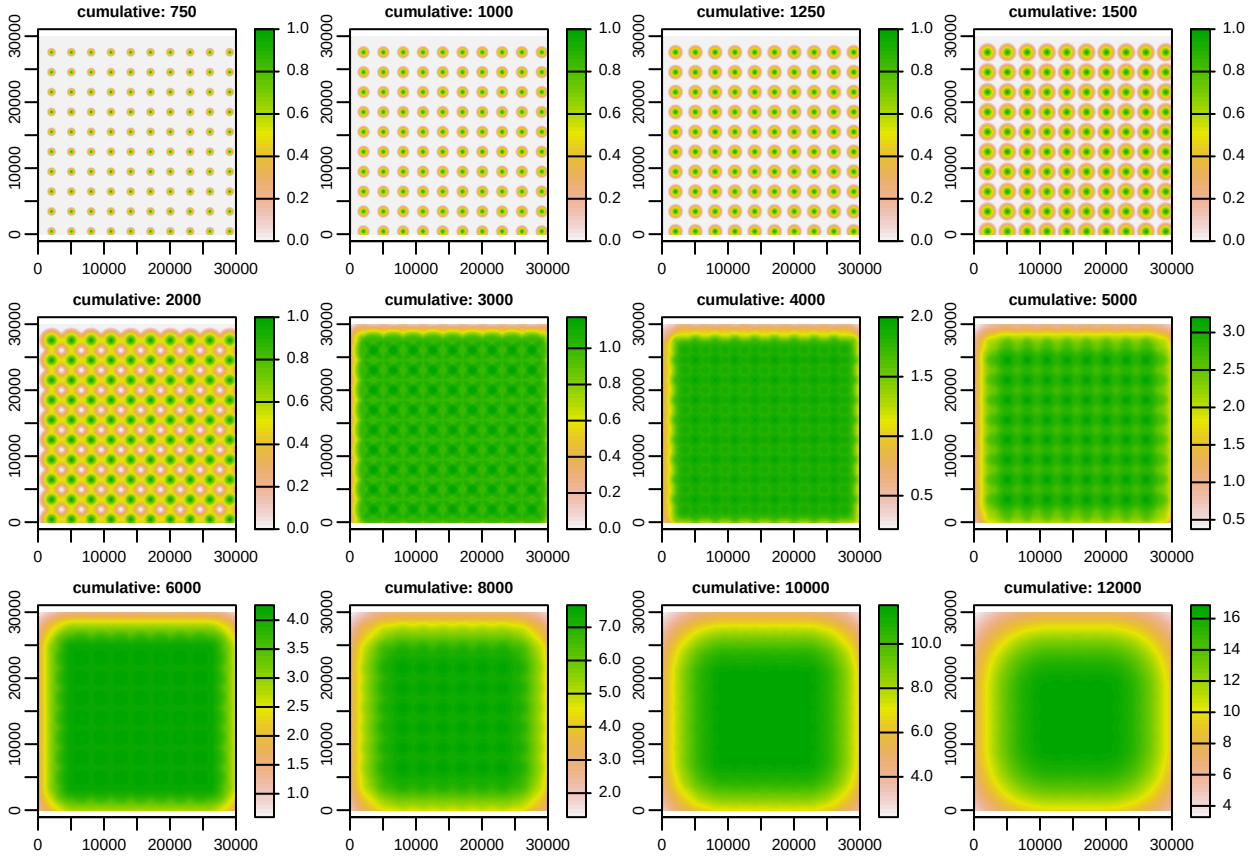

```
# show only biger scales
to_viz2 <- do.call(c, scenarios_dist$rast_nearest[indices2])
names(to_viz2) <- paste(rep("nearest", nlyr(to_viz2)), zoi_vals[6:length(zoi_vals)],
  sep = ": "
)
plot(to_viz2, nc = 4)
```



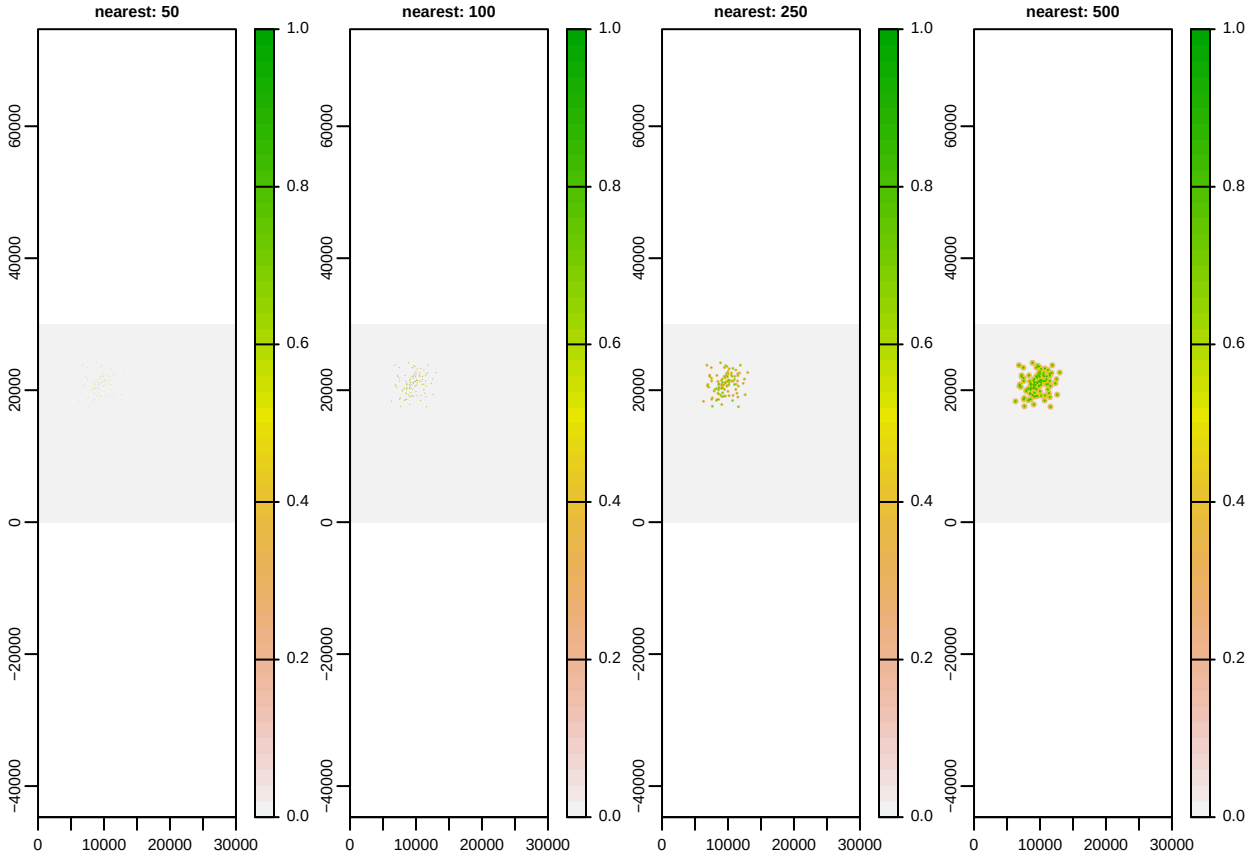

```
which_scenarios <- c(0)
indices1 <- which_scenarios * length(zoi_vals) + 2:5
indices2 <- which_scenarios * length(zoi_vals) + 6:length(zoi_vals)
to_viz1 <- do.call(c, scenarios_dist$rast_cuminf[indices1])
names(to_viz1) <- paste(rep("cumulative", nlyr(to_viz1)), zoi_vals[2:5],
  sep = ": "
)
plot(to_viz1, nc = 4)
```



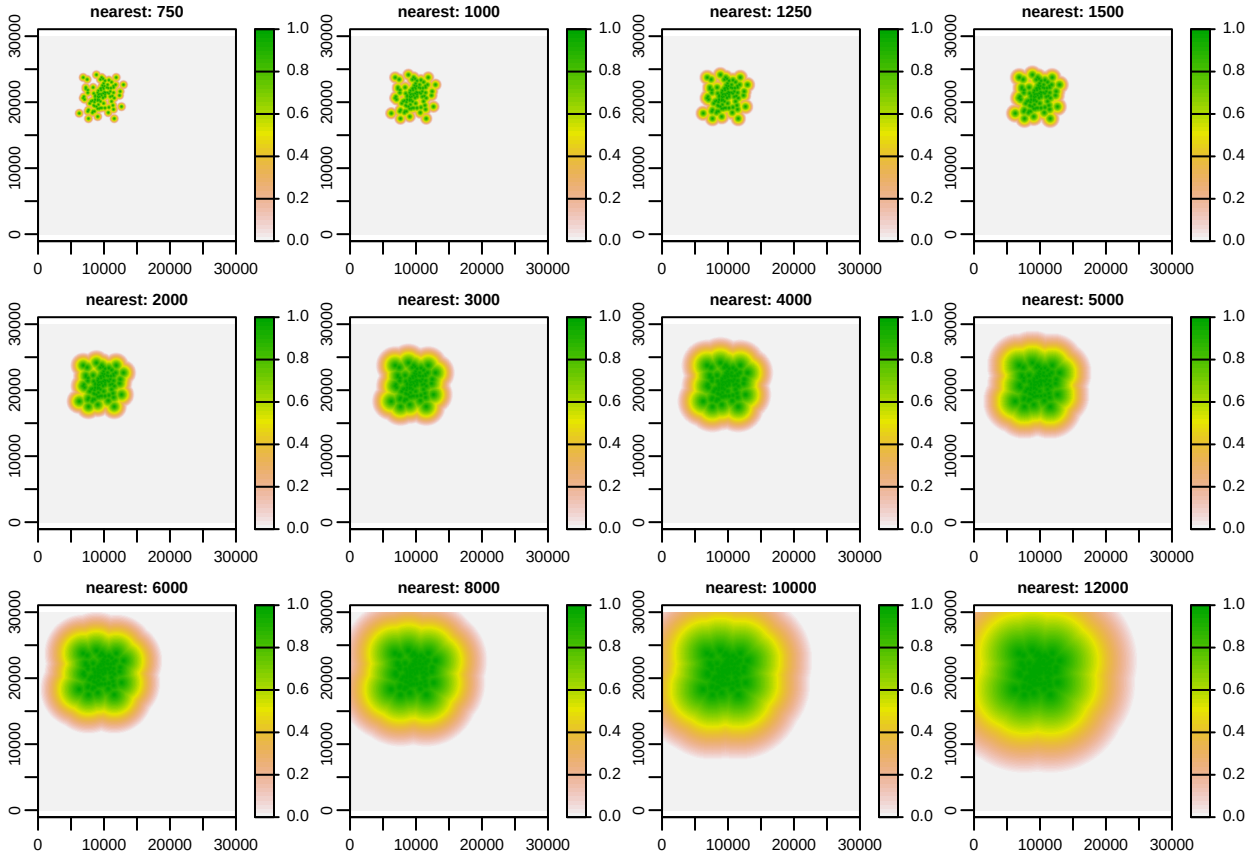

```
# show only biger scales
to_viz2 <- do.call(c, scenarios_dist$rast_cuminf[indices2])
names(to_viz2) <- paste(rep("cumulative", nlyr(to_viz2)), zoi_vals[6:length(zoi_vals)],
  sep = ": "
)
plot(to_viz2, nc = 4)
```



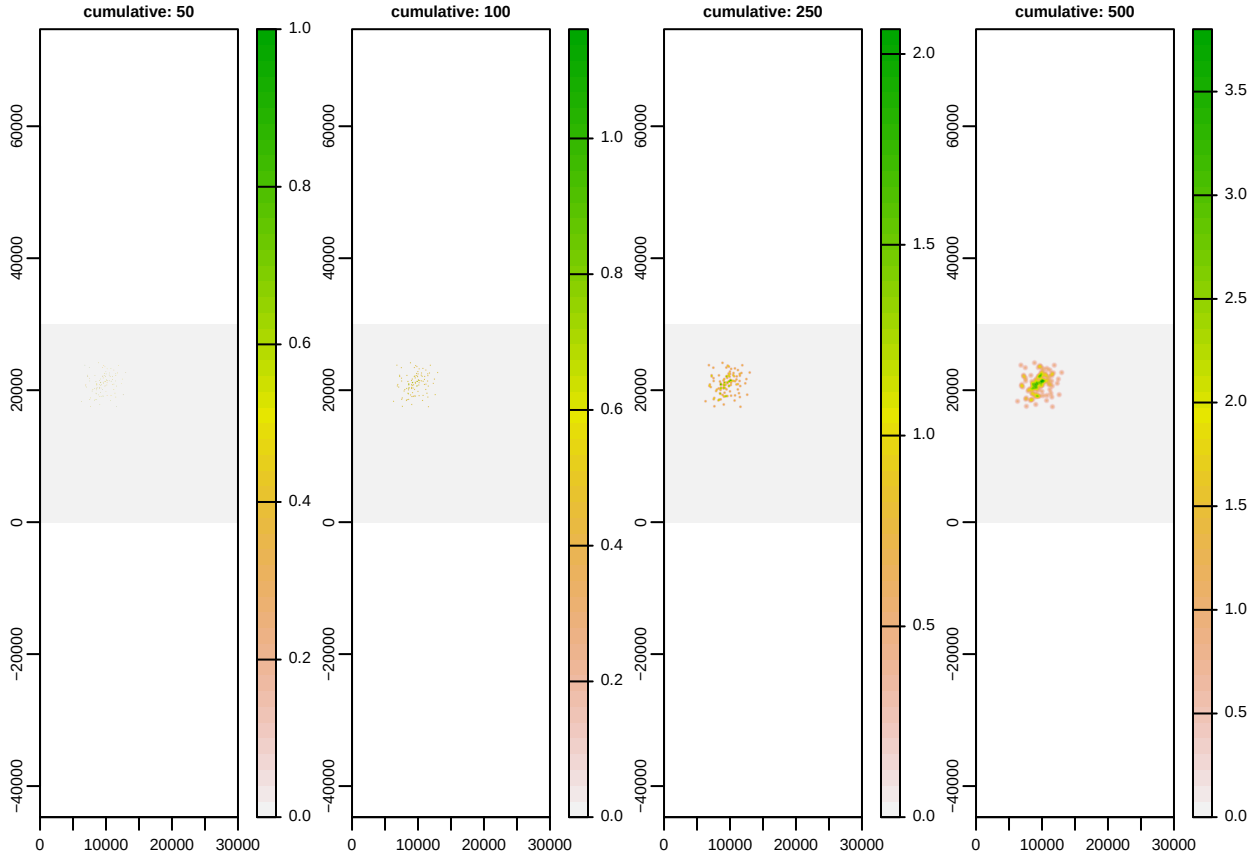

```
which_scenarios <- c(5)
indices1 <- which_scenarios * length(zoi_vals) + 2:5
indices2 <- which_scenarios * length(zoi_vals) + 6:length(zoi_vals)

to_viz1 <- do.call(c, scenarios_dist$rast_nearest[indices1])
names(to_viz1) <- paste(rep("nearest", nlyr(to_viz1)), zoi_vals[2:5],
  sep = ": "
)
plot(to_viz1, nc = 4)
```



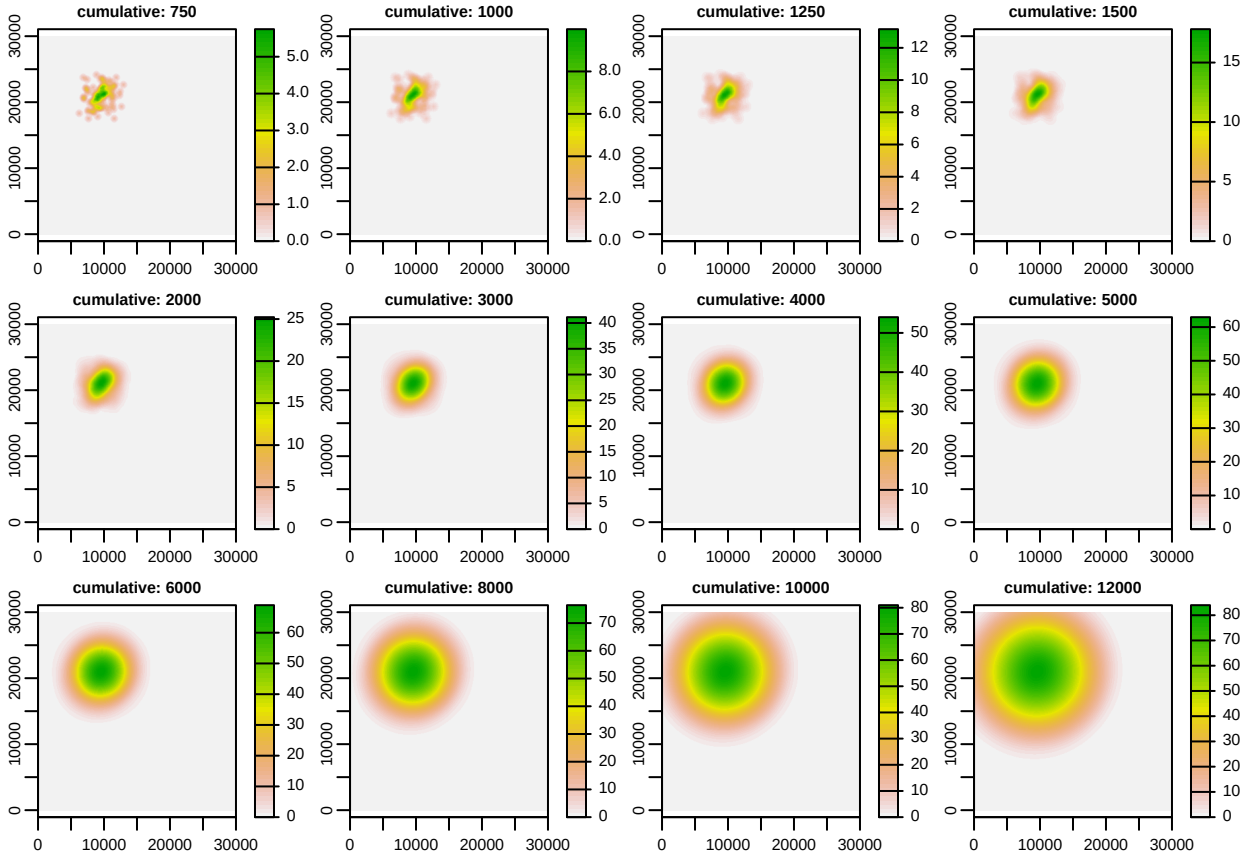

```
# show only biger scales
to_viz2 <- do.call(c, scenarios_dist$rast_nearest[indices2])
names(to_viz2) <- paste(rep("nearest", nlyr(to_viz2)), zoi_vals[6:length(zoi_vals)],
  sep = ": "
)
plot(to_viz2, nc = 4)
```




```
which_scenarios <- c(5)
indices1 <- which_scenarios * length(zoi_vals) + 2:5
indices2 <- which_scenarios * length(zoi_vals) + 6:length(zoi_vals)
to_viz1 <- do.call(c, scenarios_dist$rast_cuminf[indices1])
names(to_viz1) <- paste(rep("cumulative", nlyr(to_viz1)), zoi_vals[2:5],
  sep = ": "
)
plot(to_viz1, nc = 4)
```

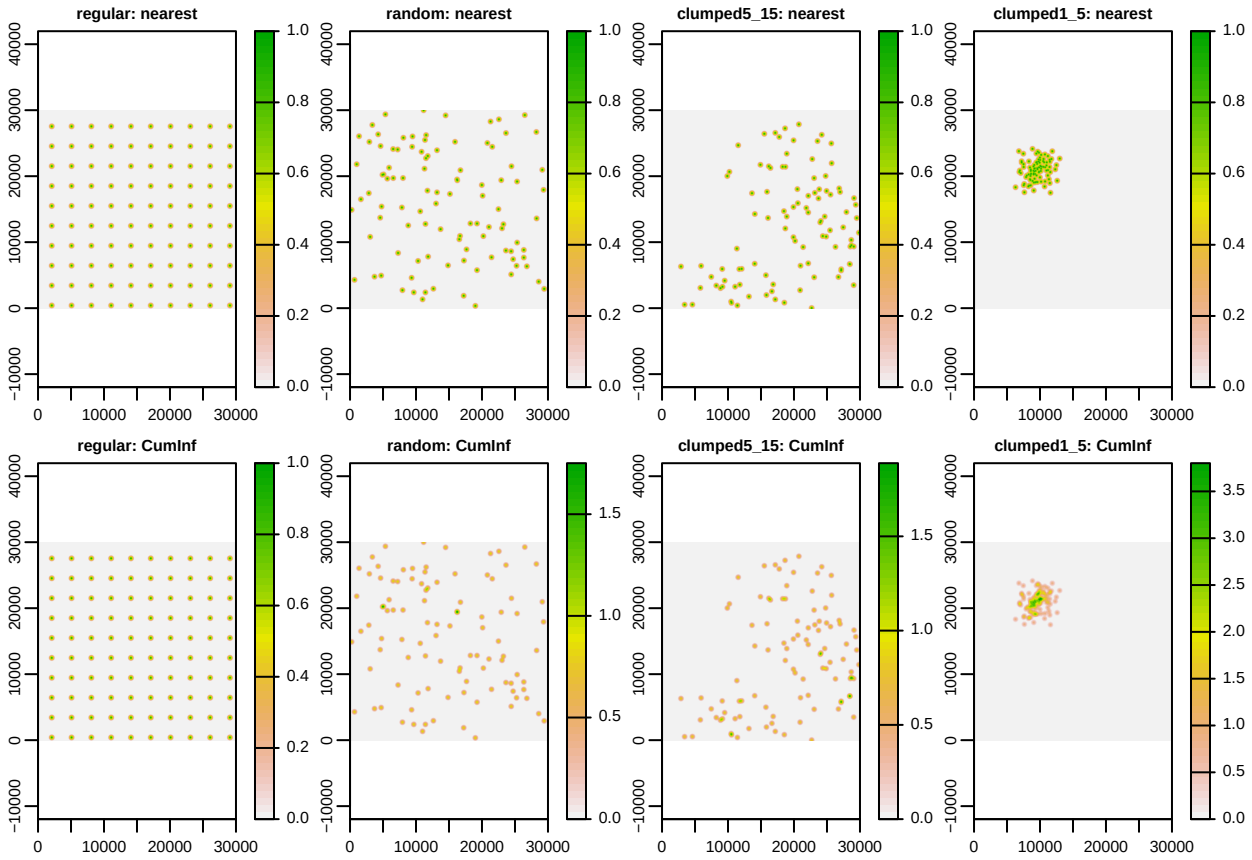



```
# show only biger scales
to_viz2 <- do.call(c, scenarios_dist$rast_cuminf[indices2])
names(to_viz2) <- paste(rep("cumulative", nlyr(to_viz2)), zoi_vals[6:length(zoi_vals)],
  sep = ": "
)
plot(to_viz2, nc = 4)
```

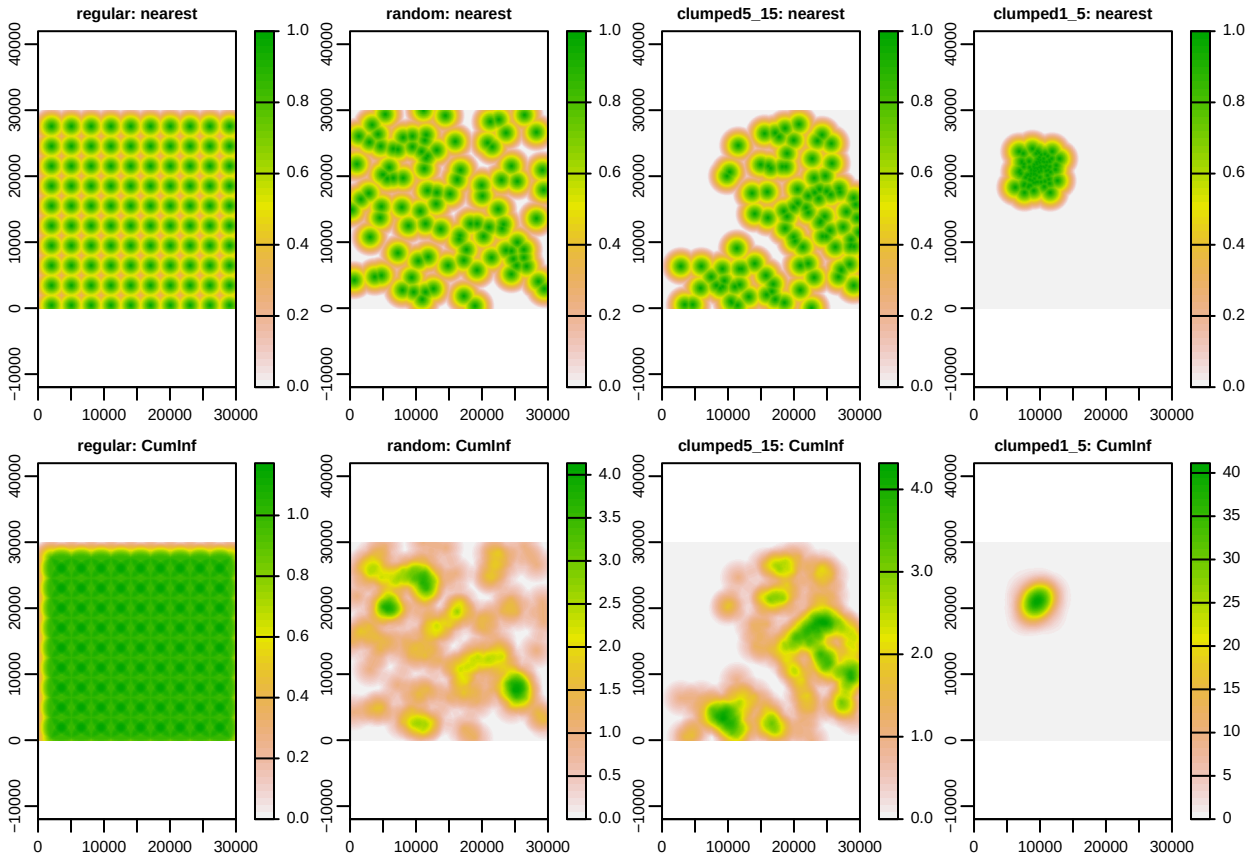


Visualize several scenarios

```
which_scenarios <- c(0:2, 5)
indices <- which_scenarios * length(zoi_vals) + 5
to_viz <- do.call(c, c(scenarios_dist$rast_nearest[indices], scenarios_dist$rast_cuminf[indices]))
names(to_viz) <- paste(rep(name[which_scenarios + 1], 2),
  rep(c("nearest", "CumInf"), each = 4),
  sep = ": "
)

plot(to_viz, nc = 4)
```




```
which_scenarios <- c(0:2, 5)
indices <- which_scenarios * length(zoi_vals) + 11
to_viz <- do.call(c, c(scenarios_dist$rast_nearest[indices], scenarios_dist$rast_cuminf[indices]))
names(to_viz) <- paste(rep(name[which_scenarios + 1], 2),
  rep(c("nearest", "CumInf"), each = 4),
  sep = ": "
)

plot(to_viz, nc = 4)
```


regular: nearest
random: nearest
clumped5_15: nearest
clumped1_5: nearest
regular: CumInf
random: CumInf
clumped5_15: CumInf
clumped1_5: CumInf