Density in execution time(s)

```
0    5   10   15   20   25   30   35
```
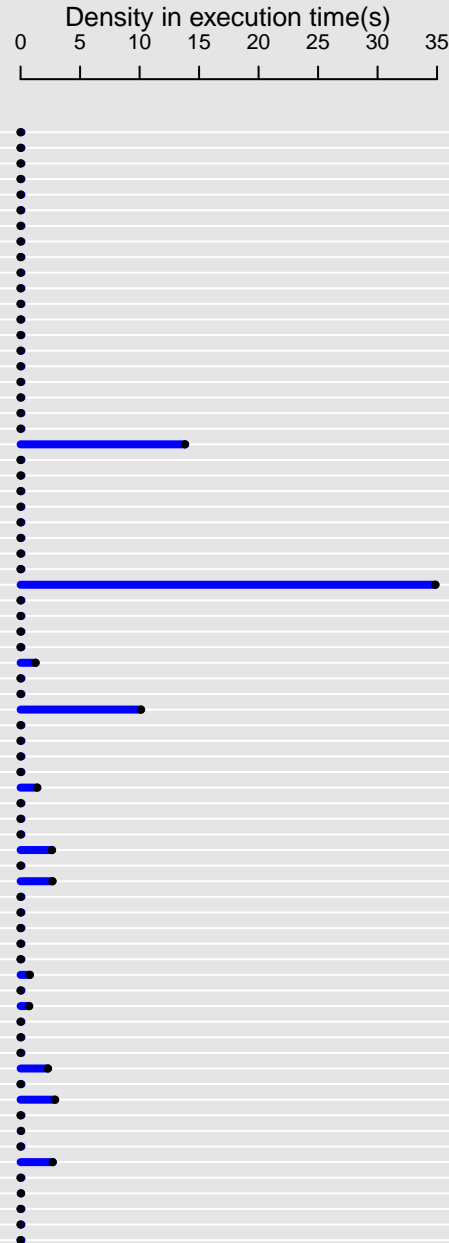
```r
 1  generatePatchySpecies2 <-
 2  function(nb.sim = 1, nb.sp = 20)
 3  {
 4    richness <- stack()
 5    richness.patch <- stack()
 6    for (j in 1:nb.sim)
 7    {
 8      sp.traits <- data.frame(T.optimum = sample(temperature.gradient,
 9                              nb.sp, replace = T),
10                  T.tolerance = sample(seq(50, 100,
11                        length = 1000),
12                        nb.sp, replace = T))
13      species <- list() # All the species will be stored in this list
14      range.stack <- stack() # Species ranges will be stored in this stack
15      range.patch.stack <- stack() # Species ranges with patched distributions will be stored in this stack
16
17      for (i in 1:nb.sp)
18      {
19        # Step 1: generating the species environmental suitability
20        # (i.e., transforming from E-space to G-space)
21        species[[i]] <- generateSpFromFun(bio1,
22                        list(bio1 = list(fun = "gauss.resp",
23                                args = list(mean. = sp.traits[i, "T.optimum"],
24                                      diff. = sp.traits[i, "T.tolerance"],
25                                      prob. = 0.99))),
26                        rescale.each.response = FALSE,
27                        rescale = FALSE)
28        # Step 2: converting environmental suitability to presence-absence,
29        # using the probabilistic conversion
30        species[[i]] <- convertToPA(species[[i]], # PA.method = "threshold",
31                        beta = 0.7, alpha = -0.05,
32                        plot = FALSE)
33
34        # Step 2.5: generate habitat patches
35        if(cellStats(species[[i]]$pa.raster, stat = 'max') == 1 &
36          length(Which(species[[i]]$pa.raster == 1, cells = T)) > 500)
37        {
38          species[[i]]$patched.pa.raster <- generate.patches(bio1, n.patches = 10, patch.size = 10,
39                        starts = sample(Which(species[[i]]$pa.raster == 1, cells = T), 10))
40        } else
41        {
42          species[[i]]$patched.pa.raster <- species[[i]]$pa.raster
43          species[[i]]$patched.pa.raster[!is.na(species[[i]]$patched.pa.raster)] <- 0 # To avoid species with less than 500 cells & range cohesion
44        }
45
46        # Step 3: we store ranges to calculate richness par cell
47        range.stack <- addLayer(range.stack,
48                    species[[i]]$pa.raster)
49        range.patch.stack <- addLayer(range.patch.stack,
50                    species[[i]]$patched.pa.raster)
51      }
52
53
54      # Richness calculation
55      richness <- addLayer(richness,
56                  sum(range.stack))
57      richness.patch <- addLayer(richness.patch,
58                    sum(range.patch.stack))
59
60      # Saving the files
61      save(species, file = paste0("./data/S1_sim", j, "_species"))
62      save(sp.traits, file = paste0("./data/S1_sim", j, "_traits"))
63      writeRaster(range.stack, paste0("./data/S1_sim", j, "_rangestacks"),
64                  overwrite = T)
65      writeRaster(richness, paste0("./data/S1_sim", j, "_richness"),
66                  overwrite = T)
67      writeRaster(range.patch.stack, paste0("./data/S1_sim", j, "_rangestacks_patch"),
68                  overwrite = T)
69      writeRaster(richness.patch, paste0("./data/S1_sim", j, "_richness_patch"),
70                  overwrite = T)
71    }
72  }
```