Quickly visualize solutions

2023-07-26

Take a glimpse at one of the solutions (reproduce Figure 1 panel B)!

Read in one of the solution

```
sol <- list.files("../data-formatted/sol/", pattern = "globiomICflat_gurobi_f455.csv", full.names = T)
solution <- read_csv(sol[[3]]) |>
  dplyr::select(id, solution_1_z1:solution_1_z26)
```

```
## Rows: 41046 Columns: 53
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (53): z1, z2, z3, z4, z5, z6, z7, z8, z9, z10, z11, z12, z13, z14, z15, ...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
ic <- read_csv("../data/outputs/2-zones/PU_lc_intensity.csv") |> rename(pu = PUID)
```

```
## Rows: 65704 Columns: 17
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Status
## dbl (16): PUID, HeathlandShrub_natural, Grassland_natural, SparseVeg_natural...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
pu_in_EU <- read_csv("../data-formatted/pu_in_EU.csv")
```

```
## Rows: 41046 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): pu, nuts2id, EU_id
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
ic <- ic |>
  left_join(pu_in_EU) |>
  rename(id = EU_id) |>
  dplyr::select(-c(pu, nuts2id)) |>
  drop_na(id)
```

```
## Joining with ‘by = join_by(pu)‘
```

```
zone_id <- read_csv("../data-formatted/zone_id.csv")
```

```
## Rows: 26 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): zone
## dbl (1): id
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
zone_id$zone
```

```
##  [1] "Cropland_low_restore"                 "Cropland_med_restore"
##  [3] "Grassland_natural_restore"            "HeathlandShrub_natural_restore"
##  [5] "MarineTransitional_natural_restore"   "Pasture_low_restore"
##  [7] "SparseVeg_natural_restore"            "Wetlands_natural_restore"
##  [9] "WoodlandForest_multi_restore"         "WoodlandForest_primary_restore"
## [11] "WoodlandForest_prod_restore"          "WoodlandForest_multi_production"
## [13] "WoodlandForest_prod_production"       "Pasture_high_production"
## [15] "Pasture_low_production"               "Cropland_med_production"
## [17] "Cropland_low_production"              "Cropland_high_production"
## [19] "HeathlandShrub_natural_conserve"      "Grassland_natural_conserve"
## [21] "SparseVeg_natural_conserve"           "Wetlands_natural_conserve"
## [23] "RiversLakes_natural_conserve"         "MarineTransitional_natural_conserve"
## [25] "WoodlandForest_primary_conserve"      "Urban_urban_lockin"
```

```
colnames(solution) <- c("id", (zone_id$zone))
```

```
solution_table <- pu_in_EU |> rename(id = pu) |> #plot_data |>
  left_join(PU_template) |>
  dplyr::select(-id) |>
  rename(id = EU_id) |>
  left_join(solution) |>
  pivot_longer(-c(nuts2id:geometry))
```

```
## Joining with ‘by = join_by(id)‘
## Joining with ‘by = join_by(id)‘
```

```
solution_table_plot <- solution_table |>
  mutate(zone = name) |>
  separate(name, c('maes_label', 'intensity', 'action'), sep = "_") |>
  mutate(maes_label = ifelse(maes_label %in% c("RiversLakes", "MarineTransitional"), "RiversLakesMarine
  mutate(zone = paste0(maes_label,"_", intensity,"_", action)) |>
  group_by(id, maes_label, intensity, action) |> mutate(value = value) |> ungroup() |>
  unique()
```

Quick bar plot of zones

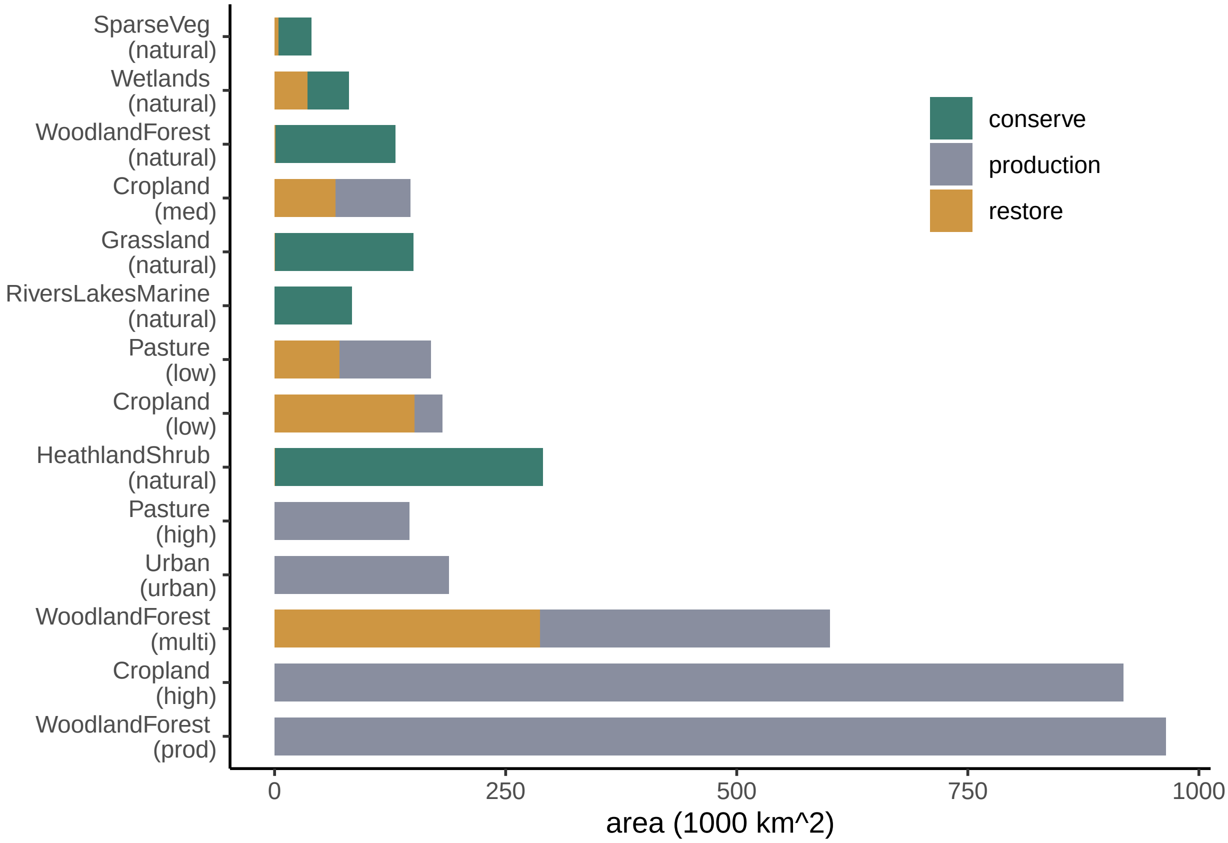
```
as_tibble(solution_table_plot) |>
  group_by(action) |>
  summarise(area = sum(value)/41046)
```

```
## # A tibble: 4 x 2
##   action       area
##   <chr>       <dbl>
## 1 conserve   0.179
## 2 lockin     0.0460
## 3 production 0.622
## 4 restore    0.150
```

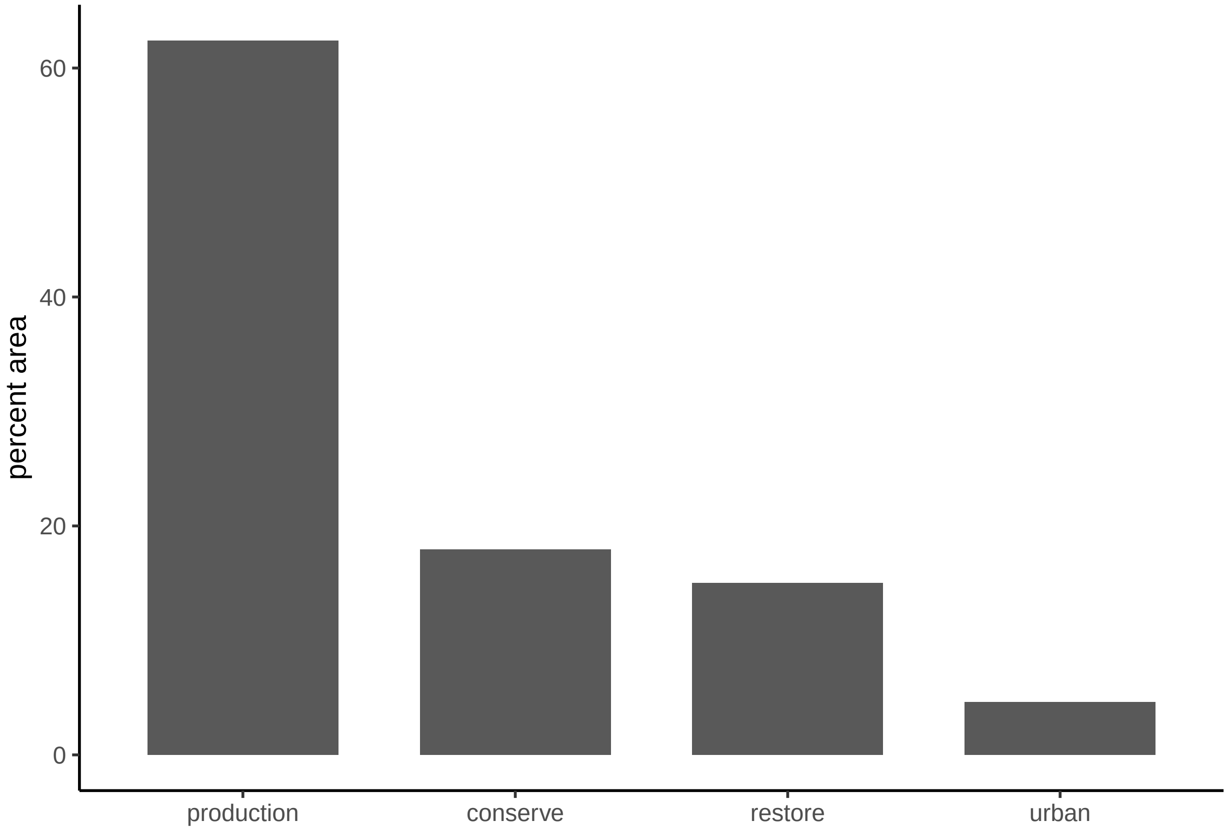
```
bar_plot <- as_tibble(solution_table_plot) |>
  group_by(zone, maes_label, intensity, action) |>
  summarise(area = sum(value)) |>
  mutate(action = ifelse(action == "lockin", "production", action)) |>
  ungroup() |> filter(area >0) |>
  mutate(intensity = ifelse(intensity == "primary", "natural", intensity)) |>
  mutate(name = paste0(maes_label, " \n (", intensity, ")")) |>
  ggplot(aes(x = reorder(name, -area), y = area/10, fill = action)) + geom_bar(stat="identity", width =
  theme_classic() +
  labs(x = element_blank(), y = "area (1000 km^2)") +
  theme(legend.position = c(0.8,0.8),
        legend.title = element_blank()) +
    scale_fill_manual(values = c(met.brewer(name="Kandinsky",n=4,type="continuous"))[c(1,3,2)]) + coord
```

```
## ‘summarise()‘ has grouped output by ’zone’, ’maes_label’, ’intensity’. You can
## override using the ‘.groups‘ argument.
```

```
bar_plot
```



```
bar_plot <- as_tibble(solution_table_plot) |>
  group_by(action) |>
  summarise(area = sum(value)) |>
  mutate(action = ifelse(action == "lockin", "urban", action)) |>
  ungroup() |> filter(area >0) |> mutate(tot_area = sum(area)) |>
  mutate(perc = area/tot_area*100) |>
  #mutate(intensity = ifelse(intensity == "primary", "natural", intensity)) |>
  #mutate(name = paste0(maes_label, " \n (", intensity, ")")) |>
  ggplot(aes(x = reorder(action, -perc), y = perc)) + geom_bar(stat="identity", width = 0.7) +
  theme_classic() +
  labs(x = element_blank(), y = "percent area") +
  theme(legend.position = c(0.8,0.8),
        legend.title = element_blank())

bar_plot
```



Maps of solution

```
solution_raster <- fasterize(st_as_sf(solution_table_plot), PU_plot, field = "value", by = "zone")
solution_raster <- rast(solution_raster)
names(solution_raster)
```

```
##  [1] "Cropland_low_restore"                 "Cropland_med_restore"
##  [3] "Grassland_natural_restore"            "HeathlandShrub_natural_restore"
##  [5] "RiversLakesMarine_natural_restore"    "Pasture_low_restore"
##  [7] "SparseVeg_natural_restore"            "Wetlands_natural_restore"
##  [9] "WoodlandForest_multi_restore"         "WoodlandForest_primary_restore"
## [11] "WoodlandForest_prod_restore"          "WoodlandForest_multi_production"
## [13] "WoodlandForest_prod_production"       "Pasture_high_production"
## [15] "Pasture_low_production"               "Cropland_med_production"
## [17] "Cropland_low_production"              "Cropland_high_production"
## [19] "HeathlandShrub_natural_conserve"      "Grassland_natural_conserve"
## [21] "SparseVeg_natural_conserve"           "Wetlands_natural_conserve"
## [23] "RiversLakesMarine_natural_conserve" "WoodlandForest_primary_conserve"
## [25] "Urban_urban_lockin"
```

```
restore <- c(1:10)
produce <- c(12:18,25)
conserve <- c(19:24)
```

Color stuff

```r
colors <- c("grey", met.brewer(name="VanGogh3",n=20,type="continuous"))
myPal <- colors
myTheme <- rasterTheme(region = myPal)
```

```r
restore <- solution_raster[[restore]]
conserve <- solution_raster[[conserve]]
produce <- solution_raster[[produce]]
```

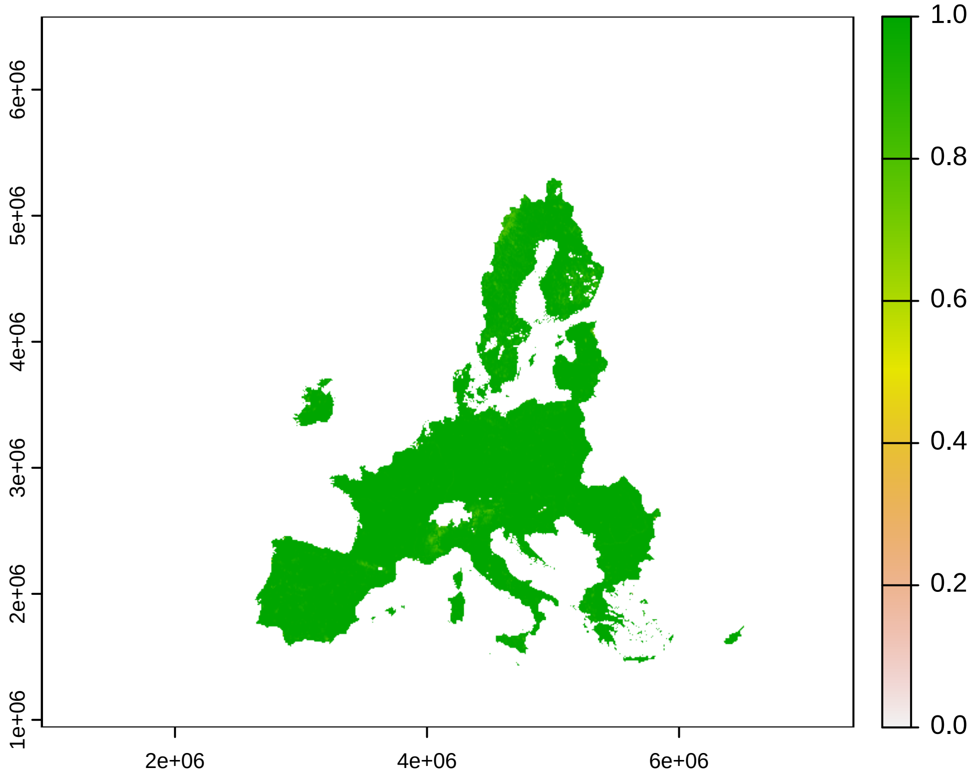
Just a check to make sure the solutions are summing to 1…

```r
solu_sum <- app(rast(solution_raster), 'sum')
plot(solu_sum)
```

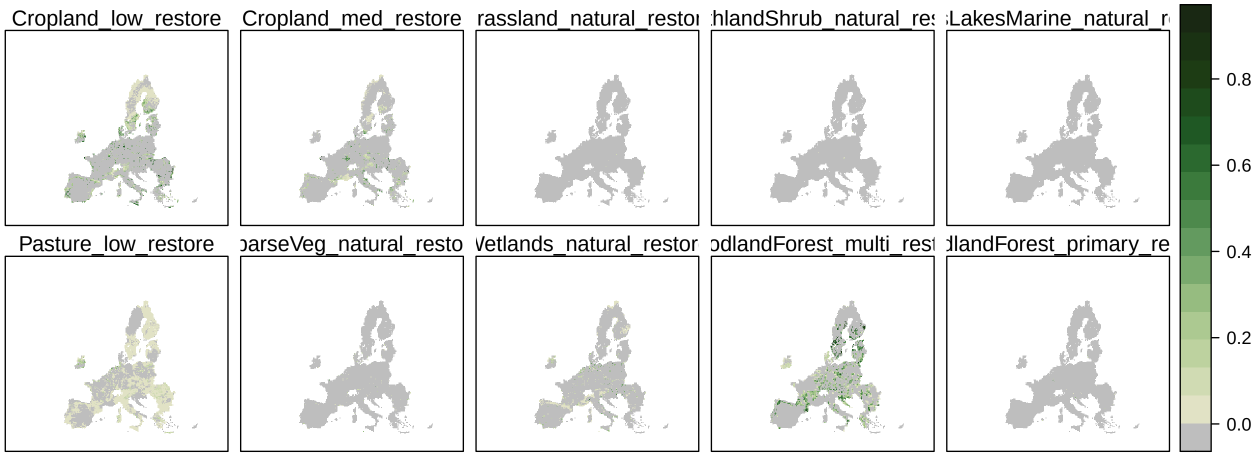
Check out restoration zones and overall restoration

```r
library(colorspace)
library(scico)
cols.v <- c("grey",scico(20, direction = -1,palette = "batlow"))
#colors <- c("grey", met.brewer(name="Isfahan1",n=20,type="continuous"))

colors <- c("grey", met.brewer(name="VanGogh3",n=20,type="continuous"))
myPal <- colors
myTheme <- rasterTheme(region = myPal)

p1 <- levelplot(restore, par.settings = myTheme,xlab=NULL, ylab=NULL, scales=list(draw=FALSE))
```

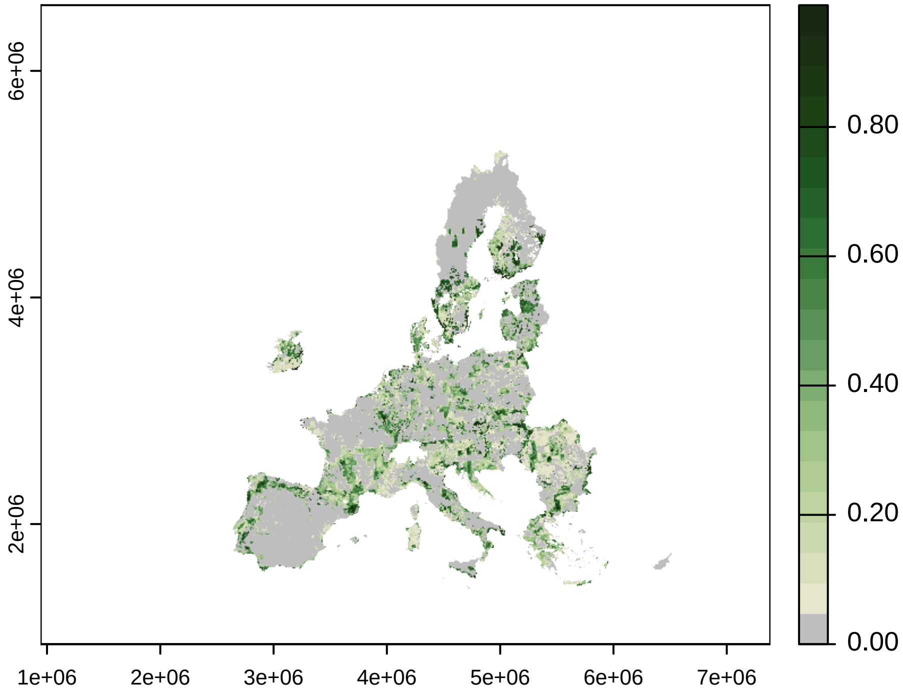
```
p2 <- levelplot(conserve, par.settings = myTheme,xlab=NULL, ylab=NULL, scales=list(draw=FALSE))

p3 <- levelplot(produce, par.settings = myTheme,xlab=NULL, ylab=NULL, scales=list(draw=FALSE))

solu_sum <- app(solution_raster, 'sum')
plot(solu_sum)
```



```
p1
```

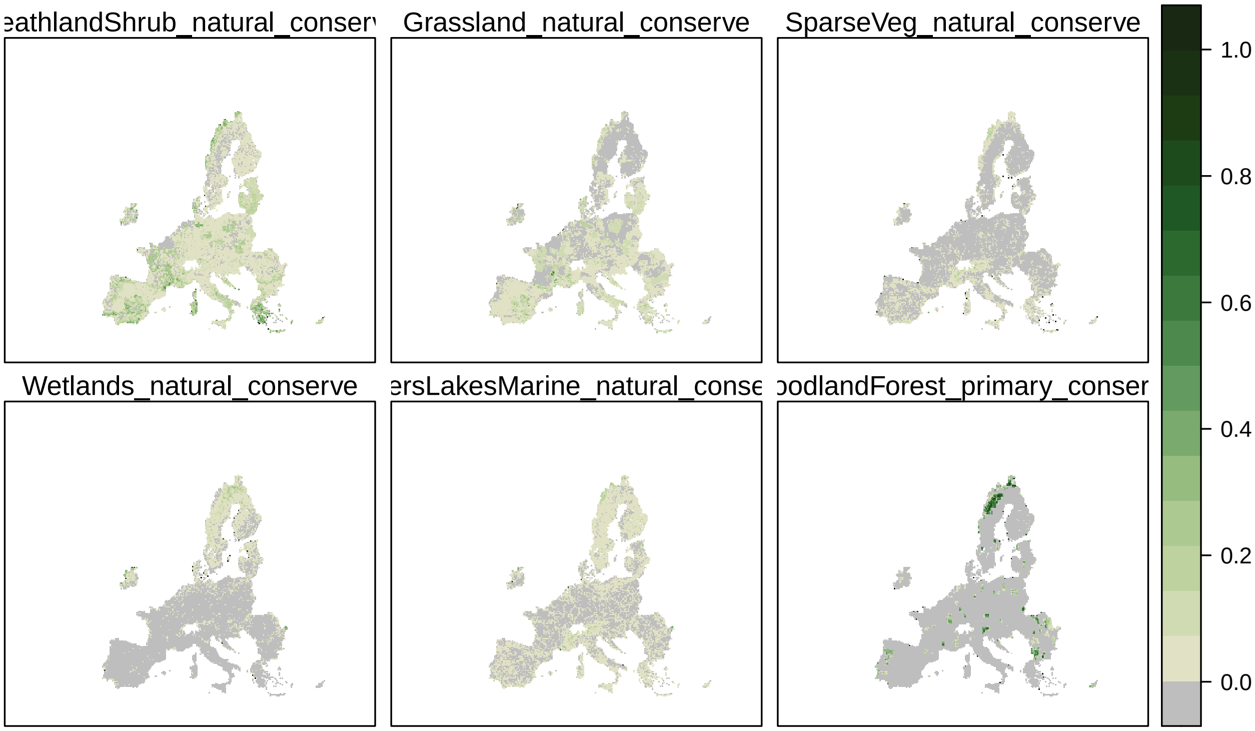


```
rest_agg <- app(restore, sum)
plot(rest_agg, col = colors)
```

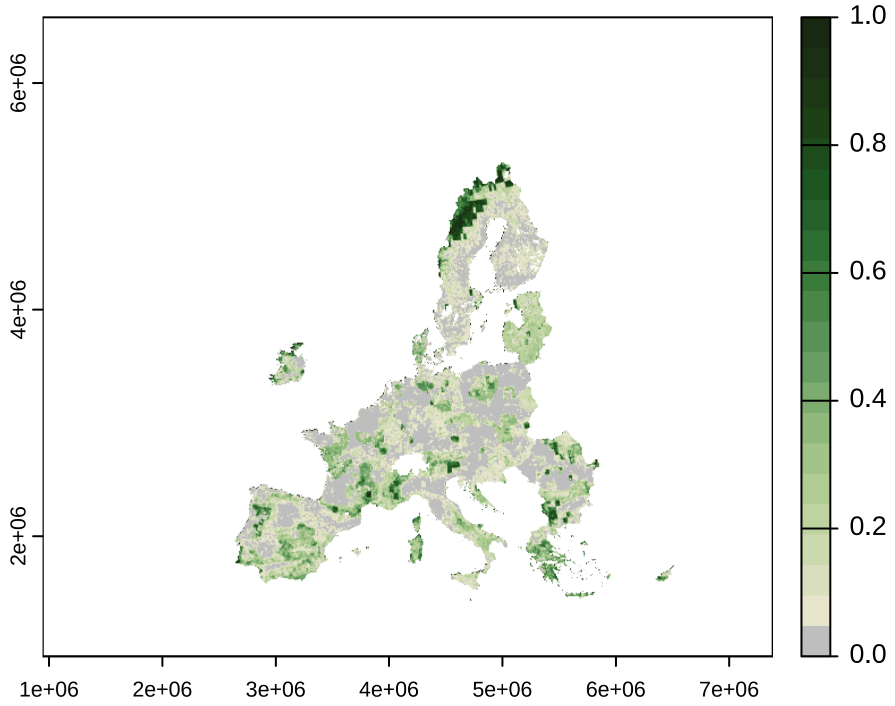


Check out conservation zones and overall conservation

```
levelplot(conserve, par.settings = myTheme,xlab=NULL, ylab=NULL, scales=list(draw=FALSE))
```

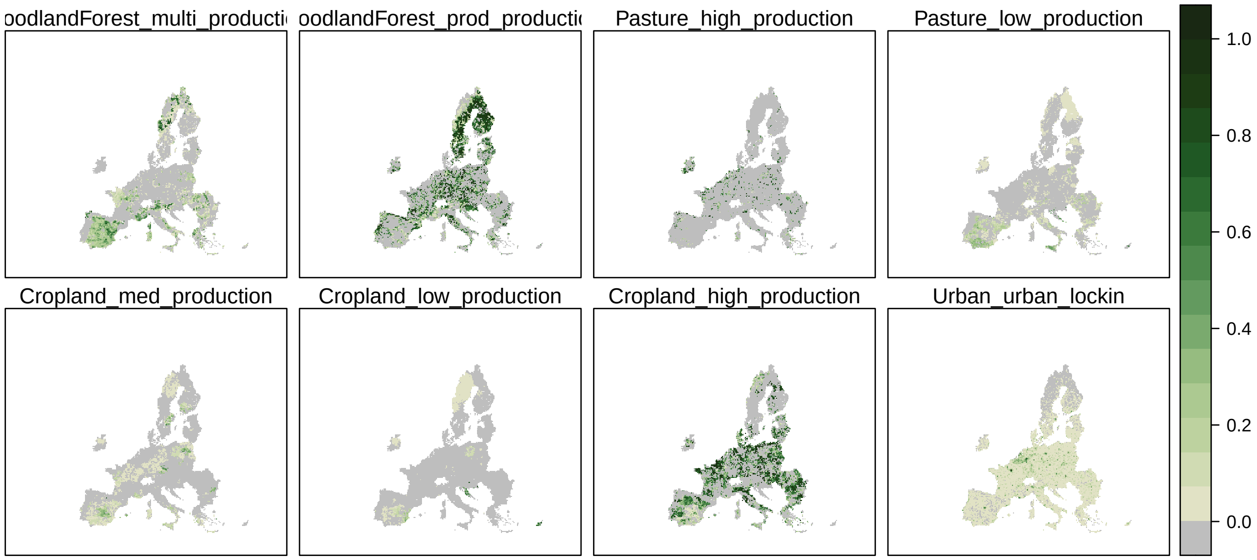

eathlandShrub_natural_conser
Grassland_natural_conserve
SparseVeg_natural_conserve
Wetlands_natural_conserve
ersLakesMarine_natural_conse
oodlandForest_primary_conser
1.0
0.8
0.6
0.4
0.2
0.0

```
conserve_agg <- app(conserve, sum)
plot(conserve_agg, col = colors)
```

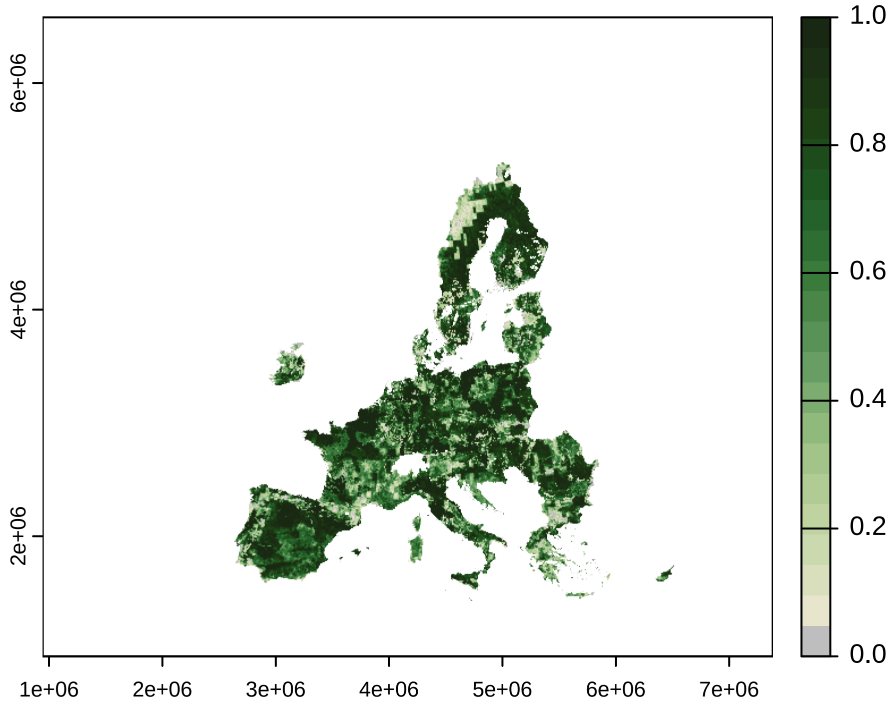


Check out production zones and overall production

```
levelplot(produce, par.settings = myTheme,xlab=NULL, ylab=NULL, scales=list(draw=FALSE))
```



```
prod_agg <- app(produce, sum)
plot(prod_agg, col = colors)
```



reshuffling

```
solution_rs <- solution |>
  pivot_longer(-id) |>
  rename(value_sol = value) |>
  separate(name, c('maes_label', 'intensity', 'action'), sep = "_") |>
  mutate(maes_label = ifelse(maes_label %in% c("RiversLakes", "MarineTransitional"), "RiversLakesMarine
  #mutate(zone = paste0(maes_label,"_", intensity,"_", action)) |>
  mutate(name = paste0(maes_label,"_", intensity))

solution_rs <- solution_rs |>
  group_by(id, name) |>
  summarise(value_sol = sum(value_sol, na.rm = T)) |>
  ungroup()
```

```
## `summarise()` has grouped output by 'id'. You can override using the `.groups`
## argument.
```

```
unique(solution_rs$name)
```

```
##  [1] "Cropland_high"              "Cropland_low"
##  [3] "Cropland_med"               "Grassland_natural"
##  [5] "HeathlandShrub_natural"     "Pasture_high"
##  [7] "Pasture_low"                "RiversLakesMarine_natural"
##  [9] "SparseVeg_natural"          "Urban_urban"
## [11] "Wetlands_natural"           "WoodlandForest_multi"
## [13] "WoodlandForest_primary"     "WoodlandForest_prod"
```

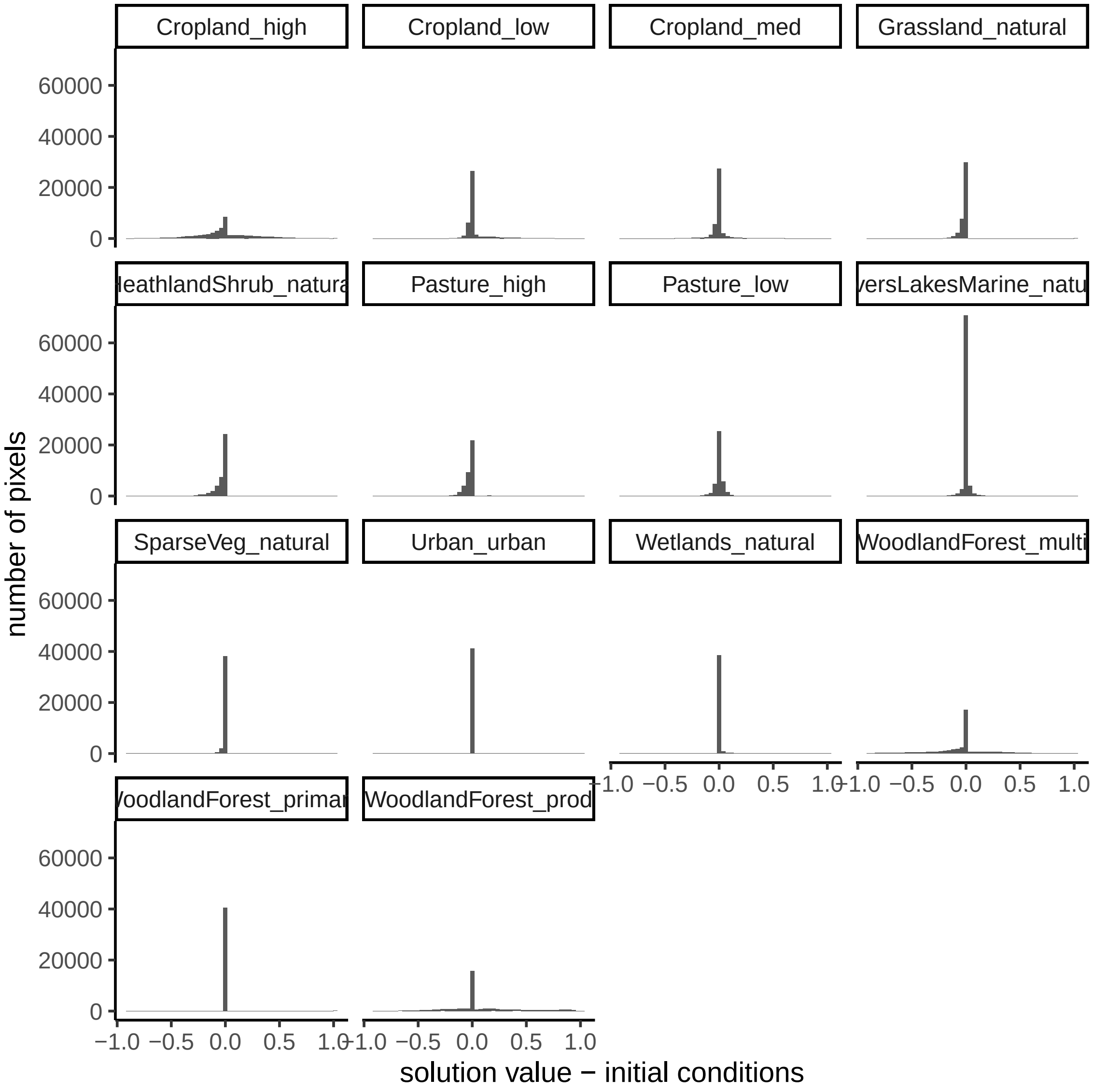
```
ic_rs <- ic |> dplyr::select(-Status) |>
  pivot_longer(-id) |>
  rename(value_ic = value) |>
  mutate(name = ifelse(name %in% c("MarineTransitional_natural", "RiversLakes_natural"), "RiversLakesMa
  full_join(solution_rs) |> mutate(value_ic = replace_na(value_ic,0),
                value_sol= replace_na(value_sol, 0),
                diff = value_sol- value_ic) |> dplyr::select(-c(value_sol,value_ic))
```

```
## Joining with ‘by = join_by(id, name)‘
```

```
ic_rs_wide <- ic_rs |> pivot_wider(names_from = name, values_from = diff)
```

```
## Warning: Values from ‘diff‘ are not uniquely identified; output will contain list-cols.
## * Use ‘values_fn = list‘ to suppress this warning.
## * Use ‘values_fn = {summary_fun}‘ to summarise duplicates.
## * Use the following dplyr code to identify duplicates.
##   {data} %>%
##   dplyr::group_by(id, name) %>%
##   dplyr::summarise(n = dplyr::n(), .groups = "drop") %>%
##   dplyr::filter(n > 1L)
```

```
ic_rs |>
  ggplot(aes(x = diff)) + geom_histogram(bins=50) + facet_wrap(~name) +
  labs(y = "number of pixels", x = "solution value - initial conditions") +
  theme_classic()
```


Cropland_high
Cropland_low
Cropland_med
Grassland_natural
HeathlandShrub_natura
Pasture_high
Pasture_low
versLakesMarine_natu
SparseVeg_natural
Urban_urban
Wetlands_natural
WoodlandForest_multi
oodlandForest_primar
WoodlandForest_prod
60000
40000
20000
0
−1.0
−0.5
0.0
0.5
1.0
number of pixels
solution value − initial conditions