Contents

0.1 Calculate overall OHI Index Scores

Congratulations if you've finished all goal model modifications as they are the most time-consuming part of the Index calculation process! In the goal model modification step, you've calculated goal status and trend, there are only a couple of more steps to do to calculate the overall index score, including calculating pressures and resilience.

0.1.1 Additional data layers for pressures and resilience calculations

So far you would have prepared the pressure and resilience matrices, as well as the appropriate data layers, a few more data layers are needed to complete the calculation. They are identified in `sub-country/conf/config.R` as follows:

```
# components describe the layer and level with which to aggregate resilience and pressures matrices for
resilience_components = list('NP'  = c('layer'='np_harvest_product_weight' , 'level'='region_id-category
                             'CS'  = c('layer'='cs_habitat_extent'         , 'level'='region_id'),
                             'CP'  = c('layer'='cp_habitat_extent_rank'    , 'level'='region_id'),
                             'HAB' = c('layer'='hab_presence'              , 'level'='region_id'))
pressures_components  = list('NP'  = c('layer'='np_harvest_product_weight' , 'level'='region_id-category
                             'CS'  = c('layer'='cs_habitat_extent'         , 'level'='region_id'),
                             'CP'  = c('layer'='cp_habitat_extent_rank'    , 'level'='region_id'),
                             'LIV' = c('layer'='le_sector_weight'          , 'level'='region_id'),
                             'ECO' = c('layer'='le_sector_weight'          , 'level'='region_id'),
                             'HAB' = c('layer'='hab_presence'              , 'level'='region_id'))
```

`np_harvst_product_weight` is also used in NP and CS status calculations, and thus do not require special preparations. The rest of the data layers need additional preparations, which can be done in the `prep` folder).

`cs_habitat_extent` is calculated as `habitat_extent * rank`. Rank refers to relative contributions of each type of habitats to carbon storage. Here is an example calculation:

```
# cs_habitat_extent = Habitat extent * rank, per Carbon Storage habitat
#                   = extent * contribution

extent = layers$data[['cs_extent']] %>%
  select(rgn_id, habitat, extent = hectare)

contribution = layers$data[['cs_contribution']] %>%
  select(rgn_id, habitat, contribution = value)

result = full_join(extent, contribution, by = c('rgn_id', 'habitat')) %>%
  mutate(hectare = extent*contribution) %>%
  select(rgn_id, habitat, hectare)
```

cp_habitat_extent_rank is calculated as habitat_extent * weight. Weight refers to relative contributions of each type of habitats to coastal protection. See this example:

```
# cp_habitat_extent_rank = Habitat extent * rank, per Coastal Protection habitat
#                        = extent * weight

habitat.wt = c('saltmarshes' = 3,
               'mangroves' = 4,
               'seagrasses' = 1,
               'coral reef' = 4)

m = layers$data[['cp_extent']] %>%
  group_by(rgn_id, habitat) %>%
  filter(year == max(year)) %>% #choose the most recent year's data
  select(-layer,
         -year,
         extent = hectare) %>%
  mutate(weight = habitat.wt[habitat],
         extent_rank = extent * weight) %>%
  select(rgn_id, habitat, extent_rank)
```

hab_presence is calculated on a binary basis. All regions with a habitat is assigned "1". Example:

```
# hab_presence: 1 for presence

m = layers$data[['hab_extent']] %>%
  group_by(rgn_id, habitat) %>%
  filter(year == max(year)) %>%  #choose the most recent year's data
  select(-layer,
         -year,
         extent = hectare) %>%
  mutate(boolean = 1) %>%
  select(rgn_id, habitat, boolean)
```

le_sector_weight assigns relative importance of each sector listed in Livelihood & Economies goal. By default, all sectors are considered equal and assigned a weight of "1". One thing to pay special attention to is the sector names, which should match component names that were listed in the pressures matrix.

0.1.2 Final calculation

Go back to calculate_scores.r, now you are ready for the final calculations:

```
# calculate scenario scores
scores = CalculateAll(conf, layers, debug=F)

# save scores as .csv file, tables and figures
write.csv(scores, 'scores.csv', na='', row.names=F)
```

After the calculation is done, you should be able to see the compiled score sheet for all goals in all regions in sub-country/scores.csv.

It is very likely that during the CalculateAll process you'll encounter problems and see error messages. In most cases, the error messages can specify what the error is and in which step it occurs, which should be helpful for trouble shooting. Some commonly occurring errors and how to fix them can be found in the Troubleshooting section of the manual.