Harvest module in Treeforce package

Harvest of Uneven stands

Monospecific case

Harvest rate

We note H the global harvest rate which will determine the amount of basal area to be harvested. This rate is defined according to the following rule

$$H = \left\{ \begin{array}{ll} 0 & if(BA_{stand} - BA_{target}) < \Delta BA_{min} \\ min(\frac{BA_{stand} - BA_{target}}{BA_{stand}}, H_{max}) & if(BA_{stand} - BA_{target}) \geq \Delta BA_{min} \end{array} \right\}$$

For example, $\Delta BA_{min} = 3m^2ha^{-1}$, $BA_{target} = 20, 25$ or $30m^2ha^{-1}$ (depending on species) and $H_{max} = 0.25$.

Note that stand basal area BA_{stand} is computed only considering trees with a dbh above d_{th} (see below).

Harvest Curve

Each tree harvest probability only depends on its diameter (d). There is a minimum diameter of harvest (d_{th}), harvest probability then increases with diameter until d_{ha}, the harvesting diameter after which harvest probability is high and constant.

We therefore considered the harvesting function (which associates a dbh to an harvesting probability)

$$h(d) = \left\{ \begin{array}{ll} 0 & if d < d_{th} \\ h_{max}(\frac{d - d_{th}}{d_{ha} - d_{th}})^k & if d_{th} \leq d < d_{ha} \\ h_{max} & if d > d_{ha} \end{array} \right\}$$

The parameter h_{max} can be tuned so that the probability for a large tree to be harvested approaches 1 after several harvesting operations: $h_{max} = 1 - \sqrt[n]{1-p}$

with n the number of harvesting operations.

For example, $d_{th} = 17.5cm$, $d_{ha} = 57.5cm$, $p = 0.99$, $n = 2$.

Parameter k defines how much the harvest preferentially selects large trees.

```r
library(latex2exp)
library(ggplot2)
dth <- 17.5
dha <- 57.5
hmax <- 0.8
k <- 2
```

Harvesting algorithm

The basal area harvested is

$$\begin{aligned} BA_{harv} &= \pi/4 \int_{d_{th}}^{d_{max}} x^2 h(x)\phi(x)dx \\ &= \pi/4 \int_{d_{th}}^{d_{ha}} x^2 h(x)\phi(x)dx + h_{max}\pi/4 \int_{d_{ha}}^{d_{max}} x^2 \phi(x)dx \\ &= BA_{th} + BA_{ha} \end{aligned}$$

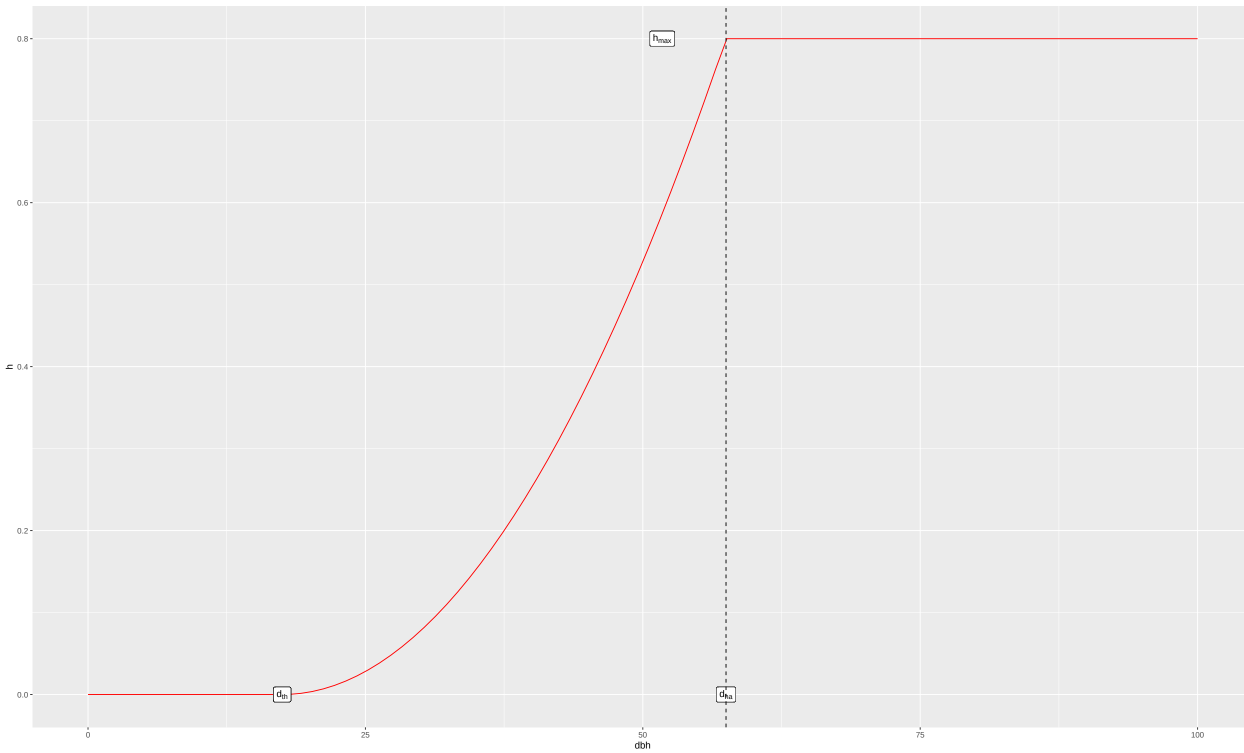

Figure 1: Example, $d_{th} = 17.5cm$, $d_{ha} = 57.5cm$, $h_{max} = 0.8$, $k = 2$.

with $\phi(x)$ the density function of diameters.

If $BA_{ha} >= H * BA_{stand}$, there is enough large trees (diameter above d_{ha}) so that the harvest will only concern large trees: $BA_{th} = 0$.

We then have two different strategies:

First one is: we adapt h_{max} so that:

$$h_{max}\pi/4 \int_{d_{ha}}^{d_{max}} x^2\phi(x)dx = H * BA_{stand}$$

Second one is: we adapt d_{ha} so that:

$$h_{max}\pi/4 \int_{d_{ha}}^{d_{max}} x^2\phi(x)dx = H * BA_{stand}$$

If $BA_{ha} < H * BA_{stand}$, we first harvest BA_{ha} and then compute k such as: $BA_{th} = H * BA_{stand} - BA_{ha}$.

Multispecific case

Abundance-based preference

As in the monospecific case, we define the global harvest rate $H = \frac{BA_{harv}}{BA_{stand}}$.

Here, BA_{stand} is divided between s species: $BA_{stand} = \sum_{i=1}^{s} BA_{stand,i}$ $BA_{harv} = \sum_{i=1}^{s} BA_{harv,i}$

We note $p_i = \frac{BA_{stand,i}}{BA_{stand}}$ and $H_i = \frac{BA_{stand,i} - BA_{harv,i}}{BA_{stand,i}} = f(p_i) * H$

We suppose that harvesting rate increases with abundance (we harvest preferentially trees with the highest proportion), which means f is an increasing function.

By definition,

$$\begin{aligned} BA_{harv} &= \sum_{i=1}^{s} BA_{harv,i} = \sum_{i=1}^{s} BA_{stand,i} * (1 - H_i) \\ &= BA_{stand} \sum_{i=1}^{s} p_i * (1 - f(P_i)H) \\ &= BA_{stand}(1 - H) \end{aligned}$$

So that we have the constraint on f:

$\sum_{i=1}^{s} p_i(1 - f(p_i)H) = 1 - H \sum_{i=1}^{s} p_i f(p_i) = 1 - H$ which is equivalent to

$\sum_{i=1}^{s} p_i f(p_i) = 1$

The case $f(p_i) = 1$ works, which leads to $H_i = H$. In that case the harvest rate is the same for every species i.

More broadly,

$$\forall \alpha > 0$$

$$f(p_i) = \frac{p_i^{\alpha-1}}{\sum_{i=1}^{s} p_i^{\alpha}}$$

For $\alpha = 2$, we for example have

$$f(p_i) = \frac{p_i}{\sum_{i=1}^{s} p_i^2}$$

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## v purrr   0.3.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x readr::edition_get()   masks testthat::edition_get()
## x dplyr::filter()        masks stats::filter()
## x purrr::is_null()       masks testthat::is_null()
## x dplyr::lag()           masks stats::lag()
## x readr::local_edition() masks testthat::local_edition()
## x dplyr::matches()       masks tidyr::matches(), testthat::matches()
```

```
grid <- expand.grid(p_1 = seq(0.1, 0.9, by = 0.1),
                    p_2 = seq(0.1, 0.9, by = 0.1),
                    alpha = c(0.5, seq(1, 10, by = 2)))

fp_i <- function(x, y, alpha){
    (x ^ (alpha - 1))/(sum(y ^ alpha) )
}

grid <- mutate(grid, sum = p_1 + p_2) %>% filter(sum == 1)
f_1 =  apply(grid, 1, function(x) fp_i(x[1], x[1:2], x[3]) - x[2])
f_2 =  apply(grid, 1, function(x) fp_i(x[2], x[1:2], x[3]) - x[1])
grid <- cbind(grid, f_1, f_2)

ggplot(grid, aes(p_1, p_2)) +
    facet_grid(. ~ alpha) +
    geom_segment(aes(xend = f_1, yend = f_2)) +
```

```
    geom_point(x = f_1, y = f_2, shape = 16) +
    NULL
```