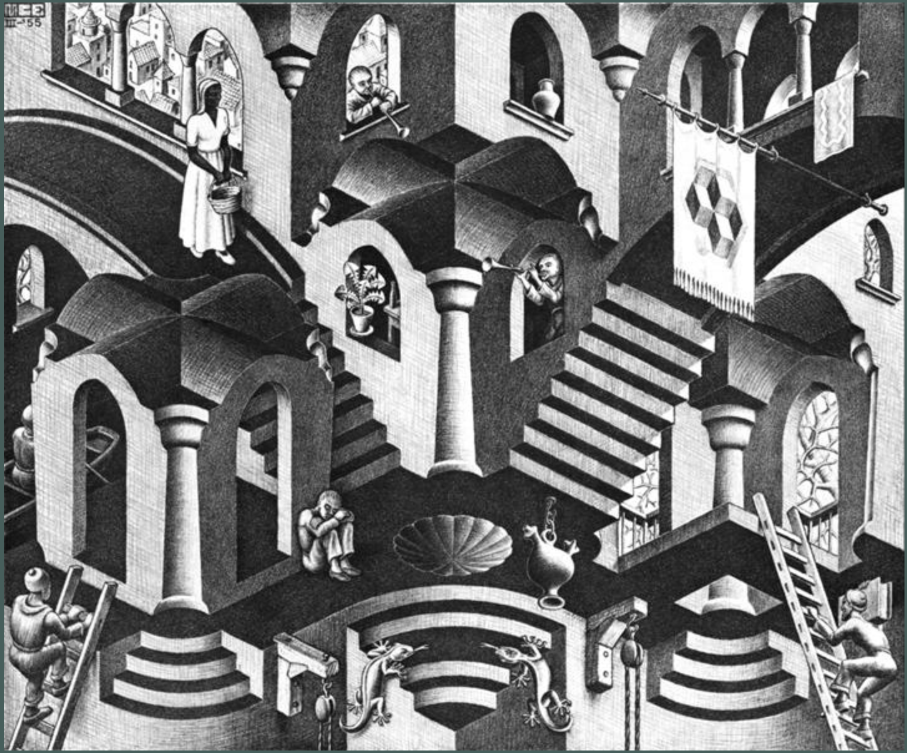
3.3 - Forest growth/dynamics (exercise)

Miquel De Cáceres, Victor Granda, Aitor Ameztegui

Ecosystem Modelling Facility

2022-06-15



Exercise setting

Objectives

1. Learn to perform simulations of forest growth and forest dynamics with medfate
2. Evaluate tree growth predictions with tree ring data
3. Compare simulated vs observed forest changes between inventories
4. Project forest dynamics with/without forest management

Exercise setting

Objectives

1. Learn to perform simulations of forest growth and forest dynamics with medfate
2. Evaluate tree growth predictions with tree ring data
3. Compare simulated vs observed forest changes between inventories
4. Project forest dynamics with/without forest management

Data

We will use data corresponding to a forest plot of sampled during the third and fourth Spanish National Forest Inventory (SNFI3) in the province of Tarragona (latitude 41° N aprox.).

Exercise setting

Objectives

1. Learn to perform simulations of forest growth and forest dynamics with medfate
2. Evaluate tree growth predictions with tree ring data
3. Compare simulated vs observed forest changes between inventories
4. Project forest dynamics with/without forest management

Data

We will use data corresponding to a forest plot of sampled during the third and fourth Spanish National Forest Inventory (SNFI3) in the province of Tarragona (latitude 41° N aprox.).

- The forest plot is dominated by Aleppo pine (*Pinus halepensis*) with an understory of composed of several shrub species.

Exercise setting

Objectives

1. Learn to perform simulations of forest growth and forest dynamics with medfate
2. Evaluate tree growth predictions with tree ring data
3. Compare simulated vs observed forest changes between inventories
4. Project forest dynamics with/without forest management

Data

We will use data corresponding to a forest plot of sampled during the third and fourth Spanish National Forest Inventory (SNFI3) in the province of Tarragona (latitude 41° N aprox.).

- The forest plot is dominated by Aleppo pine (*Pinus halepensis*) with an understory of composed of several shrub species.

- Tree ring data are available for some trees of the forest plot, because it was included in a research project focused on intraspecific variability of functional traits (FUN2FUN, granted to J. Martínez-Vilalta).

Exercise setting

Objectives

1. Learn to perform simulations of forest growth and forest dynamics with medfate
2. Evaluate tree growth predictions with tree ring data
3. Compare simulated vs observed forest changes between inventories
4. Project forest dynamics with/without forest management

Data

We will use data corresponding to a forest plot of sampled during the third and fourth Spanish National Forest Inventory (SNFI3) in the province of Tarragona (latitude 41° N aprox.).

- The forest plot is dominated by Aleppo pine (*Pinus halepensis*) with an understory of composed of several shrub species.

- Tree ring data are available for some trees of the forest plot, because it was included in a research project focused on intraspecific variability of functional traits (FUN2FUN, granted to J. Martínez-Vilalta).

- Soil has been already drawn from *SoilGrids*

Exercise setting

Objectives

1. Learn to perform simulations of forest growth and forest dynamics with medfate
2. Evaluate tree growth predictions with tree ring data
3. Compare simulated vs observed forest changes between inventories
4. Project forest dynamics with/without forest management

Data

We will use data corresponding to a forest plot of sampled during the third and fourth Spanish National Forest Inventory (SNFI3) in the province of Tarragona (latitude 41º N aprox.).

- The forest plot is dominated by Aleppo pine (*Pinus halepensis*) with an understory of composed of several shrub species.

- Tree ring data are available for some trees of the forest plot, because it was included in a research project focused on intraspecific variability of functional traits (FUN2FUN, granted to J. Martínez-Vilalta).

- Soil has been already drawn from *SoilGrids*

- Daily weather data corresponding to the plot location has been obtained with **meteoland**, corresponding to an historical period (SNFI3-SNFI4) and a future period (2015-2100) under scenario RCP 8.5 (from Earth system model MPI-ESM regionalized to Europe using model RCA4).

Exercise solution

Step 1. Load Alepo pine forest data

We are given all the necessary data, bundled in a single list:

```r
alepo <- readRDS("StudentRdata/alepo.rds")
```

Exercise solution

Step 1. Load Alepo pine forest data

We are given all the necessary data, bundled in a single list:

```
alepo <- readRDS("StudentRdata/alepo.rds")
```

Whose elements are…

Element	Description
`forest_snfi3`	Object of class `forest` with the stand structure and composition in SNFI3 (yr. 2001)
`forest_snfi4`	Object of class `forest` with the stand structure and composition in SNFI4 (yr. 2014)
`spt`	Object of class `SpatialPointsTopography` with the coordinates and topography of the plot
`soildesc`	Data frame with soil properties.
`historic_weather`	Data frame with daily weather for years 2001-2014.
`projected_weather`	Data frame with daily weather for years 2015-2100 under RCP8.5 (climate model couple MPIESM/RCA4).
`observed_growth`	Data frame with **annual** basal area increments during the 2001-2014 period for four *P. halepensis* trees in the forest plot (`T20_148`, `T14_148`, `T25_148` and `T3_148`).
`snfi34_growth`	Data frame with density, diameter and height for *P. halepensis* as measured in SNFI3 and SNFI3.

Exercise solution

Step 2. Forest stand metrics

We can use the `summary()` function for objects of class `forest` to know the leaf area index and basal area estimated at yr. 2001 (SNFI3):

```
summary(alepo$forest_snfi3, SpParamsMED)
```

```
## Tree density (ind/ha): 721.50240945
## Tree BA (m2/ha): 21.5278871
## Cover (%) trees (open ground): 100  shrubs: 100
## Shrub crown phytovolume (m3/m2): 1.04
## LAI (m2/m2) total: 3.6639431  trees: 1.4241149  shrubs: 2.2398282
## Live fine fuel (kg/m2) total: 1.5337579  trees: 0.5444354  shrubs: 0.9893226
## PAR ground (%): 14.5677246  SWR ground (%): 24.0043619
```

Exercise solution

Step 2. Forest stand metrics

We can use the `summary()` function for objects of class `forest` to know the leaf area index and basal area estimated at yr. 2001 (SNFI3):

```
summary(alepo$forest_snfi3, SpParamsMED)
```

```
## Tree density (ind/ha): 721.50240945
## Tree BA (m2/ha): 21.5278871
## Cover (%) trees (open ground): 100  shrubs: 100
## Shrub crown phytovolume (m3/m2): 1.04
## LAI (m2/m2) total: 3.6639431  trees: 1.4241149  shrubs: 2.2398282
## Live fine fuel (kg/m2) total: 1.5337579  trees: 0.5444354  shrubs: 0.9893226
## PAR ground (%): 14.5677246  SWR ground (%): 24.0043619
```

The contribution of the different species to these stand metrics can be known using:

```
species_basalArea(alepo$forest_snfi3, SpParamsMED)
```

```
##   Pinus halepensis  Quercus coccifera Pistacia lentiscus  Salvia rosmarinus   Erica multiflora
##           21.52789            0.00000            0.00000            0.00000            0.00000
```

```
species_LAI(alepo$forest_snfi3, SpParamsMED)
```

```
##   Pinus halepensis  Quercus coccifera Pistacia lentiscus  Salvia rosmarinus   Erica multiflora
##          1.4241149          0.2774996          0.3928101          1.3935065          0.1760121
```

Exercise solution

Step 2. Forest stand metrics

We repeat the same calculations for yr. 2014 (SNFI4):

```
summary(alepo$forest_snfi4, SpParamsMED)
```

```
## Tree density (ind/ha): 707.35530341
## Tree BA (m2/ha): 27.5720378
## Cover (%) trees (open ground): 100  shrubs: 100
## Shrub crown phytovolume (m3/m2): 1.133
## LAI (m2/m2) total: 4.6079012  trees: 1.5995943  shrubs: 3.0083069
## Live fine fuel (kg/m2) total: 1.6496117  trees: 0.6115207  shrubs: 1.038091
## PAR ground (%): 8.6749798  SWR ground (%): 16.3505438
```

There has been an increase of 6 m2/ha in basal area, whereas stand LAI has increased 0.94 m2/m2.

Exercise solution

Step 3. Growth simulation between SNFI3 and SNFI4

We were given soil physical characteristics, but we need to build an object of class `soil`, which we can store in the same `alepo` list:

```
alepo$soil <- soil(alepo$soildesc)
```

Exercise solution

Step 3. Growth simulation between SNFI3 and SNFI4

We were given soil physical characteristics, but we need to build an object of class `soil`, which we can store in the same `alepo` list:

```
alepo$soil <- soil(alepo$soildesc)
```

we can check the water holding capacity of the soil using:

```
sum(soil_waterFC(alepo$soil))
```

```
## [1] 391.1652
```

which is rather high but we leave it as is.

Exercise solution

Step 3. Growth simulation between SNFI3 and SNFI4

We were given soil physical characteristics, but we need to build an object of class `soil`, which we can store in the same `alepo` list:

```
alepo$soil <- soil(alepo$soildesc)
```

we can check the water holding capacity of the soil using:

```
sum(soil_waterFC(alepo$soil))
```

```
## [1] 391.1652
```

which is rather high but we leave it as is.

We now have all the elements to call function `forest2growthInput()` to generate the input for `growth()`:

```
x_alepo <- forest2growthInput(x = alepo$forest_snfi3,
                              soil = alepo$soil,
                              SpParams = SpParamsMED,
                              control = defaultControl())
```

Exercise solution

Step 3. Growth simulation between SNFI3 and SNFI4

Since the list contains also the historic weather for years 2001-2014 and topography, we are ready to simulate growth:

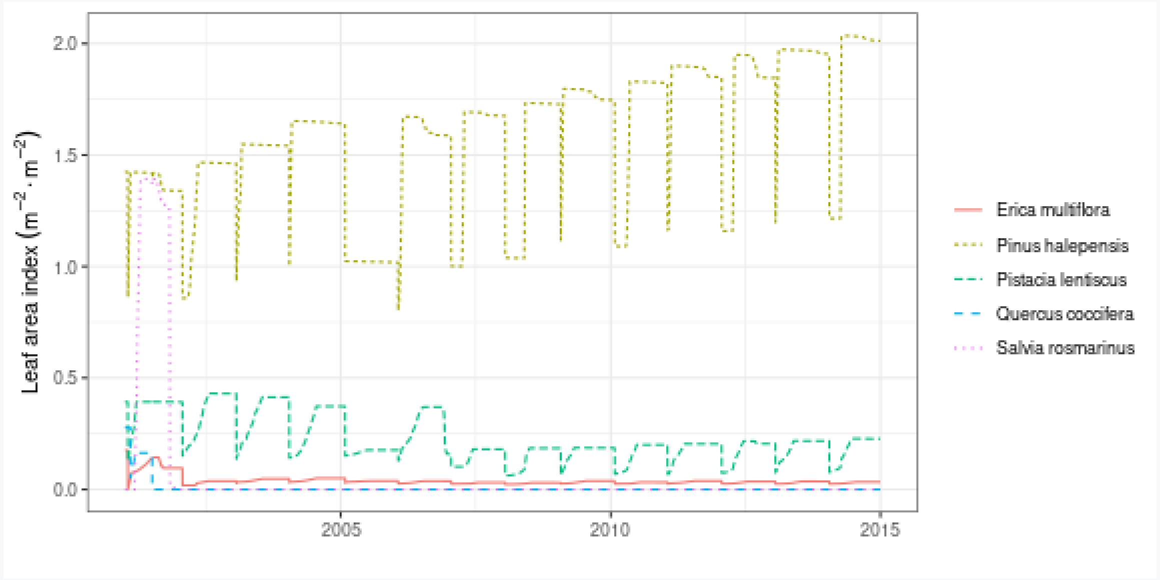
```r
G_34 <- growth(x = x_alepo,
               meteo = alepo$historic_weather,
               latitude = 41,
               elevation = alepo$spt$elevation,
               slope = alepo$spt$slope,
               aspect = alepo$spt$aspect)
```

Exercise solution

Step 4. Examine growth results

Many outputs can be inspected using `shinyplot()` but here we use `plot()` to display the LAI dynamics of the different species

```
plot(G_34, "PlantLAI", bySpecies = TRUE)
```



The model predicts an increase in LAI for *P. halepensis* (except some years), but shrub species are predicted to lose leaf area.

Exercise solution

Step 5. Evaluate tree basal area increment

We can use function `evaluation_plot()` to display the predicted and observed BAI for the four trees with measurements:

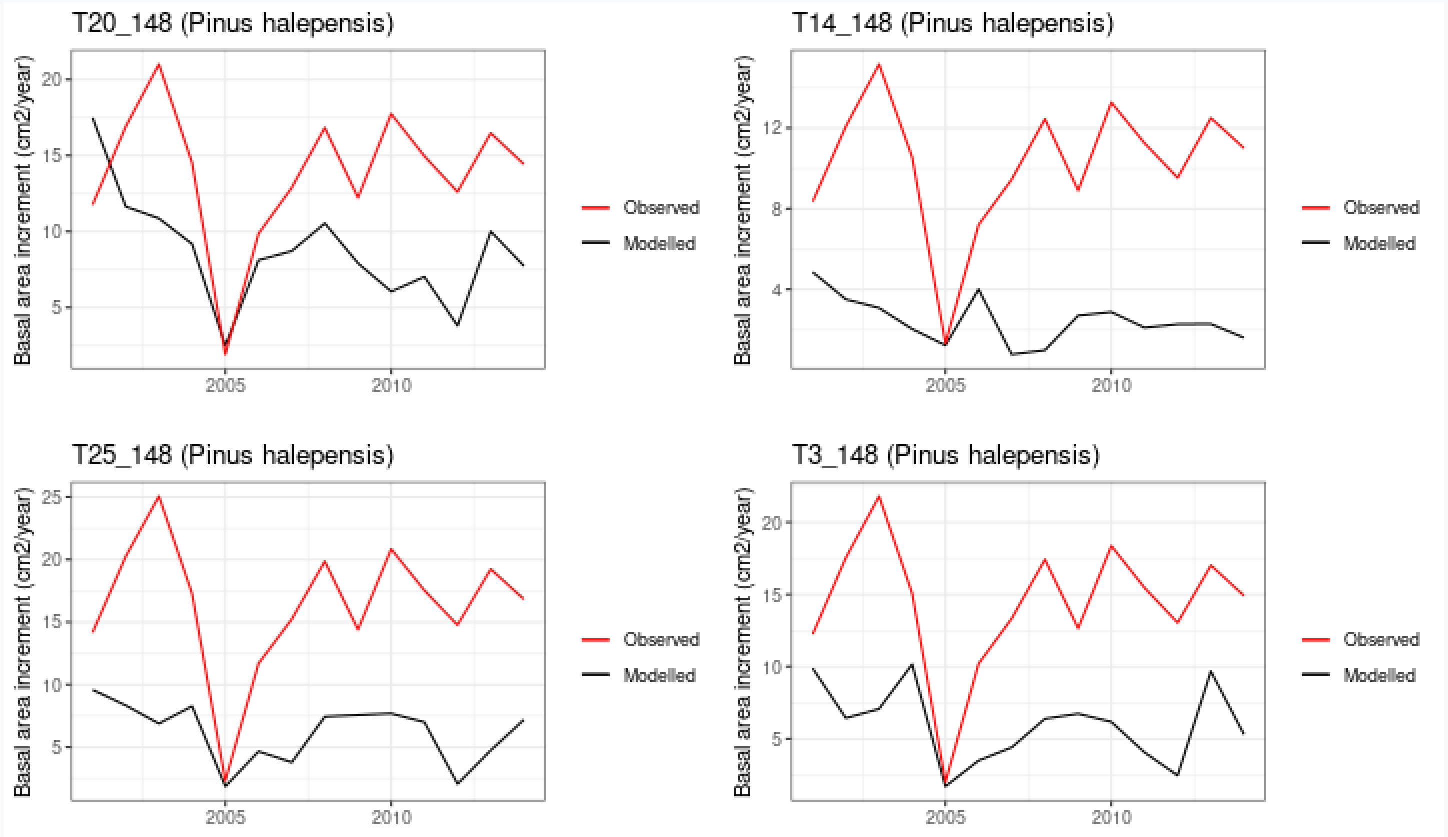
```
g1<-evaluation_plot(G_34, alepo$observed_growth, type="BAI",
                    cohort = "T20_148", temporalResolution = "year")
g2<-evaluation_plot(G_34, alepo$observed_growth, type="BAI",
                 cohort = "T14_148", temporalResolution = "year")
g3<-evaluation_plot(G_34, alepo$observed_growth, type="BAI",
                 cohort = "T25_148", temporalResolution = "year")
g4<-evaluation_plot(G_34, alepo$observed_growth, type="BAI",
                 cohort = "T3_148", temporalResolution = "year")
```

Exercise solution

Step 5. Evaluate tree basal area increment

When we display the plots we see that the model is overestimating growth in many cases:

```
plot_grid(g1,g2,g3,g4, ncol = 2, nrow=2)
```



Exercise solution

Step 5. Evaluate tree basal area increment

Tip: To decide how to proceed when a model fails to fit observations is important to know which model parameters may be responsible for a given result (this is called *sensitivity analysis*).

Exercise solution

Step 5. Evaluate tree basal area increment

Tip: To decide how to proceed when a model fails to fit observations is important to know which model parameters may be responsible for a given result (this is called *sensitivity analysis*).

In medfate, sapwood (and tree) growth is strongly controlled by parameter `RGRcambiummax`, which specifies the maximum growth rate of sapwood relative to stem diameter.

Exercise solution

Step 5. Evaluate tree basal area increment

Tip: To decide how to proceed when a model fails to fit observations is important to know which model parameters may be responsible for a given result (this is called *sensitivity analysis*).

In medfate, sapwood (and tree) growth is strongly controlled by parameter `RGRcambiummax`, which specifies the maximum growth rate of sapwood relative to stem diameter.

For *P. halepensis* its default value is:

```
SpParamsMED$RGRcambiummax[SpParamsMED$Name=="Pinus halepensis"]
```

```
## [1] 0.003688597
```

Exercise solution

Step 6. Modify maximum growth rate for P. halepensis and repeat simulations

We divide the maximum relative growth rate by two…

```
SpParamsMED$RGRcambiummax[SpParamsMED$Name=="Pinus halepensis"] <- 0.0012
```

Exercise solution

Step 6. Modify maximum growth rate for P. halepensis and repeat simulations

We divide the maximum relative growth rate by two…

```
SpParamsMED$RGRcambiummax[SpParamsMED$Name=="Pinus halepensis"] <- 0.0012
```

… rebuild the growth input …

```
x_alepo <- forest2growthInput(x = alepo$forest_snfi3,
                              soil = alepo$soil,
                              SpParams = SpParamsMED,
                              control = defaultControl())
```

Exercise solution

Step 6. Modify maximum growth rate for P. halepensis and repeat simulations

We divide the maximum relative growth rate by two…

```r
SpParamsMED$RGRcambiummax[SpParamsMED$Name=="Pinus halepensis"] <- 0.0012
```

… rebuild the growth input …

```r
x_alepo <- forest2growthInput(x = alepo$forest_snfi3,
                              soil = alepo$soil,
                              SpParams = SpParamsMED,
                              control = defaultControl())
```

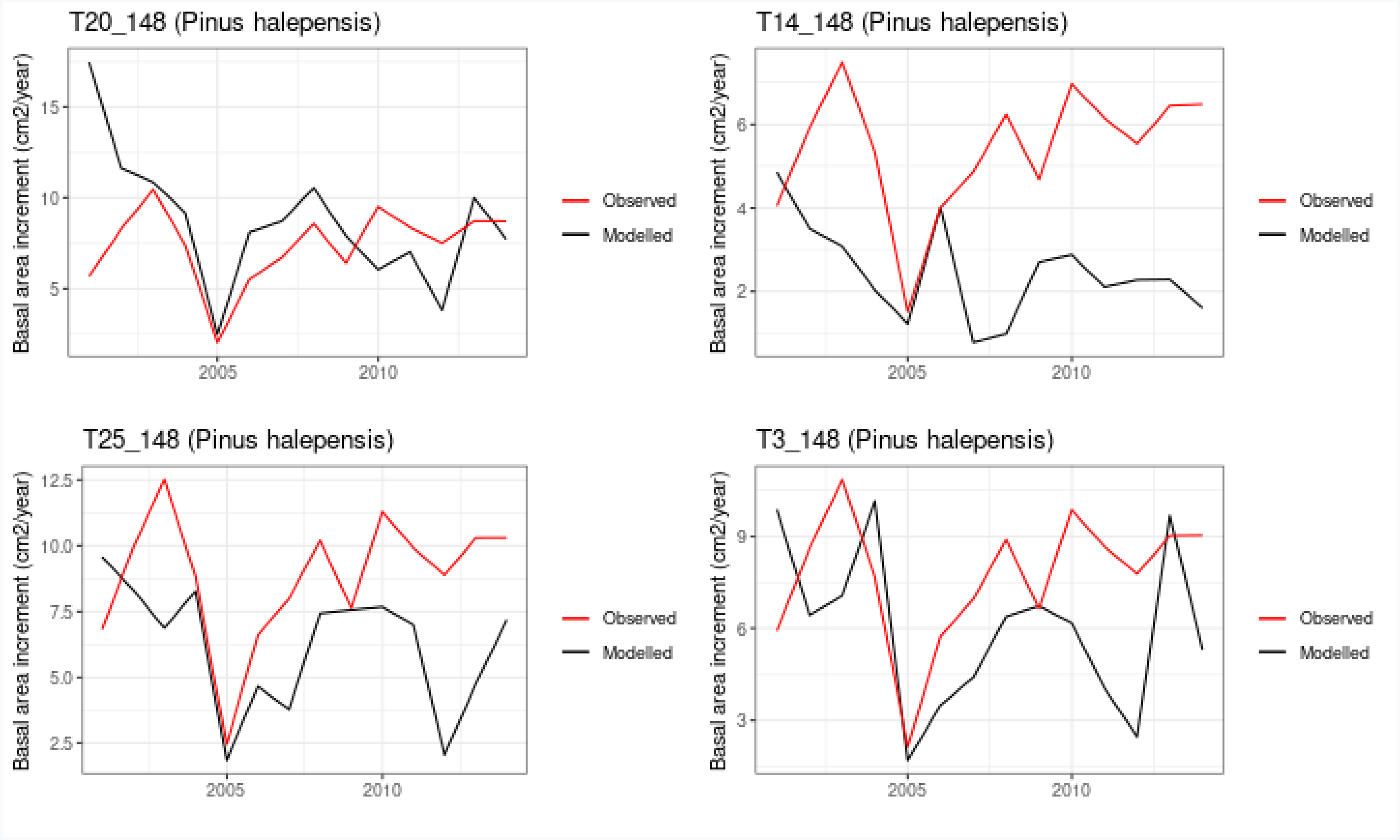
… and launch a new simulation:

```r
G_34m <- growth(x = x_alepo,
               meteo = alepo$historic_weather,
               latitude = 41,
               elevation = alepo$spt$elevation,
               slope = alepo$spt$slope,
               aspect = alepo$spt$aspect)
```

Exercise solution

Step 6. Modify maximum growth rate for P. halepensis and repeat simulations

We can inspect the fit of the new results to observed data. Overall, we obtain a better fit in terms of the mean BAI, but the model does not capture all observed interannual variation.



Exercise solution

Step 7. Reduce the number of tree cohorts

In order to speed-up forest dynamic simulations, we can reduce the number of tree cohorts, which is now:

```
nrow(alepo$forest_snfi3$treeData)
```

```
## [1] 28
```

Exercise solution

Step 7. Reduce the number of tree cohorts

In order to speed-up forest dynamic simulations, we can reduce the number of tree cohorts, which is now:

```
nrow(alepo$forest_snfi3$treeData)
```

```
## [1] 28
```

Remembering the `forest_mergeTrees()` function from exercise #1:

```
forest_red = forest_mergeTrees(alepo$forest_snfi3)
```

Exercise solution

Step 7. Reduce the number of tree cohorts

In order to speed-up forest dynamic simulations, we can reduce the number of tree cohorts, which is now:

```
nrow(alepo$forest_snfi3$treeData)
```

```
## [1] 28
```

Remembering the `forest_mergeTrees()` function from exercise #1:

```
forest_red = forest_mergeTrees(alepo$forest_snfi3)
```

The new forest object has 5 tree cohorts:

```
forest_red$treeData
```

```
##   Species         N      DBH    Height      Z50  Z95
## 1     148  14.14711 31.60000 1400.0000 522.4242 4000
## 2     148 198.05948 25.38220 1100.2943 522.4242 4000
## 3     148 159.15494 20.49330  936.3455 522.4242 4000
## 4     148 222.81692 14.62423  809.2011 522.4242 4000
## 5     148 127.32395 11.85000  820.0000 522.4242 4000
```

Exercise solution

Step 7. Reduce the number of tree cohorts

In order to speed-up forest dynamic simulations, we can reduce the number of tree cohorts, which is now:

```
nrow(alepo$forest_snfi3$treeData)
```

```
## [1] 28
```

Remembering the `forest_mergeTrees()` function from exercise #1:

```
forest_red = forest_mergeTrees(alepo$forest_snfi3)
```

The new forest object has 5 tree cohorts:

```
forest_red$treeData
```

```
##   Species         N      DBH    Height      Z50  Z95
## 1     148  14.14711 31.60000 1400.0000 522.4242 4000
## 2     148 198.05948 25.38220 1100.2943 522.4242 4000
## 3     148 159.15494 20.49330  936.3455 522.4242 4000
## 4     148 222.81692 14.62423  809.2011 522.4242 4000
## 5     148 127.32395 11.85000  820.0000 522.4242 4000
```

In the following, we will use `forest_red` to call function `fordyn()`.

Exercise solution

Step 8. Run forest dynamics simulation

Remember: unlike `spwb()` and `growth()`, we do not need to build an intermediate input object for `fordyn()` (i.e., there is no function `forest2fordynInput()`).

Exercise solution

Step 8. Run forest dynamics simulation

Remember: unlike `spwb()` and `growth()`, we do not need to build an intermediate input object for `fordyn()` (i.e., there is no function `forest2fordynInput()`).

In our call to `fordyn()`, we supply the historic weather (yrs. 2001-2014), as we did in our call to `growth()`, because we want to compare predicted changes with those observed between SNFI3 and SNFI4.

```
FD_34 <- fordyn(forest = forest_red,
                soil = alepo$soil,
                SpParams = SpParamsMED,
                control = defaultControl(),
                meteo = alepo$historic_weather,
                latitude = 41,
                elevation = alepo$spt$elevation,
                slope = alepo$spt$slope,
                aspect = alepo$spt$aspect)
```

Exercise solution

Step 8. Run forest dynamics simulation

Remember: unlike `spwb()` and `growth()`, we do not need to build an intermediate input object for `fordyn()` (i.e., there is no function `forest2fordynInput()`).

In our call to `fordyn()`, we supply the historic weather (yrs. 2001-2014), as we did in our call to `growth()`, because we want to compare predicted changes with those observed between SNFI3 and SNFI4.

```
FD_34 <- fordyn(forest = forest_red,
              soil = alepo$soil,
              SpParams = SpParamsMED,
              control = defaultControl(),
              meteo = alepo$historic_weather,
              latitude = 41,
              elevation = alepo$spt$elevation,
              slope = alepo$spt$slope,
              aspect = alepo$spt$aspect)
```

The elements of the output have the following names, which we should be able to understand before moving on (if not, see `?fordyn`).

```
names(FD_34)
```

```
##  [1] "StandSummary"    "SpeciesSummary"  "CohortSummary"   "TreeTable"       "DeadTreeTable"
##  [6] "CutTreeTable"    "ShrubTable"      "DeadShrubTable"  "CutShrubTable"   "ForestStructures"
## [11] "GrowthResults"   "ManagementArgs"  "NextInputObject" "NextForestObject"
```

Exercise solution

Step 9. Compare final stand metrics with the observed stand in SNFI4

In particular, we can examine the stand metrics of the `forest` object at the end of the simulation…

```
summary(FD_34$NextForestObject, SpParamsMED)
```

```
## Tree density (ind/ha): 919.946082936554
## Tree BA (m2/ha): 26.2644408
## Cover (%) trees (open ground): 100  shrubs: 40.3431789
## Shrub crown phytovolume (m3/m2): 0.2695373
## LAI (m2/m2) total: 2.3960396  trees: 1.6132901  shrubs: 0.7827495
## Live fine fuel (kg/m2) total: 1.0456124  trees: 0.6167566  shrubs: 0.4288558
## PAR ground (%): 29.365862  SWR ground (%): 40.3468326
```

Exercise solution

Step 9. Compare final stand metrics with the observed stand in SNFI4

In particular, we can examine the stand metrics of the `forest` object at the end of the simulation…

```
summary(FD_34$NextForestObject, SpParamsMED)
```

```
## Tree density (ind/ha): 919.946082936554
## Tree BA (m2/ha): 26.2644408
## Cover (%) trees (open ground): 100  shrubs: 40.3431789
## Shrub crown phytovolume (m3/m2): 0.2695373
## LAI (m2/m2) total: 2.3960396  trees: 1.6132901  shrubs: 0.7827495
## Live fine fuel (kg/m2) total: 1.0456124  trees: 0.6167566  shrubs: 0.4288558
## PAR ground (%): 29.365862  SWR ground (%): 40.3468326
```

… and compare them to those obtained in SNFI4 (yr. 2015) for the forest plot:

```
summary(alepo$forest_snfi4, SpParamsMED)
```

```
## Tree density (ind/ha): 707.35530341
## Tree BA (m2/ha): 27.5720378
## Cover (%) trees (open ground): 100  shrubs: 100
## Shrub crown phytovolume (m3/m2): 1.133
## LAI (m2/m2) total: 4.6079012  trees: 1.5995943  shrubs: 3.0083069
## Live fine fuel (kg/m2) total: 1.6496117  trees: 0.6115207  shrubs: 1.038091
## PAR ground (%): 8.6749798  SWR ground (%): 16.3505438
```

Exercise solution

Step 9. Compare final stand metrics with the observed stand in SNFI4

In particular, we can examine the stand metrics of the `forest` object at the end of the simulation…

```
summary(FD_34$NextForestObject, SpParamsMED)
```

```
## Tree density (ind/ha): 919.946082936554
## Tree BA (m2/ha): 26.2644408
## Cover (%) trees (open ground): 100  shrubs: 40.3431789
## Shrub crown phytovolume (m3/m2): 0.2695373
## LAI (m2/m2) total: 2.3960396  trees: 1.6132901  shrubs: 0.7827495
## Live fine fuel (kg/m2) total: 1.0456124  trees: 0.6167566  shrubs: 0.4288558
## PAR ground (%): 29.365862  SWR ground (%): 40.3468326
```

… and compare them to those obtained in SNFI4 (yr. 2015) for the forest plot:

```
summary(alepo$forest_snfi4, SpParamsMED)
```

```
## Tree density (ind/ha): 707.35530341
## Tree BA (m2/ha): 27.5720378
## Cover (%) trees (open ground): 100  shrubs: 100
## Shrub crown phytovolume (m3/m2): 1.133
## LAI (m2/m2) total: 4.6079012  trees: 1.5995943  shrubs: 3.0083069
## Live fine fuel (kg/m2) total: 1.6496117  trees: 0.6115207  shrubs: 1.038091
## PAR ground (%): 8.6749798  SWR ground (%): 16.3505438
```

The model seems to perform fairly well in terms of final tree density and basal area. However, as expected, it yields too much shrub mortality, resulting in a forest with a low understory biomass.

Exercise solution

Step 10. Projection of forest dynamics

Argument `forest` of function `fordyn()` can be used to supply the final state of a previous simulation.

Exercise solution

Step 10. Projection of forest dynamics

Argument `forest` of function `fordyn()` can be used to supply the final state of a previous simulation.

Hence, we can use this feature to start our projection from the final state of the previous call to `fordyn()` and use the projected daily weather:

```
FD_proj <- fordyn(forest = FD_34,
                  soil = alepo$soil,
                  SpParams = SpParamsMED,
                  control = defaultControl(),
                  meteo = alepo$projected_weather,
                  latitude = 41,
                  elevation = alepo$spt$elevation,
                  slope = alepo$spt$slope,
                  aspect = alepo$spt$aspect)
```

Exercise solution

Step 10. Projection of forest dynamics

Argument `forest` of function `fordyn()` can be used to supply the final state of a previous simulation.

Hence, we can use this feature to start our projection from the final state of the previous call to `fordyn()` and use the projected daily weather:

```
FD_proj <- fordyn(forest = FD_34,
              soil = alepo$soil,
              SpParams = SpParamsMED,
              control = defaultControl(),
              meteo = alepo$projected_weather,
              latitude = 41,
              elevation = alepo$spt$elevation,
              slope = alepo$spt$slope,
              aspect = alepo$spt$aspect)
```

The predicted final stand basal area is:

```
stand_basalArea(FD_proj$NextForestObject)
```

```
## [1] 52.1008
```

Exercise solution

Step 11. Management function and management arguments

We will now simulate forest dynamics including forest management.

Exercise solution

Step 11. Management function and management arguments

We will now simulate forest dynamics including forest management.

However, we need first to understand how the default management function works and the meaning of its parameters:

```
man_args <- defaultManagementArguments()
names(man_args)
```

```
##  [1] "type"                   "targetTreeSpecies"     "thinning"
##  [4] "thinningMetric"         "thinningThreshold"     "thinningPerc"
##  [7] "minThinningInterval"    "yearsSinceThinning"    "finalMeanDBH"
## [10] "finalPerc"              "finalPreviousStage"    "finalYearsBetweenCuts"
## [13] "finalYearsToCut"        "plantingSpecies"       "plantingDBH"
## [16] "plantingHeight"         "plantingDensity"       "understoryMaximumCover"
```

Exercise solution

Step 11. Management function and management arguments

We will now simulate forest dynamics including forest management.

However, we need first to understand how the default management function works and the meaning of its parameters:

```
man_args <- defaultManagementArguments()
names(man_args)
```

```
##  [1] "type"                  "targetTreeSpecies"     "thinning"
##  [4] "thinningMetric"        "thinningThreshold"     "thinningPerc"
##  [7] "minThinningInterval"   "yearsSinceThinning"    "finalMeanDBH"
## [10] "finalPerc"             "finalPreviousStage"    "finalYearsBetweenCuts"
## [13] "finalYearsToCut"       "plantingSpecies"       "plantingDBH"
## [16] "plantingHeight"        "plantingDensity"       "understoryMaximumCover"
```

Argument `thinningThreshold` specifies the stand basal area value that leads to a thinning event. Since our simulation started at 26 m2/ha and increased up to 52 m2/ha, we set the value of `thinningThreshold` to 30 m2/ha to see some effects during the simulations:

```
man_args$thinningThreshold <- 30
```

Exercise solution

Step 12. Projection of forest dynamics with management

The call to `fordyn()` is similar to the previous one, except for the specification of the management function and parameters:

```
FD_proj_man <- fordyn(forest = FD_34,
                soil = alepo$soil,
                SpParams = SpParamsMED,
                control = defaultControl(),
                meteo = alepo$projected_weather,
                latitude = 41,
                elevation = alepo$spt$elevation,
                slope = alepo$spt$slope,
                aspect = alepo$spt$aspect,
                management_function = defaultManagementFunction,
                management_args = man_args^)
```

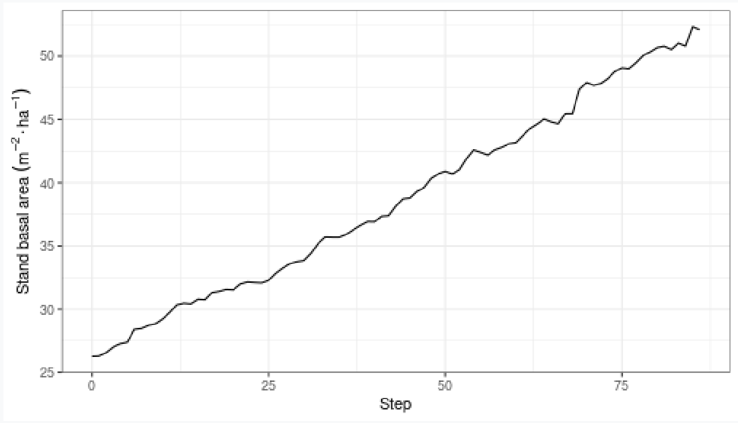
Exercise solution

Step 13. Compare forest dynamics with/without management

We can produce plots of stand basal area dynamics to compare the two simulations:

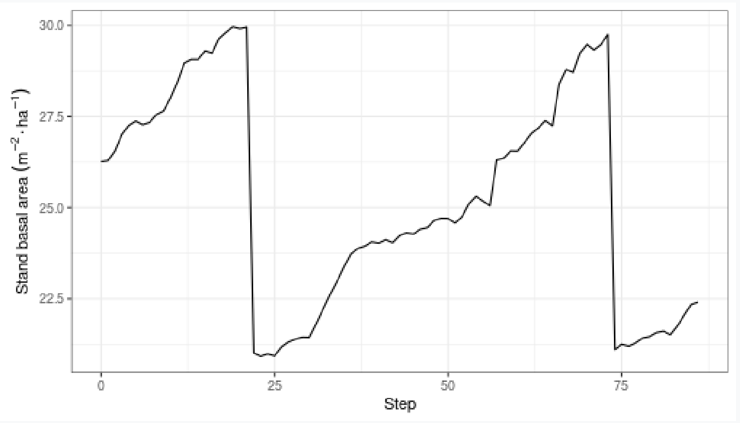
No management

```r
plot(FD_proj, "StandBasalArea")
```



Management

```r
plot(FD_proj_man, "StandBasalArea")
```



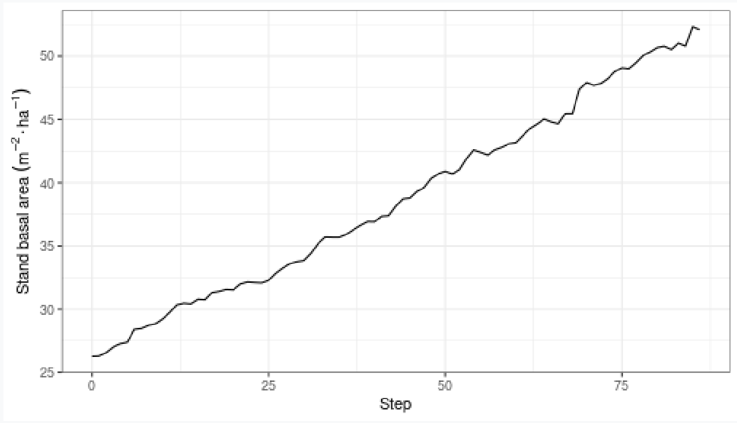
Exercise solution

Step 13. Compare forest dynamics with/without management

We can produce plots of stand basal area dynamics to compare the two simulations:

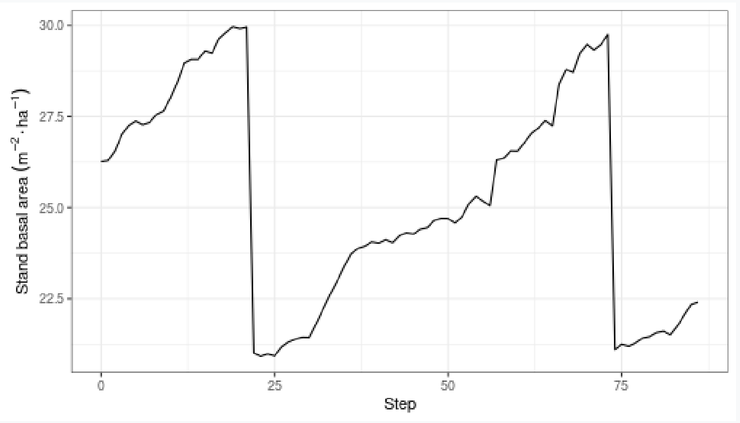
No management

```r
plot(FD_proj, "StandBasalArea")
```



Management

```r
plot(FD_proj_man, "StandBasalArea")
```



Generally speaking, the arguments `thinningThreshold` and `thinningPerc` control the frequency and intensity of thinning interventions.

Exercise solution

Step 13. Compare forest dynamics with/without management

We can also compare the final tree data frames of the `forest` objects of the two simulations:

No management

```
FD_proj$NextForestObject$treeData[,1:4]
```

```
##   Species       DBH    Height          N
## 1     148 46.242455 1649.2031   8.459126
## 2     148 40.009616 1520.4876 118.427760
## 3     148 35.124450 1463.5371  95.165164
## 4     148 29.258967 1425.4242 133.231230
## 5     148 26.490240 1430.9978  76.132131
## 6     148 21.241281 1338.1708 159.718591
## 7     148 19.273174 1281.3398 167.667172
## 8     148  9.760524  768.7657 218.076709
## 9     148  7.587567  577.5869 272.595454
```

Management

```
FD_proj_man$NextForestObject$treeData[,1:4]
```

```
##   Species      DBH   Height          N
## 1     148 47.14512 1657.693   8.459126
## 2     148 40.70770 1525.359 118.427760
## 3     148 36.02478 1469.454  54.184731
```

Exercise solution

Step 13. Compare forest dynamics with/without management

We can also compare the final tree data frames of the `forest` objects of the two simulations:

No management

```
FD_proj$NextForestObject$treeData[,1:4]
```

```
##   Species       DBH    Height          N
## 1     148 46.242455 1649.2031   8.459126
## 2     148 40.009616 1520.4876 118.427760
## 3     148 35.124450 1463.5371  95.165164
## 4     148 29.258967 1425.4242 133.231230
## 5     148 26.490240 1430.9978  76.132131
## 6     148 21.241281 1338.1708 159.718591
## 7     148 19.273174 1281.3398 167.667172
## 8     148  9.760524  768.7657 218.076709
## 9     148  7.587567  577.5869 272.595454
```

Management

```
FD_proj_man$NextForestObject$treeData[,1:4]
```

```
##   Species      DBH   Height          N
## 1     148 47.14512 1657.693   8.459126
## 2     148 40.70770 1525.359 118.427760
## 3     148 36.02478 1469.454  54.184731
```

The number of tree cohorts is much lower at the end of the simulation with forest management because by default the thinning is specified to be applied to small trees (i.e. `thinning = "below"`).

Exercise solution

Step 13. Compare forest dynamics with/without management

Finally, we can use the annual summaries produced by `fordyn()` to compare the basal area of trees dead or cut during the simulation:

No management

```
sum(FD_proj$StandSummary$BasalAreaDead)
```

```
## [1] 17.20003
```

```
sum(FD_proj$StandSummary$BasalAreaCut)
```

```
## [1] 0
```

Management

```
sum(FD_proj_man$StandSummary$BasalAreaDead)
```

```
## [1] 11.26269
```

```
sum(FD_proj_man$StandSummary$BasalAreaCut)
```

```
## [1] 18.05398
```

Exercise solution

Step 13. Compare forest dynamics with/without management

Finally, we can use the annual summaries produced by `fordyn()` to compare the basal area of trees dead or cut during the simulation:

No management

```
sum(FD_proj$StandSummary$BasalAreaDead)
```

```
## [1] 17.20003
```

```
sum(FD_proj$StandSummary$BasalAreaCut)
```

```
## [1] 0
```

Management

```
sum(FD_proj_man$StandSummary$BasalAreaDead)
```

```
## [1] 11.26269
```

```
sum(FD_proj_man$StandSummary$BasalAreaCut)
```

```
## [1] 18.05398
```

The simulation without forest management produced more dead trees than the simulation with management.

Exercise solution

Step 13. Compare forest dynamics with/without management

Finally, we can use the annual summaries produced by `fordyn()` to compare the basal area of trees dead or cut during the simulation:

No management

```
sum(FD_proj$StandSummary$BasalAreaDead)
```

```
## [1] 17.20003
```

```
sum(FD_proj$StandSummary$BasalAreaCut)
```

```
## [1] 0
```

Management

```
sum(FD_proj_man$StandSummary$BasalAreaDead)
```

```
## [1] 11.26269
```

```
sum(FD_proj_man$StandSummary$BasalAreaCut)
```

```
## [1] 18.05398
```

The simulation without forest management produced more dead trees than the simulation with management.

This arises because:

- Basal mortality rates are multiplied by the current tree density
- Drought stress is decreased in simulations with management

M.C. Escher - Concave and convex, 1955