Introduction to hisafer

hisafer is an R toolbox for the Hi-sAFe biophysical agroforestry model. It provides functions for defining, building, running, reading, analyzing, and plotting Hi-sAFe simulaitons. The first step to using hisafer is to load the library:

```
library(hisafer)
```

hisafer utilizes the tidyverse approach to R programming and data manipulation. While most consequneces of this approach are behind the scenes, one obvious example to users is that outputs from most hisafer fucntions are tibbles rather than simple data frames. This has little practical impact on your use of hisafer, but should improve your overall experience.

Six Steps to Hi-sAFe Experimentation

hisafer tools are organized via the six main steps of experimentation with Hi-sAFe: Define, Build, Run, Read, Diagnostics, Analysis. While hisafer can be used to interact with Hi-sAFe at any of these steps independently, enhanced functionality is available when using hisafer to interact with Hi-sAFe from the beginning.

1. Define

Define functions allow you to define one or more Hi-sAFe "simulations". Simulations are definied by specifying one or more Hi-SAFe input parameters from the .sim, .pld, or .tree files. Any Hi-SAFe input parameters not specified will simply inherit the parameters used during Hi-sAFe's calibration. To see which Hi-SAFe input parameters are currently supported, use:

```
hisafe_params()
```

```
## budBurstAccumulatedTemp
## budBurstTempAccumulationDateStart
## cellWidth
## coarseRootAnoxiaResistance
## colonisationThreshold
## interCropItk
## interCropSpecies
## latitude
## lueMax
## mainCropItk
## mainCropSpecies
## nbSimulations
## rootAnoxiaHalfLife
## rootHalfLife
## rootShape
## simulationDayStart
## SimulationName
## simulationNbrDays
## simulationYearStart
## slopeAspect
## slopeIntensity
## spacingBetweenRows
## spacingWithinRows
## toricSymmetry
```

```
## treeCropDistance
## treeHeight
## treeLineOrientation
## treePruningFreq
## treePruningMaxHeight
## treePruningProp
## treeRootPruningDepth
## treeRootPruningDistance
## treeRootPruningFreq
## treeSpecies
## waterTable
## weatherFile
## weededAreaRadius
## windMeanForce
```

By default, `hisafe_params()` just provides the names of supported parameters. To also see the default values, range of allowed values, and range of suggested values, use:

```
hisafe_params("all")
```

Once you know which parameter(s) you want to customize, you can define a Hi-sAFe simulation or experiment using `define_hisafe()`. Let's start with just a simple Hi-sAFe simulation in which we only want to customize **latitude** and **treeLineOrientation**.

```
hip_sim <- define_hisafe(latitude = 60, treeLineOrientation = -180)
```

```
## Error: Hi-sAFe definition errors:
## -- treeLineOrientation - must be betwen 0 and 359
```

Oops! It looks like **treeLineOrientation** must be between 0 and 359. Let's double check the details on this parameter.

```
hisafe_params("treeLineOrientation")
```

```
##
## treeLineOrientation
##  default:90
##  allowed:NA
##  min    :0
##  max    :359
##  min.sug:NA
##  max.sug:NA
```

Okay, let's try again using a new value for **treeLineOrientation**.

```
hip_sim <- define_hisafe(latitude = 60, treeLineOrientation = 90)
hip_sim
```

```
## # A tibble: 1 x 38
##   SimulationName latitude treeLineOrientation cellWidth spacingBetweenRows
##            <chr>    <dbl>               <dbl>     <dbl>              <dbl>
## 1          Sim_1       60                  90         1                 13
## # ... with 33 more variables: spacingWithinRows <dbl>,
## #   slopeIntensity <dbl>, slopeAspect <dbl>, windMeanForce <dbl>,
## #   waterTable <chr>, treeSpecies <chr>, treeHeight <dbl>,
## #   rootShape <dbl>, nbSimulations <dbl>, simulationYearStart <dbl>,
## #   simulationDayStart <dbl>, simulationNbrDays <dbl>,
## #   mainCropSpecies <chr>, interCropSpecies <chr>, mainCropItk <chr>,
```

```
## #   interCropItk <chr>, treeCropDistance <dbl>, weededAreaRadius <dbl>,
## #   weatherFile <chr>, toricSymmetry <chr>, treePruningFreq <dbl>,
## #   treePruningProp <dbl>, treePruningMaxHeight <dbl>,
## #   treeRootPruningFreq <dbl>, treeRootPruningDistance <dbl>,
## #   treeRootPruningDepth <dbl>, budBurstTempAccumulationDateStart <dbl>,
## #   budBurstAccumulatedTemp <dbl>, lueMax <dbl>,
## #   coarseRootAnoxiaResistance <dbl>, rootHalfLife <dbl>,
## #   rootAnoxiaHalfLife <dbl>, colonisationThreshold <dbl>
```

We can now see a defined Hi-sAFe simulation, which is contained within a `hip` object (for "Hi-sAFe Input Parmaeters"). Each row of a `hip` object describes a single Hi-sAFe simulation, and each column provides the value of a Hi-sAFe input parameter. For convenience, any customized parmaeters appear first in the object, which the remaining, default parameters in columns to the right. If not provided manually, a default **SimulationName** is provided for each simulation within the `hip` object.

Let's try something more complicated now. Most Hi-sAFe simulations are run in groups that vary one or more parameters over a range of values. A group of multiple, related simulations is called an "experiment". We can also use `define_hisafe()` to define an experiment by simply providing multiple values for one or more parameters. There are two methods for defining an experiment, depending on if the `define_hisafe()` agrument `factorial` is `TRUE` or `FALSE`.

If `factorial` is `FALSE`, the default, then values are recycled (i.e. such as for default behavior of `data.frame()`).

```
hip_exp_F <- define_hisafe(factorial = FALSE,
                           latitude = c(30, 60),
                           treeLineOrientation = c(0,90))
hip_exp_F
```

```
## # A tibble: 2 x 38
##   SimulationName latitude treeLineOrientation cellWidth spacingBetweenRows
##            <chr>    <dbl>               <dbl>     <dbl>              <dbl>
## 1          Sim_1       30                   0         1                 13
## 2          Sim_2       60                  90         1                 13
## # ... with 33 more variables: spacingWithinRows <dbl>,
## #   slopeIntensity <dbl>, slopeAspect <dbl>, windMeanForce <dbl>,
## #   waterTable <chr>, treeSpecies <chr>, treeHeight <dbl>,
## #   rootShape <dbl>, nbSimulations <dbl>, simulationYearStart <dbl>,
## #   simulationDayStart <dbl>, simulationNbrDays <dbl>,
## #   mainCropSpecies <chr>, interCropSpecies <chr>, mainCropItk <chr>,
## #   interCropItk <chr>, treeCropDistance <dbl>, weededAreaRadius <dbl>,
## #   weatherFile <chr>, toricSymmetry <chr>, treePruningFreq <dbl>,
## #   treePruningProp <dbl>, treePruningMaxHeight <dbl>,
## #   treeRootPruningFreq <dbl>, treeRootPruningDistance <dbl>,
## #   treeRootPruningDepth <dbl>, budBurstTempAccumulationDateStart <dbl>,
## #   budBurstAccumulatedTemp <dbl>, lueMax <dbl>,
## #   coarseRootAnoxiaResistance <dbl>, rootHalfLife <dbl>,
## #   rootAnoxiaHalfLife <dbl>, colonisationThreshold <dbl>
```

In this case, you can see that there are two simulations in the resulting `hip`, one with **latitude**=30 and **treeLineOrientation**=0, and the other with **latitude**=60 and **treeLineOrientation**=90.

If `factorial` is `TRUE`, then a factorial experiment is created, in which a simulation is defined for each possible combination of supplied values.

```
hip_exp <- define_hisafe(factorial = TRUE,
                           latitude = c(30, 60),
                           treeLineOrientation = c(0,90))
hip_exp
```

```
## # A tibble: 4 x 38
##   SimulationName latitude treeLineOrientation cellWidth spacingBetweenRows
##            <chr>    <dbl>               <dbl>     <dbl>              <dbl>
## 1          Sim_1       30                   0         1                 13
## 2          Sim_2       60                   0         1                 13
## 3          Sim_3       30                  90         1                 13
## 4          Sim_4       60                  90         1                 13
## # ... with 33 more variables: spacingWithinRows <dbl>,
## #   slopeIntensity <dbl>, slopeAspect <dbl>, windMeanForce <dbl>,
## #   waterTable <chr>, treeSpecies <chr>, treeHeight <dbl>,
## #   rootShape <dbl>, nbSimulations <dbl>, simulationYearStart <dbl>,
## #   simulationDayStart <dbl>, simulationNbrDays <dbl>,
## #   mainCropSpecies <chr>, interCropSpecies <chr>, mainCropItk <chr>,
## #   interCropItk <chr>, treeCropDistance <dbl>, weededAreaRadius <dbl>,
## #   weatherFile <chr>, toricSymmetry <chr>, treePruningFreq <dbl>,
## #   treePruningProp <dbl>, treePruningMaxHeight <dbl>,
## #   treeRootPruningFreq <dbl>, treeRootPruningDistance <dbl>,
## #   treeRootPruningDepth <dbl>, budBurstTempAccumulationDateStart <dbl>,
## #   budBurstAccumulatedTemp <dbl>, lueMax <dbl>,
## #   coarseRootAnoxiaResistance <dbl>, rootHalfLife <dbl>,
## #   rootAnoxiaHalfLife <dbl>, colonisationThreshold <dbl>
```

In this case, you can see that there are four simulations in the resulting `hip`, one for each possible combination of the supplied values of **latitude** and **treeLineOrientation**. We'll use this `hip` object for moving on to the BUILD step.

You can also use `define_hisafe_file()` to directly read a `hip` object from a *csv* file. For more information on this approach, see `?define_hisafe_file`.

2. Build

There is only one Build function: `build_hisafe()`. This function takes your `hip` object as an input and builds the actual folders/files on your computer that Hi-sAFe will read to run to the simulation(s). In addition to the `hip` object, you must supply a `path` where the simulation directory is to be created.

Optionally, you can also:

- If there are multiple simulations in the supplied `hip`, use `exp.name` to customize the name of the experiment folder, which is created first in the supplied `path` and contains each of the simulation folders.
- Specify the Hi-sAFe output profiles using `profiles`. This is important! The default for this argument is "all", which will include the very large "cells" and "voxels" outputs, slowing your Hi-sAFe simulations dramatically. To see which Hi-sAFe output profiles are supported and their export frequency, use `hisafe_profiles()`.
- Specify the saveProjectOption in Hi-sAFe, which tells Hi-sAFe to save a (large) file at the end of the simulation to allow a subsequent simulation to start where this one left off.

For more details on these options, see `?build_hisafe`.

It is good practice for the output of `build_hisafe()` to overwrite your `hip` object, as it will add the experiment/simulationpath to the object. This will enhance functionality during the RUN and READ steps.

```
hip_exp <- build_hisafe(hip = hip_exp,
                        path = "./simulations",
                        exp.name = "lat-orient_exp",
                        profiles = c("annualplot", "annualtree", "annualcrop", "monthCells", "trees", "
```

```
                          saveProjectOption = FALSE)
hip_exp
```

3. Run

Once the Hi-sAFe folders/files are created on your comupter, you are ready to run Hi-sAFe! There are two functions for running Hi-sAFe simulations: `run_hisafe()` for running a single simulation, and `run_hisafe_exp()` for running a group of simulations. To run our previous experiment on **latitude** and **treeLineOrientation** we will use `run_hisafe_exp()` by providing:

- The `hip` object.
- A `path` to the experiment folder. If `hip` updated using `build_hisafe()`, then there is no need to supply `path` as it is contained within the `hip` object.
- A character string of which simulations within the experiment to run via `simu.names`. The default "all" will run all simulations in the experiment.
- A logical `parallel` indicating whether or not to run multiple simulations in parallel on your computer. The default is `FALSE`, but it is highly encouraged to use parallel processing if your computer as the capacity! If `parallel` is `TRUE`, one fewer than the total number of available cores will be used. Alternatively, the number of cores to use can be specified directly via `num.cores`.
- The path to the Capsis folder on your computer via `capsis.path`. This is where hisafer can find Capsis and the Hi-sAFe model.

```
run_hisafe_exp(hip = hip_exp,
               simu.names = "all",
               parallel = TRUE,
               capsis.path = "/Applications/Capsis/")
```

Both `run_hisafe()` and `run_hisafe_exp()` can also be used without a `hip` object to run Hi-sAFe simulations that were not create using hisafer. To do this with `run_hisafe()`, for example, you must supply the `path` to the where the simulation folder is located and the name of the simulation via `simu.name`. Just make sure the folder/file structure is correct!

```
run_hisafe(path = "./simulations"
           simu.name  = "Sim_From_Kevin",
           capsis.path = "/Applications/Capsis/")
```

4. Read

Hi-sAFe can generate quite a lot of output data. All of this data is created within a set of (large) text files, which can be difficult to navigate with out hisafer. The Read step of Hi-sAFe experimentation reads all of this data into R for easy manipulation and analysis. Analagous to the Run step, there are two functions for reading Hi-sAFe output data: `read_hisafe()` for reading in a single simulation, and `read_hisafe_exp()` for reading in a Hi-sAFe experiment (group of simluations). `read_hisafe()` is used analagously to `run_hisafe()`.

To read in our previous experiment on **latitude** and **treeLineOrientation** we will use `read_hisafe_exp()`. The only required input is a `hip` object that has been modified by `build_hisafe()` to include the experiment's path. Alternatively, a `path` can by manually provided. You can also specify which Hi-sAFe output profiles to read in using `profiles`. While the default is to read all profiles, you may not want to read in any created "cells" or "voxels" data if you simply want to analyze annual tree growth data.

The resulting object created `read_hisafe_exp()` is a `hop` obeject (for "Hi-sAFe OutPut") of class "hop" and "hop-group". This is a list of 12 tibbles (data frames):

- `annual` - includes data from annualtree and annualplot profiles
- `daily` - includes data from trees, plot, and climate profiles

- annualcrop
- roots
- monthCells
- cells
- voxels
- variables - variable descriptions and units from all profiles
- inputs - the hip object that generated the simulation
- path - the paths to the simulation folders
- exp.plan - the manipulated input variables in the experiment
- exp.path - the path to the experiment folder

If any Hi-sAFe output profiles specified by `profiles` were not created when Hi-sAFe was run, a warning will notify you.

If any Hi-sAFe output profiles were either not created by Hi-sAFe (via the `profiles` argument of `build_hisafe_exp()`) or not selected for reading (via the `profiles` argument of `read_hisafe_exp()`), the the associated list components will be empty tibbles.

WARNING: Depending on the number of simulations, length of simulations, and which output profiles are specified, reading in Hi-sAFe data can take a few minutes!

```
hop_exp <- read_hisafe_exp(hip = hip_exp, profiles = "all")
hop_exp
```

```
## Warning: The following requested profiles do not exist:
##       --Sim_1_annualplot.txt
##       --Sim_1_annualtree.txt
##       --Sim_1_annualcrop.txt
##       --Sim_1_roots.txt
##       --Sim_1_cells.txt
##       --Sim_1_voxels.txt
```

```
##
##
## Reading:  Sim_1
## Profiles: plot, trees, climate, monthCells
## reading:  simulation inputs
## reading:  plot
## reading:  trees
## reading:  climate
## reading:  monthCells
```

```
## Warning: The following requested profiles do not exist:
##       --Sim_2_annualplot.txt
##       --Sim_2_annualtree.txt
##       --Sim_2_annualcrop.txt
##       --Sim_2_roots.txt
##       --Sim_2_cells.txt
##       --Sim_2_voxels.txt
```

```
##
##
## Reading:  Sim_2
## Profiles: plot, trees, climate, monthCells
## reading:  simulation inputs
## reading:  plot
## reading:  trees
## reading:  climate
```

```
## reading:  monthCells
```

```
## Warning: The following requested profiles do not exist:
##       --Sim_3_annualplot.txt
##       --Sim_3_annualtree.txt
##       --Sim_3_annualcrop.txt
##       --Sim_3_roots.txt
##       --Sim_3_cells.txt
##       --Sim_3_voxels.txt
```

```
##
##
## Reading:  Sim_3
## Profiles: plot, trees, climate, monthCells
## reading:  simulation inputs
## reading:  plot
## reading:  trees
## reading:  climate
## reading:  monthCells
```

```
## Warning: The following requested profiles do not exist:
##       --Sim_4_annualplot.txt
##       --Sim_4_annualtree.txt
##       --Sim_4_annualcrop.txt
##       --Sim_4_roots.txt
##       --Sim_4_cells.txt
##       --Sim_4_voxels.txt
```

```
##
##
## Reading:  Sim_4
## Profiles: plot, trees, climate, monthCells
## reading:  simulation inputs
## reading:  plot
## reading:  trees
## reading:  climate
## reading:  monthCells
```

```
## $annual
## # A tibble: 0 x 0
##
## $daily
## # A tibble: 43,832 x 220
## # Groups:   SimulationName [4]
##    SimulationName latitude treeLineOrientation       Date   Day Month
##             <fctr>   <fctr>              <fctr>     <date> <int> <int>
##  1          Sim_1       30                   0 1995-08-28    28     8
##  2          Sim_1       30                   0 1995-08-29    29     8
##  3          Sim_1       30                   0 1995-08-30    30     8
##  4          Sim_1       30                   0 1995-08-31    31     8
##  5          Sim_1       30                   0 1995-09-01     1     9
##  6          Sim_1       30                   0 1995-09-02     2     9
##  7          Sim_1       30                   0 1995-09-03     3     9
##  8          Sim_1       30                   0 1995-09-04     4     9
##  9          Sim_1       30                   0 1995-09-05     5     9
## 10          Sim_1       30                   0 1995-09-06     6     9
```

```
## # ... with 43,822 more rows, and 214 more variables: Year <int>,
## #   JulianDay <int>, stepNum <int>, activeNitHumusStock <dbl>,
## #   qLeafLitter <dbl>, biomassRestitution <dbl>, carbonBranches.x <dbl>,
## #   carbonHumification <dbl>, carbonImmobilisation <dbl>,
## #   carbonResidus <dbl>, cMicroorgVariation <dbl>, cngrain <dbl>,
## #   cnplante <dbl>, coarserootSenCn <dbl>, cropBiomass <dbl>,
## #   cropGrainNumber <dbl>, cropGrainWeight <dbl>, cropLai <dbl>,
## #   cropNitrogenBiomassStress <dbl>, cropNitrogenDemand <dbl>,
## #   cropNitrogenLaiStress <dbl>, cropNitrogenSenescenceStress <dbl>,
## #   cropPhenologicStage <dbl>, cropPlantDensity <dbl>, cropResiduCn <dbl>,
## #   cropRootDepth <dbl>, cropSenescenceWaterStress <dbl>, cropSla <dbl>,
## #   cropStomatalWaterStress <dbl>, cropTemperature <dbl>,
## #   cropTotalRootLength <dbl>, cropTurgescenceWaterStress <dbl>,
## #   cropWaterDemand <dbl>, cropWaterDemandReduced <dbl>,
## #   cropWaterPotential <dbl>, cropWaterStress <dbl>, cropYield <dbl>,
## #   drainage <dbl>, finerootSenCn <dbl>, inactiveNitHumusStock <dbl>,
## #   interceptedWater <dbl>, leafLitterCn <dbl>, maxCropBiomass <dbl>,
## #   maxCropLai <dbl>, maxCropYield <dbl>, maximalWaterStock <dbl>,
## #   microorgBiomass <dbl>, minCropBiomass <dbl>, minCropLai <dbl>,
## #   minCropYield <dbl>, mineralNitrogenStock <dbl>, nminTotal <dbl>,
## #   nminCoarseroot <dbl>, nminResCult <dbl>, nminFineroot <dbl>,
## #   nminLeaf <dbl>, nCoarseRootSenInProfHum <dbl>,
## #   nFineRootSenInProfHum <dbl>, nitrateStock <dbl>, nLeafLitter <dbl>,
## #   nitrogenAmendement <dbl>, nitrogenAvalaibleForCrops <dbl>,
## #   nitrogenBranches <dbl>, nitrogenDenitrification <dbl>,
## #   nitrogenExportation <dbl>, nitrogenExtractedByCrops <dbl>,
## #   nitrogenExtractedByTrees <dbl>,
## #   nitrogenExtractedInSaturationByCrops <dbl>,
## #   nitrogenExtractedInSaturationByTrees <dbl>,
## #   nitrogenFertilisation <dbl>, nitrogenFixation <dbl>,
## #   nitrogenHumification <dbl>, nitrogenHumusMineralisation <dbl>,
## #   nitrogenImmobilisation <dbl>, nitrogenIrrigation <dbl>,
## #   nitrogenLixiviationSTICS <dbl>, nitrogenLixiviationTOTAL <dbl>,
## #   nitrogenOrganisation <dbl>, nitrogenRain <dbl>,
## #   nitrogenResiduMineralisation <dbl>, nitrogenResidus <dbl>,
## #   nitrogenRestitution <dbl>, nitrogenVolatilisation <dbl>,
## #   nMicroorgVariation <dbl>, nminDeepRoots <dbl>, nRootSenStock <dbl>,
## #   parIncident <dbl>, parInterceptedByCrops <dbl>,
## #   parInterceptedByCropsCompetFree <dbl>, parInterceptedByTrees <dbl>,
## #   parInterceptedByTreesCompetFree <dbl>, qCoarseRootSenInProfHum <dbl>,
## #   qFineRootSenInProfHum <dbl>, qngrain <dbl>, qnplante <dbl>,
## #   rainTransmitted <dbl>, runOff <dbl>, surfaceRunOff <dbl>, tmax <dbl>,
## #   tmin <dbl>, ...
##
## $annualcrop
## # A tibble: 0 x 0
##
## $roots
## # A tibble: 0 x 0
##
## $monthCells
## # A tibble: 168,480 x 27
## # Groups:   SimulationName [4]
##    SimulationName latitude treeLineOrientation       Date   Day Month
```

```
##            <fctr>   <fctr>              <fctr>     <date> <int> <int>
##  1          Sim_1       30                   0 1995-09-01     1     9
##  2          Sim_1       30                   0 1995-09-01     1     9
##  3          Sim_1       30                   0 1995-09-01     1     9
##  4          Sim_1       30                   0 1995-09-01     1     9
##  5          Sim_1       30                   0 1995-09-01     1     9
##  6          Sim_1       30                   0 1995-09-01     1     9
##  7          Sim_1       30                   0 1995-09-01     1     9
##  8          Sim_1       30                   0 1995-09-01     1     9
##  9          Sim_1       30                   0 1995-09-01     1     9
## 10          Sim_1       30                   0 1995-09-01     1     9
## # ... with 168,470 more rows, and 21 more variables: Year <int>,
## #   JulianDay <int>, stepNum <int>, id <int>, x <dbl>, y <dbl>,
## #   cropSpeciesName <chr>, monthBiomass <dbl>, monthYield <dbl>,
## #   monthDirectPar <dbl>, monthDiffusePar <dbl>,
## #   monthDiffuseParIncident <dbl>, monthDiffuseParIntercepted <dbl>,
## #   monthDirectParIncident <dbl>, monthDirectParIntercepted <dbl>,
## #   monthEai <dbl>, monthLai <dbl>, monthRelativeDiffuseParIncident <dbl>,
## #   monthRelativeDirectParIncident <dbl>,
## #   monthRelativeTotalParIncident <dbl>, monthVisibleSky <dbl>
##
## $cells
## # A tibble: 0 x 0
##
## $voxels
## # A tibble: 0 x 0
##
## $variables
## # A tibble: 225 x 6
##     Subject   SubjectId         VariableName Units
##       <chr>       <chr>                <chr> <chr>
##  1 SafePlot Plot totals  activeNitHumusStock kg/ha
##  2 SafePlot Plot totals          qLeafLitter kg/ha
##  3 SafePlot Plot totals   biomassRestitution  t/ha
##  4 SafePlot Plot totals       carbonBranches kg/ha
##  5 SafePlot Plot totals   carbonHumification kg/ha
##  6 SafePlot Plot totals carbonImmobilisation     -
##  7 SafePlot Plot totals        carbonResidus     -
##  8 SafePlot Plot totals   cMicroorgVariation kg/ha
##  9 SafePlot Plot totals              cngrain    mm
## 10 SafePlot Plot totals             cnplante    mm
## # ... with 215 more rows, and 2 more variables: Description <chr>,
## #   VariableClass <chr>
##
## $inputs
## # A tibble: 4 x 27
##   SimulationName latitude treeLineOrientation cellWidth spacingBetweenRows
##            <chr>    <int>               <int>     <int>              <int>
## 1          Sim_1       30                   0         1                 13
## 2          Sim_2       60                   0         1                 13
## 3          Sim_3       30                  90         1                 13
## 4          Sim_4       60                  90         1                 13
## # ... with 22 more variables: spacingWithinRows <int>,
## #   slopeIntensity <int>, slopeAspect <int>, windMeanForce <int>,
```

```
## #   treeSpecies <chr>, treeHeight <int>, rootShape <int>,
## #   nbSimulations <int>, simulationYearStart <int>,
## #   simulationDayStart <int>, mainCropSpecies <chr>,
## #   interCropSpecies <chr>, treeCropDistance <dbl>,
## #   weededAreaRadius <int>, weatherFile <chr>, toricSymmetry <chr>,
## #   treePruningFreq <int>, treePruningProp <dbl>,
## #   treePruningMaxHeight <int>, treeRootPruningFreq <int>,
## #   treeRootPruningDistance <dbl>, treeRootPruningDepth <int>
##
## $path
## # A tibble: 4 x 1
##                 path
##                <chr>
## 1 ./experiment/Sim_1
## 2 ./experiment/Sim_2
## 3 ./experiment/Sim_3
## 4 ./experiment/Sim_4
##
## $exp.plan
## # A tibble: 4 x 3
##   SimulationName latitude treeLineOrientation
##           <fctr>   <fctr>              <fctr>
## 1          Sim_1       30                   0
## 2          Sim_2       60                   0
## 3          Sim_3       30                  90
## 4          Sim_4       60                  90
##
## $exp.path
## [1] "./experiment"
##
## attr(,"class")
## [1] "hop-group" "hop"       "list"
```

5. Diagnostics

Prior to performing any analyses using your `hop` data, it is highly recommended to check some basic diagnostics on your simluations to ensure that the simulations ran as expected. To do so, hisafer provides some basic diagnostic functions that allow you to compare you various simulations side by side:

- `diag_hisafe_ts()` - plots a timeseries plot for each variable in the "annual" and "daily" data of a `hop` object.
- `diag_hisafe_monthcells()` - plots a full range of tile plots for each variable in the "monthCells" data of a `hop` object.

All outputs from these Diagnostics functions will be saved to a "diagnostics" folder within the simulation folder of the `hop`.

6. Analysis

There are many different types of analyses that can be performed with Hi-sAFe output data. Conseuqnetly, there are no strict "analysis" functions in hisafer. However, there are some plotting functions that can aid in your analyses:

- `plot_hisafe_ts()` - plots "annual" or "daily" timeseries data

- `plot_hisafe_monthcells()` - plots “monthCells” data
- `plot_hisafe_cells()` - plots “cells” data
- `plot_hisafe_voxels()` - plots “voxels” data

Sometimes, during the Analysis phase, you will realize that the values of **SimulationName** that you provided during the Define phase are not quite what you wanted. To rename each **SimulationName** in your `hop` object, you can use `simu_rename()`.