Intro to healthiar

- About `healthiar`
  - Refresher - Burden of disease with relative risk
  - Refresher - Burden of disease with absolute risk
  - Package overview
- Abbreviations
- `healthiar` Examples
  - Example: `attribute_health()` with relative risk
  - Example: `attribute_health()` with relative risk & uncertainty
  - Example: `attribute_health()` with absolute risk
  - Example: `attribute_health()` for multiple geographic units (iteration) (relative risk)
  - Example: `attribute_health()` for multiple geographic units (iteration) (absolute risk)
  - Example: `attribute_health()` with user-defined ERF
  - Example: `compare()`
  - Example: `summarize_uncertainty()`
  - Example: `monetize()`
  - Example: `cba()`
  - Example: `attribute_lifetable()` for YLL
  - Example: `attribute_lifetable()` for pre-mature deaths
  - Example: `attribute_health()` for YLD
  - Example: `daly()`
  - Example: `multiexpose()`
  - Example: `prepare_mdi`
- Post-`healthiar` workflow
  - Export results
  - Visualize results

Hi there!

This vignette will tell you about `healthiar` and show you how to use `healthiar` with the help of examples.

NOTE: By using healthiar you agree to the terms of use and confirm you have read the disclaimer.

NOTE: the development of `healthiar` is still ongoing. Any feedback regarding bugs, unclear documentation, … is welcome and highly appreciated. Please provide feedback via a GitHub issue.

About `healthiar`

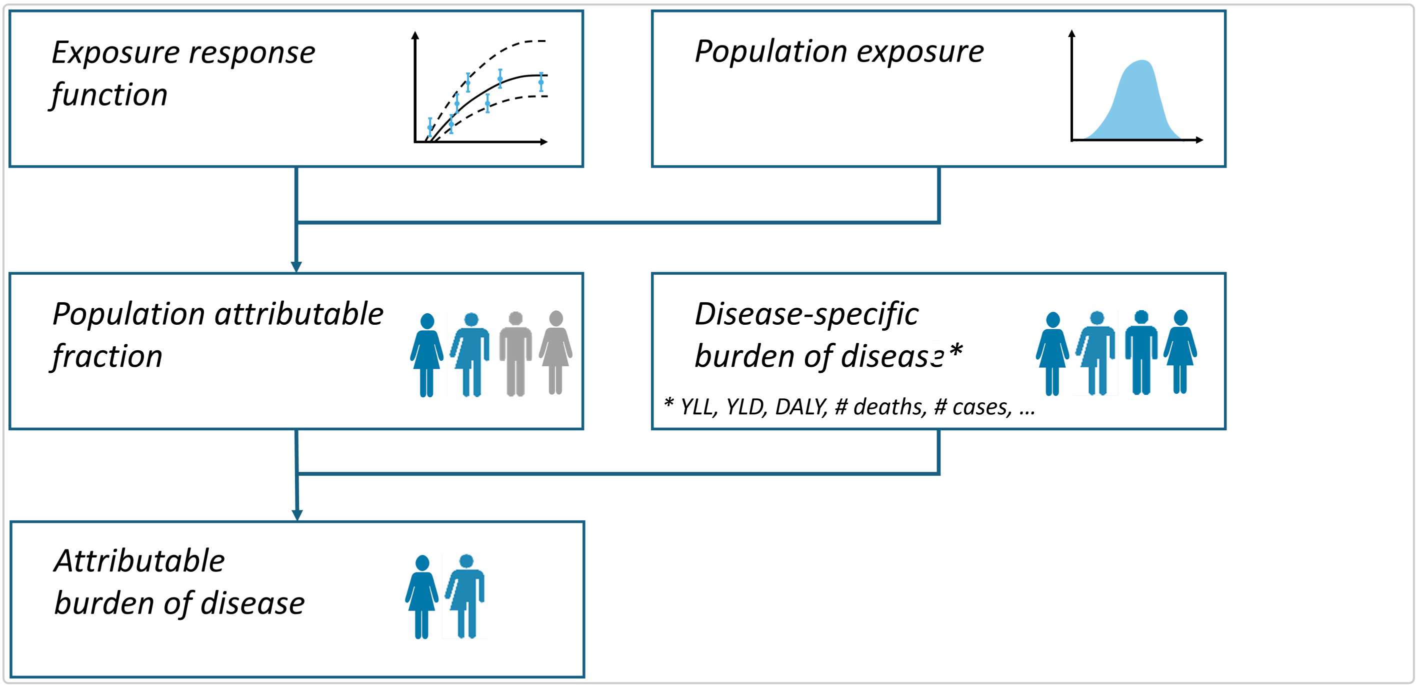
The `healthiar` functions allow you to quantify and monetize the health impacts of environmental stressors (air pollution & noise)

`healthiar` core *family members* (= functions)

- `attribute_health()` to environmental exposure (relative or absolute risk)
- `summarize_uncertainty()` Monte Carlo simulation
- `compare()` two scenarios
- `monetize()` health impacts
- `attribute_lifetable()` life table analysis (RR & AR)
- `attribute_mod()` modify an existing assessment
- `cba()` cost-benefit analysis
- `get_daly()` by adding up YLL & YLD
- `multiexpose()` approaches to consider exposure to 2 air pollutants at the same time

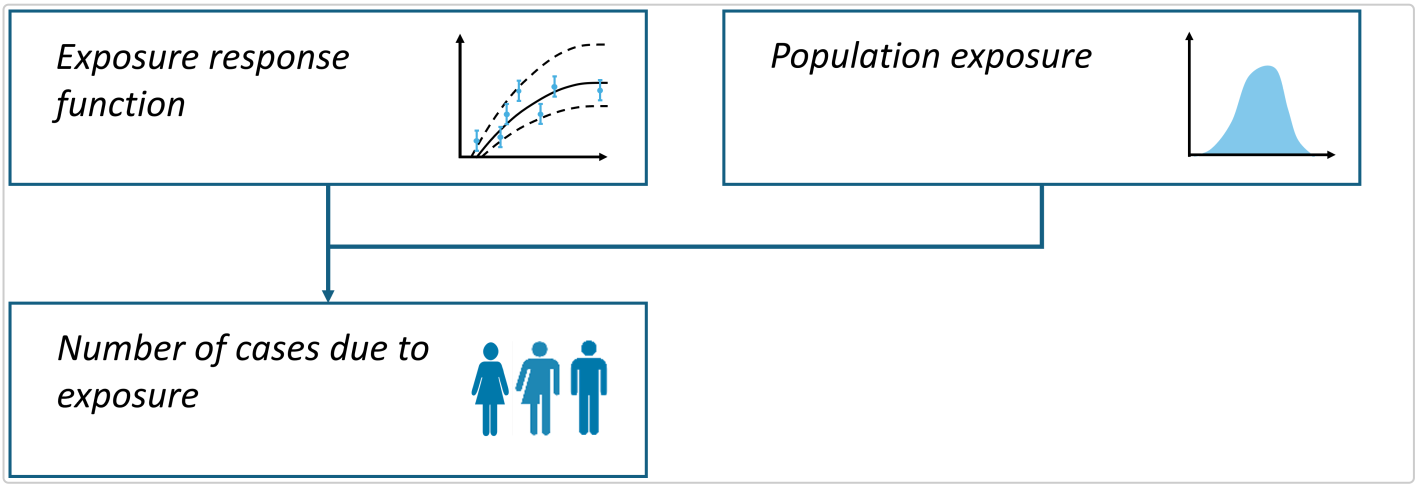
    - `socialize()` to discover inequalities in health impacts
    - `prepare_mdi()` creates the BEST-COST MDI (Multidimensional Deprivation Index)

Refresher - Burden of disease with relative risk



Exposure response function
Population exposure
Population attributable fraction
Disease-specific burden of disease*
* YLL, YLD, DALY, # deaths, # cases, ...
Attributable burden of disease

Refresher - Burden of disease with absolute risk



Package overview

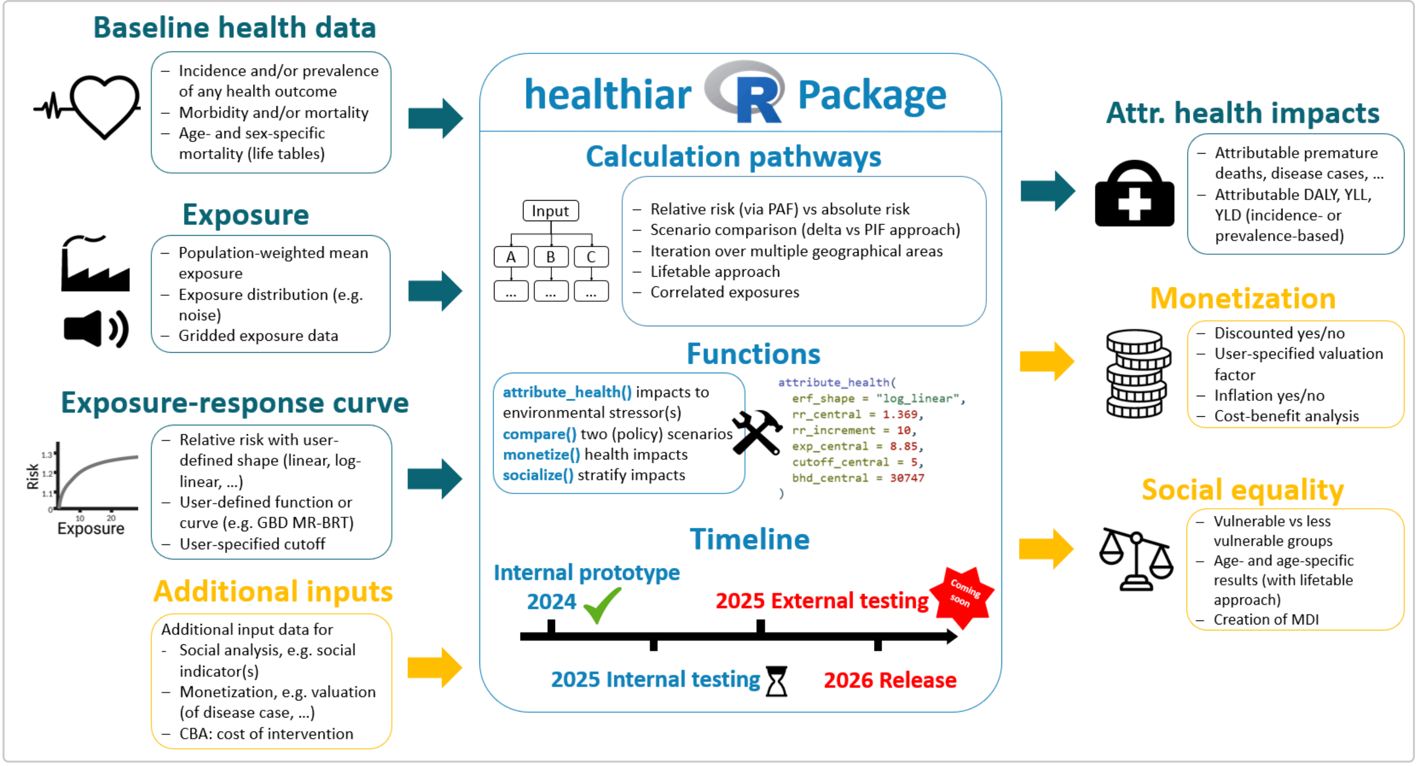

Baseline health data
Incidence and/or prevalence of any health outcome
Morbidity and/or mortality
Age- and sex-specific mortality (life tables)
Exposure
Population-weighted mean exposure
Exposure distribution (e.g. noise)
Gridded exposure data
Exposure-response curve
Risk
Exposure
Relative risk with user-defined shape (linear, log-linear, …)
User-defined function or curve (e.g. GBD MR-BRT)
User-specified cutoff
Additional inputs
Additional input data for
Social analysis, e.g. social indicator(s)
Monetization, e.g. valuation (of disease case, …)
CBA: cost of intervention
healthiar Package
Calculation pathways
Input
A
B
C
Relative risk (via PAF) vs absolute risk
Scenario comparison (delta vs PIF approach)
Iteration over multiple geographical areas
Lifetable approach
Correlated exposures
Functions
attribute_health() impacts to environmental stressor(s)
compare() two (policy) scenarios
monetize() health impacts
socialize() stratify impacts
attribute_health(erf_shape = "log_linear", rr_central = 1.369, rr_increment = 10, exp_central = 8.85, cutoff_central = 5, bhd_central = 30747)
Timeline
Internal prototype 2024
2025 External testing
Coming soon
2025 Internal testing
2026 Release
Attr. health impacts
Attributable premature deaths, disease cases, …
Attributable DALY, YLL, YLD (incidence- or prevalence-based)
Monetization
Discounted yes/no
User-specified valuation factor
Inflation yes/no
Cost-benefit analysis
Social equality
Vulnerable vs less vulnerable groups
Age- and age-specific results (with lifetable approach)
Creation of MDI

Figure: `healthiar` overview

Abbreviations

RR/rr - relative risk BHD/bhd - baseline health data CI - confidence interval

`healthiar` Examples

Example: `attribute_health()` with relative risk

Goal: attribute COPD cases to PM2.5 air pollution exposure

Function call - Hard coded

```
results_pm_copd <- attribute_health(
  erf_shape = "log_linear", # shape of the exposure-response function (ERF)
  rr_central = 1.369, # relative risk (RR) central estimate
  rr_increment = 10,  # increment for which relative risk is valid (in μg / m^3)
  exp_central = 8.85, # PM2.5 exposure (in μg / m^3) (here: population-weighted)
  cutoff_central = 5, # cutoff (in μg / m^3) below which no health effects occur
  bhd_central = 30747 # baseline health data (BHD; here: COPD incidence)
)
```

For alternative ERF shapes see the function documentation of `attribute_health()`.

Function call - Pre-loaded data

`healthiar` comes with some example data that start with `exdat_` that allow you to test functions. Some of these example data will be used in some examples in this vignette.

Now we call `attribute_health` with input data from the `healthiar` example data. Note that we can easily provide input data to the function argument using the `$` operator.

```
results_pm_copd <- attribute_health(
  erf_shape = "log_linear",
  rr_central = exdat_pm_copd$relative_risk,
  rr_increment = 10,
  exp_central = exdat_pm_copd$mean_concentration,
  cutoff_central = exdat_pm_copd$cut_off_value,
  bhd_central = exdat_pm_copd$incidence
)
```

Output structure

Every `attribute_health()` output consists of two lists ("folders")

- `health_main` contains the main results
- `health_detailed` detailed results and additional info about the assessment
  - `results_raw` tibble that contains the detailed results
  - `input_args` contains a complete list of all arguments used in the assessment (including some internal arguments)
  - `input_table` tibble that contains all input data entered by the user
  - `results_disaggregated` (relevant for assessments for more than one geographic unit) contains results per geo unit
  - `results_agg_sex` & `results_agg_age` contains results aggregated by sex and age groups, respectively

NOTE: `attribute_lifetable()` creates additional output that is specific to life table calculations

Main results

Let's inspect the main results

There exist different, equivalent ways of accessing the output

- With `$` operator: `results_pm_copd$health_main$impact_rounded` (as in the example above)
- By mouse: go to the *Environment* tab in RStudio and click on the variable you want to inspect, and then open the `health_main` results table
- With `[[]]` operator `results_pm_copd[["health_main"]]`
- With `pluck()` & `pull()`: use the `purrr::pluck` function to select a list and then the `dplyr::pull` function extract values from a specified column, e.g. `results_pm_copd |> purrr::pluck("health_main") |> dplyr::pull("impact_rounded")`

```
results_pm_copd$health_main
#> # A tibble: 1 × 25
#>   geo_id_disaggregated sex   erf_ci  exp_ci  bhd_ci cutoff_ci duration_ci impact
#>   <chr>                <chr> <chr>   <chr>   <chr>  <chr>     <chr>        <dbl>
#> 1 a                    total central central centr… central   central      3502.
#> # ℹ 17 more variables: impact_rounded <dbl>, pop_fraction <dbl>,
#> #   approach_risk <chr>, rr_increment <dbl>, erf_shape <chr>,
#> #   prop_pop_exp <dbl>, exposure_dimension <int>, exposure_type <chr>,
#> #   exp <dbl>, bhd <dbl>, cutoff <dbl>, duration <dbl>, rr <dbl>,
#> #   is_lifetable <lgl>, pop_fraction_type <chr>, rr_at_exp <dbl>, age <chr>
```

It is a table of the format `tibble` (very similar to a `data.frame`) of 3 rows and 23 columns. Let's zoom in on some relevant aspects

```
results_pm_copd$health_main |>
  select(exp, bhd, rr, erf_ci, pop_fraction, impact_rounded) |>
  knitr::kable() # For formatting reasons only: prints tibble in nice layout
```

exp	bhd	rr	erf_ci	pop_fraction	impact_rounded
8.85	30747	1.369	central	0.1138961	3502

Interpretation: this table shows us that exposure was 8.85 $\mu g/m^3$, the baseline health data (`bhd_central`) was 30747 (COPD incidence in this instance). The 1st row further shows that the impact attributable to this exposure using the central relative risk (`rr_central`) estimate of 1.369 is 3502 COPD cases, or ~11% of all baseline cases.

Some of the most results columns include:

- *impact_rounded* Rounded attributable health impact/burden
- *impact* Raw impact/burden
- *pop_fraction* Population attributable fraction (PAF)

NOTE: the main output contains more columns that provide additional information about the assessment, such as intermediate results and assessment specifications, …

Example: `attribute_health()` with relative risk & uncertainty

Goal: attribute COPD cases to PM2.5 exposure

Now we will make a similar function call, but include uncertainty in several input arguments

Function call

```
results_pm_copd <- attribute_health(
    erf_shape = "log_linear",
    rr_central = 1.369,
    rr_lower = 1.124, # lower 95% confidence interval (CI) bound of RR
    rr_upper = 1.664, # upper 95% CI bound of RR
    rr_increment = 10,
    exp_central = 8.85,
    exp_lower = 8, # lower 95% CI bound of exposure
    exp_upper = 10, # upper 95% CI bound of exposure
    cutoff_central = 5,
    bhd_central = 30747,
    bhd_lower = 28000, # lower 95% confidence interval estimate of BHD
    bhd_upper = 32000 # upper 95% confidence interval estimate of BHD
)
```

Detailed results

Let’s inspect the detailed results:

erf_ci	exp_ci	bhd_ci	impact_rounded
central	central	central	3502
lower	central	central	1353
upper	central	central	5474
central	central	lower	3189
lower	central	lower	1232
upper	central	lower	4985

erf_ci	exp_ci	bhd_ci	impact_rounded
central	central	upper	3645
lower	central	upper	1408
upper	central	upper	5697

Each row contains the estimated attributable cases (`impact_rounded`) obtained by the input data specified in the columns ending in "_ci" and the other calculation pathway specifications in that row (not shown).

- The 1st contains the estimated attributable impact when using the central estimates of relative risk, exposure and baseline health data
- The 2nd row shows the impact when using the central estimates of the relative risk, exposure in combination with the lower estimate of the baseline health data
- …

NOTE: only 9 of the 27 possible combinations are displayed due to space constraints

NOTE: only a selection of columns are shown

Example: `attribute_health()` with absolute risk

Goal: attribute cases of high annoyance to (road traffic) noise exposure

Function call

```
results_noise_ha <- attribute_health(
  approach_risk = "absolute_risk", # default is "relative_risk"
  exp_central = c(57.5, 62.5, 67.5, 72.5, 77.5), # mean of the exposure categories
  pop_exp = c(387500, 286000, 191800, 72200, 7700), # population exposed per exposure category
  erf_eq_central = "78.9270-3.1162*c+0.0342*c^2" # exposure-response function
)
```

The `erf_eq_central` argument can digest other types of functions (see section [Example: attribute_health() with user-defined ERF])

Main results

erf_eq	erf_ci	impact_rounded
78.9270-3.1162*c+0.0342*c^2	central	174232

Results per noise exposure band

```
results_noise_ha$health_detailed$results_raw
```

exposure_dimension	exp	pop_exp	impact
1	57.5	387500	49674.594
2	62.5	286000	50788.595
3	67.5	191800	46813.105
4	72.5	72200	23657.232

exposure_dimension	exp	pop_exp	impact
5	77.5	7700	3298.314

Alternatively, it's also possible to assess the impacts for a single noise exposure band.

```r
results_noise_ha <- attribute_health(
  approach_risk = "absolute_risk",
  exp_central = 57.5,
  pop_exp = 387500,
  erf_eq_central = "78.9270-3.1162*c+0.0342*c^2"
)
```

```r
results_noise_ha$health_main
```

exposure_dimension	impact
1	49674.59

Example: `attribute_health()` for multiple geographic units (iteration) (relative risk)

Goal: attribute disease cases to PM2.5 exposure in multiple geographic units, such as municipalities, provinces, countries, …

- Enter unique geo ID's as a vector to the `geo_id_disaggregated` argument (e.g. municipality names)
- Optional: aggregate geo unit-specific results by providing higher-level ID's (e.g. region names) to the `geo_id_aggregated` argument (as a vector)

Function call

```r
results_iteration <- attribute_health(
    geo_id_disaggregated = c("Zurich", "Basel", "Geneva", "Ticino", "Valais"),
    geo_id_aggregated = c("Ger","Ger","Fra","Ita","Fra"),
    rr_central = 1.369,
    rr_increment = 10,
    cutoff_central = 5,
    erf_shape = "log_linear",
    exp_central = c(11, 11, 10, 8, 7),
    bhd_central = c(4000, 2500, 3000, 1500, 500)
)
```

In this example we want to aggregate the lower-level geographic units (municipalities) by the higher-level language region (`"Ger"`, `"Fra"`, `"Ita"`)

Main results

The main output contains aggregated results if available, or disaggregated results if no aggregation IDs were provided

```r
results_iteration$health_main
```

geo_id_aggregated	impact_rounded	erf_ci	exp_ci	bhd_ci
Ger	1116	central	central	central
Fra	466	central	central	central
Ita	135	central	central	central

Main (= *aggregated*). The cumulative / summed number of stroke cases attributable to PM2.5 exposure in the 5 geo units is 1717 (using a relative risk of 1.369).

Detailed results

The geo unit-specific information and results are stored under `health_detailed`. Filter for the geo unit-specific results as follows

```
results_iteration$health_detailed$results_raw
```

geo_id_disaggregated	geo_id_aggregated	impact_rounded
Zurich	Ger	687
Basel	Ger	429
Geneva	Fra	436
Ticino	Ita	135
Valais	Fra	30

Besides the results per geo unit, `health_detailed` also contains impacts obtained through all combinations of input data central, lower and upper estimates (not shown).

Example: `attribute_health()` for multiple geographic units (iteration) (absolute risk)

(See iteration example with relative risk above for a more detailed explanation multiple geo unit analysis)

Goal: attribute high annoyance cases to noise exposure in rural and urban areas

Function call

```
results_iteration_ar <- attribute_health(
    geo_id_disaggregated = c(rep("rural", times = 5), rep("urban", times = 5)), # 2 geographic areas
    ## Both the rural and urban areas belong to the higher-level "total" region
    geo_id_aggregated = "total",
    approach_risk = "absolute_risk",
    exp_central =
      ## List of 2 elements
      ## Each element a numeric vector that contains the 5 exposure means
      rep(exdat_noise_ha$exposure_mean, times = 2),
    pop_exp = c(
      exdat_noise_ha$population_exposed_rural, # Rural population exposed
      exdat_noise_ha$population_exposed_urban # Urban population exposed
      ),
    erf_eq_central = "78.9270-3.1162*c+0.0342*c^2"
)
```

NOTE: the length of the input vectors fed to `geo_id_disaggregated`, `exp_central`, `pop_exp` must match and must be (number of geo units) x (number of exposure categories) = 2 x 5 = **10**, because we have 2 lower level geo units (`"rural"` and `"urban"`) and 5 exposure categories.

Main results

Contains the aggregated results (i.e. sum of impacts in rural and urban areas).

```
results_iteration_ar$health_main
```

geo_id_aggregated	impact_rounded	erf_ci	exp_ci
total	174232	central	central

Detailed results

Impact by geo area, in this case impact in rural and in urban area.

```
results_iteration_ar$health_detailed$results_raw
```

geo_id_disaggregated	geo_id_aggregated	impact
rural	total	7640.273
rural	total	8790.334
rural	total	5979.776
rural	total	917.455
rural	total	0.000
urban	total	42034.321
urban	total	41998.261
urban	total	40833.329
urban	total	22739.778
urban	total	3298.314

Example: `attribute_health()` with user-defined ERF

todo

Example: `compare()`

Goal: comparison of two scenarios

Function call

1. Use `attribute_health()` to calculate burden of scenarios A & B

```
scenario_A <- attribute_health(
    exp_central = 8.85,   # EXPOSURE 1
    cutoff_central = 5,
    bhd_central = 25000,
    approach_risk = "relative_risk",
    erf_shape = "log_linear",
    rr_central = 1.118,
    rr_increment = 10)


scenario_B <- attribute_health(
    exp_central = 6,      # EXPOSURE 2
    cutoff_central = 5,
    bhd_central = 25000,
    approach_risk = "relative_risk",
    erf_shape = "log_linear",
    rr_central = 1.118,
    rr_increment = 10)
```

Alternatively, the function `attribute_mod()` can be used to modify an existing scenario, e.g. `scenario_A`

```
scenario_B <- attribute_mod(
  output_attribute_1 = scenario_A,
  exp_central = 6
)
```

2. Use `compare()` to compare scenarios A & B

```
results_comparison <- compare(

  approach_comparison = "delta", # or "pif" (population impact fraction)

  output_attribute_1 = scenario_A,

  output_attribute_2 = scenario_B
)
```

Main results

The `compare()` results are very similar to `attribute_health()` results:

- `health_main` contains main comparison results
- `health_detailed`
  - `results_raw` raw comparison results
  - `scenario_1` contains results of scenario 1 (scenario A in our case)
  - `scenario_2` contains results of scenario 2 (scenario B in our case)

```
results_comparison$health_main
```

impact	impact_rounded	impact_1	impact_2	bhd	exp_1	exp_2
773.5564	774	1050.86	277.304	25000	8.85	6

Example: `summarize_uncertainty()`

You can do a Monte Carlo uncertainty analysis via the `summarize_uncertainty` function.

The outcome of the Monte Carlo analysis is added to the variable entered as the `results` argument, which is `results_pm_copd` in our case.

Two folders are added:

- `uncertainty_main` contains the central estimate and the corresponding 95% confidence intervals obtained through the Monte Carlo assessment
- `uncertainty_detailed` contains all `n_sim` simulations of the Monte Carlo assessment

Function call

```
results_pm_copd_summarized <-
  summarize_uncertainty(
    output_attribute = results_pm_copd,
    n_sim = 100
)
```

Main results

```
print(
  results_pm_copd_summarized |>
  purrr::pluck("uncertainty_main")
  )
#> # A tibble: 3 × 4
#>   geo_id_disaggregated impact_ci        impact impact_rounded
#>   <chr>                <chr>             <dbl>          <dbl>
#> 1 a                    central_estimate  3137.           3137
#> 2 a                    lower_estimate    1578.           1578
#> 3 a                    upper_estimate    5650.           5650
```

Detailed results

The folder `uncertainty_detailed` contains all single simulations. Let's look at the impact of the first simulation.

```
results_pm_copd_summarized$uncertainty_detailed$by_simulation$impact[1]
#> NULL
```

We can also look at any simulation in detail. Here's the first one:

```
results_pm_copd_summarized$uncertainty_detailed$by_simulation$health_main[1]
#> NULL
```

We see that the detailed results contain the complete `n_sim` simulations (one row for each simulation), which are saved in nested tibbles in the column `health_main`.

Example: `monetize()`

You can monetize the obtained health impacts via the `include_monetization` function.

Function call

```
results_pm_copd <-
  monetize(
    output_attribute = results_pm_copd,
    discount_shape = "exponential",
    discount_rate = 0.03,
    discount_years = 5,
    valuation = 50000 # E.g. EURO
  )
```

Main results

```
results_pm_copd$monetization_main |>
  select(erf_ci, monetized_impact) |>
  knitr::kable()
```

erf_ci	monetized_impact
central	151041153
lower	58358321
upper	236091201

We see that the monetized impact (discounted) is more than 160 million EURO.

The outcome of the monetization is added to the variable entered to the `output_attribute` argument, which is `results_pm_copd` in our case.

Two folders are added:

- `monetization_main` contains the central monetization estimate and the corresponding 95% confidence intervals obtained through the specified monetization
- `monetization_detailed` contains the monetized results for each unique combination of the input variable estimates that were provided to the initial `attribute_health()` call

Alternatively, you can also monetize (attributable) health impacts from a non-`healthiar` source.

```
results <- monetize(
  impact = 1151,
  valuation = 100
)
```

Example: `cba()`

Adding a cost-benefit analysis (CBA) using the results

Let's imagine we design a policy that would reduce air pollution to 5 $\mu g/m^3$, which is the concentration specified in the `cutoff_central` argument in the initial `attribute_health()` call. So we could avoid all COPD cases attributed to air pollution.

What would be the monetary benefit of such a policy, considering also the cost to implement the policy (estimated at 100 million EURO)? We can find out using `healthiar`'s `cba()` function.

Function call

```
cba <-
  cba(
    output_attribute = results_pm_copd,
    valuation = 50000,
    cost = 100000000,
    discount_shape = "exponential",
    discount_rate_benefit = 0.03,
    discount_rate_cost = 0.03,
    discount_years_benefit = 5,
    discount_years_cost = 5
  )
```

Main results

```
cba$cba_main |>
  select(benefit, cost, net_benefit) |>
  knitr::kable()
```

benefit	cost	net_benefit
151041153	86260878	64780274
58358321	86260878	-27902557
236091201	86260878	149830323

We see that the central and upper 95& confidence interval estimates of avoided attributable COPD cases result in a net monetary benefit of the policy, while the lower 95% confidence interval estimate results in a net cost!

The outcome of the CBA is contained in two folders are added:

- `cba_main` contains the central estimate and the corresponding 95% confidence intervals obtained
- `cba_detailed` contains additional intermediate results for both cost and benefit
  - `benefit` contains results `by_year` and raw results `health_raw`
  - `cost`

Example: `attribute_lifetable()` for YLL

Use the two arguments `approach_exposure` and `approach_newborns` to modify the lifetable calculation pathway

- `approach_exposure`
  - `"single_year"` (default) population is exposed for a single year and the impacts of that exposure are calculated
  - `"constant"` population is exposed every year
- `approach_newborns`
  - `"without_newborns"` (default) assumes that the population in the year of analysis is followed over time, without considering newborns being born
  - `"with_newborns"` assumes that for each year after the year of analysis n babies are born, with n being equal to the (male and female) population aged 0 that is provided in the arguments population_midyear_male and population_midyear_female

Function call

```
results_pm_yll <- attribute_lifetable(
  health_outcome = "yll",
  approach_exposure = "single_year",
  exp_central = 8.85,
  prop_pop_exp = 1,
  cutoff_central = 5,
  rr_central =  1.118,
  rr_increment = 10,
  erf_shape = "log_linear",
  sex = rep(c("male", "female"), each = 100),
  age_group = rep(0:99, times = 2),
  bhd_central = c(exdat_pop_1$number_of_deaths_male, exdat_pop_1$number_of_deaths_female),
  population = c(exdat_pop_male$population_2019, exdat_pop_female$population_2019),
  year_of_analysis = 2019,
  min_age = 20
)
```

Main results

Total YLL impact (for females) attributable to exposure

```
results_pm_yll$health_main$impact
#> [1] 28810.05
```

Detailed results

Detailed total YLL results per year, per age for females (males also available)

TODO: note that even thought the column name says `population_...` these are actually the YLL… column names to be adjusted

```
results_pm_yll$health_detailed$results_raw$yll_nest$total_a_central_central_central
```

age_start	population_2019	population_2020	population_2021
91	29.480668	57.48235	52.42265
92	27.542205	54.72128	49.72379
93	25.166426	50.65989	46.42645
94	22.111703	45.84761	42.11525
95	18.514985	39.91634	37.33394
96	14.505077	33.10404	31.83396
97	11.222749	25.68958	25.83831
98	8.170422	19.69677	19.62963
99	31.770254	11.07186	11.48327

Other relevant detailed results include

- `results_pm_yll[["health_detailed"]][["results_raw"]][["pop_baseline_scenario_nest"]]` population over time in baseline scenario (no exposure)
- `results_pm_yll[["health_detailed"]][["results_raw"]][["pop_impacted_scenario_nest"]]` population over time in impact scenario (with exposure)
- `results_pm_yll[["health_detailed"]][["results_raw"]][["premature_deaths_nest"]][["female_a_central_central_central_central"]] |> select(age_start, age_end, deaths_2019)` pre-mature deaths (females) in the year of exposure (here: 2019) attributable to exposure

Note: you can also access these data frames using the `$` operator

Example: `attribute_lifetable()` for pre-mature deaths

See example above “Example: `attribute_lifetable()` for YLL” for some additional info on the lifetable calculations

Function call

Main results

Total pre-mature deaths attributable to exposure

Detailed results

Example: `attribute_health()` for YLD

Function call

```
results_pm_copd_yld  <- attribute_health(
    exp_central = 8.85,
    prop_pop_exp = 1,
    cutoff_central = 5,
    bhd_central = 1000,
    rr_central = 1.1,
    rr_increment = 10,
    erf_shape = "log_linear",
    info = "pm2.5_yld",
    duration_central = 100,
    dw_central = 1
)
```

Main results

```
results_pm_copd_yld$health_main
```

erf_ci	impact
central	3602.934

Example: `daly()`

Function call

```
results_daly <- daly(
     output_attribute_yll = results_pm_yll,
     output_attribute_yld = results_pm_copd_yld
)
```

Main results

```r
## YLL
results_daly$health_main$impact_yll
#> [1] 28810.05
## YLD
results_daly$health_main$impact_yld
#> [1] 3602.934
## DALY
results_daly$health_main$impact_rounded
#> [1] 32413
```

Example: `multiexpose()`

Function call

```r
results_pm_copd <- attribute_health(
  erf_shape = "log_linear",
  rr_central = 1.369,
  rr_increment = 10,
  exp_central = 8.85,
  cutoff_central = 5,
  bhd_central = 30747
)

results_no2_copd <- attribute_mod(
  output_attribute_1 = results_pm_copd,
  exp_central = 10.9,
  rr_central = 1.031
)

results_multiplicative <- multiexpose(
  output_attribute_1 = results_pm_copd,
  output_attribute_2 = results_no2_copd,
  exposure_name_1 = "pm2.5",
  exposure_name_2 = "no2",
  approach = "multiplicative"
)
```

Main results

```r
results_multiplicative$health_main
```

impact_rounded
3988

Example: `prepare_mdi`

Note: the `ggplot2` must be installed in order to use this function!

Function call

Main results

Function output includes

- In the console
    - DESCRIPTIVE STATISTICS
    - PEARSON'S CORRELATION COEFFICIENTS
    - PEARSON'S CORRELATION COEFFICIENTS
    - CRONBACH'S α, including the reliability rating this value indicates
- Plots
    - Boxplots of the single indicators
    - Histogram of the MDI's for the geo units with a normal distribution curve

Post-`healthiar` workflow

Export results

Export as `.csv` file

```r
write.csv(x = results_pm_copd$health_main, file = "exported_results/results_pm_copd.csv")
```

Save as `.Rdata` file

```r
save(results_pm_copd, file = "exported_results/results_pm_copd.Rdata")
```

Export to Excel (as `.xlsx` file)

```r
openxlsx::write.xlsx(x = results_pm_copd$health_main, file = "exported_results/results_pm_copd.xlsx")
```

Visualize results

Visualization is out of scope of `healthiar`. You can visualize in

- R, e.g. with the `ggplot2` package (online book by the creator)
- Excel (export results first)
- Other tools