Measure 1: Savings potentials, Normalization, Proportionalization and Creation of Plots and Tables

Marlene Kindler

2025-03-19

Code for the calculation of the savings potential using the results of the LCA-studies. Normalization and Proportionalization is performed as well as the creation of plots and tables.

To execute, change:
- Login data for Nextcloud
- Adjust download location
- Adjust upload location
- Adjust URLs for uploading

Load Packages:

```
# install.packages("httr")
# install.packages("readxl")
# install.packages("dplyr")
# install.packages("writexl")
# install.packages("ggplot2")
# install.packages("tidyr")
# install.packages("openxlsx")
# install.packages("flextable")
# install.packages("officer")

library(readxl)
library(httr)
library(dplyr)
library(writexl)
library(ggplot2)
library(tidyr)
library(openxlsx)
library(flextable)
library(officer)
```

Parameters

Parameters to be adjusted:

```
measure <- "01" # "01", "02", "03" or "04"
sheet <- 2 # 2,3,4 or 5
```

```r
# name for columns
name_with_TP <- "digital TP"
name_with_BT <- "digital BT"
name_with_MT <- "digital MT"
name_without_TP <- "analogue TP"
name_without_BT <- "analogue BT"
name_without_MT <- "analogue MT"
name_SP_TP <- "SP TP"
name_SP_BT <- "SP BT"
name_SP_MT <- "SP MT"
name_SP_total <- "SP total"

name_with_TP_norm <- "norm. digital TP"
name_with_BT_norm <- "norm. digital BT"
name_with_MT_norm <- "norm. digital MT"
name_without_TP_norm <- "norm. analogue TP"
name_without_BT_norm <- "norm. analogue BT"
name_without_MT_norm <- "norm. analogue MT"
name_SP_TP_norm <- "norm. SP TP"
name_SP_BT_norm <- "norm. SP BT"
name_SP_MT_norm <- "norm. SP MT"
name_SP_total_norm <- "norm. SP total"

name_with_TP_prop <- "prop. digital TP"
name_with_BT_prop <- "prop. digital BT"
name_with_MT_prop <- "prop. digital MT"
name_without_TP_prop <- "prop. analogue TP"
name_without_BT_prop <- "prop. analogue BT"
name_without_MT_prop <- "prop. analogue MT"
name_SP_TP_prop <- "prop. SP TP"
name_SP_BT_prop <- "prop. SP BT"
name_SP_MT_prop <- "prop. SP MT"
name_SP_total_prop <- "prop. SP total"

# vector with excluded categories due to missing normalization factor
excluded_categories_nf <- c("Climate change: biogenic",
                            "Climate change: fossil",
                            "Climate change: land use and land use change",
                            "Ecotoxicity: freshwater, inorganics",
                            "Ecotoxicity: freshwater, organics",
                            "Human toxicity: carcinogenic, inorganics",
                            "Human toxicity: carcinogenic, organics",
                            "Human toxicity: non-carcinogenic, inorganics",
                            "Human toxicity: non-carcinogenic, organics")
# vector with excluded toxicity categories for Pareto rule
excluded_categories_pr <- c("Human toxicity: carcinogenic",
                            "Human toxicity: non-carcinogenic",
                            "Ecotoxicity: freshwater")

# storage location
location_download <- "C:/Users/Klene/Documents/Uni_Bremen/WS24_25/Masterarbei
```

```
t/R/Daten/"
location_upload <- "C:/Users/Klene/Documents/Uni_Bremen/WS24_25/Masterarbeit/
R/Results/"
# Login data for NextCloud
username <- "mkindler@uni-bremen.de" # username
password <- "xxxxxxx" # password
# WebDAV-URL to file
nextcloud_url <- "https://nc.uni-bremen.de/remote.php/dav/files/mkindler%40un
i-bremen.de/Masterarbeit/Masterarbeit_II/R/Daten/Ergebnisse_LCA_basic.xlsx"
nextcloud_url_2 <- "https://nc.uni-bremen.de/remote.php/dav/files/mkindler%40
uni-bremen.de/Masterarbeit/Masterarbeit_II/R/Results/"
```

Data preparation

Load data from NextCloud:

```
# download file
response <- GET(nextcloud_url, authenticate(username, password, type = "basic
"))
# Check successful downloading
if (status_code(response) == 200) {
  # Save file
  writeBin(content(response, "raw"), paste0(location_download, "Ergebnisse_LC
A_basic.xlsx"))
  cat("The file was successfully downloaded and saved locally.\n")
} else {
  cat("Error downloading the file. Status code: ", status_code(response), "\n
")
}
```

```
## The file was successfully downloaded and saved locally.
```

```
# Load results data from Excel file
results <- read_excel(paste0(location_download, "Ergebnisse_LCA_basic.xlsx"),
sheet = sheet)
```

```
## New names:
## • `Wirkungskategorien` -> `Wirkungskategorien...1`
## • `Einheit` -> `Einheit...2`
## • `Wirkungskategorien` -> `Wirkungskategorien...5`
## • `Einheit` -> `Einheit...6`
## • `Wirkungskategorien` -> `Wirkungskategorien...9`
## • `Einheit` -> `Einheit...10`
```

```
# Check data type: DataFrame
if (is.data.frame(results)) {
  results <- results
} else {
  results <- as.data.frame(results)
}
# Rename columns
results <- results %>%
```

```
  rename(
    `impact category` = `Wirkungskategorien...1`,
    unit = `Einheit...2`
    ) %>%
  select(-"Wirkungskategorien...5", -"Einheit...6", -"Wirkungskategorien...9"
, -"Einheit...10")

# Load normalization factors from Excel file
NF <- read_excel(paste0(location_download, "Ergebnisse_LCA_basic.xlsx"), shee
t = 1)

## New names:
## • `` -> `...2`
## • `` -> `...3`

# Check data type: DataFrame
if (is.data.frame(NF)) {
  NF <- NF
} else {
  NF <- as.data.frame(NF)
}
# Set column names right
NF <- NF[-1, ]
colnames(NF) <- as.character(NF[1, ])
NF <- NF[-1, ]
# Select relevant columns
NF <- NF %>%
  select(`impact category`,
         NF = `normalization factor\r\n(Matuštík 2024)`)
```

Calculations

Savings potential

Calculation of Savings Potentials:

```
results <- results %>%
  mutate(!!sym(name_SP_TP) := !!sym(name_without_TP) - !!sym(name_with_TP),
         !!sym(name_SP_BT) := !!sym(name_without_BT) - !!sym(name_with_BT),
         !!sym(name_SP_MT) := !!sym(name_without_MT) - !!sym(name_with_MT),
         !!sym(name_SP_total) := !!sym(name_SP_TP) + !!sym(name_SP_BT) + !!sy
m(name_SP_MT))
```

Normalization

Calculation of normalized values (to the planetary boudaries (Matuštík 2024)):

```
# Join of results and NF according to the impact categories
results <- left_join(results, NF, by = c("impact category" = "impact category
"))
# Convert all columns except the first two into numerical values
results <- results %>%
  mutate_at(vars(-1, -2), as.numeric)
```

```r
# Remove categories without normalization factors
results_norm <- results %>%
  filter(!`impact category` %in% excluded_categories_nf)

# Calculation of the normalised values by dividing the column values by the n
ormalization factors
results_norm <- results_norm %>%
  mutate(
    across(!c(`impact category`, unit, NF), ~ .x / NF, .names = "norm. {.col}
")
    )
```

Determine relevant impact categories

Calculation of proportion of total savings potential of each measure:

```r
# Removing the toxicity categories
results_prop <- results_norm %>%
  filter(!`impact category` %in% excluded_categories_pr)

# Calculate the sum of each normalized column and divide the values by this s
um
results_prop <- results_prop %>%
  mutate(
    !!sym(name_without_TP_prop) := !!sym(name_without_TP_norm) / sum(abs(!!sy
m(name_without_TP_norm)), na.rm = TRUE),
    !!sym(name_with_TP_prop) := !!sym(name_with_TP_norm) / sum(abs(!!sym(name
_with_TP_norm)), na.rm = TRUE),
    !!sym(name_SP_TP_prop) := !!sym(name_SP_TP_norm) / sum(abs(!!sym(name_SP_
TP_norm)), na.rm = TRUE),
    !!sym(name_without_BT_prop) := !!sym(name_without_BT_norm) / sum(abs(!!sy
m(name_without_BT_norm)), na.rm = TRUE),
    !!sym(name_with_BT_prop) := !!sym(name_with_BT_norm) / sum(abs(!!sym(name
_with_BT_norm)), na.rm = TRUE),
    !!sym(name_SP_BT_prop) := !!sym(name_SP_BT_norm) / sum(abs(!!sym(name_SP_
BT_norm)), na.rm = TRUE),
    !!sym(name_without_MT_prop) := !!sym(name_without_MT_norm) / sum(abs(!!sy
m(name_without_MT_norm)), na.rm = TRUE),
    !!sym(name_with_MT_prop) := !!sym(name_with_MT_norm) / sum(abs(!!sym(name
_with_MT_norm)), na.rm = TRUE),
    !!sym(name_SP_MT_prop) := !!sym(name_SP_MT_norm) / sum(abs(!!sym(name_SP_
MT_norm)), na.rm = TRUE),
    !!sym(name_SP_total_prop) := !!sym(name_SP_total_norm) / sum(abs(!!sym(na
me_SP_total_norm)), na.rm = TRUE)
  )

norm_sum <- sum(abs(results_prop[[name_SP_total_norm]]))
```

Determine relevant impact categories:

```
# Sort categories by proportion of total savings potential
results_prop <- results_prop %>%
  arrange(desc(abs(!!sym(name_SP_total_prop))))
# Calculation of the cumulative total and mark categories that contribute to
the sum of 0.8
results_prop <- results_prop %>%
  mutate(
    `cum. sum` = cumsum(abs(!!sym(name_SP_total_prop))),
    relevant = (`cum. sum` < 0.8) | lag(`cum. sum` < 0.8, default = FALSE)
  )
# Mark at least the highest three impact categories
results_prop$relevant[1:3] <- TRUE
```

Save as csv-file

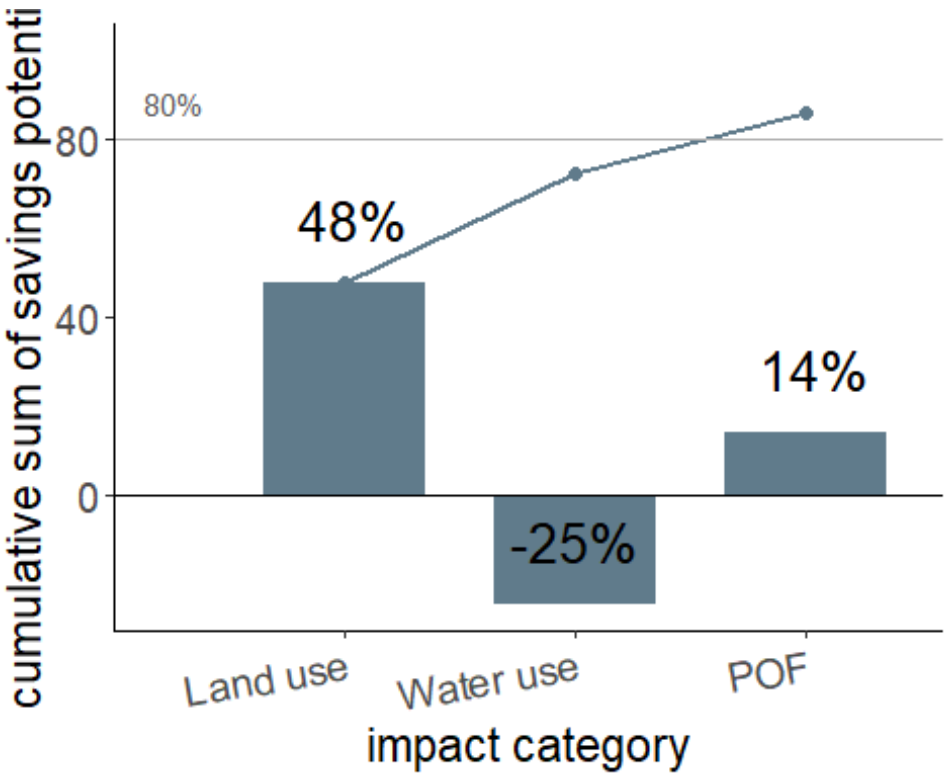
```
write.csv(results_prop, paste0(location_upload, measure, "_data.csv"), row.na
mes = FALSE)
```

Plot

Plot the relevant impact categories

```
# Ensure that the impact category retains the order in which it appears in th
e data
results_prop$`impact category` <- factor(results_prop$`impact category`, leve
ls = results_prop$`impact category`)

# Prepare data
data_plot <- results_prop %>%
  filter(relevant == TRUE) %>%
  mutate(
    across(.cols = starts_with("prop."), .fns = ~ . * 100),
    `cum. sum` = `cum. sum`*100) %>%
  select(`impact category`, name_SP_total_prop, name_SP_TP_prop, name_SP_MT_p
rop, name_SP_BT_prop, `cum. sum`)

# Rename long impact categories
data_plot <- data_plot %>%
  mutate(`impact category` = case_when(
    `impact category` == "Photochemical oxidant formation: human health" ~ "P
OF",
    `impact category` == "Eutrophication: freshwater" ~ "Euthroph. f",
    TRUE ~ `impact category`
  ))

# Ensure that the impact category retains the order in which it appears in th
e data
data_plot$`impact category` <- factor(data_plot$`impact category`, levels = d
ata_plot$`impact category`)

# Plot relvant impact categories with their proportion of the total savings p
```

```r
otential, kumulative sum
ggplot(data_plot, aes(x = `impact category`, y = !!sym(name_SP_total_prop)))
+
  geom_bar(stat = "identity", width = 0.7, fill = "lightskyblue4", show.legen
d = FALSE) +
  geom_line(aes(y = `cum. sum`, group = 1), color = "lightskyblue4", size = 1
) +
  geom_point(aes(y = `cum. sum`), color = "lightskyblue4", size = 2) +
  geom_hline(yintercept = 0, color = "black", linetype = "solid") +
  geom_hline(yintercept = 80, color = "grey70", linetype = "solid") +
  annotate("text", x = 0, y = 80, label = "80%", color = "grey35", size = 4,
hjust = -0.5, vjust = -1) +
  geom_text(aes(
    y = !!sym(name_SP_total_prop),
    label = paste0(format(round(!!sym(name_SP_total_prop), 0), nsmall = 0), "
%")
  ), vjust = -1, size = 7) +
  labs(
    x = "impact category",
    y = "cumulative sum of savings potentials"
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 10, hjust = 1, size = 15),
    axis.text.y = element_text(size = 15),
    axis.title.x = element_text(size = 17),
    axis.title.y = element_text(size = 17)
  ) +
  coord_cartesian(ylim = c(NA, 100))
```

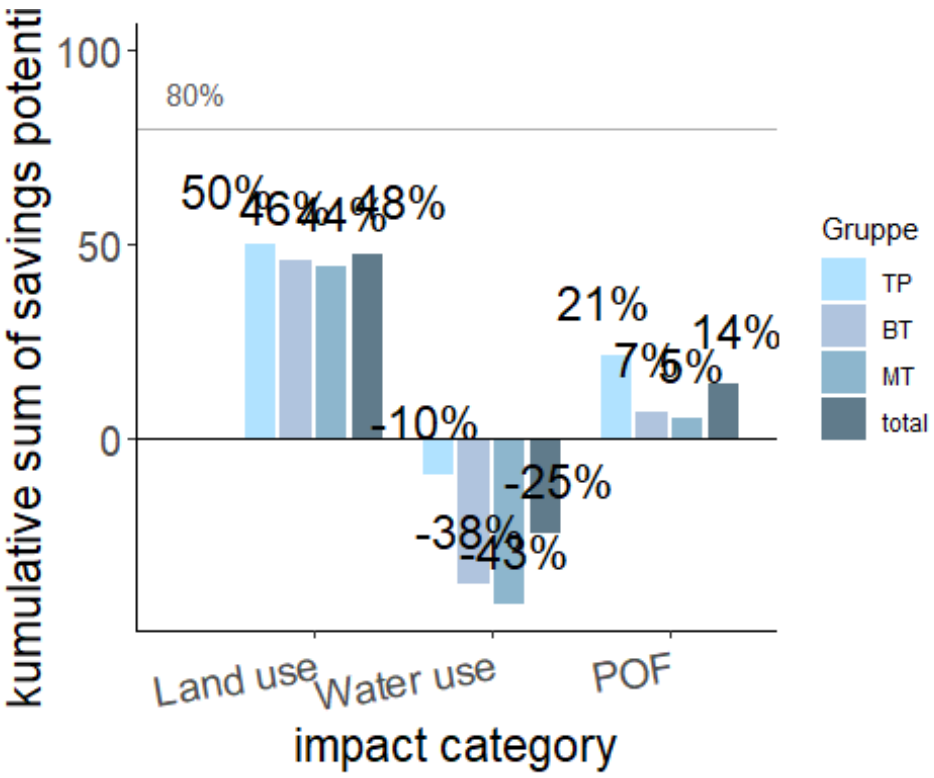



```
data_plot1 <- data_plot

# Save plot
ggsave(paste0(location_upload, measure, "_relevant_SP_prop.png"), dpi = 600)

## Saving 5 x 4 in image
```

Plot considering submeasures of measure 1:

```
# Conversion of data into long format
data_plot1 <- data_plot %>%
  select(`impact category`, !!sym(name_SP_TP_prop), !!sym(name_SP_BT_prop), !
!sym(name_SP_MT_prop), !!sym(name_SP_total_prop), `cum. sum`) %>%
  pivot_longer(
    cols = starts_with("prop. SP"),
    names_to = "category",
    values_to = "savings potential"
  ) %>%
  mutate(
    Gruppe = case_when(
      grepl("TP", category) ~ "TP",
      grepl("BT", category) ~ "BT",
      grepl("MT", category) ~ "MT",
      grepl("total", category) ~ "total"
    )
  )

# Ensure that 'impact category' is a factor and that the sequence is defined
```

```r
data_plot1$Gruppe <- factor(data_plot1$Gruppe, levels = c("TP", "BT", "MT", "
total"))

ggplot(data_plot1, aes(x = `impact category`, y = `savings potential`, fill =
Gruppe)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.8), width =
0.7) +
  scale_fill_manual(values = c("TP" = "lightskyblue1", "BT" = "lightsteelblue
",
                               "MT" = "lightskyblue3", "total" = "lightskyblu
e4")) +
  geom_hline(yintercept = 80, color = "grey70", linetype = "solid") +
  geom_hline(yintercept = 0, color = "black", linetype = "solid") +
  annotate("text", x = 0, y = 80, label = "80%", color = "grey35", size = 4,
           hjust = -0.5, vjust = -1) +
  geom_text(data = subset(data_plot1, `impact category` != "Land use"),
            aes(y = `savings potential`, label = paste0(round(`savings potent
ial`, 0), "%")),
            position = position_dodge(width = 1),
            vjust = -1, size = 6) +
  geom_text(data = subset(data_plot1, `impact category` == "Land use"),
            aes(y = `savings potential`, label = paste0(round(`savings potent
ial`, 0), "%")),
            position = position_dodge(width = 1.3),
            vjust = -1, size = 6) +
  labs(
    x = "impact category",
    y = "kumulative sum of savings potentials"
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 10, hjust = 1, size = 15),
    axis.text.y = element_text(size = 15),
    axis.title.x = element_text(size = 17),
    axis.title.y = element_text(size = 17)
  ) +
  coord_cartesian(ylim = c(NA, 100))
```

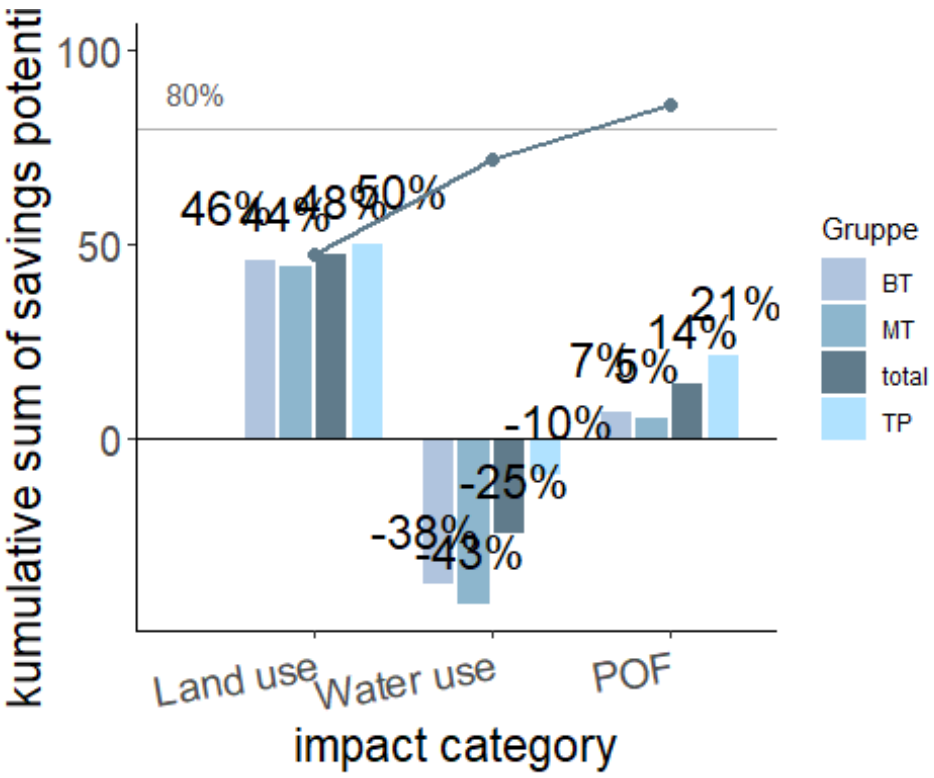



```
# Save as PNG
ggsave(paste0(location_upload, "01_relevant_SPs_complex.png"), dpi = 600)

## Saving 5 x 4 in image
```

Versuch???

```
# Conversion of data into long format
data_plot2 <- data_plot %>%
  select(`impact category`, !!sym(name_SP_TP_prop), !!sym(name_SP_BT_prop), !
!sym(name_SP_MT_prop), !!sym(name_SP_total_prop), `cum. sum`) %>%
  pivot_longer(
    cols = starts_with("prop. SP"),
    names_to = "category",
    values_to = "savings potential"
  ) %>%
  mutate(
    Gruppe = case_when(
      grepl("TP", category) ~ "TP",
      grepl("BT", category) ~ "BT",
      grepl("MT", category) ~ "MT",
      grepl("total", category) ~ "total"
    )
  )

# Berechne cumsum nur für "total"
data_plot2_total <- data_plot2 %>%
  filter(Gruppe == "total") %>%
```

```r
  arrange(`impact category`) %>%
  mutate(cumsum_total = cumsum(abs(`savings potential`)))

# Merge back the cumsum for total into the original dataset
data_plot2 <- data_plot2 %>%
  left_join(data_plot2_total %>% select(`impact category`, cumsum_total), by
= "impact category")

# Erstellen des Plots
ggplot(data_plot2, aes(x = `impact category`, y = `savings potential`, fill =
Gruppe)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.8), width =
0.7) +
  scale_fill_manual(values = c("TP" = "lightskyblue1", "BT" = "lightsteelblue
",
                               "MT" = "lightskyblue3", "total" = "lightskyblu
e4")) +
  geom_hline(yintercept = 80, color = "grey70", linetype = "solid") +
  geom_hline(yintercept = 0, color = "black", linetype = "solid") +
  annotate("text", x = 0, y = 80, label = "80%", color = "grey35", size = 4,
           hjust = -0.5, vjust = -1) +
  geom_text(data = subset(data_plot2, `impact category` != "Land use"),
            aes(y = `savings potential`, label = paste0(round(`savings potent
ial`, 0), "%")),
            position = position_dodge(width = 1),
            vjust = -1, size = 6) +
  geom_text(data = subset(data_plot2, `impact category` == "Land use"),
            aes(y = `savings potential`, label = paste0(round(`savings potent
ial`, 0), "%")),
            position = position_dodge(width = 1.3),
            vjust = -1, size = 6) +
  # Hinzufügen der kumulierten Summe für "total" als Linie
  geom_line(data = data_plot2_total, aes(x = `impact category`, y = cumsum_to
tal),
            color = "lightskyblue4", size = 1, group = 1) +
  geom_point(data = data_plot2_total, aes(x = `impact category`, y = cumsum_t
otal),
             color = "lightskyblue4", size = 2) +
  labs(
    x = "impact category",
    y = "kumulative sum of savings potentials"
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 10, hjust = 1, size = 15),
    axis.text.y = element_text(size = 15),
    axis.title.x = element_text(size = 17),
    axis.title.y = element_text(size = 17)
  ) +
  coord_cartesian(ylim = c(NA, 100))
```



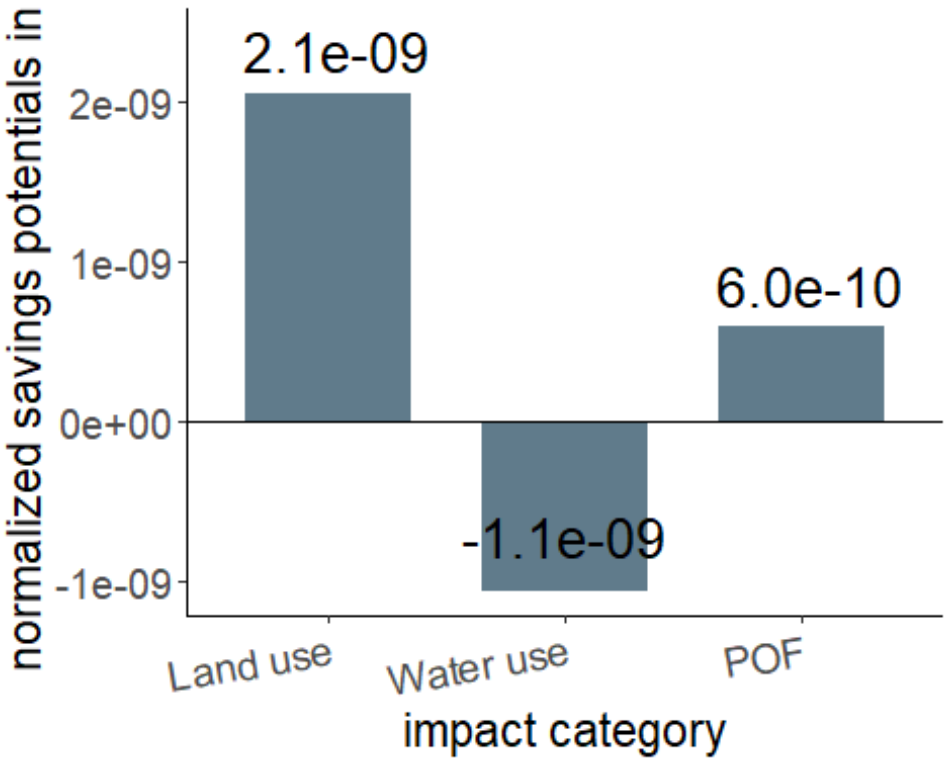
Plot relevant categories with their norm. savings potential:

```
# Ensure that the impact category retains the order in which it appears in th
e data
results_norm$`impact category` <- factor(results_norm$`impact category`, leve
ls = results_norm$`impact category`)

results_relevant <- results_prop %>%
  select(`impact category`, relevant)

results_norm <- left_join(results_norm, results_relevant, by = c("impact cate
gory" = "impact category"))

# Prepare data
data_plot <- results_norm %>%
  filter(relevant == TRUE) %>%
  select(`impact category`, name_SP_total_norm) %>%
  arrange(desc(abs(!!sym(name_SP_total_norm))))
```

```
## Warning: Using an external vector in selections was deprecated in tidysele
ct 1.1.0.
## ℹ Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(name_SP_total_norm)
##
##   # Now:
##   data %>% select(all_of(name_SP_total_norm))
##
```

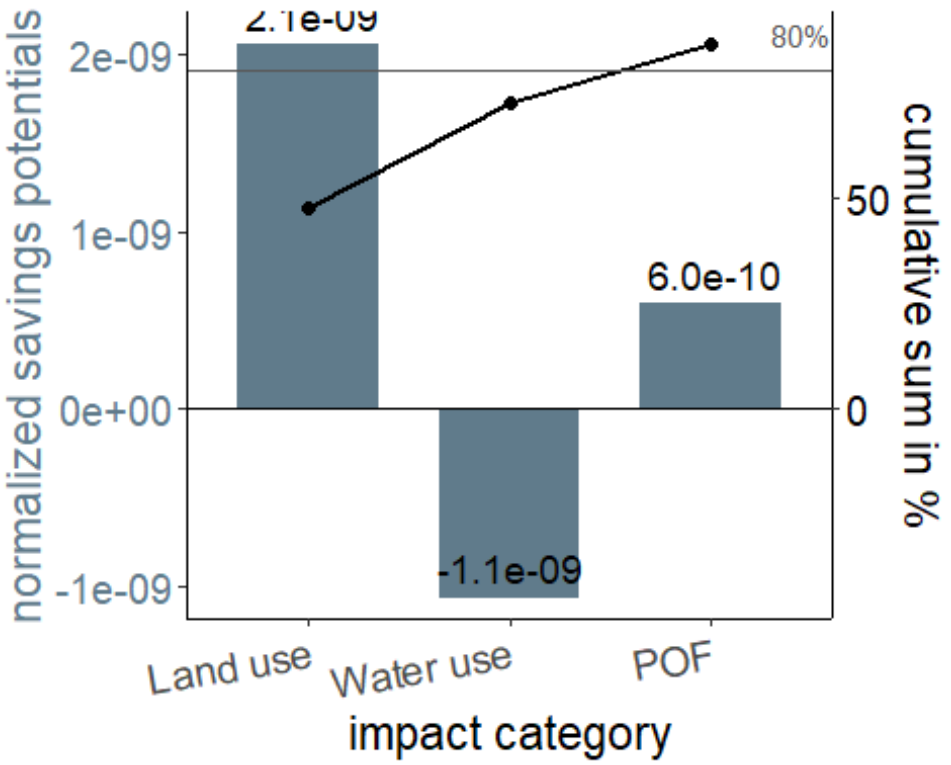
```
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# Rename long impact categories
data_plot <- data_plot %>%
  mutate(`impact category` = case_when(
    `impact category` == "Photochemical oxidant formation: human health" ~ "P
OF",
    `impact category` == "Eutrophication: freshwater" ~ "Euthroph. f",
    TRUE ~ `impact category`
  ))

# Ensure that the impact category retains the order in which it appears in th
e data
data_plot$`impact category` <- factor(data_plot$`impact category`, levels = d
ata_plot$`impact category`)

ggplot(data_plot, aes(x = `impact category`, y = !!sym(name_SP_total_norm)))
+
  geom_bar(stat = "identity", width = 0.7, fill = "lightskyblue4", show.legen
d = FALSE) +
  geom_hline(yintercept = 0, color = "black", linetype = "solid") +
  geom_text(aes(
    y = !!sym(name_SP_total_norm),
    label = format(!!sym(name_SP_total_norm), scientific = TRUE, digits = 2))
,
    vjust = ifelse(data_plot$`impact category` != "Water use", -0.5, -0.8),
    size = 7) +
  labs(
    x = "impact category",
    y = "normalized savings potentials in %"
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 10, hjust = 1, size = 15),
    axis.text.y = element_text(size = 15),
    axis.title.x = element_text(size = 17),
    axis.title.y = element_text(size = 17)
  ) +
  scale_y_continuous(expand = expansion(mult = c(0.05, 0.17)))
```




```r
# Save plot
ggsave(paste0(location_upload, measure, "_relevant_SP_norm.png"), dpi = 600)

## Saving 5 x 4 in image

# create cumsum and arrange descending
data_plot <- data_plot %>%
  arrange(desc(abs(!!sym(name_SP_total_norm)))) %>%
  mutate(
    cumsum = cumsum(abs(data_plot$`norm. SP total`/norm_sum))*100
  )

# max. values for scaling
max_norm <- max(data_plot[[name_SP_total_norm]])
max_cumsum <- max(data_plot$cumsum)

# Conversion factor
factor <- max_norm / max_cumsum

ggplot(data_plot, aes(x = `impact category`)) +
  geom_bar(aes(y = !!sym(name_SP_total_norm)), stat = "identity",
           width = 0.7, fill = "lightskyblue4", show.legend = FALSE) +
  geom_line(aes(y = cumsum * factor, group = 1), color = "black", size = 1) +
  geom_point(aes(y = cumsum * factor), color = "black", size = 2) +
  geom_hline(yintercept = 0, color = "black", linetype = "solid") +
  geom_hline(yintercept = 80 * factor, color = "grey35", linetype = "solid")
+
  annotate("text", x = 3.3, y = 80 * factor, label = "80%", color = "grey35",
```

```
size = 4, hjust = 0, vjust = -1) +
  geom_text(aes(
    y = !!sym(name_SP_total_norm),
    label = format(!!sym(name_SP_total_norm), scientific = TRUE, digits = 2))
,
    vjust = -0.5, size = 5) +
  scale_y_continuous(
    name = "normalized savings potentials",
    sec.axis = sec_axis(~ . / factor, name = "cumulative sum in %"),
  expand = expansion(mult = c(0.04, 0.06))
  ) +
  labs(x = "impact category") +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 10, hjust = 1, size = 15),
    axis.text.y.left = element_text(color = "lightskyblue4", size = 15),
    axis.text.y.right = element_text(color = "black", size = 15),
    axis.title.x = element_text(size = 17),
    axis.title.y.left = element_text(size = 17, color = "lightskyblue4"),
    axis.title.y.right = element_text(size = 17, color = "black")
  )
```




```
# Save plot
ggsave(paste0(location_upload, measure, "_relevant_SP_norm_allin1.png"), dpi
= 600)

## Saving 5 x 4 in image
```

Tables

Prepare data:

```
# Select relevant columns
results_prop <- results_prop %>%
  select(`impact category`, unit,
         !!sym(name_without_TP_prop), !!sym(name_without_BT_prop), !!sym(name
_without_MT_prop),
         !!sym(name_with_TP_prop), !!sym(name_with_BT_prop), !!sym(name_with_
MT_prop),
         !!sym(name_SP_TP_prop), !!sym(name_SP_BT_prop), !!sym(name_SP_MT_pro
p), !!sym(name_SP_total_prop))
results_norm <- results_norm %>%
  select(`impact category`, unit,
         !!sym(name_without_TP_norm), !!sym(name_without_BT_norm), !!sym(name
_without_MT_norm),
         !!sym(name_with_TP_norm), !!sym(name_with_BT_norm), !!sym(name_with_
MT_norm),
         !!sym(name_SP_TP_norm), !!sym(name_SP_BT_norm), !!sym(name_SP_MT_nor
m), !!sym(name_SP_total_norm))
# Join of 'results' and 'results_norm' according to the impact categories
results <- left_join(results, results_norm, by = c("impact category" = "impac
t category"))
results <- results %>%
  select(-unit.y) %>%
  rename(unit = unit.x)
# Join of 'results' and 'results_prop' according to the impact categories
results <- left_join(results, results_prop, by = c("impact category" = "impac
t category"))
results <- results %>%
  select(-unit.y) %>%
  rename(unit = unit.x)
# Sort descending according to normalized savings potential
results <- results %>%
  arrange(desc(abs(!!sym(name_SP_total_norm))))
```

Create tables:

```
# Only results
table_results <- results %>%
  select("impact category", "unit",
         !!sym(name_without_TP), !!sym(name_without_BT), !!sym(name_without_M
T),
         !!sym(name_with_TP), !!sym(name_with_BT), !!sym(name_with_MT),
         !!sym(name_SP_TP), !!sym(name_SP_BT), !!sym(name_SP_MT), !!sym(name_
SP_total)) %>%
  arrange(desc(abs(!!sym(name_SP_total)))) %>%
  mutate(
    across(-all_of(c("impact category", "unit")), ~ formatC(.x, format = "e",
digits = 2)))
# Create flextable
```

```r
ft <- flextable(table_results) %>%
  theme_vanilla() %>%
  autofit() %>%
  bold(part = "header") %>%
  border_outer() %>%
  border_inner_h() %>%
  border_inner_v()
# Save as PNG
tf <- tempfile(fileext = ".png")
save_as_image(x = ft, path = paste0(location_upload, measure, "_results", ".p
ng"))
```

```
## [1] "C:/Users/Klene/Documents/Uni_Bremen/WS24_25/Masterarbeit/R/Results/01
_results.png"
```

```r
# Saving potentials
table_results <- results %>%
  select(`impact category`, unit, !!sym(name_SP_total), !!sym(name_SP_total_n
orm),  !!sym(name_SP_total_prop)) %>%
  arrange(desc(abs(!!sym(name_SP_total_norm)))) %>%
  mutate(
    across(-all_of(c("impact category", "unit")), ~ formatC(.x, format = "e",
digits = 2)))
# Create flextable
ft <- flextable(table_results) %>%
  theme_vanilla() %>%
  autofit() %>%
  bold(part = "header") %>%
  border_outer() %>%
  border_inner_h() %>%
  border_inner_v()
# Save as PNG
tf <- tempfile(fileext = ".png")
save_as_image(x = ft, path = paste0(location_upload, measure, "_results_SPs",
".png"))
```

```
## [1] "C:/Users/Klene/Documents/Uni_Bremen/WS24_25/Masterarbeit/R/Results/01
_results_SPs.png"
```

```r
# Normalized results
table_results <- results %>%
  select("impact category", "unit",
         !!sym(name_without_TP_norm), !!sym(name_without_BT_norm), !!sym(name
_without_MT_norm),
         !!sym(name_with_TP_norm), !!sym(name_with_BT_norm), !!sym(name_with_
MT_norm),
         !!sym(name_SP_TP_norm), !!sym(name_SP_BT_norm), !!sym(name_SP_MT_nor
m), !!sym(name_SP_total_norm)) %>%
  arrange(desc(abs(!!sym(name_SP_total_norm)))) %>%
  mutate(
    across(-all_of(c("impact category", "unit")), ~ formatC(.x, format = "e",
digits = 2)))
```

```r
# Removing the categories without normalization factors
table_results <- table_results %>%
  filter(!`impact category` %in% excluded_categories_nf)
# Rename columns
table_results <- table_results %>%
  rename(!!sym(name_without_TP) := !!sym(name_without_TP_norm),
         !!sym(name_without_BT) := !!sym(name_without_BT_norm),
         !!sym(name_without_MT) := !!sym(name_without_MT_norm),
          !!sym(name_with_TP) := !!sym(name_with_TP_norm),
         !!sym(name_with_BT) := !!sym(name_with_BT_norm),
         !!sym(name_with_MT) := !!sym(name_with_MT_norm),
           !!sym(name_SP_TP) := !!sym(name_SP_TP_norm),
         !!sym(name_SP_BT) := !!sym(name_SP_BT_norm),
         !!sym(name_SP_MT) := !!sym(name_SP_MT_norm),
         !!sym(name_SP_total) := !!sym(name_SP_total_norm)
           )
# Create flextable
ft <- flextable(table_results) %>%
  theme_vanilla() %>%
  autofit() %>%
  bold(part = "header") %>%
  border_outer() %>%
  border_inner_h() %>%
  border_inner_v()
# Save as PNG
tf <- tempfile(fileext = ".png")
save_as_image(x = ft, path = paste0(location_upload, measure, "_results_norm"
, ".png"))
```

```
## [1] "C:/Users/Klene/Documents/Uni_Bremen/WS24_25/Masterarbeit/R/Results/01
_results_norm.png"
```

```r
# Proportion data
table_results <- results %>%
  select("impact category", "unit",
         !!sym(name_without_TP_prop), !!sym(name_without_BT_prop), !!sym(name
_without_MT_prop),
         !!sym(name_with_TP_prop), !!sym(name_with_BT_prop), !!sym(name_with_
MT_prop),
         !!sym(name_SP_TP_prop), !!sym(name_SP_BT_prop), !!sym(name_SP_MT_pro
p), !!sym(name_SP_total_prop)) %>%
  arrange(desc(abs(!!sym(name_SP_total_prop)))) %>%
  mutate(
    across(-all_of(c("impact category", "unit")), ~ formatC(.x, format = "e",
digits = 2)))
# Removing the categories without normalization factor
table_results <- table_results %>%
  filter(!`impact category` %in% excluded_categories_nf)
# Removing the toxicity categories
table_results <- table_results %>%
  filter(!`impact category` %in% excluded_categories_pr)
# Rename columns
```

```r
table_results <- table_results %>%
  rename(!!sym(name_without_TP) := !!sym(name_without_TP_prop),
         !!sym(name_without_BT) := !!sym(name_without_BT_prop),
         !!sym(name_without_MT) := !!sym(name_without_MT_prop),
          !!sym(name_with_TP) := !!sym(name_with_TP_prop),
         !!sym(name_with_BT) := !!sym(name_with_BT_prop),
         !!sym(name_with_MT) := !!sym(name_with_MT_prop),
           !!sym(name_SP_TP) := !!sym(name_SP_TP_prop),
         !!sym(name_SP_BT) := !!sym(name_SP_BT_prop),
         !!sym(name_SP_MT) := !!sym(name_SP_MT_prop),
         !!sym(name_SP_total) := !!sym(name_SP_total_prop)
           )
# Create flextable
ft <- flextable(table_results) %>%
  theme_vanilla() %>%
  autofit() %>%
  bold(part = "header") %>%
  border_outer() %>%
  border_inner_h() %>%
  border_inner_v()
# Save as PNG
tf <- tempfile(fileext = ".png")
save_as_image(x = ft, path = paste0(location_upload, measure, "_results_prop"
, ".png"))
```

```
## [1] "C:/Users/Klene/Documents/Uni_Bremen/WS24_25/Masterarbeit/R/Results/01
_results_prop.png"
```

Upload files

```r
# # List of Excel files to be uploaded
# files_to_upload <- list.files(path = location_upload, pattern = "\\.xlsx$",
full.names = TRUE)
# # Loop over all files and upload
# for (file_path in files_to_upload) {
#   # Creating the file name
#   file_name <- basename(file_path)
#   # Upload file
#   response <- PUT(
#     url = paste0(nextcloud_url_2, file_name),
#     authenticate(username, password),
#     body = upload_file(file_path)
#   )
#   # Check upload status
#   if (status_code(response) == 201) {
#     print(paste(file_name, "was successfully uploaded."))
#   } else if (status_code(response) == 204) {
#     print(paste(file_name, "was successfully replaced (no content returned)
."))
#   } else {
#     print(paste("Error uploading", file_name, ". Status-Code:", status_code
(response)))
```

```
#   }
# }
#
# # List of PNG files to be uploaded
# files_to_upload <- list.files(path = location_upload, pattern = "\\.png$",
full.names = TRUE)
# # Loop over all files and upload
# for (file_path in files_to_upload) {
#   # Create the file name
#   file_name <- basename(file_path)
#   # Upload file
#   response <- PUT(
#     url = paste0(nextcloud_url_2, file_name),
#     authenticate(username, password),
#     body = upload_file(file_path)
#   )
#   # Check upload status
#   if (status_code(response) == 201) {
#     print(paste(file_name, "was successfully uploaded."))
#   } else if (status_code(response) == 204) {
#     print(paste(file_name, "was successfully replaced (no content returned)
."))
#   } else {
#     print(paste("Error uploading", file_name, ". Status-Code:", status_code
(response)))
#   }
# }
```