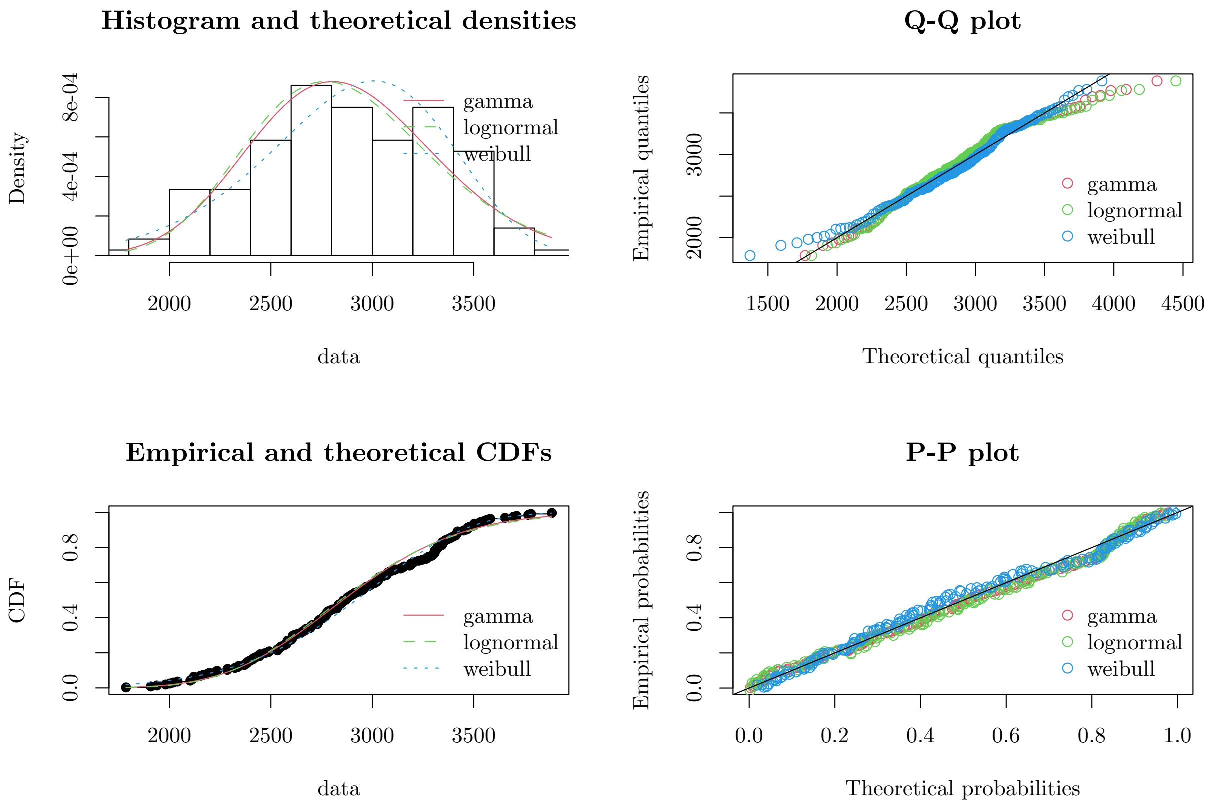
The Water Crisis by the Global Commission on the Economics of Water: A Totalising Narrative Built on Shaky Numbers

R code

Arnald Puy

```r
# PRELIMINARY ---------------------------------------------------------------

sensobol::load_packages(c("sensobol", "tidyverse", "data.table", "cowplot",
                          "scales", "rvest", "janitor", "fitdistrplus", "wesanderson"))
theme_AP <- function() {
  theme_bw() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          legend.background = element_rect(fill = "transparent",
                                           color = NA),
          legend.margin=margin(0, 0, 0, 0),
          legend.box.margin=margin(-5,-5,-5,-5),
          legend.key = element_rect(fill = "transparent",
                                    color = NA),
          strip.background = element_rect(fill = "white"),
          axis.title = element_text(size = 9),
          legend.text = element_text(size = 9),
          legend.title = element_text(size = 9),
          legend.key.width = unit(0.3, "cm"),
          legend.key.height = unit(0.3, "cm"))
}

## ----calculations, warning=FALSE------------------------------------------------------------

# SETTINGS ######################################################################

# Values used in the paper
precipitation_estimate <- 120000
precipitation_min <- precipitation_estimate - (precipitation_estimate * 0.1)
precipitation_max <- precipitation_estimate + (precipitation_estimate * 0.1)

land_runoff_estimate <- 46000
land_runoff_min <- land_runoff_estimate - (land_runoff_estimate  * 0.1)
land_runoff_max <- land_runoff_estimate + (land_runoff_estimate  * 0.1)

# RETRIEVE DATA FOR KILOCALORIES ##############################################

food.fraction <- fread("daily-per-capita-caloric-supply.csv")
old_colnames <- colnames(food.fraction)
new_colnames <- c("entity", "code", "year", "kcal")
setnames(food.fraction, old_colnames, new_colnames)
food.fraction <- food.fraction[year == 2018]
food.fraction <- food.fraction[!entity %in% c("High-income countries", "Low-income countries",
                                              "Lower-middle-income countries", "South America"
                                              "Upper-middle-income countries")]

# Check best distribution -----------------------------------------------------
```

```r
fg <- fitdist(food.fraction$kcal, "gamma")
fln <- fitdist(food.fraction$kcal, "lnorm")
fw <- fitdist(food.fraction$kcal, "weibull")

# Plot goodness of fit ---------------------------------------------------------
par(mfrow = c(2, 2))
plot.legend <- c("gamma", "lognormal","weibull")
denscomp(list(fg, fln, fw), legendtext = plot.legend)
qqcomp(list(fg, fln, fw), legendtext = plot.legend)
cdfcomp(list(fg, fln, fw), legendtext = plot.legend)
ppcomp(list(fg, fln, fw), legendtext = plot.legend)
```

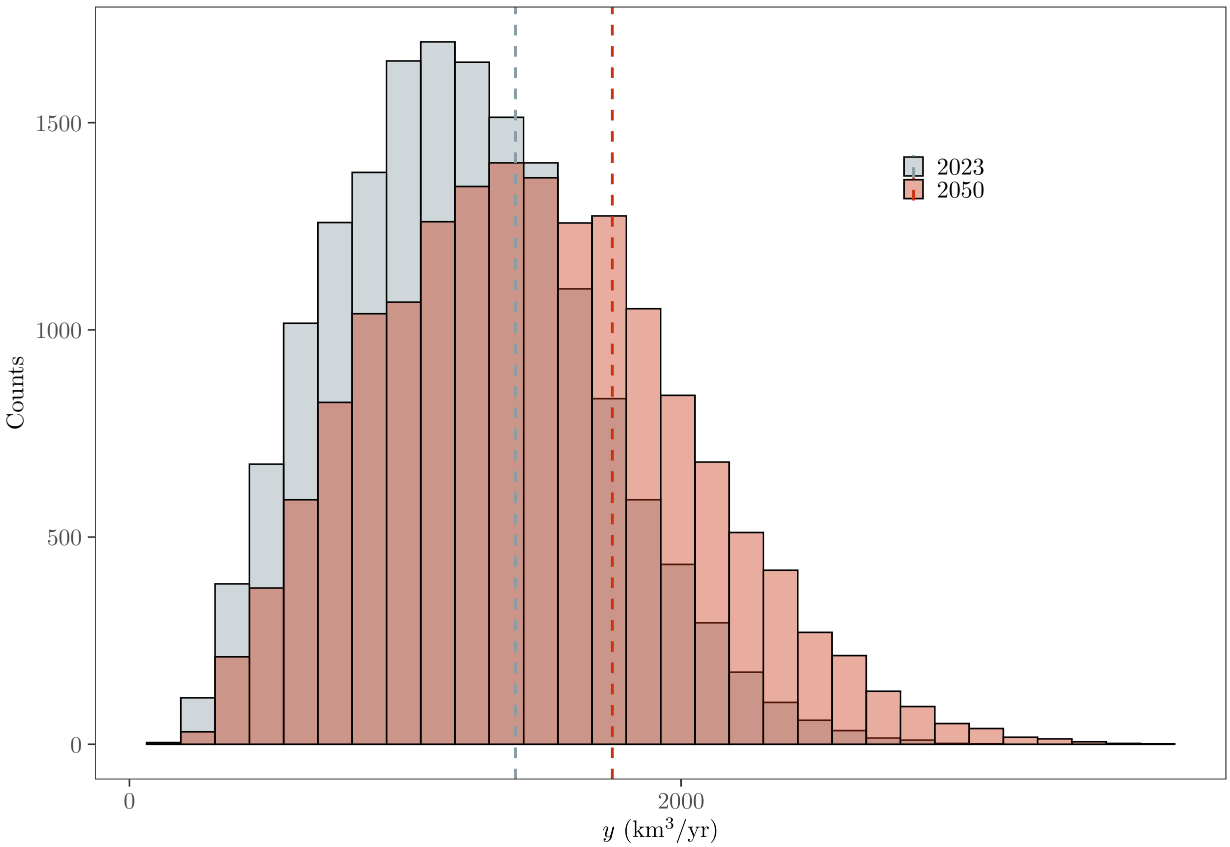



```r
# Opt for truncated weibull ----------------------------------------------------
shape <- fw$estimate[[1]]
scale <- fw$estimate[[2]]
minimum <- min(food.fraction$kcal)
maximum <- max(food.fraction$kcal)
weibull_dist <- sapply(c(minimum, maximum), function(x)
  pweibull(x, shape = shape, scale = scale))

# SAMPLE MATRIX ################################################################

N <- 2^13
params <- c("precipitation", "et_crops", "et_vegetation", "global_consumption",
```

```r
            "planetary_boundary", "W_g", "W_i", "F_i", "F_u", "$k$", "F_b",
            "$F_m$", "$F_{m_w}$", "$F_{v_w}$")

mat <- sobol_matrices(N = N, params = params)

# Uncertain parameters in Figure 1 ---------------------------------------------
mat[, "precipitation"] <- qunif(mat[, "precipitation"], precipitation_min, precipitation_max)
mat[, "et_crops"] <- qunif(mat[, "et_crops"], 5200, 5800)
mat[, "et_vegetation"] <- qunif(mat[, "et_vegetation"], 68200, 68800)
mat[, "global_consumption"] <- qunif(mat[, "global_consumption"], 3391, 5349)
mat[, "planetary_boundary"] <- qunif(mat[, "planetary_boundary"], 4000, 6000)

# Uncertain parameters for water exceedance in 2023 ----------------------------

# Estimates groundwater consumption
mat[, "W_g"] <- qunif(mat[, "W_g"], 84, 304)

# estimates irrigation water consumption
mat[, "W_i"] <- qunif(mat[, "W_i"], 1083, 1550)

# fraction of irrigation over total water consumption
mat[, "F_i"] <- qunif(mat[, "F_i"], 0.57, 0.71)

# fraction of unsustainable irrigation
mat[, "F_u"] <- qunif(mat[, "F_u"], 0.1, 0.34)

# Uncertain parameters for water exceedance in 2025 ----------------------------
mat[, "$k$"] <- qunif(mat[, "$k$"], weibull_dist[[1]], weibull_dist[[2]])

# kilocalories
mat[, "$k$"] <- qweibull(mat[, "$k$"], shape, scale)

# Fraction of blue water
mat[, "F_b"] <- qunif(mat[, "F_b"], 0.13, 0.15)

# Fraction of diet based on meat
mat[, "$F_m$"] <- qunif(mat[, "$F_m$"], 0.01, 0.35)

# Cubic meters needed to produce 1000 kcal of meat
mat[, "$F_{m_w}$"] <- qunif(mat[, "$F_{m_w}$"], 1.08, 3.8)

# Cubic meters needed to produce 1000 kcal of vegetables
mat[, "$F_{v_w}$"] <- qunif(mat[, "$F_{v_w}$"], 0.16, 1.25)

F_v <- 1 - mat[, "$F_m$"] # Fraction of diet based on vegetables
mat <- cbind(mat, F_v)
```

```r
# DEFINE MODELS ###############################################################

fun_exceedance_2023 <- function(mat) mat[, "W_g"] + mat[, "F_i"] * mat[, "W_i"] * mat[, "F_u"]


projection_fun <- function(mat, P) {

  W <- 365 * (mat[, "$k$"] * mat[, "$F_m$"] * mat[, "$F_{m_w}$"] +
                mat[, "$k$"] * mat[, "F_v"] * mat[, "$F_{v_w}$"]) / 1000

  y <- P * mat[, "F_b"] * W

  out <- list(W, y)
  names(out) <- c("W", "y")

  return(out)
}

# RUN MODELS ##################################################################

land_runoff <- mat[, "precipitation"] - mat[, "et_vegetation"] - mat[, "et_crops"]

y_2023 <- fun_exceedance_2023(mat)
exceedance_2023 <-  -1 * (mat[, "planetary_boundary"] - mat[, "global_consumption"])

population <- c(8, 9.7)
y <- lapply(population, function(P) projection_fun(mat = mat, P = P))

# ARRANGE DATA ################################################################

tmp <- lapply(y, function(x) data.table(do.call(cbind, x)))
names(tmp) <- c(2023, 2050)
dt.projections <- rbindlist(tmp, idcol = "year")

# UA / SA OF PROJECTIONS 2023 AND 2050 #########################################

dt.projections.ua <- dt.projections[, .SD[1:(2 * N)], year]

# Stats -----------------------------------------------------------------------

melt(dt.projections.ua, measure.vars = c("W", "y")) %>%
  .[, .(min = min(value),
        max = max(value),
        median = median(value)), .(year, variable)]

# Plots -----------------------------------------------------------------------
```

```r
hist.w <- dt.projections.ua[year == 2023] %>%
  ggplot(., aes(W)) +
  geom_histogram(fill = "grey", color = "black") +
  theme_AP() +
  geom_vline(xintercept = 1300, lty = 2, linewidth = 1) +
  labs(x = "$W$ (m$^3$/yr)", y = "Counts") +
  scale_x_continuous(breaks = pretty_breaks(n = 3))

dt.year <- data.table(year = c(2023, 2050), value = c(1400, 1750))
dt.year[, year:= as.factor(year)]

selected_wesanderson <- "Royal1"

hist.y <- dt.projections.ua %>%
  ggplot(., aes(y, fill = year)) +
  geom_histogram(colour = "black", alpha = 0.4, position="identity") +
  labs(x = "$y$ (km$^3$/yr)", y = "Counts") +
  theme_AP() +
  scale_x_continuous(breaks = pretty_breaks(n = 2)) +
  geom_vline(data = dt.year, aes(xintercept = value, color = year, group = year),
             linetype = 2, linewidth = 1) +
  scale_fill_manual(values = wes_palette(name = selected_wesanderson, 2),
                    name = "") +
  scale_color_manual(values = wes_palette(name = selected_wesanderson, 2),
                    name = "") +
  theme(legend.position = c(0.75, 0.8))

hist.y
```

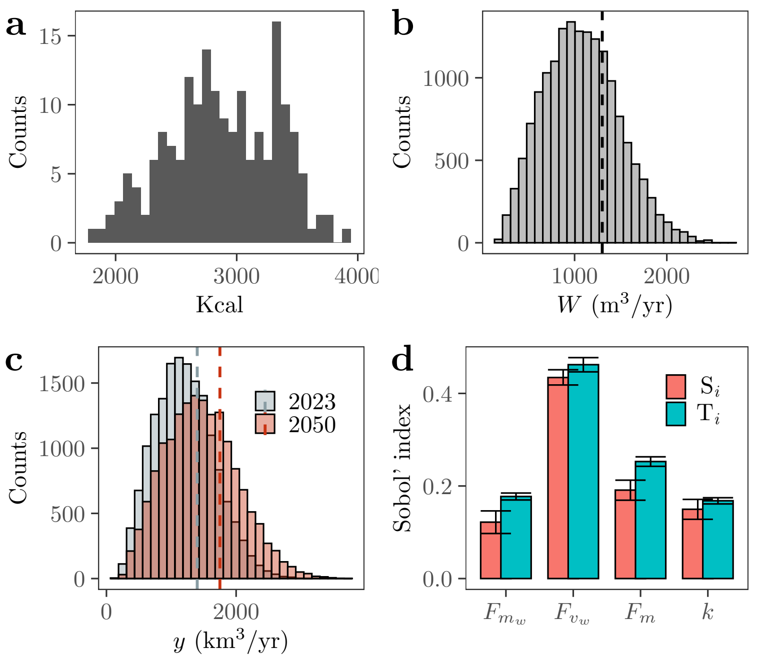



```r
# Sensitivity analysis ---------------------------------------------------------

ind <- dt.projections[year == 2023] %>%
  .[, sobol_indices(Y = y, params = params, N = N, boot = TRUE, R = 10^3,
        first = "jansen", total = "jansen")]

plot.ind <- ind$results[parameters %in% params[10:14]] %>%
  .[!parameters == "F_b"] %>%
  ggplot(., aes(parameters, original, fill = sensitivity)) +
  geom_bar(stat = "identity", position = position_dodge(0.6), color = "black") +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  labs(x = "", y = "Sobol' index") +
  geom_errorbar(aes(ymin = low.ci, ymax = high.ci),
                position = position_dodge(0.6)) +
  scale_fill_discrete(name = "",
                              labels = c(expression(S[italic(i)]),
                                         expression(T[italic(i)]))) +
  theme_AP() +
  theme(legend.position = c(0.8, 0.85))

# PLOT DISTRIBUTION OF CALORIC SUPPLY ##########################################

food.fraction <- fread("daily-per-capita-caloric-supply.csv")
old_colnames <- colnames(food.fraction)
```

```r
new_colnames <- c("entity", "code", "year", "kcal")
setnames(food.fraction, old_colnames, new_colnames)
plot.caloric <- food.fraction[year == 2018] %>%
  ggplot(., aes(kcal)) +
  geom_histogram() +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  labs(x = "Kcal", y = "Counts") +
  theme_AP()
```

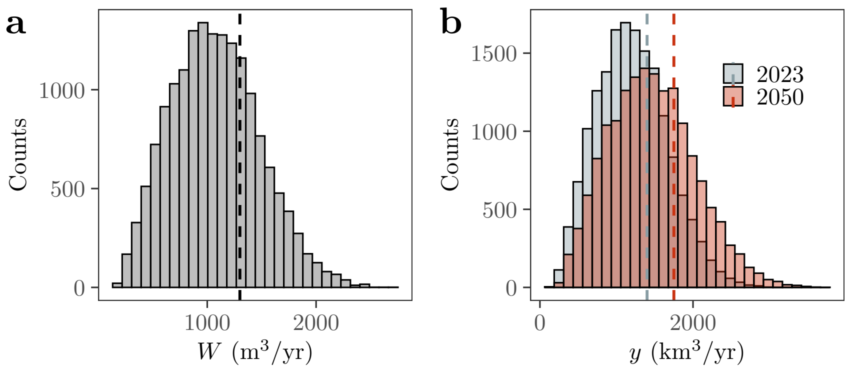
```r
# MERGE PLOTS ################################################################

plot_grid(plot.caloric, hist.w, hist.y, plot.ind, ncol = 2, labels = "auto")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



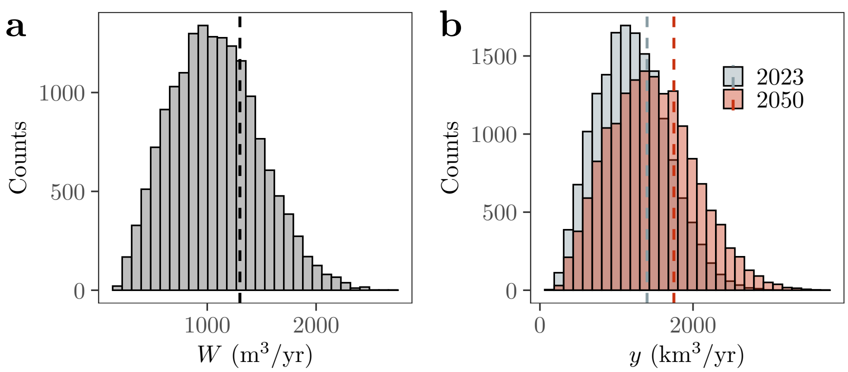

```r
###########

plot_grid(hist.w, hist.y, ncol = 2, labels = "auto")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
##########

plot_grid(hist.w, hist.y, ncol = 2, labels = "auto")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# ARRANGE DATA #################################################################

tmp <- split(dt.projections, dt.projections$year) %>%
  lapply(., function(x) x[, year:= NULL])

years <- c(2023, 2050)

out <- list()
for(i in 1:length(tmp)) {

  out[[i]] <- setnames(tmp[[i]], colnames(tmp[[i]]),
                       paste(colnames(tmp[[i]]), years[[i]], sep = "."))


}

dt <- do.call(cbind, out) %>%
  cbind(land_runoff, exceedance_2023, .) %>%
  .[1:(2 * N)] %>%
  .[, outside:= ifelse(land_runoff < land_runoff_min |
```

```r
                         land_runoff > land_runoff_max, "Yes", "No")] %>%
  .[, accessible_water_runoff:= land_runoff - 7800 - 20400] %>%
  .[, outside_runoff:= ifelse(accessible_water_runoff < 12500 |
                                accessible_water_runoff > 18500, "Yes", "No")] %>%
  .[, water_deficit:= ifelse(exceedance_2023 > 0, "Yes", "No")]

dt <- dt[, additional_water_2050:= y.2050 - y.2023] %>%
  .[, exceedance_2050:= exceedance_2023 + additional_water_2050] %>%
  .[, exceedance_by_2050:= ifelse(exceedance_2050 > 0, "Yes", "No")]

# SOME STATS ###################################################################

cols <- c("land_runoff", "accessible_water_runoff", "exceedance_2023")
summary_fun = function(x) list(min = min(x), max = max(x))
dt[, lapply(.SD, summary_fun), .SDcols = (cols)]
```

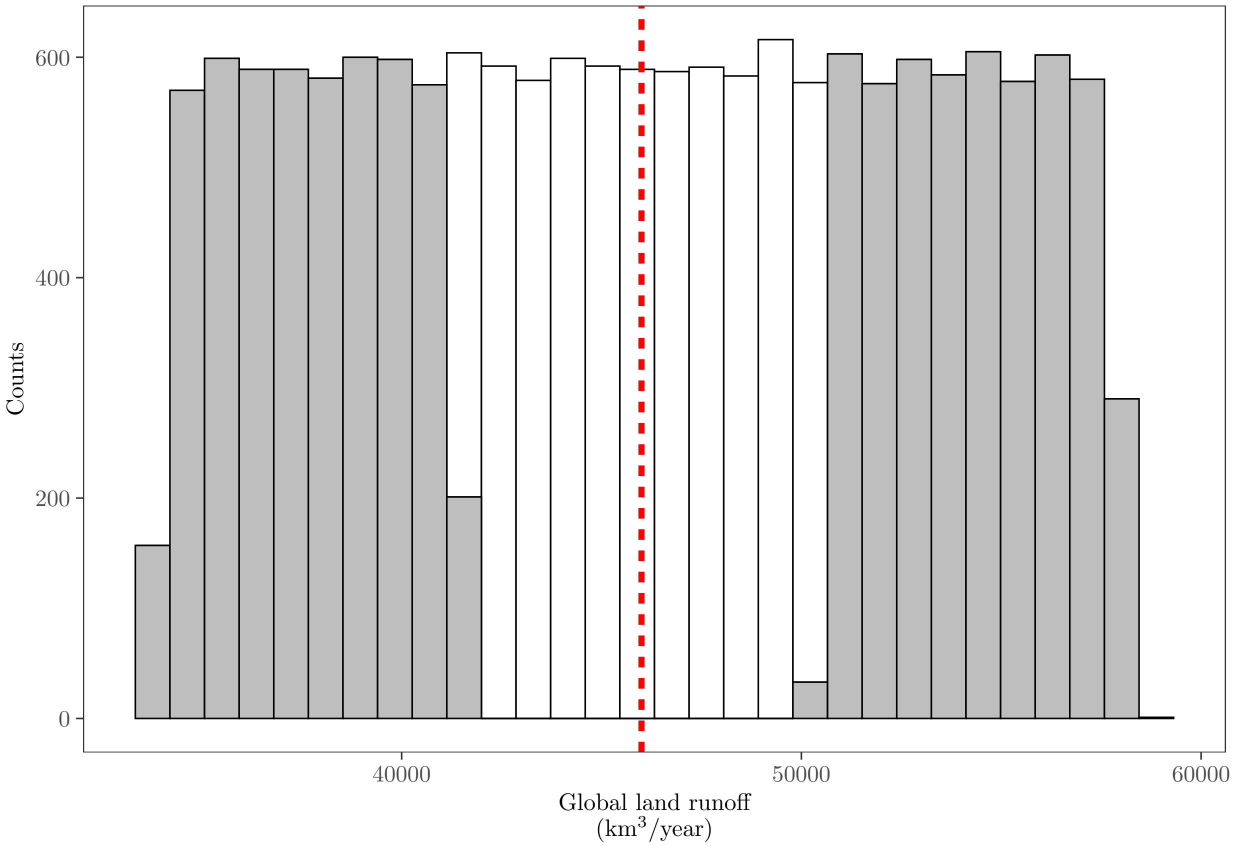
```
##    land_runoff accessible_water_runoff exceedance_2023
##         <list>                  <list>          <list>
## 1:    33441.89                5241.895       -2593.442
## 2:    58552.83                30352.83        1333.539
```

```r
tmp <- melt(dt, measure.vars = c("outside", "outside_runoff", "water_deficit")) %>%
  .[, .N, .(variable, value)]

tmp[, total:= (2^13 * 2)] %>%
  .[, prop:= N / total] %>%
  print()
```

```
##          variable  value     N total      prop
##            <fctr> <char> <int> <num>     <num>
## 1:        outside     No  6275 16384 0.3829956
## 2:        outside    Yes 10109 16384 0.6170044
## 3: outside_runoff     No  4089 16384 0.2495728
## 4: outside_runoff    Yes 12295 16384 0.7504272
## 5:  water_deficit     No 12571 16384 0.7672729
## 6:  water_deficit    Yes  3813 16384 0.2327271
```

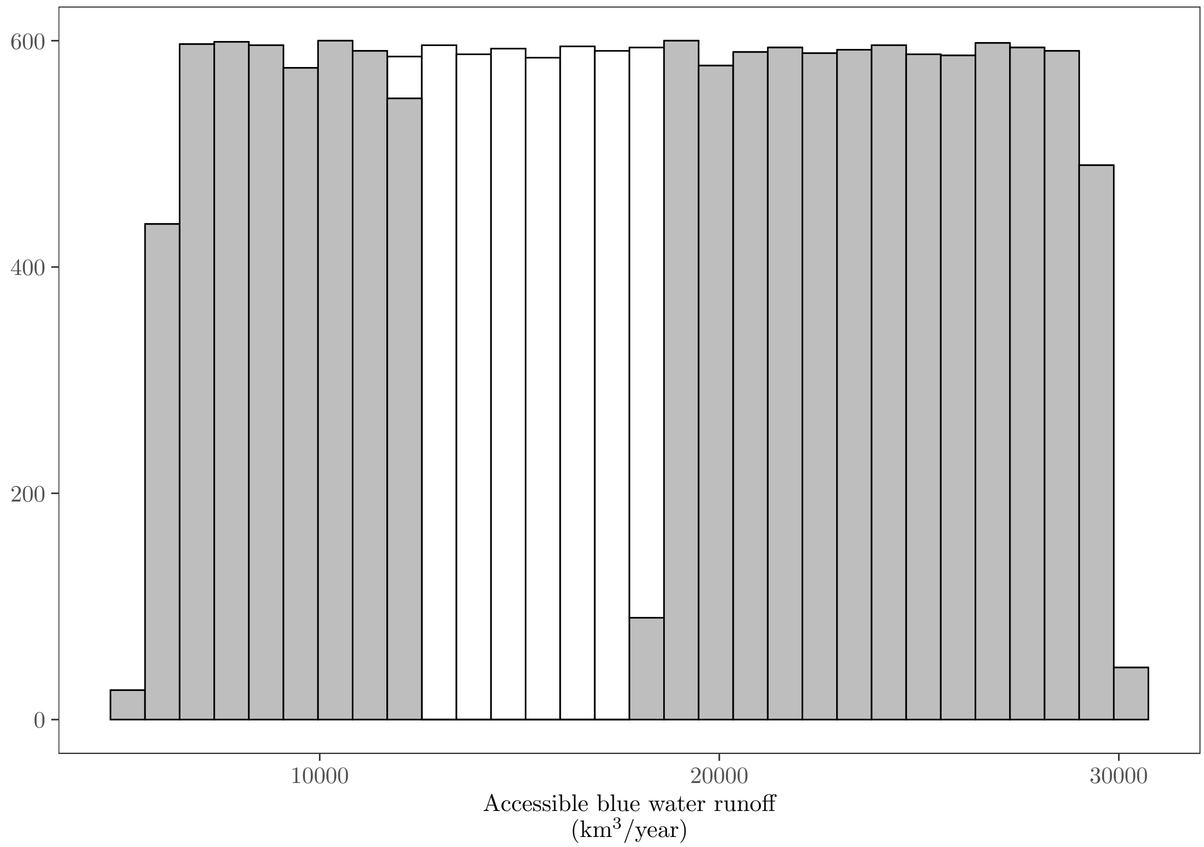
```r
# PLOT LAND RUNOFF DISTRIBUTION ################################################

plot_land_runoff <- ggplot(dt, aes(land_runoff, fill = outside)) +
  geom_histogram(colour = "black") +
  scale_fill_manual(values = c("white", "grey")) +
  theme_AP() +
  geom_vline(xintercept = land_runoff_estimate, color = "red", lty = 2, size = 2) +
  labs(x = "Global land runoff \n (km$^3$/year)", y = "Counts") +
  theme(legend.position = "none")

plot_land_runoff
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



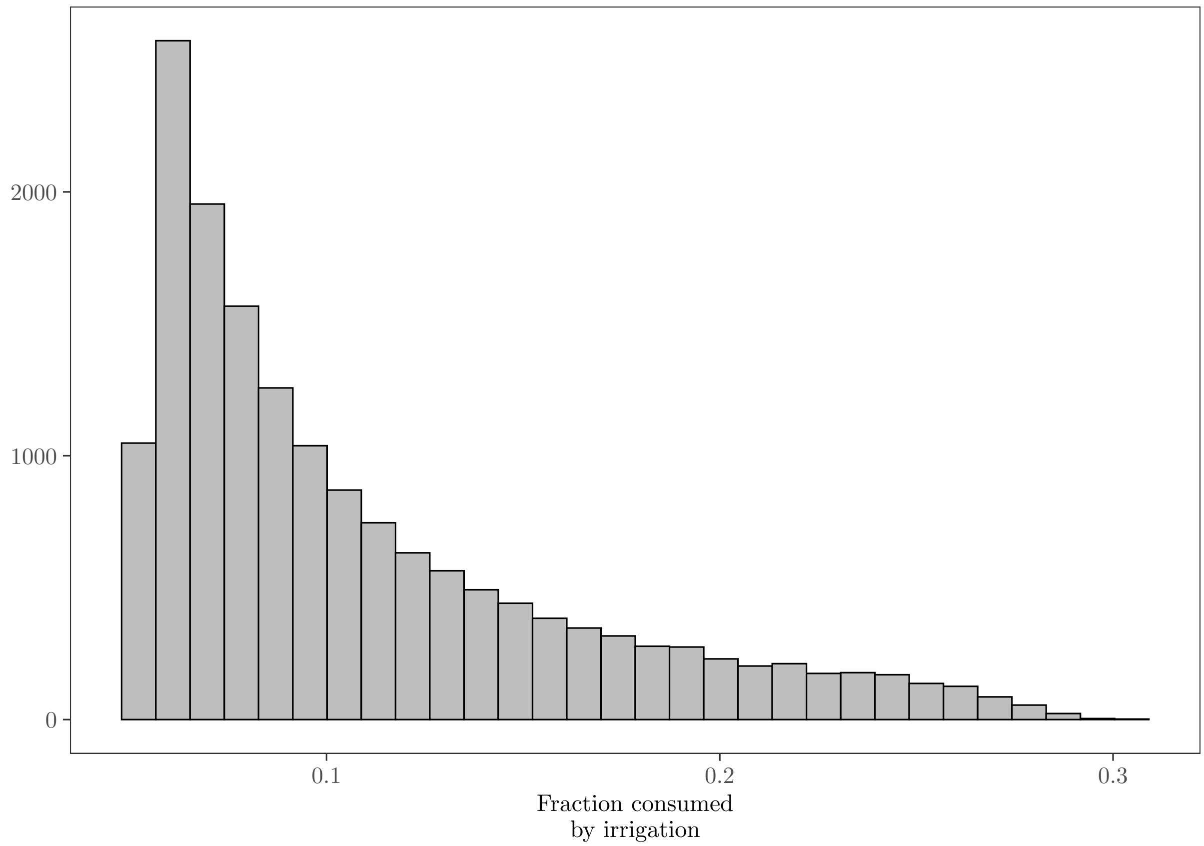

```
# PLOT ACCESSIBLE RUNOFF ###########################################################

plot_accessible_runoff <- ggplot(dt, aes(accessible_water_runoff, fill = outside_runoff)) +
  geom_histogram(colour = "black") +
  scale_fill_manual(values = c("white", "grey")) +
  theme_AP() +
  labs(x = "Accessible blue water runoff \n (km$^3$/year)", y = "") +
  theme(legend.position = "none")

plot_accessible_runoff
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
# PLOT ACCESSIBLE BLUE WATER RUNOFF CONSUMES BY IRRIGATION ####################

vec <- data.table(fraction.irrig = 1600 / dt$accessible_water_runoff)

plot.irrigation <- ggplot(vec, aes(fraction.irrig)) +
  geom_histogram(fill = "grey", color = "black") +
  labs(x = "Fraction consumed \n by irrigation", y = "") +
  theme_AP()

plot.irrigation
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

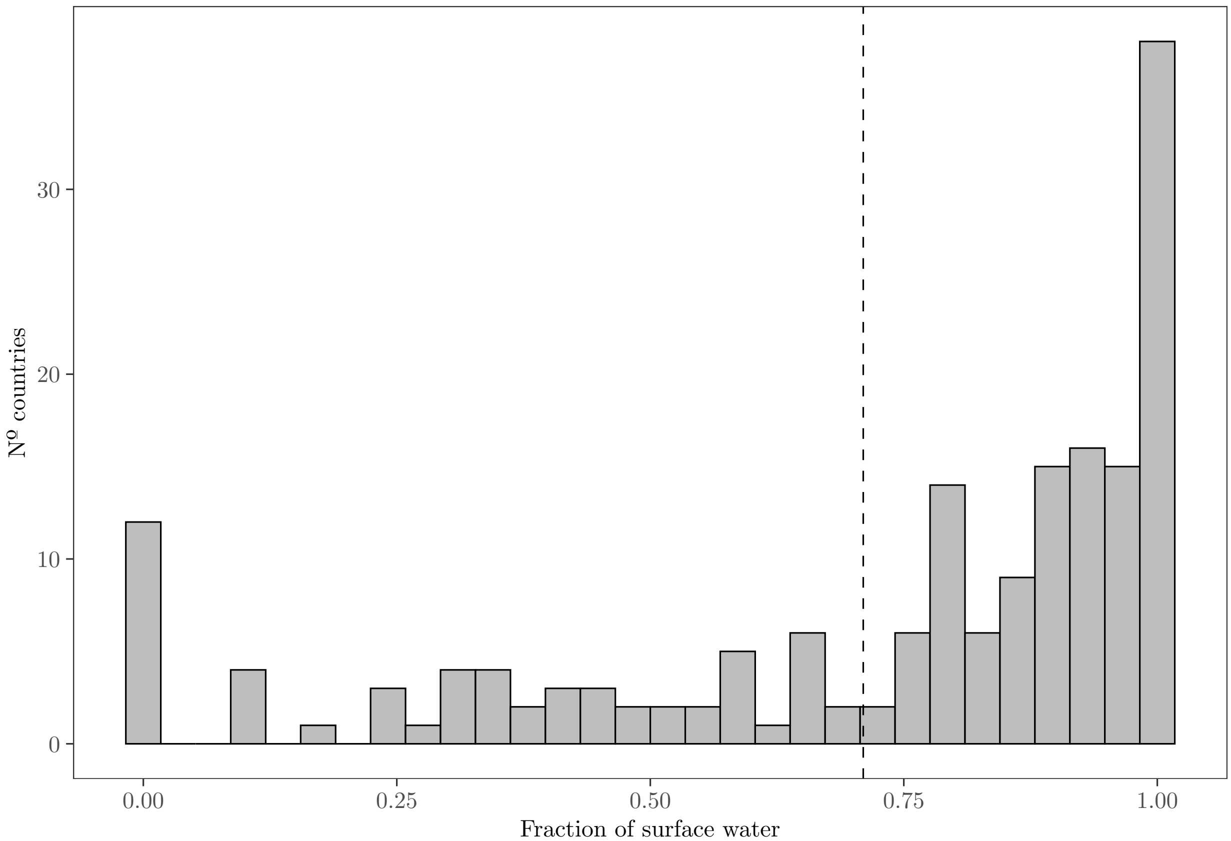



```
# FRACTION OF IRRIGATION SUPPLIED BY SURFACE WATER #############################

da <- data.table(readxl::read_xls("/Users/arnaldpuy/Documents/papers/fallacies_water_crisis/coc
da <- da[, fraction.gw:= `ICU_GW (m3 yr-1)` / `ICU (m3 yr-1)`] %>%
  .[, fraction.irr:= `ICU_SW (m3 yr-1)` / `ICU (m3 yr-1)`]


da[fraction.irr < 0.05] %>%
  .[, .(COUNTRY, fraction.irr, fraction.gw)]
```

```
##                           COUNTRY fraction.irr fraction.gw
##                            <char>        <num>       <num>
##  1:                       Bahrain  0.000000000   0.9031133
##  2:                       Denmark  0.000000000   1.0000000
##  3:                      Djibouti  0.000000000   1.0000000
##  4:                        Kuwait  0.000000000   0.6100000
##  5:        Libyan Arab Jamahiriya  0.005554530   0.9888909
##  6:                         Malta  0.005586855   0.9944131
##  7:                    Montenegro  0.003619909   0.9963801
##  8: Occupied Palestinian Territory  0.000000000   1.0000000
##  9:                          Oman  0.000000000   1.0000000
## 10:                         Qatar  0.000000000   0.9339782
## 11:                  Saudi Arabia  0.000000000   0.9700133
## 12:          United Arab Emirates  0.000000000   1.0000000
```

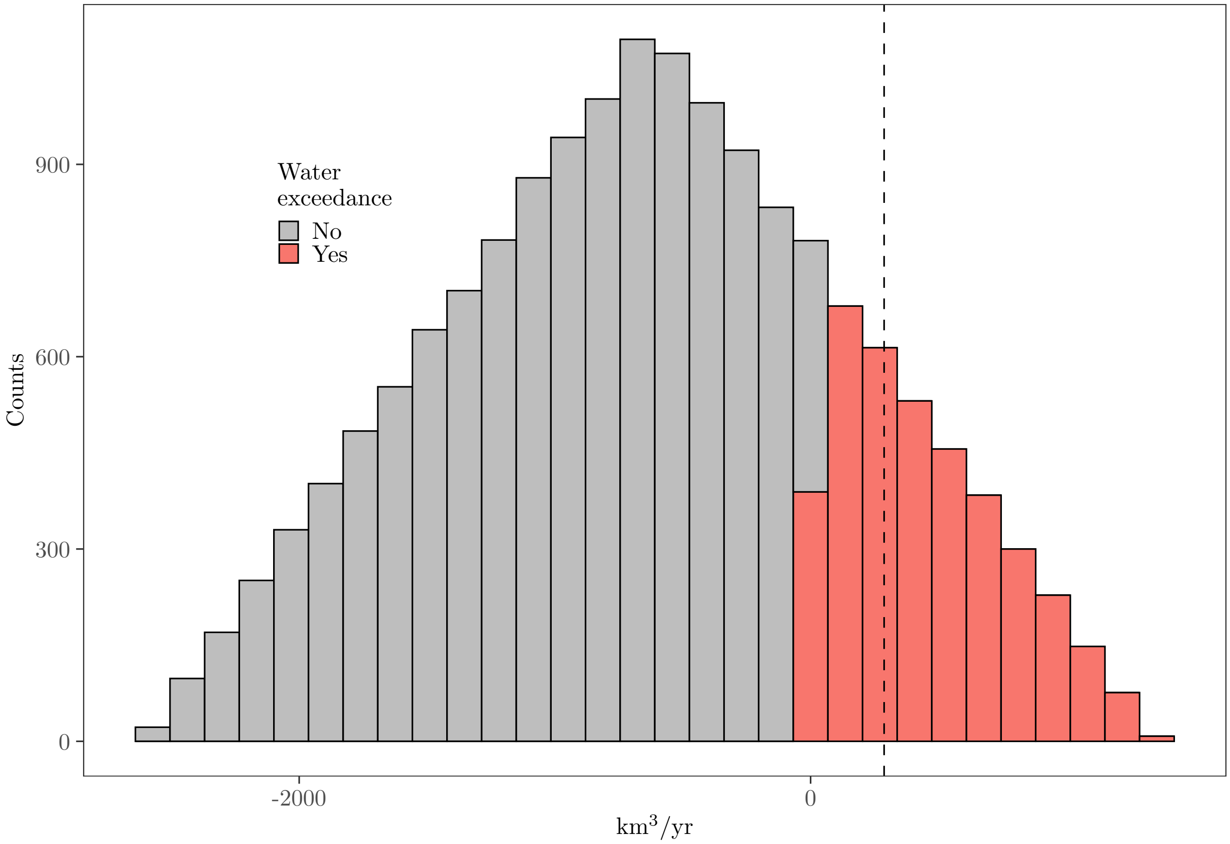
```r
fraction.surface <- ggplot(da, aes(fraction.irr)) +
  geom_histogram(color = "black", fill = "grey") +
  theme_AP() +
  geom_vline(xintercept = 0.71, lty = 2) +
  labs(x = "Fraction of surface water", y = "Nº countries")

fraction.surface
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



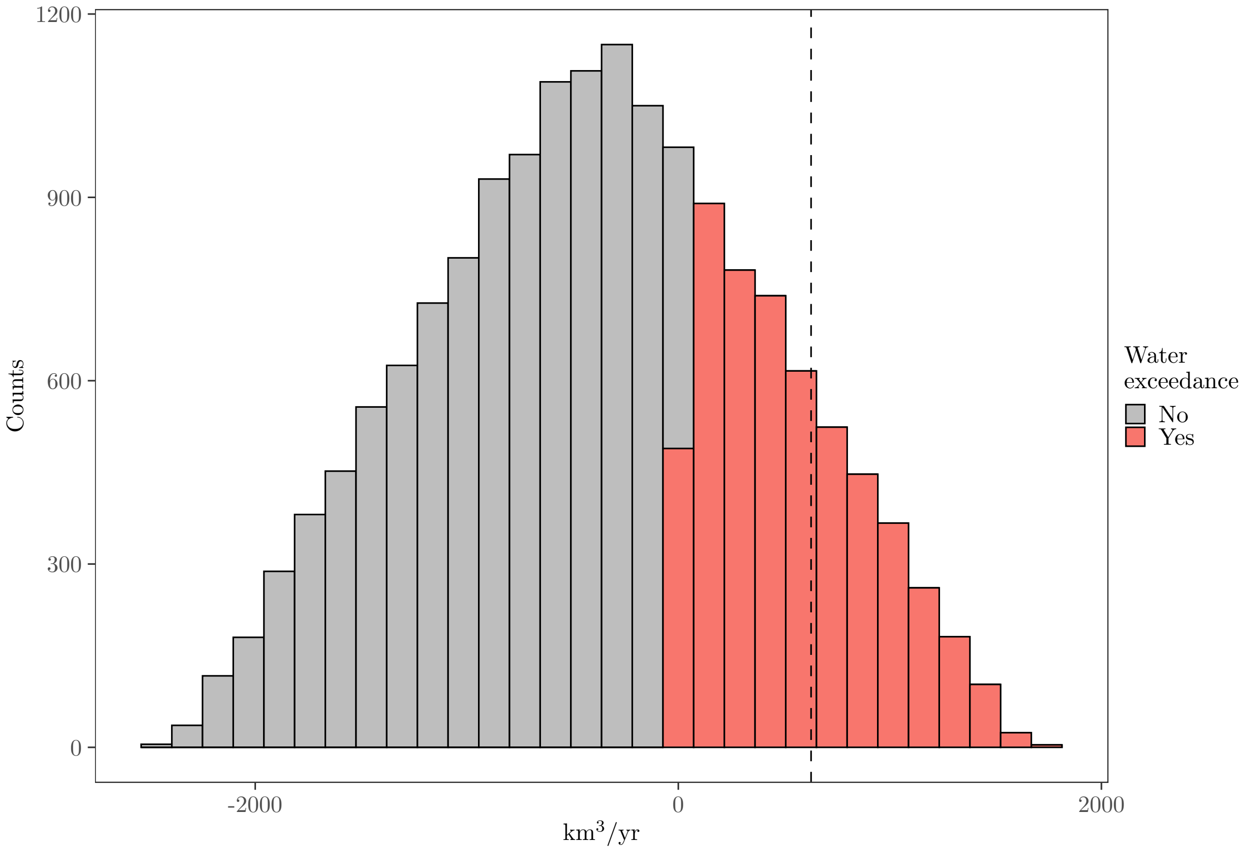

```r
# PLOT CORRECTED WATER LIMIT EXCEEDANCE FOR 2023 ##############################

plot.exceedance <- ggplot(dt, aes(exceedance_2023, fill = water_deficit)) +
  geom_histogram(color = "black") +
  scale_fill_manual(values = c("grey", "#F8766D"),
                    name = "Water \n exceedance") +
  scale_x_continuous(breaks = pretty_breaks(n = 2)) +
  labs(x = "km$^3$/yr", y = "Counts") +
  geom_vline(xintercept = 287.5, lty = 2) +
  theme_AP() +
  theme(legend.position = c(0.22, 0.73))

plot.exceedance
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
# PLOT WATER EXCEEDANCE IN 2025 ##################################################

plot.exceedance.2050 <- ggplot(dt, aes(exceedance_2050, fill = exceedance_by_2050)) +
  geom_histogram(color = "black") +
  scale_fill_manual(values = c("grey", "#F8766D"),
                    name = "Water \n exceedance") +
  scale_x_continuous(breaks = pretty_breaks(n = 2)) +
  geom_vline(xintercept = 627.5, lty = 2) +
  labs(x = "km$^3$/yr", y = "Counts") +
  theme_AP()

plot.exceedance.2050
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



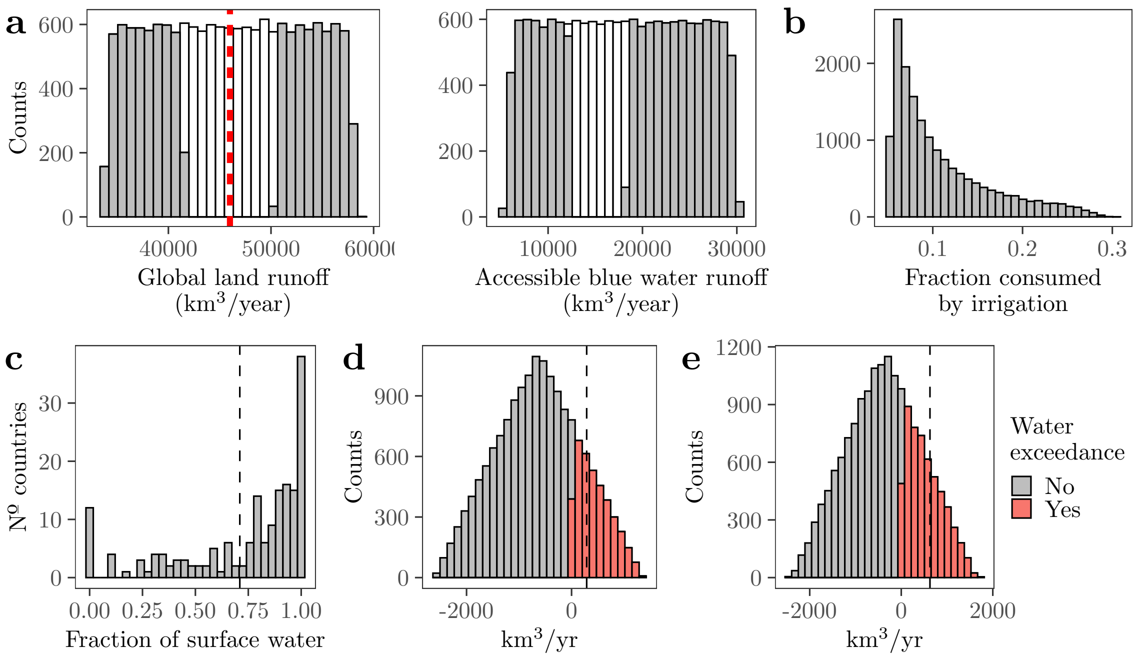

```
# MERGE PLOTS ##################################################################

top <- plot_grid(plot_land_runoff, plot_accessible_runoff, plot.irrigation,
                 labels = c("a", "", "b"), rel_widths = c(0.34, 0.32, 0.32), ncol = 3)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
bottom <- plot_grid(fraction.surface, plot.exceedance +
                      theme(legend.position = "none"), plot.exceedance.2050,
                    labels = c("c", "d", "e"), rel_widths = c(0.30, 0.305, 0.425), ncol = 3)
```

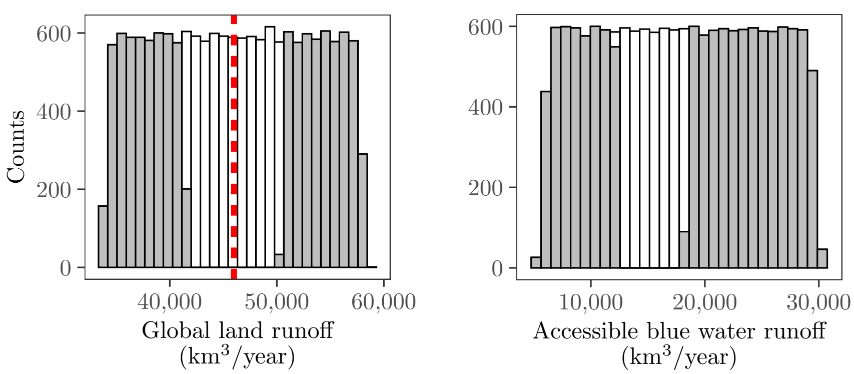
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
plot_grid(top, bottom, ncol = 1)
```




```
########

plot_grid(plot_land_runoff +
            scale_x_continuous(labels = scales::comma) +
            theme(plot.margin = unit(c(0.2, 0.45, 0.2, 0.2), "cm")),
          plot_accessible_runoff +
            scale_x_continuous(labels = scales::comma),
          ncol = 2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
########

top <- plot_grid(plot.irrigation, fraction.surface, labels = "auto")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
bottom <- plot_grid(plot.exceedance + theme(legend.position = "none"),
                    plot.exceedance.2050 + theme(legend.position = "none"),
                    ncol = 2,
                    labels = c("c", "d"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
legend <- get_legend(plot.exceedance + theme(legend.position = "top"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
legend.bottom <- plot_grid(legend, bottom, ncol = 1, rel_heights = c(0.15, 0.85))

final.up <- plot_grid(top, legend.bottom, ncol = 1)

final.down <- plot_grid(plot.caloric, plot.ind, ncol = 2, labels = c("e", "f"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
plot_grid(final.up, final.down, ncol = 1, rel_heights = c(0.65, 0.35))
```