Modeling the contribution of eflows to sustainable agriculture, food security and livelihoods in South Africa’s Limpopo basin

Eike Luedeling, Cory Whitney

We generate a holistic model to simulate the contribution of eflows to sustainable agriculture, food security and livelihoods. Spatially, we do this for only a small portion of the basin as a test-case. We apply holistic modeling approaches to generate conceptual impact pathways and quantitative models to forecast decision outcomes (see Do, Luedeling, and Whitney 2020; Lanzanova et al. 2019; Cory Whitney et al. 2018). This includes collaborative model development (C. Whitney, Shepherd, and Luedeling 2018) to assess farming futures given eflow forecasts under different management options. To build these simulations we use functions from the decisionSupport (Luedeling et al. 2021), dplyr (Wickham, François, et al. 2021), nasapower (Sparks 2021), patchwork (Pedersen 2020), tidyverse (Wickham 2021) and Evapotranspiration (Guo, Westra, and Peterson 2020) libraries in the R programming language (R Core Team 2021).

## The model

Decision-makers often wish to have a quantitative basis for their decisions. However,‘hard data’ is often missing or unattainable for many important variables, which can paralyze the decision-making processes or lead decision-makers to conclude that large research efforts are needed before a decision can be made. That is, many variables decision makers must consider cannot be precisely quantified, at least not without unreasonable effort. The major objective of (prescriptive) decision analysis is to support decision-making processes where decision makers are faced with this problem. Following the principles of Decision Analysis can allow us to make forecasts of decision outcomes without precise numbers, as long as probability distributions describing the possible values for all variables can be estimated.

The decisionSupport package implements this as a Monte Carlo simulation, which generates a large number of plausible system outcomes, based on random numbers for each input variable that are drawn from user-specified probability distributions. This approach is useful for determining whether a clearly preferable course of action can be delineated based on the present state of knowledge without the need for further information. If the distribution of predicted system outcomes does not imply a clearly preferable decision option, variables identified as carrying decision-relevant uncertainty can then be targeted by decision-supporting research.

The mcSimulation function from the decisionSupport package can be applied to conduct decision analysis (Luedeling et al. 2021). The function requires three inputs:

1. an estimate of the joint probability distribution of the input variables. These specify the names and probability distributions for all variables used in the decision model. These distributions aim to represent the full range of possible values for each component of the model.
2. a model\_function that predicts decision outcomes based on the variables named in a separate data table. This R function is customized by the user to address a particular decision problem to provide the decision analysis model.
3. numberOfModelRuns indicating the number of times to run the model function.

These inputs are provided as arguments to the mcSimulation function, which conducts a Monte Carlo analysis with repeated model runs based on probability distributions for all uncertain variables. The data table and model are customized to fit the particulars of a specific decision.

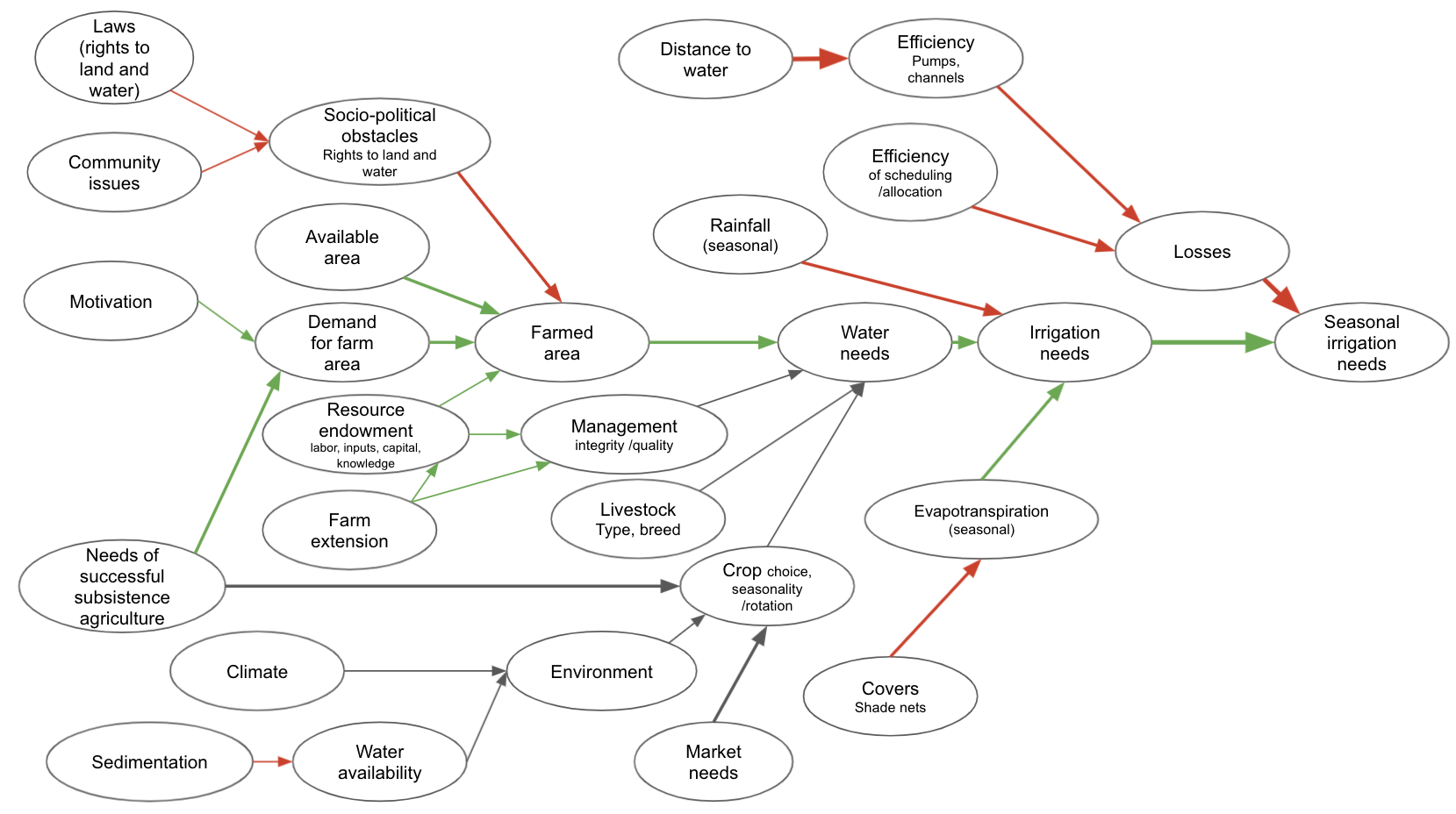
### The estimate

To support the model building process we design an input table to store the estimate values. The table is stored locally as limpopo\_input\_table.csv and contains many of the basic values for the analysis. This table contains all the input variables used in the model. Their distributions are described by 90% confidence intervals, which are specified by lower (5% quantile) and upper (95% quantile) bounds, as well as the shape of the distribution. This model uses four different distributions:

1. const – a constant value
2. norm – a normal distribution
3. tnorm\_0\_1 – a truncated normal distribution that can only have values between 0 and 1 (useful for probabilities; note that 0 and 1, as well as numbers outside this interval are not permitted as inputs)
4. posnorm – a normal distribution truncated at 0 (only positive values allowed)

For a full list of input variables with descriptions and the chosen distributions see the table at the end of this document.

### The model\_function



Model of the social effects of altered river flows on the sustainability of livelihoods in the Limpopo Basin

The decision model is coded as an R function which takes in the variables provided in the data table and generates a model output, such as the Net Present Value.

In the following we use of various decisionSupport functions, which use the tidyverse libraries (Wickham et al. 2019) including ggplot2 (Wickham, Chang, et al. 2021), plyr (Wickham 2020) and dplyr (Wickham, François, et al. 2021) among others in the [R programming language](https://www.r-project.org/) (R Core Team 2021).

Here we generate a model as a function using decisionSupport library we use the decisionSupport functions vv() to produce time series with variation from a pre-defined mean and coefficient of variation, chance\_event() to simulate whether events occur and discount() to discount values along a time series and generate a Net Present Value for our intervention comparison.

### Scenarios

The following function defines 3 scenarios:

1. No eflows: This is a scenario without eflows. Farmers extract water according to their irrigation needs. Extractions are only limited by the minimum water level that allows operating the pumps.
2. Restricted extraction: This is an eflow scenario, in which eflows are interpreted in a purely ecological sense. Whenever eflows aren’t achieved, water extraction is curtailed. There are no measures to add water to the river in such events. We simulate Scenario 1 with our own functions and some from the nasapower (Sparks 2021) and Evapotranspiration (Guo, Westra, and Peterson 2020) packages.
3. Dam releases: This is an eflow scenario, in which eflows are interpreted as encompassing the ecological as well as the smallholder irrigation requirement. In case eflows aren’t naturally met, water is released from upstream dams to ensure eflows. Extraction by smallholder farmers is restricted only by the ability to operate the pumps.

The following script contains the basic model we used to run the Monte Carlo.

limpopo\_decision\_function <- function(x, varnames){  
  
# generating boundary conditions for the simulation run   
  
# simulate how much rainwater is available  
rainfall<-sapply(1:12,function(x) eval(parse(text=paste0("prec\_",x))))  
  
effective\_rainfall<-sapply(rainfall,function(x) min(x,effprec\_high))  
effective\_rainfall<-sapply(effective\_rainfall,function(x) max(x,effprec\_low))  
  
# We compute crop water needs based on ET0   
# Computed based on the Hargreaves Samani equation, as implemented in the Evapotranspiration package)  
# Input temperature data comes from the NASAPOWER dataset   
# (accessed through the nasapower package)   
# The scenario data will be based on scenarios that represent conditions   
# during real years in the past  
# To get from ET0 to crop water use, we need to multiply ET0 with a crop  
# coefficient (kc), which is estimated for each month  
  
ET0<-sapply(1:12,function(x) eval(parse(text=paste0("ET0\_",x)))) # in mm  
  
kc<-sapply(1:12,function(x) eval(parse(text=paste0("kc\_",x)))) # in mm  
  
cropwat\_need<-ET0\*kc # in mm  
  
irrigation\_need<-cropwat\_need-effective\_rainfall # in mm  
  
# define river flow and eflow for each month ####  
# Base river flow data from 1920 to 2010, Letaba River at EWR site EWR4 (Letaba Ranch upstream Little Letaba confluence)  
pre\_livestock\_river\_flow<-sapply(1:12,function(x) eval(parse(text=paste0("river\_flow\_",x)))) # in m3 / month  
eflow<-sapply(1:12,function(x) eval(parse(text=paste0("eflow\_",x)))) # in m3 / month  
  
# watering livestock  
# assuming that this is more or less stable throughout the year, but varies a bit  
livestock\_water\_needs<-vv(livestock\_water\_need,var\_CV,12)  
  
# assuming that the eflows aren't affecting ability to water livestock and that there's always enough  
# water for all the livestock  
river\_flow<-pre\_livestock\_river\_flow-livestock\_water\_needs  
  
# calculating the farmed area  
  
demand\_for\_farm\_area<-n\_subsistence\_farmers\*necessary\_farm\_size\_per\_household  
  
farmed\_area<-min(available\_area, demand\_for\_farm\_area)\*(1-unused\_sociopolit)  
  
total\_cropwater\_need<-cropwat\_need\*farmed\_area\*10 # total water need in m3 (the 10 is the mm to m3/ha conversion)  
total\_effective\_rainfall<-effective\_rainfall\*farmed\_area\*10 # total effective rainfall  
  
# total irrigation need  
total\_irrigation\_need<-total\_cropwater\_need-total\_effective\_rainfall # in m3  
  
# water losses are calculated from the efficiency of the pumps and the water allocation  
efficiency\_pumps<-vv(effi\_pump,var\_CV,12)  
efficiency\_irrig\_scheduling<-vv(effi\_sched,var\_CV,12)  
efficiency\_pumps<-sapply(efficiency\_pumps, function(x) min(x,1))  
efficiency\_pumps<-sapply(efficiency\_pumps, function(x) max(x,0))  
efficiency\_irrig\_scheduling<-sapply(efficiency\_irrig\_scheduling, function(x) min(x,1))  
efficiency\_irrig\_scheduling<-sapply(efficiency\_irrig\_scheduling, function(x) max(x,0))  
  
water\_losses\_share<-(1-efficiency\_pumps\*efficiency\_irrig\_scheduling)  
  
irrigation\_water\_need<-total\_irrigation\_need/(1-water\_losses\_share)  
  
# eflow Scenario 1 - no eflows  
  
scen1\_usable\_river\_flow<-sapply(1:12,function(x) max(0,river\_flow[x]-minimum\_flow\_to\_operate\_pumps))  
  
# eflow Scenario 2 - eflows as a limit to extraction only  
  
# eflows are to be ensured whenever there is more water in the river than the eflow  
# requirement would mandate, i.e. farmers aren't allowed to extract water beyond  
# the eflow requirement.  
# no measures are taken to ensure that eflows are maintained at times when  
# the present flow is below the eflow requirement.   
  
scen2\_usable\_river\_flow<-sapply(1:12,function(x) max(0,river\_flow[x]-max(eflow[x],minimum\_flow\_to\_operate\_pumps)))  
  
# eflow Scenario 3 - eflows are assured by dam releases  
  
# whenever the present flow is below the eflow requirement, water is released  
# from an upstream dam to ensure that the eflows are met.  
  
adj\_river\_flow <- sapply(1:12, function(x)  
 max(river\_flow[x], eflow[x]))  
  
required\_dam\_release <- adj\_river\_flow - river\_flow  
  
scen3\_usable\_river\_flow <-  
 sapply(1:12, function(x)  
 max(0, adj\_river\_flow[x] - minimum\_flow\_to\_operate\_pumps))  
  
# calculate how much water gets extracted from the river  
  
scen1\_extracted\_river\_water <-  
 sapply(1:12, function(x)  
 min(scen1\_usable\_river\_flow[x], irrigation\_water\_need[x]))  
scen2\_extracted\_river\_water <-  
 sapply(1:12, function(x)  
 min(scen2\_usable\_river\_flow[x], irrigation\_water\_need[x]))  
scen3\_extracted\_river\_water <-  
 sapply(1:12, function(x)  
 min(scen3\_usable\_river\_flow[x], irrigation\_water\_need[x]))  
  
# calculate damage to crop production due to lack of irrigation water  
scen1\_water\_shortfall <-  
 sapply(1:12, function (x)  
 max(0, irrigation\_water\_need[x] - scen1\_extracted\_river\_water[x]))  
scen2\_water\_shortfall <-  
 sapply(1:12, function (x)  
 max(0, irrigation\_water\_need[x] - scen2\_extracted\_river\_water[x]))   
scen3\_water\_shortfall <-  
 sapply(1:12, function (x)  
 max(0, irrigation\_water\_need[x] - scen3\_extracted\_river\_water[x]))  
  
scen1\_irrigation\_shortfall<-scen1\_water\_shortfall\*(1-water\_losses\_share)  
scen2\_irrigation\_shortfall<-scen2\_water\_shortfall\*(1-water\_losses\_share)  
scen3\_irrigation\_shortfall<-scen3\_water\_shortfall\*(1-water\_losses\_share)  
  
scen1\_crop\_water\_gap<-scen1\_irrigation\_shortfall/(cropwat\_need\*farmed\_area\*10)  
scen2\_crop\_water\_gap<-scen2\_irrigation\_shortfall/(cropwat\_need\*farmed\_area\*10)  
scen3\_crop\_water\_gap<-scen3\_irrigation\_shortfall/(cropwat\_need\*farmed\_area\*10)  
  
# calculate how much water is left after farmers extracted water  
scen1\_river\_flow\_downstream<-river\_flow-scen1\_extracted\_river\_water  
scen2\_river\_flow\_downstream<-river\_flow-scen2\_extracted\_river\_water  
scen3\_river\_flow\_downstream<-adj\_river\_flow-scen3\_extracted\_river\_water  
  
# calculate outputs and differences   
  
return(list(cropwater\_need=total\_cropwater\_need,  
 yearly\_crop\_water\_need=sum(total\_cropwater\_need),  
 irrigation\_water\_need=irrigation\_water\_need,  
 yearly\_irrigation\_water\_need=sum(irrigation\_water\_need),  
 scen1\_downstream\_river\_flow=mean(scen1\_river\_flow\_downstream),  
 scen2\_downstream\_river\_flow=mean(scen2\_river\_flow\_downstream),  
 scen3\_downstream\_river\_flow=mean(scen3\_river\_flow\_downstream),  
 scen3\_dam\_release=required\_dam\_release,  
 scen3\_total\_dam\_release=sum(required\_dam\_release),  
 Downstream\_river\_flow\_1\_=scen1\_river\_flow\_downstream,  
 Downstream\_difference\_2\_vs\_1=scen2\_river\_flow\_downstream-scen1\_river\_flow\_downstream,  
 Downstream\_difference\_3\_vs\_1=scen3\_river\_flow\_downstream-scen1\_river\_flow\_downstream,  
 scen1\_crop\_water\_gap=mean(scen1\_crop\_water\_gap),  
 scen2\_crop\_water\_gap=mean(scen2\_crop\_water\_gap),  
 scen3\_crop\_water\_gap=mean(scen3\_crop\_water\_gap),  
 Crop\_water\_gap\_scen1\_=scen1\_crop\_water\_gap,  
 Crop\_water\_gap\_difference\_2\_vs\_1=scen2\_crop\_water\_gap-scen1\_crop\_water\_gap,  
 Crop\_water\_gap\_difference\_3\_vs\_1=scen3\_crop\_water\_gap-scen1\_crop\_water\_gap,  
 Mean\_Crop\_water\_gap\_difference\_2\_vs\_1=mean(scen2\_crop\_water\_gap-scen1\_crop\_water\_gap),  
 Mean\_Crop\_water\_gap\_difference\_3\_vs\_1=mean(scen3\_crop\_water\_gap-scen1\_crop\_water\_gap)))  
   
}

#### Perform a Monte Carlo simulation with scenarios

Using the model function above, we can perform a Monte Carlo simulation with the mcSimulation() function from decisionSupport. This function generates distributions of all variables in the input table as well as the specified model outputs (see return() function above) by calculating random draws in our defined limpopo\_decision\_function(). We run a visual assessment to ensure that all the variables in the input table are included in the model (erroneous variables listed there can cause issues with some of the post-hoc analyses).

The numberOfModelRuns argument is an integer indicating the number of model runs for the Monte Carlo simulation. Unless the model function is very complex, 10,000 runs is a reasonable choice (for complex models, 10,000 model runs can take a while, so especially when the model is still under development, it often makes sense to use a lower number).

We first make a scenario file, for which we can use data for 1980 to 2020.

# load data from Evapotranspiration  
data("constants")  
  
# use nasapower for evapotranspiration data  
ag\_d <- get\_power(  
 community = "ag",  
 lonlat = c(31.08,-23.7),  
 pars = c("T2M\_MAX", "T2M\_MIN", "PRECTOTCORR"),  
 dates = c("1981-01-01", "2020-12-31"),  
 temporal\_api = "daily"  
)  
  
# choose years of assessment  
years<-1981:2009  
  
# name variables  
colnames(ag\_d)[c(3:5, 8, 9, 10)] <-  
 c("Year", "Month", "Day", "Tmax", "Tmin", "Precipitation")  
  
Inputs <- ReadInputs(c("Tmin", "Tmax"), ag\_d, stopmissing = c(50, 50, 50))  
#> The maximum acceptable percentage of date indices is 50 %  
#> The maximum acceptable percentage of missing data is 50 %  
#> The maximum acceptable percentage of continuous missing data is 50 %  
  
# apply ET.HargreavesSamani from the Evapotranspiration library  
ET <-  
 ET.HargreavesSamani(  
 Inputs,  
 constants,  
 ts = "daily",  
 message = "yes",  
 AdditionalStats = "yes",  
 save.csv = "no"  
 )  
#> Hargreaves-Samani Reference Crop ET  
#> Evaporative surface: reference crop  
#> Timestep: daily  
#> Units: mm  
#> Time duration: 1981-01-01 to 2020-12-31  
#> 14610 ET estimates obtained  
#> Basic stats  
#> Mean: 4.94  
#> Max: 30.45  
#> Min: 1.35  
  
ETdata <- data.frame(year = years)  
ETdata[, month.abb[1:12]] <- NA  
for (yyyy in years)  
 ETdata[which(ETdata$year == yyyy), 2:13] <-  
 ET$ET.Monthly[as.character(yyyy + 0:11 / 12)]  
  
rain <-  
 aggregate(ag\_d$Precipitation,  
 by = list(ag\_d$Year, ag\_d$Month),  
 FUN = sum)  
raindata <- data.frame(year = years)  
raindata[, month.abb[1:12]] <- NA  
for (yyyy in years)  
 raindata[which(raindata[, 1] == yyyy), 2:13] <-  
 rain[which(rain[, 1] == yyyy), 3]  
  
  
scenario\_variables <-  
 c(paste0("river\_flow\_", 1:12),  
 paste0("ET0\_", 1:12),  
 paste0("prec\_", 1:12),  
 paste0("eflow\_", 1:12))  
  
Scenarios <- data.frame(Variable = scenario\_variables, param = "both")  
  
eflows<-read.csv("data/Letaba\_eflows\_exceedence\_m3\_per\_s.csv",fileEncoding="UTF-8-BOM")  
eflowsort <-  
 eflows[, c(1, order(unlist(sapply(colnames(eflows)[2:13], function(x)  
 which(month.abb[1:12] == x)))) + 1)]  
eflow\_exceedance<-eflowsort[which(eflowsort$Exceedence == 80),]  
eflow\_per\_month<-eflow\_exceedance[2:13]\*c(31,28,31,30,31,30,31,31,30,31,30,31)\*3600\*24  
  
# read data of present data   
present\_flows<-read.csv("data/Letaba\_modelled\_present\_flows\_m3\_per\_s.csv",fileEncoding="UTF-8-BOM")  
presentflowsort <-  
 present\_flows[, c(1, order(unlist(sapply(colnames(present\_flows)[2:13], function(x)  
 which(month.abb[1:12] == x)))) + 1)]  
presentflow\_permonth<-data.frame(cbind(presentflowsort[,1],t(t(presentflowsort[,2:13])\*c(31,28,31,30,31,30,31,31,30,31,30,31)\*3600\*24)))  
colnames(presentflow\_permonth)[1]<-"Year"  
  
# The hydrological year in the input file starts in October and runs until September. We're assuming here that the year given for each year in the data sets provided corresponds to the first calendar year of this period.  
  
presentflow\_permonth[2:nrow(presentflow\_permonth),month.abb[1:9]]<-  
 presentflow\_permonth[1:(nrow(presentflow\_permonth)-1),month.abb[1:9]]  
  
presentflow\_permonth[1,month.abb[1:9]]<-NA  
  
for (yyyy in years)  
{  
 Scenarios[, paste0("y\_", yyyy)] <- NA  
 for (mm in 1:12)  
 {  
 Scenarios[which(Scenarios$Variable == paste0("ET0\_", mm)), paste0("y\_", yyyy)] <-  
 ETdata[which(ETdata$year == yyyy), 1 + mm]  
 Scenarios[which(Scenarios$Variable == paste0("prec\_", mm)), paste0("y\_", yyyy)] <-  
 raindata[which(raindata$year == yyyy), 1 + mm]  
 Scenarios[which(Scenarios$Variable == paste0("river\_flow\_", mm)), paste0("y\_", yyyy)] <-  
 presentflow\_permonth[which(presentflow\_permonth$Year == yyyy), 1 + mm]  
 Scenarios[which(Scenarios$Variable == paste0("eflow\_", mm)), paste0("y\_", yyyy)] <-  
 eflow\_per\_month[mm]  
 }  
}  
  
# natural flows (this is for information and not used in the model)  
natural\_flows<-read.csv("data/Letaba\_modelled\_natural\_flows\_m3\_per\_s.csv",fileEncoding="UTF-8-BOM")  
  
# write the scenarios file  
write.csv(Scenarios, "data/scenarios\_1980\_2020.csv", row.names = FALSE)

Here we run the model with the scenario\_mc function from the decisionSupport package (Luedeling et al. 2021). The function essentially generates a Monte Carlo model with data from existing scenarios for some of the model inputs.

# run the model with the scenario\_mc function   
mcSimulation\_results <-  
 scenario\_mc(  
 base\_estimate = decisionSupport::estimate\_read\_csv("data/limpopo\_input\_table.csv"),  
 scenarios = read.csv("data/scenarios\_1980\_2020.csv", fileEncoding =  
 "UTF-8-BOM"),  
 model\_function = limpopo\_decision\_function,  
 numberOfModelRuns = 1e2, #run 100 times (2900 with 100 simulations of 29 scenarios)  
 functionSyntax = "plainNames"  
 )

## Results

### Water needs

Annual crop water needs

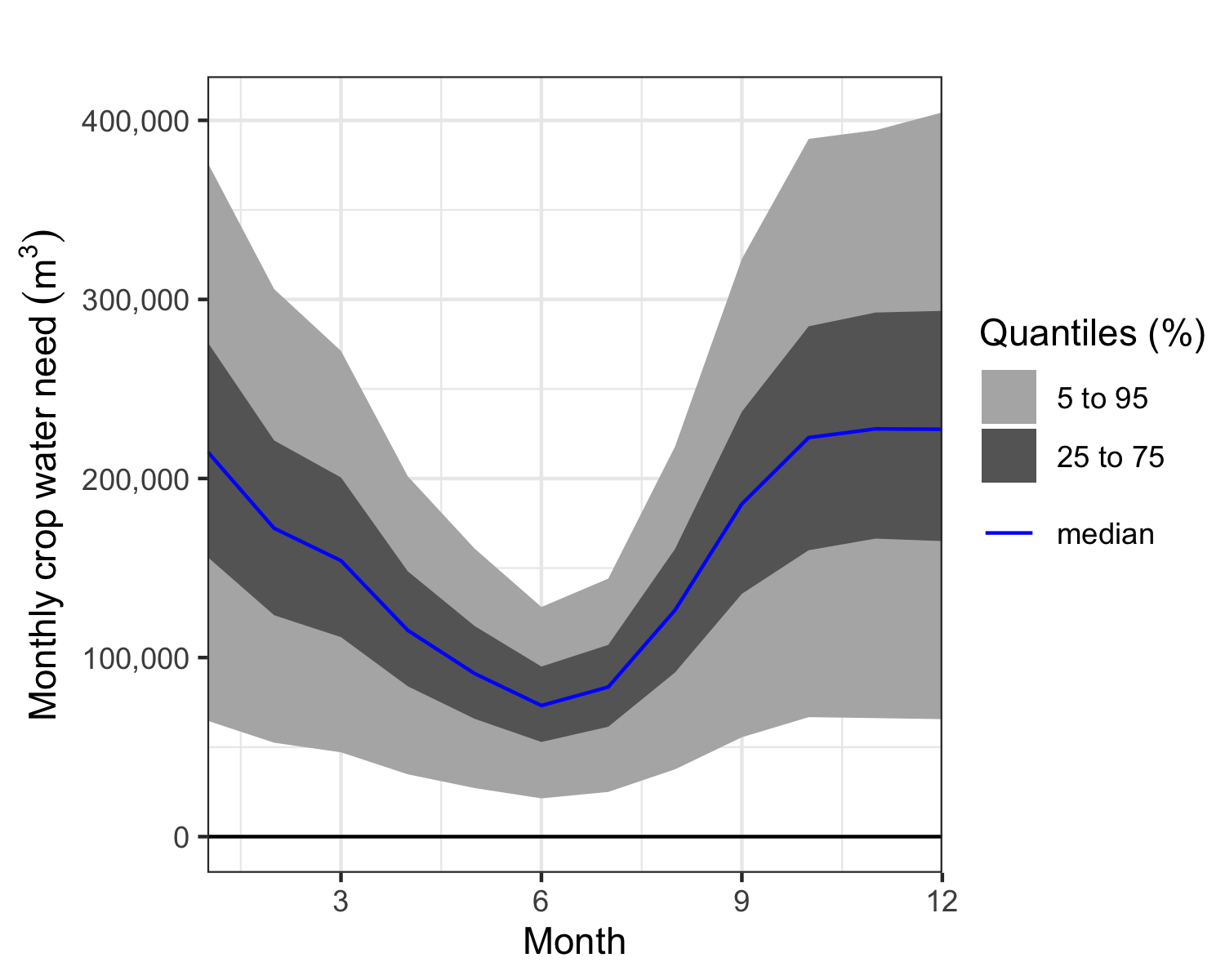
plotting\_simulations<-mcSimulation\_results  
  
decisionSupport::plot\_distributions(mcSimulation\_object = plotting\_simulations,  
 vars = c("yearly\_crop\_water\_need"),  
 method = 'smooth\_simple\_overlay',  
 x\_axis\_name = "1",  
 y\_axis\_name = "Distribution density",  
 base\_size = 13) + theme(legend.position = "none") + labs(x = expression(Total~annual~crop~water~need~(m^3)))



ggsave("figures/Fig\_1\_total\_annual\_crop\_water\_need.png", width=7, height=4)

Monthly crop water needs

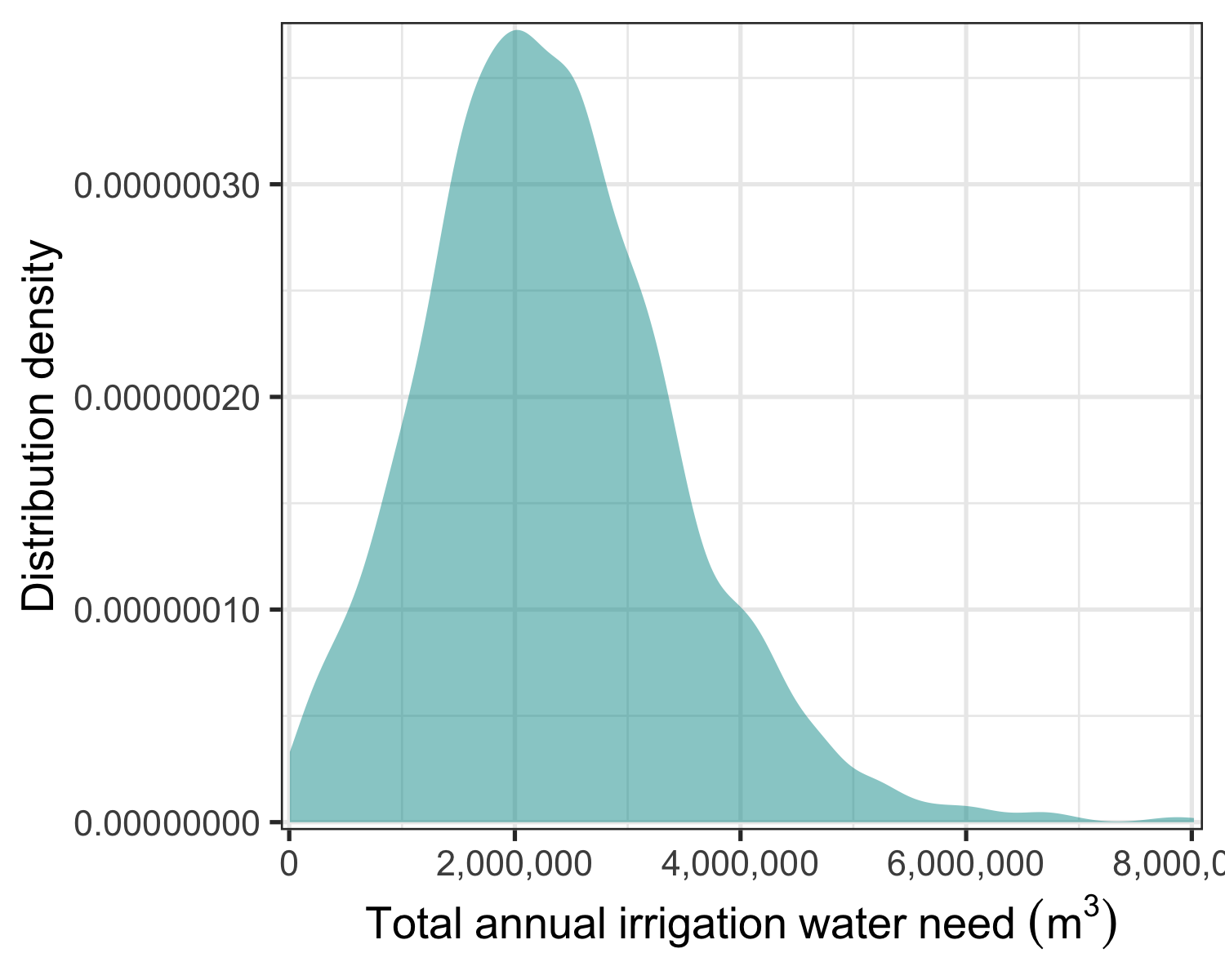
monthly\_crop\_water\_needs <- plot\_cashflow(mcSimulation\_object = plotting\_simulations,   
 cashflow\_var\_name = "cropwater\_need",  
 y\_axis\_name = "a",  
 x\_axis\_name = "Month",   
 facet\_labels = "") + labs(y = expression(Monthly~crop~water~need~(m^3)))  
  
monthly\_crop\_water\_needs



ggsave("figures/Fig\_2\_monthly\_crop\_water\_need.png", width=7, height=5)

Annual irrigation water needs

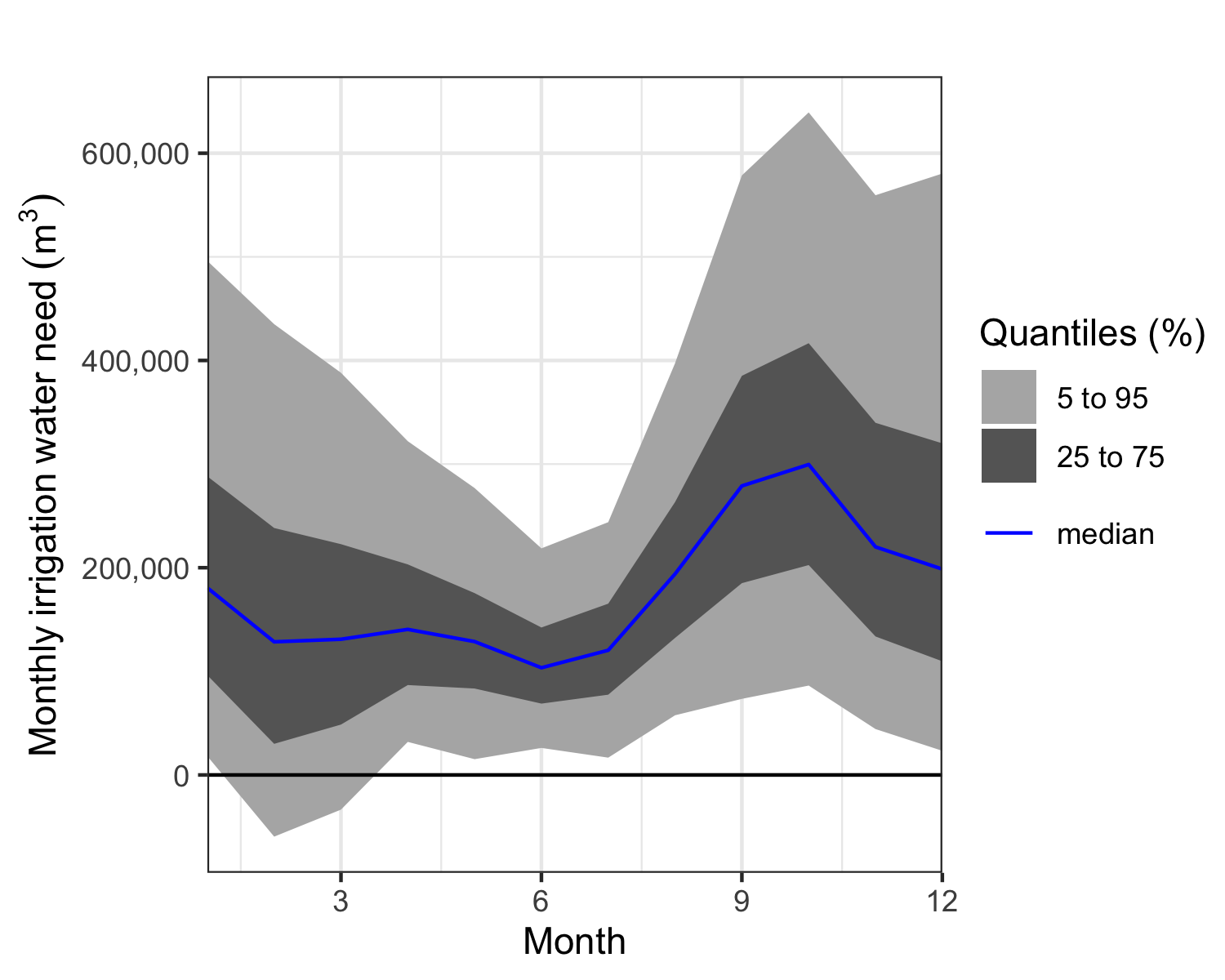
decisionSupport::plot\_distributions(mcSimulation\_object = plotting\_simulations,  
 vars = c("yearly\_irrigation\_water\_need"),  
 method = 'smooth\_simple\_overlay',  
 x\_axis\_name = "1",  
 y\_axis\_name = "Distribution density",  
 base\_size = 13) + theme(legend.position = "none") + labs(x = expression(Total~annual~irrigation~water~need~(m^3)))



ggsave("figures/Fig\_3\_total\_annual\_irrigation\_water\_need.png", width=7, height=4)

Monthly irrigation water needs

monthly\_irrigation\_water\_needs <- plot\_cashflow(mcSimulation\_object = plotting\_simulations,   
 cashflow\_var\_name = "irrigation\_water\_need",  
 y\_axis\_name = "a",  
 x\_axis\_name = "Month",   
 facet\_labels = "") + labs(y = expression(Monthly~irrigation~water~need~(m^3)))  
  
monthly\_irrigation\_water\_needs

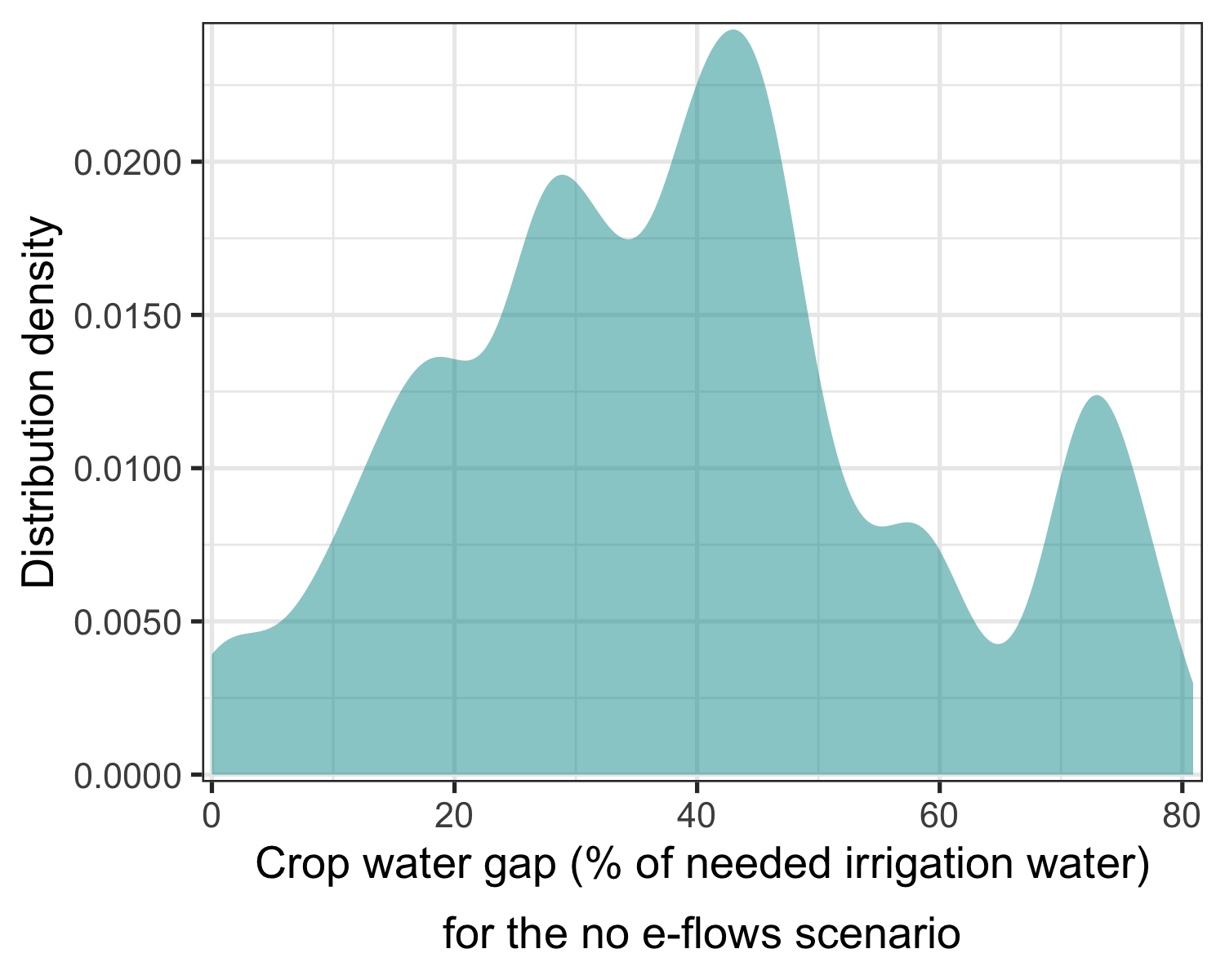


ggsave("figures/Fig\_4\_monthly\_irrigation\_water\_need.png", width=7, height=5)

### “No eflows” scenario

In the “No eflows” scenario, farmers currently face considerable periods of water shortages. On average across all months of the year, between 0 and 80% of crop water needs could not be met by irrigation water extracted from the river.

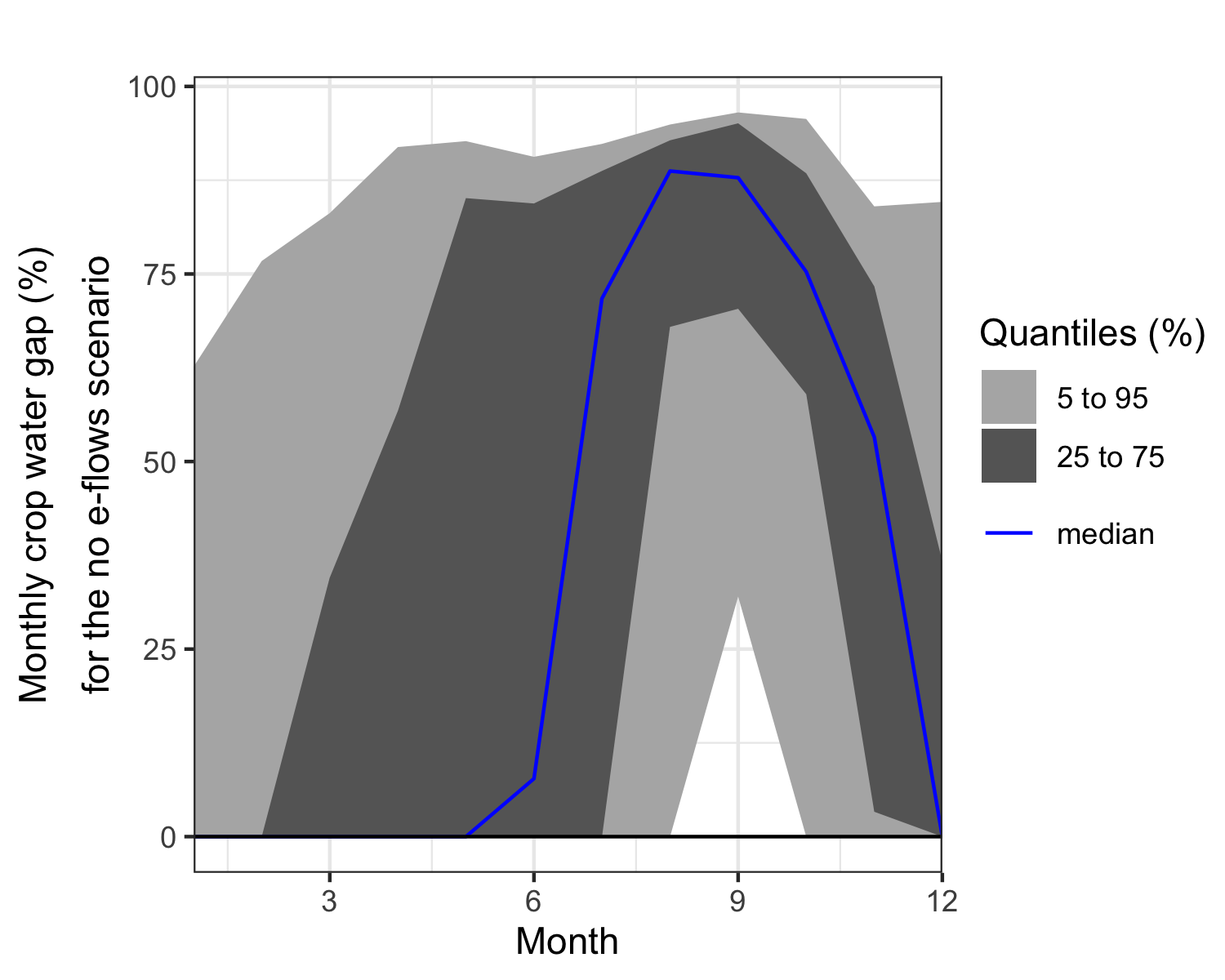
plotting\_simulations$y$scen1\_crop\_water\_gap<-plotting\_simulations$y$scen1\_crop\_water\_gap\*100  
  
decisionSupport::plot\_distributions(mcSimulation\_object = plotting\_simulations,  
 vars = c("scen1\_crop\_water\_gap"),  
 method = 'smooth\_simple\_overlay',  
 x\_axis\_name = expression(atop("Crop water gap (% of needed irrigation water)", paste("for the no e-flows scenario"))),  
 y\_axis\_name = "Distribution density",  
 base\_size = 13) + theme(legend.position = "none")



ggsave("figures/Fig\_5\_baseline\_crop\_water\_gap.png", width=7, height=4)  
  
# length(which(plotting\_simulations$y$scen1\_crop\_water\_gap==0))/length(plotting\_simulations$y$scen1\_crop\_water\_gap)  
#   
# length(which(plotting\_simulations$y$scen1\_crop\_water\_gap>50))/length(plotting\_simulations$y$scen1\_crop\_water\_gap)

The crop water gap varies across the months of the year, with greatest irrigation water shortfalls during the dry-season months.

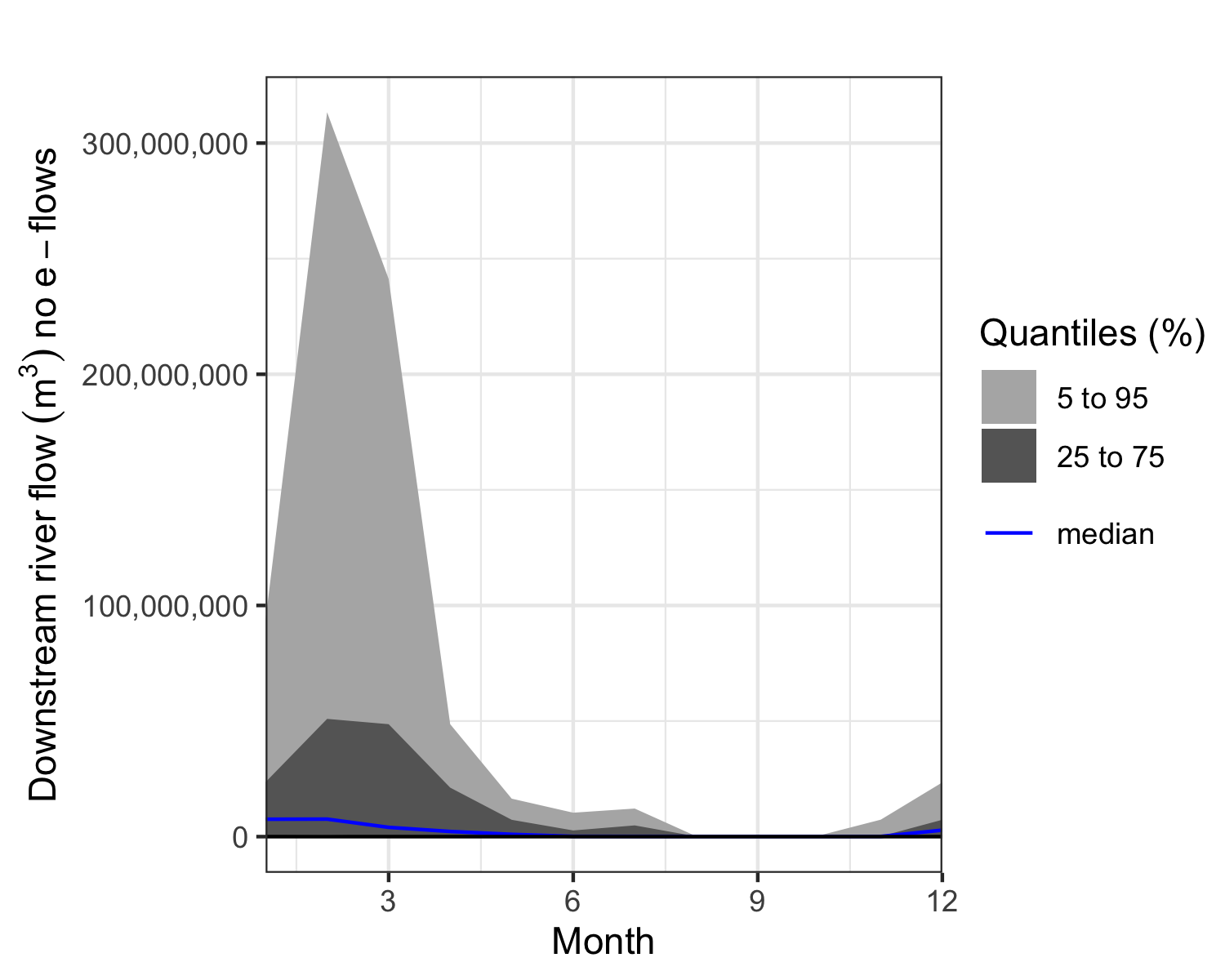
plotting\_simulations$y[,paste0("Crop\_water\_gap\_scen1\_",1:12)]<-plotting\_simulations$y[,paste0("Crop\_water\_gap\_scen1\_",1:12)]\*100  
  
monthly\_crop\_water\_gap\_baseline <- plot\_cashflow(mcSimulation\_object = plotting\_simulations,   
 cashflow\_var\_name = "Crop\_water\_gap\_scen1\_",  
 y\_axis\_name = "a",  
 x\_axis\_name = "Month",   
 facet\_labels = "") + labs(y = expression(atop("Monthly crop water gap (%)", paste("for the no e-flows scenario"))))  
  
monthly\_crop\_water\_gap\_baseline



ggsave("figures/Fig\_6\_monthly\_baseline\_crop\_water\_gap.png", width=7, height=5)

Downstream river flows also vary throughout the year.

downstream\_river\_flow\_baseline <- plot\_cashflow(mcSimulation\_object = plotting\_simulations,   
 cashflow\_var\_name = "Downstream\_river\_flow\_1\_",  
 y\_axis\_name = "a",  
 x\_axis\_name = "Month",   
 facet\_labels = "") + labs(y = expression(Downstream~river~flow~(m^3)~no~e-flows))  
  
downstream\_river\_flow\_baseline

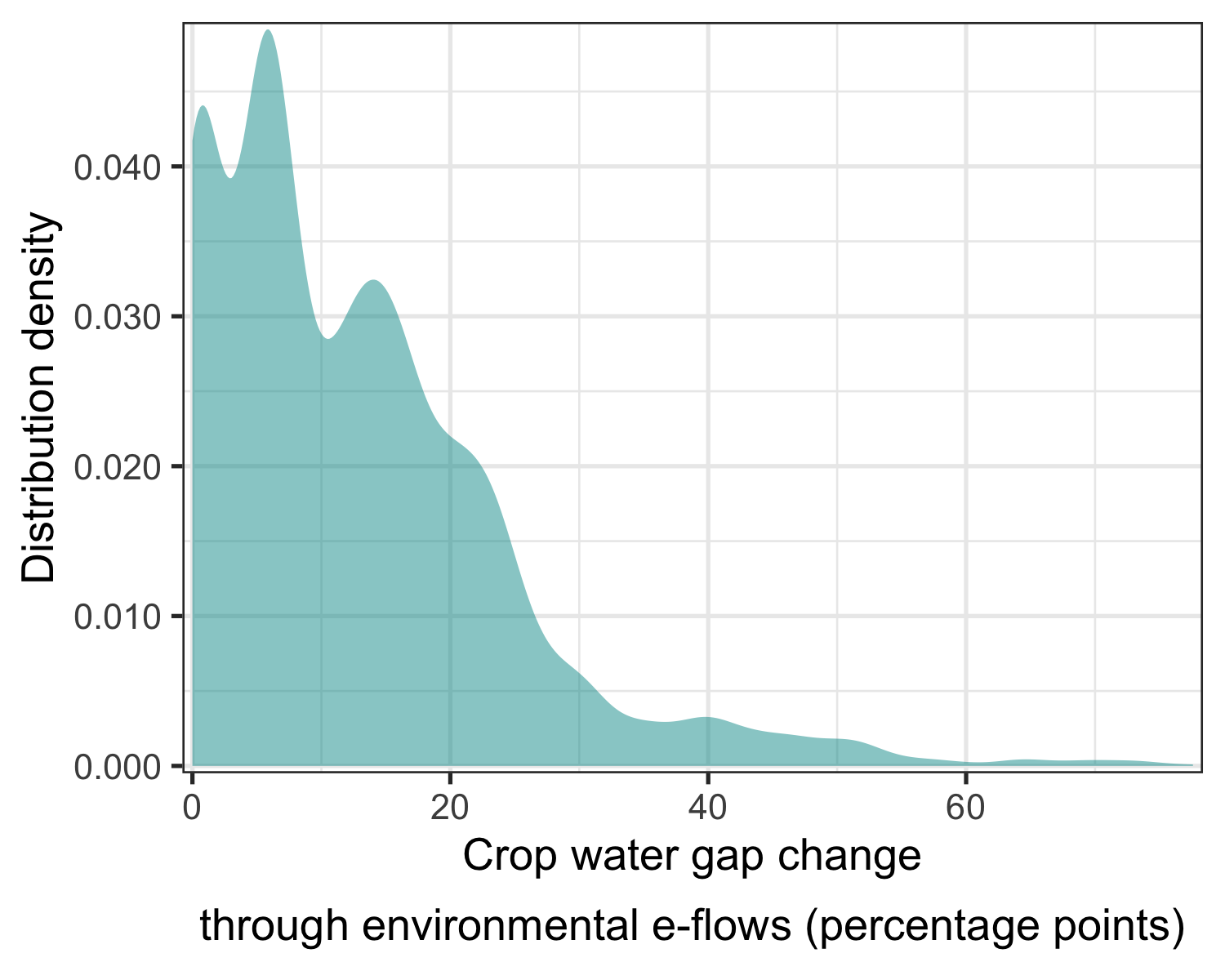


ggsave("figures/Fig\_7\_monthly\_downstream\_river\_flow.png", width=7, height=4)

### Environmental e-flows scenario

Here the distribution of the changes in the crop water gap through environmental e-flows.

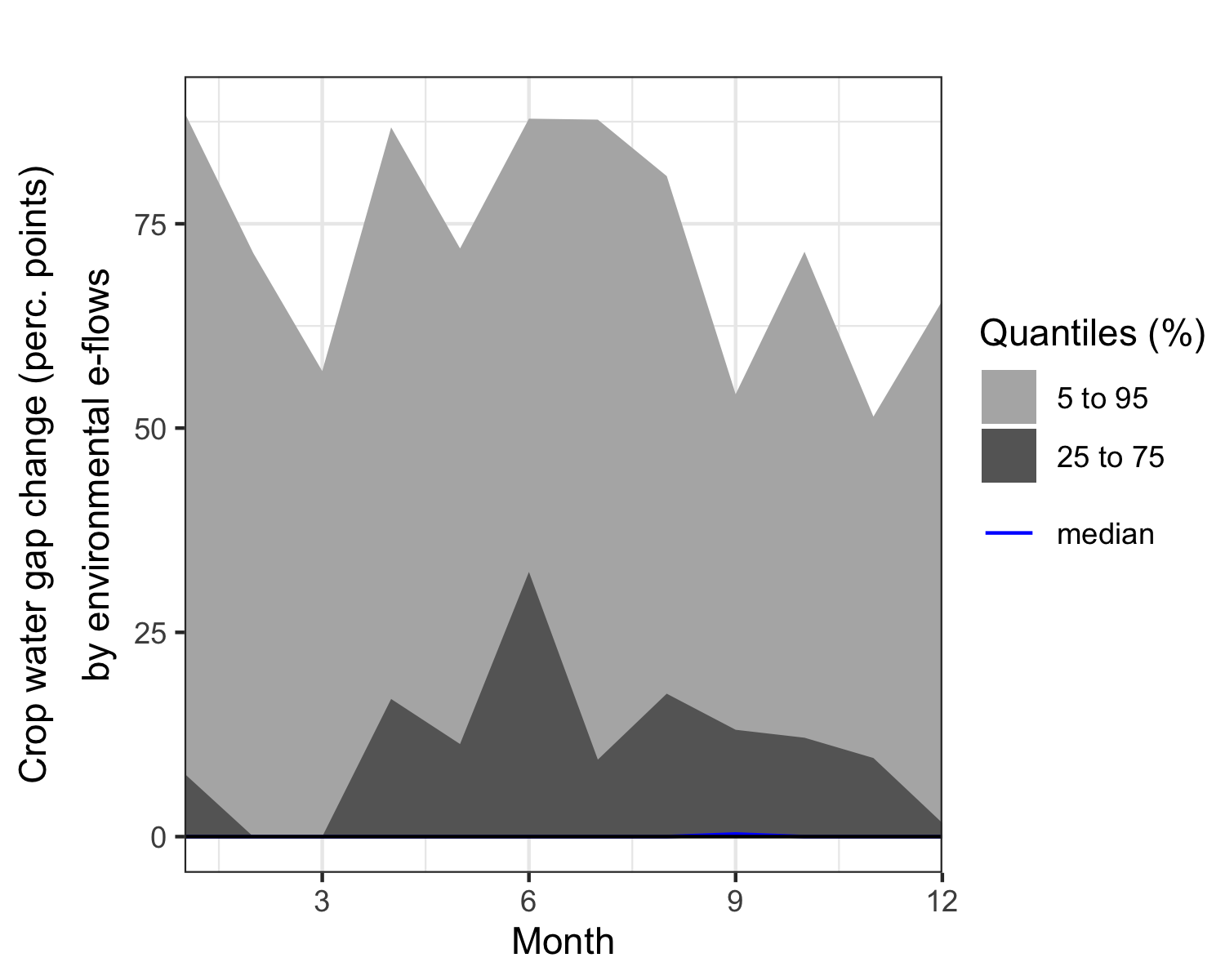
plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_2\_vs\_1<-plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_2\_vs\_1\*100  
  
decisionSupport::plot\_distributions(mcSimulation\_object = plotting\_simulations,  
 vars = c("Mean\_Crop\_water\_gap\_difference\_2\_vs\_1"),  
 method = 'smooth\_simple\_overlay',  
 x\_axis\_name = expression(atop("Crop water gap change", paste("through environmental e-flows (percentage points)"))),  
 y\_axis\_name = "Distribution density",  
 base\_size = 13) + theme(legend.position = "none")



ggsave("figures/Fig\_8\_environmental\_eflow\_policy\_effect\_crop\_water\_gap.png", width=7, height=4)  
  
# length(which(plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_2\_vs\_1<20))/length(plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_2\_vs\_1)

Here are the impacts of an environmentally focused e-flow policy on the crop water gap in each month.

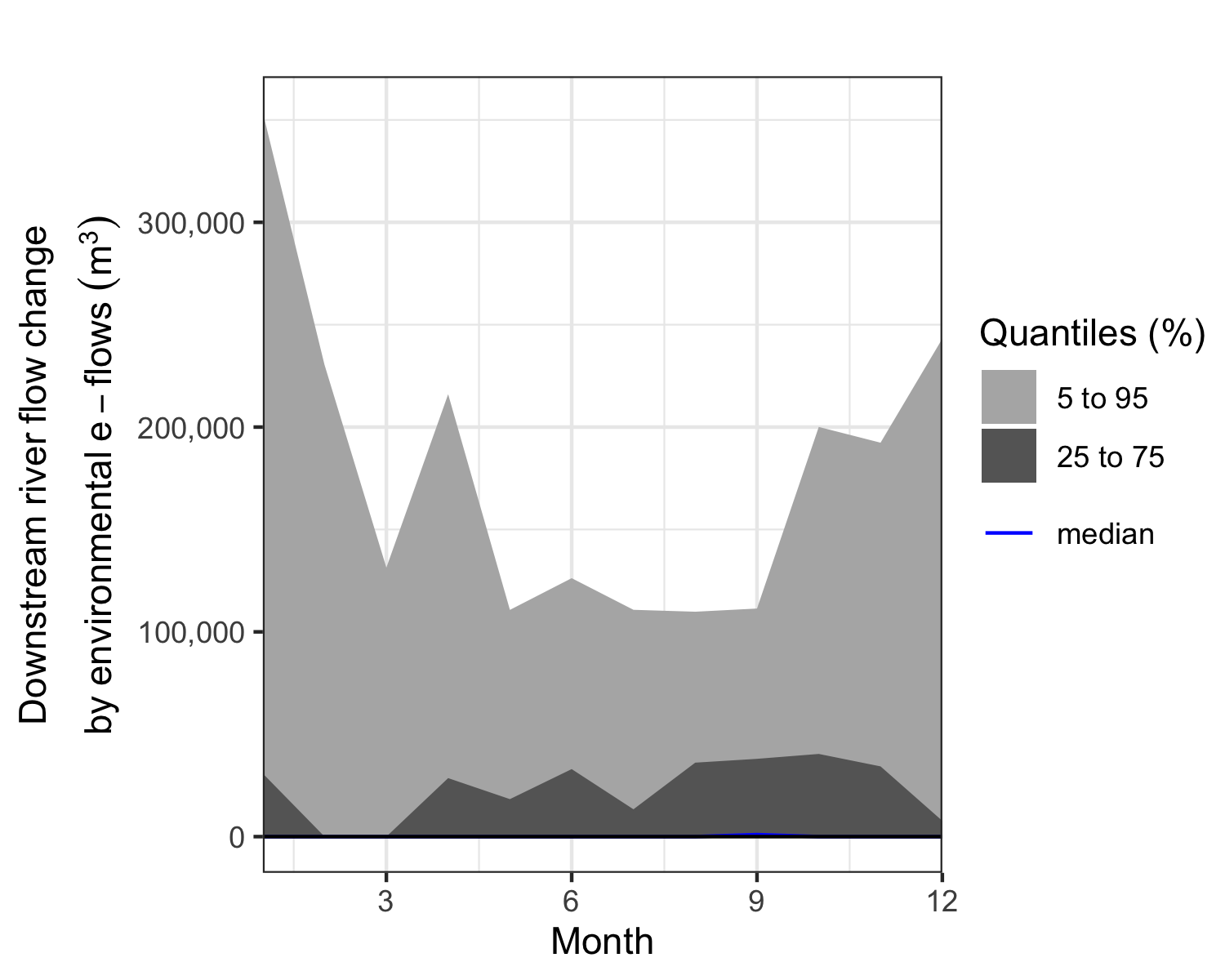
plotting\_simulations$y[,paste0("Crop\_water\_gap\_difference\_2\_vs\_1",1:12)]<-plotting\_simulations$y[,paste0("Crop\_water\_gap\_difference\_2\_vs\_1",1:12)]\*100  
  
monthly\_crop\_water\_gap\_environmental <- plot\_cashflow(mcSimulation\_object = plotting\_simulations,   
 cashflow\_var\_name = "Crop\_water\_gap\_difference\_2\_vs\_1",  
 y\_axis\_name = "a",  
 x\_axis\_name = "Month",   
 facet\_labels = "") + labs(y = expression(atop("Crop water gap change (perc. points)", paste("by environmental e-flows"))))  
  
monthly\_crop\_water\_gap\_environmental



ggsave("figures/Fig\_9\_monthly\_env\_eflow\_increase\_crop\_water\_gap.png", width=7, height=5)

Here’s the impact on streamflow:

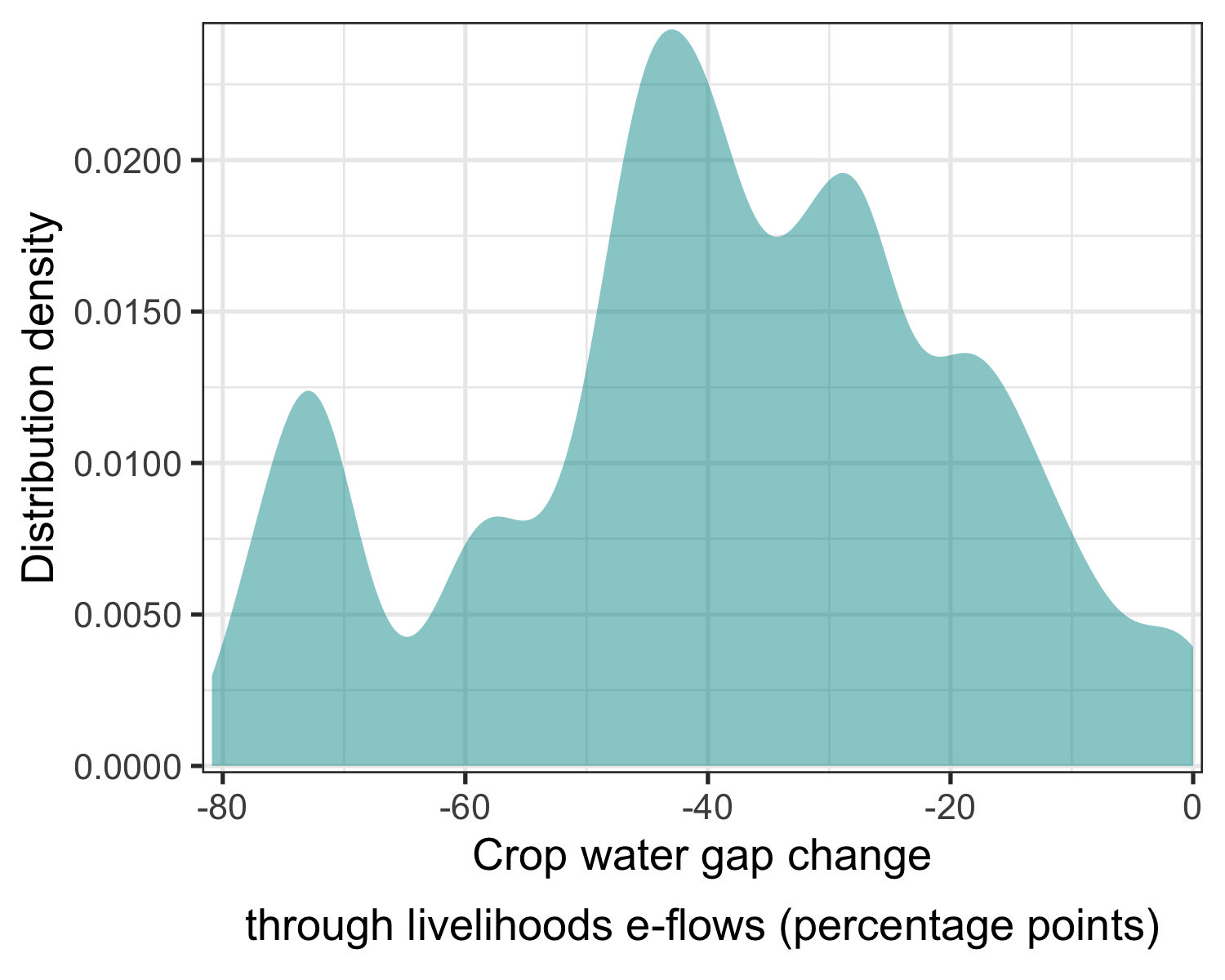
downstream\_river\_flow\_env\_flow <- plot\_cashflow(mcSimulation\_object = plotting\_simulations,   
 cashflow\_var\_name = "Downstream\_difference\_2\_vs\_1",  
 y\_axis\_name = "a",  
 x\_axis\_name = "Month",   
 facet\_labels = "") + labs(y = expression(atop(Downstream~river~flow~change, paste(by~environmental~e-flows~(m^3)))))  
  
downstream\_river\_flow\_env\_flow



ggsave("figures/Fig\_10\_monthly\_change\_in\_downstream\_river\_flow\_env\_eflows.png", width=7, height=5)

### “Livelihoods” e-flow scenario

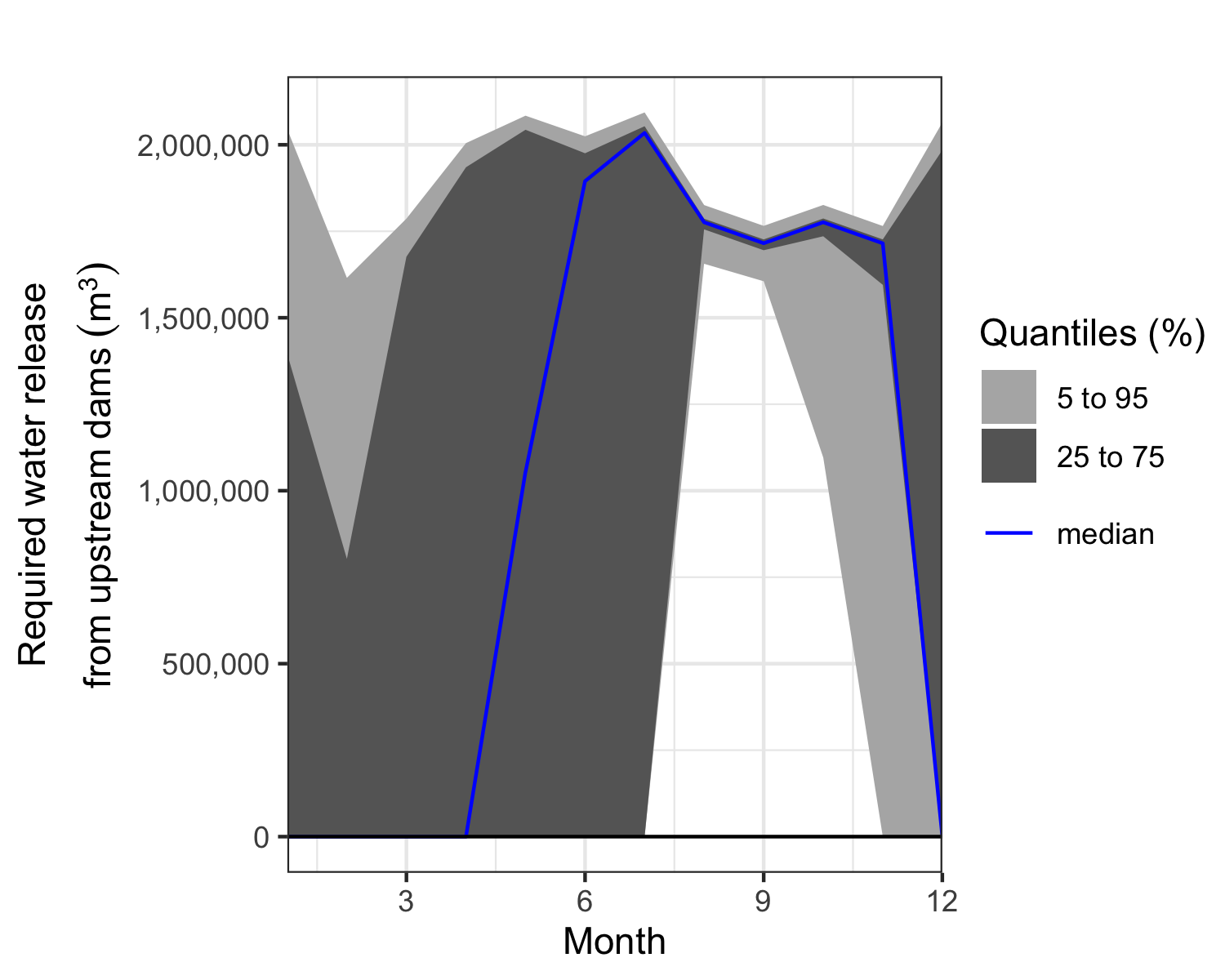
plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_3\_vs\_1<-plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_3\_vs\_1\*100  
  
decisionSupport::plot\_distributions(mcSimulation\_object = plotting\_simulations,  
 vars = c("Mean\_Crop\_water\_gap\_difference\_3\_vs\_1"),  
 method = 'smooth\_simple\_overlay',  
 x\_axis\_name = "",  
 y\_axis\_name = "Distribution density",  
 base\_size = 13) +   
 theme(legend.position = "none") +   
 labs(x = expression(atop("Crop water gap change", paste("through livelihoods e-flows (percentage points)"))))



ggsave("figures/Fig\_11\_livelihoods\_eflow\_policy\_effect\_crop\_water\_gap.png", width=7, height=4)  
  
# length(which(plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_3\_vs\_1<20))/length(plotting\_simulations$y$Mean\_Crop\_water\_gap\_difference\_3\_vs\_1)

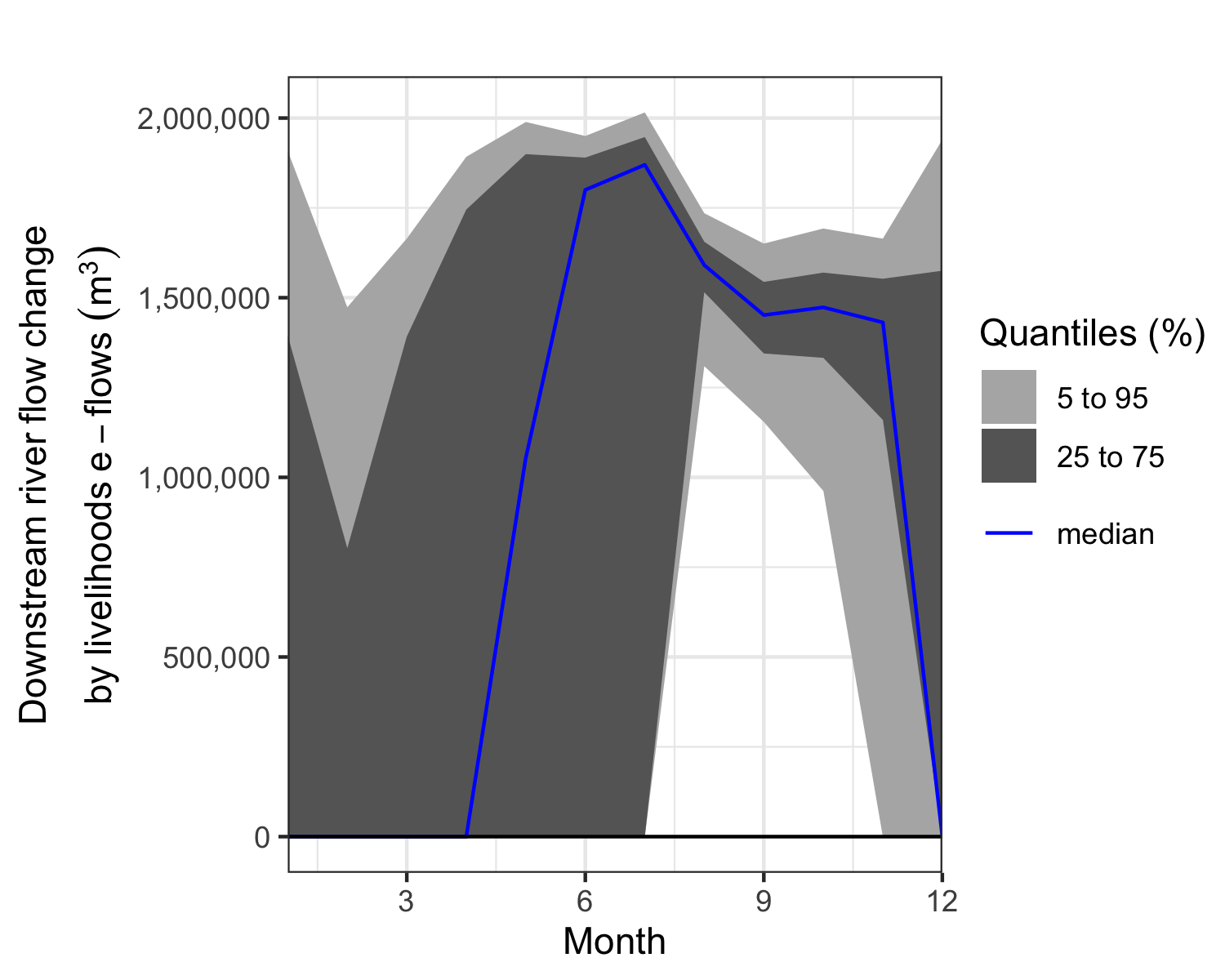
Here are the impacts of an environmentally focused e-flow policy on the crop water gap in each month.

monthly\_dam\_release <- plot\_cashflow(mcSimulation\_object = plotting\_simulations,   
 cashflow\_var\_name = "scen3\_dam\_release",  
 y\_axis\_name = "a",  
 x\_axis\_name = "Month",   
 facet\_labels = "") + labs(y = expression(atop(Required~water~release, paste(from~upstream~dams~(m^3)))))  
  
monthly\_dam\_release



ggsave("figures/Fig\_12\_required\_dam\_release.png", width=7, height=5)

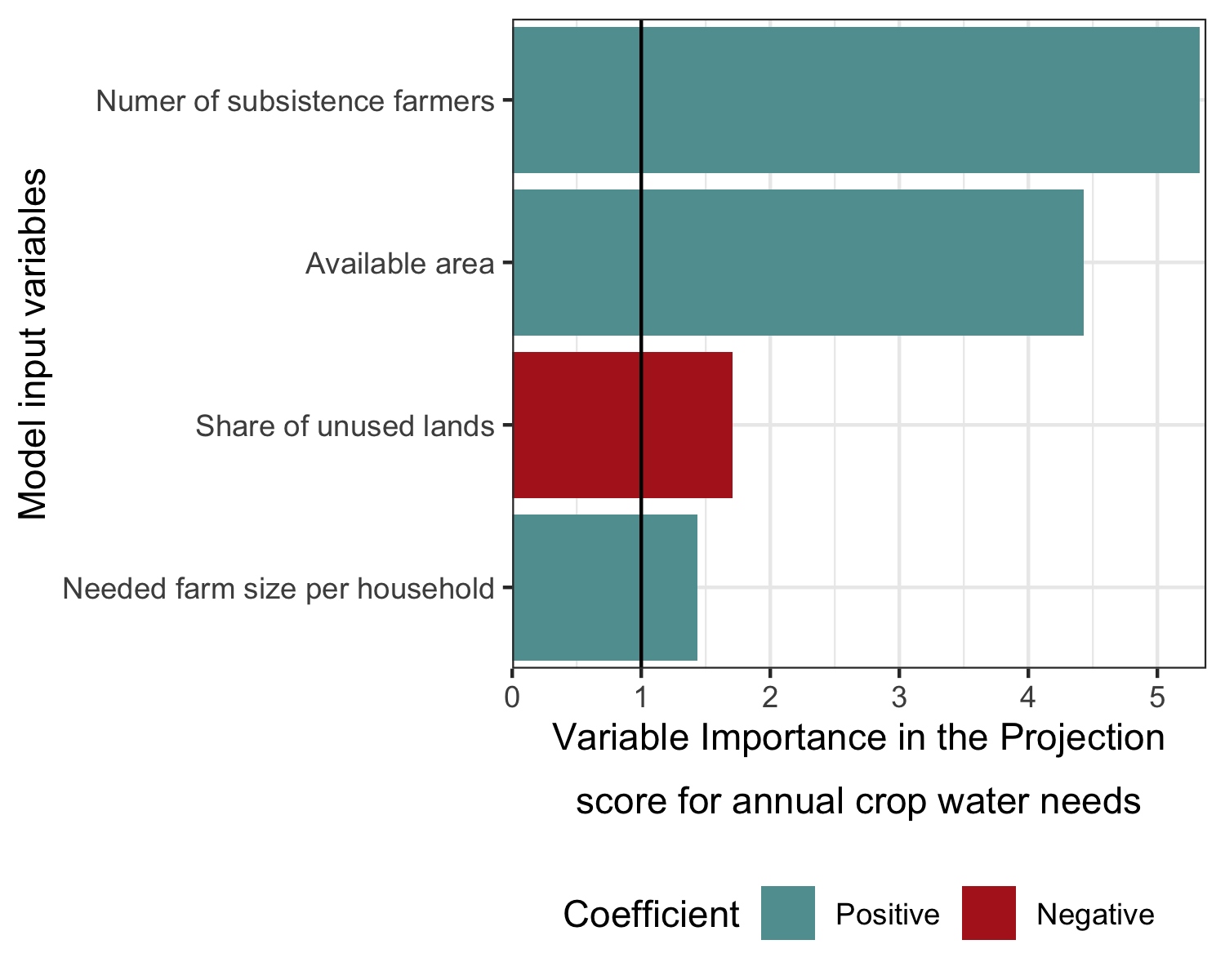
Here’s the impact on streamflow:



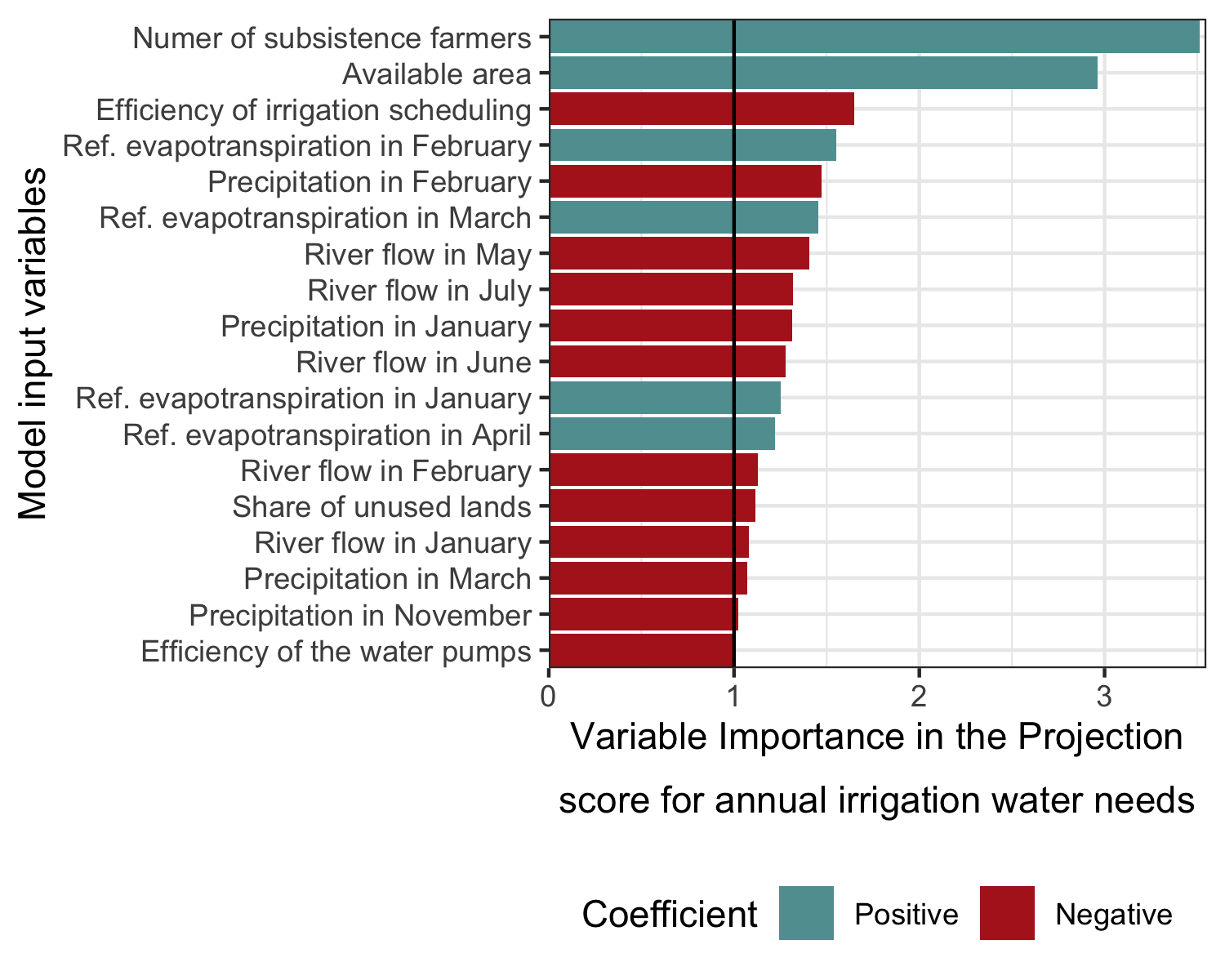
## Sensitivity analysis

We use the plsr.mcSimulation function of the decisionSupport package to run Partial Least Squares regression on the model outputs. We use the Variable Importance in the Projection (VIP) scores to identify important variables.

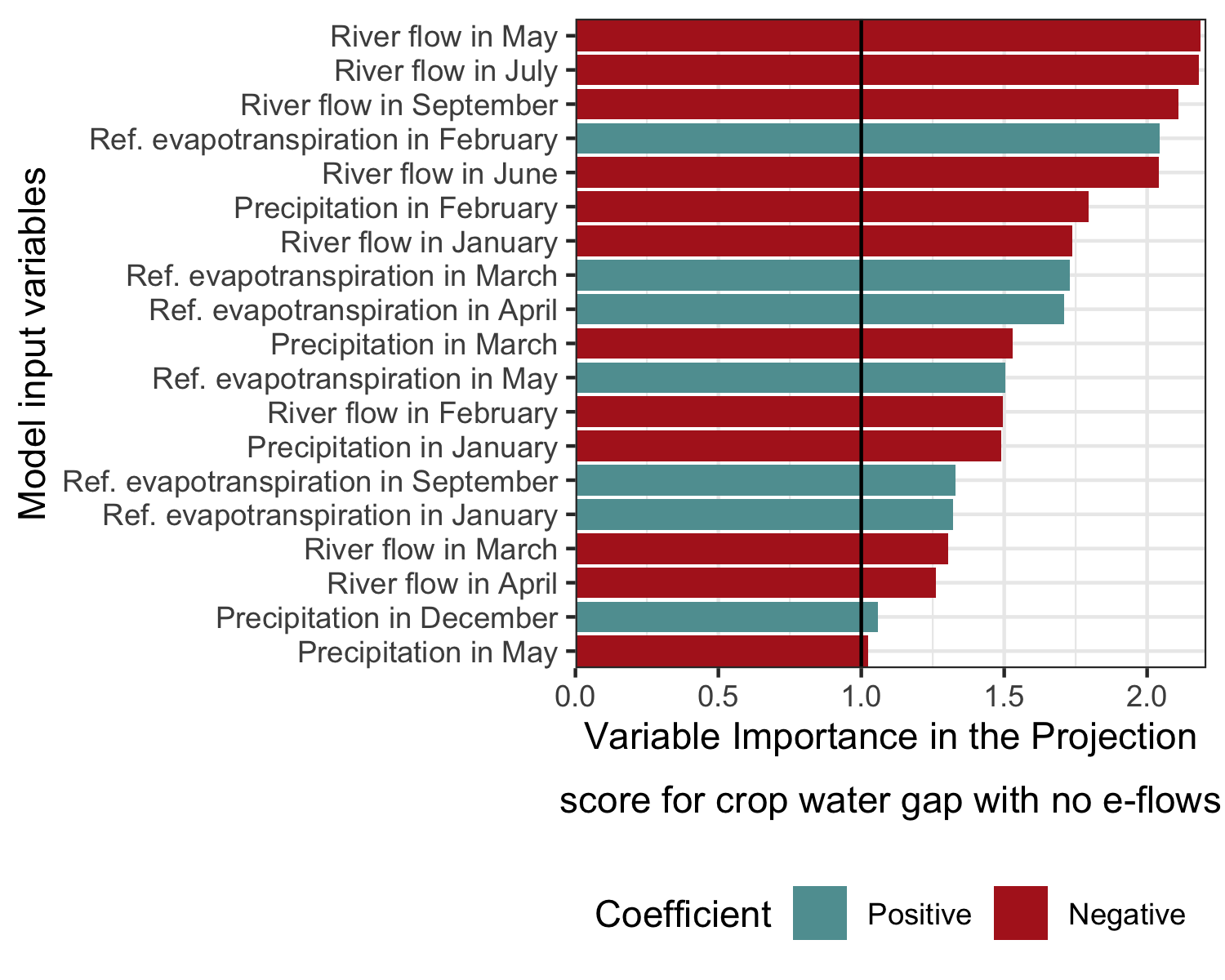
### Crop water needs



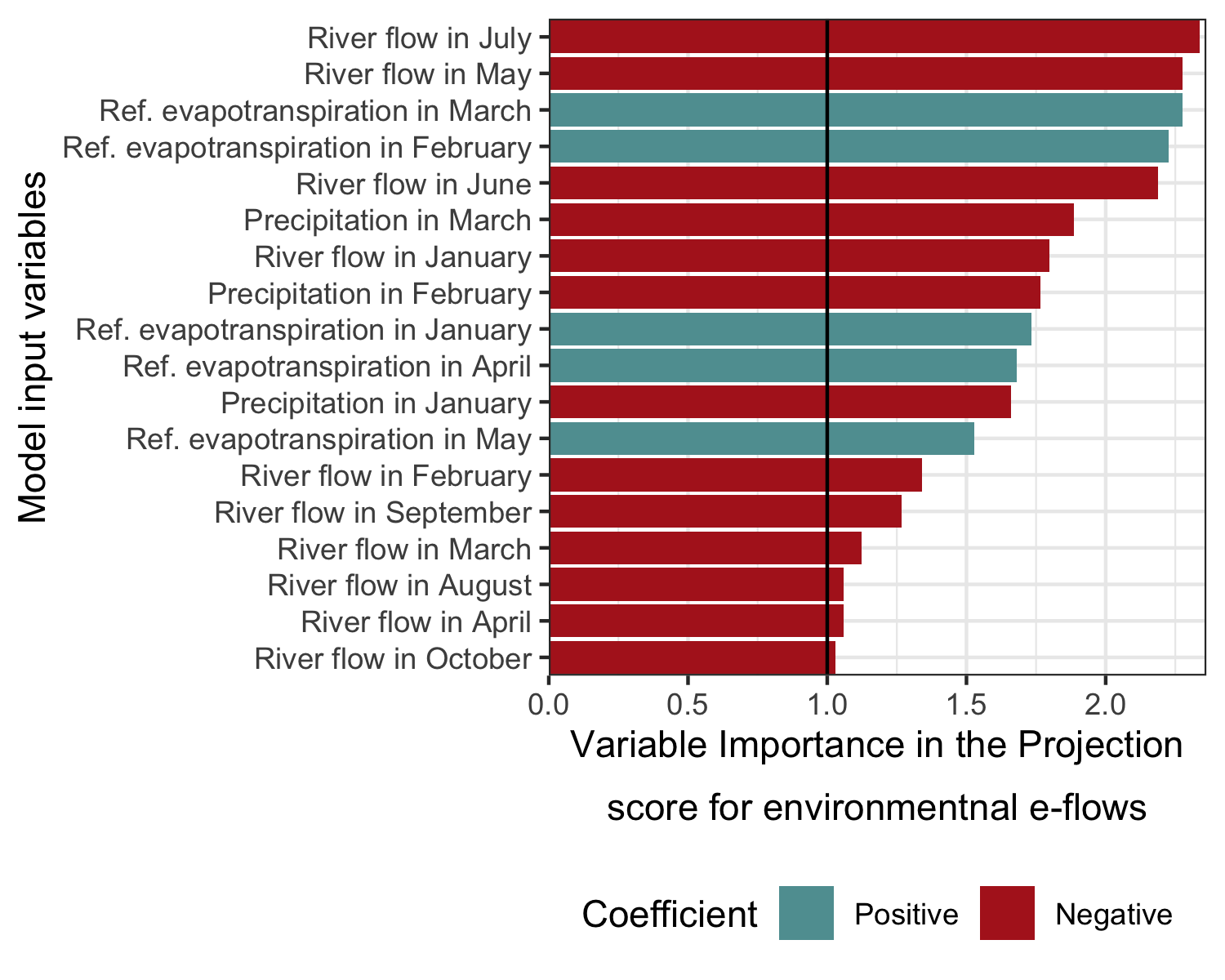
### Irrigation demand



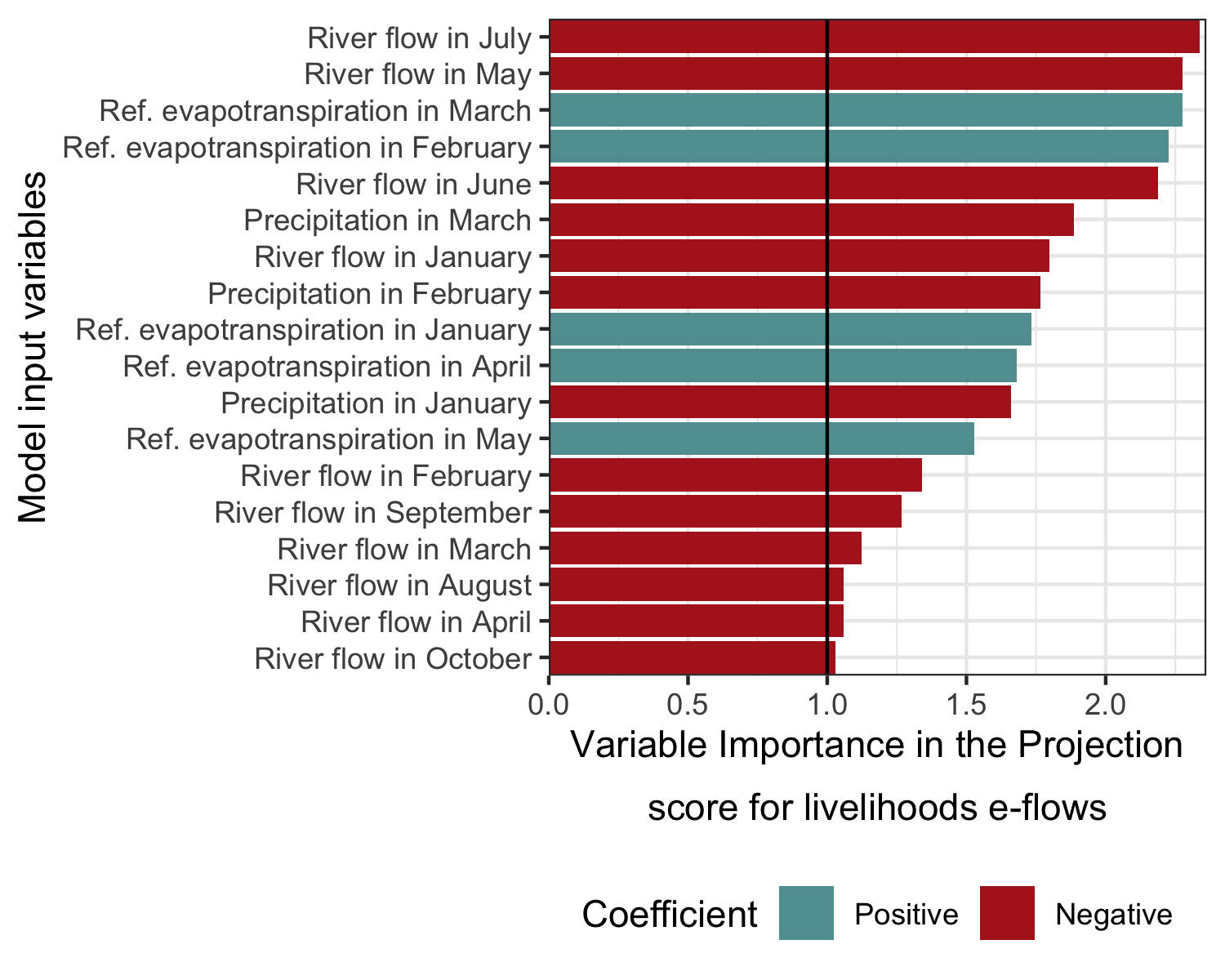
### No eflows



### Environmental e-flows



### Livelihoods e-flows



More information on all these procedures is contained in the [decisionSupport manual](https://cran.r-project.org/web/packages/decisionSupport/decisionSupport.pdf), especially under welfareDecisionAnalysis. This document was generated using the rmarkdown (Allaire et al. 2021) and knitr (Xie 2021) packages in the R programming language (R Core Team 2021).

### Estimate values

This table contains the estimate values used for the Monte Carlo analysis

| Description | variable | distribution | lower | upper | label |
| --- | --- | --- | --- | --- | --- |
| Precipitation in month 1 | prec\_1 | posnorm | 45.00 | 135.00 | Precipitation in January |
| Precipitation in month 2 | prec\_2 | posnorm | 31.00 | 93.00 | Precipitation in February |
| Precipitation in month 3 | prec\_3 | posnorm | 25.00 | 75.00 | Precipitation in March |
| Precipitation in month 4 | prec\_4 | posnorm | 12.50 | 37.50 | Precipitation in April |
| Precipitation in month 5 | prec\_5 | posnorm | 5.00 | 15.00 | Precipitation in May |
| Precipitation in month 6 | prec\_6 | posnorm | 1.00 | 3.00 | Precipitation in June |
| Precipitation in month 7 | prec\_7 | posnorm | 2.00 | 6.00 | Precipitation in July |
| Precipitation in month 8 | prec\_8 | posnorm | 3.00 | 9.00 | Precipitation in August |
| Precipitation in month 9 | prec\_9 | posnorm | 7.00 | 21.00 | Precipitation in September |
| Precipitation in month 10 | prec\_10 | posnorm | 12.50 | 37.50 | Precipitation in October |
| Precipitation in month 11 | prec\_11 | posnorm | 35.00 | 105.00 | Precipitation in November |
| Precipitation in month 12 | prec\_12 | posnorm | 45.00 | 135.00 | Precipitation in December |
|  |  |  | NA | NA |  |
| Reference evapotranspiration (ET0) mm/per ha month 1 (Hargreaves Samani equation with nasapower package) | ET0\_1 | posnorm | 144.00 | 240.00 | Ref. evapotranspiration in January |
| Reference evapotranspiration (ET0) mm/per ha month 2 | ET0\_2 | posnorm | 114.75 | 191.25 | Ref. evapotranspiration in February |
| Reference evapotranspiration (ET0) mm/per ha month 3 | ET0\_3 | posnorm | 96.00 | 160.00 | Ref. evapotranspiration in March |
| Reference evapotranspiration (ET0) mm/per ha month 4 | ET0\_4 | posnorm | 67.50 | 112.50 | Ref. evapotranspiration in April |
| Reference evapotranspiration (ET0) mm/per ha month 5 | ET0\_5 | posnorm | 52.50 | 87.50 | Ref. evapotranspiration in May |
| Reference evapotranspiration (ET0) mm/per ha month 6 | ET0\_6 | posnorm | 32.25 | 53.75 | Ref. evapotranspiration in June |
| Reference evapotranspiration (ET0) mm/per ha month 7 | ET0\_7 | posnorm | 40.50 | 67.50 | Ref. evapotranspiration in July |
| Reference evapotranspiration (ET0) mm/per ha month 8 | ET0\_8 | posnorm | 52.50 | 87.50 | Ref. evapotranspiration in August |
| Reference evapotranspiration (ET0) mm/per ha month 9 | ET0\_9 | posnorm | 71.25 | 118.75 | Ref. evapotranspiration in September |
| Reference evapotranspiration (ET0) mm/per ha month 10 | ET0\_10 | posnorm | 99.75 | 166.25 | Ref. evapotranspiration in October |
| Reference evapotranspiration (ET0) mm/per ha month 11 | ET0\_11 | posnorm | 126.00 | 210.00 | Ref. evapotranspiration in November |
| Reference evapotranspiration (ET0) mm/per ha month 12 | ET0\_12 | posnorm | 145.50 | 242.50 | Ref. evapotranspiration in December |
|  |  |  | NA | NA |  |
| Crop coefficient in month 1 | kc\_1 | posnorm | 0.90 | 1.00 | kc\_1 |
| Crop coefficient in month 2 | kc\_2 | posnorm | 0.90 | 1.00 | kc\_2 |
| Crop coefficient in month 3 | kc\_3 | posnorm | 0.90 | 1.00 | kc\_3 |
| Crop coefficient in month 4 | kc\_4 | posnorm | 0.90 | 1.00 | kc\_4 |
| Crop coefficient in month 5 | kc\_5 | posnorm | 0.90 | 1.00 | kc\_5 |
| Crop coefficient in month 6 | kc\_6 | posnorm | 0.90 | 1.00 | kc\_6 |
| Crop coefficient in month 7 | kc\_7 | posnorm | 0.90 | 1.00 | kc\_7 |
| Crop coefficient in month 8 | kc\_8 | posnorm | 0.90 | 1.00 | kc\_8 |
| Crop coefficient in month 9 | kc\_9 | posnorm | 0.90 | 1.00 | kc\_9 |
| Crop coefficient in month 10 | kc\_10 | posnorm | 0.90 | 1.00 | kc\_10 |
| Crop coefficient in month 11 | kc\_11 | posnorm | 0.90 | 1.00 | kc\_11 |
| Crop coefficient in month 12 | kc\_12 | posnorm | 0.90 | 1.00 | kc\_12 |
|  |  |  | NA | NA |  |
| Effective rainfall - minimum threshold | effprec\_low | posnorm | 5.00 | 10.00 | effprec\_low |
| Effective rainfall - maximum threshold | effprec\_high | posnorm | 90.00 | 200.00 | effprec\_high |
|  |  |  | NA | NA |  |
| Efficiency of water pumps | effi\_pump | tnorm\_0\_1 | 0.70 | 0.90 | Efficiency of the water pumps |
| Efficiency of irrigation scheduling and allocation | effi\_sched | tnorm\_0\_1 | 0.60 | 0.90 | Efficiency of irrigation scheduling |
| Coefficient of variation, ratio of the standard deviation to the mean (a measure of relative variability). | var\_CV | posnorm | 5.00 | 20.00 | var\_CV |
|  |  |  | NA | NA |  |
| Total irrigable area | available\_area | posnorm | 100.00 | 300.00 | Available area |
| Share of land that is not used because of socio-political obstacles | unused\_sociopolit | tnorm\_0\_1 | 0.20 | 0.40 | Share of unused lands |
| Number of subsistence households | n\_subsistence\_farmers | posnorm | 30.00 | 200.00 | Numer of subsistence farmers |
| Farm size per subsistence households | necessary\_farm\_size\_per\_household | posnorm | 1.50 | 2.50 | Needed farm size per household |
|  |  |  | NA | NA |  |
| eflow in month 1 | eflow\_1 | posnorm | 1658637.36 | 2487956.04 | eflow\_1 |
| eflow in month 2 | eflow\_2 | posnorm | 1953364.40 | 2930046.59 | eflow\_2 |
| eflow in month 3 | eflow\_3 | posnorm | 2172764.83 | 3259147.25 | eflow\_3 |
| eflow in month 4 | eflow\_4 | posnorm | 5094152.71 | 7641229.07 | eflow\_4 |
| eflow in month 5 | eflow\_5 | posnorm | 12093593.23 | 18140389.85 | eflow\_5 |
| eflow in month 6 | eflow\_6 | posnorm | 4593467.28 | 6890200.92 | eflow\_6 |
| eflow in month 7 | eflow\_7 | posnorm | 2895912.09 | 4343868.13 | eflow\_7 |
| eflow in month 8 | eflow\_8 | posnorm | 2484366.68 | 3726550.02 | eflow\_8 |
| eflow in month 9 | eflow\_9 | posnorm | 2173592.97 | 3260389.45 | eflow\_9 |
| eflow in month 10 | eflow\_10 | posnorm | 2052485.78 | 3078728.68 | eflow\_10 |
| eflow in month 11 | eflow\_11 | posnorm | 1670297.91 | 2505446.86 | eflow\_11 |
| eflow in month 12 | eflow\_12 | posnorm | 1419171.87 | 2128757.80 | eflow\_12 |
|  |  |  | NA | NA |  |
| Minimum river flow that allows running the pumps (in m3/month) | minimum\_flow\_to\_operate\_pumps | posnorm | 50000.00 | 150000.00 | Minimum flow required by pumps |
|  |  |  | NA | NA |  |
| river flow in month 1 (Taken from base flow MCM data from 1920 to 2010 (Letaba River at EWR site EWR4 (Letaba Ranch upstream Little Letaba confluence) )) | river\_flow\_1 | posnorm | 3289641.29 | 14884566.58 | River flow in January |
| river flow in month 2 | river\_flow\_2 | posnorm | 3552190.55 | 28211390.25 | River flow in February |
| river flow in month 3 | river\_flow\_3 | posnorm | 3629341.05 | 24557111.18 | River flow in March |
| river flow in month 4 | river\_flow\_4 | posnorm | 3593958.87 | 18063311.23 | River flow in April |
| river flow in month 5 | river\_flow\_5 | posnorm | 3506617.70 | 11756278.83 | River flow in May |
| river flow in month 6 | river\_flow\_6 | posnorm | 3448532.21 | 8821373.46 | River flow in June |
| river flow in month 7 | river\_flow\_7 | posnorm | 3270609.32 | 7597819.59 | River flow in July |
| river flow in month 8 | river\_flow\_8 | posnorm | 2770310.63 | 6595355.44 | River flow in August |
| river flow in month 9 | river\_flow\_9 | posnorm | 2475234.52 | 5976080.25 | River flow in September |
| river flow in month 10 | river\_flow\_10 | posnorm | 2195340.50 | 5425988.65 | River flow in October |
| river flow in month 11 | river\_flow\_11 | posnorm | 2306113.10 | 6163707.61 | River flow in November |
| river flow in month 12 | river\_flow\_12 | posnorm | 2699506.90 | 7293206.41 | River flow in December |
|  |  |  | NA | NA |  |
| livestock water need per month | livestock\_water\_need | posnorm | 300.00 | 2000.00 | livestock\_water\_need |

### References

Allaire, JJ, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, Winston Chang, and Richard Iannone. 2021. *Rmarkdown: Dynamic Documents for r*. <https://CRAN.R-project.org/package=rmarkdown>.

Do, Hoa, Eike Luedeling, and Cory Whitney. 2020. “Decision Analysis of Agroforestry Options Reveals Adoption Risks for Resource-Poor Farmers.” *Agronomy for Sustainable Development* 40 (3): 20. <https://doi.org/10.1007/s13593-020-00624-5>.

Guo, Danlu, Seth Westra, and Tim Peterson. 2020. *Evapotranspiration: Modelling Actual, Potential and Reference Crop Evapotranspiration*. <https://CRAN.R-project.org/package=Evapotranspiration>.

Lanzanova, Denis, Cory Whitney, Keith Shepherd, and Eike Luedeling. 2019. “Improving Development Efficiency Through Decision Analysis: Reservoir Protection in Burkina Faso.” *Environmental Modelling & Software* 115 (May): 164–75. <https://doi.org/10.1016/j.envsoft.2019.01.016>.

Luedeling, Eike, Lutz Goehring, Katja Schiffers, Cory Whitney, and Eduardo Fernandez. 2021. *decisionSupport: Quantitative Support of Decision Making Under Uncertainty*. <http://www.worldagroforestry.org/>.

Pedersen, Thomas Lin. 2020. *Patchwork: The Composer of Plots*. <https://CRAN.R-project.org/package=patchwork>.

R Core Team. 2021. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.

Sparks, Adam H. 2021. *Nasapower: NASA POWER API Client*. <https://docs.ropensci.org/nasapower/>.

Whitney, Cory, D. Lanzanova, C. Muchiri, K. Shepherd, T. Rosenstock, M. Krawinkel, J. R. S. Tabuti, and E. Luedeling. 2018. “Probabilistic Decision Tools for Determining Impacts of Agricultural Development Policy on Household Nutrition.” *Earth’s Future* 6 (3): 359–72. <https://doi.org/10.1002/2017EF000765/full>.

Whitney, C., K. Shepherd, and E. Luedeling. 2018. “Decision Analysis Methods Guide; Agricultural Policy for Nutrition.” *World Agroforestry (ICRAF)* Working Paper series (275): 40.

Wickham, Hadley. 2020. *Plyr: Tools for Splitting, Applying and Combining Data*. <https://CRAN.R-project.org/package=plyr>.

———. 2021. *Tidyverse: Easily Install and Load the Tidyverse*. <https://CRAN.R-project.org/package=tidyverse>.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D’Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. “Welcome to the tidyverse.” *Journal of Open Source Software* 4 (43): 1686. <https://doi.org/10.21105/joss.01686>.

Wickham, Hadley, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. 2021. *Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. <https://CRAN.R-project.org/package=ggplot2>.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2021. *Dplyr: A Grammar of Data Manipulation*. <https://CRAN.R-project.org/package=dplyr>.

Xie, Yihui. 2021. *Knitr: A General-Purpose Package for Dynamic Report Generation in r*. <https://yihui.org/knitr/>.