```
rm(list=ls())
# # library(stringr)
# # library(tidyr)
# # library(readtext)
# # library(XML)
# # library(processx)
# # library(dplyr)
# # library(openxlsx)
#
# # setwd("/Users/alfredo/Library/CloudStorage/OneDrive-
UniversidaddeCastilla-LaMancha/Drive/UCLM ADE/Investigacion Operativa
(2023-2024)/Tema 4 - Optimización/")
# #
# #
# # install.packages("quadprog")
# # library(quadprog)
# #
# # # Coeficientes cuadráticos
# # D <- matrix(c(4, -2, -2, 10), nrow = 2, byrow = TRUE)
# #
# # # Coeficientes lineales
# # d <- c(-6, -6)
# #
# # # Coeficientes de restricciones
# # A <- matrix(c(-1, -1, 1, -2), nrow = 2, byrow = TRUE)
# #
# # # Términos constantes en restricciones
# # b <- c(-2, -1)
# #
# # result <- solve.QP(Dmat = D, dvec = d, Amat = A, bvec = b, meq = 0)
# # result
# #
#
#
#
#
# # Ejercicio resuelto
#
# # Función objetivo a maximizar
# obj_func <- function(x) {
#   return(-(log(x[1]) + log(x[2])))
# }
#
# # Restricciones
# const_func1 <- function(x) {
#   return(10 * x[1] + 5 * x[2] - 350)
# }
#
# const_func2 <- function(x) {
#   return(0.1 * x[1] + 0.2 * x[2] - 8)
# }
# # Punto de inicio
# initial_point <- c(1, 1)
#
```

```
# # Resolviendo el problema de optimización con restricciones
# result <- constrOptim(theta = initial_point,
#                       f = obj_func,
#                       ui = matrix(c(10, 5, 0.1, 0.2),
#                                   ncol = 2, byrow = TRUE),
#                       ci = c(-350, -8))
#
# # Mostrando los resultados
# result$par  # Los valores de x y y que maximizan la función objetivo
# result$value  # El valor óptimo de la función objetivo


# # Resolviendo el problema de optimización con restricciones utilizando
optim
# result <- optim(par = initial_point, fn = obj_func, gr = NULL,
#                 lower = c(0, 0), method = "L-BFGS-B",
#                 control = list(fnscale = -1),
#                 ui = rbind(c(10, 5), c(0.1, 0.2)),
#                 ci = c(350, 8))
#
# # Mostrando los resultados
# result$par  # Los valores de x y y que maximizan la función objetivo
# result$value  # El valor óptimo de la función objetivo



library('Rsolnp')
mi_fun <- function(x){
  return( (x[1]-3)^2+(x[2]-2)^2 )
}
mis_inec <- function(x){
  z1 <- x[1]^2+x[2]^2
  z2 <- 2*x[1]+x[2]
  z3 <- x[1]+2*x[2]
  z4 <- -x[1]
  z5 <- -x[2]
  return(c(z1,z2,z3,z4,z5))
}
limb <- rep(-1000000,5)
lima <- c(5,6,4,0,0)
x0 <- c(2,2)
soluc <- solnp(x0, fun=mi_fun, ineqfun = mis_inec, ineqLB = limb, ineqUB
= lima)
print(paste("La solución óptima es x1=",soluc$pars[1]," y
x2=",soluc$pars[2],sep=""))
print("Los multiplicadores de Lagrange son los siguientes:")
print(soluc$lagrange*(-1))



# Ejercicio explicado

mi_fun <- function(x){
  return( -(log(x[1]) + log(x[2]) ) )
}
mis_inec <- function(x){
  z1 <- 10*x[1]+5*x[2]
  z2 <- 0.1*x[1]+0.2*x[2]
```

```
  return(c(z1,z2))
}
limb <- rep(-1000000,2)
lima <- c(350,8)
x0 <- c(2,2)
soluc <- solnp(x0, fun=mi_fun, ineqfun = mis_inec, ineqLB = limb, ineqUB
= lima)
print(paste("La solución óptima es x1=",soluc$pars[1]," y
x2=",soluc$pars[2],sep=""))
print("Los multiplicadores de Lagrange son los siguientes:")
print(soluc$lagrange*(-1))
```