Lab 6: Policy Search

Solomon(netID)

Fri., Mar. 8

Table of contents

using Revise  
using HouseElevation  
  
using CSV  
using DataFrames  
using DataFramesMeta  
using Distributions  
using LaTeXStrings  
using Metaheuristics  
using Plots  
using Random  
using Unitful  
using HouseElevation  
  
Plots.default(; margin=5Plots.mm)

slr\_scenarios = let  
 df = CSV.read("data/slr\_oddo.csv", DataFrame)  
 [Oddo17SLR(a, b, c, tstar, cstar) for (a, b, c, tstar, cstar) in eachrow(df)]  
 end  
  
 house = let  
 haz\_fl\_dept = CSV.read("data/haz\_fl\_dept.csv", DataFrame) # read in the file  
 row = @rsubset(haz\_fl\_dept, :DmgFnId .== 56)[1, :] # selecting ID  
 area = 2406u"ft^2"   
   
 House(  
 row;  
 area=area,  
 height\_above\_gauge=2u"ft",  
 value\_usd=506\_600,  
 )  
 end;  
  
 p = ModelParams(; house=house, years=2024:2100)  
  
 function draw\_surge\_distribution() # storm surge distribution  
 μ = rand(Normal(5, 1))  
 σ = rand(Exponential(1.5))  
 ξ = rand(Normal(0.1, 0.05))  
 return GeneralizedExtremeValue(μ, σ, ξ)  
 end  
  
 function draw\_discount\_rate() #drawing discounts rates  
 rate = rand(Normal(0.05, 0.02))   
 return max(0.0, rate)   
 end  
  
 N\_SOW = 1#\_000  
  
 sows = [  
 SOW(rand(slr\_scenarios), draw\_surge\_distribution(), draw\_discount\_rate()) for  
 \_ in 1:N\_SOW #generating random state of the world  
 ];

# 1. Objective Function

function objective\_function(a)::AbstractFloat  
  
 #sow =first(sows)  
 a\_action = Action(a) .\*1u"ft"  
   
 #a\_action = Action(a[1]\*u"ft") #  
 npv = [run\_sim(a\_action, sow, p) for sow in sows]   
  
 return -sum(npv)  
end;

function objective\_function(a::AbstractFloat, sow, p)  
 a\_action = Action(a \* 1u"ft")  
 npv = [run\_sim(a\_action, s, p) for s in sows]  
 return -sum(npv) # Minimize NPV by negating sum  
end

objective\_function (generic function with 1 method)

# Generate a range of elevations for testing objective function  
test\_elevations = 0:1.0:8  
for elevation in test\_elevations  
 obj\_value = objective\_function(elevation, first(sows), p) # Use first SOW for testing  
 println("Elevation: $elevation ft, npv: $obj\_value")  
end

Elevation: 0.0 ft, npv: 1.0684707227039985e7  
Elevation: 1.0 ft, npv: 9.971854416053293e6  
Elevation: 2.0 ft, npv: 8.85002974508802e6  
Elevation: 3.0 ft, npv: 7.192426414099719e6  
Elevation: 4.0 ft, npv: 4.945574964807747e6  
Elevation: 5.0 ft, npv: 3.2614594609767315e6  
Elevation: 6.0 ft, npv: 2.0591556015056726e6  
Elevation: 7.0 ft, npv: 1.1920762365191472e6  
Elevation: 8.0 ft, npv: 597387.7950795256

# 2. Constriants

bounds = boxconstraints(;lb=1.0, ub=14.0)  
options = Options(; time\_limit=10.0)  
algorithm = ECA(; options=options)

Algorithm Parameters  
====================  
 ECA(η\_max=2.0, K=7, N=0, N\_init=0, p\_exploit=0.95, p\_bin=0.02, ε=0.0, adaptive=false, resize\_population=false)  
  
Optimization Result  
===================  
 Empty status.

# 3. Optimization

Random.seed!(2024)  
result = optimize(objective\_function, bounds, algorithm)

# 4. Ploting

using Plots  
  
function objective\_function(a::AbstractFloat, sow, p)  
 a\_action = Action(a \* u"ft")  
 npv = [run\_sim(a\_action, s, p) for s in sows]  
 return -sum(npv) # Minimize NPV by negating sum  
end  
# Plot the objective function  
elevation\_range = 0:0.1:14  
objective\_values\_all = [objective\_function(elev, sows, p) for elev in elevation\_range]  
  
plot(elevation\_range, objective\_values\_all, xlabel="Elevation (ft)", ylabel="npv",  
 label="Objective Function", legend=:bottomright)  
  
# Display the plot

# 5. Analysis and Comments

* The analysis considers a finite number of SOWs, each represented by sea level rise, surge distribution and discount rate PDS. The main objective is to optimize the elevation decision that will return minimum npv.
* Discount rate, storm surge, and sea level rise are assummed to be the only source of uncertain associated with our objective function which make it easy to optimize the main objective in the analysis. However real world scenarios, there are other sorces of uncertainty such as uncertainty associated lifespan of the house, the depth-damage curve, probabibility of flood occuring.
* The use of a finite number of SOWs introduces a trade-off between computation efficiency and the accuracy of the optimization solution. Averaging over SOWs is a way to handle uncertainty in decision-making when there is less computer power.
* The decision parameter elevating the house is the only parameter being optimized in this work. Future analysis should consider also optimizing the discount rate, surge distribution and the sea level rise.
* Future work should consider more than one optimization algorithm or model for the analysis to see which one works best.