Lab 4: House Elevation NPV Analysis

Your Name

Thu., Feb. 8

Table of contents

using CSV  
using DataFrames  
using DataFramesMeta  
using Distributions  
using Interpolations  
using Plots  
using StatsPlots  
using Unitful  
  
Plots.default(; margin=6Plots.mm)  
  
include("depthdamage.jl")

Load in our data.

haz\_fl\_dept = CSV.read("data/haz\_fl\_dept.csv", DataFrame) # read in the file  
desc = "one story, Contents, fresh water, short duration"  
row = @rsubset(haz\_fl\_dept, :Description == desc)[1, :] # select the row I want  
dd = DepthDamageData(row) # extract the depth-damage data  
damage\_fn = get\_depth\_damage\_function(dd.depths, dd.damages) # get the depth-damage function

#47 (generic function with 1 method)

One should never pass up an opportunity to plot an aesthetically pleasing graph.

p = let  
 depths = uconvert.(u"ft", (-7.0u"ft"):(1.0u"inch"):(30.0u"ft"))  
 damages = damage\_fn.(depths)  
 scatter(  
 depths,  
 damages;  
 xlabel="Flood Depth",  
 ylabel="Damage (%)",  
 label="$(dd.description) ($(dd.source))",  
 legend=:bottomright,  
 size=(800, 400),  
 linewidth=2,  
 )  
end  
p

gauge\_dist = GeneralizedExtremeValue(5, 1, 0.1) # hypothetical gauge distribution  
offset = 7.5 # hypothetical height from house to gauge  
flood\_dist = GeneralizedExtremeValue(gauge\_dist.μ - offset, gauge\_dist.σ, gauge\_dist.ξ)

GeneralizedExtremeValue{Float64}(μ=-2.5, σ=1.0, ξ=0.1)

elevation\_cost = get\_elevation\_cost\_function() # gives us a fitted interpolator

elevation\_cost (generic function with 1 method)

house\_area = 1000u"ft^2"  
  
heights = uconvert.(u"ft", (0u"ft"):(1u"inch"):(10u"ft")) # some heights we will consider  
plot(  
 heights,  
 elevation\_cost.(heights, house\_area);  
 xlabel="How High to Elevate",  
 ylabel="Cost (USD)",  
 label="$(house\_area)",  
 tiitle="Cost of Elevating a House",  
)

Single Year Function:

function single\_year\_cost\_benefit(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh)  
   
 # calculate the expected damages  
 #h\_dist = GeneralizedExtremeValue(flood\_dist.μ - Δh.val, flood\_dist.σ, flood\_dist.ξ)  
 #h\_dist = flood\_dist  
 samples = rand(flood\_dist, 100\_000) .\* 1u"ft"  
 damages = damage\_fn.(samples)  
 #print(damages)  
 expected\_damages\_pct = mean(damages)  
 #println(expected\_damages\_pct)  
   
 c\_dmg = house\_value \* expected\_damages\_pct / 100  
  
 # calculate the cost of elevating  
 c\_constr = elevation\_cost(Δh, house\_area)  
  
 # return the total cost and benefit  
 #println(c\_constr)  
 #println(c\_dmg)  
 return -c\_constr - c\_dmg  
end

single\_year\_cost\_benefit (generic function with 1 method)

Test out our function:

house\_area = 1000u"ft^2"  
Δh = 2.5u"ft"  
house\_value = 250\_000  
h\_dist = GeneralizedExtremeValue(flood\_dist.μ - Δh.val, flood\_dist.σ, flood\_dist.ξ)  
single\_year\_cost\_benefit(h\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh)

-104042.65090962562

function npv\_cost\_benefit(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh, T, discount\_rate)  
 # calculate the costs and benefits for each year, and then discount  
 # see above!  
 #h\_dist = GeneralizedExtremeValue(flood\_dist.μ - Δh.val, flood\_dist.σ, flood\_dist.ξ)  
 npv = 0  
 npv += single\_year\_cost\_benefit(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh)  
 #print(npv)  
 for year in 2:T  
 Δh = 0u"ft"  
 #house\_value = house\_value\*(1.1^(year-1))  
 npv += single\_year\_cost\_benefit(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh) \* ((1-discount\_rate)^(year-1))  
 end   
 return npv  
end

npv\_cost\_benefit (generic function with 1 method)

Let’s make our single state of the world for a few actions

T = 10  
discount\_rate = 0.05  
house\_area = 1000u"ft^2"  
house\_value = 250\_000  
  
offset = 7.5# hypothetical height from house to gauge  
flood\_dist = GeneralizedExtremeValue(gauge\_dist.μ - offset, gauge\_dist.σ, gauge\_dist.ξ)  
  
  
for i in range(start=0, stop=10, length=5)  
 Δh = i \* 1u"ft"  
 println("For ", Δh)  
 h\_dist = GeneralizedExtremeValue(flood\_dist.μ - Δh.val, flood\_dist.σ, flood\_dist.ξ)  
 npv = npv\_cost\_benefit(h\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh, T, discount\_rate)  
 println("NPV = ", npv)  
end

For 0.0 ft  
NPV = -82840.41478863913  
For 2.5 ft  
NPV = -117326.81795419705  
For 5.0 ft  
NPV = -106751.4412200073  
For 7.5 ft  
NPV = -106946.22243743969  
For 10.0 ft  
NPV = -114793.46662313616

For the given SOW of a house 7.5 ft above sea level, a discount rate of 5%, and the flooding and damage functions given, the higher we elevate the house, the lower our net present value. The cost of elevating is very high, and with the numbers how we have calculated them, seems to not be worth it over a 10 year period.

Sensitivity test:

#let's start with 10 random samples for our discount rate:  
rates = rand(Normal(0.05, 0.03), 10)  
  
#Using the same state of the world as before:  
T = 10  
  
for i in range(start=0, stop=10, length=5)  
 Δh = i \* 1u"ft"  
 println("For ", Δh)  
 h\_dist = GeneralizedExtremeValue(flood\_dist.μ - Δh.val, flood\_dist.σ, flood\_dist.ξ)  
 npvector = npv\_cost\_benefit.(h\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh, T, rates)  
 npv = mean(npvector)  
 println("NPV = ", npv)  
end   
#npv = npv\_cost\_benefit.(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh, T, rates)  
  
#mean(npv)

For 0.0 ft  
NPV = -86596.13653533041  
For 2.5 ft  
NPV = -118109.68071171432  
For 5.0 ft  
NPV = -106953.50992942636  
For 7.5 ft  
NPV = -106968.25413983071  
For 10.0 ft  
NPV = -114828.56910869796

1. The NPV varies depending on the action. Elevating the house a small amount is worse than elevating it a larger amount. Still, for our state of the world, the best option seems to be not elevate the house.
2. In our case, we do see our values change as a result of the sensitivity test, but they don’t change enough to change the optimal decision. The highest NPV is still to not elevate the house for this state of the world. This is probably because in the current SOW, the cost of raising the house is a very large portion of the NPV, and since it’s not impacted by the discount rate, neither are the results. Also, since our MonteCarlo approximation chooses 10 values, a good portion of those are probably close to the mean, making the results of the sensitivity test closer to the original. If we only included the two extremes in a range of discount values, these results might change.
3. This analysis has a few limitations. It does not address the value of things inside the home that might be damaged when the home is flooded, nor the psychological hardship of having one’s home flooded, or the financial and emotional costs of having to deal with that. It also doesn’t address how the flood distribution may change over time if sea levels rise. All of these factors bias the results towards not elevating the house.

We can address the first limitation by trying to approximate the value of not having to deal with these events in dollars, although this is obviously a difficult and somewhat cynical activity.

For the second limitation, about how our model changes with climate change, we can alter the flood distribution as time goes on and look at the impact it has on our values. As an example, let’s see what happens if we modify our program so that it simulates sea levels rixing by 0.05 feet every year:

function SeaLevel\_cost\_benefit(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh, T, discount\_rate)  
  
 npv = 0  
 npv += single\_year\_cost\_benefit(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh)  
 sea\_level = 0  
 for year in 2:T  
 sea\_level += 0.05  
 Δh = 0u"ft"  
 flood\_dist = GeneralizedExtremeValue(flood\_dist.μ + sea\_level, flood\_dist.σ, flood\_dist.ξ)  
 npv += single\_year\_cost\_benefit(flood\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh) \* ((1-discount\_rate)^(year-1))  
 end   
 return npv  
end  
  
  
for i in range(start=0, stop=10, length=5)  
 Δh = i \* 1u"ft"  
 println("For ", Δh)  
 h\_dist = GeneralizedExtremeValue(flood\_dist.μ - Δh.val, flood\_dist.σ, flood\_dist.ξ)  
 npvector = SeaLevel\_cost\_benefit.(h\_dist, damage\_fn, elevation\_cost, house\_area, house\_value, Δh, T, rates)  
 npv = mean(npvector)  
 println("NPV = ", npv)  
  
end

For 0.0 ft  
NPV = -177994.01106462898  
For 2.5 ft  
NPV = -131859.47813537196  
For 5.0 ft  
NPV = -109413.60349756244  
For 7.5 ft  
NPV = -107542.45080655461  
For 10.0 ft  
NPV = -114996.17928321779

Suddenly the results change massively! It now makes the most sense to elevate the house by 7.5 feet!