Lab 3: Depth-Damage Models

DataFrames and Distributions

Jonah Schaechter js336

Fri., Jan. 26

Table of contents

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

# 1. Site information

Choosing Galveston Pier 21, Texas The guage is at 29° 18.6 N, 94° 47.6 W https://maps.app.goo.gl/GyanSMA2fp9rkVrT9

Our building is 302 17th St, Galveston, TX 77550, A multi-family residential unit 0.5 miles from the pier.

It is 4.41 feet or 1.34 meters above sea level in elevation. Looking at it on street view, the house appears to be on concrete blocks about 6 inches tall, giving it an effective height of 4.91 feet.

Row 98 from the data is two-story, no basement in Galveston, so we’ll be using that. The home is on concrete blocks, so we can be confident that it doesn’t have a basement.

# 2. Depth-Damage

scatter(  
 dd.depths,  
 dd.damages;  
 xlabel="Flood Depth at House",  
 ylabel="Damage (%)",  
 label="$(dd.description) ($(dd.source))",  
 legend=:bottomright,  
 size=(700, 500),  
)

function get\_depth\_damage\_function(  
 depth\_train::Vector{<:T}, dmg\_train::Vector{<:AbstractFloat}  
) where {T<:Unitful.Length}  
  
 # interpolate  
 depth\_ft = ustrip.(u"ft", depth\_train)  
 interp\_fn = Interpolations.LinearInterpolation(  
 depth\_ft,  
 dmg\_train;  
 extrapolation\_bc=Interpolations.Flat(),  
 )  
  
 damage\_fn = function (depth::T2) where {T2<:Unitful.Length}  
 return interp\_fn(ustrip.(u"ft", depth))  
 end  
 return damage\_fn  
end

get\_depth\_damage\_function (generic function with 1 method)

damage\_fn = get\_depth\_damage\_function(dd.depths, dd.damages)

#15 (generic function with 1 method)

haz\_fl\_dept = CSV.read("data/haz\_fl\_dept.csv", DataFrame)  
#first(haz\_fl\_dept, 3)  
  
include("depthdamage.jl")  
  
#demo\_row = @rsubset(haz\_fl\_dept, :Description == "one story, Contents, fresh water, short duration")[1, :,]  
demo\_row = @rsubset(haz\_fl\_dept, :Column1 == 98)[1, :,] #Row 98 from the data is two-story, no basement in Galveston  
dd = DepthDamageData(demo\_row)  
fieldnames(typeof(dd))

(:depths, :damages, :occupancy, :dmg\_fn\_id, :source, :description, :comment)

The depth damage cuve here shows how damage increases rapidly with flood depth at first, but then begins to slow down. This makes sense, as the bottom of the curve essentially reperesents the difference between not flooding,and flooding.

Once a high enough depth is reached, damage begins to increase linearly with depth.

p = let  
 depths = uconvert.(u"ft", (-10.0u"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

# 3. Expected annual damages

gauge\_dist = GeneralizedExtremeValue(5, 1.5, 0.1)

GeneralizedExtremeValue{Float64}(μ=5.0, σ=1.5, ξ=0.1)

I know plotting this isn’t necessary, but I like to plot things.

p1 = plot(  
 gauge\_dist;  
 label="Gauge Distribution",  
 xlabel="Water Level (ft)",  
 ylabel="Probability Density",  
 legend=:topright,  
 linewidth=2,  
)

offset = 4.91 #The house is 4.91 feet above the guage (Land is 4.41 ft + 0.5 ft concrete blocks the house is built on.)  
house\_dist = GeneralizedExtremeValue(gauge\_dist.μ - offset, gauge\_dist.σ, gauge\_dist.ξ)

GeneralizedExtremeValue{Float64}(μ=0.08999999999999986, σ=1.5, ξ=0.1)

plot!(p1, house\_dist; label="House Distribution", linewidth=2)

Monte Carlo Algorithm: 1. Sample values from the distribution of hazard 2. For each value, estimate the damage using the depth-damage function 3. Average the damages

1. Sample values from the distribution of hazard

N = 1000000 #N = one million  
Samples = rand(house\_dist, N) #Get one million random samples from our distribution

1000000-element Vector{Float64}:  
 0.7021772000186585  
 -0.6879410622864751  
 1.0714487604451688  
 -1.188799338410609  
 -1.9500498672995694  
 -1.4450038612734306  
 -0.8953243496933208  
 0.7299504744588537  
 0.7704292002675639  
 6.904984024408931  
 -0.6829368849941664  
 -1.8471143596940016  
 4.460988463738464  
 ⋮  
 1.4234525114279186  
 1.3789044139603357  
 -1.3267501206764187  
 -1.1662801973442314  
 1.8634888664197973  
 -0.6932928121970501  
 -1.0115492369743027  
 0.5855564630806223  
 1.1539144376348527  
 1.249561077352763  
 0.264878811965055  
 1.4594774482306792

1. For each value, estimate the damage using the depth-damage function

FtSamples = eval(Samples)u"ft"  
house\_damage = damage\_fn.(FtSamples)

1000000-element Vector{Float64}:  
 16.234835200298534  
 1.5602946885676245  
 21.42869256267101  
 0.0  
 0.0  
 0.0  
 0.5233782515333962  
 16.67920759134166  
 17.326867204281022  
 39.90498402440893  
 1.585315575029168  
 0.0  
 35.38296539121539  
 ⋮  
 23.54071506856751  
 23.273426483762016  
 0.0  
 0.0  
 26.180933198518783  
 1.5335359390147496  
 0.0  
 14.368903409289956  
 21.92348662580912  
 22.497366464116578  
 9.23806099144088  
 23.756864689384074

1. Average the damages

#print(length(house\_damage), "\n")  
AvgDamage = sum(house\_damage)/length(house\_damage)  
print("According to our Monte Carlo, the expected annual damages for this propery are ", round(AvgDamage; digits=2), " dollars.")

According to our Monte Carlo, the expected annual damages for this propery are 16.06 dollars.

# 4. Discussion

Let’s take a look at 100 of our sampled points and plot them on the depth-damage curve to see how they’re distributed.

We can see that most of the points are along the lower ends of the curve. There are, however, still a few points high on the curve, reflecting the inevitability of large floods once enough time passes and enough samples are taken.

HundSamples = rand(Samples, 100) #one hundred of our samples (Julia doesn't like to plot one million)  
HundFtSamples = eval(HundSamples)u"ft"  
Hundhouse\_damage = damage\_fn.(HundFtSamples)  
  
  
scatter(  
 HundFtSamples,  
 Hundhouse\_damage;  
 xlabel="Flood Depth at House",  
 ylabel="Damage (%)",  
 label="100 Points sampled for our MC algorithm",  
 legend=:bottomright,  
 size=(700, 500),  
)

Sea levels are expected to rise by 4.4 ft in this location by 2080 according to NOAA according to intermediate-high projections: https://coast.noaa.gov/slr/#/layer/sce/4/-10554960.5624167/3413062.4218919543/12/satellite/85/0.8/2080/interHigh/midAccretion

Let’s re-do our Monte Carlo approximation for an extra 4.4ft of sea level rise by lowering the height of our home above sea level by 4.4ft.

offset2080 = 0.51 #The house is 0.51 feet above the guage (Land is 4.41 ft + 0.5 ft concrete blocks - 4.4 ft sea level rise.)  
house\_dist2080 = GeneralizedExtremeValue(gauge\_dist.μ - offset2080, gauge\_dist.σ, gauge\_dist.ξ)

GeneralizedExtremeValue{Float64}(μ=4.49, σ=1.5, ξ=0.1)

N = 1000000 #N = one million  
Samples2080 = rand(house\_dist2080, N) #Get one million random samples from our distribution

1000000-element Vector{Float64}:  
 7.0164240982966595  
 3.391040135177236  
 7.591112682504478  
 3.739607612071232  
 4.088448786094044  
 4.542850006097713  
 6.481409156960158  
 4.837786426558501  
 5.065711009873155  
 8.211303039009763  
 7.314934823566897  
 3.1282671356899057  
 2.9423993030002578  
 ⋮  
 6.394263340605598  
 7.653679926129277  
 4.10795589332216  
 8.734423994774602  
 5.644947032104242  
 5.988179105680858  
 4.19092279332182  
 4.299335597272818  
 8.267045223758018  
 6.760936139834832  
 4.777890677371793  
 10.010604429485772

FtSamples2080 = eval(Samples2080)u"ft"  
house\_damage2080 = damage\_fn.(FtSamples2080)

1000000-element Vector{Float64}:  
 40.0  
 32.173120405531705  
 40.0  
 33.2188228362137  
 34.26534635828213  
 35.62855001829314  
 39.481409156960154  
 36.513359279675505  
 37.13142201974631  
 40.42260607801953  
 40.0  
 31.384801407069716  
 30.76959721200103  
 ⋮  
 39.3942633406056  
 40.0  
 34.32386767996648  
 41.468847989549204  
 38.28989406420848  
 38.97635821136172  
 34.57276837996546  
 34.89800679181845  
 40.534090447516036  
 39.76093613983483  
 36.33367203211538  
 44.031813288457315

AvgDamage = sum(house\_damage2080)/length(house\_damage2080)  
print("According to our Monte Carlo, the expected annual damages for this property are ", round(AvgDamage; digits=2), " dollars in the year 2080.")

According to our Monte Carlo, the expected annual damages for this property are 36.99 dollars in the year 2080.

plot!(p1, house\_dist2080; label="2080 House Distribution", linewidth=2)