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)

#31 (generic function with 1 method)

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}:  
 0.7980994035963803  
 -0.7734354541435275  
 3.285698743567531  
 3.7862951781714766  
 -0.4033087640105172  
 1.8045417759046798  
 1.6627419802071317  
 -0.6757962364137329  
 0.5070569578598624  
 1.7246069916090896  
 -0.015052342163653545  
 -1.9276696239447815  
 0.8654804956752096  
 ⋮  
 0.5587584538704435  
 2.5966998297785597  
 2.763282449979163  
 -0.5613406414542832  
 3.1746370614422226  
 -1.1787574005920245  
 -0.7263385992683814  
 -1.985743211064451  
 -2.004606223128748  
 0.4163223208699731  
 -1.4314648921403403  
 13.394203690112287

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}:  
 17.769590457542083  
 1.1328227292823623  
 31.85709623070259  
 33.35888553451443  
 2.983456179947414  
 25.827250655428077  
 24.97645188124279  
 1.6210188179313356  
 13.112911325757796  
 25.347641949654538  
 4.924738289181732  
 0.0  
 18.847687930803353  
 ⋮  
 13.940135261927097  
 29.38679931911424  
 30.053129799916654  
 2.1932967927285842  
 31.523911184326668  
 0.0  
 1.368307003658093  
 0.0  
 0.0  
 11.66115713391957  
 0.0  
 53.18261107033686

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}:  
 3.016851807388408  
 3.4839516847058025  
 4.467064643517658  
 4.322981370298557  
 8.544364497943675  
 4.839384702195508  
 8.875110425815203  
 9.079412401283223  
 5.938481088453048  
 4.819585535928354  
 4.114402333311306  
 6.529068885197809  
 6.585049770751151  
 ⋮  
 7.676175397663327  
 6.014215535627387  
 7.713743815702858  
 5.673693591315694  
 6.3856741213054  
 5.293912504623964  
 4.649306939727742  
 3.6596495706371552  
 4.436204754115479  
 4.102935076323384  
 6.481752211260847  
 7.732935982135858

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

1000000-element Vector{Float64}:  
 31.050555422165225  
 32.45185505411741  
 35.40119393055298  
 34.96894411089567  
 41.08872899588735  
 36.51815410658652  
 41.7502208516304  
 42.15882480256645  
 38.8769621769061  
 36.45875660778506  
 34.34320699993392  
 39.52906888519781  
 39.585049770751155  
 ⋮  
 40.0  
 39.014215535627386  
 40.0  
 38.347387182631394  
 39.3856741213054  
 37.58782500924792  
 35.947920819183224  
 32.978948711911464  
 35.30861426234644  
 34.30880522897015  
 39.48175221126085  
 40.0

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)