BEE 4750 Homework 3: Uncertain Sea-Level Rise and Levee Reliability

Name:

ID:

> **Due Date**
>
> Friday, 10/06/23, 9:00pm

Overview

Instructions

This assignment asks you to conduct a Monte Carlo analysis of levee reliability in the face of uncertain changes to local sea levels. You will propagate uncertainty in equilibrium climate sensitivity through the energy balance model to obtain a distribution of temperatures, which will then drive a model of sea-level rise. You will finally use this distribution to assess the probability that a planned levee will achieve its desired reliability standard.

Load Environment

The following code loads the environment and makes sure all needed packages are installed. This should be at the start of most Julia scripts.

In [ ]:

```
import Pkg
Pkg.activate(@__DIR__)
Pkg.instantiate()
```

```
  Activating project at `~/hw3-teaganraesmith`
```

In [ ]:

```julia
using Random
using Plots
using LaTeXStrings
using Distributions
using CSV
using DataFrames
```

Problems (Total: 40 Points)

Problem 1 (12 points)

Recall from class that the simple energy balance model (EBM) of planetary energy balance links changes in radiative forcing (F) to global mean temperature (T) changes through the discretized equation

$$T_{i+1} = T_i + \frac{F_i - \lambda T_i}{cd} \times \Delta t,$$

where i is the current time step, $c = 4.184 \times 10^6$ J/K/m^2 is the heat capacity of water per unit area, d is the (uncertain) depth of the mixing layer, Δt is the annual time step in seconds and $\lambda = F_{2\text{xCO}_2}/S$ is the climate feedback parameter in W/m^2/$^\circ$ C, where S is the equilibrium climate sensitivity (the uncertain equilibrium temperature change resulting from a doubling of atmospheric CO_2). Finally, while total radiative forcing can be the result of non-aerosol and aerosol effects, we do not know the relative intensity of aerosol forcing, so we represent this with an uncertain aerosol scaling factor α.

We can implement this model with the following Julia function. We will assume an ocean mixing depth $d = 100$ m and an aerosol scaling factor $\alpha = 1.3$ so we can focus on the uncertainty in S.

The last technical concern is that “global mean temperature” does not make sense in absolute terms as a marker of climate change. Instead, we typically refer to temperature changes relative to some historical pre-industrial baseline. In this case, we will use the period from 1880-1900, though this choice can vary.

In [ ]:

```julia
# we need to split up the aerosol and non-aerosol forcings when we call the function
function energy_balance_model(S, forcing_aerosol, forcing_non_aerosol)
    d = 100 # ocean mixing depth [m]
    α = 1.3 # aerosol scaling factor
```

```
    F2xCO₂ = 4.0 # radiative forcing [W/m²] for a doubling of CO₂
    λ = F2xCO₂/S

    c = 4.184e6 # heat capacity/area [J/K/m²]
    C = c*d # heat capacity of mixed layer (per area)

    F = forcing_non_aerosol + α*forcing_aerosol # radiative forcing

    Δt = 31558152.0 # annual timestep [s]

    T = zero(F)
    for i in 1:length(F)-1
        T[i+1] = T[i] + (F[i] - λ*T[i])/C * Δt
    end
    # return temperature anomaly relative to 1880-1900 baseline
    return T .- mean(T[1:21])
end
```

```
energy_balance_model (generic function with 1 method)
```

Finally, we need to load some radiative forcing data. We will use the radiative forcing scenario RCP 8.5. We can load this data, which is in a `.csv` (comma-delimited) file, into a `DataFrame`, which is a tabular data structure. Rows and columns in a `DataFrame` can be accessed using their numerical index (like a matrix), but columns also have names; you can access a particular column in a dataframe `df` by name using `df.colname` or `df[:, "colname"]`.

Of note: this data set goes from 1750–2500, so you will need to take care to make sure you are using the right years at each step. For example, here we will constrain the data to 1880–2100, which is the period we are interested in.

In [ ]:

```
# The CSV is read into a DataFrame object, and we specify that it is comma delimited
forcings_all_85 = CSV.read("data/ERF_ssp585_1750-2500.csv", DataFrame, delim=",")

# get the years corresponding to the forcings
t = Int64.(forcings_all_85[!,"year"]) # Ensure that years are interpreted as integers
# find the indices of the years 1880 and 2100
# we can do this with the indexin function
time_bounds = indexin([1880, 2100], t)
years = time_bounds[1]:time_bounds[2] # create range of years

# Separate out the individual components
forcing_co2_85 = forcings_all_85[years,"co2"]
# Get total aerosol and non-aerosol forcings
```

```
forcing_aerosol_rad_85 = forcings_all_85[years,"aerosol-radiation_interactions"]
forcing_aerosol_cloud_85 = forcings_all_85[years,"aerosol-cloud_interactions"]
forcing_aerosol_85 = forcing_aerosol_rad_85 + forcing_aerosol_cloud_85 # aerosol forcings
forcing_total_85 = forcings_all_85[years,"total"]
forcing_non_aerosol_85 = forcing_total_85 - forcing_aerosol_85 # non-aerosol forcings
```

```
221-element Vector{Float64}:
  0.42741479112315905
  0.4487940147601447
  0.4900144276528058
 -0.019811270078689047
 -1.480725700367619
 -0.3154905163510021
  0.233186559774844
  0.2923308952663089
  0.45833872365810924
  0.5168189581089915
  ⋮
  9.879065698371564
  9.946850640754889
 10.002169546141578
 10.061009359425011
 10.115684195707905
 10.182946497594184
 10.249699157847772
 10.331407709334023
 10.424544495740134
```

For this assignment, you can use the `forcing_aerosol_85` and `forcing_non_aerosol_85` vectors as is to correspond to the relevant forcings. You will need to use the vector `t` to find the appropriate years for analysis.

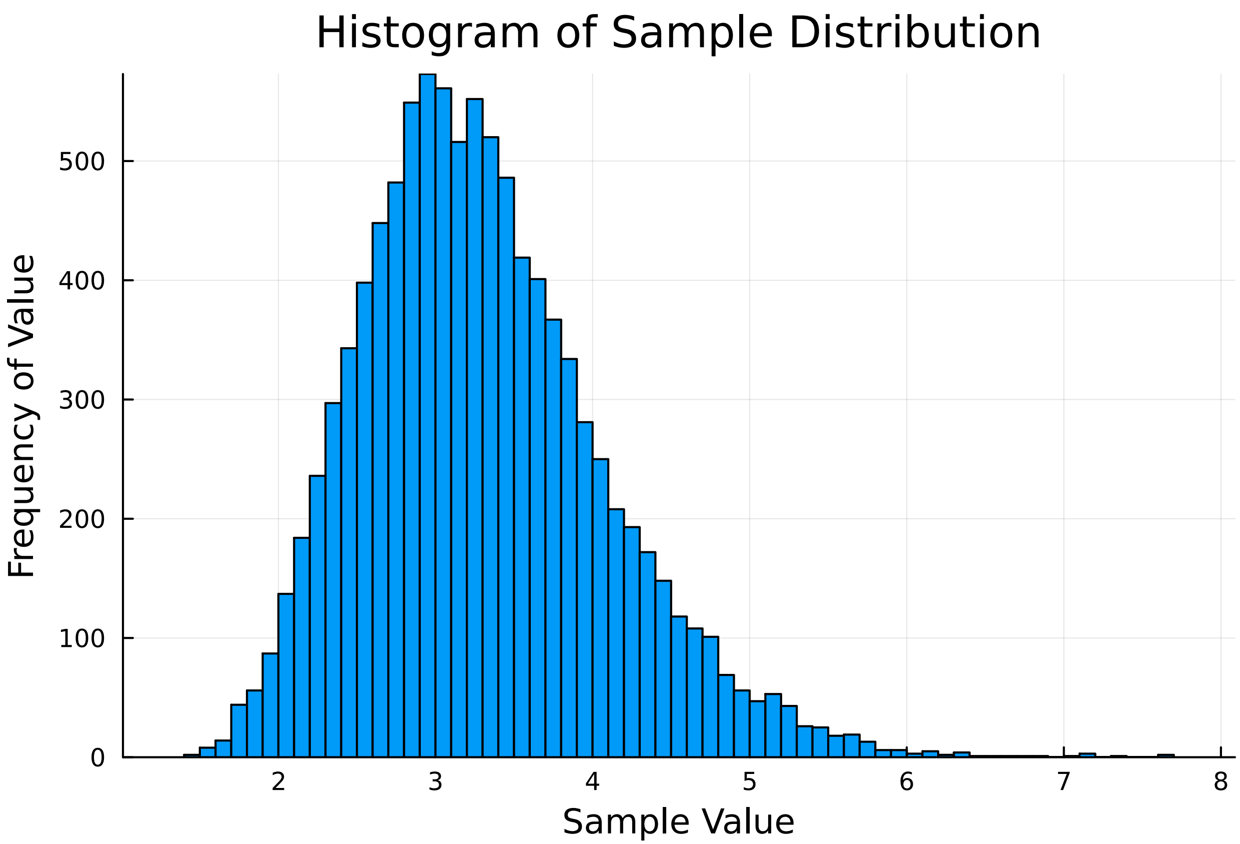
Problem 1.1 (3 points)

Assume that S is distributed according to $\mathrm{LogNormal}(\log(3.2), \log 2/3)$ (as in class). Draw 10,000 samples from this distribution and make a histogram.

In [ ]:

```
sd=log(2)/3
mean_val=log(3.2)
S_val=LogNormal(mean_val, sd) #Giving S a lognormal distribution
S=[]
for x in 1:10000 #10,000 samples
    S_new=rand(S_val) #Choosing a sample randomly within the distribution
```

```
    S=append!(S, S_new)
end

#Plotting histogram
histogram(S, legend=:false, bins=100)
ylabel!("Frequency of Value")
xlabel!("Sample Value")
title!("Histogram of Sample Distribution")
```



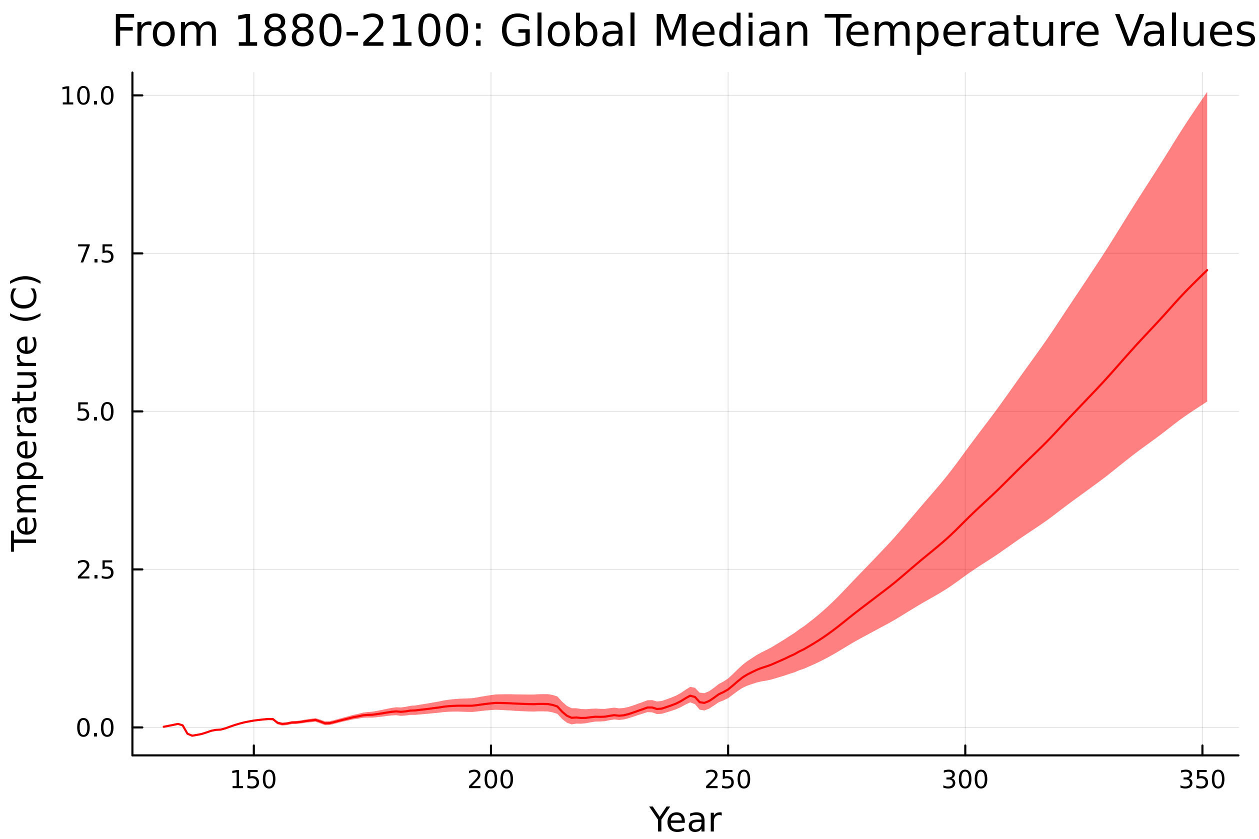
Problem 1.2 (5 points)

Use the EBM to propagate your samples of S to a distribution of global mean temperature. Plot the median and 90% predictive distribution (between the .05 and .95 quantiles) from 1880-2100.

In [ ]:

```
sd=log(2)/3
mean_val=log(3.2)
```

```julia
S_val=LogNormal(mean_val, sd) #Giving S a lognormal distribution
S=[]
for x in 1:10000 #10,000 samples
    S_new=rand(S_val) #Choosing a sample randomly within the distribution
    S=append!(S, S_new)
end

#Creating an array of number of years
years_1880_2100 = []
for idx in 1880:1:2100
    append!(years_1880_2100,idx)
end

#Creating an empty matrix with col=years length, and rows = S length
globaltemperatures = zeros((length(S)),length(years_1880_2100))

#For all years, iteration for samples of S using Energy Balance Model
for (idx,S) in pairs(S)
    globaltemperatures[idx,:] = energy_balance_model(S, forcing_aerosol_85, forcing_non_aerosol_85)
end

#Finding Quantiles of global temperatures (for each year)
#.05 quantile
temp_05 = quantile.(eachcol(globaltemperatures), 0.05)

#.95 Quantile
temp_95 = quantile.(eachcol(globaltemperatures), 0.95)

#Finding median of global temperatures (for each year)
temp_median = quantile.(eachcol(globaltemperatures), 0.5)

#Plotting median and 90% predictive
plot(years, temp_median, ribbon=(temp_median - temp_05, temp_95 - temp_median),legend=false,color="red")
xlabel!("Year")
ylabel!("Temperature (C)")
title!("From 1880-2100: Global Median Temperature Values")
```

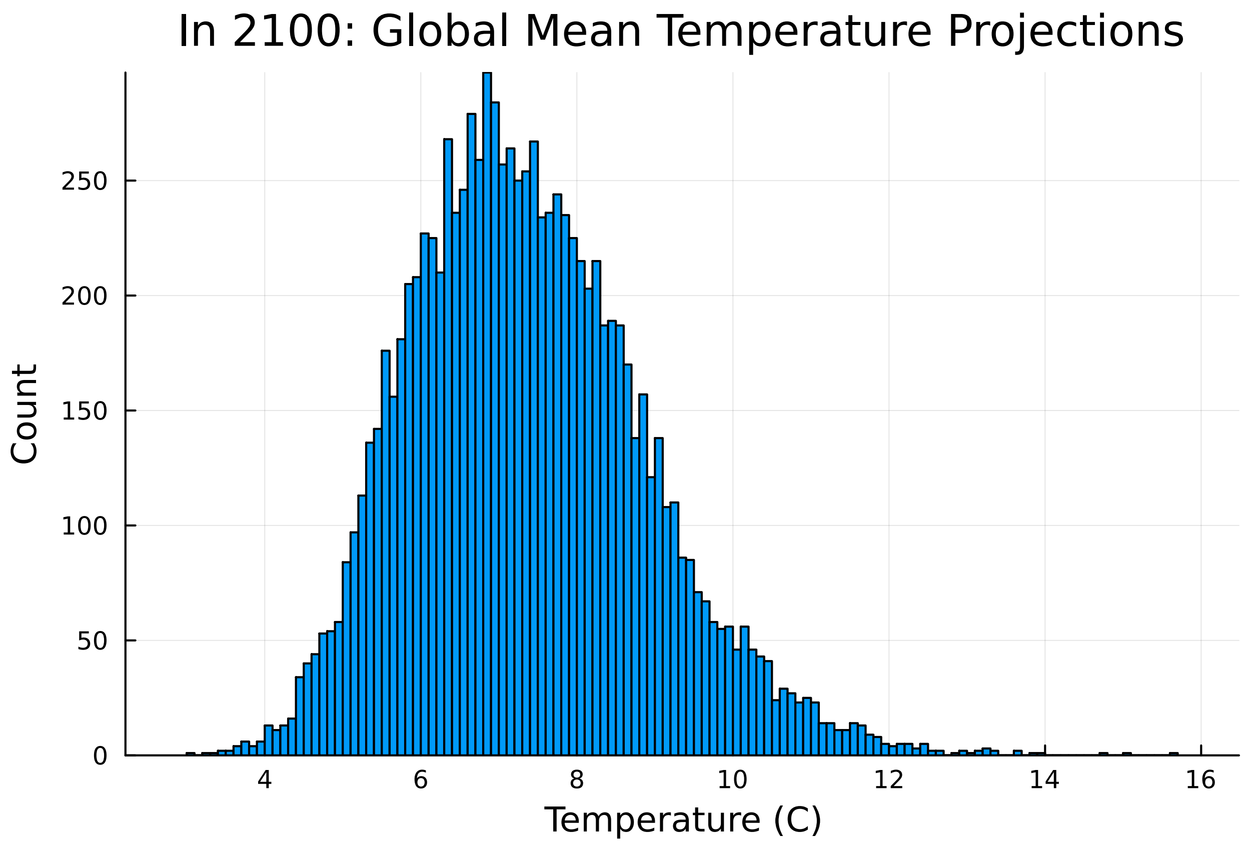


Problem 1.3 (4 points)

Make a histogram of global mean temperature projections in 2100. If you compare this distribution to the distribution of S from Problem 1.1, what do you observe?

In [ ]:

```
temp_2100 = globaltemperatures[:,length(years_1880_2100)]
histogram(temp_2100, legend=:false, bins=150)
xlabel!("Temperature (C)")
ylabel!("Count")
title!("In 2100: Global Mean Temperature Projections")
```



Compared to the histogram in 1.1, I observe that the shape is of the same distribution. This is because S still contains the given LogNormal mean and SD. More generally, I also observe that the temperature prediction is most common at around 7.5 degrees C. This indicates that this will most likely, with some variance, be the temperature in 2100 if this model is correct. The reasoning for my bins size being 150 was because it was both high enough that there were enough data points to show the shape of the graph, but not so high that the data showcased a lot of noise.

Problem 2 (15 points)

Changes to global temperatures cause changes in global sea levels through several mechanisms, including thermal expansion (the change in ocean volume due to increased heat content) and melting land-based ice. One simple way to represent this link is through the following model, proposed by Rahmstorf (2007).

$$\frac{dH}{dt} = a(T - T_0),$$

where H is the global mean sea level in mm, T is global mean temperature, T_0 is an equilibrium temperature (where there is no change in sea levels), and a is a proportionality constant. This model can be discretized to give

$$H_{i+1} - H_i = a(T_i - T_0).$$

Note that, like with global mean temperature, the notion of "global mean sea level" does not make sense in absolute terms (were sea levels ever at "zero"?). Instead, we want to normalize this relative to some historical baseline. In this case (with a view towards Problem 3), we will compute our sea levels relative to the 2010 sea level. Note that in addition to the model parameters, we also need an initial sea-level parameter H_0 which will give us the right anomaly level.

The best estimates for these parameters are:

- $a = 1.86$;
- $H_0 = -223$;
- $T_0 = -0.62$

Problem 2.1 (5 points)

Write a function `sea_level_model()` to implement the mathematical sea-level rise model described above. It should take in needed parameters and a vector of temperatures and return a vector of sea levels. To test your function, use the provided temperature series `historical_temps` (read in below) to compute the global mean sea level anomaly in 2022 (the last year of the dataset) with the parameter values above; you should get a value of approximately 40mm.

In [ ]:

```
historical_temp_data = CSV.read("data/HadCRUT.5.0.1.0.analysis.summary_series.global.annual.csv", DataFra
# column 2 is the temperature anomaly, column 1 is the year
temp_bds = indexin([1880, 1900], historical_temp_data[!, 1]) # find the index of 2010 for normalization
historical_temp_data[:, 2] .-= mean(historical_temp_data[temp_bds[1]:temp_bds[2], 2])
historical_temps = historical_temp_data[temp_bds[1]:end, 2]
```

```
143-element Vector{Float64}:
  0.07297116761904765
  0.15655772761904765
  0.09327316761904769
  0.04232884761904765
 -0.10351681238095234
 -0.08232033238095232
 -0.03210037238095231
```

```
-0.10998251238095236
 0.00942435761904764
 0.13890768761904768
 ⋮
 1.0616748976190478
 1.2139176176190476
 1.3217303776190477
 1.2339774976190476
 1.1514572476190477
 1.2798758476190477
 1.3115970476190477
 1.1506591476190478
 1.1896871476190476
```

In [ ]:

```julia
function sea_level_model(a, H_0, T_0, T)
    H=zeros(length(T)+1);
    H[1]=H_0;
    for i in 2:length(T)+1
       H[i] = H[i-1]+a*(T[i-1]-T_0)
    end
    return H[2:end].-H[132]
end
```

```
sea_level_model (generic function with 1 method)
```

In [ ]:

```julia
#Testing the function
sealevel_years = []
a=(1.86)
Z_0=(223) #reconfigured from problem statement
T_0=(-.62)

H=sea_level_model(a,Z_0,T_0,historical_temps)
leng=length(H)
sealevel_anomaly_2022=(H[leng]) #rounded value for less significant digits
print("The global mean sea level anomaly in 2022 is: ", round(sealevel_anomaly_2022), "mm")
```

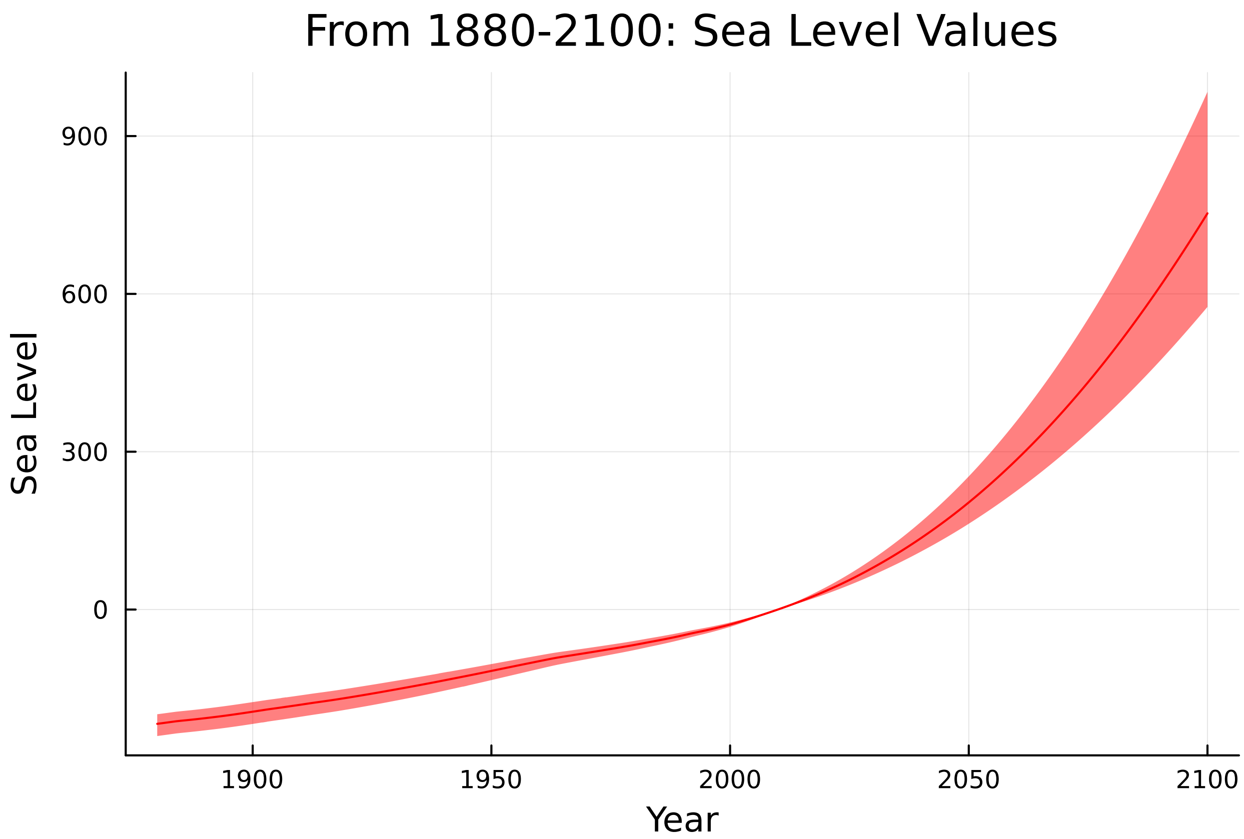
```
The global mean sea level anomaly in 2022 is: 40.0mm
```

Problem 2.2 (5 points)

Evaluate `sea_level_model()` using the projected temperature ensemble from Problem 1. Plot the 90% projection interval of the sea levels.

In [ ]:

```
#Given Constants
a=1.86
H_0=Float64(223)
T_0=-.62

H= zeros(size(globaltemperatures))

#For all years, iteration for samples of S using Energy Balance Model
for idx=1:size(globaltemperatures)[1]
    H[idx,:] = sea_level_model(a,H_0,T_0,globaltemperatures[idx,:])
end

quantiles=zeros(2,length(years_1880_2100))

#Finding Quantiles of global temperatures (for each year)
#.05 quantile
quantiles[1,:]=quantile.(eachcol(H), 0.05)
#.95 Quantile
quantiles[2,:]= quantile.(eachcol(H), 0.95)

#Finding median of global temperatures (for each year)
sealevel_median = quantile.(eachcol(H), 0.5)

plot(years_1880_2100, sealevel_median, ribbon=(sealevel_median - quantiles[1,:], quantiles[2,:] - sealeve

xlabel!("Year")
ylabel!("Sea Level")
title!("From 1880-2100: Sea Level Values")
```



Problem 2.3 (5 points)

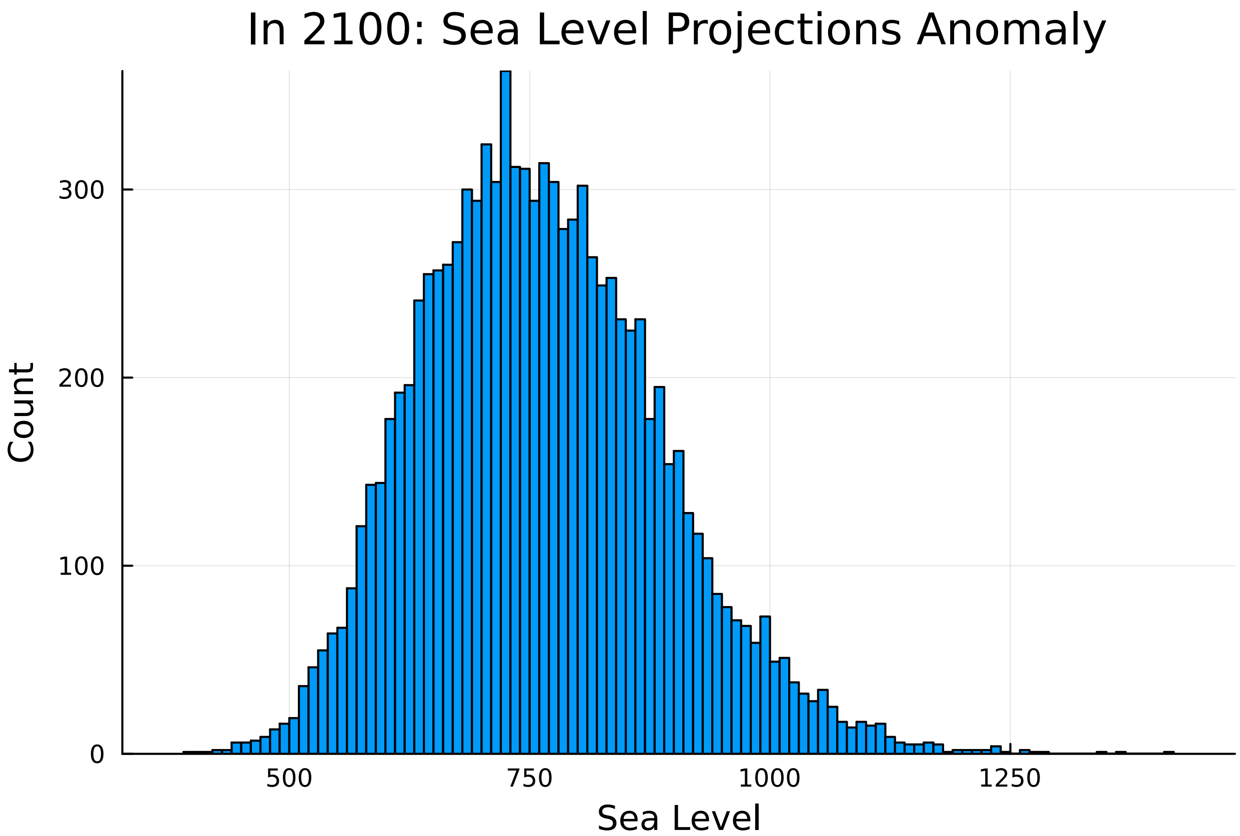
Make a histogram of the sea-level anomaly in 2100. What can you observe about how the ECS uncertainty has impacted sea-level uncertainty under this radiative forcing scenario? What might the implications be of only using the best-estimate ECS value?

ECS uncertainty has propagated the uncertainty under this scenario. Therefore the noise of the graph and relative uncertainty increases. There is significant noise in the projections additionally because by only using the best-estimate ECS value, you are propagating uncertainty in the system.

In [ ]:

```
SeaLevel_2100 = H[:,length(years_1880_2100)]
histogram(SeaLevel_2100, legend=:false, bins=150)
xlabel!("Sea Level")
```

```
ylabel!("Count")
title!("In 2100: Sea Level Projections Anomaly")
```



Problem 3 (13 points)

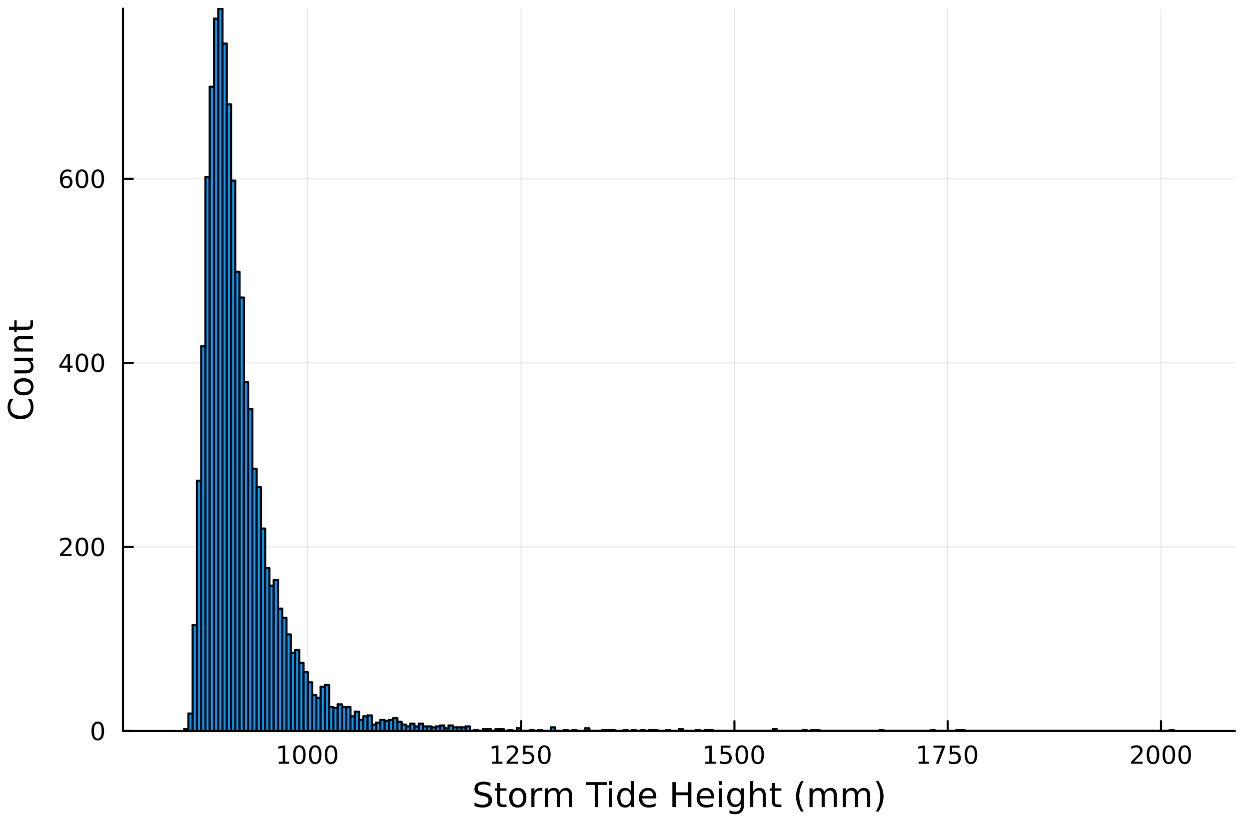
You've been asked to consult on a levee reliability analysis. For context, levees in the United States are supposed to only fail once in 100 years, or, in other words, to have at most a 1% chance of failure in a given year. We will assume for this problem that the only way in which a levee fails is by being overtopped (note: this is unrealistic).

We can assess the probability of levee overtopping by comparing its height to a distribution of extreme sea levels. A common approach is to look at the distribution of the highest sea level each year. These extreme sea levels can be obtained by combining the absolute sea level (we will use our distribution of global sea levels for this), the rate of subsidence (how much the ground sinks), and the distribution of storm tides (the highest tide level, which is often the result of storm surges combining with high tide).

Assume for this problem that:

1. the annual rate of subsidence ν is 1.2mm/yr;
2. the distribution of annual storm tide maxima, above the mean sea level, is (and is expected to continue to be) given by a $\text{GeneralizedExtremeValue}(900, 25, 0.3)$ distribution, which looks like this:

In [ ]:
```
tide_distribution = GeneralizedExtremeValue(900, 25, 0.3)
    histogram(rand(tide_distribution, 10000), xlabel="Storm Tide Height (mm)", ylabel="Count", legend=:fa
```



Feel free to just sample from `tide_distribution` in your solution below.

Problem 3.1 (2 points)

How would you use your sea-level simulations and the above information to compute a distribution of extreme sea levels in 2100 relative to 2010 mean sea level? Write down the approach in clear steps, with equations as needed.

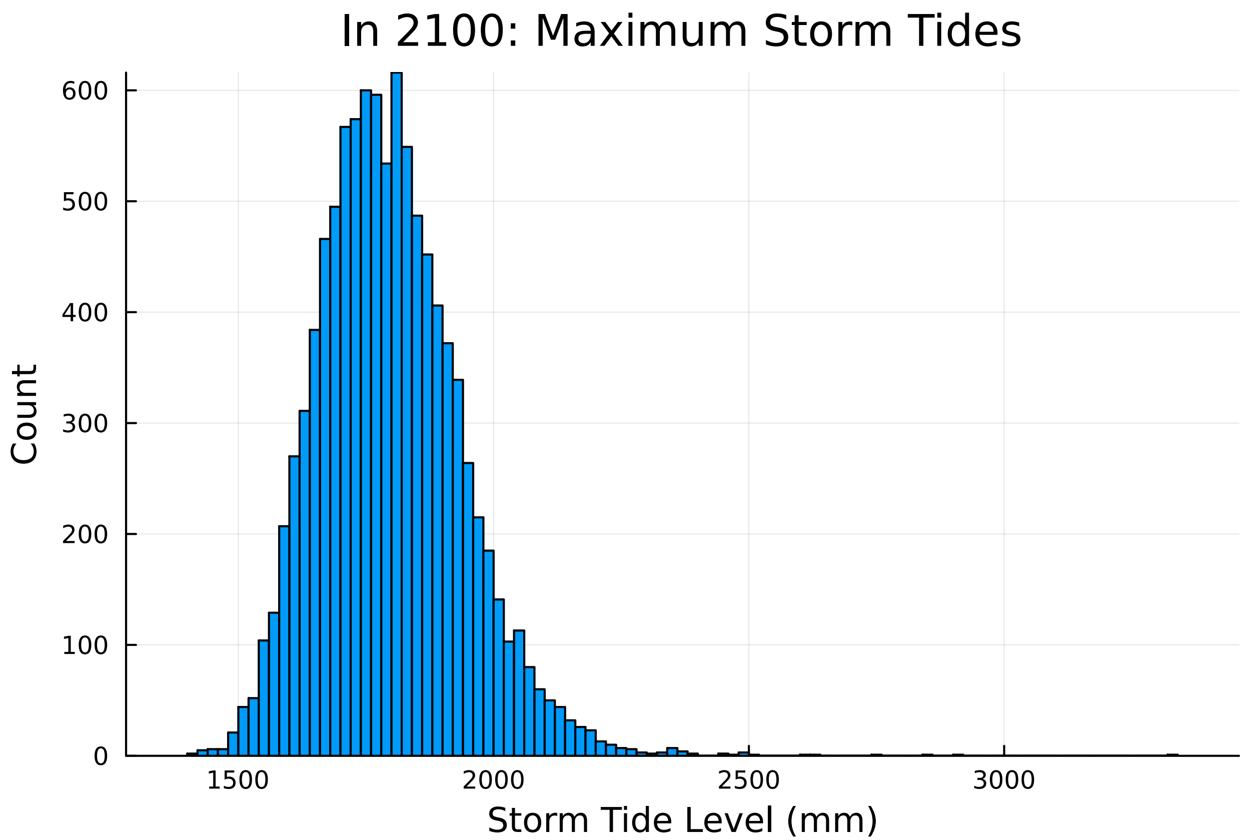
First, similar to question 1, we will be creating 10,000 samples from the given tide_distribution with random values. Next, similar to question 2, we will be using a baseline of 2010 values, so the subsidence only will be increased by the difference between 2100 and 2010 multiplied by the subsidence rate of 1.2mm/yr. The subsidence will be added to the sea level because as the sea floor sinks, the sea level increases by that amount. Next, in order to find the max sea level from storm tides, you must add the sea level for 2100 we got in problem 2 plus the max tide random samples to the subsidence value. (Max Storm Tide=problem 2 sea level+random max tide level+subsidence) This will be the value needed to use in the histogram.

Problem 3.2 (3 points)

Follow the steps above and produce a histogram of the extreme sea levels in 2100 relative to 2010.

In [ ]:

```
subsidence = (2100-2010+1)*1.2 #total subsidence as determined by subsidence level and years

maxtide_level = rand(tide_distribution,10000) #Random sample from given distribution
maxsea_level= H[:,end].+maxtide_level.+subsidence

#plotting
histogram(maxsea_level, legend=:false, bins=100)
xlabel!("Storm Tide Level (mm)")
ylabel!("Count")
title!("In 2100: Maximum Storm Tides")
```



Problem 3.3 (5 points)

The current levee was heightened in 2010 to 2m above the 2010 mean sea level. Based on your analysis above, what is the probability that the levee will be overtopped in 2100 (remember that the reliability standard is 1%)?

In [ ]:

```
levee_level=2*1000 #mm
count_over=0
#Checking to see which sea values are greater than the levee level
for idx in 1:length(maxsea_level)
    if maxsea_level[idx]>levee_level
        count_over=count_over+1;
    end
end

#Creating a probability for sea levels above levee levels
```

```
probability=count_over/length(maxsea_level)*100
print("The probability that the levee is overtopped in 2100 is: ", round(probability), "%")
```

```
The probability that the levee is overtopped in 2100 is: 7.0%
```

This probability is over the reliability standard of 1% by about 6%. This means that the levee is not heigntened enough for that year.

Problem 3.4 (3 points)

Based on your analysis, would you recommend that the levee be heightened again in the future, and if so, how high? What other information might you need, if any, to make your recommendation?

Yes, I would recommend that the levee be heightened for the future because it clearly is above the reliability standard for the year 2100. As shown below, by changing the levee height, trying to get within the 1% reliability factor, the levee height must be approximately 2176mm high (2.176m). Levees are important structurally to protecting people so most likely it would have to be heightened based on environmental regulation, but there could also be a chance that somehow it's privatized and in that case cost might also be taken into account.

In [ ]:

```
check=false
levee_level=2*1000 #mm

while check==false
    levee_level=levee_level+1 #mm
    count_over=0
    #Checking to see which sea values are greater than the levee level
    for idx in 1:length(maxsea_level)
        if maxsea_level[idx]>levee_level
            count_over=count_over+1;
        end
    end
    #Creating a probability for sea levels above levee levels
    probability=count_over/length(maxsea_level)*100
    #Checking if probability is below or equal to 1
    if probability<=1
        check=true
    end
end
print("The probability that the levee is overtopped in 2100 is: ", round(probability), "% at Levee height
```

```
The probability that the levee is overtopped in 2100 is: 1.0% at Levee height 2176mm
```

References

List any external references consulted, including classmates.

Consulted with Emma Rose Connolly and worked with information given by Gabriela the TA.