Final Project Report: Adding Sequential Analysis to the House Elevation Problem

Leanh Nguyen (Ln14)

Sun., Apr. 28

Table of contents

# 1. Introduction

## 1.1 Problem Statement

Clearly define the problem statement that your chosen feature aims to address.

Explain the significance of this problem in the context of climate risk management.

With the recent federal infrastructure bill, American cities are set to spend billions to improve their water and flood systems. However, with the growing concern about climate change’s impact on infrastructure, climate risk management is coming into the spotlight for future urban planning. However, the question of how to approach climate change forecasts remains in the minds of many researchers.

In the case of a property elevation problem, we have a property owner who is threatened by sea level rise and wants to elevate their property. However, elevating a house is a costly endeavor, so it is crucial to find the balance between protection and cost. Ideally, they want to minimize costs and maximize benefits (e.g., flood protection, human safety, etc.). A simple model could take the world’s current state(s) and a possible decision (i.e., how high to elevate the property) to generate a potential solution for the owner.

However, in the real world, most decisions made by regular people are not static, but rather sequential. In other words, people can hold off from deciding once they have more information about the world and/or have enough resources to perform an action (e.g., elevating a building). In this case, at each time step, a property owner can make a decision based on the state of the system (e.g., current sea level, current flood distribution, current building elevation, etc.) which can change at each time step. Additionally, in the real world, the property owner would also get some immediate feedback or rewards like the cost of flood insurance and the cost of elevating.

By applying sequential analysis, we can utilize an adaptive policy that allows the property owner to make more accurate decisions based on the current state of the world, which can reduce costs while maintaining or improving overall benefits. From this, researchers can scale this model up to perform over neighborhoods, cities, and regions which can help property owners make informed decisions on elevating and advise future construction to consider adaptive structures that allow for future modifications and/or elevations.

## 1.2 Selected Feature

Describe the feature you have selected to add to the existing decision-support tool.

Discuss how this feature relates to the problem statement and its potential to improve climate risk assessment.

Initial Challenges:

1. Uncertainty: Future sea level rise and storm surge predictions are uncertain.
2. Static Approach: A traditional, one-time elevation based on initial projections might not be optimal.

Sequential Analysis involves making decisions at multiple periods of time, using data and outcomes observed up to each point to refine and adjust policies. This approach is useful when uncertainties are high and the situation changes over time, a key characteristic of climate change and sea level rise.

Adaptive Policy refers to a strategy designed to be flexible and responsive to changes in the environment or system based on predefined triggers or new information, meaning that the policy can evolve as more information becomes available or as the situation changes.

Feature

1. To apply an adaptive policy, this code will utilize direct policy search, which assumes a specific functional form for a policy 𝜋 with parameters 𝜃 such that 𝑎𝑡 = 𝑓(𝑥𝑡,𝜃). The goal of policy search is to find the parameters 𝜃 that maximize the expected sum of future rewards.
2. Common choices of 𝑓 include linear decision rules, radial basis functions, binary trees, and neural networks. A primary advantage of policy search is that it can be very flexible and adaptive, and can be used with simulation-optimization frameworks. A primary disadvantage is that it can be computationally expensive and require a lot of data to estimate the parameters.
   1. Here, I define a buffer height as the minimum tolerable elevation of the house relative to the mean sea level.
   2. Also, I define a freeboard height, which is an additional heightening built on top of the minimum heightening that would bring the house back to a safe height. A freeboard height is often applied to reduce the probability of needing house elevations in consecutive years since there is an initial cost to start the elevation process and heightenings in consecutive years would be expensive.
3. All in all, by considering future conditions and adapting with buffer and freeboard heights, this would provide safer and more robust options and decisions for homeowners with properties in coastal areas, providing more benefits while balancing costs.

# 2. Literature Review

Provide a brief overview of the theoretical background related to your chosen feature.

Cite at least two relevant journal articles to support your approach (see [Quarto docs](https://quarto.org/docs/authoring/footnotes-and-citations.html) for help with citations).

## 2.1 [Using direct policy search to identify robust strategies in adapting to uncertain sea-level rise and storm surge](https://doi.org/10.1016/j.envsoft.2018.05.006)

Garner and Keller (2018)

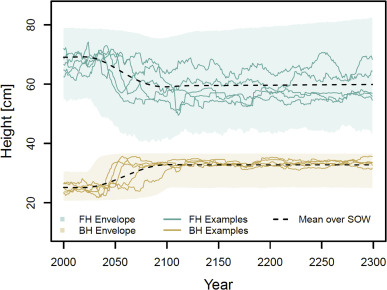
**Summary**

The article explores how computational techniques can be utilized to create effective adaptation strategies for managing risks associated with sea-level rise and storm surge under conditions of uncertainty. The article uses a method called direct policy search, which is a form of reinforcement learning, to find strategies that are robust across a range of uncertain future scenarios.

**Background and Motivation**

1. Sea-Level Rise (SLR) poses significant risks to coastal communities, ecosystems, and infrastructure worldwide. Decision makers have to constantly deal with uncertain SLR projections when designing coastal adaptation strategies and policies.
2. This paper acknowledges the significant uncertainties involved in predicting sea-level rise and the frequency and intensity of storm surges due to climate change. These uncertainties challenge traditional planning methods that rely on fixed assumptions and deterministic forecasts.

**Direct Policy Search**



The Direct Policy Search Approach

1. In this paper, DPS is used to explore a wide variety of possible adaptation policies to identify those that perform well across a range of uncertain future scenarios. This approach involves using a simulation model to evaluate the performance of different policy options under various scenarios of sea-level rise and storm surge events. The model uses an optimization algorithm to iteratively improve policy choices by evaluating their effectiveness in minimizing costs and damages across these scenarios.
2. The study suggests that adaptive policies, which can be adjusted in response to changes in climate conditions and sea-level observations, are generally more robust than static policies.
3. DPS outperforms the traditional approach in terms of Pareto-dominance (e.g., balancing conflicting objectives). Additionally, DPS provides better solution quality under uncertain sea-level rise projections and storm surge.
4. Most importantly, the new formulation achieves high-quality solutions with lower computational demands compared to intertemporal optimization methods.

**Significance and Implications**

1. The study highlights the utility of multi-objective adaptive formulations for coastal adaptation.
2. Decision makers can use this approach to craft robust strategies that adapt dynamically to changing conditions.
3. The findings extend beyond coastal adaptation and have broader applications in climate change decision problems.
4. The concept of Direct Policy Search (DPS) can be directly applied to my house elevation project, especially in managing the risks associated with sea level rise and storm damage. Here is how this approach can enhance decision-making and provide better options for homeowners through policy search and sequential analysis:
   1. Robust Strategies and Uncertainty:
      1. The paper emphasizes the need for robust strategies in the face of uncertain future projections.
      2. Similarly, this project deals with uncertainties related to sea level rise data and storm damage.
      3. By adopting robust strategies, this project can ensure that homeowners’ safety is prioritized even as conditions evolve.
   2. Direct Policy Search and Sequential Analysis:
      1. DPS is a useful approach for finding optimal solutions under uncertainty.
      2. In this project, sequential analysis plays a crucial role:
         1. Buffer and Freeboard Heights: Consider buffer and freeboard heights as part of the adaptation policies.
         2. Incremental Adjustments: The concept of adaptive policies highlighted in the paper suggests that strategies should not be static but adjustable as new data becomes available. In this project, this means setting up a system where buffer and freeboard heights can be modified over time based on ongoing sea-level rise and storm data, similar to how the paper sets up buffer and freeboard heights. Like the paper, this could involve setting predefined triggers that, when reached, would recommend re-evaluating and adjusting the elevation requirements. Moreover, similar to the paper, we utilize various formulas and concepts (e.g., slope and acceleration of sea level rise) to calculate the buffer and freeboard heights.
   3. Multi-Objective Formulation:
      1. The paper’s multi-objective approach aligns with this project’s goals of balancing trade-offs between different objectives (e.g., minimizing flood risk vs. minimizing costs).
   4. Decision Robustness for Homeowners:
      1. Just as the paper seeks robust solutions, this project also aims to provide better options for homeowners.
      2. Robust elevation policies can:
         1. Account for varying uncertain sea level rise scenarios.
         2. Adapt buffer and freeboard heights as new information comes up.

## 2.2 [Dynamic adaptive policy pathways: A method for crafting robust decisions for a deeply uncertain world](https://doi.org/10.1016/j.gloenvcha.2012.12.006)

Haasnoot et al. (2013)

**Summary**

1. The article introduces a novel approach to creating adaptable and resilient policy strategies in the face of deep uncertainties, particularly related to climate change. This method is known as Dynamic Adaptive Policy Pathways (DAPP). DAPP is designed to help policymakers develop strategies that remain viable under a range of future scenarios, ensuring flexibility and robustness against unforeseen changes.

**Background and Motivation**

1. Traditional planning approaches struggle to address uncertainties related to global and regional changes (e.g., climate change, economic changes, and technological advancements).
2. The article proposes an alternative approach that embraces uncertainty and allows for dynamic adaptation over time.
   1. Concept of Deep Uncertainty:
      1. These are situations where the future is unknown, and past data cannot reliably predict future conditions. This uncertainty is prevalent in issues like climate change, where the exact magnitude and location of impacts are difficult to predict due to complex system dynamics of the Earth and manmade systems.

**Dynamic Adaptive Policy Pathways (DAPP)**



The Dynamic Adaptive Policy Pathways approach

1. The DAPP method involves mapping out a series of potential decision pathways based on different scenarios that evolve over time. Each pathway represents a sequence of policy actions that can be adapted based on how the future plays out.
2. DAPP combines two complementary approaches:
   1. Adaptive Policymaking: This theoretical approach involves different types of actions (e.g., mitigating actions and hedging actions) and signposts to monitor whether adaptation is needed.
   2. Adaptation Pathways: These pathways provide a framework for decision-making by considering multiple possible futures and adjusting policies accordingly.
3. Overall, the DAPP method encourages planners to do the following:
   1. Create a strategic vision of the future.
   2. Commit to short-term actions.
   3. Establish a flexible framework to guide future actions.
   4. Continuously adapt the plan based on changing circumstances.
4. Adaptation Tipping Points (ATPs):
   1. The approach is grounded in identifying adaptation tipping points. These are thresholds where current policies will no longer meet the agreed objectives due to changing conditions. Recognizing these ATPs helps warn us when we need to switch from one policy pathway to another.
   2. In other words, developing pathways involves creating a “pathway map” that illustrates sequential policy steps and potential future decisions. Moreover, this pathway map includes decision points that are linked to monitoring and triggers, which inform when a policy adjustment/shift is required, as mentioned above.

**Significance and Implications**

1. The DAPP method provides a practical and effective way to navigate complex and uncertain decision-making environments.
2. It emphasizes flexibility, learning, and adaptation rather than rigid long-term plans, which are relevant for various fields (e.g., environmental policy, infrastructure planning, and climate change risk management).
3. The concept of Dynamic Adaptive Policy Pathways (DAPP) can be directly applied to my house elevation project, especially in managing the risks associated with sea level rise and storm damage. Here is how this approach can enhance decision-making and provide better options for homeowners through sequential analysis:
   1. Sequential Analysis and Deep Uncertainty:
      1. The paper emphasizes planning under deep uncertainty, which aligns with this project’s focus on sea level rise data and storm damage.
      2. By considering these uncertainties, this project can create a dynamic framework that adapts over time as new information becomes available.
      3. Sequential analysis allows this project to update decisions based on new states, which is essential for robust policy-making.
   2. Buffer Heights and Adaptive Policymaking:
      1. In this project, buffer heights are essential in protecting homes from floods and rising sea levels.
      2. By applying adaptive policymaking principles, this project can:
         1. Set short-term buffer height targets based on current data.
         2. Continuously monitor sea level rise and storm damage.
         3. Adjust buffer heights to maintain safety.
         4. Consider different varying scenarios and adapt accordingly.
   3. Freeboard Heights and Adaptation Pathways:
      1. Freeboard heights are also crucial for resilience.
      2. By applying adaptation pathways, this project can:
         1. Develop multiple freeboard height options.
         2. Evaluate their effectiveness under various sea level rise scenarios.
         3. Choose pathways that balance risk reduction, cost, and feasibility.
         4. Regularly reassess and adjust freeboard heights based on observed trends.
   4. More Homeowner Options and Increased Robustness:
      1. This project aims to provide better options for homeowners.
      2. By incorporating dynamic adaptive pathways, this project enhances robustness of decisions:
         1. Homeowners can choose from a range of elevation options.
         2. Policies remain flexible, accommodating future and changing conditions.
         3. Homeowners benefit from informed decisions that take into account uncertainties.
4. In summary, the paper’s concepts—deep uncertainty, adaptive policymaking, and adaptation pathways—can help guide this project’s approach.

# 3. Methodology

## 3.1 Implementation

You should make your modifications in either the HouseElevation or ParkingGarage module. 1. I made a new file called project.qmd which combines elements from HouseElevation and concepts from ParkingGarage. I wanted to make a new file so I can have a reference in case something goes wrong (e.g., messing up a struct variable or redefining functions).

Include code snippets and explanations where necessary to clarify the implementation process. 1. Code is below with explanations above each code block.

Detail the steps taken to implement the selected feature and integrate it into the decision-support tool.

### 3.1.1 Setup

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

ModelParamsSA contains all the variables that are constant across simulations. Differs from ModelParams

house = let  
 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  
 area = 500u"ft^2"  
 height\_above\_gauge = 12u"ft"  
 House(row; area=area, height\_above\_gauge=height\_above\_gauge, value\_usd=250\_000)  
end

Next we define how we will sample the states of the world. This section remains constant to provide comparable results with the original optimization problem.

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  
  
function draw\_surge\_distribution()  
 μ = rand(Normal(5, 1))  
 σ = rand(Exponential(1.25))  
 ξ = rand(Normal(0.1, 0.05))  
 return GeneralizedExtremeValue(μ, σ, ξ)  
end  
  
function draw\_discount\_rate()  
 return rand(Normal(0.05, 0.03))  
end  
  
function draw\_sow()  
 slr = rand(slr\_scenarios)  
 surge\_params = draw\_surge\_distribution()  
 discount = draw\_discount\_rate()  
 return SOW(slr, surge\_params, discount)  
end

Finally we can sample the SOWs. This section remains constant to provide comparable results with the original optimization problem.

Random.seed!(421521)  
N\_SOW = 10\_000  
N\_SOW\_opt = 100 # to start  
sows = [draw\_sow() for \_ in 1:N\_SOW]  
sows\_opt = first(sows, N\_SOW\_opt)

Include data from depthdamage.jl. This allows for the use of DepthDamageData parameter. This is a data structure or type that stores the depth-damage data, as well as any relevant metadata. This is extremely useful as the depths are stored in a complicated format (e.g., “ft04m” means -4 feet)

include("HouseElevation/src/depthdamage.jl")

Define haz\_fl\_dept and obtain data from DepthDamageData with row from haz\_fl\_dept

haz\_fl\_dept = CSV.read("data/haz\_fl\_dept.csv", DataFrame)  
demo\_row = @rsubset(  
 haz\_fl\_dept, :Description == "one story, Contents, fresh water, short duration"  
)[  
 1, :,  
]  
dd = DepthDamageData(demo\_row)

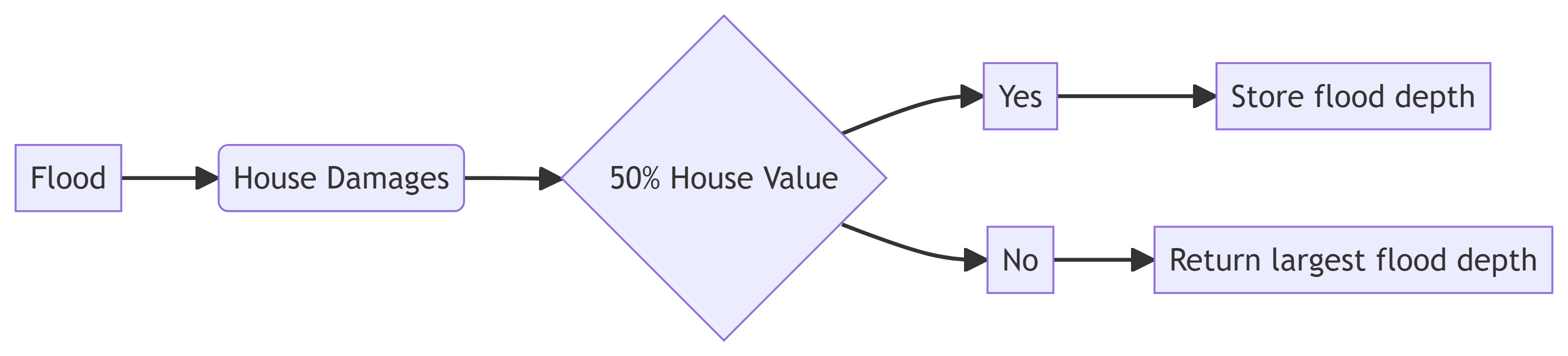
This function calculates the slope and acceleration of flood depth from haz\_fl\_dept data of the building.

function calculate\_slope\_acceleration(haz\_fl\_dept::DataFrame)  
 # Calculate the slope and acceleration of flood depth  
 flood\_depths = haz\_fl\_dept[!, :ft01]  
 slope = mean(diff(flood\_depths))  
 acceleration = mean(diff(diff(flood\_depths)))  
 return slope, acceleration, flood\_depths  
end

This function calculates damage using flood depth and the DepthDamageData parameter using linear interpolation.

function calculate\_damage(dd::DepthDamageData, flood\_depths::Vector{<:Number})  
 # Create an interpolation object based on dd.depths and dd.damages  
 interp = LinearInterpolation(dd.depths, dd.damages, extrapolation\_bc=Flat())  
   
 # Calculate damages for each flood depth  
 damages = interp.(flood\_depths)  
   
 return damages  
end

This function calculates the buffer height from depth damage, house value, and the slope and acceleration of sea level rise/flood depths.



This function follows the above flow diagram to collect data on the flood depth, which will be used to calculate the recommended buffer height. Basically, if flood damages exceed 50% of the house value, then that flood depth is stored for that year. However, if it does not exceed 50%, then the function just stores the largest flood depth from the dataset.

function calculate\_buffer\_height(dd::DepthDamageData, house\_value::Float64, slope::Float64, acceleration::Float64, flood\_depths::Vector{<:Number})  
  
 # Make slope and acceleration have units of ft/yr and ft/yr^2, respectively  
 slope = slope \* u"ft/yr"  
 acceleration = acceleration \* u"ft/yr^2"  
  
 # Define the damage threshold as 50% of the house value (FEMA)  
 damage\_threshold = 0.5 \* house\_value  
  
 # The time period is 30 years, which is the same as the original optimization problem  
 t = (2083 - 2024) \* u"yr"  
  
 # Calculate the change in flood depth due to the slope and acceleration. Used distance formula (assumption)  
 change\_in\_depth = slope\*t + 0.5\*acceleration\*t^2  
  
 # Adjust the flood depths based on the change in depth calculated above  
 flood\_depths = flood\_depths .+ change\_in\_depth  
  
 # Calculate damages for the flood depth  
 damages = calculate\_damage(dd, flood\_depths)  
  
 # Find the flood depth at which the damage exceeds the damage threshold (50%)  
 for (flood\_depth, damage) in zip(flood\_depths, damages)  
 if damage >= damage\_threshold  
 return flood\_depth  
 end  
 end  
  
 # If no flood depth exceeds the damage threshold, return the maximum flood depth from the data given  
 return maximum(flood\_depths)  
end

This calls the above functions to find buffer height. Freeboard height is set from historical data and guidance from [FEMA](https://www.fema.gov/case-study/home-above-rest-homeowner-city-galveston-elevates-home-12-feet). After Hurricane Ike, FEMA sent a forensic engineering team (a Mitigation Assessment Team [MAT]) to Galveston and the surrounding areas. The MAT recommended that houses be elevated 3 feet above the bare minimum required, which is the freeboard height for this project.

slope, acceleration, flood\_depths = calculate\_slope\_acceleration(haz\_fl\_dept)  
flood\_depths\_ft = flood\_depths .\* u"ft"  
buffer\_height = calculate\_buffer\_height(dd, Float64(250\_000), slope, acceleration,flood\_depths\_ft)  
freeboard\_height = 3\*u"ft"

Calculate cost of elevating buffer and freeboard heights. Used prices from a local contractor called [EHM](https://ehmoftexas.com/house-elevation-cost/) using slab foundation cost estimates and the provided area of the house. Additionally, this code assumes that square footage is based on ground surface area, not height, as it will be the same cost to lift.

function cost\_of\_action(buffer\_height,freeboard\_height)  
 buffer\_cost = buffer\_height\_free \* 2625 # calculate additional cost based on buffer height  
 freeboard\_cost = freeboard\_height\_free \* 2625 # calculate additional cost based on freeboard height  
 return buffer\_cost + freeboard\_cost  
end

Set up somes values to use in later functions that require dimensionless values.

# Modify run\_sim function to consider buffer height and freeboard height  
buffer\_height\_free = uconvert(u"ft", buffer\_height) |> ustrip  
freeboard\_height\_free = uconvert(u"ft", freeboard\_height) |> ustrip  
p = ModelParamsSA(; house=house, years=2024:2083, buffer\_height=buffer\_height\_free, freeboard\_height=freeboard\_height\_free)

Include data from run\_sim.jl. This allows for the use of trapz function in run\_sim.

include("HouseElevation/src/run\_sim.jl")

Redefine the run\_sim function in run\_sim.jl to include buffer and freeboard height costs

function run\_simulation(a::Action, sow::SOW, p::ModelParamsSA, buffer\_height\_free::Float64, freeboard\_height\_free::Float64)   
 # first, we calculate the cost of elevating the house  
 construction\_cost = elevation\_cost(p.house, a.Δh\_ft) + cost\_of\_action(buffer\_height\_free,freeboard\_height\_free)  
  
 # we don't need to recalculate the steps of the trapezoidal integral for each year  
 storm\_surges\_ft = range(  
 quantile(sow.surge\_dist, 0.0005); stop=quantile(sow.surge\_dist, 0.9995), length=130  
 )  
  
 eads = map(p.years) do year  
  
 # get the sea level for this year  
 slr\_ft = sow.slr(year)  
  
 # Compute EAD using trapezoidal rule  
 pdf\_values = pdf.(sow.surge\_dist, storm\_surges\_ft) # probability of each  
 depth\_ft\_gauge = storm\_surges\_ft .+ slr\_ft # flood at gauge  
 depth\_ft\_house = depth\_ft\_gauge .- (p.house.height\_above\_gauge\_ft + a.Δh\_ft) # flood @ house  
 damages\_frac = p.house.ddf.(depth\_ft\_house) ./ 100 # damage  
 weighted\_damages = damages\_frac .\* pdf\_values # weighted damage  
 # Trapezoidal integration of weighted damages  
 ead = trapz(storm\_surges\_ft, weighted\_damages) \* p.house.value\_usd  
 end  
  
 years\_idx = p.years .- minimum(p.years)  
 discount\_fracs = (1 - sow.discount\_rate) .^ years\_idx  
 ead\_npv = sum(eads .\* discount\_fracs)  
 return -(ead\_npv + construction\_cost)  
end

### 3.1.2 Optimization

We next need an objective function. Recall that we want to *maximize* NPV, but the optimization package we are using is set up to *minimize*.

function objective\_function(Δh::Vector{Float64})  
 a = Action(Δh[1])  
 npvs = [run\_simulation(a, sow, p, p.buffer\_height, p.freeboard\_height) for sow in sows\_opt]  
 return -mean(npvs)  
end

Since I’m using more SOWs here, I’ll also increase the time limit for the optimization to three minutes.

options = Options(; time\_limit=180.0, f\_tol\_rel=10.0)

We have a single decision variable, the height of the house above the ground. This can be any real number between 0 and 14 feet. The ECA algorithm is suggested as a default, so we’ll use that.

# 4. Results

# Define the optimization problem  
bounds = boxconstraints(; lb=[0.0], ub=[14.0])  
  
# Define the algorithm  
algorithm = ECA(; options=options)  
  
# Optimize  
result2 = optimize(objective\_function, bounds, algorithm)

Optimization Result  
===================  
 Iteration: 150  
 Minimum: 357415  
 Minimizer: [10.1205]  
 Function calls: 1050  
 Total time: 113.7350 s  
 Stop reason: Due to Convergence Termination criterion.

display(minimum(result2))  
display(minimizer(result2))

357415.0411468342

1-element Vector{Float64}:  
 10.12050258830505

## 4.1 Validation

As we have seen in labs, mistakes are inevitable and can lead to misleading results.

To minimize the risk of errors making their way into final results, it is essential to validate the implemented feature.

Describe the validation techniques used to ensure the accuracy and reliability of your implemented feature.

Discuss any challenges faced during the validation process and how they were addressed.

In this case, we don’t really *need* optimization – we can use brute force. We can compare by plotting the objective function for a range of elevations (from 0 to 14 ft) using all SOWs.

elevations\_try = 0:0.5:14  
actions\_try = Action.(elevations\_try)  
freeboard\_height\_free = float(freeboard\_height\_free)  
  
N\_more = 500  
npvs\_opt = [mean([run\_simulation(a, sow, p, buffer\_height\_free, freeboard\_height\_free) for sow in sows\_opt]) for a in actions\_try]  
npvs\_moore = [  
 mean([run\_simulation(a, sow, p, buffer\_height\_free, freeboard\_height\_free) for sow in first(sows, N\_more)]) for a in actions\_try  
]

Here, I wanted to see if taking no action (e.g., not considering buffer and freeboard heights when elevating) would lead to less NPV or profit for the homeowner. This was a simple check which underscored that considering buffer and freeboard height can help homeowners prepare for future conditions and save money in terms of future flood damages.

# Scenario with buffer and freeboard height  
buffer\_height\_free = float(buffer\_height\_free)  
freeboard\_height\_free = float(freeboard\_height\_free)  
action\_optimal = Action(minimizer(result2)[1])  
npv\_optimal = mean([run\_simulation(action\_optimal, sow, p, buffer\_height\_free, freeboard\_height\_free) for sow in sows\_opt])  
  
# Scenario without buffer and freeboard height  
action\_sea\_level = Action(0) # Assuming sea\_level is defined  
npv\_sea\_level = mean([run\_simulation(action\_sea\_level, sow, p, 0.0, 0.0) for sow in sows\_opt])  
  
println("NPV with buffer and freeboard height: ", npv\_optimal)  
println("NPV without buffer and freeboard height: ", npv\_sea\_level)

NPV with buffer and freeboard height: -357415.0411468342  
NPV without buffer and freeboard height: -406770.070311073

Now, I attempt to plot the aforementioned concepts of NPV with and without the buffer and freeboard heights.

elevations\_try = 0:0.5:14  
actions\_try = fill(Action(0), length(elevations\_try))  
  
N\_more = 500  
npv\_sea\_level = [mean([run\_simulation(a, sow, p, 0.0, 0.0) for sow in sows\_opt]) for a in actions\_try]  
npv\_sea\_level\_moore = [  
 mean([run\_simulation(a, sow, p, 0.0, 0.0) for sow in first(sows, N\_more)]) for a in actions\_try  
]

plot(  
 elevations\_try,  
 npvs\_opt ./ 1000;  
 xlabel="Elevation [ft]",  
 ylabel="NPV [1000 USD]",  
 label="Adaptive Policy: First $(N\_SOW\_opt) SOWs",  
 marker=:circle,  
 guidefontsize=8,  
 legendfontsize=8  
  
)  
plot!(elevations\_try, npvs\_moore ./ 1000; label="Adaptive Policy: First $(N\_more) SOWs", marker=:circle)  
  
plot!(elevations\_try, npv\_sea\_level ./ 1000; label="Static Policy: First $(N\_SOW\_opt)", marker=:square)  
plot!(elevations\_try, npv\_sea\_level\_moore ./ 1000; label="Static Policy: First $(N\_more)", marker=:square)  
vline!([minimizer(result2)]; label="Optimal", linestyle=:dash)

Key insights:

1. Our optimization appears to be working well, and maximizes the blue curve as it should.
2. There is a substantial difference between the blue and orange lines, indicating that using different SOWs (from the same distribution!) can make a big difference. Thus, the optimal elevation is highly sensitive to assumptions about the SOWs.
3. Going from zero (don’t elevate) to a small elevation is always bad, as you gain little flood protection but have to pay the fixed costs of elevation
4. Static policies (i.e., elevating once and/or not considering (changes in) buffer and freeboard hiehgts) are often not profitable as time goes on and higher elevations are needed.
5. Limitations:
   1. However, this static policy attempt is limited in its implementation. As this project does not have a static policy approach to how a homeowners or contractors may approach home elevations. In other words, action(0) is too simplistic.

The following functions attempt to perform a static policy of the above sequential analysis adaptive policy by changing the run\_simulation to only make one elevation. However, as mentioned previously, this requires many more assumptions that downplay the validation process. In other words, additional real-world data and methodology by companies is required to see a possible method to apply a static policy without considering buffer heights and freeboard heights.

#| output: false  
#| echo: false  
  
struct ActionTest{T<:Real}  
 Δh\_ft::T  
 n\_levels::Int  
end  
  
function ActionTest(Δh::T, n\_levels::Int) where {T<:Unitful.Length}  
 Δh\_ft = ustrip(u"ft", Δh)  
 return Action(Δh\_ft, n\_levels)  
end

#| output: false  
function run\_simulation\_static(a::ActionTest, sow::SOW, p::ModelParamsSA, buffer\_height\_free::Float64, freeboard\_height\_free::Float64)   
 # Create a copy of a  
 a\_copy = deepcopy(a)  
  
 # first, we calculate the cost of elevating the house  
 construction\_cost = elevation\_cost(p.house, a\_copy.Δh\_ft) + cost\_of\_action(buffer\_height\_free,freeboard\_height\_free)  
  
 # Initialize the total EAD NPV  
 total\_ead\_npv = 0.0  
  
 for year in p.years  
 # Add n\_levels in the first year  
 if year == 2024  
 a\_copy.Δh\_ft += a\_copy.n\_levels  
 end  
  
 # Do not recalculate the steps of the trapezoidal integral for each year  
 storm\_surges\_ft = range(  
 quantile(sow.surge\_dist, 0.0005); stop=quantile(sow.surge\_dist, 0.9995), length=130  
 )  
  
 # get the sea level for this year  
 slr\_ft = sow.slr(year)  
   
 # Compute EAD using trapezoidal rule  
 pdf\_values = pdf.(sow.surge\_dist, storm\_surges\_ft) # probability of each  
 depth\_ft\_gauge = storm\_surges\_ft .+ slr\_ft # flood at gauge  
 depth\_ft\_house = depth\_ft\_gauge .- (p.house.height\_above\_gauge\_ft + a\_copy.Δh\_ft) # flood @ house  
 damages\_frac = p.house.ddf.(depth\_ft\_house) ./ 100 # damage  
 weighted\_damages = damages\_frac .\* pdf\_values # weighted damage  
 # Trapezoidal integration of weighted damages  
 ead = trapz(storm\_surges\_ft, weighted\_damages) \* p.house.value\_usd  
  
 # Compute the EAD NPV for this year and add it to the total  
 ead\_npv = -(ead + construction\_cost)  
 total\_ead\_npv += ead\_npv  
 end  
  
 return total\_ead\_npv  
end

#| output: false  
  
function objective\_function\_static(Δh::Vector{Float64})  
 a = ActionTest(Δh[1], 2) # Pass Δh as the first argument to ActionTest  
 npvs = [run\_simulation\_static(a, sow, p, 0.0, 0.0) for sow in sows\_opt]  
 return -mean(npvs)  
end

# Optimize  
result3 = optimize(objective\_function\_static, bounds, algorithm)

This function attempts to use brute force to find an optimal buffer height, assuming a constant freeboard height set by Galveston. However, this function takes too long as the optimization for each buffer height takes around 1-2 minutes each.

#| output: false  
# Define ranges for buffer and freeboard heights  
using Distributed  
using SharedArrays # Add this line  
  
let   
 x = range(0; stop = 100, length = 100)  
 npvs\_array = SharedArray{Float64}(100) # Initialize a 1D shared array for npvs  
 for i=1:100  
 slope, acceleration, flood\_depths = calculate\_slope\_acceleration(haz\_fl\_dept)  
 flood\_depths\_ft = flood\_depths .\* u"ft"  
 buffer\_height = Float64(i)\*u"ft"  
 freeboard\_height = 3.0\*u"ft"  
  
 buffer\_height\_free = uconvert(u"ft", buffer\_height) |> ustrip  
 freeboard\_height\_free = uconvert(u"ft", freeboard\_height) |> ustrip  
 p = ModelParamsSA(; house=house, years=2024:2083, buffer\_height=buffer\_height\_free, freeboard\_height=freeboard\_height\_free)  
  
 bounds = boxconstraints(; lb=[0.0], ub=[14.0])  
 # Define the algorithm  
 algorithm = ECA(; options=options)  
 # Optimize  
 result3 = optimize(objective\_function, bounds, algorithm)  
 action = Action(minimizer(result3)[1])   
 npvs = mean([run\_simulation(action, sow, p, buffer\_height\_free, freeboard\_height\_free) for sow in sows\_opt])  
 npvs\_array[i] = npvs # Store npvs value in the array  
 end   
  
 plot(x, npvs\_array, xlabel="Buffer Height", ylabel="NPV") # Create a 2D plot  
end

# 5. Conclusions

## 5.1 Discussion

Analyze the implications of your results for climate risk management.

1. Overall Success
   1. Apart from the limitations discussed in a later section, this project provides a foundational base for future work.
   2. On a fundamental basis, this project showcases how adaptive policies in sequential analysis provide better value for a property owner looking to elevate their property against sea level rise. As sea level rise remains uncertain in certain factors, calculating and utilizing buffer and freeboard heights provide an extra layer of safety for properties against flood damages. Moreover, constantly updating these heights as time goes on provides more accurate information for decision makers.
   3. From this, future projects can look into applying adaptive elevations that allow property owners to increase heightenings when needed, instead of a static one-time elevation.
2. On a broader scale, this project has wide implications for sequential analysis and climate risk management, apart form elevating properties.
   1. Adaptive Decision-Making:
      1. Real-Time Adaptation: By using sequential analysis, climate risk assessments can adapt in real time to changing conditions, allowing decision-makers to respond quickly to emerging risks, such as sea-level rise.
      2. Dynamic Policies: Direct policy search enables the optimization of adaptive policies. For instance, adjusting buffer heights of coastal infrastructure based on observed sea-level trends ensures resilience against sea-level rise.
      3. Adaptive Pathways: Sequential analysis identifies adaptive pathways. These pathways guide long-term planning, ensuring that climate risk management remains effective over time.
   2. Resource Allocation and NPV:
      1. Robust Decisions: Sequential analysis provides useful information on where to allocate resources. To illustrate, it can guide decisions on flood protection infrastructure, considering factors like asset vulnerability and climate projections. Moreover, integrating NPV calculations ensures that investments yield long-term benefits.
   3. Risk Communication and Stakeholder Engagement:
      1. Transparent Risk Assessment: Sequential analysis provides a transparent process for assessing climate risks. From this, stakeholders and decision makers can understand how decisions are made and contribute to risk reduction strategies. In turn, direct policy search allows stakeholders to participate in policy optimization, enhancing the effectiveness of adaptation measures.
   4. Trade-Offs and Uncertainties:
      1. Balancing Trade-Offs: Climate risk management involves trade-offs, and sequential analysis helps weigh the costs and benefits of different adaptation options, considering uncertainties.
      2. Uncertainty: Direct policy search can incorporate uncertainty estimates into decision-making. It accounts for variability in climate projections, economic factors, and technological advancements. Although this project does not dive deep into uncertainties of certain SOWs, this project’s foundation allows for this future work to be done.
   5. Monitoring:
      1. Monitoring and Feedback: Direct policy search requires ongoing monitoring. Regular feedback loops ensure that policies remain effective and can be adjusted as needed.
3. In summary, integrating sequential analysis and direct policy search enhances climate risk management by enabling adaptive decision-making, optimizing resource allocation, engaging stakeholders, addressing uncertainties, and promoting long-term resilience. All in all, these implications contribute to building robust and sustainable communities in the face of severe climate change.

Consider the context of the class themes and discuss how your findings contribute to the understanding of climate risk assessment.

1. Climate Risk Assessment:
   1. Climate risk assessments are essential for cities and communities to identify the likelihood of future climate change hazards and their potential impacts.
   2. Key components of a climate risk assessment include:
      1. Understanding the city’s demographic, socio-economic, and environmental factors.
      2. Analyzing historical climate data.
      3. Examining and modeling climate change trends and future scenarios.
      4. Mapping climate risks and vulnerabilities.
      5. Identifying priority risks based on exposure and vulnerability.
2. Sequential Analysis:
   1. Sequential analysis refers to evaluating data or making decisions sequentially over time, often with new resources or information available at each time step.
   2. In the context of climate risk assessment, sequential analysis can be applied to assess risks as they unfold, considering evolving climate conditions and impacts.
3. Direct Policy Search:
   1. Direct policy search finds optimal policies by directly optimizing their parameters. It is quite useful in complex reinforcement learning problems.
   2. However, traditional classification-based optimization requires batch sampling of solutions, which is why DPS outperforms the traditional approach in terms of Pareto-dominance (e.g., balancing conflicting objectives).
4. Combination:
   1. Combining these concepts can enhance climate risk assessment:
      1. Adaptive Risk Assessment: Sequential analysis allows for real-time adaptation to changing climate conditions, updating risk assessments as new data becomes available.
      2. Policy Optimization for Adaptation: Direct policy search can optimize adaptation strategies. For instance, it can help determine optimal buffer and freeboard heights for flood protection infrastructure.
      3. NPV and Decision-Making: Considering net present value (NPV) in decision-making ensures robustness. Sequential analysis can guide adaptive investments, while direct policy search optimizes resource allocation.
5. Practical Example:
   1. A coastal city facing sea-level rise due to climate change.
   2. Sequential analysis monitors sea-level trends, storm surges, exposure, and vulnerabilities.
   3. Direct policy search optimizes investment decisions:
   4. Buffer heights for seawalls: Sequentially adjust buffer heights based on observed sea-level rise.
   5. Freeboard requirements for buildings: Optimize building designs to withstand future flooding.
   6. NPV calculations guide resource allocation: Invest in cost-effective measures with higher NPV.
6. In summary, integrating sequential analysis with direct policy search enhances climate risk assessment by adapting to changing conditions, optimizing adaptation strategies, and ensuring robust decisions. By considering NPV, cities can prioritize investments and solutions that yield long-term benefits in the face of climate change

Identify any limitations of your approach and suggest potential improvements for future work.

1. Freeboard and Buffer Height Calculations
   1. The Freeboard assumes a constant value of 3 feet, which may not be robust enough as conditions change. Finding a possible method to update freeboard height based on local conditions is essential to increase the robustness of the decision.
   2. The cost of Buffer Height and Freeboard Height is assumed from a local contractor in Galveston based on the area of the house with some assumptions, mainly that square footage is based on ground surface area, not height. Collecting data from more local contractors will allow for a more accurate calculation of costs by averaging the prices offered by these contractors.
   3. The calculation of the slope and acceleration of sea level rise, which is used by the buffer height calculations, is based on a general distance equation. In other words, this is a best estimate for how the sea level will rise based on data from the haz\_fl\_dept data. Thus, future iterations may need to consider additional datasets in Galveston to obtain historical sea level rise’s slope and acceleration data.
   4. The calculation of the buffer height from the flood depth relies on the damage threshold being 50% or greater than the house value. Thus, some flood depths that do not reach this damage threshold are ignored. However, these flood depths may be significant and be an issue for the property owner to continue operations or live in. Thus, future iterations may need to consider adapting this damage threshold based on a variety of factors that can change as conditions change (e.g., local sea level rise, current economic conditions, property owner preferences, etc.)
2. Runtime
   1. As the number of SOWs increases, the time to run the code exponentially increases. Future iterations should see how to optimize the code to decrease runtime. The current code is barebone and relies on many loops and repeats that may slow down the code. This is significant if we want to brute force the optimization to see if this project’s feature is properly working.
3. Uncertainties in SOWs remain
   1. As mentioned previously, there is a substantial difference between the blue and orange lines in the plot, indicating that using different SOWs (from the same distribution!) can make a big difference.
   2. Thus, the optimal elevation is highly sensitive to assumptions about the SOWs.
   3. Future iterations should consider how changing certain parameters of the SOWs can affect the results (i.e., perform a sensitivity analysis of the results). For example, how storm surge distributions are drawn, how discount rates are selected, and how states of the worlds are drawn from the dataset.

## 5.2 Conclusions

Summarize the key findings of your project and reiterate the significance of your implemented feature in addressing the problem statement.

1. Overall, from this project, I have successfully laid a foundation for the implementation of sequential analysis with direct policy search to create an adaptive policy that updates buffer and freeboard heights based on exposure/vulnerabilities and sea level trends. From the results, we can see that property owners gain more benefits by utilizing adaptive policies to consider buffer and freeboard heights, reducing costs and increasing safety. From this, future iterations can improve upon and scale this model up to perform over neighborhoods and cities, which can help inform policy makers to see what regions require immediate elevations for existing structures and plan for adaptive structures that can make future elevations.

Discuss the broader implications of your work for climate risk management and the potential for further research in this area.

1. As stated before, integrating sequential analysis and direct policy search enhances climate risk management by enabling adaptive decision-making, optimizing resource allocation, engaging stakeholders, addressing uncertainties, and promoting long-term resilience. By considering NPV, cities can prioritize investments and solutions that yield long-term benefits in the face of climate change. All in all, these implications contribute to building robust and sustainable communities in the face of severe climate change.
2. Moreover, as this project lays the bare framework for sequential analysis, future research is required before real world applications. Namely, the collection and analysis of real-world data such as costs to elevate and sea level rise acceleration will be important. This can not only benefit house elevation problems but also other wider scale issues like infrastructure improvements and recovery of essential services (e.g., fire department) after storms to name a few.
3. Moreover, for future additions to this project, analysis of how using buffer heights and freeboard heights can reduce storm damages to properties can help cement the utility of sequential analysis and adaptive policies. Apart from using models, real-world data can further help expand the benefits of sequential analysis to include decreasing storm damages, which in turn, will increase the NPV of elevating.
4. Lastly, once these techniques and methods have been fine-tuned, combining sequential analysis and direct policy search with machine learning techniques (e.g., deep reinforcement learning) can develop AI-driven decision support systems and increase efficiency for climate risk management.

# 6. References

Garner, Gregory G., and Klaus Keller. 2018. “Using Direct Policy Search to Identify Robust Strategies in Adapting to Uncertain Sea-Level Rise and Storm Surge.” *Environmental Modelling & Software* 107 (September): 96–104. <https://doi.org/10.1016/j.envsoft.2018.05.006>.

Haasnoot, Marjolijn, Jan H. Kwakkel, Warren E. Walker, and Judith ter Maat. 2013. “Dynamic Adaptive Policy Pathways: A Method for Crafting Robust Decisions for a Deeply Uncertain World.” *Global Environmental Change* 23 (2): 485–98. <https://doi.org/10.1016/j.gloenvcha.2012.12.006>.