An Introduction to Climate Risk Analysis

2023

Table of contents

Introduction

Greenhouse gas emissions have caused considerable changes in climate, including increased surface air temperatures and rising sea levels. These changes affect both the Earth and human systems in numerous ways, creating new risks and increasing previously existing ones. This book provides an introduction to some of the concepts and approaches used to quantify and analyze climate risks to help support decision-making for risk management.

This book originated as Risk Analysis in the Earth Sciences[1] and an associated Risk Analysis course at Penn State. *Risk Analysis in the Earth Sciences* has also been used in courses at the University of Illinois at Urbana-Champaign, CU Boulder, and RIT. This updated version expands on the original book by presenting a series of tutorials in the Julia programming language[2] that teach some of the Earth science and statistical concepts needed for assessing climate-related risks. These activities are intended for upper-level undergraduates, graduate students, and professionals in other areas who wish to gain insight into academic climate risk analysis.

Book Scope

Modern statistical and computational methods are essential for analyzing the complex and dynamic risks from climate change. Given the power of these tools, it is also important for practitioners to understand how to determine when a particular approach is appropriate for a given problem and to ensure that their conclusions reflect the choices made in modeling and analysis. This book is intended to introduce these skills in the context of particular applications, which serve as case studies.

The code examples in this book are provided in Julia, which is a powerful, open-source language for scientific computing and data analysis. Julia was designed to strike a balance between readability and speed, and has steadily gained popularity in the broader scientific and statistical communities.

> ⚠ Warning
>
> Completing the chapters in this textbook will not teach you to perform publication-quality or consulting-grade risk analyses. This textbook is intended primarily for educational use, and the material presented here should only be applied to "real" problems after additional training. The authors and editors of this textbook specifically

[1] https://www.scrim.psu.edu/resources/raes/
[2] https://julialang.org/

> disclaim any liability for damages associated with the use of the material presented in this textbook.

Learning Objectives

After completing the material in this book, students should be able to:

Part I

About This Book

Acknowledgements

This work was partially supported by the National Science Foundation through the Network for Sustainable Climate Risk Management (SCRiM) under NSF cooperative agreement GEO-1240507. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Other support was provided by the Center for Climate Risk Management[3] and the Rock Ethics Institute[4].

Most of these exercises were originally developed as part of a course titled Risk Analysis in the Earth Sciences[5], developed and taught by Klaus Keller in the Department of Geosciences[6] at the Pennsylvania State University[7]. The exercises discussed in this lab manual were co-developed by a large group (see the individual chapters).

[3] http://www.clima.psu.edu/index.php
[4] http://rockethics.psu.edu
[5] http://bulletins.psu.edu/undergrad/courses/G/GEOSC/450/200910SP
[6] http://www.geosc.psu.edu
[7] http://www.psu.edu

Part II

Introduction to Coastal Flood Risk

Many people live near present-day sea level (Robert J. Nicholls et al., 2008; Robert J. Nicholls et al., 2021), and rises in sea level expose these people to the possibility of flooding. For example, the ongoing increase in sea level will likely cause communities on the United States' eastern coast to experience frequent flooding within the next few decades (Spanger-Siegfried et al., 2014).

Sea level rise is caused by temperature increases, which in turn are driven by increases in carbon dioxide concentrations in the atmosphere. Carbon dioxide is produced by human activities and natural processes. Increases in carbon dioxide concentrations in the atmosphere enhance the trapping of infrared radiation near the Earth's surface and contribute to rises in surface air temperatures. As the ocean absorbs some excess heat from the atmosphere, its temperature increases, causing it to expand and causing sea level rise. Temperature increases cause melt of glaciers and ice sheets, which leads to sea level rise by adding mass to the oceans.

Data covering the last century support this relationship between atmospheric carbon dioxide concentrations, temperature, and sea level (Figure 1). The curves in the three panels of Figure 1 rise together, suggesting that these variables are related.

> **!** Important
>
> Although correlation does not prove causation, the combination of a clear relationship between variables with a plausible explanation for why they should be related is *evidence* for causation.

Because carbon dioxide mixes readily in the atmosphere, measurements of atmospheric carbon dioxide concentrations at most places on the Earth's surface are relatively representative of the globe as a whole. In contrast, both surface air temperatures and sea levels are measured at widely dispersed stations and must be aggregated to give a global mean value. Global mean temperatures must be estimated from individual weather stations with long records (Hansen et al., 2010); past sea levels are estimated using data from tide gauges (Jevrejeva et al., 2014). As one might expect, there are various methods for performing this aggregation, and the different methods give somewhat different answers. However, it seems clear that both global mean surface air temperature and global mean sea level are rising.

At the same time, precipitation rates and tropical cyclone intensities are likely to increase as the climate warms (Walsh et al., 2016). This suggests that the occurrence frequencies of extreme storm surges could also change. The combination of increasing sea levels and changing storm surge frequencies would change the distribution of extreme water levels, and therefore impact the efficacy of flood risk management strategies. In this section, we will explore some tools for quantifying and assessing the future impacts of climate change on coastal flood risk.

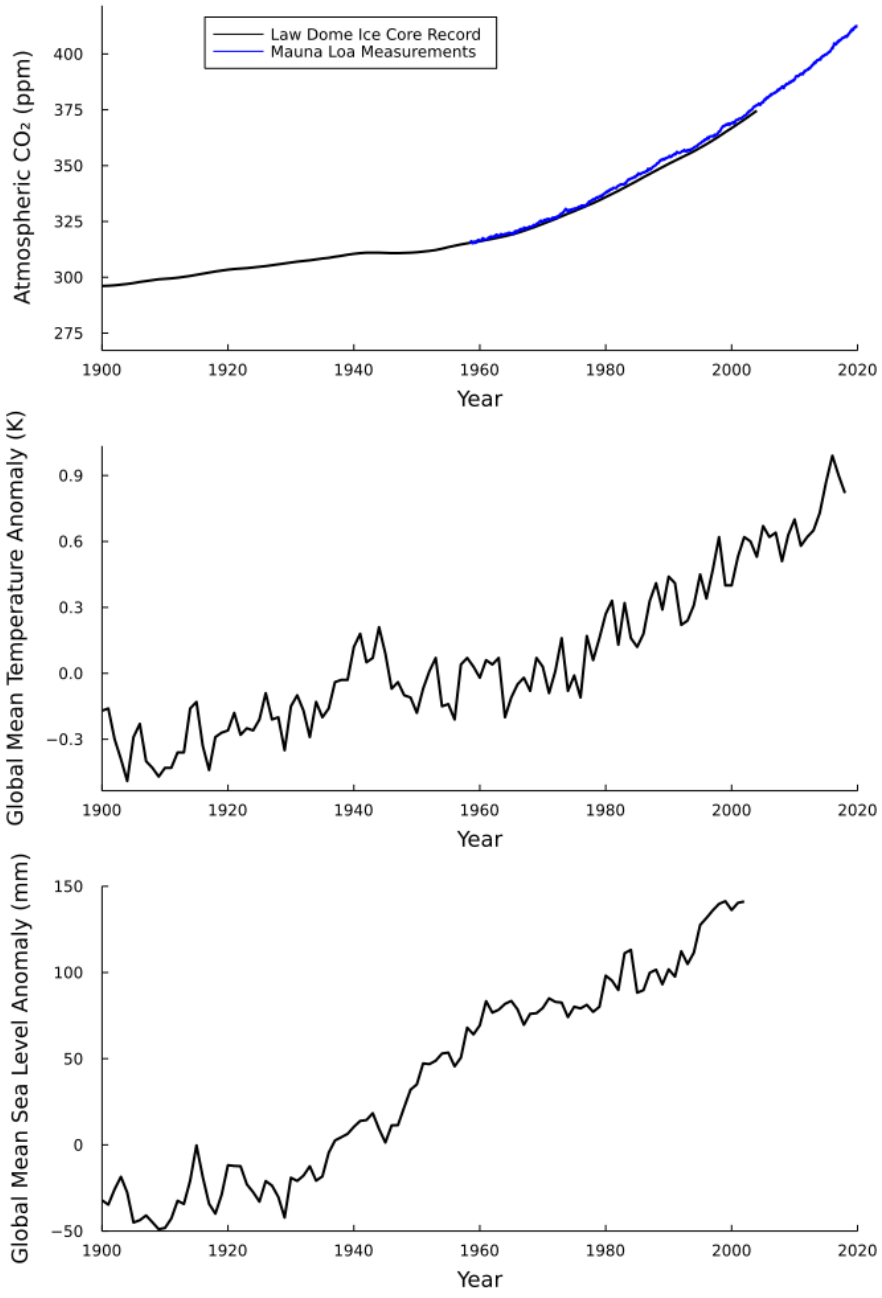

Figure 1: Atmospheric carbon dioxide concentrations (top panel), surface air temperature change (middle panel), and sea level change (bottom panel), between 1900 and ~2015. All three quantities rise over this period, possibly suggesting a causal relationship between them given a plausible theoretical connection. See text for discussion.

Chapter 1

Modeling Sea Level Rise

We often have to make inferences about system behaviors using limited data. This problem can be especially acute in climate science, where observational records of temperature and sea level rise (for example) go back a few hundred years at most. Given these limited data, how can we make inferences about the past and future of the climate system, while accurately representing our uncertainties?

In this section, we will develop and calibrate a model of global sea-level rise. In Chapter 2, we will discuss several choices for a model structure. In Chapter 3, we will propose several statistical models to represent deviations between our chosen model and historical observations, also called *residuals* or *discrepancies*.

Finally, we will learn about and apply frequentist and Bayesian approaches to parameter inferences in **?@sec-slr-bootstrap**) and **?@sec-slr-mcmc**), respectively.

The statistical model used for residuals plays an important role in sound parameter inferences and projections (Brynjarsdóttir & O'Hagan, 2014).

Chapter 2

Selecting A Sea Level Model

2.1 Candidate Models

A key consideration in any modeling exercise is the mathematical representation of the modeled process. This choice can allow or prohibit certain relationships between variables, or may result in too many or too few parameters to appropriately capture dynamics. For example, a simple linear model in time,

$$H_{\text{lin}}(t) = a + b * (t - t_0),$$

where H is global sea level in m, t is time in years and t_0 is a baseline year, does not allow for accelerated sea-level rise. A quadratic model in time,

$$H_{\text{quad}}(t) = a + b * (t - t_0) + c * (t - t_0)^2,$$

does allow for this acceleration.

However, both of these models assume that time is the key variable explaining changes in sea levels from one year to another. In , we discussed the relationship between CO_2 concentrations, atmospheric temperature, and sea levels. Rahmstorf (2007) proposed the following semi-empirical model relating sea-level rise (SLR) to changes in global mean temperatures:

$$\Delta H_{\text{emp}}(t) = \alpha \left(T(t) - T_{\text{eq}} \right),$$

where α is the sensitivity of SLR to temperature changes, and T_{eq} is the temperature when sea level is at equilibrium, that is, $H_{\text{emph}}(t) = H_{\text{emp}}(t-1)$ when $T(t) = T_{\text{eq}}$. In the next section, we will look at how we can find best-fit parameter values for these models to sea-level data and analyze the dynamics of these fitted models.

2.2 Loading and Plotting The Data

To fit the three models described in , we need to load historical SLR data (for all three models) and global mean temperature (GMT) data (for H_{emp}). We have provided two data sets in contents/flood/data.

2.2.1 Reading Data Files

The SLR data file, CSIRO_Recons_gmsl_yr_2015.csv[1] has three columns: 1. Time in years (the fractions correspond to months); 2. Global mean sea-level in mm (relative to the 1961-1990 mean); 3. Standard deviation of the observational error in mm.

The GMT data[3] has a similar structure: a header and comma-delimited. This file, HadCRUT.5.0.1.0.analysis.summary_series.global.annual.csv, also has four columns, along with a header: 1. Time in years; 2. Annual mean temperature anomaly (relative to the 1961-1990 mean); 3. The lower end of the 95% confidence interval; 4. The upper end of the 95% confidence interval.

In Julia, we can read in these files using the DelimitedFiles package[5]. The DelimitedFiles.readdlm function takes in a filename, and can also take in optional parameters like a delimiter and an output type. We can also convert the output of DelimitedFiles.readdlm to a DataFrame[6].

```
# load packages
using DataFrames
using DelimitedFiles

dat, head = readdlm("data/CSIRO_Recons_gmsl_yr_2015.csv", ',', header=true);
# use the header vector for the DataFrame column names
slr_data = DataFrame(dat, vec(head));
first(slr_data, 6)
```

	Time	GMSL (mm)	GMSL uncertainty (mm)
	Float64	Float64	Float64
1	1880.5	-158.7	24.2
2	1881.5	-153.1	24.2
3	1882.5	-169.9	23.0
4	1883.5	-164.6	22.8
5	1884.5	-143.7	22.2
6	1885.5	-145.2	21.9

Since we want to read multiple files with the same structure, let's write a function which takes in filename corresponding to a CSV file with a header and returns a DataFrame.[7] In this case, this is overkill, because we only have one more file we want to read in, but if we had many files, writing a single function would clean up the code and facilitate debugging.

```
function read_csv(fname)
    dat, head = readdlm(fname, ',', header=true) # read in the data and header
    return DataFrame(dat, vec(head)) # return the DataFrame
end

read_csv (generic function with 1 method)
```

[1] The SLR dataset was taken from Australia's Commonwealth Scientific and Industrial Research Organization (CSIRO)[2] and described in [@churchSealevelRiseLate2011]. This reconstruction of global mean sea levels spans 1880–2013.

[3] GMT data was obtained from the HadCRUT5 website[4] (Morice et al., 2021).

[5] https://docs.julialang.org/en/v1/stdlib/DelimitedFiles/

[6] https://dataframes.juliadata.org/stable/

[7] If we had noticed that our files had different structures (headers, initial lines to skip, etc.), we can either write a more complex function which allows us to change these settings, or the readdlm calls might be sufficiently different that this wouldn't be clearer or easier to debug.

Now we'll use read_csv to read in the GMT data.

```
gmt_file = "data/HadCRUT.5.0.1.0.analysis.summary_series.global.annual.csv";
gmt_data = read_csv(gmt_file);
first(gmt_data, 6)
```

	Time	Anomaly (deg C)	Lower confidence limit (2.5%)	Upper confidence limit (97.5%)
	Float64	Float64	Float64	Float64
1	1850.0	-0.417659	-0.589203	-0.246115
2	1851.0	-0.23335	-0.411868	-0.0548317
3	1852.0	-0.229399	-0.409382	-0.0494157
4	1853.0	-0.270354	-0.430009	-0.1107
5	1854.0	-0.29163	-0.432824	-0.150436
6	1855.0	-0.296951	-0.439358	-0.154545

2.2.2 Merging The Data

Now let's merge (or "join"[8]) the two DataFrames on common years. The DataFrames documentation[9] provides information on these, as well as some other, more specialized, joins.] Then we can plot the two data series.

There are many types of joins. Our goal is to focus on the SLR data, so let's use an "left join" (which includes all entries from the first listed DataFrame and entries from the second DataFrame which correspond to matching key values). First, we need to correct the "Time" column in slr_data, as those values end with a non-zero decimal.

Database-style joins allow a flexible way to merge two DataFrames (and other tabular data structures) on a common column (the "key"). There are several different styles of joins: * *Inner joins* only return rows with key values which occur in all of the merged databases; * *Outer joins* returns all rows with key values which occur in any merged database; * *Left joins* return rows with key values occurring in the first (left) database; * *Right joins* return rows with key values occurring in the second (right) database.

```
slr_data[:, :Time] = slr_data[:, :Time] .- 0.5; # remove 0.5 from Times
all_data = leftjoin(slr_data, gmt_data, on="Time"); # outer join the data frames on Tim
first(all_data, 6)
```

	Time	GMSL (mm)	GMSL uncertainty (mm)	Anomaly (deg C)	Lower confidence limit (2.5%)	
	Float64	Float64	Float64	Float64?	Float64?	
1	1879.0	-158.7	24.2	-0.303634	-0.434064	…
2	1880.0	-153.1	24.2	-0.315832	-0.440151	…
3	1881.0	-169.9	23.0	-0.232246	-0.357935	…
4	1882.0	-164.6	22.8	-0.29553	-0.420815	…
5	1883.0	-143.7	22.2	-0.346474	-0.460818	…
6	1884.0	-145.2	21.9	-0.49232	-0.602669	…

2.2.3 Plotting The Data

Now we can plot the observations. Since we merged the DataFrame, we can pass two of the columns to Plots.plot and a layout, and it will automatically plot them as subplots (though we do have to convert the y-values to a Matrix).

```
using Plots

# plot reconstructed anomalies as points
scatter(all_data[:, 1], Matrix(all_data[:, [2, 4]]), grid=:false, markersize=3, xlabel="Ye
```

[8] https://dataframes.juliadata.org/stable/man/joins/
[9] https://dataframes.juliadata.org/stable/man/joins/

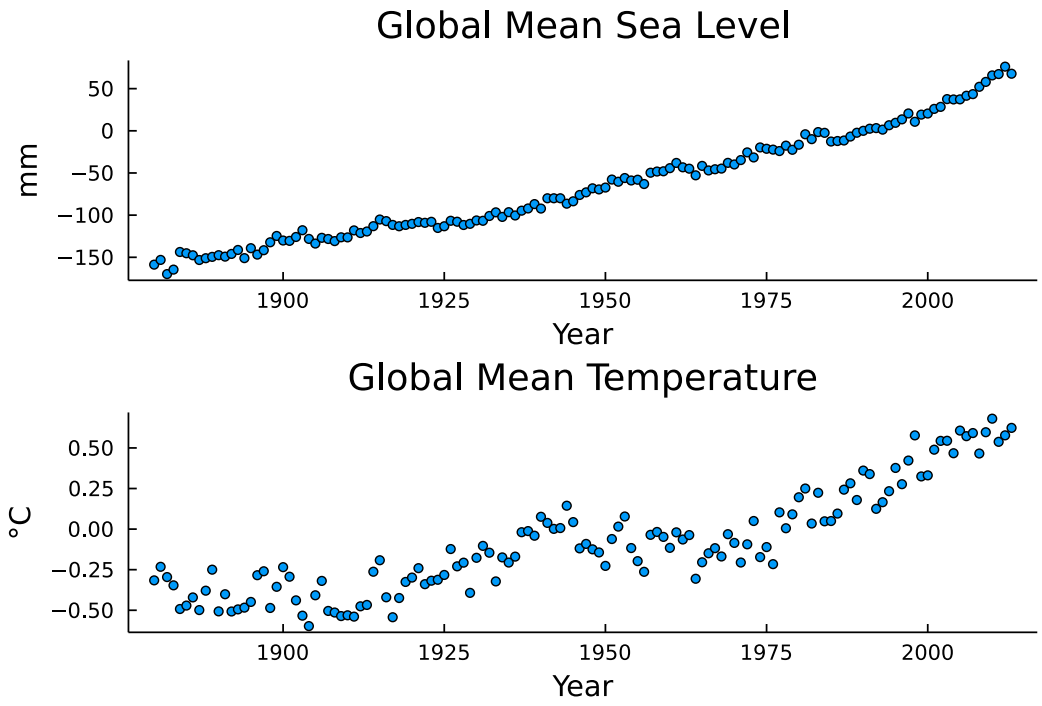

Figure 2.1: Plotted anomalies relative to 1961-1990 mean.

2.3 Fitting the Models to the Data

Now we can write functions to predict the SLR using the three candidate models.

```
# H_lin
function H_lin(a, b, t)
    slr_predict = a .+ (b .* (t .- t[1]))
    return slr_predict
end

# H_quad
function H_quad(a, b, c, t)
    slr_predict = a .+ (b .* (t .- t[1])) + (c .* (t .- t[1]).^2)
    return slr_predict
end

# H_emp
function H_empα(, ₀T, ₀H, temp)
    temp_effect = α .* (temp .- ₀T)
    slr_predict = cumsum(temp_effect) .+ ₀H
    return slr_predict
end
```

Let's also pull out the vectors of data that we will need to make our code more readable (and therefore easier to debug).

```
years = all_data[:, 1];
sealevels = all_data[:, 2];
```

```
temps = all_data[:, 4];
```

To find parameter values for the models, we will minimize the root-mean-square-error (RMSE):

$$\text{RMSE} = \sqrt{\sum_{t=1}^{T} \left(\text{pred}_t - \text{obs}_t\right)^2},$$

where t is the time index, pred_t is the model prediction, and obs_t is the data value.

We can find parameter values which minimize the RMSE using Optim.jl[10]. The Optim.optimize function minimizes a function value using one of several numerical solvers. Optim.optimize wants its input function to accept a vector of proposed parameter values and return an output which is to be minimized. This means that we need to construct a function which accepts a parameter vector and passes the relevant values to the target function, before calculating the RMSE. This "wrapper" function is what we will want to pass to Optim.optimize.

We can do this using the following code. First, let's define a function for the RMSE.

```
# the mean function is in the Statistics package, not built in
using Statistics

# y are the data, ŷ are the model predictions
rmse(y, ŷ) = sqrt(mean((y - ŷ).^2))

rmse (generic function with 1 method)
```

Next, we want to write our wrapper function. The function should accept a vector of parameters and the data, unpack the parameter vector and evaluate the function at those parameters, and then return the RMSE. The "tricky" part is unpacking the parameter vector. There are two approaches to this that we could take. The first is to unpack the vector manually, for example,

```
a, b = v
```

assigns a = v[1] and b = v[2].

The second is more versatile, but less readable: we can use the "splat" operator ...[11].[12] This operator unpacks the elements of the vector which it acts on within a function call, so f(v ...) gets interpreted as f(v[1], v[2]). This type of call can be combined with other arguments, as in:

```
function f(a, b, c)
    return a + b + c
end

v = [1.0, 2.0]
```

[10] https://julianlsolvers.github.io/Optim.jl/stable/#

[11] https://docs.julialang.org/en/v1/base/base/#...

[12] The tradeoff between the two approaches is that the use of the splat operator is more versatile, as we don't need to know how many elements are in the vector ahead of time or their names, while manually unpacking the vectors is more readable. A good practice would be to manually unpack vectors into named variables when working with a specific function. However, in this case, we want to minimize the RMSE of three functions, and we'd have to write a different wrapper for each one given the varying number of parameters.

```
c = 3.0

f(v..., c)  # => f(1.0, 2.0, 3.0)
```

The last consideration is that each of H_lin, H_quad, and H_emp accepts an auxiliary data vector after the parameters: the time vector t for the first two, and temperatures temp for H_emp. This similarity means that we can write a single wrapper and pass the name of the function, the parameter vector, the auxiliary data, and the sea level data.

```
using Optim

function minimize_rmse(fn, params, aux, dat)
    # we can call the passed function
    predict = fn(params..., aux)
    return rmse(dat, predict)
end

minimize_rmse (generic function with 1 method)
```

The only thing left to do is to call Optim.optimize on minimize_rmse with varying functions fn corresponding to our three SLR models. We can deal with the constant values for each call (fn, aux, and dat) by using anonymous functions to map the parameter vector proposed in a given solver iteration.

```
# H_lin
# this has two uncertain parameters (a, b), so we use a 2d initial vector [0.0, 1.0]
result_lin = optimize(params -> minimize_rmse(H_lin, params, years, sealevels), [0.0, 1.0]
params_lin = Optim.minimizer(result_lin);
@show params_lin;

params_lin = [-172.2866794430187, 1.5970719045709219]

# H_quad
# this has three uncertain parameters (a, b, c), so we use a 3d initial vector [0.0, 1.0, 
result_quad = optimize(params -> minimize_rmse(H_quad, params, years, sealevels), [0.0, 1.0
params_quad = Optim.minimizer(result_quad);
@show params_quad;

params_quad = [-153.94760089306058, 0.7634947234485565, 0.006267398158364036]

# H_emp
# unlike the other two models, we pass in temperatures as the auxiliary
# this has three uncertain parameters α(, ₀H, ₀T), so we use a 3d initial vector [1, 0, 0]
result_emp = optimize(params -> minimize_rmse(H_emp, params, temps, sealevels), [1.0, 0.0,
params_emp = Optim.minimizer(result_emp);
@show params_emp;

params_emp = [1.8636596824562823, -0.9711430993147957, -157.33666238607228]
```

2.4 Plotting the Fitted Models

Finally, let's plot the SLR hindcasts from all three fitted models to see how they perform.

```
hindcast_lin = H_lin(params_lin..., years);
hindcast_quad = H_quad(params_quad..., years);
hindcast_emp = H_emp(params_emp..., temps);
```

```
scatter(years, sealevels, color="black", alpha=0.7, label="Reconstructed Data", legend=:to
plot!(years, hindcast_lin, color="#66c2a5", linewidth=3, label="H_lin")
plot!(years, hindcast_quad, color="#fc8d62", linewidth=3, label="H_quad")
plot!(years, hindcast_emp, color="#8da0cb", linewidth=3, label="H_emp")
```

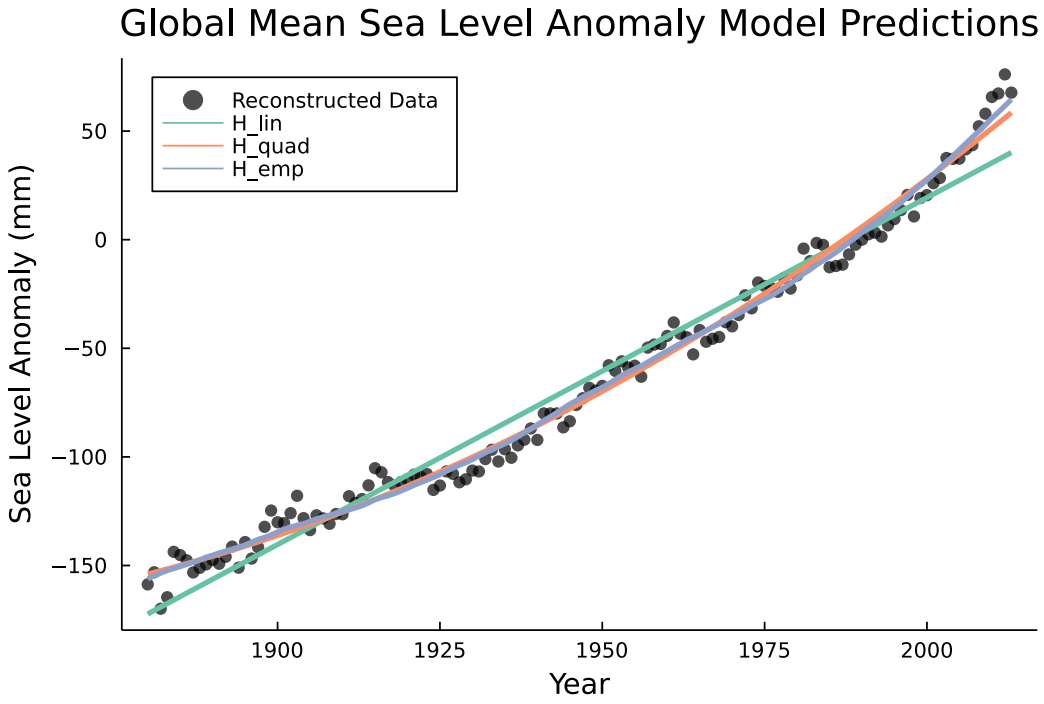

Figure 2.2: Outputs of the three fitted sea-level rise models vs. the reconstructed data. The reconstructed data are black dots, while the model fits are the colored lines.

We can see from Figure 2.2 that the linear model H_{lin} (green) fails to pick up the data trend at the start and end of the period (including the acceleration at the end of the data set, which makes sense), but the quadratic and semi-empirical models H_{quad} (red) and H_{emp} (purple) both have pretty similar outputs.

2.5 Selecting a Model

Comparing the RMSEs of the two models (which we can get from our optimization results using Optim.minimum):

```
quadratic_rmse = Optim.minimum(result_quad)
empirical_rmse = Optim.minimum(result_emp)
```

```
@show quadratic_rmse;
@show empirical_rmse;
```

```
quadratic_rmse = 6.637214473133741
empirical_rmse = 5.911184790905463
```

Since both of these models perform similarly, the choice between the two depends on what question you are asking. If you were primarily interested in inferring the historical temporal trend, H_{quad} might be more useful, since it makes the time-dependence explicit, while H_{emp} focuses on the dependence on GMT, which would be more useful for making projections given the causal link between warming and SLR.

The RMSE is slightly lower for H_{emp}, but we want to be careful to not over-interpret small differences. Visually, the main difference is that H_{emp} shows more of an increase in the last few years, which matches what appears to be an acceleration in the data. As a result, for the rest of this section, we will use H_{emp} for our SLR modeling, but you could justify using H_{quad}.

Chapter 3

Choosing A Likelihood Function

3.1 Model Calibration

Now that we have selected a sea-level rise (SLR) model, the next step in our analysis is to select parameter values. This procedure is called *model calibration*. In Chapter 2, we found values which minimized the root-mean-square error. This yielded individual parameter values, or *point estimates*, and so is an example of a deterministic approach to calibration. However, there are potential downsides to relying on point estimates. Most importantly, the use of point estimates results in a deterministic model output, with no representation of uncertainties. This is problematic, at one consequence is that we would end up with a single projection value of the global mean sea-level resulting from a given temperature trajectory. But we know that our model is very simplified, and does not perfectly match the data — why would we trust it to give us perfect values for the future?

Instead of putting too much faith in our simple model, we can instead view it as an *approximation*, which will give us insights into the approximate relationship between global mean temperature change and sea-level rise. However, we are still confronted with the problem of relying too much on point estimates of the model parameters. While our previous optimization minimized the least-squares, there could be many different values which have a similar level of error.

In general, we should be suspicious (and at the least, highly skeptical) if a given model fits data *too* perfectly. This is because there are all sorts of sources of "error", including observation/data-reconstruction errors and model simplifications. As the famous statistician George Box wrote, "all models are wrong, but some are useful" (Box, 1979): the critical question is whether a model gives us insights into key dynamics.

Our goal, then, is to quantify uncertainties relevant to future projections of sea levels. This will allow us to obtain a probabilistic representation of the contribution of SLR to future flood risk. Notice that our model,

$$\Delta H(t) = \alpha \left(T(t) - T_{\text{eq}} \right),$$

is deterministic. Figure 2.2 shows us that the best-fit model (using RMSE as the metric for deviation), did not perfectly fit the data, which is typically the case.

The difference between the data y_t and the model output $\widehat{y}_t$ at each time t is called the *residual* or the *discrepancy*, which we can denote as ω_t. That is,

$$\omega_t = y_t - \widehat{y}_t.$$

As the contributions to the discrepancy ω_t are uncertain, we can construct a statistical model for it. As the data can be viewed as being fixed for any given calibration exercise, This is an important choice, as this statistical model has an effect on the associated model parameter distributions (Brynjarsdóttir & O'Hagan, 2014).

References

Box, G. E. P. (1979). Robustness in the strategy of scientific model building. In *Robustness in Statistics* (pp. 201–236). Elsevier. https://doi.org/10.1016/B978-0-12-438150-6.50018-2

Brynjarsdóttir, J., & O'Hagan, A. (2014). Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, *30*(11), 114007. https://doi.org/10.1088/0266-5611/30/11/114007

Hansen, J., Ruedy, R., Sato, M., & Lo, K. (2010). Global Surface Temperature Change. *Reviews of Geophysics*, *48*(4). https://doi.org/10.1029/2010RG000345

Jevrejeva, S., Moore, J. C., Grinsted, A., Matthews, A. P., & Spada, G. (2014). Trends and acceleration in global and regional sea levels since 1807. *Global and Planetary Change*, *113*, 11–22. https://doi.org/10.1016/j.gloplacha.2013.12.004

Morice, C. P., Kennedy, J. J., Rayner, N. A., Winn, J. P., Hogan, E., Killick, R. E., et al. (2021). An Updated Assessment of Near-Surface Temperature Change From 1850: The HadCRUT5 Data Set. *Journal of Geophysical Research: Atmospheres*, *126*(3). https://doi.org/10.1029/2019JD032361

Nicholls, Robert J., Tol, R. S. J., & Vafeidis, A. T. (2008). Global estimates of the impact of a collapse of the west antarctic ice sheet: An application of FUND. *Clim. Change*, *91*(1), 171. https://doi.org/10.1007/s10584-008-9424-y

Nicholls, Robert J., Lincke, D., Hinkel, J., Brown, S., Vafeidis, A. T., Meyssignac, B., et al. (2021). A global analysis of subsidence, relative sea-level change and coastal flood exposure. *Nature Climate Change*, *11*(4), 338–342. https://doi.org/10.1038/s41558-021-00993-z

Rahmstorf, S. (2007). A Semi-Empirical Approach to Projecting Future Sea-Level Rise. *Science*, *315*(5810), 368–370. https://doi.org/djt4n9

Spanger-Siegfried, E., Fitzpatrick, M., & Dahl, K. (2014). *Encroaching tides - union of concerned scientists*. Cambridge, MA: Union of Concerned Scientists.

Walsh, K. J. E., McBride, J. L., Klotzbach, P. J., Balachandran, S., Camargo, S. J., Holland, G., et al. (2016). Tropical cyclones and climate change. *WIREs Climate Change*, *7*(1), 65–89. https://doi.org/10.1002/wcc.371