An Introduction to Extreme Value Statistics

Marielle Pinheiro and Richard Grotjahn

This tutorial is a basic introduction to extreme value analysis and the R package, `extRemes`. Extreme value analysis has application in a number of different disciplines ranging from finance to hydrology, but here the examples will be presented in the form of climate observations.

We will begin with a brief background on extreme value analysis, presenting the two main methods and then proceeding to show examples of each method. Readers interested in a more detailed explanation of the math should refer to texts such as Coles 2001 [1], which is cited frequently in the Gilleland and Katz `extRemes 2.0` paper [2] detailing the various tools provided in the `extRemes` package. Also, the `extRemes` documentation, which is useful for functions syntax, can be found at `http://cran.r-project.org/web/packages/extRemes/extRemes.pdf`

For guidance on the R syntax and R scripting, many resources are available online. New users might want to begin with the Cookbook for R (`http://www.cookbook-r.com/` or Quick-R (`http://www.statmethods.net/`)

Contents

Chapter 1

Background

1.1 Extreme Value Theory

In general terms, the chance that an event will occur can be described in the form of a probability. Think of a coin toss; in an ideal scenario, there is a 50% chance that the coin will land either heads or tails up in a single trial, and as multiple tosses are made, we gather additional information about the probability of landing on heads versus tails. With this knowledge, we can make predictions about the outcomes of future trials.

The coin toss scenario is an example of a simple binomial **probability distribution** (frequency of heads versus frequency of tails), but the fundamental concept can be expanded to encompass more complex scenarios, described by other probability distributions. Here, we are interested in formulating a mathematical representation of **extremes**, or events with a low probability of occurrence. The definition of extremes varies by field and methodology; in the context of climate, we will talk about extreme temperatures, such as the higher-than-average temperatures experienced over the course of a heat wave. An extreme weather event is an occurrence that deviates substantially from typical weather at a specific location and time of year. Specific definitions vary depending on the distribution of local weather patterns and method of categorization. Analysis of extreme weather is made more difficult by the fact that extreme events are, by definition, rare, and therefore reliable data is limited.

Extreme value theory deals with the stochasticity of natural variability by describing extreme events with respect to a probability of occurrence. The frequency of occurrence for events with varying magnitudes can be described as a series of identically distributed random variables

$$F = X_1, X_2, X_3, ...X_N \tag{1.1}$$

where F is some function that approximates the relationship between the magnitude of the event (variable X_N) and the probability of its occurrence.

While it is possible to do analysis with the overall distribution of temperature magnitudes, we are focusing on just the extreme temperatures, which can also be described in terms of a probability distribution function. We can use the information from the resultant distribution to analyze trends and the likelihood that catastrophic events will occur. Here are just a few of the possibilities:

- Predict how often catastrophic events are likely to occur (return level)
  - extreme temperatures (heat waves, cold air outbreaks)
  - precipitation levels, flooding and droughts
  - hurricane frequency and magnitude
- Perform simulations utilizing the distributions, and use the results to anticipate future concerns
  - How does the occurrence of current temperatures match the calculated probability of occurrence? In a changing climate, what can we expect to change in terms of temperature trends?
  - What do current precipitation levels mean for reservoir levels and overall water usage?
  - What changes can we expect for the intensity and frequency of hurricanes?

1.2 Generalized Extreme Value (GEV) versus Generalized Pareto (GP)

We will focus on two methods of extreme value analysis. The first approach, GEV, looks at distribution of block maxima (a block being defined as a set time period such as a year); depending on the shape parameter, a Gumbel, Fréchet, or Weibull[1] distribution will be produced. The second method, GP, looks at values that exceed a defined threshold[2]; depending on the shape parameter, an Exponential, Pareto, or Beta distribution will be produced.

The two methods are summarized below; a demonstration of each method follows in the next chapter.

	GEV	**GP**
Description	Distribution function of standardized maxima (or minima)— block maxima/minima approach	probability of exceeding pre-determined threshold— peaks over threshold approach
Parameters	**location** μ: position of the GEV mean	**threshold** u : Reference value for which GP excesses are calculated
	scale σ: multiplier that scales function **shape** ξ: Parameter that describes the relative distribution of the probabilities.	
General function (CDF)	for extreme value z, $G(z) = \exp\left[-\left\{1+\xi\left(\frac{z-\mu}{\sigma}\right)\right\}_+^{-1/\xi}\right]$	for threshold excess x, $H(x) = 1-\left[1+\xi\left(\frac{x-u}{\sigma_u}\right)\right]_+^{-1/\xi}$
Limit as $\xi \to 0$	Gumbel: $G(z) = \exp\left[-\exp\left\{-\left(\frac{z-\mu}{\sigma}\right)\right\}\right]$	Exponential: $H(x) = 1-\exp\left(-\frac{x-u}{\sigma}\right)$
$\xi > 0$	Fréchet	Pareto
$\xi < 0$	Weibull	Beta
Interpretation of results	Return level: value z_p that is expected to be exceeded on average once every $1/p$ periods, where $1-p$ is the probability associated with the quantile. Find z_p such that $G(z_p) = 1-p$	Return level: value x_m that is exceeded every m times. Begin by estimating ζ_u, the probability of exceeding the threshold. Then, x_m is $x_m = \begin{cases} u+\frac{\sigma_u}{\xi}[(m\zeta_u)^\xi - 1] & \xi \neq 0 \\ u+\sigma_u \ln(m\zeta_u) & \xi = 0 \end{cases}$

Table 1.1: Description of the two basic types of extreme value distributions

Probability density functions (PDFs) and cumulative distribution functions (CDFs)

The **probability density function** (as shown in Figure 1.1), plots the relative likelihood (on the y axis) that a variable will have value X (on the x axis). Contrast this with the **cumulative distribution function** (as shown in Figure 1.2), in which the probability of X is defined by integrating the PDF over the range in which the variable $\leq X$. Note that the equations in Table 1.1 are CDFs, not PDFs.

[1] Note that the Weibull distribution has a finite right endpoint; Gumbel and Fréchet have infinite right endpoints

[2] The GP function can be approximated as the tail of a GEV; the scale parameter σ_u is a function of the threshold and is equivalent to $\sigma_g + \xi(u-\mu)$, where σ_g, ξ and μ are all parameters of a corresponding GEV distribution

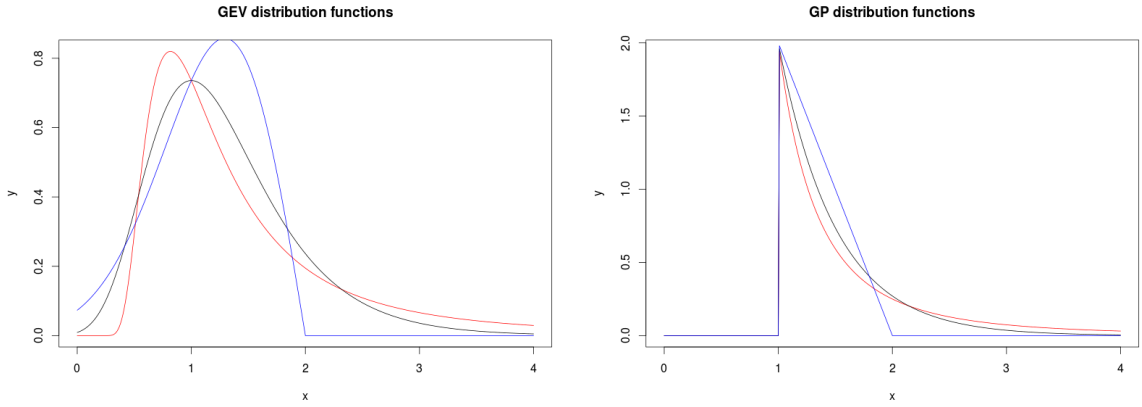


Figure 1.1: Probability density functions for (left) GEV and (right) GP. For each plot, x and μ=1, σ=0.5. For ξ, blue=-0.5, black=0, and red=0.5

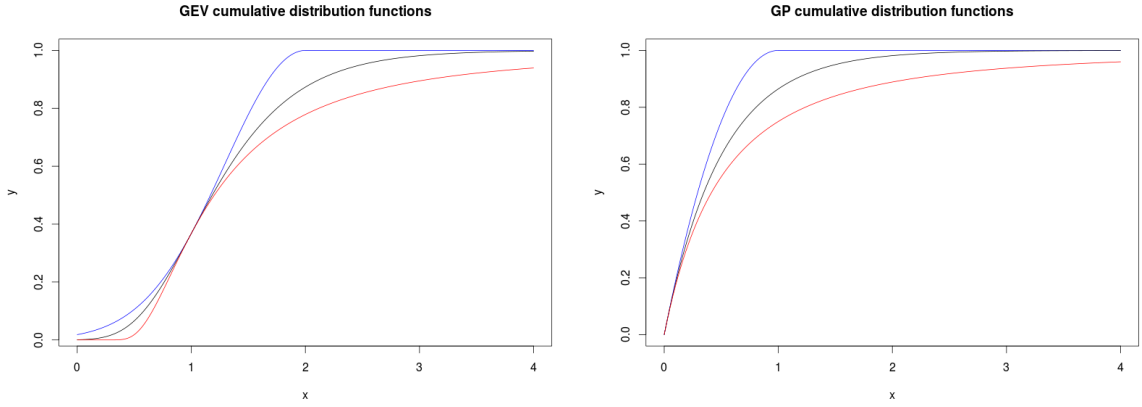


Figure 1.2: Cumulative density functions for (left) GEV and (right) GP. For each plot, x and μ=1, σ=0.5. For ξ, blue=-0.5, black=0, and red=0.5

1.3 Stationarity versus non-stationarity

In calculating the model fit, it is useful to determine whether the model distribution remains the same as time progresses. Figure 1.3 shows how the probability of the extremes might change in the future under possible proposed climate scenarios. Note that the model that was fit to an extreme temperature distribution from the 20th century might not work for 21st century values.

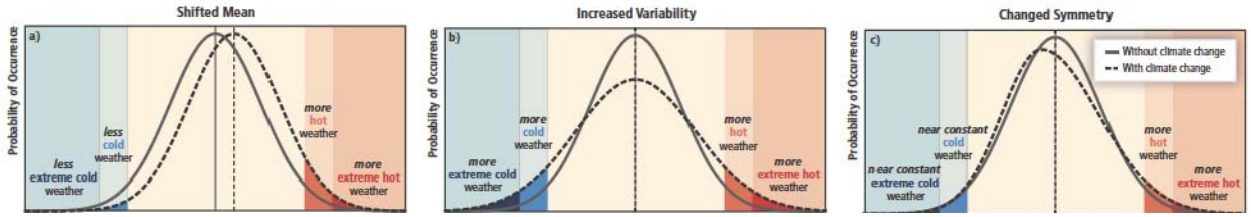


Figure 1.3: Temperature magnitudes and probability of occurrence in the context of a warmer climate. Plots obtained from IPCC Summary for Policymakers (2012), Figure SPM.3

This change can be incorporated into the model as a change with respect to time; μ, x, σ, and ξ can be represented as some sort of function of time. This is known as a **non-stationary model**, in contrast to a **stationary model** in which the model parameters are fixed constants.

For example, if we anticipated that there would be a greater proportion of extreme high temperatures in the future, the shape of a GEV function fitted to the data might increase towards a more heavy-tailed distribution, as seen in Figure 1.4.

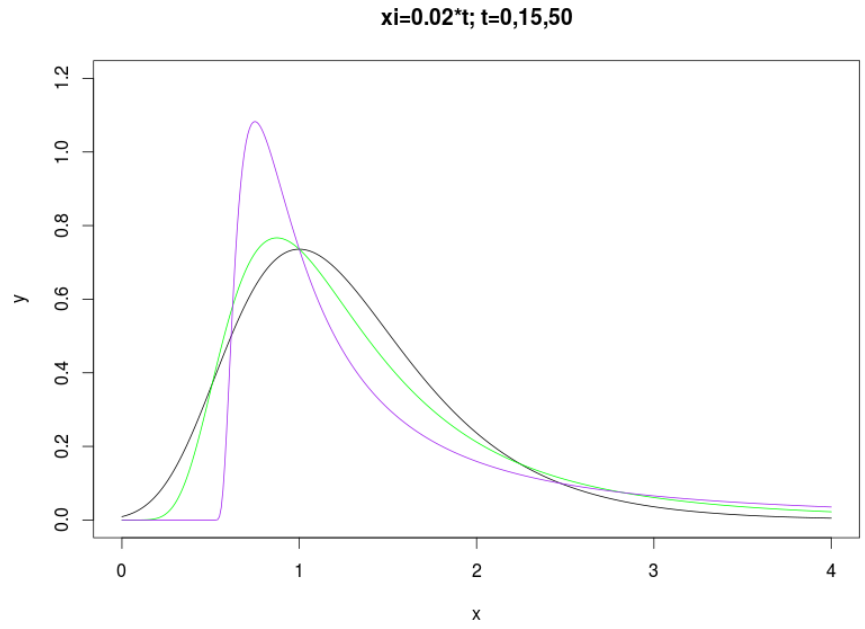


Figure 1.4: Altering ξ in a GEV distribution as a function of time: $\xi(t) = 0.02t$. Black: $t = 0$ Green: $t = 15$ Purple: $t = 50$. Other parameters remain constant.

Non-stationarity is most often seen with GP distributions where the threshold is defined as a function. If we wanted to increase the GP threshold linearly, as might be the case in an increasingly warmer climate, we could define the threshold as

$$x(t) = x_0 + x_1 t \tag{1.2}$$

where x_0 would be the initial threshold value and x_1 would increment the threshold value over time. The same can be done for the other parameters as well; for example,

The parameters could also be tailored to incorporate some sort of seasonal cycle; Figure 1.5 demonstrates that there is a definite trend to the temperature values throughout the JJAS months, so we tailor the threshold accordingly by writing the threshold as a sine function (see Appendix B.3.1 for equation). Note the difference in the number of points that exceed the non-stationary threshold (red) as opposed to the stationary threshold (blue). We gain some points in June and September, and lose some points in July and August.

Another method, which will be used in this study (see section 2.3 for a demonstration), utilizes the value of the 90th percentile for each day in the season to provide an initial estimate for a threshold value, and then fits a sum of sines and cosines to those estimates; this is known as a sum of harmonics. The 90th percentile values are plotted on Figure1.6 as the green line, with the corresponding harmonics sum plotted in black. At many spots throughout the season, the black line is reasonably close to the sine function, but there are notable dips in mid-August and mid-September.

Figure 1.7 demonstrates how the distribution of the excesses changes with each of the methods. Altering the shape of the threshold also changes the magnitude of the excesses, and this leads to a different GP distribution, highlighting the importance of choosing a proper threshold.

Limitations of non-stationary threshold in extRemes

Certain methods in the `extRemes` package will be unavailable when using a non-stationary threshold; for example, `threshrange.plot`, which is used to determine an optimal threshold value, will not work with a non-constant threshold. If you are going to use a non-stationary threshold, it is suggested that you begin by making an initial constant threshold estimate using `threshrange.plot`. Then, examine the data for any possible time-dependent trends and determine a threshold function that is near to the initial constant threshold. Finally, calculate a new dataset with threshold excesses as demonstrated in Section 2.3

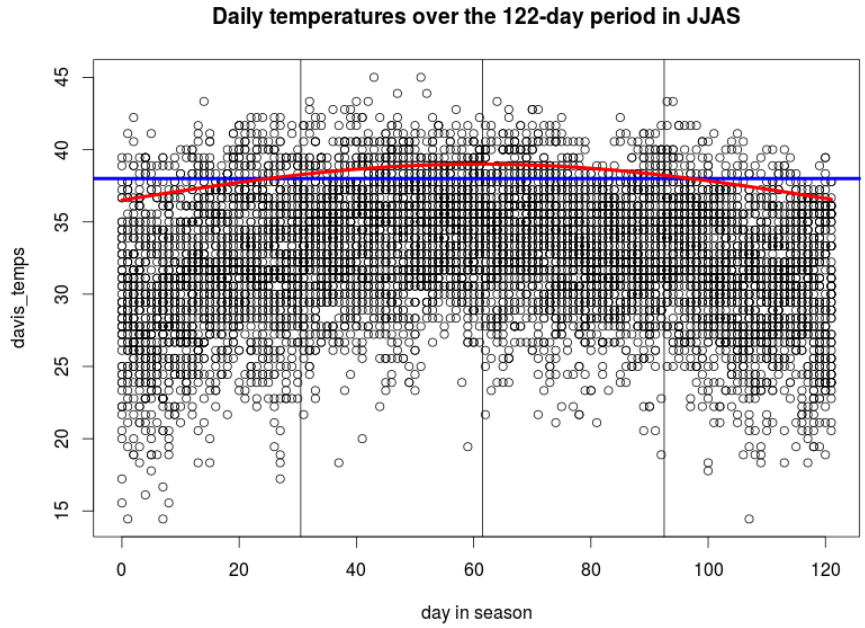


Figure 1.5: Daily maximum temperatures from 1951-2012, plotted with respect to day in season. Blue line represents a constant threshold of 37, while red line represents a threshold with equation B.2, which is a sine function intended to follow the seasonal trend. Vertical lines denote division of months.

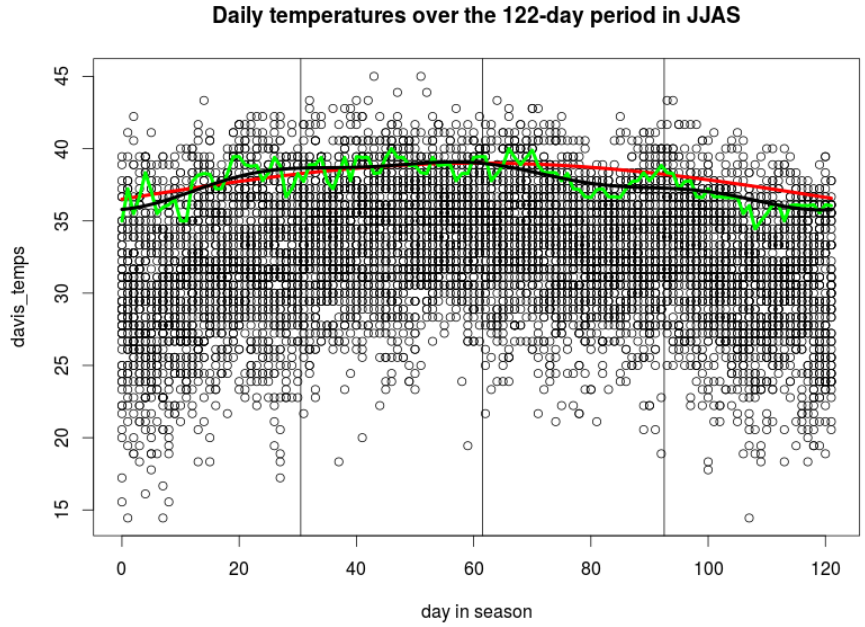


Figure 1.6: Modification of Figure 1.5 to show newly calculated threshold based on fitting harmonic equation to 90th percentile values. Sine function in red, 90th percentile values in green, fitted harmonics equation in black

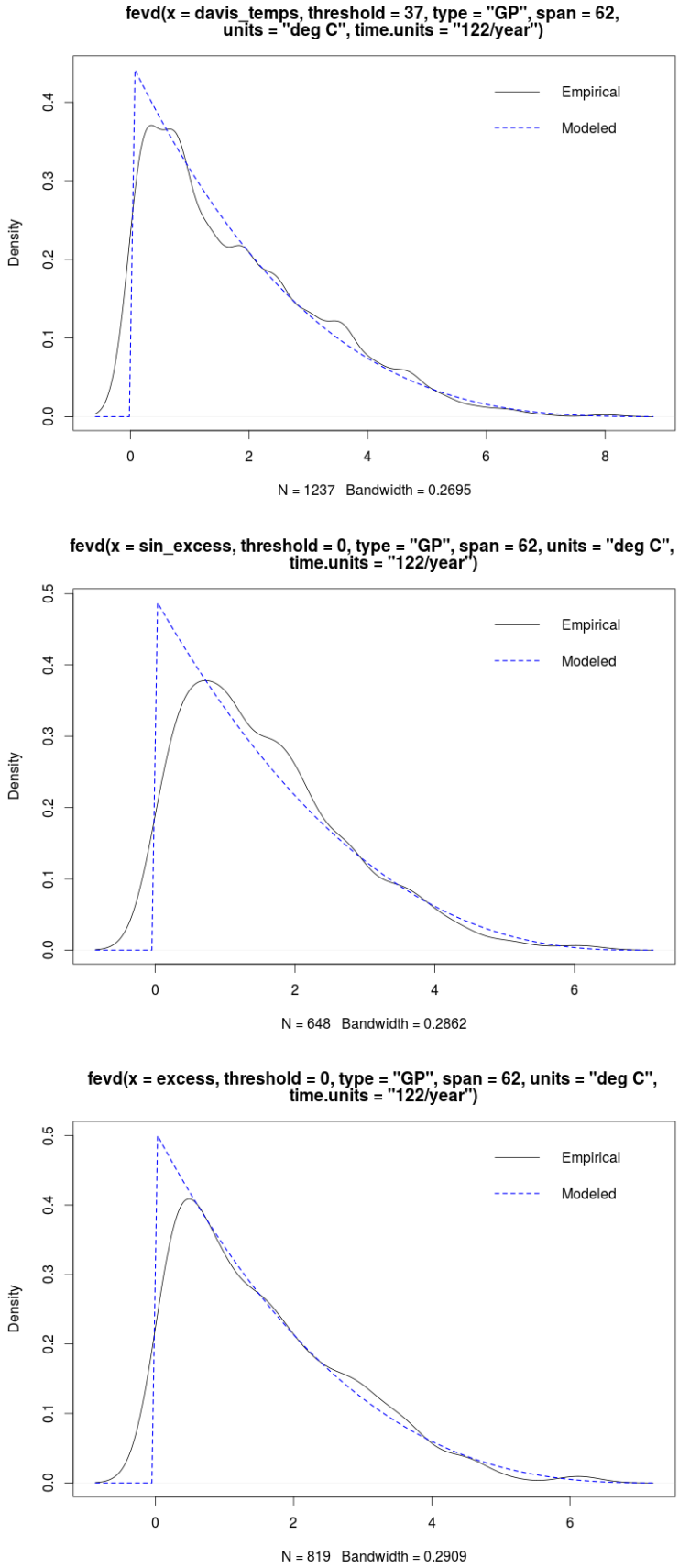


Figure 1.7: A comparison of the GP PDFs for (top) a constant threshold of 37, (middle) a sine-varying threshold, and (bottom) a harmonics-fitted threshold. Black solid lines represent the probabilities of the excesses as related to the various thresholds, and blue dashed lines represent the PDF that was calculated by `extRemes` based on the excesses.

Chapter 2

ExtRemes example: Using Davis station data from 1951-2012

Now we will turn to the application of each distribution function and interpret the results. This chapter will focus on two main questions:

1. How well does each method capture the data distribution?
2. How do the results from each method compare to one another?

Before you begin:

See Appendix C.1 for instructions on how to install and run R, if you haven't already done so. The `extRemes` library must also be installed.
Appendix C.3 contains some useful tips for subsetting and processing datasets in preparation for use with `extRemes`.

2.1 Explanation of the `fevd` input and output

The `fevd` function is the primary function in the `extRemes` package; it calculates the parameters for the specified probability distribution that best fits the data, and all other calculations are based off of this fit. See Appendix A for definitions of the various statistical parameters in the `fevd` output.

The syntax for `fevd` in this example is

`fevd(data, type, units, span, time.units, threshold...)`[1]

Usage:

1. `data`: Dataset. Make sure that your data is appropriate for the calculation method– block maxima requires a single maximum (or minimum) value per block of time (e.g. per year), as opposed to peaks over threshold, which requires all of the daily maxima per length of time being analyzed (in this case, the summer seasons for the 51-year time span).
2. `type`: specify `"GEV"` or `"GP"`
3. `units`: Units of dataset (here, `"deg C"`). Optional.
4. `span`: defines the number of years in the model (necessary for GP model)
5. `time.units`: Only needed if `span` is undefined; determines the number of years in the data. Here, we are looking at the summer months; the number of days is 122 (30 for June, 31 for July, 31 for August, 30 for September) so `time.units="122/year"`
6. `threshold`: Only necessary for GP method. This is the value x from which excesses are calculated. Appendix B explains how to determine an appropriate threshold.

[1] there are additional options beyond the ones specified here, but for the purposes of this example we are sticking to the simplest models; check the `extRemes` documentation for more extensive functionality

2.1.1 `fevd` tools used in this example

- `plot.fevd`

  Syntax: `plot(fit,type...)`

  Usage: The default `plot(fit)` with only the `fit` variable (your `fevd` variable) as an argument will return a 4-plot figure with:

  1. Top left (`type="qq"`): Quantile-quantile plots, with model quantiles on the x axis and empirical quantiles on the y axis and a black line representing the 1-1 line.
  2. Top right (`type="qq2"`): Quantile-quantile plots, with empirical quantiles on the x axis, model quantiles on the y axis, and confidence intervals as dashed grey lines. The 1-1 line is drawn as an orange dashed line and the linear fit of the quantile-quantile plot is drawn as a solid grey line.
  3. Bottom left (`type="density"`): Model (dashed blue line) and observational data (solid black line) PDFs
  4. Bottom right (`type="rl"`): Plot of the return period in years (x axis) for various temperature magnitudes (y axis)

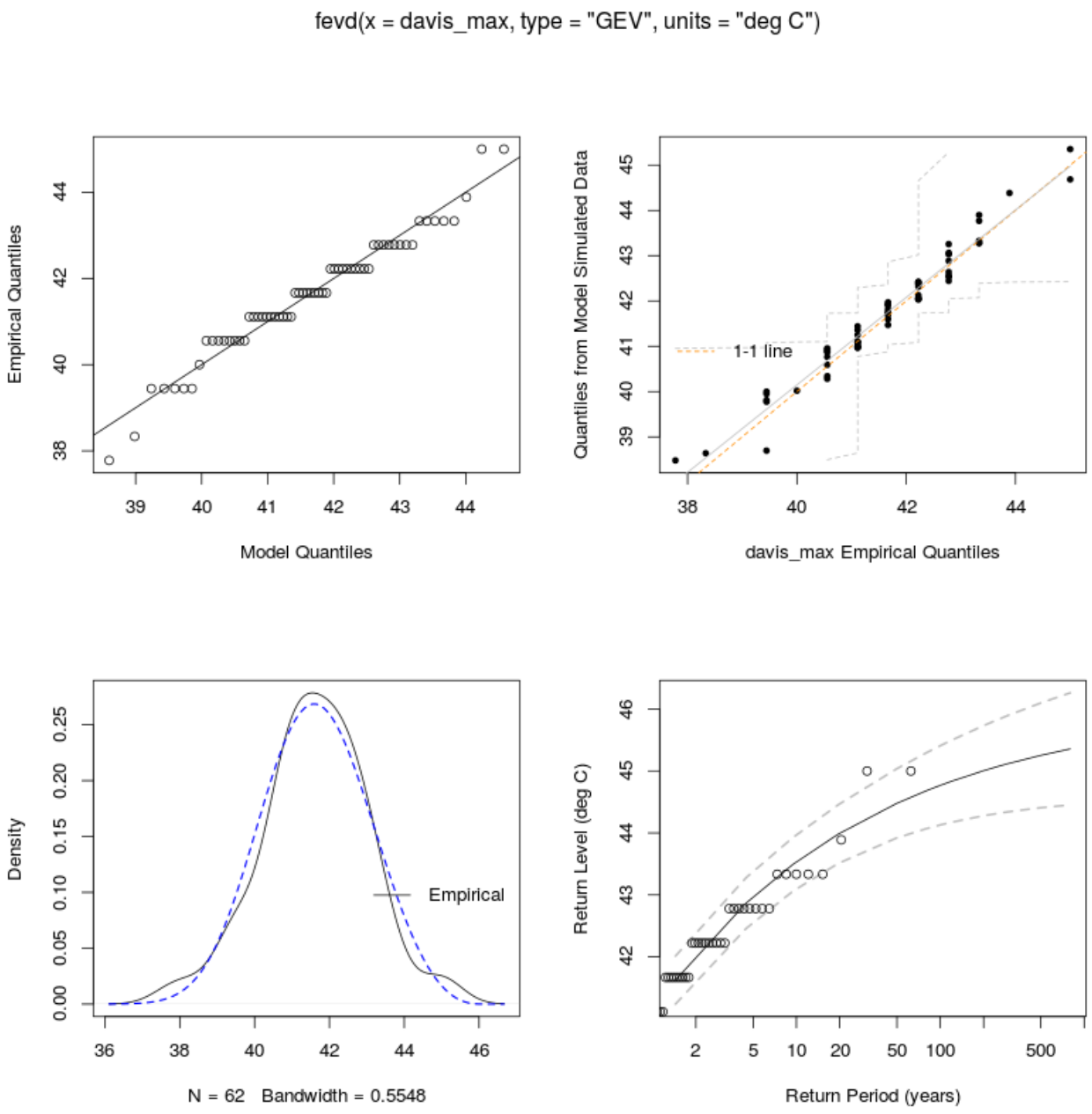


Figure 2.1: Default output for `plot.fevd` (shown for GEV model). This is `type="primary"`.

If you wish to output only a single plot, or a plot that's not one of the defaults, specify the `type` (`"probprob"`, `"qq"`, `"qq2"`, `"Zplot"`, `"hist"`, `"density"`, `"rl"`, `"trace"`)

- `pextRemes.fevd`: the probability of `fit`$\leq$`q`, where `q` is a vector of specified values `q=c(num1,num2...)`

  Syntax: `pextRemes(fit, q, lower.tail=FALSE,...)`

- `rextRemes.fevd`: Create simulated data sets based on the calculated probability distribution, where `n` is the number of random draws.

  Syntax: `rextRemes(fit, n,...)`

- `ci.fevd`: the confidence interval associated with either the fit parameters (`ci(fit,type="parameter")`) or estimated `n`-year return period temperature value (`ci(fit,type="return.level",return.period=n)`).

  Syntax: `ci(fit,type,return.period,...)`

2.2 Working with the data: Generalized Extreme Value (GEV) distribution fit

Our dataset, shown in Figure 2.2, is the 1951-2012 maximum recorded temperature (in degrees Celsius) for the months June, July, August, and September per year from an observation station based in Davis, California. There are 62 data points, one for each 122-day period per year.

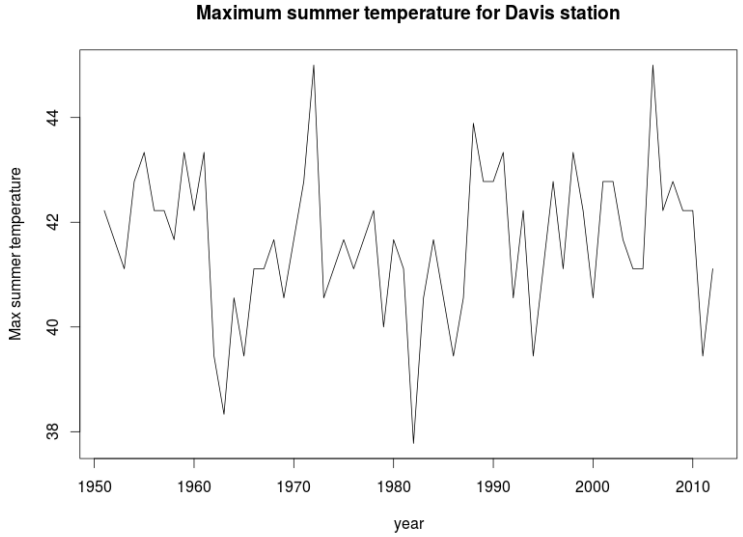


Figure 2.2: Seasonal maximum temperature for JJAS per year at Davis station, 1951-2012

Let us begin with some initial observations. Using `summary(davis_max)`, we get

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  37.78   40.69   41.67   41.61   42.64   45.00
```

There is an average maximum value of 41.61 degrees Celsius with the greatest maximum being 45 degrees Celsius for the 62-year period. Figure 2.3, generated using `plot(fit1,type="density")` and `plot(fit1,type="qq2")`, shows the PDF generated by the model against the actual distribution of block maxima.

We can examine the confidence interval of the parameters as follows:

```
ci(fit1,type="parameter")
fevd(x = davis_max, type = "GEV", units = "deg C")

[1] "Normal Approx."

         95% lower CI   Estimate 95% upper CI
location   40.7374923 41.1242733   41.5110542
scale       1.1735853  1.4348918    1.6961983
shape      -0.4124608 -0.2902778   -0.1680947
```

How well does the GEV method capture the data distribution?

Based just on observing the calculated PDF versus empirical data PDF, the model seems to have a reasonably good fit for most of the data, and the confidence intervals for the parameters reflect this.

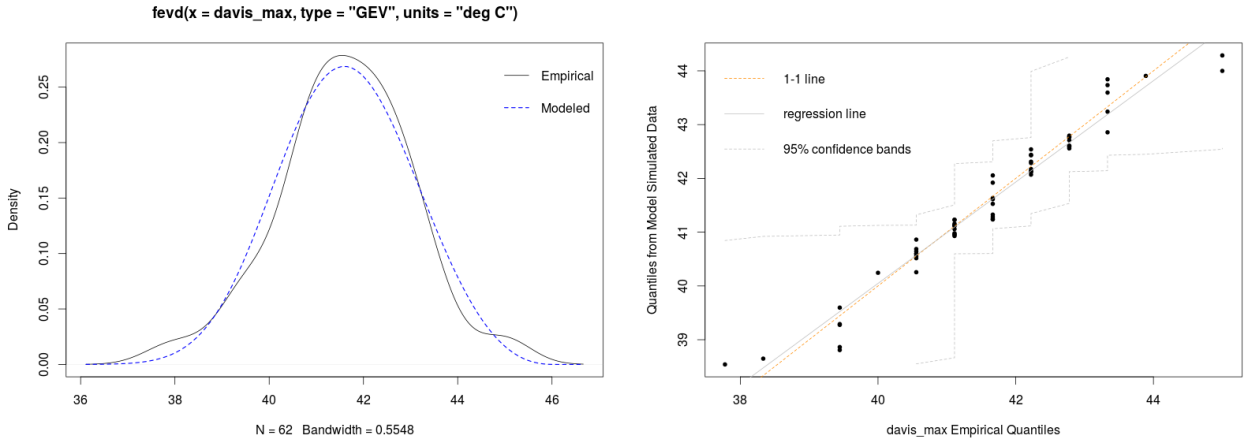


Figure 2.3: **Left:** Model distribution (blue dashed line) vs actual temperature probability distribution (black solid line) **Right:** Quantile-quantile plot with empirical quantiles on the x-axis and model quantiles on the y-axis

2.3 Working with the data: Generalized Pareto (GP) distribution fit

In the GP example, we will use the data from the same observation station, but the dataset is extended to include all recorded daily maximum temperatures for the summer months from 1951-2012, for a total of 7564 data points. Using `summary(davis_temps)`, we get

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  14.44   29.44   32.78   32.52   35.56   45.00
```

We calculate a new data set, `excess`, in which the time-varying threshold is subtracted from the temperature values. Both the original temperature values and the calculated excesses can be seen in Figure 2.4. We noted in Section 1.3 that while the sine-varying threshold is pretty good at capturing the seasonal trend, the harmonics method is best at matching seasonal fuctuations, and this is the one that will be used. To see how the sine and harmonics functions were calculated, refer to Appendix B.3[2].

We utilize a threshold of 0 when calculating the model fit. Some statistics for the distribution of temperature excesses:

```
summary(excess[excess>0])
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.005552 0.505700 1.275000 1.562000 2.317000 6.329000
```

Figure 2.5 shows a comparison of the empirical threshold excesses with the model prediction.

The confidence interval for scale and shape are shown below (threshold is excluded because it was explicitly provided in the function input).

```
ci(fit2,type="parameter")
fevd(x = excess, threshold = 0, type = "GP", span = 62, units = "deg C",
    time.units = "122/year")

[1] "Normal Approx."

      95% lower CI   Estimate 95% upper CI
scale    1.8149661  1.9770137    2.1390613
shape   -0.3180036 -0.2693377   -0.2206717
```

How well does the GP method capture the data distribution?

The selection of the harmonics-fitted threshold results in a data distribution that closely fits the calculated GP model, with the exception of the highest extremes at the tail (as with the GEV model). Both the scale and shape parameters have small standard errors.

[2]Appendix B also discusses declustering, but for simplicity's sake, we will omit declustering in this example.

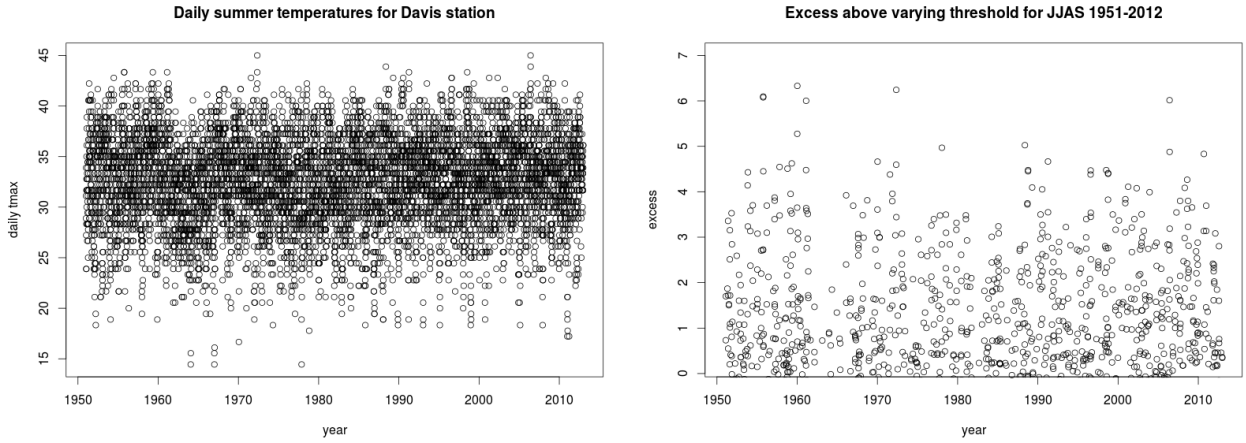


Figure 2.4: **Left:** Daily maximum temperatures for JJAS per year at Davis station, 1951-2012. **Right:** Daily temperature excesses for JJAS per year at Davis station, 1951-2012.

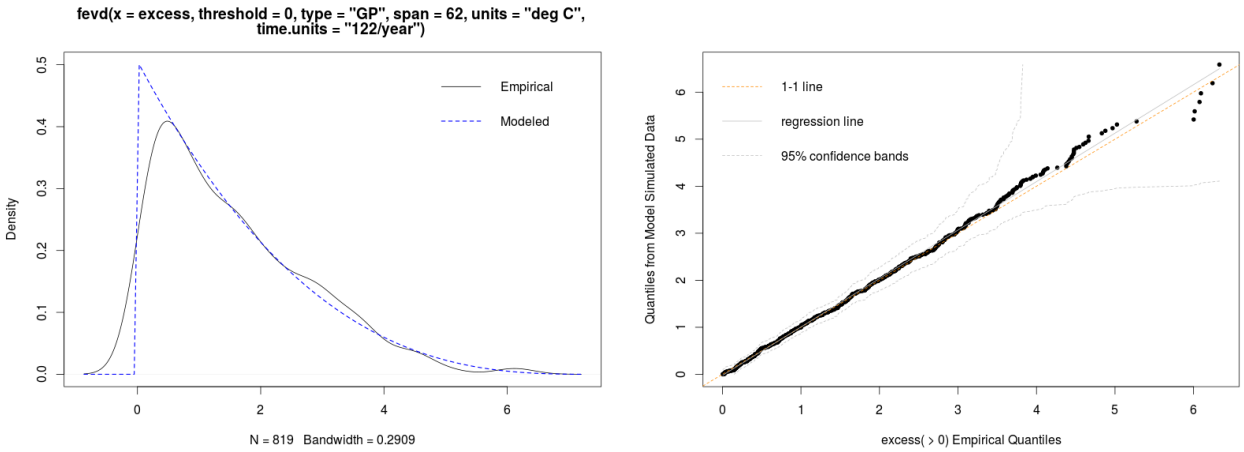


Figure 2.5: **Left:** Model distribution (blue dashed line) vs actual temperature excesses (black solid line). **Right:** Quantile-quantile plot with empirical temperature excesses on the x-axis and model quantiles on the y-axis

2.4 Using the model fit: probabilities and return periods

Based on the model fits, what is the predicted return period for various return levels? Here, we compare GEV and GP outputs.

2.4.1 The relationship between probability and return period

For the GEV method, there is an inverse relationship between probability and return period.[3]Therefore, the return period for temperature magnitude z is simply defined as

$$rp(z, p) = \frac{1}{p} \tag{2.1}$$

For the GP function, the return period calculation is complicated by the fact that there is a varying threshold. We cannot simply invert the probability for excess x, since that probability changes (as opposed to the GEV method, where there is a single values per year).

Since the extRemes package does not have a function for calculating return period from specified return level, we must estimate from a few `return.level` outputs. The method is:

1. Create a vector of return levels for a specified range of test return periods
2. find the return level with a corresponding return period that most closely matches the desired return period

This is illustrated below for an example in which we are trying to find the return period for a temperature excess of 5 degrees Celsius.

[3]Note the usage of `lower.tail=FALSE`. Recall from Table 1.1 that when calculating return period, we use the equation $G(z_p) = 1 - p$; therefore, if we omit `lower.tail=FALSE`, we will get the values for $1 - p$, rather than p.

```
#Generate a sequence of test return periods
y<-seq(4,7,0.000001)

#calculate the return levels that correspond to the return periods
rl<-return.level(fit2,y)

#Find the intersection of the desired value and the generated sequence of return levels
rl[rl>4.9999999 & rl<5.0000001]

#This returns:
5.276114 5.276115
       5        5
```

So a return level of 5 degrees excess has a return period of approximatedly 5.28 years.

2.4.2 Test case: 2013 and 2014 records

From records kept at the UC Davis climate station (temperatures converted from Fahrenheit to Celsius for consistency's sake), accessed at `http://atm.ucdavis.edu/weather/uc-davis-weather-climate-station/`, we can calculate both seasonal maxima and daily temperature excesses, and compare the predicted probabilities to the empirical results.

For GEV calculations, we are concerned with the seasonal maxima, which, in this context, are the maximum summer temperatures for JJAS in 2013 and 2014:

	June	July	August	September
2013	45.6	42.8	40.6	40
2014	41.7	42.2	41.1	40.6

Table 2.1: JJAS observed maximum temperatures for Davis in 2013 and 2014

Therefore, for 2013, the seasonal maximum would be 45.6 and for 2014, the seasonal maximum would be 42.2.

For GP calculations, we will compare the predicted probability of exceeding the threshold with the full data sets for JJAS of 2013 and 2014, also obtained from the UC Davis climate station site (shown in Figure 2.6). We can use the same threshold excess calculation function used to generate the model fit dataset to calculate the excesses for 2013 and 2014, as shown in Figure 2.7.

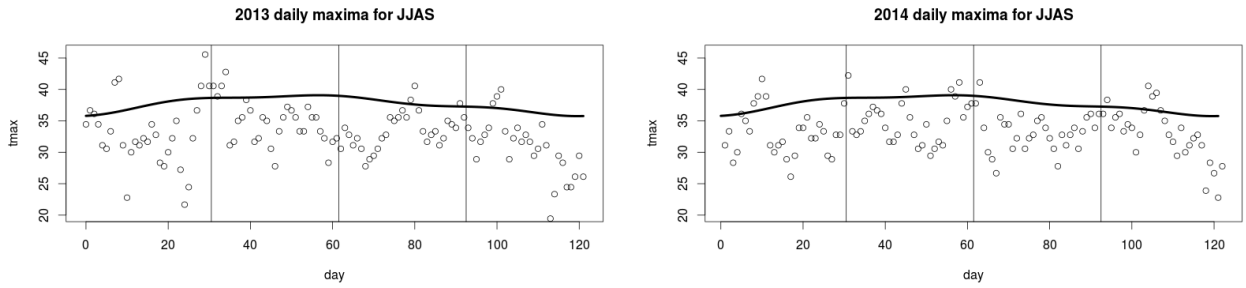


Figure 2.6: Daily temperature maxima for 2013 and 2014 with non-stationary threshold

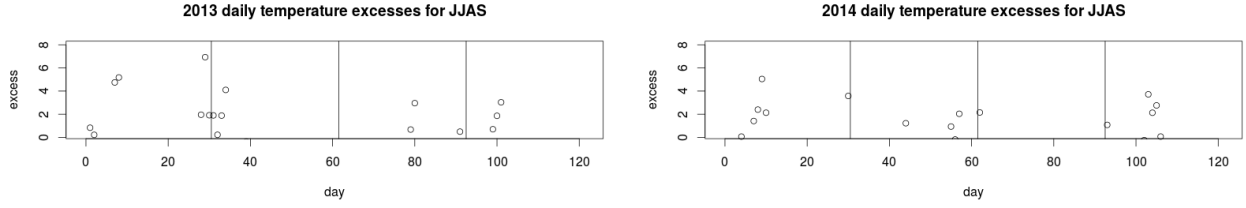


Figure 2.7: Temperature excesses for 2013 and 2014

Here are the statistics for 2013 and 2014 threshold excesses:

```
2013:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.2180  0.7003  1.8970  2.3280  3.0290  6.9300

2014:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.06429 1.14800 2.12600 2.04700 2.58000 5.04400
```

2.4.3 Comparing model probabilities and return periods

In order to directly compare the outcomes of the two methods, we will focus on the 2013 and 2014 seasonal maxima and their probabilities, and then look at the GP-calculated probabilities in terms of the excess above the threshold for these same data points. The results are shown in Table 2.2.

2.4.4 Comparing empirical probabilities and return periods

For the dataset which corresponds to the GEV model, the empirical probability is merely the fraction of the data which meets or exceeds the specified return level, and the return period is the inverse of the probability:

$$rp(Z) = \left(\frac{\text{npoints} \geq Z}{\text{nyears}}\right)^{-1}$$

The method for calculating the empirical return periods for the data which corresponds to the GP model is very similar to the GEV method, with the added parameter of number of days:

$$rp(X) = \left(\frac{\text{npoints} \geq X}{\text{nyears} \times \text{ndays}}\right)^{-1} \frac{1}{\text{ndays}}$$

2.4.5 Results: comparing empirical versus model return periods

Year	**GEV**	Model RP	Emp. RP	**GP**	Model RP	Emp. RP
2013	**45.6**	3378.15	64	**6.92**	3097.92	64
2014	**42.2**	2.86	2.29	**3.58**	> 1	0.98

Table 2.2: Maxima for 2013 and 2014 and the corresponding GEV probabilities and return periods (in years); and the same data points as threshold excesses, with the corresponding GP probabilities and return periods.

To summarize:

- 2013: For a temperature magnitude of 45.6 degrees Celsius (6.92 degrees excess above threshold), the GEV function predicts that the return period is 3378 years, while the GP function predicts that the return period is 3098 years. In both cases, the empirical return period for each dataset was 64 years.
- 2014: For a temperature magnitude of 42.2 degrees Celsius (3.58 degrees excess above threshold), the GEV function predicts that the return period is 3 years, while the GP function predicts that the return period is less than a year. The corresponding empirical return periods are fairly close to the model predictions.

Figure 2.8 shows the return level plots for the GEV and GP functions, respectively, with the 2013 and 2014 data overlaid as colored points.

2.5 Discussion: GEV vs. GP

This chapter has focused on two main questions, outlined at the beginning of the chapter.

2.5.1 How well does each method capture the data distribution?

In both the GEV and GP calculations, the empirical and model data showed good agreement with low- to mid-level extremes (in the GEV case, temperature magnitudes below 44 degrees Celsius; in the GP case, excesses less than 6 degrees Celsius). This makes sense considering that with the truly extreme extremes, they have disproportionate representation in the data. The empirical return period of 64 years for the 2013 maximum

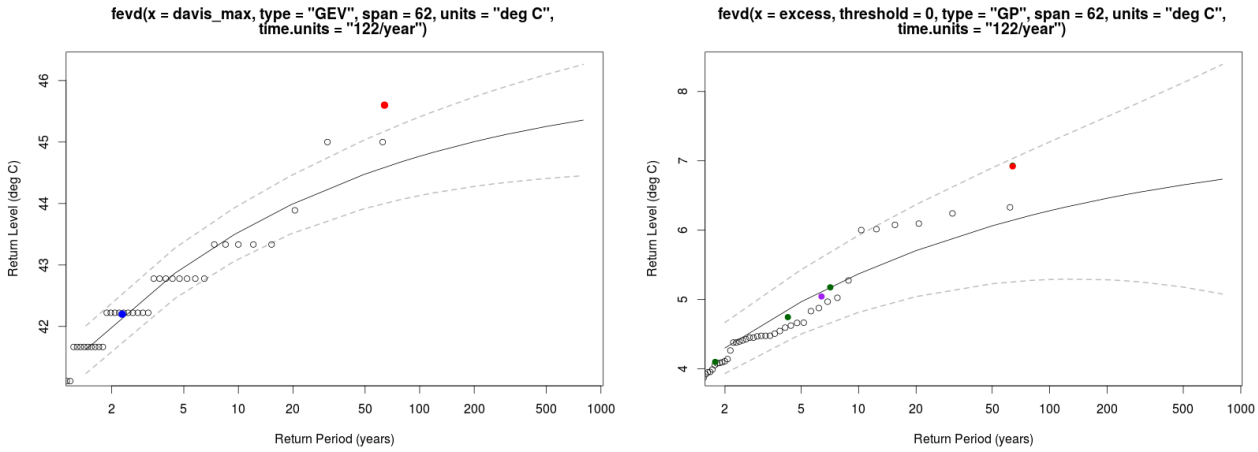


Figure 2.8: **Left:** GEV return periods and return levels. The red dot signifies the 2013 empirical data, while the blue dot signifies the 2014 empirical data. **Right:** GP return periods and return levels. The red dot signifies the point that corresponds to the 2013 maximum; the 2014 maximum falls outside of the plot window range. Other points from 2013 are plotted in green; other points from 2014 are plotted in purple.

temperature magnitude in this instance is the inverse of $\frac{1}{64}$, but if we obtained a larger dataset in which 45.6° Celsius was still the lone extreme outlier, the return period could be much larger, like 100 years ($\frac{1}{100}$) or more.

When it came to these extreme outliers, however, it's interesting to note that the GEV model seems to do worse than the GP model for the estimating 2013 maximum value's return period (see Figure 2.8 and the red point); although both methods estimate a return period of approximately 3000 years, the point falls outside of the GEV model's 95% confidence interval.

In both cases, the shape parameter was negative, meaning that the extremes with lesser magnitudes have a higher probability of occurrence than would be seen in a distribution where $\xi = 0$. However, the majority of estimation errors occurred at the extreme right end of the density function due to a higher-than-anticipated frequency of the largest extremes, as was clearly seen in the previous section.

2.5.2 How do the results from each method compare to one another?

Below are the `fevd` summaries for the GEV and GP methods, respectively.

GEV:

```
fevd(x = davis_max, type = "GEV",
units = "deg C")

[1] "Estimation Method used: MLE"

 Negative Log-Likelihood Value:  109.1557

 Estimated parameters:
  location      scale      shape
41.1242733  1.4348918 -0.2902778

 Standard Error Estimates:
  location      scale      shape
0.19734084 0.13332207 0.06233945

 Estimated parameter covariance matrix.
           location      scale      shape
location  0.0389434 -0.0013806 -0.0042352
scale    -0.0013806  0.0177748 -0.0045630
shape    -0.0042352 -0.0045630  0.0038862

 AIC = 224.3113

 BIC = 230.6927
```

GP:

```
fevd(x = excess, threshold = 0, type = "GP",
span = 62, units = "deg C",
time.units = "122/year")
[1] "Estimation Method used: MLE"

 Negative Log-Likelihood Value:  1156.561

 Estimated parameters:
    scale      shape
1.9770137 -0.2693377

 Standard Error Estimates:
    scale      shape
0.08267886 0.02483003

 Estimated parameter covariance matrix.
              scale         shape
scale  0.006835794 -0.0016936922
shape -0.001693692  0.0006165303

 AIC = 2317.122

 BIC = 2326.538
```

Each of the methods has its advantages and issues. The GP distribution has a larger initial dataset, since

it uses all of the summer daily maxima per year, while the GEV distributiononly uses the maximum seasonal temperature per year (819 points for the GP method as opposed to 62 points for the GEV method). However, the GP method requires much more guesswork, from the threshold determination and calculation to the extra steps involved in estimating return periods for corresponding return levels.

Appendix A

Explaining `fevd` output

The `fevd` output will produce a number of statistics, which are defined below.

Maximum Likelihood Estimation (MLE)

In these examples, the distribution functions are calculated utilizing a method known as **maximum likelihood estimation** to estimate the model parameters which will most closely fit the given data. Recalling that each data point has a probability associated with it, we will use the specified probability distribution function (either GEV or GP in these examples) as an initial hypothesis as to the shape of the model. The `fevd` function in `extRemes` tests combinations of parameters and comes up with the set of parameters for the probability density function that match the data and associated probabilities.
Although MLE is the default, the `extRemes` package offers a number of different methods for estimating the parameters, such as Generalized Maximum Likelihood estimation (GMLE), Bayesian, and L-moments. Each of these methods has different assumptions and optimizations.
It is important to note that a model is only an approximation of the real world; its accuracy is dependent on the quality of the data and the validity of the assumptions made. Therefore, it is important to assess the data itself before running any tests.

A.1 `fevd` output

- Negative log-likelihood:

  Likelihoods are conditional probability densities. For the probability distribution function $f(x, a)$, where x is the variable and a is the parameter, the likelihood is defined as

  $$L(a|x_1, ...x_n) = \prod_{i=1}^{n} f(x_i|a) \tag{A.1}$$

  Given x, MLEs attempt to maximize the likelihood $L(a)$ over all possible values of a.

  We use log likelihood because it transforms products into sums (an important feature when you have small likelihoods), and the natural log function is a monotone transformation. The equation above becomes

  $$\ln L(a|x_1, ..., x_n) = \sum_{i=1}^{n} \ln f(x_i|a) \tag{A.2}$$

  Values that are closer to 0 will indicate a better model fit.

- Estimated parameters: Estimated values of location/threshold, scale and shape for distribution based on the fit to the data

- Standard Error Estimates: Estimated standard deviations for each of the parameters

- Estimated parameter covariance matrix: Variance of location, scale, and parameter on the diagonal; covariance between the variables in the other parts of the matrix. If the covariance values were large, this would indicate that there was dependence between the model parameters.

- AIC (Akaike information criterion): parameter that measures relative quality of a statistical model, taking into account the tradeoff between the complexity of the model and the goodness of fit (rewards goodness of fit and penalizes increased number of parameters).

$$AIC = 2k - 2\ln(L)$$

where k is the number of parameters, and L is the maximized value of the likelihood function.

When comparing models, a model with a smaller AIC value is considered to be a "better" model.

- BIC (Bayesian information criterion): Similar to AIC:

$$BIC = (\ln(n) - \ln(2\pi))k - 2\ln(L)$$

where n is the number of data points, k is the number of parameters, and L is the maximized value of the likelihood function.

A.2 Other `fevd` options

The text seen in Chapter 2 is a summary of the `fevd` calculation; however, typing `names(fevd)`

Appendix B

GP fitting

B.1 Threshold selection

Before using the `fevd` function to calculate a GP distribution, you must define the threshold. However, deciding on the proper threshold requires some subjectivity.

A good rule of thumb is to try and determine the initial point at which further increasing the threshold value does not significantly alter the value of the scale and shape parameters (although note that the threshold cannot be too high, as this will greatly reduce the amount of available data with which to fit the distribution). `extRemes` has the `threshrange.plot` tool in order to facilitate this selection.

For `threshrange.plot`, you select a range of possible values that make sense for a threshold temperature. It performs MLE calculations and outputs threshold-scale and threshold-shape plots, with vertical lines signifying the confidence intervals for the scale and shape parameters. Try to choose the highest threshold value possible where the confidence interval is not too large. The figures below illustrate different outputs obtained based on the defined ranges.

Attempt 1 (left): The threshold should be at least larger than the mean, so the lower bound was set at 34. The confidence interval bars begin to grow large around a threshold value of about 41, indicating that there is more uncertainty associated with these values.

Attempt 2 (right): changed minimum threshold to 36 and maximum threshold to 40, and increased the number of plotted points (`nint=30`). This causes an oscillatory pattern that makes it more difficult to determine a reasonable threshold value; however, note that the oscillations begin to grow in amplitude after $x \approx 38$, which could indicate more uncertainty in the parameter estimations. Therefore, we will use a threshold value of 38 as a test, although it is worthwhile to try a number of values in this range in order to find the best fit.

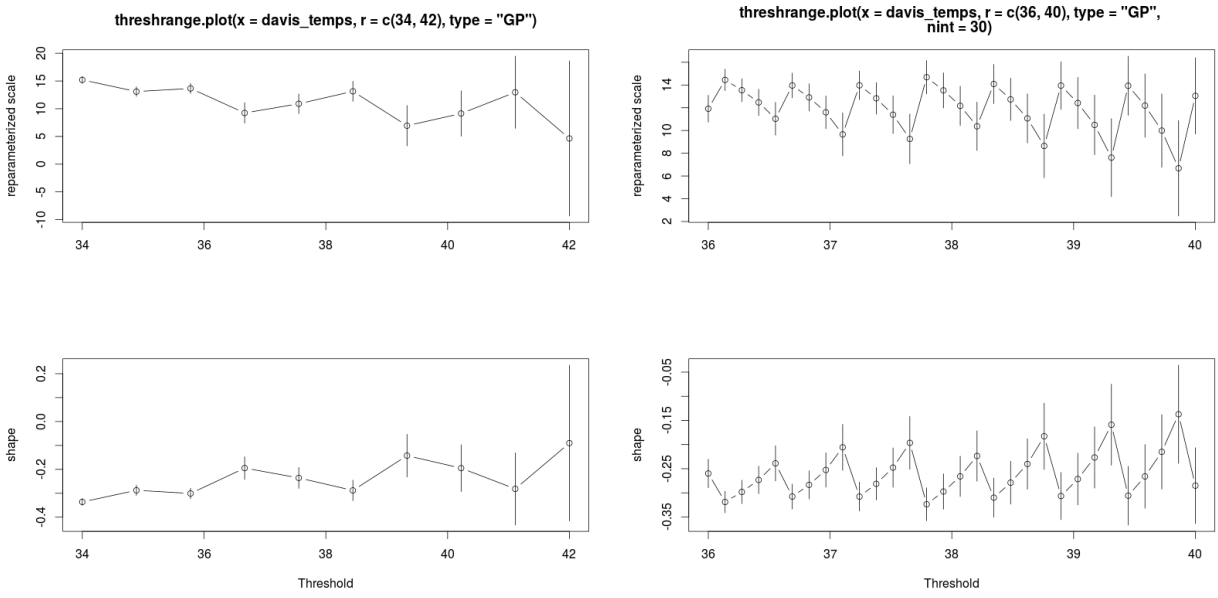
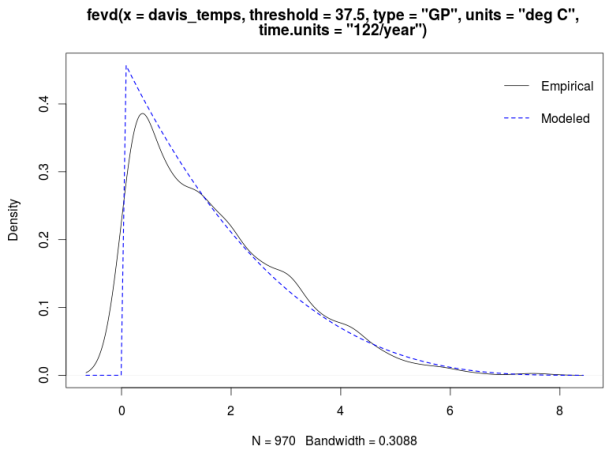


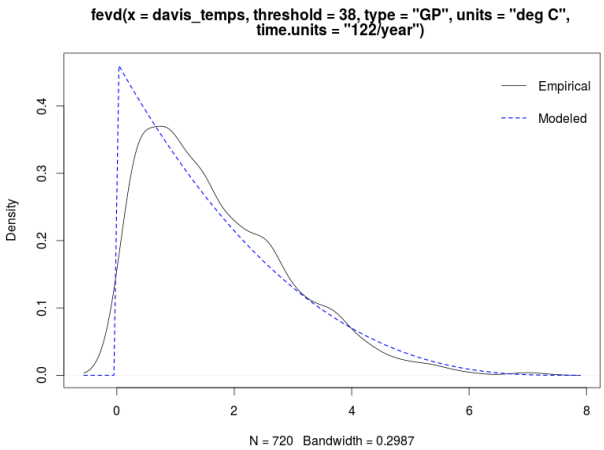
Figure B.1: `threshrange.plot`

To demonstrate the effect of the threshold value on the model fit, we have included examples of fit calculations with threshold values of 37.5 and 38.

```
fevd(x = davis_temps, threshold = 37.5,
    type = "GP", units = "deg C",
    time.units = "122/year")

[1] "Estimation Method used: MLE"

 Negative Log-Likelihood Value:  1456.672

 Estimated parameters:
    scale      shape
2.1256923 -0.2524379

 Standard Error Estimates:
    scale      shape
0.07873964 0.02018613

 Estimated parameter covariance matrix.
              scale         shape
scale  0.006199932 -0.001279104
shape -0.001279104  0.000407480

 AIC = 2917.344

 BIC = 2927.098
```

```
fevd(x = davis_temps, threshold = 38,
    type = "GP", units = "deg C",
    time.units = "122/year")

[1] "Estimation Method used: MLE"

 Negative Log-Likelihood Value:  1067.19

 Estimated parameters:
    scale      shape
2.1482576 -0.2824491

 Standard Error Estimates:
    scale      shape
0.08774070 0.01976493

 Estimated parameter covariance matrix.
              scale          shape
scale  0.007698430 -0.0014430126
shape -0.001443013  0.0003906523

 AIC = 2138.38

 BIC = 2147.539
```





If we just look at the density plots, it would seem that a threshold of 37.5 has a better fit than 38. However, parts of the `fevd` output would seem to say otherwise. The negative log-likelihood, AIC, and BIC values are all smaller for the model fit which was calculated using a threshold of 38. Part of this can be explained by the fact that the dataset for $x = 38$ contains 720 data points, as opposed to 970 data points for $x = 37.5$ (recall that AIC and BIC impose a penalty for including more information in the model, in this case more data points). The difference between standard error estimates for the scale and shape parameters is small enough to be negligible.

B.2 Declustering data

Recall from Chapter 1 that these models assume that the data points are independent of one another. The accuracy of the GEV approximation is somewhat affected when there is temporal dependence of extremes, but due to the nature of the data utilized in the model calculation (one maximum per block), the error due to temporal dependence is minimal; the effects are mainly seen on the value of the location parameter and the relative spread of the temperature magnitudes.

Temporal dependence has a much stronger effect in the GP approximation. Since all data is utilized, rather than one value within a block, it is highly likely that there will be sequential days with very similar temperatures, and this can skew the distribution of temperature excesses. **Declustering** the data reduces this temporal dependence by replacing clustered values with a single point; the other values are replaced by the threshold value, which drops them from consideration for the model fit.

In this example, we once again use the thresholds of $x = 37.5$ and $x = 38$ for declustering. In both instances the negative log-likelihood, AIC and BIC all decreased, but the standard error of the scale and shape parameters increased.

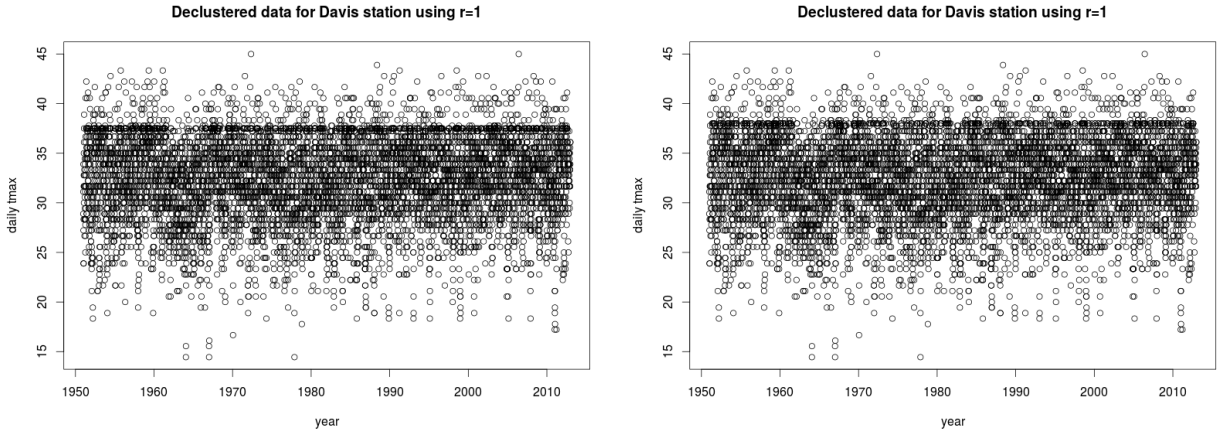
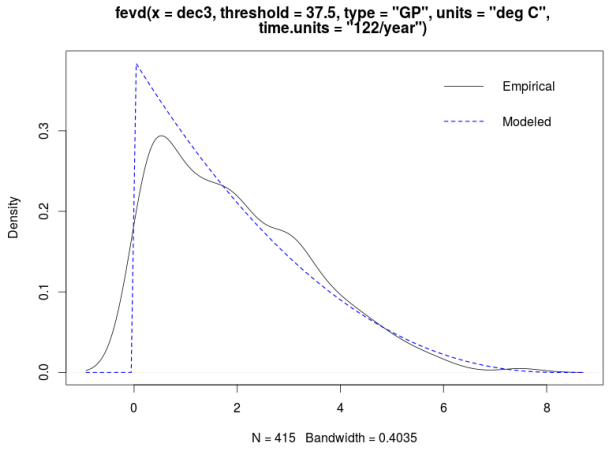
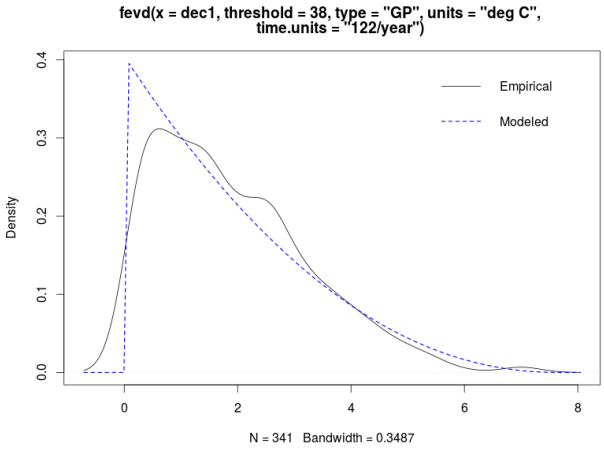


Figure B.2: **Left:** Declustered data set, $x = 37.5$; reduces dataset to 415 data points. **Right:** Declustered data set, $x = 38$; reduces dataset to 341 data points. See Appendix C.3 for declustering function.

```
fevd(x = dec3, threshold = 37.5,
    type = "GP", units = "deg C",
    time.units = "122/year")

[1] "Estimation Method used: MLE"

 Negative Log-Likelihood Value:  678.6957

 Estimated parameters:
    scale      shape
 2.5778919 -0.3115848

 Standard Error Estimates:
    scale      shape
0.14326008 0.03022026

 Estimated parameter covariance matrix.
              scale          shape
scale  0.020523450 -0.0036856145
shape -0.003685615  0.0009132639

 AIC = 1361.391

 BIC = 1369.448
```

```
fevd(x = dec1, threshold = 38,
    type = "GP", units = "deg C",
    time.units = "122/year")

[1] "Estimation Method used: MLE"

 Negative Log-Likelihood Value:  539.7462

 Estimated parameters:
   scale     shape
 2.471628 -0.322044

 Standard Error Estimates:
    scale      shape
0.14885045 0.03161028

 Estimated parameter covariance matrix.
              scale          shape
scale  0.022156457 -0.0040507285
shape -0.004050729  0.0009992098

 AIC = 1083.492

 BIC = 1091.156
```





B.3 Nonstationary threshold calculation

If the data contains any sort of trend, it is advisable to calculate a nonstationary threshold. The following sections demonstrate two possible methods.

B.3.1 Sine function

if we wished to alter the threshold parameter cyclically, this could be achieved with

$$x(t) = x_0 + a\sin(bt) \tag{B.1}$$

where x_0 would be the initial threshold and a and b would tailor the amplitude and period, respectively, of the sine wave. Figure 1.5 is a clear example of why to use this. It shows the daily maximum temperature values by day in JJAS, and there appears to be some sort of quadratic or sinusoidal trend in the temperatures as the season progresses. We would want the threshold value to reflect this rather than having a constant threshold, and so would use an equation in the form of B.1, in this case,

$$x(t) = 36.5 + \left|2.5\sin\left(\frac{\pi d}{122}\right)\right| \text{ where} \tag{B.2}$$

$$d = t - (nyears \times ndays) - 1 \tag{B.3}$$

This equation results in a curve with a minimum of 36.5 at the beginning of June and end of September, and a maximum of 39 at the end of July and beginning of August.

B.3.2 Harmonics function

We begin by calculating v, the 90th percentile for each day y in the season. The functions to output this variable can be found in Appendix C.3.1.

Next, we use linear regression to find the best fitting harmonics function:

```
b<-lm(v~sin(2*pi*y/122)+cos(2*pi*y/122)
      +sin(4*pi*y/122)+cos(4*pi*y/122)+sin(6*pi*y/122)
      +cos(6*pi*y/122)+sin(8*pi*y/61)+cos(8*pi*y/122))
```

Then, looking at the significance of the coefficients:

```
summary(b)

Call:
lm(formula = v ~ sin(2 * pi * y/122) + cos(2 * pi * y/122) +
    sin(4 * pi * y/122) + cos(4 * pi * y/122) + sin(6 * pi *
    y/122) + cos(6 * pi * y/122) + sin(8 * pi * y/61) + cos(8 *
    pi * y/122))

Residuals:
    Min      1Q  Median      3Q     Max
-2.0756 -0.6031  0.2018  0.5039  1.9932

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         37.68124    0.07285 517.243  < 2e-16 ***
sin(2 * pi * y/122)  0.68652    0.10303   6.664 1.02e-09 ***
cos(2 * pi * y/122) -1.26852    0.10303 -12.313  < 2e-16 ***
sin(4 * pi * y/122)  0.14363    0.10303   1.394  0.16603
cos(4 * pi * y/122) -0.28435    0.10303  -2.760  0.00675 **
sin(6 * pi * y/122)  0.17455    0.10303   1.694  0.09297 .
cos(6 * pi * y/122) -0.33446    0.10303  -3.246  0.00154 **
sin(8 * pi * y/61)   0.05522    0.10303   0.536  0.59300
cos(8 * pi * y/122)  0.11715    0.10303   1.137  0.25790
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8047 on 113 degrees of freedom
Multiple R-squared:  0.6612,Adjusted R-squared:  0.6372
F-statistic: 27.57 on 8 and 113 DF,  p-value: < 2.2e-16
```

We will select all coefficients with a significance of at least 0.001 (that is, all coefficients with at least 2 asterisks). Thus, the final equation will be:

```
b_eq<-(b$coefficients[1] + b$coefficients[2]*sin(2*pi*y/122)+b$coefficients[3]*cos(2*pi*y/122)
      +b$coefficients[5]*cos(4*pi*y/122)+b$coefficients[7]*cos(6*pi*y/122))
```

This equation is plotted as the black line in Figure 1.6, which also includes the previously calculated sine-varying threshold for reference.

Appendix C

Some R syntax examples

C.1 Installing and running R

R is most similar to Python in terms of the syntax. It is an interpreted language, and can be run either as a script, or line-by-line from the terminal. There are various GUI interfaces that facilitate the usage of R, such as Rstudio or Rcmdr. Both of the aforementioned programs are free and cross-platform (Windows, Mac, and Linux) but will not work unless the base R program is already installed.

C.1.1 Installing R

Go to `http://cran.r-project.org/mirrors.html` and choose your desired download source. It will direct you to the download page, where you will choose the package for your specified operating system. More detailed download instructions for each operating system can be found on the download page.

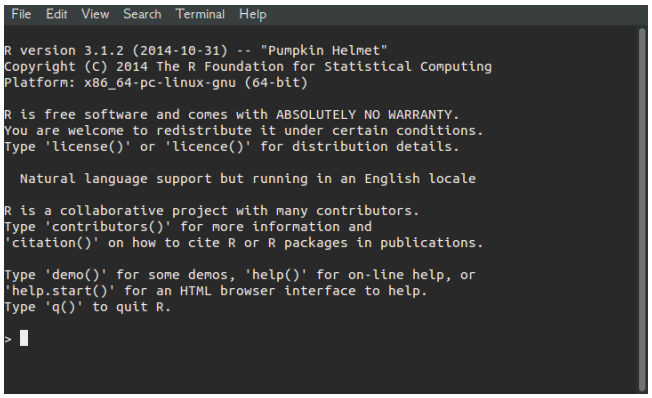
Once you have installed R, you may install a GUI interface if you wish.

Rstudio: `http://www.rstudio.com/products/rstudio/download/`

Rcmdr: See Installing Packages below. Installation notes: `http://socserv.mcmaster.ca/jfox/Misc/Rcmdr/installation-notes.html`

C.1.2 Running R

Base R (no GUI): Open up the R application. Appearances will vary by operating system; in Linux, a terminal window opens up with the R program:



while in Windows/Mac

C.1.3 Installing Packages

In the command line, type

`install.packages("package name")`

The quotes must be included. This will open up a dialog window with a number of sources to download the package from. Select a source and hit "OK". The package will be downloaded and installed.

- To install Rcmdr: `install.packages("Rcmdr")`
- To install `extRemes` package: `install.packages("extRemes")`

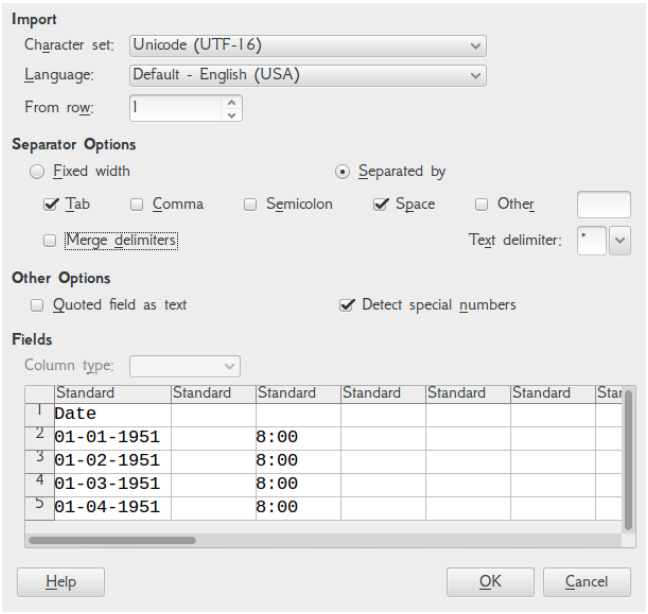
C.2 Formatting text files

In the examples, the observational data came in the form of a text document with columns that contained the dates, observation times, and maximum and minimum temperatures:

```
Date       Time            tmx tmn
01-01-1951  8:00            59  39
01-02-1951  8:00            50  33
01-03-1951  8:00            50  38
01-04-1951  8:00            58  46
...
```

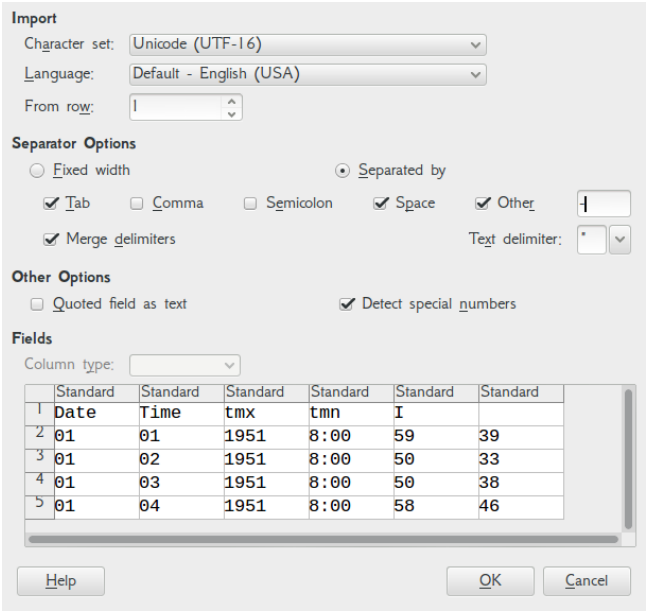
In order to get the data into an R-readable format, it is better to convert the data into a format such as CSV (comma-separated values). This is possible with a spreadsheet application such as Excel (or the open source versions such as LibreOffice Calc). Here, the examples are done in LibreOffice Calc, but the steps for Excel are very similar.

Copy all of the text in the document and open the spreadsheet application, then paste into the first cell. A window like this one should pop up:



Note that there are a couple things that need to be fixed. First of all, there are multiple empty columns. Second of all, the dates are strings rather than number, which will make subsetting difficult.

To split the date string, select the "Other" checkbox and enter the dash in the box next to it. To get rid of the blank columns, select the "Merge delimiters". The result is shown below:



The column titles will have to be adjusted to account for the date being split up, but this is now a usable dataset. Click "OK" and your formatted spreadsheet will show up. Re-label column 1 as "Month", column 2 as "Day", column 3 as "Year", column 4 as "Time", column 5 as "tmx", and column 6 as "tmn". When you are happy with the formatting, save the data as a CSV file.

C.3 Reading and subsetting data in R

Below is an example of what would be included in an R script to process this data (note that lines preceded by # are comment lines). An explanation follows each chunk of code.

```
library(extRemes)
setwd("Research/extremes/")
davis <- read.csv("Davis.csv")
```

1. Load the `extRemes` library
2. Set the working directory to the location of your data
3. Read in the CSV file that contains the data

The station data is formatted as a table with named headers for each column variable. We can subset the data in a number of different ways:

- Using the name of the max temperature column:

  ```
  tmax<-davis$tmx
  ```

  or

  ```
  tmax<-davis[,"tmx"]
  ```

  Note the comma that precedes the column variable name. This indicates that we wish to select all available rows.

- Subsetting by month: if we wanted only August values, we could do

  ```
  tmax_august<-davis[davis$Month==8,"tmx"]
  ```

- Subsetting by column and row numbers

  ```
  tmax_subset<-davis[20:40,5]
  ```

C.3.1 Some user-defined functions

- Subsetting data:

  ```
  #Grab JJAS from station data
  get_data<-function(var){
    dat<-var[((var$Month>=6) & (var$Month<=9) & (var$Year<=2012)),]
    return(dat)
  }
  ```

  This function takes one argument: the dataset that was read into R. It returns all instances where the month column is either 6, 7, 8, or 9 and the year column is less than or equal to 2012. Note the combination of conditional statements used to constrain the values.

  We then subset the data by typing the following:

  ```
  davis2<-get_data(davis)

  #Daily max temps for GP calculations
  davis_temps<-(davis2$tmx-32)*(5/9)
  ```

  The last line converts this data from degrees Fahrenheit to degrees Celsius.

- Extract seasonal maxima for GEV calculations

```
#Get max temp value for data over specified subset and convert from F to C
#This is used for GEV calculations
yearly_max<-function(var){
  year<-unique(var$Year)
  maxc<-c()
  for (i in 1:length(year)){
    maxt = max(var[var$Year==year[i],"tmx"])
    maxc[i] = (maxt-32)*(5/9)
  }
  return(maxc)
}
```

Then run this function by typing

```
davis_max<-yearly_max(davis2)
```

- Calculating temperature excesses with respect to a sine-varying threshold

```
#Function for excess wrt sine threshold
excess_calc<-function(data,ndays,nyears){
  days<-seq(0,(ndays-1))
  day_index<-rep(days,nyears)
  sin_eq = abs(2.5*sin(day_index*(pi/ndays))+36.5)
  excess<-data-sin_eq
}
```

This function takes three arguments: dataset (in this case, daily temperature maxima), the number of days per season (122 for JJAS), and the number of years in the dataset (equivalent to `span`). It works as follows:

1. generate a vector with the sequence from 0 to number of days less 1
2. replicate that vector for the specified number of years
3. calculate the non-stationary threshold value per day
4. calculate the excess by subtracting the threshold from the temperature values

In this example, the excess was calculated with `excess<-excess_calc(davis_temps,122,62)`

- Calculating the 90th data quantile for each day in the season

```
variable_thresh<-function(data,ndays,nyears){
  q90<-c()
  for (d in 1:ndays){
    subset<-data[d+ndays*seq(0,nyears-1)]
    q90[d]<-quantile(subset,probs=0.9)
  }
  return(q90)
}
```

- Declustering daily max temps per year

```
#Function for declustering data per year
#Can't just use decluster function on its own because does not take
#non-consecutive dates for new years into account

yearly_decluster<-function(var,ndays,thresh,r_val){
  num_years = length(var)/ndays
  dec_data<-c()
  for (i in 0:(num_years-1)){
    subset_start = i*ndays+1
    subset_end = ndays*(i+1)
    var_subset = var[subset_start:subset_end]
    var_decluster = decluster(var_subset,thresh,r=r_val)
    dec_data = c(dec_data,var_decluster)
  }
  return(dec_data)
}
```

This function takes 4 arguments: the dataset with the daily maxima, the number of days per block of time (in this case, 122 days for JJAS), the threshold value, and the number of additional days that constitute cluster dependence. It works as follows:

1. Calculate the number of years in the data vector
2. Subset the data vector for each year and run the decluster function with the given threshold and cluster value.
3. Append the newly declustered subset to the output vector.
4. Return the vector with the data, declustered by year.

In the example, declustered data was obtained by typing

```
yearly_decluster(davis_temps,122,38,1)
```

C.4 Plots

Figure 1.1 shows plots of the probability density functions for each of the distributions outlined in Table 1.1. These plots were generated using the `devd` function as follows:

```
#Distribution functions
x<-seq(0,4,0.01)
#Frechet
y1<-devd(x,loc=1,scale=0.5,shape=0.5,type="GEV")
#Gumbel
y2<-devd(x,loc=1,scale=0.5,shape=0,type="GEV")
#Weibull
y3<-devd(x,loc=1,scale=0.5,shape=-0.5,type="GEV")
#Pareto
y4<-devd(x,loc=1,scale=0.5,shape=0.5,type="GP")
#Exponential
y5<-devd(x,loc=1,scale=0.5,shape=0,type="GP")
#Beta
y6<-devd(x,loc=1,scale=0.5,shape=-0.5,type="GP")

#GEV plots
plot(x,y1,type='l',col='red',main="GEV distribution functions",xlab="x",ylab="y")
lines(x,y2)
lines(x,y3,col='blue')

#GP plots
plot(x,y4,type='l',col='red',main="GP distribution functions",xlab="x",ylab="y")
lines(x,y5)
lines(x,y6,col='blue')
```

Bibliography

[1] Coles, S. 2001. An Introduction to Statistical Modeling of Extreme Values. Springer-Verlag, London, United Kingdom,208 pp.

[2] Gilleland, E. and R.W. Katz. 2014. extRemes 2.0: An Extreme Value Analysis Package in R. Journal of Statistical Software.

[3] IPCC. 2014. Summary for Policymakers. In: Climate Change 2014: Impacts, Adaptation, and Vulnerability. Part A: Global and Sectoral Aspects. Contribution of Working Group II to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change [Field, C.B., V.R. Barros, D.J. Dokken, K.J. Mach, M.D. Mastrandrea, T.E. Bilir, M. Chatterjee, K.L. Ebi, Y.O. Estrada, R.C. Genova, B. Girma, E.S. Kissel, A.N. Levy, S. MacCracken, P.R. Mastrandrea, and L.L. White (eds.)]. Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA, pp. 1-32.

Index