Part III: Weibull plots

Alison Peard

2025-10-22

Define some simulation parameters. We will use a Rayleigh distribution, with scale $C = 10$, $r = 22$ events per year over $R = 16$ years, and extract the top $M = 30$ values as exceedances.

```
# Weibull parameters
w_true <- 2           # Shape parameter (typical for synoptic winds)
C <- 10               # Scale parameter
r <- 22               # Rate of independent peaks per epoch (year)
R <- 16               # Number of epochs (years of data)
M <- 30               # Number of POT values to use

# Return level to predict
MRI <- 50             # 50-year return level

# Number of bootstrap trials
n_trials <- 3 # 1000
```

First, we generate POT data from a Weibull parent.

```
set.seed(42)  # For reproducibility

# Generate parent Weibull data
N <- r * R  # Total population
parent_data <- C * (-log(runif(N)))^(1/w_true) # equiv to rweibull(N, shape = w_true, scale = C)
parent_data <- sort(parent_data, decreasing = TRUE)

# Select top M values (POT approach)
V <- parent_data[1:M]

cat("Generated POT data:\n")
```

```
## Generated POT data:
```

```
cat(sprintf("  Number of POT values (M): %d\n", M))
```

```
##   Number of POT values (M): 30
```

```
cat(sprintf("  Record length (R): %d epochs\n", R))
```

```
##   Record length (R): 16 epochs
```

```
cat(sprintf("  Total samples (N): %d\n", N))
```

```
##   Total samples (N): 352
```

```
cat(sprintf("  Rate per epoch (r): %d\n", r))
```

```
##   Rate per epoch (r): 22
```

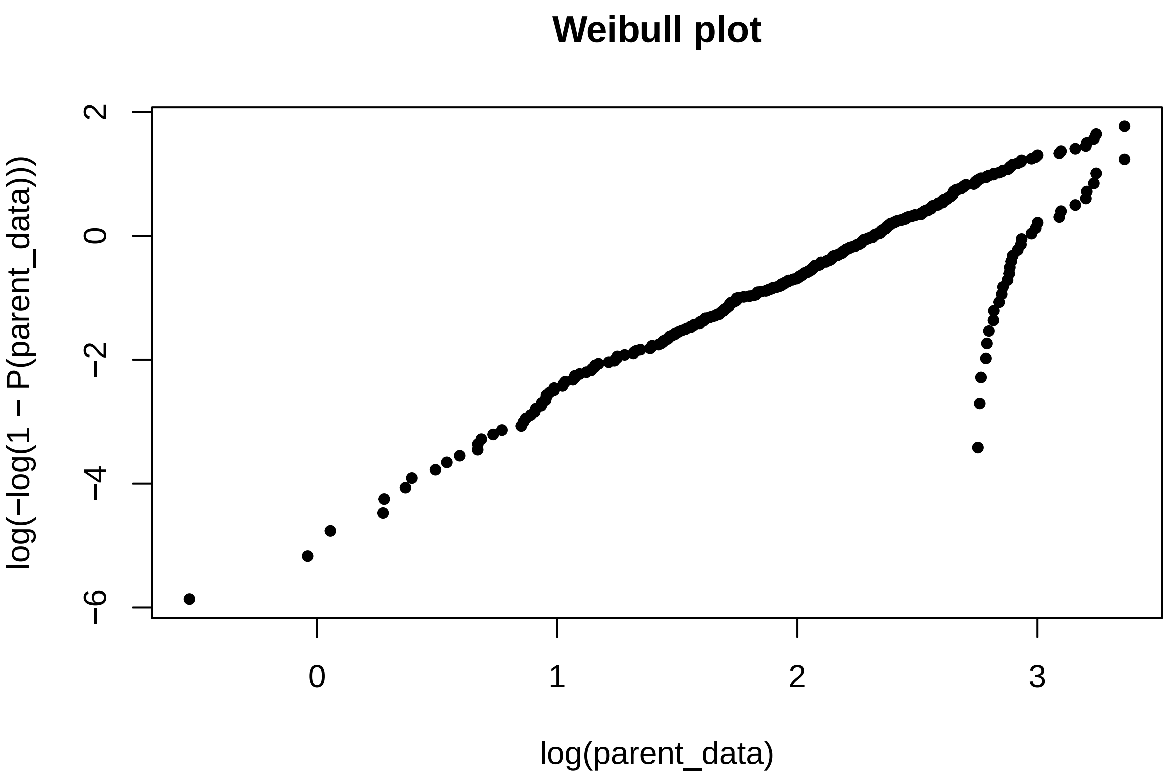
```r
cat(sprintf("  Top value: %.2f\n", max(V)))
```

```
##   Top value: 28.88
```

```r
cat(sprintf("  Threshold (M-th value): %.2f\n\n", min(V)))
```

```
##   Threshold (M-th value): 15.68
```

```r
P <- function(x) {
  m <- rank(x)
  N <- length(x)
  m / (N+1)
}

plot(log(parent_data), log(-log(1-P(parent_data))), pch=20, main="Weibull plot")
points(log(V), log(-log(1-P(V))), pch=20) # so we should fit Weibull to the parent
```



Step I: Assess climate mix / adherance to a Weibull fit

Load the PoI data from the UK power study.

```r
INPUT <- "../results/testing/pois.nc"
I_POI <- 1
I_VAR <- 1

src <- tidync(INPUT)
df  <- src |> hyper_tibble(force = TRUE)
pois <- unique(df$poi)
vars <- unique(df$field)
poi <- pois[I_POI]
var <- vars[I_VAR]
```

```r
print(paste0("Modelling " , var, " for ", poi))
```

```
## [1] "Modelling u10_gust for birmingham"
```

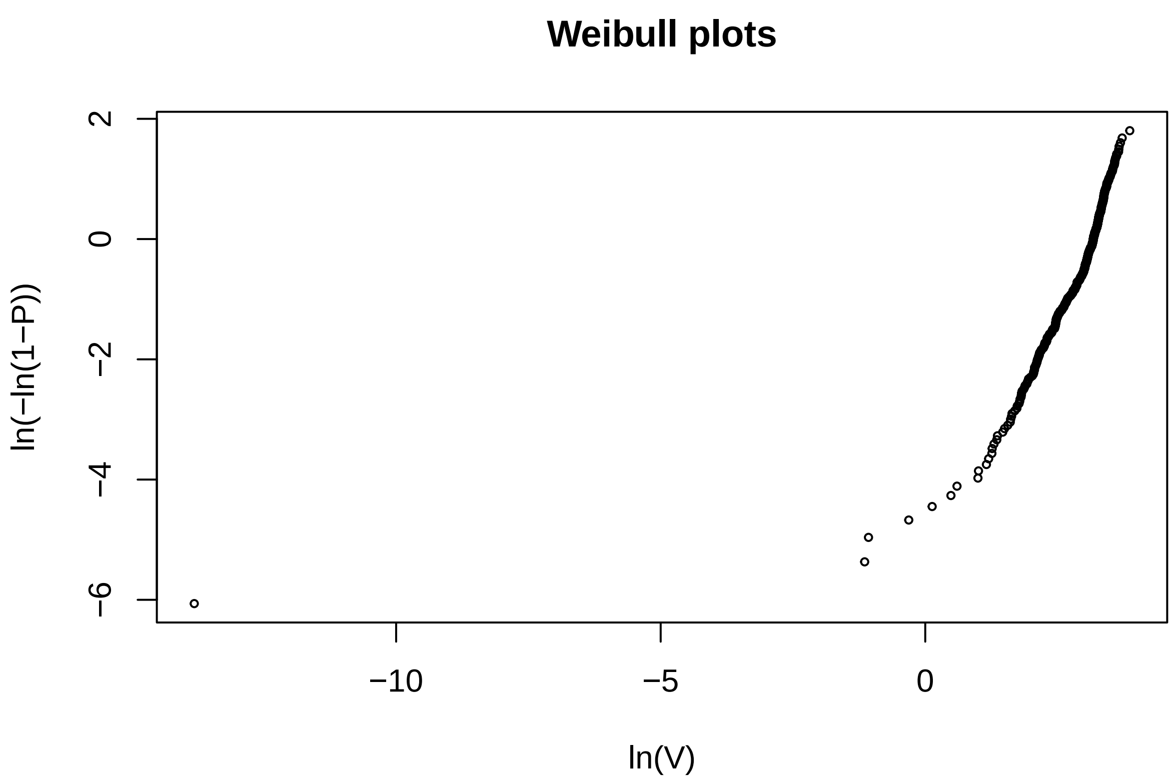
```r
df <- df[(df$poi == poi) & (df$field == var), ]
print(paste0("Extracted df with length: ", dim(df)[1], " for (", poi, ", ", var, ")"))
```

```
## [1] "Extracted df with length: 429 for (birmingham, u10_gust)"
```

```r
x <- df$anomaly
```

See how it looks on a Weibull plot.

```r
x_pos <- x + abs(min(x)) + 1e-6

V <- x_pos

par(mfcol=c(1, 1))

plot(log(V), log(-log(1-P(V))),
     cex = 0.5,
     main = "Weibull plots",
     xlab = "ln(V)", ylab = "ln(-ln(1-P))"
)
```



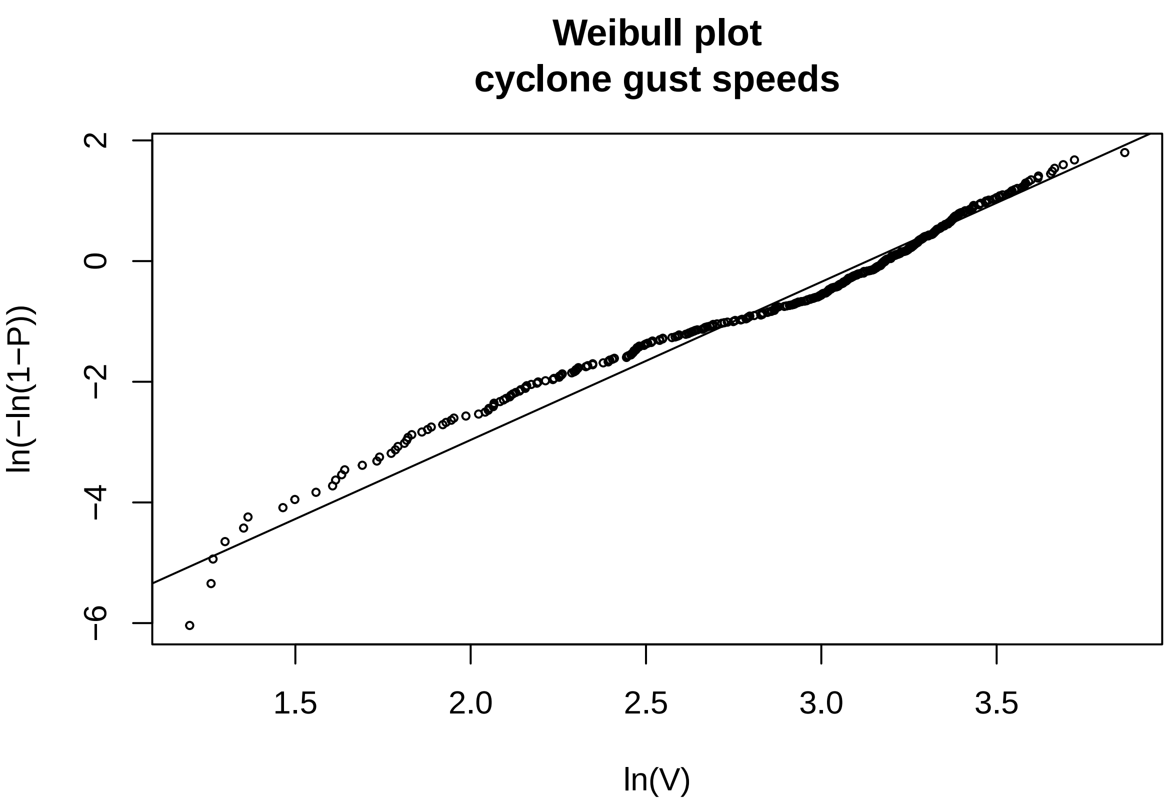
It's pretty clean, actually, we can try and fit a Weibull distribution to the parent.

```r
nll <- function(x, par) {
  shape <- par[1]
  scale <- par[2]
  -sum(dweibull(x, shape = shape, scale = scale, log = TRUE))
}
```

```
x_pos <- x + abs(min(x)) + 1e-6
N <- length(x_pos)
M <- N - 10
x_tail <- sort(x_pos, decreasing = TRUE)[1:M]

V <- x_tail

fit <- optim(c(2, mean(V)), nll, x = V,
             method = "L-BFGS-B",
             lower = c(0.01, 0.01))

w <- fit$par[1]
C <- fit$par[2]

cat(sprintf("shape = %f, scale = %f", w, C))
```

```
## shape = 2.619148, scale = 22.914162
```

```
par(mfcol=c(1, 1))

plot(log(V), log(-log(1-P(V))),
     cex = 0.5,
     main = "Weibull plot\ncyclone gust speeds",
     xlab = "ln(V)", ylab = "ln(-ln(1-P))"
)

abline(a = -w*log(C), b = w, col = "black", lwd = 1)
```



Step II: Preconditioning

Linearise the data by taking it to the power $z = z^{\xi}$.

```
library(evir)  # for Gumbel fitting if needed

yximis <- function(M = 100, R = 1) {
  y <- rep(0, M)
  var <- y
  y[1] <- -digamma(1) + log(R)  # Euler's constant = -digamma(1) = 0.5772...
  var[1] <- pi^2 / 6

  if (M > 1) {
    for (i in 2:M) {
      y[i] <- y[i-1] - 1/(i-1)
      var[i] <- var[i-1] - 1/(i-1)^2
    }
  }

  data.frame(mean = y, var = var)
}

qximis <- function(y, U, D, w) {
  (U^w + y * D^w)^(1/w)
}

pot.XMS <- function(V, R = length(V), w = 1) {
  x <- sort(V^w, decreasing = TRUE)
  y <- yximis(length(x), R) # get reduced variate
  my <- y$mean
  wt <- 1 / y$var

  lm(x ~ my, weights = wt) # fit for U and D
}
```

Fit to the data

```
fit <- pot.XMS(V, R, w = w)
# Extract parameters (note: these are in transformed space V^w)
U_w <- coef(fit)[1]
D_w <- coef(fit)[2]

# Transform back to original space
U <- U_w^(1/w)
D <- D_w^(1/w)

cat("XIMIS fitted parameters:\n")
```

```
## XIMIS fitted parameters:
```

```
cat(sprintf("  Mode (U): %.2f\n", U))
```

```
##   Mode (U): 36.53
```

```
cat(sprintf("  Dispersion (D): %.2f\n", D))
```

```
##   Dispersion (D): 23.24
```

```r
cat(sprintf("  Shape (w): %.2f (fixed/est)\n\n", w))
```

```
##   Shape (w): 2.62 (fixed/est)
```

Predict some return levels.

```r
MRI <- 10000

y_MRI <- -log(-log(1 - 1/MRI))
V_MRI_pred <- qximis(y_MRI, U, D, w)

cat(paste0(MRI, "-year return level predictions:\n"))
```

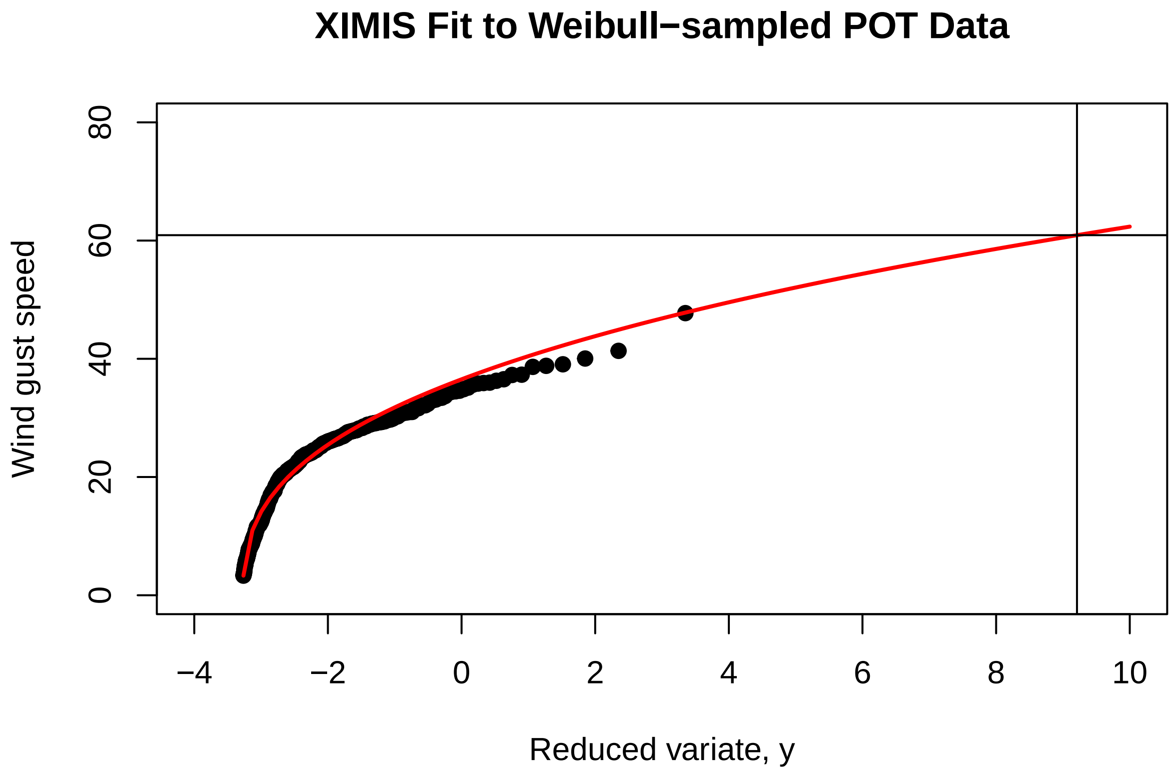
```
## 10000-year return level predictions:
```

```r
cat(sprintf("  XIMIS prediction: %.2f\n", V_MRI_pred))
```

```
##   XIMIS prediction: 60.91
```

Plot the fit.

```r
y_vals <- yximis(length(V), R)$mean

plot(y_vals, sort(V, decreasing = TRUE),
     xlab = "Reduced variate, y",
     ylab = "Wind gust speed",
     main = "XIMIS Fit to Weibull-sampled POT Data",
     pch = 19, col = "black",
     xlim = c(-4, 10),
     ylim = c(0, 80)
)

# Add fitted line
y_seq <- seq(min(y_vals), 10, length.out = 100)
V_fit <- qximis(y_seq, U, D, w)
lines(y_seq, V_fit, col = "red", lwd = 2)

# Add 50-year prediction
#points(y_MRI, V_MRI_pred, pch = 17, col = "blue", cex = 1.5)
abline(v = y_MRI)
abline(h = V_MRI_pred)
```


XIMIS Fit to Weibull−sampled POT Data
Wind gust speed
0
20
40
60
80
−4
−2
0
2
4
6
8
10
Reduced variate, y