

The `powerLaw` package: Examples

Colin S. Gillespie

Last updated: April 16, 2015

1 Discrete data: Moby Dick

The Moby Dick dataset contains the frequency of unique words in the novel Moby Dick by Herman Melville. This data set can be downloaded from

<http://tuvalu.santafe.edu/~aaronc/powerlaws/data.htm>

or loaded directly

```
library("powerLaw")
data("moby")
```

To fit a discrete power law to this data¹, we use the `displ` constructor

```
m_pl = displ$new(moby)
```

The resulting object, `m_pl`, is a `displ`² object. It also inherits the `discrete_distribution` class. After creating the `displ` object, a typical first step would be to infer model parameters.³ We can estimate the lower threshold, via

```
est = estimate_xmin(m_pl)
m_pl$setXmin(est)
```

For a given value x_{\min} , the scaling parameter is estimated by numerically optimising the log-likelihood. The optimiser is *initialised* using the analytical MLE

$$\hat{\alpha} \simeq 1 + n \left[\sum_{i=1}^n \log \left(\frac{x_i}{x_{\min} - 0.5} \right) \right]^{-1}.$$

This yields a threshold estimate of $x_{\min} = 7$ and scaling parameter $\alpha = 1.95$, which matches results found in Clauset et al. (2009).

Alternatively, we could perform a parameter scan for each value of x_{\min}

¹The object `moby` is a simple R vector.

²`displ`: discrete power law.

³When the `displ` object is first created, the default parameter values are `NULL` and x_{\min} is set to the minimum x -value.

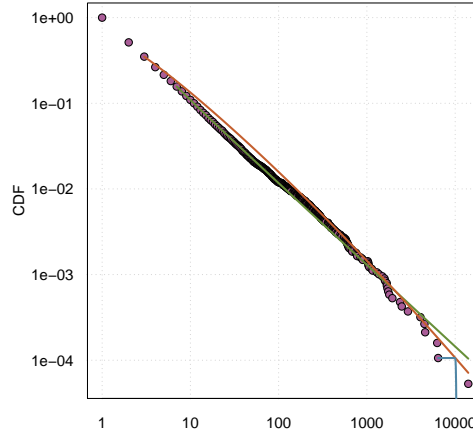


Figure 1: Data CDF of the Moby Dick data set. The fitted power law (green line), log-normal (red line) and poisson (blue) distributions are also given.

```
estimate_xmin(m_pl, pars=seq(1.8, 2.3, 0.1))
```

To fit a discrete log-normal distribution, we follow a similar procedure, except we begin by creating a `dislnorm` object⁴

```
m_ln = dislnorm$new(moby)
est = estimate_xmin(m_ln)
```

which yields a lower threshold of $x_{\min} = 3$ and parameters $(-17.9, 4.87)$. A similar procedure is applied to fit the Poisson distribution; we create a distribution object using `dispois`, then fit as before.

The data CDF and lines of best fit can be easily plotted

```
plot(m_pl)
lines(m_pl, col=2)
lines(m_ln, col=3)
lines(m_pois, col=4)
```

to obtain figure 1. It is clear that the Poisson distribution is not appropriate for this data set. However, the log-normal and power law distribution both provide reasonable fits to the data.

1.1 Parameter uncertainty

To get a handle on the uncertainty in the parameter estimates, we use a bootstrapping procedure, via the `bootstrap` function. This procedure can be applied to any distribution object.⁵ Furthermore, the bootstrap procedure can utilize multiple CPU cores to speed up inference.⁶

⁴`dislnorm`: discrete log normal object

⁵For example, `bootstrap(m_ln)`.

⁶The output of this bootstrapping procedure can be obtained via `data(bootstrap_moby)`.

```
## 5000 bootstraps using two cores
bs = bootstrap(m_pl, no_of_sims=5000, threads=2)
```

By default, the `bootstrap` function will use the maximum likelihood estimate to estimate the parameter and check all values of x_{\min} . When possible x_{\min} values are large, then it is recommend that the search space is reduced. For example, this function call

```
bootstrap(m_pl, xmin = seq(2, 20, 2))
```

will only calculate the Kolmogorov-Smirnoff statistics at values of x_{\min} equal to

$$2, 4, 6, \dots, 20.$$

A similar argument exists for the parameters.⁷

The bootstrap function, returns `bs_xmin` object that has three components:

1. The goodness of fit statistic obtained from the Kolmogorov-Smirnoff test. This value should correspond to the value obtained from the `estimate_xmin` function.
2. A data frame containing the results for the bootstrap procedure.
3. The average simulation time, in seconds, for a single bootstrap.

The bootstrap results can be explored in a variety way. First we can estimate the standard deviation of the parameter uncertainty, i.e.

```
sd(bs$bootstraps[,2])
## [1] 1.780825
sd(bs$bootstraps[,3])
## [1] 0.02428821
```

Alternatively, we can visualise the results using the `plot` function:

```
## trim=0.1 only displays the final 90% of iterations
plot(bs, trim=0.1)
```

to obtain figure 2. This top row of graphics in figure 2 give a 95% confidence interval for the mean estimate of the parameters. The bottom row of graphics give a 95% confidence for the standard deviation of the parameters. The parameter `trim` in the `plot` function controls the percentage of samples displayed.⁸ When `trim=0.1`, we only display the final 90% of data.

We can also construct histograms.

```
hist(bs$bootstraps[,2])
hist(bs$bootstraps[,3])
```

to get figure 3.

A similar bootstrap analysis can be obtained for the log-normal distribution

⁷For single parameter models, `pars` should be a vector. For the log-normal distribution, `pars` should be a matrix of values.

⁸When `trim=0`, all iterations are displayed.

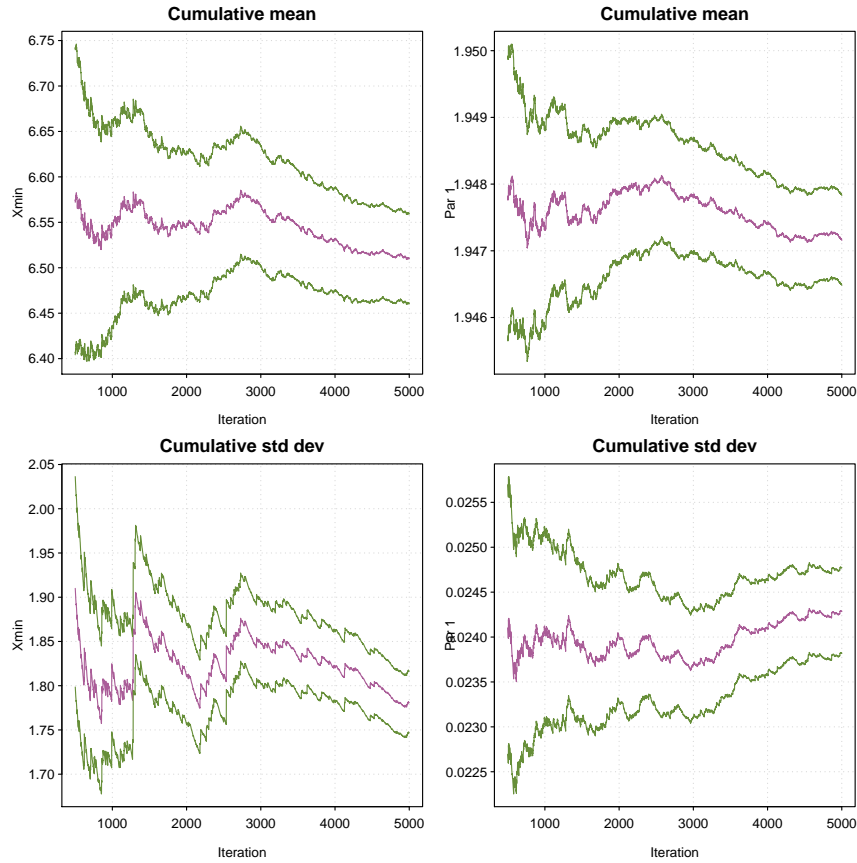


Figure 2: Results from the standard bootstrap procedure (for the power law model) using the Moby Dick data set: `bootstrap(m_pl)`. The top row shows the mean estimate of parameters x_{\min} and α . The bottom row shows the estimate of standard deviation for each parameter. The dashed-lines give approximate 95% confidence intervals. After 5,000 iterations, the standard deviation of x_{\min} and α is estimated to be 2.1 and 0.03 respectively.

```
bs1 = bootstrap(m_ln)
```

in this case we would obtain uncertainty estimates for both of the log-normal parameters.

1.2 Testing the power law hypothesis

Since it is possible to fit a power law distribution to *any* data set, it is appropriate to test whether the observed data set actually follows a power law. [Clauset et al. \(2009\)](#) suggest that this hypothesis is tested using a goodness-of-fit test, via a bootstrapping procedure. This test generates a p -value that can be used to quantify the plausibility of the hypothesis. If the p -value is large, than any difference between the empirical data and the model can be explained with statistical fluctuations. If $p \simeq 0$, then the model does not provide a plausible fit to the data and another distribution may be more appropriate. In this scenario,

H_0 : data is generated from a power law distribution.

H_1 : data is not generated from a power law distribution.

To test these hypothesis, we use the `bootstrap_p` function

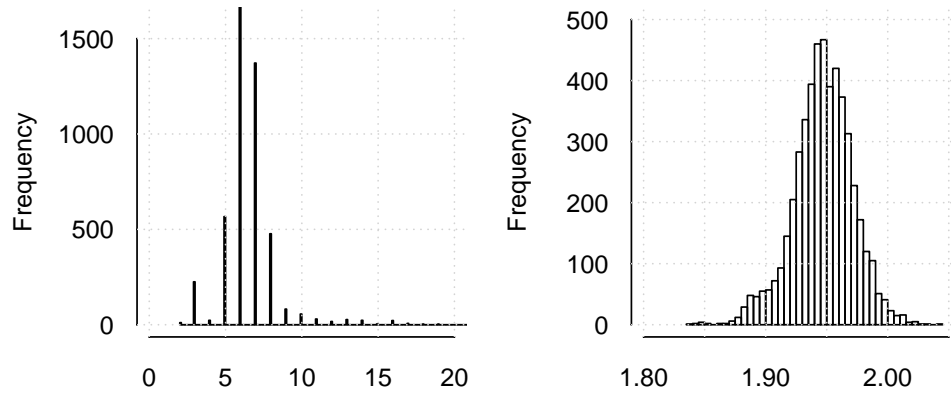


Figure 3: Characterising uncertainty in parameter values. (a) x_{\min} uncertainty (standard deviation 2) (b) α uncertainty (std dev. 0.03)

```
bs_p = bootstrap_p(m_pl)
```

The point estimate of the p -value is one of the elements of the `bs_p` object⁹

```
bs_p$p
## [1] 0.6824
```

Alternatively we can plot the results

```
plot(bs_p)
```

to obtain figure 4. The graph in the top right hand corner gives the cumulative estimate of the p -value; the final value of the purple line corresponds to `bs_p$p`. Also given are approximate 95% confidence intervals.

1.3 Comparing distributions

A second approach to test the power law hypothesis is a direct comparison of two models.¹⁰ A standard technique is to use Vuong's test, which is a likelihood ratio test for model selection using the Kullback-Leibler criteria. The test statistic, R , is the ratio of the log-likelihoods of the data between the two competing models. The sign of R indicates which model is *better*. Since the value of R is obviously subject to error, we use the method proposed by [Vuong \(1989\)](#).

To compare two distributions, each distribution must have the same lower threshold. So we first set the log normal distribution object to have the same x_{\min} as the power law object

```
m_ln$setXmin(m_pl$getXmin())
```

Next we estimate the parameters for this particular value of x_{\min} :

⁹Also given is the average time of a single bootstrap: `bs_p$sim_time = 3.76` seconds.

¹⁰While the bootstrap method is useful, it is computationally intensive and will be unsuitable for most models.

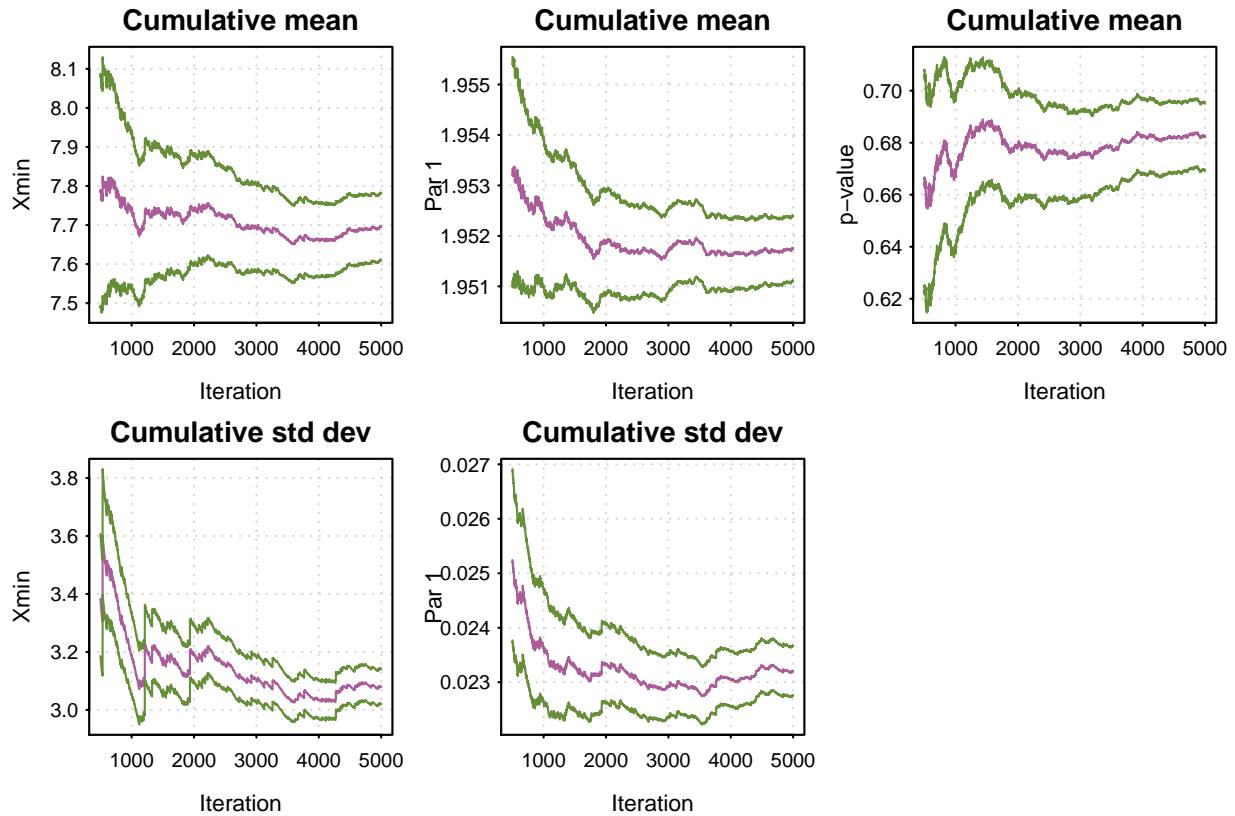


Figure 4: Results from the bootstrap procedure (for the power law model) using the Moby Dick data set: `bootstrap_p(m_pl)`. The top row shows the mean estimate of parameters x_{\min} , α and the p -value. The bottom row shows the estimate of standard deviation for each parameter. The dashed-lines give approximate 95% confidence intervals.

```
est = estimate_pars(m_ln)
m_ln$setPars(est)
```

Then we can compare distributions

```
comp = compare_distributions(m_pl, m_ln)
```

This comparison gives a p -value of 0.6824. This p -value corresponds to the p -value on page 29 of the [Clauset et al.](#) paper (the paper gives 0.69).

Overall these results suggest that one model can't be favoured over the other.

1.4 Investigating the lower bound

The estimate of the scaling parameter, α , is typically highly correlated with the threshold limit, x_{\min} . This relationship can be easily investigated with the `powerLaw` package. First, we create a vector of thresholds to scan

```
xmins = seq(1, 1001, 5)
```

then a vector to store the results

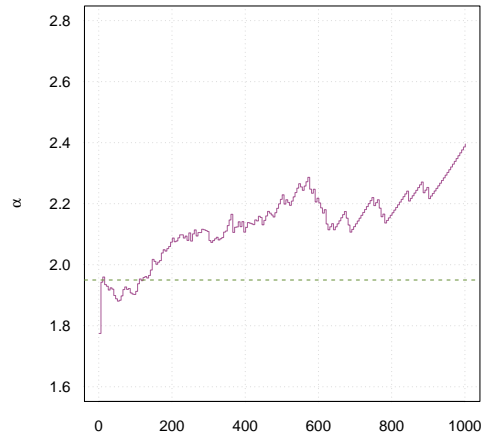


Figure 5: Estimated parameter values conditional on the threshold, x_{\min} . The horizontal line corresponds to $\alpha = 1.95$.

```
est_scan = 0*xmins
```

Next, we loop over the x_{\min} values and estimate the parameter value conditional on the x_{\min} value

```
for(i in seq_along(xmins)){
  m_pl$setXmin(xmins[i])
  est_scan[i] = estimate_pars(m_pl)$pars
}
```

The results are plotted figure 5. For this data set, as the lower threshold increases, so does the point estimate of α .

2 Continuous data: electrical blackouts

In this example, we will investigate the numbers of customers affected in electrical blackouts in the United States between 1984 and 2002 (see [Newman \(2005\)](#) for further details). The data set can be downloaded from Clauset's website

<http://tuvalu.santafe.edu/~aaronc/powerlaws/data/blackouts.txt>

and loaded into R in the usual way

```
blackouts = read.table("blackouts.txt")
```

Although the `blackouts` data set is discrete, since the values are large it makes sense to treat the data as continuous. Continuous power law objects take vectors as inputs, so

```
m_bl = conpl$new(blackouts$V1)
```

then we estimate the lower-bound via

```
est = estimate_xmin(m_bl)
```

This gives a point estimate of $x_{\min} = 94285$. We can then update the distribution object

```
m_bl$setXmin(est)
```

and plot the data with line of best fit

```
plot(m_bl)
lines(m_bl, col=2, lwd=2)
```

to get figure 6. To fit a discrete log-normal distribution we follow a similar procedure

```
m_bl_ln = conlnorm$new(blackouts$V1)
est = estimate_xmin(m_bl_ln)
m_bl_ln$setXmin(est)
```

and add the line of best fit to the plot via

```
lines(m_bl_ln, col=3, lwd=2)
```

It is clear from figure 6 that the log-normal distribution provides a better fit to this data set.

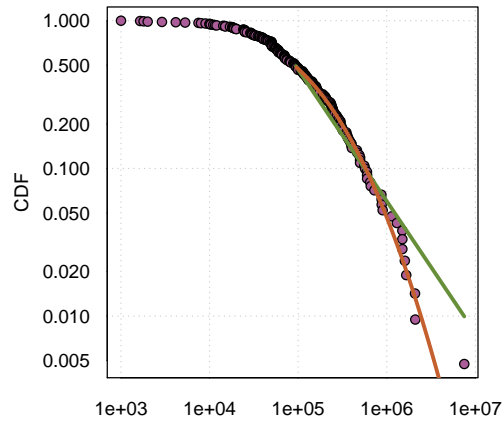


Figure 6: CDF plot of the blackout dataset with line of best fit. Since the minimum value of x is large, we fit a continuous power law as this is more efficient. The power law fit is the green line, the discrete log-normal is the red line.

3 Multiple data sets: the American-Indian war

In a recent paper, [Bohorquez et al.](#) investigated insurgent attacks in Afghanistan, Iraq, Colombia, and Peru. Each time, the data resembled power laws. [Friedman](#) used the power law nature of casualties to infer under-reporting in the American-Indian war. Briefly, by fitting a power law distribution to the observed process, the latent, unobserved casualties can be inferred ([Friedman \(2014\)](#)).

The number of casualties observed in the American-Indian War can be obtained via

```
data("native_american")
data("us_american")
```

Each data set is a data frame with two columns. The first column is number of casualties recorded, the second the conflict date

```
head(native_american, 3)
```

```
##   Cas      Date
## 1  18 1776-07-15
## 2  26 1776-07-20
## 3  13 1776-07-20
```

The records span around one hundred years, 1776 – 1890. The data is plotted in figure 7.

It is straightforward to fit a discrete power law to this data set. First, we create discrete power law objects

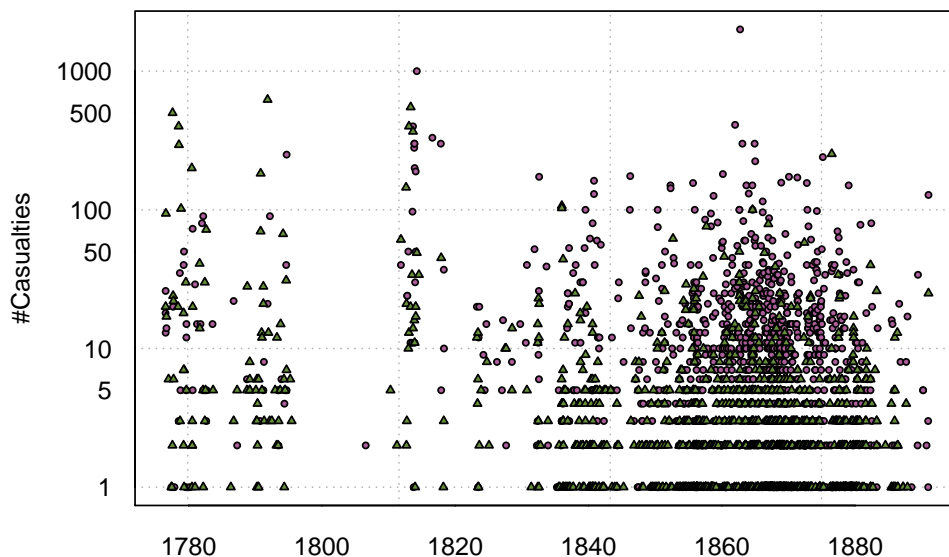


Figure 7: Casualty record for the Indian-American war, 1776 – 1890. Native Americans casualties (purple circles) and US Americans casualties (green triangles). Data taken from [Friedman \(2014\)](#).

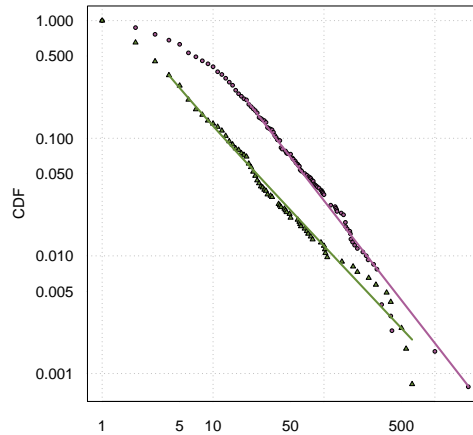


Figure 8: Plots of the CDFs for the Native American and US American casualties. The lines of best fit are also given.

```
m_na = displ$new(native_american$Cas)
m_us = displ$new(us_american$Cas)
```

then we estimate x_{\min} for each data set

```
est_na = estimate_xmin(m_na, pars=seq(1.5, 2.5, 0.01))
est_us = estimate_xmin(m_us, pars=seq(1.5, 2.5, 0.01))
```

and update the power law objects

```
m_na$setXmin(est_na)
m_us$setXmin(est_us)
```

The resulting fitted distributions can be plotted on the same figure

```
plot(m_na)
lines(m_na)
## Don't create a new plot, just store the output
d = plot(m_us, draw=FALSE)
points(d$x, d$y, col=2)
lines(m_us, col=2)
```

The result is given in figure 8. The tails of the distributions appear to follow a power law. This is consistent with the expectation that smaller-scale engagements are less likely to be recorded. However, for larger scale engagements, it is very likely that a record is made.

References

J.C. Bohorquez, S. Gourley, A.R. Dixon, M. Spagat, and N.F. Johnson. Common ecology quantifies human insurgency. *Nature*, 462(7275):911–914, 2009.

- A. Clauset, C.R. Shalizi, and M.E.J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, 2009.
- J.A. Friedman. Using power laws to estimate conflict size. *The Journal of Conflict Resolution*, 2014.
- M.E.J. Newman. Power laws, Pareto distributions and Zipf’s law. *Contemporary Physics*, 46(5): 323–351, 2005.
- Q.H. Vuong. Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica: Journal of the Econometric Society*, 57:307–333, 1989.