About the `optim` function in R

October 17, 2016

In numerous situations of interest it is not possible to compute the maximum likelihood estimator analytically, and one resorts to numerical methods. Consider the following real dataset of the 30 largest 24-hour rainfall events in Allentown, starting in 1952 and measured in inches:

```
> extremeRainfall <- c(6.37, 5.85, 4.86, 4.19, 4.15, 4.06,
+     3.67, 3.63, 3.5, 3.42, 3.37, 3.35, 3.34, 3.28, 3.24,
+     3.23, 3.23, 3.21, 3.12, 3.06, 3.02, 3.01, 3, 2.99,
+     2.92, 2.91, 2.9, 2.87, 2.82, 2.79)
```

We are going to fit this dataset with a Gumbel distribution. You can find out more about this probability distribution at
http://en.wikipedia.org/wiki/Gumbel_distribution . The choice of this distribution may be justified by the fact that it is well suited to the modelling of extreme events, and therefore potentially relevant to the dataset above [check the application section on Wikipedia for example]. The cdf of a Gumbel distribution of parameters $\mu, \beta \in \mathbb{R} \times \mathbb{R}^{++}$ is, for any $x \in \mathbb{R}$,

$$\mathbb{P}\left(X \leq x; \mu, \beta\right) = \exp\left\{-\exp\left(-(x-\mu)/\beta\right)\right\} \quad .$$

Compute the probability density function. The function below evaluates this pdf for any value `x`. Note the convenient `log.p` argument (most `p`− and `d`− `R` functions have this, see, eg `dpois` or `dexp`) which is such that the function returns the log density if (`log.p` = `TRUE`) (note that if no value is specified then by default `log.p` = `FALSE`)

```
> dgumbel <- function(x, mu, beta, log.p = FALSE) {
+     stopifnot(all(beta > 0))
+     z <- exp(-(x - mu)/beta)
+     logf <- log(z) - z - log(beta)
+     if (!log.p)
+         return(exp(logf))
+     else return(logf)
+ }
```

Check that the log-likelihood for data $x_1, x_2, \dots, x_n$ modelled as realisations of iid random variables with common distribution a Gumbel(μ, β) is

$$\ell(\theta; \mathbf{x}) = -\sum_{i=1}^{n} (x_i - \mu)/\beta + \exp\left(-(x_i - \mu)/\beta\right) + \ln(\beta) \quad ,$$

and convince yourself that it is not possible to find an analytical expression for the solution of $\nabla \ell(\theta; \mathbf{x}) = 0$. The following function returns the value of this log-likelihood for the set of observations given above. Note that here the expected input for the function is a vector `theta` = `c(mu, beta)`. This is because of the way the function `optim` works (and the function `optim` will call the `ell` function in order to carry out the maximisation). Note the tests to ensure that the input of the function is a vector of length 2 and that β (contained in the second component `theta[2]` of `theta`) is strictly positif. If this is not the case the function returns `NA` (Not Available, inadmissible choice for β).

```
> ell <- function(theta) {
+     stopifnot(is.vector(theta), length(theta) == 2)
+     if (theta[2] <= 0)
+         return(NA)
+     sum(dgumbel(extremeRainfall, mu = theta[1], beta = theta[2],
+           log.p = TRUE))
+ }
```

Now we are ready to maximise this log-likelihood with `optim`. Things to note (more details in the helpfile of the function):

1. `optim`'s default is to minimise, and to change this to maximise, pass the argument "$\texttt{control} = \texttt{list}(\texttt{fnscale} = -1)$",
2. if the parameter space is bounded, then it's best to use the argument "$\texttt{method} = \texttt{L} - \texttt{BFGS} - \texttt{B}$",
3. if using "$\texttt{L} - \texttt{BFGS} - \texttt{B}$" supply bounds in the "`lower`" and "`upper`" arguments. Don't go all the way to zero if the lower bound is, say, $\beta > 0$, use something small like $1e - 2$ instead (sometimes you have to fiddle with this number),
4. pass in an initial choice of parameters (argument '`par`') with names attached, which makes it easier to interpret the result ,
5. pass in the function to optimise as argument "`fn`",
6. later on, it will be useful to return the Hessian matrix, using the argument "$\texttt{hessian} = \texttt{TRUE}$". This is an easy way to attach names.

```
> init <- c(mu = 0, beta = 1)
> opt <- optim(par = init, fn = ell, method = "L-BFGS-B",
+     lower = c(-Inf, 0.01), upper = c(Inf, Inf), hessian = TRUE,
+     control = list(fnscale = -1, trace = 10))

final  value 27.192480
converged
```

`opt$par` tells us the value of the parameter found to maximise the likelihood, i.e. $\hat{\theta} = (\hat{\mu}, \hat{\beta})$, and `opt$value` the maximum of the log-likelihood at this point, that is $\ell(\hat{\theta}; \mathbf{x})$

```
> print(opt)

$par
       mu      beta
3.1950358 0.4508507

$value
[1] -27.19248

$counts
function gradient
      22       22

$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"

$hessian
             mu       beta
mu   -147.58969   43.82941
beta   43.82941 -205.04584
```

The Hessian can be recovered from `opt$hessian` and one can check numerically whether −`opt$hessian` is positive definite by computing its eigenvalues,

```
> eigen(-opt$hessian)

$values
[1] 228.7231 123.9124
```

```
$vectors
           [,1]       [,2]
[1,] -0.4752947 -0.8798266
[2,]  0.8798266 -0.4752947
```

which are indeed both positive. It therefore seems that we have found a value close to a local maximum of the log-likelihood. At this point we can only hope that it is a global maximum–we could however try several initial points `init` in order to check that we find the same maximum.
Let's look at the estimated distribution function, $\mathbb{P}\left(X \leq u; \hat{\mu}, \hat{\beta}\right)$ as a function of u, which we are going to compare to the non-parametric estimate of the distribution function

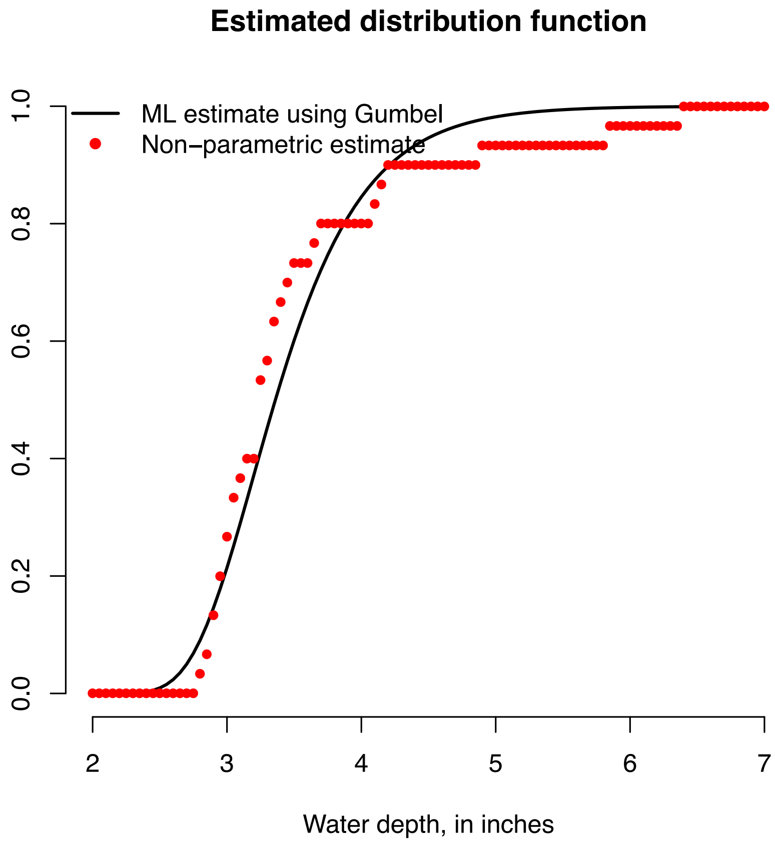
$$\hat{F}_{NP}(u) := \frac{1}{30}\sum_{i=1}^{30} \mathbb{I}\{\texttt{extremeRainfall[i]} \leq u\} \quad .$$

First, define the distribution function:

```
> pgumbel <- function(x, mu, beta) {
+     stopifnot(all(beta > 0))
+     exp(-exp(-(x - mu)/beta))
+ }
```

which we now plot with the ML estimates as parameters

```
> xvals <- seq(from = 2, to = 7, length = 101)
> plot(xvals, pgumbel(xvals, mu = opt$par[1], beta = opt$par[2]),
+     type = "l", lwd = 2, main = "Estimated distribution function",
+     xlab = "Water depth, in inches", ylab = "", bty = "n")
> FhatNP <- function(x) {
+     sapply(x, function(xv) mean(extremeRainfall <= xv))
+ }
> points(xvals, FhatNP(xvals), col = "red", pch = 16, cex = 0.85)
> legend("topleft", legend = c("ML estimate using Gumbel",
+      "Non-parametric estimate"), lty = c(1, 0), lwd = 2,
+      pch = c(NA, 16), col = c("black", "red"), bty = "n")
```



which seems to be a good fit.