6 R session: Multivariate extremes and Bayesian inference

As before, you will need to start R, and then attach the libraries `ismev` and `evd`, and the supplementary R routines we have provided, using the commands:

```
> library(ismev)
> library(evd)
> source('Rstufflee.r')
```

1. In this question we carry out a simple bivariate analysis using the block–maxima approach. The dataset `wind` has 40 rows and 3 columns; the second and third columns contain annual maximum wind speeds at Albany, New York and Hartford, Connecticut (respectively) over the period 1944 to 1983.

(a) Load the data into `R` using:

```
> data(wind)
```

and have a look at it by typing

```
> wind
```

(b) The data we want are the annual maxima for Hartford and Albany respectively, stored in columns 2 and 3. We extract them using

```
> hartford<-wind[,2]
> albany<-wind[,3]
```

and then recombine them into a vector of bivariate annual maxima using

```
> blockmax<-cbind(hartford,albany)
```

(c) We can now fit a bivariate extreme value distribution using the logistic model:

```
> fbvevd(blockmax)
```

since the logistic model is the default. You may like to experiment with other models, e.g.

```
> fbvevd(blockmax,model="bilog",std.err = FALSE)
```

although note that this model is too complex to calculate standard errors, hence the need to switch this facility off (to avoid an error!). You may like to experiment with other models.

(d) If we want to produce diagnostic plots we must first create an object containing the fits, e.g.

```
> fit1<-fbvevd(blockmax)
```

and then run the plots command

```
> plot.bvevd(fit1)
```

You may like to think about what these plots are telling us, and investigate how well different models fit these data.

2. The data set `wavesurge` contains the data on which the bivariate example in Section 4.3 was based. The data has 2894 rows and 2 columns; corresponding to the wave height and sea surge in consecutive measurements taken at Newlyn, Cornwall, between 1971 and 1977.

(a) Load the data into `R` using:

```
> data(wavesurge)
```

Now separate wave and surge using:

```
> wave<-wavesurge[,1]
> surge<-wavesurge[,2]
```

(b) You can check this has worked by plotting surge against wave height using:

```
> plot(wave,surge)
```

At this stage it would be possible to carry out univariate threshold–based analyses of each of `wave` and `surge` separately, and you may like to do this in your own time. However we will proceed directly to a bivariate analysis in the exercises below.

(c) We will first identify appropriate thresholds for the analysis. We decide to identify the empirical 95% quantile in each margin, and we can do this using:

```
> quantile(wave,0.95)
> quantile(surge,0.95)
```

We can now create an appropriate bivariate threshold vector, e.g. using:

```
> thresh<-c(6.080,0.322)
```

(d) We are now in a position to fit various bivariate models to the bivariate object `wavesurge`:

```
> fbvpot(wavesurge,thresh)
```

fits the logistic model (the default). Check you understand all of the output, including identifying the relevant model parameters.

(e) To fit the bilogistic model, use:

```
> fbvpot(wavesurge,thresh,model="bilog")
```

You may like to experiment with other models.

(f) For any particular model fit, we can explore the model fit, and various aspects of the inference, using the graphical routine `plot.bvpot()` applied to an object generated from a fit. For example, to investigate the fitted logistic model, use:

```
> fitlogistic<-fbvpot(wavesurge,thresh)
> plot.bvpot(fitlogistic)
```

3. We return to the dataset `wind` containing annual maximum wind speeds at Albany, New York and Hartford, Connecticut over the period 1944 to 1983. The first column gives corresponding years. The data set should already be in R, but if you have not done Question 1 in this Rsession, reload it using:

```
> data(wind)
```

Now separate the two sets of wind speeds using

`> albany<-wind[,2]` and
`> hartford<-wind[,3]`
The function

```
> gev.bayes(n,dataset,mustart,sigmastart,xistart, ...
... errmu,errlogsigma,errxi,sdmu,sdlogsigma,sdxi)
```

produces (approximate) draws from the posterior distribution $\pi(\mu, \sigma, \xi | \underline{y})$, where μ, σ and ξ are the location, scale and shape parameters of the GEV distribution and $\underline{y} = (y_1, y_2, \ldots, y_{40})$ are the annual wind speed maxima in years 1944, 1945, . . ., 1983. This routine uses Metropolis–Hastings sampling with a random walk update scheme for each of the parameters. As in the notes, independent Normal priors are used for μ, $\log(\sigma)$ and ξ.

The arguments in the function are defined as follows:

`n`	The number of iterations in the Metropolis–Hastings sampler
`dataset`	A single vector containing the data
`mustart`	The starting value for μ in the chain
`sigmastart`	The starting value for σ in the chain
`xistart`	The starting value for ξ in the chain
`errmu`	The random walk innovation variance for μ
`errlogsigma`	The random walk innovation variance for $\log(\sigma)$
`errxi`	The random walk innovation variance for ξ
`sdmu`	The Normal distribution prior standard deviation for μ
`sdlogsigma`	The Normal distribution prior standard deviation for $\log(\sigma)$
`sdxi`	The Normal distribution prior standard deviation for ξ

(a) Run the Metropolis–Hastings sampler for the wind speed maxima observed at Albany, NY, for 10,000 iterations, using

- $(\mu^{(0)}, \sigma^{(0)}, \xi^{(0)}) = (20, 15, 0.1)$;
- $v_\mu = v_{\log(\sigma)} = v_\xi = 0.1$;
- Large Normal prior standard deviations for μ, $\log(\sigma)$ and ξ – 10000, 10000, 100 (respectively).

Make sure you store your results somewhere, e.g. use

```
> mcmc.results1<-gev.bayes( ...
```

and ignore the `warning` message that R returns. Then `mcmc.results1` will store the 10,000 draws from the posteriors of μ, $\log(\sigma)$ and ξ, as well as the corresponding acceptance probabilities – these can be accessed by typing, for example,

```
> mcmc.results1$mu
```

(b) Now examine your output using

```
> par(mfrow=c(3,1))
> plot(ts(mcmc.results1$mu))
> plot(ts(mcmc.results1$logsigma))
> plot(ts(mcmc.results1$xi))
```

(You may want to edit the labels for the axes as we did in Part 3 of this course, using, for example, `xlab='iteration'`.) Do you think your sampler is performing well? Does it converge? If so, what is the 'burn–in' period?

(c) Remember, an overall acceptance probability for each parameter of between 30%–50% is usually good enough. Look at your acceptance probabilities for μ, $\log(\sigma)$ and ξ by typing, for example

```
> mean(mcmc.results1$aprobmu)
```

Do you think your sampler is performing well?

(d) Now run the sampler again (maybe store your results in `mcmc.results2`) but choose more appropriate starting values based on your plots in part (b) and change the variances of your random walk innovations if necessary (if you increase `errmu`, `errlogsigma` or `errxi` the corresponding acceptance probabilities will decrease). Examine your output as you did in parts (b) and (c) and check for improvement.

(e) Once you are satisfied with your MCMC, you should summarise your posteriors (after the removal of burn–in). Typing

```
> mu.burn<-mcmc.results2$mu[2000:10000]
```

would, for example, discard the first 2000 iterations as 'burn–in' and store the remainder of the posterior draws for μ in the vector `mu.burn`. After identifying an appropriate burn–in period for *your* MCMC output, use commands similar to that above to obtain vectors containing posterior draws for μ, $\log(\sigma)$ and ξ after the removal of burn–in (and store them in `mu.burn`, `logsigma.burn` and `xi.burn`).

(f) We will now look at the posterior densities of our MCMC draws for μ, σ and ξ. Type

```
> par(mfrow=c(2,2))
> plot(density(mu.burn))
> plot(density(exp(logsigma.burn)))
> plot(density(xi.burn))
```

to produce density plots of the posterior draws for the parameters μ, σ and ξ (note the transformation back to σ by exponentiation of the log(σ) vector).

(g) Find the posterior mean and standard deviation for each of the three GEV parameters by typing, for example,

`> mean(mu.burn)` and
`> sd(mu.burn)`

(h) Now we can obtain the posterior distribution for, say, the 1000–year return level by using the function `ret.level.gev` on each of the draws for μ, σ and ξ. We can do this by typing:

```
> retlevel<-vector('numeric', length(mu.burn))
> for(i in 1:length(retlevel))
+ {
+ retlevel[i]<-ret.level.gev(mu.burn[i],...
...exp(logsigma.burn[i]),xi.burn[i],1000)
+ }
```

Now typing

```
plot(density(retlevel))
```

will add a density plot of the posterior for the 1000–year return level to your panel of plots produced in part (f). Numerical summaries can be obtained in a similar fashion to (g), though owing to the (often) severe asymmetry of the posterior surface for return levels, you may want to use `median()` and not `mean()` as a summary of posterior location here.

(i) Now find maximum likelihood estimates for μ, σ, ξ and the 1000–year return level (see Part 3) and compare these with the results from your Bayesian analysis (compare m.l.e.s with posterior means, for example, and estimated standard errors with posterior standard deviations).

(j) If you have time, and are interested in this stuff, you could re–run this type of analysis on the Hartford data.

Acknowledgement
Lee Fawcett and Dave Walshaw would like to thank Jill Johnson for her help in producing some of the R routines and plots used in this short course.

References

Bortot, P., Coles, S. and Tawn, J. (2000). The Multivariate Gaussian tail model: an application to oceanographic data. *Applied Statistics*, **49**, 1, 31—50.

Coles, S.G. (2001). *An introduction to statistical modeling of extreme values*. Springer, London.

Coles, S.G. (2001b). A Random Effects Analysis of Extreme Wind Speed Data. Preprint.

Davison, A.C. and Smith, R.L. (1990). Models for Exceedances over High Thresholds (with discussion). *J. R. Statist. Soc., B*, **52**, 393—442.

Fawcett, L. (2005). Statistical Methodology for the Estimation of Environmental Extremes. PhD Thesis, University of Newcastle-upon-Tyne.

Fawcett, L. and Walshaw, D. (2006). A Hierarchical Model for Extreme Wind Speeds. *Applied Statistics*, **55**, 5, 631—646.

Fawcett, L. and Walshaw, D. (2007). Improved Estimation for Temporally Clustered Extremes. *Environmetrics*, **18**, 2, pp. 173-188

Fisher, R.A. and Tippett, L.H.C. (1928). On the estimation of the frequency distributions of the largest or smallest member of a sample. *Proc. Camb. Phil. Soc.*, **24**, 180—190.

Heffernan, J.E. and Tawn, J.A. (2004). A conditional approach for multivariate extreme values. *J. R. Statist. Soc., B, Part 2*, **66**, 1—34.

Kotz, S. and Nadarajah, S. (2000). *Extreme Value Distributions: Theory and Applications*. Imperial College Press, London.

Leadbetter, M.R., Lindgren, G. and Rootzén, H. (1983). *Extremes and Related Properties of Random Sequences and Series*. Springer–Verlag, New York.

Smith, R.L. (1989). Extreme Value Analysis of Environmental Time Series: An Application to Trend Detection in Ground–Level Ozone. *Statistical Science*, **4**, 367—393.

Smith, R.L. (1991). Regional Estimation from Spatially Dependent Data. Preprint.

Walshaw, D. (1991). Statistical Analysis of Extreme Wind Speeds. *PhD Thesis*. University of Sheffield, Sheffield.

Walshaw, D. (1994). Getting the Most From Your Extreme Wind Data: A Step by Step Guide. *J. Res. Natl. Inst. Stand. Technol.*, **99**, 399—411.