WAFO Chapter 5

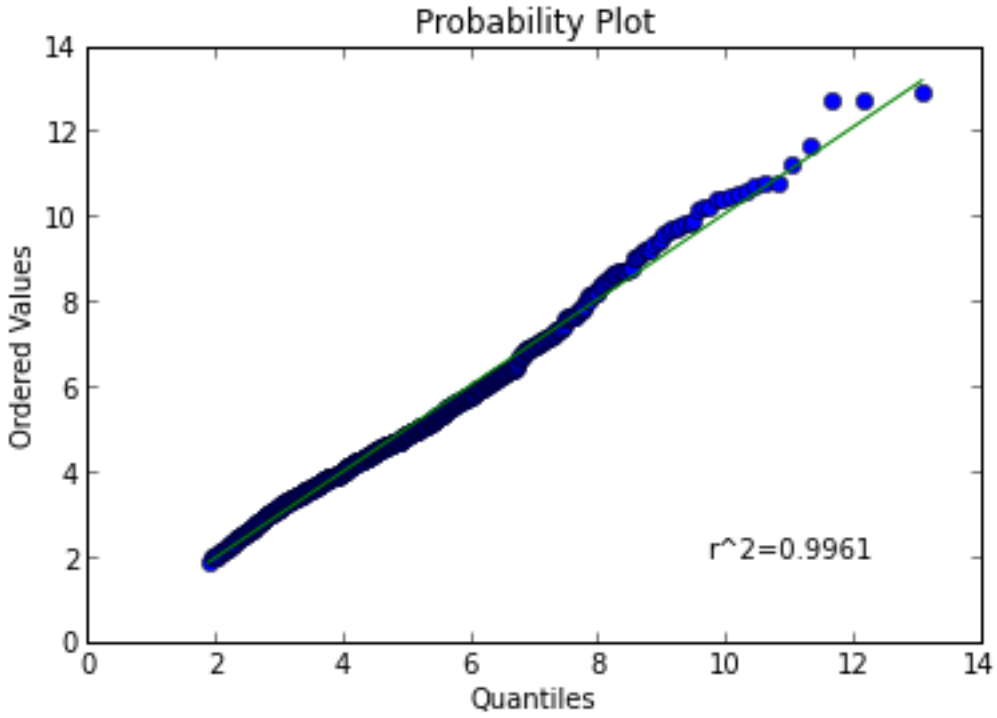
November 26, 2014

1 Chapter 5 Extreme value analysis

1.1 Section 5.1 Weibull and Gumbel papers

Significant wave-height data on Weibull paper,

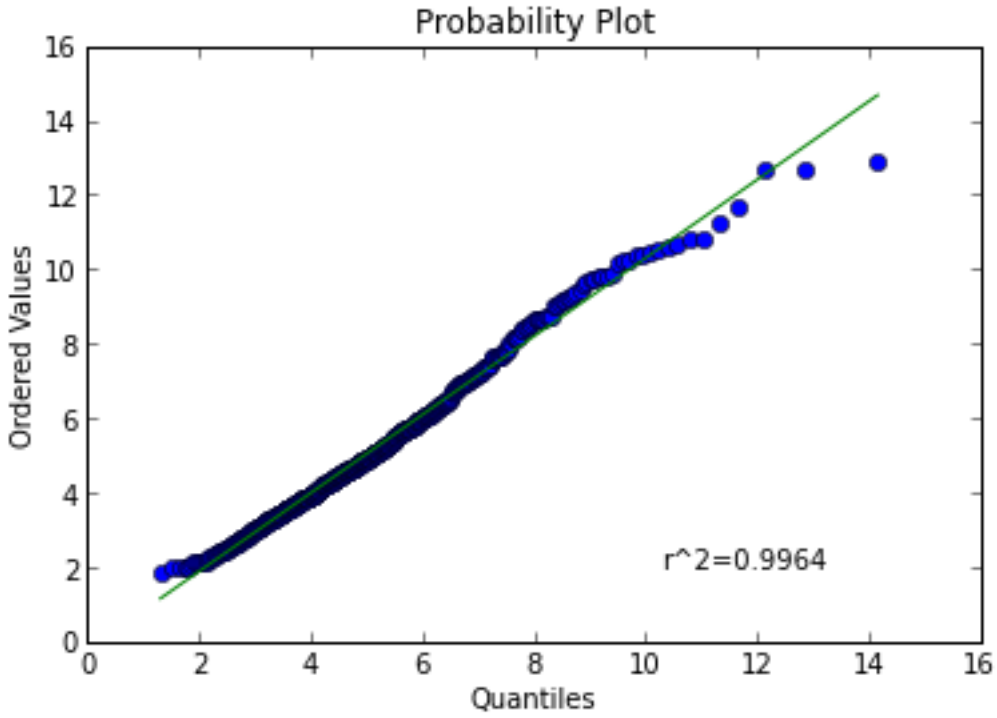
```
In [3]: clf()
        import wafo.data as wd
        import wafo.stats as ws
        import matplotlib.pyplot as plt
        Hs = wd.atlantic()
        wei = ws.weibull_min.fit2(Hs)
        tmp = ws.probplot(Hs, wei.par, dist='weibull_min', plot=plt)
```



Significant wave-height data on Gumbel paper,

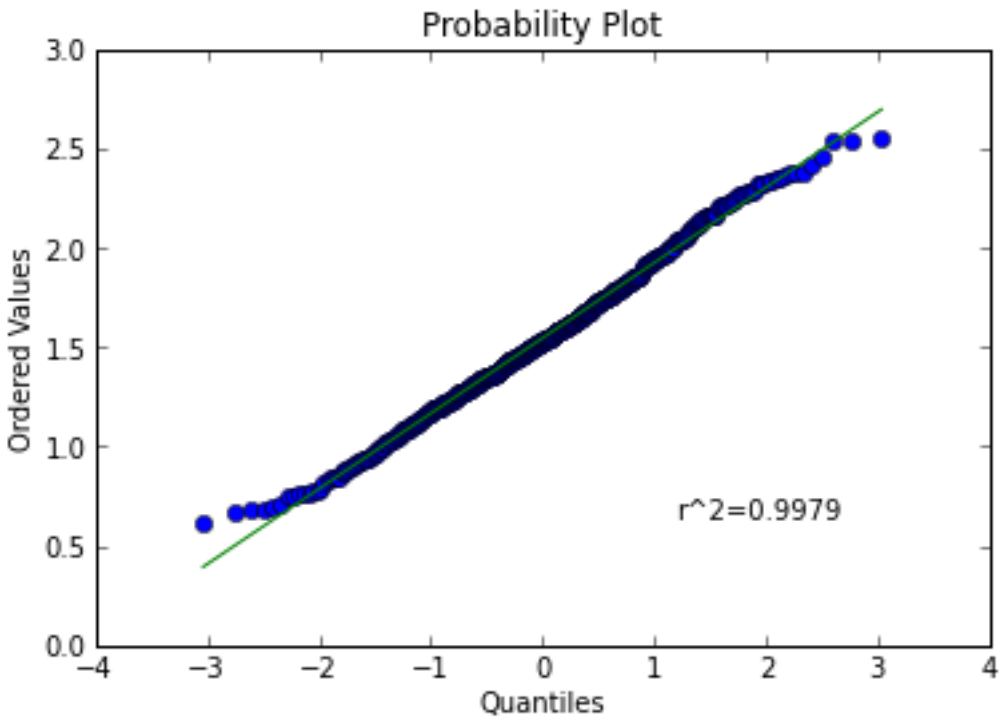
```
In [4]: gum = ws.gumbel_r.fit2(Hs)
        tmp = ws.probplot(Hs, gum.par, dist='gumbel_r', plot=plt)
```

```
c:\pab\workspace\pywafo_svn\pywafo\src\wafo\stats\estimation.py:722: RuntimeWarning: invalid value encou
  self.par_lower = self.par - zcrit * sqrt(pvar)
c:\pab\workspace\pywafo_svn\pywafo\src\wafo\stats\estimation.py:723: RuntimeWarning: invalid value encou
  self.par_upper = self.par + zcrit * sqrt(pvar)
```



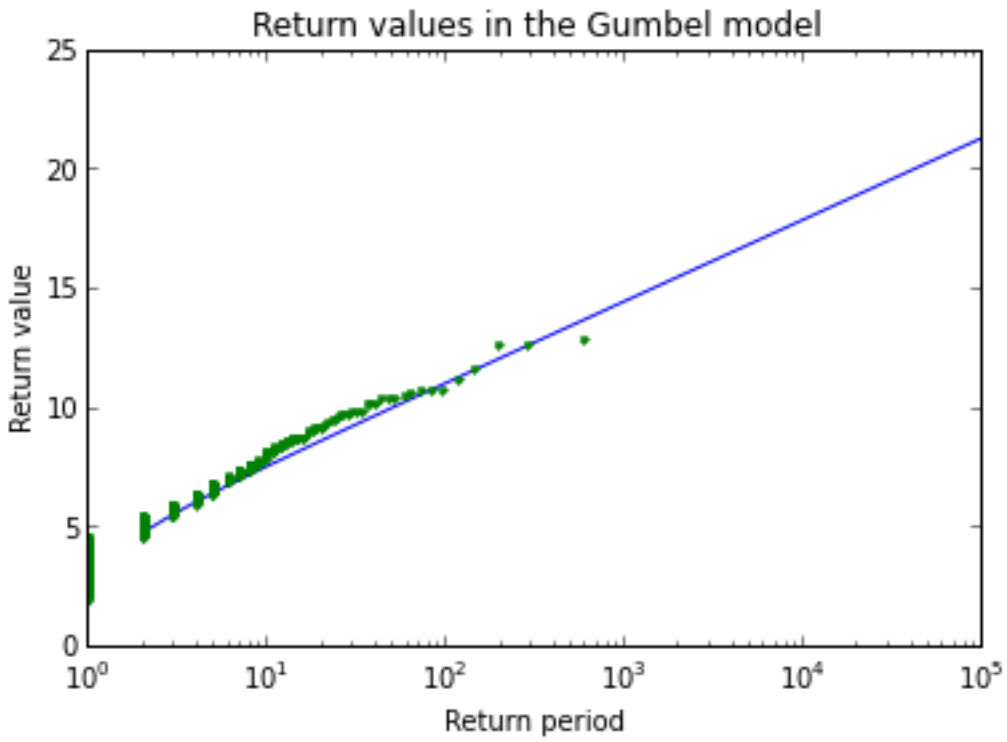
Significant wave-height data on Normal probability paper,

```
In [5]: tmp = ws.probplot(np.log(Hs), plot=plt)
```



Return values in the Gumbel distribution

```
In [6]: clf()
        T=np.r_[1:100001]
        #sT=gum.par[1] - gum.par[0]*log(-log(1-1./T));
        sT = gum.isf(1./T)
        semilogx(T,sT), hold
        N=np.r_[1:len(Hs)+1];
        Nmax=max(N);
        plot(Nmax/N,sort(Hs)[::-1],'.')
        title('Return values in the Gumbel model')
        xlabel('Return period')
        ylabel('Return value')

Out[6]: <matplotlib.text.Text at 0x6d23570>
```

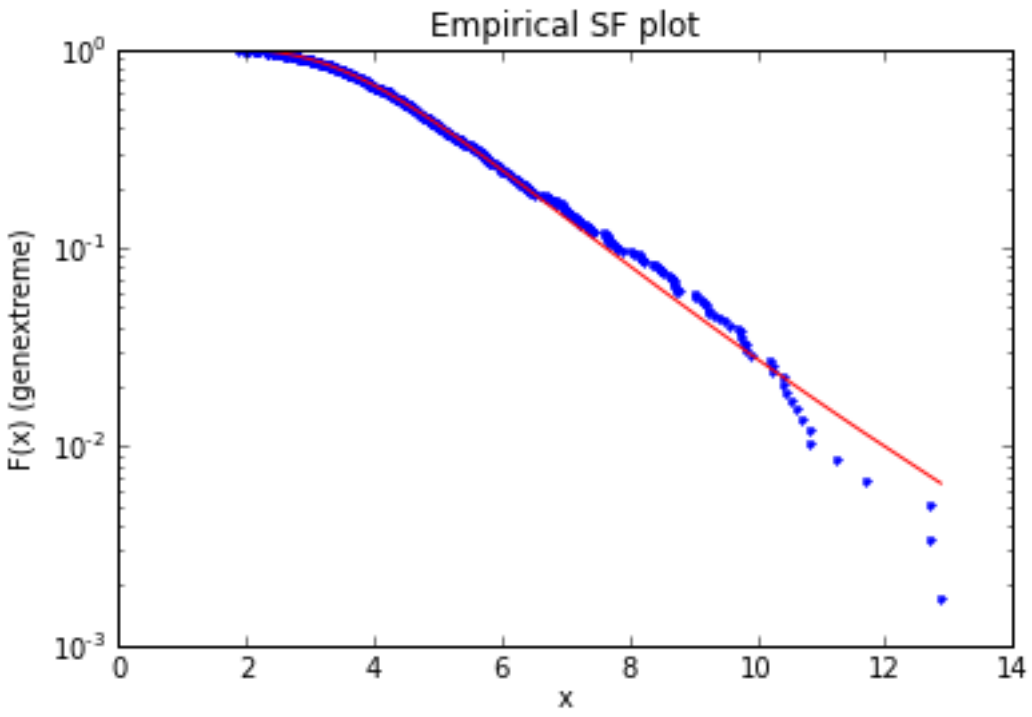


1.2 Section 5.2 Generalized Pareto and Extreme Value distributions

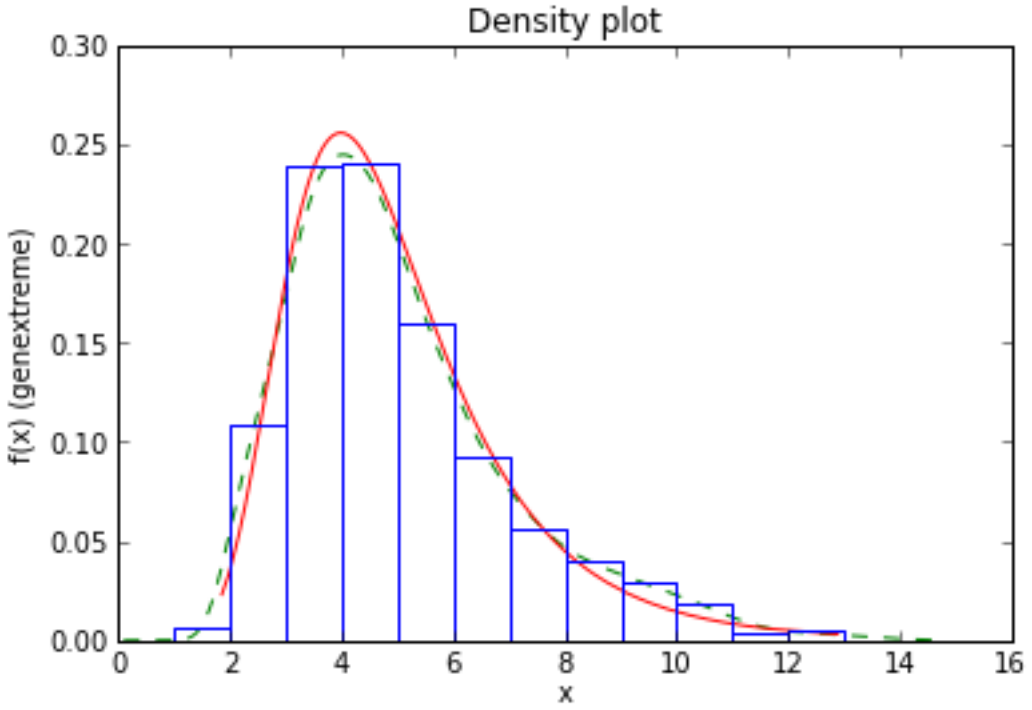
1.3 Section 5.2.1 Generalized Extreme Value distribution

Empirical distribution of significant wave-height with estimated Generalized Extreme Value distribution

```
In [7]: gev = ws.genextreme.fit2(Hs)
        gev.plotesf()
```

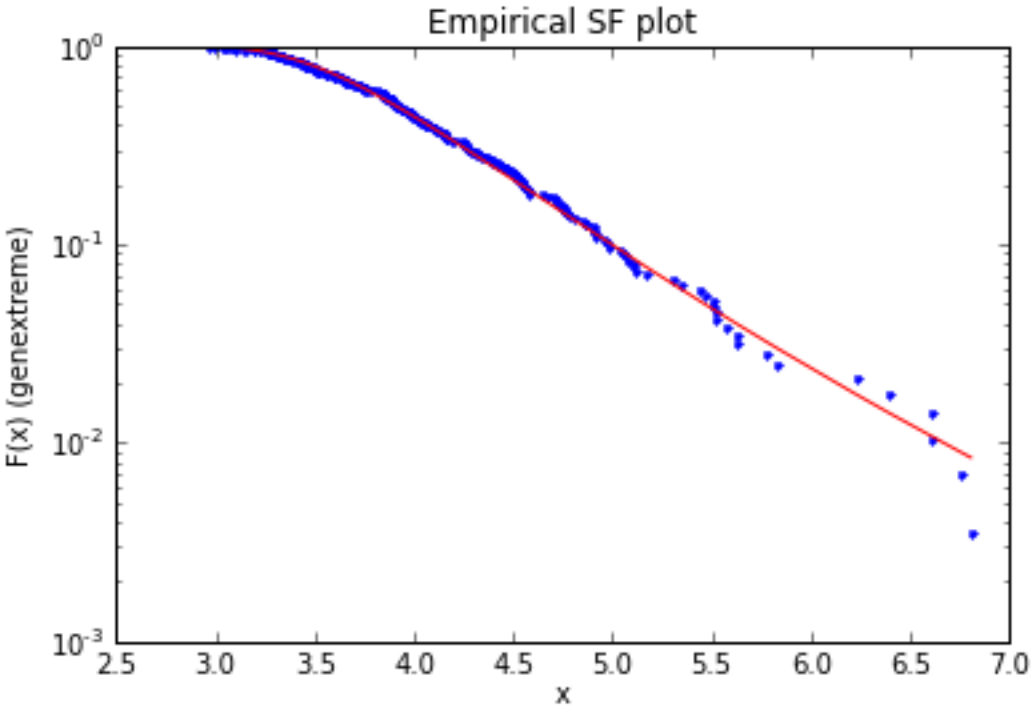



```
In [8]: import wafo.kdetools as wk
        wk.TKDE(Hs, L2=0.5)(output='plot').plot('g--')
        plt.hold(True)
        gev.plotepdf()
```



Analysis of yura87 wave data. Wave data interpolated (spline) and organized in 5-minute intervals Normalized to mean 0 and std = 1 to get stationary conditions. maximum level over each 5-minute interval analysed by GEV

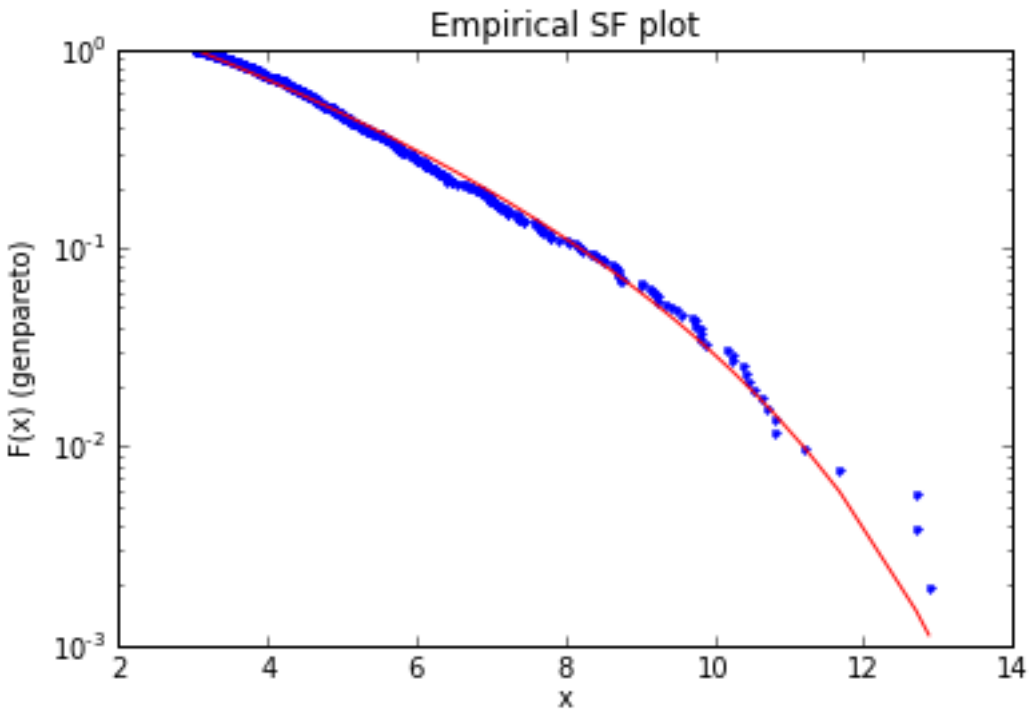
```
In [9]: import scipy.interpolate as si
        xn  = wd.yura87()
        XI  = np.r_[0:len(xn):0.25]
        N   = len(XI);
        N = N-np.mod(N,4*60*5);
        YI  = si.UnivariateSpline(xn[:,0].ravel(),xn[:,1].ravel(),k=3,s=0)(XI[:N])
        YI  = np.reshape(YI, (4*60*5, N/(4*60*5))); # Each column holds 5 minutes of interpolated data.
        Y5  = (YI-YI.mean(axis=0))/(YI.std(axis=0))
        Y5M = Y5.max(axis=0)
        Y5gev = ws.genextreme.fit2(Y5M,method='mps')
        Y5gev.plotesf()
```



1.4 Section 5.2.2 Generalized Pareto distribution

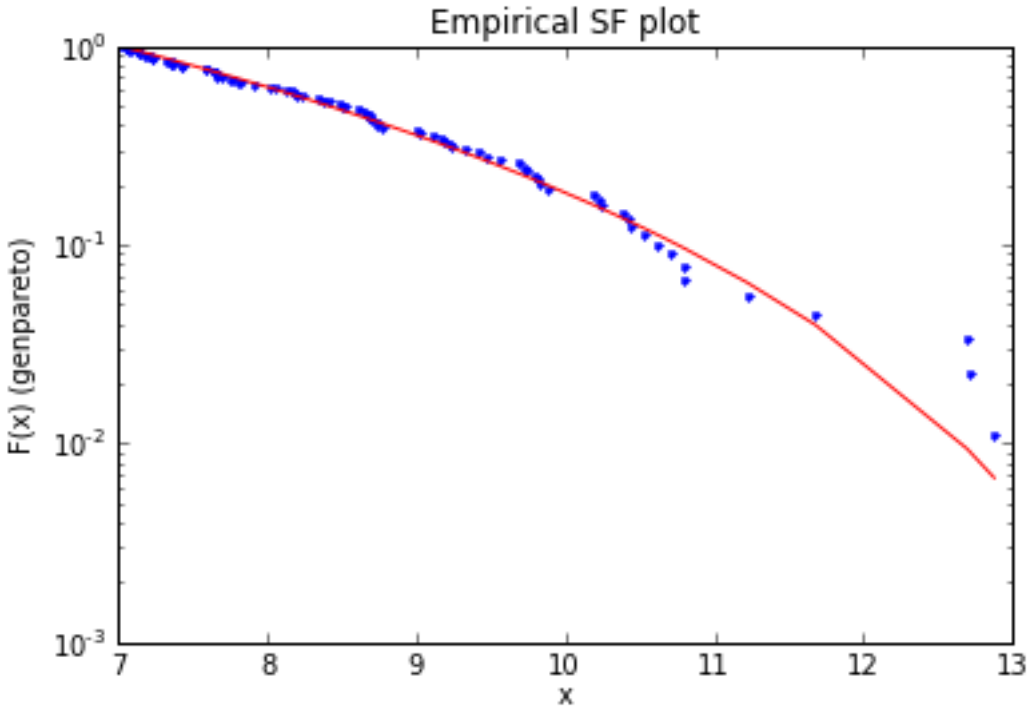
Exceedances of significant wave-height data over level 3.

```
In [10]: gpd3 = ws.genpareto.fit2(Hs[Hs>3],floc=3)
         gpd3.plotesf()
```



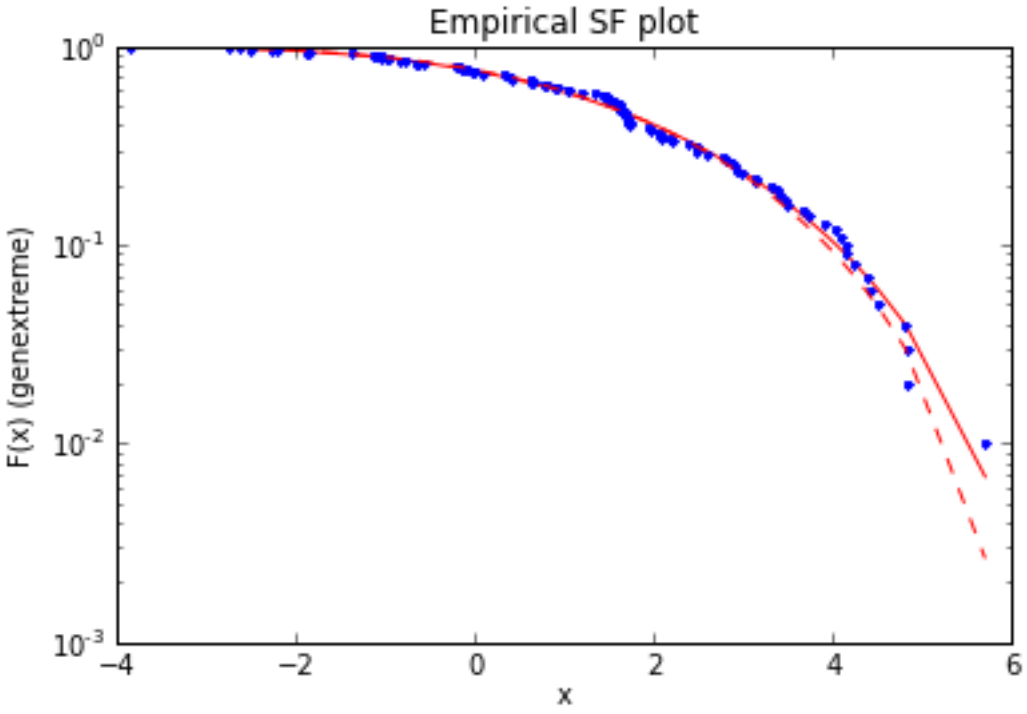
Exceedances of significant wave-height data over level 7,

```
In [11]: gpd7 = ws.genpareto.fit2(Hs[Hs>7],floc=7)
         gpd7.plotesf()
```



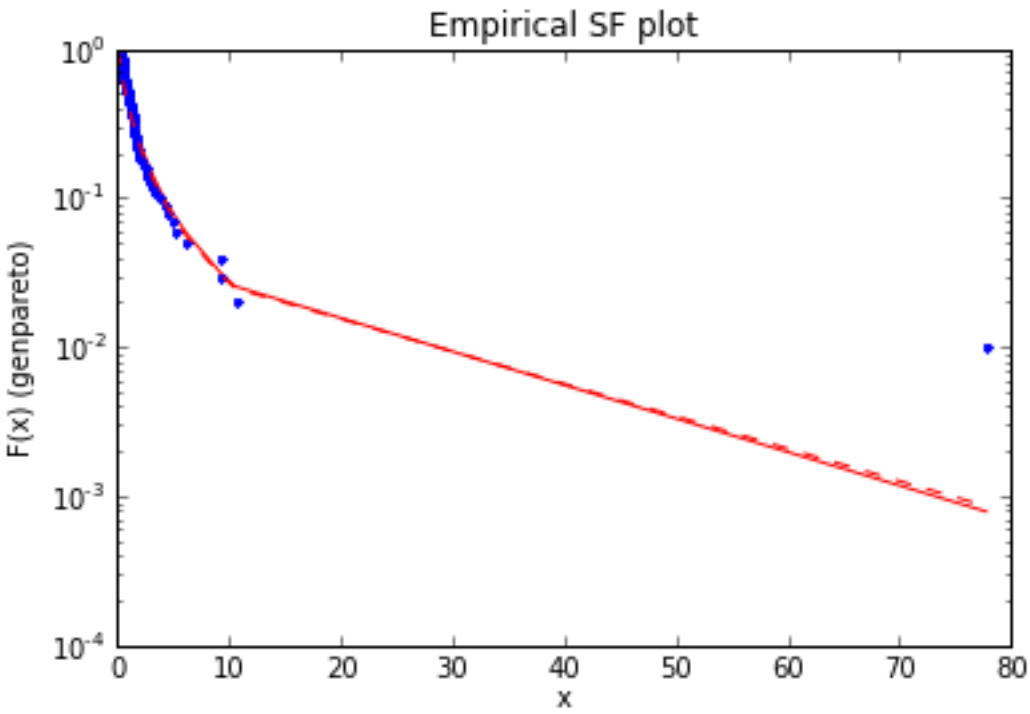
Simulates 100 values from the GEV distribution with parameters (0.3, 1, 2), then estimates the parameters using two different methods and plots the estimated distribution functions together with the empirical distribution.

```
In [17]: Rgev = ws.genextreme.rvs(0.3,1,2,size=100)
         gp = ws.genextreme.fit2(Rgev,method='mps');
         gm = ws.genextreme.fit2(Rgev,method='ml');

         gp.plotesf()
         plt.hold(True)
         gm.plotesf('r--')
```



Similarly for the GPD distribution

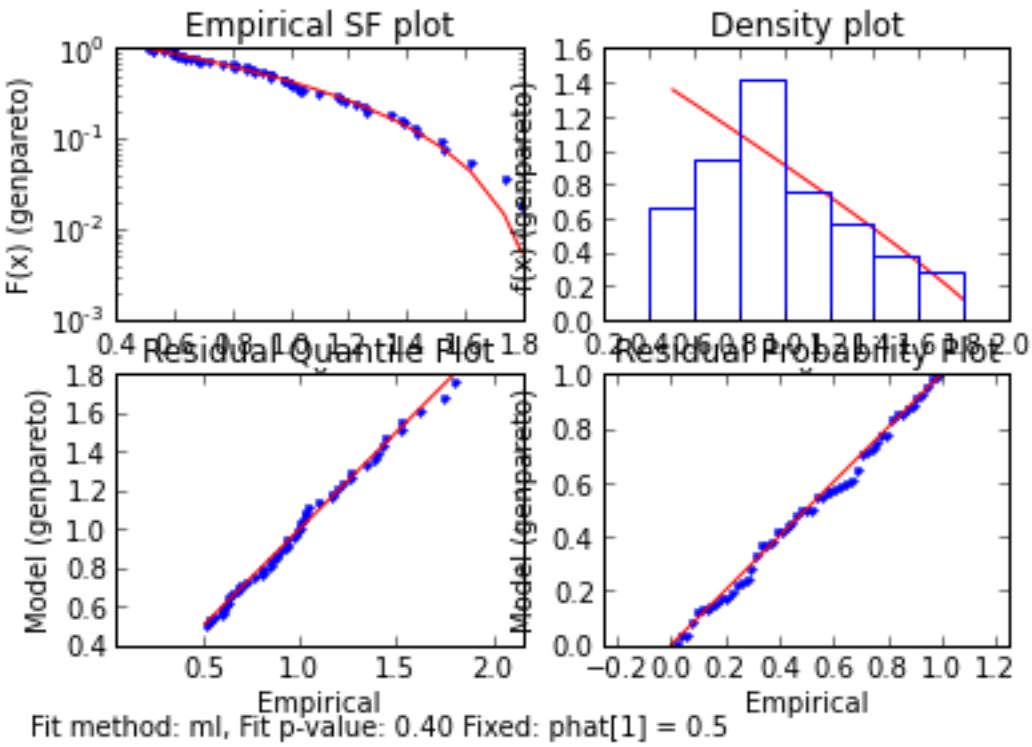
```
In [18]: Rgpd = ws.genpareto.rvs(0.4,size=100);
         gmps = ws.genpareto.fit2(Rgpd, method='mps')
         gml = ws.genpareto.fit2(Rgpd, method='ml')
         gmps.plotesf()
         plt.hold(True)
         gml.plotesf('r--')
```



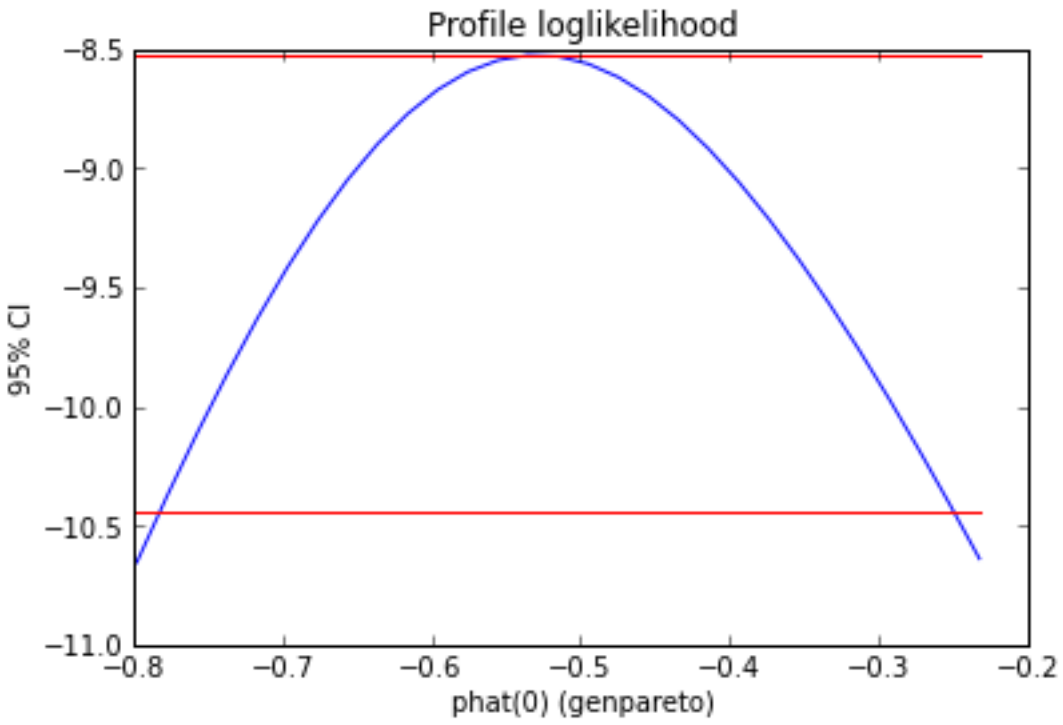
Return values for the GEV distribution

```
In []: T = logspace(1,5,10);
       sT = Y5gev.isf(1./T);


       clf
       semilogx(T,sT,T,sTlo,'r',T,sTup,'r'), hold
       N=1:length(Y5M); Nmax=max(N);
       plot(Nmax./N,sort(Y5M,'descend'),'.')
       title('Return values in the GEV model')
       xlabel('Return priod')
       ylabel('Return value')
       grid on
```

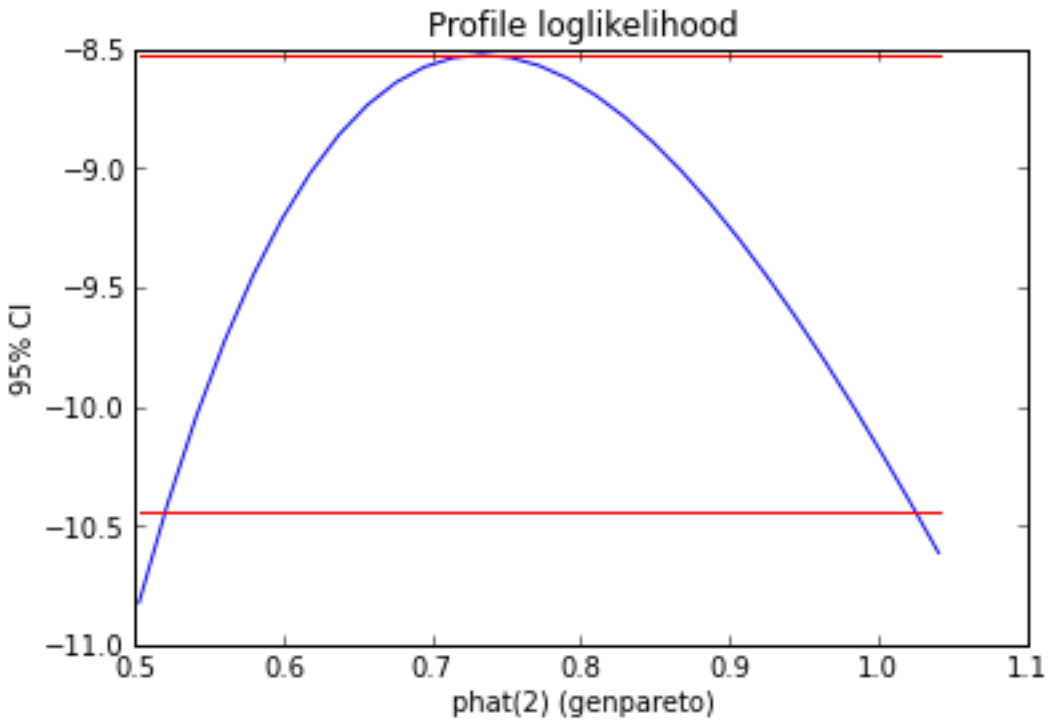
```
In [13]: import wafo.stats as ws
         R = ws.genpareto.rvs(-0.5,size=100);
         phat = ws.genpareto.fit2(R[R>.5], -.5, scale=1, floc=0.5)
         phat.plotfitsummary()
```




```
In [14]: # Better CI for phat.par[i=0] shape parameter
         Lp0 = phat.profile(i=0, pmin=-1,pmax=1)
         Lp0.plot()
         phat0_ci = Lp0.get_bounds(alpha=0.1)
         print 'phat0_ci = ', phat0_ci

phat0_ci =  [-0.73845586 -0.30183734]
```



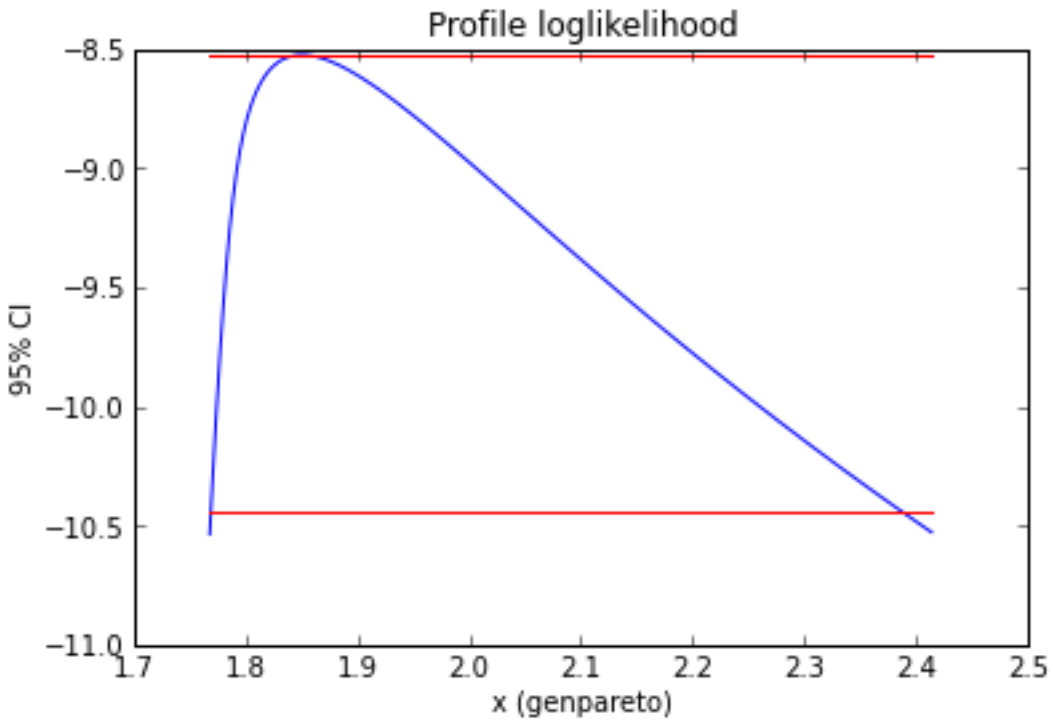

```
In [15]: # Better CI for phat.par[i=2] scale
         Lp2 = phat.profile(i=2,pmin=0.1,pmax=2)
         Lp2.plot()
         phat2_ci = Lp2.get_bounds(alpha=0.1)
         print 'phat2_ci = ', phat2_ci

phat2_ci =  [ 0.55127823  0.97075832]
```




```
In [16]: SF = 1./990
         x = phat.isf(SF)

         # CI for x
         Lx = phat.profile(i=2, x=x, link=phat.dist.link)
         Lx.plot()
         x_ci = Lx.get_bounds(alpha=0.2)
         print 'X_c = ', x_ci
```

```
X_c =  [ 1.78350616  2.09079614]
```

```
c:\pab\workspace\pywafo_svn\pywafo\src\wafo\stats\distributions.py:4011: RuntimeWarning: invalid value e
  return where((c != 0) & (-inf < log_sf), expm1(-c * log_sf) / c, -log_sf)
```


Profile loglikelihood
−8.5
−9.0
−9.5
−10.0
−10.5
−11.0
95% CI
1.7
1.8
1.9
2.0
2.1
2.2
2.3
2.4
2.5
x (genpareto)

In [16]: