ps3_6_3

December 1, 2022

0.0.1 Preliminaries

[ ]:
```
using Plots, Statistics, Distributions, Optim, Random
Random.seed!(2359112412)
```

[ ]:
```
TaskLocalRNG()
```

1 Exercise 6.3

1.1 (a) Inverse Transform Sampling

First we derive the inverse CDF of the Pareto distribution to implement the sampling procedure.

With the PDF given we can get the CDF by integration and using the condition that $F_X(1) = 0$ (note 1 is the lower bound of the support).

$$F_X(x) = C + -x^{-\alpha}$$

so looking at $F_X(1)$ gives C = 1.

The quantile function is then given by

$$F_X^{-1}(p) = (1-p)^{(-1/\alpha)}$$

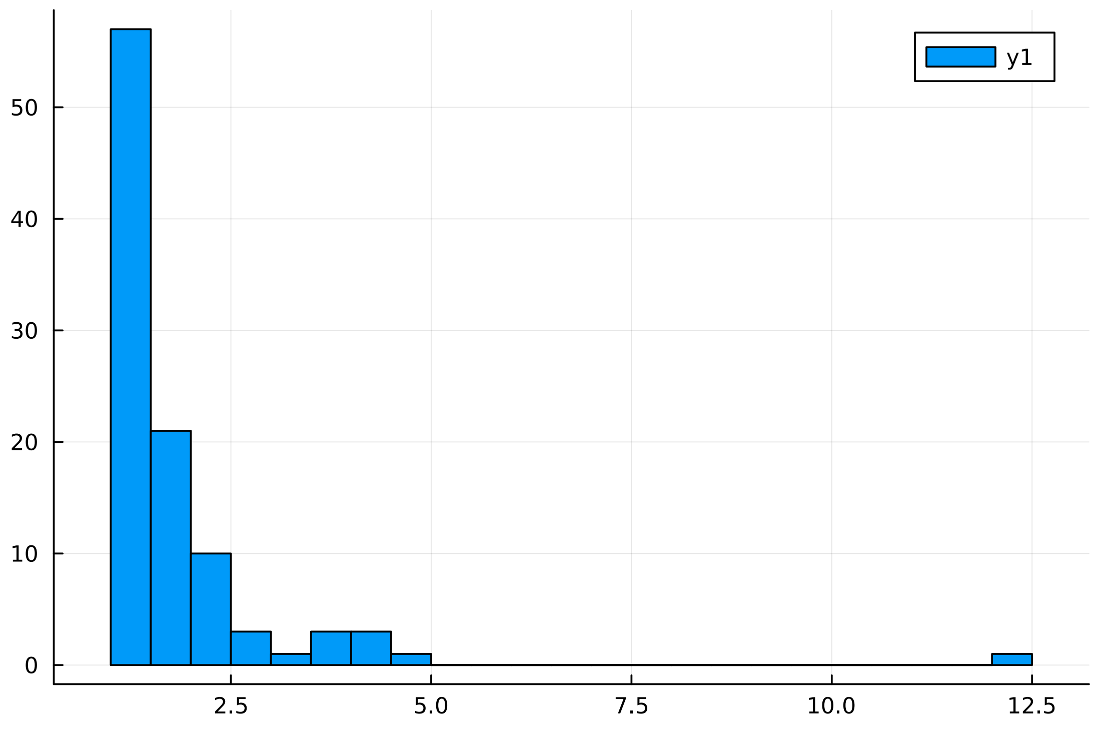
for $p \in [0, 1]$.

[ ]:
```
# Sampling Function
inv_cdf(p,  ) = (1-p)^(-(1/ ))

function sample_pareto( , n)
    x = rand(Uniform(0,1), n)
    return inv_cdf.(x,  )
end

sample2 = sample_pareto2(2, 100)
histogram(sample2)
```

[ ]:



1.2 (b) Maximum Likelihood Estimation

We first derive the log-likelihood function:

$$L_n(\alpha) = \prod_{i=1}^{N} \frac{\alpha}{X_i^{\alpha+1}}$$

which follows from the sample being iid. Taking logs and collecting terms we get the log-likelihood function

$$l_n(\alpha) = n\ln(\alpha) - \sum_{i=1}^{N}(\alpha+1)\ln(X_i).$$

To find the maximum-likelihood estimator $\hat{\alpha}_{mle}$, we maximize the above function with respect to α.

Setting the first order condition to zero and rearranging terms we get the following estimator:

$$\hat{\alpha}_{mle} = \frac{n}{\sum_{i=1}^{N}\ln(X_i)}.$$

Next we solve the problem numerically by coding the (negative) log-likelihood and using an optimization routine to find the minimum.

```
[ ]: # Define relevant functions (neg-log-likelihood, mle, simulation)
     mle_analytic(x) = length(x) / sum(log.(x))
```

```julia
negll( , x) = -(length(x) * log( ) - sum(( +1).*log.(x)))

function mle(x; lb=1, ub=10)
    f( ) = negll( , x)
    return optimize(f, lb, ub).minimizer
end

function simulate_mle(reps, n;  =2)
    out = zeros(reps, 2)

    for i in 1:reps
        sample = sample_pareto( , n)
        out[i, 1] = mle(sample)
        out[i, 2] = mle_analytic(sample)
    end

    return out
end
```

[ ]:
```
simulate_mle (generic function with 1 method)
```

The below histogram plots the simulated distribution of $\hat{\alpha}_{mle}$ for 10,000 simulation runs and a sample size of $N = 100$ and $N = 1000$. We can see that for the larger sample the asymptotic distribution of the MLE (from class) seems to be a better approximation as expected.

[ ]:
```julia
results_50 = simulate_mle(10000, 50);
results_100 = simulate_mle(10000, 100);
results_1000 = simulate_mle(10000, 1000);
```

[ ]:
```julia
# Asymptotic variance of the MLE
A(n,  ) = n/( ^2)
# B =
avar(n,  ) = 1/A(n,  )
# avar2 = B/A^2 # not sure regularity conditions hold (e.g no variance)
```

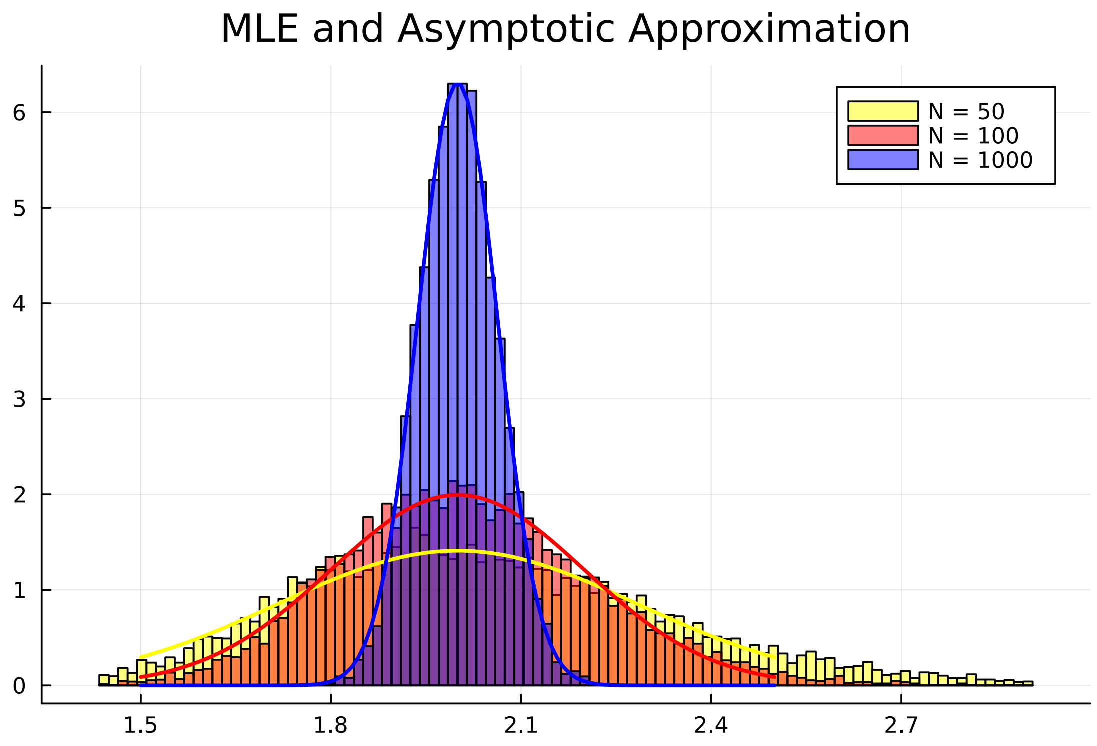
[ ]:
```
avar (generic function with 1 method)
```

[ ]:
```julia
bins = range(minimum(results_100[:,1]), stop = maximum(results_100[:,1]),␣
 ↪length = 100)

histogram(results_50[:,1], color=:yellow, normalize=:pdf, fillalpha = 0.5, bins␣
 ↪= bins, label = "N = 50")
histogram!(results_100[:,1], color=:red, normalize=:pdf, fillalpha = 0.5, bins␣
 ↪= bins, label = "N = 100")
histogram!(results_1000[:,1], color=:blue, normalize=:pdf, fillalpha = 0.5,␣
 ↪bins = bins, label = "N = 1000")
```

```
grid = range(1.5, 2.5, 100)
plot!(grid, pdf.(Normal(2, sqrt(avar(50, 2))), grid), color =:yellow, lw = 2,␣
  ↪label =:none)
plot!(grid, pdf.(Normal(2, sqrt(avar(100, 2))), grid), color =:red, lw = 2,␣
  ↪label =:none)
plot!(grid, pdf.(Normal(2, sqrt(avar(1000, 2))), grid), color =:blue, lw = 2,␣
  ↪label =:none)
plot!(title = "MLE and Asymptotic Approximation")
```

[ ]:

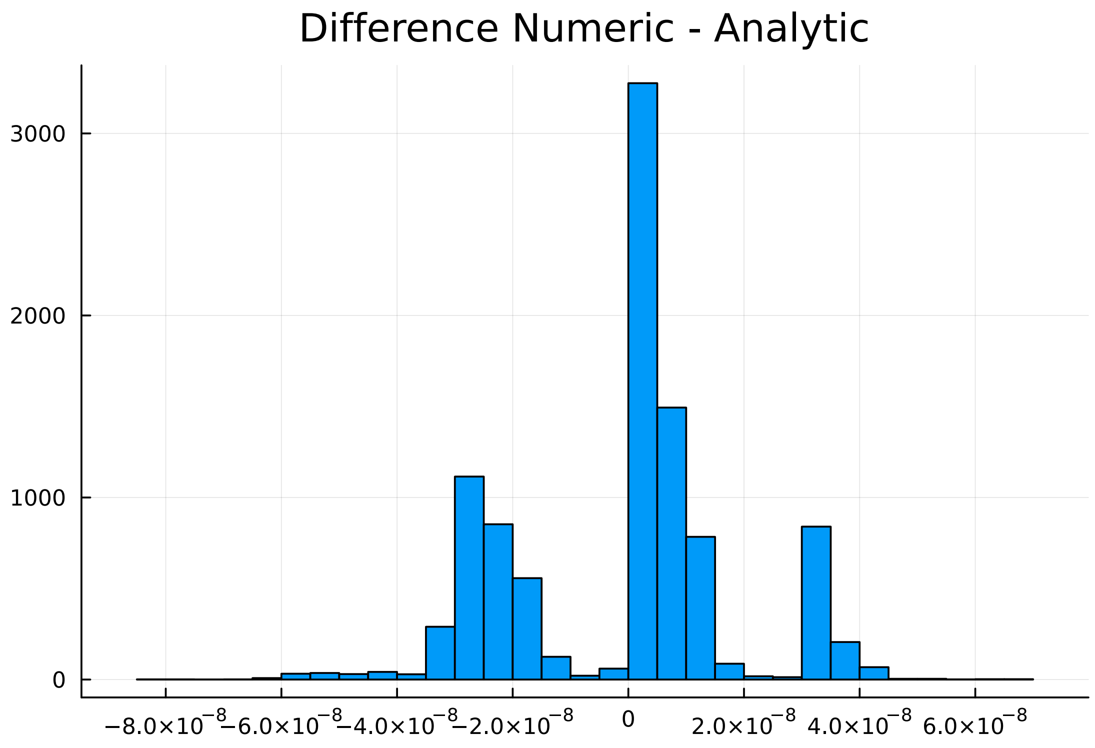


1.3 (c) Comparison Solutions

To compare the solutions we can plot their difference from the simulation results above. Essentially, they do not differ. Probably numeric approximations are very accurate for those easier cases (which probably coincide with closed-form solutions being available).

[ ]:
```
diff = results[:, 1] .- results[:, 2];
histogram(diff, label =:none, title = "Difference Numeric - Analytic")
```

[ ]:


Difference Numeric - Analytic
0
1000
2000
3000
-8.0×10^{-8}
-6.0×10^{-8}
-4.0×10^{-8}
-2.0×10^{-8}
0
2.0×10^{-8}
4.0×10^{-8}
6.0×10^{-8}