Test Tautges Bounds

Bret Larget

10/19/2017

Here is a short simulation to evaluate the bounds you found. Not surprisingly, they are not very tight. My previous study indicated that the approximation should get closer when the minimum alpha gets large, yet the bounds you found are invariant to proportional changes to the alpha parameters.

Here I show what happens with $k = 4$ as typical alphas vary from 2 to 20 to 200 to 2000 with the ratio of maximum lambda to minimum lambda being 10 and 1.1.

Set-up

$Y_1, \ldots, Y_k \sim$ independent Gamma(α_i, λ_i)
$S = \sum_{i=1}^k Y_i$
$X_i = Y_i/S$ for $i = 1, \ldots, k$
Let $m_i(\alpha, \lambda) = \mathsf{E}(X_i)$.
Let $A_i(\alpha, \lambda) = (\alpha_i/\lambda_i)/\sum_{j=1}^k(\alpha_j/\lambda_j)$
Let $\alpha_+ = \sum_{i=1}^k \alpha_i$. Let $\lambda_m = \min_i \lambda_i$ and $\lambda_M = \max_i \lambda_i$.

Tautges Bound:

$$|m_i(\alpha, \lambda) - A_i(\alpha, \lambda)| \leq \frac{\alpha_i(\lambda_M - \lambda_m)}{\alpha_+\lambda_i})$$

Note that for fixed λ but α increasing proportionally, the bounds do not change. I suspect that the true difference gets very small as α increases. Test by simulation. Pick random α and λ with small, but not tiny values. Keep λ fixed and see how $|m_i(\alpha, \lambda) - A_i(\alpha, \lambda)|$ changes when α is repeatedly multiplied by 10.

```r
## Pick parameters at random
parameters = function(k=4,mean.a=2000,shape=2,lambda.ratio=NULL)
{
  alpha = rgamma(k,shape,shape/mean.a)
  lambda = rgamma(k,shape,shape)
  if ( !is.null(lambda.ratio) )
  {
    if ( lambda.ratio < 1 )
      stop("lambda.ratio must be greater than or equal to one")
    if ( lambda.ratio == 1 )
      lambda = rep(1,k)
    else
      lambda = lambda + (max(lambda) - lambda.ratio*min(lambda))/(lambda.ratio-1)
  }
  lambda = lambda/mean(lambda)
  return(data.frame(alpha,lambda))
}

## Generate a large sample in order to
## numerically estimate the means
generate = function(p,n=1000)
{
  k = nrow(p)
  x = with(p, matrix(rgamma(n*k,rep(alpha,each=n),rep(lambda,each=n)),n,k) )
  s = apply(x,1,sum)
  for ( i in 1:n )
    x[i,] = x[i,] / s[i]
  return(x)
}

## Function to calculate bounds from p
bounds.tautges = function(p)
{
  lambda.max = max(p$lambda)
  lambda.min = min(p$lambda)
  out = with(p, alpha * (lambda.max - lambda.min) / (sum(alpha) * lambda))
  return ( out )
}

## approximate mean
approximate = function(p)
{
  w = with(p, alpha/lambda )
  return( w / sum(w) )
}

## Case where max(lambda) / min(lambda) = 10
p.0 = parameters(k=4,mean.a=2,lambda.ratio=10)

p.1 = with(p.0, data.frame(alpha=10*alpha,lambda=lambda))
p.2 = with(p.0, data.frame(alpha=100*alpha,lambda=lambda))
p.3 = with(p.0, data.frame(alpha=1000*alpha,lambda=lambda))

m.0 = apply(generate(p.0,n=1000000),2,mean)
m.1 = apply(generate(p.1,n=1000000),2,mean)
m.2 = apply(generate(p.2,n=1000000),2,mean)
m.3 = apply(generate(p.3,n=1000000),2,mean)

b.0 = bounds.tautges(p.0)
b.1 = bounds.tautges(p.1)
b.2 = bounds.tautges(p.2)
b.3 = bounds.tautges(p.3)

## approximation is the same regardless of mutiple of alpha
a.0 = approximate(p.0)

## difference
d.0 = m.0 - a.0
d.1 = m.1 - a.0
d.2 = m.2 - a.0
d.3 = m.3 - a.0

## results
df.0 = data.frame(p.0,mean=m.0,approx=a.0,diff=d.0,bound=b.0)
df.1 = data.frame(p.1,mean=m.1,approx=a.0,diff=d.1,bound=b.1)
df.2 = data.frame(p.2,mean=m.2,approx=a.0,diff=d.2,bound=b.2)
df.3 = data.frame(p.3,mean=m.3,approx=a.0,diff=d.3,bound=b.3)

## show results
show.results = function(df)
{
  cat("Lambda Ratio =",max(df$lambda)/min(df$lambda),"\n")
  cat("\n","alpha and lambda","\n",sep="")
  print(t(select(df,alpha,lambda)))
  cat("\n","means and approximations","\n",sep="")
  print(t(select(df,mean,approx)))
  cat("\n","differences and bounds","\n",sep="")
  print(t(select(df,diff,bound)))
}

show.results(df.0)
```

```
## Lambda Ratio = 10
##
## alpha and lambda
##            [,1]      [,2]      [,3]      [,4]
## alpha  3.035223 0.5206548 1.8864327 1.3043908
## lambda 2.802231 0.2802231 0.5147745 0.4027716
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.1324649 0.1695360 0.3783859 0.3196132
## approx 0.1100281 0.1887394 0.3722554 0.3289771
##
## differences and bounds
##             [,1]        [,2]        [,3]         [,4]
## diff  0.02243683 -0.01920341 0.006130474 -0.009363897
## bound 0.40489430  0.69454579 1.369869544  1.210608848
```

```
show.results(df.1)
```

```
## Lambda Ratio = 10
##
## alpha and lambda
##             [,1]      [,2]       [,3]       [,4]
## alpha  30.352233 5.2065479 18.8643265 13.0439079
## lambda  2.802231 0.2802231  0.5147745  0.4027716
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.1121886 0.1862734 0.3733193 0.3282187
## approx 0.1100281 0.1887394 0.3722554 0.3289771
##
## differences and bounds
##              [,1]         [,2]        [,3]          [,4]
## diff  0.002160575 -0.002466071 0.001063915 -0.0007584194
## bound 0.404894297  0.694545789 1.369869544  1.2106088479
```

```
show.results(df.2)
```

```
## Lambda Ratio = 10
```

```
##
## alpha and lambda
##              [,1]       [,2]        [,3]        [,4]
## alpha  303.522330 52.0654794 188.6432651 130.4390787
## lambda   2.802231  0.2802231   0.5147745   0.4027716
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.1102333 0.1884991 0.3723729 0.3288946
## approx 0.1100281 0.1887394 0.3722554 0.3289771
##
## differences and bounds
##               [,1]       [,2]         [,3]          [,4]
## diff  0.0002052737 -0.0002403 0.0001175118 -8.248551e-05
## bound 0.4048942967  0.6945458 1.3698695441  1.210609e+00
```

```
show.results(df.3)
```

```
## Lambda Ratio = 10
##
## alpha and lambda
##               [,1]        [,2]         [,3]         [,4]
## alpha  3035.223304 520.6547936 1886.4326510 1304.3907874
## lambda    2.802231   0.2802231    0.5147745    0.4027716
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.1100479 0.1887141 0.3722594 0.3289787
## approx 0.1100281 0.1887394 0.3722554 0.3289771
##
## differences and bounds
##               [,1]          [,2]         [,3]         [,4]
## diff  1.981305e-05 -0.0000253504 3.951557e-06 1.585791e-06
## bound 4.048943e-01  0.6945457891 1.369870e+00 1.210609e+00
```

```
## Case where max(lambda) / min(lambda) = 1.1
p.0 = parameters(k=4,mean.a=2,lambda.ratio=1.1)

p.1 = with(p.0, data.frame(alpha=10*alpha,lambda=lambda))
p.2 = with(p.0, data.frame(alpha=100*alpha,lambda=lambda))
p.3 = with(p.0, data.frame(alpha=1000*alpha,lambda=lambda))

m.0 = apply(generate(p.0,n=1000000),2,mean)
m.1 = apply(generate(p.1,n=1000000),2,mean)
m.2 = apply(generate(p.2,n=1000000),2,mean)
m.3 = apply(generate(p.3,n=1000000),2,mean)

b.0 = bounds.tautges(p.0)
b.1 = bounds.tautges(p.1)
b.2 = bounds.tautges(p.2)
b.3 = bounds.tautges(p.3)

## approximation is the same regardless of mutiple of alpha
a.0 = approximate(p.0)
```

```
## difference
d.0 = m.0 - a.0
d.1 = m.1 - a.0
d.2 = m.2 - a.0
d.3 = m.3 - a.0

## results
df.0 = data.frame(p.0,mean=m.0,approx=a.0,diff=d.0,bound=b.0)
df.1 = data.frame(p.1,mean=m.1,approx=a.0,diff=d.1,bound=b.1)
df.2 = data.frame(p.2,mean=m.2,approx=a.0,diff=d.2,bound=b.2)
df.3 = data.frame(p.3,mean=m.3,approx=a.0,diff=d.3,bound=b.3)

show.results(df.0)
```

```
## Lambda Ratio = 1.1
##
## alpha and lambda
##            [,1]     [,2]      [,3]     [,4]
## alpha  1.619761 1.274434 0.8058427 1.565661
## lambda 1.018673 0.957901 0.9697351 1.053691
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.3045311 0.2519644 0.1576835 0.2858210
## approx 0.3035997 0.2540282 0.1586654 0.2837067
##
## differences and bounds
##               [,1]        [,2]         [,3]        [,4]
## diff  0.0009313545 -0.00206376 -0.000981926 0.002114332
## bound 0.0289254966  0.02420257  0.015116865 0.027030187
```

```
show.results(df.1)
```

```
## Lambda Ratio = 1.1
##
## alpha and lambda
##             [,1]      [,2]      [,3]      [,4]
## alpha  16.197611 12.744344 8.0584268 15.656613
## lambda  1.018673  0.957901 0.9697351  1.053691
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.3037018 0.2538712 0.1585801 0.2838470
## approx 0.3035997 0.2540282 0.1586654 0.2837067
##
## differences and bounds
##               [,1]          [,2]          [,3]         [,4]
## diff  0.0001020582 -0.0001569905 -8.533304e-05 0.0001402653
## bound 0.0289254966  0.0242025664  1.511686e-02 0.0270301870
```

```
show.results(df.2)
```

```
## Lambda Ratio = 1.1
##
## alpha and lambda
##              [,1]       [,2]       [,3]       [,4]
## alpha  161.976113 127.443441 80.5842682 156.566125
## lambda   1.018673   0.957901  0.9697351   1.053691
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.3036204 0.2540174 0.1586558 0.2837063
## approx 0.3035997 0.2540282 0.1586654 0.2837067
##
## differences and bounds
##               [,1]          [,2]          [,3]          [,4]
## diff  2.072019e-05 -1.075825e-05 -9.579885e-06 -3.820548e-07
## bound 2.892550e-02  2.420257e-02  1.511686e-02  2.703019e-02
```

```
show.results(df.3)
```

```
## Lambda Ratio = 1.1
##
## alpha and lambda
##               [,1]        [,2]        [,3]        [,4]
## alpha  1619.761126 1274.434413 805.8426816 1565.661253
## lambda    1.018673    0.957901   0.9697351    1.053691
##
## means and approximations
##             [,1]      [,2]      [,3]      [,4]
## mean   0.3036108 0.2540249 0.1586597 0.2837046
## approx 0.3035997 0.2540282 0.1586654 0.2837067
##
## differences and bounds
##               [,1]          [,2]          [,3]          [,4]
## diff  1.113954e-05 -3.263397e-06 -5.754258e-06 -2.121884e-06
## bound 2.892550e-02  2.420257e-02  1.511686e-02  2.703019e-02
```