Assignment

Rahul Goswami

17/04/2021

Problem 1

Generate 100 random from a Bivariate Generalized exponential, Bivariate Weibull distribution.

Bivariate Generalized Exponential

To generate Random Variables From Bivariate Generalized Exponential Distribution We can use following Steps

1. Generate $U_1 \sim GE(\alpha_1, \lambda)$,$U_2 \sim GE(\alpha_2, \lambda)$,$U_3 \sim GE(\alpha_3, \lambda)$
2. Assign $X_1 = max\{U_1, U_3\}$ and $X_2 = max\{U_2, U_3\}$
3. Return (X_1, X_2)

Now to Generate Random Numbers from Generalized Exponential Distribution we can use Inverse Transform Method as follows

$$\textit{CDF of Generealized Exponential is given by}$$
$$F(x, \alpha, \lambda) = (1 - e^{-\lambda x})^{\alpha}$$
$$U = (1 - e^{-\lambda x})^{\alpha}$$
$$x = -\frac{1}{\lambda} ln\left(1 - U^{\frac{1}{\alpha}}\right)$$

Creating a function to generate Random Variable from Generalized Exponential

```
rge <- function(alpha , lambda){
  u = runif(1)
  x = -log(1 - u^(1/alpha))/lambda
  return(x)
}
```

Creating a function to generate Random Variable from Bivariate Generalized Exponential

```
rbge <- function(alpha1,alpha2,alpha3,lambda){
  u1 = rge(alpha1,lambda)
  u2 = rge(alpha2,lambda)
  u3 = rge(alpha3,lambda)
  x1 = max(u1,u3)
  x2 = max(u2,u3)
  x = c(x1,x2)
  return(x)
}
```

Code

```
a = c()
for(i in 1:100){
  x = rbge(20,24,32,0.1)
  a = rbind(a,x)

}
print(a)
```

```
##       [,1]     [,2]
## x 41.99365 48.02924
## x 84.54964 84.54964
## x 71.31671 71.31671
## x 56.29463 72.97730
## x 45.61326 43.56582
## x 59.90066 59.90066
## x 33.67610 34.38307
## x 29.71290 62.83199
## x 63.27534 63.27534
## x 43.24341 55.20256
## x 41.89455 42.34041
## x 38.74686 38.74686
## x 48.84726 48.84726
## x 41.01223 37.51943
## x 37.87980 60.20198
## x 58.44875 43.35901
## x 48.53204 43.07333
## x 46.59992 56.16185
## x 69.88092 69.88092
## x 44.33247 44.33247
## x 33.19426 33.19426
## x 37.11055 37.11570
## x 36.82111 55.42992
## x 54.58325 54.58325
## x 34.36849 35.73153
## x 45.00471 42.82809
## x 41.33643 17.06479
## x 31.41372 49.44332
## x 33.50605 56.19779
## x 38.53629 38.53629
## x 39.83927 29.60727
## x 70.34399 70.34399
## x 26.22440 55.53872
## x 35.12812 26.92272
## x 41.38854 41.38854
## x 53.48172 53.48172
## x 44.15145 44.15145
## x 34.05299 46.05530
## x 70.21891 70.21891
## x 48.69600 63.32387
## x 27.57910 34.03891
## x 55.39665 38.58795
## x 38.21781 29.86864
```

```
## x 74.78757 74.78757
## x 46.01403 62.32023
## x 29.38010 29.38010
## x 40.40693 40.40693
## x 49.76836 50.19706
## x 41.30158 44.65533
## x 47.88438 41.47846
## x 39.57274 26.45684
## x 69.01348 36.98403
## x 63.00509 50.03270
## x 53.78441 53.78441
## x 42.29096 42.29096
## x 33.70260 33.70260
## x 56.96310 45.87057
## x 32.25996 26.07941
## x 72.17983 65.00668
## x 40.51883 40.51883
## x 37.55935 42.41114
## x 87.09092 35.50510
## x 36.64516 45.95023
## x 52.41982 52.41982
## x 33.75020 53.69102
## x 49.76604 49.76604
## x 22.65993 45.00964
## x 53.19440 52.36641
## x 39.46953 52.90942
## x 36.85876 36.85876
## x 27.20792 34.83477
## x 38.99652 38.99652
## x 32.30965 43.09342
## x 29.34651 46.33507
## x 40.49562 40.49562
## x 54.09442 40.01690
## x 43.34452 43.34452
## x 44.38336 44.38336
## x 49.18426 38.36949
## x 32.13821 36.61144
## x 31.78637 31.80202
## x 27.22829 43.21675
## x 40.91782 40.91782
## x 37.00406 37.00406
## x 28.20972 40.58600
## x 47.67954 33.00677
## x 97.56384 97.56384
## x 41.85305 49.19431
## x 28.34411 54.10253
## x 29.36161 32.91317
## x 57.82818 57.82818
## x 54.90841 36.48342
## x 56.06117 56.06117
## x 46.64769 61.22385
## x 64.94450 61.34838
## x 39.27374 50.60885
## x 42.44870 42.44870
```

```
## x 56.26771 56.26771
## x 31.09044 40.52045
## x 56.80733 74.53683
```

Bivariate Weibull Distribution

To generate Random Variables From Bivariate Weibull Distribution We can use following Steps

1. Generate $U_1 \sim WE(\alpha, \lambda_1)$,$U_2 \sim WE(\alpha, \lambda_2)$,$U_3 \sim WE(\alpha, \lambda_3)$
2. Assign $X_1 = min\{U_1, U_2\}$ and $X_2 = min\{U_1, U_3\}$
3. Return (X_1, X_2)

Code

```
rbwd <- function(n,alpha,lambda1,lambda2,lambda3){
  #u1 = rweibull(n,alpha,1/lambda1)
  #u2 = rweibull(n,alpha,1/lambda2)
  #u3 = rweibull(n,alpha,1/lambda3)
  u1 = (-log(1-runif(n))/lambda1)^(1/alpha)
  u2 = (-log(1-runif(n))/lambda2)^(1/alpha)
  u3 = (-log(1-runif(n))/lambda3)^(1/alpha)
  x1 = pmin(u1,u2)
  x2 = pmin(u1,u3)
  x = cbind(x1,x2)
  return(x)
}
```

```
b = rbwd(100,10,22,3,7)
print(b)
```

```
##               x1        x2
##   [1,] 0.5663900 0.5663900
##   [2,] 0.6429338 0.6429338
##   [3,] 0.7564137 0.7564137
##   [4,] 0.5790414 0.5790414
##   [5,] 0.7086963 0.7086963
##   [6,] 0.6465000 0.6465000
##   [7,] 0.7442693 0.7442693
##   [8,] 0.5657055 0.5657055
##   [9,] 0.6318269 0.6318269
##  [10,] 0.7195953 0.7172698
##  [11,] 0.6793598 0.6793598
##  [12,] 0.6453141 0.6453141
##  [13,] 0.7418274 0.7418274
##  [14,] 0.6778108 0.6778108
##  [15,] 0.6686157 0.6857732
##  [16,] 0.7060477 0.7060477
##  [17,] 0.6018897 0.6018897
##  [18,] 0.6773092 0.7424668
##  [19,] 0.6555253 0.6555253
##  [20,] 0.6349531 0.6349531
##  [21,] 0.6765257 0.6765257
##  [22,] 0.7266444 0.7927770
```

```
##  [23,] 0.6292479 0.7605644
##  [24,] 0.6986104 0.7940235
##  [25,] 0.7297163 0.7297163
##  [26,] 0.6610732 0.6498738
##  [27,] 0.7972194 0.7972194
##  [28,] 0.7610399 0.7610399
##  [29,] 0.5533581 0.5533581
##  [30,] 0.7526247 0.7146882
##  [31,] 0.7044340 0.7202764
##  [32,] 0.7085273 0.7085273
##  [33,] 0.5440485 0.5440485
##  [34,] 0.7477012 0.7420195
##  [35,] 0.7753385 0.7753385
##  [36,] 0.6551676 0.6551676
##  [37,] 0.7216827 0.7216827
##  [38,] 0.7881534 0.7979815
##  [39,] 0.6744738 0.6744738
##  [40,] 0.7172460 0.7172460
##  [41,] 0.6260866 0.6260866
##  [42,] 0.8051348 0.7480163
##  [43,] 0.7174112 0.7174112
##  [44,] 0.7053443 0.7053443
##  [45,] 0.7834189 0.7834189
##  [46,] 0.7342257 0.7342257
##  [47,] 0.7643992 0.6341006
##  [48,] 0.7834805 0.8281456
##  [49,] 0.5598610 0.5598610
##  [50,] 0.5458333 0.5458333
##  [51,] 0.8012822 0.8220117
##  [52,] 0.8362701 0.7318067
##  [53,] 0.6593818 0.6593818
##  [54,] 0.7100151 0.7100151
##  [55,] 0.7195002 0.7195002
##  [56,] 0.7222662 0.7517411
##  [57,] 0.7379587 0.6619997
##  [58,] 0.7194300 0.6751950
##  [59,] 0.7720389 0.7720389
##  [60,] 0.7541081 0.7241915
##  [61,] 0.8404376 0.7102231
##  [62,] 0.6442624 0.6442624
##  [63,] 0.6770637 0.6770637
##  [64,] 0.6078558 0.6078558
##  [65,] 0.7383883 0.7383883
##  [66,] 0.7247244 0.6921853
##  [67,] 0.7344823 0.7712487
##  [68,] 0.6071363 0.6071363
##  [69,] 0.7180941 0.6937736
##  [70,] 0.6615459 0.6615459
##  [71,] 0.7243831 0.7243831
##  [72,] 0.7691712 0.5572681
##  [73,] 0.6948745 0.6948745
##  [74,] 0.7467419 0.7467419
##  [75,] 0.7078916 0.7078916
##  [76,] 0.7237366 0.7237366
```

```
##  [77,] 0.6157513 0.6924633
##  [78,] 0.7218467 0.7218467
##  [79,] 0.6666714 0.6666714
##  [80,] 0.7762347 0.7366074
##  [81,] 0.6627836 0.6627836
##  [82,] 0.7591942 0.7584358
##  [83,] 0.7708435 0.7708435
##  [84,] 0.6675292 0.6675292
##  [85,] 0.6770450 0.6770450
##  [86,] 0.7337176 0.7337176
##  [87,] 0.7942545 0.7942545
##  [88,] 0.5955502 0.5955502
##  [89,] 0.5742818 0.5742818
##  [90,] 0.8400619 0.8400619
##  [91,] 0.6752236 0.6752236
##  [92,] 0.7665365 0.7665365
##  [93,] 0.6776916 0.6776916
##  [94,] 0.7985080 0.5619865
##  [95,] 0.7170920 0.6269078
##  [96,] 0.6229591 0.6229591
##  [97,] 0.7535695 0.7535695
##  [98,] 0.7642843 0.7642843
##  [99,] 0.6118826 0.8093943
## [100,] 0.7288070 0.7288070
```

Problem 2

Generate 100 random from an absolutely continuous bivariate Generalized exponential, absolutely continuous Bivariate Weibull distribution

Absolutely Continuous Bivariate Generalized Exponential

To Get Random Number with absolutely continuous bivariate Generalized exponential we can use the same procedure as we have done in Problem 1. We just have to take care of singular points so we will reject the sample when $X_1 = X_2$

Code

```
c = c()
while(length(c)<200){
  x = rbge(1,2,3,4)
  if(x[1] == x[2]){
    next
  }
  c = rbind(c,x)
}
print(c)
```

```
##         [,1]      [,2]
## x 0.05828441 0.2692347
## x 0.38181758 0.5765852
## x 0.27309742 0.6464860
## x 0.54016035 0.4740354
## x 0.31583558 0.3278167
```

```
## x 0.55314925 0.3526060
## x 0.58356800 0.2140301
## x 0.57060925 0.6625897
## x 0.16706693 0.4179797
## x 0.63696634 0.3013049
## x 0.59481381 0.2228073
## x 0.50362041 0.5986448
## x 0.39295191 0.7999158
## x 0.82525859 1.2725986
## x 0.34659909 0.3475303
## x 0.16602930 0.2714359
## x 0.27710762 0.3296771
## x 0.25030462 0.3293964
## x 0.04861319 0.3429535
## x 0.75202971 0.1525736
## x 0.18832163 0.2202402
## x 0.26826883 0.3404554
## x 1.24160983 0.6482267
## x 0.32692243 0.5139547
## x 0.60903385 0.3827821
## x 0.78848676 0.4407286
## x 0.52953905 0.3936250
## x 0.45474108 0.7794283
## x 0.27192480 0.5597556
## x 0.31203670 0.5625364
## x 0.14918394 0.9307568
## x 0.25715849 0.4818839
## x 0.17875114 0.2310882
## x 0.22293471 0.3068943
## x 0.57604339 0.2589143
## x 0.19518719 0.2430268
## x 0.24269716 0.2566366
## x 0.48826872 0.4826069
## x 0.45474123 0.2833789
## x 0.55122222 0.5223691
## x 1.61524523 0.5563904
## x 0.38445522 0.4664184
## x 0.36665278 0.9579928
## x 0.13323553 0.2998806
## x 0.18772287 0.2288205
## x 0.26473617 0.8253631
## x 0.15870895 0.2418131
## x 0.23964406 0.4042824
## x 0.63448486 0.9608914
## x 0.76649586 0.8056085
## x 0.46881845 0.9066976
## x 0.37929643 0.9150091
## x 0.36465070 0.5315085
## x 0.72461349 0.1876474
## x 0.08108303 0.2022981
## x 0.20554839 0.7678355
## x 0.38013634 0.4513241
## x 0.41062291 0.3322605
## x 0.10878965 0.2810203
```

```
## x 0.20637746 0.6919916
## x 0.25581748 0.2609306
## x 0.29544432 0.6193957
## x 0.42719393 0.3445795
## x 0.24629855 0.2711945
## x 1.27241455 0.4632999
## x 0.17029050 0.1787515
## x 0.27860112 1.0405413
## x 0.28902101 0.4291638
## x 0.26153192 0.1924360
## x 0.19466803 0.7224044
## x 0.35410007 0.2755381
## x 0.16681200 1.1413403
## x 0.09178637 0.3747548
## x 0.22329185 0.2692228
## x 0.22124621 0.1562482
## x 0.54956995 0.5618586
## x 0.66665118 0.3746826
## x 0.40292678 0.3827395
## x 0.20885577 0.2446095
## x 1.32602233 0.5592191
## x 0.28995085 1.0302982
## x 0.33338769 0.4202210
## x 0.34919024 0.2520410
## x 0.45733013 0.2906634
## x 0.47762711 0.7563812
## x 0.14476516 0.1886555
## x 1.11887137 0.2308400
## x 0.22226937 0.2739824
## x 0.39461412 0.3549266
## x 0.32973939 0.2532404
## x 0.26642195 0.1254591
## x 0.42398528 0.3847822
## x 0.62465368 0.2217283
## x 0.10250132 0.4765906
## x 0.71164856 0.2783289
## x 0.55623759 0.2387400
## x 0.54780094 0.5990340
## x 0.26665145 0.2643499
## x 0.29611694 0.3093146
## x 0.29491900 0.1925865
```

Absolutely Continuous Bivariate Weibull distribution

Similarly as above we can write

```
d = c()
while(length(d)<200){
  x = rbwd(1,1,2,3,4)
  if(x[1] == x[2]){
    next
  }
  d = rbind(d,x)
}
print(d)
```

```
##                x1          x2
##   [1,] 0.102574908 0.584122013
##   [2,] 0.020688856 0.017042286
##   [3,] 0.080536704 0.053027200
##   [4,] 0.189146821 0.367614196
##   [5,] 0.356567683 0.236602566
##   [6,] 0.511513750 0.543645374
##   [7,] 0.229031056 0.013177780
##   [8,] 0.010585070 0.316884703
##   [9,] 0.092762185 0.056172600
##  [10,] 0.040736613 0.112813194
##  [11,] 0.021565008 0.184622108
##  [12,] 0.111874700 0.201424768
##  [13,] 0.031789638 0.133347148
##  [14,] 0.630770145 0.045833780
##  [15,] 0.066606853 0.576431833
##  [16,] 0.002957425 0.155116012
##  [17,] 0.320380979 0.007674417
##  [18,] 0.093631316 0.154233316
##  [19,] 0.301684865 0.091396639
##  [20,] 0.068091135 0.015088343
##  [21,] 0.070827294 0.446115747
##  [22,] 0.791257110 0.507940770
##  [23,] 0.028933420 0.026529850
##  [24,] 0.327813933 0.612126297
##  [25,] 0.333551400 0.094152773
##  [26,] 0.465143979 0.018625932
##  [27,] 0.200303386 0.056714771
##  [28,] 0.112729374 0.041364159
##  [29,] 0.062364535 0.057862578
##  [30,] 0.208683363 0.162714719
##  [31,] 0.007893803 0.097468409
##  [32,] 0.598430104 0.433590958
##  [33,] 0.320656837 0.027281458
##  [34,] 0.135393841 0.084331561
##  [35,] 0.398214435 0.664837285
##  [36,] 0.263811970 0.120640337
##  [37,] 0.265104805 0.119081707
##  [38,] 0.152144352 0.436874962
##  [39,] 0.184878271 0.010292956
##  [40,] 0.012231205 0.220132463
##  [41,] 0.220286958 0.534212443
##  [42,] 0.448791296 0.056095129
##  [43,] 0.272887363 0.204389014
##  [44,] 0.151483463 0.252932018
##  [45,] 0.424902804 0.401235442
##  [46,] 0.398336358 0.196009972
##  [47,] 0.190867465 0.112802854
##  [48,] 0.187894290 0.655223492
##  [49,] 0.068281087 0.103683068
##  [50,] 0.230764027 0.019249557
##  [51,] 0.107969923 0.089982827
```

```
##  [52,] 0.063392608 0.047649762
##  [53,] 0.105395591 0.077271989
##  [54,] 0.287141995 0.198436816
##  [55,] 0.191292144 0.426613849
##  [56,] 0.007189029 0.096597361
##  [57,] 0.203586048 0.037733710
##  [58,] 0.118295960 0.166000081
##  [59,] 0.027882401 0.123924044
##  [60,] 0.098138132 0.138334249
##  [61,] 0.071381192 0.262072409
##  [62,] 0.615346763 0.350792437
##  [63,] 0.290370418 0.157907359
##  [64,] 0.338256074 0.052132953
##  [65,] 0.467033497 0.079417469
##  [66,] 0.858987999 0.434143961
##  [67,] 0.216062560 0.061034162
##  [68,] 0.218780797 0.029376657
##  [69,] 0.177060372 0.034758069
##  [70,] 0.767205038 0.156062506
##  [71,] 0.124037186 0.221040058
##  [72,] 0.089722788 0.686534304
##  [73,] 0.284870605 0.059899113
##  [74,] 0.064369399 0.097670250
##  [75,] 0.058798278 0.376124502
##  [76,] 0.199221526 0.160141351
##  [77,] 0.015547904 0.050880136
##  [78,] 0.960791824 0.336888969
##  [79,] 0.711688789 0.755666795
##  [80,] 0.262155155 0.393162047
##  [81,] 0.088521232 0.059260416
##  [82,] 0.059459279 0.012800337
##  [83,] 0.051076551 0.383311749
##  [84,] 0.224288672 0.154111618
##  [85,] 0.330179915 0.258734261
##  [86,] 0.125011377 0.054131314
##  [87,] 0.138006604 0.073104255
##  [88,] 0.422025116 0.382902703
##  [89,] 0.223169243 0.030514069
##  [90,] 0.017223576 0.161427559
##  [91,] 0.052719304 0.072212961
##  [92,] 0.205309963 0.155574516
##  [93,] 0.058827977 0.268866257
##  [94,] 0.304481015 0.065437767
##  [95,] 0.416811661 0.084055193
##  [96,] 0.235898000 0.084237744
##  [97,] 0.035818787 0.304678825
##  [98,] 0.106559667 0.334344670
##  [99,] 0.321334963 0.490195202
## [100,] 0.654209877 0.147245611
```

Problem 3

Estimate parameters of this four bivariate distributions using EM algorithm.

EM Algorithm for Bivariate Generalized Exponential

We have stored the random numbers generated from Bivariate Generalized Exponential in variable **a,** let us define

$$I_1 = \{X_{1i} < X_{2i}\},\ I_2 = \{X_{1i} > X_{2i}\}\ and\ I_0 = \{X_{1i} = X_{2i}\}$$

```
# I0 = a[a[,1]==a[,2],]
# I1 = a[a[,1]<a[,2],]
# I2 = a[a[,1]>a[,2],]
```

Let us define three function such as

```
# uwa <-function(gamma){
# u1 <- (gamma[1]/(gamma[1]+gamma[3]))
# u2 <- (gamma[3]/(gamma[1]+gamma[3]))
# w1 <- (gamma[2]/(gamma[2]+gamma[3]))
# w2 <- (gamma[3]/(gamma[2]+gamma[3]))
# c(u1,u2,w1,w2)
# }
```

```
# updatealphas <- function(lambda,I0,I1,I2,uw){
#   n0 = length(I0)/2
#   n1 = length(I1)/2
#   n2 = length(I2)/2
#   sumi0 = sum(log(1-exp(-lambda*I0[,1])))
#   sumi1i21 = sum(log(1-exp(-lambda*rbind(I1,I2)[,1])))
#   sumi1i22 = sum(log(1-exp(-lambda*rbind(I1,I2)[,2])))
#   sumi1 = sum(log(1-exp(-lambda*I1[,1])))
#   sumi2 = sum(log(1-exp(-lambda*I2[,2])))
#   a = (n1*uw[1] + uw[2])/(sumi0 + sumi1i21+0.001)
#   b = (n1 + uw[3]*n2)/(sumi0 + sumi1i22+0.001)
#   c = (n0 + n1*uw[2] + n2*uw[4])/(sumi0 + sumi1 + sumi2+0.001)
#   #print(c(a,b,c,sumi0,sumi1i21,sumi1i22,sumi1,sumi2))
#   c(a,b,c)
# }
```

```
# updatelambda <- function(gamma,I0,I1,I2,uw){
#   #gamma[1:3] <- updatealphas(gamma[4],I0,I1,I2,uw)
#   i1Ui2 = rbind(I1,I2)
#   n0 = length(I0)/2
#   n1 = length(I1)/2
#   n2 = length(I2)/2
#   a = gamma[1]+gamma[2]+gamma[3]-1
#   b = gamma[1]+gamma[3]-1
#   c = gamma[2]+gamma[3]-1
#   d = gamma[2]-1
#   e = gamma[1]-1
#   s =(sum(I0[,1])+sum(i1Ui2[,1])+sum(i1Ui2[,2]))
#   s1 = sum((I0[,1]*(exp(-gamma[4]*I0[,1])))/(1-exp(-gamma[4]*I0[,1])))
#   s2 = sum((I1[,1]*(exp(-gamma[4]*I1[,1])))/(1-exp(-gamma[4]*I1[,1])))
#   s3 = sum((I2[,2]*(exp(-gamma[4]*I2[,2])))/(1-exp(-gamma[4]*I2[,2])))
#   s4 = sum((I1[,2]*(exp(-gamma[4]*I1[,2])))/(1-exp(-gamma[4]*I1[,2])))
```

```
#   s5 = sum((I2[,1]*(exp(-gamma[4]*I2[,1])))/(1-exp(-gamma[4]*I2[,1])))
#   l = c(a,b,c,d,e,s,s1,s2,s3,s4,s5)
#   print(l)
#   #print(gamma)
#   res = (s-a*s1-b*s2-c*s3-d*s4-e*s5)*(n0 + 2*n1 + 2*n2)
#   res
#
# }
```

Code

```
# Initialize Parameter
alpha1 = 1
alpha2 = 2
alpha3 = 3
lambda = 4
gamma = c(alpha1,alpha2,alpha3,lambda)

#Calculating u1,u2,w1,w2
lamb = 0

# while(abs(4-lamb) > 0.01){
#
# uw = uwa(gamma)
# # Updating lambda
# lambda = gamma[4]
# if(exp(-lambda)==0){lambda = -50}
# lamb = updatelambda(gamma,I0,I1,I2,uw)
# gamma[4] = lamb
#
# # Updating Alphas
#
# gamma[1:3] = updatealphas(gamma[4],I0,I1,I2,uw)
# print(lamb)
#
#
# }
```

EM Algorithm for Bivariate Generalized Weibull

```
# I0 = b[b[,1]==b[,2],]
# I1 = b[b[,1]<b[,2],]
# I2 = b[b[,1]>b[,2],]
```

Calculating u's and v's

```
# uva <-function(gamma){
# u1 <- (gamma[1]/(gamma[1]+gamma[3]))
# u2 <- (gamma[3]/(gamma[1]+gamma[3]))
# v1 <- (gamma[1]/(gamma[1]+gamma[2]))
# v2 <- (gamma[2]/(gamma[1]+gamma[2]))
# #print("uva ho gaya")
# c(u1,u2,v1,v2)}
```

Update

```
# updatealpha <- function(gamma,I0,I1,I2,uv){
#   #gamma[1:3] <- updatelambdas(gamma[4],I0,I1,I2,uv)
#   #print(gamma)
#   n0 = length(I0)/2
#   n1 = length(I1)/2
#   n2 = length(I2)/2
#   i1Ui2 = rbind(I1,I2)
#   s1 = sum((I0[,1]^(gamma[4]))*log(I0[,1]))+sum((I1[,2]^(gamma[4]))*log(I1[,2]))+sum((I2[,1]^(gamma[4
#   s2 = sum((I0[,1]^(gamma[4]))*log(I0[,1])) + sum((i1Ui2[,1]^(gamma[4]))*log(i1Ui2[,1]))
#   s3 = sum(((I0[,1]^(gamma[4]))*log(I0[,1]))* sum((i1Ui2[,2]^(gamma[4]))*log(i1Ui2[,2])))
#   s4 = sum(log(I0[,1])) + sum(log(i1Ui2[,1])+log(i1Ui2[,2]))
#   #print(gamma)
#   res = gamma[1]*s1+ gamma[2]*s2 + gamma[3]*s3 -s4
#   #l = c(s1,s2,s3,s4,res,(n0 + 2*n1 + 2*n2))
#   #print(l)
#   (n0 + 2*n1 + 2*n2)/res
#
# }
```

Update Lambdas

```
# updatelambdas <- function(alphas,I0,I1,I2,uv){
#   n0 = length(I0)/2
#   n1 = length(I1)/2
#   n2 = length(I2)/2
#   i1Ui2 = rbind(I1,I2)
#   s1 = sum(I0[,1]^alphas)
#   s2 = sum(I2[,1]^alphas)
#   s3 = sum(I1[,2]^alphas)
#   s4 = sum(i1Ui2[,1]^alphas)
#   s5 = sum(i1Ui2[,2]^alphas)
#   a = (n0+n1*uv[1] + n2*uv[3])/(s1+s2+s3)
#   b = (n1 + uv[4]*n2)/(s1+s4)
#   c = (n2 + n1*uv[2])/(s1+s5)
#   #print(c(alphas,s1,s2,s3,s4,s5,uv,a,b,c,n1,n2))
#   #print("update lambda ho gaya")
#   c(a,b,c)
# }
```

Code

```
# gamma <- c(1,1,1,1)
#
#
# for(i in 1:50){
#   uv = uva(gamma)
#   gamma[4] <- updatealpha(gamma,I0,I1,I2,uv)
#   gamma[1:3] <- updatelambdas(gamma[4],I0,I1,I2,uv)
#   print(gamma)
# }
```