R-package: Distribución Marshall-Olkin Extended Zipf

VIII JORNADAS DE USUARIOS DE R.
Albacete - Noviembre, 2016

A. Duarte-López [1], A. Casellas [2] and M. Pérez-Casany[1,3]

[1]Data Management Group (DAMA - UPC)

[2]ISGlobal, Barcelona Ctr. Int. Health Res. (CRESIB), Hospital Clínic-Universidad de Barcelona

[3]Dpt. Estadística e Investigación Operativa, Universidad Politécnica de Cataluña

DAMA-UPC. DATA MANAGEMENT
UNIVERSITAT POLITÈCNICA DE CATALUNYA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Indice

Objetivos

- Implementar la distribución Marshall-Olkin extended Zipf (MOEZipf).
- Facilitar el uso de la MOEZipf a los investigadores de diferentes áreas de investigación.
- Posibilitar la reproducción de los experimentos realizados en investigación.
- Permitir la generación de datos sintéticos con distribución MOEZipf.

Motivación

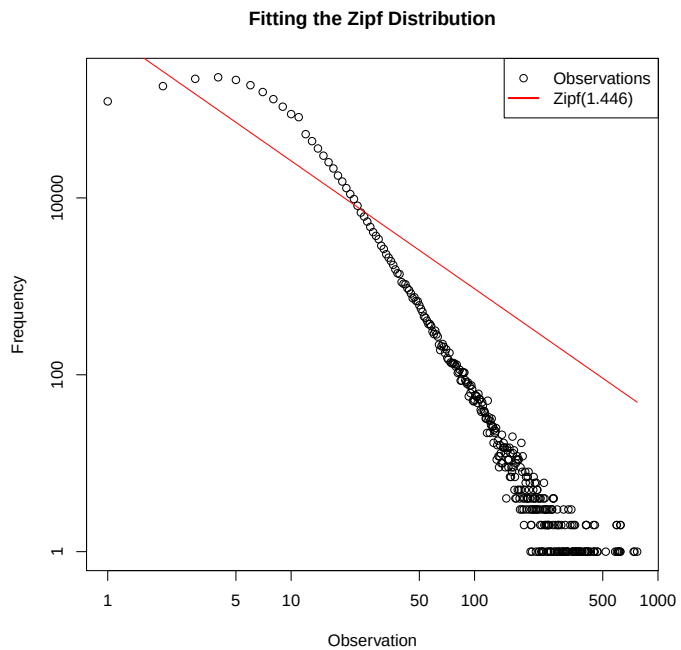
La distribución Zipf tiene su función de probabilidad:

$$P(Y = x) = \frac{x^{-\alpha}}{\zeta(\alpha)}, x = 1, 2, 3, \ldots$$

donde $\alpha > 1$ y $\zeta(\alpha)$ es la función Zeta de Riemann.

Sin embargo, el ajuste no siempre es satisfactorio debido a:

1) Concavidad/Convexidad en los primeros valores.
2) Frecuencia en 1 superior/inferior a lo esperado.



Apuntes teóricos

Transformación Marshall-Olkin

La transformación Marshall-Olkin [4] permite generalizar cualquier distribución de probabilidad mediante la adición de un parámetro extra. La familia de probabilidad extendida tiene función de supervivencia:

$$\overline{G}(x) = \frac{\beta \overline{F}(x)}{1 - \overline{\beta} \overline{F}(x)}, \beta > 0$$

donde $-\infty < x < +\infty$, $\overline{\beta} = 1 - \beta$ y $\overline{F}(x)$ es la función de supervivencia de la familia orginal.

La familia original se obtiene cuando $\beta = 1$.

Marshall-Olkin extended Zipf

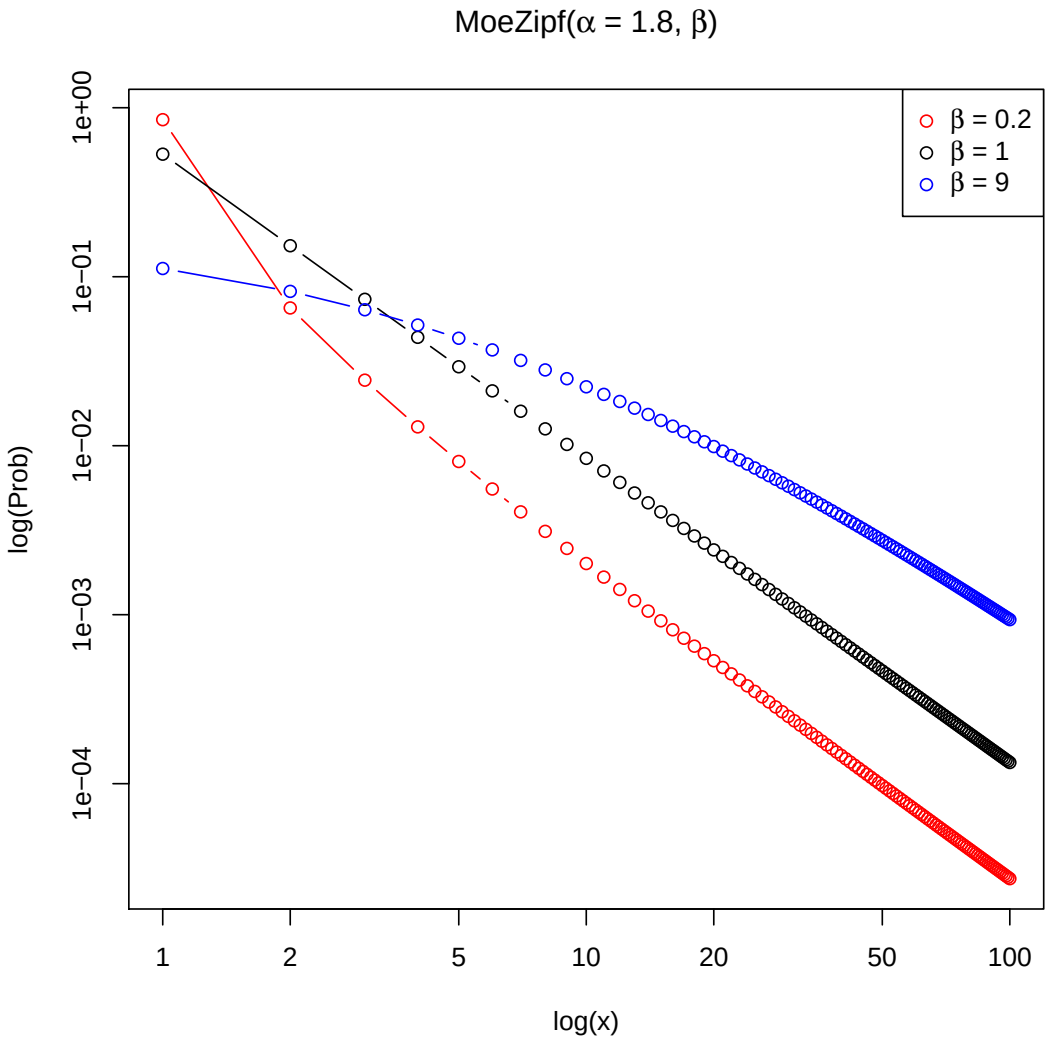
La distribución MOEZipf [5] tiene función de supervivencia igual a:

$$\overline{G}(x;\alpha,\beta) = \frac{\beta\zeta(\alpha, x+1)}{\zeta(\alpha) - \overline{\beta}\zeta(\alpha, x+1)}, \beta > 0, \alpha > 1,$$

y función de probabilidades:

$$P(Y = x) = \frac{x^{-\alpha}\beta\zeta(\alpha)}{[\zeta(\alpha) - \overline{\beta}\zeta(\alpha, x)][\zeta(\alpha) - \overline{\beta}\zeta(\alpha, x+1)]}, x = 1, 2, 3, \ldots$$

Marshall-Olkin extended Zipf



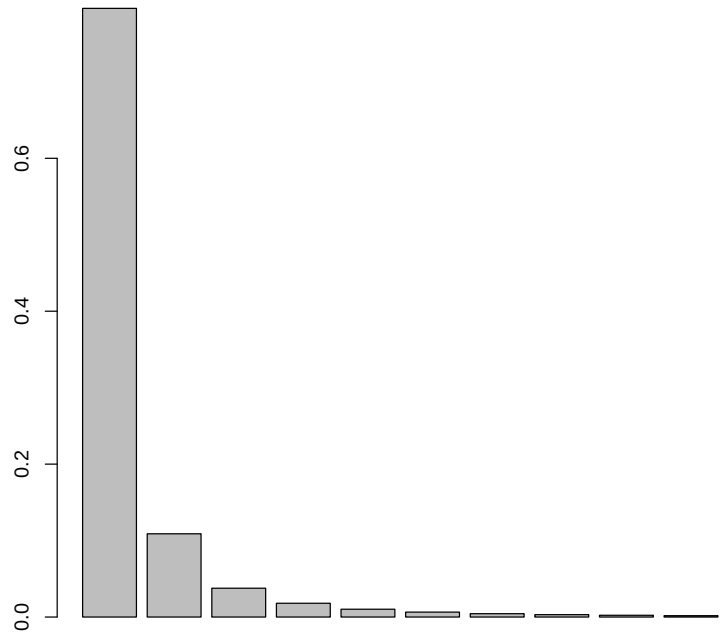
Implementación

Principales funcionalidades

dmoezipf	Función de probabilidad.
moezipfR.log.density	Logaritmo de las probabilidades.
pmoezipf	Función de probabilidad acumulada.
smoezipf	Función de supervivencia.
qmoezipf	Función de cuantiles.
rmoezipf	Generador de números aleatorios.
moezipfR.mean	Esperanza de la distribución.
moezipfR.var	Varianza de la distribución.
moezipfR.moments	$k - ésimo$ momento de la distribución.
moezipfR.loglikelihood	Valor de la log-verosimilutud.
moezipfR.fit	Estimador máximo verosímil de α y β.
moezipfR.confint	Intervalos de confianza de los parámetros.
zipf.fit	Estimador máximo verosímil de α (distribución Zipf).

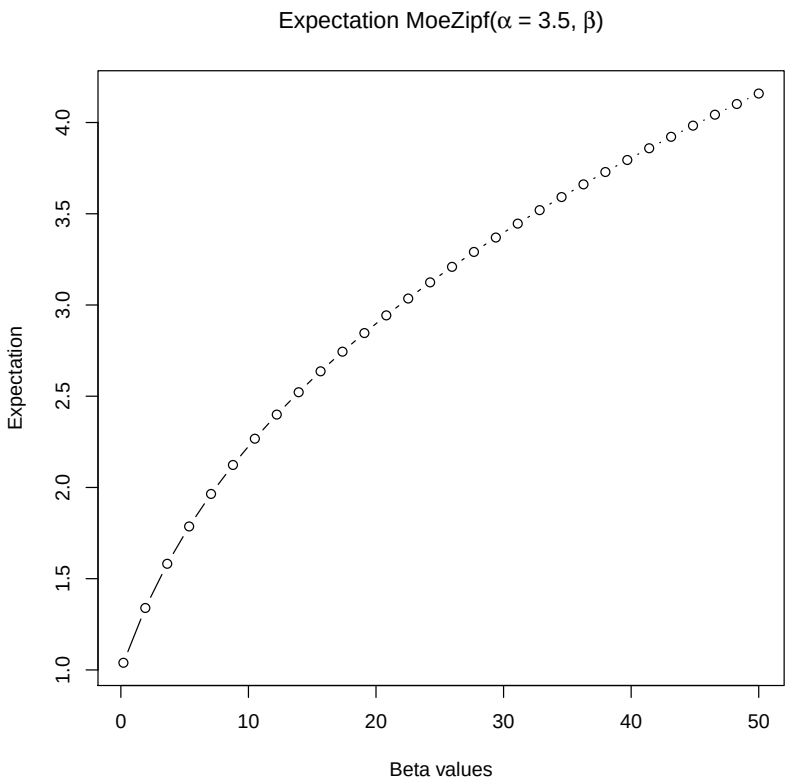
Principales funcionalidades

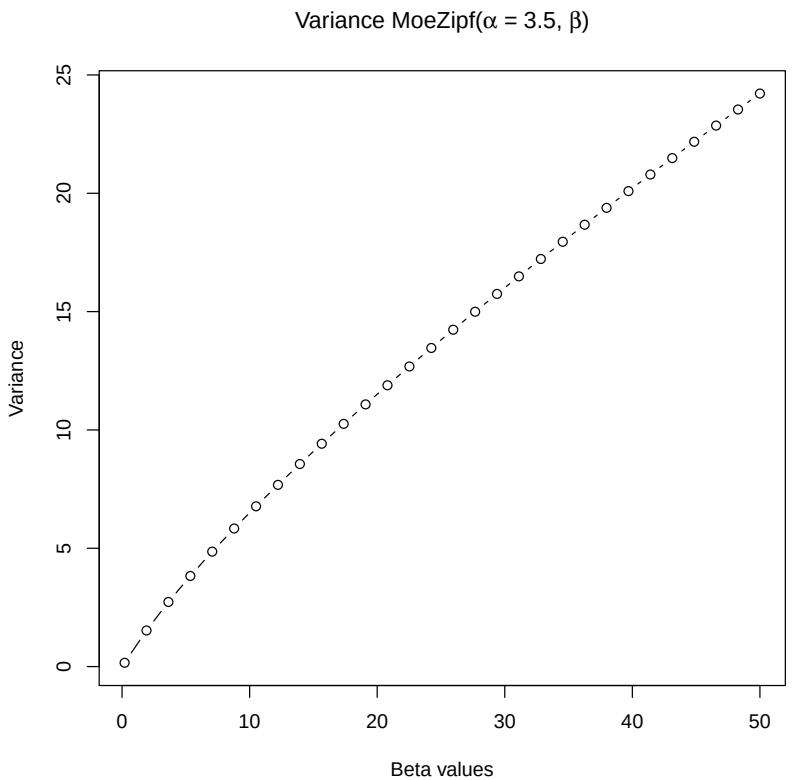
```
library(moezipfR)

> density <- dmoezipf(1:10, alpha
    = 2.5, beta = 0.75, show.plot
    = T)
 [1] 0.796105171 0.108892131
     0.037707477 0.018037945
     0.010230052 0.006449629
 [7] 0.004371329 0.003122881
     0.002322148 0.001781988
```



Principales funcionalidades

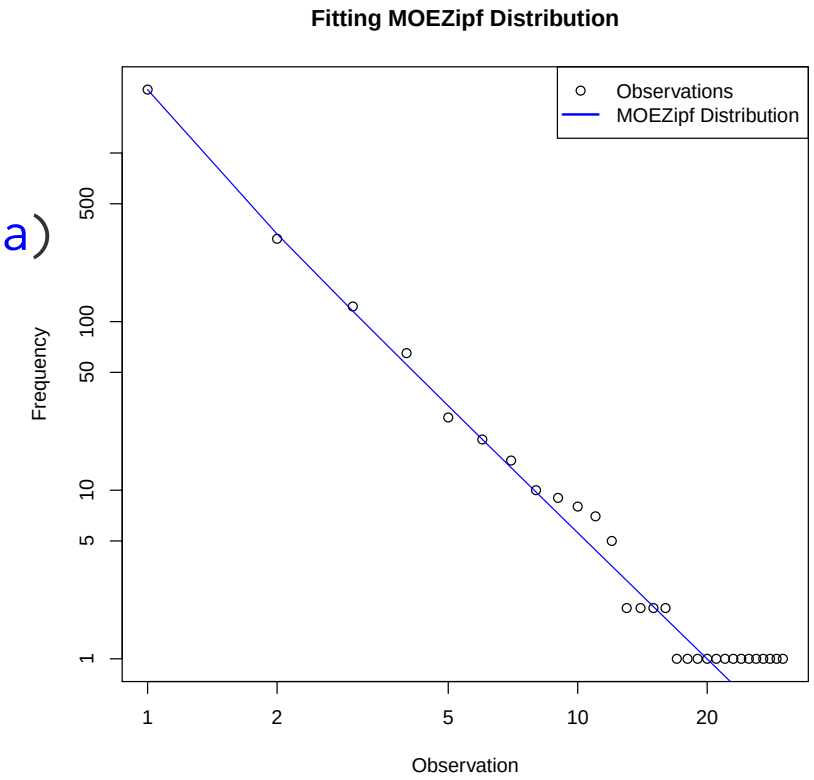
```
betas <- seq(0.2, 50, length = 30)
expectedVal <-c(); varianceVal <-c()
for(b in betas){
  expectedVal <- c(expectedVal, moezipfR.mean(alpha = 3.5,
      beta = b))
  varianceVal <- c(varianceVal, moezipfR.var(alpha = 3.5,
      beta = b))
}
```





Principales funcionalidades

```
data <- rmoezipf(3000, alpha =
    2.5, beta = 0.75)
data <-
    moezipfR.utils.getDataMatrix(data)

estimation <- moezipfR.fit(data,
    init_alpha = 1.5, init_beta =
    0.3, lower = c(1.001, 0.001),
    method = 'L-BFGS-B',
    show.plot = T)
estimation$alpha
[1] 2.479003
estimation$beta
[1] 0.7517179
```



Principales funcionalidades

```
intervals <- moezipfR.confint(data, alpha =
    estimation$alpha, beta = estimation$beta, level = 0.95)
> print(sprintf('alpha = %s Confidence Intervals (%s,
    %s)', round(estimation$alpha, 2),
    round(intervals$alpha.lowerB, 2),
    round(intervals$alpha.upperB, 2)))
[1] "alpha = 2.48 Confidence Intervals (2.33, 2.63)"
> print(sprintf('beta = %s Confidence Intervals (%s, %s)',
    round(estimation$beta, 2),
    round(intervals$beta.lowerB, 2),
    round(intervals$beta.upperB, 2)))
[1] "beta = 0.75 Confidence Intervals (0.6, 0.9)"
```

Caso de Uso: Redes reales

El data set Patents [1] incluye todas las citas entre patentes (1975 - 1999, EEUU).

Principales características:

Nodos	3774768
Aristas	16518948

Grado de un nodo: Número de conexiones.

[1] https://snap.stanford.edu/data/cit-Patents.html

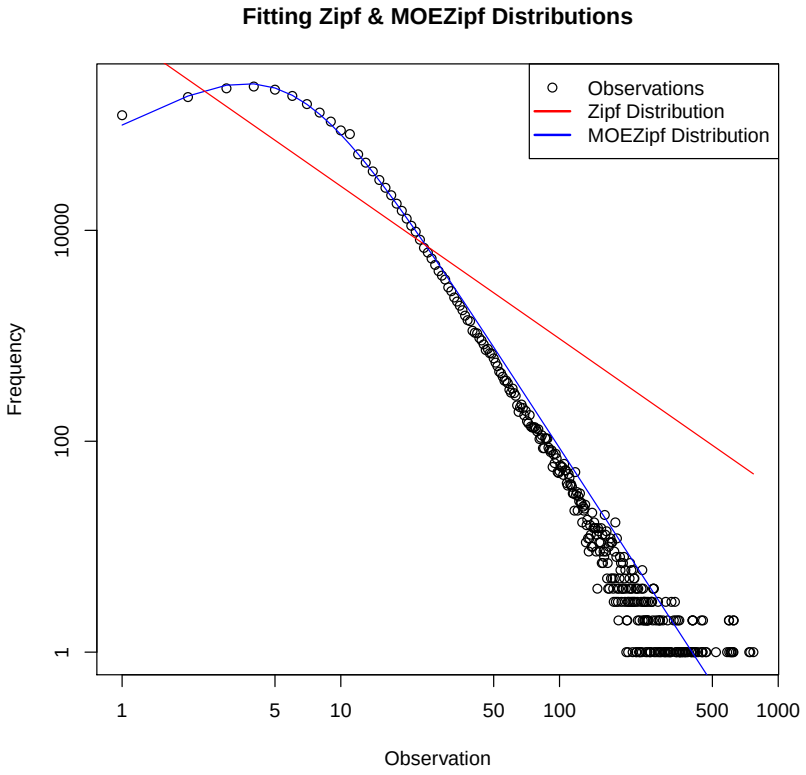
Caso de Uso: Redes reales

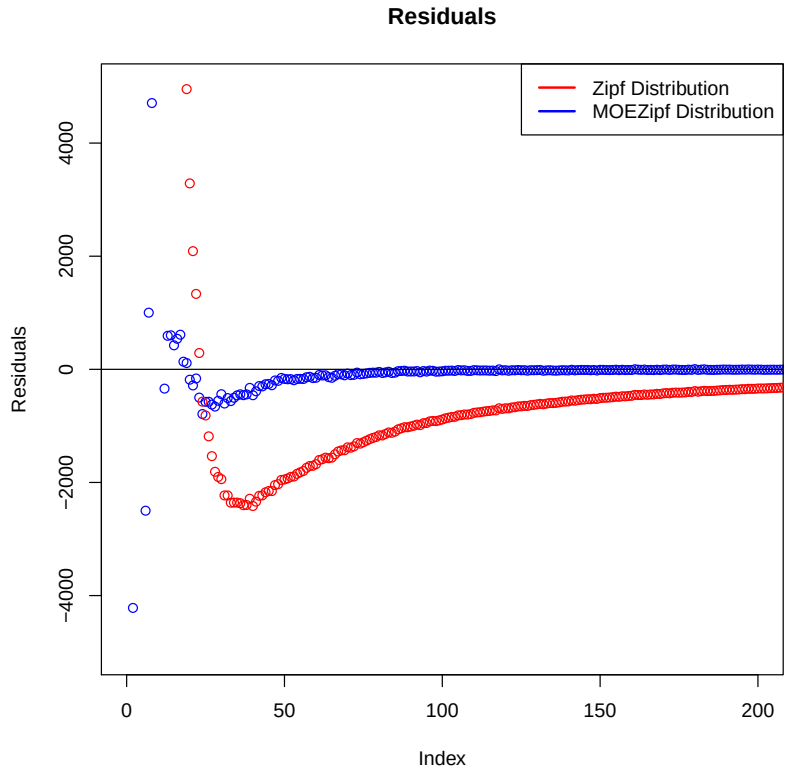
Ajuste de la distribución de grado.

```
data <- read.table(file = './outPatentsDegrees.txt',
    header=F)
data <- moezipfR.utils.getDataMatrix(data$V1)
estimation <- moezipfR.fit(data, init_alpha = 1.001,
    init_beta = 0.001, show.plot = T)
conInt <- moezipfR.confint(data, alpha = estimation$alpha,
    beta = estimation$beta)
[1] "alpha = 3.2 Confidence Intervals (3.19, 3.2)"
[1] "beta = 119.2 Confidence Intervals (118.45, 119.95)"
```

Caso de Uso: Redes reales

A la izquierda el ajuste de la distribuciones $Zipf(x; \alpha = 1.45)$ y $MOEZipf(x; \alpha = 3.2, \beta = 119.2)$, a la derecha los residuales de ambas distribuciones.





Bibliografía

[1] Aina Casellas Torrentó. La distribució zipf estesa segons la transformació marshall-olkin. 2013.

[2] Ariel Duarte-López, Arnau Prat-Pérez, and Marta Pérez-Casany. Using the marshall-olkin extended zipf distribution in graph generation. In *European Conference on Parallel Processing*, pages 493–502. Springer, 2015.

[3] Yeşim Güney, Yetkin Tuaç, and Olcay Arslan. Marshall–olkin distribution: parameter estimation and application to cancer data. *Journal of Applied Statistics*, pages 1–13, 2016.

[4] Albert W Marshall and Ingram Olkin. A new method for adding a parameter to a family of distributions with application to the exponential and weibull families. *Biometrika*, 84(3):641–652, 1997.

[5] Marta Pérez-Casany and Aina Casellas. Marshall-olkin extended zipf distribution. *arXiv preprint arXiv:1304.4540*, 2013.

Agradecimientos

Los autores quieren agradecer a Oracle Labs por el apoyo al proyecto Graphalytics y al apoyo de EC FP-7 a través del proyecto LDBC.

Ariel Duarte-López agradece la colaboración de la Agència de Gestió d' Ajuts Universitaris i de Recerca (AGAUR). Grant FI-DGR 2016.

Marta Pérez-Casany agradece al Ministerio de Ciencia e Inovación (España). Grant No. MTM2013-43992-R.

Muchas Gracias!!!