Ejemplo de análisis de Displex

En este artículo se presenta una implementación sobre el sistema de análisis estadístico R para el estudio de la disponibilidad léxica. Se ha optado por hacer la herramienta disponible mediante el repositorio GitHub.com, que permite la creación de adaptaciones y permitirá que este proyecto pueda avanzar, haciendolo disponible publicamente, tanto a nivel de uso como de implementación.

Se ha optado por hacer una exposición que, simultáneamente, sea un ejemplo de uso. Creemos que de esta forma la exposición se hará más amena y útil para el que quiera aplicar las herramientas que proporcionamos.

Instalación

Como se comenta arriba, se ha optado por el uso del repositorio GitHub.com. Esto implica que el usuario debe llevar a cabo un pequeño paso de instalación que, en nuestra opinión, no es más complejo que cualquier proceso de instalación en el sistema R. Este paso ha de hacerse únicamente cuando se pretenda instalar o actualizar el paquete en el sistema. Puesto que nuestra intención es seguir trabajando en el paquete, sería recomendable realizar esta acción de forma periódica.

Las siguientes órdenes instalan el paquete Displex del repositorio GitHub. Todo el código está implementado en R, con lo que es fácil de revisar.

```
# install.packages("devtools")
```

```
#library(devtools)
#install_github("jmss70/displex")
```

Se recomienda el uso del universo TidyVerse para el análisis de datos, ya que proporciona herramientas con una sintaxis muy potente que facilita enormemente la tarea de análisis y representación de datos.

```
# install.packages("tidyverse")
```

Llevados a cabo los pasos anteriores, y si no se ha producido ningún contratiempo, el sistema está preparado para llevar a cabo el trabajo.

Carga de las librerías y los datos

Para poder usar las librerías hay que cargarlas en nuestra sesión de trabajo, poniendo a nuestra disposición las funciones que proporcionan:

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(displex)
```

Para cargar los datos, se espera que estén en un determinado formato, utilizado por ser el que habitualmente se encuentra en este tipo de estudios, creemos que por razones históricas. Sin embargo, consideramos que son redundantes y que habría que estudiar, en un futuro cercano, establecer un estandard de codificación que sea más coherente con los modelos de datos normalizados.

Se espera que los datos estén en un archivo de texto, con campos separados por espacios: - Un campo de información del hablante - Un campo de identificación de usuario - Un campo de identificación de centro de interés - Una lista de palabras separadas por comas y en orden de realización

Un ejemplo de dos líneas sería:

```
21131 001 01 mano, pie, brazo, cerebro, pulmón, nariz, extremidad, ojo, boca, diente, pelo, oreja, culo
12131 002 01 riñón, corazón, garganta, cabeza, pierna, pie, hígado, estómago, mano, brazo, antebrazo, al
```

Suponiendo que tenemos todos los datos cargados en un archivo, denominado `datos.txt`, que estará alojado en el mismo directorio que el script de procesamiento, se podrían cargar los datos como:

```
d <- read.displex("datos.txt")
head(d)
```

```
##   infos users centers        words
## 1 21131   001      01 mano, pi....
## 2 12131   002      01 riñón, c....
## 3 12213   003      01 brazo, m....
## 4 22214   004      01 brazo, o....
## 5 12214   005      01 cabeza, ....
## 6 22213   006      01 pie, man....
```

Para cada usuario y centro de interés se carga una lista de palabras en el orden en el que se ha realizado

```
d[1,]$words
```

```
## [[1]]
##  [1] "mano"       "pie"        "brazo"      "cerebro"    "pulmón"
##  [6] "nariz"      "extremidad" "ojo"        "boca"       "diente"
## [11] "pelo"       "oreja"      "culo"       "vagina"
```

Modelo de datos

Diversos trabajos han propuesto modelos que se han ido refinando progresivamente para responder a la cuestión de la relevancia del vocabulario.Tras estudiar el modelo subyacente a tales aproximaciones, se propone una generalización para la construcción de un modelo general que permita representar de la forma más ajustada y abierta posible la evaluación de los modelos de disponibilidad.

La fórmula de cálculo de disponibilidad más comunmente aceptada es la propuesta por López-Strassburguer, que se evalúa como:

$$D(P_j) = \sum_{i=1}^{n} e^{-2.3\frac{i-1}{n-1}} \frac{f_{ij}}{I_1}$$

donde n corresponde a la máxima posición alcanzada, i representa cada posición alcanzada por un término, j es índice del término a evaluar, f_{ij} corresponde al número de veces que aparece el término j-ésimo en la posición i-ésima, e es el número de Euler (2.718282...), I_1 es el número de informantes. Obsérvese que, una vez recogidos los datos, los únicos valores variables durante la evaluación son j (índice del término), i (cada posición) y f_{ij} (las veces que aparece el término j-ésimo en la posición i-ésima).

Mediante unas operaciones de álgebra simple, se puede reescribir la expresión como:

$$D(P_j) = \sum_{i=1}^{n} e^{-2.3\frac{i-1}{n-1}} \frac{f_{ij}}{I_1} = \frac{1}{I_1} \sum_{i=1}^{n} e^{-2.3\frac{i-1}{n-1}} f_{ij} = \frac{1}{I_1} \sum_{i=1}^{n} \sum_{k=1}^{f_{ij}} e^{-2.3\frac{i-1}{n-1}}$$

Así, la fórmula de López-Strassburger se puede reinterpretar como la evaluación mediante la función $e^{-2.3\frac{i-1}{n-1}}$ de la aparición de cada término en cada posición, donde i es la posición del término, y n, I_1, e y -2.3 son valores fijos. Las evaluaciones de cada término en las diferentes posiciones se acumulan para todas las veces que el valor aparece en esa posición, $\sum_{k=1}^{f_{ij}}$, y en todas las posiciones, $\sum_{i=1}^{n}$, normalizándose posteriormente el valor mediante un valor fijo, que es el número de informantes, I_1.

Por tanto, la fórmula de López-Strassburger corresponde a una evaluación de cada término en cada posición en la que aparece, entre los distintos participantes del experimento, y un procedimiento de aglutinación de esa información para proporcionar una cualificación global para el término.

En un trabajo previo, se propuso un método alternativo para el cálculo de la disponibilidad. Se procedió a evaluar cada realización de cada término por cada hablante mediante una función de Zipf-Mandelbrot

$$\frac{k}{i}$$

con un parámetro k a determinar (hay en preparación un trabajo en el que se discuten estos parámetros, su función y necesidad) y se integraba la información proporcionada por cada muestra a las ya evaluadas mediante una ley probablística:

$$a + b - a \cdot b$$

que, generalizada a todo el experimento y expresada de forma equivalente a la fórmula de López-Strassburger, quedaría como:

$$D(P_j) = 1 - \Pi_{i=1}^{n} \Pi_{k=1}^{f_{ij}} \left(1 - \frac{k}{i}\right),$$

Como se puede observar, el modelo general es equivalente al propuesto por López-Strassburger. Lo que cambia son los procedimientos de evaluación e integración de las realizaciones de los hablantes. La razón para proponer una evaluación basada en la teoría de los conjuntos difusos es la de proporcionar un marco teórico en la que integrar la evaluación realizada.

En lugar de proporcionar un modelo de evaluación prefijado, se va a proponer un conjunto de herramientas, con opciones prefijadas, que permitan la evaluación de diferentes aproximaciones. Aunque tengamos nuestras preferencias, debido a nuestra percepción personal del problema de la determinación de la disponibilidad léxica, creemos que es importante ofrecer herramientas flexibles que permitan evaluaciones con modelos alternativos que ayuden a discutir y desarrollar alternativas.

Para ello, se proporciona la función `displex_availability`, que permite integrar la información construida a partir de los datos cargados mediante la función `read.displex`, proporcionando una evaluación de la disponibilidad de cada término en cada centro de interés. Esta función es parametrizable para poder integrar diferentes modelos de evaluación y de aglutinación.

Con la librería se proporcionan tres modelos de evaluación de las realizaciones (valor asignado inicialmente a cada térmnino para cada hablante y centro de realización, en función de su posición):

- Una ley de Zipf-Mandelbrot generalizada, `displex_zipf_law`,

  $$\frac{k}{(i+d)^a},$$

  una ley tradicionalmente propuesta para el uso en la evaluación de la frecuencia del léxico y que se ha verificado experimentalmente en varias lenguas. El parámetro i es la posición del vocablo, d es un factor de desplazamiento que permite suavizar el inicio de la curva y el valor de k corresponde a un factor de normalización. Los valores iniciales de la función son $k = 1$, $d = 0$ y $a = 1$, con lo que correspondería a $1/i$.

- Una ley exponencial generalizada, `displex_exp_law`,

$$\frac{k}{a^{i+d}}$$

  donde los valores por defecto son $a = 2$, $k = 1$ y $d = 0$, correspondiendo entonces a $\frac{1}{2^i}$.

Así mismo, se proporciona un un operador de agregación específico para la suma probabilística, `displex_additive_law`, aunque se podría utilizar cualquier operador de agregación que tenga disponible R, como `sum`, `mean`, etc.

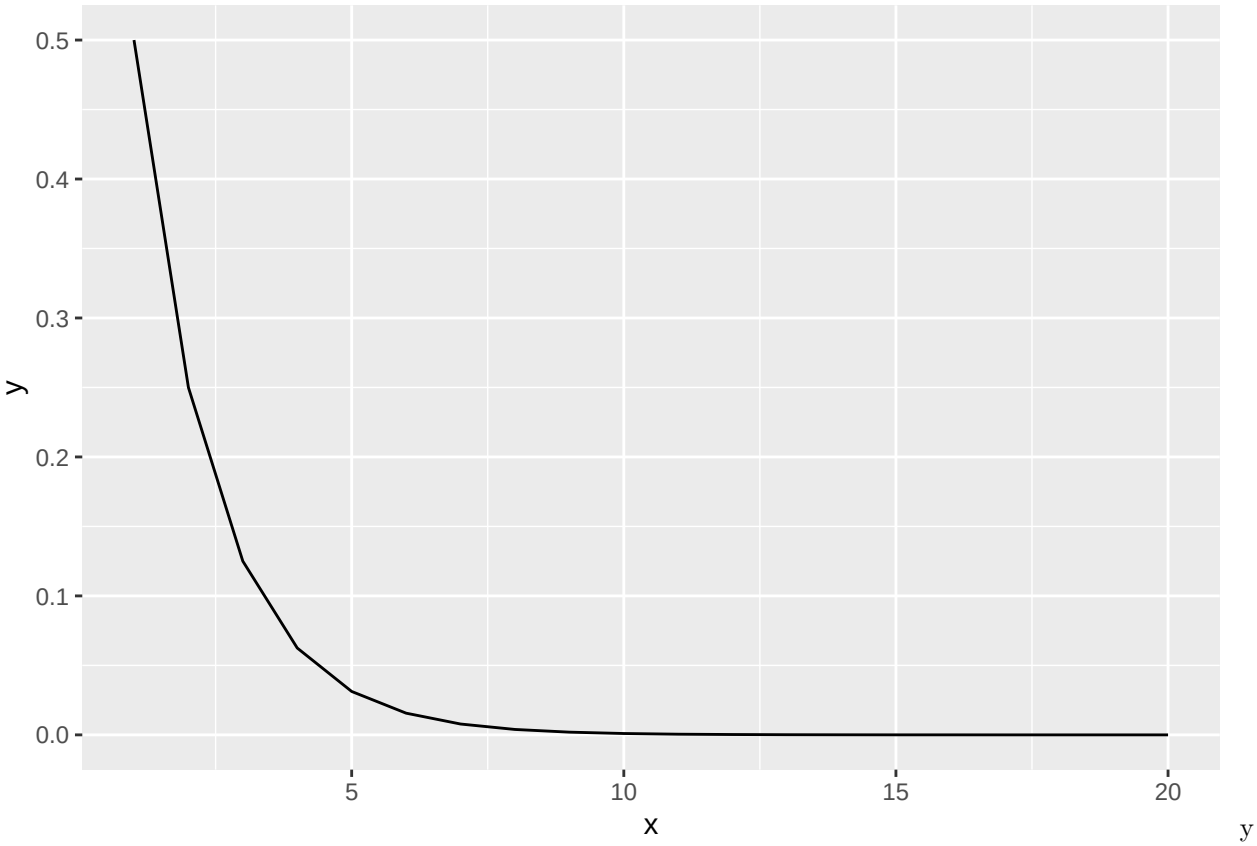
En los primeros ejemplos se considerará únicamente un centro de interés, para facilitar la representación. Más adelante se desarrollarán ejemplos con diversos centros de interés.

```
d <- d %>% filter(centers=="01")
```

Ejemplo 1. Modelo de aplicación simple: evaluación exponencial e integración mediante suma probabilística

Por defecto, se utiliza una ley exponencial $1/2^i$:

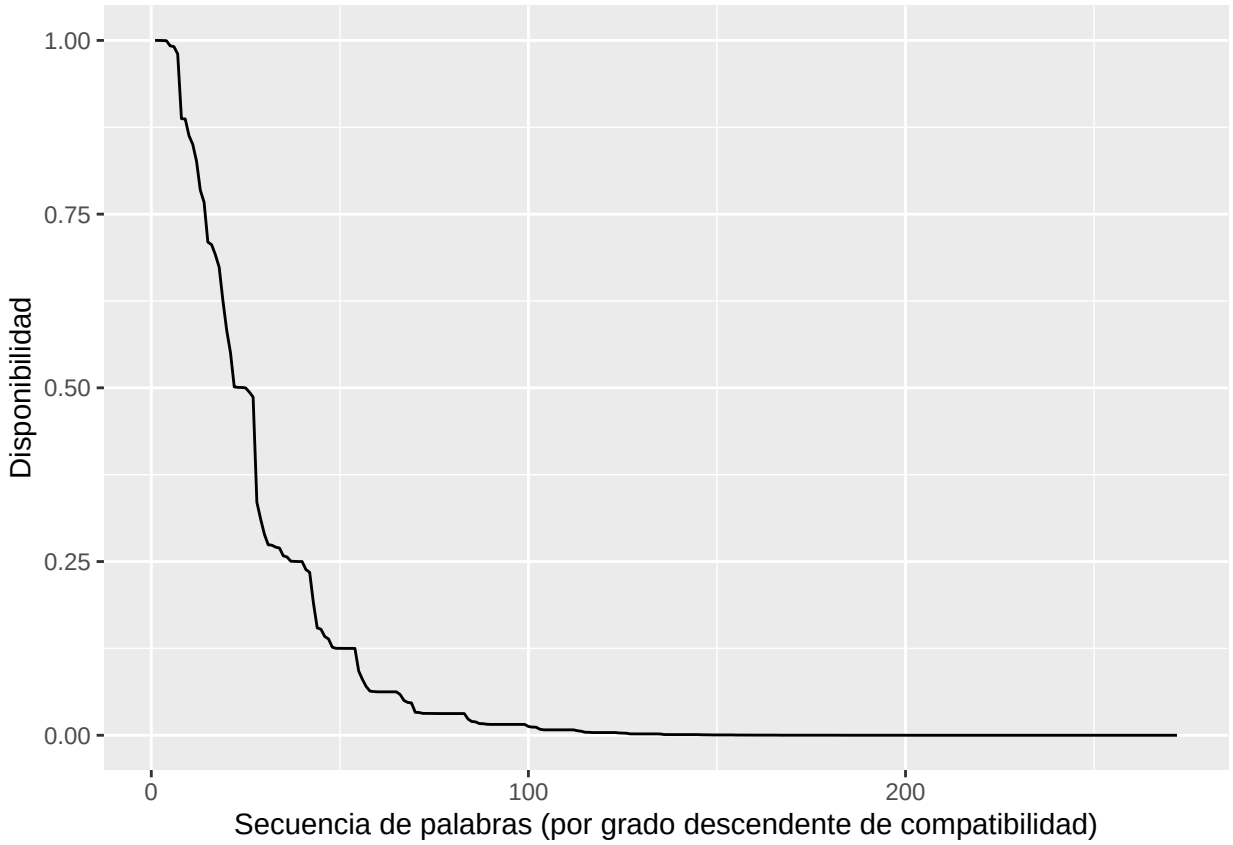
```
data.frame(x=1:20) %>%
  mutate(y=displex_exp_law(x)) %>%
  ggplot(aes(x=x,y=y)) + geom_line()
```



se integra las distintas valoraciones mediante una ley de suma probabilística:

```
expYprob <- displex_availability(d)
expYprob %>%
  arrange(-availability) %>%
  ggplot(aes(x=seq_along(words),y=availability)) + geom_line() +
```

```
  xlab("Secuencia de palabras (por grado descendente de compatibilidad)") +
  ylab("Disponibilidad")
```



Si seleccionamos los elementos más representativos y los que menos:

```
expYprob %>%
  arrange(-availability) %>%
  head(10) %>%
  select(words, availability)
```

```
## # A tibble: 10 x 2
##    words   availability
##    <chr>          <dbl>
##  1 cabeza         1.00
##  2 brazo          1.00
##  3 ojo            1.00
##  4 mano           1.00
##  5 pierna         0.992
##  6 pie            0.991
##  7 corazón        0.981
##  8 pelo           0.887
##  9 tronco         0.887
## 10 hueso          0.863
```

Ejemplo 2: Modelo de López-Strassburger

El modelo propuesto por López-Strassburger se basa en el recuentode las distintas posiciones que alcanza cada término en las listas:

```
# Recopilamos todas las palabras y sus posiciones en las listas
words = c()
pos   = c()
for (i in seq_along(d$words)) {
  words = c(words,d$words[i][[1]])
  pos   = c(pos,seq_along(d$words[i][[1]]))
}

# Organizamos las palabras y sus posiciones en una tabla
x <- table(words,pos)
x[c('cabeza','pierna','abdomen'),]
```

```
##          pos
## words      1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
##   cabeza  17  5  7  3  1  1  1  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   pierna   2  8  3  8  6  2  2  3  0  3  1  3  0  1  0  1  0  1  0  0  2  0  1
##   abdomen  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0  0  0  0  0
##          pos
## words     24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
##   cabeza   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   pierna   0  1  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0
##   abdomen  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##          pos
## words     47
##   cabeza   0
##   pierna   0
##   abdomen  0
```

Construyendo las constantes de la evaluación

```
# Máxima posición alcanzada
n <- max(vapply(d$words, function(x) {length(x)}, FUN.VALUE=1L))
# Número total de hablantes
N <- length(unique(d$users))
```

se obtiene que la posición máxima es $n = 47$ y el número de hablantes es $I_1 = 72$.

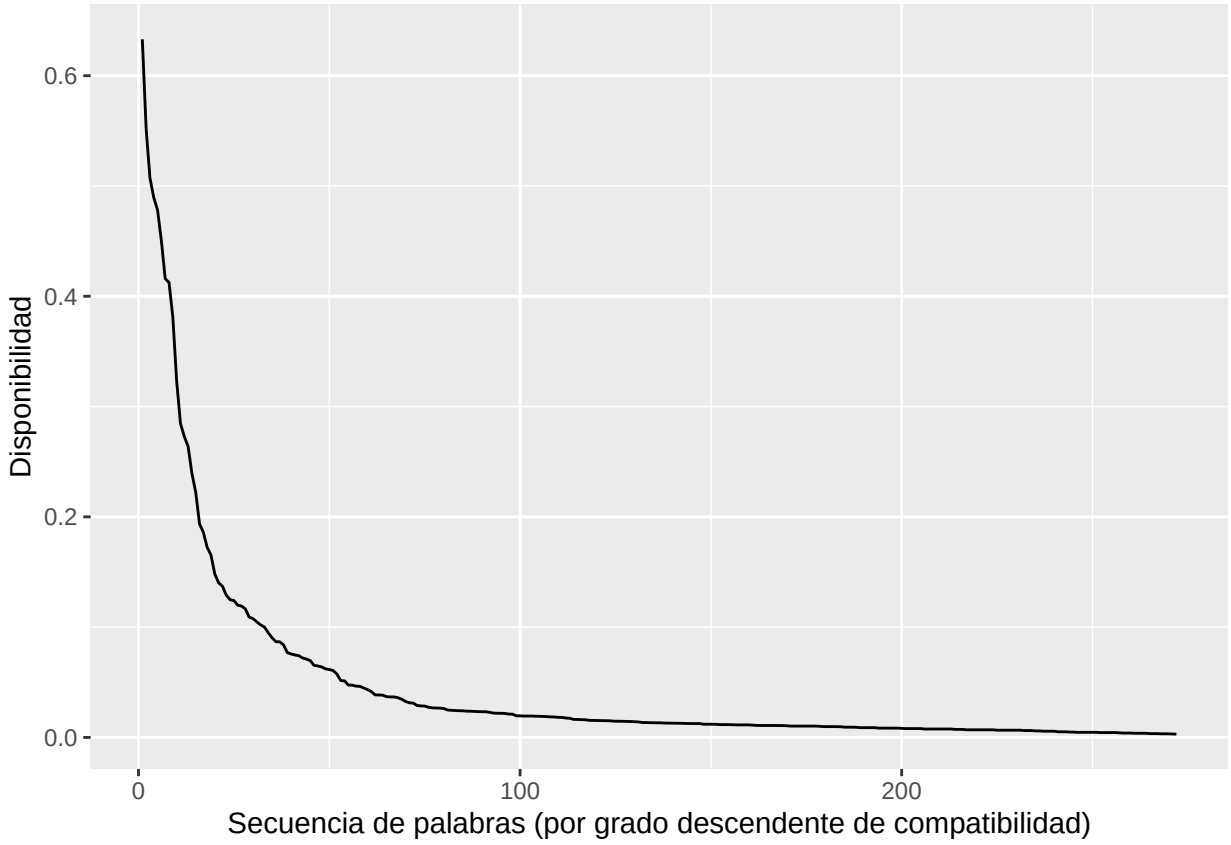
Esta información se integra mediante una ponderación por factores exponenciales, dando lugar a

```
m <- exp(-2.3*(seq_along(1:n) - 1) / (n-1))
lopezstrass <- apply(x,1,function(a) {sum(a * m / N)})
lopezstrass <- data.frame(words=names(lopezstrass), availability =lopezstrass)

lopezstrass %>%
  arrange(-availability) %>%
  select(words,availability)  %>%
  head(10) %>%
  select(words,availability)
```

```
##           words availability
## ojo         ojo    0.6329875
## brazo     brazo    0.5518081
## pierna   pierna    0.5072154
## mano       mano    0.4894822
## cabeza   cabeza    0.4780536
## pie         pie    0.4501276
```

```
## nariz     nariz    0.4159706
## dedo       dedo    0.4125253
## corazón corazón    0.3810710
## boca       boca    0.3231570
```

```
lopezstrass %>%
  arrange(-availability) %>%
  ggplot(aes(x=seq_along(words),y=availability)) + geom_line() +
  xlab("Secuencia de palabras (por grado descendente de compatibilidad)") +
  ylab("Disponibilidad")
```



Como se ha discutido anteriormente, el modelo de López-Strassburger se puede evaluar como un modelo exponencial:

$$e^{-2.3\frac{i-1}{n-1}} = \frac{1}{e^{2.3\frac{i-1}{n-1}}} = \frac{1}{\left(e^{\frac{2.3}{n-1}}\right)^{i-1}}$$

por lo que se puede modelar mediante una ley exponencial de base $a = e^{\frac{2.3}{n-1}}$ y desplazamiento $d = -1$, y utilizando como función de agregación la suma, dividida por el número de hablantes:

```
lopez_law <- function(w) {
  displex_exp_law(w,a=exp(2.3/(n-1)),d=-1)
}

lopez_reduce <- function(x) {
  sum(x) / N
}

# Se calcula la disponibilidad de cada término aplicando el modelo exponencial de López y reduciendo me
```

```
lopezstrass_nm <- displex_availability(d,law=lopez_law,reduce=lopez_reduce)
lopezstrass_nm %>% arrange(-availability) %>% head(10) %>% select(words,availability)
```

```
## # A tibble: 10 x 2
##    words   availability
##    <chr>          <dbl>
##  1 ojo            0.633
##  2 brazo          0.552
##  3 pierna         0.507
##  4 mano           0.489
##  5 cabeza         0.478
##  6 pie            0.450
##  7 nariz          0.416
##  8 dedo           0.413
##  9 corazón        0.381
## 10 boca           0.323
```

Es posible verificar que ambas formas de calcular son virtualmente idénticas, observando las diferencias entre ellas:

```
inner_join(lopezstrass %>% select(words, availability),
           lopezstrass_nm %>% select(words, availability),
           by="words") %>%
  mutate(diff=availability.x - availability.y) %>%
  select(diff) %>%
  summarise(maxdiff=max(diff))
```

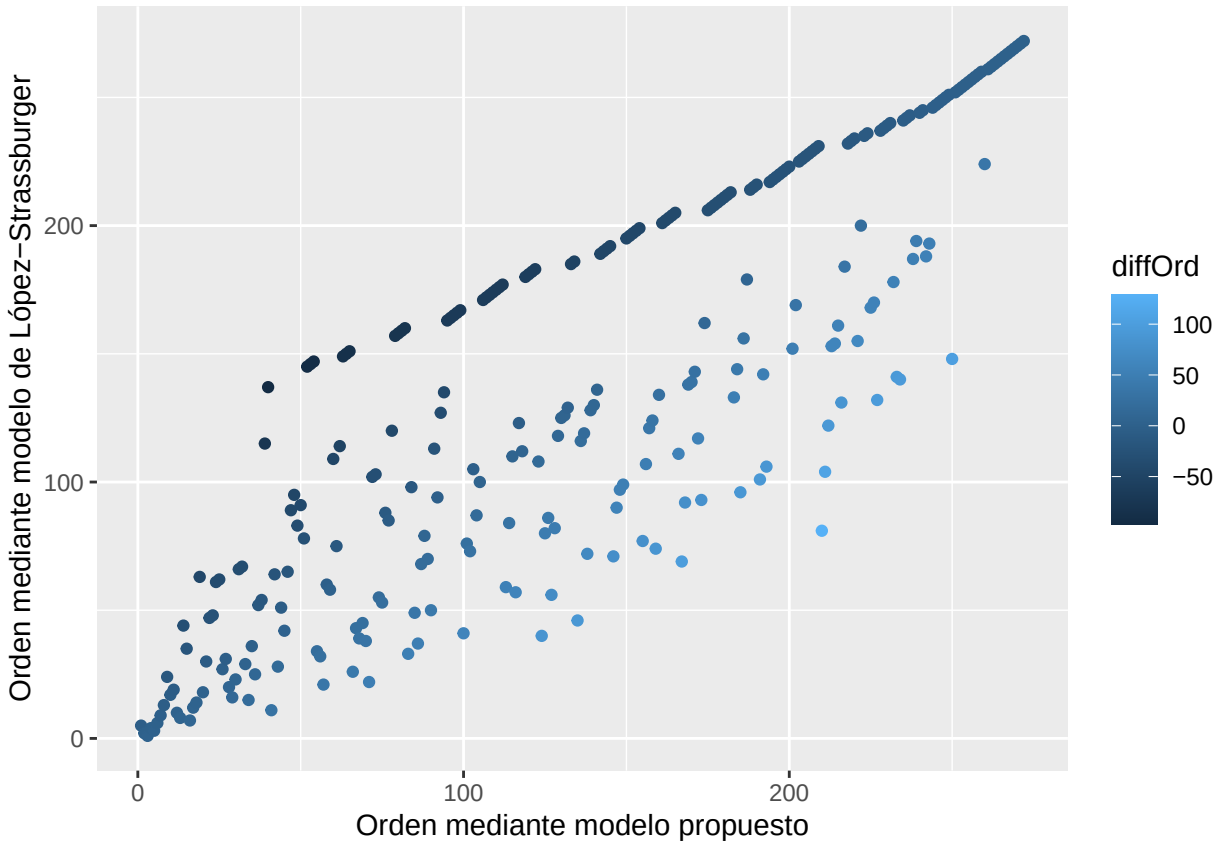
```
##        maxdiff
## 1 2.775558e-16
```

La diferencia máxima anda por el orden de 3 unidades, después de 15 ceros decimales. Lo cual es justificable por los pequeños defectos de aproximación que se realizan durante el proceso.

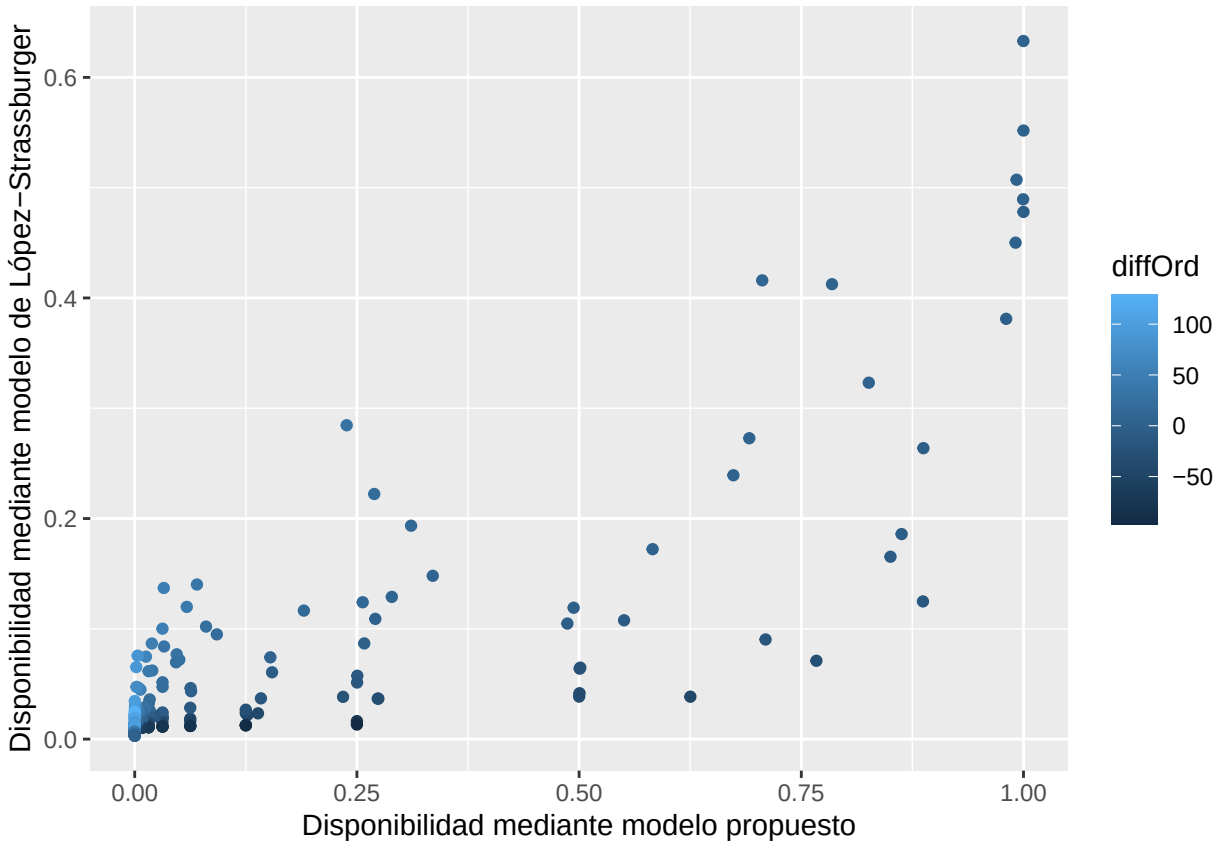
Por tanto, el procedimiento de López-Strassburger es un caso particular de aplicación de una ley exponencial, en este caso de base $a = e^{\frac{2.3}{n-1}} = 1.0512711$, y desplazamiento inicial $d = -1$.

Puesto que el objetivo de los estudios de disponibilidad léxica es, primariamente, ordenar el léxico según su accesibilidad, realizando una comparativa de los modelos exponencial y de López-Strassburger, se puede observar que tieden a clasificar en los primeros y últimos lugares los mismos términos y, en general, se observa una tendencia lineal entre las dos medidas. Es de esperar que haya diferencias, puesto que son dos modelos con cuantificaciones distintas. Pero parece evidente que hay una fuerte relación entre los dos, puesto que su función es medir el mismo concepto.

```
analysisComp <-
inner_join(
  expYprob %>% arrange(-availability) %>%
    mutate(eOrd=seq_along(words)) %>% rename(eAvailability=availability),
  lopezstrass %>% arrange(-availability) %>%
    mutate(lOrd=seq_along(words)) %>% rename(lAvailability=availability),
  by="words") %>%
  mutate(diffOrd=eOrd - lOrd)

analysisComp %>%
  ggplot(aes(x=eOrd,y=lOrd)) + geom_point(aes(color=diffOrd)) +
  xlab("Orden mediante modelo propuesto") +
  ylab("Orden mediante modelo de López-Strassburger")
```




```
analysisComp %>%
  ggplot(aes(x=eAvailability,y=lAvailability)) + geom_point(aes(color=diffOrd)) +
  xlab("Disponibilidad mediante modelo propuesto") +
  ylab("Disponibilidad mediante modelo de López-Strassburger")
```



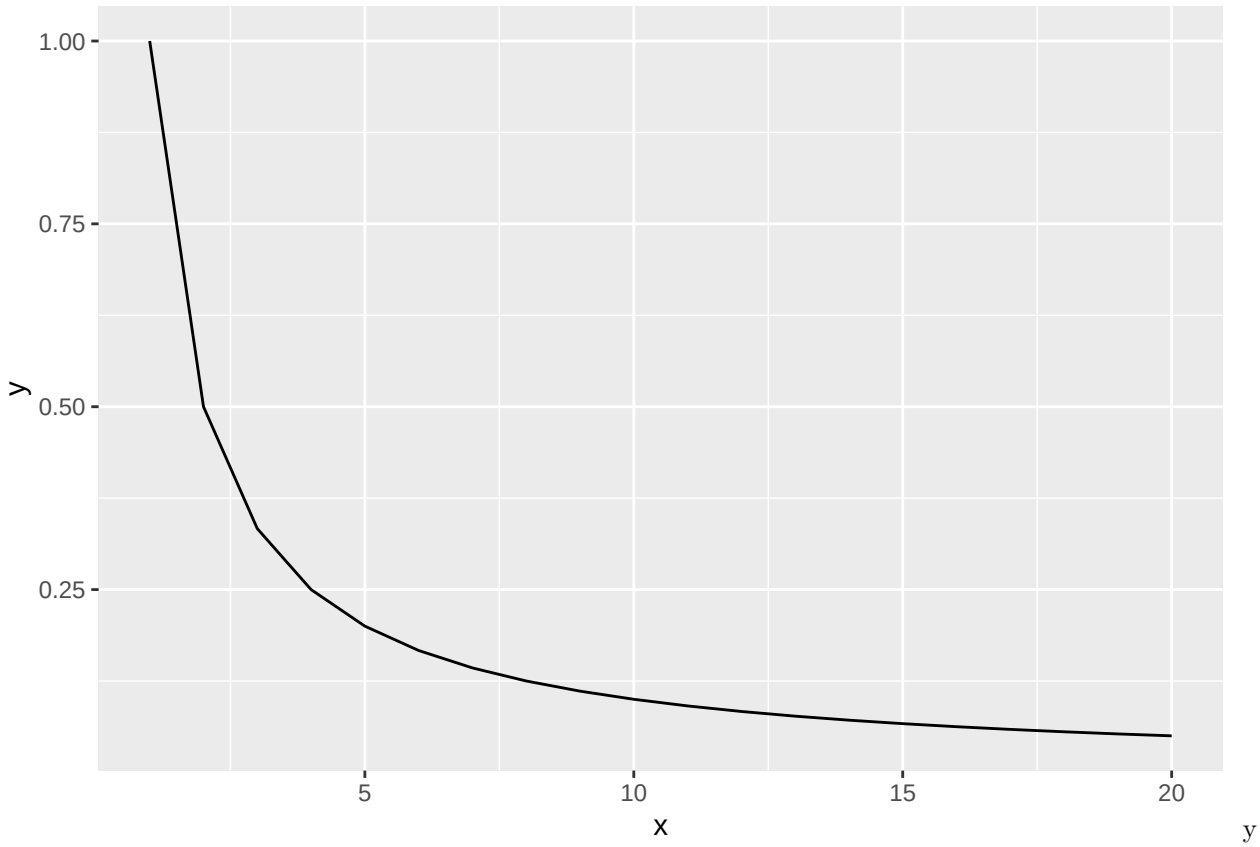
Comparando los valores de disponibilidad obtenidos se puede observar que el modelo con la suma probabilística tiende a expandir más las valoraciones que el modelo de López-Strassburger. El modelo basado en la suma probabilística, para cada nueva aparición de un término, acumula, aunque sea poco, su disponibilidad. Sin embargo, en el modelo basado en promedio de López-Strassburger, la aparición de un término en una posición posterior puede decrementar el promedio, llegándose a la paradógica situación en la que la aparición en una de las listas puede decrementar el valor de la disponibilidad.

Esta "paradoja" fue la que hizo que nos planteáramos la necesidad de un nuevo modelo de evaluación de la disponibilidad. Además, creemos que es importante que el modelo que se proponga tenga el respaldo de un marco teórico que permita posteriores desarrollos. A continuación se proponen otras posibilidades de evaluación, basados en el mismo macro modelo de evaluación individual de cada prueba y agregación de los valores.

Ejemplo 3: Ley de Zipf-Mandelbrot

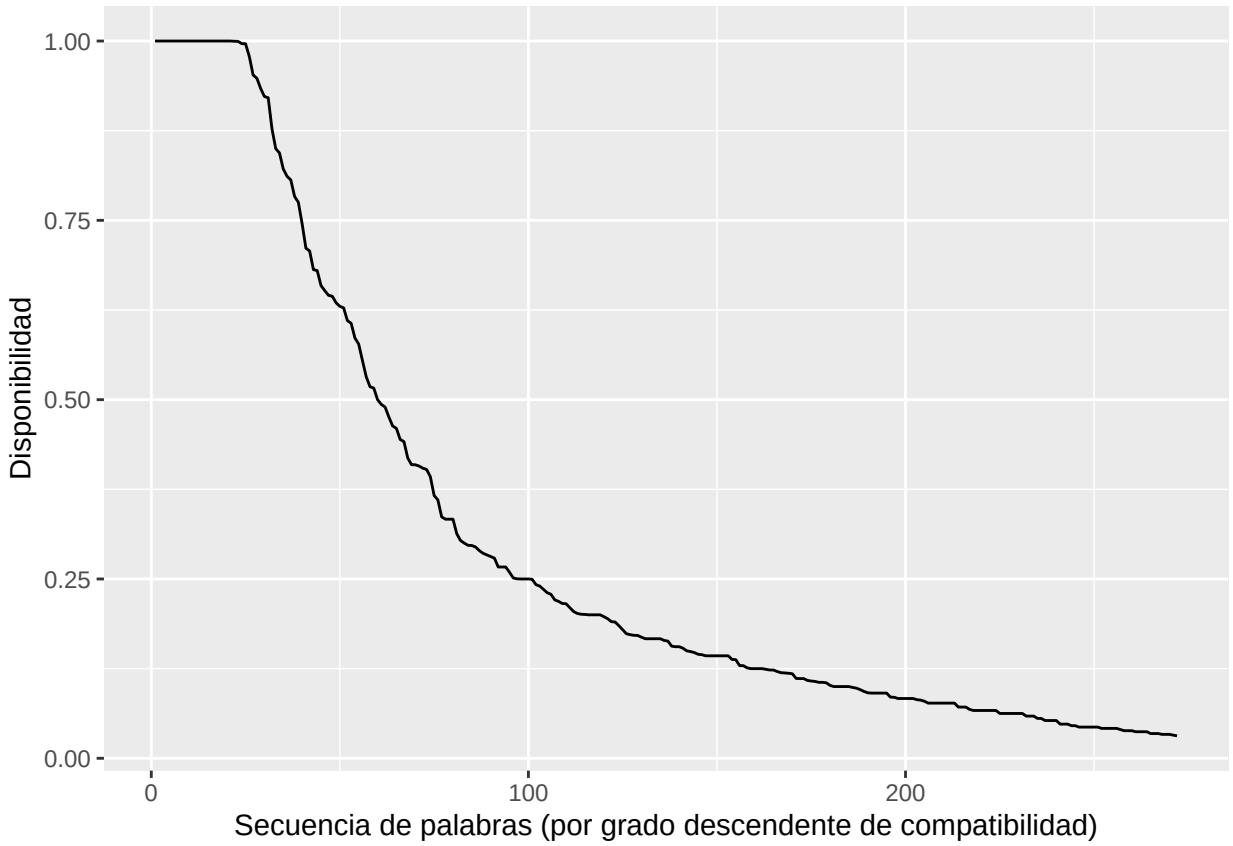
Partiendo de la ley de Zipf-Mandelbrot, expuesta anteriormente, que tiene la forma

```
data.frame(x=1:20) %>%
  mutate(y=displex_zipf_law(x)) %>%
  ggplot(aes(x=x,y=y)) + geom_line()
```



aplicando una ley de suma probabilística, se llega a una distribución como:

```
zipfYexp <- displex_availability(d,law=displex_zipf_law)

zipfYexp %>%
  arrange(-availability) %>%
  ggplot(aes(x=seq_along(words),y=availability)) + geom_line() +
  xlab("Secuencia de palabras (por grado descendente de compatibilidad)") +
  ylab("Disponibilidad")
```



Es posible observar que hay muchas valoraciones que son indistinguibles de uno.

```
zipfYexp %>%
  arrange(-availability) %>%
  head(10) %>% select(words, availability)
```

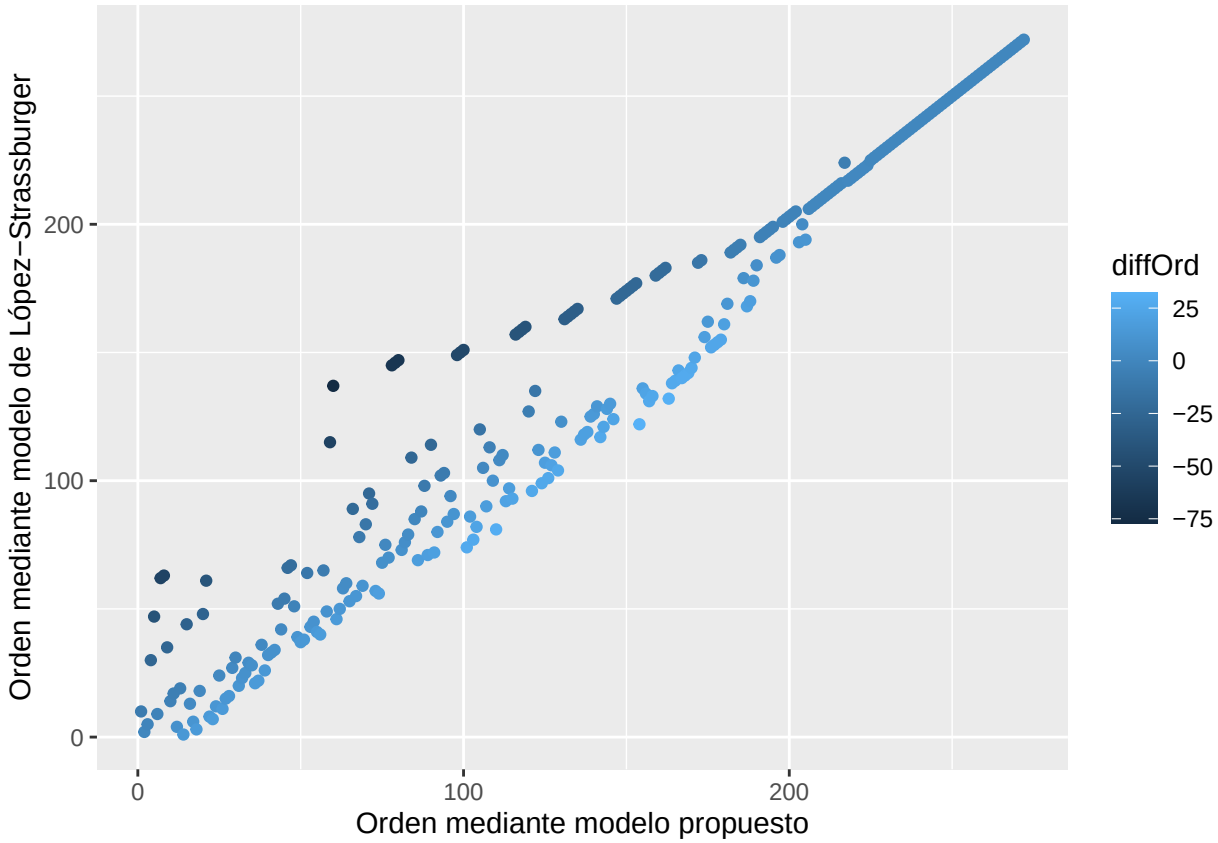
```
## # A tibble: 10 x 2
##    words      availability
##    <chr>             <dbl>
##  1 boca                  1
##  2 brazo                 1
##  3 cabeza                1
##  4 cerebro               1
##  5 cintura               1
##  6 corazón               1
##  7 enfermedad            1
##  8 esqueleto             1
##  9 extremidad            1
## 10 hígado                1
```

Esto se debe a que la ley de Zipf-Mandelbrot que hemos aplicado asigna, al primer término enunciado, el valor uno, y la ley probabilística nunca disminuye el valor. Esto lleva a que, una vez un término ha sido evaluado en una realización con el valor unidad, nunca decrementará.

Esta situación lleva a plantearse qué entendemos por una "buena representación" de un centro de interés. Un concepto que todos tenemos, más o menos, asimilado, pero que no está formalizado.

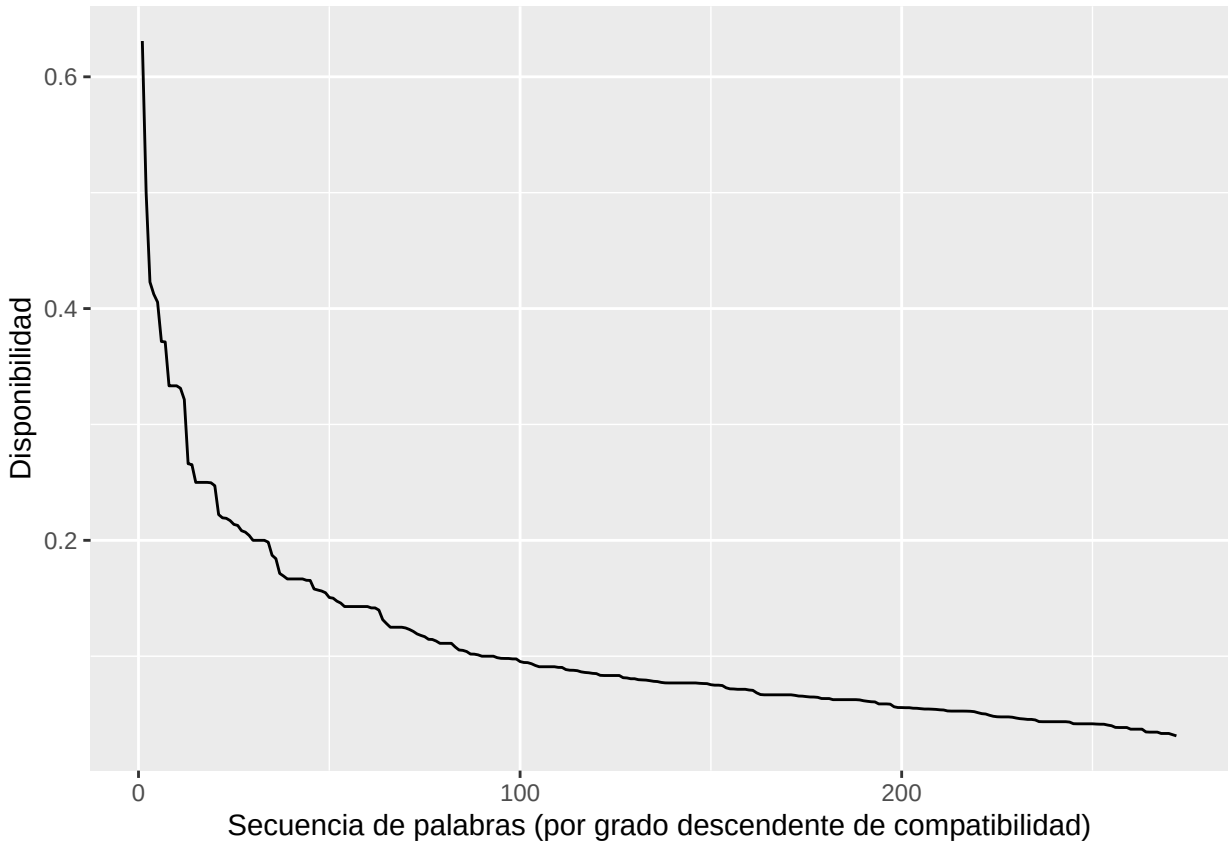
Analizando la comparativa con la de Lopez-Strassburger, se puede observar que la relación es mucho menos fuerte que en el caso estudiado anteriormente.

```
analysisComp <-
inner_join(
  zipfYexp %>% arrange(-availability) %>%
    mutate(eOrd=seq_along(words)) %>% rename(eAvailability=availability),
  lopezstrass %>% arrange(-availability) %>%
    mutate(lOrd=seq_along(words)) %>% rename(lAvailability=availability),
  by="words") %>%
  mutate(diffOrd=eOrd - lOrd)

analysisComp %>% ggplot(aes(x=eOrd,y=lOrd)) + geom_point(aes(color=diffOrd)) +
  xlab("Orden mediante modelo propuesto") +
  ylab("Orden mediante modelo de López-Strassburger")
```



Sin embargo, si en lugar de la suma probabilística se considera como agregación la media de los valores obtenidos para cada término, se obtiene la siguiente representación:

```
zipfYmean <- displex_availability(d,law=displex_zipf_law,reduce=mean)

zipfYmean %>%
  arrange(-availability) %>%
  ggplot(aes(x=seq_along(words),y=availability)) + geom_line() +
  xlab("Secuencia de palabras (por grado descendente de compatibilidad)") +
  ylab("Disponibilidad")
```

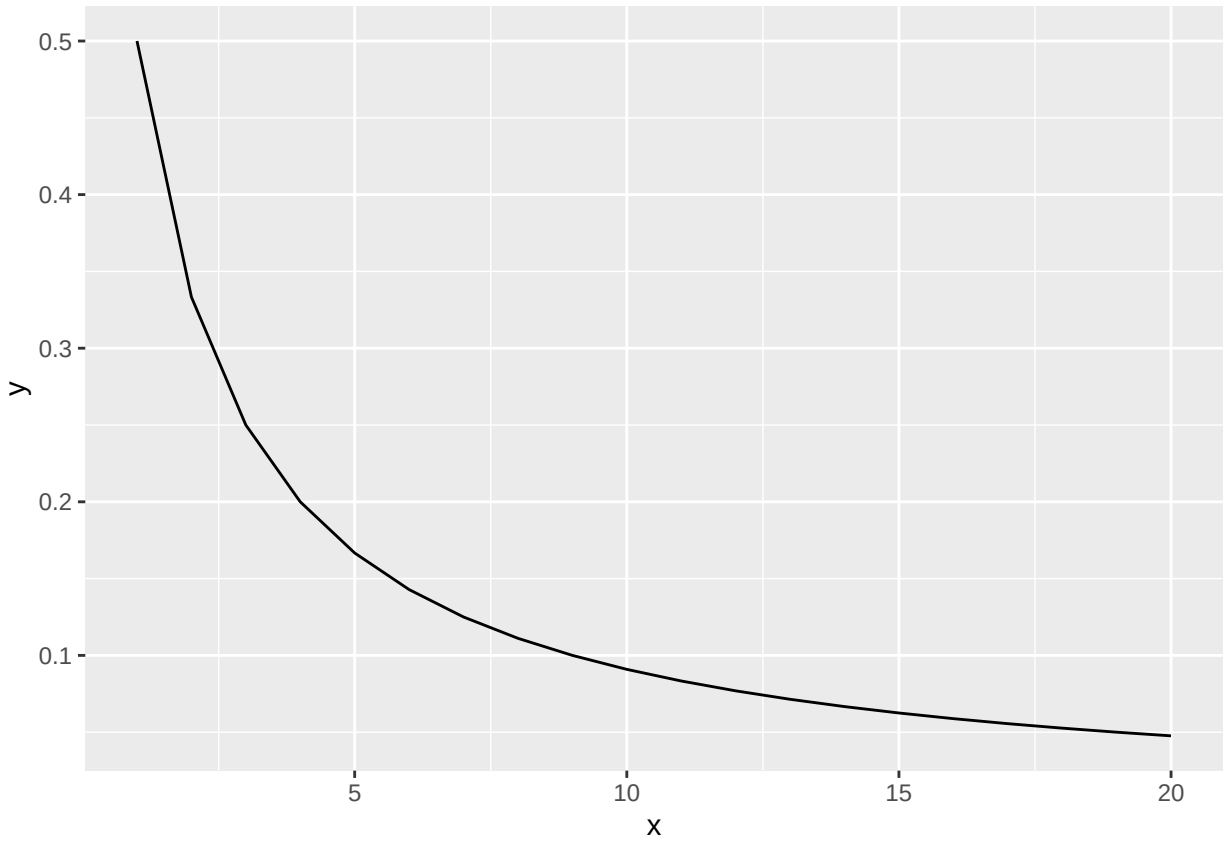


De nuevo se observa una relación poco clara con la medida de López-Strassburger:

```
analysisComp <-
inner_join(
  zipfYmean %>% arrange(-availability) %>%
    mutate(eOrd=seq_along(words)) %>% rename(eAvailability=availability),
  lopezstrass %>% arrange(-availability) %>%
    mutate(lOrd=seq_along(words)) %>% rename(lAvailability=availability),
  by="words") %>%
  mutate(diffOrd=eOrd - lOrd)

analysisComp %>% ggplot(aes(x=eOrd,y=lOrd)) + geom_point(aes(color=diffOrd)) +
  xlab("Orden mediante modelo propuesto") +
  ylab("Orden mediante modelo de López-Strassburger")
```

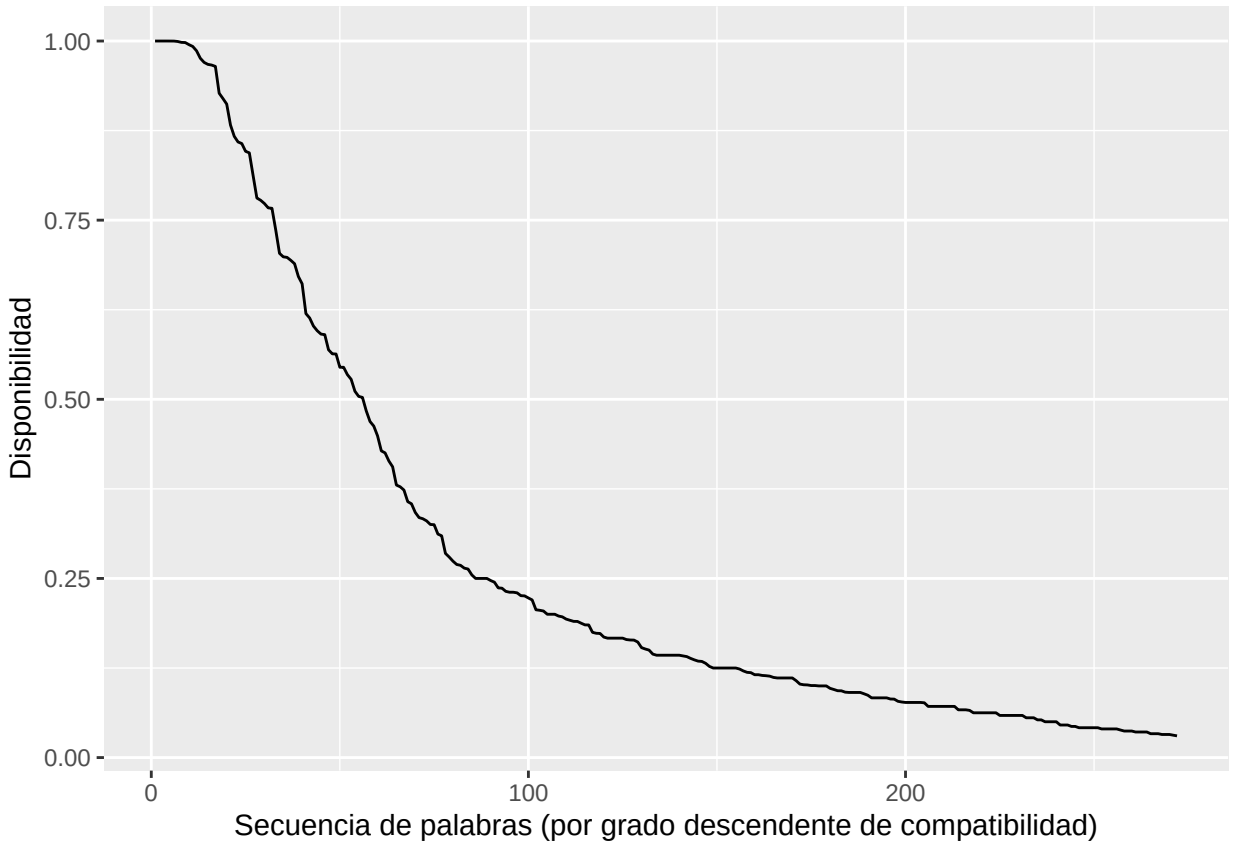
O se podría haber corregido la evaluación inicial, para evitar esa primera evaluación unidad:

```
data.frame(x=1:20) %>%
  mutate(y=displex_zipf_law(x, d=1)) %>%
  ggplot(aes(x=x,y=y)) + geom_line()
```



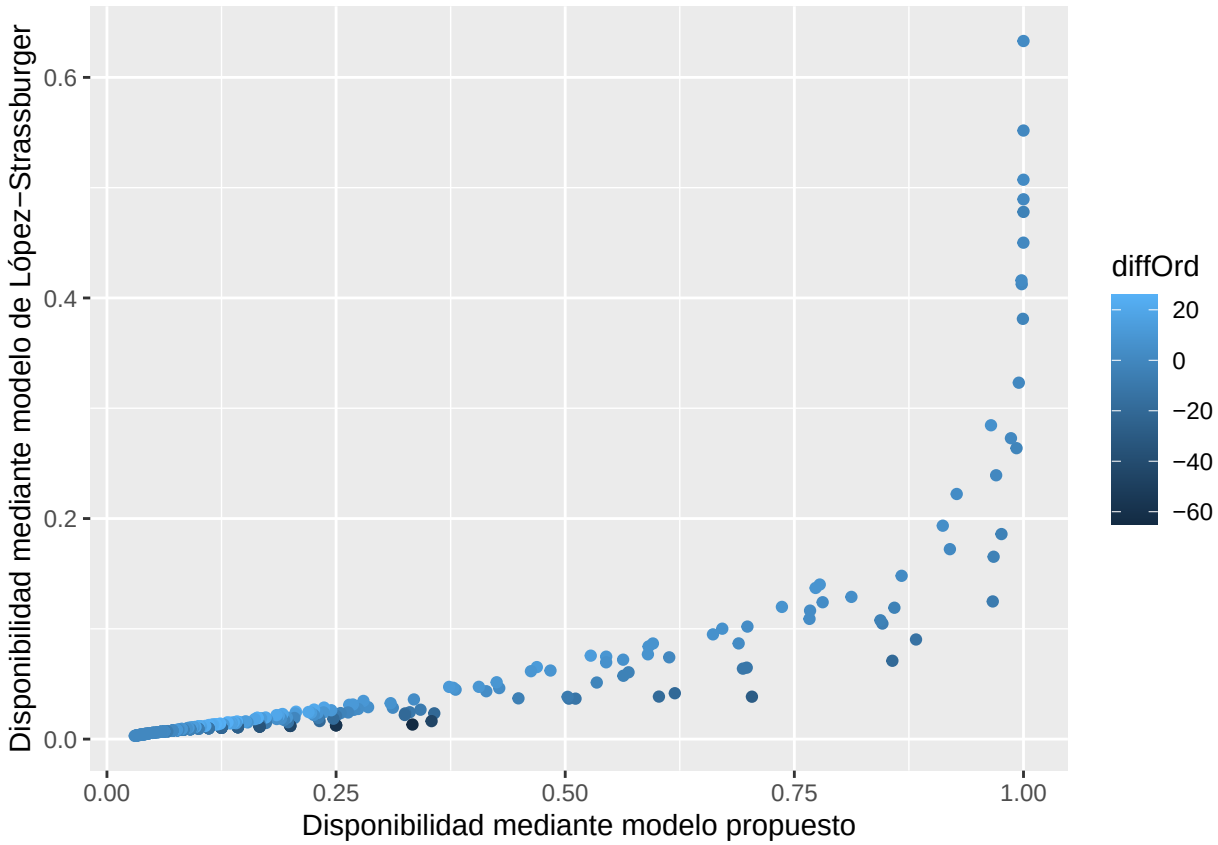
Obteniéndose una forma más conocida

```
avSanc <- displex_availability(d,
                               law=function(x) {displex_zipf_law(x,d=1)},
                               reduce=displex_additive_law)
avSanc %>%
  arrange(-availability) %>%
  ggplot(aes(x=seq_along(words),y=availability)) + geom_line() +
  xlab("Secuencia de palabras (por grado descendente de compatibilidad)") +
  ylab("Disponibilidad")
```



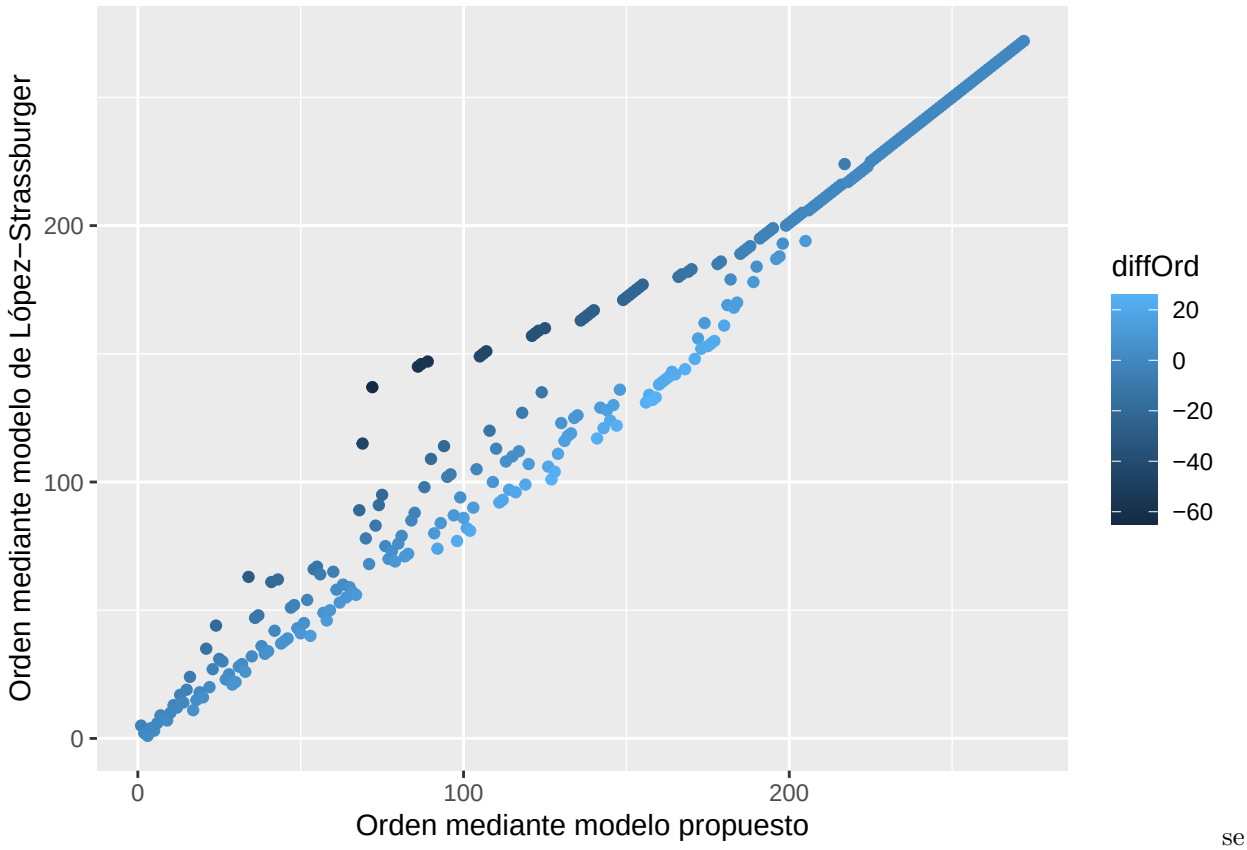
Analizando la relación entre la evaluación de López-Strassburger y esta posibilidad, se observa una clara dependencia no lineal.

```
analysisComp <-
inner_join(
  avSanc %>% arrange(-availability) %>%
    mutate(eOrd=seq_along(words)) %>% rename(eAvailability=availability),
  lopezstrass %>% arrange(-availability) %>%
    mutate(lOrd=seq_along(words)) %>% rename(lAvailability=availability),
  by="words") %>%
  mutate(diffOrd=eOrd - lOrd)

analysisComp %>%
  ggplot(aes(x=eAvailability,y=lAvailability)) + geom_point(aes(color=diffOrd)) +
  xlab("Disponibilidad mediante modelo propuesto") +
  ylab("Disponibilidad mediante modelo de López-Strassburger")
```



Este modelo establece una dispersión mayor de los datos, lo que clarifica su clasificación. Esto es debido al uso de la valoración mediante la función de Zipf-Mandelbrot, que tiene un descenso más suave que una exponencial. Sin embargo, si comparamos la ordenación de la clasificación:

```
analysisComp %>%
  ggplot(aes(x=eOrd,y=lOrd)) + geom_point(aes(color=diffOrd)) +
  xlab("Orden mediante modelo propuesto") +
  ylab("Orden mediante modelo de López-Strassburger")
```

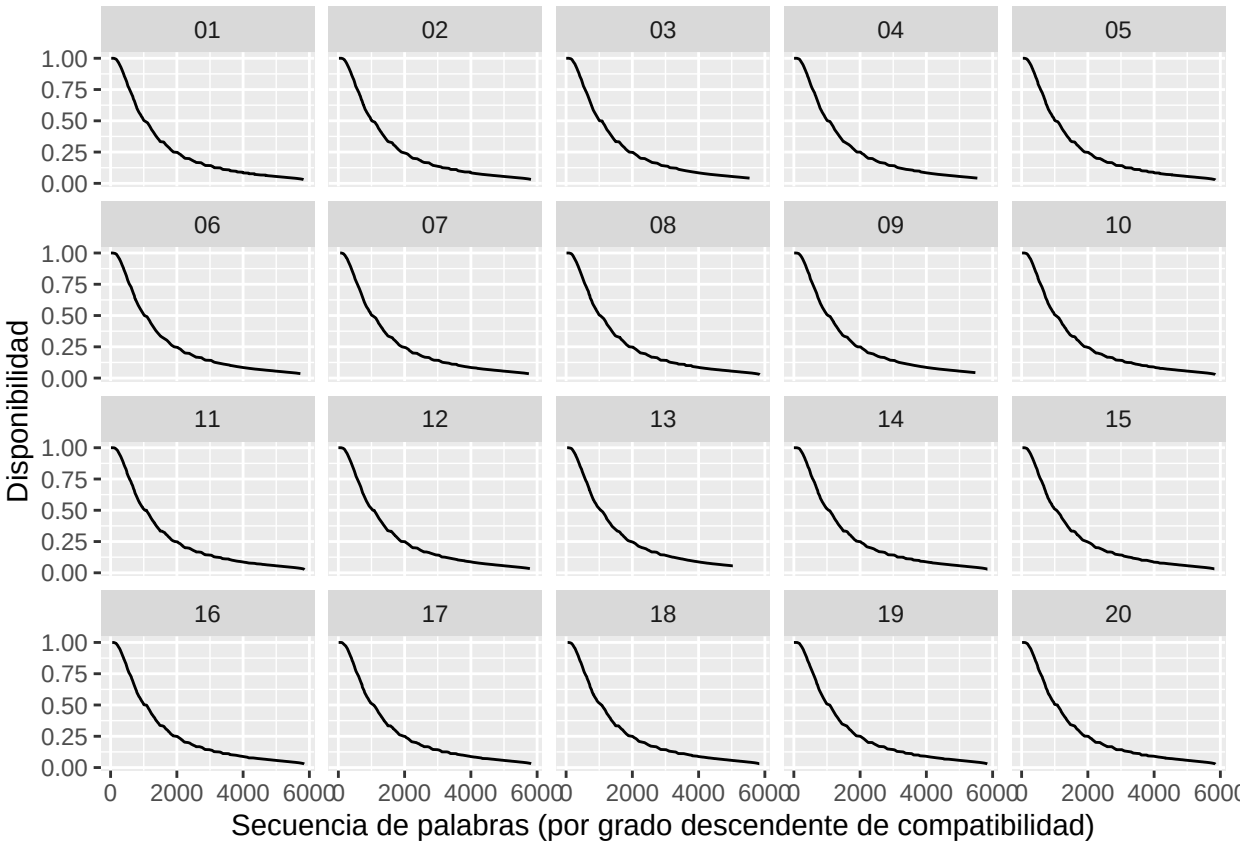


se puede observar que tienden a ser muy similares, calificando de la misma forma los elementos muy relevantes y los pocos relevantes.

Este último ejemplo es el que se propuso en nuestro anterior trabajo, en el que se puede observar una fuerte relación entre los dos modelos de evaluación, respondiendo a la misma pregunta. Sin embargo, el modelo que proponemos, al estar enmarcado dentro de un marco teórico bien establecido, la teoría de los conjuntos difusos, permite utilizar una serie de recursos teóricos que permiten llevar a cabo otros desarrollos en el estudio de la disponibilidad léxica.

Llevando a cabo este procedimiento para cada centro de interés:

```
d <- read.displex("datos.txt")
dd <- displex_availability(d,
                           law=function(x) {displex_zipf_law(x,d=1)},
                           reduce=displex_additive_law)
dd %>% arrange(-availability) %>% ggplot(aes(x=seq_along(words),y=availability)) + geom_line() +
  facet_wrap(~centers) +
  xlab("Secuencia de palabras (por grado descendente de compatibilidad)") +
  ylab("Disponibilidad")
```



Cálculo de la compatibilidad característica: Integral de Sugeno

El concepto de media o promedio en Estadística se modeliza a través del concepto de integral que, como su propio nombre indica, integra los valores de al variable ponderando por su importancia relativa, que se cuantifica como una distribución de frecuencia o probabilidad. Representa así un punto de equilibrio entre los distintos valores.

En el ámbito de las medidas difusas existe un concepto similar, que corresponde con la integral de Sugeno de una función respecto a una medida de conjuntos difusos y permite un punto de equilibrio entre los valores proporcionados a los elementos del conjunto y la medida del conjunto con los mayores valores, denominados α-cortes.

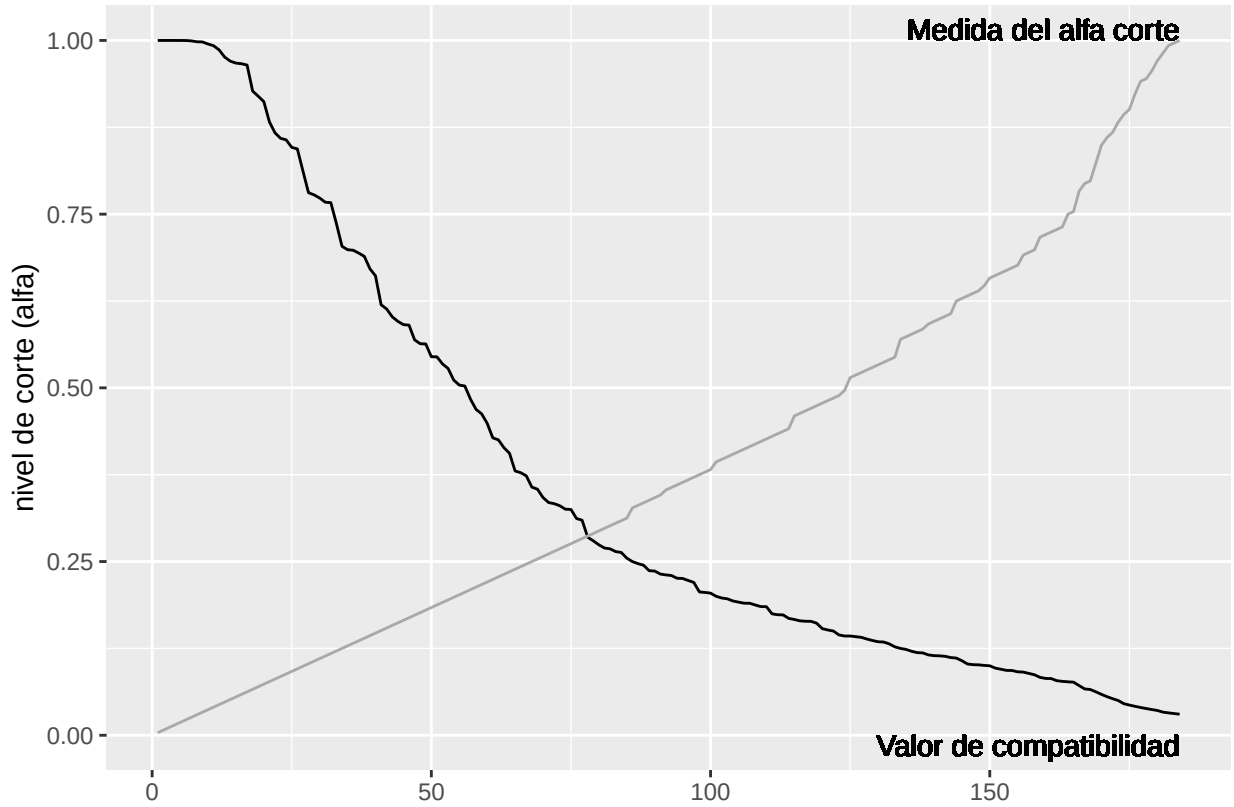
$$\int_A h(x) \circ g = \sup_{E \subseteq X} [\min(\min_{x \in E} h(x), g(A \cap E))] = \sup_{\alpha \in [0,1]} \min(\alpha, g(A \cap F_\alpha))$$

donde $F_\alpha = \{x | h(x) \geq \alpha\}$.

Tomando como $h(x)$ la disponibilidad obtenida en el estudio anterior y como $g(A)$ el tamaño relativo del conjunto (es decir, la proporción del total del conjunto A sobre todo el conjunto de términos), se obtendría un punto de equilibrio entre la compatibilidad con el centro de interés y el tamaño del conjunto.

Tomando la integral sobre todo el conjunto de términos, se construiría el FEV (Valor Difuso Esperado). Para ejemplificar este cálculo, considérese los distintos niveles de compatibilidad encontrados durante el cálculo de la disponibilidad (línea negra en el siguiente gráfico). Si calculamos para cada uno de esos valores la proporción de elementos que tienen al menos esa compatibilidad, a medida que disminuye la cota más valores alcanzan esa compatibilidad y mayor es el tamaño del conjunto (línea gris). El Valor Difuso Esperado es el valor en el que se cruzan las dos líneas.

```r
# Tomamos los valores de disponibilidad
dt <- avSanc$availability
g <- function(x) {length(x) / length(dt)}

# Determine alpha cuts and its measure
levels <- sort(unique(dt), decreasing=TRUE)
gs <- sapply(levels, function(x) {g(dt[dt >= x])})

data.frame(levels  = levels, measures = gs) %>%
  ggplot(aes(x=seq_along(levels), y=levels))  +
  geom_line(colour="black") +
  geom_line(aes(x=seq_along(levels), y=gs),colour="dark gray") + xlab("") + ylab("nivel de corte (alfa)
  geom_text(aes(x=length(levels), y=1), label="Medida del alfa corte",hjust=1,vjust=0) +
  geom_text(aes(x=length(levels), y=0), label="Valor de compatibilidad",hjust=1,vjust=1)
```



Para llevar a cabo este cálculo se proporciona la función `fuzzy.expected.value`, que devuelve el valor de compatibilidad característico.

```r
fuzzy.expected.value(avSanc$availability,g)
```

```
## [1] 0.2849462
```

Y es posible encontrar cuales son los valores característicos:

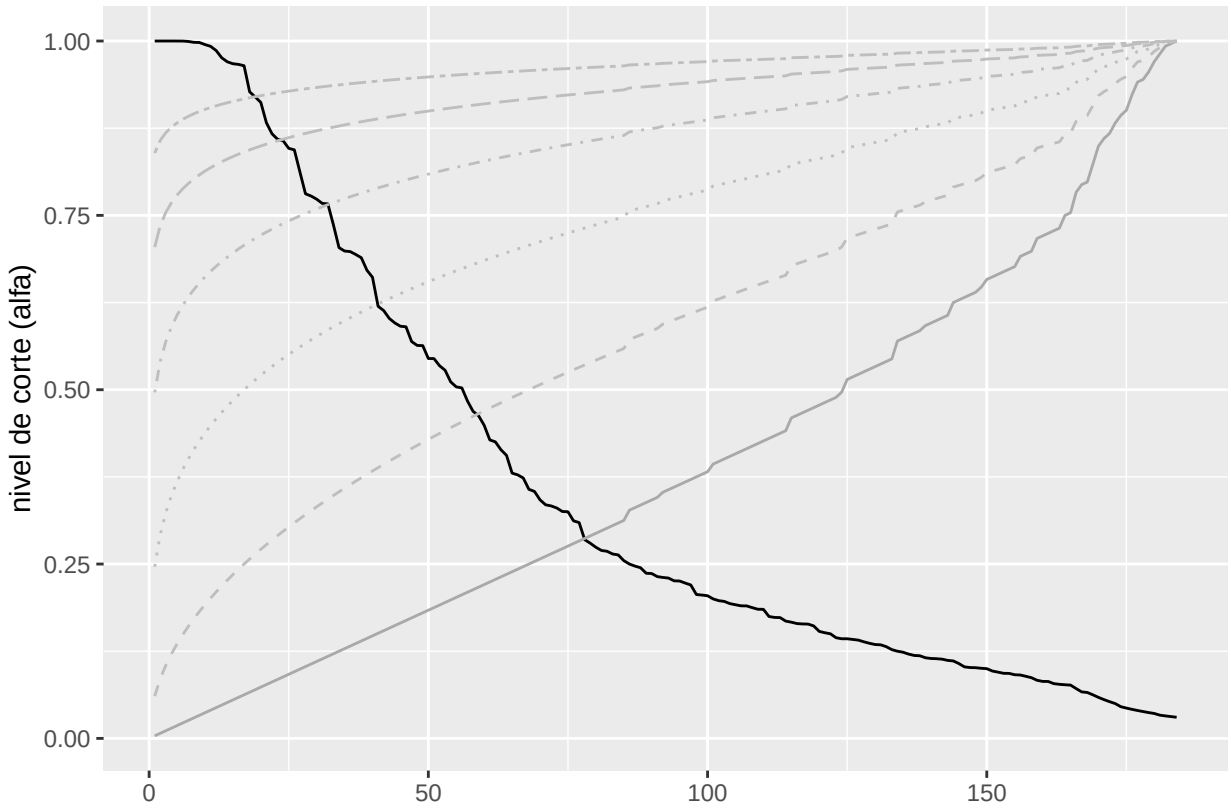
```r
avSanc %>%
  filter(availability >= fuzzy.expected.value(avSanc$availability,g)) %>%
  arrange(-availability) %>%
  pull(words)
```

```
##  [1] "cabeza"              "brazo"               "ojo"
##  [4] "mano"                "pierna"              "pie"
```

```
##  [7] "corazón"             "dedo"               "nariz"
## [10] "boca"                "pelo"               "oreja"
## [13] "hueso"               "hígado"             "músculo"
## [16] "tronco"              "uña"                "pulmón"
## [19] "riñón"               "estómago"           "extremidad"
## [22] "ceja"                "oído"               "órgano"
## [25] "cara"                "cerebro"            "codo"
## [28] "páncreas"            "rodilla"            "diente"
## [31] "sangre"              "hombro"             "cuello"
## [34] "esqueleto"           "pestaña"            "cintura"
## [37] "tibia"               "antebrazo"          "lengua"
## [40] "tobillo"             "vello"              "vena"
## [43] "enfermedad"          "espalda"            "pecho"
## [46] "labio"               "garganta"           "peroné"
## [49] "muñeca"              "bazo"               "esófago"
## [52] "culo"                "fémur"              "torso"
## [55] "salud"               "barriga"            "cadera"
## [58] "falange"             "piel"               "párpado"
## [61] "intestino"           "radio"              "tráquea"
## [64] "arteria"             "barbilla"           "bronquio"
## [67] "laringe"             "anatomía"           "frente"
## [70] "nuca"                "ombligo"            "sistema nervioso"
## [73] "columna (vertebral)" "saliva"             "deporte"
## [76] "húmero"              "pene"               "costilla"
```

Si se desea cortes de menor tamaño, hay que dar mayor relevancia al tamaño de los conjuntos, modificando la función de medida de los cortes.En este caso, se ha ido haciendo las raíces de índice potencia de dos (2,4,8,16 y 32) de los tamaños relativos de los alfa-cortes.

```
data.frame(levels  = levels, measures = gs) %>%
  ggplot(aes(x=seq_along(levels), y=levels))  +
  geom_line(colour="black") +
  geom_line(aes(x=seq_along(levels), y=gs),colour="dark gray",linetype=1) +
  geom_line(aes(x=seq_along(levels), y=sqrt(gs)),colour="gray",linetype=2) +
  geom_line(aes(x=seq_along(levels), y=sqrt(sqrt(gs))),colour="gray",linetype=3) +
  geom_line(aes(x=seq_along(levels), y=sqrt(sqrt(sqrt(gs)))),colour="gray",linetype=4) +
  geom_line(aes(x=seq_along(levels), y=sqrt(sqrt(sqrt(sqrt(gs))))),colour="gray",linetype=5) +
  geom_line(aes(x=seq_along(levels), y=sqrt(sqrt(sqrt(sqrt(sqrt(gs)))))),colour="gray",linetype=6) +
  xlab("") + ylab("nivel de corte (alfa)")
```



Se obtienen entonces sucesivos los sucesivos niveles característicos:

```
fuzzy.expected.value(avSanc$availability,function(x) {length(x) / length(avSanc$availability)})
```

```
## [1] 0.2849462
```

```
fuzzy.expected.value(avSanc$availability,function(x) {sqrt(length(x) / length(avSanc$availability))})
```

```
## [1] 0.4626262
```

```
fuzzy.expected.value(avSanc$availability,function(x) {sqrt(sqrt(length(x) / length(avSanc$availability)
```

```
## [1] 0.6196081
```

```
fuzzy.expected.value(avSanc$availability,function(x) {sqrt(sqrt(sqrt(length(x) / length(avSanc$availabi
```

```
## [1] 0.765284
```

```
fuzzy.expected.value(avSanc$availability,function(x) {sqrt(sqrt(sqrt(sqrt(length(x) / length(avSanc$ava
```

```
## [1] 0.8569412
```

```
fuzzy.expected.value(avSanc$availability,function(x) {sqrt(sqrt(sqrt(sqrt(sqrt(length(x) / length(avSan
```

```
## [1] 0.9197114
```

Y se pueden conseguir el vocabulario característico de cada centro de interés, en mayor grado de especificidad:

```
avSanc %>%
  filter(availability >= fuzzy.expected.value(avSanc$availability,function(x) {sqrt(length(x) / length(a
  arrange(-availability) %>%
  pull(words)
```

```
##  [1] "cabeza"     "brazo"      "ojo"        "mano"       "pierna"
```

```
##  [6] "pie"         "corazón"    "dedo"        "nariz"      "boca"
## [11] "pelo"        "oreja"      "hueso"       "hígado"     "músculo"
## [16] "tronco"      "uña"        "pulmón"      "riñón"      "estómago"
## [21] "extremidad" "ceja"       "oído"        "órgano"     "cara"
## [26] "cerebro"     "codo"       "páncreas"    "rodilla"    "diente"
## [31] "sangre"      "hombro"     "cuello"      "esqueleto"  "pestaña"
## [36] "cintura"     "tibia"      "antebrazo"   "lengua"     "tobillo"
## [41] "vello"       "vena"       "enfermedad" "espalda"    "pecho"
## [46] "labio"       "garganta"   "peroné"      "muñeca"     "bazo"
## [51] "esófago"     "culo"       "fémur"       "torso"      "salud"
## [56] "barriga"     "cadera"     "falange"     "piel"
```

Con sucesivos cortes se obtienen selecciones más pequeñas:

```
avSanc %>%
  filter(availability >= fuzzy.expected.value(avSanc$availability,function(x) {sqrt(sqrt(length(x) / le
  arrange(-availability) %>%
  pull(words)
```

```
##  [1] "cabeza"     "brazo"      "ojo"         "mano"       "pierna"
##  [6] "pie"         "corazón"    "dedo"        "nariz"      "boca"
## [11] "pelo"        "oreja"      "hueso"       "hígado"     "músculo"
## [16] "tronco"      "uña"        "pulmón"      "riñón"      "estómago"
## [21] "extremidad" "ceja"       "oído"        "órgano"     "cara"
## [26] "cerebro"     "codo"       "páncreas"    "rodilla"    "diente"
## [31] "sangre"      "hombro"     "cuello"      "esqueleto"  "pestaña"
## [36] "cintura"     "tibia"      "antebrazo"   "lengua"     "tobillo"
## [41] "vello"
```

```
avSanc %>%
  filter(availability >= fuzzy.expected.value(avSanc$availability,function(x) {sqrt(sqrt(sqrt(length(x)
  arrange(-availability) %>%
  pull(words)
```

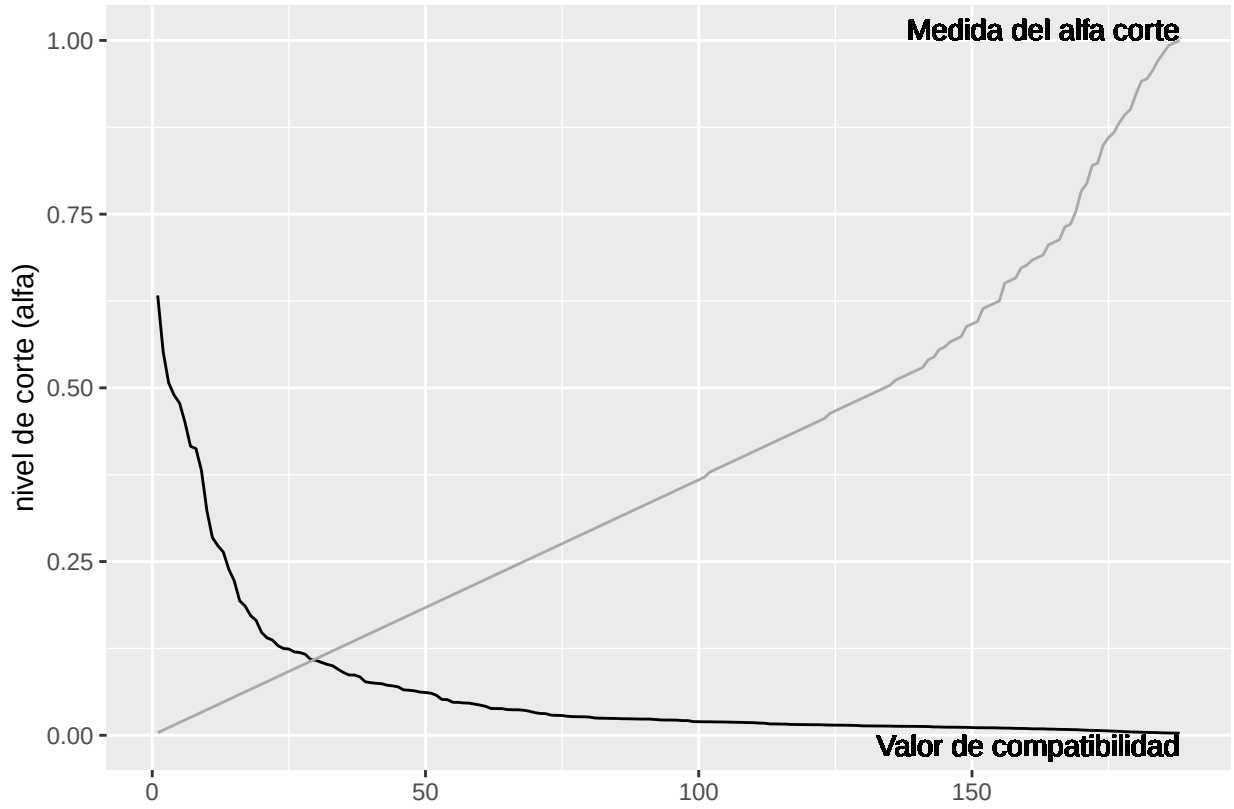
```
##  [1] "cabeza"     "brazo"      "ojo"         "mano"       "pierna"
##  [6] "pie"         "corazón"    "dedo"        "nariz"      "boca"
## [11] "pelo"        "oreja"      "hueso"       "hígado"     "músculo"
## [16] "tronco"      "uña"        "pulmón"      "riñón"      "estómago"
## [21] "extremidad" "ceja"       "oído"        "órgano"     "cara"
## [26] "cerebro"     "codo"       "páncreas"    "rodilla"    "diente"
## [31] "sangre"      "hombro"
```

```
avSanc %>%
  filter(availability >= fuzzy.expected.value(avSanc$availability,function(x) {sqrt(sqrt(sqrt(sqrt(leng
  arrange(-availability) %>%
  pull(words)
```

```
##  [1] "cabeza"     "brazo"      "ojo"         "mano"       "pierna"
##  [6] "pie"         "corazón"    "dedo"        "nariz"      "boca"
## [11] "pelo"        "oreja"      "hueso"       "hígado"     "músculo"
## [16] "tronco"      "uña"        "pulmón"      "riñón"      "estómago"
## [21] "extremidad" "ceja"       "oído"        "órgano"
```

El mismo procedimiento se puede llevar a cabo con los datos obtenidos de a partir del modelo de López-Strassburger

```r
# Tomamos los valores de disponibilidad
dt <- lopezstrass$availability
g <- function(x) {length(x) / length(dt)}

# Determine alpha cuts and its measure
levels <- sort(unique(dt), decreasing=TRUE)
gs <- sapply(levels, function(x) {g(dt[dt >= x])})

data.frame(levels  = levels, measures = gs) %>%
  ggplot(aes(x=seq_along(levels), y=levels))  +
  geom_line(colour="black") +
  geom_line(aes(x=seq_along(levels), y=gs),colour="dark gray") + xlab("") + ylab("nivel de corte (alfa)") +
  geom_text(aes(x=length(levels), y=1), label="Medida del alfa corte",hjust=1,vjust=0) +
  geom_text(aes(x=length(levels), y=0), label="Valor de compatibilidad",hjust=1,vjust=1)
```




```r
fuzzy.expected.value(lopezstrass$availability,g)
```

```
## [1] 0.1077405
```

```r
lopezstrass %>%
  filter(availability >= fuzzy.expected.value(lopezstrass$availability,g)) %>%
  arrange(-availability) %>%
  pull(words)
```

```
##  [1] "ojo"      "brazo"    "pierna"   "mano"     "cabeza"   "pie"
##  [7] "nariz"    "dedo"     "corazón"  "boca"     "uña"      "oreja"
## [13] "pelo"     "hígado"   "pulmón"   "estómago" "hueso"    "riñón"
## [19] "músculo"  "ceja"     "rodilla"  "diente"   "codo"     "tronco"
## [25] "páncreas" "cuello"   "oído"     "sangre"   "hombro"   "cerebro"
```

```
g <- function(x) {(length(x) / length(dt))^2}
fuzzy.expected.value(lopezstrass$availability,g)
```

```
## [1] 0.04546929
```

```
lopezstrass %>% filter(availability >= fuzzy.expected.value(lopezstrass$availability,g)) %>% arrange(-a
```

```
##  [1] "ojo"        "brazo"      "pierna"     "mano"       "cabeza"
##  [6] "pie"        "nariz"      "dedo"       "corazón"    "boca"
## [11] "uña"        "oreja"      "pelo"       "hígado"     "pulmón"
## [16] "estómago"   "hueso"      "riñón"      "músculo"    "ceja"
## [21] "rodilla"    "diente"     "codo"       "tronco"     "páncreas"
## [26] "cuello"     "oído"       "sangre"     "hombro"     "cerebro"
## [31] "cara"       "pestaña"    "lengua"     "tobillo"    "extremidad"
## [36] "antebrazo"  "espalda"    "pecho"      "labio"      "fémur"
## [41] "bazo"       "vena"       "muñeca"     "órgano"     "esófago"
## [46] "falange"    "cintura"    "tibia"      "cadera"     "piel"
## [51] "garganta"   "peroné"     "radio"      "culo"       "arteria"
## [56] "laringe"    "bronquio"   "intestino"
```

```
g <- function(x) {sqrt(length(x) / length(dt))}
fuzzy.expected.value(lopezstrass$availability,g)
```

```
## [1] 0.2268713
```

```
lopezstrass %>% filter(availability >= fuzzy.expected.value(lopezstrass$availability,g)) %>% arrange(-a
```

```
##  [1] "ojo"     "brazo"   "pierna"  "mano"    "cabeza"  "pie"     "nariz"
##  [8] "dedo"    "corazón" "boca"    "uña"     "oreja"   "pelo"    "hígado"
```

```
g <- function(x) {sqrt(sqrt(length(x) / length(dt)))}
fuzzy.expected.value(lopezstrass$availability,g)
```

```
## [1] 0.4125253
```

```
lopezstrass %>% filter(availability >= fuzzy.expected.value(lopezstrass$availability,g)) %>% arrange(-a
```

```
## [1] "ojo"    "brazo"  "pierna" "mano"   "cabeza" "pie"    "nariz"  "dedo"
```

Este procedimiento de cálculo del vocabulario más relevante tiene la ventaja de que son los propios valores de compatibilidad y su comportamiento los que determinan los diferentes cortes, y no listas con un tamaño prefijado. Así, la estructura de los distintos cortes van informando de la propia estructura del vocabulario.