1 Algoritmo de Cuadras-Augé

La cópula de Cuadras-Augé con parámetro $\theta \in (0,1)$ está dada por

$$C(u,v) = \min(u,v)\max(u,v)^{1-\theta}.$$

El siguiente algoritmo genera pares de variables (U,V) con distribución dada por la cópula de Cuadras-Augé. Define

$$H(w,v;\theta) = \begin{cases} \frac{w}{1-\theta}v^{\theta} & si \quad w < (1-\theta)v^{1-\theta} \\ v & si \quad (1-\theta)v^{1-\theta} \leq w < v^{1-\theta} \\ w^{1-\theta} & si \quad w \geq v^{1-\theta} \end{cases}$$

1. Para $\theta \in (0,1)$. Genera de manera independiente W,V uniformes(0,1)
2. Define U = $H(W,V;\theta)$
3. Regresa U,V

2 Ejercicios

1. Para cada valor de $\theta = 0,\ 0.1,\ 0.3,\ 0.85,\ 0.9$:
   (a) Simula Variables 1000 aleatorias (U,V) con el algoritmo de Cuadras-Augé y verifica que las marginales de (U,V) son uniformes en (0,1).
   (b) Utilizando los datos anteriores, grafica (X,Y) con $X = -log(U)$ y $Y = -log(V)$.
   (c) Verifica empiricamente que las marginales de X y Y tienen distribución exponencial.
2. Finalmente describe el procedimiento para realizar la simulación de tres variables aleatorias, usando una cópula $C(u_1,u_2,u_3)$.

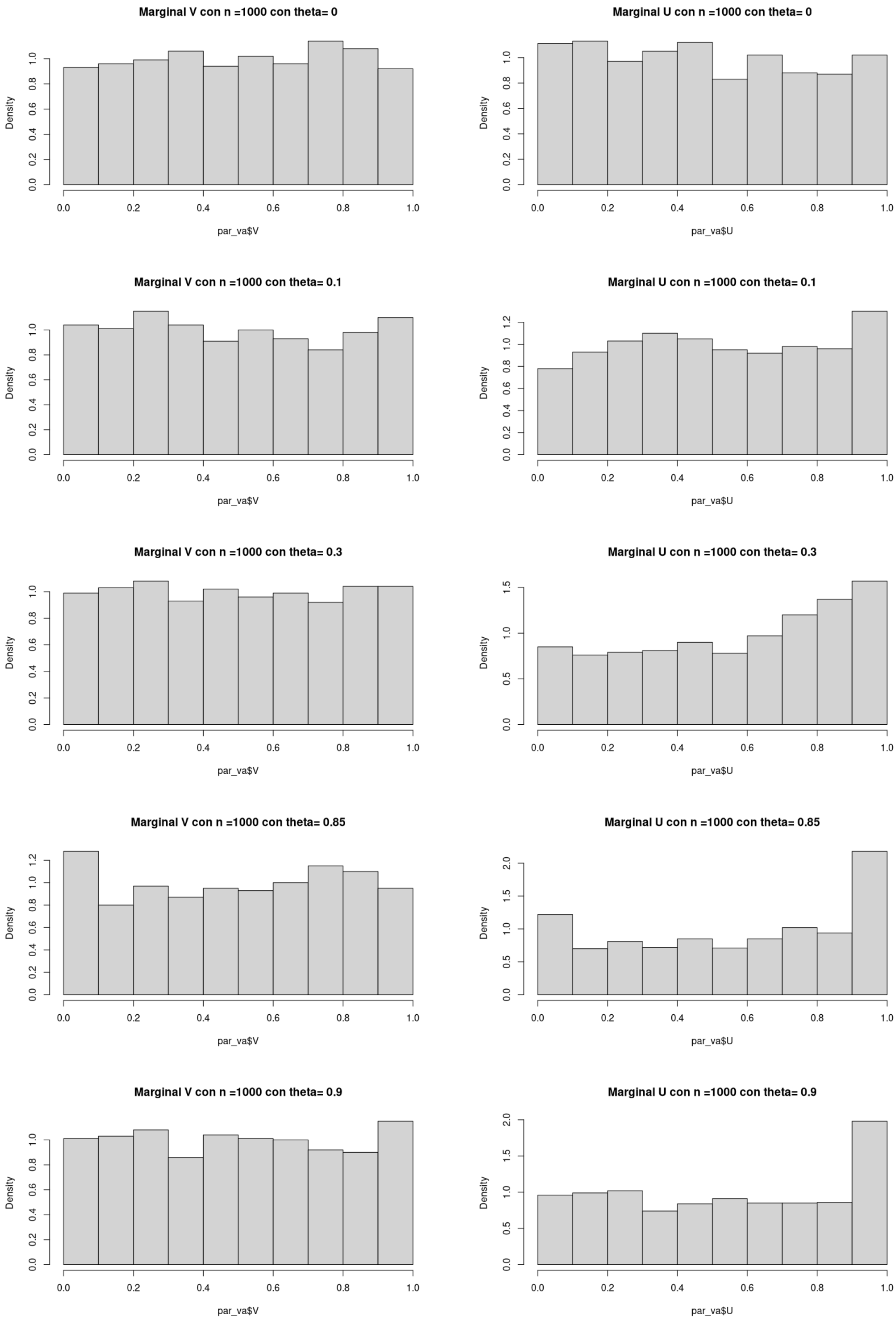
Algoritmo de Cuadras-Augé

Para la simulacion usaremos el algoritmo Cuadras-Auge que generará pares de variables con la distribución dada por la cópula de Cuadras Augé, primeros construiremos la función inversa en R de esta forma:

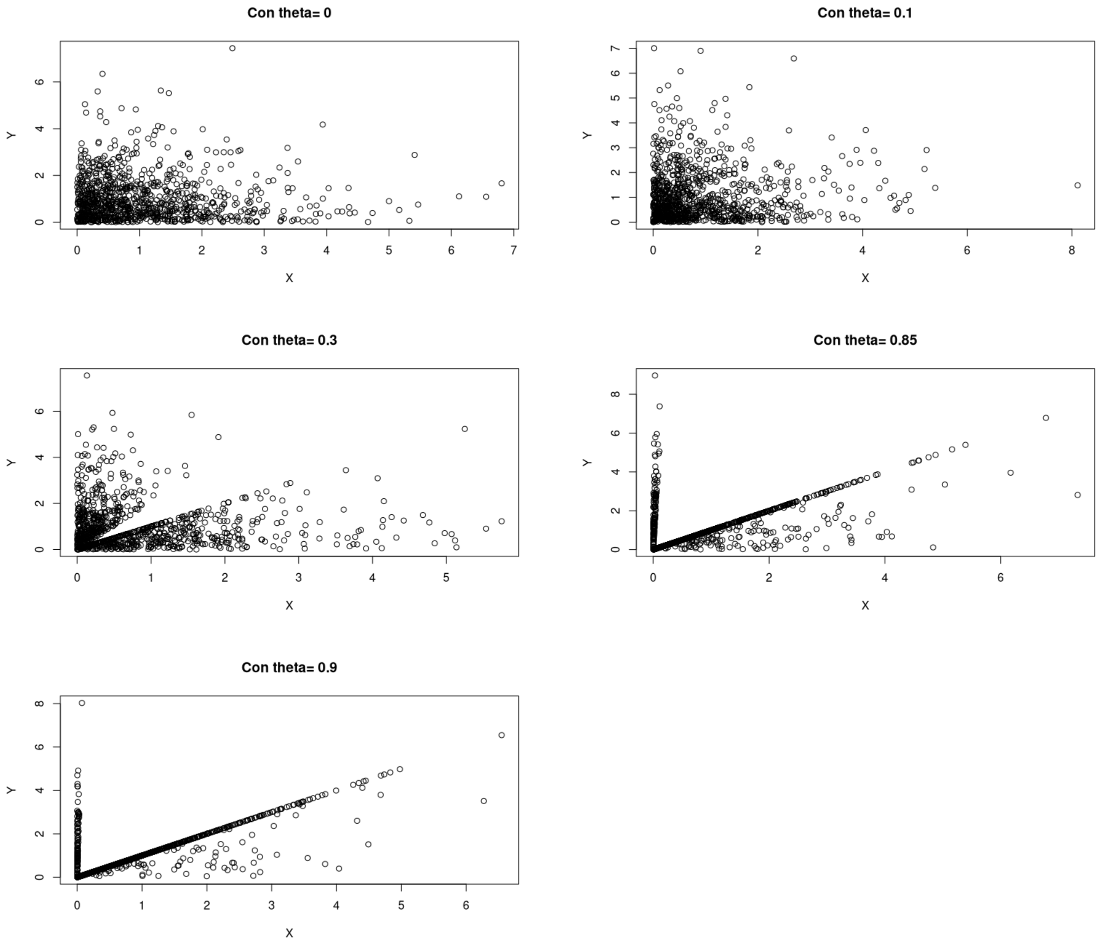
```
#'  Funcin  inversa de la cpula de Cuadras-Aug
#'
#' @param w variable aleatoria uniforme (0,1).
#' @param v variable aleatoria uniforme (0,1).
#' @param theta parmetro para distribucin.
#'
#' @return
H <- function(w, v, theta) {
  if (w  < (1-theta) * v^(1-theta)) {
    return((w/(1-theta))*v^theta)
  } else if (w  < v^(1-theta)) {
    return(v)
  } else if (w >= v^(1-theta)){
    return(w^(1-theta))
  }
}
```

Una vez implementado la usaremos en nuestro algoritmo que consistira en generar dos variables uniformes de (0,1) llamda W y V. Despues llamaremos nuestra función inversa inversa que no devolvera variables aleatorias, las almacenamos en un vector U y lo devolvemos:

```
#' Algoritmo para generar pares de variables (U,V) con distribucin dada
#' por la cpula de Cuadras-Aug
#'
#' @param n - numero de variables a simular.
#' @param theta - parmetro para la distribucin.
#'
#' @return un par de variables (U,V)
simular_cuadras_auge <- function(n, theta) {
  W <- runif(n)
  V <- runif(n)
  U <- c()
  for (i in 1:n) {
    U[i] <- H(W[i], V[i], theta)
  }
  return(list(U, V))
}
```



Observamos que con $\theta = 0$ las dos marginales son uniformes, es decir, cada valor de la variables tienen casi la misma frecuencia en salir, al ir cambiando la *thena* vemos que la marginl V va cambiando la frecuencia sesgando el histograma hacia un extremos, se puede ver claramente que cuando $\theta = 0.9$ la marginal V ya no es uniforme debido a que ya no hay la misma frecuencia en salir, vemos que en el extremo derecha hay mayor frecuencia en salir, lo que quiere decir es que exite mayor probabilidad.




```r
X = -log(par_va$U)
Y = -log(par_va$V)
plot(X, Y, main = paste("Con theta=",theta))
```

Podemos apreciar que cuando theta va incrementándose nuestro puntos generando van tomando una forma agruponse en una linea recta con pendiente positiva, vemos que al principio genera una nube puntos, es quiere decir que las variables podrian ser independientes cuando $\theta = 0$, pero conforme aumentamos θ tienden a gruparse como una linea que implica que y depende de x.

Finalmente describe el procedimiento para realizar la simulación de tres variables aleatorias, usando una cópula C(u 1 , u 2 , u 3).

```
Procedimiento
1.- Primero genereramos tres variables uniformes W,V,R de (0,1)
2.- Encontramos la función inversa de C(u 1 , u 2 , u 3 ) que sera G.
4.- Definimos U = G(W, V, R, theta)
5.- Definimos U2 = G(W, V, R, theta)
6.- Regresamos U, U2, V
```