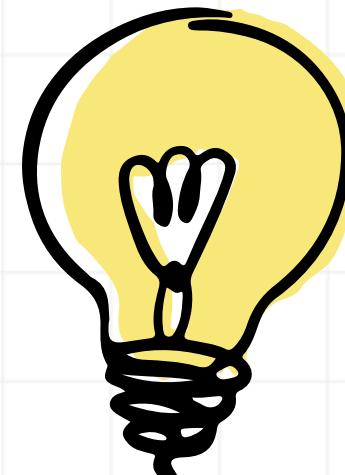


MÉTODOS COMPUTACIONALES EN ESTADÍSTICA

GENERACIÓN DE VARIABLES ALEATORIAS



CURSO 2023-2024
NOELIA CÁRDENAS SALAMANCA



MÉTODO DE LA TABLA GUÍA

GENERACIÓN DE UNA
DISTRIBUCIÓN BINOMIAL A
PARTIR DEL MÉTODO DE LA
TABLA GUÍA

MÉTODO DE FACTORIZACIÓN
DE LA MATRIZ DE
COVARIANZAS

MÉTODO DE ALIAS

GENERACIÓN DE UNA
DISTRIBUCIÓN BINOMIAL A
PARTIR DEL MÉTODO DE
ALIAS

GENERACIÓN DE UNA NORMAL
MULTIVARIANTE A PARTIR DEL
MÉTODO DE FACTORIZACIÓN DE
LA MATRIZ DE COVARIANZAS

ÍNDICE

PASO 1

Desde $i=1$ hasta n hacer $q_i=n\pi_i$
Establecer $L=\{l:q_l<1\}$ y $H=\{h:q_h\geq 1\}$
Si L ó H vacíos terminar.
Seleccionar $l \in L$ y $h \in H$.
Hacer $a_l=h$.
Eliminar l de L .
Hacer $q_h=q_h-(1-q_l)$.
Si $q_h<1$ mover h de H a L .
Ir al punto 3.

PASO 2

Generar $U, V \sim U(0,1)$.
Hacer $i=\lfloor nU \rfloor + 1$.
Si $V < q_i$ devolver $X=x_i$.
En caso contrario devolver
 $X=x_{a_i}$.

ALGORITMO DE ALIAS

¿En qué consiste el método?

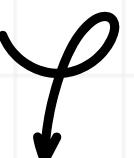
Se basa en representar la distribución de X como una mixtura (uniforme) de variables dicotómicas.

$$Q^{(i)} = \begin{cases} x_i & \text{con prob. } q_i \\ x_{a_i} & \text{con prob. } 1 - q_i \end{cases}$$

La idea es “tomar prestada” parte de la probabilidad de los valores más probables (ricos) para asignársela a los valores menos probables (pobres), recordando el valor de donde procede (almacenando el índice en a_i).

¿En qué se diferencia del resto de algoritmos?

EFICIENCIA COMPUTACIONAL



NO REQUIERE
GENERACIÓN ADICIONAL
DE NÚMEROS
ALEATORIOS.

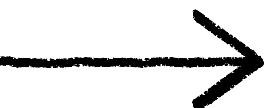
Este método se utiliza
en distribuciones
discretas

CÓDIGO EN R DEL ALGORITMO DE ALIAS

```
simulacion_alias <- function(x, prob, n, as.factor = FALSE) {
```

Inicializamos la tabla:

```
a <- numeric(length(x))
q <- prob*length(x)
low <- q < 1
high <- which(!low)
low <- which(low)
while (length(high) && length(low)) {
  l <- low[1]
  h <- high[1]
  a[l] <- h
  q[h] <- q[h] - (1 - q[l])
  if (q[h] < 1) {
    high <- high[-1]
    low[1] <- h
  } else low <- low[-1]
}
```



PASO 1

Devuelve un vector de
valores simulados de
una distribución
discreta



Generamos valores:

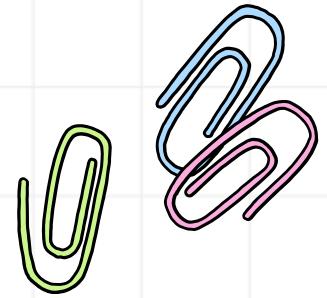
```
V <- runif(n)
i <- floor(runif(n)*length(x)) + 1
X <- x[ ifelse( V < q[i], i, a[i]) ]
if(as.factor) X <- factor(X, levels = x)
return(X)
```



PASO 2

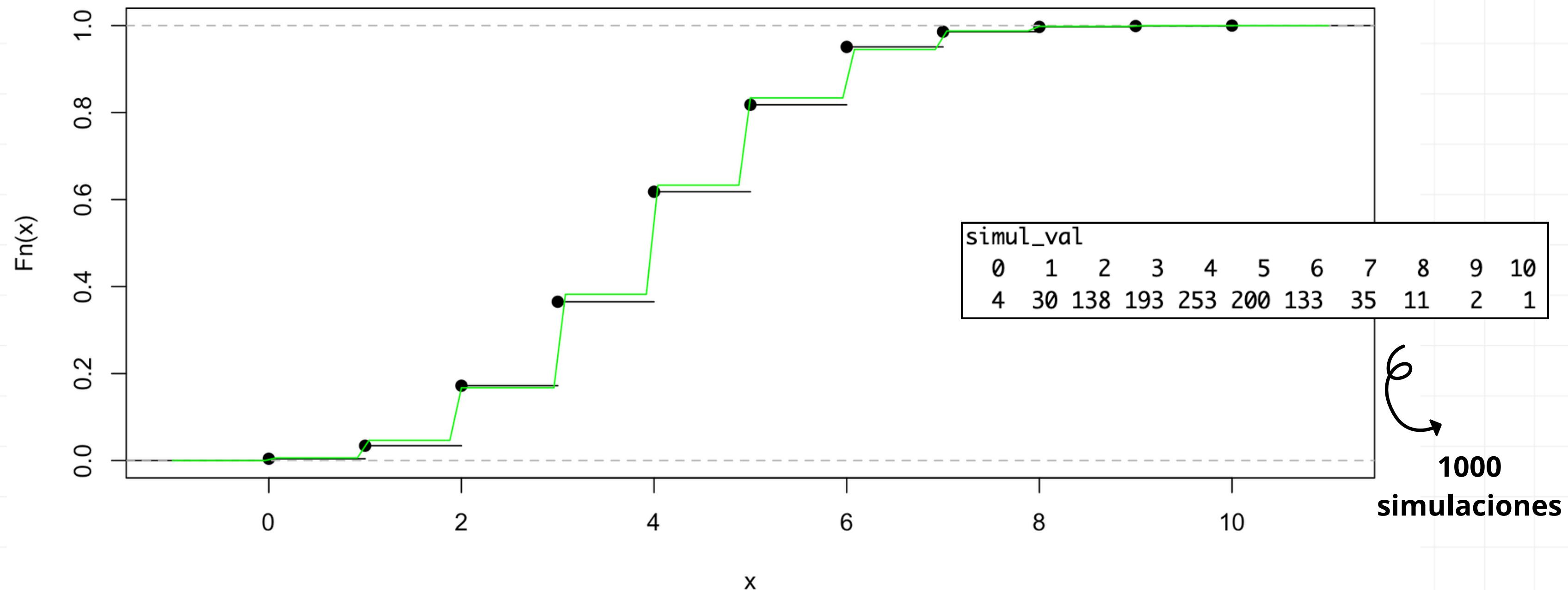
GENERACIÓN DE UNA DISTRIBUCIÓN BINOMIAL

A TRAVÉS DEL MÉTODO DE ALIAS



Tomamos como ejemplo una binomial de parámetros $n=10$ y $p=0.4$

Distribución empírica simulación de $B(10,0.4)$



GENERACIÓN DE UNA DISTRIBUCIÓN BINOMIAL

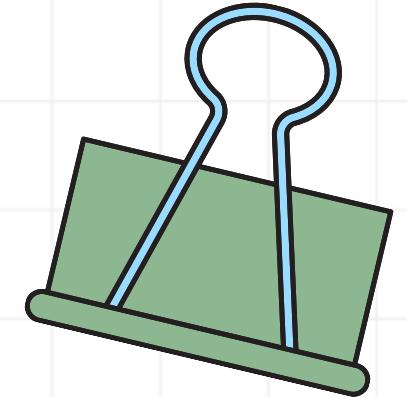
A TRAVÉS DEL MÉTODO DE ALIAS

Tomamos como ejemplo una binomial de parámetros $n=10$ y $p=0.4$

Comparación de las medias

Media teórica: 4

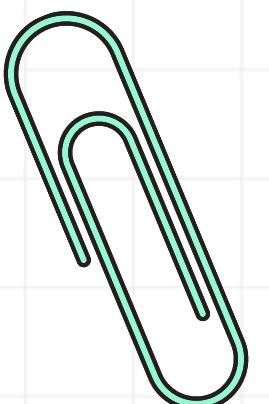
Media simulada: 4.056



Comparación de las desviaciones estándar

Desviación Estándar teórica: 1.549193

Desviación Estándar simulada: 1.545073



Comparación de las varianzas

Varianza teórica: 2.4

Varianza simulada: 2.387251

PASO 1

Hacer $F_1 = p_1$
Y desde $i = 2$ hasta n
hacer $F_i = F_{i-1} + p_i$

PASO 2

Hacer $g_1 = 1$, $i = 1$
Desde $j = 2$ hasta m hacer:
Mientras $(j-1)/m > F_i$ hacer $i = i + 1$
 $g_j = i$

PASO 3

Generar $U \sim U(0, 1)$
Hacer $j = [mU] + 1$
Hacer $i = g_j$
Mientras $U > F_i$ hacer $i = i + 1$
Devolver $X = x_i$



¿En qué consiste el método?

construir m subintervalos equiespaciados en $[0,1]$ de la forma:

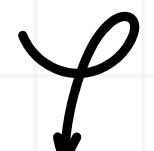
$$I_j = [u_j, u_{j+1}) = \left[\frac{j-1}{m}, \frac{j}{m} \right) \text{ para } j = 1, 2, \dots, m$$

y utilizarlos para iniciar la búsqueda.

En una tabla guía se almacenan los índices de los cuantiles correspondientes a los extremos inferiores de los intervalos:

$$g_j = Q_{\mathcal{I}}(u_j) = \inf \left\{ i : F_i \geq u_j = \frac{j-1}{m} \right\}$$

¿En qué se diferencia del resto de algoritmos?



EVITA CÁLCULOS REPETITIVOS DURANTE LA GENERACIÓN DE VALORES SIMULADOS.

ACELEERA EL PROCESO DE SIMULACIÓN

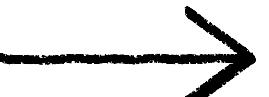
Este método se utiliza en distribuciones discretas

CÓDIGO EN R DEL ALGORITMO DE LA TABLA GUÍA

```
simulacion_tabla_guiia <- function(x, prob, m, n, as.factor = FALSE) {
```

Inicializamos la tabla y la función de distribución:

```
  Fx <- cumsum(prob)
  g <- rep(1,m)
  i <- 1
  for(j in 2:m) {
    while (Fx[i] < (j-1)/m) i <- i + 1
    g[j] <- i
  }
```



PASO 2

Simulación mediante la tabla guía:

Devuelve un vector de
valores simulados de
una distribución
discreta

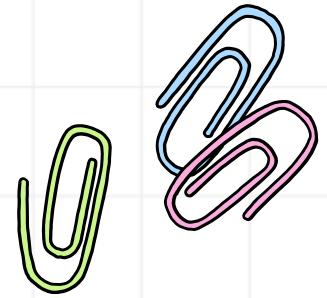
```
  X <- numeric(n)
  U <- runif(n)
  for(j in 1:n) {
    i <- i0 <- g[floor(U[j] * m) + 1]
    while (Fx[i] < U[j]) i <- i + 1
    X[j] <- x[i]
  }
  return(X)
```



PASO 3

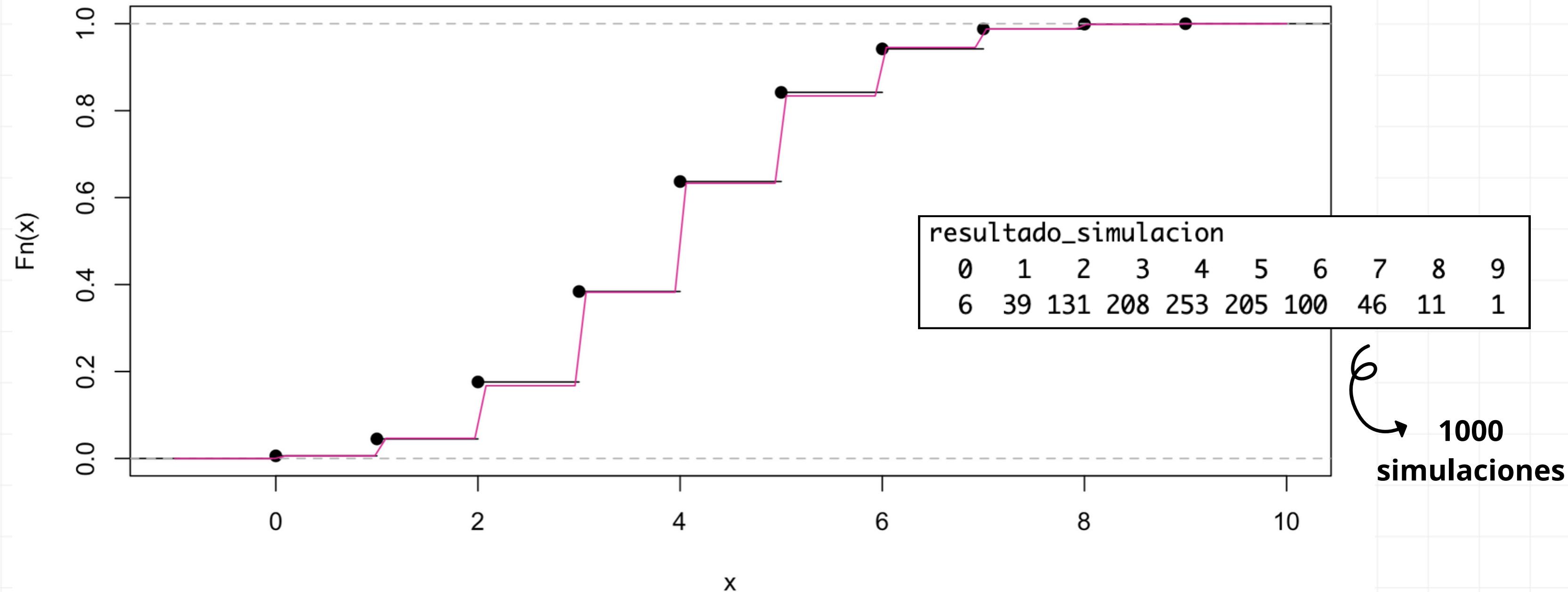
GENERACIÓN DE UNA DISTRIBUCIÓN BINOMIAL

A TRAVÉS DEL MÉTODO DE LA TABLA GUÍA



Tomamos como ejemplo una binomial de parámetros $n=10$ y $p=0.4$

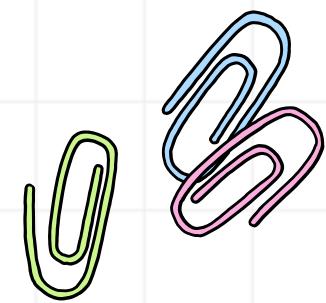
Distribución empírica simulación de $B(10,0.4)$



GENERACIÓN DE UNA DISTRIBUCIÓN BINOMIAL

A TRAVÉS DEL MÉTODO DE LA TABLA GUÍA

Tomamos como ejemplo una binomial de parámetros $n=10$ y $p=0.4$



Comparación de las medias

Media teórica: 4

Media simulada: 3.981

Comparación de las desviaciones estándar

Desviación Estándar teórica: 1.549193

Desviación Estándar simulada: 1.549529

Comparación de las varianzas

Varianza teórica: 2.4

Varianza simulada: 2.40104



VENTAJAS

Eficiencia Computacional: Especialmente cuando se generan múltiples muestras.

Adaptabilidad a Distribuciones no Uniformes

DESVENTAJAS

Complejidad en la construcción



VENTAJAS

Simplicidad en la Construcción

Gran flexibilidad

Eficiente

DESVENTAJAS

Falta de eficiencia Computacional

Limitaciones con Distribuciones no Uniformes

Son métodos muy similares que tienen un mismo objetivo, generar variables aleatorias discretas.

ALIAS

CUANDO LA EFICIENCIA EN LA GENERACIÓN DE MÚLTIPLES MUESTRAS DISCRETAS ES CRÍTICA.



TABLA GUÍA

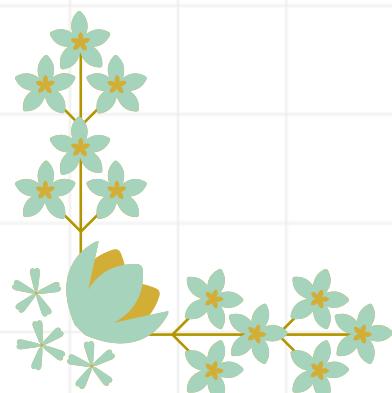
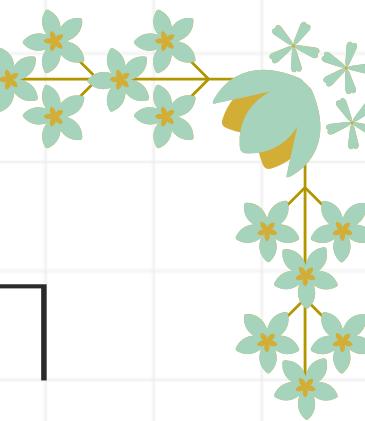
SITUACIONES DONDE LA SIMPLICIDAD Y LA CONSTRUCCIÓN RÁPIDA SON MÁS IMPORTANTES QUE LA EFICIENCIA EN LA GENERACIÓN DE MUESTRAS.

¿EN QUÉ SITUACIÓN ELEGIMOS UNO U OTRO?

ELECCIÓN DEL MÉTODO

ADECUADO PARA DISTRIBUCIONES NO UNIFORMES.

DISTRIBUCIONES UNIFORMES O CUANDO SE PUEDE APROXIMAR A UNA DISTRIBUCIÓN UNIFORME.



PASO 1

Obtener la factorización de Cholesky : $\Sigma = L^* L^t$

PASO 2

Simular $Z=(Z_1, Z_2, \dots, Z_d)$
i.i.d $N(0,1)$

PASO 3

Hacer $X = \mu + LZ$

PASO 4

Repetir pasos 2 y 3 las veces necesarias



¿En qué se
diferencia del resto
de algoritmos?

ES UN MÉTODO EFECTIVO
PARA LA SIMULACIÓN DE
DISTRIBUCIONES
MULTIVARIANTES

¿En qué consiste el método?

Este método se emplea principalmente para la simulación de una normal multivariante:

Si $X \sim N(\mu, \Sigma)$ partiendo de $Z \sim N(0, Id)$, se podrían considerar distintas factorizaciones de la matriz de covarianzas:

- Factorización de Cholesky: $\Sigma = L^* L^t$ donde L es una matriz triangular inferior por lo que:

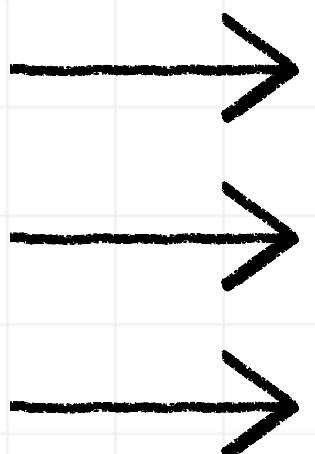
$$X = \mu + LZ \sim N(\mu, \Sigma)$$

CÓDIGO EN R DEL ALGORITMO DE FACTORIZACIÓN DE LA MATRÍZ DE COVARIANZAS

```
simulacion_normal_multivariante <- function(n, mu, Sigma) {  
  L <- t(chol(Sigma))  
  Z <- matrix(rnorm(n * length(mu)), nrow = length(mu), ncol = n)  
  X <- mu + L %*% Z  
  return(X)  
}
```



Devuelve una matriz que contiene valores simulados de la normal



PASO 1

PASO 2

PASO 3

μ : vector de medias
 Σ : matriz de var-cov

GENERACIÓN DE UNA DISTRIBUCIÓN NORMAL MULTIVARIANTE

A TRAVÉS DEL MÉTODO DE FACTORIZACIÓN DE LA MATRIZ DE VAR-COV

Tomaremos como ejemplo una normal bivariante con vector de medias $\mu=(1,2)$ y matriz de var-cov :

$$\Sigma = (2, \text{cov}(X_1, X_2), \text{cov}(X_1, X_2), 3)$$

¿CÓMO CALCULAMOS LA COVARIANZA ENTRE LAS NORMALES?

```
mu_X1 <- 1
sigma_X1 <- sqrt(2)

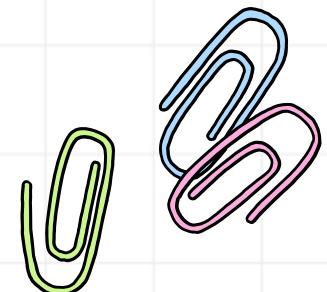
mu_X2 <- 2
sigma_X2 <- sqrt(3)

set.seed(100)

muestra_X1 <- rnorm(10000, mean = mu_X1, sd = sigma_X1)
muestra_X2 <- rnorm(10000, mean = mu_X2, sd = sigma_X2)

covarianza <- cov(muestra_X1, muestra_X2)

correlacion <- covarianza/(sigma_X1*sigma_X2)
```



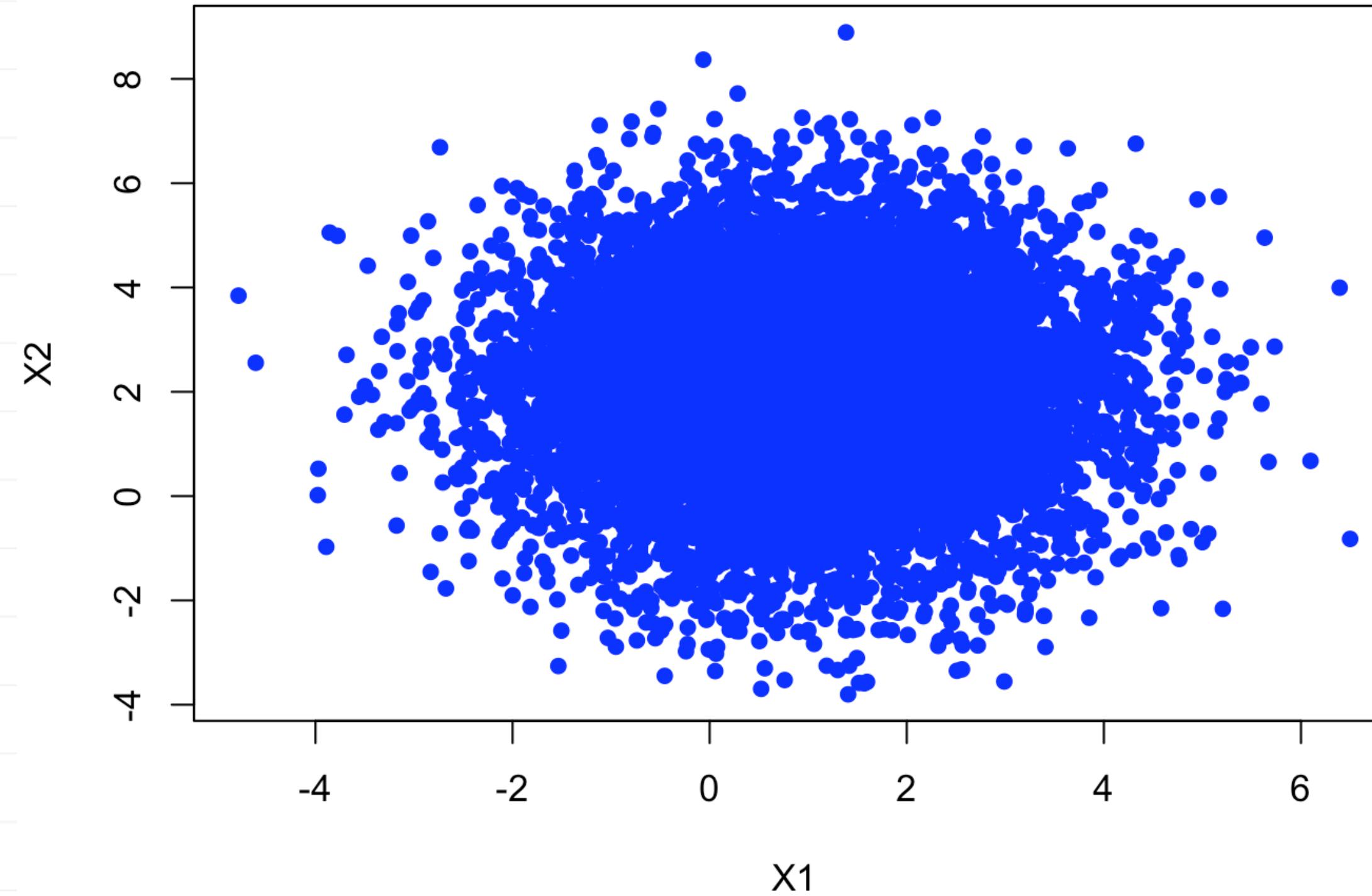
GENERACIÓN DE UNA DISTRIBUCIÓN NORMAL MULTIVARIANTE

A TRAVÉS DEL MÉTODO DE FACTORIZACIÓN DE LA MATRIZ DE VAR-COV

Gráfico de dispersión

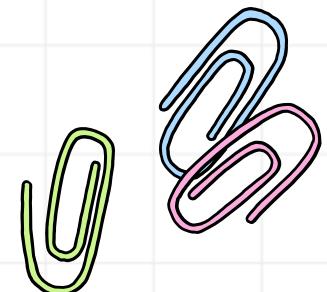
RESULTADO
COVARIANZA

-0.008920561



RESULTADO COEFICIENTE
DE CORRELACIÓN

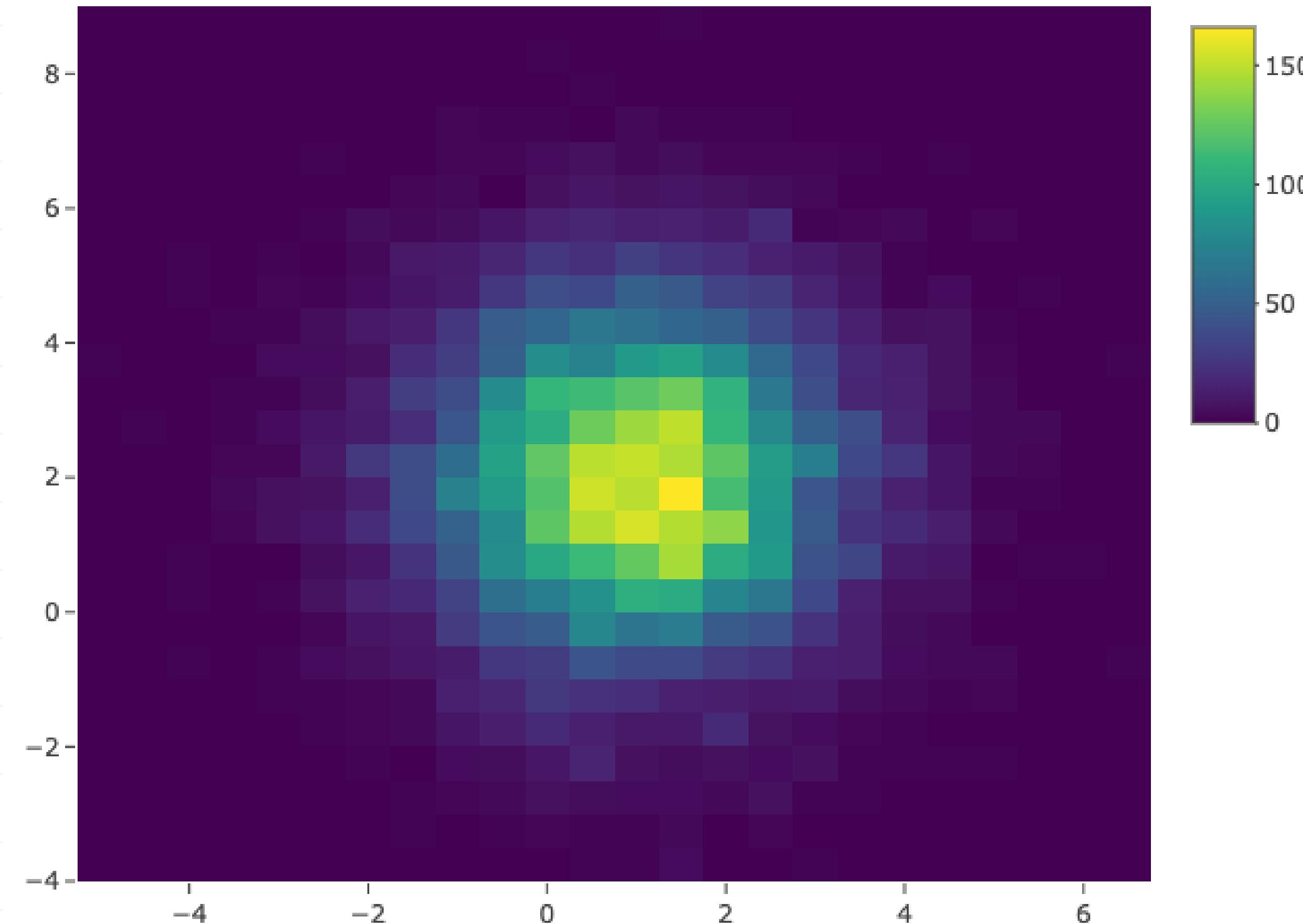
-0.003641804



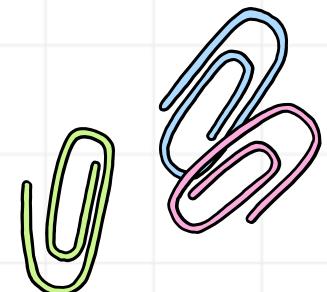
GENERACIÓN DE UNA DISTRIBUCIÓN NORMAL MULTIVARIANTE

A TRAVÉS DEL MÉTODO DE FACTORIZACIÓN DE LA MATRIZ DE VAR-COV

Histograma de $N(\mu, \Sigma)$



10000
simulaciones
de la Normal
bivariante



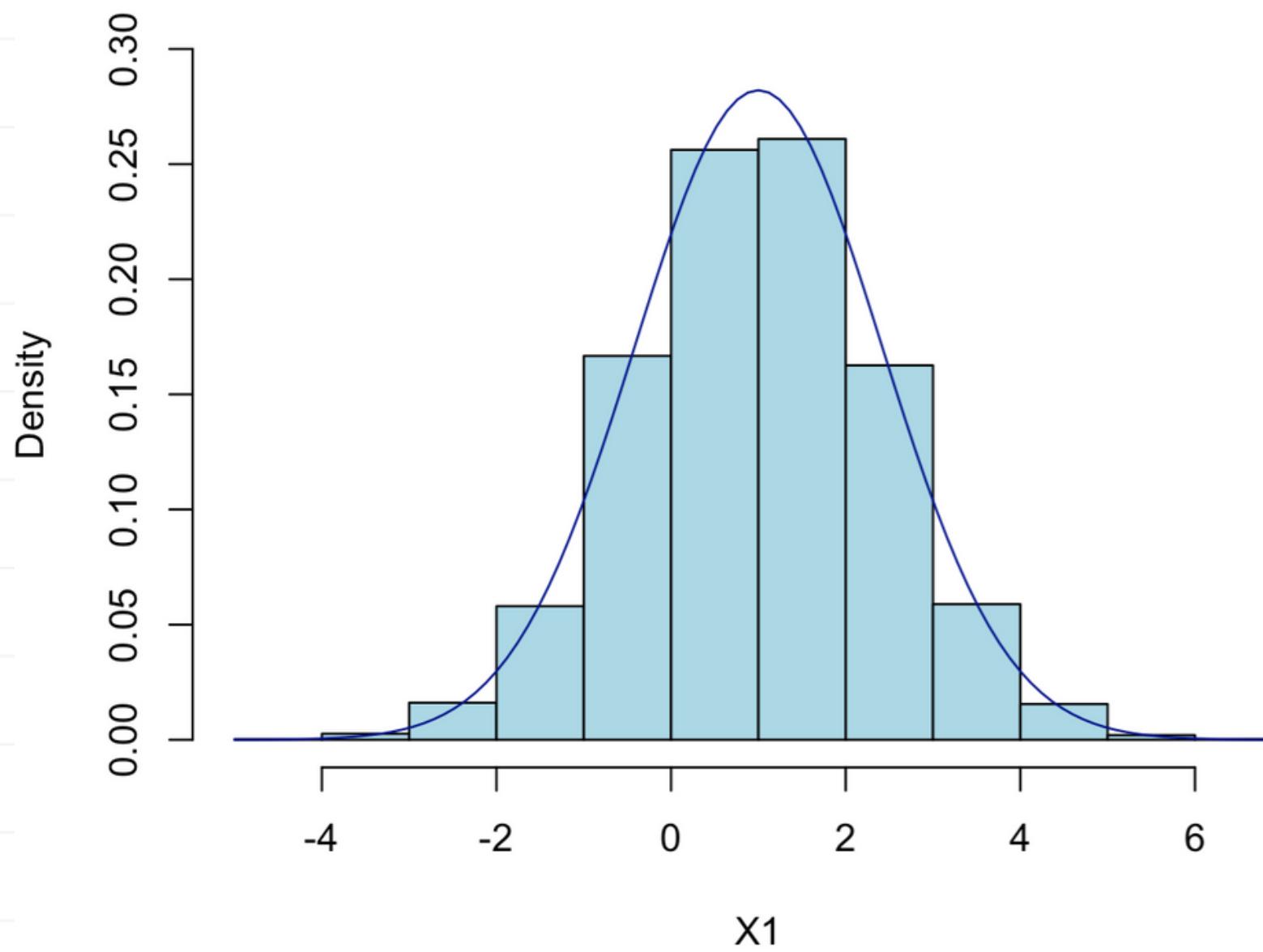
GENERACIÓN DE UNA DISTRIBUCIÓN NORMAL MULTIVARIANTE

A TRAVÉS DEL MÉTODO DE FACTORIZACIÓN DE LA MATRIZ DE VAR-COV

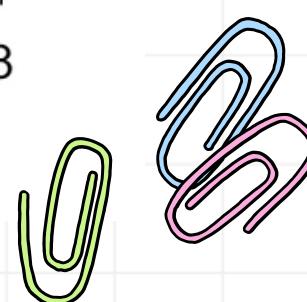
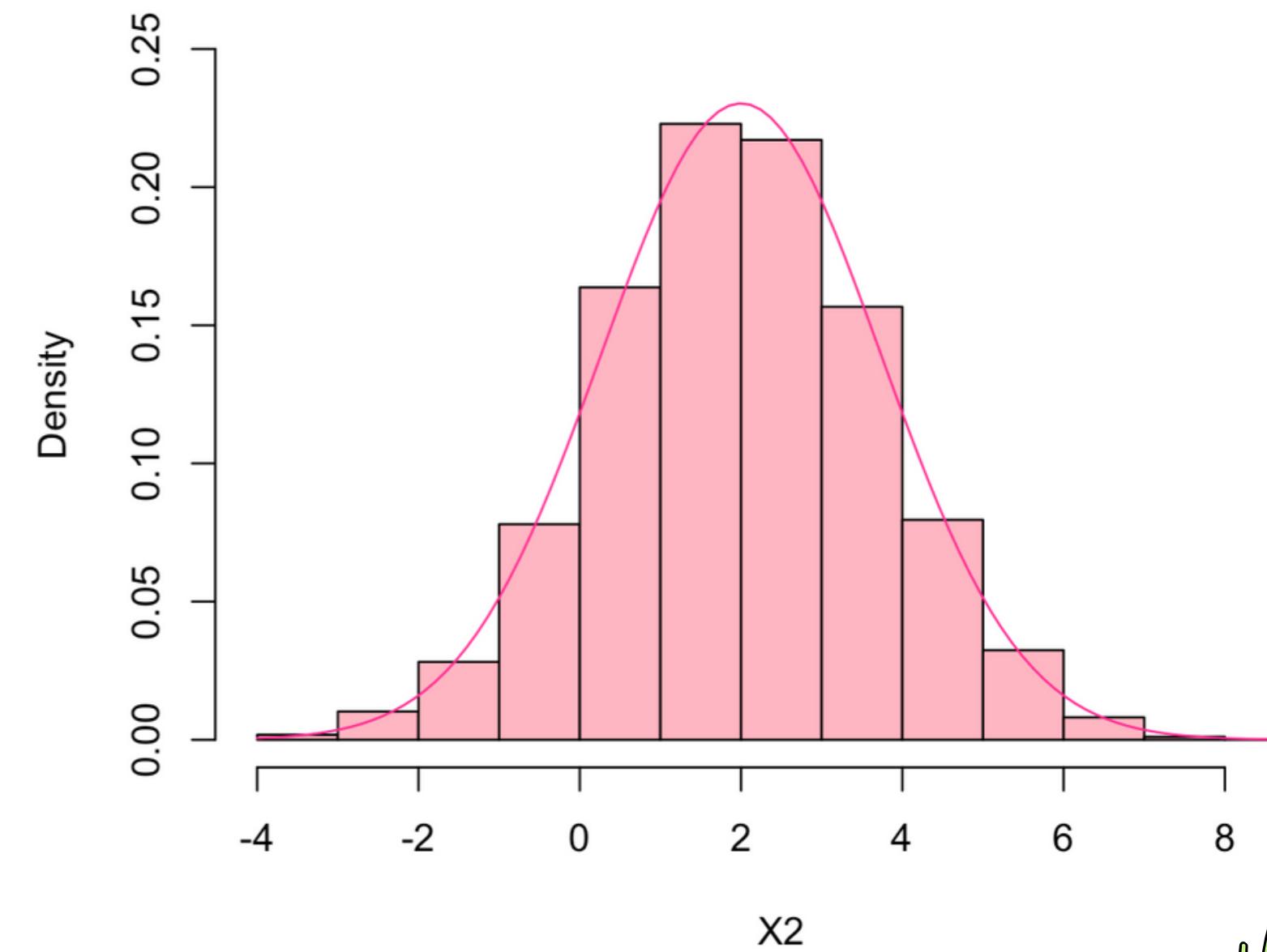
Podemos estudiar las distribuciones marginales comparándolas con su distribución original

6
10000
simulaciones
de cada Normal

Histograma de $N(1,2)$



Histograma de $N(2,3)$



GENERACIÓN DE UNA DISTRIBUCIÓN NORMAL MULTIVARIANTE

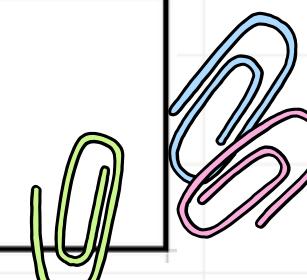
A TRAVÉS DEL MÉTODO DE FACTORIZACIÓN DE LA MATRIZ DE VAR-COV

¿Existen diferencias significativas si aumentamos el número de simulaciones?

En una normal multivariante aumentar el número de simulaciones puede significar una mejora en el ajuste

Veamos los cambios producidos siendo $X_1 \sim N(1,2)$ y $X_2 \sim N(2,3)$

Repeticiones	1.000	10.000	100.000	1.000.000	10.000.000
Asimetría X1	-0,11067890	-0,0168105233	-0,008413355	-0,0003459324	-0,0006371794
Asimetría X2	0,081940845	0,000403652	-0,006152766	-0,000715679	0,000429010
Curtosis X1	-0,138794549	0,064990342	-0,010007057	0,004519798	0,002383386
Curtosis X2	0,027990043	0,018417294	0,003028469	-0,003669688	0,001092582
Suma de distancias	0,35940434	0,10062181	0,02760165	0,00925110	0,00454216
Reducción del error respecto del anterior		-72,0%	-72,6%	-66,5%	-50,9%
Reducción del error respecto 1000 tiradas			-92,3%	-97,4%	-98,7%
Tiempo de ejecución de X en segundos según el nr. de tiradas	0,001	0,013	0,016	0,147	1,775



MUCHAS GRACIAS
POR VUESTRA
ATENCIÓN



L'