Simulación — ITAM
27 de abril de 2018 — Jorge de la Vega
2018-I

Tarea 3. Fecha de entrega: Sábado 24 de marzo 2018

Lecturas

- Generating Nonhomogeneous Poisson Process
- Generating a non-homogeneous Poisson Process
- Casella y Robert, capítulo 2
- Dagpunar, Capítulos 3 y 4. Secciones 7.1-7.3

Problemas

1. Una cadena de Markov de nacimiento y muerte es un proceso estocástico con un espacio de estados $\mathfrak{S}$ número numerable infinito y dos tipos de transiciones: *nacimientos* de i a $i+1$ y *muertes* de i a $i-1$. Definan las probabilidades de transición como:

$$P_{ij} = \begin{cases} q_i & j = i-1 \\ p_i & j = j+1 \\ 1 - p_i - q_i & j = i \\ 0 & \text{en otro caso,} \end{cases}$$

para $0 \leq p_i, q_i$ y $p_i + q_i \leq 1$.

   - Describir la matriz de transiciones de probabilidad.
   - Obtener 500 simulaciones de este proceso, considerando cualquier estado inicial.

Solución.

En este planteamiento general no dí un valor específico para las probabilidades de nacimiento y muerte. Así que para hacer la segunda parte del problema necesitamos hacer algunos supuestos.

Este proceso tiene una cadena de Markov con un número infinito numerable de estados. Entonces la matriz de transición es una matrix infinita, que se ve de la siguiente forma:

$$\mathbf{P} = \begin{array}{c} \\ \vdots \\ i-1 \\ i \\ i+1 \\ \vdots \end{array} \begin{array}{c} \begin{array}{cccccccc} \dots & i-3 & i-2 & i-1 & i & i+1 & i+2 & \dots \end{array} \\ \left[\begin{array}{cccccccc} \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & 0 & q_{i-1} & 1-p_{i-1}-q_{i-1} & p_{i-1} & 0 & 0 & \dots \\ \dots & 0 & 0 & q_i & 1-p_i-q_i & p_i & 0 & \dots \\ \dots & 0 & 0 & 0 & q_{i+1} & 1-p_{i+1}-q_{i+1} & p_{i+1} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{array}\right] \end{array}$$

Para generar una muestra aleatoria de una cadena con esta estructura, podemos inicializar la cadena en un valor cualquiera, por ejemplo $X_0 = 1$ y generar las 500 observaciones generando distribuciones 'al azar' (3 números no negativos que sumen 1, significando p_i, q_i y $1 - p_i - q_i$. Para poder considerar los casos en que la serie se mueve siempre hacia arriba o hacia abajo, podemos generar 1000 distribuciones posibles.

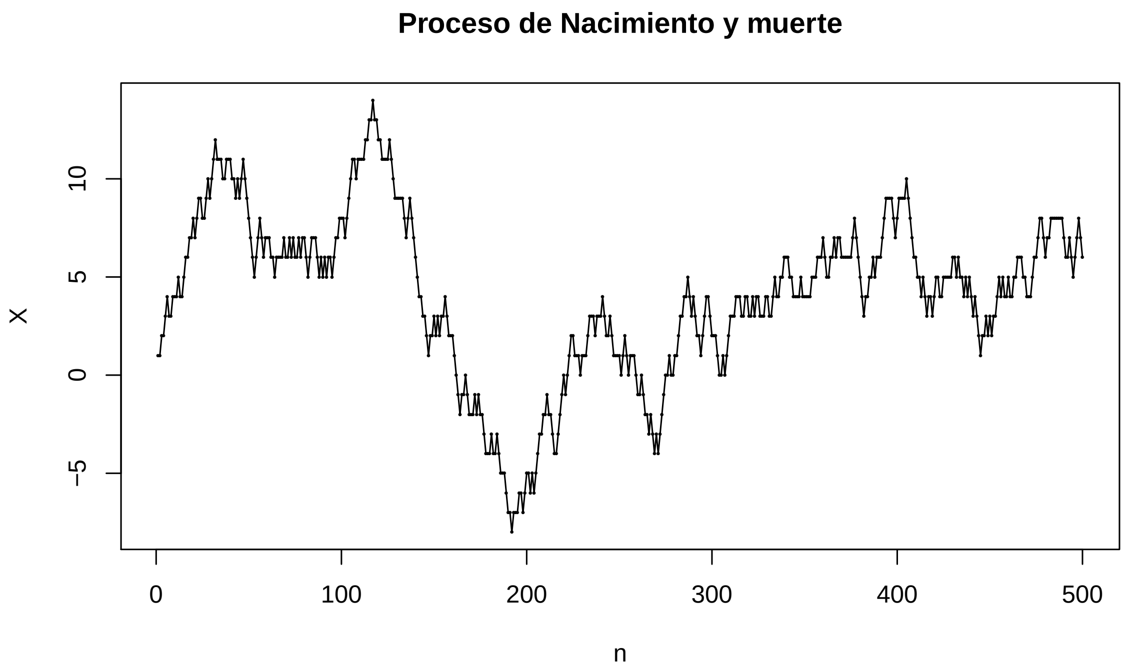
```
#matriz de distribuciones 'al azar'
U <- matrix(0, nrow=1000,ncol=3)
for(i in 1:1000){
u <- runif(3)
U[i,] <- u/sum(u)
}
head(U) #Ejemplo de las distribuciones generadas (q,1-p-q,p)
```

```
           [,1]      [,2]      [,3]
[1,] 0.60917655 0.2732258 0.1175977
[2,] 0.30775920 0.3156136 0.3766272
[3,] 0.35232983 0.2425189 0.4051513
[4,] 0.20995736 0.4287416 0.3613011
[5,] 0.21167467 0.3068334 0.4814920
[6,] 0.06778281 0.2989157 0.6333015
```

```
X <- 1 #Valor inicial de la cadena
S <- c(0,1,2) #Espacio de estados que se irá actualizando
for(i in 2:500){
        X[i] <- sample(S,1,prob=U[i,])
        S <- c(X[i]-1,X[i],X[i]+1)
}
X #Cadena generada
```

```
  [1]  1  1  2  2  3  4  3  3  4  4  4  5  4  4  5  6  6  7  7  8  7  8  9
 [24]  9  8  8  9 10  9 10 11 12 11 11 11 10 10 11 11 11 10 10  9 10  9 10
 [47] 11 10  9  8  7  6  5  6  7  8  7  6  7  7  7  6  6  5  6  6  6  6  7
 [70]  6  6  7  6  7  6  6  7  6  7  7  6  5  6  7  7  7  6  5  6  5  6  5
 [93]  6  6  5  6  7  7  8  8  8  7  8  9 10 11 11 10 11 11 11 11 12 12 13
[116] 13 14 13 13 12 12 11 11 11 11 12 11 10  9  9  9  9  9  8  7  8  9  8
[139]  7  6  5  4  4  3  3  2  1  2  2  3  2  3  2  3  3  4  3  2  2  2  1
[162]  0 -1 -2 -1 -1  0 -1 -2 -2 -2 -1 -2 -1 -2 -2 -3 -4 -4 -4 -3 -4 -4 -3
[185] -4 -5 -5 -5 -6 -7 -7 -8 -7 -7 -7 -6 -6 -7 -6 -5 -5 -6 -5 -6 -5 -4 -3
[208] -3 -2 -2 -1 -2 -2 -3 -4 -4 -3 -2 -1  0 -1  0  1  2  2  1  1  1  0  1
[231]  1  1  2  3  3  3  2  3  3  3  4  3  2  2  3  2  1  1  1  1  0  1  2
[254]  1  0  1  1  1  0 -1 -1  0 -1 -2 -2 -3 -2 -3 -4 -3 -4 -3 -2 -1  0  0
[277]  1  0  0  1  1  2  3  3  4  4  5  4  3  4  3  2  2  1  2  3  4  4  3
[300]  2  2  2  1  0  0  1  0  1  2  3  3  3  4  4  4  3  3  4  4  3  3  4
[323]  3  4  4  3  3  3  4  4  3  3  4  5  4  4  5  5  6  6  6  5  5  4  4
[346]  4  4  5  4  4  4  4  4  5  5  5  6  6  6  7  6  5  5  6  6  7  6  7
[369]  7  6  6  6  6  6  6  7  8  7  6  5  4  3  4  4  5  5  6  5  6  6  6
[392]  7  8  9  9  9  9  8  7  8  9  9  9  9 10  9  8  7  6  6  5  5  4  5
[415]  4  3  4  4  3  4  5  5  4  4  5  5  5  5  5  6  6  5  6  5  5  4  5
[438]  4  5  4  3  4  3  2  1  2  2  3  2  3  2  3  3  4  5  4  5  4  4  5
[461]  4  4  5  5  6  6  6  5  5  4  4  4  5  6  6  7  8  8  7  6  7  7  8
[484]  8  8  8  8  8  8  7  6  6  7  6  5  6  7  8  7  6
```

```
plot(X, type = "o", pch = 16, cex = 0.3,
        main = "Proceso de Nacimiento y muerte", xlab="n")
```



□

2. Para un proceso Poisson no homogéneo con función de intensidad dada por

$$\lambda(t) = \begin{cases} 5, t \in (1,2], (3,4], \cdots \\ 3, t \in (0,1], (2,3], \cdots \end{cases}$$

   a) Grafiquen una ejemplo del proceso considerando el intervalo de tiempo [0,100].

   b) Grafiquen el proceso hasta obtener 100 eventos

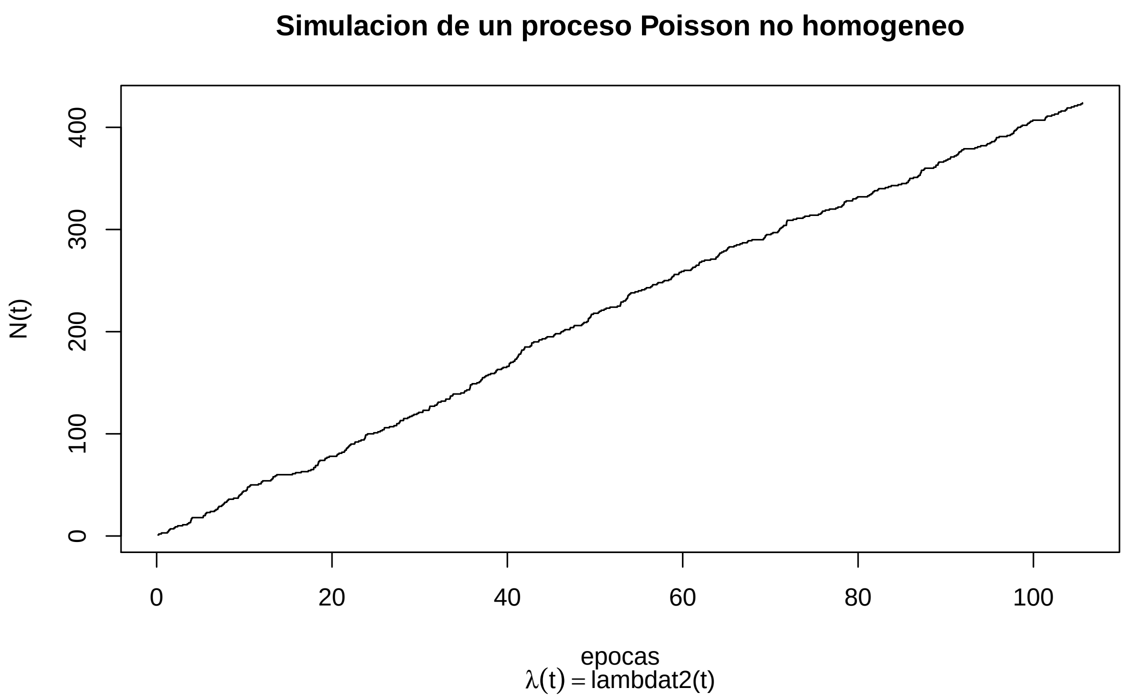
   c) Estimen la probabilidad de que el número de eventos observados en el periodo de tiempo (1.25,3] es mayor que 2.

Solución.

Para el primer problema, tenemos que definir la función pulso. Debido a que en el proceso de aceptación-rechazo se pierden algunas observaciones, el numero de observaciones necesarios para llegar al tiempo 100 se obtiene por ensayo y error, o bien, cambiar la programación y primero generar todas las exponenciales hasta que se acumule el número que necesitamos. Yo hice la otra programación.

```
lambdat2 <- function(t){
x <- paste("","{",0,"<= t & t <",1,"}",sep="")
for(i in seq(2,100,2)){x <- paste(x,paste("","{",i,"<= t & t <",i+1,"}",sep=""),sep="|")}
return(ifelse(eval(parse(text=x)),3,5))
}

poisson.nohomogeneo<-function(lambdat,n,pic=T){
lambda <- 5 #mayoriza la función lambdat
TT <- rexp(n,lambda) #genera variables exponenciales para los tiempos.
s <- cumsum(TT) #acumula los tiempos en el vector s
```

```r
u <- runif(n) #obten n uniformes
ss <- s[u <= lambdat(s)/lambda] #obten los tiempos que cumplen la condición de aceptación
Ns <- 1:length(ss) # Conteo
if(pic==T){
        plot(ss, Ns, type = "s", xlab = "epocas", ylab = "N(t)",
                        main = "Simulacion de un proceso Poisson no homogeneo",
                        sub = expression(lambda(t) == paste(lambdat2,"(t)")))
}
return(list(epocas = ss, cuenta=  Ns))
}

poisson.nohomogeneo(lambdat2,510)
```



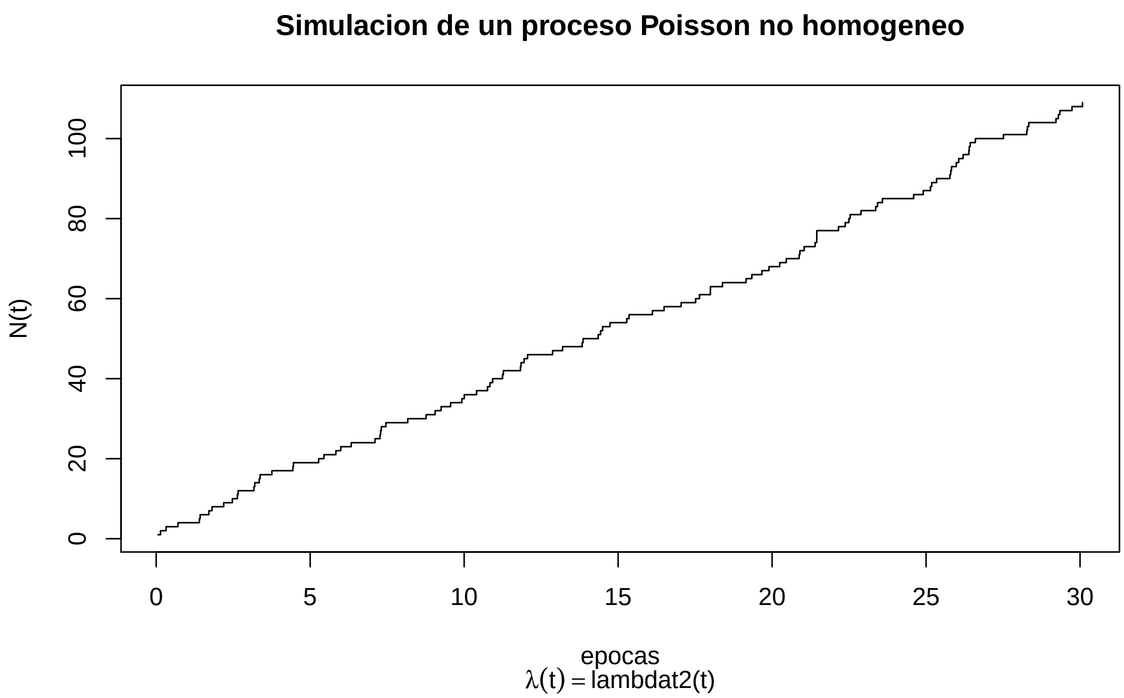

```
$epocas
  [1]   0.1554006   0.2446000   0.5428846   1.2308382   1.3437892
  [6]   1.4117306   1.5353764   1.9784658   2.1163233   2.3954954
 [11]   2.9630643   3.4895281   3.6616022   3.8620738   3.8898324
 [16]   3.9359886   3.9594790   4.0459136   5.3076061   5.3421248
 [21]   5.5550840   5.5668621   5.6917135   6.1176372   6.5983703
 [26]   6.7671288   6.9711590   7.0058973   7.0979261   7.4134397
 [31]   7.5568660   7.6750993   7.7673842   7.9870725   8.0773503
 [36]   8.2001209   8.7706298   9.3121074   9.3446669   9.4239244
 [41]   9.5750548   9.7149927   9.7612740   9.8919125  10.2052723
 [46]  10.3292382  10.3437038  10.3725449  10.5892691  10.7040460
 [51]  11.6107721  11.9122742  11.9581618  12.0853792  13.0259910
 [56]  13.1546402  13.2720918  13.2810534  13.5193987  13.6965142
 [61]  15.4918907  15.8415807  16.4884130  17.2917331  17.5983408
 [66]  17.9058043  17.9140105  18.1165380  18.1222223  18.3846956
 [71]  18.4112073  18.4564236  18.4741859  18.6040382  19.1823122
 [76]  19.2179109  19.4204984  19.6819235  20.5417841  20.6144358
 [81]  20.7886845  21.1181311  21.4092466  21.4737837  21.6258136
 [86]  21.6589016  21.7989506  21.9080873  22.0113243  22.1518877
 [91]  22.5796626  22.6328939  23.0405365  23.3147274  23.6716463
 [96]  23.7426195  23.8017312  23.8094284  23.8449803  24.0444647
[101]  24.7609361  25.2307280  25.5105036  25.7375538  25.9450652
[106]  25.9760983  26.5531030  27.0647690  27.3739583  27.3924425
[111]  27.6474692  27.7358985  27.7780904  28.1573713  28.1698904
[116]  28.6416601  28.8572965  29.1286964  29.3363813  29.6875562
[121]  29.8810283  30.3608132  30.3816721  31.0635472  31.0983239
[126]  31.1418739  31.1637821  31.7190019  31.9592312  32.0245785
```

```
[131]  32.1023364  32.4398202  32.9462421  32.9790142  33.4341871
[136]  33.4965461  33.5013284  33.7450404  33.8196401  34.6866599
[141]  35.0814918  35.1434145  35.3623576  35.6893753  35.7257141
[146]  35.7642983  35.7839267  35.7942476  35.9778144  36.5204625
[151]  36.7979621  36.9476104  37.0090360  37.1367704  37.1651566
[156]  37.4094353  37.5547295  37.8045389  38.0893780  38.5538985
[161]  38.6999098  38.7138263  38.8433948  39.3343195  39.4987141
[166]  39.9376273  40.1968322  40.2112303  40.2316045  40.3747305
[171]  40.6688181  40.8480508  40.8800261  41.0554416  41.1227367
[176]  41.2109226  41.2549099  41.3233782  41.5232908  41.5641647
[181]  41.6014835  41.6434468  41.8786853  41.9589859  41.9846320
[186]  42.5915999  42.7537706  42.7725271  42.7800767  42.9954080
[191]  43.5698161  43.6210965  43.9608283  44.3692672  44.5200459
[196]  45.2679030  45.3381721  45.4723895  46.0600514  46.1442677
[201]  46.3721188  46.5533921  47.1327674  47.1805318  47.5595651
[206]  47.6253265  48.4757807  48.6085295  48.7316241  49.0685267
[211]  49.2192376  49.2392566  49.2469025  49.3853245  49.5044909
[216]  49.5451304  49.5801358  49.8179361  50.3718605  50.4969299
[221]  50.6994869  51.0394222  51.2541499  51.7010775  52.5423370
[226]  52.8880439  52.8989933  52.9493323  52.9501786  53.2211866
[231]  53.4673222  53.5549339  53.6553662  53.7323898  53.7347531
[236]  53.8124413  53.9340079  54.0771841  54.5537275  54.9348789
[241]  55.3100079  55.6756447  55.8494790  56.3391089  56.4988267
[246]  56.5794934  57.0594817  57.1733084  57.7205348  57.8758112
[251]  58.4080613  58.6690484  58.7260679  58.8076431  58.9779431
[256]  59.0254099  59.5489301  59.6016752  59.8326854  60.1519019
[261]  60.9412645  61.0550655  61.1535692  61.4363530  61.5457800
[266]  61.8570972  61.8576325  61.9254351  62.1586850  62.4866236
[271]  63.1747176  63.7838370  63.8094664  63.9918213  64.0816862
[276]  64.1657019  64.2387573  64.4510031  64.6806394  64.9610277
[281]  65.0902663  65.1358677  65.2784858  65.8550713  66.1272233
[286]  66.5321712  66.8002410  67.3563021  67.4531962  67.8944439
[291]  69.1998563  69.2930916  69.3982087  69.4216770  69.5503963
[296]  70.0770814  70.2784359  70.8431739  70.9457500  71.0251948
[301]  71.0817005  71.2373912  71.4100201  71.5158604  71.8246767
[306]  71.8361457  71.8434015  71.8664456  71.9178494  72.5967510
[311]  72.9914812  73.7303904  73.9179872  74.4764715  75.4809549
[316]  75.7490008  75.8677018  75.9388350  76.2662132  76.7275576
[321]  77.4599831  77.6972929  78.1255930  78.2337704  78.3679499
[326]  78.4179498  78.4384832  78.6310552  79.3686135  79.4079007
[331]  79.7817783  79.9342557  81.0883738  81.2902043  81.4849721
[336]  81.6460728  81.7071503  81.8637425  82.2344686  82.3462206
[341]  83.1008035  83.4580560  83.7780749  84.5800430  84.9654276
[346]  85.5371673  85.7169314  85.7483508  85.8680553  85.8943533
[351]  86.3120277  86.7915178  86.9119744  87.0858993  87.1001556
[356]  87.1564302  87.2170852  87.2274078  87.5010613  87.6077901
[361]  88.6259785  88.8759877  88.8913620  89.1184615  89.1204573
[366]  89.2106656  89.7715812  90.0240604  90.2428457  90.5185678
[371]  90.5631550  90.9808901  91.2141278  91.3849810  91.4695410
[376]  91.5260685  91.7448440  91.8418270  92.0517531  93.3207364
[381]  93.6316263  93.9835851  94.6423382  94.7513903  95.0536885
[386]  95.2358142  95.5702989  95.6732628  95.7549123  95.7861845
[391]  96.0881728  96.9922924  97.3599971  97.5595416  97.7519691
[396]  97.7634167  97.8465781  98.0327586  98.1307726  98.2213262
[401]  98.5402874  98.7110812  99.2639346  99.3600433  99.5418364
[406]  99.6803122  99.9187488 101.3123783 101.3459100 101.4107796
[411] 101.5512848 102.0873565 102.4122375 102.8356348 102.8771079
[416] 103.1443766 103.6372564 103.7775400 103.8394305 104.3266112
[421] 104.6533279 105.0284624 105.4120130 105.5946194

$cuenta
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
 [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
 [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
 [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
 [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
 [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
[103] 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
[120] 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136
[137] 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153
[154] 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170
[171] 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187
[188] 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204
[205] 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221
[222] 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238
[239] 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255
```

```
[256] 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272
[273] 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289
[290] 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306
[307] 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323
[324] 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340
[341] 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357
[358] 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374
[375] 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391
[392] 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408
[409] 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424
```

Para obtener 100 eventos es similar, con un tamaño de muestra mucho menor:

```
poisson.nohomogeneo(lambdat2,140)
```




```
$epocas
  [1]  0.05931528  0.14070411  0.32098952  0.70692082  1.40326880
  [6]  1.42702743  1.71105955  1.81324275  2.19331680  2.47378429
 [11]  2.63777602  2.66007991  3.17354399  3.20009795  3.34666987
 [16]  3.37436597  3.75692039  4.44082255  4.45692234  5.27469690
 [21]  5.44682523  5.83638083  5.99400294  6.33397161  7.10655529
 [26]  7.26846813  7.28927706  7.31183913  7.45851541  8.16883507
 [31]  8.76445368  9.05697606  9.25190603  9.56029390  9.92620270
 [36] 10.00451482 10.40461675 10.76030644 10.83904332 10.92411429
 [41] 11.24727820 11.27383090 11.82951200 11.84797905 11.94640865
 [46] 12.05923358 12.87196413 13.19600804 13.83293541 13.86216906
 [51] 14.35488073 14.43067753 14.49748219 14.73576521 15.27914353
 [56] 15.35534828 16.11425706 16.49252243 17.04620630 17.51680380
 [61] 17.64355535 17.99600179 17.99774023 18.38968366 19.15434558
 [66] 19.34272919 19.66891541 19.89918861 20.24738027 20.45972905
 [71] 20.87806276 20.90405703 21.04146253 21.39610621 21.44879334
 [76] 21.45222961 21.45298194 22.15534785 22.37455505 22.49390817
 [81] 22.53361535 22.88602040 23.36307909 23.42389906 23.58407679
 [86] 24.59637688 24.90784498 25.14286968 25.18332950 25.34165451
 [91] 25.77260110 25.80194782 25.82569955 25.98012886 26.05996157
 [96] 26.20151309 26.39073530 26.40183248 26.43559844 26.60110613
[101] 27.51444986 28.27198948 28.29041944 28.33621260 29.21860071
[106] 29.29673932 29.34574761 29.73897545 30.07944667
```

```
$cuenta
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
 [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
 [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
 [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
 [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
 [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
[103] 103 104 105 106 107 108 109
```

Por último, para estimar la probabilidad en el intervalo dado, lo que podemos hacer es obtener N simulaciones, y calcular la proporción de esas simulaciones que dan un valor de conteo mayor a 2 en ese intervalo.

```
fr <- NULL
N <- 10000
for(i in 1:N){
      x <- poisson.nohomogeneo(lambdat2,10,pic=F);
      fr <- c(fr,ifelse(1.25 < x$epocas[x$cuenta==2] & x$epocas[x$cuenta==2] <3,1,0))}
      sum(fr)/N


[1] 0.0751
```

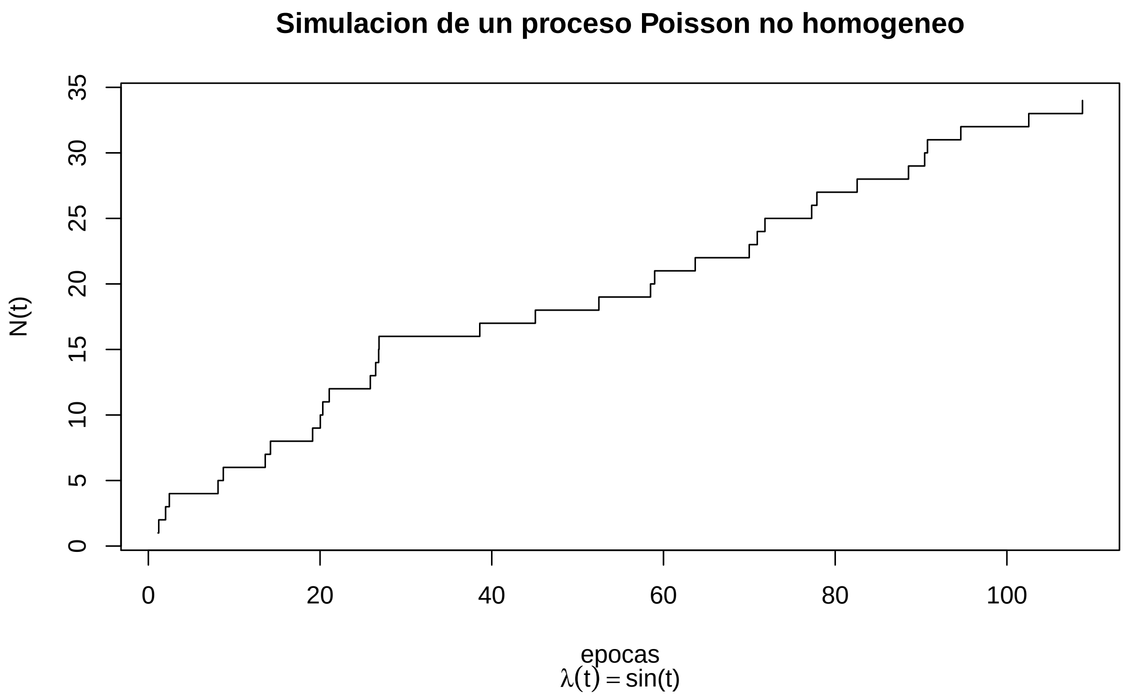
□

3. Simular un proceso Poisson no homogéneo con función de intensidad dada por $\lambda(t) = sen(t)$

Solución.

Este problema fue el que intenté resolver en clase la última vez y el programa no corrió... Ya lo corregí y comenté y aquí lo incluyo.

```
lambdat <- function(t){sin(t)}

poisson.nohomogeneo<-function(lambdat,n){

lambda <- 1 #mayoriza la función lambdat
TT <- rexp(n,lambda) #genera variables exponenciales para los tiempos.
s <- cumsum(TT) #acumula los tiempos en el vector s
u <- runif(n) #obten n uniformes
ss <- s[u <= lambdat(s)/lambda] #obten los tiempos que cumplen la condición de aceptación
Ns <- 1:length(ss) # Conteo
plot(ss, Ns, type = "s", xlab = "epocas", ylab = "N(t)",
main = "Simulacion de un proceso Poisson no homogeneo",
sub = expression(lambda(t) == paste("sin","(t)")))
return(list(epocas = ss, cuenta=  Ns))
}
x <- poisson.nohomogeneo(lambdat,100)
```




```
x

$epocas
 [1]   1.125216   1.212807   2.006949   2.446002   8.117517   8.730100
 [7]  13.615097  14.223802  19.130666  20.030652  20.319614  21.068995
[13]  25.854079  26.475694  26.824788  26.867819  38.603394  45.077050
[19]  52.475893  58.490752  58.970735  63.696917  69.987419  70.922497
[25]  71.820591  77.257359  77.867023  82.554778  88.537264  90.418397
[31]  90.749371  94.631761 102.546474 108.802561

$cuenta
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
[24] 24 25 26 27 28 29 30 31 32 33 34
```

□

4. Para un movimiento Browniano, encontrar:

   - $P(B_2 \leq 1)$
   - $E(B_4|B_1 = x)$
   - $Corr(B_{t+s}, B_s)$
   - $\text{Var}(B_4|B_1)$
   - $P(B_3 \leq 5|B_1 = 2)$

   Solución.

   - Como $B_2 \sim \mathcal{N}(0, 2)$, entonces $P(B_2 \leq 1) = P(Z \leq 1/\sqrt{2}) = \Phi(1/\sqrt{2})$

```r
pnorm(1/sqrt(2))
```

```
[1] 0.7602499
```

- $E(B_4|B_1 = x) = E(B_4 - B_1 + B_1|B_1 = x) = E(B_4 - B_1|B_1 = x) + x = E(B_4 - B_1) + x = E(B_3) + x = x$. La penúltima desigualdad es porque $B_3 \perp\!\!\!\perp B_1$. Entonces $E(B_4|B_1 = x) = x$.
- Recordando la definición de correlación: $cor(B_{t+s}, B_s) = \frac{\text{cov}((,B)_{t+s}, B_s)}{ds(B_{t+s}ds(B_s))} = \frac{min(t+s,s)}{\sqrt{t+s}\sqrt{s}} = \frac{s}{\sqrt{s(t+s)}}$.
- $Var(B_4|B_1) = Var(B_4 - B_1 + B_1|B_1) = Var(B_4 - B_1|B_1) + Var(B_1|B_1) = Var(B_4 - B_1) = Var(B_3) = 3$.
- $P(B_3 \leq 5|B_1 = 2) = P(B_3 - B_1 \leq 5 - B_1|B_1 = 2) = P(B_3 - B_1 \leq 3) = P(B_2 \leq 3)$

```
pnorm(3,0,sqrt(2))

[1] 0.9830526
```

$\square$

5. Supongan que X,Y son iid $\mathcal{U}(0,1)$ y definan Z como

$$Z = \begin{cases} 1 & \text{si } X^2 + Y^2 \leq 1 \\ 0 & \text{en otro caso} \end{cases}$$

- Obtener $E(Z)$.
- Simulando Z, escribir un programa para estimar π.

Solución.

Para obtener el valor de $E(Z)$, basta con simular puntos en el cuadrado $[0,1] \times [0,1]$, y quedarnos con los puntos que cumplen estar dentro de la circunferencia.

```
AreaSector <- function(n){
x <- runif(n); y <- runif(n)
z <- ifelse(x^2+y^2<=1,1,0)
return(sum(z)/n)
}
AreaSector(10000)

[1] 0.7839
```

Esto corresponde a la cuarta parte de la circunferencia. Entonces, para estimar π, debemos resolver la ecuación $4E(Z) = \pi r^2 = \pi$

```
pihat <- 4*AreaSector(10000000)
pihat

[1] 3.141426
```

$\square$

6. Supongan que la acción XYZ se vende hoy por \$80 por acción y sigue un movimiento browniano geométrico con drift 0.10 y volatilidad 0.5. Encuentren la probabilidad de que en 90 días el precio de XYZ se eleve a por lo menos \$100.

Solución.

Si consideramos que la unidad de valuación es 1 año, entonces, si S_t denota el precio de la acción XYZ después de t años, y considerando que 90 días es 1/4 de año,

$$\begin{aligned} P(S_{0.25} \geq 100) &= P(80e^{0.25\mu+\sigma Z_{0.25}} \geq 100) \\ &= P(0.1 * 0.25 + 0.5 * Z_{0.25} \geq \log(100/80)) \\ &= P(Z_{0.25} \geq 0.396) = 0.214 \end{aligned}$$

```
1-pnorm((log(100/80)-0.1/4)/0.5,0,sqrt(0.25))

[1] 0.214013
```

□

7. El precio de una acción se modela con un movimiento Browniano geométrico con drift $\mu = -0.25$ y volatilidad $\sigma = 0.4$. La acción actualmente se vende a \$35. Supongan que hay una opción para comprar esa acción en 6 meses a \$40. Encuentren la ganancia esperada de la opción.

   Solución.

   En este problema, sabemos que la discretización del Browniano geométrico es $S_t = S_0 e^{\mu\Delta t + \sigma\epsilon\sqrt{\Delta t}}$. En el problema dado, $S_0 = 35$, $\mu = -0.25$, $\sigma = 0.40$ y $Deltat = 0.5$. Entonces, podemos generar una muestra de tamaño n de los valores $S_{0.5} = 35 * \exp(-0.25 * 0.5 + 0.4\sqrt{0.5}\epsilon)$ y calcular el estimador de $E(max(0, S_{0.5} - 40))$ con el promedio $\frac{\sum_{i=1}^{n} max(0,\hat{S}_{0.5,i}-40)}{n}$.

```
n <- 1000000
S05 <-NULL
S05  <- 35*exp(-0.25*0.5+0.4*sqrt(0.5)*rnorm(n,0,1))
sum(ifelse(S05-40<0,0,S05-40))/n

[1] 1.273901
```

□

8. En varios problemas de la vida real nos interesa calcular con bastante precisión probabilidades en las colas. Supongamos que nos interesa estimar $P(X > 20)$, con $X \sim \mathcal{N}(0,1)$. Pueden comprobar que simular X no servirá. La mejor forma de resolver el problema es expresar esa probabilidad como una integral y usar un cambio de variable para reescribir esa integral como una esperanza bajo la distribución $\mathcal{U}(0, 1/20)$. Deduzcan una aproximación de Monte Carlo a $P(X > 20)$ junto con una estimación de error.

   Solución.

   El ejercicio propone hacer un cambio de variable para tener mayor control sobre la región que interesa integrar.

   Noten que

$$\begin{aligned} P(X > 20) = \int_{20}^{\infty} \phi(x)dx &= \int_{1/20}^{0} \phi(1/u)(-1/u^2)du \\ &= \int_{0}^{1/20} \phi(1/u)(1/u^2)du \\ &= 20E_U(\phi(1/u)/u^2) \end{aligned}$$

donde el valor esperado lo tomamos de $U \sim \mathcal{U}(0, 1/20)$. Entonces:

```
n <- 1000000
u <- runif(n,0,1/20)
theta <- 20*mean((1/u^2)*dnorm(1/u,0,1))
theta

[1] 1.107071e-86
```

□

9. Marginalización de Monte Carlo es una técnica para calcular una densidad marginal cuando se simula de una densidad conjunta. Sea $(X_i, Y_i) \sim f_{XY}(x, y)$, independiente, y la correspondiente densidad marginal $f_X(x) = \int f_{XY}(x, y)dy$.

   - Sea $w(x)$ una densidad arbitraria. Mostrar que

     $$lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \frac{f_{XY}(x^*, y_i)w(x_i)}{f_{XY}(x_i, y_i)} = \int \int \frac{f_{XY](x^*,y)w(x)}}{f_{XY}(x, y)} f_{XY}(x, y)dxdy = f_X(x^*).$$

     La fórmula anterior provee un estimado de Monte Carlo de f_X, cuando la distribución conjunta es conocida salvo una constante.
   - Sea $X|Y = y \sim \mathcal{G}(y, 1)$ y $Y \sim \exp(1)$. Usar la técnica de arriba para graficar la densidad marginal de X (pueden usar cualquier densidad). Comparar con la marginal exacta.
   - Mostrar que si se elije $w(x) = f_X(x)$ funciona para producir la distribución marginal y que es óptima en el sentido de que la varianza del estimador resultante es menor.

Solución.

- Argumentemos primero la primera igualdad. Por definición

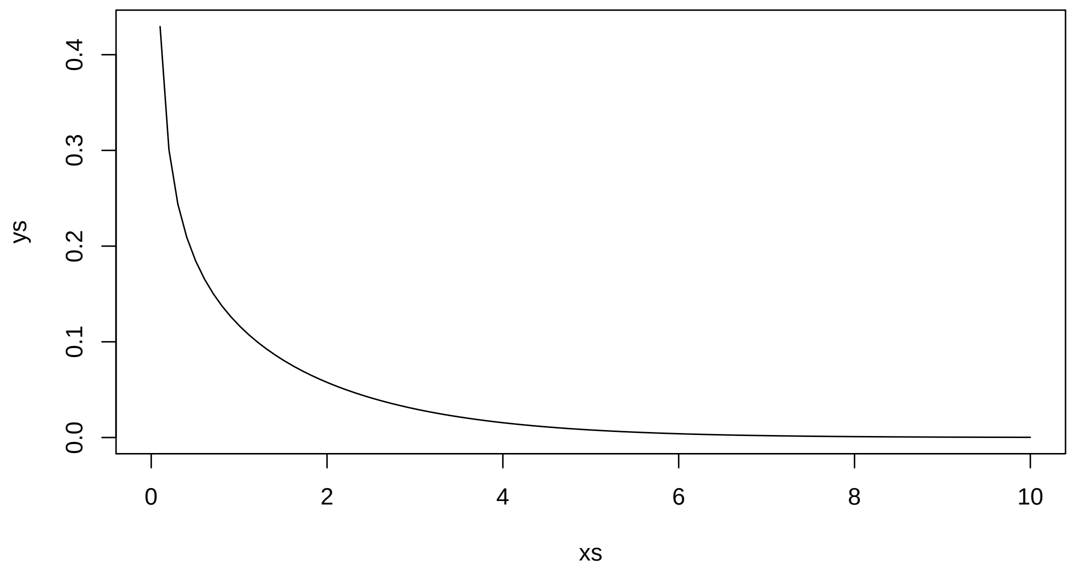
  $$\int \int \frac{f_{XY}(x^*, y)w(x)}{f_{XY}(x, y)} f_{XY}(x, y)dxdy = E_{(X.Y)} \left[\frac{f_{XY}(x^*, y)w(x)}{f_{XY}(x, y)}\right]$$

  Entonces, aplicando la Ley de los grandes números, obtenemos el primer resultado, que es aplicar a la muestra (X_i, Y_i) la función $\frac{f_{XY}(x^*,y)w(x)}{f_{XY}(x,y)}$ y promediar respecto a f_{XY}.
  Para la segunda parte, notando a la misma integral como función de $w(x)$ y aplicando Fubini:

  $$\int \int f_{XY}(x^*, y)w(x)fdxdy = \int f_{XY}(x^*, y) \left[\int w(x)dx\right] dy = \int f_{XY}(x^*, y)dy = f_X(x^*)$$

- Para aplicar el resultado, supongamos que $w(x) = \phi(x)$, la distribución normal estándar. Además, la expresión para $f_{XY}(x, y) = f(x|y)f_y(y)$ se simplifica, porque la expresión para $f_Y(y) = e^{-y}$ se cancela en el numerador y el denominador. Entonces el muestreo queda de la siguiente manera:

  ```
  n <- 10000
  y <- rexp(n,1)
  x <- rgamma(n,y,1)
  xs <- seq(0,10,length=100)
  ys <- NULL
  for (i in xs){
  ys[match(i,xs)] <- mean(dgamma(i,y,1)*dnorm(x)/dgamma(x,y,1))
  }
  plot(xs,ys,xlim=c(0,10),type="l")
  ```



Sólo hay un problema para poder comparar con la marginal exacta. Con la información dada, la marginal exacta sería de la forma:

$$f(x) = \int_0^\infty \frac{1}{\Gamma(y)} x^{y-1} e^{-x} e^{-y} dy$$

La cual no se puede integrar explícitamente. Muy probablemente el autor (Robert-Casella) cometieron un error al escribir los parámetros de la densidad gamma, invirtiendo su posición.

- Si se elige $w(x) = f_X(x)$, como indica esta sección, las ecuaciones quedan como:

$$\frac{1}{n}\sum_{i=1}^{n} \frac{f_{XY}(x^*, y_i) f_X(x_i)}{f_{XY}(x_i, y_i)} = \frac{1}{n}\sum_{i=1}^{n} \frac{f_X(x^*) f_{Y|X}(y_i|x^*) f_X(x_i)}{f_X(x_i) f_{Y|X}(y_i|x_i)} = f_X(x^*) \frac{1}{n}\sum_{i=1}^{n} \frac{f_{Y|X}(y_i|x^*)}{f_{Y|X}(y_i|x_i)}$$

Entonces el promedio produce $f_X(x^*)$ por una constante que estima al 1.

Ahora, para demostrar que se reduce la varianza, descomponemos la varianza del estimador del siguiente modo:

$$\begin{aligned} Var\left(\frac{f_{XY}(x^*, y_i) f_X(x_i)}{f_{XY}(x_i, y_i)}\right) &= Var\left[E\left(\frac{f_{XY}(x^*, y_i) f_X(x_i)}{f_{XY}(x_i, y_i)}\,|x_i\right)\right] + \\ &\quad E\left[Var\left(\frac{f_{XY}(x^*, y_i) f_X(x_i)}{f_{XY}(x_i, y_i)}\,|x_i\right)\right] \end{aligned}$$

El primer término es

$$E\left(\frac{f_{XY}(x^*, y_i) f_X(x_i)}{f_{XY}(x_i, y_i)} | x_i\right) = f_X(x^*) E\left(\frac{f_{Y|X}(y_i|x^*)}{f_{Y|X}(y_i|x_i)} | x_i\right) \frac{w(x_i)}{f_X(x_i)} = f_X(x^*)$$

que es una constante, por lo que su varianza es 0.

□

10. Un hospital tiene 5 ambulancias para emergencias. El área de cobertura de casos se aproxima con un circulo de diámetro 5km, con el hospital en el centro. La distribución física de los accidentes es un proceso Poisson en el espacio, con una tasa de λ casos por hora. Si una ambulancia no está disponible el paciente tiene que esperar hasta que alguna se libere. Una ambulancia siempre toma una ruta en línea recta a la escena de la emergencia y regresa al hospital. Supongan que las ambulancias viajan a una velocidad constante de ν km/hr y que sólo se requiere una ambulancia para cada emergencia.

   - Mostrar que el tiempo total de viaje de retorno (x horas) se puede muestrear tomando $x = 10\sqrt{U}/v$ donde $U \sim \mathcal{U}(0,1)$
   - Simular el sistema para obtener para cada paciente el tiempo entre la ocurrencia de la emergencia y la llegada al hospital.

Solución.

- Si R denota la distancia del punto (X,Y) al origen, tenemos que encontrar primero la probabilidad de que $R \leq r$. Recordando lo que vimos del proceso Poisson en el espacial con parámetro λ en un conjunto A, primero se simula el número de puntos N en A de acuerdo a la distribución Poisson con parámetro $\lambda|A|$, luego se generan N puntos uniformemente distribuídos en A.
  Considerando que los puntos son uniformes,

$$P(R \leq r) = \int\int_D f_{X,Y}(x,y)\,dx\,dy$$

  donde $D = \{(x,y) \in \mathbb{R}^2 | x^2 + y^2 \leq r\}$ es el disco de diametro r. Para resolver la integral anterior, sabemos que X y Y son uniformes independientes en $(-a,a)$ y podemos convertir a coordenadas polares para facilitar la integración:

$$\begin{aligned} P(R \leq r) &= \int\int_D f_{X,Y}(x,y)\,dx\,dy = \int\int_D f_X(x) f_Y(y)\,dx\,dy \\ &= \int\int_D \frac{1}{2a}\frac{1}{2a}\,dx\,dy \\ &= \frac{1}{4a^2}\int\int_D dx\,dy = \frac{1}{4a^2}\int_0^{2\pi}\int_0^r r dr\,d\theta \\ &= \frac{1}{4a^2}\int_0^{2\pi}\frac{r^2}{2}d\theta = \frac{2\pi r^2}{8a^2} = \frac{\pi r^2}{4a^2} \end{aligned}$$

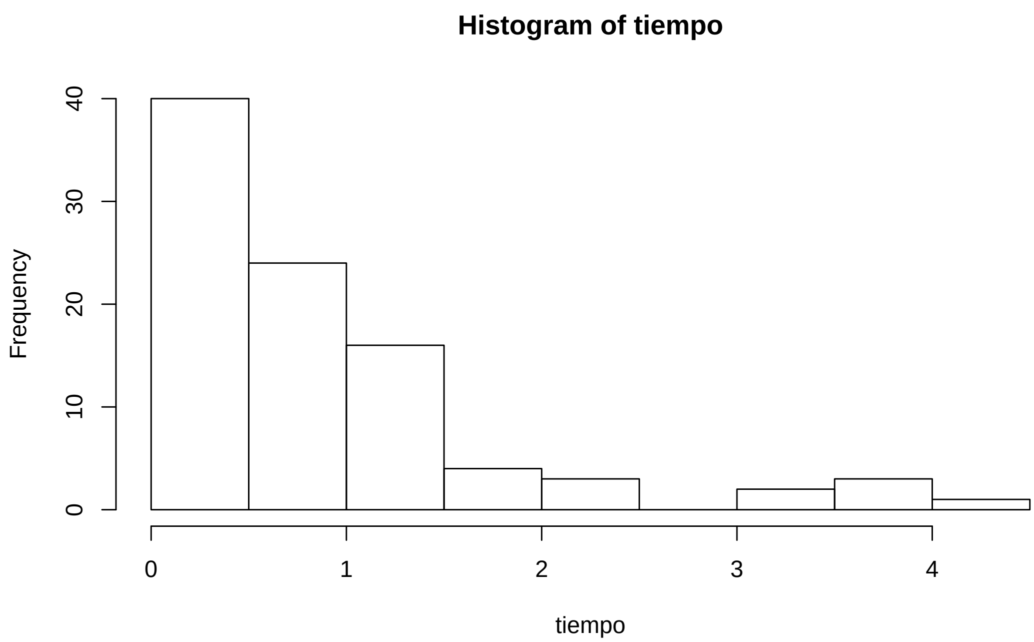
  Entonces $P(R \leq r) = P(X^2 + Y^2 \leq r^2) = \frac{\pi r^2}{4a^2}$. Ahora bien, como el punto tiene que ocurrir necesariamente en el circulo de radio 5, debemos considerar la probabilidad condicional a estar en el círculo. Entonces:

$$P(R \leq r | R \leq 5) = \frac{P(R \leq r, R \leq 5)}{P(R \leq 5)} = \frac{\pi min(5,r)^2/4a^2}{\pi 5^2/4a^2} = (r/5)^2$$

  Por lo tanto, dado que estamos en el circulo, $F_R(r) = (r/5)^2$, por lo que $r = 5\sqrt{U}$. Como el trayecto es de ida y vuelta, se obtiene que la distancia recorrida es $2r = 10\sqrt{U}$. Por último, como el tiempo está dado por la fórmula $t = d/v$, entonces $t = 10\sqrt{U}/v$.

- El siguiente código simula el sistema. Los tiempos están dados en el vector solución. El ejercicio asume que se simulan 100 horas del proceso.

```
simulaAmbulancias <- function(amb=5){
#inicialmente no hay siniestros y todas las ambulancias están desocupadas.
# q = número de pacientes esperando ambulancia (la cola)
# b = número de ambulancias ocupadas
# A = tiempo de la siguiente emergencia
# amb = número de ambulancias.
# TD[i]  = tiempo de dejada en hospital de la ambulancia i
# TA[i] = tiempo de levantamiento de la emergencia de la ambulancia i.
# Ar[i] = tiempo de llegada del paciente que es el j-ésimo en la cola para ambulancia.
# clck = es el reloj de simulación
# simtim = duración de la simulación

simtim <- 100
clock <- 0 #inicializa el reloj de simulación
b <- 0
q <- 0 #inicialmente no hay emergencias.
A <- rexp(1) #se genera una emergencia en el tiempo.
Ar <- tiempo <- NULL
TD <- rep(Inf,amb)
TA <- rep(Inf,amb)
while(clock < simtim){
clock <- min(A,TD)
if (clock==A) {
q <- q+1
A <- clock + rexp(1) #nueva emergencia se programa
Ar[q] <- clock
} else {
j <- which(TD==clock)
tiempo <- append(tiempo,clock-TA[j])
TA[j] <- 0
TD[j] <- Inf
b <- b-1
}
if (q > 0 && b < 5) {
j <- min(which(TD==Inf)) #identifica una ambulancia libre
TD[j] <- clock + rexp(1,1)
TA[j] <- Ar[1]
Ar[1] <- Inf
Ar <- sort(Ar)
q <- q-1
b <- b+1
}
}
return(list(tiempo=tiempo,TA=TA,TD=TD,Ar=Ar,q=q,b=b))
}
tiempo <- simulaAmbulancias()$tiempo
hist(tiempo,breaks=10)
```



□

11. Usar Monte Carlo para encontrar un intervalo de confianza del 95 % para la siguiente integral:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} exp\left\{\frac{1}{2}\left[x^2+(y-1)^2-\frac{x(y-1)}{10}\right]\right\}dxdy$$

Solución.

Cabe hacer notar que como está la integral, es probable que no converja debido a que el exponente crece sin cota. El exponente debe ser con signo menos, el error es mío en este caso.

Aunque posiblemente esto pase, propondré una solución en ambos casos.

- Para el caso de la integral como está. Esta integral se puede escribir como una integral definida con el mismo límite de integración a y pensar en muestras en el cuadrado uniforme, así que si la integral es θ,

$$\hat{\theta} = lim_{a\to\infty} 4a^2 E[h(X,Y)]$$

  con $X, Y \sim \mathcal{U}(-a, a)$ independientes.

```
theta <- function(a){
n <- 1e7
x <- runif(n,-a,a)
y <- runif(n,-a,a)
h <- exp(0.5*(x^2+(y-1)^2-x*(y-1)/10))
return(mean(4*a^2*h))
}
theta(10)

[1] 1.940047e+48
```

```r
theta(100)
```

```
[1] Inf
```

```r
theta(10000)
```

```
[1] Inf
```

- Para el caso de la integral

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} exp\left\{-\frac{1}{2}\left[x^2+(y-1)^2-\frac{x(y-1)}{10}\right]\right\}dxdy$$

  se puede usar las mismas uniformes, o notar que el kernel corresponde a una normal multivariada con media $\boldsymbol{\mu}=(0,1)$ y matriz de varianzas y covarianzas $\boldsymbol{\Sigma}=\begin{pmatrix}1 & -1/20\\ -1/20 & 1\end{pmatrix}$. Sin embargo, lo haré considerando la misma función de arriba.

```r
theta <- function(a){
n <- 1e7
x <- runif(n,-a,a)
y <- runif(n,-a,a)
h <- exp(-0.5*(x^2+(y-1)^2-x*(y-1)/10))
return(mean(4*a*a*h))
}
theta(10)
```

```
[1] 6.28662
```

```r
theta(100)
```

```
[1] 6.417803
```

```r
theta(1000)
```

```
[1] 7.373891
```

□

12. Otro modelo para simular precios de acciones es el siguiente modelo binomial: si S_i denota el precio en el tiempo ih donde $i=0,1,2,\ldots,$ y h es un incremento de tiempo positivo. Sean μ y σ la tasa de interés y la volatilidad respectivamente. Sean

$$\begin{aligned}
u &= \frac{1}{2}\left(e^{-\mu h}+e^{(\mu+\sigma^2)h}+\frac{1}{2}\sqrt{(e^{-\mu h}+e^{(\mu+\sigma^2)h})^2-4}\right)\\
\nu &= u^{-1}\\
p &= \frac{e^{\mu h}-\nu}{u-\nu}
\end{aligned}$$

Entonces

$$S_i = X_i S_{i-1}$$

donde $X_i, i=0,1,\ldots$ son variables Bernoulli independientes con distribución $P(X_i=u)=p$, $P(X_i=\nu)=1-p$ para toda i.

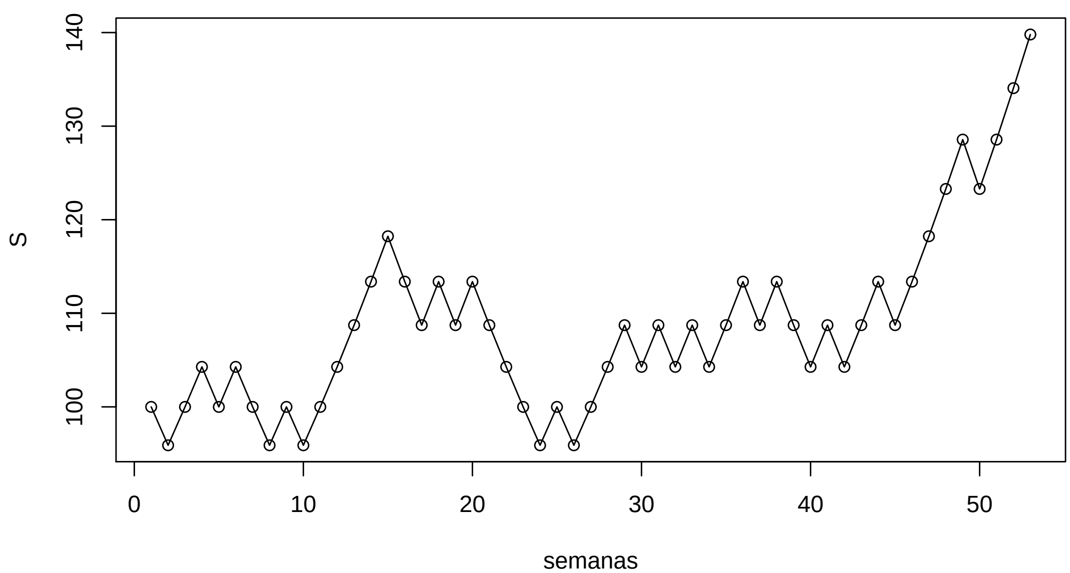
- Simular el precio al final de cada semana durante el siguiente año con $S_0=100$, $\mu=0.2$ por año, $\sigma=0.3$ por año, y $h=1/52$ años.
- Supongan que hay 252 dias hábiles en un año. Hacer $h=1/252$. Para cualquier realización, sea $S_{max}=max\{S_j|j=0,1,\ldots,756\}$. Sea pérdida $=S_{max}-S_{756}$. La pérdida denota la diferencia entre vender la acción en el pico de su precio durante los siguientes tres años y venderla después de tres años. Simular 200 realizaciones de la pérdida y construir su distribución empírica.

Solución.

La solución de este ejercicio es directa, de las definiciones dadas.

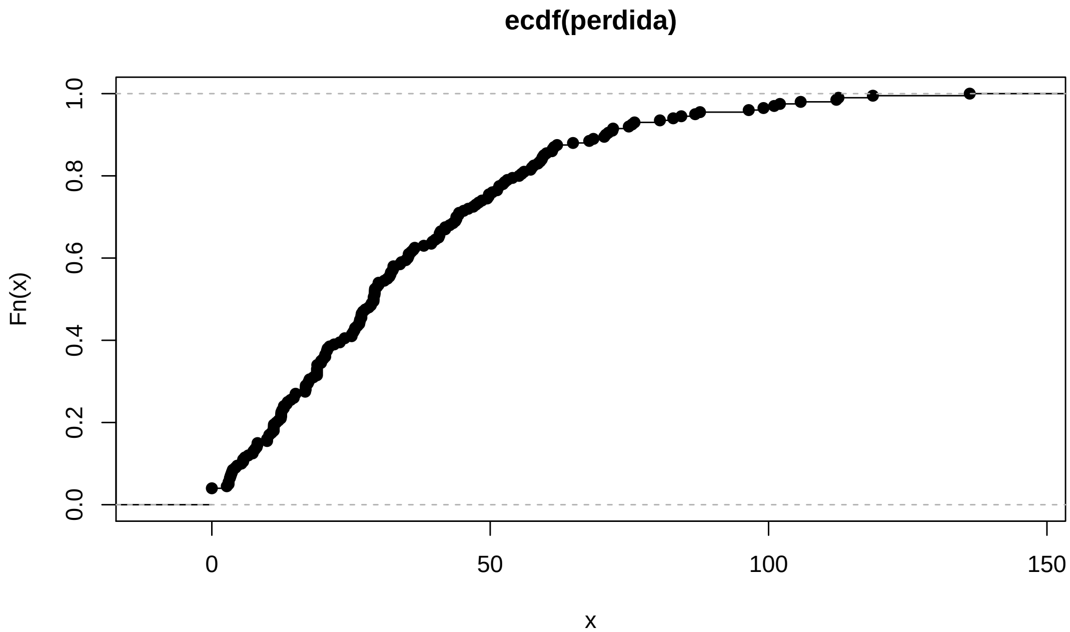
- El siguiente programa simula los precios al final de cada semana por el primer año. En un año hay 52 semanas.

```
h <- 1/52
mu <- 0.2
sigma2 <- 0.3^2
S0 <- 100

u <- 0.5*(exp(-mu*h) + exp((mu+sigma2)*h)) + 0.5*sqrt((exp(-mu*h) + exp((mu+sigma2)*h))^2-4)
v <- 1/u
p <- (exp(mu*h)-v)/(u-v)

#saltos bernoulli
X <- rbinom(52,1,p)
X <- ifelse(X==0,v,u)
S <- S0
for(i in 2:(length(X)+1)) S[i] <- X[i-1]*S[i-1]
plot(S,type="o",xlab="semanas")
```



- Aquí usamos los mismos parámetros que en el inciso anterior, excepto $h = 1/252$.

```
h <- 1/252
u <- 0.5*(exp(-mu*h) + exp((mu+sigma2)*h)) + 0.5*sqrt((exp(-mu*h) + exp((mu+sigma2)*h))^2-4)
v <- 1/u
p <- (exp(mu*h)-v)/(u-v)

perdida <- NULL
for(j in 1:200){ #200 realizaciones
X <- rbinom(756,1,p)
X <- ifelse(X==0,v,u)
S <- S0
for(i in 2:(length(X)+1)) S[i] <- X[i-1]*S[i-1]
perdida[j] <- max(S)- S[757]
}
plot(ecdf(perdida))
```



□

13. Un proceso Poisson compuesto es un proceso estocástico $\{X_t|t \geq 0\}$ que se puede representar como una suma aleatoria $X_t = \sum_{i=1}^{N_t} Y_i$ donde $\{N_t|t \geq 0\}$ es un proceso Poisson y Y_i son iid e independientes del proceso N_t. Escriban un programa para simular un proceso $\mathcal{P}(\lambda)$-$\mathcal{G}(\cdot,\cdot)$ donde Y tiene distribución gamma. Estimen la media y la varianza de X_{10} para varias elecciones de los parámetros y comparar con los valores teóricos.

Solución.

Para simular este proceso, para cada t, debemos seguir el siguiente algoritmo:

a) Genera T_i exponenciales con parámetro λ, y define $N_t = i$ donde i es tal que $\sum_{k=1}^{i} T_k \leq t$ y $\sum_{k=1}^{i+1} T_k > t$.

b) Genera N_t variables aleatorias $\mathcal{G}(\cdot,\cdot)$ y define $X_t = \sum_{i=1}^{N_t} Y_i$

Entonces podemos hacer una función que tenga como argumentos los valores que se escojan para la poisson y la gamma:

```
PoissonComp <- function(t,lambda,alfa,beta){
i <- 0
T <- 0
while (T <= 10){
T <- T + rexp(1,lambda)
i <- i+1
}
X <- sum(rgamma(i,shape=alfa,scale=beta))
return(X)
}
# Genera una muestra aleatoria para calcular la media y la varianza
X10 <- NULL
n <- 100000
for(i in 1:n)X10[i] <- PoissonComp(10,2,1,3)
mean(X10)
```

```
[1] 62.96273
```

```r
var(X10)
```

```
[1] 367.1631
```

El valor teórico de la media y varianza del proceso Poisson compuesto está dado por:

$$E(X_t) = E(\sum_{i=1}^{N_t} Y_i) = E(E(\sum_{i=1}^{N_t} Y_i|N_t)) = E(N_t E(Y_1)) = \lambda t E(Y_1)$$

Del mismo modo, la varianza está dada por:

$$\begin{aligned} Var(X_t) &= E(Var(X_t|N_t)) + Var(E(X_t|N_t)) \\ &= E(N_t Var(Y_1)) + Var(N_t E(Y_1)) \\ &= Var(Y_1)E(N_t) + E(Y_1)^2 Var(N_t) \\ &= Var(Y_1)\lambda t + E(Y_1)^2 \lambda t \\ &= \lambda t(Var(Y_1) + E(Y_1)^2) = \lambda t E(Y_1^2) \end{aligned}$$

Entonces, para la distribución $\mathcal{G}(\alpha, \beta)$, con $t = 10, \lambda = 2, \alpha = 1, \beta = 3$, $E(X_{10}) = 10\lambda\alpha\beta = 60$ y $Var(X_{10}) = 10\lambda\alpha(1+\alpha)\beta^2 = 360$

$\square$