1 Standardowy proces ruchu Browna

1.1 Proces ruchu

Funkcja *pr_r_B* jest podstawową funkcją generującą standardowy proces ruchu Browna. Argumentami tej funkcji są N i T. T oznacza koniec wektora czasu procesu, natomiast N determinuje nam podział czasu na mniejsze odcinki.

```
pr_r_B<-function(N,T){

  delta_t<-T/N

  wektor_tj<-seq(0,T, by=delta_t)

  B<-c(0)

  ksi<-rnorm(N,0,1)

  for( i in 2:(N+1)){

    B[i]<-B[i-1]+sqrt(delta_t)*ksi[i-1]

  }
  return(cbind(wektor_tj, B))
}
```

Wywołujemy funkcję z argumentami $N = 16$ i $T = 1$.

```
N<-16
T<-1
pr_r_B(N,T)
```

	wektor_tj	B
1	0.0000	0.0000000
2	0.0625	-0.4384926
3	0.1250	-0.7379566
4	0.1875	-0.9421078
5	0.2500	-0.9351102
6	0.3125	-1.2568278
7	0.3750	-0.8594092
8	0.4375	-1.0824449
9	0.5000	-0.9734669
10	0.5625	-0.9533878
11	0.6250	-1.2330899
12	0.6875	-1.3776755
13	0.7500	-1.7107002
14	0.8125	-2.2406792
15	0.8750	-2.1539525
16	0.9375	-2.5195324
17	1.0000	-2.9990000

Rysunek 1: Wygenerowany proces.

W rezultacie otrzymujemy tabelę zawierającą 17 (czyli $N+1$) momentów w czasie wraz z wygenerowanymi wartościami procesu ruchu Browna w tych momentach.

Zobaczmy jeszcze jak będzie wyglądał wygenerowany proces ruchu Browna dla $T = 1$, z podziałem czasu według $N = 32$.

```
N<-32
T<-1
pr_r_B(N,T)
```

	wektor_tj	B
1	0.00000	0.000000000
2	0.03125	-0.101918995
3	0.06250	0.062222821
4	0.09375	-0.012905442
5	0.12500	-0.211967139
6	0.15625	-0.239901756
7	0.18750	-0.238547489
8	0.21875	-0.068387649
9	0.25000	-0.361311820
10	0.28125	-0.195535234
11	0.31250	-0.197536253
12	0.34375	-0.413413254
13	0.37500	-0.255156969
14	0.40625	-0.202173472
15	0.43750	-0.146525888
16	0.46875	-0.229521148
17	0.50000	-0.025221901
18	0.53125	-0.421829008
19	0.56250	-0.418328095
20	0.59375	-0.249235559
21	0.62500	0.005238864
22	0.65625	0.003053000
23	0.68750	0.099404629
24	0.71875	0.292782778
25	0.75000	0.281940376
26	0.78125	0.608112629
27	0.81250	1.120160280
28	0.84375	0.974617328
29	0.87500	1.000095489
30	0.90625	1.023428422
31	0.93750	0.988384534
32	0.96875	1.267688631
33	1.00000	1.066773041

Rysunek 2: Wygenerowany proces.

Otrzymujemy 33 wygenerowane wartości, czas kończy się na $T = 1$.

Tabela nie przedstawia jednak dobrze procesu, dlatego w następnym podrozdziale narysujemy trajektorie.

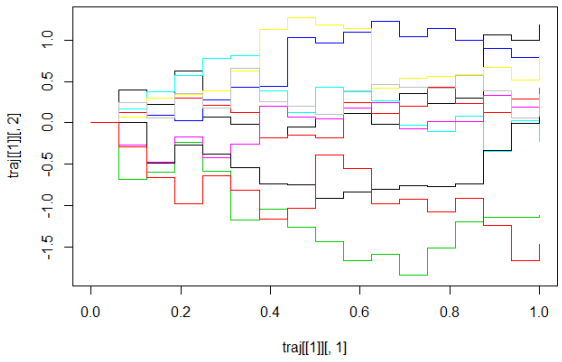
1.2 Trajektorie

Funkcja generująca wykresy trajektorii procesu ruchu Browna nazywa się *wykresy_Brown*. Jej argumentami są N, T- argumenty użyte w funkcji *pr_r_Browna* oraz M - ilość trajektorii które chcemy narysować.

```
wykresy_Brown <-function(N ,T,M){

  traj <-list()
  ymin<-c()
  ymax<-c()
  for( i in 1:M){

    traj[[i]]<-pr_r_B(N,T)
    ymin[i]<-min(traj[[i]][,2])
    ymax[i]<-max(traj[[i]][,2])
  }

  plot(traj[[1]][,1],traj[[1]][,2],type='s',ylim=c(min(ymin),max(ymax)))

  for( i in 2:M){
    lines(traj[[i]][,1],traj[[i]][,2],type='s',col=i)
  }
}
```

Zacznijmy od narysowania 10 trajektorii standardowego procesu ruchu Browna dla $T = 1$ i $N = 16$. (Tabela 1 z podrozdziału 1.1).

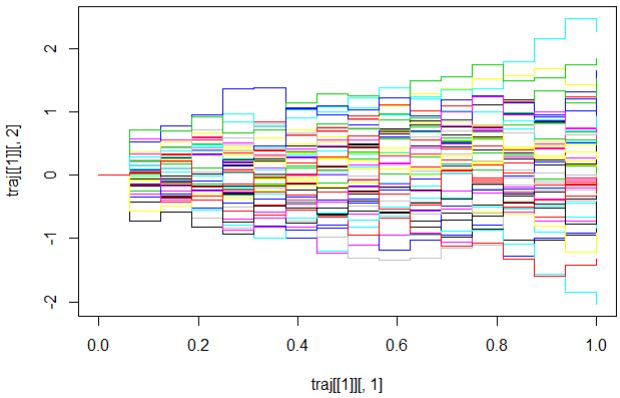
```
N<-16
T<-1
M<-10
wykresy_Brown(N,T,M)
```



Rysunek 3: Wygenerowane trajektorie.

Wykres jest dośc, czytelny, zobaczmy co stanie się gdy zwiększymy ilość trajektorii na wykresie do 50.

```
N<-16
T<-1
M<-50
wykresy_Brown(N,T,M)
```

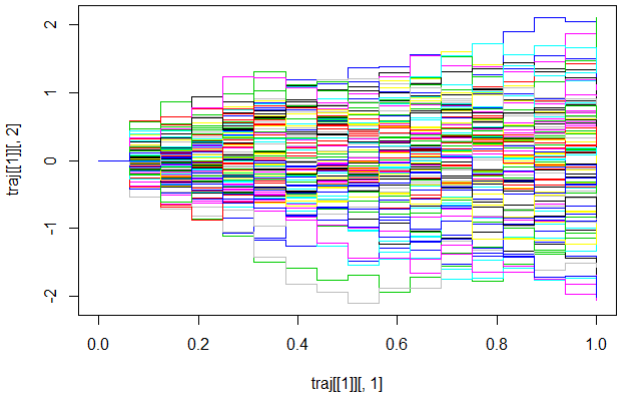


Rysunek 4: Wygenerowane trajektorie.

Zwiększenie ilości trajektorii sprawia, że wykres jest znacznie mniej czytelny. Za to widzimy, że przy wygenerowaniu 50 trajektorii wartości w punkcie $T = 1$ należą do przedziału około -2 do 2.

Sprawdźmy jeszcze jak wygląda wykres dla $M = 100$.

```
N<-16
T<-1
M<-100
wykresy_Brown(N,T,M)
```

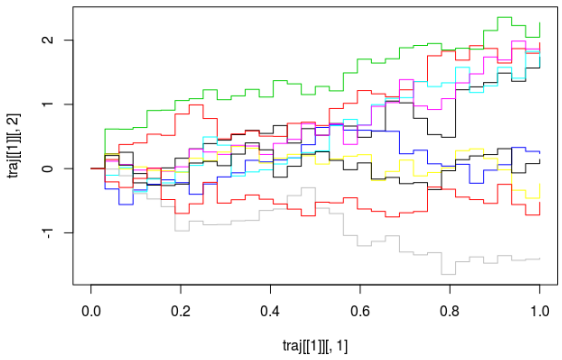


Rysunek 5: Wygenerowane trajektorie.

Z tego wykresu nie jesteśmy już w stanie odczytać poszczególnych trajektorii, jednak ponownie widzimy, że w punkcie $T = 1$ wartości wahają się od około -2 do 2.

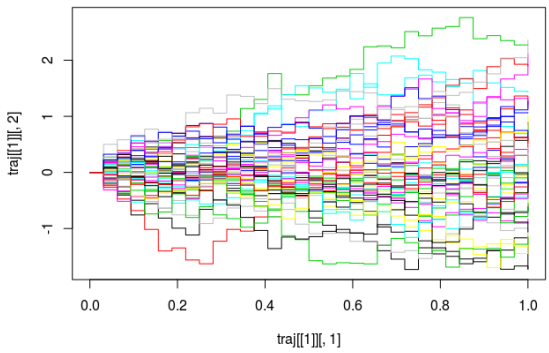
Teraz narysujsmy 10 trajektorii standardowego procesu ruchu Browna dla $T = 1$ i $N = 32$. (Tabela 2 z podrozdziału 1.1).

```
N<-32
T<-1
M<-10
wykresy_Brown(N,T,M)
```



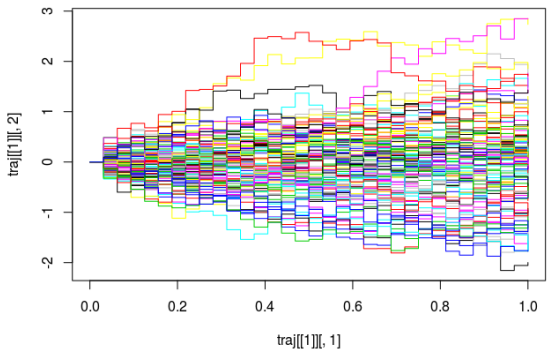
Rysunek 6: Wygenerowane trajektorie.

```
N<-32
T<-1
M<-50
wykresy_Brown(N,T,M)
```



Rysunek 7: Wygenerowane trajektorie.

```
N<-32
T<-1
M<-100
wykresy_Brown(N,T,M)
```



Rysunek 8: Wygenerowane trajektorie.

Dla $N = 32$ i $T = 1$ widzimy, że w przypadku 50 i 100 trajektorii mimo, że podział były gęstszy niż dla $N = 16$, nie zmienia się zakres wartości w punkcie $T = 1$ - nadal jest to około -2 do 2.

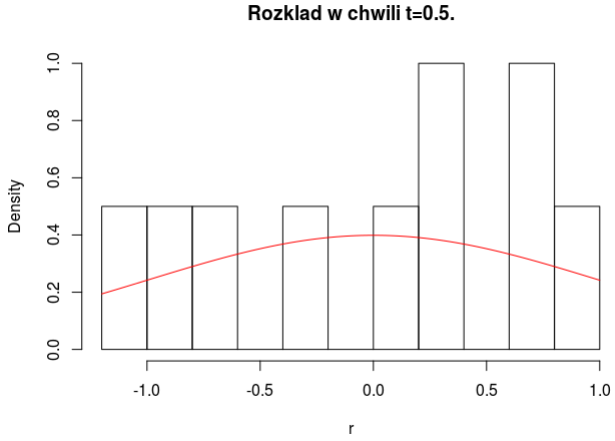
1.3 Rozkład w chwili t.

Zbadamy teraz jak wygląda rozkład procesu w wybranej chwili t. Posłuży nam do tego funkcja *rozklad*. Jej argumentami są N, T- argumenty funkcji *pr_r_B*, M- ilosć generowanych procesów oraz t- moment w którym chcemy sprawdzić rozkład. Ważne jest, aby argument t był mniejszy od T.

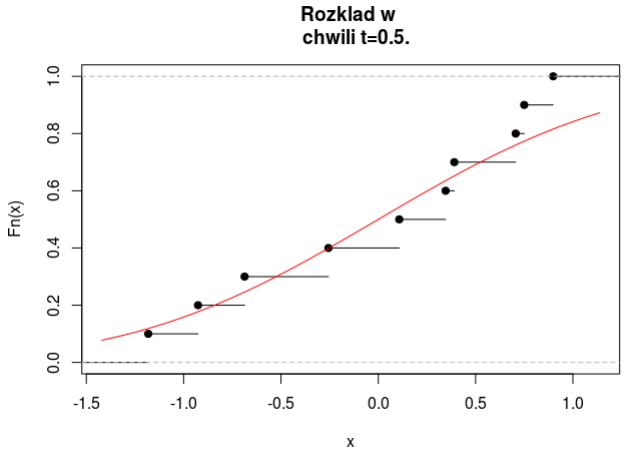
```
rozklad<-function(t,N,T,M){
r<-c()
traj<-list()
for( i in 1:M){
traj[[i]]<-pr_r_B(N,T)
r[i]<-traj[[i]][sum(traj[[i]][,1]<=t),2]
}
hist(r,10, prob=TRUE, main =
paste("Rozklad w chwili t=" , t, ".", sep=""))
curve(dnorm(x,0,sqrt(t)),add=TRUE, col=2)
plot(ecdf(r), main = paste("Rozklad w
chwili t=" , t, ".", sep=""))
curve(pnorm(x,0,sqrt(t)),add=TRUE, col=2)

}
```

Zaczniemy od sprawdzenia jak wygląda rozkład w chwili $t = 0.5$. Na początek weźmy $M = 10$.

```
N<-16
T<-1
M<-10
t<-0.5
rozklad(t,N,T,M)
```



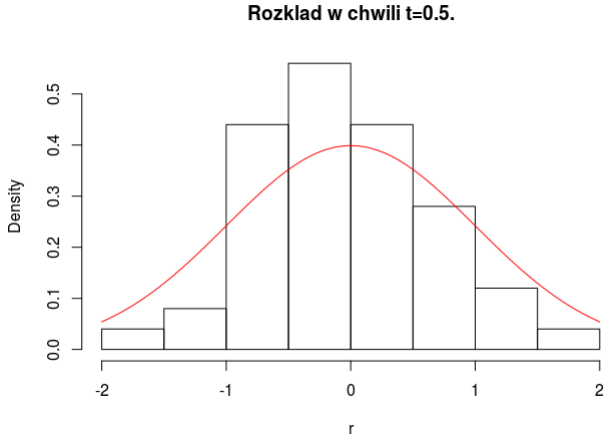
Rysunek 9: Rozkład w chwili t.



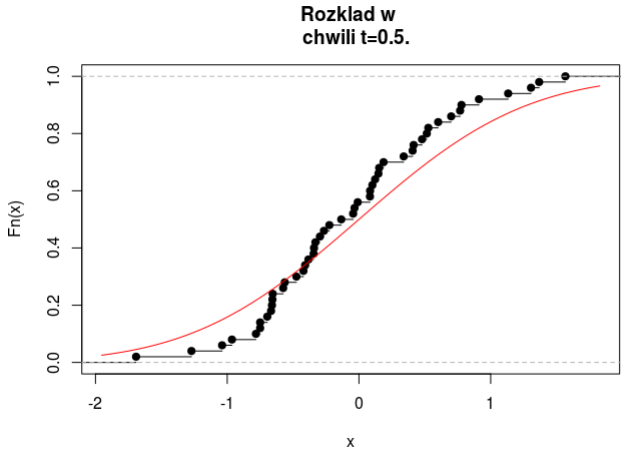
Rysunek 10: Dystrybuanta rozkładu.

Po histogramie widać, że 10 trajektorii to jest zdecydowanie za mało, dlatego sprawdźmy rozkład dla $M = 50$.

```
N<-16
T<-1
M<-50
t<-0.5
rozklad(t,N,T,M)
```



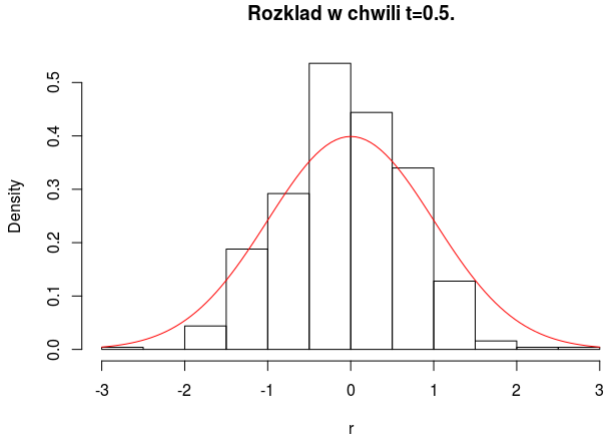
Rysunek 11: Rozkład w chwili t.



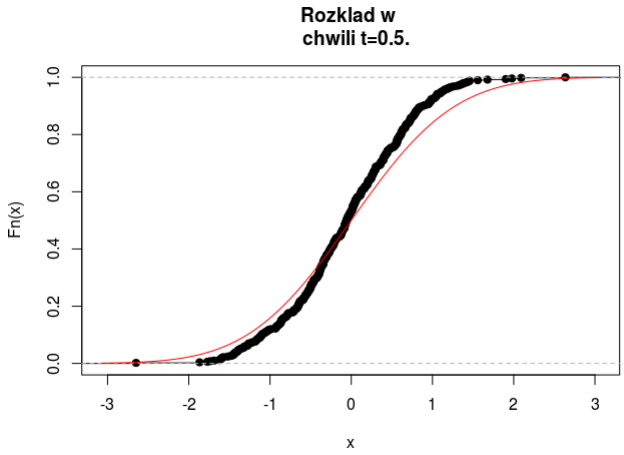
Rysunek 12: Dystrybuanta rozkładu.

Widzimy, że dla $M = 50$ histogram i dystrybuanta zaczynaja przypominać rozkład normalny. Dlatego narysujmy jeszcze ten sam proces dla $M = 500$.

```
N<-16
T<-10
M<-500
t<-5
rozklad(t,N,T,M)
```



Rysunek 13: Rozkład w chwili t.

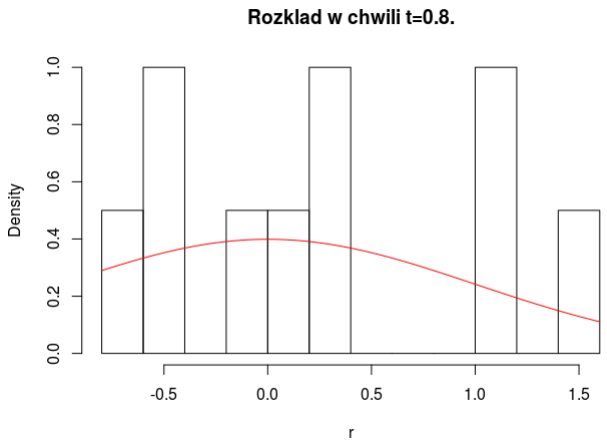


Rysunek 14: Dystrybuanta rozkładu.

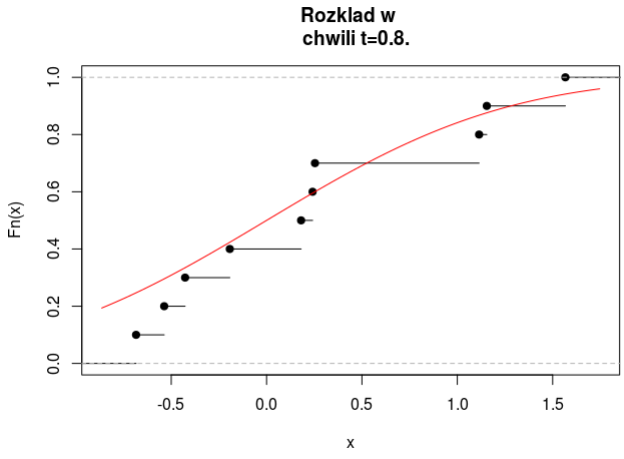
Widzimy więc, że rozkład w chwili $t = 0.5$ pasuje do krzywej standardowego rozkładu normalnego. Rozkład w chwili $t = 0.5$ jest więc standardowym rozkładem normalnym.

Zbadamy dodatkowo rozkład dla $t = 0.8$. Ponownie zróbmy to dla $M = 10, 50, 500$.

```
N<-16
T<-1
M<-10
t<-0.8
rozklad(t,N,T,M)
```

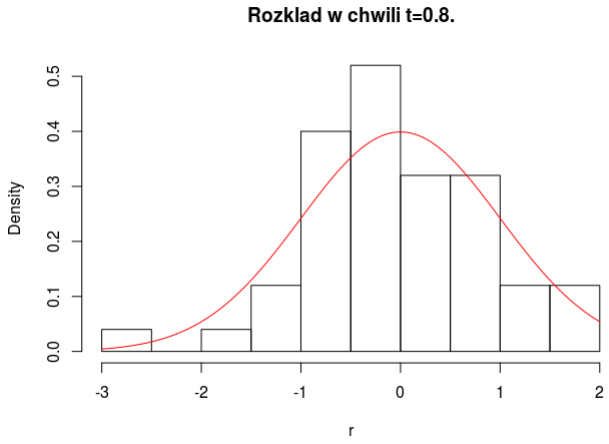


Rysunek 15: Rozkład w chwili t.

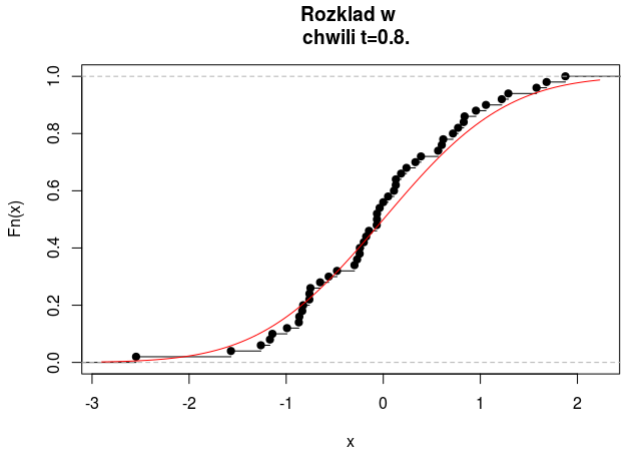


Rysunek 16: Dystrybuanta rozkładu.

```
N<-16
T<-1
M<-50
t<-0.8
rozklad(t,N,T,M)
```

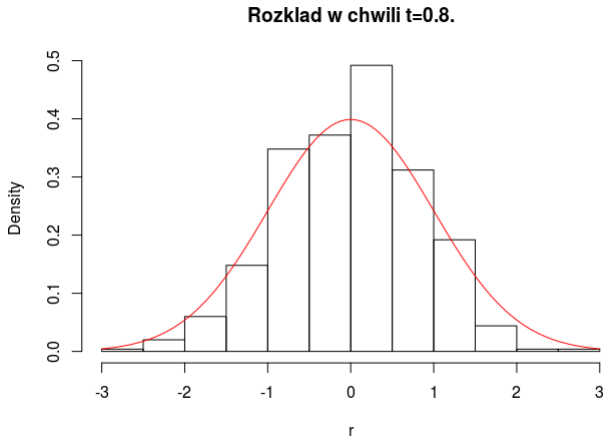


Rysunek 17: Rozkład w chwili t.

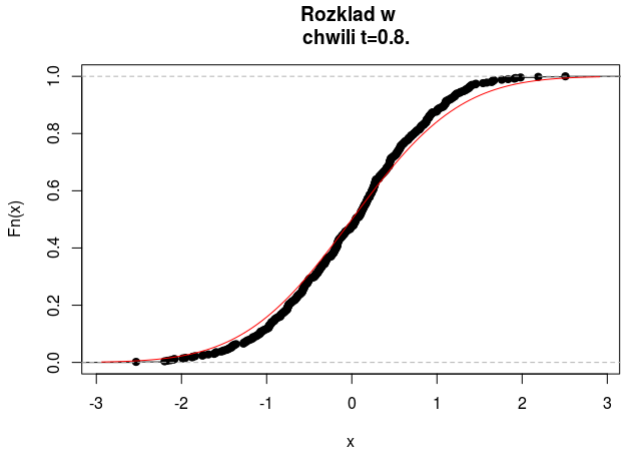


Rysunek 18: Dystrybuanta rozkładu.

```
N<-16
T<-1
M<-500
t<-0.8
rozklad(t,N,T,M)
```



Rysunek 19: Rozkład w chwili t.



Rysunek 20: Dystrybuanta rozkładu.

Z 3 histogramów i wykresów dystrybuant widzimy, że w $t = 0.8$ rozkład również pasuje do teoretycznego rozkładu standardowego normalnego.

1.4 Funkcja kowariancji

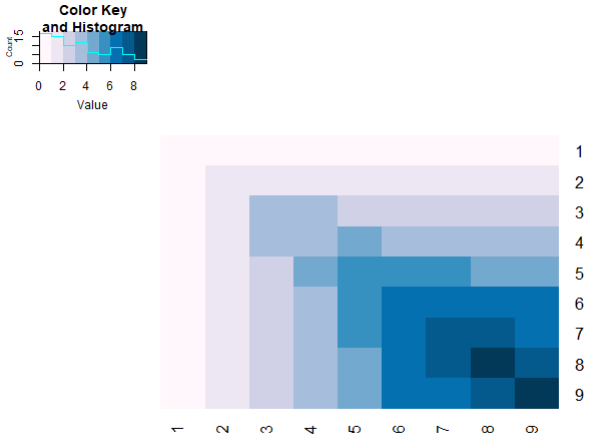
W tym podrodziale wyznaczamy funkcję liczącą kowariancję między poszczególnymi momentami procesu ruchu Browna. Funkcja *covariance* zwraca macierz kowariancji oraz mapę ciepła tej kowariancji po podaniu argumentów N, T - argumenty funkcji pr_r_B oraz M- ilość generowanych procesów.

```
covariance  <- function(M, T, N ){

  kowariancja<-data.frame(matrix(data=NA, nrow=N+1, ncol=N+1))

  traj<-list()

  for (i in 1:M){

    traj[[i]] <- pr_r_B(N,T)

  }

  wektor_czasow<-traj[[1]][,1]

  for( i in 1:length(wektor_czasow)){

    x<-c()

    for( j in 1:M){

      x[j]<-traj[[j]][which(traj[[j]][,1]==wektor_czasow[i]),2]
   }

    for(j in 1:length(wektor_czasow)) {
      y<-c()

      for( k in 1:M){

        y[k]<-traj[[k]][which(traj[[k]][,1]==wektor_czasow[j]),2]
    }
      kowariancja[i,j]<-cov(x,y)

    }
  }

  rownames(kowariancja)<-1:length(wektor_czasow)
  colnames(kowariancja)<-1:length(wektor_czasow)


  return(kowariancja)
}
```

```
N<-8
T<-10
M<-100
k<-covariance(M,T,N)
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```

	1	2	3	4	5	6	7	8	9
1	0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
2	0	1.226886	1.577291	1.499301	1.391267	1.222232	1.089792	1.073297	1.037807
3	0	1.577291	3.342541	3.028792	2.980824	2.783167	2.672248	2.466197	2.304347
4	0	1.499301	3.028792	3.875867	4.038909	4.015195	3.983711	3.790730	3.651727
5	0	1.391267	2.980824	4.038909	5.483307	5.282319	5.250471	4.966464	4.818048
6	0	1.222232	2.783167	4.015195	5.282319	6.456815	6.526236	6.221473	6.067153
7	0	1.089792	2.672248	3.983711	5.250471	6.526236	7.701899	7.296062	6.988965
8	0	1.073297	2.466197	3.790730	4.966464	6.221473	7.296062	8.114810	7.829833
9	0	1.037807	2.304347	3.651727	4.818048	6.067153	6.988965	7.829833	9.056837

Rysunek 21: Macierz kowariancji.

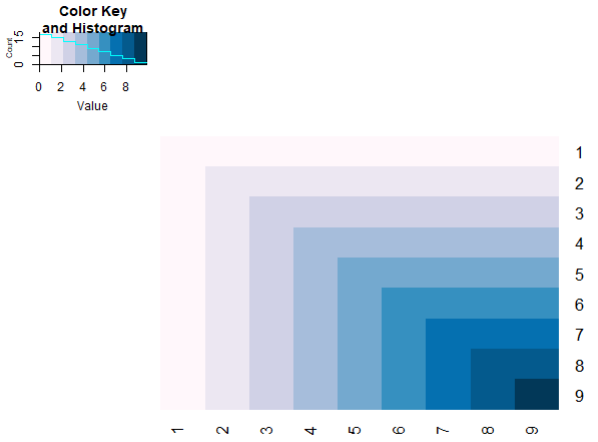


Rysunek 22: Mapa ciepła kowariancji.

```
N<-8
T<-10
M<-1000
k<-covariance(M,T,N)
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```

	1	2	3	4	5	6	7	8	9
1	0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
2	0	1.266528	1.235787	1.268962	1.302319	1.260189	1.248890	1.297589	1.292497
3	0	1.235787	2.456285	2.431875	2.479300	2.418788	2.455679	2.495619	2.525511
4	0	1.268962	2.431875	3.611120	3.668811	3.649608	3.684809	3.732406	3.733986
5	0	1.302319	2.479300	3.668811	5.041042	5.023197	5.113142	5.090387	5.087935
6	0	1.260189	2.418788	3.649608	5.023197	6.222891	6.223500	6.178889	6.128357
7	0	1.248890	2.455679	3.684809	5.113142	6.223500	7.491564	7.528083	7.482449
8	0	1.297589	2.495619	3.732406	5.090387	6.178889	7.528083	8.722320	8.673499
9	0	1.292497	2.525511	3.733986	5.087935	6.128357	7.482449	8.673499	9.826457

Rysunek 23: Macierz kowariancji.



Rysunek 24: Mapa ciepła kowariancji.

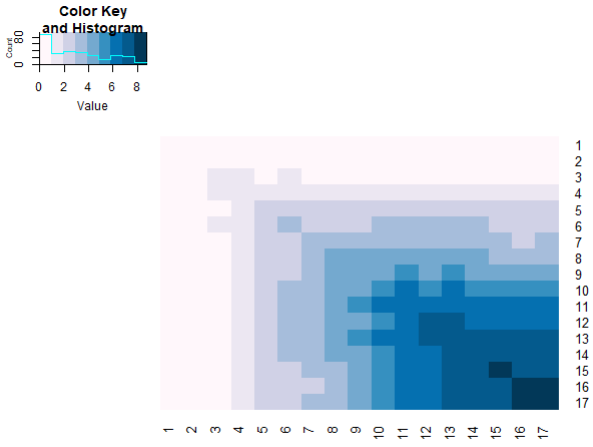
Widzimy wygenerowane dwie macierze kowariancji dla $N = 8, T = 10$, które różnią się ilością generowanych procesów M. W obu przypadkach widzimy, że kowariancja jakiegokolwiek elementu z 0 wynosi 0. Widać to także na mapie ciepła - jasny kolor w górnym oraz lewym brzegu odpowiada za wartości bliskie 0. Widzimy też na obu mapach, że im bliżej dolnego prawego rogu, tym kolor na mapie jest ciemniejszy. Z macierzy możemy odczytać, że najwyższa wartość kowariancji jest bardzo bliska $T = 10$ i występuje ona właśnie w dolnym prawym rogu. Dodatkowo widzimy, że przy $M = 1000$ przejście kolorów na mapie ciepła jest łagodniejsze niż przy $M = 100$.

Teraz sprawdźmy jak wygląda kowariancja gdy przyjmiemy $N = 16$. Niech T ponownie będzie równe 10.

```
N<-16
T<-10
M<-100
k<-covariance(M,T,N)
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0	0.66	0.73	0.68	0.62	0.55	0.52	0.59	0.71	0.75	0.66	0.59	0.62	0.61	0.57	0.55	0.50
3	0	0.73	1.47	1.36	1.15	1.12	1.08	1.15	1.34	1.36	1.23	1.07	1.24	1.27	1.30	1.31	1.32
4	0	0.68	1.36	1.82	1.46	1.45	1.42	1.46	1.74	1.62	1.58	1.49	1.57	1.55	1.49	1.42	1.36
5	0	0.62	1.15	1.46	1.73	1.73	1.68	1.81	1.96	1.84	1.79	1.80	1.87	1.88	1.77	1.79	1.69
6	0	0.55	1.12	1.45	1.73	2.21	2.10	2.24	2.33	2.20	2.12	2.12	2.22	2.32	2.17	2.23	2.19
7	0	0.52	1.08	1.42	1.68	2.10	2.52	2.60	2.74	2.63	2.59	2.56	2.70	2.76	2.61	2.63	2.58
8	0	0.59	1.15	1.46	1.81	2.24	2.60	3.24	3.39	3.34	3.25	3.37	3.48	3.53	3.36	3.40	3.26
9	0	0.71	1.34	1.74	1.96	2.33	2.74	3.39	4.04	3.98	3.90	4.08	4.23	4.31	4.15	4.18	4.01
10	0	0.75	1.36	1.62	1.84	2.20	2.63	3.34	3.98	4.58	4.40	4.66	4.81	4.87	4.71	4.80	4.65
11	0	0.66	1.23	1.58	1.79	2.12	2.59	3.25	3.90	4.40	4.78	5.06	5.27	5.22	5.03	5.13	4.99
12	0	0.59	1.07	1.49	1.80	2.12	2.56	3.37	4.08	4.66	5.06	5.95	6.12	6.09	5.85	6.01	5.76
13	0	0.62	1.24	1.57	1.87	2.22	2.70	3.48	4.23	4.81	5.27	6.12	6.95	7.03	6.80	6.99	6.84
14	0	0.61	1.27	1.55	1.88	2.32	2.76	3.53	4.31	4.87	5.22	6.09	7.03	7.71	7.45	7.66	7.61
15	0	0.57	1.30	1.49	1.77	2.17	2.61	3.36	4.15	4.71	5.03	5.85	6.80	7.45	7.94	8.11	8.07
16	0	0.55	1.31	1.42	1.79	2.23	2.63	3.40	4.18	4.80	5.13	6.01	6.99	7.66	8.11	8.96	8.96
17	0	0.50	1.32	1.36	1.69	2.19	2.58	3.26	4.01	4.65	4.99	5.76	6.84	7.61	8.07	8.96	9.53

Rysunek 25: Macierz kowariancji.

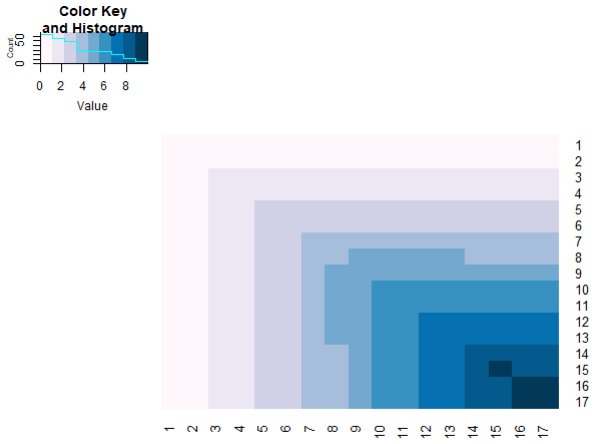


Rysunek 26: Mapa ciepła kowariancji.

```
N<-16
T<-10
M<-1000
k<-covariance(M,T,N)
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
trace="none",col=brewer.pal(9,"PuBu") )
```

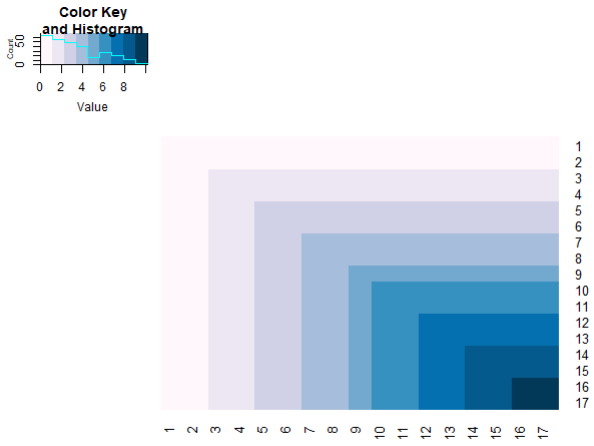
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0	0.62	0.62	0.62	0.64	0.63	0.60	0.59	0.61	0.63	0.60	0.59	0.57	0.58	0.59	0.63	0.60
3	0	0.62	1.31	1.30	1.32	1.33	1.27	1.27	1.32	1.35	1.30	1.28	1.25	1.26	1.26	1.26	1.21
4	0	0.62	1.30	1.93	1.93	1.95	1.87	1.90	1.98	2.01	1.95	1.92	1.90	1.91	1.90	1.91	1.87
5	0	0.64	1.32	1.93	2.52	2.52	2.47	2.50	2.59	2.64	2.60	2.57	2.56	2.58	2.56	2.54	2.46
6	0	0.63	1.33	1.95	2.52	3.17	3.12	3.17	3.28	3.29	3.29	3.24	3.22	3.23	3.22	3.18	3.11
7	0	0.60	1.27	1.87	2.47	3.12	3.74	3.77	3.87	3.84	3.87	3.83	3.84	3.81	3.81	3.78	3.69
8	0	0.59	1.27	1.90	2.50	3.17	3.77	4.42	4.52	4.50	4.53	4.49	4.50	4.38	4.38	4.35	4.30
9	0	0.61	1.32	1.98	2.59	3.28	3.87	4.52	5.26	5.22	5.24	5.23	5.25	5.14	5.14	5.10	5.06
10	0	0.63	1.35	2.01	2.64	3.29	3.84	4.50	5.22	5.76	5.79	5.73	5.70	5.59	5.62	5.58	5.59
11	0	0.60	1.30	1.95	2.60	3.29	3.87	4.53	5.24	5.79	6.48	6.49	6.49	6.37	6.44	6.38	6.42
12	0	0.59	1.28	1.92	2.57	3.24	3.83	4.49	5.23	5.73	6.49	7.11	7.11	6.99	7.06	7.02	7.03
13	0	0.57	1.25	1.90	2.56	3.22	3.84	4.50	5.25	5.70	6.49	7.11	7.73	7.63	7.69	7.64	7.63
14	0	0.58	1.26	1.91	2.58	3.23	3.81	4.38	5.14	5.59	6.37	6.99	7.63	8.16	8.24	8.17	8.11
15	0	0.59	1.26	1.90	2.56	3.22	3.81	4.38	5.14	5.62	6.44	7.06	7.69	8.24	8.89	8.83	8.80
16	0	0.63	1.26	1.91	2.54	3.18	3.78	4.35	5.10	5.58	6.38	7.02	7.64	8.17	8.83	9.39	9.38
17	0	0.60	1.21	1.87	2.46	3.11	3.69	4.30	5.06	5.59	6.42	7.03	7.63	8.11	8.80	9.38	9.99

Rysunek 27: Macierz kowariancji.



Rysunek 28: Mapa ciepła kowariancji.

```
N<-16
T<-10
M<-5000
k<-covariance(M,T,N)
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```

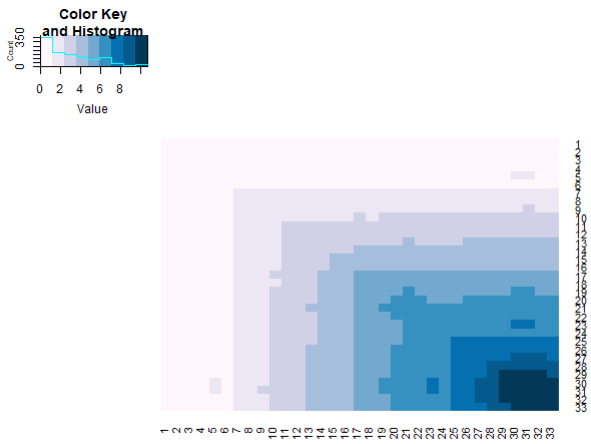


Rysunek 29: Mapa ciepła kowariancji.

Dla $N = 16$ wygenerowaliśmy 3 kowariancje. Wnioski są analogiczne do wykresów dla $N = 8$. Widzimy jednak, że dopiero dla $M = 5000$ kolory ciemnieją regularnie.

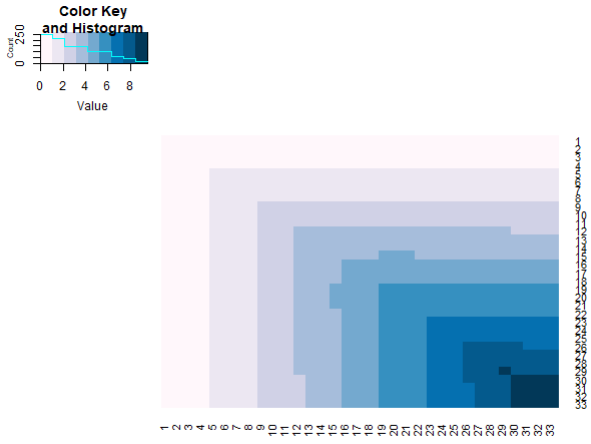
Na koniec zobaczmy jeszcze jak wygląda mapa ciepła kowariancji dla $N = 32$.

```
N<-32
T<-10
M<-100
k<-covariance(M,T,N)
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```



Rysunek 30: Mapa ciepła kowariancji.

```
N<-32
T<-10
M<-1000
k<-covariance(M,T,N)
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
trace="none",col=brewer.pal(9,"PuBu") )
```



Rysunek 31: Mapa ciepła kowariancji.

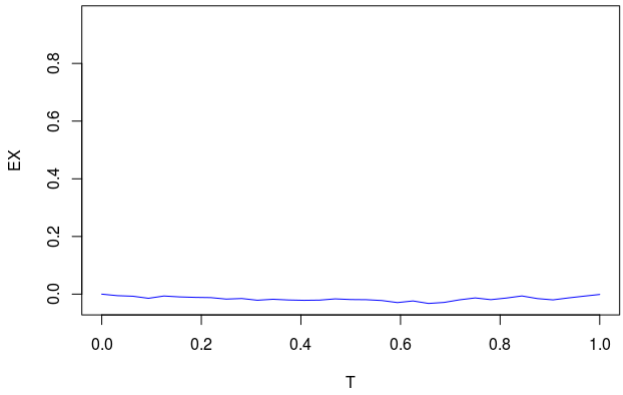
Widzimy, że mapa ciepła dla różnych N wygląda analogicznie.

1.5 Funkcja wartości oczekiwanej i wariancji

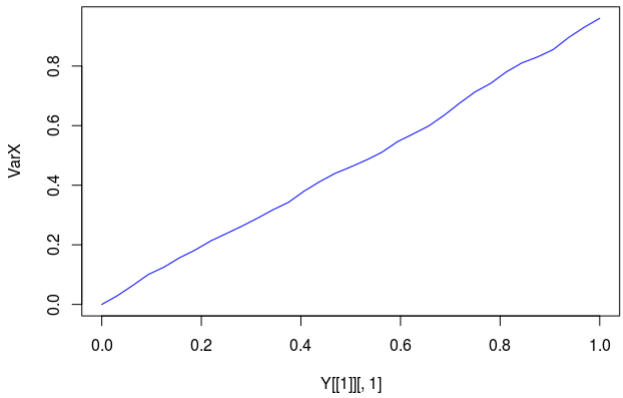
```
EXVar<-function(M, f, arg){

Y<-list()
Y2<-list()

for( i in 1:M){

Y[[i]]<-do.call(f,arg)
Y2[[i]]<-cbind(Y[[i]][,1], (Y[[i]][,2])^2)

}
suma<-do.call('rbind', Y)
suma<-aggregate(suma[,2], by=list(suma[,1]), FUN=sum)

suma2<-do.call('rbind', Y2)
suma2<-aggregate(suma2[,2], by=list(suma2[,1]), FUN=sum)

EX=suma[,2]/M
VarX=suma2[,2]/M-(EX)^2

plot(Y[[1]][,1], EX, type='l', xlab="T",
ylim =c(min(min(EX), min(VarX)),
max(max(EX), max(VarX))))
plot(Y[[1]][,1], VarX, col=2, type='l')
}
```

Funkcja *EXVar* jest uniwersalną funkcją, która zwraca nam dwa wykresy- wykres wartości oczekiwanej i wariancji. W jej argumentach musimy podać liczbę iteracji, które chcemy wykonać, nazwę funkcji, w naszym wypadku procesu, który chcemy zbadać, oraz listę argumentów podanej funkcji.

```
M<-1000
N<-32
T<-1
f<-"pr_r_B"
arg<-list(N=N, T=T)
EXVar(M,f,arg)
```



Rysunek 32: Wartość oczekiwana.



Rysunek 33: Wariancja.

Teoretyczna wartość oczekiwana standardowego procesu ruchu Browa wynosi 0 - zgadza się to z wykresem. Teoretyczna wariancja natomiast wynosi t, to również potwierdzone jest przez wykres.

2 Geometryczny ruch Browna

2.1 Proces ruchu

Przechodzimy teraz do geometrycznego ruchu Browna. Użyjemy do wygenerowania tego procesu napisanej wcześniej funkcji *pr_r_B*.

```
pr_r_B<-function(N,T){


  delta_t<-T/N

  wektor_tj<-seq(0,T, by=delta_t)

  B<-c(0)

  ksi<-rnorm(N,0,1)

  for( i in 2:(N+1)){

    B[i]<-B[i-1]+sqrt(delta_t)*ksi[i-1]

  }

  return(cbind(wektor_tj, B))



}
geom_r_B<-function(N,T, x0, mu, s){

  S<-c(x0)

  B<-pr_r_B(N,T)

  t<-B[,1]

  for( i in 2:nrow(B)){
    S[i]<-x0*exp((mu-0.5*s^2)*t[i]+s*B[i,2])
  }

  return(cbind(t,S))

}
```

W funkcji *geom_r_B* pojawiają się dodatkowe argumenty. x_0- stan początkowy oraz parametry μ i σ.

```
N<-16
T<-10
x0<-1
mu<-0
s<-1
geom_r_B(N,T, x0, mu, s)
```

	t	S
1	0.000	1.000000000
2	0.625	1.281647977
3	1.250	0.953908144
4	1.875	0.219387250
5	2.500	0.219626947
6	3.125	0.196871626
7	3.750	0.501405154
8	4.375	0.379661115
9	5.000	0.455042217
10	5.625	0.333331428
11	6.250	0.191787261
12	6.875	0.084249150
13	7.500	0.100049366
14	8.125	0.016211808
15	8.750	0.047514628
16	9.375	0.002422147
17	10.000	0.004775288

Rysunek 34: Wygenerowany proces.

Widzimy, że dzięki parametrowi $x0 = 1$ w chwili $t = 0$ wartość procesu równa się 1.

Zobaczmy jeszcze jak wygląda proces gdy $N = 32$.

```
N<-32
T<-10
x0<-1
mu<-0
s<-1
geom_r_B(N,T, x0, mu, s)
```

	t	S
1	0.0000	1.00000000
2	0.3125	0.49928039
3	0.6250	0.26684298
4	0.9375	0.24805189
5	1.2500	0.12686712
6	1.5625	0.09809530
7	1.8750	0.03174584
8	2.1875	0.02450451
9	2.5000	0.02991111
10	2.8125	0.03925983
11	3.1250	0.03003414
12	3.4375	0.02679024
13	3.7500	0.07265649
14	4.0625	0.05686012
15	4.3750	0.08036941
16	4.6875	0.07439739
17	5.0000	0.15171989
18	5.3125	0.05328957
19	5.6250	0.12263646
20	5.9375	0.11784629
21	6.2500	0.11418424
22	6.5625	0.15518176
23	6.8750	0.21011900
24	7.1875	0.12211447
25	7.5000	0.10898542
26	7.8125	0.25277876
27	8.1250	0.06725416
28	8.4375	0.05338144
29	8.7500	0.02686200
30	9.0625	0.02258275
31	9.3750	0.01808070
32	9.6875	0.02528479
33	10.0000	0.01296485

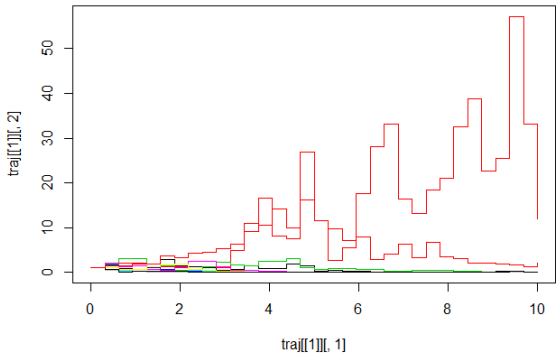
Rysunek 35: Wygenerowany proces.

Liczba okresów wynosi teraz 33, ponownie widzimy, że w chwili $t = 0$ wartość procesu wynosi 1 poprzez ustawienie $x0 = 1$.

2.2 Trajektorie

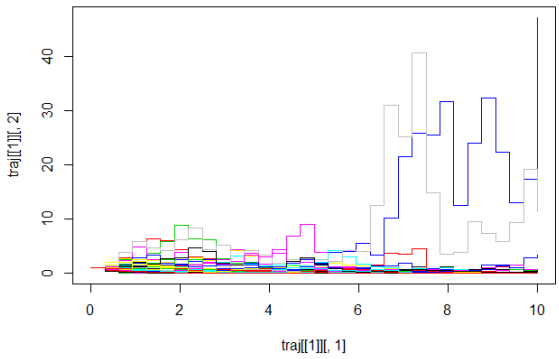
Funkcja *wykresy_geom* rysuje trajektorie geometrycznego ruchu Browna. Pojawia sę dodatkowy parametr M odpowiedzialny za ilość generowanych procesów.

```
wykresy_geom<-function(N,T, x0, mu, s,M){

  n<-c()

  traj <-list()

  ymin<-c()
  ymax<-c()

  for( i in 1:M){

    traj[[i]]<-geom_r_B(N,T, x0, mu, s)
    n[i]<-nrow(traj[[i]])
    ymin[i]<-min(traj[[i]][,2])
    ymax[i]<-max(traj[[i]][,2])

  }

  plot(traj[[1]][,1],traj[[1]][,2],type='s',  ylim=c(min(ymin), max(ymax)))

  for( i in 2:M){

    lines(traj[[i]][,1],traj[[i]][,2],type='s',col=i)

  }

}

N<-32
T<-10
x0<-1
mu<-0
s<-1
M<-10
wykresy_geom(N,T,x0,mu,s,M)
```



Rysunek 36: Wygenerowane trajektorie.

```
N<-32
T<-10
x0<-1
mu<-0
s<-1
M<-50
wykresy_geom(N,T,x0,mu,s,M)
```

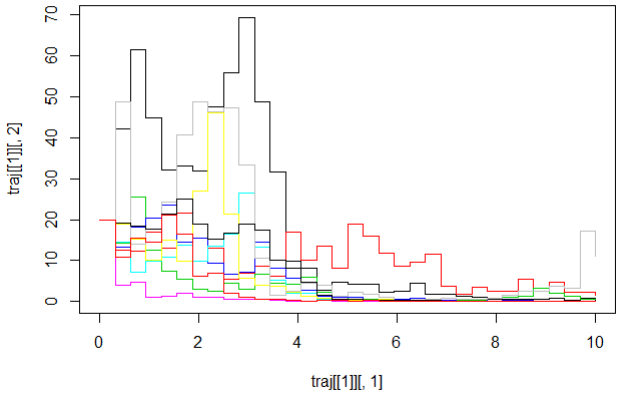


Rysunek 37: Wygenerowane trajektorie.

Dwa powyższe wykresy przedstawiają geometryczny proces ruchu Browna dla $T = 10, N = 32, \mu = 0, \sigma = 1$, zaczynający się w punkcie $x_0 = 1$. Pierwszy wykres to $M = 10$ trajektorii, drugi to $M = 50$. Wykresy są bardzo podobne.

Zobaczmy jak zmiana x_0 wpłynie na wykres trajektorii.

```
N<-32
T<-10
x0<-20
mu<-0
s<-1
M<-10
wykresy_geom(N,T,x0,mu,s,M)
```

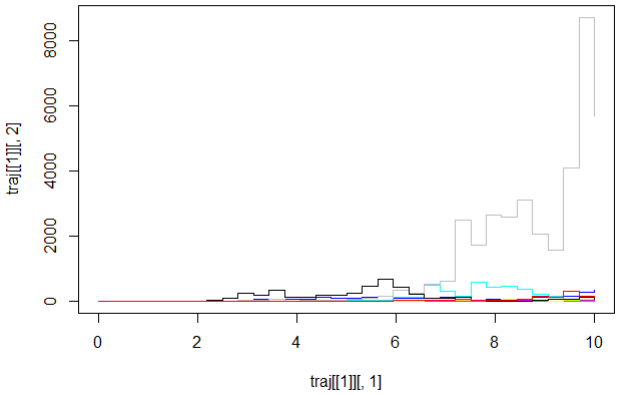


Rysunek 38: Wygenerowane trajektorie.

Na wykresie widać, że początkowe części trajektorii znajdują się "wyżej", co spowodowane jest ich rozpoczęciem w wartości 20.

Zobaczmy jeszcze jak na trajektorie wpływa zmiana parametru μ. Parametr x_0 z powrotem ustalmy na 1.

```
N<-32
T<-10
x0<-1
mu<-0.8
s<-1
M<-10
wykresy_geom(N,T,x0,mu,s,M)
```

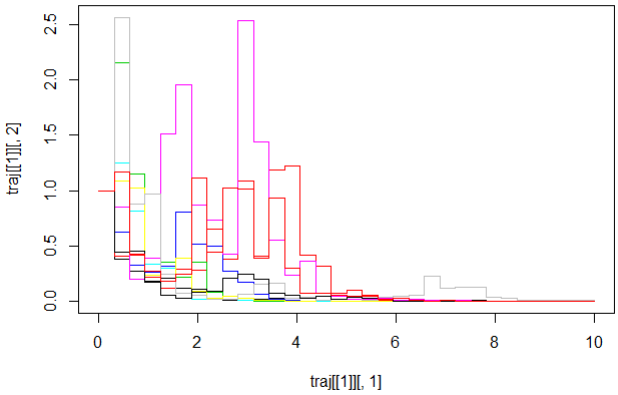


Rysunek 39: Wygenerowane trajektorie.

Okazuje się, że zmiana μ z 0 na 0.8 sprawia, że trajektorie są bardziej płaskie, większośc z nich ma na całym odcinku $0-10$ wartości nie przekraczające około 500.

Sprawdzimy jeszcze jak zachowują się procesy po zmianie parametru σ z 1 na 1.5. Parametr μ z powrotem ustawamy na 0.

```
N<-32
T<-10
x0<-1
mu<-0
s<-1.5
M<-10
wykresy_geom(N,T,x0,mu,s,M)
```



Rysunek 40: Wygenerowane trajektorie.

Okazuje się, że zwiększenie parametru σ do 1.5 wpłynęło na wyższe wartości na początku procesu i widoczny spadek praktycznie do 0 od około $t = 5$.

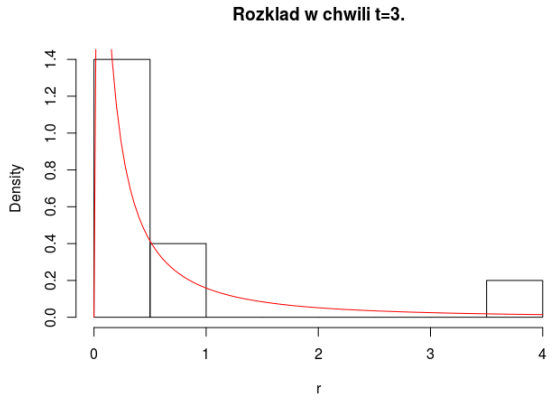
2.3 Rozkład w chwili t.

Teraz zbadamy jak wygląda rozkład geometrycznego procesu ruchu Browna. Analogicznie do standardowego procesu ruchu Browna, mamy funkcję *rozklad_geom*. Jej argumentami są argumenty funkcji *geom_r_B* oraz dodatkowo M- ilość generowanych trajektorii.

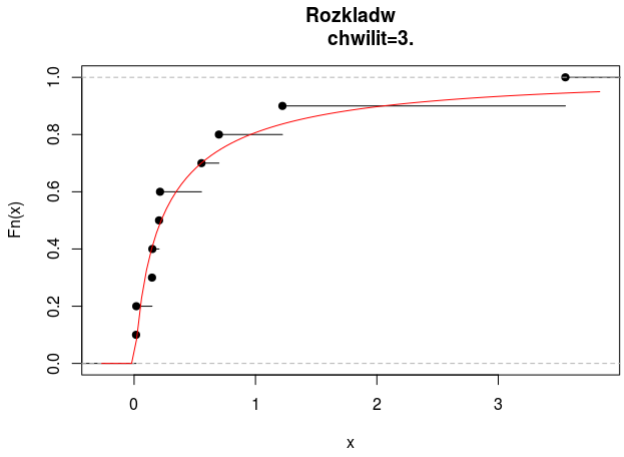
```
rozklad_geom<-function(t,N,T,x0, mu, s,M){
r<-c()
        traj<-list()
        for( i in 1:M){
                traj[[i]]<-geom_r_B(N,T,x0,mu,s)
                r[i]<-traj[[i]][sum(traj[[i]][,1]<=t),2]
        }
        hist(r,10, prob=TRUE, main =
        paste("Rozklad␣w␣chwili␣t=" , t, ".", sep=""))
        curve(dlnorm(x,log(x0)+mu*t-0.5*(s^2)*t,s*sqrt(t)), add=TRUE, col=2)
        plot(ecdf(r), main = paste("Rozkladw
␣␣␣␣␣␣␣␣chwilit=" , t, ".", sep=""))
        curve(plnorm(x,log(x0)+mu*t-0.5*(s^2)*t,s*sqrt(t)), add=TRUE, col=2)
}
```

Zacznijmy od zbadania rozkładu w punkcie $t = 3$, przy $N = 32, T = 10$. Wygenerujemy najpierw $M = 10$ procesów.

```
N<-32
T<-10
x0<-1
mu<-0
s<-1
M<-10
t<-3
rozklad_geom(t,N,T,x0, mu, s,M)
```



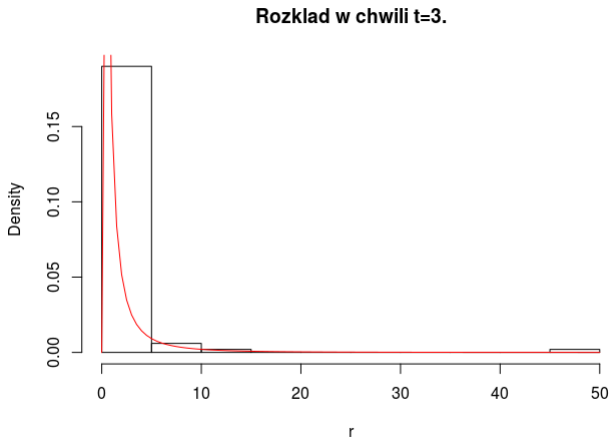
Rysunek 41: Rozkład w chwilii t.



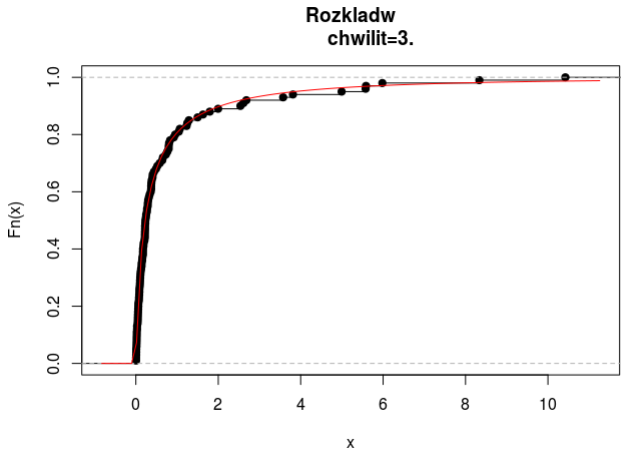
Rysunek 42: Dystrybuanta rozkładu.

Dla $M = 10$ wykresy nie są dobrze, dopasowane do wartości teoretycznych. Spróbujmy podwyższyć M.

```
N<-32
T<-10
x0<-1
mu<-0
s<-1
M<-100
t<-3
rozklad_geom(t,N,T,x0, mu, s,M)
```



Rysunek 43: Rozkład w chwilii t.



Rysunek 44: Dystrybuanta rozkładu.

Teraz histogram i wykres dystrybuanty pasuje do krzywej teoretycznej, która pochodzi z rozkładu lognormalnego.

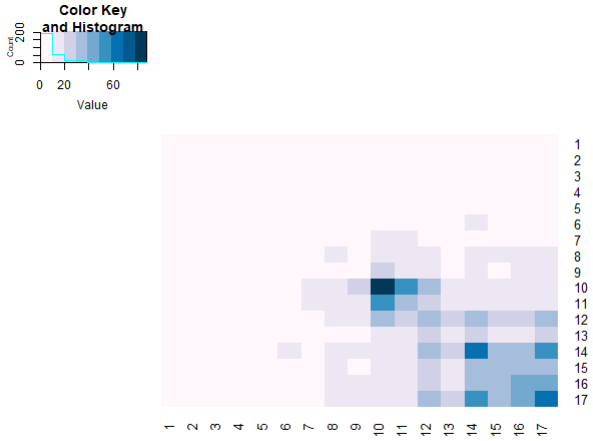
2.4 Funkcja kowariancji

Funkcja *covariance_geom* liczy kowariancję dla geometrycznego procesu ruchu Browna, analogicznie jak w standardowym procesie.

```
covariance_geom  <- function(M, T, N ,x0,mu,s){

  kowariancja<-data.frame(matrix(data=NA, nrow=N+1, ncol=N+1))
  traj<-list()

  for (i in 1:M){

    traj[[i]] <- geom_r_B(N,T,x0,mu,s)

  }
  wektor_czasow<-traj[[1]][,1]

  for( i in 1:length(wektor_czasow)){

    x<-c()

    for( j in 1:M){

      x[j]<-traj[[j]][which(traj[[j]][,1]==wektor_czasow[i]),2]


    }

    for(j in 1:length(wektor_czasow)) {

      y<-c()

      for( k in 1:M){

        y[k]<-traj[[k]][which(traj[[k]][,1]==wektor_czasow[j]),2]

      }
      kowariancja[i,j]<-cov(x,y)

    }

  }

  rownames(kowariancja)<-1:length(wektor_czasow)
  colnames(kowariancja)<-1:length(wektor_czasow)

  return(kowariancja)

}
```

Aby zobaczyć jak wygląda kowariancja dla geometrycznego procesu, wywołamy ją z różnymi argumentami. Na początek $x_0 = 1, \mu = 0, \sigma = 1$.

```
M<-1000
T<-10
N<-16
x0<-1
mu<-0
s<-1

k<-covariance_geom(M,T,N,x0,mu,s)

heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
trace="none",col=brewer.pal(9,"PuBu") )
```

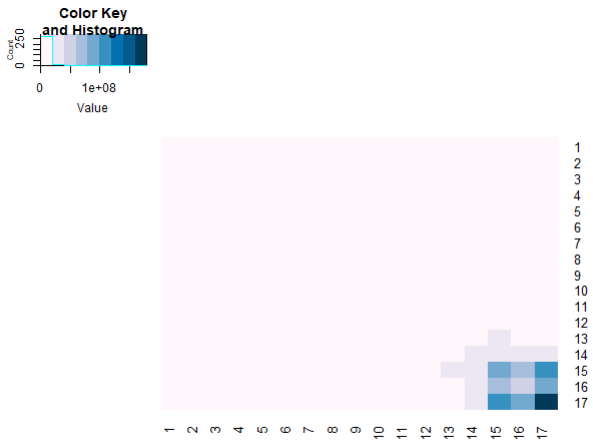


Rysunek 45: Mapa ciepła kowariancji.

Mapa ciepła kowariancji znacznie różni się od tej dla standardowego procesu ruchu Browna. Widzimy stosunkowo niewiele wartości wyższych, większość mapy jest bardzo jasna, co oznacza w większości bardzo niskie wartości kowariancji. Zobaczmy jak zmiana μ na 0.8 wpłynie na wykres.

```
M<-1000
T<-10
N<-16
x0<-1
mu<-0.8
s<-1

k<-covariance_geom(M,T,N,x0,mu,s)
```

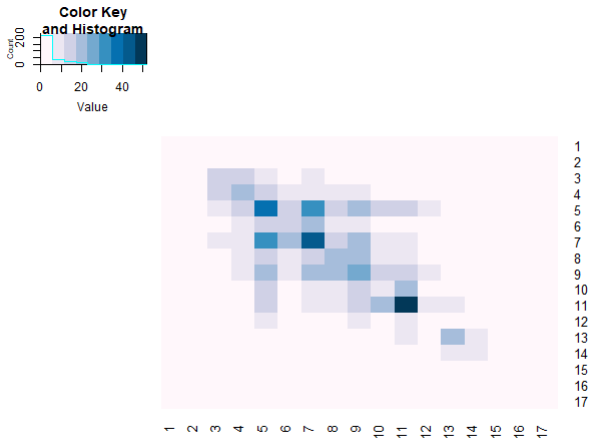
```
heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```



Rysunek 46: Mapa ciepła kowariancji.

Ilość przyciemnionych części mapy zmalała, teraz jedynie część bliska prawego dolnego rogu ma większą kowariancję, znacząca część mapy ma bardzo jasny kolor, czyli niską kowariancję. Teraz zmienimy parametr σ na 1.5.

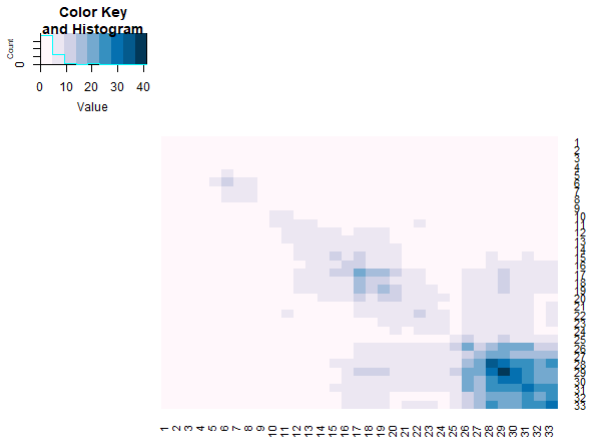
```
M<-1000
T<-10
N<-16
x0<-1
mu<-0
s<-1.5

k<-covariance_geom(M,T,N,x0,mu,s)

heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```



Rysunek 47: Mapa ciepła kowariancji.

Ciemniejsze obszary występują teraz na środku mapy, kowariancja na wszystkich brzegach jest w tym wypadku niska.
Na koniec sprawdzimy jeszcze jak wygląda wykres gdy zmienimy $N = 32$.

```
M<-1000
T<-10
N<-32
x0<-1
mu<-0
s<-1

k<-covariance_geom(M,T,N,x0,mu,s)

heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```



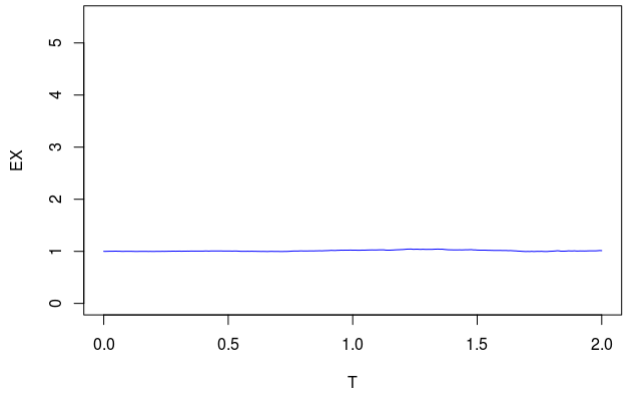
Rysunek 48: Mapa ciepła kowariancji.

Wykres wygląda porównywalnie do $N = 16$. Ciemniejsze obszary występują w prawym dolnym rogu mapy.

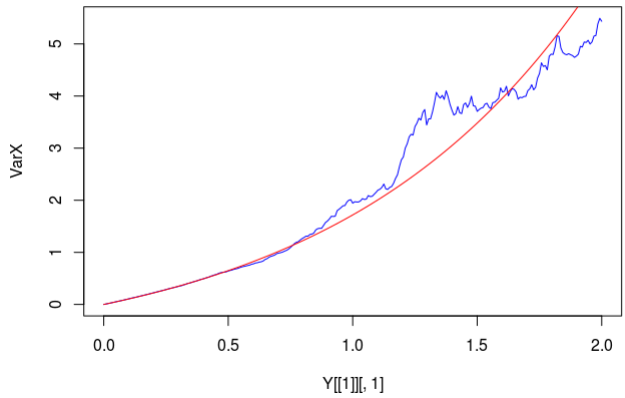
2.5 Funkcja wartości oczekiwanej i wariancji

Aby wyznaczyć funkcję wartości oczekiwanej i wariancji skorzystamy z napisanej już funkcji *EXVar*.

```
EXVar<-function(M, f, arg){

Y<-list()
Y2<-list()

for( i in 1:M){

Y[[i]]<-do.call(f,arg)
Y2[[i]]<-cbind(Y[[i]][,1], (Y[[i]][,2])^2)

}
suma<-do.call('rbind', Y)
suma<-aggregate(suma[,2], by=list(suma[,1]), FUN=sum)

suma2<-do.call('rbind', Y2)
suma2<-aggregate(suma2[,2], by=list(suma2[,1]), FUN=sum)

EX=suma[,2]/M
VarX=suma2[,2]/M-(EX)^2

plot(Y[[1]][,1], EX, type='l', xlab="T",
ylim =c(min(min(EX), min(VarX)),
max(max(EX), max(VarX))))
plot(Y[[1]][,1], VarX, col=2, type='l')
}

M<-10000
N<-256
T<-2
x0<-1
mu<-0
s<-1
f<-"geom_r_B"
arg<-list(N=N, T=T, x0=x0, mu=mu, s=s)
EXVar(M,f,arg)
curve((x0^2)*exp(2*mu*x)*(exp((s^2)*x)-1), add=TRUE, col=2)
```



Rysunek 49: Wartość oczekiwana.



Rysunek 50: Wariancja.

Teoretyczna wartość oczekiwana geometrycznego ruchu Browa wynosi $x_0 e^{\mu t}$ - w tym przypadku $x_0 = 1$, więc wykres wartości oczekiwanej jest poprawny. Teoretyczna wariancja natomiast wynosi $x_0^2 e^{2\mu t}(e^{\sigma^2 t} - 1)$- to również potwierdzone jest przez wykres wraz z nałożoną krzywą.

3 Most Browna

3.1 Proces ruchu

Korzystając ponownie z funkcji generującej standardowy proces ruchu Browna, tworzymy funkcję *most_Browna*.

```
pr_r_B<-function(N,T){

  delta_t<-T/N

  wektor_tj<-seq(0,T, by=delta_t)

  B<-c(0)

  ksi<-rnorm(N,0,1)

  for( i in 2:(N+1)){

    B[i]<-B[i-1]+sqrt(delta_t)*ksi[i-1]

  }
  return(cbind(wektor_tj, B))
}
most_Browna<-function(N,T,x,y){

  W<-c()

  B<-pr_r_B(N,T)

  t<-B[,1]

  for( i in 1:nrow(B)){
    W[i]<-x+B[i,2]-(t[i]/T)*(x+B[nrow(B),2]-y)
  }

  return(cbind(t,W))

}
```

Zaczniemy od wygenerowania mostu Browna dla argumentów $x, y = 1$.

```
N<-8
T<-10
x<-1
y<-1
most_Browna(N,T,x,y)
```

	t	W
1	0.00	1.0000000
2	1.25	0.1399735
3	2.50	1.5990199
4	3.75	3.5517984
5	5.00	2.0421568
6	6.25	-0.1712639
7	7.50	-1.1693160
8	8.75	-1.8800966
9	10.00	1.0000000

Rysunek 51: Wygenerowany proces.

Łatwo zauważyć, że wartości w $t = 0$ i $t = 10$ odpowiadają wartościom x i y.

```
N<-16
T<-10
x<-1
y<-2
most_Browna(N,T,x,y)
```

	t	W
1	0.000	1.0000000
2	0.625	0.9472766
3	1.250	0.4517630
4	1.875	1.3911865
5	2.500	1.8070016
6	3.125	1.6804769
7	3.750	1.3624440
8	4.375	0.9834097
9	5.000	-0.7055217
10	5.625	-1.3298590
11	6.250	-0.8684822
12	6.875	0.1559465
13	7.500	2.4089089
14	8.125	2.6333035
15	8.750	1.9482849
16	9.375	1.2339254
17	10.000	2.0000000

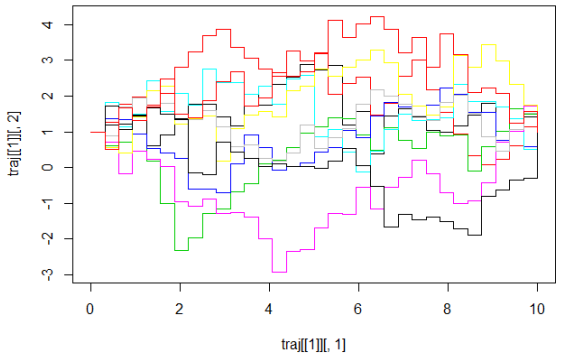
Rysunek 52: Wygenerowany proces.

Gdy zmienimy wartość y na 2, oraz dodatkowo wydłużymy podział, w $t = 0$ mamy niezmiennie wartość 1, a na końcu procesu czyli w $t = 10$ pojawiła nam się zmieniona wartość $y = 2$.

3.2 Trajektorie

Funkcja *wykresy_most* rysuje zadaną M ilość trajektorii mostu Browna.

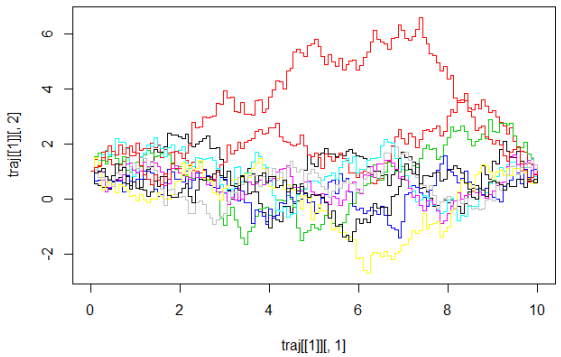
```
wykresy_most <-function(N,T, x, y,M){

  n<-c()

  traj <-list()

  ymin<-c()
  ymax<-c()

  for( i in 1:M){

    traj[[i]]<-most_Browna(N,T, x, y)
    n[i]<-nrow(traj[[i]])
    ymin[i]<-min(traj[[i]][,2])
    ymax[i]<-max(traj[[i]][,2])

  }

  plot(traj[[1]][,1],traj[[1]][,2],type='s',  ylim=c(min(ymin), max(ymax)))

  for( i in 2:M){

    lines(traj[[i]][,1],traj[[i]][,2],type='s',col=i)

  }

}

N<-32
T<-10
x<-1
y<-1
M<-10
wykresy_most(N,T, x, y,M)
```



Rysunek 53: Wygenerowane trajektorie.

Widzimy, że trajektorie mostu Browna mają zupełnie inny kształt. Dzięki zadaniu parametrów x i y zaczynają się one i kończą w ustalonych punktach.

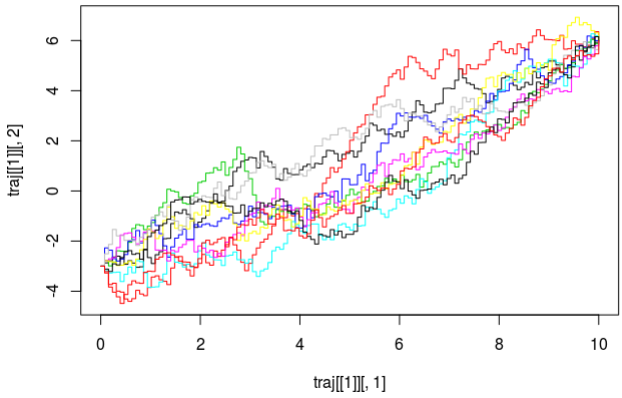
```
N<-128
T<-10
x<-1
y<-1
M<-10
wykresy_most(N,T, x, y,M)
```



Rysunek 54: Wygenerowane trajektorie.

Przy zmianie podziałki na 128 niezmienny pozostaje początek i koniec trajektorii.

```
N<-128
T<-10
x<-(-3)
y<-6
M<-10
wykresy_most(N,T, x, y,M)
```



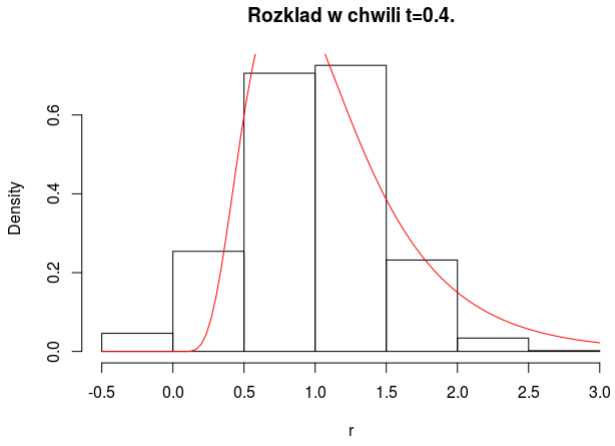
Rysunek 55: Wygenerowane trajektorie.

Ostatni wykres jest niesymetryczny, dzieje się tak ponieważ ustaliliśmy, że ma się on zaczynać w -3 a kończyć w 6.

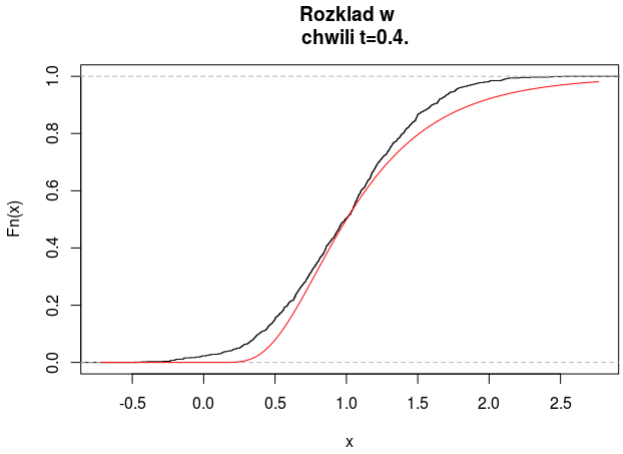
3.3 Rozkład w chwili t.

Funkcja *rozklad_most* bada rozkład mostu Browna w wybranej chwili t.

```
rozklad_most<-function(t,N,T,x, y,M){
        r<-c()
        traj<-list()
        for( i in 1:M){
                traj[[i]]<-most_Browna(N,T,x,y)
                r[i]<-traj[[i]][sum(traj[[i]][,1]<=t),2]
        }
        hist(r,10, prob=TRUE, main =
        paste("Rozklad w chwili t=" , t, ".", sep=""))
        curve(dlnorm(x,0,sqrt(t*(T-t)/T)), add=TRUE, col=2)
        plot(ecdf(r), main = paste("Rozklad w chwili t=",
         t, ".", sep=""))
        curve(plnorm(x,0,sqrt(t*(T-t)/T)), add=TRUE, col=2)
}

N<-64
T<-1
x<-1
y<-1
M<-1000
t<-0.4
rozklad_most(t,N,T,x, y,M)
```



Rysunek 56: Rozkład w chwili t.



Rysunek 57: Rozkład w chwili t.

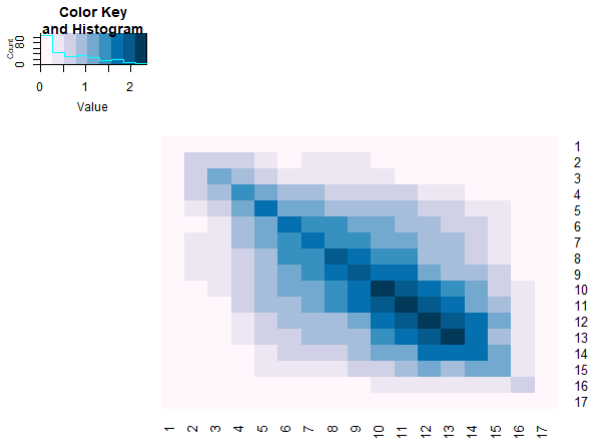
Rozkład w przykładowej chwili $t = 0.4$ zgadza się z rozkładem teoretycznym - jest to rozkład normalny o średniej 0 i wariancji $t * (T - t)/T$.

3.4 Funkcja kowariancji

Tak jak we wcześniejszych procesach, mamy funkcję *covariance_most* wyznaczającą kowariancję procesu.

```
covariance_most  <- function(M, T, N ,x,y){ #M=100

  kowariancja<-data.frame(matrix(data=NA, nrow=N+1, ncol=N+1))
  traj<-list()

  for (i in 1:M){

    traj[[i]] <- most_Browna(N,T,x,y)

  }

  wektor_czasow<-traj[[1]][,1]

  for( i in 1:length(wektor_czasow)){

    x<-c()

    for( j in 1:M){

      x[j]<-traj[[j]][which(traj[[j]][,1]==wektor_czasow[i]),2]

    }

    for(j in 1:length(wektor_czasow)) {

      y<-c()

      for( k in 1:M){

        y[k]<-traj[[k]][which(traj[[k]][,1]==wektor_czasow[j]),2]


      }

      kowariancja[i,j]<-cov(x,y)

    }

  }

  rownames(kowariancja)<-1:length(wektor_czasow)
  colnames(kowariancja)<-1:length(wektor_czasow)

  return(kowariancja)
}
```

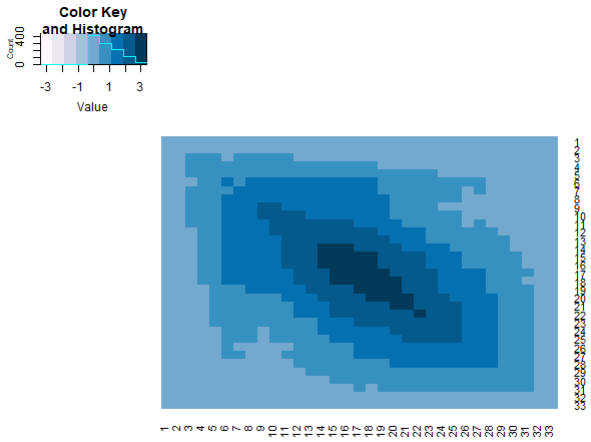
```
N<-16
T<-10
x<-1
y<-1
M<-100
k<-covariance_most(M,T,N,x,y)

heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```



Rysunek 58: Mapa ciepła kowariancji.

Okazuje się, że w przypadku mostu Browna największe wartości kowariancji skupiają się "wzdłuż przekątnej". Inaczej niż we wcześniejszych procesach, gdzie ciemniejsza część była bliżej prawego dolnego rogu mapy.

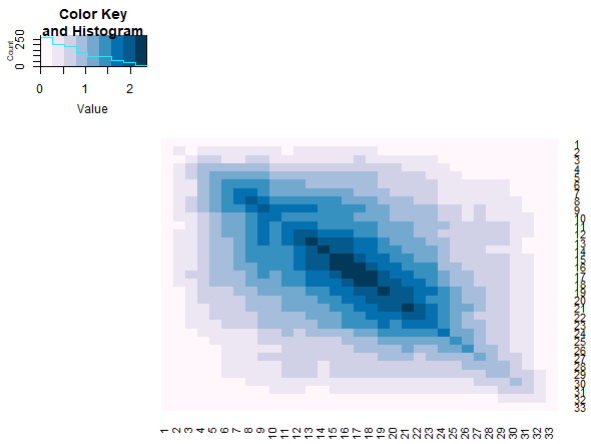
```
N<-32
T<-10
x<-1
y<-1
M<-100
k<-covariance_most(M,T,N,x,y)

heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```



Rysunek 59: Mapa ciepła kowariancji.

Przy zmianie N z 16 na 32 przyciemnia się cała mapa, co oznacza że kowariancje w poszczególnych momentach są wyższe, pozostaje jednak tendencja do ciemniejszych obszarów wzdłuż przekątnej.

```
N<-32
T<-10
x<-6
y<-1
M<-100

k<-covariance_most(M,T,N,x,y)

heatmap.2(as.matrix(k),Colv = NA, Rowv = NA,
 trace="none",col=brewer.pal(9,"PuBu") )
```



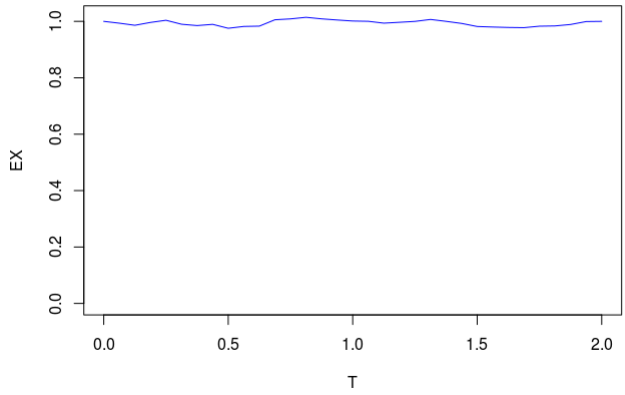
Rysunek 60: Mapa ciepła kowariancji.

Tym razem zmieniła się wartość $x = 6$. Po raz kolejny najwyższe kowariancje występują "na środku" z tendencją do występowania na przekątnej. Na brzegach obserwujemy jasny kolor, czyli niskie kowariancje.

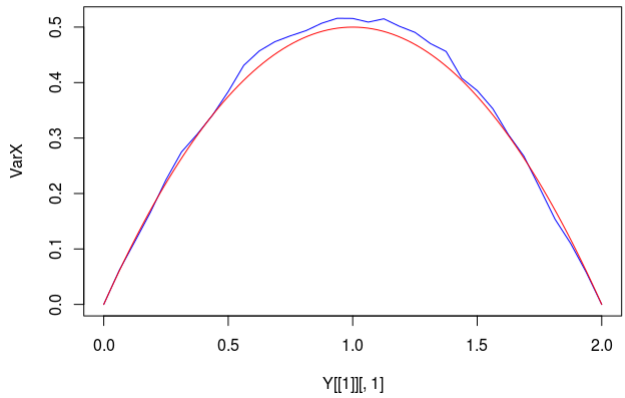
3.5 Funkcja wartości oczekiwanej i wariancji

Aby wyznaczyć funkcję wartości oczekiwanej i wariancji ponownie skorzystamy z napisanej już funkcji *EXVar*.

```
EXVar<-function(M, f, arg){

Y<-list()
Y2<-list()

for( i in 1:M){

Y[[i]]<-do.call(f,arg)
Y2[[i]]<-cbind(Y[[i]][,1], (Y[[i]][,2])^2)

}
suma<-do.call('rbind', Y)
suma<-aggregate(suma[,2], by=list(suma[,1]), FUN=sum)

suma2<-do.call('rbind', Y2)
suma2<-aggregate(suma2[,2], by=list(suma2[,1]), FUN=sum)

EX=suma[,2]/M
VarX=suma2[,2]/M-(EX)^2

plot(Y[[1]][,1], EX, type='l', xlab="T",
ylim =c(min(min(EX), min(VarX)),
max(max(EX), max(VarX))))
plot(Y[[1]][,1], VarX, col=2, type='l')
}

M<-1000
N<-32
T<-2
x<-1
y<-1
f<-"most_Browna"
arg<-list(N=N, T=T, x=x, y=y)
EXVar(M,f,arg)
curve(x*(T-x)/T, add=TRUE, col=2)
```



Rysunek 61: Wartość oczekiwana.



Rysunek 62: Wariancja.

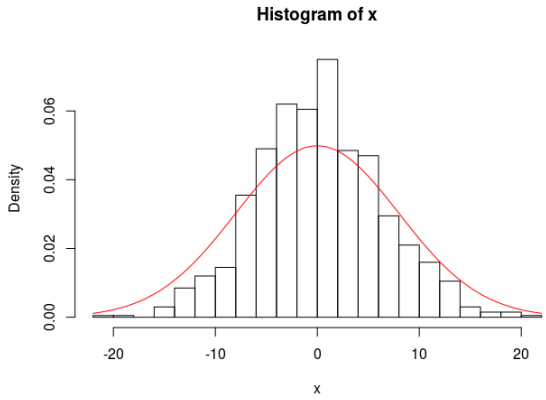
Teoretyczna wartość oczekiwana mostu Browa wynosi 1 - potwierdza to wykres wartości oczekiwanej. Teoretyczna wariancja natomiast wynosi $\frac{t(T-t)}{T}$- to również potwierdzone jest przez wykres wraz z nałożoną krzywą.

4 Zadania

4.1 Zadanie 2.8

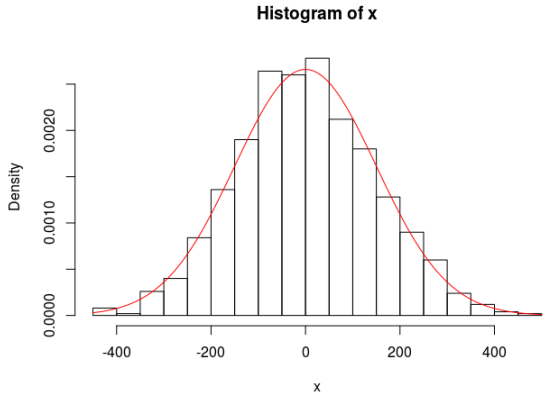
Zadanie 2-8: Dla standardowego procesu ruchu Browna $\{B_t, t \geqslant 0\}$ znajdź rozkład $B_1 + B_2 + ... + B_N$ dla ustalonej liczby $N \in \mathbb{N}$.

```
pr_r_B<-function(N,T){


        delta_t<-T/N

        wektor_tj<-seq(0,T, by=delta_t)

        B<-c(0)

        ksi<-rnorm(N,0,1)

        for( i in 2:(N+1)){

                B[i]<-B[i-1]+sqrt(delta_t)*ksi[i-1]

        }

        return(cbind(wektor_tj, B))
}

f<-function(N,T){

x<-replicate(1000, sum(pr_r_B(N,T)[,2]))
hist(x,20,prob=TRUE)
curve(dnorm(x,0,N*(N+1)*(2*N+1)/6),add=TRUE, col=2)

}

N<-10
T<-1
f(N,T)
```



Rysunek 63: Histogram rozkładu.

```
N<-256
T<-1
f(N,T)
```



Rysunek 64: Histogram rozkładu.

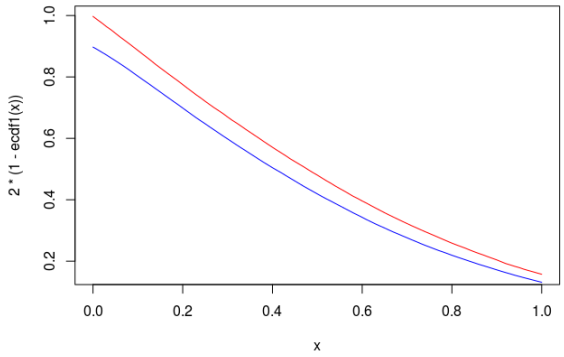
4.2 Zadanie 2.25

Zadanie 2-25: Niech $\{B_t, t \geqslant 0\}$ będzie standardowym procesem ruchu Browna. Zdefiniujmy proces $\{M_t, t \geqslant 0\}$, tzw. proces maksimum

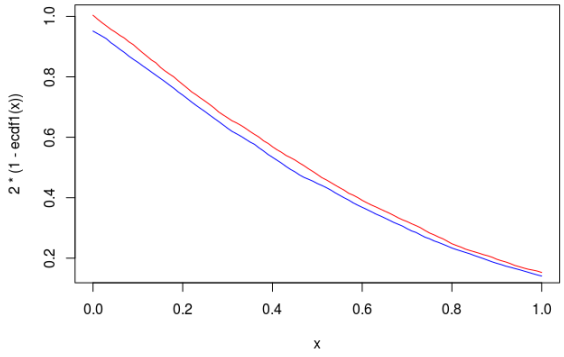
$$M_t = max_{0 \leqslant s \leqslant t}\{B_s\}.$$

Wykaż, że dla dowolnego $a > 0$, $P(M_t > a) = 2P(B_t > a) = P(|B_t| > a) = 2(1 - \Phi(\frac{a}{\sqrt{t}}))$.

```
proc_max<-function(N, T, t, M){ #M-ile trajektorii

traj<-list()
wt1<-c()
traj_max<-list()
wt2<-c()

for( i in 1:M){
traj[[i]]<-pr_r_B(N,T)
wt1[i]<-traj[[i]][sum(traj[[i]][,1]<=t),2]
traj_max[[i]]<-cbind(traj[[i]][,1], cummax(traj[[i]][,2]))
wt2[i]<-traj_max[[i]][sum(traj_max[[i]][,1]<=t),2]
}

x <- c(wt1, wt2)
ecdf1 <- ecdf(wt1)
ecdf2 <- ecdf(wt2)
curve(2*(1-ecdf1(x)),  col="red")
curve(1-ecdf2(x), col="blue", add=TRUE)
}

N<-64
T<-1
t<-0.5
M<-10000
proc_max(N,T,t,M)
```

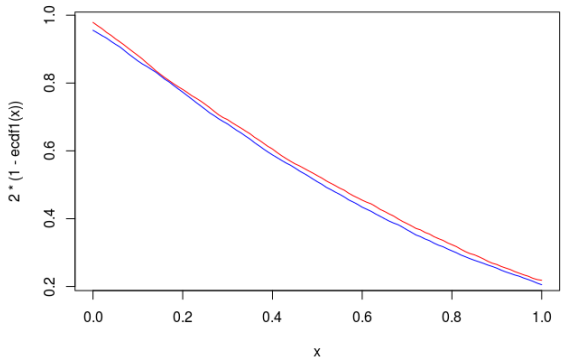



```
N<-256
T<-1
t<-0.5
M<-10000
proc_max(N,T,t,M)
```



Widzimy, że przy $N = 256$ krzywe są bardziej podobne do siebie. Zobaczmy jeszcze jak wyglądają krzywe przy zmianie t.

```
N<-256
T<-1
t<-0.7
M<-10000
proc_max(N,T,t,M)
```



Przy zmianie $t = 0.7$ krzywe są zbliżone do siebie.

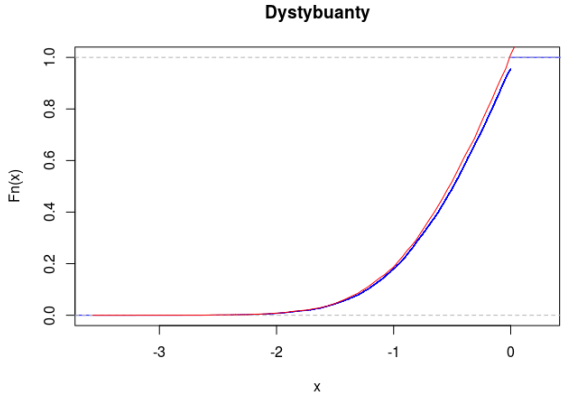
4.3 Zadanie 2.26

Zadanie 2-26: Niech $\{B_t, t \geqslant 0\}$ będzie standardowym procesem ruchu Browna. Zdefiniujmy proces $\{m_t, t \geqslant 0\}$, tzw. proces minimum

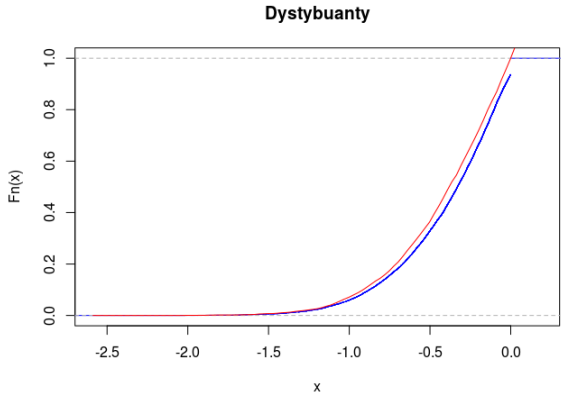
$$m_t = min_{0 \leqslant s \leqslant t}\{B_s\}.$$

Wykaż, że dla dowolnego $a < 0$, $P(M_t \leqslant a) = 2P(B_t \geqslant -a) = 2P(B_t \leqslant a)$.

```
proc_min<-function(N, T, t, M){ #M-ile trajektorii

        traj<-list()
        wt1<-c()
        traj_min<-list()
        wt2<-c()

        for( i in 1:M){

        traj[[i]]<-pr_r_B(N,T)
        wt1[i]<-traj[[i]][sum(traj[[i]][,1]<=t),2]
        traj_min[[i]]<-cbind(traj[[i]][,1], cummin(traj[[i]][,2]))
        wt2[i]<-traj_min[[i]][sum(traj_min[[i]][,1]<=t),2]
        }

        ecdf1 <- ecdf(wt1)
        ecdf2 <- ecdf(wt2)
        plot(ecdf2, col='blue', main="Dystybuanty")
        curve(2*ecdf1(x), col="red", add=TRUE)
}

N<-256
T<-1
M<-10000
t<-0.6
proc_min(N,T,t,M)
```




```
N<-256
T<-1
M<-10000
t<-0.3
proc_min(N,T,t,M)
```



W obu przypadkach dystrybuanty się pokrywają. Widzimy, że automatycznie oś x zawiera wartości ujemne, tak było podane w zadaniu ($a < 0$).

4.4 Zadanie 2.31

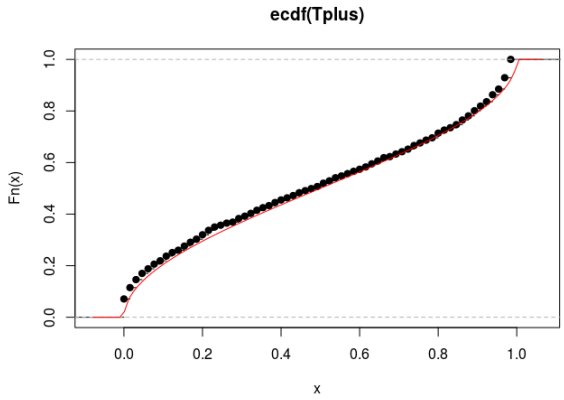
Zadanie 2-8: *(Prawo arcusa sinusa). Niech* $\{B_t, t \geqslant 0\}$ *będzie standardowym procesem ruchu Browna. Prawdopodobieństwo, że proces nie ma zer na odcinku* (a, b), $0 < a < b$, *wynosi* $\frac{2}{\pi} arcsin\sqrt{\frac{a}{b}}$. Zilustruj twierdzenie za pomocą symulacji komputerowych.

```
install.packages("VaRES")
library(VaRES)

pr_arcsin<-function(N,T, M){

        traj<-list()
        Tplus<-c()
        Tmax<-c()
        L<-c()
        for( i in 1:M){

         traj[[i]]<-pr_r_B(N,T)
         Tplus[i]<-sum(traj[[i]][,2]>0)/nrow(traj[[i]])
         Tmax[i]<-traj[[i]][which(traj[[i]][,2]==max(traj[[i]][,2])),1]

        j<-nrow(traj[[i]])

        while(j>1 && sign(traj[[i]][j,2])*sign(traj[[i]][j-1,2])>-1) {
                j<-j-1
        }

        L[i]<-traj[[i]][max(j-1,1),1]
}

plot(ecdf(Tplus))
curve(parcsine(x,a=0, b=1), add=TRUE, col=2)

hist(Tplus, 10, prob=TRUE)
curve(darcsine(x, 0, 1), add=TRUE, col=2)

plot(ecdf(Tmax))
curve(parcsine(x,a=0, b=1), add=TRUE, col=2)

hist(Tmax, 10, prob=TRUE)
curve(darcsine(x, 0, 1), add=TRUE, col=2)

plot(ecdf(L))
curve(parcsine(x,a=0, b=1), add=TRUE, col=2)

hist(L, 10, prob=TRUE)
curve(darcsine(x, 0, 1), add=TRUE, col=2)
}
```

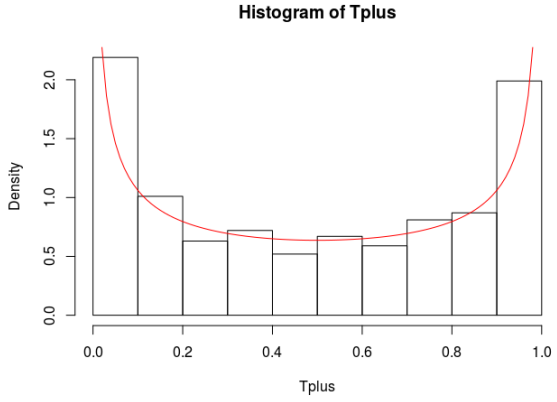
Do napisania funkcji *pr_arcsin* posłużyliśmy się pakietem *VaRES*, konkretnine funkcjami wyznaczającymi krzywe teoretyczne - *parcsine* i *darcsine*. Wywołajmy funkcję z parametrami $T = 1, N = 64, M = 1000$.

```
N<-64
T<-1
M<-1000
pr_arcsin(N,T,M)
```

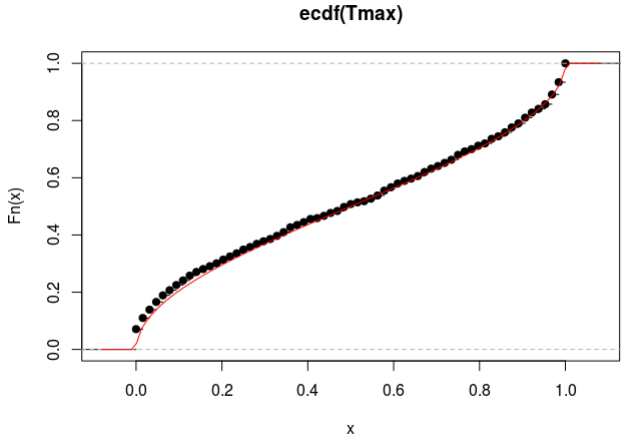
W wyniku wywołania tej funkcji otrzymamy 4 wykresy. Zobaczymy czy wykresy pochodzące z symulacji zgadzają się z krzywymi teoretycznymi.



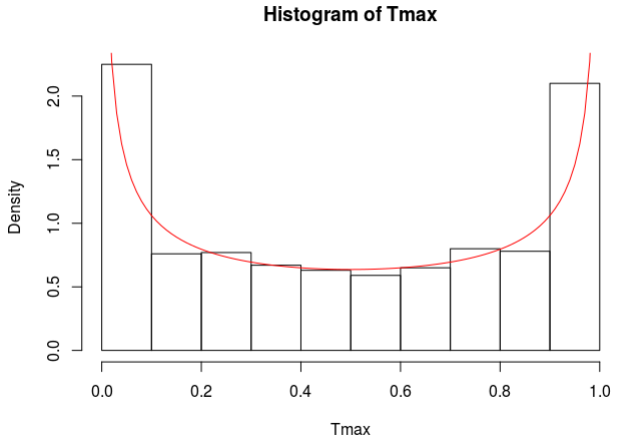
Rysunek 65: Dystrybuanta rozkładu części dodatniej procesu.



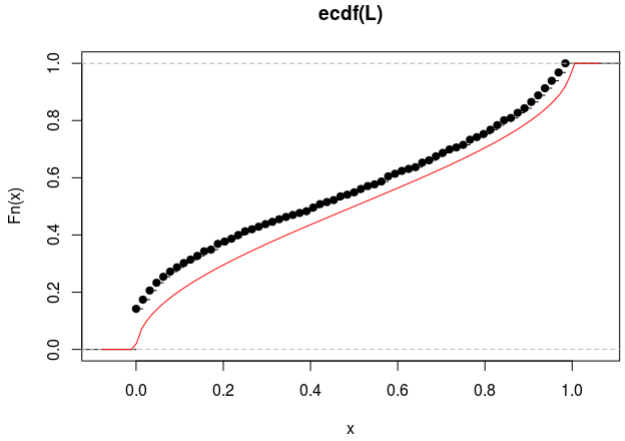
Rysunek 66: Histogram rozkładu części dodatniej procesu.



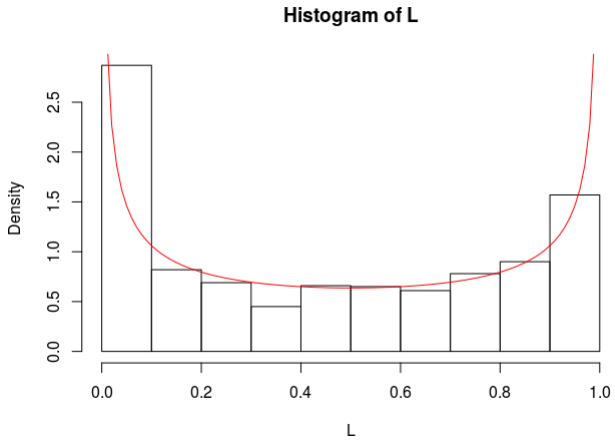
Rysunek 67: Dystrybuanta rozkładu wystąpienia maksimum procesu.



Rysunek 68: Histogram rozkładu wystąpienia maksimum procesu.



Rysunek 69: Dystrybuanta rozkładu wystąpienia ostatniego momentu "przed zerem".



Rysunek 70: Histogram rozkładu wystąpienia ostatniego momentu "przed zerem".

Histogramy i dystrybuanty pokrywają się z wartościami teoretycznymi, co daje nam dowód twierdzenia z użyciem symulacji.