Grundlagen der Simulation und Statistik von dynamischen Systemen

Brownschen Bewegung und drei Methoden für dessen Simulation.

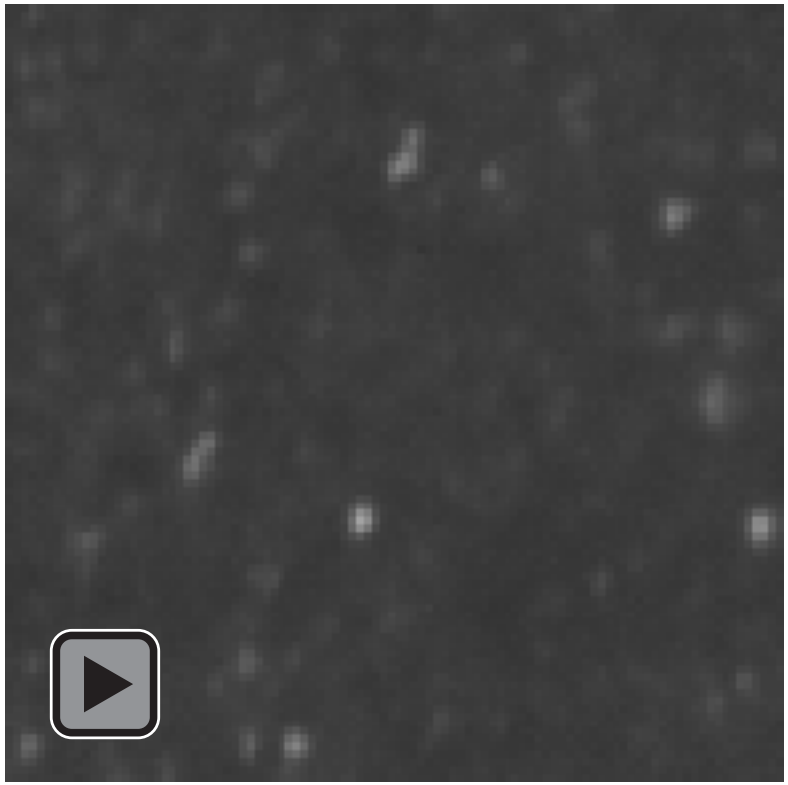
Jakob Richter

TU Dortmund: Fakultät Statistik

18. April 2012

Inhalt

Einleitung: Was ist die Brownsche Bewegung?



Einleitung: Was ist die Brownsche Bewegung?

Entdeckung & Ursache

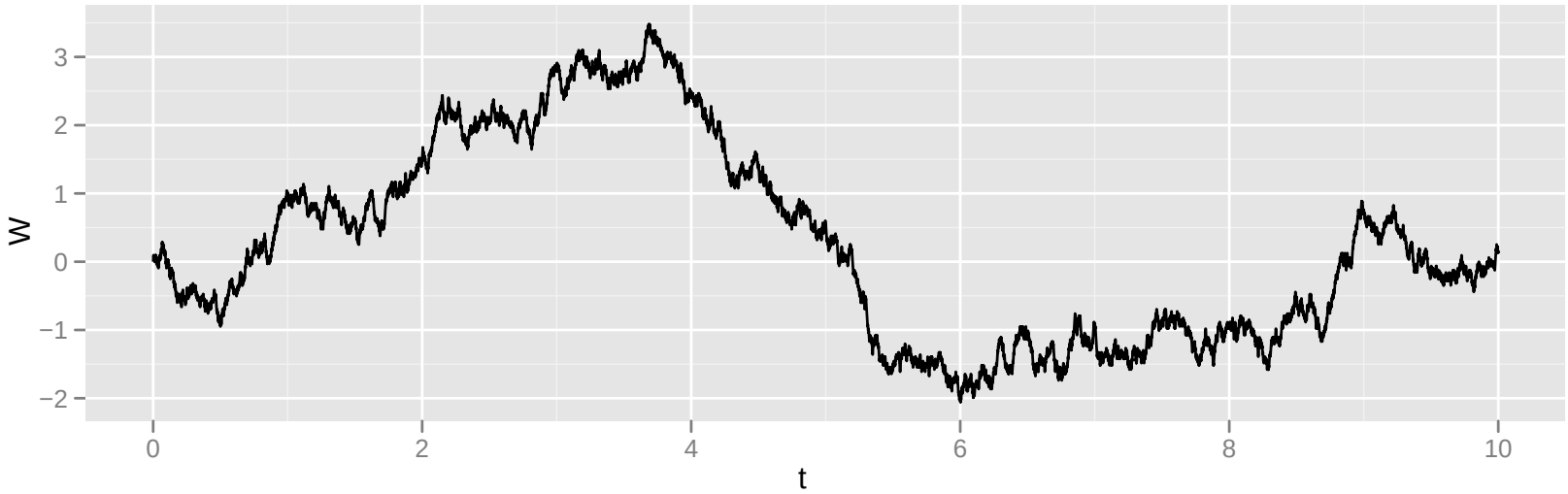
- 1828 entdeckte Robert Brown die unregelmäßige Bewegung von Pollen in Wasser
- 1863 Nachweis der Ursache durch Christian **Wiener**: Moleküle der Flüssigkeit prallen mit Teilchen zusammen
  - Etwa 10^{21} mal pro Sekunde
  - Mit gleicher Wahrscheinlichkeit aus allen Richtungen
  - Resultierende Bewegung: **Random Walk**

Einleitung: Der Wiener Prozess als Spezialfall

Hier behandelter Spezialfall:
Zufällige Bewegungsänderung in einer Dimension *(Wiener Prozess)*

Betrachtung . . .

- eines Merkmals
- im Verlauf einer Zeitspanne



Theorie: Wiener Prozess

Stochastischer Prozess

- $\Gamma = [0, \infty) \subset \mathbb{R}$ Zeitachse
- $W(t)$ Zufallsvariable in Abhängigkeit von $t \in \Gamma$
- W Zufallsvariable auf Wahrscheinlichkeitsraum $(\Omega, \mathcal{F}, \mathbb{P})$
- somit $\{W(t), t \in \Gamma\}$ Familie von Zufallsvariablen
- W messbare Abbildung: $W : \Omega \times \Gamma \to \mathbb{R}$.

Theorie: Wiener Prozess

Voraussetzungen für den Wiener Prozess

1. $W(0) = 0$ fast sicher: $\mathrm{P}(W(0) = 0) = 1$
2. W hat auf disjunkten Zeitabschnitten unabhängige Zuwächse: $\forall\, 0 \leqslant t_1 < t_2 \leqslant t_3 < t_4 : W(t_2) - W(t_1)$ und $W(t_4) - W(t_3)$ sind stochastisch unabhängig.
3. Die Zuwächse sind Normalverteilt: $\forall\, 0 \leqslant s < t : W(t) - W(s) \sim \mathrm{N}(0, t - s)$

Pfad des Wiener Prozesses:

- Punktemenge:$(W(t), t)$

Theorie: Wiener Prozess

Korollar

$$Cov(W(s), W(t)) = min(s, t)$$

Beweis für $s < t$:

$$\begin{aligned}
Cov(W(s), W(t)) &= Cov(W(s), W(s) + (W(t) - W(s))) \\
&= Cov(W(s), W(s)) + \\
&\quad Cov(W(s) - W(0), W(t) - W(s)) \\
\text{mit (2) der Def.:} &= Var(W(s)) + 0 \\
&= Var(W(s) - W(0)) \\
\text{mit (3) der Def.:} &= s \quad \square
\end{aligned}$$

Analog für den Fall $t < s$.

Theorie: Pseudozufallszahlen

Standardvariante in R: **Mersenne-Twister**

Nutzung garantiert:

- Pseudozufallszahlen entsprechen der Verteilung
- Reproduzierbarkeit
- Unabhängigkeit aufeinanderfolgender Zufallszahlen
- große Periodenlänge: keine Wiederholung des Musters

Simulation des Wiener Prozesses: Allgemeines Vorgehen

- Einteilung der Zeitachse in feste Schritte: $\Delta t > 0$.
- Berechne $W(t)$ für alle $t = k \cdot \Delta t$, $k \in \{1, \ldots, N\}$
- Berechne $W(t)$ aufbauend auf $W(t - \Delta t)$
- Beginne mit $W(0) = 0$

Simulation des Wiener Prozesses: Normalverteilte Zuwächse

Idee: Simuliere Zuwächse direkt aus Normalverteilung

Vorgehen

1. Ziehe Zufallszahl z aus der Standardnormalverteilung.
2. Setze $W((\mathrm{k}+1)\cdot\Delta\mathrm{t}) = W(\mathrm{k}\cdot\Delta\mathrm{t}) + z\cdot\sqrt{\Delta\mathrm{t}}$.
3. $\mathrm{k} = \mathrm{k}+1$
4. Wenn $\mathrm{k} \leqslant \mathrm{N}$, wiederhole von Schritt 1 an.

Simulation des Wiener Prozesses: Normalverteilte Zuwächse

Erfüllt dieser Algorithmus die Voraussetzungen?

1. $P(W(0) = 0) = 1$: Gegeben durch Algorithmus.
2. $\forall\, k_1 < k_2 \leqslant k_3 < k_4\ ,\ k_1, k_2, k_3, k_4 \in \{0, \ldots, N\}$:
   $W(k_2 \cdot \Delta t) - W(k_1 \cdot \Delta t)$ und $W(k_4 \cdot \Delta t) - W(k_3 \cdot \Delta t)$ sind stochastisch unabhängig: Wird durch Pseudozufallszahlengenerator hinreichend erfüllt.
3. $\forall\, 0 \leqslant k_1 < k_2\ ,\ k_1, k_2 \in \{0, \ldots, N\}$:
   $W(k_2 \cdot \Delta t) - W(k_1 \cdot \Delta t) \sim N(0, (k_2 - k_1) \cdot \Delta t)$ ist Erfüllt mit $S_n := W(k_2 \cdot \Delta t) - W(k_1 \cdot \Delta t) = X_{k_1+1} + \ldots + X_{k_2}$ und X_i $i \in \{1, \ldots, N\}$ u.i.v. $X_1 \sim N(0, \Delta t)$ gilt:
   $E(S_n) = 0$ und
   $Var(S_n) = \sum_{i=k_1+1}^{k_2} Var(X_i) = (k_2 - k_1) \cdot \Delta t$.

Simulation des Wiener Prozesses: Normalverteilte Zuwächse

R-Code

```
1  #Zeitintervall und Feinheit
2  T <- 1; N <- 100
3
4  #Zeitschritte
5  Delta <- T/N
6
7  #Zeit-Achse
8  t <- seq(0,T,length.out=N+1)
9
10 #Simulation und Aufsummierung der Zuwaechse
11 S <- cumsum(sqrt(Delta) * rnorm(N))
12
13 W <- c(0, S)                    #0 als ersten Wert.
14
15 plot(t,W, type="l")             #Linienplot
```

Simulation des Wiener Prozesses: Normalverteilte Zuwächse

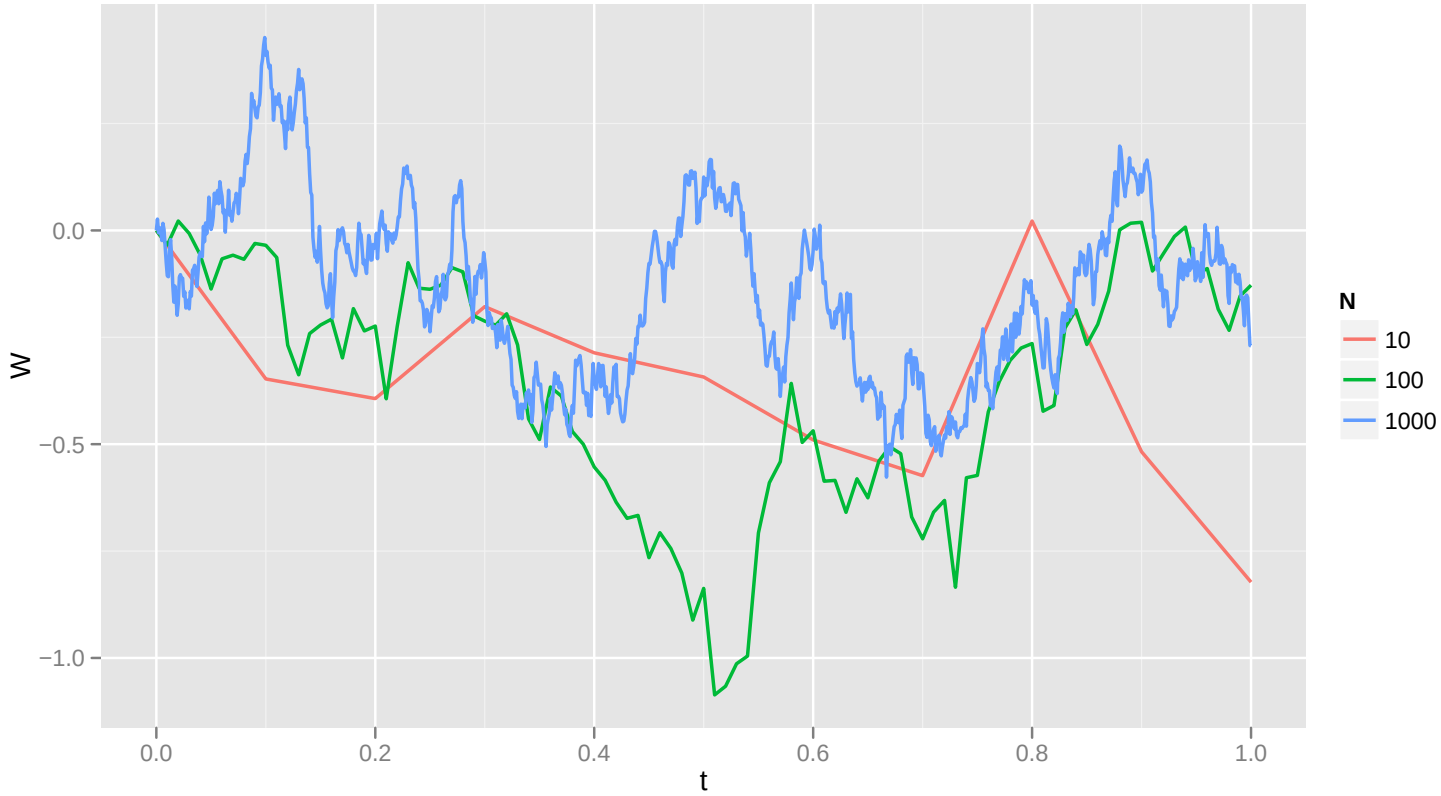


Abbildung: Simulierter Pfad der Brownschen Bewegung durch N normalverteilte Zuwächse

Simulation des Wiener Prozesses: Random Walk

Idee: Simuliere Pfad durch zufällige Richtung bei konstanter Bewegung.

Vorgehen

1. Wähle zufällig $z = -1$ oder $z = +1$ mit gleicher Wahrscheinlichkeit.
2. Setze $W((\mathrm{k}+1)\cdot\Delta\mathrm{t}) = W(\mathrm{k}\cdot\Delta\mathrm{t}) + z\cdot\sqrt{\Delta\mathrm{t}}$.
3. $\mathrm{k} = \mathrm{k}+1$
4. Wenn $\mathrm{k} \leqslant \mathrm{N}$, wiederhole von Schritt 1 an.

Simulation des Wiener Prozesses: Random Walk

Erfüllt dieser Algorithmus die Voraussetzungen?

1. & 2. analog zu oben.

3. $W(k_2 \cdot \Delta t) - W(k_1 \cdot \Delta t) \sim N(0, (k_2 - k_1) \cdot \Delta t)$:
   Gilt asymptotisch mit $n := k_2 - k_1 \to \infty$ und ZGWS für
   $S_n = X_{k_1+1} + \ldots + X_{k_2}, \quad Z := \frac{1}{\sqrt{\Delta t}} \cdot X$

$$P(Z = z) = \begin{cases} 0,5 & \text{für } z = -1 \\ 0,5 & \text{für } z = +1 \end{cases}$$

$$E(Z) = 0 \Rightarrow E(X) = 0 \Rightarrow E(S_n) = 0$$

$$Var(Z) = E(Z^2) + E(Z)^2 = 0,5 \cdot (-1)^2 + 0,5 \cdot (+1)^2 = 1$$

$$\Rightarrow Var(X) = Var(\sqrt{\Delta t} \cdot Z) = \Delta t$$

$$\Rightarrow Var(S_n) = Var(\sum_{i=k_1+1}^{k_2} X_i) = (k_2 - k_1) \cdot \Delta t$$

Simulation des Wiener Prozesses: Random Walk

R-Code

```
1  #Zeitintervall und Feinheit
2  T <- 1; N <- 100
3
4  #Zeitschritte
5  Delta <- T/N
6
7  #Zeit-Achse
8  t <- seq(0,T,length.out=N+1)
9
10 #Simulation und Aufsummierung der Zuwaechse
11 S <- cumsum(sample(x=c(-1,1),size=n,replace=T) * sqrt(
       Delta))
12
13 W <- c(0,S)                          #0 als ersten Wert.
14
15 plot(t,W, type="l")                  #Linienplot
```

Simulation des Wiener Prozesses: Random Walk

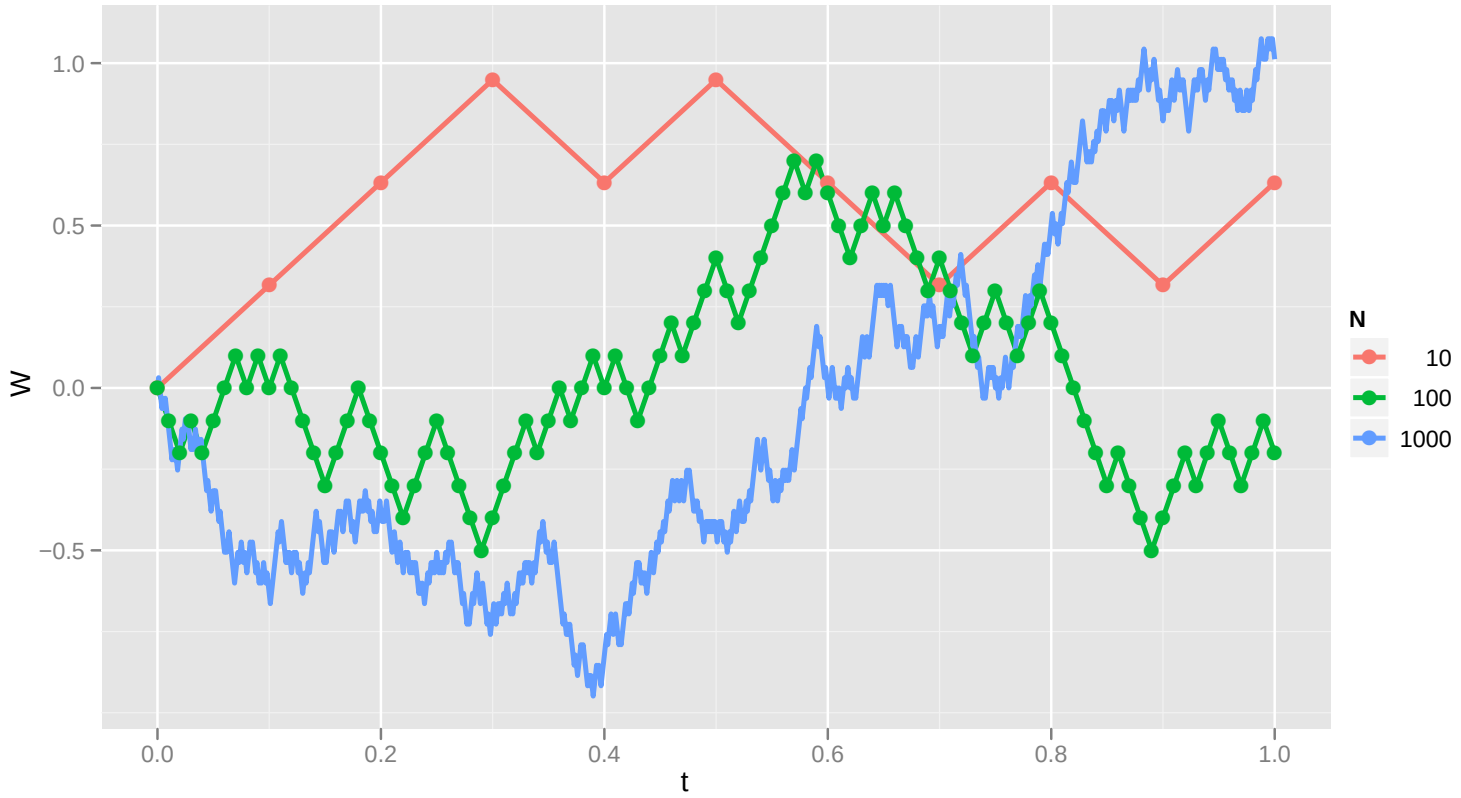


Abbildung: Simulierte Pfade der Brownschen Bewegung durch jeweils N Bernoulli-verteilte Zufallsvariablen. Die Punkte markieren die Stellen, an denen eine neue Bewegung simuliert wurde.

Simulation des Wiener Prozesses: Karhunen-Loève Approximation

Ansatz: Approximiere Wiener Prozess mit „glattem“ Pfad

Berechnung

Erzeuge N standardnormalverteilte Zufallszahlen $X_1, \ldots, X_N$

$$W(t) = \sum_{i=0}^{N} X_i \phi_i(t), \quad 0 \leqslant t \leqslant T$$

$$\text{mit} \quad \phi_i(t) = \frac{2\sqrt{2T}}{(2i+1)\pi} \cdot \sin\left(\frac{(2i+1)\pi t}{2T}\right)$$

Simulation des Wiener Prozesses: Karhunen-Loève Approximation

R-Code

```
1  phi <- function(i,t,T){
2  (2 * sqrt(2*T)) / ((2*i+1)*pi) * sin(((2*i+1)*pi*t)/(2
       *T))
3  }
4
5  Gen <- N*10 #Feinheit des Plots
6
7  t <- seq(0,T,length.out=Gen+1) #Zeitachse
8
9  Z <- rnorm(N) #Zufallszahlenerzeugung
10
11 W <- numeric(Gen+1) #Initialisierung der y-Achse
12
13 for(i in 2:(Gen+1)){
14     #Wende fuer jeden t-Wert die Formel an.
15     W[i] <- sum(Z*sapply(1:N, function(x) phi(x,t[i],T)
           ))
16 }
17 plot(t,W, type="l")
```

Simulation des Wiener Prozesses: Karhunen-Loève Approximation

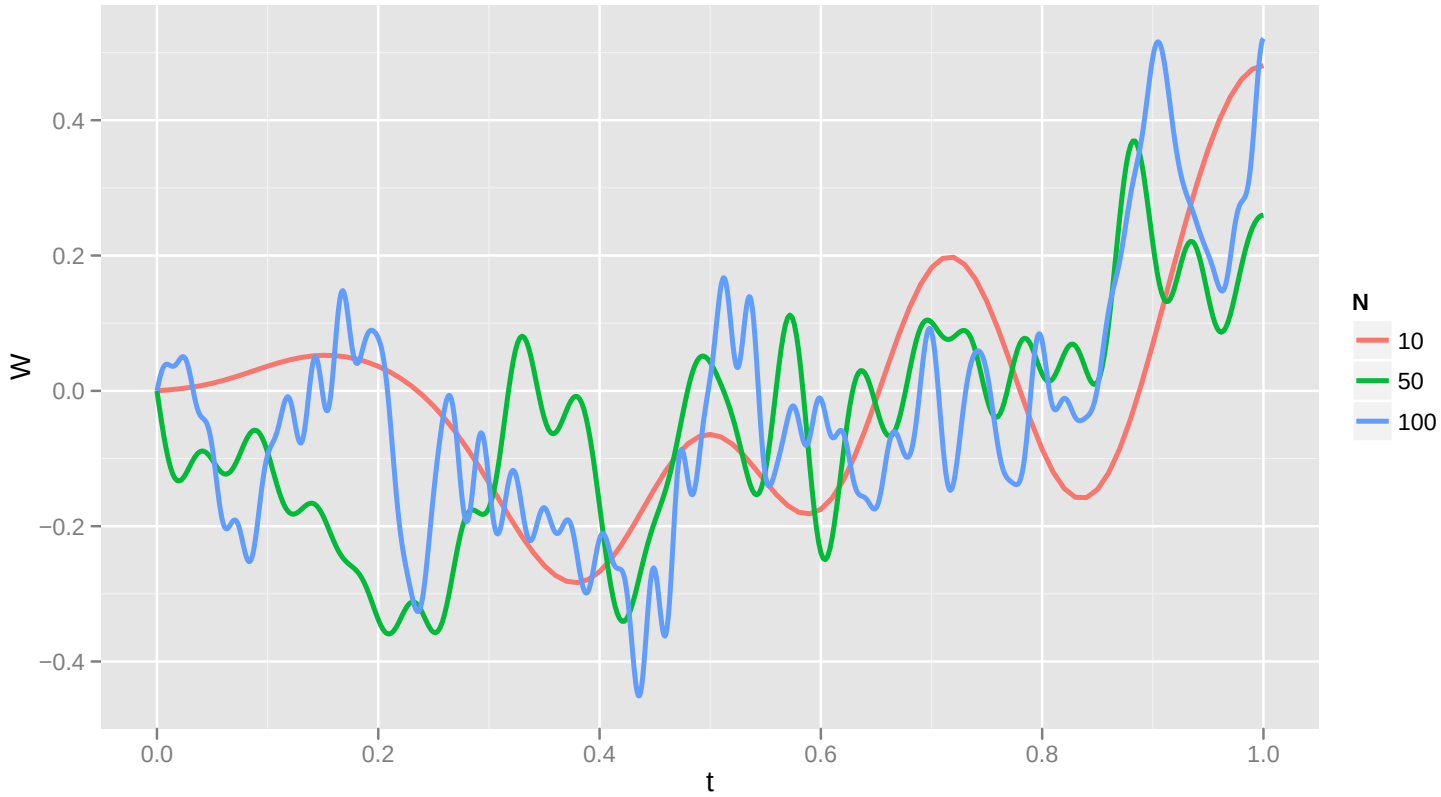


Abbildung: Simulierte Pfade der Brownschen Bewegung durch die Karhunen-Loève Approximation mit N Zufallszahlen.

Differenzierbarkeit des Wiener Prozesses

Satz

Der Pfad des Wiener Prozesses ist in keinem Punkt differenzierbar.

Differenzialquotient hat keinen Grenzwert:

$$\lim_{\Delta t \to 0} \frac{|W(t+\Delta t) - W(t)|}{\Delta t} \simeq \lim_{\Delta t \to 0} \frac{\sqrt{\Delta t}}{\Delta t} = +\infty \ ,$$

Differenzierbarkeit des Wiener Prozesses

Satz

Der Pfad des Wiener Prozesses ist in keinem Punkt differenzierbar.

Differenzialquotient hat keinen Grenzwert: **Veranschaulichung:**

$$\forall t \in [1, \infty) : P(|W(t + \Delta t) - W(t)| > l \cdot \Delta t)$$
$$= P(X < -l \cdot \Delta t \vee X > l \cdot \Delta t) \xrightarrow{\Delta t \to 0} 1 \text{ mit } X \sim N(0, \Delta t).$$

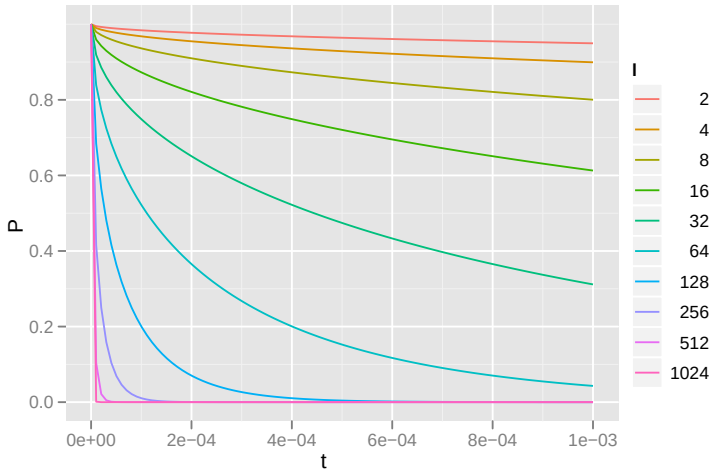


Abbildung: Obige Wahrscheinlichkeit geht auch für großes l bei kleinem Δt gegen 1.

Differenzierbarkeit des Wiener Prozesses

Veranschaulichung durch Simulation:
Betrachte Differentialquotient an sehr feiner Simulation mit normalverteilten Zuwächsen.

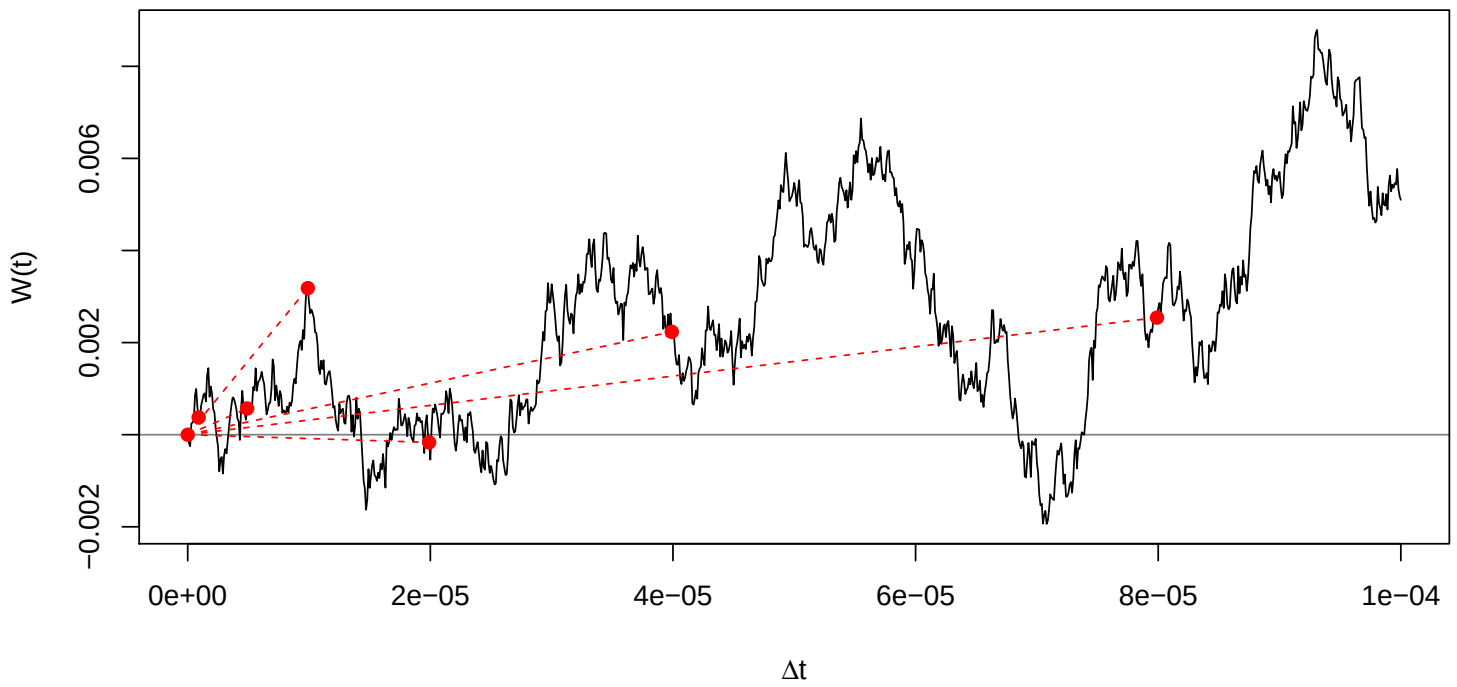


Abbildung: Differentialquotient bei $t = 0$ geht mit $\Delta t \downarrow 0$ gegen $+\infty$

Differenzierbarkeit des Wiener Prozesses

Veranschaulichung durch Simulation:
Betrachte Differentialquotient an sehr feiner Simulation mit normalverteilten Zuwächsen.

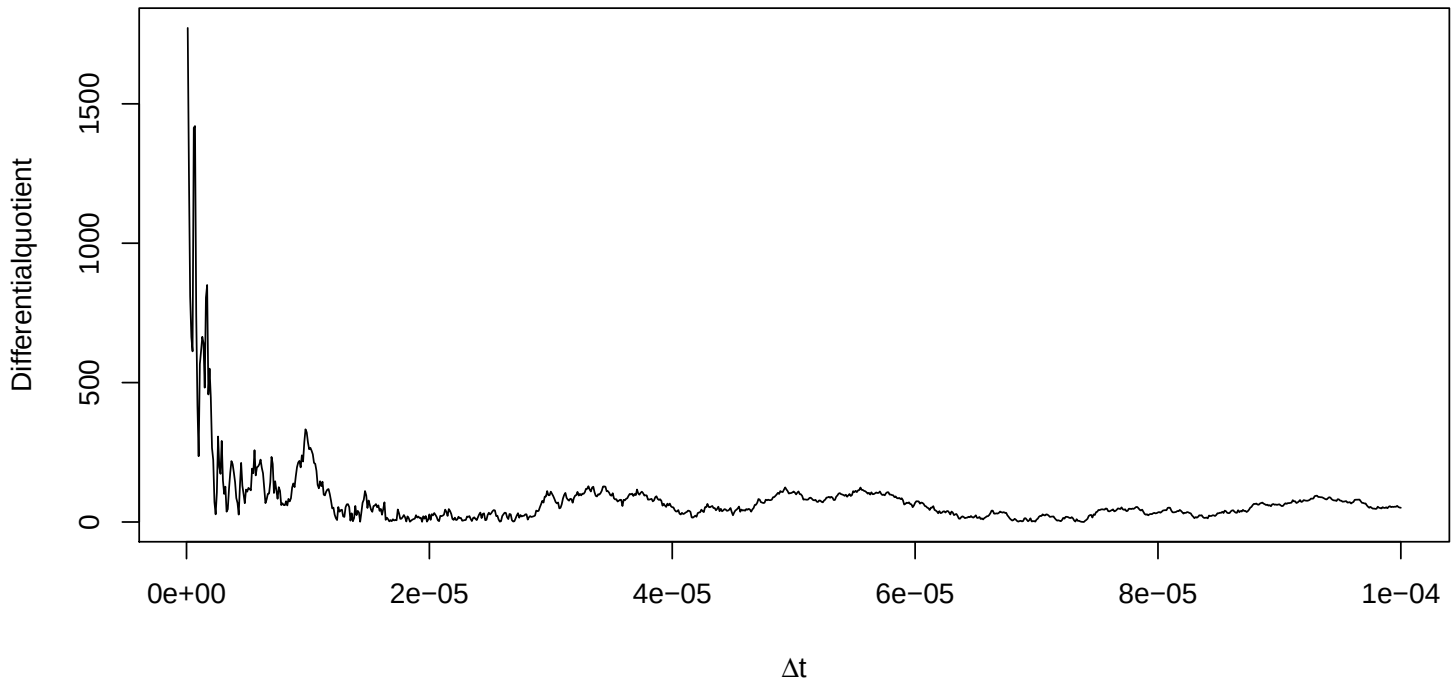


Abbildung: Differentialquotient bei $t = 0$ geht mit $\Delta t \downarrow 0$ gegen $+\infty$

Zusammenfassung

Wiener Prozess

- Zuwächse unabhängig
- Zuwächse Normalverteilt mit $\mathrm{E}(X_{\Delta t}) = 0$ und $\mathrm{Var}(X_{\Delta t}) = \Delta t$
- $\mathrm{Cov}(W(s), W(t)) = \min(s, t)$
- Pfad nirgends differenzierbar

Simulation durch normalverteilte Zuwächse

- benötigt wenig erzeugte Zufallszahlen für gutes Ergebnis

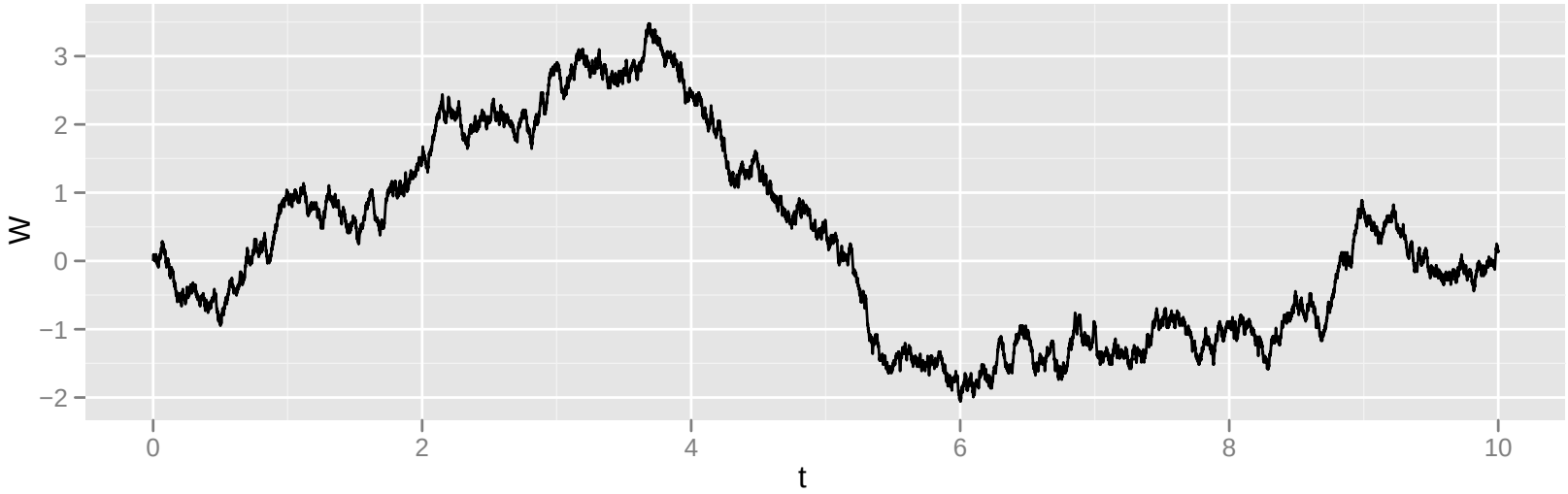
Simulation des Random Walks

- viele Zufallszahlen
- einfach zu erzeugende Zufallszahlen

Karhunen-Loève Approximation

- mit wenigen Zufallszahlen schon gute Approximation
- rechenintensiv

Zusammenfassung



Literatur

Literatur

- Iacus, Stefano M. (2008). *Simulation and Inference for Stochastic Differential Equations*. Springer.
- Klenke, A. (2008). *Wahrscheinlichkeitstheorie*. Springer.
- Meintrup, D. und Stefan Schäffler (2004). *Stochastik: Theorie Und Anwendungen*. Springer.

Software

- *R (2.13.2)*. R Development Core Team
- *ggplot2 (R Package, 2.13.1)*. Hadley Wickham

```
#Brownsche Prozesse
#Jakob Richter

library(ggplot2)
######################
# Aus Normalverteilung
######################
set.seed(44147)
T <- 1 #Zeitintervall
Ns <- c(10,100,1000)

plotres <- lapply(Ns, function(N){
 Delta <- T/N # Zeitschritte
 W <- c(0, cumsum( sqrt(Delta) * rnorm(N)))
 t <- seq(0,T, length.out=N+1)
 res <- cbind.data.frame(W=W,t=t,N=N)
 return(res)
})

plotres <- do.call("rbind",plotres)
plotres <- as.data.frame(plotres)
plotres$N <- as.factor(plotres$N)

pdf("normal.pdf",width=9,height=5)
g <- ggplot(plotres,aes(x=t,y=W,color=N))
g + geom_line(size=0.7)
dev.off()

#############
# Random Walk
#############
set.seed(44137)
T <- 1
ns <- c(10,100,1000)

randres <- lapply(ns,function(n){
 t <- seq(0,T,length.out=n+1)
 S <- cumsum(sample(x=c(-1,1),size=n,replace=T))
 W <- c(0,S)
 W <- W * sqrt(T/n) #hier liegt ein Fehler im Buch vor.
 cbind.data.frame(W=W,t=t,N=n)
})

randres <- do.call("rbind",randres)
randres <- as.data.frame(randres)
randres$N <- as.factor(randres$N)

pdf("limit.pdf",width=9,height=5)
h <- ggplot(randres,aes(x=t,y=W,color=N))
h <- h + geom_line(size=1)

#Punkte an den Sprungstellen fuer kleine N
datpoints <- randres[as.vector(randres$N)<=100,]
h + geom_point(data=datpoints,aes(x=t,y=W,color=N),size=3)
dev.off()

pdf("limit_step.pdf",width=9,height=5)
h <- ggplot(steprandres,aes(x=t,y=W,color=N))
h + geom_step(size=0.7)
dev.off()

##########################
# Karhunen-Loeve expansion
##########################
set.seed(44137)

phi <- function(i,t,T){
 (2 * sqrt(2*T)) / ((2*i+1)*pi) * sin(((2*i+1)*pi*t)/(2*T))
}

T <- 1
ns <- c(10,50,100) #sollten vielfache voneinander sein.
Zs <- rnorm(max(ns))

karres <- lapply(ns, function(n){
 Gen <- n*10
 t <- seq(0,T,length.out=Gen+1)
 #Waehlen immer die mittlere Zufallszahl aus.
 mitte <- ceiling(seq(from=max(ns)/(2*n), to=max(ns), by=max(ns)/n))
 Z <- Zs[mitte]
 W <- numeric(Gen+1)
 for(i in 2:(Gen+1)){
  W[i] <- sum(Z*sapply(1:n, function(x) phi(x,t[i],T)))
 }
 cbind.data.frame(W=W,t=t,N=n)
})

karres <- do.call("rbind",karres)
karres <- as.data.frame(karres)
karres$N <- as.factor(karres$N)

pdf("karhunen.pdf",width=9,height=5)
i <- ggplot(karres,aes(x=t,y=W,color=N))
i + geom_line(size=1)
dev.off()

####################
# Differenzierbarkeit
####################
set.seed(123)
N <- 1000
T <- 0.0001

Delta <- T/N # time increment
W <- c(0, cumsum( sqrt(Delta) * rnorm(N)))
t <- seq(0,T, length.out=N+1)
quot <- abs(W[-1]-W[1])/abs(t[-1]-t[1])
pdf("diffquot.pdf",width=9,height=5)
plot(t[-1],quot, type="l",ylab="Differentialquotient",xlab=expression(Delta*t))
dev.off()
pdf("diffquot_bsp.pdf",width=9,height=5)
plot(t,W,type="l",xlab=expression(Delta*t),ylab="W(t)")
abline(h=0,col=rgb(0,0,0,0.5))
itms <- c(10,50,100,200,400,800)
points(c(0,t[itms]),c(0,W[itms]),pch=19,col="red")
for(i in itms){
 lines(x=c(0,t[i]),y=c(0,W[i]),col="red",lty=2)
}
dev.off()

fac <- 2^(1:10)
dt <- seq(0.001,1e-12,length.out=100)
res <- lapply(fac, function(l) {
 ps <-sapply(dt,function(x) 1-(pnorm(l*x,sd=sqrt(x))-pnorm(-l*x,sd=sqrt(x))))
 cbind.data.frame(P=ps,l=l,t=dt)
})
res <-do.call("rbind",res)
res$l <- as.factor(res$l)

pdf("pgt.pdf",width=9,height=5)
g <- ggplot(data=res,aes(x=t,y=P,color=l))
g + geom_line()
dev.off()

pdf("pgtsml.pdf",width=6,height=4)
g + geom_line(size=0.5)
dev.off()
```