1.

The first equation *dx/dt = vcosφ* represents the velocity of the particle in the x-axis direction, which is equal to v times the cosine of the angle φ. The velocity of a particle in the x-axis direction is related to its current orientation φ.

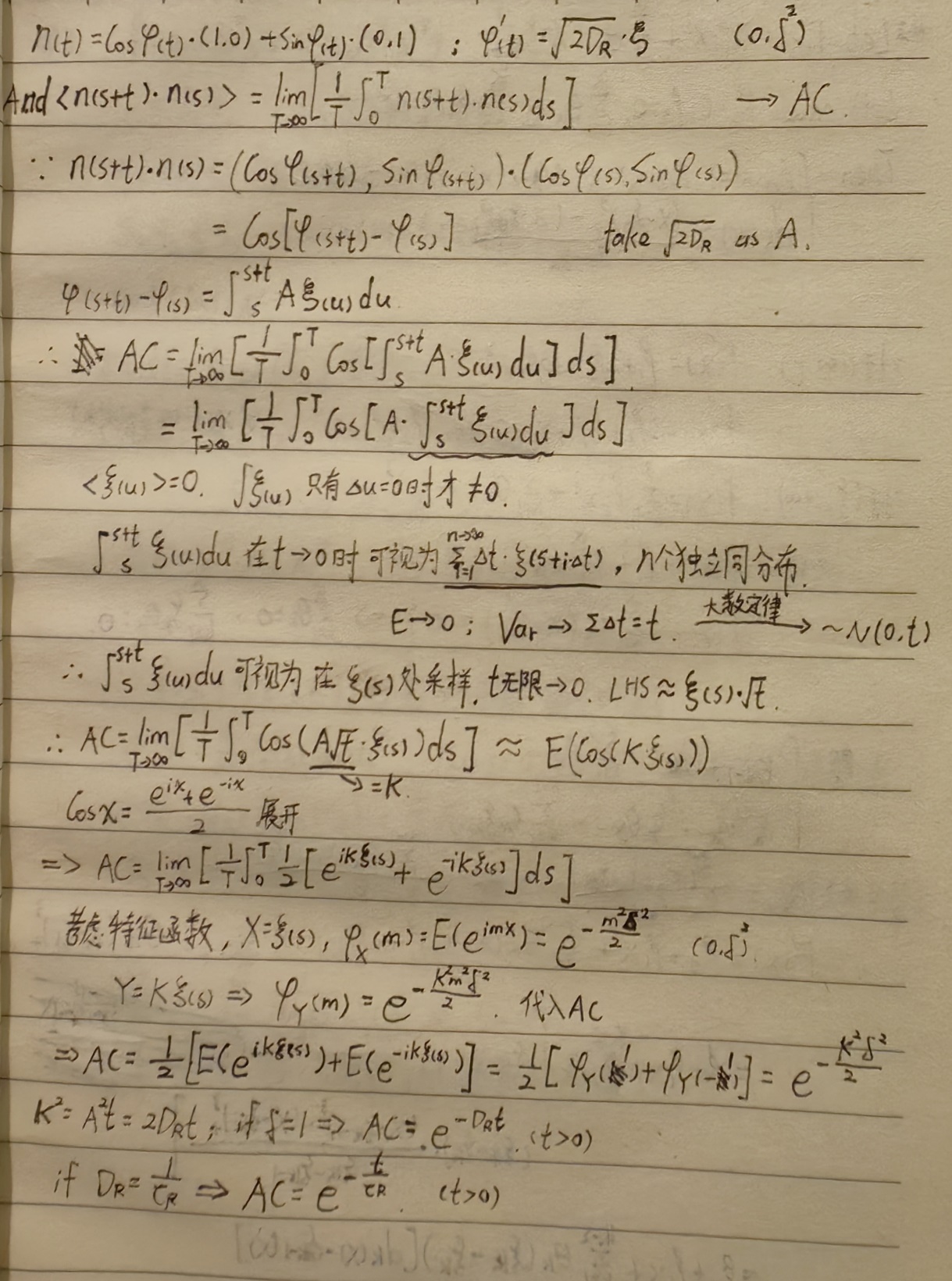
The second equation *dy/dt = vsinφ* represents the velocity of the particle in the y-axis direction, which is equal to v times the sine of the angle φ. The velocity of a particle in the y-axis direction is related to its current orientation φ.

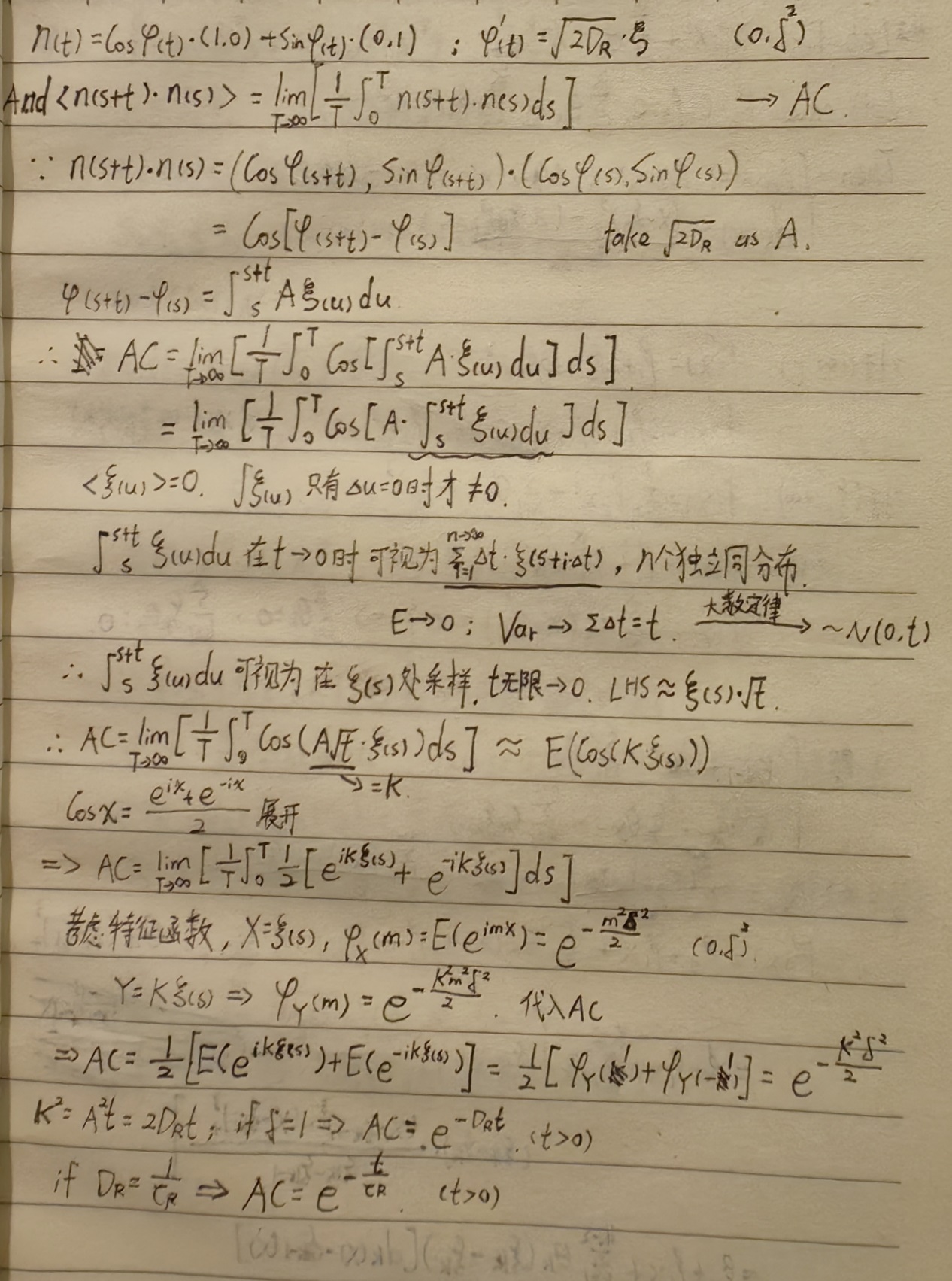
The third equation *dφ/dt = sqrt(2D\_R) \* ξ* describes the change of the particle's angle φ with time. Here ξ is a random variable obeying Gaussian white noise, representing the randomness caused by rotational diffusion. The sqrt(2D\_R) on the right side of the equation represents the intensity of the influence of the rotational diffusion coefficient D\_R on the noise.

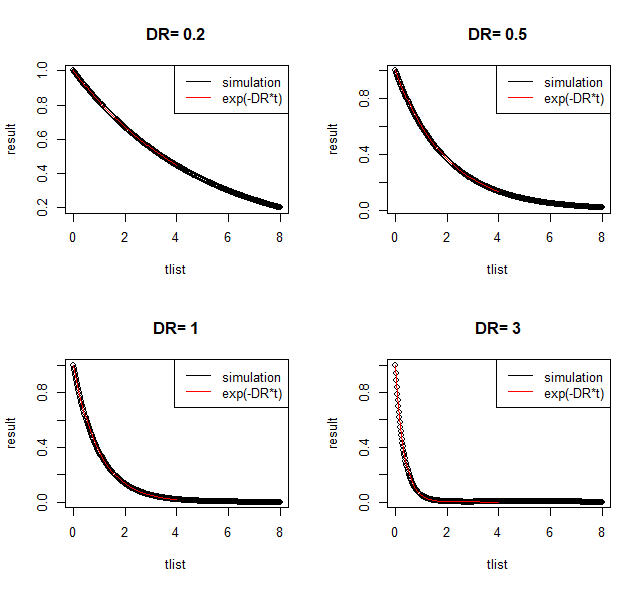
2.

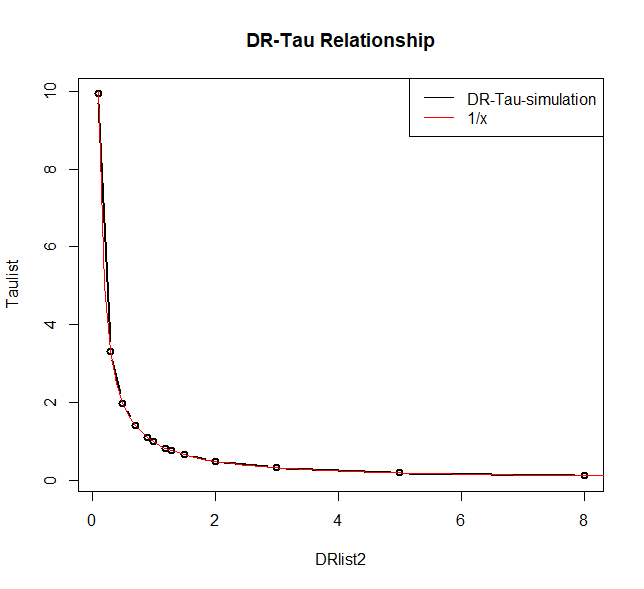
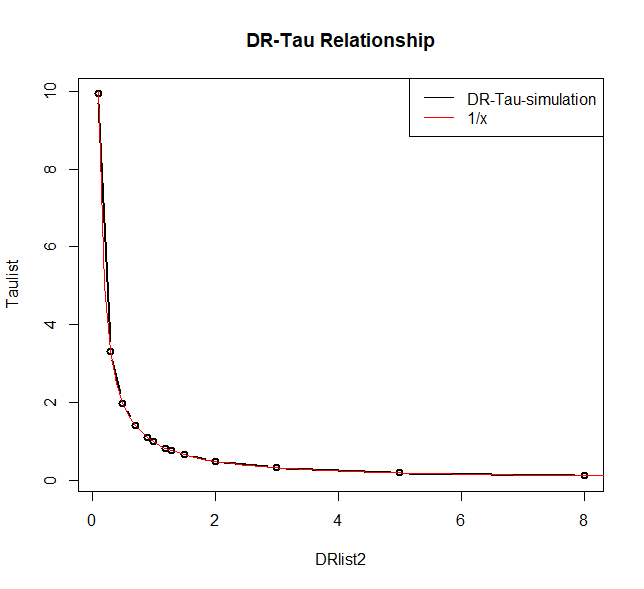
The meaning of this equation is that by calculating the dot product of the motion direction at different time points, we can obtain the autocorrelation of the motion direction. The autocorrelation function takes a maximum value of 1 when t=0, indicating that the movement direction is exactly the same at the same time point. As the time interval t increases, the autocorrelation function gradually decreases. The exponential decay form exp(-|t|/τ\_R) expresses the memory of the motion direction, that is, the motion direction maintains relevance within a short time scale, but gradually loses relevance as time increases. This equation reveals that the direction of motion of a self-propelled particle is continuous. The longer the direction duration τ\_R, the particle's direction of motion remains consistent for a longer period of time; while when τ is shorter, the particle's direction changes more frequently.

3.

Handwritting prove:



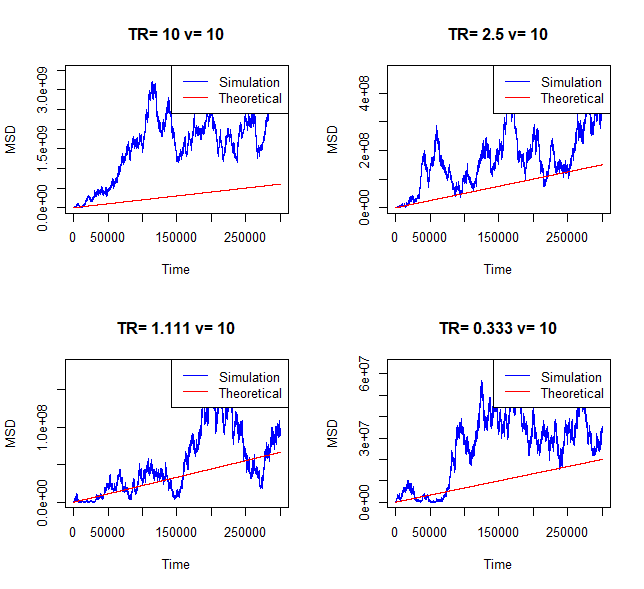
Simulation result:

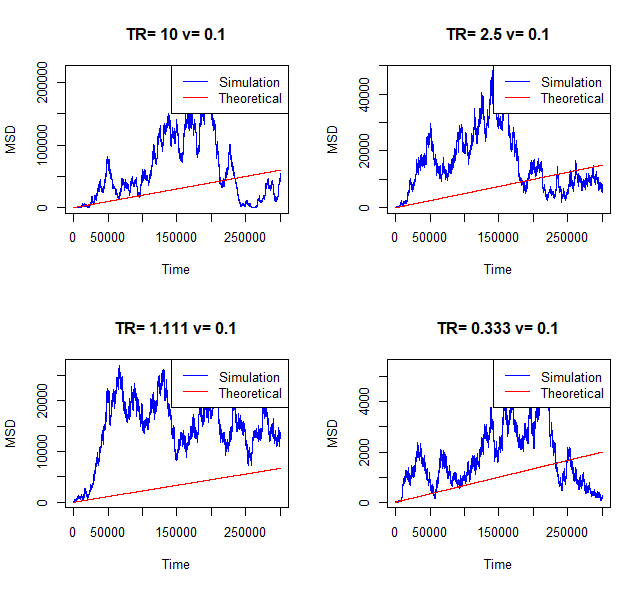
Find τ and compare the relationship between τ and D\_R :

4.

This equation shows that when t is much larger than τ\_R, the mean square displacement is proportional to time t. The growth rate of MSD(t) is linear and proportional to time t, as a particle moves with a constant velocity v over a longer time scale, its MSD also increases in a linear manner. The coefficient represents the magnitude of the displacement. When t is much larger than τ\_R, this product can be approximately regarded as a constant.

5.

Verify MSD:



Appendix

#3

DRlist=c(0.2,0.5,1,3)

tlist=seq(0,8,0.02)

integration=function(DR,t,T=10000){

set.seed(1)

samples=rnorm(n=T,0,1)

integrand=cos(sqrt(DR\*2)\*sqrt(t)\*samples)

integral=sum(integrand)/T

return(integral)

}

par(mfrow=c(2,2))

T=100000

result=c()

for (DR in DRlist){

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

result[i]=integration(DR,tlist[i],T)

}

plot(tlist,result,type="p",col="black",main=paste("DR=",DR))

curve(exp(-DR\*x),from=0,to=4,add=TRUE,col="red",lwd=1.5)

legend("topright", legend = c("simulation", "exp(-DR\*t)"), col = c("black", "red"), lty = 1)

}

par(mfrow=c(1,1))

#求Tau并对比Tau和DR的关系

model=function(t, Tau) exp(-t / Tau)

init\_params=list(Tau=1)

DRlist2=c(0.1,0.3,0.5,0.7,0.9,1,1.2,1.3,1.5,2,3,5,8)

Taulist=c()

for (DR in DRlist2){

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

result[i]=integration(DR,tlist[i],T)

}

fit=nls(result ~ model(tlist,Tau), start =init\_params)

Tau=coef(fit)["Tau"]

Taulist=c(Taulist,Tau)

}

plot(DRlist2,Taulist,type="b",col="black",lwd=2,main="DR-Tau Relationship")

curve(1/x,from=0,to=10,add=TRUE,col="red",lwd=1)

legend("topright", legend = c("DR-Tau-simulation", "1/x"), col = c("black", "red"), lty = 1)

#5

par(mfrow=c(2,2))

# 设置参数值

vlist=c(0.1,10) # 速度

DRlist3=c(0.1,0.4,0.9,3) # 扩散系数

# v=2 # 速度

# DR=0.5 # 扩散系数

# 设置模拟参数

nsteps=1000000 # 模拟步数

dt=0.3 # 时间步长

# 初始化变量

x=c(0) # 初始位置x

y=c(0) # 初始位置y

phi=c(0) # 初始角度φ

# 模拟运动轨迹

for (v in vlist) {

for (DR in DRlist3) {

TR=1/DR

set.seed(100\*v)

for (i in 2:nsteps) {

xi=rnorm(1) # 生成高斯白噪声

dx\_dt=v\*cos(phi[i-1]) # 计算x轴速度

dy\_dt=v\*sin(phi[i-1]) # 计算y轴速度

dphi\_dt=sqrt(2\*DR)\*xi # 计算角速度

x[i]=x[i-1]+dx\_dt\*dt # 更新x坐标

y[i]=y[i-1]+dy\_dt\*dt # 更新y坐标

phi[i]=phi[i-1]+dphi\_dt\*dt # 更新角度

}

# 计算模拟结果的 MSD

t=dt\*(1:nsteps) # 时间向量

MSD=x^2+y^2 # MSD = x^2 + y^2

# 计算理论结果的 MSD

MSD\_theoretical=(2\*v^2/DR)\*t

# 绘制模拟结果和理论结果

plot(t, MSD, type="l", col="blue", xlab="Time", ylab="MSD", main=paste("TR=",round(TR, 3),"v=",v))

lines(t, MSD\_theoretical, type="l", col="red")

legend("topright", legend=c("Simulation", "Theoretical"), col=c("blue", "red"), lty=1)

}

}