Chapter 10
KINETIC MONTE CARLO SIMULATIONS

Abstract While previous chapters looked at examples of MC methods with a fixed time interval, in this chapter we present several examples of Kinetic Monte Carlo methods (KMC). KMC methods are based on the concept that the variable time intervals between random events follow an exponential distribution. R codes are provided here for KMC methods applied to luminescence signals as stochastic birth and death processes. The chapter concludes with a different example of the KMC method, which describes quantum tunneling processes and the anomalous fading effect from a microscopic point of view. We compare the stochastic KMC results and the corresponding CV% uncertainties with the solution of the corresponding deterministic differential equation.

Code 10.1: Stochastic CW-OSL process using KMC

```
#  Stochastic CW-OSL process using KMC
rm(list = ls(all=T))
library(matrixStats)
mcruns<-100
n0<-500
mu<-.03
tmax<-100
times<-(1:tmax)
nMatrix<-matrix(NA,nrow=tmax-1,ncol=mcruns)
system.time(
for (k in 1:mcruns)
{n<-n0
 allt<-t<-0.5
    for(i in 1:n){
    P<-mu
```

```
    t<-t+rexp(1)/(P*n)
    n<-n-1
    allt<- rbind(allt,t)
    }
 depth.class <- cut(allt,times, include.lowest = TRUE)
 singlerun <- tapply(seq(from=n0,to=0,by=-1), depth.class,
mean,na.rm = FALSE)
 singlerun<-as.vector(singlerun)
 nMatrix[,k]<-singlerun
})
par(mfrow=c(1,3))
matplot(nMatrix,typ="l",col="green",ylim=c(0,700),
xlab="Time [s]",ylab="Remaining electrons")
timesMC<-seq(from=1,to=tmax-1,by=1)
avgn<-rowMeans(nMatrix,na.rm=TRUE)
avgCWOSL<-mu*avgn
sd<-rowSds(nMatrix,na.rm=TRUE)
cv<-100*sd/avgn
pch<-c(NA,NA,1,NA)
lty<-c(NA,NA,NA,"solid")
col<-c(NA,NA,"black","red")
lines(timesMC,avgn,ylab="Remaining electrons n(t)",
     xlab="Time [s]")
curve(n0*exp(-P*x),from=0,to=150,add=TRUE,col="red",lwd=2)
legend("topright",bty="n",c("(a)  n(t)"," ","MC (M=100)",
  "Analytical"),pch=pch,lty=lty,col=col)
plot(timesMC,avgCWOSL,ylab="CWOSL",xlab="Time [s]",ylim=c(0,21))
legend("topright",bty="n",c("(b)  CW-OSL"," ","MC (M=100)",
   "Analytical"),pch=pch,lty=lty,col=col)
curve(n0*mu*exp(-mu*x),from=0,max(times),add=TRUE,
col="red",lwd=2)
plot(timesMC,cv,ylab="CV[%]",xlab="Time [s]",ylim=c(0,40))
curve(100*sqrt((exp(mu*x)-1)/n0),0,max(times),add=TRUE,
      col="red",lwd=2)
legend("topleft",bty="n",c("(c)  CV[%]"," ","MC (M=100)",
   "Analytical"),pch=pch,lty=lty,col=col)

  ##    user  system elapsed
  ##    0.37    0.00    0.38
```

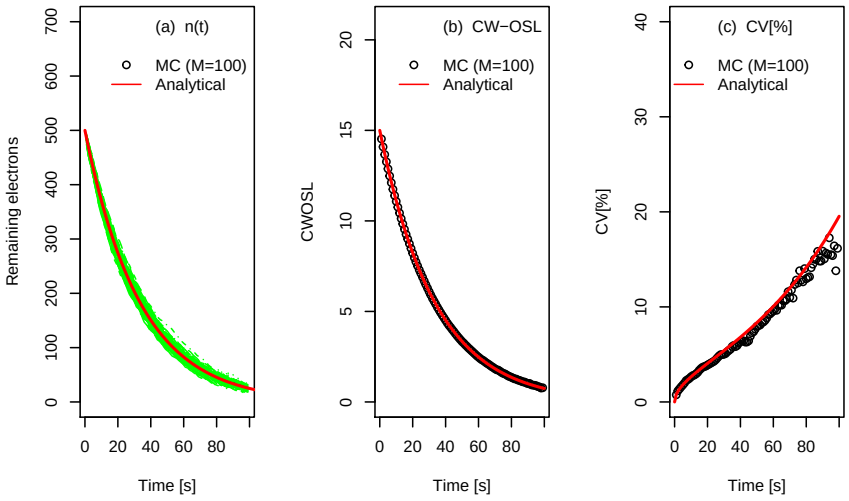

Fig. 10.1: Simulation of CW-OSL as a stochastic death process, using KMC.

Code 10.2: Stochastic LM-OSL process using KMC

```
# Stochastic LM-OSL process using KMC
rm(list = ls(all=T))
library(matrixStats)
mcruns<-100
n0<-500
tmax<-100
A<-.1
nMatrix<-matrix(NA,nrow=tmax-1,ncol=mcruns)

times<-seq(1,tmax-1,1)
system.time(
for (k in 1:mcruns)
{n<-n0
 allt<-t<-1
    for(i in 1:n-1){
    P<-A*t/tmax
    t<-t+rexp(1)/(P*n)
    n<-n-1
```

```
    allt<- rbind(allt,t)
    }
 depth.class <- cut(allt,seq(1:tmax), include.lowest = TRUE)
 singlerun <- tapply(seq(from=n0,to=0,by=-1), depth.class,
mean,na.rm = FALSE)
 singlerun<-as.vector(singlerun)
 nMatrix[,k]<-singlerun
})
par(mfrow=c(1,3))
matplot(nMatrix,typ="l",col="green",ylim=c(0,700),
xlab="Time [s]",ylab="Remaining electrons")
avgn<-rowMeans(nMatrix,na.rm = TRUE)
cv<-100*rowSds(nMatrix,na.rm=TRUE)/avgn
# plots
pch<-c(NA,NA,1,NA)
lty<-c(NA,NA,NA,"solid")
col<-c(NA,NA,"black","red")
lines(seq(from=1,to=tmax-1,by=1),avgn,lwd=2,col="black")
curve(n0*exp(-A*x^2/(2*(tmax-1))),1,tmax,lwd=3,
col="red",add=TRUE)
legend("topright",bty="n",c("(a)  n(t) for LM-OSL",
" ","MC","Analytical"),pch=pch,lty=lty,col=col)
avgLM<-A*(times/(tmax-1))*as.numeric(avgn)
plot(times,avgLM,ylab="LMOSL",xlab="Time [s]",ylim=c(0,15))
legend("topright",bty="n",c("(b)  LM-OSL"," ","MC",
"Analytical"), pch=pch,lty=lty,col=col)
curve(A*n0*exp(-A*x^2/(2*tmax))*x/tmax,0,tmax,add=TRUE,
col="red",lwd=2)

plot(times,cv,ylab="CV[%]",xlab="Time [s]",ylim=c(0,120))
curve(100*sqrt((exp(A*x^2/(2*tmax))-1)/n0),0,tmax,add=TRUE,
col="red",lwd=2)
legend("topleft",bty="n",c("(c)  CV[%]"," ","MC (M=100)",
 "Analytical"), pch=pch,lty=lty,col=col)
```

```
##    user  system elapsed
##    0.39    0.00    0.39
```

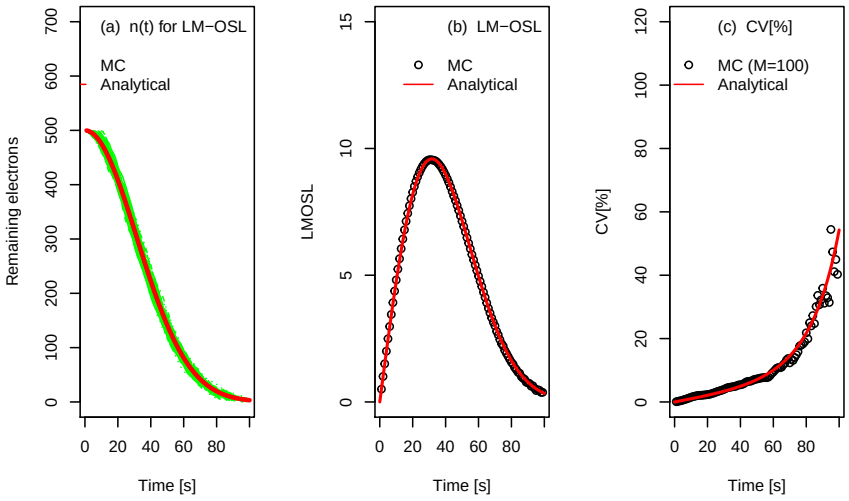

Fig. 10.2: Simulation of stochastic LM-OSL process using KMC method. The parameters are n_0 =500 trapped electrons at time $t = 0$, maximum stimulation time $P = 100$ s, LM-OSL stimulation rate $A = 0.1$ s^{-1}, and $M = 100$ MC runs. (a) Plot of the $M = 100$ MC runs, (b) Plot of the average MC $< n(t) >$, (c) Plot of the coefficient of variation *CV[%]*. The solid lines in these graphs represent the analytical solutions for the LM-OSL deterministic process.

Code 10.3: Stochastic first order TL process using KMC

```
# Stochastic first order TL process using KMC
rm(list = ls(all=T))
library(matrixStats)
mcruns<-100
n0<-500
tmax<-110
s<-1e13
E<-1
kb<-8.617e-5

times<-seq(1,tmax-1,1)
nMatrix<-matrix(NA,nrow=tmax-1,ncol=mcruns)
```

```
system.time(
for (k in 1:mcruns)
{n<-n0
 allt<-t<-30
    for(i in 1:n-1){
    P<-s*exp(-E/(8.617e-5*(t+273)))
    t<-t+rexp(1)/(P*n)
    n<-n-1
    allt<- rbind(allt,t)
    }
 depth.class <- cut(allt,seq(1:tmax), include.lowest = TRUE)
 singlerun <- tapply(seq(from=n0,to=0,by=-1), depth.class,
mean,na.rm = FALSE)
 singlerun<-as.vector(singlerun)
 nMatrix[,k]<-singlerun
})
# Calculate analytical solution
x1<-times+273
k<-function(u) {integrate(function(p){s*exp(-E/(kb*p))},
                          273,u)[[1]]}
y1<-lapply(x1,k)
x<-unlist(x1)
y<-unlist(y1)
errn<-sqrt(n0*(exp(-y)-exp(-2*y)))
nanalyt<-n0*exp(-y)
TLanalyt<-n0*s*exp(-E/(kb*x))*exp(-y)
# plot MC and analytical
par(mfrow=c(1,3))
matplot(nMatrix,typ="l",col="green",xlim=c(30,100),
ylab="Remaining electrons",
ylim=c(0,700),xlab=expression("Temperature ["^"o"*"C]"))
avgn<-rowMeans(nMatrix,na.rm = TRUE)
cv<-100*rowSds(nMatrix,na.rm=TRUE)/avgn
pch<-c(NA,NA,1,NA)
lty<-c(NA,NA,NA,"solid")
col<-c(NA,NA,"black","red")
lines(seq(from=1,to=tmax-1,by=1),avgn,lwd=2,col="black",
typ="p",pch=1)
legend("topright",bty="n",c("(a)  n(t)"," ","MC",
 "Analyt."),pch=pch,lty=lty,col=col)
lines(x-273,nanalyt,col="red",lwd=2)
k<-function(u){s*exp(-E/(kb*(u+273)))}
y<-lapply(times,k)
TLMC<-as.numeric(y)*as.numeric(avgn)
```

```
plot(times,TLMC,col="black",ylab="TL",
ylim=c(0,23),xlab=expression("Temperature ["^"o"*"C]"))
lines(x-273,TLanalyt,col="red",lwd=2)
legend("topleft",bty="n",c("(b)  TL"," ","MC",
  "Analyt."), pch=pch,lty=lty,col=col)
plot(times,cv,ylab="CV[%]",ylim=c(0,150),col="blue",
     xlab=expression("Temperature ["^"o"*"C]"))
legend("topleft",bty="n",c("(c)  CV[%]"," ","MC"))

  ##    user  system elapsed
  ##    0.41    0.00    0.40
```

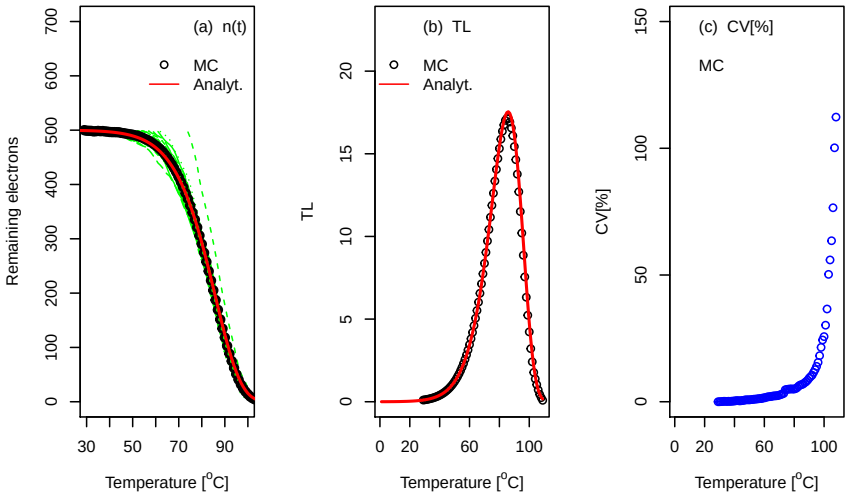

Fig. 10.3: Simulation of stochastic first order TL process using a KMC method. The parameters are n_0 =500 trapped electrons at time $t = 0$, maximum temperature $T = 110$°C, frequency factor $s = 10^{13}$ s^{-1}, energy $E = 1$ eV, heating rate $\beta = 1$ K/s. (a) Plot of the $M = 100$ MC runs, (b) Plot of the average MC $< n(t) >$, (c) Plot of the coefficient of variation CV[%]. The solid lines represent the analytical solutions for the TL deterministic process.

Code 10.4: Microscopic description of quantum tunneling

```
# Original Mathematica program by Vasilis Pagonis
# R version written by Johannes Friedrich, 2018
# The code reproduces Fig 2 of Pagonis and Kulp (2010)
rm(list = ls(all = TRUE)) # empties the environment
library("plot3D")
library("FNN")
## Define Parameters ----
sideX <- 200e-9 # lenght of quader in m
sideX_nm <- sideX*1e9 # length of quader in nm
s_tun <- 3e15
alpha <- 4e9
N_pts <- 100
clusters <- 50
rho_prime <- 1e-5

N_centers <- as.integer(rho_prime * sideX^3 *3 *alpha^3/(4*pi))
rho <- N_centers/ sideX^3
all_times_matrix <- matrix(NA,nrow = N_pts,ncol = clusters)
## Run MC -----
for(j in 1:clusters) {
  if(j %% 100 == 0) print(j)
  n_pts <- N_pts
  n_centers <- N_centers
  xyz_traps <- data.frame(
    x = sample(1:sideX_nm, N_pts, replace = TRUE),
    y = sample(1:sideX_nm, N_pts, replace = TRUE),
    z = sample(1:sideX_nm, N_pts, replace = TRUE)
  )
  xyz_centers <- data.frame(
    x = sample(1:sideX_nm, n_centers, replace = TRUE),
    y = sample(1:sideX_nm, n_centers, replace = TRUE),
    z = sample(1:sideX_nm, n_centers, replace = TRUE)
  )
    ### r calc distances ------
  for(i in 1:(n_pts)){
    ## find next neighbours with package FNN
    dist <- FNN::get.knnx(data = as.matrix(xyz_centers),
                          query = as.matrix(xyz_traps),
                          k = 1)
    all_dist <- as.data.frame(dist)
    P <- runif(n = length(all_dist$nn.dist), min = 0, max = 1)
    # P <- runif(n = 1, min = 0, max = 1)
    recomb_time <- - s_tun^(-1) * exp(alpha * all_dist$nn.dist *
    1e-9) * log(1-P)
```

```
    e_remove <- which.min(recomb_time)
    h_remove <- all_dist$nn.index[e_remove]
    ##remove index from data.frame
    xyz_centers <- xyz_centers[-h_remove,]
    xyz_traps <- xyz_traps[-e_remove,]
    all_times_matrix[i,j] <- recomb_time[e_remove]
  } # end n_pts loop
  all_times_matrix[,j] <- cumsum(all_times_matrix[,j])
} ## end cluster-loop
all_times_matrix <- log10(all_times_matrix)
### plot results ------
times_avg <- rowMeans(all_times_matrix)
matplot(x = all_times_matrix,
        y = (N_pts-1):0,xlim = c(-10,20), col = "grey",
        ylab = "remaining electrons",
        xlab = "log10(times)",pch = 1)
points(
  x = times_avg,  y = (N_pts-1):0, col = "blue")
sd <- apply(all_times_matrix, 1, sd)
sd_error <- sd/sqrt(N_pts)
## plot error bars
arrows(times_avg-sd_error,
       (N_pts-1):0,
       times_avg+sd_error,
       length=0.05,angle=90, code=3)
t <- 10^seq(-15,20,1)
lines(
  x = log10(t),
  y = N_pts * exp(-rho_prime * log(1.8 * s_tun * t)^3),
col = "red",lwd=3)
legend("topright",bty="n",
        legend = c("500 MC runs", "MC average",
        "analytical solution"),
        col = c("grey", "blue", "red"),
        pch = c(1,1,NA),lwd = c(NA,NA,2))
```

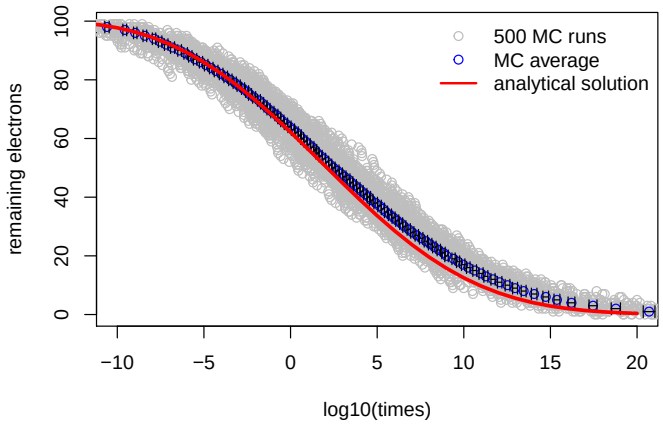

Fig. 10.4: Simulation of microscopic description of ground state quantum tunneling in luminescent materials with the parameters given in the text. The gray area indicates the results from M =500 simulations of the same system, and the solid circles indicate the average of the 500 runs. The standard error of the 500 runs is about equal to the drawing size of individual circles. The solid line represents the analytical Eq.(**??**). For more details see Pagonis and Kulp [42].