DTU Compute

02425
November 17, 2022
UHT/uht

Exercise 11: The generator

Following section 11.2.3, consider the linear stochastic differential equation

$$dX_t = -X_t\, dt + \sqrt{2}\, dB_t, \quad X_0 = x$$

evolving on the interval $X_t \in (-l, l)$ Let $\tau = \inf\{t : |X_t| \geq l\}$ denote the first time of exit from the interval. We aim to find $\mathbf{E}^{X_0=x}\tau$. Moreover, we want to find the probability of exiting to the right, i.e. $\mathbb{P}^{X_0=x}(X_\tau = 1)$

Question 1 **Monte Carlo simulation:** Estimate $\mathbb{P}^{X_0=x}(X_\tau = l)$ and $\mathbf{E}^{X_0=x}\tau$, for $x = l/2$, by simulating $N = 1,000$ realizations of X_t until τ. Take $l = 1$ and use a sufficiently fine time step. Repeat for $l = 2$ and even larger values of l.

Solution: We start by setting up the model. In this solution, we only show results with one value of l, viz. 2, not to clutter the solution with too many plots.

```
require(SDEtools)
require(bvpSolve)
require(MASS)

f <- function(x) -x
g <- function(x) sqrt(2)

x0 <- 0.5
l <- 2

## "Reasonable" time step
dt <- 1e-3
```

The following function simulates until the stopping time:

```
## Simulate with the Euler method until exit from domain
sim <- function(dummy)
{
    x <- x0
    t <- 0
    r <- 0
    while(abs(x)<l)
    {
        t <- t + dt
        x <- x + f(x)*dt + g(x)* rnorm(1,sd=sqrt(dt))
        r <- r + x^2 *dt
    }

    return(c(t=t,x=x,r=r))
}
```

We can now run a number of samples and estimate the probability of exit to the right as well as the time to exit:

```
## Obtain N samples
N <- 1000
res <- sapply(1:N,sim)

print(paste("Mean time to exit: ",mean(res[1,])))


## [1] "Mean time to exit:  4.45387099999992"


print(paste("Probability of exit to the right: ",mean(res[2,]>0)))


## [1] "Probability of exit to the right:  0.544"
```

Question 2 **Finding $\mathbb{P}^{X_0=x}(X_\tau = l)$ using a backward equation:** Write a boundary value problem which governs $h(x) := \mathbb{P}^{X_0=x}(X_\tau = l)$. Solve the boundary value problem using whichever method you prefer. Plot the solution for the values of l you used in the previous and compare the results. *Hint:* Applicable methods are: Analytical solution, e.g. in terms of the scale function. Numerical solution using a built-in solver of boundary value problems (e.g. `bvp4c` in `Matlab`, `bvpSolve` in `R`). Numerical solution using the generator obtained from `fvade` - in that case, use boundary condition `'e'` to "extend" the generator and include the absorbing boundary points.

Solution: The BVP governing the probability of exit is

$$Lh = V'f + \frac{1}{2}g^2V'' = -xV'(x) + V'' = 0$$

with boundary conditions $h(-l) = 0$, $h(l) = 1$. Let us first determine the scale function s. Its derivative $\phi = s'$ is given by the ODE

$$\phi'(x) = x\phi$$

so

$$\phi(x) = \phi(0)e^{x^2/2}$$

We can, arbitrarily, set $\phi(0) = 1$ to get

$$s(x) = \int_0^x e^{y^2/2}\, dy.$$

This can be written in terms of the imaginary error function, erfi:

$$s(x) = \sqrt{2}\text{erfi}(x/\sqrt{2})$$

although this is not overly useful.

The following code illustrates multiple ways of computing the result. We first solve it numerically as a boundary value problem.

```
## Solving the BVP with bvpSolve
fun <- function(x,y,p)
    list(c(y[2],-2/g(x)/g(x)*f(x)*y[2]))

x <- seq(-l,l,length=201)
sol <- bvpshoot(yini = c(0,NA),yend=c(1,NA),x=x,func=fun,guess=0)
```

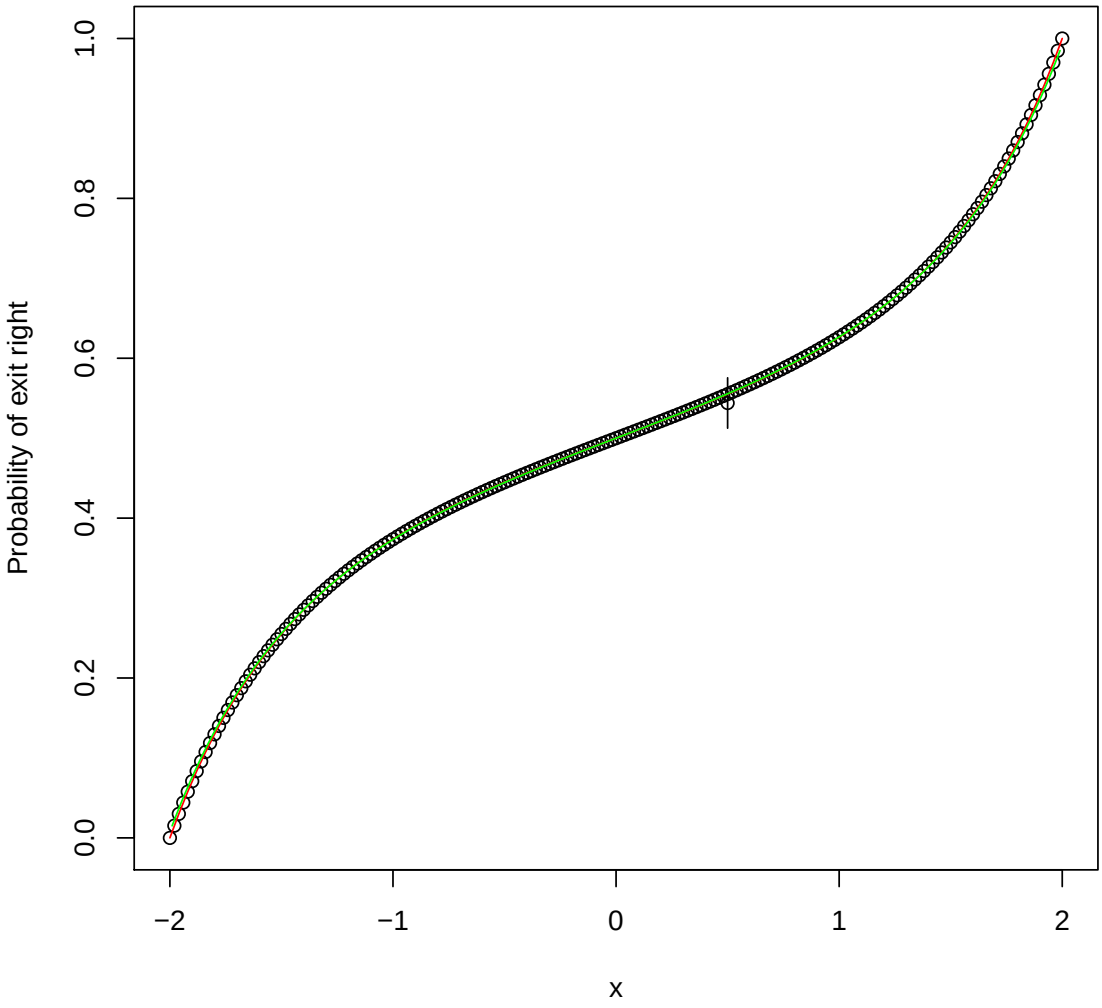
The next computes the scale function with numerical integration, and fixes the constant to match the boundary conditions.

```
## Compute the solution from the scale function, using numerical integration
phi <- function(x) exp(integrate(function(y)-2*f(y)/g(y)/g(y),lower=0,upper=x)$value)
phiv <- function(x) sapply(x,phi)  # Vectorize
s <- function(x) integrate(phiv,lower=0,upper=x)$value
sv <- function(x) sapply(x,s)      # Vectorize

sm <- s(-l)  # Boundary points
sp <- s(+l)

ss <- function(x) (sv(x)-sm)/(sp-sm)   # Scaled scale function (sic)
```

```
## Discretize the generator and find its null space
xi <- seq(-l,l,length=201)
xc <- xi[-1] - 0.5*diff(xi)
G <- fvade(f,function(x)0.5*g(x)*g(x),xi,'e')


## Loading required package:  Matrix


sG <- Null(t(G))

coefs <- solve(sG[c(1,nrow(sG)),],c(0,1))

h <- sG %*% coefs
```

We can now plot all the solution to compare

```
## bvpsolve
plot(sol[,1],sol[,2],xlab="x",ylab="Probability of exit right")

## Monte carlo
points(x0,mean(res[2,]>0))
lines(c(x0,x0),mean(res[2,]>0)+c(-2,2)*sqrt(var(res[2,]>0)/ncol(res)))

## Scale function by numerical integration
plot(ss,from=-l,to=l,add=TRUE,col="red")

## With the generator
lines(xc,h[2:(length(h)-1)],col="green")
```



We see that they all agree.

For small values of l, the solution approaches a straight line connecting the two endpoints. As the domain is increased, a plateau emerges around $x = 0$ where h varies less and is roughly equal to $1/2$; in stead, the variation in h concentrates at the boundaries. In this situation, the drift is likely to take the process to the equilibrium value of $x = 0$ where it will remain a long time until a random fluctuation takes it to one of the boundaries. Only when the initial condition is near the boundary does it have an effect, namely through the event that the noise takes the particle to the boundary *before* it relaxes to the origin.

Question 3 **Finding $\mathbf{E}\tau$ using a backward equation:** Write a boundary value problem which governs $k(x) := \mathbf{E}^{X_0=x}\tau$. Solve the problem, plot the solution, and compare with the Monte Carlo solution as well as with the analytical result (The numerical computations become increasingly demanding (and/or inaccurate) when l is increased; both the Monte Carlo simulations and the numerical solution. Explain why. *Hint:* The same methods apply. If using `fvade`, it is slightly easier to use absorbing boundary conditions (`'a'`).

Solution: Now the boundary value problem is (compare the notes)

$$Lk + 1 = -xk'(x) + k'' + 1 = 0, \quad k(-l) = k(l) = 0.$$

The notes give an analytical solution in terms of an integral that must be evaluated nunerically. Here, we solve it numerically as a BVP:

```
## Expected time to exit
fun <- function(x,y,p)
    list(c(y[2],-2/g(x)/g(x)*(f(x)*y[2]+1)))

x <- seq(-l,l,length=201)
sol.bvp <- bvpshoot(yini = c(0,NA),yend=c(0,NA),x=x,func=fun,guess=0)
```

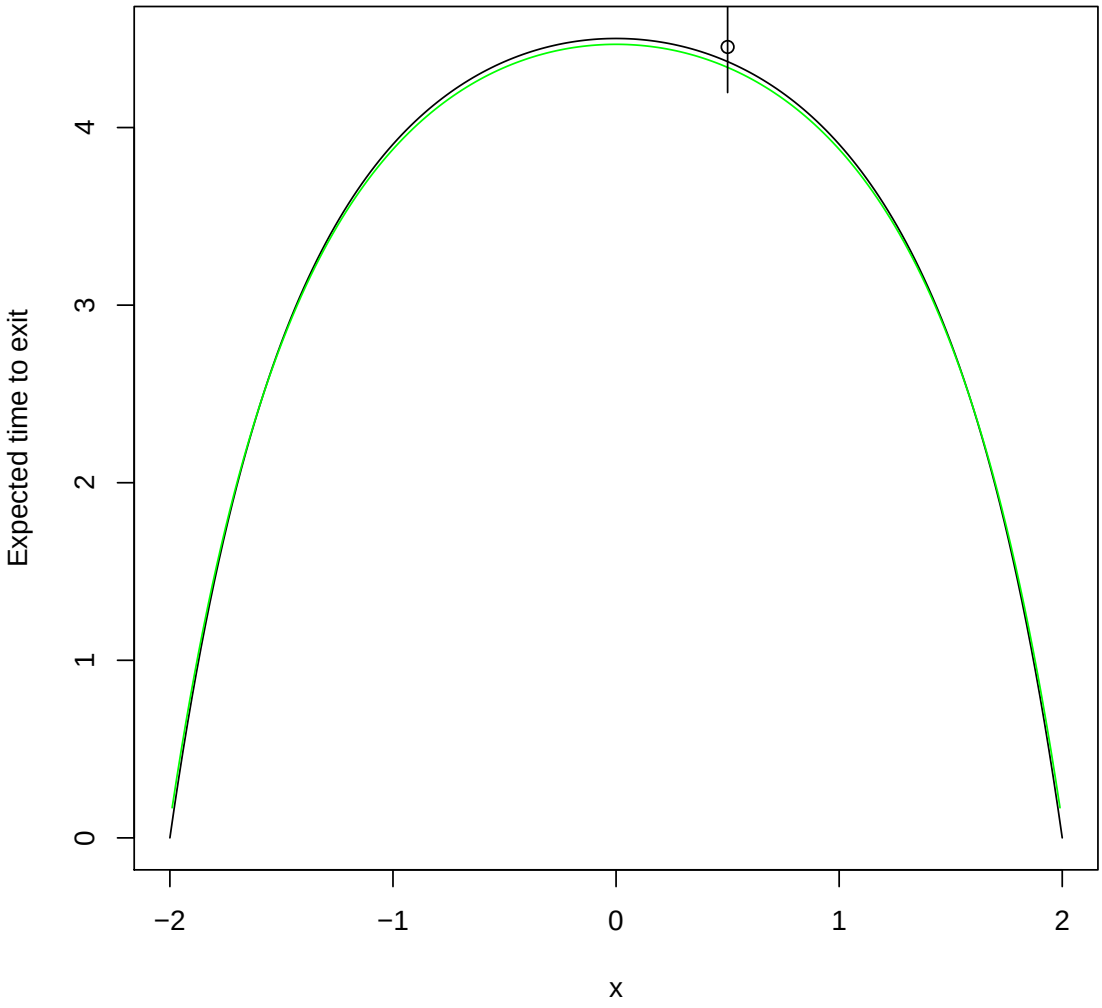
We can also use symmetry to convert it to an initial value problem: We know that $h'(0) = 0$ and that we can add any constant to a solution and the differential equation itself will still be satisfied. So we can solve, say, the problem on the interval $x \in [0, 1]$, first using initial conditions $k(0) = 0$, $k'(0) = 0$. Then, with that solution in hand, we can take care of symmetry and the boundary condition by setting $k(x) := k(|x|) - k(1)$.

We can also use the discretized generator:

```
G <- fvade(f,function(x)0.5*g(x)*g(x),xi,'a')
k.fvade <- solve(G,rep(-1,nrow(G)))
```

We plot the solutions for comparison:

```
## Plot bvp solution
plot(sol.bvp[,1],sol.bvp[,2],type="l",xlab="x",ylab="Expected time to exit")

## Add solution from generator
lines(xc,k.fvade,col="green")

## Add Monte Carlo; add confidence interval
points(x0,mean(res[1,]))
lines(rep(x0,2),mean(res[1,]) + c(2,-2)*sqrt(var(res[1,])/N))
```



Also here, when the domain is small so that the noise is dominating, we recover approximately the quadratic scaling from space to time which characterizes pure diffusion. When the noise level is low, we get a plateau around $x = 0$ where the time to absorption varies less with the initial condition; the reasoning is the same as in the previous question.

Question 4 **Finding the expected total pay-off:** Find, as a function of x, the expected total pay-off

$$\mathbf{E}^{X_0=x} \int_0^\tau X_t^2 \, dt$$

Extend the Monte Carlo simulation to compute also this payoff and compare with the numerical solution.

Solution: In this case the boundary value problem is

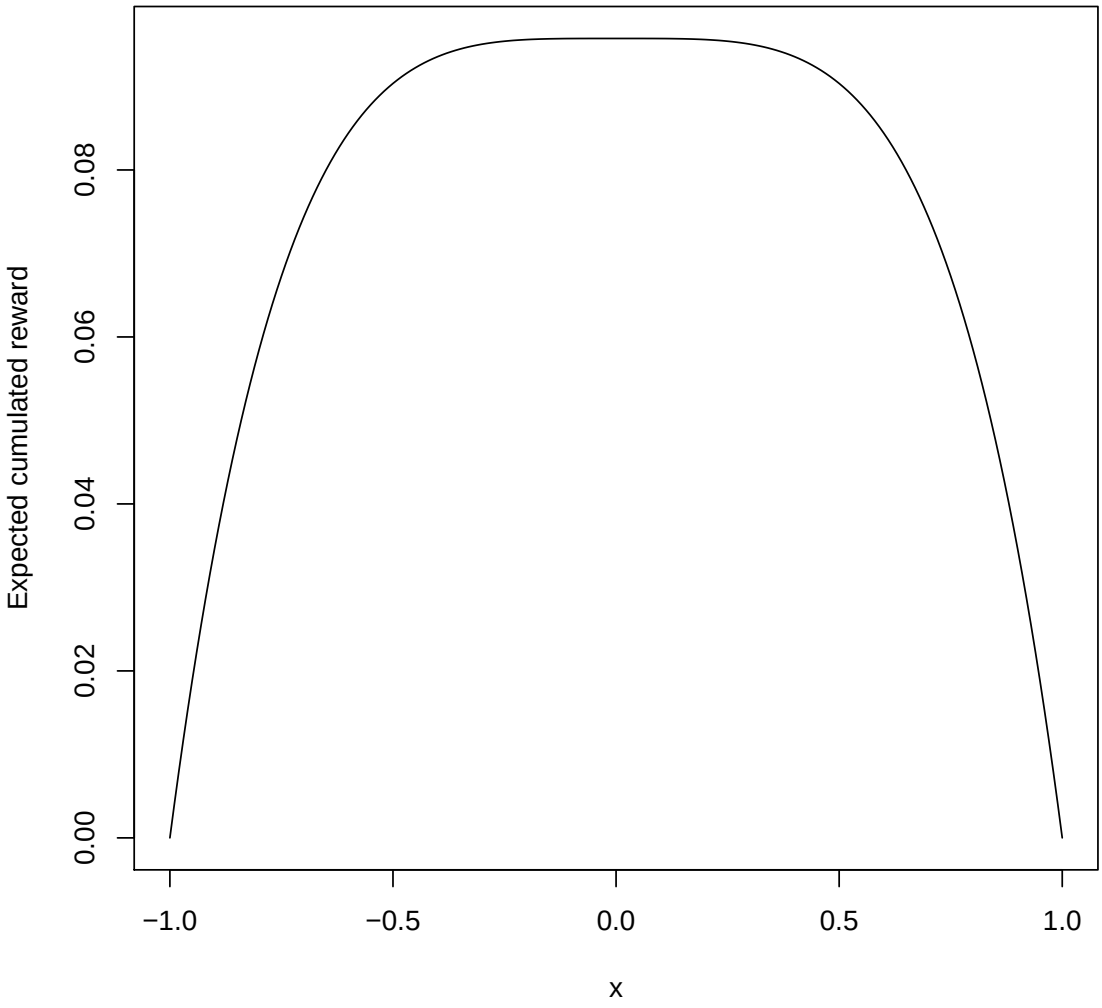
$$(Lk)(x) + x^2 = 0, \quad k(-1) = k(1) = 0.$$

We solve as a BVP, using the discretized generator, and with Monte Carlo:

```
## Cumulated reward (x squared)
fun <- function(x,y,p)
    list(c(y[2],-2/g(x)/g(x)*(f(x)*y[2]+x^2)))

x <- seq(-1,1,0.01)
sol.bvp <- bvpshoot(yini = c(0,NA),yend=c(0,NA),x=x,func=fun,guess=0)


## Compute solution from generator
G <- fvade(f,function(x)0.5*g(x)*g(x),xi,'a')
k.fvade <- solve(G,-xc^2)
```

We plot the solutions:

```
plot(sol.bvp[,1],sol.bvp[,2],type="l",xlab="x",ylab="Expected cumulated reward")
lines(xc,k.fvade,col="green")

## Add Monte Carlo; add confidence interval
points(x0,mean(res[3,]))
lines(rep(x0,2),mean(res[3,]) + c(2,-2)*sqrt(var(res[3,])/N))
```



Again, we see the good agreement and the plateau around $x = 0$. In this case the plateau is more pronounced, because the running reward x^2 vanishes at the equilibrium, so $k''(0) = 0$.

Is the origin attainable under the CIR and GBM processes?

Question 5 **The CIR process:** Example 11.4.1 in the notes establishes a criterion under which the Cox-Ingersoll-Ross process may reach the origin. Fill in the details in the argument, thus verifying the criterion.

Solution: The scale function is given by $s' = \phi$ where

$$\phi(x) = \exp\left(\int_{x_0}^{x} \frac{2\lambda(x-\xi)}{\gamma^2 x}\, dx\right) \quad .$$

We can compute this integral explicitly:

$$\phi(x) = \exp\left(\frac{2\lambda}{\gamma^2}(x - x_0) - \frac{2\lambda\xi}{\gamma^2}\log(x/x_0)\right)$$

or

$$\phi(x) = e^{2\lambda(x-x_0)/\gamma^2}\ (x_0/x)^{2\lambda\xi/\gamma^2} \quad .$$

Note that $\phi(x)$ diverges as $x \to 0$ but is continuous elsewhere. The question is if ϕ is integrable over $(0, x_0)$. Since the first term, the exponential, is continuous at 0, ϕ will be integrable if and only if the latter term is integrable. This is the case if and only if

$$2\lambda\xi/\gamma^2 < 1.$$

(In that case, an antiderivative is, up to scale, $x^{1-2\lambda\xi/\gamma^2}$ which is continuous at 0). In conclusion, the origin is attainable if and only if

$$2\lambda\xi < \gamma^2 \quad .$$

Note that this condition only involves the drift $f(0) = \lambda\xi$ at the boundary, and the diffusivity near the boundary, i.e. $D'(0) = \gamma^2/2$.

We see that the boundary is attainable if the drift at the boundary weak, compared to the diffusivity gradient. This is natural, since it is the noise that must push the state on to the boundary and in doing so overcome the drift that pushes the state away.

Question 6 **Geometric Brownian motion:** Example 11.4.2 establishes a criterion for which geometric Brownian motion may converge to the origin. Fill in the details in the argument, thus verifying the criterion.

Solution: The scale function is given by $s' = \phi$ where

$$\phi(x) = \exp\left(\int_{x_0}^{x} \frac{-2rx}{\sigma^2 x^2}\ dx\right) \quad .$$

We can compute this integral explicitly:

$$\phi(x) = \exp\left(\frac{2r}{\sigma^2}\log(x_0/x)\right) = (x_0/x)^{2r/\sigma^2}$$

So, (discarding the constants, because the scale function is only uniquely defined up to shift and scale)

$$s(x) = (x_0/x)^{2r/\sigma^2 - 1}.$$

Here we assume $2r/\sigma^2 \neq 1$. (The case $2r/\sigma^2 = 1$ must be trated separately; there we find that the scale function is a log function which diverges at $x = 0$). We see that the scale function is continuous at 0 if and only if $2r/\sigma^2 - 1 < 00$. Hence the origin is attainable if and only if

$$2r < \sigma^2.$$

In words, the origin is attainable if the drift is towards the origin, or if the drift is away from the origin but weaker than the noise. Keep this in mind next week when we talk about stability.