DS 290: MODELLING AND SIMULATION

This demo contains implementations of certain topics discussed in this course.

Random Number Generation

Linear congruential generator

$$X_{i+1} = (\mathrm{a}\mathrm{X}_i + c) \bmod m \;\; i = 0, 1, 2....$$

$c \neq 0$ mixed congruential generator
$c = 0$ multiplicative congruential method

Here we choose to test the two random number generation schemes by their maximal period.

Case 1: mixed congruential generator

```
clear all;
x0 = 27;
a = 17;
c = 43;
m = 100;
% expects random numbers between 0 and 99
r = [];
n =  5 ; % number of random numbers generated
p = x0;
for i=1:1:n
    X(i) = mod(a*p + c, m);
    r = [r, X(i)];
    p = X(i);
end
r
```

```
r = 1×5
     2    77    52    27     2
```

Case 2: Multiplicative congruential method

```
clear all;
a = 13 ; m = 64; x0 = 1;
n = 100; % number of random numbers
p = x0;
r = [];
for i=1:1:n
    X(i) = mod(a*p , m);
    if any(r == X(i))
        period = i-1
        break
    end
    r = [r, X(i)];
    p = X(i);
end
```

```
period = 16
```

```
r
```

```
r = 1×16
    13    41    21    17    29    57    37    33    45     9    53    49    61 ···
```

Appropriate choice of a, X_0 and c must be used. Refer [Learmonth and Lewis, 1973; Lewis et al., 1969]

Random number generation using matlabs inbuilt function

```
clear all;
n = 100;
r = [];
for i=1:1:n
    a = randi([0,63]);
    if any(r == a)
        period = i -1
        break
    end
    r = [r, a];
end
```

```
period = 4
```

Test for random numbers:

check for uniformity:

Null hypothesis : $H_0 : R_i \sim \text{Uniform}[0, 1]$

Alternate hypothesis: $H_1 : R_i \nsim \text{Uniform}[0, 1]$

frequency test: Here we demonstrate the kolmogorov-smirnov test

Does the test pass for n=100, 500, 1000 given below? Try changing the value of n and observe for yourself.

```
% The value of alpha taken is 0.05
n = 50; % number of random variables
a = 17 ; m = 2^(3) ; x0 = 27;
c = 43;
p = x0;
r = [];
Dp = zeros(n,1);
Dm = zeros(n,1);
for i=1:1:n
    X(i) = (mod(a*p+c , m))/m;
    r = [r, X(i)];
    p = X(i);
end
r = sort(r);
for i=1:1:n
    Dp(i) = (i/n) - r(i);
    Dm(i) = r(i) - ((i-1)/n);
```

```
end
D = max(max(Dp), max(Dm));
Dcritic = 1.36/sqrt(n);
if D <= Dcritic
    disp('uniform distribution')
else
    disp('null hypothesis is rejected')
end
```

```
uniform distribution
```

```
r
```

```
r = 1×50
    0.0048    0.0059    0.0419    0.0463    0.0677    0.0807    0.0889    0.1090 ···
```

Now lets conduct the same test for matlabs random number generator

```
% The value of alpha taken is 0.05
n = 100000; % number of random variables
r = [];
Dp = zeros(n,1);
Dm = zeros(n,1);
for i=1:1:n
    X(i) = rand();
    r = [r, X(i)];
end
r = sort(r);
for i=1:1:n
    Dp(i) = (i/n) - r(i);
    Dm(i) = r(i) - ((i-1)/n);
end
D = max(max(Dp), max(Dm));
Dcritic = 1.36/sqrt(n);
if D <= Dcritic
    disp('uniform distribution')
else
    disp('null hypothesis is rejected')
end
```

```
uniform distribution
```

check for independence:

autocorrelation test:

Null hypothesis : $H_0 : R_i \sim \text{Independently}$

Alternate hypothesis: $H_1 : R_i \nsim \text{Independently}$

For testing large sequence of random numbers (denoted by M), we denote the test statisitc as

$$z_0 = \frac{\widehat{\rho}_{\mathrm{il}}}{\sigma_{\widehat{\rho}_{\mathrm{il}}}}$$

where $\widehat{\rho}_{\mathrm{il}} = \frac{1}{M+1}\left[\sum_{k=0}^{M} R_{i+\mathrm{kl}} R_{i+(k+1)l}\right] - 0.25$

$$\sigma_{\widehat{\rho}_{\mathrm{il}}} = \frac{\sqrt{13M+7}}{12(M+1)}$$

if z_0 satisfies the bounds donot reject the null hypothesis $-z_{-\frac{\alpha}{2}} \le z_0 \le z_{\frac{\alpha}{2}}$

Does the test pass for n=100, 500, 1000 given below? Try changing the value of n and observe for yourself.

```
%[Learmonth and Lewis, 1973; Lewis et al., 1969]
% The value of alpha taken is 0.05
n = 50; % number of random variables
M = n-2;
a = 17 ; m = 2^(3) ; x0 = 27;
c = 43;
p = x0;
r = [];
Dp = zeros(n,1);
Dm = zeros(n,1);
for i=1:1:n
    X(i) = (mod(a*p+c , m))/m;
    r = [r, X(i)];
    p = X(i);
end
s = 0;
for k=1:1:M
    s = s+ r(k)*r(k+1);
end
s = s/(M+1) - 0.25;
rho = s;
sigma = sqrt(13*M+7)/(12*(M+1));
z0 = rho/sigma;
zcritic = 1.96;
if (z0 >= -zcritic) && (z0 <= zcritic)
    disp('null hypothesis is not rejected')
else
    disp('null hypothesis is rejected')
end
```

```
null hypothesis is not rejected
```

Now lets conduct the test for the matlabs random number generator.

```
% The value of alpha taken is 0.05
n = 100000; % number of random variables
```

```
M = n-2;
r = [];
Dp = zeros(n,1);
Dm = zeros(n,1);
for i=1:1:n
    X(i) = rand();
    r = [r, X(i)];
end
s = 0;
for k= 1:1:M
    s = s + r(k)*r(k+1);
end
s = s/(M+1) - 0.25;
rho = s;
sigma = sqrt(13*M+7)/(12*(M+1));
z0 = rho/sigma;
Zcritic = 1.96;
if (z0 >= -zcritic) && (z0 <= zcritic)
    disp('null hypothesis is not rejected')
else
    disp('null hypothesis is rejected')
end
```

```
null hypothesis is not rejected
```

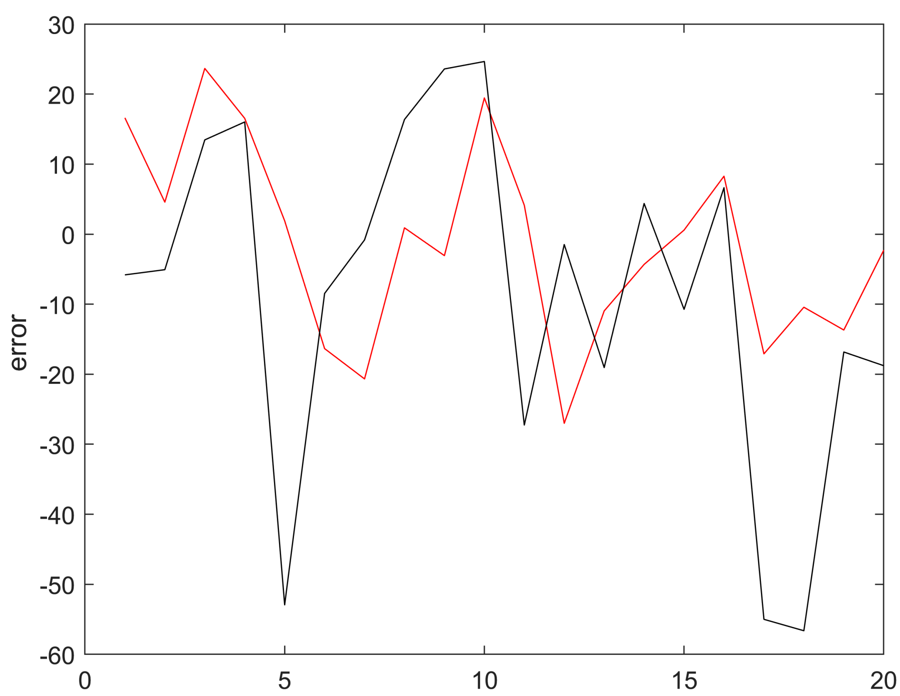
Input modelling

Pageno:381 (Jerry Banks)

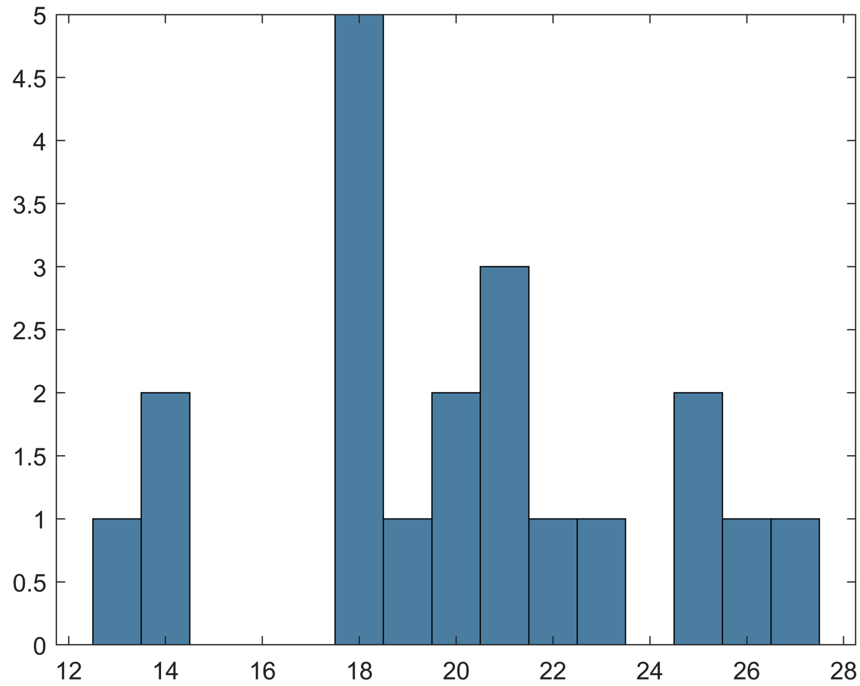
The number of patrons staying at a small hotel in successive nights was observed to be 20,14,21,19,14,18,21,25,27,26,22,18,13,18,18,18,25,23,20,21. Fit both an AR(1) and EAR(1) model to the data. Tell which model could provide a better fit by looking at the histogram.

```
%====================================
%       AR(1) MODEL
%====================================
a = [20,14,21,19,14,18,21,25,27,26,22,18,13,18,18,18,25,23,20,21];
xbar = mean(a);
vbar = var(a);
s = 0;
sizev = size(a,2);
for i=1:1:sizev-1
    s = s + a(i)*a(i+1);
end
covar = (1/sizev)*(s - (sizev - 1)*xbar^(2));
phi = covar/vbar;
mu = xbar ; sigmae = vbar^(2)*(1 - phi^(2));
%================================================
ar = zeros(sizev,1);
t = 1;
ar(t) = sqrt(sigmae/(1-phi^(2)))*randn()+ mu;
t = t + 1;
while(t <= sizev)
    epst = sqrt(sigmae)*randn() ;
```

```
    ar(t) = mu + phi*(ar(t-1) - mu) + epst;
    t = t + 1 ;
end
%=================================================
%           EAR(1)  MODEL
%================================================
PHI = phi ;
LAMBDA = (1/xbar);
ear = zeros(sizev,1);
t = 1;
ear(t) = exprnd(1/LAMBDA);
t = t + 1;
while(t <= sizev)
    u = rand();
    if u <= phi
        ear(t) = phi*(ear(t-1));
    else
        ear(t) = phi*(ear(t-1)) + exprnd(1/LAMBDA);
    end
    t = t + 1 ;
end
%=======================================
%plottting the errors of input modelling
errorar = a' - ar;
errorear = a' - ear;
point = 1:1:sizev;
close all;
plot(point , errorar, '-r'),ylabel('error '),hold on;
plot(point , errorear, '-k')
```



```
figure(2)
histogram(a)
```



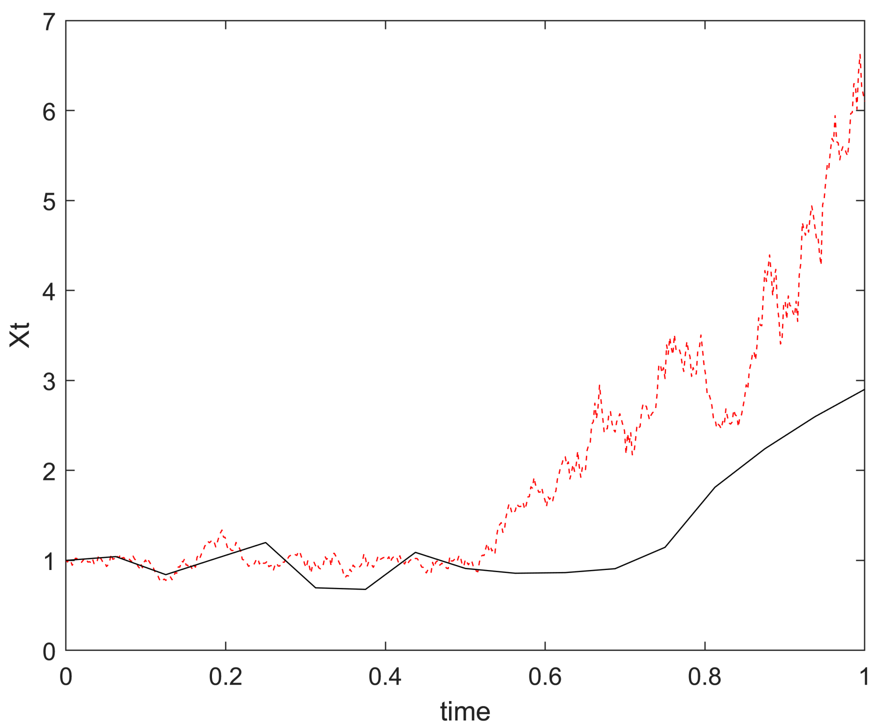
Numerical solution of stochastic differential equations

considering the sde $\mathrm{dX}_t = a\,X_t\,\mathrm{dt} + b\,X_t\,\mathrm{dW}_t$

The euler scheme $Y_{n+1} = Y_n + \mathrm{a}\mathrm{Y}_n\,\Delta_n + b\,Y_n\,\Delta W_n$

```
% Here Y0 = 1.0, a = 1.5, b = 1.0. T = 1
delta = 2^(-4);
%========================================
%   Exact Solution
%========================================
x0 = 1.0;
b = 1.0;
a = 1.5;
t = 0:delta:1;
delta2 = 2^(-9);
t2 = 0:delta2:1;
Xt = zeros(size(t2,2),1);
delw = sqrt(delta2)*randn(size(t2,2)-1,1);
wt = zeros(size(t2,2),1);
for i=2:1:size(t2,2)
    wt(i) = wt(i-1) + delw(i-1);
end
Xt(1) = x0;
for j = 2:1:size(t2,2)
    Xt(j) = x0*exp((a - 0.5*b^(2))*t2(j) + b*wt(j));
end
```

```
%========================================
%    Euler maruyama method
%========================================
delw = sqrt(delta)*randn(size(t,2)-1,1);
Y0 = 1;
Y = zeros(size(t,2),1);
Y(1) = 1.0;
for j=2:1:size(t,2)
    Y(j) = Y(j-1) + a*Y(j-1)*delta + b*Y(j-1)*delw(j-1);
end
plot(t2,Xt, '--r'), hold on;
plot(t,Y,'-k')
xlabel('time'); ylabel('Xt')
```



Demonstartion of strong convergence of euler-maruyama scheme by finding the error estimate

Here the number of batches taken M = 20 and number of sample paths = 100

```
clear all;
%==============================================================
M = 20; % denote the number of batches
N = 100; % denotes the number of sample paths
Del = [2^(-3),2^(-4),2^(-5),2^(-6)];
errorestimate = zeros(size(Del,2),1);
c = 1;
x0 = 1.0;
b = 1.0;
a = 1.5;
for delta = Del
    delta
    su = 0;
```

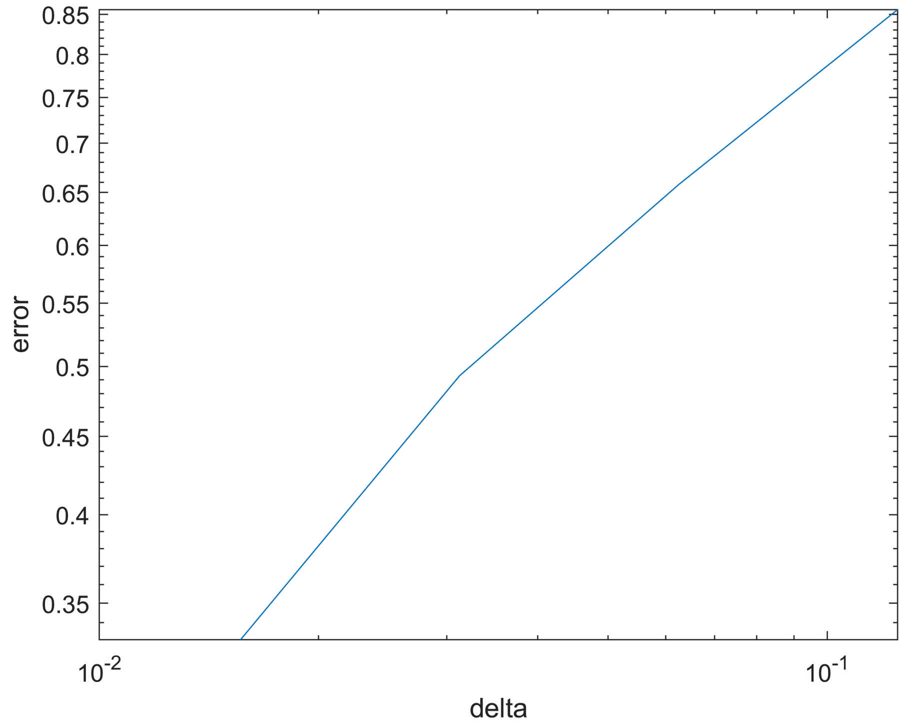
```
    for m = 1:1:M
        t = 0:delta:1;
        Xt = zeros(size(t,2),1);
        wt = zeros(size(t,2),1);
        Xt(1) = x0;
        for p=1:1:N
            delw = sqrt(delta)*randn(size(t,2)-1,1);
            for i=2:1:size(t,2)
                wt(i) = wt(i-1) + delw(i-1);
            end

            for j = 2:1:size(t,2)
                Xt(j) = x0*exp((a - 0.5*b^(2))*t(j) + b*wt(j));
            end

            Y0 = 1;
            Y = zeros(size(t,2),1);
            Y(1) = 1.0;
            for j=2:1:size(t,2)
                Y(j) = Y(j-1) + a*Y(j-1)*delta + b*Y(j-1)*delw(j-1);
            end
            su = su + abs(Y(end) - Xt(end));
        end
    end
    errorestimate(c) = su/(M*N);
    c = c + 1;
end
```

```
delta = 0.1250
delta = 0.0625
delta = 0.0313
delta = 0.0156
```

```
errorestimate
```

```
errorestimate = 4×1
    0.8562
    0.6577
    0.4931
    0.3312
```

```
figure(4)
loglog(Del,errorestimate)
ylabel('error')
xlabel('delta')
```



0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
error
10^{-2}
10^{-1}
delta