Statistics 572
Homework 6

Alejandro Robles

November 15th, 2018

1 In-Class

Generate random variable of size 200 from $Gamma(3, 2)$ using the Monte Carlo integration, estimate $E[g(x)]$ and $Var[g(x)]$, where

1. $g(x) = \sqrt{x}$
2. $g(x) = 20\%$ (Trimmed mean, mean of mid 80% of sorted data)
3. $g(x) = Q_3(x)$ (Third Quantile)

For (2) and (3) use Monte Carlo simulation.

MATLAB Code:

```
% Parameters
n = 200; alpha = 3; beta = 2;

% data
x = gamrnd(alpha,beta, 1, n);
x2 = gamrnd(alpha,beta, n, n);

% Part A UDF Definition found at the end of homework
[mu, variance] = mc_integral(x, @sqrt, 'Square Root');

% Part B
[mu2, variance2] = mc_integral(x2, @(x) (trimmean(x,0.2)), 'Trimmed Mean');

% Part C
[mu3, variance3] = mc_integral(x2, @(x) (quantile(x,0.75)), 'Third Quartile');
```

Result:

```
Estimated Expected and Variance for g(x) = Square Root : (2.37437, 0.490211)
Estimated Expected and Variance for g(x) = Trimmed Mean : (5.98747, 0.0610785)
Estimated Expected and Variance for g(x) = Third Quartile : (7.8474, 0.139685)
```

Discussion: The approximation can be made as accurate as needed by increasing n. Because we are assuming i.i.d the variance of the estimate should be more accurate.

2 In-Class

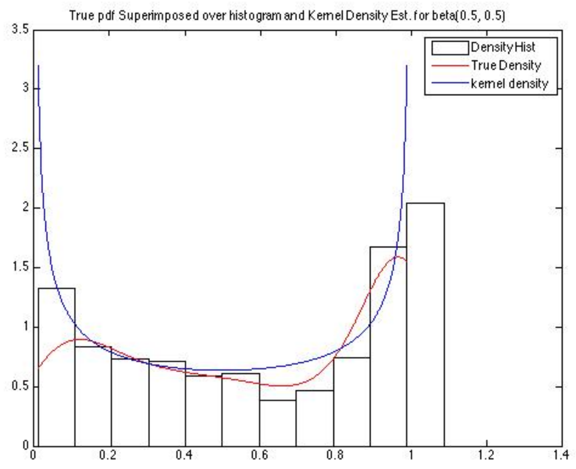
Build up the MC whose invariant distribution is $B(\alpha, \beta)$.

1. Generate random sample of size n using using M-H (Burning in 5%). Use proposal $U(X_t{-}0.5, X_t{+}0.5)$
2. Kernel Density estimate for random sample generated in (1).
3. Plot density histogram for random sample in (1), kernel density, and true density in the same graph.
4. Do Monte Carlo simulation to estimate ISE of estimate in (2) and MSE at x = 0.2, 0.5, 0.8.

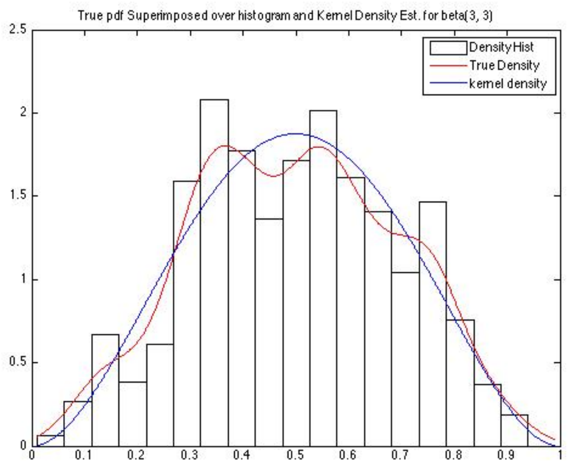
MATLAB Code:

```
% Parameters
params = [3, 0.5];
burnin = 0.05;
n=1000; m=100;
x0 = [0.2 0.5 , 0.8];
k = length(x0);
counter = 1;

for param = params;
    alpha = param;
    beta  = alpha;

    fhatmatrix=zeros(m,k);
    ISE=zeros(m,k);
    MSE=zeros(m,k);

    figure(counter);
    % ********* Part 1 ********
    [x, mixrate] = mc_mh(alpha, beta, n, burnin);
    % ********* Part 2 & 3 ********
    fhatnorm = kernel_density(x, alpha, beta, true);

    % ******* Part 4: Monte Carlo Simulation ********

    for i=1:m
        [x, mixrate] = mc_mh(alpha, beta, n, burnin);
        [ise, mse, fhatnorm] = mc_ISE(x, x0, alpha, beta);
        fhatmatrix(i,:)= fhatnorm;
        MSE(i,:)=mse;
        ISE(i,:)=ise;
    end

    MSE_hat = mean(MSE);
    ISE_hat = mean(ISE);

    sprintf('For alpha = beta = %g\n, MSE and ISE are ', alpha)
    disp(MSE_hat);
    disp(ISE_hat);
    counter = counter + 1;

end
```

Result:



(a) $\alpha = \beta = 0.5$



(b) $\alpha = \beta = 3$

Figure 1: $B(\alpha, \beta)$

```
For alpha = beta = 3
MSE   0.1660    0.4359    0.1588
ISE   0.7606    0.7606    0.7606

For alpha = beta = 0.5
MSE   0.2921    0.1513    0.2909
ISE   0.7343    0.7343    0.7343
```

Discussion: As we can, for $\alpha = \beta = 3$, the Kernel Density Estimate fits the the true density more closely. In Figure (a), we see that within tails the kernel density fits nicely but towards the tails it doesn't. Maybe this could be fixed by playing with the burning rate.

The ISE estimates are very similar for both set of parameters but the MSE is on average slightly smaller with $\alpha = 0.5$.

User Defined Functions:

@mc integral

```
function [mu, variance] = mc_integral(data, statistic, statname)
% ***** E(x) *****
g_x = statistic(data);
mu = mean(g_x);

% ***** E(x^2)*****
g_x2 = statistic(data).^2;
e_x2 = mean(g_x2);

% ***** Var(x) *****
variance = e_x2 - mu.^2;

sprintf('Estimated Expected and Variance for g(x) = %s : (%g, %g) \n', statname, mu, variance)
```

@mc ISE

```
function [ISE, MSE, fhatnorm] = mc_ISE(x, x0, alpha, beta)
n = length(x0);
fx0 = betapdf(x0,alpha,beta);
h1 = 1.06 * n^(-1/5) * std(x0);
fhatnorm = zeros(1,n);

for i = 1:n;
    fnorm = exp(-(1/(2*h1^2))*(x0-x(i)).^2)/sqrt(2*pi)/h1;
    fhatnorm = fhatnorm+fnorm/(n);
end

MSE = (fhatnorm-fx0).^2;
ISE = sum((fhatnorm-fx0).^2);
```

@mc mh

```
function [x, mixrate] = mc_mh(alpha, beta, n, burnin)
x = zeros(1,n);
x(1) = rand(1); ind=0;
for i = 2:n
y = unifrnd(x(i-1)-.5,x(i-1)+.5);
    u = rand(1);
    if y<=0 || y>=1
        x(i)=x(i-1);
        else
            delta = min([1 betapdf(y,alpha,beta)/betapdf(x(i-1),alpha,beta)]);
            if u <= delta
            x(i) = y;
            ind=ind+1;
            else
            x(i) = x(i-1);
        end
    end
end

%burnin
mixrate = ind/n;
n1 = burnin *n;
x=x(n1+1:n);
```

@kernel density

```
function fhatnorm = kernel_density(x, alpha, beta, do_plot)

%kernel density estimate
n=length(x);
xx = linspace(.01,.99, n);
fhatnorm = zeros(size(xx));
h = 1.06*n^(-1/5)*std(x);
for i=1:length(xx)
% get each kernel function evaluated at x
% centered at data
    fnorm=exp(-(1/(2*h^2))*(xx-x(i)).^2)/sqrt(2*pi)/h;
    fhatnorm = fhatnorm+fnorm/(n);
end

if do_plot
%histogram
    t0 = .01;
    tm = .99;
    rng = tm - t0;
    nbin = ceil(rng/h);
    bins = t0:h:(nbin*h+t0);
    % Get the bin counts vk.
    vk = histc(x,bins);
    % Normalize to make it a bona fide density.
    % We do not need the last count in fhat.

    fhat = vk/(n*h);
    fhat(end) = [];
    % To plot this, use bar with the bin centers.
    tm = max(bins);
    bc = (t0+h/2):h:(tm-h/2);
    bar(bc,fhat,1,'w')
    hold on

    plot(xx,fhatnorm, 'r')
    plot(xx,betapdf(xx,alpha,beta), 'b')
    legend('Density Hist', 'True Density','kernel density')
    title(sprintf('True pdf Superimposed over histogram and Kernel Density Est. for beta(%g, %g)', a
    hold off
end
```