Generating a RV with a Specified CDF

Let X be a continuous random variable defined over a sample space $\mathbf{S}$ with a specified CDF $F_X(x)$. MATLAB has two buit-in random number generators `rand.m` and `randn.m` that generate random numbers that are either unifromly distributed or gaussian distributed. Our goal in this section is to come up with an algorithm to generate samples of a random variable that have the specified CDF. For this purpose, let us first consider the random variable $\mathbf{Y}$ defined via

$$\mathbf{Y} = F_X(\mathbf{X}).$$

From the properties of the CDF for a continuous RV we know that the CDF is a monotonically increasing function between 0 and 1,i.e.,

$$x_1 \leq x_2 \Longleftrightarrow F_X(x_1) \leq F_X(x_2), \quad F_X(x) \in [0,1].$$

and from the fact that X is a continuous RV we can show that $F_X(x)$ is a uniformly continuous function and therefore the transformation $F_X(\mathbf{X})$ is a bijective mapping from $\mathbf{R}$ to the interval $[0,1]$. Hence a unique inverse $F_X^{-1}(x)$ for the CDF exists. Using the set equivalence method the CDF of the transformed variable $\mathbf{Y}$ we obtain:

$$F_Y(y) = \Pr(Y \leq y) = \Pr(X \leq F_X^{-1}(y)) = F_X(F_X^{-1}(y)) = y, y \in [0,1].$$

The PDF of the random variable obtained by taking the derivative of the CDF in the above expression is therefore :

$$f_Y(y) = \begin{cases} 1 & y \in [0,1] \\ 0 & \text{otherwise.} \end{cases}$$

This implies that the RV generated from the transformation $\mathbf{Y} = F_X(\mathbf{X})$ is uniform on the interval $[0,1]$, i.e., $Y \sim U([0,1])$. This fact forms the basis for the algorithm to generate samples of a RV with a specified CDF:

- Generate N samples of a random variable $Z \sim [0,1]$ using the function `rand.m`.
- Generate the inverse CDF $F_X^{-1}(x)$ using a built–in MATLAB function or a written function.
- Apply the transformation F_X^{-1} on the samples of Z.

```
%*************************************************************
%        Random variable Generator
%        This mfunction generates random L random numbers
%        that are distributed with a specified distribution
%        and specified parameters.
%        AUTHOR: Balu Santhanam
%        DATE  : 09/10/00
%        FUNCTION SYNOPSIS:
%        x = rndv(L,para,type)
%
%        Parameter Description:
%        L       :   number of samples
%        para    :   parameter vector
%        type    :   string containing the type
%                    of RV needed with the following options
%        'gauss' :   Gaussian with parameters [mu,sigma]
%        'unif'  :   Uniform with parameters : [a,b].
%        'betad' :   Beta with parameters : [m,n]
%        'lapl'  :   Laplacian with parameters : [mu,sigma]
%        'expn'  :   Exponential with parameter lambda.
%*************************************************************
function x = rndv(L,para,type)
if nargin ~= 3
   error('Insufficient input')
end
u = rand(1,L); % Generate uniform RV
switch type
case 'gauss' ; %  Case where a Gaussian is requested
   mu = para(1); sigma = para(2);
   x = norminv(u,mu, sigma);
   fprintf('%s\t%f\t%f\n','Gaussian with parameters',mu,sigma)
case 'unif' ;  %  Case where a uniform distr. is needed
   a = para(1); b = para(2); x = u*(b-a) + a;
   fprintf('%s\t%f\t%f\n','Uniform with parameters',a,b)
case 'betad' ; % Case where a Beta distribution is needed
   m = para(1); n = para(2); x = betainv(u,m,n);
   fprintf('%s\t%f\t%f\n','Beta with parameters',m,n)
case 'lapl' ; % When laplacian distribution is needed
   mu = para(1); sigma = para(2);
   index1 = find(u < 0.5); index2 = find(u >= 0.5);
   x(index1) = log(2*u(index1)); x(index2) = -log(2*(1-u(index2)));
   x = x*sigma + mu;
   fprintf('%s\t%f\t%f\n','Laplacian with parameters:',mu,sigma)
case 'expn' ;  % When a exponential distribution is needed
   lambda = para(1); x = -(1/lambda)*log(1-u);
   fprintf('%s\t%f\n','Exponential with parameter',lambda)
otherwise
   error('Invalid RV type, Reenter the type again')
end;
```

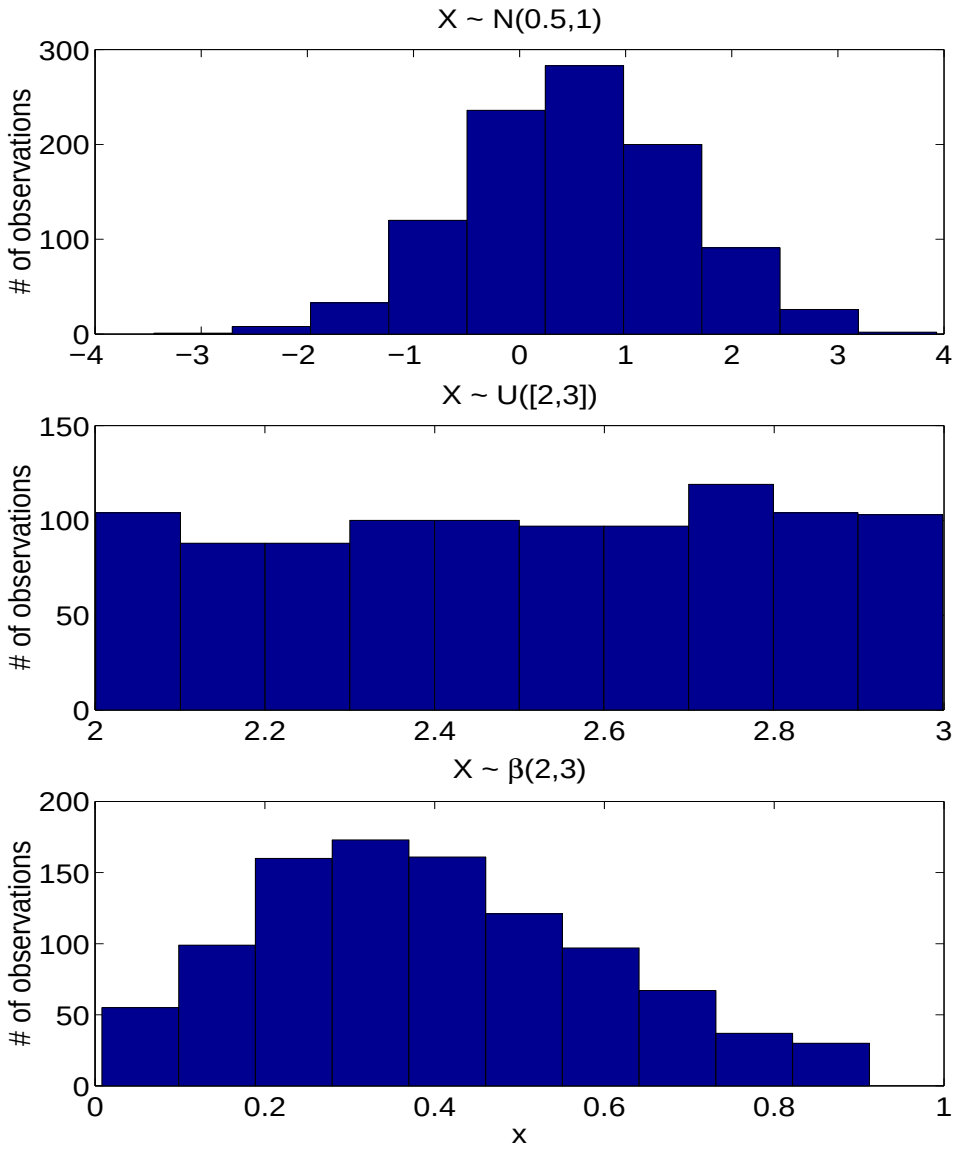

Figure 1: Histograms of output generated via the function `rndv.m` for the cases: (a) 'gauss', (b) 'unif',(c) 'beta'.

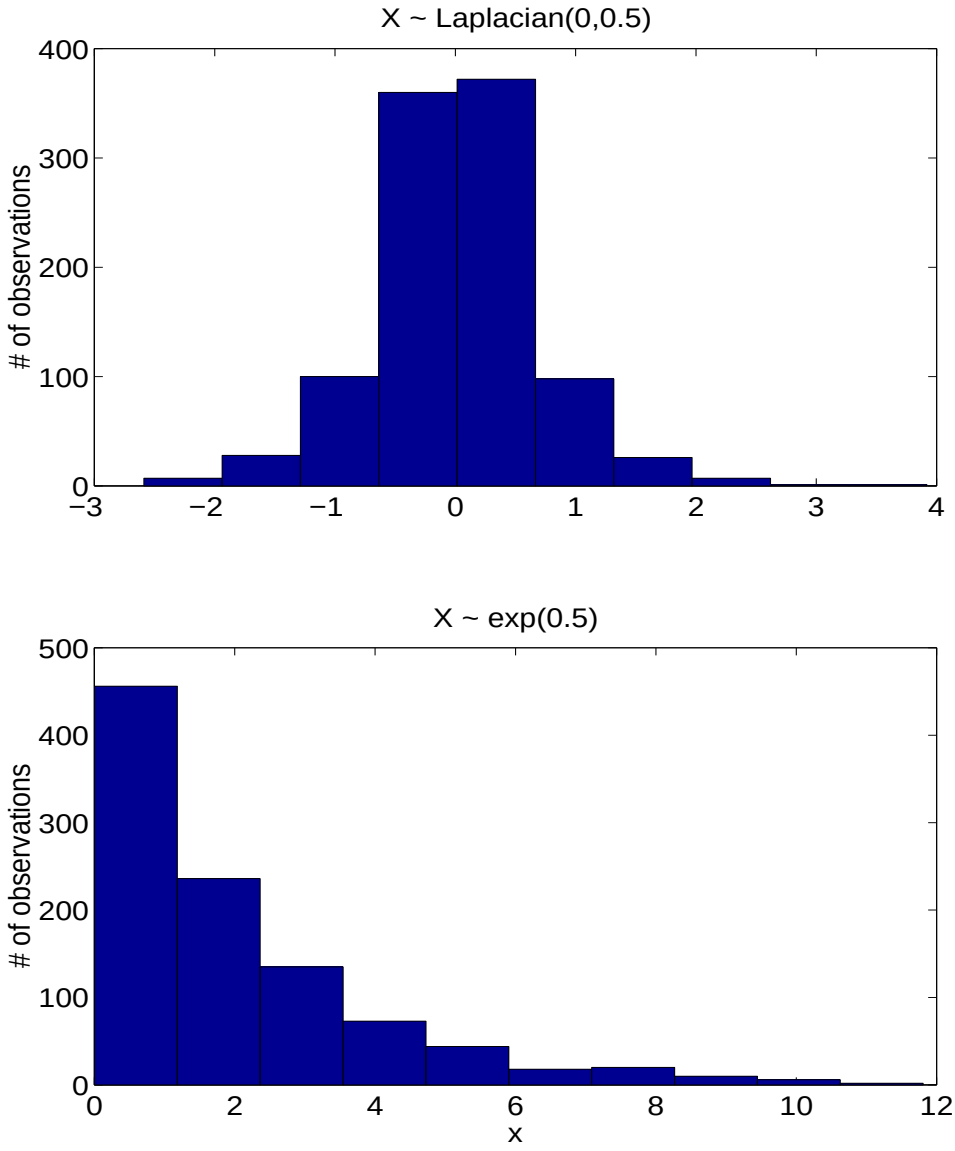

Figure 2: Histograms of output generated via the function `rndv.m` for the cases: (d) 'lapl', (e) 'expn'.