```
function K = MTGP_covPeriodicisoUU_shift(hyp, x, z, i)

% Stationary covariance function for a smooth periodic function, with period p:
%
% Based on the covPeriodicisoUU.m function of the GPML Toolbox -
%   with the following changes:
%       - only elements of x(:,1:end-1)/z(:,1:end-1) will be analyzed,
%       - x(:,end)/z(:,end) will be ignored, as it contains only the label information
%       - independent of the label all x values will have the same hyp
%       - feature scaling hyperparameter is fixed to 1
%       - output scaling hyperparameter is fixed to 1
%       - function considers additional hyperparameter theta_s for features shift
%           (function is limited to 1D features)
%
% The covariance function is parameterized as:
% k(x,y) = exp( -2*sin^2( pi*||(x-theta_s)-(y-theta_s)||/p ) )
%
% where the hyperparameters are:
%
% hyp = [ log(p)]
%           theta_s(1)
%           ...
%           theta_s(nL-1)]
%
% by Robert Duerichen
% 04/02/2014

if nargin<2, K = 'nL'; return; end                  % report number of parameters
if nargin<3, z = []; end                                   % make sure, z exists
xeqz = numel(z)==0; dg = strcmp(z,'diag') && numel(z)>0;        % determine mode

% n = size(x,1);
nL = max(x(:,2));                                  % get number of labels
p   = exp(hyp(1));                                  % period
shift = (hyp(2:end));                              % time shift hyp

%% perform shift
for ii = 2:nL
   x(x(:,2)== ii,1) = x(x(:,2)== ii,1)+shift(ii-1);
   if ~isempty(z)
       z(z(:,2)== ii,1) = z(z(:,2)== ii,1)+shift(ii-1);
   end
end

% precompute distances
if dg                                                               % vector kxx
  K = zeros(size(x(:,1),1),1);
else
  if xeqz                                                 % symmetric matrix Kxx
    K = sqrt(sq_dist(x(:,1)'));
  else                                                   % cross covariances Kxz
    K = sqrt(sq_dist(x(:,1)',z(:,1)'));
  end
end

K = pi*K/p;
if nargin<4                                                        % covariances
    K = sin(K); K = K.*K; K =   exp(-2*K);
else                                                               % derivatives
    if i<=nL
        if i==1
            R = sin(K); K = 4*exp(-2*R.*R).*R.*cos(K).*K;
        else  % derivatives of the shift hyperparameters
          dim = mod(i,2)+1;
          ind_i = (x(:,2) ==i);
          ind_ni = (x(:,2) ~=i);
          B = zeros(length(x));
          B(ind_ni,ind_i) = ones(sum(ind_ni),sum(ind_i));
          B(ind_i,ind_ni) = -ones(sum(ind_i),sum(ind_ni));

          R = sin(K);
          A = repmat(x(:,dim) ,[1 length(x)]);

          K = 4.*exp(-2*R.*R).*R.*cos(K).*pi./p.*sign(A-A');

          K = B.*K;
      end
    else
        error('Unknown hyperparameter')
    end

end
```