MAE 6258 HW11

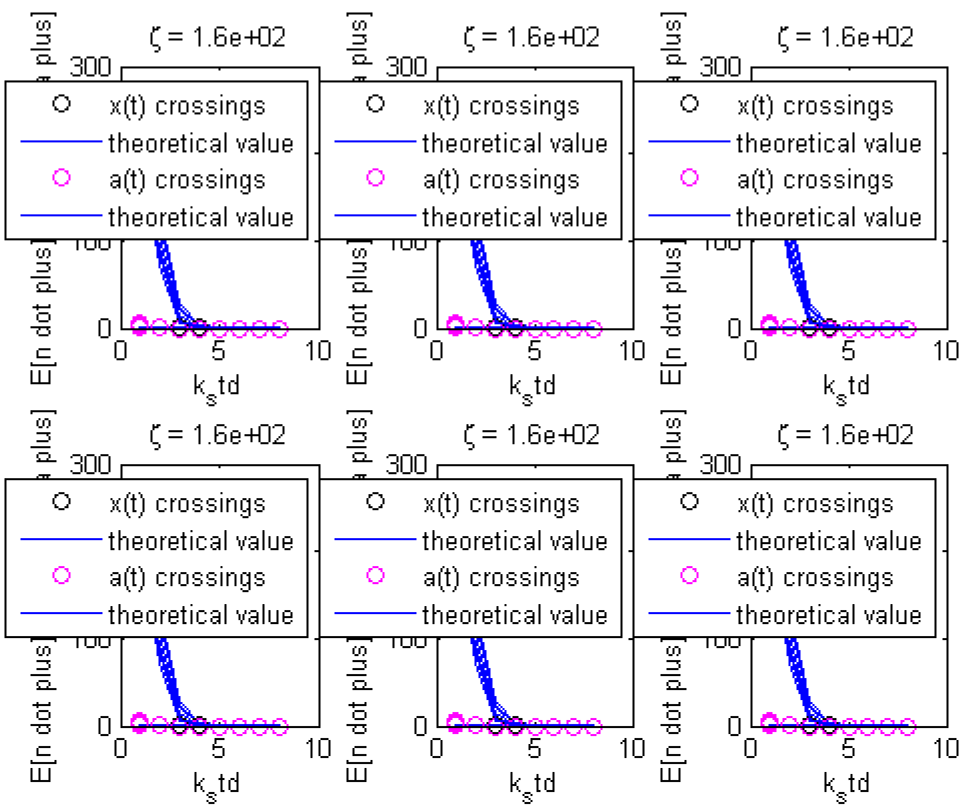
```
%Randy Schur
%4/15/15
%Adapted from yield_failure.m
```

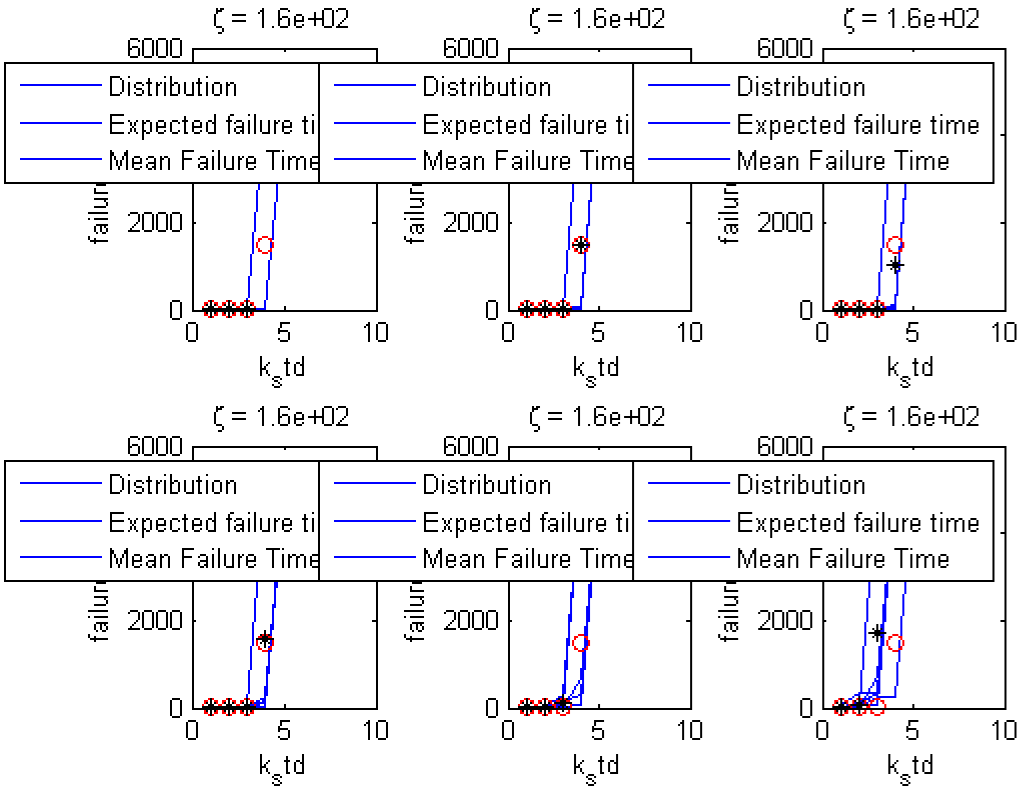
Problem 1

```
clc
clear
close all

c = [.1 1 10 100 1000 10000];
t_failure = zeros(6,8,10);
for times = 1:10
    for zetas = 1:length(c)
    % create linear system
    m = 5;
    % % c = .1;
    wn = 1*2*pi;                             % natural frequency
    k = wn.^2*m;
    G = tf(1,[m c(zetas) k]);

    zeta = c(zetas)/(2*sqrt(m*k))  ;                   % damping ratio
    Tc = 1/(wn*zeta);                           % time constant of transients

    % create Gaussian white noise input
    dt = .01;
    N = 500000;
    t = linspace(0,(N-1)*dt,N);
    S0 = 1;


    u = randn(N,1)*sqrt(2*pi*S0/dt);

    % simulate system with input u, plot input vs. output
    y = lsim(G,u,t);
    ind = (t > 5*Tc);   % trim transient response
    t = t(ind);
    t = t - t(1);
    N = length(t);
    u = u(ind);
    y = y(ind);


    % mean-square of output
    R_yy = xcorr(y,'unbiased');
    Ey2 = R_yy(N);
    Ey2a = mean(y.^2);
    Ey2_theory = pi/(c(zetas)*k);
```

```
    for k_std = 1:8
    % failure estimates
    % k_std = 3;                                  % # of standard deviations t
    U = k_std*std(y);       % std(y) = sqrt(Ey2a)

    %calculate n_dot_plus(k*sigma_x)
    [pks, locs] = findpeaks(y);
    ind_fplus = (pks >= U);
    pos_pks = find(ind_fplus>0);
    n_dot_plus(k_std, times) = size(pos_pks,1)/t(end);
    a_crosses = 0;
    for count = 2:length(pos_pks)
        diff = pos_pks(count) - pos_pks(count-1);
        if diff >sqrt(Ey2a)*sqrt(6/pi)/(pi*zeta*U)
           a_crosses = a_crosses+1;
        end
    end
    n_dot_plus_a(k_std, times) = a_crosses/t(end);
    if numel(pos_pks)>0
       t_failure(zetas, k_std, times) = t(pos_pks(1));
    else
       t_failure(zetas, k_std, times) = t(end);
    end

    %theoretical values
    n_dot_plus_theory(k_std, times) = 1*exp(-k_std^2/2);
    sigma_x_sq = Ey2_theory;
    K = 2*zeta*wn^3*sigma_x_sq/pi;
    sigma_xdot_sq = pi*K/(2*zeta*wn^3)*((1+pi^2*zeta^2)/12);
    sigma_1_sq = pi^3*zeta*K/(24*wn);

    n_dot_plus_a_theory(k_std, times) = U*sqrt(sigma_1_sq)/(sqrt(2*pi)*sigma_x

    end

end
figure(1)
for asdf=1:zetas
    subplot(2,3, asdf)
    plot(1:k_std, mean(n_dot_plus,2), 'ko', 1:k_std, n_dot_plus_theory(:,times
    hold on
    plot(1:k_std, mean(n_dot_plus_a(:,times),2), 'mo', 1:k_std, n_dot_plus_a_t
    legend( 'x(t) crossings', 'theoretical value', 'a(t) crossings', 'theoreti
    str = strcat('\zeta', sprintf(' = %1.1d', zeta ));
    title(str)
    xlabel('k_std')
    ylabel('E[n dot plus] and E[n dot a plus]')
end
figure(2)
for i=1:6
   subplot(2,3,i)
   plot(1:k_std, t_failure(i,:,times))
   hold on
    str = strcat('\zeta', sprintf(' = %1.1d', zeta ));
```

```
        title(str)
        xlabel('k_std')
        ylabel('failure time')
    end
end

%part b
figure(2)
E_tf = min(5000, 1/2.*exp((1:k_std).^2./2));

for i=1:6
   subplot(2,3,i)
   plot(1:k_std, E_tf, 'ro')
   hold on
   plot(1:k_std, mean(t_failure(i,:,:),3), 'k*')
    str = strcat('\zeta', sprintf(' = %1.1d', zeta ));
    title(str)
    xlabel('k_std')
    ylabel('failure time')
    legend('Distribution','Expected failure time', 'Mean Failure Time')
end
```





Published with MATLAB® R2013b