Appendix

No.1 Use Newton's method to solve the system of equations

```
clear
clc

% F(X)
F = @(x) [(x(1)^2 + x(2)^2 + x(3)^2 -3) (x(1)^2 + x(2)^2 - x(3)^2 -1) (x(1)+x(2)+x(3) - 3)]';

% Jacobian Matrix
J = @(x) [2*x(1)  2*x(2)  2*x(3); 2*x(1)  2*x(2) -2*x(3); 1 1 1];

% initial values
xo = [0.9,0.8,0.95]';

max_iter = 50;
tol = 1e-8;

% Newton method
for k = 1:max_iter
   dx = -inv(J(xo))*F(xo);
   x = xo + dx;
   if norm(x-xo) < tol
      fprintf('Number of iterations taken = %d',k);
      break
   end
   xo = x;

end
x
```

No.3

```
sig = 0.15;
Y = @(t,m) (m(1).*exp(m(2).*t))./sig;
t = [1 2 4 5 8]';
y = [3.2939 4.2699 7.1749 9.3008 20.259]'./sig;

m1 =2; m2 = 0; mo = [m1 m2]';

tol = 1e-6;

J = @(t,m) [(exp(m(2).*t))./sig (m(1).*t.*exp(m(2).*t))./sig];
%J(t,mo)

% Gaus-Newton method
for k = 1:max_iter
   dm = -inv(J(t,mo)'*J(t,mo))*J(t,mo)'*(Y(t,mo)-y);
   m = mo + dm;
   if norm(m-mo) < tol
      fprintf('Number of iterations taken = %d',k);
      break
   end
   mo = m;
```

```
end
%J(t,m)

disp(['Resulting parameter estimates are [',num2str(m'),']'])
%chi-square
chi_s = 0;
for i = 1:length(t)
    chi_s = chi_s + ((Y(t(i),m)-y(i)))^2;

end

disp(['chi-square obs = ',num2str(chi_s)])

% pvalue
[m,n] = size(J(t,m));
p = 1 - chi2cdf(chi_s,m);
disp(['pvalue = ',num2str(p)])
```