1.

The Jacobian Matrix, $J(x) = \begin{bmatrix} 2x & 2y & 2z \\ 2x & 2y & -2z \\ 1 & 1 & 1 \end{bmatrix}$

I choose the initial estimate by guessing small positive values near one.

2. Assume that $G(m) = Gm$, i.e. G is linear

a) Prove that $J(m^{(k)}) = G$ for all $m^{(k)}$, $k = 1, 2, \dots, n$

$$\text{let } G(m) = [g_1(m), g_2(m) \dots g_m(m)]^T$$

then,

$$J(m^{(k)}) = \frac{\partial}{\partial m}(G(m)) = \begin{bmatrix} \frac{\partial}{\partial m_1}(g_1(m)) & \cdots & \frac{\partial}{\partial m_n}(g_1(m)) \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial m_1}g_m(m)) & \cdots & \frac{\partial}{\partial m_n}(g_m(m)) \end{bmatrix}$$

Suppose that $G(m) = Gm \Rightarrow g(m) = gm$

So,

$$Gm = \underbrace{\begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & \ddots & \vdots \\ g_{m1} & \cdots & g_{mn} \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} m_1 \\ \vdots \\ m_n \end{bmatrix}}_{m}$$

There by we get a dot product between the rows of G and m, then

$$J(m^{(k)}) = \frac{\partial}{\partial m_k}(g_{kj} m_k) = g_{kj} \quad , j = 1, 2, \dots m$$

hence $\underline{J(m^{(k)}) = g_{kj} = G \ \forall \ m^{(k)}}$

b) Show that one iteration of the Gauss-Newton Method as defined by (9.29) will give the linear least square estimation.

$$J(m^{(k)})^T J(m^{(k)}) \Delta m = -J(m^{(k)})^T F(m^{(k)}) \quad — ①$$

from 2(a) suppose $G(m) = Gm \Rightarrow J(m^{(k)}) = G \Rightarrow F(m^{(k)}) = Gm^{(k)} - d$

therefore Equation ① becomes

$$G^T G \Delta m = -G^T (Gm^{(k)} - d)$$

but $\Delta m = m^{(k+1)} - m^{(k)}$

$$G^T G (m^{(k+1)} - m^{(k)}) = -G^T G m^{(k)} + G^T d$$

$$G^T G m^{(k+1)} - \cancel{G^T G m^{(k)}} = -\cancel{G^T G m^{(k)}} + G^T d$$

$$m^{(k+1)} = -(G^T G)^{-1} G^T d$$

So $m^{(k+1)} = -(G^T G)^{-1} G^T d$ gives the linear least squares estimate.

At $k=0$, $m' = -(G^T G)^{-1} G^T d$, therefore after one iteration of the Gauss-Newton method, we obtain the linear least squares estimate.

8. Report the Jacobian Matrix

$$J(x) = \begin{bmatrix} (e^{m_2 t_1})/\sigma & (m_1 t_1 e^{m_2 t_1})/\sigma \\ \vdots & \vdots \\ (e^{m_2 t_5})/\sigma & (m_1 t_5 e^{m_2 t_5})/\sigma \end{bmatrix}$$

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

```
Number of iterations taken = 26

x = 3×1
    1.0000
    1.0000
    1.0000
```

No.3

```
Number of iterations taken = 14

Resulting parameter estimates are [2.5411      0.2595]

chi−square obs = 2.8813e−07

pvalue = 1
```

Since the value of the p-value is 1, then we reject the null hypothesis, since the fit of the model predictions to the data is almost exact, which is not realistic hence, the parameter estimates are not good.

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)])
```