Q1.

a). $Q(Ex) = (Ex)^T \cdot A \cdot Ex$

$$= (x - M^{-1}Ax)^T \cdot A \cdot (x - M^{-1}Ax)$$

$$= (x-y)^T \cdot A(x-y)$$

$$= (x^T - y^T)(Ax - Ay)$$

$$= x^TAx - y^TAx - x^TAy + y^TAy$$

$Q(x) = x^TAx$

$$-y^TDy = y^T(-D)y = y^T(-M - M^T + A)y$$

$$= -y^TMy - y^TM^Ty + y^TAy$$

since $y = M^{-1}Ax$ , $My = Ax$

$$y^T = x^TA^T(M^{-1})^T$$

$$y^TM^T = x^TA^T$$

$$RHS = x^TAx - y^TMy - y^TM^Ty + Ay$$

$$= x^TAx - y^TAx - x^TA^Ty + y^TAy$$

since A is symmetric,

$$A^T = A$$

$\therefore RHS = x^TAx - y^TAx - x^TAy + y^TAy = LHS$

$\therefore$ prooved

Q1.

b). since $e^{k+1} = E \cdot e^k$ on slides P19

$LHS = Q(e^{k+1}) = Q(E \cdot e^k)$

$RHS = Q(e^k) - (y^k)^T D y^k$

since $Q(Ex) = Q(x) - y^T D y$ for $y = M^{-1} A x$

substitude $x$ ~~with~~ $e^k$

$LHS = Q(Ee^k)$

$RHS = Q(e^k) - (M^{-1} A e^k)^T D \cdot (M^{-1} A e^k)$

$= Q(e^k) - (y^k)^T D \cdot y^k$

$\therefore Q(Ee^k) = Q(e^k) - (y^k)^T D \cdot y^k$

c) since $A$ is SPD

$\therefore$ the diagonal of $A$ is positive

$\therefore D$ is also SPD

$\therefore$ the inner product of $D$ $(y^k)^T D (y^k) > 0$

since $Q(e^{k+1}) - Q(e^k) = -(y^k)^T D (y^k) < 0$

$\therefore Q(e^{k+1}) < Q(e^k)$

$\therefore Q(e^k)$ is a converging sequence

$\therefore$ the GS method converges for any $x^k$

since $Q(e') = e'^T A e'$ is the largest term in $Q(e^k)$

As $A$ is symmetric and positive, $Q(e') > 0$

$\therefore$ for all $Q(e^k) < Q(e')$ are all positive

$\therefore x^T A x > 0$ for any $x \neq 0$ $\therefore A$ must be PD

Q2.

$$(y_{k+2}-y_{k+1})^T(y_{k+1}-y_k)$$

$$=(y_{k+1}+\alpha_{k+1}d_{k+1}-y_{k+1})^T(y_k+\alpha_k d_k-y_k)$$

$$=(\alpha_{k+1}d_{k+1})^T(\alpha_k d_k)$$

since $\alpha_k, \alpha_{k+1}$ are constants

$\therefore$ LHS $=\alpha_k\alpha_{k+1}(d_{k+1}^T d_k)$

$$(d_{k+1})^T d_k=(b-Ay_{k+1})^T d_k$$

$$=(b-Ay^k-\alpha_k Ad^k)^T d_k$$

$$=(d_k-\alpha_k Ad^k)^T d_k$$

$$=d_k^T d_k-\alpha_k d_k^T A^T d_k$$

since $\alpha_k=\dfrac{d_k^T\cdot d_k}{d_k^T\cdot A\cdot d_k}$

$$d_{k+1}^T\cdot d_k=d_k^T d_k-\frac{d_k^T\cdot d_k}{d_k^T\cdot A^T d_k}\cdot d_k^T A^T d_k$$ as $A$ is symmetric

$$=d_k^T d_k-d_k^T d_k=0$$

$$\therefore (y_{k+2}-y_{k+1})^T\cdot(y_{k+1}-y_k)=0$$

Q3、 $y = [x_0, x_1, \cdots, x_{n-1}, \underbrace{0, 0, \cdots, 0}_{n}]$

$\omega = e^{-\frac{\pi i}{n}}$

$\hat{y}_k = \sum_{j=0}^{2n-1} y_j \cdot (e^{-\frac{\pi i}{n}})^{jk}$ since $y_k = 0$ for $k \in [n, 2n-1]$

$\therefore \hat{y}_k = \sum_{j=0}^{n-1} y_j \omega^{jk} + \sum_{j=n}^{2n-1} y_j \omega^{jk}$

$= \sum_{j=0}^{n-1} y_j \omega^{jk}, \; k \in [0, 2n-1]$

for $k$ is even, let $k = 2m$, $m \in [0, n-1]$

$\hat{y}_{2m} = \sum_{j=0}^{n-1} y_j \cdot e^{-\frac{2\pi i jm}{n}}$

$\hat{x}_m = \sum_{j=0}^{n-1} x_j \cdot e^{-\frac{2\pi i jm}{n}}$

since $y_j = x_j$ for $j \in [0, n-1]$

$\therefore \hat{y}_{2m} = \hat{x}_m$ for $m \in [0, n-1]$

$\therefore$ for $k$ is even

$\hat{y}_k = \hat{x}_{k/2}$

Q3. for $k$ is odd, Let $k=2m+1$, $m\in[0,n-1]$

$$\hat{y}_{2m+1}=\sum_{j=0}^{n-1} y_j\, e^{-\frac{\pi i}{n}\cdot j\cdot(2m+1)}=\sum_{j=0}^{n-1} x_j\cdot e^{-\frac{\pi i}{n}2mj}\cdot e^{-\frac{\pi i j}{n}}$$

$$\hat{x}_m=\sum_{j=0}^{n-1} x_j\cdot e^{-\frac{2\pi i}{n}\cdot mj}=\sum_{j=0}^{n-1} x_j\cdot e^{-\frac{\pi i}{n}\cdot 2mj}$$

$$y_j=\frac{1}{2n}\sum_{k=0}^{2n-1}\hat{y}_k\, e^{-\frac{\pi i}{n}\cdot jk}$$

$$x_j=\frac{1}{n}\sum_{k=0}^{n-1}\hat{x}_k\, \omega_\alpha^{-jk}$$

$$\hat{y}_{2m+1}=\sum_{j=0}^{n-1} y_j\cdot e^{-\frac{\pi i}{n}j(2m+1)}$$

$$=\sum_{j=0}^{n-1} x_j\cdot e^{-\frac{\pi i}{n}j(2m+1)}$$

$$=\sum_{j=0}^{n-1}\frac{1}{n}\left(\sum_{k=0}^{n-1}\hat{x}_k\cdot e^{\frac{2\pi i}{n}jk}\right)\cdot e^{-\frac{\pi i}{n}j(2m+1)}$$

$$=\frac{1}{n}\sum_{j=0}^{n-1} e^{-\frac{\pi i}{n}j(2m+1)}\cdot\sum_{k=0}^{n-1}\hat{x}_k e^{\frac{2\pi i}{n}jk}$$

$$\hat{y}_k=\frac{1}{n}\sum_{j=0}^{n-1} e^{-\frac{\pi i}{n}jk}\cdot\sum_{m=0}^{n-1}\hat{x}_m\cdot e^{\frac{2\pi i}{n}jm}$$

$$=\frac{1}{n}\sum_{j=0}^{n-1}\sum_{m=0}^{n-1}\hat{x}_m\cdot e^{\frac{\pi i}{n}(2m-k)}$$

Q3.

$$\hat{y}_k = \frac{1}{n} \sum_{j=0}^{n-1} \sum_{m=0}^{n-1} \hat{x}_m \cdot e^{\frac{\pi i}{n} \cdot j \cdot (2m-k)}$$

$$= \frac{1}{n} \sum_{m=0}^{n-1} \sum_{j=0}^{n-1} \hat{x}_m \cdot e^{\frac{\pi i (2m-k)}{n} \cdot j}$$

$$= \frac{1}{n} \sum_{m=0}^{n-1} \hat{x}_m \cdot \frac{1 - e^{\frac{\pi i (2m-k) n}{n}}}{1 - e^{\frac{\pi i (2m-k)}{n}}}$$

$$= \frac{1}{n} \sum_{m=0}^{n-1} \hat{x}_m \cdot \frac{1 - e^{2m\pi i - k\pi i}}{1 - e^{\frac{\pi i (2m-k)}{n}}}$$

$$= \frac{1}{n} \sum_{m=0}^{n-1} \hat{x}_m \cdot \frac{1 - e^{2m\pi i} \cdot e^{-k\pi i}}{1 - e^{\frac{\pi i (2m-k)}{n}}}$$

since $m \in \mathbb{Z}$, $k$ is odd

$$\therefore e^{2m\pi i} = 1, \quad e^{-k\pi i} = -1$$

$$\therefore \hat{y}_k = \frac{1}{n} \sum_{m=0}^{n-1} \hat{x}_m \frac{1 - (-1) \cdot 1}{1 - e^{\frac{\pi i (2m-k)}{n}}}$$

$$= \frac{2}{n} \sum_{m=0}^{n-1} \hat{x}_m \cdot \left(1 - e^{\frac{\pi i (2m-k)}{n}}\right)^{-1}$$

when $k$ is odd

Q4. $x'(t) = 10x(t) + 11t - 5t^2 - 1, \quad x(0) = 0, \quad t \in [0,3]$

$$x'(t) - 10 \cdot x(t) = 11t - 5t^2 - 1$$

Integrating factor: $\mu(t) = e^{\int -10 \cdot dt} = e^{-10t}$

$$\mu(t) \cdot x'(t) + \mu(t)(-10x(t)) = \mu(t) \cdot (11t - 5t^2 - 1)$$

$$\mu(t) \cdot x(t) = \int \mu(t) \cdot (11t - 5t^2 - 1) \cdot dt$$

$$\mu(t) \cdot x(t) = \int e^{-10t} \cdot (11t - 5t^2 - 1)\, dt$$

$$RHS = 11\int t \cdot e^{-10t}\, dt - 5\int t^2 \cdot e^{-10t}\, dt - \int e^{-10t}\, dt$$

$$= 11 \cdot \left[-\frac{t e^{-10t}}{10} + \int \frac{e^{-10t}}{10}\, dt\right] - 5\left[-\frac{t^2 e^{-10t}}{10} + \int \frac{t e^{-10t}}{5}\, dt\right] + \frac{e^{-10t}}{10} + C$$

$$= 11 \cdot \left[-\frac{t e^{-10t}}{10} - \frac{e^{-10t}}{100}\right] - 5\left[-\frac{t^2 e^{-10t}}{10} + \frac{1}{5} \cdot \left(-\frac{t e^{-10t}}{10} - \frac{e^{-10t}}{100}\right)\right] + \frac{e^{-10t}}{10} + C$$

$$= \frac{1}{2} \cdot t^2 \cdot \mu(t) - t \cdot \mu(t) + C$$

$$\therefore x(t) = \frac{1}{2}t^2 - t + C \cdot e^{10t}$$

for $x(0) = 0$

$C = 0$

$$\therefore x(t) = \frac{1}{2}t^2 - t$$

for $x(0) = \varepsilon$

$0 - 0 + C = \varepsilon$

$\therefore C = \varepsilon$

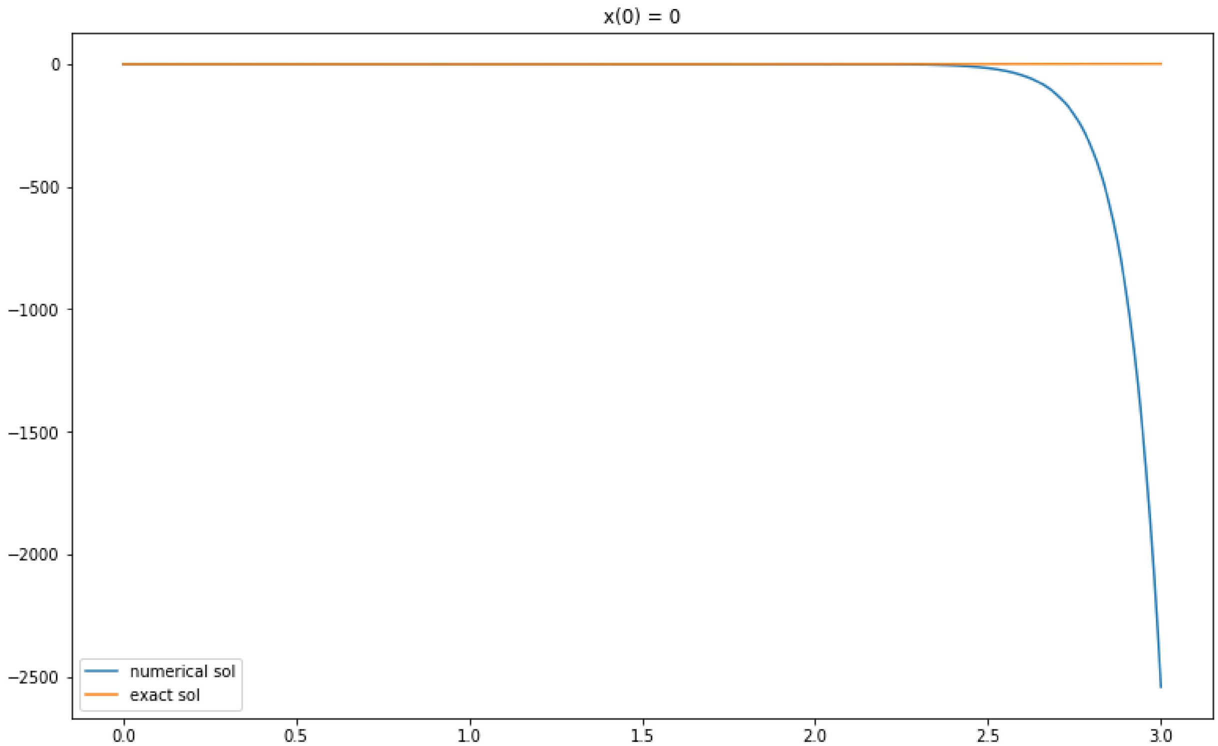
$$\therefore x(t) = \frac{1}{2}t^2 - t + \varepsilon \cdot e^{10t}$$

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
def dxdt(t, x):
    return 10*x+11*t-5*pow(t, 2)-1


def x(t):
    return pow(t, 2)/2-t


def RK4(x0, t0, tmax, h):
    ts = []
    xs = []
    ts.append(t0)
    xs.append(x0)
    i = 0
    while ts[i]+h <= tmax:
        k1 = dxdt(ts[i], xs[i])
        k2 = dxdt(ts[i] + 0.5 * h, xs[i] + 0.5 * h * k1)
        k3 = dxdt(ts[i] + 0.5 * h, xs[i] + 0.5 * h * k2)
        k4 = dxdt(ts[i] + h, xs[i] + h * k3)

        xs.append(xs[i] + (h / 6)*(k1 + 2 * k2 + 2 * k3 + k4))
        ts.append(ts[i] + h)
        i += 1
    return ts, xs
```

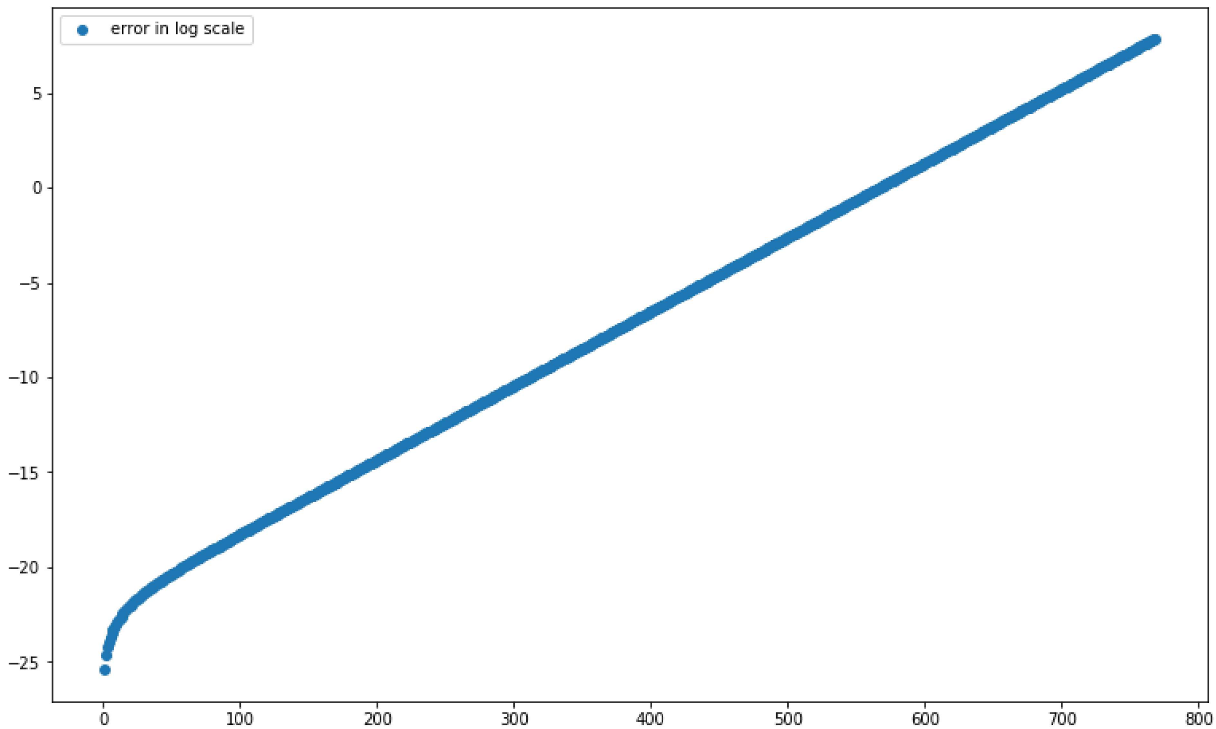
In [3]:
```
h = pow(2, -8)
t_numerical, x_numerical = RK4(0, 0, 3, h)
t_exact = np.linspace(0, 3, len(x_numerical))
x_exact = x(t_exact)
```

In [4]:
```
plt.figure(figsize=[13, 8])
plt.plot(t_numerical, x_numerical, label='numerical sol')
plt.plot(t_exact, x_exact, label='exact sol')
plt.title('x(0) = 0')
plt.legend(loc='best')
plt.show()
```



In [5]:
```
err = np.log(abs(x_numerical-x_exact))
slope = np.polyfit(t_numerical[50:],err[50:],1)[0]
print(f"the slope of the linear relationship between numerical solution and exac
xslength = len(err)
plt.figure(figsize=[13, 8])
plt.scatter(np.arange(xslength), err, label='error in log scale')
plt.legend(loc='best')
plt.show()
```

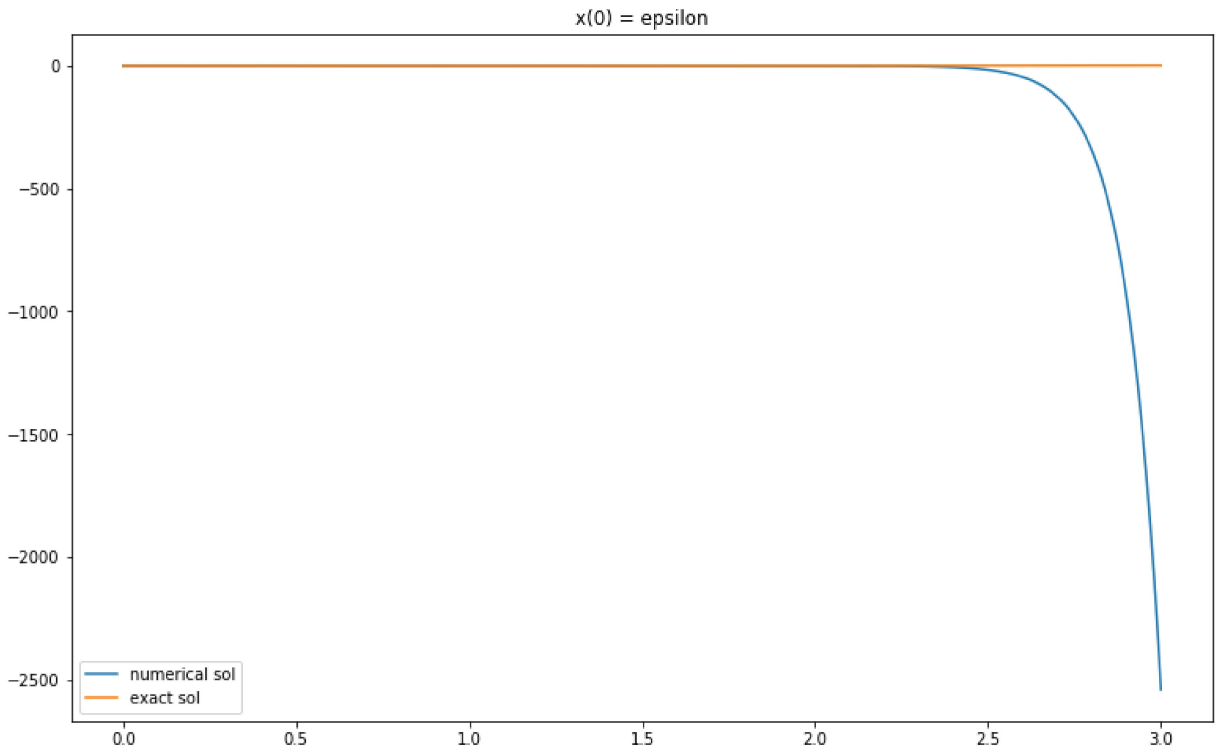
```
the slope of the linear relationship between numerical solution and exact solut
ion is 10.010627200924409

C:\Users\zhc-2\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: RuntimeWarn
ing: divide by zero encountered in log
  """Entry point for launching an IPython kernel.
```




In [6]:
```
x0err = x_numerical[50] - x_exact[50]
xn_err = x_numerical[xslength-1]-x_exact[xslength-1]
print(f" the difference between x0 times 7^18 {x0err*pow(h/6,xslength-50)}\n the
```

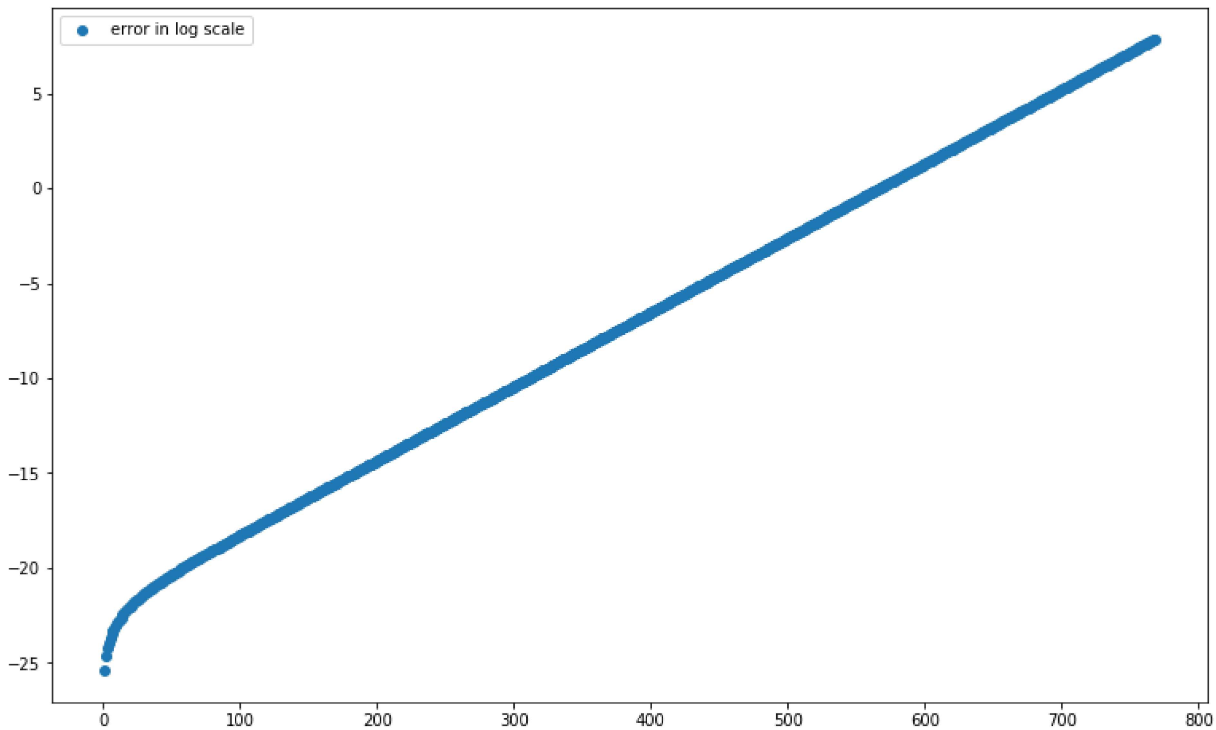
```
 the difference between x0 times 7^18 -0.0
 the difference between x18 -2541.5180896071597
```

In [7]:
```
float_info = np.finfo(float)
epsilon = float_info.eps
t_num_eps, x_num_eps = RK4(0,epsilon,3,h)
plt.figure(figsize=[13, 8])
plt.plot(t_num_eps, x_num_eps, label='numerical sol')
plt.plot(t_exact, x_exact, label='exact sol')
plt.title('x(0) = epsilon')
plt.legend(loc='best')
plt.show()
```



In [8]:
```
err_eps = np.log(abs(x_num_eps-x_exact))
slope_eps = np.polyfit(t_num_eps[50:],err_eps[50:],1)[0]
print(f"the slope of the linear relationship between numerical solution with x(0
plt.figure(figsize=[13, 8])
plt.scatter(np.arange(xslength), err, label='error in log scale')
plt.legend(loc='best')
plt.show()
```

```
the slope of the linear relationship between numerical solution with x(0) = eps
ilon and exact solution is 10.01062720383668

C:\Users\zhc-2\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: RuntimeWarn
ing: divide by zero encountered in log
  """Entry point for launching an IPython kernel.
```



The errors clearly forms a line in log scale after arpund 20 iterations, this means the error is multiplied by a certain number when we using the reccurence formulae to solve the ODE. The change of initial value by machine epsilon does not affect this result quite much.

This logrithmic linearity indicates that the error is roughly increased by 10 in each iteration. We canm find this error might occur because of each time we call function dxdt, x is multiplied by 10 and therefore leads to the significant increase in error.

In [ ]: