# 第四题

## Matlab源码

%% test the algorithm

max\_n=50;

depature = zeros(1,max\_n);

for n = 1:max\_n

U = orth(rand(n,n));

A = U\*diag(10\*rand(1,n))\*U';

exp\_A\_by\_function = matrix\_exponential(A);

exp\_A = U\*exp(diag(1:n))\*U';

depature(1,n) = norm(exp\_A\_by\_function-exp\_A,'fro');

end

figure()

plot([1:max\_n],log10(depature),'--o');

xlabel("矩阵阶数")

ylabel("log10(depature)")

[t,s] = title('误差（取对数）','矩阵阶数：1~50')

%% implement the scaling\_and\_squaring algorithm for computing the matrix exponential

% (combined with truncated Taylor series)

function exp\_M = matrix\_exponential(M)

n = size(M);

n = n(1,1);

e = eig(M);

max\_eigenvalue = max(e);

k = ceil(log2(max\_eigenvalue/0.001));

M\_scaled = M/(2^k);

exp\_M\_scaled = eye(n) + M\_scaled + (1/2)\*M\_scaled\*M\_scaled + (1/6)\*M\_scaled\*M\_scaled\*M\_scaled + (1/24)\*M\_scaled\*M\_scaled\*M\_scaled\*M\_scaled;% truncated Taylor series

exp\_M = exp\_M\_scaled;

for i = 1:k

exp\_M = exp\_M\*exp\_M;

end

end

## 试验结果

