Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

SKU: 9781498757775-SOLUTIONS

Chapter 2

Fundamentals of MATLAB Programming

Exercise 2.1 *In MATLAB environment, the following statements can be given*

`tic, A=rand(500); B=inv(A); norm(A*B-eye(500)), toc`

Run the statements and observe results. If you are not sure with the commands, just use the on-line `help` *facilities to display information on the related functions. Then explain in detail the statement and the results.*

Solution What the statements actually do is to calculate and verify the inverse matrix $\boldsymbol{B}$ of a 500×500 randomly generated matrix $\boldsymbol{A}$, and measure the total time consumes. It can be found that the precision reaches 10^{-12}-level, and the time required is around one second.

Exercise 2.2 *Suppose that a polynomial can be expressed by* $f(x) = x^5 + 3x^4 + 4x^3 + 2x^2 + 3x + 6$. *If one wants to substitute* x *by* $(s-1)/(s+1)$, *the function* $f(x)$ *can be changed into a function of* s. *Use the Symbolic Math Toolbox to do the substitution and get the simplest result.*

Solution One should declare the two variables s and x as symbolic variables, then the `subs()` function should be used to do variable substitution. Finally, simplification of the results should be performed

```
>> syms s x, f=x^5+3*x^4+4*x^3+2*x^2+3*x+6;
   F=subs(f,x,(s-1)/(s+1)), F=simplify(F)
```

which leads to the result $F = \dfrac{3 + 23s + 54s^2 + 70s^3 + 19s^5 + 23s^4}{(s+1)^5}$.

Exercise 2.3 *Please simplify* $\sin(k\pi + \pi/6)$, *for any integer* k.

Solution In the new versions of MATLAB, integer symbolic variables are supported. The original problem can be solved with the following statements, with the result $(-1)^k/2$.

```
>> syms k, assume(k,'integer'); simplify(sin(k*pi+pi/6))
```

Exercise 2.4 *Input the matrices* $\boldsymbol{A}$ *and* $\boldsymbol{B}$ *into MATLAB workspace where*

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \\ 2 & 3 & 4 & 1 \\ 3 & 2 & 4 & 1 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 1+\mathrm{j}4 & 2+\mathrm{j}3 & 3+\mathrm{j}2 & 4+\mathrm{j}1 \\ 4+\mathrm{j}1 & 3+\mathrm{j}2 & 2+\mathrm{j}3 & 1+\mathrm{j}4 \\ 2+\mathrm{j}3 & 3+\mathrm{j}2 & 4+\mathrm{j}1 & 1+\mathrm{j}4 \\ 3+\mathrm{j}2 & 2+\mathrm{j}3 & 4+\mathrm{j}1 & 1+\mathrm{j}4 \end{bmatrix}.$$

It is seen that $\boldsymbol{A}$ *is a* 4×4 *matrix. If a command* $\boldsymbol{A}(5,6) = 5$ *is given, what will happen?*

Solution The two matrices can be specified easily with the following statements

```
>> A=[1 2 3 4; 4 3 2 1; 2 3 4 1; 3 2 4 1]
   B=[1+4i 2+3i 3+2i 4+1i;  4+1i 3+2i 2+3i 1+4i;
      2+3i 3+2i 4+1i 1+4i; 3+2i 2+3i 4+1i 1+4i];
```


K27591_SM_Cover.indd 10 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

If further the command

```
>> A(5,6)=5
```

is used, and columns and rows in the statements are all greater than the current size of $\boldsymbol{A}$, zero terms are introduced to the extended part of $\boldsymbol{A}$, then the (5,6)th term is assigned to 5.

Exercise 2.5 *Command* $\boldsymbol{A}$=`rand(3,4,5,6,7,8,9,10,11)` *can be used to generate a multi-dimensional array. How many elements are there in the array? Please find the sum of all its elements.*

Solution The multi-dimensional array can be generated first, and the size of the array can be measured with the following statements, and it can be seen that there are a total of 19958400 elements in the array, and the sum under this run is about 9.9791×10^6. Since the quantities in the array are generated randomly, the sums under each run are different.

```
>> A=rand(3,4,5,6,7,8,9,10,11); prod(size(A)), sum(A(:))
```

Exercise 2.6 *Find the first 200 digits of the irrational numbers* $\sqrt{2}$, $\sqrt[6]{11}$, $\sin 1^\circ$, e^2, $\ln 21$.

Solution Since the quantities are irrational numbers, Symbolic Toolbox should be used to display all the first 200 digits. The commands to use are as follows, and the displays are omitted here.

```
>> e1=vpa(sqrt(sym(2)),200), e2=vpa(sym(11)^(1/6),200),
   e3=vpa(sin(sym(pi/180)),200), e4=vpa(exp(sym(2)),200),
   e5=vpa(log(sym(21)),200),
```

Exercise 2.7 *Please show the following identical equations*

$$(i)\ \mathrm{e}^{\mathrm{j}\pi}+1=0,\quad (ii)\ \frac{1-2\sin\alpha\cos\alpha}{\cos^2\alpha-\sin^2\alpha}=\frac{1-\tan\alpha}{1+\tan\alpha}.$$

Solution To show identities, the best way is to move everything to the left of the equation sign, and simplify the result. If the simplified quantities of the left-hand-side are zeros, the equations are proven. The two identical equations can be proven directly with the following statements

```
>> simplify(exp(sym(pi)*1i)+1)
   syms a;
   simplify((1-2*sin(a)*cos(a))/(cos(a)^2-sin(a)^2)...
        -(1-tan(a))/(1+tan(a)))
```

Exercise 2.8 *If* $f(x)=x^2-x-1$, *please find* $f(f(f(f(f(f(f(f(f(f(x))))))))))$, *and also express the result in a polynomial. What is the degree of the polynomial?*

Solution In recent versions of MATLAB, symbolic function definition of $f(x)$ is allowed. Therefore, the composite function can be constructed. The degree of the final polynomial is as high as 1024, and the expanded polynomial can be obtained, and omitted. It should be noted that when $f(x)$ is to be used, it should be declared with `syms` command first, otherwise it may not be recognized.

```
>> syms x f(x); f(x)=x^2-x-1; F=f(f(f(f(f(f(f(f(f(f(x))))))))));
   expand(F), n=length(sym2poly(ans))-1
```

K27591_SM_Cover.indd 11 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

Exercise 2.9 *If the mathematical functions are known*

$$f(x) = \frac{x \sin x}{\sqrt{x^2+2}(x+5)},\ g(x) = \tan x,$$

please find out the functions $f(g(x))$ *and* $g(f(x))$.

Solution With the `subs()` function, variable substitutions can be made, to compute the two composite functions

```
>> syms x; f=x*sin(x)/sqrt(x^2+2)/(x+5);
   g=tan(x); F1=simplify(subs(f,x,g)), F2=simplify(subs(g,x,f))
```

and the results obtained are

$$F_1(x) = \frac{\sin(\tan x)\ \tan x}{(\tan x + 5)\sqrt{\tan^2 x + 2}},\ F_2(x) = \tan\left(\frac{x \sin x}{\sqrt{x^2+2}\,(x+5)}\right)$$

Alternatively, the following statements can be used in new the versions, and the same results can be obtained

```
>> syms x g(x) f(x); f(x)=x*sin(x)/sqrt(x^2+2)/(x+5);
   g(x)=tan(x); F1=simplify(f(g(x))), F2=simplify(g(f(x)))
```

Exercise 2.10 *Since double-precision scheme is quite limited in representing accurately large numbers, symbolic calculations are often used to find factorials of large numbers. Please use numerical and symbolic methods to calculate* C_{50}^{10}*, where,* $\mathrm{C}_m^n = m!/[n!(m-n)!]$. *Alternatively function* `nchoosek(sym(`m`),`n`)` *can be used.*

Solution The factorial $n!$ can be evaluated with `prod(1:`n`)`, and it is also possible to evaluated from `gamma(`n`+1)`. If n is very large, n should be converted to symbolic data type first. The following statements can be used

```
>> m=50; n=10; r1=gamma(m+1)/gamma(n+1)/gamma(m-n+1)
   m=sym(m); n=sym(n); r2=gamma(m+1)/gamma(n+1)/gamma(m-n+1)
```

and the numerical and analytical solutions can be obtained as $r_1 = 1.027227816999992 \times 10^{10}$, $r_2 = 10272278170$. Alternatively, `nchoosek()` function can be used and the same result can be obtained.

```
>> r3=nchoosek(sym(m),n)
```

Exercise 2.11 *For a matrix* $\boldsymbol{A}$*, if one wants to extract all the even rows to form matrix* $\boldsymbol{B}$*, what command should be used? Suppose that matrix* $\boldsymbol{A}$ *is defined by* $\boldsymbol{A}$`=magic(8)`*, establish matrix* $\boldsymbol{B}$ *with suitable statements and see whether the results are correct or not.*

Solution Even row extraction of matrix $\boldsymbol{A}$ can easily be performed by

```
>> A=magic(8), B=A(2:2:end,:)
```

Exercise 2.12 *Please list all the positive integers such that it is a multiple of 11 and does not exceed 1000. Please also find all the integers which are multiples of 11 in the* $[3000, 5000]$ *interval.*

Solution The first item is 11, and the others can be generated with colon expression, with an increment of 11. The expected integers can be obtained with

```
>> i1=11:11:1000
```

The integers in the interval $[3000, 5000]$ can be extracted with the following statements

K27591_SM_Cover.indd 12 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

```
>> i2=11:11:5000; i2=i2(i2>=1000)
```

Exercise 2.13 *How many prime numbers are there in the interval* $[1, 1000000]$*? Please find the product of all the prime numbers in the interval. What is the number and how many digits are there? Measure the time elapsed in the evaluation.*

Solution There are 78498 prime numbers in the interval. The total number of the product has 433636 decimal digits, and it can be obtained in around 10 seconds.

```
>> length(primes(1000000)), tic, prod(primes(sym(1000000))); toc
   vpa(log10(ans))
```

Exercise 2.14 *It is known in Example 2.12 that* `gcd()` *and* `lcm()` *functions can be used to find the greatest common divisor and least common multiple of two entities only. Please write functions* `gcds()` *and* `lcms()` *such that arbitrary number of entities can be processed.*

Solution With the quantity `varargin`, the two functions can be written as follows. Both of the functions support arbitrary numbers of input arguments.

```
function k=gcds(varargin)
k=varargin{1}; for i=2:nargin, k=gcd(k,varargin{i}); end

function k=lcms(varargin)
k=varargin{1}; for i=2:nargin, k=lcm(k,varargin{i}); end
```

Having written the above functions, the least common multiple of 1, 2, 3, $\cdots$, 10, 11, 12 can be evaluated with the following statements, and the result is $K = 27720$.

```
>> K=lcms(1,2,3,4,5,6,7,8,9,10,11,12)
```

Exercise 2.15 *Implement the following piecewise function where* $\boldsymbol{x}$ *can be given by scalar, vectors, matrices or even other multi-dimensional arrays, the returned argument* $\boldsymbol{y}$ *should be the same size as that of* $\boldsymbol{x}$*. The parameters* h *and* D *are scalars.*

$$\boldsymbol{y} = f(\boldsymbol{x}) = \begin{cases} h, & \boldsymbol{x} > D \\ h/D\boldsymbol{x}, & |\ \boldsymbol{x}\ | \leqslant D \\ -h, & \boldsymbol{x} < -D. \end{cases}$$

Solution Two methods can be used, and the best one is with the use of the relationship expression in a clever way

```
>> y=h*(x>D) + h/D*x.*(abs(x)<=D) -h*(x<-D);
```

An alternative method is by the use of loops and condition structures

```
>> for i=1:length(x)
      if x(i)>D, y(i)=h;
      elseif abs(x(i))<=D, y(i)= h/D*x(i); else, y(i)=-h; end
   end
```

The structure of the latter statements are easy to understand, however the former is applicable not only for vector $\boldsymbol{x}$, but also to other data structures such as matrices or three-dimensional arrays.

Exercise 2.16 *A recursive formula is given by* $x_{n+1} = \dfrac{x_n}{2} + \dfrac{3}{2x_n}$*, with* $x_1 = 1$*. Please find*

K27591_SM_Cover.indd 13 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

a suitable number n, such that the terms after x_n approaches to a certain constant. The accuracy requirement is 10^{-14}. Please find also the constant.

Solution The constant is also referred to as steady-state value, it means that $||x_{k+1}-x_k|| \leqslant \epsilon$, where ϵ is the pre-specified small number, for instant, 10^{-14}. It is not difficult to find the suitable $k = 6$, and the steady-state value obtained is $x_{k+1} = 1.7321$. It can be seen that the sequence approaches to the irrational number $\sqrt{3}$. The recursive formula is with quite high efficiency, since only 6 terms are needed to approach to the irrational $\sqrt{3}$.

```
>> k=1; x0=1;
   while (1), x1=x0/2+3/(2*x0);
      if abs(x1-x0)<1e-14, break; else, k=k+1; x0=x1; end
   end
```

Exercise 2.17 *Please calculate* $S = \prod_{n=1}^{\infty}\left(1+\frac{2}{n^2}\right)$, *under precision requirement of* $\epsilon = 10^{-12}$.

Solution It seems that the product can be obtained with loop structure, and the result thus obtained is 9.5648. Please pay attention to the condition $|1 - p_1| \leqslant 10^{-10}$. It should not be written as $|p_1| \leqslant 10^{-12}$.

```
>> p=1;
   for n=1:10000, p1=(1+2/n^2); if abs(1-p1)<=1e-12, break; end
      p=p*p1;
   end
```

When checking the value of n, it is found that the $n = 10000$, which means that even all the terms are tested, the condition is not satisfied. To solve the accuracy problem, more terms must be tried, for instance, set the terms to 10000000000, or a larger number. While this method is rather time consuming. The actual n is $n = 1414230$, and $p = 9.5667$.

```
>> p=1;
   for n=1:100000000000000,
      p1=(1+2/n^2); if abs(1-p1)<=1e-12, break; end
      p=p*p1;
   end
```

Exercise 2.18 *It is known that*

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots.$$

Let $x = 1$, the following formula can be derived

$$\pi \approx 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \cdots\right).$$

Please calculate the approximate value of π, with precision requirement 10^{-6}.

Solution With the accumulation approach below, the approximate value of π can be found, and it is quite time consuming, in this example, 4 seconds are needed, since 500001 terms are needed to meet the requirement. If the error tolerance is set to an even smaller number, much more time is needed. The readers may try to change the `1e-6` term to `1e-8` and see what happens - 4 minutes on my computer.

```
>> s=0; s0=1; tic
```

K27591_SM_Cover.indd 14 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

```
for i=1:100000000,
   s1=1/(2*i-1)*s0; s0=-s0;
   if abs(s1)<1e-6, break; end, s=s+s1;
end
toc, S=4*s, i
```

Exercise 2.19 *Generate a* 100×100 *magic matrix, and find out the entities greater than 100, and substitute the entities by 0.*

Solution The magic matrix can be generated, and all the terms greater than 100 can be found, and set by force with the following statements

```
>> A=magic(100); i=find(A>100); A(i)=0;
```

Exercise 2.20 *Evaluate using numerical method the sum*

$$S = 1 + 2 + 4 + \cdots + 2^{62} + 2^{63} = \sum_{i=0}^{63} 2^i,$$

the use of vectorized form is suggested. Check whether accurate solutions can be found and why. Find the accurate sum using the symbolic computation methods. What happened in numerical and analytical solutions, if the number of terms is increased to 640.

Solution The following statements can be used to evaluate numerically the sum, however due to the limitations of the 64-bit `double` data type, the result $s_1 = 1.844674407370955 \times 10^{19}$ is not accurate.

```
>> s1=sum(2.^[0:63])
```

To solve the problem accurately, the symbolic data type should be used instead. The new statement should be

```
>> s2=sum(sym(2).^[0:63])
```

with $s_2 = 18446744073709551615$. One may even replace the term 63 by 1000 to evaluate accurately $\sum_{i=0}^{1000} 2^i$, which is not possible by numerical data types. The accurate result is

214301721437253464189685009812000362112280962341106721488750077674070210224987224498639675763139171625518934583510629365037429057138462808719691551493971496078691355496484619708421492101247422837559083643060929499671638825347975351183310878921541258291423929553730843353208596633052487736744113361387 51.

Exercise 2.21 *Please use two algorithms to solve the equation* $f(x) = x^2 \sin(0.1x+2) - 3 = 0$.

(i) **Bisection method**. *If in an interval* (a, b), $f(a)f(b) < 0$, *there will be at least one solution. Take the middle point* $x_1 = (b - a)/2$, *and based on the relationship of* $f(x_1)$ *and* $f(a)$, $f(b)$, *determine in which half interval there exists solutions. Middle point in the new half interval can then be taken. Repeat the process until the size of the interval is smaller than the pre-specified error tolerance* ϵ. *Find the solution with bisection method in interval* $(-4, 0)$, *with* $\epsilon = 10^{-10}$.

(ii) **Newton–Raphson method**. *Select an initial guess of* x_n, *the next approximation can be obtained with* $x_{n+1} = x_n - f(x_n)/f'(x_n)$. *If the two points are close enough, i.e.,* $|x_{n+1} - x_n| < \epsilon$, *where* ϵ *is the error tolerance. Find the solution with* $x_0 = -4$, *and* $\epsilon = 10^{-12}$.

Solution No matter which algorithm is adopted, the equation should be described as an anonymous function in MATLAB as

K27591_SM_Cover.indd 15 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

```
>> f=@(x)x.^2.*sin(0.1*x)-3;
```

(i) **Bisection algorithm**. Bisection algorithm can be used to solve the equation, and the function can be written as follows. A central point x in the interval can be obtained, and the value of the function can be compared with $f(a)$ and $f(b)$. If the signs are different, the values of a or b are updated, until finally a very small interval can be found, and the central point can be regarded as the root.

```
function x=p_math_2_18a(f,a,b,err)
if f(a)*f(b)>0,
  error('bisectional method failed since f(a)*f(b)>0.'); end
while abs(b-a)>err, x=(a+b)/2;
   if f(x)*f(a)<0, b=x; else, a=x; end
end
```

The root of the equation can be obtained as $x=3.1242$, and the value of the function at this point is $f(x)=-2.2\times10^{-11}$.

```
>> x=p_math_2_18a(f,0,4,1e-10), f(x)
```

(ii) **Gradient algorithm**: Since the derivative $f'(x)$ is needed, and it can be obtained as $f'(x)=2x\sin(x/10)+(x^2\cos(x/10))/10$, the function can be expressed with anonymous function

```
>> syms x; diff(x^2*sin(0.1*x)-3),
   f1=@(x)2*x.*sin(x/10)+(x.^2.*cos(x/10))/10;
```

Alternatively, gradient algorithm can be used to solve the equation

```
function x=p_math_2_18b(f,f1,x0,err)
while (1), x=x0-f(x0)/f1(x0);
   if abs(f(x))>err, x0=x; else, break; end
end
```

The following statements can be used to find the result $x=3.1242$, with $f(x)=1.83\times10^{-12}$.

```
>> x=p_math_2_18b(f,f1,-4,1e-10), f(x)
```

Exercise 2.22 *Write an M-function* `mat_add()` *with the* $\boldsymbol{A}=$`mat_add(`$\boldsymbol{A}_1,\boldsymbol{A}_2,\boldsymbol{A}_3,\cdots$`)` *syntax. It is required that arbitrary number of input arguments* $\boldsymbol{A}_i$ *are allowed.*

Solution With the use of `varargin`, the function below can be designed.

```
function A=mat_add(varargin)
A=0; for i=1:length(varargin), A=A+varargin{i}; end
```

The `try-catch` structure can further be used to solve the above problem.

```
function A=mat_add(varargin)
try
   A=0; for i=1:length(varargin), A=A+varargin{i}; end
catch, error(lasterr); end
```

K27591_SM_Cover.indd 16 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

Exercise 2.23 *A MATLAB function can be written whose syntax is*

$$\boldsymbol{v} = [h_1, h_2, h_m, h_{m+1}, \cdots, h_{2m-1}] \quad \textit{and} \quad \boldsymbol{H} = \texttt{myhankel}(\boldsymbol{v})$$

where the vector $\boldsymbol{v}$ *is defined, and out of it, the output argument should be an* $m \times m$ *Hankel matrix.*

Solution Many methods can be used to solve the above problem:

(i) The most straightforward method is the use of double loop structure to implement $H_{i,j} = h_{i+j-1}$ such that

```
function H=myhankel(v)
m=(length(v)+1)/2;
for i=1:m, for j=1:m, H(i,j)=v(i+j-1); end, end
```

(ii) For a certain column (or row), $\boldsymbol{a}_i = [h_i, h_{i+1}, \cdots, h_{i+m-1}]$. Thus single loop structure can be used to generate the Hankel matrix

```
function H=myhankel(v)
m=(length(v)+1)/2; for i=1:m, H(i,:)=v(i:i+m-1); end
```

(iii) Based on the existing `hankel()` function, one can write

```
function H=myhankel(v)
m=(length(v)+1)/2; H=hankel(v(1:m),v(m:end));
```

Exercise 2.24 *From matrix theory, it is known that if a matrix* $\boldsymbol{M}$ *is expressed as* $\boldsymbol{M} = \boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}\boldsymbol{B}^{\mathrm{T}}$*, where* $\boldsymbol{A}$*,* $\boldsymbol{B}$ *and* $\boldsymbol{C}$ *are the matrices of relevant sizes, the inverse of* $\boldsymbol{M}$ *can be calculated by the following algorithm*

$$\boldsymbol{M}^{-1} = \Big(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}\boldsymbol{B}^{\mathrm{T}}\Big)^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}\Big(\boldsymbol{C}^{-1} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{B}\Big)^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}^{-1}$$

The matrix inversion can be carried out using the formula easily. Suppose that there is a 5×5 *matrix* $\boldsymbol{M}$*, from which the three other matrices can be found.*

$$\boldsymbol{M} = \begin{bmatrix} -1 & -1 & -1 & 1 & 0 \\ -2 & 0 & 0 & -1 & 0 \\ -6 & -4 & -1 & -1 & -2 \\ -1 & -1 & 0 & 2 & 0 \\ -4 & -3 & -3 & -1 & 3 \end{bmatrix}, \quad \boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 4 \end{bmatrix},$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 2 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & -1 & 0 & 1 \end{bmatrix}.$$

Write the statement to evaluate the inverse matrix. Check the accuracy of the inversion. Compare the accuracy of the inversion method and the direct inversion method with `inv()` *function.*

Solution Based on the partition formula, the following function can be written

```
function Minv=part_inv(A,B,C)
Minv=inv(A)-inv(A)*B*inv(inv(C)+B'*inv(A)*B)*B'*inv(A);
```

For the given matrices, the following two methods can be used

K27591_SM_Cover.indd 17 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

```
>> M=[-1,-1,-1,1,0; -2,0,0,-1,0; -6,-4,-1,-1,-2;
      -1,-1,0,2,0;-4,-3,-3,-1,3];
   A=[1,0,0,0,0; 0,3,0,0,0; 0,0,4,0,0; 0,0,0,2,0; 0,0,0,0,4];
   B=[0,1,1,1,1; 0,2,1,0,1; 1,1,1,2,1; 0,1,0,0,1; 1,1,1,1,1];
   C=[1,-1,1,-1,-1; 1,-1,0,0,-1; 0,0,0,0,1; 1,0,-1,-1,0; 0,1,-1,0,1];
   M1=inv(M), % method 1, direct numerical solution
   M2=part_inv(A,B,C)   % method 2, with partition formula
   Ms=inv(sym(M)); e1=norm(double(Ms)-M1), e2=norm(double(Ms)-M2)
```

The appearance of $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ are the same, however, the precision might be different, since $e_1 = 1.5232\times10^{-16}, e_2 = 1.5271\times10^{-15}$. It can be concluded that normally when direct functions exist, it should be used, rather than using any other indirect methods, to avoid accumulative errors.

Exercise 2.25 *Please generate the first 300 terms of the extended Fibonacci sequence* $T(n) = T(n-1) + T(n-2) + T(n-3), n = 4, 5, \cdots$, *with* $T(1) = T(2) = T(3) = 1$.

Solution The best way to solve the problem is to use the loop structure, and it is found that the last term is 58787881866910273134690475472636892696354699930587907559253479117 85909345498275.

```
>> T=sym([1 1 1]);
   for i=4:300, T(i)=T(i-1)+T(i-2)+T(i-3); end
```

Please note that recursive structure is not recommended, and in fact, it cannot be used in solving such a problem.

Exercise 2.26 *Consider the following iterative model*

$$\begin{cases} x_{k+1} = 1 + y_k - 1.4x_k^2 \\ y_{k+1} = 0.3x_k \end{cases}$$

with initial conditions $x_0 = 0$, $y_0 = 0$. *Write an M-function to evaluate the sequence* x_i, y_i. *30000 points can be obtained by the function to construct the* $\boldsymbol{x}$ *and* $\boldsymbol{y}$ *vectors. The points can be expressed by a dot, rather than lines. In this case, the so-called* Hénon attractor *can be drawn.*

Solution Loop structure can be used to implement the recursive formula, and the Hénon attractor can be drawn as shown in Figure 2.1. Note that, the option `'.'` should be used to indicate the sequence.

```
>> n=30000; x=zeros(1,n); y=x;
   for i=1:n-1, x(i+1)=1+y(i)-1.4*x(i)^2; y(i+1)=0.3*x(i); end
   plot(x,y,'.')
```

Exercise 2.27 *The well-known* Mittag–Leffler function *is defined as*

$$f_\alpha(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + 1)},$$

where $\Gamma(x)$ *is* Γ*-function which can be evaluated with* `gamma(x)`. *Write an M-function with syntax* $\boldsymbol{f}$ = `mymittag(`α`,`$\boldsymbol{z}$`,`ϵ`)`, *where* ϵ *is the error tolerance, with default value of* $\epsilon = 10^{-6}$. *Argument* $\boldsymbol{z}$ *is a numeric vector. Draw the curves for Mittag–Leffler functions with* $\alpha = 1$ *and* $\alpha = 0.5$.

Solution The `while` loop structure can be used for the problem. If the norm of the

K27591_SM_Cover.indd 18 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

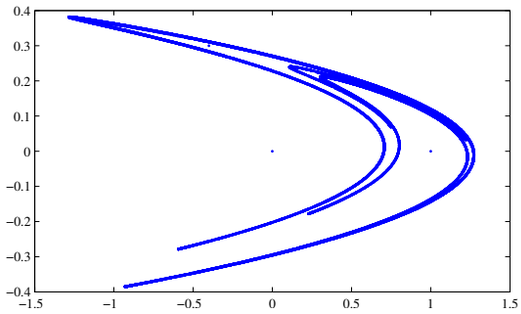


FIGURE 2.1: Hénon attractor

increment is smaller than the pre-specified small value ϵ, the loop can be terminated, and the result can be returned.

```
function f=mymittag(a,z,err)
f=0; k=0; if nargin==2, err=1e-6; end
while (1)
   df=z.^k/gamma(a*k+1);
   if norm(df)>err, f=f+df; k=k+1; else, break; end
end
```

With the above function, the two Mittag–Leffler function curves can be obtained as shown in Figure 2.2. The curve for $\alpha = 1$ is equivalent to exponential function e^z. Better Mittag–Leffler function implementation will be given in Chapter 8 of the book.

```
>> z=0:0.01:1; y1=mymittag(1,z,1e-6);
   y2=mymittag(0.5,z,1e-6); plot(z,y1,'-',z,y2,'--')
```

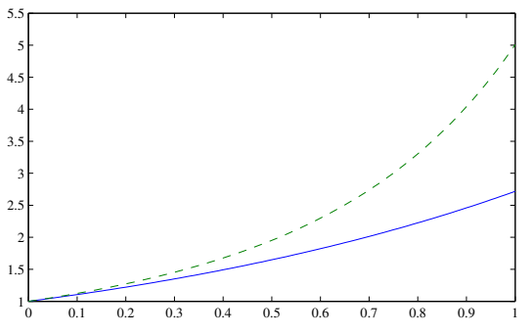


FIGURE 2.2: Two Mittag–Leffler function curves

Exercise 2.28 Chebyshev polynomials *are mathematically defined as*

$$T_1(x) = 1,\ T_2(x) = x,\ T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x),\ n = 3, 4, 5, \cdots.$$

Please write a recursive function to generate a Chebyshev polynomial, and compute $T_{10}(x)$. *Please write a more efficient function as well to generate Chebyshev polynomials, and find* $T_{30}(x)$.

Solution Two MATLAB functions can be written to generate Chebyshev polynomials,

K27591_SM_Cover.indd 19 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

while in the first function, recursive structure is used, and in the second one, loop structure is used. It is recommended to use the second function, since the recursive structure is time consuming and cannot be used for large N.

```
function T=chebyshev_poly(n,x)
if n==0, T=1; elseif n==1, T=x;
else, T=2*x*chebyshev_poly(n-1,x)-chebyshev_poly(n-2,x); end

function T=chebyshev_poly1(n,x)
T0=1; T1=x; for k=3:n+1, T=collect(2*x*T1-T0); T0=T1; T1=T; end
```

The Chebyshev polynomial $T_{30}(x)$ can be obtained with

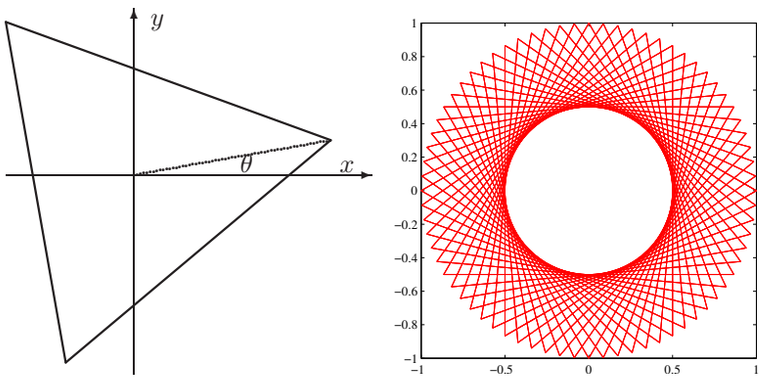
```
>> syms x; T=chebyshev_poly1(30,x), collect(T)
```

Exercise 2.29 *A regular triangle can be drawn by MATLAB statements easily. Use the loop structure to design an M-function that, in the same coordinates, a sequence of regular triangles can be drawn, each by rotating a small angle, for instance,* $5°$*, from the previous one.*

Solution To rotate counter-clockwise an equilateral triangle by the angle θ, the new triangle can be illustrated as shown in Figure 2.3 (a). The critical points of the new triangle are respectively $(\cos\theta, \sin\theta)$, $(\cos(\theta+120°), \sin(\theta+120°))$ and $(\cos(\theta+240°), \sin(\theta+240°))$, then back to point $(\cos\theta, \sin\theta)$. The triangle can be drawn easily. Increment the angle θ continuously and draw in a loop a set of triangles, the resulted graphical display is shown in Figure 2.3 (b), with the command

```
>> draw_triangles(5,'r') % 5 and 'r' for increment angle and color
```

and the M-function `draw_triangles()` is listed below



(a) sketch of the rotation (b) final results

FIGURE 2.3: Graphical display of a set of triangles

```
function draw_triangles(delta,col)
t=[0,120,240,0]*pi/180; xxx=[]; yyy=[];
for i=0:delta:360
   tt=i*pi/180; xxx=[xxx; cos(tt+t)]; yyy=[yyy; sin(tt+t)];
end
plot(xxx',yyy',col), axis('square')
```

K27591_SM_Cover.indd 20 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

Selecting the increment to other values such as $\Delta\theta = 2, 1, 0.1$, one may further observe the results.

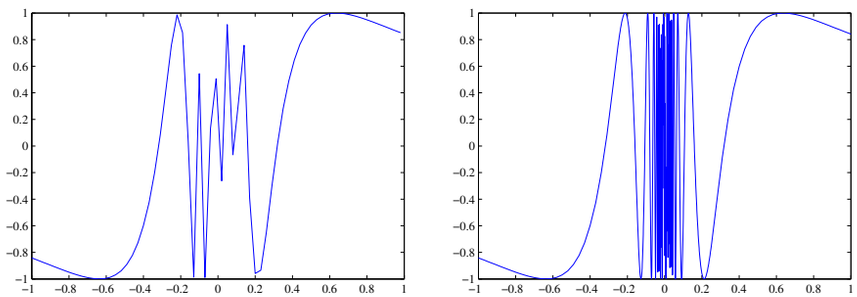
Exercise 2.30 *Select suitable step-sizes and draw the function curve for* $\sin(1/t)$, $t \in (-1, 1)$.

Solution In ordinary graphics mode, when a step-size of 0.03 is used, the curve of the function is shown in Figure 2.4 (a). However the curve is harsh.

```
>> t=-1:0.03:1; y=sin(1./t); plot(t,y)
```

If variable step-size is used, the curves are shown in Figure 2.4 (b).

```
>> t=[-1:0.03: -0.25, -0.248:0.001:0.248, 0.25:.03:1];
   y=sin(1./t); plot(t,y)
```



(a) fixed step-size (b) variable step-size

FIGURE 2.4: The curves of $\sin(1/t)$ under different step-sizes

It can be concluded from the example that, the curves obtained should be verified, before it can be put into practical used.

Exercise 2.31 *For suitably assigned ranges of* θ*, draw polar plots for the following functions.*

(i) $\rho = 1.0013\theta^2$, *(ii)* $\rho = \cos(7\theta/2)$, *(iii)* $\rho = \sin(\theta)/\theta$, *(iv)* $\rho = 1 - \cos^3(7\theta)$.

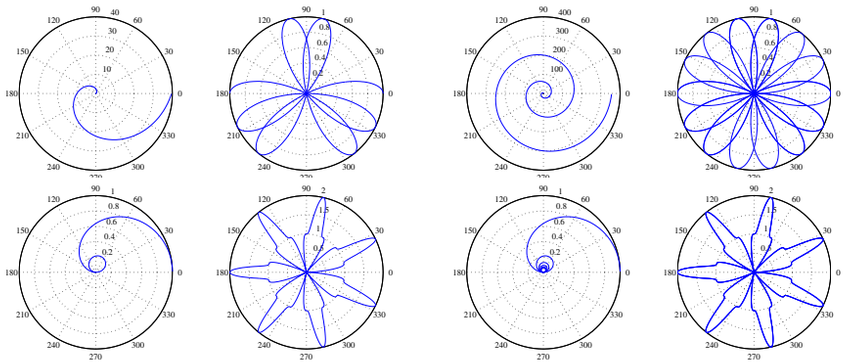
Solution It seems that drawing polar plot is an easy task. However one should be very careful in verifying the results by choosing different θ intervals. Also dot operation of vectors must be used. Comparing the plots for different θ ranges shown in Figures 2.5 (a) and (b). It can be seen that when the range of θ is small, some of the polar plots may not be complete.

```
>> t=0:0.01:2*pi; subplot(221), polar(t,1.0013*t.^2),  % (i)
   subplot(222), polar(t,cos(7*t/2)) % (ii)
   subplot(223), polar(t,sin(t)./t)                    % (iii)
   subplot(224), polar(t,1-(cos(7*t)).^3)              % (iv)
   figure; t=0:0.01:6*pi; % repeat the previous commands, get figure (b)
```

Exercise 2.32 *Please draw the curves of* $x \sin x + y \sin y = 0$ *for* $-50 \leqslant x, y \leqslant 50$.

Solution This is an implicit function, and the plot of the function can be drawn easily with the following statements, as shown in Figure 2.6.

K27591_SM_Cover.indd 21 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am



(a) $\theta \in (0, 2\pi)$ (b) $\theta \in (0, 6\pi)$

FIGURE 2.5: Polar plots for different θ ranges

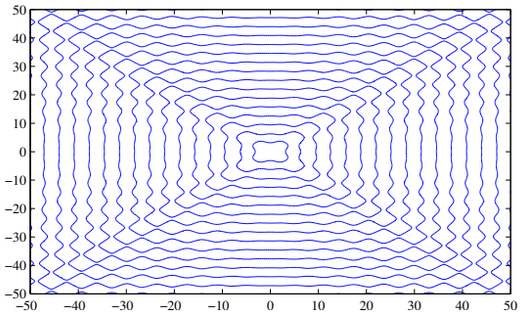


FIGURE 2.6: Curve of the implicit function

```
>> ezplot('x*sin(x)+y*sin(y)',[-50,50])
```

Exercise 2.33 *Find the solutions to the following simultaneous equations using graphical methods and verify the solutions.*

$$(i)\ \begin{cases} x^2+y^2=3xy^2 \\ x^3-x^2=y^2-y, \end{cases} \qquad (ii)\ \begin{cases} \mathrm{e}^{-(x+y)^2+\pi/2}\sin(5x+2y)=0 \\ (x^2-y^2+xy)\mathrm{e}^{-x^2-y^2-xy}=0. \end{cases}$$

Solution (i) The two equations can all be expressed by implicit function drawing command `ezplot()`, and the intersections are the solutions of the simultaneous equations, as shown in Figure 2.7 (a). One may zoom the plots and find more accurate values of the intersections.

```
>> ezplot('x^2+y^2-3*x*y^2'); hold on, ezplot('x^3-x^2=y^2-y')
```

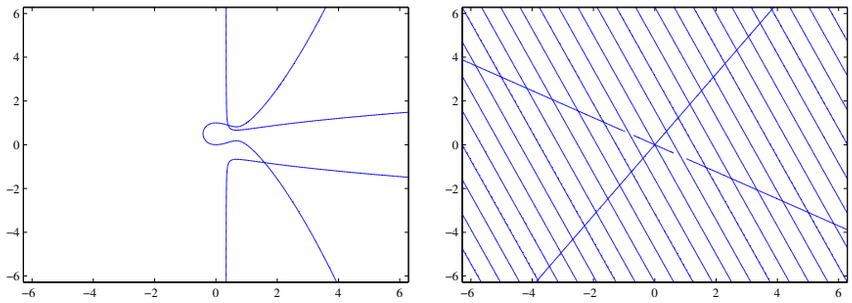
(ii) The equations can be drawn with the following statements, and the curves of the equations are as shown in Figure 2.7 (b).

```
>> ezplot('exp(-(x+y)^2+pi/2)*sin(5*x+2*y)'), hold on
   ezplot('(x^2-y^2+x*y)*exp(-x^2-y^2-x*y)')
```

Exercise 2.34 *Please save a* 100×100 *magic matrix into an Excel file.*

Solution The matrix can be generated and saved in the Excel file, with the following statements.

K27591_SM_Cover.indd 22 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am



(a) equations in (i) (b) equations in (ii)

FIGURE 2.7: Graphical interpretations of the solutions

```
>> A=magic(100); xlswrite('myfile.xls',A)
```

Exercise 2.35 *Assume that the power series expansion of a function is*

$$f(x) = \lim_{N\to\infty} \sum_{n=1}^{N} (-1)^n \frac{x^{2n}}{(2n)!}.$$

If N is large enough, power series $f(x)$ converges to a certain function $\hat{f}(x)$. Please write a MATLAB program that plots the function $\hat{f}(x)$ in the interval $x \in (0, \pi)$. Observe and verify what function $\hat{f}(x)$ is.

Solution From the general term $g_n(x) = x^{2n}/(2n)!$, it is easily found that the accumulative relationship is $g_{n+1}(x)/g_n(x) = -x^2/(2n(2n-1))$, therefore selecting an error tolerance $\epsilon = 10^{-10}$, with loop structure, the function can be evaluated with

```
>> x=linspace(0,pi,50); y=zeros(size(x));
   e=1e-10; dg=ones(size(x)); n=0;
   while norm(dg)>e
      n=n+1; dg=-dg.*x.^2/2/n/(2*n-1); y=y+dg;
   end
   plot(x,y)
```

The plot obtained is as shown in Figure 2.8 and it seems that the actual function is $y(x) = \cos x - 1$.

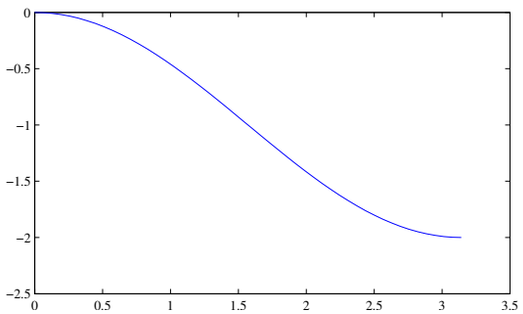


FIGURE 2.8: Obtained curve of the function

K27591_SM_Cover.indd 23 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

Exercise 2.36 *Draw the 3D surface plots for the functions xy and $\sin xy$ respectively. Also draw the contours of the functions. View the 3D surface plot from different angles, especially with orthographic views.*

Solution The following commands can be used to draw the 3D surface and contour lines of the functions. The function `view()` can be used to change view points and also one can rotate the 3D surfaces manually. The surfaces and contours of the two functions can be obtained as shown in Figure 2.9. It is interesting to note that when x and y are small — not necessarily approach zero, the values of the functions xy and $\sin xy$ are similar.

```
>> [x,y]=meshgrid(-1:.1:1); z1=x.*y; z2=sin(z1);
   subplot(211), surf(x,y,z1), subplot(212), contour(x,y,z1,30)
   figure;
   subplot(211), surf(x,y,z2), subplot(212), contour(x,y,z2,20)
```

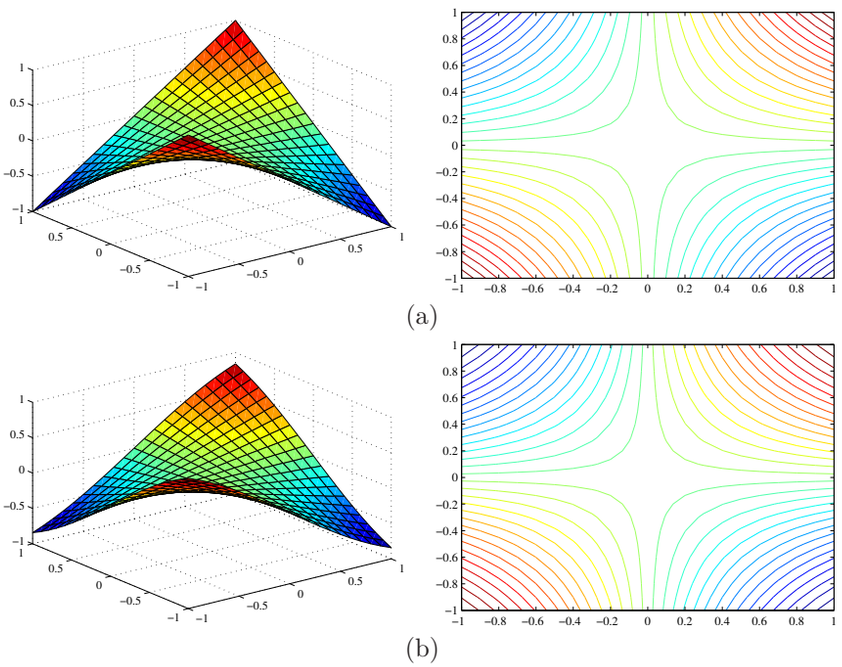


FIGURE 2.9: Surfaces and contours

Exercise 2.37 *Please draw 2D and 3D Lissajous figures under the parametric equations $x = \sin t, y = \sin at$, and $z = \sin bt$, for different parameters a and b, where, please try the following rational and irrational parameters and see what may happen.*

(i) $a = 1/2$, $b = 1/3$, *(ii)* $a = \sqrt[8]{2}$, $b = \sqrt{3}$.

Solution (i) For rational a, no matter how large the range of t is, neat phase plot shown in Figure 2.10 can be obtained.

```
>> t=0:0.01:10000; a=1/2; b=1/3; plot(sin(a*t),sin(t*b))
```

(ii) If a and/or b are irrational numbers, when the range of t is selected very large, the phase plot will fill the whole $(-1, 1)$ square in the x-y plane. To have a better understanding of the process, it is strongly recommended to use `comet()` function to replace `plot()`.

K27591_SM_Cover.indd 24 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

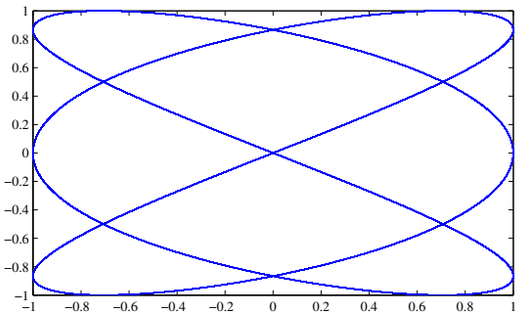


FIGURE 2.10: Phase plot

```
>> a=2^(1/8); b=sqrt(3); plot(sin(a*t),sin(t*b))
```

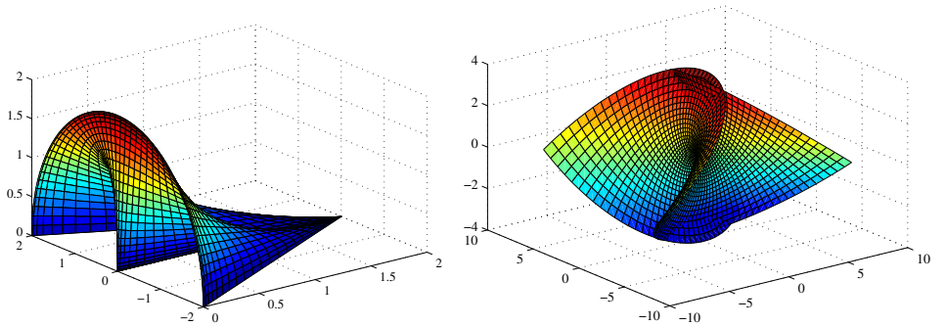
Exercise 2.38 *For the parametric equations [1], please draw the surfaces*

(i) $x = 2\sin^2 u\cos^2 v, y = 2\sin u\sin^2 v, z = 2\cos u\sin^2 v, -\pi/2 \leqslant u, v \leqslant \pi/2,$

(ii) $x = u - \dfrac{u^3}{3} + uv^2,\ y = v - \dfrac{v^3}{3} + vu^2,\ z = u^2 - v^2,\ -2 \leqslant u, v \leqslant 2.$

Solution The parametric equations can be drawn directly with the following statements, as shown in Figure 2.11 (a).

```
>> syms u v; x=2*sin(u)^2*cos(v)^2; y=2*sin(u)*sin(v)^2;
   z=2*cos(u)*sin(v)^2; ezsurf(x,y,z,[-pi/2,pi/2,-pi/2,pi/2])
```



(a) surface in (i) (b) surface in (ii)

FIGURE 2.11: Graphical interpretations of the solutions

(ii) Similarly, the parametric equations can be drawn easily with the following statements, as shown in Figure 2.11 (b).

```
>> x=u-u^3/3+u*v^2; y=v-v^3/3+v*u^2; z=u^2-v^2;
   ezsurf(x,y,z,[-2,2,-2,2])
```

Exercise 2.39 *A vertical cylinder can be described by parametric equation* $x = r\sin u, y = r\cos u, z = v$*, along* z *axis, with* r *the radius. If* x *and* z *are swopped, the cylinder can be*

K27591_SM_Cover.indd 25 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:36 am

represented along x axis. Please draw several cylinders with different radii and directions together in the same coordinate.

Solution The cylinders can be drawn with the following statements, as shown in Figure 2.12.

```
>> syms u v; r=1; x=r*sin(u); y=r*cos(u); z=v;
   ezsurf(x,y,z,[-2,2]), hold on
   x=0.5*sin(u); y=v; z=0.5*cos(u); ezsurf(x,y,z,[-4,4])
```

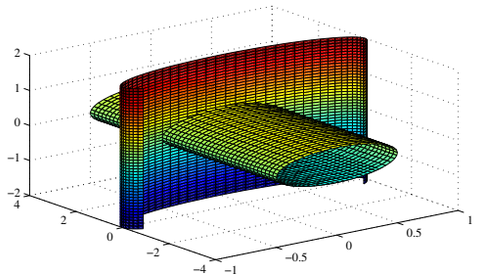


FIGURE 2.12: The intersection of the two cylinders

Exercise 2.40 *Please draw a cone, whose top is at (0,0,2), and bottom at the plane $z = 0$, with a radius of 1.*

Solution The edge of the cone can be calculated first, and the cone can be drawn with the following statements, as shown in Figure 2.13.

```
>> z0=0:0.1:1; r=1-z0; [x,y,z]=cylinder(r,100); surf(x,y,2*z)
```

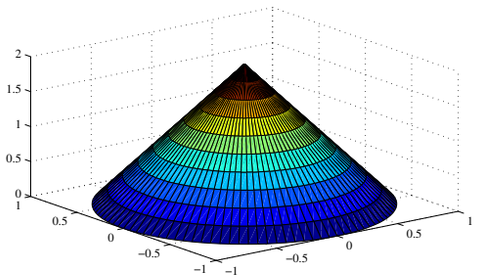


FIGURE 2.13: The surface of the cone

Exercise 2.41 *In graphics command, there is a trick in hiding certain parts of the plot. If the function values are assigned to* `NaN`*s, the point on the curve or the surface will not be shown. Draw first the surface plot of the function $z = \sin xy$. Then cut off the region that satisfies $x^2 + y^2 \leqslant 0.5^2$.*

Solution The mesh grid data of a rectangular region can be generated first and the function values can be calculated. Then find all the points in the region satisfying $x^2+y^2 \leqslant 0.5^2$, and set the values to `NaN`'s. The 3D surface of the given function excluding the region, shown in Figure 2.14.

K27591_SM_Cover.indd 26 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:37 am

```
>> [x,y]=meshgrid(-1:.1:1); z=sin(x.*y);
   ii=find(x.^2+y.^2<=0.5^2); z(ii)=NaN; surf(x,y,z)
```

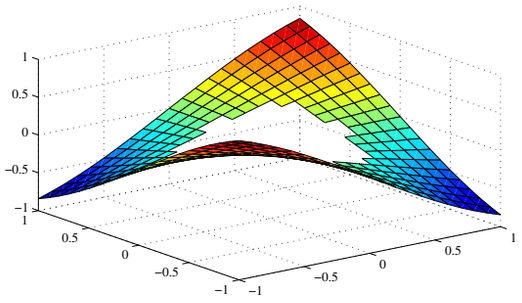


FIGURE 2.14: 3D surface with a region cut off

Exercise 2.42 *Please draw the 3D surface of* $f(x,y)=\dfrac{\sin\sqrt{x^2+y^2}}{\sqrt{x^2+y^2}}$ *for* $-8\leqslant x,y\leqslant 8$.

Solution For simplicity, it is better to use the function `ezsurf()` function to draw the surface plot, as shown in Figure 2.15.

```
>> ezsurf('sin(sqrt(x^2+y^2))/sqrt(x^2+y^2)',[-8,8])
```

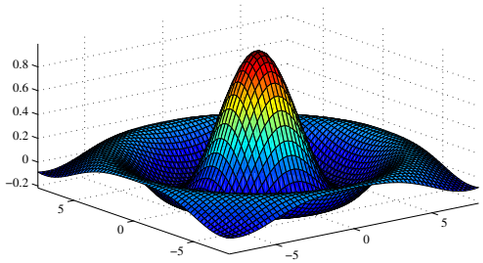


FIGURE 2.15: The 3D surface plot

Exercise 2.43 *Lambert W function is a commonly used special function, its mathematical form is* $W(z)\mathrm{e}^{W(z)}=z$. *Please draw the curve of the function.*

Solution Denote $y=W(z)$, the Lambert W function can be written as an equation of y, where $y\mathrm{e}^y=x$, where $x=z$ is the independent variable. In this case, the curve of Lambet W function can be drawn with the following command, as shown in Figure 2.16.

```
>> ezplot('y*exp(y)=x')
```

Exercise 2.44 *Draw the surface plot and contour plots for the following functions. Draw also with the functions* `surfc()`, `surfl()`, *and* `waterfall()`, *and observe the results.*

(i) $z=xy$, *(ii)* $z=\sin x^2y^3$, *(iii)* $z=\dfrac{(x-1)^2y^2}{(x-1)^2+y^2}$, *(iv)* $z=-xy\,\mathrm{e}^{-2(x^2+y^2)}$.

K27591_SM_Cover.indd 27 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:37 am

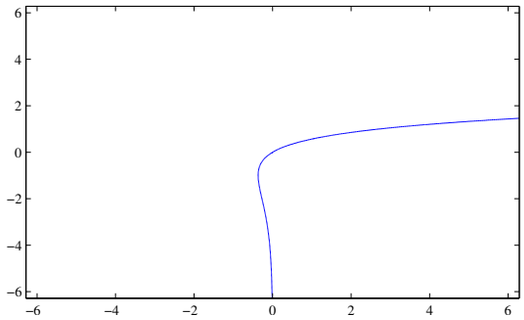


FIGURE 2.16: The curve of Lambert W function

Solution Take (ii) for example. Mesh grid can be generated first in the interested area, for instance, $(-1, 1)$, and the value of the function can be computed, and various 3D plots can be obtained as shown in Figure 2.17.

```
>> [x y]=meshgrid(-1:0.1:1); z=sin(x.^2.*y.^3);
   subplot(221), surf(x,y,z), subplot(222), surfl(x,y,z),
   subplot(223), surfc(x,y,z), subplot(224), waterfall(x,y,z),
```

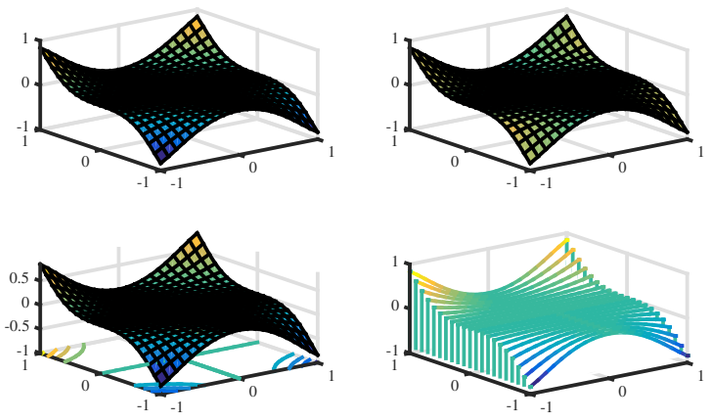


FIGURE 2.17: Different representation of the 3D surface

Exercise 2.45 *Please draw the surface of the three-dimensional implicit function*

$$(x^2 + xy + xz)\mathrm{e}^{-z} + z^2yx + \sin(x + y + z^2) = 0,$$

Solution The third-party `ezimplot3()` function can be used to draw the surface of the implicit function, as shown in Figure 2.18.

```
>> ezimplot3('(x^2+x*y+x*z)*exp(-z)+z^2*y*x+sin(x+y+z^2)')
```

Exercise 2.46 *Please draw the two following two surfaces and observe the intersections*

$$x^2 + y^2 + z^2 = 64,\ y + z = 0.$$

K27591_SM_Cover.indd 28 Scientific Computing with MATLAB 2nd edition Xue Solutions Manual 09/04/16 10:37 am

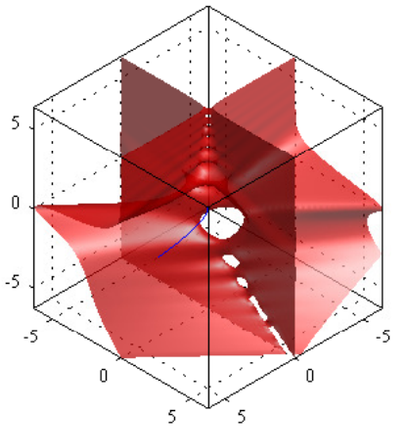


FIGURE 2.18: The 3D implicit function curve

Solution The two implicit functions can be drawn in the same coordinate, with the following statements, as shown in Figure 2.19.

```
>> ezimplot3('x^2+y^2+z^2-64'), ezimplot3('y+z')
```

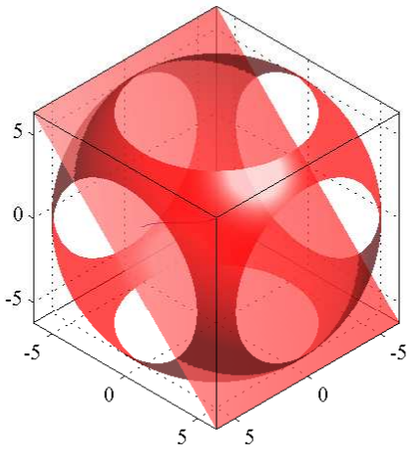


FIGURE 2.19: The 3D implicit function curve

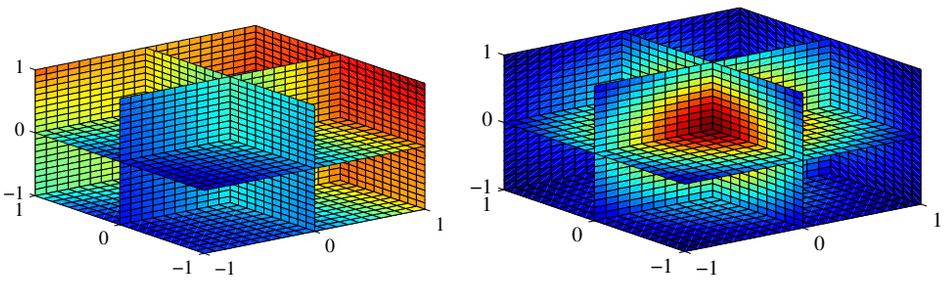
Exercise 2.47 *Please draw the sliced volume visualization for the following functions*

(i) $V(x,y,z) = \sqrt{\mathrm{e}^{x}+\mathrm{e}^{(x+y)-xy}+\mathrm{e}^{(x+y+z)/3-xyz}}$, *(ii)* $V(x,y,z) = \mathrm{e}^{-x^2-y^2-z^2}$.

Solution The 3D grid data can be generated first, and the values of the function can

be evaluated with dot operations. Volume visualization with slices from the data can be performed easily with the following statements, as shown in Figure 2.20.

```
>> [x y z]=meshgrid(-1:0.1:1);
   V=sqrt(exp(x)+exp((x+y)-x.*y)+exp((x+y+z)/3-x.*y.*z));
   slice(x,y,z,V,[0 1],[0 1],[-1,0])
   figure; V=exp(-x.^2-y.^2-z.^2); slice(x,y,z,V,[0 1],[0 1],[-1,0])
```



(a) function in (i) (b) function in (ii)

FIGURE 2.20: Volume visualization with slices

Alternatively, with the `vol_visual4d()` interface designed in the book, the volume visualization with standard slices for function in (ii) can be obtained directly, as shown in Figure 2.21.

```
>> vol_visual4d(x,y,z,V)
```

The slices can be adjusted easily with the controls in the interface.

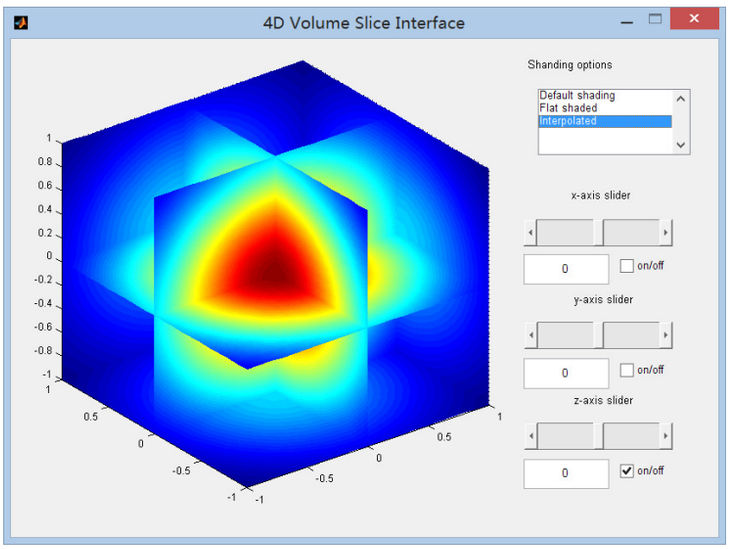


FIGURE 2.21: Slices in the example

Chapter 2

Fundamentals of MATLAB Programming

Scientific Computing with MATLAB, 2nd Edition

CRC/Taylor & Francis Press

Chinese version by Tsinghua University Press

PPT by Wenbin Dong and Jun Peng, Northeastern University, PRC

Proofread by Dingyu Xue & YangQuan Chen

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Chapter 2 Fundamentals of MATLAB Programming



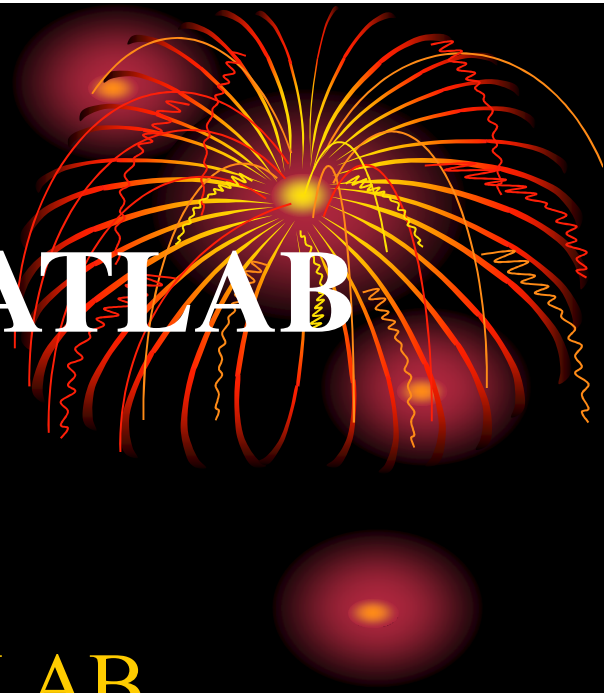
- Fundamentals of MATLAB Programming
- Fundamental Mathematical Calculations
- Flow Control Structures of MATLAB Language
- Writing and Debugging MATLAB Functions
- Two-, Three- Four-dimensional Graphics

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Advantages of MATLAB

- MATLAB has the following advantages:
  - Clarity and high efficiency
  - Scientific computation, covers almost all the useful topics in math and engineering
  - Graphics facilities
  - Comprehensive toolboxes and block-sets, designed by experts of almost all disciplines
  - Powerful simulation facilities, unite the sub-systems of different domains together

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual



2.1 Fundamentals of MATLAB Programming

- Variables and constants in MATLAB
- Data structures
- Basic structure of MATLAB
- Colon expressions and sub-matrices extraction

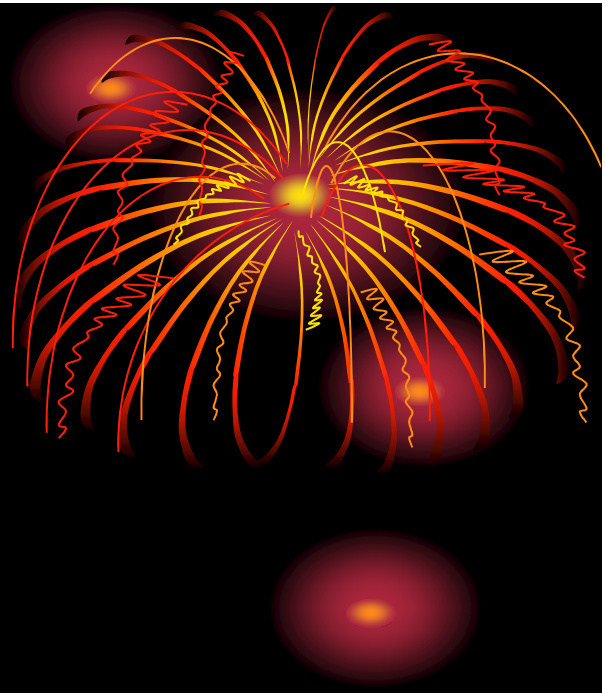
Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.1.1 Variables and Constants in MATLAB



- Variables in MATLAB
  - Starting with a letter followed by other characters:
  - Case-sensitive: **Abc ≠ABc**
  - Valid names: MYvar12, MY_Var12 and MyVar12_
  - Invalid variable names: 12MyVar, _MyVar12
- Constants in MATLAB:
  - **`eps, i, j, pi, NaN, Inf, i=sqrt(-1)`**
  - **`lastwarn, lasterr`**

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual



2.1.2 Data structures

- Double-precision data type
- Symbolic data type
- Other data types

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Double-precision Data Types

- IEEE standard, 64 bits (8 bytes)
  - 11 bits for exponential
  - 52 bits for numerical
  - 1 sign bit.



- Use double() to convert other types to double.
- Data range: 15 decimal digits

$$-1.7 \times 10^{308} \text{ to } 1.7 \times 10^{308}$$

- Other data types:
  - `int8(), int16(), int32(), uint16(), uint32()`

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Symbolic Data Type

- Usually used in formula derivations and analytical solutions
  - variable declaration

```
syms var_list var_props
```

  - Data type conversion $B = \texttt{sym}(A)$
  - display the symbolic variables in any precision

$\texttt{vpa}(A)$, $\texttt{vpa}(A, n)$

✓The default value: 32 decimal digits.

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

More on Symbolic Variables

- More data types: integer
- Conditions specification
  - assume function

```
>> syms t x; assume(t<=4); assume(x~=2)
```

  - assumeAlso function

```
>> syms x real; assume(x>=-1);
   assumeAlso(x<5)
```

$$-1 \le x < 5$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.1 Symbolic Variables

- Please declare a positive symbolic integer k such that it is a multiple of 13, and it does not exceed 3000
  - Not possible in earlier versions
  - How to declare $3000/13 \approx 230.7$



```
>> assume(k1<=230); assumeAlso(k1>0);
   assumeAlso(k1,'integer'); k=13*k1
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.2 Value of π

- Display the first 300 digits of π.
- MATLAB command



```
>> vpa(pi,300)
```

- One may further increase the number of digits to display.
- For extremely large number of digits, process may be slow

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

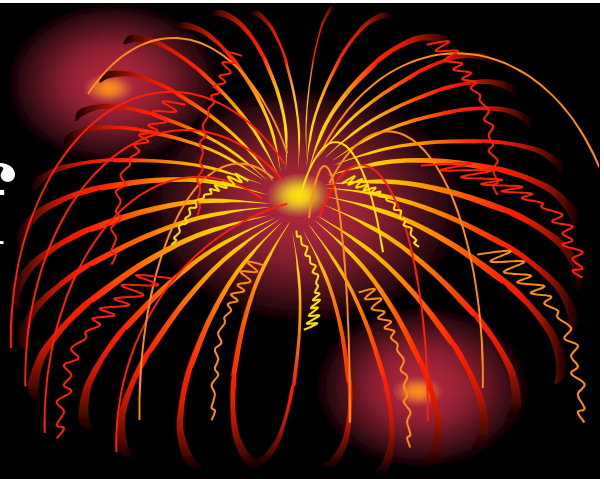
Other Data Types



- Strings: used to store messages. Single quotes
- Multi-dimensional arrays:
  - a direct extension of matrices with multiple indices.
- Cell arrays:
  - to put a set of data of different types under a single variable, expressed by { }.
- Classes and objects:
  - used in Object-Oriented Programming.

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.1.3 Basic Structures of MATLAB



- Direct assignment
  - The basic structure of this type of statement is

    ```
    variable = expression
    ```

  - A semicolon can prevent the results from display.
  - Reserved variable: `ans`, store the result of the latest statements without left-hand-variable
- Function call, returns many arguments

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.3 Matrix Input

- Enter matrix into MATLAB

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{bmatrix}$$

- Commands



```
>> A=[1,2,3; 4 5,6; 7,8 0]
```

- Other commands



```
>> A=[1,2,3; 4 5,6; 7,8 0];
   A=[[A; [1 2 3]], [1;2;3;4]];
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.4 Complex Matrix

- Enter complex matrix into MATLAB

$$B = \begin{bmatrix} 1+j9 & 2+j8 & 3+j7 \\ 4+j6 & 5+j5 & 6+j4 \\ 7+j3 & 8+j2 & 0+j1 \end{bmatrix}$$

- MATLAB commands



```
>> B=[1+9i,2+8i,3+7i;
      4+6i 5+5i,6+4i; 7+3i,8+2i 1i]
```

- Things to avoid:



```
>> B=[1 +9i,2+8i,3+7j;
      4+6j 5+5i,6+4i; 7+3i,8+2j 1i]
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Function Call Statements

- Function call statement

```
[returned_arguments]=
        function_name(input_arguments)
```

- Function call examples

```
[U  S  V]=svd(X)
```

- One function may be called in different ways
  - Built-in functions, *.m functions,
  - Anonymous functions, inline functions
  - Overload functions

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.1.4 Colon Expressions and Sub-matrices Extraction



- Colon expression is an effective way in defining row vectors.

$$v = s_1 : s_2 : s_3$$

- Start value s_1, increment s_2 and final value s_3
  - default increment: 1, when s_2 is missing
  - Value of s_3 not necessarily included in v

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.5 Make Vectors

- For different increments, establish vectors for $t \in [0, \pi]$

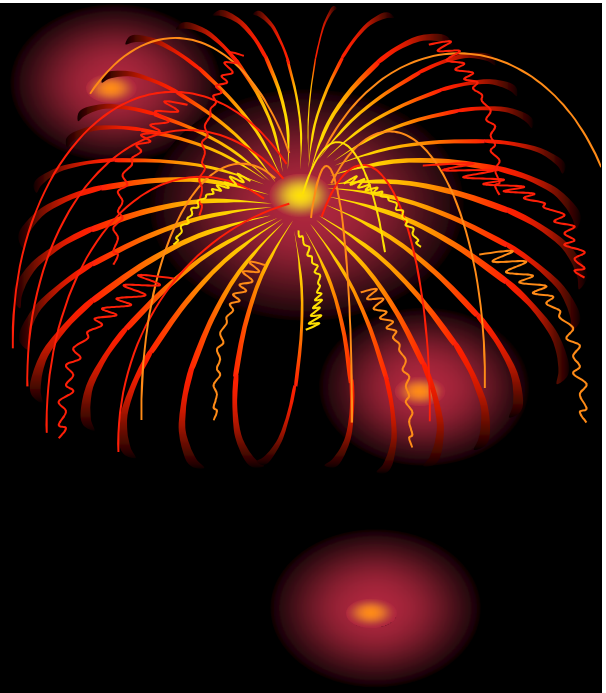
```
>> v1=0: 0.2: pi
>> v2=0: -0.1: pi
>> v3=0:pi
>> v4=pi:-1:0
```

- To include π

```
>> v5=linspace(0,pi,200);
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Sub-matrix Extraction



- Basic format

$$B=A(v_1,v_2)$$

- Arguments
  - v_1 numbers of the rows
  - v_2 numbers of the columns
  - :, all the columns or rows, depending on the position of it
  - Key word end

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.6 Sub Matrices

- Different sub-matrices can be extracted from the given matrix

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{bmatrix}$$

- MATLAB command



```
>> A=[1,2,3; 4,5,6; 7,8,0];
   B1=A(1:2:end, :)
   B2=A([3,2,1],[1 1 1])
   B3=A(:,end:-1:1)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.2 Fundamental Mathematical Calculations

- Algebraic operations of matrices
- Logic operations of matrices
- Relationship operations of matrices
- Simplifications and presentations of analytical results
- Basic number theory computations

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

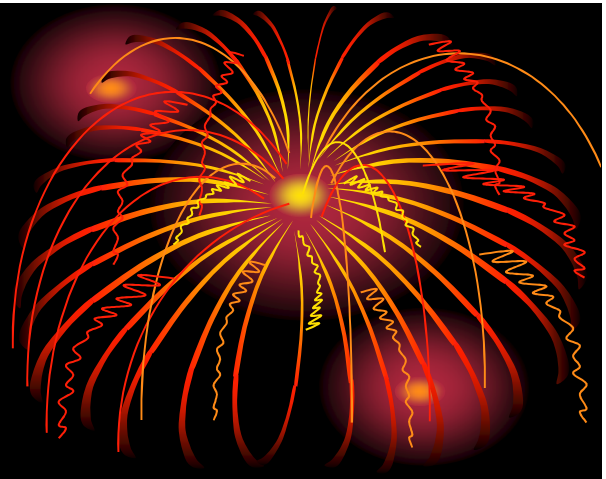
2.2.1 Algebraic operations of matrices



- Matrix transpose
- Matrix addition and subtraction
- Matrix multiplications
- Matrix divisions
- Matrix flip and rotations
- Matrix power
- Matrix dot operations

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix transpose



- Matrix representation:
  - Matrix A, n rows and m columns, is referred to as an $n \times m$ matrix
- Hermitian transpose $C = A'$

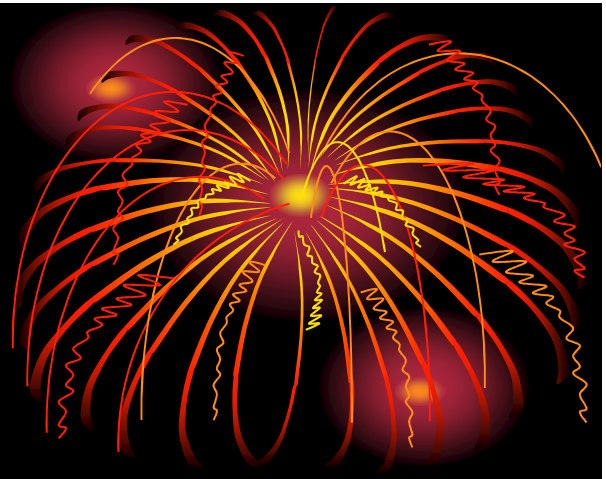
$$C = A^{\mathrm{H}}, c_{ji} = a_{ij}^{*},\ i = 1, \cdots, n,\ j = 1, \cdots, m$$

- Simple transpose $C = A.'$

$$B = A^{\mathrm{T}}, b_{ji} = a_{ij},\ i = 1, \cdots, n,\ j = 1, \cdots, m$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix Addition and Subtraction



- Mathematical representations

$$c_{ij} = a_{ij} \pm b_{ij},\ i = 1, \cdots, n,\ j = 1, \cdots, m$$

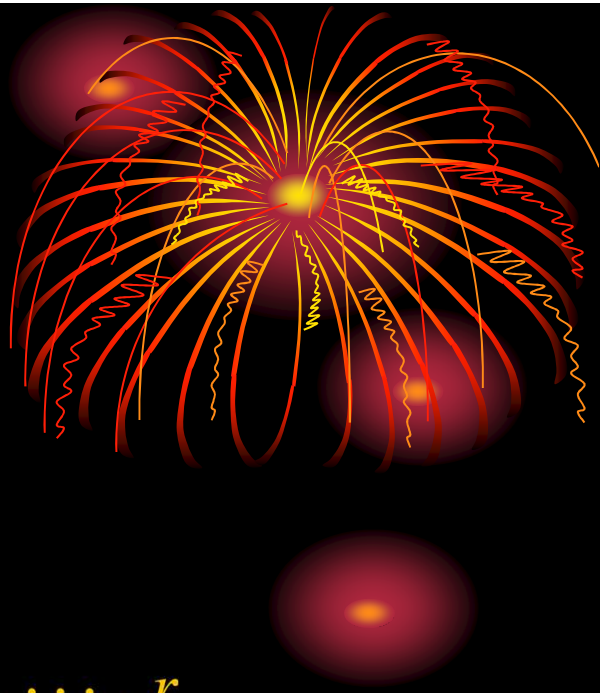
- Difficult to program under C, like ***A*B***
- MATLAB implementation

$$C = A + B \qquad C = A - B$$

- Note: either variable can be a scalar
- If not compatible, error messages given

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix Multiplication



- Math expression $C = AB$

$$c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}$$

$$i = 1, 2, \cdots, n, \quad j = 1, 2, \cdots, r$$

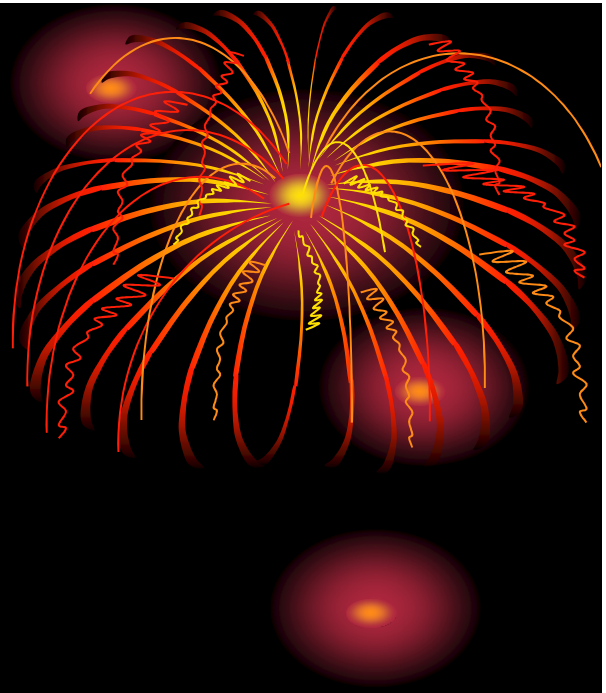
- MATLAB expression

```
C=A*B
```

- Note: compatibility auto-checked
  - Applies to complex and others

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix Division



- Matrix left division
  - Solve the linear equations:

$$AX = B$$

  - MATLAB solution:

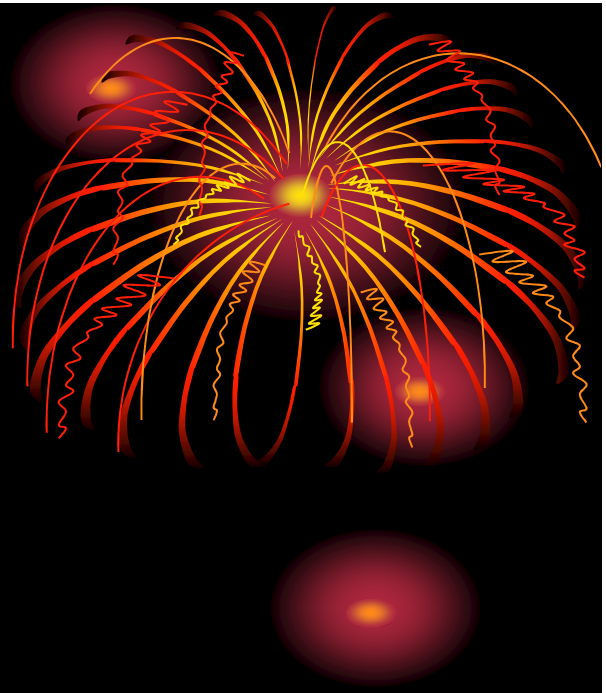
$$X = A\backslash B$$

  - Least squares solution
  - If A is a non-singular square matrix. Then,

$$X = A^{-1}B$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix Right Division



- Mathematical expression XA=***B***
  - MATLAB solution

$$X = B/A$$

  - Least squares solution
  - If A is a nonsingular square matrix. Then,

$$X = BA^{-1}$$

  - More precisely,

$$B/A=(A'\backslash B')'$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix Flips and Rotations

- left-right flip

$$b_{ij} = a_{i,n+1-j} \qquad B = \texttt{fliplr}(A)$$

- up-down flip

$$c_{ij} = a_{m+1-i,j} \qquad B = \texttt{flipud}(A)$$

- Rotate 90^o , counterclockwise

$$d_{ij} = a_{j,n+1-i} \qquad D = \texttt{rot90}(A)$$

- How to rotate 180^o , 270^o ?

$$D = \texttt{rot90}(A,k)$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix Power

- A must be a square matrix
- Matrix A to the power x.
  - Math description

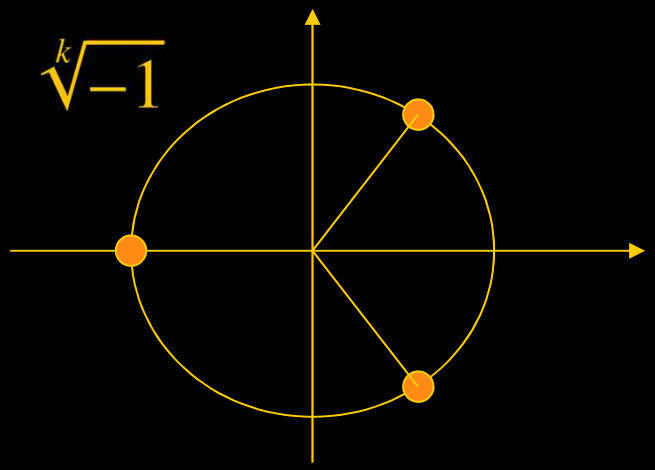
$$F = A^x$$

  - MATLAB command

```
F=A^x
```

  - Rotations needed

$$\gamma = e^{2\pi j/k}$$



Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.7 Cubic Roots

- Given matrix A

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{bmatrix}$$

- MATLAB code



```
>> A=[1,2,3; 4,5,6; 7,8,0];
   C=A^(1/3), e=norm(A-C^3)
```

- Other two roots



```
>> j1=exp(sqrt(-1)*2*pi/3);
   A1=C*j1, A2=C*j1^2,
   norm(A-A1^3), norm(A-A2^3)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Matrix Dot Operations

- Element-by-element operation
  - Dot product `C=A.*B` $c_{ij} = a_{ij}b_{ij}$
  - Dot power `B=A.^A` $b_{ij} = a_{ij}^{a_{ij}}$

$$\begin{bmatrix} 1^1 & 2^2 & 3^3 \\ 4^4 & 5^5 & 6^6 \\ 7^7 & 8^8 & 0^0 \end{bmatrix} = \begin{bmatrix} 1 & 4 & 27 \\ 256 & 3125 & 46656 \\ 823543 & 16777216 & 1 \end{bmatrix}$$



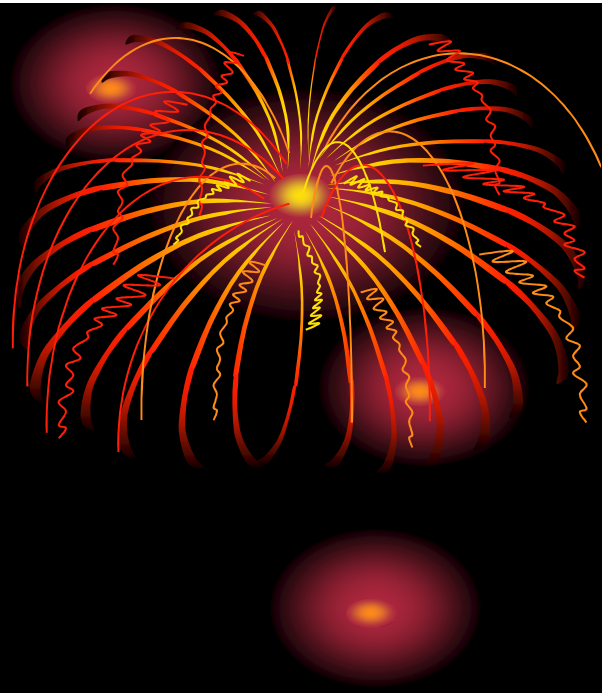
```
>> A=[1,2,3; 4 5,6; 7,8 0]; B=A.^A
```

  - Curve of a function $y = f(x) = x^2,\ y_i = x_i^2$

```
y=x.^2
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.2.2 Logic Operations



- Logical variables
  - For new version of MATLAB
  - Non-zero means logic 1
- Logical operations (element-by-element)
  - "And" operation `A & B`
  - "Or" operation `A | B`
  - "Not" operation `B =~ A`
  - Exclusive Or `xor(A, B)`

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.2.3 Relationship Operations of Matrices

- Allowed comparisons:
  - >, >=, <, <=, ==,~=, find(), all(), any()
- Examples:



```
>> A=[1,2,3; 4 5,6; 7,8 0];
   i=find(A>=5)'
```



```
>> a1=all(A>=5)
   a2=any(A>=5)
```

2.2.4 Simplifications and Conversions



- Function simple() can be used to simplify mathematical formula – old version:

```
s1=simple(s)
[s1,how]=simple(s)
```

  - In new versions, use simplify()
- Other commonly used simplification functions
  - numden(), collect(), expand(), factor()

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.8 Polynomial

□ Find the simplest form of the polynomial

$$P(s) = (s+3)^2(s^2+3s+2)(s^3+12s^2+48s+64)$$

□ Process it with various functions



```
>> syms s;
   P=(s+3)^2*(s^2+3*s+2)*(s^3+12*s^2+48*s+64)
```

In old versions

```
>> P1=simple(P)
   [P2,m]=simple(P)
```

□ Simplify the polynomial



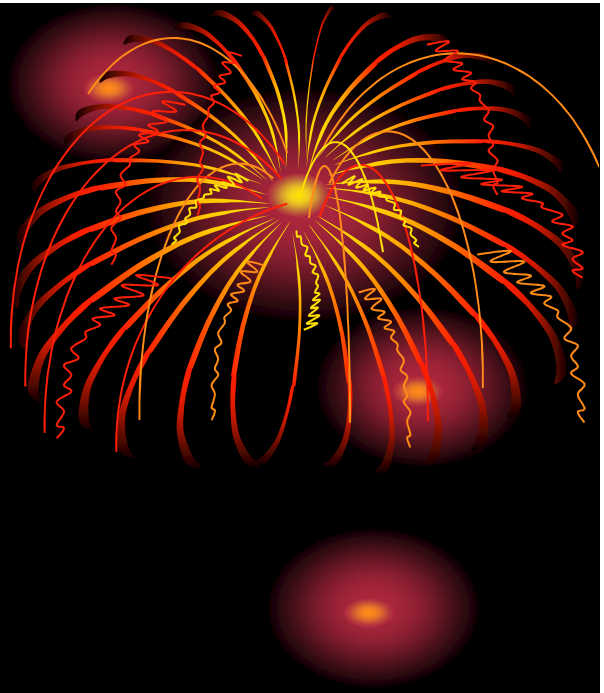
```
>> P4=simplify(P)
```



```
>> P3=expand(P)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Variable Substitution



- Two commands to use

$$f_1 = \texttt{subs}(f, x_1, x_1^*)$$
$$f_1 = \texttt{subs}(f, \{x_1, x_2, \cdots, x_n\}, \ldots \{x_1^*, x_2^*, \cdots, x_n^*\})$$

- It is run on the dot operation basis
- Convert to LATEX expression

$$f_1 = \texttt{latex}(f)$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.9 Math Display

- Function

$$f(t) = \cos(at + b) + \sin(ct)\sin(dt)$$

- Use taylor() to evaluate its Taylor expression and convert the results in LATEX



```
>> syms a b c d t;
   f=cos(a*t+b)+sin(c*t)*sin(d*t);
   f1=taylor(f);
   latex(f1)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

- By MATLAB

```
\cos \left( b \right) -\sin \left( b \right)
at+ \left( -1/2\,\cos \left( b \right)
{a}^{2}+cd \right) {t}^{2}+1/6\,\sin \left( b
\right) {a}^{3}{t}^{3}+ \left( 1/24\,\cos
\left( b \right) {a}^{4}-1/6\,c{d}^{3}-
1/6\,{c}^{3}d \right) {t}^{4}-{\frac
{1}{120}}\,\sin \left( b \right)
{a}^{5}{t}^{5}
```

- By LaTeX

$$\cos b - at\sin b + \left(-\frac{a^2\cos b}{2} + cd\right)t^2 + \frac{a^3\sin b}{6}t^3 + \left(\frac{a^4\cos b}{24} - \frac{cd^3}{6} - \frac{c^3 d}{6}\right)t^4 - \frac{a^5\sin b}{120}t^5$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.2.5 Basic Discrete Mathematics Computations

Function	Calling format
floor()	$n = \texttt{floor}(x)$
ceil()	$n = \texttt{ceil}(x)$
round()	$n = \texttt{round}(x)$
fix()	$n = \texttt{fix}(x)$
rat()	$[n, d] = \texttt{rat}(x)$
rem()	$B = \texttt{rem}(A, C)$
gcd()	$k = \texttt{gcd}(n, m)$
lcm()	$k = \texttt{lcm}(n, m)$
factor()	$\texttt{factor}(n)$
isprime()	$v_1 = \texttt{isprime}(v)$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.10 Finding Integers

- Data set
  - -0.2765, 0.5772,1.4597, 2.1091, 1.191,-1.6187
- Observe the results from different rounding functions.



```
>> A=[-0.2765,0.5772,1.4597,...
      2.1091,1.191,-1.6187];
   v1=floor(A), v2=ceil(A)
   v3=round(A), v4=fix(A)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.11 Rationalization

- 3x3 Hilbert matrix can be specified with the statement A = hilb(3), perform the rational transformation.



```
>> A=hilb(3); [n,d]=rat(A)
```

- result

$$n = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \; d = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{bmatrix}.$$

Example 2.12 GCD/LCM

- Two numbers 1856120 , 1483720
  - get the GCD (greatest common divisor)
  - LCM (least common multiple)
  - prime factor decomposition to the least common multiplier
- Better to use symbolic form



```
>> m=sym(1856120); n=sym(1483720);
   gcd(m,n), lcm(m,n),
   factor(lcm(n,m))
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.13 Prime Numbers

- Prime numbers in 1-1000



```
>> A=1:1000; B=A(isprime(A))
```

  - Two statements. Recall C programming

- The prime numbers obtained

2	29	67	107	157	199	257	311	367	421	467	541	599	647	709	769	829	887	967
3	31	71	109	163	211	263	313	373	431	479	547	601	653	719	773	839	907	971
5	37	73	113	167	223	269	317	379	433	487	557	607	659	727	787	853	911	977
7	41	79	127	173	227	271	331	383	439	491	563	613	661	733	797	857	919	983
11	43	83	131	179	229	277	337	389	443	499	569	617	673	739	809	859	929	991
13	47	89	137	181	233	281	347	397	449	503	571	619	677	743	811	863	937	997
17	53	97	139	191	239	283	349	401	457	509	577	631	683	751	821	877	941	
19	59	101	149	193	241	293	353	409	461	521	587	641	691	757	823	881	947	
23	61	103	151	197	251	307	359	419	463	523	593	643	701	761	827	883	953	

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.3 Flow Control Structures of MATLAB Language



- With Control structures, complicated and sophisticated programs can be written
  - Loop control structures
  - Conditional control structures
  - Switch structure
  - Trial structure

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.3.1 Loop Control Structures

- The for loop structures

```
for i=v
    loop programs body,
end
```

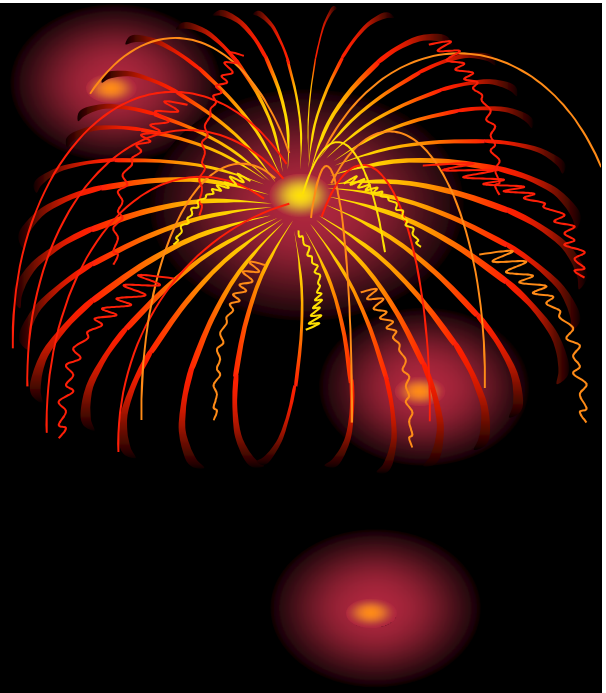
  - Every term in *v* is taken

- If *v* is a matrix, *i* pick up one column at a time

```
for (i=0; i<=n-1; i++)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

While Loops



- The while loop structures

```
while (condition)
    loop structure body,
end
```

- Loops can be nested
  - Different structures have different characteristics
  - Loops can be broken with break command

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.14 Sum of Sequence

□ Compute the sum of $\sum_{i=1}^{100} i$ using loop structures.



```
>> s1=0; for i=1:100, s1=s1+i; end
   s2=0; i=1;
   while (i<=100), s2=s2+i; i=i+1;
   end
```

□ the simplest statement



```
>> sum(1:100)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.15 How Many Terms

- Find the minimum value of m such that

$$\sum_{i=1}^{m} i > 10000$$

- Only while can be used

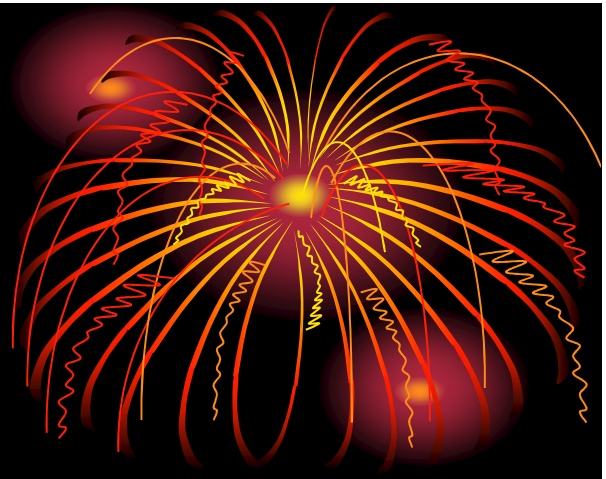


```
>> s=0; m=0;
   while (s<=10000), m=m+1;
   s=s+m; end, [s,m]
```

- For structure cannot act alone for the problem

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.16 Loops or Vectorization



- Evaluate the sum of the series

$$S = \sum_{i=1}^{100000} \left(\frac{1}{2^i} + \frac{1}{3^i} \right)$$

- MATLAB loop code, and vectorized code



```
>> tic, s=0;
   for i=1:100000, s=s+1/2^i+1/3^i;
   end; toc
   tic, i=1:100000;
   s=sum(1./2.^i+1./3.^i); toc
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.3.2 Conditional Control Structures

```
if (condition 1)
    statement group 1
elseif (condition 2)
    statement group 2
        ⋮   ⋮
else
    statement group n + 1
end
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.17 Another Way

- Using for and if statements to determine the minimum m

$$\sum_{i=1}^{m} i > 10000$$

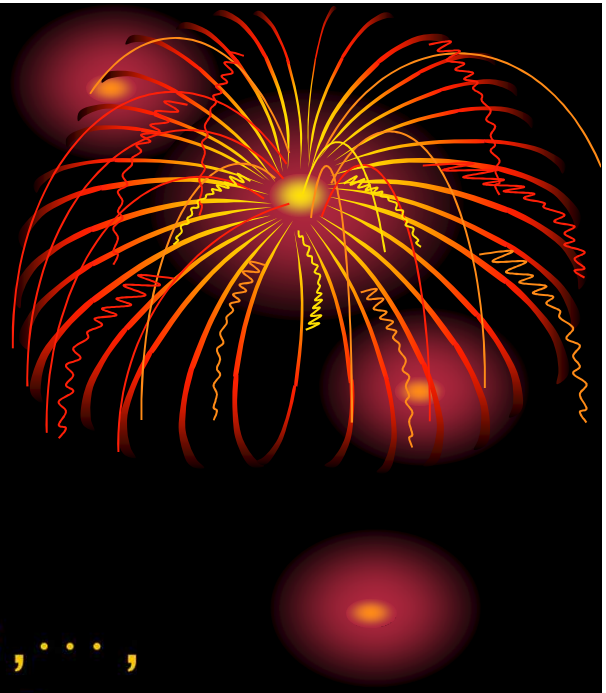


```
>> s=0;
   for i=1:10000, s=s+i;
       if s>10000, break;
   end, end
```

- It is more complicated than the while structure.

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

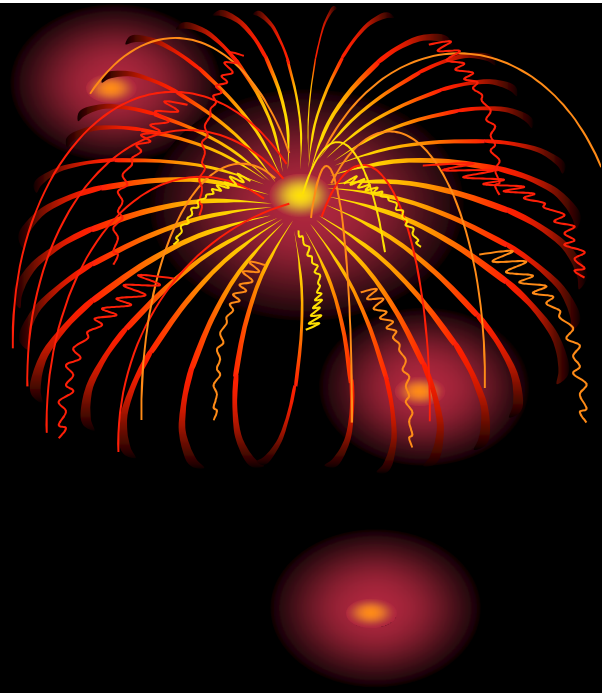
2.3.3 Switch Structure



```
switch switch expression
case expression 1, statements 1
case {expression 2, expression 3,···,
      expression m}, statements 2
      ⋮
otherwise, statements n
end
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.3.4 Trial Structure



- This is a brand new structure

```
try,   statement group 1,
catch,   statement group 2,
end
```

- Advantages:
  - An error trap
  - More efficient algorithm implementation

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

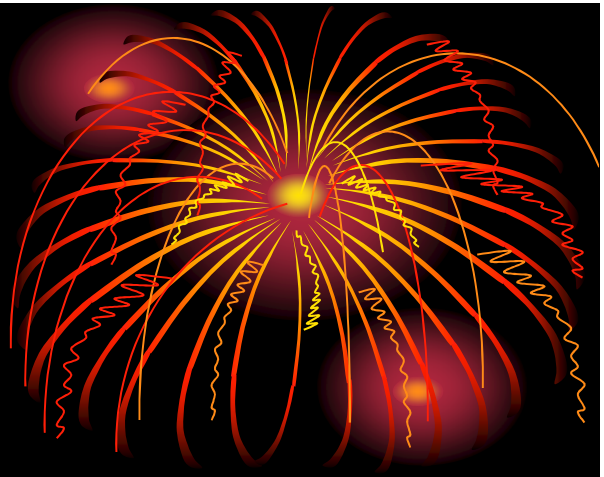


2.4 Writing and Debugging MATLAB Functions

- Basic structure of MATLAB functions
- Programming of functions with variable inputs/outputs
- Inline functions and anonymous functions

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.4.1 Basic Structure of MATLAB Functions



- Function structures

```
function [return argument list] ...
              =funname(input argument list)
    comments led by %
    input and output variables check
    main body of the function
```

- Important functions
  - nargin, nargout, varargin, varargout

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.18 Why Functions are Needed?

- Problem

$$\sum_{i=1}^{m} i > 10000$$

- M-script can be written and saved as an M-file.
  - If 10000 is changed to other values, the M-file should be modified
- A new file format (function) is needed

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.19 M-Function Implementation

- Write an M-function for Example 2.18
- M-function

```
function [m,s]=findsum(k)
s=0; m=0;
while (s<=k), m=m+1; s=s+m; end
```

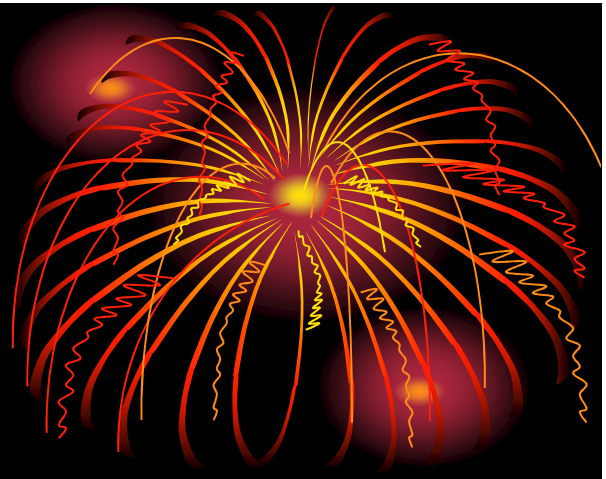
- Example



```
>> [m1,s1]=findsum(145323)
```

- Advantages: no need to modify for *N*

Example 2.20 Hilbert Matrix Generation



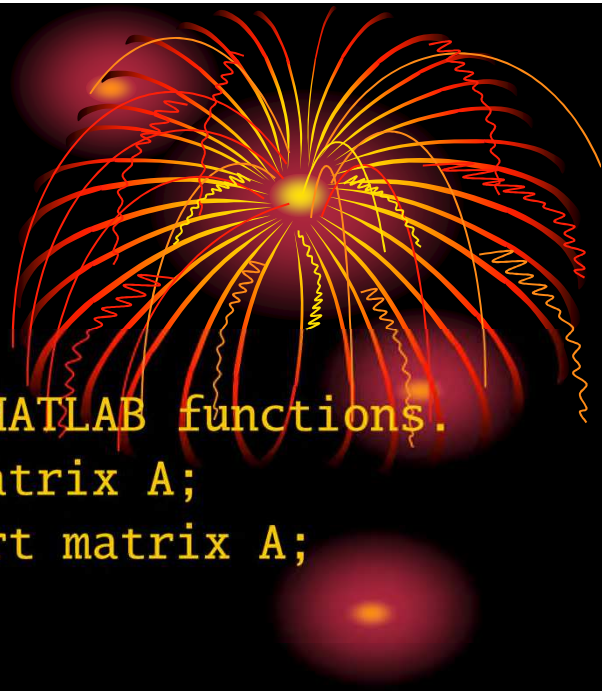
- Write an M-function for $n \times m$ Hilbert matrix

$$h_{i,j} = \frac{1}{i + j - 1}$$

- Requirements
  - If only 1 input argument given, generate a square matrix
  - Write suitable help information
  - Check the numbers of inputs and outputs

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

□ MATLAB function



```
function A=myhilb(n, m)
%MYHILB The function is used to illustrate MATLAB functions.
% A=MYHILB(N, M) generates an NxM Hilbert matrix A;
% A=MYHILB(N) generate and NxN square Hilbert matrix A;
%
%See also: HILB.

% Designed by Professor Dingyu XUE, Northeastern University, PRC
% 5 April, 1995, Last modified by DYX at 30 July, 2001
if nargout>1, error('Too many output arguments.'); end
if nargin==1, m=n; % if one input argument used, square matrix
elseif nargin==0 | nargin>2
   error('Wrong number of iutput arguments.');
end
for i=1:n, for j=1:m, A(i,j)=1/(i+j-1); end, end
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

- On-line help commands



```
>> help myhilb
>> doc myhilb
```

- Generate Hilbert matrices



```
>> A1=myhilb(4,3)
   A2=myhilb(4)
```

- Limitations: cannot generate symbolic matrix
  - Modify kernel statements

```
[i,j]=meshgrid(1:m,1:n); A=1./(i+j-1);
```



```
>> A3=myhilb(sym(4),3)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.21 Recursive Implementation of Factorials

- Recursive relations $n! = n(n-1)!$
- Recursive function implementation

```
function k=my_fact(n)
if nargin~=1,
   error('Error: Only one input variable accepted');
end
if abs(n-floor(n))>eps | n<0
   error('n should be a non-negative integer');
end
if n>1, k=n*my_fact(n-1);
elseif any([0 1]==n), k=1; end
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Factorial Calculations

- Calculate 11!



```
>> my_fact(11)
```

- Other functions `factorial(n)`

  `prod(1:n)` `gamma(1+n)`

- Symbolic computation

```
gamma(1+sym(n))
factorial(sym(n))
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.22 Not Suitable for Recursive Implementation

- Implement Fibonacci sequence
  - Fibonacci sequence

$$a_1 = a_2 = 1$$

$$a_k = a_{k-1} + a_{k-2}\ ,\ k = 3, 4, \cdots$$

  - Recursive implementation in MATLAB

```
function a=my_fibo(k)
 if k==1 | k==2, a=1;
 else, a=my_fibo(k-1)+my_fibo(k-2);
 end
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

- The 25th term (takes 3 seconds)

```
>> tic, my_fibo(25), toc
```

- With for loop, 1000th term can be obtained

```
>> tic, a=[1,1];
   for k=3:1000, a(k)=a(k-1)+a(k-2);
   end, toc
```

- Note: not suitable to use recursive structures
- Symbolic computation

```
>> tic, a=sym([1,1]);
   for k=3:100, a(k)=a(k-1)+a(k-2);
   end, toc
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.4.2 Handling Variable Input and Returned Arguments

- Functions with variable numbers of arguments
  - In/out arguments: varargin, varargout
- Extracting actual input arguments
  - varargin{1}, varargin{2},…, varargin{n}
- Storage and transfer of the arguments



Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.22 Handling Arbitrary Number of Input Arguments

- conv(): calculate product of 2 polynomials
- Handle arbitrary number of polynomials
- MATLAB programming
  - Extract 1 term from varargin at a time

```
function a=convs(varargin), a=1;
for i=1:length(varargin), a=conv(a,varargin{i}); end
```



```
>> P=[1 2 4 0 5]; Q=[1 2];
   F=[1 2 3]; D=convs(P,Q,F), E=conv(conv(P,Q),F)
   G=convs(P,Q,F,[1,1],[1,3],[1,1])
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.4.3 Inline and Anonymous Functions



- inline function (not recommended)
  - No need to create an M-file

```
fun = inline(function expression,...
             list of variables)
```

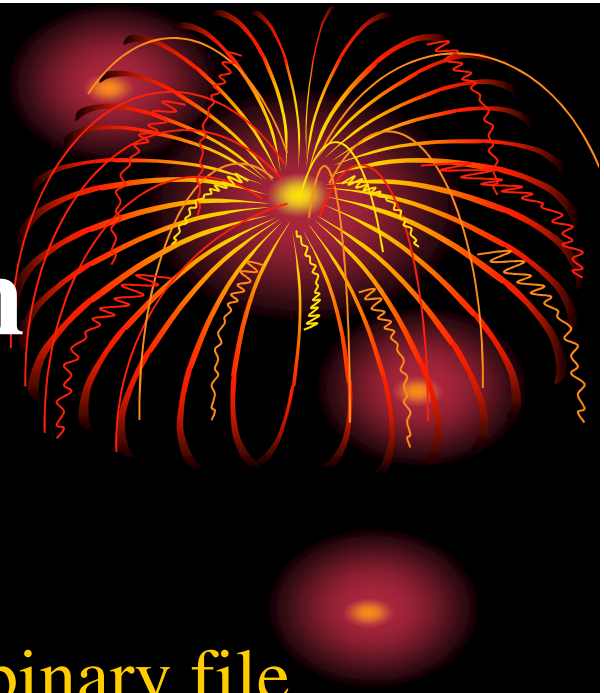
- For MATLAB7.0+, recommend anonymous
  - Direct use of variables in MATLAB workspace

```
f = @(list of variables)function_contents
```

- Other types of functions – private, overload

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.4.4 Pseudo Code and Source Code Protection



- Why pseudo code?
  - Increase execution speed
  - Protect source code: ASCII file to binary file
- Creating pseudo code

```
pcode mytest  pcode *.m
pcode mytest -inplace
```

- Must reserve source code in safe place
- .p files not reversible

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.5 Two-dimensional Graphics

- Scientific visualization is important in scientific research. Convert data into graphs
  - Basic statements in 2D graphics
  - Plots with multiple vertical axes
  - Other 2D graphics facilities
  - Plotting implicit functions
  - Access data in files (*.txt or *.xls)

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.5.1 Basic Statements in 2D Plots



- Two sequences

$$t = t_1, t_2, \cdots, t_n \qquad y = y_1, y_2, \cdots, y_n$$

  - Construct vectors

$$t = [t_1, t_2, \cdots, t_n]$$

$$y = [y_1, y_2, \cdots, y_n]$$

- Use the data to draw plots

```
plot(t,y)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.24 Function Plotting and Validations

- Draw plot for function

$$y = \sin(\tan x) - \tan(\sin x),\ \ x \in [-\pi, \pi]$$

- MATLAB code



```
>> x=[-pi : 0.05: pi];
   y=sin(tan(x))-tan(sin(x)); plot(x,y)
```

- Problem: how to validate curves?
  - Select different step sizes

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Variable Step-size Vectors

- Use different step-sizes
  - Use the same small step-size

```
>> x=[-pi:0.00001:pi];
   y=sin(tan(x))-tan(sin(x)); plot(x,y)
```

- Use variable step-size, small around $\pm\pi/2$



```
>> x=[-pi:0.05:-1.8,-1.801:.001:-1.2,...
      -1.2:0.05:1.2,1.201:0.001:1.8,...
      1.81:0.05:pi];
   y=sin(tan(x))-tan(sin(x)); plot(x,y)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.25 Piecewise Functions

- Draw saturation function

$$y = \begin{cases} 1.1\text{sign}(x), & |x| > 1.1 \\ x, & |x| \le 1.1 \end{cases}$$

- MATLAB draw (mutual exclusive conditions)



互斥

```
>> x=[-2:0.02:2];
   y=1.1*sign(x).*(abs(x)>1.1)+...
     x.*(abs(x)<=1.1); plot(x,y)
```

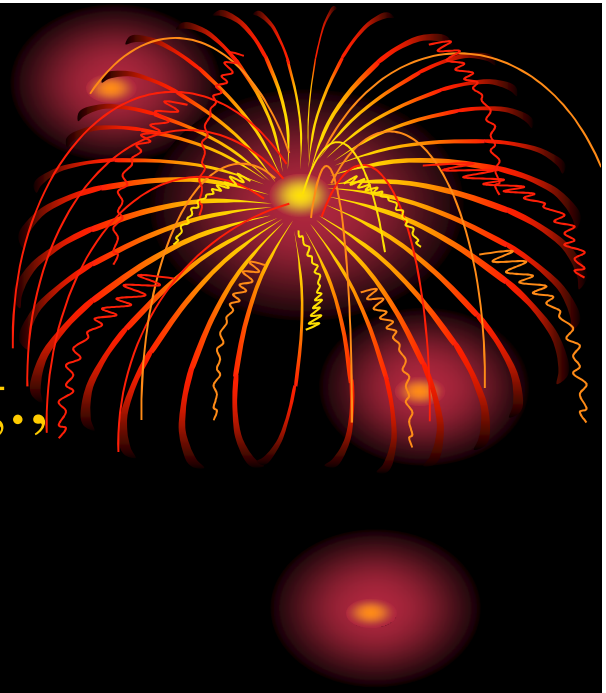
- Simpler statement: draw poly-lines



```
>> plot([-2,-1.1,1.1,2],[-1.1,-1.1,1.1,1.1])
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Other Syntaxes



- t is vector, while y is a matrix, e.g.,

$$y = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mn} \end{bmatrix}$$

- t and y are matrices, size of t and y the same
- More pairs of vectors or matrices

$$(t_1, y_1), (t_2, y_2), \cdots, (t_m, y_m),$$

$$\texttt{plot}(t_1, y_1, t_2, y_2, \cdots, t_m, y_m)$$

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

A More General Syntax

- Change the properties directly

h = plot(t_1, y_1, option 1, t_2, y_2, option 2, $\cdots$, t_m, y_m, option m)

line type	colour		marks	
'-'	'b'	'c'	'*'	'pentagram'
'--'	'g'	'k'	'.'	'o'
':'	'm'	'r'	'x'	'square'
'-.'	'w'	'y'	'v'	'diamond'
'none'			'^'	'hexagram'
			'>'	'<'

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Object Properties Extraction and Settings

- Figure window, curve, axes are objects
- Object properties can be assigned with `set()`
- Properties can be extracted with `get()`

```
set(handle,'p_name 1',...
    p_value 1,'p_name 2',p_value 2, ···)
v = get(object, 'p_name')
```

- Object properties can be set with menus

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.5.2 Plots with Multiple Vertical Axes

- Sometimes the differences in the two plots are huge, plotyy() function should be used
- Example 2-26: Draw the two functions

$$y_1 = \sin x,\ y_2 = 0.01\cos x$$



```
>> x=0:0.01:2*pi; y1=sin(x);
   y2=0.01*cos(x); plot(x,y1,x,y2,'--')
```

- More plotting functions for more axes
  - plotyyy(), plot4y(), plotxx() functions
  - Downloadable from MATLAB File Exchange

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.5.3 Drawing Other Forms of 2D Graphics

- Different functions to use

bar(x,y)	comet(x,y)
compass(x,y)	errorbar(x,y,y_m,y_M)
feather(x,y)	fill(x,y,c)
hist(y,n)	loglog(x,y)
polar(x,y)	quiver(x,y)
stairs(x,y)	stem(x,y)
semilogx(x,y)	semilogy(x,y)

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.27 Polar Plots

- Draw polar function curves

$$\rho = 5\sin(4\theta/3),\ \rho = 5\sin(\theta/3)$$

- Drawing curves



```
>> theta=0:0.01:2*pi; rho=5*sin(4*theta/3);
   polar(theta,rho)
```

  - Incomplete – find the period



```
>> rho=5*sin(theta/3);
   polar(theta,rho)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Drawing plots as Subplots

- Divide graphics window into several parts

```
subplot(n,m,k)
```

  - n and m are rows and columns
  - k is the serial number of the portion of window
- Also setting with menu items
  - Add an axis
  - Drag mouse button to adjust axis size

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.27 Different 2D Plots as Subplots in Windows

- Sinusoidal function is used as an example
- Divide graph window as 2x2
- Draw in different portions



```
>> t=0:.2:2*pi; y=sin(t);
   subplot(2,2,1), stairs(t,y)
   subplot(2,2,2), stem(t,y)
   subplot(2,2,3), bar(t,y)
   subplot(2,2,4), semilogx(t,y)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.5.4 Drawing Implicit Functions

- Example of an implicit function

$$x^2 \sin(x + y^2) + y^2 e^{x+y} + 5\cos(x^2 + y) = 0$$

- Drawing implicit functions

```
ezplot(implicit function expression)
```

  - Represent implicit functions as strings
  - Default region is $[-2\pi, 2\pi]$

- Other syntaxes `ezplot(im_function,[`x_m`,`x_M`])`

`ezplot(im_function,[`x_m`,`x_M`,`y_m`,`y_M`])`

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.29 Implicit Function

- Please draw

$$x^2 \sin(x + y^2) + y^2 e^{x+y} + 5\cos(x^2 + y) = 0$$

- MATLAB statements -- handles



```
>> h=ezplot('x^2 *sin(x+y^2) +...
            y^2*exp(x+y)+5*cos(x^2+y)');
   set(h,'Color','b')
```

- Over a large region



```
>> h=ezplot('x^2 *sin(x+y^2) +y^2*exp(x...
          +y)+5*cos(x^2+y)',[-10 10]);
   set(h,'Color','b')
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.5.5 Decoration of Plots

- Direct use of toolbar
- Text decorations
  - Special symbols LaTeX
  - Subscript and superscripts, with ^ and _

```
a_2^2+b_2^2=c_2^2
```

$$a_2^2 + b_2^2 = c_2^2$$

  - LaTeX advantages. Better to use overpic package
- New functions since MATLAB 7.0

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.5.6 Access Data Files



- Commands save and load

```
save mydat A B C
save /ascii mydat.dat A B C
X = load(filename)
```

- Exchange between MATLAB & Excel

```
X = xlsread(filename,range)
```

```
'B6:C67'
```

  - Write file: xlswrite()

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2-30 Access Excel File

- Known Excel file: census.xls – population
  - Open file  `>> open('census.xls')`
  - Rows 5-67 stores data
  - B column stores year, C columns, population
- Read into MATLAB, the plotting



```
>> X=xlsread('census.xls','B5:C67');
   t=X(:,1); p=X(:,2);
   plot(t,p)
```

- Simpler method: Copy & Paste

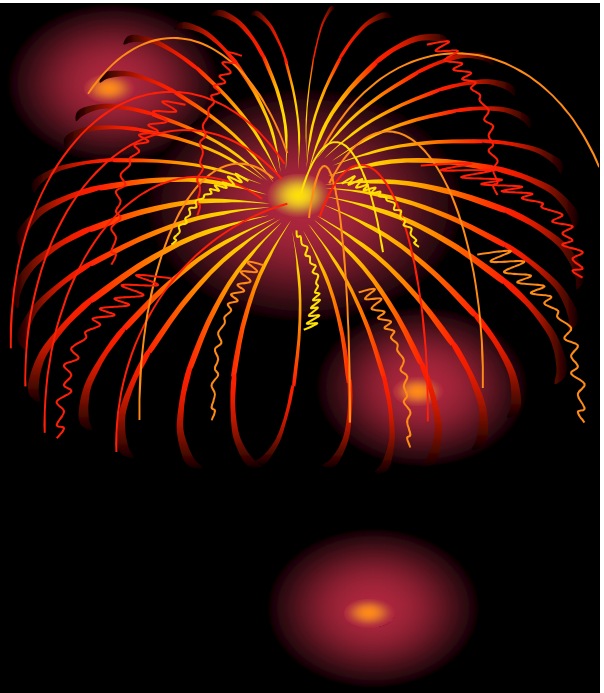
Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.6 Three-dimensional Graphics

- Difficult to obtain with other computer languages, easy to obtain with MATLAB
- Main topics in the section
  - 3D curves
  - 3D surfaces
  - View-point setting
  - 3D implicit function plots
  - Rotating 3D graphs

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.6.1 3D Curves



- Plotting 3D curves

```
plot3(x,y,z)
plot3(x1,y1,z1,option 1,...
      x2,y2,z2,option 2,...
      xm,ym,zm,option m)
```

- Other 3D plotting functions
  - stem3, fill3, bar3
  - Draw trajectory dynamically: comet3

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.31 Position of a Particle in 3D Space

- 3D parametric equation

$$x(t) = t^3 \sin(3t)e^{-t}, y(t) = t^3 \cos(3t)e^{-t}, z = t^2$$

  - Position of a particle (change with time)
  - where, $t \in [0, 2\pi]$

- MATLAB plotting (dot calculation)



```
>> t=0:0.01:2*pi; x=t.^3.*sin(3*t).*exp(-t);
   y=t.^3.*cos(3*t).*exp(-t);
   z=t.^2; plot3(x,y,z), grid
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Other 3D Plots

- With function `stem3()`



```
>> stem3(x,y,z); hold on;
   plot3(x,y,z), grid; hold off
```

- Dynamic trajectory



```
>> figure; comet3(x,y,z)
```

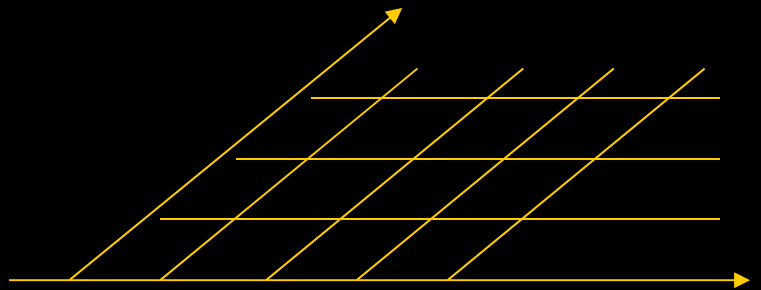
- Toolbar usages
  - View point adjustment in 3D plots
  - Read coordinate, zooming facilities

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.6.2 3D Surface Plots

- Draw surfaces

```
[x,y]=meshgrid(v1, v2)
z= ...,     for instance  z=x.*y
surf(x,y,z) or  mesh(x,y,z)
```



- Other functions
  - surfl(), surfc()
- Contour plots
  - contour(), contours()

Example 2.32 3D Surface

- Given function with 2 variables

$$z = f(x, y) = (x^2 - 2x)e^{-x^2-y^2-xy}$$

- MATLAB plot



```
>> [x,y]=meshgrid(-3:0.1:3,-2:0.1:2);
   z=(x.^2-2*x).*exp(-x.^2-y.^2-x.*y);
   mesh(x,y,z)
```

- Surface plot



```
>> surf(x,y,z)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.33 Another Function

- Two-variable function

$$z = f(x,y) = \frac{1}{\sqrt{(1-x)^2+y^2}} + \frac{1}{\sqrt{(1+x)^2+y^2}}$$

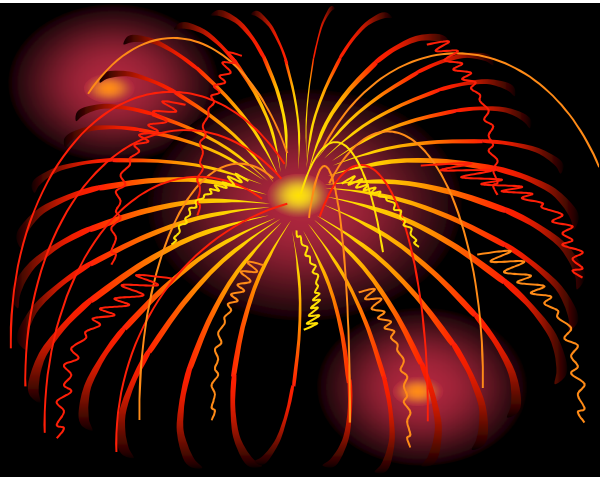
- Draw 3D surface



```
>> [x,y]=meshgrid(-2:.1:2);
   z=1./(sqrt((1-x).^2+y.^2))+...
        1./(sqrt((1+x).^2+y.^2));
   surf(x,y,z), shading flat
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Step-size Selection



- With variable step-sizes



```
>> xx=[-2:.1:-1.2,-1.1:0.02:-0.9,...
       -0.8:0.1:0.8,0.9:0.02:1.1, 1.2:0.1:2];
   yy=[-1:0.1:-0.2, -0.1:0.02:0.1, 0.2:.1:1];
   [x,y]=meshgrid(xx,yy);
   z=1./(sqrt((1-x).^2+y.^2))+...
       1./(sqrt((1+x).^2+y.^2));
   surf(x,y,z), shading flat;
   zlim([0,15])
```

- Singular point, to be discussed in Ch5

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.34 Handling Piecewise Functions

□ Draw surface for piecewise function

$$p(x_1, x_2) = \begin{cases} 0.5457 \exp(-0.75x_2^2 - 3.75x_1^2 - 1.5x_1), & x_1 + x_2 > 1 \\ 0.7575 \exp(-x_2^2 - 6x_1^2), & -1 < x_1 + x_2 \le 1 \\ 0.5457 \exp(-0.75x_2^2 - 3.75x_1^2 + 1.5x_1), & x_1 + x_2 \le -1. \end{cases}$$

➢Evaluation of piecewise functions

✓Mutually exclusive relations, dot operations

✓Loops, rather complicated

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Drawing Piecewise Surfaces

- Math form

$$p(x_1, x_2) = \begin{cases} 0.5457\exp(-0.75x_2^2 - 3.75x_1^2 - 1.5x_1), & x_1 + x_2 > 1 \\ 0.7575\exp(-x_2^2 - 6x_1^2), & -1 < x_1 + x_2 \le 1 \\ 0.5457\exp(-0.75x_2^2 - 3.75x_1^2 + 1.5x_1), & x_1 + x_2 \le -1. \end{cases}$$

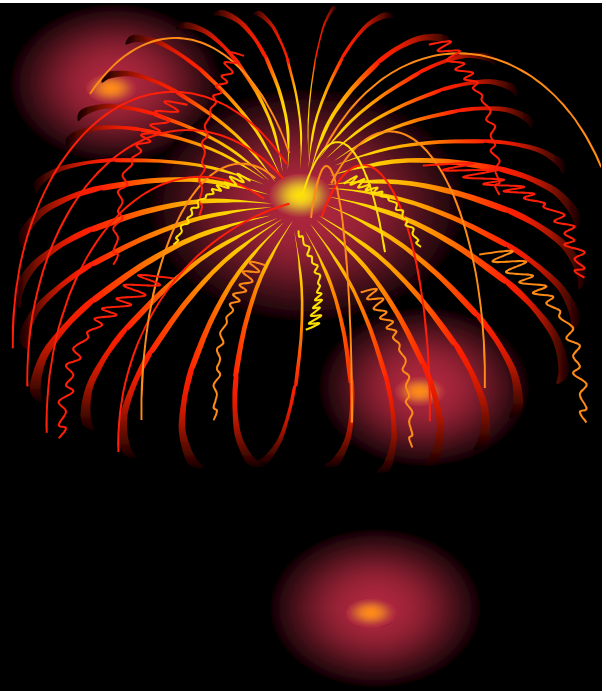
- MATLAB plotting
  - Generate mesh grid, evaluate then plot



```
>> [x1,x2]=meshgrid(-1.5:.1:1.5,-2:.1:2);
   p=0.5457*exp(-0.75*x2.^2-3.75*x1.^2-1.5*x1).*(x1+x2>1)+...
     0.7575*exp(-x2.^2-6*x1.^2).*((x1+x2>-1)& (x1+x2<=1))+...
     0.5457*exp(-0.75*x2.^2-3.75*x1.^2+1.5*x1).*(x1+x2<=-1);
   surf(x1,x2,p), xlim([-1.5 1.5]);
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.6.3 Contour Plots



- Various of contour plots
  - Direct draw `contour(x,y,z,n)`
  - Contours with labels

```
[C,h] = contour(x,y,z,n)
clabel(C,h)
```

  - 3D contours

`contour3(x,y,z,n)` `contourf(x,y,z,n)`

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2-35 Contours

- Example 2-34, piecewise function
  - Generate data, draw contours



```
>> [x1,x2]=meshgrid(-1.5:.1:1.5,-2:.1:2);
   p=0.5457*exp(-0.75*x2.^2-3.75*x1.^2-1.5*x1).*(x1+x2>1)+...
     0.7575*exp(-x2.^2-6*x1.^2).*((x1+x2>-1)& (x1+x2<=1))+...
     0.5457*exp(-0.75*x2.^2-3.75*x1.^2+1.5*x1).*(x1+x2<=-1);
   [C,h]=contour(x1,x2,p); clabel(C,h)
```

- 3D contours



```
>> contourf(x1,x2,p);
   figure; contour3(x1,x2,p,30)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.6.4 Plots of 3D Implicit Functions

- 3D implicit function $f(x,y,z)=0$
- Download ezimplot3() from File Exchange

```
ezimplot3(fun,[xm,xM,ym,yM,zm,zM])
```

  - Default region [-2π,2π]
- Fun can be of
  - Implicit function strings
  - Anonymous functions
  - M-functions

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2-36 3D Implicit Function Plots

- Given 3D implicit functions

$$x(x,y,z) = x\sin\left(y+z^2\right) + y^2\cos\left(x+z\right)$$
$$+zx\cos\left(z+y^2\right) = 0$$

- Drawing statements

```
>> f='x*sin(y+z^2)+y^2*cos(x+z)+...
      z*x*cos(z+y^2)';
   f=@(x,y,z)x*sin(y+z^2)+y^2*cos(x+z)+...
      z*x*cos(z+y^2);
>> ezimplot3(f,[-1 1]);
>> f1='x^2+y^2+z^2-1'; ezimplot3(f1,[-1 1]);
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.6.5 View-point Specifications

- Two ways of setting view points
  - Direct use of toolbar
  - Command `view()`

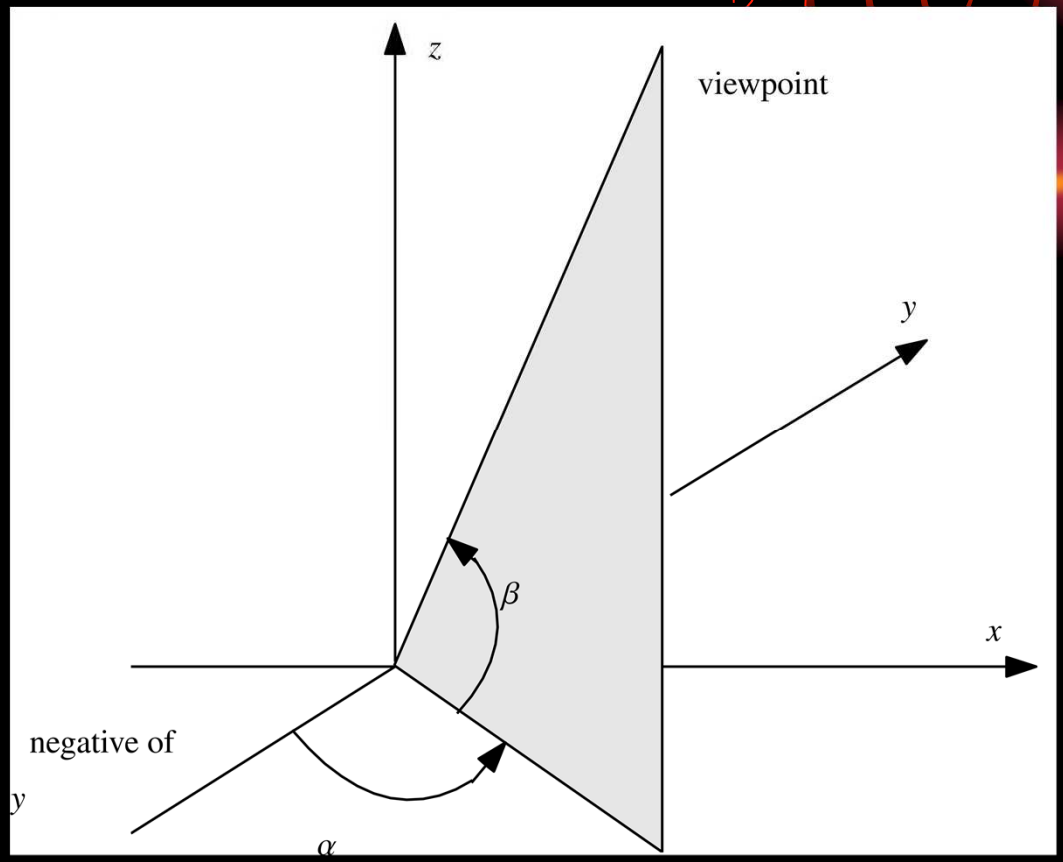
    `view(`α`,`β`)`
  - Read current view points `[`α`,`β`]=view(3)`
- α is azimuth angle, β is elevation angle
- Uniquely define the view-point

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Definition of View-points

- Settings

  `view(`α, β`)`



Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2.37 Orthographic Views of 3D Plots

三视图

- Given function

$$z = f(x, y) = (x^2 - 2x)e^{-x^2-y^2-xy}$$

- Default angle extraction

```
>> [a b]=view(3);
```

- 3D surface plot



```
>> [x,y] = meshgrid(-3:0.1:3,-2:0.1:2);
   z=(x.^2-2*x).*exp(-x.^2-y.^2-x.*y);
   subplot(224), surf(x,y,z),
   subplot(221), surf(x,y,z), view(0,90);
   subplot(222), surf(x,y,z), view(90,0);
   subplot(223), surf(x,y,z), view(0,0);
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.6.6 Rotations of 3D Surfaces

- Rotating function

  `rotate(h,`v`,`α`)`

  - where
    - ✓Handles of 3D plot h
    - ✓Rotate baseline by vector v
    - ✓Angle is α
  - Example: rotate along positive direction of x axis, v = [1,0,0]

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2-38 Rotating 3D Plots

- Rotate the piecewise plot

```
>> [x,y]=meshgrid(-1:.04:1,-2:.04:2);
   z= 0.5457*exp(-0.75*y.^2-3.75*x.^2-1.5*x).*(x+y>1)+...
      0.7575*exp(-y.^2-6*x.^2).*((x+y>-1) & (x+y<=1))+...
      0.5457*exp(-0.75*y.^2-3.75*x.^2+1.5*x).*(x+y<=-1);
   h=surf(x,y,z);
```

- Rotate along x axis, baseline v=[1,0,0]

```
>> rot_ax=[1,0,0]; rotate(h,rot_ax,15)
```

- Rotate along baseline v =[1 1 1]

```
>> h=surf(x,y,z); rot_ax=[1,1,1];
   rotate(h,rot_ax,15)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Animation of Rotations

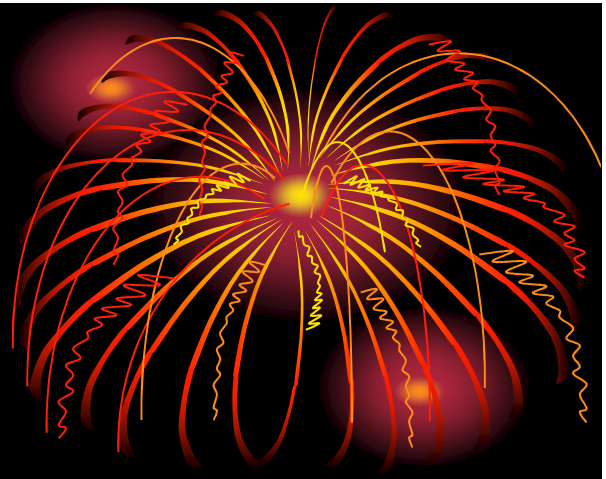
- Rotate along *x* axis for 360 degrees
  - Rotate 1 degree per step (0.01s pause)
  - Loop structure needed



```
>> figure; h=surf(x,y,z);
   r_ax=[1 0 0]; axis tight
   for i=0:360,
      rotate(h,r_ax,1); pause(0.01),
   end
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

2.7 Drawing 4D Plots



- 4D plots
  - Time-driven 3D animation
  - Slice view of 3D plots: volume visualization
  - Examples
    - ✓CT images, temperature or density of a 3D solid
    - ✓Slices are needed to view inside
    - ✓Flow rate and concentration of liquid
  - MATLAB function

$$\texttt{slice}(x, y, z, V, x_1, y_1, z_1)$$

体视化

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2-39 Slice Views of 3D Solid Objects

- Function

$$V(x,y,z)=\sqrt{x^x+y^{(x+y)/2}+z^{(x+y+z)/3}}$$

  - Ordinary slices
    - ✓Generate 3D mesh grids
    - ✓Calculate function values in the grids (dot operation)
    - ✓Plotting



```
>> [x,y,z]=meshgrid(0:0.1:2);
   V=sqrt(x.^x+y.^((x+y)/2)+z.^((x+y+z)/3));
   slice(x,y,z,V,[1 2],[1 2],[0 1]);
```

Special Slices

- Construct a plane at $z = 1$
- Rotate 45^o the plane along x axis
- Slice with the plane



```
>> [x0,y0]=meshgrid(0:0.1:2);
   z0=ones(size(x0));
   h=surf(x0,y0,z0); rotate(h,[1,0,0],45);
   x1=get(h,'XData'); y1=get(h,'YData');
   z1=get(h,'ZData');
   slice(x,y,z,V,x1,y1,z1),
   hold on, slice(x,y,z,V,2,2,0)
```


Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

A Volume Visualization Interface



- A volume visualization interface is developed vol_visual4d()
- Syntax
  - Generate volume data x, y, z, V
  - Call the function

    `vol_visual4d(`x`,`y`,`z`,`V`)`
  - Freely setting different slices

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Example 2-40 Use the Volume Visualization Interface

- Function

$$V(x,y,z)=\sqrt{x^x+y^{(x+y)/2}+z^{(x+y+z)/3}}$$

- Slice plotting
  - Use scrollbars to adjust the slices
  - Select whether a slice is shown



```
>> [x,y,z]=meshgrid(0:0.1:2);
   V=sqrt(x.^x+y.^((x+y)/2)+z.^((x+y+z)/3));
   vol_visual4d(x,y,z,V);
```

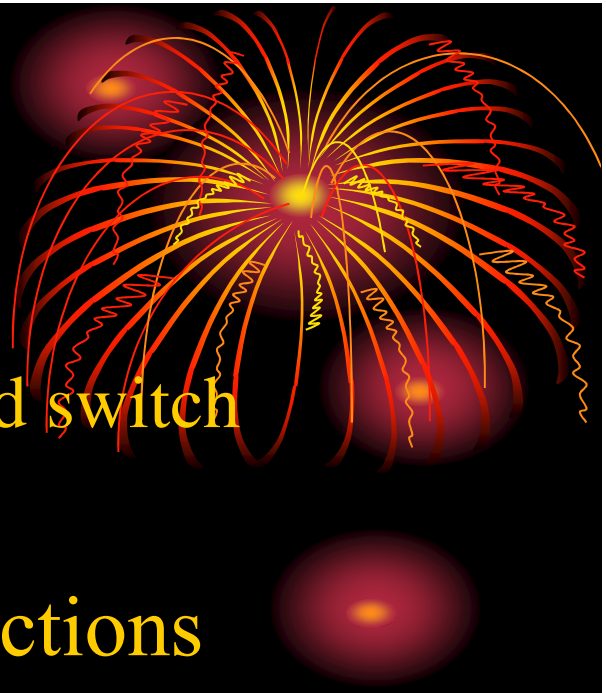

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual

Chapter Summary



- Fundamentals of MATLAB programming
  - Constants and variables
  - Data types and statement structures
  - Colon expression and sub-matrix extraction
- Matrix computation
  - Algebraic, logic and relational operations
  - Expression simplification and conversion
  - Some discrete math computation

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual



- Control flow structures
  - Two kinds of loops, conditional and switch
  - Trial structure in MATLAB
- Main stream programming -- functions
  - Standard function structures
  - Examming input and output arguments
  - With arbitrary number of in/out arguments
  - Recursive structures
  - Pseudo-code manipulations

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual



- Two-dimensional graphics
  - Descartes, polar, logarithmic and bars
  - Implicit function plotting
  - Decorations of plots
  - View point setting
- Three-dimensional plots and surfaces
  - 3D implicit functions
  - Rotations of plots
- Four-dimensional volume visualization
  - An easy-to-use GUI

Scientific Computing with MATLAB 2nd edition Xue Solutions Manual