《数学软件》第2讲

Thinking in Matrix - Vectorization

主讲教师：胡贤良

浙江大学数学科学学院

暑期短学期课程, 7月6日-7月13日, 2020年

可视化技术

计算的结果终要转化成视觉可度量的图形，是科学计算与数据分析的一项重要技术。各类数学软件均能提供相关的工具，使得任何数值计算人员可以不去研究“苦涩” 的可视化理论。

- 二维数据可视化
  - plot,ezplot,fplot等
  - stem, quiver, polar, contour等
  - title, x-label, grid on hold on ...
  - subplot, figure句柄
- 三维数据可视化
  - meshgrid
  - mesh 和surf
  - trimesh - 表面三角网格可视化
  - trisurf 和patch

二维绘图函数plot命令解析

利用MATLAB绘图命令将科学与工程数据绘制成常用二维图形

plot	X-Y方向绘图
loglog	X-Y方向双对数绘图
semilogx	对数绘图(x轴取对数)
semilogy	对数绘图(y轴取对数)

上述绘图函数参数形式类似，只须掌握plot即可融会贯通：

```
plot(X,Y,'b*-'); %指定线型、颜色
plot(X,Y,'linewidth',5,'markersize',10) % 指定画线粗细标记大小
```

折线的线型、颜色等参数

字母	颜色	标点	线型	名称	线型
y	黄	.	点线	square	正方形
m	紫	◯	圈线	diamond	菱形
c	青	×	×线	pentagram	五角星
r	红	+	+字线	hexagram	六角星
g	绿	-	实线		
b	蓝	*	星形线		
w	白	:	点线		
k	黑	-.	点划线		

其他图形元素控制

- 标题设定: title('your title here')
- 坐标轴控制: axis on; axis equal; axis([a0, b0, a1, b1]);
- 坐标轴名称: xlabel('x');ylabel('y');
- 网格线: grid on; grid off;
- 色彩控制: colorbar; colormap;
- 图例文字说明: legend('str1','str2','str3',...);
- 任意位置的文字说明: text(x,y,'string');

绘图完成后，可用 print 命令导出图形为 .png、.jpg、.tiff、.eps 或 .epsc 等格式的图形文件，供撰写报告或文挡使用

绘图的一般过程

1. 准备数据
2. 设定绘制图形的类型和方式
3. 将数据绘制在指定位置
4. 添加各种所需图形元素
5. 输出图形到指定设备

三维绘图基本函数

二维图形中，曲线是构成图形的基本元素，而在三维图形中，虽然曲线还广泛存在，但是更加常用的元素是曲面，在MATLAB中常称为网格面。先介绍几个基本函数：

- 网格生成(meshgrid):

  [X, Y] = meshgrid(-2:.2:2, -2:.2:2);
- 规则网格下图形显示(mesh、surf)

  Z = X .* exp(-X.*X - Y.*Y);
- mesh(X,Y,Z) % 绘制网格曲面
- surf(X,Y,Z) % 绘制光滑曲面
- 非规则网格表示的图形显示(trimesh、trisurf)： 参考help.
- plot3(X,Y,Z,'option'); % 绘制参数曲线，如

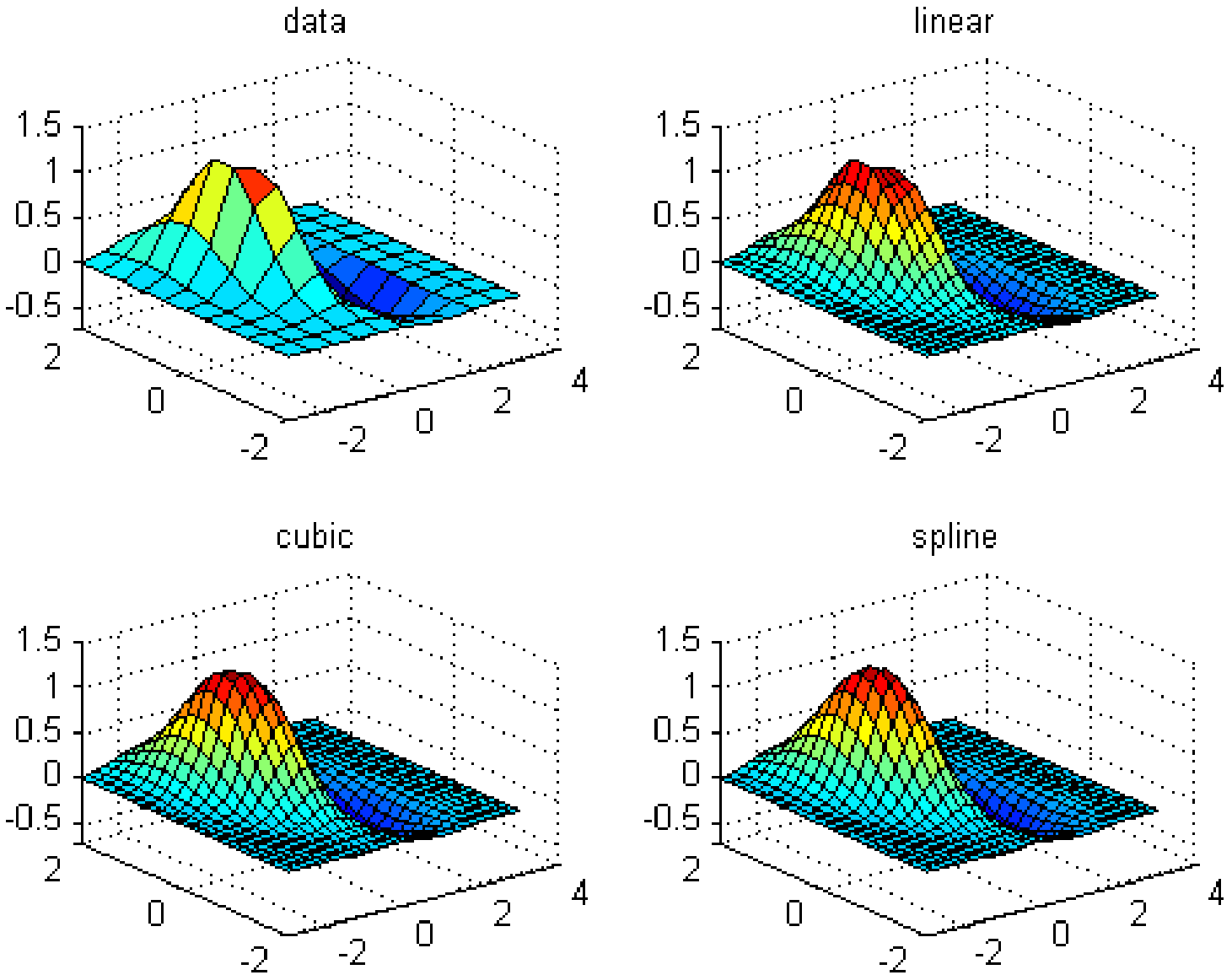
$$x = \sin(t), y = \cos(t), z = t, t \in [0, 10\pi]$$

案例：二元函数插值

在$[-3,3]\times[-2,2]$上绘制如下函数表示的曲面

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

```
[x,y] = meshgrid(-3:0.6:3, -2:0.4:2);
z = (x.*x - 2*x).*exp(-x.*x - y.*y - x.*y);
[xx,yy] = meshgrid(-3:0.2:3, -2:0.2:2);
z1 = interp2(x, y, z, xx, yy);
z2 = interp2(x, y, z, xx, yy, 'cubic');
z3 = interp2(x, y, z, xx, yy, 'spline');
surf(x,y,z);   axis([-3,4,-2,2,-0.7,1.5]);
```

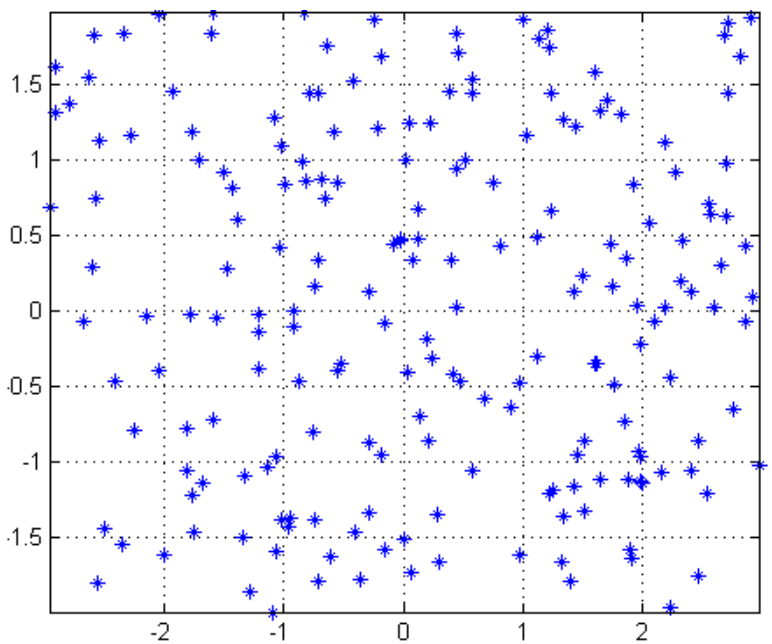


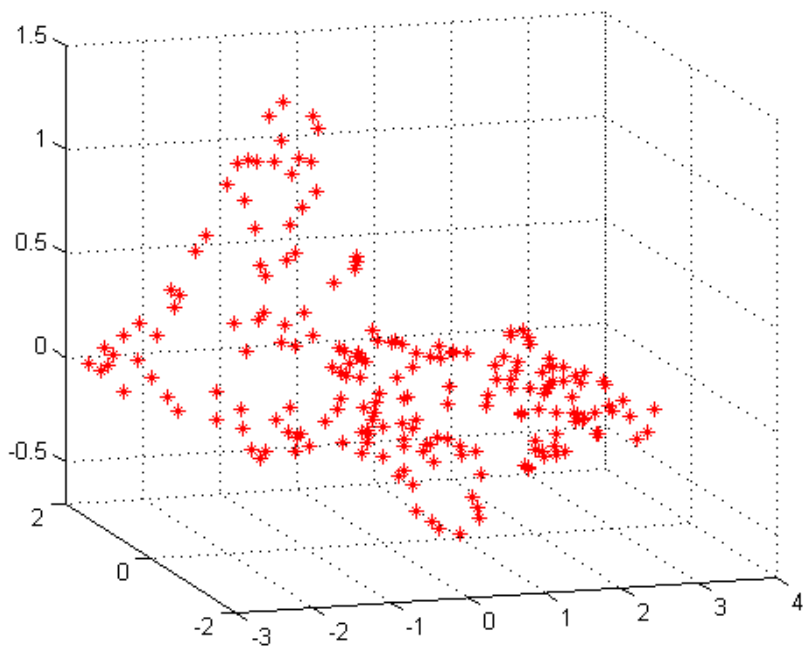
请自行绘制上述图形，并考虑如何绘制Klein瓶图像?

随机点

interp2要求型值点必须是乘积型！考虑随机点

```
1  x = -3 + 6*rand(199,1);
2  y = -2 + 4*rand(199,1);
3  z = (x.*x - 2*x).*exp(-x.*x - y.*y - x.*y);
```

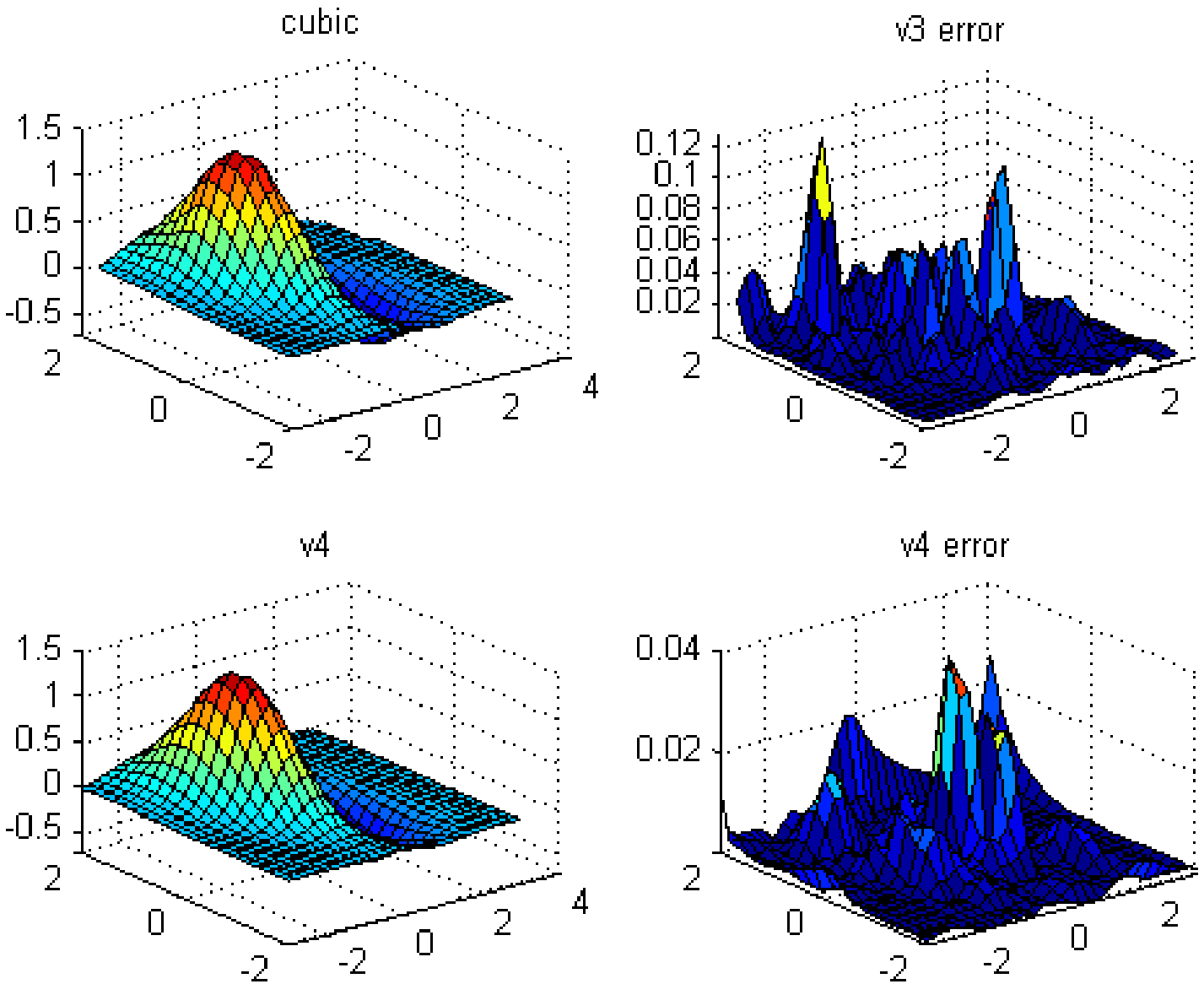




利用函数griddata(第三方工具箱函数)

```
[xx,yy] = meshgrid(-3:0.2:3, -2:0.2:2);
zz = (xx.*xx - 2*xx).*exp(-xx.*xx - yy.*yy - xx.*yy);
z3 = griddata(x, y, z, xx, yy, 'cubic');
z4 = griddata(x, y, z, xx, yy, 'v4');

subplot(2,2,1); surf(xx, yy, z3);
 title('cubic'); axis([-3,4,-2,2,-0.7,1.5]);
subplot(2,2,2); surf(xx, yy, abs(zz-z3));
 title('v3 error'); axis tight;
subplot(2,2,3); surf(xx, yy, z4);
 title('v4'); axis([-3,4,-2,2,-0.7,1.5]);
subplot(2,2,4); surf(xx, yy, abs(zz-z4));
 title('v4 error'); axis tight;
```



结论：用'v4'选项的插值结果明显优于'cubic'算法。

色彩与光照

- 色彩控制：colormap *map_name*。其中参数map_name可取

hsv	色度饱和度	colorcube	增强立方色图
hot	黑红黄白色图	jet	HSV变异图
gray	线性灰度图	prism	棱镜色图
bone	蓝色色调图	cool	青色和洋红色阴影色图
copper	线性纯铜色图	autumn	红色黄色阴影色图
pink	粉红色色图	spring	洋红黄色阴影色图
white	全白色图	winter	蓝色绿色阴影色图
flag	红白蓝黑色图	summer	绿色黄色阴影色图

- 光照控制(surf, surfl, shading)

低级绘图函数

- 更基本的绘图函数line,使用更灵活(画三角网格)
- 采用不同的形式（如直角坐标、参数、极坐标）画出单位圆$x^2 + y^2 = 1$，并用命令fill以不同的颜色填充内部。
- 尝试用多种方案将下列坐标点所表示的函数可视化

  x = {0,1,2,3,4,5,6,7,8,9,10,11}

  y = {-0.0093, -0.0040, 0.0040, 0.0140, 0.0400, 0.1107, 0.2696, 0.5748, 1.0992, 1.9304, 3.1707}×1.0e+003

向量化运算

任意给定两个向量，如$a = [1\ 2\ 3], b = [5, 6, 7]$

- 加法$c = a + b$;
- 数乘$c = \alpha * a$，其中$\alpha = 0.25$;
- 内积$c = a * b'$或$c = dot(a, b)$;
- 外积$C = a' * b$;

矩阵与向量运算

$$A = \begin{bmatrix} 1 & -1 & 2 \\ 2 & 4 & 0 \\ 3 & 0 & 2 \end{bmatrix}, \quad x = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix} \rightarrow Ax = \begin{bmatrix} -9 \\ 16 \\ -2 \end{bmatrix}$$

```
A = [1,-1,2; 2,4,0; 3,0,2]; x = [2  3  4]'; b = A*x;
```

矩阵与矩阵运算

$$B = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix}, \quad A * B = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 12 & 12 & 12 & 12 \\ 10 & 10 & 10 & 10 \end{bmatrix}$$

```
B = [2,2,2,2;2,2,2,2;2,2,2,2]; C = A*B
```

矩阵的数组运算

与矩阵运算不同在于数组运算是同维矩阵对应元素之间的运算。

1. 数组四则运算：.*与./
2. 幂(.^)、指数(exp)、对数(log)和开方(sqrt)运算
3. 逻辑关系函数运算：

   any、all、find、ianan、isnumeric、isempty、...

例: 一组三角形顶点给定(triangle.mat)，试计算它们的面积。

```
v = 0.5*abs(tri(:,2,1).*tri(:,3,2) - tri(:,3,1).*tri(:,2,2) + ...
            tri(:,3,1).*tri(:,1,2) - tri(:,1,1).*tri(:,3,2) + ...
            tri(:,1,1).*tri(:,2,2) - tri(:,2,1).*tri(:,1,2));
```

tri(i,j,k)表示第i个三角形的第j个顶点的第k个坐标。

线性方程组迭代解法

对于$A \in R_{n\times n}, b \in R_n$,且$A$非奇异的线性方程组

$$Ax = b,$$

可以采用如下迭代求解技术:

- Jacobi迭代法
- Gauss-Seidel迭代法
- 两步迭代法
- SOR迭代法

1、Jacobi迭代

$$A = D - L - U$$

$$x^{(k+1)} = D^{-1}(L+U)x^{(k)} + D^{-1}b = Bx^{(k)} + f \tag{1}$$

```
function y = jacobi(A,b,x);
D = diag(diag(A));
L = -tril(A,-1);
U = -triu(A,1);
B = D\(L+U);
f = D\b;
y = B*x + f;
while norm(y - x) > 1.0e-6
    x = y;
    y = B*x + f;
end
```

2、Gauss-Seidel迭代

$$x^{(k+1)} = (D-L)^{-1}Ux^{(k)} + (D-L)^{-1}b = Gx^{(k)} + f \qquad (2)$$

$$x^{(k+1)} = Gx^{(k)} + f$$

```
function y = seidel(A,b,x);
D = diag(diag(A));
L = -tril(A,-1);
U = -triu(A,1);
G = (D-L)\U;
f = (D-L)\b;
y = G*x + f;
while norm(y - x) > 1.0e-6
    x = y;
    y = G*x + f;
end
```

- 初值为0，精度要求为10^{-6},求解

$$\begin{cases} 10x_1 - x_2 = 9 \\ -x_1 + 10x_2 - 2x_3 = 7 \\ -2x_2 + 10x_3 = 6 \end{cases}$$

- 一般Gauss-Seidel迭代比Jacobi迭代收敛快些，但是:

$$\begin{pmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 9 \\ 7 \\ 6 \end{pmatrix}$$

G-S迭代的改进

当A对称正定时，可以用正-反结合的GS迭代以提高效率：

$$(D-L)x^{(k+\frac{1}{2})} = Ux^{(k)}+b \quad (1)$$

$$(D-U)x^{(k+1)} = Lx^{(k+\frac{1}{2})}+b \quad (2)$$

求解如下的方程组:

$$\begin{pmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & -1 \\ 0 & 3 & -1 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ 25 \\ -11 \\ 15 \end{pmatrix}$$

非线性迭代法

Example

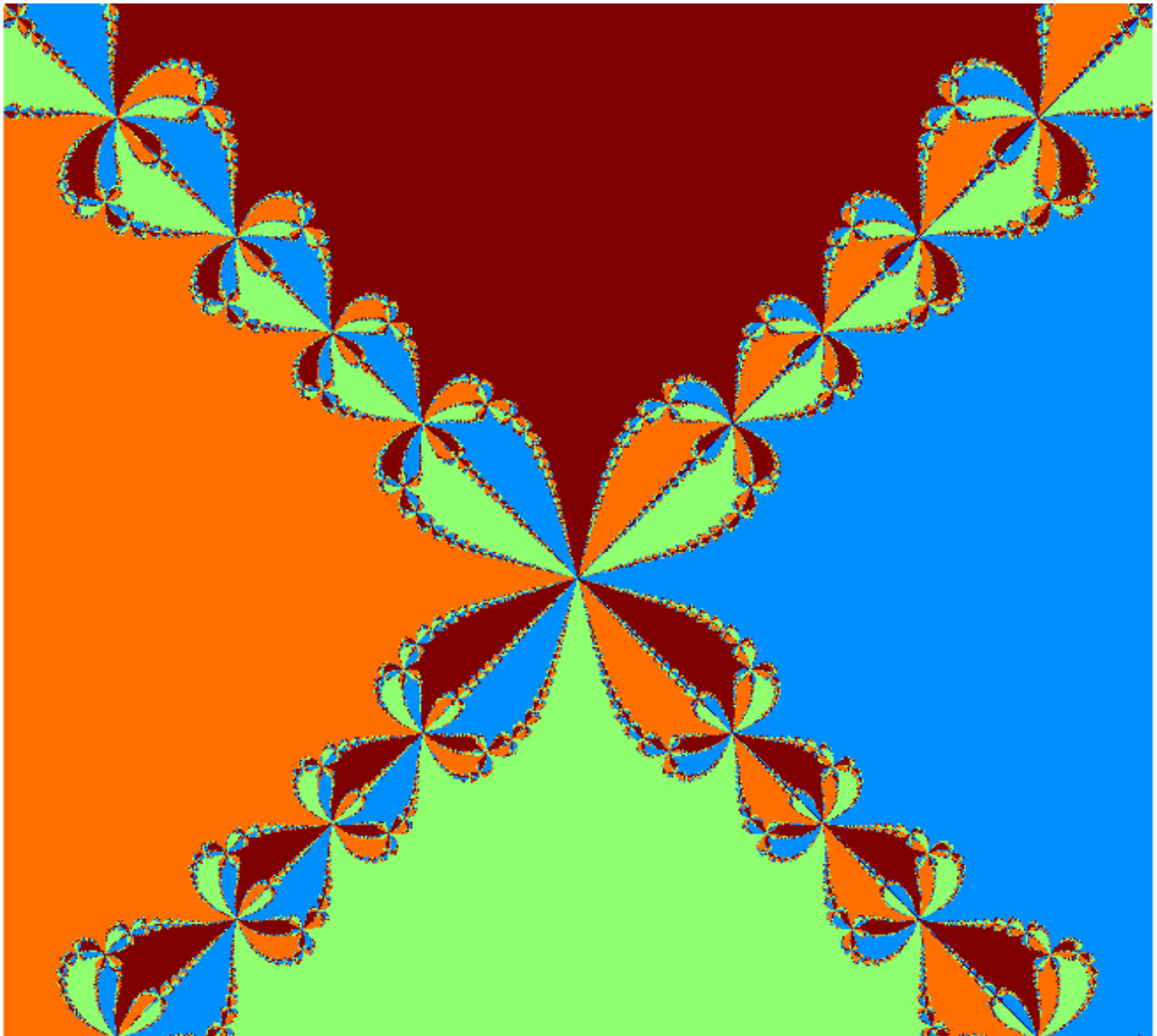
Newton迭代的向量化(newtonJulia.m) 以复平面上的任意点为初值考察非线性迭代:

$$z = z^2 + c$$

考察以复平面上$[-2,2] \times [-2,2]$内的点为初值的Newton迭代求方程$z^4 - 1 = 0$ 的根可以得到所谓的吸引盆（收敛到同一个根的所有初值点用同种颜色标示）

```
1  t = -2:0.002:2;
2  [x,y] = meshgrid(t);
3  z0 = x + i*y;
4  z1 = z0 - 0.25*(z0.^4-1)./z0.^3;
5  k = 1;
6  err = abs(z1-z0);
```

多个点同时迭代结果



- 当参数c取固定的复数,可以得到Julia集;
- 不同参数c的选取得到的形状不一样;
- 对比带循环版本,并比较它们性能。

关于矩阵的范数与条件数

MATLAB求矩阵范数： norm(A,p),其中p=1,2或inf.

条件数的计算：

- cond(A)或cond(A,2)—调用svd(A)
- cond(A,1)或cond(A,inf)—调用inv(A)
- condest(A)—估算大型稀疏矩阵的1-特征值

矩阵分解

矩阵分解是线性代数基本问题之一，MATLAB提供了

- LU分解–lu(A) => A = L(下三角矩阵)*U(上三角矩阵)
- Cholesky分解–chol(A) => 对称情形，$U = L^T$
- QR分解–qr(A)
- 特征值分解–**eig**(A)
- 奇异值分解–**svd**(A)

超定线性方程

若A是$m \times n$ 矩阵，且$m >> n$。解的存在性结论有：

1. A列满秩$\leftrightarrow$ 存在唯一
2. A秩亏损$\leftrightarrow$ 解仍存在，但不唯一

求解方法

1. 最小二乘法(正规方程方法):方程两边左乘A^T再求广义逆
2. 利用QR分解。下面是自带的函数说明(» help qr)

*[Q,R] = qr(A), where A is m-by-n, produces an m-by-n upper triangular matrix R and an m-by-m unitary matrix Q so that A = Q*R*

Example (最小二乘法)

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} \\ 0 & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} \frac{2}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & \frac{3}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & 0 & \frac{2}{\sqrt{3}} \end{bmatrix}$$

特征值(Eigenvalue)问题

对于给定的方阵A，求特征值λ和特征向量x(非零向量)

$$(A-\lambda)x=0, x\neq 0$$

这对于研究振动和线性常微分方程有重要的意义。上式表示

$$det(A-\lambda)=0$$

得到关于λ的n次特征方程，所以A有n个特征值。意味着

$$AX=X\Lambda \Rightarrow A=X\Lambda X^{-1}$$

其中X的每列是特征向量，是可逆的；Λ是由A的n个特征值为对角元素的对角矩阵。就得到了矩阵A的特征值分解，为了避免直接用特征多项式的根计算，Octave提供函数eig和eigs。

Example

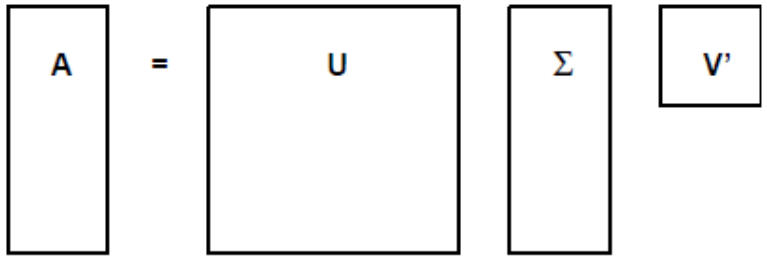
求矩阵$A = \begin{bmatrix} 133 & 6 & 135 \\ 44 & 5 & 46 \\ -88 & -6 & -90 \end{bmatrix}$的所有特征值和特征向量。

关于奇异值分解(Singular Value Decomposition)

当矩阵A非方阵，即体现了不同（维数）空间之间的变换时，相应地有非负**奇异值**σ和**奇异值向量对**u和v，满足：

$$Av = \sigma u, A^H u = \sigma v$$

在MATLAB中有：$[U, \Sigma, V] =$ svd(A)



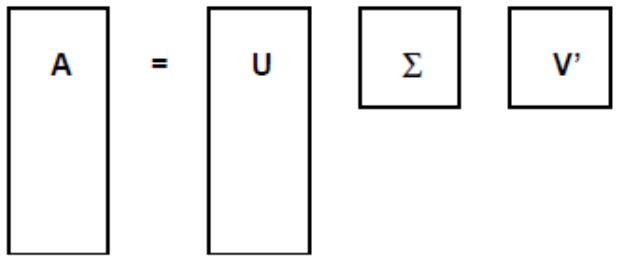


Figure: 完全和简化的奇异值分解

SVD分解的一个例子

$$A = \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix} = \begin{bmatrix} 2^{-\frac{1}{2}} & -2^{-\frac{1}{2}} \\ 2^{-\frac{1}{2}} & 2^{-\frac{1}{2}} \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 2^{-\frac{1}{2}} & -2^{-\frac{1}{2}} \\ 2^{-\frac{1}{2}} & 2^{-\frac{1}{2}} \end{bmatrix}$$

tip: 当极小化$\|A\mathbf{x} - \mathbf{b}\|_2$时,若$A$秩亏或 " 接近 " 于秩亏，常称为秩亏的最小二乘问题。SVD可计算其低维近似解。

SVD的几个应用

在求解诸如数字图像恢复、某些积分方程的解或从噪声数据中提取信号等病态问题中,常用的正则化（regularization）来解决秩亏问题.

- http://zh.wikipedia.org/wiki/奇异值分解
- http://www.cnblogs.com/LeftNotEasy/archive/2011/01/19/svd-and-applications.html

例子：基于SVD的图像压缩

```
load clown.mat;    colormap('gray');
[u,s,v]=svd(X);   k = 15;
uk=u(:,1:k);sk=s(1:k,1:k);vk=v(:,1:k);
Ak=uk*sk*vk'; image(Ak);
ita =norm(X-Ak,'fro')./norm(X,'fro')
[m,n]=size(X);   omega=(m+n).*k/(m.*n)
```

Everything is a matrix

- 线性方程组
- 向量化运算
- 数字图像与视频

稀疏矩阵

大型稀疏线性方程组来源于广大数值模拟问题,如有限元方法。
具有和一般的线性方程组通用的形式:

$$Ax = b$$

- A的大部分元素为0 !
- A的稀疏度:$\frac{nnz}{n^2}$
- 主要关心的问题: 如何高效地求解该线性系统

稀疏矩阵的基本操作

- S = sparse(i,j,s,m,n); 转化成满矩阵 A = full(S);
- 数量关系：[i,j,s] = find(S) 与[m,n] = size(S);
- 带状矩阵是特殊类型的稀疏矩阵，其他生成稀疏(带状)矩阵的函数如S = spdiags([a,b,c],[-1,0,1],n,n);
- 从文件导入：load + spconvert
- 图示稀疏矩阵：spy(S)

例:采用多种途径建立如下形式的稀疏矩阵，并比较与满矩阵计算性能:

$$A = \begin{bmatrix} -4 & 1 & & \\ 1 & -4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -4 \end{bmatrix}_{n\times n}$$

稀疏线性方程组迭代解法

经验表明：直接法（左除）为10万阶左右的稀疏线性方程组提供了高效率的求解方法；而规模超过100万阶的，迭代法是比较合理的选择。Matlab/Octave提供了一系列基于所谓Krylov子空间的迭代算法，它们包括：

- cgs，pcg：A对称正定
- bicg,bicgstab
- gmres：A可非对称
- minres

预条件处理(Preconditioner)可以大大改善迭代的收敛速度

从数据文件导入稀疏矩阵，比较不同规模下各种求解法的性能。

对应的一些操作函数

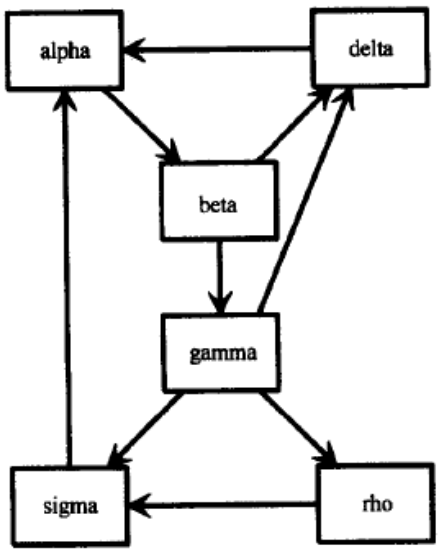
- 代数特征/奇异值：eigs, svds;
- 不完全矩阵分解：luinc, cholinc;
- 矩阵范数：normest和条件数：condest;

案例：PageRank原理

PageRank算法是Larry Page和Sergey Brin提出的算法，也因为Google的成功而被广泛关注。

- 将网页按照被链接/被访问的几率进行排序是合理的；
- 从一组基本的网页出发，追踪网页间的超链接关系，可以利用Markov的原理计算出每个网页被访问到的概率；

从图论的角度看，网络的邻接矩阵元素$G_{ij} = 1$表示从网页j链接到网页i，否则为0；容易将G规范化为列和1的转移概率矩阵，使方程$x = Ax$存在非零解。向量x的初始值可选为和1的PageRank。



-

用**幂法**、**解方程法**和**逆迭代法**求解上述问题。

稀疏矩阵练习

试求解给定格式的稀疏线性方程组，要求

- 比较左除、cgs、bicg、gmres、pcg等求解方法的效率；
- 对所给4个不同规模的线性系统进行测试，写下你的结论；
- 数据从这里下载http://www.math.zju.edu.cn/xlhu/dat/mat/testmat4.html
- 编写程序导入".mat"中稀疏矩阵。文件是按照如下的三元组格式给出的：

```
nrow ncol nnz          #稀疏矩阵的行数、列数以及非零元个数,nrow=
i(1) j(1) v(1)         #第一个非零元的行、列以及数值
i(2) j(2) v(2)
...
i(nnz) j(nnz) v(nnz)   #最后一个非零元的行、列以及数值
nrhs                   #右端项向量长度nrhs=nrow
```

脚本调试

脚本调试除了找到程序的语法问题之外，更重要的是发现潜在的逻辑错误。几个关键操作：

- 断点设置
- 逐步执行
- 变量查看
- 逻辑分析

尝试练习调试以解决程序中的逻辑或算法错误！

程序性能分析工具

程序性能的提高方法是多样的，在MATLAB程序设计中改进程序性能有其自身的特点。MATLAB除了提供诸如tic–toc时间统计函数外，也提供了一个更为实用的程序性能监测工具profile：

```
profile on;
your_command_here;
profile viewer;
```

程序运行结束后可以跳出一个运行时每个函数消耗的时间统计。

“从现在开始，一个月后，如果我再看这些代码，我能否理解他们是干什么的？”

—— Steve Lord

代码优化常用原则

- 统一编程风格，提高程序的可读性；
- 利用MATLAB向量化加快运行速度（减少循环嵌套）；
- 循环中尽量避免选择结构(polyinterp.m, find函数)；
- 占用内存大的变量（矩阵）尽可能一次性分配内存；
- 避免在频率高的循环中调用绘图函数和符号计算；
- 利用MATLAB与c/fortran的接口混合编程提高运算效率；

本文关于编程风格的内容源自于《MATLAB 编程风格指南》感谢Genial的翻译工作，原作者是Richard Johnson

可读性之–变量命名规则

- 变量名反应其意义或用途,小写开头大小写混合

  采用： html，isUsaSpecific，checkTiffFromat

  避免：hTML，isUSASpecific，checkTIFFFormat;
- 前缀n和m的使用：nFiles，nSegments, mRows;
- 循环变量以i、j、k 等为前缀，当有复数时禁用i、j;
- 常数（全局变量）大写+下划线:

  MAX_ITERATIONS，COLOR_RED;
- 函数名小写且具有意义（max、gcd等数学函数除外），

  如采用computetotalwidth(); 避免使用compwid()

可读性之–代码格式

- 一行代码应该只包含一个可执行语句（允许JIT加速）；
- 短的单个if，for 或者while语句可以写在一行
- 表达式中适当增加空格或空白
  - &、+、*、、=等常规的操作符两边，逗号后面空格；
  - 将较长的数学公式分行对齐；
  - 块（block）内部的一个逻辑组语句应该通过一个空白行；
  - 块（blocks）之间应该用个多行分隔。
- 注释简洁易读(注释不能改变写得很糟糕的代码效果)
  - 函数头注释应支持help查询，最后一句重申函数语句行；
  - 函数头注释应该讨论对输入参数的特殊要求等；
  - 函数头注释通常包括版权以及修改日期等信息；
  - 所有的注释语句应该用英语(Recommend)

常见MATLAB问题汇总

1. 什么是MATLAB? 它有哪些功能?
2. 向量和矩阵有何关系? 哪些方式可创建4×4的矩阵?
3. 怎样删除矩阵的一行或一列? 离散的数据怎样拼接成矩阵?
4. MATLAB能开多大的数组? 矩阵和数组相同吗?
5. MATLAB如何寻找需要执行命令?
6. 如何查找相关函数的功能和调用格式?
7. 运行时为什么在屏幕上无结果显示?
8. 为什么小数点后只显示四位? 如何中断当前运算?
9. Matlab/Octave绘图的基本步骤有哪些? 怎样将图形用于LaTeX?
10. 如何计算程序运行的时间? 如何格式化输出?
11. 如何避免被0除? nan是怎样产生的?
12. Matlab/Octave高效编程要注意哪些问题? 与c的主要区别?

作业与上机导引

1. 线性方程组求解的lu分解法和迭代法：利用Matlab/Octave各生成100x100和2000x2000的满线性方程组进行测试
2. 非线性方程迭代，类似课件中Julia分形图的绘制。并给出三次和五次代数多项式对应的图形。
3. 关于稀疏矩阵。从sparse*.mat导入不同尺寸的稀疏线性方程组并做如下的测试
   - 用左除以及不同的迭代法（cgs/pcg/gmres/bicg等）求解
   - 比较直接法和迭代法的效率（运算时间），并记录四个不同规模测试矩阵消耗的CPU时间
   - 记录他们消耗的内存（可利用windows任务管理器查看进程MATLAB占用内存，若时间较短可以循环多次取平均）
   - 从以上数据你可以得到什么结论?
4. PageRank算法原理与实践，参考C. Moler博士的教材