

Breu manual de referencia per Matlab i per Octave:

Jaume Amoros, UPC, Barcelona
<http://www-mal.upc.es/~amoros>

2003/12/9

Entrar a Matlab u Octave: fer doble click a la icona

Directori de treball del programa (en ell podem deixar dades par a que les llegeixi, o recuperar les solucions que haguem escrit):

- En Matlab: es pot triar
 - En Octave per Unix/Linux: des d'on es cridi al programa
 - En Octave 2.1.36 per Windows: es
- C:\Archivos de Programa\GNU Octave 2.1.36\octave_files

Demandar ajuda: fer

help

Sortir de Matlab o d'Octave:

exit

GRAMATICA ELEMENTAL:

Al apretar Intro executa les ordres
Si l'ordre acaba en ; no ensenya resultat
Si no acaba en ; el mostra per pantalla

Per a que ens ensenyi el contingut de la variable A, fer

A

ALGEBRA LINEAL
=====

MANIPULACIONS ELEMENTALS:

Definir una matriu (files separades per ; elements de la fila per ,):

```
A=[11,12;21,22];
```

Element (1,2) de la matriu A (1a fila, 2a columna):

```
A(1,2)
```

Matriu formada per les files 2 a 5 i columnes 1 a 8 de A:

```
B=A(2:5,1:8);
```

Id=Identitat 15x15:

```
Id=eye(15);
```

C=matriu de zeros 3x3, D matriu de zeros 4x6

```
C=zeros(3);  
D=zeros(4,6);
```

Tamany de la matriu A:

```
size(A)
```

Es crea nova matriu $m \times n$ B2 alterant mides del vector $m \times n$ B:

```
B2=reshape(B,m,n)
```

Definir un vector de 7 components (vector= matriu $1 \times n$)

```
v=[1 2 3 4 5 6 7];
```

Reordenar els components d'un vector v de mes petit a mes gran en cv , i guardar en perm la permutacio que s'ha fet a v ($cv(i)=v(perm(i))$)

```
[cv,perm]=sort(v);
```

Transposar una matriu A:

```
T=A';
```

Element maxim d'un vector v :

```
top=max(v);
```

Coefficient maxim d'una matriu A:

```
top=max(max(A));
```

Producte escalar de dos vectors u, v :

```
p=dot(u,v);
```

Producte vectorial de dos vectors u, v de \mathbb{R}^3 :

```
pv=cross(u,v)
```

CALCULS AVANCATS:

Invertir una matriu A:

```
B=inv(A);
```

determinant de la matriu A:

```
det(A)
```

Calcula el rang de A:

```
rank(A)
```

Calcula el rang de A identificant a 0 valors singulars menors que 10^{-7} :

```
r=rank(A,1e-7);
```

Porta la matriu A a forma esglaonada (sense pivotatge per columnes):

```
E=rref(A); %NOMES EN MATLAB
```

Porta A a forma esglaonada identificant a zero els valors singulars menors que 10^{-7}

```
E=rref(A,1e-7); %NOMES EN MATLAB
```

Porta A a forma esglaonada identificant a zero els valors singulars menors que 10^{-7} i a sobre torna vector b tal que $b(i)$ =columna i-esima d'una base del subespai de les columnes de A:

```
[E,b]=rref(A,1e-7); %NOMES EN MATLAB
```

Calcula una base ortonormal del nucli de A com a columnes de Z:

```
Z=null(A);
```

Calcula una base ortonormal de la imatge de A com a columnes de Q (ull! NO es la Q de desc QR):

```
Q=orth(A);
```

Calcula la descomposicio en valors singulars de A (U,V ortogonals, S la forma diagonal):

```
[U,S,V]=svd(A);
```

Calcula la descomposicio QR de la matriu A (Q ortogonal, R triang. superior) (A pot ser rectangular plantada):

```
[Q,R]=qr(A);
```

Calcula la solucio x del sistema $Ax=b$ (sistemes matricials tambe) ULL!:

Si el sistema es compatible determinat dona la solucio
Si el sistema es compatible indeterminat dona UNA solucio i no avisa (la primera que li troba l'algoritme, pivotatge complet)
Si el sistema es incompatible pero sobredeterminat dona solucio per minims quadrats, i no avisa
Si el sistema es incompatible i la matriu no te rang maximal dona missatge de que no hi ha solucio

```
x=A\b;
```

Calcula la pseudo-inversa de A (Moore-Penrose, minims quadrats)

```
P=pinv(A);
```

Calcula la forma diagonal D de A i la matriu C que te per columnes la base de VEPs (canvi de base de veps a canonica) (si A es simetrica dona base de VEPs ortonormal):

```
[C,D]=eig(A);
```

Calcula una llista de valors propis de A (en columna)

```
vaps=eig(A);
```

Calcula el vector p de coeficients del polinomi caracteristic $\det(t \cdot \text{Id} - A)$ (coeficients ordenats de grau maxim a zero):

```
p=poly(A);
```

Calcula el nombre de condicio de A (VAP maxim/VAP minim) (lent!!)

```
c=cond(A);
```

Calcula l'invers del nombre de condicio_1 per un algoritme Lapack mes eficient

```
r=rcond(A); %NOMES EN MATLAB
```

Calcula la norma de la matriu (1,2,infinit) (tambe de vectors)

```
n1=norm(A,1);  
n2=norm(A,2);  
ni=norm(A,inf);
```

COMPLEXES:

Opera amb nombres complexos (es retornen resultats en rectangular)

```
(1-i)^4, (5-7i)/(2+i)
```

Conjuga el nombre complex r

```
rc=conj(r);
```

Calcula el modul i argument (en $(-\pi, \pi]$) de z

```
r=abs(z);  
theta=arg(z);
```

POLINOMIS:

Guardem en un vector p el polinomi $p(x)=3x^2-2x+11$

```
p=[3,-2,11]
```

Dona un vector columna amb els zeros del polinomi p (guardat com a vector, coeficients ordenats de grau mes gran a mes petit)

```
s=roots(p);
```

Multipliqua els polinomis p i q

```
pr=conv(p,q);
```

Divideix el polinomi p per d, guardant el quocient a q i el reste a r

```
[q,r]=deconv(p,d);
```

Avalua el polinomi p en la matriu quadrada A:

```
C=polyvalm(p,A);
```

TRIGONOMETRIA:

Funcions trigonometriques: sin(t),cos(t),tan(t) (t en rd)

Arc cosinus de a (en rd) (per a en [-1,1]: $0 \leq \text{angle} \leq \pi$)
(per $a < -1, a > 1, a$ complex no real: arc cosinus complex)

```
ang=acos(a);
```

Arc sinus de b (en rd, funciona com l'arc cosinus)

```
ang2=asin(b);
```

Arc tangent de c (en rd, per c real torna l'angle a $[-\pi/2, \pi/2]$)

```
ang3=atan(c);
```

OPERACIONS EN MODE MATRICIAL:

Precaucio: aquestes operacions, indicades amb un punt, NO SON les operacions aritmetiques habituals amb matrius:

Divideix cada coeficient (i,j) de la matriu P pel coeficient (i,j) de la matriu D (P i Q son de la mateixa mida):

```
Q=P./D;
```

Idem per multiplicar cada coeficient de P pel corresponent de Q:

```
M=P.*Q;
```

Crea una nova matriu que te per terme (i,j) l'arrel quadrada del coeficient (i,j) de P:

```
S=sqrt(P);
```

CALCUL
=====

(indiquem nomes l'operacio mes tipica)

Crea un vector fila x amb 11 valors equiespaiats de 0 a 2:

```
x=linspace(0,2,11);
```

Calcula el vector y(x) de valors de la funcio tan(sqrt(x)) pel vector de valors de x donat:

```
y=tan(sqrt(x));
```

GRAFICS

=====

La diferencia entre Matlab i Octave es mes substancial.

Dibuixa el grafic de valors del vector y:

```
plot(y);
```

Dibuixa el grafic de valors de y amb eix OX el vector x:

```
plot(x,y);
```

Dibuixa el grafic de la funcio $R^2 \rightarrow R$ amb valors donats per la matriu Jv2 (eix x ve al front, eix y ortogonal a la pantalla):

```
surf(Jv2); %NOMES EN MATLAB
```

Dibuixa un camp vectorial que te components Jx2', Jy2'

```
quiver(Jx2',Jy2'); %NOMES EN MATLAB
```

Demana que tots els dibuixos que segueixin es superposin en la mateixa figura

```
hold on
```

Demana que deixi de superposar dibuixos i reinicii la figura actual

```
hold off
```

GRAFICS PER GNUPLOT AMB OCTAVE:

Dibuixa el graf de la funcio $R^2 \rightarrow R$ amb valors donats per la matriu Jv2 (eix x ve al front, eix y ortogonal a la pantalla):

```
gsplot(Jv2);
```

Canvia el punt de vista (per defecte el pt de vista es 60,30) (els angles: 0,0: eixos x,y en la pantalla de manera standard, z surt fora. El primer angle es la rotacio en OX, el segon es la rotacio en l'eix OZ despres de la primera)

```
gset view 45,30
```

Posa els intervals dels eixos x,y,z:

```
axis([0,25,0,40,0,100]);
```

Posa etiquetes en els eixos:

```
xlabel('eix x');  
ylabel('eix y');  
zlabel('A/cm^2');
```

Redibuixa el graf amb els nous parametres de pt de vista, eixos...
(en Octave 2.1.36 per Windows no fa falta, els canvis es fan al picar-los)

```
shg
```

PROGRAMACIO: INSTRUCCIONS DE CONTROL
=====

AVIS: les operacions en mode vectorial de Octave i Matlab permeten fer la mateixa feina que un bucle mes rapidament en moltes ocasions.

Bucle: calcula una successio de rectes interpoladores

```
for i=1:n,  
    p(i,:)=polyfit(x(i,:),y(i,:),1);  
end;
```

Condicional while: avanc,a en una llista fins que troba un element no nul

```
while (v(i)==0)  
    i=i+1;  
endwhile;
```

Condicional if: Si a es negatiu li canvia el signe

```
if (a<0)  
    a=-a;  
endif;
```

Condicional if else: mira si a es positiu o negatiu i incrementa el contador que toqui

```
if (a>=0)  
    positius=positius+1;  
else  
    negatius=negatius+1;  
endif;
```

Cadena if elseif: mirem si n es parell i si es multiple de quatre:

```
if (rem(n,2)!=0)  
    printf("Nombre senar.\n");  
elseif (rem(n,4)==0)
```

```
    printf("Nombre multiple de 4.\n");
else
    printf("Nombre parell, no multiple de 4.\n")
endif;
```

Instruccions aniuades: compta igualtats entre coeficients de A i B

```
coinc=0;
for i=1:ma,
    for j=1:na,
        for k=1:mb,
            for l=i:nb,
                if (A(i,j)==B(k,l))
                    coinc=coinc+1;
                endif;
            end;
        end;
    end;
end;
```

CREACIO DE FUNCIONS I PROGRAMES

=====

NOM I UBICACIO

Els executables Octave/Matlab han de tenir terminacio .m (per exemple rouche.m grafica.m ...), i han d'estar al directori de treball del programa (indicat a l'inici d'aquest fitxer).

PROGRAMES (SCRIPTS):

Un programa (o script) es un fitxer.m que conte qualsevol llista de comandes que tinguin sentit en Octave/matlab.

Executem el programa graf_2var.m

```
graf_2var
```

Discussio dels programes/scripts:

Es aconsellable intercalar linees de comentaris explicant que fa cada part del programa, que son les variables... Les linees de comentari comencen per %

Exemple de script: graf_2var.m

Els scripts tenen dos inconvenients:

- Les variables que fan anar son les de la sessio d'Octave oberta, poden trepitjar alguna variable important sense que ens n'adonem.

- Per a passar dades a un script cal tenir-les ja posades en variables Octave, o demanar-les com a l'apartat LECTURA I ESCRIPURA.

I a vegades, un avantatge: son mes facilis d'escriure que les funcions.

FUNCIONS:

Una funcio es tambe un fitxer.m que conte una llista de comandes Octave/matlab, pero te dos avantatges importants respecte dels scripts:

- Li podem passar dades com a arguments al cridar-la
- La funcio només podra alterar les variables que li indiquem en la nostra sessio d'octave, i cap més.

Executem la funcio allisat.m, passant-li com a parametres una matriu M2 i un escalar tall, i guardem com a M4,T les dues matrius que allisat retorna:

```
[M4,T]=allisat(M2,tall);
```

(veure la funcio allisat.m als exemples per saber que fa, vigileu a escriure les dades d'entrada i sortida en l'ordre especificat per la funcio)

LECTURA I ESCRIPTURA DE DADES

=====

DE/A FITXER, ESTIL MATLAB

Guarda les variables A,C,D en matrius.txt com a texte (NO com a executable Octave/Matlab)

```
save -ascii matrius.txt A C D
```

Llegeix les variables C,D de matrius.txt com a texte (si omitim -ascii intentara detectar per si sol el format en que son les dades)

```
load -ascii matrius.txt C D
```

DE/A FITXER, ESTIL C

Notacio general: \n vol dir saltar de linea, en llegir o en escriure

ESCRITURA (texte, executable Octave/Matlab)

Guardarem variables/matrius amb titol, en texte per a poder llegir-les a ull, pero en format executable d'Octave/Matlab per a poder carregar-les totes de manera rapida (veure seguent punt: LECTURA)

Obre el fitxer resultats.m per a escriure en format de texte (PRECAUCIO: si resultats.m existia, el seu contingut es perdra)

```
results=fopen("resultats.m","w");
```

Escriu un titol amb % devant per a que Octave el reconegui com a comentari

```
fprintf(results,"%s\n",'Matriu B calculada');
```

Escriu el comencament de la matriu

```
fprintf(results,"%s\n",'B=[');
```

Escriu la matriu B de nombres reals (que te tres columnes),
coeficients separats per comes

```
fprintf(results,"%g,%g,%g\n",B);
```

Escriu el final de la matriu

```
fprintf(results,"%s\n",''];
```

Tanca el fitxer, que queda en el directori de treball

```
fclose(results);
```

LECTURA (texte, executable Octave/Matlab)

Llegeix totes les dades contingudes en el fitxer resultats.m,
executable octave/matlab en el directori de treball:

```
resultats
```

LECTURA (texte, format arbitrari, estil C)

Obre el fitxer dades.txt per a llegir en format de text

```
entrada=fopen("dades.txt","r");
```

Llegeix una linea de titol en text i passa d'ella

```
fscanf(entrada,"%s\n");
```

Llegeix un nombre real/enter n de la primera linea

```
n=fscanf(entrada,"%g\n","C");
```

Llegeix una matriu B de 3 columnes i n files
guardada una fila per linea, coefs separats per comes

```
for i=1:n,  
[B(i,1),B(i,2),B(i,3)]=fscanf(entrada,"%g,%g,%g\n","C");  
end;
```

Tanca el fitxer

```
fclose(entrada);
```

PER PANTALLA:

L'entrada de dades per pantalla (comandes scanf,input...)
encara no esta ben resolta en Octave 2.1.x

Sortida: amb el printf, va com en C.

BIBLIOGRAFIA

=====

Es poden fer moltes mes coses amb Octave i Matlab,
i algunes de les que han sortit aqui es poden fer
millor, tot i que menys senzillament. Per tot
això useu l'ajuda del programa, els manuals

i tractats de programacio.

Pagina web d'Octave

<http://www.octave.org>

Conte el manual, octave_manual.pdf, bastant complet,
també disponible a <http://www-mal.upc.es/~amoros/octave.html>

Pagina web de Matlab:

<http://www.mathworks.com>

L'ajuda online del Matlab es molt completa.