Resolução de sistema de equações lineares

Dado o sistema de equações abaixo:

$0.2x + 0.5y = 1$
$0.3x + 0.2y = 0.6$

Duas técnicas são possíveis, porém uma é mais correta do que a outra...

a) solução matricial:

Podemos representar as equações acima em sua forma matricial:

$A\ X\ -\ b = 0$
$A\ X\ = b$
$A^{-1}\ A\ X = A^{-1}\ b$
$I\ X\ =\ A^{-1}\ b$
$X\ = A^{-1}\ b$

Onde:

I = matriz identidade

$A = \begin{matrix} 0.2 & 0.5 \\ 0.3 & 0.2 \end{matrix}$

$B = \begin{matrix} 1.0 \\ 0.6 \end{matrix}$

Então, a solução Matlab fica:

```
% Resolucao de sistema de equacoes lineares:
%
%  0.2x + 0.5y = 1
%  0.3x + 0.2y = 0.6
%

A=[0.2 0.5
   0.3 0.2];

B=[1.0
   0.6];

x=A\B
```

A última linha faz a operação “X $= A^{-1}$ b” . Também poderíamos ter usado o comando “inv” do Matlab:

```
x=inv(A)*B
```

b) solução por otimização:

Outro modo de pensar a solução desse sistema de equações lineares seria aborda-lo como um problema de otimização: desejo obter o melhor par “x,y” que resolva o sistema. As funções de otimização do Matlab buscam a solução que “zera” uma equação. Ou seja, os valores de x e y que levam ambas as equações a zero.

Assim, vamos re-escrever as equações tornando o lado direito igual a zero:

$0.2x + 0.5y = 1$
$0.3x + 0.2y = 0.6$

$0.2x + 0.5y – 1 = 0$
$0.3x + 0.2y - 0.6 = 0$

Usando “fsolve”, o arquivo principal fica:

```
options = optimset('Display','iter');
X = fsolve('bio',[2 1],options)
```

Na primeira linha declaramos que desejamos usar uma opção avançada do “fsolve”. Essa opção se chama “mostrar as iterações do calculo”, ou seja, mostrar a rota feita entre o chute inicial e a solução encontrada. Para conhecer outras opções avançadas disponíveis consulte o help do comando “optimset”.

A segunda linha faz a chamada de fsolve propriamente. O primeiro argumento é o nome do arquivo de função que contém a equação (‘bio’). O segundo argumento são os chutes iniciais (“2” para x, e “1” para y). O terceiro argumento passa a variável “options”, criada na linha anterior com as opções avançadas.

A função “bio” terá a seguinte codificação:

```
function [fobj] = bio(par),
 x=par(1);
 y=par(2);

eq1= (x*0.2) +  (y*0.5) - 1;
eq2= (x*0.3) +  (y*0.2) - 0.6;

fobj=(eq1^2)+(eq2^2);
```

Essa função recebe dois parâmetros (“x” e “y”), e são esses parâmetros que devem ser manipulados por fsolve de modo a encontrar uma combinação deles que permita zerar ambas as equações. O objetivo, “zerar ambas as equações”, é explicado ao fsolve através da fórmula `fobj=(eq1^2)+(eq2^2);` Ou seja, no caso ideal, eq1 e eq2 são zero, logo fobj é zero, e o par xy foi encontrado. Em qualquer outra situação, fobj é maior que zero, o que indica ao fsolve que a solução ainda não foi encontrada.

Mas essas duas abordagens levam a mesma resposta? **<u>Não!!!!</u>** A resposta <u>correta</u> é dada pela primeira abordagem. A resposta encontrada via fsolve é a solução de um problema de minimização. Assim, vejamos:

Dado os chutes iniciais 0.5 e 2:

Iteration	Func-count	Residual	Step-size	derivative
1	2	0.00015625	1	-2.02e-006
2	8	0.000152272	1	-1.96e-006
3	14	0.00013496	0.0987	-5.34e-005
4	20	0.000125546	0.071	-1.55e-007
5	26	0.000121676	0.031	-1.58e-007
6	32	0.000121338	0.00277	-9.5e-007
7	38	0.000120939	0.00329	-1.83e-008
8	44	0.00012047	0.00389	-2.21e-008
9	50	0.000119919	0.00458	-2.9e-008
10	56	0.000119275	0.00537	-5.12e-008
11	62	0.000118528	0.00628	-8.03e-008
12	68	0.000117664	0.0073	-4.28e-008
13	74	0.000116672	0.00844	-3.71e-008

14	80	0.000115544	0.0097	-3.88e-008
15	86	0.00011427	0.011	-4.26e-008
16	92	0.000112849	0.0125	-4.76e-008
17	98	0.000111281	0.0139	-5.36e-008
18	104	0.000109575	0.0154	-6.03e-008
19	110	0.000107745	0.0168	-6.75e-008
20	116	0.000105814	0.018	-7.49e-008
21	123	9.47285e-005	0.105	-4.53e-006
22	129	9.33136e-005	0.0148	-2.46e-006
23	135	9.18125e-005	0.0161	-5.8e-008
24	142	8.36486e-005	0.0893	-3.27e-006
25	148	8.25152e-005	0.0135	-1.76e-006
26	154	8.13068e-005	0.0147	-4.65e-008
27	160	8.00359e-005	0.0157	-5.07e-008
28	167	7.2968e-005	0.0887	-2.84e-006
29	173	7.1995e-005	0.0132	-1.53e-006
30	179	7.0967e-005	0.0143	-4.1e-008
31	186	6.55224e-005	0.077	-2.16e-006
32	192	6.47182e-005	0.0122	-1.16e-006
33	198	6.38653e-005	0.0132	-3.39e-008

Maximum number of function evaluations exceeded
Increase OPTIONS.maxFunEvals

X =   0.5722   1.7163

A solução encontrada foi: x = 0.5722 e   y = 1.7163, com uma função objetivo (fobj) valendo 6.38653e-005. Isso é quase zero, mas não é zero!

Resolvendo pelo método **a)** temos: x = 0.9091 e y = 1.6364. Essa é a resposta “matemática” do sistema.

Uma observação final: métodos de minimização como o usado no item **b)** podem gerar respostas diferentes para diferentes chutes iniciais. Experimente mudar o chute inicial e veja como a resposta muda!