Aula 05

Bibliografia: Apresentação

Cláudio R. Lucinda

FEA-RP/USP

Estrutura da Aula

1. MATLAB
   - Detalhes Operacionais
   - Funções
   - Matemática Simbólica
   - Otimização

MATLAB-Importando Dados

- `readtable` - Para importar dados organizados em colunas em uma tabela
- `csvread` - Importar um arquivo separado por vírgula em uma matriz.
- `dlmread` - Importar um arquivo em que os campos estão separados por um delimitador
- `textscan` - Importar um arquivo de texto em um `cell array`
- Import Wizard - o assistente para a importação de dados
- OBS: Todos eles possuem os análogos para a geração dos dados

MATLAB- `path`

- Vamos agora falar sobre os detalhes operacionais.
- O MATLAB consiste de um arquivo executável relativamente pequeno, chamado de `kernel` e uma coleção de muitas, MUITAS, funções no HD do seu micro. Depois que você instalou direitinho o programa, ele sabe onde encontrar as funções dele.
- Existem muitas situações, todavia, em que você pode exigir que o MATLAB interaja com as funções que você criou. Como podemos ensinar pro MATLAB onde estão os seus arquivos?
- O MATLAB armazena uma lista de diretórios em uma variável interna chamada `path`.
- Na hora de permitir que o MATLAB use uma das suas funções, é nesta variável que ele vai olhar pra saber onde vai

MATLAB- `path` - (II)

- Existe um jeito de garantir que os seus comandos não ficam perdidos, que é alterando o conteúdo da variável `path`
- Por exemplo, os comandos abaixo adicionam o diretório `c:\user\home\John\` no `path`

```
>> P=path;
>> path(P,'c:\user\home\John\')
```

MATLAB- `cd`

- O MATLAB tem sempre um diretório de trabalho. Para ver o que tem neste diretório, você pode usar os comandos `what` ou `dir` no comando do MATLAB.
- Para saber qual é o diretório de trabalho, dependendo da versão ele é apresentado no alto da tela, ou você pode usar o comando `cd`
- Você pode usar o comando `cd` para mudar o diretório. O código abaixo dá um exemplo disso.
- Nota: Mudar o diretório de trabalhos significa que o MATLAB vai achar seus códigos lá – neste sentido, é equivalente ao `path` de antes. No entanto, ele é melhor porque pode ser usado como base de referências relativas.

```
>> cd 'c:\user\home\John\';
```

MATLAB- Detalhes Operacionais

- Bom, após a aula passada com uma breve introdução ao MATLAB, hoje iremos começar com alguns detalhes importantes.
- O comando `diary('file')` instrui o MATLAB a registrar tudo o que é feito na janela do MATLAB e a salvar os resultados no arquivo denominado `'file'`. Ao digitar `diary on` ou `diary offvocê` alterna o registro. Arquivos de diários antigos podem ser visualizados por meio de um editor de texto.
- No MATLAB, eles podem ser visualizados com o comando `type file`.

MATLAB- Funções

- Para que um arquivo-m seja uma função, é preciso iniciar com a palavra `function` seguida pelas variáveis de saída entre colchetes, o nome da função e as variáveis de entrada.

Esta função Pega como argumento uma matriz `A` e retorna o produto matricial `C`.

```
function [C] = mult(A)
r = rank(A);
C = A'*A;
```

MATLAB- Matemática Simbólica

- Podemos também usar o MATLAB para manipular variáveis simbólicas – se a sua versão possui o Toolbox de Matemática Simbólica.
- Por exemplo, o código abaixo simplifica a função $\sin x^2 + \cos x^2$
- Inicialmente vamos precisar explicar que os argumentos da função são variáveis simbólicas.
- Depois, é só construir a função

```
>>syms x
>>simplify((sin(x))^2+(cos(x))^2)
ans =
1
```

MATLAB- Matemática Simbólica

```
>>syms x
>>num = 3*x^2 + 6*x -1;
>>denom = x^2 + x - 3;
>>f = num/denom
f =
(3*x^2 + 6*x - 1)/(x^2 + x - 3)

>>ezplot(f)


>>f1 = diff(f)
f1 =
(6*x + 6)/(x^2 + x - 3) - ((2*x + 1)*(3*x^2 + 6*x - 1))/(x^2 + x - 3)^2
>>f1 = simplify(f1)
f1 =
 -(3*x^2 + 16*x + 17)/(x^2 + x - 3)^2
>>pretty(f1)

       2
    3 x  + 16 x + 17
  - ----------------
       2          2
     (x  + x - 3)
>>crit_pts = solve(f1)
```

MATLAB- Otimização

- Para entendermos melhor como funciona o toolbox de otimização do MATLAB, precisamos entender a natureza dos problemas que temos.
- Os problemas de otimização podem ser
  - Programação Linear
  - Programação Quadrática
  - Programação Não-Linear

MATLAB- Programação Linear

- Quando a função objetivo é linear nos argumentos da otimização E as restrições que o problema pode ter também são lineares nos argumentos, esse é um problema de Programação Linear.
- Vou fazer tudo em termos de minimização.

$$\min_{\mathbf{x}} \mathbf{c}^{\mathbf{T}}\mathbf{x}$$

$$t.q$$

$$\mathbf{A}\mathbf{x} = a$$

$$\mathbf{C}\mathbf{x} \leq b$$

$$lb \leq \mathbf{x} \leq ub$$

MATLAB- Programação Quadrática

- Quando a função objetivo é quadrática nos argumentos da otimização E as restrições que o problema pode ter são lineares nos argumentos, esse é um problema de Programação Quadrática.
- Vou fazer tudo em termos de minimização.
- Problemas de carteira (portifólio) são exemplos deste tipo de problema

$$\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^{\mathbf{T}}\mathbf{Q}\mathbf{x} + \mathbf{q}^{\mathbf{T}}\mathbf{x}$$

$$t.q$$

$$\mathbf{A}\mathbf{x} = a$$

$$\mathbf{C}\mathbf{x} \leq b$$

$$lb \leq \mathbf{x} \leq ub$$

MATLAB- Programação Não-Linear

- Quando a função objetivo é nãao linear nos argumentos da otimização E as restrições que o problema pode ter não necessariamente são lineares nos argumentos, esse é um problema de Programação Não Linear.
- Vou fazer tudo em termos de minimização.
- Os problemas de otimização que enfrentamos em econometria são exemplos desses

$$\begin{aligned} &\min_{\mathbf{x}} f(\mathbf{x}) \\ &\quad t.q \\ &g_i(\mathbf{x}) = a_i \quad i \in 1, 2, \cdots, n \\ &g_j(\mathbf{x}) \leq b_j \quad j \in 1, 2, \cdots, N \\ &lb_k \leq \mathbf{x} \leq ub_k \quad k \in 1, 2, \cdots, M \end{aligned}$$

MATLAB- Funções de Otimização

- Linear and Quadratic Minimization problems.
  - linprog - Linear programming.
  - quadprog - Quadratic programming.
- Nonlinear zero finding (equation solving).
  - fzero - Scalar nonlinear zero finding.
  - fsolve - Nonlinear system of equations solve (function solve).
- Linear least squares (of matrix problems).
  - lsqlin - Linear least squares with linear constraints.
  - lsqnonneg - Linear least squares with nonnegativity constraints.

MATLAB- Funções de Otimização (II)

- Nonlinear minimization of functions.
  - fminbnd - Scalar bounded nonlinear function minimization.
  - fmincon - Multidimensional constrained nonlinear minimization.
  - fminsearch - Multidimensional unconstrained nonlinear minimization, by Nelder-Mead direct search method.
  - fminunc - Multidimensional unconstrained nonlinear minimization.
  - fseminf - Multidimensional constrained minimization, semi-infinite constraints.
- Nonlinear least squares (of functions).
  - lsqcurvefit - Nonlinear curvefitting via least squares (with bounds).
  - lsqnonlin - Nonlinear least squares with upper and lower bounds.
- Nonlinear minimization of multi-objective functions.
  - fgoalattain - Multidimensional goal attainment optimization
  - fminimax - Multidimensional minimax optimization.

MATLAB- Funções de Otimização (III)

- A estrutura geral dos comandos do MATLAB é a mesma, descrita abaixo. Para isso, é necessário construir uma função – denotada `fun`, que pode ter outros argumentos que não os da otimização (no nosso caso, a matriz de dados)
- Além disso, precisamos estipular um ponto inicial da otimização - `x0`.
- Caso tenhamos restrições não lineares, precisamos passar estas funções também.
- Caso a função dependa de mais de um argumento e você vai otimizar só um deles, você pode fazer `@(b) fun(b,x)`.

```
x = fminunc(fun,x0)
x = fminunc(fun,x0,options)
x = fminunc(problem)
[x,fval] = fminunc(...)
[x,fval,exitflag] = fminunc(...)
[x,fval,exitflag,output] = fminunc(...)
[x,fval,exitflag,output,grad] = fminunc(...)
[x,fval,exitflag,output,grad,hessian] = fminunc(...)
```