方程式求根

Notebook | MATLAB

https://youtu.be/O6ilfxV7rXo?si=2tmeVqfab7aKanLV

Symbolic Root Finding Approach 符号求根方法

- 在符号上进行数学运算，而不是数字
- 使用“符号变量”来执行符号数学运算
- 使用sym或syms来创建**符号变量**

```
syms x
x + x + x
(x + x + x)/4



x=sym('x');
```

```
x + x + x
(x + x + x)/4
```

👉 求解函数的根时，需要先使用syms或sym将设置为符号变量。

Symbolic Root Finding——solve()

```
syms x;
y=x*sin(x)-x;
%y=x*sin(x)-x=0  求解根x
solve(y,x);
%y  equation; x  symbol
```

Solving Multiple Equations

Solving Multiple Equations

- Solve this equation using symbolic approach:

$$\begin{cases} x - 2y = 5 \\ x + y = 6 \end{cases}$$

```
syms x y
eq1 = x - 2*y - 5;
eq2 = x + y - 6;
A = solve(eq1,eq2,x,y)
```

Solving Equations Expressed in Symbols

```
syms a b x;
solve(a*x^2-b==0,a)
```

```
syms x a b;
```

```
solve('a*x^2-b');
```

- Find the matrix inverse using symbolic approach

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

```
% 定义符号变量
syms a b c d;
% 构建矩阵
M = [a b; c d];
% 计算逆矩阵
M_inv = inv(M);
% 显示结果
disp(M_inv)
```

符号化解微分：diff()

MATLAB中的微分函数为：
diff(f,x,n)

diff函数求函数f对变量x的n阶导数

```
syms x;
y=4*x^5;
yprime=diff(y)
```

符号化解积分：int()

Calculate the integral of a symbolic function:

$$z = \int y dx = \int x^2 e^x dx, \qquad z(0) = 0$$

```
syms x;
y=x^2*exp(x);
```

```
z=int(y);
%求出不定积分含有的一个常数项
z=z-subs(z,x,0)
```

MATLAB 中的 `subs()` 函数用于符号表达式中的**符号替换**。它允许你将符号变量替换为具体的数值或另一个符号表达式。

主要用法：

```
subs(expr, old, new)
```

- `expr`：你想要进行替换的符号表达式或符号矩阵。
- `old`：你想要替换的符号（可以是一个符号或符号数组）。
- `new`：要替换成的值（可以是数值、符号变量，甚至是另一个符号表达式）。

在 MATLAB 中，`int()` 函数用于对符号表达式进行**不定积分**或**定积分**计算。`int()` 是 MATLAB 符号工具箱中的一个函数，常用于解析计算积分（积分可以是符号积分或数值积分）。

用法及语法：

```
int(f)                  % 计算 f 对默认变量的不定积分
int(f, v)               % 计算 f 对变量 v 的不定积分
int(f, v, a, b)         % 计算 f 对变量 v 的定积分，积分上下限为 a 到 b
```

- `f`：要积分的符号表达式或函数。

> - `v`：积分变量（可选，如果不指定，MATLAB 会自动选择表达式中的符号变量）。
> - `a` 和 `b`：定积分的下限和上限（可选，用于计算定积分）。

Numeric Root Finding Approach 数值求根方法

Function Handles(@)

- A handle is a pointer to a function
- Can be used to pass functions to other functions

```
xy_polt(@sin,0:0.01:2*pi)
```

fsolve()

A numeric root solver.

For example, solve this equation:

$$fx = 1.2x + 0.3 + x \bullet sin(x)$$

```
f2 = @(x) (1.2*x+0.3+x*sin(x));
fsolve(f2,0)
%f2  A function handle
%0   Initial guess
```

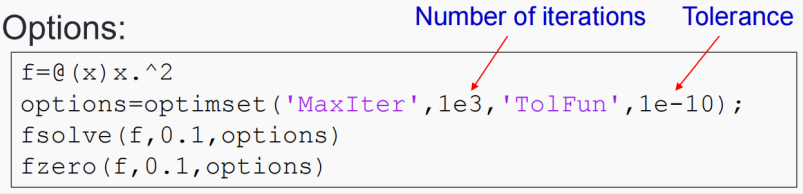
fzero()

```
f=@(X)X.^2;
fzero(f,0.1)
% =fsolve(f,0)
```

x = fzero(fun,x0) 尝试求出 fun(x) = 0 的点 x。求出fun在x0附近的零点，故x0可以是一个区间[xx1 xx2]。此解是 fun(x) 变号的位置 - fzero 无法求函数（例如 x^2）的根。

x = fzero(fun,x0,options) 使用 options 修改求解过程。

- Options:



x = fzero(problem) 对 problem 指定的求根问题求解。

Finding Roots Of Polynomials:roots()——多项式求根

```
roots([3 4 5 6 7])
%中括号内为多项式系数
```

Numeric Root Finding Methods

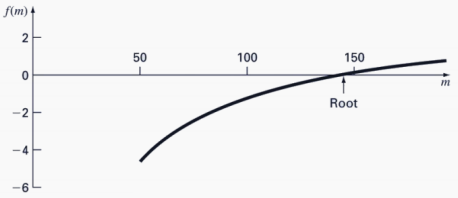
Bisection Method (Bracketing)——二分法

Start with an interval that contains the root

Bisection Method (Bracketing)

Assumptions:

- $f(x)$ continuous on $[l, u]$
- $f(l) \cdot f(u) < 0$



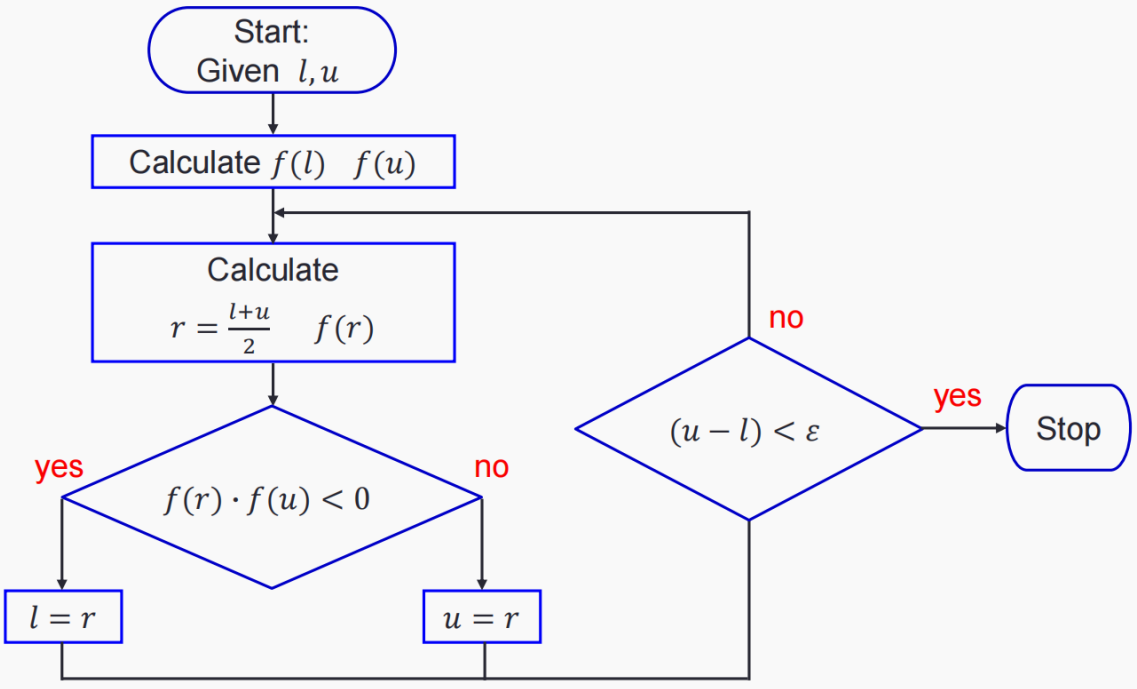
Algorithm:

Loop

1. $r = (l + u)/2$
2. If $f(r) \cdot f(u) < 0$ then new interval $[r, u]$
   If $f(l) \cdot f(r) < 0$ then new interval $[l, r]$

End

Bisection Algorithm Flowchart



Newton-Raphson Method (Open)——牛顿法

💡 Start with one or more initial guess points

Newton-Raphson Method (Open)

Assumption:

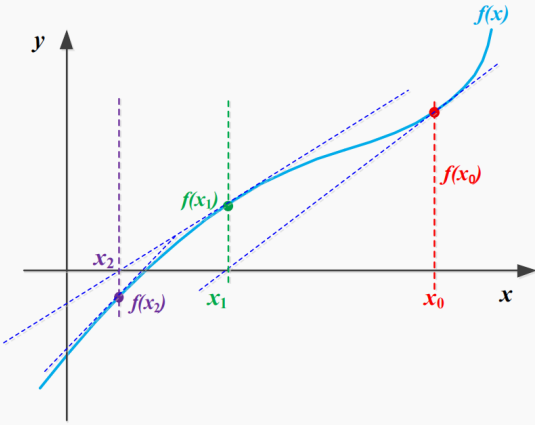
- $f(x)$ continuous
- $f'(x)$ known

Algorithm:

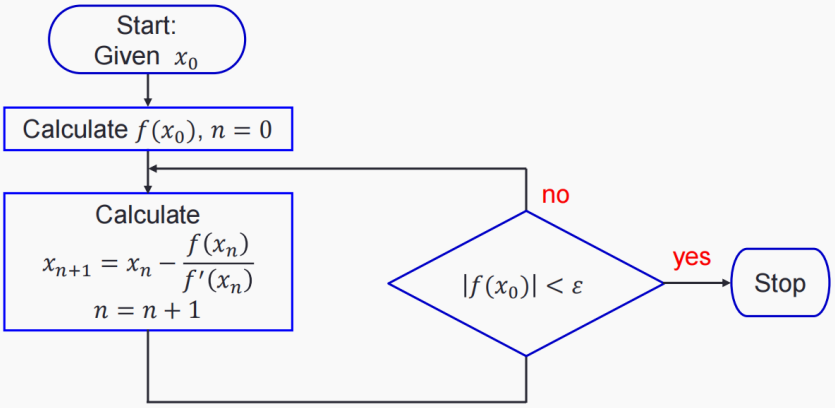
Loop

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

End



Newton-Raphson Algorithm Flowchart



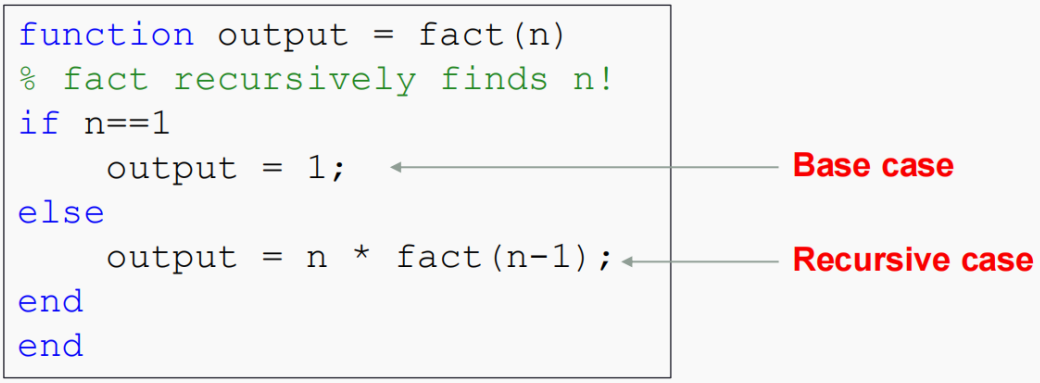
Recursive Functions 递归函数

阶乘递归函数

- 函数包含递归情况和基本情况
- 当达到基本情况时，函数停止

Factorial Recursive Function

- The function includes a recursive case and a base case
- The function stops when it reaches the base case



factorial——输入的阶乘全页折叠

```
f = factorial(n)
```

f = factorial(n) 返回所有小于或等于 n 的正整数的乘积，其中 n 为非负整数值。如果 n 为数组，则 f 包含 n 的每个值的阶乘。f 与 n 具有相同的数据类型和大小。

n 的阶乘通常使用感叹号字符以数学表示法写为 n!。

```
f = factorial(10)
% >>f = 3628800
```

Symbolic vs. Numeric

Symbolic vs. Numeric

	Advantages	Disadvantages
Symbolic	• Analytical solutions • Lets you intuit things about solution form	• Sometimes can't be solved • Can be overly complicated
Numeric	• Always get a solution • Can make solutions accurate • Easy to code	• Hard to extract a deeper understanding

📖 abs()——函数绝对值