牛顿迭代法

2017-12-24 | 2018-08-17 | Studies | 0 | 阅读次数： 308
5k | 0:04

> 伽罗瓦云：五次及以上多项式方程没有根式解。但面对生活中求解各种复杂方程的真实需求，是否束手无策？非也，今用牛顿迭代法高效求解方程的近似根。本文介绍了牛顿迭代法的推导和其计算机实现，并使用牛顿迭代法实现了开方等运算。

牛顿迭代法

定义

我们已知：微分是曲线的线性逼近。
对于二阶连续可导函数 $y=f(x)$，设 r 是 $f(x)=0$ 的根。对曲线上的任意一点 $A(x_0,y_0)$，作微分 $y'=f'(x)\mathrm{d}x$, 交 x 轴于一点 $(x_1,0)$，如下图所示，称 x_1 为 r 的一次近似值。

注意到x_1与函数的零点仍有一段距离，
过 x_1 作 x 轴垂线，
交 $y=f(x)$ 于$B(x_1,y_1)$，
继续作微分 $y'=f'(x_1)\mathrm{d}x$，
交 x 轴于 $(x_2,0)$，
称 x_2 为 r 的二次近似值。
如下图，此时 x_2 与 r 更加接近。

重复上述过程，经过50次迭代，得下图. 此时第50次近似值已经十分接近 r。

定义:
设二阶连续可导函数 $y=f(x)$，r 是 $f(x)=0$ 的根，选取 x_0 作为 r 的初始近似值，过点 $(x_0,f(x_0))$ 做曲线 $y=f(x)$ 的微分 L, 求出 L 与 x 轴交点的横坐标 $x_1=x_0-\frac{f(x0)}{f'(x_0)}$，称 x_1 为 r 的一次近似值。过点 $(x_1,f(x_1))$ 做曲线 $y=f(x)$ 的切线，并求该切线与x轴交点的横坐标 $x_2=x_1-\frac{f(x1)}{f'(x_1)}$，称 x_2 为 r 的二次

近似值。重复以上过程，得 r 的近似值序列，其中 $x_{n+1}=x_n-\frac{f(xn)}{f'(x_n)}$，称为 r 的 $n+1$ 次近似值，上式称为 **牛顿迭代序列**。

收敛条件

关于牛顿法的局部收敛性，我们有如下定理：

定理：设 x^* 是方程 $f(x)=0$ 的根，f 在某个包含 x^* 为内点的区间内足够光滑，且 $f'(x)\neq 0$。那么存在 x^* 的一个邻域 N(x^*)=[x^*-\delta,x^*+\delta]，使得对于任意 x_0 \in N(x^*)，牛顿法产生的迭代序列以不低于二阶的收敛速度收敛于解 x^*。

证明：
牛顿法是对应于函数 $\phi(x)=x-\frac{f(x)}{f'(x)}$ 的不动点迭代。我们有

$$\phi'(x)=\frac{f(x)f''(x)}{[f'(x)]^2}$$

若 $f'(x)\neq 0$，则有 \phi'(x^*)=0。因此，牛顿迭代法是局部收敛的。

和不动点收敛类似，对于牛顿法迭代，我们有

$$x_{k+1}-x^*=\phi(x_k)-\phi(x^*)=\phi(x^*)+\phi'(x^*)(x_k-x^*)+\frac{\phi''(\xi_k)}{2}(x_k-x^*)^2-\phi(x^*)$$

$$=\frac{\phi''(\xi_k)}{2}(x_k-x^*)^2$$

式中，ξ_k 位于 x_k 和 x^* 之间。因此

$$\lim_{k\to+\infty}\frac{|x_{k+1}-x^*|}{|x_k-x^*|^2}=\lim_{k\to+\infty}\frac{|\phi''(\xi_k)|}{2}=\frac{|\phi''(x^*)|}{2}$$

计算机实现

说明

首先我们的目标是近似求解函数的根。
由于牛顿迭代法迭代过程只限于一阶导数，
我们可以使用差分的方式来近似计算某点导数值：

$$f'(x)=\frac{f(x+eps)-f(x)}{eps}$$

其中的 eps 取决于我们对最终解精度的要求。

进而，我们由上一节中牛顿迭代公式有：

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

注意到 $\lim_{n\to+\infty} f(x_n) = 0$ ，
迭代最终收敛，则当我们迭代到 $|f(x_n)| < eps$ 时就可以退出迭代，
此时的 x_n 即为近似解。

注：在由于计算机内部对实数的存储是有误差的，所以对于 ϵ 的选取应格外小心。

Python 实现

```
def diff(f,x):
    return (f(x+eps)-f(x))/eps
def getZero(f,x0):
    n=0
    s=x0
    while True:
        s=s-f(s)/diff(f,s)
        n=n+1
        if math.fabs(f(s))<eps:
            return (s,n)
```

- $getZero(f, x_0)$ 函数接受一个函数 f ，和迭代起点 x_0
- ϵ 即 eps 为解的精度
- n 为迭代次数
- s 为最终的解值

使用计算机求平方根

我们可以使用上面的代码求解平方根。
首先求解平方根即求函数 $f(x) = x^2 - a$ 的零点。

调用以上函数即

```
f=lambda x: x**2-a
x, n=getZero(f,x0)
print(x,n,f(x))
```

我们取 $\epsilon = 10^{-15}, a = 2, x_0 = 10$ 时，程序输出

```
1   1.4142135623730951 18 4.440892098500626e-16
```

说明我们得到的近似解为 $\sqrt{2} \approx 1.4142135623730951$，迭代了 18 次，
此处的 $f(x_n) = 4.440892098500626 * 10^{-16}$，可见 $|f(x_n)| < 10^{-15} = \epsilon$，满足精度要求。

接下来我们使用系统自带的 求平方根函数 求值，其结果为

```
1   1.4142135623730951
```

由此可见我们使用牛顿迭代法求得的近似解已与系统提供的求根函数精度相等，能满足一般的需求。

较复杂函数的零点近似求解

接下来以一个较复杂函数

$$f(x) = \cos((2 - \sin(x))^{\arctan(x)})$$

为例演示函数零点的求解过程。

此函数在 $x = 0$ 附近的图像如下：

为求出函数在 $x = 2$ 附近的零点，
我们取 $\epsilon = 10^{-10}, x_0 = 2$，程序输出

```
1   2.567793875101797 5 -1.1263042511219229e-14
```

则经过 5 次迭代，我们得到近似解 2.567793875101797。

进一步的，我们可以通过其具体迭代过程发现，迭代的步长缩小得很快，这符合之前的预期：

$$x_0 = 2$$

$$x_1 = 3.048910600444615$$

$$x_2 = 2.5508408648839507$$

$$x_3 = 2.5679378839996416$$

$$x_4 = 2.567793884737295$$

$$x_5 = 2.567793875101797$$

总结

使用牛顿迭代法，我们可以快速近似计算满足收敛条件的函数的零点。

这种方法解决了很多非线性方程组的求根问题，简化了许多工程问题的计算。

> 参考资料：同济大学计算数学教研室《现代数值计算》（人民邮电出版社）

CS # Math

‹ 线性筛法及扩展

iExpr 反馈收集 ›