非线性方程组求根实验题

吴佳龙 2018013418

摘要

结合理论分析和编程计算，运用不同算法求解非线性方程，并比较他们的优劣。运用的算法分别为不动点迭代法、Steffensen 迭代法和 Newton 迭代法。

1 问题

求解非线性方程组

$$x^3 + 2x^2 + 10x - 20 = 0 \tag{1}$$

在 $x_0 = 1$ 附近的根。准确解为 $x^* = 1.368808107\cdots$，希望精度达到 10^{-7}

2 不动点迭代法

2.1 算法原理

将方程 $f(x) = 0$ 改写为 $x = \varphi(x)$ 。设定初始值 x_0，迭代

$$x_{k+1} = \varphi(x_k), k = 1, 2, \cdots$$

2.2 收敛性分析

设 x^* 为函数 φ 的不动点，φ' 在 x^* 的某个邻域 S 上存在且连续，且 $|\varphi'(x^*)| < 1$，则不动点迭代法局部收敛。

若存在常数 $L \in (0, 1)$，在区域内满足

$$|\varphi(x) - \varphi(y)| \leq L|x - y|, \forall x, y \in S$$

，则不动点迭代法产生的序列满足

$$|x^* - x_k| \leq \frac{L}{1 - L}|x_k - x_{k-1}|$$

2.3 算法实现

不动点迭代法的 MATLAB 实现如下：

```
function [x, x_series] = myFixedPoint(x0,
    phi, eps, kmax)
% 不动点迭代法 解 x=phi(x)
% kmax: 最大迭代次数
% eps: 相邻两次迭代结果小于eps，则终止

x = x0;
x_series = [x0];
for k = 1:kmax
    last_x = x;
    x = phi(last_x);
    x_series = [x_series x];
    if abs(x−last_x) < eps
        break
    end
end
end
```

2.4 计算结果

对于方程 1，分别选择迭代函数：

$$x_{k+1} = \varphi_1(x_k) = \frac{20 - 2x_k^2 - x_k^3}{10} \tag{2}$$

$$x_{k+1} = \varphi_2(x_k) = \sqrt[3]{20 - 10x_k - 2x_k^2} \tag{3}$$

调用函数

```
myFixedPoint(1, @(x)(20−2*x^2−x^3)/10.0,
    1e−8, 50)
```

和

```
myFixedPoint(1, @(x)nthroot(20−10*x−2*x
    ^2,3), 1e−8, 50)
```

计算。迭代产生的序列如表 1 所示。

可以看到 φ_1 和 φ_2 产生的迭代序列都不收敛，这是由于 $|\varphi_1'(x^*)|$ 和 $|\varphi_2'(x^*)|$ 的绝对值都大于 1，不满足局部收敛的条件。

Table 1: 不动点迭代法

	φ_1 2	φ_2 3
x_0	1.0	1.0
x_1	1.7	2.0
x_2	0.9307	-2.0
x_3	1.74614203626	3.17480210394
x_4	0.857796793737	-3.17171550601
x_5	1.78971892824	3.1614381025
x_6	0.78611746376	-3.16164373894
x_7	1.82782332719	3.16233360997
x_8	0.721147914713	-3.16231990025
x_9	1.85848552855	3.16227393009
x_{10}	0.66729126817	-3.16227484407
x_{11}	1.8812314848	3.16227790884
x_{12}	0.626419796624	-3.16227784791
x_{13}	1.89693882451	3.16227764359
x_{14}	0.597734533785	-3.16227764765
x_{15}	1.90718643312	3.16227766127
x_{16}	0.578815600138	-3.162277661
x_{17}	1.91360258592	3.16227766009

3 Steffensen 迭代法

3.1 算法原理

将 Aitken 加速方法和不动点迭代法结合起来就得到了 Steffensen 迭代法。

$$\begin{cases} y_k = \varphi(x_k) \\ z_k = \varphi(y_k) \\ x_{k+1} = x_k - \frac{(y_k - x_k)^2}{z_k - 2y_k + x_k} \end{cases}$$

Steffensen 迭代法可以看做将 $(x_k, \varepsilon(x_k) = y_k - x_k)$ 和 $(y_k, \varepsilon(y_k) = z_k - y_k)$ 连结延长到与 x 轴的交点作为新的近似值 x_{k+1}。

3.2 收敛性分析

根据课本上的定理 2.3:

Theorem 1. 设函数 φ 是不动点迭代法的迭代函数，x^* 是 φ 的不动点，在 x^* 邻域 φ 有 $p+1$ 阶导数存在且连续，对 $p = 1$，若 $\varphi'(x^*) \neq 1$，则 *Steffensen* 方法是二阶的。若不动点迭代法是 $p(> 1)$ 阶收敛的，则 *Steffensen* 方法是 $2p - 1$ 阶收敛的。

这说明，Steffensen 方法可以把不收敛的不动点迭代法改进为二阶收敛的方法。

3.3 算法实现

Steffensen 加速方法的 MATLAB 实现如下:

```
function [x, x_series] = mySteffensen(x0,
    phi, eps, kmax)
% Steffensen加速方法 解 x=phi(x)
% kmax: 最大迭代次数
% eps: 相邻两次迭代结果小于eps，则终止

x = x0;
x_series = [x0];
for k = 1:kmax
    last_x = x;
    y = phi(x);
    z = phi(y);
    x = x- (y-x)^2/(z-2*y+x);
    x_series = [x_series x];
    if abs(x-last_x) < eps
        break
    end
end
end
```

3.4 计算结果

对于 φ_1 和 φ_2 分别调用

```
mySteffensen(1, @(x)(20-2*x^2-x^3)/10.0, 1
    e-8, 50)
```

和

```
mySteffensen(1, @(x)nthroot(20-10*x-2*x
    ^2,3), 1e-8, 50)
```

计算。产生的迭代序列如表 2 所示。

Table 2: Steffensen 迭代法

	φ_1 2 + Steffensen	φ_2 3 + Steffensen
x_0	1.0	1.0
x_1	1.2	1.33349213911
x_2	1.27374039955	1.36841543911
x_3	1.30830003901	1.36880805831
x_4	1.34727521974	1.36880810782
x_5	1.36672391381	1.36880810782
x_6	1.36878928488	
x_7	1.36880810629	
x_8	1.36880810782	

可以看到，Steffensen 方法将原来不收敛的不动点迭代法改进为收敛的算法，且是二阶收敛。

4 Newton 迭代法

4.1 算法原理

为求解方程 $f(x)=0$，等价于求函数 $f(x)$ 的零点，Newton 法对曲线上的点 $(x_k, f(x_k))$ 作切线，切线与 x 轴的焦点作为新的近似值。

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

4.2 收敛性分析

有课本上定理 4.1 ：

Theorem 2. 设 $f(x^*)=0, f'(x^*)\neq 0$，且 f 在包含 x^* 的一个区间上有二阶连续导数，则 *Newton* 迭代法局部收敛到 x^*，且至少二阶收敛，并有

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

4.3 算法实现

Newton 法的 MATLAB 实现如下：

```
function [x, x_series] = myNewton(x0, f, df,
    eps, kmax)
% Newton法 解 f(x)=0
% df: f的导函数
% kmax: 最大迭代次数
% eps: 相邻两次迭代结果小于eps，则终止
x = x0;
x_series = [x0];
for k = 1:kmax
    last_x = x;
    x = x-f(x)/df(x);
    x_series = [x_series x];
    if abs(x-last_x) < eps
        break
    end
end
end
```

4.4 计算结果

调用

```
f = @(x)x^3+2*x^2+10*x-20;
df = @(x)3*x^2+4*x+10;
myNewton(1, f, df, 1e-8, 50)
```

产生的迭代序列如表 3 所示。

Table 3: Newton 迭代法

	Newton
x_0	1.0
x_1	1.41176470588
x_2	1.36933647059
x_3	1.36880818862
x_4	1.36880810782
x_5	1.36880810782

可以看到 Newton 快速地收敛到解。

5 方法比较

将不同算法迭代产生的序列作图 1。

可以看到未改进的不动点迭代法产生的迭代序列产生剧烈的振动，结果不收敛。而 Steffensen 迭代法和 Newton 迭代法都是二阶收敛算法，都能在很少的迭代次数时收敛到较精确的解。不同的 φ 选取也会对收敛速度产生影响。

6 总结

本次实验分别采用不动点迭代法、Steffensen 迭代法和 Newton 迭代法求解非线性方程 1，并在使用不动点迭代法和 Steffensen 法时分别选取了两种不同的迭代函数 2 3。即通过理论分析迭代法的收敛性，又进行编程计算，观察算法产生的迭代序列，得到了与理论相符合的结果。

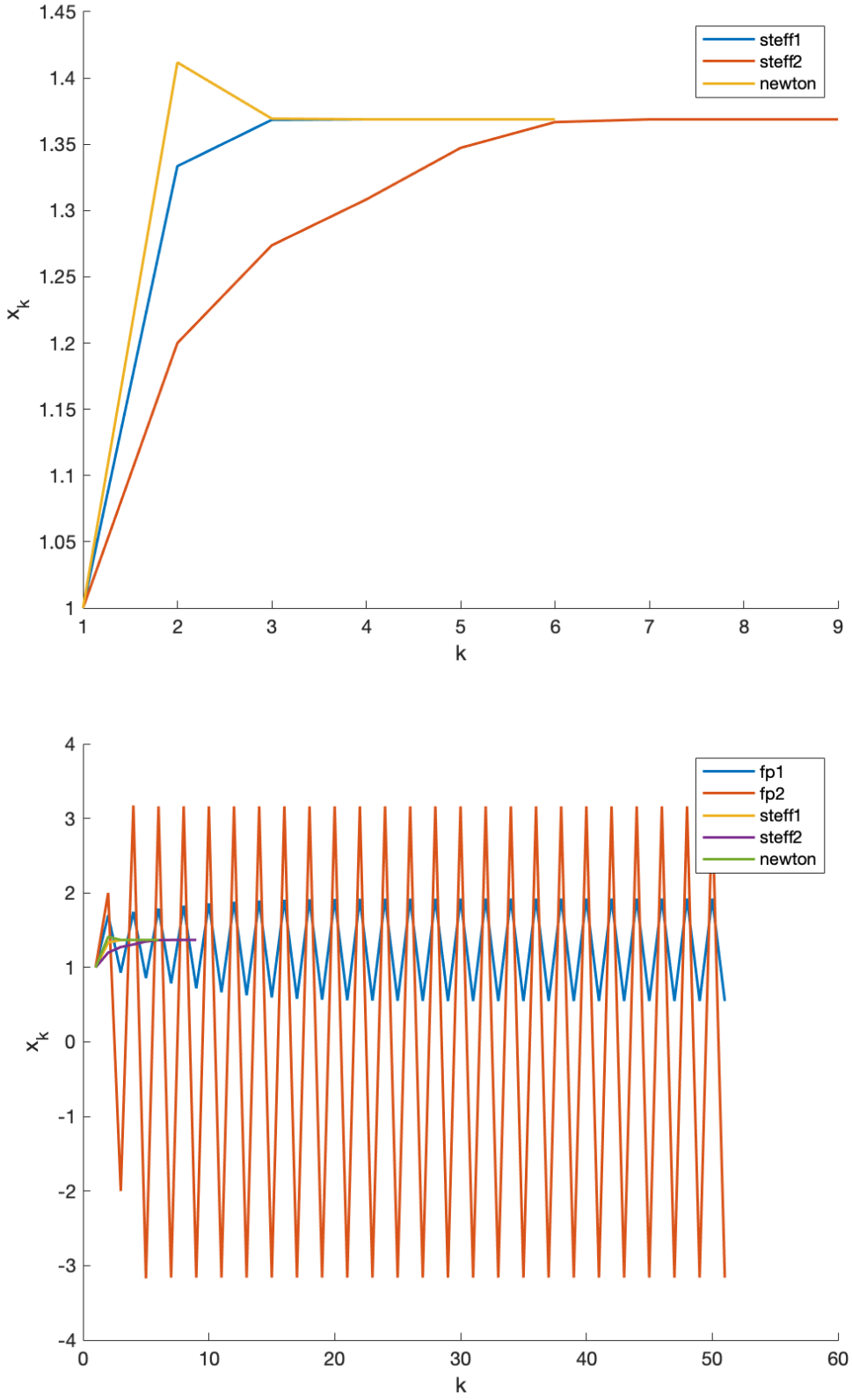

Figure 1: 不同算法的迭代序列