非线性方程组的迭代算法

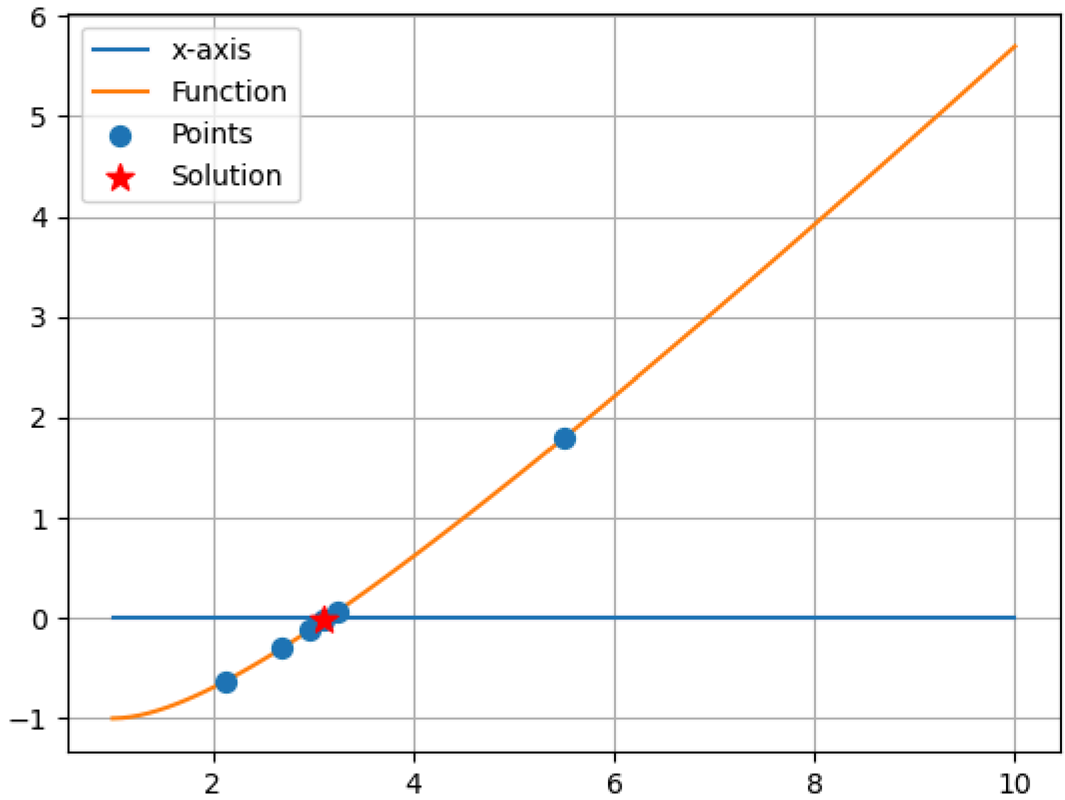
本章主要学习如何通过数值方法求解下述非线性方程组:

$$f(x) = (x-s)^m \phi(x) = 0$$

这里，s称为方程的m重根，对应的x为m重零点。

对分法

对分法的求解概念简单，若函数$f(x) \in [a,b]$上连续且发生变号，即$f(a)f(b) < 0$，那么就证明函数在区间(a,b)间必然有一个根，那么我们可以通过不断求解区间的中点的方法，对比根在哪个子区间内完成求解。这一简单的方法示意如下，求解代码写在`Bisection.py`中。



需要注意的是，对分法在智能求解单根或者奇数重根，不能求解偶数重根（$y=x^2$仅碰到了$y=0$）和复数根。

简单迭代法

简单迭代法将原先的非线性方程表达为:

$$x_{k+1} = \phi(x_k)$$

这样可以通过迭代求解；然而不同的非线性方程可以有不同的迭代公式，其最终是否能够收敛取决于$\phi(x)$本身。

收敛性判断

对于初始值$x_0 \in [a, b]$，若简单迭代法收敛，那么必须满足以下两个条件:

- 自映射：当 $x \in [a, b], \phi(x) \in [a, b]$
- 压缩性：当 $x \in (a, b), |\phi(x)'| \le L < 1$，$L$为小于一的常数

其中第一个条件为收敛性提供了一个范围约束，第二个条件可以通过微分中值定理这样理解:

$$|x_{k+1} - s| = |\phi(x_k) - \phi(s)| = |\phi'(\epsilon)| \cdot |x_k - s| \le L \cdot |x_k - s|$$

相当于每次迭代的误差都会转变为原先误差的L倍，若保证误差逐渐变小，必须保证$L < 1$

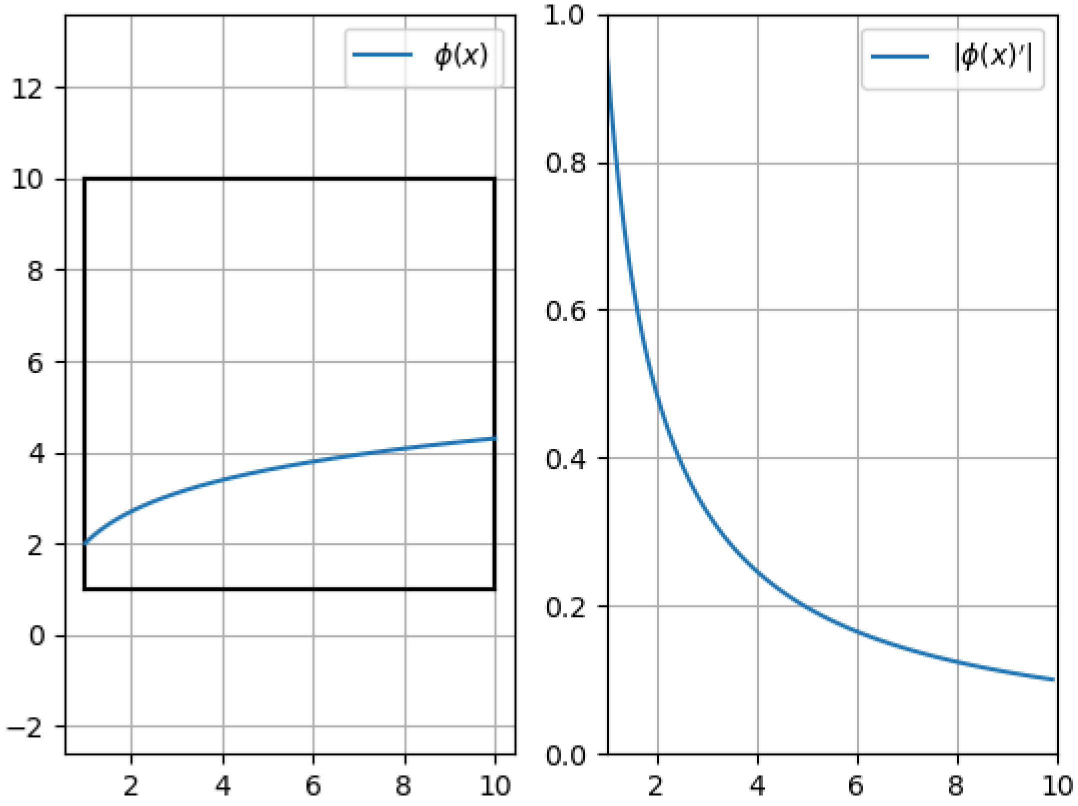
满足上述条件可以保证求解的序列收敛，并可以成立误差估计式:

- $|s - x_k| \le \frac{L^k}{1-L}|x_1 - x_0|$
- $|s - x_k| \le \frac{L}{1-L}|x_k - x_{k-1}|$

在实际求解时，采用以下标准:

$$\frac{|x_k - x_{k-1}|}{|x_k|} \le \eta$$

上述简单迭代法的代码事件写在`Simple_Iter.py`中，其中也同步绘制了其两个收敛性的判断，其中第一个条件相当于$\phi(x)$必须包含在a,b构成的矩形区域内:



收敛速度

定义$e_k = s - x_k$,那么我们可以定义收敛因子c以及r阶收敛速度:

$$\lim_{k \to \infty} \frac{|e_{k+1}|}{|e_k|^r} = c$$

上述极限也可求解到某阶$k \geq K$时进行判断

- $r = 1$，线性收敛，$0 < c \leq 1$
- $r = 2$，平方收敛
- $r > 1$，统称为超线性收敛

在先前的收敛性判断中，简单迭代法是线性收敛的，若能够满足以下条件:

- $\phi^{(i)}(s) = 0, (i = 1, 2, \cdots, m - 1)$
- $\phi^{(m)}(s) \neq 0$

那么简单迭代法序列m阶收敛于s。

Steffensen迭代法

Steffensen迭代法在简单迭代法的基础上做了些小改动，使其可以加速收敛:

由x_k,x_{k+1},x_{k+2}三个迭代值，可通过微分中值定理得到:

$$x_{k+1}-s=\phi'(\epsilon_k)(x_k-s)$$
$$x_{k+2}-s=\phi'(\epsilon_{k+1})(x_{k+1}-s)$$

对上述式子进行变型:

$$\frac{x_{k+1}-s}{x_k-s}\approx\frac{x_{k+2}-s}{x_{k+1}-s}\rightarrow s\approx x_k-\frac{(x_{k+1}-x_k)^2}{x_{k+2}-2x_{k+1}+x_k}$$

这样我们就得到了Steffensen迭代法:

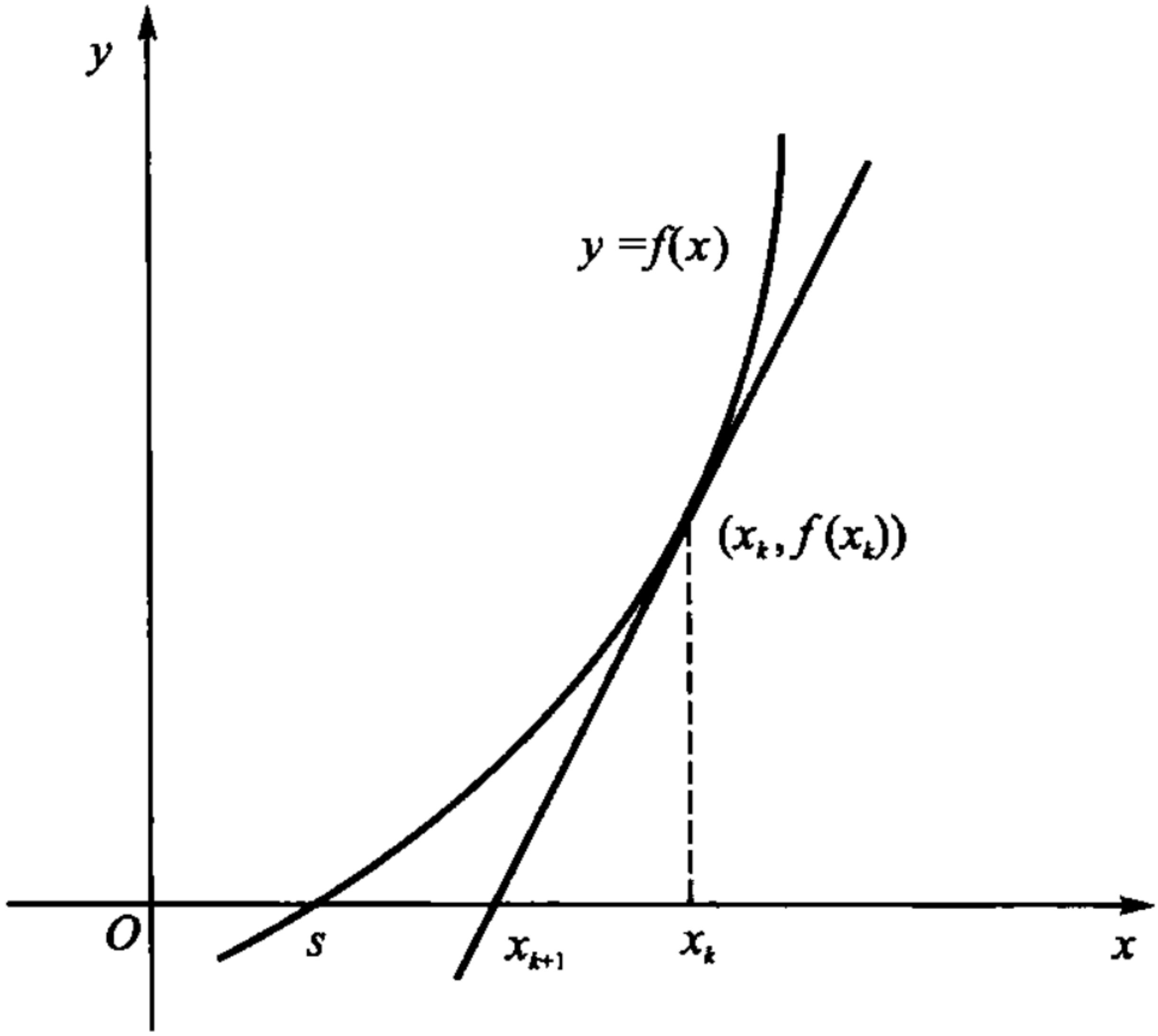
$$x_{k+1}=x_k-\frac{(\phi(x_k)-x_k)^2}{\phi(\phi(x_k))-2\phi(x_k)+x_k}$$

Steffensen迭代法二阶收敛于s。

上述的Steffensen和简单迭代法，都可以用于求解方程的**实数和复数根**。

牛顿法

牛顿法(Newton's Method)可求解非线性方程组的**实数和复数根**，也可以用于求解有m**重根**的情况。在求解实数根时，由于牛顿法具有强几何意义，其也称之为**切线法**。



牛顿法的基本迭代式如下：

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

收敛性判断

存在小范围和大范围两个收敛定理。

对于**小范围收敛定理**：

- $f''(x)$连续且$f'(x) \neq 0$，那么x_0在靠近s的某个范围内收敛于s。
- $f''(x) \neq 0$，且$x_0 \neq s$，则序列时平方收敛的。

对于**大范围收敛定理**:

- $f(x)$在区间(a,b)内变号，即$f(a)f(b)<0$
- 在区间内$f''(x)$不变号
- 在区间内$f'(x)\neq 0$
- $f(x_0)f''(x_0)>0$
- 满足上述四个条件，那么牛顿法产生的序列**单调平方收敛**于s。

求解m重根的牛顿法

对于例如$f(x)=(x-2)^2$的函数，其有多重根，可以发现:

$$f(s)=f'(s)=\cdots=f^{(m-1)}(s)=0$$

通过求解极限证明发现，在这种情况下，牛顿法虽然收敛，但是收敛速度只是线性的。若已知方程的重根数m，使用以下迭代求解公式，收敛速度可以变为二阶:

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

一般情况下不知道重根数m，我们可以考虑求解方程$u(x)=0$，此时方程的解具有单重根:

$$u(x)=\frac{f(x)}{f'(x)}$$

此时求解上述方程的牛顿迭代法最终可以整理为下述形式，并具有二阶收敛速度:

$$x_{k+1}=x_k-\frac{f(x_k)f'(x_k)}{[f'(x_k)]^2-f(x_k)f''(x_k)}$$

上述的经典和求重根的牛顿法代码整理在`Newton.py`中。

割线法

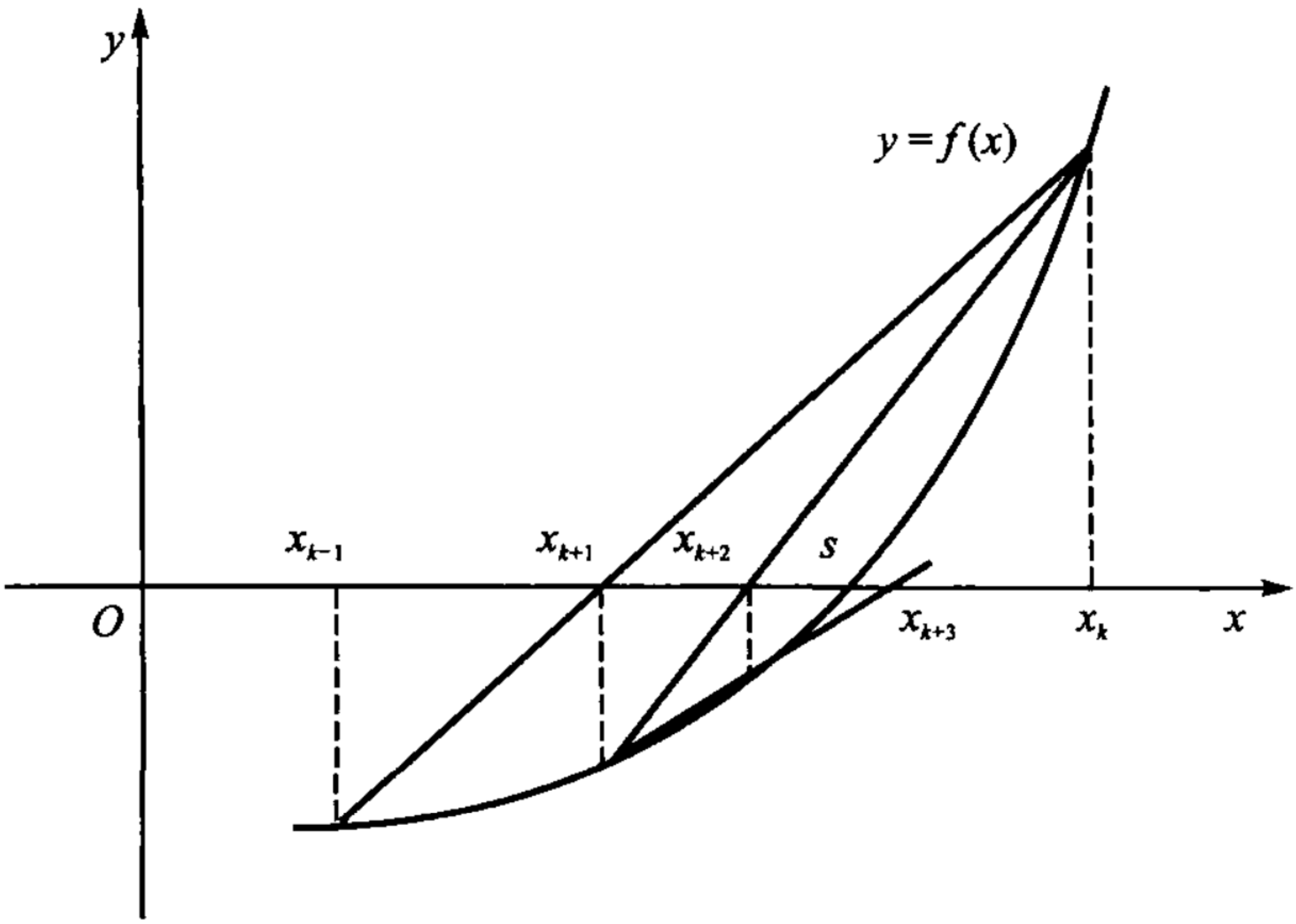
在牛顿法中，$f(x)$导数的计算复杂，在不知道显性表达式的情况下求解麻烦，为此使用差分形式:

$$f'(x)\approx\frac{f(x_k)-f(x_{k-1})}{x_k-x_{k-1}}$$

这样产生的迭代表达式为:

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

同样由于求解实根的几何意义，其称为割线法。



若$f''(x)$连续且$f'(x)\neq 0$那么在s附近的割线法产生的序列至少有1.618阶收敛速度。

若我们固定初始点$(x_0, f(x_0))$，那么我们可以得到**单点割线法**:

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_0)}{f(x_k) - f(x_0)}$$

单点割线法也有同牛顿法类似的收敛定理，仅仅增加额外的一项$f(x_0)f(x_1) < 0$:

- $f(x)$在区间(a, b)内变号，即$f(a)f(b) < 0$
- 在区间内$f''(x)$不变号
- 在区间内$f'(x) \neq 0$
- $f(x_0)f''(x_0) > 0, f(x_0)f(x_1) < 0$
- 满足上述四个条件，那么单点割线法产生的序列**一阶收敛**于s。

这两个割线方法的代码实践写在了`Secant.py`文件中。