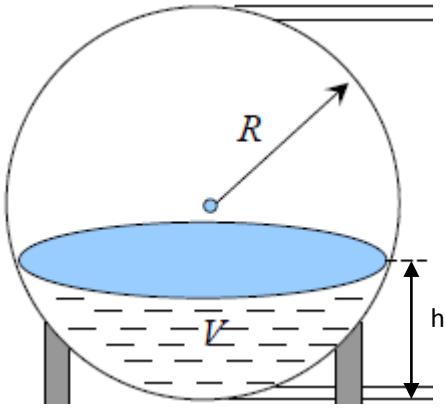
第二次作业

问题回顾 一个单位球体被平面切成两部分，其中一部分的体积是另一部分体积的 3 倍，确定从球中心到平面的距离。（精度自定，如精确到小数点后 10 位）



摘自课件 2 pp53

问题分析 该平面显然将球分为两个部分，一部分为“优球冠”，另一部分为“劣球冠”，根据两者之间 3 倍的关系，必然有：

$$V_{优球冠} = \frac{1}{4}V_{全球} \tag{1}$$

对于优半球球冠有如下公式计算其体积，h 表示球冠的高度：

$$V = \pi h^2 \frac{3R - h}{3}$$

球的体积计算公式有：

$$V = \frac{4}{3}\pi R^3$$

代入(1)式即可得到：

$$\frac{h^2(3R - h)}{R^3} = 1 \tag{2}$$

其中

$$h = R - d$$

其中 d即球心到平面的距离，即题中要求的量。将(2)式化简为标准的多项式形式：

$$f\left(\frac{h}{R}\right) = \left(\frac{h}{R}\right)^3 - 3\frac{h}{R} + 1$$

本问题转化为求一个非线性一元多项式求的根

$$f(x) = x^3 - 3x + 1$$

题中所求的 $d = (1 - x)R$ 。虽然 3 次方程具有**求根公式 [1]** ，但为了不失一般性地求解非线性方程，用以下各种方法进行求解。在讨论具体的求解方法之前，我们先确定计算f(x)的

方法，因为这在后面的方法中都会反复用到，为了降低计算次数，对于一个最高次项系数归一化的一元多项式可用下面的方法计算减少计算量[ii]。

$$f(x) = \left((\cdots\left((x + a_1)x + a_2\right)\cdots)x + a_{n-1}\right)x + a_n \tag{3}$$

$$f'(x) = \left(\left(\cdots\left((nx - (n-1)a_1)x + (n-2)a_2\right)x + \cdots\right)x + 2a_{n-2}\right)x + a_{n-1} \tag{4}$$

特别说明的是结果表明要得到本题的计算并不困难，因此文档中没有用表列出计算中每一次迭代的计算值，而只是画出了迭代过程的示意图和举出了最后的结果[1]，如有需要明细的迭代值，只需要运行给出程序便可以得到每一次迭代结果。

目录

二分法 3
试位法 4
混合方法和综合考虑 6
迭代法 7
牛顿法 8
牛顿法的讨论 9
牛顿法思考——“导数求取” 9
牛顿法思考——“避开极值点” 10
求多项式的全部实数根的方法概述 11
小结 11

[1] 最终结果在小结中也有给出

二分法

在确定方程感兴趣的根值区间后，找到一个两端点函数值异号的子区间，并考察该子区间的中点的函数值，反复利用根的存在性定理，循环减半根估计区间，递归求得确定精度下的根的估计值。由该问题的特殊性，可以得到一个估计区间，即$x = \frac{h}{R}\epsilon(0,1)$，通过球冠体积的单调性，能够肯定，该区间内只有一个根，因此对于此问题肯定可以求的一个有物理意义的根。

用(x_r, x_l)来表示我们所感兴趣的求根区间。初始化时为(0,1)，MATLAB 脚本程序如下，当小数点后 10 位的数值没有变化时，停止计算。

```
% This program is developed to demonstrate a Bolzano method on
% solving a algebra equation an iterval [x_l, x_r] should be given
% to locate the root as a feature of this method.
clear;
x_l = 0;  % the initial value of the left bound of x is 0
x_r = 1;  % the initial value of the right bound of x is 1
x_m = mean([x_l,x_r]); % x_m is the mean value of the two bounds
x_g = 0; % x_t is the target value of the Bolzano method
e_a = 5e-11; % the principal of 5 Significant Digits

if f(x_l)*f(x_r)>0
   error('invalid input');
else
while (abs(x_g-x_m)>e_a)    % tolerance of accuracy
if f(x_m)==0              % find the accurate root
   break;
else if f(x_l)*f(x_m)<0;
   x_r = x_m;            % update the new estimation to the interval
   else x_l = x_m;
   end
end
x_g = x_m;                % current estimation
x_m = mean([x_r,x_l]);
end
end
```

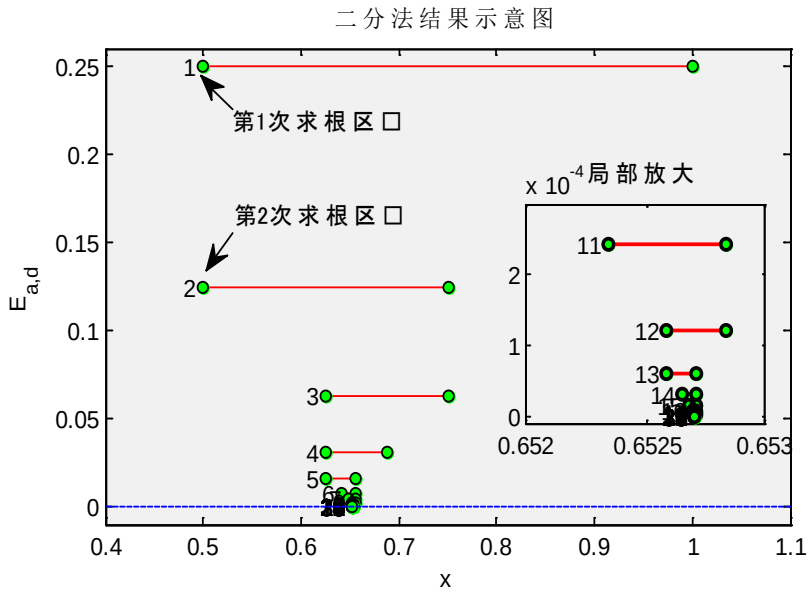
由于二分法的特殊性，我们可以准确地知道循环次数，即

$$n = \left\lceil log_2(\frac{x_r - x_l}{E_{a,d}}) \right\rceil$$

这样程序中的高亮的 while 循环可以换做如下，省去了判断循环标志时的计算量：

```
n=ceil(log(x_r-x_l)/e_a/log(2))
for i=1:n
```

本例中用了 34 次迭代达到了小数点后 10 位有效数字的计算准则，即$E_{a,d} = 5e - 11$，x=0.6527036447；即$\mathrm{d} = 0.3472963553 \times \mathrm{R}$。下面用图部分表示出迭代过程。



注：图中红色线段表示每一次迭代的求根区间，左边的数字表示迭代的顺序数，即第几次迭代

试位法

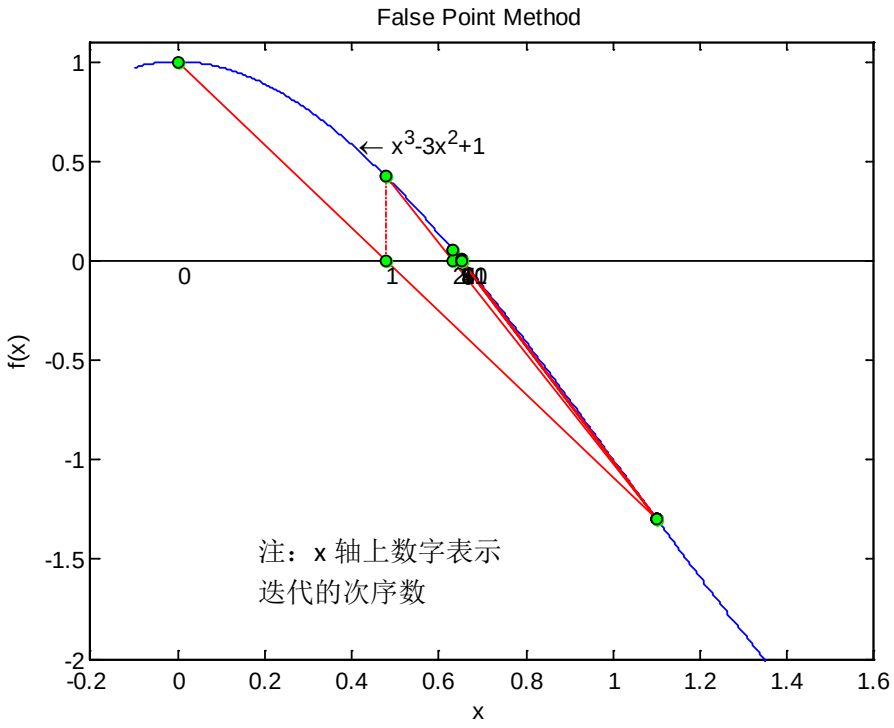
与二分法不同，每次区间的划分点不在中点，而是两区间函数值点连线与 x 轴的交点，递归地做这样的划分求解最终的结果，图示结果附后。

```
% this program is developed to demonstrate the False Position Method
% this version cannot deal with some bad condition in solving the root
clear;
x_l = 0;  % the initial value of the left bound of x is 0
x_r = 1.1;  % the initial value of the right bound of x is 1
x_c = x_r; % x_c is the cross point of the two bound-line and the x-axis
x_t = 0; % x_t is the target ROOT of the eqation
e_a = 5e-11; % the principal of 5 Significant Digits
i = 0; % iteration tag
while (abs(x_t-x_c)>e_a)
   x_t=x_c;        % update the target value with the cross point
if f(x_t)==0
    break;
else
    x_c = x_r - f(x_r)*(x_l-x_r)/(f(x_l)-f(x_r));     % get the cross point

    if f(x_c)*f(x_r)<0
        x_l = x_c;
    else x_r = x_c;
```

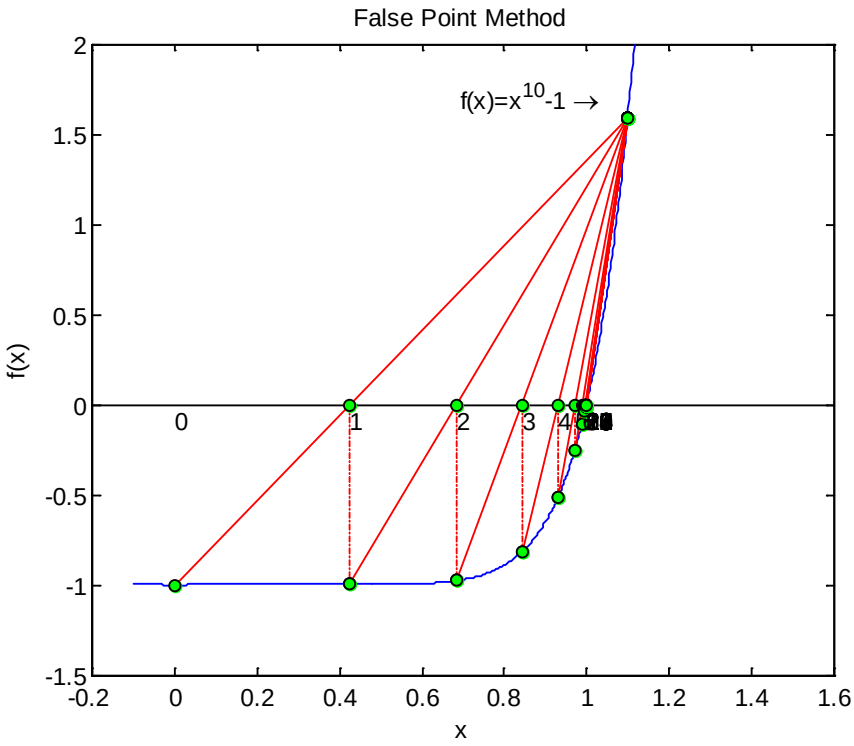
```
    end
end
i = i+1;
end
```

经过 12 次迭代得到 x=0.6527036447 与真实值比较小数点后有 10 位有效数字。迭代过程图示如下：



根据计算结果可以看到，此例中试位法较二分法更为快速地收敛到了想要的解，且并也已经出现“一个划界点保持不动”的现象（在区间右端点初值取值不当，如本例去x_r=1 时不会出现这种现象），但却没有出现“导致很差的收敛性”[iii]。

仔细分析可以发现，这是由于本例子中的多项式是三阶的，更附近的函数值变化不剧烈；即没有出现课本上例子“$f(x) = x^{10} - 1$”中函数值接近于“0”但是自变量却非常不接近于根的现象，如下迭代过程图



因此，可以采用一种简便地修改此缺陷的方法，即将程序结束的条件做一修改，修改前面程序中高亮的 while 语句，使得增加估计根的误差容限 abs(f(x_c))< epsilon。

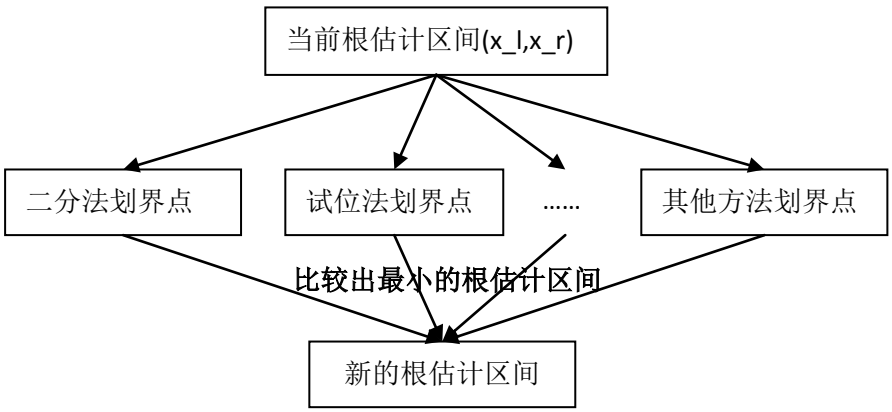
```
while (abs(x_t-x_c)>e_a && abs(f(x_c))< epsilon)
```

但可想而知，这种方法在恶劣的情况下会极大地增加计算量。

混合方法和综合考虑

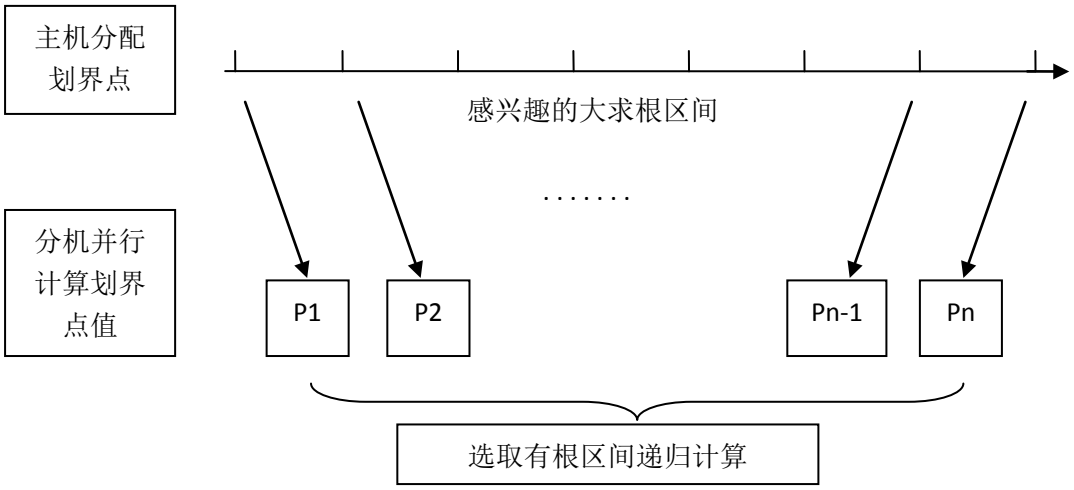
充分观察二分法与试位法的特点，可以发现试位法本希望改变二分法中根估计区间缩小过慢的情况，但是当出现一个划界点保持不同的情况时，根估计区间缩小反而不如二分法中每次缩小一半。事实作为“划界”法，既是寻找一种最优的划界策略，使得根估计区间迅速的缩小。

下面就说明一种结合多种划界法设计的混合划界法：



在综合计算划界点的前提下，这种方法可以保证根估计区间以最快速度递减。

这种并行利用多算法的思想还可以迁移出另外的思想，对于一个变量维数较高的，或者是计算较复杂的非线性函数，可以采用N分法做划界，然后分配给多个并行机计算。



上述方法也可以在单 CPU 下计算，但是却不能对收敛速度有任何改进，但是这种平凡的算法却可以很容易推广到多机并行计算当中去，相比而言，下面将要采用的“开方法”或者迭代法却没有如此容易。我想，在科学求根计算要求量越来越大以及云计算或者并行计算大力发展的趋势下，考虑算法并行性是很有意义的。

迭代法

迭代法用迭代方程$x_n = g(x_{n-1})$求解不动点。在应用到对于求方程$f(x) = 0$的根上，最朴素的迭代方程构造是：

$$g(x) = f(x) + x$$

这类方程的收敛性推导，运用中值定理

$$\begin{aligned} x - x^* &= g'(\theta)(x - x^*) \\ &= (f'(\theta) + 1)(x - x^*) \end{aligned}$$

当在整个求根区间上满足$|f'(\theta) + 1| < 1$时，迭代算法是有效的。为了加快收敛，同时令

$$\begin{cases} g(x) = \varphi \cdot f(x) + x \\ g'(x) = \varphi \cdot f'(\mathrm{x}) + 1 = 0 \end{cases}$$

这时可以最快速度收敛，可以发现，上面式子稍变形就可以得到：

$$\varphi = -\frac{1}{f'(x)}$$

即得到下面要使用的牛顿方法

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

由此可见这一类迭代方法的“收敛性”意义上的最优进化即是牛顿方法。对于其他构造迭代方程的方法这里就不再讨论。

牛顿法

迭代方程是：

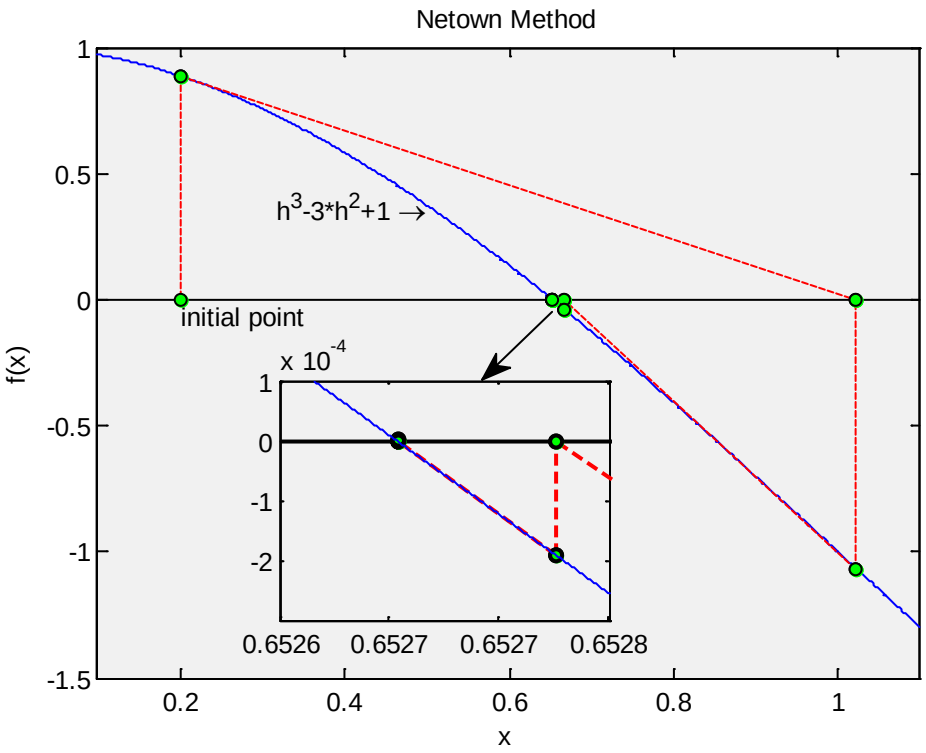
$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

同前面的过程，采用绝对预定误差$E_{a,d} = 5e-11$，使得结果小数点后有 10 位有效数字。先给出程序，再图示求解。

```
% Newtown method is developed to solve the root of nonlinear equation
% f and df stands for the function and the differential function given
% in the method as the Beginning of this assignment document
clear;
x1 = 1;
x2 = 0.2;
e = 5e-11;          % Absolute tolerance of error

while (abs(x2-x1)>e)
    x1 = x2;
    x2 = x1-f(x1)/df(x1);      % Newtown formula
end
```

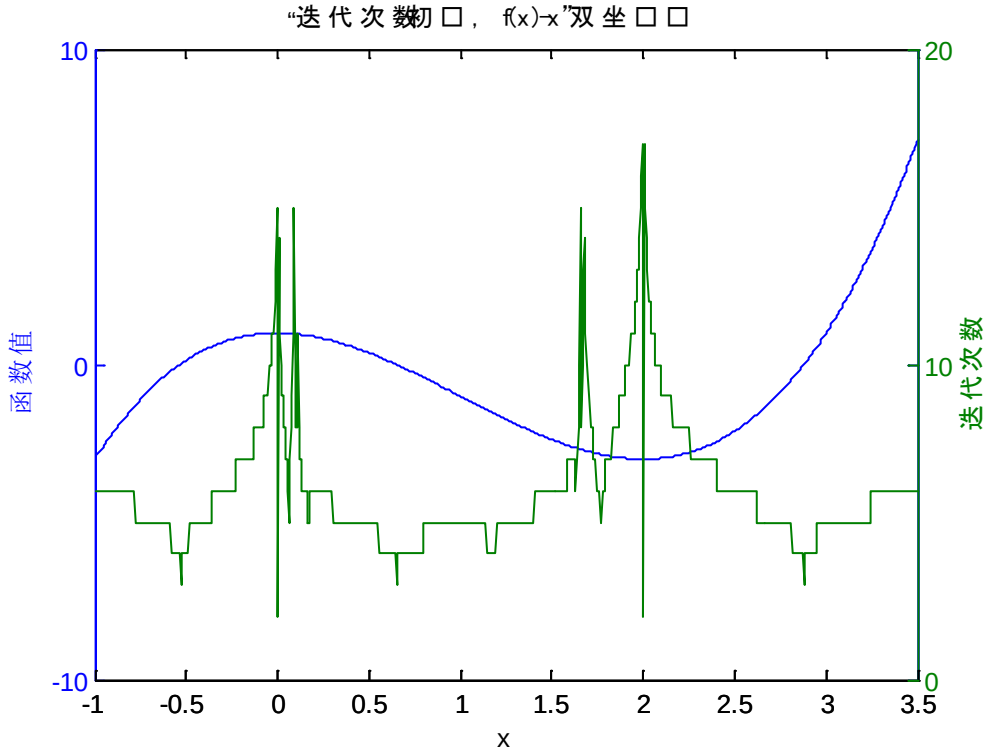
可以看到代码编写容易，最后结果 6 次迭代得到了结果 x=0.6527036447，满足小数点后 10 位有效数字的要求，迭代过程图示如下：



牛顿法的讨论

在正常情况下，牛顿法收敛速度很快，但是正如课本上列举，有几种情况会使得牛顿法陷入一些缺陷中。

下面讨论当改变迭代初值时，牛顿法迭代次数的变化：



明显看到，当迭代初值接近 f'(x)=0 时，会出现迭代次数增多，和收敛到不感兴趣的根的情况（这个图中没有标出）。由此看来，选取牛顿法的初值将是十分重要的，要尽量避免选取在极值点附近。关于怎样避开极值点，后面会有思考。

牛顿法思考——“导数求取”

当待求根方程为多项式方程时，根据本文第 1 页所给出的方法就可以立即求取。

当待求根方程为非多项式方程时，求取导数就显得比较困难，一种朴素的改进就是所谓的用差分近似微分：

$$f'(x) \approx \frac{f(x_i + \delta x_i) - f(x_i)}{\delta x_i}$$

$$x_{i+1} = x_i - \frac{\delta x_i f(x_i)}{f(x_i + \delta x_i) - f(x_i)}$$

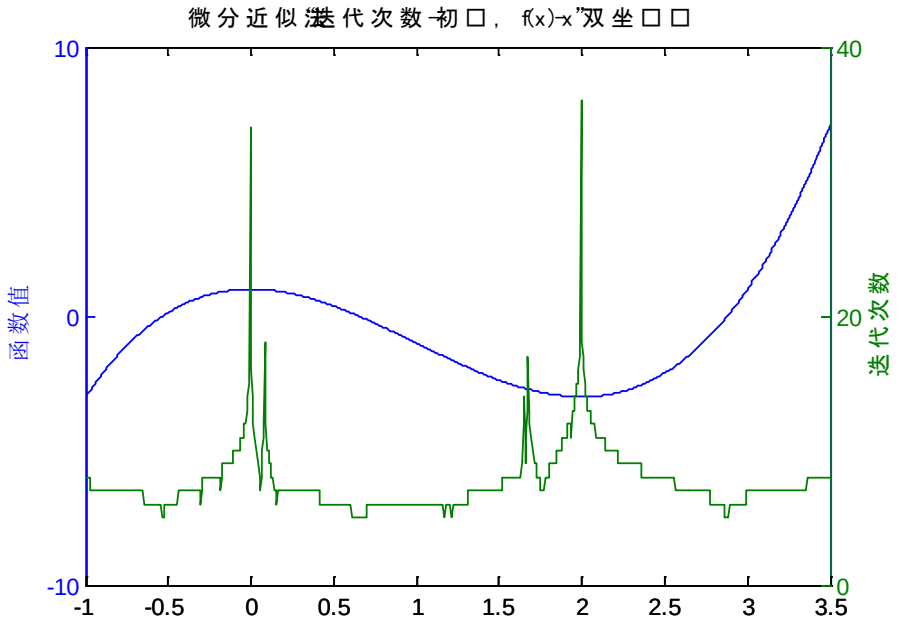
可以看到，当δx_i 取到恰好为$x_i + \delta x_i = x_{i-1}$时，这种方法既是课本中介绍的割线法，这里不再讨论。

下面给出“近似微分”方法的一个程序，本例中取δx_i为绝对增量即 $\eta = \delta x_i \equiv 0.01$；同

时也可以取相对增量形式 $\eta = \frac{\delta x_i}{x_i} = 0.01$。

```
% Extended Newtown method is developed to solve the root of nonlinear
% equation f stands for the function
clear;
x1 = 1;
x2 = 1.1;
e_a = 5e-11;
ita = 0.01;
while (abs(x2-x1)>e_a)
    x1 = x2;
    x2 = x1-ita*f(x1)/(f(x1+ita)-f(x1));
end
```

求解结果依然达到精度要求。虽然它可以应对非多项式的非线性方程，但是它依然不能解决迭代初值在极值附近的问题。可以得到与牛顿法中类似的双坐标分析图：



牛顿法思考——“避开极值点”

最朴素的思想是确定迭代求根的界限，避免因振荡、慢收敛或发散的情况导致无限循环。但是如何确定这个界限也是十分有技巧的。查阅大量文献后发现，在求解多项式方程的领域内，这个问题牵扯到了求多项式全部根的问题。

求多项式的全部实数根的方法概述[iv]

前文探讨的数值求根方法大多是针对于求感兴趣区间内的一个根，但是很多场合，比如控制领域希望知道所有的零极点，这就对非线性方程，特别是多项式方程的全部根求解提出了要求。一种可行的方法是将整个区间分解成多个可以用牛顿法求解的子区间，逐个求解。

要得到子区间，首先要求一个总体的上下界。

首先，查阅大量文献，由文献【iii】可以在整个实数域上求出所有根的一个上下界：

- 所有根上界 B 的求法：用首相系数对多项式的全部系数归一化，得新的首相 a0 =1 , 设 ak (k ≥1)是首相系数 a0 以后第一个为负数的系数。若不存在这样的 ak , 即全部系数为正, F(x) = 0 不会有正根, 此时直接令 B = 0；若存在，再设 Q 是所有负系数的绝对值的最大值, 则 F(x) = 0 正根上界为 $B = 1 + \sqrt[k]{Q}$；
- 所有根下界 A 的求法：利用上面的方法，先求方程 F(- x) = 0 的正根上界 B′,则原方程 F(x) = 0 的负根下界为 A = - B′ 。
- 由于 A ≤0 且 B ≥0，根 x = 0 是一定会落在区间[A , B] 中的。于是得到 F(x) = 0 实根的上下界[A , B]。

然后，反复利用(3)、(4)式，递归调用求解过程求解，于是目前可以假设已经得到了方程的所有极点 $\{x_k\}$ 。

其次，针对每个区间使用牛顿迭代法，不妨选取初始点为子区间的中点【脚注 [2]】，根据给定误差标准 ε ，求出所有实根的数值解。

小结

本文由实际问题出发，按照划界法以及开方法为主线，阐述了 5 中求解方法，较深入地讨论每一种求解方法，以及其中的注意要点，并用每一种方法求得了最终的结果（小数点后 10 位有效数字）即球心到该划分平面的距离为 $d = 0.3472963553 \times R$，R 为球的半径。

同时本文花费大量精力在每一种方法上进行横向比较与发散思维，得到了许多有趣而有意义的讨论结果，如关于：并行性、多项式求根、求所有实数根等。但是由于时间有限，很多有趣的结论以及方法还没有来得及实现，还有一些想法没能够表达出来。

总的来看，非线性方程的求根更具其应用场合不同、应用要求不同而可以产生不同的对策、方法，“最好”的办法就是对实际问题实际分析，最终得到一个最适合问题本身的求解策略。如对于本次作业的原题，用球分割的物理概念先估计一个大概的根 x 必在(0.5,1)之间，再用牛顿法求解，就会十分迅速地收敛到想要精度的根。特别说明的是，数值求根时“小数点后 10 位有效数”与“10 位有效数字”两种不同的要求，其计算方法的停止条件是不同的，前者是对绝对误差做约束，而后者是对相对误差做约束。

最后要说明的是作业中提到的各种方法均只适用于连续函数的求解，因为划界法和开方法的理论基础就是基于连续函数。如果要对非连续的函数，可以如遗传算法、模拟退火等随机优化算法。

[2] 另一种可行的方法是多项式的二节极点（可获得最大的 f′(x)）

[i] http://baike.baidu.com/view/1382952.htm

[ii] 廖章钜，牛顿选代法与剖分相结合的一种多项式求根算法，北京联合大学学报，1998 年 3 月第 12 卷

[iii] 课件 第二章 pp20

[iv] 王祖樾，方程与多项式[M]，杭州，浙江人民出版社，1974，pp82 - 88