# 实验二

1. **实验内容**

1.3 编程实现阻尼牛顿法。

要求：（a）设定阻尼因子的初始值及解得误差阈值；（b）阻尼因子用逐次折半法更新；（c）打印每个迭代步的最终值及近似解。用所编程序求解：

（1）

（2）

1. **解题思路**

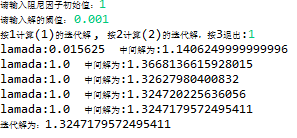
（注：程序均采用Java实现）

1. 根据方程写出；
2. 根据课本提供的阻尼牛顿法算法伪代码实现。
3. **算法设计**

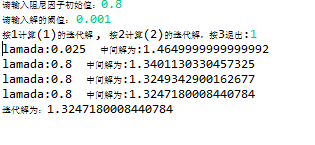
|  |
| --- |
| **import** java.util.Scanner;  **public** **class** Exp2  {  **static** **double** f1(**double** x)  {  **return** Math.*pow*(x, 3)-x-1;  }    **static** **double** f2(**double** x)  {  **return** -Math.*pow*(x,3)+5\*x;  }    **static** **double** df1(**double** x)  {  **return** Math.*pow*(x, 2)\*3 - 1;  }    **static** **double** df2(**double** x)  {  **return** -3\*Math.*pow*(x, 2) + 5;  }    **static** **public** **void** cal\_f1(**double** l , **double** e)  {  **double** xk\_1 = 0;  **double** xk = 0.6;  **double** xk1 = 0;  **while** (Math.*abs*(xk-xk\_1)>e)  {  **double** s = *f1*(xk)/*df1*(xk);  xk1 = xk - s;  **double** tmpl = l;  **while** (Math.*abs*(*f1*(xk1)) >= Math.*abs*(*f1*(xk)))  {  xk1 = xk - tmpl\*s;  tmpl = tmpl/2;  }  xk\_1 = xk;  xk = xk1;  System.*out*.println("lamada:"+tmpl+" 中间解为:"+xk );  }  System.*out*.println("迭代解为："+xk);    }    **static** **public** **void** cal\_f2(**double** l , **double** e)  {  **double** xk\_1 = 0;  **double** xk = 1.2;  **double** xk1 = 0;  **while** (Math.*abs*(xk-xk\_1)>e)  {  **double** s = *f2*(xk)/*df2*(xk);  xk1 = xk - s;  **double** tmpl = l;  **while** (Math.*abs*(*f2*(xk1)) >= Math.*abs*(*f2*(xk)))  {  xk1 = xk - tmpl\*s;  tmpl = tmpl/2;  }  xk\_1 = xk;  xk = xk1;  System.*out*.println("lamada:"+tmpl+" 中间解为:"+xk );  }  System.*out*.println("迭代解为："+xk);  }    **public** **static** **void** main(String[] args)  {  System.*out*.print("请输入阻尼因子初始值：");  Scanner s = **new** Scanner(System.*in*);  **double** lamada = s.nextDouble();  System.*out*.print("请输入解的阈值： ");  **double** emislon = s.nextDouble();  System.*out*.print("按1计算(1)的迭代解 , 按2计算(2)的迭代解，按3退出:");  **while** (**true**)  {  **int** choose = s.nextInt();  **if**(choose == 3)  **break**;  **if**(choose == 1)  {  *cal\_f1*(lamada , emislon);  }  **if**(choose == 2)  {  *cal\_f2*(lamada , emislon);  }  }    }  } |

1. **实验结果和结论**

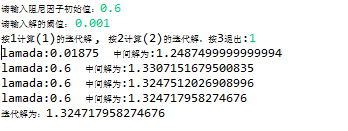
=0.001;



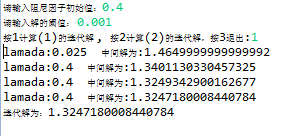
=0.001;



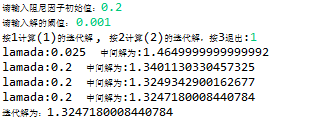
=0.001;



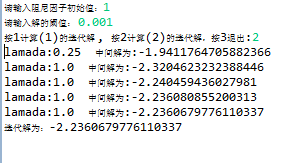
=0.001;



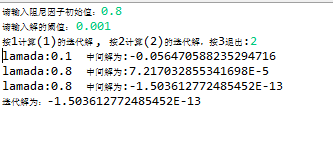
=0.001;



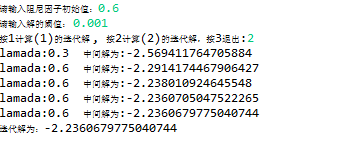
=0.001；



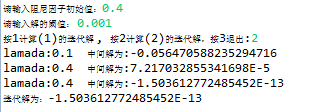
=0.001；



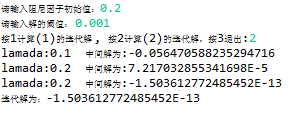
=0.001;



=0.001；



=0.001;



**五、遇到的问题**

开始时的思路一直是输入方程对应的函数，但是发现这是行不通的。好的做法是将待解方程对应的函数作为参数传入。在实现将函数作为参数传入的时候出了一些参数形式和调用方式的错误，后来到网上查找正确的传参和调用方法后解决问题。我们也可以看到，某些阻尼值是不收敛的。