gsl Example:
`roots.c` 的功能分析

吴泓鹰
数学与应用数学 (强基计划)　3210101890

2022 年 7 月 2 日

1　功能分析

简单版:该代码使用了布伦特方法 (Brent's method)[1]来求解函数 $f(x) = x^2 - 5$ 的根 (roots), 输出结果为 $x = \sqrt{5}$ 的估计之。

复杂版: 该代码设定初始化区间 $[a_0, b_0]$ 使得 $f(a_0) \cdot f(b_0) < 0$, 通过 k 次迭代得到区间 $[a_k, b_k]$ 使得 $f(a_k) \cdot f(b_k) < 0$, 其中 a_k, b_k 分别是上一次迭代中根的估计值与 a_{k-1}, b_{k-1} 中使得满足条件 $f(a_k) \cdot f(b_k) < 0$ 的一项 (可能顺序反过来), 当判断得到的估计值满足精度要求 (这里是 $|f| < 0.001$) 时停止迭代, 输出结果。迭代法求根的估计值一般使用二分法、割线法 (线性插值) 和逆二次插值法, 通过某种方法[1] 判断出最优选择使用其中的一个完成根的估计值的求取。

2　输出结果

在终端输入 `make` 后在.\bin 目录下生成可执行文件 `roots`, 在终端输入.\bin\roots 即可得到如下所示的输出结果:

```
using brent method
iter [    lower,     upper]      root         err  err(est)
1 [1.0000000, 5.0000000] 1.0000000 -1.2360680 4.0000000
2 [1.0000000, 3.0000000] 3.0000000 +0.7639320 2.0000000
3 [2.0000000, 3.0000000] 2.0000000 -0.2360680 1.0000000
4 [2.2000000, 3.0000000] 2.2000000 -0.0360680 0.8000000
```

[1]布伦特方法是在二分法或试位法的基础上, 借助二次插值方法进行加速, 有利用反插值方法来简化计算而形成的一种方法, 详见wikipedia

```
5 [2.2000000, 2.2366300] 2.2366300 +0.0005621 0.0366300
Converged:
6 [2.2360634, 2.2366300] 2.2360634 -0.0000046 0.0005666
```

第一列表示迭代次数，第二列第三列分别表示 a_k, b_k，第四列表示上一次迭代求出来的根的估计值，第五列第六列分别表示与 $\sqrt{5}$ 的误差和上一次估计值的误差，`Converged:`后表示最终结果。