编写求 arctan(x)的函数

数值分析与算法 第二次大作业

刘柏

自 36 | 2013011548

目录

1. 实验内容

（1） 目标：编写程序实现求 arctan(x)的函数；

（2） 要求：采用课程中至少 2 种方法；算法可达到任意精度。

2. 需求分析

2.1. 任意精度数值结构

由于要求算法本身能够达到任意精度，而现成的 C#中的浮点数类都存在最大精度的限制，所以需要编写一个数据结构存储数值，使之能表示任意精度的实数，并能够方便地进行四则运算。

2.2. 计算 arctan(x)

该部分为本次作业的核心，拟采用 4 种方法进行编写：最佳逼近法（泰勒展开法）、数值积分法（外推加速法）、常微分方程求解法（改进欧拉法）、方程求根法（牛顿法）。

注：在本题之下，改进欧拉法其实等价于复化梯形公式的数值积分方法，其在速度上会慢于外推加速法，不如直接采用外推加速法。所以最终未选用改进欧拉法。

2.3. 图形界面

一般的计算工具都有 GUI 界面，编写相应的操作界面能够大大方便数据的输入与结果的观测，故还需进行操作界面的 GUI 编写。

3. 方案设计

3.1. 模块设计

3.1.1. 任意精度数值结构

该部分采用 C#实现。先创建 Number 类，用于储存数值的整数部分、小数部分、整数部分长度、小数部分长度等信息。

3.1.2. 计算 arctan(x)

由于 C#几乎所有操作都需要在类类型中完成，故在 3.1.1.的基础上，创建 Calculation 类，用于进行四则运算、幂指数运算、数值截取等操作，为计算 arctan(x)作准备。

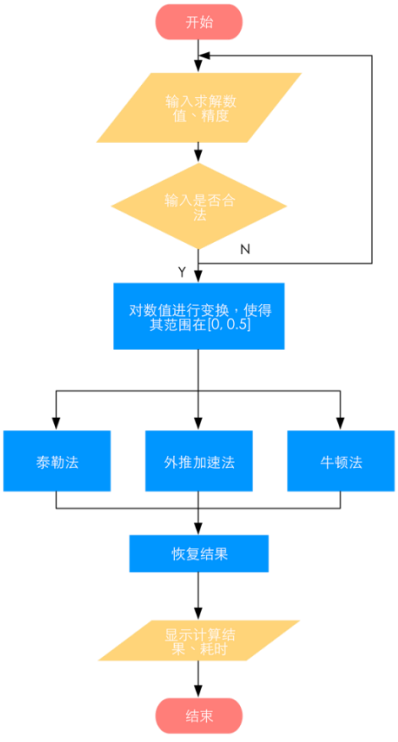
在计算 arctan(x)时，定义 Taylor，Romberg，Newton，分别代表泰勒法、外推加速法、牛顿法三种方法。

3.1.3. 图形界面

由于 WPF(Windows Presentation Foundation)具有设计方便、易于编辑的优点，且其与 C#配套得非常好，故该部分利用 WPF 实现。

3.2. 程序框图

绘制程序框图如下。



4. 基本原理与误差分析

4.1. 区间变换

4.1.1. 原理

由于在 x 较大时，部分方法收敛速度很慢甚至不能收敛，且若 x 为负数时，在自己设定的结构体进行四则运算比较复杂，故设想通过变换，将 x 映射到$y \in [0,\frac{1}{2}]$，先计算$arctany$，再根据变换关系求出$arctanx$.
具体而言，根据三角函数的性质，我们有如下结论：

$$\begin{cases} arctanx + arctan\dfrac{1}{x} = \dfrac{\pi}{2} \\ arctanx + arctany = \dfrac{\pi}{4} \Rightarrow y = \dfrac{1-x}{1+x} \\ \arctan(-x) = -arctanx \end{cases}$$

据此，可以规定变换关系如下

$$y = \begin{cases} -\dfrac{1}{x}, x < -2 \\ \dfrac{x+1}{x-1}, -2 \le x < -1 \\ \dfrac{1+x}{1-x}, -1 \le x < -0.5 \\ -x, -0.5 \le x < 0 \\ x, 0 \le x \le 0.5 \\ \dfrac{1-x}{1+x}, 0.5 < x \le 1 \\ \dfrac{x-1}{1+x}, 1 < x \le 2 \\ \dfrac{1}{x}, x > 2 \end{cases}$$

如此一来，可将 x 变换到$y \in [0,\frac{1}{2}]$，可以比较便捷地计算出$arctany$，之后再根据如下变换关系计算即可得到$arctanx$：

$$arctanx=\begin{cases} arctany-\dfrac{\pi}{2}, x<-2 \\ -arctany-\dfrac{\pi}{4}, -2\le x<-1 \\ arctany-\dfrac{\pi}{4}, -1\le x<-0.5 \\ -arctany, -0.5\le x<0 \\ arctany, 0\le x\le 0.5 \\ \dfrac{\pi}{4}-arctany, 0.5<x\le 1 \\ arctany+\dfrac{\pi}{4}, 1<x\le 2 \\ \dfrac{\pi}{2}-arctany, x>2 \end{cases}$$

4.1.2. 误差分析

（1） 观测误差

无。

（2） 模型误差

无。

（3） 方法误差

无。

（4） 舍入误差

假设每一步的存贮误差为$\frac{1}{2}\cdot 10^{-m}$.

x 的舍入误差是$|\delta x|\le\frac{1}{2}\cdot 10^{-m}$，π的舍入误差$|\delta\pi|\le\frac{1}{2}\cdot 10^{-m}$

代入 4.1.1.中的变换关系，可得

$$|\delta y|\le\begin{cases} \dfrac{1}{4}|\delta x|, x<-2 \\ \dfrac{1}{4}|\delta x|, -2\le x<-1 \\ \dfrac{8}{9}|\delta x|, -1\le x<-0.5 \\ |\delta x|, -0.5\le x<0 \\ |\delta x|, 0\le x\le 0.5 \\ \dfrac{8}{9}|\delta x|, 0.5<x\le 1 \\ \dfrac{1}{4}|\delta x|, 1<x\le 2 \\ \dfrac{1}{4}|\delta x|, x>2 \end{cases}$$

根据舍入误差对下一步结果的影响，有

$$|\delta y| \le \max_x \left|\frac{\partial y}{\partial x}\right| \cdot |\delta x| + \frac{1}{2} \cdot 10^{-m} \le |\delta x| + \frac{1}{2} \cdot 10^{-m} \le 1 \cdot 10^{-m}$$

在计算$arctany$时，有

$$|\delta(arctany)| \le \max_y \left|\frac{1}{1+y^2}\right| \cdot |\delta y| + \frac{1}{2} \times 10^{-m} \le \left|\frac{1}{1+0}\right| \cdot |\delta y| + \frac{1}{2} \cdot 10^{-m}$$

$$= |\delta y| + \frac{1}{2} 10^{-m} \le \frac{3}{2} \cdot 10^{-m}$$

在计算$arctanx$时，有

$$|\delta(arctanx)| \le \max_{arctany} \left|\frac{\partial(arctanx)}{\partial(arctany)}\right| \cdot |\delta(arctany)|$$

$$+ \max_{\pi} \left|\frac{\partial(arctanx)}{\partial \pi}\right| \cdot |\delta \pi| + \frac{1}{2} \cdot 10^{-m}$$

$$= |\delta(arctany)| + \frac{1}{2}|\delta \pi| + \frac{1}{2} \cdot 10^{-m} \le \frac{9}{4} \cdot 10^{-m}$$

（5） 总误差

综合（1）（2）（3）（4），总误差$|e| \le \frac{9}{4} \cdot 10^{-m}$.

4.2. 泰勒法

4.2.1. 原理

根据$arctanx' = \frac{1}{1+x^2} = \sum_{n=0}^{+\infty}(-1)^n \cdot x^{2n}, |x| < 1$，对两边进行不定积分，可得

$$arctanx = \sum_{n=0}^{+\infty} \frac{(-1)^n \cdot x^{2n+1}}{2n+1}, |x| < 1$$

故有$arctanx = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots + \frac{(-1)^n \cdot x^{2n+1}}{2n+1} + o(x^{2n+1}), |x| < 1$

4.2.2. 误差分析

（1） 观测误差

无。

（2） 模型误差

由 4.1.1.可知，经过区间变换后，可将 x 变换到$y \in [0, \frac{1}{2}]$，此时易知泰勒展开收敛到$arctany$，故模型误差不存在。

（3） 方法误差

泰勒展开的拉格朗日余项为$\Delta_n(y) = arctan\xi^{(n+1)} \frac{y^n}{(n+1)!}$，其中$\xi \in [0, \frac{1}{2}]$

设$\mathrm{M}=\max_{y\in[0,\frac{1}{2}]}\left|arctany^{(n+1)}\right|$，则$|\Delta_n(y)|\le\frac{M}{2^n(n+1)!}$

相应地，$|\Delta_n(x)|\le\max_{arctany}\left|\frac{\partial(arctanx)}{\partial(arctany)}\right|\cdot|\Delta_n(y)|=\frac{M}{2^n(n+1)!}$

（4） 舍入误差

假设每一步的存贮误差为$\frac{1}{2}\cdot10^{-m}$.

x 的舍入误差是$|\delta x|\le\frac{1}{2}\cdot10^{-m}$，π的舍入误差$|\delta\pi|\le\frac{1}{2}\cdot10^{-m}$

由 4.1.2（4），有

$$|\delta y|\le\max_x\left|\frac{\partial y}{\partial x}\right|\cdot|\delta x|+\frac{1}{2}\cdot10^{-m}\le|\delta x|+\frac{1}{2}\cdot10^{-m}\le1\cdot10^{-m}$$

那么根据舍入误差对下一步结果的影响，有

$$\begin{aligned}|\delta(arctany)|&\le\max_y\left|\frac{\partial(arctany)}{\partial y}\right|\cdot|\delta y|+\frac{1}{2}\cdot10^{-m}\\&=\max_y|1-y^2+y^4-y^6+\cdots+(-1)^n\cdot y^{2n}|\cdot|\delta y|+\frac{1}{2}\cdot10^{-m}\\&=\max_y\left(\frac{1-(-1)^{n+1}\cdot y^{2n+2}}{1+y^2}\right)|\delta y|+\frac{1}{2}\cdot10^{-m}\\&<\left[\frac{1+\left(\frac{1}{2}\right)^{2n+2}}{1+0}\right]|\delta y|+\frac{1}{2}\cdot10^{-m}=\left[\frac{3}{2}+\left(\frac{1}{2}\right)^{2n+2}\right]\cdot10^{-m}\end{aligned}$$

故舍入误差为

$$\begin{aligned}|\delta_n(x)|=|\delta(arctanx)|&\le\max_{arctany}\left|\frac{\partial(arctanx)}{\partial(arctany)}\right|\cdot|\delta(arctany)|\\&+\max_\pi\left|\frac{\partial(arctanx)}{\partial\pi}\right|\cdot|\delta\pi|+\frac{1}{2}\cdot10^{-m}\\&=|\delta(arctany)|+\frac{1}{2}|\delta\pi|+\frac{1}{2}\cdot10^{-m}\\&\le\left[\frac{9}{4}+\left(\frac{1}{2}\right)^{2n+2}\right]\cdot10^{-m}\end{aligned}$$

（5） 总误差

综合（1）（2）（3）（4），总误差

$$\begin{aligned}|e|&=|\Delta_n(x)|+|\delta_n(x)|\\&\le\frac{M}{2^n(n+1)!}+\left[\frac{9}{4}+\left(\frac{1}{2}\right)^{2n+2}\right]\cdot10^{-m}\end{aligned}$$

（6） 事前估计法

当要求精度满足小数点后 20 位时，要求$|e| < \frac{1}{2} \cdot 10^{-20}$，可令

$$\begin{cases} |\Delta_n(x)| < \dfrac{1}{4} \cdot 10^{-20} \\ |\delta_n(x)| < \dfrac{1}{4} \cdot 10^{-20} \end{cases}$$

考虑到$\mathrm{M} = \max\limits_{\mathrm{y} \in [0, \frac{1}{2}]} \left| arctany^{(n+1)} \right|$在本例中难以计算，考虑采用替代方法。注意到$\mathrm{y} \in [0, \frac{1}{2}]$时，$\frac{y^{2n+1}}{2n+1} > \frac{y^{2n+3}}{2n+3}$，故$|o(x^{2n+1})| < \max\limits_{\mathrm{y} \in [0, \frac{1}{2}]} \left| \frac{y^{2n+3}}{2n+3} \right| = \frac{1}{2^n(2n+3)}$

解$|\Delta_n(x)| < \frac{1}{2^n(2n+3)} < \frac{1}{4} \cdot 10^{-20}$，可得$\mathrm{n} > 61.46$，取$\mathrm{n} = 62$即可。

将 n 代入$|\delta_n(x)| \leq \left[\frac{9}{4} + \left(\frac{1}{2} \right)^{2n+2} \right] \cdot 10^{-m} < \frac{1}{4} \cdot 10^{-20}$，可得$\mathrm{m} > 20.95$，取$\mathrm{m} = 21$即可。

故可知，要求精度满足小数点后 20 位时，只需保证每一步存贮误差不超过$\frac{1}{2} \cdot 10^{-21}$即可。

4.3. 外推加速法

4.3.1. 原理

由于$arctanx' = \frac{1}{1+x^2}$，有

$$arctanx = \int_0^x \frac{1}{1+u^2} du$$

为能速度较快地计算数值积分，可采用外推加速法如下。

设$T_0^{(k)}$表示二分 k 次的复化梯形公式(T_{2k})，$T_n^{(k)}$表示$\left\{ T_0^{(k)} \right\}$的 n 次加速结果，则有

$$T_n^{(k)} = \frac{4^n}{4^n - 1} T_{n-1}^{(k+1)} - \frac{1}{4^n - 1} T_{n-1}^{(k)}$$

其中

$$T_0^{(k)}(x) = \frac{x}{2^{k+1}} \sum_{i=0}^{2^k - 1} \left[\frac{1}{1 + \left(\frac{ix}{2^k} \right)^2} + \frac{1}{1 + \left(\frac{(i+1)x}{2^k} \right)^2} \right]$$

每加速一次，收敛速度提高 2 阶。

4.3.2. 误差分析

（1） 观测误差

无。

（2） 模型误差

无。

（3） 方法误差

假设选取$T_n^{(k)}$作为结果。

当求积公式代数精度为l时，余项表达式为

$$\Delta_l(x)=\frac{1}{(l+1)!}\left[\frac{1}{l+2}(b^{l+2}-a^{l+2})-\sum_{k=0}^{n}A_k x_k^{l+1}\right]f^{(l+1)}(\eta)$$

在进行 n 次加速后，代数精度为$2n+1$；设$\max\limits_{\eta\in[a,b]}\left|f^{(l+1)}(\eta)\right|=M$，可得到近似如下。

$$|\Delta_n(y)|=|\Delta(arctany)|\le\sum_{i=0}^{2^k-1}\frac{M}{(2n+2)!}\frac{y}{2n+3}\left[\left(\frac{i+1}{2^k}\right)^{2n+3}-\left(\frac{i}{2^k}\right)^{2n+3}\right]$$

$$=\frac{My}{(2n+3)!}\le\frac{M}{2(2n+3)!}$$

故方法误差为

$$|\Delta_n(x)|=|\Delta(arctanx)|\le\max_{arctany}\left|\frac{\partial(arctanx)}{\partial(arctany)}\right||\Delta(arctany)|=\frac{M}{2(2n+3)!}$$

（4） 舍入误差

假设每一步的存贮误差为$\frac{1}{2}\cdot10^{-m}$.

x 的舍入误差是$|\delta x|\le\frac{1}{2}\cdot10^{-m}$，π的舍入误差$|\delta\pi|\le\frac{1}{2}\cdot10^{-m}$

由 4.1.2（4），有

$$|\delta y|\le\max_x\left|\frac{\partial y}{\partial x}\right|\cdot|\delta x|+\frac{1}{2}\cdot10^{-m}\le|\delta x|+\frac{1}{2}\cdot10^{-m}\le1\cdot10^{-m}$$

根据舍入误差对下一步结果的影响，有

$$|\delta_0(y)|\le\max_y\left|\frac{\partial T_0^{(k)}(y)}{\partial y}\right|+\frac{1}{2}\cdot10^{-m}$$

$$\le\frac{1}{2^{k+1}}\max_y\left|\frac{d\left(\sum_{i=0}^{2^k-1}\left[\frac{y}{1+\left(\frac{iy}{2^k}\right)^2}+\frac{y}{1+\left(\frac{(i+1)y}{2^k}\right)^2}\right]\right)}{dy}\right|\cdot|\delta y|+\frac{1}{2}$$

$$\cdot10^{-m}\le\frac{1}{2^{k+1}}\cdot[2^k\cdot(y+y)]|\delta y|+\frac{1}{2}\cdot10^{-m}\le\frac{1}{2}|\delta y|+\frac{1}{2}\cdot10^{-m}$$

$$\le1\times10^{-m}$$

相应地有

$$|\delta_n(y)| = |\delta(arctany)|$$

$$\leq \max_y \left| \frac{\partial\left(\frac{4^n}{4^n-1}T_{n-1}^{(k+1)}(y) - \frac{1}{4^n-1}T_{n-1}^{(k)}(y)\right)}{\partial y} \right| |\delta_{n-1}(y)| + \frac{1}{2}\cdot 10^{-m}$$

$$\leq \frac{5}{3}|\delta_{n-1}(y)| + \frac{1}{2}\cdot 10^{-m} \leq \left(\frac{5}{3}\right)^n \delta_0(y) + \left[\left(\frac{5}{3}\right)^n - 1\right]\cdot\frac{3}{4}\cdot 10^{-m}$$

$$\leq \left[\frac{7}{4}\cdot\left(\frac{5}{3}\right)^n - \frac{3}{4}\right]\cdot 10^{-m}$$

故舍入误差为

$$|\delta_n(x)| = |\delta(arctanx)| \leq \max_{arctany}\left|\frac{\partial(arctanx)}{\partial(arctany)}\right|\cdot|\delta(arctany)|$$

$$+ \max_{\pi}\left|\frac{\partial(arctanx)}{\partial\pi}\right|\cdot|\delta\pi| + \frac{1}{2}\cdot 10^{-m}$$

$$= |\delta(arctany)| + \frac{1}{2}|\partial\pi| + \frac{1}{2}\cdot 10^{-m}$$

$$\leq \frac{7}{4}\cdot\left(\frac{5}{3}\right)^n \cdot 10^{-m}$$

（5） 总误差

综合（1）（2）（3）（4），总误差

$$|e| = |\Delta_n(x)| + |\delta_n(x)|$$

$$\leq \frac{M}{2(2n+3)!} + \frac{7}{4}\cdot\left(\frac{5}{3}\right)^n \cdot 10^{-m}$$

（6） 事前估计法

当要求精度满足小数点后 20 位时，要求$|e| < \frac{1}{2}\cdot 10^{-20}$，可令

$$\begin{cases} |\Delta_n(x)| < \dfrac{1}{4}\cdot 10^{-20} \\ |\delta_n(x)| < \dfrac{1}{4}\cdot 10^{-20} \end{cases}$$

考虑到$\mathrm{M} = \max_{\eta\in[a,b]}\left|f^{(l+1)}(\eta)\right|$在本例中难以计算，考虑采用替代方法。在实际求解过程中，注意到加速次数基本不超过 8 次，可取$\mathrm{n} = 8$.

将 n 代入$|\delta_n(x)| \leq \frac{7}{4}\cdot\left(\frac{5}{3}\right)^n \cdot 10^{-m} < \frac{1}{4}\cdot 10^{-20}$，可得$\mathrm{m} > 22.62$，取$\mathrm{m} = 23$即可。

故可知，要求精度满足小数点后 20 位时，只需保证每一步存贮误差不超过$\frac{1}{2}\cdot 10^{-23}$即可。

4.4. 牛顿法

4.4.1. 原理

求解 arctanx 可化归为求解关于 u 的方程$\tan u - x = 0$

记$f(u) = \tan u - x$

选$\varphi(u) = u - \frac{f(u)}{f'(u)} = u - \left(\frac{\sin 2u}{2} - x\frac{\cos 2u+1}{2}\right)$

令$u_{n+1} = \varphi(u_n)$，进行迭代即可逼近u^*

4.4.2. 误差分析

（1） 观测误差

无。

（2） 模型误差

在进行坐标变换后的前提下进行分析。

由于$y \in [0, \frac{1}{2}]$，$u \in [0, \arctan(\frac{1}{2})]$，容易验证其满足牛顿法在$[0, \arctan(\frac{1}{2})]$上收敛的四个条件，故不存在模型误差。

（3） 方法误差

由$u_{n+1} = u_n - \left(\frac{\sin 2u_n}{2} - y\frac{\cos 2u_n+1}{2}\right)$，且其为二阶收敛，有

$$\Delta_{n+1}(y) = \frac{\varphi''(t_n)}{2}\Delta_n(y)^2$$

设$M = \max_{t\in[0,\arctan(\frac{1}{2})]} |\varphi''(t)| = \max_{t\in[0,\arctan(\frac{1}{2})]} \left|\frac{2\sin t}{(\cos t)^3}\right| = \frac{5}{4}$，而$|\Delta_0(y)| \le \arctan(\frac{1}{2})$，可得

$$|\Delta_n(y)| = \Delta(arctany) \le \left(\frac{M}{2}\right)^{2^n-1} |\Delta_0(y)|^{2^n} \le \frac{8}{5}\left(\frac{5}{8}\arctan\frac{1}{2}\right)^{2^n}$$

故方法误差为

$|\Delta_n(x)| = |\Delta(arctanx)| \le \max_{arctany} \left|\frac{\partial(arctanx)}{\partial(arctany)}\right| |\Delta(arctany)| = \frac{8}{5}\left(\frac{5}{8}\arctan\frac{1}{2}\right)^{2^n}$.

（4） 舍入误差

假设每一步的存贮误差为$\frac{1}{2}\cdot 10^{-m}$.

x 的舍入误差是$|\delta x| \le \frac{1}{2}\cdot 10^{-m}$，π的舍入误差$|\delta\pi| \le \frac{1}{2}\cdot 10^{-m}$.

由 4.1.2（4），有

$$|\delta y| \le \max_x \left|\frac{\partial y}{\partial x}\right| \cdot |\delta x| + \frac{1}{2}\cdot 10^{-m} \le |\delta x| + \frac{1}{2}\times 10^{-m} \le 1\cdot 10^{-m}$$

那么根据舍入误差对下一步结果的影响，有

$$|\delta_{n+1}(y)| \le \max_{u_n}\left|\frac{\partial\left[u_n-\left(\frac{sin2u_n}{2}-y\frac{cos2u_n+1}{2}\right)\right]}{\partial u_n}\right||\delta_n(y)|$$
$$+\max_{y\in\left[0,\frac{1}{2}\right]}\left|\frac{\partial\left[u_n-\left(\frac{sin2u_n}{2}-y\frac{cos2u_n+1}{2}\right)\right]}{\partial y}\right||\delta y|+\frac{1}{2}\cdot 10^{-m}$$
$$\le |\delta_n(y)|+|\delta y|+\frac{1}{2}\cdot 10^{-m}\le\frac{3(n+1)}{2}\cdot 10^{-m}$$

故舍入误差为

$$|\delta_n(x)|=|\delta(arctanx)|\le\max_{arctany}\left|\frac{\partial(arctanx)}{\partial(arctany)}\right|\cdot|\delta(arctany)|$$
$$+\max_{\pi}\left|\frac{\partial(arctanx)}{\partial\pi}\right|\cdot|\delta\pi|+\frac{1}{2}\cdot 10^{-m}$$
$$=|\delta(arctany)|+\frac{1}{2}|\delta\pi|+\frac{1}{2}\cdot 10^{-m}$$
$$\le\frac{6n+3}{4}\cdot 10^{-m}$$

（5） 总误差

综合（1）（2）（3）（4），总误差

$$|e|=|\Delta_n(x)|+|\delta_n(x)|$$
$$\le\frac{8}{5}\left(\frac{5}{8}\arctan\frac{1}{2}\right)^{2^n}+\frac{6n+3}{4}\cdot 10^{-m}$$

（6） 事前估计法

当要求精度满足小数点后 20 位时，要求$|e|<\frac{1}{2}\cdot 10^{-20}$，可令

$$\begin{cases}|\Delta_n(x)|<\frac{1}{4}\cdot 10^{-20}\\ |\delta_n(x)|<\frac{1}{4}\cdot 10^{-20}\end{cases}$$

解$|\Delta_n(x)|\le\frac{8}{5}\left(\frac{5}{8}\arctan\frac{1}{2}\right)^{2^n}<\frac{1}{4}\cdot 10^{-20}$，可得$n>5.37$，取$n=6$即可。

将 n 代入$|\delta_n(x)|\le\frac{6n+3}{4}\cdot 10^{-m}<\frac{1}{4}\cdot 10^{-20}$，可得$m>21.59$，取$m=22$即可。

故可知，要求精度满足小数点后 20 位时，只需保证每一步存贮误差不超过$\frac{1}{2}\cdot 10^{-22}$即可。

5. 实现方法

5.1. 任意精度数值结构

为实现该结构，定义类 Number，其中考虑到计算的方便性、初始化的易行性，进行如下定义：

```
public int[] intPart, decPart;          //整数部分，小数部分
public int intLength, decLength;        //整数部分长度，小数部分长度
public int sign;                        //正负号
```

5.2. 任意精度运算

该部分的主要功能是对 Number 类进行加、减、乘、除、幂、阶乘等运算，并提供数值截取、数值大小比较、小数点移位、数值补齐等操作，定义 Calculation 类如下表所示。

函数	功能
public Number Add(Number numA, Number numB, int accuracy)	任意精度加法
public Number Subtract(Number numA, Number numB, int accuracy)	任意精度减法
public Number Multiply(Number numA, Number numB, int accuracy)	任意精度乘法
public Number Divide(Number numA, Number numB, int accuracy)	任意精度除法
public Number Power(Number num, Number index, int accuracy)	任意精度幂计算
public Number Factorial(Number num)	阶乘计算
public Number NumFill(Number num, int fillIntLength, int fillDecLength)	数字补全（按照要求在空当处补“0”）
public Number NumTrim(Number num, int accuracy)	数字修剪（按照精度要求截取数字并删去多余的 0）
public Number NumMove(Number num, int distance)	数字移位（按照要求对小数点进行左右移位）
public Number NumIntSelect(Number num, int startIndex, int endIndex)	整数部分截取（按

	照要求截取整数部分）
public int AbsCompare(Number numA, Number numB)	数值比较（比较两数的绝对值大小）

其中，基本四则运算的实现思路为：

（1） 加法：将 numA 与 numB 补零，使得二者整数部分长度和小数部分长度相同，然后从小数末位开始，逐渐向前进行按位加法以及进位、借位等操作；

（2） 减法：更改 numB，使之成为-numB，再进行加法运算；

（3） 乘法：先通过小数点移位，将 numB 化为整数，再利用加法进行竖式乘法计算，最后将小数点移回去即可；

（4） 除法：先通过小数点移位，将 numA，numB 均化为整数，再利用减法进行竖式除法运算。

5.3. 泰勒法实现

在进行 4.1.所示的区间变换，将 x 映射到$y \in [0,\frac{1}{2}]$后，利用

$$arctany = \sum_{n=0}^{+\infty} \frac{(-1)^n \cdot y^{2n+1}}{2n+1}$$

即可进行计算。

在计算过程中，加法、减法、乘法均进行精确计算，除法则取数值精度为 accuracy+3。当$\frac{(-1)^n \cdot y^{2n+1}}{2n+1}$在 accuracy+3 精度下为 0 时，停止计算。

再利用 4.1.中的对应关系，即可根据$arctany$计算出$arctanx$.

5.4. 外推加速法实现

在进行 4.1.所示的区间变换，将 x 映射到$y \in [0,\frac{1}{2}]$后，利用

$$\begin{cases} T_0^{(k)}(y) = \dfrac{y}{2^{k+1}} \displaystyle\sum_{i=0}^{2^k-1} \left[\frac{1}{1+\left(\frac{iy}{2^k}\right)^2} + \frac{1}{1+\left(\frac{(i+1)y}{2^k}\right)^2} \right] \\ T_m^{(k)}(y) = \dfrac{4^m}{4^m-1} T_{m-1}^{(k+1)}(y) - \dfrac{1}{4^m-1} T_{m-1}^{(k)}(y) \end{cases}$$

即可进行计算（其中$T_m^{(k)}$表示$\{T_0^{(k)}\}$的 m 次加速结果）。

在具体实现中，当计算得到的相邻的$T_m^{(k)}(y)$在 accuracy+3 的精度下相同时，停止计算。

再利用 4.1.中的对应关系，即可根据$arctan$y计算出$arctan$x.

5.5. 牛顿法实现

在进行 4.1.所示的区间变换，将 x 映射到y ∈ $[0,\frac{1}{2}]$后，利用

$$u_{n+1} = u_n - \left(\frac{sin2u_n}{2} - y\frac{cos2u_n + 1}{2}\right)$$

进行迭代即可逼近u^*.

其中$sin2u_n$、$cos2u_n$可利用泰勒展开求得。

在选取初值u_0时，考虑到y ∈ $[0,\frac{1}{2}]$时，y 与tany的取值较为接近，故u^*与 y 也较为接近，

可令$u_0 \approx y$.

在具体实现中，当$\left(\frac{sin2u_n}{2} - y\frac{cos2u_n+1}{2}\right)$在 accuracy+3 的精度下等于 0 时，停止计算

再利用 4.1.中的对应关系，即可根据$arctan$y计算出$arctan$x.

6. 实验结果

6.1. 程序实现情况

6.1.1. 基本情况

（1） 语言：C#；

（2） 框架：Microsoft .NET Framework 4.5 (x86 和 x64)；

（3） IDE：Visual Studio 2013.

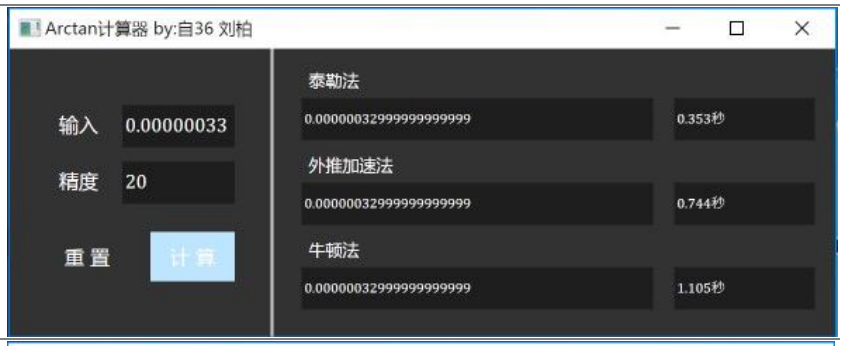
6.1.2. 基本功能

主界面如下图所示。

对典型数值的 20 位精度计算结果如下表所示。

数值	计算结果
-4	
-1.5	
-0.9	
-0.3	

0	 Arctan计算器 by:自36 刘柏 输入 0 精度 20 重置 计算 泰勒法 0.00000000000000000000 0.362秒 外推加速法 0.00000000000000000000 0.707秒 牛顿法 0.00000000000000000000 1.058秒
0.2	 Arctan计算器 by:自36 刘柏 输入 0.2 精度 20 重置 计算 泰勒法 0.19739555984988075837 0.346秒 外推加速法 0.19739555984988075837 0.884秒 牛顿法 0.19739555984988075837 1.588秒
0.6	 Arctan计算器 by:自36 刘柏 输入 0.6 精度 20 重置 计算 泰勒法 0.54041950027058415544 0.404秒 外推加速法 0.54041950027058415544 1.025秒 牛顿法 0.54041950027058415544 1.837秒
1.6	 Arctan计算器 by:自36 刘柏 输入 1.6 精度 20 重置 计算 泰勒法 1.01219701145133418326 0.766秒 外推加速法 1.01219701145133418326 1.373秒 牛顿法 1.01219701145133418326 2.209秒
5	 Arctan计算器 by:自36 刘柏 输入 5 精度 20 重置 计算 泰勒法 1.37340076694501586086 0.372秒 外推加速法 1.37340076694501586086 0.961秒 牛顿法 1.37340076694501586086 1.701秒

0.00000033	
100000000	

由上表可以看出，精度为 20 位时，在实数范围内，计算时间都在秒的量级，且三种方法计算得到的**结果完全一致**，经 Mathematica 验证，所得**结果正确**。

6.1.3. 附加功能

（1） 任意精度计算

本程序提供了精度改变的操作接口，用户能很方便地设置所需精度。例如，输入 0.5，精度要求 40 位小数点时，计算结果如下图所示。

不过需要注意的是，此时计算耗时会明显增加，大概会到 10 秒的量级。

（2） 友好用户界面

本程序提供了易于操作、简便直观的用户界面，还提供了“重置”按钮方便用户操作。

（3） 禁止非法输入

对文本框进行设置，使得用户在“输入”文本框仅能输入数字、负号、小数点，在“精度”文本框仅能输入数字，避免了非法输入带来的程序运行异常。

（4） 计时

为便于对比不同方法的速度，特设立运算耗时显示框，如下图所示。

6.2. 方法比较

方法	计算代价	理论收敛速度	实际收敛速度	存储要求
泰勒法	耗时最短	2n+1 阶收敛（n 为级数中所取的项数）	最快	不超过$\frac{1}{2}\cdot 10^{-21}$
外推加速法	泰勒法 2~3 倍的时间	2n+2 阶收敛（n 为加速次数）	较慢	不超过$\frac{1}{2}\cdot 10^{-23}$
牛顿法	泰勒法 3~5 倍的时间	2 阶收敛	最慢	不超过$\frac{1}{2}\cdot 10^{-22}$

6.3. 原因分析

（1） 泰勒法

可以看到，在十几个测试样例中，泰勒法的运算速度总是最快的。

考虑泰勒法的原理，可以发现，泰勒法中运算量主要集中在幂运算，而在除法运算中，除数都是 3、5、7 之类的整数。我在实现存储结构的四则运算时，采用的是竖式除法，假如除数不是整数，会将其通过小数点移位变成整数，相应地，竖式计算次数会增加。所以泰勒法速度比较快的原因可能在于除法运算耗时比较短。

（2） 外推加速法

外推加速法的运算速度明显慢于泰勒法，大概会消耗泰勒法 2~3 倍的时间。

考虑外推加速法的原理，可以发现，外推加速法中，需要用到 arctanx 的导数$\frac{1}{1+x^2}$，除数不一定是整数。我在实现存储结构的四则运算时，采用的是竖式除法，假如除数不是整数，会将其通过小数点移位变成整数，相应地，竖式计算次数会增加。具体到本例，若用户取 20 位精度，则在计算$\frac{1}{1+\left(\frac{iy}{2^k}\right)^2}$时，$\left(\frac{iy}{2^k}\right)^2$会有 2·accuracy 左右的精度，竖式计算次数会翻倍，除法就慢了。

（3） 牛顿法

牛顿法在三种方法中最慢，大概会消耗泰勒法 3~5 倍的时间。

考虑牛顿法原理，可能原因在于，每次迭代都需要利用泰勒法求$sin2u_n$、$cos2u_n$，使得运算次数明显增多。

7. 问题与解决方法

本次大作业理论与实现相并重，在完成的过程中遇到了不少问题，现将问题描述与解决方案介绍如下。

7.1. 乘除法运算速度问题

我在一开始实现 Calculation 类时，对乘法、除法采取的实现方法是：

（1） 乘法：将乘数通过小数点移位化成整数，累加被乘数，次数为乘数通过小数点移位化成的整数；

（2） 除法：将被除数通过小数点移位放大$10^{accuracy}$倍，再用除数去减被除数，直到差为负数。

对乘法与除法进行测试，一切正常。但是当我将其应用到泰勒法中，发现运算速度非常慢——精度为 8 位都要算将近一分钟。经过分析，发现原因在于除法的实现方法效率太低，原来的方法复杂度为 $O(10^n)$ ，每增加一位精度，除法运算时间都会增加 10 倍左右。

所以采用竖式除法的方法重新实现了除法，这样一来复杂度骤降为 O(n)，同时也将乘法的实现方式改为了竖式乘法，大大提升了运算速度。如 6.中所示，泰勒法在精度为 20 位时，

所用时间基本在 0.1 秒的数量级。

7.2. 改进欧拉法运算速度问题

我原本打算将问题改写为常微分方程求解的形式，用改进欧拉法来求解。在将其实现后，发现运算速度非常慢，基本没有办法在可接受时间内算到 20 位小数的精度。

分析原因，我发现，在本题之下，改进欧拉法其实等价于复化梯形公式的数值积分方法，其在速度上会慢于外推加速法，不如直接采用外推加速法。所以最终我决定选用外推加速法。

7.3. 正负相加问题

在实现牛顿法时，发现计算结果不收敛。进行调试后，发现问题在于$\left(\frac{sin2u_n}{2}-y\frac{cos2u_n+1}{2}\right)$不收敛。进一步地，发现$sin2u_n$计算结果正常，$cos2u_n$计算结果不对，在检查了很久 Newton 类后，发现问题在于，有一个地方在进行负数与正数的相加时，符号位反了。

于是到 Calculation 类中分析，一开始我以为问题出在交换数字的时候的深拷贝与浅拷贝的问题，改过来后问题依旧。最后才发现，原因在于 NumTrim()函数中，有一处语句忘记加到逻辑判断处理模块了，导致在某些情况下计算结果符号位不能被更新。改正后问题得到解决，

7.4. 牛顿法收敛速度问题

在开始时，牛顿法时常会出现不收敛的情况，在仔细分析一番后，发现原因在于，我在进行乘法运算时，并未限制乘法运算的精度，所以有时候在多进行几次乘法运算后，$\left(\frac{sin2u_n}{2}-y\frac{cos2u_n+1}{2}\right)$小数部分的长度会非常长，很难在短时间内等于相应精度下的 0，导致运算不能及时终止。

解决方法也比较简单，在牛顿法中将部分关键乘法部分的精度限制到 accuracy+3 即可。

8. 实验体会

本次大作业花了我两周左右的时间，自己在上面投入了非常多的精力，同时也收获了很多。

在一开始时，我花了大量的时间实现任意精度数值存储结构、相应的运算方法及配套的数字处理工具，在保证了各个方法的正确性和鲁棒性后才进行具体数值计算方法的实现。现在想来，在前期保证基础工具的有效性与实用性是非常重要的，毕竟在之后的其他数值方法的实现中，需要频繁地用到这些工具。这让我深刻体会到高质量的底层实现是多么重要。

分模块的思想也在本次试验中比较突出，基础运算、泰勒法、外推加速法、牛顿法、界面各为模块，相互之间比较独立，为调试带来了不少便利。

虽然代码实现与写报告的过程非常繁重，但在这一过程中，当自己成功实现任意精度四则运算、成功计算出和 Mathematica 一样的计算结果、成功将运算时间控制在秒数量级时，内心还是很有成就感的。