# 《数值分析与算法》

## 数值方法求$\pi^X$

张嘉玮

自 61

2016011528

2018 年 12 月 22 日星期六

# 目录

## 2.3 【$\ln(\pi)$】

### 2.3.1 改进的泰勒展开$\ln(\pi)$



### 2.3.2 数值积分之梯形梯形公式$\ln(\pi)$



### 2.3.3 数值积分之复化辛普生公式$\ln(\pi)$

## 2.4 【$\pi^X$】

### 2.4.1 泰勒展开$\pi^X$



### 2.4.2 改进的欧拉方法$\pi^X$



### 2.4.3 程求根结合辛普生公式$\pi^X$

[2] 周老师讲义及补充资料.....53

# 1、需求分析

本次实验基于数值方法求解按顺序$\pi,\ln(\pi),\pi^X$。采用包括最佳逼近、数值积分、数值微分、常微分方程求解等方法对各个数值进行求解。本次作业的的关键点在于根据不同的精度要求，对求解方法进行误差分许，进而确定不同方法做需要的迭代方法。

算法的关键环节在于误差分析，特别是上一步的误差对下一步的影响。

# 2. 算法设计与分析

## 2.1 程序截图

【$\pi$】

A、泰勒展开

B、改进的泰勒展开

C、数值积分之复化梯形公式

【$\ln(\pi)$】

A、改进的泰勒展开

B、数值积分之梯形梯形公式

C、数值积分之复化辛普生公式

【$\pi^X$】

A、泰勒展开

B、改进的欧拉方法

### C、方程求根结合辛普生公式

```
################  PI ##############################
方法一：泰勒展开求 PI ：
The number of iterations : 4000017
 PI = 3.141593

方法二：改进的泰勒展开求 PI<用于下一步> ：
The number of iterations : 30
 PI = 3.141592653589793

方法三：梯形公式 PI ：
The number of iterations : 1155
 PI = 3.141593


################  Ln(PI) ##############################
方法一：改进的泰勒展开求 Ln（PI）：
The number of iterations : 10
 Ln(PI) = 1.144730

方法二：梯形公式求 Ln（PI）：
The number of iterations : 1826
 Ln(PI) = 1.144730

方法三：辛普生公式 Ln（PI）<用于下一步>：
The number of iterations : 1738
 Ln(PI) = 1.1447298858494


################  (PI)^X ##############################
方法一：  泰勒展开求 PI^X：
请输入PI的次方 X（1~10）:10
The number of iterations : 54
 PI^10.000000 = 93648.047476

方法二：  改进的欧拉方法求 PI^X：
请输入PI的次方 X（1~10）:10
The number of iterations : 33554432
 PI^10.000000 = 93648.047476

方法三：  方程求根结合辛普生方法求积分 求 PI^X：
请输入PI的次方 X（1~4）:3
The number of iterations : 4
 PI^3.000000 = 31.006277

请按任意键继续. . .
```

*注:”The number of interactions”表示程序循环迭代的次数

## 2.2 求 $\pi$

### 2.2.1 泰勒展开求$\pi$

#### 2.2.1.1 算法原理

计算 PI，常用的方法是反三角函数。考虑：

$$\arctan(1)=\frac{\pi}{4}\longrightarrow\pi=4*\arctan(1)$$

同时有：

$$\arctan'(x)=\frac{1}{1+x^2}$$

$$\frac{1}{1+x^2}=1-x+x^2-\cdots+(-1)^n x^n+\cdots$$

则：

$$\arctan(x)=\int\frac{1}{1+x^2}dx=x-\frac{1}{3}*x^3+\frac{1}{5}*x^5-\cdots+\frac{(-1)^{n+1}}{2*n-1}x^{2n-1}-\cdots$$

$$\frac{\pi}{4}=1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\cdots+\frac{(-1)^n}{2*n-1}+\cdots$$

根据这样的展开，按照精度要求取不同的 n,便可以得到符合进度要求的 PI。

#### 2.2.1.2 误差分析

根据泰勒展开求解 PI 时，没有模型误差和观测误差，下面对方法（截断）误差和舍入误差进行分析。

**A、方法误差**

当将泰勒展开在第 N 项处进行截断，带来的方法误差为：

$$\Delta n = 4 * \sum_{n=N+1}^{\infty} \frac{(-1)^n}{2*n-1}$$

$$\Delta n < \frac{4}{2*N+1}$$

B、舍入误差

此处假设计算机在进行加减乘除运算时不带来舍入误差，仅仅在最后保存变量的数据时有存储误差。则每一步带来的存储误差为：$\delta = 0.5*10^{-m}$（其中 m 为所选变量的精确位数）。则在进行 N 步迭代后，引入的存储误差为：

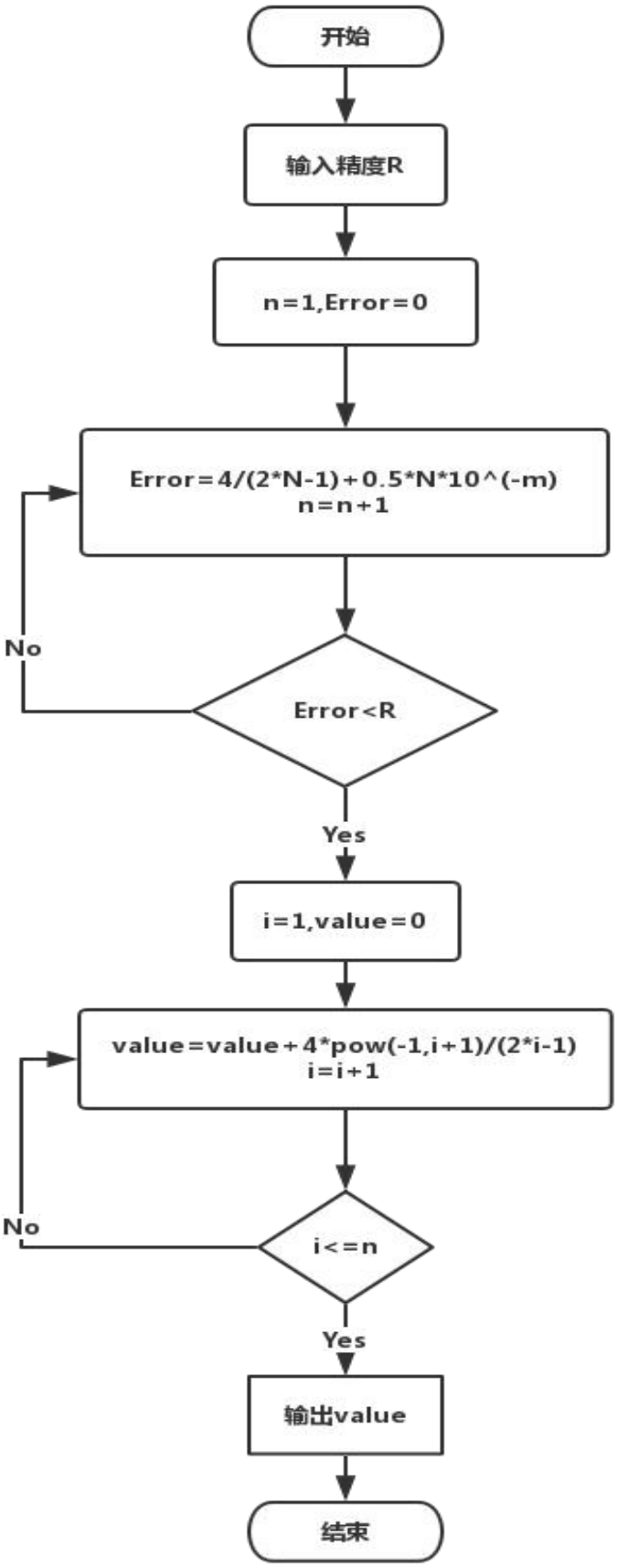
$$\delta n = N * \delta = 0.5 * N * 10^{-m}$$

C、综合考虑各种误差，总误差为：

$$Rn = \Delta n + \delta n$$

$$Rn \leq \frac{4}{2*N+1} + 0.5 * N * 10^{-m}$$

2.2.1.3 程序框图


开始
输入精度R
n=1,Error=0
Error=4/(2*N-1)+0.5*N*10^(-m)
n=n+1
No
Error<R
Yes
i=1,value=0
value=value+4*pow(-1,i+1)/(2*i-1)
i=i+1
No
i<=n
Yes
输出value
结束

#### 2.2.1.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加法 2 次，乘法 1 次，除法 1 次，则共需加法 2*N 次，除法和乘法各 N 次。

结合下面的实验，当需要 6 位小数精度时，需要加法：8000034 次；乘法和除法各 4000017 次。

#### 2.2.1.5 收敛速度

根据泰勒展开的通项，以及 $Rn \le \frac{4}{2*N-1} + 0.5*N*10^{-m}$，可知误差线性下降，$\mathrm{Rn} \sim O\left(\frac{1}{n}\right)$.

#### 2.2.1.6 实验

为了与各种方法进行对比，这一统一取精确到小数点后第 6 位。且变量统一使用 long double 类型的，其误差可精确到小数点后第 18 位（即 m=18,下同）。

当精确到小数点后第 6 位时，需要迭代 4000017 次，可见，线性收敛导致需要迭代的次数非常之多。当需要更高的精度要求时，不再适合。

### 2.2.2 改进的泰勒展开求 $\pi$

#### 2.2.2.1 算法原理

由于上面的泰勒展开其收敛速度是线性收敛，当需要更高精度的需求时便不再适合。因此需要收敛速度更快的泰勒展开。

任然考虑反三角函数：

$$\arctan(\frac{\sqrt{3}}{3})=\frac{\pi}{6}\longrightarrow\pi=6*\arctan(\frac{\sqrt{3}}{3})$$

则基于 2.1.2 的分析，有：

$$\frac{\pi}{6}=\arctan(\frac{\sqrt{3}}{3})$$

$$\frac{\pi}{6}=\frac{\sqrt{3}}{3}\left[1-\frac{1}{3*3}+\frac{1}{3^2*5}-\frac{1}{3^3*7}+\cdots+\frac{(-1)^n}{3^n*(2*n-1)}+\cdots\right]$$

根据这样的展开，按照精度要求取不同的 n,便可以得到符合进度要求的 PI。

### 2.2.2.2 误差分析

根据泰勒展开求解 PI 时，没有模型误差和观测误差，下面对方法（截断）误差和舍入误差进行分析。

#### A、方法误差

当将泰勒展开在第 N 项处进行截断，带来的方法误差为：

$$\Delta n=2\sqrt{3}*\sum_{n=N+1}^{\infty}\frac{(-1)^n}{3^n*(2*n-1)}$$

$$\Delta n<\frac{2\sqrt{3}}{3^N*(2*N+1)}$$

#### B、舍入误差

此处同样假设计算机在进行加减乘除运算时不带来舍入误差，仅仅在最后保存变量的数据时有存储误差。则每一步带来的存储误差为：$\delta=0.5*10^{-m}$（其中 m 为所选变量的精确位数）。则在进

行 N 步迭代后，引入的存储误差为：

$$\delta n = N * \delta = 0.5 * N * 10^{-m}$$

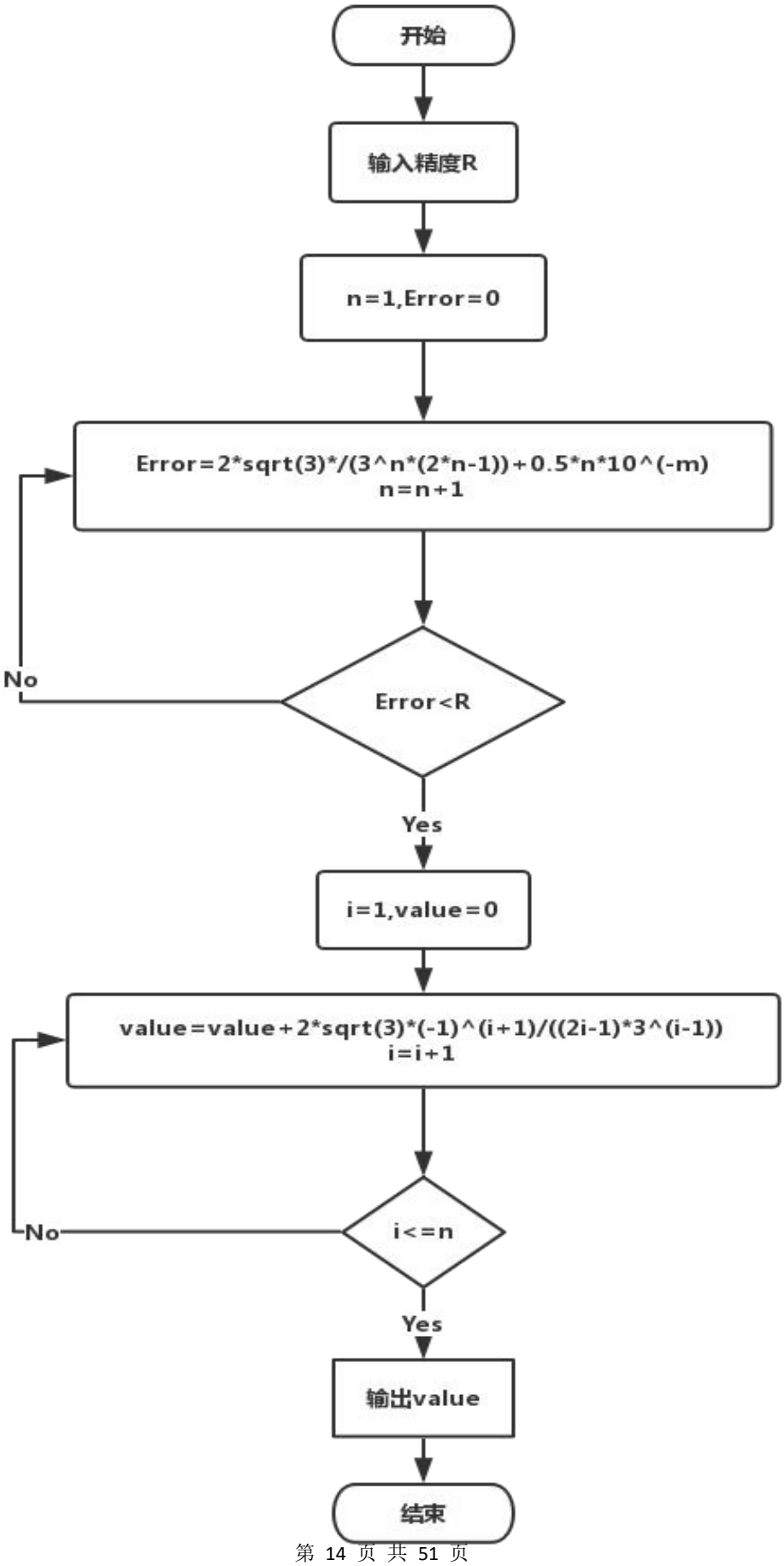
其中，m=18。

C、将两类误差求和便得到了总的误差：

$$Rn = \Delta n + \delta n$$

$$Rn \le \frac{2\sqrt{3}}{3^N * (2 * N + 1)} + 0.5 * N * 10^{-m}$$

2.2.2.3 程序框图


开始
输入精度R
n=1,Error=0
Error=2*sqrt(3)*/(3^n*(2*n-1))+0.5*n*10^(-m)
n=n+1
No
Error<R
Yes
i=1,value=0
value=value+2*sqrt(3)*(-1)^(i+1)/((2i-1)*3^(i-1))
i=i+1
No
i<=n
Yes
输出value
结束

#### 2.2.2.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加法 2 次，乘法 2 次，除法 1 次，则共需加法 2*N 次，乘法 2N 次，除法 N 次。

结合下面的实验，当需要 6 位小数精度时，需要加法 24 次；乘法 24 次；除法 12 次。

#### 2.2.2.5 收敛速度

根据泰勒展开的通项，以及 $Rn \le \frac{2\sqrt{3}}{3^N * (2*N+1)} + 0.5*N*10^{-m}$，可知其收敛速度较快，粗略估计，每两项便可以产生一个 PI 的小数进度，$\mathrm{Rn} \sim O\left(\frac{1}{3^n * n}\right)$.

#### 2.2.2.6 实验

当精确到小数点后第 6 位时，需要迭代 12 次，可见，其收敛速度极快，适合将其求解到更高进度，以带入后续操作。

综合各方面的误差，使其精确到小数点后 15 位，需要迭代 30 次，此时，需要加法 60 次，乘法 60 次，除法 30 次。

### 2.2.3 数值积分之复化梯形公式求 $\pi$

#### 2.2.3.1 算法原理

采用数值积分的方法对其进行积分，而这里选取较为常用的复化梯形公式，进行计算。计算的原理基于下面的积分等式：

$$\int_0^1 \frac{1}{1+x^2}dx = \frac{\pi}{4}$$

$$\pi = 4*\int_0^1 \frac{1}{1+x^2}dx$$

将区间进行 n 等分，在每个小区间上使用梯形公式，然后求和便可以得到 PI 的值。

$$h = \frac{1}{N}, f(x) = \frac{1}{1+x^2}$$

$$I = \sum_{k=0}^{N-1} \frac{h}{2}\left[f(x_k) + f(x_{k+1})\right]$$

$$I = \frac{h}{2}\left[f(0) + 2\sum_{k=1}^{N-1} f(x_k) + f(1)\right]$$

### 2.2.3.2 误差分析

同样，此方法有两类误差：方法误差和舍入误差。

#### A、方法误差

$$\Delta = \sum_{k=0}^{N-1} -\frac{h^3}{12} f''(\eta_k) = -\frac{(b-a)h^2}{12} f''(\eta)$$

$$f(x) = \frac{4}{1+x^2}$$

$$f''(x) = \frac{24x^2 - 8}{(1+x^3)^3}$$

$$\max(f''(x))\Big|_{x\in[0,1]} = 8$$

则放大输入误差后可得：$\Delta < \frac{2}{3n^2}$

**B、舍入误差**

同样，假设计算机在进行加减乘除运算时不引入误差，仅有在保存数据时会带来舍入误差，那么总的舍入误差：

$$\delta n = N * \delta = 0.5 * N * 10^{-m}$$

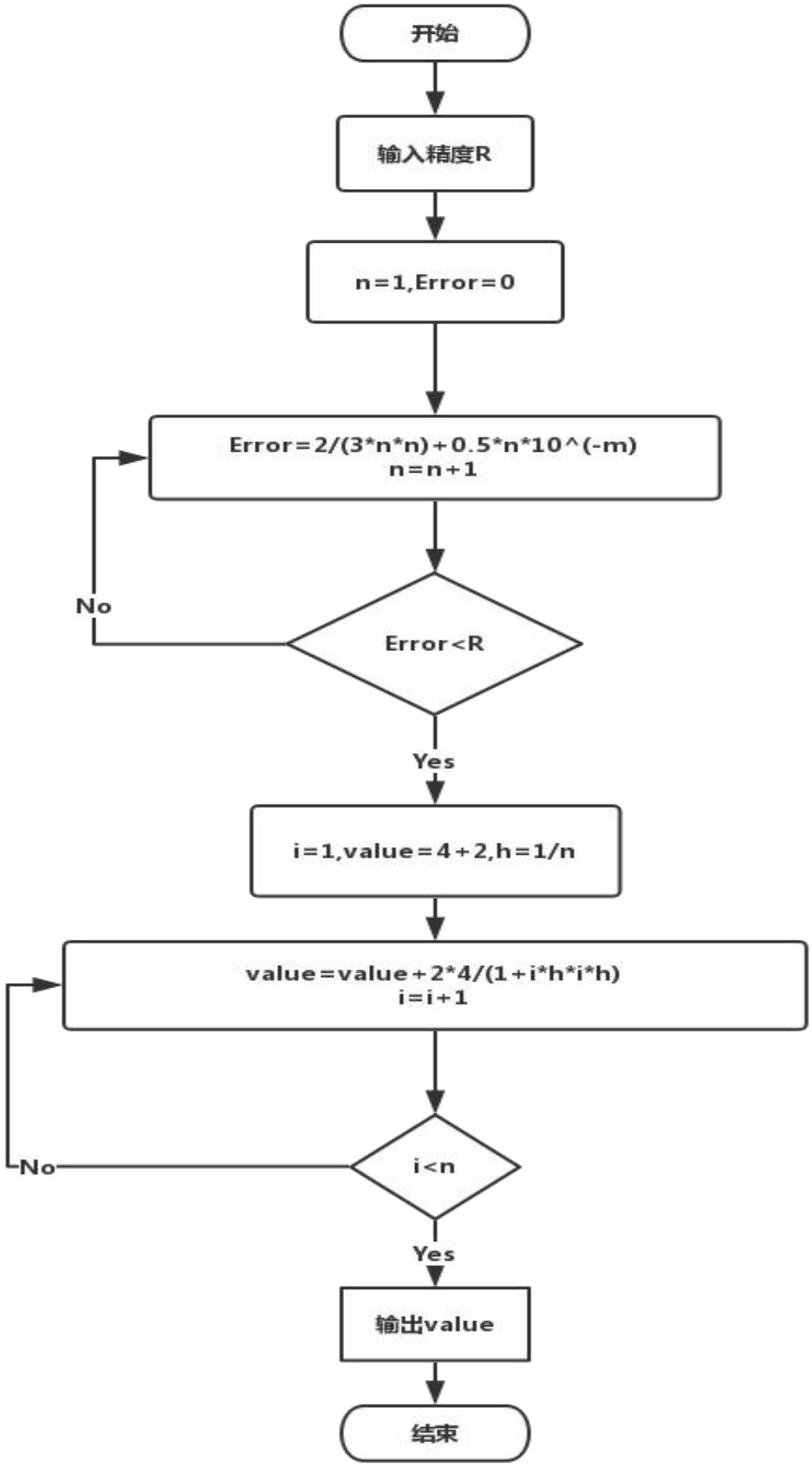
其中，m=18。

**C、将两类误差求和便得到了总的误差：**

$$Rn = \Delta + \delta n$$

$$Rn \le \frac{2}{3 * N^2} + 0.5 * N * 10^{-m}$$

**2.2.3.3 程序框图**


开始
输入精度R
n=1,Error=0
Error=2/(3*n*n)+0.5*n*10^(-m)
n=n+1
No
Error<R
Yes
i=1,value=4+2,h=1/n
value=value+2*4/(1+i*h*i*h)
i=i+1
No
i<n
Yes
输出value
结束

#### 2.2.3.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加法 2 次，乘法 4 次，除法 1 次，则共需加法 2*N 次，乘法 4*N 次，除法 N 次。

结合下面的实验，当需要 6 位小数精度时，需要加法：2310 次；乘法 4620 次，除法：1155 次。

#### 2.2.3.5 收敛速度

根据泰勒展开的通项，以及 $Rn \le \frac{2}{3*N^2} + 0.5*N*10^{-m}$，可知其收敛速度适中，二阶收敛 $\mathrm{Rn} \sim O\left(\frac{1}{n^2}\right)$.

#### 2.2.3.6 实验

当精确到小数点后第 6 位时，需要迭代 1155 次，可见，需要的迭代次数适中，即其适合精度要求适中的求解。

### 2.2.4 方法比较

以精度要求为 6 为基准，可得下面的实验结果：

| 方法 | 迭代次数 | 计算代价（加法+乘法+除法） |
|---|---|---|
| 泰勒展开 | 4000017 | 8000034+4000017+4000017 |
| 改进的泰勒展开 | 12 | 24+24+12 |
| 数值积分之复化梯形公式 | 1155 | 2310+4620+1155 |

## 2.3 求 $\ln(\pi)$

### 2.3.1 改进的泰勒展开求 $\ln(\pi)$

#### 2.3.1.1 算法原理

由于传统的欧拉展开式其收敛速度是线性收敛，当精度要求比较高时，线性收敛不在适合，因此需要对原本的泰勒展开进行改进。改进的思想主要是用换元的方法。

泰勒展开：

$$\ln(1+x)=x-\frac{x^2}{2}+\frac{x^3}{3}+\cdots+(-1)^{n-1}\frac{x^n}{n}+\cdots$$

换元，令 $x=\frac{1+u}{1-u}(u\in(-1,1))$，则：

$$\ln(x)=\ln(\frac{1+u}{1-u})=\ln(1+u)-\ln(1-u)$$

分别展开，则可以得到：

$$\ln(x)=2\left(u+\frac{u^3}{3}+\frac{u^5}{5}+\cdots\right)$$

这里：$x=\pi^*$，为上一步求得的 PI 值。

根据进度要求选取合适的 n，将前 n 项求和便可以近似得到 ln(PI) 的值。

#### 2.3.1.2 误差分析

分析误差可知，此处的误差有三个方面的来源：方法误差；舍入误差；上一步求得的 PI 的误差。

**A、方法误差**

$$\Delta n = \left|\ln(\pi) - S_{2n-1}\right|$$

$$\Delta n = 2\left|\frac{u^{2*n+1}}{2*n+1} + \cdots\right|$$

$$\Delta n \le 2*\frac{1}{2n+1}*u^{2n+1}\left|1+u^2+u^4+\cdots\right|$$

$$\Delta n < \frac{u^{2n-1}}{4*(2n+1)}$$

$$\text{其中：u} = \frac{\pi-1}{\pi+1}$$

**B、舍入误差**

此处同样假设计算机在进行加减乘除运算时不带来舍入误差，仅仅在最后保存变量的数据时有存储误差。则每一步带来的存储误差为：$\delta = 0.5*10^{-m}$（其中 m 为所选变量的精确位数）。则在进行 N 步迭代后，引入的存储误差为：

$$\delta n = N*\delta = 0.5*N*10^{-m}$$

其中，m=18。

**C、PI 引起的误差**

假设 PI 的精确到小数点后第 NLast1 位,考虑通项带来的误差：

$$f(x)=2*\frac{x^{2n-1}}{2n-1}$$

$$f'(x)=2*x^{2n-2}$$

$$\beta_{f(x)}\le\left|f'(x)\right|*\Delta_x$$

此处的 x 就是上面的 u，$x=\frac{\pi-1}{\pi+1}$，x<0.6，则可得到 PI 引起的总的误差为：

$$\beta\le 2*\left|1+x^2+x^4+\cdots+x^{2n-2}\right|*\Delta_x$$

$$\left|\mathrm{x}\right|\le 0.6$$

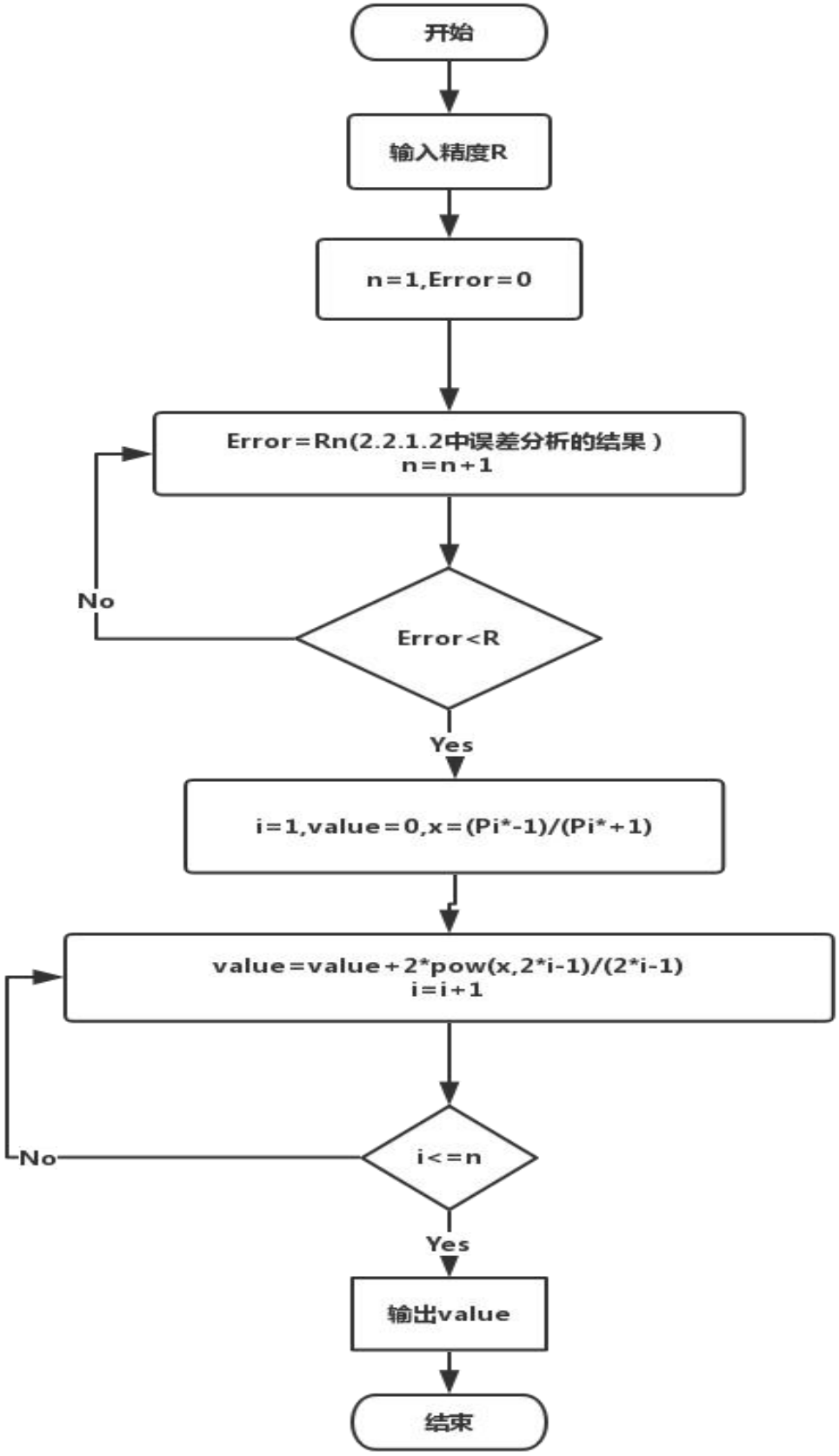
此处的$\Delta_x$与$\pi^*$的精度同量级，精确到小数点后第 NLast1 位。

**D、总的误差**

总误差即是将上面的所有误差相加（$x=\frac{\pi-1}{\pi+1}$）：

$$Rn\le\frac{x^{2N-1}}{4*(2N+1)}+0.5*N*10^{-m}+2*\left|\sum_{k=1}^{N}0.6^{2k-2}\right|*0.5*10^{-NLast1}$$

### 2.3.1.3 程序框图


开始
输入精度R
n=1,Error=0
Error=Rn(2.2.1.2中误差分析的结果）
n=n+1
No
Error<R
Yes
i=1,value=0,x=(Pi*-1)/(Pi*+1)
value=value+2*pow(x,2*i-1)/(2*i-1)
i=i+1
No
i<=n
Yes
输出value
结束

#### 2.3.1.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加减法共 4 次，乘法 3 次，除法 1 次，则共需加减法 4*N 次，乘法 3*N 次，除法 N 次。

结合下面的实验环节，当需要 6 位小数精度时，需要加减法：40 次；乘法 30 次，除法：10 次。

#### 2.3.1.5 收敛速度

总的误差如下：

$$Rn \le \frac{x^{2N-1}}{4*(2N+1)} + 0.5*N*10^{-m} + 2*\left|\sum_{k=1}^{N} 0.6^{2k-2}\right| *0.5*10^{-NLast1}$$

可知其收敛速度为 $O(\frac{x^{2N-1}}{4*(2N+1)}), x<0.6$，可知，其收敛速度较快。

#### 2.3.1.6 实验

当精确到小数点后第 6 位时，需要迭代 10 次，可见，其收敛速度较快，适合特别高的精度要求的场合。

同样也发现，随着迭代次数的增加，其方法误差在减小 ，但是由上一步所得的 PI 引起的误差却在增大，这是泰勒展开的缺点所在。

### 2.3.2 数值积分之复化梯形公式求 $\ln(\pi)$

#### 2.3.2.1 算法原理

基于下面的积分：

$$\ln(\pi) = \int_1^{\pi} \frac{1}{x} dx$$

采用数值积分的梯形公式:

$$h=\frac{\pi-1}{N},f(x)=\frac{1}{x}$$

$$I=\sum_{k=0}^{N-1}\frac{h}{2}\left[f(x_k)+f(x_{k+1})\right]$$

$$I=\frac{h}{2}\left[f(1)+2\sum_{k=1}^{N-1}f(x_k)+f(\pi)\right]$$

### 2.3.2.2 误差分析

同样，此处的误差有三个方面的来源：方法误差；舍入误差；上一步求得的 PI 的误差。

#### A、方法误差

$$\Delta=\sum_{k=0}^{N-1}-\frac{h^3}{12}f''(\eta_k)=-\frac{(b-a)h^2}{12}f''(\eta)$$

$$f(x)=\frac{1}{x}$$

$$f''(x)=\frac{2}{x^3}$$

$$\max(f''(x))\Big|_{x\in[1,\ PI]}=2$$

进行放缩:

$$\Delta\le\left|\frac{(\pi-1)^3}{12}*\frac{2}{n^2}\right|<\frac{10*2}{12*n^2}=\frac{5}{3*n^2}$$

**B、舍入误差**

此处同样假设计算机在进行加减乘除运算时不带来舍入误差，仅仅在最后保存变量的数据时有存储误差。则每一步带来的存储误差为：$\delta = 0.5*10^{-m}$（其中 m 为所选变量的精确位数）。则在进行 N 步迭代后，引入的存储误差为：

$$\delta n = N*\delta = 0.5*N*10^{-m}$$

其中，m=18。

**C、PI 引起的的误差**

$$I = \frac{h}{2}\left[ f(1) + 2\sum_{k=1}^{N-1} f(x_k) + f(\pi) \right]$$

考虑最终的计算式。可知，h 和 x 均和 PI 有关系，因此其值均受 PI 的精度所限。

则总的误差上限为：

$$\beta \le \frac{\partial I}{\partial h} * \Delta h + \frac{\partial I}{\partial (I/h)} * \Delta (I/h)$$

$$\frac{\partial I}{\partial h} = \left[ f(1) + 2\sum_{k=1}^{N-1} f(x_k) + f(\pi) \right] \le 2*N$$

$$\frac{\partial I}{\partial (I/h)} * \Delta (I/h) \le \left| \sum 0.5*h*\frac{1}{x_k} \right| \Delta x \le \cdots \le 2*\Delta x$$

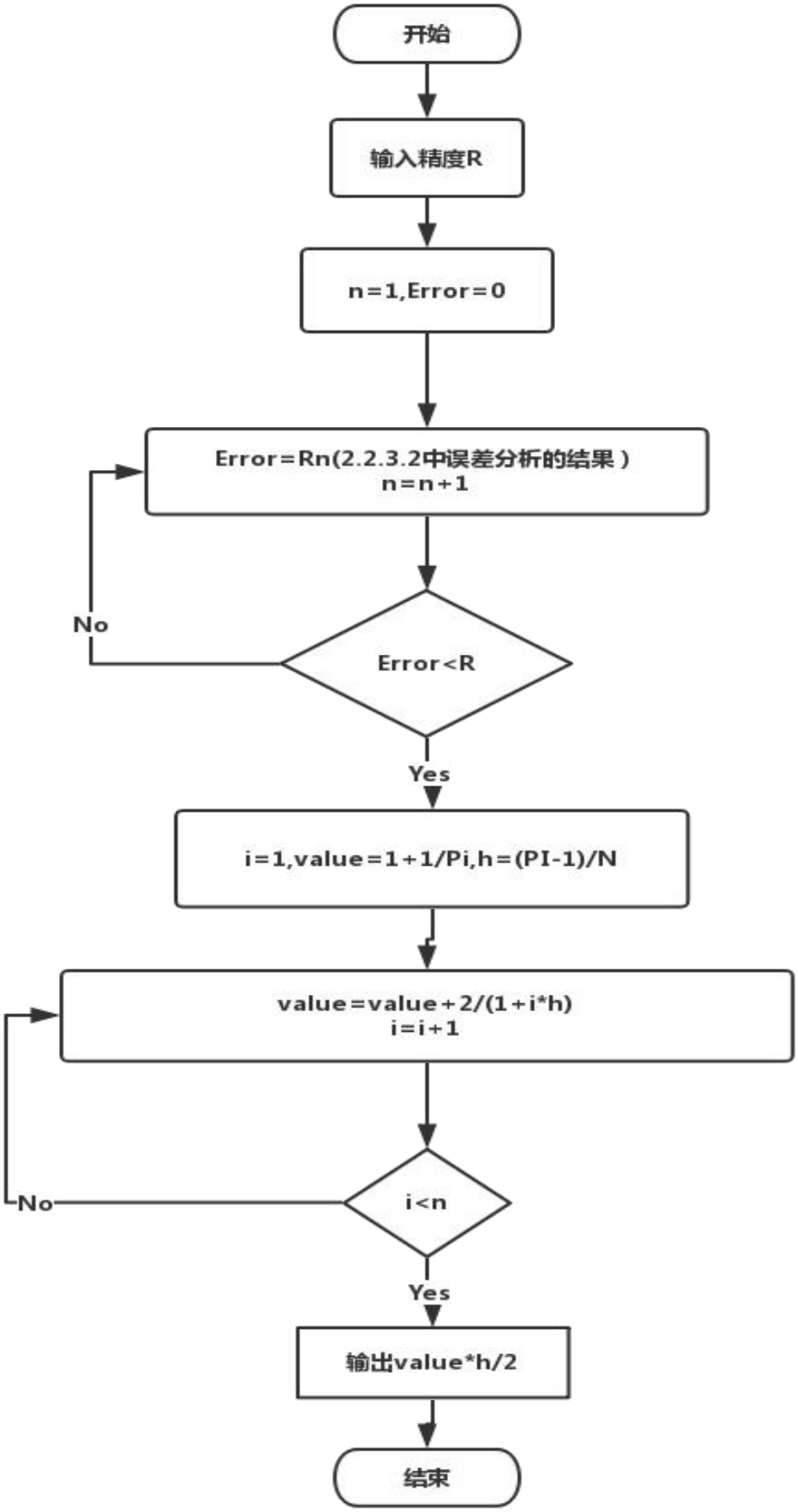
则 PI 带来的误差为 $4*\Delta Pi$，可见，这个值为定值。

**D、总的误差**

$$Rn \le \frac{5}{3*N^2} + 0.5*N*10^{-m} + 4*0.5*10^{-NLast1}$$

此处 NLast1 上一步求得的 PI 的精度，m=18。

#### 2.3.2.3 程序框图


开始
输入精度R
n=1,Error=0
Error=Rn(2.2.3.2中误差分析的结果）
n=n+1
No
Error<R
Yes
i=1,value=1+1/Pi,h=(PI-1)/N
value=value+2/(1+i*h)
i=i+1
No
i<n
Yes
输出value*h/2
结束

#### 2.3.2.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加减法共 3 次，乘法 1 次，除法 1 次，则共需加减法 3*N 次，乘法 N 次，除法 N 次。

结合下面的实验，当需要 6 位小数精度时，需要加法：1826*3=5478 次；乘法 1826 次，除法：1826 次。

#### 2.3.2.5 收敛速度

根据泰勒展开的通项，以及 $Rn \le \frac{5}{3*N^2} + 0.5*N*10^{-m} + 4*0.5*10^{-NLast1}$，可知其收敛速度适中，二阶收敛 $\mathrm{Rn} \sim O\left(\frac{1}{n^2}\right)$.

#### 2.3.2.6 实验

当精确到小数点后第 6 位时，需要迭代 1826 次，可见，其收敛速度较适中，适合较高的精度要求的场合，但是由于其值要用于下一步的计算，其收敛速度还是有所欠缺。

### 2.3.3 数值积分之复化辛普生公式求 $\ln(\pi)$

#### 2.3.3.1 算法原理

基于下面的积分：

$$\ln(\pi) = \int_1^{\pi} \frac{1}{x} dx$$

采用数值积分的辛普生公式：

$$h=\frac{\pi-1}{N},f(x)=\frac{1}{x}$$

$$I=\sum_{k=0}^{N-1}\frac{h}{6}\left[f(x_k)+4f(x_k+\frac{h}{2})+f(x_{k+1})\right]$$

$$I=\frac{h}{6}\left[f(1)+2\sum_{k=1}^{N-1}f(x_k)+4\sum_{k=0}^{N-1}f(x_k+\frac{h}{2})+f(\pi)\right]$$

### 2.3.3.2 误差分析

同样，此处的误差有三个方面的来源：方法误差；舍入误差；上一步求得的 PI 的误差。

**A、方法误差**

$$\Delta=\sum_{k=0}^{N-1}-\frac{h^5}{2880}f^{(4)}(\eta_k)=-\frac{(b-a)h^4}{2880}f^{(4)}(\eta)$$

$$f(x)=\frac{1}{x}$$

$$f^{(4)}(x)=\frac{24}{x^5}$$

$$\max(f''(x))\Big|_{x\in[0,1]}=24$$

进行放缩：

$$\Delta\le\frac{46*24}{2880*n^4}$$

**B、舍入误差**

此处同样假设计算机在进行加减乘除运算时不带来舍入误差，

仅仅在最后保存变量的数据时有存储误差。则每一步带来的存储误差为：$\delta = 0.5*10^{-m}$（其中 m 为所选变量的精确位数）。则在进行 N 步迭代后，引入的存储误差为：

$$\delta n = N*\delta = 0.5*N*10^{-m}$$

其中，m=18。

**C、PI 引起的的误差**

$$I = \sum_{k=0}^{N-1} \frac{h}{6}\left[ f(x_k) + 4f(x_k + \frac{h}{2}) + f(x_{k+1}) \right]$$

考虑最终的计算式。可知，h 和 x 均和 PI 有关系，因此其值均受 PI 的精度所限。

则总的误差上限为：

$$\beta \le \frac{\partial I}{\partial h} * \Delta h + \frac{\partial I}{\partial (I/h)} * \Delta(I/h)$$

$$\frac{\partial I}{\partial h} = \frac{1}{6}\left[ f(1) + 2\sum_{k=1}^{N-1} f(x_k) + 4f(x_k + \frac{h}{2}) + f(\pi) \right] \le N$$

$$\frac{\partial I}{\partial (I/h)} * \Delta(I/h) \le 2 * \Delta x$$

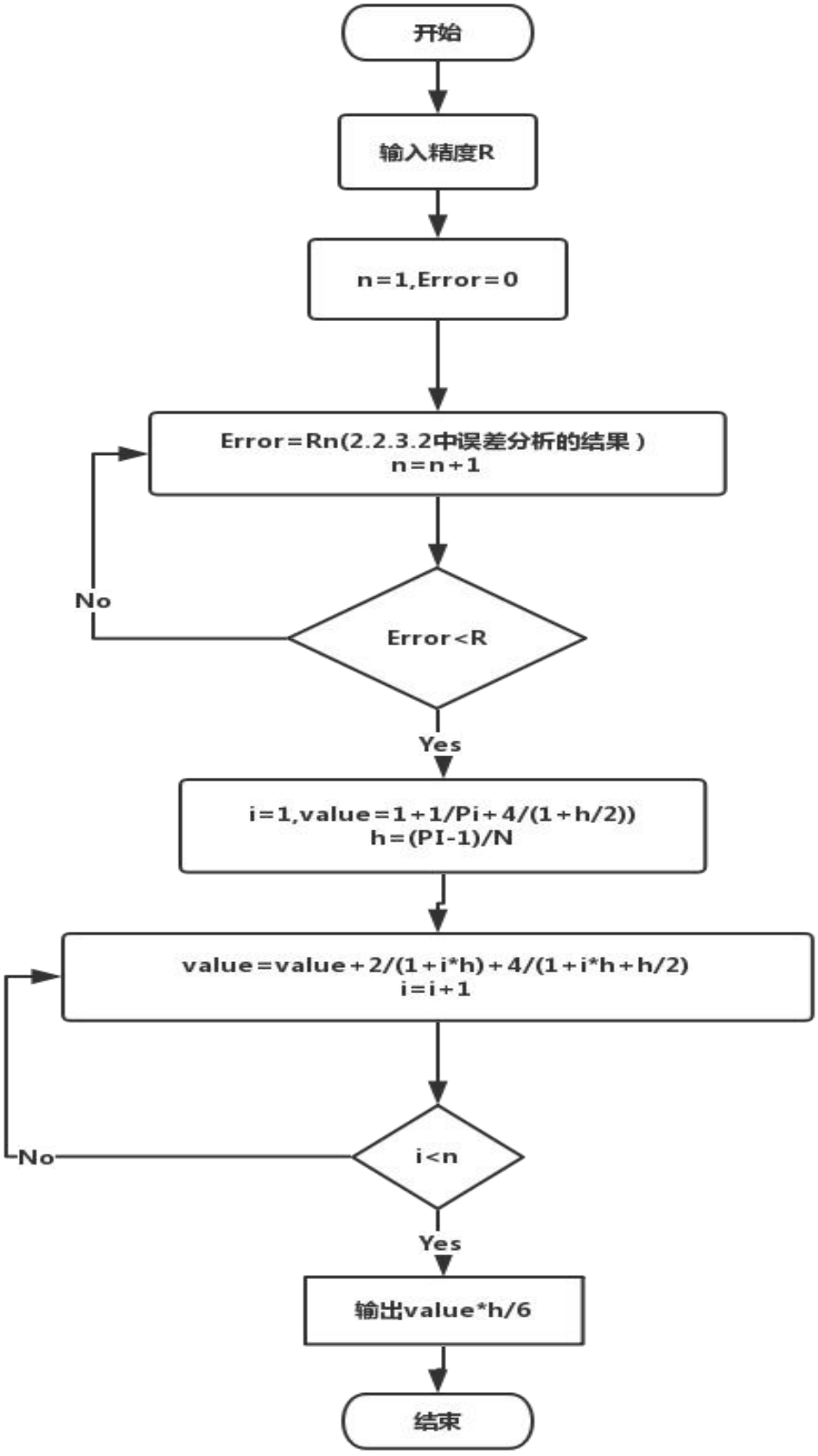
则 PI 带来的误差为 $3*\Delta Pi$，可见，这个值为定值，即不随 N 的增大而增大。

**E、总的误差**

$$Rn \le \frac{49*24}{2880*N^4} + 0.5*N*10^{-m} + 3*0.5*10^{-NLast1}$$

此处 NLast1 上一步求得的 PI 的精度,m=18。

### 2.3.3.3 程序框图


开始
输入精度R
n=1,Error=0
Error=Rn(2.2.3.2中误差分析的结果）
n=n+1
No
Error<R
Yes
i=1,value=1+1/Pi+4/(1+h/2))
h=(PI-1)/N
value=value+2/(1+i*h)+4/(1+i*h+h/2)
i=i+1
No
i<n
Yes
输出value*h/6
结束

#### 2.3.3.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加减法共 5 次，乘法 2 次，除法 2 次，则共需加减法 5*N 次，乘法 2*N 次，除法 2*N 次。

结合下面的实验，当需要 6 位小数精度时，需要加法：30*5=150 次；乘法 60 次，除法：60 次。

2.3.3.5 收敛速度

根据泰勒展开的通项，以及 $Rn \le \frac{49*24}{2880*N^4} + 0.5*N*10^{-m} + 3*0.5*10^{-NLast1}$，可知其收敛速度适中，四阶收敛 $\mathrm{Rn} \sim O\left(\frac{1}{n^4}\right)$.

#### 2.3.3.6 实验

当精确到小数点后第 6 位时，需要迭代 30 次，可见，其收敛速度较快，适合高的精度要求的场合。同时上一步的 PI 对其影响也较小，故选择辛普生公式的计算结果进行下一步的计算。

当精确到小数点后第 13 位时，需要迭代 1738 次。

### 2.3.4 方法对比

以精度要求为 6 为基准，可得下面的实验结果：

| 方法 | 迭代次数 | 计算代价（加法+乘法+除法） |
|---|---|---|
| 泰勒展开 | 10 | 40+30+10 |
| 数值积分之梯形公式 | 1826 | 5478+1826+1826 |

| | | |
|---|---|---|
| 数值积分之辛普生公式 | 30 | 150+60+60 |

以数值积分辛普生公式的计算结果作为下一步的计算值。精确到了小数点后第 13 位，需要迭代 1783 次。

## 2.4 求$\pi^X$

根据 2.2 的结果，获得了 ln(PI)的结果则有：

$$\pi^X = e^{\ln(\pi)}$$

根据上式，便可以求得$\pi^X$的值。

### 2.4.1 泰勒展开求$\pi^X$

#### 2.4.1.1 算法原理

将$e^x$在 x=0 处进行泰勒展开，有：

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots \frac{x^n}{n!} + \frac{x^{n+1}}{(n+1)!}e^{\xi}$$

令$x = X\ln\pi$根据精度以及 X，确定 n 的大小，便可以将泰勒展开的前 n 项求和，得到$\pi^X$的估计值。

#### 2.4.1.2 误差分析

同样，此处的误差有三个方面的来源：方法误差；舍入误差；上一步求得的 ln(PI)的误差。

**A. 方法误差**

截断误差来自上面的展开式的最后一项。

$$\Delta = \frac{e^{\xi}}{(n+1)!}(X\ln(\pi))^{n+1}$$

$$\xi \in (0,\ X\ln(\pi))$$

将截断误差进行放缩：

$$Xceil = \lceil X\ln(\pi) \rceil$$

$$\Delta < \frac{3^{Xceil}}{(n+1)!}Xceil^{n+1}$$

B. 舍入误差

此处同样假设计算机在进行加减乘除运算时不带来舍入误差，仅仅在最后保存变量的数据时有存储误差。则每一步带来的存储误差为：$\delta = 0.5*10^{-m}$（其中 m 为所选变量的精确位数）。则在进行 N 步迭代后，引入的存储误差为：

$$\delta n = N*\delta = 0.5*N*10^{-m}$$

其中，m=18。

C. $\ln(\pi)$ 引起的误差

有 $\ln(\pi)$ 引起的误差可有下面的推导求出：

$$\beta = \Delta e^{X\ln(\pi)}$$

$$\beta = \left(0+\frac{x}{1!}+\frac{x^2}{2!}+\cdots\cdots+\frac{x^{n-1}}{(n-1)!}\right)\Delta X\ln(\pi)$$

$$\beta = X*\left(0+\frac{x}{1!}+\frac{x^2}{2!}+\cdots\cdots+\frac{x^{n-1}}{(n-1)!}\right)\Delta\ln(\pi)$$

$$\beta \le X*\left(0+\frac{Xceil}{1!}+\frac{Xceil^2}{2!}+\cdots\cdots+\frac{Xceil^{n-1}}{(n-1)!}\right)\Delta\ln(\pi)$$

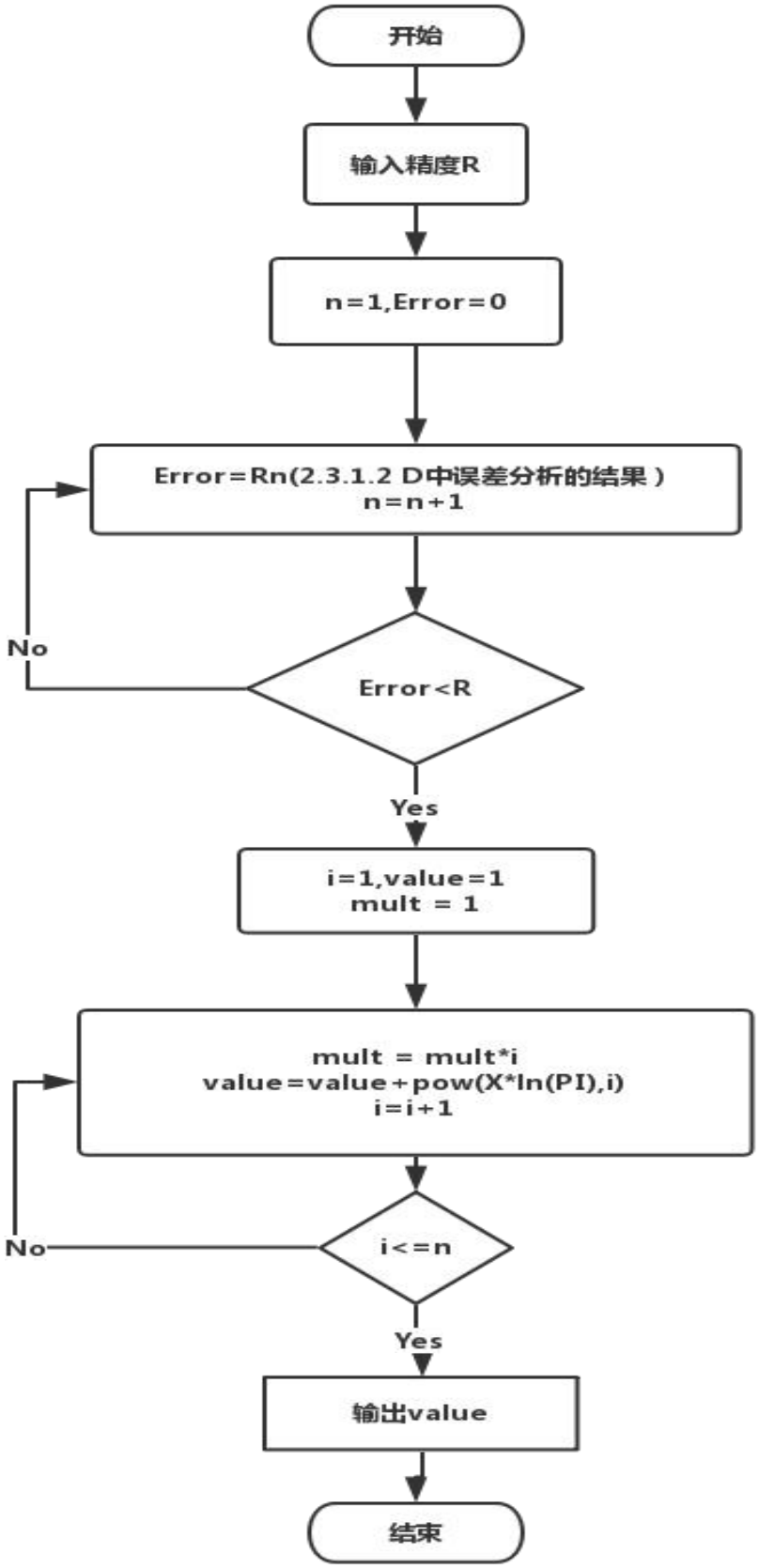
D. 总误差

总误差即将所有的误差相加即可求得上限。

$$Rn \le \frac{3^{Xceil}}{(n+1)!} Xceil^{n+1} + 0.5 * N * 10^{-m} + X * \left( 0 + \frac{Xceil}{1!} + \frac{Xceil^2}{2!} + \cdots\cdots + \frac{Xceil^{n-1}}{(n-1)!} \right) \Delta \ln(\pi)$$

其中，m=18。

### 2.4.1.3 程序框图


开始
输入精度R
n=1,Error=0
Error=Rn(2.3.1.2 D中误差分析的结果）
n=n+1
No
Error<R
Yes
i=1,value=1
mult = 1
mult = mult*i
value=value+pow(X*ln(PI),i)
i=i+1
No
i<=n
Yes
输出value
结束

#### 2.4.1.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加减法共 2 次，乘法 1 次，除法 1 次，则共需加减法 2*N 次，乘法 N 次，除法 N 次。

#### 2.4.1.5 收敛速度

有误差分析可知，此过程以$\frac{1}{(N+1)!}$的速度收敛，收敛速度极快。

#### 2.4.1.6 实验

当精确到小数点后第 6 位时，需要迭代 54 次，可见，其收敛速度较快，适合高的精度要求的场合。此时，共需加减法 2*54 次，乘法 54 次，除法 54 次。

### 2.4.2 常微分方程之改进的欧拉算法求$\pi^X$

#### 2.4.2.1 算法原理

有最后使用的函数$f(x)=e^x$可知，$f'(x)=e^x$。而函数有精确值$f(0)=1$，则求解问题即可化为：

已知：$y'=y$，且$y(0)=1$，求$y(X\ \ln(\pi))$

即问题可以转化为常微分方程求解的问题。这里使用计算机较为常用的改进的欧拉法解此问题。

#### 2.4.2.2 误差分析

同样，此处的误差有三个方面的来源：方法误差；舍入误差；上一步求得的 ln(PI)的误差。

**A. 方法误差**

根据改进的欧拉法的计算公式，可知，方法累计误差为：

$$\Delta_{n+1} \le (1 + hM + \frac{h^2}{2}M^2)\Delta_n + \left(\frac{LM}{4} + \frac{T}{12}\right) * h^3$$

其中：

$$\left|\frac{\partial f}{\partial y}\right| \le M, |y''(x)| \le L, |y'''(x)| \le T$$

而根据 $f'(x) = e^x$ 可取：

$$M = 1, L = T = e^{Xceil}$$

其中：$Xceil = \lceil X \ln(\pi) \rceil$

则代入公式：

$$\Delta_{n+1} \le (1 + h + \frac{h^2}{2})\Delta_n + \left(\frac{Xceil}{3}\right) * h^3$$

构建等比数列以及求和可知：

$$\Delta_{n+1} \le (1 + h + \frac{h^2}{2})^n (\Delta_0 + \frac{\frac{Xceil}{3}h^3}{h + \frac{h^2}{2}}) - \frac{\frac{Xceil}{3}h^3}{h + \frac{h^2}{2}}$$

则总的方法误差为：

$$\Delta \le ((1 + h + \frac{h^2}{2})^n - 1) * \frac{h^2}{1 + \frac{h}{2}} * \frac{Xceil}{3}$$

**B. 舍入误差**

用分析方法误差的方法同样可以求得舍入误差：

$$\delta \le (1 + h + \frac{h^2}{2})^n * (\delta_0 + \frac{0.5 * 10^{-m}}{h}) - \frac{0.5 * 10^{-m}}{h}$$

**C. $\ln(\pi)$ 引起的误差**

求解公式关于$\ln(\pi)$求得可得：

误差：

$$r = \ln(\pi)$$

$$\beta = |\frac{\partial e^{Xr}}{\partial r}| \Delta r = \left|Xe^{Xr}\right| \Delta r \le X * pow(2.8, Xceil) * \Delta r$$

### D、总误差

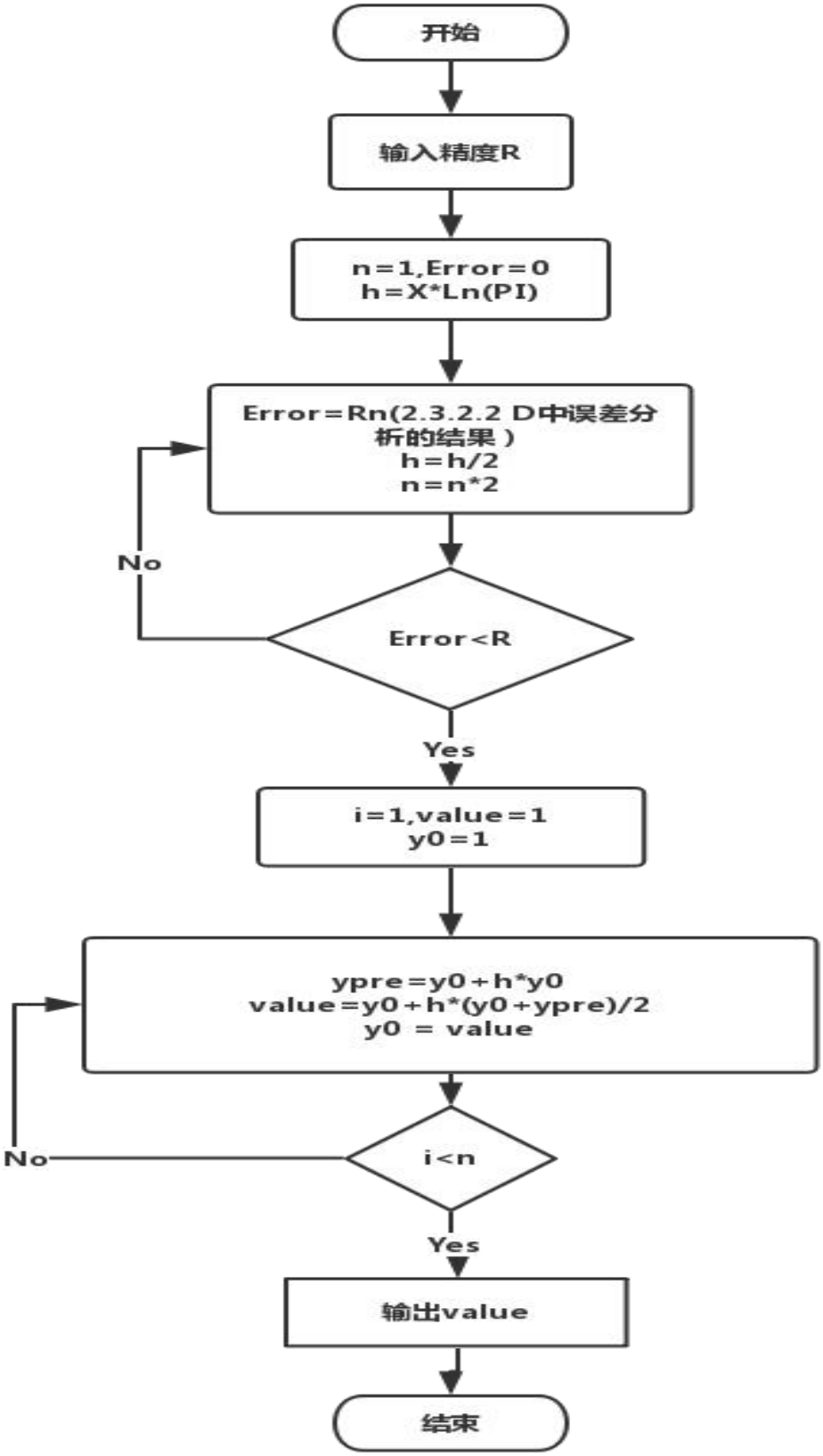
总误差即将所有的误差相加即可求得上限。

$$Rn \le ((1+h+\frac{h^2}{2})^n - 1) * \frac{h^2}{1+\frac{h}{2}} * \frac{Xceil}{3}$$

$$+ (1+h+\frac{h^2}{2})^n * (\delta_0 + \frac{0.5*10^{-m}}{h}) - \frac{0.5*10^{-m}}{h}$$

$$+ |\frac{\partial e^{Xr}}{\partial r}| \Delta r = \left|Xe^{Xr}\right| \Delta r \le X * pow(2.8, Xceil) * \Delta r$$

## 2.4.2.3 程序框图


开始
输入精度R
n=1,Error=0
h=X*Ln(PI)
Error=Rn(2.3.2.2 D中误差分析的结果）
h=h/2
n=n*2
No
Error<R
Yes
i=1,value=1
y0=1
ypre=y0+h*y0
value=y0+h*(y0+ypre)/2
y0 = value
No
i<n
Yes
输出value
结束

#### 2.4.2.4 计算代价

根据误差要求，假设需要计迭代 N 步，则根据就成图可知，每一步迭代需要加减法共 4 次，乘法 2 次，除法 1 次，则共需加减法 4*N 次，乘法 2*N 次，除法 N 次。

#### 2.4.2.5 收敛速度

考虑方法误差：

$$\Delta \le ((1+h+\frac{h^2}{2})^n - 1) * \frac{h^2}{1+\frac{h}{2}} * \frac{Xceil}{3}$$

故此方法近似一阶收敛。

#### 2.4.2.6 实验

当精确到小数点后第 6 位时，需要迭代 33554432 次，可见，其收敛速度相对较慢。此时，共需加减法 4*33554432 次，乘法 2*33554432 次，除法 33554432 次。

### 2.4.3 方程求根与数值积分

#### 2.4.3.1 算法原理

通过求解方程 $\int_1^x \frac{1}{t} dt = X * \ln(\pi)$，便可以得到 $\pi^X$。

对于方程求根，选择二阶收敛的牛顿法，即：

$$f(x) = \int_1^x \frac{1}{t} dt - X * \ln(\pi)$$

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

$$=x(1+X\ln(\pi)-\int_1^x\frac{1}{t}dt)$$

此处，为了使得迭代较快，选取初始值 $X_0=\left|\pi^X\right|$（上取整）。同时，上面的迭代当中含有积分项，因此在计算时需要计算积分。这里选取收敛速度相对较快的复化辛步生积分：

$$h=\frac{\pi^X\text{-}1}{N},f(x)=\frac{1}{x}$$

$$I=\sum_{k=0}^{N-1}\frac{h}{6}\left[f(x_k)+4f(x_k+\frac{h}{2})+f(x_{k+1})\right]$$

$$I=\frac{h}{6}\left[f(1)+2\sum_{k=1}^{N-1}f(x_k)+4\sum_{k=0}^{N-1}f(x_k+\frac{h}{2})+f(\pi)\right]$$

具体的，当 X 比较小时，比如 1 时，即在计算 $\int_1^\pi\frac{1}{t}dt$ 的近似值；

当 X 比较大时，比如 X=10，即在计算 $\int_1^{\pi^{10}}\frac{1}{t}dt$ 的近似值，而 $\pi^{10}\approx93648.047476$，可见积分区间非常之大，这便会导致**舍入误差**累计较大，下面将会具体分析。

### 2.4.3.2 误差分析

根据求解公式，可见误差分为下面的四个部分：

#### A、牛顿法误差

牛顿法二阶收敛，$\varphi''(x)=-\frac{1}{x}$，$x\in\left[\!\left[\lfloor\pi^X\rfloor,\lfloor\pi^X+1\rfloor\right]\!\right]$，则：

$$M = \max\left|\varphi''(x)\right| = \frac{1}{\left\lfloor \pi^{X} \right\rfloor}$$

则第 n 步的局部误差为:

$$e_{n+1} \le \frac{M}{2!} e^{2}{}_{n}$$

迭代得:

$$e_{n+1} \le \frac{M^{n}}{2^{n}} e_{0}{}^{2^{n}} \le \frac{M^{n}}{2^{n}}$$

此处取利普希差常数 L 为 0.5.

**B、Ln(PI)的误差所导致的误差**

整体考虑，即最终计算的跟有误差。则引起的误差为:

$$\beta_1 = \left| \frac{\partial \mathrm{e}^{X \ln PI}}{\partial \ln PI} \right| * \Delta \ln PI = \left| X * e^{X \ln PI} \right| * \Delta \ln PI$$

**C、复化辛普生公式的误差**

复化辛普生公式的误差已在 2.3.3 节当中做了详细分析，这里需要注意的是，积分上限改变了，则导数的上限值也会相应的发生变化。其中:

方法误差:

$$\Delta=\sum_{k=0}^{N-1}-\frac{h^5}{2880}f^{(4)}(\eta_k)=-\frac{(b-a)h^4}{2880}f^{(4)}(\eta)$$

$$f(x)=\frac{1}{x}$$

$$f^{(4)}(x)=\frac{24}{x^5}$$

$$\max(f''(x))\Big|_{x\in[1,\ X\ln PI]}=24$$

进行放缩:

$$\Delta\le\frac{pow(PI,X)_{上限}*24}{2880*n^4}$$

此处，为了处理方便，统一辛普生同时的积分精确到小数点后12位。

则这一环节的误差:

$$\beta_2=N*\pi^X*0.5*10^{-12}$$

**D、舍入误差**

假设计算机在进行加减乘除运算时不引入误差,仅有在保存数据时会带来舍入误差，那么总的舍入误差:

$$\delta_n\le L*\delta_{n-1}+0.5*10^{-m}$$

$$L=0.5$$

$$\delta_n\le 0.5^n*\delta_0+(1-0.5^n)*10^{-m}$$

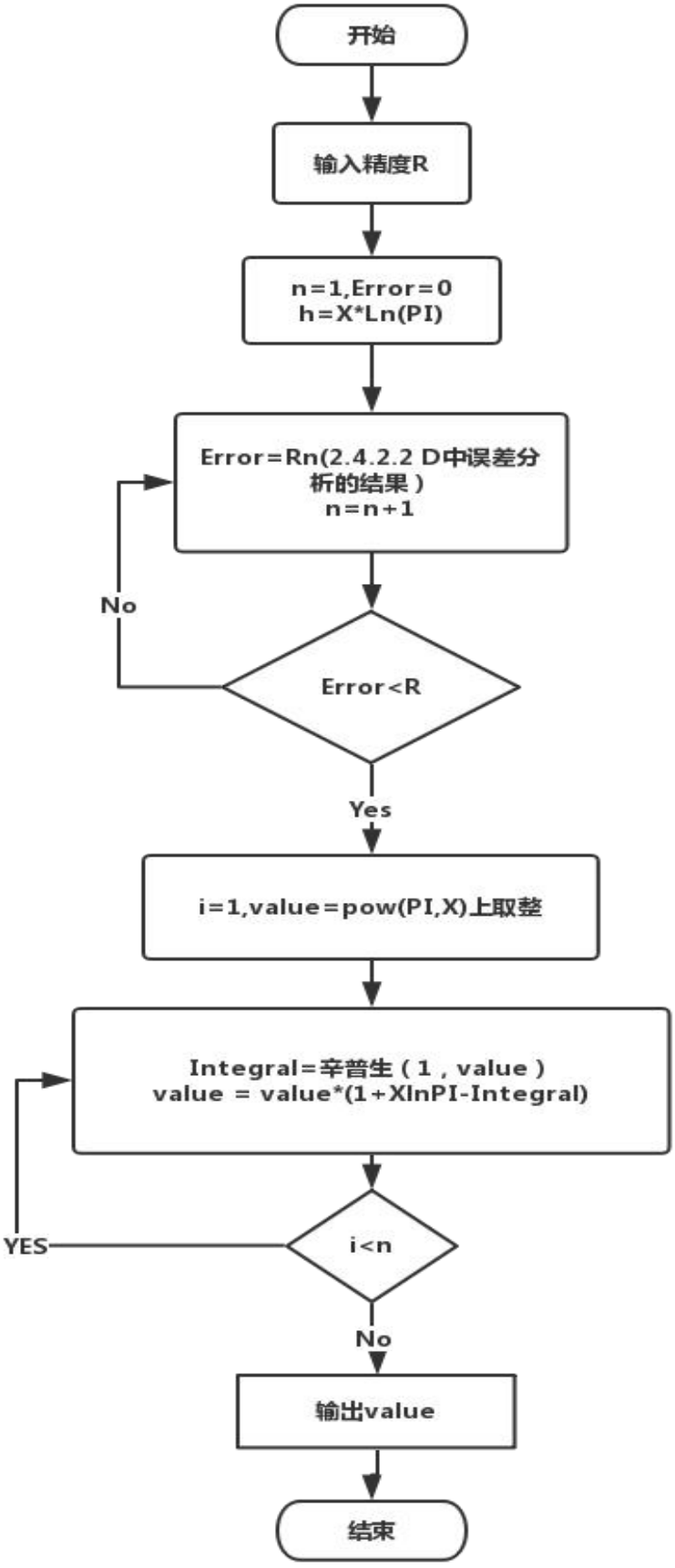
其中，m=18，通过构造等比数列求得总误差。

**E、总误差**

即为上面的误差求和:

$$\mathrm{E}rror = \Delta + \beta_1 + \beta_2 + \delta_n$$

### 2.4.3.3 程序框图


开始
输入精度R
n=1,Error=0
h=X*Ln(PI)
Error=Rn(2.4.2.2 D中误差分析的结果）
n=n+1
No
Error<R
Yes
i=1,value=pow(PI,X)上取整
Integral=辛普生（1，value）
value = value*(1+XlnPI-Integral)
YES
i<n
No
输出value
结束

#### 2.4.3.4 计算代价

对于牛顿迭代公式，每次迭代需要加法 3 次，乘法 1 次，每一步还需进行一次辛普生的积分计算，需加减法 5*N’次，乘法 2*N’次，除法 2*N’次，

#### 2.4.3.5 收敛速度

外层的牛顿积分公式二阶收敛；内层的辛普生公式四阶收敛。整体二阶收敛。

#### 2.4.3.6 实验

基于算法原理的分析，当 X 比较大时，比如 X=10，对于辛普生公式即在计算 $\int_1^{\pi^{10}} \frac{1}{t} dt$ 的近似值，而 $\pi^{10} \approx 93648.047476$，积分区间非常之大，这导致**舍入误差**累计过大，已经达不到精度要求，此时便不再适合。

经过试验，可得当 X~[1,4]时，这种方法行之有效。X 再大，便不再适合。

经过试验可得，当 X=4 是，精确到小数点后时，需要迭代 4 次，这也和初始点的选取有关。但由于期间包含 4 次辛普生求解过程，因此这样的方法的整体计算量要大于泰勒展开和改进的欧拉法。

### 2.4.4 方法对比

**X=10**

| 方法 | 迭代次数 | 计算代价（加法+乘法+除法） |
|---|---|---|

| 泰勒展开 | 54 | 108+54+54 |
|---|---|---|
| 常微分方程求解之改进的欧拉方法 | 33554432 | 4*33554432+2*33554432+33554432 |

X=4

| 方法 | 迭代次数 | 计算代价（加法+乘法+除法） |
|---|---|---|
| 泰勒展开 | 28 | 56+28+28 |
| 常微分方程求解之改进的欧拉方法 | 262144 | 4*262144+2*262144+262144 |
| 牛顿法方程求根 | 外层：4<br>内层辛普生：<br>132838+131421+131425+131425 | 外层：3*4+1*4+0*4<br>辛普生：5*N’+2*N’+2*N’ |

## 3. 收获反思

本次大作业较上次大作业虽然任务看似少了，但是要是真正地把能用的方法都进行尝试和实现，花费的精力丝毫不比第一次大作业少。在这次大作业当中，我尝试和实现了多种方法，每一个值都有三种实现办法。其中涉及到了【泰勒展开】【复化梯形公式求积分】【辛普生公式求积分】【改进的欧拉法常微分方程】【牛顿法求方程的根】

等数值接法，对所学的方法都有所设计。通过这次大作业，不仅对原本的公式有了更加深刻和具体的认识，同时也对数值分析的所有应用做了系统复习。 深刻认识到，在使用计算机对数值问题进行求解时，误差分析的重要性以及实际问题当中理想值与计算机所得值之间的关系。

## 4、参考文献

[1] 李庆扬、王能超、易大义. 数值分析（第五版）

[2] 周老师讲义及补充资料