《计算物理》第 2 次作业

1900017812 高乐耘*

2022 年 10 月 11 日

目录

I. 线性方程组求解 **1**

II. 三次样条插值 **2**

A. 边界条件为 $S''(x_0) = S''(x_2) = 0$ 3

B. 边界条件为 $S'(x_0) = f'(x_0)$ 和 $S'(x_2) = f'(x_2)$ 4

III. 切比雪夫近似 **6**

IV. 龙格现象 **8**

A. 龙格现象 8

B. 龙格函数的 20 点切比雪夫展开 10

C. 龙格函数的 21 点三次样条插值 12

V. 三次样条函数与计算机绘图 **14**

A. 九个采样点精确值 14

B. 参数方程九点三次样条插值 15

C. 绘图 16

D. 平滑连接之源 16

A 项目源代码 **16**

I. 线性方程组求解

笔者在第 1 次作业中已编写了高斯消元法和楚列斯基分解法求解线性方程组的代码并进行了单元测试，它们分别位于源文件 hilbert/gem.c 和 hilbert/cholesky.c 中。笔者在本题工作目录 lin/ 下直接拷贝了它们的副本，并在源文件 lin/lin.c 中实现了问题求解流程。在工作目录下执行 `make run` 命令获得本题解答文件 lin/lin.out，其内容如下：

*电子邮件地址: seeson@pku.edu.cn 手机号: 13759115414

```
gem:
      0.07          0.1         0.08         0.07            1
      0.05  -0.00571429    0.0314286         0.03            1
      0.06  -0.00142857        0.025       0.0425            1
      0.05  -0.00142857       -0.005        0.001            1
cholesky:
  0.223607      0.31305     0.268328     0.223607         0.23
   0.31305    0.0447214   -0.0894427  -3.10317e-16         0.32
  0.268328   -0.0894427    0.141421     0.212132         0.33
  0.223607  -3.10317e-16    0.212132    0.0707107         0.31
cdm:
  0.223607      0.31305     0.268328     0.223607            1
   0.31305    0.0447214   -0.0894427  -3.10317e-16            1
  0.268328   -0.0894427    0.141421     0.212132            1
  0.223607  -3.10317e-16    0.212132    0.0707107            1
```

输出内容三项 `gem`，`cholesky` 和 `cdm` 分别表示对题中线性方程组的增广矩阵进行高斯消元、楚列斯基分解和楚列斯基求解方程组的结果。其中 `gem` 和 `cdm` 输出的最后一列为方程的解向量，它们一致地等于 $(1,1,1,1)'$，`cholesky` 项下的输出为对该增广矩阵的系数矩阵部分进行楚列斯基分解后将得到的下三角矩阵对角线以下（不含）的部分和上三角矩阵对角线以上（含）的部分叠加在一起（实际上二者的对角线元素是对应相等的），而右端向量部分不变，所得到的结果。取其系数矩阵的上三角部分，立即得到所求的上三角矩阵：

$$H=\begin{pmatrix} 0.223607 & 0.31305 & 0.268328 & 0.223607 \\ 0 & 0.0447214 & -0.0894427 & -3.10317\times 10^{-16} \\ 0 & 0 & 0.141421 & 0.212132 \\ 0 & 0 & 0 & 0.0707107 \end{pmatrix} \tag{1}$$

II. 三次样条插值

设样条函数：

$$S(x)=\begin{cases} S_1(x), & x_0\le x\le x_1, \\ S_2(x), & x_1<x\le x_2. \end{cases} \tag{2}$$

其中 $S_1(x)$ 和 $S_2(x)$ 为三次函数。我们有：

$$\begin{aligned} &S_1(x_0)=f(x_0),\ S_1(x_1)=f(x_1), \\ &S_2(x_1)=f(x_1),\ S_2(x_2)=f(x_2), \\ &S_1'(x_1)=S_2'(x_1),\ S_1''(x_1)=S_2''(x_1), \end{aligned} \tag{3}$$

考虑到 $S_1''(x)$ 和 $S_2''(x)$ 均为一次函数，分别适用于两点式方程：

$$\begin{aligned} S_1''(x)&=S_1''(x_0)\frac{x-x_1}{x_0-x_1}+S_1''(x_1)\frac{x-x_0}{x_1-x_0}=S''(x_0)\frac{x-x_1}{x_0-x_1}+S''(x_1)\frac{x-x_0}{x_1-x_0} \\ S_2''(x)&=S_2''(x_1)\frac{x-x_2}{x_1-x_2}+S_2''(x_2)\frac{x-x_1}{x_2-x_1}=S''(x_1)\frac{x-x_2}{x_1-x_2}+S''(x_2)\frac{x-x_1}{x_2-x_1} \end{aligned} \tag{4}$$

对其积分一次得到：

$$\begin{aligned} S_1'(x) &= S''(x_0)\frac{(x-x_1)^2}{2(x_0-x_1)} + S''(x_1)\frac{(x-x_0)^2}{2(x_1-x_0)} + C_1 \\ S_2'(x) &= S''(x_1)\frac{(x-x_2)^2}{2(x_1-x_2)} + S''(x_2)\frac{(x-x_1)^2}{2(x_2-x_1)} + C_2 \end{aligned} \tag{5}$$

代入 $S_1'(x_1) = S_2'(x_1)$ 立即得到：

$$S''(x_1)\frac{x_1-x_0}{2} + C_1 = S''(x_1)\frac{x_1-x_2}{2} + C_2 \;\Rightarrow\; C_1 - C_2 = S''(x_1)\frac{x_0-x_2}{2} \tag{6}$$

对式 (5) 再积分一次得到：

$$\begin{aligned} S_1(x) &= S''(x_0)\frac{(x-x_1)^3}{6(x_0-x_1)} + S''(x_1)\frac{(x-x_0)^3}{6(x_1-x_0)} + C_1x + D_1 \\ S_2(x) &= S''(x_1)\frac{(x-x_2)^3}{6(x_1-x_2)} + S''(x_2)\frac{(x-x_1)^3}{6(x_2-x_1)} + C_2x + D_2 \end{aligned} \tag{7}$$

代入式 (3) 前四式有：

$$\begin{aligned} f(x_0) = S_1(x_0) &= S''(x_0)\frac{(x_0-x_1)^2}{6} + C_1x_0 + D_1 \\ f(x_1) = S_1(x_1) &= S''(x_1)\frac{(x_1-x_0)^2}{6} + C_1x_1 + D_1 \\ f(x_1) = S_2(x_1) &= S''(x_1)\frac{(x_1-x_2)^2}{6} + C_2x_1 + D_2 \\ f(x_2) = S_2(x_2) &= S''(x_2)\frac{(x_2-x_1)^2}{6} + C_2x_2 + D_2 \end{aligned} \tag{8}$$

消去 D_1 和 D_2 得：

$$\begin{aligned} f(x_0) - f(x_1) &= (S''(x_0) - S''(x_1))\frac{(x_0-x_1)^2}{6} + C_1(x_0-x_1) \\ f(x_1) - f(x_2) &= (S''(x_1) - S''(x_2))\frac{(x_1-x_2)^2}{6} + C_2(x_1-x_2) \end{aligned} \tag{9}$$

式 (6) 和式 (9) 共三个方程，五个未知量，添加两个边界条件后可以顺利求解。

A. 边界条件为 $S''(x_0) = S''(x_2) = 0$

在此边界条件下，式 (9) 简化为：

$$\begin{aligned} f(x_0) - f(x_1) &= -S''(x_1)\frac{(x_0-x_1)^2}{6} + C_1(x_0-x_1) \\ f(x_1) - f(x_2) &= +S''(x_1)\frac{(x_1-x_2)^2}{6} + C_2(x_1-x_2) \end{aligned} \tag{10}$$

变形得：

$$\begin{aligned} C_1 &= \frac{f(x_0)-f(x_1)}{x_0-x_1} + S''(x_1)\frac{x_0-x_1}{6} \\ C_2 &= \frac{f(x_1)-f(x_2)}{x_1-x_2} - S''(x_1)\frac{x_1-x_2}{6} \end{aligned} \tag{11}$$

代入式 (6) 即得：

$$\frac{f(x_0)-f(x_1)}{x_0-x_1}+S''(x_1)\frac{x_0-x_1}{6}-\frac{f(x_1)-f(x_2)}{x_1-x_2}+S''(x_1)\frac{x_1-x_2}{6}=S''(x_1)\frac{x_0-x_2}{2} \tag{12}$$

整理得：

$$S''(x_1)=\frac{3}{x_0-x_2}\left(\frac{f(x_0)-f(x_1)}{x_0-x_1}-\frac{f(x_1)-f(x_2)}{x_1-x_2}\right) \tag{13}$$

定义 0 阶和 1 阶向后差分：

$$\Delta^{(0)}x_i=x_i-x_{i-1},\ \Delta^{(1)}f_i=\frac{f(x_{i-1})-f(x_i)}{x_{i-1}-x_i},\ i=1,2 \tag{14}$$

我们得到了以下结论：

$$\begin{aligned} S''(x_1)&=\frac{3}{\Delta^{(0)}x_1+\Delta^{(0)}x_2}(\Delta^{(1)}f_2-\Delta^{(1)}f_1)\\ C_1&=\Delta^{(1)}f_1-S''(x_1)\frac{\Delta^{(0)}x_1}{6}\\ C_2&=\Delta^{(1)}f_2+S''(x_1)\frac{\Delta^{(0)}x_2}{6} \end{aligned} \tag{15}$$

同时，考虑式 (8)，我们得到：

$$\begin{aligned} D_1&=f(x_0)-C_1x_0\\ D_2&=f(x_2)-C_2x_2 \end{aligned} \tag{16}$$

至此，我们给出了无关于 $f(x)$ 具体形式的样条函数 $S(x)$ 包含的全部系数的解析表达式。

对于题中所给的数值，使用程序 spl/spl-1 直接求值得到：

```
Sppx1 = -2.3816e+00
C1 = 1.3132e-01
C2 = -9.4041e-01
D1 = 1.0000e+00
D2 = 1.5359e+00
S1(x) = -6.6156e-01*(x - 0.0)^3 + 1.3132e-01*x + 1.0000e+00
S2(x) = 1.3231e+00*(x - 0.9)^3 + -9.4041e-01*x + 1.5359e+00
```

样条函数 $S(x)$ 即为：

$$S(x)=\begin{cases}-0.66156x^3+0.13132x+1.0000, & 0\le x\le 0.6,\\ 1.3231(x-0.9)^3-0.94041x+1.5359, & 0.6<x\le 0.9.\end{cases} \tag{17}$$

B. 边界条件为 $S'(x_0)=f'(x_0)$ 和 $S'(x_2)=f'(x_2)$

由式 (5) 得：

$$\begin{aligned} f'(x_0)&=S_1'(x_0)=S''(x_0)\frac{x_0-x_1}{2}+C_1\\ f'(x_2)&=S_2'(x_2)=S''(x_2)\frac{x_2-x_1}{2}+C_2 \end{aligned} \tag{18}$$

代入式 (6) 得：

$$f'(x_0) - S''(x_0)\frac{x_0 - x_1}{2} - f'(x_2) + S''(x_2)\frac{x_2 - x_1}{2} = S''(x_1)\frac{x_0 - x_2}{2} \tag{19}$$

整理得：

$$S''(x_0)\frac{x_1 - x_0}{2} + S''(x_1)\frac{x_2 - x_0}{2} + S''(x_2)\frac{x_2 - x_1}{2} = f'(x_2) - f'(x_0) \tag{20}$$

将式 (18) 代入式 (9) 得：

$$\begin{aligned}
\frac{f(x_0) - f(x_1)}{x_0 - x_1} &= (S''(x_0) - S''(x_1))\frac{x_0 - x_1}{6} + f'(x_0) - S''(x_0)\frac{x_0 - x_1}{2} \\
\frac{f(x_1) - f(x_2)}{x_1 - x_2} &= (S''(x_1) - S''(x_2))\frac{x_1 - x_2}{6} + f'(x_2) - S''(x_2)\frac{x_2 - x_1}{2}
\end{aligned} \tag{21}$$

整理得：

$$\begin{aligned}
S''(x_0)\frac{x_0 - x_1}{3} + S''(x_1)\frac{x_0 - x_1}{6} &= f'(x_0) - \frac{f(x_0) - f(x_1)}{x_0 - x_1} \\
S''(x_1)\frac{x_1 - x_2}{6} + S''(x_2)\frac{x_1 - x_2}{3} &= \frac{f(x_1) - f(x_2)}{x_1 - x_2} - f'(x_2)
\end{aligned} \tag{22}$$

两式相加得：

$$S''(x_0)\frac{x_0 - x_1}{3} + S''(x_1)\frac{x_0 - x_2}{6} + S''(x_2)\frac{x_1 - x_2}{3} = f'(x_0) - \Delta^{(1)}f_1 + \Delta^{(1)}f_2 - f'(x_2) \tag{23}$$

加上式 (20) 的 2/3 倍得：

$$S''(x_1)\frac{x_2 - x_0}{6} = \Delta^{(1)}f_2 - \Delta^{(1)}f_1 - \frac{1}{3}(f'(x_2) - f'(x_0)) \tag{24}$$

由此立即得到：

$$S''(x_1) = \frac{6(\Delta^{(1)}f_2 - \Delta^{(1)}f_1) - 2(f'(x_2) - f'(x_0))}{\Delta^{(0)}x_1 + \Delta^{(0)}x_2} \tag{25}$$

再由式 (22) 解出：

$$\begin{aligned}
S''(x_0) &= \frac{3}{\Delta^{(0)}x_1}(\Delta^{(1)}f_1 - f'(x_0)) - \frac{1}{2}S''(x_1) \\
S''(x_2) &= \frac{3}{\Delta^{(0)}x_2}(f'(x_2) - \Delta^{(1)}f_2) - \frac{1}{2}S''(x_1)
\end{aligned} \tag{26}$$

对于 C_1 和 C_2，可以使用式 (18) 求出：

$$\begin{aligned}
C_1 &= f'(x_0) + S''(x_0)\frac{\Delta^{(0)}x_1}{2} \\
C_2 &= f'(x_2) - S''(x_2)\frac{\Delta^{(0)}x_2}{2}
\end{aligned} \tag{27}$$

最后，我们再由式 (8) 求出 D_1 和 D_2：

$$\begin{aligned}
D_1 &= f(x_0) - S''(x_0)\frac{(\Delta^{(0)}x_1)^2}{6} - C_1x_0 \\
D_2 &= f(x_2) - S''(x_2)\frac{(\Delta^{(0)}x_2)^2}{6} - C_2x_2
\end{aligned} \tag{28}$$

至此，我们给出了无关于 $f(x)$ 具体形式的样条函数 $S(x)$ 包含的全部系数的解析表达式。

对于题中所给的函数 $f(x) = \cos x^2$，其导数为 $f'(x) = -2x\sin x^2$。对于题中所给的数值，使用程序 spl/spl-2 直接求值得到：

```
Sppx0 = 3.9886e-01
Sppx1 = -1.8661e+00
Sppx2 = -3.8908e+00
C1 = 1.1966e-01
C2 = -7.2009e-01
D1 = 9.7607e-01
D2 = 1.3959e+00
S1(x) = -1.1080e-01*(x - 0.6)^3
    + -5.1836e-01*(x - 0.0)^3 + 1.1966e-01*x + 9.7607e-01
S2(x) = 1.0367e+00*(x - 0.9)^3
    + -2.1616e+00*(x - 0.6)^3 + -7.2009e-01*x + 1.3959e+00
```

样条函数 $S(x)$ 即为：

$$S(x)=\begin{cases}-0.11080(x-0.6)^3-0.51836x^3+0.11966x+0.97607, & 0\le x\le 0.6,\\ 1.0367(x-0.9)^3-2.1616(x-0.6)^3-0.72009x+1.3959, & 0.6<x\le 0.9.\end{cases} \tag{29}$$

III. 切比雪夫近似

本题需要使用多项式计算。笔者在头文件 che/defs.h 中定义了多项式结构体类型 `poly_t`（che/defs.h#L8）并声明了一组操作多项式的函数（che/defs.h#L14），它们定义在源文件 che/poly.c 中，其测试代码为 che/poly-test.c。

笔者编写了一组函数分别实现了 0 至 n 阶切比雪夫多项式的生成、切比雪夫近似系数 c_m，$m\in\{0,1,\ldots,n\}$ 的计算和给定各 c_m 时目标函数 $f(x)$ 的直至 n 阶切比雪夫近似 $S_n(x)$ 对于给定自变量 x 的求值功能。它们的原型位于头文件 che/defs.h#L34，实现位于源文件 che/chebyshev.c 中。笔者在程序 che/chebyshev-test.c 中对切比雪夫多项式的生成方法 `che_gen()`，切比雪夫近似的求值方法 `che_eval()` 进行了测试，而切比雪夫近似系数的计算方法 `che_approx()` 则留给本题所求解的具体问题进行测试。

`che_gen()` 方法（che/chebyshev.c#L4）使用如下递推关系生成不小于 n 阶的各切比雪夫多项式：

$$T_0(x):=1;\quad T_1(x):=x;\quad T_i(x):=2xT_{i-1}(x)-T_{i-2}(x),\ \forall i:2\le i\le n \tag{30}$$

`che_eval()` 方法（che/chebyshev.c#L71）使用相同的递推关系进行求值，避免了以多项式幂次为基进行展开，从而保持了数值稳定性。

`che_approx()` 方法（che/chebyshev.c#L40）使用了如下公式计算各近似系数 c_m：

$$c_m=\frac{2-\delta_{0i}}{n+1}\sum_{i=0}^{n}T_m(x_{n+1,i})f(x_{n+1,i}) \tag{31}$$

其中 $x_{n+1,i}=(i+1/2)\pi/(n+1)$ 表示 $T_{n+1}(x)$ 的第 i 个零点。`che_approx()` 函数只计算一次各 $x_{n+1,i}$ 和 $f(x_{n+1,i})$，将计算结果分别保存到数组 `x`（che/chebyshev.c#L44）和 `y`（che/chebyshev.c#L45）中复用，并使用已经计算好的切比雪夫多项式数组 `T`（che/chebyshev.c#L40）分别代入各 $x_{n+1,i}$ 求值 $T_m(x_{n+1,i})$，从而最小化了计算量。

笔者在源文件 che/che-log2.c 中实现了本题的求解过程。其中用于切比雪夫近似的目标函数定义如下：

```c
/*- che/che-log2.c -*/
#define number double

number f(number x)
{
  return log(x * 0.5 + 1.5) / log(2);
}
```

这里我们引入了综量 $z = 0.5x+1.5$，由于切比雪夫近似要求 $x \in [-1, 1]$，我们有 $z \in [1, 2]$，函数 `f()` 计算的就是 $\log_2 z$ 的结果。下文中的统计图我们对自变量使用了变换 $x := 0.5x+1.5$，从而能够反映出 $\log_2 x$ 在 $x \in [1, 2]$ 上切比雪夫近似的结果。

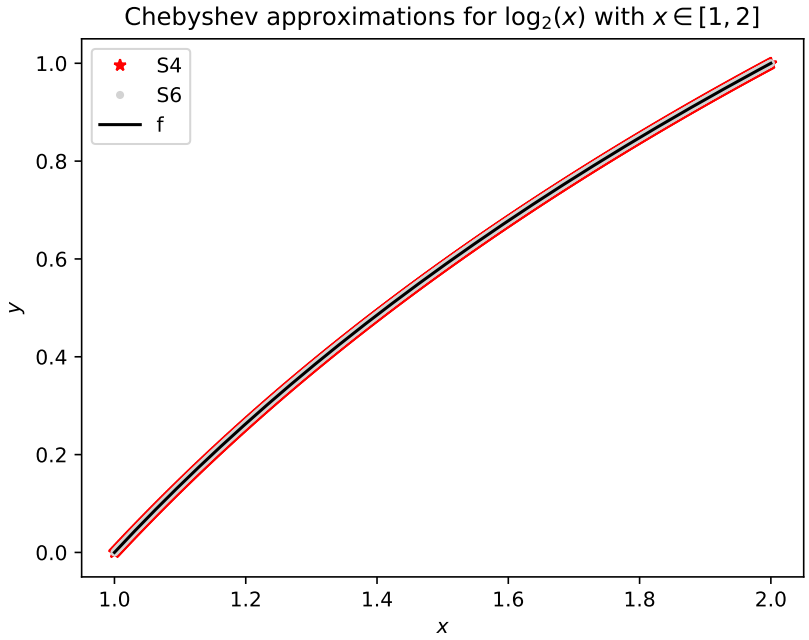

图 1: 直至 4 阶（S4）和 6 阶（S6）切比雪夫近似对 $\log_2 x$（f）在 $x \in [1, 2]$ 上的结果

图 1 和 2 分别给出了直至 4 阶和 6 阶切比雪夫近似对 $\log_2 x$ 在 $x \in [1, 2]$ 上的结果和误差。前者显示，真实函数图像与两种近似图像完全重合，这验证了我们可以由较小的 n 获得极好的近似效果，显示了切比雪夫近似的强大威力。后者与前者一致地显示，即使是 $n = 4$ 时，切比雪夫近似的误差仍然非常小，且在零点附近上下波动，而 $n = 6$ 时则明显更为精确。

此外，让我们看一下程序给出的各展开系数 c_m（为方便阅读已对输出 che/che-log2.log 中的空白字符进行了重排）：

```
4:
5.43106612705316683e-01  4.95054630259919359e-01  -4.24687057987235272e-02
4.85587890340776623e-03  -6.12817782883818413e-04
6:
5.43106606331171693e-01  4.95054672534052942e-01  -4.24689766328674029e-02
4.85768197639158648e-03  -6.25078597739142086e-04  8.57567965448542154e-05
-1.19963548550064716e-05
```

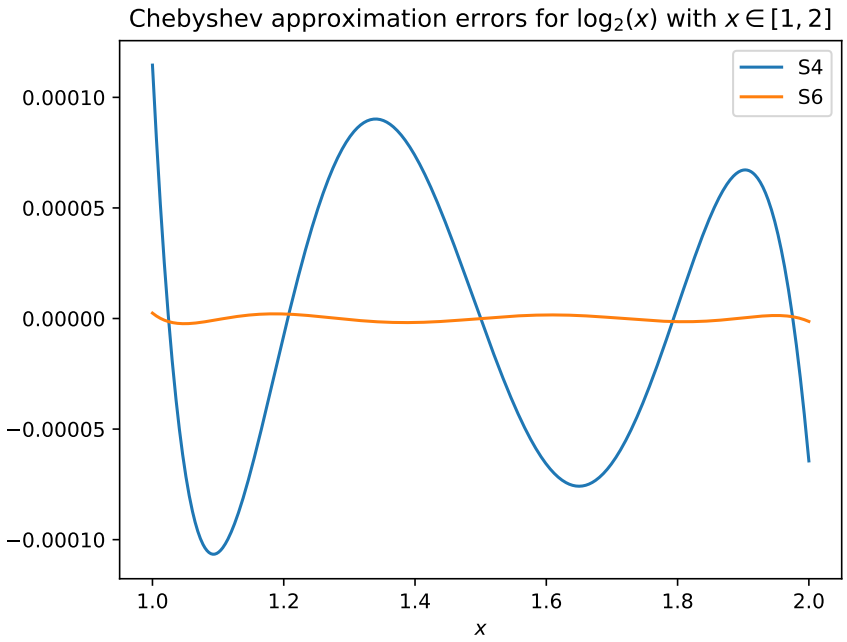

图 2: 直至 4 阶（S4）和 6 阶（S6）切比雪夫近似对 $\log_2 x$ 在 $x \in [1, 2]$ 上的误差

首先我们看到，对于 $n = 4$ 和 $n = 6$ 的前 5 个系数而言，它们整体上非常接近，这验证了我们按照式 (31) 而非积分表达式计算各 c_m 的合理性。更为重要的是，$n = 6$ 的第 6 个系数，$c_5 \sim 10^{-4}$，加上切比雪夫多项式在 $[-1, 1]$ 上绝对值不超过 1 的重要性质，这解释了按照 $n = 4$ 近似时误差 $\sim 10^{-4}$（从图 2 中可见）的原因。此外，可以预测 $c_7 \sim 10^{-6}$–10^{-5}，这与图 2 中所看到的 $n = 6$ 的误差 $\sim 10^{-5}$ 是一致的。从以上分析中我们再一次看到，正是切比雪夫多项式在其定义域内绝对值不超过 1 的性质，保证了切比雪夫展开的精确性和有效性。题中所给的函数在我们刚才研究的区间内是无限光滑的，所以 c_m 随 m 的增大绝对值便可快速衰减。我们验证了这样的结论：对于光滑性较好的函数，可以使用不大的 n，便获得极好的切比雪夫近似。

IV. 龙格现象

A. 龙格现象

本题需要实现多项式插值。为方便日后使用，笔者在项目源文件中一次性编写了拉格朗日插值（runge/intp.c#L3）、牛顿插值（runge/intp.c#L34）和 Neville 插值（runge/intp.c#L68）函数，它们拥有如下统一的原型：

```
int intp_l(poly_t *P, int n, const number x[n + 1], const number y[n + 1]);
int intp_n(poly_t *P, int n, const number x[n + 1], const number y[n + 1]);
int intp_v(poly_t *P, int n, const number x[n + 1], const number y[n + 1]);
```

其测试代码位于 runge/intp-test.c 中。

本小题解决方案实现于 runge/runge.c 中。经过比较，发现 Neville 插值在支撑点处的总体误差最小，故选用 Neville 插值进行后续实验。运行程序，标准输出和标准错误流中将

写入作图程序 runge/runge-plot 所需的全部数据，其中标准错误流中包含了本题所求的表格（前两列分别为龙格函数和对称等间距二十一点多项式插值函数在同一给定自变量下的函数值，最后一列给出了二者之差的绝对值）：

```
runge(−1.00) = 0.0384615    S20(−1.00) = 0.0384615     diff = 2.55218e−10
runge(−0.95) = 0.0424403    S20(−0.95) = −39.9524      diff = 39.9949
runge(−0.90) = 0.0470588    S20(−0.90) = 0.0470588     diff = 1.05544e−10
runge(−0.85) = 0.052459     S20(−0.85) = 3.45496       diff = 3.4025
runge(−0.80) = 0.0588235    S20(−0.80) = 0.0588235     diff = 7.78419e−12
runge(−0.75) = 0.06639      S20(−0.75) = −0.447052     diff = 0.513442
runge(−0.70) = 0.0754717    S20(−0.70) = 0.0754717     diff = 2.86909e−13
runge(−0.65) = 0.0864865    S20(−0.65) = 0.202423      diff = 0.115936
runge(−0.60) = 0.1          S20(−0.60) = 0.1           diff = 3.14776e−13
runge(−0.55) = 0.116788     S20(−0.55) = 0.08066       diff = 0.0361283
runge(−0.50) = 0.137931     S20(−0.50) = 0.137931      diff = 3.78864e−14
runge(−0.45) = 0.164948     S20(−0.45) = 0.179763      diff = 0.0148142
runge(−0.40) = 0.2          S20(−0.40) = 0.2           diff = 8.32667e−17
runge(−0.35) = 0.246154     S20(−0.35) = 0.238446      diff = 0.00770791
runge(−0.30) = 0.307692     S20(−0.30) = 0.307692      diff = 0
runge(−0.25) = 0.390244     S20(−0.25) = 0.395093      diff = 0.00484915
runge(−0.20) = 0.5          S20(−0.20) = 0.5           diff = 0
runge(−0.15) = 0.64         S20(−0.15) = 0.636755      diff = 0.00324466
runge(−0.10) = 0.8          S20(−0.10) = 0.8           diff = 2.22045e−16
runge(−0.05) = 0.941176     S20(−0.05) = 0.94249       diff = 0.00131391
runge(+0.00) = 1            S20(+0.00) = 1             diff = 4.44089e−16
runge(+0.05) = 0.941176     S20(+0.05) = 0.94249       diff = 0.00131391
runge(+0.10) = 0.8          S20(+0.10) = 0.8           diff = 3.33067e−16
runge(+0.15) = 0.64         S20(+0.15) = 0.636755      diff = 0.00324466
runge(+0.20) = 0.5          S20(+0.20) = 0.5           diff = 6.10623e−16
runge(+0.25) = 0.390244     S20(+0.25) = 0.395093      diff = 0.00484915
runge(+0.30) = 0.307692     S20(+0.30) = 0.307692      diff = 1.55431e−15
runge(+0.35) = 0.246154     S20(+0.35) = 0.238446      diff = 0.00770791
runge(+0.40) = 0.2          S20(+0.40) = 0.2           diff = 4.82947e−15
runge(+0.45) = 0.164948     S20(+0.45) = 0.179763      diff = 0.0148142
runge(+0.50) = 0.137931     S20(+0.50) = 0.137931      diff = 1.60705e−14
runge(+0.55) = 0.116788     S20(+0.55) = 0.08066       diff = 0.0361283
runge(+0.60) = 0.1          S20(+0.60) = 0.1           diff = 1.20459e−13
runge(+0.65) = 0.0864865    S20(+0.65) = 0.202423      diff = 0.115936
runge(+0.70) = 0.0754717    S20(+0.70) = 0.0754717     diff = 8.49959e−13
runge(+0.75) = 0.06639      S20(+0.75) = −0.447052     diff = 0.513442
runge(+0.80) = 0.0588235    S20(+0.80) = 0.0588235     diff = 4.03924e−12
runge(+0.85) = 0.052459     S20(+0.85) = 3.45496       diff = 3.4025
runge(+0.90) = 0.0470588    S20(+0.90) = 0.0470588     diff = 6.54408e−11
runge(+0.95) = 0.0424403    S20(+0.95) = −39.9524      diff = 39.9949
runge(+1.00) = 0.0384615    S20(+1.00) = 0.0384615     diff = 2.10443e−10
```

图 3 展示了龙格函数二十一点对称等间距多项式内插形成的龙格现象。考虑到比例尺问题，纵坐标使用了对数单位。从图中清楚地看出，各采样点（支撑点）处插值函数与龙格函数精确符合；自变量值在 0 附近时，插值函数也和龙格函数符合较好；但在自变量绝对值接

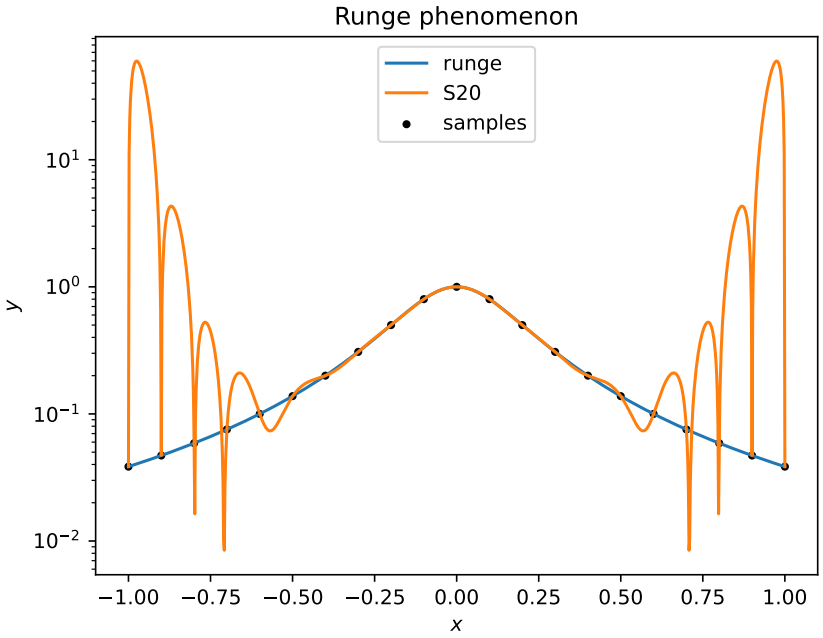

图 3: 龙格现象

近 1 且远离支撑点时，插值函数出现剧烈振荡，并显著偏离龙格函数。从中我们看到了龙格现象的各个典型特征。

B. 龙格函数的 20 点切比雪夫展开

将 runge/runge.c 中的插值函数修改为切比雪夫展开，得到程序 runge/che.c。运行之，下表写入标准错误：

```
runge(−1.000) = 0.0384615    S19(−1.000) = 0.0370158    diff = 0.00144578
runge(−0.997) = 0.0386906    S19(−0.997) = 0.0386906    diff = 1.87302e−13
runge(−0.988) = 0.0393884    S19(−0.988) = 0.040869     diff = 0.00148062
runge(−0.972) = 0.0405884    S19(−0.972) = 0.0405884    diff = 5.75401e−13
runge(−0.951) = 0.0423501    S19(−0.951) = 0.0407581    diff = 0.00159195
runge(−0.924) = 0.0447651    S19(−0.924) = 0.0447651    diff = 9.81229e−13
runge(−0.891) = 0.0479678    S19(−0.891) = 0.0497709    diff = 0.00180313
runge(−0.853) = 0.0521516    S19(−0.853) = 0.0521516    diff = 1.36145e−12
runge(−0.809) = 0.0575947    S19(−0.809) = 0.0554297    diff = 0.002165
runge(−0.760) = 0.0647022    S19(−0.760) = 0.0647022    diff = 1.64226e−12
runge(−0.707) = 0.0740741    S19(−0.707) = 0.0768585    diff = 0.00278447
runge(−0.649) = 0.0866208    S19(−0.649) = 0.0866208    diff = 1.75668e−12
runge(−0.588) = 0.103764     S19(−0.588) = 0.0998631    diff = 0.00390051
runge(−0.522) = 0.127794     S19(−0.522) = 0.127794     diff = 1.66281e−12
runge(−0.454) = 0.162531     S19(−0.454) = 0.16864      diff = 0.00610958
runge(−0.383) = 0.214539     S19(−0.383) = 0.214539     diff = 1.3565e−12
runge(−0.309) = 0.295221     S19(−0.309) = 0.284124     diff = 0.0110975
runge(−0.233) = 0.423295     S19(−0.233) = 0.423295     diff = 8.75855e−13
runge(−0.156) = 0.620427     S19(−0.156) = 0.643749     diff = 0.023322
runge(−0.078) = 0.866629     S19(−0.078) = 0.866629     diff = 2.90878e−13
```

runge(+0.000) = 1	S19(+0.000) = 0.96241	diff = 0.0375903
runge(+0.078) = 0.866629	S19(+0.078) = 0.866629	diff = 3.26073e−13
runge(+0.156) = 0.620427	S19(+0.156) = 0.643749	diff = 0.023322
runge(+0.233) = 0.423295	S19(+0.233) = 0.423295	diff = 9.05331e−13
runge(+0.309) = 0.295221	S19(+0.309) = 0.284124	diff = 0.0110975
runge(+0.383) = 0.214539	S19(+0.383) = 0.214539	diff = 1.37859e−12
runge(+0.454) = 0.162531	S19(+0.454) = 0.16864	diff = 0.00610958
runge(+0.522) = 0.127794	S19(+0.522) = 0.127794	diff = 1.67891e−12
runge(+0.588) = 0.103764	S19(+0.588) = 0.0998631	diff = 0.00390051
runge(+0.649) = 0.0866208	S19(+0.649) = 0.0866208	diff = 1.76908e−12
runge(+0.707) = 0.0740741	S19(+0.707) = 0.0768585	diff = 0.00278447
runge(+0.760) = 0.0647022	S19(+0.760) = 0.0647022	diff = 1.65336e−12
runge(+0.809) = 0.0575947	S19(+0.809) = 0.0554297	diff = 0.002165
runge(+0.853) = 0.0521516	S19(+0.853) = 0.0521516	diff = 1.37289e−12
runge(+0.891) = 0.0479678	S19(+0.891) = 0.0497709	diff = 0.00180313
runge(+0.924) = 0.0447651	S19(+0.924) = 0.0447651	diff = 9.93081e−13
runge(+0.951) = 0.0423501	S19(+0.951) = 0.0407581	diff = 0.00159195
runge(+0.972) = 0.0405884	S19(+0.972) = 0.0405884	diff = 5.8524e−13
runge(+0.988) = 0.0393884	S19(+0.988) = 0.040869	diff = 0.00148062
runge(+0.997) = 0.0386906	S19(+0.997) = 0.0386906	diff = 1.91278e−13
runge(+1.000) = 0.0384615	S19(+1.000) = 0.0370158	diff = 0.00144578

上表共 41 行自变量按照 $\cos((1-i/40)\pi)$, $i=0,1,\ldots,40$ 选取。显然，$i=1,3,\ldots,39$ 时，我们分别得到了 $T_{20}(x)$ 的 20 个零点，它们为本小题切比雪夫展开中所使用的节点，解析上逼进函数在此偏差应该为 0。上表给出的数值结果与之相符。另外，从上表中，我们看到非节点处切比雪夫近似的偏差也大都非常小，图 4 将更为清晰地显示出这一事实。

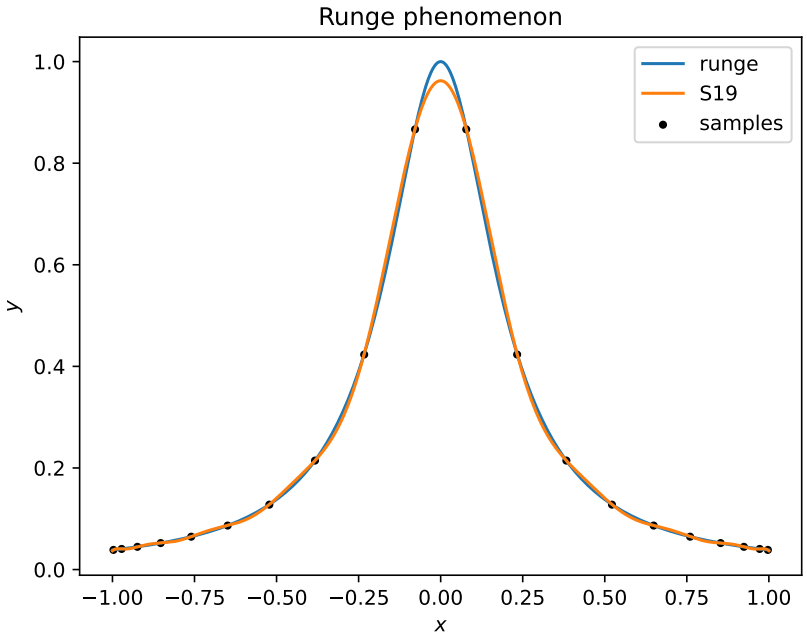

图 4: 龙格函数的 20 点切比雪夫近似结果

观察图 4 不难发现，20 点切比雪夫近似在龙格函数光滑性较好的范围内，即自变量远离

0 的部分，切比雪夫近似函数整体上的近似效果都非常好，特别是自变量绝对值接近 1 的部分，这正是 21 点对称等间距多项式插值表现最差的地方。特别地，切比雪夫近似在整个区间没有出现绝对值很大的近似值，也没有表现出剧烈振荡，对于远离 0 的自变量区间函数值的预测是可靠的。

就龙格函数这个具体的例子而言，切比雪夫近似的整体效果要比多项式插值好得多，这与切比雪夫展开善于对一般光滑函数进行整体近似的普遍规律是一致的。但同时我们也注意到，在龙格函数曲率较大的部分，即自变量接近 0 的部分，切比雪夫近似出现了较为明显的偏离，有趣的是，这正是 20 阶多项式近似处理得较好的部分。这个例子也启发我们，对于不同感兴趣的研究对象，这里即龙格函数不同的自变量区间，我们应当选择不同的近似方法进行描述，以求扬长避短。

C. 龙格函数的 21 点三次样条插值

本题需要求解三对角方程。笔者在源文件 runge/tdm.c 中实现了利用 Thomas 追赶法求解三对角方程的函数 `tdm_tri()`（runge/tdm.c#L35）：

```
int tdm_tri(int n, int m, number d[n], number l[n], number u[n], number r[n][m]);
```

其中 `n`，`m`，`d`，`l`，`u`，`r` 分别表示矩阵行数，右端矩阵列数（$m = 1$ 时退化为右端向量），对角线处元素，对角线下方元素，对角线上方元素和右端矩阵，其中 `l[0]` 和 `u[n - 1]` 没有意义，函数 `tdm_tri()` 不使用它们。测试代码 runge/tdm-test.c 中使用两个实例测试了这一函数，确保函数功能符合预期。

笔者在源文件 runge/cubic.c 中实现了在两端点二阶导数值给定这一边界条件下的三次样条插值函数 `cub_2pp()`（runge/cubic.c#L3）：

```
int cub_2pp(int n, number c[n][4],
    const number x[n + 1], const number y[n + 1], number ypp0, number yppn);
```

其中 `n`，`c`，`x`，`y`，`ypp0`，`yppn` 分别表示函数分段数，每段三次函数的参数，各采样点的横坐标，各采样点的纵坐标，左端点二次导数值和右端点二次导数值。函数假定 `x` 序列是严格递增的，不满足这一约定的函数调用将产生未定义的结果。`cub_2pp()` 求解三次样条函数矩的方法如下：

$$l_i M_{i-1} + d_i M_i + u_i M_{i+1} = B_i := \Delta^{(1)} y_i - \Delta^{(1)} y_{i-1},\ i = 1, 2, \ldots, n-1 \tag{32}$$

其中 $l_i = \Delta^{(0)} x_{i-1}/6$，$d_i = (\Delta^{(0)} x_{i-1} + \Delta^{(0)} x_i)/3$，$u_i = \Delta^{(0)} x_i/6$。这里 $\Delta^{(0)} x_i := x_{i+1} - x_i$，$\Delta^{(1)} y_i := \Delta^0 y_i / \Delta^0 x_i$ 均为向前差分，与我们之前的定义有区别。由于 M_0 和 M_n 已分别由参数 `ypp0` 和 `yppn` 给定，对于任意 $n > 1$，我们令（runge/cubic.c#L32 和 runge/cubic.c#L33）：

$$B_1 := B_1 - l_1 M_0,\ B_{n-1} := B_{n-1} - u_{n-1} M_n \tag{33}$$

而后 d_1 至 d_{n-1}，l_2 至 l_{n-1} 和 u_1 至 u_{n-2} 组成的三对角矩阵配上 B_1 至 B_{n-1} 作为右端向量使用函数 `tdm_tri` 求解（runge/cubic.c#L34）立即得到 M_1 至 M_{n-1}（特别地，$n = 2$ 时 $n - 1 = 1$，B_1 和 B_{n-1} 为同一元素，但这不影响我们操作的正确性）。这一插值函数的测试代码位于源文件 runge/cubic-test.c 中。

在本题中，我们使用最简单的自然边界条件，即令 $M_0 = M_n = 0$。

将程序 runge/runge.c 中的多项式插值替换为三次样条插值，稍作修改，得到本题求解程序 runge/cub.c。运行之，标准错误中写入下表：

```
runge(-1.00) = 0.0384615    S20(-1.00) = 0.0384615      diff = 0
runge(-0.95) = 0.0424403    S20(-0.95) = 0.0425342      diff = 9.38981e-05
runge(-0.90) = 0.0470588    S20(-0.90) = 0.0470588      diff = 0
runge(-0.85) = 0.052459     S20(-0.85) = 0.0524313      diff = 2.7729e-05
runge(-0.80) = 0.0588235    S20(-0.80) = 0.0588235      diff = 0
runge(-0.75) = 0.06639      S20(-0.75) = 0.0663941      diff = 4.01189e-06
runge(-0.70) = 0.0754717    S20(-0.70) = 0.0754717      diff = 0
runge(-0.65) = 0.0864865    S20(-0.65) = 0.0864736      diff = 1.28553e-05
runge(-0.60) = 0.1          S20(-0.60) = 0.1            diff = 0
runge(-0.55) = 0.116788     S20(-0.55) = 0.116787       diff = 1.4467e-06
runge(-0.50) = 0.137931     S20(-0.50) = 0.137931       diff = 0
runge(-0.45) = 0.164948     S20(-0.45) = 0.164865       diff = 8.38962e-05
runge(-0.40) = 0.2          S20(-0.40) = 0.2            diff = 0
runge(-0.35) = 0.246154     S20(-0.35) = 0.246268       diff = 0.000114313
runge(-0.30) = 0.307692     S20(-0.30) = 0.307692       diff = 0
runge(-0.25) = 0.390244     S20(-0.25) = 0.38942        diff = 0.000824331
runge(-0.20) = 0.5          S20(-0.20) = 0.5            diff = 1.11022e-16
runge(-0.15) = 0.64         S20(-0.15) = 0.643169       diff = 0.00316894
runge(-0.10) = 0.8          S20(-0.10) = 0.8            diff = 0
runge(-0.05) = 0.941176     S20(-0.05) = 0.938866       diff = 0.00231026
runge(+0.00) = 1            S20(+0.00) = 1              diff = 0
runge(+0.05) = 0.941176     S20(+0.05) = 0.938866       diff = 0.00231026
runge(+0.10) = 0.8          S20(+0.10) = 0.8            diff = 0
runge(+0.15) = 0.64         S20(+0.15) = 0.643169       diff = 0.00316894
runge(+0.20) = 0.5          S20(+0.20) = 0.5            diff = 0
runge(+0.25) = 0.390244     S20(+0.25) = 0.38942        diff = 0.000824331
runge(+0.30) = 0.307692     S20(+0.30) = 0.307692       diff = 0
runge(+0.35) = 0.246154     S20(+0.35) = 0.246268       diff = 0.000114313
runge(+0.40) = 0.2          S20(+0.40) = 0.2            diff = 2.77556e-17
runge(+0.45) = 0.164948     S20(+0.45) = 0.164865       diff = 8.38962e-05
runge(+0.50) = 0.137931     S20(+0.50) = 0.137931       diff = 0
runge(+0.55) = 0.116788     S20(+0.55) = 0.116787       diff = 1.4467e-06
runge(+0.60) = 0.1          S20(+0.60) = 0.1            diff = 0
runge(+0.65) = 0.0864865    S20(+0.65) = 0.0864736      diff = 1.28553e-05
runge(+0.70) = 0.0754717    S20(+0.70) = 0.0754717      diff = 0
runge(+0.75) = 0.06639      S20(+0.75) = 0.0663941      diff = 4.01189e-06
runge(+0.80) = 0.0588235    S20(+0.80) = 0.0588235      diff = 6.93889e-18
runge(+0.85) = 0.052459     S20(+0.85) = 0.0524313      diff = 2.7729e-05
runge(+0.90) = 0.0470588    S20(+0.90) = 0.0470588      diff = 0
runge(+0.95) = 0.0424403    S20(+0.95) = 0.0425342      diff = 9.38981e-05
runge(+1.00) = 0.0384615    S20(+1.00) = 0.0384615      diff = 0
```

这一次我们看到，三次样条函数在各节点处几乎没有误差，且在非节点处保持了非常小的相对误差。这一现象在图 5 中看得更为清楚：

从图 5 中明显看出，视觉上三次样条函数与龙格函数精确符合。无论是在节点处还是节点外，三次样条函数都完整还原了龙格函数的形貌。特别地，对于切比雪夫近似表现较差的

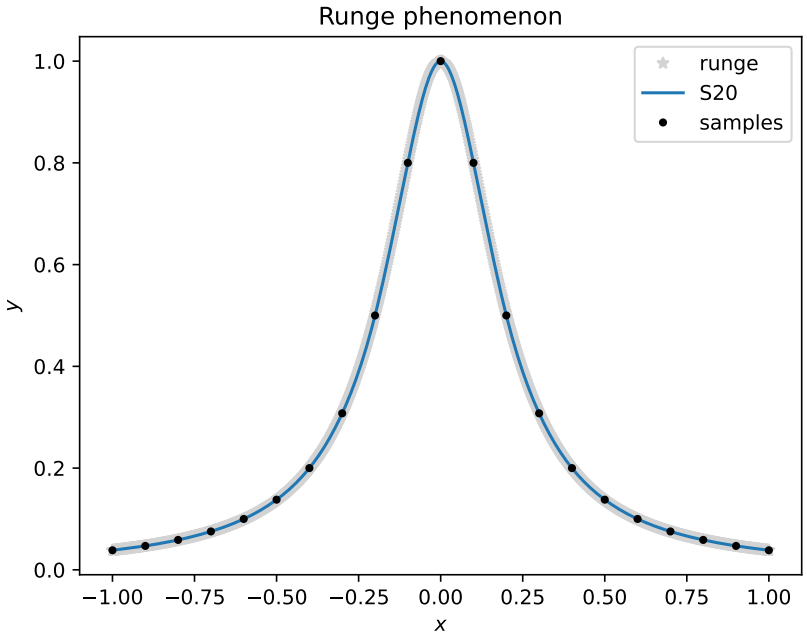

图 5: 龙格函数的 21 点三次样条插值结果

部分，即 $|x|$ 接近 0 的部分，三次样条函数依然与龙格函数符合较好；对于多项式内插表现较差的部分，即 $|x|$ 接近 1 的部分，三次样条函数没有出现振荡和发散的现象。三次样条函数分段近似的思想，从根本上解决了多项式插值在多项式次数增加时出现振荡的可能，同时利用函数局部光滑的特征，三次样条函数能够在各个部分分别较好地还原函数的形貌，从而在整体上精确描绘函数特征。这体现出三次样条函数在插值问题中的优越性，也为三次样条函数在计算机图形学中的广泛应用创造了条件。

V. 三次样条函数与计算机绘图

A. 九个采样点精确值

程序 plot/plot.c 在标准错误中写入下表：

```
NOTE: M_PI = 0x1.921fb54442d18p+1
t = +0x0.0000000000000p+0   x = +0x0.0000000000000p+0   y = +0x0.0000000000000p+0
t = +0x1.921fb54442d18p-1   x = +0x1.a827999fcef32p-3   y = +0x1.a827999fcef30p-3
t = +0x1.921fb54442d18p+0   x = +0x1.1a62633145c06p-54  y = +0x1.fffffffffffffp-1
t = +0x1.2d97c7f3321d2p+1   x = -0x1.3504f333f9de6p+0   y = +0x1.3504f333f9de6p+0
t = +0x1.921fb54442d18p+1   x = -0x1.0000000000000p+1   y = +0x1.1a62633145c07p-52
t = +0x1.f6a7a2955385ep+1   x = -0x1.3504f333f9de8p+0   y = -0x1.3504f333f9de6p+0
t = +0x1.2d97c7f3321d2p+2   x = -0x1.a79394c9e8a0cp-53  y = -0x1.0000000000001p+0
t = +0x1.5fdbbe9bba775p+2   x = +0x1.a827999fcef35p-3   y = -0x1.a827999fcef38p-3
t = +0x1.921fb54442d18p+2   x = +0x0.0000000000000p+0   y = -0x0.0000000000000p+0
```

为了保证精确，笔者在此使用了十六进制浮点数，其 C99 标准中 `printf()` 和 `scanf()` 的类型限定符为 `"a"`。值得一提的是笔者并不知道如何使用 Python 直接读取这种浮点数，所

以利用 `ctypes` 模块调用 `libc` 中的 `strtod()` 函数实现了 Python 版本带有错误检查的字符串转浮点函数 `atof()`（plot/atof.py#L10）。

B. 参数方程九点三次样条插值

同上，本题为了简单，使用了两端点处二阶导数为零的自然边界条件。由于数学表达太过于繁琐，请允许笔者借用 Python 的语法对分段函数进行描述。我们的程序 plot/plot.c 在标准错误中紧接着上表打印出：

```
xc = lambda t: {
 0.0e+00 <= t <= 7.9e-01:
  -0.0e+00*(t-7.9e-01)**3-7.9e-02*(t-0.0e+00)**3+3.1e-01*(t-0.0e+00)+0.0e+00,
 7.9e-01 <  t <= 1.6e+00:
  +7.9e-02*(t-1.6e+00)**3-5.4e-01*(t-7.9e-01)**3+1.9e-02*(t-7.9e-01)+2.5e-01,
 1.6e+00 <  t <= 2.4e+00:
  +5.4e-01*(t-2.4e+00)**3+1.6e-01*(t-1.6e+00)**3-2.0e+00*(t-1.6e+00)+2.6e-01,
 2.4e+00 <  t <= 3.1e+00:
  -1.6e-01*(t-3.1e+00)**3+7.4e-01*(t-2.4e+00)**3-1.4e+00*(t-2.4e+00)-1.3e+00,
 3.1e+00 <  t <= 3.9e+00:
  -7.4e-01*(t-3.9e+00)**3+1.6e-01*(t-3.1e+00)**3+1.4e+00*(t-3.1e+00)-2.4e+00,
 3.9e+00 <  t <= 4.7e+00:
  -1.6e-01*(t-4.7e+00)**3-5.4e-01*(t-3.9e+00)**3+2.0e+00*(t-3.9e+00)-1.3e+00,
 4.7e+00 <  t <= 5.5e+00:
  +5.4e-01*(t-5.5e+00)**3-7.9e-02*(t-4.7e+00)**3-1.9e-02*(t-4.7e+00)+2.6e-01,
 5.5e+00 <  t <= 6.3e+00:
  +7.9e-02*(t-6.3e+00)**3+0.0e+00*(t-5.5e+00)**3-3.1e-01*(t-5.5e+00)+2.5e-01,
}[True]
yc = lambda t: {
 0.0e+00 <= t <= 7.9e-01:
  -0.0e+00*(t-7.9e-01)**3+3.6e-01*(t-0.0e+00)**3+4.3e-02*(t-0.0e+00)+0.0e+00,
 7.9e-01 <  t <= 1.6e+00:
  -3.6e-01*(t-1.6e+00)**3-2.2e-01*(t-7.9e-01)**3+1.4e+00*(t-7.9e-01)+3.4e-02,
 1.6e+00 <  t <= 2.4e+00:
  +2.2e-01*(t-2.4e+00)**3-6.7e-01*(t-1.6e+00)**3+5.4e-01*(t-1.6e+00)+1.1e+00,
 2.4e+00 <  t <= 3.1e+00:
  +6.7e-01*(t-3.1e+00)**3-2.4e-16*(t-2.4e+00)**3-2.0e+00*(t-2.4e+00)+1.5e+00,
 3.1e+00 <  t <= 3.9e+00:
  +2.4e-16*(t-3.9e+00)**3+6.7e-01*(t-3.1e+00)**3-2.0e+00*(t-3.1e+00)+3.6e-16,
 3.9e+00 <  t <= 4.7e+00:
  -6.7e-01*(t-4.7e+00)**3+2.2e-01*(t-3.9e+00)**3+5.4e-01*(t-3.9e+00)-1.5e+00,
 4.7e+00 <  t <= 5.5e+00:
  -2.2e-01*(t-5.5e+00)**3-3.6e-01*(t-4.7e+00)**3+1.4e+00*(t-4.7e+00)-1.1e+00,
 5.5e+00 <  t <= 6.3e+00:
  +3.6e-01*(t-6.3e+00)**3+0.0e+00*(t-5.5e+00)**3+4.3e-02*(t-5.5e+00)-3.4e-02,
}[True]
```

这两个 Python 语句完备（但由于篇幅所限从而有效精度不高地）描述了 x 和 y 关于 t 的九点三次样条插值函数。

C. 绘图

笔者于图 6 中同时画出了心脏线及其参数方程九点三次样条插值近似曲线进行对比，其中所有节点用黑色小圆点标出。

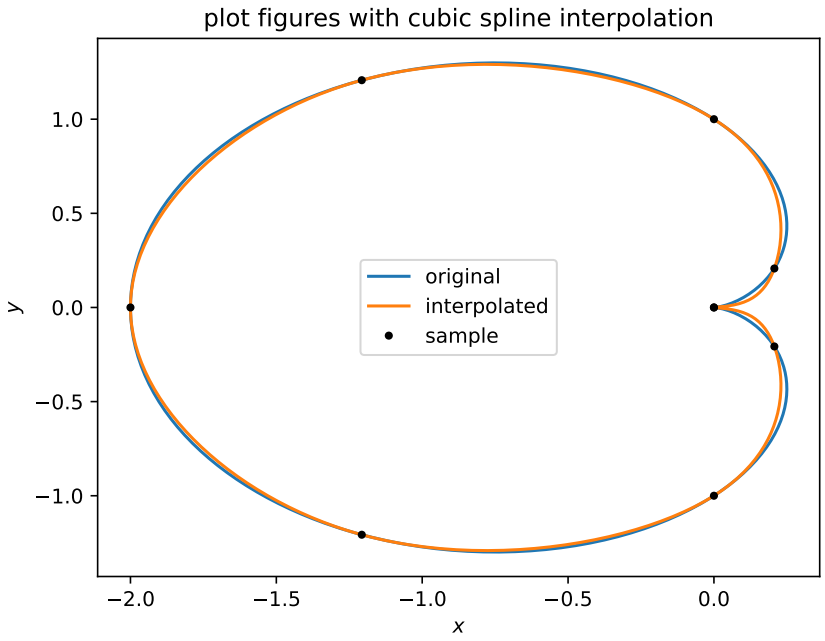

图 6: 九点参数方程三次样条插值绘制心脏线

从图 6 中明显看出，节点处三次样条插值曲线与心脏线完美相交。而由于取点数目较少，在远离节点处，特别是图右侧曲率较大的部分，三次样条函数与心脏线存在一些偏差。整体上，三次样条曲线非常平滑，各部分之间没有间断的痕迹。

D. 平滑连接之源

三次样条曲线能平滑连接平面上的离散点，根源于三次样条曲线每段无限光滑且保证节点处一阶和二阶导数连续的性质。以上参数方程三次样条插值的过程，在数学上构造了一条整体二阶可微的光滑曲线，其中一阶可微保证了曲线没有间断之处，二阶可微保证了曲线延伸方向没有突变之处，如此好的性质在人肉眼看来便是平滑了。

A 项目源代码

参见 `https://github.com/lyazj/numphy-02`。