考试允许带**简单的计算器**，切勿使用智能手机、电脑等设备！
学院规定，试题必须用英文出，请同学们做好准备。英文的试题中的英文词汇很少，基本上都是在课堂上使用过的。我们尽量在最后两次课和复习中，概括所有可能出现的词汇。

复习（**待完善**）

1、 Chapter 1

- Significant digit（有效数字）

  Approximate value

  True value

  定义 若近似值x^*的误差限是某一位的半个单位，该位到x^*的第一位非零数字共有 n 位，就说x^*有 n 位有效数字。

  切勿简单地比较有几个数字相同。一定要先计算绝对误差的绝对值，然后看小于哪位的半个单位，最后再往前数到首个非零的数字。

  例如，假设$x^* = 12.34567$有 5 位有效数字，$x^* = 12.34567$，则

  $$\varepsilon(x^*) = |x^* - x| < \frac{1}{2} \times 0.0001 = \frac{1}{2} \times 10^{-3} = \frac{1}{2} \times 10^{2-5}$$ Absolute error *bound*

  注意绝对误差限与有效数字之间的关系

  $$\varepsilon_r(x^*) = \frac{\varepsilon(x^*)}{|x^*|}$$ Relative error *bound*

  $$e(x^*) = x^* - x$$ Absolute error

  $$e_r(x^*) = \frac{e(x^*)}{x^*}$$ Relative error

- Iteration for solving x=g(x)
  - Fixed point Theorem

    The key point is to select an interval [a, b] , y=g(x) satisfies

    1. 自映射：[a, b]到[a, b]的映射
    2. 压缩：存在$K < 1$，使得

    $$|g(x_1) - g(x_2)| \le K|x_1 - x_2|$$

    或

    $$|g'(x)| \le K$$

    Results:

    There is one and only one fixed point: x=g(x)

    For any initial point x_0 in [a, b], the iteration $x_n = g(x_{n-1})$ converges to fixed point
- 常见的迭代格式
  1. Newton's Methods

Newton-Raphson iteration function（牛顿迭代函数）

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

Recursive rule（递推规则）

$$x_{n+1}=x_n-\frac{f(x_n)}{f'(x_n)}$$

Speed of convergence（收敛速度）：Order of convergence is 2 for single root
1 for multiple root.

注意应用牛顿迭代法的前提条件：

1、 函数 f 在闭区间[a,b]二阶连续可微
2、 函数 f 在闭区间有零点 p
3、 在零点的导数非零（单根）： $f'(p)\neq 0$

2. Secant method（弦截法）

Iteration formula（迭代公式）

$$p_{n+1}=p_n-\frac{f(p_n)}{f(p_n)-f(p_{n-1})}(p_n-p_{n-1})$$

Speed of convergence（收敛速度）：Order of convergence is $\frac{\sqrt{5}+1}{2}$

- Horner's method（秦九韶算法）

Exercise Write out a **recursive rule** to evaluate a polynomial and it derivative at the same time。

$$b_n=a_n, c_n=0$$
$$\begin{cases} b_k=a_k+x_0 b_{k+1} \\ c_k=b_{k+1}+x_0 c_{k+1} \end{cases} \qquad k=n-1,n-2,\ldots,0$$

则

$$\begin{cases} b_0=P(x_0) \\ c_0=P'(x_0) \end{cases}$$

- Bisection
  - $[a_0,b_0]\subset[a_1,b_1]\subset[a_2,b_2]\subset[a_3,b_3]\ldots$
  - Approximate value $c_k=\frac{a_k+b_k}{2}$
  - 误差估计

$$\varepsilon(x^*)=|c_k-x|<\frac{b_k-a_k}{2}=\frac{b_0-a_0}{2^{k+1}}$$

2、 Chapter 2

- Gaussian Elimination（高斯消去法）
  - Elimination process

- Triangular factorization

  A=LU，对应于不做行交换的高斯消去法

  PA=LU，对应于行交换的高斯消去法

  参见作业和讲义上的分解算法

- Solution of a Linear System

  1. Elimination and back substitute
     - 直接使用高斯消去法求解方程组，要选列主元以减少误差
  2. Using triangular factorization
     - 首先将矩阵 A 做 PA=LU 分解，在此基础上再来求解方程组。

$$AX = b$$
$$PAX = Pb$$
$$LUX = Pb$$
$$\begin{cases} LY = Pb \\ UX = Y \end{cases}$$

     - 求解 n 个方程组就可以得到矩阵的逆矩阵

$$\text{设} A^{-1} = (X_1, X_2, ..., X_n), \text{ 则}$$
$$A(X_1, X_2, ..., X_n) = (e_1, e_2, ..., e_n)$$
$$AX_k = e_k \qquad k = 1, 2, ..., n$$

- Computational Complexity

  1. $AX = b$ $\qquad \frac{n^2 - n}{2} + \frac{n^2 + n}{2} = n^2$
  2. 求逆矩阵的计算量 $n^2 \cdot n = n^3$

- Iterative Methods for Linear Systems，掌握迭代格式
  - Jacobi Iteration
  - Gauss-Seidel Iteration

  矩阵的表示形式 $X_k = BX_{k-1} + f$ 和一般的代数表示形式

- Vector norm, Matrix norm
  1. 懂得如何计算向量的 1 范数、二范数和无穷范数

$$\text{Let } X = (x_1, x_2, ..., x_n), then$$
$$\|X\|_1 = |x_1| + |x_2| + ... + |x_n|$$
$$\|X\|_2 = \sqrt{(x_1)^2 + (x_2)^2 + ... + (x_n)^n}$$
$$\|X\|_\infty = \max(|x_1|, |x_2|, ..., |x_n|)$$

  2. 矩阵的 1 范数（列范数）和无穷范数（行范数）

     2 范数的计算公式参见教案
  3. 掌握线性方程组迭代收敛的条件

$$\text{当} \|B\| < 1 \text{时，迭代 } X_{k+1} = BX_k + f \text{ 收敛}$$

3、 Chapter 3

(书上介绍了求解插值问题的三种方法

拉格朗日插值基法

牛顿插值公式

如果节点是 Chebysev 多项式的零点，那么利用正交性，方便地求出插值多项式)

- Lagrange Interpolation（拉格朗日插值）
  - Lagrange Interpolation base function（拉格朗日插值基函数）
  - Lagrange Interpolation polynomial（拉格朗日插值多项式）
  - Error function（插值公式的误差估计）

$$f(x)-L_N(x)=\frac{f^{(N+1)}(\xi)}{(N+1)!}(x-x_0)(x-x_1)...(x-x_N)$$

其中ξ是位于x, x_1,x_2,...,x_n之间的一个点。

- Newton Interpolation（牛顿插值）
  - Divided difference(均差，差商)
  - Divided difference table（均差表）
    1. 给出 n+1 个函数值约束，要懂得计算均差表
    2. 在此基础上再算出 Newton Interpolation polynomial
  - The relation between divided difference and derivative

$$f[x_0,x_1,x_2,...,x_n]=\frac{f^{(n)}(\xi)}{n!}$$

Where $\min(x_0,x_1,x_2,...,x_n)\le\xi\le\max(x_0,x_1,x_2,...,x_n)$

  - Newton polynomial
- Chebysheve Polynomial（切比雪夫多项式）
  - Recurrence relation（切比雪夫多项式的递推关系）

$$\begin{cases}T_0(x)=1\\T_1(x)=x\\T_k(x)=2xT_{k-1}(x)-T_{k-2}(x)\end{cases}$$

  - Trigonometric representation（三角表示法）

    The Chebysheve polynomial **of degree n** can be represented in trigonometric form as

$$T_N(x)=\cos(N\arccos(x))$$

  - Zeros（切比雪夫多项式零点的计算）.

    The Chebysheve Polynomial **of degree n** has n distinct zeros:

$$\cos\left(\frac{2k+1}{2n}\pi\right),k=0,1,2,...,n-1$$

  - point of alternation（切比雪夫多项式的交错点的计算）在交错点的值交替地为为-1、1。

$$\cos(N\arccos(x)) = \pm 1$$
$$N\arccos(x) = k\pi$$
$$\arccos(x) = \frac{k\pi}{N}$$
$$x = \cos\left(\frac{k\pi}{N}\right)$$

N+1 个交错点：$1, \cos\left(\frac{\pi}{N}\right), \cos\left(\frac{2\pi}{N}\right), ..., \cos\left(\frac{(N-1)\pi}{N}\right), -1$

- 以 N+1 次 Chebyshev 多项式的 N+1 零点为节点，讨论 Lagrange 插值误差

$$|f(x) - L_N(x)| = \left|\frac{f^{(N+1)}(\xi)}{(N+1)!}(x-x_0)(x-x_1)...(x-x_N)\right|$$
$$= \left|\frac{f^{(N+1)}(\xi)}{(N+1)!}\right| \cdot \left|\frac{P_{N+1}(x)}{2^N}\right|$$
$$= \left|\frac{f^{(N+1)}(\xi)}{(N+1)!}\right| \cdot \left|\frac{\cos(N\arccos(x))}{2^N}\right|$$
$$\le \frac{1}{2^N}\left|\frac{f^{(N+1)}(\xi)}{(N+1)!}\right|$$

4、 Chapter 4

- Spline function（样条函数）
  - boundary conditions
    1. Clamped cubic spline（第一类边界条件，夹持条件）
    2. Natural cubic spline（自然边界条件）
       Second boundary condition 第二类边界条件
    3. Third boundary conditions
  - three-moment equation（三弯矩方程）
    1. 内部点的约束规则

$$\mu_k m_{k-1} + 2m_k + \lambda_k m_{k+1} = g_k \qquad k = 1, 2, 3, ..., n-1$$

    2. 如何记忆边界约束

第一类边界条件：虚构出两个点，与端点重合，并利用均差和导数之间的关系

$$f[x_0, x_0] = f'(x_0)$$

第二类边界条件：m_0 、 m_n 已知

n-1 个未知量，n-1 个方程

第三类边界条件：$m_0 = m_n$

不把 m_0 看着未知变量

内部点 n-1 个，n-1 个方程

边界点 x_n 建立的方程记忆要点：在 x_n 右边虚构一个点：

$$\begin{cases} x_{n+1} = x_n + (x_1 - x_0) \\ y_{n+1} = y_1 \end{cases}$$

3. 样条函数的表达式

- Curve Fitting（曲线拟合）
  - Least – Squares line（最小二乘拟合直线）

    列正规方程，解方程组！

    求 A、B 有两种方法：（1）解方程组；（2）习题中介绍的更稳定的算法
  - 求非线性曲线拟合

    1. Transformation for Data Linearization；

    参见 Example 4.4

    2、直接按照定义，求偏导数，列方程组。
  - Linear Least Squares（最小二乘拟合一组函数的线性组合）
    - Problem: Let $f(x) = \sum_{j=1}^{M} c_j f_j(x)$, find out $\{c_j\}$, so that it minimize

$$E(c_1, c_2, ..., c_M) = \sum_{k=1}^{N} (f(x_k) - y_k)^2 = \sum_{k=1}^{N} \left(\sum_{j=1}^{M} c_j f_j(x_k) - y_k \right)^2$$

    - 给定一组基函数，The process to derive the normal equation（推导出最小二乘拟合曲线的正规方程）

      对 $E(c_1, c_2, ..., c_M)$ 两边求偏导数（共 M 个），并令偏导数等于零，

      得到一个含 M 个未知变量的方程组（M 个），用矩阵的形式可以表示为

$$F^T F C = F^T Y$$

    - How to solve **normal equation**（正规方程）

      一般来说，F^T 不是可逆矩阵，甚至不是一个方阵。

      $F^T F$ 是一个对称矩阵，当基函数相互独立时，他们是非奇异的！

Example

Suppose M=3， $f_1(x) = 1$，$f_2(x) = x$，$f_3(x) = x^2$，则

$$F = \begin{pmatrix} f_1(x_1) & f_2(x_1) & f_3(x_1) \\ f_1(x_2) & f_2(x_2) & f_3(x_2) \\ f_1(x_3) & f_2(x_3) & f_3(x_3) \\ & & \\ f_1(x_N) & f_2(x_N) & f_3(x_N) \end{pmatrix} = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ & & \\ 1 & x_N & x_N^2 \end{pmatrix}, C = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}, Y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \\ y_N \end{pmatrix}$$

$$F^TFC = F^TY$$

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ x_1 & x_2 & x_3 & .. & x_N \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_N^2 \end{pmatrix} \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ & & \\ 1 & x_N & x_N^2 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ x_1 & x_2 & x_3 & .. & x_N \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_N^2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \\ y_N \end{pmatrix}$$

$$\begin{pmatrix} N & x_1+x_2+x_N & x_1^2+x_2^2+x_N^2 \\ x_1+x_2+x_N & x_1^2+x_2^2+x_N^2 & x_1^3+x_2^3+x_N^3 \\ x_1^2+x_2^2+x_N^2 & x_1^3+x_2^3+x_N^3 & x_1^4+x_2^4+x_N^4 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} y_1+y_2+y_N \\ x_1y_1+x_2y_2+x_Ny_N \\ x_1^2y_1+x_2^2y_2+x_N^2y_N \end{pmatrix}$$

如果选择的基函数不是$1, x, x^2$，可能可以得到更简单的表达式，比如对角矩阵！

5、 Chapter 5

- Degree of precision of a quadrature（积分）
  - 用$1, x, x^2, x^3, \dots$逐个代入，直到求积公式的左右不相等！
  - N+1 个节点插值型求积公式的代数精度至少 N
  - 如果 N 为偶数，且 N+1 个节点是等分点，则代数精度为 N+1。
    （梯形公式=1、Simpson formula=3）
- Stability （稳定性）
  Error propagation:
  if the sum of all absolute of coefficient >(b-a), then the quadrature(求积分) formula is unstable.
- Trapezoidal formula， composite formula
  - 懂得推导，牢记公式
    1. $\int_a^b f(x)dx \approx (b-a)\left(\frac{1}{2}f(a)+\frac{1}{2}f(b)\right)$
    2. 系数之和为 b-a，b-a 是不可避免的。
    3. $T(f,h)=\frac{h}{2}(f(a)+f(b))+h\sum_{k=1}^{M-1}f(x_k)$
  - 懂得误差分析
- Simpson rule, composite simpson rule
  - 懂得推导，牢记公式
    1. $\int_a^b f(x)dx \approx (b-a)\left(\frac{1}{6}f(a)+\frac{4}{6}f(\frac{a+b}{2})+\frac{1}{6}f(b)\right)$
    2. $S(f,h)=\frac{h}{3}(f(a)+f(b))+\frac{2h}{3}\sum_{k=1}^{M-1}f(x_{2k})+\frac{4h}{3}\sum_{k=1}^{M-1}f(x_{2k-1})$
  - 懂得误差分析
- Gauss-Legendre Integration
  - Degree of precision
    1. N 个节点，待定的量有 2N 个，可以达到 2N-1 次代数精度
    2. The nodes is just the roots of Legendre polynomial of degree N。
  - Legendre polynomial

1. Recursive rule

Legendre 多项式

$$\begin{cases} P_0(x)=1 \\ P_1(x)=x \\ (n+1)P_{n+1}(x)=(2n+1)xP_n(x)-nP_{n-1}(x) \end{cases}$$

2. Derivative representation

$$P_n(x)=\frac{1}{2^n n!}\frac{d^n}{dx^n}\left\{\left(x^2-1\right)^n\right\}$$

$$\frac{1}{2^3 3!}\frac{d^3}{dx^3}\left\{\left(x^2-1\right)^3\right\}$$
$$=\frac{1}{2^3 3!}\frac{d^3}{dx^3}\left\{x^6-3x^4+3x^2-1\right\}$$
$$=\frac{1}{2^3 3!}\left(6\cdot 5\cdot 4x^3-3\cdot 4\cdot 3\cdot 2x\right)=....$$

3. 几个最简单的高斯求积公式

$$\int_{-1}^{1} f(x)dx \approx \mathrm{f}\left(-\frac{\sqrt{3}}{3}\right)+\mathrm{f}\left(\frac{\sqrt{3}}{3}\right)$$

$$\int_{-1}^{1} f(x)dx \approx \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right)+\frac{8}{9}f(0)+\frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right)$$

- Romberg Algorithm（龙贝格算法）

  例如 **64** 等分区间的算法

T(f, 64h)		
T(f, 32h)	**S(f, 32h)**	
T(f, 16h)	**S(f, 16h)**	**B(f, 16h)**
T(f, 8h)	**S(f, 8h)**	**B(f, 8h)**
T(f, 4h)	**S(f, 4h)**	**B(f, 4h)**
T(f, 2h)	**S(f, 2h)**	**B(f, 2h)**
T(f, h)	**S(f, h)**	**B(f, h)**

T(f,b-a)		
T(f, (b-a)/2)	**S(f, (b-a)/2)**	
T(f, (b-a)/4)	**S(f, (b-a)/4)**	**B(f, (b-a)/4)**
T(f, (b-a)/8)	**S(f, (b-a)/8)**	**B(f, (b-a)/8)**
T(f, 4h)	**S(f, 4h)**	**B(f, 4h)**
T(f, 2h)	**S(f, 2h)**	**B(f, 2h)**
T(f, h)	**S(f, h)**	**B(f, h)**

例如：

$$S(f,h)=\frac{4}{3}T(f,h)-\frac{1}{3}T(f,2h)$$

$$B(f,h)=\frac{16}{15}S(f,h)-\frac{16}{15}S(f,2h)$$

也可以改写成

$$S(f,h)=T(f,h)+\frac{1}{3}\left(T(f,h)-T(f,2h)\right)$$

Which is better? 从稳定性和计算复杂性的两个方面来看，后者更好！

于是计算 S(f,h)等近似值即可以直接计算，也可以利用 Romberg 算法来计算。

- Richardson improvement

1. $\int_a^b f(x)dx=T(f,h)+E_T(f,h)=T(f,h)+a_1h^2+a_2h^4+a_3h^6+.....$
2. 消去 h^2 项（term），得到 Simpson 公式
   - $\int_a^b f(x)dx=T(f,2h)+E_T(f,2h)=T(f,2h)+4a_1h^2+16a_2h^4+64a_3h^6+.....$

     联立消去 h^2 项

     $\int_a^b f(x)dx=S(f,h)+b_1h^4+b_2h^6+.....$
3. 消去 h^4 项（term），得到 Boole 公式
4. 消去 h^6 项（term），得到 Romberg 公式

6、 Chapter 6 Solution of Differential equation（求解常微分方程）

$$\begin{cases}\dfrac{dy}{dx}=f(x,y)\\ y(x_0)=y_0\end{cases}$$

Establish recursive rule， and generate a sequence of points：

$$(x_0,y_0),(x_0+h,y_1),(x_0+2h,y_2),...,(x_0+nh,y_n)$$

$$y(x_0+nh)\approx y_n$$

近似值（approximate value）： y_n

准确值（true value）: $y(x_0+nh)$

- Lipschitz 条件和解的存在性与唯一性
- Euler’s Method

教材上的例子 Example6.2、6.4

- Exercise 7

  微分方程按 Euler 方法离散化

  M 等分区间[a,b]， 然后计算出 y_M ，就得到 y(b)的近似值。

- Exercise 8 不满足 Lipschitz 条件，解不唯一！y=0 和 $y=t^{3/2}$ 是两个不同的解。

  在一个固定点 t_0 ，解方程组，得到数学上的 The true value is $t_0^{3/2}$

  n 等分区间 $[0,t_0]$， $h=\frac{t_0}{n}$ ，按照 Euler 方法计算出近似值 y_n

  最后取极限，证明极限不等于 The true value $t_0^{3/2}$ ！

局部离散化误差 local discretization error $O(h^2)$

全局离散化误差 global discretization error $O(h)$

- Heun's Method

理解如何将微分方程离散化

能够得到高一阶的精度

局部离散化误差 local **discretization** error $O(h^3)$

全局离散化误差 global discretization erre $O(h^2)$