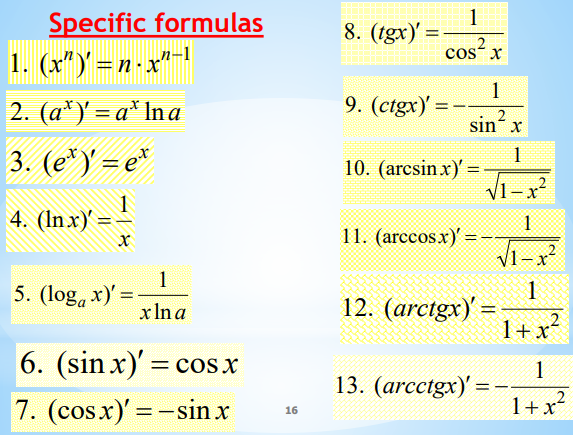
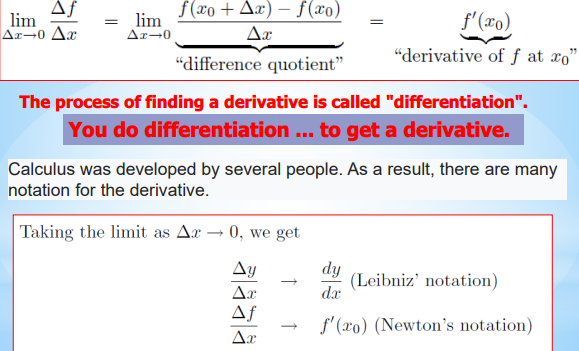
12月26日15:45(4-318室)**考试准备问题：**

# 1. Introduction to Derivatives. Derivatives formulas.导数定义、公式。



# 2. Maple can perform symbolic and numeric differentiation. Maple 上微分的表示和计算。

Diff( 5\*x^9 - 9\*x^2 + 3\*exp(x) , x) = diff( 5\*x^9 - 9\*x^2 + 3\*exp(x) , x)



Diff((x + 1)\*ln(x), **x** $ **3**) = diff((x + 1)\*ln(x), **x** $ **3**)



# 3. Computing partial derivatives.偏导数的计算。

f := (x, y) -> x^5 + y\*x

R1 := D[1](f); R1 := (x, y) -> 5\*x^4 + y R2 := D[2](f); R2 := (x, y) -> x

R1(1, 2); 7

f := (x, y) -> 6\*y^2 + 2\*x^3;

eq5 := diff(f(x, y), x $ 2); eq6 := diff(f(x, y), y $ 2);

# 4. Methods for specifying functions.确定函数的方法。

f := (x, y) -> cos(Pi\*x\*y)\*y + sin(x + Pi\*y);

x := 0.01;y := 0.999;

eq8 := evalf(f(x, y));

# 5. Distinction between Functional Operators and Other Expressions.函数运算与其他表达式的区别。

f:=(x)→x+1(函数)与g:=x+1(表达式)

To evaluate the functional operator f at a value of x:Specify the value as an argument to f

计算函数运算符f当x=22时的值，会将该值指定为f的参数：f(22) ;23

To evaluate the expression g at a value of x:You must use the eval command.

计算 表达式 g当x=22时的值： 需要用到eval 函数：eval(q,x=22);23

# 6. Several ways to find the extrema of functions.求函数极值的几种方法。

## 1. Search for extrema of functions by zeros of the first derivative.找一阶导数为零的点

## 2. Finding extrema using the extrema function.极值函数求极值。

## 3. Finding the minimum and maxima of functions用最小值、最大值函数求极值

## 4. Solving linear programming problems in the MAPLE package 使用simplex库求极值

# 7. Fermat's Theorem.费马定理。

If f has a local maximum or minimum at c , and if f ‘(c) exists, then f ‘(c)= 0

# 8. Finding the minimum and maxima of functions.求函数的最小值和最大值。

## 1. Search for extrema of functions by zeros of the first derivative.找一阶导数为零的点

y1 := expand((x - 3)\*(x - 1)\*x\*(x + 2)); #expand - 展开括号

dy1 := simplify(diff(y1, x)); #simplify - 简化表达式，

extrem := evalf(solve(dy1 = 0, x)); #一阶导为0时，可得到其极值



factor - 因式分解 normal - 共同点(交点) combine - 转换幂(或三角表达式) collect - 带来相似的成员。

## 2. Finding extrema using the extrema function.极值函数求极值。

extrema(expr,constrs,vars,’s’)

expr：表达式；constrs:约束/约束集合(约束条件或者区间) vars:变量/变量集合 ’s’:未评估的名称

y := x -> x^2/(x - 1);

extrema(y(x), {}, {x}, 's'); {0,4} # 求得极值y=0, y=4

s; {{x=0},{x=2}} # 极值处x的值

## 3. Finding the minimum and maxima of functions用最小值、最大值函数求极值

minimize(expr,opt1,opt2,…optn) maximize(expr,opt1,opt2,…optn)

opt1,opt2,…optn附加的约束条件

fn := 30\*x^4 - (x + 1)^2 + 10\*y^2

minimize(fn); -1.454109060

evalf(maximize(fn));Float(infinity) #当数值计算不出时使用evalf（）函数

## 4. Solving linear programming problems in the MAPLE package 使用simplex库求极值

包：with(simplex)

minimize (objective function, {restrictions}, NONNEGATIVE)

maximize (objective function, {restrictions}, NONNEGATIVE)

NONNEGATIVE：表示传入的变量是非负的 restrictions：约束条件

with(simplex)：

z := 2\*x1 + 2\*x2;

maximize(z,{-6<=3\*x1-2\*x2,3<= 3\*x1+x2,x1<= 3},NONNEGATIVE);{x1 = 3,x2 =15/2}

subs(%, z); 21 #使用 subs 命令将最优解代入目标函数 z 中，以获取最优解的数值。

# 9. Solving linear programming problems in the MAPLE package.线性相关问题的包。

## 1包名：with(simplex)

minimize (objective function, {restrictions}, NONNEGATIVE)

maximize (objective function, {restrictions}, NONNEGATIVE)

NONNEGATIVE：表示传入的变量是非负的 restrictions：约束条件

with(simplex)：

z := 2\*x1 + 2\*x2;

maximize(z,{-6<=3\*x1-2\*x2,3<= 3\*x1+x2,x1<= 3},NONNEGATIVE);{x1 = 3,x2 =15/2}

subs(%, z); 21 #使用 subs 命令将最优解代入目标函数 z 中，以获取最优解的数值。

## 2包名：with(Optimization)

with(Optimization);with(plots);

z := 5\*x + 4\*y;

init := [x + y <= 4, 0.5 <= x, 0.5 <= y]; # 约束条件

LPSolve(z, init); # 默认为最小化问题

LPSolve(z, init, maximize); # 最大值

图形中的展现：

p1 := inequal(init, x = 0 .. 5, y = 0 .. 4, optionsfeasible = (color = red)); # inequal 命令创建一个不等式图形

#optionsfeasible = (color = red) 参数设置可行区域的颜色为红色

p2 := contourplot(z, x = 0 .. 5, y = 0 .. 4); #使用 contourplot 命令创建一个等高线图

display([p1, p2]);

# 10. Plotting in Maple.MAPLE包：plots

## 包名：with(plots)：

Transparency:透明度0 to 1(1全透明)，'legend' ：函数的线条标题(这个函数是哪条线)thickness：厚度

f := x\*sin(x); fderiv := diff(f, x);

fplot := plot(f, x = -2\*Pi .. 2\*Pi, 'color' = "Red", 'legend' = f, 'thickness' = 4);

fdplot := plot(fderiv, x = -2\*Pi .. 2\*Pi, color = "Green", legend = fderiv);

display([fplot, fdplot], 'title' = "A function and its derivative", 'titlefont' = ["Helvetica", 16]);

## 三维图：

plot3d( sin(x)\*sin(y), x=-2 Pi..2 Pi , 'title'=A sin\*`sin\_\_ `3 d Plot);

## 点图：

pointplot([[0, 1], [1, -1], [3, 0], [4, -3]], 'color' = ["Red", "Green", "Black", "Blue"], 'symbol' = 'asterisk', 'symbolsize' = 35, 'view' = [-1 .. 5, -4 .. 2])

Symbols： box(框), cross(十字叉), circle(圈), point(点),diamond (菱)等

多边形图：

polygonplot3d(Matrix([[0, 1, 1], [1, -1, -2], [3, 0, 5], [1, 1, 1]], 'datatype' = 'float'), 'color' = "Green", 'axes' = 'boxed')

## 隐函数图：

implicitplot(x^2 - y^2 = 1, x = -4 .. 4, y = -4 .. 4, color = green, thickness = 2)

# 11. The Method of Least Squares in the case of linear dependence.线性相关的最小二乘法。

restart;

with(plots);

X := [-2, -1, 0, 1, 2, 3]; Y := [5.6, 5., 4.3, 4., 3.6, 3.];

q0 := plot([[X[i], Y[i]] $ (i = 1 .. 6)], style = point, color = red); display(q0);

n := 6; c11 := 0; c12 := 0; c21 := 0; c22 := 0; d1 := 0; d2 := 0;

for i to n do # n为点数

c11 := X[i]^2 + c11;

c12 := c12 + X[i];

c21 := c21 + X[i];

c22 := n;

d1 := X[i]\*Y[i] + d1;

d2 := d2 + Y[i];

end do;

eq := {d1 = a\*c11 + b\*c12, d2 = a\*c21 + b\*c22};eq := {4.0 = 19 a + 3 b, 25.5 = 3 a + 6 b}

e := solve(eq);e := {a = -0.5000000000, b = 4.500000000}

a := rhs(e[1]); a := -0.50

b := rhs(e[2]); b := 4.5

y := a\*x + b; y := -0.50 x + 4.50

q1 := plot(y, x = -3 .. 5, color = red);display(q0, q1);

# 12. The Method of Least Squares in the case of a quadratic dependence.二次相关的最小二乘法。

restart;with(plots);

X := [-1., 0., 1., 2., 3., 4., 5.]; Y := [-9.8, -3.1, 0.3, -1.2, -6.1, -14.7, -28.2];

q0 := plot([[X[i], Y[i]] $ (i = 1 .. 7)], style = point, color = red);display(q0);

n := 7;

c11 := 0;c12 := 0;c13 := 0;

c21 := 0;c22 := 0;c23 := 0;

c31 := 0;c32 := 0;c33 := 0;

d1 := 0; d2 := 0; d3 := 0;

for i to n do

c11 := X[i]^4 + c11; c12 := X[i]^3 + c12; c13 := X[i]^2 + c13;

c21 := X[i]^3 + c21; c22 := X[i]^2 + c22; c23 := c23 + X[i];

c31 := X[i]^2 + c31; c32 := c32 + X[i]; c33 := n;

d1 := X[i]^2\*Y[i] + d1; d2 := X[i]\*Y[i] + d2; d3 := d3 + X[i]^0\*Y[i];

end do;

eq := { a\*c11 + b\*c12 + c\*c13 = d1, a\*c21 + b\*c22 + c\*c23 = d2, a\*c31 + b\*c32 + c\*c33 = d3};

{56. a + 14. b + 7 c = -62.80, 224. a + 56. b + 14. c = -210.4, 980. a + 224. b + 56. c = -1009.4}

e := solve(eq); {a = -1.997619048, b = 4.961904762, c = -2.914285714}

a := rhs(e[1]);a := -1.997619048

b := rhs(e[2]);b := 4.961904762

c := rhs(e[3]);c := -2.914285714

P := a\*x^2 + b\*x + c; P := -1.997619048 x + 4.961904762 x - 2.914285714

q1 := plot(P, x = -6 .. 6, color = green);display(q0, q1);

# 13. Definite integrals.定积分。

不定积分：

Int(x^2 + 3, x) = int(x^2 + 3, x)



Int(10\*x + 5, x = 3 .. 6) = int(10\*x + 5, x = 3 .. 6)



# 14. Numerical Integration. The formula of rectangles.数值积分1:矩形法

restart;

f := x -> x^4 - 6\*sqrt(x);

centr\_P := proc(f, a, b, n)

local i, S; S := 0;

for i from 0 to n - 1 do

S := S + (a + (i + 1/2)\*(b - a)/n)^4 - 6\*sqrt(a + (i + 1/2)\*(b - a)/n);

end do;

S := evalf(S\*(b - a)/n);

end proc;

centr\_P(f, 1, 10, 1000); 19877.29541

# 15. Numerical Integration. Trapezoidal formula.数值积分2:梯形法

f := x -> x^4 - 6\*sqrt(x);

trapezoid\_P := proc(f, a, b, n)

local i, S; S := 0;

for i from 0 to n - 2 do

S := S + f(a + (i + 1)\*(b - a)/n);

end do;

S := S + (f(a) + f(b))/n;

S := evalf(S\*(b - a)/n); print(S);

end proc;

evalf(trapezoid\_P(f, 1, 10, 1000));19875.81278

# 16. Numerical Integration. Simpson's formula.数值积分3:辛普森法

f := x -> x^4 - 6\*sqrt(x);

n := 20; a := 1; b := 10; S := 0; h := (b - a)/n;

for i by 2 to n do

S := evalf(S + f(a + (i - 1)\*h) + 4\*f(h\*i + a) + f(a + (i + 1)\*h));

end do;

Simpson\_A := S\*h/3;

Simpson\_A; 19877.35853

# 17. The concept of functional and variation problems.泛函问题和变分问题的概念。

## 泛函问题（Functional Problem）：

泛函问题是研究函数的集合上的函数的性质和行为的问题。在泛函问题中，函数被视为对象，而不是简单的数值。泛函是将函数映射到实数的函数，通常用积分形式表示。泛函问题的目标是找到满足一定条件的函数，使得泛函取得最大值或最小值。

## 变分问题（Variational Problem）：

变分问题是泛函问题的一种特殊形式，其中涉及到对函数的变分。变分是指对函数进行微小的变化或变动。在变分问题中，我们试图找到一个函数，使得在给定的约束条件下泛函取得极值。

变分问题通常涉及到找到一个函数，使得泛函的变分为零。这个条件可以用欧拉-拉格朗日方程表示，它是变分问题的基本方程。通过求解欧拉-拉格朗日方程，我们可以找到满足约束条件的极值函数。

# 18. The Euler equation.欧拉方程

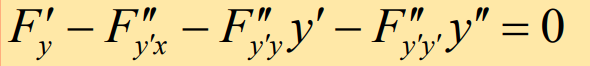
欧拉方程是变分法中的一条基本方程，用于求解给定变分问题的极值函数（即优化或最小化某个泛函的函数）。

欧拉方程通过考虑形式为以下的泛函来导出：J[y] = ∫(a, b) F(x, y, y') dx

其中 y = y(x) 是未知函数，y' = dy/dx 表示其关于 x 的导数，F 是关于 x、y 和 y' 的函数。

为了找到 J[y] 的极值函数，欧拉方程表明函数 y 必须满足以下微分方程：

d/dx (dF/dy') - dF/dy = 0 或者等价地写为：dF/dy - d/dx (dF/dy') = 0：

再次求导得：

restart;with(plots);

F := (x, Y, DY) -> 2\*Y\*x + Y\*DY + DY^2;

x0 := 0; x1 := 2; y0 := 0;

J := int(F(x, y(x), diff(y(x), x)), x = x0 .. x1);



F\_ := F(x, Y, DY); dF\_x := diff(F\_, x); dF\_Y := diff(F\_, Y); dF\_DY := diff(F\_, DY);

eq := dF\_Y - diff(dF\_x, DY) - diff(dF\_Y, DY)\*DY - diff(F\_, DY $ 2)\*D2Y = 0 eq := 2\*x - 2\*D2Y = 0

eq1 := subs(Y = y(x), DY = diff(y(x), x), D2Y = diff(y(x), x $ 2), eq) eq1 := 2\*x - 2\*diff(y(x), x, x) = 0

rez := dsolve(eq1) rez := y(x) = 1/6\*x^3 + \_C1\*x + \_C2

assign(rez); y(x); 1/6\*x^3 + \_C1\*x + \_C2

us := subs(Y = y(x), DY = diff(y(x), x), D2Y = diff(y(x), x $ 2), dF\_DY) = 0us := x^2 + 2\*\_C1 + 1/6\*x^3 + \_C1\*x + \_C2 = 0

right := subs(x = x1, us) right := 16/3 + 4\*\_C1 + \_C2 = 0

left := subs(x = x0, y(x)) = y0 left := \_C2 = 0

rez1 := solve({left, right}) rez1 := {\_C1 = -4/3, \_C2 = 0}

y := x -> subs(rez1, y(x)) y := x -> 1/6\*x^3 - 4/3\*x

F(x, y(x), diff(y(x), x)); 2\*(1/6\*x^3 - 4/3\*x)\*x + (1/6\*x^3 - 4/3\*x)\*(x^2/2 - 4/3) + (x^2/2 - 4/3)^2

J;-112/45

q0 := plot(y(x), x = x0 .. x1, title = "...", axes = NORMAL, labels = ["x", "y"], thickness = 1, font = [TIMES, 12]);

display(q0);