Maple

MA 141

Lesson 1 Calculations

TOPIC: introduction to Maple

MATERIAL:

Assigning Variables:

A := # ; or assign(name=#) ;

unassigning Variables:

A := ‘A’ ;

Names for Variables:

* exceptions: I(square root of -1), Pi, gamma(Euler's constant)…

Digits:

Digits := 30 ; or evalf(#,# of digit)

Built-in Functions:

exponential exp(x) /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 2.55.14 PM.png

natural logarithm ln(x)

trigonometric(radians) sin(x), cos(x), tan(x)…

inverse trigonometric arcsin(x), arcos(x), arctan(x)…

absolute value abs(x)

square root sqrt(x)

nth root surd(x,n)

sign signum(x)—(+1 if x>0, 0 if x=0, -1 if x<0)

Lesson 2 Expressions, Functions, Equations and Allgebra

TOPIC: expression and function

MATERIAL:

Expression:

subs( x=#, variableName) ;

convert(expression, surd) ; —used for fractional power

Function:

f := x -> x^2 ;

Converting:

function to expression: f := x-> ;

subs(x=#, f(x)) ;

expression to function: expr := ;

f := unapply(expr, x) ;

Manipulating Expression:

simplify(expr) ;

expand(expr) ;

factor (expr) ;

Equation:

lhs(equationName) ; —left hand side

rhs(equationName) ; —right hand side

Solving equation:

solve(equation, variable) ；

solving RootOf:

allvalues( variableName[# of index]) ;

allvalues( RootOf(…, index=#)) ;

solving sets:

solve({equations}, {variables}).

EXAMPLE:

expr to f:

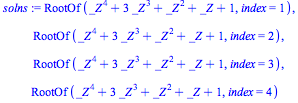
/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.40.55 PM.png /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.44.03 PM.png

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.41.27 PM.png /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.41.34 PM.png

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.41.41 PM.png /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.41.55 PM.png

solving RootOf:

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.48.16 PM.png /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.48.31 PM.png

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.48.40 PM.png 

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.50.15 PM.png or /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.50.59 PM.png

solving sets:

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.52.33 PM.png /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.52.51 PM.png

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.52.41 PM.png /Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.52.57 PM.png

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.54.28 PM.png

/Users/wentingzheng/Desktop/Screen Shot 2016-09-10 at 3.54.07 PM.png

Lesson 3 Graphs and Limits

TOPIC: plot a graph; find limits

MATERIAL:

Lists:

list [] no order

set {} order

Plotting Graphs:

plot(expr, x=#..#) ;

using surd to plot an expression with an odd root of a negative number using ^

plot(expr, x=#..#, y=#..#, discount=true) ; no vertical asymptote

plot(expr, x=#..#, color=…) ;

plot(expr, x=#..#, thickness=#) ;

plot(expr1, expr2, x=#..#, color=[…, …]) ;

Limits:

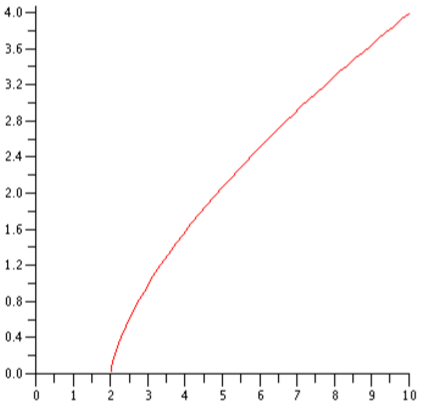
limit(expr, x=#, left/right) ;

EXAMPLE:

plotting an odd root of a negative number using ^

../../../Screen%20Shot%202016-09-10%20at%204.15.41%20PM.png ../../../Screen%20Shot%202016-09-10%20at%204.16.03%20PM.png

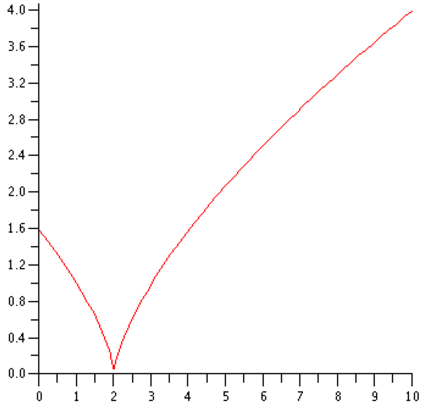
../../../Screen%20Shot%202016-09-10%20at%204.17.00%20PM.png



RIGHT WAY:

../../../Screen%20Shot%202016-09-10%20at%204.17.48%20PM.png ../../../Screen%20Shot%202016-09-10%20at%204.18.01%20PM.png

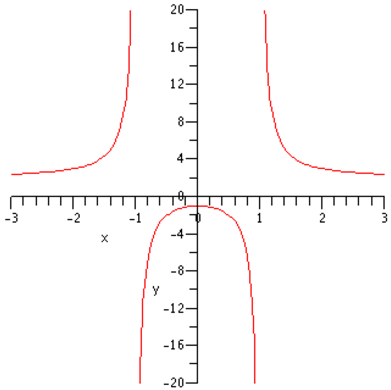
../../../Screen%20Shot%202016-09-10%20at%204.18.10%20PM.png



plotting without vertical asymptote:

../../../Screen%20Shot%202016-09-10%20at%204.20.38%20PM.png ../../../Screen%20Shot%202016-09-10%20at%204.20.50%20PM.png

../../../Screen%20Shot%202016-09-10%20at%204.21.12%20PM.png



Lesson 4 Differentiation and Implicit-plot

TOPIC: (implicit) differentiation; graph an equation; package  
MATERIAL:

Differentiation:

expression: diff(expr, variable$#) ; —$# means higher differentiation

function: D(f);

(D@D@D)(f); = (D@@3)(f); —higher differentiation

Graph of Equations and Packages:

with(plots):

implicitplot (eqn, x=#..#, y=#..#, grid= [#,#], color=…) ;

plots[implicitplot] (eqn, x=#..#, y=#..#) ;

Implicit Differentiation and Display:

implicitdiff (eqn, y, x) ; —dy/dx

display ({eqn1, eqn2}) ;

EXAMPLE:

implicit diff:

../../../Screen%20Shot%202016-09-10%20at%205.11.28%20PM.png ../../../Screen%20Shot%202016-09-10%20at%205.11.34%20PM.png

../../../Screen%20Shot%202016-09-10%20at%205.11.45%20PM.png ../../../Screen%20Shot%202016-09-10%20at%205.11.54%20PM.png

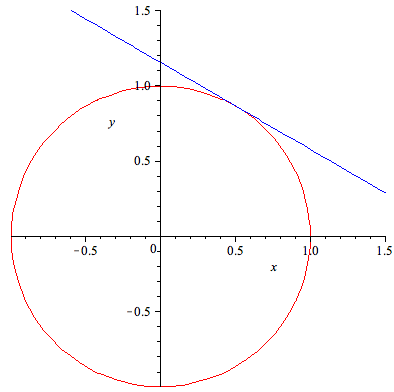
display:

../../../Screen%20Shot%202016-09-10%20at%205.14.45%20PM.png

../../../Screen%20Shot%202016-09-10%20at%205.15.17%20PM.png

../../../Screen%20Shot%202016-09-10%20at%205.16.29%20PM.png

../../../Screen%20Shot%202016-09-10%20at%205.16.57%20PM.png



MA 241

Lesson 1 Definite Integrals

TOPIC: area under a curve

MATERIAL:

Package: with(student):

Riemann Sums:

leftbox/middlebox/rightbox ( f(x) , x = a..b, n) ;

leftsum/middlebox/rightsum ( f(x) , x = a..b, n) ;

* decreasing function, leftsum > rightsum

Increasing the Number of b=Boxes:

lefts := n->leftsum ( f(x), x=a..b, n ) ;

Limit ( lefts(n), n=infinity ) ;

Riemann Integrals:

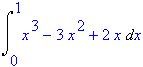
Int (f(x), x=a..b ) ;

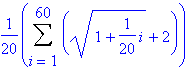
value(%) ;

Function Theorem of Calculus:

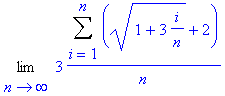
int ( f(x), x ) ;

EXAMPLE:

> Int(x^3-3\*x^2+2\*x, x=0..1) ; 

> rightsum(sqrt(x)+2, x=1..4, 60) ; 

> g := n->rightsum(sqrt(x)+2, x+1..4, n) ; /Users/wzheng8/Desktop/Screen Shot 2016-09-10 at 10.26.12 AM.png

> Limit(g(n), n=infinity) ; 

Lesson 2 Numerical Integration

TOPIC: two other method to approximate area

MATERIAL:

The Trapezoid Sum:

with(student) :

trapezoid(expr, x=a..b, n) ;

* trapezoid sum underestimates while middle sum overestimates

Simpson’s Rule:

simpson(expr, x=a..b, n) ;

* accuracy: midpoint > trapezoid > simpson > right/left point

Error Analysis:

error estimate K/n^2. 当n=1时估计K值

comparison of the five area approximation commands

Lesson 3 Differential Equations and Euler’s Method

TOPIC: draw direction field, Euler’s rule

MATERIAL:

Differential Equation:

two ways:

dep := diff(y(x), x);

dexp :=

plotting direction fields:

package: with(DEtools):

DEplot (*diff(y(x), x)*, y(x), x=a..b, y=c..d)

Euler’s Method:

loop command

for varname (from # by #) to #

do

a[i] = command;

command 2;

od;

sequence command returns all the results of the repeated operation

seq(i, i = # .. #) or seq(a[i], i = # .. #)

Exact Solution to Differential Equation

dsolve(equation)

EXAMPLE:

f := x-> x^2;

df := D(f);

deq := diff(y(x), x) = 2\*x + y(x);

dexp := 2\*x + y(x);

dep := diff(y(x), x) = dexp;

dsolve(deq);

with(DEtools):

DEplot(dep, y(x), x= -2 .. 2, y= -2 .. 2);

xpt := -1;

ypt := 1.4;

newpoint[0] := [xpt, ypt];

deltax := .2

for a to 10

do

slope := 2\*xpt + ypt;

ypt := ypt + deltax\*slope;

xpt := xpt +deltax;

newpoint[a] := [xpt, ypt];

od;

lostofpoints := [seq(newpoint[i], i= 0.. 10)];

approx2 := plot(listofpoints, color=blue):

dirfield2 := DEplot(deq, y(x), x = -2 .. 2, y = 0 .. 4):

display([approx2, dirfield2]);

Lesson 4 Logistic and Linear Differential Equations

TOPICS: logistic differential equation, 2nd order linear differential equation, amplitude, period, phase shift

MATERIAL:

Logistic Equation

dsolve( (*1/P(t)) \* diff(P(t),t) = 0.08 \* (1-P(t) / 1000)* );

dsolve( {*(1/P(t)) \* diff(P(t),t) = 0.08 \* (1-P(t) / 1000)* , P(0)=100} , P(t));

\* {} with restriction

Simple Harmonic Motion

dsolve ( *{ diff(s(t),t,t) = -9 \* s(t), s(0)=0, D(s)(0)=6}* , s(t) );

sol := rhs(%);

Damping

External Forcing

EXAMPLE:

Lesson 5 Taylor Approximations to Functions

TOPICS:

MATERIAL:

Logistic Equation

Simple Harmonic Motion

Damping

External Forcing

EXAMPLE:

MA 242

Lesson 1 Vectors

TOPIC: vector calculations

MATERIAL:

Defining a Vector:

A := [*x, y, z*] or subs({x = 1, y = 2, z = 3}, A)

Vec : = [2, 3, 4] – [3, 4, 5]

Linear Algebra package - linalg

Length of a Vector:

norm(*vector*, 2); --- Euclidean norm

Inner(Dots) and Cross Product:

innerprod(A, B);

crossprod(A, B);

area: norm(crossprod(A, B), 2);

type(crossprod(A, B), list) returns false – checking the type of the value

convert(crossprod(A, B), list) converts non-list to list

Angle between Two Vectors:

angle(A, B);

EXAMPLE:

Find a vector that is orthogonal to the plane containing the three pointsP=(1,2,3), Q=(4,5,6) and R=(-2,7,11). 

vec(pq):=[4,5,6]-[1,2,3];

vec(pr):=[-2,7,11]-[1,2,3];

orthogonal\_vector:=convert(crossprod(vec(pq),vec(pr)),list);