Limits

Differentiation
- Explicit differentiation
- Implicit differentiation

Integration
- Integrals
- Numerical integration

Summary

MAT 305: Mathematical Computing

Differentiation and Integration in Sage

John Perry

University of Southern Mississippi

Fall 2011

Outline

1. Limits
2. Differentiation
   - Explicit differentiation
   - Implicit differentiation
3. Integration
   - Integrals
   - Numerical integration
4. Summary

Maxima, Sympy, GSL

Maxima: symbolic Calculus

- Storied history
- Written in LISP
- Sometimes buggy in non-obvious ways

Sympy: symbolic Calculus

- Far more recent
- Written in Python
- Often slower than Maxima
- Eventual replacement for Maxima?

GSL: numerical Calculus

- GNU Scientific Library

Outline

1. Limits
2. Differentiation
   - Explicit differentiation
   - Implicit differentiation
3. Integration
   - Integrals
   - Numerical integration
4. Summary

The `limit()` command

`limit(` $f(x)$, `x=`a, *direction*`)` where

- $f(x)$ is a function in x
- $a \in \mathbb{R}$
- *direction* is optional, but if used has form
    - `dir='left'` or
    - `dir='right'`

Examples

```
sage: limit(x**2-1,x=4)
15
```

```
sage: limit(x/abs(x),x=0)
und
```

(Translation: “undefined”)

```
sage: limit(x/abs(x),x=0,dir='right')
1
```

```
sage: limit(x/abs(x),x=0,dir='left')
-1
```

```
sage: limit(sin(1/x),x=0)
ind
```

(Translation: “indeterminate, but bounded”)

Examples with infinite limits

```
sage: limit(e**(-x),x=infinity)
0
```

```
sage: limit((1+1/x)**x,x=infinity)
e
```

(An indeterminate form!)

```
sage: limit((3*x**2-1)/(2*x**2+cos(x)),x=infinity)
3/2
```

```
sage: limit(ln(x)/x,x=infinity)
0
```

(Another indeterminate form!)

```
sage: limit(x/ln(x),x=infinity)
+Infinity
```

Outline

1. Limits
2. Differentiation
   - Explicit differentiation
   - Implicit differentiation
3. Integration
   - Integrals
   - Numerical integration
4. Summary

Limits

Differentiation

Explicit differentiation

Implicit differentiation

Integration

Integrals

Numerical integration

Summary

The `diff()` command

`diff(` $f(x)$, x, n`)` where

- $f(x)$ is a function of x
- differentiate f with respect to x
    - *"semi-optional"*: mandatory if f has other variables
    - e.g., partial differentiation, or unknown constants
- *(optional)* compute the nth derivative of $f(x)$

Examples

```
sage: diff(e**x)
e^x
sage: diff(x**10, 5)
30240*x^5
sage: diff(sin(x), 99)
-cos(x)
sage: var('y')
y
sage: diff(x**2+y**2-1)
```

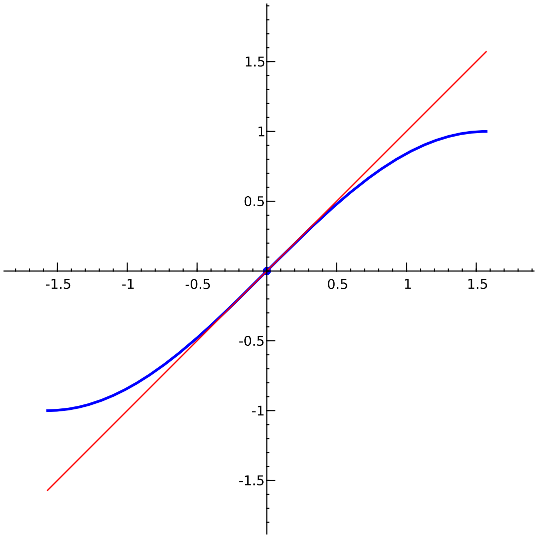
. . . output cut. . .

```
ValueError: No differentiation variable specified.
sage: diff(x**2+y**2-1, x)
2*x
```

1 picture = 1000 words

```
sage: f(x) = sin(x)

sage: df(x) = diff(f)

sage: m0 = df(0)

sage: p0 = point((0,f(0)),pointsize=30)

sage: fplot = plot(f,xmin=-pi/2,xmax=pi/2,
                     thickness=2)

sage: tan_line = plot(m0*(x-0)+f(0),xmin=-pi/2,
                        xmax=pi/2,rgbcolor=(1,0,0))

sage: show(p0+fplot+tan_line,aspect_ratio=1)
```

1 picture = 1000 words



The `implicit_diff()` command

There is no `implicit_diff()` command. To differentiate implicitly,

- define `yvar` as a variable using the `var()` command;
- define `yf` as an implicit function of x using the `function()` command;
- move everything to one side (as in implicit plots);
- differentiate the non-zero side of the equation; and
- `solve()` for `diff(yf)`

Example

```
sage: y = function('y',x)
sage: y
y(x)
```

(…so y is recognized as a function of x)

```
sage: diff(y)
D[0](y)(x)
```

(Sage's $\frac{dy}{dx}$)

```
sage: expr = x**2 + y**2 - 1
sage: diff(expr)
2*y(x)*D[0](y)(x) + 2*x
sage: deriv = diff(expr)
sage: solve(deriv,diff(y))
[D[0](y)(x) == -x/y(x)]
```

…*that is,* $y'(x) = -\frac{x}{y}$.

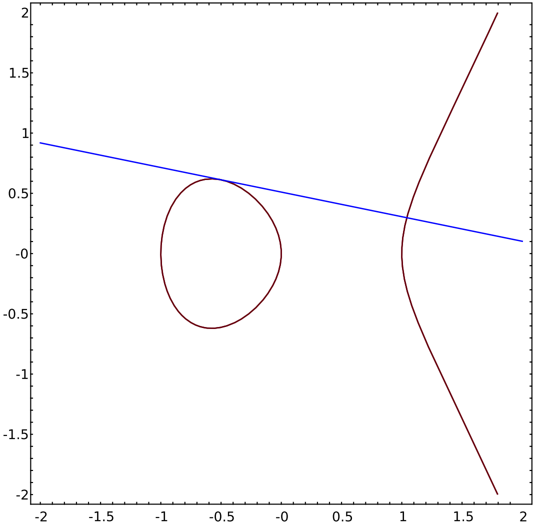
Use computer memory, not yours

```
sage: y = function('y',x)

sage: yprime = diff(y)

sage: deriv = diff(x**2+y**2-1)

sage: solve(deriv,yprime)
[D[0](y)(x) == -x/y(x)]
```

1 picture = 1000 words

```
sage: yvar = var('y')
sage: yf = function('y',x)
sage: yprime = diff(yf)
sage: f = yf**2 - x**3 + x
sage: df = diff(f)
sage: solve(df,yprime)
[D[0](y)(x) == 1/2*(3*x^2 - 1)/y(x)]
sage: mx(x,yvar) = 1/2*(3*x**2 - 1)/yvar
sage: fplot = implicit_plot(yvar**2-x**3+x,
                (x,-2,2),(yvar,-2,2),color='red')
sage: tan_line = plot(mx(-0.5,sqrt(3/8))
                        *(x+0.5)+sqrt(3/8),
                        xmin=-2,xmax=2)
sage: show(fplot+tan_line,aspect_ratio=1)
```

1 picture = 1000 words



Outline

1. Limits
2. Differentiation
   - Explicit differentiation
   - Implicit differentiation
3. Integration
   - Integrals
   - Numerical integration
4. Summary

The `integral()` command

`integral(` $f(x)$ `,` x `,` $xmin$ `,` $xmax$ `)` where

- $f(x)$ is a function of the (optional) variable x
- (optional) *xmin* and *xmax* are limits of integration

Example

```
sage: integral(x**2)
1/3*x^3

sage: integral(x**2,x,0,1)
1/3

sage: integral(1/x,x,1,infinity)
...output cut...
ValueError: Integral is divergent.

sage: integral(1/x**2,x,1,infinity)
1
```

Beware the Jabberwock…

```
sage: integral(1/x**3,x,1,infinity)
```

…output cut…

```
ValueError: Integral is divergent.
```

(What the—? a Maxima bug!)

(This error should not occur in Sage after version 4.1.1)

His vorpal sword in hand…

Fortunately, Sympy works great for this integral:

```
sage:  integral(1/x**3,1,infinity,
                algorithm='sympy')
1/2
```

(Correct answer!)

Snicker snack!

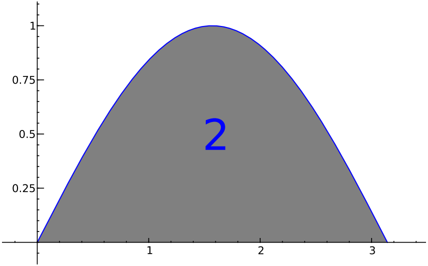
Maxima bug confirmed

`http://trac.sagemath.org/sage_trac/ticket/6420`

- Maxima 5.13.0 was correct
  - in older versions of Sage
- Bug introduced in Maxima 5.16.3
  - Sage 4.0.2–4.1.1
- Bug fixed in Maxima 5.18.1
  - Sage 4.1.2 $\longleftarrow$ Maxima 5.19

1 picture $\stackrel{?}{=}$ 1000 words

```
sage:  fplot = plot(sin(x),xmin=0,xmax=pi,
                    fill='axis')

sage:  farea = integral(sin(x),x,0,pi)

sage:  areatext = text(farea,(pi/2,0.5),fontsize=40)

sage:  fplot+areatext
```



Numerical integration: Review

Not all integrals can be simplified into elementary functions

Numerical integration: Review

Not all integrals can be simplified into elementary functions

Example

$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int e^{-x^2}\,dx$ *(Gaussian error function)*

```
sage: integral(e^(-x^2))
1/2*sqrt(pi)*erf(x)
```

Numerical integration: Review

Not all integrals can be simplified into elementary functions

Example

$$\int_{-1}^{1} \sqrt{1+\frac{4x^2}{1-x^2}}\,dx$$

(arclength of an ellipse)

```
sage: integral(sqrt(1+4*x**2/(1-x**2)),-1,1)
integrate(sqrt(-4*x^2/(x^2 - 1) + 1), x, -1, 1)
```

The `numerical_integral()` command

`numerical_integral(` $f(x)$, *xmin*, *xmax*, *options*`)` where

- $f(x)$ is a function of the defined variable x
- *xmin* and *xmax* are the limits of integration
- *options* include
  - *max_points*, the maximum number of sample points (default: 87)

Gives two results!!!

- approximation to area
- error bound
- returned as Python tuple

Example

```
sage: numerical_integral(sqrt(1+4*x**2/(1-x**2)),
                          -1,1)
(4.8442240644980235, 4.5351915253605327e-06)
```

- error bound is approximately $4.535 \times 10^{-6} \approx .000004535$
- so arclength is approximately $2 \times 4.84422 = 9.68844$

Improving the estimate

```
sage: numerical_integral(sqrt(1+4*x**2/(1-x**2)),
                               -1,1,max_points=250)
(4.8442240644980235, 4.5351915253605327e-06)
```

- error bound is approximately $4.535 \times 10^{-6} \approx .000004535$
- so arclength is approximately $2 \times 4.84422 = 9.68844$

Doesn't seem to improve :-(

Worsening the estimate

```
sage:  numerical_integral(sqrt(1+4*x**2/(1-x**2)),
                              -1,1,max_points=10)
(4.8363135584457568, 0.69875843576683905)
```

- error bound is approximately $0.7\ldots$!
- so arclength is somewhere on interval $(4.1, 5.5)$

Ouch!

Accessing only the integral

- `[i-1]` extracts the ith element of an ordered collection (list, tuple, etc.)
- first entry of result of `numerical_integral()` is the approximation

```
sage: app_int = numerical_integral(
                 sqrt(1+4*x**2/(1-x**2)),-1,1)
sage: app_int[0]
4.8442240644980235
```

Outline

1. Limits
2. Differentiation
   - Explicit differentiation
   - Implicit differentiation
3. Integration
   - Integrals
   - Numerical integration
4. Summary

Summary

- Sage relies on Maxima for symbolic integration and differentiation
    - Usually works fine
    - Eventual replacement: Sympy
    - Can call Sympy now
        - some things don't work
- Implicit differentiation requires some effort
    - define y as a function of x, not as a variable
- Numerical integration through GSL