MAT 305: Mathematical Computing

Calculus and Algebra in Sage

John Perry

University of Southern Mississippi

Spring 2017

Outline

Outline

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
```

Careful with infinite limits

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

Eh, what?

Careful with infinite limits

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

Eh, what?

> *the limit of the absolute value of the expression is positive infinity, but the limit of the expression itself is not positive infinity or negative infinity*
>
> — *Maxima documentation*

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

Outline

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
```

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

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

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

…*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)]
```

Outline

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

Assuming stuff

```
sage: var('p')
p
sage: integral(1/x**p, x, 1, infinity)
ValueError: Computation failed since Maxima
requested additional constraints; using the 'assume'
command before evaluation *may* help (example of
legal syntax is 'assume(q>0)', see 'assume' for more
details)
Is p positive, negative or zero?
```

Assuming stuff

```
sage: var('p')
p
sage: integral(1/x**p, x, 1, infinity)
ValueError: Computation failed since Maxima
requested additional constraints; using the 'assume'
command before evaluation *may* help (example of
legal syntax is 'assume(q>0)', see 'assume' for more
details)
Is p positive, negative or zero?
sage: assume(p > 1)
sage: integrate(1/x**p, x, 1, infinity)
1/(p - 1)
```

Don’t assume too much (or too little)

```
sage: assume(p <= 1)
ValueError: Assumption is inconsistent
```

Don't assume too much (or too little)

```
sage: assume(p <= 1)
ValueError: Assumption is inconsistent
sage: forget()
sage: assume(p <= 1)
sage: integrate(1/x**p, x, 1, infinity)
ValueError: Computation failed since Maxima
requested additional constraints; using the 'assume'
command before evaluation *may* help (example of
legal syntax is 'assume(q>0)', see 'assume' for more
details)
Is p positive, negative or zero?
```

Don't assume too much (or too little)

```
sage: assume(p <= 1)
ValueError: Assumption is inconsistent
sage: forget()
sage: assume(p <= 1)
sage: integrate(1/x**p, x, 1, infinity)
ValueError: Computation failed since Maxima
requested additional constraints; using the 'assume'
command before evaluation *may* help (example of
legal syntax is 'assume(q>0)', see 'assume' for more
details)
Is p positive, negative or zero?
```

The problem: $p \leq 0$ implies $\int x^q \, dx$ where $q > 0$

Don't assume too much (or too little)

```
sage: assume(p <= 1)
ValueError: Assumption is inconsistent
sage: forget()
sage: assume(p <= 1)
sage: integrate(1/x**p, x, 1, infinity)
ValueError: Computation failed since Maxima
requested additional constraints; using the 'assume'
command before evaluation *may* help (example of
legal syntax is 'assume(q>0)', see 'assume' for more
details)
Is p positive, negative or zero?
```

The problem: $p \leq 0$ implies $\int x^q \, dx$ where $q > 0$

```
sage: assume(p > 0)
sage: integrate(1/x**p, x, 1, infinity)
ValueError: Integral is divergent.
```

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

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int e^{-x^2}\, dx$$

(Gaussian error function)

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

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...!
- 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

Structure

- Mathematical operations take place in well-defined structures
- In this class, we primarily use rings and fields

"Ring"?!? *"Field"?!?*

Ring: ordinary arithmetic guaranteed, *except* division

- $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ (integers)
- $\mathbb{Q} = \{{}^{a}/_{b} : a, b \in \mathbb{Z}, b \neq 0\}$ (rationals, "quotients")
- $\mathbb{R} = \left\{\pm a_0 a_1 \ldots a_m . a_{m+1} a_{m+1} \ldots\right\}$ (reals, "lengths")
- $\mathbb{C} = \left\{a + bi : a, b \in \mathbb{R}, i^2 = -1\right\}$ (complex, "complete")

"Ring"?!? *"Field"?!?*

Ring: ordinary arithmetic guaranteed, *except* division

- $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ (integers)
- $\mathbb{Q} = \{a/b : a, b \in \mathbb{Z}, b \neq 0\}$ (rationals, "quotients")
- $\mathbb{R} = \{\pm a_0 a_1 \ldots a_m . a_{m+1} a_{m+1} \ldots\}$ (reals, "lengths")
- $\mathbb{C} = \{a + bi : a, b \in \mathbb{R}, i^2 = -1\}$ (complex, "complete")

Field: division guaranteed, too (except 0)

- $\mathbb{Q}, \mathbb{R}, \mathbb{C}$
- *not* $\mathbb{Z}$
- $\mathbb{N} = \{0, 1, 2, \ldots\}$ not even a ring

"Ring"?!? *"Field"?!?*

Ring: ordinary arithmetic guaranteed, *except* division

- $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ (integers)
- $\mathbb{Q} = \{a/b : a, b \in \mathbb{Z}, b \neq 0\}$ (rationals, "quotients")
- $\mathbb{R} = \{\pm a_0 a_1 \ldots a_m . a_{m+1} a_{m+1} \ldots\}$ (reals, "lengths")
- $\mathbb{C} = \{a + bi : a, b \in \mathbb{R}, i^2 = -1\}$ (complex, "complete")

Field: division guaranteed, too (except 0)

- $\mathbb{Q}, \mathbb{R}, \mathbb{C}$
- *not* $\mathbb{Z}$
- $\mathbb{N} = \{0, 1, 2, \ldots\}$ not even a ring

(Intuitive descriptions, not formal definitions)

Sage notation for common rings

- Integers: `ZZ` $\mathbb{Z}$
- Rationals: `QQ` $\mathbb{Q}$
- Reals: `RR` $\mathbb{R}$
  (Sage *approximates* w/53 bits precision)
- Complex: `CC` $\mathbb{C}$
  (Sage *approximates* w/53 bits precision)

Advanced rings

- Algebraic reals: `AA` $\overline{\mathbb{Q}}$
  (algebraic closure of $\mathbb{Q}$)
- Finite fields: `GF`(n) $\mathbb{Z}_n$
  (n power of prime; if not first power, specify string as name for generator)
- Finite rings: `ZZ.quo`(n) $\mathbb{Z}_n$
  (n must be an integer)
- Symbolic: `SR`
  (can use expressions with symbols as entries)

Sage guesses object’s type

```
sage: a = 1
sage: type(a)
<type 'sage.rings.integer.Integer'>
sage: b = 2/3
sage: type(b)
<type 'sage.rings.rational.Rational'>
sage: d = 1 + I
sage: type(d)
<type 'sage.symbolic.expression.Expression'>
```

Coercion

```
sage:  d = 1 + I
sage:  type(d)
<type 'sage.symbolic.expression.Expression'>
sage:  d2 = CC(1 + I)
sage:  type(d2)
<type 'sage.rings.complex_number.ComplexNumber'>
sage:  real_part(d)
1
sage:  real_part(d2)
1.00000000000000
```

Some complex commands

Suppose $z = a + bi$

`real_part(z)`	a, real part
`imag_part(z)`	b, imaginary part
`norm(z)`	$a^2 + b^2$, Euclidean norm (size)

```
sage: norm(d)
2
sage: norm(d2)
2.00000000000000
```

Modular arithmetic

Remainders v. ring arithmetic

```
sage: Z_23 = ZZ.quo(23)
sage: a = Z_23(5)
sage: a*a
2
sage: a**22
1
```

$\mathbb{Z}_{23}$

$5 \times 5 = 25$, remainder by 23

famous result

Why would you want this?

```
sage: 5**(2**127 - 2) % (2**127 - 1)
RuntimeError: exponent must be at most
9223372036854775807
```

$2^{127} - 1$ is prime

$2 \times 4 + 3 \times 6 = 26$, remainder by 7

Why would you want this?

```
sage: 5**(2**127 - 2) % (2**127 - 1)
RuntimeError: exponent must be at most
9223372036854775807
sage: R = ZZ.quo(2**127 - 1)
sage: a = R(5)
sage: a**(2^127 - 2)
1
```

$2^{127} - 1$ is prime

$2 \times 4 + 3 \times 6 = 26$, remainder by 7

$\mathbb{Z}_{2^{127}-1}$

famous result, again

Outline

Summary

- Sage can do basic calculus & algebra
  - usually works fine
  - may need to supply assumptions
  - bugs can appear; *think* about answers
- Implicit differentiation requires some effort
  - define y as a function of x, not as a variable
- Numerical integration possible