Derivace funkce jedné proměnné

Robert Mařík

27. listopadu 2010

1 Výpočet derivace

K výpočtu derivace slouží příkaz `diff`. Používáme jej ve tvaru `diff(funkce, proměnná)` a výsledkem je symbolický výraz. Je možno též použít zápis `funkce.diff(proměnná)`. Obsahuje-li funkce jenom jednu proměnnou, stačí psát `funkce.diff()`. Výsledkem je potom funkce.

Sage code
```
diff(exp(-x)*sin(x),x)
```

$-e^{(-x)} \sin(x) + e^{(-x)} \cos(x)$

Sage code
```
(exp(-x)*sin(x)).diff(x)
```

$-e^{(-x)} \sin(x) + e^{(-x)} \cos(x)$

Sage code
```
f(x)=sin(2*x)  # definujeme funkci f
f
```

$x \mapsto \sin(2x)$

Sage code
```
f.diff()   # funkce, ktera je derivaci funkce f
```

$x \mapsto 2 \cos(2x)$

Sage code
```
g=f.diff()  # funkce g je definovana jako derivace funkce f
g(x)  # derivace v bode x
```

$2 \cos(2x)$

Sage code
```
g(pi/3)     # derivace v bode x=pi/3
```

-1

Sage code
```
# totez jinak - zderivujeme /diff/ a dosadime /subs/
diff(f(x),x).subs(x=pi/3)
```

-1

Ve většině případů je vhodné derivaci upravit. K tomu je vhodné použít například `simplify_full`, `factor`, `expand`, `simplify_trig` a další funkce pro manipulaci se symbolickými výrazy - viz základy použití programu Sage.

1. příklad (derivace racionální funkce)

---

[0]Podporováno grantem FRVŠ 131/2010.

Sage code

```
f(x)=x^3/(x^2-1)^2
f
```

$$x \mapsto \frac{x^{3}}{(x^{2}-1)^{2}}$$

Sage code

```
diff(f(x),x) # derivace zlomku
```

$$\frac{-4\,x^{4}}{(x^{2}-1)^{3}} + \frac{3\,x^{2}}{(x^{2}-1)^{2}}$$

Sage code

```
(_).simplify_full() # uprava zlomku
```

$$\frac{-x^{4}+3\,x^{2}}{x^{6}-3\,x^{4}+3\,x^{2}-1}$$

Sage code

```
(_).factor()  # uprava rozkladem na soucin
```

$$\frac{-(x^{2}+3)x^{2}}{(x-1)^{3}(x+1)^{3}}$$

Pro výpočet stacionárních bodů musíme derivaci položit rovnu nule. Derivaci vypočteme příkazem `diff(f)` (protože funkce obsahuje jenom jednu proměnnou, program sám pozná že má derivovat podle x). Nulové body nalezneme příkazem `solve`. Protože jsme nezadali rovnici, program si sám doplní`solve(diff(f)==0,x)`.

Sage code

```
# nalezeni stacionarnich bodu
# v uvahu prichazeji pouze relane koreny
solve(diff(f),x)
```

$$\left[x = -i\,\sqrt{3}, x = i\,\sqrt{3}, x = 0\right]$$

Program vrátil kořeny v komplexním oboru. Pokud je chceme automaticky odebrat, můžeme použít následující trik s tvorbou seznamů a funkcemi `rhs` pro extrahování pravé strany, `imag` pro imaginární část a `n` pro numerickou aproximaci.

Sage code

```
# nalezeni stacionarnich bodu
# pouze koreny s nulovou imaginarni casti
[i for i in solve(diff(f),x) if i.rhs().imag().n()==0]
```

$$[x = 0]$$

2. příklad (derivace trigonometrické funkce)

Sage code

```
f(x)=sin(x)/cos(x)^2
f  # trigonometricka funkce
```

$$x \mapsto \frac{\sin(x)}{\cos(x)^{2}}$$

Sage code

```
diff(f(x),x)  # derivace bez upravy
```

$$\frac{2\,\sin(x)^{2}}{\cos(x)^{3}} + \frac{1}{\cos(x)}$$

Sage code

```
diff(f(x),x).factor()  # uprava zlomku
```

$$\frac{2\,\sin(x)^{2}+\cos(x)^{2}}{\cos(x)^{3}}$$

Sage code

```
diff(f(x),x).simplify_trig()  # uprava goniometrickymi identitami
```

$$\frac{\sin(x)^{2}+1}{\cos(x)^{3}}$$

1.1 Derivace vyšších řádů

Derivaci vyšších řádů vypočteme opět příkazem `diff`, je však nutno jako další parametr uvést řád derivace.

Sage code

```
diff(exp(3*x),x,2) # druha derivace
```

$9\,e^{(3\,x)}$

3. příklad (druhá derivace)

Sage code

```
f(x)=x^2/(x-1)   # definjeme funkci
diff(f(x),x,2).factor()   # druha derivace (po uprave pomoci factor)
```

$$\frac{2}{(x-1)^3}$$

Sage code

```
diff(diff(f(x),x),x).factor()   # totez, ale derivujeme derivaci
```

$$\frac{2}{(x-1)^3}$$

2 Aplikace derivace

2.1 Taylorův polynom

Taylorův polynom funkce $f(x)$ se středem v bodě a stupně n vytvoříme v programu Sage příkazem `taylor(f(x),x,a,n)`. Pro tečnu samostatný příkaz není, protože tečna je Taylorův polynom stupně 1.

Sage code

```
taylor(sin(x),x,0,5)
```

$$\frac{1}{120}\,x^5 - \frac{1}{6}\,x^3 + x$$

Sage code

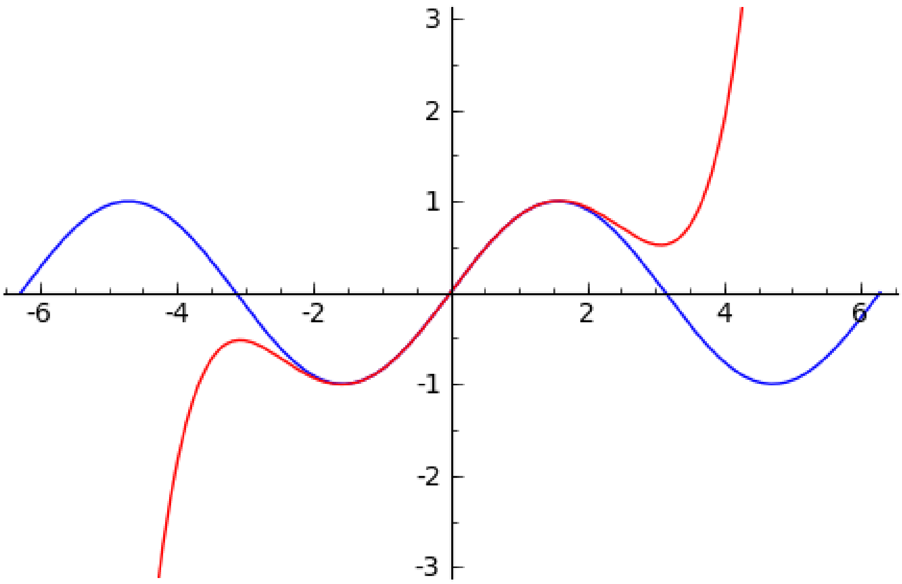
```
taylor(ln(x),x,1,5)
```

$$\frac{1}{5}\,(x-1)^5 - \frac{1}{4}\,(x-1)^4 + \frac{1}{3}\,(x-1)^3 - \frac{1}{2}\,(x-1)^2 + x - 1$$

Graf funkce a graf Taylorova polynomu můžeme snadno zakrelist do jednoho obrázku příkazem `plot`.

Sage code

```
f(x)=sin(x)                                       # definice funkce
xmin, xmax, ymin, ymax = -2*pi, 2*pi, -3, 3       # nastaveni parametru pro kresleni
a = 0                                             # stred Taylorova polynomu
g(x)=taylor(f(x),x,a,5)                           # Tayloruv polynom stupne 5
p1 = plot(f(x),(x,xmin,xmax))                     # graf funkce
p2 = plot(g(x),(x,xmin,xmax), color='red')        # graf Taylorova polynomu
show(p1+p2, ymax = ymax, ymin = ymin)             # zobrazeni obou grafu soucasne
```



V okolí počátku jsou funkční hodnoty funkce a Taylorova polynomu přibližně stejné

Sage code
```
x0 = 0.5      # bod v okoli stredu Taylorova polynomu
f(x0),g(x0)   # porovnani funkcnich hodnot
```

$(0.479425538604203, 0.479427083333333)$

Sage code
```
abs(f(x0)-g(x0))  # absolutni chyba
```

$1.54472913033166 \times 10^{-6}$

Sage code
```
(_)/f(x0)  # relativni chyba
```

$3.22204181034864 \times 10^{-6}$

2.2 Lokální extrémy, intervaly monotonie

Z lokálních extrémů jsou podezřelé nulové body derivace (*stacionární body*) a body, kde derivace neexistuje. Stacionární body najdeme tak, že položíme derivaci rovnu nule. Derivaci vypočítáme příkazem `diff`, rovnici kdy je derivace rovna nule vyřešíme příkazem `solve`. Každý z těchto příkazů má druhý parametr se jménem proměnné. Pozor na to, že příkaz solve hledá řešení v oboru komplexních čísel a případné komplexní čísla je tedy nutno z výstupu odstarnit ručně. (Nebo je možno nadefinovat proceduru, která tuto práci udělá za nás.)

Sage code
```
f(x)=x/(x^2+3)^2
f                     # definice funkce
```

$x \mapsto \frac{x}{\left(x^2+3\right)^2}$

Sage code
```
solve(diff(f(x),x)==0,x)  # stacionarni body (derivace nula)
```

$\left[x = \left(-1\right), x = 1\right]$

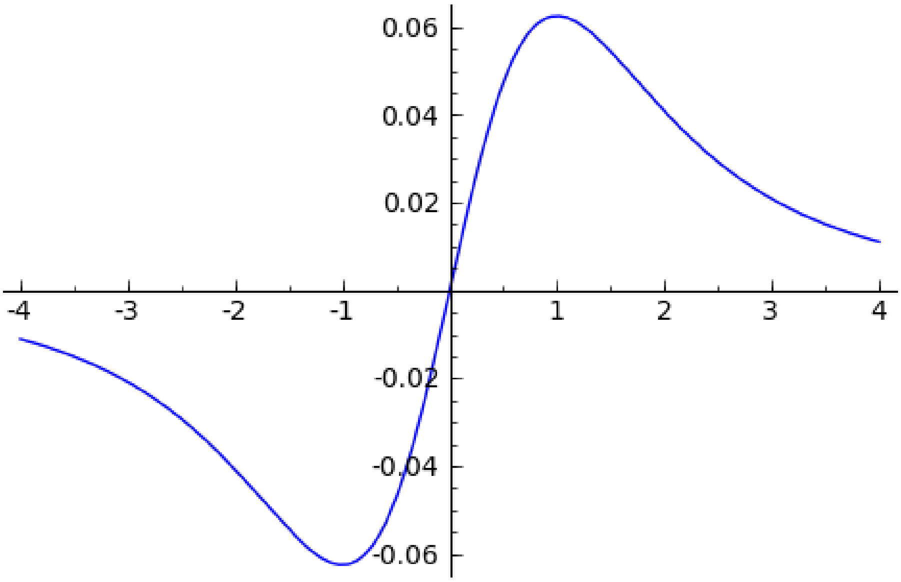
Sage code
```
solve(diff(f(x),x)>0,x)  # interval rustu (derivace kladna)
```

$\left[\left[x > \left(-1\right), x < 1\right]\right]$

Sage code
```
solve(diff(f(x),x)<0,x)   # interval klesani (derivace zaporna)
```

$[[x < (-1)], [x > 1]]$

Sage code
```
plot(f(x),(x,-4,4)) # graf funkce pro kontrolu
```



Inflexní body najdeme tam, kde je druhá derivace rovna nule nebo neexistuje. Body, kde je druhá derivace rovna nule, se nazývají *kritické body*.

Sage code
```
solve(diff(f(x),x,2)==0,x)
```

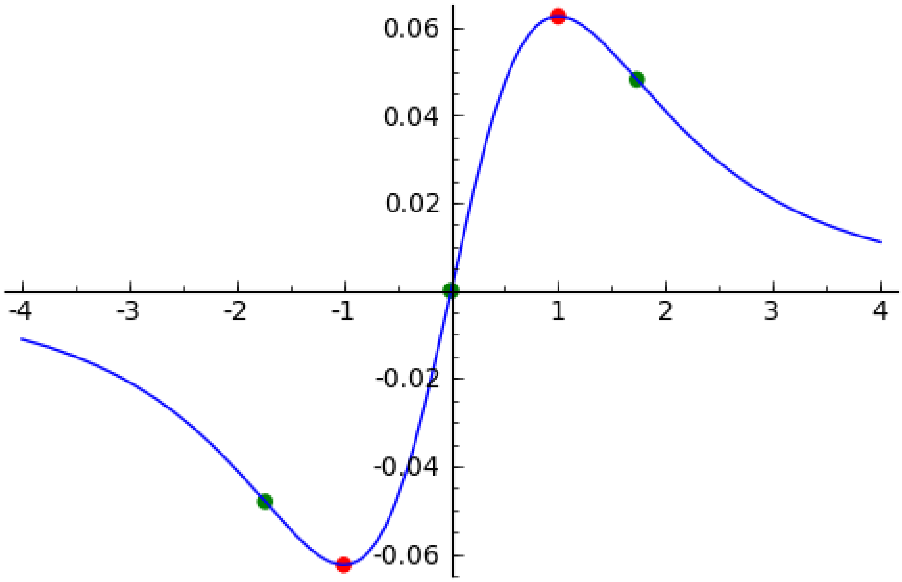
$\left[x = -\sqrt{3}, x = \sqrt{3}, x = 0\right]$

Pro zakreslení do grafu potřebujeme ke všem důležitým bodům vypočítat funkční hodnotu. Probereme seznam z výstupu příkazu `solve`, z každé položky vezmeme jenom pravou stranu příkazem `rhs` a tu dosadíme do funkce f. Výstupem je seznam uspořádaných dvojic tvaru $(a, f(a))$, kde a je buď stacionární bod nebo kritický bod.

Sage code
```
stac_body = [(i.rhs(),f(i.rhs())) \
         for i in solve(diff(f(x),x)==0,x) if i.rhs().imag().n()==0]
krit_body = [(i.rhs(),f(i.rhs())) \
         for i in solve(diff(f(x),x,2)==0,x) if i.rhs().imag().n()==0]
stac_body, krit_body
```

$$\left(\left[\left(-1, -\frac{1}{16}\right), \left(1, \frac{1}{16}\right)\right], \left[\left(-\sqrt{3}, -\frac{1}{36}\sqrt{3}\right), \left(\sqrt{3}, \frac{1}{36}\sqrt{3}\right), (0, 0)\right]\right)$$

Sage code
```
p = plot(f(x),(x,-4,4))                        # graf funkce
p += point2d(stac_body, color='red', size=30)    # pridame stacionarni body
p += point2d(krit_body, color='green', size=30) # pridame kriticke body
p
```



Pokusíme se vše automatizovat.

Sage code

```
def prubeh(f,(x,xmin,xmax),ymin=None, ymax=None):
    r"""
    Procedura nakresli graf funkce f na intervalu xmax, xmin, vypocte nulove, stacionarni a
    kriticke body.

    Je mozno zadat nepovinne parametry ymin a ymax pro rozsah hodnot na svisle ose.
    """
    P = plot(f,(x,xmin,xmax))
    html(r"<h3>Funkce: $ y=%s$</h3>"%latex(f))
    html(r"1. derivace: $ y'=%s$"%latex(factor(diff(f,x))))
    html(r"2. derivace: $ y'=%s$"%latex(factor(diff(f,x,2))))

    _temp = solve(f,x, solution_dict=True)
    _nulove_body = [i[x] for i in _temp if i[x].imag().n() == 0]

    _temp = solve(diff(f,x),x, solution_dict=True)
    _stac_body = [i[x] for i in _temp if i[x].imag().n() == 0]

    _temp = solve(diff(f,x,2),x, solution_dict=True)
    _krit_body = [i[x] for i in _temp if i[x].imag().n() == 0]

    if len(_stac_body)>0:
        P += point2d([(i,f(x=i)) for i in _stac_body], color='red', size=30, zorder=8)
    if len(_krit_body)>0:
        P += point2d([(i,f(x=i)) for i in _krit_body], color='yellow', size=15, zorder=9)

    html(r"Nulové body: $%s$"%latex(_nulove_body))
    html(r"Stacionární body: $%s$ (v grafu červeně)"%latex(_stac_body))
    html(r"Kritické body: $%s$ (v grafu žlutě)"%latex(_krit_body))
    if P.ymax()>500 and ymax is None:
        ymax = 20
    if P.ymin()<-500 and ymin is None:
        ymin = -20
    show(P,ymax=ymax,ymin=ymin)
```

Použití je jednoduché ...

Sage code

```
prubeh(x^3/(x+1),(x,-4,4))
```

2.2.1 Funkce: $y = \frac{x^3}{x+1}$

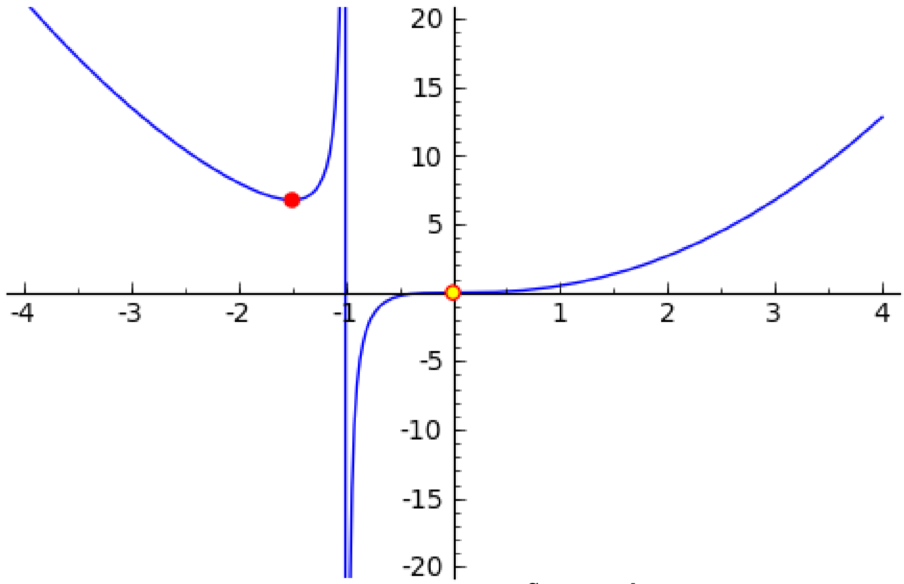
1. derivace: $y' = \frac{(2\,x+3)x^2}{(x+1)^2}$

2. derivace: $y' = \frac{2\,(x^2+3\,x+3)x}{(x+1)^3}$

Nulové body: $[0]$

Stacionární body: $\left[-\frac{3}{2}, 0\right]$ (v grafu červeně)

Kritické body: $[0]$ (v grafu žlutě)



Sage code

```
prubeh(x^2*(x-2),(x,-4,4),ymax=2,ymin=-2)
```

2.2.2 Funkce: $y = (x-2)x^2$

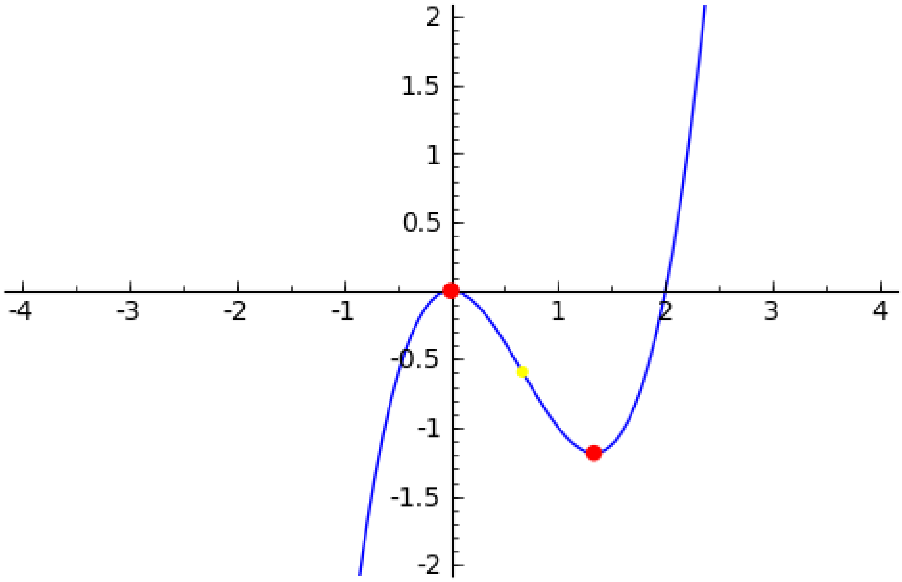
1. derivace: $y' = (3\,x-4)x$

2. derivace: $y' = 6\,x-4$

Nulové body: $[0, 2]$

Stacionární body: $\left[\frac{4}{3}, 0\right]$ (v grafu červeně)

Kritické body: $\left[\frac{2}{3}\right]$ (v grafu žlutě)



2
1.5
1
0.5
-4
-3
-2
-1
1
2
3
4
-0.5
-1
-1.5
-2