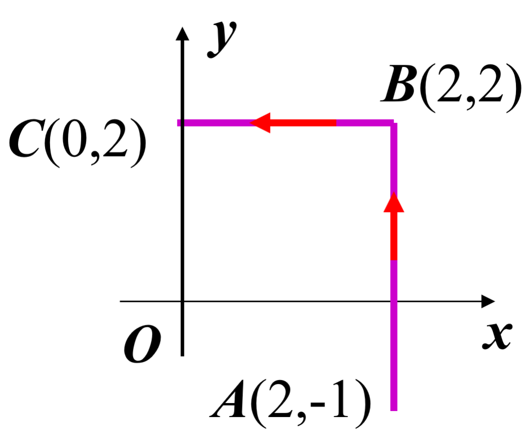
高数下习题

线面积分的计算

> *先判断是否与积分路径有关*
>
> - $$\frac{\partial P}{\partial y} = \frac{\partial Q}{\partial x}$$ $$I = \oint_L Pdx + Qdy = 0$$
> - $$\frac{\partial P}{\partial y} \neq \frac{\partial Q}{\partial x}$$ $$I = \pm \iint_D (\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}) = 0$$
> - $$A = \frac{1}{2}\oint_L x\,dy - y\,dx$$

1. 计算$\int_L (x^2-2xy)dx + (y^2-2xy)dy$，$L$从点$A(2,-1)$到$B(2,2)$再到$C(0,2)$的折线

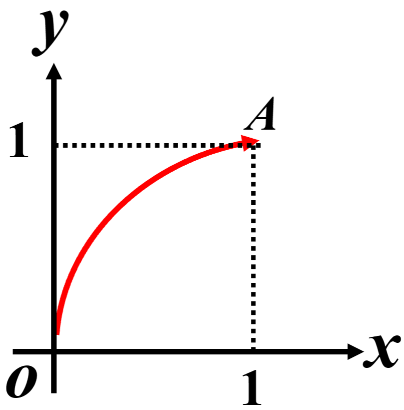
   (最基本对曲线积分的算法)



```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x, y = symbols('x y')

eq1 = y**2 - 2 * x * y
eq2 = x**2 - 2 * x * y

res = integrate(eq1, (y, -1, 2)).subs(x, 2) + integrate(eq2, (x, 2, 0)).subs(y, 2)
pprint(res.simplify())

# 7/3
```

2. 计算$I=\int_L (x^2+2xy)dx+(x^2+y^4)dy$，其中$L$为由点$O(0,0)$到点$A(1,1)$的曲线$y=\sin\frac{\pi}{2}x$

(与积分路径无关的条件使用)




```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x, y = symbols('x y')

eq1 = x**2 + 2 * x * y
eq2 = x**2 + y**4

pprint(diff(eq1, y))
print()
pprint(diff(eq2, x))
print()

```

```python
res = integrate(eq1, (x, 0, 1)).subs(y, 0) + integrate(eq2, (y, 0, 1)).subs(x, 1)

pprint(res)

# 2⋅x

# 2⋅x

# 23
# ──
# 15
```

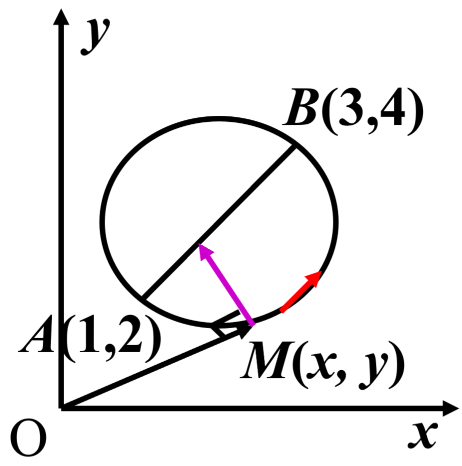
3. 计算$I=\int_L(e^x\sin y-my)dx+(e^x\cos y-m)dy$，其中$L$为由点$(a,0)$到点$(0,0)$的上半圆周$x^2+y^2=ax$，$y\geq 0$
(格林公式的应用)

```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x, y, m, a = symbols('x y m a')

eq1 = E**x * sin(y) - m * y
eq2 = E**x * cos(y) - m

pprint(diff(eq1, y).simplify())
print()
pprint(diff(eq2, x))
print()

eq3 = eq2.diff(x) - eq1.diff(y)
pprint(eq3)
print()

res = eq3 * (a / 2)**2 * pi / 2
pprint(res)

#       x
# -m + ℯ ⋅cos(y)

#  x
# ℯ ⋅cos(y)
```

```

# m

#     2
# π·a ·m
# ───────
#   8
```

4. 设平面力场的大小等于作用点到原点的距离，方向为作用点的向径方向按逆时针旋转90°，试求质点沿曲线$(x-2)^2+(y-3)^2=2$，从点$A(1,2)$按逆时针移动到点$B(3,4)$时场力所作的功

(格林公式的应用)



$$\vec{F}=(-y,x)$$

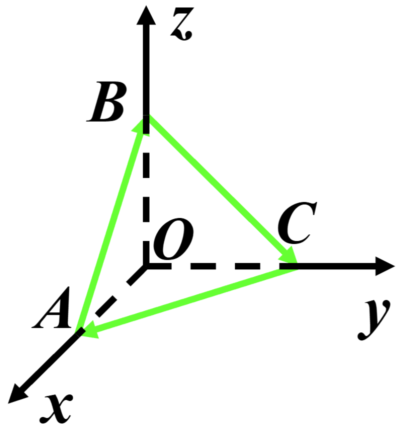
$$W=\oint_{AB}\vec{F}\cdot d\vec{S}-\int_{\vec{BA}}\vec{F}\cdot(dx,dy)$$

```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x,y=symbols('x y')

res=2*sqrt(2)**2*pi/2+integrate(-(x+1)+x,(x,1,5))

pprint(res)

# -4 + 2⋅π
```

$$A=\frac{1}{2}\oint_L x\,dy-y\,dx$$

5. 求力$\vec{F}=(y,z,x)$沿有向闭曲线Γ所作的功，其中Γ为平面$x+y+z=1$被三个坐标面所截成三角形的整个边界，从z轴正向看去沿顺时针（注意逆时针为正方向）方向

(斯托克斯公式的应用)

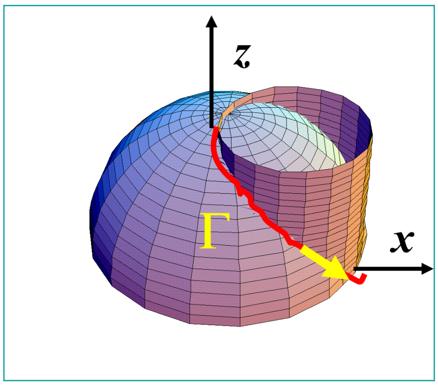


三角形区域设为：Σ，方向向量$\vec{n}=\dfrac{1}{\sqrt{3}}(1,1,1)$

$$\begin{aligned}
W &= \oint_{\Gamma} ydx+zdy+xdz \\
&= -\iint_{\Sigma}\begin{vmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ y & z & x \end{vmatrix} dS \\
&= -\frac{1}{\sqrt{3}}\iint_{\Sigma}(-3)\,dS \\
&= \sqrt{3}\iint_{Dxy} dxdy = \frac{3}{2}
\end{aligned}$$

6. 计算$\oint_{\Gamma} y^2dx+z^2dy+x^2dz$,其中$\Gamma$是球面$x^2+y^2+z^2=R^2$和柱面$x^2+y^2=Rx$的交线$(z\geq 0)$，从$z$轴正向看为逆时针方向

(换元法的应用)



令$x=\frac{R}{2}+\frac{R}{2}\cos t$，$y=\frac{R}{2}\sin t$，$(0\leq t\leq 2\pi)$

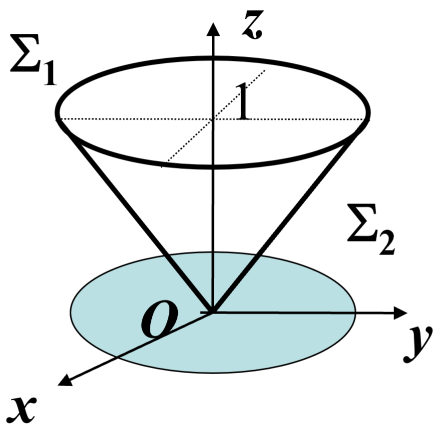
代入球面方程可得：$z=R\sin\frac{t}{2}$

```
from sympy import *
```

```python

init_printing(use_unicode=True, wrap_line=False)

t, R = symbols('t R')

x = R / 2 + R / 2 * cos(t)
y = R / 2 * sin(t)
z = R * sin(t / 2)

eq = y**2 * x.diff(t) + z**2 * y.diff(t) + x**2 * z.diff(t)
res = integrate(eq, (t, 0, 2 * pi))

pprint(res.simplify())

#      3
# -π⋅R
# ───────
#    4
```

7. 设有一物质曲面Σ是由$z=\sqrt{x^2+y^2}$及$z=1$所围立体的边界曲面，曲面的面密度函数为$f(x,y,z)=x^2+y^2$，求该曲面的质量M

(第一类曲面积分的算法)

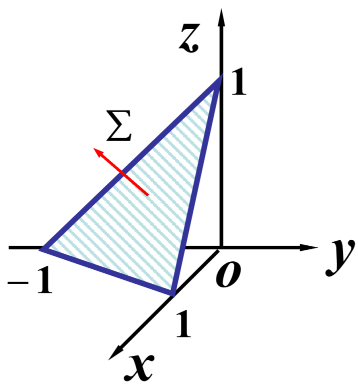



```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x, y, R, theta = symbols('x y R theta')

f = x**2 + y**2
z1 = sqrt(f)
```

```python
z2 = Symbol('1')

eq1 = f * sqrt(1 + z1.diff(x)**2 + z1.diff(y)**2)
eq2 = f * sqrt(1 + z2.diff(x)**2 + z2.diff(y)**2)

eq3 = eq1 + eq2
eq3 = eq3.subs(x, R * cos(theta)).subs(y, R * sin(theta)) * Matrix(
    [R * cos(theta), R * sin(theta)]).jacobian(Matrix([theta, R])).det()
res = integrate(eq3, (theta, 0, 2 * pi), (R, 0, 1))

pprint(res.simplify())

# π·(-√2 - 1)
# ───────────
#      2
```

8. 计算 $I=\iint_{\Sigma}[f(x,y,z)+x]\,dydz+[2f(x,y,z)+y]dzdx+[f(x,y,z)+z]dxdy$，$f(x,y,z)$为连续函数，$\Sigma$为平面$x-y+z=1$在第四象限部分的上侧

(归一法求解)



$\because$ Σ的法向量：$\vec{n}=(1,-1,1)$

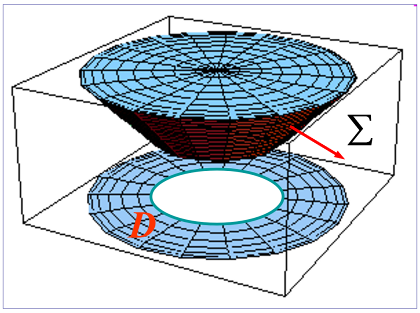
$\therefore \cos\alpha=\frac{1}{\sqrt{3}},\cos\beta=-\frac{1}{\sqrt{3}},\cos\gamma=\frac{1}{\sqrt{3}}$

$$I=\iint_{\Sigma}(\,P\frac{\cos\alpha}{\cos\gamma}+Q\frac{\cos\beta}{\cos\gamma}+R\,)\,dxdy$$

```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x, y = symbols('x y')
```

```python
f = Function('f')

z = 1 - x + y

eq1 = f(x, y, z) + x
eq2 = 2 * f(x, y, z) + y
eq3 = f(x, y, z) + z

res = integrate(eq1 - eq2 + eq3, (y, -x - 1, 0), (x, 0, 1))

pprint(res)

# 3/2
```

9. 计算 $I=\iint_{\Sigma} ydydz-xdzdx+z^2dxdy$，其中 Σ 为曲面 $z=\sqrt{x^2+y^2}$被平面$z=1,z=2$所截部分的外侧(归一法求解)

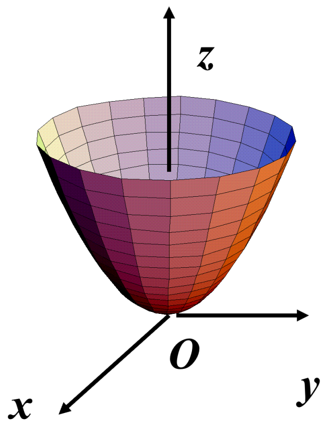



```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x, y, R, theta = symbols('x y R theta')
z = sqrt(x**2 + y**2)

cos_alpha = z.diff(x) / sqrt(1 + z.diff(x)**2 + z.diff(y)**2)
cos_beta = z.diff(y) / sqrt(1 + z.diff(x)**2 + z.diff(y)**2)
cos_gamma = -1 / sqrt(1 + z.diff(x)**2 + z.diff(y)**2)

eq1 = y
eq2 = -x
eq3 = z**2

```

```python
eq4 = eq1 * cos_alpha / cos_gamma + eq2 * cos_beta / cos_gamma + eq3

eq4 = eq4.subs(x, R * cos(theta)).subs(y, R * sin(theta)) * Matrix(
    [R * cos(theta), R * sin(theta)]).jacobian(Matrix([theta, R])).det()

res = integrate(eq4, (theta, 0, 2 * pi), (R, 1, 2))

pprint(res)

# -15⋅π
# ─────
#   2
```

10. 计算$\iint_{\Sigma} x^2y^2z\,dxdy$，$\Sigma$：曲面$z=x^2+y^2(z\leq 1)$下侧

(高斯公式)



添加曲面Σ'：$z=1(x^2+y^2\leq 1)$上侧

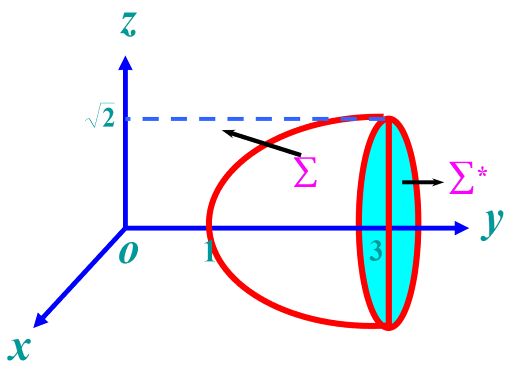
$$
\begin{aligned}
I &= \iint_{\Sigma+\Sigma'} - \iint_{\Sigma'} \\
&= \iint_{\Sigma+\Sigma'} x^2y^2z\,dxdy - \iint_{\Sigma'} x^2y^2z\,dxdy \\
&= \iiint_{\Omega} x^2y^2\,dV - \iint_{D} x^2y^2\,dxdy
\end{aligned}
$$

```python
from sympy import *

init_printing(use_unicode=True, wrap_line=False)

x, y, z = symbols('x y z')

eq = x**2 * y**2
```

```python

res = Integral(eq, (z, x**2 + y**2, 1), (x, -sqrt(1 - y**2), sqrt(1 - y**2)),
               (y, -1, 1)) - Integral(eq, (x, -sqrt(1 - y**2), sqrt(1 - y**2)),
                                      (y, -1, 1))

pprint(res)
print()
pprint(res.doit().simplify())

#            --------                       --------
#           /      2                       /      2
#     1   \/   1 - y                  1   \/   1 - y      1
#     /        /                     /        /         /
#    |        |                     |        |         |
#    |        |       2  2          |        |         | 2  2
# -  |        |      x *y  dx dy +  |        |         |x *y  dz dx dy
#    |        |                     |        |         |
#   /        /                     /        /         /
#   -1      --------              -1      --------  2   2
#          /      2                       /      2  x  + y
#       -\/  1 - y                     -\/  1 - y

# -  π
# ----
#  32
```

11. 计算$I = \iint_{\Sigma} (8y+1)x\, dydz + 2(1-y^2)\, dzdx - 4yz\, dxdy$，其中$\Sigma$是由曲线$\begin{cases} z = \sqrt{y-1} \\ x = 1 \end{cases}(1 \leq y \leq 3)$绕$y$轴旋转一周所成的曲面，它的法向量与$y$轴正向的夹角恒大于$\dfrac{\pi}{2}$

(高斯公式)



添加曲面Σ′： $y=3(x^2+z^2\le 2)$的外侧

$$
\begin{aligned}
I &= \iint_{\Sigma+\Sigma'} - \iint_{\Sigma}' \\
&= \iint_{\Sigma+\Sigma'} [(8y+1)x\,dydz + 2(1-y^2)\,dzdx - 4yz\,dxdy\,] - \iint_{D} 2(1-3^2)\,dzdx
\end{aligned}
$$

```python
from sympy import *

init_printing(use_unicode=False, wrap_line=False)

x, y, z = symbols('x y z')

eq1 = (8 * y + 1) * x
eq2 = 2 * (1 - y**2)
eq3 = -4 * y * z

eq_ = eq1.diff(x) + eq2.diff(y) + eq3.diff(z)

res = Integral(eq_, (y, 1 + x**2 + z**2, 3),
               (x, -sqrt(2 - z**2), sqrt(2 - z**2)),
               (z, -sqrt(2), sqrt(2))) + 32 * pi

pprint(res)
print()
pprint(res.doit().simplify())

#             ________
#     ___    /      2
#   \/ 2   \/  2 - z        3
#     /         /           /
#    |         |           |
#    |         |           |      1 dy dx dz + 32·π
#    |         |           |
#   /         /           /
#    ___     ________  2    2
```

```
# -\/ 2     /      2  x  + z  + 1
#        -\/  2 - z

# 34· π
```

无穷级数

> *性质:*
>
> - *收敛+收敛$\Longrightarrow$收敛*
> - *收敛+发散$\Longrightarrow$发散*
> - *发散+发散$\Longrightarrow$未必发散*
> - *收敛级数 加括号后所成的级数仍收敛于原7级数的和，但收敛级数去括号后所成的级数不一定收敛*
>
> *收敛级数的必要条件:*
>
> *设收敛级数*$S=\sum_{n=1}^{\infty}u_n$*，则必有*$\lim_{n\to\infty}n_n=0$
>
> *柯西审联原理（用做证明题）：*
>
> *级数*$\sum_{n=1}^{\infty}u_n$*收敛*$\Longleftrightarrow\forall\varepsilon>0$*，*$\exists N\in N_+$*，当*$n>N$*时，对*$\forall p\in N_+$*，有*$|n_{n+1}+u_{n+2}+\cdots+u_{n+p}|<\varepsilon$
>
> P*收敛级数:*
>
> $$\sum_{n=1}^{\infty}\frac{1}{n^P},\quad(P>1)$$
>
> *调和级数：*
>
> $$\sum_{n=1}^{\infty}\frac{1}{n}$$
>
> ***调和级数和P收敛级数是两个常用的比较级数***
>
> *比较审敛法:*
>
> *设两正向级数*$\sum_{n=1}^{\infty}u_n$*，*$\sum_{n=1}^{\infty}v_n$*满足*$\lim_{n\to\infty}\frac{u_n}{v_n}=l$*，则有*
>
> - *当*$0<l<\infty$*时，两个级数同时收敛或发散*
> - *当*$l=0$*时，且*$\sum_{n=1}^{\infty}v_n$*收敛时，*$\sum_{n=1}^{\infty}u_n$*也收敛*

> - 当$l=\infty$且$\sum\limits_{n=1}^{\infty} v_n$，发散时，$\sum\limits_{n=1}^{\infty} u_n$也发散
>
> *比值审敛法：*
>
> *设*$\sum u_n$*为正向级数，且*$\lim\limits_{n\to\infty}\frac{u_{n+1}}{u_n}=\rho$*，则：*
>
> - 当$\rho<1$时，级数收敛
> - 当$\rho>1$或$\rho=\infty$时，级数发散
>
> *注意：当*$\lim\limits_{n\to\infty}\frac{u_{n+1}}{n}=1$*时，级数可能收敛也可能发散*

1. 求$\sum\limits_{n=1}^{\infty}\frac{1}{n^3+3n^2+2n}$的值

$$
\begin{aligned}
&\frac{1}{n^3+3n^2+2n}\\
=&\frac{1}{n(n+1)(n+2)}\\
=&\frac{1}{2}\frac{(n+2)-n}{n(n+1)(n+2)}\\
=&\frac{1}{2}\left[\frac{1}{n(n+1)}-\frac{1}{(n+1)(n+2)}\right]
\end{aligned}
$$

```python
from sympy import *

init_printing(use_unicode=False, wrap_line=False)

n = symbols("n")

eq = 1 / (n**3 + 3 * n**2 + 2 * n)

res = Sum(eq, (n, 1, oo))

pprint(res)
print()
pprint(res.doit().simplify())

#   oo
# ____
# \   `
#  \           1
#   \   ---------------
#   /    3      2
#  /    n  + 3*n  + 2*n
```

```
# /___,
# n = 1

# 1/4
```

2. 计算$S_n=\sum_{n=1}^{\infty}\frac{2n-1}{2^n}$

$$
\begin{aligned}
&S_n-\frac{1}{2}S_n\\
=&\left(\frac{1}{2}+\frac{2}{2^2}+\frac{5}{2^3}+\cdots+\frac{2n-1}{2^n}\right)-\left(\frac{1}{2^2}+\frac{2}{2^3}+\frac{5}{2^4}+\cdots+\frac{2n-1}{2^{n+1}}\right)\\
=&\frac{1}{2}+\frac{1}{2}+\frac{1}{2^2}+\frac{1}{2^3}+\cdots+\frac{1}{2^{n-1}}-\frac{2n-1}{2^{n+1}}
\end{aligned}
$$

```python
from sympy import *

init_printing(use_unicode=False, wrap_line=False)

n = symbols("n")

eq = (2 * n - 1) / 2**n

res = Sum(eq, (n, 1, oo))

pprint(res.simplify())
print()
pprint(res.doit().simplify())

#   oo
#  ___
#  \  `
#   \     -n
#   /    2  *(2*n - 1)
#  /__,
# n = 1

# 3
```

3. 利用柯西审敛原理判别级数$\sum_{n=1}^{\infty}\frac{1}{n^2}$敛散性

对$\forall p\in N_+$，有：

$|u_(n+1)+u_{n+2}+\cdots+u_{n+p}|$

$=\frac{1}{(n+1)^2}+\frac{1}{(n+2)^2}+\cdots+\frac{1}{(n+p)^2}$

$<(\frac{n}{n+1}-\frac{n+1}{n+2})+\cdots+(\frac{n+p-1}{n+p})$

$=(\frac{1}{n}-\frac{1}{n+1})+(\frac{1}{n+1}-\frac{1}{n+2}+\cdots+(\frac{1}{(n+p-1)}-\frac{1}{n+p})$

$=\frac{1}{n}-\frac{1}{n+p}<\frac{1}{n}$

4. 判别级数$\sum_{n=1}^{\infty}\sin\frac{1}{n}$的敛散性

（比较审敛法，调和级数）

$\because$

$$\lim_{n\to\infty}\frac{\sin\frac{1}{n}}{\frac{1}{n}}=1,\quad \sum_{n=1}^{\infty}\frac{1}{n}\text{发散}$$

$\therefore\ \sum_{n=1}^{\infty}\sin\frac{1}{n}$ 发散

5. 判别级数$\sum_{n=1}^{\infty}\ln(1+\frac{1}{n^2})$的敛散性

（比较审敛法，P收敛级数）

$\because$

$$\lim_{n\to\infty}\frac{\ln(1+\frac{1}{n^2})}{\frac{1}{n^2}}=1,\quad \sum_{n=1}^{\infty}\frac{1}{n^2}\text{收敛}$$

$\therefore\sum_{n=1}^{\infty}\ln(1+\frac{1}{n^2})$收敛

6. 判别级数$\sum_{n=1}^{\infty}\frac{1}{3^n-n}$的敛散性

（比较收敛法，等比级数）

$\because$

$$\lim_{n\to\infty}\frac{\frac{1}{3^n-n}}{\frac{1}{3^n}}=\lim_{n\to\infty}\frac{1}{1-\frac{n}{3^n}}=1,\quad \sum_{n=1}^{\infty}\frac{1}{3^n}\text{收敛}$$

$\therefore\frac{1}{3^n-n}$收敛

7. 判别$\sum_{n=1}^{\infty}\frac{3n-2}{n^3-2n^2+n}$收敛性

$\because$

$$\lim_{n=1} \frac{\frac{3n-2}{n^3-2n^2+n}}{\frac{1}{n^2}} = 3, \quad \sum_{n=1}^{\infty} \frac{1}{n^2}\text{收敛}$$

$$\therefore \sum_{n=1}^{\infty} \frac{3n-2}{n^3-2n^2+n}\text{收敛}$$