Sympy+-+Numpy

January 23, 2018

1 Usando Simpy

Simpy es un módulo que nos permite el manejo de computacion matematica y simbólica Entres sus funciones destaca: 1. Simplificar Terminos 2. Hacer Derivadas 3. Hacer integrales con y sin intervalos 4. Hacer Matrices

1.1 Simplificacion

```
In [8]: from sympy import *
        x, y, z = symbols('x y z')
        init_printing(use_unicode = True)
        pitagoras = simplify(sin(x)**2 + cos(x)**2)
        print(pitagoras)
        a = simplify((x**3+x**2-x-1)/(x**2+2*x+1))
        print(a)
        b = simplify(gamma(x)/gamma(x-2))
        #print(b)
```

```
1
x - 1
```

1.2 Expand

Con esta función expandimos las funciones polionominales

```
In [15]: termino = expand((x+1)**2)
         print(termino)
         termino_2 = expand((x+2)*(x-3))
         print(termino_2)
```

```
x**2 + 2*x + 1
x**2 - x - 6
```

1.3 Factor

Esta funcion es lo opuesto a "Expandir", esta funcion reduce los terminos hasta un valor irreducibe

```
In [18]: factor(x**3 - x**2 + x-1)
         factor(x**2*z + 4*x*y*z + 4*y**2*z)
```

Out[18]:

$$z\,(x+2y)^2$$

1.4 Cancel

La funcion cancel tomará cualquier función racional y la colocará en la forma estándar (canónica)

```
In [29]: expresion = 1/x + (3*x/2-2)/(x-4)
         print("Expresion normal")
         #print(expresion)
         print("Expresion usando Cancel")
         cancel(expresion)
         #print(b)
```

```
Expresion normal
Expresion usando Cancel
```

Out[29]:

$$\frac{3x^2-2x-8}{2x^2-8x}$$

```
In [34]: funcion = (x*y**2 - 2*x*y*z + x*z**2 + y**2 - 2*y*z + z**2)/ (x**2-1)
         funcion
         cancel(funcion)
```

Out[34]:

$$\frac{1}{x-1}\left(y^2-2yz+z^2\right)$$

```
In [41]: funcion_2 = (4*x**3 + 21*x**2 + 10*x+12)/(x**4 + 5*x**3+5*x**2+4*x)
         funcion_2
         #cancel(funcion_2)
         ###Solo efectúa cambios si la funcion así lo permite
```

Out[41]:

$$\frac{4x^3+21x^2+10x+12}{x^4+5x^3+5x^2+4x}$$

1.5 Apart

La funcion apart realizará una descomposición parcial de una función racional

```
In [48]: funcion_3 = (4*x**3 +21*x**2 +10*x+12)/(x**4+5*x**3+5*x**2+4*x)
         #funcion = (x*y**2 - 2*x*y*z + x*z**2 + y**2 - 2*y*z + z**2)/ (x**2-1)
         funcion_3
         apart(funcion_3)
         #apart(funcion)
```

Out[48]:

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

1.6 Usando Derivadas

```
In [51]: diff(cos(x), x)
         diff(exp(x**2), x)
         diff(x**2, x)
```

Out[51]:

$$2x$$

```
In [52]: diff(x**4,x,3)
         diff
```

Out[52]:

$$24x$$

```
In [55]: funcion= (x**2+3)
         funcion_3 = (4*x**3 +21*x**2 +10*x+12)/(x**4+5*x**3+5*x**2+4*x)
         derivada = Derivative(funcion_3)
         derivada
```

Out[55]:

$$\frac{d}{dx}\left(\frac{4x^3+21x^2+10x+12}{x^4+5x^3+5x^2+4x}\right)$$

1.7 Usando Integrales

```
In [57]: integrate(cos(x), x)
         integrate(x**2,x)
```

Out[57]:

$$\frac{x^3}{3}$$

$$3$$

In [61]:
```
###integrar por intervalos
integrate(x**2,(x, 0, 900))
##para integrar con intervalos "infinitos" se escribe con 2 letras "o" = oo
```

Out[61]:

$$243000000$$

In [63]:
```
Integral(sqrt(1/x), x)
Integral(x**2+3,x)
```

Out[63]:

$$\int x^2 + 3\,dx$$

1.8 Matrices

In [64]:
```
Matrix([[1,2],[3,4]])
```

Out[64]:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

2 Usando Numpy

In [68]:
```
import numpy as np
a = np.array([1,2,3,4])
b = np.array([(1,2),(2,3)])
b
```

Out[68]:
```
array([[1, 2],
       [2, 3]])
```

In [72]:
```
####areglo del 1 al 15
arreglo = np.arange(15).reshape(3,5)
print(arreglo)
forma = arreglo.shape ###forma matricial
tamaño = arreglo.size ## Tamaño de la matriz £Cuiantos elementos tiene?
print("El tamaño de la matrz de  forma matricial es:")
print(forma)
print("los elemntos dentro de la matriz son: ")
print(tamaño)
```

```
[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]]
El tamaño de la matrz de  forma matricial es:
(3, 5)
```

```
los elemntos dentro de la matriz son:
15
```

```
In [73]: Matriz_complex = np.array([[2,3],[5,6]], dtype= complex)
         print("matriz compleja")
         Matriz_complex

matriz compleja


Out[73]: array([[ 2.+0.j,  3.+0.j],
                [ 5.+0.j,  6.+0.j]])
```

2.1 Variantes de matrices

```
In [74]: a = np.zeros((3,4))
         print("MAtriz de ceros")
         print(a)

MAtriz de ceros
[[ 0.  0.  0.  0.]
 [ 0.  0.  0.  0.]
 [ 0.  0.  0.  0.]]


In [75]: a = np.ones((2,2))
         print("Matriz de unos")
         print(a)

Matriz de unos
[[ 1.  1.]
 [ 1.  1.]]


In [78]: ##c = np.array([[1,2],[3,4]], dtype=int16)
         ###Matriz vacia
         c = np.empty((3,3))
         print(c)

[[  9.48606040e-322   4.28848981e-321   2.47032823e-323]
 [  3.93871109e-316   0.00000000e+000   3.23815565e-319]
 [  3.94177035e-316   0.00000000e+000   5.79304851e-312]]


In [84]: ####Secuencia de numeros con "arange"
         mz = np.arange(10,6000000, 2)
         print("Matriz que inicia en 10 y termina en 60, con paso de 2 en 2")
         print(mz)
```

```
        ###La sentencia Linspace no da la cantidad de elementos que deseamos imprimir
        ms = np.linspace(0, 2, 900)
        print("de 0 a 2 que me impromа 3 numeros")
        print(ms)
```

```
Matriz que inicia en 10 y termina en 60, con paso de 2 en 2
[     10      12      14 ..., 5999994 5999996 5999998]
de 0 a 2 que me impromа 3 numeros
[ 0.          0.00222469  0.00444939  0.00667408  0.00889878  0.01112347
  0.01334816  0.01557286  0.01779755  0.02002225  0.02224694  0.02447164
  0.02669633  0.02892102  0.03114572  0.03337041  0.03559511  0.0378198
  0.04004449  0.04226919  0.04449388  0.04671858  0.04894327  0.05116796
  0.05339266  0.05561735  0.05784205  0.06006674  0.06229143  0.06451613
  0.06674082  0.06896552  0.07119021  0.07341491  0.0756396   0.07786429
  0.08008899  0.08231368  0.08453838  0.08676307  0.08898776  0.09121246
  0.09343715  0.09566185  0.09788654  0.10011123  0.10233593  0.10456062
  0.10678532  0.10901001  0.11123471  0.1134594   0.11568409  0.11790879
  0.12013348  0.12235818  0.12458287  0.12680756  0.12903226  0.13125695
  0.13348165  0.13570634  0.13793103  0.14015573  0.14238042  0.14460512
  0.14682981  0.14905451  0.1512792   0.15350389  0.15572859  0.15795328
  0.16017798  0.16240267  0.16462736  0.16685206  0.16907675  0.17130145
  0.17352614  0.17575083  0.17797553  0.18020022  0.18242492  0.18464961
  0.1868743   0.189099    0.19132369  0.19354839  0.19577308  0.19799778
  0.20022247  0.20244716  0.20467186  0.20689655  0.20912125  0.21134594
  0.21357063  0.21579533  0.21802002  0.22024472  0.22246941  0.2246941
  0.2269188   0.22914349  0.23136819  0.23359288  0.23581758  0.23804227
  0.24026696  0.24249166  0.24471635  0.24694105  0.24916574  0.25139043
  0.25361513  0.25583982  0.25806452  0.26028921  0.2625139   0.2647386
  0.26696329  0.26918799  0.27141268  0.27363737  0.27586207  0.27808676
  0.28031146  0.28253615  0.28476085  0.28698554  0.28921023  0.29143493
  0.29365962  0.29588432  0.29810901  0.3003337   0.3025584   0.30478309
  0.30700779  0.30923248  0.31145717  0.31368187  0.31590656  0.31813126
  0.32035595  0.32258065  0.32480534  0.32703003  0.32925473  0.33147942
  0.33370412  0.33592881  0.3381535   0.3403782   0.34260289  0.34482759
  0.34705228  0.34927697  0.35150167  0.35372636  0.35595106  0.35817575
  0.36040044  0.36262514  0.36484983  0.36707453  0.36929922  0.37152392
  0.37374861  0.3759733   0.378198    0.38042269  0.38264739  0.38487208
  0.38709677  0.38932147  0.39154616  0.39377086  0.39599555  0.39822024
  0.40044494  0.40266963  0.40489433  0.40711902  0.40934372  0.41156841
  0.4137931   0.4160178   0.41824249  0.42046719  0.42269188  0.42491657
  0.42714127  0.42936596  0.43159066  0.43381535  0.43604004  0.43826474
  0.44048943  0.44271413  0.44493882  0.44716352  0.44938821  0.4516129
  0.4538376   0.45606229  0.45828699  0.46051168  0.46273637  0.46496107
  0.46718576  0.46941046  0.47163515  0.47385984  0.47608454  0.47830923
  0.48053393  0.48275862  0.48498331  0.48720801  0.4894327   0.4916574
  0.49388209  0.49610679  0.49833148  0.50055617  0.50278087  0.50500556
  0.50723026  0.50945495  0.51167964  0.51390434  0.51612903  0.51835373
  0.52057842  0.52280311  0.52502781  0.5272525   0.5294772   0.53170189
```

0.53392659 0.53615128 0.53837597 0.54060067 0.54282536 0.54505006
0.54727475 0.54949944 0.55172414 0.55394883 0.55617353 0.55839822
0.56062291 0.56284761 0.5650723 0.567297 0.56952169 0.57174638
0.57397108 0.57619577 0.57842047 0.58064516 0.58286986 0.58509455
0.58731924 0.58954394 0.59176863 0.59399333 0.59621802 0.59844271
0.60066741 0.6028921 0.6051168 0.60734149 0.60956618 0.61179088
0.61401557 0.61624027 0.61846496 0.62068966 0.62291435 0.62513904
0.62736374 0.62958843 0.63181313 0.63403782 0.63626251 0.63848721
0.6407119 0.6429366 0.64516129 0.64738598 0.64961068 0.65183537
0.65406007 0.65628476 0.65850945 0.66073415 0.66295884 0.66518354
0.66740823 0.66963293 0.67185762 0.67408231 0.67630701 0.6785317
0.6807564 0.68298109 0.68520578 0.68743048 0.68965517 0.69187987
0.69410456 0.69632925 0.69855395 0.70077864 0.70300334 0.70522803
0.70745273 0.70967742 0.71190211 0.71412681 0.7163515 0.7185762
0.72080089 0.72302558 0.72525028 0.72747497 0.72969967 0.73192436
0.73414905 0.73637375 0.73859844 0.74082314 0.74304783 0.74527253
0.74749722 0.74972191 0.75194661 0.7541713 0.756396 0.75862069
0.76084538 0.76307008 0.76529477 0.76751947 0.76974416 0.77196885
0.77419355 0.77641824 0.77864294 0.78086763 0.78309232 0.78531702
0.78754171 0.78976641 0.7919911 0.7942158 0.79644049 0.79866518
0.80088988 0.80311457 0.80533927 0.80756396 0.80978865 0.81201335
0.81423804 0.81646274 0.81868743 0.82091212 0.82313682 0.82536151
0.82758621 0.8298109 0.8320356 0.83426029 0.83648498 0.83870968
0.84093437 0.84315907 0.84538376 0.84760845 0.84983315 0.85205784
0.85428254 0.85650723 0.85873192 0.86095662 0.86318131 0.86540601
0.8676307 0.86985539 0.87208009 0.87430478 0.87652948 0.87875417
0.88097887 0.88320356 0.88542825 0.88765295 0.88987764 0.89210234
0.89432703 0.89655172 0.89877642 0.90100111 0.90322581 0.9054505
0.90767519 0.90989989 0.91212458 0.91434928 0.91657397 0.91879867
0.92102336 0.92324805 0.92547275 0.92769744 0.92992214 0.93214683
0.93437152 0.93659622 0.93882091 0.94104561 0.9432703 0.94549499
0.94771969 0.94994438 0.95216908 0.95439377 0.95661846 0.95884316
0.96106785 0.96329255 0.96551724 0.96774194 0.96996663 0.97219132
0.97441602 0.97664071 0.97886541 0.9810901 0.98331479 0.98553949
0.98776418 0.98998888 0.99221357 0.99443826 0.99666296 0.99888765
1.00111235 1.00333704 1.00556174 1.00778643 1.01001112 1.01223582
1.01446051 1.01668521 1.0189099 1.02113459 1.02335929 1.02558398
1.02780868 1.03003337 1.03225806 1.03448276 1.03670745 1.03893215
1.04115684 1.04338154 1.04560623 1.04783092 1.05005562 1.05228031
1.05450501 1.0567297 1.05895439 1.06117909 1.06340378 1.06562848
1.06785317 1.07007786 1.07230256 1.07452725 1.07675195 1.07897664
1.08120133 1.08342603 1.08565072 1.08787542 1.09010011 1.09232481
1.0945495 1.09677419 1.09899889 1.10122358 1.10344828 1.10567297
1.10789766 1.11012236 1.11234705 1.11457175 1.11679644 1.11902113
1.12124583 1.12347052 1.12569522 1.12791991 1.13014461 1.1323693
1.13459399 1.13681869 1.13904338 1.14126808 1.14349277 1.14571746
1.14794216 1.15016685 1.15239155 1.15461624 1.15684093 1.15906563
1.16129032 1.16351502 1.16573971 1.1679644 1.1701891 1.17241379

1.17463849 1.17686318 1.17908788 1.18131257 1.18353726 1.18576196
1.18798665 1.19021135 1.19243604 1.19466073 1.19688543 1.19911012
1.20133482 1.20355951 1.2057842 1.2080089 1.21023359 1.21245829
1.21468298 1.21690768 1.21913237 1.22135706 1.22358176 1.22580645
1.22803115 1.23025584 1.23248053 1.23470523 1.23692992 1.23915462
1.24137931 1.243604 1.2458287 1.24805339 1.25027809 1.25250278
1.25472747 1.25695217 1.25917686 1.26140156 1.26362625 1.26585095
1.26807564 1.27030033 1.27252503 1.27474972 1.27697442 1.27919911
1.2814238 1.2836485 1.28587319 1.28809789 1.29032258 1.29254727
1.29477197 1.29699666 1.29922136 1.30144605 1.30367075 1.30589544
1.30812013 1.31034483 1.31256952 1.31479422 1.31701891 1.3192436
1.3214683 1.32369299 1.32591769 1.32814238 1.33036707 1.33259177
1.33481646 1.33704116 1.33926585 1.34149055 1.34371524 1.34593993
1.34816463 1.35038932 1.35261402 1.35483871 1.3570634 1.3592881
1.36151279 1.36373749 1.36596218 1.36818687 1.37041157 1.37263626
1.37486096 1.37708565 1.37931034 1.38153504 1.38375973 1.38598443
1.38820912 1.39043382 1.39265851 1.3948832 1.3971079 1.39933259
1.40155729 1.40378198 1.40600667 1.40823137 1.41045606 1.41268076
1.41490545 1.41713014 1.41935484 1.42157953 1.42380423 1.42602892
1.42825362 1.43047831 1.432703 1.4349277 1.43715239 1.43937709
1.44160178 1.44382647 1.44605117 1.44827586 1.45050056 1.45272525
1.45494994 1.45717464 1.45939933 1.46162403 1.46384872 1.46607341
1.46829811 1.4705228 1.4727475 1.47497219 1.47719689 1.47942158
1.48164627 1.48387097 1.48609566 1.48832036 1.49054505 1.49276974
1.49499444 1.49721913 1.49944383 1.50166852 1.50389321 1.50611791
1.5083426 1.5105673 1.51279199 1.51501669 1.51724138 1.51946607
1.52169077 1.52391546 1.52614016 1.52836485 1.53058954 1.53281424
1.53503893 1.53726363 1.53948832 1.54171301 1.54393771 1.5461624
1.5483871 1.55061179 1.55283648 1.55506118 1.55728587 1.55951057
1.56173526 1.56395996 1.56618465 1.56840934 1.57063404 1.57285873
1.57508343 1.57730812 1.57953281 1.58175751 1.5839822 1.5862069
1.58843159 1.59065628 1.59288098 1.59510567 1.59733037 1.59955506
1.60177976 1.60400445 1.60622914 1.60845384 1.61067853 1.61290323
1.61512792 1.61735261 1.61957731 1.621802 1.6240267 1.62625139
1.62847608 1.63070078 1.63292547 1.63515017 1.63737486 1.63959956
1.64182425 1.64404894 1.64627364 1.64849833 1.65072303 1.65294772
1.65517241 1.65739711 1.6596218 1.6618465 1.66407119 1.66629588
1.66852058 1.67074527 1.67296997 1.67519466 1.67741935 1.67964405
1.68186874 1.68409344 1.68631813 1.68854283 1.69076752 1.69299221
1.69521691 1.6974416 1.6996663 1.70189099 1.70411568 1.70634038
1.70856507 1.71078977 1.71301446 1.71523915 1.71746385 1.71968854
1.72191324 1.72413793 1.72636263 1.72858732 1.73081201 1.73303671
1.7352614 1.7374861 1.73971079 1.74193548 1.74416018 1.74638487
1.74860957 1.75083426 1.75305895 1.75528365 1.75750834 1.75973304
1.76195773 1.76418242 1.76640712 1.76863181 1.77085651 1.7730812
1.7753059 1.77753059 1.77975528 1.78197998 1.78420467 1.78642937
1.78865406 1.79087875 1.79310345 1.79532814 1.79755284 1.79977753
1.80200222 1.80422692 1.80645161 1.80867631 1.810901 1.8131257

```
 1.81535039  1.81757508  1.81979978  1.82202447  1.82424917  1.82647386
 1.82869855  1.83092325  1.83314794  1.83537264  1.83759733  1.83982202
 1.84204672  1.84427141  1.84649611  1.8487208   1.85094549  1.85317019
 1.85539488  1.85761958  1.85984427  1.86206897  1.86429366  1.86651835
 1.86874305  1.87096774  1.87319244  1.87541713  1.87764182  1.87986652
 1.88209121  1.88431591  1.8865406   1.88876529  1.89098999  1.89321468
 1.89543938  1.89766407  1.89988877  1.90211346  1.90433815  1.90656285
 1.90878754  1.91101224  1.91323693  1.91546162  1.91768632  1.91991101
 1.92213571  1.9243604   1.92658509  1.92880979  1.93103448  1.93325918
 1.93548387  1.93770857  1.93993326  1.94215795  1.94438265  1.94660734
 1.94883204  1.95105673  1.95328142  1.95550612  1.95773081  1.95995551
 1.9621802   1.96440489  1.96662959  1.96885428  1.97107898  1.97330367
 1.97552836  1.97775306  1.97997775  1.98220245  1.98442714  1.98665184
 1.98887653  1.99110122  1.99332592  1.99555061  1.99777531  2.        ]
```

2.2 Operaciones con matricex

Para realizar operacion con matrices se debe asegurar que se cumplan con las propiedades de las mismas, de manera contraria python no permitirá hacer la operación y nos marcará algún error. a resolución de matrices se hace de manera automática

```
In [96]: a = np.array([[1,2], [2,3]])
         b = np.array([[5,6],[7,8]])
         print("Matrices originales")
         print(a)
         print(b)
         print("Resta de matrices")
         c = b-a
         print(c)
         print("suma de matrices")
         c = b+a
         print(c)
         print("£Multiplicacion si, pero elemento a elemento no de manera operacion matricial")
         c = np.multiply(a,b) ###multiplicacion
         print(c)
         print("Factorial usanod asterisco *")
         print(a*b) ##factorial?
         print("Multiplicacion de matrices de manera matricial")
         c = np.dot(a,b) ### Manera matricial mutiplicacion
         print(c)

Matrices originales
[[1 2]
 [2 3]]
[[5 6]
 [7 8]]
Resta de matrices
```

```
[[4 4]
 [5 5]]
suma de matrices
[[ 6  8]
 [ 9 11]]
£Multiplicacion si, pero elemento a elemento no de manera operacion matricial
[[ 5 12]
 [14 24]]
Factorial usanod asterisco *
[[ 5 12]
 [14 24]]
Multiplicacion de matrices de manera matricial
[[19 22]
 [31 36]]
```

2.3 Usando Álgebra Lineal

Se puede implementar algunas opperaciones de Álgebra Lineal Básica

```
In [105]: a = np.matrix([[1,2,3],[4,5,6],[7,8,9]])
          print("Matriz original")
          print(a)
          transpuesta = a.T ###Matriz traspuesta
          print("Matriz traspuesta")
          print(transpuesta)
          print("Matriz inversa")
          ##inversa = a.linalg.inv() ##matriz inversa
          ##print(inversa)
          print("Determinante de la matriz")
          deter= np.linalg.det(a)
          print(deter)
          print("Traza de una matriz")
          traza = np.trace(a)
          print(traza)
```

```
Matriz original
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Matriz traspuesta
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Matriz inversa
Determinante de la matriz
6.66133814775e-16
Traza de una matriz
```

15

2.4 Ejercicio:

Usando numpy y/o sympy realizar: 1. 2 derivadas 2. 2 integrales 3. 2 simplificaciones de terminos 4. 2 operacione con matrices