problem with pi

pi

In the program which is running, i use the pi from numpy currently.

In 249

```python
from sympy import sin, cos, pi, exp, prod, sqrt
import numpy as np
import math
display(pi)
display(np.pi)
display(math.pi)
```

π

```
3.141592653589793
```

```
3.141592653589793
```

In 250

```python
display(sin(pi))
display(sin(np.pi))
display(sin(math.pi))
```

0

$1.22464679914735 \cdot 10^{-16}$

$1.22464679914735 \cdot 10^{-16}$

- If we use pi built in sympy, then the calculations will be accurate, but slower.
- But if we use math.pi or np.pi, the results may not be as accurate, but they are of an acceptable accuracy and the speed of computation will be higher.

for example:(f12)

$$
\begin{aligned}
f(x) &= \frac{\pi}{D}\begin{cases} 10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i \\ -1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] \\ +(y_D - 1)^2\Big\} + \sum_{i=1}^{D} u_i \end{cases} \\
u_i &= \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & a \le x_i \le a \\ k(-x_i - a)^m & x_i < -a \end{cases} \\
y &= 1 + (x_i + 1)/4
\end{aligned}
\tag{1}
$$

By using math.pi

Although the fitness is not zero, but the accuracy is acceptable.

```
In 259
import math
from sympy import sin, cos,exp, prod, sqrt
pi = math.pi
x=[-1 for i in range(10)] # optimal solution
print("x is {0}".format(x))
y = [1 + (i + 1) / 4 for i in x]
print("y is {0}".format(y))
part3 = sum([U(i, 10, 100, 4) for i in x])
part2 = sum([power(y[i] - 1, 2) * (1 + 10 * power(sin(y[i + 1] * pi), 2)) for i in range(len(x) - 1)])
fitness = (pi / 50) * (10 * power(sin(y[0] * pi), 2) + part2 + power(y[-1] - 1, 2)) + part3
print('fitness is %.50f'%fitness)
```

```
x is [-1, -1, -1, -1, -1, -1, -1, -1, -1, -1]
y is [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
fitness is 0.00000000000000000000000000000000942326863071983471
```

By using pi from sympy:

```
In 260
from sympy import sin, cos,exp, prod, sqrt,pi
x=[-1 for i in range(10)] # optimal solution
print("x is {0}".format(x))
y = [1 + (i + 1) / 4 for i in x]
print("y is {0}".format(y))
part3 = sum([U(i, 10, 100, 4) for i in x])
part2 = sum([power(y[i] - 1, 2) * (1 + 10 * power(sin(y[i + 1] * pi), 2)) for i in range(len(x) - 1)])
fitness = (pi / 50) * (10 * power(sin(y[0] * pi), 2) + part2 + power(y[-1] - 1, 2)) + part3
print("fitness is {0}".format(fitness.evalf()))
```

```
x is [-1, -1, -1, -1, -1, -1, -1, -1, -1, -1]
y is [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
fitness is 0
```

problem with e

In the program which is running, i use exp(1) as e.

First, there is a difference between 1 and 1.0: to be honest, I don't understand this difference.

```
In 266
from sympy import sin, cos,exp, prod, sqrt,pi
display(exp(1))
display(exp(1.0))
```

e

2.71828182845905

Second, there is also a difference between the e from sympy, numpy and math.

again, the computation efficiency is better if we use math.e or np.e and this can also produce an acceptable accuracy.

```python
from sympy import exp
import numpy as np
import math
print(np.e)
print(math.e)
print(exp(1.0))
```

```
2.718281828459045
2.718281828459045
2.71828182845905
```

problem with f12

This table is from appendix A:

(CBMA_Darts_playing_robot A.pdf)

Test Functions

$F1 = \sum_{i=1}^{50} x_i^2$
$F2 = \sum_{i=1}^{50} \lvert x_i \rvert + \prod_{i=1}^{50} \lvert x_i \rvert$
$F3 = \sum_{i=1}^{50} \left(\sum_i = 1^{50}(x_i)^2\right)$
$F4 = max \lvert x_i \rvert$
$F5 = \sum_{i=1}^{49} \left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right)$
$F6 = \sum_{i=1}^{49} (x_i + 0.5)^2$
$F7 = \sum_{i=1}^{50} (ix_i^4 + rand[0,1]$
$F8 = \sum_{i=1}^{50} \left(-x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)\right)$
$F9 = \sum_{i=1}^{50} (x_i^2 - 10\cos(2\pi x_i) + 10)$
$F10 = -20\exp\left(-0.2\sqrt{0.02\sum_{i=1}^{50} x_i^2}\right) - \exp\left(0.02\sum_{i=1}^{50}\cos 2\pi x_i\right) + 20 + e$
$F11 = \frac{1}{4000}\sum_{i=1}^{50} x_i^2 - \prod_{i=1}^{50}\cos\frac{x_i}{\sqrt{i}} + 1$
$F12 = \frac{\pi}{50}\left(10\sin(\pi y_1) + \sum_{i=1}^{49}(y_i - 1)^2(1 + 10\sin^2 \pi y_{i+1} + (y_{50} - 1)^2\right) + \sum_{i=1}^{50} u(x_i, 10, 100, 4)$
$F13 = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{49}(x_i - 1)^2(1 + \sin^2(3\pi x_i + 1)) + (x_{50} - 1)^2(1 + \sin^2(2\pi x_{50}))) + \sum_{i=1}^{50} u(x_i, 5, 100, 4)$
a=[-32 -16 0 16 32 -32 -16 0 16 32 -32 -16 0 16 32 -32 -16 0 16 32 -32 -16 0 16 32; -32 -32 -32 -32 -32 -16 -16 -16 -16 -16 0 0 0 0 0 16 16 16 16 16 32 32 32 32 32]; $F14 = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$

a=[.1957 .1947 .1735 .16 .0844 .0627 .0456 .0342 .0323 .0235 .0246];
b=[.25 .5 1 2 4 6 8 10 12 14 16]; b=1./b;
$F15 = \sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^2$

$F16 = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2{-}^4$
$F17 = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$
$F18 = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times$ $\times(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$

a=[3 10 30; .1 10 35; 3 10 30; .1 10 35];
c=[1 1.2 3 3.2];
p=[.3689 .117 .2673; .4699 .4387 .747; .1091 .8732 .5547; .03815 .5743 .8828];
$F19 = -\sum_{i=1}^{4}(c_i\exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$

a=[10 3 17 3.5 1.7 8;.05 10 17 .1 8 14;3 3.5 1.7 10 17 8;17 8 .05 10 .1 14];
c=[1 1.2 3 3.2];
p=[.1312 .1696 .5569 .0124 .8283 .5886;.2329 .4135 .8307 .3736 .1004 .9991;
.2348 .1415 .3522 .2883 .3047 .6650;.4047 .8828 .8732 .5743 .1091 .0381];
$F20 = -\sum_{i=1}^{4}(c_i\exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$

a=[4 4 4 4; 1 1 1 1; 8 8 8 8; 6 6 6 6; 3 7 3 7];
c=[.1 .2 .2 .4 .4];
$F21 = -\sum_{i=1}^{5}\left((X - a_i)(X - a_i)^T + c_i\right)^{-1}$

a=[4 4 4 4; 1 1 1 1; 8 8 8 8; 6 6 6 6; 3 7 3 7; 2 9 2 9; 5 5 3 3];
c=[.1 .2 .2 .4 .4 .6 .3];
$F22 = -\sum_{i=1}^{7}\left((X - a_i)(X - a_i)^T + c_i\right)^{-1}$

a=[4 4 4 4; 1 1 1 1; 8 8 8 8; 6 6 6 6; 3 7 3 7; 2 9 2 9; 5 5 3 3; 8 1 8 1; 6 2 6 2; 7 3.6 7 3.6];
c=[.1 .2 .2 .4 .4 .6 .3 .7 .5 .5];
$F23 = -\sum_{i=1}^{10}\left((X - a_i)(X - a_i)^T + c_i\right)^{-1}$

$U(x, a, k, m) = k((x - a)^m)(x > a) + k((-x - a)^m)(x < (-a))$

This table is from the reference-2 of appendix A:

(Marine Predators Algorithm: A nature-inspired metaheuristic.pdf)

Table 17
Mathematical formulation and properties of unimodal, multimodal and composition functions.

Functions	Dim	Domain	Global opt
$TF_1(x) = \sum_{i=1}^{d} x_i^2$	50	[−100,100]	0
$TF_2(x) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	50	[−100,100]	0
$TF_3(x) = \sum_{i=1}^{d} (\sum_{j=1}^{d} x_j)^2$	50	[−100,100]	0
$TF_4(x) = \text{Max}\{\lvert x_i \rvert, 1 \le i \le d\}$	50	[−100,100]	0
$TF_5(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	50	[−30,30]	0
$TF_6(x) = \sum_{i=1}^{d-1} ([x_i + 0.5])^2$	50	[−100,100]	0
$TF_7(x) = \sum_{i=1}^{d} ix_i^4 + random[0, 1]$	50	[−1.28,1.28]	0
$TF_8(x) = \sum_{i=1}^{d} -x_i \sin(\sqrt{\lvert x_i \rvert})$	50	[−500,500]	− 418.98 × d
$TF_9(x) = \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i) + 10]$	50	[−5.12,5.12]	0
$TF_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$	50	[−32,32]	0
$TF_{11}(x) = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\cos(\frac{x_i}{\sqrt{i}}) + 1$	50	[−600,600]	0
$TF_{12}(x) = \frac{\pi}{d}\{10\sin(\pi y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2[1 + 10\ sin^2(\pi y_{i+1})] + (y_d - 1)^2\} + \sum_{i=1}^{d} u(x_i, 10, 100, 4)$	50	[−50,50]	0
$TF_{13}(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{d-1} (x_i - 1)^2[1 + sin^2(3\pi x_i + 1)] + (x_d - 1)^2[1 + \sin^2(2\pi x_d)]\} + \sum_{i=1}^{d} u(x_i, 5, 100, 4)$	50	[−50,50]	0
$TF_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}\right)^{-1}$	2	[−65,65]	1
$TF_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[−5,5]	0.00030
$TF_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[−5,5]	−1.0316
$TF_{17}(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})cosx_1 + 10$	2	[−5,5]	0.398
$TF_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times$ $[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	[−2,2]	3
$TF_{19}(x) = -\sum_{i=1}^{4} c_i\ exp(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	3	[1,3]	−3.86
$TF_{20}(x) = -\sum_{i=1}^{4} c_i\ exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	6	[0,1]	−3.32
$TF_{21}(x) = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0,10]	−10.1532
$TF_{22}(x) = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0,10]	−10.4028
$TF_{23}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0,10]	−10.5363

(*continued on next page*)

the problem is that f12 is wrong.

according to appendix A, f12 is wrong, because it forgets to calculate a square.

In 272

```
from sympy import sin, cos,exp, prod, sqrt,pi
x=np.random.randn(10) # optimal solution
print("x is {0}".format(x))
y = [1 + (i + 1) / 4 for i in x]
print("y is {0}".format(y))
part3 = sum([U(i, 10, 100, 4) for i in x])
part2 = sum([power(y[i] - 1, 2) * (1 + 10 * power(sin(y[i + 1] * pi), 2)) for i in range(len(x) - 1)])
fitness = (pi / 50) * (10 * sin(pi * y[0]) + part2 + power(y[-1] - 1, 2)) + part3
print("fitness is {0}".format(fitness.evalf()))
```

```
x is [ 0.50008573 -0.38286646 -0.63271955  0.61837971  0.44157458 -1.47596304
 -0.12000569  0.49532425 -0.29672095  0.61865806]
y is [1.3750214321666552, 1.1542833847313685, 1.0918201129730603, 1.404594927070172, 1
.360393644998172, 0.8810092407265956, 1.2199985782493443, 1.3738310620380254, 1.1758197621214894,
 1.4046645143701129]
fitness is -0.334439507523002
```

optimal is zero

The right version is the following

354 Words

$F1 = \sum_{i=1}^{50} x_i^2$
$F2 = \sum_{i=1}^{50} \lvert x_i\rvert + \prod_{i=1}^{50} \lvert x_i\rvert$
$F3 = \sum_{i=1}^{50} (\sum_i = 1^{50}(x_i)^2)$
$F4 = max\lvert x_i\rvert$
$F5 = \sum_{i=1}^{49} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$
$F6 = \sum_{i=1}^{49} (x_i + 0.5)^2$
$F7 = \sum_{i=1}^{50} (ix_i^4 + rand[0,1]$
$F8 = \sum_{i=1}^{50} \left(-x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$
$F9 = \sum_{i=1}^{50} (x_i^2 - 10\cos(2\pi x_i) + 10)$
$F10 = -20\exp\left(-0.2\sqrt{0.02\sum_{i=1}^{50} x_i^2}\right) - \exp\left(0.02\sum_{i=1}^{50} \cos 2\pi x_i\right) + 20 + e$
$F11 = \frac{1}{4000}\sum_{i=1}^{50} x_i^2 - \prod_{i=1}^{50} \cos\frac{x_i}{\sqrt{i}} + 1$
$F12 = \frac{\pi}{50}\left(10\sin(\pi y_1) + \sum_{i=1}^{49} (y_i - 1)^2(1 + 10\sin^2 \pi y_{i+1} + (y_{50} - 1)^2\right) + \sum_{i=1}^{50} u(x_i, 10, 100, 4)$
$F13 = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{49} (x_i - 1)^2(1 + \sin^2(3\pi x_i + 1)) + (x_{50} - 1)^2(1 + \sin^2(2\pi x_{50}))) + \sum_{i=1}^{50} u(x_i, 5, 100, 4)$
a=[-32 -16 0 16 32 -32 -16 0 16 32 -32 -16 0 16 32 -32 -16 0 16 32 -32 -16 0 16 32; -32 -32 -32 -32 -32 -16 -16 -16 -16 -16 0 0 0 0 0 16 16 16 16 16 32 32 32 32 32]; $F14 = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}\right)^{-1}$

a=[.1957 .1947 .1735 .16 .0844 .0627 .0456 .0342 .0323 .0235 .0246];
b=[.25 .5 1 2 4 6 8 10 12 14 16]; b=1./b;
$F15 = \sum_{i=1}^{11} \left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^2$

$F16 = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2 -^4$
$F17 = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$
$F18 = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times$ $\times(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$

a=[3 10 30; .1 10 35; 3 10 30; .1 10 35];
c=[1 1.2 3 3.2];
p=[.3689 .117 .2673; .4699 .4387 .747; .1091 .8732 .5547; .03815 .5743 .8828];
$F19 = -\sum_{i=1}^{4} (c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$

In 274

```
from sympy import sin, cos,exp, prod, sqrt,pi
x=[ 0.50008573,-0.38286646,-0.63271955, 0.61837971, 0.44157458,-1.47596304
 ,-0.12000569, 0.49532425,-0.29672095, 0.61865806] # optimal solution
print("x is {0}".format(x))
y = [1 + (i + 1) / 4 for i in x]
print("y is {0}".format(y))
part3 = sum([U(i, 10, 100, 4) for i in x])
part2 = sum([power(y[i] - 1, 2) * (1 + 10 * power(sin(y[i + 1] * pi), 2)) for i in range(len(x) - 1)])
fitness = (pi / 50) * (10 * power(sin(y[0] * pi), 2) + part2 + power(y[-1] - 1, 2)) + part3
print("fitness is {0}".format(fitness.evalf()))
```

```
x is [0.50008573, -0.38286646, -0.63271955, 0.61837971, 0.44157458, -1.47596304, -0.12000569,
0.49532425, -0.29672095, 0.61865806]
y is [1.3750214325, 1.154283385, 1.0918201125, 1.4045949275, 1.360393645, 0.88100924, 1.2199985775,
 1.3738310624999999, 1.1758197625, 1.404664515]
fitness is 0.782400636886830
```

i found this mistake because i saw the formula in antoher paper.

problem with f18

as I mentioned before, f8 can find the global minima at 98/100 chance, but there were two values which are under the minima.

Here is the reason:

each function has a domain, at first, we generate a group of individuals, okay.

and through crossover and mutation operations we generate the new individual.

the problem is, the range of each gene in the new individual may be out of domain range.

the initial program does not check the domian for the new individual, and in most cases, it won't be out of the domian range since that the range for mutation is very narrow. (my fault)

if one gene is out of the domian, then the global minima is no longer meaningful

I already modified the program several days ago.

I added the code for checking the domian after a new indivudal is generated.