# Characteristic cone for equation ideal

This program takes the generators of the cone and the partial difference equations of discrete autonomous nD system as inputs and gives a 'yes/no' certificate according to whether the given cone is a characteristic cone for the given system or not.

First the dimension of the system, the number of partial difference equations and the number of cone generating vectors are taken as input. Then the difference equations and the vectors are given by the user. As output the program will give if the given cone is a characteristic cone or not.

The dimension of the system determines the number of free variables of the system. According to the input given by the user, the free variables are defined in x, as x1, x2, x3.... So the variables that are valid during matrix input are x1, x1^-1, x2, x2^-1, x3, x3^-1.... To make the input more readable, and reduce typo errors, user can use y variables instead of x inverses i.e. user can give input in form of x1, y1, x2, y2, x3, y3..., where y1, y2, y3... are considered as x1^-1, x2^-1, x3^-1...

Transposed column vectors are taken as input for generating the cone.
For example: c1 = 0 1 0 is actually the vector [0 1 0]^T

EXAMPLE 1:

- Dimension of the system: 3
- Number of difference equations: 2
- Number of vectors for defining the cone: 3

- Enter the partial difference equations:
- f1= x1^2+x2^2+x3^2-1
- f2= x2*x3-1

- please enter the transposed column vectors to define the cone:
- v1= -1 0 0
- v2= 0 -1 0
- v3= 0 0 -1

- This is a characteristic cone

EXAMPLE 1:

-Dimension of the system: 3
-Number of partial difference equations: 2
-Number of vectors for generating the cone: 3

-Enter the partial difference equations:
-f1= x1^5*x3 + x2^6 + x3^7*x2
-f2= x3*x1^-1

Authors: Ayan Sengupta, Mousumi Mukherjee and Debasattam Pal

-please enter the transposed column vectors to define the cone:
-v1= 1 0 0
-v2= 0 -1 0
-v3= 0 0 -1

-This is a characteristic cone

Some predefined functions will give more information than just an 'yes/no' answer

EXAMPLE:

```
-remainder
-[-t1*t2^2 - t1*t3^2 + t1, t3, t2, t1, t2, t3]

-P_ring
-Multivariate Polynomial Ring in x1, x2, x3, y1, y2, y3, t1, t2, t3
 over Real Field with 53 bits of precision

-difference_poly
-[x1^2 + x2^2 + x3^2 - 1.00000000000000, x2*x3 - 1.00000000000000]

-cone_poly
-[-y1 + t1, -y2 + t2, -y3 + t3]

-G
-[x1 + t1*t2^2 + t1*t3^2 - t1, x2 - t3, x3 - t2, y1 - t1, y2 - t2, y3 - t3,
 t1^2*t2 + t1^2*t3^3 - t1^2*t3 + t3, t1^2*t3^4 - t1^2*t3^2 + t1^2 + t3^2,
  t2*t3 - 1.00000000000000]
```

HOW TO RUN THE CODE IN SAGE:

First open SAGE from the terminal. Then you can load a file in a Sage session with the command "load".

example:

if your file is in the current directory where you started sage then:
```
load("your_file.sage")
```

if your file is in another directory then :
```
load("~/the_file_location/your_file.sage")
```

Authors: Ayan Sengupta, Mousumi Mukherjee and Debasattam Pal