CoCoA-5 - Bug #1216

RationalSolve: gives wrong answer

07 Aug 2018 18:17 - John Abbott

Status:	Closed	**Start date:**	07 Aug 2018
Priority:	Normal	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	bug	**Estimated time:**	5.55 hours
Target version:	CoCoA-5.4.0	**Spent time:**	5.55 hours

Description

RationalSolve gives the wrong answer in this example:

```
/**/ use ZZ/(3)[x,y];
/**/ RationalSolve([x,x*y]);
[]
```

I think the result should be [[0,0],[0,1],[0,-1]]

Investigate and fix.

Related issues:

Related to CoCoA-5 - Bug #724: RationalSolve: wrongly complains about non zer...	**Closed**	**02 Jun 2015**
Related to CoCoA-5 - Bug #1215: RationalSolve: gives "Error: must be non-zero"	**Closed**	**07 Aug 2018**
Related to CoCoA-5 - Bug #1574: ApproxSolveTF	**New**	**03 Feb 2021**
Related to CoCoA-5 - Bug #1573: ApproxSolve: very imprecise	**Closed**	**30 Jan 2021**

History

#1 - 07 Aug 2018 18:19 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

Here is a 0-dim failing example:

```
use ZZ/(3)[x,y];
RationalSolve([x,x*y,y]);
[]
```

The result must surely be [[0,0]].

#2 - 08 Aug 2018 11:34 - John Abbott

The point is that RationalSolve tries to be clever about deciding whether it should look for solutions in affine space or in projective space. However, it does not tell the user which type of solution it returns.

If all the polys are homogeneous then RationalSolve returns "projective" solutions; otherwise it returns "affine" solutions.

With the args [x,x*y] the solution found is projective, and the answer [0,1] is correct; other apparent solutions such as [0,2] are the same projective

point.

With the args [x,x*y,y] the cheap trick used for detecting the special case of an ideal whose radical is the "irrelevant ideal" does not work; so this is regarded as homogeneous input, and the correct answer is that there are no rational projective solutions.

SUGGESTION it would be better if the function returned an indication of whether the solutions produced are affine or projective.

#3 - 08 Aug 2018 12:41 - John Abbott

What should RationalSolve do if the input is homogeneous but the grading is non-standard?

#4 - 08 Aug 2018 12:48 - John Abbott

Assuming we still want to have the automatic selection between affine and projective solving, I suggest the following criterion:
if the ideal (generated by the given polys) is zero-dim in the subring generated by the indets actually appearing in the list of polys then use affine solving otherwise if the ideal is homog then use projective solving. If neither case applies then error.

#5 - 08 Aug 2018 17:49 - John Abbott

Another example...

```
use ZZ/(3)[x,y,z];
RationalSolve([x,x*(y-z)]);
--> ERROR: Value must be non-zero
--> [CoCoALib] factor(x)
--> WHERE: at line 102 (column 23) of RationalPoints.cpkg5
-->   LinearFacs := [g in factor(f).factors | deg(g) = 1];
-->                       ^^^^^^^^^
```

#6 - 05 Apr 2019 20:14 - John Abbott

- Related to Bug #724: RationalSolve: wrongly complains about non zero-dim even in finite char added

#7 - 05 Apr 2019 20:17 - John Abbott

- Related to Bug #1215: RationalSolve: gives "Error: must be non-zero" added

#8 - 01 Oct 2019 14:08 - John Abbott

- Target version changed from CoCoA-5.3.0 to CoCoA-5.4.0

#9 - 31 Jan 2021 10:43 - John Abbott

Bernhard Andraschko reportes the following:

```
Use QQ[x,y];
RationalSolve([x^2+y^2]);
RationalSolve([x^2,y^2]);
```

In both cases, it normally should return [[0,0]], but it does not.
In the second case it even claims homog system is not 1-dimensional --- whatever that means.

#10 - 03 Feb 2021 17:26 - John Abbott

I now think it might be better to avoid the "clever" function RationalSolve which tries to guess whether it should look for affine solutions or projective ones.
The heuristic it uses is not entirely transparent, and makes some strange mistakes at times.

Comments? Opinions?

#11 - 03 Feb 2021 22:39 - John Abbott

- Related to Bug #1574: ApproxSolveTF added

#12 - 04 Feb 2021 21:05 - John Abbott

- Assignee set to John Abbott

- % Done changed from 10 to 60

I have modified my copy of the code. It now behaves more "predictably".

RationalSolve now searches only for *affine* solutions, and will complain if the input is not 0-dim (and char = 0).
I have added a new fn **RationalSolveHomog** which computes *projective* solns.
I have stopped exporting RationalAffinePoints and RationalProjectivePoints (for the time being, at least) -- anyway, they were not documented.

For example, RationalSolve([x^2+y^2]); now gives error that input is not a 0-dim system.
One could argue that it should give an empty output (but doesn't that imply being able to solve diophantine systems in general?).

The other examples here seem to give plausible results. I do still feel slightly uneasy about allowing non-0-dim systems if the coeff field is finite.

#13 - 05 Feb 2021 15:41 - John Abbott

I am wondering about changing the return value of RationalSolve into a record, *e.g.*
currently we get

```
/**/ RationalSolve([x^2+y^2-2,x*y-1]);
[[-1,  -1],  [1,  1]]
```

new proposal

```
/**/ RationalSolve([x^2+y^2-2,x*y-1]);
record[indets := [x,  y],  solns := [[-1,  -1],  [1,  1]]]
```

This way it is clearer how to interpret the result: *i.e.* which indets were actually used (and which ignored), and the order of them.

Comments? Opinions?

#14 - 10 Feb 2021 20:50 - John Abbott

Anna has approved the suggestion in comment 13 (just above) about returning a record.
Now I must impl, and revise the doc. Hope to do this soon.

#15 - 10 Feb 2021 21:28 - John Abbott

Now I wonder what the names of the fields in the records should be.
It may be sensible to use different names for the list of *affine* solns, and the list of *projective* solns.
Or is that against KISS?

Possible names are **AffinePts** and **ProjectivePts** (or **ProjPts**).

PS: I have implemented the suggestion in comment 13, incl revised doc.

#16 - 10 Feb 2021 21:33 - John Abbott

- % Done changed from 60 to 70

Feedback and constructive criticism are welcome!

I'd like to close this issue soon.

#17 - 11 Feb 2021 17:56 - Anna Maria Bigatti

> Possible names are **AffinePts** and **ProjectivePts** (or **ProjPts**).

I vote for **AffinePts** and **ProjectivePts**

#18 - 12 Feb 2021 00:08 - John Abbott

- Status changed from In Progress to Feedback

- % Done changed from 70 to 90

I have implemented the new field names, and updated doc, tests, manual.

New state: Feedback.

#19 - 16 Sep 2021 13:00 - Anna Maria Bigatti

- Related to Bug #1573: ApproxSolve: very imprecise added

#20 - 24 Sep 2021 21:50 - John Abbott

- Status changed from Feedback to Closed

- % Done changed from 90 to 100

- Estimated time set to 5.55 h

This has been in feedback for 7 months.
I think we mention in the release notes that it is a backward INcompatible change.