CoCoA-5 - Bug #1171

RealRoots: first point is sometimes wrong?

03 Apr 2018 18:25 - Anna Maria Bigatti

Status:	Closed	**Start date:**	03 Apr 2018
Priority:	Normal	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	bug	**Estimated time:**	6.25 hours
Target version:	CoCoA-5.4.2	**Spent time:**	6.30 hours

Description

It seems that sometimes (very rarely) the first root given by RealRoots is wrong

```
/**/ use QQ[x];
/**/ f :=   x^25 +1500*x^24 -3472*x^23 +75*x^22 -2888*x^21 +12984*x^20 +(-22927/4)*x^19 +11578*x^18 -22029*x^17 +(18981/2)*x^16 -21819*x^15 +29021*x^14 -12855*x^13 +25232*x^12 -25419*x^11 +4494*x^10 -20702*x^9 +14894*x^8 -1920*x^7 +12502*x^6 -92*x^5 +1700*x^4 -6628*x^3 -616*x^2 -520*x +920;

/**/ RR := RealRoots(f, 1/1000);
/**/ DecimalStr(subst(f, x, RR[1].inf));
-671001937258376671013879113638951064799173158922473366163496632995950081 3.670
/**/ DecimalStr(subst(f, x, RR[2].inf));
0.034
/**/ DecimalStr(subst(f, x, RR[3].inf));
-0.418
```

Related issues:

Related to CoCoALib - Feature #1173: Upper bound for value of poly in an inte...	**New**	**04 Apr 2018**
Related to CoCoA-5 - Bug #1573: ApproxSolve: very imprecise	**Closed**	**30 Jan 2021**

History

#1 - 03 Apr 2018 18:27 - Anna Maria Bigatti

I found this bug while debugging ApproxSolve.
I had noticed that, when the answer was wrong, it was just on the first point.

#2 - 03 Apr 2018 18:57 - Anna Maria Bigatti

It seems that actually the problem is in the instability of the roots: the approximate root is probably close to the real solution, but the evaluation is not close to 0.

```
/**/ f :=   x^25 +1500*x^24 -3472*x^23 +75*x^22 -2888*x^21 +12984*x^20 +(-22927/4)*x^19 +11578*x^18 -22029*x^17 +(18981/2)*x^16 -21819*x^15 +29021*x^14 -12855*x^13 +25232*x^12 -25419*x^11 +4494*x^10 -20702*x^9 +14894*x^8 -1920*x^7 +12502*x^6 -92*x^5 +1700*x^4 -6628*x^3 -616*x^2 -520*x +920;

/**/ [DecimalStr(subst(f, x, rr.inf)) | rr in RealRoots(f, 10^(-50))];
["-1.095",  "0.000",  "-0.000",  "0.000",  "-0.000"]

/**/ [DecimalStr(subst(f, x, rr.inf)) | rr in RealRoots(f, 10^(-100))];
["-0.000",  "0.000",  "-0.000",  "0.000",  "-0.000"]
```

Another example on this pathology.

```
/**/ f :=   x^25  -3472*x^23 +11578*x^18 +x^7 +92*x^5 +x^4 -x^3 -x^2 -x +920;
/**/ [DecimalStr(subst(f, x, rr.inf)) | rr in RealRoots(f, 10^(-100))];
--> 0 0 0

/**/ [DecimalStr(subst(f, x, rr.inf)) | rr in RealRoots(f, 10^(-10))];
```

```
--> ["-17497148070995812479243 92.282",  "0.000",  "-188291563539990536990082 0.013"]
```

#3 - 04 Apr 2018 11:34 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 50

One can estimate the "stability" of the root by computing the value of the derivative at that root. By Taylor expansion we have f(alpha+eps) = f(alpha) + eps*deriv(f)(alpha) + other terms.

Probably we would want an upper bound for abs(Df(beta)) where beta is a "good approximation" to alpha.

#4 - 04 Apr 2018 11:49 - John Abbott

- Related to Feature #1173: Upper bound for value of poly in an interval added

#5 - 05 Apr 2018 14:19 - John Abbott

It is not as simple as I first thought. While the first order approximation I mentioned in comment 3 is correct, it is only a first order approximation.

I made an implementation, and I tried the example in comment 2. That worked OK.
But with the polynomial **f^2+f** problems arose, presumably because the first order approximation was not good.

It would be nice to have a fn which returns intervals with the guarantee that **max(abs(f(x)) | x in [a,b]) < epsilon** where **epsilon** is the bound specified by a user.

#6 - 05 Apr 2018 14:26 - John Abbott

I suggest a function with the following semantics:

- call **RealRootsSmallValue(f, eps)**
- result is a list of closed intervals whose interiors are disjoint such that
  - width of interval is less than **eps** (less than or equal?)
  - value of **f** over each interval is less than **eps** (less than or equal?)

It is permitted to return intervals which are "too small".

What do you think? Suggestions for the fn name?

#7 - 16 May 2018 14:27 - John Abbott

I am not so happy with the suggested interface in comment 6: the parameter eps is doing double duty (a bound for the accuracy of the root, and a bound for the value of the function). Having two separate parameters for these bounds would be semantically neater (but possibly more awkward for the caller?)

Comments? Suggestions?

#8 - 25 Jul 2018 17:16 - John Abbott

- Target version changed from CoCoA-5.2.4 to CoCoA-5.3.0

#9 - 01 Oct 2019 14:16 - John Abbott

- Target version changed from CoCoA-5.3.0 to CoCoA-5.?.?

#10 - 11 Mar 2020 21:35 - John Abbott

- Target version changed from CoCoA-5.?.? to CoCoA-5.4.2

#11 - 12 Feb 2021 00:15 - John Abbott

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

#12 - 22 Feb 2021 20:47 - John Abbott

We could try to mimic the hack described in issue #1573, where the values are checked at the approx solutions, and refinement continues until the values are "heuristically small".
However such a change would make RealRoots semantically unclean.

Maybe it is just simpler to add a comment in the manual pointing the user to ApproxSolve if they want "heuristically small" values?

#13 - 01 Mar 2021 21:13 - John Abbott

- Assignee set to John Abbott

- % Done changed from 50 to 60

I have added a short comment to the documentation.

I can see that it might be nice to have a version of RealRoots which takes 2 "epsilon" values (one for the interval width, and for the max value over each interval). But how would one easily distinguish these two "epsilons" with rather different meanings?
Also, it might not be trivial to ensure that the polynomial is small over the entire interval see issue #1173)

#14 - 14 Mar 2023 22:01 - John Abbott

- Status changed from In Progress to Closed

- % Done changed from 60 to 100

- Estimated time set to 6.25 h

I think the revised documentation is helpful here; enough so, that I shall close the issue.