

Re: Maple's linear algebra routine and higher precision arithmetic

Source: <http://sci.tech-archive.net/Archive/sci.math.symbolic/2006-05/msg00167.html>

- *From:* israel@xxxxxxxxxxx (Robert Israel)
 - *Date:* 31 May 2006 16:00:11 GMT
-

In article <pan.2006.05.31.15.21.21.254816@xxxxxxxxxxx>, Ronald Haynes <ronald.haynes@xxxxxxxxxxx> wrote:

On Mon, 29 May 2006 13:12:03 -0700, Ronald Bruck wrote:

In article <pan.2006.05.29.18.47.40.640507@xxxxxxxxxxx>, Ronald Haynes <ronald.haynes@xxxxxxxxxxx> wrote:

Hi, many thanks... I just want to test the validity of some lower precision results with results generated by increasing levels of precision.

Would defining the matrix entries as `-1.0 4.0 1.0` do the trick to force floats?

In Maple—yes. If you do this in Mathematica you're well-advised to use a notation like `N[-1,64]`, `N[4, 64]`, `N[1,64]`. The "4.0" will be read as a standard double-precision floating-point number. (Mathematica doesn't have a "Digits" command—each number has its own intrinsic precision.)

But you asked about Maple, not Mathematica, so this is just for your general edification...

Hi, upon simple testing this does not seem to work. i.e. setting `Digits:=1` and computing the inverse does not give different results from setting `Digits:=100` and computing the inverse.

Sometime ago I recall hearing that Maple was changing that way it dealt with linear algebra routines — maybe by using NAG routines (or similar)

Re: Maple's linear algebra routine and higher precision arithmetic

under the hood? If Maple is actually using compiled routines how could `Digits:=` whatever actually have an effect, I would presume the routines would be double precision by default and not be affected by a change of `Digits` (except possibly by rounding the input matrix). In my case the matrix is integer valued. In fact only a slight modification of the `tridiag(-1,4,-1)` matrix.

The environment variable `UseHardwareFloats` determines whether floating-point operations on Arrays are done in hardware (which is essentially double-precision) or software (with whatever setting of `Digits` is current). The default value is "deduced", meaning that hardware is used if `Digits <= evalhf(Digits)` (i.e. the precision of hardware floating-point, which is 14 on my Windows system), and software is used if `Digits > evalhf(Digits)`. Compiled NAG code is used for both hardware and software floating-point computations in `LinearAlgebra`.

Thus:

```
A:= <<7.0, 2.0>|<3.0, 5.0>>;
```

```
Digits:= 1;
```

```
A^(-1); # this will use hardware floats
```

```
[ 0.172413793103448260 -0.103448275862068950]
[ ]
[-0.0689655172413792956 0.241379310344827569 ]
```

```
UseHardwareFloats:= false:
```

```
A^(-1); # now it will use software floats with Digits = 1
```

```
[ 0.1 -0.06]
[ ]
[-0.04 0.2 ]
```

```
UseHardwareFloats:= deduced: Digits:= 40:
```

```
A^(-1); # this will use software floats with Digits = 40
```

```
[0.1724137931034482758620689655172413793104 ,
-0.1034482758620689655172413793103448275862]
```

Re: Maple's linear algebra routine and higher precision arithmetic

[-0.06896551724137931034482758620689655172416 ,

0.2413793103448275862068965517241379310345]

Robert Israel israel@xxxxxxxxxxx

Department of Mathematics <http://www.math.ubc.ca/~israel>

University of British Columbia Vancouver, BC, Canada