

Re: linalg[leastsqrs] in Maple V R4

Source: <http://sci.tech-archive.net/Archive/sci.math.symbolic/2006-08/msg00047.html>

- *From:* "Julian V. Noble" <jvn@xxxxxxxxxxxx>
 - *Date:* Mon, 31 Jul 2006 11:08:07 -0400
-

Robert Israel wrote:

In article <1154311713.046436.297030@xx>, lcw1964 <leslie.wright@xxxxxxxxxxxx> wrote:

I have decided to write my own leastsqrs routine using linalg(Svd) which seems to be a bit more consistent. This is how I have tested it out and have convinced myself it is adequate for my own purposes.

I execute `dvec:=evalf(linalg[Svd](u,Ubig,V))`. This returns a vector of the singular values, all of which are nonzero and not too small. To execute the least squares estimate as described well in section 2.6 of Numerical Recipes, I create the orthonormal "left" matrix `U:=evalm(delcols(Ubig, 10..72))`, and the diagonal matrix of inverses of the singular values by `isingmat:=diag(seq(1/dvec[i],i=1..9))`. The right matrix `V` is just what I need and can use it without modification. given this, the least squares solution is

```
sol:=evalm(V&*isingmat&*transpose(U));
```

I get the exact same result every time when I run everything after a restart, and the consistency is reassuring. The results are pretty close, at least in the first few digits, to the linalg[leastsqrs] output, though the consistency of the latter is not reassuring.

As a test, I attempt to reconstruct the original matrix `u` from my results, and compute its difference from the original:

```
evalm(U&*diag(seq(dvec[i],i=1..9))&*transpose(V);  
evalm("-u");
```

The output here reveals differences indicating that at most 2 or 3 SDs were lost. With `Digits:=60`, for example, this 72x9 matrix of differences has elements ranging between about $1E-57$ down to zero. For the stuff I am doing, this is tolerable and understandable.

I have no easy way of proving that the least squares result generated

Re: linalg[leastsqrs] in Maple V R4

by this approach is "truer" or "better" than that given by the built-in leastsqrs routine—according to the theory it is supposed to be the "right" answer—but I do seem to think that it is at least more consistent, and I will infer more accuracy and precision since the matrices and vectors generated by the Svd process that are prerequisites for the least squares calculation seem to maintain reasonable precision, as one can infer from the fairly close agreement when one reconstructs the original matrix from its decomposed parts.

This is an intermediate step in a larger algorithm (please refer to section 5.13 of Numerical Recipes if you are curious what I am trying to port into Maple), and I will let you know how I make out.

Les

It would be nice to know more about u and bb , but I think this is what is happening: leastsqrs works by solving the system $(u^T u) x = u^T bb$. I suspect that your matrix $u^T u$ is ill-conditioned (or perhaps singular), which can cause numerical problems here. Using Svd, as you have done, should work better.

I thought that the LinearAlgebra routine LeastSquares was more sophisticated, but an experiment showed that this is not necessarily the case. On the other hand, in LinearAlgebra you can use the pseudoinverse (obtained in LinearAlgebra as `MatrixInverse(..., method=pseudo)`) and obtain good results.

Robert Israel israel@xxxxxxxxxxx

Department of Mathematics <http://www.math.ubc.ca/~israel> University of British Columbia
Vancouver, BC, Canada

I am surprised that polynomial least-squares fits in Maple are not done using Gram polynomials. Is there some obvious reason NOT to use that method?

--

Julian V. Noble
Professor Emeritus of Physics
University of Virginia

.