Subject: Re: problem inverting matrix
Posted by Lars Schmidt on Mon, 06 Jan 2003 18:18:50 GMT
View Forum Message <> Reply to Message

Hello Paul,

thanks for the code, I will intensively test it.
Because of the fact that I generate a surface with that I can easily judge
if the inversion was o.k.
I have compared my results to MathCAD which I trust very much and I got big
deviations.
I will let you know about the results.
LARS.

"Paul van Delst" <paul.vandelst@noaa.gov> schrieb im Newsbeitrag
news:3E19BACA.DE05CB82@noaa.gov...
> Lars Schmidt wrote:
>>
>> Hello,
>>
>> I desperately hope someone of you has got an idea.
>> I try to invert a matrix using "invert" of course with floating-point
>> values.
>> I get like explosion of values although the status variable I used with
>> "invert" did not return any error and I also used the "/double" keyword.
>> Has anybody an idea or experience with that. Are these numerical
problems of
>> the gaussian ellemination??
>
> Matrix inversion can be tricky depending on what your data looks like. I
attached a piece
> of code I wrote a while back to use the IDL Numerical Recipes SVD
functionality to do the
> inversion. I never used it too much since the SVD stuff always gave me
slightly different
> answers than some fortran test code - I think that was tracked down to an
implementation
> problem of the NR SVD stuff in an earlier version of IDL that has since
been corrected.
> But I'm not sure. At any rate, test the attached code until you're blue in
the face before
> using it for anything important. Most of it is simply checking inputs and
whatnot, but
> still.....
>
> paulv
>

> p.s. Let me, uh, reiterate my suggestion to test this code if you use it for anything.....
> :o)
>
> --
> Paul van Delst
> CIMSS @ NOAA/NCEP/EMC
> Ph: (301)763-8000 x7274
> Fax:(301)763-8545
>
>
> ;---------------- cut here
> ;+
> ; NAME:
> ; svd_matrix_invert
> ;
> ; PURPOSE:
> ; To compute the pseudoinverse of a matrix
> ;
> ; CATEGORY:
> ; Linear Algebra
> ;
> ; CALLING SEQUENCE:
> ; result = svd_matrix_invert(A, check_inv = check_inv, $
> ;   double = double)
> ;
> ; INPUTS:
> ; A: MxN matrix to be inverted.
> ;
> ; KEYWORD PARAMETERS:
> ; check_inv: Set this keyword to plot out the surface of the
> ;   result matrix multiplied with the input. The plot
> ;   should be the identity matrix.
> ; double: Set this keyword to use double precision arithmetic.
> ;   This is recommended.
> ;
> ; OUTPUTS:
> ; The function returns the matrix inversion result.
> ;
> ; SIDE EFFECTS:
> ; None known.
> ;
> ; RESTRICTIONS:
> ; None known.
> ;
> ; PROCEDURE:
> ; The MxN input matrix is factorised into the form,
> ;

```
>  ;        A = U ## W ## transpose( V )
>  ;
>  ;      where U = MxM column-orthogonal matrix; the left singular vectors,
>  ;            W = NxN diagonal matrix of the singular values, and
>  ;            V = NxN orthogonal matrix; the right singular vectors.
>  ;
>  ;      The pseudoinverse of A, Ainv(+), can then be found from,
>  ;
>  ;        Ainv(+) = V ## Winv(+) ## transpose( U )
>  ;
>  ;      If A is square and non-singular, then Ainv(+) = Ainv where Ainv is
>  ;      the inverse of A.
>  ;
>  ;      The matrix Winv(+) is calculated by simply taking the reciprocal
of
>  ;      the singular values, EXCEPT where the singular values are less
than
>  ;      machine precision. Where singular values are less than machine
>  ;      precision, their reciprocal is simply set to zero.
>  ;
>  ; EXAMPLE:
>  ;      To calculate the inverse of a matrix A and plot the product Ainv
## A
>  ;      type:
>  ;
>  ;        Ainv = svd_matrix_invert( A, /check_inv )
>  ;
>  ; MODIFICATION HISTORY:
>  ;      Written by:     Paul van Delst, CIMSS/SSEC, 02-July-1997
>  ;
>  ; $Date: 1998/01/07 15:18:13 $
>  ; $Id: svd_matrix_invert.pro,v 1.4 1998/01/07 15:18:13 paulv Exp $
>  ; $Log: svd_matrix_invert.pro,v $
>  ; Revision 1.4  1998/01/07 15:18:13  paulv
>  ; IDL 5.0 array description used.
>  ; Category changed from Retrieval to Linear Algebra.
>  ; Tidied up informational MESSAGE output.
>  ;
>  ; Revision 1.3  1998/01/05 22:16:59  paulv
>  ; Plotting window created if CHECK_INV keyword set.
>  ;
>  ; Revision 1.2  1997/12/28 20:34:21  paulv
>  ; Added in warning message for when inverse singular values are
>  ; less than machine precision.
>  ; Put STOP statement in CHECK_INV IF block.
>  ;
>  ; Revision 1.1  1997/12/28 19:06:46  paulv
>  ; Initial revision
```

```
> ;
> ;
> ;
> ;-
>
> FUNCTION svd_matrix_invert, a, check_inv = check_inv, DOUBLE = DOUBLE
>
>
> ; ---------------------------------
> ; Check that the input matrix is 2-D
> ; ---------------------------------
>
>   sz = SIZE( a )
>
>   IF ( ( sz[ 0 ] NE 2 ) OR $
>        ( sz[ 1 ] EQ 1 ) OR $
>        ( sz[ 2 ] EQ 1 ) ) THEN BEGIN
>     MESSAGE, 'Invalid input matrix', /INFO
>     RETURN, -1
>   ENDIF
>
>   n = sz[ 1 ]
>
>
> ; ------------------
> ; Check the keywords
> ; ------------------
>
>   IF ( NOT KEYWORD_SET( check_inv ) ) THEN check_inv = 0
>
>   IF ( NOT KEYWORD_SET( DOUBLE ) ) THEN BEGIN
>     DOUBLE = 0
>     winv   = FLTARR( n, n )
>     zero   = 0.0
>     one    = 1.0
>   ENDIF ELSE BEGIN
>     DOUBLE = 1
>     winv   = DBLARR( n, n )
>     zero   = 0.0d
>     one    = 1.0d
>   ENDELSE
>
>
> ; --------------------
> ; Decompose the matrix
> ; --------------------
>
>   SVDC, a, w, u, v, DOUBLE = DOUBLE
```

```
>
>
>  ; ---------------------------------------------
>  ; Determine which singular values are too small and
>  ; which ones are ok.
>  ; ---------------------------------------------
>
>    loc_invalid = WHERE( w LE ( MACHAR( DOUBLE = DOUBLE ) ).EPS,
count_invalid )
>    loc_valid   = WHERE( w GT ( MACHAR( DOUBLE = DOUBLE ) ).EPS,
count_valid )
>
>
>  ; ---------------------------------------------
>  ; Invert the singular values, setting the too-small
>  ; ones to zero
>  ; ---------------------------------------------
>
>    IF ( count_valid GT 0 ) THEN w[ loc_valid ] = one / w[ loc_valid ]
>    IF ( count_invalid GT 0 ) THEN BEGIN
>      w[ loc_invalid ] = zero
>      MESSAGE, STRCOMPRESS( STRING( count_invalid ), /REMOVE_ALL ) + ' zero
singular
>  values', /INFO
>    ENDIF
>
>
>  ; -----------------------------------------------------
>  ; Check that the inverse of the singular values are valid
>  ; -----------------------------------------------------
>
>    loc_invalid = WHERE( w LE ( MACHAR( DOUBLE = DOUBLE ) ).EPS,
count_invalid )
>    IF ( count_invalid GT 0 ) THEN BEGIN
>      MESSAGE, STRCOMPRESS( STRING( count_invalid ), /REMOVE_ALL ) + $
>               ' inverse singular values < machine precision', /INFO
>    ENDIF
>
>
>  ; ------------------------------
>  ; Fill the singular value matrix
>  ; ------------------------------
>
>    i = INDGEN( n )
>    winv[ i, i ] = w
>
>
>  ; ---------------------------
```

```
>  ; Calculate the matrix inverse
>  ; ---------------------------
>
>    ainv = v ## winv ## TRANSPOSE( u )
>
>
>  ; ---------------------------
>  ; Plot the result of Ainv ## A
>  ; ---------------------------
>
>    IF ( check_inv NE 0 ) THEN BEGIN
>
>      IF ( !D.NAME EQ 'X' ) THEN BEGIN
>
>        WINDOW, /FREE, TITLE = 'SVD_MATRIX_INVERT check'
>        SURFACE, ainv ## a, /LEGO, $
>                 TITLE = 'Check of SVD matrix inversion', $
>                 XTITLE = 'Column index', YTITLE = 'Row index', $
>                 CHARSIZE = 1.5
>
>      ENDIF
>
>    ENDIF
>
>
>  ; -----------------
>  ; Return the result
>  ; -----------------
>
>    RETURN, ainv
>
>
>  END
```