Subject: Re: How to solve a homogeneous system(Ax=0) with a gauss elimination method that x is not zero.
Posted by marc schellens[1] on Wed, 05 Nov 2003 13:19:56 GMT
View Forum Message <> Reply to Message

Look at the
GS_ITER
function.

cheers,
marc

jhkim wrote:
> I would like to solve a homogeneous system (Ax=0) with non-trivial
> solution (x is not zero) using a Gauss elimination.
> Please, let me know how to make a program with IDL. A is a 44 * 44
> matrix.

Subject: Re: How to solve a homogeneous system(Ax=0) with a gauss elimination method that x is not zero.
Posted by planets on Thu, 06 Nov 2003 01:56:29 GMT
View Forum Message <> Reply to Message

Thank you for the advice. However, The function can't solve the problem.
Please,let me know another solution or correct my program.

My sample program is below.
=========
```
pro test

a=dblarr(3,3)
b=dblarr(3)
result=dblarr(3)

a=[[1,3,1],[3,4,5],[4,2,1]]
b[*]=1.0

result=gs_iter(a,b)

print, result

end
```

==========

Marc Schellens <m_schellens@hotmail.com> wrote in message

news:<3FA8F8FC.7040009@hotmail.com>...
> Look at the
> GS_ITER
> function.
>
> cheers,
> marc
>
> jhkim wrote:
>> I would like to solve a homogeneous system (Ax=0) with non-trivial
>> solution (x is not zero) using a Gauss elimination.
>> Please, let me know how to make a program with IDL. A is a 44 * 44
>> matrix.

Subject: Re: How to solve a homogeneous system(Ax=0) with a gauss elimination method that x is not zero.
Posted by Mark Hadfield on Thu, 06 Nov 2003 04:07:42 GMT
View Forum Message <> Reply to Message

jhkim wrote:
> Thank you for the advice. However, The function can't solve the problem.
> Please,let me know another solution or correct my program.
>
> My sample program is below.
> =========
> pro test
>
> a=dblarr(3,3)
> b=dblarr(3)
> result=dblarr(3)
>
> a=[[1,3,1],[3,4,5],[4,2,1]]
> b[*]=1.0
>
> result=gs_iter(a,b)
>
> print, result
>
> end

If you set the CHECK keyword, then GS_ITER will report the useful information that:

Input matrix is not in Diagonally Dominant form.
Algorithm may not converge.

This seems to be your problem.

I have forgotten what little I ever knew about solving matrix equations
with IDL, but I recall that the more robust solution techniques involve
a decomposition of A. For example LU decomposition (LUDC or LA_LUDC,
LUSOL or LA_LUSOL), Cholesky decomposition (CHOLDC or LA_CHOLDC, CHOLSOL
or LA_CHOLSOL, only for positive-definite A) and singular-value
decomposition (SVDC or LA_SVD, SVDSOL).

For what it's worth, here is an SVD example I wrote for myself some time
ago. Note that A is not square: SVD can be used for over-determined or
under-determined sets of equations. This makes it good for
hack-it-and-see mathematicians like me, who like to get a solution even
if it's wrong.

```
pro mgh_example_matrix_svd, TRANSPOSE=transpose

   compile_opt IDL2

   a = [[1.0, 2.0, -1.0, 2.5], $
        [1.5, 3.3, -0.5, 2.0], $
        [3.1, 0.7,  2.2, 0.0], $
        [0.0, 0.3, -2.0, 5.3], $
        [2.1, 1.0,  4.3, 2.2], $
        [0.0, 5.5,  3.8, 0.2]]

   if keyword_set(transpose) then a = transpose(a)

   print, 'a:'
   print, a

   la_svd, a, w, u, v

   print, 'u:'
   print, u

   print, 'v:'
   print, v

   ;; Zero small elements of w

   small = where(w lt 1.E-6*max(w), n_small)
   if n_small gt 0 then w[small] = 0

   print, 'w:'
   print, w

   ;; Recreate original matrix
```

```
    aa = u ## diag_matrix(w) ## transpose(v)

    print, 'max(abs(aa-a))'
    print, max(abs(aa-a))

end
```

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)