Subject: Re: 3d matrices and LUSOL
Posted by the_cacc on Thu, 24 Jan 2002 23:22:30 GMT
View Forum Message <> Reply to Message

"M" <mrmanish@bigfoot.com> wrote in message news:<a2pj2t$jv8$1@yarrow.open.ac.uk>...
> Hi all,
>
> I am in desperate need of help!!
>
> I have a set of linear equations represented as matrix arrays which need to
> be solved using the LU decomposition technique. The two arrays consist of a
> 14 x 14 array, and a 14 x 1 array, but each element in the matrices itself
> is an array of 221 elements (ie the matrices are 3dimensional...?)
>
> So i need to solve the system using LUDC and LUSOL, but i have to do it 221
> times (ie a solution for each 'layer' of the matrices)
>
> The first question is, is there a way to declare the two input arrays as
> 3-d? I tried defining the matrix using matrix=[[a,b,..],[...,...,...] etc]
> where a,b,... = arrays, but this isn't recognised as a 14 x14 square matrix
> which is 221 elements 'deep'. Instead, it expands each array across the
> row, making it a 3094 x 14 matrix. (it needs to be square to run LUDC)
>
> Is there a way i can force IDL to see it as a 'layered' 3-d matrix?
>
> What i need to achieve is a 3 dimensional 14 x 1 solution array, again 221
> elements 'deep'. To get this, could i simply run LUDC and LUSOL as normal
> provided the inputs are 3d matrices, or do i need to somehow loop the
> procedures so it produces solutions one 'layer' at a time and builds them
> into the 3 d solution matrix ?
>
> I don't know if anyone has any idea of what i'm talking about, but i only
> just understand it myself!!
> Apologies if it makes no sense whatsoever, suffice to say i'm a little
> confused right now!
>
>
> Any insight into the above would be great,
>
> thanks,
>
> Manish.

Hi,

From what you say, it sounds like you want to be solving the 14x14
problem 221 times, ie.

For the first 14x14 matrix (A1) and 14x1 vector (b1), solve for x1:
x1 = INVERT(A1) ## b1

Then for the second case: x2 = INVERT(A2) ## b2

and so on 221 times. You'll then have 221 14x1 vectors x1,x2,... which
together give a 2D matrix (221x14).

Is this what you were expecting as your answer ? If so, then yay! If
not, I haven't understood the problem :(

Of course, you can define the individual matrices in larger matrices,
ie. A = FLTARR(14,14,221), b = FLTARR(14,221) and x = FLTARR(14,221)
then loop as follows:

```
FOR i = 0, 220 DO x[*,i] = INVERT(A[*,*,i]) ## b[*,i]
```

NOTE: using LUDC and LUSOL directly is slightly faster than using
INVERT, but you may prefer INVERT initially since it makes the code
simpler.

Ciao.

Subject: Re: 3d matrices and LUSOL
Posted by Craig Markwardt on Thu, 24 Jan 2002 23:22:57 GMT

View Forum Message <> Reply to Message

"M" <mrmanish@bigfoot.com> writes:
> Hi all,
>
> I am in desperate need of help!!
>
> I have a set of linear equations represented as matrix arrays which need to
> be solved using the LU decomposition technique. The two arrays consist of a
> 14 x 14 array, and a 14 x 1 array, but each element in the matrices itself
> is an array of 221 elements (ie the matrices are 3dimensional...?)
>
> So i need to solve the system using LUDC and LUSOL, but i have to do it 221
> times (ie a solution for each 'layer' of the matrices)
>
> The first question is, is there a way to declare the two input arrays as
> 3-d? I tried defining the matrix using matrix=[[a,b,..],[...,...,...] etc]
> where a,b,... = arrays, but this isn't recognised as a 14 x14 square matrix
> which is 221 elements 'deep'. Instead, it expands each array across the
> row, making it a 3094 x 14 matrix. (it needs to be square to run LUDC)
>

> Is there a way i can force IDL to see it as a 'layered' 3-d matrix?

First of all, IDL can do up to eight dimensions. Since you mention
the notation [[a,b,...]], you are probably not getting the syntax
exactly right. It doesn't really matter though. If you really have a
3094 x 14 matrix, it is straightforward to use REFORM to make that
221x14x14 matrix (ie, matrix = reform(matrix,221,14,14))

I am pretty sure that LUDC/LUSOL will not handle a "3-d" matrix. Can
your solution be applied component by component? I.e., can you solve
each of the 221 14x14 matrix equations separately using a FOR loop?

Good luck,
Craig

--
 -- --------------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 -- --------------

Subject: Re: 3d matrices and LUSOL
Posted by Manish on Fri, 25 Jan 2002 16:13:59 GMT
View Forum Message <> Reply to Message

Cheers guys, that should sort it - it was exactly what I needed!

I'll give it a go with INVERT first with the looped arrays, maybe try and
refine it to LUSOL later on if I get I working!

Both of you, thanks for your help again

Manish.

"trouble" <the_cacc@hotmail.com> wrote in message
news:5f9f0a23.0201241522.7f538c34@posting.google.com...
> "M" <mrmanish@bigfoot.com> wrote in message
news:<a2pj2t$jv8$1@yarrow.open.ac.uk>...
>> Hi all,
>>
>> I am in desperate need of help!!
>>
>> I have a set of linear equations represented as matrix arrays which need

to
>> be solved using the LU decomposition technique. The two arrays consist
of a
>> 14 x 14 array, and a 14 x 1 array, but each element in the matrices
itself
>> is an array of 221 elements (ie the matrices are 3dimensional...?)
>>
>> So i need to solve the system using LUDC and LUSOL, but i have to do it
221
>> times (ie a solution for each 'layer' of the matrices)
>>
>> The first question is, is there a way to declare the two input arrays as
>> 3-d? I tried defining the matrix using matrix=[[a,b,..],[...,...,...]
etc]
>> where a,b,... = arrays, but this isn't recognised as a 14 x14 square
matrix
>> which is 221 elements 'deep'. Instead, it expands each array across the
>> row, making it a 3094 x 14 matrix. (it needs to be square to run LUDC)
>>
>> Is there a way i can force IDL to see it as a 'layered' 3-d matrix?
>>
>> What i need to achieve is a 3 dimensional 14 x 1 solution array, again
221
>> elements 'deep'. To get this, could i simply run LUDC and LUSOL as
normal
>> provided the inputs are 3d matrices, or do i need to somehow loop the
>> procedures so it produces solutions one 'layer' at a time and builds
them
>> into the 3 d solution matrix ?
>>
>> I don't know if anyone has any idea of what i'm talking about, but i
only
>> just understand it myself!!
>> Apologies if it makes no sense whatsoever, suffice to say i'm a little
>> confused right now!
>>
>>
>> Any insight into the above would be great,
>>
>> thanks,
>>
>> Manish.
>
> Hi,
>
> From what you say, it sounds like you want to be solving the 14x14
> problem 221 times, ie.
>

> For the first 14x14 matrix (A1) and 14x1 vector (b1), solve for x1:
> x1 = INVERT(A1) ## b1
>
> Then for the second case: x2 = INVERT(A2) ## b2
>
> and so on 221 times. You'll then have 221 14x1 vectors x1,x2,... which
> together give a 2D matrix (221x14).
>
> Is this what you were expecting as your answer ? If so, then yay! If
> not, I haven't understood the problem :(
>
> Of course, you can define the individual matrices in larger matrices,
> ie. A = FLTARR(14,14,221), b = FLTARR(14,221) and x = FLTARR(14,221)
> then loop as follows:
>
> FOR i = 0, 220 DO x[*,i] = INVERT(A[*,*,i]) ## b[*,i]
>
>
> NOTE: using LUDC and LUSOL directly is slightly faster than using
> INVERT, but you may prefer INVERT initially since it makes the code
> simpler.
>
> Ciao.

Subject: Re: 3d matrices and LUSOL
Posted by James Kuyper on Fri, 25 Jan 2002 17:18:07 GMT
View Forum Message <> Reply to Message

M wrote:

> Hi all,
>
> I am in desperate need of help!!
>
> I have a set of linear equations represented as matrix arrays which need to
> be solved using the LU decomposition technique. The two arrays consist of a
> 14 x 14 array, and a 14 x 1 array, but each element in the matrices itself
> is an array of 221 elements (ie the matrices are 3dimensional...?)
>
> So i need to solve the system using LUDC and LUSOL, but i have to do it 221
> times (ie a solution for each 'layer' of the matrices)
>
> The first question is, is there a way to declare the two input arrays as
> 3-d? I tried defining the matrix using matrix=[[a,b,..],[...,...,...] etc]
> where a,b,... = arrays, but this isn't recognised as a 14 x14 square matrix
> which is 221 elements 'deep'. Instead, it expands each array across the
> row, making it a 3094 x 14 matrix. (it needs to be square to run LUDC)

>
> Is there a way i can force IDL to see it as a 'layered' 3-d matrix?

Yes. If 'a', 'b', 'c' etc are 221-element arrays, then

```
   [ [[a],[b]], [[c],[d]] ]
```

defines a (221,2,2) array. If you need a different ordering of indices
(I suspect that you want (2,2,221)), then you need to use transpose()
and reform(). In APL, which is part of IDL's ancestry, the equivalent of
transpose() could hanndle arrays of any dimension. However, in IDL
transpose() won't work on arrays of rank higher than 2. Therefore, do
the following:

```
big = reform(transpose([[a],[b],[c],[d]]),2,2,221)
```

You'll need to call LUDC and LUSOL seperately for each of the 221
layers. Therefore, if there's any advantage to forming them all into one
big array, that advantage will have to lie in some other part of the code.