Subject: singular value decompostion
Posted by Dave Bazell on Thu, 01 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

I am trying to use the IDL routine SVDC to do principal component analysis. In order to understand SVD better I was doing an example I found online. However, the IDL SVD routine gives me different results than the online example.

x = [[1,2],[3,4],[5,6],[7,8]]

matlab, which uses linpac gives (to two decimal places):

[U,S,V] = svd(x) where X = U S transpose(V)

U = .15 .82 -.39 -.38
.35 .42 .24 .80
.55 .02 .70 .46
.74 -.38 -.54 .04

S = 14.3 0
0 .62

V = .64 -.77
.77 .64

IDL gives

svdc, x,w,u,v,/column

w = 14.2691 0.626828

u = -0.641423 -0.767187
-0.767187 0.641423
0.00000 0.00000
0.00000 0.00000

v = -0.152483 -0.349918 -0.547354 -0.744789
0.822647 0.421375 0.0201032 -0.381169
0.547723 -0.730297 -0.182574 0.365149
0.00000 0.408249 -0.816496 0.408248

clearly the eigenvalues are the same but the u and v matricies are exchanged. But what really bothers me is that some values are changed from positive to negative. And the IDL V does not have the same values as the MATLAB U.

What am I doing wrong? Even if I leave out the /column in the call to

svdc, I don't get the right answers.

The eigenvalues do not correspond to the eigenvalues returned by the IDL routine pcomp which calculates principal components. I thought PCA could be done using SVD but I don't see the correspondence.

Any help would be appreciated.

Thanks.

Dave
bazell@home.com

Subject: Re: singular value decompostion
Posted by Dave Bazell on Thu, 01 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

H T Onishi wrote:

> This is not a complete answer but perhaps adds some insight.
>
> First of all I believe that your U from Matlab is in error. The .46 should
> be -.46.
>
> Second, if you define x as below and then do svdc,x,w,u,v,/col and then
> stuff w into the diagonal elements of a 4x4 array, say ww, and finally
> compute u # ww # transpose(v), you will get x back.
>
> x = u # ww # transpose(v)
>
> as it should.
>
> Unfortunately the # operator in IDL does the row/col reversal from
> "standard" matrix multiplication.
>
> However, if you compute transpose(v) ## w ## u then you will also get x
> back. Since ## computes the "standard" matrix multiply this means that
>
> u from Matlab = transpose(v) from IDL
> v from Matlab = transpose(u) from IDL
>
> both of which are true for the non-singular vectors with the exception of
> sign differences. I don't think that the sign differences are important
> since these vectors will span the same subspaces of a 4 dimensional real
> vector space.
>
> Finally, regarding the column vectors in U from Matlab which do not

> correspond to the column vectors in transpose(V) from IDL, these vectors
> span the two dimensional null space for this particular linear
> transformation and they are therefore arbitrary within this null space as
> long as they are normalized and orthogonal. Another way of putting this is
> that the two null col vectors in U can be combined linearly to give the two
> col vectors in transpose(V). Try this out. (This is how I decided that the
> .46 should be -.46)
>
> Howard Onishi

Thanks, that clarifies some points. You were correct, I copied down a .46
rather than a -.46.

I will run through your exercise to make sure I understand correctly.

Do you have any experience with Principal component analysis and how that can
be done with SVD?

Thanks again,

Dave

Subject: Re: singular value decompostion
Posted by fogel on Thu, 01 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

H T Onishi (htonishi@home.com) wrote:
: This is not a complete answer but perhaps adds some insight.
:
: First of all I believe that your U from Matlab is in error. The .46 should
: be -.46.

Doubtful. The SVD is not unique in the sense that
any U*S*V' = A, within roundoff of A, is a correct
solution.

One can search the archives of comp.soft-sys.matlab
and find a posting by Cleve Moler (Matlab creator,
linear algebra guru, etc.) stating as much. Try
looking for 'SVD' about two years ago.

[snip...]

HTH.

- David

Subject: Re: singular value decompostion
Posted by H T Onishi on Fri, 02 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

One more suggestion. If you want a nice introductory discussion about SVD look in the Numerical Recipes book by Press, et. al. or go to http://cfata2.harvard.edu/nr/ where the book is online. It's fun reading if you're a vector space kind of guy.

Also, regarding Matlab, whenever any of my young engineers or scientists asks about whether to use Matlab or IDL I always recommend Matlab for hard core numerical analysis. IDL's strength is definitely not numerical analysis. Matlab is highly recommended by the numerical analysis community. I use IDL extensively though for quick analyses that require graphics output and it's mapping tools are also very nice. But when I'm ready for the real thing there's nothing like Fortran! :-) If you're into Fortran try Netlib.

Howard Onishi

Subject: Re: singular value decompostion
Posted by H T Onishi on Fri, 02 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

> Thanks, that clarifies some points. You were correct, I copied down a .46
> rather than a -.46.
>
> I will run through your exercise to make sure I understand correctly.
>
> Do you have any experience with Principal component analysis and how that
can
> be done with SVD?
>
> Thanks again,
>
> Dave
>

I am not an expert in the SVD algorithm. The note by Fogel should be heeded and you may want to search the appropriate newsgroup for more info. I believe that the SVD algorithm implemented in IDL was taken from Numerical Recipes and there seems to be a lot of controversy over those routines. You may want to read http://math.jpl.nasa.gov/nr/nr-alt.html for more info. I know that I had to "tweak" an early version of the SVD routine from Numerical Recipes to get it to work properly -- problems with underflow.

Regarding Principal Component Analysis, I believe that SVD is the key algorithm used to extract the PC's. I know that it has been used with some

success to remove background clutter from imagery that contains moving targets. Again I am not an expert here (in fact not even a novice) but from what I understand a set of images of the same scene -- possibly multi-spectral -- is combined into a large matrix. Each image is turned into a vector and the set of vectors is combined into a large matrix. These vectors span a vector space and the PCs corresponding to the largest singular values represent clutter vectors that can then be subtracted from the original image. I think the PCs are in the V matrix, but there is a 50% chance that I'm wrong about that. That's about all I know. I'm sure there are many more applications. I suggest you do a search with AltaVista to see what you can find! Or you can be more conventional and do a literature search, which might be more fruitful.

Howard
