Subject: Re: Sinagular Value in Complex Array
Posted by Lars G. Hanson on Tue, 29 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Hi Javier,

Complex singular value decomposition can be expressed in terms of decomposition of real matrices. I have included some code I wrote a while ago. pinv() computes the pseudo-inverse (you can use the RSI-supplied SVD routines for this, but I remember my version as being faster -- it does SVD by finding eigenvectors of the 'small' covariance matrix). pinv_complex() uses pinv() for calculating the pseudo inverse of a complex matrix.

Have fun, /Lars

Lars G. Hanson, DRCMR

On Tue, 29 Jun 1999, gabriel rodriguez ibeas wrote:

> I'm a student from ETSI Telecomunicaciones of Madrid and my
> name is Javier.
> I'd like to calculate the singular value of a complex array but
>
> I can't use svdc of IDL because this procedure return singular value of
>
> the real part.
> I'm looking for some procedure to resolve this problem with
> complex data and for this reason I require your help.
>
>
>

```
function pinv_complex, M, rel_thresh=rel_thresh
  if not keyword_set(rel_thresh) then rel_thresh=1.e-8
  xdim=(size(M))(1)
  ydim=(size(M))(2)
  M2=dblarr(2*xdim,2*ydim)
  M2(0:xdim-1,0:ydim-1)=double(M)
  M2(xdim:2*xdim-1,0:ydim-1)=-imaginary(M)
  M2(0:xdim-1,ydim:2*ydim-1)=imaginary(M)
  M2(xdim:2*xdim-1,ydim:2*ydim-1)=double(M)
  catch, Error_status
  if Error_status ne 0 then begin
;      print, !err_string
     Minv=0.*transpose(M)
  end else begin
```

```
    M2inv=pinv(M2, /double, rel_thresh=rel_thresh)
     Minv=M2inv(0:ydim-1,0:xdim-1)-dcomplex(0,1)*M2inv(ydim:2*ydi m-1,0:xdim-1)
  end
  return, Minv
end

function pinv, A, U=U, V=V, rel_thresh=rel_thresh, double=double, $
              principal=principal
  if not keyword_set(double) then double=1
  if not keyword_set(rel_thresh) then rel_thresh=1e-8
  ncol=(size(a))(1)
  nrow=(size(a))(2)
  nelem=(ncol < nrow)
  gamsqr_inv=make_array(nelem,nelem, type=(size(A))(3), value=0.)
  principal=reform(gamsqr_inv(0,*))
  if (nrow le ncol) then begin
     covar=A##transpose(A)
     trired,  covar, eigenvals, E, double=double
     triql, eigenvals, E, covar, double=double
     U=transpose(covar)
     absthresh=rel_thresh^2*max(eigenvals)
     for i=0,nelem-1 do $
       if (eigenvals(i) gt absthresh) then $
       gamsqr_inv(i,i)=1./eigenvals(i)
     Ainv=transpose(A)##U##gamsqr_inv##transpose(U)
  end else begin
     covar=transpose(A)##A     ; covariance
     trired,  covar, eigenvals, E, double=double
     triql, eigenvals, E, covar, double=double
     V=transpose(covar)
     absthresh=rel_thresh*max(eigenvals)
     for i=0,nelem-1 do $
       if (eigenvals(i) gt absthresh) then $
       gamsqr_inv(i,i)=1./eigenvals(i)
     Ainv=V##gamsqr_inv##transpose(V)##transpose(A)
  end
  principal=sqrt(eigenvals)*(eigenvals gt absthresh)
  return, Ainv
end
```