Subject: Re: Principle Componets Analysis
Posted by yp on Fri, 24 Aug 2007 10:24:47 GMT

View Forum Message <> Reply to Message

On Aug 24, 11:21 am, Yaswant Pradhan <Yaswant.Prad...@gmail.com>
wrote:
> On Aug 24, 3:59 am, David Fanning <n...@dfanning.com> wrote:
>
>
>
>> kBob writes:
>>>   I find the book Image Analysis, Classification and Change Detection
>>> in Remote Sensing, with Algorothms for ENIV/IDL by Morton Canty handy.
>>> He provides ENVI/IDL code to do the PCAs.
>
>> Well, I'm ashamed to say, I had read part's of Mort's book
>> earlier in the week and found I needed, well, more remedial
>> help. Quite frankly, I didn't understand a word of it. :-(
>
>> The Lindsay Smith tutorial, on the other hand, was crystal
>> clear. So much so that I came back to my office and wrote up
>> the example in IDL, just to see if I could follow it.
>
>> It turns out, that the PCOMP function in IDL gives essentially
>> the same answer as the tutorial (this for Jeff's benefit), but
>> the values are scaled slightly differently. However they
>> plot on exactly the same line in the end. Here is the code
>> I used.
>
>> ; Method according to the Lindsay Smith tutorial:
>> ;http://tinyurl.com/3aaeb
>
>> x = [2.5, 0.5, 2.2, 1.9, 3.1, 2.3, 2.0, 1.0, 1.5, 1.1]
>> y = [2.4, 0.7, 2.9, 2.2, 3.0, 2.7, 1.6, 1.1, 1.6, 0.9]
>
>> xmean = x - Mean(x)
>> ymean = y - Mean(y)
>> Window, XSIZE=600, YSIZE=800
>> !P.MULTI=[0,1,2]
>> Plot, xmean, ymean, PSYM=7
>
>> dataAdjust = Transpose([[xmean], [ymean]])
>> covArray = Correlate(dataAdjust, /COVARIANCE, /DOUBLE)
>> eigenvalues = EIGENQL(covArray, EIGENVECTORS=eigenvectors, /DOUBLE)
>
>> Print, 'EIGENVALUES: ', eigenvalues
>> Print, 'EIGENVECTORS: '
>> Print, eigenvectors

```
>
>>  rowFeatureVector = eigenvectors[0,*] ; Take first principle component.
>>  ;rowFeatureVector = eigenvectors
>>  finalData = Transpose(rowFeatureVector) ## Transpose(dataAdjust)
>>  Plot, finaldata+Mean(x), finaldata+mean(y), PSYM=7
>>  !P.MULTI=0
>
>>  ; Method using PCOMP in IDL library.
>>  data = Transpose([[x],[y]])
>>  r = PCOMP(data, /COVARIANCE, NVARIABLES=1, EIGENVALUES=ev, /STANDARDIZE)
>>  Print, 'IDL EIGENVALUES: ', ev
>
>>  ; Compare methods.
>>  Window, 1
>>  PLOT, r
>>  OPLOT, finalData, LINESTYLE=2
>
>>  Window, 2
>>  PLOT, r + Mean(x), r + Mean(y), PSYM=2
>>  OPLOT, finalData + Mean(x), finalData + Mean(y), PSYM=7
>>  END
>
>>  This is really nice stuff and has me EXTREMELY jazzed about
>>  the potential of it. :-)
>
>>  Cheers,
>
>>  David
>>  --
>>  David Fanning, Ph.D.
>>  Fanning Software Consulting, Inc.
>>  Coyote's Guide to IDL Programming:http://www.dfanning.com/
>>  Sepore ma de ni thui. ("Perhaps thou speakest truth.")
>
>  Hi David,
>  Yes, both methods are essentially same except that the data in
>  Method#1 are NOT standardised. You will get exactly same result if you
>  do
>  xmean = (x - Mean(x) / Stddev(x)
>  ymean = (y - Mean(y) / STddev(y)
>
>  --yas
```

whoops... missed a parenthesis, should read xmean = (x - Mean(x)) / Stddev(x) and likewise.