Subject: Re: Principle Componets Analysis
Posted by yp on Fri, 24 Aug 2007 17:30:48 GMT
View Forum Message <> Reply to Message

On Aug 24, 4:46 pm, David Fanning <da...@dfanning.com> wrote:
> Yaswant Pradhan writes:
>> 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)
>
> Well, not exactly. Did you run the example with this change?
> I get something quite a bit different, although still "correct"
> I think.
>
> ;** ***
> ; Method according to the Lindsay Smith tutorial:
> ;http://tinyurl.com/3aaeb6
>
> 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)) / STDDEV(x, /DOUBLE)
> ymean = (y - Mean(y)) / STDDEV(y, /DOUBLE)
> 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;, COLOR=FSC_Color('yellow')
>
> Window, 2
> PLOT, r + Mean(x), r + Mean(y), PSYM=2
> OPLOT, finalData + Mean(x), finalData + Mean(y), $
>   PSYM=7;, COLOR=FSC_Color('yellow')
> END
> ;** ***
>
> The curves in Window 1 are worse than they were without
> making the change you suggest.
>
> Cheers,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming:http://www.dfanning.com/

Not exactly, what I was looking at in your code is whether you get the
same eigenvalues/vecors or not. I've changed the code slightly for
dimension compatibility. What I would look at to compare both methods
is - (i) eigenvalues (maginitude), (ii) sign (direction) of the
components.

```
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)) / STDDEV(x, /DOUBLE)
ymean = (y - Mean(y)) / STDDEV(y, /DOUBLE)
;Window, XSIZE=600, YSIZE=800
;!P.MULTI=[0,1,2]
;Plot, xmean, ymean, PSYM=7


dataAdjust = Transpose([ [xmean], [ymean] ])
;dataAdjust = ([ [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 = transpose(eigenvectors)
finalData = (dataAdjust) ## (rowFeatureVector)
;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, EIGENVALUES=ev, /STANDARDIZE, /DOUBLE);,
NVARIABLES=1)
Print, 'IDL EIGENVALUES: ', ev


; Compare methods.
Window, 1, title='1st component'
PLOT, r[0,*]
OPLOT, finalData[0,*], LINESTYLE=2;, COLOR=FSC_Color('yellow')
OPLOT, [0,10],[0,0]


Window, 2, title='2nd component'
PLOT, finalData[1,*], LINESTYLE=2;, COLOR=FSC_Color('yellow')
OPLOT, r[1,*]
OPLOT, [0,10],[0,0]
```