Applied Machine Learning for Business and Economics

AMLBE

Profesor : Raúl Muñoz R.

Análisis de Componentes Principales (PCA) y Reducción de Dimensionalidad

- El mercado presenta por si mismo al observador una «superficie» de inconmensurable superficie de datos y movimientos. Los links entre esos datos y movimientos, muchos de los cuales presentan mecanismos sistemáticos ocultos a simple vista

- Uno de nuestros objetivos será, por medio de reducción de dimensionalidad y PCA entender la estructura core del mercado (lo cual es extrapolable a cualquier tipo de problema donde la multidimensionalidad es el problema)

- EL análisis de componentes principales, PCA, tiene como principal supuesto, que el mercado esta dirigido por un set de factores lineales no correlacionados, lo cual puede ser utilizado a nuestro favor para encontrar relaciones entre variables de mercado y variables económicas, generando un profundo entendimiento de las fuerzas conductoras detrás de, entre otras cosas, las curvas de rendimiento

- El objetivo de entender e identificar unos pocos mecanismos claves detrás de todos los movimientos de mercado puede ser matemáticamente abordado extrayendo la información más importante de un determinado set de datos de mercado

- Expresando esto un poco mas formalmente, esto significa reducir la dimensionalidad , con las dimensiones remanentes conteniendo el resto de la información

- Así, el resultado de este ejercicio revelará: el numero, la fuerza y forma de los mecanismos de mercado

- Dado que el PCA es una herramienta del algebra lineal, necesitamos primero representar el mercado en forma de matriz . La forma directa es por lo tanto expresar la información estructural contenida en el mercado por medio del análisis de matriz de covarianza, procediendo bajos los siguientes pasos:

1. Normalización de las variables llevándolas a z-scores
2. Transformar la matriz de covarianza en la base ortonormal de sus eigenvectors, recuerde que:

Si $Ax = \lambda x\ (x \neq 0), para\ una\ matriz\ A, luego\ el\ vector\ x\ es\ llamado\ un\ eigenvector\ de\ A$

$y\ el\ número\ \lambda\ es\ el\ asociado\ eigenvalue\ de\ A$

3. Teorema: Para toda matriz de covarianza *Cov, es cierto que,*

$$Cov = B^{-1}\begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \dots & 0 \\ 0 & 0 & \lambda_n \end{pmatrix}B$$

Donde λ_i corresponde aa los eigenvalues de Cov, mientras que la matriz B consisten en los eigenvectores de Cov

- Interpretación intuitiva: la matriz B actúa como una transformación del sistema de coordenadas y nos permite considerar la matriz de covarianza de una base ortonormal dada por sus eigenvectors. Dado que estos eigenvectors son ortogonales ellos descomponen la matriz de covarianza en relaciones descorrelacionadas , además el eigenvector asociado con el mayor eigenvalue en valor absoluto representaría la relación estructural más importante en el dataset de mercado…

- Veamos un ejemplo: Analicemos la database de la curva swap chilena comprendida entre el 16 de enero de 2017 hasta el 25 de setpiembre de 2017

	CHSWP2 ICCH Curncy	CHSWP3 ICCH Curncy	CHSWP4 ICCH Curncy	CHSWP5 ICCH Curncy	CHSWP6 ICCH Curncy	CHSWP7 ICCH Curncy	CHSWP8 ICCH Curncy	CHSWP9 ICCH Curncy	CHSWP10 ICCH Curncy
16-01-2017	2.98	3.15	3.36	3.58	3.75	3.89	4	4.1	4.16
17-01-2017	2.95	3.11	3.32	3.54	3.72	3.86	3.98	4.07	4.14
18-01-2017	2.95	3.11	3.31	3.54	3.72	3.86	3.99	4.08	4.15
19-01-2017	2.96	3.12	3.33	3.55	3.74	3.87	4.01	4.1	4.17
20-01-2017	2.93	3.11	3.33	3.55	3.73	3.86	4.01	4.09	4.16
23-01-2017	2.92	3.09	3.32	3.55	3.72	3.85	4	4.08	4.15
24-01-2017	2.95	3.13	3.35	3.58	3.74	3.88	4.01	4.1	4.17
25-01-2017	2.97	3.17	3.38	3.59	3.75	3.9	4.02	4.12	4.19
26-01-2017	2.99	3.2	3.42	3.63	3.78	3.93	4.05	4.15	4.22
27-01-2017	2.99	3.21	3.43	3.65	3.79	3.94	4.06	4.16	4.23
30-01-2017	2.99	3.21	3.43	3.65	3.8	3.94	4.07	4.17	4.23
31-01-2017	2.97	3.19	3.4	3.61	3.77	3.91	4.05	4.13	4.19
01-02-2017	2.98	3.21	3.42	3.62	3.78	3.92	4.06	4.14	4.2

- **Paso 1:** Normalización de la data:

- x_mean=np.mean(x,axis=0)
- x_desv=np.std(x,axis=0)
- x_norm=(x-x_mean)/x_desv

x_norm - DataFrame

Index	ISWP2 ICCH Curn	ISWP3 ICCH Curn	ISWP4 ICCH Curn	ISWP5 ICCH Curn	ISWP6 ICCH Curn	ISWP7 ICCH Curn	ISWP8 ICCH Curn	ISWP9 ICCH Curn	SWP10 ICCH Curr
2017-01-16 00:00:00	1.134143667	0.8644305778	0.7694794004	0.8755695425	0.8683615069	0.7386374139	0.6746417168	0.7642980895	0.6705434751
2017-01-17 00:00:00	0.9642174343	0.5884472374	0.4584703048	0.5392580626	0.5878635449	0.4215578662	0.437734232	0.3864916475	0.4155171042
2017-01-18 00:00:00	0.9642174343	0.5884472374	0.3807180309	0.5392580626	0.5878635449	0.4215578662	0.5561879744	0.5124271282	0.5430302896
2017-01-19 00:00:00	1.020859512	0.6574430725	0.5362225787	0.6233359326	0.7748621862	0.5272510488	0.7930954592	0.7642980895	0.7980566605
2017-01-20 00:00:00	0.8509332793	0.5884472374	0.5362225787	0.6233359326	0.6813628655	0.4215578662	0.7930954592	0.6383626089	0.6705434751
2017-01-23 00:00:00	0.7942912018	0.4504555671	0.4584703048	0.6233359326	0.5878635449	0.3158646836	0.6746417168	0.5124271282	0.5430302896
2017-01-24 00:00:00	0.9642174343	0.7264389076	0.6917271265	0.8755695425	0.7748621862	0.6329442313	0.7930954592	0.7642980895	0.7980566605
2017-01-25 00:00:00	1.077501589	1.002422248	0.9249839483	0.9596474125	0.8683615069	0.8443305965	0.9115492016	1.016169051	1.053083031
2017-01-26	1.190785744	1.209409753	1.235993044	1.295958892	1.148859469	1.161410144	1.266910429	1.393975493	1.435622588

Formato | Redimensionar | Color de fondo | Min/max de columna | OK | Cancel

```
       CHSWP2 ICCH Curncy  CHSWP3 ICCH Curncy  CHSWP4 ICCH Curncy
count        1.740000e+02        1.740000e+02        1.740000e+02
mean         4.492575e-15        5.430522e-15       -9.835045e-15
std          1.002886e+00        1.002886e+00        1.002886e+00
min         -1.754602e+00       -1.688415e+00       -1.640841e+00
25%         -7.350449e-01       -7.224736e-01       -8.438803e-01
50%         -1.686241e-01       -1.705069e-01       -2.413002e-01
75%          1.006699e+00        9.161775e-01        9.249839e-01
max          1.813849e+00        1.968364e+00        1.935764e+00

       CHSWP5 ICCH Curncy  CHSWP6 ICCH Curncy  CHSWP7 ICCH Curncy
count        1.740000e+02        1.740000e+02        1.740000e+02
mean        -2.119441e-14        1.858730e-14       -2.487984e-14
std          1.002886e+00        1.002886e+00        1.002886e+00
min         -1.562689e+00       -1.936618e+00       -2.220772e+00
25%         -8.900657e-01       -8.146263e-01       -7.410671e-01
50%         -3.855985e-01       -2.536303e-01       -2.126012e-01
75%          9.386279e-01        8.683615e-01        7.386374e-01
max          2.136738e+00        2.177352e+00        2.324035e+00

       CHSWP8 ICCH Curncy  CHSWP9 ICCH Curncy  CHSWP10 ICCH Curncy
count        1.740000e+02        1.740000e+02         1.740000e+02
mean         1.228950e-14       -1.430194e-14        -7.922382e-15
std          1.002886e+00        1.002886e+00         1.002886e+00
min         -2.642063e+00       -2.761895e+00        -2.899826e+00
25%         -7.468032e-01       -6.209922e-01        -7.002233e-01
50%         -1.545345e-01        1.346207e-01         3.297755e-02
75%          7.930955e-01        7.328142e-01         6.705435e-01
max          2.096087e+00        2.149588e+00         2.328215e+00
```

- **Paso 2:** Cálculo de la matriz de covarianza de la data normalizada,
- cov=np.cov(x_norm,rowvar=0)

	0	1	2	3	4	5	6	7	8
0	1.006	0.996	0.975	0.944	0.908	0.846	0.769	0.650	0.620
1	0.996	1.006	0.995	0.969	0.939	0.888	0.812	0.700	0.671
2	0.975	0.995	1.006	0.993	0.972	0.933	0.869	0.773	0.750
3	0.944	0.969	0.993	1.006	0.994	0.971	0.925	0.845	0.831
4	0.908	0.939	0.972	0.994	1.006	0.989	0.944	0.872	0.860
5	0.846	0.888	0.933	0.971	0.989	1.006	0.974	0.923	0.914
6	0.769	0.812	0.869	0.925	0.944	0.974	1.006	0.981	0.969
7	0.650	0.700	0.773	0.845	0.872	0.923	0.981	1.006	0.998
8	0.620	0.671	0.750	0.831	0.860	0.914	0.969	0.998	1.006

- **Paso 3:** Descomposición de la matriz de covarianza en sus eigenvalues y eigenvectors, además dentro de la matriz de eigenvalues las normalizaremos para determinar el porcentaje de varianza explicada y elegir un numero óptimo de factores para la reducción

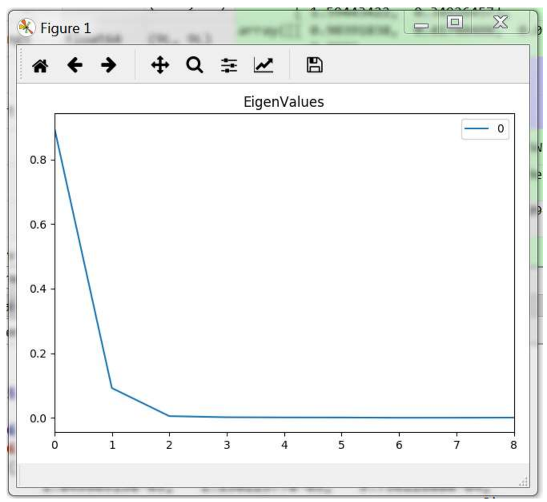
- (evals,evecs)=np.linalg.eig(cov)
- wf=evals/np.sum(evals)

	0	1	2	3	4	5	6	7	8
0	8.122	0.835	0.048	0.017	0.011	0.009	0.002	0.003	0.006

	0	1	2	3	4	5	6	7	8
0	0.897	0.092	0.005	0.002	0.001	0.001	0.000	0.000	0.001

Index	0	1	2	3	4	5	6	7	8
0	0.897	0.989	0.995	0.997	0.998	0.999	0.999	0.999	1

$$ratio\ de\ varianza\ explicada = \frac{\lambda_j}{\sum_{j=1}^{d} \lambda_j}$$



- De acá ya es posible observar que los dos primeros componentes principales explican el 99% de la varianza total. Por lo anterior podríamos reconstruir la curva completa, prácticamente sin errores la curva original sólo con 2 factores…

- **Paso 3:** Descomposición de la matriz de covarianza en sus eigenvalues y eigenvectors, además dentro de la matriz de eigenvalues las normalizaremos para determinar el porcentaje de varianza explicada:

- (evals,evecs)=np.linalg.eig(cov)
- wf=evals/np.sum(evals)

EVALS

	0	1	2	3	4	5	6	7	8
0	8.122	0.835	0.048	0.017	0.011	0.009	0.002	0.003	0.006

- Cada uno de los eigenvalues tiene asociado un eigenvector, el cual, si bien es «adimensional» describe el comportamiento de las variables originales frente a movimientos en los factores, aun desconocidos, «x»

	0	1	2	3	4	5	6	7	8
0	0.317	0.457	0.438	0.255	0.593	0.064	-0.240	-0.134	-0.046
1	0.328	0.391	0.183	0.018	-0.307	0.240	0.531	0.506	0.124
2	0.340	0.273	0.008	-0.331	-0.529	-0.012	-0.646	-0.056	-0.013
3	0.348	0.131	-0.101	-0.352	0.012	-0.301	0.467	-0.525	-0.378
4	0.349	0.050	-0.504	-0.060	0.285	-0.374	-0.022	0.142	0.614
5	0.347	-0.098	-0.572	0.268	0.069	0.618	-0.060	-0.074	-0.273
6	0.339	-0.278	0.084	0.624	-0.250	-0.503	-0.051	0.149	-0.266
7	0.318	-0.460	0.369	0.051	-0.157	0.265	0.102	-0.436	0.503
8	0.312	-0.496	0.199	-0.483	0.318	0.039	-0.090	0.457	-0.254

EVECS

- **Paso 4:** Generación de los «factores» no correlacionados a partir de los factores originales y los eigenvectors extraidos de la matriz de covarianzas:

- factors = np.dot(evecs.T,x_norm.T).T

Index	factor 1	factor 2	factor 3	factor 4	factor 5	factor 6	factor 7	factor 8	factor 9
factor 1	1.000	0.000	0.000	0.000	0.000	-0.000	-0.000	-0.000	0.000
factor 2	0.000	1.000	-0.000	0.000	-0.000	-0.000	-0.000	-0.000	0.000
factor 3	0.000	-0.000	1.000	0.000	0.000	-0.000	0.000	0.000	0.000
factor 4	0.000	0.000	0.000	1.000	0.000	0.000	-0.000	-0.000	-0.000
factor 5	0.000	-0.000	0.000	0.000	1.000	-0.000	-0.000	-0.000	0.000
factor 6	-0.000	-0.000	-0.000	0.000	-0.000	1.000	-0.000	-0.000	0.000
factor 7	-0.000	-0.000	0.000	-0.000	-0.000	-0.000	1.000	-0.000	0.000
factor 8	-0.000	-0.000	0.000	-0.000	-0.000	-0.000	-0.000	1.000	0.000
factor 9	0.000	0.000	0.000	-0.000	0.000	0.000	0.000	0.000	1.000

Index	factor 1	factor 2	factor 3	factor 4	factor 5	factor 6	factor 7	factor 8	factor 9
2017-01-16 00:00:00	2.45	0.281	0.184	0.023	0.234	0.0283	0.019	-0.075	0.081
2017-01-17 00:00:00	1.59	0.349	0.204	0.0842	0.314	-0.0257	-0.0215	-0.00135	0.0376
2017-01-18 00:00:00	1.69	0.174	0.285	0.129	0.346	-0.046	0.0241	0.0239	0.038
2017-01-19 00:00:00	2.15	-0.0288	0.324	0.118	0.32	-0.1	-0.0337	0.059	0.0951
2017-01-20 00:00:00	1.93	-0.00663	0.273	0.106	0.186	-0.196	-0.0226	0.0382	0.0348
2017-01-23 00:00:00	1.65	0.052	0.248	0.0731	0.211	-0.241	-0.0191	-0.0462	-0.00682
2017-01-24 00:00:00	2.33	0.0375	0.227	-0.00738	0.194	-0.0998	0.0275	-0.0473	-0.0198
2017-01-25 00:00:00	2.88	-0.0192	0.306	-0.0658	0.107	0.0583	0.0176	0.042	0.00473
2017-01-26 00:00:00	3.77	-0.237	0.284	-0.13	0.0252	0.0375	0.0175	0.0169	-0.0217
2017-01-27 00:00:00	4.07	-0.326	0.255	-0.173	-0.00995	0.0116	0.0694	-0.0145	-0.0496
2017-01-30 00:00:00	4.18	-0.412	0.264	-0.0979	-0.0327	-0.0496	0.0742	-0.0385	0.0396
2017-01-31 00:00:00	3.29	-0.0585	0.237	0.0712	-0.0638	-0.111	0.0533	0.0316	0.0111
2017-02-01 00:00:00	3.63	-0.0849	0.254	0.0485	-0.129	-0.0926	0.0389	0.0674	0.0199
2017-02-02 00:00:00	2.92	0.412	0.143	0.0845	-0.109	-0.109	-0.0334	0.0132	-0.0192
2017-02-03 00:00:00	2.51	0.578	0.154	0.0482	-0.00924	-0.0466	-0.000709	0.0288	-0.111
2017-02-06 00:00:00	2.78	0.482	0.0272	0.0233	0.114	0.0308	0.00966	-0.00886	-0.0575
2017-02-07 00:00:00	2.39	0.684	-0.012	0.00823	0.0138	-0.0805	-0.0211	-0.00632	-0.0594
2017-02-08 00:00:00	2.47	1.05	0.0035	-0.109	0.0204	-0.168	-0.00163	-0.056	-0.0288
2017-02-09 00:00:00	2.64	1.13	-0.0746	-0.126	0.0266	-0.144	0.00212	-0.0717	-0.0271
2017-02-10 00:00:00	2.63	1.13	-0.0626	-0.0994	0.0466	-0.126	0.089	-0.0325	-0.0175
2017-02-13 00:00:00	2.75	0.938	0.0278	-0.0501	0.129	0.0407	0.0943	-0.0511	0.0783
2017-02-14 00:00:00	3.64	0.736	0.0396	-0.0705	0.0796	0.0488	0.0414	-0.0396	0.081
2017-02-15 00:00:00	4.1	0.604	0.0926	-0.106	0.0763	0.175	0.00836	-0.0159	0.0894
2017-02-16 00:00:00	4.13	0.736	0.0842	-0.122	0.0674	0.161	-0.0317	0.0621	0.031

- La cual es fácil de verificar que son, no correlacionados y recordemos que los primeros 2 factores son capaces de explicar un 98,9% del total de la varianza

- Dado lo anterior, podemos trabajar con sólo las 2 primeras columnas y recuperar virtualmente sin errores la curva de tasas original

- **Paso 5:** Es importante lograr <u>interpretar</u> los factores «adimensionales» o «variables de estado» e internar entender que relación tienen con los factores observables de mercado, para lo cual utilizaremos una sencilla relación para determinar los **loadings,** que básicamente es la correlación entre los factores adimensionales ortogonales y las variables originales (propuesto, verifique dicha relación):

$$\sqrt{Eigenvalue_i} \times Eigenvector_i = Loadings$$

EVALS

	0	1	2	3	4	5	6	7	8
0	8.122	0.835	0.048	0.017	0.011	0.009	0.002	0.003	0.006

EVECS

	0	1	2	3	4	5	6	7	8
0	0.317	0.457	0.438	0.255	0.593	0.064	-0.240	-0.134	-0.046
1	0.328	0.391	0.183	0.018	-0.307	0.240	0.531	0.506	0.124
2	0.340	0.273	0.008	-0.331	-0.529	-0.012	-0.646	-0.056	-0.013
3	0.348	0.131	-0.101	-0.352	0.012	-0.301	0.467	-0.525	-0.378
4	0.349	0.050	-0.504	-0.060	0.285	-0.374	-0.022	0.142	0.614
5	0.347	-0.098	-0.572	0.268	0.069	0.618	-0.060	-0.074	-0.273
6	0.339	-0.278	0.084	0.624	-0.250	-0.503	-0.051	0.149	-0.266
7	0.318	-0.460	0.369	0.051	-0.157	0.265	0.102	-0.436	0.503
8	0.312	-0.496	0.199	-0.483	0.318	0.039	-0.090	0.457	-0.254

- **Paso 5:** Es importante lograr <u>interpretar</u> los factores «adimensionales» o «variables de estado» e internar entender que relación tienen con los factores observables de mercado, para lo cual utilizaremos una sencilla relación para determinar los **loadings,** que básicamente es la correlación entre los factores adimensionales ortogonales y las variables originales (propuesto, verifique dicha relación):

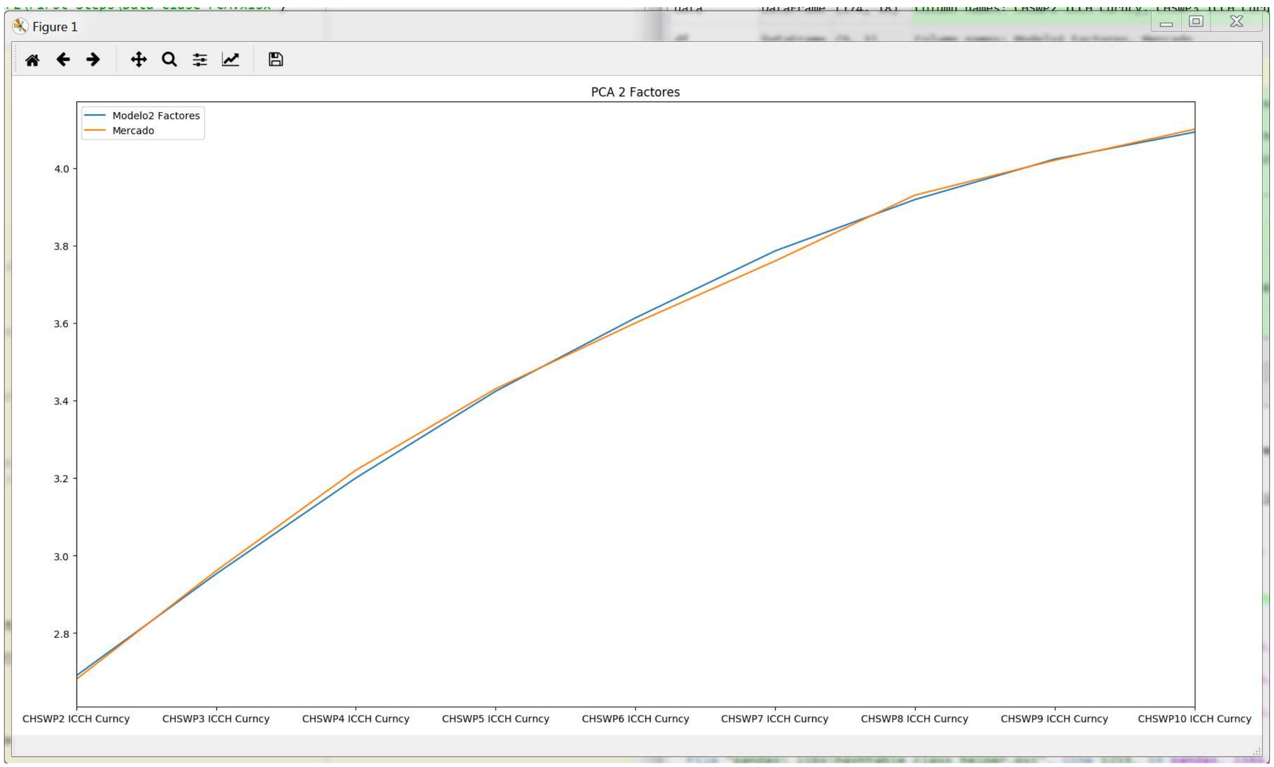
$$\sqrt{Eigenvalue_i} \times Eigenvector_i = Loadings$$

Index	factor 1	factor 2	factor 3	factor 4	factor 5	factor 6	factor 7	factor 8	factor 9
CHSWP2 ICCH Curncy	0.904	0.417	0.096	0.033	0.063	0.006	-0.011	-0.007	-0.003
CHSWP3 ICCH Curncy	0.935	0.358	0.040	0.002	-0.032	0.023	0.024	0.027	0.009
CHSWP4 ICCH Curncy	0.968	0.249	0.002	-0.043	-0.056	-0.001	-0.029	-0.003	-0.001
CHSWP5 ICCH Curncy	0.993	0.120	-0.022	-0.046	0.001	-0.028	0.021	-0.028	-0.028
CHSWP6 ICCH Curncy	0.994	0.045	-0.110	-0.008	0.030	-0.035	-0.001	0.008	0.046
CHSWP7 ICCH Curncy	0.988	-0.089	-0.126	0.035	0.007	0.058	-0.003	-0.004	-0.021
CHSWP8 ICCH Curncy	0.965	-0.254	0.019	0.081	-0.027	-0.047	-0.002	0.008	-0.020
CHSWP9 ICCH Curncy	0.905	-0.420	0.081	0.007	-0.017	0.025	0.005	-0.024	0.038
CHSWP10 ICCH Curncy	0.890	-0.453	0.044	-0.063	0.034	0.004	-0.004	0.025	-0.019

- **Paso 6:** Reconstrucción del sistema con los factores reducidos. En este caso demostramos que con 2 variables eramos capaces de recuperar cerca del 99% de la varianza bajo la matriz de covarianza bajo análisis, por lo cual, a continuación intentaremos reconstruir la curva original, por medio de la utilización de los 2 primeros componentes principales…
- Trabajaremos con n_fact=2, y debemos hacer el proceso inverso al anterior pues debemos volver desde el «mundo de los factores ortogonales» hacia el mundo real en nuestra representación reducida de la curva de tasas…
- **evecs_sub=evecs[:,0:n_fact]; → utiliza sólo los 2 primeros eigenvectors**
- **factors_sub=factors[:,0:n_fact] → consecuentemente utiliza sólo los 2 primeros factotes asociados a los eigenvectors anteriores**
- **proj=pd.DataFrame(np.dot(evecs_sub,factors_sub.T).T,index=x.index,columns=x.columns)*x_desv+x_mean**
- **drift=proj-x → diferencias entre las tasas reconstruidas con 2 factores y las tasas de mercado**
- Los resultados observados son sorprendentes y pueden resumirse por medio del comando describe() del DataFrame Drift

Index	CHSWP2 ICCH Curncy	CHSWP3 ICCH Curncy	CHSWP4 ICCH Curncy	CHSWP5 ICCH Curncy	CHSWP6 ICCH Curncy	CHSWP7 ICCH Curncy	CHSWP8 ICCH Curncy	CHSWP9 ICCH Curncy	CHSWP10 ICCH Curncy
min	-0.062	-0.022	-0.021	-0.021	-0.041	-0.039	-0.031	-0.018	-0.019
25%	-0.010	-0.007	-0.007	-0.006	-0.008	-0.009	-0.005	-0.005	-0.005
50%	-0.000	-0.001	-0.001	0.001	0.001	0.000	0.000	-0.000	-0.001
mean	-0.000	-0.000	0.000	0.000	-0.000	0.000	-0.000	0.000	-0.000
75%	0.012	0.006	0.006	0.006	0.011	0.010	0.006	0.005	0.005
std	0.021	0.010	0.010	0.009	0.014	0.014	0.009	0.008	0.007
max	0.054	0.031	0.031	0.023	0.025	0.026	0.021	0.026	0.018
count	174.000	174.000	174.000	174.000	174.000	174.000	174.000	174.000	174.000

- **Paso 6:** Reconstrucción del sistema con los factores reducidos. () del DataFrame Drift



- **Paso 7:** Extensiones,

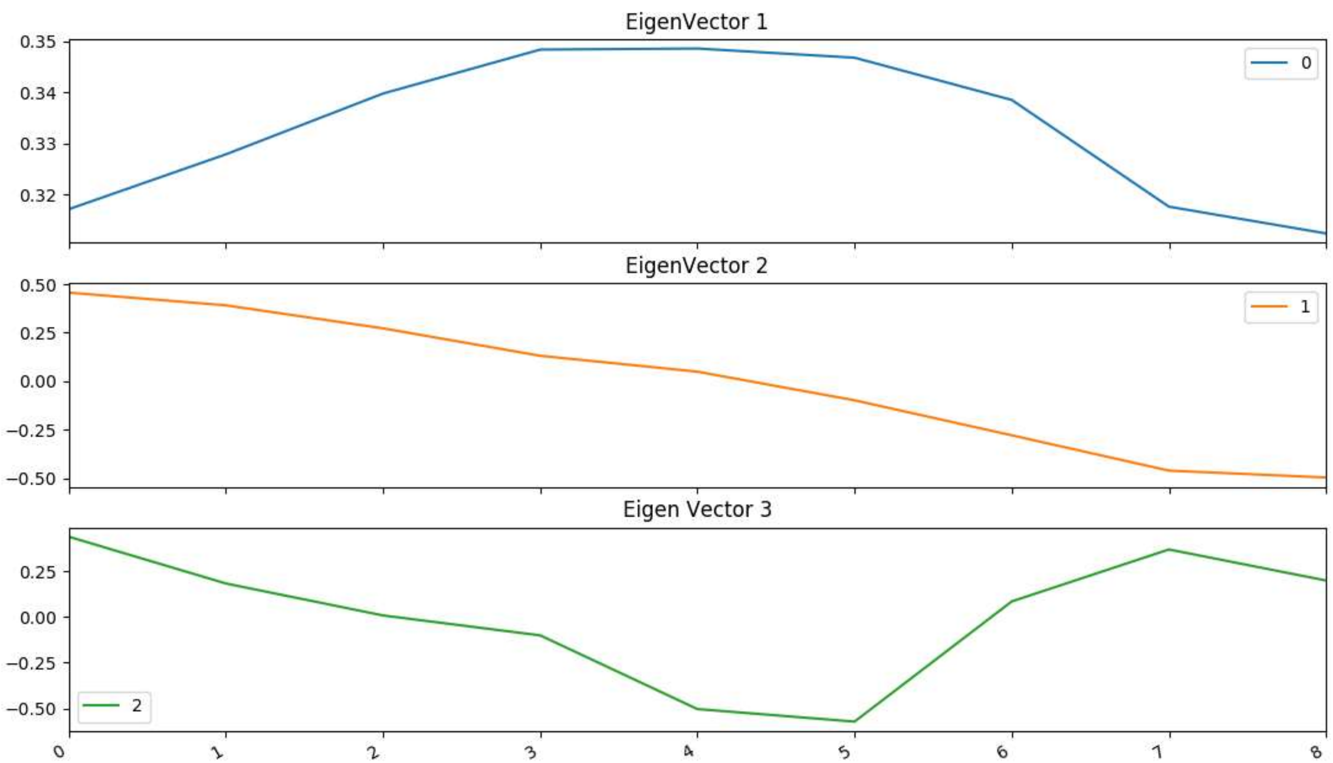
7.1 Interpretación de los eigenvectors: Notar que el primer eigenvector sólo tiene entradas positivas frente al factor 1, por lo cual puede interpretarse como el eigenvector direccional, además de la matriz de loadings puede apreciarse que los tenors de 5y y 6y son los que presentan mayor direccionalidad

Por su parte el segundo eigenvector tiene un cambio de signo frente a movimientos del factor 2 por lo que captura movimientos de curvatura

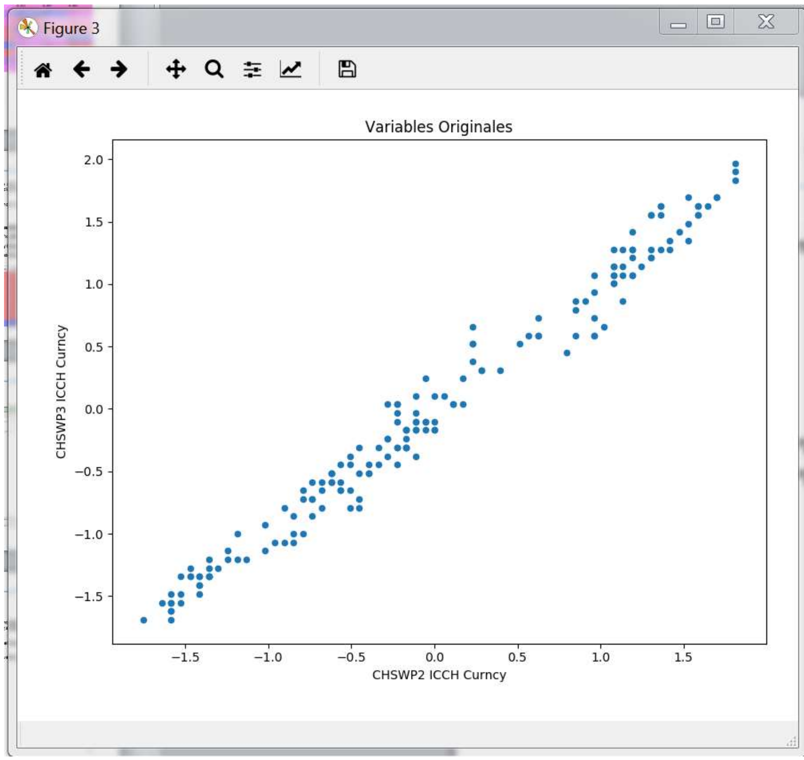
Finalmente en curvas más complejas a veces es necesario llegar hasta el tercer eigenvector para capturas cambios en la convexidad de la curva de tasas

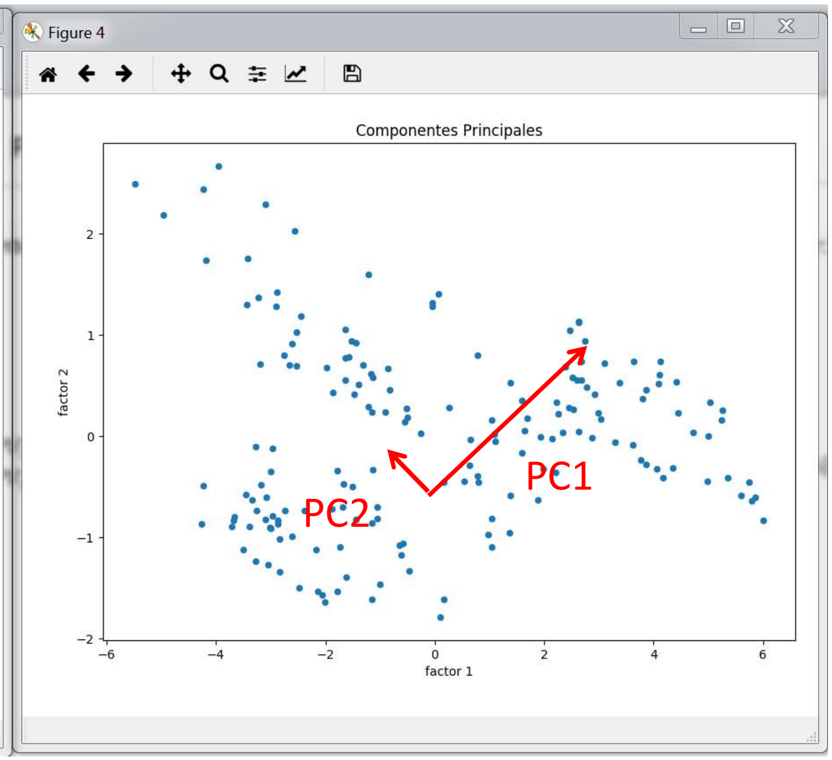
	0	1	2
0	0.317	0.457	0.438
1	0.328	0.391	0.183
2	0.340	0.273	0.008
3	0.348	0.131	-0.101
4	0.349	0.050	-0.504
5	0.347	-0.098	-0.572
6	0.339	-0.278	0.084
7	0.318	-0.460	0.369
8	0.312	-0.496	0.199

- **Paso 7:** Extensiones,



- **Datos antes y después de Ortogonalización…**





- **Paso 7:** Extensiones, lo anterior puede ser muy útil para hacer coberturas y determinar hedges ratios, a saber:

- $\frac{n_5}{n_{10}} = \frac{BPV_{10}}{BPV_5} x \frac{e_{1,10}}{e_{1,5}}$ → 5x10 swap spread PCA neutral (bonus)

- $\begin{pmatrix} n_2 \\ n_{10} \end{pmatrix} = \begin{pmatrix} BPV_2 e_{1,2} & BPV_{10} e_{1,10} \\ BPV_2 e_{2,2} & BPV_{10} e_{2,10} \end{pmatrix}^{-1} \begin{pmatrix} -n_5 BPV_5 e_{1,5} \\ -n_5 BPV_5 e_{2,5} \end{pmatrix}$ → 2x5x10 butterly swap spread PCA neutral (bonus)

- La primera expresión permite determinar los nocionales que permiten estructurar una posición en un spread de 2y x 5y PCA neutral

- La segunda expresión permite determinar los nocionales en 2y, 5y y 10y de manera tal que son neutrales a la direccionalidad y slope de la curva, ergo es un trade full convexidad, la cual esta representada en el tercer eigenvector del sistema

- Por razones pedagógicas, hasta el momento hemos realizado todo el análisis «manualmente» utilizando principalmente la librería numpy de Python, sin embargo the pythonic way nos ofrece realizar todo este trabajo de manera mucho más eficiente por medio de la librería scikit-learn…

```python
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
```

→ Importación de las librerías a utilizar

```python
# Escalamiento de la data

scaler = StandardScaler()
scaler.fit(x)
StandardScaler(copy=True, with_mean=True, with_std=True)
media=scaler.mean_
sdv=scaler.std_
sd=pd.DataFrame(scaler.transform(x),index=x.index,columns=x.columns)
```

→ Normalización de la data como parte del preprocessing, por otra parte dejamos en la memoria la media y desviación estándar de las variables normalizadas

	CHSWP2 ICCH Curncy	CHSWP3 ICCH Curncy	CHSWP4 ICCH Curncy
count	1.740000e+02	1.740000e+02	1.740000e+02
mean	1.371827e-15	-4.213424e-15	-2.699309e-15
std	1.002886e+00	1.002886e+00	1.002886e+00
min	-1.754602e+00	-1.688415e+00	-1.640841e+00
25%	-7.350449e-01	-7.224736e-01	-8.438803e-01
50%	-1.686241e-01	-1.705069e-01	-2.413002e-01
75%	1.006699e+00	9.161775e-01	9.249839e-01
max	1.813849e+00	1.968364e+00	1.935764e+00

	CHSWP5 ICCH Curncy	CHSWP6 ICCH Curncy	CHSWP7 ICCH Curncy
count	1.740000e+02	1.740000e+02	1.740000e+02
mean	-1.032380e-14	6.288712e-15	-1.072578e-15
std	1.002886e+00	1.002886e+00	1.002886e+00
min	-1.562689e+00	-1.936618e+00	-2.220772e+00
25%	-8.900657e-01	-8.146263e-01	-7.410671e-01
50%	-3.855985e-01	-2.536303e-01	-2.126012e-01
75%	9.386279e-01	8.683615e-01	7.386374e-01
max	2.136738e+00	2.177352e+00	2.324035e+00

	CHSWP8 ICCH Curncy	CHSWP9 ICCH Curncy	CHSWP10 ICCH Curncy
count	1.740000e+02	1.740000e+02	1.740000e+02
mean	-3.006695e-15	8.062676e-15	-8.003974e-15
std	1.002886e+00	1.002886e+00	1.002886e+00
min	-2.642063e+00	-2.761895e+00	-2.899826e+00
25%	-7.468032e-01	-6.209922e-01	-7.002233e-01
50%	-1.545345e-01	1.346207e-01	3.297755e-02
75%	7.930955e-01	7.328142e-01	6.705435e-01
max	2.096087e+00	2.149588e+00	2.328215e+00

- Posteriormente la librería sklearn hace el trabajo pesado por nosotros:

```
# Fit de modelo PCA con 2 factores

pca=PCA(n_components=2)
pca.fit(sd)
evals=pca.explained_variance_                           # corresponde a los eigenvalues
evecs=pca.components_.T                                 # corresponde a los eigenvectores
var_expl=pca.explained_variance_ratio_                  # Varianza explicada por cada componente principal
factores=pca.fit_transform(sd)                          # Factores ortogonales
variables_estand=pca.inverse_transform(factores)        # Recuperación de las variables estandarizadas usando 2 factores
variables_originales=variables_estand*sdv.T+media.T     # Recuperación de las variables originales usando 2 factores
```

evals - Arreglo de NumPy

	0
0	8.075
1	0.830

evecs - Arreglo de NumPy

	0	1
0	0.317	0.457
1	0.328	0.391
2	0.340	0.273
3	0.348	0.131
4	0.349	0.050
5	0.347	-0.098
6	0.339	-0.278
7	0.318	-0.460
8	0.312	-0.496

var_expl - Arreglo de NumPy

	0
0	0.897
1	0.092

Análisis de Componentes Principales (PCA) y Reducción de Dimensionalidad: Scikit Learn

- **Posteriormente la librería sklearn hace el trabajo pesado por nosotros:**

```
# Fit de modelo PCA con 2 factores

pca=PCA(n_components=2)
pca.fit(sd)
evals=pca.explained_variance_                      # corresponde a los eigenvalues
evecs=pca.components_.T                            # corresponde a los eigenvectores
var_expl=pca.explained_variance_ratio_             # Varianza explicada por cada componente principal
factores=pca.fit_transform(sd)                     # Factores ortogonales
variables_estand=pca.inverse_transform(factores)   # Recuperación de las variables estandarizadas usando 2 factores
variables_originales=variables_estand*sdv.T+media.T  # Recuperación de las variables originales usando 2 factores
```

factores - Arreglo de NumPy

	0	1
0	2.449	0.281
1	1.594	0.349
2	1.688	0.174
3	2.152	-0.029
4	1.927	-0.007
5	1.648	0.052
6	2.334	0.038
7	2.875	-0.019
8	3.769	-0.237
9	4.066	-0.326
10	4.179	-0.412
11	3.294	-0.059
12	3.629	-0.085

variables_estand - Arreglo de NumPy

	0	1	2	3	4	5	6	7	8
0	0.905	0.913	0.909	0.890	0.868	0.822	0.751	0.649	0.626
1	0.665	0.660	0.637	0.601	0.573	0.519	0.443	0.346	0.325
2	0.615	0.622	0.621	0.611	0.597	0.568	0.523	0.456	0.441
3	0.670	0.695	0.724	0.746	0.749	0.749	0.737	0.697	0.687

variables_originales - Arreglo de NumPy

	0	1	2	3	4	5	6	7	8
0	2.940	3.157	3.378	3.582	3.750	3.898	4.006	4.091	4.157
1	2.897	3.120	3.343	3.547	3.718	3.869	3.980	4.067	4.133
2	2.888	3.115	3.341	3.549	3.721	3.874	3.987	4.076	4.142
3	2.898	3.125	3.354	3.565	3.737	3.891	4.005	4.095	4.161

- Existen muchas otras formas de seleccionar variables reduciendo la dimensionalidad de un problema basado en criterios de varianza, por ejemplo, el **Variance Threshold**, el cual remueve todas aquellas carácterísticas cuya varianza no reune undeterminado threshold:

cov - DataFrame

Index	CHSWP2 ICCH Curncy	CHSWP3 ICCH Curncy	CHSWP4 ICCH Curncy	CHSWP5 ICCH Curncy	CHSWP6 ICCH Curncy	CHSWP7 ICCH Curncy	CHSWP8 ICCH Curncy	CHSWP9 ICCH Curncy	CHSWP10 ICCH Curncy
CHSWP10 ICCH Curncy	0.009	0.008	0.008	0.008	0.007	0.007	0.006	0.006	0.006
CHSWP9 ICCH Curncy	0.009	0.008	0.008	0.008	0.007	0.007	0.007	0.006	0.006
CHSWP8 ICCH Curncy	0.011	0.010	0.009	0.009	0.009	0.008	0.007	0.007	0.006
CHSWP7 ICCH Curncy	0.014	0.012	0.011	0.011	0.010	0.009	0.008	0.007	0.007
CHSWP6 ICCH Curncy	0.017	0.015	0.013	0.013	0.012	0.010	0.009	0.007	0.007
CHSWP5 ICCH Curncy	0.020	0.017	0.015	0.014	0.013	0.011	0.009	0.008	0.008
CHSWP4 ICCH Curncy	0.022	0.019	0.017	0.015	0.013	0.011	0.009	0.008	0.008
CHSWP3 ICCH Curncy	0.025	0.021	0.019	0.017	0.015	0.012	0.010	0.008	0.008
CHSWP2 ICCH Curncy	0.031	0.025	0.022	0.020	0.017	0.014	0.011	0.009	0.009

```
import pandas as pd
from sklearn.feature_selection import VarianceThreshold
cov=x.cov()
sel = VarianceThreshold(threshold=0.01)
filtered_data=sel.fit_transform(x)
```

filtered_data - Arreglo de NumPy

	0	1	2	3	4
0	2.980	3.150	3.360	3.580	3.750
1	2.950	3.110	3.320	3.540	3.720
2	2.950	3.110	3.310	3.540	3.720
3	2.960	3.120	3.330	3.550	3.740
4	2.930	3.110	3.330	3.550	3.730
5	2.920	3.090	3.320	3.550	3.720
6	2.950	3.130	3.350	3.580	3.740
7	2.970	3.170	3.380	3.590	3.750
8	2.990	3.200	3.420	3.630	3.780
9	2.990	3.210	3.430	3.650	3.790

- Existen muchas otras alternativas basadas en regresiones, covarianzas o bien otros criterios de información para seleccionar atributos, a saber:

- **La «F-Regression»:** En el caso de la regresión F, se regresiona cada variable individualmente y se determinan los valores F y p-value de cada regresión para posteriormente rankear y elegir las de mayor test F o bien menores p-values

- **Mutual Information:** Intuitivamente, la información mutua media mide la información que *X* e *Y* comparten, o de otro modo, mide en cuánto el conocimiento de una variable reduce nuestra incertidumbre sobre la otra. Por ejemplo, si *X* e *Y* son independientes, entonces conocer *X* no da información sobre *Y* y viceversa, por lo que su información mutua es cero. En el otro extremo, si *X* e *Y* son idénticas entonces toda información proporcionada por *X* es compartida por *Y*, es decir, saber *X* determina el valor de *Y* y viceversa. Por ello, la información mutua media es igual a la información contenida en *Y* (o *X*) por sí sola, también llamada la entropía de *Y* (o *X*: claramente si *X* e *Y* son idénticas tienen idéntica entropía).

- La información mutua media cuantifica la dependencia entre la distribución conjunta de *X* e *Y* y la que tendrían si *X* e *Y* fuesen independientes. La información mutua media es una medida de dependencia en el siguiente sentido: $I(X; Y) = 0$ sí y solo sí *X* e *Y* son variables aleatorias independientes. Esto es fácil de ver en una dirección: si *X* e *Y* son independientes, entonces $p(x,y) = p(x)\ p(y)$, y por tanto:

- Ejemplo en Python:

```python
import numpy as np
from sklearn.feature_selection import f_regression, mutual_info_regression

y=x.iloc[:,0]
x=x.iloc[:,1:]

f_test, _ = f_regression(x, y)
f_test /= np.max(f_test)

mi = mutual_info_regression(x, y)
mi /= np.max(mi)
```

f_test - Arreglo de Nu

	0
0	1.000
1	0.322
2	0.153
3	0.091
4	0.050
5	0.029
6	0.015
7	0.013

mi - Arreglo de NumPy

	0
0	1.000
1	0.791
2	0.717
3	0.596
4	0.454
5	0.444
6	0.379
7	0.388