

Time Series Filtering

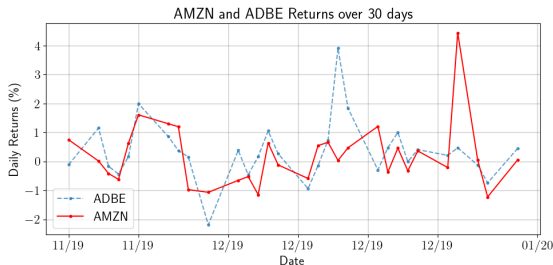


Zella Baig

Supervisors :

Mohsin Javed (BlackRock)

Yuji Nakatsukasa (University of Oxford)



- ▶ Problem: We wish to extract some measure of “similarity” between stocks, but they are noisy. Can we separate **noise** and **signal**?
 - ▶ Useful in e.g. pairs trading.
- ▶ A possible solution: **Singular Spectrum Analysis (SSA)**.

Consider¹ a time series $Z_T = (z_1, \dots, z_T)$. With fixed *window length* L and with $K := T - L + 1$:

1. Construct the (Hankel) trajectory matrix:

$$\mathbf{X} := \begin{bmatrix} z_1 & z_2 & z_3 & \dots & z_K \\ z_2 & z_3 & z_4 & \dots & z_{K+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ z_L & z_{L+1} & z_{L+2} & \dots & z_T \end{bmatrix} \quad (1)$$

¹Hassani, Mahmoudvand, et al. 2011.

2. Compute SVD (via HMT algorithm) of \mathbf{X} :

$$\mathbf{X} = U\Sigma V^T = \sum_{i=1}^n u_i v_i^T \sigma_i$$

3. Truncate the SVD to r rank-1 matrices, with *rank* r chosen s.t. $r \leq n$:

$$\mathbf{X} \approx \mathcal{X} = \sum_{i=1}^r u_i v_i^T \sigma_i$$

4. \mathcal{X} is not necessarily Hankel, so re-diagonalise on the off-diagonals to reconstruct a de-noised series $\bar{Z}_T = (\bar{z}_1, \dots, \bar{z}_T)$ from the averaged Hankel matrix

$$\bar{\mathbf{X}} := \begin{bmatrix} \bar{z}_1 & \bar{z}_2 & \bar{z}_3 & \dots & \bar{z}_K \\ \bar{z}_2 & \bar{z}_3 & \bar{z}_4 & \dots & \bar{z}_{K+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{z}_L & \bar{z}_{L+1} & \bar{z}_{L+2} & \dots & \bar{z}_T \end{bmatrix} \quad (2)$$

1. To set L , we first define the *weighted inner product*

$$(Z_T, Y_T)_w := \sum_{i=1}^T \min \{i, L, T - i + 1\} z_i y_i \quad (3)$$

with associated norm

$$\|Z_T\|_w^2 := (Z_T, Z_T)_w.$$

2. We then construct the w -correlation

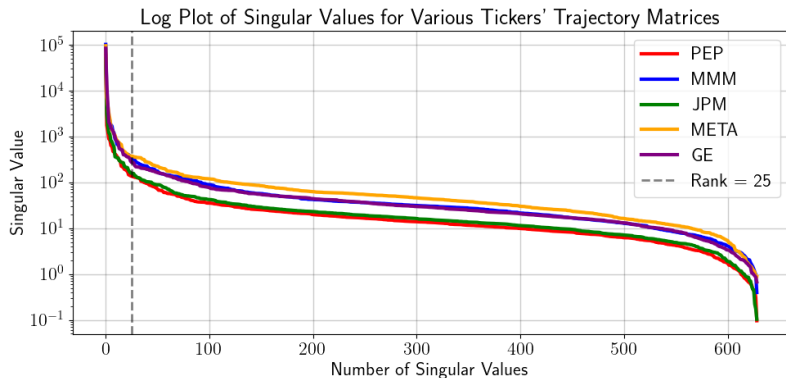
$$\rho_w(Z_T, Y_T) := \frac{(Z_T, Y_T)_w}{\|Z_T\|_w \|Y_T\|_w}$$

which we use as a measure of separability.

It can be shown² that a choice of $L = \frac{T+1}{2}$ minimises ρ_w .

²Hassani, Kalantari, et al. 2017.

To choose r , examine the scree plot showing a knee at approximately 25 singular values.



We measure similarity of two time (de-noised) time series using the **Time Warped Edit Distance**³ (TWED). First define

$$f(x_i, y_j) = |x_i - y_j|$$

for two time series $X_T = (x_1, \dots, x_T)$ and $Y_T = (y_1, \dots, y_T)$ as a “difference measurement”.

³Marteau 2008.

Next, we initialise a grid $D_{i,j}$ s.t.

$$D_{0,0} = 0,$$

$$D_{0,j} = \infty \quad j = 1, \dots, T,$$

$$D_{i,0} = \infty \quad i = 1, \dots, T.$$

Y_T	1	∞			
	1	∞			
	9	∞			
	0	∞	∞	∞	
		5	-3	1	
		X_T			

Figure 1: Initialised TWED grid.

Then define the *TWED-Closeness* by $D_{T,T}$, where we construct

$$D_{i,j} = \min \{D_{i-1,j-1} + \Gamma_{X,Y}, D_{i-1,j} + \Gamma_X, D_{i,j-1} + \Gamma_Y\} \quad (4)$$

for $1 \leq i, j \leq T$, where

$$\Gamma_{X,Y} = f(x_i, y_j) + f(x_{i-1}, y_{j-1}) + 2\nu|i - j|, \quad (5)$$

$$\Gamma_X = f(x_i, x_{i-1}) + \nu + \lambda, \quad (6)$$

$$\Gamma_Y = f(y_j, y_{j-1}) + \nu + \lambda, \quad (7)$$

with

- ▶ λ : deletion penalty
- ▶ ν : stiffness parameter

Y_T	1	∞	18	17	16
	1	∞	13	12	13
	9	∞	4	13	14
		0	∞	∞	∞
			5	-3	1
		X_T			

Figure 2: Populated TWED grid, with $\nu = \lambda = 0.5$. $D_{T,T} = 16$.

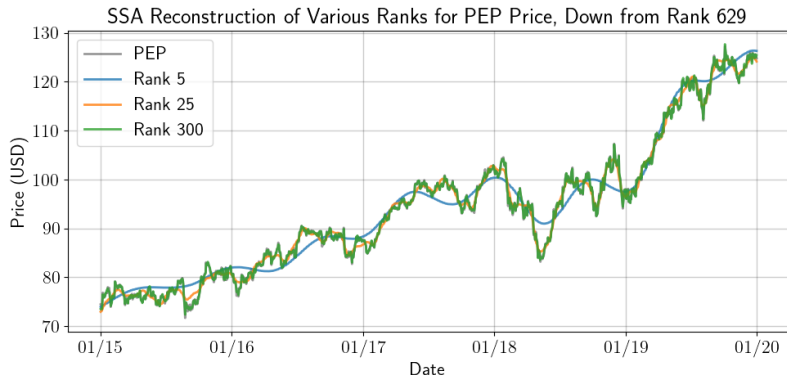


Figure 3: Different rank SSA reconstructions. Note underfitting at $r = 5$, and overfitting at $r = 300$.

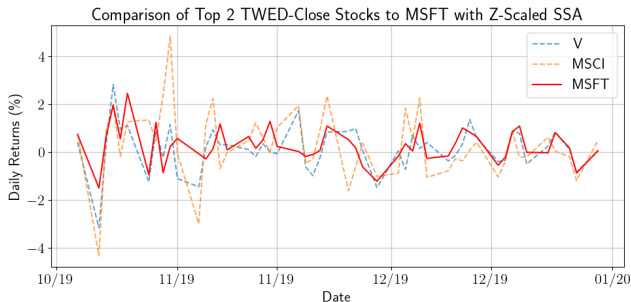
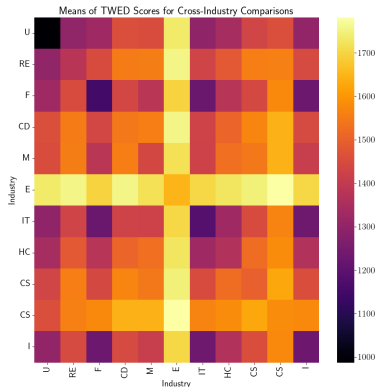


Figure 4: Returns over 50 days for the top 2 most “similar” stocks to MSFT. Note how when returns diverge, they eventually reconverge.

With n time series of length m , TWED-scoring complexity is $O(m^2n^2) \sim$ **12 hours** for 500 stocks over 5 years.

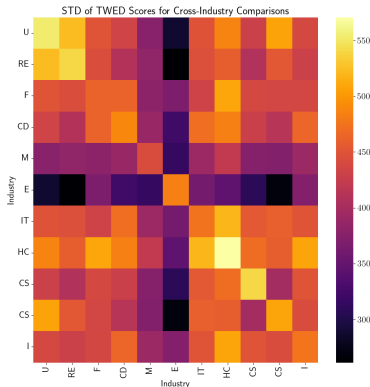
Key takeaways:

- Energy, Consumer Staples sector dissimilar to other sectors.
- Utilities, Finance, IT show strong inter-similarity.

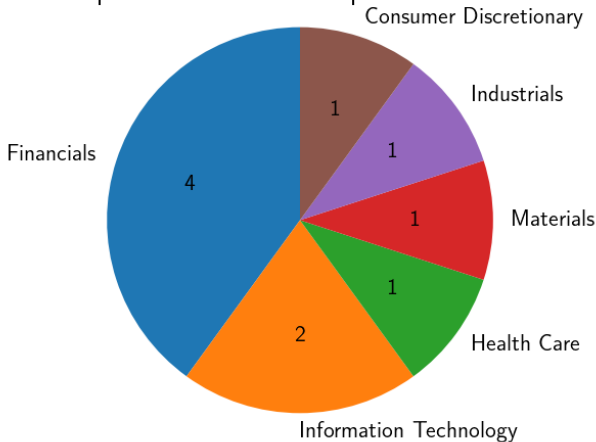


Key takeaways:

- Energy sector conclusions strong.
- Utilities, Health Care conclusion very weak.



Top 10 TWED-Close Composition for JPM



- ▶ Neural-network based approaches
 - ▶ There is promising work being done on *Siamese Neural Networks*⁴, which use a pair of Recurrent Neural Networks that share weights to classify time series.
- ▶ Fine-tuning the choice of SSA parameters to better classify data
- ▶ Forecasting?

⁴Hou et al. 2019.

- (1) Dey, S.; Dutta, A.; Toledo, J. I.; Ghosh, S. K.; Lladós, J.; Pal, U. SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification, Number: arXiv:1707.02131, 2017.
- (2) Ghodsi, M.; Hassani, H.; Rahmani, D.; Silva, E. S. *Journal of Applied Statistics* **2018**, *45*, Publisher: Taylor & Francis _eprint: <https://doi.org/10.1080/02664763.2017.1401050>, 1872–1899.
- (3) Hassani, H.; Kalantari, M.; Yarmohammadi, M. *Comptes Rendus Mathématique* **2017**, *355*, 1026–1036.
- (4) Hassani, H.; Mahmoudvand, R.; Zokaei, M. *Comptes Rendus Mathématique* **2011**, *349*, 987–990.

- (5) Hou, L.; Jin, X.; Zhao, Z. In *2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 2019, pp 1–6.
- (6) Marteau, P.-F. Time Warp Edit Distance, Number: arXiv:0802.3522, 2008.
- (7) Serrà, J.; Arcos, J. L. *Knowledge-Based Systems* **2014**, 67, 305–314.

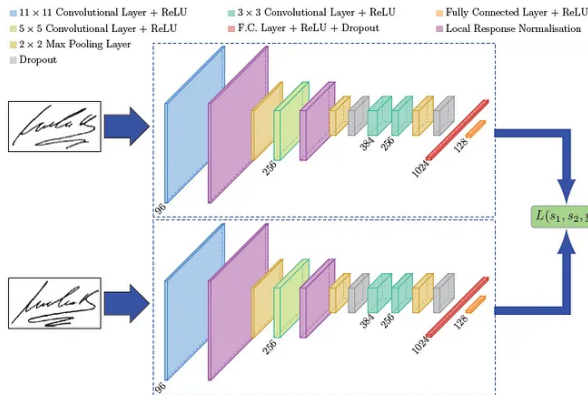


Figure 5: Overview of an SNN, as used in SigNet⁵.

⁵Dey et al. 2017.

There exist two different types of SSA forecasting: recurrent, and vector. We go over them in turn:

1. **Recurrent forecasting**⁶: Consider the left singular vectors u_1, u_2, \dots, u_r . Take their L^{th} components, denoted π_i , and define

$$v^2 := \sum_{i=1}^r \pi_i^2. \quad (8)$$

Denote by \hat{u}_i the $L - 1 \times 1$ vector which is u_i with the final component removed.

⁶Ghodsi et al. 2018.

Then define

$$A = (\alpha_{L-1}, \dots, \alpha_1)^T = \frac{1}{1-v^2} \sum_{i=1}^r \pi_i \hat{u}_i,$$

and thus construct

$$z_t = \begin{cases} \bar{z}_t & t = 1, \dots, T, \\ \sum_{i=1}^{L-1} \alpha_i z_{t-i} & t = T+1, \dots, T+h, \end{cases}$$

for a forecast to h steps ahead.

2. **Vector forecasting**⁷: First define

$$\hat{\mathbf{U}} = [\hat{u}_1, \dots, \hat{u}_r],$$

and construct

$$\mathbf{\Pi} = \hat{\mathbf{U}}\hat{\mathbf{U}}^T + (1 - v^2) \mathbf{A}\mathbf{A}^T.$$

Finally, construct the operator Θ s.t.

$$\Theta V := \begin{bmatrix} \mathbf{\Pi}\hat{V} \\ \mathbf{A}^T\hat{V} \end{bmatrix},$$

where \hat{V} denotes the vector V with the last element removed.

⁷Ghodsia et al. 2018.

Define now

$$\Xi_i = \begin{cases} \mathcal{X}_i & i = 1, \dots, K, \\ \Theta \Xi_{i-1} & i = K + 1, \dots, K + h + L - 1, \end{cases}$$

where \mathcal{X}_i are the columns of \mathcal{X} . Next construct

$$\Xi = [\Xi_1, \dots, \Xi_{K+h+L-1}],$$

and hankelise to get the matrix $\bar{\Xi}$ from which we recover an “extended” time series containing forecasted values.