

What Drives Cryptocurrency Returns? Learning Market Regimes Through the Sparse Statistical Jump Model

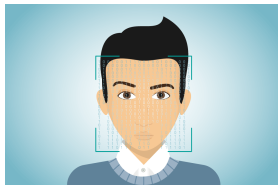
Erik Lindström

Centre for Mathematical Sciences
Lund University, Sweden

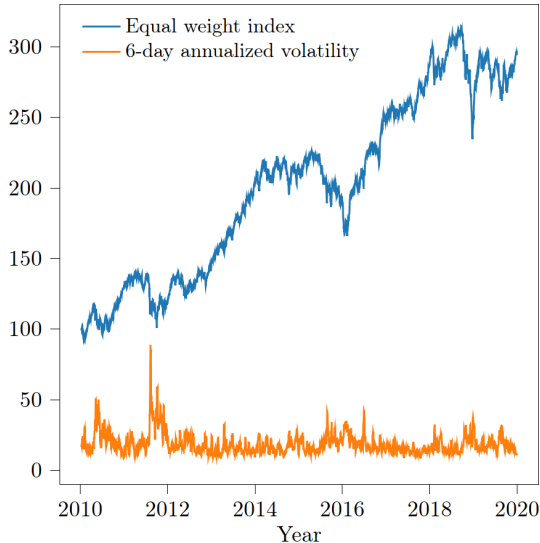
NordStat 2023, Gothenburg, June 19th-22nd, 2023

Joint work with Federico Cortese (Univ. Milano-Bicocca) and
Petter N. Kolm (Courant Institute, NYU)

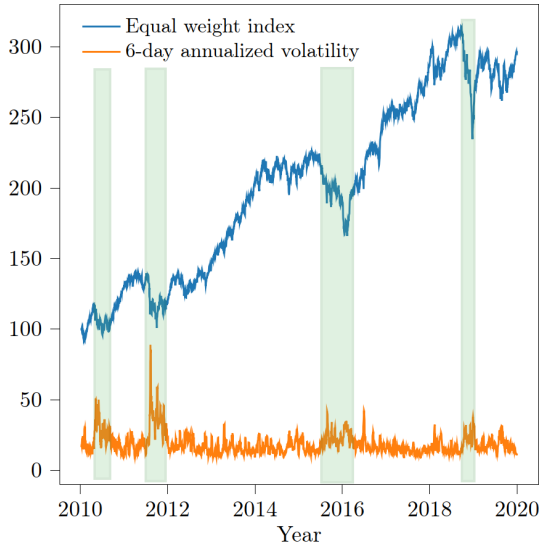
Applications for regime switching models



What are the market states? How can we benefit from knowing them?



What are the market states? Ad hoc classification – exit during turbulent periods, participate otherwise



- **Sensitive** to model misspecification
 - Markov assumption
 - Conditional distributions
 - *These problems multiply in higher dimensions!*
- Practical considerations
 - Not **robust** to initialization or outliers
 - Hidden states lack **persistence**
 - Needs a lot of **data** to produce useful estimates
 - **Computationally** expensive due to slow convergence
- Nontrivial to incorporate **exogenous** features

- Extensions of the classical HMM
 - other sojourn-time distributions, HSMMs (Bulla and Bulla, 2006)
 - other conditional distributions (Bulla, 2011; Nystrup et al, 2015)
 - time-varying parameters (Nystrup et al., 2017)
 - shrinkage estimation (Fiecas et al., 2017)
 - Bayesian priors (Fox et al., 2011)
 - Multiperiod Asset allocation (Nystrup et al., 2017a, 2018, 2019)
- Dimensionality reduction (Filion et al., 2010)
 - no guarantee that reduced feature space retains distinguishing features
- Feature saliency HMM (Adams et al., 2016)
 - Very slow, only feasible to low dimensional models
- System identification class of (Statistical) Jump models (Bemporad et al., 2018; Nystrup et al., 2020a, b)
- Feature selection in clustering (Witten and Tibshirani, 2010), generalized to Sparse Statistical Jump Model (Nystrup et al., 2021)

- Introduction to (Sparse) Statistical Jump Models
- Case study on cryptocurrencies

Jump model



Fitting jump models

Bemporad et al. (2018) proposed to fit jump models by minimizing

$$\sum_{t=1}^{T-1} [\ell_{mode}(\mathbf{y}_t, \theta_{s_t}) + \ell_{trans}(\lambda, s_t, s_{t-1})] + \ell(\mathbf{y}_T, \theta_{s_T}), \quad (1)$$

over the model parameters θ_{s_t} and state sequence $s_1, \dots, s_T \in [1, \dots, K]$

Later, (Nystrup et al., 2020b) considered *standardized features* $\tilde{\mathbf{y}} \in \mathbb{R}^p$ generated from data

$$\sum_{t=1}^{T-1} [\|\tilde{\mathbf{y}}_{t,p} - \boldsymbol{\mu}_{s_t}\|_2^2 + \lambda \mathbb{I}_{s_t \neq s_{t-1}}] + \|\tilde{\mathbf{y}}_{T,p} - \boldsymbol{\mu}_{s_T}\|_2^2, \quad (2)$$

Note:

- Jump penalty $\lambda \geq 0$ is a regularization parameter
- This class *can* nest many other models, including HMMs

Bempored et al, (2018) proved that state and parameter estimation that the joint loss function can be optimized by alternating two separate steps

- ① Initialize state sequence $\mathbf{s}$ as $s_1^0, \dots, s_T^0$
- ② **Iterate** for $j = 1, \dots$
 - a. Fit model parameters $\boldsymbol{\mu}^j = \operatorname{argmin}_{\boldsymbol{\mu}} \sum_{t=1}^T \|\mathbf{y}_t - \boldsymbol{\mu}_{s_t^{j-1}}\|^2$
 - b. Fit state sequence using dynamic programming
$$\mathbf{s}^j = \operatorname{argmin}_{\mathbf{s}} \left\{ \sum_{t=1}^{T-1} [\|\mathbf{y}_t - \boldsymbol{\mu}_{s_t^j}^j\|^2 + \lambda \mathbb{I}_{s_t \neq s_{t+1}}] + \|\mathbf{y}_T - \boldsymbol{\mu}_{s_T^j}^j\|^2 \right\}$$
- ③ **Until** $\mathbf{s}^j = \mathbf{s}^{j-1}$

Convergence often within a few iterations, even with poor SNR.

Main results in (Nystrup et al., 2020b)

- Not quite as efficient as MLE on LARGE samples (efficiency loss typically within a few %)
- But far better small sample properties (!)
 - Robust again poor initialization
 - Robust distributional misspecification
 - Robust misspecified sojourn-time distribution

Modified version for streaming data in (Nystrup et al., 2020a)

Following (Witten & Thishirani, 2010), Nystrup et al, (2021) propose to fit **sparse** jump models by solving

$$\begin{aligned} & \text{maximize} && \mathbf{w}' \sum_{k=1}^K n_k (\boldsymbol{\mu}_k - \bar{\boldsymbol{\mu}})^2 - \lambda \sum_{t=1}^{T-1} \mathbb{I}_{s_t \neq s_{t+1}} \\ & \text{subject to} && \|\mathbf{w}\|_2^2 \leq 1 \\ & && \|\mathbf{w}\|_1 \leq \kappa, \\ & && w_p \geq 0 \quad \forall p \end{aligned} \tag{3}$$

with respect to the model parameters $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K$, state sequence $s_1, \dots, s_T$, and feature weights $w_1, \dots, w_P$

Notation:

- Tuning parameter $1 \leq \kappa \leq \sqrt{P}$ controls the degree of sparsity

Sparse jump model fitting

- Initialize feature weights $\mathbf{w}$ as $w_1^0, \dots, w_P^0 = \frac{1}{\sqrt{P}}$
- **Iterate** for $i = 1, \dots$
 - ① Compute sequence of weighted features $\mathbf{z}_t = \mathbf{y}_t \cdot \text{diag}(\sqrt{\mathbf{w}^{i-1}})$
 - ② Initialize state sequence $\mathbf{s}$ as $s_1^0, \dots, s_T^0$
 - ③ **Iterate** for $j = 1, \dots$
 - a. Fit model parameters $\boldsymbol{\mu}^j = \underset{\boldsymbol{\mu}}{\text{argmin}} \sum_{t=1}^T \|\mathbf{z}_t - \boldsymbol{\mu}_{s_t^{j-1}}\|^2$
 - b. Fit state sequence
$$\mathbf{s}^j = \underset{\mathbf{s}}{\text{argmin}} \left\{ \sum_{t=1}^{T-1} [\|\mathbf{z}_t - \boldsymbol{\mu}_{s_t^j}^j\|^2 + \lambda \mathbb{I}_{s_t \neq s_{t+1}}] + \|\mathbf{z}_T - \boldsymbol{\mu}_{s_T^j}^j\|^2 \right\}$$
 - ④ **Until** $\mathbf{s}^j = \mathbf{s}^{j-1}$
 - ⑤ Update weights $\mathbf{w}^i$ using soft thresholding while holding the parameters $\boldsymbol{\mu}^j$ and state sequence $\mathbf{s}^j$ fixed
- **Until** $\|\mathbf{w}^i - \mathbf{w}^{i-1}\|_1 / \|\mathbf{w}^{i-1}\|_1 < 10^{-4}$

Python code is available as supplementary to the paper at
<https://doi.org/10.1016/j.eswa.2021.115558>

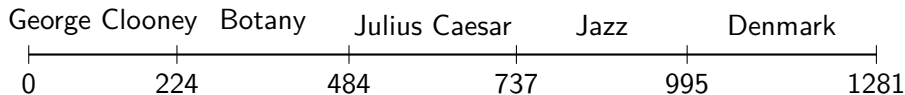
We compare the sparse jump model to models with and without feature selection on simulated data where the true state sequence is known

Main findings include:

- Accuracy of jump model can be improved through feature selection
- Sparse jump model has highest accuracy in nearly all settings
- Sparse jump model is more robust to (correlated) noise

NLP application

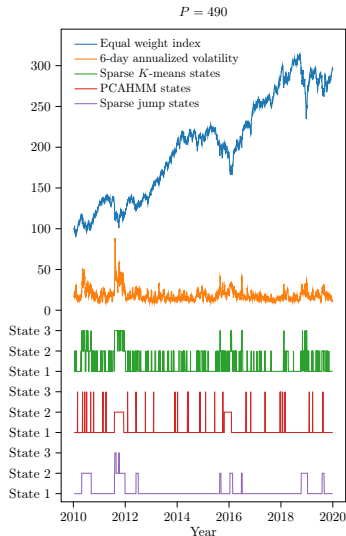
Based on Hallac et al. (2019), used the Word2Vec embedding to create a 1281×300 dimensional data set from the lead paragraphs of the corresponding wikipedia articles.



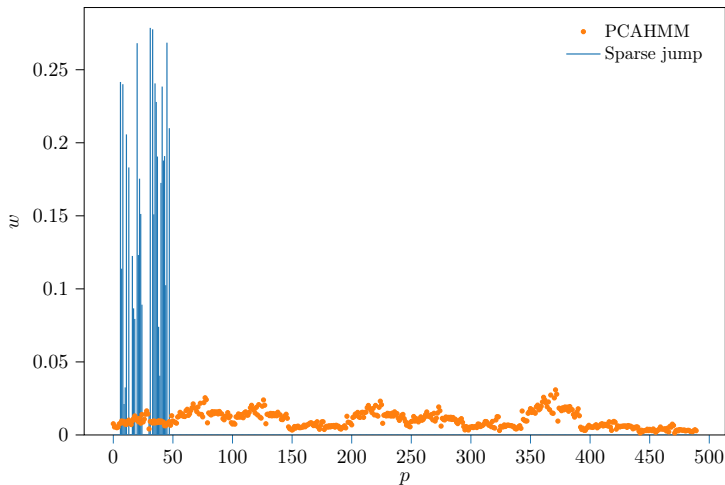
	$P = 300$	$P = 600$	$P = 1200$
GGs	3.5 (0.0)	23 (5.2)	28 (2.5)
Jump	35 (81)	83 (143)	144 (185)
Sparse jump	0.48 (7.3)	0.39 (7.0)	0.0 (0.0)

- When $P = 300$, GGS takes 2 min 18 sec and sparse jump 6.6 sec

Application to industry portfolio returns



Feature selection on the 49 industry portfolios



Case study: cryptocurrencies



Pennoni et al., 2022 models the five largest cryptocurrencies with a Gaussian HMM. They find that the optimal model is a model with 6 or 7 regimes, where the persistence for each state is only a few days.

Case study: cryptocurrencies

Pennoni et al., 2022 models the five largest cryptocurrencies with a Gaussian HMM. They find that the optimal model is a model with 6 or 7 regimes, where the persistence for each state is only a few days.

We consider the five largest and most liquid cryptocurrencies: Bitcoin (BTC), Ethereum (ETH), Ripple (XRP), Litecoin (LTC), and Bitcoin Cash (BCH from Jan 2019 to Sept 2022). The features, $P = 409$, can be sorted into three groups

- **Financial Market Features**, based on time series from the equity, fixed income, foreign exchange and commodity markets, describing cross-asset dynamics between cryptocurrencies and traditional markets.
- **Sentiment Features**, e.g. proxies for public attention toward cryptocurrencies
- **Crypto Market-Related Features** derived directly from cryptocurrency prices, trade volumes, and blockchain- related metrics.

SJM applied to crypto currencies

We report in (Cortese et al., 2023), that a model with only **three** states is the preferred model, indicating that features can eliminate spurious states. Moreover, only 19 features are selected.



FIGURE 2. Cumulative log-returns of BTC, ETH, XRP, LTC, and BCH over the period January 2018 - September 2022, together with the state sequence obtained from the SJM in green (bull), yellow (neutral) and red (bear).

Table 1: State-conditional means and standard deviations (SD) of the five cryptocurrency log-returns obtained from the SJM model.

	State 1		State 2		State 3	
	Mean (%)	SD (%)	Mean (%)	SD (%)	Mean (%)	SD (%)
BTC	1.02	3.64	-0.26	3.15	-1.39	5.72
ETH	1.36	4.55	-0.29	4.23	-2.05	7.38
XRP	0.87	6.27	-0.26	4.80	-1.69	7.42
LTC	1.22	5.22	-0.47	4.17	-1.87	7.51
BCH	1.31	6.15	-0.46	4.59	-2.57	8.37

Furthermore, correlation in state 1 is modest, state 2 larger and state 3 high.

Features

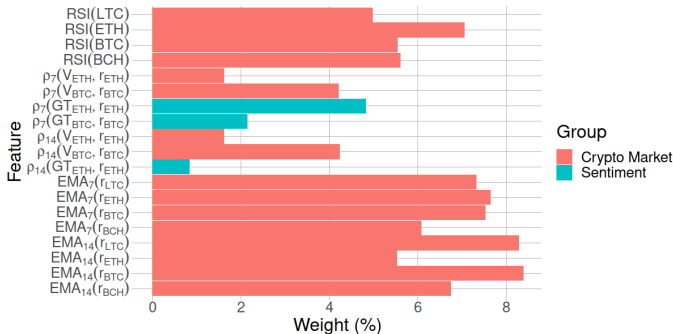


Figure 1: The selected features are RSIs for BTC, ETH, LTC and BCH; 7- and 14-day exponentially weighted linear correlations of log-differences of volumes and log-returns for BTC and ETH; 7-day exponentially weighted linear correlations of log-differences of Google Trend data, GT, and log-returns for BTC and ETH; 14-day exponentially weighted linear correlations of log-differences of GT and log-returns for BTC; 7- and 14-day EMAs of log-returns of BTC, ETH, LTC and BCH.

Features

Feature	Weight(%)	Bull	Neutral	Bear
RSI(LTC)	5.0	63.03	43.96	31.81
RSI(ETH)	7.0	66.68	45.03	31.61
RSI(BTC)	5.5	66.46	45.71	31.58
RSI(BCH)	5.6	62.55	43.55	28.60
$\rho_7(V_{ETH}, r_{ETH})$	1.6	0.21	-0.09	-0.32
$\rho_7(V_{BTC}, r_{BTC})$	4.2	0.24	-0.07	-0.36
$\rho_7(GT_{ETH}, r_{ETH})$	4.8	0.31	-0.08	-0.35
$\rho_7(GT_{BTC}, r_{BTC})$	2.1	0.26	-0.10	-0.38
$\rho_{14}(V_{ETH}, r_{ETH})$	1.6	0.22	-0.06	-0.29
$\rho_{14}(V_{BTC}, r_{BTC})$	4.2	0.18	-0.04	-0.29
$\rho_{14}(GT_{ETH}, r_{ETH})$	0.8	0.14	-0.08	-0.27
$EMA_7(r_{ETH})$	7.6	1.08%	-0.25%	1.50%
$EMA_7(r_{LTC})$	7.3	0.92%	-0.34%	-1.56%
$EMA_7(r_{BTC})$	7.5	0.80%	-0.15%	-1.18%
$EMA_7(r_{BCH})$	6.1	0.98%	-0.39%	-2.04%
$EMA_{14}(r_{ETH})$	5.5	0.78%	-0.17%	-1.00%
$EMA_{14}(r_{LTC})$	8.3	0.65%	-0.27%	-1.12%
$EMA_{14}(r_{BTC})$	8.4	0.60%	-0.10%	-0.86%
$EMA_{14}(r_{BCH})$	6.7	0.65%	-0.33%	-1.46%

- We reviewed a recently developed alternative to HMMs with robust statistical properties and attractive convergence properties
- Large scale feature selection extends the set of problems that can be analyzed.
- The framework was applied to five major cryptocurrencies, finding persistent and economically sound market regimes.
- The selected features are "*momentum*" and *sentiment features*, rather than macro or market related features.

Adams, S., P. A. Beling, and R. Cogill.

“Feature selection for hidden Markov models and hidden semi-Markov models.”
IEEE Access, vol. 4 (2016), pp. 1642–1657.

Bemporad, A., V. Breschi, D. Piga, and S. Boyd.

“Fitting jump models.”
Automatica, vol. 96 (2018), pp. 11–21.

Bulla, J.

“Hidden Markov models with t components. Increased persistence and other aspects.”
Quantitative Finance, vol. 11, no. 3 (2011), pp. 459–475.

Bulla, J. and I. Bulla.

“Stylized facts of financial time series and hidden semi-Markov models.”
Computational Statistics & Data Analysis, vol. 51, no. 4 (2006), pp. 2192–2209.

References II



Cortese, F., P. N. Kolm, and E. Lindstrom.

“What drives cryptocurrency returns? a sparse statistical jump model approach.”

Digital Finance (2023).

URL <http://dx.doi.org/10.1007/s42521-023-00085-x>.

Fiecas, M., J. Franke, R. von Sachs, and J. T. Kamgaing.

“Shrinkage estimation for multivariate hidden Markov models.”

Journal of the American Statistical Association, vol. 112, no. 517 (2017), pp. 424–435.

Filion, G. J., J. G. van Bommel, U. Braunschweig, W. Talhout, J. Kind, L. D. Ward, W. Brugman, I. J. de Castro, R. M. Kerkhoven, H. J. Bussemaker, and B. van Steensel.

“Systematic protein location mapping reveals five principal chromatin types in *Drosophila* cells.”

Cell, vol. 143, no. 2 (2010), pp. 212–224.

Fox, E. B., E. B. Sudderth, M. I. Jordan, and A. S. Willsky.

“A sticky HDP-HMM with application to speaker diarization.”

Annals of Applied Statistics, vol. 5, no. 2A (2011), pp. 1020–1056.

References III



Hallac, D., P. Nystrup, and S. Boyd.

“Greedy Gaussian segmentation of multivariate time series.”

Advances in Data Analysis and Classification, vol. 13, no. 3 (2019), pp. 727–751.

Nystrup, P., S. Boyd, E. Lindström, and H. Madsen.

“Multi-period portfolio selection with drawdown control.”

Annals of Operations Research, vol. 282, no. 1-2 (2019), pp. 245–271.

Nystrup, P., B.W. Hansen, H.O. Larsen, H. Madsen, and E. Lindström.

“Dynamic allocation or diversification: A regime-based approach to multiple assets.”

The Journal of Portfolio Management, vol. 44, no. 2 (2017a), pp. 62–73.

Nystrup, P., P. N. Kolm, and E. Lindström.

“Greedy online classification of persistent market states using realized intraday volatility features.”

Journal of Financial Data Science, vol. 2, no. 3 (2020a), pp. 25–39.

Nystrup, P., P. N. Kolm, and E. Lindström.

“Feature selection in jump models.”

Expert Systems with Applications, vol. 184 (2021), p. 115558.

Nystrup, P., E. Lindström, and H. Madsen.

“Learning hidden Markov models with persistent states by penalizing jumps.”

Expert Systems with Applications, vol. 150 (2020b), p. 113307.

Nystrup, P., H. Madsen, and E. Lindström.

“Stylised facts of financial time series and hidden markov models in continuous time.”

Quantitative Finance, vol. 15, no. 9 (2015), pp. 1531–1541.

Nystrup, P., H. Madsen, and E. Lindström.

“Long memory of financial time series and hidden Markov models with time-varying parameters.”

Journal of Forecasting, vol. 36, no. 8 (2017b), pp. 989–1002.

Nystrup, P., H. Madsen, and E. Lindström.

“Dynamic portfolio optimization across hidden market regimes.”

Quantitative Finance, vol. 18, no. 1 (2018), pp. 83–95.

Pennoni, F., F. Bartolucci, G. Forte, and F. Ametrano.

“Exploring the dependencies among main cryptocurrency log-returns: A hidden markov model.”

Economic Notes, vol. 51, no. 1 (2022), p. e12193.

Witten, D. M. and R. Tibshirani.

“A framework for feature selection in clustering.”

Journal of the American Statistical Association, vol. 105, no. 490 (2010), pp. 713–726.