NRU HSE
MSSA 2021

Times Series Final Project
Bitcoin forecasting

Authors:
Artemij Kiel
Roman Pavlyutin
Lev Tsipes

Moscow 2022

Introduction

The goal of the following study is to create a model capable of forecasting the next month's Bitcoin rates. This model would serve as a guideline for IB and hedge-fund analysts to decide what behavior they expect from the crypto currency: up- or downward movement.

We are using data collected throughout the period from January 1st 2018 to November 2nd 2021. It contains daily rates of 13 different crypto-currencies.

The hypothesis of this study is that crypto-currencies are cointegrated and react to each others' previous values, and thus, we will be able to use those past observations to predict future Bitcoin rates.

The original data frame did not include descriptions of what exact crypto-currencies were in there, but during the data preparation stage, we managed to find out some of those names by comparing our data to Yahoo Finance graphs. We were able to conclude that Bitcoin, the crypto-currency of our interest, was labeled as cr1 in the data file, while Etherium was number 2 and BTC-Cash was number 5. On top of that, in the original file only Bitcoin had data starting from 2018, while most other crypto-currencies had been tracked only from September 1st 2020. Since our hypothesis is that those crypto-currencies are cointegrated, we decided to drop those 2 years worth of Bitcoin observations, because we wouldn't be able to use them. Moreover, we have noticed that some unnamed crypto-currencies had even less observations, so we made a decision to drop those rather than narrow the time periods even further.

The next step of data preparation was normalization. The reason is that different crypto-currencies have different scales, so in order to increase its accuracy and integrity, we needed to normalize the data.

Finally, before attempting to build any models, we had to remove any possible trends. After doing so, we have created a new data frame in Excel by melting the original file. The new data frame includes crypto-currencies' labels as column variables, and their normalized and detrended rates at specific dates as rows.

OLS

Initially, we started with the very basic OLS model to get 'a taste' of our data.

Source	SS	df	MS			
				Number of obs	=	428
				F(11, 416)	=	105.91
Model	3.90515821	11	.355014383	Prob > F	=	0.0000
Residual	1.39450952	416	.003352186	R-squared	=	0.7369
				Adj R-squared	=	0.7299
Total	5.29966773	427	.0124114	Root MSE	=	.0579

cr_1_norm	Coefficient	Std. err.	t	P>|t|	[95% conf.	interval]
cr_2_norm	.4867079	.0308764	15.76	0.000	.4260146	.5474012
cr_5_norm	-.026924	.0328997	-0.82	0.414	-.0915944	.0377464
cr_6_norm	.0743252	.0299072	2.49	0.013	.0155371	.1331132
cr_7_norm	.2148907	.0372824	5.76	0.000	.1416053	.2881761
cr_9_norm	.0213782	.0254873	0.84	0.402	-.0287218	.0714781
cr_10_norm	.0006663	.036569	0.02	0.985	-.0712168	.0725495
cr_11_norm	.0912049	.0351325	2.60	0.010	.0221455	.1602643
year_int	5.249827	1.747794	3.00	0.003	1.814217	8.685437
month_int	.4395954	.1458961	3.01	0.003	.15281	.7263809
day_int	.0155212	.0047867	3.24	0.001	.0061122	.0249303
A	-.014296	.0047939	-2.98	0.003	-.0237193	-.0048727
_cons	-10608.64	3531.864	-3.00	0.003	-17551.16	-3666.115

We see that there is no connection between BTC and most other currencies. We also see that we did right detrending the currencies. The Breusch Godfrey test shows, that we are not able to handle the autocorrelation in this model.

Breusch-Godfrey LM test for autocorrelation

lags(*p*)	chi2	df	Prob > chi2
1	126.077	1	0.0000
2	129.872	2	0.0000
3	132.131	3	0.0000
4	133.165	4	0.0000
5	140.969	5	0.0000
6	142.613	6	0.0000
7	143.065	7	0.0000
8	143.123	8	0.0000
9	144.839	9	0.0000
10	145.029	10	0.0000
11	145.084	11	0.0000
12	145.102	12	0.0000
13	145.515	13	0.0000
14	145.838	14	0.0000

15	145.880	15	0.0000
16	146.918	16	0.0000
17	152.656	17	0.0000
18	156.088	18	0.0000
19	156.217	19	0.0000

H0: no serial correlation

ARIMAX model

To continue, we started with a more complex model, ARIMAX. Firstly, we checked BTC for unit-roots with dfuller test and did not get enough evidence to reject null hypothesis of unit-root presence.

```
. dfuller cr_1_detrend

Dickey-Fuller test for unit root          Number of obs  = 427
Variable: cr_1_detrend                    Number of lags =   0

H0: Random walk without drift, d = 0

                                       Dickey-Fuller
                   Test         ———————— critical value ————————
                statistic           1%           5%          10%
————————————————————————————————————————————————————————————————
 Z(t)              -1.296       -3.446       -2.873       -2.570
————————————————————————————————————————————————————————————————
MacKinnon approximate p-value for Z(t) = 0.6309.
```

We started with a simple model with all variables included:

```
arima d.cr_1_detrend d.cr_2_detrend d.cr_5_detrend d.cr_6_detrend d.cr_7_detrend d.cr_8_detrend d.cr_9_detrend d.cr_10_detrend d.cr_11_detrend, ar(1)
```

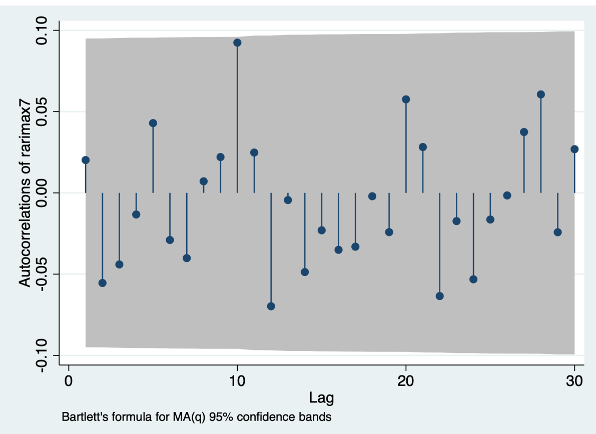
```
Sample: 1 thru 427                              Number of obs     =        427
                                                Wald chi2(9)      =    2315.41
Log likelihood = 1180.031                       Prob > chi2       =     0.0000

------------------------------------------------------------------------------
D.           |                 OPG
cr_1_detrend | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
cr_1_detrend |
 cr_2_detrend|
         D1. |   .3411286   .0497337     6.86   0.000     .2436524    .4386048
             |
 cr_5_detrend|
         D1. |   .1102562   .0369847     2.98   0.003     .0377676    .1827448
             |
 cr_6_detrend|
         D1. |   .1028654   .0415371     2.48   0.013     .0214542    .1842765
             |
 cr_7_detrend|
         D1. |   .1643756   .0417399     3.94   0.000     .0825669    .2461842
             |
 cr_8_detrend|
         D1. |   .1140178   .0291744     3.91   0.000      .056837    .1711986
             |
 cr_9_detrend|
         D1. |   .0112275   .0438582     0.26   0.798     -.074733     .097188
             |
cr_10_detrend|
         D1. |   .0169627   .0283872     0.60   0.550    -.0386752    .0726007
             |
cr_11_detrend|
         D1. |   .0070254   .0150931     0.47   0.642    -.0225566    .0366073
             |
       _cons |   .0003062   .0010994     0.28   0.781    -.0018486     .002461
-------------+----------------------------------------------------------------
ARMA         |
          ar |
         L1. |   .2696114   .0349487     7.71   0.000     .2011132    .3381097
-------------+----------------------------------------------------------------
      /sigma |   .0152589   .0003631    42.02   0.000     .0145472    .0159706
------------------------------------------------------------------------------
Note: The test of the variance against zero is one sided, and the two-sided
      confidence interval is truncated at zero.
```

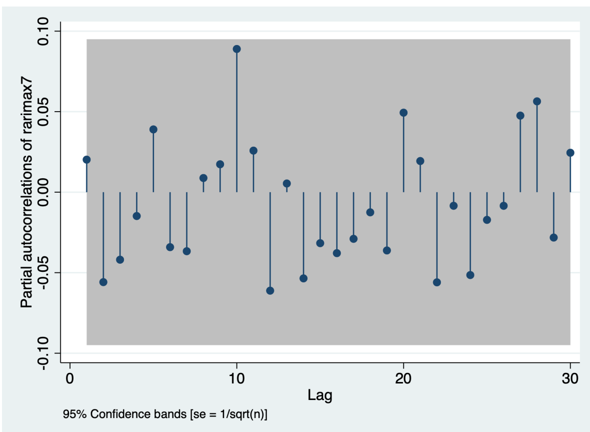
After multiple iterations we found that the only significant exogenous currency for BTC ARIMA model would be the fifth one. We also realized, that if we want to predict, we need to use lagged versions of exogenous variables.The final ARIMAX model equation with AR and MA tuning based on AC and PAC plots would look like:

```
arima d.cr_1_detrend d.l.cr_5_detrend, ar(1, 4) ma(8, 26) [i.e
ARIMA({1,4},{1},{8,26})]
```

```
ARIMA regression

Sample: 2 thru 427                              Number of obs     =        426
                                                Wald chi2(5)      =     104.98
Log likelihood = 992.8461                       Prob > chi2       =     0.0000

------------------------------------------------------------------------------
D.           |                 OPG
cr_1_detrend | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
cr_1_detrend |
cr_5_detrend |
         LD. |   -.230922   .0353856    -6.53   0.000    -.3002766   -.1615674
             |
       _cons |   .0002699   .0022258     0.12   0.903    -.0040927    .0046324
-------------+----------------------------------------------------------------
ARMA         |
          ar |
         L1. |   .3272777   .0478276     6.84   0.000     .2335373    .4210181
         L4. |   .1326453   .0397568     3.34   0.001     .0547234    .2105672
             |
          ma |
         L8. |  -.1115206   .0435242    -2.56   0.010    -.1968264   -.0262147
        L26. |   .1226859    .036032     3.40   0.001     .0520645    .1933073
-------------+----------------------------------------------------------------
      /sigma |    .023509   .0005929    39.65   0.000     .0223469    .0246711
------------------------------------------------------------------------------
```

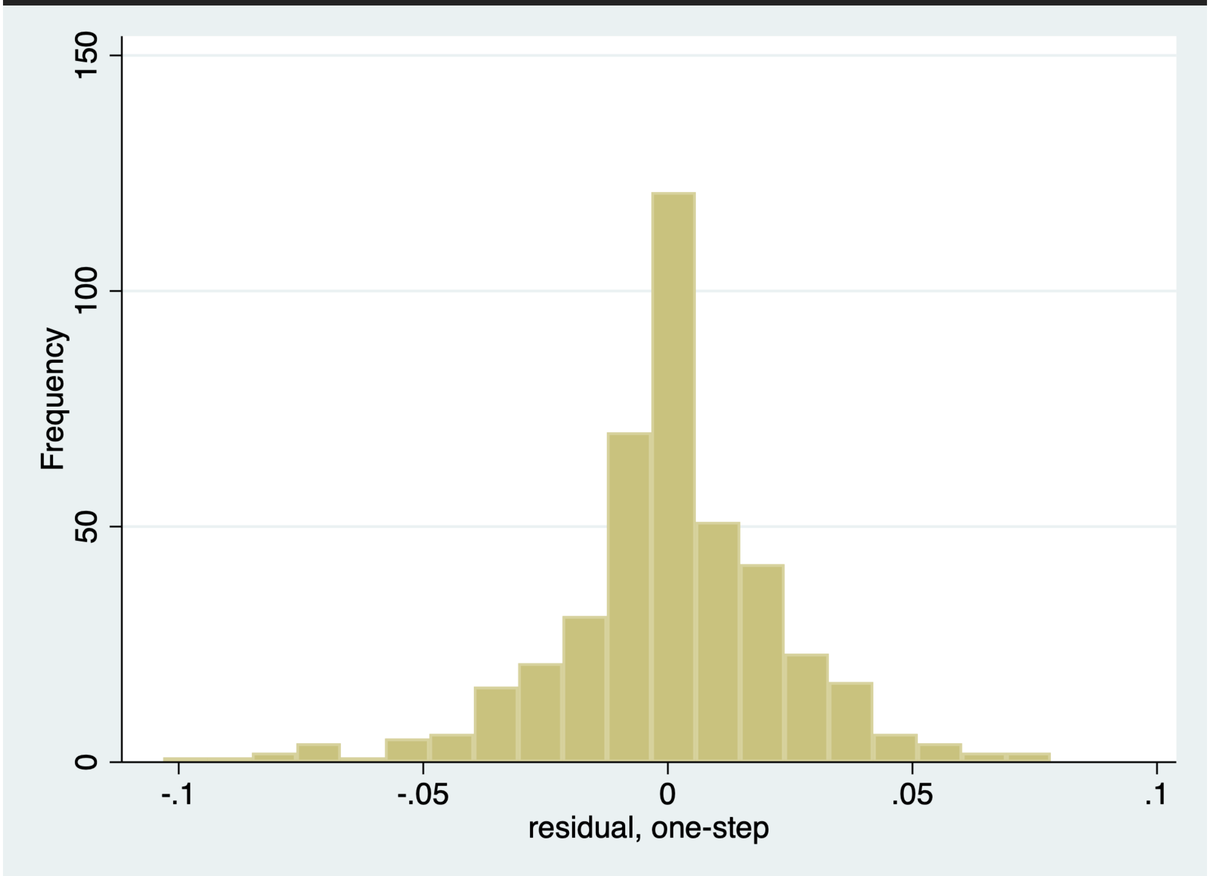
Since our goal is to predict a month forward and BTC is a dynamically changing currency, so we cared only about the closest thirty periods (days in our case).



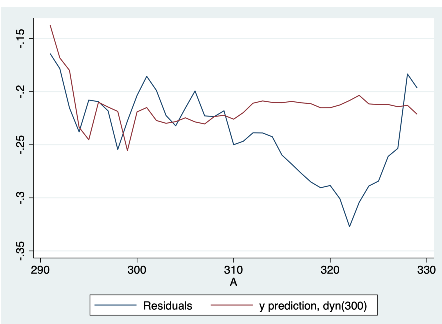


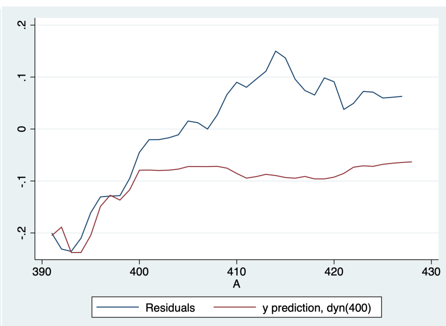
```
. wntestq rarimax7

Portmanteau test for white noise
---------------------------------------------
 Portmanteau (Q) statistic =     36.2168
 Prob > chi2(40)           =      0.6413
```

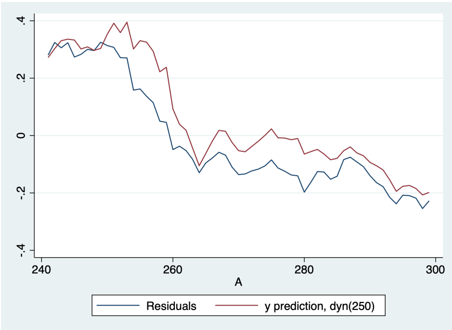
```
. sktest rarimax7

Skewness and kurtosis tests for normality
                                                       ----- Joint test -----
    Variable |       Obs   Pr(skewness)   Pr(kurtosis)   Adj chi2(2)  Prob>chi2
-------------+------------------------------------------------------------------
    rarimax7 |       426        0.0001         0.0000          33.90     0.0000
```

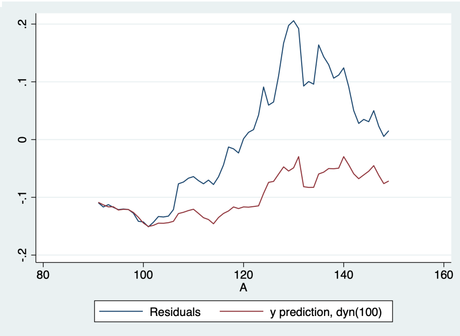


We have achieved a white noise data generating process. However, the residuals are technically not normally distributed. However, since visually the distribution is close to normal, we decided to step down and test it forecasting power abilities. For this purpose, we use the 100th, 250th, 300th and the 400th periods for dynamic forecasting.









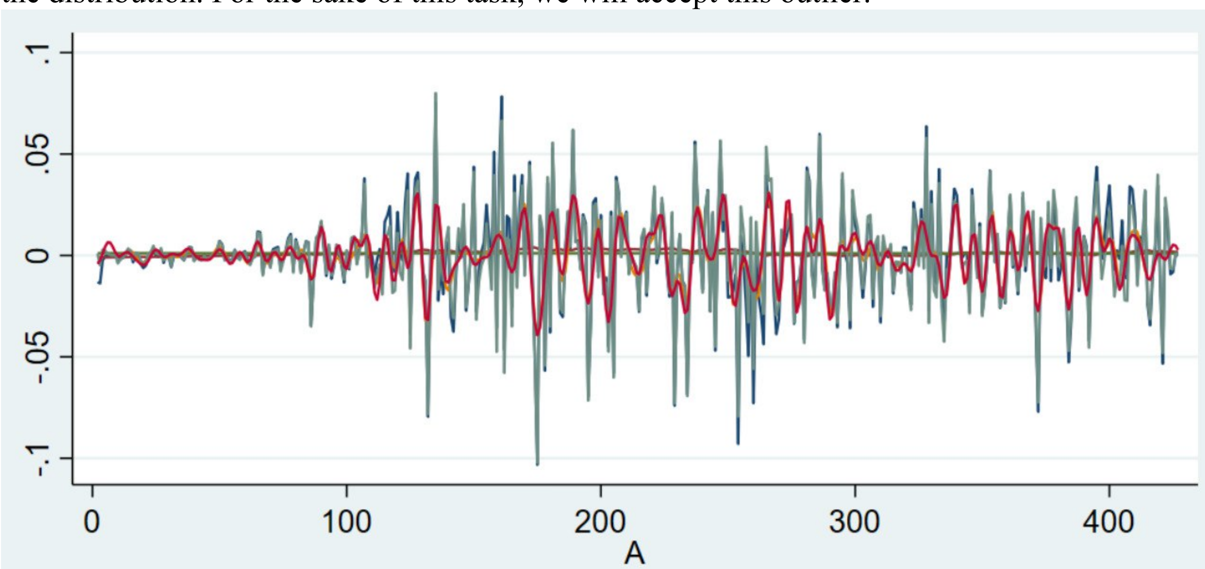
We see that the model makes flat predictions which does not suit us. However, the model performs incredibly well for the 250th period. Also, even though the configuration catches the silhouette of the data for the 100th amazingly the model does not show any steady raise (except for fluctuations) for periods 100-120. Overall, this model 'passed' 2 out of our 4 forecasting tests

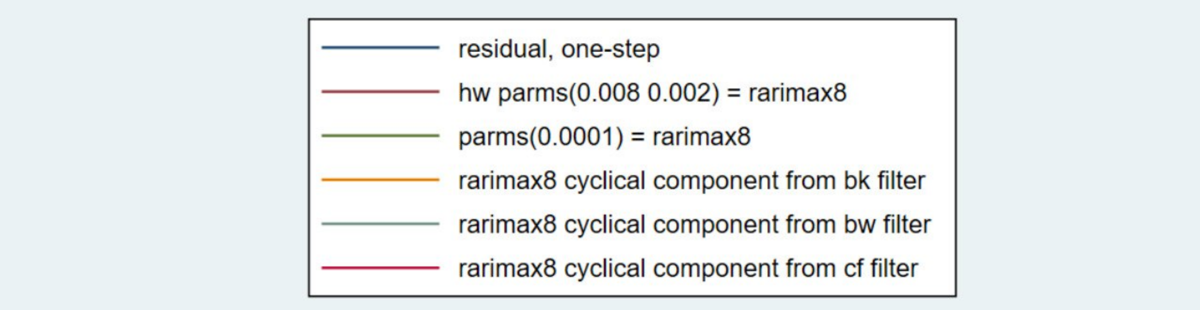
However, due to the assumption breach, we decided to try the same ARIMAX model for the filtered data. We have also added the 11th currency since it turned out to be statistically significant.

When choosing filter, we started with The Baxter-King filter that is intended to explicitly deal with the periodicity of a cycle. Currencies have unpredictable cycles in it, and therefore the theory would not suggest to use this filter. By applying their band-pass filter to a series, they produce a new series that does not contain fluctuations at higher or lower than those of the cycle. We see that the filter cut out only the noise in the upper and lower part of the distribution.

Then, The Butterworth filter is a high-pass filter, meaning that it only removes the low-frequency components. That is what happened here. The filter performed fairly well on these data, but we still see a lot of outliers in the spectral density function.

The visualization of the CF-filter looks like the best performance. It identified the stochastic noise in the cycle and cut out the right data. But we still have one outlier in the beginning of the distribution. For the sake of this task, we will accept this outlier.

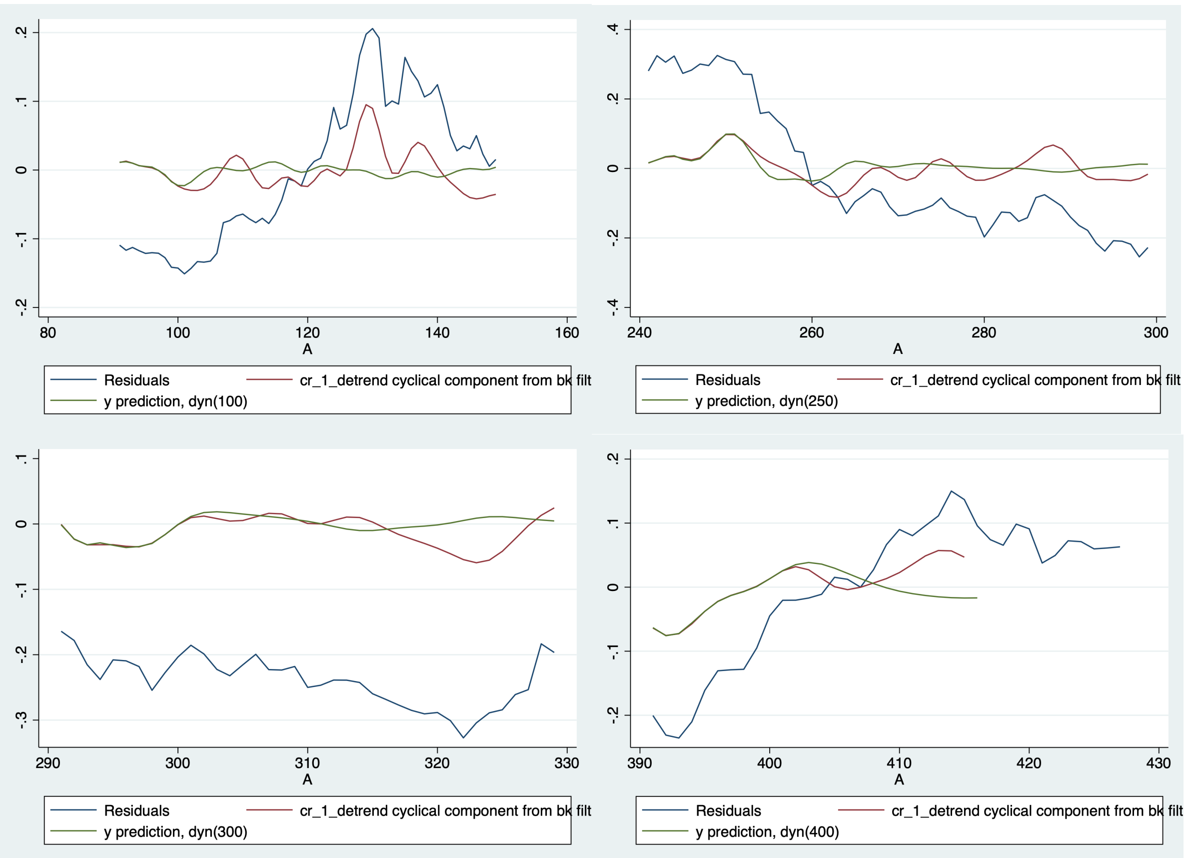




So,
-The Butterworth-filter only cut out the peaks.
- Holt Winters cut out nearly everything and we should not use this one, because of the loss of information.
- The Christiano-FItzgerald cut out outliers and still shows a lot of information (red graph). We should use this one.

However, even though the plot for comparison and the pergrams let us conclude to use the fitzgerald filter, we stayed with the BK filter. The reason for that is that STATA did not have enough computing powers to include all necessary AR and MA terms to get rid of AC showing us ‘initial values not feasible’. So, below are the calculations for BK filter

```
arima BTCfiltered l._11filtered l._5filtered, ar(1,2,5,9) ma(1,2,3,4,5,7,9)
```



We see, the filtered forecasting performance turned out to be too bad to be considered and ends up just showing straight lines. So, we would stand with the following ARIMAX model:

```
arima  d.cr_1_detrend  d.l.cr_5_detrend,  ar(1,  4)  ma(8,  26)  [i.e
ARIMA({1,4},{1},{8,26})]
```

VEC

Next, we tried VECM model. We also used the 5th currency, since it is also unit-rooted the same way and is directly connected with BTC as we discovered in the introduction part.

```
.  dfuller cr_5_detrend

Dickey-Fuller test for unit root          Number of obs  = 427
Variable: cr_5_detrend                    Number of lags =   0

H0: Random walk without drift, d = 0

                                       Dickey-Fuller
                   Test     ________ critical value ________
              statistic           1%           5%          10%
______________________________________________________________
 Z(t)            -1.911       -3.446       -2.873       -2.570
______________________________________________________________
MacKinnon approximate p-value for Z(t) = 0.3270.
```

```
. varsoc cr_5_detrend cr_1_detrend, maxlag(30)

Lag-order selection criteria

   Sample: 30 thru 427                              Number of obs = 398
```

Lag	LL	LR	df	p	FPE	AIC	HQIC	SBIC
0	339.178				.00063	-1.69436	-1.68643	-1.67433
1	1848.32	3018.3	4	0.000	3.3e-07	-9.2579	-9.23409	-9.1978
2	1868.28	39.92	4	0.000	3.0e-07	-9.3381	-9.29843*	-9.23794*
3	1873.77	10.972	4	0.027	3.0e-07*	-9.34557*	-9.29002	-9.20534
4	1875.48	3.4276	4	0.489	3.0e-07	-9.33408	-9.26267	-9.15379
5	1880.98	10.995	4	0.027	3.0e-07	-9.3416	-9.25432	-9.12125
6	1881.7	1.4321	4	0.839	3.1e-07	-9.3251	-9.22195	-9.06468
7	1885.87	8.3554	4	0.079	3.1e-07	-9.32599	-9.20697	-9.02551
8	1893.34	14.934	4	0.005	3.0e-07	-9.34342	-9.20853	-9.00286
9	1897.13	7.585	4	0.108	3.0e-07	-9.34237	-9.19161	-8.96176
10	1898.73	3.1972	4	0.525	3.0e-07	-9.33031	-9.16368	-8.90962
11	1900.49	3.5228	4	0.474	3.1e-07	-9.31906	-9.13656	-8.85831
12	1901.83	2.6725	4	0.614	3.1e-07	-9.30567	-9.1073	-8.80486
13	1906.01	8.3549	4	0.079	3.1e-07	-9.30656	-9.09233	-8.76569
14	1912.82	13.626	4	0.009	3.1e-07	-9.3207	-9.09059	-8.73976
15	1914.28	2.9152	4	0.572	3.1e-07	-9.30792	-9.06195	-8.68692
16	1915.29	2.0281	4	0.731	3.2e-07	-9.29292	-9.03107	-8.63185
17	1917.22	3.8564	4	0.426	3.2e-07	-9.28251	-9.00479	-8.58137
18	1922.17	9.8982	4	0.042	3.2e-07	-9.28727	-8.99369	-8.54607
19	1923.46	2.5897	4	0.629	3.2e-07	-9.27368	-8.96423	-8.49242
20	1940.11	33.298	4	0.000	3.0e-07	-9.33724	-9.01192	-8.51591
21	1944.88	9.5308	4	0.049	3.0e-07	-9.34109	-8.9999	-8.4797
22	1945.93	2.1063	4	0.716	3.1e-07	-9.32628	-8.96922	-8.42482
23	1949.13	6.3965	4	0.171	3.1e-07	-9.32225	-8.94932	-8.38073
24	1950.25	2.2354	4	0.693	3.1e-07	-9.30777	-8.91897	-8.32618
25	1957.49	14.493	4	0.006	3.1e-07	-9.32408	-8.91942	-8.30243
26	1962.33	9.6678*	4	0.046	3.1e-07	-9.32827	-8.90774	-8.26656
27	1963.03	1.4055	4	0.843	3.1e-07	-9.31171	-8.8753	-8.20992
28	1964.8	3.5321	4	0.473	3.1e-07	-9.30048	-8.8482	-8.15863
29	1966.45	3.3074	4	0.508	3.2e-07	-9.28869	-8.82054	-8.10678
30	1966.88	.8553	4	0.931	3.2e-07	-9.27074	-8.78672	-8.04876

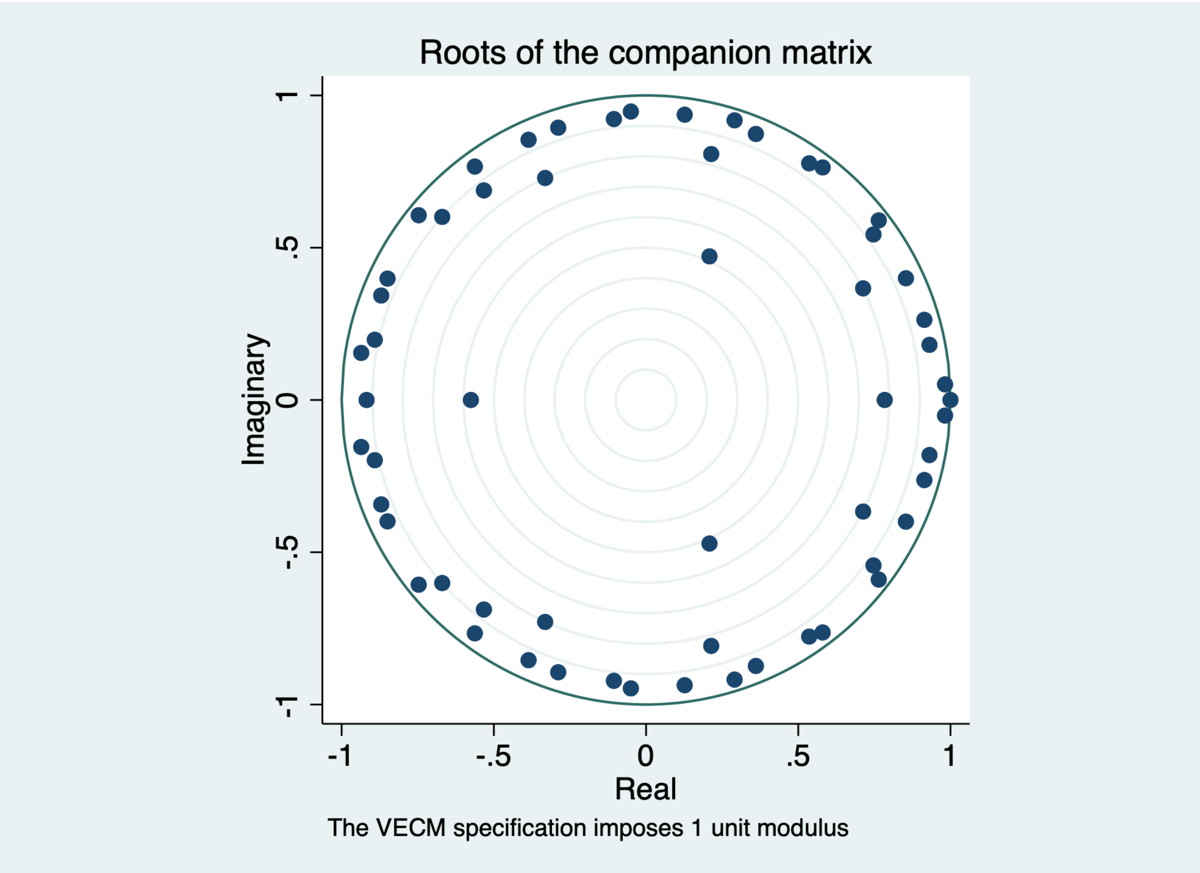
```
veclmar, mlag(30)

Lagrange-multiplier test
```

lag	chi2	df	Prob > chi2
1	4.0156	4	0.40390
2	7.9804	4	0.09230
3	3.4428	4	0.48663
4	9.9467	4	0.04134
5	4.1356	4	0.38797
6	14.0922	4	0.00701
7	12.1829	4	0.01604
8	6.2743	4	0.17958
9	3.4281	4	0.48889
10	5.1408	4	0.27315
11	4.2314	4	0.37560
12	10.5592	4	0.03199
13	15.2095	4	0.00429
14	2.3436	4	0.67284
15	2.3608	4	0.66973
16	3.9864	4	0.40786
17	16.5495	4	0.00236
18	7.5550	4	0.10931
19	21.4684	4	0.00026
20	17.6686	4	0.00143
21	3.1021	4	0.54089
22	3.9561	4	0.41197
23	2.4548	4	0.65275
24	5.8727	4	0.20886
25	9.1994	4	0.05630
26	4.4818	4	0.34471
27	1.2121	4	0.87610
28	1.1663	4	0.88361
29	1.2561	4	0.86878
30	5.4626	4	0.24303

H0: no autocorrelation at lag order

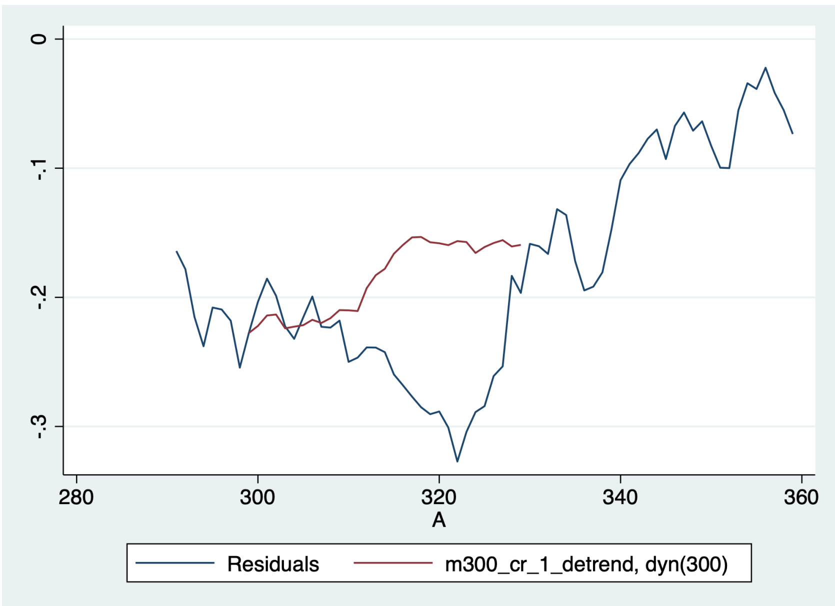
Initialy we started with `vec cr_5_detrend cr_1_detrend, rank(1) lag(3)` model but as shown below we do not deal correctly with AC in lags # 4, 6, 7, 12,13, 19 and 20. Iteratevely we cameup with the model of lag(29). Obviously, it is a gigantic model but it is the least that could be done to deal with AC. It also leads to the biggest part of all coefficients being statistically insignificant, however, mathematically it is impossible to get rid of them.

The final VEC model is `vec cr_5_detrend cr_1_detrend, rank(1) lag(29)` (we use one rank since there are only two variables. We tried to add more IVs, but they all turned out to be insignificant). The graph below proves the model stability.

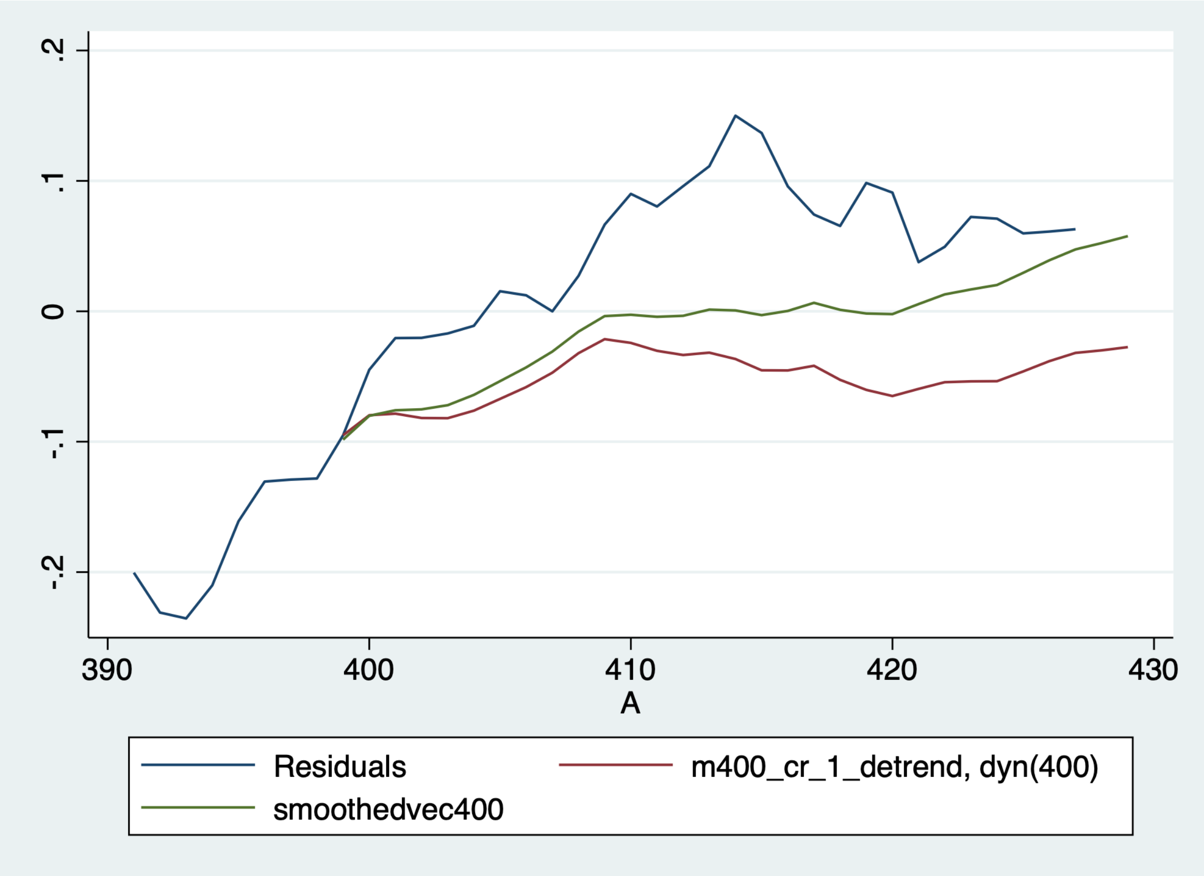


From the performance we can conclude several things. On the hand, the model predict specific dips worse than our ARIMAX model (for example, the 100th dynamic predictions) and is generally far smoother. We suggest, that the reason is the excessive tail of lags. Probably, lags harmonize each other and soothe the line. However, on the other hand, the VECM makes better trend predictions. If 2/4 times ARIMAX makes straight lines, VECM make up/downward going predictions that match the overal movement of observations.

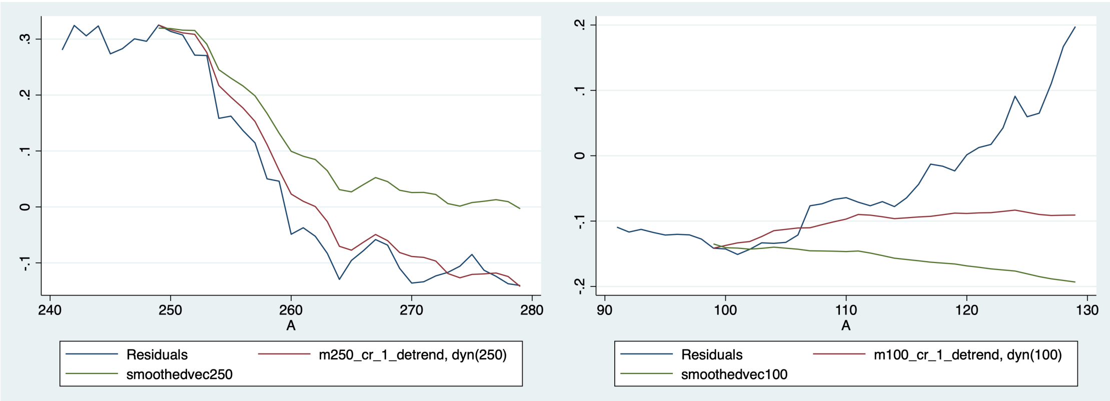
It may seem that for 300th period VECM did a bad job, predicting completely opposite movement from the actual. However, the dip is a well-known unexplained drop. However, if we take a look at a broader period, it is obvious that the model predicted the currency exchange range the way it would be if it did not exeperience the drop. So, we cannot blame the VECM for this error and may actually conclude it did a better performance than ARIMAX, that drew just a straight line, which is wrong in any way.



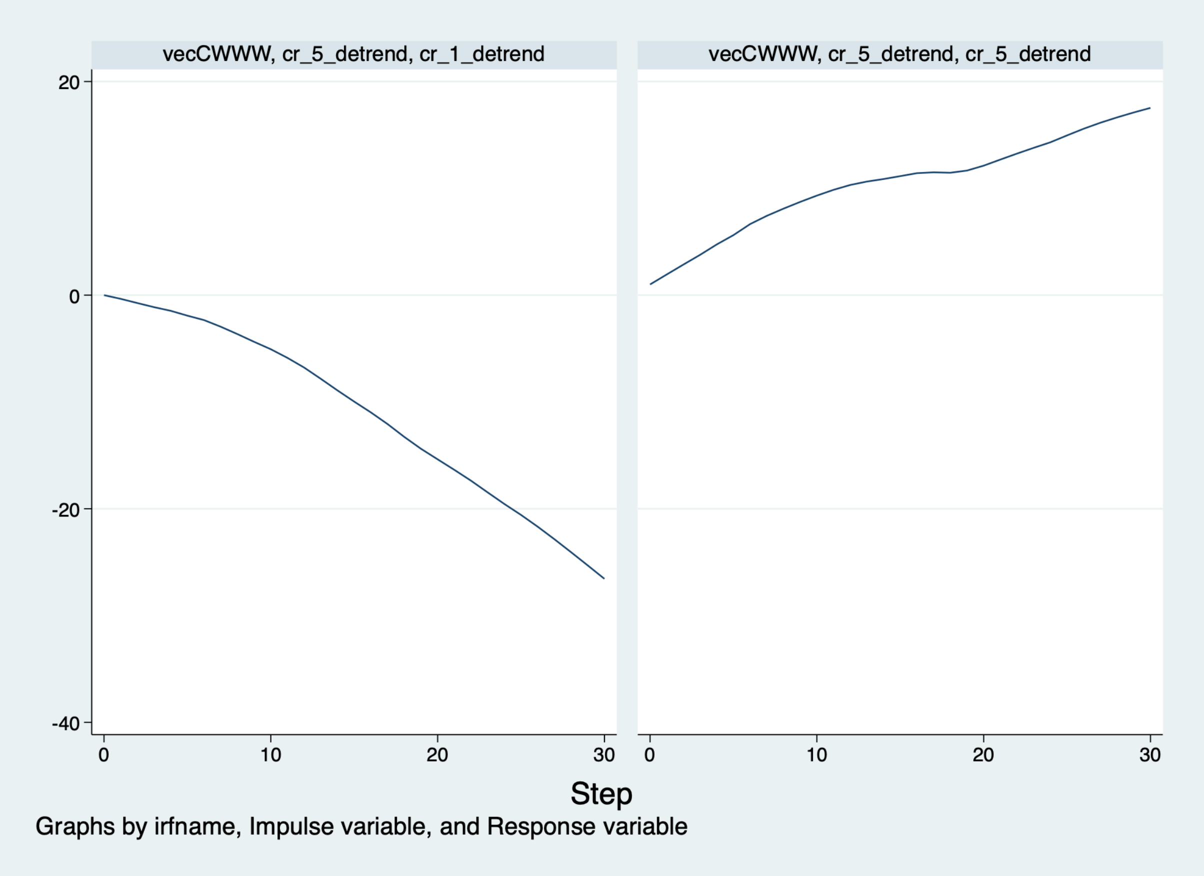
We can also mark another VECM draback: it dramatically underestimates the shifts. Its forecasts cath the verall movement correctly but mess up with the absolute values. It is also caused by the long tail that the model has. We attempted to correct the situation with adding smoother hw predictions and generating the new variable that consists 50% of smoother prediction and 50% of VECM. The weights could be changed by the user, but the main idea behid is the following. Let's take a look at the 400th dynamic prediction.



When stanfing at the 400th period, the model suggests that the raise in the exchange rate will not be big based on previous information. However, if the analyst knows that this is a new trend and the raise is due to not some unexplained crypto nature but because of new laws or hype, they are recommended to add filter to enhance the positions of the new recent trend. This trick, however, should be done cautiously, since it can seriously worsen predictions.



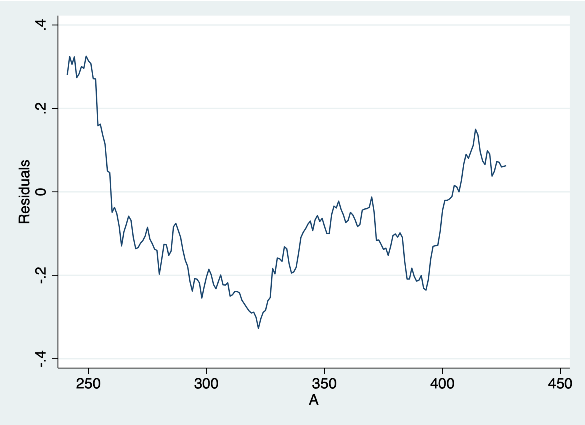
Overall, VECM model is quite a good predictor for our goal. We wanted to train a model that would correctly predict raise/fall and VECM does this pretty well, even though the absolut numbers are not a complete match. BTC-cash helped a lot, since this currency is directly tied with BTC.



We would suggest further experiments with the model:

1) Take a bigger period (we have only 1 year)
2) Get more variables into the model
3) Try larger lags.

Illustrating the third suggestion, we can see that probably there is some sort of cyclicality from 300^{th} period that is not caught due to its big span



Conclusions

In this project we have created 2 adequately performing models: ARIMAX and VECM. ARIMAX is like a lotttery: it either suggests a plain line or predicts the fluctuations and cathes new microtrends with amazing quality. VECM, on the other side, is less sensitive and does worse job in predicting absolute numbers, yet it does better job in prediting the overall dynamics outperforming ARIMAX. Hence, we would recommend the following schema: use both models. If they suggest the same trend, chose ARIMAX, it is more detailed. If ARIMAX suggests a straignt line, choose VECM