[BatchGetSymbols](https://github.com/msperlin/BatchGetSymbols" \t "_blank) is my most downloaded package by any count. Computation time, however, has always been an issue. While downloading data for 10 or less stocks is fine, doing it for a large ammount of tickers, say the SP500 composition, gets very boring.

I’m glad to report that time is no longer an issue. Today I implemented a parallel option for BatchGetSymbols. If you have a high number of cores in your computer, you can seriously speep up the importation process. Importing SP500 compositition, over 500 stocks, is a breeze.

Give a try. The new version is already available in Github:

devtools::install\_github('msperlin/BatchGetSymbols')

It should be in CRAN soon.

**How to use parallel**

Very simple. Just set you parallel plan with future::plan and use input do.parallel = TRUE in BatchGetSymbols. If you are not sure how many cores you have available, just run the following code to figure it out:

future::availableCores()

## system

## 16

#devtools::install\_github('msperlin/BatchGetSymbols')

library(BatchGetSymbols)

# get tickers from SP500

df.sp500 <- GetSP500Stocks()

tickers <- df.sp500$tickers

future::plan(future::multisession,

workers = 10) # use 10 cores (future::availableCores())

# dowload data for 50 stocks

l.out <- BatchGetSymbols(tickers = tickers[1:50],

first.date = '2010-01-01',

do.parallel = TRUE,

do.cache = FALSE)

##

## Running BatchGetSymbols for:

## tickers = MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AMG, AFL, A, APD, AKAM, ALK, ALB, ARE, ALXN, ALGN, ALLE, AGN, ADS, LNT, ALL, GOOGL, GOOG, MO, AMZN, AEE, AAL, AEP, AXP, AIG, AMT, AWK, AMP, ABC, AME, AMGN, APH, APC, ADI, ANSS, ANTM, AON, AOS, APA, AIV, AAPL

## Downloading data for benchmark ticker

## ^GSPC | yahoo (1|1)

## Running parallel BatchGetSymbols with 10 cores (16 available)

##

##

Progress: ──────────────────────────────────────────────────────────────────────────────────────── 100%

Progress: ────────────────────────────────────────────────────────────────────────────────────────────── 100%

Progress: ────────────────────────────────────────────────────────────────────────────────────────────── 100%

Progress: ──────────────────────────────────────────────────────────────────────────────────────────────── 100%

##

##

## MMM | yahoo (1|50) - Got 100% of valid prices | Good job!

## ABT | yahoo (2|50) - Got 100% of valid prices | OK!

## ABBV | yahoo (3|50) - Got 67.7% of valid prices | OUT: not enough data (thresh.bad.data = 75.0%)

## ABMD | yahoo (4|50) - Got 100% of valid prices | OK!

## ACN | yahoo (5|50) - Got 100% of valid prices | Good job!

## ATVI | yahoo (6|50) - Got 100% of valid prices | OK!

## ADBE | yahoo (7|50) - Got 100% of valid prices | Good stuff!

## AMD | yahoo (8|50) - Got 100% of valid prices | Feels good!

## AAP | yahoo (9|50) - Got 100% of valid prices | Good stuff!

## AES | yahoo (10|50) - Got 100% of valid prices | OK!

## AMG | yahoo (11|50) - Got 100% of valid prices | Feels good!

## AFL | yahoo (12|50) - Got 100% of valid prices | Good stuff!

## A | yahoo (13|50) - Got 100% of valid prices | Nice!

## APD | yahoo (14|50) - Got 100% of valid prices | Feels good!

## AKAM | yahoo (15|50) - Got 100% of valid prices | Nice!

## ALK | yahoo (16|50) - Got 100% of valid prices | Nice!

## ALB | yahoo (17|50) - Got 100% of valid prices | Youre doing good!

## ARE | yahoo (18|50) - Got 100% of valid prices | Got it!

## ALXN | yahoo (19|50) - Got 100% of valid prices | OK!

## ALGN | yahoo (20|50) - Got 100% of valid prices | OK!

## ALLE | yahoo (21|50) - Got 58.2% of valid prices | OUT: not enough data (thresh.bad.data = 75.0%)

## AGN | yahoo (22|50) - Got 100% of valid prices | Nice!

## ADS | yahoo (23|50) - Got 100% of valid prices | Good stuff!

## LNT | yahoo (24|50) - Got 100% of valid prices | Got it!

## ALL | yahoo (25|50) - Got 100% of valid prices | OK!

## GOOGL | yahoo (26|50) - Got 100% of valid prices | OK!

## GOOG | yahoo (27|50) - Got 100% of valid prices | Good job!

## MO | yahoo (28|50) - Got 100% of valid prices | Got it!

## AMZN | yahoo (29|50) - Got 100% of valid prices | Looking good!

## AEE | yahoo (30|50) - Got 100% of valid prices | Youre doing good!

## AAL | yahoo (31|50) - Got 100% of valid prices | Got it!

## AEP | yahoo (32|50) - Got 100% of valid prices | OK!

## AXP | yahoo (33|50) - Got 100% of valid prices | Well done!

## AIG | yahoo (34|50) - Got 100% of valid prices | Nice!

## AMT | yahoo (35|50) - Got 100% of valid prices | Youre doing good!

## AWK | yahoo (36|50) - Got 100% of valid prices | Mais contente que cusco de cozinheira!

## AMP | yahoo (37|50) - Got 100% of valid prices | Good job!

## ABC | yahoo (38|50) - Got 100% of valid prices | Looking good!

## AME | yahoo (39|50) - Got 100% of valid prices | Got it!

## AMGN | yahoo (40|50) - Got 100% of valid prices | Looking good!

## APH | yahoo (41|50) - Got 100% of valid prices | Well done!

## APC | yahoo (42|50) - Got 100% of valid prices | Well done!

## ADI | yahoo (43|50) - Got 100% of valid prices | Well done!

## ANSS | yahoo (44|50) - Got 100% of valid prices | Looking good!

## ANTM | yahoo (45|50) - Got 100% of valid prices | Feels good!

## AON | yahoo (46|50) - Got 100% of valid prices | Got it!

## AOS | yahoo (47|50) - Got 100% of valid prices | Well done!

## APA | yahoo (48|50) - Got 100% of valid prices | Good stuff!

## AIV | yahoo (49|50) - Got 100% of valid prices | Youre doing good!

## AAPL | yahoo (50|50) - Got 100% of valid prices | Looking good!

glimpse(l.out)

## List of 2

## $ df.control:Classes 'tbl\_df', 'tbl' and 'data.frame': 50 obs. of 6 variables:

## ..$ ticker : chr [1:50] "MMM" "ABT" "ABBV" "ABMD" ...

## ..$ src : chr [1:50] "yahoo" "yahoo" "yahoo" "yahoo" ...

## ..$ download.status : chr [1:50] "OK" "OK" "OK" "OK" ...

## ..$ total.obs : int [1:50] 2335 2335 1581 2335 2335 2335 2335 2335 2335 2335 ...

## ..$ perc.benchmark.dates: num [1:50] 1 1 0.677 1 1 ...

## ..$ threshold.decision : chr [1:50] "KEEP" "KEEP" "OUT" "KEEP" ...

## $ df.tickers:'data.frame': 112080 obs. of 10 variables:

## ..$ price.open : num [1:112080] 83.1 82.8 83.9 83.3 83.7 ...

## ..$ price.high : num [1:112080] 83.4 83.2 84.6 83.8 84.3 ...

## ..$ price.low : num [1:112080] 82.7 81.7 83.5 82.1 83.3 ...

## ..$ price.close : num [1:112080] 83 82.5 83.7 83.7 84.3 ...

## ..$ volume : num [1:112080] 3043700 2847000 5268500 4470100 3405800 ...

## ..$ price.adjusted : num [1:112080] 65.8 65.4 66.3 66.4 66.8 ...

## ..$ ref.date : Date[1:112080], format: "2010-01-04" ...

## ..$ ticker : chr [1:112080] "MMM" "MMM" "MMM" "MMM" ...

## ..$ ret.adjusted.prices: num [1:112080] NA -0.006264 0.014182 0.000717 0.007046 ...

## ..$ ret.closing.prices : num [1:112080] NA -0.006264 0.014182 0.000717 0.007046 ...