# From Technical Debt to Technical Capital

Denali Lumma

denali@doubling.io

- Engineering at Netflix, 23andMe, Okta, Salesforce, and Uber
- Founder at Doubling, offering technical advisory services to portfolio companies at Y Combinator and other venture capital groups
- Community Guide, DevOps Research and Assessment (DORA)
- Lead the Software Development Life Cycle group at Modular
- Based in Los Gatos, California with my husband and two children

Assertion:

Accurate forecasting of future value of a codebase or a company that produces code is possible via "cycle time variance."

# Agenda

- Technical Debt
- Technical Capital
- Forecasting in Finance
- Time Series Data
- Gaussian Distribution, Mean, Median
- Standard Deviation, T-Score, Z-Score
- Changes Over Time
- Reduce Variation, Fields, Thinkers
- Cycle Time
- Cycle Time Variance

Open Source Examples

- Machine Learning Frameworks
- Databases
- Browsers
- Frontend Frameworks
- Code Editors

# The Tech Debt Metaphor

A lot of people use the term "technical debt" to describe the state of an organization's code base. The implication is that "technical debt" is like consumer credit card debt.

The more "tech debt" you take on, the less new features can be made in the future.

In other words, "tech debt" is compounding.

When we talk about "tech debt" we are forecasting future state of the code base.

"An ounce of prevention is worth a pound of cure.

Benjamin Franklin

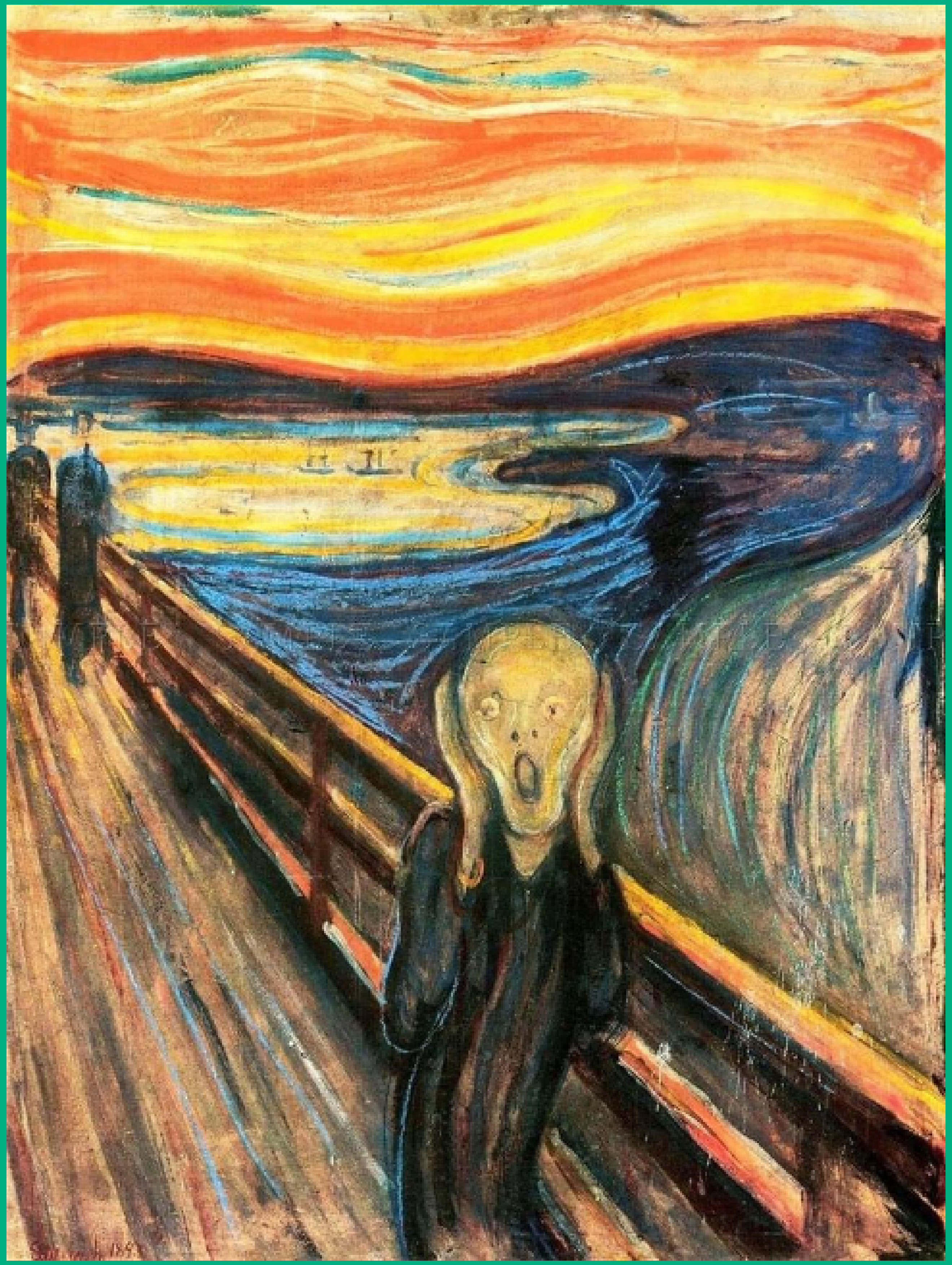
Tech debt != credit card debt

The primary problem to be solved with software development is understanding

# What do we mean when we say "tech debt"?

Technical debt is simply the answer to the question:
How hard or easy is it to comprehend the system?
How understandable is the system?

# Technical debt is like losing your mind



Technical Debt is working with a system that can not be understood or is very hard to understand

# How much technical debt do you have?

The answer is simply how easily can the system be understood?

# Understandability is easy to measure

How quickly can a change be made? Cycle time.

Rather than "technical debt", a better way to understand the state of the code base is through the lens of its future value. We can refer to this as "technical capital".

In the world of finance, there are metrics used to forecast the future value of a company. The same frameworks can also be used to forecast the future value of a code base.

# Forecasting

# Time Series Data

Time series data is a type of data that is recorded over a series of consecutive and equally spaced time intervals. In time series data, each data point is associated with a specific timestamp, allowing for the analysis of how a particular variable changes over time.

# Financial Time Series Data Example

## Closing Stock Price by Day

| Date | Closing Price (USD) |
| --- | --- |
| 2023-09-01 | 100.00 |
| 2023-09-04 | 101.25 |
| 2023-09-05 | 99.75 |
| 2023-09-06 | 102.50 |
| 2023-09-07 | 103.75 |
| 2023-09-08 | 104.50 |
| 2023-09-11 | 105.25 |
| 2023-09-12 | 105.00 |
| 2023-09-13 | 107.00 |
| 2023-09-14 | 106.25 |

The "Date" column represents the trading days when the stock market was open.

The "Closing Price (USD)" column contains the closing prices of the stock for each trading day. The closing price is the price at which the stock traded when the market closed for the day.

# Software Development Life Cycle Time Series Data Example

## Code Change Cycle Time (85th Percentile) by Day

| Date | Cycle Time (Minutes 85th Percentile) |
| --- | --- |
| 2023-09-01 | 700 |
| 2023-09-04 | 701 |
| 2023-09-05 | 997 |
| 2023-09-06 | 1025 |
| 2023-09-07 | 1037 |
| 2023-09-08 | 1045 |
| 2023-09-11 | 105 |
| 2023-09-12 | 1050 |
| 2023-09-13 | 1070 |
| 2023-09-14 | 106 |

The "Date" column represents the active development days.

The "Cycle Time" column contains the 85th percentile time to complete a change that finished on that day.

# Mean

## (Value Based)

The sum of all values divided by the number of values. Also known as the average.

# Median

## (Position Based)

The median represented the middle value in a dataset when it is ordered as a list in ascending order. It is the value that separates the higher half from the lower half of the data.

# Percentile

The median is the 50th percentile. Other percentiles can be calculated as well, for example, the 25th percentile or the 75th percentile.

# Gaussian Distribution

A Gaussian Distribution, also known as a "bell curve," is a way of showing how things are usually grouped together. It's called a bell curve because it looks like a bell when we draw it. In this kind of curve, most things are in the middle, and only a few are very different or unusual. It helps us understand how common or uncommon something is in a group.

# Key Properties

**Mean (μ)**

The mean of a Gaussian distribution represents the central or expected value of the distribution. It is the point around which the data is centered, and it corresponds to the peak of the bell-shaped curve.

**Standard Deviation (σ)**

The standard deviation of a Gaussian distribution measures the spread or dispersion of the data. A smaller standard deviation indicates that data points are clustered closely around the mean, while a larger standard deviation indicates more spread.

# Example Data Set

Example data set:

[1, 1, 1, 1, 2, 10, 17]

# What is the average or mean?

To find the average (also known as the mean) of a data set, you sum up all the values in the set and then divide by the total number of values. In this case, the data set is [1, 1, 1, 1, 2, 10, 17], and there are 7 values in the set.

So, you sum up all the values:

$1 + 1 + 1 + 1 + 2 + 10 + 17 = 33$

Now, divide the sum by the total number of values (7):

Average = $33 / 7 \approx 4.7143$ (rounded to four decimal places)

So, the average of the data set is approximately 5.

# What is the median?

## The median is different than the mean (or average).

To find the median of a data set, ensure the data in arranged in ascending order (from lowest to highest) and then determine the middle value. If there is an even number of values, you take the average of the two middle values.

For:

[1, 1, 1, 1, 2, 10, 17]

There are 7 values in the data set, which is an odd number. The middle value is the fourth value when they are arranged in ascending order:

1, 1, 1, [1], 2, 10, 17

So, the median of the data set is 1.

# What is the 75th percentile?

Calculate the position of 75 in the dataset:

Position = (Percentile / 100) * (N + 1)

Where N is the total number of data points. In this case, N = 7 (there are 7 data points).

Position = (75 / 100) * (7 + 1) = 0.75 * 8 = 6

So, the 75th percentile falls at the position 6 in the ordered data set. Now, find the value at this position, which is the 6th value in the ordered list:

The 6th value in the ordered list is 10.

Therefore, the 75th percentile of the data set [1, 1, 1, 1, 2, 10, 17] is 10.

# What is the standard deviation of the data set?

To calculate the standard deviation of a data set, follow these steps:

1. Find the mean (average) of the data set.
2. Subtract the mean from each data point to find the deviation from the mean for each data point.
3. Square each of these deviations.
4. Find the mean of the squared deviations.
5. Take the square root of the mean of the squared deviations.

## Step 1: Calculate the mean (average)

Mean = $(1 + 1 + 1 + 1 + 2 + 10 + 17) / 7 = 33 / 7 \approx 4.7143$ (rounded to four decimal places)

## Step 2: Calculate the deviation from the mean for each data point

Deviation from Mean = (Data Point - Mean)

Deviation for 1st data point = 1 - 4.7143 ≈ -3.7143

Deviation for 2nd data point = 1 - 4.7143 ≈ -3.7143

Deviation for 3rd data point = 1 - 4.7143 ≈ -3.7143

Deviation for 4th data point = 1 - 4.7143 ≈ -3.7143

Deviation for 5th data point = 2 - 4.7143 ≈ -2.7143

Deviation for 6th data point = 10 - 4.7143 ≈ 5.2857

Deviation for 7th data point = 17 - 4.7143 ≈ 12.2857

# Step 3: Square each of these deviations

Squared Deviation for 1st data point ≈ $(-3.7143)^2 \approx 13.7959$

Squared Deviation for 2nd data point ≈ $(-3.7143)^2 \approx 13.7959$

Squared Deviation for 3rd data point ≈ $(-3.7143)^2 \approx 13.7959$

Squared Deviation for 4th data point ≈ $(-3.7143)^2 \approx 13.7959$

Squared Deviation for 5th data point ≈ $(-2.7143)^2 \approx 7.3842$

Squared Deviation for 6th data point ≈ $(5.2857)^2 \approx 27.9541$

Squared Deviation for 7th data point ≈ $(12.2857)^2 \approx 151.2204$

# Step 4: Find the mean of the squared deviations

Mean of Squared Deviations = (13.7959 + 13.7959 + 13.7959 + 13.7959 + 7.3842 + 27.9541 + 151.2204) / 7 ≈ 35.3473

# Step 5: Take the square root of the mean of the squared deviations to find the standard deviation

Standard Deviation ≈ $\sqrt{(35.3473)} \approx 5.9498$ (rounded to four decimal places)

So, the standard deviation of the data set is approximately 6.

# In Summary

For this dataset: [1, 1, 1, 1, 2, 10, 17]

- Mean: 5
- Median: 1
- 75th Percentile: 10
- Standard Deviation: 6

# T-Score

- In statistics, T-score is a standardized score that indicates how many standard deviations an individual data point is away from the mean of a distribution.
- A positive T-score indicates that the data point is above the mean, while a negative T-score indicates that the data point is below the mean.

The formula for calculating the T-score is:

T-score = $(x - \mu) / (s / \sqrt{n})$

where:

- $x$ is the individual data point you're interested in
- $\mu$ is the mean of the distribution
- $s$ is the standard deviation of the distribution
- $n$ is the number of data points

# Z-Score

- Similar to a T-Score but assumes access to the entire data set.

# Changes Over Time

- Rolling average
- Trendline
- Relationships between T-Score and Z-Score to alert on unusual shifts up or down over time

# Ito Calculus

Ito Calculus is a branch of mathematics used to describe and work with the random movements of things, like stock prices or particles. It helps us understand and predict how these things change over time, taking into account the uncertainty and randomness involved. It's a valuable tool in fields like finance for modeling and analyzing complex systems affected by randomness.

# Harry Markowitz

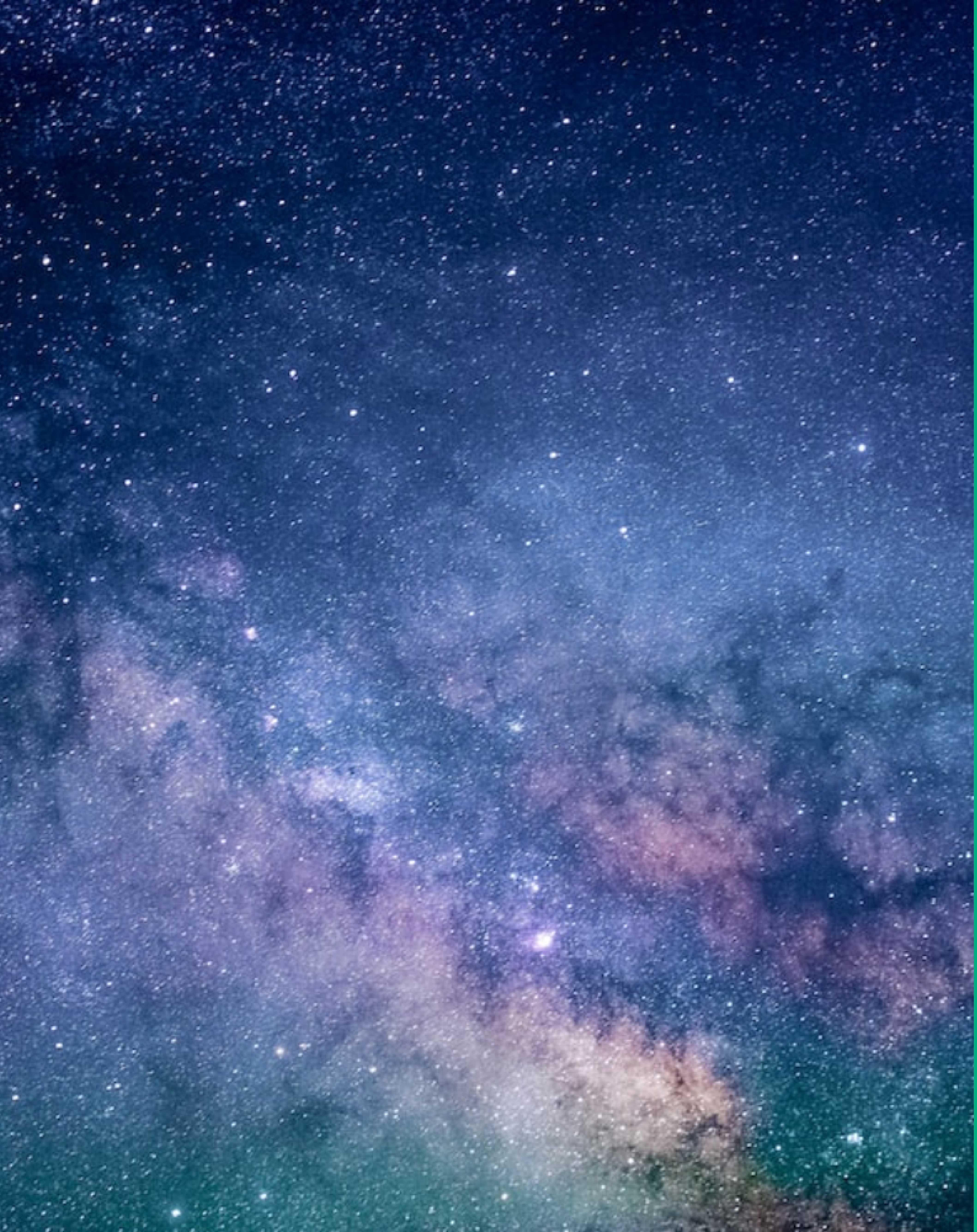
- First big idea: select investments with the highest mean return and lowest variance.
- Evolution of the concept into Modern Portfolio Theory (MPT): Balancing risk and return through diversification.
- Life's work with highest honors: reduce variation.

# Reduction of Variation Across Fields

Manufacturing and Production, Quality Control and Assurance, Healthcare, Supply Chain Management, Statistics, Agriculture, Financial Risk Management, Construction, Transportation and Logistics, Customer Service, Food Production, Energy Efficiency, Energy Management Systems (EMS), Project Management...

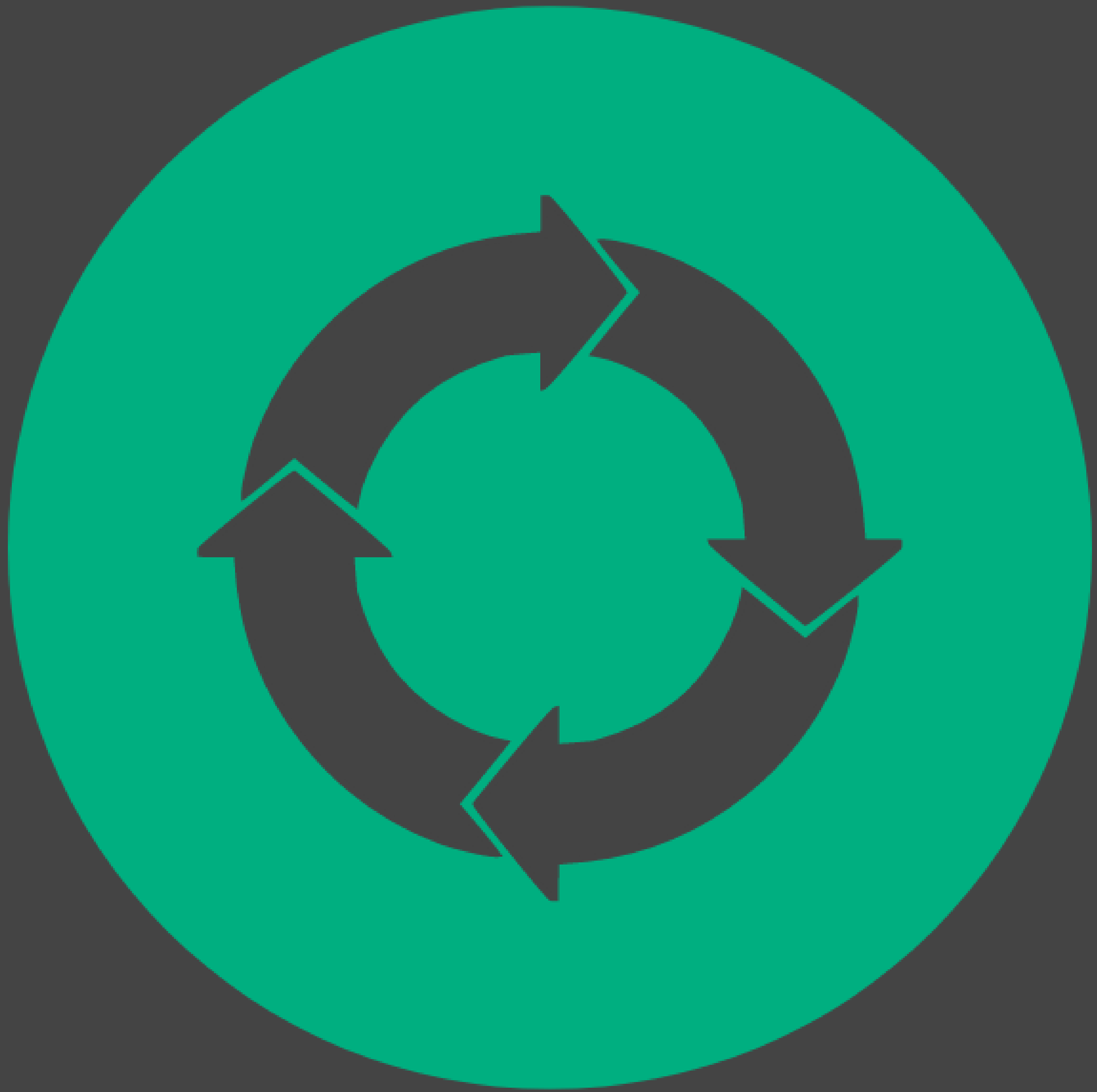
# Reduction of Variation Historical Thinkers

- Frederick W. Taylor (1856-1915)
- Walter A. Shewhart (1891-1967)
- W. Edwards Deming (1900-1993)



# Universal Goal: Reduce Variation

# Measuring Technical Debt and Technical Capital



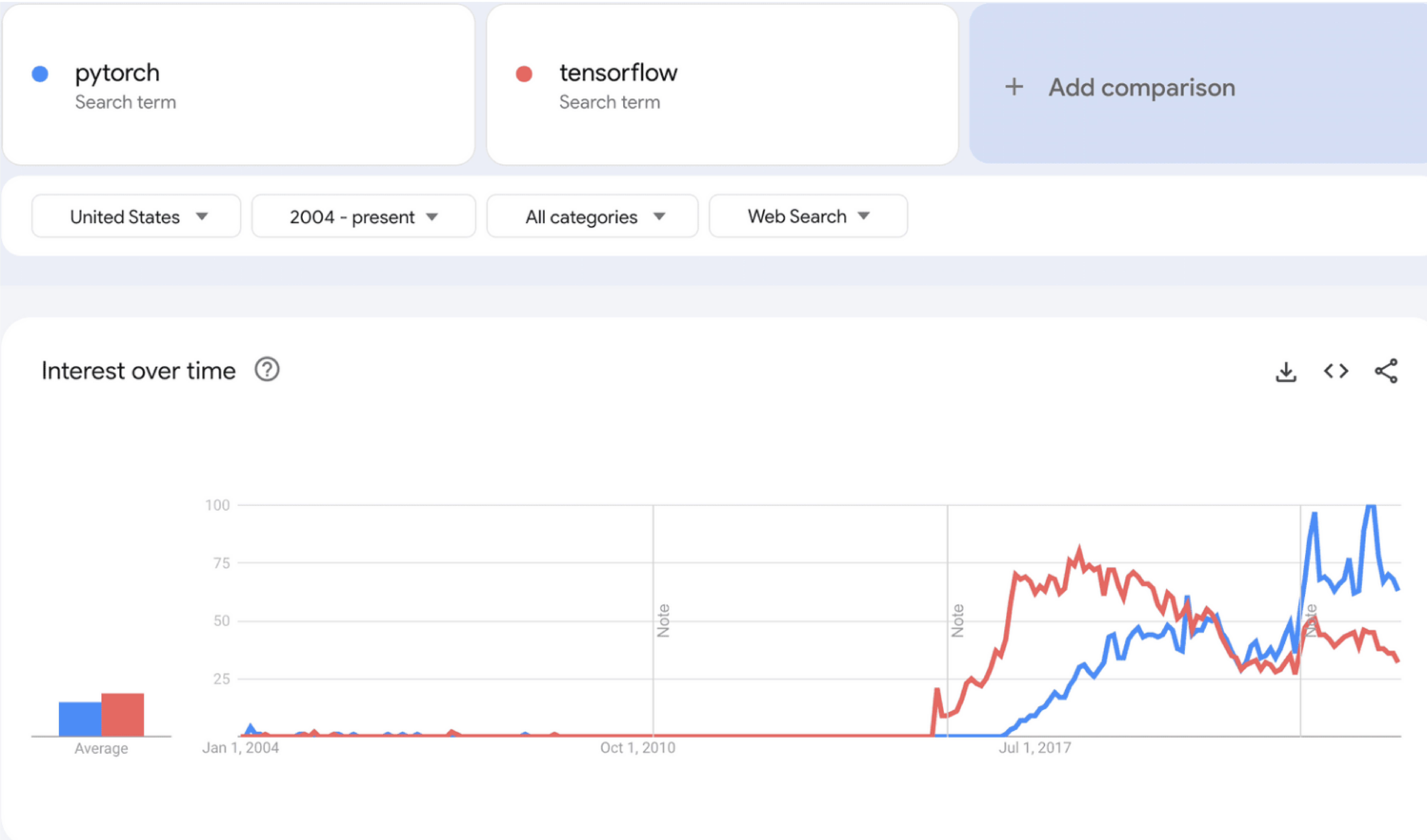
The understandability of a system, (the technical debt) can be measured with cycle time.

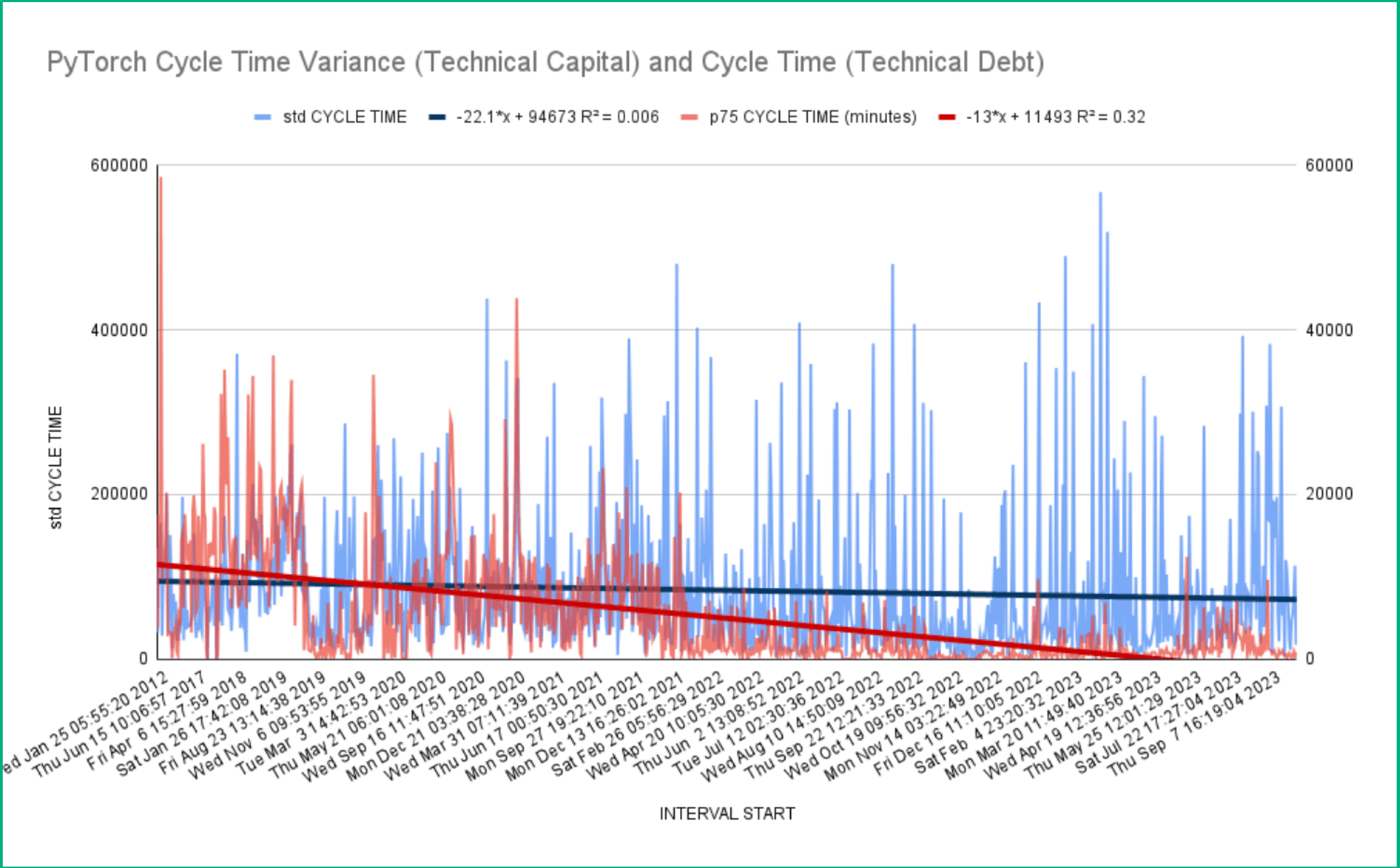
The future value of a code base (the technical capital) can be forecasted with cycle time variance.

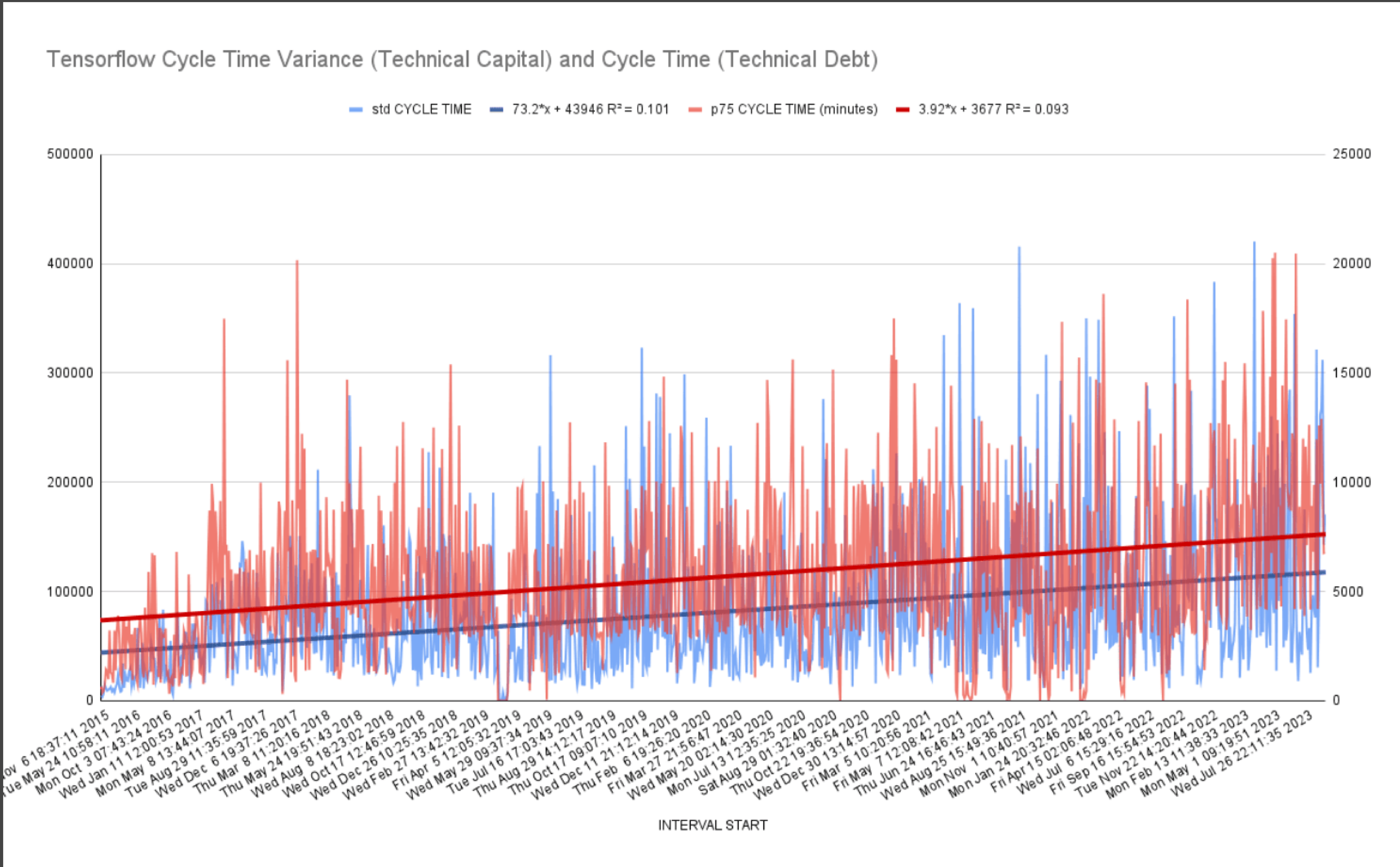
# Open Source Examples

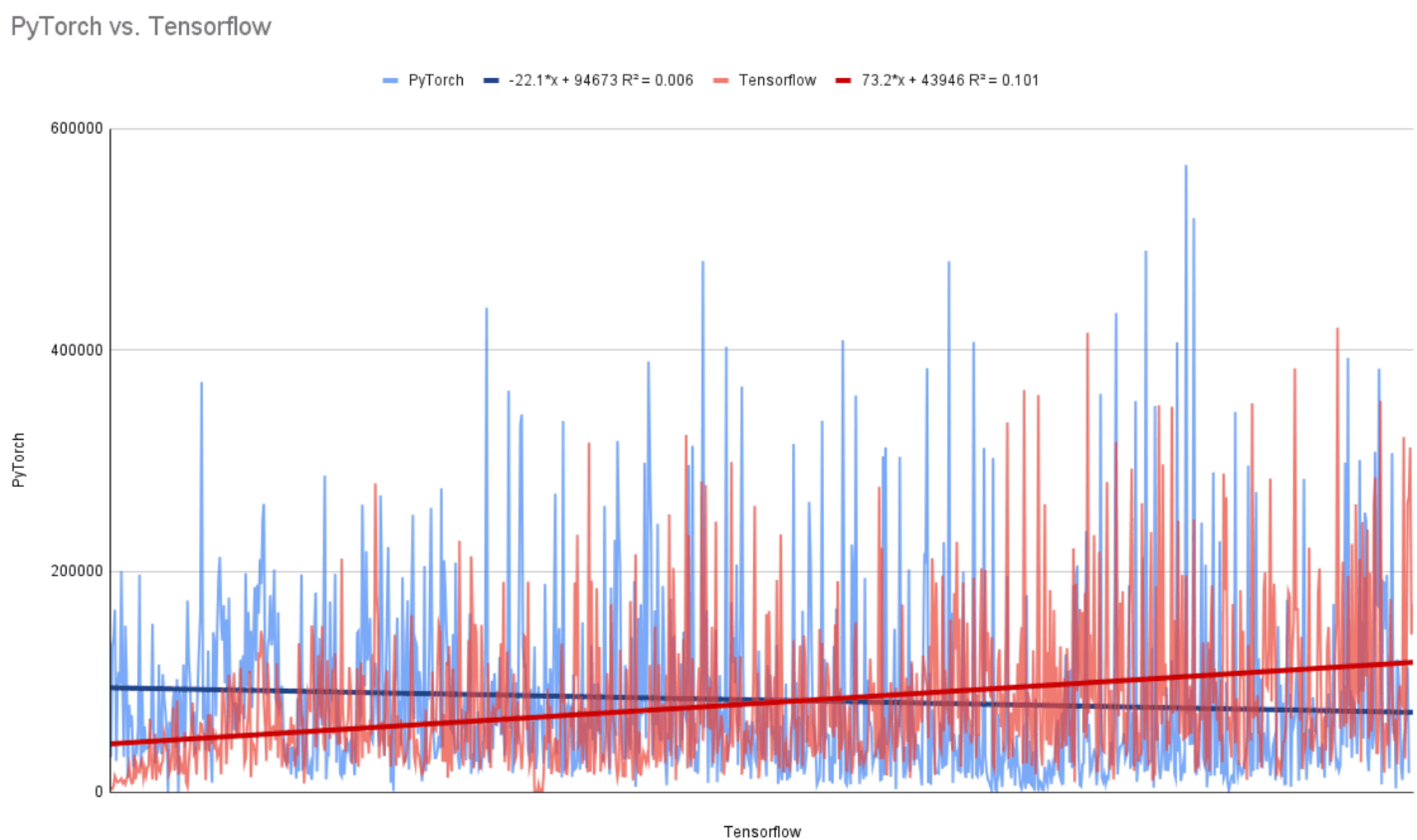
# Machine Learning Frameworks

# Tensorflow | PyTorch


pytorch
Search term
tensorflow
Search term
+ Add comparison
United States
2004 - present
All categories
Web Search
Interest over time
100
75
50
25
Note
Note
Note
Average
Jan 1, 2004
Oct 1, 2010
Jul 1, 2017


PyTorch Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
-22.1*x + 94673 R² = 0.006
p75 CYCLE TIME (minutes)
-13*x + 11493 R² = 0.32
600000
400000
200000
0
60000
40000
20000
0
std CYCLE TIME
INTERVAL START


Tensorflow Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
73.2*x + 43946 R² = 0.101
p75 CYCLE TIME (minutes)
3.92*x + 3677 R² = 0.093
500000
400000
300000
200000
100000
0
25000
20000
15000
10000
5000
0
Tue May 24 10:58:11 2016
Mon Oct 3 07:43:24 2016
Wed Jan 11 12:00:53 2017
Mon May 8 13:44:07 2017
Tue Aug 29 11:35:59 2017
Wed Dec 6 19:37:26 2017
Thu Mar 8 11:20:16 2018
Thu May 24 19:51:43 2018
Wed Aug 8 18:23:02 2018
Wed Oct 17 12:46:59 2018
Wed Dec 26 10:25:35 2018
Wed Feb 27 13:42:32 2019
Fri Apr 5 12:05:32 2019
Wed May 29 09:37:34 2019
Tue Jul 16 17:03:43 2019
Thu Aug 29 14:12:17 2019
Thu Oct 17 09:07:10 2019
Wed Dec 11 21:12:14 2019
Thu Feb 6 19:26:20 2020
Fri Mar 27 21:56:47 2020
Wed May 20 02:14:30 2020
Mon Jul 13 12:35:25 2020
Sat Aug 29 01:32:40 2020
Thu Oct 22 19:36:54 2020
Wed Dec 30 13:14:57 2020
Fri Mar 5 10:20:56 2021
Fri May 7 12:08:42 2021
Thu Jun 24 16:46:43 2021
Wed Aug 25 15:49:36 2021
Mon Nov 1 10:40:57 2021
Mon Jan 24 20:32:46 2022
Fri Apr 15 02:06:48 2022
Wed Jul 6 15:29:16 2022
Fri Sep 16 15:54:53 2022
Tue Nov 22 14:20:44 2022
Mon Feb 13 11:38:33 2023
Mon May 1 09:19:51 2023
Wed Jul 26 22:11:35 2023
INTERVAL START

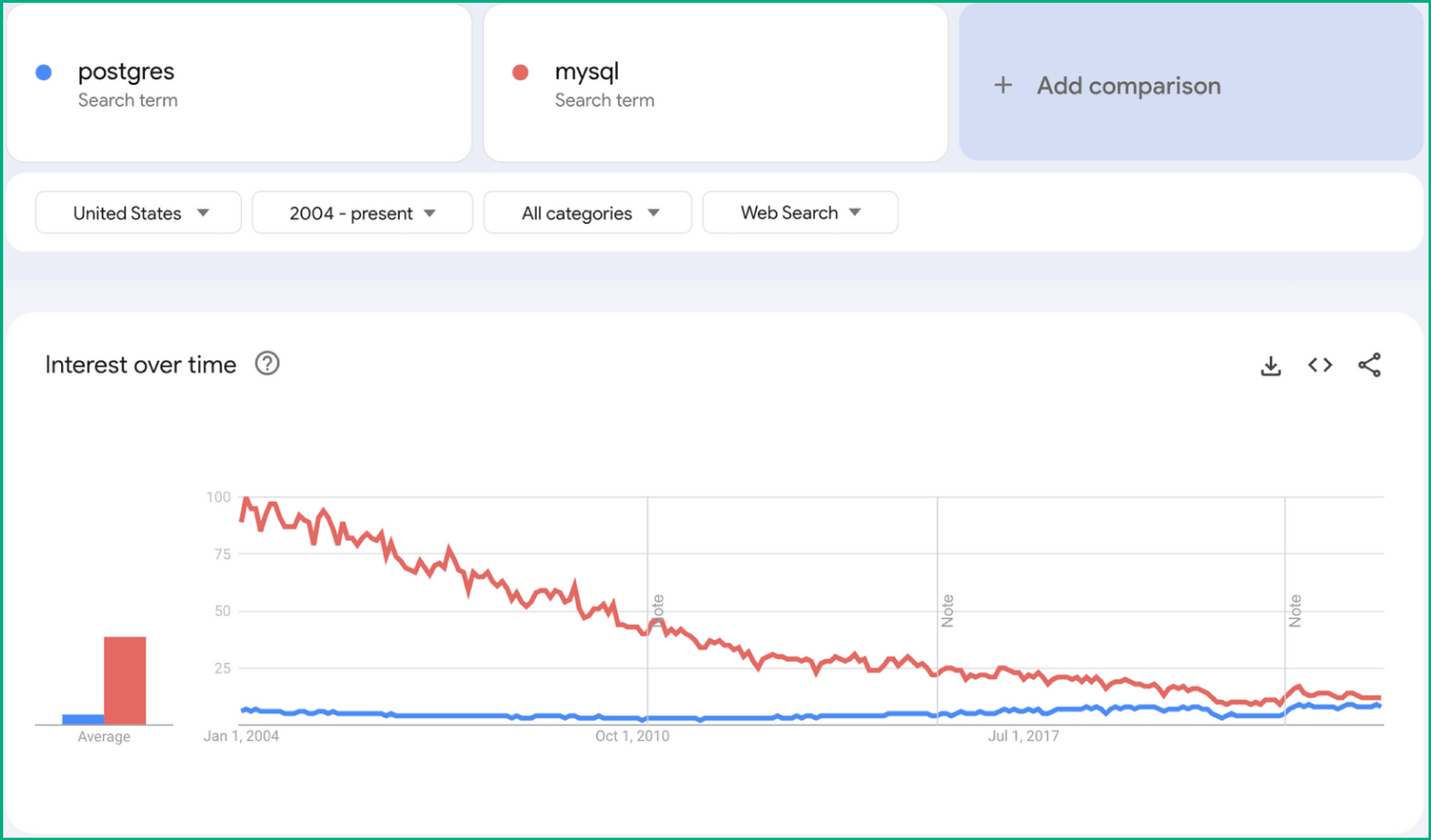


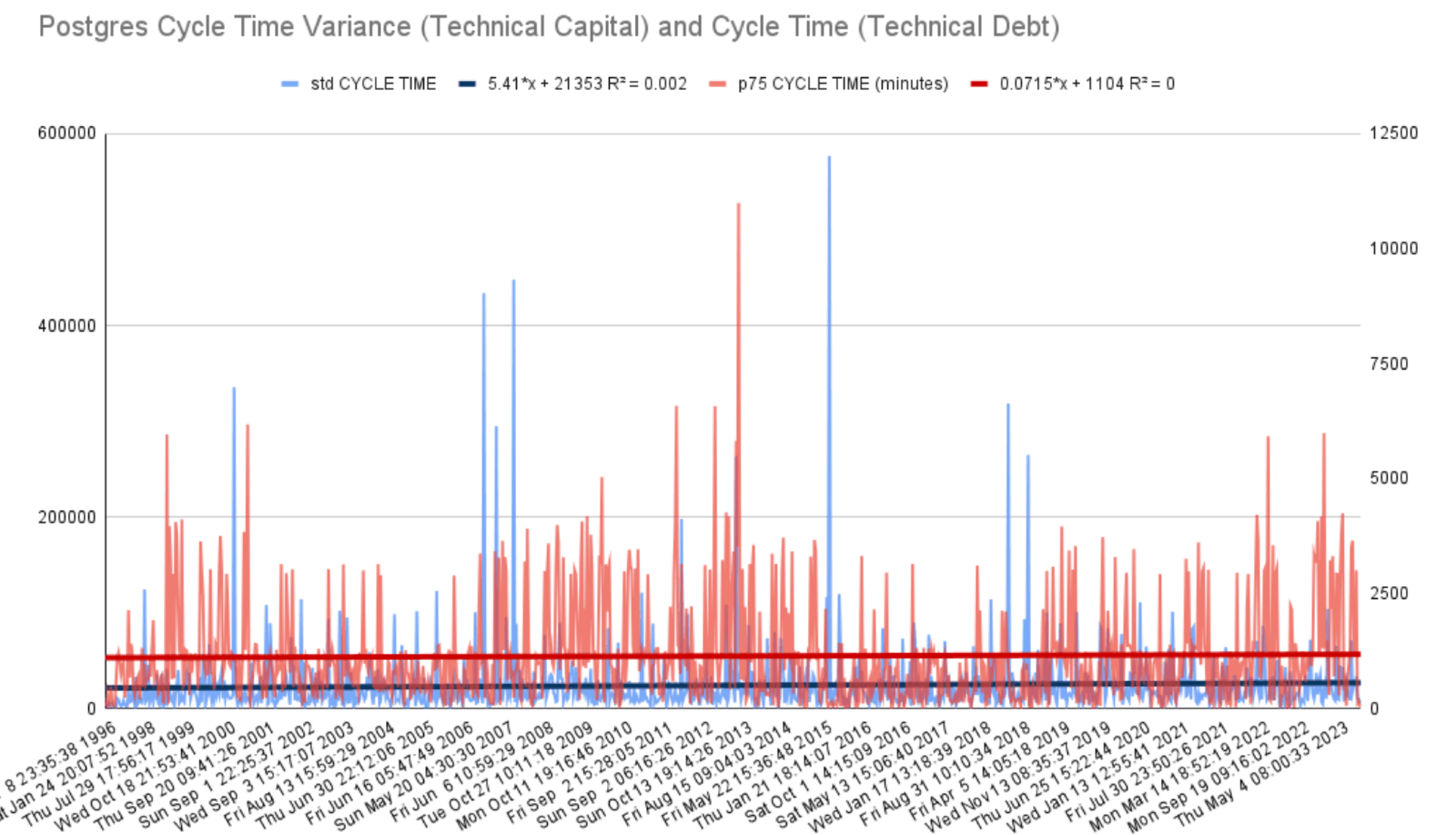
PyTorch vs. Tensorflow
PyTorch
-22.1*x + 94673 R² = 0.006
Tensorflow
73.2*x + 43946 R² = 0.101
600000
400000
200000
0
PyTorch
Tensorflow

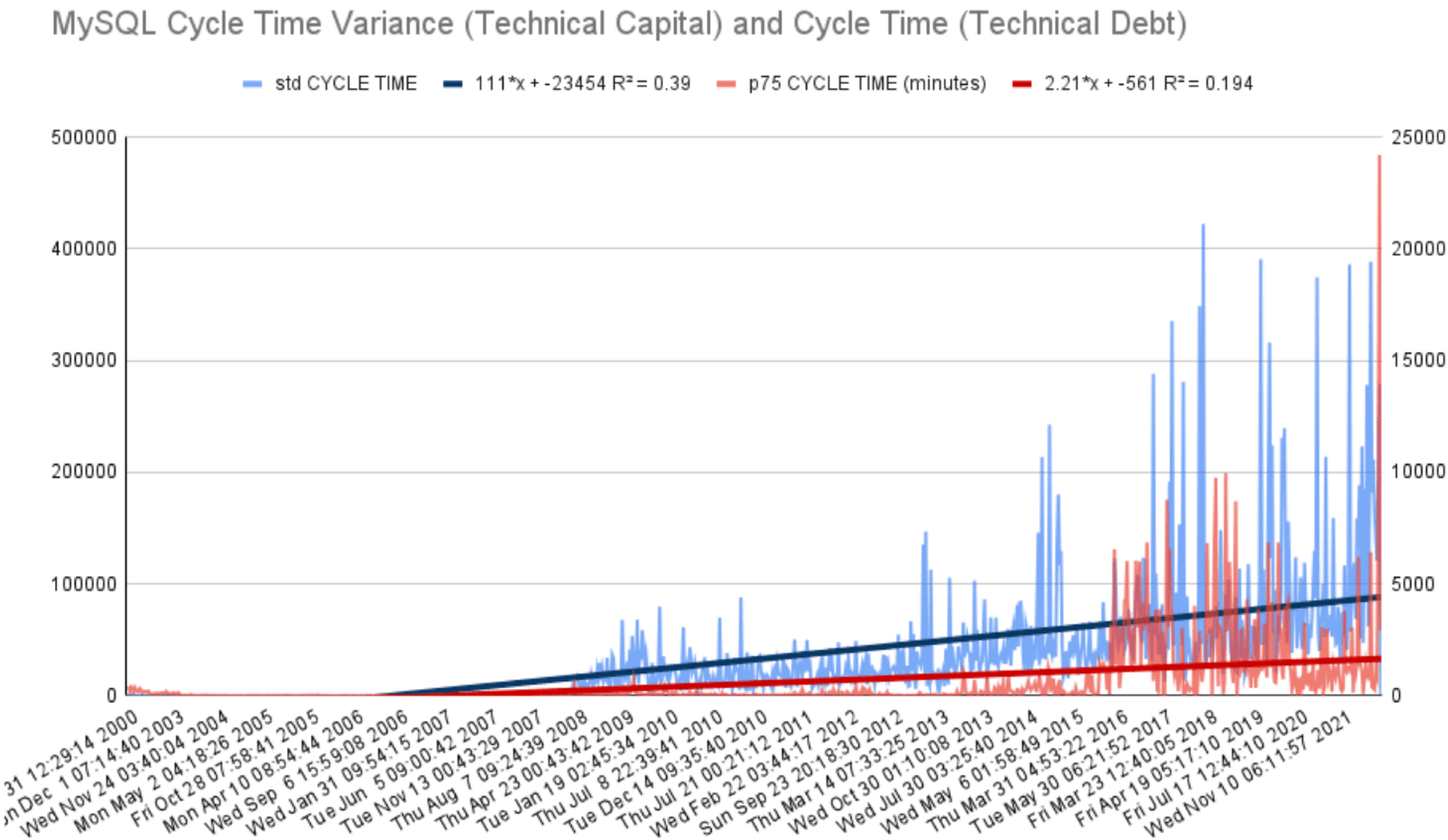
The interest (proxy for value) shift away from Tensorflow toward PyTorch occurred at about the same time the standard deviation cycle time for the two projects crossed.

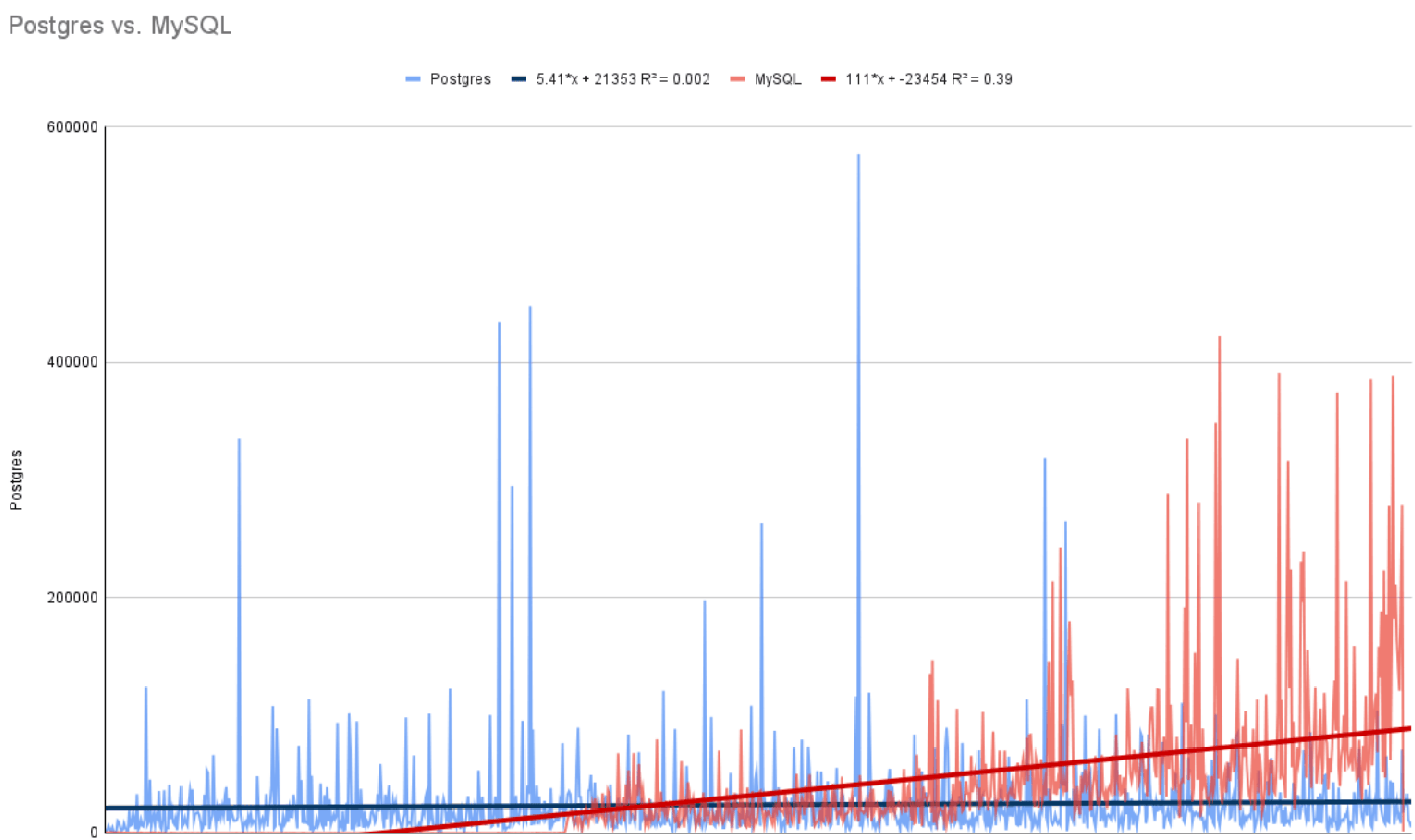
# Databases

MySQL | Postgres

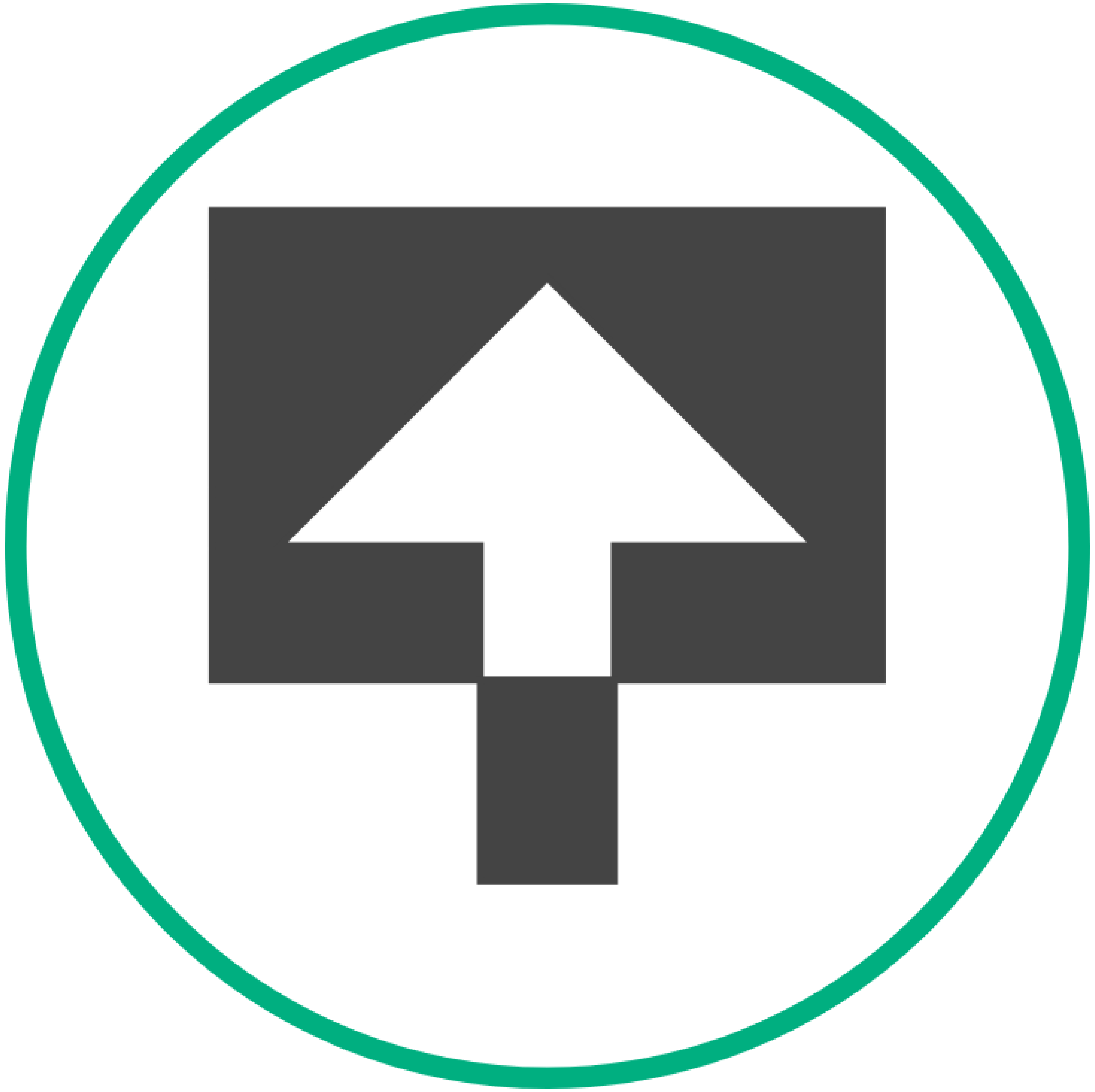

postgres
Search term
mysql
Search term
+ Add comparison
United States
2004 - present
All categories
Web Search
Interest over time
100
75
50
25
Average
Jan 1, 2004
Oct 1, 2010
Jul 1, 2017
Note
Note
Note


Postgres Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
5.41*x + 21353 R² = 0.002
p75 CYCLE TIME (minutes)
0.0715*x + 1104 R² = 0
600000
400000
200000
0
12500
10000
7500
5000
2500
0
8 23:35:38 1996
Jan 24 20:07:52 1998
Thu Jul 29 17:56:17 1999
Wed Oct 18 21:53:41 2000
Thu Sep 20 09:41:26 2001
Sun Sep 1 22:25:37 2002
Wed Sep 3 15:17:07 2003
Fri Aug 13 15:59:29 2004
Thu Jun 30 22:12:06 2005
Fri Jun 16 05:47:49 2006
Sun May 20 04:30:30 2007
Fri Jun 6 10:59:29 2008
Tue Oct 27 10:11:18 2009
Mon Oct 11 19:16:46 2010
Fri Sep 2 15:28:05 2011
Sun Sep 2 06:16:26 2012
Sun Oct 13 19:14:26 2013
Fri Aug 15 09:04:03 2014
Fri May 22 15:36:48 2015
Thu Jan 21 18:14:07 2016
Sat Oct 1 14:15:09 2016
Sat May 13 15:06:40 2017
Wed Jan 17 13:18:39 2018
Fri Aug 31 10:10:34 2018
Fri Apr 5 14:05:18 2019
Wed Nov 13 08:35:37 2019
Thu Jun 25 15:22:44 2020
Wed Jan 13 12:55:41 2021
Fri Jul 30 23:50:26 2021
Mon Mar 14 18:52:19 2022
Mon Sep 19 09:16:02 2022
Thu May 4 08:00:33 2023


MySQL Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
111*x + -23454 R² = 0.39
p75 CYCLE TIME (minutes)
2.21*x + -561 R² = 0.194
500000
400000
300000
200000
100000
0
25000
20000
15000
10000
5000
0
31 12:29:14 2000
Dec 1 07:14:40 2003
Wed Nov 24 03:40:04 2004
Mon May 2 04:18:26 2005
Fri Oct 28 07:58:41 2005
Mon Apr 10 08:54:44 2006
Wed Sep 6 15:59:08 2006
Wed Jan 31 09:54:15 2007
Tue Jun 5 09:00:42 2007
Tue Nov 13 00:43:29 2007
Thu Aug 7 09:24:39 2008
Thu Apr 23 00:43:42 2009
Tue Jan 19 02:45:34 2010
Thu Jul 8 22:39:41 2010
Tue Dec 14 09:35:40 2010
Thu Jul 21 00:21:12 2011
Wed Feb 22 03:44:17 2012
Sun Sep 23 20:18:30 2012
Thu Mar 14 07:33:25 2013
Wed Oct 30 01:10:08 2013
Wed Jul 30 03:25:40 2014
Wed May 6 01:58:49 2015
Thu Mar 31 04:53:22 2016
Tue May 30 06:21:52 2017
Fri Mar 23 12:40:05 2018
Fri Apr 19 05:17:10 2019
Fri Jul 17 12:44:10 2020
Wed Nov 10 06:11:57 2021



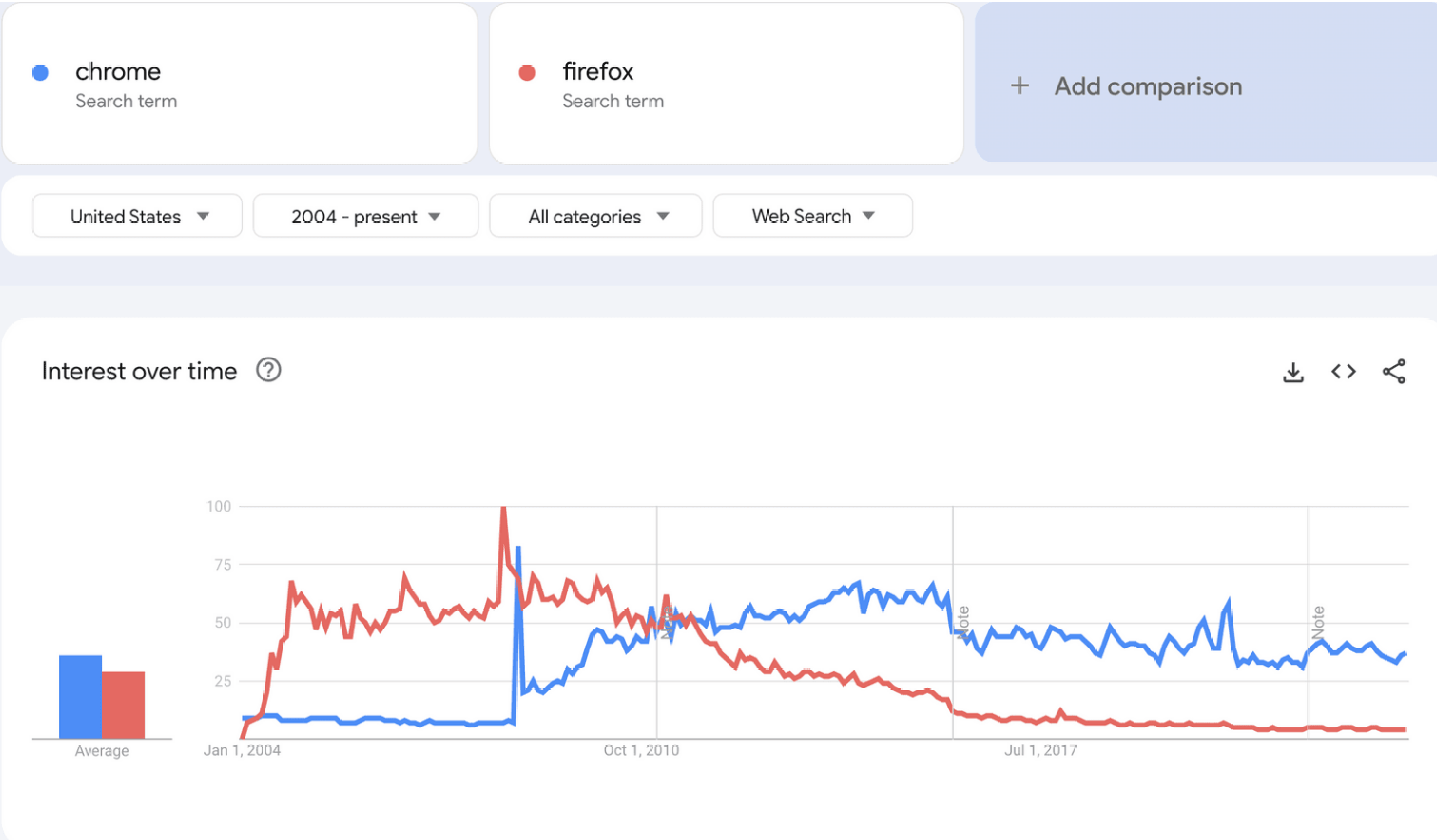
Postgres vs. MySQL
Postgres
5.41*x + 21353 R² = 0.002
MySQL
111*x + -23454 R² = 0.39
600000
400000
200000
0
Postgres

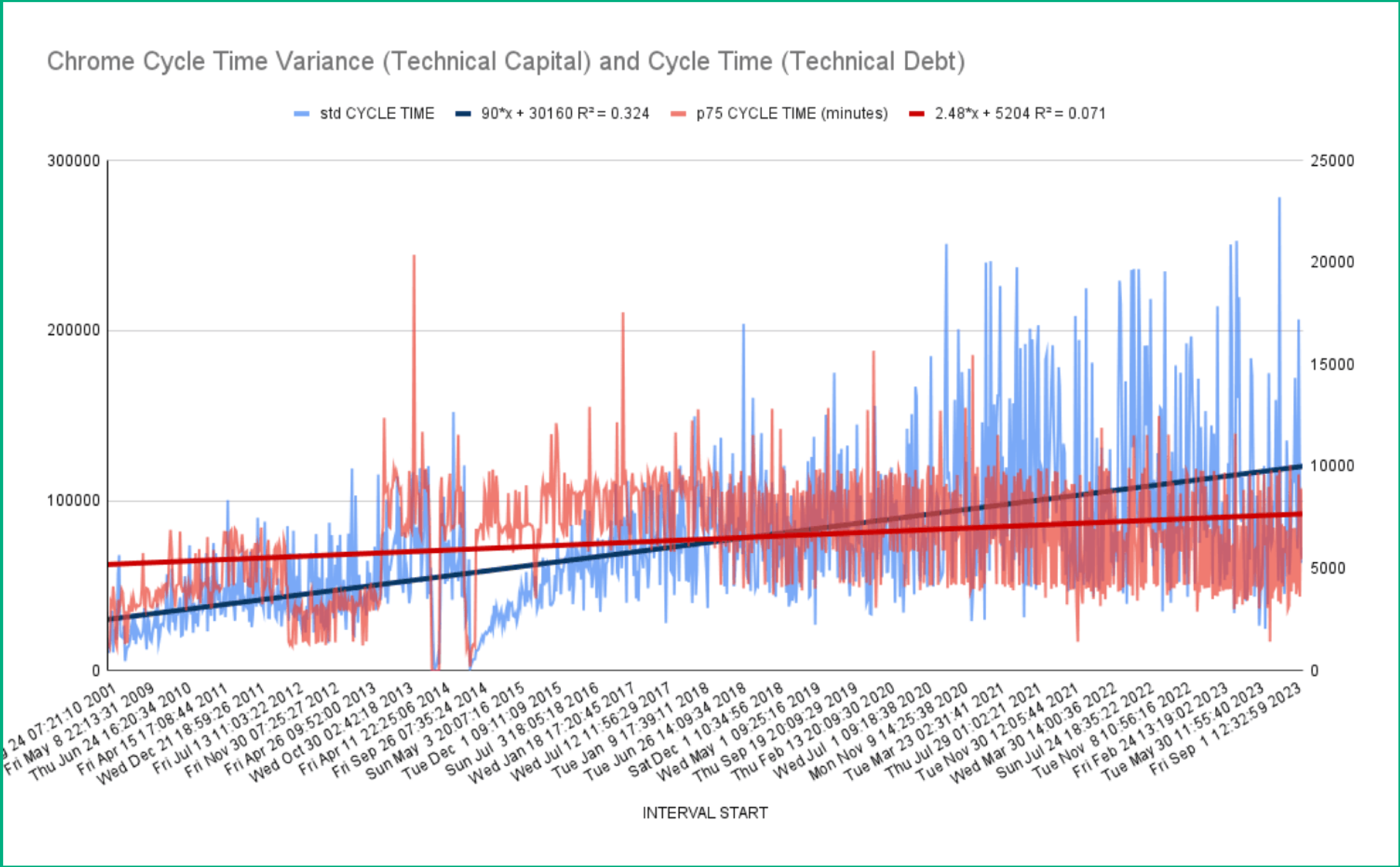


MySQL's drop in developer interest (proxy for value) appears to be the inverse of their rise in cycle time variance.

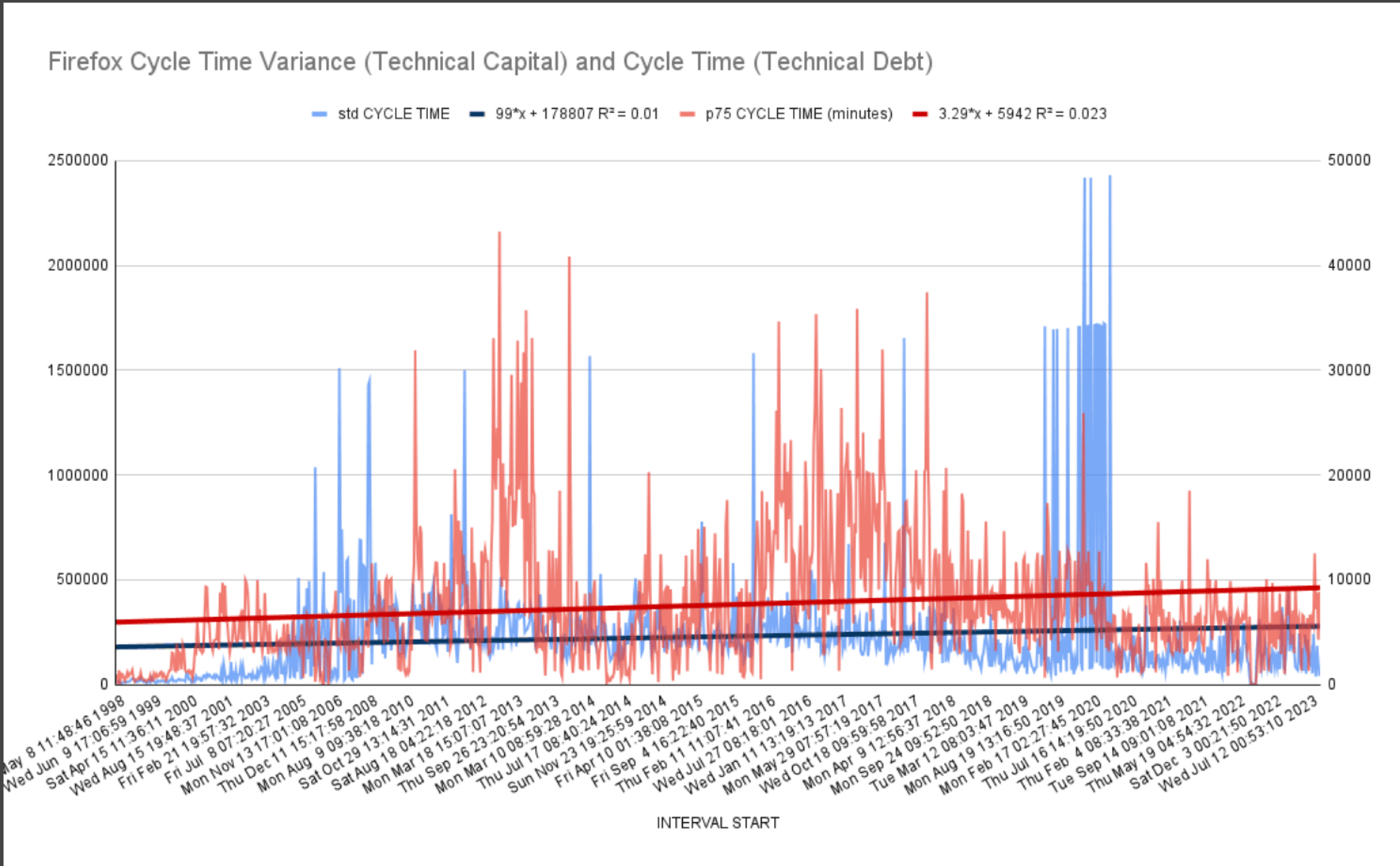
# Browsers

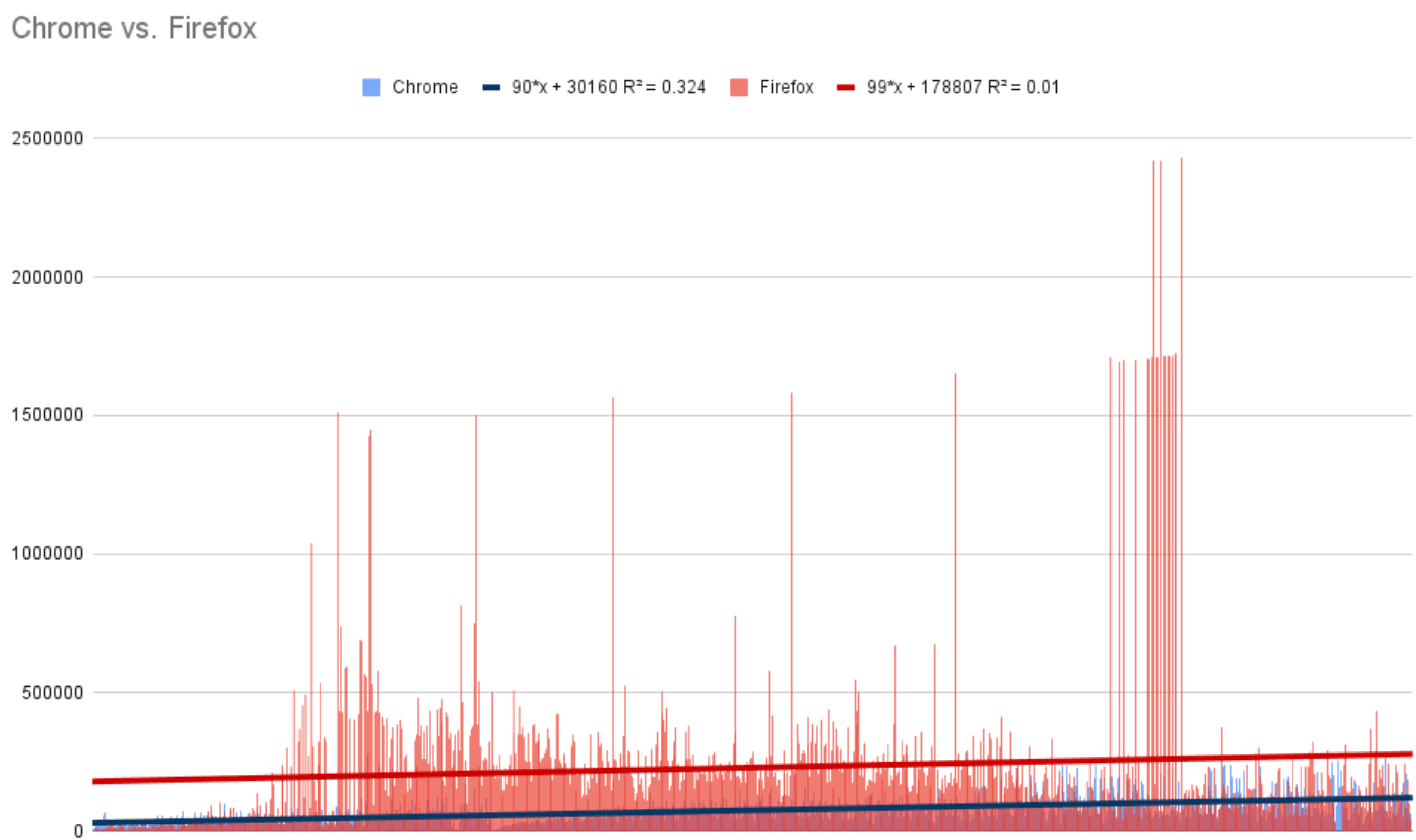
# Firefox | Chrome


chrome
Search term
firefox
Search term
+ Add comparison
United States
2004 - present
All categories
Web Search
Interest over time
100
75
50
25
Average
Jan 1, 2004
Oct 1, 2010
Jul 1, 2017
Note
Note
Note



Chrome Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
90*x + 30160 R² = 0.324
p75 CYCLE TIME (minutes)
2.48*x + 5204 R² = 0.071
300000
200000
100000
0
25000
20000
15000
10000
5000
0
INTERVAL START


Firefox Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
99*x + 178807 R² = 0.01
p75 CYCLE TIME (minutes)
3.29*x + 5942 R² = 0.023
2500000
2000000
1500000
1000000
500000
0
50000
40000
30000
20000
10000
0
ay 8 11:48:46 1998
Wed Jun 9 17:06:59 1999
Sat Apr 15 11:36:11 2000
Wed Aug 15 19:48:37 2001
Fri Feb 21 19:57:32 2003
Fri Jul 8 07:20:27 2005
Mon Nov 13 17:01:08 2006
Thu Dec 11 15:17:58 2008
Mon Aug 9 09:38:18 2010
Sat Oct 29 13:14:31 2011
Sat Aug 18 04:22:18 2012
Mon Mar 18 15:07:07 2013
Thu Sep 26 23:20:54 2013
Mon Mar 10 08:59:28 2014
Thu Jul 17 08:40:24 2014
Sun Nov 23 19:25:59 2014
Fri Apr 10 01:38:08 2015
Fri Sep 4 16:22:40 2015
Thu Feb 11 11:07:41 2016
Wed Jul 27 08:18:01 2016
Wed Jan 11 13:19:13 2017
Mon May 29 07:57:19 2017
Wed Oct 18 09:59:58 2017
Mon Apr 9 12:56:37 2018
Mon Sep 24 09:52:50 2018
Tue Mar 12 08:03:47 2019
Mon Aug 19 13:16:50 2019
Mon Feb 17 02:27:45 2020
Thu Jul 16 14:19:50 2020
Thu Feb 4 08:33:38 2021
Tue Sep 14 09:01:08 2021
Thu May 19 04:54:32 2022
Sat Dec 3 00:21:50 2022
Wed Jul 12 00:53:10 2023
INTERVAL START

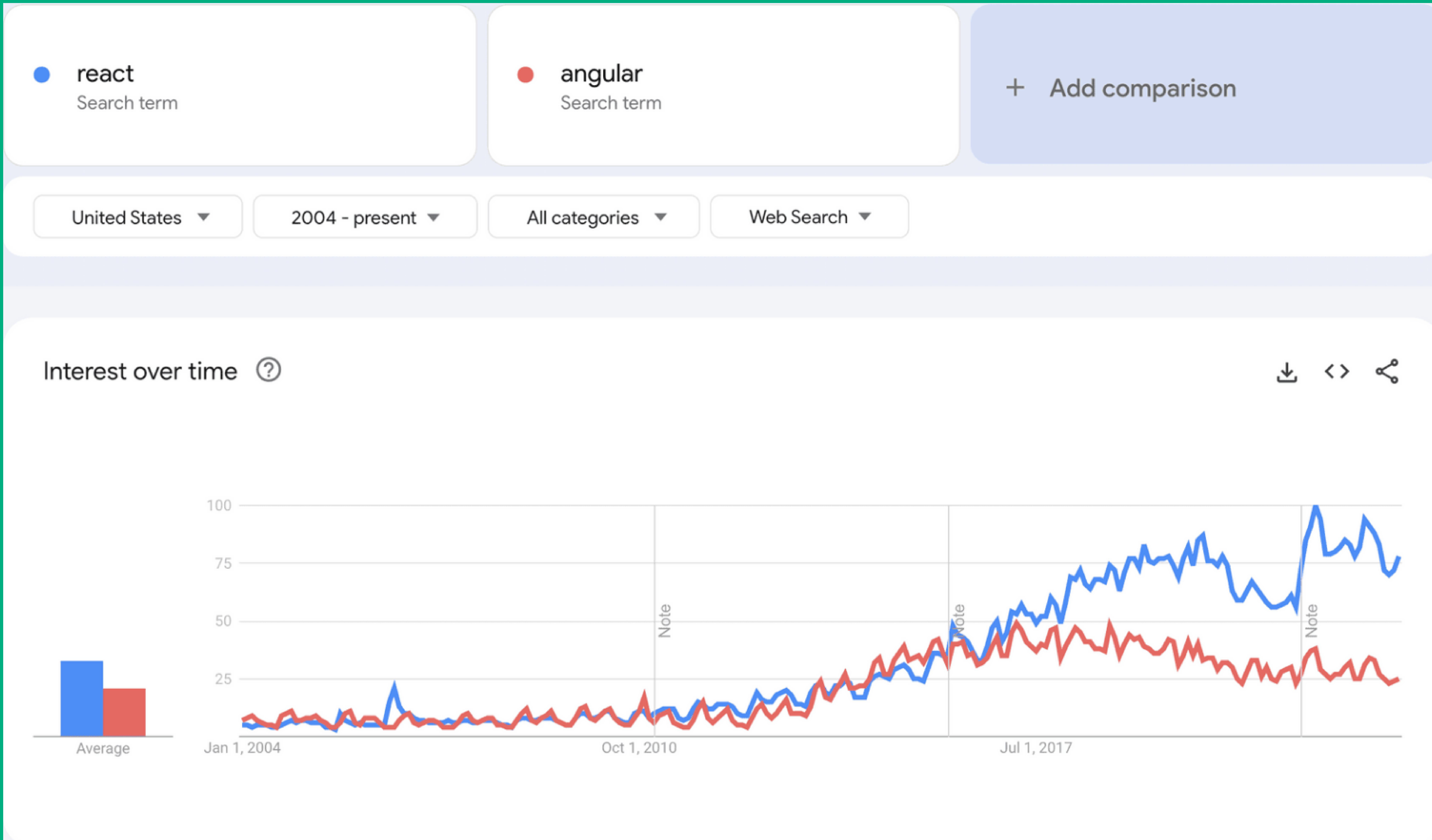


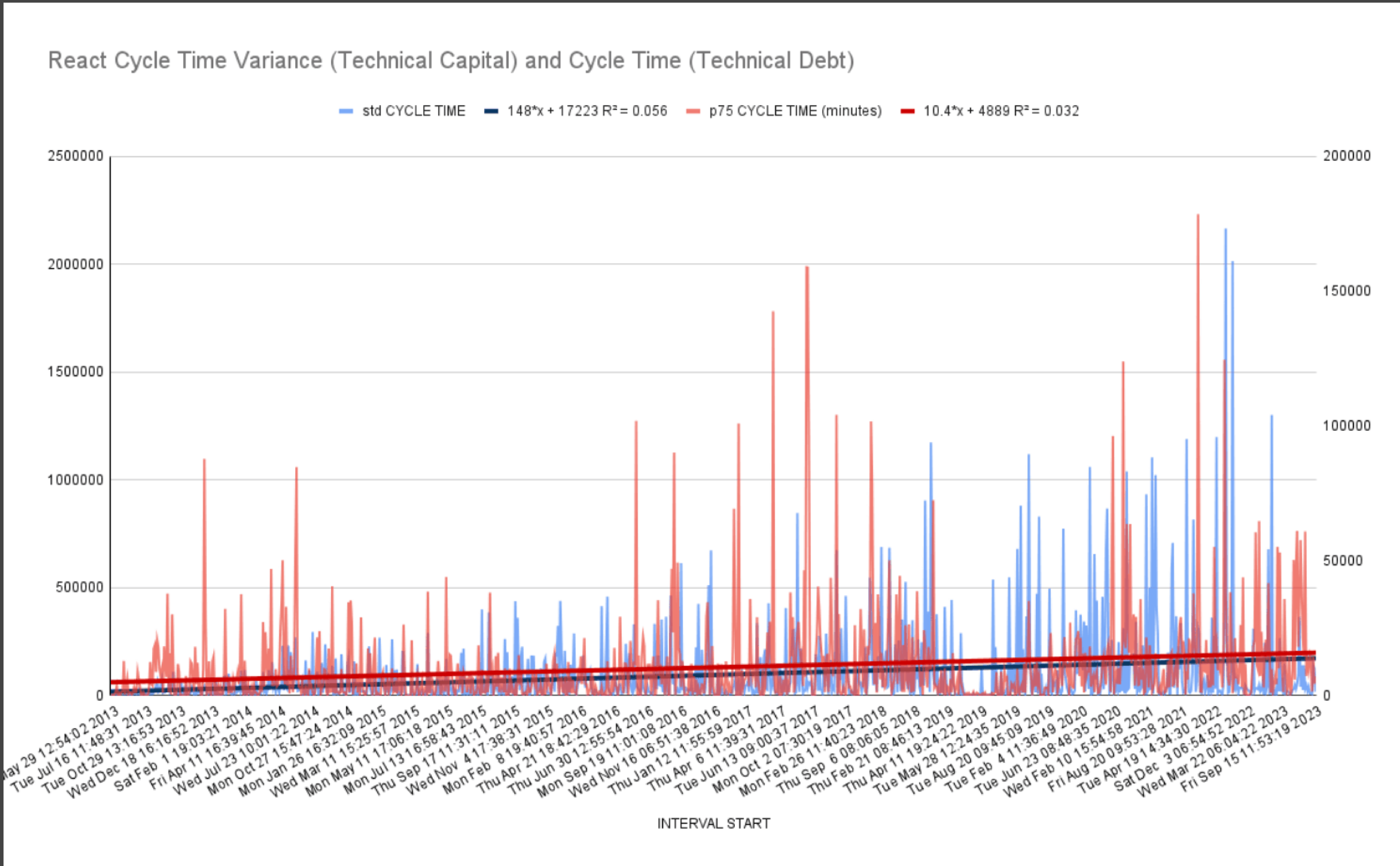
Chrome vs. Firefox
Chrome
90*x + 30160 R² = 0.324
Firefox
99*x + 178807 R² = 0.01
2500000
2000000
1500000
1000000
500000
0

The rate of increase of cycle time variance for Chrome and Firefox are similar, though Chrome is slightly better at a factor of 90 whereas Firefox shows a factor of 99. Chrome is also significantly below Firefox in terms of variance. This is expected given their position of dominance in developer interest (proxy for value).

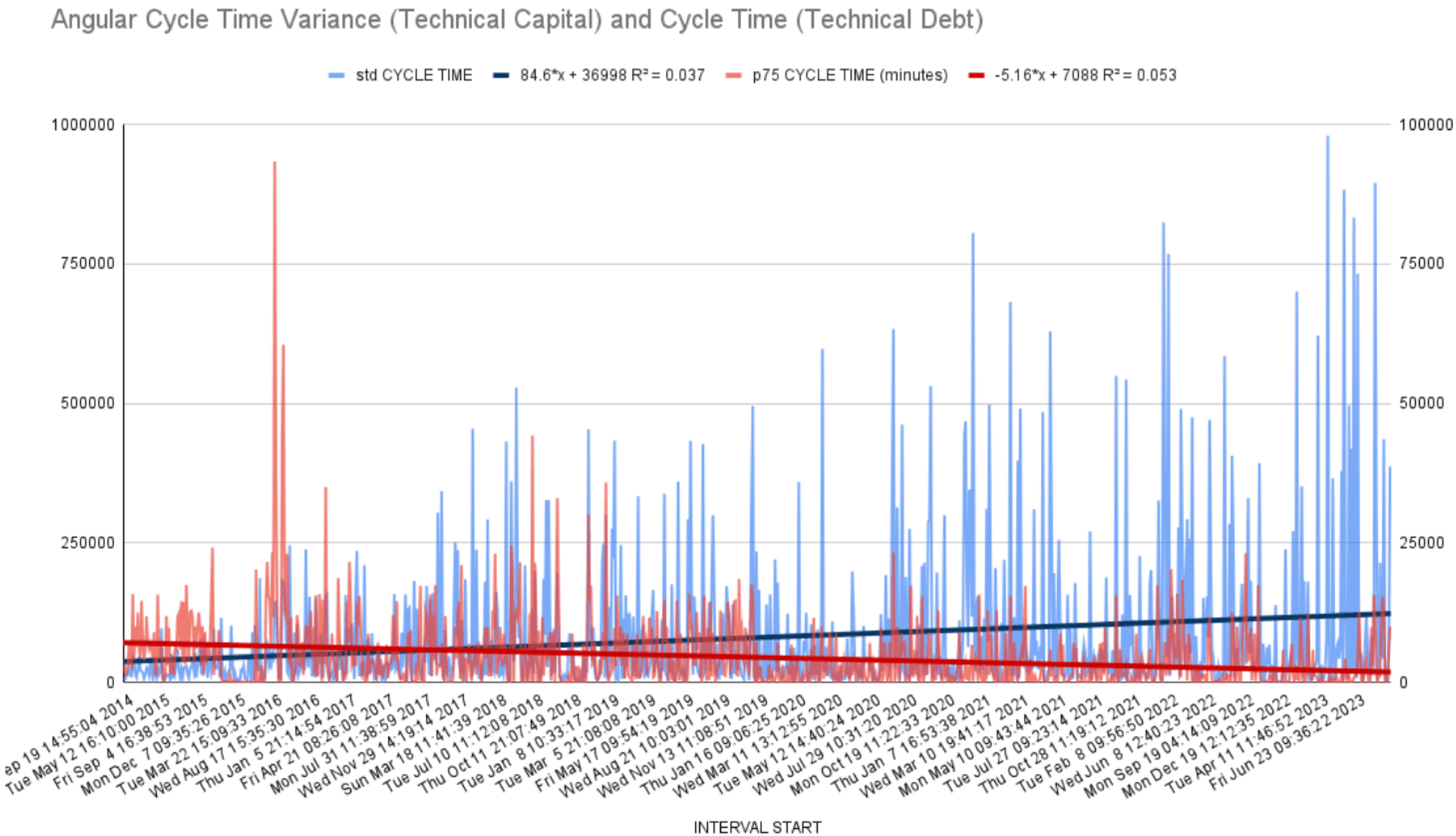
# Frontend Frameworks

Angular | React

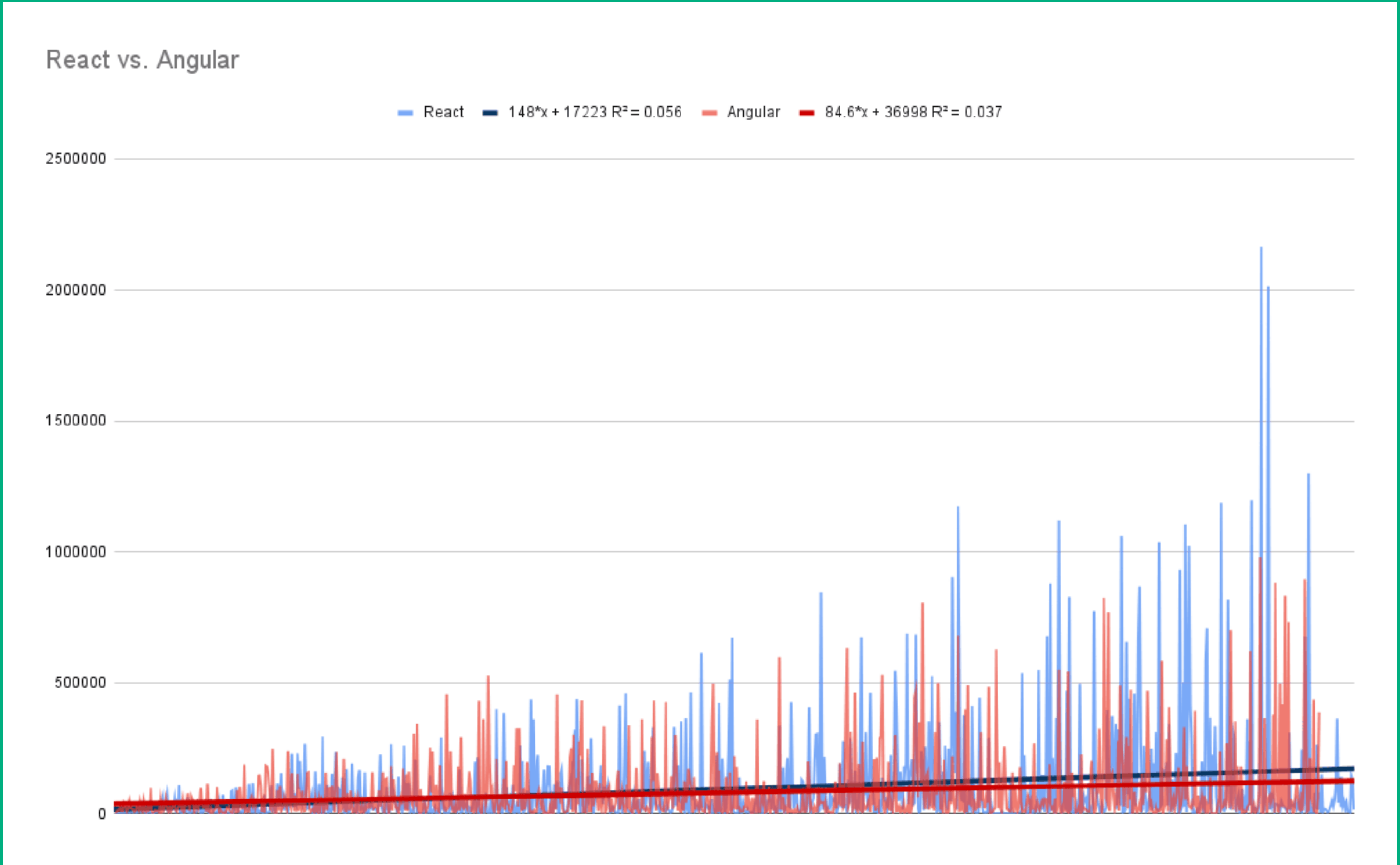

react
Search term
angular
Search term
+ Add comparison
United States
2004 - present
All categories
Web Search
Interest over time
100
75
50
25
Note
Note
Note
Average
Jan 1, 2004
Oct 1, 2010
Jul 1, 2017



React Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
148*x + 17223 R² = 0.056
p75 CYCLE TIME (minutes)
10.4*x + 4889 R² = 0.032
2500000
2000000
1500000
1000000
500000
0
200000
150000
100000
50000
0
Tue Jul 16 11:48:31 2013
Tue Oct 29 13:16:53 2013
Wed Dec 18 16:16:52 2013
Sat Feb 1 19:03:21 2014
Fri Apr 11 16:39:45 2014
Wed Jul 23 10:01:22 2014
Mon Oct 27 15:47:24 2014
Mon Jan 26 16:32:09 2015
Wed Mar 11 15:25:57 2015
Mon May 11 17:06:18 2015
Mon Jul 13 16:58:43 2015
Thu Sep 17 11:31:11 2015
Wed Nov 4 17:38:31 2015
Mon Feb 8 19:40:57 2016
Thu Apr 21 18:42:29 2016
Thu Jun 30 12:55:54 2016
Mon Sep 19 11:01:08 2016
Wed Nov 16 06:51:38 2016
Thu Jan 12 11:55:59 2017
Thu Apr 6 11:39:31 2017
Tue Jun 13 09:00:37 2017
Mon Oct 2 07:30:19 2017
Mon Feb 26 11:40:23 2018
Thu Sep 6 08:06:05 2018
Thu Feb 21 08:46:13 2019
Thu Apr 11 19:24:22 2019
Tue May 28 12:24:35 2019
Tue Aug 20 09:45:09 2019
Tue Feb 4 11:36:49 2020
Tue Jun 23 08:48:35 2020
Wed Feb 10 15:54:58 2021
Fri Aug 20 09:53:28 2021
Tue Apr 19 14:34:30 2022
Sat Dec 3 06:54:52 2022
Wed Mar 22 06:04:22 2023
Fri Sep 15 11:53:19 2023
INTERVAL START



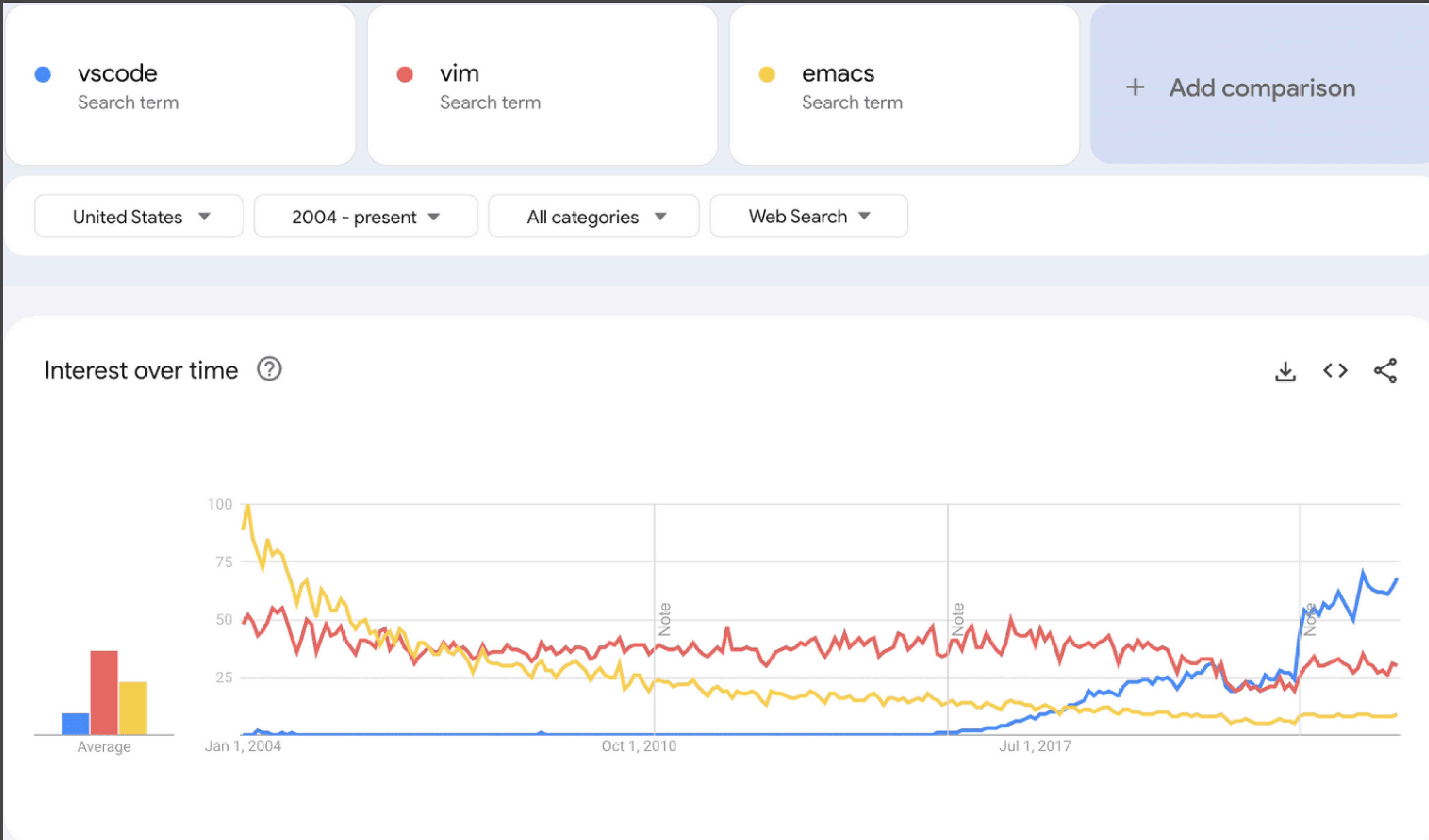
Angular Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
84.6*x + 36998 R² = 0.037
p75 CYCLE TIME (minutes)
-5.16*x + 7088 R² = 0.053
1000000
750000
500000
250000
0
100000
75000
50000
25000
0
Tue May 12 16:10:00 2015
Fri Sep 4 16:38:53 2015
Mon Dec 7 09:35:26 2015
Tue Mar 22 15:09:33 2016
Wed Aug 17 15:35:30 2016
Thu Jan 5 21:14:54 2017
Fri Apr 21 08:26:08 2017
Mon Jul 31 11:38:59 2017
Wed Nov 29 14:19:14 2017
Sun Mar 18 11:41:39 2018
Tue Jul 10 11:12:08 2018
Thu Oct 11 21:07:49 2018
Tue Jan 8 10:33:17 2019
Tue Mar 5 21:08:08 2019
Fri May 17 09:54:19 2019
Wed Aug 21 10:03:01 2019
Wed Nov 13 11:08:51 2019
Thu Jan 16 09:06:25 2020
Wed Mar 11 13:12:55 2020
Tue May 12 14:40:24 2020
Wed Jul 29 10:31:20 2020
Mon Oct 19 11:22:33 2020
Thu Jan 7 16:53:38 2021
Wed Mar 10 19:41:17 2021
Mon May 10 09:43:44 2021
Tue Jul 27 09:23:14 2021
Thu Oct 28 11:19:12 2021
Tue Feb 8 09:56:50 2022
Wed Jun 8 12:40:23 2022
Mon Sep 19 04:14:09 2022
Mon Dec 19 12:12:35 2022
Tue Apr 11 11:46:52 2023
Fri Jun 23 09:36:22 2023
INTERVAL START

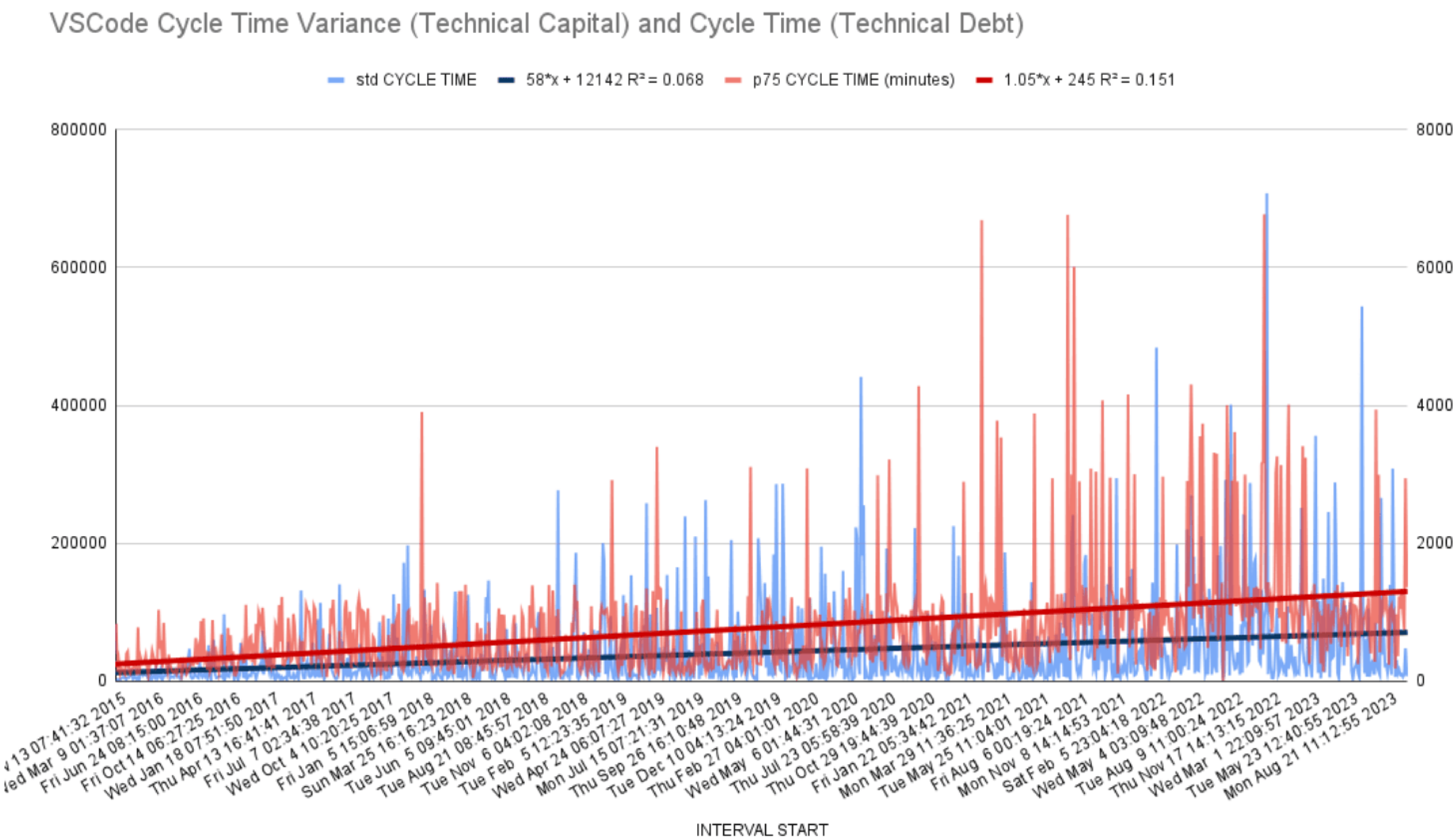

React vs. Angular
React
148*x + 17223 R² = 0.056
Angular
84.6*x + 36998 R² = 0.037
2500000
2000000
1500000
1000000
500000
0

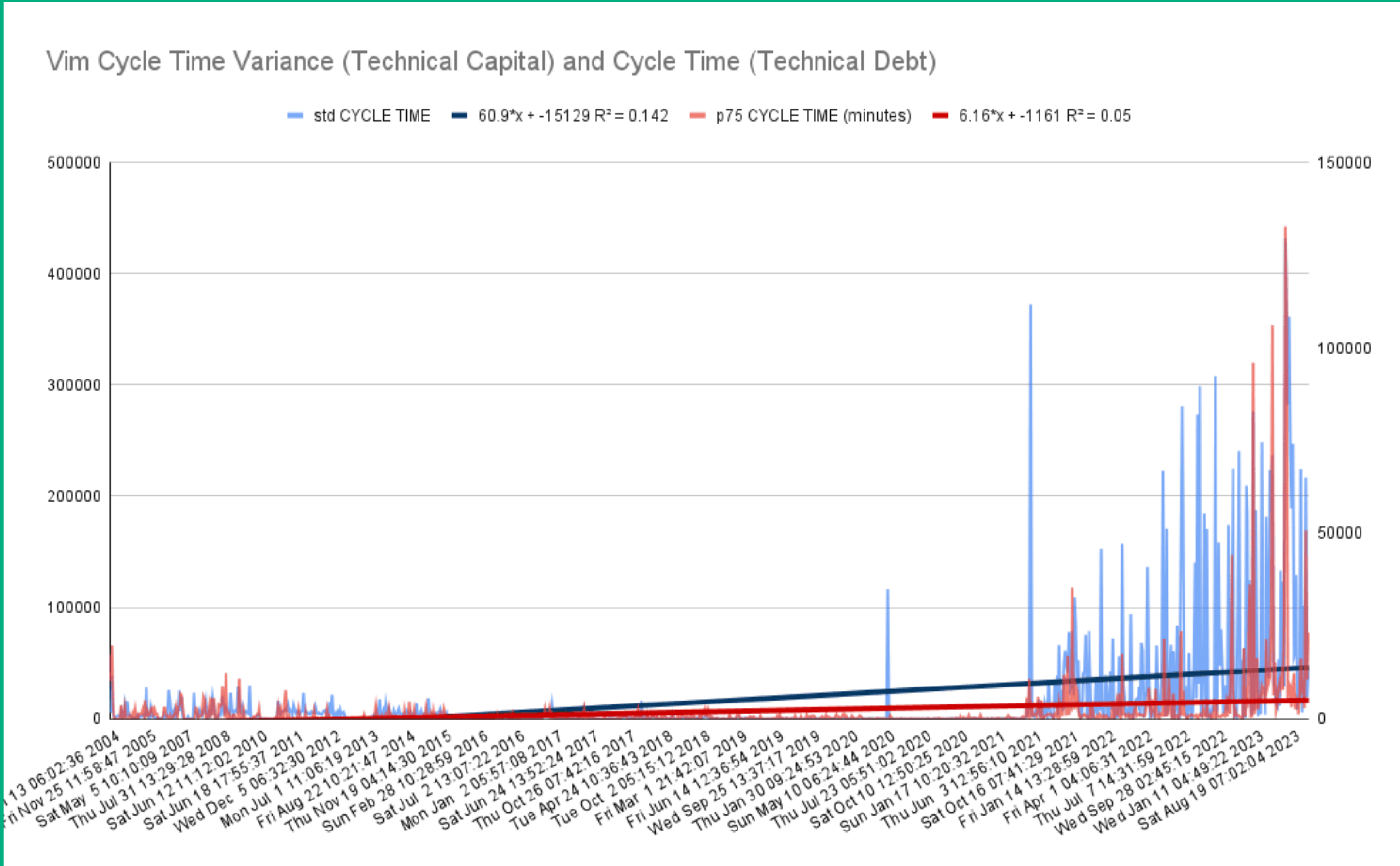
Both Angular and React are moving in the wrong direction with little distinction. There is room for a third player.

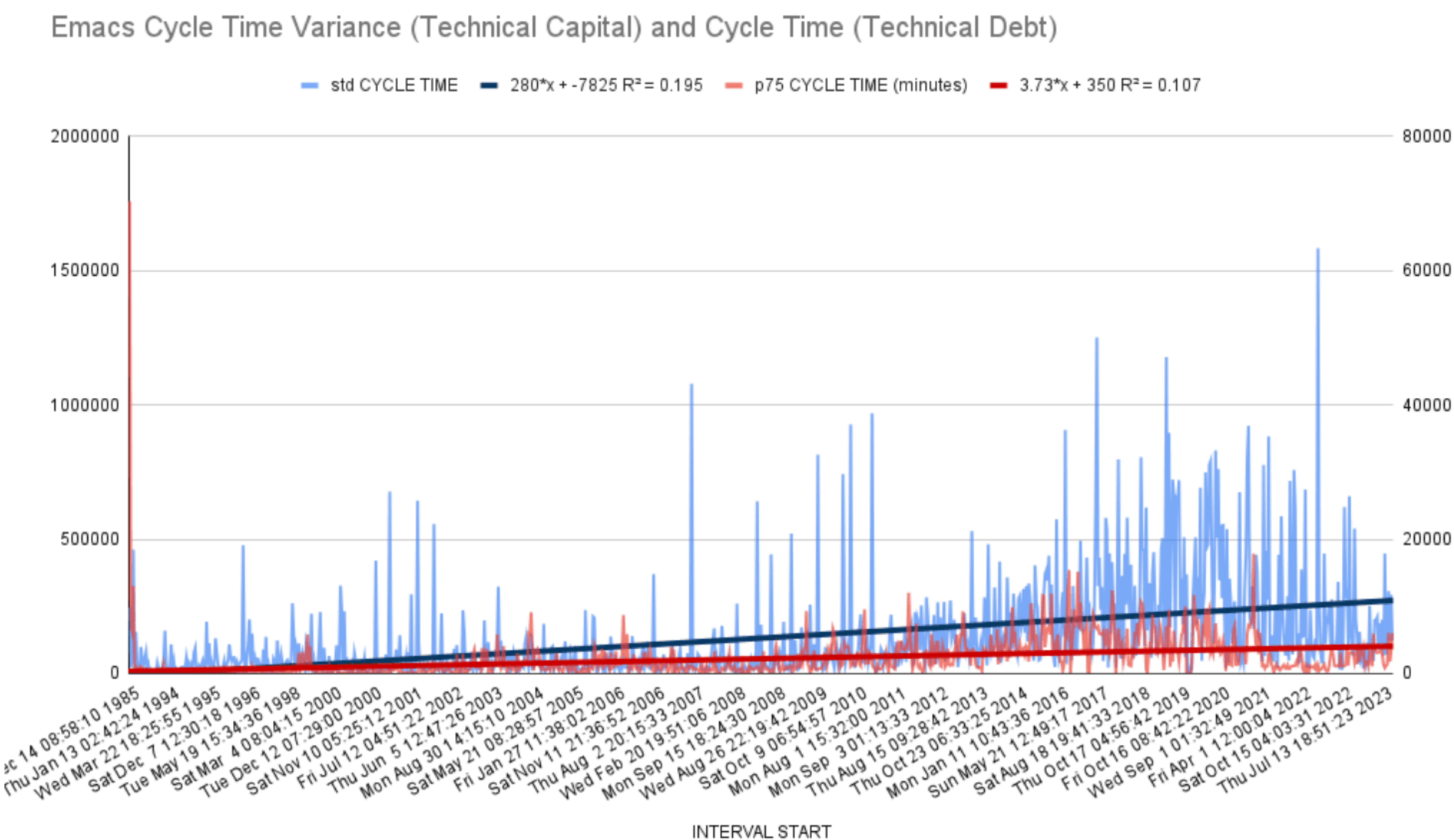
# Code Editors

Emacs | Vim | VSCode

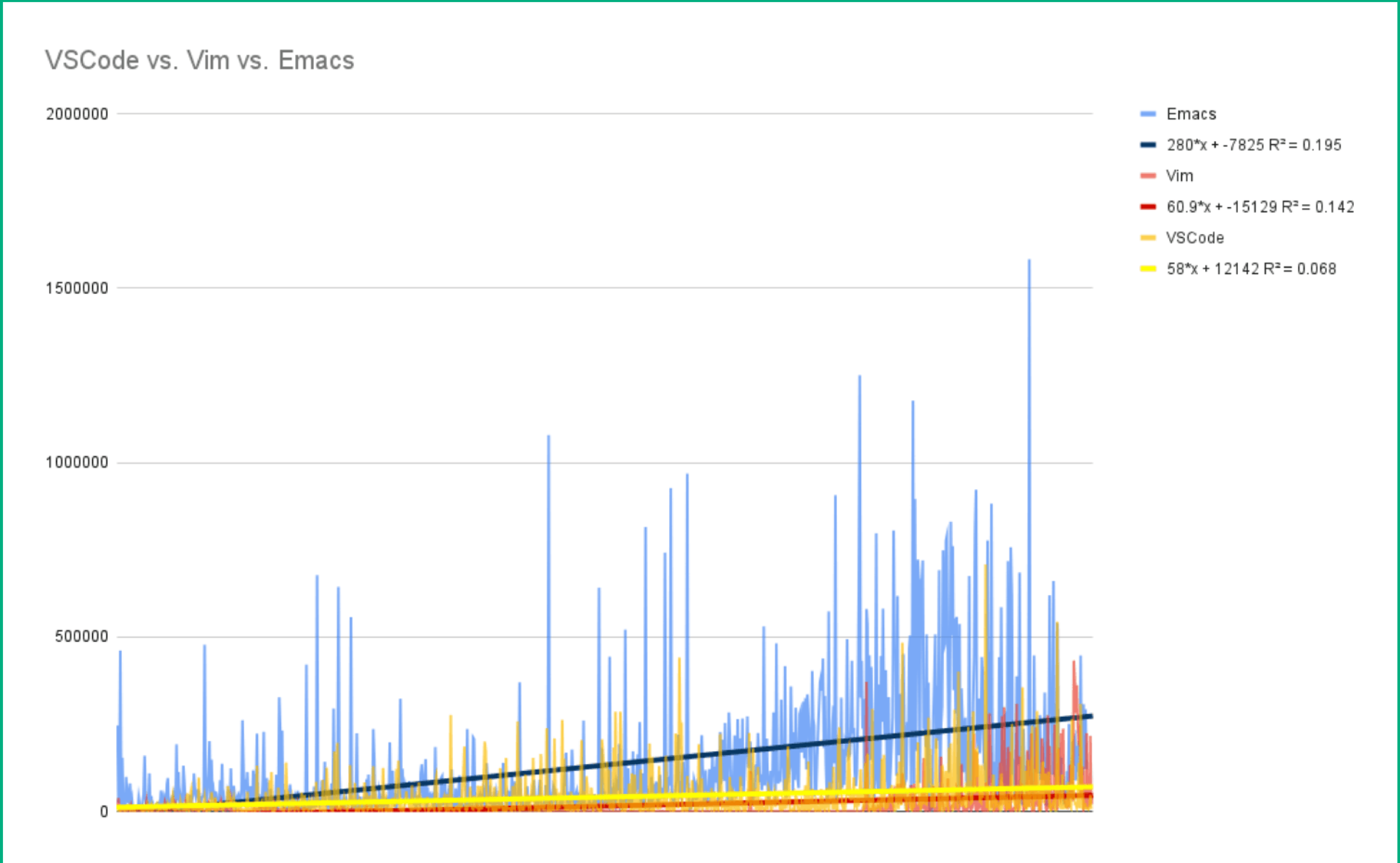

vscode
Search term
vim
Search term
emacs
Search term
+ Add comparison
United States
2004 - present
All categories
Web Search
Interest over time
100
75
50
25
Note
Note
Note
Average
Jan 1, 2004
Oct 1, 2010
Jul 1, 2017


VSCode Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
58*x + 12142 R² = 0.068
p75 CYCLE TIME (minutes)
1.05*x + 245 R² = 0.151
800000
600000
400000
200000
0
8000
6000
4000
2000
0
Fri Jun 24 08:15:00 2016
Fri Oct 14 06:27:25 2016
Wed Jan 18 07:51:50 2017
Thu Apr 13 16:41:41 2017
Fri Jul 7 02:34:38 2017
Wed Oct 4 10:20:25 2017
Fri Jan 5 15:06:59 2018
Sun Mar 25 16:16:23 2018
Tue Jun 5 09:45:01 2018
Tue Aug 21 08:45:57 2018
Tue Nov 6 04:02:08 2018
Tue Feb 5 12:23:35 2019
Wed Apr 24 06:07:27 2019
Mon Jul 15 07:21:31 2019
Thu Sep 26 16:10:48 2019
Tue Dec 10 04:13:24 2019
Thu Feb 27 04:01:01 2020
Wed May 6 01:44:31 2020
Thu Jul 23 05:58:39 2020
Thu Oct 29 19:44:39 2020
Fri Jan 22 05:34:42 2021
Mon Mar 29 11:36:25 2021
Tue May 25 11:04:01 2021
Fri Aug 6 00:19:24 2021
Mon Nov 8 14:14:53 2021
Sat Feb 5 23:04:18 2022
Wed May 4 03:09:48 2022
Tue Aug 9 11:00:24 2022
Thu Nov 17 14:13:15 2022
Wed Mar 1 22:09:57 2023
Tue May 23 12:40:55 2023
Mon Aug 21 11:12:55 2023
INTERVAL START


Vim Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
60.9*x + -15129 R² = 0.142
p75 CYCLE TIME (minutes)
6.16*x + -1161 R² = 0.05
500000
400000
300000
200000
100000
0
150000
100000
50000
0
13 06:02:36 2004
Fri Nov 25 11:58:47 2005
Sat May 5 10:10:09 2007
Thu Jul 31 13:29:28 2008
Sat Jun 12 11:12:02 2010
Sat Jun 18 17:55:37 2011
Wed Dec 5 06:32:30 2012
Mon Jul 1 11:06:19 2013
Fri Aug 22 10:21:47 2014
Thu Nov 19 04:14:30 2015
Sun Feb 28 10:28:59 2016
Sat Jul 2 13:07:22 2016
Mon Jan 2 05:57:08 2017
Sat Jun 24 13:52:24 2017
Thu Oct 26 07:42:16 2017
Tue Apr 24 10:36:43 2018
Tue Oct 2 05:15:12 2018
Fri Mar 1 21:42:07 2019
Fri Jun 14 12:36:54 2019
Wed Sep 25 13:37:17 2019
Thu Jan 30 09:24:53 2020
Sun May 10 06:24:44 2020
Thu Jul 23 05:51:02 2020
Sat Oct 10 12:50:25 2020
Sun Jan 17 10:20:32 2021
Thu Jun 3 12:56:10 2021
Sat Oct 16 07:41:29 2021
Fri Jan 14 13:28:59 2022
Fri Apr 1 04:06:31 2022
Thu Jul 7 14:31:59 2022
Wed Sep 28 02:45:15 2022
Wed Jan 11 04:49:22 2023
Sat Aug 19 07:02:04 2023



Emacs Cycle Time Variance (Technical Capital) and Cycle Time (Technical Debt)
std CYCLE TIME
280*x + -7825 R² = 0.195
p75 CYCLE TIME (minutes)
3.73*x + 350 R² = 0.107
2000000
1500000
1000000
500000
0
80000
60000
40000
20000
0
ec 14 08:58:10 1985
Thu Jan 13 02:42:24 1994
Wed Mar 22 18:25:55 1995
Sat Dec 7 12:30:18 1996
Tue May 19 15:34:36 1998
Sat Mar 4 08:04:15 2000
Tue Dec 12 07:29:00 2000
Sat Nov 10 05:25:12 2001
Fri Jul 12 04:51:22 2002
Thu Jun 5 12:47:26 2003
Mon Aug 30 14:15:10 2004
Sat May 21 08:28:57 2005
Fri Jan 27 11:38:02 2006
Sat Nov 11 21:36:52 2006
Thu Aug 2 20:15:33 2007
Wed Feb 20 19:51:06 2008
Mon Sep 15 18:24:30 2008
Wed Aug 26 22:19:42 2009
Sat Oct 9 06:54:57 2010
Mon Aug 1 15:32:00 2011
Mon Sep 3 01:13:33 2012
Thu Aug 15 09:28:42 2013
Thu Oct 23 06:33:25 2014
Mon Jan 11 10:43:36 2016
Sun May 21 12:49:17 2017
Sat Aug 18 19:41:33 2018
Thu Oct 17 04:56:42 2019
Fri Oct 16 08:42:22 2020
Wed Sep 1 01:32:49 2021
Fri Apr 1 12:00:04 2022
Sat Oct 15 04:03:31 2022
Thu Jul 13 18:51:23 2023
INTERVAL START


VSCode vs. Vim vs. Emacs
2000000
1500000
1000000
500000
0
Emacs
280*x + -7825 R² = 0.195
Vim
60.9*x + -15129 R² = 0.142
VSCode
58*x + 12142 R² = 0.068

As you would expect, cycle time variance follows developer interest (proxy for value). The leader in the market, VSCode, has the lowest variance (trending to 58). The second most popular, Vim, has the second lowest variance (trending to 61). The least popular, Emacs, has the highest variance (280).

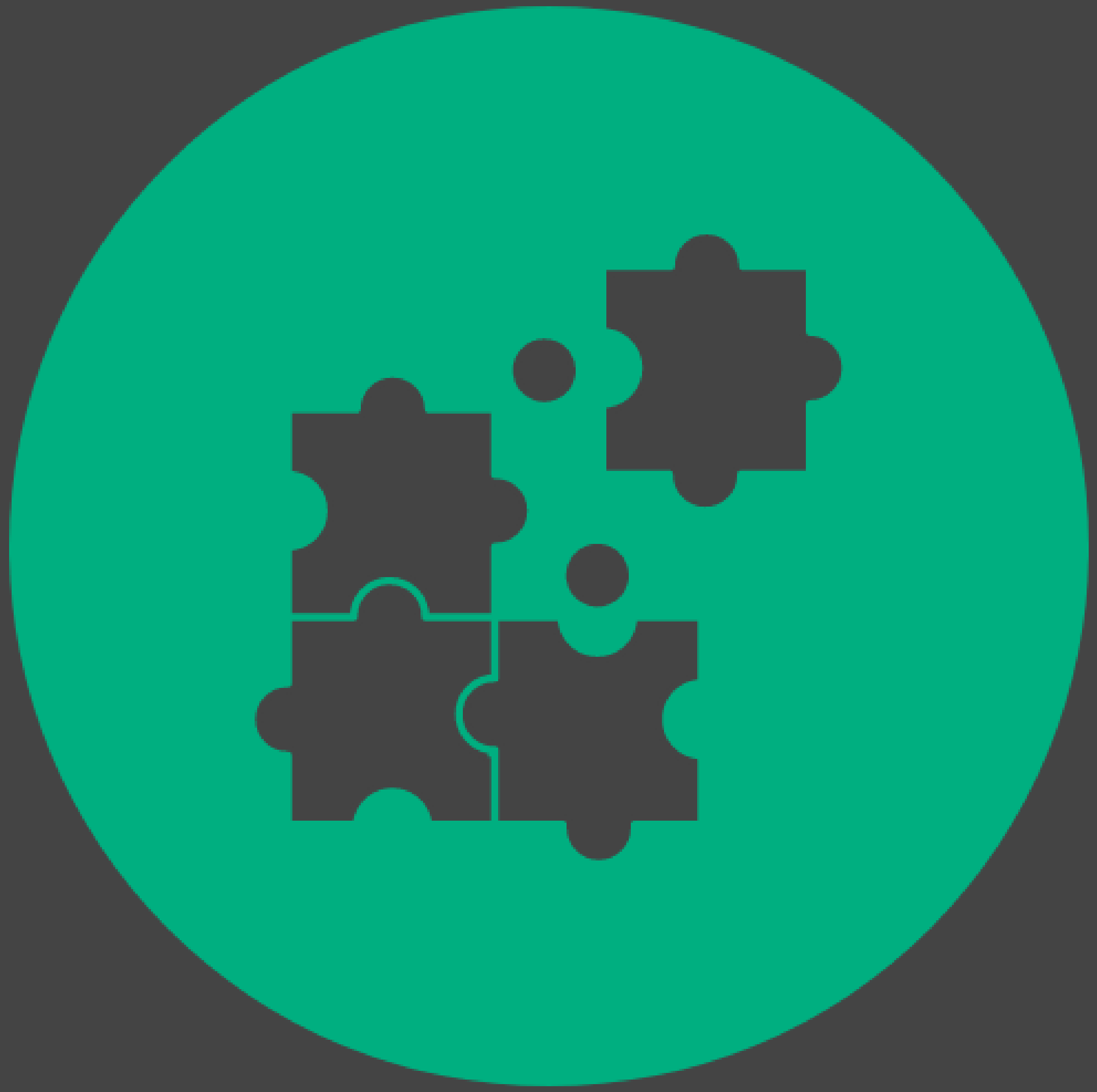
# Situation: Technology Company Valuation

Venture capitalists face challenges in valuing technology companies; the "unicorn" phenomenon, speculation and wild guesses

# Solution: Data-Driven Valuation with Cycle Time Variance

- Evaluate a technology company's growth potential by modeling its DORA metrics
- Beyond cycle time, look at cycle time variance
- Invest in the business with the lowest cycle time variance

# Situation: Technology Company Management



Operators face a range of problems daily: technical complexity, communication gaps, real-time visibility, measuring progress, scope changes and adaptability, balancing priorities, risk assessment, team dynamics, vendor and partner management, shift in supply chain and regulatory. Understanding how to best support and guide innovation efforts can be difficult.

# Solution: Data Driven Management with Cycle Time Variance

- Though the problem and solution space is complex and hard to understand, it is possible to measure the impact of challenges and success of management changes on technical work within the organization.
- Cycle time variance points to future value of the software, "Technical Capital"
- Work to minimize cycle time variance

# Conclusion

The application of measurement and reduction of variance offers a way for company operators to set their business up for increased future value and provides investors with a way to make data informed decisions. Technical debt can be measured with cycle time. Technical capital (a forecast of the future value of the code base) can be measured with cycle time variance.

# Thank you

Denali Lumma

denali@doubling.io

# Appendix

# Concerns with Data and Statistics

“There are three kinds of lies: Lies, Damned Lies, and Statistics.

Mark Twain

" Only ever trust statistics you have fabricated yourself

Winston Churchill

# Using Data in Science

# Experiment Methods

**Controlled**

Manipulate one or more independent variables while keeping other variables constant

**Natural**

Observe the variables of the system under study in different settings

**Observational**

Observe and gather data in one setting

# Experiment Methods

## Most rigorous to least

- Randomized Controlled Trials (RCTs)
- Quasi-Experiments
- Field Experiments
- Laboratory Experiments
- Natural Experiments
- Correlational Studies
- Case Studies
- Observational Studies
- Surveys
- Descriptive Studies

# Using Data in Finance

# Quarterly Financial Report

- Profit and Loss
- Balance Sheet
- Cash Flow
- Retained Earnings

# Each of the metrics included in the financial report has problems but by looking at all of them you get a comprehensive picture of the business

The questions to answer:

- (Investors) What is this business's potential value in the future?
- (Operators) What can be done to manage for better future business value?

Everyone agrees there is a lot of subtlety. But that doesn't stop people from trying, and it shouldn't.

# Using Data in Business

# Goal, Signal and Metric

**Goal**

What we want to achieve.

*(Resource Efficiency in Software Development)*

**Signal**

How we might know that we've achieved the goal. Often can't be measured directly.

*(Velocity and Quality)*

**Metric**

A proxy for a signal, a thing we can measure. It might not be the ideal measurement, but it is something that we believe is close enough.

*(Cycle Time, Change Failure Percentage)*

# Key Metrics in Finance

# Income Statement (Profit and Loss Statement)

**Revenue (Sales)**

The total amount of money generated by the sale of goods or services.

**Expenses**

The costs incurred to generate revenue, including operating expenses, cost of goods sold, interest, and taxes.

**Net Income (Profit)**

Revenue minus expenses, representing the company's profit or loss for a specific period.

# Balance Sheet

**Assets**

Resources owned by the company, such as cash, inventory, property, and investments.

**Liabilities**

Debts and obligations owed by the company, including loans, accounts payable, and accrued expenses.

**Equity**

The residual interest in the assets after deducting liabilities, often represented as owner's equity or shareholder's equity.

# Cash Flow Statement

### Operating Cash Flow

Cash generated or used in the company's core operating activities.

### Investing Cash Flow

Cash used for investing in assets like property, equipment, or investments.

### Financing Cash Flow

Cash from or used for financing activities, including borrowing, repaying debt, and issuing or buying back stock.

### Net Cash Flow

The overall change in cash and cash equivalents during a specific period.

# Statement of Retained Earnings

### Beginning Retained Earnings

The retained earnings balance at the start of the reporting period.

### Net Income

The profit or loss for the period as reported in the income statement.

### Dividends

The amount of earnings distributed to shareholders as dividends.

### Ending Retained Earnings

The final retained earnings balance after adjusting for net income and dividends.

# Notes to Financial Statements (Footnotes)

These provide additional information and context about various aspects of the financial statements, such as accounting policies, contingencies, and significant transactions.

These primary metrics help investors, creditors, and stakeholders assess a company's financial health, profitability, liquidity, and overall performance. They are essential for making informed decisions about investing, lending, or transacting with the business.

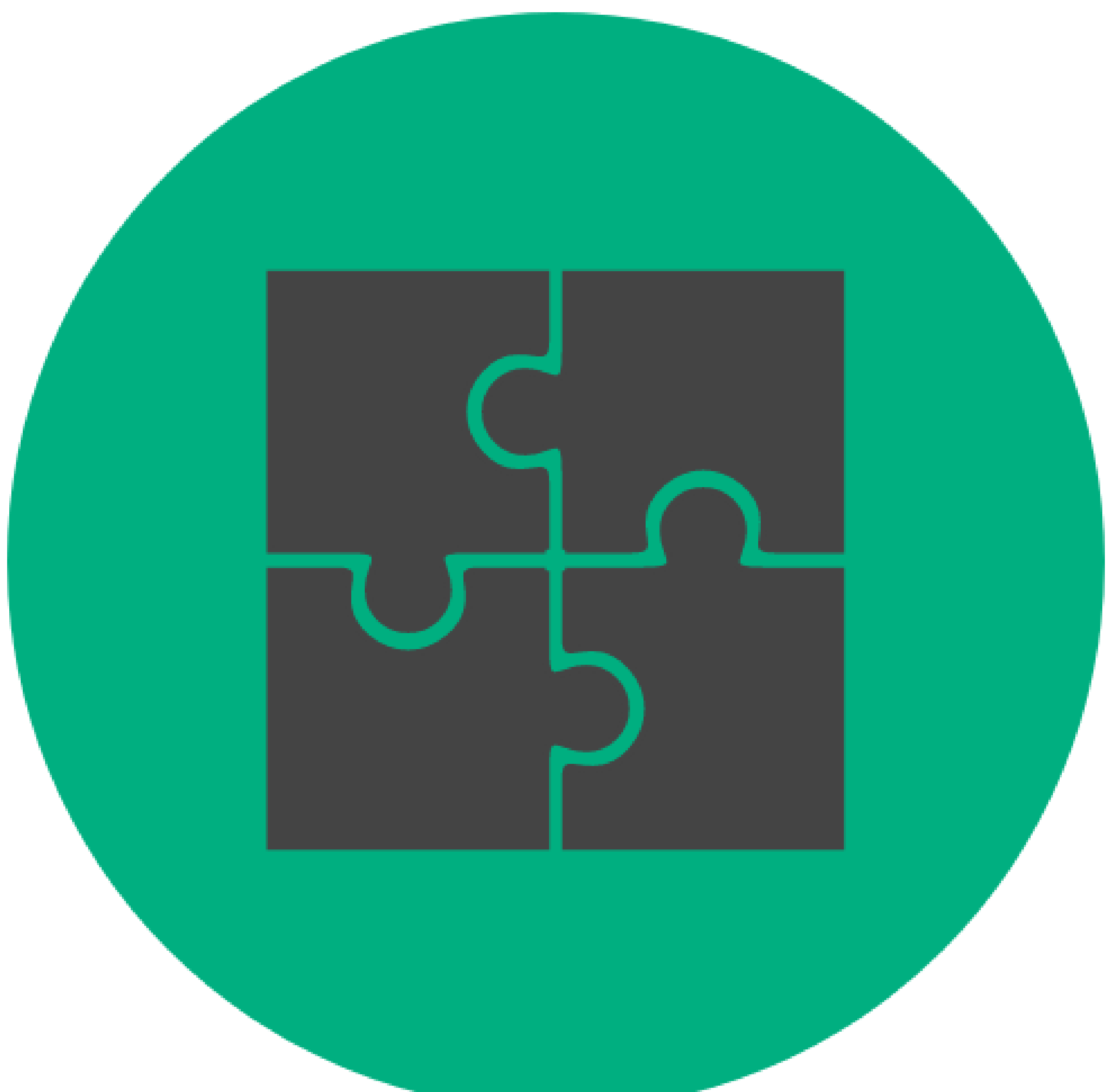
# Stock Price

## There is one metric that shows the result of collective forecasting of value

The value of a stock represents the current market price at which a share of that stock can be bought or sold on a stock exchange. It is determined by supply and demand factors in the market and can fluctuate throughout the trading day as investors buy and sell shares. The term "market capitalization" is often used to describe the total value of a company's outstanding shares of stock, which is calculated by multiplying the stock price by the total number of shares outstanding.

# The Trouble with Metrics

## All metrics can be misleading or be gamed



- Sales can be gamed by subsidizing products (Uber)
- Operating earnings show unusually high expenses due to stock based compensation (Cloudflare)
- Sell off assets, do a big layoff, sell an acquisition (nothing to do with revenue)

What about private companies?

# Let's be honest

A lot of investing is done with very little data

### Venture Capital

- "Street cred", investor reputation
- Try to be associated with the most "prestigious" investor (lead and follow investors)
- Limited deal flow means having to take some deals, whatever they are
- Bet on the founders

### Private Equity

- Business that are done growing
  - Very profitable, but not growing (Cigarette companies)
  - Buy this and maximize efficiency, combined, separating, etc, financialize
  - Financial engineering

# Key Metrics in the Customer Lifecycle

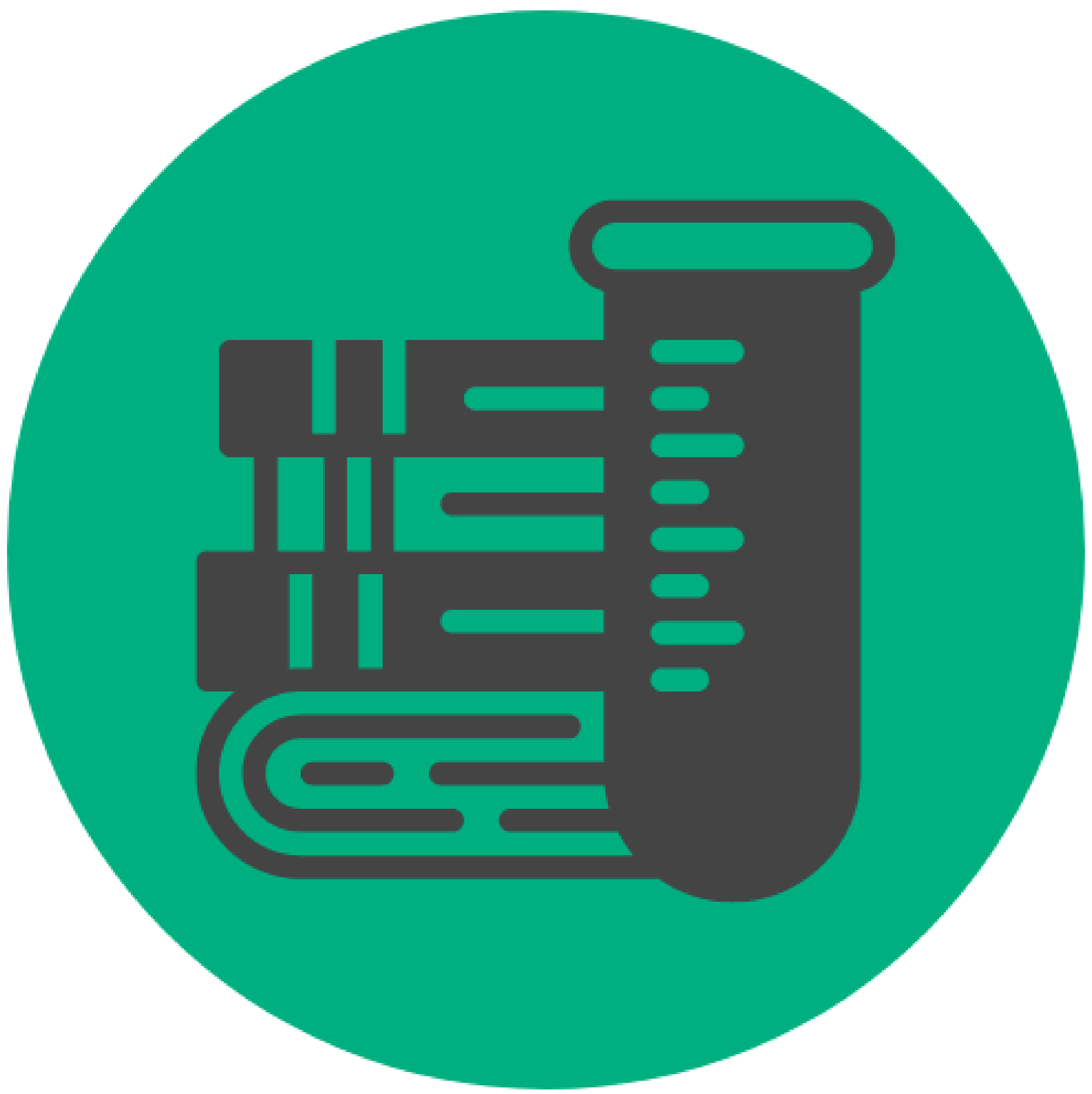
- **Monthly Active Users (MAU) or Daily Active Users (DAU)** These metrics indicate the number of users who engage with the company's product on a regular basis.
- **Conversion Rates** Conversion rates measure how effectively the company can turn users into paying customers or encourage specific actions (e.g., signing up for a newsletter, downloading an app).
- **Retention Rate** This metric measures the percentage of customers who continue to use the product over a specific period, such as a month or year.
- **NPS (Net Promoter Score)** NPS measures customer satisfaction and loyalty by asking users how likely they are to recommend the product to others.

# Software Development Lifecycle Metrics

In 2018, researchers set out to discover...

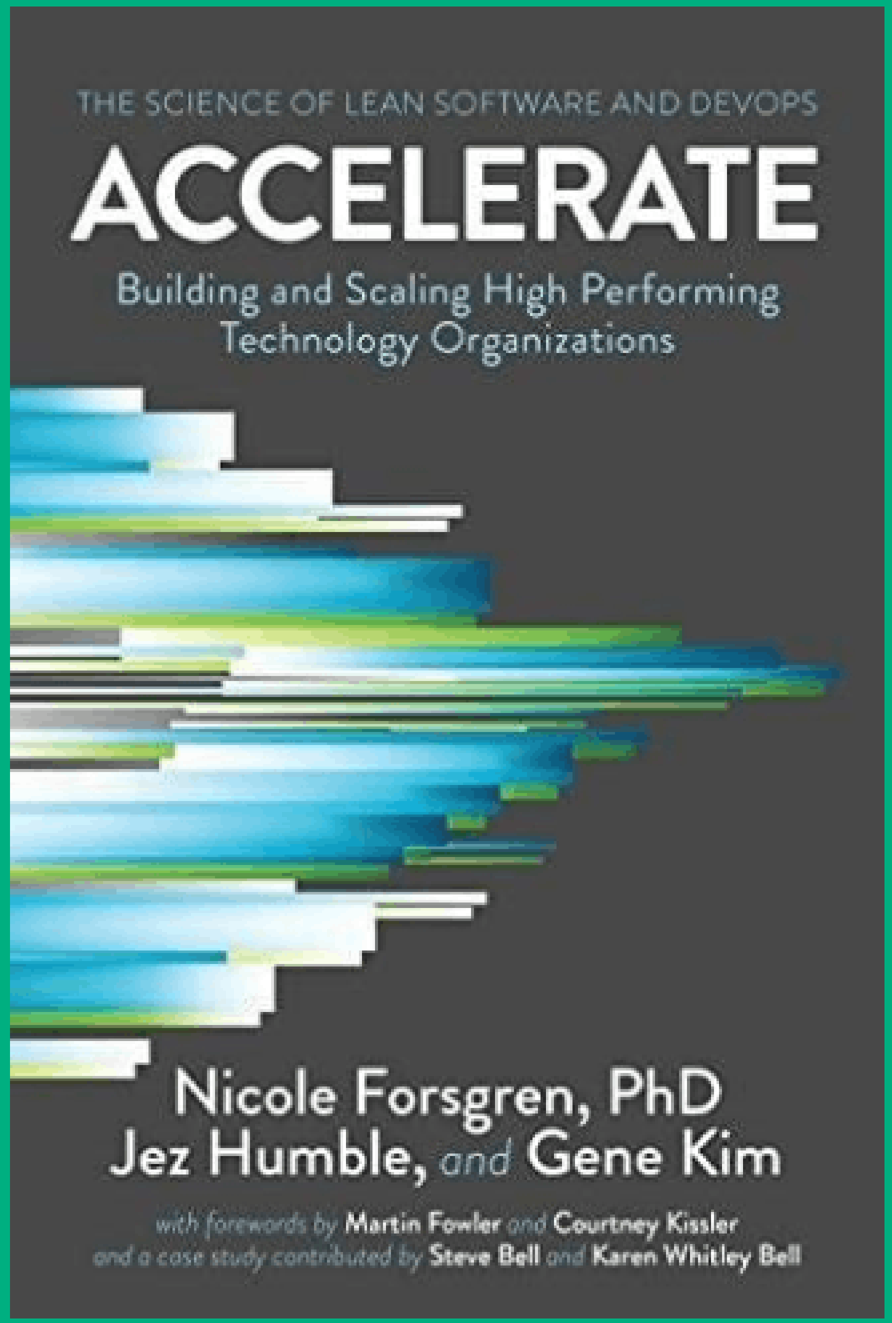
# What makes high performing organizations great?

# DevOps Research and Assessment



Research team that spent six years conducting surveys and analyzing data to answer the question: "What behaviors result in the best performing software engineering teams and what metrics show measurement of this?"

# Data Driven, Rigorous Research




- Survey responses: 23,000
- Organizations: 2,000
- Size: Small startups under 5 people to large enterprises with over 10,000 employees
- Types: Cutting edge internet companies, highly regulated industries including finance, healthcare and government
- Age of code: Greenfield and legacy code

# What does DORA stand for?

## Context of Software Development

- Nickelodeon: "Dora the Explorer", an American media franchise centered on an eponymous animated interactive fourth wall children's television series

- Software: The name of the team researching software development capabilities, metrics and business outcomes: DevOps Research and Assessment

## Gold Standard

The DORA research results and data have become the standard of measurement for people responsible for tracking engineering performance in their organization.

# 24 Key Capabilities

# Continuous Delivery

- Use Version Control for all Production Artifacts
- Automate Your Deployment Process
- Implement Continuous Integration
- Use Trunk-Based Development Methods
- Implement Test Automation
- Support Test Data Management
- Shift Left on Security
- Implement Continuous Delivery (CD)

# Architecture

- Use a Loosely Coupled Architecture
- Architect for Empowered Teams

# Product and Process

- Gather and Implement Customer Feedback
- Make the Flow of Work Visible through the Value Stream
- Work in Small Batches
- Foster and Enable Team Experimentation

# Management and Monitoring

- Have a Lightweight Change Approval Processes
- Monitor across Application and Infrastructure to Inform Business Decisions
- Check System Health Proactively
- Improve Processes and Manage Work with Work-In-Process (WIP) Limits
- Visualize Work to Monitor Quality and Communicate throughout the Team

# Cultural

- Support a Generative Culture
- Encourage and Support Learning
- Support and Facilitate Collaboration among Teams
- Provide Resources and Tools that Make Work Meaningful
- Support or Embody Transformational Leadership

# Key Metrics in the Software Development Lifecycle

# 4 Key Metrics

- Cycle Time
- Deployment Frequency
- MTTR (Mean Time to Resolution)
- Change Fail Rate

# Velocity Metrics

# Cycle Time

Time from start of change (branch creation) to successful release of a code change (merge to main)

# Deployment Frequency

Number of deployments (merge to main) over time

# Quality Metrics

# Change Failure Rate

Number of change fail fixes (bugs, hotfixes and reverts) divided by total number of changes

# Mean Time to Resolution

Mean time to resolve a production incident

# Velocity and Quality

## No Trade-Off

- High performers do better at both (quality and velocity)
- Low performers do worse at both (quality and velocity)

# Abstract

Finance professionals have sophisticated and nuanced ways of understanding, defining, and leveraging capital. We often speak about "technical debt", but instead consider a new concept coined "technical capital" and how to borrow investment techniques from the financial world to inform technology decisions.