UNIVERSITY OF OREGON

Repository Analysis of Open-Source and Scientific Software Development Projects

Kanika Sood[1], Boyana Norris[1], Anshu Dubey[2], Lois McInnes[2]
University of Oregon[1], Argonne National Laboratory[2]

February 25, 2019

MS2: Scientific Software: Practices, Concerns, and Solution Strategies

SIAM Conference on Computational Science and Engineering

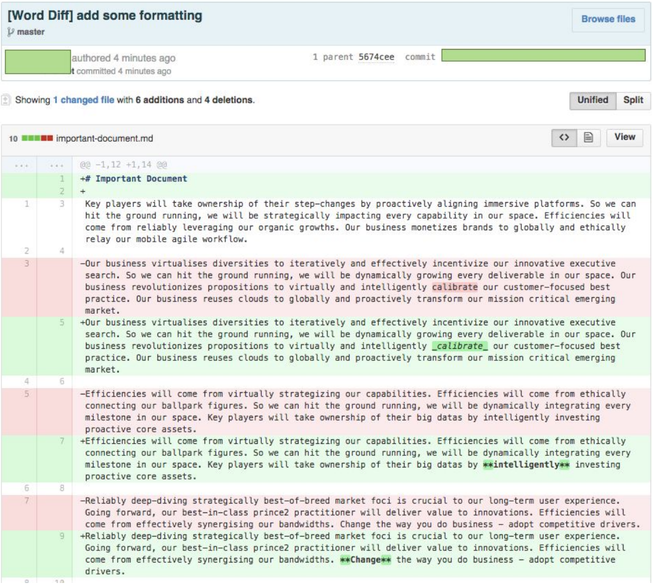
Introduction

- Scientific software is rapidly growing in capabilities, accuracy, performance.
- Developer productivity has received less attention than app. performance/ publications.
- We propose new time-dependent metrics that can help quantify team productivity.
- The metrics can be used to better understand the trends of software development workflows and provide objective measurements of productivity.
- We demonstrate our approach on several HPC software projects.

Disclaimer: The goal of this research is to explore new software metrics that can provide insight into productivity more effectively than existing metrics. These (or any) metrics provide partial perspectives but cannot capture a complete view of the complexities of scientific software projects.

Why is quantifying productivity hard?

Standard metrics



[Word Diff] add some formatting

master

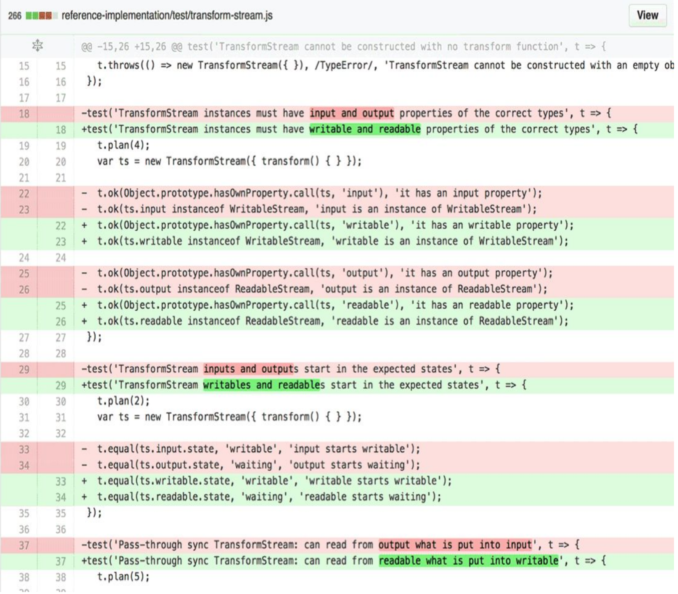
[redacted] authored 4 minutes ago
committed 4 minutes ago

1 parent 5674cee commit [redacted]

Showing 1 changed file with 6 additions and 4 deletions.

10 important-document.md

```
@@ -1,12 +1,14 @@
+# Important Document
+
 Key players will take ownership of their step-changes by proactively aligning immersive platforms. So we can
 hit the ground running, we will be strategically impacting every capability in our space. Efficiencies will
 come from reliably leveraging our organic growths. Our business monetizes brands to globally and ethically
 relay our mobile agile workflow.

-Our business virtualises diversities to iteratively and effectively incentivize our innovative executive
 search. So we can hit the ground running, we will be dynamically growing every deliverable in our space. Our
 business revolutionizes propositions to virtually and intelligently calibrate our customer-focused best
 practice. Our business reuses clouds to globally and proactively transform our mission critical emerging
 market.
+Our business virtualises diversities to iteratively and effectively incentivize our innovative executive
 search. So we can hit the ground running, we will be dynamically growing every deliverable in our space. Our
 business revolutionizes propositions to virtually and intelligently _calibrate_ our customer-focused best
 practice. Our business reuses clouds to globally and proactively transform our mission critical emerging
 market.

-Efficiencies will come from virtually strategizing our capabilities. Efficiencies will come from ethically
 connecting our ballpark figures. So we can hit the ground running, we will be dynamically integrating every
 milestone in our space. Key players will take ownership of their big datas by intelligently investing
 proactive core assets.
+Efficiencies will come from virtually strategizing our capabilities. Efficiencies will come from ethically
 connecting our ballpark figures. So we can hit the ground running, we will be dynamically integrating every
 milestone in our space. Key players will take ownership of their big datas by **intelligently** investing
 proactive core assets.

-Reliably deep-diving strategically best-of-breed market foci is crucial to our long-term user experience.
 Going forward, our best-in-class prince2 practitioner will deliver value to innovations. Efficiencies will
 come from effectively synergising our bandwidths. Change the way you do business - adopt competitive drivers.
+Reliably deep-diving strategically best-of-breed market foci is crucial to our long-term user experience.
 Going forward, our best-in-class prince2 practitioner will deliver value to innovations. Efficiencies will
 come from effectively synergising our bandwidths. **Change** the way you do business - adopt competitive
 drivers.
```



266 reference-implementation/test/transform-stream.js

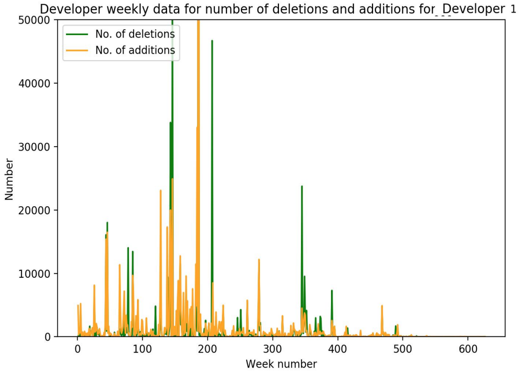
```
@@ -15,26 +15,26 @@ test('TransformStream cannot be constructed with no transform function', t => {
   t.throws(() => new TransformStream({ }), /TypeError/, 'TransformStream cannot be constructed with an empty ob
 });

-test('TransformStream instances must have input and output properties of the correct types', t => {
+test('TransformStream instances must have writable and readable properties of the correct types', t => {
   t.plan(4);
   var ts = new TransformStream({ transform() { } });

-  t.ok(Object.prototype.hasOwnProperty.call(ts, 'input'), 'it has an input property');
-  t.ok(ts.input instanceof WritableStream, 'input is an instance of WritableStream');
+  t.ok(Object.prototype.hasOwnProperty.call(ts, 'writable'), 'it has an writable property');
+  t.ok(ts.writable instanceof WritableStream, 'writable is an instance of WritableStream');

-  t.ok(Object.prototype.hasOwnProperty.call(ts, 'output'), 'it has an output property');
-  t.ok(ts.output instanceof ReadableStream, 'output is an instance of ReadableStream');
+  t.ok(Object.prototype.hasOwnProperty.call(ts, 'readable'), 'it has an readable property');
+  t.ok(ts.readable instanceof ReadableStream, 'readable is an instance of ReadableStream');
 });

-test('TransformStream inputs and outputs start in the expected states', t => {
+test('TransformStream writables and readables start in the expected states', t => {
   t.plan(2);
   var ts = new TransformStream({ transform() { } });

-  t.equal(ts.input.state, 'writable', 'input starts writable');
-  t.equal(ts.output.state, 'waiting', 'output starts waiting');
+  t.equal(ts.writable.state, 'writable', 'writable starts writable');
+  t.equal(ts.readable.state, 'waiting', 'readable starts waiting');
 });

-test('Pass-through sync TransformStream: can read from output what is put into input', t => {
+test('Pass-through sync TransformStream: can read from readable what is put into writable', t => {
   t.plan(5);
```

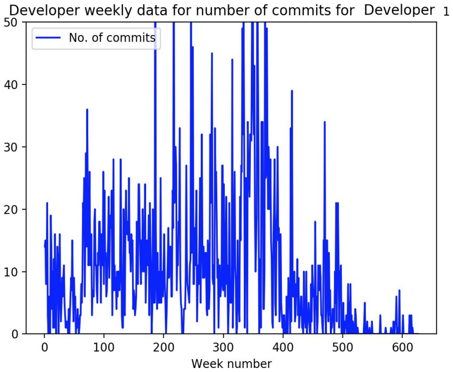
Current metrics like NLOC are insufficient for quantifying software productivity

Metrics

- Commit frequency
- Total additions, deletions
- NLOC

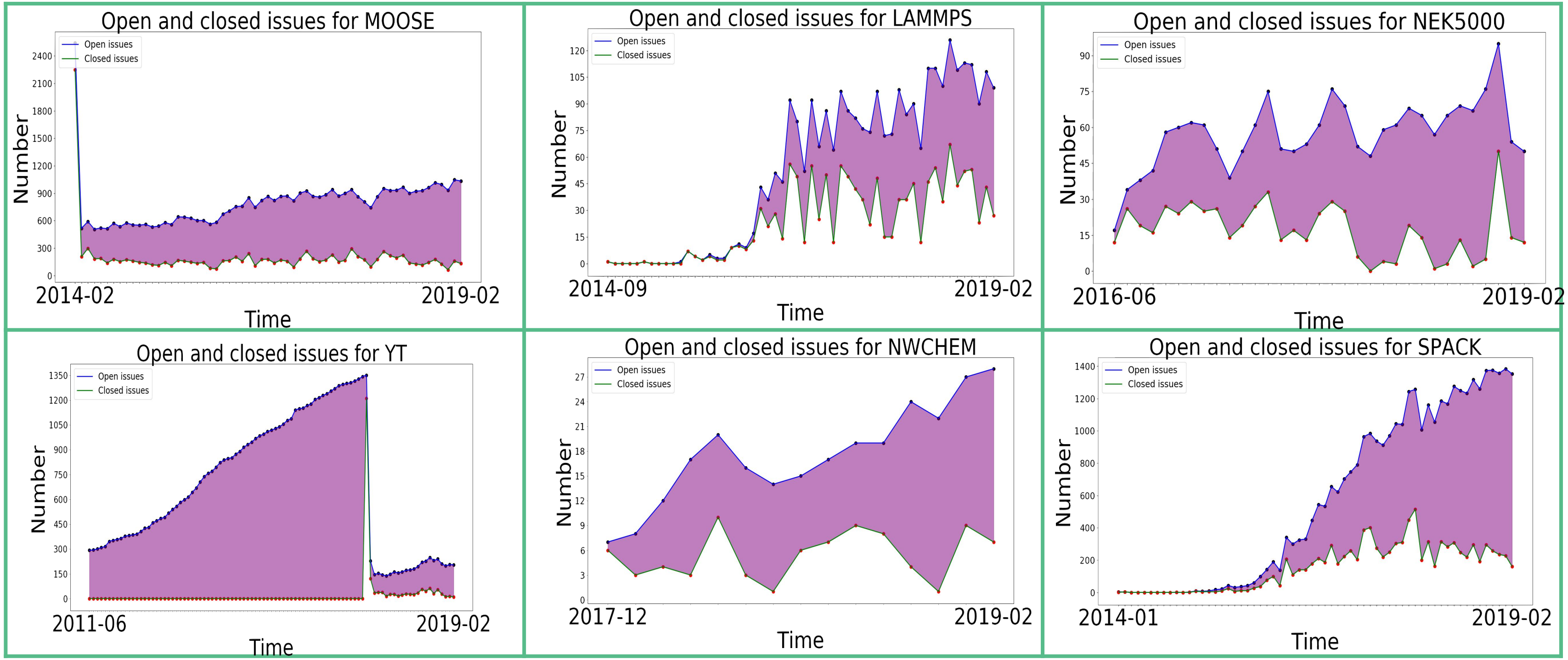
- Issue requests
- Issue categorization
- Project change -NLOC/Files/complexity
- Topics of discussions
- Developer activity
- Developer contributions
- Project reliance


Developer weekly data for number of deletions and additions for Developer 1

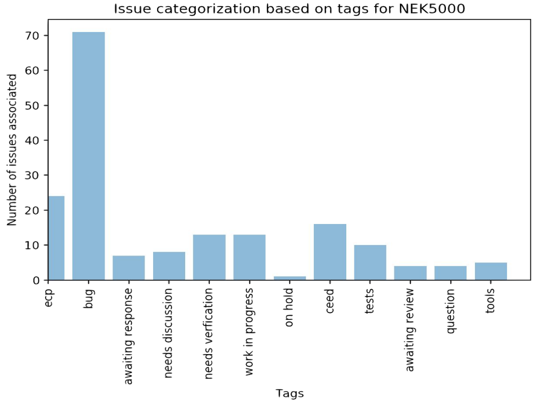

Developer weekly data for number of commits for Developer 1

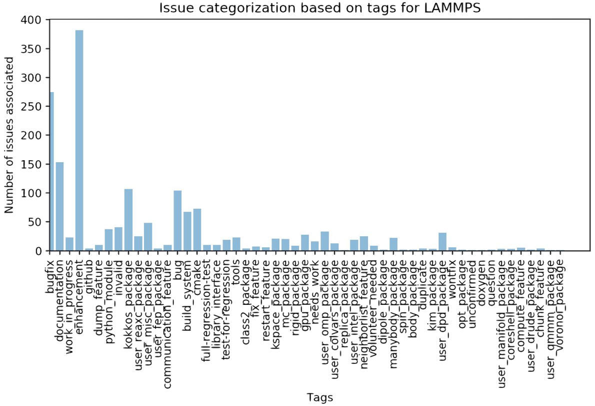
How engaged are the user and developer communities?

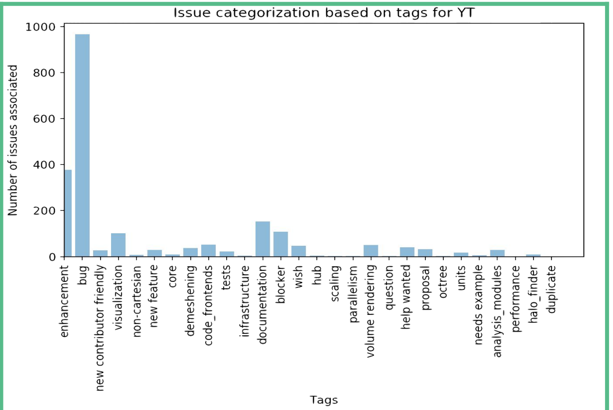
Issues

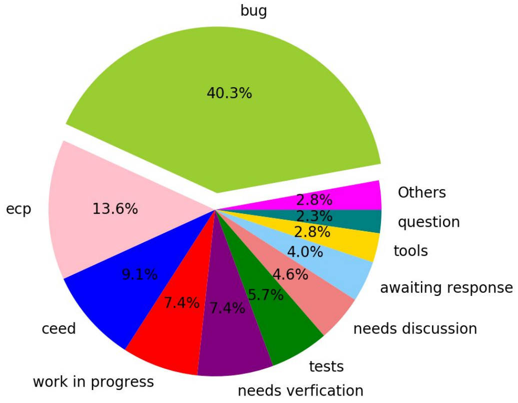


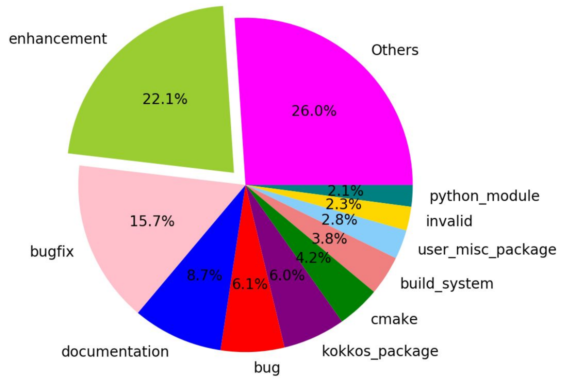
What are hot topics (based on issues)?

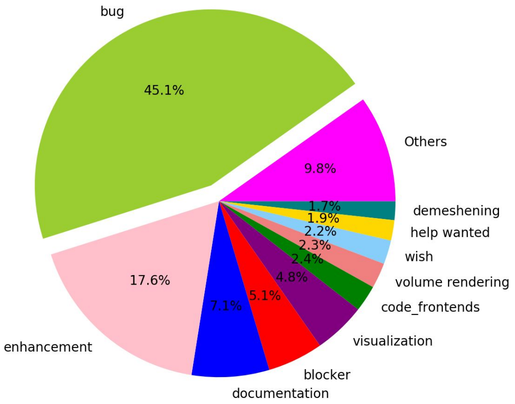






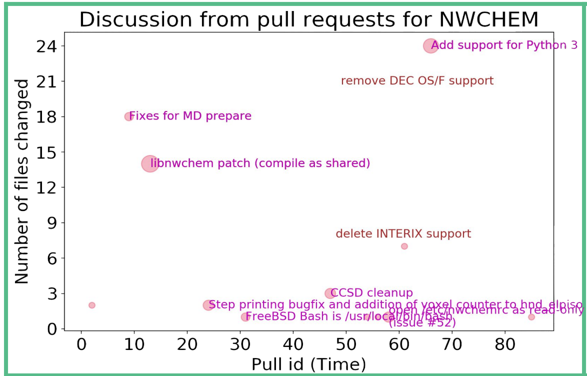


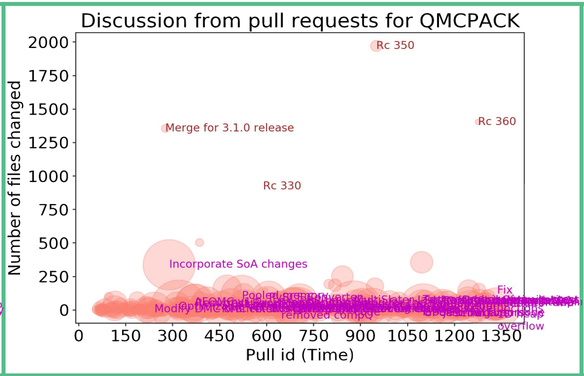


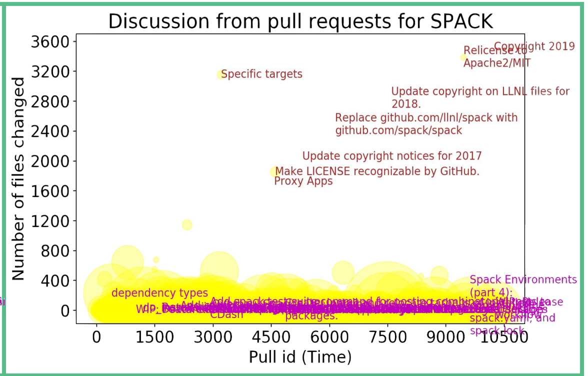


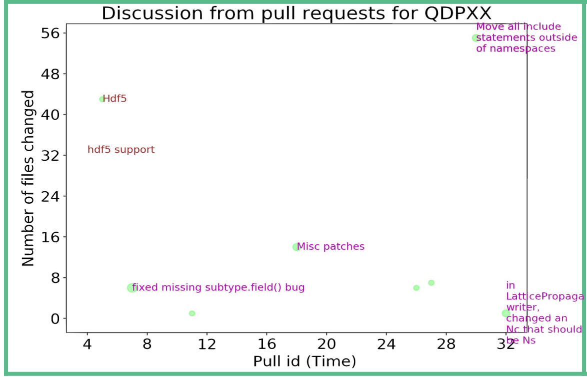
What topics dominate changes/discussion?

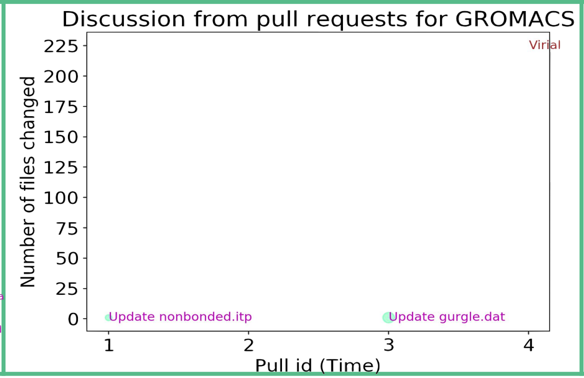
Pull requests

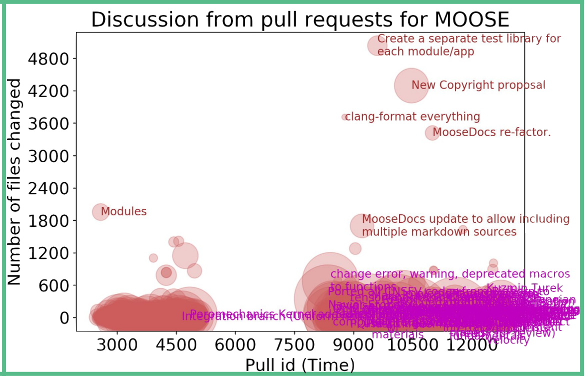










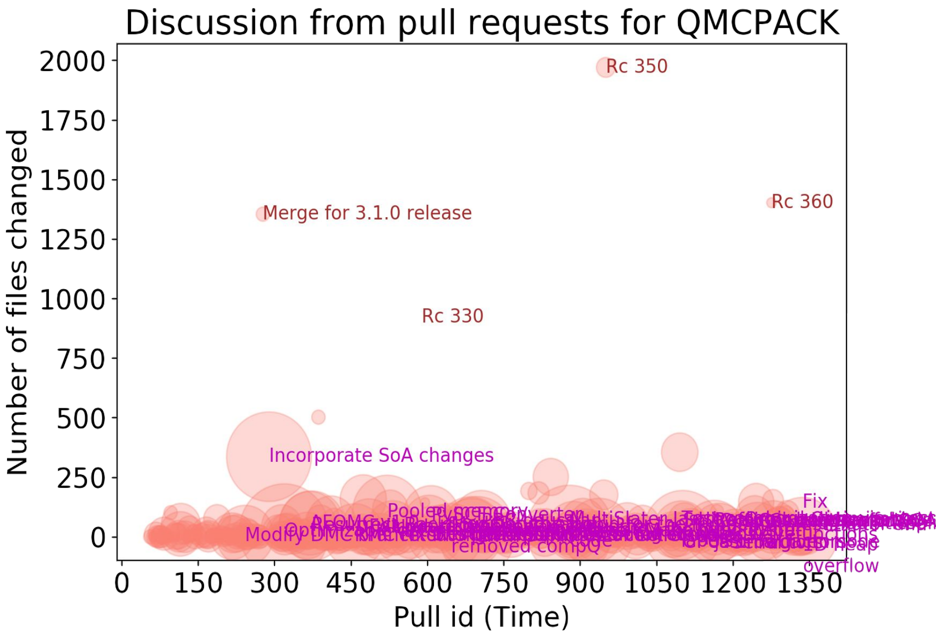


What topics dominate changes/discussion?

Pull requests

Topic	No. of files changed
Rc 350	1971
Rc 360	1403

Topic	No. of comments
Optimizable determinants	91
Multislater-Jastrow Orbital Optimization code	84



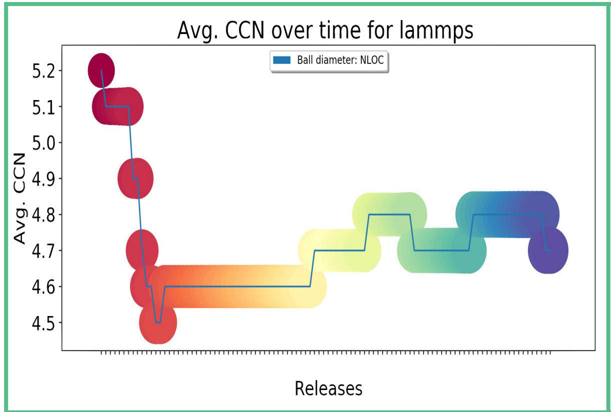
Code analysis

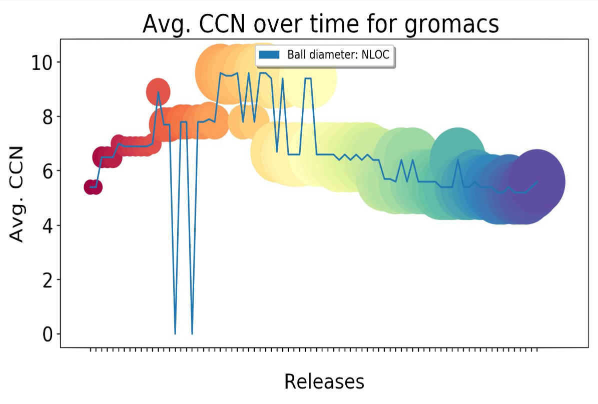
Cyclomatic Complexity (CCN)

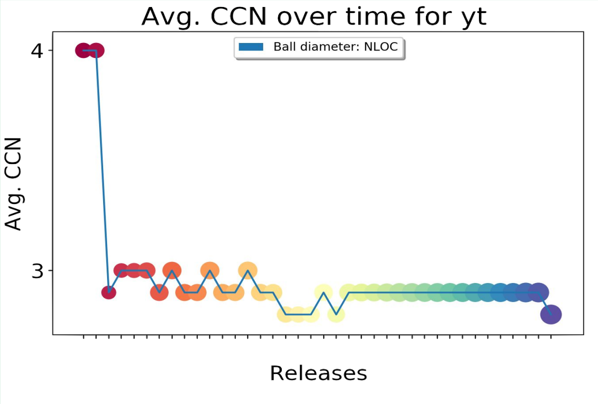
- Quantitative measure of the number of independent paths in the code
- Statements (S1, Sn): nodes, control paths from S1-> Sn: edges
- Computed for each function
- Smaller value : better
- Tool: Lizard
-
```
================================================
  NLOC    CCN   PARAM  length  location
------------------------------------------------
   118      2     50      18 func@6-13@./main.c
```

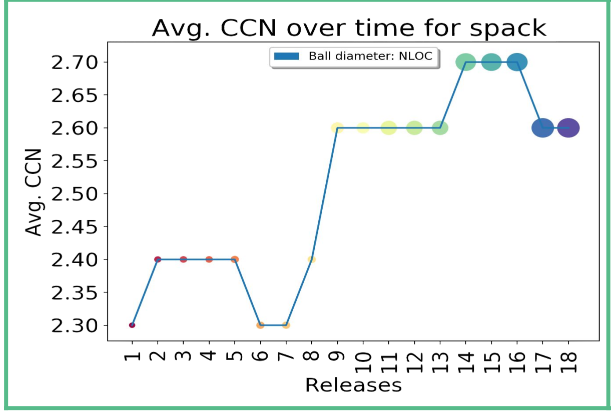
How is code complexity changing over time?

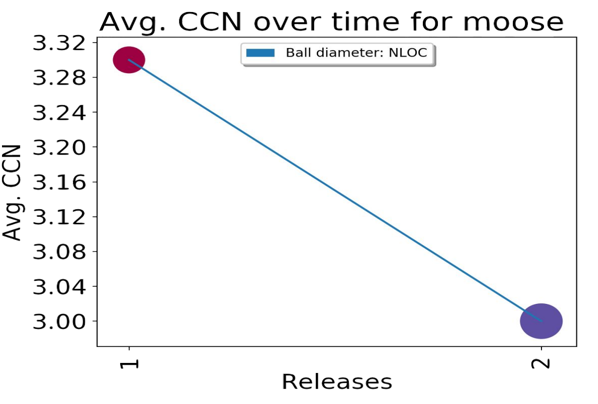
CCN

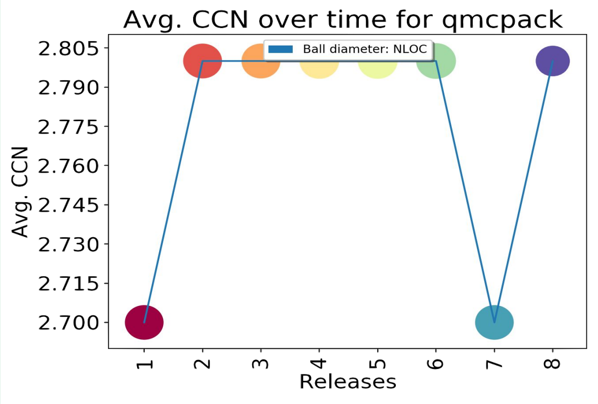

Avg. CCN over time for lammps


Avg. CCN over time for gromacs


Avg. CCN over time for yt

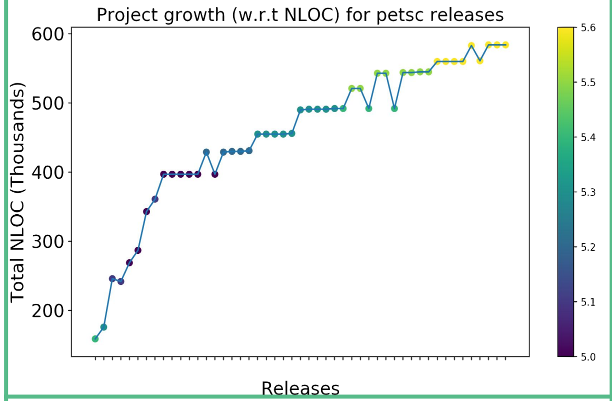

Avg. CCN over time for spack

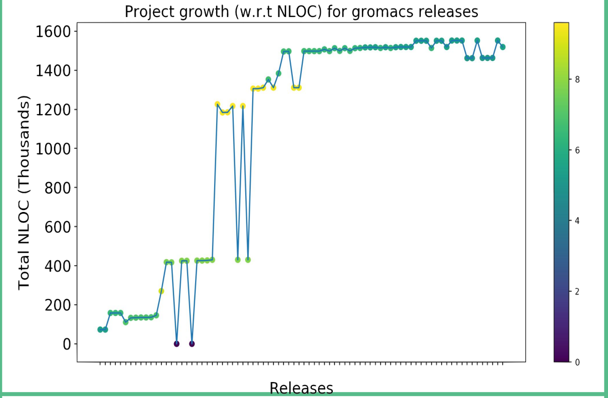

Avg. CCN over time for moose

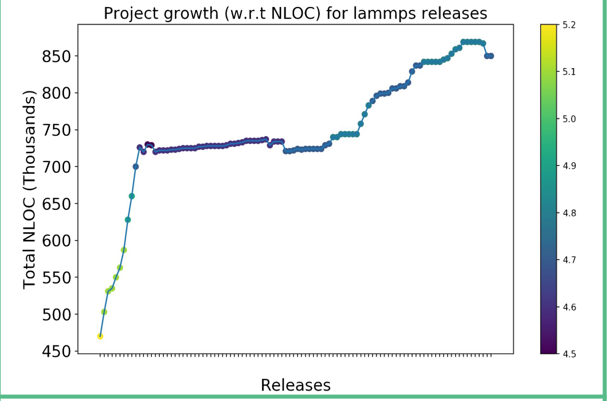

Avg. CCN over time for qmcpack

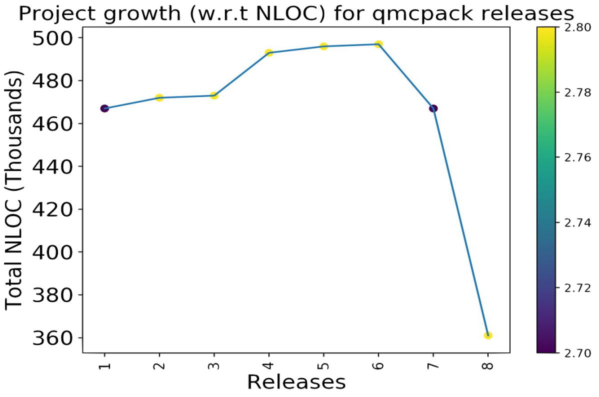
How is the project size changing over time?

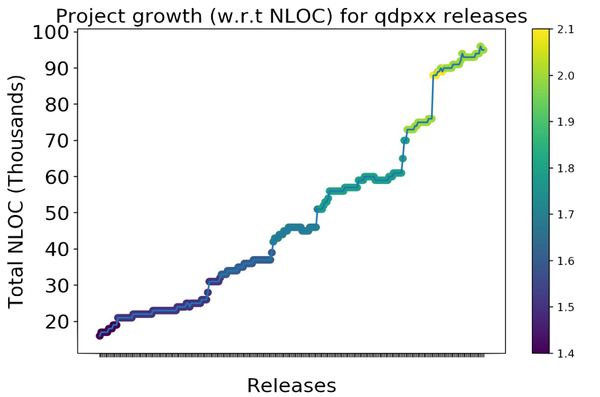
Number of lines of code (NLOC)

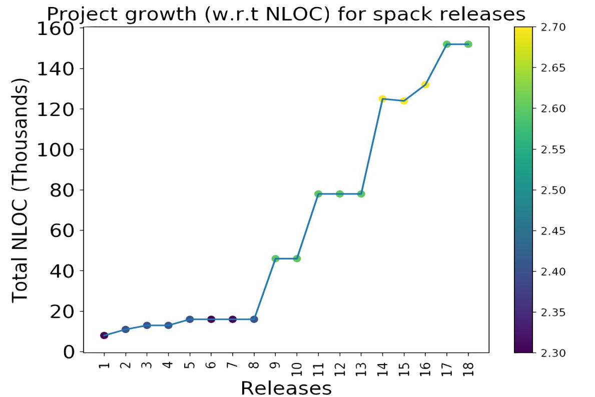




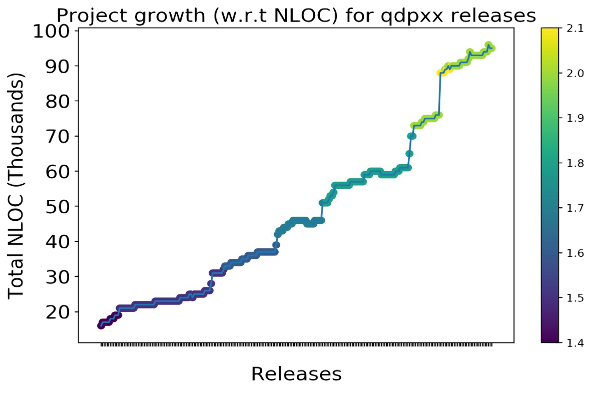


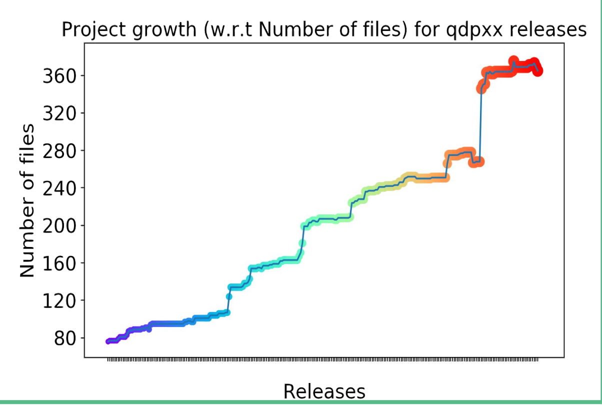


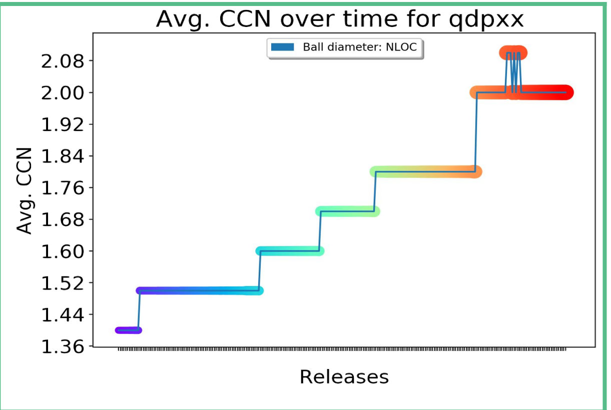




Project evolution: QDPXX

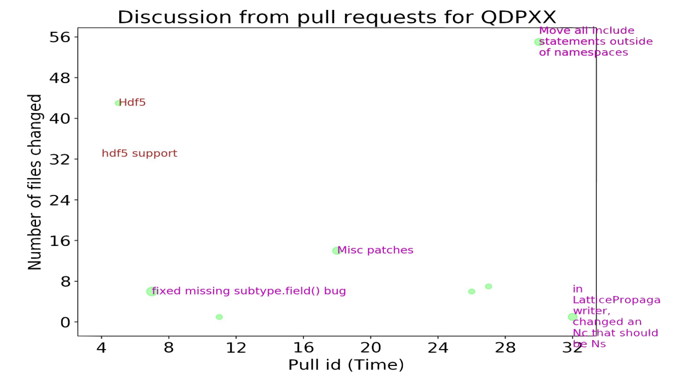


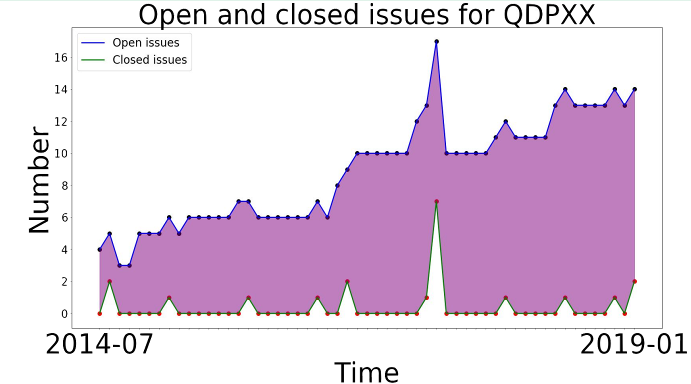




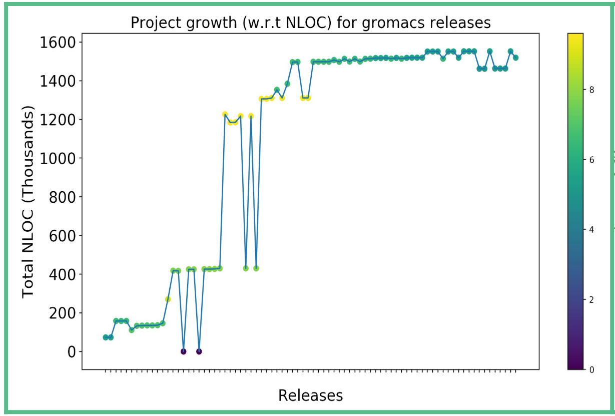
Color coding: Avg. CCN

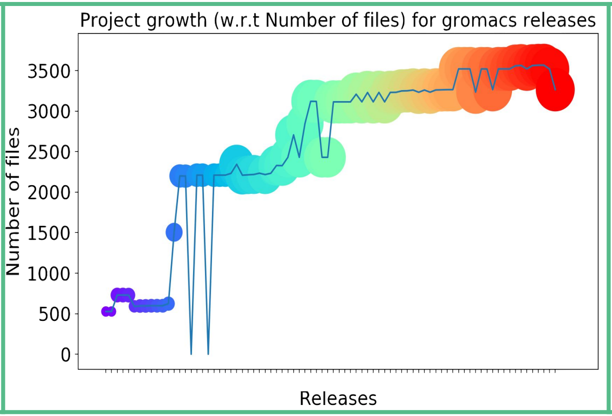
Color coding: Time

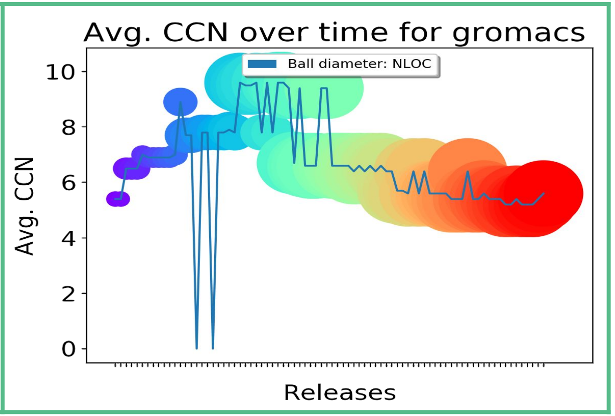




Project evolution: GROMACS

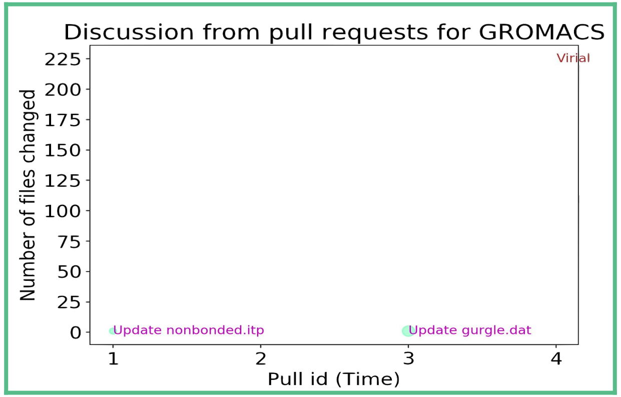






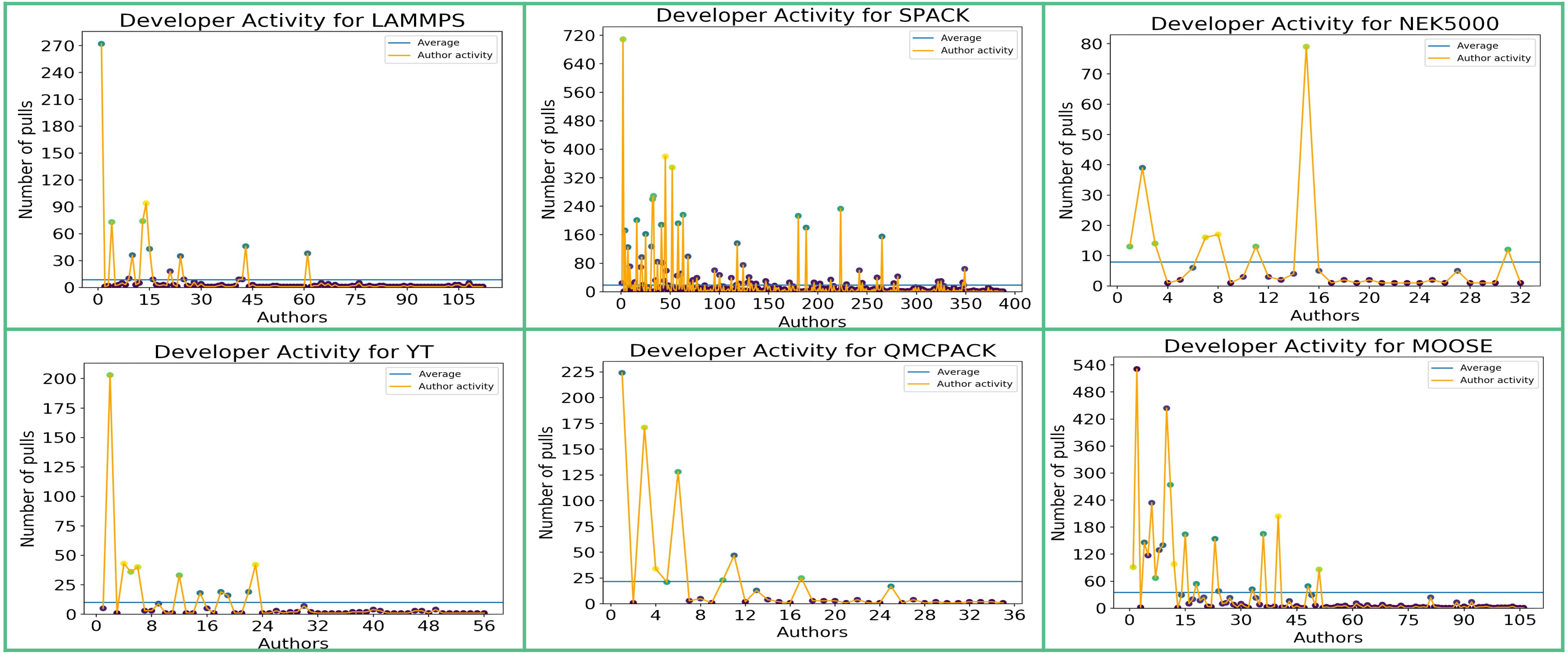
Color coding: Avg. CCN

Color coding: Time



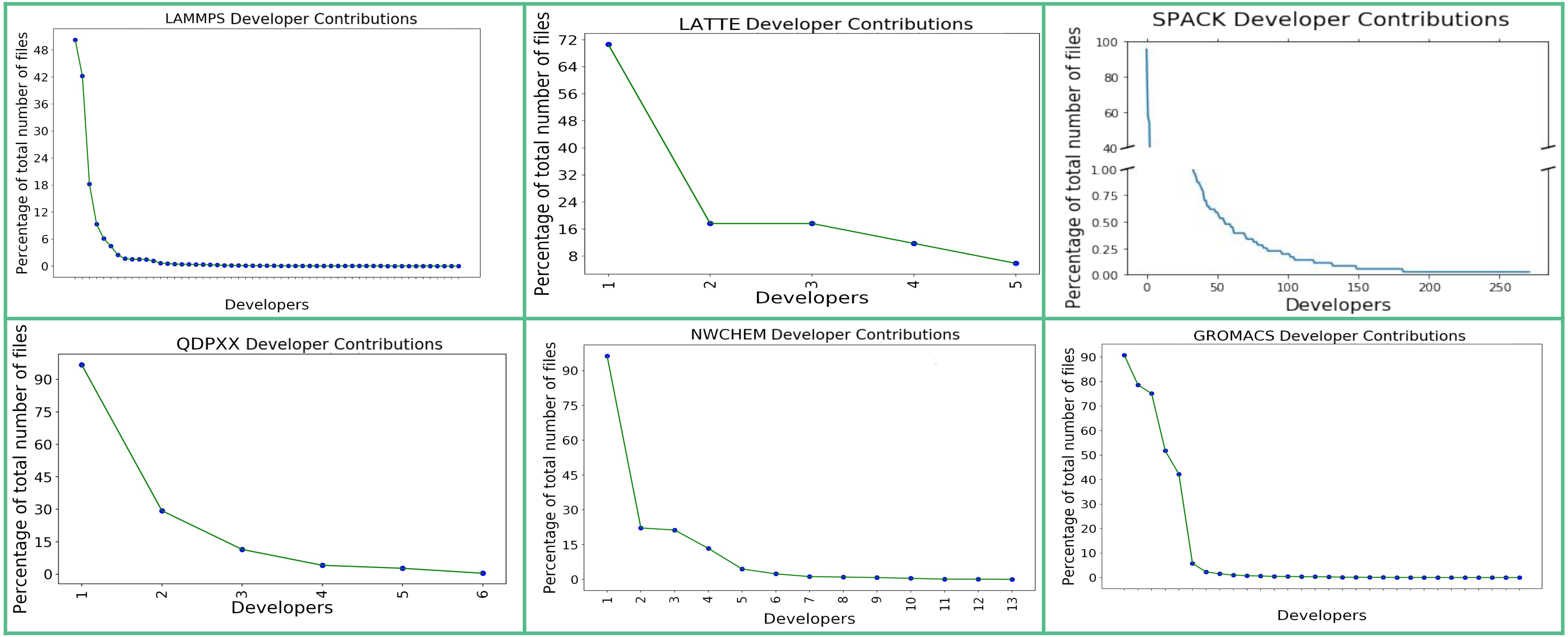
How active is the developer community?

Developer activity based on pull requests



What is the project reliance on individual developers?

Percentage of total number of code files



Conclusion

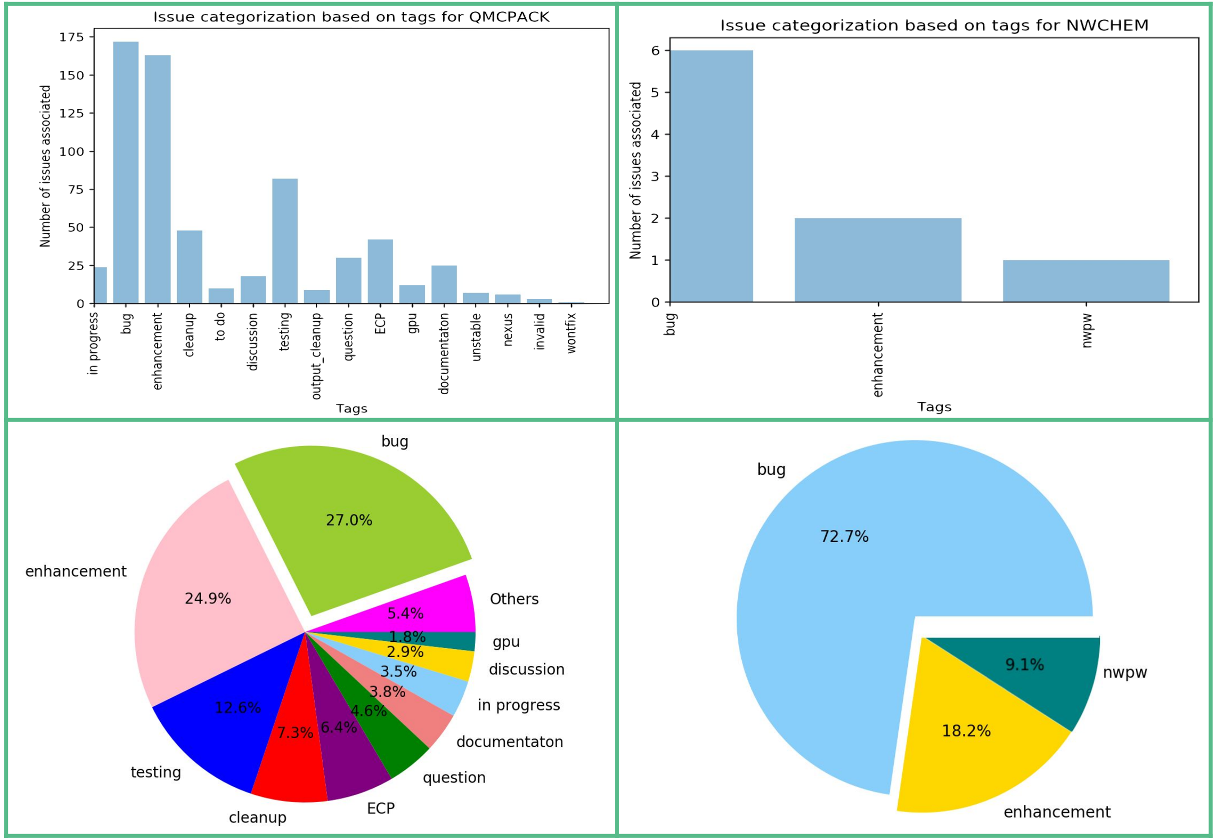
- Compute metrics to better understand software development practices
- Fine-grain analysis for individual files, individual developers, patterns in software trends, and project reliance
- Study the impact of code size and changes in code complexity over project lifetimes.
  - Discover opportunities to reduce cost and increase scientific output
  - Guide future project planning

Disclaimer: The goal of this research is to explore new software metrics that can provide insight into productivity more effectively than existing metrics. These (or any) metrics provide partial perspectives but cannot capture a complete view of the complexities of scientific software projects.

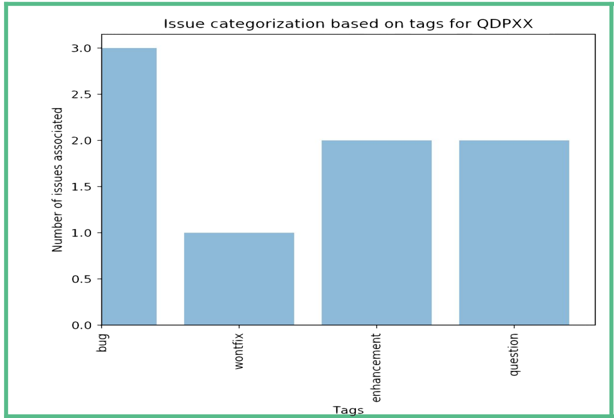
Thank you

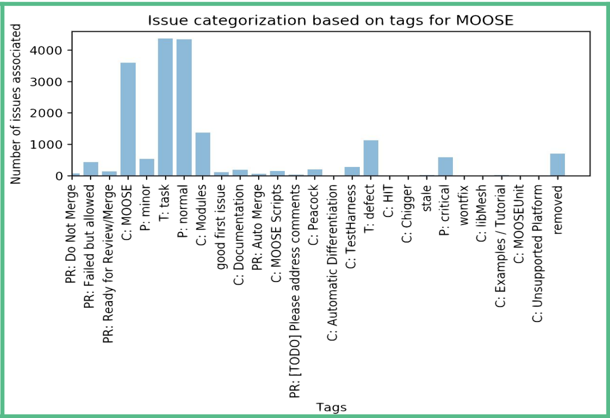
What are hot topics (based on issues)?

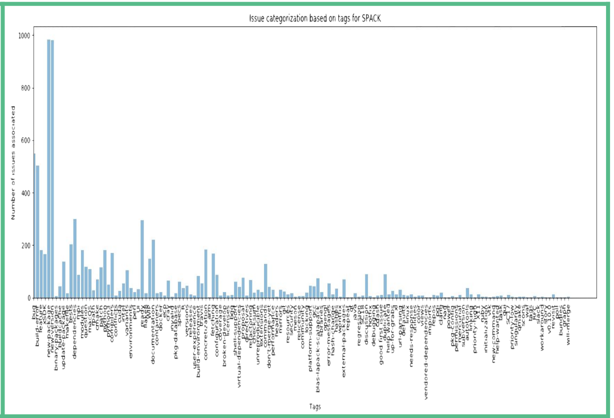
Issue categories

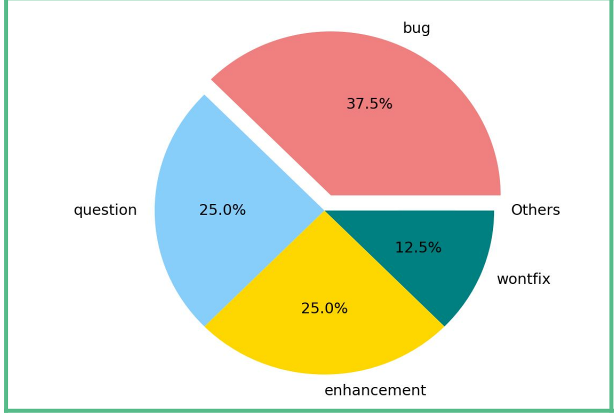


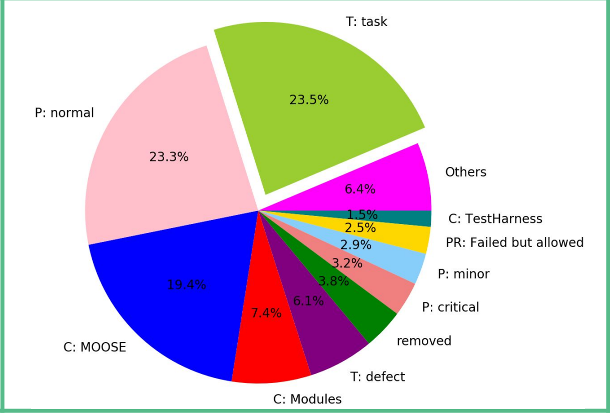
What are hot topics (based on issues)?

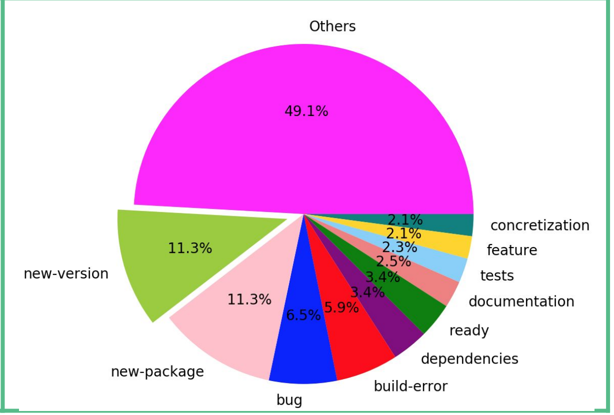




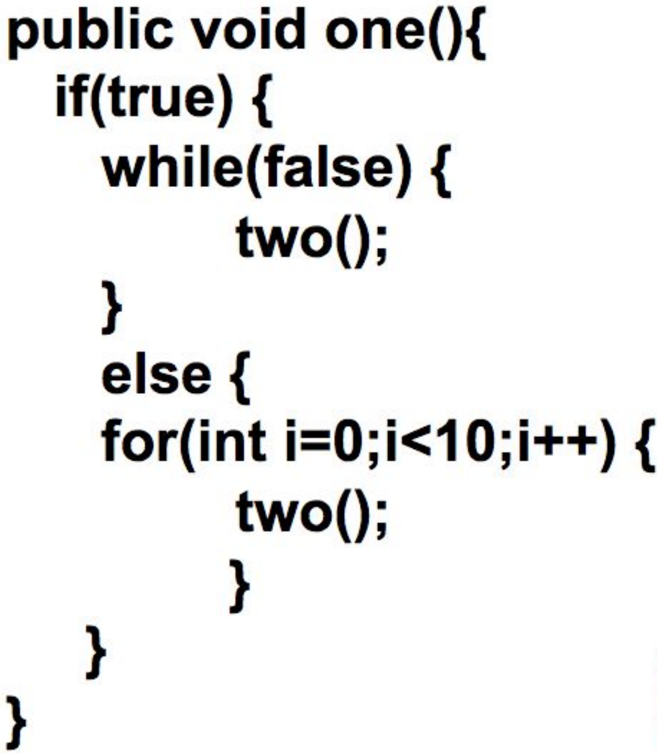








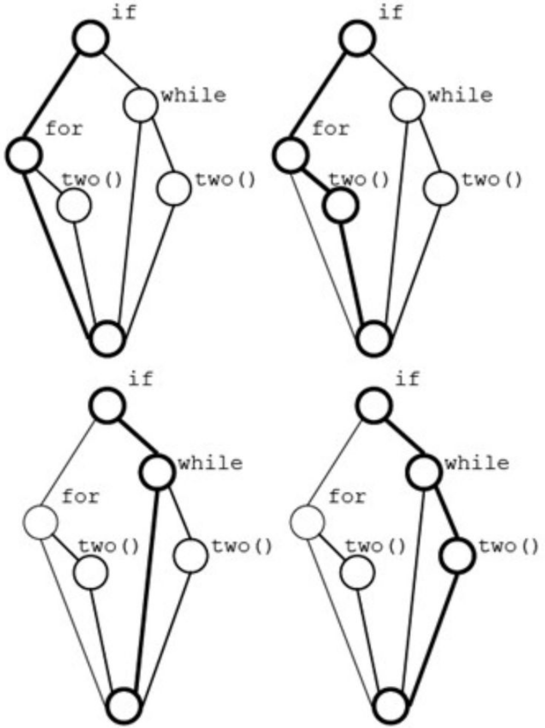
Code complexity



```
public void one(){
  if(true) {
    while(false) {
            two();
    }
    else {
    for(int i=0;i<10;i++) {
            two();
            }
    }
}
```

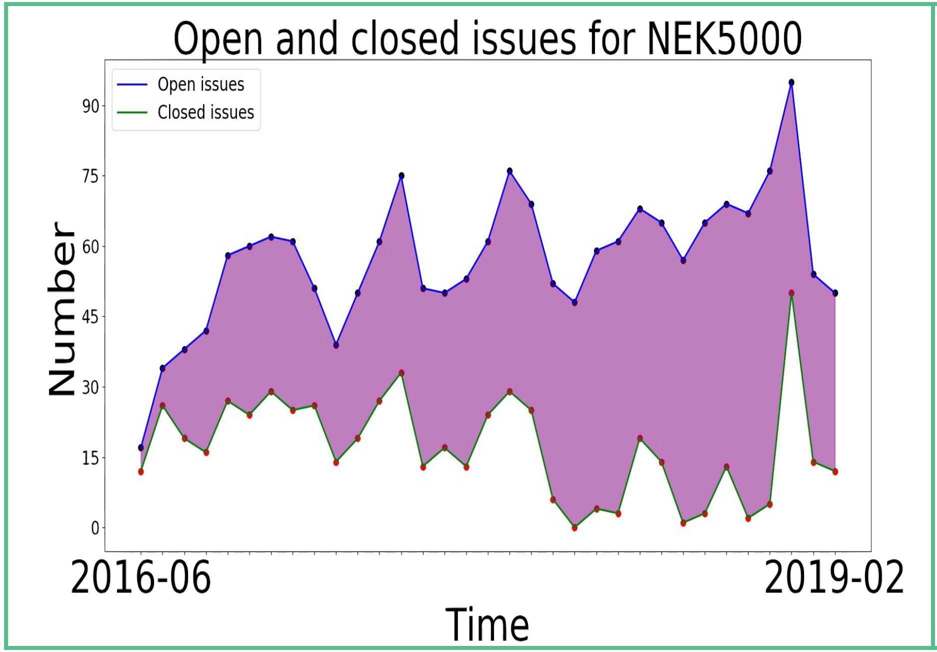
CCN = e - n + 2p

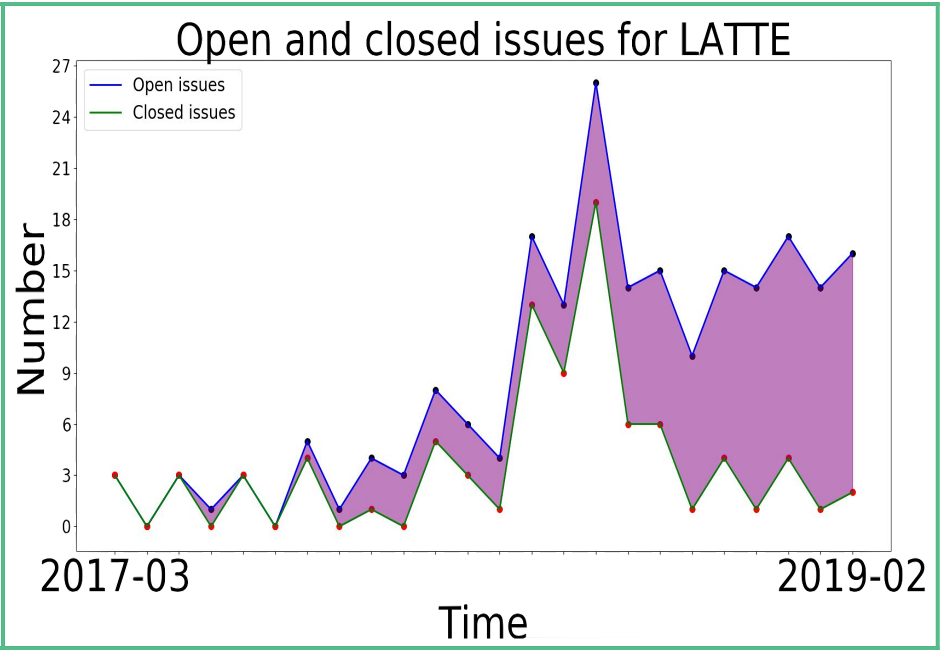
8 - 6 + 2(1) = 4



How engaged is the user and developer community?

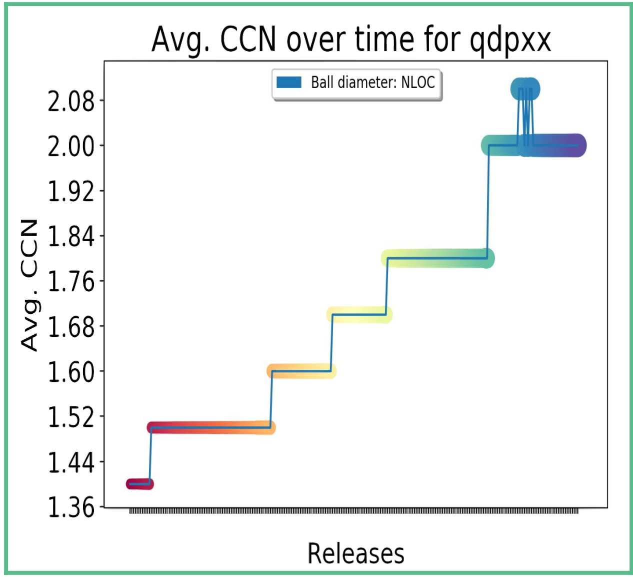
Issue requests

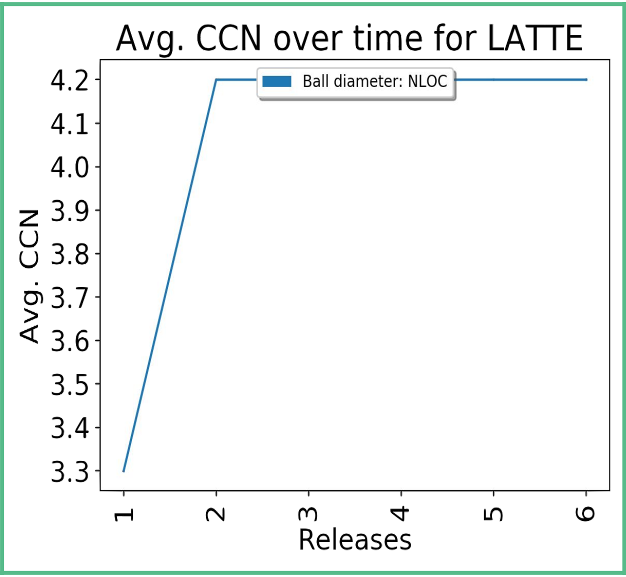


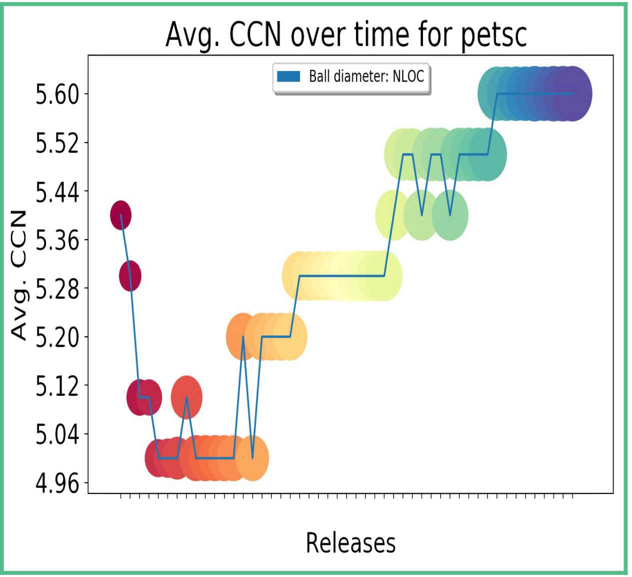


How is code complexity changing over time?

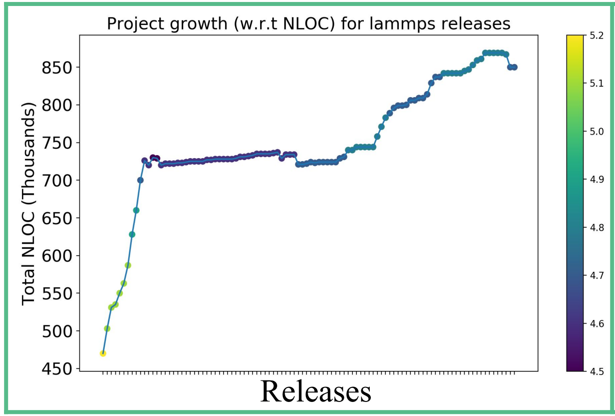
CCN

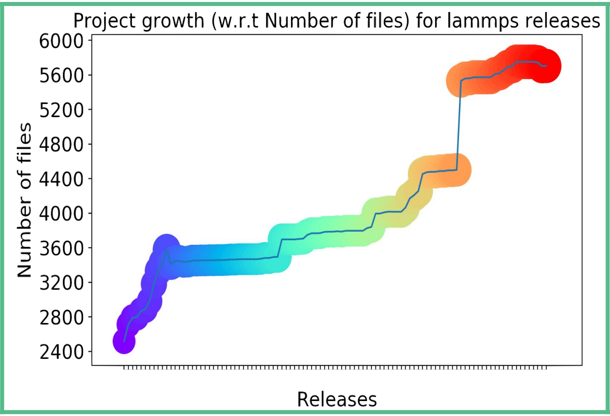


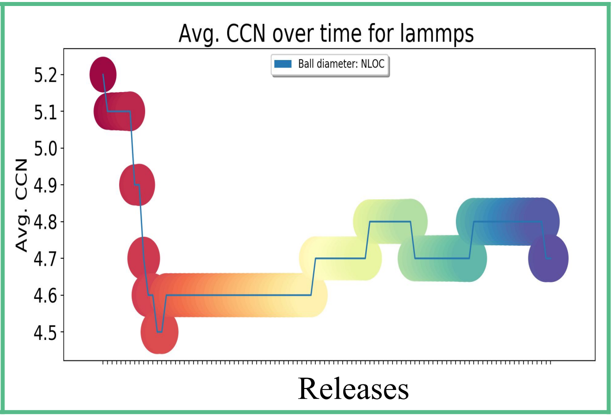




Project evolution: LAMMPS

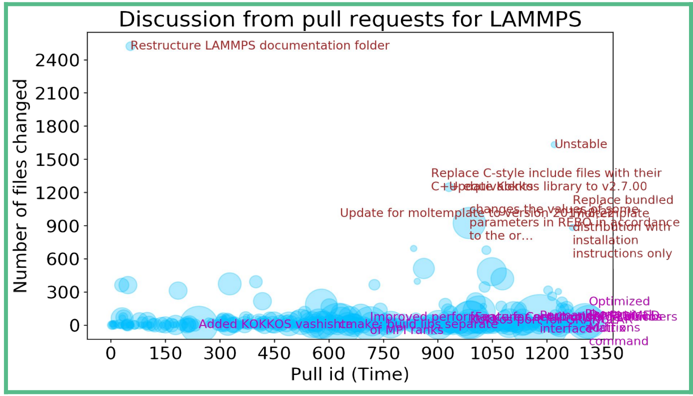


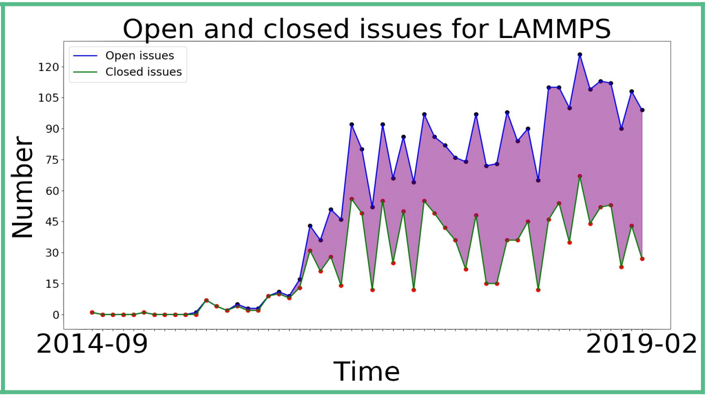




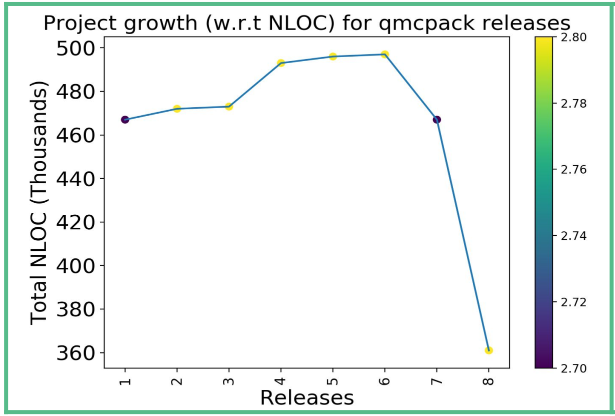
Color coding: Avg. CCN

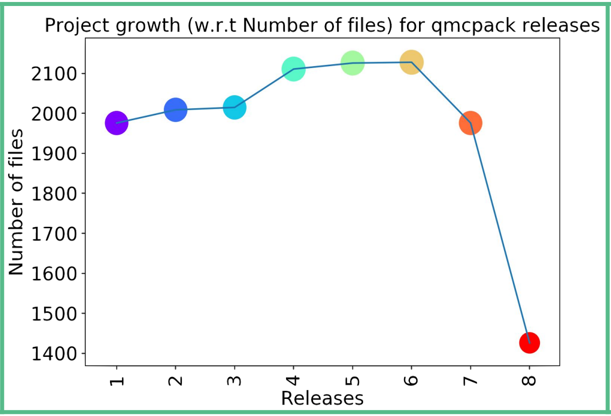
Color coding: Time

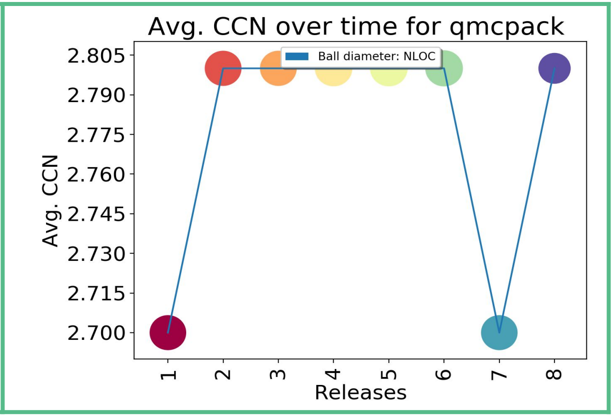




Project evolution: QMCPACK

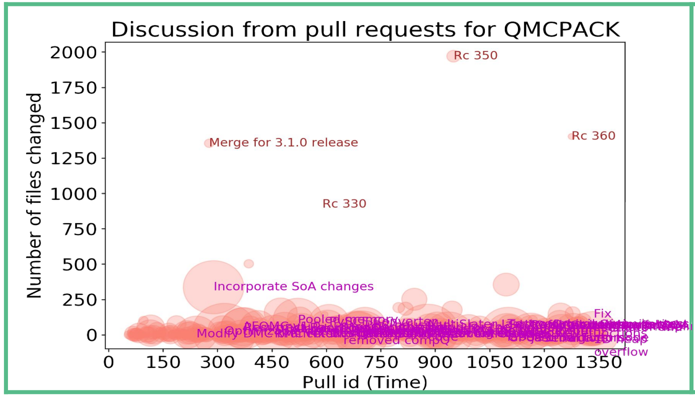


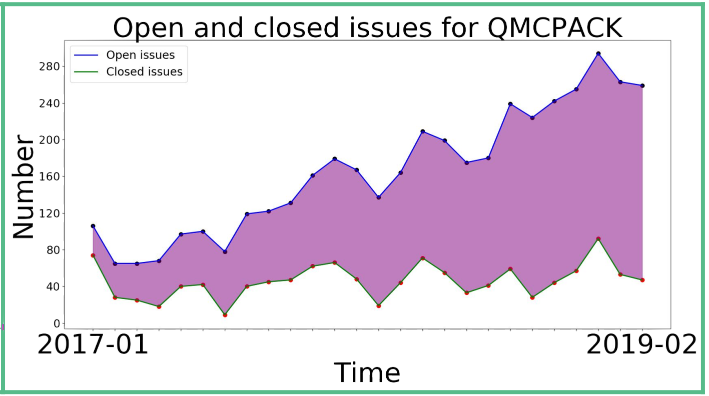




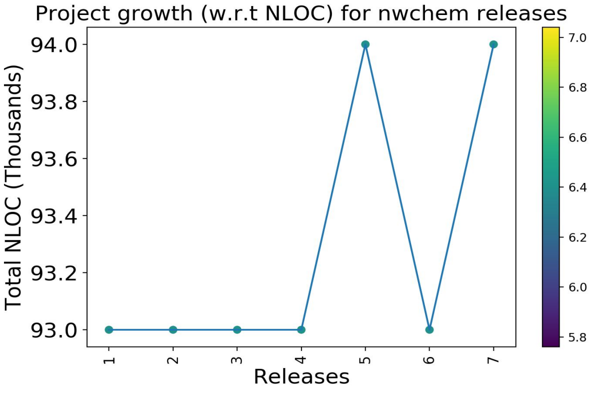
Color coding: Avg. CCN

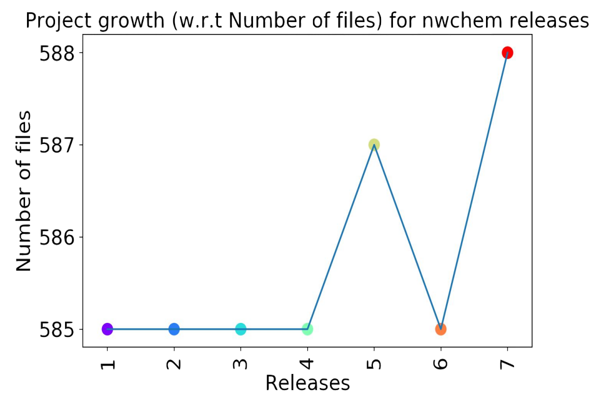
Color coding: Time

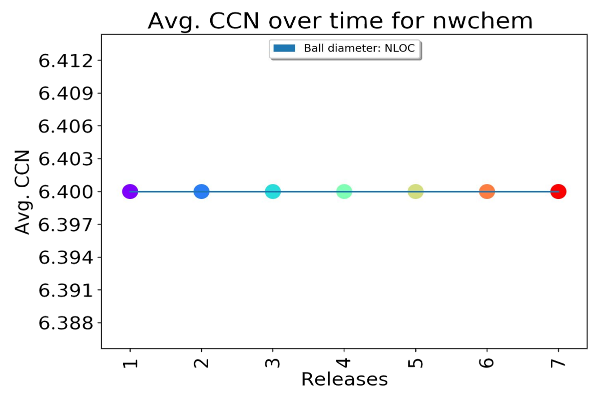




Project evolution: NWCHEM

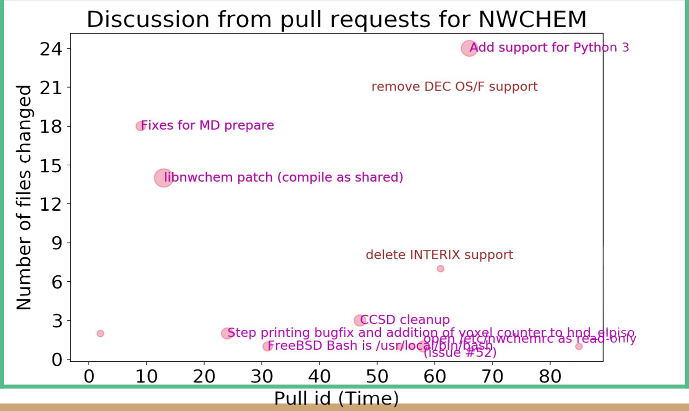


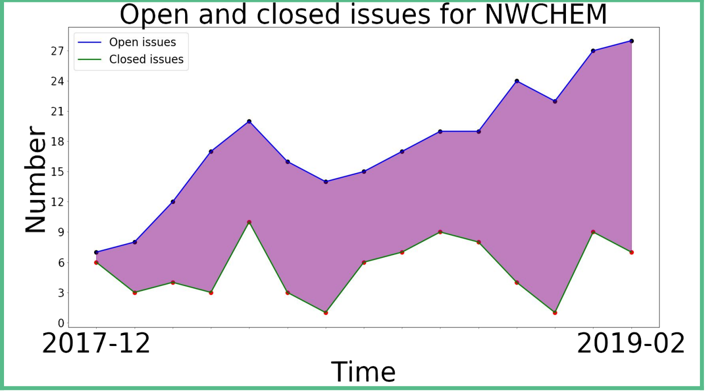




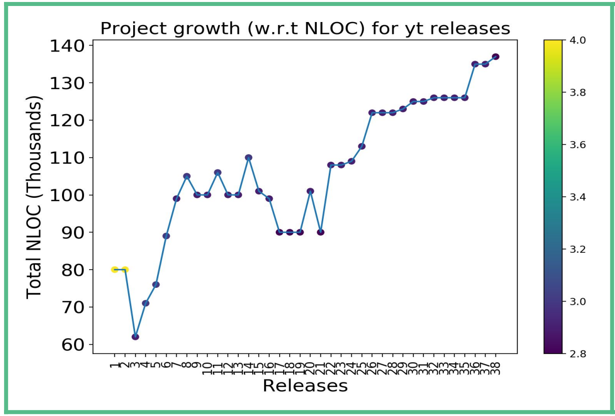
Color coding: Avg. CCN

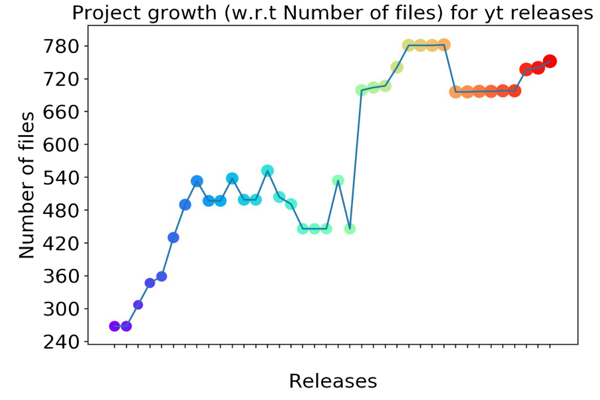
Color coding: Time

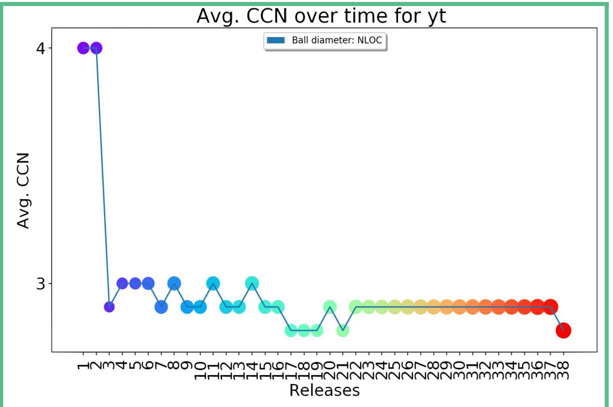




Project evolution: YT

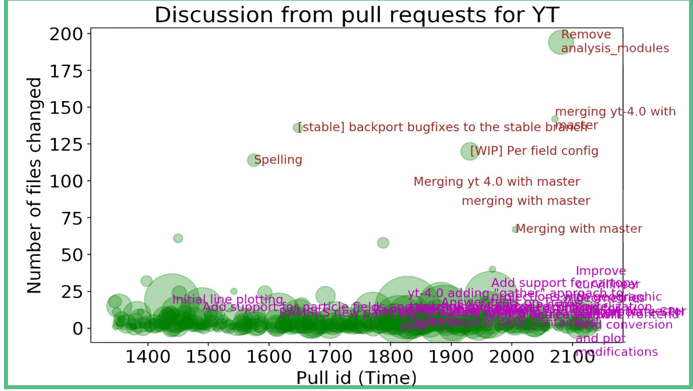


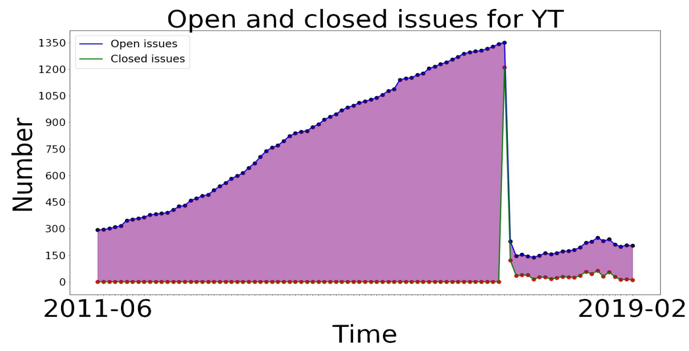




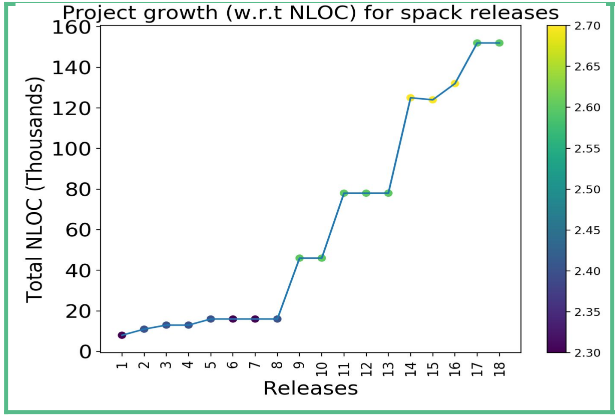
Color coding: Avg. CCN

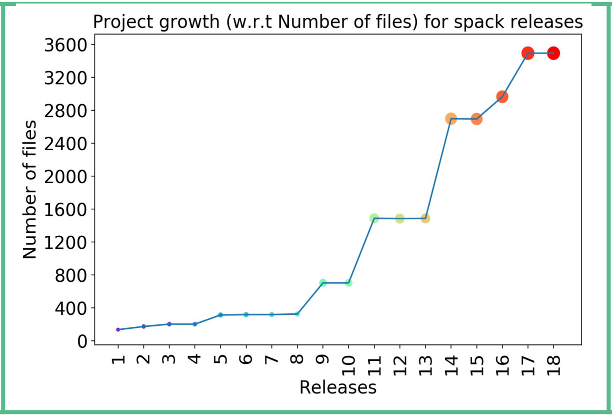
Color coding: Time

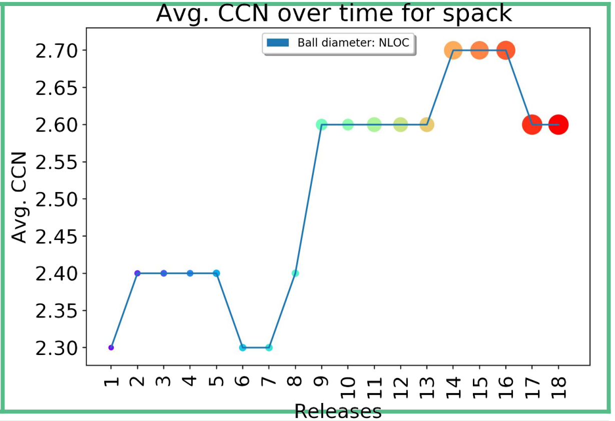




Project evolution: SPACK

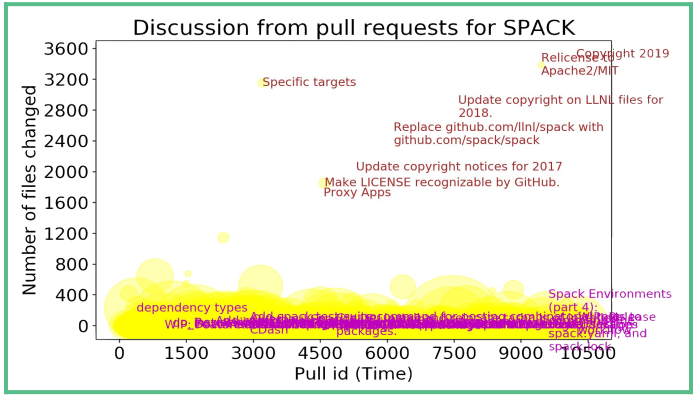


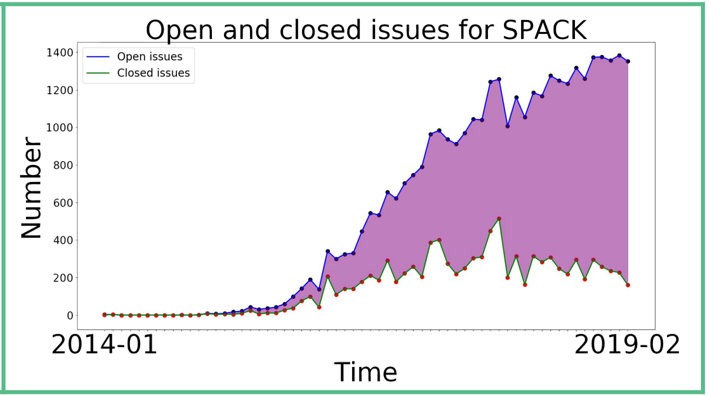




Color coding: Avg. CCN

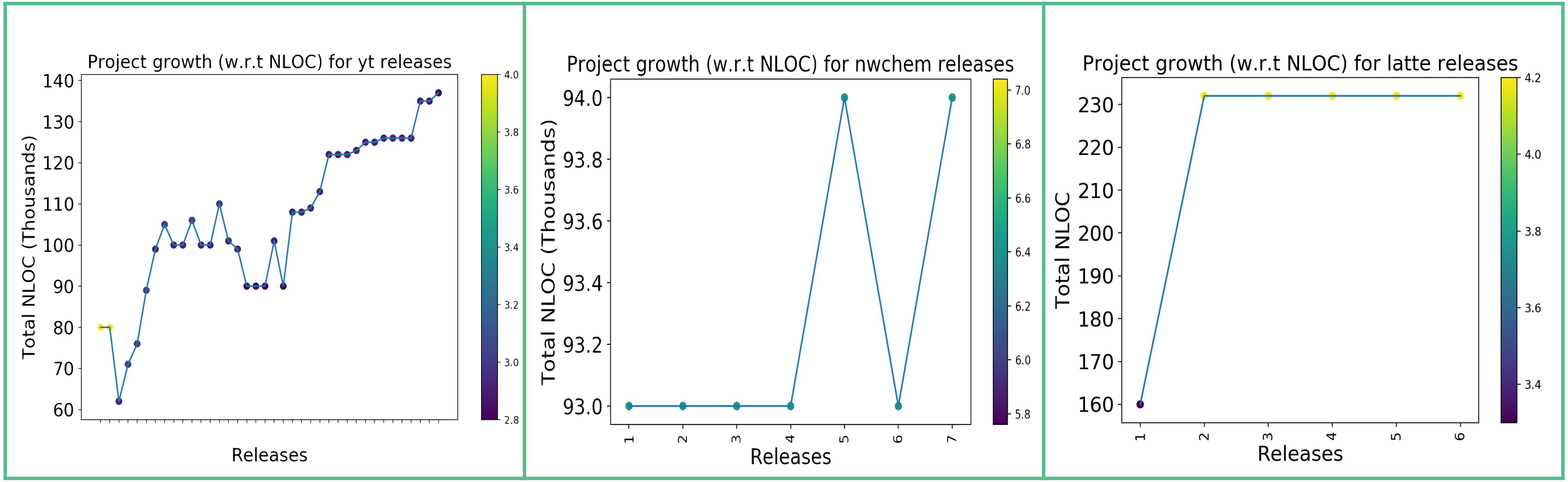
Color coding: Time





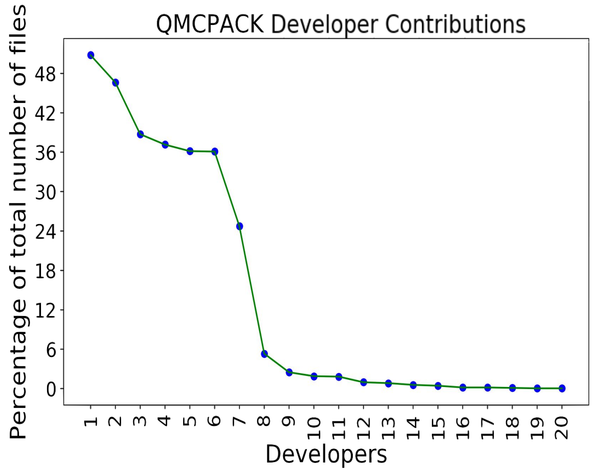
How is the project size changing over time?

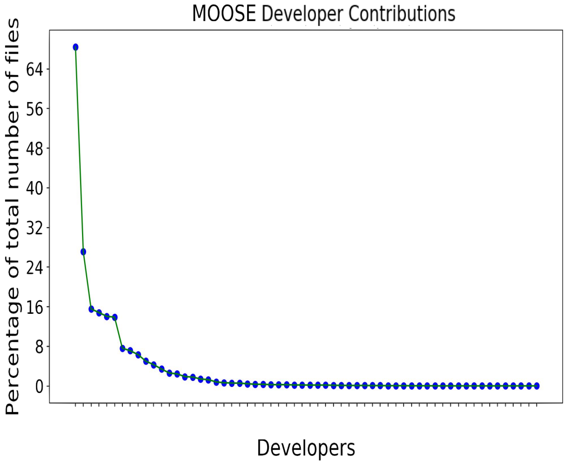
Number of lines of code (NLOC)

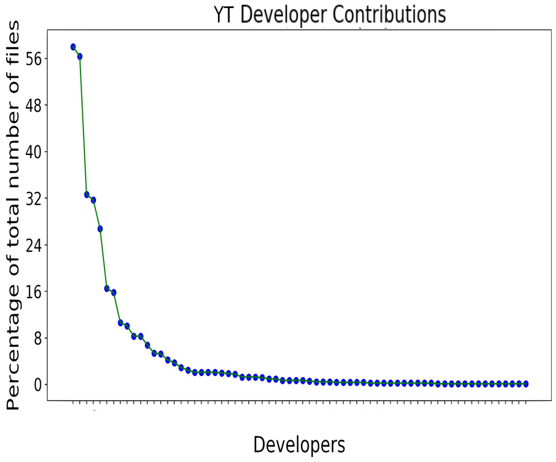


What is the project reliance on individual developers?

Percentage of total number of code files

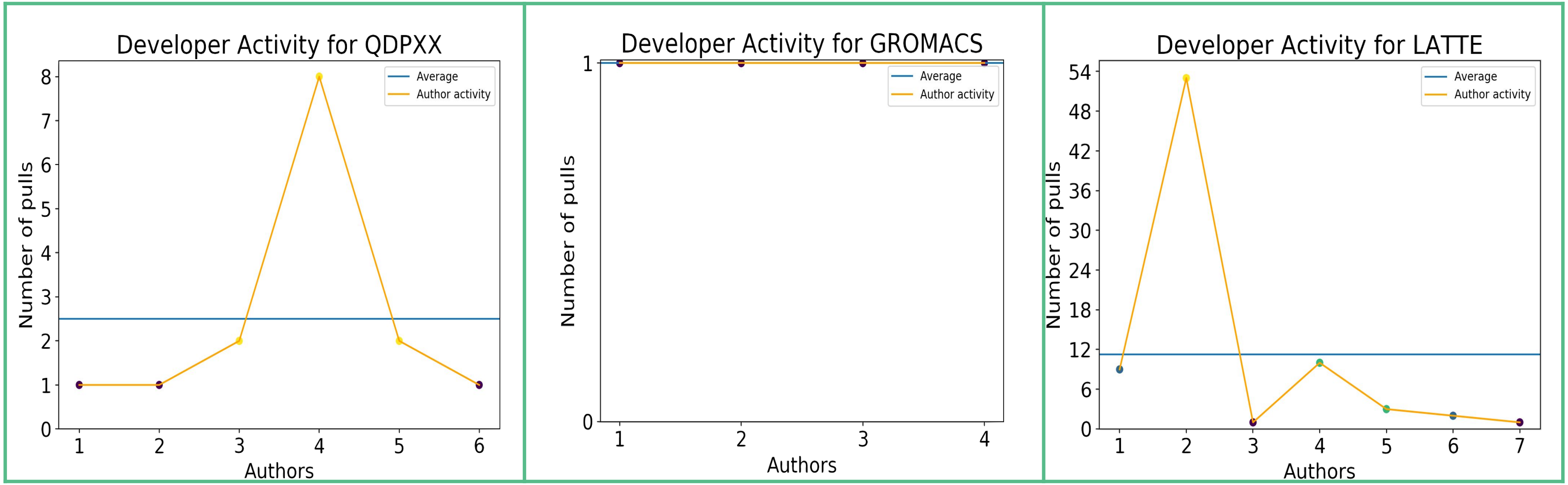






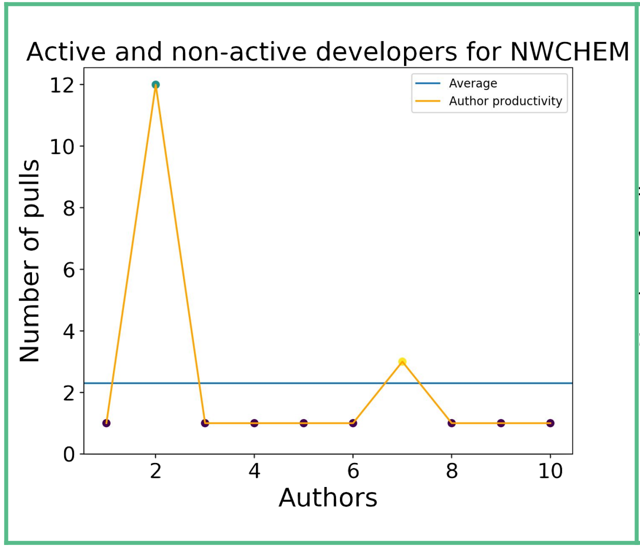
How active is the developer community?

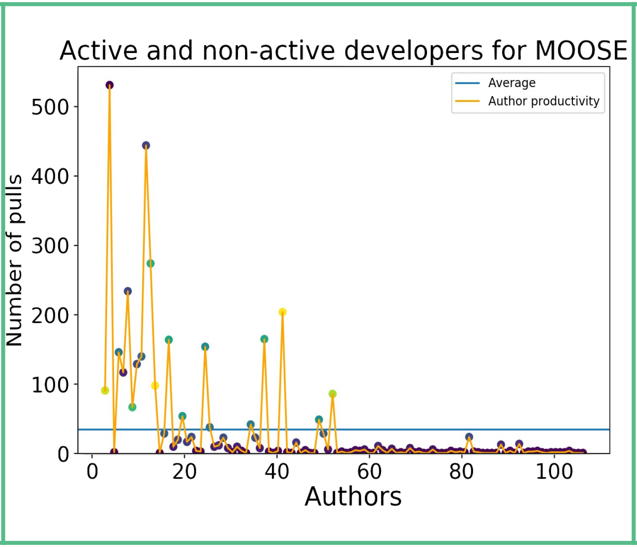
Developer activity based on pull requests

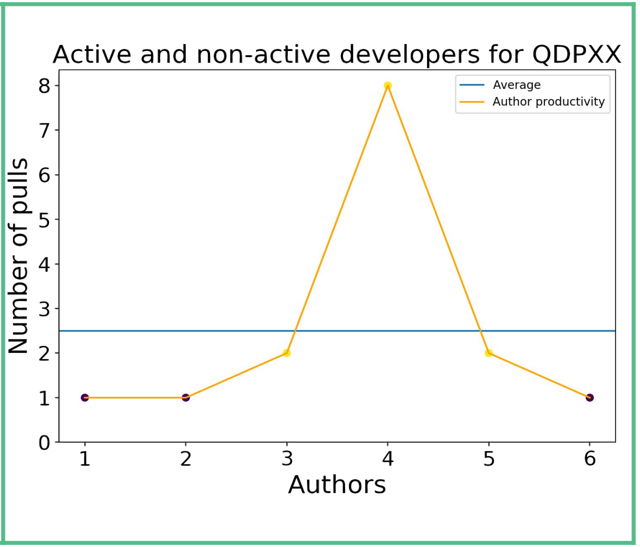


How active is the developer community?

Developer activity based on pull requests

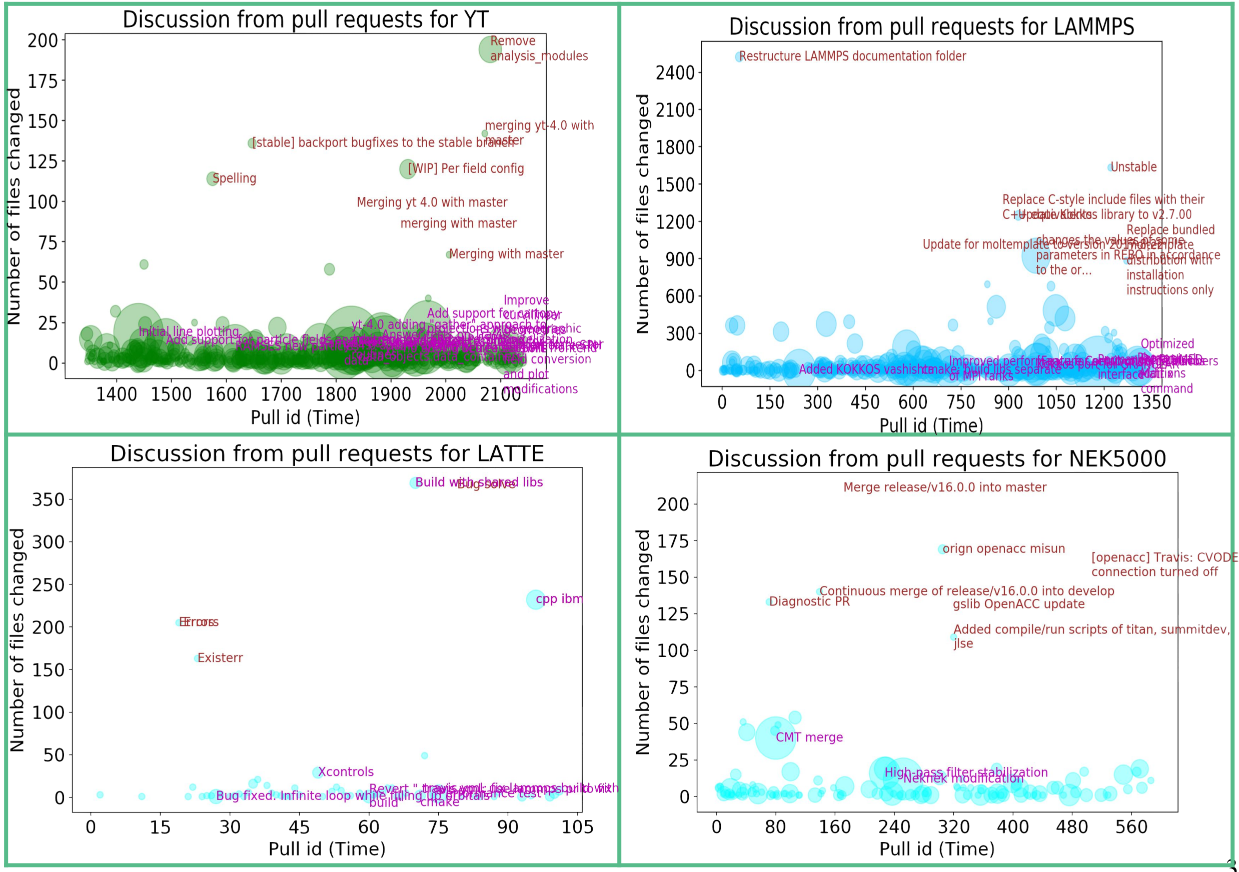






What topics dominate changes/discussion?

Pull requests


Discussion from pull requests for YT
Number of files changed
Pull id (Time)
Remove analysis_modules
merging yt-4.0 with master
[stable] backport bugfixes to the stable branch
[WIP] Per field config
Spelling
Merging yt 4.0 with master
merging with master
Merging with master
Improve
Add support for cartopy
yt-4.0 adding "gather" approach to
Initial line plotting
Add support for particle fields
conversion
and plot
modifications
Discussion from pull requests for LAMMPS
Number of files changed
Pull id (Time)
Restructure LAMMPS documentation folder
Unstable
Replace C-style include files with their
Update Kokkos library to v2.7.00
Replace bundled
Update for moltemplate to version 2017-
changes the values of some parameters in REBO in accordance to the or...
distribution with installation instructions only
Optimized
Added KOKKOS vashishta
cmake: build libs separate
Improved perfor
of MPI ranks
interface
command
Discussion from pull requests for LATTE
Number of files changed
Pull id (Time)
Build with shared libs
Bug solve
cpp ibm
Errors
Existerr
Xcontrols
Bug fixed. Infinite loop while filling up orbitals
Revert "
build
travis.yml: fix lammps build
cmake
Discussion from pull requests for NEK5000
Number of files changed
Pull id (Time)
Merge release/v16.0.0 into master
orign openacc misun
[openacc] Travis: CVODE connection turned off
Continuous merge of release/v16.0.0 into develop
Diagnostic PR
gslib OpenACC update
Added compile/run scripts of titan, summitdev, jlse
CMT merge
High-pass filter stabilization
Nekнek modification