ASE 2017
October 30 - November 3
Urbana-Champaign, Illinois

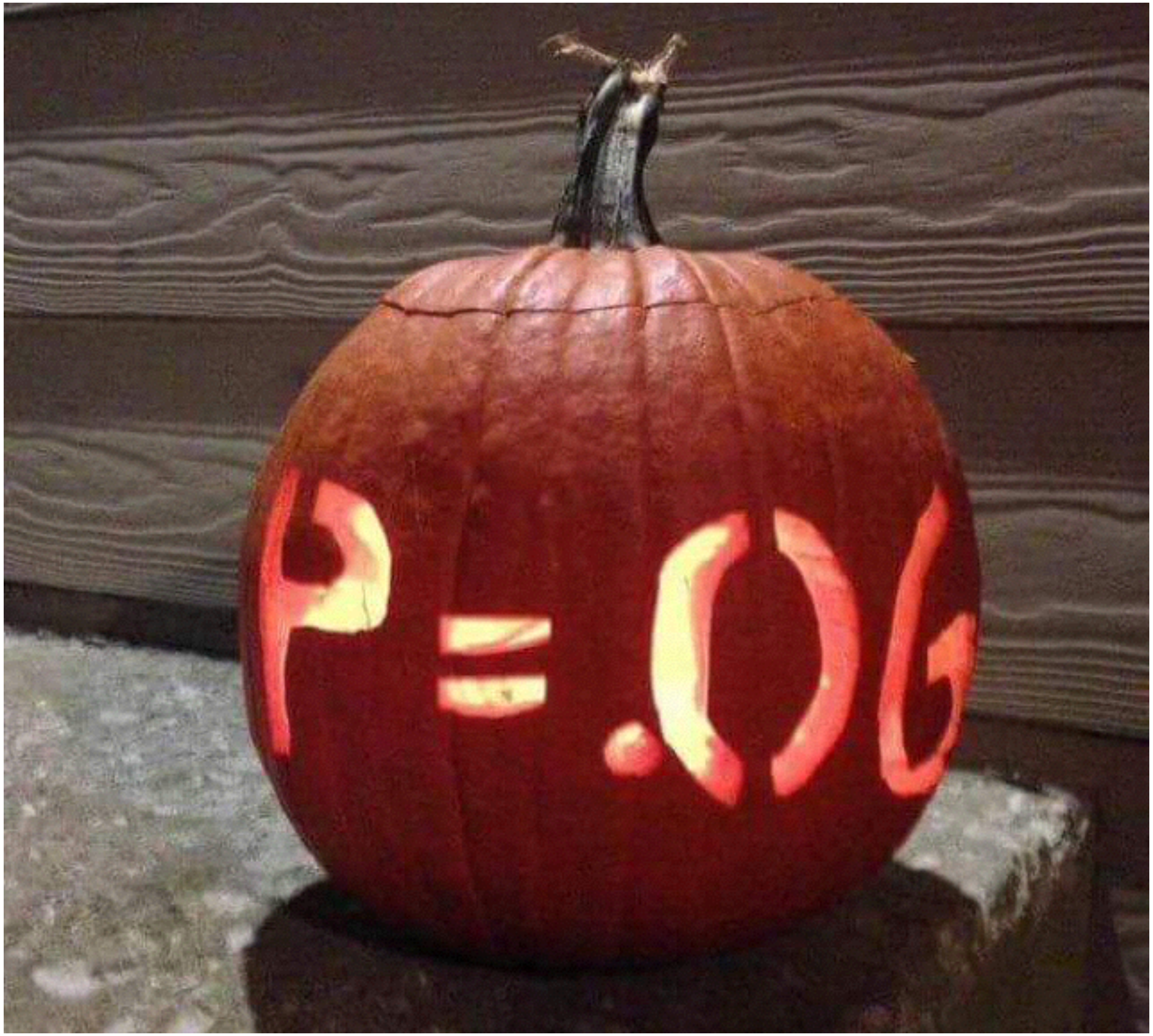
NSF
NSFC
National Natural Science Foundation of China

The Impact of Continuous Integration on Other Software Development Practices: A Large-Scale Empirical Study

Yangyang Zhao	Alexander Serebrenik	Yuming Zhou	Vladimir Filkov	Bogdan Vasilescu
Nanjing U	TU Eindhoven	Nanjing U	DECAL at UC Davis	STRUDEL at CMU
	@aserebrenik		@vfilkov	@b_vasilescu

STRUDEL SOCIO-TECHNICAL RESEARCH USING DATA EXCAVATION LAB
isr institute for SOFTWARE RESEARCH Carnegie Mellon University
NANJING UNIVERSITY 1902
TU/e Eindhoven University of Technology
UCDAVIS UNIVERSITY OF CALIFORNIA

Happy Halloween!



https://imgur.com/0TBJ9OW

Interventions are common in software engineering

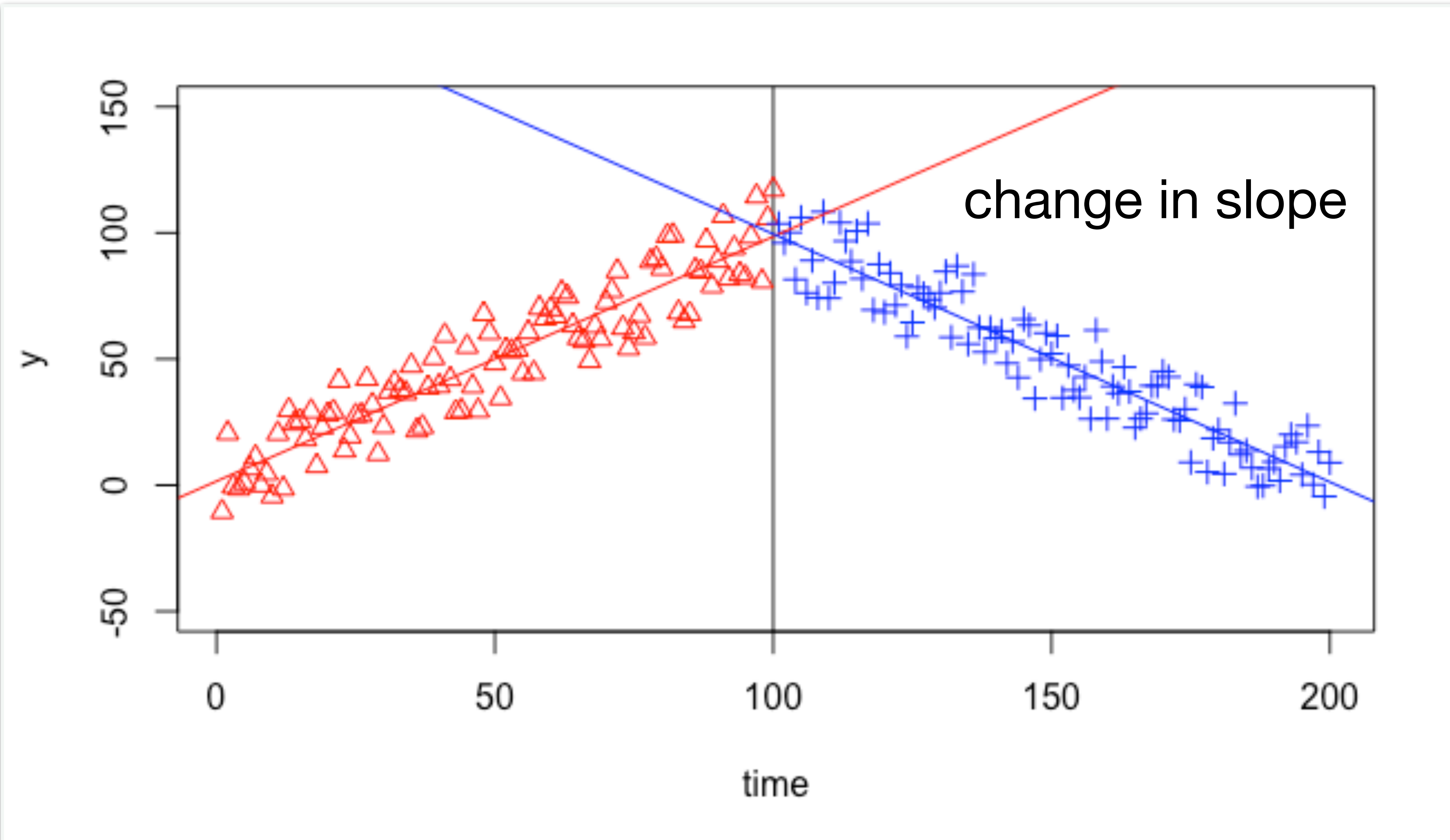
- SVN —> git
- push —> pull request
- ? —> continuous integration
- …

Interventions are common in software engineering

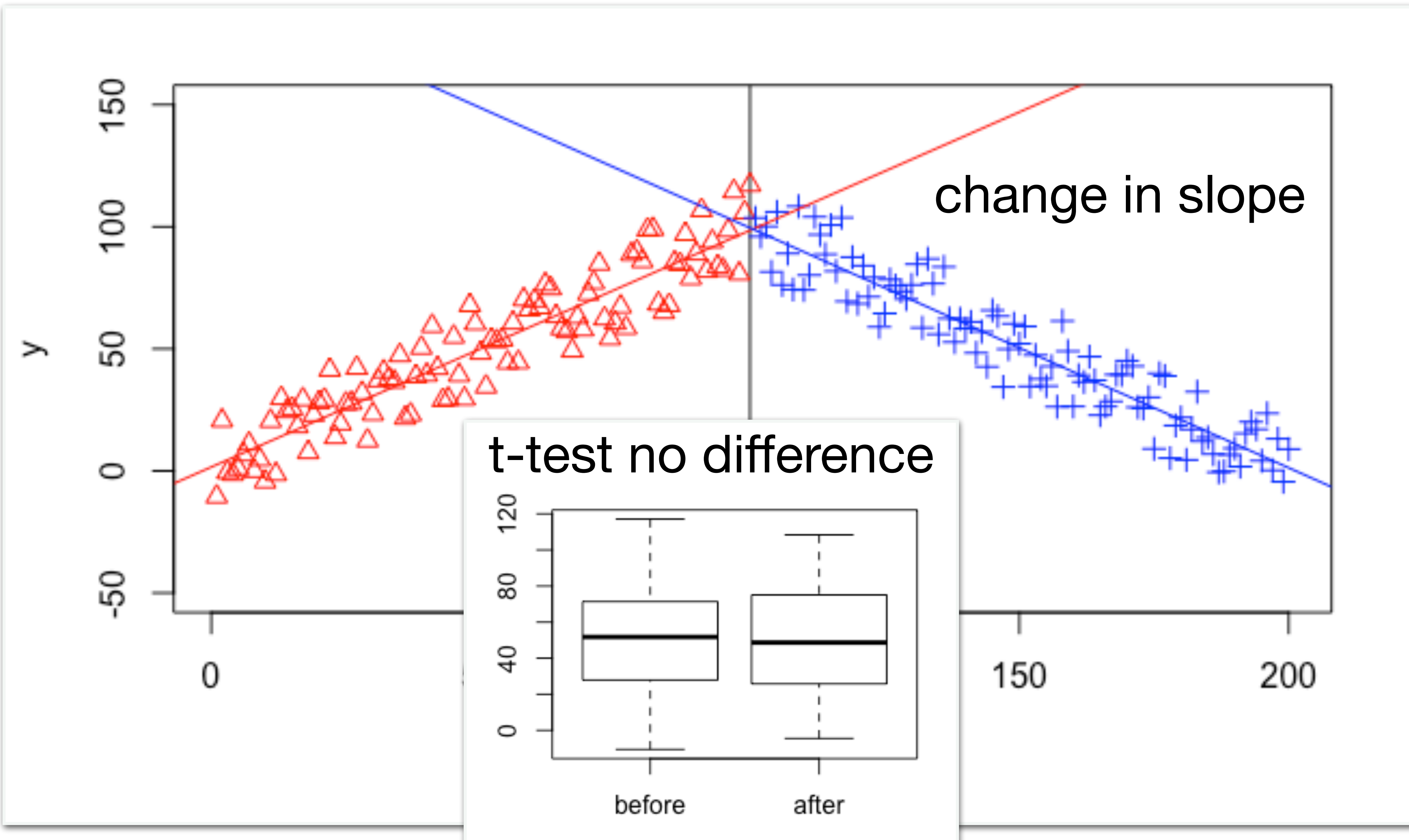
- SVN —> git
- push —> pull request
- ? —> continuous integration
- …

How to measure effects using trace data?

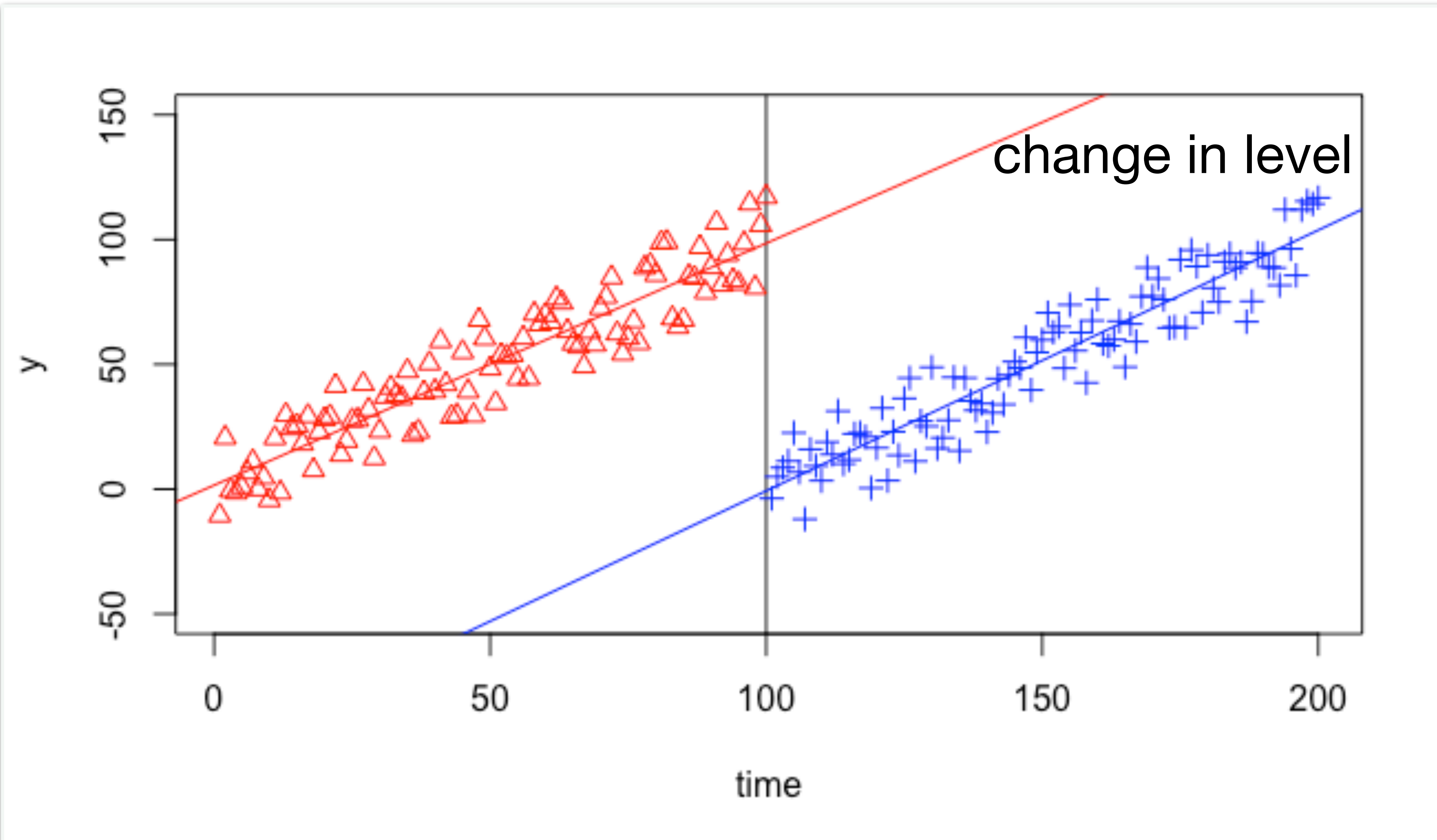
Evaluating the effects of an intervention: *before vs. after*



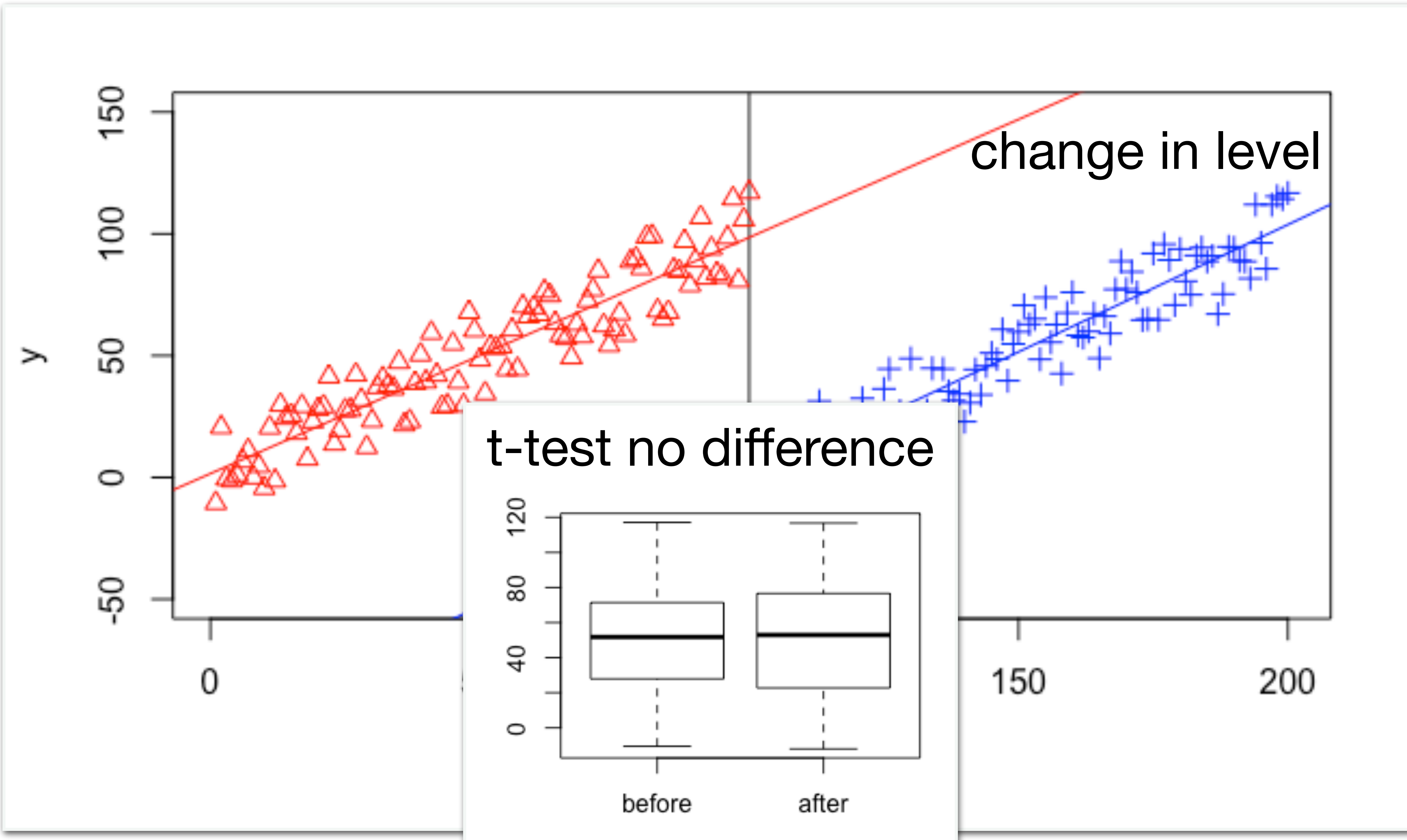
Evaluating the effects of an intervention: *before vs. after*



Evaluating the effects of an intervention: *before vs. after*



Evaluating the effects of an intervention: *before vs. after*

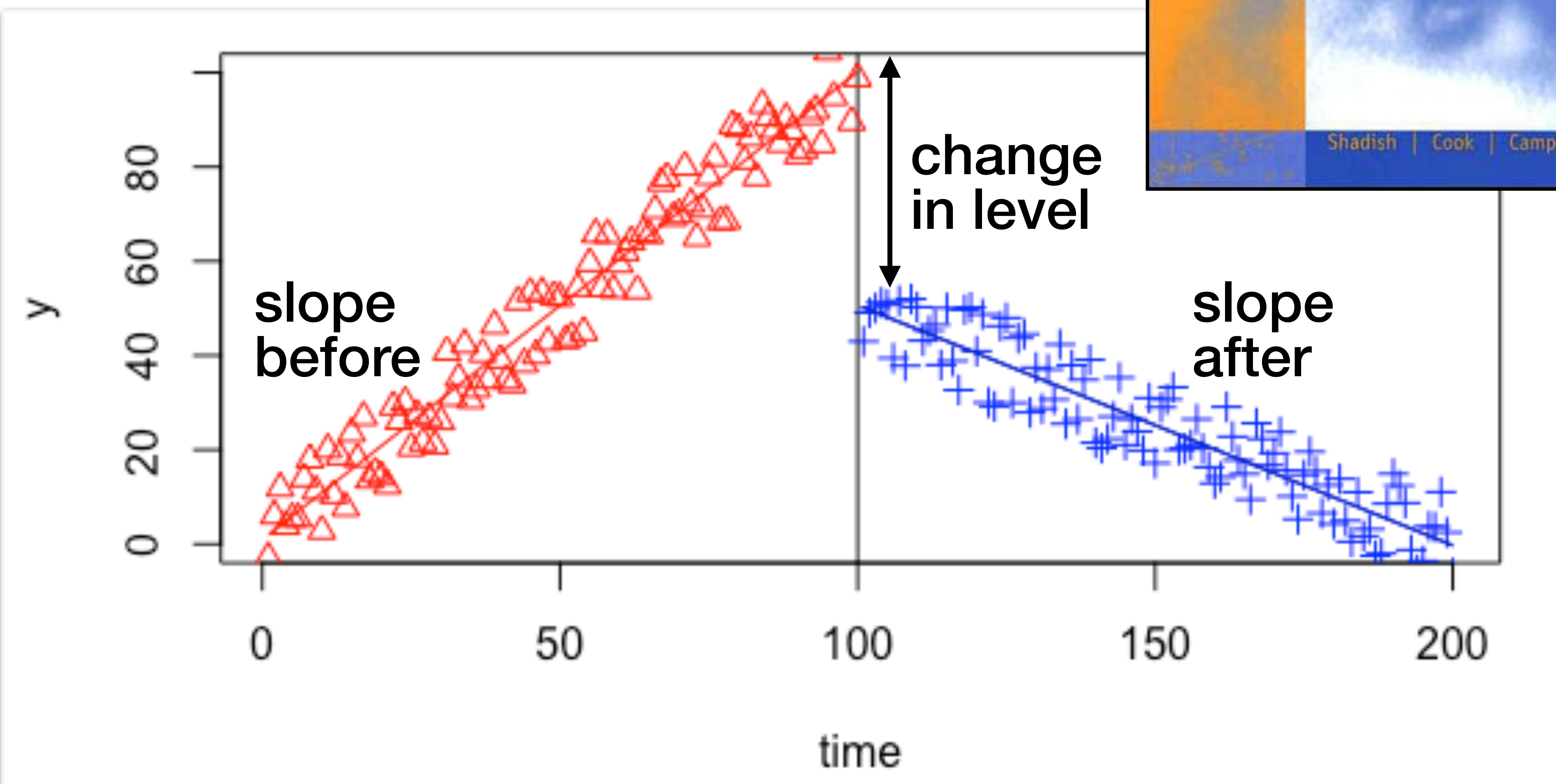


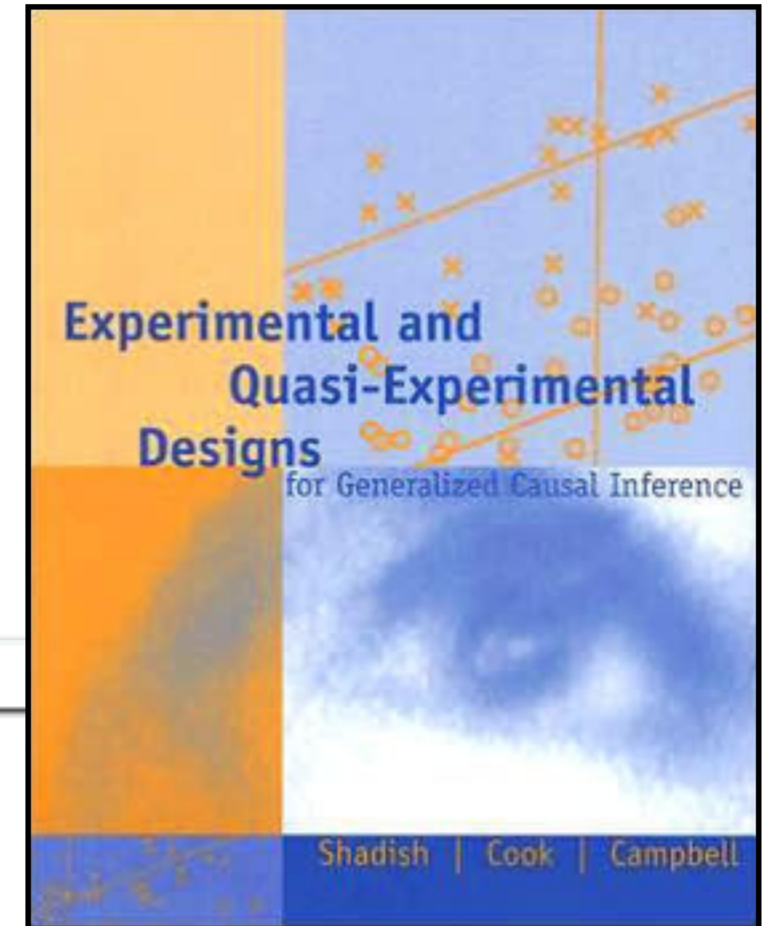
Today

Methodology to empirically study the effects of an intervention

(continuous integration)

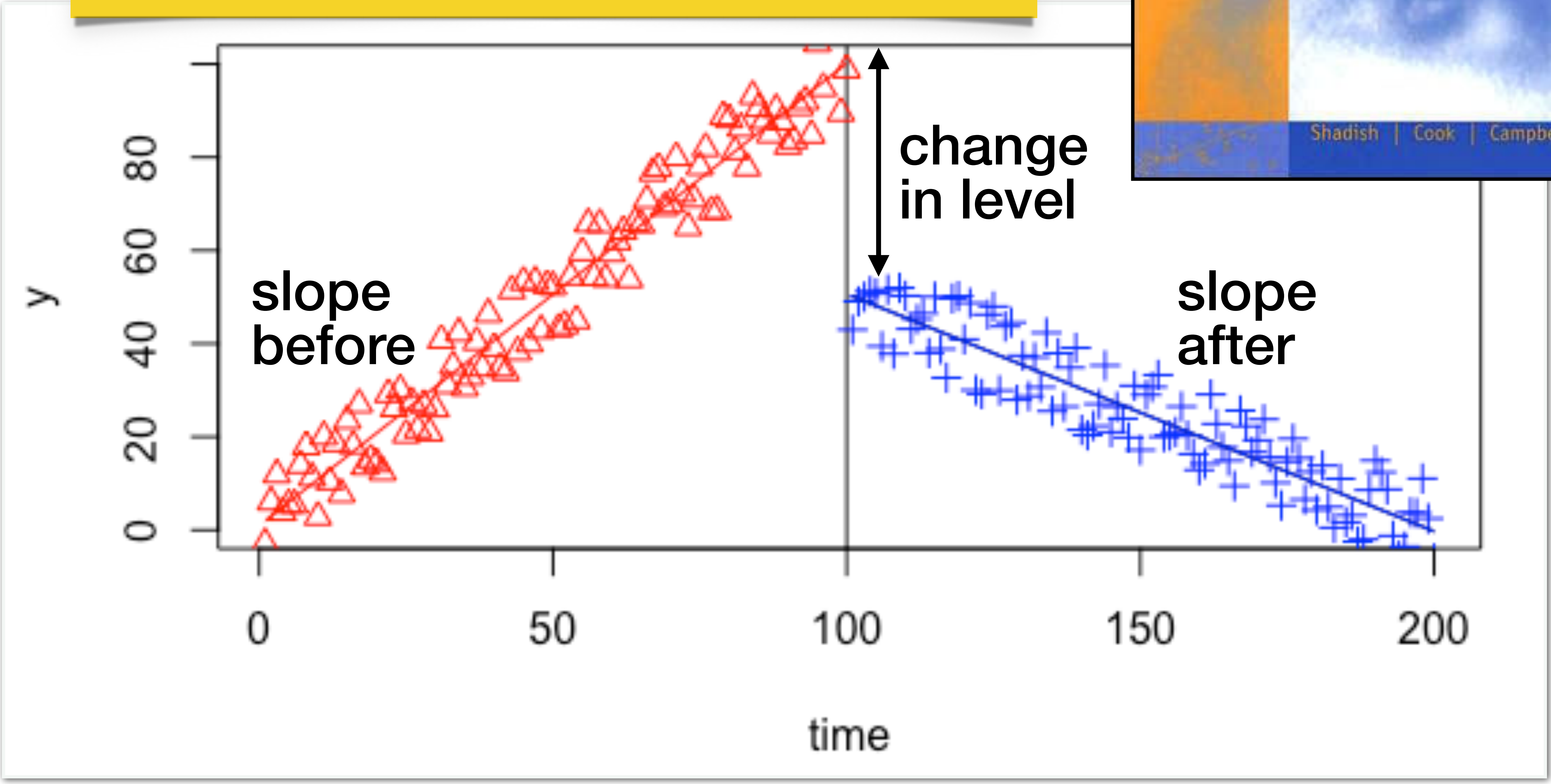
Interrupted time series

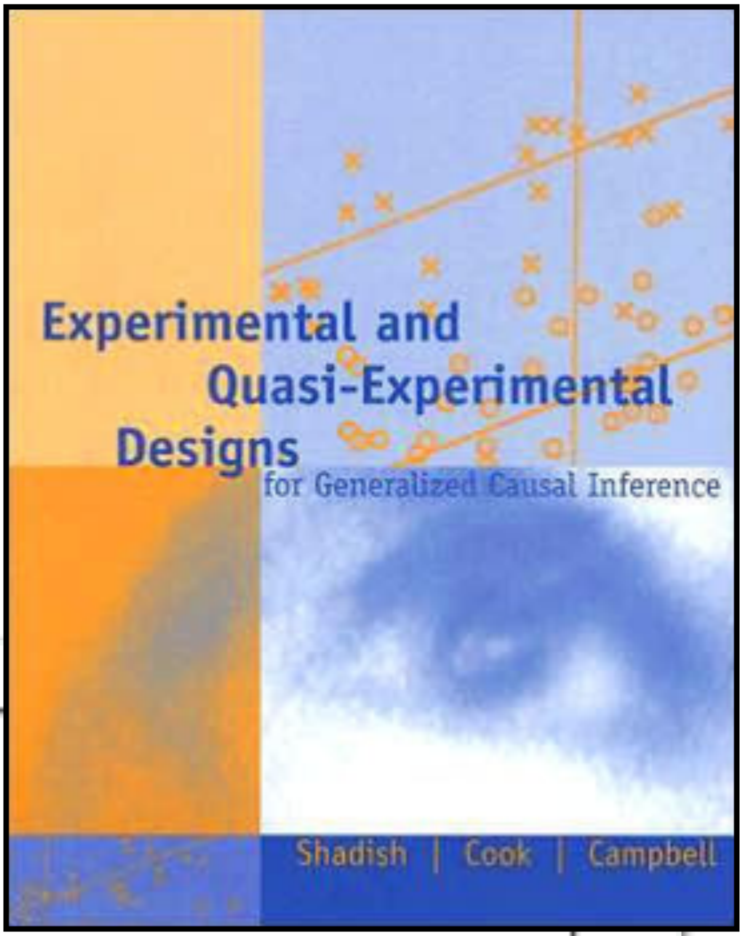


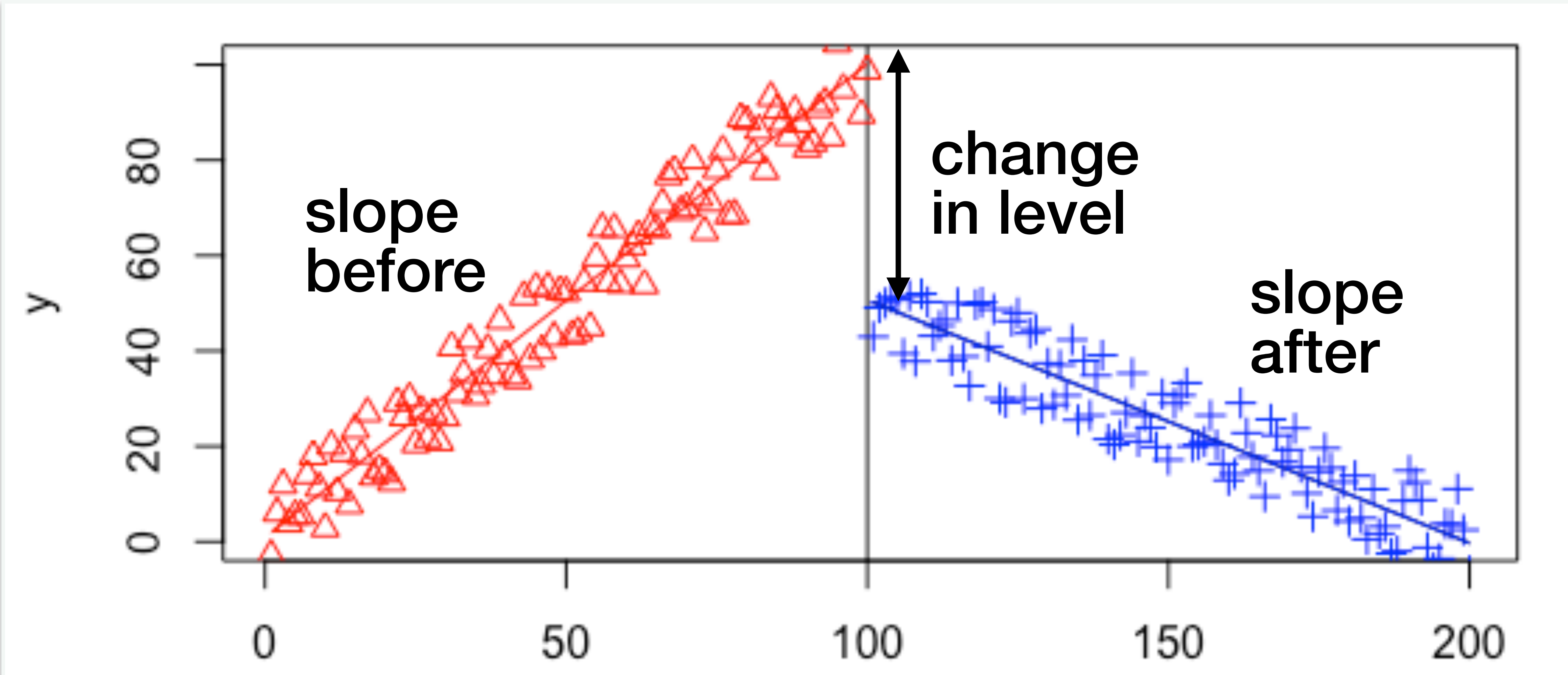


Interrupted time series

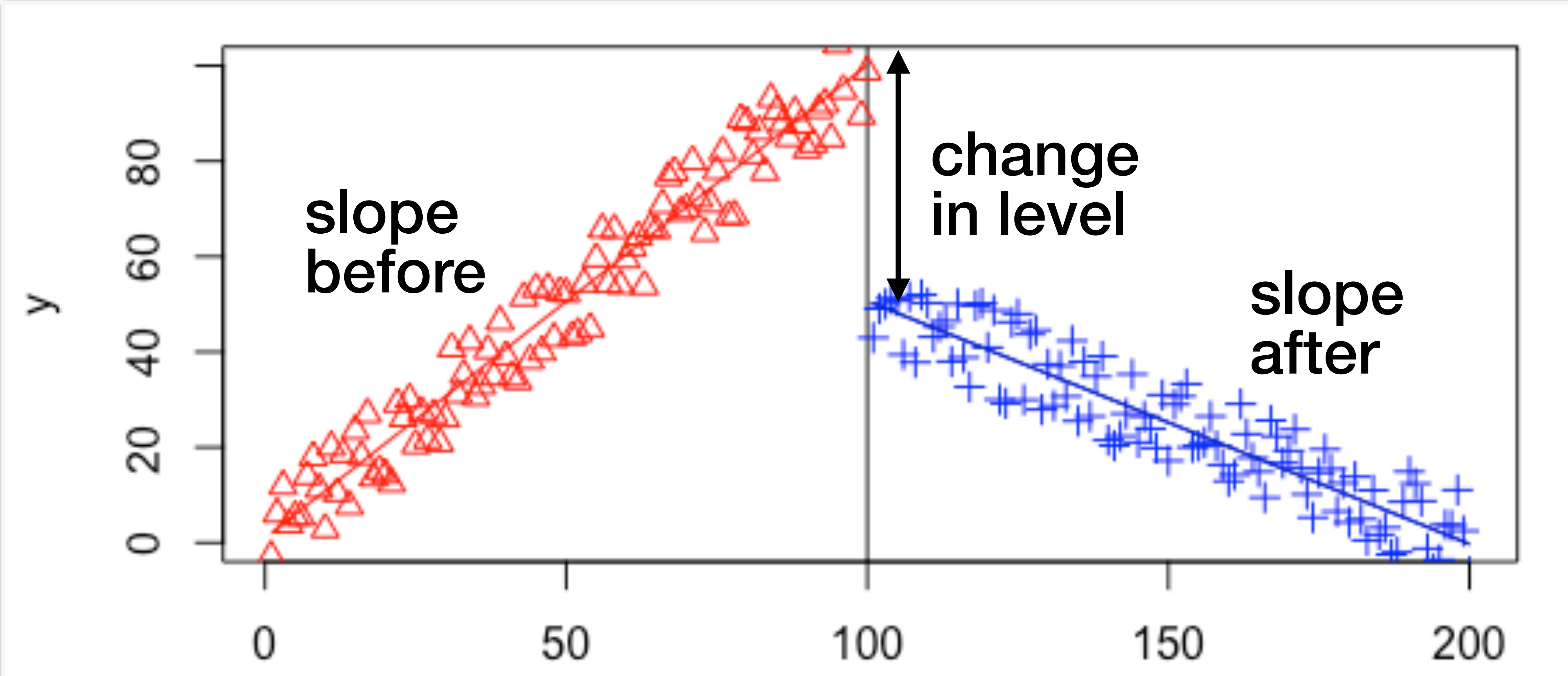
Multiple regression w/ controls for confounds



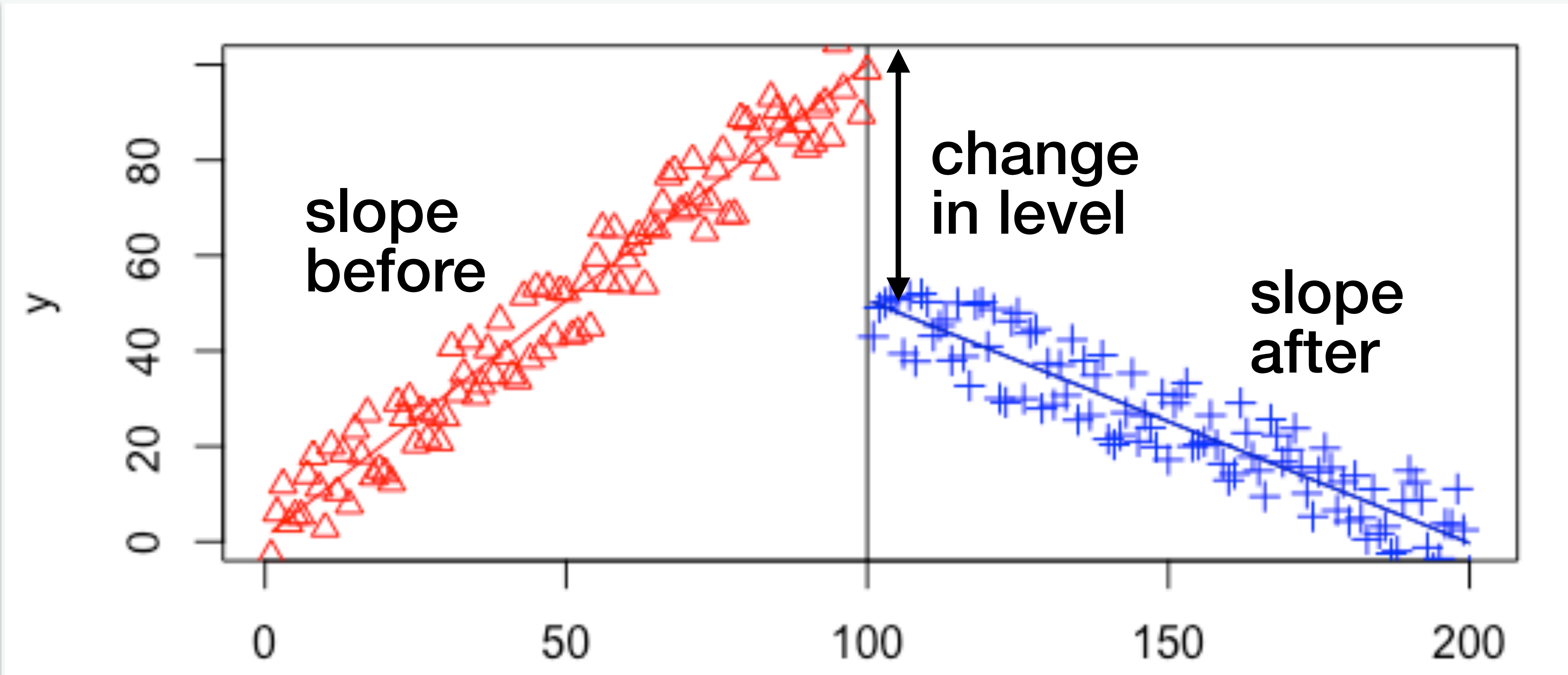




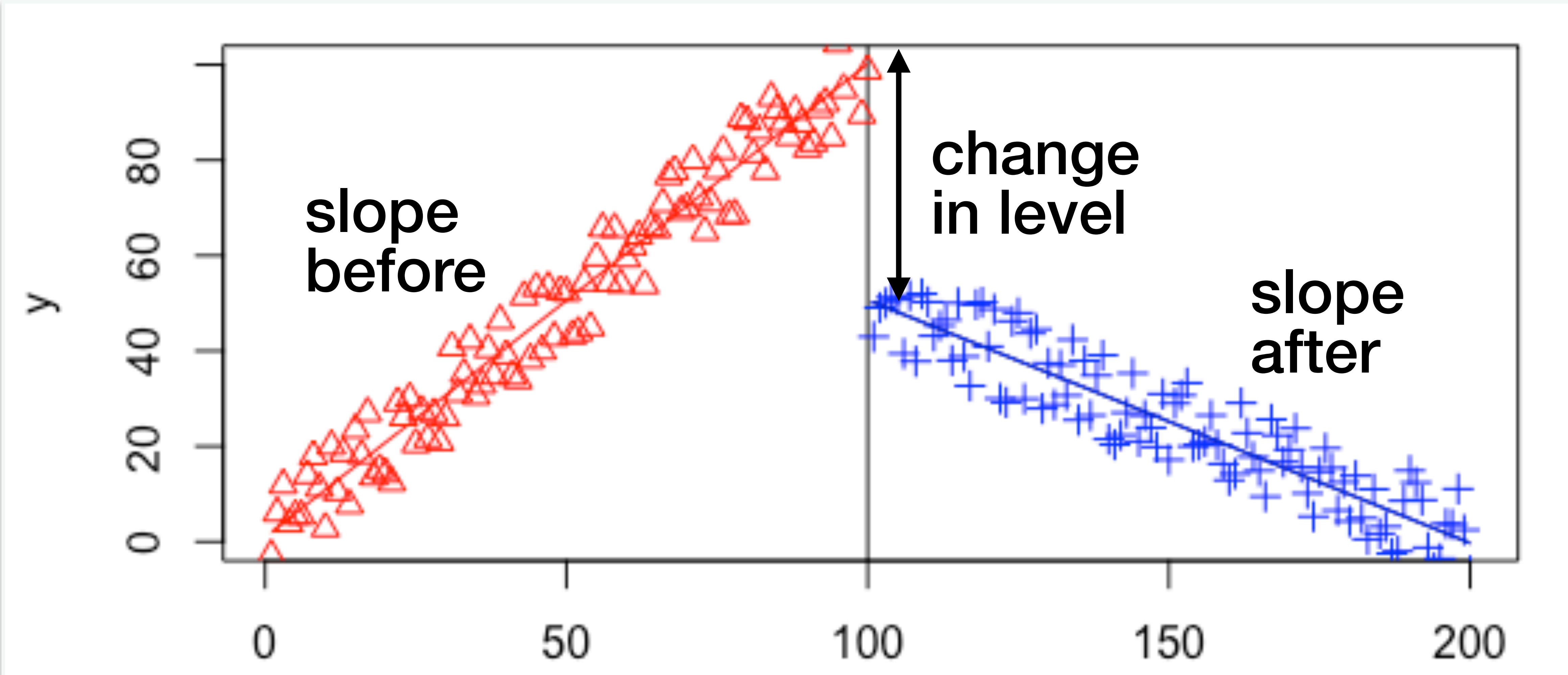
y
0
20
40
60
80
0
50
100
150
200
slope before
change in level
slope after



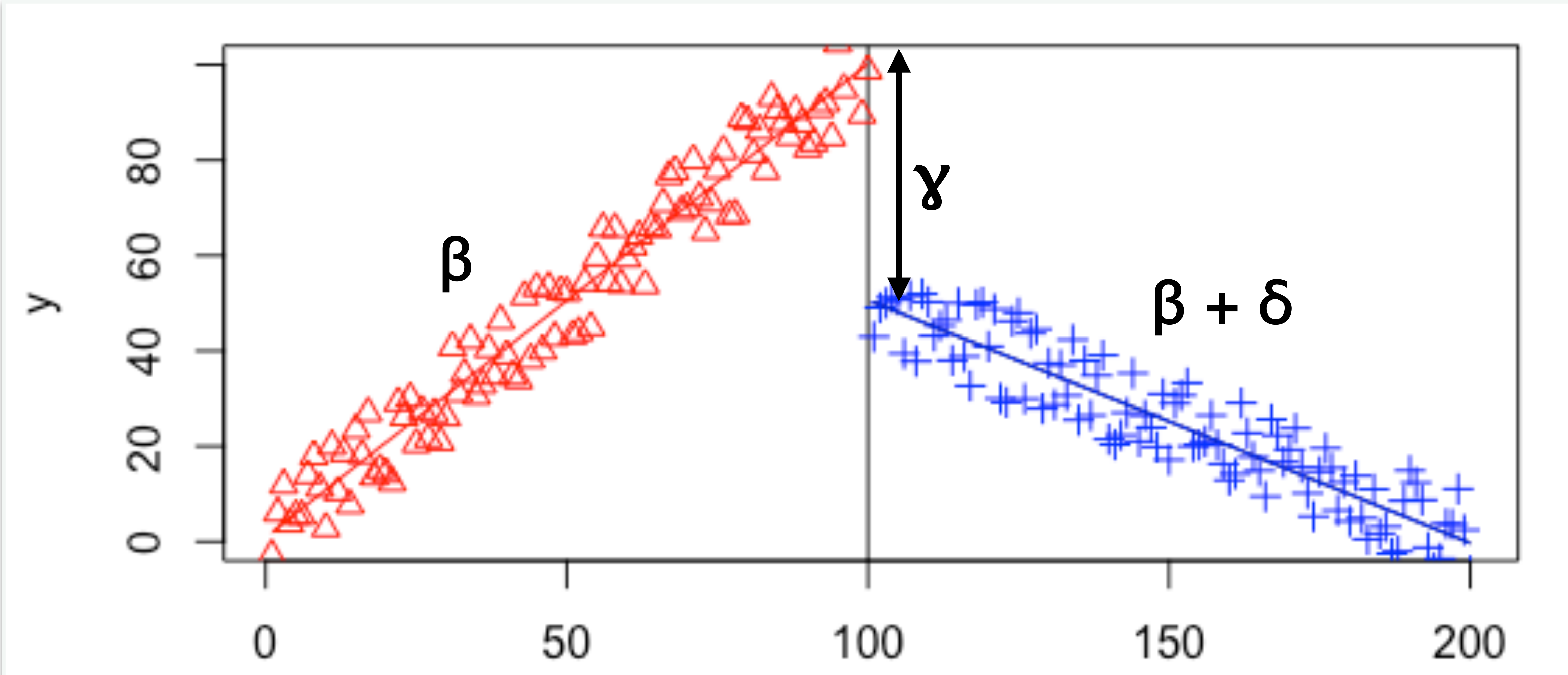
time: 1 2 3 … … … 100 101 102 … … … 200



time:	1	2	3 … … …	100	101	102	… … …	200
time after intervention:	0	0	0 … … …	0	1	2	… … …	100



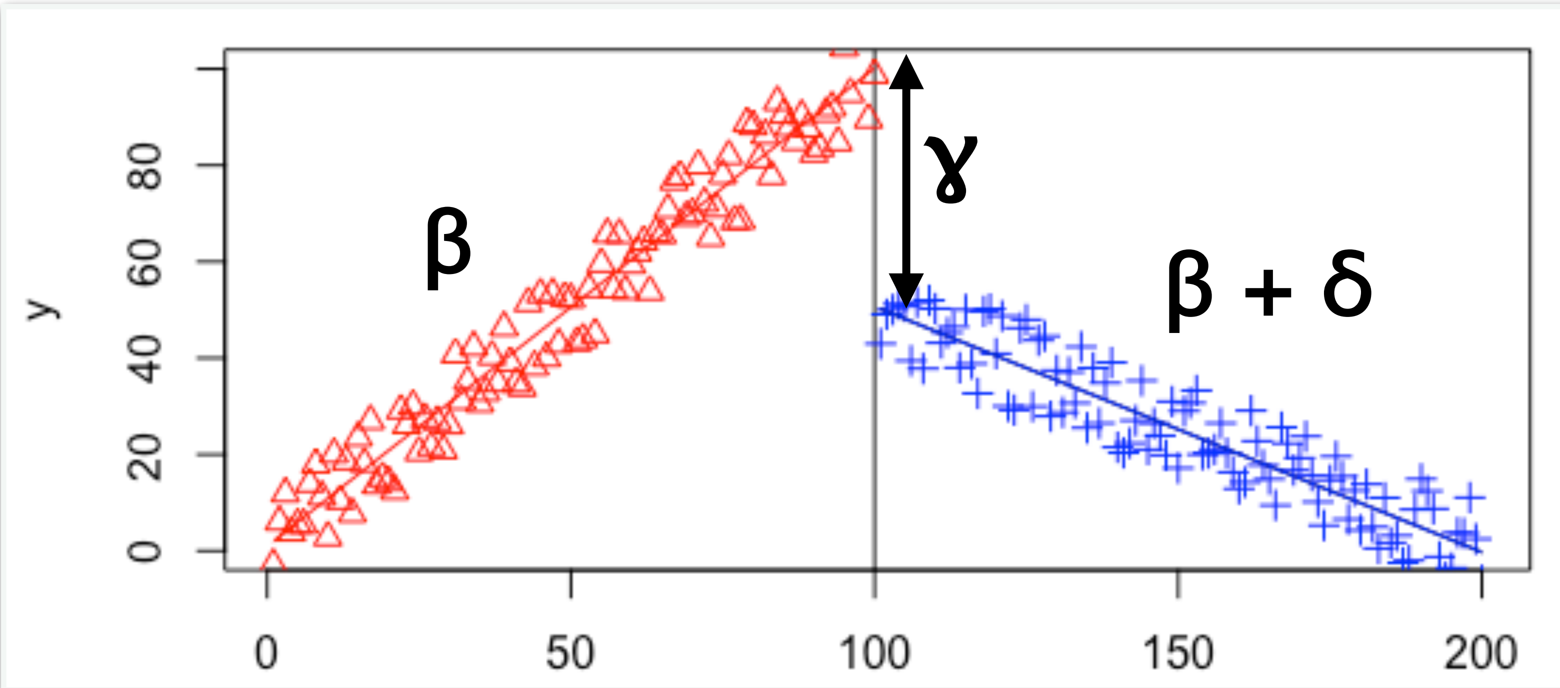
time:	1	2	3	… … …	100	101	102	… … …	200
time after intervention:	0	0	0	… … …	0	1	2	… … …	100
intervention:	F	F	F	… … …	T	T	T	… … …	T



time:	1	2	3	… … …	100	101	102	… … …	200
time after intervention:	0	0	0	… … …	0	1	2	… … …	100
intervention:	F	F	F	… … …	T	T	T	… … …	T

$$y_i = \alpha + \beta \cdot \text{time}_i + \gamma \cdot \text{intervention}_i + \delta \cdot \text{time\_after\_intervention}_i + \varepsilon_i$$

lm in R



$$y_i = \beta \cdot time_i + \gamma \cdot intervention_i + \delta \cdot time\_after\_intervention_i + \varepsilon_i$$

- β ~ 1
- γ ~ -50
- β + δ ~ -0.5

	Dependent variable:
	y
time	0.991^{***}
intervention	-48.678^{***}
time_after_intervention	-1.500^{***}
Constant	1.007
Observations	200
R^2	0.967
Adjusted R^2	0.967
Residual Std. Error	4.844 (df = 196)
F Statistic	$1{,}924.910^{***}$ (df = 3; 196)
Note:	$^{*}p<0.1$; $^{**}p<0.05$; $^{***}p<0.01$

Effects of adopting Travis CI

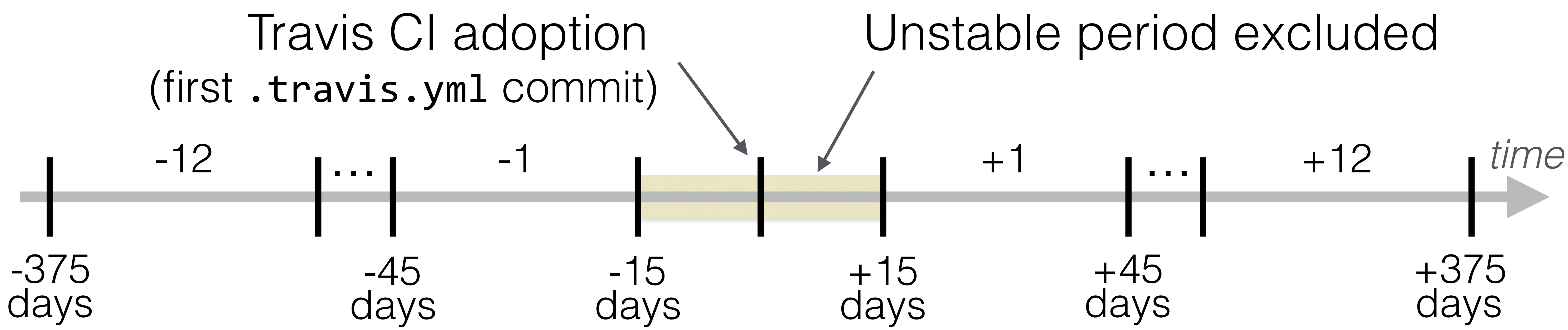
Why CI?

Lots of folklore, e.g., Martin Fowler:

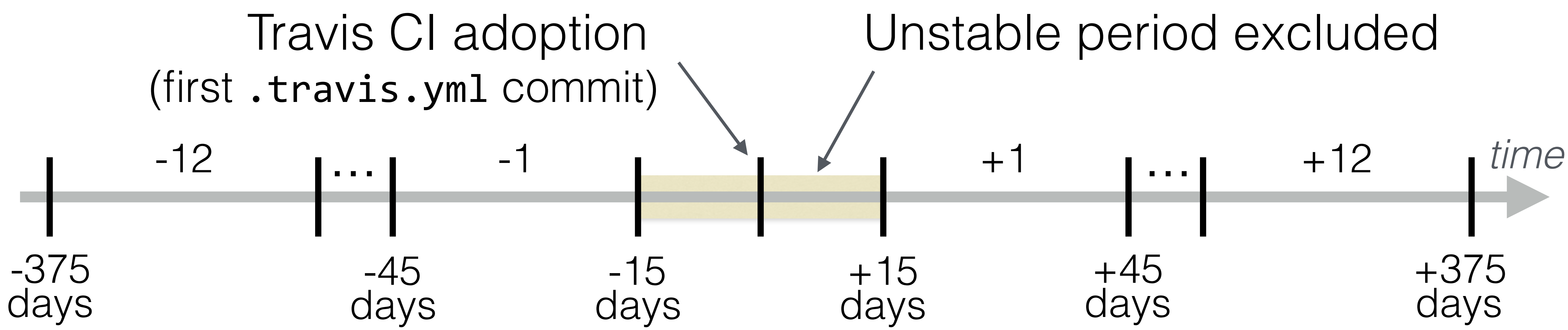
- Everyone Commits To the Mainline Every Day
- Fix Broken Builds Immediately
- Keep the Build Fast
- …

https://martinfowler.com/articles/originalContinuousIntegration.html

Adoption of Travis CI



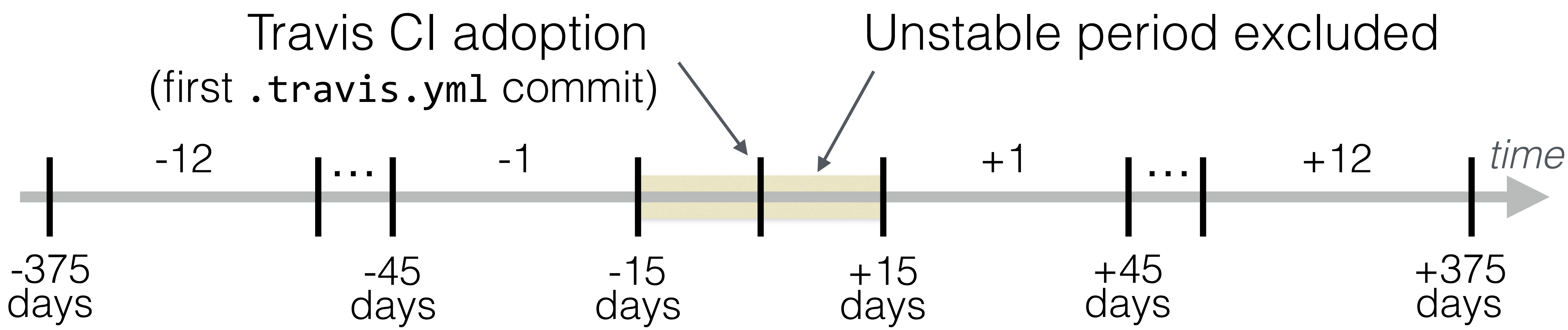
Adoption of Travis CI



Starting sample:

165,549 GitHub projects using Travis

Adoption of Travis CI



Starting sample:

165,549 GitHub projects using Travis

24 active periods

x

7 programming languages

RQs

More frequent
commits

Created by Ramesha
from Noun Project

Impact on
automated
testing?

Created by Misha
from Noun Project

Smaller code
changes

Created by Barracuda
from Noun Project

More issues and
pull requests closed

Created by Genius Icons
from Noun Project

Quick pull requests
resolution

Churn

	Churn in non-merge commits	Churn in merge commits
Intercept (α)	1,336***	-1,297**
log(TotalCommits)	0,529**	1,113**
AgeAtTravis	-0,003*	-0,005**
log(NumAuthors)	-0,233**	-0,522**
time (β)	-0,007	-0,012*
interventionTrue (ɣ)	0,071	0,220**
time_after_intervention (δ)	-0,009	-0,022**

Churn

	Churn in non-merge commits	Churn in merge commits
Intercept (α)	1,336***	-1,297**
log(TotalCommits)	0,529**	1,113**
AgeAtTravis	-0,003*	-0,005**
log(NumAuthors)	-0,233**	-0,522**
time (β)	-0,007	-0,012*
interventionTrue (ɣ)	0,071	0,220**
time_after_intervention (δ)	-0,009	-0,022**

Control variables

Churn

	Churn in non-merge commits	Churn in merge commits
Intercept (α)	1,336***	-1,297**
log(TotalCommits)	0,529**	1,113**
AgeAtTravis	-0,003*	-0,005**
log(NumAuthors)	-0,233**	-0,522**
time (β)	-0,007	-0,012*
interventionTrue (ɣ)	0,071	0,220**
time_after_intervention (δ)	-0,009	-0,022**

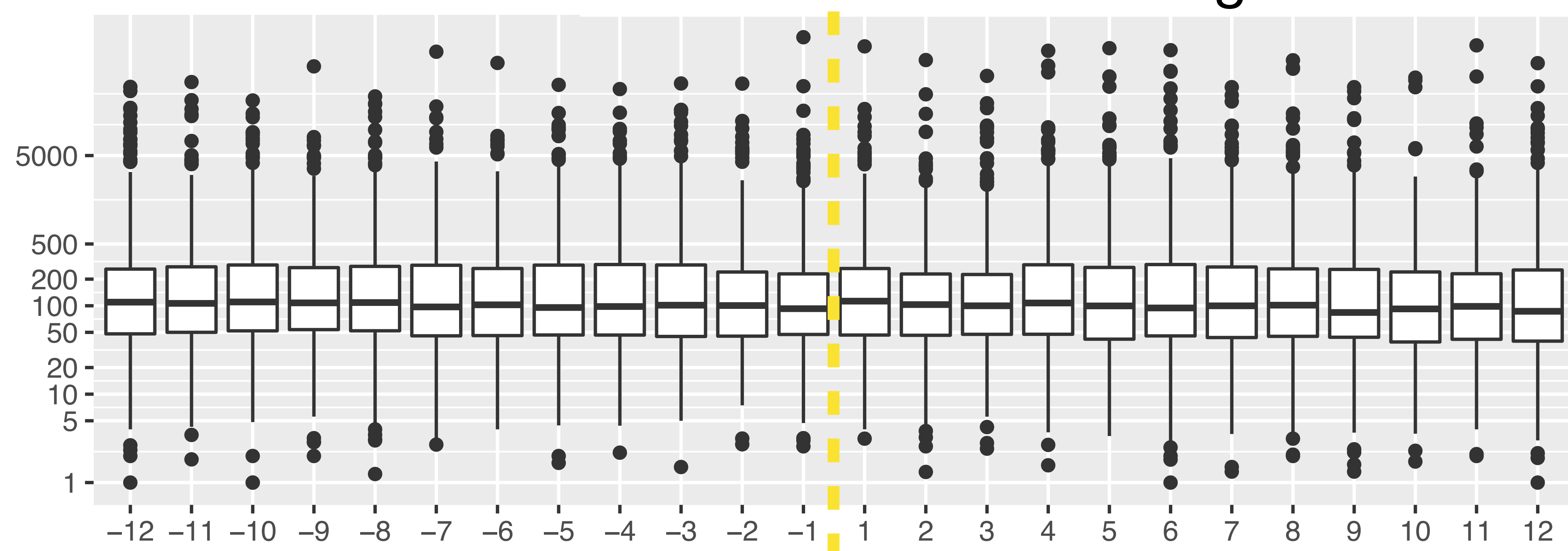
Control variables

n.s.

Churn in **non-merge commits** is **not affected** by time or Travis CI

Churn

Churn in non-merge commits



time (β)	-0,007	-0,012*
interventionTrue (γ)	n.s.	0,220**
time_after_intervention (δ)	-0,009	-0,022**

Churn in **non-merge commits** is **not affected** by time or Travis CI

Churn

	Churn in non-merge commits	Churn in merge commits
Intercept (α)	1,336***	-1,297**
log(TotalCommits)	0,529**	1,113**
AgeAtTravis	-0,003*	-0,005**
log(NumAuthors)	-0,233**	-0,522**
time (β)	-0,007	-0,012*
interventionTrue (ɣ)	0,071	**0,220****
time_after_intervention (δ)	-0,009	-0,022**

Control variables

n.s.

Churn in **non-merge commits** is **not affected** by time or Travis CI

Discontinuity in merge com.: preparation for transition, clean-up

Churn

	Churn in non-merge commits	Churn in merge commits
Intercept (α)	1,336***	-1,297**
log(TotalCommits)	0,529**	1,113**
AgeAtTravis	-0,003*	-0,005**
log(NumAuthors)	-0,233**	-0,522**
time (β)	-0,007	**-0,012***
interventionTrue (ɣ)	0,071	0,220**
time_after_intervention (δ)	-0,009	**-0,022****

Control variables

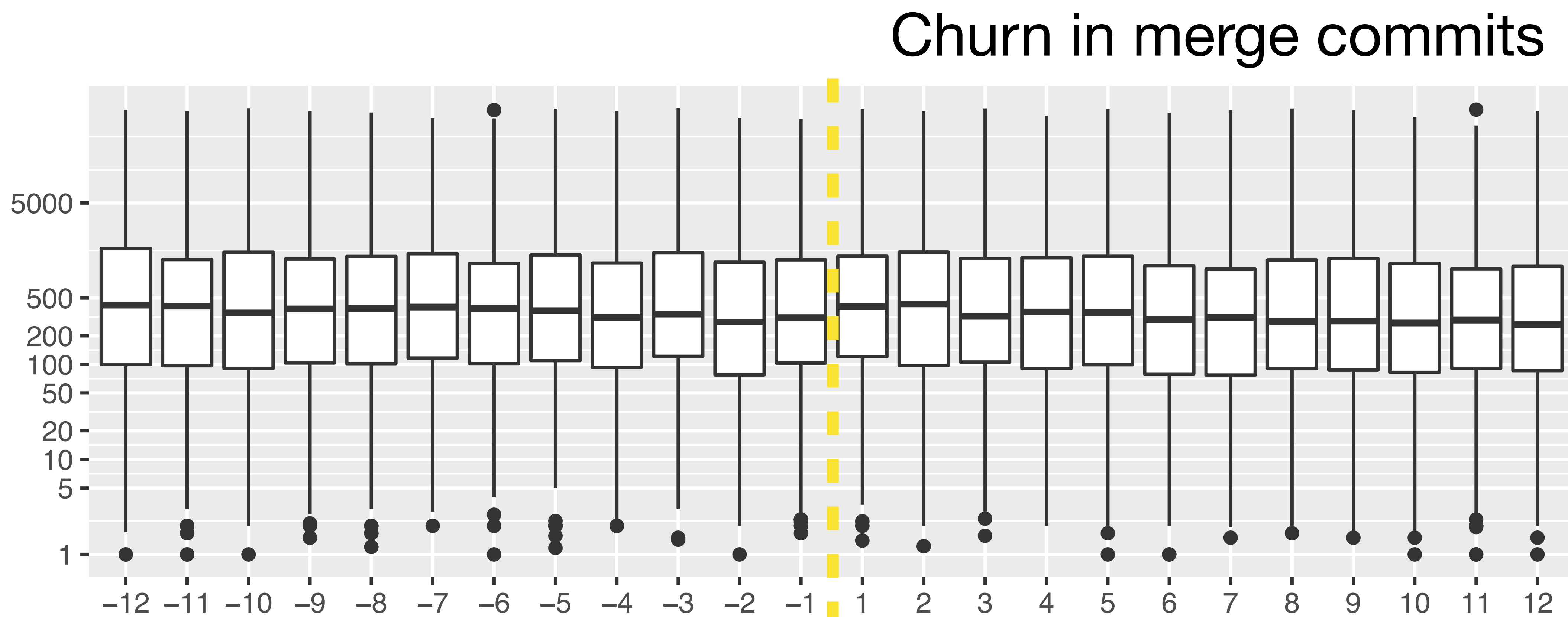
n.s.

Churn in **non-merge commits** is **not affected** by time or Travis CI

Discontinuity in merge com.: preparation for transition, clean-up

Decrease in churn in **merge commits** is **amplified** by Travis CI

Churn



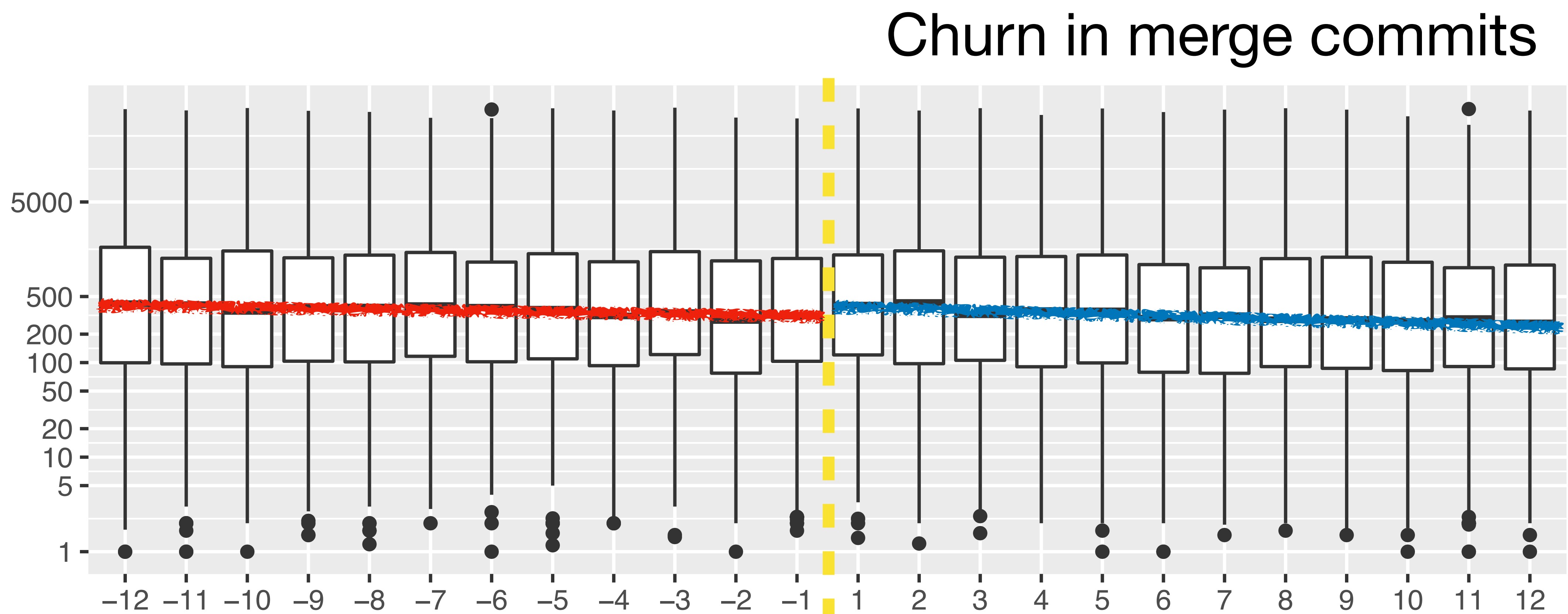
time (β)	-0,007	**-0,012***
interventionTrue (ɣ)	0,071 n.s.	0,220**
time_after_intervention (δ)	-0,009	**-0,022****

Churn in **non-merge commits** is **not affected** by time or Travis CI

Discontinuity in merge com.: preparation for transition, clean-up

Decrease in churn in **merge commits** is **amplified** by Travis CI

Churn



time (β)	-0,007	**-0,012***
interventionTrue (ɣ)	0,071 n.s.	0,220**
time_after_intervention (δ)	-0,009	**-0,022****

Churn in **non-merge commits** is **not affected** by time or Travis CI

Discontinuity in merge com.: preparation for transition, clean-up

Decrease in churn in **merge commits** is **amplified** by Travis CI

Triangulation: user survey

55

introduced Travis to their projects

Discontinuity in merge commits: preparation for transition, clean-up

R25: “contributors couldn’t be trusted to run test suite on their own”

R38: Travis as “a part of automated package/release effort”

Decrease in churn in **merge commits** is **amplified** by Travis CI

R4: “commits became smaller and more frequent, to check the build; pull requests became easier to check”

Closed PRs

Among others:

- On average, more PRs are being closed per unit time after adopting Travis CI

ESEC/FSE '15

Quality and Productivity Outcomes Relating to Continuous Integration in GitHub

Bogdan Vasilescu[†*], Yue Yu[‡†*], Huaimin Wang[‡], Premkumar Devanbu[†], Vladimir Filkov[†]

[†]Department of Computer Science
University of California, Davis
Davis, CA 95616, USA
{vasilescu, ptdevanbu, vfilkov}@ucdavis.edu

[‡]College of Computer
National University of Defense Technology
Changsha, 410073, China
{yuyue, hmwang}@nudt.edu.cn

ABSTRACT

Software processes comprise many steps; coding is followed by building, integration testing, system testing, deployment, operations, among others. Software process integration and automation have been areas of key concern in software engineering, ever since the pioneering work of Osterweil; market pressures for Agility, and open, decentralized, software development have provided additional pressures for progress in this area. But do these innovations actually help projects? Given the numerous confounding factors that can influence project performance, it can be a challenge to discern the effects of process integration and automation. Software project ecosystems such as GitHub provide a new opportunity in this regard: one can readily find large numbers of projects in various stages of process integration and automation, and gather data on various influencing factors as well as productivity and quality outcomes. In this paper we use large, historical data on process metrics and outcomes in GitHub projects to discern the effects of one specific innovation in process automation: *continuous integration*. Our main finding is that continuous integration improves the productivity of project teams, who can integrate more outside contributions, without an observable diminishment in code quality.

Categories and Subject Descriptors

D.2.5 [**Software Engineering**]: Testing and Debugging—*Testing tools*

General Terms

Experimentation, Human Factors

Keywords

Continuous integration, GitHub, pull requests

*Bogdan Vasilescu and Yue Yu are both first authors, and contributed equally to the work.

1. INTRODUCTION

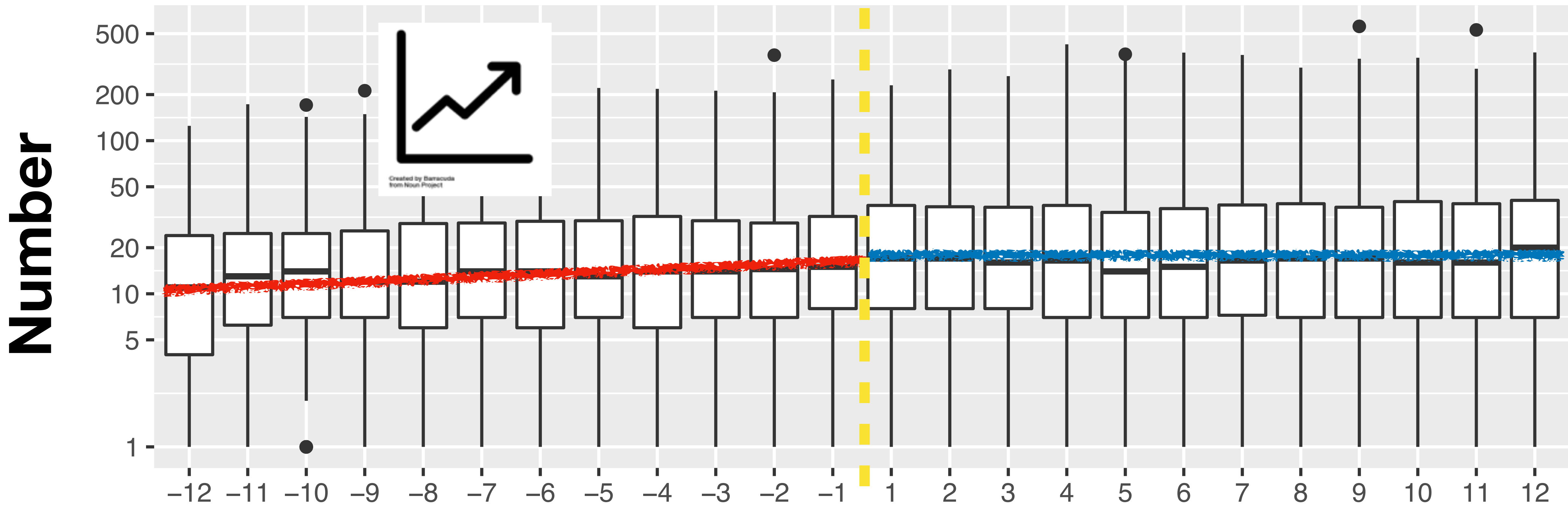
Innovations in software technology are central to economic growth. People place ever-increasing demands on software, in terms of features, security, reliability, cost, and ubiquity; and these demands come at an increasingly faster rate. As the appetites grow for ever more powerful software, the human teams working on them have to grow, and work more efficiently together.

Modern games, for example, require very large bodies of code, matched by teams in the tens and hundreds of developers, and development time in years. Meanwhile, teams are globally distributed, and sometimes (*e.g.*, with open source software development) even have no centralized control. Keeping up with market demands in an agile, organized, repeatable fashion, with little or no centralized control, requires a variety of approaches, including the adoption of technology to enable process automation. Process Automation *per se* is an old idea, going back to the pioneering work of Osterweil [32]; but recent trends such as open-source, distributed development, cloud computing, and software-as-a-service, have increased demands for this technology, and led to many innovations. Examples of such innovations are distributed collaborative technologies like `git` repositories, forking, pull requests, continuous integration, and the DEVOPS movement [36]. Despite rapid changes, it is difficult to know how much these innovations are helping improve project outcomes such as productivity and quality. A great many factors such as code size, age, team size, and user interest can influence outcomes; therefore, teasing out the effect of any kind of technological or process innovation can be a challenge.

The GitHub ecosystem provides a very timely opportunity for study of this specific issue. It is very popular (increasingly so) and hosts a tremendous diversity of projects. GitHub also comprises a variety of technologies for distributed, decentralized, social software development, comprising version control, social networking features, and process automation. The development process on GitHub is more democratic than most open-source projects: *anyone* can submit contributions in the form of *pull requests*. A pull request is a candidate, proposed code change, sometimes responsive to a previously submitted modification request (or *issue*). These pull requests are reviewed by project insiders (*aka* core developers, or integrators), and accepted if deemed of sufficient quality and utility. Projects that are more popular and widely used can be expected to attract more interest, and more pull requests; these will have to be

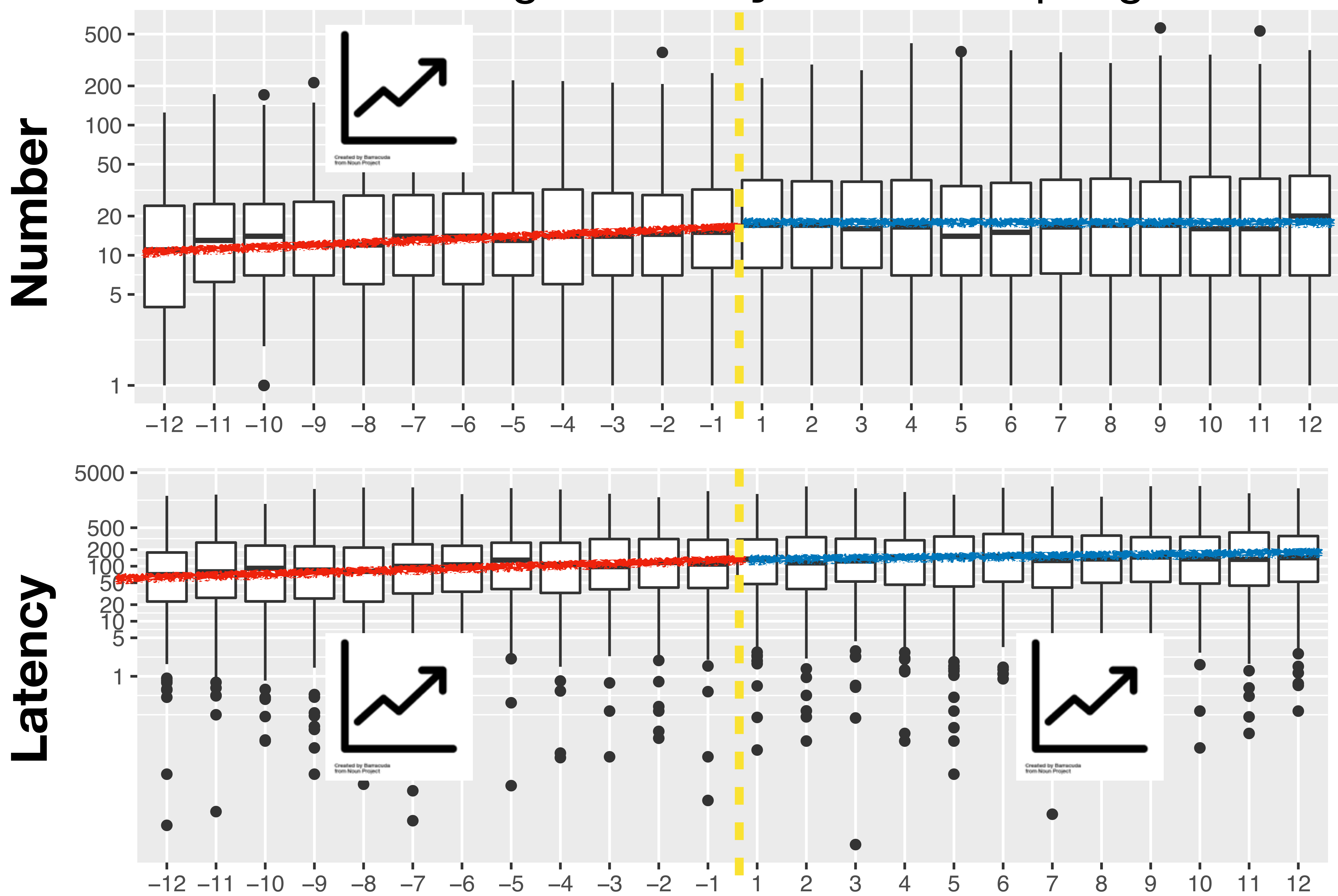
Closed PRs

Increasing trend **only before** adopting Travis CI



Closed PRs

Increasing trend **only before** adopting Travis CI



RQs

More frequent commits

Both before and after Travis

Impact on automated testing?

↓missing files/dep
↑comp/exec errors
↑failed tests

Created by Ramesha from Noun Project

Smaller code changes

Affected only for merge commits

Created by Misha from Noun Project

More issues and pull requests closed

Increasing trend slowed down

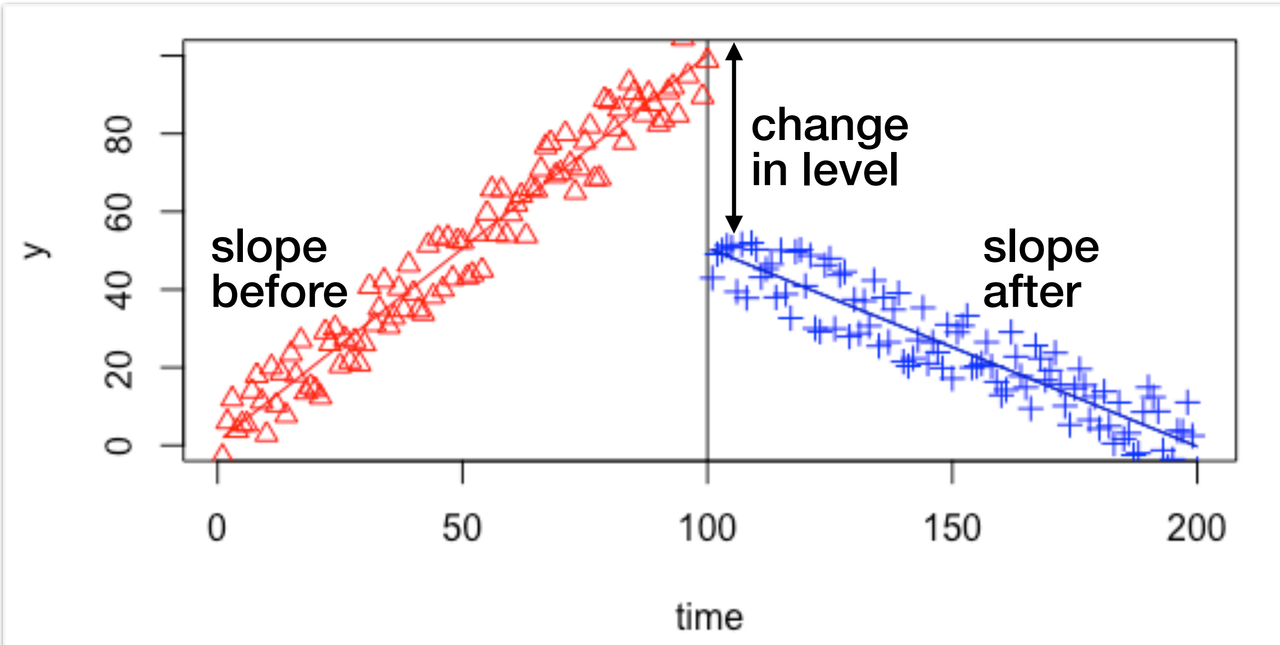
Created by Barracuda from Noun Project

Quick pull requests resolution

increases pre-Travis, flattened out by Travis
duration not affected

Created by Genius Icons from Noun Project

Interrupted time series



$$y_i = \alpha + \beta \cdot time_i + \gamma \cdot intervention_i + \delta \cdot time\_after\_intervention_i + \varepsilon_i$$

NSFC

Award: BK20130014

NSF

Awards: 1717415, 1717370

RQs

Impact on automated testing?

↓missing files/dep
↑comp/exec errors
↑failed tests

More frequent commits

Not affected by Travis

Smaller code changes

Affected only for merge commits

More issues and pull requests closed

increasing trend slowed down

Quick pull requests resolution

increases pre-Travis, flattened out by Travis
duration not affected

Yangyang Zhao
Alexander Serebrenik
Yuming Zhou
Vladimir Filkov
Bogdan Vasilescu