Three habits to bridge research code and sustainable software

January 21, 2016

Chris Gates (cgates@umich.edu)
UM Bioinformatic Core
DCMB Masters student

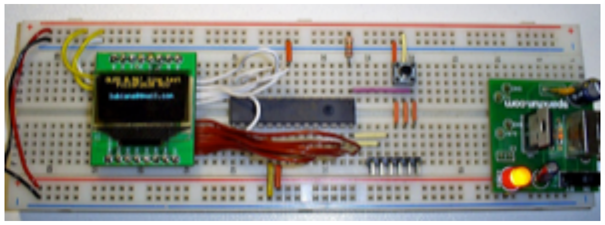
Agenda

- Background / Motivations
- Research code and sustainable software
- Habits:
  - Version control
  - Testing
  - Pairing

Background / motivations

- About me
  - 20 yrs in IT; 15 years in Software Engineering
  - 2007 Compendia (Oncomine)
  - 2012 UM Bioinformatics Core
    - What the core does
    - What I do
      - IT
      - Bioinformatics projects
      - Software engineering

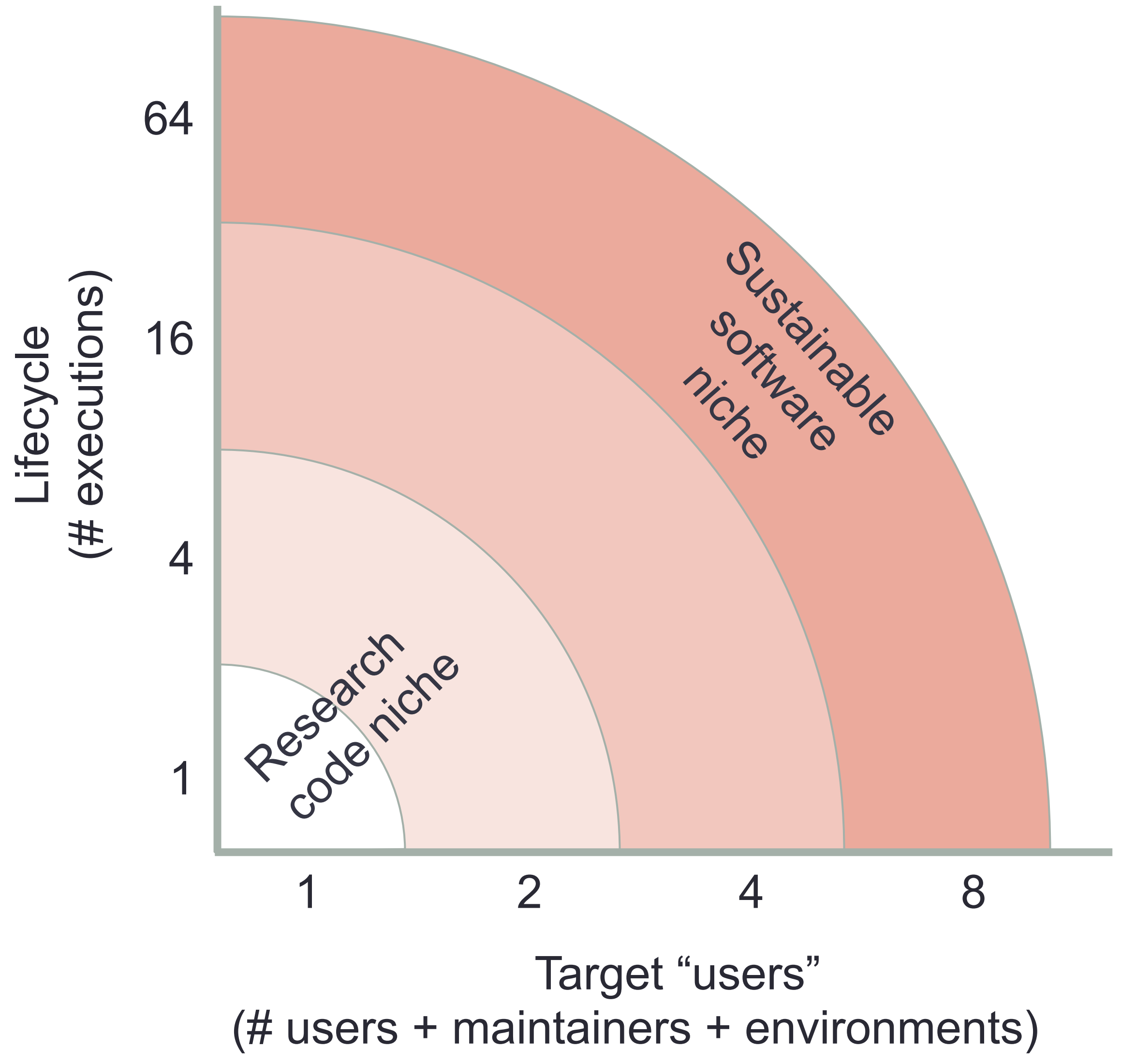
Research code and sustainable software are often distinct



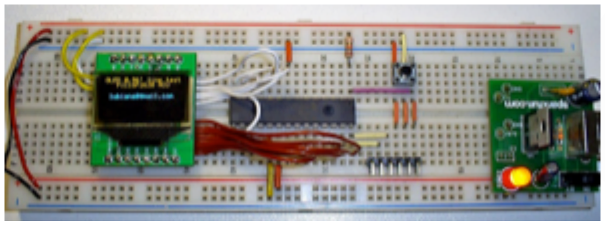
	Research code
Optimized for	Enabling discovery
Interaction	Informal; Interactive exploration
Operational knowledge	Implicit (authors)
Target users	Authors; Subject Experts
Lifecycle	Short; mostly development

Heroux (2009), Wilson (2014)

Operational profiles create distinct niches

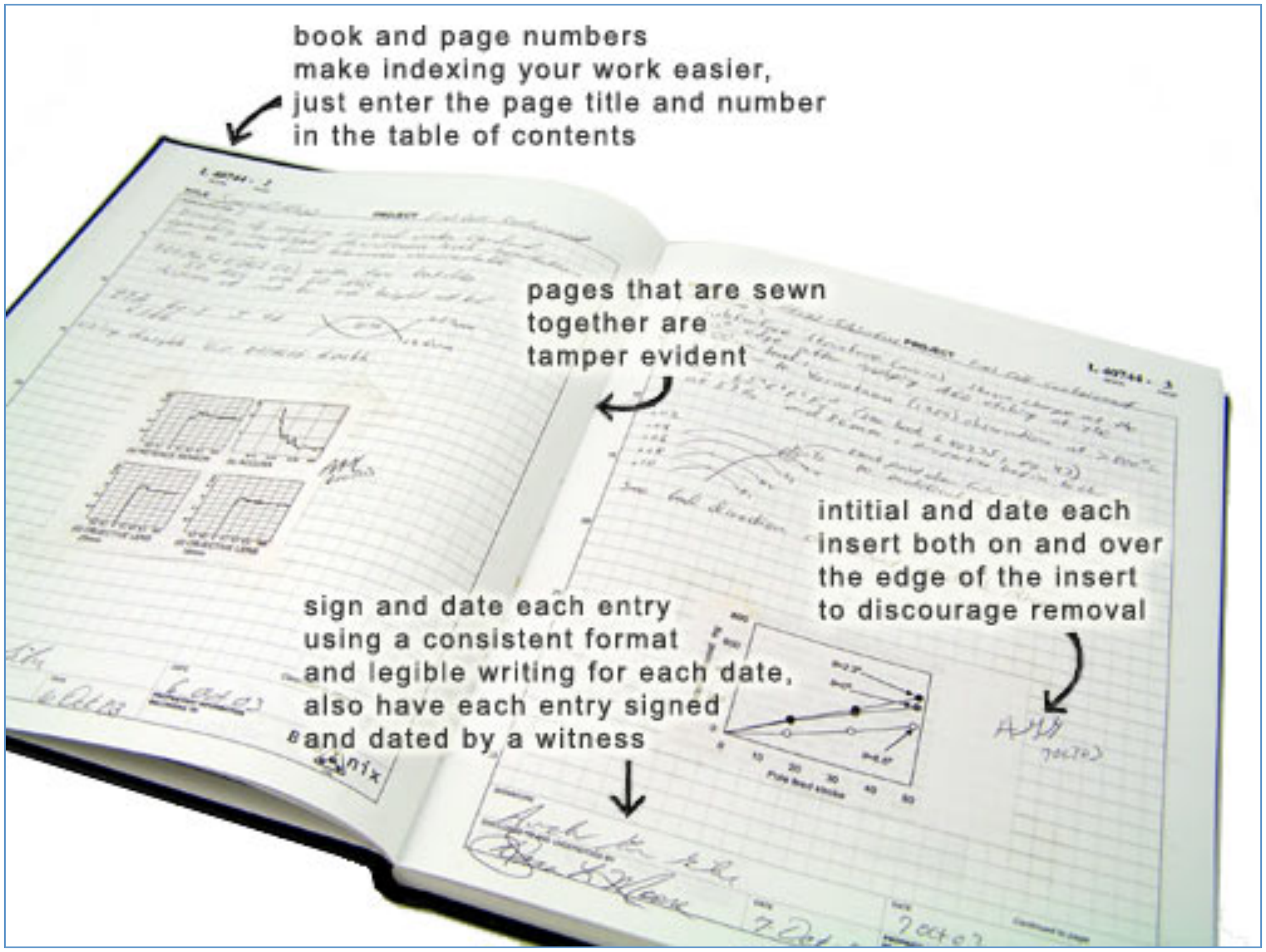


Research code and sustainable software share many values



Research code	Sustainable software
Reproducibility	Robustness Portability Reusability
Early publication	Time to market
Correctness	
Simplicity	

I. Version control is a lab notebook for files

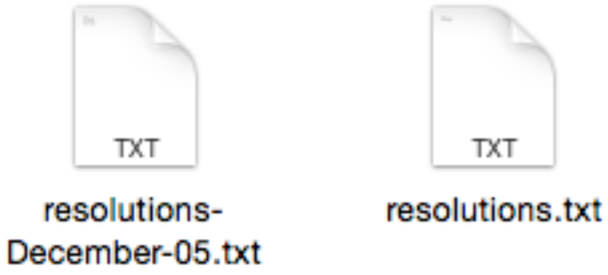

book and page numbers make indexing your work easier, just enter the page title and number in the table of contents
pages that are sewn together are tamper evident
intitial and date each insert both on and over the edge of the insert to discourage removal
sign and date each entry using a consistent format and legible writing for each date, also have each entry signed and dated by a witness

You use version control now

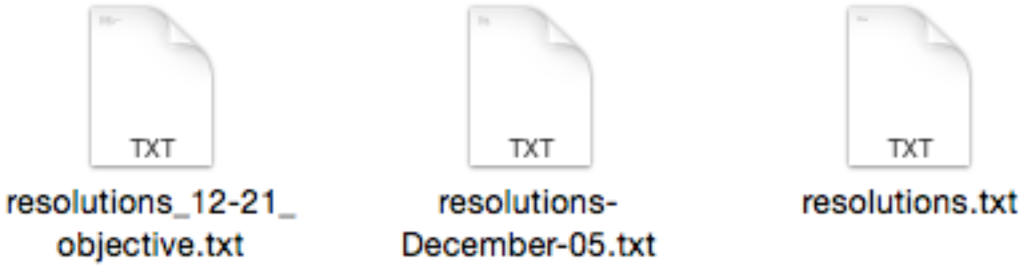
floss
stop cursing
lose weight

TXT
resolutions.txt

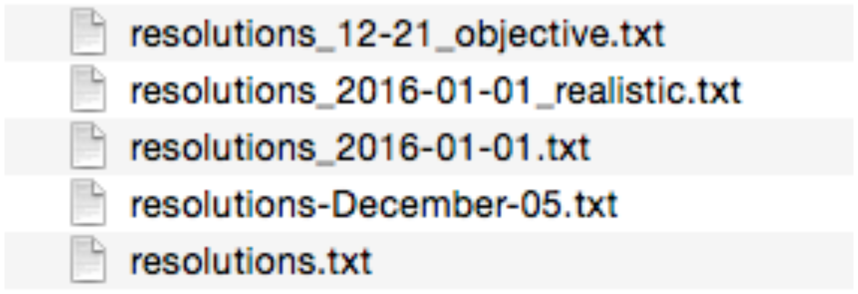
floss
stop cursing
lose weight
exercise more
cut out sugar



floss 1x a day
limit cursing in front on kids
lose 10 pounds by April 15
exercise 2x a week
cut out desserts after lunch



But using a file system as version control is problematic



resolutions_12-21_objective.txt
resolutions_2016-01-01_realistic.txt
resolutions_2016-01-01.txt
resolutions-December-05.txt
resolutions.txt

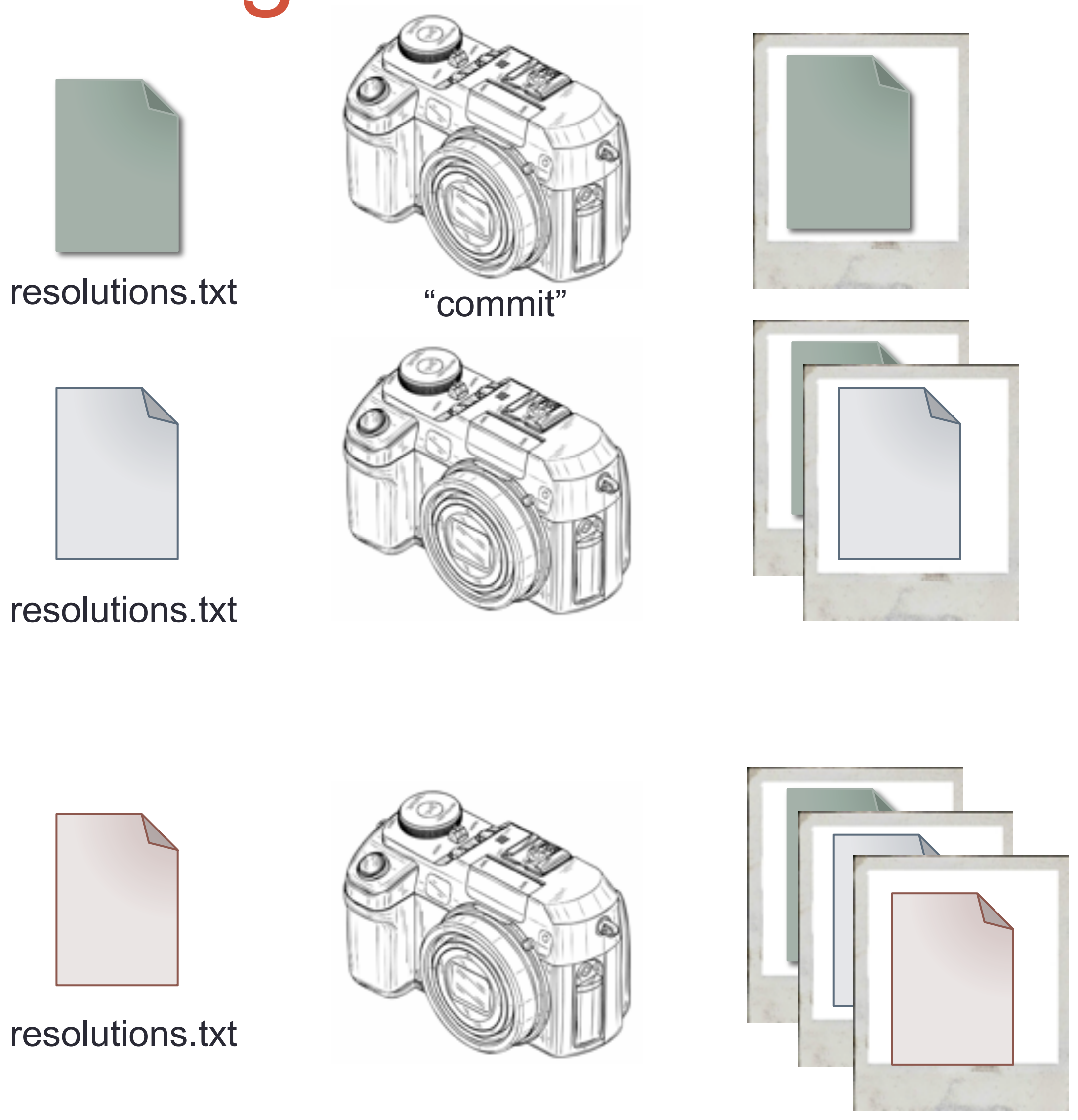
- Which is the most current file?
- What is the order of revisions?
  - What version did I use on December 23?
- Why was the file changed on Jan 1, 2016?
  - Who made that change?

Version control using Git

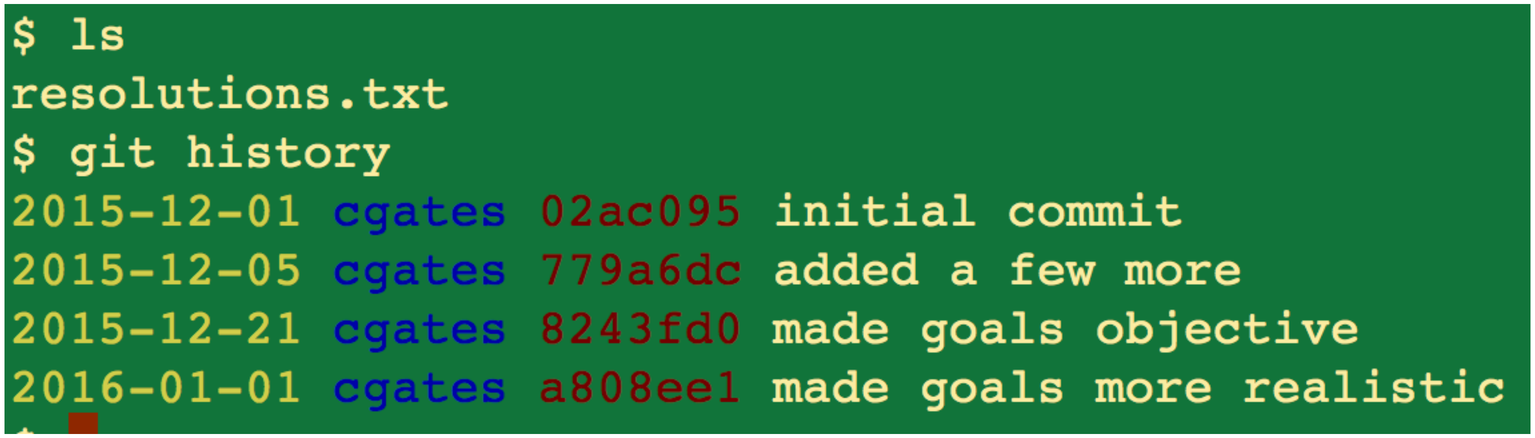
floss
stop cursing
lose weight

floss
stop cursing
lose weight
exercise more
cut out sugar

floss 1x a day
limit cursing in front on kids
lose 10 pounds by April 15
exercise 2x a week
cut out desserts after lunch



Version control using Git

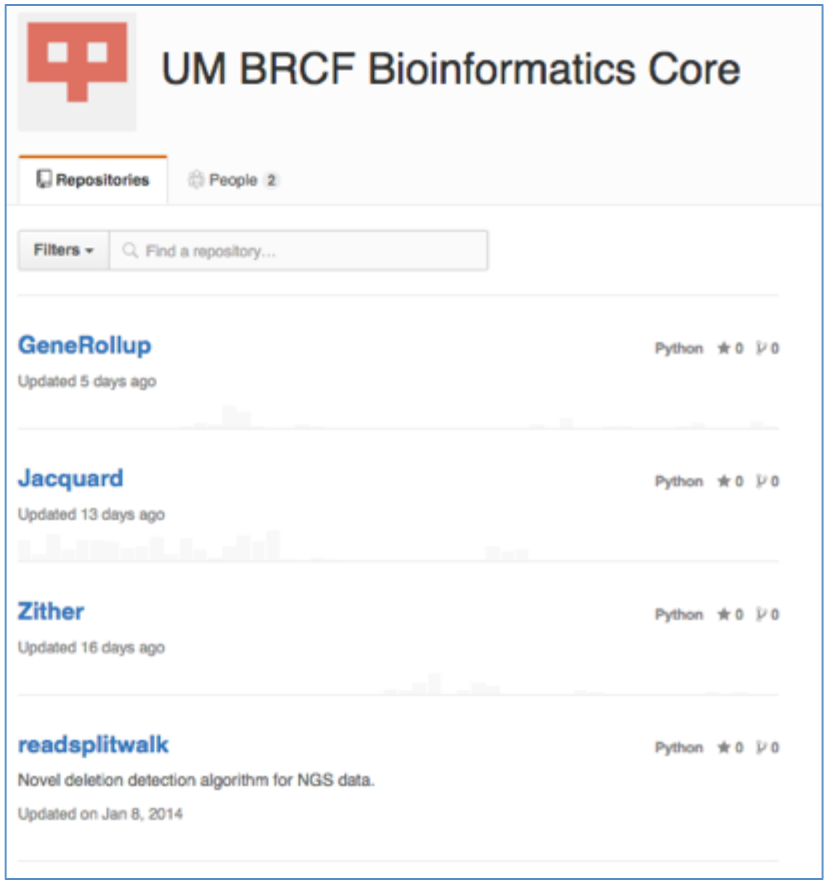


```
$ ls
resolutions.txt
$ git history
2015-12-01 cgates 02ac095 initial commit
2015-12-05 cgates 779a6dc added a few more
2015-12-21 cgates 8243fd0 made goals objective
2016-01-01 cgates a808ee1 made goals more realistic
```

- Which is the most current file? (resolutions.txt)
- What is the order of revisions? (as above)
  - What version did I use on December 23? (made goals objective)
- Why was the file changed on 1/1/2016? (“more realistic”)
  - Who made that change? (cgates)

Benefits of Git and GitHub

- Git
  - Provenance and history
  - Simpler/cleaner
  - Backup
- **Github** (Hosted version control)
  - Free for public projects
  - Better backup
  - Collaboration
    - Sharing
    - Publishing
    - Cooperative development


UM BRCF Bioinformatics Core

GeneRollup
Updated 5 days ago
Python ★0 0

Jacquard
Updated 13 days ago
Python ★0 0

Zither
Updated 16 days ago
Python ★0 0

readsplitwalk
Novel deletion detection algorithm for NGS data.
Updated on Jan 8, 2014
Python ★0 0

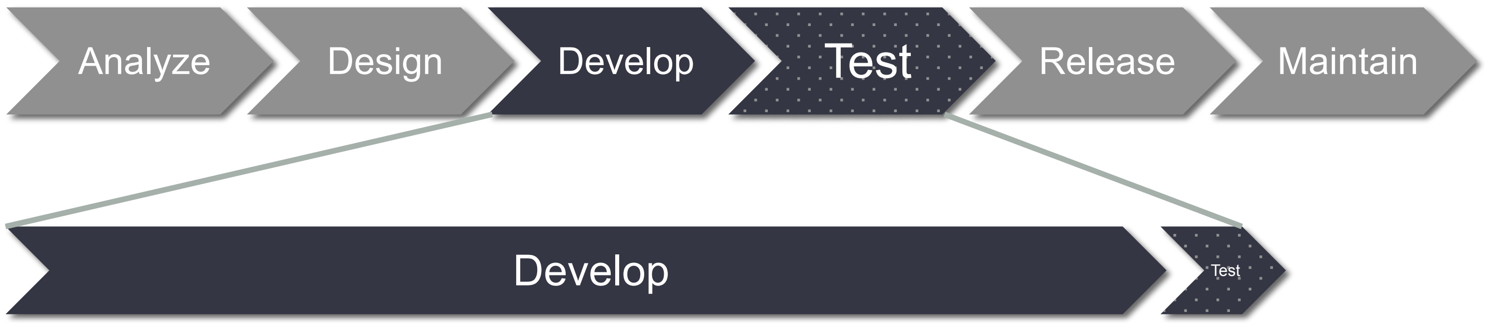
Chacon (Pro Git), Wilson (2014),

Version Control: Threats to adoption

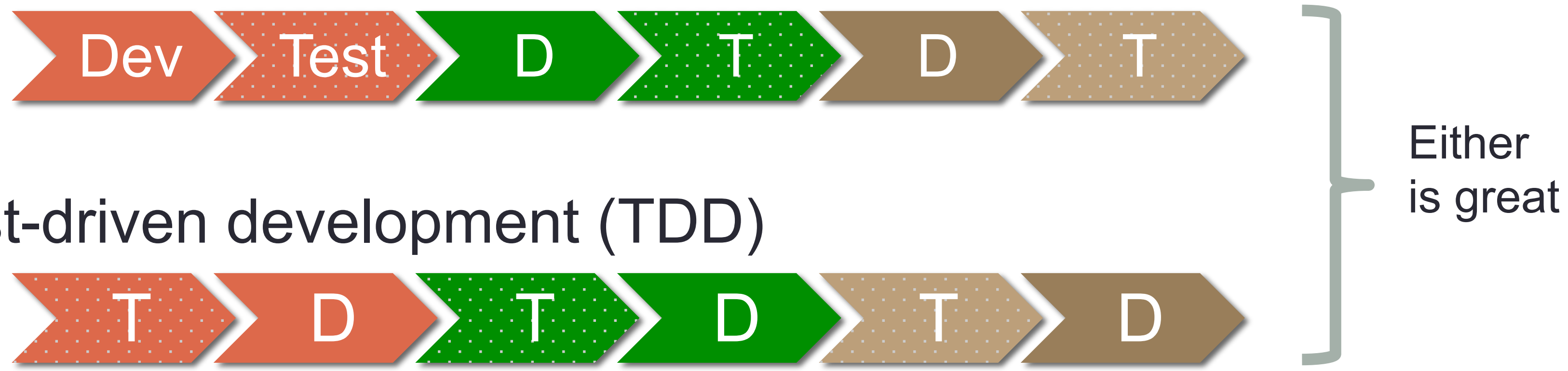
- Big files
  - Don’t version control things you don’t edit by hand
- Privacy
  - Github/Bitbucket – cheap private accounts
  - Private hosting is easy for basic projects

II. Testing

- Code and fix (ad hoc testing)
- Traditional (waterfall) software development lifecycle



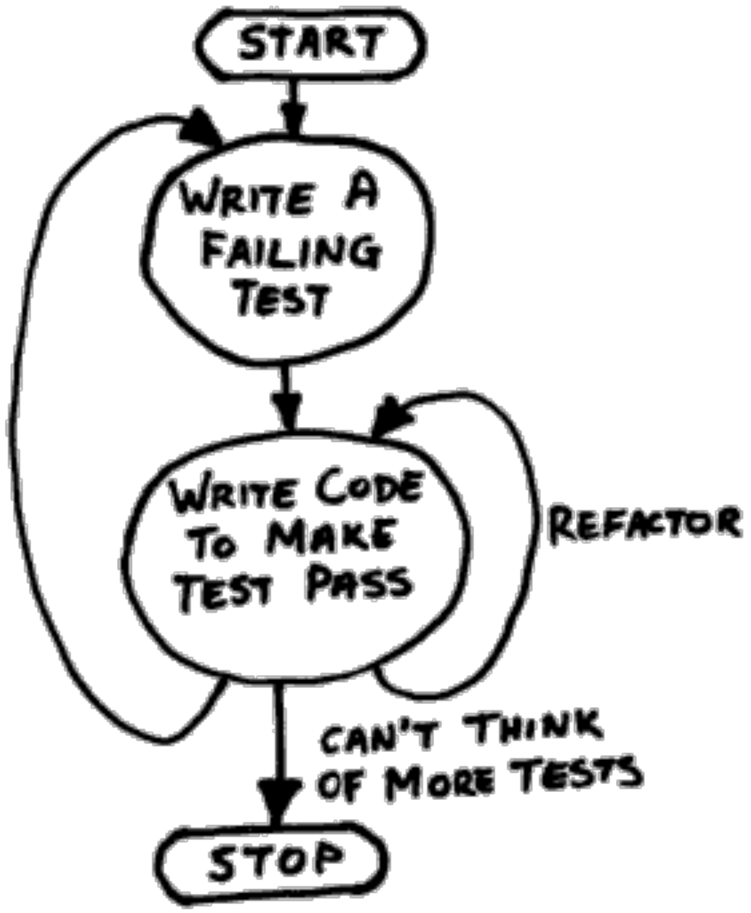
- Unit testing (Automated, iterative testing)



- Test-driven development (TDD)

Beck (1999), Beck (2003)

TDD Example: Roman Numerals

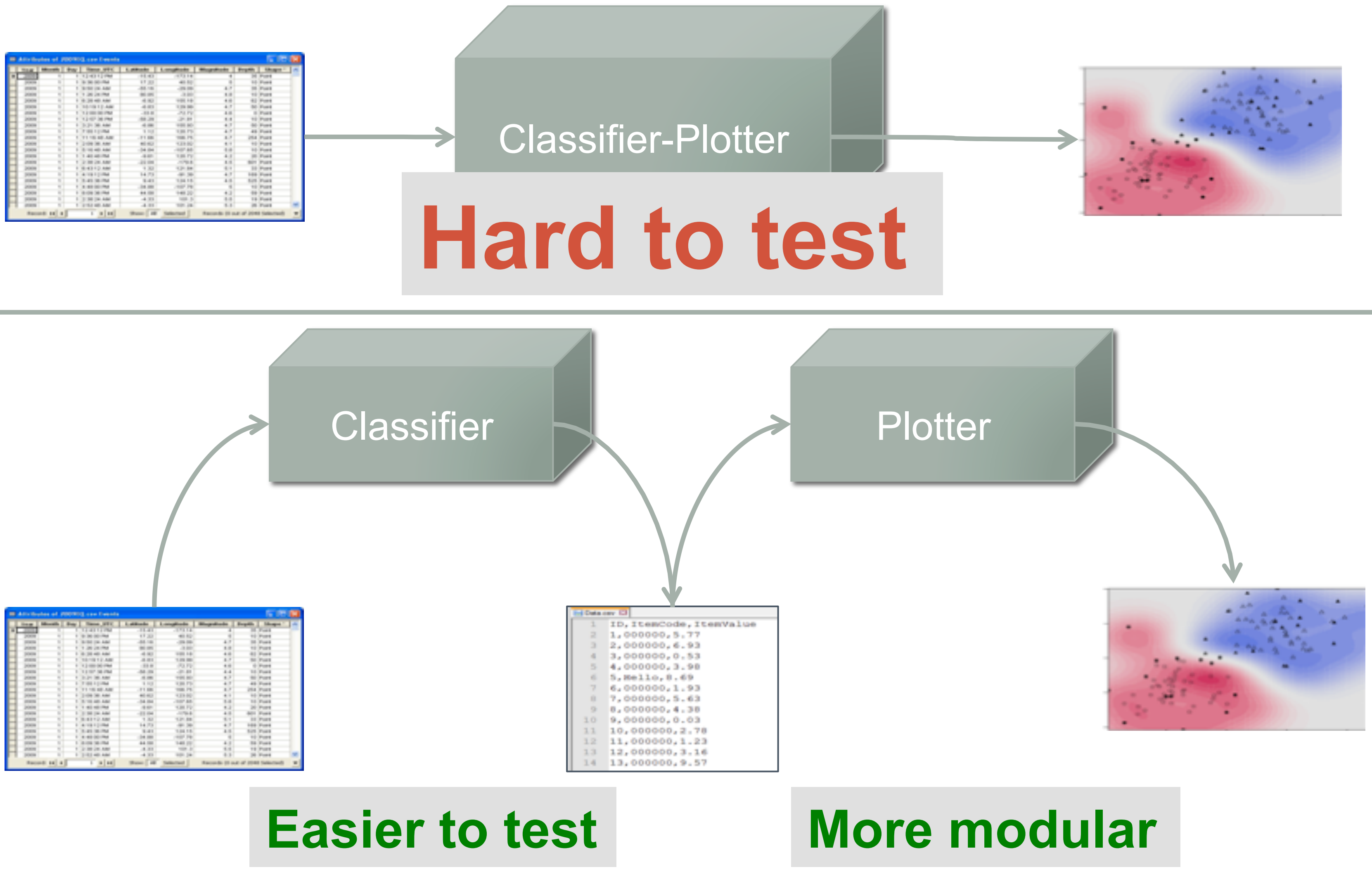


I → 1

II → 2

V → 5

Testing influences your design



```
ID,ItemCode,ItemValue
1,000000,5.77
2,000000,6.93
3,000000,0.53
4,000000,3.98
5,Hello,8.69
6,000000,1.93
7,000000,5.63
8,000000,4.38
9,000000,0.03
10,000000,2.78
11,000000,1.23
12,000000,3.16
13,000000,9.57
```

Beck (2003), Sandve (2013)

Benefits of automated unit testing

- Improves correctness during development
- Encourages re-use
- Passing tests quantify progress
- Reduces regressions over time
- Typically correlates with higher quality than “code and fix”

BfxCore projects	Unit tests
AmpliconSoftClipper	71
Epee	717
Jacquard	537
Nephroseq	8315
Zither	53

Beck (2003), Makinen (2014), Nagappan (2008), Rafique (2013)

Testing: Threats to adoption

- Stochastic algorithms harder (use/allow seeding)
- Big data slower (use small data)
- UI hard to test (separate data and presentation)
- Benefit smaller on simpler problems
- Startup cost
- Testing doesn’t guarantee correct behavior (thanks, Volkswagen!)
- Need a good problem model

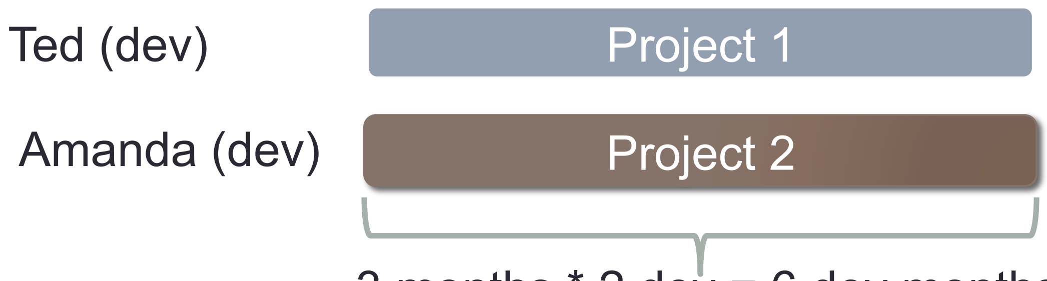
III. Pair-programming

Two people, one keyboard

Beck (1999), Cockburn (2000), Williams (2000), Williams (2002)

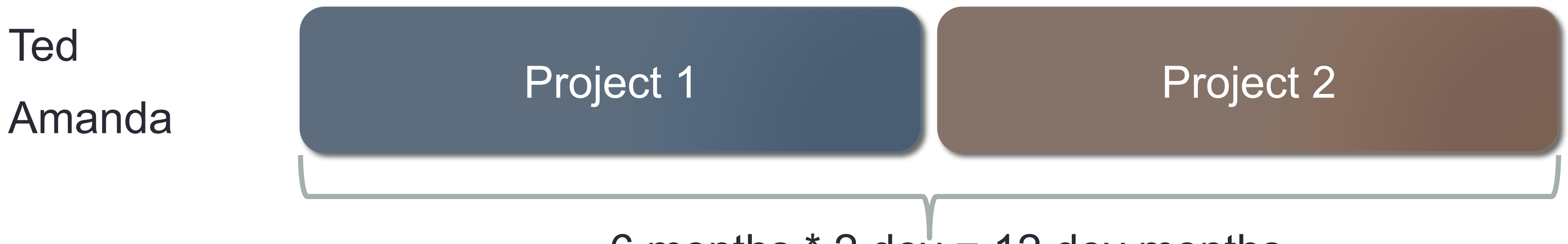
Economics of pairing

Parallel development (conventional)



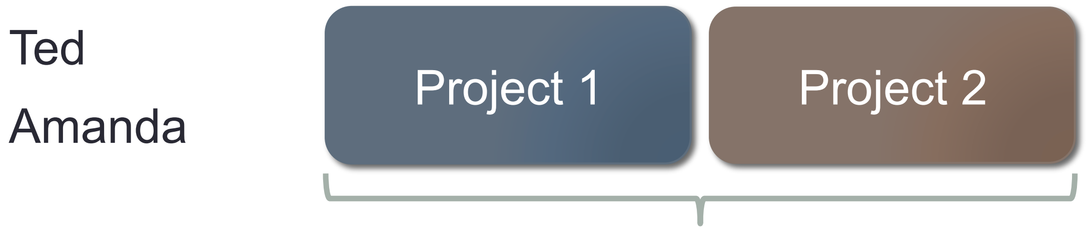
3 months * 2 dev = 6 dev months

If development were about typing, you would expect:



6 months * 2 dev = 12 dev months

But in actuality, developing is more about problem solving:



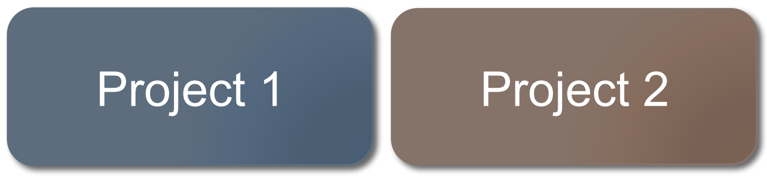
3.3 months * 2 dev = 6.6 dev months

Benefits of pairing

Actual results

Ted

Amanda

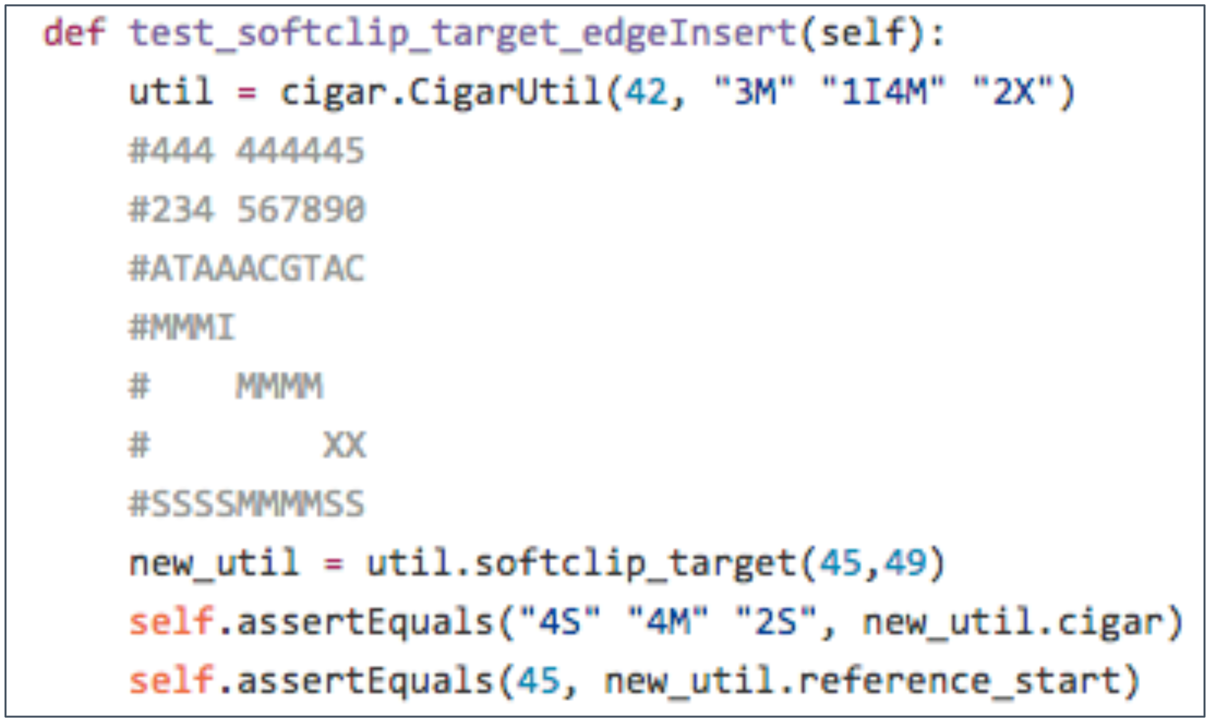


- **(120% effort)**
- Shared understanding
- Homogenous code
- Early publication
- Simpler management

Beck (1999), Cockburn (2000), Williams (2000), Williams (2002)

Pairing at BfxCore

Projects
AmpliconSoftClipper
CRIDA
Epee
Jacquard
Nephroseq
Zither
(others)



```
def test_softclip_target_edgeInsert(self):
    util = cigar.CigarUtil(42, "3M" "1I4M" "2X")
    #444 444445
    #234 567890
    #ATAAACGTAC
    #MMMI
    #    MMMM
    #       XX
    #SSSSMMMMSS
    new_util = util.softclip_target(45,49)
    self.assertEquals("4S" "4M" "2S", new_util.cigar)
    self.assertEquals(45, new_util.reference_start)
```

Pair-programming: threats to adoption

- Logistics
- Mentorship
- Culture of individual ownership
- Furniture

Habits can benefit both research code and sustainable software

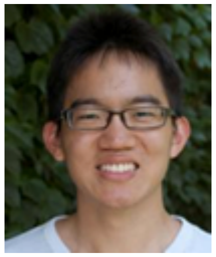
Habit / Value	Version control	Testing	Pairing
Reproducibility	✓	✓	✓
Correctness		✓	✓
Publication	✓		✓
Simplicity	✓	✓	✓

- Habits don't make good decisions; they just make bad decisions more painful.
- Note that adoption of any habit (including good habit) reduces efficiency at the outset.
- Wilson (2014): Science is more than a body of knowledge – it's a way of doing things that enables and encourages collaboration.

Thanks and questions

- Bioinformatics core

- Ana Grant
- Bob Boguski
- Divya Kriti
- Pete Ulintz
- Jessica Bene
- Kevin Meng
- Ross Patterson



References (1)

- UM BIOINF575
- UM EECS398
- Software carpentry: http://software-carpentry.org/
- Software Sustainability Institute: http://www.software.ac.uk/
- Chacon: Pro Git: https://git-scm.com/book/en/v2
- Gentzkow, Shapiro: Code and Data for the Social Sciences: A Practitioner's Guide: http://www.brown.edu/Research/Shapiro/pdfs/CodeAndData.pdf

References (2)

- Beck K, Extreme programming explained: embrace change, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1999
- Beck, K.: Test-Driven Development: By Example. Addison-Wesley (2003)
- Cockburn A, Williams L. The costs and benefits of pair programming. Extreme programming examined, pages 223–247, 2000.
- Heroux MA, Willenbring JM. Barely sufficient software engineering: 10 practices to improve your cse software. In Software Engineering for Computational Science and Engineering, 2009. SECSE '09. ICSE Workshop on, pages 15–21, May 2009.
- Loman N, Watson M. So you want to be a computational biologist? Nat Biotechnol. 2013;31: 996–998. doi: 10.1038/nbt.2740. pmid: 24213777
- Mäkinen S, Münch J. "Effects of Test-Driven Development : A Comparative Analysis of Empirical Studies" in Software Quality. Model-Based Approaches for Advanced Software and Systems Engineering : 6th International Conference, SWQD 2014, Vienna, Austria, January 14-16, 2014. Proceedings , pp. 155-169 Lecture Notes in Business Information Processing , vol. 166 . , 10.1007/978-3-319-03602-1_10
- Nagappan N, et al. Realizing quality improvement through test driven development: results and experiences of four industrial teams
- Empirical Softw. Eng., 13 (June 2008), pp. 289–302
- Noble WS. A quick guide to organizing computational biology projects. PLoS Computational Biology. 2009 Jul;5(7):e1000424 doi: 10.1371/journal.pcbi.1000424. pmid:19649301
- Osborne JM, Bernabeu MO, Bruna M, Calderhead B, Cooper J, Dalchau N, et al. Ten Simple Rules for Effective Computational Research. PLoS Comput Biol. 2014;10: e1003506 doi: 10.1371/journal.pcbi.1003506. pmid:24675742
- Rafique Y, Misic VB. The Effects of Test-Driven Development on External Quality and Productivity: A Meta-Analysis, IEEE Transactions on Software Engineering, v.39 n.6, p.835-856, June 2013 [doi>10.1109/TSE.2012.28]
- Sandve GK, Nekrutenko A, Taylor J, Hovig E. Ten Simple Rules for Reproducible Computational Research. PLoS Comput Biol. 2013;9:e1003285. doi: 10.1371/journal.pcbi.1003285. pmid:24204232
- Williams L, et al. Strengthening the Case for Pair Programming, IEEE Software, v.17 n.4, p.19-25, July 2000 [doi>10.1109/52.854064]
- Williams L, Kessler R. Pair Programming Illuminated, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 2002
- Wilson et al. Best Practices for Scientific Computing. 2014 PLoS Biol 12(1): e1001745. doi:10.1371/journal.pbio.1001745