PrairieLearn:
A flexible platform for writing randomized, auto-grading questions

Matt West

Mechanical Science and Engineering

University of Illinois at Urbana-Champaign

With Mariana Silva, Nathan Walters, Tim Bretl, Dave Mussulman, Craig Zilles, Geoffrey Herman, Nicolas Nytko, Cinda Heeren, Jonatan Schroeder, Tim Yang, Philipp Hieronymi, James Balamuta, and many more

March 2nd, 2021

What is PrairieLearn?

- Flexible web platform for randomized question generation and auto-grading

Standard tools to write questions

- Open-source project used by hundreds of faculty

and more…

- Similar to WeBWork, LON-CAPA, WebAssign but more flexible and extensible

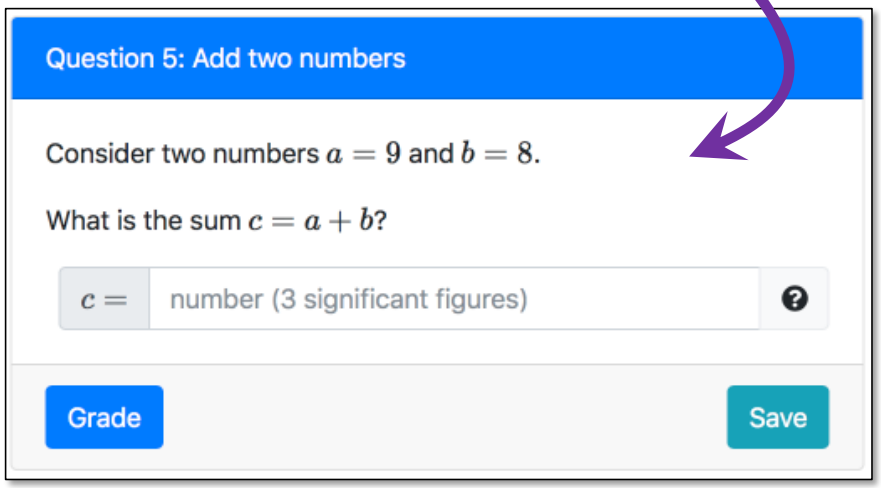
“Add numbers” question generator

question.html

```
<pl_question_panel>
  <p> Consider two numbers $a = {{params.a}}$ and $b = {{params.b}}$.
  <p> What is the sum $c = a + b$?
</pl_question_panel>

<pl_number_input answers_name="c" comparison="sigfig" digits="3" label="$c=$"/>
```

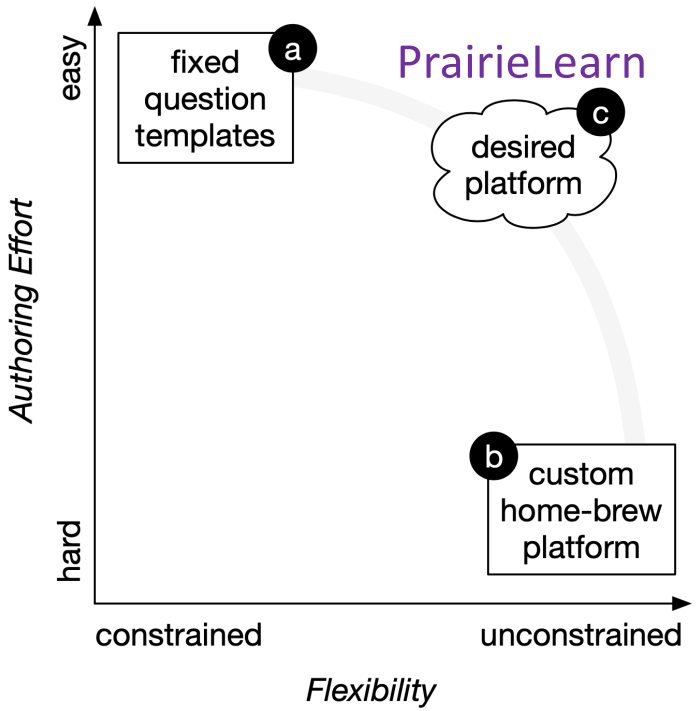
server.py

```
import random

def generate(data):

    # Sample two random integers between 5 and 10 (inclusive)
    a = random.randint(5, 10)
    b = random.randint(5, 10)

    # Put these two integers into data['params']
    data['params']['a'] = a
    data['params']['b'] = b

    # Compute the sum of these two integers
    c = a + b

    # Put the sum into data['correct_answers']
    data['correct_answers']['c'] = c
```



Student view

Make easy things easy, hard things possible



Learning principles:

- Scaffolded practice
- Immediate feedback
- Frequent testing
- Help from human instructors

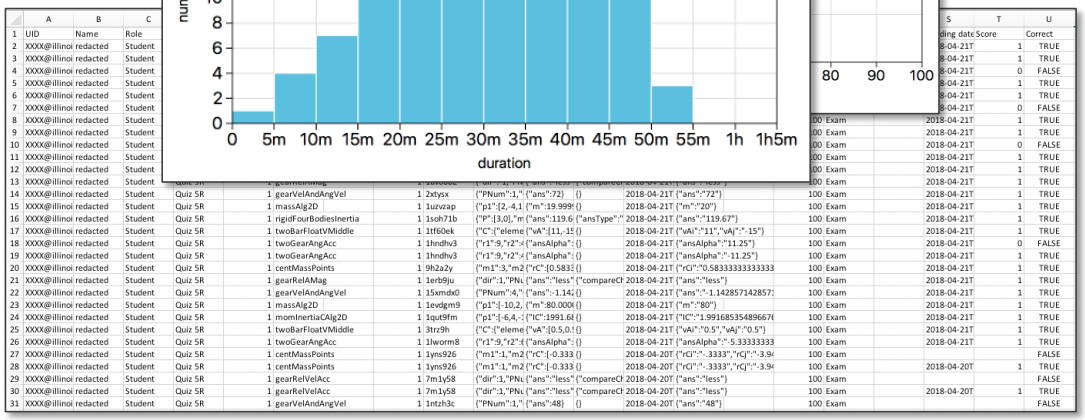
Analytics and data

Record every action taken by students

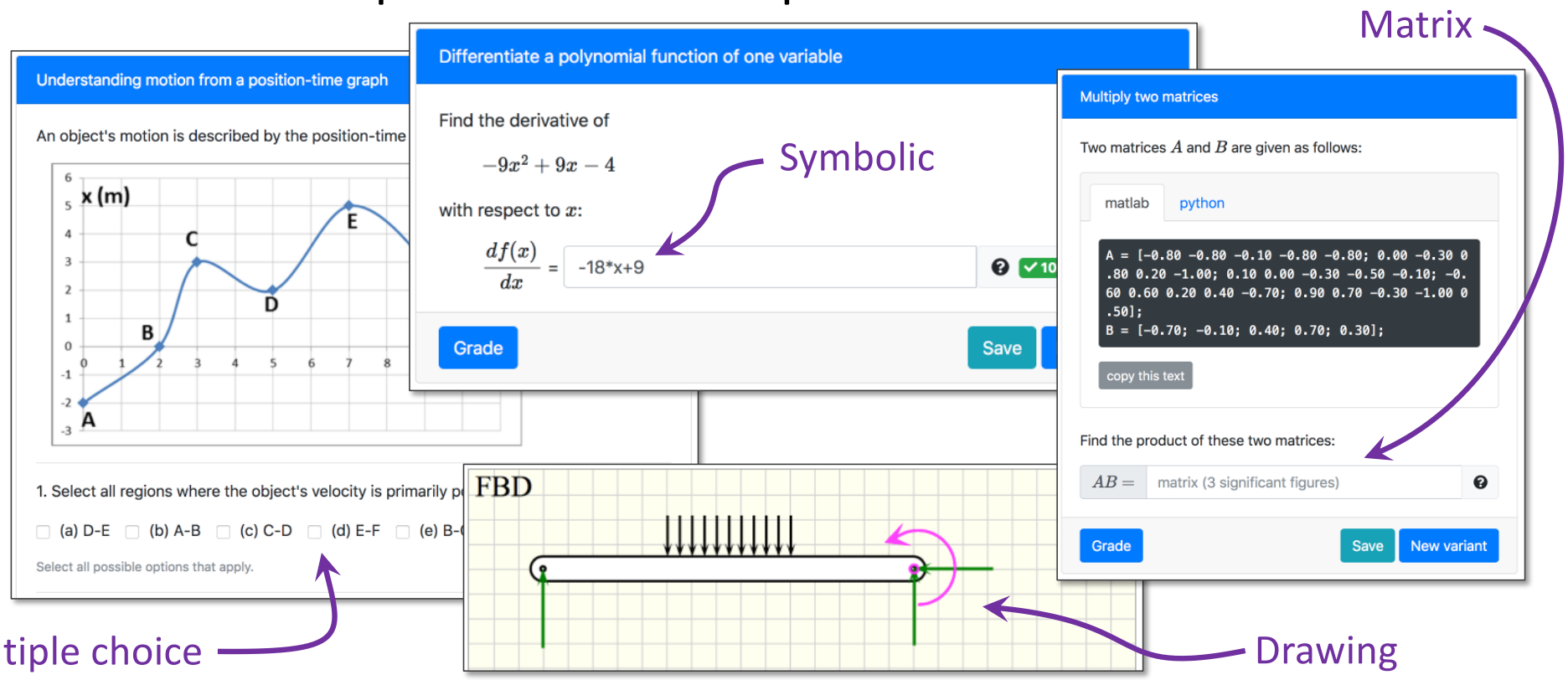
Ready-made visualizations

All raw data is downloadable

2018-04-21 14:00:13 (CDT)	@illinois.edu	View variant	I-6.1 (gearRelVelAcc)	S-7#1	
2018-04-21 14:00:18 (CDT)	@illinois.edu	Submission	I-6.1 (gearRelVelAcc)	S-7#1	{"submitted_answer":{"ans":"greater"}}
2018-04-21 14:00:18 (CDT)	@illinois.edu	Grade submission	I-6.1 (gearRelVelAcc)	S-7#1	{"score":1,"correct":true,"feedback":null,"true_answer":{"ans":"greater"},"submitted_answer":{"ans":"greater"}}
2018-04-21 14:00:18 (CDT)	@illinois.edu	Score question	I-6.1 (gearRelVelAcc)	S-7#1	{"points":10,"max_points":10,"score_perc":100}
2018-04-21 14:00:18 (CDT)	@illinois.edu	Score assessment			{"points":10,"max_points":70,"score_perc":14.285714285 7143}
2018-04-21 14:00:19 (CDT)	@illinois.edu	View variant	I-6.1 (gearRelVelAcc)	S-7#1	
2018-04-21 14:00:21 (CDT)	@illinois.edu	View assessment overview			
2018-04-21 14:00:22 (CDT)	@illinois.edu	View variant	I-4 (gearVelAndAngVel)	S-6#1	
2018-04-21 14:03:33 (CDT)	@illinois.edu	Submission	I-4 (gearVelAndAngVel)	S-6#1	{"submitted_answer":{"ans":".33333"}}



Rich set of question components

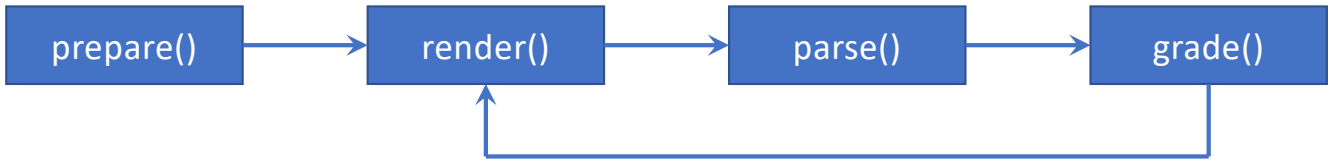


Easily add new custom elements that anyone can use

Elements: a flexible extension mechanism

```
1  <pl_question_panel>
2    <p> Consider two numbers $a = {{params.a}}$ and $b = {{params.b}}$.
3    <p> What is the sum $c = a + b$?
4  </pl_question_panel>
5
6  <pl_number_input answers_name="c" comparison="sigfig" digits="3" label="$c=$"/>
```

PrairieLearn *element:* A reusable package of HTML, Python, and JavaScript



Share between questions and courses

Symbolic element

question.html

```
<pl-question-panel>
  <p> Find the derivative of $f(x) = {{params.f}}$ with respect to $x$.</p>
</pl-question-panel>

$\dfrac{df(x)}{dx}$ = <pl-symbolic-input answers-name="df" variables="x"></pl-symbolic-input>
```

server.py

```
import sympy, random

def generate(data):

    x = sympy.symbols('x')
    a = random.randint(1,10)
    b = random.randint(1,10)
    c = random.randint(1,10)

    f = a*x**2 + b*x + c
    df = sympy.diff(f, x)

    data["params"]["f"] = sympy.latex(f)
    data["correct_answers"]["df"] = str(df)
```

Symbolic math in Python

Student view

Demo: Randomized symbolic calculations

Find the derivative of $f(x) = 8x^2 + 7x + 6$ with respect to x.

$\frac{df(x)}{dx} =$ 7 + 2*8*x 100%

Save & Grade | Save only | New variant

Correct answer

$\frac{df(x)}{dx} = 16x + 7$

Symbolic comparison for grading

Matrix elements

question.html

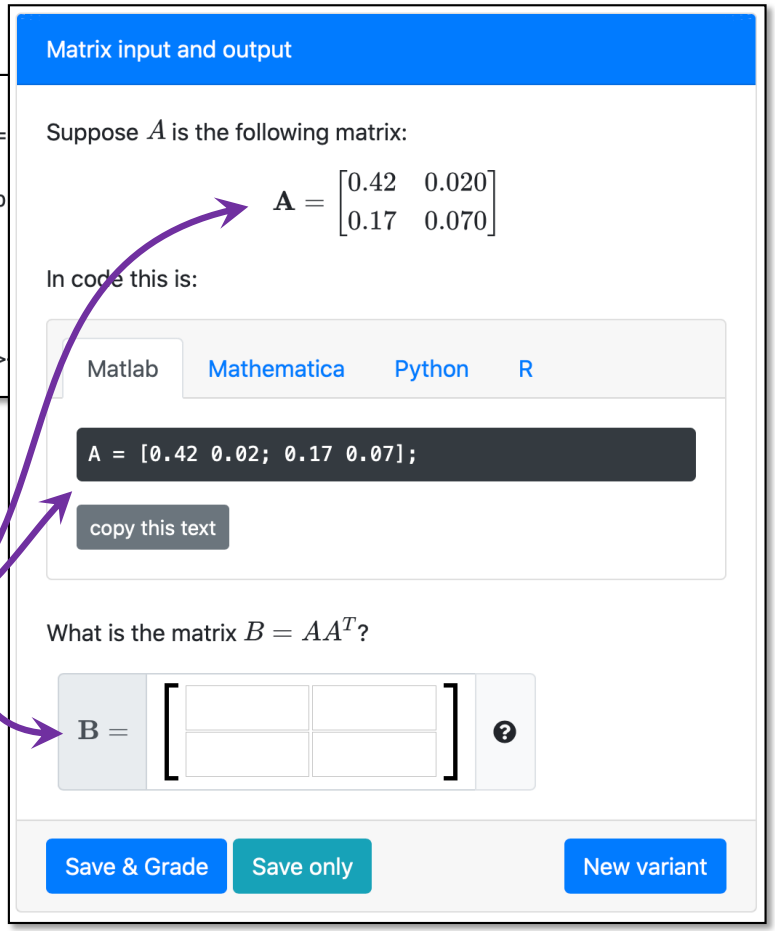
```
<pl-question-panel>
  <p> Suppose $A$ is the following matrix: $$A = <pl-matrix-latex params-name="A" digits=

  <p> In code this is: <pl-variable-output digits="2"><variable params_name="A">A</variab

  <p> What is the matrix $B = AA^T$? </p>
</pl-question-panel>

<pl-matrix-component-input answers-name="B" label="$B =$" comparison="sigfig" digits="2">
```

server.py

```
import numpy as np
import prairielearn as pl

def generate(data):

    A = np.random.rand(2, 2)
    A = np.round(A, 2)

    B = A @ A.T

    data["params"]["A"] = pl.to_json(A)
    data["correct_answers"]["B"] = pl.to_json(B)
```

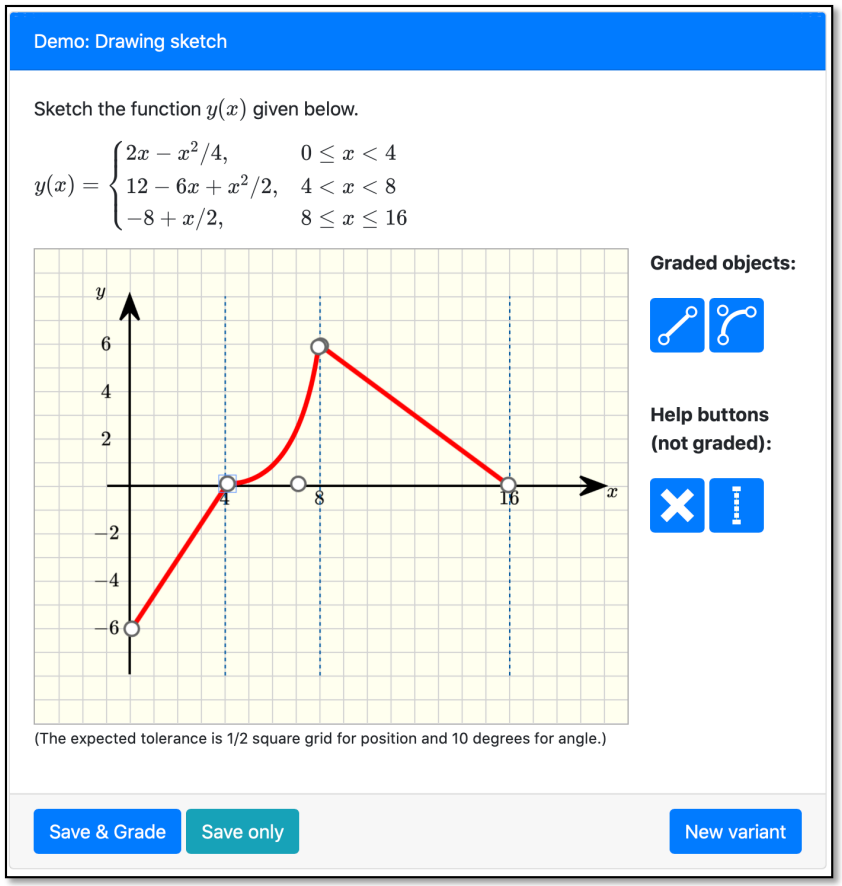
Many input and output formats

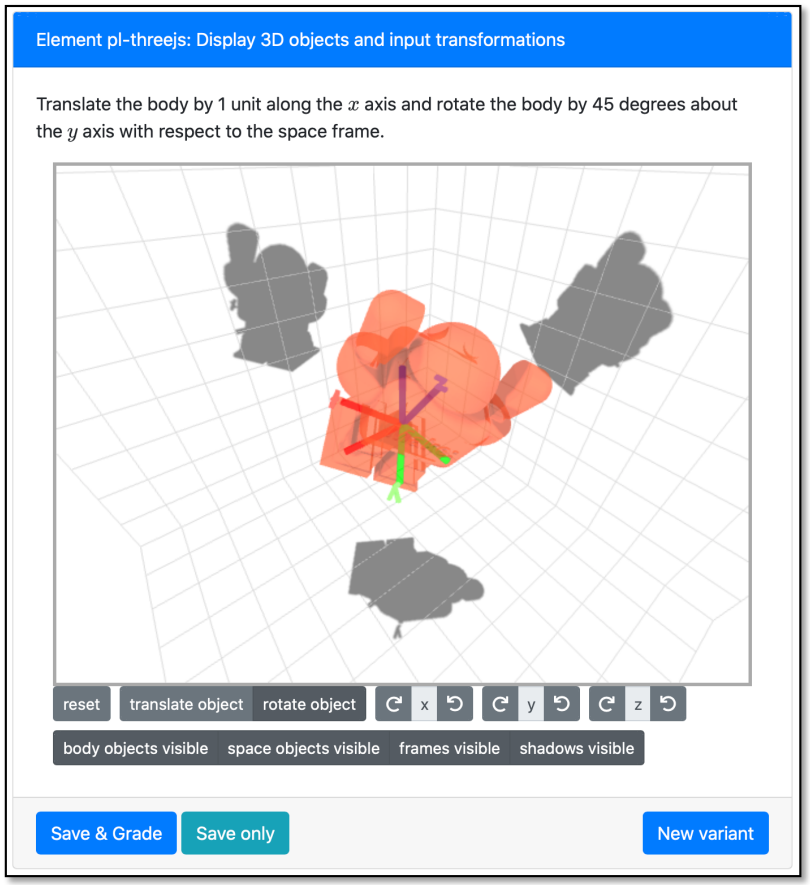
Student view



Diagrams and 3D objects

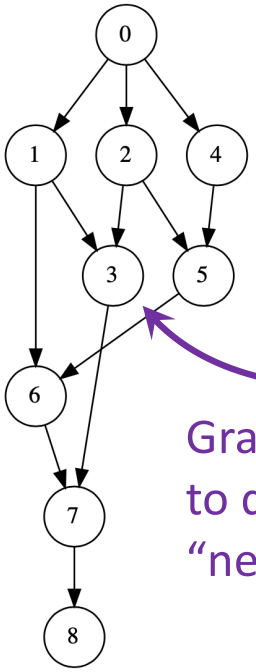
Built using existing open-source web libraries





Proof blocks

Scaffolded practice for proof writing



Graded by matching to directed graph of "needed by" relations

Isomorphism Preserves Bipartite

Drag and drop **ALL** of the blocks below to create a proof of the following statement.

If graphs G and H are isomorphic and G is bipartite, then H is bipartite.

Drag from here:

7 One of u and v is in $L(H)$, and the other is in $R(H)$.

4 Let $\langle u - v \rangle$ be an edge in H. (If instead there are no edges in H, then H is trivially bipartite and we are done.)

6 One of $f(u)$ and $f(v)$ is in $L(G)$, and the other is in $R(G)$.

5 $\langle f(u) - f(v) \rangle$ is an edge in G

8 H is bipartite. (end of proof)

2 Let $L(G)$ and $R(G)$ be a partitioning of G's vertices such that each edge of G has one endpoint in each set.

Construct your solution here:

0 Assume G and H are isomorphic graphs and G is bipartite.

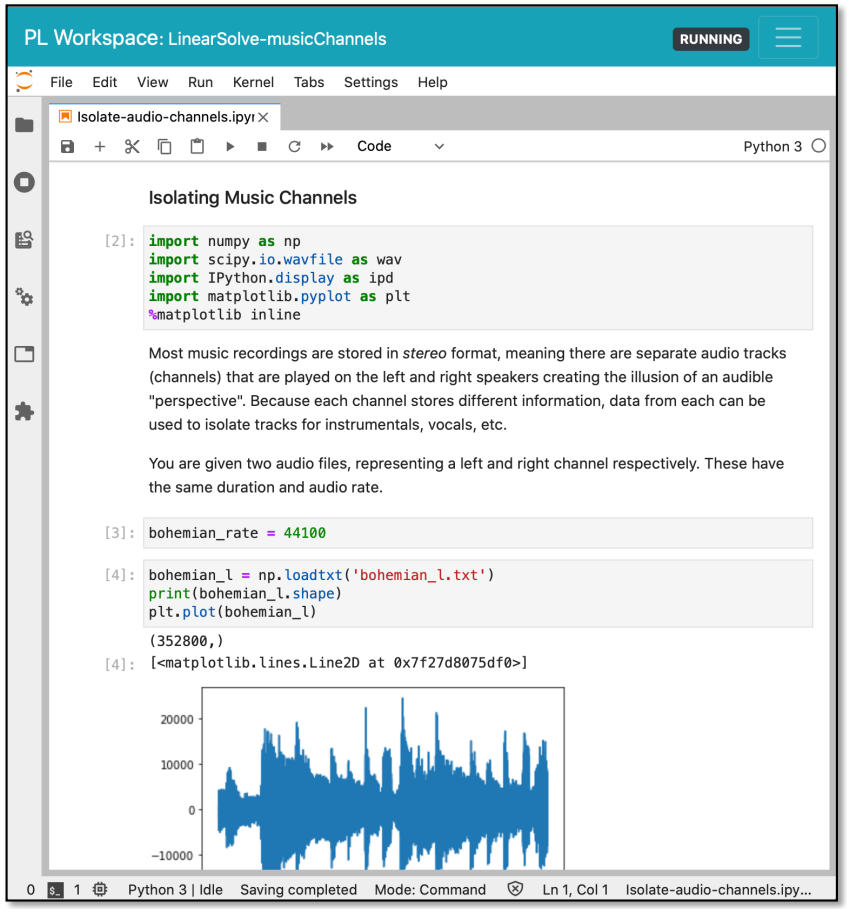
1 Let f be an isomorphism $V(H) \to V(G)$

3 Define $L(H) = \{v | f(v) \in L(G)\}$ and $R(H) = \{v | f(v) \in R(G)\}$

Save & Grade | Save only | New variant

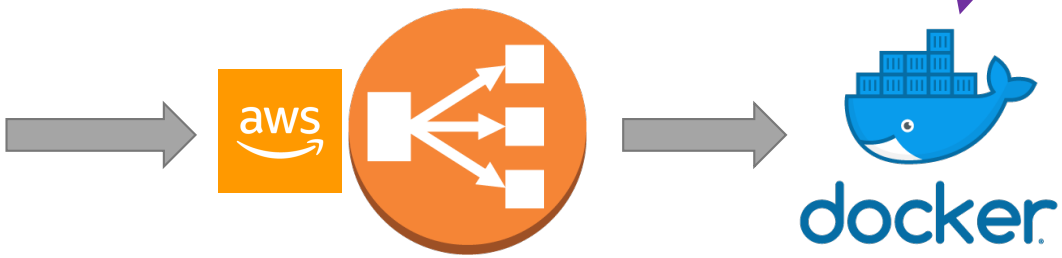
Poulsen et al., "Drag and Drop Proofs With Automated Feedback", 2021 (preprint)

Workspaces and containerized grading

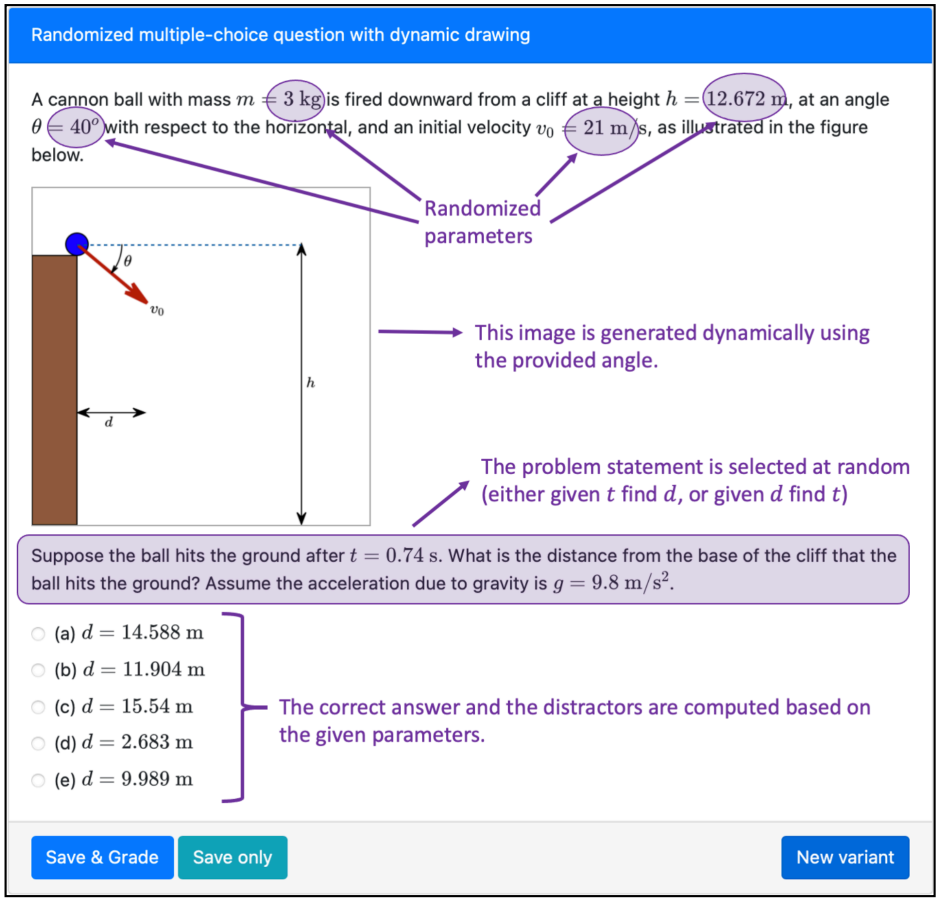


Workspaces run inside PrairieLearn for Jupyter Notebooks, VS Code, R Studio, and more

Safely auto-graded at scale in course-specific containers



Advanced randomization



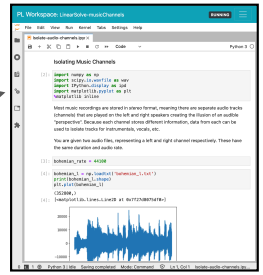
Python can generate HTML to change anything about the question

Write questions once, use everywhere

Computerized question pool

Randomization allows re-use
Auto-grading allows feedback and trying again

Homeworks with immediate feedback

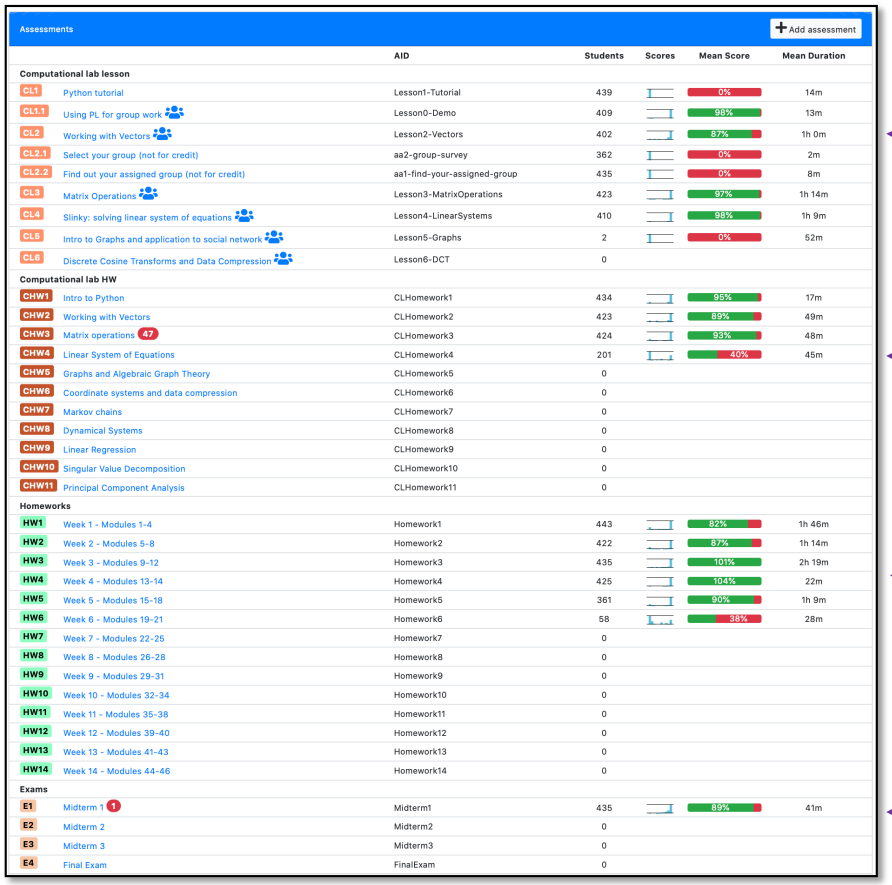


Computation anywhere

Automated exams in proctored facility

Exams for remote students

Case Study: Linear Algebra at Illinois



Assessments — + Add assessment

		AID	Students	Scores	Mean Score	Mean Duration
Computational lab lesson						
CL1	Python tutorial	Lesson1-Tutorial	439		0%	14m
CL1.1	Using PL for group work	Lesson0-Demo	409		98%	13m
CL2	Working with Vectors	Lesson2-Vectors	402		87%	1h 0m
CL2.1	Select your group (not for credit)	aa2-group-survey	362		0%	2m
CL2.2	Find out your assigned group (not for credit)	aa1-find-your-assigned-group	435		0%	8m
CL3	Matrix Operations	Lesson3-MatrixOperations	423		97%	1h 14m
CL4	Slinky: solving linear system of equations	Lesson4-LinearSystems	410		98%	1h 9m
CL5	Intro to Graphs and application to social network	Lesson5-Graphs	2		0%	52m
CL6	Discrete Cosine Transforms and Data Compression	Lesson6-DCT	0			
Computational lab HW						
CHW1	Intro to Python	CLHomework1	434		95%	17m
CHW2	Working with Vectors	CLHomework2	423		89%	49m
CHW3	Matrix operations 47	CLHomework3	424		93%	48m
CHW4	Linear System of Equations	CLHomework4	201		40%	45m
CHW5	Graphs and Algebraic Graph Theory	CLHomework5	0			
CHW6	Coordinate systems and data compression	CLHomework6	0			
CHW7	Markov chains	CLHomework7	0			
CHW8	Dynamical Systems	CLHomework8	0			
CHW9	Linear Regression	CLHomework9	0			
CHW10	Singular Value Decomposition	CLHomework10	0			
CHW11	Principal Component Analysis	CLHomework11	0			
Homeworks						
HW1	Week 1 - Modules 1-4	Homework1	443		82%	1h 46m
HW2	Week 2 - Modules 5-8	Homework2	422		87%	1h 14m
HW3	Week 3 - Modules 9-12	Homework3	435		101%	2h 19m
HW4	Week 4 - Modules 13-14	Homework4	425		104%	22m
HW5	Week 5 - Modules 15-18	Homework5	361		90%	1h 9m
HW6	Week 6 - Modules 19-21	Homework6	58		38%	28m
HW7	Week 7 - Modules 22-25	Homework7	0			
HW8	Week 8 - Modules 26-28	Homework8	0			
HW9	Week 9 - Modules 29-31	Homework9	0			
HW10	Week 10 - Modules 32-34	Homework10	0			
HW11	Week 11 - Modules 35-38	Homework11	0			
HW12	Week 12 - Modules 39-40	Homework12	0			
HW13	Week 13 - Modules 41-43	Homework13	0			
HW14	Week 14 - Modules 44-46	Homework14	0			
Exams						
E1	Midterm 1 1	Midterm1	435		89%	41m
E2	Midterm 2	Midterm2	0			
E3	Midterm 3	Midterm3	0			
E4	Final Exam	FinalExam	0			

Group-work computational labs with Python (Markov chains, image processing, etc)

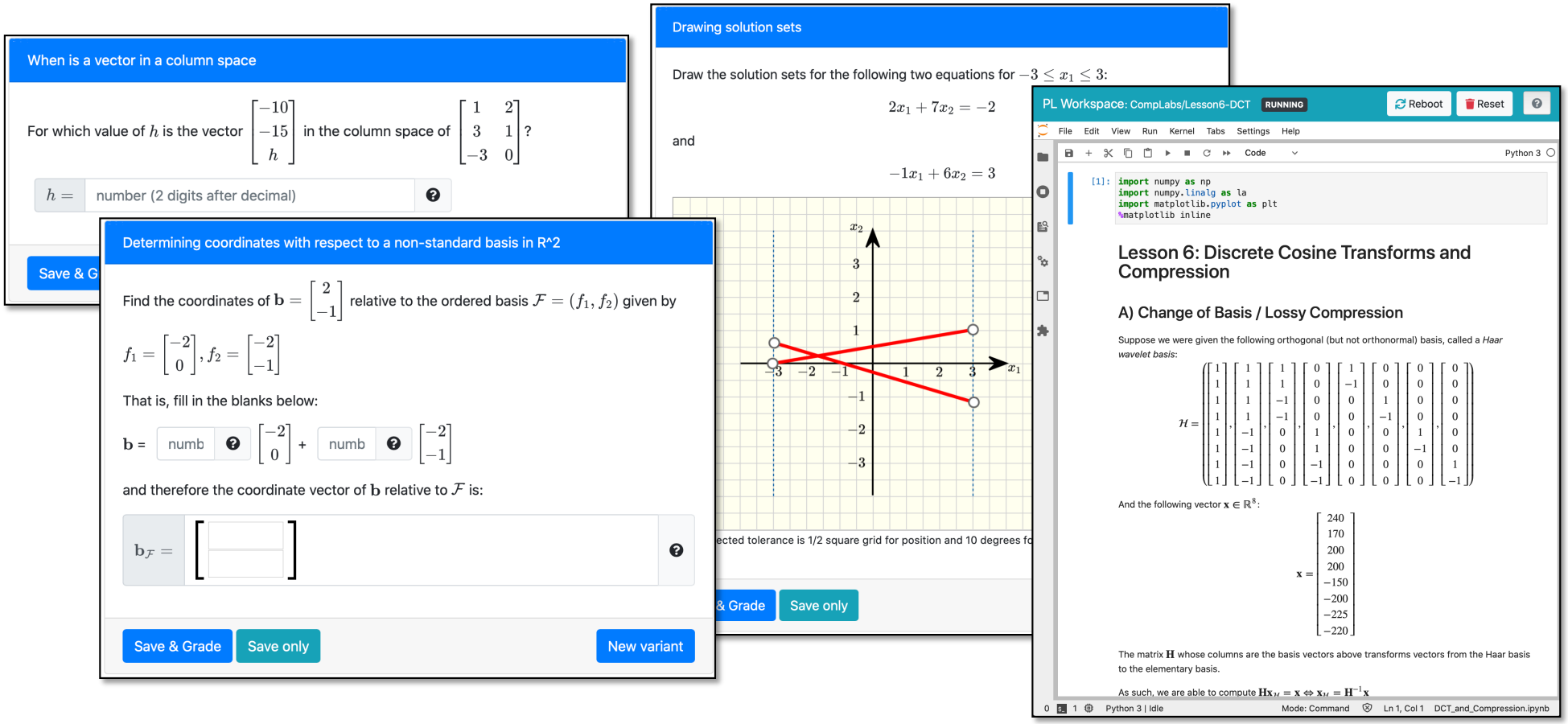
Computational homework

"Regular" homework, unlimited attempts on randomized questions

Randomized, auto-graded exams with declining points

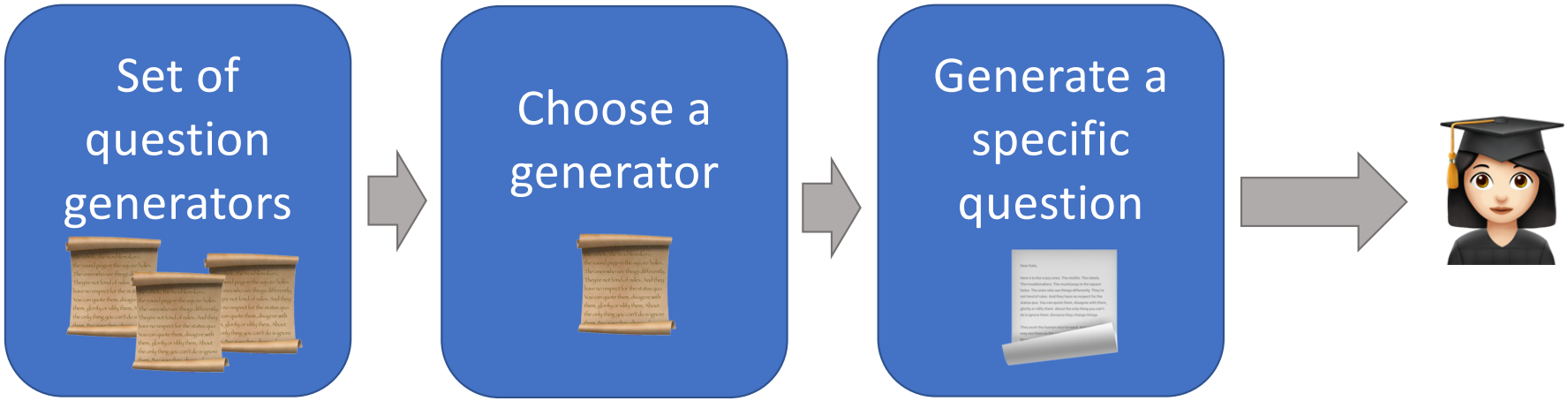
All created by Philipp Hieronymi, Mariana Silva, Nicolas Nytko, et al.

Case Study: Linear Algebra at Illinois



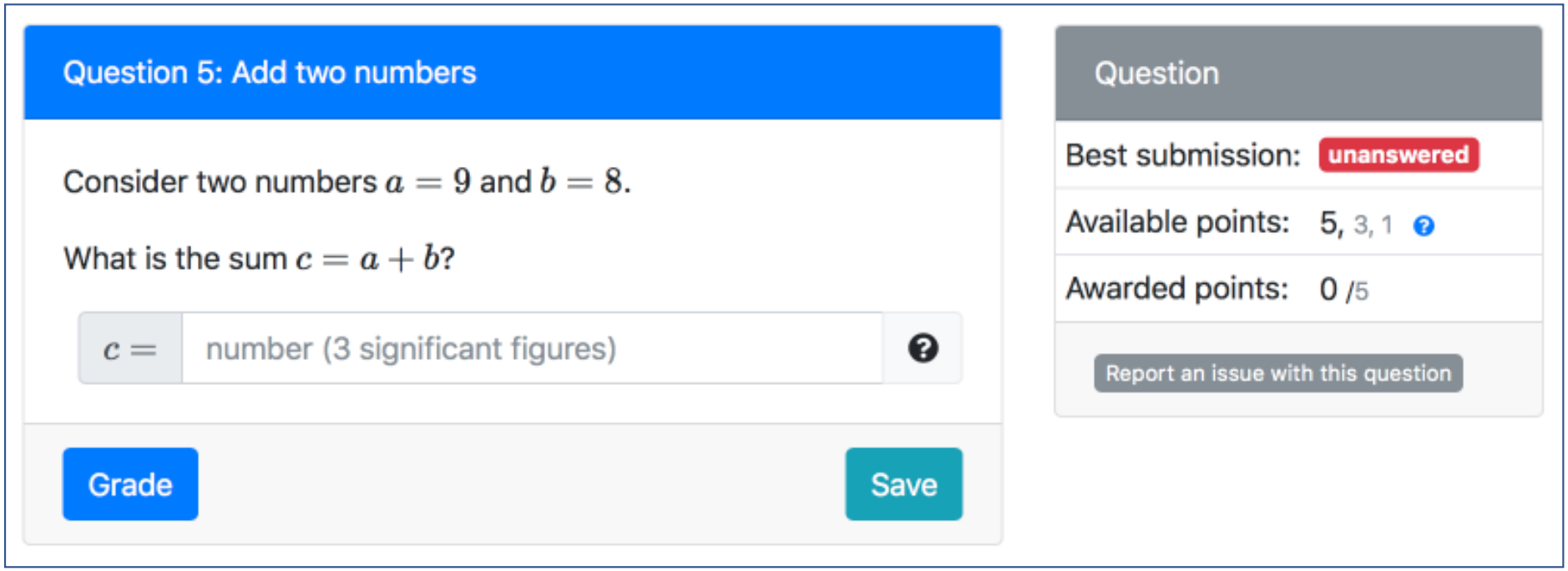
Randomized exam generation

- Randomize question selection as well as question parameters:



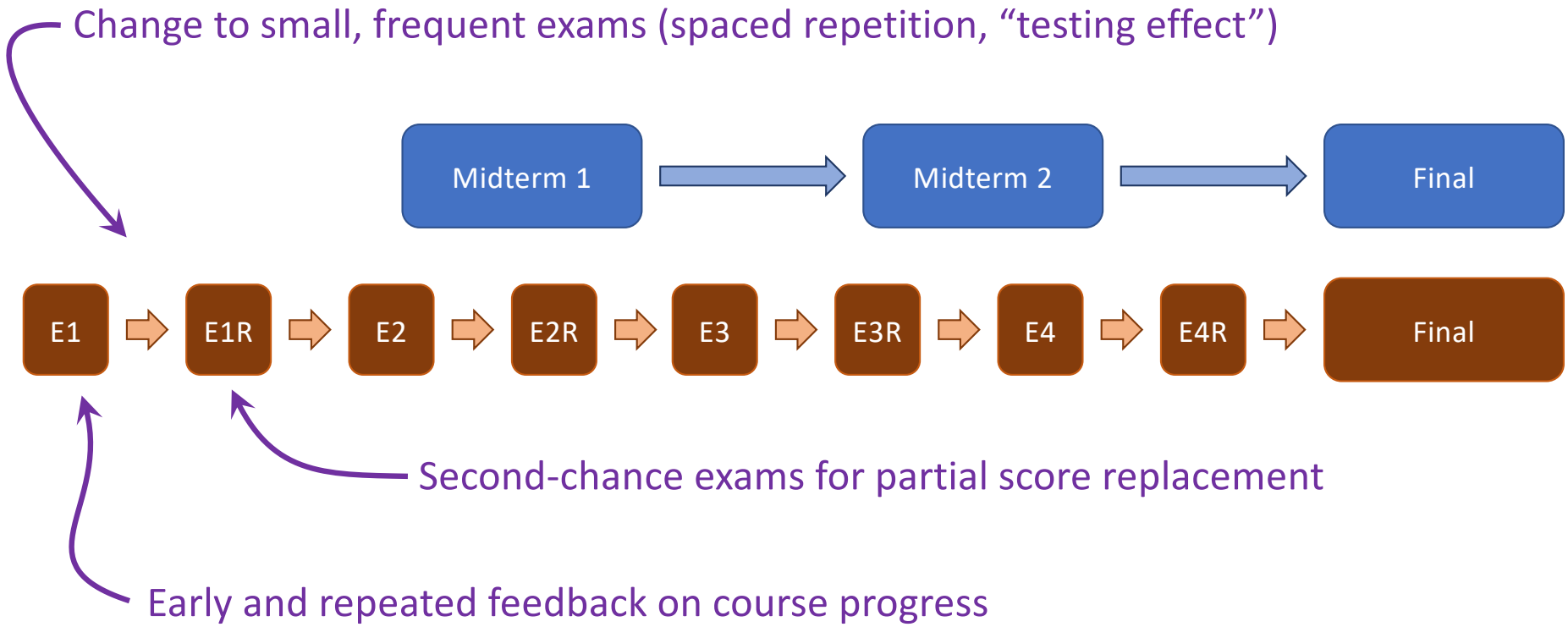
- Statistics before and after exam to ensure fairness and quality

Retries give partial credit with mastery

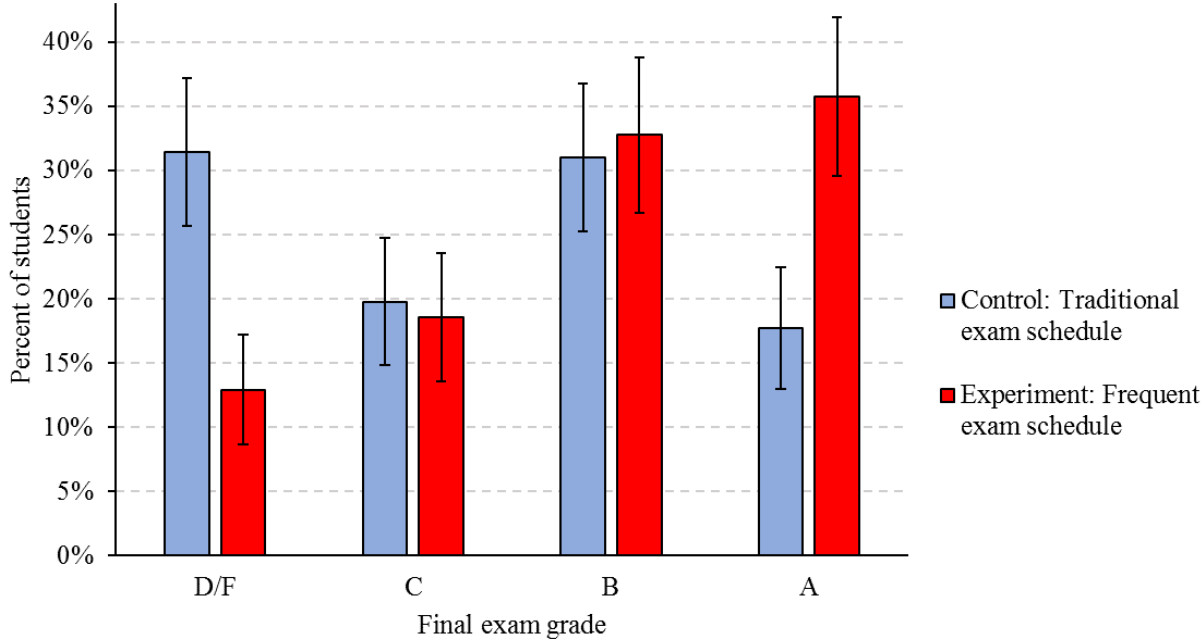


Immediate auto-grading allows trying again for partial credit

Changing the way we test



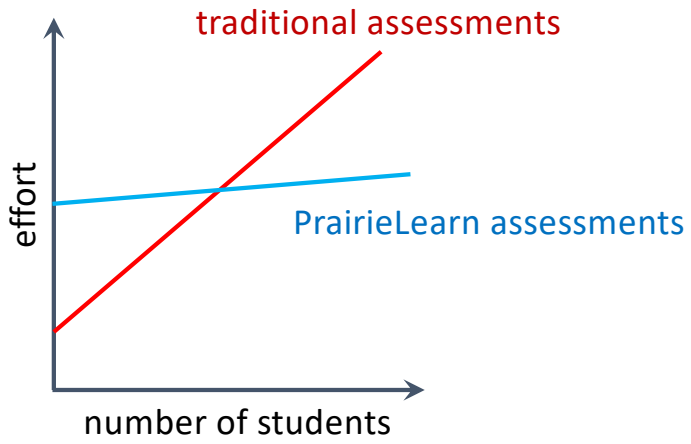
Learning outcomes improve

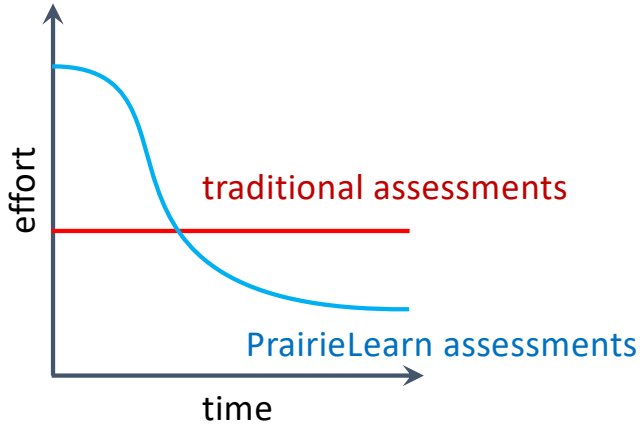


- "Introductory Solid Mechanics": sophomore engineering, 250 students
- Same instructor, same content, same pen-and-paper final exam

Morphew et al., *Applied Cognitive Psychology*, 2020

Most efficient at scale and over time



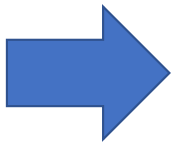


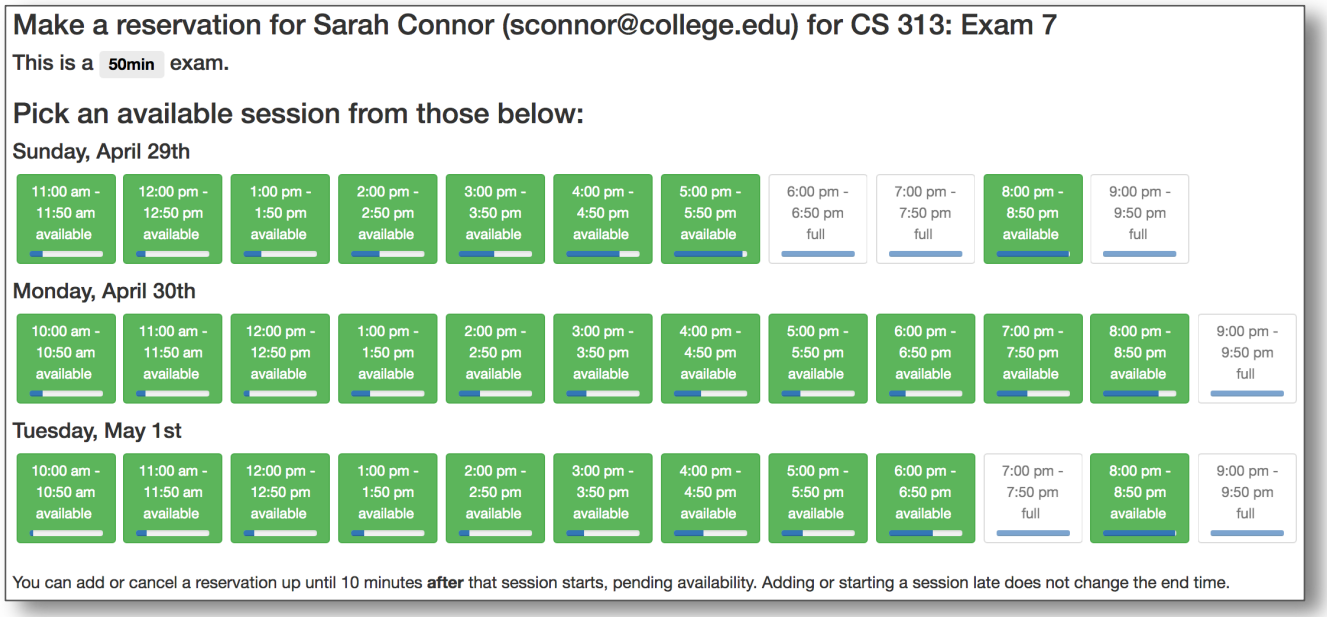
- Immediate benefits for student learning at any scale
- Free up course staff time to help students and create better learning activities

CBTF: Computer-Based Testing Facility

Each exam runs for about 4 days, 10am to 10pm

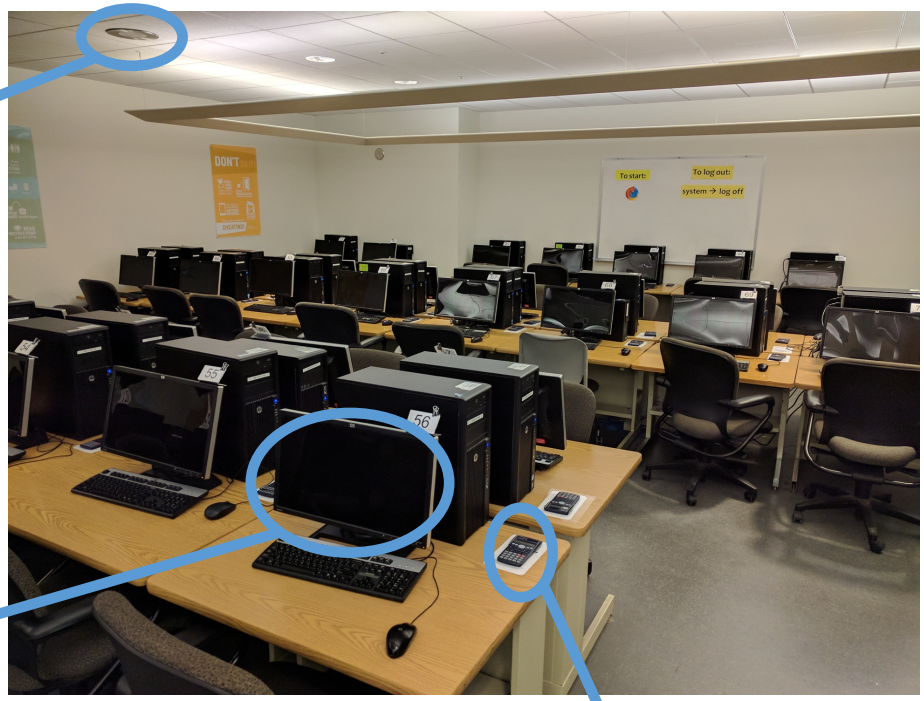
Student email:
Exam is available





Unlimited rescheduling allowed before the scheduled timeslot

Take the exam in a secure environment



Security cameras

Professional proctors

Privacy screens

Calculators

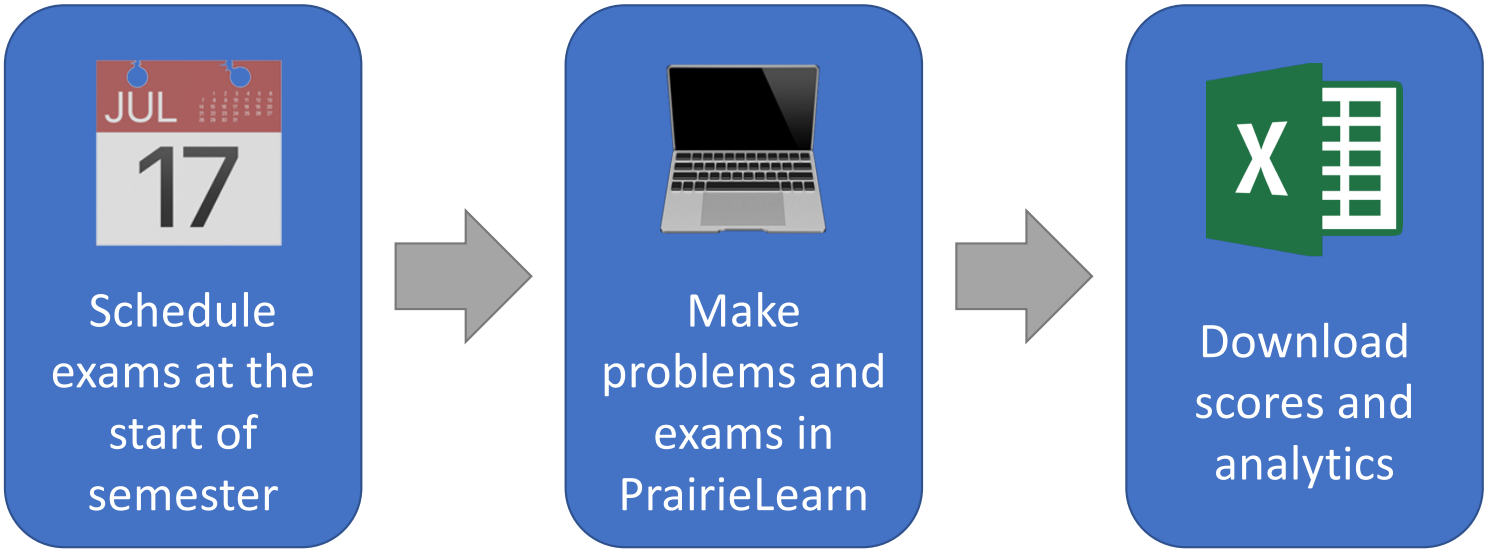
Firewalled internet

Full software platform (Python, RStudio, etc)

ID card swipe to check in

Many different exams concurrently

Instructors focus on exam creation and data



No conflict handling, no proctoring, no sick students, no fuss.

Disability accommodations automatically handled

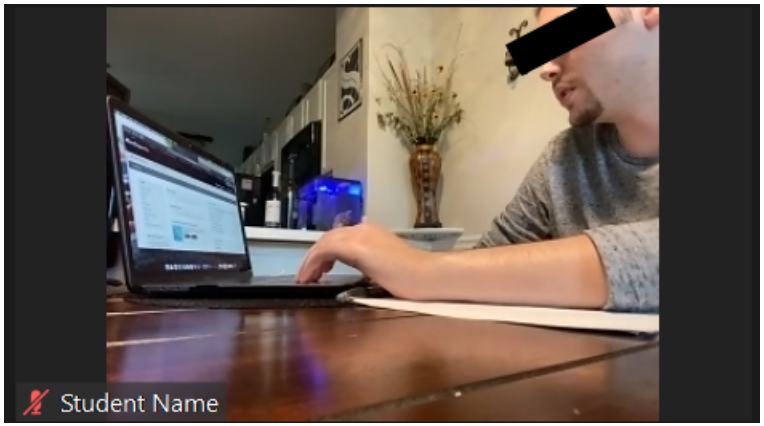
Reduced-distraction seating

Reduced-distraction computers in separate cubicles

Extended-time exams managed by the scheduling software

Computerized exams in the time of COVID-19

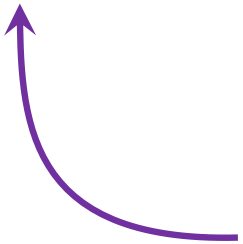
- Up to 10,000 exams per week via dedicated Zoom proctors
- Students do exams on their laptops, using phones for proctor Zoom



- Managed by same scheduling software as the physical CBTF

Want to try PrairieLearn?

- Docs: https://prairielearn.readthedocs.io/
- Live site: https://www.prairielearn.org
- Code: https://github.com/PrairieLearn/PrairieLearn



The hardest thing is spelling "Prairie" correctly