Estimates

Issues with Estimates

- Difficult to do well
- Most often our estimates will be optimistic
- Generally, doesn’t consider the team dynamic
  - Different skills, experience, coordination overhead, personalities, etc.
- Relative productivity is hard to measure without existing data
- Arbitrary increases to estimates is just a guess (back to square one)

Estimates…

- identify “approximately” how difficult a feature may be or how long it “might” take to complete
- **reveal assumptions about the user stories (or other work item)**
  - Record the assumptions and get clarification from the customer
- help clarify user stories
- are done when the estimates converge, and a consensus is reached
- are less effective avoid when people focus on defending their choices rather than discussing thoughts and concerns
- should not be used against members of the team

Work Hours/Days

Pros:

+ Straightforward
+ Preferred to stakeholders with a more traditional business background
+ A precise unit, but not necessarily ”accurate”
+ Easier adjustments for teams that gain or lose members

Cons:

- Implies exactly when something will be finished
- Varies widely from person to person
- Time consuming estimation
- Open to more scrutiny from management
- Interruptions are generally not included in the estimate

Story Points

- An abstraction to hours or days worked
- Focused on the **overall effort**
- The raw values are unimportant (1, 2, 3… or 100, 200, 300… etc.
- Relative values **ARE** important (the ratio between the values)
  - E.g., a story worth 2 points should be twice the effort as a story estimated at 1 and two-thirds the effort of a story estimated at 3 (and so on).
- Must include everything that affects the effort
  - Amount of work to do
  - Complexity
  - Risk or Uncertainty

Story Points

Pros:

- Tend to be more accurate estimates
- Reduce planning time
- Points values stay constant, but the number of points delivered during an iteration can change
- Effort not time is easier to commit to

Cons:

- Point are imprecise
- Work best when you have a stable team
- Can be misused (or used to assign “blame”)
- Management tends to care about their bottom-line: hours and dollars
- Initially more difficult to estimate points per iteration

Planning Poker

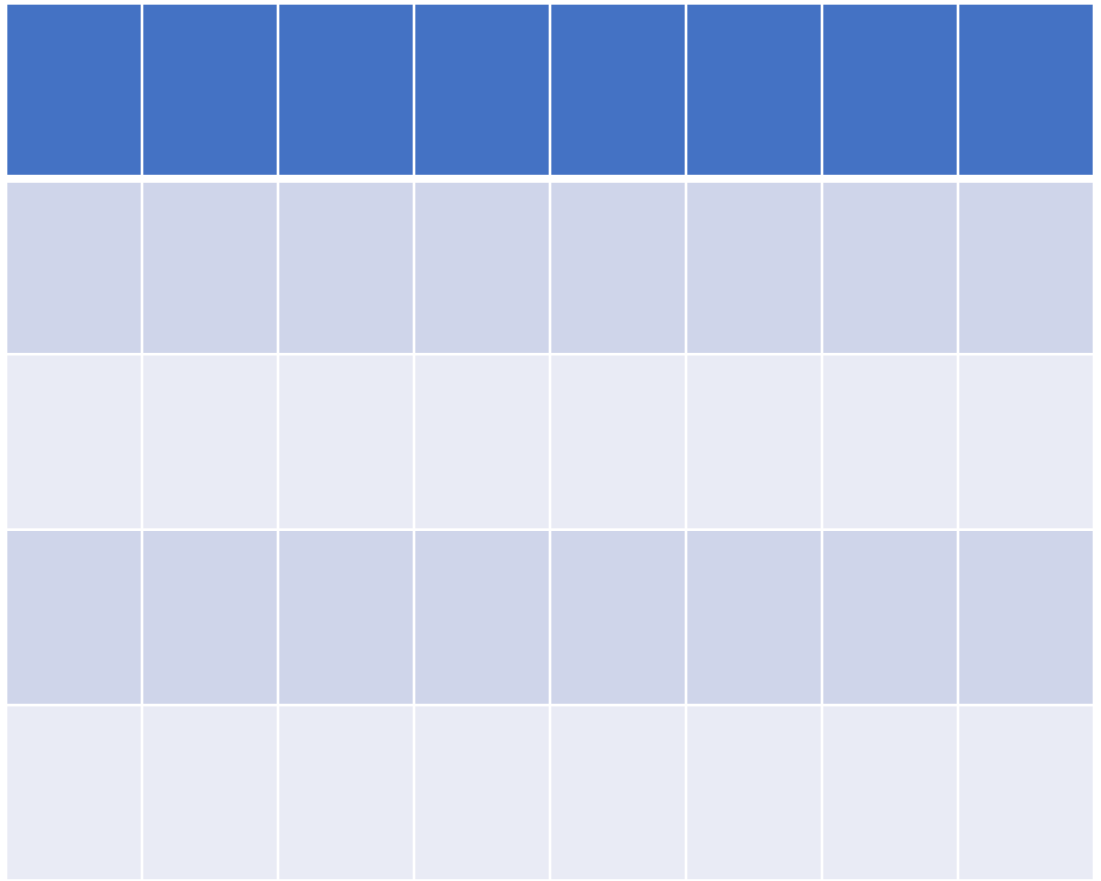
- Each team member gets a small set of cards used to estimate some unit of work
- Each member picks a card and keeps it value hidden
- Cards are revealed AFTER everyone has chosen their estimate value
- If everyone has revealed the same card (a consensus is achieved) then that value becomes the estimate
- Otherwise, EVERYONE discusses their assumptions, concerns, etc. that lead to their estimate value
- After the discussion, another round of planning poker takes place until consensus is reached

Why Planning Poker?

- Replaces getting a verbal estimate from each developer (not ideal)
- Planning poker is a better way pick and reveal estimates
- Revealing estimations simultaneously reduces bias in the estimation process
- Easy to understand
- Can be used for days/hours or story points

Swimlane Estimation

- Create 8 columns and DO NOT LABEL THE COLUMNS
- Split all the work items among the developers and have them **silently** arrange them in the columns
- Indicate that Left-most is “easiest” and right-most is “hardest”



Swimlane Estimation

- Once everyone has placed their work items ask the team to review the board and **silently** move any items if they disagree with the column choice
- Once this is finished ask the team to rate their confidence with the choices.
- If confidence is low, discuss and allow for changes

x	x	x	x	x	x	x	x
	x	x	x	x	x		
		x		x			
				x			

Swimlane Estimation

- Ask the team if they think they will get anything to work on that is smaller (easier) than the items in the left most column.
  - If no, then the first column starts at 1 or 2
  - If yes, then the first column starts at 3 or 5*
- Label the columns

1	2	3	5	8	13	20	!
x	x	x	x	x	x	x	x
	x	x	x		x		x
		x	x				x

*The increments are a modified Fibonacci sequence (after 20 is 40, 100, and then an unknown or infinity)

Swimlane Estimation

- The last column is usually a “!” or some other value to indicate that work item estimate is too large and needs refinement (usually deconstruction)

1	2	3	5	8	13	20	!
x	x	x	x	x	x	x	x
	x	x	x		x		x
		x	x				x

Swimlane Estimation Guidelines

- If people can’t decide on a work item’s position, set it aside for discussion
- 5 is typically a “medium” sized story for an iteration
- Anything greater than 8-13 requires further decomposition
- Seriously consider also decomposing stories of size 8-13
- Anything greater than 20 is (in reality) an *unknown* needing further investigation and decomposition before it can be sized sensibly

Why Swimlane Estimation?

- Lack of talking and concrete numbers avoids the “anchoring” problem
  - an individual speaks up with their view on what size a work item should be ***before*** the remainder of the team have selected their view on the size and influences/anchors all others estimates for related work items to a given point
- Faster than planning poker
- Works best with story points

What about productivity?!

Estimates only show roughly how must time/effort something might take, but not how much work you and your team can get done during an iteration.

Measuring Productivity

- For longer running projects (if there are good records) you can see the performance of past iterations and average the amount of work accomplished
- Using this historical data, we can calculate the **velocity** or average productivity of the team:
  - Total hours/days/story points completed divided by the number of iterations
    - 200 hours / 4 iterations = 50 hours per iteration
    - 75 days / 4 iterations = 18 days of work per iteration
    - 96 story points / 3 iterations = 32 story points per iteration
- What if you don't have any data…

Estimating Velocity in an Iteration

- HFSD states that a velocity of .7 is a safe starting point for your project
- Remember if you have a one-month iteration (30 days), after removing weekends and holidays we only have approximately 20 days
- With respect to total working days:
  - 20 days * .7 velocity = 14 working days
- With respect to an iteration's total user story estimates:
  - 30 days / .7 = 43 days of work with velocity (too much for a 20-day iteration)
  - Usually done this way as it's more intuitive than "removing" workdays.
- If you forget when to multiply or divide, with respect to estimates, the number should always **increase.**

DO NOT ALTER RECORDED ESTIMATES ON YOUR USER STORIES TO INCLUDE VELOCITY!

Adjusting your recorded estimates with velocity means you will need to change all your estimates again if/when velocity changes. Simply note the velocity used for your team.

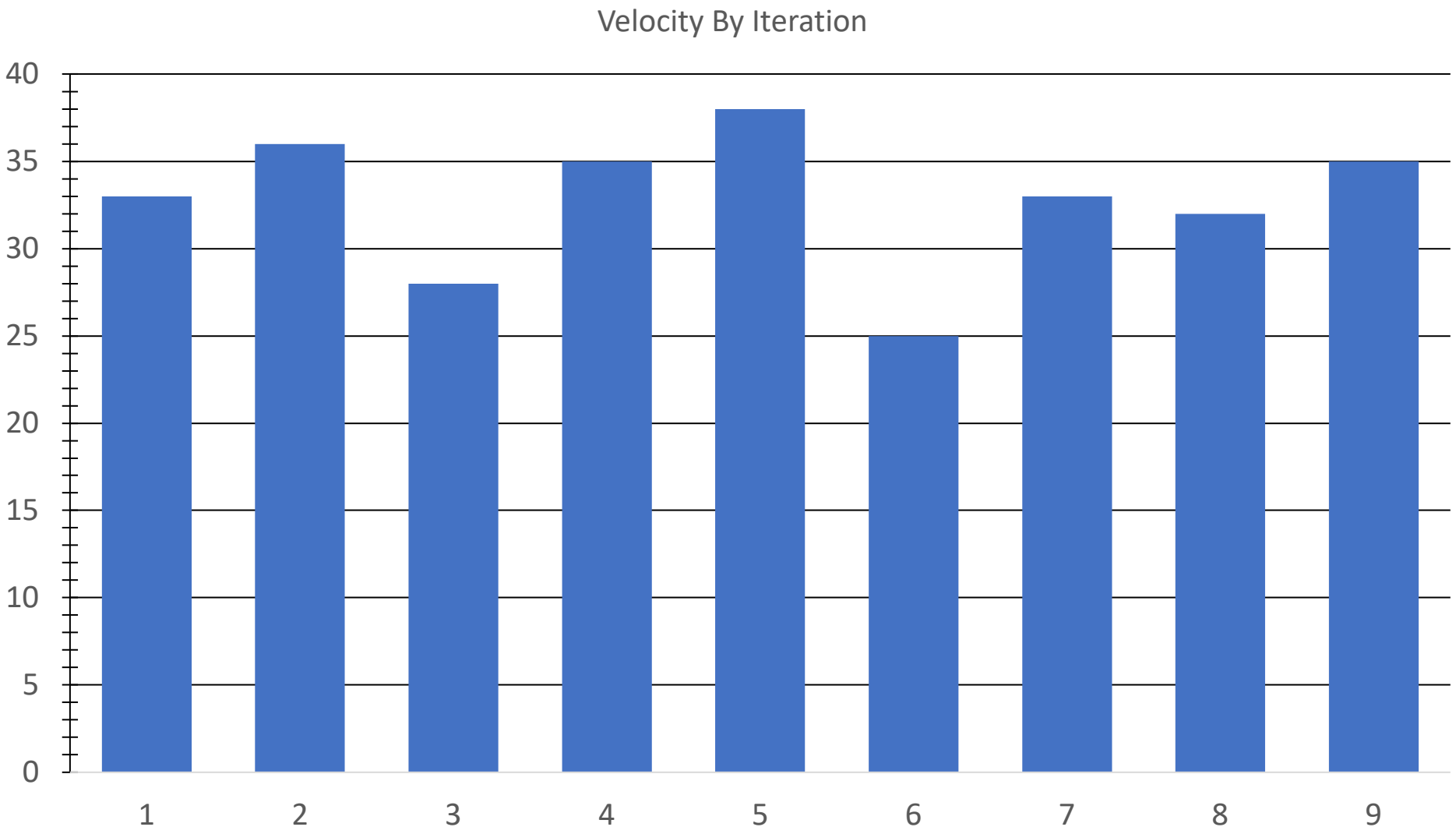
Velocity can change due to…

- Project complexity
- Team size
- Uniformity in team membership
- Team ability to concentrate on user stories and activities
- System outages
- Lack of stakeholder engagement
- Unexpected absences in the team
- Etc.

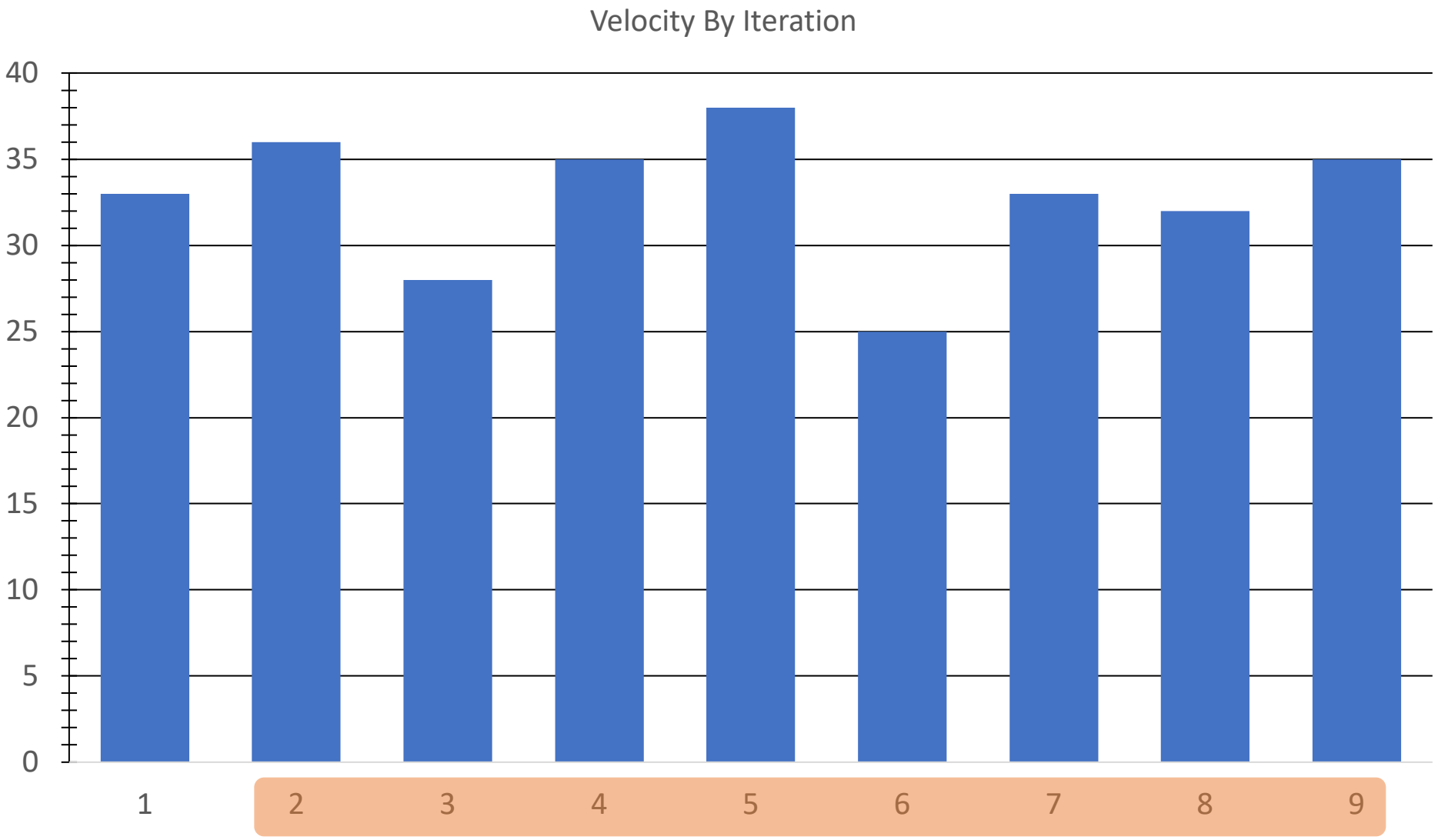
Wait! What if I'm using story points?

- Since story points don't necessarily map one-to-one to time you have to guess for the first few iterations
  - Didn't get everything done the first time? Depending how close you were you can try the same number of points again or less. **DON'T ASSIGN MORE TO CATCH UP.**
  - Done way ahead of time? Assign more points to the next iteration.
- Once you have a few iterations done, then you can take that average and start to get more accurate
- The 0.7 starting velocity is just an estimate. Even with hours/days this can be wrong
- Remember that each iteration is a learning experience. If you record and use this information effectively, it does get easier over time.

The Value of Historical Data

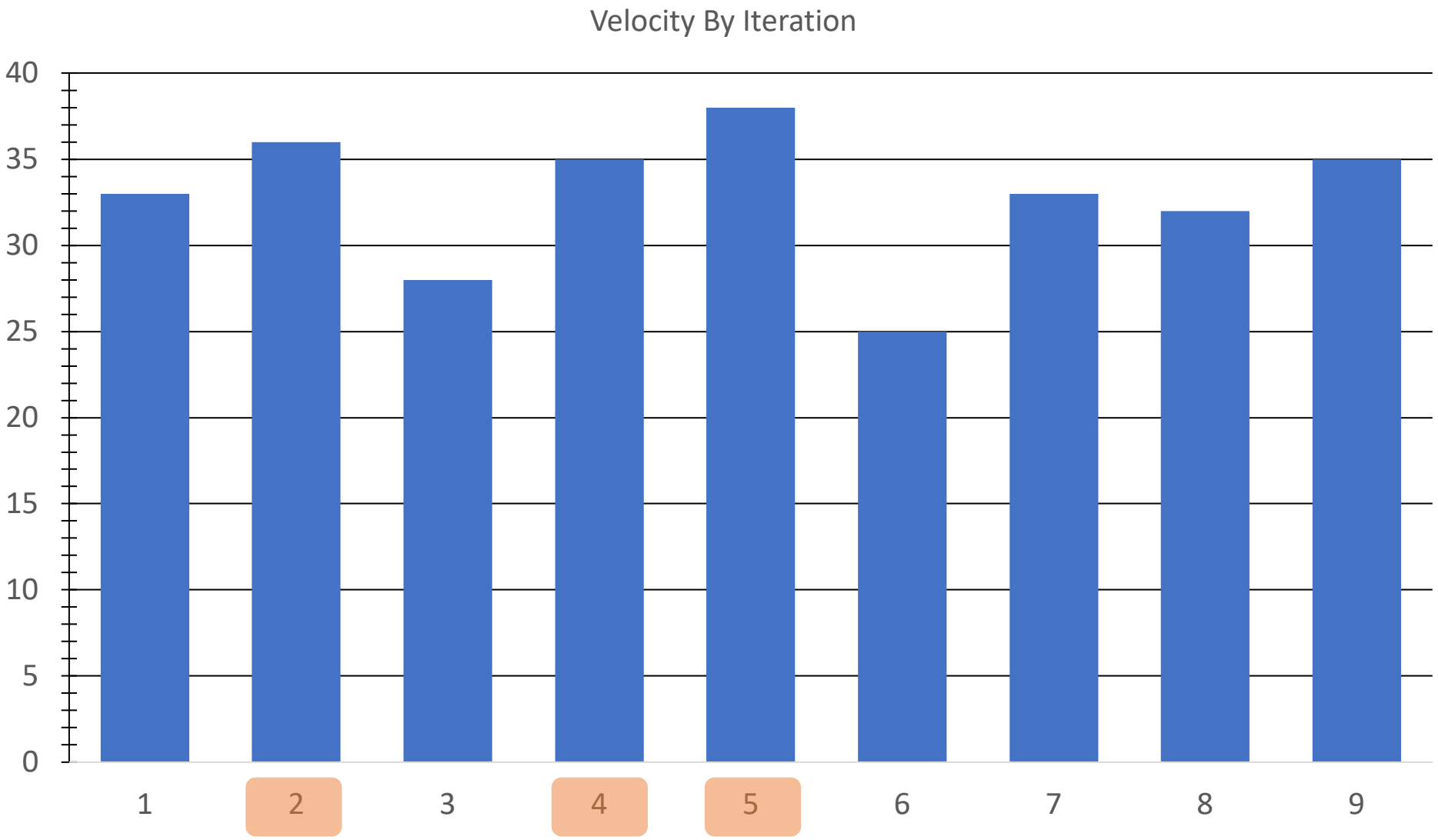


The Value of Historical Data



Avg. Last 8 = 33 points

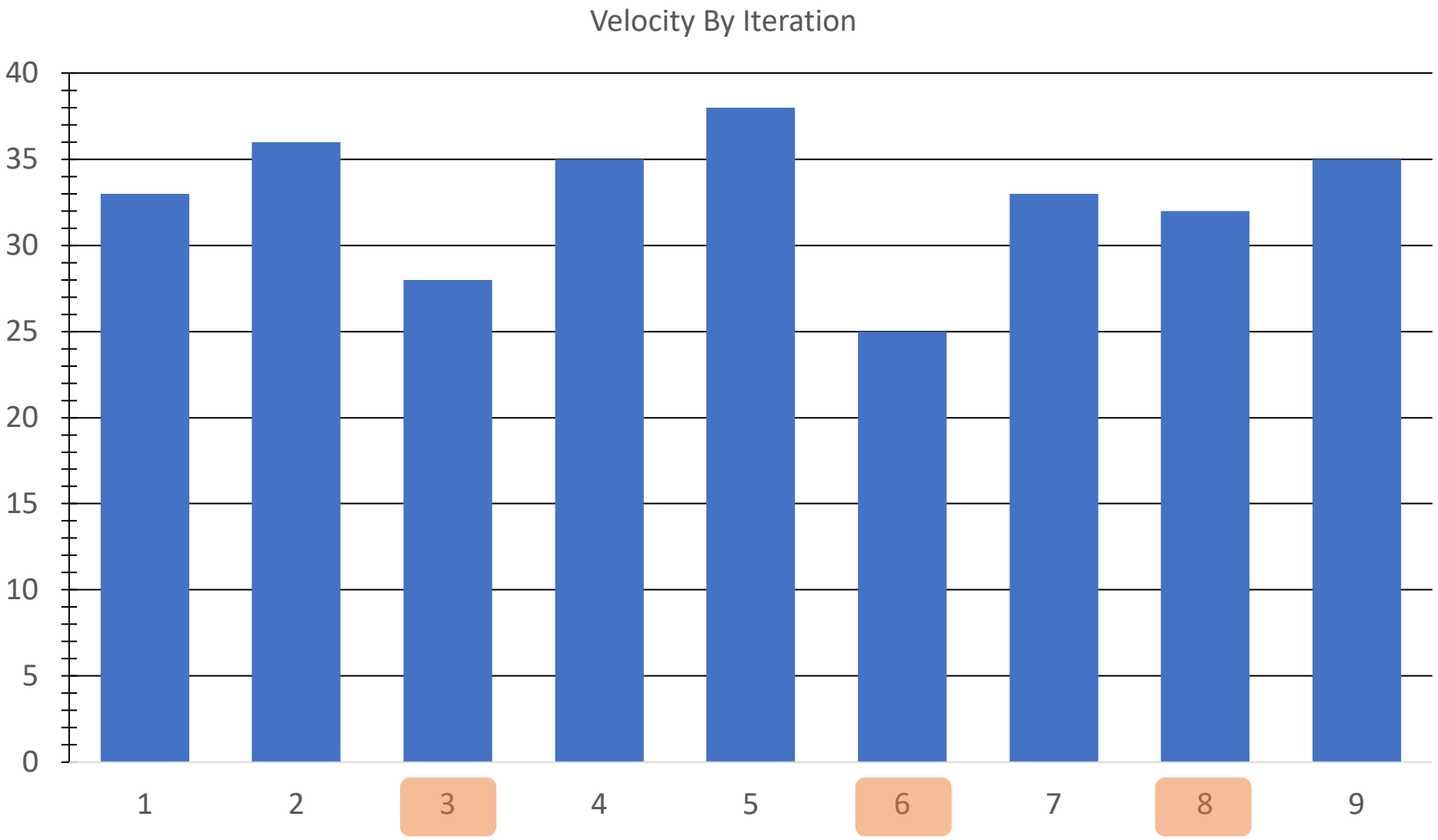
The Value of Historical Data



Avg. Last 8 = 33 points

Avg. Top 3 = 36 points

The Value of Historical Data



Avg. Last 8 = 33 points

Avg. Top 3 = 36 points

Avg. Bottom 3 = 28 points

Can we have a feature in X days?

Work organized by priority

Can we have a feature in X days?

Work organized by priority

Avg. Bottom 3 = 28 points

Avg. Last 8 = 33 points

Avg. Top 3 = 36 points

Can we have a feature in X days?

Work organized by priority

Avg. Bottom 3 = 28 points

Avg. Last 8 = 33 points

Avg. Top 3 = 36 points

Probably

Can we have a feature in X days?

Work organized by priority

Avg. Bottom 3 = 28 points

Avg. Last 8 = 33 points

Avg. Top 3 = 36 points

Probably

Maybe

Can we have a feature in X days?

Work organized by priority

Avg. Bottom 3 = 28 points

Avg. Last 8 = 33 points

Avg. Top 3 = 36 points

Probably

Maybe

No