answerBox

A description of how the answer box works, and it's optional arguments.

How the answer box works

How to generate/use the answer box.

The answer box is the result of the `\answer` command, and is the bread and butter of the math aspect of Xronos. It has a number of optional arguments (covered below) and one required arguemnt - the content author's answer. Thus if you want to provide an open-answer input for students and the correct answer for that spot is x^2+3x+1 you would type `\answer{x^2 + 3x + 1}`. Note that the answer box must be put in mathmode. The previous problem would then look like:

Explanation. *The correct answer is* x^2+3x+1*:* $\boxed{x^2+3x+1}$.

How the answer box validates an answer

It uses more than a dozen algorithms to check if the student's provided answer is mathematically equivalent to the content author's provided answer. Most of these are used as shortcuts to allow faster authentication - for example, if the two answers are exact string matches it will immediately be marked as correct. But if the validation fails (returns false) it proceeds to the next validation method. If every validation method returns false/incorrect, *then* the answer is marked wrong. If *any* of the validators returns "correct", then the answer is marked correct. So the multiple validation techniques are more of a sieve than different validation aspects.

Most noteworthy is that the *final* validator in this sieve is computationally intensive (hence we want to try and do 'quick' checks first) but it is exceptionally powerful. It uses the identity theorem from complex analysis - which essentially says that, if the two functions agree on a subset of $\mathbb{C}$ with an accumulation point, then they are the same function.

In this case, Xronos picks a few hundred points just off the real axis in the complex plane that mimic a converging sequence in some sense (a sequence whose distance between points is strictly decreasing), and tests the student given answer and the author-given answer. If they agree (up to machine precision) on more than 98% of those points, it declares them equal. The nature of machine precision requires the 98%, but essentially, if Xronos marks it correct due to this algorithm, it is *exceptionally* unlikely that the answer isn't a correct answer - especially at the student level we use Xronos (i.e. calc sequence and below).

Optional Arguments

There are also a number of optional arguments/key-values that can be provided to the `\answer` command, for a variety of effects. Most of these are "key-value" pairs, meaning that they have the form "key=value". I outline the key names below and will give examples of values one might use for the given key.

tolerance: The Tolerance key-value sets a $\pm$ value on the author's (numeric) answer that will be accepted from a student. For example, the answer box `\answer[tolerance=0.1]{\pi}` would accept anything in the range of $[\pi - 0.1, \pi + 0.1]$ as correct. This was primarily intended to be used for graphical applications, to allow for "fuzzy answers" on a graph.

validator: This key-value allows you to load a validator in-line from a prior javascript environment. It's important to note here that **this specific validator call** actually uses the Xronos validation system, and not the generic javascript/python call. As a result, it is *far* superior to the validator environment, but it also only works from a single answer box, which makes it difficult to use for more complex problems.

In practice, you would use a javascript environment to define a js function that returns some kind of true/false type value first. Once you have that 'validateFunc' function, you then call it in the answer box as `\answer[validator=validateFunc]{answer-here}` and the validation of the student answer will be entirely decided by the validateFunc you wrote earlier - which may or may not use the 'answer-here' content-author's answer.

id: This is used to set an id for the student's answer, which can be called by a validator or some other code. So something like `\answer[id=x]{}` would take whatever the student input into that answer box and assign it as the value of the python variable "x", which could be called in a validator environment or some other code somewhere. I haven't played around with this much *other* than in custom validator environments.

format: The Format key-value can change what format the answer box expects to receive (and thus how to validate). For example, using the command `\answer[format=string]{test}` let's the answer box know that it is evaluating the student answer as a direct string, rather than an equation. As a result, it would only accept the *string* " test ". Without the "format=string" key-value, any permutation of the letters t, e, s, and t would be accepted as they would be interpreted as variables - thus "et^2s" would have been accepted.

Currently, the only value I remember for this key is the "string" value, but I know there are others in theory.

given: This is largely ignored now, but this dictates whether the answer for the answer box is provided in the corresponding pdf that is generated from

the ximera file. This can be used to make a "teachers edition" version of the notes, rather than a "student" version. For example, the command `\answer[given=true]{\pi}` would display π in the pdf, whereas `\answer[given=false]{\pi` would just show a question mark in the box. Note that given=false is the default behavior.

Potential Pitfalls and Problems

Xronos Errs on the side of the student

Generally, answer validation sides with the student. In most cases this philosophy isn't relevant, but it is noteworthy here because this means that if the `\answer` command can't understand your provided answer, then it will default to mark *any* answer from the student correct. In practice, if you find that you have an answer box that is marking anything/everything correct, this is probably because there is some kind of typo or character in the answer box that it cannot understand.

Answer Box can be a little too good...

Although the validation technique is impressively comprehensive in practice, that can be problematic when you want the answer in a specific *form*. For example, if Xronos will take any function that is mathematically equivalent to the form the author provided, it becomes very difficult to validate whether or not the student's answer is, say, factored. You can often work around this problem with clever question design, or by writing a custom validator to capture the specific property or aspect that you are trying to assess (as it is less important to only validate the equality, and more important to validate the form - often in addition to the equality, of the answer).

Xronos is still running on a computer

Also remember that Xronos is still running on a computer, which means it doesn't *truly* understand real numbers. Indeed, it only really understands computable numbers (which we won't delve into here) but the short version, is that anything that is beyond machine-precision can't really be assessed accurately by a machine. For example, the following answer box is expecting the answer $2x$, but try inputting $2x + 2^{-40}$ and see what happens.

Explanation. $\boxed{2x}$.

Xronos accepts it as correct, even though we obviously know that's not true. But that's because 2^{-40} is a number that is too small for machine precision - so to the computer *running* Xronos, it is effectively indistinguishable from

zero. This is just an inherent restriction based on the machine running Xronos, rather than Xronos itself - but it can be important to be aware of, since it can crop up in other ways. For example, if you have something like $e^{-200|x|^{100}}$, then almost regardless of what number Xronos picks in the complex plane, chances are excellent that the result would be too small for machine precision to detect - meaning that the entire *function* is indistinguishable from the constant zero. This can crop up as a result of integrating or differentiating weird functions that end up having extra terms that it is important for students to realize exist (e.g. you want to verify they used the chain rule and not just the base derivative rule) but the "extra part" doesn't seem to matter to Xronos - this is typically an issue of machine precision.

Answers are exposed via right-click

For accessibility reasons (like screen readers and the like) the library that powers the answer box has a bunch of features bundled in, one of which is the ability to right-click and get different formatting of its contents. Unfortunately, if you right-click and go to "show math as" and then "tex commands" you can very easily see the contents of the answer command that was used in the answer file. This information isn't necessary for accessibility (after all, the answer box is suppose to contain the actual content supplied by the student, and doesn't have any content the student needs to know about) but it's tied to the behavior of the underlying library that supports *all* the rendered math on the page, so we can't remove it without killing *all* accessibility.

However, it is easily countered in the specific case of the answer box, which we detail in the how to hide answers section of the documentation, for those that are interested.