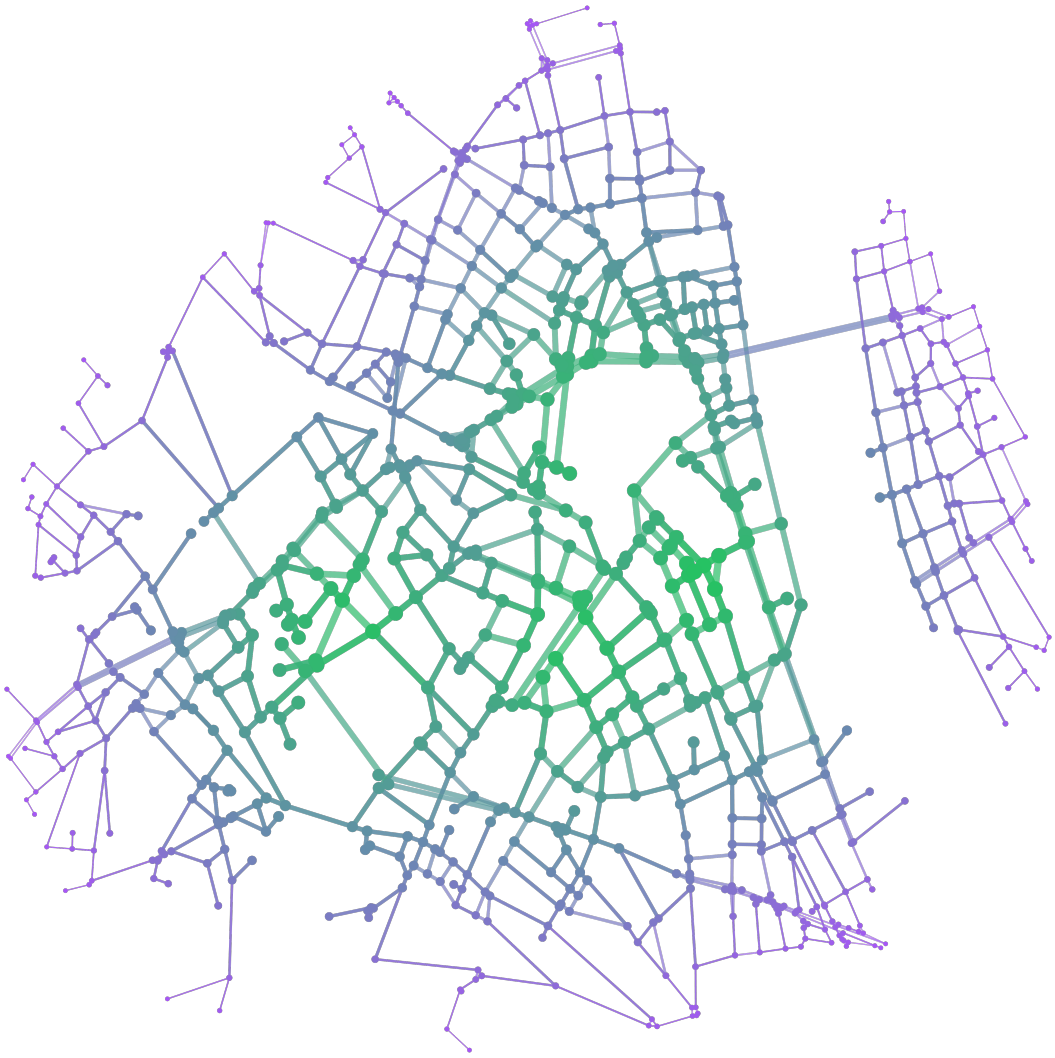
MEB

Mathematical Essays from Bonn



Bonn, 2026.1

Essays

You can see the hills if you look in the right coordinates

Manuel Hinz

Biologists in the field of ion channel gating are particularly interested in **dwell times**, which describe how long a particular ion channel is open or closed. In the process of modeling specific channels, they have to find mathematical explanations that nicely describe their very noisy data. Even in the case without noise, this task is somehow quite hard when approached naively. In this essay, I will explain one of the tricks they use, which most of you will have seen in a different context: **Change of variables.**

In the task I am describing, the biologists measure a certain voltage and write down how long the voltage stays in different regions. This could look something like this:

5 seconds with 0.01V (closed), 1 second with 1.1V (open), 7 seconds with 0.1V (closed), ...

They then gather these times for both the open and the closed case and write down histograms to describe how often a gate was open or closed for some length of time t.

To better understand how they proceed, let's have a look at what the distribution of these dwell times is assumed to be by writing down[1] its probability density function (often called pdf):

$$f_o(t) = \sum_{i=1}^{N_o} a_i \lambda_i e^{-\lambda_i t}$$

In general, density functions are closely related to how likely a certain value of the dwell time is; in practice $b_i(t) \cdot f_o(t)$ is a good approximation for the proportion of values one expects to be in a bin of size $b_i(t)$ around t. If you know about other random variables such as gaussian random variables, the function above has the same role as $C \cdot \exp\left(\frac{-(x-\mu)^2}{\sigma^2}\right)$ for the gaussian (or normal) distribution.

To understand the pdf in our setting, we need to know that all λ_i are positive and the a_i are also positive and sum to one. In some sense, this describes the process of first randomly choosing one of the λ_i, each with probability a_i, and then sampling a value from an exponential distribution with parameter λ_i.

The problem is that the densities look like Figure 1, so it's hard to distinguish mixtures with one or two components. When trying to estimate all the unknown parameters, i.e. N_o, a_i, λ_i, this becomes a problem, especially when the data is noisy! So the ion channel community had to find a solution, which was formalized by Sigworth and Sine [2], whose argumentation we follow to better understand the log transformation.

[1] This needs some assumptions and is due to Fredkin et al. [1]

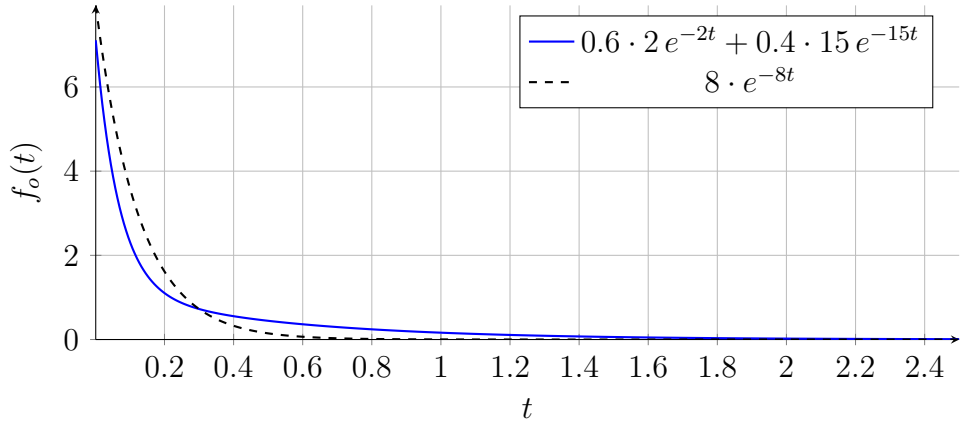


Figure 1: Two densities with a different number of components and different rates.

Instead of looking at the pdf directly, they changed the coordinate system by looking at $G(x) = F(\exp(x))$, where $F(t)$ is the cumulative distribution function, i.e. the integral from 0 to t of f_o.
We will now focus on the $N_o = 1$ case, i.e. we are looking at a exponential random variable with rate λ. Doing so simplifies the math, allowing us to later use the gained insights to make predictions for $N_o > 1$ as well!
To get back a pdf from G, we simply have to do the inverse operation, i.e. differentiate:

$$\begin{aligned} G'(x) = g(x) &= f_o(\exp(x)) \cdot \exp(x) \\ &= \exp(x + \ln(\lambda) - \exp(x + \ln(\lambda))). \end{aligned}$$

We define $z \coloneqq x + \ln(\lambda)$ and $g_0(z) \coloneqq \exp(z - \exp(z))$, so we can write

$$g(x) = g_0(x + \ln(\lambda)) = g_0(z).$$

Luckily, we can understand g_0 quite easily, we can show it has single extremum at 0:

$$g_0'(z) = \underbrace{\exp(z - \exp(z))}_{\neq 0} \cdot \underbrace{(1 - \exp(z))}_{=0 \iff z=0}.$$

And in particular, it is a maximum:

$$\begin{aligned} g_0''(0) &= \exp(0 - \exp(0)) \cdot ((1 - \exp(0))^2 - \exp(0)) \\ &= \exp(-1) \cdot (-1) < -0. \end{aligned}$$

Remember now that $g(x) = g_0(x + \ln(\lambda))$, i.e. it is the same graph just shifted. In particular,

we now know that g has a single maximum at $x_\lambda = -\ln(\lambda)$ with $g(x_\lambda) = e^{-1}$. Notice that the value does not depend on λ!
Now, the idea is that in the logarithmic coordinates, the mixtures are just weighted sums of these *hills*. This allows for two things:

1. Every histogram that does not create a simple hill has more than one component, i.e. $N_o > 1$.

2. Maybe we can infer the number of components by counting the number of hills?

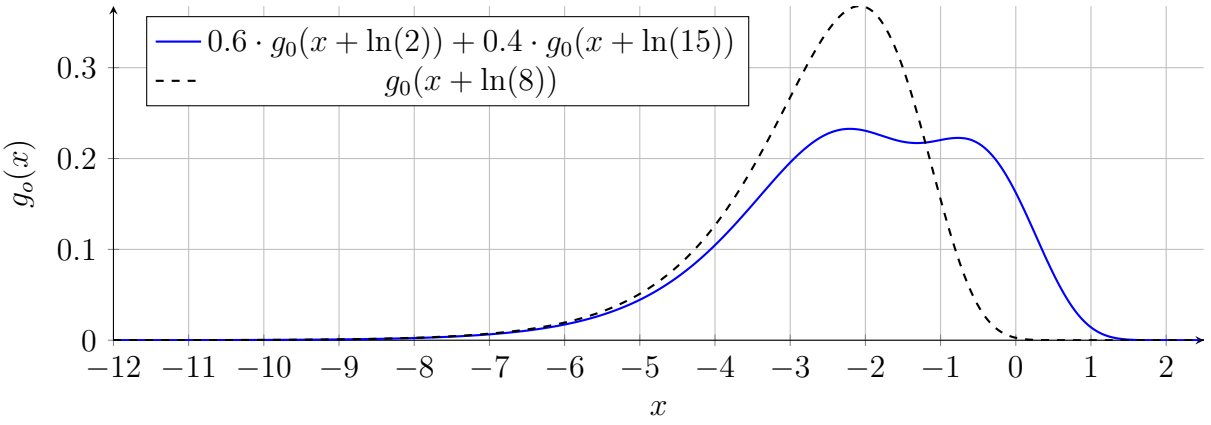


Figure 2: The same two densities, this time log-transformed.

If we look at Figure 2, we can see that we can indeed notice a difference in the shape of the curves between simple exponential and a mixture of exponentials! This second idea is a bit harder to understand and formalize, which does not keep it from being used in practice. I proved the following version of this idea in my master's thesis, formalizing what biologists and biophysicists have been doing for years:

Theorem

Under mild assumptions, the number of open states N_o is bounded from below by the number of maxima M_o and stationary points of inflection (saddle points in one dimension) S_o of the log-transformed pdf g_o of the open dwell times, that is

$$M_o + S_o \leq N_o.$$

Sadly, the proof does not fit in the rest of this essay, but as opposed to a certain proof which might or might not have existed at the time, you can actually read it quite soon by visiting the Library of Mathematics, here in Bonn! I do have to admit that the proof uses some tools from spline theory and includes some annoying calculations, but I also hint at an alternative proof via linear differential operators.

I want to finish this essay with two remarks.

1. I find this application of change of variables $t \to x$ to be very interesting, since they do it to visually inspect histograms and to fit some mathematical model, rather than to compute an integral which is the typical application.

2. If you want to read a real biology paper which uses these very ideas I can really recommend the paper by Tewes et al. [3]. It is a nice read, contains some mystery for those with interest in biology and displays what we have seen here quite nicely in practice.

Manuel Hinz is a master's student currently writing his thesis on the inference of aggregated Markov chains in the biological setting of ion channel gating. He is interested in statistics and its computational challenges, as well as science communication. You can find his website here: `https://mssh.dev`.

References

[1] Donald R. Fredkin, John A. Rice, and Mauricio Montal. Identification of aggregated Markovian models: Application to the nicotinic acetylcholine receptor. In Lucien M. Le Cam, Richard A. Olshen, Ching-Shiu Cheng, Jerzy Neyman, and Jack Kiefer, editors, *Proceedings of the Berkeley Conference in Honor of Jerzy Neyman and Jack Kiefer*, The Wadsworth Statistics/Probability Series, pages 269–289. Wadsworth Advanced Books & Software ; Institute of Mathematical Statistics.

[2] F.J. Sigworth and S.M. Sine. Data transformations for improved display and fitting of single-channel dwell time histograms. 52(6):1047–1054.

[3] Noel Tewes, Beatrice Kubitzki, Flandrit Bytyqi, Nikola Metko, Sebastian Mach, Gerhard Thiel, and Oliver Rauh. Mutation in pore-helix modulates interplay between filter gate and Ba2+ block in a Kcv channel pore. 156(5):e202313514.

Count Graph

Gina Pohlenz

Count Graph (German: *Graf Graph*) is a multi-player game wherein the Commoners (all but one player) must find out the rule their Count imposes on the graphs of the county.

Set-Up

The preparation is rather simple: You only need pen and paper, a whiteboard with markers, a blackboard and some chalk, or some other surface and a suitable writing utensil. Decide who will be the first Count by your preferred method, choose whether you want to play this game with simple, labeled, multi- or hypergraphs and prepare a small tallyboard as shown in Figure 1.

C	
O	

Figure 1: The empty tallyboard

Now, the Count must pick a *rule*, meaning a property of the chosen class of graphs so that the Count can determine in relatively short finite time whether a given drawn graph (i.e., one that isn't too big for the chosen surface) has this property. Then, the Count must draw one graph with the property and another one without, and must mark them as such. This can be done with a ✓ and a ×, respectively. Of course, the Count is free to choose whichever pair of symbols is preferred.

Start Playing!

Now, players take turns at the drawing surface with the goal of figuring out the rule. Each turn consists of at least two and at most three steps:

1. Draw a graph of the agreed-upon class.

   > **Remark.** It helps to test graphs that are close to previously drawn graphs. This helps you pin down the property.

2. Guess whether it fulfills the Count's rule.

   If there are multiple Commoners, talk about what you think. Then, submit your guess to the Count: "We think that this graph [satisfies | does not satisfy] your rule!"

   If and only if your guess is correct, the Count adds a tally to the C-row. Else, the Commoners have to make do with a disappointed Count. In either case, the Count marks the graph with ✓ or ×.

3. While $O < C$, you may propose a feasible rule.

   > **Remark.** "You may" indeed means that you can skip this step and start the next turn immediately.

   Feasible means that no graph drawn so far may contradict the property you are proposing. If your proposed rule is already contradicted, the Count simply tells you so and you may give it another try. In the case that your property is feasible but not what the Count chose, the Count must give you a counterexample. To do so, the Count has two options: The Count may draw a graph that satisfies the Count's rule but not yours; or the Count may draw a graph that satisfies your proposed rule but not the Count's. In either case, the Count marks it with ✓ or ×. Then, a tally is added to the O-row. If still $O < C$, the Commoner may propose another rule. If that's not the case, the next turn starts.

   > **Remark.** If betwen your proposed property and the Count's, one implies the other, then the Count has only one option.

End of Round

A round comes to its end once a Commoner states a property for which the Count cannot draw/give a counterexample.

> **Remark.** This means that one may end up with something merely equivalent to the stated property: The Commoner must not state it exactly as the Count thought of it!

Now, this Commoner becomes the Count and a new round may start!

Rule Inspiration

If you are playing with simple graphs, you could try:

- A graph is complete
- A graph is a path
- A graph is a cycle
- A graph is a tree
- A graph must be bipartite
- Every node has prime degree
- There are no circuits with length smaller than five
- A graph is planar (this may be tough to check for the Count!)
- A graph has as many nodes with even degree as with odd degree

> **Remark.** Some properties may theoretically be hard to verify for the Count, but graphs drawn by hand can usually be checked. If the Count does not know whether some drawn graph has the chosen property, it can simply be skipped.

Modifications

There are different ways of adapting the game. By picking "harder" rules or by starting with misleading examples, the Count may slow down the round. The Count must be aware of the knowledge of the Commoners: If they don't know what a prime number is, it's best not to have them appear in the rule.

The Count may allow the Commoners to "draw"/test a larger set or class of graphs, e. g. "all paths" or "all graphs with more than six nodes.". After such an attempt, the Commoners must guess whether all graphs of this set/class satisfy the rule or whether none of them do. If one of these is the case, step 2 goes as usual. However, if some but not all graphs of the set/class have the Count's property, one tally is added to the O-row as punishment.

You could also make the game harder by playing it competitively. Instead of a shared tallyboard, draw one for each Commoner and keep score individually. Then, step 2 is modified so that instead of a discussion followed by a guess, each Commoner gives a thumbs-up (✓) or thumbs-down (×) on the Count's count of three. Guessing correctly leads to a tally in that Commoner's C-row.

You can also play semi-cooperatively by keeping individual tallyboards but discussing possible rules!

Gina Pohlenz is a Master's student in Mathematics who likes to play around with graphs. The game presented here is inspired by the more haptic Zendo*, which is great fun as well!*

Strengths and Limits of Formalisation: CW complexes as an example

Hannah Scholz

Introduction

Mathematical formalisation is an area of mathematics that has seen exponential growth in the last few years. In simple terms, formalisation is the practice of writing a proof in a programming language and then using the computer to verify the validity of the argument.
This works because of a correspondence between proofs and programs called the Curry-Howard isomorphism. Essentially, a proof that A implies B can be seen as a function that sends a proof of A to a proof of B. So when you write a proof of "A implies B" in a programming language, you are specifying such a function. Checking the validity of the proof then corresponds to checking that this function indeed sends proofs of A to proofs of B.
With the advent of powerful AI tools, formalisation has come to be seen as a magic tool that can lend absolute credibility to anyone using it. Every day we see new claims of solutions to previously unsolved problems supposedly backed by formalisation. Therefore, it is critical that we understand what formalisation can actually be helpful for and where its limits lie.
This is the focus of the following essay. I will explore this topic using, as an example, my formalisation of CW complexes in the language Lean that I did together with and supervised by Floris van Doorn. This text assumes no prior knowledge of formalisation or CW complexes. I will not explain basic terms from topology but the text should still be understandable without knowing them. This essay is heavily inspired by the essay "Why formalize mathematics?" written by Patrick Massot [4]. If this topic interests you, please check out that essay as well!

Defining CW complexes

CW complexes are an important class of spaces from the area of topology. Their importance is owed to the fact that they are very general spaces that at the same time allow for very powerful results. Intuitively, a CW complex is the result of glueing n-cells together along their edges where an n-cell is a continuous image of an n-disc. You can see examples of n-cells of different dimensions in Figure 1.
By gluing these cells together you can build a lot of the spaces that you already know. In Figure 2, you can see that intervals, the real line and spheres are all CW complexes.

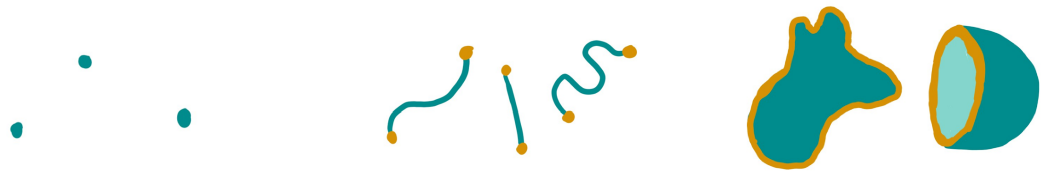


Figure 1: 0-, 1- and 2-cells.

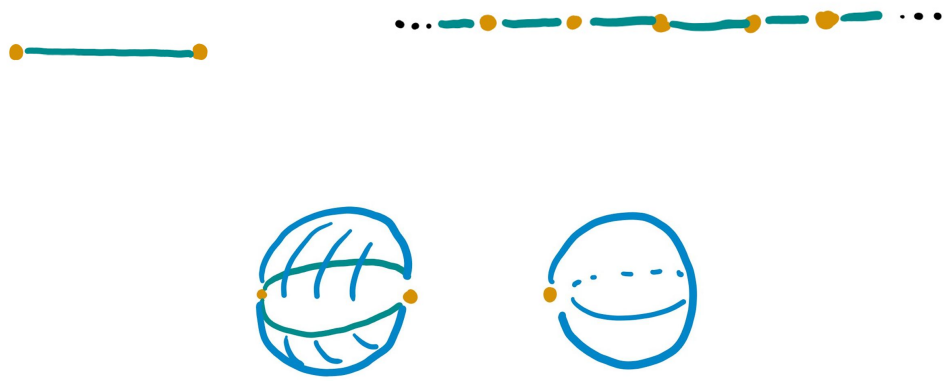


Figure 2: CW structures on the interval, real line and two structures on the 2-sphere.

Now, we should move from this intuition to the concrete definition. I know that it looks very long and scary but don't worry: I will offer an intuitive explanation below and you don't need to understand all of the details.

Definition 1

Let X be a Hausdorff space. An *(absolute) CW complex* on X consists of a family of indexing sets $(I_n)_{n\in\mathbb{N}}$ and a family of continuous maps $(Q_i^n\colon D^n \to X)_{n\in\mathbb{N},i\in I_n}$ called *characteristic maps* with the following properties:

(i) $Q_i^n|_{\mathrm{int}(D^n)} : \mathrm{int}(D^n) \to Q_i^n(\mathrm{int}(D^n))$ is a homeomorphism for every $n \in \mathbb{N}$ and $i \in I_n$. We call $e_i^n :- Q_i^n(\mathrm{int}(D^n))$ an *(open) n-cell* and $\overline{e}_i^n :- Q_i^n(D^n)$ a *closed n-cell*.

(ii) Two different open cells are disjoint.

(iii) For each $n \in \mathbb{N}$ and $i \in I_n$ the *cell frontier* $\partial e_i^n :- Q_i^n(\partial D^n)$ is contained in the union of a finite number of closed cells of a lower dimension.

(iv) A set $A \subseteq X$ is closed if the intersections $A \cap \overline{e}_i^n$ are closed for all $n \in \mathbb{N}$ and $i \in I_n$.

(v) The union of all closed cells is X.

Here D^n refers to the closed unit n-disc. The convention for what the symbol D^n means is different in topology to other fields of mathematics (it is often used to refer to the open unit disc).

Now for a short explanation: Property (i) tells you that the open disk can be deformed in a reasonable way (e.g. no ripping or gluing). Note that the notation for the closed n-cell $\overline{e}_i^n$ is very confusing: it looks like the closed cell is the closure of the open cell but this is not the case in general. Property (iii) is called "closure finiteness" (which is the "C" in "CW complex"). It tells you that the cell frontier needs to be glued to cells of lower dimensions and that the new cell cannot stretch over infinitely many lower cells. Again, the notation for the cell frontier is confusing: it is not generally the frontier of the open cell. The "W" in "CW complex" stands for "weak topology". This is Property (iv). It tells you that the topology on X is determined by the cells. Lastly, Property (v) tells you that the whole space is made up of the cells.

Let us proceed to the Lean version of this definition which you can find in Figure 3. Again, this is very long and surely overwhelming if you do not know any Lean. I will try to offer an explanation below. You don't need to understand this line by line.

```
class CWComplex.{u} {X : Type u} [TopologicalSpace X] (C : Set X) where
  cell (n : ℕ) : Type u
  map (n : ℕ) (i : cell n) : PartialEquiv (Fin n → ℝ) X
  source_eq (n : ℕ) (i : cell n) : (map n i).source = ball 0 1
  continuousOn (n : ℕ) (i : cell n) : ContinuousOn (map n i) (closedBall 0 1)
  continuousOn_symm (n : ℕ) (i : cell n) :
    ContinuousOn (map n i).symm (map n i).target
  pairwiseDisjoint' :
    (univ : Set (Σ n, cell n)).PairwiseDisjoint
        (fun ni ↦ map ni.1 ni.2 '' ball 0 1)
  mapsTo' (n : ℕ) (i : cell n) : ∃ I : Π m, Finset (cell m),
    MapsTo (map n i) (sphere 0 1) (⋃ (m < n) (j ∈ I m),
        map m j '' closedBall 0 1)
  closed' (A : Set X) (hAC : A ⊆ C) :
    (∀ n j, IsClosed (A ∩ map n j '' closedBall 0 1)) → IsClosed A
  union' : ⋃ (n : ℕ) (j : cell n), map n j '' closedBall 0 1 = C
```

Figure 3: The definition of an absolute CW complex in Lean.

The first line says that we want to define the class of CW complexes `C` in a topological space `X`. Lean works with type theory instead of set theory, so you will see the word `Type` appear a lot in the code. Just mentally replace it with the word "Set". So, `{X : Type u}` means that `X` is a set. `[TopologicalSpace X]` means that `X` is a topological space and `(C : Set X)` means that `C` is a subset of `X`. What the different brackets mean is not important here.

You can see that this is already different from Definition 1 since we are defining CW complexes as subspaces instead of the spaces themselves. The reason for this change is that a lot of CW complexes have a natural ambient space whose topology is easy to understand. For example, in Figure 2 I drew CW structures on the interval and the 2-sphere. These have the natural ambient spaces $\mathbb{R}$ and $\mathbb{R}^3$ respectively which have nicer topologies than the subspace topologies induced on our CW complexes. When we do maths on paper, this is a detail that we would just gloss over and ignore in most of our proofs. However, in formalisation we need to do all proofs in full detail and small issues like these can become painful. This leads us to our first limit:

> **Limit 1**
>
> Sometimes, a mathematical definition or statement needs to be modified to work well for formalisation.

Why is this a limit? This is because whatever statement about CW complexes I want to formalise, it will never be exactly the same as the original paper statement because we are working with a different definition. So the reader of my Lean code needs to check for themselves that my definition is still equivalent to the definition on paper.

Now, let's get back to our code in Figure 3. In line 2, we are defining the indexing sets `cell n`. These are just what we called I_n in Definition 1. In line 3, we define the characteristic maps `map n i` that correspond to Q_i^n from Definition 1. We define them as `PartialEquiv (Fin n → ℝ) X` where `Fin n → ℝ` is just a weird way to write $\mathbb{R}^n$. A `PartialEquiv` is a little bit like a pair of inverse maps: it consists of two maps in opposite directions. In this case, we have the forward map called `map n i` going from $\mathbb{R}^n$ to `X` and the backward map called `(map n i).symm` going in the opposite direction. What makes this a *partial* equivalence is that these two maps are only inverses when restricted to certain sets in the domain and codomain. The relevant set in the domain, i.e. here $\mathbb{R}^n$, is called `(map n i).source` and the relevant set in the codomain, i.e. `X`, is called `(map n i).target`. For this definition to make sense, we require that the set in the codomain is the image of the set in the domain under the forward map. Why do we use such a weird definition? Again, constantly restricting to subsets is surprisingly annoying, so we prefer to have maps defined on our entire space. So this is another instance of Limit 1.

In line 4, we say that the specific set in $\mathbb{R}^n$ on which the maps are inverses is the open unit ball. This is just another way of saying that the restriction of `map n i` to the open unit ball is a bijection.

Then in line 5, we require `map n i` to be continuous on the closed unit ball and in lines 6 and 7, we require the backwards map `(map n i).symm` to be continuous on the set `(map n i).target` which in this case, as explained above, is just the image of the open unit ball under `map n i`. So all together, lines 3 to 7 capture Property (i) of Definition 1. We can observe our first strength here:

Strength 1

Formalised definitions and notation are unambiguous.

Where before the notation D^n could mean different things to different mathematicians, `closedBall 0 1` is defined in the mathematical library "Mathlib" of Lean and by clicking on it in a code editor, you will get sent to the precise definition of `closedBall` which is

```
def closedBall (x : α) (ε : ℝ) := { y | dist y x ≤ ε }
```

i.e. all the points `y` that have distance to `x` less than or equal to ε. So even if the name was something unhelpful like `D` instead of `closedBall`, the meaning would still be clear.
For you to understand the next limit that I want to present, I first need to tell an embarrassing anecdote about line 4 of the Lean definition: for almost a year after I started working on this project (so in particular during my entire Bachelor thesis that covered it), this line mistakenly read

```
source_eq (n : ℕ) (i : cell n) : (map n i).source = closedBall 0 1
```

which requires the characteristic map to be a bijection on the entire closed ball. This in particular excludes examples like the second construction on the 2-sphere in Figure 2 where the cell frontier is mapped to a single point. I only figured this out when I tried to implement the examples shown in the figure and this particular example was not possible. The limitation here is an extension of Limit 1:

Limit 2

Formalisation cannot verify a definition, i.e. it cannot check that the formalised notion captures the paper definition.

Of course, the "wrong" definition that I wrote wasn't wrong in any mathematical sense. Like in paper mathematics, you are free to define whatever object you want in Lean. I had just accidentally defined a more restrictive version of CW complexes. This issue can happen to humans and it often happens to AIs. It can be used to make it seem like you have proven an amazing complicated statement when, in fact, three files deep a definition is translated incorrectly which makes the final statement trivial.
Sometimes, a mismatch between the paper and Lean definition is even intentional. You might have noticed that the Lean definition in Figure 3 does not require the space to be Hausdorff. This is because you can define the structure entirely without it and by not including it, users of my code have more freedom about the order in which they choose to prove things. However, this means that the correct way to state "X is a CW complex" in Lean is `[CWComplex X] [T2Space X]` (where "T_2-space" is another name for Hausdorff spaces). This could lead to misunderstandings if you don't check the definitions carefully or read the documentation.

Let us finish discussing the Lean definition in Figure 3. The lines that we have not discussed are mostly just translations of our properties from Definition 1. Their precise statements are not particularly important so I will just tell you which statement they correspond to. Don't worry about understanding the code. Lines 8 to 10 correspond to Property (ii), lines 11 to 13 describe closure finiteness i.e. Property (iii), lines 14 and 15 describe the weak topology i.e. Property (iv) and the last line corresponds to Property (v).

Proving things about CW complexes

Next, I want to come to the most obvious strength of formalisation:

> **Strength 2**
>
> Formalisation can verify whether a proof is correct.

This can be useful for a number of reasons that I wish to discuss in this section. Firstly, this is very convenient for tedious proofs. For my thesis, I was proving a lot of very basic lemmas that had very annoying proofs. Before I can present an example, I need to give a definition first.

> **Definition 2**
>
> Let X be a CW complex in the sense of Definition 1. We define $X_n := \bigcup_{m<n+1} \bigcup_{i\in I_m} \overline{e}_i^m$ and call it the *n-skeleton* of X for $-1 \leq n \leq \infty$.

A simple but important fact is that a CW complex cannot just be expressed as the union of its closed cells but also as the union of its open cells. This fact is derived from the following lemma.

> **Lemma 3**
>
> $X_n = \bigcup_{m<n+1} \bigcup_{i\in I_m} e_i^m$ for every $-1 \leq n \leq \infty$.

I have put the proof of this lemma in Figure 4. Please don't try to actually go through it. This is just to illustrate how annoying these proofs are when you actually write out all the details. No one doubts that the statement of Lemma 3 is correct but (to my knowledge) no books write out proofs like these in excruciating detail either. So if you want a detailed proof, it is left to you to figure out and Lean can be of help here. I write proofs like these directly in Lean because I can think more clearly about my next step when I don't need to worry about the last one.
Of course, this is more of a toy example but Strength 2 has started to become useful in research mathematics: in 2020, Peter Scholze posed a challenge called the "Liquid Tensor Experiment" to the formalisation community [5]. He wanted a proof from his work about condensed mathematics verified because he still had slight doubts about its correctness. After the relevant proof had been formalised in an effort led by Johan Commelin, he wrote another blog post [6] expressing his satisfaction and explaining that the formalisation deepened his understanding of the maths involved and led to simplifications. We can extract another strength here:

We show this by induction over $-1 \leq n \leq \infty$. For the base case, assume that $n = -1$. Then we get $X_{-1} = \bigcup_{m<0} \bigcup_{i \in I_m} \overline{e}_i^m = \varnothing = \bigcup_{m<0} \bigcup_{i \in I_m} e_i^m$.
For the induction step, assume that that the statement is true for n. We now show that it also holds for $n+1$.

$$
\begin{aligned}
X_{n+1} &= \bigcup_{m<n+2} \bigcup_{i \in I_m} \overline{e}_i^m \\
&= \bigcup_{i \in I_{n+1}} \overline{e}_i^{n+1} \cup \bigcup_{m<n+1} \bigcup_{i \in I_m} \overline{e}_i^m \\
&= \bigcup_{i \in I_{n+1}} \overline{e}_i^{n+1} \cup X_n \\
&\overset{(1)}{=} \bigcup_{i \in I_{n+1}} \overline{e}_i^{n+1} \cup \bigcup_{m<n+1} \bigcup_{i \in I_m} e_i^m \\
&= \bigcup_{i \in I_{n+1}} e_i^{n+1} \cup \bigcup_{i \in I_{n+1}} \partial e_i^{n+1} \cup \bigcup_{m<n+1} \bigcup_{i \in I_m} e_i^m \\
&\overset{(2)}{=} \bigcup_{i \in I_{n+1}} e_i^{n+1} \cup \bigcup_{m<n+1} \bigcup_{i \in I_m} e_i^m \\
&= \bigcup_{m<n+2} \bigcup_{i \in I_m} e_i^m
\end{aligned}
$$

Note that (1) holds by induction and (2) holds by closure finiteness.
Now we can move on to the case $n = \infty$.

$$
\begin{aligned}
X_\infty &= \bigcup_{m<\infty+1} \bigcup_{i \in I_m} \overline{e}_i^m \\
&= \bigcup_{m<\infty+1} \bigcup_{l<m+1} \bigcup_{i \in I_l} \overline{e}_i^l \\
&= \bigcup_{m<\infty+1} X_m \\
&\overset{(1)}{=} \bigcup_{m<\infty+1} \bigcup_{l<m+1} \bigcup_{i \in I_l} e_i^l \\
&= \bigcup_{m<\infty+1} \bigcup_{i \in I_m} e_i^m
\end{aligned}
$$

Where (1) holds by induction.

Figure 4: The proof of Lemma 3.

Strength 3

Formalising can clarify and aid understanding of the mathematics.

I have experienced this on a smaller scale with the proof of the main theorem in my thesis. This theorem says that the product of two CW complexes is a CW complex under certain conditions. To state this theorem we first need another definition:

Definition 4

Let X be a topological space.

(i) We call X *compactly coherent* if a set $A \subseteq X$ is open iff for all compact sets $C \subseteq X$, the intersection $A \cap C$ is open in C.

(ii) We can also turn an arbitrary X into a compactly coherent space by passing to a fine enough topology (the precise definition of which is irrelevant). We call the resulting space the *compact coherentification* of X.

The two definitions above cannot be found in the literature under these names. A "compactly coherent space" is usually called a "compactly generated space" or a "k-space" and the "compact coherentification" is typically referred to as the "k-ification". So why am I making up new terms here? This is a funny consequence of Strength 1. In the mathematical literature there are three non-equivalent types of spaces that get called "compactly generated". To my knowledge, this distinction was first discovered by Wikipedia contributors in [7]. The three notions all agree on Hausdorff spaces which I assume is the reason no one had cared previously. One of the three notions had already been implemented in Lean as `CompactlyGeneratedSpace`, so I had to choose a different name (which was suggested by Steven Clontz).

Now we can state two results about the product of CW complexes:

Theorem 5

Let X and Y be CW complexes with the respective families of characteristic maps $(Q_i^n \colon D^n \to X)_{n\in\mathbb{N},i\in I_n}$ and $(P_j^m \colon D^m \to Y)_{m\in\mathbb{N},j\in J_m}$.

(i) Assume that $X \times Y$ is compactly coherent. Then $X \times Y$ is a CW complex with characteristic maps $(Q_i^n \times P_j^m \colon D^n \times D^m \to C \times E)_{n,m\in\mathbb{N},i\in I_n,j\in J_m}$ and indexing sets $K_l = \bigcup_{n+m=l} I_n \times J_m$.

(ii) In general, the compact coherentification of $X \times Y$ is a CW complex.

I am not going to give the proof but you can find a proof (which I would call more of a proof sketch) in [3]. I hope that you can nonetheless imagine how careful you need to be to actually work with the different topologies and to describe which finite set of cells each new cell attaches to. In fact, in a talk I held in my Bachelor thesis seminar, I explained that I wanted to formalise the proof of this theorem and went through the most interesting part of it. When I formalised it later, I realised that I had gotten confused with the topologies and that my argument was incorrect. Luckily, none of the attendees of that talk had noticed it either.

I believe that most students who read through delicate proofs like this one don't actually understand all these intricate details. I certainly did not. But of course this understanding doesn't come from knowing that a result has been formalised, it comes from doing this formalisation yourself or from stepping through the formal proof line by line. This is where we find another limit:

Limit 3

Formalisation cannot guarantee that proofs are "beautiful", reusable or even readable at all.

This is a big issue with AI-generated Lean code. Often, the person that generated it doesn't have the knowledge to or doesn't want to clean up the proofs to make them understandable to humans. This basically limits the use of such a formalisation to verifying whether a result is correct or not.

I want to end this section with the obvious question: have any big formalisation projects ever found the result they were formalising to be false? To my knowledge, the answer is no. Usually, some errors are discovered along the way but they seem to have always been fixable. This was the case for example in the Liquid Tensor Experiment mentioned above. However, I think this is mostly because the results chosen for formalisation so far have either been standard results from university curricula or very famous theorems. In both of these cases, the results have been under high academic scrutiny and were therefore always likely to be correct. But I think it is only a question of time until a lesser known paper gets proven false through formalisation.

Conclusion

I hope this essay could give you a little bit of an overview of why people choose to formalise mathematics and of how the random person on the internet claiming that they have a formalised proof of an important unsolved problem was probably misled by their AI.
I should also leave you with probably the most important strength of formalisation:

> **Strength 4**
>
> Formalisation is fun!

In a Quanta article [2], Amelia Livingston is quoted saying

> You can do 14 hours a day in it and not get tired and feel kind of high the whole day. You're constantly getting positive reinforcement.

If this makes you want to try out Lean, I suggest you start by playing the Natural Numbers Game by Kevin Buzzard which can be found at `https://adam.math.hhu.de/`. If you want to read about how AI is currently affecting mathematics and formalisation I recommend you read the essay "Mathematicians in the Age of AI" by Jeremy Avigad [1]. If you want to read about how AI performances at the IMO can be interpreted, you can read the blog post "AI at IMO 2025: a round-up" by Kevin Buzzard. Lastly, if you want to hear more reasons for formalising maths, I want to repeat my recommendation from the introduction which is "Why formalize mathematics?" by Patrick Massot [4].

Hannah Scholz is a second year Master's student in Bonn. She wrote her Bachelor's thesis about formalising CW complexes in the proof assistant Lean which she has loved ever since. You can find out more about her on her webpage at `https://scholzhannah.de`.

References

[1] Jeremy Avigad. Mathematicians in the age of AI, 2026.

[2] Kevin Hartnett. Building the mathematical library of the future. *Quanta Magazine*. Accessed: 2026-03-25.

[3] Allen Hatcher. *Algebraic topology*. Cambridge Univ. Press, 13. print. edition, 2010.

[4] Patrick Massot. Why formalize mathematics?, 2021.

[5] Peter Scholze. Liquid tensor experiment. Xena, December 2020. Accessed: 2026-03-24.

[6] Peter Scholze. Half a year of the liquid tensor experiment: Amazing developments. Xena, June 2021. Accessed: 2026-03-24.

[7] Wikipedia contributors. Compactly generated space — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Compactly_generated_space&oldid=1334561412, 2026. Accessed: 2026-03-25.

Was Zahnstocher uns über π verraten

Manuel Hinz und Gina Pohlenz

Experimentaufbau

Am Pi-Day, Anlass der 13. Bonner Mathenacht, wurde folgendes Experiment von einigen Schüler*innen und Student*innen durchgeführt:
Als Vorbereitung werden auf ein großes Stück Papier parallele Linien gezeichnet, deren Abstand doppelt so groß ist wie ein Zahnstocher lang. Danach werfen wechselnde Freiwillige in *möglichst zufälliger* Weise eine Menge an Zahnstochern auf das Papier. Dabei wird nach jedem Wurf gezählt, wie viele der nun auf dem Papier liegenden Zahnstocher eine der Linien kreuzen.
Zuletzt wird die Gesamtanzahl geworfener Zahnstocher durch die Anzahl Zahnstocher, die eine Linie gekreuzt haben, geteilt. Dieses Ergebnis nennen wir die *Zahnstocher-Annäherung*.

Durchführung im Rahmen der 13. Bonner Mathenacht

Bei unserer Ausführung wurden in 137 Würfen insgesamt 2788 Zahnstocher geworfen, von denen 867 eine Linie kreuzten. Unser Ergebnis ist also der Bruch $\frac{2788}{867} = \frac{164}{51} \approx 3,2157$. Damit kommt unsere Zahnstocher-Annäherung der Zahl Pi ($\pi \approx 3,1415$) mit einem Abstand von ungefähr $0,0740$ zwar nicht sehr nah, aber warum unser Bruch immerhin in der Nähe von π liegt, wollen wir im Folgenden erklären.

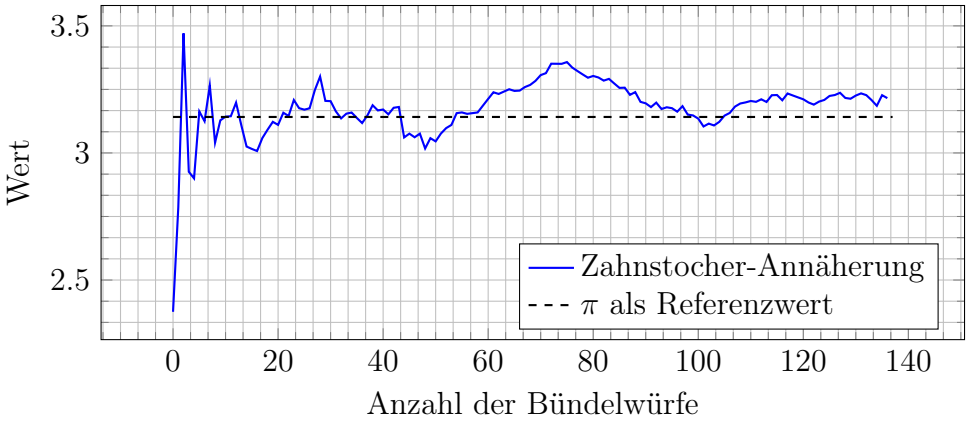


Abbildung 1: Die Zahnstocher-Annäherung über den Nachmittag hinweg

Wo ist π versteckt?

Nun stellt sich natürlich die Frage, was die Linien und Zahnstocher über π wissen, denn irgendwo muss π schließlich in der mathematischen Beschreibung vorkommen, um von den Zahnstocherwürfen angenähert werden zu können.
Wir wollen nun das ganze mit Hilfe von Abbildung 2 verstehen. Stell dir wie im Bild vor, dass das Papier kariert ist, und überlege, warum die Zahnstocher fast nie so landen, dass beide Enden auf einer der gemalten Linien liegen. Fast immer landet mindestens eines der Zahnstocherenden rechts von einer Linie.

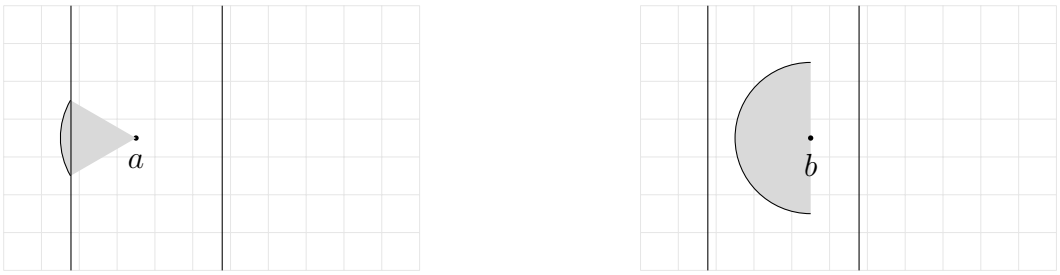


Abbildung 2: Kreise und Zahnstocher?

Dann können wir Folgendes überlegen: Für jedes der kleinen Quadrate auf dem Papier zeichnen wir einen Halbkreis um die Mitte des Quadrates, der den Radius einer Zahnstocherlänge hat, und schauen uns an, welcher Anteil dieses Halbkreises (gedacht wie in einem Kuchendiagramm) die Linie links vom Mittelpunkt trifft. Das stellen wir uns so vor, dass aus a lauter Strecken schießen, die so lang wie ein Zahnstocher sind, von denen einige die Linie kreuzen und andere sie verfehlen. Wir sehen in Abbildung 2 am Punkt a, dass ein gewisser Anteil der Strahlen, hier ein Drittel des Halbkreises, also 60°, in der Tat die Linie durchkreuzt. Anders sieht das bei Punkt b aus! Hier trifft der ganze Halbkreis die Linie nicht. Diesen Anteil nennen wir den *Linie-wird-gekreuzt-Anteil* dieser Ecke.
Wenn wir nun eine ganz feine Kästchenunterteilung des Papiers wählen, dürfen wir annehmen, dass das weiter rechts liegende Ende eines Zahnstochers auf einer Kästchenecke landet. Und wenn wir zudem ohne konkretes Ziel werfen, können wir annehmen, dass es für alle Kästchenecken gleich wahrscheinlich ist, dass unser rechtes Zahnstocherende dort landet. Dann können wir die Wahrscheinlichkeit dafür, dass ein einzelner Zahnstocher eine Linie kreuzt, so berechnen, indem wir die Linie-wird-gekreuzt-Anteile aller Kästchenecken addieren und diese Summe durch die Anzahl Kästchenecken teilen.
Der Winkel, mit dem der Zahnstocher auf dem Papier liegt, wenn wir wissen, in welcher Kästchenecke das weiter rechts liegende Ende gelandet ist, ist nach unserem ziellosen Wurf zufällig, und so können wir annehmen, dass jeder Landewinkel gleich wahrscheinlich ist. Damit können

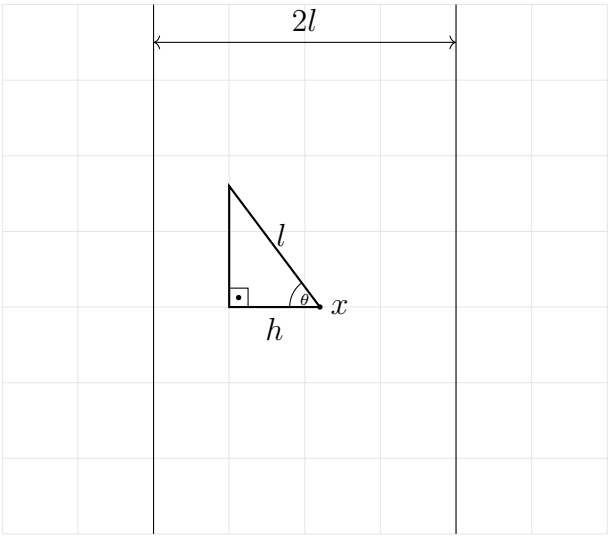


Abbildung 3: Trigonometrie

wir uns zusätzlich überlegen: Landet das weiter rechts liegende Ende des Zahnstochers in einer Kästchenecke näher an der Linie, haben wir eine größere Chance, dass unser Zahnstocher die Linie kreuzt. Oder anders gesagt: Ein größerer Teil des Halbkreises trifft die Linie!

Um von hier zu π zu kommen, können wir nun die Geometrie des Kreises nutzen, indem wir jeden Linie-wird-gekreuzt-Anteil in Abhängigkeit von der Kästchenecke darstellen. Betrachten wir dafür Abbildung 3. Sagen wir, das weiter rechts liegende Zahnstocherende ist bei Koordinate (x, y) gelandet. Der Halbkreis, den wir uns darum vorstellen, sieht unabhängig vom Koordinatenwert y gleich aus. Vom Winkel θ ist nun abhängig, ob der Zahnstocher, dessen Länge im Bild mit l bezeichnet ist, die linke Linie kreuzt. Wenn wir uns denken, dass die linke Linie entlang x-Wert 0 verläuft, dann kreuzt der Zahnstocher die Linie, wenn $h > x$. Doch wie können wir h in Abhängigkeit von θ beschreiben? Dafür brauchen wir die Kosinusfunktion. In einem rechtwinkligen Dreieck wie dem abgebildeten ist der Kosinuswert von θ gegeben durch $\cos(\theta) = \frac{h}{l}$, also dem Verhältnis von anliegender kurzer Seite durch die längste Seite des Dreiecks. Stellen wir die Gleichung um, erhalten wir $\cos(\theta) \cdot l = h > x$ als Bedingung. Wenn wir weiterhin elementar vorgehen wollen, müssten wir nun für eine feine Kästchenunterteilung für jede Kästchenecke entlang einer horizontalen Strecke ausrechnen, welche Winkel θ diese Bedingung erfüllen. Diese Linie-wird-gekreuzt-Anteile merken wir uns dann, addieren sie zusammen und teilen sie durch die Anzahl der betrachteten Kästchenecken. Dies ist allerdings ziemlich anstrengend. Glücklicherweise ist die Kosinusfunktion gut untersucht und es ergibt sich, dass in diesem Prozess für feinere und feinere Kästchenunterteilungen π angenähert wird. Wenn du dich mit Integralen auskennst, kannst du weiter unten nachlesen, wie genau π auftaucht.

Wenn du die Funktionen, die in den Berechnungen auftreten, nicht kennst, kannst du stattdessen auf deinem Papier für jede Kästchenecke den Anteil des Kreises messen, welcher die Linie trifft. Diese Anteile könntest du direkt für eine grobe Annäherung benutzen. Bei einer großen Anzahl an Kästchen, die für eine gute Annäherung wichtig sind, wird dies allerdings schnell mühselig. Auch hier fällt dir bestimmt auf, dass dafür nur der horizontale Abstand zwischen Kästchenecke und Linie wichtig ist.
Was bringt uns nun der Zufall? wir haben im Experiment eben nicht die Winkel gemessen und addiert, sondern nur Zahnstocher geworfen. Nun ist die grundlegende Idee, dass wenn man Experimente mit zufälligem Ausgang oft genug wiederholt, sich die Zahnstocher-Annäherung der Linienkreuzungswahrscheinlichkeit eines einzelnen Zahnstocherwurfs annähert! Ein klassisches Beispiel dafür ist der Münzwurf: Wirft man viele Male eine Münze und zählt, wie oft diese Kopf zeigt, so nähert sich das Verhältnis $\frac{K}{N}$ von Kopf-Würfen K an allen Würfen N der Wahrscheinlichkeit $\frac{1}{2}$ an, dass eine einzelne Münze Kopf zeigt. Hier müssen wir natürlich einiges annehmen, zum Beispiel, dass die Münze nicht gezinkt ist. Ähnliches müssen wir auch bei unserem Zahnstocherexperiment beachten!

Hinweise zum Wurfverhalten und Fehlerquellen

Ähnlich wie bei der Münze kann etwas schiefgehen, wenn man nicht richtig zufällig wirft. Im Falle der Münze könnte eine der Seiten zum Beispiel schwerer sein, doch was kann hier bei uns schief gehen?
Nimmt man einen Haufen Zahnstocher als Bündel in die Hand und wirft dieses nur nach vorne, so bleiben sicherlich mehr Zahnstocher in der ursprünglichen Ausrichtung liegen. Das würde unsere Rechnungen unnütz machen, da wir eben angenommen haben, dass jeder Landewinkel gleich wahrscheinlich ist. Ebenso könnte man überlegen, ob Größe oder die dominante Hand der werfenden Person einen Einfluss hat, aber hier kommt uns zu Hilfe, dass mehrere Freiwillige geworfen haben und sich solche Effekte bei genug Personen gegenseitig ausgleichen sollten.

Theoretischer Hintergrund

Wir wollen nun noch einmal dort ansetzen, wie durch eine feinere Kästchenunterteilung die Annäherung an π besser gelingt. Dafür müssen wir zuerst die Bedingung $\cos(\theta) \cdot l > x$ nach θ umstellen, wofür wir eine (monoton fallende) Umkehrfunktion des Kosinus brauchen. Diese Funktion $\arccos\colon [-1,1] \to [0,\pi]$ nimmt ein Verhältnis von anliegender kurzer Seite zu längster Dreiecksseite und gibt einen Winkel aus, dessen benachbarte Seiten dieses Verhältnis haben. Somit erhalten wir als Bedingung $0 < \theta < \arccos(\frac{x}{l})$. Der Linie-wird-gekreuzt-Anteil zu einem x ist also $\frac{\arccos(\frac{x}{l})}{\pi}$.

Nun ist aber zufällig, wo genau das weiter rechts liegende Zahnstocherende landet, weswegen wir diese Wahl von x mit einem Integral herausrechnen. Das Integral verwirklicht den Wunsch, einer immer feinere Kästchenunterteilung zu wählen. Da der Linie-wird-gekreuzt-Anteil genau dann größer 0 ist, wenn $0 < x < l$, müssen wir nur für solche x überhaupt die Anteile bestimmen (ist dir das auch schon aufgefallen?). Als Normalisierungsfaktor brauchen wir dann noch $\frac{1}{l}$, da wir unsere x-Werte aus dem $l$breiten Bereich rechts von der linken Linie beziehen, und erhalten somit

$$\int_0^l \frac{\arccos(\frac{x}{l})}{\pi} \cdot \frac{1}{l} \, \mathrm{d}x = \frac{1}{\pi}$$

als Wahrscheinlichkeit, dass ein geworfener Zahnstocher die Linie kreuzt.
Dass unsere Zahnstocherannäherung π und nicht $\frac{1}{\pi}$ nahe ist, liegt daran, dass wir nicht den Anteil linienkreuzender Zahnstocher an allen geworfenen Zahnstochern bestimmt haben, sondern umgekehrt die Anzahl geworfener Zahnstocher durch die Anzahl linienkreuzender Zahnstocher geteilt hatten. Dass unser Experiment überhaupt hilft, diese (umgekehrte) Einzelwahrscheinlichkeit anzunähern, lässt sich mit dem Gesetz der großen Zahl erklären: Es besagt, dass für Experimente dieser Art, bei denen bei der Wiederholung des immer gleichen Ablaufs keine Abhängigkeiten zwischen den Durchführungen bestehen, die Einzeltrefferwahrscheinlichkeit durch den Anteil der erfolgreichen Durchgänge an allen Durchgängen angenähert wird, wobei ein "Treffer" bei uns ein eine Linie kreuzender Zahnstocher ist. Kehrt man nun diesen Anteilsbruch um, nähert man sich dem multiplikativ inversen Wert der Einzeltrefferwahrscheinlichkeit an, in unserem Fall also π.

Manuel und Gina haben nach der Bonner Mathenacht Feuer gefangen, mehr Leute für Annäherungsverfahren zu begeistern.

Contributors

We are thankful to Stefan Hartmann and the Hausdorff Center for Mathematics for supporting this project by providing us with a relatively permament e-mail address and by funding a printed edition of this collection!

Authors

Manuel Hinz, Gina Pohlenz, Hannah Scholz

LaTeXand Design

The document layout is based on a modified version of Evan Chen's `evan.sty` package.
The title graphic was created by Manuel Hinz.

You?

If you are a math person from Bonn and would like to help with the next edition of MEB, you can write us an e-mail at infomeb@uni-bonn.de. If you want to be added to our WhatsApp community, include your phone number! To get started with writing your essay, clone our repository https://github.com/MEB-Collective/meb.

Licensing