Analiza matematyczna dla informatyków.

Mieczysław Cichoń, ver. 2.1/2021

Mieczysław Cichoń - WMI UAM

Plan wykładu.

- Program wykładu.
- Motywacje, czyli po co matematyka (analiza matematyczna) informatykom: kilka uwag o motywacji.
- Literatura.
- Ponieważ poza notatkami z wykładu potrzebujemy dostępne dla wszystkich **ujednolicone** źródła, to decydujemy się na dwa podstawowe plus każdorazowo (w miarę potrzeby) materiały dodatkowe. Są to:
  [K] M. Mrozek, "Analiza matematyczna I. Notatki do wykładu matematyki komputerowej", UJ, Kraków, 2013. UJ Kraków
  [W] P. Strzelecki, "Analiza matematyczna I", UW, Warszawa, 2012. UW Warszawa

Szkice wykładów.

- Każdy wykład rozpoczynamy od podania stron stanowiących jego podstawę - najlepiej byłoby przeczytać podane fragmenty **przed** danym wykładem, aby lepiej śledzić moje komentarze do materiału...
- W czasie wskazanym będę odpowiadał np. na czacie MS Teams na pytania dotyczące wykładów (najlepiej: bieżących!).

Literatura.

- M.Mrozek, "Analiza matematyczna I. Notatki do wykładu matematyki komputerowej", UJ, Kraków, 2013.
- P.Strzelecki, "Analiza matematyczna I", UW, Warszawa, 2012.
- M. Moszyński, "Analiza matematyczna dla informatyków", UW, Warszawa, 2010.
- M.Oberguggenberger, A.Ostermann, "Analysis for Computer Scientists", Springer, London, 2011.
- A. Sołtysiak, "Analiza matematyczna", UAM, 2009.
- A. Ralston, "Wstęp do analizy numerycznej", PWN, Warszawa, 1983.
- D.B. Small, J.M. Hosnack, "Ćwiczenia z analizy matematycznej z zastosowaniem systemów obliczeń symbolicznych", WNT, Warszawa, 1995.
- wykłady prof. P. Domańskiego (†) do wykładu DANI1
- motywacje własne – opublikowane na stronie
- opracowania własne (MC) – publikowane na stronie WWW na bieżąco według realizacji materiału

Wykład 1.

(3h) Cele nauczania analizy dla informatyków.

Szkic teorii aksjomatycznej liczb rzeczywistych, w tym kresy, zapis dziesiętny liczb rzeczywistych. Liczby wymierne. Potęga o wykładniku rzeczywistym. Istnienie pierwiastka.

Uwagi o arytmetyce komputerowej.

Czy to na pewno wykład konieczny dla informatyków?

"Wybrałem studia z informatyki, czyli po co mi matematyka i analiza matematyczna?!"

Przemyślenie (DLA) studentów ...

Studiujecie informatykę tylko dla "papierka", ale wiecie już najlepiej co będziecie robić i "wiecie", że matematyka nie będzie Wam potrzebna. Serio?? Będziecie pracowali ze 40 lat - a teraz pomyślcie jak wyglądała informatyka 40 lat temu i co robią ówcześnie kształceni informatycy! Kto się nie dostosował - ten nie pracuje! Taki kierunek Państwo wybrali... Aby zrozumieć co i dlaczego zmienia się w informatyce oraz mieć zdolność dostosowania - trzeba zrozumieć matematyczne podstawy informatyki...

No dobrze, **nie jest** argumentem, że program studiów układają (również) informatycy, a więc wiedzą co robią... (choć to prawda). Czy poniższe uwagi są dla wszystkich *"informatyków"*? Tak, chyba, że ktoś ma specyficznie wąską wizję pracy (obsługa jednego programu np. bazy danych czy sieci komputerowej), ale nawet wtedy wypadałoby *rozumieć* co się robi - zwłaszcza gdy coś **nie działa**...

- Taki program studiów informatyki to nie jest to nasz wymysł. Ciekawe programy matematyki (i analizy matematycznej) mają uczelnie z Cambridge czy Princeton. I mówimy tu tylko o wstępnych informacjach ("Mathematics for Computer Science"), bo dalszy program (i ich, i nasz...) zawiera kolejne przedmioty poszerzające wiedzę matematyczną (to będzie pokazane na wykładzie). Nie wspomnę o uczelniach, które tego nie uczą bo... to zbyt łatwe i studenci taki "wstęp" sami sobie opracują, a omawia się zaawansowane działy matematyki (np. Politechnika w Zurychu i teoria falek). Jako ćwiczenie - proszę poszukać na wspomnianych uczelniach programów, o których napisałem...
- Polecam też książkę z której sam korzystam: M. Oberguggenberger, A. Ostermann, *Analysis for Computer Scientists: Foundations, Methods, and Algorithms*, Springer Science and Business Media, 2011. (biblioteka elektroniczna)
- To mało konkretne, a kto miałby to czytać? :-) Dla matematyków - zbyt nieściśle podane, dla informatyków - niezauważalne (oczywiste?).

- Ale jak mam podać (proste) przykłady, skoro Czytelnik dopiero *studiuje* informatykę? Wybiorę więc system komentarzy (informatycy - praktycy zgodzą się lub nie - zależy co robią...). Zacznę od liczb: w szkole średniej były działania na liczbach naturalnych, całkowitych czy wymiernych. A liczby rzeczywiste były ... mało objaśniane. Nie bez powodu. Na wykładzie podamy sobie aksjomatykę liczb rzeczywistych (z konsekwencjami) - czyli to, czego nie było w szkole średniej (za trudne?).
- Niestety - dla komputera to JEST za trudne! Tak naprawdę obliczenia są prowadzone na zbiorze liczb całkowitych (i to nie wszystkich :-)). Arytmetykę komputerową omówi kto inny, ale wypadałoby "pomóc" komputerowi (znając liczby rzeczywiste - ograniczać błędy, przyspieszać obliczenia itp., np. kolejność wykonywania operacji).
- Ależ nie potrzebujemy liczb rzeczywistych (no, skoro komputer ich nie "zna", to nie potrzeba). To po co funkcje, ich własności itp.?

Przykład - pytanie.

$$x^2 - 2px + q = 0 \tag{1}$$

W szkole stosowaliśmy algorytm zakładający obliczanie pierwiastków wzorami (proszę mnie sprawdzić...)

$$x_1 = p + \sqrt{p \cdot p - q} \tag{2a}$$

$$x_2 = p - \sqrt{p \cdot p - q} \tag{2b}$$

Poprawny algorytm? Wrócimy do tematu jeszcze na tym wykładzie. Źródło problemu? Też wyjaśnimy - ale tu będzie potrzebna matematyka...

- Popatrzmy: biorąc ciągi rzeczywiste dodatnie a_n i b_n (np. iteracje pewnych obliczanych wielkości) mamy wybrać "lepszy" z nich i oczekujemy $a_n = O(b_n)$ (pojęcie powszechne w ocenach algorytmów). Jak to **najłatwiej** sprawdzić? Obliczyć pewne granice górne (a cóż to jest?)... Nie jest łatwiej? To np. (w uproszczeniu - przepraszam matematyków) skorzystać z funkcji f i g takich, że $f(n) = a_n$ i $g(n) = b_n$ oraz (o ile można) z reguły de l'Hôspitala (por. wykład z analizy...).
- Odnośnie granic ciągów: zastosowania w informatyce są oczywiste (?), definicję podamy na wykładzie, a ja proponuję wymyśleć algorytm obliczeniowy (z zadaną dokładnością ε). Pytanie: jak zakończyć obliczenia? Oby nie poprzez badanie różnicy pomiędzy dwoma kolejnymi krokami (tj. $|a_{k+1} - a_k| < \varepsilon$)! Proszę spróbować dla $a_n = \log n$, $\varepsilon = 2^{-10}$ i odpowiednio dużych k...

- Wróćmy na chwilę do ogółu: ścisłość rozumowania (dowody) - przecież sprawdzenie poprawności każdego **algorytmu** to **dowód**, czyli należy poznać zarówno metody dowodowe (np. indukcja), jak i weryfikację założeń. Typowy (niestety) błąd w opracowaniach "dla informatyków" to brak weryfikacji, czy badany obiekt **istnieje** i **jest jednoznacznie** określony. A to z kolei wymaga często kolejnej wiedzy matematycznej.
- Prosty przykład: wiele osób utożsamia algorytm ze wzorem! A gdzie założenia gwarantujące poprawność? Gdy szukamy rozwiązań równania nieliniowego $f(x) = 0$, $x \in [a, b]$, to wzór na iteracje jest prosty: wybieramy punkty "początkowe" iteracji x_0 i x_1 (no dobrze, tu też jest reguła jak to zrobić np. "falsi") i kładziemy $x_{n+1} = x_n - \frac{f(x_i-1)(x_i-1-x_i-2)}{f(x_i-1)-f(x_i-2)}$. I już? Działa?
  No - nie zawsze (przykłady na wykładzie lub ćwiczeniach). A założenia **gwarantujące** zbieżność metody? Funkcja musi mieć pochodne rzędu I i II ciągłe (czyli $f \in C^{(2)}(a, b)$) i mające stały znak w (a, b), no i ma mieć *jedno* rozwiązanie w tym przedziale. Ale... to jednak pojęcia z analizy matematycznej... (a inne algorytmy korzystają z teorii interpolacji).

- Inna sprawa: czy wiesz skąd wynikają zmiany formatu zapisu plików (przynajmniej większość :-))? Przykład dla grafiki. Jeśli "nie potrzeba" matematyki, funkcji i analizy matematycznej, to pozostaniemy przy grafice bitmapowej: punkt po punkcie podajmy jego atrybuty (np. położenie, składowe barwy czy jasności). Ale jeśli to dla kogoś niewygodne, to pozostaje grafika rastrowa (lub nawet wektorowa choćby PDF, CDR czy SVG) np. JPEG. I nie wchodząc w szczegóły: potrzebna jest (dyskretna) transformata Fouriera. A jej wprowadzenie "matematyczne" to ciągi i szeregi funkcyjne, szeregi Fouriera i ciągła transformata Fouriera... Za skomplikowane? To dlaczego **i jak** powstał kolejny format JPEG 2000 (podpowiem: dyskretna transformata falkowa, bardzo "porządna" matematyka!)?
- Jeśli chcemy coś jeszcze usprawnić, to najpierw wypada zrozumieć jak działa aktualne rozwiązanie, wdrożyć nowe **matematyczne** idee i sprawdzić (pod kątem zastosowań w informatyce(!)). Może warto rozważyć naukę matematyki?

- Klasyczną procedurą w informatyce jest stosowanie ciągów zdefiniowanych rekurencyjnie. Co prawda, to będzie omawiane poza podstawową analizą matematyczną (Analiza I), ale bez jej podstaw nie da się tego nauczyć, Do konstrukcji funkcji tworzących (podstawowa metoda dla rekurencji) potrzeba będzie zrozumienie m.in. pojęć: funkcja i jej własności, szereg potęgowy i jego promień zbieżności (do tego sporo kombinatoryki i algebry). W uzupełnieniu: skoro komputer prowadzi obliczenia na liczbach całkowitych, to proszę przemyśleć jak korzystamy przy jego pomocy z funkcji (np. w arkuszu kalkulacyjnym)? Jak obliczać wartości takich funkcji jak $f(x) = \sin x$, bo przecież od czasu do czasu z tego korzystamy, nieprawdaż?

- Jeśli gdzieś napotkacie hasło "ten wielomian (lub inna funkcja) przybliża daną...", to podstawowe pytanie brzmi: co to znaczy "przybliża"? Musimy sprecyzować co jest "odległość pomiędzy funkcjami" (metryka), no i podać jak to można obliczyć (oszacować) oraz *dlaczego* jest więcej niż jedna taka metoda ("np. "zbieżność średniokwadratowa").
- Metody numeryczne, to z punktu widzenia informatyki "matematyka stosowana" czy "metody matematyczne informatyki". Wszystkie metody całkowania numerycznego czy numerycznego rozwiązywania równań różniczkowych bazują na definicjach czy własnościach opartych na analizie matematycznej. Podobnie interpolacja czy aproksymacja numeryczna. Nie będę wchodził w szczegóły - pozostawię to na osobny przedmiot "Metody numeryczne" (proszę zabrać na niego notatki z analizy!).

- Kolejny przykład: często słyszycie "błąd metody numerycznej", "ta metoda jest lepsza (dokładniejsza), (szybciej zbieżna)". Co się za tym kryje? Ogólnie: bardzo często będzie potrzebne rozwinięcie funkcji zgodnie ze wzorem Taylora (analiza!) i oszacowanie reszty (niekiedy w postaci całkowej...). Choć (niestety) studenci kojarzą takie zagadnienie jako "był taki wzór...".
- Coś spoza wykładu z "Analizy I" (ale na jej podbudowie): analiza harmoniczna, szeregi i transformaty Fouriera itp. mają szerokie zastosowania w informatyce (np. algorytm Cooleya i Tuckeya FFT z zastosowaniami czy algorytm uczenia Kushilevitza i Mansoura - ponownie o szczegóły proszę pytać na właściwych przedmiotach informatycznych). Nie mówiąc już o teorii sygnałów... Por. też Daniel Stefanković, *Fourier transforms in computer science*, University of Chicago, Department of Computer Science, TR-2002-03.

- Wielu z Państwa napotka przedmioty z uczenia/nauczania maszynowego. Podstawą tamtych metod jest znajdowanie warości ekstremalnych funkcji opisujących wagi w systemie. A czynnikiem oceny "optymalności" jest np. funkcja błędu. Czysta "analiza matematyczna stosowana" ;-) (plus algorytmy jak to zrealizować na komputerze). Nawet *wybór* algorytmu z gotowej biblioteki (co nie zawsze ma sens) wymaga zrozumienia podstaw działania wielu algorytmów bazujących na metodach analizy matematycznej (i algebry liniowej).
- Inna dziedzina: rozpoznawanie obrazów i grafika komputerowa. Tam będzie ważny splot (oczywiście: w wersji dyskretnej), a przydatne też inne operacje na zbiorach czy metryki.
- itd.

Do obejrzenia...

Proponuję obejrzeć ten filmik (i kolejne z tego cyklu):

"Do czego ta matma?" (np. wprowadzenie w temat)

Strony do lektury na wykład 1...

Czytamy najpierw:

[K] : strony 13-16 (do tego źródła będę kierował po motywacje lub zastosowania informatyczne : nieformalne intuicje informatyczne w tym skrypcie są w kolorze brązowym, aspekty obliczeniowe analizy matematycznej, ilustrowane programami komputerowymi w języku C++ oraz instrukcjami programu Mathematica złożone są czcionką w kolorze niebieskim)
i dalej

[K] : strony 119-129

[W] : strony 1-16
(lub alternatywnie: z innego wykładu strony 11-22.

Zwracamy uwagę na: *istnienie pierwiastków, sprawdzenia jakie aksjomaty nie są spełnione w arytmetyce komuterowej, zasada Archimedesa i jej konsekwencje (np. funkcja entier), gęstość zbioru liczb wymiernych w $\mathbb{R}$ i jej konsekwencje, kresy zbiorów.*

[K] - kolory...

1.2.2 Ciągi przybliżeń

W praktyce wygodnie ograniczyć się do mianowników postaci $m = 10^n$. Niestety ręczne poszukiwanie stosownego licznika jest bardzo żmudne, a dla dużych n praktycznie niewykonalne. Potrzebujemy algorytmu, który dla zadanego mianownika 10^n pozwoli nam wyznaczyć stosowny licznik i_n oraz komputera, który sprawnie wykona obliczenia w tym algorytmie. Łatwo zauważyć, że

$$i_n = \max\{\, i \mid i * i < 2 * 100^n \,\}.$$

Sugeruje to algorytm, który przy zadanym n sprawdza nierówność $i * i < 2 * 100^n$ dla kolejnych liczb naturalnych i i zwraca i_n jako ostatnie i, dla którego nierówność ta zachodzi. Wtedy $x_n := \frac{i_n}{10^n}$ jest przybliżeniem $\sqrt{2}$, które, jak łatwo sprawdzić, spełnia

$$|x_n - \sqrt{2}| \leqslant \frac{1}{10^n}.$$

Kolejne liczby x_n tworzą tzw. ciąg liczbowy, w tym przypadku tzw. ciąg przybliżeń. Ciąg przybliżeń, to ciąg skonstruowany w celu przybliżania pewnej liczby niewymiernej liczbami wymiernymi. Ciągi przybliżeń to nie jedyne ciągi rozważane w matematyce, choć ich rola praktyczna jest bardzo istotna. Badanie ciągów to jeden z głównych obiektów zainteresowań analizy matematycznej.

Wprowadźmy w programie Mathematica rozdzielony średnikami ciąg instrukcji

```
n = 3; i = 1; While[ i^2 < 2*100^n, i = i + 1]; {(i-1)/10^n, i/10^n}
```

Pojawiają się tu zmienne `i` *oraz* `n`. *Instrukcja* `While` *to instrukcja pętli. Jak długo spełniony jest warunek* `i^2 < 2*100^n` *wykonywana jest instrukcja* `i = i + 1`. *Ponieważ przed wykonaniem instrukcji* `While` *ustawiliśmy zmienną* `n` *na 3, a* `i` *na 1, pętlę wykonujemy dla kolejnych wartości* $\mathtt{i} = 1, 2, \ldots$ *aż zajdzie nierówność* $\mathtt{i\hat{}2} \geqslant 2000000$. *Mathematica wyprowadza jako wynik ostatnie obliczone wyrażenie, którym jest para ułamków* `{(i-1)/10^n,i/10^n}`. *W rozważanym przypadku otrzymamy*

$$\{\frac{707}{500}, \frac{283}{200}\}.$$

jako oszacowanie od dołu i od góry $\sqrt{2}$ z dokładnością do $\frac{1}{1000}$.

Podstawiając w powyższym listingu za `n` *kolejne liczby naturalne i wykonując obliczenia otrzymamy przybliżenia $\sqrt{2}$ od dołu i od góry o coraz lepszej*

8.5 Twierdzenie o istnieniu pierwiastka.

Naszym celem w tym rozdziale jest udowodnienie, że w zbiorze liczb rzeczywistych dodatnich pierwiaski istnieją. W szczególności ostatecznie zamkniemy sprawę sensowności $\sqrt{2}$. Nim jednak twierdzenie to udowodnimy potrzebujemy dwa pomocnicze twierdzenia: własność Archimedesa oraz gęstość $\mathbb{Q}$ w $\mathbb{R}$.

8.5.1 Własność Archimedesa

Twierdzenie 8.5.1 *Ciało liczb rzeczywistych ma własność Archimedesa, tzn.*

$$\forall a, b \in \mathbb{R}^+ \; \exists n \in \mathbb{N} \quad na > b.$$

Dowód: (*) Dowód poprowadzimy nie wprost. Załóżmy zatem, że istnieją liczby rzeczywiste dodatnie a, b takie, że dla każdej liczby naturalnej n zachodzi $na \leqslant b$. Oznacza to, że b jest majorantą zbioru

$$A := \{\, x \in \mathbb{R} \mid \exists n \in \mathbb{N}\; x = na \,\}.$$

Niech $c := \sup A$ będzie jego kresem gónym. Ponieważ $c - a < c$, więc $c - a$ nie jest majorantą zbioru A. Zatem znajdziemy $n_0 \in \mathbb{N}$ takie, że $n_0 a > c - a$. Ale wtedy $(n_0 + 1)a > c$, a w konsekwencji c nie jest majorantą A, a tym bardziej kresem górnym zbioru A. Otrzymana sprzeczność dowodzi twierdzenia. □

Wniosek 8.5.2 *Ciało liczb wymiernych ma wartość Archimedesa.*

Dowód: ćwiczenie.

8.5.2 Gęstość zbioru w zbiorze

Definicja 8.5.3 Niech X będzie przestrzenią liniowo uporządkowaną, a A niech będzie podzbiorem X. Mówimy, że A jest *gęsty* w X jeżeli

$$\forall x, y \in X \quad x < y \;\Rightarrow\; \exists z \in A \; x < z < y.$$

Gdy $A = X$, mówimy, że A jest *gęsty w sobie*.

Twierdzenie 8.5.4 *Zbiór liczb wymiernych jest gęsty w zbiorze liczb rzeczywistych.*

8.5.3 Pierwiastki

Teraz możemy już udowodnić twierdzenie o istnieniu n-tego pierwiastka

Twierdzenie 8.5.5 *(o istnieniu n-tego pierwiastka)*

$$\forall n \in \mathbb{N}_1 \forall x \in \mathbb{R}^* \exists! y \in \mathbb{R}^* : \quad y^n = x$$

Dowód: (*) Pokażemy najpierw, że pierwiastek istnieje co najwyżej jeden. Dla dowodu nie wprost, przyjmijmy, że istnieją $y_1, y_2 \in \mathbb{R}^*$ takie, że $y_1 \neq y_2$ oraz $y_1^n = y_2^n$. Dla ustalenia uwagi możemy przyjąć, że $y_1 < y_2$ (dowód gdy $y_1 > y_2$ jest analogiczny). Na mocy twierdzenia 8.3.2 mamy $y_1^n < y_2^n$ co przeczy $y_1^n = y_2^n$ i dowodzi jednoznaczności.

Dla dowodu istnienia zauważmy najpierw, że $0^n = 0$, zatem teza jest oczywista gdy $x = 0$. Możemy więc założyć, że $x > 0$. Rozważmy zbiór

$$E := \{ u \in \mathbb{R}^* \mid u^n < x \}.$$

Oczekujemy, że kres górny tego zbioru jest poszukiwanym pierwiastkiem. Musimy jednak najpierw pokazać, że E jest zbiorem niepustym i ograniczonym od góry.

Pokażemy, że $E \neq \varnothing$. Niech $t := \frac{x}{x+1}$. Mamy $0 < t < 1$ oraz $t < x$. Mnożąc nierówności $0 < t < 1$ obustronnie przez $t > 0$ dostajemy z własności ciała uporządkowanego dostajemy $0 < t^2 < t$. Ale $t < 1$, zatem $0 < t^2 < 1$. Rozumując indukcyjnie stwierdzamy, że również $0 < t^{n-1} < 1$. Mnożąc tę nierówność obustronnie przez $t >$) dostajemy $t^n < t$. Zatem $t^n < x$ co oznacza, że $t \in E$. Tak więc $E \neq \varnothing$. Pokażemy z kolei, że $1 + x$ jest majorantą zbioru E. Gdyby tak nie było, znaleźlibyśmy $s \in E$, takie, że $s > 1 + x$. Wtedy $s^n > s > x$. Zatem $s \notin E$. Otrzymana sprzeczność pokazuje, że $1 + x$ rzeczywiście jest majorantą zbioru E, zatem E jest ograniczony od góry. Zatem ma kres górny.

Połóżmy $y := \sup E$. Pokażemy, że rzeczywiście $y^n = x$. Dla dowodu nie wprost załóżmy, że $y^n \neq x$. Mamy do rozważenia dwa przypadki: $y^n < x$ lub $y^n > x$. Rozważmy najpierw przypadek $y^n < x$.

Liczymy teraz, że da się skonstruować $y_1 > y$ takie, że $y_1^n < y$, co przeczyłoby $y = \sup E$. Naturalne jest poszukiwać y_1 w postaci $y_1 = y + h$. Pytanie jak dobrać h? Mamy mieć $(y + h)^n < x$ co jest równoważne

$$(y + h)^n - y^n < x - y^n. \tag{8.10}$$

Wiemy ze wzoru (7.6), że

$$(y + h)^n - y^n = h\left((y + h)^{n-1} + (y + h)^{n-2}y + \cdots y^{n-1}\right) < hn(y + h)^n.$$

Zatem, dla $h < 1$ mamy

$$(y + h)^n - y^n < hn(y + 1)^n.$$

Stąd, by zagwarantować (8.10), wystarczy zapewnić, że

$$hn(y+1)^n < x - y^n,$$

a w tym celu wystarczy wziąć $h < \frac{x-y^n}{n(y+1)^n}$. Jesteśmy teraz gotowi, by ze zrozumieniem wrócić do formalnego dowodu.

Dobierzmy $h \in \mathbb{Q}$ takie, że

$$0 < h < \min\{1, \frac{x-y^n}{n(y+1)^n}\}.$$

Istnienie takiego h zapewnia nam gęstość $\mathbb{Q}$ w $\mathbb{R}$ (twierdzenie 8.5.4). Łatwo sprawdzamy, że wtedy

$$(y+h)^n - y^n < hn(y+1)^n < x - y^n.$$

Zatem $(y+h) < x$, czyli $y+h \in E$. Ale $y+h > y = supE$, skąd $y+h \notin E$. Otrzymana sprzeczność wyklucza nierówność $y^n < x$.

Pozostaje pokazać, że nierówność $y^n > x$ też prowadzi do sprzeczności. *Tym razem liczymy, że uda się skonstruować $y_2 < y$, również będące majorantą zbioru E, co doprowadzi do sprzeczności z faktem, że y jako kres górny jest najmniejszą majorantą. Poszukujemy y_2 w postaci $y-k$. Chcemy mieć $(y-k)^n > x$ co jest równoważne postulatowi*

$$y^n - (y-k)^n < y^n - x$$

Rozpisując podobnie jak poprzednio mamy

$$y^n - (y-k)^n = k\left(y^{n-1} + y^{n-2}(y-k) + \cdots + (y-k)^{n-1}\right) < kny^{n-1}.$$

Podpowiada to jak dobrać k.

Dobierzmy $k \in \mathbb{Q}$ takie, że

$$0 < k < \frac{y^n - x}{ny^{n-1}}.$$

Łatwo sprawdzamy, że przy tak dobranym k mamy $(y-k)^n > x$. Twierdzimy, że $y-k$ jest majorantą zbioru E. Gdyby tak nie było, znaleźlibyśmy $v \in E$ takie, że $v > y-k$. Ale wtedy $v^n > (v-k)^n > x$, co przeczy $v \in E$. Zatem $y-k$ rzeczywiście jest majorantą zbioru E. Ale $y-k < y$, skąd y nie jest najmniejszą majorantą $E<$ zatem $y \neq \sup E$. Otrzymana sprzeczność dowodzi, że $y^n = x$. □

Nieujemną liczbę, której n-ta potęga jest równa liczbie $x \geqslant 0$ nazywamy n-tym *pierwiastkiem* z x i oznaczamy $\sqrt[n]{x}$.

Funkcja “entier”.

Jedną z podstawowych konsekwencji aksjomatyki liczb rzeczywistych jest istnienie funkcji “entier”. Jest ona używana (i użyteczna) w arytmetyce komuterowej (i to mimo braku liczb rzeczywistych w takich obliczeniach).

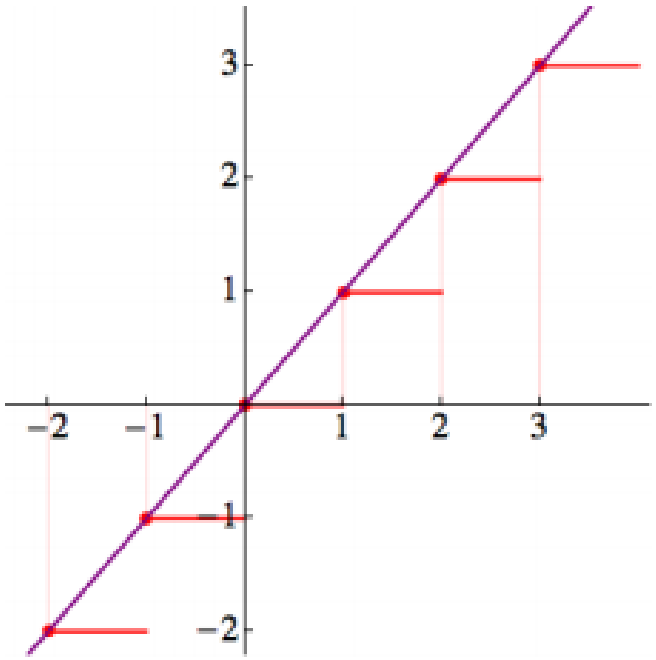
Definicja 1.15 (Entier, czyli część całkowita liczby rzeczywistej). Dla każdej liczby $x \in \mathbb{R}$ określamy

$$[x] = \sup\{k \in \mathbb{Z} : \ k \leq x\}\,. \tag{1.4}$$

Zatem, na przykład, $[7] = 7 = -[-7]$, $[\frac{1}{3}] = 0$, $[\sqrt{2}] = 1$, ale $[-\sqrt{2}] = -2$, gdyż największą liczbą całkowitą nie przekraczającą $-1{,}4142\ldots = -\sqrt{2}$ jest właśnie -2.

Stwierdzenie 1.16. *Dla każdego $x \in \mathbb{R}$ liczba $[x]$ jest całkowita i spełnia nierówności*

$$[x] \leq x < [x] + 1\,. \tag{1.5}$$



Funkcja entier i funkcja $f(x) = x$. Lewy koniec każdego z poziomych odcinków wykresu $[x]$ należy do tego wykresu, a prawy – nie.

Oto niedługi dowód, dla zainteresowanych formalizacją. To, że $[x] \in \mathbb{Z}$, wynika z definicji cześci całkowitej i własności zbioru $\mathbb{Z}$, wyrażonej w Twierdzeniu 1.14: zbiór

$$K = \{k \in \mathbb{Z} : k \leq x\}$$

jest ograniczony z góry przez x, a liczba $[x]$ jest największym elementem K. Mamy $[x] \leq x$ wprost z definicji supremum. Gdyby $[x] + 1 \leq x$, to byłoby

$$\sup K = [x] < [x] + 1 \in K,$$

a to jest sprzeczność z definicją supremum. □

Wiemy już zatem (z grubsza), co to są liczby rzeczywiste, naturalne, całkowite i wymierne. Wiemy też, że istnieją liczby niewymierne; jedną z nich jest $\sqrt{2}$. Sformułujmy jeszcze jedną ważną własność, którą ma zarówno zbiór $\mathbb{Q}$ liczb wymiernych, jak i zbiór $\mathbb{R} \setminus \mathbb{Q}$ liczb niewymiernych.

Definicja 1.17. Zbiór $A \subset \mathbb{R}$ nazywa się gęsty, jeśli dla wszystkich $x, y \in \mathbb{R}$, $x < y$, istnieje element $a \in A$ taki, że $x < a < y$.

Twierdzenie 1.18. *Zarówno zbiór $\mathbb{Q}$ liczb wymiernych, jak i zbiór $\mathbb{R} \setminus \mathbb{Q}$ liczb niewymiernych, są gęste w $\mathbb{R}$.*

Uwagi.

Do powtórzenia na ćwiczeniach: liczby naturalne, indukcja matematyczna (np. strony 100-102).

Uwaga: w materiałach jest istotne twierdzenie o istnieniu pierwiastka z liczby naturalnej wraz z dowodem, że jest to albo liczba naturalna (kiedy?), albo **niewymierna**. Ponadto dowód na niewymierność pierwiastków jest w tym materiale.

Ale: skoro niewymierna to (jak się okaże) nie ma dokładnej reprezentacji komputerowej - czyli warto *pomyśleć* jak ją najlepiej obliczyć...

Liczby naturalne i całkowite.

Zbiór liczb naturalnych $\mathbb{N}$ definiowany był przez matematyka włoskiego G. Peano około 1891 r., a definicja podana przez niego ma postać aksjomatyczną. W zbiorze $\mathbb{N}$ można wykonywać **działania dodawania oraz mnożenia.**

Najmniejszym zbiorem zawierającym zbiór liczb naturalnych w którym można wykonywać **działanie odejmowania** (oprócz powyższych dwóch działań) jest zbiór liczb całkowitych, który piszemy $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$ lub jak zwykle oznaczamy symbolem $\mathbb{Z}$. Jest oczywiste, że zbiór $\mathbb{Z}$ jest sumą mnogościową zbioru liczb naturalnych, zbioru liczb całkowitych ujemnych oraz zbioru jednoelementowego, którym jest liczba 0.

Liczby wymierne.

W wyniku praktycznej działalności człowieka związanej z mierzeniem różnorodnych wielkości ukształtowało się pojęcie liczby wymiernej. Zbiór liczb wymiernych oznaczamy zwykle symbolem $\mathbb{Q}$. Zbiór $\mathbb{Q}$ został zdefiniowany na przełomie XIX i XX w. Definicja zbudowana na pojęciu liczby całkowitej podana została przez niemieckiego matematyka i fizyka H.G. Grassmana, natomiast definicja oparta na pojęciu liczby rzeczywistej podana została przez niemieckiego matematyka D. Hilberta.

W zbiorze $\mathbb{Q}$ można wykonywać **działania dodawania, odejmowania, mnożenia i dzielenia z wyjątkiem dzielenia przez 0**, a ponadto zbiór $\mathbb{Q}$ jest gęsty, tzn. nie ma w nim liczb sąsiednich. Liczby wymierne możemy interpretować jako punkty na osi liczbowej. Jeżeli przyjmiemy, że odcinek e jest jednostką na osi liczbowej, to działanie $\frac{r}{s}$ przekształca odcinek e w odcinek $\frac{r}{s}e$, a punkt który jest prawym końcem tego odcinka uważamy za wynik działania $\frac{r}{s}$.

A teraz liczby, które są
"zbyt skomplikowane"
dla komputera...

Czyli: musi je poznać informatyk!

Liczby rzeczywiste - aksjomaty.

Liczby, których nie można przedstawić wzorem $\frac{l}{m}$, gdzie $l, m \in \mathbb{Z}$ oraz $m \neq 0$ nazywamy liczbami niewymiernymi. Przykładami liczb niewymiernych są np. liczby π, $\sqrt{2}$, $\sqrt{18}$ i inne. To jednak tylko przykłady!

W cytowanej literaturze można znaleźć różne definicje zbioru liczb rzeczywistych. W **ujęciu aksjomatycznym** pojęcie zbioru $\mathbb{R}$ oraz arytmetyki liczb rzeczywistych stanowi sześć pojęć pierwotnych $(\mathbb{R}, 0, 1, +, \cdot, <)$, gdzie $\mathbb{R}$ jest pewnym zbiorem , 0 i 1 są elementami $\mathbb{R}$ natomiast $+$ i $\cdot$ oznaczają działania dwuargumentowe określone w $\mathbb{R}$ oraz $<$ oznacza relację dwuargumentową określoną w $\mathbb{R}$. Do tego aksjomaty określają własności tych obiektów.

Do zapoznania się z aksjomatami polecam: tu - strony 6-11, [W], strony 2-6 lub [K] strony 119-120

I Aksjomaty dodawania

Określone jest działanie $+ : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ dodawania:

(1) $x + y = y + x$ — *przemienność dodawania*

(2) $(x + y) + z = x + (y + z)$ — *łączność dodawania*

(3) $\exists\, 0 \in \mathbb{R}\ \forall\, x \in \mathbb{R}: \quad x + 0 = x$ — *istnienie elementu neutralnego dodawania, istnienie zera*

(4) $\forall\, x \in \mathbb{R}\ \exists\, y \in \mathbb{R}: \quad x + y = 0$ — *istnienie elementu przeciwnego*

II Aksjomaty mnożenia

Określone jest działanie $\cdot : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ mnożenia:

(5) $x \cdot y = y \cdot x$ — *przemienność mnożenia*

(6) $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ — *łączność mnożenia*

(7) $\exists\, 1 \neq 0 \in \mathbb{R}\ \forall\, x \in \mathbb{R}: \quad x \cdot 1 = x$ — *istnienie elementu neutralnego mnożenia, istnienie jedynki*

(8) $\forall\, x \neq 0 \in \mathbb{R}\ \exists\, y: \quad x \cdot y = 1$ — *istnienie elementu odwrotnego*

(9) $x \cdot (y + z) = x \cdot y + x \cdot z$ — *rozdzielność mnożenia względem dodawania*

III Aksjomaty porządku

Określona jest relacja $\leq \quad (\subseteq \mathbb{R} \times \mathbb{R})$ porządku:

(10) $x \leq x$ — *zwrotność*

(11) jeśli $x \leq y$ i $y \leq x$, to $x = y$ — *słaba antysymetria*

(12) jeśli $x \leq y$ i $y \leq z$, to $x \leq z$ — *przechodniość*

(13) $x \leq y$ lub $y \leq x$ — *spójność*

(14) jeśli $x \leq y$, to $x + z \leq y + z$

(15) jeśli $0 \leq x$ i $0 \leq y$, to $0 \leq x \cdot y$

IV Aksjomat kresu górnego

(16) Każdy niepusty ograniczony z góry podzbiór w $\mathbb{R}$ ma kres górny.

Uwaga: skoro ludziom trudno przyswoić aksjomaty liczb rzeczywistych i operować nimi, to jak ma to zrobić komputer? Niestety: nie może!

Kluczowe informacje: strony 119-129 !!

To informatyk musi przewidzieć skutki braku liczb rzeczywistych w **arytmetyce komputerowej**!

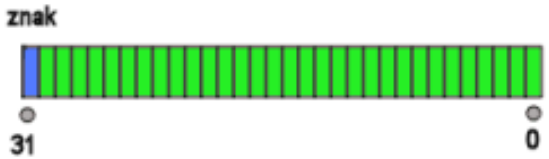
Zarówno liczby, jak i funkcje wydają się być nam dobrze znane, ALE... nie komputerom.

Ograniczenie zakresu arytmetyki komputerowej.

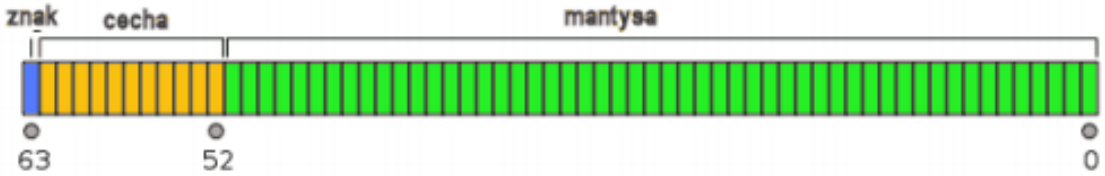
8.2 Liczby reprezentowalne

W pamięci komputera możemy reprezentować jedynie pewien skończony podzbiór zbioru liczb. Używane są różne podzbiory. W przypadku liczb, na których operuje procesor są to pewne podzbiory zbioru liczb całkowitych oraz pewne podzbiory zbioru liczb wymiernych, najczęściej liczby całkowite typu **int** oraz wymierne typu **double**.

Do reprezentacji liczb typu **int** używa się zazwyczaj 32 cyfr dwójkowych, czyli inaczej bitów (rys.8.1). Sama liczba zapisana jest przy pomocy 31 bitów, a 32-gi bit używany jest do zapisania znaku. Pozwala to na zapisywanie liczb całkowitych z zakresu -2147483648 do $+2147483647$.



Rysunek 8.1: 32-bitowe słowo (4 bajty) używane do reprezentacji liczb typu **int**.



Rysunek 8.2: 64-bitowe słowo (8 bajtów) używane do reprezentacji liczb typu **double**.

Liczby typu **double** przechowywane są zazwyczaj na 64 bitach, z czego jeden bit reprezentuje znak s, 11 bitów cechę N, a pozostałe 52 mantysę m będącą ułamkiem o mianowniku 2^{52} z przedziału $[1, 2)$ (rys.8.2). Umożliwa to zapisywanie liczb w postaci

$$x = sm2^N,$$

a także, przy pomocy specjalnego kodu, liczby zero. Wykorzystanie cechy i mantysy pozwala na przybliżanie liczb o bardzo dużym zakresie, od bardzo małych, rzędu 10^{-323}, do bardzo dużych, rzędu 10^{308}, przy zachowaniu w miarę stałej dokładności, wynoszącej około 15 cyfr znaczących.

Liczby rzeczywiste.

Komputer może posługiwać się typem *real*, ale nie są to jednak liczby rzeczywiste, a jest on

skończonym zbiorem reprezentantów przedziałów. Patrz: paragraf 8.2, strony 120-130.

Całą pracę musi wykonać *programista*....

Obliczenia numeryczne = obliczenia (przybliżone) na danych typu *real* .

A już np. liczba rzeczywista $e \in \mathbb{R}$ będzie bezpośrednio potrzebna (np. przy zagadnieniach wizualizacji) - wprowadzimy ją na wykładzie.

Uwaga: w związku z tym nie można oczekiwać, że aksjomaty liczb rzeczywistych (a więc i nasze oczekiwania wobec własności np. działań) bądą spełnione w arytmetyce komputerowej!

Arytmetyka komputerowa.

Każdą liczbę x, $x \neq 0$ można reprezentować jednoznacznie w postaci wykładniczej $x = s \cdot m \cdot 2^c$, gdzie $s \in \{-1, 1\}$ jest znakiem liczby, $m \in [\frac{1}{2}, 1)$ nazywamy **mantysą**, a $c \in \mathbb{Z}$ nazywamy **cechą**. W komputerze m i c są reprezentowane za pomocą skończonej ilości bitów.

W związku ze skończonym zakresem wartości przyjmowanych przez cechę $c_{\min} \leq c \leq c_{\max}$ w reprezentacji zmiennoprzecinkowej można reprezentować tylko liczby ze **skończonego zakresu** $\frac{1}{2}2^{c_{\min}} \leq |x| < 2^{c_{\max}}$ oraz 0. W związku ze skończoną ilością t bitów mantysy reprezentacja zmiennoprzecinkowa może różnić się od wartości ścisłej: $\tilde{x}_t = rd(x) = x(1 + \epsilon)$, gdzie $|\epsilon| \leq 2^{-t}$.

Polecam zapoznać się z tym materiałem! Ponieważ jednak nie do końca wierzę w wykonanie polecenia, to napiszę inaczej: będę uważał ten materiał za obowiązkowy...

Zero. Przybliżenia.

Pewnie większość czytelników napotkała w "C++" funkcję *epsilon()* lub FLT_EPSILON, DBL_EPSILON , LDBL_EPSILON (dla typów zmiennoprzecinkowych). Ta pierwsza wg dokumentacji to "najmniejsza liczba większa niż 1 reprezentowalna w podwójnej precyzji". Czyli przy 64-bit IEEE podwójnej precyzji, mamy 52-bitową mantysę i 11-bitową cechę:

$$1.0000000000000000000000000000000000000000000000000000 \times 2^0 = 1$$

A więc:

$$1.0000000000000000000000000000000000000000000000000001 \times 2^0 = 1 + 2^{-52}$$

Czyli

$$epsilon() = (1 + 2^-52) - 1 = 2^{-52}.$$

O arytmetyce komputerowej - cd.

Może po przeczytaniu poprzednich uwag będą osoby, któe stwierdzą, że wykonywały obliczenia np. na większych liczbach. Tak - mówimy tu o programach korzystających z reprezentowania liczb na poziomie **procesora** (słowa maszynowe).

Jeśli musimy wykonywać dokładniejsze obliczenia, to należy skorzystać z programu, który pomija te ograniczenie (lub napisać własny...), np pakiety matematyczne czy programy obliczeniowe. Wtedy operacje na liczbach ograniczonych wielkością pamięci - ale oczywiście to wymaga więcej operacji i/lub dłuższych czasów obliczeń (*np. szachy i "Deep Blue"*). To i tak ma pewne ograniczenia, nasze zasady **pozostają w mocy**, tylko zakres liczbowy czy precyzja ulega zmianie... Oczywiście, czym innym jest *przechowywanie* liczb - można np. obliczać kolejne miejsca po przecinku (np. π) i zapisywać, ale operowanie na takich liczbach to inny problem.

Dla zainteresowanych...

Tutaj (dla zainteresowanych, bo to nie ten przedmiot) więcej informacji: dlaczego zapisać liczby zmiennopozycyjnie i jak: analiza błędów w metodach numerycznych.

Nas interesuje jak kontrolować obliczenia w arytmetyce komputera (czego nie potrafi - które aksjomaty nie są spełnione) i dlaczego to programista odpowiada za obliczenia.

A tutaj ciekawe materiały o reprezentacji liczb i zestaw ćwiczeń. Polecam!

Część tego wykładu zajmować nam będą metody **JAK** skutecznie reprezentować liczby niewymierne na komputerze - na razie wiemy DLACZEGO to konieczne... Te *algorytmy* znajdą swoje podstawy w matematyce (analizie matematycznej).

Unikamy problemów... liczby rzeczywiste

Na początek banalne zagadnienie:

$$\underbrace{\frac{1}{3}\cdot\frac{1}{3}\cdot\ldots\cdot\frac{1}{3}}_{\text{n - krotnie}}\cdot\underbrace{3\cdot 3\cdot\ldots\cdot 3}_{\text{n - krotnie}} = ??$$

No to proszę zrobić symulację np. w arkuszu kalkulacyjnym dla "dużych" n...

Czyli: nie ma "jednego" zera, a działania nie są przemienne...

Przykład - arkusz kalkulacyjny.

C	D	E	F	G	H	I
	645					
0,333333	0	3	5,5362E+307		0	
	644					
0,333333	5,4189E-308	3	1,8454E+307		1	

Liczymy 1/3 do potęgi 645 i (niżej) do 644 i odpowiednio z 3 do tych potęg, oraz ich iloczyny - ale jak różne wyniki (0 oraz 1)!

Problemy z zerem...

Drugie pytanie: stosujemy wzory na pierwiastki trójmianu czy wzory Viète'a ?? Sprawdzić na

$$w(x) = x^2 - k \cdot x + 1$$

dla odpowiednio (do stosowanej arytmetyki zmiennopozycyjnej) dużych k...

Uwaga: Warto rozważać wzory Viète'a gdy wzory na pierwiastki mogą prowadzić do problemów z obliczaniem pierwiastka (o pierwiastkach - nieco później).

Obliczanie zer równania kwadratowego

$$x^2 - 2px + q = 0 \tag{3}$$

W szkole stosowaliśmy algorytm zakładający obliczanie pierwiastków wzorami

$$x_1 = p + \sqrt{p \cdot p - q} \tag{4a}$$

$$x_2 = p - \sqrt{p \cdot p - q} \tag{4b}$$

Poprawny algorytm wygląda jednak następująco: obliczamy ten pierwiastek, który jest bardziej "odległy od zera np. dla którego mamy we wzorze dodawanie (dla $p > 0$ ryzyko $p - \sqrt{p \cdot p - q}$ "bliskie zeru"), a drugi obliczamy na podstawie wzoru Viète'a $x_1 \cdot x_2 = q$

Kontrprzykład - arkusz...

			x^2 - 2px + q = 0				
	p	150000	300000	x1	sprawdzamy: wzory Viete'a		
					x1*x2 = q ??	NIE	
	q	0,000001	0	x2	x1+x2 = 2p ??	TAK	dlaczego?

A więc obliczenia np. w arkuszu kalkulacyjnym mogą prowadzić do błędnych wyników - uwaga na "zero maszynowe"!

if $p \geq 0$ then

$$\begin{aligned} x_1 &= p + \sqrt{p \cdot p - q} \\ x_2 &= q / x_1 \end{aligned} \tag{5}$$

if $p < 0$ then

$$\begin{aligned} x_2 &= p - \sqrt{p \cdot p - q} \\ x_1 &= q / x_2 \end{aligned} \tag{6}$$

Kiedy algorytm *naiwny* jest niepoprawny? Może on generować znaczny błąd w chwili gdy np. $p \cdot p \gg q$, czyli wtedy $p - \sqrt{p^2 - q} \approx p - \sqrt{p^2} = 0$ (oczywiście: maszynowe zero).

Zadanie samodzielne: Zaimplementować oba te algorytmy. Poszukać takich wartości p i q, dla których algorytmy dają różne wyniki. Który algorytm jest dokładniejszy?

Dla mniej pracowitych: przeanalizować algorytmy ze stron [K] strony 136-137 i zapoznać się z wnioskami ze strony 138.
Przy okazji badań ciągów wyjaśnimy problem...

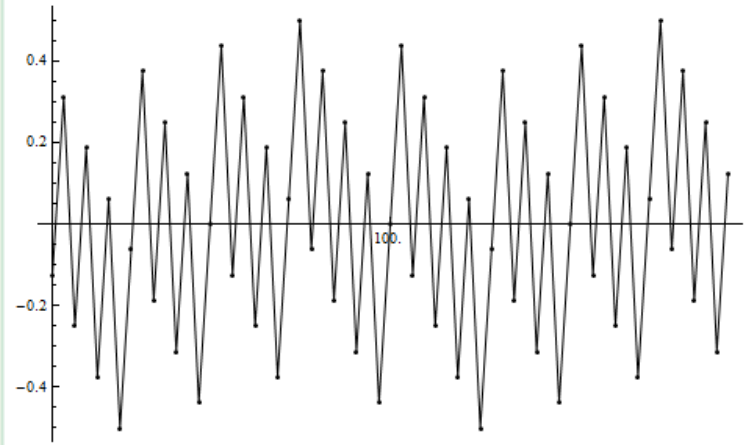
Mathematica.

Ilustracja arytmetyki komputerowej w (dobrym) programie matematycznym "Mathematica". Ma trochę więcej możliwości niż "zwykłe programy"... Prezentacja: Skrypt ilustracyjny - potrzebny darmowy *CDF Player* lub *Mathematica*.

```
SetArithmetic[4, 10];
```

- Teraz przebadamy arytmetyke na tej konkretnej maszynie. Poniższy wykres przedstawia wartość względną błędu obliczenia wartości funkcji f(x) = x^2 w małym sąsiedztwie argumentu x = 100. Widać, że wartość błędu zmienia się nieregularnie. Jednostki na osi pionowej to Ulp (tj. odległość dwóch sąsiednich liczb maszynowych w otoczeniu wartości).

```
MicroscopicErrorPlot[x^2, {x, 100}]
```



- Poniższy wykres pokazuje z kolei wygląd funkcji w arytmetyce komputerowej, ukazując "schodkową" naturę tych funkcji. Liczbę x podnosimy do potęgi a/b i pokazujemy wykres w okolicy liczby x=c.

Istnienie pierwiastka z liczby nieujemnej.

To już było na slajdach jako wprowadzenie, ale teraz czytamy [K] strony 128-129. Koniecznie trzeba zwrócić uwagę na dwa elementy tezy: "pierwiastek istnieje" i "jest co najwyżej jeden" (jednoznaczność). To dwa stałe i konieczne elementy w każdym użyciu obiektu **w informatyce**! Symbol $\exists!$ czytamy właśnie "istnieje dokładnie jeden".

Istotne: czytamy "brązowe" komentarze... Patrz też [W] str. 11-13.

I dalej ważny fakt: twierdzenie 1.19 [W] (str. 15 - por. kolejne slajdy) rozstrzyga, że pierwiastki z liczb naturalnych są albo naturalne, albo niewymierne! Ten ostatni przypadek jest problemem w obliczeniach na komputerze - muszą być obliczane w sposób przybliżony.

Twierdzenie 1.19. *Jeśli $n, k \in \mathbb{N}$ i $k \geq 2$, to $x = n^{1/k}$ jest albo liczbą naturalną, albo liczbą niewymierną.*

Dowód. Aby lepiej zilustrować najważniejszy pomysł dowodu, rozpatrzymy najpierw przypadek $k = 2$.

Przypuśćmy, że $0 < x = \sqrt{n} \notin \mathbb{N}$, ale jednak $x \in \mathbb{Q}$. Wtedy zbiór

$$A = \{m \in \mathbb{N} : \ mx \in \mathbb{N}\}$$

jest niepusty; to wynika wprost z definicji zbioru liczb wymiernych $\mathbb{Q}$. Niech m_0 będzie najmniejszym elementem A; wtedy oczywiście $m_0 x = l \in \mathbb{N}$.

Połóżmy $m_1 = m_0(x - [x])$. Z nierówności $0 < x - [x] < 1$ (pamiętajmy: x nie jest liczbą całkowitą) wynika, że $0 < m_1 < m_0$. Ponadto,

$$m_1 = m_0 x - m_0[x] = l - m_0[x] \in \mathbb{Z}\,,$$

a więc m_1 jest liczbą naturalną, bo $m_1 > 0$. Wreszcie, mamy

$$0 < m_1 x = m_0 x^2 - m_0 x[x] = m_0 n - l[x] \in \mathbb{Z}\,,$$

a więc liczba $m_1 x$ też jest naturalna. To oznacza, że $m_1 \in A$ i $m_1 < m_0$, a przy tym m_0 jest *najmniejszym* elementem w zbiorze A. Otrzymaliśmy sprzeczność, która oznacza, że $x = \sqrt{n}$ nie może być liczbą wymierną. To kończy dowód twierdzenia w przypadku $k = 2$.

Pokażemy teraz, jak rozważyć przypadek ogólny. Załóżmy, że $x = n^{1/k} \notin \mathbb{N}$. Przypuśćmy, że $x \in \mathbb{Q}$; pokażemy, że to założenie prowadzi do sprzeczności. Niech

$$B = \{s \in \mathbb{N} : \ x^s \in \mathbb{N}\}\,.$$

Zbiór $B \subset \mathbb{N}$ jest niepusty ($k \in B$, bowiem $x^k = n$), więc zawiera element najmniejszy s_0; przy tym $s_0 > 1$, gdyż $1 \notin B$. Oznaczmy

$$x^{s_0} = n_0\,. \tag{1.6}$$

Rozważmy zbiór

$$A = \{m \in \mathbb{N} : \text{ wszystkie liczby } mx, mx^2, \ldots, mx^{s_0-1} \text{ są naturalne}\}.$$

Jest to zbiór niepusty, gdyż $x, x^2, \ldots, x^{s_0-1}$ są liczbami wymiernymi. Niech m_0 będzie najmniejszym elementem zbioru A. Dla wygody oznaczmy

$$m_0 x = l_1 \in \mathbb{N}, \quad m_0 x^2 = l_2 \in \mathbb{N}, \quad \ldots, \quad m_0 x^{s_0-1} = l_{s_0-1} \in \mathbb{N}\,.$$

Ponieważ s_0 to najmniejszy element zbioru B, więc

$$0 < \varepsilon := x^{s_0-1} - \left[x^{s_0-1}\right] < 1$$

(liczba x^{s_0-1} nie jest naturalna, gdyż wtedy $s_0 - 1$ należałoby do B). Połóżmy $m_1 = \varepsilon m_0$. Wtedy $0 < m_1 < m_0$. Ponadto,

$$0 < m_1 = m_0\varepsilon = m_0 x^{s_0-1} - m_0\left[x^{s_0-1}\right] = l_{s_0-1} - m_0\left[x^{s_0-1}\right] \in \mathbb{Z}\,,$$

więc m_1 jest liczbą naturalną. Wreszcie, nietrudno sprawdzić, że

$$m_1 x \in \mathbb{N}, \quad m_1 x^2 \in \mathbb{N}, \quad \ldots, \quad m_1 x^{s_0-1} \in \mathbb{N}\,.$$

Istotnie, niech j będzie dowolną z liczb $1, 2, \ldots, s_0 - 1$. Wtedy

$$\begin{aligned} m_1 x^j = m_0\varepsilon x^j &= m_0 x^{j-1} x^{s_0} - m_0 x^j \left[x^{s_0-1}\right] \\ &\overset{(1.6)}{=} m_0 x^{j-1} n_0 - l_j \left[x^{s_0-1}\right] \\ &= \begin{cases} m_0 n_0 - l_1\left[x^{s_0-1}\right] & \text{gdy } j = 1, \\ l_{j-1} n_0 - l_j\left[x^{s_0-1}\right] & \text{gdy } j > 1, \end{cases} \end{aligned}$$

a więc $m_1 x^j$ jest liczbą całkowitą. Do tego oczywiście $m_1 x^j > 0$, więc $m_1 x^j$ jest liczbą naturalną. Zatem, z definicji zbioru A i dowolności j, liczba $m_1 \in A$.

Otrzymaliśmy więc

$$m_0 = \inf A > m_1 \in A\,.$$

Jest to sprzeczność, która kończy dowód. □

Funkcja pierwiastek $\sqrt{x}$.

Z podanych materiałów wynika, w szczególności, **istnienie pierwiastka** z każdej liczby rzeczywistej nieujemnej. **Jak go jednak obliczyć na komputerze?**

Metoda Newtona-Raphsona (bazująca na geometrii - pierwiastek jako pole kwadratu o boku x, rozpoczynamy od prostokąta i zmniejszamy różnicę pomiędzy długościami boków korzystając ze średniej arytmetycznej) $a > 0$:

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right) \ , \ x_1 = \frac{a}{2}.$$

Sprawdzić, że ten ciąg jest zbieżny do $\sqrt{a}$ - lub znaleźć kres dolny zbioru takich liczb...

Jak widać: osiąganie kresu zbioru i tak najprościej wykonać na komputerze za pomocą ciągów i dlatego sporo informacji o nich podamy na kolejnym wykładzie!

Systemy pozycyjne.

Wszyscy wiedzą, że komputer nie korzysta z zapisów liczb w systemie dziesiętnym.

Prezentacja: konwersja w programie "Mathematica".

Prezentacja nie rozstrzyga wszystkiego. To pytanie: w zapisie dziesiętnym liczby wymierne mają rozwinięcia skończone lub nieskończone okresowe. A jak jest w zapisie dwójkowym takich liczb? Np. $\left(\frac{1}{3}\right)_2$?
(wskazówka do poszukiwań: " błędy reprezentacji")

Tutaj: pewna wskazówka (ale nie dowód)... w formacie CDF...

Kresy - definicje.

Por. strona 106. Niech $A \subseteq \mathbb{R}$ będzie niepusty.

Ograniczeniem górnym zbioru A nazywamy liczbę $s \in \mathbb{R}$ spełniającą: : $s \geqslant a$ dla wszystkich elementów $a \in A$.

Analogicznie ograniczeniem dolnym zbioru nazywamy liczbę niewiększą od wszystkich liczb tego zbioru.

Kresem górnym zbioru A nazywamy najmniejsze z górnych ograniczeń tego zbioru, tj. liczbę $s \in \mathbb{R}$ spełniającą:

(1) s jest ograniczeniem górnym zbioru A;

(2) jeśli $s' \in \mathbb{R}$ jest ograniczeniem górnym zbioru A, to $s \leqslant s'$.

Analogicznie **kresem dolnym** zbioru nazywamy największe ograniczenie dolne tego zbioru.

Kres górny zbioru A oznaczamy $\sup(A)$, kres dolny $\inf(A)$.

Kresy.

Aksjomat kresu dolnego: każdy niepusty i ograniczony z dołu podzbiór zbioru liczb rzeczywistych ma kres dolny.

Kresy zbiorów: np. obliczanie $\sqrt{x}$ (bazujące na nierówności pomiędzy średnimi): niech $a_0 = 1$, $b_0 = x > 0$. Definiujemy

$$a_{n+1} = \frac{2a_n \cdot b_n}{a_n + b_n},$$

$$b_{n+1} = \frac{a_n + b_n}{2}.$$

Sprawdzić, że $\sqrt{x} = \sup_{n\in\mathbb{N}} \{a_n\} = \inf_{n\in\mathbb{N}} \{b_n\}$.

Literatura dodatkowa.

Podoba mi się też skrypt prof. Paluszyńskiego (UWr) te części materiału, które są zgodne z naszym programem można poczytać (raczej na ćwiczenia).

Tajne/poufne.

Przykładowe zagadnienia na egzamin - po tym wykładzie...

- Zasada Archimedesa (aksjomat $\mathbb{R}$) i jej znaczenie dla arytmetyki komputerowej (najlepiej 3 podane na wykładzie zastosowania, uzadsadnić: **np.** gęstość $\mathbb{Q}$ w $\mathbb{R}$; $1/n < \varepsilon$; czy funkcja "entier").
- Liczby wymierne w zapisie binarnym mogą mieć rozwinięcia okresowe (np. $a = 0,1$). Ich reprezentacja *double* lub *float* w arytmetyce komputerowej może więc być obarczona błędem i warto unikać obliczeń za pomocą takich liczb. Zbiór liczb wymiernych jest gęsty w zbiorze liczb rzeczywistych $\mathbb{R}$: a czy zbiór liczb wymiernych o skończonym rozwinięciu binarnym $\mathcal{B}$ również jest gęsty w $\mathbb{R}$? Rozważ **ciągi** liczb z $\mathcal{B}$ i ich granice (punkty skupienia zbioru $\mathcal{B}$).
- Podaj przykłady aksjomatów liczb rzeczywistych, które nie są spełnione w arytmetyce komputerowej (różnej precyzji).
- Problem istnienia pierwiastka z liczby nieujemnej (twierdzenie). Kiedy $\sqrt{x}$ może być liczbą wymierną?