Analiza matematyczna dla informatyków.

Mieczysław Cichoń, ver. 4.3/2023

Mieczysław Cichoń - WMI UAM

Działania na szeregach zbieżnych.

Trzy podstawowe własności już znamy: suma, różnica i iloczyn przez stałą szeregów zbieżnych są zbieżne.

Pytanie: **czy i jak** można mnożyć szeregi (czyli również liczby reprezentowane nimi na komputerze)?

Mnożenie szeregów jest uogólnieniem mnożenia sum skończonych. Załóżmy, że dane są dwa szeregi zbieżne $\sum_{n=1}^{\infty} a_n = a$ oraz $\sum_{n=1}^{\infty} b_n = b$. Dla utworzenia iloczynu szeregów $(\sum_{n=1}^{\infty} a_n) \cdot (\sum_{n=1}^{\infty} b_n)$ należy dodać wszystkie możliwe składniki postaci $a_i b_j$. W tym celu tworzymy tablicę

$$\begin{array}{cccc} a_1 b_1 & a_1 b_2 & a_1 b_3 \ldots \\ a_2 b_1 & a_2 b_2 & a_2 b_3 \ldots \\ a_3 b_1 & a_3 b_2 & a_3 b_3 \ldots \\ \multicolumn{3}{r}{\ldots\ldots\ldots\ldots\ldots\ldots} \end{array}$$

oraz ustalamy sposób dodawania wyrazów występujących w tej tablicy.

Iloczyn Cauchy'ego szeregów.

Oznaczmy

$$\sum_{n=1}^{\infty} c_n = \left(\sum_{n=1}^{\infty} a_n\right)\left(\sum_{n=1}^{\infty} b_n\right) .$$

(a) Jeżeli wyrazy c_n zdefiniujemy wzorem

$$c_n = \sum a_i b_j \ ,$$

gdzie $n = i \cdot j$, a więc iloczyn wskaźników jest stały, to otrzymujemy sposób mnożenia metodą Dirichleta.

(b) Gdy $c_n = a_n b_1 + a_{n-1} b_2 + \ldots + a_1 b_{n-2} + a_0 b_{n-1}$, a więc dodajemy wyrazy leżące na bocznych przekątnych powyższej tablicy, to otrzymujemy sposób mnożenia metodą Cauchy'ego.

(c) Jeżeli wyrazy szeregu $\sum_{n=1}^{\infty} c_n$ zdefiniujemy wzorem

$$c_n = a_n b_1 + a_n b_2 + \ldots + a_n b_{n-1} + a_n b_n + a_{n-1} b_n + \ldots + a_2 b_n + a_1 b_n$$

to otrzymujemy sposób mnożenia szeregów według kwadratów (sumujemy te wyrazy powyższej tablicy, które leżą na dolnym i prawym boku odpowiedniego kwadratu tablicy).

Python.

Tu wersja dla obliczania kolejnych elementów iloczynu:

```
def cauchy_product(a, b):
    # Obliczamy iloczyn dwóch szeregów za pomocą metody Cauchy'ego.
    result = [0] * (len(a) + len(b) - 1)
    for i in range(len(a)):
        for j in range(len(b)):
            result[i+j] += a[i] * b[j]
    return result
```

Stosujemy:

```
a = [1, 2, 3
b = [4, 5, 6]
product = cauchy_product(a, b)
print("Iloczyn Cauchy'ego: ", product)
```

Czy (i jak) możnaby to kontynuować? Proszę uzupełnić kod o sumowanie uzyskanego szeregu!

Zbieżności iloczynów szeregów.

Twierdzenie. (twierdzenie Cauchy'ego). *Jeżeli szeregi $\sum_{n=1}^{\infty} a_n$ oraz $\sum_{n=1}^{\infty} b_n$ są bezwzględnie zbieżne, to ich iloczyn $\sum_{n=1}^{\infty} c_n$ jest bezwzględnie zbieżny i zachodzi wzór*

$$\sum_{n=1}^{\infty} c_n = a \cdot b \ ,$$

przy czym sposób sumowania jest dowolny.

Twierdzenie. (twierdzenie Mertensa[1]). *Jeżeli szeregi $\sum_{n=1}^{\infty} a_n$ oraz $\sum_{n=1}^{\infty} b_n$ są zbieżne oraz przynajmniej jeden z nich jest bezwzględnie zbieżny, to ich iloczyn $\sum_{n=1}^{\infty} c_n$ otrzymany metodą mnożenia Cauchy'ego jest zbieżny i zachodzi wzór $\sum_{n=1}^{\infty} c_n = a \cdot b \quad .$*

[1] F. Mertens: profesor Uniwersytetu Jagiellońskiego, w okresie 1865-1884

Sumy szeregów.

Na ogół obliczanie sumy szeregu może być skomplikowane. Na razie poznaliśmy, w zasadzie, jeden przypadek, gdy potrafimy ją obliczyć: szeregi geometryczne: **jeżeli** $|q| < 1$, **to**

$$\sum_{n=1}^{\infty} q^n = \frac{1}{1-q}.$$

Nieco później poznamy kilka innych metod (na razie to niemożliwe...), ale zauważmy, że **gdybyśmy** wiedzieli, że szeregi są zbieżne, to można skorzystać z własności szeregów zbieżnych i najpierw sprawdzać zbieżność kolejnych, a w efekcie co najmniej **obliczać przybliżone sumy szeregów** - co może być wystarczające (dla sum częściowych S_n szeregów **zbieżnych (!)** mamy: $\lim_{n\to\infty} S_n = S$, a więc S_n "przybliża" wartość S).

Python.

Jeśli już znamy trochę matematyki to obliczenia są łatwe:

```
def geometric_series_sum(q):
    if abs(q) >= 1:
        return float('inf')  # szereg jest rozbieżny
    else:
        return 1 / (1 - q)
```

Czyli co uzyskamy po takim obliczeniu?:

```
q = 0.25
sum = geometric_series_sum(q)
print("Suma: ", sum)
```

Oznacza to, że głównym celem dla nas jest

sprawdzanie CZY szereg jest zbieżny.

Takie twierdzenia, które orzekają zbieżność szeregu (lub: nie) w zależności od pewnych własności ciągów (a_n) nazywamy kryteriami zbieżności.

Szeregi o wyrazach nieujemnych.

Rozważmy teraz szeregi $\sum_{n=1}^{\infty} a_n$ o wyrazach dodatnich $a_n > 0$ lub nieujemnych $a_n \geqslant 0$ dla $n \in \mathbb{N}$.

Jest oczywiste, że przy takim założeniu o wyrazach szeregu ciąg sum częściowych (s_n) jest albo rosnący albo niemalejący.

Wynika stąd więc, że szeregi o wyrazach nieujemnych są albo zbieżne, co możemy zapisać $\sum_{n=1}^{\infty} a_k < \infty$, albo są rozbieżne do $+\infty$.

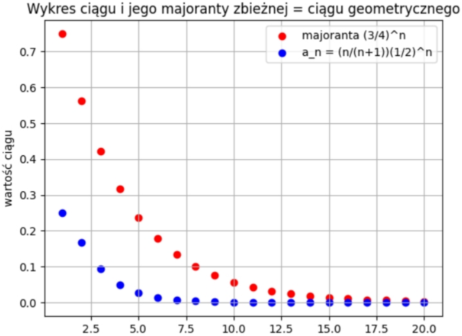
Podamy kilka twierdzeń, które pozwalają rozstrzygnąć ten problem.

Kryterium porównawcze.

Jeżeli wyrazy szeregów $\sum_{n=1}^{\infty} a_n$ oraz $\sum_{n=1}^{\infty} b_n$ dla prawie wszystkich n spełniają nierówności

$$0 \leqslant a_n \leqslant b_n \ ,$$

to gdy szereg $\sum_{n=1}^{\infty} b_n$ (nazywany majorantą) jest zbieżny, wówczas szereg $\sum_{n=1}^{\infty} a_n$ jest także zbieżny, natomiast gdy szereg $\sum_{n=1}^{\infty} a_n$ (nazywany minorantą) jest rozbieżny to rozbieżny jest również szereg $\sum_{n=1}^{\infty} b_n$.



Stwierdzenie 4.12 (kryterium porównawcze, wersja I). *Załóżmy, że* $a_n, b_n > 0$ *i istnieją takie liczby* $c > 0$ *i* $n_0 \in \mathbb{N}$, *że* $a_n \leq c \cdot b_n$ *dla wszystkich* $n \geq n_0$. *Wtedy*

(a) *Ze zbieżności szeregu* $\sum_{n=1}^{\infty} b_n$ *wynika zbieżność szeregu* $\sum_{n=1}^{\infty} a_n$;

(b) *Z rozbieżności szeregu* $\sum_{n=1}^{\infty} a_n$ *wynika rozbieżność szeregu* $\sum_{n=1}^{\infty} b_n$.

Przykład 4.15. Jeśli $q \in (0,1)$, to szereg o wyrazach $a_n = nq^n$ jest zbieżny. Istotnie, weźmy dowolne $s \in (q,1)$. Ponieważ $(n+1)/n \to 1$, więc dla wszystkich dostatecznie dużych n jest

$$\frac{a_{n+1}}{a_n} = \frac{n+1}{n} q < s = \frac{b_{n+1}}{b_n}, \qquad \text{gdzie } b_n = s^n.$$

Ponieważ dla każdego $s \in (0,1)$ szereg geometryczny $\sum s^n$ jest zbieżny, więc szereg $\sum nq^n$ jest zbieżny. To wynika z punktu (a) ostatniego kryterium. $\square$

Przykład 4.16. Postępując praktycznie tak samo, jak w poprzednim przykładzie, można stwierdzić, że szereg $\sum_{n=1}^{\infty} n^k q^n$, gdzie k jest ustaloną liczbą naturalną i $q \in (0,1)$, jest zbieżny. $\square$

Kryterium o zagęszczaniu

Twierdzenie. Jeżeli (a_n) jest malejącym ciągiem liczb rzeczywistych, to szeregi

$$\sum_{n=1}^{\infty} a_n$$

oraz

$$\sum_{n=1}^{\infty} b_n,$$

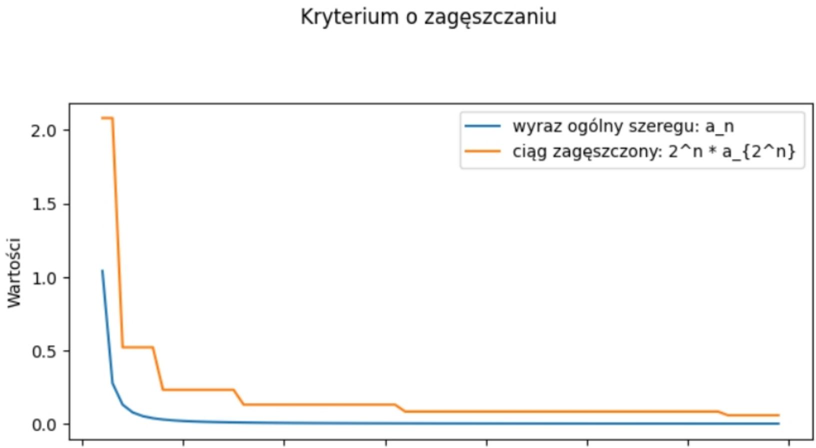
gdzie

$$b_n = 2^n \cdot a_{2^n}$$

są albo jednocześnie zbieżne, albo jednocześnie rozbieżne.

Przykład: szereg harmoniczny.

Niech $a_n = \frac{1}{n}$. Wtedy $b_n = 2^n \cdot \frac{1}{2^n} = 1$. Czyli szereg $\sum_{n=1}^{\infty} b_n = \sum_{n=1}^{\infty} 1 = +\infty$ (nie zachodzi warunek konieczny zbieżności szeregu $\sum_{n=1}^{\infty} b_n$. Na mocy kryterium o zagęszczaniu szereg harmoniczny jest więc rozbieżny.


Kryterium o zagęszczaniu
wyraz ogólny szeregu: a_n
ciąg zagęszczony: 2^n * a_{2^n}
Wartości
2.0
1.5
1.0
0.5
0.0

Kryterium ilorazowe (d'Alemberta).

Kryterium ilorazowe, (d'Alemberta). Niech dany będzie szereg $\sum_{n=1}^{\infty} a_n$, gdzie $a_n > 0$ dla $n = 1, 2, \ldots$ oraz niech

$$g = \lim_{n\to\infty} \frac{a_{n+1}}{a_n} \ .$$

Szereg $\sum_{n=1}^{\infty} a_n$ jest zbieżny, gdy $g < 1$; jest rozbieżny, gdy $g > 1$, (przypadek gdy $g = 1$ wymaga **osobnego zbadania**).

Dla szeregu $\sum_{n=1}^{\infty} \frac{1}{n} \equiv \sum_{n=1}^{\infty} A_n$ spełniony jest warunek $\lim_{n\to\infty} \frac{A_{n+1}}{A_n} = \lim_{n\to\infty} \frac{n}{n+1} = 1$, a jak wiemy szereg ten jest rozbieżny.

Dla szeregu $\zeta(2) = \sum_{n=1}^{\infty} \frac{1}{n^2} \equiv \sum_{n=1}^{\infty} B_n$ mamy warunek $\lim_{n\to\infty} \frac{B_{n+1}}{B_n} = \lim_{n\to\infty} \left(\frac{n}{n+1}\right)^2 = 1$, a szereg $\zeta(2)$ jest jednak zbieżny.

Przykład.

Rozpatrzmy szereg

$$\sum_{n=1}^{\infty} \frac{3^n n!}{n^n}.$$

Zbadajmy jego zbieżność.

R o z w i ą z a n i e. Dany szereg jest rozbieżny, ponieważ po zastosowaniu kryterium ilorazowego stwierdzamy, że

$$\lim_{n\to\infty} \frac{3^{n+1}(n+1)!}{(n+1)^{n+1}} \cdot \frac{n^n}{3^n n!} = 3 \lim_{n\to\infty} \left(\frac{n}{n+1}\right)^n = \frac{3}{e} > 1 \ .$$

Szereg jest rozbieżny.

Kryterium pierwiastkowe (Cauchy'ego).

Załóżmy, że dany jest szereg $\sum_{n=1}^{\infty} a_n$ taki, że $a_n \geqslant 0$ i niech

$$q = \lim_{n\to\infty} \sqrt[n]{a_n}$$

lub

$$q = \limsup_{n\to\infty} \sqrt[n]{a_n}\ .$$

Szereg $\sum_{n=1}^{\infty} a_n$ jest zbieżny, gdy $q < 1$, jest rozbieżny gdy $q > 1$, (w przypadku gdy $q = 1$ kryterium Cauchy'ego nie daje rozstrzygnięcia).

Kryterium Raabe'go.

Jest ono mocniejsze od kryterium Cauchy'ego, a więc także mocniejsze od kryterium d'Alemberta (ale za to jego warunek jest nieco bardziej skomplikowany do obliczenia...).

Kryterium Raabe'go. Niech dany będzie szereg $\sum_{n=1}^{\infty} a_n$, którego wyrazy są dodatnie i niech spełniony będzie następujący warunek

$$\lim_{n\to\infty} n\left(\frac{a_n}{a_{n+1}} - 1\right) = \gamma \ .$$

Jeżeli $\gamma > 1$, to szereg $\sum_{n=1}^{\infty} a_n$ jest zbieżny, jeżeli $\gamma < 1$, to szereg $\sum_{n=1}^{\infty} a_n$ jest rozbieżny, natomiast w przypadku gdy $\gamma = 1$ kryterium nie daje rozstrzygnięcia.

Szeregi naprzemienne.

To postać szeregu, która jest często spotykana, a co więcej ma bardzo proste kryterium zbieżności.

Szeregiem naprzemiennym lub przemiennym nazywamy szereg postaci $\sum_{n=1}^{\infty}(-1)^{n+1}a_n$, gdzie $a_n \geqslant 0$ dla każdej liczby naturalnej n. Widzimy więc, że szereg naprzemienny to szereg postaci

$$a_1 - a_2 + a_3 - a_4 + \ldots + (-1)^{n+1}a_n + \ldots$$

Kryterium Leibniza. Jeżeli (a_n) jest ciągiem malejącym zbieżnym do zera, to szereg naprzemienny

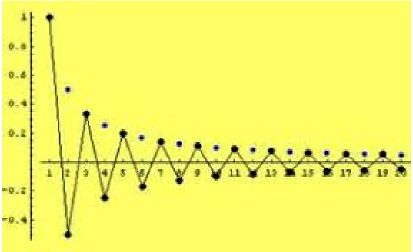
$$\sum_{n=1}^{\infty}(-1)^{n+1}a_n$$

jest zbieżny. Co więcej można oszacować błąd przybliżenia jego sumy S przez sumy częściowe S_n:

$$|S - S_N| \leqslant |a_{N+1}|.$$

Szeregi naprzemienne II.

Ostatni fragment tezy kryterium Leibniza jest kluczowy w zastosowaniu szeregów naprzemiennych w informatyce - i tłumaczy występowanie algorytmów, w których kolejne "poprawki" obliczeń są brane kolejno "z nadmiarem" i z"z niedomiarem". Zawsze znamy precyzję oszacowania...



Wyrazy (a_n) naprzemiennie zmieniają znak...

Python.

```
def alternating_series_sum(n):
    s = 0
    for i in range(1, n+1):
        s += ((-1) ** (i+1)) / i
    error = abs(s - (-1)*((-1) ** (n+1)) / (n+1))
    return s, error
```

Proszę zwrócić uwagę na wzór na szacowanie błędu...

Uwagi.

Wróćmy do zagadnienia poruszanego na pierwszym wykładzie: jak wyznaczyć błąd przybliżenia np. liczby π skoro nie znamy jej dokładnej wartości? Przecież nie możemy tych wielkości odjąć!

Zapamiętajmy: samą liczbę π można wyznaczyć jako sumę szeregu na wiele różnych metod, np. wspomniany wcześniej algorytm Chudnovskiego jest "najszybszy", ale jednak *potrzebujemy* szeregów naprzemiennych i kryterium Leibniza. Może liczymy nim powoli, ale za to szacujemy błąd i to jest cenne.

```
import math

def compute_pi(n):
    pi = 0
    sign = 1
    for i in range(n):
        pi += sign / (2*i+1)
        sign *= -1
    return pi*4

n = 1000
approx_pi = compute_pi(n)
print("Przybliżona wartość liczby pi: {:.4f}".format(approx_pi))
print("Różnica między przybliżeniem a wartością dokładną: {:.10f}".format(math.pi - approx_pi))
```

```
Przybliżona wartość liczby pi: 3.1406
Różnica między przybliżeniem a wartością dokładną: 0.0009999997
```

Kryterium Dirichleta.

Twierdzenie. Jeśli ciąg sum częściowych (s_n) (szeregu utworzonego z ciągu (a_n)) jest ograniczony, a ciąg (b_n) dąży monotonicznie do zera, to szereg $\sum_{n=1}^{\infty} a_n \cdot b_n$ jest zbieżny.

Przykład. Rozważmy szereg liczbowy $\sum_{n=1}^{\infty} \frac{\sin n}{n}$. Chcemy zbadać jego zbieżność.

Możemy zastosować kryterium Dirichleta, przyjmując:
$a_n = \frac{1}{n}$, $b_n = \sin n$.
Ciąg a_n jest malejący i $\lim_{n\to\infty} a_n = 0$. Ciąg b_n jest ciągiem o monotonicznie malejących wartościach i jest ograniczony (bo $|\sin n| \leqslant 1$).
Zatem na mocy kryterium Dirichleta, szereg $\sum_{n=1}^{\infty} \frac{\sin n}{n}$ jest zbieżny.

Kryterium Abela.

Twierdzenie. Jeśli szereg $\sum_{n=1}^{\infty} a_n$ jest zbieżny, zaś ciąg $(b_n)_{n \geqslant 1}$ jest monotonicznie zbieżny, to szereg $\sum_{n=1}^{\infty} a_n \cdot b_n$ jest zbieżny.

Może wydawać się, że te kryteria będą mało przydatne w informatyce. Duży błąd! To one będą podstawowym narzędziem w badaniach pewnej klasy szeregów (trygonometrycznych, Fouriera), które będą stosowane np. w teorii sygnałów, analizie czy kompresji obrazów,

- W dziedzinie analizy numerycznej, kryteria Abela i Dirichleta są często stosowane w algorytmach numerycznych do analizy zbieżności i oceny błędów numerycznych. Na przykład, w **algorytmie Romberga**, kryterium Dirichleta jest wykorzystywane do oceny dokładności numerycznej, poprzez analizę zbieżności szeregu interpolacyjnego. W trakcie algorytmu tworzona jest tabela wartości całek na coraz drobniejszych siatkach. Dla każdej kolejnej siatki obliczana jest ekstrapolacja Richardsona, a następnie sprawdzana jest zbieżność szeregu interpolacyjnego: oceniana jest przy pomocy kryterium Dirichleta. Jeśli szeregi te są zbieżne, to można uznać, że wartość całki na siatce h_i została obliczona z dostateczną dokładnością, i można zakończyć algorytm. Jeśli szeregi te nie są zbieżne, to algorytm kontynuuje, tworząc kolejną tabelę wartości na siatkach o jeszcze mniejszych długościach boków, aż do uzyskania wymaganej dokładności.

  Czyli kryteria te są używane do określenia, kiedy warto zatrzymać iteracje algorytmu, aby uzyskać odpowiednio dokładne wyniki i ocenić czy następne iteracje algorytmu przyniosą znaczącą poprawę wyniku, czy też błąd osiąga już akceptowalny poziom i nie ma potrzeby kontynuowania obliczeń.

- W dziedzinie przetwarzania sygnałów, kryteria te są wykorzystywane do analizy i projektowania filtrów cyfrowych. W tym przypadku szereg Fouriera jest stosowany do opisu właściwości sygnału i filtrów, a kryteria te są wykorzystywane do oceny zbieżności szeregu Fouriera i projektowania efektywnych filtrów cyfrowych.

  W projektowaniu filtrów cyfrowych, stosuje się te kryteria aby dobrać odpowiednią liczbę próbek sygnału, co pozwala na efektywną reprezentację i implementację filtrów. Poprzez odpowiedni dobór ilości współczynników szeregu Fouriera, można ograniczyć rozmiar i złożoność filtrów cyfrowych, co jest ważne w zastosowaniach o ograniczonych zasobach obliczeniowych.

Kryterium Abela opisuje zachowanie szeregów Fouriera w dziedzinie częstotliwości i określa, kiedy szereg jest zbieżny punktowo w dziedzinie czasu. Jest używane do określenia, czy można ograniczyć ilość współczynników szeregu Fouriera, aby zachować zadowalającą jakość reprezentacji sygnału.

Kryterium Dirichleta, z kolei, jest używane do określenia warunków, które muszą być spełnione przez szereg Fouriera, aby był zbieżny do funkcji pierwotnej (sygnału) w dziedzinie czasu. Pomaga w zrozumieniu, jakie wartości współczynników szeregu Fouriera są wymagane, aby sygnał mógł być dokładnie odtworzony.

A może wystarczy komputer?

Niektórym może się wydawać, że zamiast używać matematyki - kryterów zbieżności, to wystarczy symulacja komputerowa. *Pomarzyć można!*

Przy w miarę prostych szeregach to może być poprawna sugestia (ale i tak kryterium rozstrzyga, a nie sugeruje). Ale bywa gorzej.

Na początek szereg harmoniczny $\sum_{n=1}^{\infty} \frac{1}{n}$. Liczymy sumy częściowe (tu przyda się komputer) i małe zaskoczenie: suma 100 wyrazów $S_{100} \approx 5,6$ - dość daleko od hipotezy rozbieżności. No to milion: $S_{1000000} \approx 14,8$. nadal "niezbyt duża" liczba. No to kiedy będzie więcej niż 100? Ktoś chce zrobić symulację? Nie radzę! Potrzeba około 10^{43} wyrazów! A i tak "jak daleko stąd do nieskończoności"! Nie radzę wnioskować o zbieżności szeregu na podstawie symulacji komputerowych...

```
[2]: def harmonic_recursive(n):
         if n == 1:
             return 1
         else:
             return 1.0/n + harmonic_recursive(n-1)
```

```
[3]: harmonic_recursive(1000)
```

```
[3]: 7.485470860550345
```

```
[11]: harmonic_recursive(2000)
```

```
[11]: 8.178368103610282
```

```
[10]: harmonic_recursive(2600)
```

```
[10]: 8.440674684275999
```

A może wystarczy komputer - cd I.

No to może komputer pozwoli sprawdzić **łatwo** rozbieżność?

Niestety - proszę zapoznać się z przykładami 4.24 i 4.25 w materiale [W] strony 49-51. Mogłoby się wydawać, że szereg Kempnera jest "zbliżony" do harmonicznego, ale jest zbieżny!

Przykład 4.25 (szereg Kempnera). Niech $\mathcal{A}$ będzie zbiorem tych liczb naturalnych, w których zapisie dziesiętnym w ogóle nie występuje cyfra 9. Wtedy szereg

$$\sum_{n\in\mathcal{A}} \frac{1}{n}$$

jest *zbieżny*, a jego suma S nie przekracza liczby 80. Aby się o tym przekonać, oznaczmy

Porównać też: [K] strony 189-193.

Czy są jakieś pomysły jak wykorzystać komputer do obliczenia przybliżonej wartości tej sumy szeregu?
Bo na razie nikomu nie udało się tego dokonać :-)

A może wystarczy komputer - cd II.

A może szereg harmoniczny to jakiś wyjątek? Niestety - może być nawet znacznie gorzej... Jeden przykład:

$$\sum_{n=3}^{\infty} \frac{1}{n \ln n \ln(\ln n)}.$$

Proszę spróbować policzyć sumy częściowe, ale nie ustawiać zbyt wysoko swoich wymagań na sumę! Nawet $S_N \approx 10$ to trudne ($N \approx 10^{0,7 \cdot 10^{90}}$) i nie ma szans na domowym komputerze...

To pozwala mi przypomnieć problem "stopu" algorytmu rekurencyjnego, gdyby ktoś badał przyrost dwóch kolejnych obliczeń, to bardzo szybko wyjdzie mu "zero maszynowe" i ... błędny wniosek...

Zostańmy przy matematyce... Patrz też [W] strona 49 (przykłady 4.21 i 4.22).