Dodatek do skryptu z Analizy Matematycznej I

Sławomir Kolasiński, Michał Jóźwikowski

4 listopada 2013

1 Wprowadzenie

1.1 Programy typu CAS

Niniejszy dodatek ma na celu zaprezentowanie sytuacji, w których obliczenia komputerowe ułatwiają rozwiązywanie problemów matematycznych. Oczywiście komputer nie może zastąpić człowieka w procesie uzasadniania i dowodzenia różnych twierdzeń matematycznych. Może jednak przeprowadzić za nas żmudne obliczenia. Niekiedy sama obserwacja wystarczająco dużej porcji danych eksperymentalnych pozawala wyciągać wnioski natury ogólnej i naprowadza na poprawne rozwiązanie zadania. Zaprezentujemy tutaj przykłady podobnych sytuacji.
Obecnie mamy do dyspozycji wiele narzędzi do prowadzenia obliczeń na komputerach. Są to programy typu CAS - z angielskiego *Computer Algebra System*. W [2] można znaleźć listę istniejących narzędzi tego typu wraz z podsumowaniem ich możliwości. Wiele z nich jest dostępnych jako wolne oprogramowanie, a funkcjonalność większości jest w zupełności wystarczająca do naszych zastosowań.
W niniejszym skrypcie będziemy posługiwać się dwoma programami: komercyjnym *Mathematica* oraz wolnym *Maxima*. Program *Mathematica* w wersji 6.0.1.0 jest dostępny dla studentów wydziału MIM w laboratorium komputerowym na maszynie `students`. Program *Maxima* każdy może zainstalować na własnym komputerze na zasadach wolnego oprogramowania. Warto też wspomnieć o narzędziu [9] dostępnym za darmo w formie serwisu internetowego. Serwis ten potrafi wykonywać podstawowe operacje matematyczne jak obliczanie granic funkcji, różniczkowanie, całkowanie i wiele więcej.
Skrypt podzielony jest na części, których objętość powinna odpowiadać mniej więcej jednym zajęciom laboratoryjnym. Na początku każdej części podajemy właściwy dla niej zakres materiału, a także podstawowe polecenia programu *Mathematica*, które Czytelnik powinien przyswoić. Zazwyczaj nie będziemy komentować treści i składni komend, uważając że przykłady ich praktycznego użycia są same w sobie dostatecznym wyjaśnieniem. Domyślnie wszystkie obliczenia prowadzimy w *Mathematice*[1]. Aby przepisać je do programu *Maxima* należy użyć słownika podanego na początku każdej części. Niektóre polecenia nie posiadają swoich odpowiedników, lecz przy odrobinie wysiłku można je samemu zaimplementować. W zadaniach do samodzielnego rozwiązania mogą także pojawić się nieopisane w skrypcie funkcje. Czytel-

[1] W wersji 7.0.1.0 dostępnej na wydziale MIM.

nik sam będzie musiał znaleźć odpowiednie definicje w dokumentacji dostępnej w internecie [5] oraz [7].

1.2 Obliczenia symboliczne

Nowe umiejętności: operacje algebraiczne, obliczenia numeryczne, przekształcanie wyrażeń, funkcje elementarne, definiowanie własnych funkcji.

Skrypt: Rozdziały 1.1, 3. 4.4. i 4.5. (podlinkować)

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`2-4^5/7`	`2-4^5/7`	arytmetyka
`2 3*7`	`2*3*7`	mnożenie w *Mathematice* można zapisać $\star$ lub spacją
`N[Pi],N[Pi,5]`	`bfloat(%pi),` `fpprec: 5; bfloat(%pi)`	wartość numeryczna liczby (z ilością cyfr znaczących)
`%, %%, %%%`	`%, %th(`i`)`	użycie poprzednich wyników
`Simplify,FullSimplify`	`ratsimp, trigsimp, radcan, fullratsimp,`	upraszcza wyrażenie
`Sin[x],Exp[x],Log[x]`	`sin(x),exp(x),log(x)`	funkcje elementarne
`Tan[x], ArcSin[x]...`	`tan(x), asin(x)...`	
`Expand`	`expand`	wymnaża wyrażenie
`Collect`	`facsum`	porządkuje wyrażenie względem potęg
`Factor`	`factor`	sprowadza wyrażenie do postaci iloczynowej
`Together`	`ratsimp`	sprowadza ułamki do wspólnego mianownika
`Apart`	`partfrac`	rozkłada na ułamki proste
`f[x_]:=x^3`	`f(x):=x^3`	funkcje użytkownika
`Clear[f]`	`kill(f)`	usuwa definicję zmiennej

W tym rozdziale omówimy podstawy prowadzenia obliczeń, przekształcania wyrażeń i używania funkcji w programie *Mathematica*. Przedstawimy tu w sposób systematyczny szereg prostych operacji i procedur, które będę używane w dalszej części niniejszego skryptu. Ich stosowanie jest zazwyczaj intuicyjne, dlatego też pominięcie tego rozdziału nie powinno mieć wpływu na zrozumienie pozostałych. Niemniej polecamy go szczególnie tym Czytelnikom, którzy jeszcze nigdy nie używali programów typu CAS.
Prowadzenie obliczeń z użyciem podstawowych operacji algebraicznych (dodawanie, odejmo-

wanie, mnożenie, dzielenie, potęgowanie) jest bardzo naturalne i nie wymaga specjalnych wyjaśnień. Drobnym niuansem jest fakt, że w programie *Mathematica* znak mnożenia „$\star$" można zastąpić spacją (przez co użycie jest prostsze i bardziej zbliżone do obliczeń które wykonujemy na piśmie). Przykładowo

```
5*7 + 4/3 + (2 3 - 1/3)^3
```

daje w wyniku $\frac{5894}{27}$. Wartość numeryczna tego wyniku to około 218.296. Możemy to obliczyć wpisując polecenie

```
N[%].
```

Użyliśmy tutaj `N[...]`, czyli funkcji zwracającej wartość numeryczną liczby i symbolu `%`, który zastępuje ostatni obliczony wynik. Operację `%` możemy iterować, a dokładność numeryczną zwiększać[2], przykładowo

```
N[%%,10]
```

zwróci wartość 218.2962963.
Co ważne, programy typu CAS umożliwiają przeprowadzanie obliczeń symbolicznych na zmiennych ogólnych (do PS: czy to właściwa nazwa?). Wpiszmy

```
5*7 + 5 x + 3 y - (2 x + 3).
```

Mathematica sama uprości to wyrażenie do postaci $32 + 3x + 3y$. Przy okazji warto zaznaczyć, że rachunek symboliczny wymaga pewnej ostrożności. Porównajmy następujące dwie komendy

```
x+x 2
x+x2.
```

W pierwszym przypadku program zwrócił wynik $3x$ interpretując spację jako mnożenie. W drugim *Mathematica* potraktowała napis `x2` jako nazwę nowej zmiennej.
Gdy zajmujemy się bardziej skomplikowanymi wyrażeniami do uzyskania uproszczonej formy wyniku konieczne może być użycie polecenia `Simplify` lub `FullSimplify`. Przykładowo wyrażenie

```
5*7 + 5 x + 3 y - (2 x + 3)/3
```

nie redukuje się samoczynnie do prostszej postaci. Ale już

```
Simplify[%]
```

zwraca uporządkowany wynik $34 + \frac{13x}{3} + 3y$.
Mathematica obsługuje całą paletę funkcji elementarnych (`Exp`, `Log`, `Sin`, `Cos`, `Tan`, `Cot`, `Sqrt`, `ArcSin`, `Sinh`, itd.). Ich nazwy są albo, jak w wymienionych przykładach, takie same jak standardowo używane oznaczenia, albo też są opisem funkcji w jezyku angielskim (na przykład `Floor`, `Ceiling`, `IntegerPart` itp.). Podobnie jak z prostymi wyrażeniami symbolicznymi, również w przypadku funkcji, komputer potrafi sam, lub po zachęcie komendą `Simplify`, dokonać pewnych uproszczeń. Przykładowo,

[2]Domyślnie *Mathematica* podaje wynik z dokładnością do sześciu cyfr znaczących.

```
Exp[Sin[ArcCos[x]]] + Log[x^5] + Sin[2/3 Pi]
Simplify[Sin[x]^2-Cos[x]^2].
```

W powyższym przykładzie użyliśmy symbolu `Pi` na oznaczenie liczby π. Również inne ważne stałe matematyczne, takie jak liczba urojona i, nieskończoność ∞, liczba e, czy liczba Eulera γ, oznaczamy w programie *Mathematica* w sposób intuicyjny: `Pi`, `I`, `Infinity`, `E`, `EulerGamma`, itd.
Wracając do tematu redukcji wzorów do prostszej postaci, warto pamiętać że niektórych wyrażeń *Mathematica* nie uprości bez dodatkowych założeń. Najczęstszym powodem jest fakt, że dla programu domyślną dziedziną funkcji jest płaszczyzna zespolona $\mathbb{C}$, a nie prosta rzeczywista $\mathbb{R}$. Dodatkowe założenia możemy dodać na kilka równoważnych sposobów. Warto to prześledzić na przykładach:

```
Simplify[Sqrt[x^2], Element[x, Reals]]
Simplify[Sqrt[x^2], Assumptions -> {x < 0}]
Assuming[x > 0, Simplify[Log[x^5]]].
```

W palecie przydatnych narzędzi rachunkowych nie powinno zabraknąć poleceń: `Expand`, `Collect`, `Factor`, `Together` i `Apart`, które często pomagają zaoszczędzić nieprzyjemnych rachunków związanych z przekształcaniem wzorów. Pierwsze z omawianych poleceń służy do rozwijania wyrażenia poprzez wymnożenie nawiasów. Wpiszmy

```
Expand[(x + y)^3 + (2 x + 5 y)^2].
```

Dostaliśmy wynik w postaci sumy. Możemy go teraz w prosty sposób uporządkować względem potęg y:

```
Collect[%,y].
```

Komenda `Factor` jest odwrotnością `Expand` i służy do sprowadzania wyrażenia do postaci iloczynowej (o współczynnikach wymiernych). Na przykład `Factor[x^2 + 5 x + 4]` rozłoży dany wielomian na czynniki stopnia pierwszego. Ostatnie dwie spośród wymienionych wcześniej komend służą do przekształcania ułamków i funkcji wymiernych. `Together` pozwala sprowadzić wyrażenia do wspólnego mianownika, natomiast `Apart` szuka rozkładu na ułamki proste. Czytelnik może zaobserwować działanie tych funkcji wpisując kolejno polecenia

```
a = x + (x + 2)/((x^2 - 3) (x^2))
Together[a]
Apart[a].
```

Na koniec warto wspomnieć o definiowaniu własnych poleceń i oznaczeń. W poprzednim przykładzie przypisaliśmy zmiennej a wartość $x+\frac{(x+2)}{((x^2-3)(x^2))}$. Od tej pory a będzie stale przypisana taka właśnie wartość, w związku z czym, na przykład, polecenie

```
Simplify[x^2 (x^2-3) a]
```

zwróci jako wynik wielomian $2 + x - 3x^3 + x^5$. Warto pamiętać, że napis „`a=b`" nie jest dla programu *Mathematica* zdaniem logicznym, tylko operacją przypisania zmiennej a wartości b. Natomiast podwójny znak równości `a==b` program potraktuje jako pytanie o wartość zdania logicznego „$a = b$". Czytelnik może łatwo sprawdzić różnicę wpisując polecenia `2=3` oraz `2==3` i porównując odpowiedzi programu.
Przypiszmy teraz zmiennej x wartość 2. W konsekwencji zmianie ulegnie również zależna od x wartość zmiennej a:

```
x=2
a.
```

Przypisywanie zmiennym wartości, a następnie traktowanie ich jakby były zmiennymi wolnymi to jedna z najczęstszych przyczyn błędów. Przykładowo, jeżeli będziemy chcieli teraz wykonać pewną operację na funkcji zmiennej x, wynik może nie być tym czego oczekiwaliśmy. Procedura

```
Simplify[Sin[x]^2 - Cos[x]^2]
```

zamiast oczekiwanego $-\cos(2x)$ zwróci wynik obliczony w punkcie $x = 2$. Aby wyczyścić wartość przypisaną zmiennej używamy komendy `Clear[a]` (można też brutalnie zrestartować jądro programu poleceniem `Exit[]`). Dla uniknięcia błędów przydatne może być zapytanie `?a`, które pozwala sprawdzić jaki jest obecny status danego symbolu lub polecenia.
Mathematica pozwala też definiować użytkownikowi własne funkcje i procedury. Składnia takiej operacji wygląda następująco:

```
f[x_]:=x^3+2.
```

Zdefiniowaliśmy w ten sposób funkcję wielomianową $f(x) = x^3+2$. Dalej możemy swobodnie wykonywać operacje na tej funkcji posługując się oznaczeniem $f(x)$. Przykładowo wielkość `f[f[y]]` będzie teraz wielomianem dziewiątego stopnia zmiennej y, natomiast `f[1]` da w wyniku 3.

2 Granice ciągów i funkcji

2.1 Obliczanie prostych granic

Nowe umiejętności: obliczanie granic ciągów i funkcji, pętla for.

Skrypt: Rozdziały 2, 5. (podlinkować)

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`Limit[Sin[x],x->1]`	`limit(sin(x), x, 1)`	granica funkcji
`For[`*start*,*test*,*krok*,*ciało*`]`	`for` *zmienna*`:` *start* `while` *test* `do` *ciało*`; done`	pętla for

Programy typu CAS świetnie radzą sobie z obliczaniem granic funkcji elementarnych w danym punkcie. Dla przykładu, wykorzystując program *Mathematica* można szybko obliczyć granice

$$\lim_{x\to -1}\frac{x^4+3x^2-4}{x+1} \quad \text{oraz} \quad \lim_{x\to 0}\frac{\sqrt{x^2+1}-1}{\sqrt{x^2+25}}$$

wpisując

```
Limit[(x^4+3*x^2-4)/(x+1),x->-1]
Limit[(Sqrt[x^2+1]-1)/(Sqrt[x^2+25] - 5),x->0].
```

Komputer poradzi sobie również z bardziej skomplikowanymi przykładami

$$\lim_{x\to\infty}\left(\frac{3x-1}{3x+1}\right)^{2x-5}, \quad \lim_{x\to 1}(1-x)\operatorname{tg}(\tfrac{\pi x}{2}) \quad \text{lub} \quad \lim_{x\to\frac{\pi}{4}}\frac{\cos 2x}{\sin x-\cos x}\,.$$

Wystarczy wpisać

```
Limit[((3x - 1)(3x + 1))^(2x-5), x->Infinity]
Limit[(1-x)Tan[x*Pi/2], x->1]
Limit[Cos[2x]/(Sin[x] - Cos[x]), x->Pi/4].
```

Trzeba jednak uważać, gdyż czasem odpowiedź podawana przez programy CAS jest błędna. Przykładowo wpisując w *Mathematica*

```
Limit[Sqrt[1 - Cos[x]]/Sin[x] , x->0]
```

otrzymamy odpowiedź $\frac{1}{\sqrt{2}}$ podczas gdy powyższa granica nie istnieje[3]. Łatwo to sprawdzić obliczając granice jednostronne

```
f[x_]:=Sqrt[1 - Cos[x]]/Sin[x]
Limit[f[x], x->0, Direction -> 1]
Limit[f[x], x->0, Direction -> -1].
```

Zauważmy, że dla wygody wpowyższych obliczeniach zdefiniowaliśmy funkcję $f(x) := \frac{\sqrt{1-\cos(x)}}{\sin(x)}$.
Do tej pory obliczaliśmy granice funkcji elementarnych, z którymi komputery radzą sobie bardzo dobrze. O wiele gorzej sprawy się mają gdy przejdziemy do granic funkcji nieelementarnych. Spróbujmy obliczyć granicę

$$\lim_{x\to 0} x\lfloor\tfrac{1}{x}\rfloor\,.$$

Wpisujemy `Limit[x * Floor[1/x], x->0]`, lecz to nie daje żadnego rezultatu. *Mathematica* nie jest w stanie tego obliczyć, podczas gdy każdy student matematyki w mgnieniu oka poda poprawną odpowiedź.
Programy CAS nie radzą sobie też najlepiej z obliczaniem granic ciągów liczbowych, gdy parametr z którym przechodzimy do granicy przyjmuje tylko wartości naturalne. W *Mathematica* mamy do dyspozycji słówko `Assumptions`, które pomaga nam w takich sytuacjach

[3] Domyślnie *Mathematica* oblicza granicę prawostronną.

```
Limit[Sin[Pi*n], n->Infinity, Assumptions->Element[n,Integers]]
Limit[Cos[2*Pi*n], n->Infinity, Assumptions->Element[n,Integers]].
```

Jednak już poniższa granica nie zostanie obliczona prawidłowo

```
Limit[Cos[Pi*n], n->Infinity, Assumptions->Element[n,Integers]].
```

Zobaczmy teraz co się stanie, gdy poprosimy komputer o obliczenie granicy $\lim_{x\to+\infty}\sin(x^2)$, która nie istnieje

```
Limit[Sin[x^2], x -> Infinity].
```

W odpowiedzi *Mathematica* podała przedział `Interval[{-1, 1}]` do którego należą wartość rozważanej funkcji.
Dla pełni obrazu wspomnijmy jeszcze o jednym ciągu

$$a_n = n\sin(2\pi en!)\,.$$

Granicy a_n przy $n\to\infty$ *Mathematica* nie obliczy. Co więcej, nawet wypisywanie na komputerze dowolnie wielu kolejnych elementów tego ciągu nic nie pomaga. Można to zrobić używając pętli `For` oraz polecenia `N` zwracającego wartość numeryczną liczby:

```
For[n = 1, n<100, n++;Print[N[Sin[2 Pi E n!]]]] .
```

W wypisanym wyniku nie widać żadnych prawidłowości. W tym przykładzie błędy popełniane przez komputer przy obliczaniu wartości a_n są na tyle duże, że nie pomagają nam wyznaczyć granicy. Zainteresowany Czytelnik może sam spróbować obliczyć tę granicę wykorzystując fakt, że $e=\sum_{k=0}^{\infty}\frac{1}{k!}$ oraz to, że $\sin(2k\pi+x)=\sin(x)$ dla $k\in\mathbb{Z}$.

Problemy do samodzielnego rozwiązania:

Zadanie 2.1. Obliczyć następujące granice wspomagając się oprogramowaniem CAS

$$\lim_{n\to\infty}\frac{(n!)^{1/n}}{n}\,,\qquad \lim_{n\to\infty} n\sin(1/n)\,,\qquad \lim_{x\to0} x^{1/x}\,,$$
$$\lim_{x\to\infty} x^{1/x}\,,\qquad \lim_{n\to\infty}\frac{2^n n!}{n^n}\,,\qquad \lim_{n\to\infty}\frac{(2n)!}{(n!)^2}\,,$$
$$\lim_{n\to\infty}\left(\frac{(2n)!}{(n!)^2}\right)^{1/n}\,.$$

Zadanie 2.2. Obliczyć następjącą granicę wspomagając się oprogramowaniem CAS

$$\lim_{n\to\infty}\frac{\sum_{i=1}^{n} i!}{\sum_{i=1}^{n}(i+1)!}\,.$$

Zadanie 2.3. Obliczyć następjącą granicę wspomagając się oprogramowaniem CAS

$$\lim_{n\to\infty} n\sum_{i=1}^{n}\frac{1}{n^2+i}\,.$$

2.2 Ciągi rekurencyjne

Nowe umiejętności: badanie ciagów rekurencyjnych, rozwiązywanie równań.

Skrypt: Rozdział 2. (podlinkować)

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`Solve[f[x]==0,x]`	`solve(f(x),x)`	rozwiązywanie równań
`Reduce[f[x]>g[x],x]`	*brak*	rozwiązywanie nierówności
`Simplify[`*formuła*`]`	`radcan(`*formuła*`)`	upraszczanie wyrażenia

W poprzednim podrozdziale pokazaliśmy jak w programach typu CAS obliczać granice prostych funkcji oraz kiedy nie da się tego zrobić. W tym paragrafie zobaczymy jak wykorzystać komputer do badania ciągów zdefiniowanych rekurencyjnie. W tej sytuacji odpowiedni program może okazać się niezastąpiony gdyż obserwacja wystarczająco wielu początkowych elementów ciągu często pozwala nam wyrobić sobie właściwą intuicję i zgadnąć rozwiązanie.

Zadanie 2.4. Ciąg a_n o wyrazie początkowym $a_1 = \frac{1}{5}$ zadany jest rekurencją

$$a_{n+1} = \frac{1}{2}\left(a_n + \frac{5}{a_n}\right). \tag{1}$$

Zbadać zbieżność tego ciągu i obliczyć ewentualną granicę.

Rozwiązanie:

Ponieważ zależność (1) daje się zapisać w postaci $a_{n+1} = f(a_n)$ zacznijmy od zdefiniowania odpowiedniej funkcji

```
f[x_] := 1/2 (x + 5/x).
```

Dla nabrania intuicji możemy wypisać kilkanaście pierwszych wyrazów ciągu.

```
a[1] = 0.2;
For[i=1, i<15, i++, a[i+1]=f[a[i]]; Print[N[a[i]]]].
```

Jak widzimy wyniki szybko stabilizują się blisko liczby $2,23607$. Spodziewamy się zatem, że nasz ciąg będzie zbieżny do skończonej granicy $g \approx 2,23607$. Oczywiście jeśli takie g istnieje, to musi spełniać równanie[4] $g = f(g)$. By znaleźć g wpisujemy

```
Solve[f[x]==x, x].
```

Wynikiem są dwa rozwiązania $g = \pm\sqrt{5}$. Jest jasne, że startując z $a_1 > 0$ nigdy nie dostaniemy $a_n < 0$, a zatem możemy odrzucić ujemne rozwiązanie. Został nam jeden kandydat na granicę $g = \sqrt{5} \approx 2.23607$.

[4] Wynika to z ciągłości f.

Często owocnym pomysłem bywa badanie monotoniczność rozważanego ciągu[5]. Interesuje nas odpowiedź na pytanie kiedy $a_{n+1} = f(a_n) > a_n$?

```
Reduce[{x>0, x<f[x]}, x].
```

Okazuje się, że ma to miejsce dokładnie wtedy, gdy $a_n \in [0, \sqrt{5})$. Zbadajmy z kolei, kiedy ten warunek ma miejsce.

```
Reduce[{x>0, f[x]<Sqrt[5]}, x].
```

Wynikiem jest `False`, co oznacza, że dla wszystkich $x > 0$ zachodzi $f(x) \geq \sqrt{5}$ (Czytelnik zechce to udowodnić na bazie nierówności pomiędzy średnimi).
Podsumujmy nasze rozważania. Wiemy, że $f(x) \geq \sqrt{5}$ o ile tylko $x > 0$. Wobec tego wszystkie wyrazy ciągu a_n począwszy do $a_2 = f(a_1)$ będą nie mniejsze niż $\sqrt{5}$. Ponadto, jeżeli $x \geq \sqrt{5}$ to $f(x) \leq x$, a zatem począwszy od $n = 2$ będziemy mieli

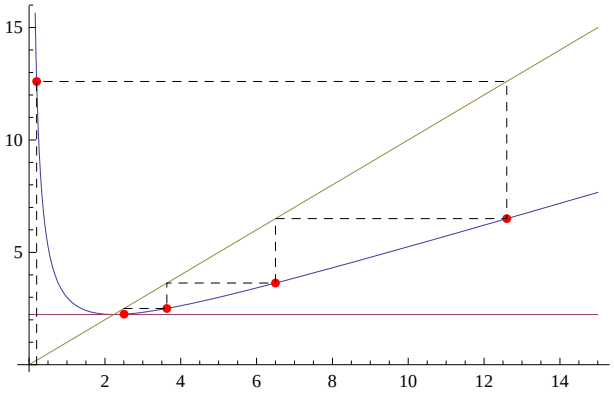
$$a_{n+1} = f(a_n) \leq a_n.$$

Ciąg a_n jest zatem nierosnący począwszy od $n = 2$ i jednocześnie ograniczony z dołu (na przykład przez 0, ale też przez $\sqrt{5}$), więc zbieżny. Jego granicą jest oczywiście $g = \sqrt{5}$ wyznaczone wcześniej.
Na koniec dwie uwagi. Ciekawym pomysłem bywa często zamiana badanego ciągu a_n na ciąg $a_n - g$ lub $\frac{a_n}{g}$, gdzie g jest kandydatem na granicę. W rozważanej sytuacji $b_n := \frac{a_n}{\sqrt{5}}$ spełnia rekurencję

$$b_n = \frac{1}{2}\left(b_n + \frac{1}{b_n}\right)$$

o analogicznych własnościach jak poprzednio, ale nieco prostszą do badania.
Często warto także zwizualizować sobie przebieg funkcji zadającej rekurencję na wykresie (więcej na ten temat w podrozdziale 3.1).

```
wykres = Plot[{f[x], Sqrt[5], x}, {x, 0, 15}].
```



[5]Ponownie obserwacja empiryczna może tutaj pomóc – spróbuj wypisać początkowe wyrazy ciągu z większą dokładnością numeryczną.

Wyznaczone wcześniej własności funkcji ($f(x) > \sqrt{5}$, $f(x) > x$ dla $x > \sqrt{5}$) dość łatwo zinterpretować na powyższym rysunku. Zaznaczyliśmy też tam dodatkowo punkty o współrzędnych $(a_n, a_{n+1} = f(a_n))$ dla kilku początkowych wyrazów ciągu. Układają się one na łamanej zbiegającej do punktu $(\sqrt{5}, \sqrt{5} = f(\sqrt{5}))$. Czy potrafisz zinterpretować tę łamaną?

Zadanie 2.5. Zbadać zbieżność ciągu

$$a_0 = a\,, \qquad a_{n+1} = a_n^2 - 4a_n + 5$$

w zależności od parametru a.

Rozwiązanie:

Mając już pewne obycie z podobnymi ciągami obserwujemy, że nasz ciąg ma postać $a_{n+1} = f(a_n)$, gdzie $f(x) = x^2 - 4x + 5$. Spodziewamy się, że ciąg ten będzie zbieżny do pewnego punktu stałego f, tzn. że granica g będzie spełniać $g = f(g)$. Oczywiście jeśli a_n jest zbieżny do skończonej granicy g, to musi zachodzić równość[6] $g = f(g)$.

```
Solve[x^2 - 4x + 5 == x, x].
```

Mamy dwa rozwiązania, czyli dwóch kandydatów na granicę

$$\frac{5-\sqrt{5}}{2} \simeq 1.38 \quad \text{oraz} \quad \frac{5+\sqrt{5}}{2} \simeq 3.61\,.$$

Oczekujemy, że ustalając wartość $a_0 = a$ pomiędzy tymi dwoma liczbami uzyskamy ciąg zbieżny do jednej z nich. Postępując standardowo musimy pokazać, że nasz ciąg jest ograniczony i monotoniczny. Żeby oszczędzić sobie trudu, wypiszmy kilka pierwszych wyrazów a_n dla $a_0 = 1,5$

```
a[0]=1.5
f[x_]:=x^2 - 4x + 5
For[i=1, i<15, i++, a[i]=f[a[i-1]]]
For[i=0, i<15, i++, Print[a[i]]].
```

W wyniku dostaniemy następujący ciąg liczb

```
1.5
1.25
1.5625
1.19141
1.65382
1.11984
1.77469
1.05077
1.90104
1.00979
1.98051
```

[6] Wynika to z ciągłości f.

```
1.00038
1.99924
1.
2.
```

Widać teraz, że nasz ciąg nie jest monotoniczny. Obserwując pierwsze 15 wyrazów możemy ponadto wyciągnąć wniosek, że ciąg a_n oscyluje pomiędzy wartością 1 i 2. Zmieniając wartość a_0 możemy obserwować co się dzieje jeśli startujemy z innego miejsca.

Zadanie 2.6. Obliczyć wartość pierwszych 20 wyrazów ciągu a_n dla

- $a_0 = \frac{1}{2}(5 + \sqrt{5}) - 0.01$
- $a_0 = \frac{1}{2}(5 + \sqrt{5}) + 0.01$
- $a_0 = \frac{1}{2}(5 - \sqrt{5}) - 0.01$
- $a_0 = \frac{1}{2}(5 - \sqrt{5}) + 0.01$
- $a_0 = \frac{1}{2}(5 - \sqrt{5}) - 1 - 0.01$
- $a_0 = \frac{1}{2}(5 - \sqrt{5}) - 1 + 0.01$.

Uwaga 1. Przy wypisywaniu wartości liczbowych warto skorzystać z funkcji `N[...]`.

W wyniku poczynionych obserwacji możemy pokusić się o następujące

Stwierdzenie.

- *Jeśli* $a_0 = \frac{1}{2}(5 \pm \sqrt{5})$*, to ciąg* a_n *jest stały.*
- *Jeśli* $a_0 \in \left(\frac{1}{2}(5 - \sqrt{5}) - 1, \frac{1}{2}(5 - \sqrt{5})\right) \cup \left(\frac{1}{2}(5 - \sqrt{5}), \frac{1}{2}(5 + \sqrt{5})\right)$*, to ciąg* a_n *ma dwa zbieżne podciągi* a_{2n} *oraz* a_{2n-1}*. Jeden z tych podciągów zbiega do* 1*, a drugi do* 2*.*
- *Jeśli* $a_0 < \frac{1}{2}(5 - \sqrt{5}) - 1$ *bądź* $a_0 > \frac{1}{2}(5 + \sqrt{5})$*, to ciąg* a_n *rozbiega do* $+\infty$*.*

Oczywiście nie mamy jeszcze dowodu, a jedynie empiryczne obserwacje. Ścisłe uzasadnienie powyższego stwierdzenia pozostawiamy Czytelnikowi. Przydatne mogą być następujące uwagi:

- Chcąc badać ciągi a_{2n} i a_{2n-1} warto wykonać następujące operacje

  ```
  f[x_]:=x^2 - 4x + 5
  Simplify[f[f[x]]]
  Solve[f[f[x]] == x, x].
  ```

  Pomoże nam to udowodnić monotoniczność podciągów a_{2n} i a_{2n-1}.

- Dla a_0 w przedziale $\left(\frac{1}{2}(5 - \sqrt{5}) - 1, 1\right)$ warto obliczyć a_1 i w ten sposób sprowadzić ten przypadek do wcześniej rozpatrzonych.

Problemy do samodzielnego rozwiązania:

Zadanie 2.7. Rozważmy ciąg Fibonacciego F_n zadany rekurencyjnie

$$F_{n+2} = F_{n+1} + F_n$$

dla wartości początkowych $F_1 = F_2 = 1$. Zbadać zbieżność ciągu $x_n := \frac{F_n}{F_{n+1}}$.
Zadanie 2.8. Zbadać zachowanie ciągu

$$a_0 = a\,, \qquad a_{n+1} = a_n^2 - 9a_n + 20$$

w zależności od parametru a.
Zadanie 2.9. Zbadać zachowanie ciągu

$$a_0 = a\,, \qquad a_{n+1} = -\tfrac{3}{2}a_n^2 + \tfrac{11}{2}a_n - 2$$

w zależności od parametru a.
Zadanie 2.10. Zbadać zachowanie ciągu

$$a_0 = a\,, \qquad a_{n+1} = \tfrac{3}{2}a_n^2 - \tfrac{13}{2}a_n + 8$$

w zależności od parametru a.
Zadanie 2.11. Zbadać zachowanie ciągu

$$a_0 = a\,, \qquad a_{n+1} = \tfrac{3}{2}a_n^2 - \tfrac{1}{2}a_n - 1$$

w zależności od parametru a.

2.3 Rekurencje liniowe

Nowe umiejętności: rozwiązywanie ogólnych rekurencji liniowych.
Skrypt: Rozdział 2. (podlinkować)

Poniższe zadanie posłuży nam jako pretekst do dość ogólnych rozważań i wyciągania wniosków na temat rekurencji liniowych.
Zadanie 2.12. Znaleźć wzór jawny na n-ty wyraz ciągu rekurencyjnego

$$a_{n+2} = 2a_{n+1} + 2a_n, \tag{2}$$

$$a_1 = 1 \quad a_2 = 4. \tag{3}$$

Rozwiązanie:
Jeśli nie dysponujemy lepszym pomysłem możemy zacząć od wypisania kilkunastu początkowych wyrazów ciągu. Być może patrząc na wyniki wpadnie nam coś do głowy.

```
a[1]=1, a[2]=4
For[i=2, i<15, i++, a[i+1] =2 a[i]+2 a[i-1];Print[a[i]]].
```

Podstawową obserwacją jaka się nasuwa jest to, że ciąg a_n szybko rośnie. Spróbujmy zbadać charakter tego wzrostu: czy ciąg zachowuje się podobnie do ciągu n^α, czy raczej jak ciąg geometryczny q^n? Ciekawego spostrzeżenia możemy dokonać badając na przykład ilorazy sąsiednich wyrazów:

```
For[i=2, i<15, i++, Print[N[a[i]/a[i-1]]]].
```

Okazuje się, że wartości tych ilorazów dość szybko stają się bliskie liczbie $2,73205$. Widzimy zatem, ze nasz ciąg zachowuje się podobnie do ciągu geometrycznego o przyroście $2,73205$. Postawmy zatem pytanie czy ciąg geometryczny $a_n = q^n$ może spełniać rekurencję postaci (2)? Wstawiając a_n w powyższej postaci do (2) dostaniemy równanie $q^{n+2} = 2q^{n+1} + q^n$. Po odrzuceniu mało interesującego przypadku $q = 0$ otrzymamy równanie kwadratowe[7]

$$q^2 = 2q + 2,$$

którego pierwiastkami są $q_1 = 1-\sqrt{3} \approx -0,73205$ oraz $q_2 = 1+\sqrt{3} \approx 2,73205$ (oczywiście to równanie także można rozwiązać za pomocą programu *Mathematica* używając funkcji `Solve`). Drugi ze znalezionych pierwiastków wygląda znajomo – to liczba wyznaczona doświadczalnie przy badaniu ilorazów sąsiednich wyrazów w ciągu! Zobaczmy zatem jak bardzo podobny jest ciąg a_n do ciągu geometrycznego $(1 + \sqrt{3})^n$?

```
For[i=2, i<15, i++, Print[N[a[i]/(1+Sqrt[3])^i]]].
```

Jak widzimy powyższe ilorazy robią się coraz bliższe liczbie $\frac{1}{2}$. Zbadajmy zatem jak zachowuje się ciąg $b_n := a_n - \frac{1}{2}(1 + \sqrt{3})^n$ powtarzając rozumowanie użyte do badania ciągu a_n.

```
b[n_]:=a[n]-1/2(1+Sqrt[3])^n
For[i=2, i<15, i++, Print[N[b[i]]]]
For[i=2, i<15, i++, Print[N[b[i]/b[i-1]]]].
```

Co ciekawe, ciąg b_n zachowuje się podobnie jak ciag geometryczny o postępie $-0.732051 \approx 1 - \sqrt{3}$! Sprawdźmy jak wygląda odpowiedni iloraz:

```
For[i=2, i<15, i++, Print[N[b[i]/(1-Sqrt[3])^i]]].
```

Z uzyskanych wyników można wnioskować, że $b_n = \frac{1}{2}(1 - \sqrt{3})^n$, skąd oczywiście

$$a_n = \frac{1}{2}(1 + \sqrt{3})^n + \frac{1}{2}(1 - \sqrt{3})^n. \tag{4}$$

Rzecz jasna przeprowadzone wyżej obliczenia nie są dowodem matematycznym, tylko procedurą pozwalającą wyrobić sobie pewne intuicje i znaleźć odpowiedniego kandydata na rozwiązanie. Czytelnik zechce samodzielnie przeprowadzić prosty dowód indukcyjny, że ciąg opisany wzorem (4) faktycznie spełnia warunki (2) i (3).

[7] Równanie to nazywamy *równaniem charakterystycznym* rekurencji (2).

Warto jednak przyjrzeć się postaci wyniku i pewnym elementom naszego rozumowania. To co pokazaliśmy ściśle to fakt, że ciągi q_1^n i q_2^n spełniają rekurencję (2). Natomiast nasze rozwiązanie które spełnia dodatkowo warunki początkowe (3) jest pewną kombinacją liniową $aq_1^n + bq_2^n$ tych szczególnych ciągów. Chwila zastanowienia wystarczy by wyjaśnić tę sytuację – ponieważ rekurencja (2) jest liniowa, jeżeli dwa ciągi a_n i $a_n^{'}$ spełniają ją to każda ich kombinacja liniowa też będzie ją spełniać. Wystarczy teraz znaleźć taką kombinację, która będzie spełniała warunki początkowe (3)!
Dysponując powyższą wiedzą możemy rozwiązać pokrewne problemy znacznie szybciej. Dla przykładu rozwiążmy rekurencję

$$a_{n+2} = 2a_{n+1} + 2a_n,$$
$$a_1 = 1 \quad a_2 = 2.$$

Wiemy już, że rozwiązania należy szukać w postaci $a_n = a(1+\sqrt{3})^n + b(1-\sqrt{3})^n$, a współczynniki a i b należy dobrać tak, żeby spełnione były warunki początkowe.

```
Solve[{a(1+Sqrt[3])+b(1-Sqrt[3])==1,
       a(1+Sqrt[3])^2+b(1-Sqrt[3])^2==2},{a,b}] .
```

Rozwiązaniem jest $a = \frac{3+\sqrt{3}}{6(1+\sqrt{3})}$ i $b = \frac{-1}{2\sqrt{3}}$, a więc

$$a_n = \frac{3+\sqrt{3}}{6(1+\sqrt{3})}(1+\sqrt{3})^n + \frac{-1}{2\sqrt{3}}(1-\sqrt{3})^n.$$

W istocie to co zrobiliśmy wyżej można zgrabnie uogólnić w następujący sposób.

Twierdzenie 1. *Załóżmy, że ciąg x_n spełnia rekurencję*

$$x_{n+2} = Ax_{n+1} + Bx_n$$

z warunkami początkowymi x_1 i x_2. Załóżmy, że równanie charakterystyczne

$$q^2 = Aq + B$$

ma dokładnie dwa różne pierwiastki rzeczywiste q_1 i q_2. Wówczas n-ty wyraz ciągu x_n dany jest wzorem

$$x_n = aq_1^n + bq_2^n,$$

gdzie liczby a i b są rozwiązaniami układu równań liniowych

$$aq_1 + bq_2 = x_1$$
$$aq_1^2 + bq_2^2 = x_2.$$

Problemy do samodzielnego rozwiązania:
Zadanie 2.13. Znaleźć wzór na n-ty wyraz ciągu Fibonacciego

$$F_{n+2} = F_{n+1} + F_n,$$
$$F_1 = F_2 = 1.$$

Korzystając z wyniku obliczyć granice ciągu $x_n := \frac{F_n}{F_{n+1}}$. Porównać wynik z wynikiem zadania z poprzedniego podrozdziału.

Zadanie 2.14. Rozwiązać rekurencję

$$b_{n+2} = 2b_{n+1} + 2b_n - 3,$$
$$b_1 = 2 \quad b_2 = 5.$$

Wskazówka: Rozważyć ciąg $a_n = b_n + c$ i dobrać c tak, aby pozbyć się wyrazu wolnego w zależności rekurencyjnej. Zaproponować ogólną metodę rozwiązywania rekurencji postaci

$$x_{n+2} = Ax_{n+1} + Bx_n + C.$$

Zadanie 2.15. Udowodnić twierdzenie 1.

Zadanie 2.16. Co się stanie, gdy w twierdzeniu 1 równanie charakterystyczne ma pierwiastek podwójny (na przykład dla rekurencji $a_{n+2} = 2a_{n+1} - a_n$)? Czy można wtedy zawsze rozwiązać równanie na współczynniki a i b? Dlaczego nie? Spróbuj znaleźć ogólną postać rozwiązania w takiej sytuacji. Co się dzieje gdy równanie charakterystyczne ma pierwiastki zespolone (na przykład dla rekurencji $b_{n+2} = -b_{n+1} - b_n$)?

Zadanie 2.17. Powtórzyć rachunki z rozwiązania zadania ze strony 12 wypisując więcej wyrazów ciągów a_n i b_n (powiedzmy 50). Zaobserwować, że ilorazy $\frac{b_n}{b_{n-1}}$ przestają być bliskie -0.732051 dla $n > 20$. Dzieje się tak dlatego, że liczby b_n obliczane są z pewną skończoną dokładnością. Ponieważ ciąg b_n zbiega do 0 (i to szybko), w pewnym momencie dokładność staje się porównywalna z rzeczywistymi wartościami b_n i drobne błędy w wartości b_n przekładają się na duże błędy w ilorazie. Zobacz, że numeryczne wartości ilorazu zmieniają znak z ujemnego na dodatni, co oczywiście nie powinno mieć miejsca. Warto pamiętać o problemach tego typu i nie ufać bezkrytycznie liczbom podawanym przez komputer.

Zadanie 2.18. *Mathematica* dysponuje procedurą `RSolve[`*równanie*,*ciąg*,*indeks*`]`, która służy do rozwiązywania rekurencji. Przykładowo, do obliczenia wzoru ogólnego na rekurencję rozważaną na początku tego rozdziału składnia powinna wyglądać następująco:

```
RSolve[{a[n+2] ==2 a[n+1]+2 a[n], a[1]==1, a[2]==4}, a[n], n].
```

Użyć powyższej funkcji do sprawdzenia, czy wyniki uzyskane w poprzednich zadaniach są prawidłowe. Spróbować użyć tej funkcji do wyliczenia ogólnych wzorów na rekurencje rozważane w poprzednim rozdziale.

2.4 Fraktale

Nowe umiejętności: wprowadzenie do teorii fraktali.

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`While[`*test*`,`*body*`]`	`for` *zmienna: start* `while` *test* `do` *body*`; done`	pętla while
`Table[`*expr*`,{`*min,max,krok*`}]`	`makelist(`*expr, zmienna, min, max*`)`	generowanie tabeli z wynikami operacji
`ArrayPlot[`*tabela*`]`	`Plot2d([discrete,`*lista*`],...)`	wykres na podstawie danych z tabeli/listy

Zbiory fraktalne (albo *fraktale*) to obiekty cechujące się pewnym samopodobieństwem. Jednym z najbardziej znanych przykładów takich zbiorów jest *krzywa trójkątowa Sierpińskiego* – będąca symbolem polskich Olimpiad Matematycznych. Teoria zbiorów fraktalnych zaczęła rozwijać się w latach 70tych w dużej mierze dzięki możliwościom obliczeniowym komputerów. W tym podrozdziale pokażemy jak tworzyć takie zbiory z użyciem oprogramowania CAS. Więcej informacji na temat fraktali znajdzie Czytelnik w monografii [3].
W podrozdziale 2.2 widzieliśmy jak badać zbieżność ciągów zadanych rekurencyjnie, tzn. zadanych w postaci

$$a_0 = a\,, \qquad a_n = f(a_{n-1})\,, \tag{5}$$

gdzie f jest pewną funkcją. Pracując z takimi ciągami można się zastanawiać czy jest jakaś ogólna reguła rządząca ich zachowaniem. Jednym z nasuwających się pytań jest: *jak zmienia się zachowanie ciągu, gdy zmieniamy wartość początkową* $a_0 = a$*?* Do tej pory badaliśmy ciągi o wartościach rzeczywistych ale najciekawsze rzeczy dzieją się gdy przejdziemy do ciągów o wartościach zespolonych.
Na początek ustalmy funkcję $f : \mathbb{C} \to \mathbb{C}$. Niech to będzie wielomian stopnia 2, np.

$$f(z) = z^2 + c\,.$$

Liczba $c \in \mathbb{C}$ jest parametrem, który będziemy zmieniać. Dla ustalenia uwagi połóżmy $c = 1 - \frac{1}{2}(1 + \sqrt{5})$. W programie *Mathematica* definiujemy powyższą funkcję następująco

```
c=1-0.5*(1+Sqrt[5])
f[z_]:=z^2+c .
```

Pytamy się: *dla jakich wartości* a_0 *ciąg rekurencyjny dany przez* (5) *będzie zbieżny?* W podrozdziale 2.2 widzieliśmy, że w ogólności takie ciągi nie muszą być zbieżne ale mogą oscylować wokół pewnych wartości. Mogliśmy się też przekonać, że przyjmując za a_0 różne wartości ciąg albo oscylował albo był rozbieżny do nieskończoności. W takim razie lepiej zmienić nasze pytanie na: *dla jakich wartości* a_0 *ciąg* a_n *jest ograniczony?* By na nie odpowiedzieć zastosujemy pewną heurystykę[8]. Obliczymy kilka (np. 10) pierwszych wyrazów ciągu a_n i uznamy, że jest

[8] *Heurystyka* w informatyce to metoda znajdowania rozwiązań, dla której nie ma gwarancji znalezienia rozwiązania optymalnego, a często nawet prawidłowego. Heurystyki używa się jej często, gdy algorytm wyliczania pełnego rozwiązania nie jest znany, albo jest zbyt kosztowny obliczeniowo.

on ograniczony jeśli moduł wszystkich dziesięciu kolejnych wyrazów jest mniejszy niż np. 10. Zdefiniujemy teraz w *Mathematica* funkcję, która będzie sprawdzać, czy dla danego $a_0 = a$ otrzymamy ciąg ograniczony

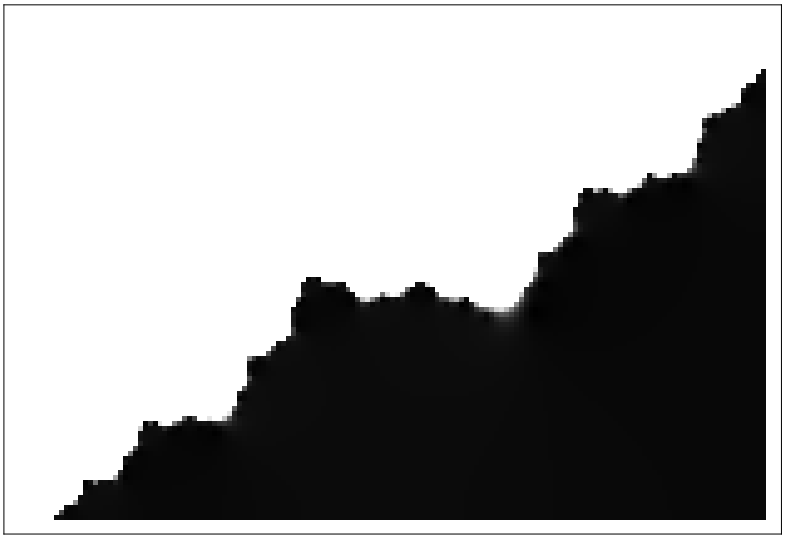
```
ogr[a_]:=(z=a; n=0;
          While[Abs[z]<10 && n<10, z=N[f[z]]; ++n];
          1-Min[{Abs[z],10}]/10) .
```

Przyjrzyjmy się powyższemu poleceniu. Definiujemy funkcję `ogr` zmiennej `a`. Funkcja ta wykorzystuje dodatkową zmienną `z`, która będzie przechowywać kolejne obliczone wartości ciągu oraz zmienną `n`, która kontroluje ilość iteracji pętli. Następnie wykonujemy pętlę, w której dokonujemy podstawienia `z = N[f[z]]` tak długo aż `z` będzie miało moduł większy od 10 lub aż wykonamy 10 powtórzeń. Na koniec funkcja `ogr` zwraca wartość `1 - Min[{Abs[z],10}] / 10`, czyli 1 minus minimum z modułu `z` i 10 podzielone przez 10. Taka definicja zapewnia, że `ogr[a]` zawsze zwróci wartość z przedziału $(0, 1)$. Wartość bliska 0 oznacza, że ciąg jest ograniczony, a wartość bliska 1 oznacza, że jest nieograniczony.
Żeby zobaczyć co się dzieje dla poszczególnych punktów na płaszczyźnie zespolonej wygenerujemy obrazek. W tym celu posłużymy się funkcjami programu *Mathematica* `ArrayPlot` oraz `Table`. Pierwsza z nich rysuje obrazek na podstawie danych przekazanych jako tablica, a druga generuje tablicę.

```
ArrayPlot[Table[ogr[x + I*y], {y,-1,0,0.01}, {x,-1.5,0,0.01}]] .
```

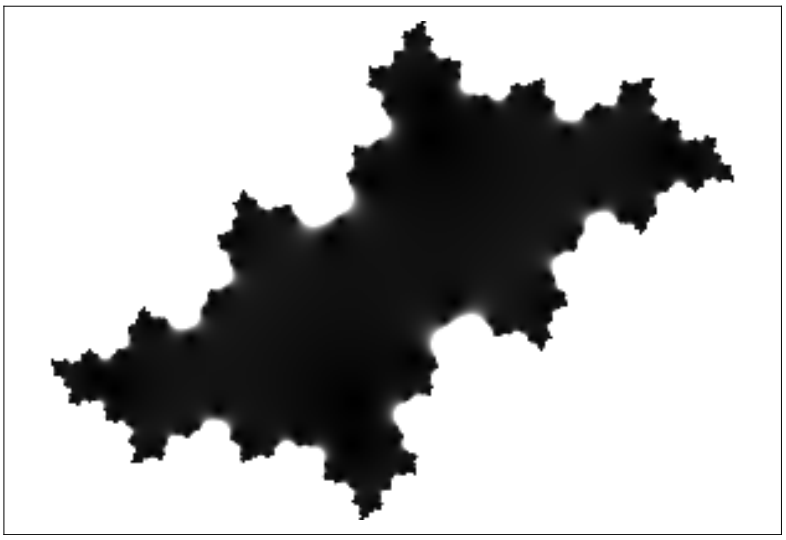
Polecenie `Table[ogr[x + I*y], {y,-1,0,0.01}, {x,-1.5,0,0.01}]` generuje tablicę dwuwymiarową zawierającą wartości funkcji `ogr[x + I*y]`. Parametry `{y,-1,0,0.01}` oraz `{x,-1.5,0,0.01}` wskazują w jakich przedziałach zmieniają się x i y. Tutaj wartości x zmieniają się w przedziale $[-1,5; 0]$, a wartości y w przedziale $[-1; 0]$. Liczba 0.01 mówi o ile zmienia się wartość x i y gdy przesuwamy się o jeden piksel na obrazku. Ten sposób generowania obrazków jest wysoce nieefektywny i zanim komputer skończy pracować może minąć kilka minut. Niemniej jednak po pewnym czasie na ekranie powinien pojawić się Rysunek 1. Brzeg zaciemnionego obszaru jest bardzo nieregularny. Zmieniając parametry `{y,-1,0,0.01}` i `{x,-1.5,0,0.01}` na `{y,-1,0,0.001}` oraz `{x,-1.5,0,0.001}` możemy zwiększyć dokładność (oczywiście kosztem czasu obliczeń). Zbiór widoczny na rysunku nazywa się, w naukowej nomenklaturze, *wypełnionym zbiorem Julii* wielomianu $f(z)$.
Zainteresowany Czytelnik może dowiedzieć się więcej na temat tych zbiorów wybierając się na wykład z Układów Dynamicznych lub zaglądając do monografii [3]. Tutaj sygnalizujemy jedynie, że temat jest ciekawy i wysoce nietrywialny.
Spróbujmy zmienić nieco parametry. Ustawmy $c = (\phi - 2) + (\phi - 1)i$, gdzie $\phi = \frac{1}{2}(1 + \sqrt{5})$ i wykonajmy nasz program od początku.

```
phi = 0.5*(1 + Sqrt[5])
c = (phi - 2) + (phi - 1)*I
f[z_]:=z^2 + c
ogr[a_]:=(z = a; n = 0;
          While[Abs[z] < 10 && n < 10, z = N[f[z]]; ++n];
```



Rysunek 1: Punkt $(0, 0)$ znajduje się w prawym dolnym rogu. Startując z punktów w ciemnym obszarze ciąg a_n pozostaje ograniczony.

```
          1 - Min[{Abs[z],10}] / 10)
ArrayPlot[Table[ogr[x + I*y], {y,-1,1,0.01}, {x,-1.5,1.5,0.01}]] .
```



Rysunek 2: Kolejny przykład fraktala.

Na ekranie powinien pojawić się Rysunek 2.

Czytelnik może poeksperymentować z innymi wartościami parametru c. Proponujemy wypró-

bować następujące wartości

$$c = -0,8 + 0,156i\,,$$
$$c = 0,285 + 0,01i\,,$$
$$c = -1,5\,,$$
$$c = i\,.$$

W tym kontekście wypada jeszcze wspomnieć znanym *zbiorze Mandelbrota*. Jest to podzbiór płaszczyzny zespolonej $\mathbb{C}$ zawierający te liczby $c \in \mathbb{C}$, dla których ciąg (5) z warunkiem początkowym $a_0 = 0$ pozostaje ograniczony.

Problemy do samodzielnego rozwiązania:
Zadanie 2.19. Narysować zbiory Julii dla wartości parametru c sugerowanych wyżej.
Zadanie 2.20. Poeksperymentować z pakietami `fractals` oraz `dynamics` w programie *Maxima*.
Zadanie 2.21. Przy pomocy funkcji `julia` z pakietu `dynamics` wygenerować obraz zbioru Julii dla wartości parametru c sugerowanych wyżej.
Zadanie 2.22. Za pomocą oprogramowania CAS narysować zbiór Mandelbrota.

3 Badanie funkcji rzeczywistych

3.1 Wykresy funkcji

Nowe umiejętności: rysowanie wykresów krzywych zadanych na różne sposoby.

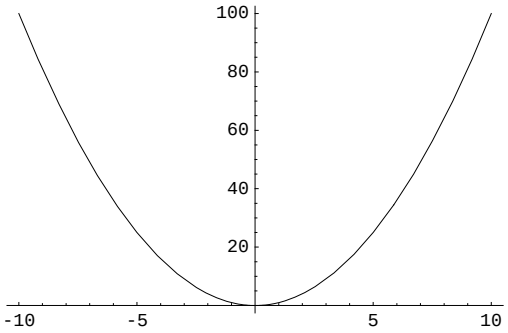
Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`Plot[`*funkcja*, *zakres*`]`	`plot2d`	wykres funkcji
`ParametricPlot`	`plot2d([parametric,...])`	wykres funkcji danej parametrycznie
`PolarPlot`	`plot2d([parametric,...])`	wykres funkcji danej we współrzędnych biegunowych
`ContourPlot`	`contour_plot`	wykres poziomic funkcji (funkcje uwikłane)
`Animate[`*wykres*, *param.*`]`	*brak*	dynamiczne wykresy

Graficzne przedstawienie obiektów matematycznych pozwala często lepiej zrozumieć ich naturę. W tej części zajmiemy się obrazowaniem krzywych.
Prosty wykres tworzymy za pomocą polecenia `Plot[`*funkcja*,*zakres*`]`. Przykładowo

```
Plot[x^2, {x, -10, 10}]
```

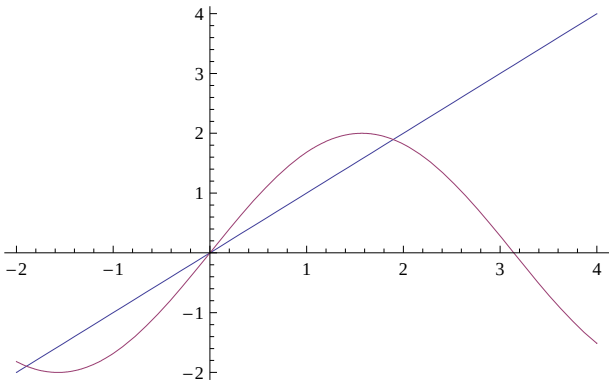
daje następujący efekt:



Na jednym obrazku możemy też umieścić kilka wykresów jak w poniższym przykładzie

```
Plot[{x, 2 Sin[x]}, {x, -2, 4}].
```

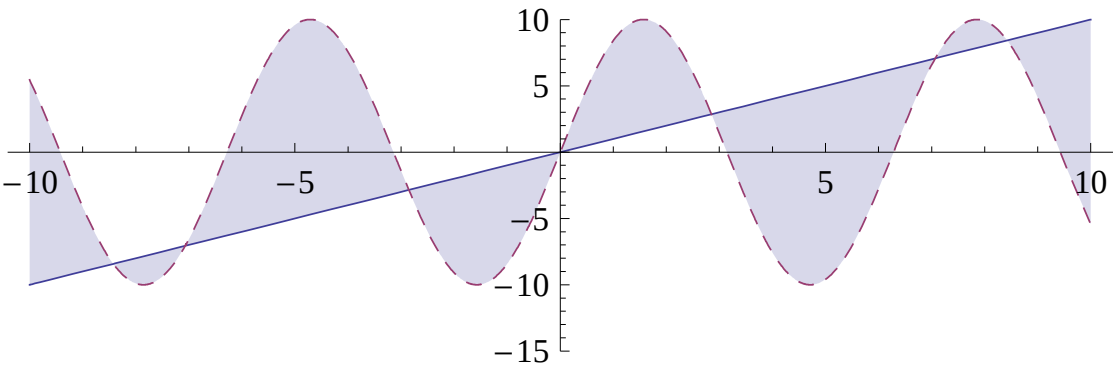
Wynik wygląda następująco:



Ten sam efekt uzyskalibyśmy także następującą sekwencją komend:

```
wyk1=Plot[x,{x,-2,4}];
wyk2=Plot[2 Sin[x],{x,-2,4}];
Show[wyk1,wyk2].
```

Jeśli zależy nam na bardziej wyrafinowanym wyglądzie możemy kazać *Mathematice* zmienić podstawowe parametry wykresu. Wymieńmy kilka ważniejszych: `PlotRange` określa zakres wartości jaki ma być pokazany, `AspectRatio` definuje stosunek wysokości do szerokości rysunku, `Filling` definiuje kolorowanie, natomiast `PlotStyle` określa styl formatowania wykresu. Szczegółowe informacje, w tym także omówienie wielu dodatkowych opcji wraz z przykładami, można znaleźć w obszernej dokumentacji programu *Mathematica* [5].
Przykładowo taki efekt



Uzyskamy za pomocą komendy

```
Plot[{x, 10 Sin[x]}, {x, -10, 10}, PlotRange -> {-15, 10},
AspectRatio -> 0.3, Filling -> {1 -> {2}},
PlotStyle -> {Automatic, Dashed}].
```

W praktycznych zastosowaniach nie zawsze mamy do czynienia z wykresami funkcji opisywalnych prostym wzorem. Czasami chcemy narysować krzywą daną w postaci parametrycznej, albo uwikłanej. *Mathematica* dysponuje odpowiednimi narzędziami aby poradzić sobie również w takiej sytuacji. Omówimy teraz kilka prostych przykładów.

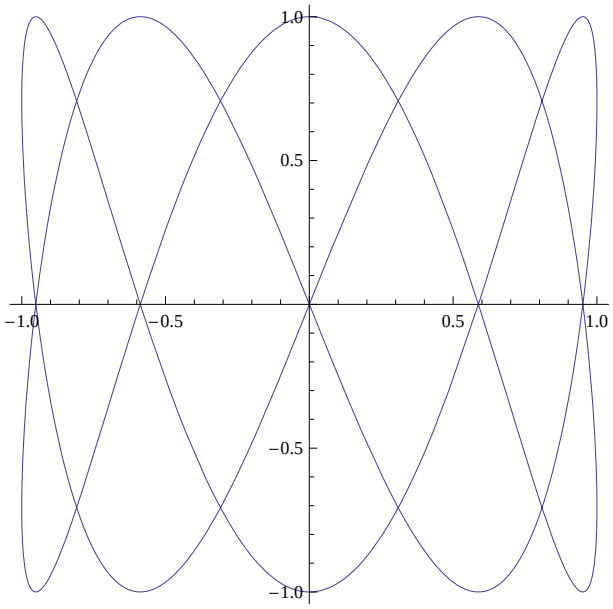
Krzywą w postaci parametrycznej nazywamy zbiór opisany następująco

$$C = \{(x(t), y(t)) : t \in [a, b]\},$$

gdzie x i y są pewnymi funkcjami zmiennej t. Do opisu tego typu zbiorów *Mathematica* dysponuje funkcją `ParametricPlot`. Przykładowo

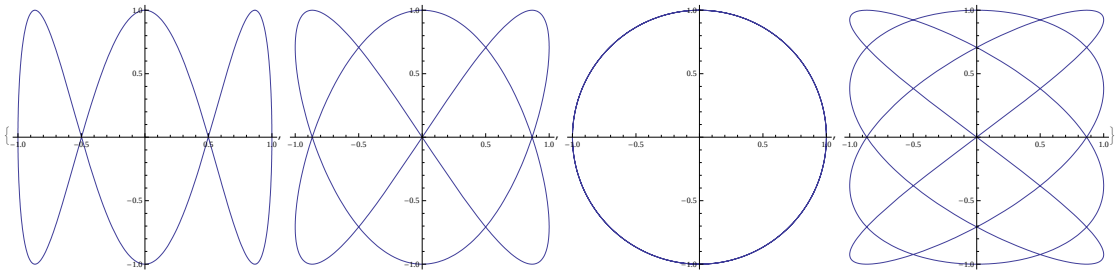
```
ParametricPlot[{Sin[2 t], Cos[5 t]}, {t, 0, 2 Pi}]
```

i końcowy efekt:



Powyższa krzywa to szczególny przypadek tak zwanych *krzywych Lissajous*, które pojawiają się w naturalny sposób w opisie drgań harmonicznych. Ogólnie zbiory takie mają postać $x(\phi) = A\sin(a\phi)$ i $y(\phi) = B\cos(b\phi + \phi_0)$. Odpowiada to nałożeniu na siebie dwóch drgań o różnych częstościach a i b, różnych amplitudach A i B oraz przesunięciu w fazie o $\frac{\pi}{2} + \phi_0$. Krzywą zamkniętą uzyskujemy wtedy i tylko wtedy, gdy stosunek częstości $\frac{a}{b}$ jest liczbą wymierną. Porównajmy kilka krzywych tego typu

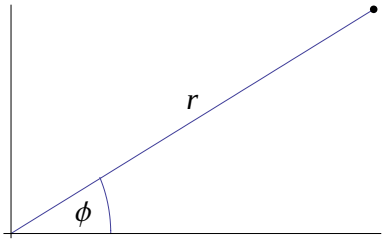
```
Table[ParametricPlot[{Sin[k t], Cos[3 t]}, {t, 0, 2 Pi}], {k,1,4,1}] .
```



Użyliśmy do tego celu polecenia `Table`. Sekwencja `{k,1,4,1}` określa zakres jaki przebiega parametr k. Są to klejne liczby od 1 do 4 rosnące co 1.
Przy okazji warto wspomnieć o funkcji `Animate`. Czytelnik zechce wpisać komendę

```
Animate[ParametricPlot[{Sin[k t], Cos[3 t]}, {t, 0, 4 Pi}], {k, 1, 4}].
```

Za pomocą suwaka możemy teraz zmieniać wartość parametru k na przedziale $[1, 4]$ i obserwować jak reaguje na to wykres krzywej.
Innym sposobem opisu krzywej jest postać biegunowa $r = r(\phi)$, gdzie r jest odległością danego punktu od środka układu współrzędnych, natomiast ϕ to kąt pod jakim dany punkt widzimy względem osi OX.

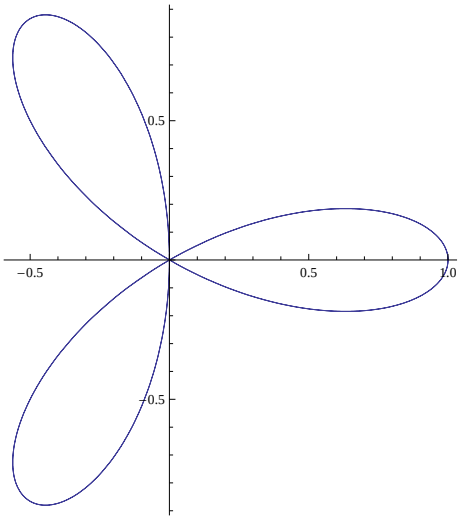


Postać biegunowa jest specjalną formą postaci parametrycznej, gdzie parametryzujemy za pomocą kąta ϕ:

$$x(\phi) = r(\phi)\cos(\phi),$$
$$y(\phi) = r(\phi)\sin(\phi).$$

Najprostszym przykładem krzywej tego typu jest okrąg. Odpowiednie równanie to oczywiście $r = \text{const}$.
Krzywą zadaną w postaci biegunowej rysujemy za pomocą polecenia `PolarPlot`. Jako przykład użycia rozważmy *trójlistnik*:

```
PolarPlot[ Cos[3 Phi], {Phi, 0, 2 Pi}] .
```

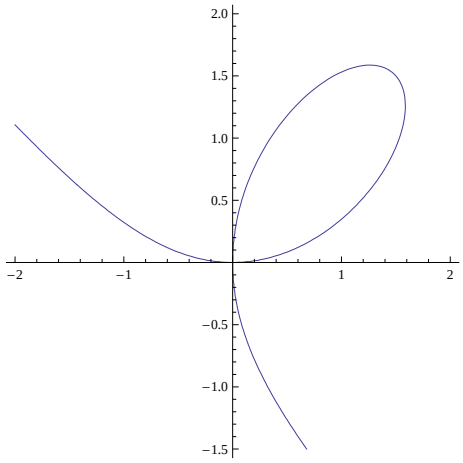


Na koniec zobaczmy jak można wykreślać krzywe zadane w sposób uwikłany jako rozwiązania pewnego równania

$$f(x, y) = c.$$

Służy do tego funkcja `ContourPlot`. Z jej pomocą możemy narysować na przykład krzywą nazywaną *liściem Kartezjusza*:

```
ContourPlot[y^3 + x^3 - 3 x y == 0, {x, -2, 2}, {y, -1.5, 2},
 Axes->True, Frame->False]
```



Problemy do samodzielnego rozwiązania:

Zadanie 3.1. Narysować wykres *cykloidy* (krzywej jaką zakreśla punkt na kole toczącym się po prostej), danej w postaci parametrycznej

$$x(t) = a(t - \lambda \sin(t)), \quad y(t) = a(1 - \lambda \cos(t)).$$

Wielkość a odpowiada promieniowi toczącego się koła, natomiast $\lambda \cdot a$ oznacza odległość rozważanego punktu od środka koła. Dla $\lambda > 1$ mówimy o cykloidzie wydłużonej a dla $\lambda < 1$ o skróconej.

Zadanie 3.2. Znaleźć postać parametryczną i narysować wykres *epicykloidy*, czyli krzywej jaką zakreśla punkt koła toczącego się po zewnętrznej krawędzi innego koła. Zaobserwować, że krzywą zamkniętą otrzymamy tylko wtedy, gdy stosunek promieni obu kół jest liczbą wymierną. Czy potrafisz to udowodnić?

Zadanie 3.3. Narysować *ślimak Pascala* dany w postaci biegunowej równaniem $r = a\cos(\phi) + l$. Ślimak Pascala dla parametrów $a = l$ nazywamy *kardioidą*. Wykazać, że kardioida to epicykloida powstała przez toczenie punktu na brzegu koła o promieniu a po zewnętrznej krawędzi koła o takim samym promieniu.

Zadanie 3.4. Narysować *krzywą stożkową* zadaną biegunowo zależnością $r = \frac{p}{1+e\cos(\phi)}$, gdzie $e > -1$ i $p > 0$ są parametrami. Zbadać jak zmienia się charakter krzywych w zależności od parametru e. Spróbuj inaczej opisać te krzywe dla $e = -1$, $e \in (-1, 0)$ i $e = 0$.

Zadanie 3.5. Przedstawić liść Kartezjusza w postaci parametrycznej. Wykazać, że prosta $x + y + 1 = 0$ jest asymptotą tej krzywej.

3.2 Przebieg zmienności funkcji

Nowe umiejętności: różniczkowanie funkcji, badanie ich przebiegu, wyznaczanie ekstremów, punktów przegięcia, asymptot, itp..

Skrypt: Rozdziały 5 i 6. (podlinkować)

Mathematica	*Maxima*	Komentarz
`D[f[x],x]`	`diff(f(x),x)`	pochodna funkcji
`Factor[`*expr*`]`	`factor(`*expr*`)`	rozkład na czynniki
`Select[`*lista,pred*`]`	`lreduce(lambda([l,e],` `if` *pred* `then cons(e,l)` `else l),` *lista* `,[]);`	wybiera z listy te elementy, które spełniają predykat *pred*

Przy badaniu przebiegu zmienności funkcji, oprogramowanie typu CAS okazuje się bardzo przydatne. Dla sporej klasy funkcji $f : \mathbb{R} \to \mathbb{R}$ możemy automatycznie obliczyć pochodne, znaleźć pierwiastki równań $f(x) = 0$, czy $f'(x) = 0$, a nawet znaleźć ekstrema lokalne i obliczyć granice w nieskończoności. Wymienione operacje możemy wykonać zarówno symbolicznie jak i numerycznie. Niezmiernie przydatne okazuje się często naszkicowanie wykresu rozpatrywanej funkcji. Przy tym wszystkim trzeba jednak pamiętać, że programy typu CAS to tylko narzędzia, z których trzeba inteligentnie korzystać. Należy mieć na uwadze, że obliczenia na komputerze są zawsze obarczone pewnym błędem, wynikającym z ograniczonej dokładności rachunków. W szczególności wykresy funkcji rysowane na ekranie monitora należy zawsze traktować jako pewne przybliżenie, które choć zazwyczaj pozwala budować prawidłową intuicję, czasem może być zwodnicze. Zacznijmy od prostego przykładu

Zadanie 3.6. Zbadać przebieg zmienności funkcji danej wzorem

$$f(x) = \frac{64 - 120x + 70x^2 - 15x^3 + x^4}{-54 + 45x - 12x^2 + x^3}\,.$$

Rozwiązanie:
Dla ustalenia uwagi przypomnijmy co należy zrobić:

- znaleźć maksymalny podzbiór $\mathbb{R}$, na którym można określić f powyższym wzorem,
- znaleźć zera f,
- znaleźć granice w krańcach przedziałów określoności,
- znaleźć zbiór, w którym f jest różniczkowalna,
- znaleźć przedziały monotoniczności i ekstrema lokalne
- znaleźć przedziały wypukłości i wklęsłości oraz punkty przegięcia,
- znaleźć asymptoty jeśli istnieją,
- naszkicować wykres.

Funckcja f jest ilorazem dwóch wielomianów, więc maksymalną dziedziną funkcji będzie zbiór tych x, dla których mianownik się nie zeruje. Należy zatem znaleźć pierwiastki równania

$$-54 + 45x - 12x^2 + x^3 = 0.$$

Wystarczy wykonać polecenie:

```
Solve[-54 + 45x - 12x^2 + x^3 == 0, x].
```

Dowiemy się, że są dwa rozwiązania $x = 3$ oraz $x = 6$, przy czym $x = 3$ jest podwójne. Maksymalną dziedziną naszej funkcji jest zatem zbiór

$$\mathrm{Dom}_f = (-\infty, 3) \cup (3, 6) \cup (6, \infty).$$

Do znalezienia miejsc zerowych naszej funkcji możemy posłużyć się poleceniem `Factor`

```
Factor[64 - 120x + 70x^2 - 15x^3 + x^4].
```

Dowiemy się stąd, że $f(x) = 0$ wtedy i tylko wtedy, gdy $x = 1$, $x = 2$, $x = 4$ lub $x = 8$.
Do obliczenia granic na krańcach przedziałów określoności posłuży nam znane już polecenie `Limit`[9]

```
Limit[f[x],x -> -Infinity]
Limit[f[x],x -> 3, Direction -> 1]
Limit[f[x],x -> 3, Direction -> -1]
Limit[f[x],x -> 6, Direction -> 1]
Limit[f[x],x -> 6, Direction -> -1]
Limit[f[x],x -> Infinity].
```

Stąd mamy

$$\lim_{x\to-\infty} f(x) = -\infty \qquad \lim_{x\to3^-} f(x) = -\infty \qquad \lim_{x\to3^+} f(x) = -\infty$$
$$\lim_{x\to6^-} f(x) = +\infty \qquad \lim_{x\to6^+} f(x) = -\infty \qquad \lim_{x\to\infty} f(x) = +\infty\,.$$

Dla wyznaczenia punktów różniczkowalności f należy obliczyć pochodną i zobaczyć gdzie jest dobrze określona. W *Mathematice* wyliczanie pochodnej funkcji jednej zmiennej jest bardzo proste. Wystarczy napisać

`f'[x]` albo `D[f[x],x]`

Wynik może nie być zbyt czytelny, dlatego skorzystamy z funkcji `Simplify`

```
fp[x_] := Simplify[f'[x]]
fp[x]
```

i dowiemy się, że pochodna jest opisana wzorem

$$f'(x) = \frac{-1200 + 1608x - 780x^2 + 182x^3 - 21x^4 + x^5}{(-6+x)^2(-3+x)^3}\,.$$

Od razu widać, że pochodna istnieje wszędzie poza punktami $x = 6$ oraz $x = 3$, czyli f jest różniczkowalna w całej swojej dziedzinie.
Ponieważ f jest różniczkowalna wszędzie gdzie jest określona, więc do zbadania monotoniczności możemy posłużyć się pochodną. Szukamy miejsc zerowych pochodnej

[9] Uwaga! Przypomnijmy, że parametr `Direction -> 1` oznacza granicę lewostronną, a `Direction -> -1` granicę prawostronną.

```
Solve[fp[x] == 0, x]
```

co niestety nie daje satysfakcjonującego rezultatu. W naszym przypadku pierwiastki równania $f'(x) = 0$ nie wyrażają się żadnym ładnym wzorem. Pozostaje nam zatem operować na numerycznych przybliżeniach. Piszemy

```
N[Solve[fp[x] == 0, x]]
```

i dowiadujemy się, że jest dokładnie jeden pierwiastek rzeczywisty, równy w przybliżeniu $x_0 = 1,62937$. Możemy go wyłuskać z listy wszystkich rozwiązań przy pomocy polecenia

```
x0 = Select[x /. N[Solve[fp[x] == 0, x]],Element[#,Reals]&][[1]].
```

Musimy zbadać znak pochodnej na przedziałach $(-\infty, x_0)$, $(x_0, 3)$, $(3, 6)$ oraz $(6, \infty)$. Wiemy, że f' jest ciągła w swojej dziedzinie i zmienia znak tylko w x_0, wystarczy więc wybrać po jednym punkcie z każdego z przedziałów $(-\infty, x_0)$, $(x_0, 3)$, $(3, 6)$ oraz $(6, \infty)$ i sprawdzić jaki znak ma f' w każdym z tych punktów. Weźmy punkty 1, 2, 5 i 7. Wykonujemy polecenia

```
fp[1] > 0
fp[2] > 0
fp[5] > 0
fp[7] > 0
```

i uzyskujemy odpowiedzi

```
True
False
True
True.
```

W takim razie f jest rosnąca na przedziale $(-\infty, x_0)$, ma maksimum lokalne w x_0, a dalej maleje do $-\infty$ zbliżając się do $x = 3$. Następnie, na przedziale $(3, 6)$, rośnie od $-\infty$ do $+\infty$ przecinając oś OX w punkcie 4. Podobnie na przedziale $(6, \infty)$ rośnie od $-\infty$ do $+\infty$ przecinając oś OX w punkcie 8.
W następnej kolejności badamy wypukłość i wklęsłość. Obliczymy w tym celu drugą pochodną jeśli istnieje. Wykonujemy

```
fb[x_] := Simplify[f''[x]]
fb[x]
```

i natychmiast dostajemy odpowiedź

$$f''(x) = \frac{144 + 2040x - 1284x^2 + 282x^3 - 22x^4}{(-6+x)^3(-3+x)^4}\,.$$

Widzimy, że druga pochodna jest dobrze określona wszędzie poza punktami 3 i 6, więc nasza funkcja jest dwukrotnie różniczkowalna w całej swojej dziedzinie i możemy posłużyć się drugą pochodną do wyznaczania przedziałów wypukłości i wklęsłości. Jak poprzednio wykonujemy

```
{x1,x2} = Select[x /. N[Solve[fb[x] == 0, x]],Element[#,Reals]&]
```

i znajdujemy w ten sposób dwa rzeczywiste miejsca zerowe drugiej pochodnej równe w przybliżeniu $x_1 = -0,0676635$ oraz $x_2 = 4,49721$. Polecenia

```
fb[-1] > 0
fb[1] > 0
fb[4] > 0
fb[5] > 0
fb[7] > 0
```

dają odpowiedzi

```
True
False
False
True
False
```

więc nasza funkcja jest wypukła na przedziale $(-\infty, x_1)$, wklęsła na $(x_1, 3) \cup (3, x_2)$, ponownie wypukła na $(x_2, 6)$ i wklęsła na $(6, \infty)$. Ponieważ pierwiastki równania $f''(x) = 0$ były jednokrotne (każdy występował tylko raz na liście rozwiązań), więc od razu możemy wnioskować, że punkty przegięcia znajdują się w x_1 i x_2.
Wiemy już o istnieniu dwóch asymptot pionowych w punktach $x = 3$ i $x = 6$. Pozostaje sprawdzić, czy f ma asymptoty w $\pm\infty$. Obliczamy w tym celu granice

```
Limit[f[x]/x, x -> -Infinity]
Limit[f[x]/x, x -> Infinity]
```

i dowiadujemy się, że obie istnieją[10] i są równe 1. Dalej wykonujemy

```
Limit[f[x] - x, x -> -Infinity]
Limit[f[x] - x, x -> Infinity]
```

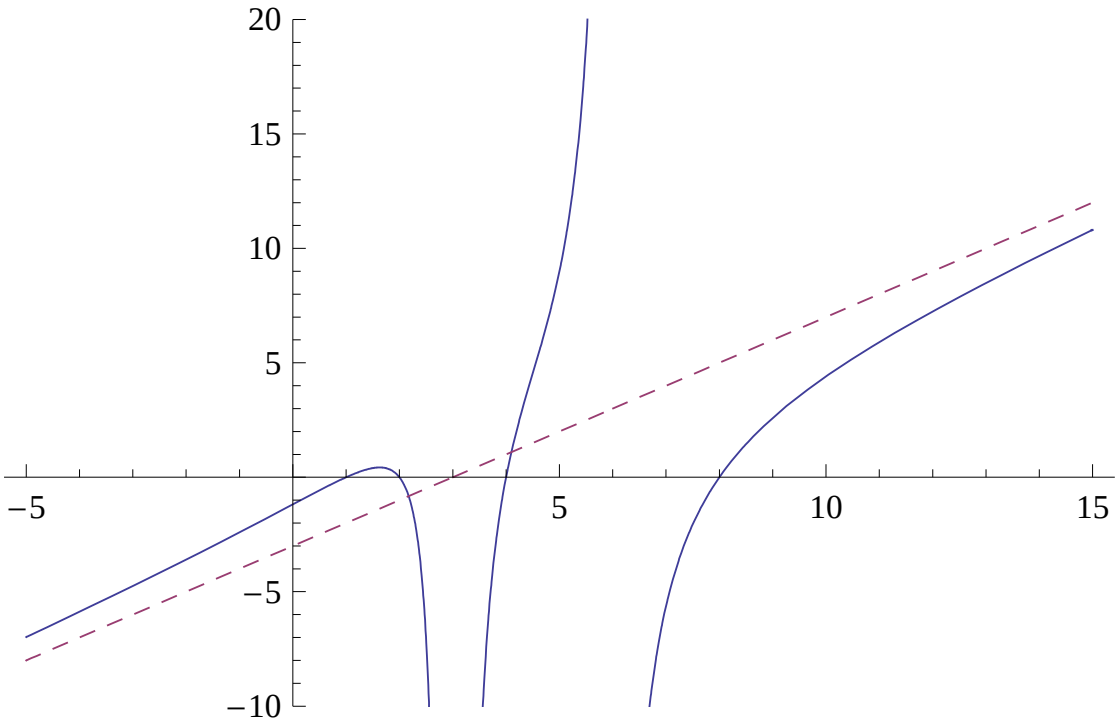
uzyskując w obu przypadkach odpowiedź -3. W takim razie obie asymptoty istnieją i są opisane równaniem

$$y = x - 3\,.$$

Na koniec naszkicujemy wykres badanej funkcji. Zaznaczymy na nim również obliczoną asymptotę $y = x - 3$. W *Mathematice* wystarczy napisać

```
Plot[{f[x], x - 3}, {x, -5, 15}, PlotRange -> {-10, 20},
     PlotStyle -> {Automatic, Dashed}, Exclusions -> {3, 6}]
```

[10]Uwaga! Przypominamy, że z istnienia granicy $\lim_{x\to\infty} f(x)/x$ nie wynika jeszcze istnienie asymptoty, np. $f(x) = sin(x)$.



Rysunek 3: Wykres funkcji $f(x)$ wraz z asymptotą.

by uzyskać szkic wykresu (zobacz Rysunek 3). Dodatkowe parametry zostały wprowadzone po to aby poprawić czytelność rysunku. Drobnym mankamentem jest to, że z wykresu trudno byłoby wnioskować o istnieniu punktu przegięcia w x_1. Poza tymi drobiazgami[11], wykres jaki uzyskaliśmy dobrze odpowiada temu co wiemy o funkcji f.

Problemy do samodzielnego rozwiązania:

Zadanie 3.7. Zbadać przebieg zmienności funkcji danej wzorem

$$f(x) = \frac{18x^3 + 39x^2 - 51x + 12}{3x^2 + 7x - 6}\,.$$

Zadanie 3.8. Zbadać przebieg zmienności funkcji danej wzorem

$$f(x) = \sqrt{|x|}(x^2 + x - 2)\,.$$

Zadanie 3.9. Zbadać przebieg zmienności funkcji danej wzorem

$$f(x) = e^x(x - 5)(x - 2)(x^2 + x + 1)\,.$$

Uwaga 1: Ze względu na ograniczoną dokładność obliczeń, niektóre wyliczone przez komputer wartości numeryczne mogą mieć małą część urojoną podczas gdy powinny być rzeczywiste.

[11] Aby zaobserwować punkt przegięcia można narysować mniejszy fragment wykresu:
`Plot[f[x], {x, 3, 6}, PlotRange ->{-20, 30}]`.

Trzeba mieć to na uwadze interpretując podawane wyniki. W takim przypadku pomocna może być funkcja `Chop`. Zaobserwuj jej działania na następującym przykładzie:

```
N[Sqrt[5] Coth[2^10 ArcCoth[1/Sqrt[5]]]]
Chop[%].
```

Uwaga 2: Przy szkicowaniu wykresu należy umiejętnie posłużyć się parametrami `PlotRange` oraz `AspectRatio`, by na wykonanym rysunku można było zaobserwować ekstrema lokalne i zmiany w monotoniczności.

Zadanie 3.10. Zbadać przebieg zmienności funkcji danej wzorem

$$f(x) = |x-1|^{1/3}(2-|x-2|^{3/2})(-|x-3|^{3/4})x(x-4)\,.$$

Wskazówka: W *Mathematice*, zamiast używać standardowej funkcji `Abs`, która jest zbyt ogólna[12], warto napisać własną funkcję obliczającą wartość bezwzględną z liczby rzeczywistej

```
abs[x_] := If[x < 0, -x, x].
```

Wtedy *Mathematica* nie będzie już miała problemów z upraszczaniem wyrażeń. Czytelnik sam może sprawdzić jaka jest różnica wykonując polecenia

```
Simplify[abs[x-1]*abs[x-2], 1 < x < 2]
Simplify[Abs[x-1]*Abs[x-2], 1 < x < 2].
```

Zadanie 3.11. Dla funkcji z poprzedniego zadania wykonać polecenia

```
Minimize[{f[x], 0 < x < 1}, x]
Maximize[{f[x], 0 < x < 1}, x].
```

Wykonać te same polecenia dla funkcji f zdefiniowanej przy pomocy standardowej funkcji `Abs`. Jaka jest różnica?

Uwaga: Wykonanie powyższych komend może zająć kilka minut.

Zadanie 3.12. Zbadać przebieg zmienności funkcji danej przez

$$f(x) = \begin{cases} \exp(-x^{-2})\exp(-(x-1)^{-2}) & \text{jeśli } x \in (-\infty,0)\cup(0,1)\cup(1,\infty), \\ 0 & \text{jeśli } x \in \{0,1\}\,. \end{cases}$$

3.3 Aproksymacja funkcji ciągłych wielomianami

Nowe umiejętności: przybliżanie funkcji wielomianami.

Skrypt: Rozdział 7.3. (podlinkować)

Nowe funkcje:

[12] Funkcja `Abs` jest zdefiniowana jako zespolony moduł.

Mathematica	*Maxima*	Komentarz
`InterpolatingPolynomial`	`lagrange` (wymaga załadowania biblioteki `interpol`)	przybliżenie wielomianowe

---

Na wykładach z analizy matematycznej Czytelnik dowiedział się, że każdą funkcję ciągłą na przedziale domkniętym można przybliżać wielomianami w sposób jednostajny (Twierdzenia 7.15 i 7.16). W tym celu posługiwaliśmy się tak zwanymi *wielomianami Bernsteina*. Dla funkcji ciągłej $f : [0,1] \to \mathbb{R}$ definiujemy przybliżający ją ciąg wielomianów

$$B_n(f)(x) = \sum_{k=0}^{n} \binom{n}{k} f\left(\tfrac{k}{n}\right) x^k (1-x)^{n-k}\,.$$

Powyższe wyrażenie jest dość zawiłe. Również dowód, że ciąg $B_n(f)$ jest zbieżny jednostajnie do f jest dość skomplikowany. Czytelnik może zastanawiać się czy nie można tego zrobić jakoś prościej. Narzucającym się pomysłem jest tzw. interpolacja przez punkty wykresu.
Wiadomo, że wielomian stopnia k jest wyznaczony jednoznacznie przez swoje wartości w $k+1$ różnych punktach. Istotnie, weźmy $k+1$ punktów $x_0, \ldots, x_k$ i ustalmy $k+1$ wartości $y_0, \ldots, y_k$. Każdy wielomian stopnia k ma $k+1$ współczynników i może być zapisany jako $W(x) = a_0 + a_1 x + \ldots + a_k x^k$. Istnieje dokładnie jeden wielomian $W(x)$ spełniający zależność $W(x_i) = y_i$ dla wszystkich $i = 0, \ldots, k$. Każde z $k+1$ równań $W(x_i) = y_i$ jest równaniem liniowym względem współczynników $a_0, \ldots, a_k$. Istnienie rozwiązania wynika teraz z tego, że odpowiedni wyznacznik Vandermonde'a jest niezerowy o ile tylko $x_i \neq x_j$ dla $i \neq j$.
W programach CAS istnieją gotowe funkcje, które potrafią znaleźć wielomian o zadanych wartościach w zadanych punktach. Przykładowo, wykonując w programie *Mathematica* polecenie

```
InterpolatingPolynomial[{{0,0},{1,-1},{2,0}},x]
```

dostajemy odpowiedź

```
(-2+x)x .
```

Łatwo sprawdzić, że jest to poprawna odpowiedź – wykres wielomianu $W(x) = x(x-2)$ rzeczywiście przechodzi przez punkty $(0,0)$, $(1,-1)$ i $(2,0)$. W *Maximie* ten sam efekt uzyskamy wpisując

```
load(interpol);
lagrange([[0,0],[1,-1],[2,0]]) .
```

Problemy do samodzielnego rozwiązania:
Zadanie 3.13. Napisać w *Mathematica* procedurę `intpoly[f_,n_]`, która przyjmuje dwa argumenty: pewną funkcję $f : [0,1] \to \mathbb{R}$ oraz liczbę naturalną n, a w wyniku daje odpowiedni wielomian interpolacyjny stopnia n, którego wykres przechodzi przez punkty $(\frac{k}{n}, f(\frac{k}{n}))$ dla każdego $k = 0, \ldots, n$.
Wskazówka: Warto posłużyć się tutaj poleceniem `Table`, by wygenerować automatycznie długą listę postaci `{{0,f[0]},{1/n,f[1/n]},...,{1,f[1]}}`.

Zadanie 3.14. Za pomocą zdefiniowanej wyżej procedury `intpoly` wygenerować wielomian stopnia n dla wymienionych niżej funkcji. Korzystając z polecenia `Plot` stworzyć wykres funkcji f oraz odpowiedniego wielomianu interpolacyjnego. Czy wielomian dobrze przybliża daną funkcję? Czy jakość przybliżenia staje się coraz lepsza wraz ze wzrostem n?

- $f(x) = \sin(\pi x)$ dla $n = 3, 5, 10, 20$,
- $f(x) = \sin(3\pi x)$ dla $n = 3, 5, 10, 20$,
- $f(x) = (x + 0.01)^{-1}$ dla $n = 2, 5, 10, 20, 30$,
- $f(x) = |x - 0.5|$ dla $n = 2, 3, 4, 5, 10, 20, 30$,
- $f(x) = |x - 0.5|^{3/2}$ dla $n = 2, 3, 4, 5, 10, 20, 30$,
- $f(x) = \exp(1 - 1/x^2)$ dla $n = 2, 3, 4, 5, 10, 20$.

Zadanie 3.15. Napisać w *Mathematica* procedurę `bernstein[f_, n_]`, która przyjmuje dwa argumenty: pewną funkcję $f : [0, 1] \to \mathbb{R}$ oraz liczbę naturalną n, a w wyniku daje wielomian Bernsteina stopnia n dla funkcji f.

Zadanie 3.16. Za pomocą zdefiniowanej wyżej procedury `bernstein` wygenerować wielomian stopnia n dla wymienionych niżej funkcji. Korzystając z polecenia `Plot` stworzyć wykres funkcji f oraz odpowiedniego wielomianu Bernsteina. Czy wielomian dobrze przybliża daną funkcję? Czy jakość przybliżenia staje się coraz lepsza wraz ze wzrostem n?

- $f(x) = \sin(\pi x)$ dla $n = 3, 5, 10, 20$,
- $f(x) = \sin(3\pi x)$ dla $n = 3, 5, 10, 20$,
- $f(x) = (x + 0.01)^{-1}$ dla $n = 2, 5, 10, 20, 30$,
- $f(x) = |x - 0.5|$ dla $n = 2, 3, 4, 5, 10, 20, 30$,
- $f(x) = |x - 0.5|^{3/2}$ dla $n = 2, 3, 4, 5, 10, 20, 30$,
- $f(x) = \exp(1 - 1/x^2)$ dla $n = 2, 3, 4, 5, 10, 20$.

Uwaga! Zauważ, że kolejne wielomiany Bernsteina nie muszą pokrywać się z przybliżaną funkcją w coraz większej liczbie punktów. Inaczej ma się sprawa z wielomianami interpolacyjnymi, które z definicji pokrywają się z przybliżaną funkcją na coraz większym zbiorze. Na tym właśnie polega różnica między *interpolacją*, a *aproksymacją*. Funkcje aproksymujące zadaną funkcję są blisko tejże funkcji (w odpowiednio dobranym sensie) ale mogą się z nią nie pokrywać w żadnym punkcie.

3.4 Znajdowanie miejsc zerowych funkcji ciągłych

Nowe umiejętności: implementacja standardowych metod obliczania miejsc zerowych funkcji ciągłych.

Skrypt: Rozdziały 5.3, A.3. (podlinkować)

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`FindRoot[f[x],{x,`*start*`}]`	`find_root(f(x),x,`*start,end*`)`	numeryczne przybliżanie pierwiastka równania

Z wykładu analizy (Twirdzenie 5.39) wiemy, że każda funkcja ciągła ma tzw. *własność Darboux*: jeśli w punkcie x mamy $f(x) = a$, a w punkcie y mamy $f(y) = b$ oraz $a < b$, to dla każdej liczby c spełniającej $a < c < b$ znajdziemy punkt z leżący pomiędzy x i y taki, że $f(z) = c$. Innymi słowy, jeśli funkcja ciągła przyjmuje w jakiś punktach wartości a i b, to musi też przyjąć każdą wartość pośrednią. Obserwacja ta pozawala przybliżać miejsca zerowe dowolnej funkcji ciągłej. Jeśli tylko potrafimy znaleźć dwa punkty x i y takie, że $f(x) > 0$ i jednocześnie $f(y) < 0$, to gdzieś pomiędzy x i y musi być taki punkt z, że $f(z) = 0$. Możemy teraz zbliżyć się do liczby z wybierając pewien punkt w leżący pomiędzy x i y. Jeśli $f(w) > 0$, to punkt z musi leżeć pomiędzy w i y, a jeśli $f(w) < 0$, to z leży pomiędzy x i w. W każdym z tych dwóch przypadków zmniejszyliśmy długość przedziału, w którym szukamy punktu z. Powtarzając tę procedurę możemy zbliżyć się do z na dowolnie małą odległość.

W zależności od tego w jaki sposób będziemy wybierać punkt w, nasza metoda może zbiegać do z szybciej lub wolniej. Jeśli za każdym razem punkt w wybierzemy w środku przedziału, tzn. przyjmiemy

$$w = \tfrac{1}{2}(x + y)\,,$$

to rozpatrywany przedział będzie się kurczył w każdym kroku dwukrotnie, czyli geometrycznie. Metoda ta nazywa się metodą *bisekcji* i jest całkiem szybko zbieżna, ale istnieją metody od niej lepsze.

Przykładowo, możemy poprowadzić *sieczną* wykresu funkcji przez punkty $(x, f(x))$ i $(y, f(y))$ i ustalić punkt w w miejscu przecięcia się tej siecznej z osią OX. Daje to następujący wzór

$$w = \frac{xf(y) - yf(x)}{f(y) - f(x)}\,.$$

Jeśli funkcja f jest nie tylko ciągła ale też różniczkowalna, to możemy postępować inaczej. Zaczynając z punktu x_0 (o którym powinniśmy wiedzieć a priori, że jest blisko miejsca zerowego), wyznaczamy styczną do wykresu przechodzącą przez punkt $(x_0, f(x_0))$, a następnie ustalamy punkt x_1 w miejscu przecięcia się tej stycznej z osią OX. Dalej powtarzamy tę operację wyzna-

czając kolejne punkty $x_2, x_3, \ldots$. Procedura ta definiuje następujący ciąg rekurencyjny

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}\,.$$

Oczywiście możemy tak robić tylko jeśli pochodna $f'(x_n)$ jest różna od zera, czyli wtedy gdy styczna do wykresu nie jest równoległa do osi OX. Metoda ta nazywa się *metodą Newtona*.
O wielu innych metodach znajdowania miejsc zerowych funkcji ciągłych można poczytać na stronach Wolfram MathWorld [6]. Czytelnik dowie się też o nich więcej z wykładu z Matematyki Obliczeniowej lub Metod Numerycznych prowadzonych na naszym wydziale. Polecamy też zajrzeć do podręczników [1, 4, 8].
W programie *Mathematica* mamy do dyspozycji funkcję `FindRoot`, która przybliża miejsca zerowe danej funkcji metodą Newtona. Można też użyć metody siecznych przekazując dodatkowy parametr postaci `Method -> "Secant"`.
Zadanie 3.17. Znaleźć (w przybliżeniu) wszystkie pierwiastki równania $2^x = x^2$.
Łatwo zobaczyć, że nasze równanie spełnione jest dla $x = 2$ oraz $x = 4$. Czytelnik zapewne potrafi udowodnić, że nie ma więcej dodatnich rozwiązań. W punkcie $x = 0$ mamy $2^0 = 1$ oraz $0^2 = 0$, więc lewa strona jest większa ale już dla $x < -1$ mamy $2^x < x^2$, więc pomiędzy -1 i 0 musi być jeszcze jeden pierwiastek. Pomocny może być w tym momencie wykres. Definiujemy funkcję

```
f[x_] := x^2 - 2^x
```

a następnie wykonujemy

```
FindRoot[f[x], {x,0}]
FindRoot[f[x], {x,-1,0}, Method -> "Secant"] .
```

W odpowiedzi uzyskujemy

```
{x -> -0.766665}
{x -> -0.766665} .
```

By dostać dokładniejszy wynik możemy posłużyć się opcjami `WorkingPrecision` oraz `AccuracyGoal`

```
FindRoot[f[x], {x,0}, WorkingPrecision -> 20, AccuracyGoal -> 20]

{x -> -0.76666469596212309311} .
```

Co ciekawe metoda siecznych zaimplementowana w funkcji `FindRoot` nie zawsze daje wynik z przedziału podanego na początku. By się o tym przekonać wystarczy wykonać

```
FindRoot[f[x], {x,-1,1.9}, Method -> "Secant"] .
```

Problemy do samodzielnego rozwiązania:
Zadanie 3.18. Obliczyć liczbę π z dokładnością do 20 miejsc po przecinku wykorzystując

funkcję `FindRoot`. Wynik można porównać z wartością numeryczną używaną przez program *Mathematica* `N[Pi,20]`.

Zadanie 3.19. Znaleźć (w przybliżeniu) wszystkie rozwiązania równania

$$(x-1)(x+1) = \tfrac{1}{5}\sin(10x)\,.$$

Zadanie 3.20. Znaleźć (w przybliżeniu) wszystkie rozwiązania równania

$$\left(x-\tfrac{1}{2}\right)(x+1) = \tfrac{1}{2}\sin(13x)\,.$$

Zadanie 3.21. Znaleźć (w przybliżeniu) wszystkie rozwiązania równania

$$\operatorname{arctg}(50\sin(x)) = 2\cos(x).$$

Zadanie 3.22. Napisać funkcję `odw[f_,s_]`, która przyjmie dwa argumenty: różniczkowalną i różnowartościową funkcję f oraz argument s, a zwraca wartość $f^{-1}(s)$.
Wskazówka: $f^{-1}(s) = t \quad \Longleftrightarrow \quad f(t) = s$.

Zadanie 3.23. Przy pomocy funkcji `odw` zdefiniować funkcję odwrotną do funkcji $\sinh(x) = \frac{e^x - e^{-x}}{2}$. Narysować wykres tak zdefiniowanej funkcji i porównać z wykresem funkcji $\operatorname{arcsinh}$.

Zadanie 3.24. Punkty zbiory $A \in \mathbb{R}^2$ spełniają równanie $(y-x)^2 = x^5$. Wykorzystując `FindRoot` zdefiniować funkcję $y(x)$ parametryzującą zbiór A w otoczeniu punktu $(1,0)$. Narysować wykres. Porównać z wykresem uzyskanym za pomocą funkcji `ContourPlot` (por. rozdział 3.1)

Zadanie 3.25. Dana jest funkcja dwóch zmiennych $F(x,y) = \exp(xy)(x+y)$. Wykorzystując `FindRoot` zdefiniować funkcję $y(x)$ parametryzującą poziomicę $F^{-1}(1)$ w otoczeniu punktu $(0,1)$. Narysować wykres.

4 Ciągi i szeregi funkcyjne

4.1 Szeregi liczbowe

Nowe umiejętności:

Skrypt: Rozdział 4. (podlinkować)

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`Sum[`*wyr*, *zakres*`]`	`sum(`*wyr*,*zmienna*, *od*,*do*`)`	suma
`NSum[`*wyr*, *zakres*`]`	*brak*	sumowanie numeryczne

Rozdział poświęcony badaniu ciągów i szeregów funkcyjnych zaczniemy od krótkiego omówienia możliwości programu *Mathematica* w zakresie obróbki sum skończonych i szeregów liczbowych.

Podstawową komendą służącą do obliczania sum jest `Sum[`*wyrażenie*, *zakres*`]`. Zacznijmy od rozwiązania słynnego zadania Gaussa[13] znalezienia sumy wszystkich liczb naturalnych od 1 do 100:

```
Sum[i, {i, 1, 100}].
```

To samo zadanie możemy także rozwiązać od razu w większej ogólności:

```
Sum[i, {i, 1, n}],
```

podobnie jak inne zadania podobnego typu, na przykład problem obliczenia sumy kolejnych piątych potęg $1 + 2^5 + 3^5 + \ldots + n^5$:

```
Sum[i^5, {i, 1, n}].
```

Mathematica poradzi sobie także doskonale z innymi problemami. Bez trudu pozwoli nam na przykład wyznaczyć sumy

$$\sum_{i=2}^{n} \frac{1}{(4i-1)(4i+3)}, \qquad \sum_{1}^{n} i^2 q^i, \qquad \text{czy} \qquad \sum_{k=1}^{n} \sin(kx).$$

Wystarczy wpisać

```
Sum[1/((4 i - 1) (4 i + 3)), {i, 2, n}]
Sum[i^2 q^i, {i, 1, n}]
Sum[Sin[k x], {k, 1, n}]
```

i odczytać odpowiedź.
Uwaga! W niektórych przypadkach wynik może zastać wyrażony za pomocą funkcji specjalnych.
Całkiem dobrze poradzimy sobie również z sumami zawierającymi symbole Newtona, na przykład $\sum_{k=1}^{n} k^2 \binom{n}{k}$ obliczymy bez trudu pisząc

```
Sum[k^2 Binomial[n, k], {k, 1, n}].
```

Jednak już w przypadku bardziej skomplikowanej sumy $\sum_{k=1}^{n} \binom{n}{k} \cdot \binom{n}{n-k}$ wynik zostanie zapisany w terminach funkcji Γ, zamiast w prostszej formie $\binom{2n}{n} + 1$ (Czytelnik może spróbować samodzielnie udowodnić tę równość).
Mathematica pozwala także na obliczanie sum szeregów. W tym celu wystarczy użyć omówionego wcześniej polecenia `Sum` ustawiając zakres sumowania na nieskończoność (`Infinity`). Jako ilustrację obliczymy kilka prostych sum:

$$\sum_{n=1}^{+\infty} n^2 q^n, \qquad \sum_{n=1}^{+\infty} \frac{1}{n^4}, \qquad \text{oraz} \qquad \sum_{n=1}^{+\infty} \frac{\sin(n)}{n}.$$

Odpowiednie polecenia to:

[13] Podobno problem ten mały Gauss musiał rozwiązać za karę.

```
Sum[n^2 q^n, {n, 1, Infinity}]
Sum[1/n^4, {n, 1, Infinity}]
Sum[Sin[n ]/n, {n, 1, Infinity}].
```

Zobaczmy jak *Mathematica* zareaguje w przypadku szeregu harmonicznego $\sum \frac{1}{n}$

```
Sum[1/n, {n, 1, Infinity}].
```

Odpowiedzią jest komunikat, że podana suma jest rozbieżna.
Przy bardziej skomplikowanych szeregach *Mathematica* zazwyczaj nie potrafi podać sumy, co zresztą nie powinno nas bardzo dziwić jako że, poza paroma prostymi przypadkami, my także nie potrafimy tego zazwyczaj zrobić. W takim wypadku program zwróci po prostu wyjściową formułę. Dotyczy to zarówno szeregów zbieżnych jak i rozbieżnych, przykładowo

```
Sum[Sin[1/n^2], {n, 1, Infinity}]
Sum[1/(n Log[n]), {n, 2, Infinity}].
```

W takiej sytuacji pomocne może być polecenie sumowania numerycznego `NSum`. W większości przykładów z którymi będziemy mieli do czynienia na ćwiczeniach z analizy program zwróci skończoną wartość w przypadku szeregu zbieżnego i informację o błędzie w przypadku szeregu rozbieżnego. W tym ostatnim przypadku również uzyskamy wynik liczbowy – będzie to liczba, którą program uzyskał w momencie zastopowania algorytmu sumującego. Dwa poprzednio rozpatrywane szeregi dobrze oddają tę prawidłowość

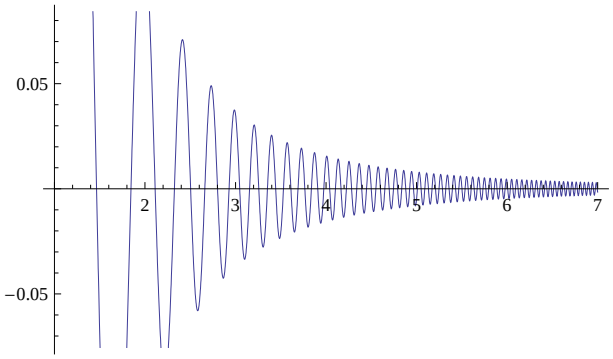
```
NSum[Sin[1/n^2], {n, 1, Infinity}]
NSum[1/(n Log[n]), {n, 2, Infinity}].
```

Chociaż odpowiedzi programu w większości przypadków pokrywają się z tym co wiemy o zbieżności rozpatrywanych szeregów, należy je w najlepszym razie traktować jako wskazówki, a nie jako dowody zbieżności bądź rozbieżności badanych szeregów. Bardzo wiele zależy od tego jak opisane są składniki naszego szeregu. Generalnie jeśli są to liczby postaci $f(n)$, gdzie f jest dość regularną funkcją (na przykład monotoniczną) wówczas możemy oczekiwać, że odpowiedzi programu będą prawidłowe. Ale w przypadku funkcji gorzej uwarunkowanych numerycznie program może nie być w stanie obliczyć ewidentnie zbieżnej sumy. Dotyczy to szczególnie szeregów postaci $\sum f(n)$, gdzie f jest funkcją „szybko oscylującą". Dobrym przykładem może być szereg $\sum \frac{\sin(n^3)}{n^3}$. Zobaczmy jak będzie wyglądać odpowiedź programu na zadanie obliczenia jego sumy:

```
NSum[Sin[n^3]/n^3, {n, 1, Infinity}]
```

i jak wygląda „szybko oscylująca" funkcja $\frac{\sin(x^3)}{x^3}$:

```
Plot[Sin[x^3]/x^3, {x, 1, 7}]
```



Podsumowując, do obliczeń programu *Mathematica* dotyczących szeregów należy podchodzić z ograniczonym zaufaniem. Nie znaczy to oczywiście, że nie da się użyć komputera aby zbadać zbieżność konkretnego szeregu. Przykłady takiego postępowania podamy w kolejnym podrozdziale.

4.2 Szereg Taylora, szeregi potęgowe

Nowe umiejętności: Rozwijanie funkcji w szereg Taylora, operacje na szeregach potęgowych.

Skrypt: Rozdziały 6.4, 8. (podlinkować)

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`Series[f[x], {x, x0, 10}]`	`taylor(f(x), x, x0, 10)`	rozwija funkcję f w szereg wokół `x0` do wyrazów zadanego rzędu
`Normal[`*szereg*`]`	`taytorat(`*szereg*`)`	ucina resztę szeregu (wyrazy $o(x^n)$)
`SeriesCoefficient`	*brak*	podaje pojedynczy współczynnik rozwinięcia
`InverseSeries`	`revert`	rozwinięcie funkcji odwrotnej
`ComposeSeries`	*brak*	składanie szeregów

Mathematica dysponuje bardzo przydatnym poleceniem `Series` służącym do znajdowania rozwinięcia danej funkcji w szereg potęgowy. Jego działanie prześledźmy na prostym przykładzie – rozwińmy funkcję wykładniczą do wyrazów 10 rzędu:

```
Series[Exp[x], {x, 0, 10}].
```

Polecenie `Series` pozwala na znalezienie rozwinięcia danej funkcji tylko do wyrazów ustalonego skończonego rzędu, a więc zastąpienie liczby 10 w powyższym poleceniu wielkością `Infinity` albo `k` nie da żadnych rezultatów. Czy nie ma zatem sposobu na znalezienie wyrazu ogólnego rozwinięcia funkcji $\exp(x)$? Okazuje się, że takim narzędziem jest polecenie `SeriesCoefficient`, które pozwala znaleźć interesujący nas wyraz ustalonego rzędu rozwinięcia danej funkcji. Przykładowo

```
SeriesCoefficient[Exp[x], {x, 0, k}]
```

zwróci współczynnik $\frac{1}{k!}$. Zobaczmy jeszcze jaki będzie efekt, gdy zapytamy się o 5-ty współczynnik rozwinięcia wokół 1 niesprecyzowanej a priori funkcji $f(x)$:

```
SeriesCoefficient[f[x], {x, 1, 5}].
```

W wyniku otrzymaliśmy $\frac{1}{120}f^{(5)}(1)$, a więc ogólny wzór na 5-ty współczynnik rozwinięcia Taylora funkcji $f(x)$.
Mathematica pozwala wykonywać wiele operacji na szeregach potęgowych, takich jak ich mnożenie, składanie, czy znajdowanie rozwinięcia funkcji odwrotnej. Zobaczmy to na kilku prostych przykładach. Na początek pomnóżmy przez siebie dwa szeregi: szereg $x + 2x^2 - 5x^4 + o(x^4)$ i rozwinięcie $\sin(x)$ wokół 0 do wyrazów rzędu 10:

```
(x + 2 x^2 - 5 x^4 + O[x]^5) Series[Sin[x], {x, 0, 10}].
```

Program zwrócił wynik $x^2 + 2x^3 - \frac{x^4}{4} - \frac{16x^5}{3} + o(x^6)$, a więc automatycznie zwinął wszystkie wyrazy wyższych rzędów do wyrazu $o(x^6)$.
Zobaczmy co się stanie gdy zastosujemy polecenie `InverseSeries` do rozwinięcia funkcji $\exp(x)$ do wyrazów rzędu 7 wokół punktu 0.

```
Series[Exp[x], {x, 0, 7}];
InverseSeries[%].
```

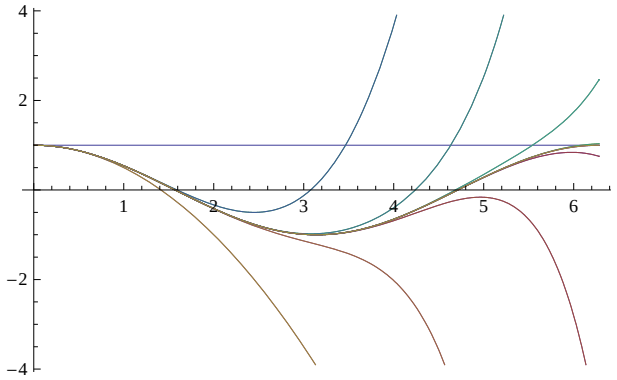
W wyniku uzyskaliśmy szereg potęgowy o wyrazach rzędu 7 wokół punktu 1. Rzut oka wystarcza by stwierdzić, że to fragment rozwinięcia logarytmu, a więc funkcji odwrotnej do $\exp$ wokół punktu $\exp(0) = 1$.
Kolejną przydatną funkcją jest polecenie +ComposeSeries+, służące do znajdowania rozwinięcia złożenia dwóch szeregów. Przykładowo, rozwinięcie złożenia $\sin(\mathrm{tg}(x))$ możemy wyznaczyć albo znanym nam już poleceniem `Series`, albo składając rozwinięcia sinusa i tangensa:

```
Series[Sin[Tan[x]], {x, 0, 5}]
ComposeSeries[Series[Sin[y], {y, 0, 5}], Series[Tan[x], {x, 0, 5}]].
```

Na koniec warto omówić bardzo przydatną funkcję `Normal`. Ucina ona resztę danego szeregu robiąc z niego zwykły wielomian. Dzięki temu możemy na przykład narysować jego wykres. Zobaczmy jak wygląda 20 pierwszych wielomianów Taylora przybliżających kosinusa:

```
Plot[Evaluate[Table[Normal[Series[Cos[x], {x, 0, n}]], {n, 20}]], {x, 0, 2 Pi}]
```

i efekt końcowy:



Zadanie 4.1. Wyznacz wszystkie liczby $a, b \in \mathbb{R}$ dla których granica

$$\lim_{x\to 0} \frac{x - (a + b\cos x)\sin x}{x^5}$$

jest skończona.

Rozwiązanie:

Koncepcyjnie zadanie nie powinno nastręczać żadnych problemów. Wystarczy tak dobrać liczby a i b, aby w rozwinięciu licznika wokół zera występowały wyrazy rzędu co najmniej piątego. Zobaczmy do jakich warunków to prowadzi

```
Series[x - (a + b Cos[x]) Sin[x], {x, 0, 5}].
```

W wyniku dostaliśmy szereg

$$(1 - a - b)x + \frac{1}{6}(a + 4b)x^3 + \left(-\frac{a}{120} - \frac{2b}{15}\right)x^5 + o(x^5).$$

Warunkiem koniecznym i wystarczającym jest zatem zerowanie się współczynników przy potęgach pierwszej i trzeciej w powyższym wyrażeniu. Można to zrobić elegancko w następujący sposób:

```
Series[x - (a + b Cos[x]) Sin[x], {x, 0, 5}] == O[x]^5;
Solve[%, {a, b}].
```

Rozwiązaniem jest $a = \frac{4}{3}$ i $b = -\frac{1}{3}$.

Zajmiemy się teraz zadaniem z kombinatoryki, z pozoru nie związanym z tematem szeregu Taylora.

Zadanie 4.2. Na ile sposobów, dysponując monetami 1, 2 i 5-cio groszowymi, można wydać 99 groszy reszty?

Oczywiście jednym ze sposobów rozwiązania powyższego problemu będzie rozważenie wszystkich możliwości, a więc 99 groszy można przedstawić jako 99 razy 1 grosz, 97 razy 1 grosz i 1 raz 2 grosze, itd. Będzie z tym sporo pracy, ale w końcu pewnie znajdziemy odpowiedź. Tym niemniej, powtórzenie tego rozumowania dla większej różnorodności dostępnych monet, albo

większej kwoty byłoby już bardzo żmudne. Spróbujmy więc podejść do sprawy bardziej metodycznie. Tłumacząc zadanie na bardziej abstrakcyjny język stawiamy pytanie na ile sposobów możemy przedstawić liczbę 99 w postaci kombinacji

$$99 = k \cdot 1 + l \cdot 2 + m \cdot 5, \tag{6}$$

gdzie k, l, m są elementami zbioru $\mathbb{N} \cup \{0\}$.
Zapomnijmy na chwilę o ograniczeniu na sumę powyższej kombinacji i rozważmy wszystkie możliwe wyrażenia postaci $k \cdot 1 + l \cdot 2 + m \cdot 5$. Dokładniej, rozważmy szereg potęgowy zmiennej x następującej postaci

$$S(x) = \sum_{k,l,m \in \mathbb{N} \cup \{0\}} x^{k \cdot 1 + l \cdot 2 + m \cdot 5}.$$

Wobec oczywistej równości $x^{k \cdot 1 + l \cdot 2 + m \cdot 5} = x^{k \cdot 1} \cdot x^{l \cdot 2} \cdot x^{m \cdot 5}$, bez trudu dostajemy, że $S(x)$ jest iloczynem Cauchy'ego trzech szeregów (Definicja 4.47, Twierdzenie 4.48):

$$S(x) = \left(\sum_{k \in \mathbb{N} \cup \{0\}} x^{k \cdot 1} \right) \cdot \left(\sum_{l \in \mathbb{N} \cup \{0\}} x^{l \cdot 2} \right) \cdot \left(\sum_{m \in \mathbb{N} \cup \{0\}} x^{m \cdot 5} \right) = \frac{1}{1-x} \cdot \frac{1}{1-x^2} \cdot \frac{1}{1-x^5}.$$

Czynniki w powyższym iloczynie to szeregi geometryczne zbieżne jednostajnie w kole $|x| < 1$, a więc również $S(x)$, jako iloczyn Cauchy'ego, jest zbieżny jednostajnie dla $|x| < 1$. Zauważmy teraz, że wyraz stopnia 99 w $S(x)$ to dokładnie

$$\sum_{\substack{k,l,m \in \mathbb{N} \cup \{0\} \\ k \cdot 1 + l \cdot 2 + m \cdot 5 = 99}} x^{k \cdot 1 + l \cdot 2 + m \cdot 5} = a_{99} x^{99},$$

gdzie a_{99} jest liczbą wszystkich trójek (k, l, m) spełniających (6), a więc interesującą nas wielkością. Proszę zauważyć, że mamy do czynienia z ciekawą (i jednocześnie dość częstą) sytuacją: rozważanie ogólniejszych trójek nieograniczonych warunkiem (6) pozwoliło nam rozwiązać problem dla specyficznej wartości 99.
Oczywiście nasze rozwiązanie jest na razie czysto teoretyczne: liczba sposobów to współczynnik przy wyrazach odpowiedniego stopnia w rozwinięciu funkcji

$$S(x) = \frac{1}{(1-x)(1-x^2)(1-x^5)}.$$

Szczęśliwie mamy jednak w zanadrzu maszynerię programu *Mathematica*:

```
f[x_]:=1/((1-x)(1-x^2)(1-x^5));
s=Series[f[x], {x, 0, 99}];
SeriesCoefficient[s, 99].
```

W wyniku dostaliśmy 530. Jako ciekawostkę możemy dodać, że funkcję f można zdefiniować także następującą elegancką komendą

```
f[x_]:= 1/Apply[Times, 1 - x^{1, 2, 5}].
```

Stosujemy tutaj (`Apply`) operator mnożenia (`Times`) do wyrazów postaci $1 - x^a$, gdzie a jest elementem listy `{1, 2, 5}`.

Problemy do samodzielnego rozwiązania:

Zadanie 4.3. Wiedząc, że

$$f(x) = x + \frac{x^2}{4} - \frac{x^3}{7} + \frac{x^5}{2} + \frac{x^6}{12} + o(x^6)$$

znajdź piąty wielomian Taylora funkcji $g(x)$ jeśli

$$\bullet\ g(x) = f(\sin(x))\,, \qquad \bullet\ g(x) = f(x)\arcsin(x)\,, \qquad \bullet\ g(x) = \operatorname{tg}(f(x))\,.$$

Zadanie 4.4. Obliczyć na ile sposobów można przedstawić 100 złotych za pomocą dostępnych w polskim obiegu monet. Uwaga, obliczenia mogą zająć trochę czasu. Ilokrotnie otrzymana liczba przewyższa odległość od Ziemi do Słońca wyrażoną w milimetrach? Ilokrotnie zmniejszy się ta liczba, gdy wycofamy z obiegu monety pięciozłotowe, a ilokrotnie gdy wycofamy monety jednogroszowe?

Zadanie 4.5. Na ile sposobów można przedstawić 99 groszy za pomocą monet 1, 2 i 5-cio groszowych mając do dyspozycji tylko 5 monet jednogroszowych?

4.3 Badanie zbieżności punktowej i jednostajnej

Nowe umiejętności: Badanie zbieżności jednostajnej i punktowej ciągów i szeregów funkcyjnych, używanie kryterium Weierstrassa, używanie twierdzenia o różniczkowaniu ciągów i szeregów funkcyjnych.

Skrypt: Rozdziały 7. (podlinkować)

Spróbujemy teraz użyć programu *Mathematica* aby wspomóc proces badania zbieżności jednostajnej i punktowej ciągów i szeregów funkcyjnych.

Zadanie 4.6. Zbadać zbieżność punktową i jednostajną ciągu funkcji

$$f_n(x) = nx\exp\left(-\frac{nx^2}{2}\right)$$

na zbiorze $[0, +\infty)$.

Rozwiązanie:

Zacznijmy od zdefiniowania funkcji f_n:

```
f[n_,x_]:=n x Exp[-n x^2/2].
```

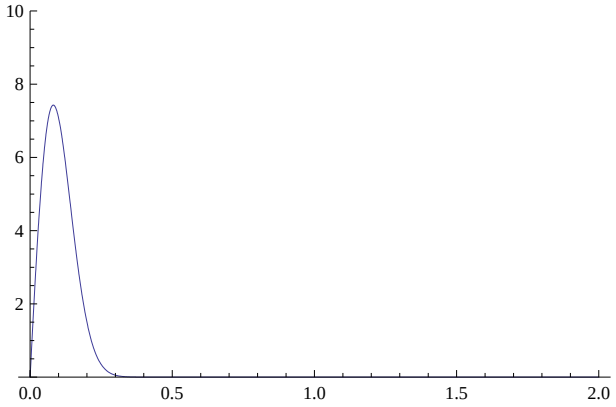
Zbieżność punktową możemy bez trudu zbadać korzystając ze znanego nam już polecenia `Limit`:

```
Limit[f[n,x], n->Infinity, Assumptions->Element[x,Reals]].
```

Wynikiem jest 0 niezależne od x, co oznacza, że $f_n(x)$ jest zbieżny punktowo do funkcji stałej $f_0 \equiv 0$. Jest to oczywiście także naturalny kandydat na granicę jednostajną. Spróbujmy zwizualizować sobie zachowanie funkcji f_n. Wygodnie będzie posłużyć się znanym nam już poleceniem `Animate`.

```
Animate[Plot[f[n, x], {x, 0, 2}, PlotRange -> {0, 10}], {n, 1, 500}].
```

Zmieniając wartość n w zakresie od 1 do 500 możemy obserwować zachowanie funkcji f_n.



Jak widać mamy do czynienia z „wędrującym pagórkiem": wraz ze wzrostem n maksima funkcji f_n rosną i przesuwają się w stronę 0. Takie zachowanie w oczywisty sposób wyklucza zbieżność jednostajną funkcji f_n do f_0. Spróbujmy to teraz formalnie udowodnić. W tym celu zbadamy funkcję f_n. Rozważmy nierówność $f_n^{'}(x) > 0$.

```
Reduce[{D[f[n,x],x]>0,x>=0},x] .
```

Okazuję się, że $f_n^{'}(x) > 0$ dla x-ów z przedziału $(0, \frac{1}{\sqrt{n}})$. Wynika stąd, że f_n rośnie na przedziale $(0, \frac{1}{\sqrt{n}})$, osiąga maksimum równe $\frac{\sqrt{n}}{\sqrt{e}}$ w punkcie $\frac{1}{\sqrt{n}}$ i maleje (do zera) na przedziale $(\frac{1}{\sqrt{n}}, +\infty)$. Ponieważ $f_n(x) \geq 0$ dla $x > 0$, zatem

$$\frac{\sqrt{n}}{\sqrt{e}} = \sup_{x\in[0,+\infty)} f_n(x) = \sup_{x\in[0,+\infty)} |f_n(x)| = \sup_{x\in[0,+\infty)} |f_n(x) - 0| = \|f_n - 0\| .$$

Wynika stąd, że $\|f_n - 0\| \to +\infty$ przy $n \to \infty$, czyli ciąg f_n nie ma granicy jednostajnej.
Rozwiązanie (*sposób 2*)**:**
Wykluczyć zbieżność jednostajną możemy także prościej o ile nieco wnikliwiej przyjrzymy się funkcjom f_n. Zauważmy mianowicie, że

$$f_n(x) = \sqrt{n} \cdot \sqrt{n}x \exp\left(-\frac{(\sqrt{n}x)^2}{2}\right) = \sqrt{n}g(\sqrt{n}x) ,$$

gdzie $g(x) = x\exp\left(-\frac{x^2}{2}\right)$. Oznaczmy $a := \sup_{y\in[0,+\infty)} g(y)$. Mamy teraz

$$\sup_{x\in[0,+\infty)} f_n(x) = \sqrt{n}\sup_{x\in[0,+\infty)} g(\sqrt{n}x) = \sqrt{n}\sup_{y\in[0,+\infty)} g(y) = \sqrt{n}\cdot a\,.$$

Wobec tego, skoro $a > 0$, mamy $\sup_{x\in[0,+\infty)} f_n(x) \underset{n\to+\infty}{\longrightarrow} +\infty$.

Zadanie 4.7. Wykazać, że funkcja $F(x) = \sum_{n=1}^{\infty}\sqrt{x}\exp(-n^2x)$ jest dobrze określona i ciągła na przedziale $(0,+\infty)$. Rozstrzygnąć, czy F jest ciągłą w zerze.

Rozwiązanie:

Podobnie jak w poprzednim zadaniu zacznijmy od zdefiniowania rodziny funkcji $f_n(x) = \sqrt{x}\exp(-n^2x)$

```
f[n_,x_]:=Sqrt[x] Exp[-n^2 x].
```

Naturalną strategią w tego typu problemach jest użycie twierdzenia mówiącego, że granica jednostajna ciągu funkcji ciągłych jest funkcją ciągłą. Funkcje f_n są oczywiście ciągłe, spróbujemy zatem zbadać zbieżność jednostajną szeregu $\sum f_n$ na przedziale $(0,+\infty)$. Narzucającym się podejściem jest użycie twierdzenie Weierstrassa (Stwierdzenie 7.13). Obliczmy więc normy $\|f_n\|$ dla poszczególnych składników. Zbadamy w tym celu kiedy pochodna $f_n^{'}$ jest dodatnia. Zastosowanie polecenia `Reduce` do nierówności $f_n^{'}(x) > 0$ nie przynosi rezultatu, wobec tego musimy postępować na raty:

```
Simplify[D[f[n,x],x]>0].
```

Otrzymujemy $\exp(-n^2x)\frac{\left(1-2n^2x\right)}{\sqrt{x}} > 0$. Ponieważ $x > 0$, a funkcja wykładnicza przyjmuje tylko wartości dodatnie, wystarczy że zbadamy nierówność $\left(1-2n^2x\right) > 0$:

```
Reduce[(1 - 2 n^2 x) > 0, x].
```

Z otrzymanych wyników wynika, że f_n rośnie na przedziale $(0, \frac{1}{2n^2})$, osiąga maksimum w punkcie $\frac{1}{2n^2}$, które wynosi $\frac{1}{\sqrt{2}n}\exp(-\frac{1}{2})$ i maleje (do zera) na przedziale $(\frac{1}{2n^2},+\infty)$. Wynika stąd, że szereg $\sum\|f_n\|$ zachowuje się jak szereg harmoniczny, a więc nie jest zbieżny. Podejście poprzez twierdzenie Weierstrassa nie udało się zatem.

Zauważmy jednak, że tak naprawdę obliczyliśmy normę $\|f_n\|$ na całym przedziale $[0,+\infty)$, podczas gdy interesuje nas tylko ciągłość na przedziale $(0,+\infty)$. Wystarczy więc, że pokażemy zbieżność jednostajną na zbiorach postaci $[a,+\infty)$, gdzie a jest dowolną (ale ustaloną) liczbą dodatnią. Wówczas będziemy mogli wywnioskować, że F jest dobrze określone i ciągłe na przedziałach $[a,+\infty)$, a ponieważ zbiorami tej postaci można pokryć cały przedział $(0,+\infty)$, wynikać stąd będzie także ciągłość F na $(0,+\infty)$.

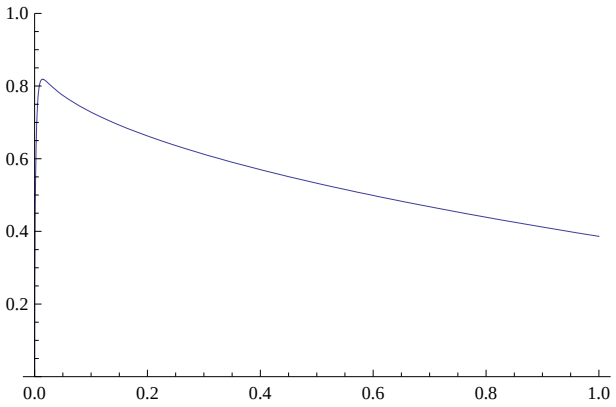
Z przeprowadzonej wcześniej analizy znaku pochodnej wynika, że

$$\sup_{x\in[a,+\infty)} f_n(x) = \begin{cases} f_n(\frac{1}{2n^2}) & \text{gdy } \frac{1}{2n^2}\in[a,+\infty) \\ f_n(a) & \text{w przeciwnym przypadku.}\end{cases}$$

Ponieważ przy ustalonym $a > 0$ prawie wszystkie liczby naturalne n spełniają zależność $\frac{1}{2n^2} < a$, mamy $\|f_n\|_{[a,+\infty)} = f_n(a)$ dla prawie wszystkich n. Wobec tego zbieżność szeregu norm

$\sum \|f_n\|_{[a,+\infty)}$ i szeregu $\sum f_n(a) = \sum \sqrt{a}\exp(-n^2 a)$ są równoważne. Ostatni szereg jest oczywiście zbieżny a zatem, na mocy twierdzenia Weierstrassa, szereg $\sum f_n(x)$ jest zbieżny jednostajnie nie zbiorach postaci $[a, +\infty)$, gdzie $a > 0$. W szczególności funkcja $F(x)$ jest ciągła na $[a, +\infty)$ jako granica jednostajna ciągu funkcji ciągłych.
Zajmijmy się z kolei ciągłością F w zerze. Oczywiście $F(0) = 0$. Jak wiemy, poprzednio nie udało się nam użyć twierdzenie Weierstrassa aby dowieść ciągłości w zerze. Powodem był fakt, że funkcje f_n miały maksima rzędu $\frac{1}{n}$ które wysumowane tworzą szereg rozbieżny. Maksima te są zlokalizowane w punktach $\frac{1}{2n^2}$, a więc skupiają się w zerze. Musimy rozstrzygnąć, czy wkład funkcji f_n do sumy jest na tyle mały, żeby $F(x)$ było bliskie 0 o ile x jest dostatecznie bliskie 0. Ponieważ odpowiedź nie wydaje się oczywista spróbujmy przyjrzeć się badanemu szeregowi. Narysujemy w tym celu wykresy ciągu sum częściowych F_n szeregu F. Użycie funkcji `Animate` pozwoli nam zaobserwować zachowanie się tego ciągu dla różnych n.

```
F[n_,x_]:=Sum[f[k,x],{k,1,n}];
Animate[Plot[F[n, x], {x, 0, 1}, PlotRange -> {0, 1}], {n, 1, 100}] .
```



Z wykresu widać, że sumy częściowe $F_n(x)$ dla x-ów bliskich zera stabilizują się blisko wartości 0.8, żeby potem gwałtownie spaść do zera. Takie zachowanie sugeruje, że w granicy będziemy mieli do czynienia ze skokiem wartości funkcji, a więc F nie będzie ciągła w zerze.
Spróbujmy to udowodnić. W tym celu wystarczy skonstruować ciąg x_k zbieżny do 0 i znaleźć liczbę $a > 0$ taką aby $F(x_k) > a$ dla każdego k. Zauważmy, że funkcja $f_n(x)$ jest iloczynem czynnika $\sqrt{x}$, który jest mały dla x bliskich zera i czynnika $\exp(-n^2 x)$, który dla małych x jest bliski jedności (dokładniej $\exp(-n^2 x) \approx 1$ dla $x = o(\frac{1}{n^2})$). Wobec tego wydaje się dość naturalne aby dobrać x_k w taki sposób, żeby $\exp(-n^2 x_k)$ było bliskie 1 dla dużej liczby n-ów. Wówczas do $F(x_k)$ wkład będzie miało wiele małych przyczynków rzędu $\sqrt{x_k}$, które w sumie dadzą duży wkład. Tę strategię bardzo łatwo zrealizować biorąc na przykład ciąg $x_k = \frac{1}{k^2}$. Wówczas[14] możemy łatwo oszacować:

$$F(x_k) > \sum_{n=1}^{k} f_n(x_k) = \sum_{n=1}^{k} \sqrt{\frac{1}{k^2}} \exp\left(-\frac{n^2}{k^2}\right) \geq \sum_{1}^{k} \frac{1}{k} \cdot \frac{1}{e} = \frac{1}{e} > 0.$$

[14]Tak naprawdę dowolny ciąg zbieżny do zera z prawej strony będzie dobry. Wynika to, oczywiście, z ciągłości F na przedziale $(0, +\infty)$.

Ponieważ $x_k \to 0$ ale $F(x_k)$ nie zbiega do $F(0) = 0$ przy $k \to +\infty$, udowodniliśmy, że F nie jest ciągła w zerze.

Zadanie 4.8. Wykazać, że funkcja $F(x) = \sum_{n=1}^{\infty} \frac{x^2}{x^4+n^4}$ jest dobrze określona i klasy C^1 na całej prostej rzeczywistej $\mathbb{R}$.

Rozwiązanie:

Jak wiadomo z wykładu (Twierdzenie 7.19), jeżeli szereg funkcyjny $F(x) = \sum f_n(x)$ jest zbieżny w jednym punkcie i szereg pochodnych $\sum f_n'(x)$ jest zbieżny jednostajnie, wówczas szereg $\sum f_n(x)$ jest również zbieżny jednostajnie i ponadto funkcja F jest różniczkowalna oraz zachodzi równość $F'(x) = \sum f_n'(x)$ (inaczej mówiąc można różniczkować wyjściowy szereg „wyraz po wyrazie").

Spróbujmy zastosować powyższe twierdzenie do szeregu w zadaniu. Ze zbieżnością badanego szeregu w punkcie $x = 0$ nie ma najmniejszych problemów, bowiem $F(0) = \sum 0 = 0$. Zajmijmy się teraz szeregiem pochodnych.

```
f[n_, x_] := x^2/(x^4 + n^4);
D[f[n, x], x];
Simplify[%].
```

Jak widzimy zachodzi równość $f_n'(x) = \frac{2x\left(n^4-x^4\right)}{(n^4+x^4)^2}$. Spróbujmy zbadać zbieżność jednostajną korzystając z twierdzenia Weierstrassa. W tym celu zbadajmy przebieg funkcji $f_n'(x)$:

```
pf[n_, x_] := (2 x (n^4 - x^4))/(n^4 + x^4)^2;
Reduce[D[pf[n,x],x]>0,x].
```

Z analizy znaku drugiej pochodnej $f_n''(x)$ łatwo wywnioskować, że pochodna $f_n'(x)$ ma dwa lokalne maksima – w punktach $x_n := -\left(2+\sqrt{\frac{11}{3}}\right)^{\frac{1}{4}} n$ i $y_n := (2-\sqrt{\frac{11}{3}})^{\frac{1}{4}} n$. Wartości w tych punktach są postaci $c \cdot \frac{1}{n^3}$, gdzie c to stała liczbowa niezależna od n (najprościej wyliczyć to za pomocą komend `Simplify[pf[n,xn],n>0]` i `Simplify[pf[n,yn],n>0]`, gdzie `xn` i `yn` to odpowiednio punkty x_n i y_n). Ponieważ pochodna $f_n'(x)$ jest funkcją antysymetryczną i jej granice w plus i minus nieskończoności wynoszą zero wnioskujemy, że $\|f_n'\| = c \cdot \frac{1}{n^3}$. Szereg norm $\sum \|f_n'\|$ jest zatem zbieżny, a zatem $\sum f_n'$ jest zbieżny jednostajnie. Ponieważ f_n' są funkcjami ciągłymi, więc także $\sum f_n'(x) = F'(x)$ jest funkcją ciągłą.

Uwaga! W powyższym rozwiązaniu główną korzyścią korzystania z komputera było ominięcie dość pracochłonnych rachunków potrzebnych do znalezienia maksimum pochodnej f_n', a w konsekwencji normy $\|f_n'\|$. Można jednak w prosty sposób obejść się bez tych wyliczeń, za pomocą oszacowania:

$$\left|f_n'(x)\right| = \left|\frac{2x\left(n^4-x^4\right)}{(n^4+x^4)^2}\right| \leq \frac{2|x|\left(n^4+x^4\right)}{(n^4+x^4)^2} = \frac{2|x|}{(n^4+x^4)} \leq \frac{2}{n^3}\,.$$

Ostatnią nierówność zostawiamy jako ćwiczenie dla Czytelnika. Ponieważ uzyskane oszacowanie nie zależy od x wnioskujemy, że

$$\|f_n'\| \leq \frac{2}{n^3}\,.$$

Zbieżność szeregu $\sum \|f_n'\|$ wynika teraz z kryterium porównawczego (porównujemy z szeregiem $\sum \frac{2}{n^3}$).

Problemy do samodzielnego rozwiązania:

Zadanie 4.9. Zbadać zbieżność punktową i jednostajną ciągów

$$f_n(x) = \frac{n}{n+x^2} \quad \text{dla } x \in \mathbb{R}\,,$$
$$f_n(x) = \frac{n^2}{(n+x)^2} \quad \text{dla } x \in \mathbb{R}_+\,,$$
$$f_n(x) = \exp\left(-\sqrt{n}\left(x-\tfrac{1}{n}\right)^2\right) \quad \text{dla } x \in \mathbb{R}\,.$$

Zadanie 4.10. Zbadać zbieżność punktową i jednostajną ciągu na $\mathbb{R}$

$$f_n(x) = \frac{1}{n}\cos(nx) - n\sin(x) + \cos(x) + n\sin\left(x+\tfrac{1}{n}\right).$$

Zadanie 4.11. Zbadać, czy funkcja dana wzorem

$$F(x) = \sum_{n=1}^{\infty} \frac{x}{n\sqrt{n}\exp\left((nx-1)^2\right)}$$

jest dobrze określona i ciągła na całej osi $\mathbb{R}$. Czy jest klasy C^1?

5 Całka

5.1 Całkowanie

Nowe umiejętności: obliczanie całek oznaczonych i nieoznaczonych, całkowanie numeryczne.

Skrypt: Rozdział 9. (podlinkować)

Nowe funkcje:

Mathematica	*Maxima*	Komentarz
`Integrate[f[x],x]`	`integrate(f(x),x)`	całka nieoznaczona
`Integrate[f[x],{x,a,b}]`	`integrate(f(x),x,a,b)`	całka oznaczona
`NIntegrate[f[x],{x,a,b}]`	`romberg(f(x),x,a,b)`	całkowanie numeryczne

Obliczanie całek oznaczonych i nieoznaczonych w programie *Mathematica* nie nastręcza większych trudności. Przykładowo wyrażenia

$$\int x^5\cos x\,\mathrm{d}x \quad \text{i} \quad \int_1^3 e^{-x}x^3\,\mathrm{d}x$$

obliczymy wpisując polecenia

```
Integrate[x^5 Cos[x],x]
Integrate[Exp[-x] x^3, {x,1,3}].
```

Jak wiadomo, obliczenie całek nieoznaczonych z funkcji elementarnych często wyprowadza poza klasę funkcji elementarnych. Przykładem takiej sytuacji jest całka

$$\int \exp(-x^2)\, dx\,.$$

Programy typu CAS często potrafią podać poprawną wartość całki wyrażoną poprzez tka zwane *funkcje specjalne*. Przykładowo, wpisując w *Mathematice*

```
Integrate[2/Sqrt[Pi]Exp[-x^2],x]
```

otrzymamy odpowiedź $\mathrm{Erf}(x)$. Funkcja Erf to tak zwana *funkcja błędu Gaussa*, która jest zdefiniowana jako funkcja pierwotna dla $\frac{2}{\sqrt{\pi}}\exp(-x^2)$. W trakcie pracy z programami CAS Czytelnik może spotkać jeszcze wiele innych funkcji tego rodzaju – między innymi poznane na wykładzie funkcje gamma i beta Eulera (Rozdział 10). Funkcje te są dobrze zbadane i pewne ich własności są „zaszyte" w oprogramowaniu CAS. W szczególności daje się obliczać przybliżone (czasem też dokładne) wartości danej funkcji. Przykładowo wpisać w *Mathematice*

```
Integrate[Exp[-x^2],{x,-Infinity,Infinity}]
```

co da poprawny wynik $\sqrt{\pi}$. *Maxima* również poradzi sobie z tą całką – wystarczy wpisać

```
integrate(exp(-x^2),x,-minf,inf) .
```

Czasem zdarza się jednak, że poszukiwana całka nie wyraża się żadnym sensownym wzorem zawierającym jakiekolwiek funkcje znane naszemu programowi. Próbując obliczyć taką całkę, dostaniemy w odpowiedzi dokładnie to co wpisaliśmy[15]. Do obliczania konkretnych wartości całek oznaczonych można się wówczas posłużyć całkowaniem numerycznym. Przykładowo, do obliczenia całki

$$\int_0^1 \sin(\cos(x))\, dx$$

możemy użyć polecenia

```
NIntegrate[Sin[Cos[x]],{x,0,1}] .
```

W *Maximie* analogiczne polecenie wygląda następująco

```
load(romberg);
romberg(sin(cos(x)),x,0,1); .
```

[15]Mamy do czynienia z analogiczną sytuacją jak w przypadku szeregów liczbowych

Trzeba mieć jednak świadomość, że nie wszystkie całki daje się obliczać w ten sposób. Całkowanie numeryczne wykorzystuje pewien algorytm, który liczy pole pod wykresem danej funkcji przy pomocy sumowania pól małych prostokątów lub innych, prostych figur. Funkcje, które bardzo szybko oscylują nie dają się scałkować w ten sposób[16]. Czytelnik może przeprowadzić test próbując obliczyć wartości całek

$$\int_0^1 \sin\left(\tfrac{1}{x}\right)\,dx \quad \text{oraz} \quad \int_0^\infty \frac{\sin(\exp(x))}{x}\,dx\,.$$

Zadanie 5.1. Oblicz całkę

$$\int_0^\pi \frac{\sin kx}{\sin x}\mathrm{d}x\,,$$

gdzie $k \in \mathbb{N}$.
Rozwiązanie (*sposób 1*)**:**
Mathematica nie jest w stanie obliczyć tej całki dla ogólnego k, nawet przy dodatkowych założeniach. Polecenie

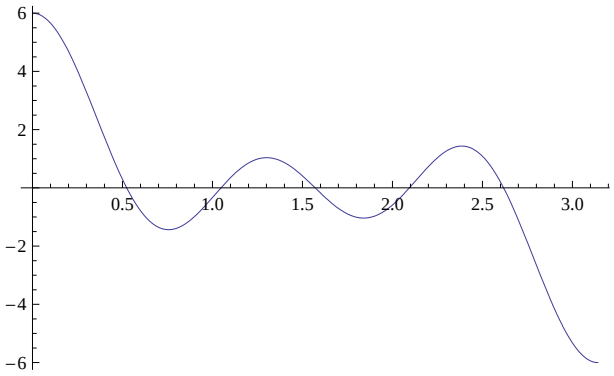
```
Assuming[Element[k, Integers], Integrate[Sin[k x]/Sin[x], {x, 0, Pi}]]
```

nie daje żadnych rezultatów – program nie jest w stanie skończyć wyliczeń. Spróbujmy zatem zmienić sposób postępowania i wypisać rezultaty dla kilkunastu różnych wartości k:

```
Table[Integrate[Sin[k x]/Sin[x], {x, 0, Pi}], {k, 1, 15, 1}] .
```

W odpowiedzi dostaliśmy naprzemienny ciąg składający się z liczb π i 0. Spodziewamy się zatem, że wartość całki dla nieparzystych k wynosi π, natomiast dla k parzystych 0. Ten drugi fakt dość prosto wykazać. Naszkicujmy wykres jednej z rozpatrywanych funkcji

```
Plot[Sin[6 x]/Sin[x], {x, 0, Pi}] .
```



Widzimy, że jest on antysymetryczny względem punktu $\frac{\pi}{2}$. Jest to również prawdą dla dowolnego parzystego k

[16] Znów nasuwa się analogia z numerycznym sumowaniem

```
f[k_,x_]:=Sin[k x]/Sin[x];
Simplify[f[2 k, x] + f[2 k, Pi - x], Element[k, Integers]].
```

Stąd wprost wynika, że całka z funkcji $\frac{\sin(2kx)}{\sin(x)}$ po odcinku $[0,\pi]$ da zero.
Aby wykazać, że całka dla nieparzystych k daje π spróbujmy przeprowadzić dowód indukcyjny. Początek indukcji nie nastręcza problemów:

$$\int_0^\pi \frac{\sin(x)}{\sin(x)}\mathrm{d}x = \int_0^\pi 1\mathrm{d}x = \pi.$$

Porównajmy teraz dwa kolejne wyrażenia przekształcając wyrażenie za pomocą funkcji `TrigFactor` służącej do rozkladania wyrażenia na iloczyn czynników trygonometrycznych

```
TrigFactor[f[2 k + 3, x] - f[2 k + 1, Pi - x]].
```

W wyniku dostaliśmy iloczyn trzech czynników

$$-2\cos\left(\frac{1}{2}(1+2k)(\pi-x)+\frac{1}{2}(3+2k)x\right)\cdot\frac{1}{\sin(x)}\cdot\sin\left(\frac{1}{2}(1+2k)(\pi-x)-\frac{1}{2}(3+2k)x\right).$$

Przekształćmy teraz dwa skrajne czynniki za pomocą funkcji `Simplify`[17]

```
Simplify[-2 Cos[1/2 (1+2 k) (Pi-x) + 1/2 (3+2 k) x], Element[k, Integers]]
Simplify[Sin[1/2 (1+2 k) (Pi-x) - 1/2 (3+2 k) x], Element[k, Integers]]  .
```

Okazuje się, że pierwsze z nich to po prostu $(-1)^k\sin(x)$, a drugie $(-1)^k\cos(2(k+1)x)$. W konsekwencji badana różnica kolejnych wyrażeń to po prostu $\cos(2(k+1)x)$. Łatwo sprawdzić, że odpowiednia całka znika

$$\int_0^\pi \cos(2(k+1)x)\mathrm{d}x = 0.$$

Rozwiązanie (*sposób 2*)**:**
Drugi sposób rozwiązania jest nieco bardziej pomysłowy i będzie od nas wymagał użycia liczb zespolonych. Rachunki będą na tyle elementarne, że nie będziemy potrzebować pomocy komputera. Przypomnijmy mianowicie, że $\sin(x) = \frac{e^{\mathrm{i}\,x}-e^{-\mathrm{i}\,x}}{2\,\mathrm{i}}$. Wobec tego

$$\frac{\sin(kx)}{\sin(x)} = \frac{e^{\mathrm{i}\,kx}-e^{-\mathrm{i}\,kx}}{e^{\mathrm{i}\,x}-e^{-\mathrm{i}\,x}}.$$

Korzystając ze wzoru skróconego mnożenia otrzymujemy

$$\frac{\sin(kx)}{\sin(x)} = e^{\mathrm{i}(k-1)x}+e^{\mathrm{i}(k-3)x}+e^{\mathrm{i}(k-5)x}+\ldots+e^{-\mathrm{i}(k-1)x}.$$

Grupując teraz wyrazy parami i korzystając z tego, że $\cos(x) = \frac{e^{\mathrm{i}\,x}+e^{-\mathrm{i}\,x}}{2}$ otrzymamy

$$\frac{\sin(kx)}{\sin(x)} = \cos((k-1)x)+\cos((k-3)x)+\cos((k-5)x)+\ldots,$$

[17] Użycie `Simplify` bezpośrednio do całego wyrażenia zwija je do postaci początkowej.

gdzie ostatnim wyrazem jest $\cos(kx)$ lub $1 = \cos(0x)$ w zależności od parzystości k. Całkując ostatnią równość stronami łatwo dowiedziemy, że wynikiem jest 0 dla k parzystych i π dla k nieparzystych.

Problemy do samodzielnego rozwiązania:

Zadanie 5.2. Wykorzystując oprogramowanie CAS spróbować obliczyć funkcje pierwotne następujących funkcji

$$f(x) = \sin\left(\tfrac{1}{x}\right) \quad \text{oraz} \quad g(x) = \operatorname{tg}\sin(x)\,.$$

Zadanie 5.3. Obliczyć całkę

$$\int \exp(5x)\cos(x+3)x^3\,dx\,.$$

Zadanie 5.4. Obliczyć całkę

$$\int (\sin(2x)+\cos(x))(\operatorname{tg}(x)+5)\,dx\,.$$

Zadanie 5.5. Niech $k, l \in \mathbb{N}$. Obliczyć

$$\int_0^{2\pi} \sin(kx)\sin(lx)\,dx \quad \text{oraz} \quad \int_0^{2\pi} \sin(kx)\cos(lx)\,dx\,.$$

5.2 Zastosowania rachunku całkowego

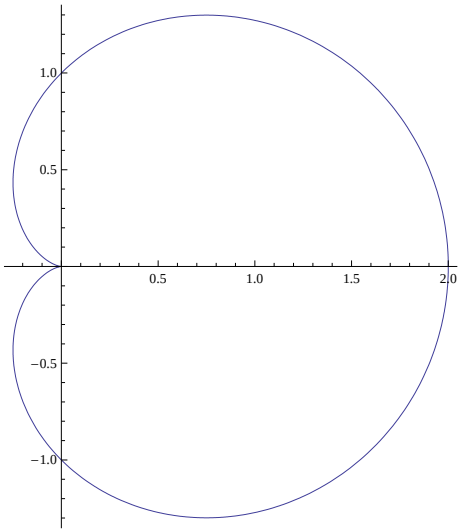
Nowe umiejętności: obliczanie długości krzywych, pól powierzchni i objętości brył.
Skrypt: Rozdział 9.4. (podlinkować)

Zadanie 5.6. Oblicz pole i długość kardioidy danej we współrzędnych biegunowych równaniem $r(\phi) = 1 + \cos(\phi)$.
Rozwiązanie (*sposób 1*)**:**
Rozwiązywanie zacznijmy od zdefiniowania pomocniczych funkcji

```
r[Phi_]:=1+Cos[Phi]
x[Phi_]:=r[Phi] Cos[Phi]
y[Phi_]:=r[Phi] Sin[Phi]
```

i narysowania wykresu kardioidy

```
PolarPlot[r[Phi],{Phi,0,2 Pi}] .
```



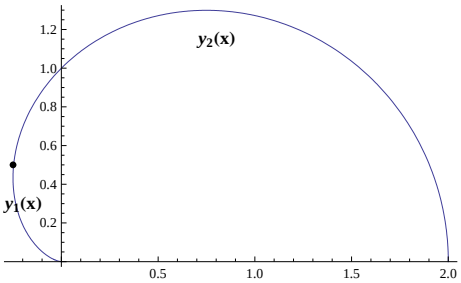
Z rysunku widać od razu, że kardioida jest symetryczna względem osi OX. Wystarczy zatem, że obliczymy interesujące nas wielkości dla jej górnej połowy – jest to część sparametryzowana przez kąt $\phi \in [0, \pi]$.
Długość możemy łatwo wyliczyć ze wzoru na długość krzywej zadanej parametrycznie

$$L = 2 \int_0^{\pi} \sqrt{(x'(\phi))^2 + (y'(\phi))^2} \mathrm{d}\phi,$$

gdzie różniczkujemy względem ϕ. W *Mathematice*:

```
L=2 Integrate[Sqrt[D[x[Phi],Phi]^2+D[y[Phi],Phi]^2],{Phi,0,Pi}] .
```

Wynikiem jest $L = 8$.
Nieco bardziej skomplikowanie przedstawia się sprawa z polem powierzchni. Spróbujmy najpierw wyrazić y jako funkcję x. Widzimy, że dla pewnych wartości x zależność ta nie jest jednoznaczna. Będziemy mieli do czynienia z dwoma gałęziami funkcji $y(x)$ (Rysunek 4)



Rysunek 4: Dwie gałęzie funkcji $y(x)$.

Z zależności $x(\phi) = (1 + \cos(\phi))\cos(\phi)$ łatwo się przekonać, że $x(\phi)$ jest malejącą funkcją kosinusa dla $\cos(\phi) \in [-1, -\frac{1}{2}]$ i rosnącą funkcją kosinusa dla $\cos(\phi) \in [-\frac{1}{2}, 1]$. Punkt gra-

niczny $\cos(\phi) = -\frac{1}{2}$ to oczywiście wierzchołek paraboli $(z+1)z$. Odpowiada on $\phi = \frac{2}{3}\pi$ oraz $x(\phi) = -\frac{1}{4}$. Ogólniej $x \in [2, -\frac{1}{4}]$ dla $\phi \in [0, \frac{2}{3}\pi]$ oraz $x \in [-\frac{1}{4}, 0]$ dla $\phi \in [\frac{2}{3}\pi, \pi]$.
W celu wyznaczenia zależności $y(x)$ wyliczmy najpierw kosinus kąta ϕ jako funkcję x.

```
Solve[x == (1 + Cos[Phi]) Cos[Phi], Cos[Phi]].
```

Otrzymane rozwiązanie $\cos(\phi) = \frac{1}{2}(-1 + \sqrt{1+4x})$ dla $x \in [-\frac{1}{4}, 2]$ opisuje górną gałąź, natomiast $\cos(\phi) = \frac{1}{2}(-1 - \sqrt{1+4x})$ dla $x \in [0, -\frac{1}{4}]$ dolną gałąź wykresu. Korzystając z jedynki trygonometrycznej i wstawiając wyliczone wartości $\cos(\phi)$ do zależności $y(\phi) = (1+\cos(\phi))\sin(\phi)$ otrzymamy:

$$y_1(x) = \frac{1}{2}(1 - \sqrt{1+4x})\sqrt{1 - \frac{1}{4}\left(1 - \sqrt{1+4x}\right)^2}$$
$$y_2(x) = \frac{1}{2}(1 + \sqrt{1+4x})\sqrt{1 - \frac{1}{4}\left(1 + \sqrt{1+4x}\right)^2}.$$

Pole otrzymamy jako różnicę odpowiednich całek nieprzyjemne rachunki (po uprzednim zdefiniowaniu funkcji $y_1(x)$ i $y_2(x)$) zostawiając komputerowi

```
S=-2 Integrate[y1[x],{x,-1/4,0}]+2 Integrate[y2[x],{x,-1/4,2}];
Simplify[%] .
```

Rezultatem jest $S = \frac{3}{2}\pi$.
Rozwiązanie (*sposób 2*)**:**
Jak widzimy powyższe rozwiązanie jest dość skomplikowane ze względu na konieczność wyliczania y jako funkcji x z uwikłanej zależności. Dużo prościej jest potraktować $y(\phi)$ i $x(\phi)$ jak swego rodzaju zamianę zmiennych. Zależność $y(\phi)$ możemy traktować jako złożenie nieznanej zależności $y = y(x)$ z podstawieniem $x = x(\phi)$. Teraz pole wyliczamy jako

$$\begin{aligned} S = &-2\int_{-\frac{1}{4}}^{0} y_1(x)\mathrm{d}x + 2\int_{-\frac{1}{4}}^{2} y_2(x)\mathrm{d}x = \\ &-2\int_{\frac{2}{3}\pi}^{\pi} y_1(x(\phi))x'(\phi)\mathrm{d}\phi + 2\int_{\frac{2}{3}\pi}^{0} y_2(x(\phi))x'(\phi)\mathrm{d}\phi = \\ &-2\int_{\frac{2}{3}\pi}^{\pi} y(\phi)x'(\phi)\mathrm{d}\phi + 2\int_{\frac{2}{3}\pi}^{0} y(\phi)x'(\phi)\mathrm{d}\phi = 2\int_{\pi}^{0} y(\phi)x'(\phi)\mathrm{d}\phi. \end{aligned}$$

Zauważmy, że tym razem nie musimy przejmować się już jednoznacznością funkcji $y(x)$. W tym przypadku jednoznaczna jest parametryzacja za pomocą kąta ϕ. Zauważmy również, że pochodna $x'(\phi)$ jest dla kątów z przedziału $\phi \in (\frac{2}{3}\pi, \pi)$ ujemna, co odpowiada „cofaniu się" wartości x i w rezultacie braniu odpowiedniego pola ze znakiem minus – por. rysunek 4.
Sprawdźmy jeszcze czy wartość pola obliczonego drugą metodą zgada się z tą obliczoną wcześniej:

```
2 Integrate[y[Phi] D[x[Phi],Phi],{Phi,Pi,0}] .
```

Zadanie 5.7. Oblicz objętość i pole powierzchni bocznej bryły powstałej przez obrót pętli danej parametrycznie

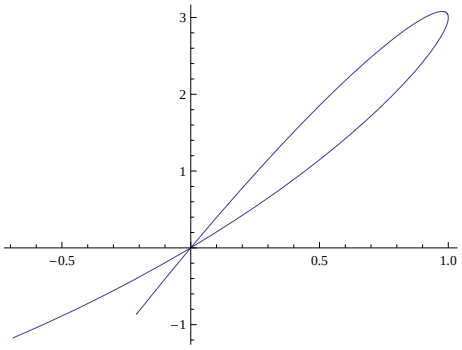
$$\begin{aligned} x(t) &= 2t - t^2, \\ y(t) &= 4t - t^3, \end{aligned} \tag{7}$$

wokół osi OX.

Rozwiązanie:

Zacznijmy od zdefiniowania odpowiednich funkcji i narysowania wspomnianej pętli

```
x[t_]:=2 t-t^2;
y[t_]:=4 t- t^3;
ParametricPlot[{x[t], y[t]}, {t, -0.3, 2.1}, AspectRatio -> 0.75]
```



Optycznie widać, że krzywa zadana równaniem (7) ma jedyne samoprzecięcie w punkcie $(0, 0)$. Faktycznie łatwo sprawdzić, że ów punkt odpowiada dwóm wartościom parametru $t = 0$ i $t = 2$ (oczywiście może to też za nas obliczyć komputer). Przypomnijmy, że objętość bryły powstałej przez obrót wykresu funkcji $f(x)$ określonej na odcinku $[a, b]$ wokół osi OX zadana jest wzorem

$$\pi \int_a^b y^2(x)\mathrm{d}x.$$

W rozważanym problemie moglibyśmy wyznaczyć dwie gałęzie funkcji $y = y(x)$ rozwiązując zależność uwikłaną (7) a następnie odjąć od siebie odpowiednie całki jak w pierwszym rozwiązaniu poprzedniego zdania. Zamiast tego potraktujmy parametryzację (7) jako zmianę zmiennych. Wówczas

$$V = \pi \int_0^2 y^2(t)x'(t)\mathrm{d}t.$$

Odpowiedni rachunek

```
Pi Integrate[y[t]^2 D[x[t], t], {t, 0, 2}]
```

prowadzi do wyniku $V = -\frac{64}{34}\pi$.

Oczywiście otrzymany wynik nie ma sensu – objętość nie może być przecież liczbą ujemną. Błąd który popełniliśmy łatwo jednak wyjaśnić. Parametr $t \in [0, 1]$ odpowiada x rosnącemu

od 0 do 1 wzdłuż dolnej gałęzi pętli, natomiast $t \in [1,2]$ malejącemu x od 1 do 0 wzdłuż górnej gałęzi. Wobec tego obliczona całka obejmuje objętość bryły otrzymanej przez obrót dolnej gałęzi z plusem i objętość analogicznej bryły dla górnej gałęzi z minusem. W świetle tych spostrzeżeń łatwo stwierdzić, że $-\frac{64}{34}\pi$ to minus objętość.
Zajmijmy się teraz obliczaniem pola powierzchni rozważanej bryły. Przypomnijmy, że pole powierzchni bryły powstałej przez obrót wykresu funkcji $y(x)$ nad $[a,b]$ wokół osi x wyraża się wzorem

$$S = \pi \int_a^b y(x)\sqrt{1 + y'(t)^2}\mathrm{d}t.$$

Czytelnik może łatwo sprawdzić, że podstawienie dane przez parametryzację $(x(t), y(t))$ prowadzi do wzoru

$$S = \pi \int_{t_0}^{t_1} y(t)\sqrt{x'(t)^2 + y'(t)^2}\mathrm{d}t.$$

Numeryczna wartość tej całki dla parametryzacji (7)

```
Pi NIntegrate[y[t] Sqrt[D[x[t], t]^2 + D[y[t], t]^2], {t, 0, 2}]
```

wynosi około 10.0269π.

Problemy do samodzielnego rozwiązania:

Zadanie 5.8. Obliczyć całki $\int_0^{2\pi} \sqrt{r(\phi)^2 + r'(\phi)^2}\mathrm{d}\phi$ i $\int_0^{2\pi} r(\phi)^2\mathrm{d}\phi$ dla krzywych zadanych parametrycznie: okręgu $r(\phi) = r_0$ i kardioidy $r(\phi) = 1+\cos(\phi)$. Porównać wyniki ze znanymi długościami i polami tych krzywych. Spróbuj wykazać, że powyższe wzory opisują długość krzywej i pole zakreślane przez każdą krzywą zamkniętą daną w postaci parametrycznej $r = r(\phi)$.
Zadanie 5.9. Sprawdzić, że kardioidę można opisać następującym równaniem

$$(x^2 + y^2)^2 + 2x(x^2 + y^2) = y^2.$$

Można użyć programu *Mathematica* do wyliczenia y jako funkcji x i porównania ze wzorem wyliczonym wcześniej. Można również sprawdzić, że punkty zadane biegunowo $r = 1 + \cos(\phi)$ spełniają powyższe równanie. Narysować wykres krzywej zadanej w powyższy sposób korzystając z funkcji `ContourPlot` (por. rozdział 3.1).
Zadanie 5.10. Obliczyć objętość i pole powierzchni bocznej torusa powstałego przez obrót okręgu $(x - R)^2 + y^2 = r^2$ wokół osi OY. Zakładamy, że $R > r$. =

Literatura

[1] Å. Björck and G. Dahlquist. *Metody numeryczne*. PWN, 1987.

[2] Wikipedia: Comparison of computer algebra systems.
http://en.wikipedia.org/wiki/Comparison_of_computer_algebra_systems.

[3] K.J. Falconer. *Techniques in fractal geometry*. Wiley, 1997.

[4] J.M. Jankowscy and M. Dryja. *Przegląd metod i algorytmów numerycznych, tom I i II*. Biblioteka inżynierii oprogramowania. WNT, 1995.

[5] Mathematica Documentation.
http://reference.wolfram.com/.

[6] Mathworld: Root-Finding Algorithm.
http://mathworld.wolfram.com/Root-FindingAlgorithm.html.

[7] Maxima Manual.
http://maxima.sourceforge.net/docs/manual/en/maxima.html.

[8] A. Ralston. *Wstep do analizy numerycznej*. PWN, 1971.

[9] WolframAlpha.
http://www.wolframalpha.com/.