Projekt z Maxima

Davyd Antoniuk

21 listopada 2023

Spis treści

Wziąłem funkcję: $g(x) = \frac{3}{x^2-4}$
Pochodna tej funkcji to: $g'(x) = -\frac{6x}{(x^2-4)^2}$
Druga pochodna to: $g''(x) = \frac{18x^2+24}{(x^2-4)^3}$

1 Dzedzina funkcji

Kod:

```
roots: solve(x^2-4=0, x);
print("Dziedzina: R \\ {", roots, "}");
```

Komentarz:
Kod znajduje pierwiastki równania kwadratowego $x^2 - 4 = 0$ przy użyciu funkcji *solve*. Następnie wyświetla informację o dziedzinie równania, która jest równa zbiorowi liczb rzeczywistych R z wyłączeniem pierwiastków znalezionych przez funkcję *solve*.
Wynik:

Dziedzina: R \ { [x = −2, x = 2] }

2 Punkty przecięcia z OX

Kod:

```
punkty_ox: solve(g(x)=0,x);
if ( length ( punkty_ox ) < 1) then
   print (" Nie ma punktów przecięcia z OX ")
else
   print (" Punkty przecięcia z OX :" , punkty_ox );
```

Komentarz:
Kod oblicza punkty przecięcia funkcji $g(x)$ z osią OX (oś pozioma) poprzez rozwiązanie równania $g(x) = 0$ przy użyciu funkcji *solve*. Jeśli nie zostanie znaleziony żaden punkt przecięcia (długość listy $punkty_o x$ jest mniejsza niż 1), wyświetlany jest komunikat informujący o braku punktów przecięcia z osią OX. W przeciwnym razie, wyświetlane są znalezione punkty przecięcia z osią OX. Kod ten pozwala sprawdzić, czy funkcja $g(x)$ przecina oś OX i, jeśli tak, podaje współrzędne tych punktów.
Wynik:

Nie ma punktów przecięcia z OX

3 Punkty przecięcia z OY

Kod:

```
punkty_oy: g(0);
print("Punkt przecięcia z OY:", punkty_oy);
```

Komentarz:
Kod oblicza wartość funkcji $g(x)$ w punkcie $x = 0$, co jest równoważne z punktem przecięcia z osią OY (oś pionowa). Wynik tego obliczenia jest przypisywany do zmiennej $punkty_oy$. Następnie, za pomocą instrukcji print, wyświetlany jest komunikat zawierający wartość punktu przecięcia z osią OY.
Wynik:

Punkt przecięcia z Oy: $-\left(\frac{3}{4}\right)$

4 Ekstrema lokalne

Kod:

```
extrema: solve(diff(g(x), x) = 0, x);

if length(extrema) > 0 then (
   for i: 1 thru length(extrema) do (
      extrema_znacz: g(extrema[i]),
      druga_poch: diff(g(x), x, 2),
      druga_poch_znacz: subst(x = extrema[i], druga_poch),

      if rhs(druga_poch_znacz) < 0 then (
         print("Maksimum lokalne w punkcie ", extrema[i],
               ": ", rhs(extrema_znacz))
      )
      elseif rhs(druga_poch_znacz) > 0 then (
         print("Minimum lokalne w punkcie ", extrema[i],
               ": ", rhs(extrema_znacz))
      )
      else (
         print("Punkt przecięcia w punkcie ", extrema[i],
               ": ", rhs(extrema_znacz))
      )
   )
)

else (
print("Funkcja nie ma ekstremów")
);
```

Komentarz:

Kod znajduje ekstrema funkcji $g(x)$ poprzez rozwiązanie równania różniczkowego $diff(g(x), x) = 0$. Dla każdego ekstremum oblicza wartość funkcji $g(x)$ w tym punkcie oraz drugą pochodną i jej wartość. Na podstawie znaku drugiej pochodnej identyfikuje, czy jest to maksimum lokalne, minimum lokalne czy punkt przecięcia. Wyświetla wyniki lub informację o braku ekstremów.
Wynik:

Maksimum lokalne w punkcie $x = 0$: $-\left(\frac{3}{4}\right)$

5 Asymptoty pionowe, poziomowe, ukośne

5.1 Asymptoty pionowe

Kod:

```
pionowe_asymptoty: roots;
if length(pionowe_asymptoty) > 0 then (
   print("Pionowe asymptote:", pionowe_asymptoty)
)
else (
   print("Funkcja nie ma pionowych asymptot")
);
```

Komentarz:
Kod oblicza pionowe asymptoty funkcji i wyświetla je, jeśli istnieją. W przeciwnym razie informuje o braku pionowych asymptot.
Wynik:

Pionowe asymptote: $[x = -2, x = 2]$

5.2 Asymptoty poziomowe, ukośne

```
asympt_poz_prawo: limit(g(x), x, infinity);
asympt_poz_lewo: limit(g(x), x, -infinity);

if is(infinity = asympt_poz_prawo) and
   is(infinity = asympt_poz_lewo) then (
   a: limit(g(x)/x, x, infinity),
   b: limit(g(x) - a*x, x, infinity),
   if is(a = infinity) and is(b = infinity) then (
      print("Funkcja nie ma asymptot ukosnych")
   )
   else (
      y: a * x + b,
```

```
      print("g(x) ma asymptotę ukośną obustronną y =", y)
   )
)
elseif (is(infinity = asympt_poz_prawo) and
        not -infinity = asympt_poz_lewo) or
        (is(infinity = asympt_poz_prawo) and
        -infinity = asympt_poz_lewo) then (
   print("F(x) ma asymptotę prawostronną od",
         asympt_poz_prawo, "do +nieskończoność")
)
elseif (is(infinity = asympt_poz_lewo) and
        not -infinity = asympt_poz_prawo) or
        (is(infinity = asympt_poz_prawo)
        and -infinity = asympt_poz_lewo) then (
   print("F(x) ma asymptotę lewostronną  od - nieskończoność do",
         asympt_poz_lewo)
)
else (
   if is(asympt_poz_prawo = asympt_poz_lewo) then (
      print("Poziomowa obustronna asymptota: y =", asympt_poz_prawo)
   )
   else (
       print("Funkcja ma asymptotę lewostronną od - nieskończoność do",
             asympt_poz_lewo,  "i asymptotę prawostronną od",
             asympt_poz_prawo, "do    +nieskończoność")
   )
);
```

Komentarz:
Kod sprawdza istnienie i charakter asymptot funkcji. W przypadku asymptoty poziomej, oblicza jej równanie. Jeśli funkcja ma asymptoty ukosne, oblicza ich równania i wyświetla. Jeśli istnieje asymptota prawostronna, informuje o jej przedziale. Analogicznie dla asymptoty lewostronnej. Jeśli istnieje asymptota pozioma, wyświetla jej równanie. W przeciwnym razie, jeśli funkcja ma zarówno asymptotę lewostronną, jak i prawostronną, informuje o przedziałach.
Wynik:

```
Pozioma obustronna asymptota: y =  0
```

6 Granice na krańcach dziedziny, i dla elementów poza dziedziną

Kod:

```
print("Granica na - nieskończoność:", limit(g(x), x, -inf));
print("Granica na +nieskończoność:", limit(g(x), x, inf));
```

```
for i: 1 thru length(roots) do (
   if (is(limit(g(x), x, roots[i], plus) = limit(g(x),
       x, roots[i], minus))) then
      print("Granica w punkcie", roots[i], "istnieje")
   else
      print("Granica w punkcie", roots[i], "nie istnieje")
);
```

Komentarz:
Kod oblicza granice funkcji w nieskończoności oraz w punktach zerowych. Sprawdza, czy granica istnieje dla każdego z tych punktów. Jeśli granica w danym punkcie istnieje, jest wyświetlana informacja o tym fakcie. W przeciwnym razie, informuje się, że granica w danym punkcie nie istnieje.
Wynik:

Granica na − nieskończoność: 0 Granica na +nieskończoność: 0

Granica w punkcie x = −2 nie istnieje

Granica w punkcie x = 2 nie istnieje

7 Monotoniczność

Kod:

```
pkt_mononotoniczne: roots$
pochodna_zero: realpart(solve(diff(g(x), x)=0, x))$
pkt_mononotoniczne: append(pkt_mononotoniczne, pochodna_zero)$
pkt_mononotoniczne: sort(listify(setify(pkt_mononotoniczne)),
lambda([p1, p2], rhs(p1) < rhs(p2)))$
for i: 1 thru length(pkt_mononotoniczne) do (
   pkt_przedzialow: subst(x = pkt_mononotoniczne[i] - 0.1,
   diff(g(x), x)),
   if (is(pkt_mononotoniczne[i] = pkt_mononotoniczne[1])) then (
      if (rhs(pkt_przedzialow) > 0) then
         print("Rosnąca na przedziale od - nieskończoność do "
               , pkt_mononotoniczne[i])
      else
         print("Malejąca na przedziale od - nieskończoność
               do ",pkt_mononotoniczne[i])
   )
   else (
      if (rhs(pkt_przedzialow) > 0) then
         print("Rosnąca od ", pkt_mononotoniczne[i - 1],
               " do ", pkt_mononotoniczne[i])
      else
```

```
            print("Malejąca od ",pkt_mononotoniczne[i - 1],
                  " do ", pkt_mononotoniczne[i])
        )
)$
if (rhs(subst(x = pkt_mononotoniczne[length(pkt_mononotoniczne)] + 0.1,
    diff(g(x), x))) > 0) then
   print("Rosnąca od ", pkt_mononotoniczne[length(pkt_mononotoniczne)],
         " do  +nieskończoność")
else
   print("Malejąca od ",
          pkt_mononotoniczne[length(pkt_mononotoniczne)],
          " do  +nieskończoność");
```

Komentarz:
Ten kod znajduje punkty monotoniczności funkcji oraz określa, czy funkcja rośnie czy maleje na poszczególnych przedziałach. Najpierw oblicza się punkty, w których pochodna funkcji wynosi zero, a następnie sortuje się je rosnąco. Następnie w pętli sprawdza się wartości pochodnej przed i po każdym punkcie monotonicznym, aby określić, czy funkcja rośnie czy maleje. Na końcu sprawdza się, czy funkcja rośnie lub maleje w przedziale od ostatniego punktu monotonicznego do nieskończoności. Wyświetla się informacje o monotoniczności na poszczególnych przedziałach oraz w ostatnim przedziale.
Wynik:

Rosnąca na przedziale od – nieskończoność do $x = -2$

Rosnąca od $x = -2$ do $x = 0$

Malejąca od $x = 0$ do $x = 2$

Malejąca od $x = 2$ do +nieskończoność

8 Wklęsłość / wypukłość funkcji

Kod:

```
pochodna_zero : realpart ( solve ( second_der = 0 , x )) $
pochodna_zero : sort ( append ( pochodna_zero , roots )) $
for i : 1 thru length (pochodna_zero ) do
(
   pkt_przedzialow : subst ( x =pochodna_zero [ i ] -0.1 ,
   diff ( g ( x ) ,x ,2)) ,
   if ( is ( pochodna_zero [ i ]= pochodna_zero [1])) then
   (
      if ( rhs ( pkt_przedzialow ) >0) then
         print (" Wypukla od - nieskończoność do " ,
```

```
                pochodna_zero [ i ])
         else
            print (" Wklęsla od - nieskończoność do " ,
                pochodna_zero [ i ])
      )
      else
      (
         if ( rhs ( pkt_przedzialow ) >0) then
            print (" Wypukla od " , pochodna_zero [i -1] ," do " ,
                pochodna_zero [ i ])
         else
            print (" Wklęsla od " , pochodna_zero [i -1] ," do " ,
                pochodna_zero [ i ])
      )
   ) $
   ostatni_element : poch_zero [ length ( poch_zero )] $
   if ( rhs ( subst ( x = ostatni_element +0.1 ,
       diff ( g( x ) ,x ,2))) >0) then
      print (" Wypukla od " , ostatni_element ,
            " do + nieskończoność ")
   else
      print (" Wklęsla od " , ostatni_element , " do  + nieskończoność");
```

Komentarz:
Ten kod znajduje punkty, w których druga pochodna funkcji wynosi zero lub punkty ekstremalne, a następnie określa, czy funkcja jest wypukła czy wklęsła na poszczególnych przedziałach. Najpierw oblicza się punkty, w których druga pochodna funkcji wynosi zero i łączy się je z wcześniej znalezionymi punktami ekstremalnymi. Następnie w pętli sprawdza się wartości drugiej pochodnej przed i po każdym punkcie, aby określić, czy funkcja jest wypukła czy wklęsła. Na końcu sprawdza się, czy funkcja jest wypukła lub wklęsła w przedziale od ostatniego punktu do nieskończoności. Wyświetla się informacje o krzywiźnie na poszczególnych przedziałach oraz w ostatnim przedziale.
Wynik:

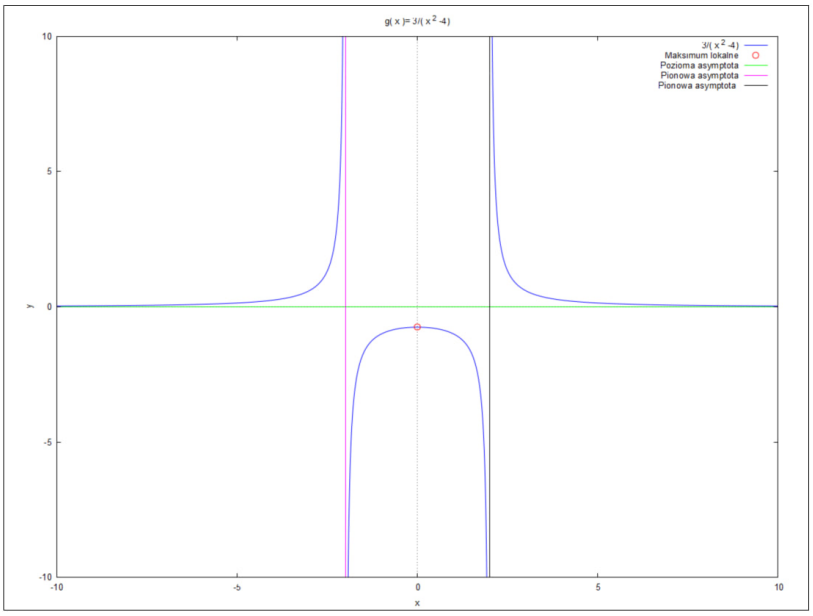
Wypukla od – nieskończoność do $x = -2$
Wklęsla od $x = -2$ do $x = 2$
Wypukla od $x = 2$ do + nieskończoność

9 Wykres

Kod:

```
wxplot2d ([ g ( x ) ,[ discrete , [ rhs ( extrema )] ,
[ g ( rhs ( extrema ))]] ,
asympt_poz_prawo ,
[ parametric , rhs ( pionowe_asymptoty [1]) , t , [t , -100 , 100]] ,
[ parametric , rhs ( pionowe_asymptoty [2]) , t , [t , -100 , 100]]] ,
[x , -10 ,10],
[y , -10 ,10],
[ style , lines , points , lines , lines , lines ] ,
[ legend , "3/( x ^2 -4)" , "Maksimum lokalne" ,
"Pozioma asymptota" ,"Pionowa asymptota" , "Pionowa asymptota "] ,
[ point_type , circle ] ,
[ xlabel , " x "] ,
[ ylabel , " y "] ,
[ title , "g( x )= 3/( x ^2 -4)"]
) $
```

Komentarz:
Ten kod generuje wykres funkcji $g(x)$ oraz dodaje punkty ekstremalne, asymptoty poziome i pionowe. Wykres jest ograniczony do przedziału od -10 do 10 dla osi x i y. Punkty ekstremalne są oznaczone jako punkty, asymptoty poziome jako linie, a asymptoty pionowe jako linie parametryczne. Wykres zawiera legendę oraz etykiety osi x i y.
Wynik:



Rysunek 1: Wykres funkcji $g(x) = \frac{3}{x^2-4}$