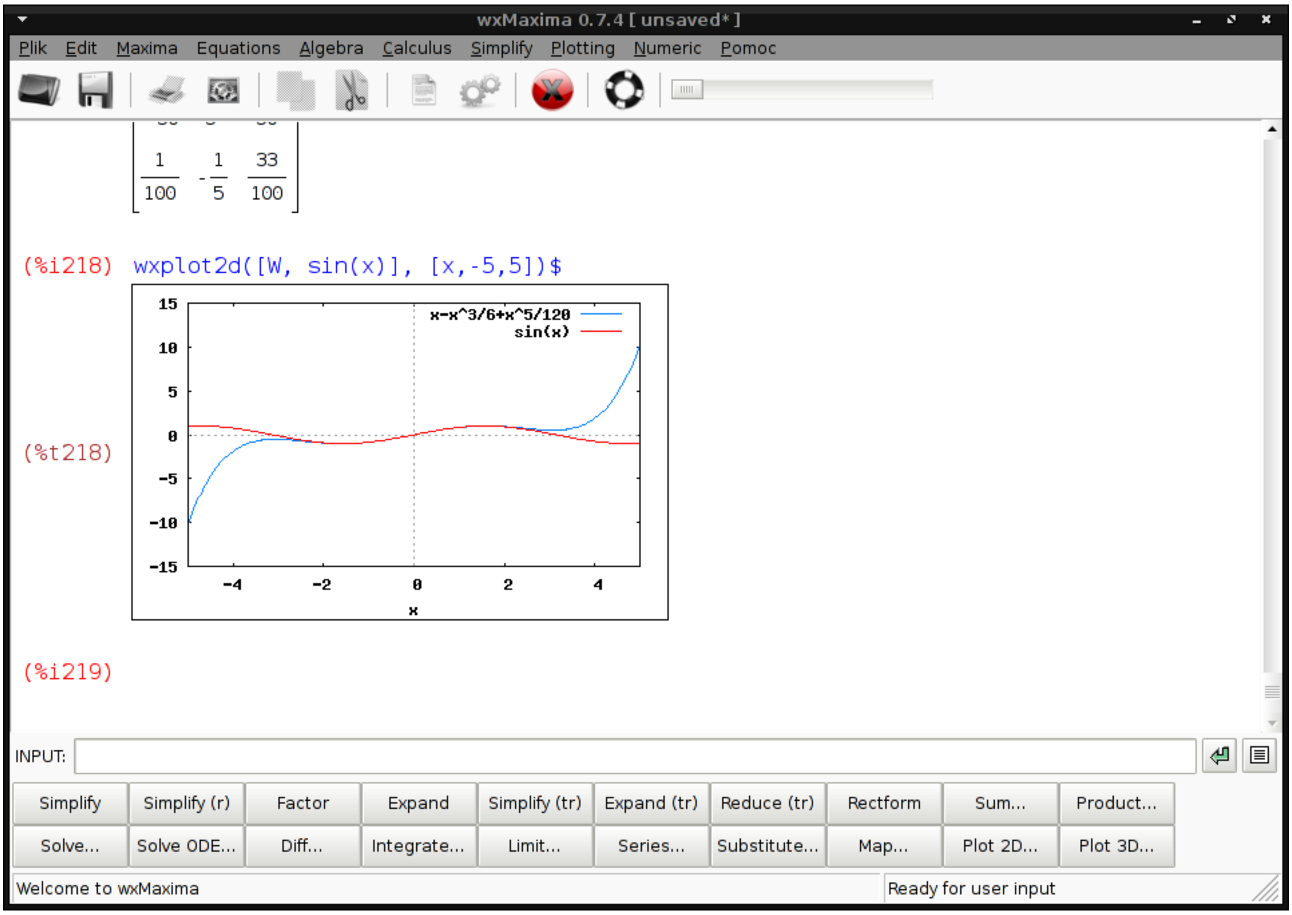
wxMaxima

Rafał Topolnicki
rafal.topolnicki@dragonia.pl
www.rtopolnicki.dragonia.pl

Jest rok 1998. William Shelter uzyskuje pozwolenie na publikacje kodu źródłowego programu Macsyma na licencji GPL. Przy okazji zmienia nazwę programu na Maxima. Od tego czasu minęło 10 lat w ciągu których Maxima przeżyła burzliwy rozwój. Ale po kolei. Macsyma, rozwijana w MIT od końca lat 60tych, jest programem typu CAS (Computer algerba system), służącym do wykonywania obliczeń symbolicznych. W dużym skrócie oznacza to, że program potrafi wykonywać obliczenia nie tylko na liczbach ale również na symbolach, przez co spektrum jego zastosowań znacząco rośnie. Wśród wielu funkcji programu znajdziemy te związanie z algebrą - operacje na wektorach, macierzach i listach, rozwiązywanie równań oraz układów równań, jak i analizą matematyczną – obliczanie pochodnej, całkowanie, rozwijanie w szereg Taylora, znajdowanie granic funkcji, znajdowanie NNW i NWD oraz wiele innych. Dodatkowo do dyspozycji mamy wiele narzędzie do upraszczania zapisu. Jeżeli dodać do tego obliczanie sum i iloczynów skończonych jak i nieskończonych, rysowanie wykresów funkcji (za pomocą gnuplota) oraz język programowania, uzyskujemy potężna narzędzie niezwykle pomocne zarówno na studiach jak i w późniejszej pracy. Wszystko za darmo i zapakowane w intuicyjny interfejs.



wxMaxima - przykładowa sesja

Czy jest tytułowa wxMaxima? Sama Maxima narzędziem działającym w trybie tekstowym dlatego jej obsługa może czasami przysparzać problemów. wxMaxima jest nakładką graficzną, która umożliwia wygodną pracę oraz nie zmusza nas do nauki i pilnowania składni. Oba programu prawie na pewno znajdują się w repozytoriach twojej dystrybucji. Teraz jest najwyższy czas na instalacje. U

mnie na systemie Archlinux po uruchomieniu wxMaximy pojawiała się informacja o niemożności połączenia się z Maximą aby temu zaradzić wystarczy dodać do pliku /etc/rc.conf w ustawieniach sieci linię
lo="lo 127.0.0.1"

Podstawy podstaw

Nie sposób w tak krótkim artykule przedstawić wszystkich możliwości tego programu dlatego chciałbym zasygnalizować jedynie te najważniejsze. Zaraz po uruchomieniu na ekranie oprócz komunikatu powitalnego pojawi się znak zachęty (%i1) oznaczający że Maxima czeka na pierwsze dane lub polecenia. Z początku użyjemy programu jak kalkulatora, wpisując w okienku tekstowym, po INPUT, 2+1, oto co ujrzymy na ekranie:
(%i1) 2+1;
(%o1) 3
%in to n-te wejścia (od INPUT) a %on – n-ty wyjście (od OUTPUT). W dalszej części artykułu niektóre wyjścia będą pomijane. Wynik ostatniego działania przechowuje zmienna %, dlatego aby podnieść wynik dodawania 2 i 1 do kwadratu wpisujemy %^2 lub %i1^2 lub wreszcie %o1^2 na każdym razem w odpowiedzi uzyskamy 9 (^ można stosować wymiennie z **). Operatory matematyczne wyglądają tak samo jak w większości języków programowania. Ponieważ Maxima jest „kalkulatorem" dowolnej precyzji dlatego obliczenie 100! nie jest większym problemem (display_ascii)

Maxima przechowuje najczęściej używane stałe matematyczne takie jak π, e czy *i*. Odwołujemy się do nich poprzez % (%pi, %e, %i). Funkcje matematyczne wyglądają tak samo jak na całym świecie, oczywistym jest więc sin(x), cosh(x), atan(x) (=arctg(x)), log(x) (=ln(x)), sqrt(x), exp(x) (=e^x) itd. Ważne jest aby nie stosować skróconego zapisu mnożenia – należy pisać 3*x a nie 3x.
wxMaxima stara się wykonywać obliczenia dokładnie dlatego wyniki nie są podawane w przybliżeniu lecz w sposób bardziej matematyczny, przykładowo:
i: sin(%pi/4)
o: $\frac{1}{\sqrt{(2)}}$
Aby podać wynik numeryczny wystarczy nacisnąć Ctrl+F
o: .7071067811865475

Możemy używać zmiennych oraz funkcji. W wypadku tych pierwszych operator przypisania ma postać : w wypadku drugich :=. Przykładowo:
i: b:7
i: a:2
i: a+b
o: 9
sqrt(a-b)
$\sqrt{(5)}\%i$
W ostatnim przykładzie, słusznie otrzymaliśmy wynik w ciele liczb zespolonych.
Utwórzmy następujące funkcje:

$$f(x)=\frac{x}{\sqrt{\ln(x)}} \quad g(x)=\frac{1}{\cos(3x)} \quad h(x)=\sqrt{(x^x+x)}$$

i: f(x):=x/sqrt(log(x))
i: g(x):=1/cos(3*x)
i: h(x):=sqrt(x**x + x)

Aby obliczyć wartość funkcji h(x) dla argumentu np. 5 wystarczy napisać
i: h(5)
o: $\sqrt{3130}$
Jeżeli interesuje nas przybliżona wartość, tak jak poprzednio wystarczy wcisnąć Ctrl+F lub od razu policzyć h(5.0). Użycie ułamka dziesiętnego zamiast zwykłego sygnalizuje Maximie iż interesuje nas dziesiętna postać wyniku. Jak nie trudno się domyślić funkcje możemy składać – zobaczmy co uzyskamy po przeliczeniu np. h(h(2)) czy czegoś bardziej skomplikowanego g(h(-2+f(3))).
Utwórzmy teraz jakaś przydatną funkcje – logarytm dziesiętny:
i: log10(x):=log(x)/log10
Szybkie sprawdzenie poprawności:
i: log10(10)=1
Wszystko działa.
Szczególnie żmudne bywa szukanie dzielników danej liczby nie mówiąc już o dzielnikach wielomianów. Nie ma sensu się ograniczać, więc znajdźmy dzielniki 35!: i: factor(35!). Wynik jest spektakularny ale tak naprawę każdy byłby go, po odpowiednio długim czasie, w stanie podać. Nie można tego powiedzieć o factor(35!+1) – Maxima radzi sobie i z tym. Poszukajmy dzielników wielomianów, zasada jest ta sama:
i: factor(x**5+6*x**4+10*x**3-11*x-6)
o: (x-1)*(x+1)^2*(x+2)*(x+3)
Do „rozwijania” wielomianów możemy używamy polecenia expand:
i: expand((x+1)^10)
o: $x^4+4x^3+6x^2+4x+1$
Maxima oferuje również inne funkcje rozwijające i upraszczające zapis z wykorzystaniem funkcji trygonometrycznych: trigexpand, trigsimp, trigreduce.

Menu

Przejdźmy teraz do omawiania menu programu:
Calculus
Tutaj szczególnie często będziemy zaglądać rozwiązując zadania z analizy.
-Integrate:
Obliczamy całki oznaczone jak i nieoznaczone. W wypadku tych pierwszych podajemy granice całkowania, w wypadku gdy są to „specjalne” wartości, takie jak e, π czy nieskończoności, możemy skorzystać z menu. Dostępne są również dwa algorytmy całkowania numerycznego, można z nich skorzystać gdy „normalna” metoda nie przynosi skutku. Policzmy więc całkę nieoznaczoną funkcji g(x) w poprzednich przykładach: Calculus-Integrate. W polu *Integrate* wpisujemy g(x). W polu *by variable* określamy co jest zmienną całkowania – w tym wypadku x. Resztę pozostawiamy bez zmian. Otrzymany wynik możemy uprościć wybierając przycisk Simplify (tr) w dolnego menu, co jest równoznaczne z użyciem funkcji trigsimp. W wypadku całek oznaczonych, postępujemy identycznie i dodatkowo podajemy granice całkowania.
W tym miejscu widać po raz pierwszy wyższość komercyjnych programów takich jak np. Mathematica. Często zdarza się, że radzi sobie ona z przykładami z przy których Maxima jest bezradna. W takim wypadku warto odwiedzić stronę [1].
-Differentiate
Tutaj obliczymy pochodną funkcji. W wielu wypadkach Maxima poda wynik w bardzo skomplikowanej postaci wtedy możemy skorzystać z opcji upraszczających zapis. Przykładowo:
i: diff(x^x,x)
o: $x^x(\log(x)+1)$
W okienku *times* możemy ustalić stopień szukanej pochodnej.
Jeżeli wynik podany przez Maxime jest inny od tego który sami otrzymaliśmy to albo popełniliśmy błąd w obliczeniach albo oba wyniki są poprawne lecz podane w innej postaci. Aby to sprawdzić

wystarczy odjąć swój wynik od tego podanego przez program – jeśli wyjdzie 0 to wszystko jest w porządku.

-Find limit

Szuka granic obu- lub jedno- stronnych funkcji w punkcie jak i w nieskończoności. Na początek coś prostego, definicja liczby e: $e = \lim_{x \to \infty} (1 + \frac{1}{x})^x$

i: limit((1+(1/x))^(x), x, inf);

o: %e

A teraz coś trudniejszego: $\lim_{x \to \infty} xt(1-t^2)^{x-1}$

i: limit(x*t*(1-t**2)**(x-1), x, inf);

Is |t^2-1| - 1 positive, negative, or zero?p;

Is (t-1)*(t+1) positive, negative, or zero?p;

o: infinity

Wartość granicy przy *x* zmierzającym do nieskończoności zależy od wartości parametru *t*, dlatego Maxima dopytuje się o jego wartość i w zależności od odpowiedzi (w przykładzie odpowiedzią było 2 razy positive) oblicza granicę.

Trzeba przyznać, że szukanie granic funkcji nie jest mocną stroną Maximy i w wielu przypadkach nie przyjdzie nam ona z pomocą.

-Get series

Rozwijanie funkcji wokół wybranego punktu w szereg Taylora dowolnego stopnia (regulowany przez parametr depth). Przykładowo, przybliżymy funckje g(x) wielomianem 10-stopnia w punkcie x=0: *Expression:* g(x) *variable:* x *around:* 0 *detph:* 10.

i: taylor(g(x), x, 0, 10);

o: $\frac{1+(9x^2)}{2} + \frac{135x^4}{8} + \frac{4941x^6}{80} + \frac{201933x^8}{896} + \frac{36829809x^{10}}{44800} + \ldots$

Warto zaznaczyć, że w przytoczonych przykładach wszędzie mogliśmy zamiast nazw funkcji używać ich wzorów.

Rysowanie wykresów

wxMaxima do rysowanie wykresów może korzystać z pomocy gnuplota i należy przyznać, że te dwa programu bardzo dobrze się uzupełniają. Przykładowo narysujmy wykres funkcji sin(ln(x)) w przedziale [0:10]. Z menu wybieramy Plotting – Plot 2D, jako expression podajemy sin(log(x)) oraz odpowiednio zakres zmiennej x. Jeżeli w Format: wybierzemy gnuplot wykres zostanie pokazany w nowym oknie, jeżeli natomiast zdecydujemy się na inline, wykres zostanie narysowany wewnątrz sesji wxMaximy. Podając wzór funkcji możemy oczywiście używać dotychczasowych wejść i wyjść (operator %) jak i zdefiniowanych funckji. Przykładowo, jeśli chcemy sprawdzić jak dokładne jest przybliżenie funkcji sin(x) szeregiem Maclaurina 5-ego stopnia:

i: taylor(sin(x), x, 0, 5);

o: $x - \frac{x^3}{6} + \frac{x^5}{120} + \ldots$

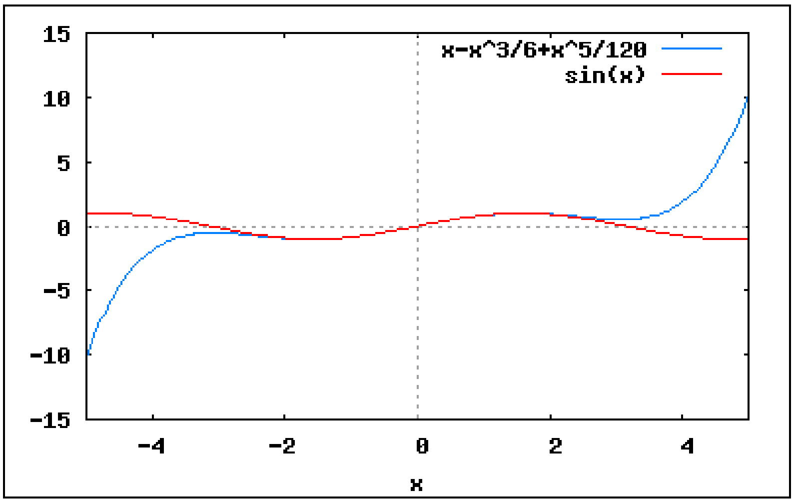
i: W:% (zapisanie wyniku do zmiennej W)

Plotting-Plot 2D, *Expression*: W, sin(x). Jeżeli na jednym wykresie chcemy umieścić kilka funkcji, oddzielamy je od siebie przecinkiem.

Możemy również pokusić się o wykresy trójwymiarowe. Tutaj też najlepiej korzystać z gnuplota. Powiedzmy, na początek, że interesuje nas wygląd jak wygląda funkcja $F(x,y) = \sqrt{|x^2 + y^2 - 8\pi e|}$. Definiujemy ją więc w Maximie:

i: F(x,y):=sqrt(abs(x**2+y**2-8*%pi*%e))

a następnie Plotting->Plot 3D i uzupełniamy tak jak na rysunku. Opcja gird ustala gęstość siatki natomiast włączenie pm3d spowoduje, że program oprócz siatki naniesie i odpowiednio pokoloruje również powierzchnię. Rysunek XXX przedstawia dodatkowo wykres funkcji sin(x)+cos(y). W oknie Plot 3D zamiast inline wybrano gnuplot.



Funkcja sinus oraz jej przybliżenie

Następną użyteczną funkcją jest Calculate Sum czyli obliczanie sum skończonych i nieskończonych. Z tej opcji możemy korzystać gdy musimy rozsądzić o zbieżności szeregu. Zaczniemy od szeregu geometrycznego: q+q^2+q^3 ....: *Sum of:* q^k *by variable:* k *from:* 1 *to:* inf. Zostaniemy poproszeni o odpowiedzenie czy |q|-1 jest dodatnie, ujemne czy równe 0. Gdy jest ujemne wynikiem będzie wzór na sumę szeregu geometrycznego. Teraz sprawdzimy ile wynosi $\sum_{k=1}^{\infty} (\frac{1}{k})^2$

i: sum((1/k)^2, k, 1, inf), simpsum;

o: $\frac{\%pi^2}{6}$

Wynik jest zgodny z oczekiwaniami. W identyczny sposób możemy wyznaczyć iloczyny.

Algebra

Jak zostało już powiedziane Maxima docelowo służy do obliczeń algebraicznych. W tym opisie skupimy się jednak tylko na tych najprostszych i najbardziej przydatnych. Zajmijmy się macierzami. Z menu Algebra wybieramy Enter Matrix, następnie podajemy wymiary macierzy, np 3x3. W menu type możemy wybrać rodzaj macierzy – normalna, symetryczna, diagonalna, antysymetryczna, my wybierzemy normalną a następnie wypełnimy ją kolejnymi liczbami:

i: matrix(
 [7,2,1],
 [8,7,4],
 [1,0,3]
);

o: $\begin{bmatrix} 7 & 2 & 1 \\ 8 & 7 & 4 \\ 1 & 0 & 3 \end{bmatrix}$

Nazwijmy tą macierz M,

i: M:%

Obliczymy jej wyznacznik. Wybieramy Algebra-Determinant

i: determinant(M)

o: 100
Znajdź teraz macierz odwrotną – Alebra-Invert matrix (lub polecenie invert(M)) a wynik przetransponujmy Alebra-Transpose matrix (lub polecenie transpose(M)). Jeżeli chcemy nazwać jakąś macierz możemy to uczynić na dwa sposoby, najpierw ją wprowadzić a następnie przypisać jej nazwę, tak jak zrobiliśmy to w wypadku macierzy M, - i: A:% lub też wpisać w polu Input A: a dopiero później wybrać z menu Enter matrix. Możenie macierzy odbywa się za pomocą . a nie *. Użycie tej drugiej spowoduje jedynie przemnożenie przez siebie odpowiadających sobie elementów obu macierzy. Tak jak zostało powiedziane już na początku, Maxima służy do wykonywania obliczeń symbolicznych, więc jeśli chcielibyśmy wyprowadzić ogólny wzór na iloczyn dwóch macierzy 3x3 przez siebie, wystarczy pierwszą z nich nazwać A i wypełnić elementami od a1 do i1 a drugą nazwać B i wypełnić elementami od a2 do i2 a następnie wykonać mnożenie A.B. Policzmy też iloczyn A i A^{-1}
i: A.invert(A)
otrzymamy straszny wynik, klikamy na przycisk Simplify i zgodnie z oczekiwaniami otrzymujemy macierz jednostkową.

Podobnie w przykładach np z całkowaniem śmiało możemy szukać wzorów ogólnych na funkcje zawierające dowolne stałe np: $\int \arcsin(\frac{a}{x})\,dx$ $\int arctg(px+1)\,dx$ $\int (\frac{m}{x})^{p}\,dx$. Identycznie w przypadku obliczania pochodnej czy szukania granic. Za każdym razem należy jednak zaznaczać co jest zmienną a co stała. Warto przy tym zaznaczyć, że funkcje arcus w języku angielskim to *asin acos atan actan.*

Maxima potrafi również rozwiązywać równania. Przykładowo rozwiążmy $x^3+x+2=0$. Z menu Equations wybieramy Solve i wpisujemy równanie, określamy niewiadomą (x) i po chwili mamy odpowiedź:
o: $\left[x=\frac{-(\sqrt{(7)}\%i-1)}{2,}\ x=\frac{(\sqrt{(7)}\%i+1)}{2,}\ x=-1\right]$ }{2},x=\frac{\sqrt{7}\,i+1}{2},x=-1]$$).
Ciekawskim polecam znaleźć pierwiastki równania $x^4+x+2=0$. Istnieją równania w których szukanie dokładnych rozwiązań jest niemożliwe, wtedy pomocne jest Solve numerically. Maxima potrafi również rozwiązywać układy równań. Jeśli do czynienia mamy z układem równań liniowych wybieramy Equations – Solve linear system np:
x+3y-2z=5
3x+5y+6z=7
2x+4y+3z=p
Wpisujemy równanie a w zmiennych wybieramy x,y,z (p jest parametrem). Natychmiast otrzymamy wynik:
o: [x=41-7*p,y=3*p-16,z=p-6]
W wypadku innych równań korzystamy z Equations-Solve algebraic system, obsługa wygląda identycznie.

Wydaje mi się, że przedstawiłem wszystkie najważniejsze możliwości programu. Warto jednak pamiętać, iż umie on znacznie, znacznie więcej. Na koniec warto odwiedzić stronę Maximy [2] oraz bardzo krótki przewodnik po niej [3] (ang)

[1] http://integrals.wolfram.com/index.jsp
[2] www.maxima.sourceforge.net
[3] http://math-blog.com/2007/06/04/a-10-minute-tutorial-for-solving-math-problems-with-maxima/