**ORODJE »REDUCE«**

**Mathematical tool »Reduce«**

**Marko Žugelj**

**marko.zugelj@student.fmf.uni-lj.si**

**Povzetek**

Z izdelavo seminarske naloge sem želel predstaviti orodje Reduce. Orodje Reduce sodi v skupino programov za simbolno računanje. Namenjen je predvsem delu s polinomi, poenostavljanju izrazov, analitičnem integriranju in odvajanju, reševanju enačb in delu z gama matrikami. Poleg osnovnega delovanja obstajajo še številni paketi, s katerimi lahko razširimo osnovno delovanje, tako da je orodje uporabno na številnih matematičnih vejah. Uporaba programa ni zahtevna, potrebno se je le naučiti sintakso, ki pa je precej sorodna sintaksi podobnih programov. Možnosti uporabe pa sem prikazal še z nekaj konkretnimi zgledi. Edina pomanjkljivost programa je po mojem mnenju ta, da je precej neznan in je posledično težje tudi do dodatne dokumentacije in pomoči.

**Ključne besede**

matematika, simbolno računanje, algebra, reševanje, analiza

**Abstract**

By making this seminar paper I wanted to present mathematical tool Reduce. Tool Reduce is one of the computer algebra systems. It was mainly designed to do operations on polynomials, simplify expressions, analytic integration and differentiation, solving equations and operations on Gamma matrices. Beside basic operations there are several packages which enable us to solve problems from variety of mathematical fields. Use of tool is not complicated, though we have to learn the syntax, which is quite similar to syntax of other tools like this. To present capabilities of tool I added a few concrete mathematical examples. The only defect of the tool in my opinion is, that it is rather unknown and it is quite hard to find additional documentation and help.

**Key words**

mathematics, computer algebra system, algebra, solving, calculus

**Uvod**

Že pračlovek se je, ko je hotel kaj narediti, najprej ozrl po okolici in poiskal nekaj, kar bi mu utegnilo biti v pomoč in olajšalo delo. In to lastnost smo ljudje ohranili vse do danes. Le da se danes ne oziramo več toliko v naravo, kot pa v informacijsko tehnologijo, ki se je v zadnjih letih zelo hitro razvija. Uporabljamo jo že skoraj na vsakem koraku, naj si bo to v srednji šoli, fakulteti, službi, ali pa kar doma. S časom pa se veča tudi število bolj ali manj uporabnih programov, ki so na voljo za končnega uporabnika.

Matematičnih izrazov že dolgo ne računamo več samo na roke, ampak so nam v pomoč številni programi, vendar je zaradi velikega števila le teh izbira primernega precej težka. Namen prispevka je predstavitev matematičnega orodja Reduce. Reduce je sistem za simbolno računanje in je kot tak uporaben za reševanje številnih problemov iz analize in algebre, pa tudi drugih matematičnih vej.

V prispevku bom predstavil osnovne lastnosti in značilnosti programa ter najosnovnejše ukaze. Potem si bomo še pogledali, kako naložimo pakete in na ta način občutno razširimo zmožnosti programa. Ogledali si bomo, kako rešujemo konkretne matematične probleme.

Glavni namen prispevka ni, da bi bralca prepričali, naj orodje namesti na svoj računalnik, ampak predstaviti ustrezne informacije, da bo bralec lahko sam presodil, ali mu je lahko orodje v pomoč pri nekem problemu.

**Predstavitev orodja**

**1. Osnovne značilnosti orodja**

Reduce je leta 1960 ustvaril Anthony C. Hearn. Napisan je v programskem jeziku Lisp. Deluje na operacijskih sistemih Microsoft Windows, Linux, Unix, Macintosh. Prvotno je bila njegova cena $695, vendar se je avtor odločil, da bo orodje kot brezplačno bolj konkurenčno plačljivim programom in se bo na ta način tudi nadalje razvijalo. Torej je od decembra 2008 brezplačno in se oddaja pod pogoji BSD licence.

Orodje je dostopno na spodnji spletni strani:

<http://reduce-algebra.sourceforge.net/>

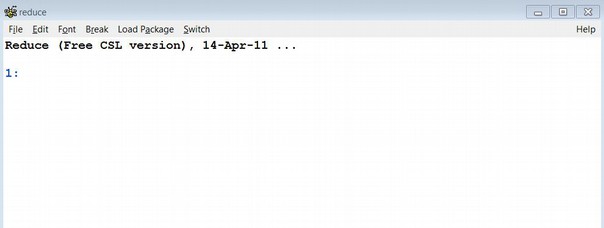
Kaj vse lahko z orodjem Reduce počnemo?

Kot že omenjeno sodi med programe za simbolno računanje in je kot tak namenjen predvsem delu s polinomi, poenostavljanju izrazov, analitičnem integriranju in odvajanju, reševanju enačb in delu z gama matrikami. Poleg osnovnih ukazov lahko delovanje razširimo še s številnimi paketi, ki omogočajo delo s seznami, vektorji, matrikami in linearno algebro, numerično matematiko, razvijanje v Taylorjevo vrsto, pakete za delo s tenzorji, limitami, diferencialnimi enačbami, parcialnimi diferencialnimi enačbami, Lijevo algebro, Laplacejevimi transformacijami, Fourierjevo transformacijo ...

Glede na to, da ima v sebi vgrajenih že veliko ukazov vendar znotraj programa ni možno pogledati lastnosti ukaza, kot je to možno v nekaterih drugih programih, je zelo praktično, da si pomagamo z dokumentacijo, ki je objavljena na internetni strani: <http://www.reduce-algebra.com/documentation.htm>. Tukaj je na voljo uporabniški vodič in pa dokumentacija za vsak paket posebej.

**2. Osnovni videz orodja**

Ko program namestimo in odpremo vidimo naslednje okno:



Slika 1: Osnovni zaslon

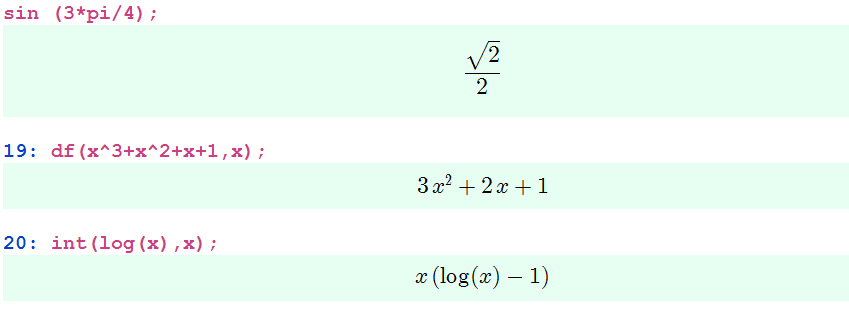
Ukaz File -> Save ne shrani pravilno skriptne datoteke, če ne pokličemo ustreznih ukazov. Za pravilno shranjevanje prepišemo ukaze iz uporabniškega vodiča v razdelku OUT Command. Lahko pa z istimi ukazi ustvarimo prazno datoteko, ki jo lahko odpremo in urejamo kar v beležnici, nato pa odpremo z ukazom File -> Read. Za samo delo v programu pa je najenostavneje, če uporabljamo tipkovnico, za kar pa se je potrebno malo potruditi, ker so nekateri ukazi na slovenski tipkovnici drugačni kot v meniju Edit. Tako na primer do zadnjega vnosa pridemo z ukazom CTRL-P in ne CTRL-^, kot to piše v meniju EDIT.

Paket naložimo iz menija Packages ali pa z ukazom load\_package »ime\_paketa«;.

**3. Reševanje nalog**

Osnovni ukazi

Osnovno delo s programom je precej enostavno. Najprej pa moramo omeniti določene posebnosti. Tako je na primer potrebno za vsakim ukazom zapisati ; ali pa $. Če napišemo $ se vrednost ne izpiše, v nasprotnem pa se vrednost ne izpiše. Program ne loči med malimi in velikimi črkami. Spremenljivki priredimo vrednost z ukazom :=. Program pozna osnovne operacije kot so +,-,\*,/,^. Ima vgrajene številne funkcije, katerim v okroglih oklepajih naštejemo argumente, ki jih ločimo z vejico, na primer: sin(kot), abs(število), odvod po spremenljivki x zapišemo kot df(izraz,x), podobno tudi integral int(izraz,x).



Slika 2: Osnovni ukazi

Sicer pa je v orodju še nekaj ukazov, ki jih je koristno vedeti, vendar bomo te predstavili kar na primerih.

3.1. Krivulja z enačbo y=4/x ima dve tangenti z naklonskim kotom 135°. Zapiši enačbi tangent. (Vir: Splošna matura 2008, spomladanski rok, 7 naloga)

Vemo: Odvod je tangens naklonskega kota, rešimo enačbo, zapišemo premici.

Koda:

y:= 4/x;

odvod:= df(y,x);

solve(odvod=tan(3\*pi/4),x);

x1:=2;

x2:=-2;

y1:= sub(x=x1,y);

y2:= sub(x=x2,y);

k1:= sub(x=x1, odvod);

k2:= sub(x=x2, odvod);

solve((t1-y1)/(x-x1)=k1,t1);

solve((t2-y2)/(x-x2)=k2,t2);

Najprej smo v y shranili predpis za y. Z ukazom df izračunamo odvod y po x. Rešujemo enačbo, ko je odvod enak tangensu 135° oz 3\*pi/4. Uporabimo ukaz solve, kateremu podamo enačbo (v resnici podamo seznam v zavitih oklepajih, vendar, ker je samo ena enačba to ni potrebno, ko imamo sistem enačb pa je uporaba zavitih oklepajev nujna) in pa neznanko, po kateri rešujemo enačbo (podobno kot prej, podamo seznam neznank). Dobimo seznam rešitev, katere shranimo v x1 in x2. Sedaj, ko imamo x, pa bi radi izračunali še vrednost y koordinate v tej točki, za kar uporabimo ukaz sub (ang. substitute). Ukazu sub najprej podamo novo vrednost, nato pa izraz. Če imamo: y1:=sub(x=x1,y), pomeni, da se v y1 shrani, kar dobimo, če v izrazu y x nadomestimo z x1. Na podoben način izračunamo še vrednosti odvoda v x1 in x2 (k1 in k2). Nato iz znane enačbe za linearno funkcijo izrazimo še enačbi premic t1 in t2, torej iskani rešitvi sta t1 in t2.

3.2 Dana sta vektorja a=[7,-1,-1] in b=[2,1,1]. Poišči taka vektorja c in d, da velja c+d=a, vektor c je vzporeden vektorju b, vektor d pa je pravokoten na b.

Ideja: Nalogo bomo rešili z uporabo paketa avector, ki nam omogoča delo z vektorji. Uporabimo lastnosti vektorjev in rešimo sistem enačb.

load\_package avector;

a := avec(7,-1,-1);

b := avec(2,1,1);

c:= k\*b;

d:= avec(d1,d2,d3);

vsota:= c+d;

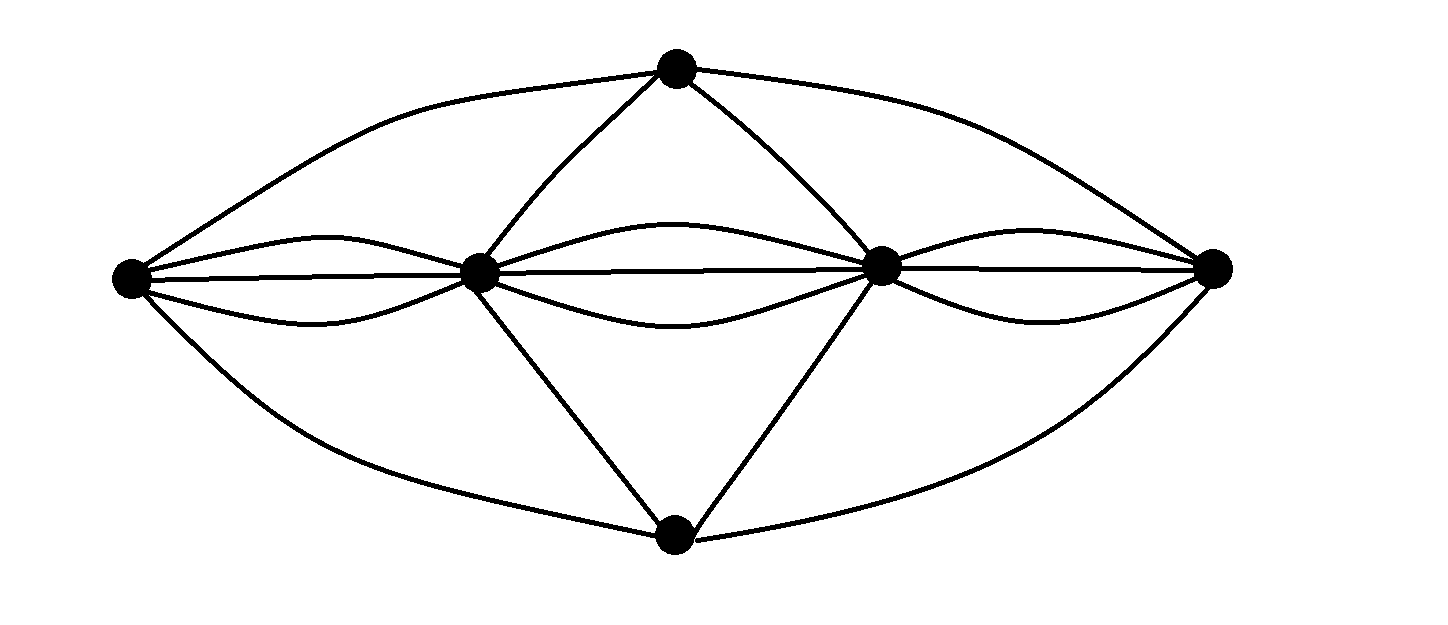
resitev:=first(solve({vsota(0)=a(0),vsota(1)=a(1), vsota(2)=a(2), dot(b,d)=0},{d1,d2,d3,k}));

c:=sub(part(resitev,4),k)\*b;

d:=avec(sub(part(resitev,1),d1),sub(part(resitev,2),d2),sub(part(resitev,3),d3));

Najprej naložimo paket avector. Shranimo vektorja a in b, tako da napišemo ukaz avec, nato pa naštejemo komponente. Vektor c je vzporeden vektorju b, zato c lahko zapišemo kot nek večkratnik vektorja b. Vektor d je neznan, torej ga shranimo s tremi neznanimi komponentami. Definiramo si pomožen vektor vsota, ki je vsota vektorjev c in d (vektorje seštevamo kar z znakom +). Sedaj imamo 4 enačbe in 4 neznanke, kar pomeni, da lahko rešimo sistem enačb. Prve tri enačbe primerjamo vektorja a in vsota po komponentah, pri čemer je a(0) prva komponenta vektorja a, a(1) druga in a(2) tretja komponenta. Za četrto enačbo pa uporabimo dejstvo, da sta vektorja pravokotna natanko tedaj, kadar je njun skalarni produkt enak 0. Skalarni produkt vektorjev b in d izračunamo z ukazom dot(b,d). Solve nam vrne seznam rešitev, posamezno rešitev pa dobimo z rezanjem seznama, torej part(sez, i) pomeni i-ta komponenta seznama. Sicer bi lahko rešitev »razbrali« že iz seznama, vendar z rezanjem seznama in substitucijami lahko potem konstruiramo nov vektor in vrnemo le-tega, kar je tudi bolj ustrezno.

3.2 Poišči število vpetih dreves grafa.



Slika 3: Graf

Ideja: Oštevilčimo vozlišča. Naredimo seznam vozlišč (i,j,k), kar pomeni, da je med vozliščema i in j k povezav. Definiramo ustrezno funkcijo. Uporabimo Lagrangevo metodo.

Koda:

load\_package linalg;

load\_package assist;

procedure um (sez,n);

begin matrix matrika, diagonala;

matrika:= band\_matrix(0,n);

diagonala:= band\_matrix(0,n);

for l:=1:length(sez) do

<< el:=part(sez,l);

g:= first(el);

h:= second(el);

k:= third(el);

matrika (g,h):= -k;

diagonala(h,h):=diagonala(h,h)+k;

diagonala(g,g):=diagonala(g,g)+k >>;

matrika:=tp(matrika)+matrika+diagonala;

return matrika;

end;

sez:={{1,2,1},{1,3,3},{3,4,3},{2,3,1},{2,4,1},{2,5,1},{4,5,3},{1,6,1},{3,6,1},{4,6,1},{5,6,1}};

matrika:=um(sez,6);

matrika:=remove\_columns(matrika,1);

matrika:=remove\_rows(matrika,1);

det(matrika);

Reduce nam omogoča pisanje lastnih funkcij, uporabo pogojnih stavkov in zank. Ker imamo opravka z matrikami naložimo paket linalg in paket assist za delo s seznami. Definiramo funkcijo, ki iz seznama vrne ustrezno matriko. To storimo z ukazom procedure ime\_funkcije (vhodni argumenti). Ker imamo v funkciji več ukazov moramo na začetek napisati begin, na konec pa end. Ker želimo shraniti lokalne spremenljivke iz zanke moramo rezervirati prostor, kar storimo tako, da na začetku za begin napišemo matrix (kar pomeni, da bosta spremenljivki tipa matrike) in pa imeni spremenljivk. Spremenljivki matrika (kamor bomo shranili ne-diagonalne) in diagonala (kamor bomo shranili diagonalne) sta na začetku n\*n matriki samih ničel, kar dobimo z ukazom band\_matrix(0,n). Z zanko se sprehodimo po seznamu in ustrezno spreminjamo elemente v matriki. Posebej za diagonalne in ostale. Vrnemo vsoto matrike, diagonale in transponirane matrike (ukaz tp(matrika), zaradi simetrije; med g in h je isto kot med h in g). Novo matriko dobimo tako, da pokličemo funkcijo, ki smo jo ravnokar napisali. Z ukazom remove\_columns(matrika, i) iz matrike odstranimo i-ti stolpec. Sicer lahko odstranimo poljubnega, vendar je najvarneje, če odstranimo prvega, saj ta vedno obstaja. Podobno z ukazom remove\_columns odstranimo prvo vrstico. Izračunamo še determinanto matrike, tako da uporabimo ukaz det(matrika).

3.4 Klasificiraj stacionarne točke funkcije f(x,y)=x4+4xy+y4+1

Ideja: Poiščemo stacionarne točke in z uporabo Hessejeve matrike klasificiramo točke.

Koda:

load\_package linalg;

procedure f(x,y);

x^4+4\*x\*y+y^4+1;

solve({df(f(x,y),y)=0,df(f(x,y),x)=0},{x,y});

b:=hessian(f(x,y),{x,y});

b1:=sub(x=1,y=-1, b);

det b1;

if (det b1 > 0 and b1(2,2) < 0) then write "MAKSIMUM" else if (det b1 > 0 and b1(2,2) > 0) then write "MINIMUM" else if (det b1 < 0) then write "SEDLO" else write "ne vemo";

Naložimo paket linalg, saj bomo potrebovali Hessejevo matriko. Funkcijo f shranimo kot svojo funkcijo. Poiščemo stacionarne točke, tako da rešimo sistem enačb, kjer sta oba parcialna odvoda enaka 0. Z ukazom hessian definiramo matriko b. Ukaz prejme funkcijo in seznam spremenljivk. S substitucijo izračunamo Hessejevo matriko v konkretni stacionarni točki iz seznama rešitev (zgoraj v (1,-1)) in izračunamo determinanto. S kombiniranjem pogojnih stavkov lahko poskrbimo za lep izpis rešitve. Pišemo z ukazom write.

**4. Mnenja drugih uporabnikov**

Na spletu je precej malo mnenj in člankov o orodju Reduce. Na spletni strani sourceforge.net je orodje prejelo 7 ocen, vsi pa so mu dali oceno 5. Večinoma uporabniki cenijo uprabnost in enostavno uporabo. Tako je uporabnik codeprese (2012) zapisal: »Very useful project!«, uporabnik nicolasperkins (2012) pa je zapisal »Easy to install and use«.

**Zaključek**

Med pisanjem seminarske naloge sem spoznal orodje Reduce. Spoznal sem osnovne ukaze in lastnosti ter nekatere možnosti uporabe. Orodje mi je bilo dokaj všeč. Program ima določene posebnosti, na katere se je potrebno navaditi, kar pa velja za večino programov. Sicer se s plačljivimi programi, kot je na primer Mathematica ne more primerjati niti vizualno niti funkcionalno, vendar je potrebno imeti v mislih, da je program brezplačen in menim, da lahko dobro opravi vlogo alternative. Mogoče je rahlo moteče, da je poleg programa še veliko paketov in je potrebno iskati še ustrezen paket in njegovo dokumentacijo. Če bi bili vsi paketi združeni v program in bi bila vsa dokumentacija na enem mestu, bi bilo lažje. Največja pomanjkljivost je pomanjkanje dodatne literature, saj program uporablja relativno majhno število ljudi, posledično je tudi težko najti pomoč na spletu, če se nam kje zatakne. Sicer pa je osnovna dokumentacija dobro napisana, sam nisem imel večjih težav, kljub temu, da sem program uporabljal prvič.

Orodje se mi zdi uporabno za vse, učence, dijake in študente. Lahko ga uporabljamo kot računalo in za računanje kotnih funkcij ter preprostih enačb v osnovni šol, integriranje in odvajanje v srednji šoli ali pa s paketi razširimo zmožnosti za uporabo na fakultetah ali za profesionalno uporabo.

**Viri**

1. codeprese (2012),[http://sourceforge.net/projects/reduce-algebra/reviews/?sort=created\_date&stars=0#reviews-n-ratings](http://sourceforge.net/projects/reduce-algebra/reviews/?sort=created_date&stars=0" \l "reviews-n-ratings) (ogled 18.3.2015)
2. [http://en.wikipedia.org/wiki/Reduce\_%28computer\_algebra\_system%29](http://en.wikipedia.org/wiki/Reduce_(computer_algebra_system)) (ogled 18.3.2015)
3. <http://reduce-algebra.sourceforge.net/> (ogled 18.3.2015)
4. <http://www.reduce-algebra.com/documentation.htm> (ogled 18.3.2015)
5. nicolasperkins (2012), [http://sourceforge.net/projects/reduce-algebra/reviews/?sort=created\_date&stars=0#reviews-n-ratings](http://sourceforge.net/projects/reduce-algebra/reviews/?sort=created_date&stars=0" \l "reviews-n-ratings) (ogled 18.3.2015)