lhs_equals_rhs_integral.wxm

GNU General Public License 2017 Stephen Athel Abbott.

A development document for Geometric Algebra with wxMaxima
to test the vector calculus functions within the GAwxM environment,
contains...
Initialization
Loading of functions (intrinsic and GA specific)
Pseudoscalar definition (specifies the space dimension) and
Calculation of the inverse pseudoscalar used to generate the dual of a multivector
Enumeration of the standard basis for the specified dimension

To confirm the equality of...
Integration over a region of the surface defined by an inverted (convex)
paraboloid using the (vector) derivative of a vector function on the surface (manifold)
i.e. the left hand side (lhs) of the fundamental theory of geometric calculus
with...
Integration over the boundary of a region of the surface defined by an inverted (convex)
paraboloid using the line integral of a vector function on a set of boundary curves
i.e. the right hand side (rhs) of the identity for the fundamental theory
because...
we would like to show a particular numerical example of the Fundamental Theorem;
this is the same example as that in document...lhs_equals_rhs_changevar.wxm but
with manual calculation of the new limits for a change of variable;

Initialization

```
(%i145) reset()$
        kill(all)$
        stardisp:true$
        stringdisp:true$
        noundisp:true$
        simp:true$
        dotdistrib:true$
        derivabbrev:true$
        lispdisp:true$
```

load intrinsic (maxima or lisp) function files

```
(%i8) load("basic")$
      load("facexp")$
      load("functs")$
      load("scifac")$
```

batchload GA specific (maxima) function files;
the initialization file, maxima-init.mac is in the /home directory
and contains the variable, maxima_userdir, and this allows the paths
in file_search_maxima to be extended to point to the GA functions
ldisplay(maxima_userdir)$

```
(%i12) batchload("gafns0")$
       batchload("gafns1")$
       batchload("gafns2")$
       batchload("gafns3")$
       batchload("gafns4")$
       batchload("gafns5")$
       batchload("gafns6")$
```

batchload GC specific (maxima) function files;
the paths in file_search_maxima are also extended to point to the GC functions

```
(%i19) batchload("gcfns1")$
       batchload("gcfns2")$
       batchload("gcfns3")$
```

the pseudoscalar and its inverse
the lowest useable dimension pseudoscalar should be e1~e2 i.e. Plen = 2
global parameters, Pseudos, Pvar[] and Plen, I and Ii

```
(%i22) Pseudos:{e1,e2,e3}$
       Pvar:listofvars(Pseudos)$
       Plen:length(Pvar)$
       I:Pseudos$
       ni:(Plen-1)*Plen/2$
       Ii:(-1)^ni*I$
       kill(ni)$
       ldisplay(Pvar,Plen,I,Ii)$
```

```
(%t29) Pvar=[e1,e2,e3]
(%t30) Plen=3
(%t31) I={e1,e2,e3}
(%t32) Ii=-{e1,e2,e3}
```

global array parameters
lstbases[], nbases[], maxnbases

```
(%i33) array(lstbases,Plen)$
       array(nbases,Plen)$
       eset:setify(Pvar)$
       for ng:1 thru Plen do
       block(nbases[ng]:combination(Plen,ng),
       lstbases[ng]:full_listify(powerset(eset,ng)))$
       maxnbases:(combination(Plen,floor(Plen/2)))$
       kill(eset,ng)$
```

global arrays nbases[] and lstblds[] are used for grader(M) in gafns4.wxm
lstblds[] is an array of lists of blades and allblds[] is a list of all blades

```
(%i39) array(lstblds,Plen)$
       for ng:1 thru Plen do
       block(lstb:lstbases[ng],
       lstblds[ng]:makelist(list2blade(lst),lst,lstb))$
       allblds:[]$
       for ng:1 thru Plen do
       block(allblds:append(allblds,lstblds[ng]))$
       ldisplay(allblds)$
       kill(lstb,ng)$
```

```
(%t43) allblds=[{e1},{e2},{e3},{e1,e2},{e1,e3},{e2,e3},{e1,e2,e3}]
```

find the inverse of the canonical basis as a global list, invblds[], like allblds

```
(%i45) invblds:[]$
       for k:1 thru Plen do
       block(ni:k*(k-1)/2,
       nk:(-1)^ni,
       lst:nk*lstblds[k],
       invblds:append(invblds,lst))$
       ldisplay(invblds)$
```

```
(%t47) invblds=[{e1},{e2},{e3},-{e1,e2},-{e1,e3},-{e2,e3},-{e1,e2,e3}]
```

end of Initialization

set derivabbrev:false$

```
(%i48) derivabbrev:false$
```

using the (M)anifold functions found in file gcfns3.wxm,
we must find the left hand side (lhs) of the integral identity

parameterize an inverted (convex) paraboloid surface and find the basis

```
(%i49) bx:u*{e1}+v*{e2}-(u*u+v*v)*{e3}$
       bu:diff(bx,u)$
       bv:diff(bx,v)$
```

find the reciprocal of the basis

```
(%i52) tgtbasis:['bu,'bv]$
       rectgt:reciprocM(tgtbasis)$
       b1:rectgt[1]$
       b2:rectgt[2]$
```

define the vector function on the surface...

```
(%i56) bf:u*bu$
```

form the vector derivative; "vector del" &* "vector bf" = bivector + scalar

```
(%i57) bfstr:"bf"$
       parlst:[u,v]$
       reclst:['b1,'b2]$
       ldisplay(parlst,reclst)$
       vectordelM(bfstr,parlst,reclst);
```

```
(%t60) parlst=[u,v]
(%t61) reclst=[b1,b2]
(%o62) b2&* diff(bf,v)+b1&* diff(bf,u)
```

```
(%i63) ev(%)$
       delbf:facsum(%,allblds)$
       ldisplay(delbf)$
```

(%t65) $delbf=-\frac{2*\{e1,e3\}*u*(8*v^2+1)-4*v^2-16*\{e2,e3\}*u^2*v-4*\{e1,e2\}*u*v-8*u^2-1}{4*v^2+4*u^2+1}$

this is the anticlockwise bivector tangent to the convex paraboloid

```
(%i66) bubv:bu&^bv$
       ldisplay(bubv)$
```

```
(%t67)/R/ bubv=2*{e2,e3}*u-2*{e1,e3}*v+{e1,e2}
```

now we need a double integral over u=[1,2],v=[-1,+1] with boundary curves u=1,2 and v=+/-1

first form the integrand, integ, of the left hand side (lhs) of the fundamental theorem
and grade the geometric product

```
(%i68) bubv&*delbf$
       facsum(%,allblds)$
       integ:grader(%)$
       deninteg:integ[Plen+1]$
       scalarinteg:integ[0]/deninteg$
       integ[2]/deninteg$
       bivecinteg:facsum(%,allblds)$
       ldisplay(scalarinteg,bivecinteg)$
```

```
(%t75)/R/ scalarinteg=-8*u*v
(%t76) bivecinteg=-2*{e1,e3}*v+4*{e2,e3}*u+{e1,e2}
```

recombine the two and display the double integral

```
(%i77) lhsinteg:scalarinteg+bivecinteg$
       J1:'integrate(lhsinteg,v)$
       J2:'integrate(J1,u)$
       ldisplay(J2)$
```

(%t80) $J2=\int\int(-8*u-2*\{e1,e3\})*v+4*\{e2,e3\}*u+\{e1,e2\}dv\,du$

for the inner part of the iterated integral

```
(%i81) J1lhs:'integrate(lhsinteg,v)$
       ldisplay(J1lhs)$
```

(%t82) $J1lhs=\int(-8*u-2*\{e1,e3\})*v+4*\{e2,e3\}*u+\{e1,e2\}dv$

using a functional method to evaluate at the limits

```
(%i83) expr:ev(J1lhs,nouns)$
       define(J1fuv(u,v),expr)$
```

```
(%i85) J1upper:J1fuv(u,+1);
       J1lower:J1fuv(u,-1);
       J2integ:J1upper-J1lower;
```

(%o85) $4*\{e2,e3\}*u+\frac{-8*u-2*\{e1,e3\}}{2}+\{e1,e2\}$

(%o86) $-4*\{e2,e3\}*u+\frac{-8*u-2*\{e1,e3\}}{2}-\{e1,e2\}$

(%o87) $8*\{e2,e3\}*u+2*\{e1,e2\}$

```
(%i88) J2lhs:'integrate(J2integ,u)$
       ldisplay(J2lhs)$
```

(%t89) $J2lhs=\int 8*\{e2,e3\}*u+2*\{e1,e2\}du$

to demonstrate an (unecessary) change of variable from u to y with f(u,y)=2*u+1-y=0

```
(%i90) J2ch:changevar(J2lhs,2*u+1-y,y,u)$
       ldisplay(J2ch)$
```

(%t91) $J2ch=\int 2*\{e2,e3\}*y-2*\{e2,e3\}+\{e1,e2\}dy$

```
(%i92) expr:ev(J2ch,nouns)$
       ldisplay(expr)$
       define(J2fy(y),expr)$
```

(%t93) $expr=\{e2,e3\}*y^2-2*\{e2,e3\}*y+\{e1,e2\}*y$

the limits u=1,2 become y=3,5

```
(%i95) J2upper:J2fy(5);
       J2lower:J2fy(3);
```

```
(%o95) 15*{e2,e3}+5*{e1,e2}
(%o96) 3*{e2,e3}+3*{e1,e2}
```

```
(%i97) J2upper-J2lower$
       ev(%,numer)$
       Jlhs:expand(%);
```

```
(%o99) 12*{e2,e3}+2*{e1,e2}
```

now we must find the right hand side (rhs) of the integral identity

the lhs needed a double integral over u=[1,2],v=[-1,+1] for the region, M
the rhs needs line integrals over four boundary curves, dM; u=1,2 and v=+/-1

the direction of the curves must be clockwise from above; here they are taken in sequence

u=1, v=[-1,+1]
v=1, u=[+1,+2]
u=2, v=[+1,-1]
v=-1,u=[+2,+1]

using the same parameterization as above

```
(%i100) exprbx:bx$
        define(bxf(u,v),exprbx);
```

(%o101) $bxf(u,v):=-\{e3\}*(v^2+u^2)+\{e2\}*v+\{e1\}*u$

and the same expression from above for the vector function on the surface

```
(%i102) exprbf:bf$
        define(bfuv(u,v),exprbf);
```

```
(%o103) bfuv(u,v):=u*({e1}-2*{e3}*u)
```

```
(%i104) bxf(1,v)$
        diff(%,v);
        integ1:%&*bfuv(1,v)$
        facsum(%,allblds)$
        Jp1:'integrate(%,v,-1,+1);
```

(%o105) $\{e2\}-2*\{e3\}*v$

(%o108) $\int_{-1}^{1} 2*\{e1,e3\}*v+4*v-2*\{e2,e3\}-\{e1,e2\}dv$

```
(%i109) bxf(u,1)$
        diff(%,u)$
        integ2:%&*bfuv(u,1)$
        facsum(%,allblds)$
        Jp2:'integrate(%,u,1,2);
```

(%o113) $\int_{1}^{2} u*(4*u^2+1)du$

```
(%i114) bxf(2,v);
        diff(%,v);
        integ3:%&*bfuv(2,v)$
        facsum(%,allblds);
        Jp3:'integrate(%,v,+1,-1);
```

(%o114) $-\{e3\}*(v^2+4)+\{e2\}*v+2*\{e1\}$

(%o115) $\{e2\}-2*\{e3\}*v$

(%o117) $4*\{e1,e3\}*v+16*v-8*\{e2,e3\}-2*\{e1,e2\}$

(%o118) $-\int_{-1}^{1} 4*\{e1,e3\}*v+16*v-8*\{e2,e3\}-2*\{e1,e2\}dv$

```
(%i119) bxf(u,-1)$
        diff(%,u)$
        integ4:%&*bfuv(u,-1)$
        facsum(%,allblds)$
        Jp4:'integrate(%,u,2,1);
```

(%o123) $-\int_{1}^{2} u*(4*u^2+1)du$

```
(%i124) ev(Jp1,nouns);
        ev(Jp2,nouns);
        ev(Jp3,nouns);
        ev(Jp4,nouns);
```

(%o124) $-4*\{e2,e3\}-2*\{e1,e2\}$

(%o125) $\frac{33}{2}$

(%o126) $16*\{e2,e3\}+4*\{e1,e2\}$

(%o127) $-\frac{33}{2}$

this has some redundant lines (for more complex examples)

```
(%i128) Jp:Jp1+Jp2+Jp3+Jp4$
        ev(Jp,nouns)$
        facsum(%,allblds)$
        Jrhs:ev(%,numer)$
```

and finally test for equality

```
(%i132) is(equal(Jlhs,Jrhs))$
        ldisplay(%,Jlhs,Jrhs)$
```

```
(%t133) %=true
(%t134) Jlhs=12*{e2,e3}+2*{e1,e2}
(%t135) Jrhs=12*{e2,e3}+2*{e1,e2}
```

Created with wxMaxima.