

```
def basic_multivector_operations_3D():
    Print_Function()
    g3d = Ga('e*x|y|z')
    (ex,ey,ez) = g3d.mv()
    A = g3d.mv('A','mv')
    print A.Fmt(1,'A')
    print A.Fmt(2,'A')
    print A.Fmt(3,'A')
    print A.even().Fmt(1,'%A_{+}')
    print A.odd().Fmt(1,'%A_{-}')
    X = g3d.mv('X','vector')
    Y = g3d.mv('Y','vector')
    print 'g-{ij} = ',g3d.g
    print X.Fmt(1,'X')
    print Y.Fmt(1,'Y')
    print (X*Y).Fmt(2,'X*Y')
    print (X^Y).Fmt(2,'X^Y')
    print (X|Y).Fmt(2,'X|Y')
    print cross(X,Y).Fmt(1,r'X\times Y')
    return
```

Code Output:

$$A = A + A^xe_x + A^ye_y + A^ze_z + A^{xy}e_x \wedge e_y + A^{xz}e_x \wedge e_z + A^{yz}e_y \wedge e_z + A^{xyz}e_x \wedge e_y \wedge e_z$$

$$\begin{aligned} A = & A \\ & + A^xe_x + A^ye_y + A^ze_z \\ & + A^{xy}e_x \wedge e_y + A^{xz}e_x \wedge e_z + A^{yz}e_y \wedge e_z \\ & + A^{xyz}e_x \wedge e_y \wedge e_z \end{aligned}$$

$$\begin{aligned} A = & A \\ & + A^xe_x \\ & + A^ye_y \\ & + A^ze_z \\ & + A^{xy}e_x \wedge e_y \\ & + A^{xz}e_x \wedge e_z \\ & + A^{yz}e_y \wedge e_z \\ & + A^{xyz}e_x \wedge e_y \wedge e_z \end{aligned}$$

$$A_+ = A + A^{xy}e_x \wedge e_y + A^{xz}e_x \wedge e_z + A^{yz}e_y \wedge e_z$$

$$A_- = A^xe_x + A^ye_y + A^ze_z + A^{xyz}e_x \wedge e_y \wedge e_z$$

$$g_{ij} = \begin{bmatrix} (e_x \cdot e_x) & (e_x \cdot e_y) & (e_x \cdot e_z) \\ (e_x \cdot e_y) & (e_y \cdot e_y) & (e_y \cdot e_z) \\ (e_x \cdot e_z) & (e_y \cdot e_z) & (e_z \cdot e_z) \end{bmatrix}$$

$$X = X^xe_x + X^ye_y + X^ze_z$$

$$Y = Y^xe_x + Y^ye_y + Y^ze_z$$

$$\begin{aligned} XY = & ((e_x \cdot e_x) X^xY^x + (e_x \cdot e_y) X^xY^y + (e_x \cdot e_y) X^yY^x + (e_x \cdot e_z) X^xY^z + (e_x \cdot e_z) X^zY^x + (e_y \cdot e_y) X^yY^y + (e_y \cdot e_z) X^yY^z + (e_y \cdot e_z) X^zY^y + (e_z \cdot e_z) X^zY^z) \\ & + (X^xY^y - X^yY^x) e_x \wedge e_y + (X^xY^z - X^zY^x) e_x \wedge e_z + (X^yY^z - X^zY^y) e_y \wedge e_z \end{aligned}$$

$$X \wedge Y = (X^xY^y - X^yY^x) e_x \wedge e_y + (X^xY^z - X^zY^x) e_x \wedge e_z + (X^yY^z - X^zY^y) e_y \wedge e_z$$

$$X \cdot Y = (e_x \cdot e_x) X^xY^x + (e_x \cdot e_y) X^xY^y + (e_x \cdot e_y) X^yY^x + (e_x \cdot e_z) X^xY^z + (e_x \cdot e_z) X^zY^x + (e_y \cdot e_y) X^yY^y + (e_y \cdot e_z) X^yY^z + (e_y \cdot e_z) X^zY^y + (e_z \cdot e_z) X^zY^z$$

$$X \times Y = \frac{1}{\sqrt{(e_x \cdot e_x)(e_y \cdot e_y)(e_z \cdot e_z) - (e_x \cdot e_x)(e_y \cdot e_z)^2 - (e_x \cdot e_y)^2(e_z \cdot e_z) + 2(e_x \cdot e_y)(e_x \cdot e_z)(e_y \cdot e_z) - (e_x \cdot e_z)^2(e_y \cdot e_y)}} \left((e_x \cdot e_y)(e_y \cdot e_z) X^xY^y - (e_x \cdot e_y)(e_y \cdot e_z) X^yY^x + (e_x \cdot e_y)(e_z \cdot e_z) X^xY^z - (e_x \cdot e_y)(e_z \cdot e_z) X^zY^x \right.$$

```
def basic_multivector_operations_2D():
    Print_Function()
    g2d = Ga('e*x|y')
    (ex,ey) = g2d.mv()
    print 'g-{ ij } =',g2d.g
    X = g2d.mv('X','vector')
    A = g2d.mv('A','spinor')
    print X.Fmt(1,'X')
    print A.Fmt(1,'A')
    print (X|A).Fmt(2,'X|A')
    print (X<A).Fmt(2,'X<A')
    print (A>X).Fmt(2,'A>X')
    return
```

Code Output:

$$g_{ij} = \begin{bmatrix} (e_x \cdot e_x) & (e_x \cdot e_y) \\ (e_x \cdot e_y) & (e_y \cdot e_y) \end{bmatrix}$$
$$X = X^x \boldsymbol{e}_x + X^y \boldsymbol{e}_y$$
$$A = A + A^{xy} \boldsymbol{e}_x \wedge \boldsymbol{e}_y$$
$$X \cdot A = -A^{xy} ((e_x \cdot e_y) X^x + (e_y \cdot e_y) X^y) \boldsymbol{e}_x + A^{xy} ((e_x \cdot e_x) X^x + (e_x \cdot e_y) X^y) \boldsymbol{e}_y$$
$$X|A = -A^{xy} ((e_x \cdot e_y) X^x + (e_y \cdot e_y) X^y) \boldsymbol{e}_x + A^{xy} ((e_x \cdot e_x) X^x + (e_x \cdot e_y) X^y) \boldsymbol{e}_y$$
$$A|X = A^{xy} ((e_x \cdot e_y) X^x + (e_y \cdot e_y) X^y) \boldsymbol{e}_x - A^{xy} ((e_x \cdot e_x) X^x + (e_x \cdot e_y) X^y) \boldsymbol{e}_y$$

```
def basic_multivector_operations_2D_orthogonal():
    Print_Function()
    o2d = Ga('e*x|y',g=[1,1])
    (ex,ey) = o2d.mv()
    print 'g-{ ii } =',o2d.g
    X = o2d.mv('X','vector')
    A = o2d.mv('A','spinor')
    print X.Fmt(1,'X')
    print A.Fmt(1,'A')
    print (X*A).Fmt(2,'X*A')
    print (X|A).Fmt(2,'X|A')
    print (X<A).Fmt(2,'X<A')
    print (X>A).Fmt(2,'X>A')
    print (A*X).Fmt(2,'A*X')
    print (A|X).Fmt(2,'A|X')
    print (A<X).Fmt(2,'A<X')
    print (A>X).Fmt(2,'A>X')
    return
```

Code Output:

$$g_{ii} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$
$$X = X^x \boldsymbol{e}_x + X^y \boldsymbol{e}_y$$
$$A = A + A^{xy} \boldsymbol{e}_x \wedge \boldsymbol{e}_y$$
$$XA = (AX^x - A^{xy} X^y) \boldsymbol{e}_x + (AX^y + A^{xy} X^x) \boldsymbol{e}_y$$
$$X \cdot A = -A^{xy} X^y \boldsymbol{e}_x + A^{xy} X^x \boldsymbol{e}_y$$
$$X|A = -A^{xy} X^y \boldsymbol{e}_x + A^{xy} X^x \boldsymbol{e}_y$$
$$X|A = AX^x \boldsymbol{e}_x + AX^y \boldsymbol{e}_y$$
$$AX = (AX^x + A^{xy} X^y) \boldsymbol{e}_x + (AX^y - A^{xy} X^x) \boldsymbol{e}_y$$

$$A\cdot X=A^{xy}X^ye_x-A^{xy}X^xe_y$$

$$A\rfloor X=AX^xe_x+AX^ye_y$$

$$A\lrcorner X=A^{xy}X^ye_x-A^{xy}X^xe_y$$

```
def check_generalized_BAC_CAB_formulas():
    Print_Function()
    g4d = Ga('a b c d')
    (a,b,c,d) = g4d.mv()
    print 'g_{ij} =',g4d.g
    print '\\bm{a|(b*c)} =',a|(b*c)
    print '\\bm{a|(b^c)} =',a|(b^c)
    print '\\bm{a|(b^c^d)} =',a|(b^c^d)
    print '\\bm{a|(b^c)+c|(a^b)+b|(c^a)} =',(a|(b^c))+(c|(a^b))+(b|(c^a))
    print '\\bm{a*(b^c)-b*(a^c)+c*(a^b)} =',a*(b^c)-b*(a^c)+c*(a^b)
    print '\\bm{a*(b^c^d)-b*(a^c^d)+c*(a^b^d)-d*(a^b^c)} =',a*(b^c^d)-b*(a^c^d)+c*(a^b^d)-d*(a^b^c)
    print '\\bm{(a^b)|(c^d)} =',(a^b)|(c^d)
    print '\\bm{((a^b)|c)|d} =',((a^b)|c)|d
    print '\\bm{(a^b)\\times (c^d)} =',com(a^b,c^d)
    return
```

Code Output:

$$g_{ij} = \begin{bmatrix} (a\cdot a) & (a\cdot b) & (a\cdot c) & (a\cdot d) \\ (a\cdot b) & (b\cdot b) & (b\cdot c) & (b\cdot d) \\ (a\cdot c) & (b\cdot c) & (c\cdot c) & (c\cdot d) \\ (a\cdot d) & (b\cdot d) & (c\cdot d) & (d\cdot d) \end{bmatrix}$$

$$\boldsymbol{a}\cdot(\boldsymbol{bc})=-(a\cdot c)\boldsymbol{b}+(a\cdot b)\boldsymbol{c}$$

$$\boldsymbol{a}\cdot(\boldsymbol{b}\wedge\boldsymbol{c})=-(a\cdot c)\boldsymbol{b}+(a\cdot b)\boldsymbol{c}$$

$$\boldsymbol{a}\cdot(\boldsymbol{b}\wedge\boldsymbol{c}\wedge\boldsymbol{d})=(a\cdot d)\boldsymbol{b}\wedge\boldsymbol{c}-(a\cdot c)\boldsymbol{b}\wedge\boldsymbol{d}+(a\cdot b)\boldsymbol{c}\wedge\boldsymbol{d}$$

$$\boldsymbol{a}\cdot(\boldsymbol{b}\wedge\boldsymbol{c})+\boldsymbol{c}\cdot(\boldsymbol{a}\wedge\boldsymbol{b})+\boldsymbol{b}\cdot(\boldsymbol{c}\wedge\boldsymbol{a})=0$$

$$\boldsymbol{a}(\boldsymbol{b}\wedge\boldsymbol{c})-\boldsymbol{b}(\boldsymbol{a}\wedge\boldsymbol{c})+\boldsymbol{c}(\boldsymbol{a}\wedge\boldsymbol{b})=3\boldsymbol{a}\wedge\boldsymbol{b}\wedge\boldsymbol{c}$$

$$\boldsymbol{a}(\boldsymbol{b}\wedge\boldsymbol{c}\wedge\boldsymbol{d})-\boldsymbol{b}(\boldsymbol{a}\wedge\boldsymbol{c}\wedge\boldsymbol{d})+\boldsymbol{c}(\boldsymbol{a}\wedge\boldsymbol{b}\wedge\boldsymbol{d})-\boldsymbol{d}(\boldsymbol{a}\wedge\boldsymbol{b}\wedge\boldsymbol{c})=4\boldsymbol{a}\wedge\boldsymbol{b}\wedge\boldsymbol{c}\wedge\boldsymbol{d}$$

$$(\boldsymbol{a}\wedge\boldsymbol{b})\cdot(\boldsymbol{c}\wedge\boldsymbol{d})=-(a\cdot c)(b\cdot d)+(a\cdot d)(b\cdot c)$$

$$((\boldsymbol{a}\wedge\boldsymbol{b})\cdot\boldsymbol{c})\cdot\boldsymbol{d}=-(a\cdot c)(b\cdot d)+(a\cdot d)(b\cdot c)$$

$$(\boldsymbol{a}\wedge\boldsymbol{b})\times(\boldsymbol{c}\wedge\boldsymbol{d})=-(b\cdot d)\boldsymbol{a}\wedge\boldsymbol{c}+(b\cdot c)\boldsymbol{a}\wedge\boldsymbol{d}+(a\cdot d)\boldsymbol{b}\wedge\boldsymbol{c}-(a\cdot c)\boldsymbol{b}\wedge\boldsymbol{d}$$

```
def rounding_numerical_components():
    Print_Function()
    o3d = Ga('e_x e_y e_z',g=[1,1,1])
    (ex,ey,ez) = o3d.mv()
    X = 1.2*ex+2.34*ey+0.555*ez
    Y = 0.333*ex+4*ey+5.3*ez
    print 'X =',X
    print 'Nga(X,2) =',Nga(X,2)
    print 'X*Y =',X*Y
    print 'Nga(X*Y,2) =',Nga(X*Y,2)
    return
```

Code Output:

$$X=1\cdot2\boldsymbol{e}_x+2\cdot34\boldsymbol{e}_y+0\cdot555\boldsymbol{e}_z$$

$$Nga(X,2)=1\cdot2\boldsymbol{e}_x+2\cdot3\boldsymbol{e}_y+0\cdot55\boldsymbol{e}_z$$

$$XY=12\cdot7011$$

$$+4\cdot02078\boldsymbol{e}_x\wedge\boldsymbol{e}_y+6\cdot175185\boldsymbol{e}_x\wedge\boldsymbol{e}_z+10\cdot182\boldsymbol{e}_y\wedge\boldsymbol{e}_z$$

$$Nga(XY,2)=13\cdot 0$$

$$+4\cdot 0\boldsymbol{e}_x\wedge\boldsymbol{e}_y+6\cdot 2\boldsymbol{e}_x\wedge\boldsymbol{e}_z+10\cdot 0\boldsymbol{e}_y\wedge\boldsymbol{e}_z$$

```
def derivatives_in_rectangular_coordinates():
    Print_Function()
    X = (x,y,z) = symbols('x y z')
    o3d = Ga('e_x e_y e_z',g=[1,1,1],coords=X)
    (ex,ey,ez) = o3d.mv()
    grad = o3d.grad
    f = o3d.mv('f','scalar',f=True)
    A = o3d.mv('A','vector',f=True)
    B = o3d.mv('B','bivector',f=True)
    C = o3d.mv('C','mv')
    print 'f =',f
    print 'A =',A
    print 'B =',B
    print 'C =',C
    print 'grad*f =',grad*f
    print 'grad|A =',grad|A
    print 'grad*A =',grad*A
    print '-I*(grad^A) =',-o3d.E()*(grad^A)
    print 'grad*B =',grad*B
    print 'grad^B =',grad^B
    print 'grad|B =',grad|B
    return
```

Code Output:

$$f=f$$

$$A=A^xe_x+A^ye_y+A^ze_z$$

$$B=B^{xy}e_x\wedge e_y+B^{xz}e_x\wedge e_z+B^{yz}e_y\wedge e_z$$

$$C=C$$

$$+C^xe_x+C^ye_y+C^ze_z$$

$$+C^{xy}e_x\wedge e_y+C^{xz}e_x\wedge e_z+C^{yz}e_y\wedge e_z$$

$$+C^{xyz}e_x\wedge e_y\wedge e_z$$

$$\nabla f=\partial_xfe_x+\partial_yfe_y+\partial_zfe_z$$

$$\nabla\cdot A=\partial_xA^x+\partial_yA^y+\partial_zA^z$$

$$\nabla A=(\partial_xA^x+\partial_yA^y+\partial_zA^z)$$

$$+(-\partial_yA^x+\partial_xA^y)e_x\wedge e_y+(-\partial_zA^x+\partial_xA^z)e_x\wedge e_z+(-\partial_zA^y+\partial_yA^z)e_y\wedge e_z$$

$$-I(\nabla\wedge A)=(-\partial_zA^y+\partial_yA^z)e_x+(\partial_zA^x-\partial_xA^z)e_y+(-\partial_yA^x+\partial_xA^y)e_z$$

$$\nabla B=(-\partial_yB^{xy}-\partial_zB^{xz})e_x+(\partial_xB^{xy}-\partial_zB^{yz})e_y+(\partial_xB^{xz}+\partial_yB^{yz})e_z$$

$$+(\partial_zB^{xy}-\partial_yB^{xz}+\partial_xB^{yz})e_x\wedge e_y\wedge e_z$$

$$\nabla\wedge B=(\partial_zB^{xy}-\partial_yB^{xz}+\partial_xB^{yz})e_x\wedge e_y\wedge e_z$$

$$\nabla\cdot B=(-\partial_yB^{xy}-\partial_zB^{xz})e_x+(\partial_xB^{xy}-\partial_zB^{yz})e_y+(\partial_xB^{xz}+\partial_yB^{yz})e_z$$

```
def derivatives_in_spherical_coordinates():
    Print_Function()
    X = (r,th,phi) = symbols('r theta phi')
    s3d = Ga('e_r e_theta e_phi',g=[1,r**2,r**2*sin(th)**2],coords=X,norm=True)
    (er,eth,ephi) = s3d.mv()
```

```
grad = s3d.grad
f = s3d.mv( 'f' , 'scalar' ,f=True)
A = s3d.mv( 'A' , 'vector' ,f=True)
B = s3d.mv( 'B' , 'bivector' ,f=True)
print  'f =' ,f
print  'A =' ,A
print  'B =' ,B
print  'grad*f =' ,grad*f
print  'grad|A =' ,grad|A
print  '-I*(grad^A) =' ,(-s3d.E()*(grad^A)).simplify()
print  'grad^B =' ,grad^B
```

Code Output:

$$f=f$$

$$A=A^r\boldsymbol{e}_r+A^\theta\boldsymbol{e}_\theta+A^\phi\boldsymbol{e}_\phi$$

$$B=B^{r\theta}\boldsymbol{e}_r\wedge\boldsymbol{e}_\theta+B^{r\phi}\boldsymbol{e}_r\wedge\boldsymbol{e}_\phi+B^{\phi\phi}\boldsymbol{e}_\theta\wedge\boldsymbol{e}_\phi$$

$$\boldsymbol{\nabla}f=\partial_rf\boldsymbol{e}_r+\frac{1}{r}\partial_\theta f\boldsymbol{e}_\theta+\frac{\partial_\phi f}{r\sin(\theta)}\boldsymbol{e}_\phi$$

$$\boldsymbol{\nabla}\cdot A=\frac{1}{r}\left(r\partial_rA^r+2A^r+\frac{A^\theta}{\tan(\theta)}+\partial_\theta A^\theta+\frac{\partial_\phi A^\phi}{\sin(\theta)}\right)$$

$$-I(\boldsymbol{\nabla}\wedge A)=\frac{1}{r}\left(\frac{A^\phi}{\tan(\theta)}+\partial_\theta A^\phi-\frac{\partial_\phi A^\theta}{\sin(\theta)}\right)\boldsymbol{e}_r+\frac{1}{r}\left(-r\partial_rA^\phi-A^\phi+\frac{\partial_\phi A^r}{\sin(\theta)}\right)\boldsymbol{e}_\theta+\frac{1}{r}\left(r\partial_rA^\theta+A^\theta-\partial_\theta A^r\right)\boldsymbol{e}_\phi$$

$$\boldsymbol{\nabla}\wedge B=\frac{1}{r}\left(r\partial_rB^{\phi\phi}-\frac{B^{r\phi}}{\tan(\theta)}+2B^{\phi\phi}-\partial_\theta B^{r\phi}+\frac{\partial_\phi B^{r\theta}}{\sin(\theta)}\right)\boldsymbol{e}_r\wedge\boldsymbol{e}_\theta\wedge\boldsymbol{e}_\phi$$

```
def noneuclidian_distance_calculation():
    Print_Function()
    from sympy import solve,sqrt
    Fmt(1)
    g = '0 # #,# 0 #,# # 1'
    nel = Ga( 'X Y e' ,g=g)
    (X,Y,e) = nel.mv()
    print  'g_{ij} =' ,nel.g
    print  '%(X\W Y)^{2} =' ,(X^Y)*(X^Y)
    L = X^Y^e
    B = L*e # DℳL 10.152
    Bsqr = (B*B).scalar()
    print  '##L = X\W Y\W e \\text{ is a non-euclidian line}'
    print  'B = L*e =' ,B
    BeBr =B*e*B.rev()
    print  '%BeB^{\dagger} =' ,BeBr
    print  '%B^{2} =' ,B*B
    print  '%L^{2} =' ,L*L # DℳL 10.153
    (s,c,Binv,M,S,C,alpha) = symbols('s c (1/B) M S C alpha')
    XdotY = nel.g[0,1]
    Xdote = nel.g[0,2]
    Ydote = nel.g[1,2]
    Bhat = Binv*B # DℳL 10.154
    R = c+s*Bhat # Rotor R = exp(alpha*Bhat/2)
    print  '##s = \\f{\sinh}{\alpha/2} \\text{ and } c = \\f{\cosh}{\alpha/2}'
    print  '%e^{\alpha B/{2\abs{B}}} =' ,R
    Z = R*X*R.rev() # DℳL 10.155
    Z.obj = expand(Z.obj)
    Z.obj = Z.obj.collect([Binv,s,c,XdotY])
    Z.Fmt(3,'%RXR^{\dagger}')
    W = Z|Y # Extract scalar part of multivector
```

```

# From this point forward all calculations are with sympy scalars
#print '#Objective is to determine value of C = cosh(alpha) such that W = 0'
W = W.scalar()
print '%W = Z\\cdot Y =',W
W = expand(W)
W = simplify(W)
W = W.collect([s*Binv])
M = 1/Bsq
W = W.subs(Binv**2,M)
W = simplify(W)
Bmag = sqrt(XdotY**2-2*XdotY*Xdote*Ydote)
W = W.collect([Binv*c*s,XdotY])
#Double angle substitutions
W = W.subs(2*XdotY**2-4*XdotY*Xdote*Ydote,2/(Binv**2))
W = W.subs(2*c*s,S)
W = W.subs(c**2,(C+1)/2)
W = W.subs(s**2,(C-1)/2)
W = simplify(W)
W = W.subs(1/Binv,Bmag)
W = expand(W)
print '%S = \\f{\\sinh}{\\alpha} \\text{ and } C = \\f{\\cosh}{\\alpha}',
print 'W =',W
Wd = collect(W,[C,S],exact=True,evaluate=False)
Wd_1 = Wd[one]
Wd_C = Wd[C]
Wd_S = Wd[S]
print '%\\text{Scalar Coefficient} =',Wd_1
print '%\\text{Cosh Coefficient} =',Wd_C
print '%\\text{Sinh Coefficient} =',Wd_S
print '%\\abs{B} =',Bmag
Wd_1 = Wd_1.subs(Bmag,1/Binv)
Wd_C = Wd_C.subs(Bmag,1/Binv)
Wd_S = Wd_S.subs(Bmag,1/Binv)
lhs = Wd_1+Wd_C*C
rhs = -Wd_S*S
lhs = lhs**2
rhs = rhs**2
W = expand(lhs-rhs)
W = expand(W.subs(1/Binv**2,Bmag**2))
W = expand(W.subs(S**2,C**2-1))
W = W.collect([C,C**2],evaluate=False)
a = simplify(W[C**2])
b = simplify(W[C])
c = simplify(W[one])
print '%\\text{Require } aC^2+bC+c = 0'
print 'a =',a
print 'b =',b
print 'c =',c
x = Symbol('x')
C = solve(a*x**2+b*x+c,x)[0]
print '%b^2-4ac =',simplify(b**2-4*a*c)
print '%\\f{\\cosh}{\\alpha} = C = -b/(2a) =',expand(simplify(expand(C)))
return

```

Code Output:

$$g_{ij} = \begin{bmatrix} 0 & (X \cdot Y) & (X \cdot e) \\ (X \cdot Y) & 0 & (Y \cdot e) \\ (X \cdot e) & (Y \cdot e) & 1 \end{bmatrix}$$

$$(X \wedge Y)^2 = (X \cdot Y)^2$$

$L = X \wedge Y \wedge e$ is a non-euclidian line

$$B = Le = \mathbf{X} \wedge \mathbf{Y} - (Y \cdot e) \mathbf{X} \wedge e + (X \cdot e) \mathbf{Y} \wedge e$$

$$BeB^\dagger = (X \cdot Y) (-(X \cdot Y) + 2(X \cdot e)(Y \cdot e)) e$$

$$B^2 = (X \cdot Y) ((X \cdot Y) - 2(X \cdot e)(Y \cdot e))$$

$$L^2 = (X \cdot Y) ((X \cdot Y) - 2(X \cdot e)(Y \cdot e))$$

$$s = \sinh(\alpha/2) \text{ and } c = \cosh(\alpha/2)$$

$$e^{\alpha B/2|B|} = c + (1/B)s\mathbf{X} \wedge \mathbf{Y} - (1/B)(Y \cdot e)s\mathbf{X} \wedge e + (1/B)(X \cdot e)s\mathbf{Y} \wedge e$$

$$W = Z \cdot Y = (1/B)^2 (X \cdot Y)^3 s^2 - 4(1/B)^2 (X \cdot Y)^2 (X \cdot e)(Y \cdot e) s^2 + 4(1/B)^2 (X \cdot Y)(X \cdot e)^2 (Y \cdot e)^2 s^2 + 2(1/B)(X \cdot Y)^2 cs - 4(1/B)(X \cdot Y)(X \cdot e)(Y \cdot e) cs + (X \cdot Y) c^2$$

$$S = \sinh(\alpha) \text{ and } C = \cosh(\alpha)$$

$$W = (1/B)(X \cdot Y)C\sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)} - (1/B)(X \cdot e)(Y \cdot e)C\sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)} + (1/B)(X \cdot e)(Y \cdot e)\sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)} + S\sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)}$$

$$\text{Scalar Coefficient} = (1/B)(X \cdot e)(Y \cdot e)\sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)}$$

$$\text{Cosh Coefficient} = (1/B)(X \cdot Y)\sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)} - (1/B)(X \cdot e)(Y \cdot e)\sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)}$$

$$\text{Sinh Coefficient} = \sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)}$$

$$|B| = \sqrt{(X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e)}$$

$$\text{Require } aC^2 + bC + c = 0$$

$$a = (X \cdot e)^2 (Y \cdot e)^2$$

$$b = 2(X \cdot e)(Y \cdot e)((X \cdot Y) - (X \cdot e)(Y \cdot e))$$

$$c = (X \cdot Y)^2 - 2(X \cdot Y)(X \cdot e)(Y \cdot e) + (X \cdot e)^2 (Y \cdot e)^2$$

$$b^2 - 4ac = 0$$

$$\cosh(\alpha) = C = -b/(2a) = -\frac{(X \cdot Y)}{(X \cdot e)(Y \cdot e)} + 1$$

```
def conformal_representations_of_circles_lines_spheres_and_planes():
    Print_Function()
    global n,nbar
    Fmt(1)
    g = '1 0 0 0 0,0 1 0 0 0,0 0 1 0 0,0 0 0 0 2,0 0 0 2 0'
    c3d = Ga('e_1 e_2 e_3 n \\bar{n}',g=g)
    (e1,e2,e3,n,nbar) = c3d.mv()
    print 'g-{ij} =',c3d.g
    e = n+nbar
    #conformal representation of points
    A = make_vector(e1, ga=c3d) # point a = (1,0,0) A = F(a)
    B = make_vector(e2, ga=c3d) # point b = (0,1,0) B = F(b)
    C = make_vector(-e1, ga=c3d) # point c = (-1,0,0) C = F(c)
    D = make_vector(e3, ga=c3d) # point d = (0,0,1) D = F(d)
    X = make_vector('x',3, ga=c3d)
    print 'F(a) =',A
    print 'F(b) =',B
    print 'F(c) =',C
    print 'F(d) =',D
    print 'F(x) =',X
    print '#a = e1, b = e2, c = -e1, and d = e3'
    print '#A = F(a) = 1/2*(a*a*n+2*a-nbar), etc.'
    print '#Circle through a, b, and c'
    print 'Circle: A^B^C^X = 0 =',(A^B^C^X)
    print '#Line through a and b'
    print 'Line : A^B^n^X = 0 =',(A^B^n^X)
```

```
print '#Sphere through a, b, c, and d'
print 'Sphere: A^B^C^D^X = 0 =' ,(((A^B)^C)^D)^X
print '#Plane through a, b, and d'
print 'Plane : A^B^n^D^X = 0 =' ,(A^B^n^D^X)
L = (A^B^e)^X
L.Fmt(3,'Hyperbolic\\;\\; Circle: (A^B^e)^X = 0')
```

Code Output:

$$g_{ij} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 2 & 0 \end{bmatrix}$$

$$F(a) = e_1 + \frac{1}{2}n - \frac{1}{2}\bar{n}$$

$$F(b) = e_2 + \frac{1}{2}n - \frac{1}{2}\bar{n}$$

$$F(c) = -e_1 + \frac{1}{2}n - \frac{1}{2}\bar{n}$$

$$F(d) = e_3 + \frac{1}{2}n - \frac{1}{2}\bar{n}$$

$$F(x) = x_1e_1 + x_2e_2 + x_3e_3 + \left(\frac{1}{2}(x_1)^2 + \frac{1}{2}(x_2)^2 + \frac{1}{2}(x_3)^2\right)n - \frac{1}{2}\bar{n}$$

a = e1, b = e2, c = -e1, and d = e3 A = F(a) = 1/2*(a*a*n+2*a-nbar), etc. Circle through a, b, and c

$$Circle : A \wedge B \wedge C \wedge X = 0 = -x_3e_1 \wedge e_2 \wedge e_3 \wedge n + x_3e_1 \wedge e_2 \wedge e_3 \wedge \bar{n} + \left(\frac{1}{2}(x_1)^2 + \frac{1}{2}(x_2)^2 + \frac{1}{2}(x_3)^2 - \frac{1}{2}\right)e_1 \wedge e_2 \wedge n \wedge \bar{n}$$

Line through a and b

$$Line : A \wedge B \wedge n \wedge X = 0 = -x_3e_1 \wedge e_2 \wedge e_3 \wedge n + \left(\frac{x_1}{2} + \frac{x_2}{2} - \frac{1}{2}\right)e_1 \wedge e_2 \wedge n \wedge \bar{n} + \frac{x_3}{2}e_1 \wedge e_3 \wedge n \wedge \bar{n} - \frac{x_3}{2}e_2 \wedge e_3 \wedge n \wedge \bar{n}$$

Sphere through a, b, c, and d

$$Sphere : A \wedge B \wedge C \wedge D \wedge X = 0 = \left(-\frac{1}{2}(x_1)^2 - \frac{1}{2}(x_2)^2 - \frac{1}{2}(x_3)^2 + \frac{1}{2}\right)e_1 \wedge e_2 \wedge e_3 \wedge n \wedge \bar{n}$$

Plane through a, b, and d

$$Plane : A \wedge B \wedge n \wedge D \wedge X = 0 = \left(-\frac{x_1}{2} - \frac{x_2}{2} - \frac{x_3}{2} + \frac{1}{2}\right)e_1 \wedge e_2 \wedge e_3 \wedge n \wedge \bar{n}$$

```
def properties_of_geometric_objects():
    Print_Function()
    global n, nbar
    Fmt(1)
    g = '# # # 0 0, '+ \
        '# # # 0 0, '+ \
        '# # # 0 0, '+ \
        '0 0 0 0 2, '+ \
        '0 0 0 2 0'
    c3d = Ga('p1 p2 p3 n \\bar{n}',g=g)
    (p1,p2,p3,n,nbar) = c3d.mv()
    print 'g-{ij} =',c3d.g
    P1 = F(p1)
    P2 = F(p2)
    P3 = F(p3)
```



```
print '\\text{Extracting direction of line from }L = P1\\W P2\\W n'
L = P1^P2^n
delta = (L|n)|nbar
print '(L|n)|\\bar{n} = ', delta
print '\\text{Extracting plane of circle from }C = P1\\W P2\\W P3'
C = P1^P2^P3
delta = ((C^n)|n)|nbar
print '((C^n)|n)|\\bar{n}= ', delta
print '(p2-p1)^(p3-p1)= ', (p2-p1)^(p3-p1)
return
```

Code Output:

$$g_{ij} = \begin{bmatrix} (p_1 \cdot p_1) & (p_1 \cdot p_2) & (p_1 \cdot p_3) & 0 & 0 \\ (p_1 \cdot p_2) & (p_2 \cdot p_2) & (p_2 \cdot p_3) & 0 & 0 \\ (p_1 \cdot p_3) & (p_2 \cdot p_3) & (p_3 \cdot p_3) & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 2 & 0 \end{bmatrix}$$

Extracting direction of line from $L = P1 \wedge P2 \wedge n$

$$(L \cdot n) \cdot \bar{n} = 2\boldsymbol{p_1} - 2\boldsymbol{p_2}$$

Extracting plane of circle from $C = P1 \wedge P2 \wedge P3$

$$((C \wedge n) \cdot n) \cdot \bar{n} = 2\boldsymbol{p_1} \wedge \boldsymbol{p_2} - 2\boldsymbol{p_1} \wedge \boldsymbol{p_3} + 2\boldsymbol{p_2} \wedge \boldsymbol{p_3}$$

$$(p2 - p1) \wedge (p3 - p1) = \boldsymbol{p_1} \wedge \boldsymbol{p_2} - \boldsymbol{p_1} \wedge \boldsymbol{p_3} + \boldsymbol{p_2} \wedge \boldsymbol{p_3}$$

```
def extracting_vectors_from_conformal_2_blade():
    Print_Function()
    Fmt(1)
    print r'B = P1\\W P2'
    g = '0 -1 #, '+ \
        '-1 0 #, '+ \
        '# # #'
    c2b = Ga('P1 P2 a', g=g)
    (P1,P2,a) = c2b.mv()
    print 'g_{ij} = ', c2b.g
    B = P1^P2
    Bsqr = B*B
    print '%B^{2} = ', Bsqr
    ap = a-(a^B)*B
    print "a' = a-(a^B)*B =", ap
    Ap = ap+ap*B
    Am = ap-ap*B
    print "A+ = a'+a'*B =", Ap
    print "A- = a'-a'*B =", Am
    print '%(A+)^{2} = ', Ap*Ap
    print '%(A-)^{2} = ', Am*Am
    aB = a|B
    print 'a|B = ', aB
    return
```

Code Output:

$$B = P1 \wedge P2$$

$$g_{ij} = \begin{bmatrix} 0 & -1 & (P_1 \cdot a) \\ -1 & 0 & (P_2 \cdot a) \\ (P_1 \cdot a) & (P_2 \cdot a) & (a \cdot a) \end{bmatrix}$$

$$B^2 = 1$$

$$a' = a - (a \wedge B)B = -(P_2 \cdot a) \boldsymbol{P_1} - (P_1 \cdot a) \boldsymbol{P_2}$$

$$A+ = a' + a'B = -2(P_2 \cdot a) \mathbf{P}_1$$

$$A- = a' - a'B = -2(P_1 \cdot a) \mathbf{P}_2$$

$$(A+)^2 = 0$$

$$(A-)^2 = 0$$

$$a \cdot B = -(P_2 \cdot a) \mathbf{P}_1 + (P_1 \cdot a) \mathbf{P}_2$$

```

def reciprocal_frame_test():
    Print_Function()
    Fmt(1)
    g = '1 # #, '+ \
        '# 1 #, '+ \
        '# # 1 '
    ng3d = Ga('e1 e2 e3 ',g=g)
    (e1,e2,e3) = ng3d.mv()
    print 'g_{ij} =',ng3d.g
    E = e1^e2^e3
    Esq = (E*E).scalar()
    print 'E =',E
    print '%E^{2} =',Esq
    Esq_inv = 1/Esq
    E1 = (e2^e3)*E
    E2 = (-1)*(e1^e3)*E
    E3 = (e1^e2)*E
    print 'E1 = (e2^e3)*E =',E1
    print 'E2 = -(e1^e3)*E =',E2
    print 'E3 = (e1^e2)*E =',E3
    w = (E1|e2)
    w = w.expand()
    print 'E1|e2 =',w
    w = (E1|e3)
    w = w.expand()
    print 'E1|e3 =',w
    w = (E2|e1)
    w = w.expand()
    print 'E2|e1 =',w
    w = (E2|e3)
    w = w.expand()
    print 'E2|e3 =',w
    w = (E3|e1)
    w = w.expand()
    print 'E3|e1 =',w
    w = (E3|e2)
    w = w.expand()
    print 'E3|e2 =',w
    w = (E1|e1)
    w = (w.expand()).scalar()
    Esq = expand(Esq)
    print '%(E1\\cdot e1)/E^{2} =',simplify(w/Esq)
    w = (E2|e2)
    w = (w.expand()).scalar()
    print '%(E2\\cdot e2)/E^{2} =',simplify(w/Esq)
    w = (E3|e3)
    w = (w.expand()).scalar()
    print '%(E3\\cdot e3)/E^{2} =',simplify(w/Esq)
    return

```

Code Output:

$$g_{ij} = \begin{bmatrix} 1 & (e_1 \cdot e_2) & (e_1 \cdot e_3) \\ (e_1 \cdot e_2) & 1 & (e_2 \cdot e_3) \\ (e_1 \cdot e_3) & (e_2 \cdot e_3) & 1 \end{bmatrix}$$
$$E = e_1 \wedge e_2 \wedge e_3$$
$$E^2 = (e_1 \cdot e_2)^2 - 2(e_1 \cdot e_2)(e_1 \cdot e_3)(e_2 \cdot e_3) + (e_1 \cdot e_3)^2 + (e_2 \cdot e_3)^2 - 1$$
$$E1 = (e2 \wedge e3)E = \left((e_2 \cdot e_3)^2 - 1\right) e_1 + ((e_1 \cdot e_2) - (e_1 \cdot e_3)(e_2 \cdot e_3)) e_2 + (- (e_1 \cdot e_2)(e_2 \cdot e_3) + (e_1 \cdot e_3)) e_3$$
$$E2 = -(e1 \wedge e3)E = ((e_1 \cdot e_2) - (e_1 \cdot e_3)(e_2 \cdot e_3)) e_1 + \left((e_1 \cdot e_3)^2 - 1\right) e_2 + (- (e_1 \cdot e_2)(e_1 \cdot e_3) + (e_2 \cdot e_3)) e_3$$
$$E3 = (e1 \wedge e2)E = (- (e_1 \cdot e_2)(e_2 \cdot e_3) + (e_1 \cdot e_3)) e_1 + (- (e_1 \cdot e_2)(e_1 \cdot e_3) + (e_2 \cdot e_3)) e_2 + \left((e_1 \cdot e_2)^2 - 1\right) e_3$$
$$E1 \cdot e2 = 0$$
$$E1 \cdot e3 = 0$$
$$E2 \cdot e1 = 0$$
$$E2 \cdot e3 = 0$$
$$E3 \cdot e1 = 0$$
$$E3 \cdot e2 = 0$$
$$(E1 \cdot e1)/E^2 = 1$$
$$(E2 \cdot e2)/E^2 = 1$$
$$(E3 \cdot e3)/E^2 = 1$$

```
def signature_test():
    Print_Function()
    e3d = Ga('e1 e2 e3 ',g=[1,1,1])
    print 'g =', e3d.g
    print r'%Signature = (3,0)\: I =', e3d.I(), '\: I^{2} =', e3d.I()*e3d.I()
    e3d = Ga('e1 e2 e3 ',g=[2,2,2])
    print 'g =', e3d.g
    print r'%Signature = (3,0)\: I =', e3d.I(), '\: I^{2} =', e3d.I()*e3d.I()
    sp4d = Ga('e1 e2 e3 e4 ',g=[1,-1,-1,-1])
    print 'g =', sp4d.g
    print r'%Signature = (1,3)\: I =', sp4d.I(), '\: I^{2} =', sp4d.I()*sp4d.I()
    sp4d = Ga('e1 e2 e3 e4 ',g=[2,-2,-2,-2])
    print 'g =', sp4d.g
    print r'%Signature = (1,3)\: I =', sp4d.I(), '\: I^{2} =', sp4d.I()*sp4d.I()
    e4d = Ga('e1 e2 e3 e4 ',g=[1,1,1,1])
    print 'g =', e4d.g
    print r'%Signature = (4,0)\: I =', e4d.I(), '\: I^{2} =', e4d.I()*e4d.I()
    cf3d = Ga('e1 e2 e3 e4 e5 ',g=[1,1,1,1,-1])
    print 'g =', cf3d.g
    print r'%Signature = (4,1)\: I =', cf3d.I(), '\: I^{2} =', cf3d.I()*cf3d.I()
    cf3d = Ga('e1 e2 e3 e4 e5 ',g=[2,2,2,2,-2])
    print 'g =', cf3d.g
    print r'%Signature = (4,1)\: I =', cf3d.I(), '\: I^{2} =', cf3d.I()*cf3d.I()
    return
```

Code Output:

$$g = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
$$Signature = (3,0) \, I = e_1 \wedge e_2 \wedge e_3 \, I^2 = -1$$

$$g = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

$$Signature = (3,0) \ I = \frac{\sqrt{2}}{4} \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3; I^2 = -1$$

$$g = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

$$Signature = (1,3) \ I = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 \ I^2 = -1$$

$$g = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \end{bmatrix}$$

$$Signature = (1,3) \ I = \frac{1}{4} \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 \ I^2 = -1$$

$$g = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Signature = (4,0) \ I = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 \ I^2 = 1$$

$$g = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

$$Signature = (4,1) \ I = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 \wedge \mathbf{e}_5 \ I^2 = -1$$

$$g = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & -2 \end{bmatrix}$$

$$Signature = (4,1) \ I = \frac{\sqrt{2}}{8} \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 \wedge \mathbf{e}_5 \ I^2 = -1$$

```
def Fmt_test():
    Print_Function()
    e3d = Ga('e1 e2 e3',g=[1,1,1])
    v = e3d.mv('v','vector')
    B = e3d.mv('B','bivector')
    M = e3d.mv('M','mv')
    Fmt(2)
    print '#Global $Fmt = 2$'
    print 'v =',v
    print 'B =',B
    print 'M =',M
    print '#Using $.Fmt()$ Function'
    print 'v.Fmt(3) =',v.Fmt(3)
    print 'B.Fmt(3) =',B.Fmt(3)
    print 'M.Fmt(2) =',M.Fmt(2)
    print 'M.Fmt(1) =',M.Fmt(1)
    print '#Global $Fmt = 1$'
    Fmt(1)
    print 'v =',v
    print 'B =',B
    print 'M =',M
    return
```

Code Output: Global $Fmt = 2$

$$v = v^1 \mathbf{e}_1 + v^2 \mathbf{e}_2 + v^3 \mathbf{e}_3$$
$$B = B^{12} \mathbf{e}_1 \wedge \mathbf{e}_2 + B^{13} \mathbf{e}_1 \wedge \mathbf{e}_3 + B^{23} \mathbf{e}_2 \wedge \mathbf{e}_3$$

$$M = M$$
$$+ M^1 \mathbf{e}_1 + M^2 \mathbf{e}_2 + M^3 \mathbf{e}_3$$
$$+ M^{12} \mathbf{e}_1 \wedge \mathbf{e}_2 + M^{13} \mathbf{e}_1 \wedge \mathbf{e}_3 + M^{23} \mathbf{e}_2 \wedge \mathbf{e}_3$$
$$+ M^{123} \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3$$

Using $.Fmt()$ Function

$$v \cdot Fmt(3) = v^1 \mathbf{e}_1$$
$$+ v^2 \mathbf{e}_2$$
$$+ v^3 \mathbf{e}_3$$

$$B \cdot Fmt(3) = B^{12} \mathbf{e}_1 \wedge \mathbf{e}_2$$
$$+ B^{13} \mathbf{e}_1 \wedge \mathbf{e}_3$$
$$+ B^{23} \mathbf{e}_2 \wedge \mathbf{e}_3$$

$$M \cdot Fmt(2) = M$$
$$+ M^1 \mathbf{e}_1 + M^2 \mathbf{e}_2 + M^3 \mathbf{e}_3$$
$$+ M^{12} \mathbf{e}_1 \wedge \mathbf{e}_2 + M^{13} \mathbf{e}_1 \wedge \mathbf{e}_3 + M^{23} \mathbf{e}_2 \wedge \mathbf{e}_3$$
$$+ M^{123} \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3$$

$$M \cdot Fmt(1) = M + M^1 \mathbf{e}_1 + M^2 \mathbf{e}_2 + M^3 \mathbf{e}_3 + M^{12} \mathbf{e}_1 \wedge \mathbf{e}_2 + M^{13} \mathbf{e}_1 \wedge \mathbf{e}_3 + M^{23} \mathbf{e}_2 \wedge \mathbf{e}_3 + M^{123} \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3$$

Global $Fmt = 1$

$$v = v^1 \mathbf{e}_1 + v^2 \mathbf{e}_2 + v^3 \mathbf{e}_3$$
$$B = B^{12} \mathbf{e}_1 \wedge \mathbf{e}_2 + B^{13} \mathbf{e}_1 \wedge \mathbf{e}_3 + B^{23} \mathbf{e}_2 \wedge \mathbf{e}_3$$
$$M = M + M^1 \mathbf{e}_1 + M^2 \mathbf{e}_2 + M^3 \mathbf{e}_3 + M^{12} \mathbf{e}_1 \wedge \mathbf{e}_2 + M^{13} \mathbf{e}_1 \wedge \mathbf{e}_3 + M^{23} \mathbf{e}_2 \wedge \mathbf{e}_3 + M^{123} \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3$$