

Physics

Maple provides a state-of-the-art environment for algebraic computations in Physics, with emphasis on ensuring that the computational experience is as natural as possible. The theme of the [Physics project](#) for Maple 2021 has been the consolidation of the functionality introduced in previous releases, together with significant enhancements in the areas of Particle Physics, Quantum Mechanics, Tensor computations and General Relativity.

As part of its commitment to providing the best possible computational environment in Physics, Maplesoft launched a [Maple Physics: Research and Development](#) website in 2014, which enabled users to download research versions of the package, ask questions, and provide feedback. The results from this accelerated exchange have been incorporated into the Physics package in Maple 2021. The presentation below illustrates both the novelties and the kind of mathematical formulations that can now be performed.

Feynman Diagrams and Feynman Integrals	Weyl scalars, Petrov types and canonical forms for tetrads
Simplification of products of Dirac matrices	Documentation advanced examples
Manipulation and simplification of expressions involving non-commutative tensor operators	Miscellaneous
Functional differentiation and differentiation of spinors	See Also
New commands ToCovariant and ToContravariant	

Feynman Diagrams and Feynman Integrals

Feynman Diagrams are the cornerstone of calculations in particle physics (collisions involving from the proton to the Higgs boson), for example at the [CERN](#). As an introduction for people not working in the area, see "[Why Feynman Diagrams are so important](#)". In connection, Maple 2020 presented a **full rewriting** of the [FeynmanDiagrams](#) command including a myriad of new capabilities.

In Maple 2021, in addition, we implemented:

- New options in the *FeynmanDiagrams* command, to handle the typical situation where one wants to study scattering amplitudes using different gauge choices for the fields' propagators; or a generic form of a propagator, useful to perform the computation with models that require a non-standard form for them; or request that the propagators used are displayed on the screen above the returned result; or computing the amplitudes without external legs normalization factors.
- A new [FeynmanIntegral](#) module to evaluate the Feynman integrals that appear in the

output of the [FeynmanDiagrams](#) command, including package's commands for the basic steps involved, that is: to [Parametrize](#) or to [Evaluate](#) in one go, with different options, all the Feynman integrals found in an expression typically returned by [FeynmanDiagrams](#).

Examples - new FeynmanDiagrams options

> *restartv, with(Physics) :*

New: you can set a vector and a *mixed* spacetime-spinor field (i.e. 3/2 spinor). In this example, indicate also that A_μ is massless

> *Setup(coordinates = X, spinorindices = lowercaselatin, anticommutativeprefix = Q, op = {A, Q},
massless = A, tensors = {Q[\mu, a], A[\mu]})*
** Partial match of 'op' against keyword 'quantumoperators'*
** Partial match of 'massless' against keyword 'masslessfields'*
Systems of spacetime coordinates are: {X = (x1, x2, x3, x4)}

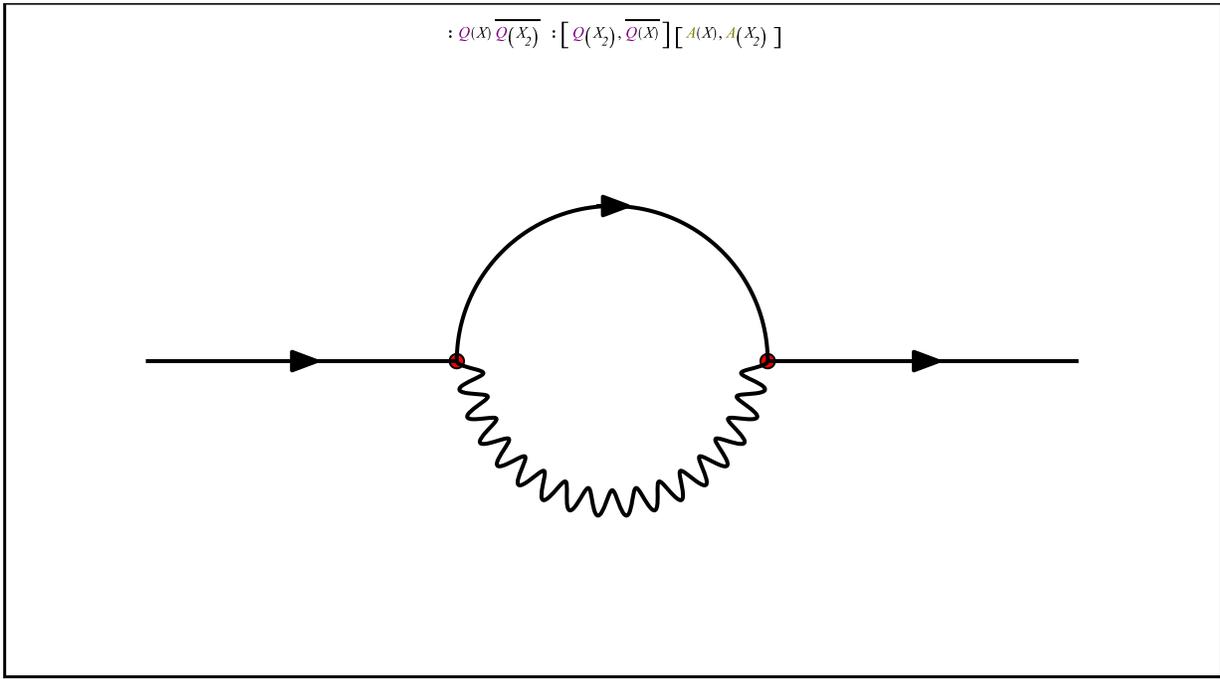
[anticommutativeprefix = {Q}, coordinatesystems = {X}, masslessfields = {A}, **(1)**
quantumoperators = {A, Q}, spinorindices = lowercaselatin, tensors = {A_\mu, \gamma_\mu, \sigma_\mu, Q_{\mu, a},
\partial_\mu, g_{\mu, \nu}, \epsilon_{\alpha, \beta, \mu, \nu}, X_\mu}]

Now write down a minimal interaction Lagrangian, similar but different from QED, for testing purposes

> $L := \alpha \left(\gamma^\mu \right)_{a, i} \left(\gamma^\nu \right)_{i, j} \left(\gamma^\rho \right)_{j, b} \overline{Q_{\mu, a}(X)} Q_{\rho, b}(X) A_\nu(X) :$

Compute the amplitude for a process where there are 1 incoming and 1 outgoing particles of the same Q kind; this is the self-energy diagram:

> *FeynmanDiagrams(L, incomingparticles = [Q], outgoing = [Q], numberofloops = 1, diagrams)*



$$\begin{aligned}
 & - \int \frac{1}{8 \pi^3 (p_2^2 + i\epsilon) ((P_1 + p_2)^2 - m_Q^2 + i\epsilon)} \left((u_Q)_{\alpha, c} (\vec{P}_1) \right. \\
 & \overline{(u_Q)_{\beta, d} (\vec{P}_2)} g_{\lambda, \tau} \alpha^2 (\gamma^\kappa)_{l, e} (\gamma^\beta)_{d, k} (\gamma^\tau)_{k, l} (\gamma^\alpha)_{n, c} (\gamma^\sigma)_{f, m} (\gamma^\lambda)_{m, n} \left(\left(g_{\kappa, \sigma} \right. \right. \\
 & \left. \left. - \frac{(P_{1\kappa} + p_{2\kappa})(P_{1\sigma} + p_{2\sigma})}{m_Q^2} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{e, f} + m_Q \delta_{e, f} \right) \right. \\
 & \left. + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{e, g}}{3 m_Q} + \frac{(\gamma_\kappa)_{e, g}}{3} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{g, h} \right. \right. \\
 & \left. \left. - m_Q \delta_{g, h} \right) \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{f, h}}{m_Q} + (\gamma_\sigma)_{h, f} \right) \right) \delta(-P_2 + P_1) dp_2^4
 \end{aligned} \tag{2}$$

In this result, there are two things that are new in *FeynmanDiagrams*:

- a) it can now handle 3/2 spinor fields, with 1 spacetime and 1 spinor indices;
- b) it is now setting the gauge for the massless field A_μ in a way that can be changed.

To see all that, we implemented a new [userinfo](#) message, so that one can see, exactly, the form of the propagator being used. Set the corresponding [infolevel](#) as follows and run the computation again (skip producing the diagram's drawing)

> `infolevel[FeynmanDiagrams] := 3 :`

> *FeynmanDiagrams*(L , *incomingparticles* = [Q], *outgoing* = [Q], *numberofloops* = 1)

$$\text{Propagator used: } \left(\Delta(A_\lambda(p_2), A_\tau(-p_2)) = \frac{-I g_{\lambda\tau}}{p_2^2 + I\epsilon} \right)$$

$$\text{Propagator used: } \left(\Delta(Q_{\kappa,e}(P_1 + p_2), Q_{\sigma,f}^I(-P_1 - p_2)) \right)$$

$$= \frac{1}{(P_1 + p_2)^2 - m_Q^2 + I\epsilon} \left(I \left(\left(g_{\kappa\sigma} - \frac{(P_{1\kappa} + p_{2\kappa})(P_{1\sigma} + p_{2\sigma})}{m_Q^2} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{e,f} + m_Q \delta_{e,f} \right) + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{e,g}}{3 m_Q} + \frac{(\gamma_\kappa)_{e,g}}{3} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{g,h} - m_Q \delta_{g,h} \right) \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{f,h}}{m_Q} + (\gamma_\sigma)_{h,f} \right) \right) \right)$$

$$- \left[- \frac{1}{8 \pi^3 (p_2^2 + I\epsilon) ((P_1 + p_2)^2 - m_Q^2 + I\epsilon)} \left((u_Q)_{\alpha,c}(\vec{P}_1) \right) \right] \quad (3)$$

$$\overline{(u_Q)_{\beta,d}(\vec{P}_2)} g_{\lambda\tau} \alpha^2 (\gamma^\kappa)_{l,e} (\gamma^\beta)_{d,k} (\gamma^\tau)_{k,l} (\gamma^\alpha)_{n,c} (\gamma^\sigma)_{f,m} (\gamma^\lambda)_{m,n} \left(\left(g_{\kappa\sigma} - \frac{(P_{1\kappa} + p_{2\kappa})(P_{1\sigma} + p_{2\sigma})}{m_Q^2} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{e,f} + m_Q \delta_{e,f} \right) + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{e,g}}{3 m_Q} + \frac{(\gamma_\kappa)_{e,g}}{3} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{g,h} - m_Q \delta_{g,h} \right) \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{f,h}}{m_Q} + (\gamma_\sigma)_{h,f} \right) \right) \delta(-P_2 + P_1) dp_2^4$$

In the user-information presented above in black, we see the form used for the propagator $\Delta(Q_{\kappa,e}(P_1 + p_2), Q_{\sigma,f}^I(-P_1 - p_2))$ of the 3/2 spin $Q_{\mu,a}$ field, and $\Delta(A_\lambda(p_2), A_\tau(-p_2))$ for the A_μ field including the gauge used for it. For that, *FeynmanDiagrams* is now using a gauge

term of the form $\frac{(1 - \xi_A) p_{2\lambda} p_{2\tau}}{p_2^2 + I\epsilon}$ where ξ_A is set to 1, 0, ∞ or to ξ_A itself, respectively for

the *Feynman*, *Landau unitary* and *arbitrary* gauges. By default the *Feynman* and *unitary* gauges are respectively used for massless and massive fields. For example, to run the same computation using the *Landau* gauge, you can use the new *propagatorgauge* option:

> *FeynmanDiagrams*(*L*, *incomingparticles* = [*Q*], *outgoing* = [*Q*], *numberofloops* = 1, *gauge* = *Landau*)

* Partial match of 'gauge' against keyword 'propagatorgauge'

$$\text{Propagator used: } \Delta(A_\lambda(p_2), A_\tau(-p_2)) = \frac{\text{I} \left(-g_{\lambda\tau} + \frac{p_{2\lambda} p_{2\tau}}{p_2^2 + \text{I}\epsilon} \right)}{p_2^2 + \text{I}\epsilon}$$

$$\begin{aligned} \text{Propagator used: } & \Delta(Q_{\kappa,e}(P_1 + p_2), Q_{\sigma,f}(-P_1 - p_2)) \\ &= \frac{1}{(P_1 + p_2)^2 - m_Q^2 + \text{I}\epsilon} \left(\text{I} \left(\left(g_{\kappa\sigma} - \frac{(P_{1\kappa} + p_{2\kappa})(P_{1\sigma} + p_{2\sigma})}{(P_1 + p_2)^2 + \text{I}\epsilon} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{e,f} + m_Q \delta_{e,f} \right) \right. \right. \\ & \left. \left. + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{e,g}}{3 m_Q} + \frac{(\gamma_\kappa)_{e,g}}{3} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{g,h} \right. \right. \right. \\ & \left. \left. \left. - m_Q \delta_{g,h} \right) \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{f,h}}{m_Q} + (\gamma_\sigma)_{h,f} \right) \right) \right) \end{aligned}$$

$$\begin{aligned} - \int & \frac{1}{8\pi^3 (p_2^2 + \text{I}\epsilon) ((P_1 + p_2)^2 - m_Q^2 + \text{I}\epsilon)} \left((\mathbf{u}_Q)_{\alpha,c}(\vec{P}_1) (\mathbf{u}_Q)_{\beta,d}(\vec{P}_2) \left(-g_{\lambda\tau} \right. \right. \\ & \left. \left. + \frac{p_{2\lambda} p_{2\tau}}{p_2^2 + \text{I}\epsilon} \right) \alpha^2 (\gamma^\kappa)_{l,e} (\gamma^\beta)_{d,k} (\gamma^\tau)_{k,l} (\gamma^\alpha)_{n,c} (\gamma^\sigma)_{f,m} (\gamma^\lambda)_{m,n} \left(\left(g_{\kappa\sigma} \right. \right. \right. \\ & \left. \left. - \frac{(P_{1\kappa} + p_{2\kappa})(P_{1\sigma} + p_{2\sigma})}{(P_1 + p_2)^2 + \text{I}\epsilon} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{e,f} + m_Q \delta_{e,f} \right) \right. \\ & \left. + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{e,g}}{3 m_Q} + \frac{(\gamma_\kappa)_{e,g}}{3} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{g,h} \right. \right. \\ & \left. \left. - m_Q \delta_{g,h} \right) \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{f,h}}{m_Q} + (\gamma_\sigma)_{h,f} \right) \right) \delta(-P_2 + P_1) dp_2^4 \end{aligned} \quad (4)$$

In the user-information presented above you see the different form used for the propagator $\Delta(A_\lambda(p_2), A_\tau(-p_2))$.

The arbitrary value of *propagatorgauge* option makes the value of ξ_A remain unset, so that you can set the gauge afterwards, possibly differently for each field, depending on convenience

> *FeynmanDiagrams(L, incomingparticles = [Q], outgoing = [Q], numberofloops = 1, gauge = arbitrary)*

* Partial match of 'gauge' against keyword 'propagatorgauge'

$$\text{Propagator used: } \left(\Delta(A_\lambda(p_2), A_\tau(-p_2)) = \frac{\text{I} \left(-g_{\lambda\tau} + \frac{(1 - \xi_A) p_{2\lambda} p_{2\tau}}{p_2^2 + \text{I}\epsilon} \right)}{p_2^2 + \text{I}\epsilon} \right)$$

$$\begin{aligned} \text{Propagator used: } & \left(\Delta(Q_{\kappa e}(P_1 + p_2), Q_{\sigma f}(-P_1 - p_2)) \right. \\ & = \frac{1}{(P_1 + p_2)^2 - m_Q^2 + \text{I}\epsilon} \left(\text{I} \left(\left(g_{\kappa\sigma} - \frac{(1 - \xi_Q)(P_{1\kappa} + p_{2\kappa})(P_{1\sigma} + p_{2\sigma})}{(P_1 + p_2)^2 - \xi_Q m_Q^2 + \text{I}\epsilon} \right) \left((P_{1\nu} \right. \right. \right. \\ & + p_{2\nu}) (\gamma^\nu)_{ef} + m_Q \delta_{ef} \left. \left. \left. + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{eg}}{3 m_Q} + \frac{(\gamma_\kappa)_{eg}}{3} \right) \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{gh} \right. \right. \right. \\ & \left. \left. \left. - m_Q \delta_{gh} \right) \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{fh}}{m_Q} + (\gamma_\sigma)_{hf} \right) \right) \right) \end{aligned}$$

$$\begin{aligned} - & \left[\frac{1}{8 \pi^3 (p_2^2 + \text{I}\epsilon) ((P_1 + p_2)^2 - m_Q^2 + \text{I}\epsilon)} \left((\mathbf{u}_Q)_{\alpha c}(\vec{P}_1) \overline{(\mathbf{u}_Q)_{\beta d}(\vec{P}_2)} \left(-g_{\lambda\tau} \right. \right. \right. \quad (5) \\ & + \frac{(1 - \xi_A) p_{2\lambda} p_{2\tau}}{p_2^2 + \text{I}\epsilon} \left. \left. \left. \alpha^2 (\gamma^\kappa)_{le} (\gamma^\beta)_{dk} (\gamma^\tau)_{kl} (\gamma^\alpha)_{nc} (\gamma^\sigma)_{fm} (\gamma^\lambda)_{mn} \delta(-P_2 \right. \right. \right. \\ & + P_1 \left. \left. \left. \left((P_{1\nu} + p_{2\nu}) (\gamma^\nu)_{ef} + m_Q \delta_{ef} \right) \left(g_{\kappa\sigma} - \frac{(P_{1\kappa} + p_{2\kappa})(P_{1\sigma} + p_{2\sigma})}{(P_1 + p_2)^2 - \xi_Q m_Q^2 + \text{I}\epsilon} \right) \left(\right. \right. \right. \\ & \left. \left. \left. - \xi_Q \right) \left((P_1 + p_2)^2 - \xi_Q m_Q^2 + \text{I}\epsilon \right)^{-1} \right) + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{eg}}{3 m_Q} + \frac{(\gamma_\kappa)_{eg}}{3} \right) \left((P_{1\nu} \right. \right. \end{aligned}$$

$$+ p_{2\nu}) \left(\gamma^{\nu} \right)_{g,h} - m_Q \delta_{g,h} \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{f,h}}{m_Q} + (\gamma_{\sigma})_{h,f} \right) \Big) \Big) dp_2^4$$

Note in the result above the occurrence of gauge fixing terms ξ_A and ξ_Q for each of the fields. These are implemented as *FeynmanDiagrams:-xi[field]* where *field* is any of *A* or *Q*. For example, you can now substitute each of them according to convenience, e.g. set the Feynman gauge for the massless field A_{μ} and the Landau gauge for the $Q_{\mu,a}$ field

> *subs(FeynmanDiagrams:-xi[A]=1, FeynmanDiagrams:-xi[Q]=0, (5))*

$$- \int \frac{1}{8 \pi^3 (p_2^2 + I\epsilon) ((P_1 + p_2)^2 - m_Q^2 + I\epsilon)} \left((\mathbf{u}_Q)_{\alpha,c} (\vec{P}_1) \right. \tag{6}$$

$$\left. \overline{(\mathbf{u}_Q)_{\beta,d} (\vec{P}_2)} g_{\lambda,\tau} \alpha^2 (\gamma^{\kappa})_{l,e} (\gamma^{\beta})_{d,k} (\gamma^{\tau})_{k,l} (\gamma^{\alpha})_{n,c} (\gamma^{\sigma})_{f,m} (\gamma^{\lambda})_{m,n} \delta(-P_2} \right.$$

$$+ P_1 \left(\left((P_{1\nu} + p_{2\nu}) (\gamma^{\nu})_{e,f} + m_Q \delta_{e,f} \right) \left(g_{\kappa,\sigma} - (P_{1\kappa} + p_{2\kappa}) (P_{1\sigma} \right. \right.$$

$$+ p_{2\sigma}) \frac{1}{(P_1 + p_2)^2 + I\epsilon} \Big) + \left(\frac{(P_{1\kappa} + p_{2\kappa}) \delta_{e,g}}{3 m_Q} + \frac{(\gamma_{\kappa})_{e,g}}{3} \right) \left((P_{1\nu} \right.$$

$$+ p_{2\nu}) (\gamma^{\nu})_{g,h} - m_Q \delta_{g,h} \left(\frac{(P_{1\sigma} + p_{2\sigma}) \delta_{f,h}}{m_Q} + (\gamma_{\sigma})_{h,f} \right) \Big) \Big) dp_2^4$$

An extra level of flexibility is given by the new option *usepropagators* which, when set to false, makes *FeynmanDiagrams* return the abstract form $\Delta(Q_{\kappa,e}(P_1 + p_2), Q_{\sigma,f}(-P_1 - p_2))$ instead of any specific form for the propagator

> *FeynmanDiagrams(L, incomingparticles = [Q], outgoing = [Q], numberofloops = 1, usepropagators = false)*

$$- \int \frac{1}{8 \pi^3} \left((\mathbf{u}_Q)_{\alpha,c} (\vec{P}_1) \right. \tag{7}$$

$$\left. \overline{(\mathbf{u}_Q)_{\beta,d} (\vec{P}_2)} \alpha^2 (\gamma^{\kappa})_{j,e} (\gamma^{\beta})_{d,i} (\gamma^{\tau})_{i,j} (\gamma^{\alpha})_{h,c} (\gamma^{\sigma})_{f,g} (\gamma^{\lambda})_{g,h} \delta(-P_2} \right.$$

$$+ P_1 \Delta(A_{\lambda}(p_2), A_{\tau}(-p_2)) \Delta(Q_{\kappa,e}(P_1 + p_2), Q_{\sigma,f}(-P_1 - p_2)) \Big) dp_2^4$$

In this result, $\Delta(A_{\lambda}(p_2), A_{\tau}(-p_2))$ is implemented as a new *FeynmanDiagrams:-Propagator*

function; there are two occurrences

> `indets((7), FeynmanDiagrams:-Propagator(anything, anything))`

$$\{\Delta(A_\lambda(p_2), A_\tau(-p_2)), \Delta(Q_{\kappa e}(P_1 + p_2), Q_{\sigma f}(-P_1 - p_2))\} \quad (8)$$

Finally, the convention is to add a normalization factor for each external leg (incoming or outgoing particle). That external normalization depends on the spin of the field [3]. For example, in the result (7) above, that is the origin of the factor $8\pi^3$ in the denominator. In some contexts, however, it is usual or convenient to compute without these normalization factors. For that purpose you can use the new option, `externalnormalization=false`

> `FeynmanDiagrams(L, incomingparticles = [Q], outgoing = [Q], numberofloops = 1, usepropagators = false, externalnormalization = false)`

$$- \int - (u_Q)_{\alpha, c}(\vec{P}_1) \overline{(u_Q)_{\beta, d}(\vec{P}_2)} \alpha^2 (\gamma^\kappa)_{j, e} (\gamma^\beta)_{d, i} (\gamma^\tau)_{i, j} (\gamma^\alpha)_{h, c} (\gamma^\sigma)_{f, g} (\gamma^\lambda)_{g, h} \delta(-P_2 + P_1) \Delta(A_\lambda(p_2), A_\tau(-p_2)) \Delta(Q_{\kappa e}(P_1 + p_2), Q_{\sigma f}(-P_1 - p_2)) dp_2^4 \quad (9)$$

Examples - new FeynmanIntegral module

> `restart; with(Physics) :`

> `Setup(mathematicalnotation = true, coordinates = X, quantumoperators = phi)`

Systems of spacetime coordinates are: $\{X = (x1, x2, x3, x4)\}$

$$[\text{coordinatesystems} = \{X\}, \text{mathematicalnotation} = \text{true}, \text{quantumoperators} = \{\phi\}] \quad (10)$$

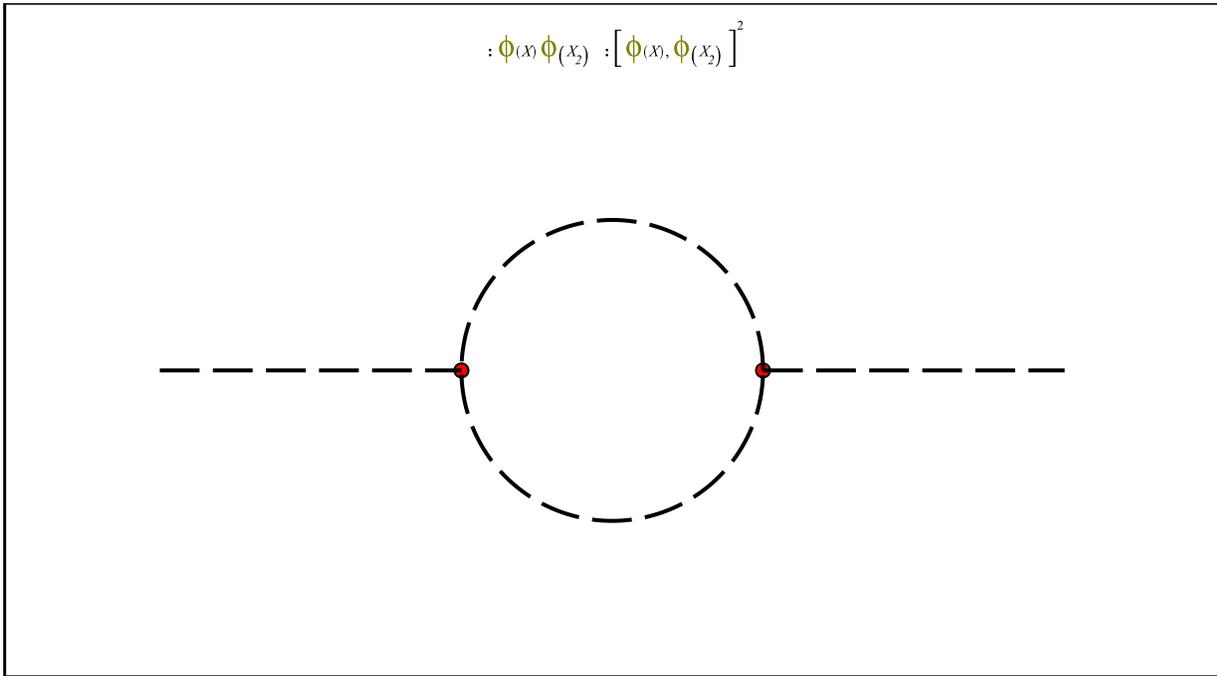
Let L be the interaction Lagrangian

> $L := \lambda \phi(X)^3$

$$L := \lambda \phi(X)^3 \quad (11)$$

A process with one incoming and one outgoing particle a 1-loop

> `FeynmanDiagrams(L, incomingparticles = [phi], outgoingparticles = [phi], numberofloops = 1, diagrams)`



$$\int \frac{9 \lambda^2 \delta(-P_2 + P_1)}{8 \pi^3 \sqrt{E_1 E_2} (p_2^2 - m_\phi^2 + i\epsilon) ((P_1 + p_2)^2 - m_\phi^2 + i\epsilon)} dp_2^4 \quad (12)$$

Load the package

> `with(FeynmanIntegral)`

`[Evaluate, Parametrize, ε]` (13)

Parametrize the Feynman integral (12) using Feynman parameters

> `Parametrize((12))`

$$\int \frac{9 \lambda^2 \delta(-P_2 + P_1) \left(\int_0^1 \int_0^1 \frac{\delta(-1 + \xi_1 + \xi_2)}{(\xi_1 (-m_\phi^2 + p_2^2) + \xi_2 ((P_1 + p_2)^2 - m_\phi^2))^2} d\xi_2 d\xi_1 \right)}{8 \pi^3 \sqrt{E_1 E_2}} dp_2^4 \quad (14)$$

The `Evaluate` command is used to evaluate *Feynman* integrals. In the following input, we request for the momentum integration to not be performed, receiving, basically, the result above but with the integrations over the Feynman parameters swapped with the one over momentum

> `Evaluate((12), momentumintegration=false)`

(15)

$$\frac{1}{8 \pi^3 \sqrt{E_1 E_2}} \left(9 \lambda^2 \delta(-P_2 + P_1) \left(\int_0^1 \int_0^1 \delta(-1 + \xi_1 + \xi_2) \right) \right. \\ \left. \frac{1}{\left(\xi_1 (-m_\phi^2 + p_2^2) + \xi_2 ((P_1 + p_2)^2 - m_\phi^2) \right)^2} dp_2^4 d\xi_2 d\xi_1 \right) \quad (15)$$

By default, however, *Evaluate* will perform the momentum integration dp_2^4 but in $D - 2\epsilon$ dimensions, so that its divergences get expressed as poles of [Gamma functions](#), and also the Feynman parameters integration, returning a result in terms of the dimensional parameter ϵ

> *Evaluate*((12))

$$\frac{1}{\pi \sqrt{E_1 E_2}} \left(-\frac{9I}{2} \lambda^2 \delta(-P_2 + P_1) \pi^{2-\epsilon} \Gamma(\epsilon) \left(\sum_{n=0}^{\infty} \sum_{l=0}^{\infty} \left(\frac{(-1)^n \Gamma(\epsilon + n + n_l) m_\phi^{-2n_l - 2\epsilon - 2n} P_1^{2n + 2n_l} \Gamma(2n + n_l + 1)}{4 \Gamma(2 + 2n + n_l) \pi^2 \Gamma(\epsilon) \Gamma(1 + n) \Gamma(1 + n_l)} \right) \right) \right) \quad (16)$$

The dimensional parameter is represented by *FeynmanIntegral:-varepsilon* and, because the *FeynmanIntegral* package is loaded, you can refer to it directly as *varepsilon*. To compute this integral *expanding* the dimensional parameter, keeping terms up to $O(\epsilon)$, and not integrating over the Feynman parameters, use

> *Evaluate*((12), *expanddimension = true*)

* Partial match of 'expanddimension' against keyword 'expanddimensionalparameter'

$$\frac{1}{\pi \sqrt{E_1 E_2} \epsilon} \left(-\frac{9I}{8} \lambda^2 \delta(-P_2 + P_1) \left(\int_0^1 \int_0^1 \delta(-1 + \xi_1 + \xi_2) \left(\epsilon \ln(\pi) + \epsilon \ln \left(\left(\xi_2^2 - \xi_2 \right) P_1^2 + m_\phi^2 \right) + \epsilon \gamma - 1 \right) d\xi_2 d\xi_1 \right) \right) \quad (17)$$

The same integral can be computed using *alpha* parameters

> *Evaluate*((12), *parameters = alpha*)

* Partial match of 'parameters' against keyword 'kindofparameters'

$$-\frac{1}{8 \pi^3 \sqrt{E_1 E_2}} \left(9 \lambda^2 \delta(-P_2 + P_1) (-1)^{-4\epsilon} \pi^{2-\epsilon} \int_0^\infty \right) \quad (18)$$

$$\int_0^{\infty} \left(e^{I(-\alpha_1 - \alpha_2) m_\phi^2 + I\alpha_2 P_1^2} e^{\frac{I(\alpha_1 + \alpha_2)(\epsilon - 1)\pi - 2IP_1^2\alpha_2^2}{2\alpha_1 + 2\alpha_2}} (\alpha_1 + \alpha_2)^{\epsilon - 2} d\alpha_1 d\alpha_2 \right)$$

> Evaluate((12), parameters = alpha, expanddimension = true)

* Partial match of 'parameters' against keyword 'kindofparameters'

* Partial match of 'expanddimension' against keyword 'expanddimensionalparameter'

$$\frac{9I}{8} \lambda^2 \delta(-P_2 + P_1) \left(\int_0^{\infty} \int_0^{\infty} \frac{e^{I(-\alpha_1 - \alpha_2) m_\phi^2 + I\alpha_2 P_1^2} e^{\frac{-IP_1^2\alpha_2^2}{\alpha_1 + \alpha_2}}}{(\alpha_1 + \alpha_2)^2} d\alpha_1 d\alpha_2 \right) \pi \sqrt{E_1 E_2} \quad (19)$$

The Parametrize also work using the α -parameters

> Parametrize((12), kindofparameters = alpha)

$$\int \frac{9 \lambda^2 \delta(-P_2 + P_1) \left(\int_0^{\infty} \int_0^{\infty} -e^{Ip_2(2P_1\alpha_2 + p_2\alpha_1 + p_2\alpha_2)} e^{I(-\alpha_1 - \alpha_2) m_\phi^2 + I\alpha_2 P_1^2} d\alpha_1 d\alpha_2 \right)}{8 \pi^3 \sqrt{E_1 E_2}} dp_2^4 \quad (20)$$

References

- [1] Smirnov, V.A., **Feynman Integral Calculus**. Springer, 2006.
- [2] Weinzierl, S., **The Art of Computing Loop Integrals**, <https://arxiv.org/abs/hep-ph/0604068v1>
- [3] Weinberg, S., **The Quantum Theory Of Fields**. Cambridge University Press, 2005.
- [4] Bogoliubov, N.N., and Shirkov, D.V. **Quantum Fields**. Benjamin Cummings, 1982.

Simplification of products of Dirac matrices

The simplification of products of [Dirac matrices](#) got stronger in Maple 2021 in several ways. As an example, set $\sigma_{\mu, \nu}$ as a quantum operator (noncommutative) tensor proportional to the commutator of Dirac matrices

> restart : with(Physics) :

> Setup(*op* = sigma, *tensors* = sigma[μ, ν])

* Partial match of 'op' against keyword 'quantumoperators'

$$\overline{\left[\text{quantumoperators} = \{\sigma\}, \text{tensors} = \{\gamma_\mu, \sigma_\mu, \partial_\mu, g_{\mu, \nu}, \sigma_{\mu, \nu}, \epsilon_{\alpha, \beta, \mu, \nu}\} \right]} \quad (21)$$

Enter the definition of $\sigma^{\mu, \nu}$

> Define $\left(\sigma^{\mu, \nu} = \frac{I(\gamma^\mu \gamma^\nu - \gamma^\nu \gamma^\mu)}{2} \right)$

Defined objects with tensor properties

$$\{\gamma_\mu, \sigma_\mu, \partial_\mu, g_{\mu, \nu}, \sigma_{\mu, \nu}, \epsilon_{\alpha, \beta, \mu, \nu}\} \quad (22)$$

Consider proving the following value for the commutator $\left[\sigma^{\mu, \nu}, \sigma^{\alpha, \beta} \right]_-$

> Commutator $\left(\sigma^{\mu, \nu}, \sigma^{\alpha, \beta} \right) = 2 I \left(g^{\nu, \alpha} \sigma^{\mu, \beta} - g^{\mu, \alpha} \sigma^{\nu, \beta} + g^{\mu, \beta} \sigma^{\nu, \alpha} - g^{\nu, \beta} \sigma^{\mu, \alpha} \right)$

$$\left[\sigma^{\mu, \nu}, \sigma^{\alpha, \beta} \right]_- = 2 I \left(g^{\alpha, \mu} \sigma^{\beta, \nu} - g^{\alpha, \nu} \sigma^{\beta, \mu} - g^{\beta, \mu} \sigma^{\alpha, \nu} + g^{\beta, \nu} \sigma^{\alpha, \mu} \right) \quad (23)$$

Insert σ 's definition in this expression

> sigma[definition]

$$\sigma^{\mu, \nu} = \frac{I}{2} \left(\gamma^\mu \gamma^\nu - \gamma^\nu \gamma^\mu \right) \quad (24)$$

> SubstituteTensor(sigma[definition], (23))

$$\left[\frac{I}{2} \left(\gamma^\mu \gamma^\nu - \gamma^\nu \gamma^\mu \right), \frac{I}{2} \left(\gamma^\alpha \gamma^\beta - \gamma^\beta \gamma^\alpha \right) \right]_- = 2 I \left(\frac{I g^{\alpha, \mu} \left(\gamma^\beta \gamma^\nu - \gamma^\nu \gamma^\beta \right)}{2} - \frac{I g^{\alpha, \nu} \left(\gamma^\beta \gamma^\mu - \gamma^\mu \gamma^\beta \right)}{2} - \frac{I g^{\beta, \mu} \left(\gamma^\alpha \gamma^\nu - \gamma^\nu \gamma^\alpha \right)}{2} + \frac{I g^{\beta, \nu} \left(\gamma^\alpha \gamma^\mu - \gamma^\mu \gamma^\alpha \right)}{2} \right) \quad (25)$$

Expand the commutator and all the products

> Expand((25))

$$\frac{\gamma^\beta \gamma^\nu \gamma^\alpha \gamma^\mu}{2} + \frac{\gamma^\nu \gamma^\beta \gamma^\mu \gamma^\alpha}{2} - \frac{\gamma^\beta \gamma^\alpha \gamma^\mu \gamma^\nu}{2} + \frac{\gamma^\alpha \gamma^\beta \gamma^\mu \gamma^\nu}{2} + \frac{\gamma^\nu \gamma^\beta \gamma^\alpha \gamma^\mu}{2} \quad (26)$$

$$\begin{aligned}
& + \frac{\gamma^\alpha \gamma^\mu \gamma^\beta \gamma^\nu}{2} + \frac{\gamma^\mu \gamma^\alpha \gamma^\nu \gamma^\beta}{2} + \frac{\gamma^\mu \gamma^\alpha \gamma^\beta \gamma^\nu}{2} - \frac{\gamma^\alpha \gamma^\nu \gamma^\beta \gamma^\mu}{2} \\
& - \frac{\gamma^\nu \gamma^\alpha \gamma^\mu \gamma^\beta}{2} - \frac{\gamma^\nu \gamma^\alpha \gamma^\beta \gamma^\mu}{2} - \frac{\gamma^\mu \gamma^\beta \gamma^\nu \gamma^\alpha}{2} - \frac{\gamma^\mu \gamma^\beta \gamma^\alpha \gamma^\nu}{2} \\
& - \frac{\gamma^\alpha \gamma^\beta \gamma^\nu \gamma^\mu}{2} + \frac{\gamma^\beta \gamma^\alpha \gamma^\nu \gamma^\mu}{2} - \frac{\gamma^\beta \gamma^\mu \gamma^\alpha \gamma^\nu}{2} = -g^{\alpha, \mu} \gamma^\beta \gamma^\nu + g^{\alpha, \mu} \gamma^\nu \gamma^\beta \\
& + g^{\alpha, \nu} \gamma^\beta \gamma^\mu - g^{\alpha, \nu} \gamma^\mu \gamma^\beta + g^{\beta, \mu} \gamma^\alpha \gamma^\nu - g^{\beta, \mu} \gamma^\nu \gamma^\alpha - g^{\beta, \nu} \gamma^\alpha \gamma^\mu + \\
& g^{\beta, \nu} \gamma^\mu \gamma^\alpha
\end{aligned}$$

Proving that, in this expression, the left-hand side is equal to the right-hand side involves the simplification of sums of products of four (on the left-hand side) and of two (on the right-hand side) [Dirac matrices](#) taking into account the algebra rules they satisfy

> *Library:-DefaultAlgebraRules(Dirac)*

$$[\gamma_\mu, \gamma_\nu]_+ = 2 g_{\mu, \nu} \quad (27)$$

In Maple 2021, this simplification can be performed in one go by taking the left-hand side minus the right-hand side and sending that to the simplifier

> *Simplify((lhs - rhs) (26))*

$$0 \quad (28)$$

This simplification can now also be performed in steps. Take the commutator on the left-hand side of (23)

> *lhs(23)*

$$[\sigma^{\mu, \nu}, \sigma^{\alpha, \beta}]_- \quad (29)$$

> *Expand(29)*

$$\sigma^{\mu, \nu} \sigma^{\alpha, \beta} - \sigma^{\alpha, \beta} \sigma^{\mu, \nu} \quad (30)$$

> *SubstituteTensor(sigma[definition], (30))*

$$\begin{aligned}
& \frac{1}{2} (\gamma^\mu \gamma^\nu - \gamma^\nu \gamma^\mu) \frac{1}{2} (\gamma^\alpha \gamma^\beta - \gamma^\beta \gamma^\alpha) - \frac{1}{2} (\gamma^\alpha \gamma^\beta - \gamma^\beta \gamma^\alpha) \frac{1}{2} (\gamma^\mu \gamma^\nu - \gamma^\nu \\
& \gamma^\mu)
\end{aligned} \quad (31)$$

Although this intermediate result (31) involves products of four Dirac matrices, their combination is such that the result can be expressed in terms of products of only two of matrices

> *Simplify*((31))

$$-2 g^{\beta v} \gamma^{\alpha} \gamma^{\mu} + 2 g^{\beta \mu} \gamma^{\alpha} \gamma^v - 2 g^{\alpha v} \gamma^{\mu} \gamma^{\beta} + 2 g^{\alpha \mu} \gamma^v \gamma^{\beta} \quad (32)$$

This result is different from the expanded form of the right-hand side of (23)

> *Expand*(*SubstituteTensor*(*sigma*[*definition*], *rhs*((23))))

$$-g^{\alpha \mu} \gamma^{\beta} \gamma^v + g^{\alpha \mu} \gamma^v \gamma^{\beta} + g^{\alpha v} \gamma^{\beta} \gamma^{\mu} - g^{\alpha v} \gamma^{\mu} \gamma^{\beta} + g^{\beta \mu} \gamma^{\alpha} \gamma^v - g^{\beta \mu} \gamma^v \gamma^{\alpha} - g^{\beta v} \gamma^{\alpha} \gamma^{\mu} + g^{\beta v} \gamma^{\mu} \gamma^{\alpha} \quad (33)$$

The Simplifier, however, can not only prove that (32) - (33) is equal to zero but can also depart from (33) and arrive at (32)

> *Simplify*((33))

$$-2 g^{\beta v} \gamma^{\alpha} \gamma^{\mu} + 2 g^{\beta \mu} \gamma^{\alpha} \gamma^v - 2 g^{\alpha v} \gamma^{\mu} \gamma^{\beta} + 2 g^{\alpha \mu} \gamma^v \gamma^{\beta} \quad (34)$$

That is so because, in Maple 2021, a new normal form for the ordering in products of noncommutative operators got implemented. The computation above also indicates that there are two different kinds of simplifications at work here: one that maps a sum or products of four Dirac matrices into simpler products of also four matrices, and another one that maps onto a sum of products of two Dirac matrices. To see that, consider again the expanded form (31) of the commutator $[\sigma^{\mu, v}, \sigma^{\alpha, \beta}]$

> (31)

$$-\frac{(\gamma^{\mu} \gamma^v - \gamma^v \gamma^{\mu})(\gamma^{\alpha} \gamma^{\beta} - \gamma^{\beta} \gamma^{\alpha})}{4} + \frac{(\gamma^{\alpha} \gamma^{\beta} - \gamma^{\beta} \gamma^{\alpha})(\gamma^{\mu} \gamma^v - \gamma^v \gamma^{\mu})}{4} \quad (35)$$

The default approach maps the sum of products of four Dirac matrices into products of two Dirac matrices using the algebra rules they satisfy

> *Simplify*((31))

$$-2 g^{\beta v} \gamma^{\alpha} \gamma^{\mu} + 2 g^{\beta \mu} \gamma^{\alpha} \gamma^v - 2 g^{\alpha v} \gamma^{\mu} \gamma^{\beta} + 2 g^{\alpha \mu} \gamma^v \gamma^{\beta} \quad (36)$$

One can also use those algebra rules to only sort the products with some preferred ordering as pivot (new option in [Simplify](#)), resulting in products of also four matrices

> *Simplify*((31), *ordering* = $[\gamma^{\alpha}, \gamma^{\beta}, \gamma^{\mu}, \gamma^v]$)

$$-\gamma^{\mu} \gamma^v \gamma^{\alpha} \gamma^{\beta} + \gamma^{\alpha} \gamma^{\beta} \gamma^{\mu} \gamma^v \quad (37)$$

This result is not just an expansion of (35),

> *Expand*((35))

$$\begin{aligned}
& -\frac{\gamma^\mu \gamma^\nu \gamma^\alpha \gamma^\beta}{4} + \frac{\gamma^\mu \gamma^\nu \gamma^\beta \gamma^\alpha}{4} + \frac{\gamma^\nu \gamma^\mu \gamma^\alpha \gamma^\beta}{4} - \frac{\gamma^\nu \gamma^\mu \gamma^\beta \gamma^\alpha}{4} + \frac{\gamma^\alpha \gamma^\beta \gamma^\mu \gamma^\nu}{4} \\
& -\frac{\gamma^\alpha \gamma^\beta \gamma^\nu \gamma^\mu}{4} - \frac{\gamma^\beta \gamma^\alpha \gamma^\mu \gamma^\nu}{4} + \frac{\gamma^\beta \gamma^\alpha \gamma^\nu \gamma^\mu}{4}
\end{aligned} \tag{38}$$

but the result of *sorting first* all the products in (35), using the algebra rules $[\gamma_\mu, \gamma_\nu]_+ = 2g_{\mu, \nu}$ and only then expand

$$\begin{aligned}
& > \text{SortProducts}\left(\text{(35)}, [\gamma^\alpha, \gamma^\beta, \gamma^\mu, \gamma^\nu]\right) \\
& -\gamma^\alpha \left(\gamma^\beta \gamma^\mu \gamma^\nu - [\gamma^\beta, \gamma^\mu \gamma^\nu]_- \right) + g^{\alpha\beta} \gamma^\mu \gamma^\nu - g^{\mu, \nu} g^{\alpha\beta} + g^{\mu, \nu} \gamma^\alpha \gamma^\beta \\
& + [\gamma^\alpha, \gamma^\mu \gamma^\nu]_- \gamma^\beta + \frac{(2\gamma^\alpha \gamma^\beta - 2g^{\alpha\beta})(2\gamma^\mu \gamma^\nu - 2g^{\mu, \nu})}{4}
\end{aligned} \tag{39}$$

> Expand((39))

$$-\gamma^\mu \gamma^\nu \gamma^\alpha \gamma^\beta + \gamma^\alpha \gamma^\beta \gamma^\mu \gamma^\nu \tag{40}$$

Proving identities is a key - sometimes, as in this case related to Dirac matrices - non-trivial operation. Another way of verifying identities like (23) is to compute *all the components* of the tensorial equation,

> (Expand@lhs - rhs) ((23))

$$\begin{aligned}
& \sigma^{\mu, \nu} \sigma^{\alpha, \beta} - \sigma^{\alpha, \beta} \sigma^{\mu, \nu} - 2I \left(g^{\alpha, \mu} \sigma^{\beta, \nu} - g^{\alpha, \nu} \sigma^{\beta, \mu} - g^{\beta, \mu} \sigma^{\alpha, \nu} + g^{\beta, \nu} \right. \\
& \left. \sigma^{\alpha, \mu} \right)
\end{aligned} \tag{41}$$

each of which is, in turn, a 4 x 4 matrix. If the identity is true, for each value of each of the four spacetime free indices α, β, μ, ν , we expect that, performing all the matricial operations, we get for result a matrix 4 x 4 matrix of zeros. That computation can be performed with [TensorArray](#) using its option *performmatrixoperations*

> TensorArray((41), performmatrixoperations, output = setofequations)

$$\left\{ \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0 \right\} \tag{42}$$

So the identity is true. Note that $4^4 = 256$ resulting matrices. This computation for each value of the four spacetime indices can be displayed so compactly because of using the option *output = setofequations*, and repeated elements in a set are automatically removed.

Thus, the 256 matrices of zeros, each of which is equal to 0, appear as only one.

As a more challenging computation, less of a black box, one can express everything in (41) in terms of Dirac matrices

> `Expand(SubstituteTensor(sigma[definition], (41)))`

$$\begin{aligned}
 & -\frac{\gamma^\mu \gamma^\nu \gamma^\alpha \gamma^\beta}{4} + \frac{\gamma^\mu \gamma^\nu \gamma^\beta \gamma^\alpha}{4} + \frac{\gamma^\nu \gamma^\mu \gamma^\alpha \gamma^\beta}{4} - \frac{\gamma^\nu \gamma^\mu \gamma^\beta \gamma^\alpha}{4} + \frac{\gamma^\alpha \gamma^\beta \gamma^\mu \gamma^\nu}{4} \\
 & - \frac{\gamma^\alpha \gamma^\beta \gamma^\nu \gamma^\mu}{4} - \frac{\gamma^\beta \gamma^\alpha \gamma^\mu \gamma^\nu}{4} + \frac{\gamma^\beta \gamma^\alpha \gamma^\nu \gamma^\mu}{4} + g^{\alpha,\mu} \gamma^\beta \gamma^\nu - g^{\alpha,\mu} \gamma^\nu \gamma^\beta \\
 & - g^{\alpha,\nu} \gamma^\beta \gamma^\mu + g^{\alpha,\nu} \gamma^\mu \gamma^\beta - g^{\beta,\mu} \gamma^\alpha \gamma^\nu + g^{\beta,\mu} \gamma^\nu \gamma^\alpha + g^{\beta,\nu} \gamma^\alpha \gamma^\mu - \\
 & g^{\beta,\nu} \gamma^\mu \gamma^\alpha
 \end{aligned} \tag{43}$$

For each of the four values of each of the four indices α, β, μ, ν , use now the underlying Dirac matrices

> `Library:-RewriteInMatrixForm(Dgamma[~])`

$$\gamma^\mu \tag{44}$$

$$\begin{aligned}
 & = \left[\left[\begin{array}{cccc} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{array} \right], \left[\begin{array}{cccc} 0 & 0 & 0 & -I \\ 0 & 0 & I & 0 \\ 0 & I & 0 & 0 \\ -I & 0 & 0 & 0 \end{array} \right], \left[\begin{array}{cccc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array} \right], \right. \\
 & \left. \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{array} \right] \right]
 \end{aligned}$$

and compute *all of the 4 x 4 x 4 x 4 matrices* that result from multiplying the four and two Dirac matrices in each term of (43) (that is what [TensorArray](#) did, internally, to compute the result (42))

> `TensorArray((43), performmatrixoperations, output=setofequations)`

$$\left\{ \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0 \right\} \tag{45}$$

Manipulation and simplification of expressions involving non-commutative tensor operators

Significant improvements under the hood happened regarding of simplification in the presence of non-commutative tensor operators. The simplifications need to take into account commutator rules, symmetries under permutation of indices of tensorial subexpressions, and use Einstein's sum rule for repeated indices. Related to that, Maple 2021 includes relevant enhancements in the [Simplify](#), [SubstituteTensor](#) and [SortProducts](#) commands.

With these developments in place, it is now possible, for example, to systematically derive, step-by-step, the SO(4) symmetry of the hydrogen atom and its spectrum entering only the main definition formulas, followed by only simplification commands, and without using previous knowledge of the result. A presentation of this work is now in the [arXiv](#) and is submitted for publication in the journal *Computer Physics Communications* as a novel way of tackling these kinds of problems using computers.

In brief, deriving the SO(4) symmetry of the hydrogen atom is about deriving the following four commutator rules between its Hamiltonian H , the angular momentum tensor L_m and the Runge-Lenz tensor Z_n ,

$$[H, L_n]_- = 0$$

$$[H, Z_n]_- = 0$$

$$[L_m, Z_n]_- = i \hbar \epsilon_{m, n, o} Z_o$$

$$[Z_m, Z_n]_- = -2 \frac{i \hbar}{m_e} H \epsilon_{m, n, o} L_o$$

where for hermiticity purpose L_m is defined in quantum mechanics as the symmetrized form

$$\vec{Z} = \frac{1}{2 m_e} (\vec{L} \times \vec{p} - \vec{p} \times \vec{L}) + \kappa \frac{\vec{r}}{r}$$

Since H commutes with both \vec{L} and \vec{Z} , defining

$$M_n = \sqrt{-\frac{m_e}{2H}} Z_n,$$

these commutation rules can be rewritten as

$$[L_m, L_n]_- = i \hbar \epsilon_{m, n, o} L_o$$

$$[L_m, M_n]_- = i \hbar \epsilon_{m, n, o} M_o$$

$$[M_m, M_n]_- = i \hbar \epsilon_{m, n, o} L_o$$

This set constitutes the Lie algebra of the SO(4) group. As an illustration of the computational capabilities in Maple 2021, in what follows we derive the first two of these commutators, $[H, L_n]_- = [H, Z_n]_- = 0$, departing from basic principles and followed by applying only a few simplification commands to equation labels.

Setting the problem

Formulating the problem requires loading the *Physics* package, and we set the imaginary unit to be represented by a lowercase Latin i letter instead of the default uppercase I.

> *restart; with(Physics) : interface(imaginaryunit=i) :*

The context for this problem is Cartesian coordinates and a 3D Euclidean space where all of $\{\hbar, \kappa, m_e\}$ are real objects. We chose lowercase letters to represent tensor indices and the use of automatic simplification (i.e., *automatically simplify the size of everything being displayed*)

> *Setup(coordinates=cartesian, realobjects={\hbar, \kappa, m_e}, automaticsimplification=true, dimension=3, metric=Euclidean, spacetimeindices=lowercaselatin, quiet)*

[automaticsimplification=true, coordinatesystems={X}, dimension=3, metric={{(1, 1) (46)}
= 1, (2, 2) = 1, (3, 3) = 1}, realobjects={\hbar, \kappa, m_e, x, y, z}, spacetimeindices
=lowercaselatin]

Next, we set the quantum Hermitian operators (not *Z*, we derive that property for it further below) and related commutators:

– the dimensionless potential $V = \frac{1}{r}$ is assumed to commute with position, not with momentum - the commutation rule with p_k is derived further below;

– the commutator rules between position X_n on the one hand, and linear p_k and angular momentum L_k on the other hand, are the departure point, entered using the inert form of the [Commutator](#) command. Tensors are indexed using the standard Maple notation for indexation, [].

> *Setup(quantumoperators={Z}, hermitianoperators={V, H, L, X, p},*

$$\begin{aligned}
\text{algebrarules} = \{ & \\
& \%Commutator(p[k], p[l]) = 0, \\
& \%Commutator(X[k], p[l]) = i \cdot \hbar \cdot g_{-}[k, l], \\
& \%Commutator(L[j], L[k]) = i \cdot \hbar \cdot LeviCivita[j, k, n] \cdot L[n], \\
& \%Commutator(p[j], L[k]) = i \cdot \hbar \cdot LeviCivita[j, k, n] \cdot p[n], \\
& \%Commutator(X[j], L[k]) = i \cdot \hbar \cdot LeviCivita[j, k, n] \cdot X[n], \\
& \%Commutator(X[k], V(X)) = 0 \} \\
[\text{algebrarules} = \{ [L_j, L_k]_- = i \hbar \epsilon_{j, k, n} L_n, [p_j, L_k]_- = i \hbar \epsilon_{j, k, n} p_n, [p_k, p_l]_- = 0, [X_j, L_k]_- & \quad (47) \\
= i \hbar \epsilon_{j, k, n} X_n, [X_k, p_l]_- = i \hbar g_{k, l}, [X_k, V(X)]_- = 0 \}, \text{hermitianoperators} = \{H, L, V, p, x, & \\
y, z\}, \text{quantumoperators} = \{H, L, V, Z, p, x, y, z\} & \}
\end{aligned}$$

Define the *tensor* quantum operators representing the linear momentum, angular momentum and the Runge-Lenz vectors

> Define($p[k]$, $L[k]$, $Z[k]$, quiet)

$$\left\{ \gamma_a, L_k, \sigma_a, Z_k, \partial_a, g_{a, b}, p_k, \epsilon_{a, b, c}, X_a \right\} \quad (48)$$

For readability, avoid redundant display of functionality

> CompactDisplay($(V, G)(X)$)

$$\begin{aligned}
& V(X) \text{ will now be displayed as } V \\
& G(X) \text{ will now be displayed as } G \quad (49)
\end{aligned}$$

The Hamiltonian for the hydrogen atom is entered as

$$\begin{aligned}
> H = \frac{p[l]^2}{2 \cdot m_e} - \kappa \cdot V(X) \\
H = \frac{p_l^2}{2 m_e} - \kappa V \quad (50)
\end{aligned}$$

Definition of $V(X)$ and related identities

We use the dimensionless potential $V(X)$

$$\begin{aligned}
> V(X) = \frac{1}{(X[l]^2)^{\frac{1}{2}}} \\
V = (X_l^2)^{-\frac{1}{2}} \quad (51)
\end{aligned}$$

The gradient of $V(X)$ is

> $d_{-}[n](51)$

$$\partial_n(V) = -\left(X_l^2\right)^{-\frac{3}{2}} X_n \quad (52)$$

where we note that all these commands (including product and power), *distribute over equations*. So that

> `subs((rhs = lhs)((51)^3), (52))`

$$\partial_n(V) = -V^3 X_n \quad (53)$$

Equivalently, from (51) one can deduce $V^3 X_l^2 = V$ that will be used afterwards

> `(rhs = lhs)\left(\frac{V(X)^3}{(51)^2}\right)`

$$V^3 X_l^2 = V \quad (54)$$

The commutation rules between linear and angular momentum and of the potential V(X)

By definition

> `L[q] = LeviCivita[q, m, n] · X[m] · p[n]`

$$L_q = \epsilon_{m, n, q} X_m p_n \quad (55)$$

so, since the system knows about $[X_k, V(X)]_- = 0$, we get

> `Commutator((55), V(X))`

$$[L_q, V]_- = \epsilon_{m, n, q} X_m [p_n, V]_- \quad (56)$$

To derive the value of $[p_n, V]_-$, as usual in paper and pencil computations we set p_n as a *differentialoperator* and introduce an arbitrary test function $G(X)$

> `Setup(differentialoperators = {[p[k], [x, y, z]])}`

$$[differentialoperators = {[p_k [X]]}] \quad (57)$$

Applying now to $G(X)$ the differential operator p_n found in the commutator of the right-hand side of (56)

> `(lhs = Library:-ApplyProductsOfDifferentialOperators@rhs)((56) · G(X))`

$$[L_q, V]_- G = \epsilon_{m, n, q} X_m (p_n(VG) - Vp_n(G)) \quad (58)$$

where `ApplyProductsOfDifferentialOperators` is a command of the *Physics:-Library*. The result of $p_l(G(X))$ is not known to the system at this point. Define then an explicit representation for p_n as the differential operator in configuration space $p_n = -i \hbar \partial_n$

> $p := u \rightarrow -i \hbar \cdot d_{op(procname)}(u)$

$$p := u \mapsto -i \hbar \partial_{op(procname)}(u) \quad (59)$$

where in the above $op(procname)$ represents the index with which the differential operator p_n is called. With this definition, the right-hand side of (58) automatically evaluates to

> (58)

$$[L_q, V]_- G = -i \epsilon_{m, n, q} \hbar X_m \partial_n(V) G \quad (60)$$

So that using (53) $\equiv \partial_n(V) = -V^3 X_n$ and multiplying by $G(X)^{-1}$,

> *SubstituteTensor*((53), (60)) · $G(X)^{-1}$

$$[L_q, V]_- = i \epsilon_{m, n, q} \hbar X_m V^3 X_n \quad (61)$$

from where we get the first commutation rule:

> *Simplify*((61))

$$[L_q, V]_- = 0 \quad (62)$$

Likewise, from the *inert = active* form of $[p_q, V(X)]_-$

> (%Commutator = Commutator)($p[q], V(X)$)

$$[p_q, V]_- = [p_q, V]_- \quad (63)$$

by applying this equation to the test function $G(X)$ we get

> (*lhs = Library:-ApplyProductsOfDifferentialOperators@rhs*)((63) · $G(X)$)

$$[p_q, V]_- G = -i \hbar \partial_q(V) G \quad (64)$$

> *SubstituteTensor*((53), (64)) · $G(X)^{-1}$

$$[p_q, V]_- = i \hbar V^3 X_q \quad (65)$$

In the same way, for $[p_q, V^3]_-$ we get

> (%Commutator = Commutator)($p[q], V(X)^3$)

$$[p_q, V^3]_- = [p_q, V^3]_- \quad (66)$$

> (*lhs = Library:-ApplyProductsOfDifferentialOperators@rhs*)((66) · $G(X)$)

$$[p_q, V^3]_- G = -i \hbar \left(\partial_q(V) V^2 + V \partial_q(V) V + V^2 \partial_q(V) \right) G \quad (67)$$

> *SubstituteTensor*((53), (67)) · $G(X)^{-1}$

$$[p_q, V^3]_- = i \hbar (V^3 X_q V^2 + V^4 X_q V + V^5 X_q) \quad (68)$$

> (lhs = Simplify@rhs) ((68))

$$[p_q, V^3]_- = 3 i \hbar V^5 X_q \quad (69)$$

Adding now these new commutation rules to the setup of the problem, they will be taken into account in subsequent uses of *Simplify*

> (62), (65), (69)

$$[L_q, V]_- = 0, [p_q, V]_- = i \hbar V^3 X_q, [p_q, V^3]_- = 3 i \hbar V^5 X_q \quad (70)$$

> Setup((70))

$$\left[\text{algebrarules} = \left\{ [L_j, L_k]_- = i \hbar \epsilon_{j, k, n} L_n, [L_q, V]_- = 0, [p_j, L_k]_- = i \hbar \epsilon_{j, k, n} p_n, [p_k, p_l]_- = 0, \right. \right. \quad (71)$$

$$\left. \left. [p_q, V]_- = i \hbar V^3 X_q, [p_q, V^3]_- = 3 i \hbar V^5 X_q, [X_j, L_k]_- = i \hbar \epsilon_{j, k, n} X_n, [X_k, p_l]_- = i \hbar g_{k, l}, [X_k, V]_- = 0 \right\} \right]$$

Undo *differentialoperators* to work using two different approaches, with and without them.

> Setup(differentialoperators = none)

$$[\text{differentialoperators} = \text{none}] \quad (72)$$

Commutation rules between the Hamiltonian and each of the angular momentum and Runge-Lenz tensors

Departing from the Hamiltonian (50) $\equiv H = \frac{p_l^2}{2 m_e} - \kappa V$ and the definition of angular momentum (55) $\equiv L_q = \epsilon_{m, n, q} X_m p_n$, by taking their commutator we get

> Commutator((50), (55))

$$[H, L_q]_- = \frac{-i \epsilon_{m, n, q} \hbar (-X_m V^3 X_n \kappa m_e + p_l p_n g_{l, m})}{m_e} \quad (73)$$

> Simplify((73))

$$[H, L_q]_- = 0 \quad (74)$$

That is one of the two commutators we wanted to derive. For the commutator between the Hamiltonian and Runge-Lenz tensor, start from its definition

$$Z[k] = \frac{1}{2 m_e} \cdot \text{LeviCivita}[a, b, k] \cdot (L[a] \cdot p[b] - p[a] \cdot L[b]) + \kappa \cdot V(X) \cdot X[k]$$

$$Z_k = \frac{\epsilon_{a,b,k} (L_a p_b - p_a L_b)}{2 m_e} + \kappa V X_k \quad (75)$$

This tensor is Hermitian

> (75) – Dagger((75))

$$Z_k - Z_k^\dagger = \frac{2 \kappa V X_k m_e - 2 \kappa X_k V m_e + \epsilon_{a,b,k} (L_a p_b + L_b p_a - p_a L_b - p_b L_a)}{2 m_e} \quad (76)$$

> Simplify((76))

$$Z_k - Z_k^\dagger = 0 \quad (77)$$

Since the system knows about the commutation rule between linear and angular momentum,

> (%Commutator = Commutator)(L[a], p[b])

$$[L_a, p_b]_- = i \hbar \epsilon_{a,b,n} p_n \quad (78)$$

the expression (75) for Z_k can be simplified

> Simplify((75))

$$Z_k = \frac{i \hbar p_k}{m_e} + \kappa V X_k - \frac{\epsilon_{a,b,k} p_a L_b}{m_e} \quad (79)$$

and the angular momentum removed from the the right-hand side using (55)

$\equiv L_q = \epsilon_{m,n,q} X_m p_n$, so that Z_k gets expressed entirely in terms of p_k , X and V

> Simplify(SubstituteTensor((55), (79)))

$$Z_k = \frac{-i \hbar p_k + \kappa V X_k m_e - X_k p_m^2 + X_m p_k p_m}{m_e} \quad (80)$$

Taking the commutator between (50) $\equiv H = \frac{p_l^2}{2 m_e} - \kappa V$, and this expression for Z_k we

have the starting point towards showing that $[H, Z_k]_- = 0$

> $\frac{2 m_e}{\kappa \hbar}$ Simplify(Commutator((50), (80)))

$$\frac{2 m_e [H, Z_k]_-}{\kappa \hbar} = \hbar V^3 X_k + \hbar V^5 X_a^2 X_k - 2 i p_k V + 2 i V X_a X_k p_a V^2 - 2 i X_a X_k p_a V^3 + 2 i X_a^2 p_k V^3 \quad (81)$$

In order to use the identities

> (54), $V(X)^2$. (54)

$$V^3 X_l^2 = V, V^5 X_l^2 = V^3 \quad (82)$$

we sort the products using the ordering shown in the left-hand sides

> *SortProducts*((81), $[V(X)^5, V(X)^3, X[a]^2]$)

$$\frac{2 m_e [H, Z_k]_-}{\kappa \hbar} = \hbar V^3 X_k - 5 \hbar V^5 X_a^2 X_k - 2 i p_k V + 2 i V X_a X_k p_a V^2 - 2 i X_a X_k p_a V^3 + 2 i V^3 X_a^2 p_k \quad (83)$$

> *SubstituteTensor*((82), (83))

$$\frac{2 m_e [H, Z_k]_-}{\kappa \hbar} = -4 \hbar V^3 X_k - 2 i p_k V + 2 i V X_a X_k p_a V^2 - 2 i X_a X_k p_a V^3 + 2 i V p_k \quad (84)$$

> *Simplify*((84))

$$\frac{2 m_e [H, Z_k]_-}{\kappa \hbar} = -2 \hbar (V^3 X_k - V^5 X_a^2 X_k) \quad (85)$$

> $\frac{\kappa \hbar}{2 m_e}$ *SubstituteTensor*((82), (85))

$$[H, Z_k]_- = 0 \quad (86)$$

And this is the second commutator we wanted to derive.

Functional differentiation and differentiation of spinors

• In Maple 2021 you can compute derivatives, and functional derivatives of spinor fields and with respect to them.

> *restart*; *with*(*Physics*) :

For illustration purposes set an anticommutative prefix and use lowercase Latin letters to represent spinor indices

> *Setup*(*anticommutativeprefix* = psi, *spinorindices* = lowercase)

$$[\text{anticommutativeprefix} = \{\psi\}, \text{spinorindices} = \text{lowercase latin}] \quad (87)$$

Load the library of physics types

> *with*(*Library*:-*PhysicsType*) :

Any anticommutative variable with spinor indices is now recognized by the system as a Dirac spinor

$$\begin{aligned} > \text{type}(\Psi_a, \text{DiracSpinor}) \\ & \qquad \qquad \qquad \text{true} \end{aligned} \qquad (88)$$

The conjugate, or Dagger too

$$\begin{aligned} > \text{type}(\overline{\Psi}_a, \text{DiracSpinor}) \\ & \qquad \qquad \qquad \text{true} \end{aligned} \qquad (89)$$

$$\begin{aligned} > \text{type}(\Psi_a^\dagger, \text{DiracSpinor}) \\ & \qquad \qquad \qquad \text{true} \end{aligned} \qquad (90)$$

To simplify expressions involving Ψ_a using Einstein's sum rule for repeated indices, indicate to the system that Ψ_a is a *tensor*

$$\begin{aligned} > \text{Define}(\Psi_a) \\ & \qquad \qquad \qquad \text{Defined objects with tensor properties} \\ & \qquad \qquad \qquad \{ \gamma_\mu, \sigma_\mu, \partial_\mu, g_{\mu\nu}, \Psi_a, \epsilon_{\alpha\beta\mu\nu} \} \end{aligned} \qquad (91)$$

You can now compute derivatives of expressions involving spinors with respect spinors according to

$$\begin{aligned} > (\%diff=diff) (\text{psi}[a], \text{psi}[b]) \\ & \qquad \qquad \qquad \frac{\partial}{\partial \Psi_b} \Psi_a = \delta_{a,b} \end{aligned} \qquad (92)$$

$$\begin{aligned} > (\%diff=diff) (\text{conjugate}(\text{psi}[a]), \text{conjugate}(\text{psi}[b])) \\ & \qquad \qquad \qquad \frac{d}{d\Psi_b} \overline{\Psi}_a = \delta_{a,b} \end{aligned} \qquad (93)$$

$$\begin{aligned} > (\%diff=diff) (\text{Dagger}(\text{psi}[a]), \text{Dagger}(\text{psi}[b])) \\ & \qquad \qquad \qquad \frac{d}{d\Psi_b^\dagger} \Psi_a^\dagger = \delta_{a,b} \end{aligned} \qquad (94)$$

Conjugate or Dagger functions are considered *independent* from the functions they are applied on

$$\begin{aligned} > (\%diff=diff) (\text{Dagger}(\text{psi})[a], \text{psi}[b]) \\ & \qquad \qquad \qquad \frac{d}{d\Psi_b} \Psi_a^\dagger = 0 \end{aligned} \qquad (95)$$

- In addition to standard derivatives, you can now compute functional derivatives with

respect to spinor functions. Consider, for example, an interaction Lagrangian where $\psi_{\mu, a}$ is of anticommutative type, but now a 3/2 spinor field (with one spacetime and one spinor indices), and in the Lagrangian there is a quantum vector field A_μ

> Setup(coordinates = [X, Y], op = A, tensors = { $\psi[\mu, a]$, $A[\mu]$ })

* Partial match of 'op' against keyword 'quantumoperators'

Systems of spacetime coordinates are: $\{X = (x1, x2, x3, x4), Y = (y1, y2, y3, y4)\}$

$$\left[\text{coordinatesystems} = \{X, Y\}, \text{quantumoperators} = \{A\}, \text{tensors} = \{A_\mu, \gamma_\mu, \sigma_\mu, \partial_\mu, g_{\mu, \nu}, \psi_{\mu, a}, \overline{\psi}_{\mu, a}, \epsilon_{\alpha, \beta, \mu, \nu}, X_\mu, Y_\mu \} \right] \quad (96)$$

The Lagrangian is

$$\begin{aligned} > L := \alpha \left(\gamma^\mu \right)_{a, i} \left(\gamma^\nu \right)_{i, j} \left(\gamma^\rho \right)_{j, b} \overline{\psi}_{\mu, a}(X) \psi_{\rho, b}(X) A_\nu(X) \\ L := \alpha \left(\gamma^\mu \right)_{a, i} \left(\gamma^\nu \right)_{i, j} \left(\gamma^\rho \right)_{j, b} \overline{\psi}_{\mu, a}(X) \psi_{\rho, b}(X) A_\nu(X) \end{aligned} \quad (97)$$

In this Lagrangian, α is a coupling constant, γ^μ are the Dirac matrices and $\overline{\psi}_{\mu, a}$ represents the Dirac conjugate of the $\psi_{\mu, a}$ spinor field. The vertex factor in the Feynman rules for this Lagrangian can now be computed using functional differentiation using the ordering the fields appear in the noncommutative product in (97), that is

$$\begin{aligned} > \%Fundiff(L, \overline{\psi}_{\alpha, c}(Y), \psi_{\beta, k}(Y), A_\sigma(Y)) = Fundiff(L, \overline{\psi}_{\alpha, c}(Y), \psi_{\beta, k}(Y), A_\sigma(Y)) \\ \left(\frac{\delta}{\delta A_\sigma(Y)} \right) \left(\frac{\delta}{\delta \psi_{\beta, k}(Y)} \right) \left(\frac{\delta}{\delta \overline{\psi}_{\alpha, c}(Y)} \right) \left(\alpha \left(\gamma^\mu \right)_{a, i} \left(\gamma^\nu \right)_{i, j} \left(\gamma^\rho \right)_{j, b} \right. \\ \left. \overline{\psi}_{\mu, a}(X) \psi_{\rho, b}(X) A_\nu(X) \right) = \alpha \left(\gamma^\mu \right)_{a, i} \left(\gamma^\nu \right)_{i, j} \left(\gamma^\rho \right)_{j, b} \delta^{(4)}([x1 - y1, x2 - y2, x3 \\ - y3, x4 - y4])^3 \delta_\mu^\alpha \delta_{a, c} \delta_\rho^\beta \delta_{b, k} \delta_\nu^\sigma \end{aligned} \quad (98)$$

Check the indices of both sides of this result

> Check(((98)), all)

The products in the given expression check ok.

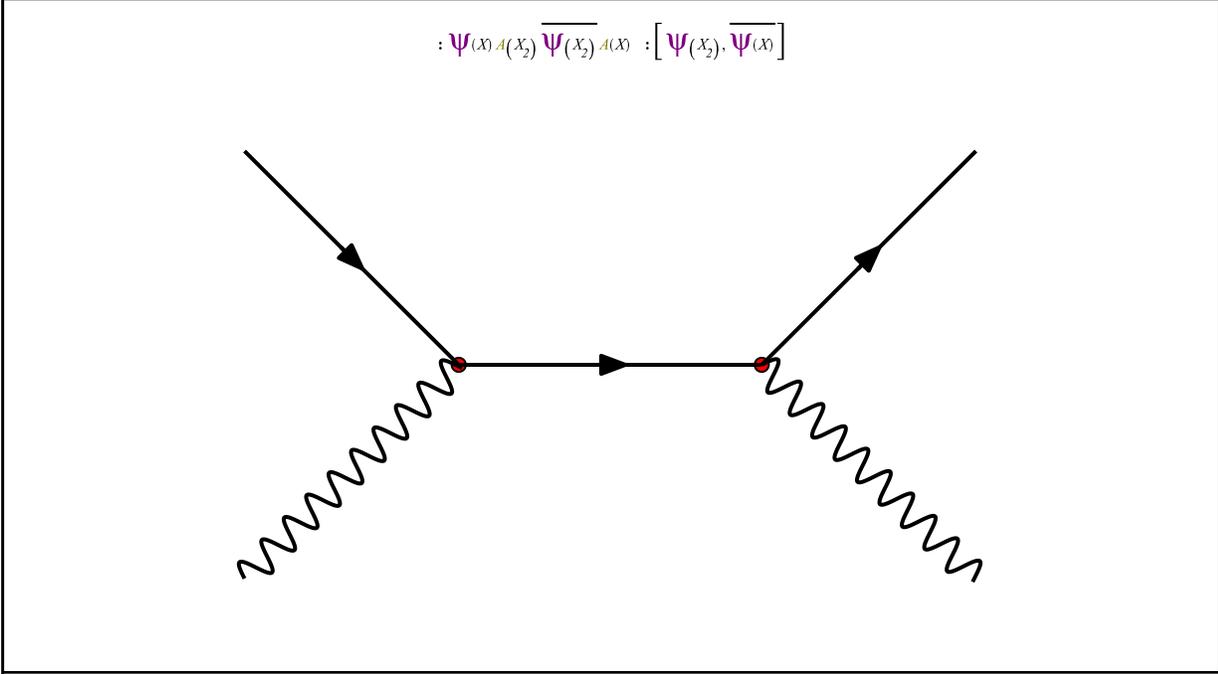
The repeated indices per term are: $[\{\dots\}, \{\dots\}, \dots]$, the free indices are: $\{\dots\}$

$$\left([\{a, b, i, j, \mu, \nu, \rho\}], \{c, k, \sim\text{alpha}, \sim\text{beta}, \sim\text{sigma}\} \right) = \left([\{a, b, i, j, \mu, \nu, \rho\}], \{c, k, \sim\text{alpha}, \sim\text{beta}, \sim\text{sigma}\} \right) \quad (99)$$

In the following example, the scattering amplitude of a process with two incoming and outgoing particles of the ψ and A types at tree-level, the vertex factors are computed

using functional differentiation as shown above in (98)

> *FeynmanDiagrams*(L , incomingparticles = $[\Psi, A]$, outgoing = $[\Psi, A]$, numberofloops = 0, diagrams)



$$\left(I(u_\Psi)_{\alpha, c}(\vec{P}_1) (\epsilon_A)_\beta(\vec{P}_2) \overline{(u_\Psi)_{\kappa, d}(\vec{P}_3)} \right. \quad (100)$$

$$\begin{aligned} & \overline{(\epsilon_A)_\lambda(\vec{P}_4)} \alpha^2 (\gamma^\sigma)_{l, e} (\gamma^\kappa)_{d, k} (\gamma^\lambda)_{k, l} (\gamma^\alpha)_{n, c} (\gamma^\tau)_{f, m} (\gamma^\beta)_{m, n} \left(\left(g_{\sigma, \tau} \right. \right. \\ & \left. \left. - \frac{(P_{1\sigma} + P_{2\sigma})(P_{1\tau} + P_{2\tau})}{m_\Psi^2} \right) \left((P_{1\nu} + P_{2\nu}) (\gamma^\nu)_{e, f} + m_\Psi \delta_{e, f} \right) \right. \\ & \left. + \left(\frac{(P_{1\sigma} + P_{2\sigma}) \delta_{e, g}}{3 m_\Psi} + \frac{(\gamma_\sigma)_{e, g}}{3} \right) \left((P_{1\nu} + P_{2\nu}) (\gamma^\nu)_{g, h} \right. \right. \\ & \left. \left. - m_\Psi \delta_{g, h} \right) \left(\frac{(P_{1\tau} + P_{2\tau}) \delta_{f, h}}{m_\Psi} + (\gamma_\tau)_{h, f} \right) \right) \delta(-P_3^\lambda - P_4^\lambda + P_1^\lambda + P_2^\lambda) \Big/ \\ & \left(8 \pi^2 \sqrt{E_2 E_4} \left((P_{1\omega} + P_{2\omega}) (P_1^\omega + P_2^\omega) - m_\Psi^2 + i\epsilon \right) \right) + \left(I(u_\Psi)_{\alpha, c} \left(\right. \right. \end{aligned}$$

$$\begin{aligned}
& \overrightarrow{P_1} (\boldsymbol{\epsilon}_A)_\beta (\overrightarrow{P_2}) (\mathbf{u}_\Psi)_{\kappa, d} (\overrightarrow{P_3}) \\
& \overline{(\boldsymbol{\epsilon}_A)_\lambda (\overrightarrow{P_4})} \alpha^2 (\gamma^\sigma)_{l, e} (\gamma^\kappa)_{d, k} (\gamma^\beta)_{k, l} (\gamma^\alpha)_{n, c} (\gamma^\tau)_{f, m} (\gamma^\lambda)_{m, n} \left(\left(g_{\sigma, \tau} \right. \right. \\
& \left. \left. - \frac{(P_{1\sigma} - P_{4\sigma})(P_{1\tau} - P_{4\tau})}{m_\Psi^2} \right) \left((P_{1\nu} - P_{4\nu}) (\gamma^\nu)_{e, f} + m_\Psi \delta_{e, f} \right) \right. \\
& \left. + \left(\frac{(P_{1\sigma} - P_{4\sigma}) \delta_{e, g}}{3 m_\Psi} + \frac{(\gamma_\sigma)_{e, g}}{3} \right) \left((P_{1\nu} - P_{4\nu}) (\gamma^\nu)_{g, h} \right. \right. \\
& \left. \left. - m_\Psi \delta_{g, h} \right) \left(\frac{(P_{1\tau} - P_{4\tau}) \delta_{f, h}}{m_\Psi} + (\gamma_\tau)_{h, f} \right) \right) \delta \left(-P_3^\chi - P_4^\chi + P_1^\chi + P_2^\chi \right) \Bigg/ \\
& \left(8 \pi^2 \sqrt{E_2 E_4} \left((P_{1\omega} - P_{4\omega}) (P_1^\omega - P_4^\omega) - m_\Psi^2 + I \epsilon \right) \right)
\end{aligned}$$

- It is possible to functionally differentiate a product of noncommutative functions using a different ordering than the one of the product provided that the commutation rules between the functions are known. For example, for the Lagrangian L , you can set the commutation rules for A_μ and $\Psi_{\alpha, j}$ as follows, without indicating the functionality, so that the rules are valid in general

$$\begin{aligned}
& \text{Setup}(\%Commutator(A_\mu, A_\nu) = 0, \%Commutator(\Psi[\mu, a], A[\nu]) = 0) \\
& \quad \left[\text{algebrarules} = \left\{ [A_\mu, A_\nu]_- = 0, [\Psi_{\mu, a}, A_\nu]_- = 0 \right\} \right] \tag{101}
\end{aligned}$$

Then you have, for any X or Y ,

$$\begin{aligned}
& \text{(%Commutator = Commutator)}(A[\mu](X), \text{psi}[\nu, a](Y)) \\
& \quad [A_\mu(X), \Psi_{\nu, a}(Y)]_- = 0 \tag{102}
\end{aligned}$$

Avoid repeated display of functionality with

$$\begin{aligned}
& \text{CompactDisplay}((A, \Psi)(X)) \\
& \quad A(X) \text{ will now be displayed as } A \\
& \quad \Psi(X) \text{ will now be displayed as } \Psi \tag{103}
\end{aligned}$$

The functional derivatives of L with regards to each of the fields

$$\begin{aligned}
& \%Fundiff(\%L, A_\alpha(Y)) = Fundiff(L, A_\alpha(Y)) \tag{104}
\end{aligned}$$

$$\left(\frac{\delta}{\delta A_{\alpha}(Y)} \right) L = \alpha \left(\gamma^{\mu} \right)_{a,i} \left(\gamma^{\nu} \right)_{i,j} \left(\gamma^{\rho} \right)_{j,b} \delta^{(4)}([x1 - y1, x2 - y2, x3 - y3, x4 - y4]) \delta_{\nu}^{\alpha} \overline{\Psi_{\mu,a} \Psi_{\rho,b}} \quad (104)$$

> %Fundiff(%L, $\Psi_{\alpha,j}(Y)$) = Fundiff(L, $\Psi_{\alpha,j}(Y)$)

$$\left(\frac{\delta}{\delta \Psi_{\alpha,j}(Y)} \right) L = -\alpha \left(\gamma^{\mu} \right)_{a,i} \left(\gamma^{\nu} \right)_{i,c} \left(\gamma^{\rho} \right)_{c,b} \delta^{(4)}([x1 - y1, x2 - y2, x3 - y3, x4 - y4]) \delta_{\rho}^{\alpha} \delta_{b,j} \overline{\Psi_{\mu,a} A_{\nu}} \quad (105)$$

> %Fundiff(%L, $\overline{\Psi_{\alpha,j}(Y)}$) = Fundiff(L, $\overline{\Psi_{\alpha,j}(Y)}$)

$$\left(\frac{\delta}{\delta \overline{\Psi_{\alpha,j}(Y)}} \right) L = \alpha \left(\gamma^{\mu} \right)_{a,i} \left(\gamma^{\nu} \right)_{i,c} \left(\gamma^{\rho} \right)_{c,b} \delta^{(4)}([x1 - y1, x2 - y2, x3 - y3, x4 - y4]) \delta_{\mu}^{\alpha} \delta_{a,j} \overline{\Psi_{\rho,b} A_{\nu}} \quad (106)$$

>

New commands ToCovariant and ToContravariant

When working with tensors in spaces where the covariant and contravariant tensor's components have a different value (the underlying metric is not Euclidean) one frequently wants to express formulations with some or all of the tensors's indices expressed either in covariant or contravariant form. In previous Maple releases, also in Maple 2021, you can raise or lower free indices multiplying by the metric and performing the contraction. That, however, involves a whole simplification process not always desired, and does not result in flipping the character of repeated indices. To handle the whole manipulation operation, in Maple 2021 there are two new commands: [ToCovariant](#) and [ToContravariant](#).

Examples

Consider the following tensorial expression

> restart; with(Physics) :

> Setup(coordinates = cartesian, tensors = A[μ])

Systems of spacetime coordinates are: $\{X = (x, y, z, t)\}$

$$\left[\text{coordinatesystems} = \{X\}, \text{tensors} = \left\{ A_{\mu}, \gamma_{\mu}, \sigma_{\mu}, \partial_{\mu}, g_{\mu, \nu}, \epsilon_{\alpha, \beta, \mu, \nu}, X_{\mu} \right\} \right] \quad (107)$$

> CompactDisplay(A(X))

(108)

$A(X)$ will now be displayed as A (108)

> $F[\mu, \nu] = d_{[\mu]}(A[\nu](X)) - d_{[\nu]}(A[\mu](X))$
 $F_{\mu, \nu} = \partial_{\mu}(A_{\nu}) - \partial_{\nu}(A_{\mu})$ (109)

> Define((109))

Defined objects with tensor properties

$\{A_{\mu}, \gamma_{\mu}, F_{\mu, \nu}, \sigma_{\mu}, \partial_{\mu}, g_{\mu, \nu}, \epsilon_{\alpha, \beta, \mu, \nu}, X_{\mu}\}$ (110)

The components of $F_{\mu, \nu}$ are

> $F[]$

$$F_{\mu, \nu} = \begin{bmatrix} 0 & (A_2)_x - (A_1)_y & (A_3)_x - (A_1)_z & (A_4)_x - (A_1)_t \\ (A_1)_y - (A_2)_x & 0 & (A_3)_y - (A_2)_z & (A_4)_y - (A_2)_t \\ (A_1)_z - (A_3)_x & (A_2)_z - (A_3)_y & 0 & (A_4)_z - (A_3)_t \\ (A_1)_t - (A_4)_x & (A_2)_t - (A_4)_y & (A_3)_t - (A_4)_z & 0 \end{bmatrix} \quad (111)$$

New in Maple 2021: when you request the *contravariant* components of a tensor definition as $F^{\mu, \nu}$, the output is expressed using the *contravariant* components of the tensors involved, in this case A^{μ}

> $F[\sim]$

$F^{\mu, \nu}$ (112)

$$= \begin{bmatrix} 0 & -(A^2)_x + (A^1)_y & -(A^3)_x + (A^1)_z & -(A^4)_x - (A^1)_t \\ -(A^1)_y + (A^2)_x & 0 & -(A^3)_y + (A^2)_z & -(A^4)_y - (A^2)_t \\ -(A^1)_z + (A^3)_x & -(A^2)_z + (A^3)_y & 0 & -(A^4)_z - (A^3)_t \\ (A^1)_t + (A^4)_x & (A^2)_t + (A^4)_y & (A^3)_t + (A^4)_z & 0 \end{bmatrix}$$

Consider now an expression that has free and repeated indices, for example

> Define(A, B, G)

Defined objects with tensor properties

$\{B, G, A_{\mu}, \gamma_{\mu}, F_{\mu, \nu}, \sigma_{\mu}, \partial_{\mu}, g_{\mu, \nu}, \epsilon_{\alpha, \beta, \mu, \nu}, X_{\mu}\}$ (113)

> $B[\beta] G[\nu, \alpha] A[\alpha] F[\mu, \nu] + G[\rho, \alpha] B[\alpha] F[\mu, \rho] A[\beta]$

.....

$$A_{\beta} G_{\rho, \alpha} F_{\mu}^{\rho} B^{\alpha} + B_{\beta} G_{\nu, \alpha} A^{\alpha} F_{\mu}^{\nu} \quad (114)$$

There are several tensors with covariant indices, and the free and repeated indices are

> Check((114), all)

The repeated indices per term are: [{...}, {...}, ...], the free indices are: { ... }

$$[\{ \alpha, \rho \}, \{ \alpha, \nu \}, \{ \beta, \mu \}] \quad (115)$$

where we see the free indices are μ and β , both covariant. To have *all* the tensors in (114) (but for the metric) with *all* their indices contravariant, use

> ToContravariant((114))

$$g_{\beta, \kappa} B^{\kappa} g_{\nu, \sigma} g_{\alpha, \tau} G^{\sigma, \tau} A^{\alpha} g_{\lambda, \mu} F^{\lambda, \nu} + g_{\beta, \nu} A^{\nu} g_{\lambda, \rho} g_{\alpha, \sigma} G^{\lambda, \sigma} g_{\kappa, \mu} F^{\kappa, \rho} B^{\alpha} \quad (116)$$

> indets((116), specindex([G, A, B, F]))

$$\{ A^{\alpha}, A^{\nu}, B^{\alpha}, B^{\kappa}, F^{\kappa, \rho}, F^{\lambda, \nu}, G^{\lambda, \sigma}, G^{\sigma, \tau} \} \quad (117)$$

Note this result is mathematically equal to (114) - all what happened in (116) is that the covariant versions were replaced by the contravariant ones at the cost of adding metric factors, so

> Simplify((114) - (116))

$$0 \quad (118)$$

Likewise,

> ToCovariant((114))

$$A_{\beta} G_{\rho, \alpha} g^{\kappa, \rho} F_{\mu, \kappa} g^{\alpha, \nu} B_{\nu} + B_{\beta} G_{\nu, \alpha} g^{\alpha, \kappa} A_{\kappa} g^{\lambda, \nu} F_{\mu, \lambda} \quad (119)$$

> indets((119), specindex([G, A, B, F]))

$$\{ A_{\beta}, A_{\kappa}, B_{\nu}, B_{\nu}, F_{\kappa, \mu}, F_{\lambda, \mu}, G_{\nu, \alpha}, G_{\rho, \alpha} \} \quad (120)$$

This type of manipulation is relevant in different contexts. For example, when computing the [TensorArray](#) of components of a tensorial expression, that result in expressed using the contravariant (or covariant) components of the tensors involved.

You can selectively apply these two commands command using its options *onlytheseindices*, for example to have only the index μ expressed in contravariant form use

> ToContravariant((114), onlytheseindices = [μ])

$$A_{\beta} G_{\rho, \alpha} g_{\mu, \nu} F^{\nu, \rho} B^{\alpha} + B_{\beta} G_{\nu, \alpha} A^{\alpha} g_{\kappa, \mu} F^{\kappa, \nu} \quad (121)$$

A similar operation, which however results in an expression that is not mathematically

equal to the departing one is that where we want to *flip* the character of the free indices, optionally also of the repeated indices. For example, in (114) and (116) μ is covariant, but suppose we want to raise it turning it contravariant

> *ToContravariant*((114), *changecharacter*)

* *Partial match of 'changecharacter' against keyword 'changecharacteroffreeindices'*

$$B^\beta G_{\nu, \alpha} A^\alpha F^{\mu, \nu} + A^\beta G_{\rho, \alpha} F^{\mu, \rho} B^\alpha \quad (122)$$

So now the free indices μ and β are both contravariant

> *Check*((122), *free*)

The free indices are: { ... }

{~beta, ~mu} (123)

$$G_{\nu, \alpha} g^{\alpha, \beta} A_\beta F_\mu^\nu + G_{\rho, \alpha} F_\mu^\rho g^{\alpha, \nu} B_\nu \quad (124)$$

Or, selectively,

> *ToContravariant*((114), *changecharacter*, *onlytheseindices* = [μ])

* *Partial match of 'changecharacter' against keyword 'changecharacteroffreeindices'*

$$A_\beta G_{\rho, \alpha} F^{\mu, \rho} B^\alpha + B_\beta G_{\nu, \alpha} A^\alpha F^{\mu, \nu} \quad (125)$$

> *Check*((125), *free*)

The free indices are: { ... }

{ β , ~mu} (126)

Compare (125) with the different forms of the same expression

> *ToContravariant*((114), *onlytheseindices* = [μ])

$$A_\beta G_{\rho, \alpha} g_{\mu, \nu} F^{\nu, \rho} B^\alpha + B_\beta G_{\nu, \alpha} A^\alpha g_{\kappa, \mu} F^{\kappa, \nu} \quad (127)$$

> *ToContravariant*((114), *changecharacter*, *contractwiththemetric*)

* *Partial match of 'changecharacter' against keyword 'changecharacteroffreeindices'*

$$g^{\beta, \nu} g^{\kappa, \mu} A_\nu G_{\rho, \alpha} F_\kappa^\rho B^\alpha + g^{\beta, \kappa} g^{\lambda, \mu} B_\kappa G_{\nu, \alpha} A^\alpha F_\lambda^\nu \quad (128)$$

Weyl scalars, Petrov types and canonical forms for tetrads

In spite of recent advances in General Relativity, the computation of Petrov types, Weyl scalars and canonical forms for tetrads (mappings between a general spacetime and a local Galilean system of references) continue to be key in the classification of solutions to

Einstein's equations and for establishing equivalences between two given spacetimes. For these purposes, Maple's Physics includes a subpackage, [Tetrads](#), with all the relevant related functionality.

In previous Maple releases, and also in Maple 2021, once the spacetime metric $g_{\mu, \nu}$ and the tetrad $e_{\mu, a}$ are set (using [Setup](#)), the entering `Petrov()` and `Weyl[scalars]` returns the respective results *for that metric and tetrad set*. Also, a change in the signature, or using a textbook reference that shows a tetrad derived for a different signature frequently resulted in puzzling situations, where what is expected to be a tetrad does not verify the tetrad's defining equations.

Both situations are addressed in Maple 2021 so that problems with the signature are easily detected and easily corrected, and experiment with different forms of the tetrads and Weyl scalars can now *all be performed without setting the tetrad*.

For these purposes, in Maple 2021:

- We added a new command, `WeylScalars`, to the `Tetrads` package;
- extended [PetrovType](#) and [TransformTetrad](#) to handle different forms of the tetrads *even if they are not set*;
- extended the [IsTetrad](#) command to produce helpful and directly usable messages in the typical situation where we input what we think is a correct tetrad, but that is true only if we change the signature or the position of the time-like component in the list of coordinates;
- extended the [Redefine](#) command so that it also redefines tetrads according to indicated changes in the signature.

Examples

In a recent [question in Mapleprimes](#), one of the spacetime (metric) solution to Einstein's equations, from chapter 27 of the book of Exact Solutions to Einstein's equations [1] was discussed. One of the issues was about computing a tetrad for the book's solution [27, 37, 1] such that the corresponding *Weyl* scalars are in canonical form.

The Maple 2021 developments in this area allows for clearly reviewing the problem as follows.

> `restart; with(Physics) :`

The starting point is this image of page 421 of the book of Exact Solutions to Einstein's equations, formulas (27.37)

$$\begin{aligned}
ds^2 &= 2\omega^1\omega^2 - 2\omega^3\omega^4 \\
&= 2r^2P^{-2}d\zeta d\bar{\zeta} - 2du dr - 2Hdu^2, \quad P_{,r} = 0, \\
m^i &= (P/r, 0, 0, 0), \quad \bar{m}^i = (0, P/r, 0, 0), \\
l^i &= (0, 0, -H, 1), \quad k^i = (0, 0, 1, 0).
\end{aligned}
\tag{27.37}$$

Load the corresponding solution [27, 37, 1] from Maple's database of solutions to Einstein's equations

> g_[[27, 37, 1]]

Systems of spacetime coordinates are: {X= (z, zb, r, u)}

Default differentiation variables for d_, D_ and dAlembertian are: {X= (z, zb, r, u)}

The Robinson and Trautman (1962) metric in coordinates [z, zb, r, u]

Parameters: [P(z, zb, u), H(X)]

Comments: admits geodesic, shearfree, twistfree null congruence, rho=-1/r=rho_b

Resetting the signature of spacetime from (- - - +) to (+ + + -) in order to match the signature in the database of metrics

Setting lowercaselatin_is letters to represent space indices

$$g_{\mu, \nu} = \begin{bmatrix} 0 & \frac{r^2}{P(z, zb, u)^2} & 0 & 0 \\ \frac{r^2}{P(z, zb, u)^2} & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & -2 H(X) \end{bmatrix}
\tag{129}$$

> CompactDisplay((129))

H(X) will now be displayed as H

P(z, zb, u) will now be displayed as P (130)

The assumptions on the metric's parameters are

> Assume(P(z, zb, u) > 0, H(X) :: real, r ≥ 0) :

The line element is as shown in the second line of the book's image above

> g_[lineelement]

$$\frac{2 r^2 \mathbf{d}(z) \mathbf{d}(zb)}{P^2} - 2 \mathbf{d}(r) \mathbf{d}(u) - 2 H \mathbf{d}(u)^2
\tag{131}$$

Load [Tetrads](#)

> with(Tetrads) :

Setting lowercaselatin_ah letters to represent tetrad indices

Defined as tetrad tensors (see ?Physics,tetrads), $e_{a,\mu}$, $\eta_{a,b}$, $\gamma_{a,b,c}$, $\lambda_{a,b,c}$

Defined as spacetime tensors representing the NP null vectors of the tetrad formalism (see ?

Physics,tetrads), l_μ , n_μ , m_μ , \bar{m}_μ

(132)

The Petrov type of this spacetime solution is

> PetrovType()

"II"

(133)

The null tetrad computed by the Maple system using a general algorithms is

> Setup(tetrad=null) :

> e_[]

$$e_{a,\mu} = \begin{bmatrix} \frac{-Ir}{P} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{(H^2+1)^{1/4}} & \frac{H}{(H^2+1)^{1/4}} \\ 0 & 0 & 0 & -(H^2+1)^{1/4} \\ 0 & \frac{-Ir}{P} & 0 & 0 \end{bmatrix}$$

(134)

According to the help page [TransformTetrad](#), the canonical form of the Weyl scalars for each different Petrov type is

	Ψ_0	Ψ_1	Ψ_2	Ψ_3	Ψ_4	Residual invariance
Petrov type I	0	$\neq 0$	$\neq 0$	1	0	none
Petrov type II	0	0	$\neq 0$	1	0	none
Petrov type III	0	0	0	1	0	none
Petrov type D	0	0	$\neq 0$	0	0	rotations of Class III
Petrov type N	0	0	0	0	1	rotations of Class II

So for type II, when the tetrad is in canonical form, we expect only Ψ_2 and Ψ_3 different from 0. For the tetrad computed automatically, however, the scalars are not in canonical form:

> Weyl[scalars]

$$\begin{aligned}
\Psi_0 &= -\frac{P \left((H_{z,z}) P + 2 (H_z) (P_z) \right)}{r^2 \sqrt{H^2 + 1}}, \Psi_1 & (135) \\
&= \frac{-\frac{1}{2} \left((H_{r,z}) P^2 r - (P_{u,z}) r P - 2 P^2 (H_z) + (P_u) (P_z) r \right)}{P r^2 (H^2 + 1)^{1/4}}, \Psi_2 \\
&= \frac{(H_{r,r}) r^2 - 2 (H_r) r - 2 (P_{z,zb}) P + 2 (P_z) (P_{zb}) + 2 H}{6 r^2}, \Psi_3 = 0, \Psi_4 = 0
\end{aligned}$$

The question is, thus, how to bring the tetrad $e_{a,\mu}$ (equation (134)) into *canonical form*.

The plan for that is outlined in Chapter 7, by Chandrasekhar, page 388, of the book "*General Relativity, an Einstein centenary survey*", edited by S.W. Hawking and W. Israel. In brief, for Petrov type II, use a transformation of use *Class*₂ to make $\Psi_0 = \Psi_1 = 0$, then a transformation of *Class*₁ making $\Psi_4 = 0$, finally use a transformation of *Class*₃ making $\Psi_3 = 1$. For an explanation of these transformations see the help page for [TransformTetrad](#). This plan, however, is applicable if and only if the starting tetrad results in $\psi_4 \neq 0$, which we see in (135) it is not the case, so we need, in addition, before applying this plan, to perform a transformation of *Class*₁ making $\psi_4 \neq 0$.

In what follows, first we use *TransformTetrad* with the optional argument *canonicalform* to perform all the calculation in one step. This problem was out of reach in previous Maple releases and *is now tractable in different ways* in Maple 2021.

Next, we use the prescription, step by step, as outlined in Chapter 7, by Chandrasekhar, page 388, of the book "*General Relativity, an Einstein centenary survey*", edited by S.W. Hawking and W. Israel. The possibility of performing a step by step computation as shown below is also new in Maple 2021, and illustrates well how to get the result exploiting the new, advanced functionality.

All the transformations performed automatically, in one go

To arrive in one go to a tetrad whose *Weyl* scalars are in canonical form as shown in the table above, use the optional argument *canonicalform*:

> $T_5 := \text{TransformTetrad}(\text{canonicalform}) :$

The resulting tetrad got assigned to T_5 and is not displayed because of its length:

> $\text{length}(T_5)$

New in Maple 2021, the command *WeylScalars* can compute the Weyl scalars without having to set the tetrad (as in T_5) first. That is convenient to see if the result obtained is indeed what we want, even when at this point the tetrad is different from T_5 :

> $e_{[\]}$

$$e_{a, \mu} = \begin{bmatrix} \frac{-Ir}{P} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{(H^2 + 1)^{1/4}} & \frac{H}{(H^2 + 1)^{1/4}} \\ 0 & 0 & 0 & -(H^2 + 1)^{1/4} \\ 0 & \frac{-Ir}{P} & 0 & 0 \end{bmatrix} \quad (137)$$

So, instead of having to set the tetrad, this new *WeylScalars* directly receives T_5 as an argument and performs the calculation

> *WeylScalars*(T_5)

$$\Psi_0 = 0, \Psi_1 = 0, \Psi_2 = \frac{(H_{r,r}) r^2 - 2 (H_r) r - 2 (P_{z,zb}) P + 2 (P_z) (P_{zb}) + 2 H}{6 r^2}, \Psi_3 = 1, \Psi_4 = 0 \quad (138)$$

We see, comparing with the table before (135), that this is the expected form of these scalars, so T_5 is indeed a tetrad in canonical form.

The transformations, step-by-step, leading to the same canonical form (138) of the Weyl scalars

In brief, the plan outlined by Chandrasekhar in Chap.7 is: for Petrov type II, use a transformation of *Class*₂ to make $\Psi_0 = \Psi_1 = 0$, then a transformation of *Class*₁ making $\Psi_4 = 0$, finally use a transformation of *Class*₃ making $\Psi_3 = 1$. For an explanation of these transformations see the help page for [TransformTetrad](#). The plan is applicable if and only if the starting tetrad results in $\Psi_4 \neq 0$, which we see in (138) is not the case, so we need, in addition, before following this plan, to perform a transformation of *Class*₁ making $\Psi_4 \neq 0$.

It is possible to start from the tetrad (134) and the Weyl scalars (135), which however have radicals, against readability. Without loss of generality, we use instead, as departing tetrad, a version of it free of radicals as shown in the book of solutions to Einstein's equations, that is

$$\begin{aligned}
> e_{-}[a, \mu] = & \left(\begin{array}{cccc} 0 & 0 & 0 & -1 \\ 0 & \frac{r}{P(z, zb, u)} & 0 & 0 \\ \frac{r}{P(z, zb, u)} & 0 & 0 & 0 \\ 0 & 0 & -1 & -H(X) \end{array} \right) \\
e_{a, \mu} = & \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & \frac{r}{P} & 0 & 0 \\ \frac{r}{P} & 0 & 0 & 0 \\ 0 & 0 & -1 & -H \end{bmatrix}
\end{aligned} \tag{139}$$

> *IsTetrad*((139))

Type of tetrad: null

true

(140)

You can use the new *WeylScalars* corresponding to this book's tetrad (139), and see they are free of radicals but not in canonical form, using

> *WeylScalars*((139))

$$\begin{aligned}
\Psi_0 = & \frac{P \left((H_{z,z}) P + 2 (H_z) (P_z) \right)}{r^2}, \Psi_1 \\
= & \frac{(H_{r,z}) P^2 r - (P_{u,z}) r P - 2 P^2 (H_z) + (P_u) (P_z) r}{2 r^2 P}, \Psi_2 \\
= & \frac{(H_{r,r}) r^2 - 2 (H_r) r - 2 (P_{z,zb}) P + 2 (P_z) (P_{zb}) + 2 H}{6 r^2}, \Psi_3 = 0, \Psi_4 = 0
\end{aligned} \tag{141}$$

Set this tetrad simpler than (137) as the starting point

> *Setup*((139))

$$\left[\text{tetrad} = \left\{ (1, 4) = -1, (2, 2) = \frac{r}{P}, (3, 1) = \frac{r}{P}, (4, 3) = -1, (4, 4) = -H \right\} \right] \tag{142}$$

Step 0

Since in (141) $\Psi_4 = 0$, in this step we use a transformation of *Class*₁, that introduces a complex parameter *E*. To get $\Psi_4 \neq 0$ any value of *E* suffices; we use $E = 1$

> *TransformTetrad*(*nullrotationwithfixedl_*)

$$\begin{bmatrix} \frac{Ir}{P} & 0 & -I\bar{E} & -I\bar{E}H \\ 0 & 0 & -I & -IH \\ \frac{IEr}{P} & \frac{I\bar{E}r}{P} & -I\bar{E}E & -I\bar{E}EH - I \\ 0 & \frac{Ir}{P} & -IE & -IEH \end{bmatrix} \quad (143)$$

> $T_0 := eval((143), E=1)$

$$T_0 := \begin{bmatrix} \frac{Ir}{P} & 0 & -I & -IH \\ 0 & 0 & -I & -IH \\ \frac{Ir}{P} & \frac{Ir}{P} & -I & -IH - I \\ 0 & \frac{Ir}{P} & -I & -IH \end{bmatrix} \quad (144)$$

The new *WeylScalars* allows to check the transition to canonical form step by step. For example, in this step, T_0 resulted in a tetrad, $\Psi_4 \neq 0$ as expected

> $WeylScalars(T_0)[-1]$

$$\begin{aligned} \Psi_4 = & \frac{1}{r^2 P} \left(2 (H_{r,z}) P^2 r + (H_{z,z}) P^3 + 2 (H_z) (P_z) P^2 + (H_{r,r}) P r^2 - 4 P^2 (H_z) \right. \\ & - 2 (P_{z,zb}) P^2 + 2 (P_z) (P_{zb}) P - 2 (P_{u,z}) r P - 2 (H_r) P r + 2 (P_u) (P_z) r \\ & \left. + 2 HP \right) \end{aligned} \quad (145)$$

Step 1

Next is a transformation of *Class₂* to make $\Psi_0 = 0$. For Petrov type II that also implies on $\Psi_1 = 0$. This transformation introduces a parameter B (see [TransformTetrad](#)) that, according to the plan outlined by Chandrasekhar, should be set as one of the two identical roots (out of four) of the *principal polynomial*. To see the *principal polynomial*, or, directly, its *roots* you can use the [PetrovType](#) command:

> $PetrovType(principalroots='R')$

$$\text{"II"} \quad (146)$$

The pair of identical roots is equal to -1

> $R[1..2]$

$$[-1, -1] \quad (147)$$

So the transformed tetrad T_1 , taken at B equal to this multiple root, is

> $T_1 := eval(TransformTetrad(T_0, nullrotationwithfixedn_), B=-1)$

$$T_1 := \begin{bmatrix} 0 & -\frac{r}{P} & 0 & 1 \\ 0 & 0 & 0 & -1 \\ -\frac{r}{P} & 0 & 1 & H \\ \frac{r}{P} & 0 & 0 & 0 \end{bmatrix} \quad (148)$$

Check the corresponding Weyl scalars: we now have $\Psi_0=0$ and $\Psi_1=0$

> $WeylScalars(T_1)[1..2]$

$$\Psi_0=0, \Psi_1=0 \quad (149)$$

Step 2

Next is a transformation of $Class_1$ that makes $\Psi_4=0$. This transformation introduces again a parameter E , that according to Chandrasekhar's plan can be taken equal to one of the roots of Weyl scalar Ψ_4 that corresponds to the transformed tetrad. So we need to proceed in three steps:

- a. transform the tetrad introducing a parameter E in the tetrad's components
- b. compute the Weyl scalars for that transformed tetrad
- c. take $\Psi_4 = 0$ and solve for E
- d. apply the resulting value of E to the transformed tetrad obtained in step a.

a. Transform the tetrad and for simplicity take E real

> $Assume(E :: real)$

$$\{E::real\} \quad (150)$$

> $T_2 := TransformTetrad(T_1, nullrotationwithfixedl_)$

(151)

$$T_2 := \begin{bmatrix} \frac{I r (E - 1)}{P} & 0 & I & I H \\ \frac{I r}{P} & 0 & 0 & 0 \\ \frac{I E r (E - 1)}{P} & \frac{-I r}{P} & I E & I (E H - E + 1) \\ \frac{I E r}{P} & 0 & 0 & -I \end{bmatrix} \quad (151)$$

b. Compute Ψ_4 for this tetrad

> *simplify(WeylScalars(T₂)[-1])*

$$\Psi_4 = \frac{1}{r^2 P} \left(r^2 P (E - 1)^2 (H_{r,r}) - 2 r P^2 (E - 1) (H_{r,z}) + (H_{z,z}) P^3 - 2 P^2 (E - 1)^2 (P_{z,zb}) + 2 r P (E - 1) (P_{u,z}) - 2 r P (E - 1)^2 (H_r) + 4 P^2 \left(E + \frac{P_z}{2} - 1 \right) (H_z) + 2 (E - 1) \left((P (E - 1) (P_{zb}) - (P_u) r) (P_z) + H P (E - 1) \right) \right) \quad (152)$$

c. Solve $\Psi_4 = 0$ discarding the case $E = 0$ which implies on no transformation

> *simplify(solve({rhs(152)} = 0, E ≠ 0}, {E}, explicit)[1])*

Warning, solve may be ignoring assumptions on the input variables.

$$\left\{ E = \left((H_{r,r}) P r^2 + (H_{r,z}) P^2 r - 2 (H_r) P r - (P_{u,z}) r P + (P_u) (P_z) r - 2 P^2 (H_z) - 2 (P_{z,zb}) P^2 + 2 (P_z) (P_{zb}) P + 2 H P + (-P^4 ((H_{r,r}) r^2 - 2 (H_r) r - 2 (P_{z,zb}) P + 2 (P_z) (P_{zb}) + 2 H) (H_{z,z}) + (H_{r,z})^2 P^4 r^2 + (-2 r^2 (P_u) - 2 r^2 P (P_u) (P_z) + 4 r (H_z) P^3) (P_{u,z}) + 4 (P_{z,zb}) (H_z) (P_z) P^4 + 4 (H_z)^2 P^4 + 4 P^2 (P_z) ((H_r) P r - (P_z) (P_{zb}) P - (P_u) r - H P) (H_z) + (P_u)^2 (P_z)^2 r^2)^{1/2} \right) / \left(P ((H_{r,r}) r^2 - 2 (H_r) r - 2 (P_{z,zb}) P + 2 (P_z) (P_{zb}) + 2 H) \right) \right\}$$

d. Apply this result to the tetrad (151). In doing so, do not display the result, just measure its length (corresponds to two+ pages)

> *T₃ := simplify(eval(T₂, (153)[1])) :*

> length(T_3)

$$12553 \quad (154)$$

Check the first two and the last scalars, we expect $\Psi_0 = \Psi_1 = \Psi_4 = 0$

> WeylScalars(T_3):

%[1..2], %[-1]

$$\Psi_0 = 0, \Psi_1 = 0, \Psi_4 = 0 \quad (155)$$

Step 3

Use a transformation of $Class_3$ making $\Psi_3 = 1$. Such a transformation (see [TransformTetrad](#))

changes $\Psi_3' = A e^{-1\Omega} \Psi_3$, where we need to take $A e^{-1\Omega} = \frac{1}{\Psi_3}$, and without loss of generality

we can take $\Omega = 0$. Check first the value of Ψ_3 in the last tetrad computed, for that purpose you can use again the new *WeylScalars* without having to set anything before

> WeylScalars(T_3)[4]

$$\Psi_3 \quad (156)$$

$$\begin{aligned} &= \frac{1}{2 P r^2} \left(- (H_{r,r}) (H_{z,z}) P^4 r^2 - 2 (H_{r,r}) (H_z) (P_z) P^3 r^2 + (H_{r,z})^2 P^4 r^2 \right. \\ &+ 2 (H_{z,z}) (H_r) P^4 r + 2 (P_{z,zb}) (H_{z,z}) P^5 - 2 (H_{z,z}) (P_{zb}) (P_z) P^4 \\ &+ 4 (H_r) (H_z) (P_z) P^3 r + 2 (H_{r,z}) (P_u) (P_z) P^2 r^2 + 4 (P_{z,zb}) (H_z) (P_z) P^4 \\ &- 4 (H_z) (P_{zb}) (P_z)^2 P^3 - 4 (H_{r,z}) (H_z) P^4 r - 2 (P_{u,z}) (H_{r,z}) P^3 r^2 - 2 (H_{z,z}) H P^4 \\ &+ (P_u)^2 (P_z)^2 r^2 - 4 (H_z) (P_u) (P_z) P^2 r - 2 (P_{u,z}) (P_u) (P_z) P r^2 \\ &\left. - 4 H (H_z) (P_z) P^3 + 4 (H_z)^2 P^4 + 4 (P_{u,z}) (H_z) P^3 r + (P_{u,z})^2 P^2 r^2 \right)^{1/2} \end{aligned}$$

So, the transformed tetrad T_4 to which corresponds *Weyl* scalars in canonical form, with

$\Psi_0 = \Psi_1 = \Psi_4 = 0$ and $\Psi_3 = 1$, is

> $T_4 := \text{simplify} \left(\text{eval} \left(\text{TransformTetrad} (T_3, \text{boostsn_l_plane}), A = \frac{1}{\text{rhs}(\text{(156)})} \right) \right)$:

Check this result

> IsTetrad(T_4)

Type of tetrad: null

true

$$(157)$$

Compute the scalars

> *WeylScalars*(T_4)

$$\Psi_0=0, \Psi_1=0, \Psi_2 = \frac{(H_{r,r}) r^2 - 2 (H_r) r - 2 (P_{z,zb}) P + 2 (P_z) (P_{zb}) + 2 H}{6 r^2}, \Psi_3=1, \Psi_4 = 0 \quad (158)$$

These scalars obtained step-by-step are the same scalars computed in one go in (138)

> (138)

$$\Psi_0=0, \Psi_1=0, \Psi_2 = \frac{(H_{r,r}) r^2 - 2 (H_r) r - 2 (P_{z,zb}) P + 2 (P_z) (P_{zb}) + 2 H}{6 r^2}, \Psi_3=1, \Psi_4 = 0 \quad (159)$$

Changing the signature and redefining tetrads

Finally, consider a change in the signature, from the current value

> *Setup*(signature)

$$[\text{signature} = + + + -] \quad (160)$$

to (+ - - -)

> *Setup*(signature = "+ - - -")

$$[\text{signature} = + - - -] \quad (161)$$

The tetrad got automatically redefined

> $e_{[}$

$$e_{a,\mu} = \begin{bmatrix} 0 & \frac{r}{P} & 0 & 0 \\ \frac{r}{P} & 0 & 0 & 0 \\ 0 & 0 & 0 & I(H^2 + 1)^{1/4} \\ 0 & 0 & \frac{-I}{(H^2 + 1)^{1/4}} & \frac{-IH}{(H^2 + 1)^{1/4}} \end{bmatrix} \quad (162)$$

It is not equal to (139) anymore

> (139)

(163)

$$e_{a,\mu} = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & \frac{r}{P} & 0 & 0 \\ \frac{r}{P} & 0 & 0 & 0 \\ 0 & 0 & -1 & -H \end{bmatrix} \quad (163)$$

Suppose, however, that you have not noticed that the signature is different than in the book and input this form (139). In previous Maple releases, `IsTetrad` will rightly tell (163) *is not a tetrad* because, for the signature set, (163) does not match the tetrads definition

> `e_[definition]`

$$e_{a,\mu} e_b^\mu = \eta_{a,b} \quad (164)$$

How could you discover that the problem with (163) is just a change in the signature and the position of the time-like component? New in Maple 2021, the command `IsTetrad` now analyses the situation before returning *false* and points out the solution

> `IsTetrad((139))`

Warning, the given components form a (null) tetrad, with a covariant spacetime index, only if you change the signature from (+ ---) to (+ + + -).

You can do that by entering (copy and paste): `Setup(signature="+ + + -")`

false (165)

Besides resolving the problem as indicated (you may not want to change the signature), also new in Maple 2021, you can resolve the situation directly using `Redefine`, that now also *redefines tetrads*. You want to redefine (139) from the signature where it is valid, (+ + + -), to the current signature (in such a case you do not need to specify `tosignature=...`)

> `Redefine((139), fromsignature = "+ + + - "`)

$$e_{a,\mu} = \begin{bmatrix} 0 & 0 & -1 & -IH \\ 0 & 0 & 0 & -I \\ 0 & \frac{Ir}{P} & 0 & 0 \\ \frac{Ir}{P} & 0 & 0 & 0 \end{bmatrix} \quad (166)$$

> `IsTetrad(%)`

Type of tetrad: null

(167)

This new functionality of [Redefine](#) is useful in general, and also to get a simpler tetrad (e. g. without radicals, as in the above) when you know such a simpler form for any other signature, as in this example.

Documentation advanced examples

One of the most important parts of the Physics project is its documentation; the illustration of the use of the package in different scenarios. The three relevant help pages for that are

- The [Physics,Examples](#)
- The [Physics,Tensors](#)
- The [Physics,Updates](#)

For Maple 2021, the first of these pages got extended with four sections: "[Vectors in Spherical Coordinates using Tensor Notation](#)", "[The equations of motion in curvilinear coordinates, tensor notation and Coriolis force](#)", "[The EnergyMomentum tensor for the Proca Lagrangian](#)" and "[The Gross-Pitaevskii field equations for a quantum system of identical particles](#)", covering new material in Vector Analysis, Mechanics and Classical Field Theory.

Miscellaneous

A number of minor changes happened in several places of the Physics library for Maple 2021, improving performance and the computational experience.

> *restart, with(Physics) :*

- When you set a spacetime metric,
 - a. the matrix form is now automatically shown - there is no need for additionally entering $g_{[]}$ to see it;
 - b. the signature in use is shown;
 - c. a kind of letter to represent *spaceindices* is automatically set.

> *Setup(metric = sc)*

Systems of spacetime coordinates are: $\{X = (r, \theta, \phi, t)\}$

Default differentiation variables for $d_{_}$, $D_{_}$ and $dAlembertian$ are: $\{X = (r, \theta, \phi, t)\}$

Setting `lowercaselatin_is` letters to represent space indices

The Schwarzschild metric in coordinates $[r, \theta, \phi, t]$

Parameters: $[m]$

Signature: $(- - - +)$

$$g_{\mu, \nu} = \begin{bmatrix} \frac{r}{2m-r} & 0 & 0 & 0 \\ 0 & -r^2 & 0 & 0 \\ 0 & 0 & -r^2 \sin(\theta)^2 & 0 \\ 0 & 0 & 0 & \frac{r-2m}{r} \end{bmatrix}$$

$$\left[\text{metric} = \left\{ (1, 1) = \frac{r}{2m-r}, (2, 2) = -r^2, (3, 3) = -r^2 \sin(\theta)^2, (4, 4) = \frac{r-2m}{r} \right\}, \right. \\ \left. \text{spaceindices} = \text{lowercaselatin_is} \right] \quad (168)$$

Using Setup, you also have the *quiet* option to avoid verbosity and reproduce the behavior of Maple releases previous to 2021:

$$\begin{aligned} > \text{Setup}(\text{metric} = [12, 12, 2], \text{quiet}) \\ & \quad [\text{metric} = \{ (1, 4) = -1, (2, 3) = 1, (4, 4) = -2 b^2 \zeta \xi \}] \end{aligned} \quad (169)$$

Note however that, when you load a metric from the database, a change in sign of the signature happens; compare with the one displayed in black above the matrix form of $g_{\mu, \nu}$ in (168) with the one set now:

$$\begin{aligned} > \text{Setup}(\text{signature}) \\ & \quad [\text{signature} = + + + -] \end{aligned} \quad (170)$$

Changes like this one in the signature are now *always presented on the screen*, unless you explicitly use the *quiet* option. The same display of the matrix form and signature happens when you [set the metric using g_](#); here, e.g. min, or minkowski or Minkowski all serve the same purpose

$$> g_{-}[mi]$$

The Minkowski metric in coordinates $[v, \zeta, \xi, u]$
Signature: $(+ + + -)$

$$g_{\mu, \nu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \quad (171)$$

• Several enhancements happened in *Library:-IsTensorialSymmetric* and

Library:-IsTensorialAntisymmetric, making the simplification of nested tensor functions, including the non-commutative tensorial differential operators, work more efficiently and successfully - see [related section above](#).

- Independent of the above, several enhancements happened in *Library:-GetTensorSymmetryProperties*. Consider a typical situation, of a product of noncommutative tensorial quantum operators in a Cartesian space

> *Coordinates(Cartesian)*

$$\text{Systems of spacetime coordinates are: } \{X = (x, y, z, t)\} \quad \{X\} \quad (172)$$

> *g_[eu]*

The Euclidean metric in coordinates [x, y, z, t]

$$g_{\mu, \nu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (173)$$

Set the coordinates as quantum operators

> *Setup(op = {X})*

** Partial match of 'op' against keyword 'quantumoperators'*

$$[\text{quantumoperators} = \{t, x, y, z\}] \quad (174)$$

In spite of being a product of noncommutative operators, this expression has some tensorial symmetries:

> $X_{\beta} X_{\alpha} \partial_{\nu} (\partial_{\rho} (f(X)))$

$$X_{\beta} X_{\alpha} \partial_{\nu} (\partial_{\rho} (f(X))) \quad (175)$$

> *Library:-GetTensorSymmetryProperties((175))*

$$\{[1, 2], [3, 4]\}, \emptyset \quad (176)$$

The above means (175) is symmetric with respect to exchange of its first and second indices, respectively α and β , and also under an exchange of ν , and ρ . Clear the setting of X_{α} as a noncommutative quantum operator to continue the presentation

> *Setup(clear, op = {X})*

** Partial match of 'op' against keyword 'quantumoperators'*

[quantumoperators = none] (177)

- Implement tensorial equation substitutions where the left-hand side is of type `+`

> g_[mi]

The Minkowski metric in coordinates [x, y, z, t]

Signature: (+ + + -)

$$g_{\mu, \nu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \quad (178)$$

> Define(A, G, F, B, H)

Defined objects with tensor properties

$$\{A, B, F, G, H, \gamma_{\mu}, \sigma_{\mu}, \partial_{\mu}, g_{\mu, \nu}, \gamma_{i, j}, T_{\mu, \nu}, \epsilon_{\alpha, \beta, \mu, \nu}, X_{\mu}\} \quad (179)$$

> G[v, α] A[α] F[μ, v] + G[ρ, α] B[α] F[μ, ρ]

$$G_{\nu, \alpha} A^{\alpha} F_{\mu}^{\nu} + G_{\rho, \alpha} F_{\mu}^{\rho} B^{\alpha} \quad (180)$$

In the above, *F* is a common *tensor-factor*. Consider then substituting in (180) the following subexpression of type `+`

> G[α, β] A[β] + G[α, β] B[β] = H[α]

$$G_{\alpha, \beta} A^{\beta} + G_{\alpha, \beta} B^{\beta} = H_{\alpha} \quad (181)$$

> Simplify(SubstituteTensor((181), (180)))

$$F_{\mu, \alpha} H^{\alpha} \quad (182)$$

To verify this result swap left-hand side and right-hand side in (181) and substitute back into (182), removing H_{α} , then compare with the starting expression (180)

> SubstituteTensor((rhs = lhs) ((181)), (182))

$$F_{\mu, \alpha} \left(A^{\beta} G_{\beta}^{\alpha} + B^{\beta} G_{\beta}^{\alpha} \right) \quad (183)$$

> Simplify((180) - (183))

$$0 \quad (184)$$

- [TensorArray](#) now accepts a new *listofequations* option, consistent with the previously existing *setofequations*. When the left-hand side of the equations returned is *matricial*, then provided the dimension of the matrices can be determined, the right-hand side of

the equations is the corresponding matrix of zeros.

- Change default simplification in `TensorArray`, from *do nothing* to [normal](#). This resolves the typical situation where we expect zeros in the components of tensorial expressions but those zeros are visible only after some form of simplification. You can still use the option *simplifier* to not have any simplification, e.g. *simplifier = (u -> u)*.
- Added the keyword *freeindices* in `Library:-FlipCharacterOfFreeIndices`, to be consistent with the implementation of that keyword in `Library:-FlipCharacterOfRepeatedIndices`
- Greek glyphs, for example ``μ`` and ``~μ`` are now valid tensor indices like `mu` and `~mu`. Note however that ``μ`` is not considered equal to `mu`.
- Change in design: tensors that are [defined using tensorial equations with tensors in the right-hand side](#), when their *contravariant form* is requested, instead of returning the *covariant form* multiplied by the metric to raise the indices, we now raise the indices of the tensors of the right-hand side.
- [Trace](#) is now more context-intelligent: if one side of a relation (equation or inequation) is verifiable as matricial, then the other side is considered too; e.g. if the left-hand side involves Dirac spinors and the right-hand side does not, when taking the trace the right-hand side is considered multiplied by the 4x4 identity matrix. For example,

> `Library:-DefaultAlgebraRules(Dirac)`

$$[\gamma_\mu, \gamma_\nu]_+ = -2 g_{\mu, \nu} \quad (185)$$

where the minus sign on the right-hand side of the algebra rule above is due to the change in sign in the signature introduced in (169),

> `Trace(expand((185)))`

$$-8 g_{\mu, \nu} = -8 g_{\mu, \nu} \quad (186)$$

You can verify these tensorial expressions as usual using [TensorArray](#)

> `TensorArray(expand((185)), matrix)`

** Partial match of 'matrix' against keyword 'performmatrixoperations'*

$$\left[\left[\begin{array}{cccc} -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \end{array} \right] = -2, \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] \right] = 0, \quad (187)$$

$$\begin{aligned}
& \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \quad \left[\begin{array}{cccc} -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \end{array} \right] = -2, \quad \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \\
& \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \\
& \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \quad \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \quad \left[\begin{array}{cccc} -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \end{array} \right] = -2, \\
& \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \\
& \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \quad \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \quad \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] = 0, \quad \left[\begin{array}{cccc} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{array} \right] = 2
\end{aligned}$$

where on the right-hand sides it is implicit a 4 x 4 identity matrix.

- Extend the definition of the type *Library:-PhysicsType:-DiracSpinor*: any anticommutative prefix having at least one spinor index is considered a DiracSpinor. So it is now sufficient to have a spinor index to be considered a DiracSpinor. This change makes room for the case of *mixed indices* (the happen in several contexts, e.g in quantum field models with 3/2 spinor fields). To save having to type long keywords repeatedly, use macro

> *macro(DiracSpinor = Library:-PhysicsType:-DiracSpinor) :*

> *Setup(anticommutativeprefix = Q, spinorindices = lowercase_ah)*

[anticommutativeprefix = {Q}, spinorindices = lowercaselatin_ah] (188)

> *type(Q[a, mu], DiracSpinor)*

true (189)

- There in one new *Physics:-Library:-EqualizeRepeatedIndices*, to equalize the repeated indices in different terms of tensorial expressions, as an first straightforward form of

simplification. This type of equalization also frequently helps to factorize tensorial expressions. Take for instance the expression (180)

> (180)

$$A^\alpha F_\mu^\nu G_{\nu,\alpha} + G_{\rho,\alpha} F_\mu^\rho B^\alpha \quad (190)$$

Some indices can be *equalized*

> *Library:-EqualizeRepeatedIndices*((180))

$$A^\alpha F_\mu^\nu G_{\nu,\alpha} + G_{\nu,\alpha} F_\mu^\nu B^\alpha \quad (191)$$

This result is factorizable

> *factor*((191))

$$F_\mu^\nu G_{\nu,\alpha} (A^\alpha + B^\alpha) \quad (192)$$

In Maple 2021 this new routine is automatically used by the Physics [Simplify](#) command.

See Also

[Index of New Maple 2021 Features](#), [Physics](#), [Computer Algebra for Theoretical Physics](#), [The Physics project](#), [The Physics Updates](#)