SLAC-PUB-12867, ZU-TH 25/07

S@M, a Mathematica Implementation of the Spinor Formalism

D. Maître [a,b] , P. Mastrolia [a]

[a] *Institut für Theoretische Physik*
University of Zürich
Winterthurerstrasse 190, CH-8057 Zürich

[b] *Stanford Linear Accelerator Center*
2575 Sand Hill Road
Menlo Park, CA 94025

Abstract

In this paper we present the package S@M (Spinors@Mathematica) which implements the spinor formalism in Mathematica. The package allows the use of complex-spinor algebra along with the multi-purpose features of Mathematica. The package defines the spinor objects with their basic properties along with functions to manipulate them. It also offers the possibility of evaluating the spinorial objects numerically at every computational step. The package is therefore well suited to be used the context of on-shell technology, in particular for the evaluation of scattering amplitudes at tree- and loop-level.

Key words: Spinor formalism, Mathematica
PACS: 11.80.Cr, 12.38.Bx

PROGRAM SUMMARY/NEW VERSION PROGRAM SUMMARY

Manuscript Title: S@M, a Mathematica Implementation of the Spinor Formalism
Authors: D. Maître, P. Mastrolia
Program Title:S@M
Programming language: Mathematica
Keywords: Spinor formalism, Mathematica.
PACS: 11.80.Cr, 12.38.Bx
Classification: 4.4 Feynman diagrams, 5 Computer Algebra, 11.1 General, High Energy Physics and Computing

Nature of problem: Implementation of the spinor formalism
Solution method: Mathematica implementation

Work supported in part by US Department of Energy contract DE-AC02-76SF00515

LONG WRITE-UP

1 Introduction

Theoretical understanding of background processes is essential to single out interesting signals in the rich landscape of events which will take place at the forthcoming CERN experiment, the Large Hadron Collider (LHC). Many methods are available for computing Standard Model backgrounds at the leading order (LO) in perturbation theory [1–5], based either on automatic summation of tree-level Feynman diagram, or off-shell recursive algorithms for currents [6–8]. But quantitative estimates for most processes require a calculation with next-to-leading order (NLO) accuracy - see for instance [9]. NLO calculations require knowledge of both virtual and real-radiation corrections to the LO process. While the real-radiation corrections can be computed using tree-level techniques, the bottleneck for the availability of results with NLO level accuracy [10–19] is the non-trivial evaluation of one-loop virtual corrections. New approaches tackling the evaluation of one-loop multi-parton amplitudes have recently been under intense development [16, 17, 20–51].

The spinor helicity formalism [52] for scattering amplitudes has proven an invaluable tool in perturbative computation since its development in the 1980's, being responsible for the discovery of compact representations of tree and loop amplitudes. Instead of Lorentz inner products of momenta, it relies on the more fundamental spinor products. These neatly capture the analytic properties of on-shell scattering amplitudes, like the factorization behavior on multi-particle-channels. The recent boost in the progress of evaluating on-shell scattering amplitudes is due to turning qualitative information on their analytic properties into quantitative tools for computing them.

On-shell methods [53] restrict the propagating states to the physical ones and the spinor-helicity formalism is therefore well suited to avoid the (intermediate) treatment of unphysical degrees of freedom whose effects disappear from final results. Moreover, on-shell methods are tailored for the *parallel* treatment of sets of diagrams which share a common kinematic structure, such as multi-particle poles at tree-level and branch-cuts at loop-level [54, 55]. They are therefore suitable for extracting analytic information from simpler amplitudes in a recursive/iterative fashion, since the singularities of scattering amplitude are determined by lower-point amplitudes in the case of poles and by lower-loop ones in the case of cuts [56–58].
On-shell methods were originally used in [59] and in the more recent systematized implementations for the completion of all six-gluon helicity amplitudes [30, 35, 36, 39, 41, 42, 46–48] and the calculation of the six-photon amplitudes [31, 60] in agreement with the numerical results of [29, 33] and

[61, 62] respectively.

The singularity information can be extracted by defining amplitudes for suitable complex, yet on-shell, values of the external momenta - an idea that stemmed from Witten's development of twistor string theory [63–66]. Generating complex momenta by modifying spinor variables, considered as fundamental objects, leads to new ways to exploit the kinematic properties of helicity amplitudes. The new-born complex momenta have the property of preserving overall momentum conservation and on-shell nature. Complex kinematics allow the exploration of singularities of on-shell amplitudes and the use of factorization information to reconstruct tree amplitudes recursively from their poles. The application of factorization in the on-shell method is realized through gluing lower-point tree amplitudes to form higher points ones linked by on-shell yet complex propagating particles. The construction of tree amplitudes via on-shell recursion essentially amounts to a reversal of the *collinear limit*. This is made possible by complexifying momenta in their spinorial representation, and results in a quadratic recursion, the BCFW recurrence relation [67], which works for massive particles [68–71] as well.

Complex kinematics are useful for the fulfillment of generalized unitarity conditions as well. At one loop, generalized unitarity corresponds to requiring *more* than two internal particles to be on-shell, such constraints cannot be realized in general with real Minkowski momenta.
The application of unitarity as an on-shell method of calculation is based on two principles: *i)* sewing tree amplitudes together to form one-loop amplitudes; *ii)* decomposing loop-amplitudes in terms of a basis of scalar loop-integrals [72,73]. Matching the generalized cuts of the amplitude with the cuts of basic integrals provides an efficient way to extract the *rational* coefficients from the decomposition. The unitarity method [35, 36] provides a technique for producing functions with the correct branch cuts in all channels [74], as determined by products of tree amplitudes.

The use of four-dimensional states and momenta in the cuts enable the construction of the (poly)logarithmic terms in the amplitudes, but generically drops rational terms, which have to be recovered independently.
More recent improvements to the unitarity method [40] use complex momenta within generalized unitarity, allowing for a simple and purely algebraic determination of box integral coefficients from quadruple-cuts. Using double- and triple-unitarity cuts have led to very efficient techniques for extracting triangle and bubble integral coefficients analytically [42, 48, 50, 75]. In particular in [42, 48, 50] the phase-space integration has been reduced to the extraction of residues in spinor variables and, at the occurrence, to trivial Feynman-parametric integration. This approach has been used to compute analytically some contributions to the six-gluon amplitude [42, 48], and the calculation of the complete six-photon amplitudes [60], whose cut-constructibility was

shown in [31] . Other approaches have combined the knowledge of the generic structure of loop-*integrand* [25, 76] with the simplification induced by cut-constraints, ending up with a unitarity-motivated loop-integral decomposition [32, 33, 62, 75].

The four-dimensional version of the unitarity method leaves the pure rational-function terms in the amplitudes undetermined. New approaches to computing rational terms use an optimized organization of Feynman diagrams, by focusing the standard tensor reduction to tensor integrals which could generate the rational terms [30, 31].
A recent investigation [77] on the source of rational terms has been exploring the idea of their generation via a set of Lorentz-violating counterterms.
Alternatively, these rational functions can be characterized by their kinematic poles. An efficient means for constructing these terms from their poles and residues is based on BCFW-like recursion relations [43–47].
The rational parts of amplitudes can be also detected with D-dimensional unitarity cuts [37, 49, 58, 78, 79], and in [49, 51] the benefits of four-dimensional spinorial integration [42, 48, 50] have been extended to work within the dimensional regularization scheme and with massive particles.

In this paper we present the package S@M (Spinors@Mathematica) which implements the spinor formalism in Mathematica. The package allows the use of complex-spinor algebra along with the multi-purpose features of Mathematica. The package provides

- the definitions of the spinor objects with their basic properties,
- functions to manipulate them
- numerical evaluation.

These capabilities make the package `S@M` suitable for, for example,

- the generation of complex spinors associated with solutions of multi-particle factorization and of generalized-cut conditions;

- the implementation of BCFW-like recurrence relations for constructing high-multiplicity tree amplitudes [67, 68] and rational coefficients [43–45, 80];

- the decomposition of massive momenta onto massless ones, useful for the implementation of the MHV-rules [81, 82] and for the BCFW-shift of massive legs [70];

- the algebraic manipulation of products of tree amplitudes with complex spinors sewn in unitarity-cuts.

The tool presented here is therefore oriented towards the evaluation of helicity

amplitudes at LO and beyond, which are relevant for the phenomenology of the Standard Model, for the study of the so called constructible theories [83], and for the investigation of the ultraviolet-behavior of gravity (see [84, 85] and references therein).

The paper is organized as follows. Section 2 defines the notation used by the package, Section 3 describes the implementation. In last section, we show three more involved examples of the use of the package `S@M`.

2 Notation

This section describes the conventions and notation used in the package.

2.1 Two-dimensional Spinors

The two dimensional spinors λ and $\tilde{\lambda}$ for a massless fermion with four momentum p are defined through the Dirac equation

$$\not{p}\lambda(p) = 0\,, \qquad \tilde{\lambda}(p)\not{p} = 0\,, \tag{1}$$

with

$$\not{p} = p_\mu \sigma^\mu \equiv \begin{pmatrix} p_- & -p_-^\perp \\ -p_+^\perp & p_+ \end{pmatrix}, \qquad \sigma^\mu = (1, \vec{\sigma})\ , \tag{2}$$

$$\sigma^1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma^2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma^3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\ . \tag{3}$$

The solutions of opposite chirality are

$$\lambda_a(p) = c \begin{pmatrix} p_+ \\ p_+^\perp \end{pmatrix} \qquad \text{and} \qquad \tilde{\lambda}_{\dot{a}}(p) = \tilde{c} \begin{pmatrix} p_+ & p_-^\perp \end{pmatrix}. \tag{4}$$

We define $\underline{\lambda}$ and $\underline{\tilde{\lambda}}$ as the spinor with up-indices, obtained by contracting the spinors λ and $\tilde{\lambda}$ with the ϵ-tensor,

$$\underline{\lambda}(p) \equiv \lambda^a(p) = \epsilon^{ab}\lambda_b(p) = c\begin{pmatrix} p_+^\perp & -p_+ \end{pmatrix}, \; \underline{\tilde{\lambda}} \equiv \tilde{\lambda}^{\dot{a}} = \epsilon^{\dot{a}\dot{b}}\tilde{\lambda}_{\dot{b}} = \tilde{c}\begin{pmatrix} p_-^\perp \\ -p_+ \end{pmatrix} \quad (5)$$

where $\epsilon^{ab} = \epsilon^{\dot{a}\dot{b}} = i\sigma_2$.
The underlined notation for spinors carrying up-indices has been introduced here to avoid the introduction of indices in the package S@M. By making the normalization choice

$$c = \tilde{c} = \frac{1}{\sqrt{p_+}}\,, \quad (6)$$

we have

$$\not{P}_{\dot{a}a} = \tilde{\lambda}_{\dot{a}}\lambda_a\,, \qquad \not{P}^{\dot{a}a} = \tilde{\lambda}^{\dot{a}}\lambda^a, \quad (7)$$

so that the expressions for the two-dimensional spinors read,

$$\lambda_a(p) = \begin{pmatrix} \sqrt{p_+} \\ \frac{p_+^\perp}{\sqrt{p_+}} \end{pmatrix} \qquad \text{and} \qquad \tilde{\lambda}_{\dot{a}}(p) = \begin{pmatrix} \sqrt{p_+}\,, & \frac{p_-^\perp}{\sqrt{p_+}} \end{pmatrix}\,. \quad (8)$$

The the above formulas hold as well for spinors associated to any massless complex four-vectors. For massless real momenta we can use the identity,

$$\sqrt{p_+}\sqrt{p_-} = \sqrt{p_+p_-} = \sqrt{p_-^\perp p_+^\perp} = \sqrt{p_1^2 + p_2^2}\,, \quad (9)$$

to write the corresponding spinor,

$$\lambda_a(p) = \frac{1}{\sqrt{p_+}}\begin{pmatrix} p_+ \\ p_+^\perp \end{pmatrix} = \begin{pmatrix} \sqrt{p_+} \\ \sqrt{p_-}e^{i\phi} \end{pmatrix}\,, \qquad e^{i\phi} = \frac{p_1 + ip_2}{\sqrt{p_1^2 + p_2^2}}\,. \quad (10)$$

$$\tilde{\lambda}_{\dot{a}}(p) = \begin{pmatrix} \sqrt{p_+}\,, & \sqrt{p_-}e^{-i\phi} \end{pmatrix}\,. \quad (11)$$

2.2 Four-dimensional Spinors

Positive and negative energy solutions of the four dimensional massless Dirac equation are identical up to normalization conventions. By closely following the definitions of [57], the solutions of definite helicity

$$u_\pm(k) = \tfrac{1}{2}(1 \pm \gamma_5)u(k) \quad \text{and} \quad v_\mp(k) = \tfrac{1}{2}(1 \pm \gamma_5)v(k) \tag{12}$$

and their conjugates

$$\overline{u_\pm(k)} = \overline{u(k)}\tfrac{1}{2}(1 \mp \gamma_5) \quad \text{and} \quad \overline{v_\mp(k)} = \overline{v(k)}\tfrac{1}{2}(1 \mp \gamma_5)\ . \tag{13}$$

can be chosen to be equal to each other [1]. For the numerical evaluation of spinor products with slashed matrices insertion, we use the Dirac γ matrices defined as,

$$\gamma^0 \ = \ \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} \end{pmatrix}\ , \qquad \gamma^i \ = \ \begin{pmatrix} \mathbf{0} & \sigma^i \\ -\sigma^i & \mathbf{0} \end{pmatrix}\ , \qquad \gamma_5 \ = \ \begin{pmatrix} \mathbf{0} & \mathbf{1} \\ \mathbf{1} & \mathbf{0} \end{pmatrix}\ , \tag{14}$$

where the entries $\mathbf{0}$ and $\mathbf{1}$ are 2×2-matrices. In this representation, the massless spinors can be chosen as follows,

$$u_+(k) = v_-(k) = \frac{1}{\sqrt{2}}\begin{bmatrix} \lambda_a(p) \\ \lambda_a(p) \end{bmatrix}, \quad u_-(k) = v_+(k) = \frac{1}{\sqrt{2}}\begin{bmatrix} \tilde{\lambda}_{\dot{a}}(p) \\ -\tilde{\lambda}_{\dot{a}}(p) \end{bmatrix}. \tag{15}$$

2.2.1 Label Representation

The spinor formalism is based on the algebraic manipulation of basic objects such as the spinor inner products. They are independent of the explicit representation used for expressing the spinors themselves. It is useful to have a notation for the spinors that is not bound to a particular explicit representation. In S@M we will call this notation the "Label" representation. We define

$$\begin{aligned} u_+(k_i) \ &= \ v_-(k_i) \equiv |k_i^+\rangle \equiv |i\rangle \equiv \lambda(k_i)\,, \\ u_-(k_i) \ &= \ v_+(k_i) \equiv |k_i^-\rangle \ \equiv |i] \equiv \tilde{\lambda}(k_i)\ , \end{aligned} \tag{16}$$

and for the conjugate spinors

$$\begin{aligned} \overline{u_+(k_i)} = \overline{v_-(k_i)} &\equiv \left[k_i^+\right| \equiv [i| \equiv \underline{\lambda}(k_i), \\ \overline{u_-(k_i)} = \ \overline{v_+(k_i)} &\equiv \langle k_i^-| \equiv \langle i| \equiv \underline{\tilde{\lambda}}(k_i)\ , \end{aligned} \tag{17}$$

[1] Note that for negative energy solutions, the helicity is the negative of the chirality or γ_5 eigenvalue

2.3 Spinor Products

The scalar (or inner) products for massless spinors are defined as,

$$\langle i\, j\rangle \equiv \langle k_i^- | k_j^+ \rangle \;=\; \overline{u_-(k_i)} u_+(k_j) = \lambda^a(k_i)\lambda_a(k_j) = \underline{\lambda}(k_i)\lambda(k_j), \tag{18}$$

$$[i\, j] \equiv \langle k_i^+ | k_j^- \rangle \;=\; \overline{u_+(k_i)} u_-(k_j) = \tilde{\lambda}_{\dot{a}}(k_i)\tilde{\lambda}^{\dot{a}}(k_j) = \tilde{\lambda}(k_i)\underline{\tilde{\lambda}}(k_j). \tag{19}$$

The helicity projection implies that products like $[i|j\rangle$ vanish.

In the rest of the paper we will call *spinor products* the $\langle\,\rangle$- and $[\,]$-type spinor inner products, and not the external spinor-product for the spinorial decomposition of slashed-matrices, see Eq.(7).

The spinor products are, up to a phase, square roots of Lorentz products, in fact they are related to momenta inner product through the identity,

$$\langle i\, j\rangle [j\, i] = \langle i^- | j^+ \rangle \langle j^+ | i^- \rangle = \mathrm{Tr}(\tfrac{1}{2}(1-\gamma_5)\; \not{k}_i\; \not{k}_j) = 2k_i \cdot k_j = s_{ij}, \tag{20}$$

where $s_{ij} \;=\; (k_i + k_j)^2 \;=\; 2k_i \cdot k_j$.

We also have the useful identities:

Gordon identity:

$$\langle i|\gamma^\mu|i] \;= [i|\gamma^\mu|i\rangle \;=\; 2k_i^\mu. \tag{21}$$

Projection operator:

$$|i\rangle[i| = \frac{(1+\gamma_5)}{2}\; \not{k}_i\;, \qquad |i]\langle i| = \frac{1-\gamma_5}{2}\; \not{k}_i\;, \qquad |i\rangle[i| + |i]\langle i| = \not{k}_i\;. \tag{22}$$

Antisymmetry:

$$\langle j\, i\rangle = -\langle i\, j\rangle, \qquad [j\, i] = -[i\, j], \qquad \langle i\, i\rangle = [i\, i] = 0\;. \tag{23}$$

Schouten identity:

$$\langle i\, j\rangle\langle k\, l\rangle \;=\; \langle i\, k\rangle\langle j\, l\rangle + \langle i\, l\rangle\langle k\, j\rangle. \tag{24}$$

Spinor re-definition:

$$\not{k}|i\rangle = |k(i)], \qquad \langle i|\not{k} = -[k(i)|,$$
$$\not{k}|i] = |k(i)\rangle, \qquad [i|\not{k} = -\langle k(i)|. \tag{25}$$

2.4 Massive Spinors Products

In the spinor helicity formalism, spinors associated to particles of momentum $p_I$which are solutions of the massive Dirac equation,

$$\overline{u^{\pm}(p_I)}\ (\not{p}_I - m_I)\ u^{\pm}(p_I) = 0\ ,$$

can be as well represented with *bra-ket* notation:

$$u^{+}(p_I) \equiv |I\rangle\ , \qquad \overline{u^{+}(p_I)} \equiv \langle I|\ , \tag{26}$$
$$u^{-}(p_I) \equiv |I]\ , \qquad \overline{u^{-}(p_I)} \equiv [I|\ , \tag{27}$$

where the angle-bracket denote the two spin-polarization states with respect to a fixed reference axis, rather than helicity states. Spinor products in this case can be written as [70, 71]

$$\overline{u^{-}(p_I)}u^{+}(p_J) \equiv \langle I\,J\rangle\ , \qquad \overline{u^{+}(p_I)}u^{-}(p_J) \equiv [I\,J]\ ,$$
$$\overline{u^{+}(p_I)}u^{+}(p_J) \equiv [I J\rangle\ , \qquad \overline{u^{-}(p_I)}u^{-}(p_J) \equiv \langle I J]\ . \tag{28}$$

One can use the light-cone decomposition of a massive spinor in terms of two massless one [86]. Accordingly, by introducing an arbitrary massless reference momentum q, we can construct a massless momentum (p_i) associated to any massive one (p_I),

$$p_i^{\mu} = p_I^{\mu} - \frac{p_I^2}{2p_I \cdot q} q^{\mu} = p_I^{\mu} - \frac{m_I^2}{2p_i \cdot q} q^{\mu}\ , \qquad p_i^2 = 0 = q^2\ . \tag{29}$$

Correspondingly, the spinor variables read,

$$|I\rangle = |i\rangle + \frac{m_I}{[i\,q]}|q]\ , \tag{30}$$
$$|I] = |i] + \frac{m_I}{\langle i\,q\rangle}|q\rangle\ . \tag{31}$$
$$\tag{32}$$

With the above formulas we can express the spinor product for massive particles of Eqs (28) in terms of the ones involving only massless spinors [71]

$$\langle I\, J\rangle = \langle i\, j\rangle \ , \tag{33}$$
$$[I\, J] = [i\, j] \ , \tag{34}$$
$$[I\ J\rangle = \left(\frac{m_I}{s_{iq}} + \frac{m_J}{s_{jq}}\right) [i|\not{q}|j\rangle \ , \tag{35}$$
$$\langle I\ J] = \left(\frac{m_I}{s_{iq}} + \frac{m_J}{s_{jq}}\right) \langle i|\not{q}|j] \ , \tag{36}$$

where $s_{iq} = (p_i+q)^2 = 2p_i\cdot q = [iq]\langle qi\rangle$. We notice that in the *r.h.s* of the above equations only massless spinors do appear. Therefore the above definitions of spinor products associated to massive particles can have their straightforward implementation in `S@M`.

3 Mathematica Implementation

In the following section we present the Mathematica implementation `S@M`. The Mathematica notebook file `S@M_Definitions.nb` containing all the examples shown in this section is distributed with the package.

3.1 Input and Output

In the notebook, both the input and the output has been designed to look like the familiar bra-ket representation.

$$\langle \bullet, \bullet\rangle\, , [\bullet, \bullet]\, , \langle \bullet, \bullet, \bullet] \ldots$$

In the console version, the functions can only by inputed using their names, in the notebook, several other input methods are available.

- Output from a previous evaluation can be copy-pasted as input for the next evaluation.
- Some basic functions and objects like the spinor products are available from a palette that is opened when the package is loaded within the notebook.
- Spinor products can be entered using the characters sequnce "ESC < ESC", "ESC > ESC","[", "]", "—" if the input format is `TraditionalForm`.

The examples in the following sections are displayed as they appear in the Mathematica notebook.

Similar output can be obtained in the console version by using the command `TraditionalForm`. The package is loaded using

```
<< Spinors`
-------  SPINORS @ MATHEMATICA (S@M)  -------


        Version:  S@M 1.0 (7-OCT-2007)

 Authors:
    Daniel Maitre (SLAC),
    Pierpaolo Mastrolia (University of Zurich)

A list of all functions provided by the package
is stored in the variable
    $SpinorFunctions
```

The package should be loaded at the beginning of the Mathematica session.

3.2 General Structure

Massless spinor variables are the fundamental objects for the construction of both inner spinor-products (associated to Lorentz invariants), and external spinor-products (associated to slashed-matrices).
In `S@M`, massless spinors are expected to be *labeled* by either symbols or integers. The spinor products are independent of the explicit representation of the spinors. Objects that do not refer to an explicit representation, like spinor products in their $\langle\bullet,\bullet\rangle$, $[\bullet,\bullet]$ form, are said to be in the "label" representation. To allow more flexibility in the use of the package two explicit spinor representations have been implemented, the two-dimensional (Weyl) and the four-dimensional (Dirac) ones. The different representations and the names are summarized in the following table.

Label	2 dimensional	4 dimensional
$\|s\rangle$,$\langle s\|$	$\lambda(k_s)$, $\underline{\lambda}(k_s)$	$u_+(k_s)$, $\overline{u_-(k_s)}$
`s`	`La[s]`,`CLa[s]`	`USpa[s]`, `UbarSpa[s]`
$\|s]$,$[s\|$	$\tilde{\lambda}(k_s)$, $\underline{\tilde{\lambda}}(k_s)$	$u_-(k_s)$, $\overline{u_+(k_s)}$
`s`	`Lat[s]`,`CLat[s]`	`USpb[s]`, `UbarSpb[s]`

The functions provided by the packages act on (defined) spinor-labels, and work in all the representations. The flexibility of `S@M` relies in the multiple use of a given symbol defined as massless spinor, which can represent at the same time either the spinor itself or its associated slashed matrix, with the automatic understanding of its interpretation according to the context.

3.3 New Functions

The new functions introduced by the package are described in the following section. A list of all functions can be found in Appendix A.

3.3.1 Spinors Declaration

In `S@M` objects called (and declared as) spinors are considered to be the solution of the *massless* Dirac equation. That is not a restriction on the usability of the package, since solutions of the massive Dirac equation can be constructed from massless spinors. The symbol representing a given spinor is used to represent at the same time, and depending on the context, the spinor itself, its vector or the corresponding slashed matrix.

`DeclareSpinor`

The function `DeclareSpinor` can be called with one or a sequence of arguments. It declares its arguments to be spinors. If undeclared variables are used as spinors, some automatic properties will not be applied and most functions cannot be used.

```
DeclareSpinor[a, b, c, d, e, f, i, j, k, l, s, t, u]
{a, b, c, d, e, f, i, j, k, l, s, t, u} added to the list of spinors
```

Integer labels for spinors do not have to be declared, for more details, see the section on `Sp` below. If a symbol is defined as a spinor, it can also be used to represent both its Lorentz vector (see page 14) or its corresponding slashed matrix.

`SpinorQ`

`SpinorQ` tests whether its argument has been declared as a spinor or not, it returns `True` if so and `False` otherwise. It can be used for example in patterns.

```
SpinorQ[a]
MatchQ[b, _?SpinorQ]
True
True
```

`UndeclareSpinor`

The function `UndeclareSpinor` removes its argument from the list of spinors.

`Sp`

Spinors can be labeled by integers using the function `Sp`. The object `Sp[n]`

is considered as a spinor. In the scalar products `Spaa`, `Spab`, `Spba` and `Spbb` described below, integer arguments are automatically wrapped into the `Sp` function.

```
Sp[1]
Sp[1] // FullForm
SpinorQ[Sp[1]]
```

```
1
Sp[1]
True
```

In `StandardForm` and `TraditionalForm`, the function `Sp` is not displayed, only its argument.

The function can be made visible by using `FullForm`.

3.3.2 Spinor Representations

`La, Lat, CLa, CLat`

The two-dimensional representations of the spinor `s` are given in the following table.

$\lambda(k_s)$	$\tilde{\lambda}(k_s)$	$\underline{\lambda}(k_s)$	$\underline{\tilde{\lambda}}(k_s)$
`La[s]`	`Lat[s]`	`CLa[s]`	`CLat[s]`

The arguments of these functions are spinor labels or integers. In the latter case, the argument, say `i`, is automatically converted to the corresponding spinor label, using `Sp[i]` (see above).

`La[s]`, `Lat[s]`, `CLa[s]` and `CLat[s]` are linear in their spinor argument.

```
La[2 a - b]
% // TraditionalForm
CLat[a + b + c]
% // TraditionalForm
```

```
2 La[a] - La[b]
2 λ(a) – λ(b)
CLat[a] + CLat[b] + CLat[c]
λ̲̃(a) + λ̲̃(b) + λ̲̃(c)
```

The contraction of two different two-dimensional spinors is implemented using the Mathematica `Dot` operator and automatically displayed in a standard order (appropriate for the numerical evaluation, see page 21).

```
CLa[1].La[2] // TraditionalForm
La[2].CLa[1] // TraditionalForm
```

```
λ̲(1).λ(2)
λ̲(1).λ(2)
```

`USpa, USpb, UbarSpa, UbarSpb`

The four-dimensional representation of the spinor `s` are given in the following table.

$u_+(k_s)$	$u_-(k_s)$	$\overline{u_-(k_s)}$	$\overline{u_+(k_s)}$
`USpa[s]`	`USpb[s]`	`UbarSpa[s]`	`UbarSpb[s]`

These four functions are linear in their spinor argument.

```
USpa[2 a - b]
% // TraditionalForm
UbarSpb[a + b + c]
% // TraditionalForm
```

2 USpa[a] - USpa[b]

$2u_+(a) - u_+(b)$

UbarSpb[a] + UbarSpb[b] + UbarSpb[c]

$\overline{u_+}(a) + \overline{u_+}(b) + \overline{u_+}(c)$

The contraction of two different two-dimensional spinors is implemented using the Mathematica `Dot` operator and automatically displayed in a standard order (appropriate for the numerical evaluation, see page 21).

```
UbarSpa[1].USpa[2] // TraditionalForm
USpa[2].UbarSpa[1] // TraditionalForm
```

$\overline{u_-}(1).u_+(2)$

$\overline{u_-}(1).u_+(2)$

3.3.3 Lorentz Vectors

In `S@M` we call Lorentz vector any 4-dim vector of the form, $k^\mu = (k^0, k^1, k^2, k^3)$.

`DeclareLVector`

The function `DeclareLVector` can be called with one or a sequence of arguments. It declares its arguments as Lorentz vectors.

Momenta associated to spinors (declared through `DeclareSpinor`) do not need to be declared, and one can use the symbol of the spinor to represent the corresponding Lorentz vector as well. To represent the Lorentz vector of a spinor labeled by an integer `i`, one may use `Sp[i]`.

```
DeclareLVector[p1, p2, p3]
{p1, p2, p3} added to the list of Lorentz vectors
```

`UndeclareLVector`

The function `UnDeclareLVector` removes its argument from the list of the Lorentz vectors.

```
UndeclareLVector[p3]
p3 removed from the list of Lorentz vectors
```

`LVectorQ`

`LVectorQ` tests whether its argument can be interpreted as a Lorentz vector or not. It returns `True` if so and `False` otherwise. It can be used for example in patterns.

```
LVectorQ[p1]
LVectorQ[p3]
LVectorQ[Sp[5]]
True

False

True

MatchQ[p1, _?LVectorQ]
True
```

3.3.4 Minkowski Products

`MP[p,q]`

The function `MP[p,q]` represents the Minkowski product

$$p \cdot q = p^0 q^0 - \vec{p} \cdot \vec{q}.$$

The two arguments can be either spinors or Lorentz vectors or linear combinations thereof.

Integer arguments are interpreted as the four vectors associated to the spinor `Sp[i]`. `MP` is symmetric in its arguments, so they are automatically sorted. The vector product is linear.

```
MP[2, 1]
MP[2 p1, 3 p2 - 5 Sp[1]]
MP[1, 2]

6 MP[p1, p2] - 10 MP[p1, 1]
```

`MP2[p]`

The function `MP2[p]` is a shortcut for `MP[p,p]`.

```
MP2[3 p1 - p2]
9 MP[p1, p1] - 6 MP[p1, p2] + MP[p2, p2]
```

The function `MP` can be used with explicit four vector representations:

```
MP[{5.3, 1.4, -1.2, 3.7}, {1.2, -0.7, 0, 1.1}]
3.27
```

3.3.5 Invariants

`s[i,j]`

The function `s[i,j]` represents the kinematic invariant given by the square of the sum of two momenta,

$$s_{ij} = (p_i + p_j)^2.$$

The two arguments can be either spinors or Lorentz vectors.

Integer arguments are interpreted as the four vectors associated to the spinor `Sp[i]`. Since the scalar product `s` is symmetric in its arguments, they are automatically sorted.

```
s[a, b]
s[4, 1]
sab
s1×4
```

The function `s` also accepts more than two arguments (which can be spinors or Lorentz vectors) for multi-particle invariants,

$$s_{i\ldots j} = (p_i + \ldots + p_j)^2.$$

```
s[1, 2, 3]
s[4, 3, 2]
s1×2×3
s2×3×4
```

3.3.6 Slashed Matrices

Slashed matrices are in general contractions of Lorentz momenta with gamma-matrices $\not{P} = P^\mu \gamma_\mu$. There are three representations for the slashed matrices in the package, as summarized in the following table.

Label	2 dimensional	4 dimensional
$\not{P}$	$P^{\dot{a}a}$, $P_{a\dot{a}}$	$\not{P}$
`Sm[P]`	`Sm2[P]`, `CSm2[P]`	`Sm4[P]`
$\|b\rangle\langle a\| + \|a\rangle[b\|$	$\tilde{\underline{\lambda}}(k_b).\underline{\lambda}(k_a)$, $\lambda(k_a).\tilde{\lambda}(k_b)$	$\overline{u_+(a)}.u_-(b) + \overline{u_-(b)}.u_+(a)$
`SmBA[b,a]`	`SmBA2[b,a]`, `CSmBA2[b,a]`	`SmBA4[b,a]`

In `S@M` slashed matrices are used either inside spinor products or as explicit matrices. There are four different types of slashed matrices.

- slashed matrices corresponding to a declared (massless) spinor. These slashed matrices are labeled with the same symbol as their spinor. The slashed matrix associated with the massless spinor `a` are represented by: `Sm[a]`, in the label representation; `Sm2[a]` or `CSm2[a]`, in the two-dimensional representation; and `Sm4[a]`, in the four-dimensional representation. Inside spinor products in the label representation, the function `Sm` is not necessary and can be omitted.
- external products of spinors. These slashed matrices are represented by the functions: `SmBA`, in the label representation; `SmBA2` or `CSmBA2`, in the two dimensional representation; and `SmBA4`, in the four-dimensional representation.
- slashed matrices corresponding to a declared (possibly massive) vector. These slashed matrices are labeled with the same symbol as their vector. The slashed matrix associated with the vector `P` are represented by: `Sm[P]` in the label representation; `Sm2[P]` or `CSm2[P]`, in the two dimensional representation; and `Sm4[p]`, in the four-dimensional representation. Inside spinor products in the label representation, the function `Sm` is not necessary and can be omitted.
- other slashed matrices unrelated to a declared vector or spinor. These slashed matrices have to be declared. The slashed matrix declared with the symbol `Q` is represented by `Sm[Q]` in the label representation, `Sm2[Q]` or `CSm2[Q]`, in the two dimensional representation and `Sm4[Q]`, in the four-dimensional representation. Inside spinor products in the label representation, the function `Sm` is not necessary and can be omitted.

`DeclareSMatrix`

The function `DeclareSMatrix` can be called with one or a sequence of arguments. It declares its arguments to be slashed matrices. Slashed matrices corresponding to declared spinors and Lorentz vectors are are labeled by the same symbol as the spinor or Lorentz vector and thus do not need to be re-defined. If undeclared variables are used as slashed matrices, some automatic properties will not be applied, and most functions cannot be used.

```
DeclareSMatrix[P, Q, R, S, T]
{P, Q, R, S, T} added to the list of slashed matrices
```

UndeclareSMatrix

The function UndeclareSMatrix removes its argument from the list of the slashed matrices.

SMatrixQ

SMatrixQ tests whether its argument has been declared as a slashed matrix or not, it returns True if so and False otherwise. It can be used for example in patterns.

```
SMatrixQ[P]
MatchQ[Q, _?SMatrixQ]
True

True
```

SmBA

The object SmBA[b,a] represents slashed matrices formed by the tensor product of two spinors, like

$$|b\rangle\langle a| + |a\rangle[b| \,.$$

The arguments a and b are spinors labels. SmBA is linear in both arguments. If the two arguments are equal, SmBA[a,a] is automatically replaced by Sm[a].

```
SmBA[1, 2]
SmBA[b + c, a]
SmBA[a, a]
SmBA[1, 2]

SmBA[b, a] + SmBA[c, a]

Sm[a]
```

3.3.7 Slashed Matrices Representations

Sm

The object Sm is used for slashed matrices corresponding to previously declared spinors and Lorentz vectors. Sm can be called with one argument, being either a spinor label or a vector label. In particular, slashed matrices associated either to spinors (declared through DeclareSpinor), or to vectors (declared through DeclareLVector) are automatically declared. One can use the symbol of the spinor, say s, or the one of the vector, say P

to represent the corresponding slashed matrix by means of `Sm[s]` or `Sm[P]` respectively.

The object `Sm` is linear in its argument.

```
SMatrixQ[Sm[a]]
Sm[a + 2 b]
True

Sm[a] + 2 Sm[b]
```

`Sm2, CSm2`

The two-dimensional representations of the slashed matrix `P` are given in the following table.

$P^{\dot{a}a}$	$P_{a\dot{a}}$
`Sm2[P]`	`CSm2[P]`

The functions `Sm2` and `CSm2` are linear in their spinor argument. The contraction with another slashed matrix or with a spinor is implemented using the Mathematica `Dot` operator.

```
Sm2[a - b]
Sm2[a + b].CSm2[c + d]
Sm2[a] - Sm2[b]

Sm2[a].CSm2[c] + Sm2[a].CSm2[d] + Sm2[b].CSm2[c] + Sm2[b].CSm2[d]
```

`Sm4`

The four-dimensional representations of the slashed matrix `P` is given by `Sm4[P]`. This function is linear in its spinor argument. The contraction of two different four-dimensional matrices or with four-dimensional spinors is implemented using the Mathematica `Dot` operator.

```
Sm4[a - b]
Sm4[a].Sm4[a + b]
Sm4[a] - Sm4[b]

Sm4[a].Sm4[b]
```

`SmBA2`

In the two-dimensional representation, the slashed matrices build from two spinors

$$\tilde{\underline{\lambda}}^{\dot{b}}.\underline{\lambda}^{a} \qquad \text{and} \qquad \lambda_a.\tilde{\lambda}_{\dot{b}}$$

are represented by the two function `SmBA2[b,a]`, `CSmBA2[b,a]` respectively. These functions are linear in both spinor arguments.

```
SmBA2[b, a + b]
CSmBA2[b, a + Sp[1]]
Sm2[b] + SmBA2[b, a]
CSmBA2[b, a] + CSmBA2[b, 1]
```

Their multiplication with other slashed matrices are implemented using the Mathematica `Dot` operator. `SmBA2` and `CSmBA2` are replaced by their tensor product representation when they are inserted in a spinor chain.

```
Lat[1].SmBA2[2, 3].La[4] // TraditionalForm
CLa[a].CSmBA2[b, c].Sm2[d].La[e] // TraditionalForm
```

$$-\underline{\lambda}(3).\lambda(4)\,\tilde{\lambda}(2).\tilde{\underline{\lambda}}(1)$$

$$\underline{\lambda}(a).\lambda(c)\,\tilde{\lambda}(b).\mathrm{Sm2}(d).\lambda(e)$$

`SmBA4`

The slashed matrix build from two spinors,

$$|b]\langle a| + |b]\langle a| \ ,$$

is represented in the four-dimensional representation by the function `SmBA4[b,a]`. This function is linear in both spinor arguments.

```
SmBA4[b, a + b]
Sm4[b] + SmBA4[b, a]
```

Its multiplication with other slashed matrices is implemented using the Mathematica `Dot` operator. `SmBA4` is replaced by its tensor product representation when inserted in a spinor chain.

```
UbarSpa[1].SmBA4[2, 3].USpb[4] // TraditionalForm
UbarSpb[a].SmBA4[b, c].Sm4[d].USpb[e] // TraditionalForm
```

$$-\overline{u_-}(1).u_+(3)\,\overline{u_+}(4).u_-(2)$$

$$-\overline{u_+}(b).u_-(a)\,\overline{u_-}(c).\mathrm{Sm4}(d).u_-(e)$$

3.3.8 Spinor Products

Spinor products are represented in `S@M` by four different objects: `Spaa`, `Spab`, `Spba` and `Spbb`, according to the following table.

$\langle a...b\rangle$	$\langle a...b]$	$[a...b\rangle$	$[a...b]$
`Spaa[a,...,b]`	`Spab[a,...,b]`	`Spba[a,...,b]`	`Spbb[a,...,b]`

The left- and right-most arguments are spinors and the intermediate arguments are slashed matrices or objects that can be interpreted as slashed matrices such as spinors or Lorentz vectors. `Spaa` and `Spbb` expect an even number of (declared) arguments whereas `Spab` and `Spba` expect an odd number

of (declared) arguments. If not all the arguments are integers (and therefore automatically interpreted as spinors) or declared either as spinors, Lorentz vectors or slashed matrices, the properties listed below cannot be applied.

-Standard order

The spinor products have a normal ordering for their arguments. If the rightmost and leftmost elements are spinors, the middle elements are slashed matrices (or can be interpreted as such) and in addition if the spinors are not in the standard order, the spinor product is ordered using the identities

$$\begin{aligned} \langle b\,a\rangle &= -\langle a\,b\rangle\,, & [b\,a] &= -[a\,b]\,,\\ \langle b|Q...P|a\rangle &= -\langle a|P...Q|b\rangle\,, & [b|Q...P|a] &= -[a|P...Q|b]\,,\\ [b|P|a\rangle &= \langle a|P|b]\,, & [b|Q...P|a\rangle &= \langle a|P...Q|b]\,. \end{aligned}$$

A special case of these identities is the on-shell condition

$$\langle a\,a\rangle = 0, \quad [a\,a] = 0\,.$$

```
Spbb[a, a]
Spaa[nospinor, nospinor]
Spaa[b, a]
Spbb[a, P, Q, b]
Spaa[a, b] Spbb[a, b]
```

```
0
```

$\langle \text{nospinor} \,|\, \text{nospinor} \rangle$

$-\langle a \mid b \rangle$

$-[b \mid Q \mid P \mid a]$

$-\langle a \mid b \rangle\, [b \mid a]$

The normal ordering of the `Spaa` and `Spbb` products are opposite so that the products $\langle a\,b\rangle\,[b\,a]$ are displayed in this usual way.

The same rules apply to the spinor products written in the two- and four dimensional representations.

```
CLa[2].La[1]
Lat[3].CLat[4]
CLa[1].CSm2[3].Sm2[2].La[1]
```

```
-CLa[1].La[2]
```

```
-Lat[4].CLat[3]
```

```
-CLa[1].CSm2[2].Sm2[3].La[1]
```

```
CLat[3].CSm2[2].CLa[1]
```

```
Lat[3].Sm2[2].La[1]
```

```
UbarSpa[2].USpa[1]
UbarSpb[3].USpb[4]
UbarSpa[1].Sm4[3].Sm4[2].USpa[1]
```

```
-UbarSpa[1].USpa[2]
```

```
-UbarSpb[4].USpb[3]
```

```
-UbarSpa[1].Sm4[2].Sm4[3].USpa[1]
```

```
UbarSpb[3].Sm4[2].USpa[1]
```

```
UbarSpa[1].Sm4[2].USpb[3]
```

-Linearity

If the arguments of a spinor product have been defined as spinors using `DeclareSpinor` or `DeclareSMatrix`, then linear combinations of spinors are automatically expanded.

```
Spaa[a + b, c]
Spaa[2 b, c]
Spab[a + 2 b, P, 3 c - d]
⟨a | c⟩ + ⟨b | c⟩
2 ⟨b | c⟩
3 ⟨a | P | c] - ⟨a | P | d] + 6 ⟨b | P | c] - 2 ⟨b | P | d]
Lat[a + b].Sm2[2 c + d].La[e] // TraditionalForm
UbarSpb[a + b].Sm4[2 c + d].USpa[e] // TraditionalForm
2 λ̃(a).Sm2(c).λ(e) + λ̃(a).Sm2(d).λ(e) + 2 λ̃(b).Sm2(c).λ(e) + λ̃(b).Sm2(d).λ(e)
2 u_(e).Sm4(c).u_(a) + 2 u_(e).Sm4(c).u_(b) + u_(e).Sm4(d).u_(a) + u_(e).Sm4(d).u_(b)
```

The linearity works for integer labels too, but one has to write the `Sp` function explicitly, since the sum or product of the integers is done before wrapping the result with the function `Sp`.

```
Spaa[1 + 2, 2 coeff]
Spaa[coeff Sp[1], 2]
Spaa[Sp[1] + Sp[2], 3]
⟨3|2 coeff⟩
coeff ⟨1 | 2⟩
⟨1 | 3⟩ + ⟨2 | 3⟩
```

-Syntax correction

When all the arguments of a spinor product are declared either as spinor, Lorentz vector or slashed matrix and their number does not match the expected number for the particular spinor product type (even for `Spaa`, `Spbb`, for odd for `Spab`, `Spba`), the type of the spinor product is changed automatically and a warning is issued.

```
Spaa[1, Q, 2]
Spab[1, 2]
Spba[1, Q, P, 2]
Spbb[1, Q, 2]
Spaa::wrongNbrSM :
 Wrong number of slashed matrices in the <...> product . Automatically changed to <...]
⟨1|Q|2]
Spab::wrongNbrSM :
 Wrong number of slashed matrices in the <...] product . Automatically changed to <...>
⟨1|2⟩
Spba::wrongNbrSM :
 Wrong number of slashed matrices in the [...> product . Automatically changed to [...]
-[2 | P | Q | 1]
Spbb::wrongNbrSM :
 Wrong number of slashed matrices in the [...] product . Automatically changed to <...]
⟨2|Q|1]
```

-Slashed matrices insertion

The Dirac equation and on-shell condition

$$\not{a}\,|a\rangle = 0, \quad \not{a}\,|a] = 0, \quad \not{a}\not{a} = a^2 = 0,$$

are used for spinors used as slashed matrices. The inserted `SmBA` objects are automatically expanded, as shown in the following examples.

```
Spab[1, a, a]
Spbb[1, a, a, 2]
```

0

0

```
Lat[a].Sm2[a].La[b]
CLa[1].CSm2[a].Sm2[a].La[2]
```

0

0

```
UbarSpb[a].Sm4[a].USpa[b]
UbarSpa[1].Sm4[a].Sm4[a].USpa[2]
```

0

0

```
Spab[1, SmBA[2, 3], 4]
Spaa[1, SmBA[2, 3], SmBA[4, 5], 6]
```

$-\langle 1\,|\,3\rangle\,[4\,|\,2]$

$-\langle 1\,|\,3\rangle\,\langle 5\,|\,6\rangle\,[4\,|\,2]$

```
Lat[1].SmBA2[b, a].La[2] // TraditionalForm
UbarSpa[1].SmBA4[b, a].USpb[2] // TraditionalForm
```

$\underline{\lambda}(a).\lambda(2)\,\tilde{\lambda}(1).\underline{\tilde{\lambda}}(b)$

$\overline{u_-}(a).u_+(1)\,\overline{u_+}(2).u_-(b)$

3.3.9 Spinor Manipulations

`ExpandSToSpinors, ConvertSpinorsToS`

The function `ExpandSToSpinors`, `ConvertSpinorsToS` convert invariants `s` to spinor products and conversely.

```
ExpandSToSpinors[s[1, 2] s[2, 3]]
```

$\langle 1\,|\,2\rangle\,\langle 2\,|\,3\rangle\,[2\,|\,1]\,[3\,|\,2]$

```
ConvertSpinorsToS[%]
```

$s_{1\times 2}\,s_{2\times 3}$

```
Lat[1].Sm2[2].CSm2[1].Sm2[3].La[1]
ConvertSpinorsToS[%]
```

Lat[1].Sm2[2].CSm2[1].Sm2[3].La[1]

$s_{1\times 2}\,s_{1\times 3}$

```
UbarSpa[1].Sm4[2].Sm4[1].USpa[3]
ConvertSpinorsToS[%] // TraditionalForm
```

UbarSpa[1].Sm4[2].Sm4[1].USpa[3]

$\overline{u_-}(1).u_+(3)\,s_{1\times 2}$

`SpOpen, SpClose`

The function `SpOpen` decomposes spinor chains containing any slashed ma-

trix that corresponds to a massless spinor with products of smaller spinor chains, by applying the definition of such a matrix in terms of its opposite-chirality spinors,

$$\not{k} = |k]\langle k| + |k\rangle[k| \ .$$

The function `SpClose` has the reverse effect as that of `SpOpen`. It attempts to replace products of spinor products with spinor chains containing slashed matrices.
Both the functions can take either one or two arguments. The first argument is the expression to be manipulated; the second argument must be a spinor. With one argument, the functions open or close wherever possible.

```
SpOpen[Spaa[1, 2, 3, 4, 5, 6]]
⟨1 | 2⟩ ⟨3 | 4⟩ ⟨5 | 6⟩ [3 | 2] [5 | 4]
SpOpen[Spab[1, 2, P, 3, 4]]
-⟨1 | 2⟩ ⟨3 | P | 2] [4 | 3]
SpOpen[Spaa[1, 2, P, 3, 4, Q, 5, 6]]
-⟨1 | 2⟩ ⟨5 | 6⟩ ⟨3 | P | 2] ⟨4 | Q | 5] [4 | 3]
SpOpen[Spaa[1, 2, 3, 4, 1, 2, 3, 4, 2, 3, 1, 2]]
SpClose[%]
⟨1 | 2⟩^3 ⟨2 | 3⟩ ⟨3 | 4⟩^2 [3 | 1] [3 | 2]^2 [4 | 1] [4 | 2]
⟨1|2|3|4|1|2|4|3|2|1|3|2⟩
Lat[5].Sm2[4].CSm2[3].Sm2[2].La[1] // TraditionalForm
SpOpen[%] // TraditionalForm
SpClose[%] // TraditionalForm
λ̃(5).Sm2(4).CSm2(3).Sm2(2).λ(1)
λ̲(1).λ(2) λ̲(3).λ(4) λ̃(3).λ̲̃(2) λ̃(5).λ̲̃(4)
λ̃(5).Sm2(4).CSm2(3).Sm2(2).λ(1)
UbarSpa[1].Sm4[2].Sm4[3].Sm4[4].USpb[5] // TraditionalForm
SpOpen[%] // TraditionalForm
SpClose[%] // TraditionalForm
ū_(1).Sm4(2).Sm4(3).Sm4(4).u_(5)
ū_(1).u_+(2) ū_(3).u_+(4) ū_+(3).u_(2) ū_+(5).u_(4)
ū_(1).Sm4(2).Sm4(3).Sm4(4).u_(5)
```

If there are different possibilities of reconstructing the spinor chain, `SpClose` does not search for the longest possible spinor chain. The result will depend on the ordering of the spinor labels and might not be invariant under relabeling of the spinor labels.

```
SpOpen[Spaa[1, 2, P, 4, Q, 2]]
SpClose[%]
⟨1 | 2⟩ ⟨2 | Q | 4] ⟨4 | P | 2]
⟨1 | 2⟩ ⟨4 | P | 2 | Q | 4]
```

If a spinor is given as a second argument, `SpOpen` and `SpClose` will only open or close spinor chains containing this specified spinor.

```
SpOpen[Spab[a, b, c, d, e], c]
⟨a | b | c] ⟨c | d | e]
SpOpen[Spab[a, b, c, d, e]]
SpClose[%, c]
⟨a | b⟩ ⟨c | d⟩ [c | b] [e | d]
-⟨a | b⟩ ⟨d | c | b] [e | d]
Spaa[1, 2, 3, 4] // To2DimSpinor
SpOpen[%, 2]
SpClose[%]
CLa[1].CSm2[2].Sm2[3].La[4]
CLa[1].La[2] Lat[2].Sm2[3].La[4]
CLa[1].CSm2[2].Sm2[3].La[4]
SpOpen[UbarSpb[d].Sm4[c].Sm4[b].USpb[a]] // TraditionalForm
SpClose[%, c] // TraditionalForm
-u_(b).u+(c) u+(b).u_(a) u+(d).u_(c)
u+(b).u_(a) u_(b).Sm4(c).u_(d)
```

To2DimSpinor

The function To2DimSpinor converts spinor products in the label representation into the two-dimensional representation.

```
To2DimSpinor[Spaa[a, b, c, d] + Spab[a, c, d]];
% // TraditionalForm
λ̃(d).Sm2(c).λ(a) + λ̲(a).CSm2(b).Sm2(c).λ(d)
To2DimSpinor[Spab[a, P, d]];
% // TraditionalForm
λ̃(d).Sm2(P).λ(a)
```

There is an ambiguity in the conversion of explicit slashed matrices (see Sm), when they are not embedded in a spinor chain. In those cases To2DimSpinor convert the slashed matrices to the Sm2 type. In case of Dot products of slashed matrices, the product is transformed to a Dot product starting with a Sm2 matrix.

```
To2DimSpinor[Sm[a + b] + SmBA[c, d]]
Sm2[a] + Sm2[b] + SmBA2[c, d]
To2DimSpinor[Sm[a].Sm[b].Sm[c]]
Sm2[a].CSm2[b].Sm2[c]
```

To4DimSpinor

The function To4DimSpinor converts spinor products in the label representation into the four-dimensional representation.

```
To4DimSpinor[Spaa[a, b, c, d] + Spab[a, c, d]];
% // TraditionalForm
u_(a).Sm4(c).u_(d) + u_(a).Sm4(b).Sm4(c).u+(d)
To4DimSpinor[Spab[a, P, d]];
% // TraditionalForm
u_(a).Sm4(P).u_(d)
To4DimSpinor[Sm[a + b] + SmBA[c, d]]
Sm4[a] + Sm4[b] + SmBA4[c, d]
```

`ToSpinorLabel`

The function `To2SpinorLabel` converts spinor products in the two- or four-dimensional representation into the label notation.

```
To2DimSpinor[Spaa[a, b, c, d] + Spab[a, c, d]] // TraditionalForm
ToSpinorLabel[%]
```

$\tilde{\lambda}(d).\mathrm{Sm2}(c).\lambda(a) + \underline{\lambda}(a).\mathrm{CSm2}(b).\mathrm{Sm2}(c).\lambda(d)$

```
⟨a | b | c | d⟩ + ⟨a | c | d]
La[1].Lat[1] + La[2].Lat[2] + La[1].Lat[2] // TraditionalForm
% // ToSpinorLabel
```

$\lambda(1).\tilde{\lambda}(1) + \lambda(1).\tilde{\lambda}(2) + \lambda(2).\tilde{\lambda}(2)$

```
Sm[1] + Sm[2] + SmBA[2, 1]
Sm2[P].CSm2[Q] // ToSpinorLabel
CSm2[P].Sm2[Q].La[3] // ToSpinorLabel
P.Q
⟨P.Q⟩[3]
UbarSpb[4].Sm4[3].Sm4[2].USpb[1] // TraditionalForm
% // ToSpinorLabel
UbarSpa[2].Sm4[R].Sm4[Q].Sm4[P].USpb[1] // TraditionalForm
% // ToSpinorLabel
```

$\overline{u_+}(4).\mathrm{Sm4}(3).\mathrm{Sm4}(2).u_-(1)$

$[4|3|2|1]$

$\overline{u_-}(2).\mathrm{Sm4}(R).\mathrm{Sm4}(Q).\mathrm{Sm4}(P).u_-(1)$

$\langle 2|R|Q|P|1]$

`Compactify`

Given a slashed matrix, say $\not{P}$, and a spinor of the $|\bullet\rangle$-type, say $|b\rangle$, one can construct the massless spinor

$$\not{P}\,|b\rangle \ ,$$

which indeed is of the opposite chirality $|\bullet]$-type. In fact, one can define,

$$\not{P}\,|b\rangle \equiv |P(b)] \,, \quad \langle b|\,\not{P} \equiv -\,[P(b)| \ ,$$

and, similarly,

$$\not{P}\,|b] \equiv |P(b)\rangle \,, \quad [b|\,\not{P} \equiv -\,\langle P(b)| \ .$$

In the package, the notation used for the spinor obtained by applying the slashed matrix to the spinor `a` is `P[a]`.

```
SpinorQ[P[b]]
SpinorQ[(P.Q)[b]]
True
True
```

These objects are recognized as spinors. The application of more than one slashed matrix to a spinor is done using the Mathematica `Dot` operator. The function `Compactify` uses this definition to reduce spinor products with inserted slashed matrices to spinor products of two spinor objects.

```
Compactify[Spab[a, P, b]]
Compactify[Spaa[a, P, Q, b]]
Compactify[Spbb[a, P, Q, b]]
Compactify[CLa[a].CSm2[P].Sm2[Q].La[b]]
```

$\langle a|P(b)\rangle$

$\langle a|(P.Q)[b]\rangle$

$[(Q.P)[a]|b]$

```
CLa[a].La[(P.Q)[b]]
```

One can specify a spinor as a second argument for `Compactify`. In this case the spinor products containing the given spinor are compactified in such a way that the specified spinor is left untouched.

```
Compactify[Spab[a, P, b] + Spab[a, P, c], b]
Compactify[Spaa[a, P, Q, b] + Spab[a, P, c], b]
Compactify[UbarSpa[b].Sm4[P].USpb[c] + UbarSpa[a].Sm4[P].Sm4[Q].USpa[b], b] // TraditionalForm
Compactify[Lat[c].Sm2[P].La[b] + CLa[a].CSm2[P].Sm2[Q].La[b], b] // TraditionalForm
```

$\langle a \mid P \mid c] - [P(a) \mid b]$

$-\langle b \mid (Q.P)[a]\rangle + \langle a \mid P \mid c]$

$\overline{u_-}(b).u_+(P(c)) - \overline{u_-}(b).u_+((Q.P)[a])$

$\underline{\lambda}(b).\lambda(P(c)) - \underline{\lambda}(b).\lambda((Q.P)[a])$

The functions `ACompactify[x,a]` and `BCompactify[x,a]` work the same way as `Compactify[x,a]` but only spinor products containing $[a\rangle$ or $|a]$ respectively are compactified.

```
Compactify[Spab[1, P, 3] + Spab[3, P, 1], 3]
ACompactify[Spab[1, P, 3] + Spab[3, P, 1], 3]
BCompactify[Spab[1, P, 3] + Spab[3, P, 1], 3]
```

$-\langle P(1) \mid 3\rangle + [3 \mid P(1)]$

$-\langle P(1) \mid 3\rangle + \langle 1 \mid P \mid 3]$

$\langle 3 \mid P \mid 1] + [3 \mid P(1)]$

`UnCompact`

The function `UnCompact` uncompactifies the spinor products compactified with `Compactify`.

```
Compactify[Spab[a, P, b]]
UnCompact[%]
Compactify[Spab[a, P, Q, R, b], b]
UnCompact[%]
Compactify[Lat[b].Sm2[P].La[a], b]
UnCompact[%]
```

$\langle a|P(b)\rangle$

$\langle a|P|b]$

$-[(R.Q.P)[a] \mid b]$

$\langle a|P|Q|R|b]$

```
-Lat[P[a]].CLat[b]
Lat[b].Sm2[P].La[a]
```

One can specify a spinor as a second argument. In this case only the spinor products where the Dirac matrices are compactified onto the specified spinor

will be uncompactified.

```
Compactify[Spaa[a, P, Q, b] + Spab[b, P, c], b]
UnCompact[%]
UnCompact[%%, c]
UnCompact[%%%, a]
⟨b | P(c)⟩ - ⟨b | (Q.P)[a]⟩
⟨a | P | Q | b⟩ + ⟨b | P | c]
-⟨b | (Q.P)[a]⟩ + ⟨b | P | c]
⟨b | P(c)⟩ + ⟨a | P | Q | b⟩
```

`Schouten`

The function `Schouten` applies the Schouten identities

$$\langle i\,j\rangle\langle k\,\ell\rangle = \langle i\,\ell\rangle\langle k\,j\rangle + \langle i\,k\rangle\langle j\,\ell\rangle\ , \qquad [i\,j][k\,\ell] = [i\,\ell][k\,j] + [i\,k][j\,\ell]\ .$$

There are three different applications of the function.

`Schouten[x,i,j,k,l]`

The function with four spinor arguments will search `x` for occurrences of the products $\langle i\,j\rangle\langle k\,\ell\rangle$ or $[i\,j][k\,\ell]$ and replace it using the above identities.

```
Spaa[i, j] Spaa[k, l]
Schouten[%, i, j, k, l]
⟨i | j⟩ ⟨k | l⟩
-⟨i | l⟩ ⟨j | k⟩ + ⟨i | k⟩ ⟨j | l⟩
CLa[i].La[j] CLa[k].La[l] // TraditionalForm
Schouten[%, i, j, k, l] // TraditionalForm
λ̲(i).λ(j) λ̲(k).λ(l)
λ̲(i).λ(k) λ̲(j).λ(l) - λ̲(i).λ(l) λ̲(j).λ(k)
```

`Schouten[x,i,j,k]`

The function with three spinor arguments will search for occurrences of the spinor product $\langle i\,j\rangle$ or $[i\,j]$ and will try to use the Schouten identity to combine it with the spinor `k`.

```
Spaa[i, j] Spaa[k, l] Spaa[a, b] + Spaa[i, j] Spaa[a, l] Spaa[k, b]
-⟨a | l⟩ ⟨b | k⟩ ⟨i | j⟩ + ⟨a | b⟩ ⟨i | j⟩ ⟨k | l⟩
Schouten[%, i, j, k]
-⟨a | l⟩ (⟨b | j⟩ ⟨i | k⟩ - ⟨b | i⟩ ⟨j | k⟩) + ⟨a | b⟩ (-⟨i | l⟩ ⟨j | k⟩ + ⟨i | k⟩ ⟨j | l⟩)
```

`Schouten[x,l]`

The function with one spinor arguments will search for structures like

$$\frac{\langle\ell\,u\rangle}{\langle\ell\,s\rangle\langle\ell\,t\rangle}\ , \qquad \frac{[\ell\,u]}{[\ell\,s][\ell\,t]}\ ,$$

and will use the Schouten identities to split them into partial fractions,

$$\frac{\langle\ell\,u\rangle}{\langle\ell\,s\rangle\langle\ell\,t\rangle} = \frac{\langle s\,u\rangle}{\langle\ell\,s\rangle\langle s\,t\rangle} - \frac{\langle t\,u\rangle}{\langle\ell\,t\rangle\langle s\,t\rangle}\ , \qquad \frac{[\ell\,u]}{[\ell\,s][\ell\,t]} = \frac{[s\,u]}{[\ell\,s][s\,t]} - \frac{[t\,u]}{[\ell\,t][s\,t]}\ .$$

The function also works for spinor products with embedded Dirac matrices.

```
Spaa[l, u]
----------------------
Spaa[l, s] Spaa[l, t]
Schouten[%, l]
   ⟨l | u⟩
-------------- -
⟨l | s⟩ ⟨l | t⟩
   ⟨s | u⟩          ⟨t | u⟩
-------------- - --------------
⟨l | s⟩ ⟨s | t⟩   ⟨l | t⟩ ⟨s | t⟩
```

`ASchouten, BSchouten`

The function `ASchouten` and `BSchouten` behave like `Schouten` but apply the Schouten identity selectively on $|\ell\rangle$-variables and $|\ell]$-variables respectively.

```
Spaa[l, u]                    Spbb[l, u]
---------------------- + ----------------------
Spaa[l, S, T, s] Spaa[l, t]   Spbb[l, s] Spbb[l, t]


      ⟨l | u⟩          [u | l]
-------------------- - ----------
⟨l | t⟩ ⟨l | S | T | s⟩ [s | l] [t | l]
Schouten[%, l]
ASchouten[%%, l]
BSchouten[%%%, l]
      ⟨t | u⟩          ⟨s | T | S | u⟩      [u | s]      [u | t]
- ---------------- + ---------------------- - ---------- + ----------
  ⟨l | t⟩ ⟨s | T | S | t⟩ ⟨l | S | T | s⟩ ⟨s | T | S | t⟩ [s | l] [t | s] [t | l] [t | s]
      ⟨t | u⟩          ⟨s | T | S | u⟩      [u | l]
- ---------------- + ---------------------- - ----------
  ⟨l | t⟩ ⟨s | T | S | t⟩ ⟨l | S | T | s⟩ ⟨s | T | S | t⟩ [s | l] [t | l]
      ⟨l | u⟩          [u | s]      [u | t]
-------------------- - ---------- + ----------
⟨l | t⟩ ⟨l | S | T | s⟩ [s | l] [t | s] [t | l] [t | s]
```

`ASpinorReplace[x,a,n],BSpinorReplace[x,a,n]`

The functions `ASpinorReplace` and `BSpinorReplace` replace the spinor `a` in expression `x` with `n`. `ASpinorReplace` only replaces the spinors $|a\rangle$ and `BSpinorReplace` replaces only $|a]$. Slashed matrices corresponding to the spinor `a` will be split according to

$$\not{a} = |a\rangle\,[a| + |a]\,\langle a|$$

and the appropriate component will be replaced.

```
ASpinorReplace[Spab[2, 1, 2], 2, a]
BSpinorReplace[Spab[2, 1, 2], 2, a]
⟨a|1|2]
⟨2|1|a]
CLa[a].La[b] + Lat[a].CLat[b] // TraditionalForm
ASpinorReplace[%, b, c] // TraditionalForm
UbarSpa[a].USpa[b] - UbarSpb[b].USpb[a] // TraditionalForm
BSpinorReplace[%, b, c] // TraditionalForm
```

$\underline{\lambda}(a).\lambda(b) - \tilde{\lambda}(b).\underline{\tilde{\lambda}}(a)$

$\underline{\lambda}(a).\lambda(c) - \tilde{\lambda}(b).\underline{\tilde{\lambda}}(a)$

$\overline{u_-}(a).u_+(b) - \overline{u_+}(b).u_-(a)$

$\overline{u_-}(a).u_+(b) - \overline{u_+}(c).u_-(a)$

`ASpinorShift[x,a,s],BSpinorShift[x,a,s]`

The functions `ASpinorShift` and `BSpinorShift` shift the spinor variable `a` in expression `x` with `s`. `ASpinorShift` only shifts the spinor $|a\rangle$ and `BSpinorShift` only shifts the spinors $|a]$. Slashed matrices corresponding to the spinor `a` will be split according to

$$\not{a} = |a\rangle\,[a| + |a]\,\langle a|$$

and the appropriate component will be shifted.

```
ASpinorShift[Spaa[1, 2] Spbb[4, 2], 2, 3]
BSpinorShift[Spaa[1, 2] Spbb[4, 2], 2, 3]
(⟨1 | 2⟩ + ⟨1 | 3⟩) [4 | 2]
⟨1 | 2⟩ ([4 | 2] + [4 | 3])
ASpinorShift[CLa[a].La[b] + Lat[a].CLat[b], b, z c] // TraditionalForm
BSpinorShift[UbarSpa[a].USpa[b] - UbarSpb[b].USpb[a], b, z c] // TraditionalForm
```

$$\underline{\lambda}(a).\lambda(b) + z\,\underline{\lambda}(a).\lambda(c) - \tilde{\lambda}(b).\underline{\tilde{\lambda}}(a)$$

$$\overline{u_-}(a).u_+(b) - \overline{u_+}(b).u_-(a) - z\,\overline{u_+}(c).u_-(a)$$

The shift parameter `s` will be interpreted as a spinor of the appropriate chirality. It can be a sum of spinors.

```
ASpinorShift[Spab[a, c, b], c, x d + 5 e]
(⟨a | c⟩ + x ⟨a | d⟩ + 5 ⟨a | e⟩) [c | b]
```

To account for more generic spinor definitions and shifts composite-spinors may be required. We can use

$$\texttt{a},\quad \texttt{P[b]},\quad \texttt{(P.Q)[c]},\quad \texttt{(P.Q.R)[d]},\quad \ldots$$

as the shift argument of the `Shift`-functions to represent the compactified objects like

$$|a\rangle,\quad |P(b)\rangle,\quad |P.Q(c)\rangle,\quad |P.Q.R(d)\rangle,\quad \ldots$$

in the case of an `A`-shift. The expressions obtained using this kind of arguments can be later un-compactified by using `UnCompact`.

```
DeclareSMatrix[M, M1, M2]
ASpinorShift[Spaa[a, b], b, M[c]]
% // UnCompact
{M, M1, M2} added to the list of slashed matrices
⟨a | b⟩ + ⟨a | M(c)⟩
⟨a | b⟩ + ⟨a | M | c]
ASpinorShift[Spaa[a, b], b, (M1.M2)[c]]
% // UnCompact
⟨a | b⟩ + ⟨a | (M1.M2)[c]⟩
⟨a | b⟩ + ⟨a | M1 | M2 | c⟩
```

`ShiftBA[b,a,z][x]`

The function `ShiftBA[b,a,z]` performs the shifts combination

$$|b] \to |b] - z\,|a]\,, \qquad |a\rangle \to |a\rangle + z\,|b\rangle\,. \tag{37}$$

The arguments `a` and `b` must be declared as spinors.

```
ShiftBA[3, 2, z][Spaa[2, 4]]
ShiftBA[3, 2, z][Spbb[3, 1]]
⟨2 | 4⟩ + z ⟨3 | 4⟩
-z [2 | 1] + [3 | 1]
ShiftBA[c, b, z][UbarSpa[a].USpa[b]] // TraditionalForm
ShiftBA[3, 2, z][Lat[a].Sm2[2].La[b]] // TraditionalForm
```

$$\overline{u_-}(a).u_+(b) + z\,\overline{u_-}(a).u_+(c)$$

$$z\,\underline{\lambda}(b).\lambda(3)\,\tilde{\lambda}(2).\underline{\tilde{\lambda}}(a) + \tilde{\lambda}(a).\mathrm{Sm2}(2).\lambda(b)$$

3.3.10 Constants

`Gamma0, Gamma1, Gamma2, Gamma3, Gamma5` are the γ-matrices in the representation (14).

```
{Gamma0, Gamma1, Gamma2, Gamma3, Gamma5} // TraditionalForm
SequenceForm @@ MatrixForm /@ {Gamma0, Gamma1, Gamma2, Gamma3, Gamma5} // TraditionalForm
```

$$\{\gamma_0, \gamma_1, \gamma_2, \gamma_3, \gamma_5\}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \\ 0 & i & 0 & 0 \\ -i & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

The γ-matrices can be used as slashed matrices too.

```
Spab[1, Gamma1, 1] / 2
```

$$\frac{1}{2}\,\langle 1 \mid \gamma_1 \mid 1]$$

`ProjPlus, ProjMinus` are the helicity projectors $(1 \pm \gamma_5)/2$.

```
{ProjPlus, ProjMinus} // TraditionalForm
SequenceForm @@ MatrixForm /@ {ProjPlus, ProjMinus} // TraditionalForm
```

$$\{P_+, P_-\}$$

$$\begin{pmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{1}{2} & 0 & -\frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}$$

The symbol `$SpinorFunctions` contains a list of all functions of the package.

```
$SpinorsFunctions
{ACompactify, ASchouten, ASpinorReplace, ASpinorShift, BCompactify, BSchouten, BSpinorReplace,
 BSpinorShift, CLa, CLat, Compactify, ConvertSpinorsToS, CSm2, CSmBA2, DeclareLVector,
 DeclareLVectorMomentum, DeclareSMatrix, DeclareSpinor, DeclareSpinorMomentum,
 ExpandSToSpinors , GammaC, Gamma1, Gamma2, Gamma3, Gamma5, GenMomenta, La, Lat,
 LVectorQ, MP, MP2, Num4V, PfromBiSpinor, PfromCSm2, PfromSm2, PfromSm4, ProjMinus,
 ProjPlus, s, Schouten, ShiftBA, Sm, Sm2, Sm4, SMatrixQ, SmBA, SmBA2, SmBA4, Sp, Spaa,
 Spab, Spba, Spbb, SpClose, SpinorQ, SpOpen, To2DimSpinor, To4DimSpinor, UbarSpa,
 UbarSpb, UnCompact, UndeclareLVector, UndeclareSMatrix, UndeclareSpinor, USpa, USpb}
```

3.3.11 Numerics

The spinor products have a numerical implementation. The first step to use this numerical implementation is to define or generate the four momenta for the spinors. For this there are several possibilities described in the following section.

3.4 Momentum Generation

`DeclareSpinorMomentum`

This function takes as first argument the spinor whose momentum should be set, the second specifies the momentum. It can be one of the following.

- The four vector in the form of a list `{E,p1,p2,p3}`, (note that it is the responsibility of the user to make sure that the vector is indeed an on-shell vector)
- two explicit spinors, the first in the λ form and the second in the $\tilde{\lambda}$ form of (9). This can be used with the `La` and `Lat` functions. If the momentum associated with a spinor is defined using two spinors, these spinors will be used for the numerical evaluation of the spinor products.

If the first argument was not declared as a spinor using `DeclareSpinor`, it will be declared.

```
DeclareSpinorMomentum[a, {√14, 1, 2, 3}]
DeclareSpinorMomentum[b, {√11, 1, 1, 3}]
DeclareSpinorMomentum[c, {3, 2, 1, -2}]
DeclareSpinorMomentum[d, {{0.9}, {-0.5 - 0.2 i}}, {{0.9, -0.5 + 0.2 i}}]
DeclareSpinorMomentum[e, La[a] // N, Lat[b] // N]
Momentum for spinor a set to {√14, 1, 2, 3}.
Momentum for spinor b set to {√11, 1, 1, 3}.
Momentum for spinor c set to {3, 2, 1, -2}.
Momentum for spinor d set to {0.55 + 0. i, -0.45 + 0. i, -0.18 + 0. i, 0.26 + 0. i}.
Momentum for spinor e set to
 {3.4927 + 0.0766203 i, 1.00053 + 0.451416 i, 1.48451 + 0.0325661 i, 3.03298 - 0.0766203 i}.
```

`DeclareLVectorMomentum`

One can also define four-vectors that are not associated with a spinor (for example when they are not light-like) with the function `DeclareLVectorMomentum`. It takes two arguments, first the symbol for the vector to be defined and second the value of the four vector in the form of a list.

```
DeclareLVectorMomentum[p, {1, 0, 0, 0}]
{p} added to the list of Lorentz vectors
Four Momentum p set to {1, 0, 0, 0}.
```

`GenMomenta`

The function `GenMomenta[{s1,...,sn}]` generates arbitrary on-shell four

momenta for the spinors `s1,...,sn`. They are generated so that they sum up to zero momentum.

```
GenMomenta[{1, 2, 3, 4, d, e}]
Momenta for the spinors d, e, 1, 2, 3, 4 generated.
```

The function `GenMomenta[{s1,...,sn}->{p0,p1,p2,p3}]` generates arbitrary on-shell four momenta for the spinors `s1,...,sn`, so that the sum of these momenta is equal to the vector $p = (p_0, p_1, p_2, p_3)$.

```
GenMomenta[{1, 2, 3, 4} → {2, 1, 0, 0}]
Momenta for the spinors 1, 2, 3, 4 generated.
GenMomenta[{a, b, c, d} → {√2, 1, 0, 1}]
Momenta for the spinors a, b, c, d generated.
GenMomenta[{1, 2, 3, 4} → {0, 0, 0, 1}]
Momenta for the spinors 1, 2, 3, 4 generated.
```

The momenta can be produced with `GenMomenta` in a reproducible way by seeding the random number generator with the Mathematica command `SeedRandom`.

```
SeedRandom[1234]
GenMomenta[{1, 2, 3, 4, 5}]
Momenta for the spinors 1, 2, 3, 4, 5 generated.
```

`GenMomenta` can take as a second argument the precision with which the momenta should be generated. Too large precision slows down the numerical evaluation.

```
GenMomenta[{1, 2, 3, 4, 5}, 50]
Num4V[1]
Momenta for the spinors 1, 2, 3, 4, 5 generated.
{0.60813329745052498386439659585223412903627903903183 72,
 -0.52081262932486510111714577454992367910665370105327 11,
 -0.03320514031408368242899764451490741933638219897818 81,
 0.31221423936215877784548869245375091396392282439174 01}
```

3.5 Numerical Functions

Once the momenta associated with the spinors are declared or generated, several numerical functions are accessible.

`Num4V`

The value of the momentum associated to the spinor a is accessible through the function `Num4V`.

```
Num4V[a]
{1.70189185885142 96, 1.40603656156837 39, -0.42601343196147 34, 0.85907487598355 81}
```

`Num4V` is linear in the same way as `MP`.

```
Num4V[2 a + b]
{3.721558979299065, 2.9C8C933C3275366, -C.7449C6615859C27, 2.CC1498467C57566}
```

The explicit numerical representation of the spinors are accessible by applying the Mathematica function `N` on the two- or four-dimensional representation, both are represented in a matrix notation. `La`, `CLat`, `USpa` and `USpb` are column vectors, whereas `CLa`, `Lat`, `UbarSpa` and `UbarSpb` are row vectors.

```
{La[c], CLa[c]} // TraditionalForm
% // N // TraditionalForm
{λ(c), λ̲(c)}
{( 2.48327 ), ( 1.38665 − 0.942001 i   −2.48327 )}
                1.38665 − 0.942001 i
{Lat[c], CLat[c]} // TraditionalForm
% // N // TraditionalForm
{λ̃(c), λ̲̃(c)}
{( 2.48327   1.38665 + 0.942001 i ), ( 1.38665 + 0.942001 i )}
                                     ( −2.48327 )
USpa[c] // TraditionalForm
% // N
UbarSpa[c] // TraditionalForm
% // N
u+(c)
{{1.75593}, {C.98C5C8 - C.666C95 i}, {1.75593}, {C.98C5C8 - C.666C95 i}}
u−(c)
{{C.98C5C8 - C.666C95 i, -1.75593, C.98C5C8 - C.666C95 i, -1.75593}}
```

Spinor products with or without inserted slashed matrices corresponding to spinor momenta as well as the invariants `s[a,...,b]` can be evaluated numerically.

```
Spaa[a, b] // N
Spbb[a, b] // N
Spab[1, 3, 2] // N
Spaa[1, 3, 4, 2] // N
s[a, e] // N

C.483C13 - C.427498 i

-C.483C13 - C.427498 i

1.43884 - 1.C5C22 i

-1.64152 - 2.15347 i

-C.9C9762
```

Once the numerical value of four vectors are defined, one can use them in spinor products with slashed matrices, vector products `MP[p,q]`, `MP2[p]` and invariants `s[p,q]`.

```
DeclareLVectorMomentum[p, {1, 0, 0, 0}]
DeclareLVectorMomentum[q, {1, 0, 0, -1}]
Four Momentum p set to {1, 0, 0, 0}.
{q} added to the list of Lorentz vectors
Four Momentum q set to {1, 0, 0, -1}.
Spab[a, p, b] // N
Spaa[a, p, q, b] // N
1.31278 - 0.154359 i
1.36241 - 0.412794 i
s[p, q] // N
MP[p, Sp[2]] // N
MP2[p + q - Sp[4]] // N
3.
0.725326
8.76299
```

PfrmSm2, PfromCSm2

The four vector can be extracted from the numerical two-dimensional representation of its slashed matrix using the function PfromSm2 or PfromCSm2, depending on whether the matrix is of the Sm2 or CSm2 type.

```
PfromSm2[Sm2[2] // N] // Chop
Num4V[2] // N
PfromSm2[λ̃(2).λ(1) // N] // Chop
PfromCSm2[CSmBA2[2, 1] // N] // Chop
{0.725326, -0.299499, -0.42836, -0.502897}
{0.725326, -0.299499, -0.42836, -0.502897}
{0.414319 - 0.235551 i, -0.432629 + 0.42751 i, -0.443834 - 0.176593 i, 0.0381312 + 0.235551 i}
{0.414319 - 0.235551 i, -0.432629 + 0.42751 i, -0.443834 - 0.176593 i, 0.0381312 + 0.235551 i}
```

PfrmSm4

The four vector can be extracted from the numerical four-dimensional representation of its slashed matrix using the function PfromSm4.

```
PfromSm4[Sm4[3] // N] // Chop
Num4V[3] // N
{0.797155, 0.778221, 0.128146, 0.115789}
{0.797155, 0.778221, 0.128146, 0.115789}
```

4 Simple Examples

In this section we will show three simple examples of the use of the S@M package. In the first example, we will re-derive the five gluon MHV amplitude $A^{\text{tree}}(1^-, 2^-, 3^+, 4^+, 5^+)$ using the BCFW construction [67], in the second, we compute a box integral coefficient numerically using the quadruple cut technique. The third example illustrates the evaluation of cut integrals using

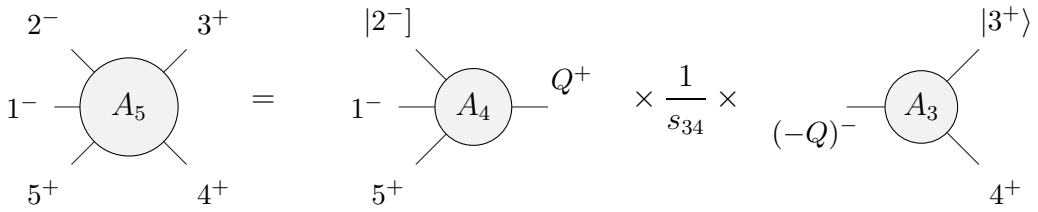

Fig. 1. BCFW construction of the five-gluon MHV amplitude using a $|23\rangle$ shift.

the spinor integration method. The examples of this section are distributed with the package.

4.1 BCFW Construction

In this example, we will re-derive the very well-known five-gluon MHV amplitude $A^{\text{tree}}(1^-, 2^-, 3^+, 4^+, 5^+)$ using the BCFW construction [67]. The Mathematica notebook file `S@M_BCFW.nb` containing this example is distributed with the package. We start by defining the MHV amplitudes.

```
MHV[a_, b_, c__] := i Spaa[a, b]^4 / Times @@ Spaa @@@ Partition[{a, b, c, a}, 2, 1]
MHVbar[a_, b_, c__] :=
  (-1)^Length[{a, b, c}] i Spbb[a, b]^4 / Times @@ Spbb @@@ Partition[{a, b, c, a}, 2, 1]
```

Since we know the answer we can use it to check the calculation

```
A0 = MHV[1, 2, 3, 4, 5]
                 i ⟨1 | 2⟩^3
-(-------------------------------------)
  ⟨1 | 5⟩ ⟨2 | 3⟩ ⟨3 | 4⟩ ⟨4 | 5⟩
```

We will use the numerical implementation to check the result, so we need to generate a set of five on-shell momenta which sum up to zero momentum.

```
GenMomenta[{1, 2, 3, 4, 5}]
Momenta for the spinors 1, 2, 3, 4, 5 generated.
```

We will use the shift

$$\lambda_2 \to \lambda_2, \quad \lambda_3 \to \lambda_3 + z\lambda_3, \quad \tilde{\lambda}_2 \to \tilde{\lambda}_2 - z\tilde{\lambda}_3, \quad \tilde{\lambda}_3 \to \tilde{\lambda}_3.$$

Now we are ready to start the computation. With the shift we chose, there is only one momentum partition contributing namely $\{5, 1, 2\}\{3, 4\}$, see Figure 1. The amplitude is then given by

```
DeclareSpinor[Q, mQ]
{Q, mQ} added to the list of spinors
                                  i
A = MHV[1, 2, Q, 5] ---- MHVbar[3, 4, mQ]
                                 s34
                  i ⟨1 | 2⟩^3 [4 | 3]^3
-------------------------------------------------
s34 ⟨Q | 2⟩ ⟨Q | 5⟩ ⟨1 | 5⟩ [3 | mQ] [4 | mQ]
```

First we need to find the value of z that puts the propagator

$$s_{34}(z) = Q(z)^2 = (p_3(z) + p_4(z))^2$$

on-shell.

```
zsol = Solve[ShiftBA[2, 3, z][s[3, 4]] == 0, z] // ExpandSToSpinors // SpOpen // Flatten
```

$$\left\{z \to -\frac{\langle 3 \mid 4\rangle}{\langle 2 \mid 4\rangle}\right\}$$

We can check numerically that the formula for the amplitude is right, but for that we need a numerical expression for the spinors $|-Q]$ and $|Q\rangle$ associated with the shifted momentum $Q = p_3(z) + p_4(z)$. This can be done with the functions `DeclareSpinorMomentum` and `PformSm2`

```
ShiftBA[2, 3, z][Sm2[3] + Sm2[4]]
Qs = % /. zsol
```

$$z\,\text{CLat}[3].\text{CLa}[2] + \text{Sm2}[3] + \text{Sm2}[4]$$

$$\text{Sm2}[3] + \text{Sm2}[4] - \frac{\text{CLat}[3].\text{CLa}[2]\,\langle 3 \mid 4\rangle}{\langle 2 \mid 4\rangle}$$

```
DeclareSpinorMomentum[Q, PfromSm2[Qs // N]]
```

```
Momentum for spinor Q set to
 {-0.357954 - 0.401137 i, -0.032426 + 0.513217 i, 0.241304 - 0.486258 i, -0.588384 - 0.446289 i}.
```

```
DeclareSpinorMomentum[mQ, -PfromSm2[Qs // N]]
```

```
Momentum for spinor mQ set to
 {0.357954 + 0.401137 i, 0.032426 - 0.513217 i, -0.241304 + 0.486258 i, 0.588384 + 0.446289 i}.
```

Now we can numerically evaluate the shifted amplitude and check that we get the right result.

```
ShiftBA[2, 3, z][A]
```

$$\frac{i\,\langle 1 \mid 2\rangle^3\,[4 \mid 3]^3}{s34\,\langle Q \mid 2\rangle\,\langle Q \mid 5\rangle\,\langle 1 \mid 5\rangle\,[3 \mid mQ]\,[4 \mid mQ]}$$

```
% /. s34 → s[3, 4] // N
```

$$-0.00573036 - 0.0112245\,i$$

```
N[A0]
```

$$-0.00573036 - 0.0112245\,i$$

Here we see that the numerical result matches with the known MHV result. We can also recover the analytic formula for the MHV amplitude. For that we first need to convert the spinors $|-Q\rangle$ into $|Q\rangle$ using

$$|-Q\rangle = \pm i\,|Q\rangle\,, \qquad |-Q] = \pm i\,|Q]$$

where the $\pm$ sign convention has no impact here since the spinor $|Q]$ appears twice.

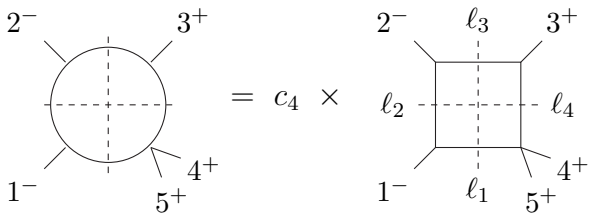

Fig. 2. Quadruple-cut for the determination of the coefficient of the box integral $I_4(0,0,0,s_{45})$ in the one-loop five-gluon amplitude, with a gluon propagating in the loop.

```
A /. mQ → i Q
```

$$-\frac{i\langle 1|2\rangle^3[4|3]^3}{s34\langle Q|2\rangle\langle Q|5\rangle\langle 1|5\rangle[3|Q][4|Q]}$$

```
SpClose[%, Q]
```

$$-\frac{i\langle 1|2\rangle^3[4|3]^3}{s34\langle 1|5\rangle\langle 2|Q|3]\langle 5|Q|4]}$$

```
ShiftBA[2, 3, z][% /. Q → Sm[3] + Sm[4]] /. zsol
```

$$-\frac{i\langle 1|2\rangle^3[4|3]^2}{s34\langle 1|5\rangle\left(-\frac{\langle 2|5\rangle\langle 3|4\rangle}{\langle 2|4\rangle}+\langle 3|5\rangle\right)\langle 2|4|3]}$$

```
% /. s34 → s[3, 4] // ExpandSToSpinors // SpOpen // Simplify
```

$$\frac{i\langle 1|2\rangle^3}{\langle 1|5\rangle\langle 3|4\rangle(\langle 2|5\rangle\langle 3|4\rangle-\langle 2|4\rangle\langle 3|5\rangle)}$$

We can make the result look more familiar using the Schouten identity

```
Schouten[%, 2, 5, 3, 4]
```

$$-\frac{i\langle 1|2\rangle^3}{\langle 1|5\rangle\langle 2|3\rangle\langle 3|4\rangle\langle 4|5\rangle}$$

we can now check that this is the expected known result.

```
MatchQ[%, MHV[1, 2, 3, 4, 5]]
True
```

4.2 A Box Coefficient

We will compute the coefficient of the one mass box $I_4(0,0,0,s_{45})$ of the one-loop five-gluon amplitude $A_5^{1-\text{loop}}(1^-,2^-,3^+,4^+,5^+)$ with a gluon propagating in the loop, see Figure 2. The Mathematica notebook file `S@M_Cut.nb` containing this example is distributed with the package. First we generate a momentum configuration with five momenta and define the loop momentum L.

```
SeedRandom[1111]
GenMomenta[{1, 2, 3, 4, 5}]
Momenta for the spinors 1, 2, 3, 4, 5 generated.
DeclareLVectorMomentum[L, {L0, L1, L2, L3}]
{L} added to the list of Lorentz vectors
Four Momentum L set to {L0, L1, L2, L3}.
```

Note that using the command `SeedRandom[...]` before `GenMomenta` allows us to reproduce the same numerical momentum configuration in another session, so that comparison of numerical results are easier.

Now we define the four propagator momenta `l1,l2,l3,l4`

```
l1 = L
l2 = L - Sp[1]
l3 = l2 - Sp[2]
l4 = l3 - Sp[3]
L
L - 1
L - 1 - 2
L - 1 - 2 - 3
```

and solve for the components of the loop momenta `L` using the on-shell constraints for the four propagators.

```
sols = Solve[{
    MP2[l1] == 0,
    MP2[l2] == 0,
    MP2[l3] == 0,
    MP2[l4] == 0
   } // N, {L0, L1, L2, L3}]
{{L0 → 1.12013 - 0.261024 i, L1 → 0.492206 - 1.6121 i,
  L2 → -1.84975 + 1.77393 i, L3 → -2.39887 - 1.57675 i}, {L0 → 1.12013 + 0.261024 i,
  L1 → 0.492206 + 1.6121 i, L2 → -1.84975 - 1.77393 i, L3 → -2.39887 + 1.57675 i}}
```

Using the first solution, we define the spinors corresponding to the propagator momenta (both incoming and outgoing).

```
DeclareSpinorMomentum[pl1, Num4V[l1] /. sols[[1]]]
DeclareSpinorMomentum[ml1, -Num4V[l1] /. sols[[1]]]
DeclareSpinorMomentum[pl2, Num4V[l2] /. sols[[1]]]
DeclareSpinorMomentum[ml2, -Num4V[l2] /. sols[[1]]]
DeclareSpinorMomentum[pl3, Num4V[l3] /. sols[[1]]]
DeclareSpinorMomentum[ml3, -Num4V[l3] /. sols[[1]]]
DeclareSpinorMomentum[pl4, Num4V[l4] /. sols[[1]]]
DeclareSpinorMomentum[ml4, -Num4V[l4] /. sols[[1]]]

{pl1} added to the list of spinors
Momentum for spinor pl1 set to
 {1.12013 - 0.261024 i, 0.492206 - 1.6121 i, -1.84975 + 1.77393 i, -2.39887 - 1.57675 i}.
{ml1} added to the list of spinors

...
```

The output has been shortened for readability. We can now compute the coefficient by inserting these spinors into the amplitudes at each corner of the box.

```
sol1h1 = MHV[ml1, pl4, 4, 5] MHV[1, ml2, pl1] MHVbar[ml3, pl2, 2] MHVbar[3, ml4, pl3] // N // Chop
0
sol1h2 = MHV[ml1, pl4, 4, 5] MHVbar[ml2, pl1, 1] MHV[pl2, 2, ml3] MHVbar[3, ml4, pl3] // N // Chop
-41.0671 + 62.7569 i
```

Where `solh1` and `solh2` are the two contributions corresponding to the two (not obviously vanishing) helicity configurations for the propagator momenta. The first helicity configuration gives a vanishing coefficient. The second solution for the loop momentum yields only vanishing solutions:

```
DeclareSpinorMomentum[pl1, Num4V[l1] /. sols[[2]]]
DeclareSpinorMomentum[ml1, -Num4V[l1] /. sols[[2]]]
DeclareSpinorMomentum[pl2, Num4V[l2] /. sols[[2]]]
DeclareSpinorMomentum[ml2, -Num4V[l2] /. sols[[2]]]
DeclareSpinorMomentum[pl3, Num4V[l3] /. sols[[2]]]
DeclareSpinorMomentum[ml3, -Num4V[l3] /. sols[[2]]]
DeclareSpinorMomentum[pl4, Num4V[l4] /. sols[[2]]]
DeclareSpinorMomentum[ml4, -Num4V[l4] /. sols[[2]]]

Momentum for spinor pl1 set to
 {1.12013 + 0.261024 i, 0.492206 + 1.6121 i, -1.84975 - 1.77393 i, -2.39887 + 1.57675 i}.
Momentum for spinor ml1 set to
 {-1.12013 - 0.261024 i, -0.492206 - 1.6121 i, 1.84975 + 1.77393 i, 2.39887 - 1.57675 i}.
Momentum for spinor pl2 set to
 {-0.28478 + 0.261024 i, -0.713906 + 1.6121 i, -2.16198 - 1.77393 i, -1.74957 + 1.57675 i}.

...
sol2h1 = MHV[ml1, pl4, 4, 5] MHV[1, ml2, pl1] MHVbar[ml3, pl2, 2] MHVbar[3, ml4, pl3] // N // Chop
0
sol2h2 = MHV[ml1, pl4, 4, 5] MHVbar[ml2, pl1, 1] MHV[pl2, 2, ml3] MHVbar[3, ml4, pl3] // N // Chop
0
```

We can verify that the result is as expected [87]

```
i s[1, 2] s[2, 3] / 2 MHV[1, 2, 3, 4, 5] // N
-20.5335 + 31.3785 i
(sol1h1 + sol1h2 + sol2h1 + sol2h2) / 2
-20.5335 + 31.3785 i
```

4.3 Spinor Integration

In the following example we discuss how `S@M` can be used to perform the evaluation of a double-cut by means of the spinor-integration method described in [42, 48]. The Mathematica notebook file `S@M_SpinorIntegration.nb` containing this example is distributed with the package. In particular, we consider the cut in the s_{34}-channel of the one-loop four-gluon amplitude $A_4^{\text{one-loop}}(1^-, 2^+, 3^-, 4^+)$ with a gluon running around the loop. The s_{34}-cut has two contributions, ac-

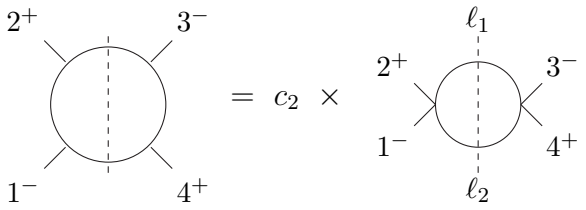

Fig. 3. Double-cut for the determination of the coefficient of the bubble integral $I_2(s_{34})$ in the one-loop 4-point gluon amplitude, with a gluon propagating in the loop.

cording to the choice of the internal helicity. We will only take one of the two possibilities into account, namely

$$C_1 = \int d\Phi \; A^{\rm tree}(1^-, 2^+, \ell_1^-, \ell_2^+) \times A^{\rm tree}(\ell_2^-, \ell_1^+, 3^-, 4^+) \; ,$$

where dΦ is the standard Lorentz invariant two-body phase-space, see Figure 3 We begin with the definition of the tree-level amplitude, and of the cut-integral

```
GenMomenta[{1, 2, 3, 4, 5, 6}]
Momenta for the spinors 1, 2, 3, 4, 5, 6 generated.
DeclareSpinor[l1, l2, λ]
{l1, l2, λ} added to the list of spinors
DeclareSpinorMomentum[λ, {Sqrt[3], 1, 1, 1}]
Momentum for spinor λ set to {√3, 1, 1, 1}.
DeclareLVectorMomentum[P34, Num4V[3] + Num4V[4]]
{P34} added to the list of Lorentz vectors
Four Momentum P34 set to
 {3.8171160357 4C39234, 1.27297206351920934, -2.4C1580264C9676221, C.394768C6691977571}.
mpmp[x1_, x2_, x3_, x4_] :=
  Spaa[x1, x3]^4 / (Spaa[x1, x2] * Spaa[x2, x3] * Spaa[x3, x4] * Spaa[x4, x1]);
```

By using the momentum conservation $\ell_1 - \ell_2 = k_3 + k_4 \equiv P_{34}$, one can eliminate the dependence on ℓ_2, and write the integrand just in terms of ℓ_1

```
Cut[1] = dΦ * mpmp[1, 2, l1, l2] * mpmp[l2, l1, 3, 4]
                  dΦ ⟨l1 | 1⟩^4 ⟨l2 | 3⟩^4
- ------------------------------------------------------
  ⟨l1 | l2⟩^2 ⟨l1 | 2⟩ ⟨l1 | 3⟩ ⟨l2 | 1⟩ ⟨l2 | 4⟩ ⟨1 | 2⟩ ⟨3 | 4⟩
Cut[1] = Cut[1] /. {
   Spaa[x_, l2] → Spab[x, P34, l1] / Spbb[l2, l1],
   Spaa[l2, x_] → Spba[l1, P34, x] / Spbb[l1, l2]
  }
-(dΦ ⟨l1 | 1⟩^4 ⟨3 | P34 | l1]^4) /
   (⟨l1 | 2⟩ ⟨l1 | 3⟩ ⟨1 | 2⟩ ⟨3 | 4⟩ ⟨l1 | P34 | l1]^2 ⟨1 | P34 | l1] ⟨4 | P34 | l1])
```

Then we redefine the loop spinor variables according to the CSW [81] prescription:

$$|\ell_1\rangle = \sqrt{t}|\lambda\rangle \; , \tag{38}$$

$$|\ell_1] = \sqrt{t}|\lambda] \; , \tag{39}$$

$$\int d\Phi = \int d\lambda \int \frac{t\ dt}{\langle\lambda|P_{34}|\lambda]}\ \delta\left(t - \frac{s_{34}}{\langle\lambda|P_{34}|\lambda]}\right) \tag{40}$$

$$\int d\lambda = \int\limits_{|\lambda]=|\lambda\rangle^*} \langle\lambda\ d\lambda\rangle[\lambda\ d\lambda]\ , \tag{41}$$

```
Cut[1] = ASpinorReplace[Cut[1], l1, Sqrt[t] * λ]
```

$$-\left(d\Phi\,\langle\lambda\mid 1\rangle^4\,\langle 3\mid P34\mid l1]^4\right)\Big/\left(\langle\lambda\mid 2\rangle\,\langle\lambda\mid 3\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid l1]^2\,\langle 1\mid P34\mid l1]\,\langle 4\mid P34\mid l1]\right)$$

```
Cut[1] = BSpinorReplace[Cut[1], l1, Sqrt[t] * λ]
```

$$-\left(d\Phi\,\langle\lambda\mid 1\rangle^4\,\langle 3\mid P34\mid \lambda]^4\right)\Big/\left(\langle\lambda\mid 2\rangle\,\langle\lambda\mid 3\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid \lambda]^2\,\langle 1\mid P34\mid \lambda]\,\langle 4\mid P34\mid \lambda]\right)$$

We can perform the t-integration by substituting the value of t as imposed by the δ-function ,

```
Cut[1] = Cut[1] //. dΦ → dλ * t / Spab[λ, P34, λ]
```

$$-\left(d\lambda\,t\,\langle\lambda\mid 1\rangle^4\,\langle 3\mid P34\mid \lambda]^4\right)\Big/\left(\langle\lambda\mid 2\rangle\,\langle\lambda\mid 3\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid \lambda]^3\,\langle 1\mid P34\mid \lambda]\,\langle 4\mid P34\mid \lambda]\right)$$

```
Cut[1] = Cut[1] //. {t → s[3, 4] / Spab[λ, P34, λ]}
```

$$-\left(d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^4\,\langle 3\mid P34\mid \lambda]^4\right)\Big/\left(\langle\lambda\mid 2\rangle\,\langle\lambda\mid 3\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid \lambda]^4\,\langle 1\mid P34\mid \lambda]\,\langle 4\mid P34\mid \lambda]\right)$$

Now we choose to integrate by parts in $|\lambda]$ and to extract residue in $|\lambda\rangle$. To that aim we have to cast the integrand in a suitable form. We perform the reduction on the $|\lambda]$-variable by means of the Schouten identities,

```
Cut[1] = BSchouten[Cut[1], λ] // Expand;
Cut[1] = Cut[1] //. (sp : (Spaa | Spab | Spbb))[x__, P34, P34, y__] → s[3, 4] * sp[x, y]
```

$$\frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^2\,\langle\lambda\mid 3\rangle\,\langle 1\mid 3\rangle}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle^2\,\langle 1\mid 2\rangle\,\langle\lambda\mid P34\mid\lambda]^2} - \frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle\,\langle\lambda\mid 3\rangle\,\langle 1\mid 3\rangle^2}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid\lambda]^2} -$$

$$\frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^3\,\langle\lambda\mid 3\rangle\,\langle 3\mid 4\rangle}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle^3\,\langle 1\mid 2\rangle\,\langle\lambda\mid P34\mid\lambda]^2} - \frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^3\,\langle\lambda\mid 3\rangle\,\langle 3\mid P34\mid\lambda]}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle^2\,\langle 1\mid 2\rangle\,\langle\lambda\mid P34\mid\lambda]^3} +$$

$$\frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^2\,\langle\lambda\mid 3\rangle\,\langle 1\mid 3\rangle\,\langle 3\mid P34\mid\lambda]}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid\lambda]^3} - \frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^3\,\langle\lambda\mid 3\rangle\,\langle 3\mid P34\mid\lambda]^2}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid\lambda]^4} -$$

$$\frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle\,\langle 1\mid 3\rangle^2}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle\,\langle 1\mid 2\rangle\,\langle\lambda\mid P34\mid\lambda]\,\langle 4\mid P34\mid\lambda]} + \frac{d\lambda\,s_{3\times 4}\,\langle 1\mid 3\rangle^3}{\langle\lambda\mid 2\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle\lambda\mid P34\mid\lambda]\,\langle 4\mid P34\mid\lambda]} +$$

$$\frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^2\,\langle 1\mid 3\rangle\,\langle 3\mid 4\rangle}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle^2\,\langle 1\mid 2\rangle\,\langle\lambda\mid P34\mid\lambda]\,\langle 4\mid P34\mid\lambda]} - \frac{d\lambda\,s_{3\times 4}\,\langle\lambda\mid 1\rangle^3\,\langle 3\mid 4\rangle^2}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 4\rangle^3\,\langle 1\mid 2\rangle\,\langle\lambda\mid P34\mid\lambda]\,\langle 4\mid P34\mid\lambda]} -$$

$$\frac{d\lambda\,s_{3\times 4}\,\langle 1\mid 3\rangle^4}{\langle\lambda\mid 2\rangle\,\langle\lambda\mid 3\rangle\,\langle 1\mid 2\rangle\,\langle 3\mid 4\rangle\,\langle 1\mid P34\mid\lambda]\,\langle 4\mid P34\mid\lambda]}$$

```
Cut[1, 4] = Coefficient[Cut[1], 1 / Spab[λ, P34, λ]^4] * 1 / Spab[λ, P34, λ]^4 // Expand
```

The integrand is thus expressed as sum of four terms according to their $|\lambda]$ dependence.

$$\frac{[\bullet\,\lambda]^n}{\langle\lambda|P_{34}|\lambda]^{n+2}} \qquad n = 0, 1, 2 \quad \text{or} \quad \frac{1}{\langle\lambda|P_{34}|\lambda]\langle\bullet|P_{34}|\lambda]}$$

$$-\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle^3\langle\lambda|3\rangle\langle3|P34|\lambda]^2}{\langle\lambda|2\rangle\langle\lambda|4\rangle\langle1|2\rangle\langle3|4\rangle\langle\lambda|P34|\lambda]^4}$$

```
Cut[1, 3] = Coefficient[Cut[1], 1/Spab[λ, P34, λ]^3] * 1/ Spab[λ, P34, λ]^3 // Expand
```

$$-\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle^3\langle\lambda|3\rangle\langle3|P34|\lambda]}{\langle\lambda|2\rangle\langle\lambda|4\rangle^2\langle1|2\rangle\langle\lambda|P34|\lambda]^3}-\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle^2\langle\lambda|3\rangle\langle1|3\rangle\langle3|P34|\lambda]}{\langle\lambda|2\rangle\langle\lambda|4\rangle\langle1|2\rangle\langle3|4\rangle\langle\lambda|P34|\lambda]^3}$$

```
Cut[1, 2] = Coefficient[Cut[1], 1/Spab[λ, P34, λ]^2] * 1/ Spab[λ, P34, λ]^2 // Expand
```

$$\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle^2\langle\lambda|3\rangle\langle1|3\rangle}{\langle\lambda|2\rangle\langle\lambda|4\rangle^2\langle1|2\rangle\langle\lambda|P34|\lambda]^2}$$

$$\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle\langle\lambda|3\rangle\langle1|3\rangle^2}{\langle\lambda|2\rangle\langle\lambda|4\rangle\langle1|2\rangle\langle3|4\rangle\langle\lambda|P34|\lambda]^2}\quad\frac{d\lambda\, s_{2\times4}\,\langle\lambda|1\rangle^3\langle\lambda|3\rangle\langle3|4\rangle}{\langle\lambda|2\rangle\langle\lambda|4\rangle^3\langle1|2\rangle\langle\lambda|P34|\lambda]^2}$$

```
Cut[1, 1] = Coefficient[Cut[1], 1 / Spab[λ, P34, λ]] * 1 / Spab[λ, P34, λ] // Expand
```

$$-\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle\langle1|3\rangle^2}{\langle\lambda|2\rangle\langle\lambda|4\rangle\langle1|2\rangle\langle\lambda|P34|\lambda]\langle4|P34|\lambda]}+\frac{d\lambda\, s_{3\times4}\,\langle1|3\rangle^3}{\langle\lambda|2\rangle\langle1|2\rangle\langle3|4\rangle\langle\lambda|P34|\lambda]\langle4|P34|\lambda]}-$$
$$\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle^2\langle1|3\rangle\langle3|4\rangle}{\langle\lambda|2\rangle\langle\lambda|4\rangle^2\langle1|2\rangle\langle\lambda|P34|\lambda]\langle4|P34|\lambda]}-\frac{d\lambda\, s_{3\times4}\,\langle\lambda|1\rangle^3\langle3|4\rangle^2}{\langle\lambda|2\rangle\langle\lambda|4\rangle^3\langle1|2\rangle\langle\lambda|P34|\lambda]\langle4|P34|\lambda]}$$

```
Trm4[1] = Expand[Cut[1, 4] //. dλ → dea * deb]
```

For brevity, we discuss only the integration of the term defined as `Cut[1,4]`. To keep track of the integration we use the explicit definition of $d\lambda$ given above, $d\lambda = \langle\lambda\; d\lambda\rangle[\lambda\; d\lambda] =$ `dea * deb` . After simplifying the integrand with trivial spinor identities, we integrate by parts in $|\lambda]$, using the identity

$$[\lambda\; d\lambda]\frac{[\eta\,\lambda]^n}{\langle\lambda|P_{34}|\lambda]^{n+2}} = \frac{[d\lambda\;\partial_{|\lambda]}]}{(n+1)}\frac{[\eta\,\lambda]^{n+1}}{\langle\lambda|P_{34}|\lambda]^{n+1}\langle\lambda|P_{34}|\eta]} \tag{42}$$

for $n = 2$ and $|\eta] = |4]$,

$$-\frac{\text{dea deb}\; s_{3\times4}\,\langle\lambda|1\rangle^3\langle\lambda|3\rangle\langle3|P34|\lambda]^2}{\langle\lambda|2\rangle\langle\lambda|4\rangle\langle1|2\rangle\langle3|4\rangle\langle\lambda|P34|\lambda]^4}$$

```
IdySet = {
   Spab[4, P34, x_] → Spaa[4, 3] * Spbb[3, x],
   Spab[3, P34, x_] → Spaa[3, 4] * Spbb[4, x],
   Spab[x_, P34, 4] → Spaa[x, 3] * Spbb[3, 4],
   Spab[x_, P34, 3] → Spaa[x, 4] * Spbb[4, 3]
  };
Trm4[1] = Trm4[1] //. IdySet
```

$$-\frac{\text{dea deb}\; s_{3\times4}\,\langle\lambda|1\rangle^3\langle\lambda|3\rangle\langle3|4\rangle[4|\lambda]^2}{\langle\lambda|2\rangle\langle\lambda|4\rangle\langle1|2\rangle\langle\lambda|P34|\lambda]^4}$$

```
IBPidy = deb Spbb[4, λ]^2 / Spab[λ, P34, λ]^4 → Spbb[4, λ]^3 / (3 Spab[λ, P34, 4] Spab[λ, P34, λ]^3)
```

$$\frac{\text{deb}\,[4|\lambda]^2}{\langle\lambda|P34|\lambda]^4}\to\frac{[4|\lambda]^3}{3\,\langle\lambda|P34|\lambda]^3\langle\lambda|P34|4]}$$

```
Trm4[1] = Trm4[1] //. IBPidy
```

$$\frac{\text{dea}\; s_{3\times4}\,\langle\lambda|1\rangle^3\langle\lambda|3\rangle\langle3|4\rangle[4|\lambda]^3}{3\,\langle\lambda|2\rangle\langle\lambda|4\rangle\langle1|2\rangle\langle\lambda|P34|\lambda]^3\langle\lambda|P34|4]}$$

The final integration over $|\lambda\rangle$ can be performed by summing over the residues at the simple poles in $|\lambda\rangle$. The expression of `Trm4[1]` has two simple poles

at $|\lambda\rangle = |2\rangle, |4\rangle$. We notice that the residue at $|\lambda\rangle = |4\rangle$ vanishes, due to the term $[4\,\lambda]^3$ in the numerator. Therefore the result is given just by the residue at $|\lambda\rangle = |2\rangle$,

```
Trm4[1] = Trm4[1] //. IdySet
```

$$\frac{\text{dea}\; s_{3\times4} \langle\lambda\,|\,1\rangle^3 \langle 3\,|\,4\rangle\, [4\,|\,\lambda]^3}{3\,\langle\lambda\,|\,2\rangle\, \langle\lambda\,|\,4\rangle\, \langle 1\,|\,2\rangle\, \langle\lambda\,|\,\text{P34}\,|\,\lambda]^3\, [4\,|\,3]}$$

Two comments are in order. In this simple case, after the $|\lambda]$-integration, the integrand contained only simple poles in $|\lambda\rangle$. In general it may well happen that higher poles are present. Should this be the case, one can apply the function `ASchouten` in order to single out the simple poles beneath the higher ones, and then take the residue in $|\lambda\rangle$.
The integration of `Cut[1,3]` and `Cut[1,2]` proceeds along the lines just outlined. The result, as for `Cut[1,4]`, will be a combination of rational functions of spinor products. Therefore they all contribute to the coefficient of the $\ln(s_{34})$ of the whole amplitude.
The integration of `Cut[1,1]` is a bit different. In order to apply the integration by-parts in $|\lambda]$, one has to introduce a Feynman parameters. Then, the integration over the spinor variables can be carried on as outlined above. Finally, one performs the integration over the Feynman parameter. The last parametric integration is responsible for the rising of the logarithmic terms of the double-cut, whose coefficient can be directly assigned to the (poly-)logarithms of the whole amplitude.

5 Conclusion and Outlook

We have presented the Mathematica package `S@M` whose aim is to provide its user with a tool for performing the basic spinor algebra, plus the spinor-shifts needed for a very efficient analytic evaluation of scattering amplitudes at tree- and loop-level, accompanied by the support of the numerical evaluation at every computational stage.
The basic properties of the functions introduced within `S@M` render it a very flexible program which could be further enriched with additional routines designed for more specific tasks.

Acknowledgments

We wish to thank Zvi Bern, John Conley, Darren Forde, Thomas Gehrmann, Harald Ita, David Kosower, My Phuong Le and Tommer Wizanski for useful comments on the manuscript and for experimenting with un-mature versions

of the package. We thank the Galileo Galilei Institute for Theoretical Physics for the hospitality and the INFN for partial support during the completion of this work. The work of D. M. was supported by the Swiss National Science Foundation (SNF) under contracts 200020-109162 and PBZH2-117028 and by the US Department of Energy under contract DE-AC02-76SF00515. The work of P.M. was supported by the Marie-Curie-EIF under the contract MEIF-CT-2006-0214178.

A Functions Index

The following table lists the functions provided by the package and the page where they are described.

A list of all the functions is stored in the variable `$SpinorFunctions`.

References

[1] T. Stelzer and W. F. Long, Comput. Phys. Commun. **81**, 357 (1994) [hep-ph/9401258].

[2] A. Pukhov *et al.*, hep-ph/9908288.

[3] F. Krauss, R. Kuhn and G. Soff, JHEP **0202**, 044 (2002) [hep-ph/0109036].

[4] M. L. Mangano, M. Moretti, F. Piccinini, R. Pittau and A. D. Polosa, JHEP **0307**, 001 (2003) [hep-ph/0206293].

[5] A. Kanaki and C. G. Papadopoulos, Comput. Phys. Commun. **132**, 306 (2000) [hep-ph/0002082].

[6] F. A. Berends and W. T. Giele, Nucl. Phys. B **306**, 759 (1988).

[7] F. A. Berends, W. T. Giele and H. Kuijf, Nucl. Phys. B **321**, 39 (1989).

[8] D. A. Kosower, Nucl. Phys. B **335**, 23 (1990);
F. Caravaglios and M. Moretti, Phys. Lett. B **358**, 332 (1995) [hep-ph/9507237];
P. Draggiotis, R. H. P. Kleiss and C. G. Papadopoulos, Phys. Lett. B **439**, 157 (1998) [hep-ph/9807207];
F. Caravaglios, M. L. Mangano, M. Moretti and R. Pittau, Nucl. Phys. B **539**, 215 (1999) [hep-ph/9807570].

[9] C. Buttar *et al.*, arXiv:hep-ph/0604120.

[10] W. Beenakker, S. Dittmaier, M. Krämer, B. Plümper, M. Spira and P. M. Zerwas, Phys. Rev. Lett. **87**, 201805 (2001) [hep-ph/0107081]; Nucl. Phys. B **653**, 151 (2003) [hep-ph/0211352];
S. Dawson, L. H. Orr, L. Reina and D. Wackeroth, Phys. Rev. D **67**, 071503 (2003) [hep-ph/0211438];
S. Dawson, C. Jackson, L. H. Orr, L. Reina and D. Wackeroth, Phys. Rev. D **68**, 034022 (2003) [hep-ph/0305087].

[11] W. B. Kilgore and W. T. Giele, Phys. Rev. D **55**, 7183 (1997) [hep-ph/9610433].
Z. Nagy, Phys. Rev. Lett. **88**, 122003 (2002) [hep-ph/0110315]; Phys. Rev. D **68**, 094002 (2003) [hep-ph/0307268].

[12] J. Campbell and R. K. Ellis, Phys. Rev. D **65**, 113007 (2002) [hep-ph/0202176];
J. Campbell, R. K. Ellis and D. L. Rainwater, Phys. Rev. D **68**, 094021 (2003) [hep-ph/0308195].

[13] S. Dawson, C. B. Jackson, L. Reina and D. Wackeroth, Mod. Phys. Lett. A **21**, 89 (2006) [hep-ph/0508293];
F. Febres Cordero, L. Reina and D. Wackeroth, Phys. Rev. D **74**, 034007 (2006) [hep-ph/0606102].

[14] J. M. Campbell, R. K. Ellis and G. Zanderighi, JHEP **0610**, 028 (2006) [hep-ph/0608194].

[15] A. Brandenburg, S. Dittmaier, P. Uwer and S. Weinzierl, Nucl. Phys. Proc. Suppl. **135**, 71 (2004) [hep-ph/0408137];
S. Dittmaier, P. Uwer and S. Weinzierl, hep-ph/0703120.

[16] A. Lazopoulos, K. Melnikov and F. Petriello, hep-ph/0703273.

[17] A. Lazopoulos, K. Melnikov and F. J. Petriello, arXiv:0709.4044 [hep-ph].

[18] S. Dittmaier, S. Kallweit and P. Uwer, arXiv:0710.1577 [hep-ph].

[19] J. M. Campbell, R. K. Ellis and G. Zanderighi, arXiv:0710.1832 [hep-ph].

[20] A. I. Davydychev, Phys. Lett. B **263**, 107 (1991).

[21] A. Ferroglia, M. Passera, G. Passarino and S. Uccirati, Nucl. Phys. B **650**, 162 (2003) [hep-ph/0209219].

[22] Y. Kurihara *et al.*, Nucl. Phys. B **654**, 301 (2003) [hep-ph/0212216]; Nucl. Phys. Proc. Suppl. **157**, 231 (2006).

[23] A. Denner and S. Dittmaier, Nucl. Phys. B **658**, 175 (2003) [hep-ph/0212259]; Nucl. Phys. B **734**, 62 (2006) [hep-ph/0509141].

[24] W. T. Giele and E. W. N. Glover, JHEP **0404**, 029 (2004) [hep-ph/0402152].

[25] F. del Aguila and R. Pittau, JHEP **0407**, 017 (2004) [arXiv:hep-ph/0404120]; R. Pittau, arXiv:hep-ph/0406105.

[26] R. K. Ellis, W. T. Giele and G. Zanderighi, Phys. Rev. D **72**, 054018 (2005) [Erratum-ibid. D **74**, 079902 (2006)] [hep-ph/0506196].

[27] T. Binoth, J. P. Guillet, G. Heinrich, E. Pilon and C. Schubert, JHEP **0510**, 015 (2005) [hep-ph/0504267].

[28] R. K. Ellis, W. T. Giele and G. Zanderighi, Phys. Rev. D **73**, 014027 (2006) [hep-ph/0508308].

[29] R. K. Ellis, W. T. Giele and G. Zanderighi, JHEP **0605**, 027 (2006) [hep-ph/0602185].

[30] Z. Xiao, G. Yang and C. J. Zhu, Nucl. Phys. B **758**, 53 (2006) [hep-ph/0607017].

[31] T. Binoth, J. P. Guillet and G. Heinrich, JHEP **0702**, 013 (2007) [hep-ph/0609054].

[32] G. Ossola, C. G. Papadopoulos and R. Pittau, Nucl. Phys. B **763**, 147 (2007) hep-ph/0609007].

[33] R. K. Ellis, W. T. Giele and Z. Kunszt, arXiv:0708.2398 [hep-ph].

[34] M. Krämer and D. E. Soper, Phys. Rev. D **66**, 054017 (2002) [hep-ph/0204113]; Z. Nagy and D. E. Soper, JHEP **0309**, 055 (2003) [hep-ph/0308127].

[35] Z. Bern, L. J. Dixon, D. C. Dunbar and D. A. Kosower, Nucl. Phys. B **425**, 217 (1994) [hep-ph/9403226].

[36] Z. Bern, L. J. Dixon, D. C. Dunbar and D. A. Kosower, Nucl. Phys. B **435**, 59 (1995) [hep-ph/9409265].

[37] Z. Bern and A. G. Morgan, Nucl. Phys. B **467**, 479 (1996) [hep-ph/9511336]; Z. Bern, L. J. Dixon, D. C. Dunbar and D. A. Kosower, Phys. Lett. B **394**, 105 (1997) [hep-th/9611127].

[38] Z. Bern, L. J. Dixon and D. A. Kosower, Nucl. Phys. Proc. Suppl. **51C**, 243 (1996) [hep-ph/9606378]; JHEP **0001**, 027 (2000) [hep-ph/0001001]; Z. Bern, A. De Freitas and L. J. Dixon, JHEP **0203**, 018 (2002) [hep-ph/0201161].

[39] S. J. Bidder, N. E. J. Bjerrum-Bohr, L. J. Dixon and D. C. Dunbar, Phys. Lett. B **606**, 189 (2005) [hep-th/0410296].

[40] R. Britto, F. Cachazo and B. Feng, Nucl. Phys. B **725**, 275 (2005) [hep-th/0412103].

[41] S. J. Bidder, N. E. J. Bjerrum-Bohr, D. C. Dunbar and W. B. Perkins, Phys. Lett. B **612**, 75 (2005) [hep-th/0502028].

[42] R. Britto, E. Buchbinder, F. Cachazo and B. Feng, Phys. Rev. D **72**, 065012 (2005) [hep-ph/0503132].

[43] Z. Bern, L. J. Dixon and D. A. Kosower, Phys. Rev. D **71**, 105013 (2005) [hep-th/0501240].

[44] Z. Bern, L. J. Dixon and D. A. Kosower, Phys. Rev. D **72**, 125003 (2005) [hep-ph/0505055].

[45] Z. Bern, L. J. Dixon and D. A. Kosower, Phys. Rev. D **73**, 065013 (2006) [hep-ph/0507005]; D. Forde and D. A. Kosower, Phys. Rev. D **73**, 061701 (2006) [hep-ph/0509358].

[46] C. F. Berger, Z. Bern, L. J. Dixon, D. Forde and D. A. Kosower, Phys. Rev. D **74**, 036009 (2006) [hep-ph/0604195].

[47] C. F. Berger, Z. Bern, L. J. Dixon, D. Forde and D. A. Kosower, Phys. Rev. D **75**, 016006 (2007) [hep-ph/0607014].

[48] R. Britto, B. Feng and P. Mastrolia, Phys. Rev. D **73**, 105004 (2006) [hep-ph/0602178].

[49] C. Anastasiou, R. Britto, B. Feng, Z. Kunszt and P. Mastrolia, Phys. Lett. B **645**, 213 (2007) [hep-ph/0609191]; JHEP **0703**, 111 (2007) [hep-ph/0612277].

[50] P. Mastrolia, Phys. Lett. B **644**, 272 (2007) [hep-th/0611091].

[51] R. Britto and B. Feng, hep-ph/0612089.

[52] F. A. Berends, R. Kleiss, P. De Causmaecker, R. Gastmans and T. T. Wu, Phys. Lett. B **103**, 124 (1981);
M. Caffo and E. Remiddi, Helv. Phys. Acta **55** (1982) 339;
P. De Causmaecker, R. Gastmans, W. Troost and T. T. Wu, Nucl. Phys. B **206**, 53 (1982);
Z. Xu, D. H. Zhang and L. Chang, TUTP-84/3-TSINGHUA;
R. Kleiss and W. J. Stirling, Nucl. Phys. B **262**, 235 (1985);
J. F. Gunion and Z. Kunszt, Phys. Lett. B **161**, 333 (1985);
Z. Xu, D. H. Zhang and L. Chang, Nucl. Phys. B **291**, 392 (1987).

[53] Z. Bern, L. J. Dixon and D. A. Kosower, Annals Phys. **322**, 1587 (2007) [arXiv:0704.2798 [hep-ph]].

[54] Z. Bern and G. Chalmers, Nucl. Phys. B **447**, 465 (1995) [hep-ph/9503236].

[55] Z. Bern, V. Del Duca and C. R. Schmidt, Phys. Lett. B **445**, 168 (1998) [hep-ph/9810409];
Z. Bern, V. Del Duca, W. B. Kilgore and C. R. Schmidt, Phys. Rev. D **60**, 116001 (1999) [hep-ph/9903516];
D. A. Kosower and P. Uwer, Nucl. Phys. B **563**, 477 (1999) [hep-ph/9903515];
S. D. Badger and E. W. N. Glover, JHEP **0407**, 040 (2004) [hep-ph/0405236].

[56] S. J. Parke and T. R. Taylor, Phys. Rev. Lett. **56**, 2459 (1986);
M. L. Mangano and S. J. Parke, Phys. Rept. **200**, 301 (1991).

[57] L. J. Dixon, in *QCD & Beyond: Proceedings of TASI '95*, ed. D. E. Soper (World Scientific, 1996) [hep-ph/9601359].

[58] Z. Bern, L. J. Dixon and D. A. Kosower, Ann. Rev. Nucl. Part. Sci. **46**, 109 (1996) [hep-ph/9602280].

[59] Z. Bern, L. J. Dixon and D. A. Kosower, Nucl. Phys. B **513**, 3 (1998) [hep-ph/9708239].

[60] T. Binoth, T. Gehrmann, G. Heinrich and P. Mastrolia, hep-ph/0703311.

[61] Z. Nagy and D. E. Soper, Phys. Rev. D **74**, 093006 (2006) [hep-ph/0610028].

[62] G. Ossola, C. G. Papadopoulos and R. Pittau, 0704.1271 [hep-ph].

[63] E. Witten, Commun. Math. Phys. **252**, 189 (2004) [hep-th/0312171].

[64] R. Penrose, J. Math. Phys. **8**, 345 (1967).

[65] F. Cachazo and P. Svrček, PoS **RTN2005**, 004 (2005) [hep-th/0504194].

[66] L. J. Dixon, PoS **HEP2005**, 405 (2006) [hep-ph/0512111].

[67] R. Britto, F. Cachazo and B. Feng, Nucl. Phys. B **715**, 499 (2005) [hep-th/0412308].
R. Britto, F. Cachazo, B. Feng and E. Witten, Phys. Rev. Lett. **94**, 181602 (2005) [hep-th/0501052].

[68] S. D. Badger, E. W. N. Glover, V. V. Khoze and P. Svrček, JHEP **0507**, 025 (2005) [hep-th/0504159].

[69] S. D. Badger, E. W. N. Glover and V. V. Khoze, JHEP **0601**, 066 (2006) [hep-th/0507161];
D. Forde and D. A. Kosower, Phys. Rev. D **73**, 065007 (2006) [hep-th/0507292];
K. J. Ozeren and W. J. Stirling, Eur. Phys. J. C **48**, 159 (2006) [hep-ph/0603071];
C. F. Berger, V. Del Duca and L. J. Dixon, Phys. Rev. D **74**, 094021 (2006) [hep-ph/0608180].

[70] C. Schwinn and S. Weinzierl, JHEP **0704**, 072 (2007) [hep-ph/0703021].

[71] A. Hall, arXiv:0710.1300 [hep-ph].

[72] Z. Bern, L. J. Dixon and D. A. Kosower, Phys. Lett. B **302**, 299 (1993) [Erratum-ibid. B **318**, 649 (1993)] [hep-ph/9212308];
Z. Bern, L. J. Dixon and D. A. Kosower, Nucl. Phys. B **412**, 751 (1994) [hep-ph/9306240].

[73] J. Fleischer, F. Jegerlehner and O. V. Tarasov, Nucl. Phys. B **566**, 423 (2000) [hep-ph/9907327];
T. Binoth, J. P. Guillet and G. Heinrich, Nucl. Phys. B **572**, 361 (2000) [hep-ph/9911342];
G. Duplančić and B. Nižić, Eur. Phys. J. C **35**, 105 (2004) [hep-ph/0303184].

[74] L. D. Landau, Nucl. Phys. **13**, 181 (1959);
S. Mandelstam, Phys. Rev. **112**, 1344 (1958); Phys. Rev. **115**, 1741 (1959);
R. E. Cutkosky, J. Math. Phys. **1**, 429 (1960);
R. J. Eden, P. V. Landshoff, D. I. Olive, J. C. Polkinghorne, *The Analytic S Matrix* (Cambridge University Press, 1966).

[75] D. Forde, 0704.1835 [hep-ph].

[76] R. Pittau, Comput. Phys. Commun. **104**, 23 (1997) [hep-ph/9607309]; Comput. Phys. Commun. **111**, 48 (1998) [hep-ph/9712418].

[77] A. Brandhuber, B. Spence, G. Travaglini and K. Zoubos, 0704.0245 [hep-th].

[78] W. L. van Neerven, Nucl. Phys. B **268**, 453 (1986).

[79] A. Brandhuber, S. McNamara, B. J. Spence and G. Travaglini, JHEP **0510**, 011 (2005) [hep-th/0506068].

[80] Z. Bern, N. E. J. Bjerrum-Bohr, D. C. Dunbar and H. Ita, JHEP **0511**, 027 (2005) [hep-ph/0507019].

[81] F. Cachazo, P. Svrček and E. Witten, JHEP **0409**, 006 (2004) [hep-th/0403047].

[82] K. Risager, JHEP **0512**, 003 (2005) [hep-th/0508206].

[83] P. Benincasa and F. Cachazo, arXiv:0705.4305 [hep-th].

[84] J. Bedford, A. Brandhuber, B. J. Spence and G. Travaglini, Nucl. Phys. B **721**, 98 (2005) [arXiv:hep-th/0502146];

F. Cachazo and P. Svrcek, arXiv:hep-th/0502160;

N. E. J. Bjerrum-Bohr, D. C. Dunbar, H. Ita, W. B. Perkins and K. Risager, JHEP **0601**, 009 (2006) [arXiv:hep-th/0509016];

A. Brandhuber, S. McNamara, B. Spence and G. Travaglini, JHEP **0703**, 029 (2007) [arXiv:hep-th/0701187];

P. Benincasa, C. Boucher-Veronneau and F. Cachazo, arXiv:hep-th/0702032.

[85] Z. Bern, J. J. Carrasco, D. Forde, H. Ita and H. Johansson, arXiv:0707.1035 [hep-th];

Z. Bern, L. J. Dixon and R. Roiban, Phys. Lett. B **644**, 265 (2007) [arXiv:hep-th/0611086].

[86] D. A. Kosower, Phys. Rev. D **71**, 045007 (2005) [arXiv:hep-th/0406175].

[87] Z. Bern, L. J. Dixon and D. A. Kosower, Phys. Rev. Lett. **70**, 2677 (1993) [arXiv:hep-ph/9302280];

S. J. Bidder, N. E. J. Bjerrum-Bohr, D. C. Dunbar and W. B. Perkins, Phys. Lett. B **612**, 75 (2005) [arXiv:hep-th/0502028];

J. Bedford, A. Brandhuber, B. J. Spence and G. Travaglini, Nucl. Phys. B **712**, 59 (2005) [arXiv:hep-th/0412108].