GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Fakultät für
Physik

Bachelor's Thesis

Positivitätsanalyse einer nichttrivialen 4-Punkt-Korrelationsfunktion in 4D konformer Quantenfeldtheorie

Positivity analysis of a non trivial 4-point correlation function in 4D conformal quantum field theory

prepared by

Nikolai Wyderka

from Göttingen

at the Institut für Theoretische Physik

Thesis period: 29th April 2013 until 5th August 2013

First referee: Prof. Dr. Karl-Henning Rehren

Second referee: Prof. Dr. Laura Covi

Abstract

Positivity analysis of correlation functions in axiomatic conformal quantum field theory is a nontrivial issue. The partial wave analysis provides a powerful tool to test for positivity, however, the computation of partial waves remains difficult. Recent advances in the method of intertwining operators make this computation feasible even in four spacetime dimensions. In this thesis we outline basics of the theory of intertwining operators and develop the necessary computer algebra tools to calculate partial waves. We then apply them to an exotic four-point structure of currents found by Yassen Stanev that cannot arise in free theories, making it an interesting candidate for an interacting theory provided its positivity. We finally show that it is not positive on its own and give certain bounds on combined theories of the exotic, free Dirac and free Bose currents to form a possibly positive theory.

Contents

1 Introduction **1**

2 General framework **3**

2.1 Wightman Axioms 3
2.2 Polarization vectors 6
2.3 Conformal invariance 6
2.4 Correlation functions in CFT 9
2.5 Positivity problem and Operator Product Expansion 12
2.6 Partial Wave Analysis 14
2.7 The exotic four-point structure 15
2.8 The free Bose and Dirac currents in CFT 16

3 Reduction of n-point functions **19**

3.1 Intertwining property 19
3.2 Derivation of a partial differential equation for the operators 20
3.3 Positivity tests 23

4 Development and description of used CAS tools **25**

4.1 Operator derivation 25
4.2 Generating the ansatz 27
4.2.1 Scalar intertwiners 27
4.2.2 Pseudo scalar intertwiners 28
4.3 Representation and differentiation of the correlation functions 28
4.4 Documentation and manual 29

5 Applications and results **31**

5.1 Treating the exotic structure 31
5.2 Contributions of representations $\kappa = 3$, $L = 0$ and $\kappa = 4$, $L = 0$ . . . 32
5.3 Combining the exotic with the free bosonic and fermionic structure . 34

5.4 Conclusions 41

6 Summary and outlook **43**

A Documentation **45**

A.1 *Maple* scripts 45

A.1.1 utils.mpl 45

A.1.2 diffsolver.mpl 45

A.1.3 diffops.mpl 46

A.2 External scripts 47

B Sources **49**

B.1 utils.mpl 49

B.2 diffsolver.mpl 49

B.3 diffops.mpl 56

Bibliography **61**

Nomenclature

L	The rank of a tensor field.
d	The scaling dimension of a tensor field.
2κ	The twist of a field, $2\kappa := d - L$.
$[F]_{(i,j)}$	The symmetrization of F under the exchange of $i \leftrightarrow j$
η	The Minkowski metric $\eta = \mathrm{diag}(1, -1, -1, -1)$.
$\iota^x_{x_1,x_2}$	The evaluation map setting $x_1 = x_2$ and calling the result x. If x is omitted, the result is called x_1.
$\langle A \rangle$	Short notation for the inner product $\langle \Omega, A\Omega \rangle$.
$\lfloor a \rfloor$	The integer part of a.
$\mathbb{M}$	The Minkowski space $\mathbb{M} := \mathbb{R}^{1,3}$.
$\mathcal{S}(\mathbb{M})$	The Schwartz space of functions on $\mathbb{M}$.
∇_i	The derivative w.r.t. y_i.
∇_{v_i}	The derivative w.r.t. w_i.
Ω	The distinct vacuum state.
$\overline{\mathbb{M}}$	The conformally compactified Minkowski space.
ρ_{ij}	$\rho_{ij} := (x_i - x_j - 0ie_0)^2$.
$\varphi(v, x)$	The tensor field $v_{\mu_1,\mu_2,\ldots}\varphi^{\mu_1,\mu_2,\cdots}$ at spacetime position x.
$a \cdot b$	The product $a_\mu b_\nu \eta^{\mu\nu}$.

J	Conserved current of representation $\kappa = 1$, $L = 1$.
w_i	The derivative ∂_i^v w.r.t. v_i.
$x \wedge y$	Wedge product of the Lorentz vectors x and y, $x \wedge y := x_\mu y_\nu - x_\nu y_\mu$.
y_i	The derivative ∂_i w.r.t. x_i, considered as a variable of the polynomial intertwining operator E.

1 Introduction

The theory of quantum fields (QFT) has been developed in the 1940s and 1950s as an attempt to fix several conceptional flaws of quantum mechanics. The facts that QM is a necessarily non-relativistic theory, that concepts like the Dirac sea are needed to explain anti particles and that it cannot describe creation and annihilation processes of particles are just a few examples of the theorie's weaknesses. The former makes it impossible to describe massless particles like the common photon, which behaves necessarily relativistic.

Today, the theory of quantum fields has been applied to many fields of physics, featuring most prominently particle physics, leading to the development of the Standard Model, where it is used to describe three of the four fundamental forces of nature. As a result, the development of this theory is still vivid today and of great importance not just in particle physics, but also in statistical mechanics, condensed matter physics, cosmology and many more.

However, ever since quantum field theories have been used, most predictions were derived using a perturbative approach that gives rise to many divergent integrals, for example UV and IR divergences. To circumvent them, one can drop the concept of Lagrangians, which make the use of renormalization theory inevitable, and instead work with the assumption that the quantized fields are given, fulfilling a basic set of axioms to ensure important physical properties like Lorentz invariance. From this setting, it is possible to derive powerful results like the LSZ formula, which allows the calculation of scattering amplitudes in terms of vacuum expectation values of products of fields, called correlation functions.

In axiomatic quantum field theory the knowledge of all correlation functions allows the reconstruction of the whole theory, including the Hilbert space and therefore its inner product; a sound theory of course implies that the inner product is positive. Whether this is the case is a nontrivial issue and subject to recent research.

In conformal quantum field theory, where the symmetry group is enlarged to all transformations that leave orientation and angles invariant, the structure of corre-

lation functions of products of n fields is fixed and well understood, the problem of positivity of two-point functions reduces to the determination of its sign. This encourages expressing higher functions in terms of two-point functions, which is done by partial wave analysis: A positive n-point function is positive if this holds for all its partial two-point functions too.

However, determining the partial waves is nontrivial. In lower space time dimensions, using Casimir operators is a common way of approaching this problem, but is getting far too complicated in 4D.

Instead, Rehren, Wallenhorst and Neumann found another method of so called intertwining differential operators [6] to determine the partial waves. Within this thesis we will use the intertwining method to investigate the positivity of an exotic four-point structure of four current fields found by Stanev in [12], which is of high interest as it cannot arise in non interacting theories and all valid theories found so far in four dimensional QFT correspond to free fields.

Within this thesis we will proceed as follows: The first part of chapter 2 will describe the basic axiomatic setup in which we will work and will collect the necessary implications of the conformal symmetry for quantum field theories with a focus on the structure of correlation functions. The second part will consider the partial wave analysis as a tool to reduce the general n-point functions to $n-1$-point functions and introduces the correlation function of interest along with the four-point functions of the free Bose and Dirac currents for later comparison.

Chapter 3 introduces the method of intertwining differential operators and roughly sketches the origin of this method. The differential equation that generates the intertwining operators is derived and finally we summarize the possible positivity tests used later in the application of the method on the exotic four-point structure.

The necessary computer tools to apply the method are developed and presented in chapter 4. Their applications and positivity tests are conducted in chapter 5.

The thesis is summarized in chapter 6 and concludes with an outlook.

2 General framework

The "usual" approach to QFT, following the historical path of the development of quantized field theories, results in a great number of problems, including the lack of a consistent mathematical description of quantum fields. In order to work with a solid mathematical foundation, we will use the axiomatic framework instead: Proposing basic features on the theory and its quantum fields, one can derive mathematically sound theories. These basic features normally include locality and positivity features, to prevent violation of causality and to allow a probability interpretation of observables. Additionally one requires at least a symmetry of the Poincaré group to ensure observation of the same physical rules by every spectator.

Within this thesis, we will use the Wightman axioms [13], formulated by A. S. Wightman in 1950. Throughout this chapter we will use the metric tensor $\eta = \mathrm{diag}(1, -1, -1, -1)$ and natural units $\hbar = c = 1$.

2.1 Wightman Axioms

Instead of formulating the axioms for the fields φ themselves, it is in this case more appropriate to consider the equivalent formulation for the correlation functions, or Wightman functions, defined as the vacuum expectation value of a product of fields. The reconstruction theorem [11] then allows to recover the whole QFT from the correlation functions. We will however stick to correlation functions, because they will be of importance in the later chapters.

As a basis for our theory, we will need

- a separable Hilbert space $\mathcal{H}$,
- the (distinct) normalized vacuum state $\Omega \in \mathcal{H}$, which is invariant under the action of
- a covering group P of the Poincaré group (as we want to enlarge the symmetry later on) and $U(g)$, a unitary representation for $g \in P$, acting on $\mathcal{H}$,

- the set of fields $\{\varphi_i\}_{i=1\dots N}$ and their adjoints $\{\varphi_i^\star\}_{i=1\dots N}$, where $\varphi_i : \mathcal{S}(\mathbb{M}) \to \mathcal{O}(\mathcal{H})$ are operator valued distributions. Here, $\mathcal{S}(\mathbb{M})$ denotes the Schwartz space of functions on Minkowski space and $\mathcal{O}(\mathcal{H})$ is the set of all operators on $\mathcal{H}$ that are defined on a dense linear subset $D \subset \mathcal{H}$ with $\Omega \in D$. D shall be invariant under the action of P and all operators in $\mathcal{O}(\mathcal{H})$ shall project vectors of D to D again. We need to treat the fields as distributions, as e.g., commutation relations of the fields yield δ-distributions and can therefore not be treated as mere functions. However, most of the time we will ignore this fact and work with the fields $\varphi(x)$ as if they were functions unless the distinction becomes necessary. Equations involving $\varphi(x)$ are then to be thought of multiplied by a test function and integrated over.

Definition 1. In order to work with correlation functions, we define the n-point *correlation function* (or Wightman function) $W_{\varphi_{i_1}\dots\varphi_{i_n}}(x_1,\dots,x_n)$ of a subset of fields $\{\varphi_i\}_{i\in I}$ as

$$W_{\varphi_{i_1}\dots\varphi_{i_n}}(x_1,\dots,x_n) := \langle\Omega, \varphi_{i_1}(x_1)\dots\varphi_{i_n}(x_n)\Omega\rangle\,,$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product of $\mathcal{H}$. If the involved fields are arbitrary, we omit them from the notation. Again, the Wightman function W is to be thought of as a tempered distribution $W(f_1,\dots,f_n)$ with test functions $f_1,\dots,f_n$.

We can now define our QFT:

Definition 2. The set of all correlation functions and the data given above are called a *Wightman QFT*, if they fulfill the following axioms [10, 13]:

Hermiticity For all correlation functions $W_{\varphi_{i_1}\dots\varphi_{i_n}}(x_1,\dots x_n)$ holds

$$W_{\varphi_{i_1}\dots\varphi_{i_n}}(x_1,\dots,x_n) = \overline{W_{\varphi^\star_{i_n}\dots\varphi^\star_{i_1}}(x_n,\dots,x_1)}.$$

This is needed to ensure hermiticity of the reconstructed inner product, given by the reconstruction theorem for vectors $\varphi_{i_1}(f_1)\dots\varphi_{i_n}(f_n)\Omega$ as

$$\begin{aligned}&\langle\varphi_{i_1}(f_1)\dots\varphi_{i_n}(f_n)\Omega, \varphi_{j_1}(g_1)\dots\varphi_{j_n}(g_n)\Omega\rangle\\ &\quad := W_{\varphi^\star_{i_n}\dots\varphi^\star_{i_1}\varphi_{j_1}\dots\varphi_{j_n}}(\bar f_n,\dots,\bar f_1, g_1,\dots,g_n).\end{aligned}$$

Locality If x_i and x_{i+1} are separated spacelike, i.e. $(x_i - x_{i+1})^2 < 0$ using $\eta = \mathrm{diag}(1, -1, \ldots)$, for any correlation function W holds

$$W(\ldots, x_i, x_{i+1}, \ldots) = W(\ldots, x_{i+1}, x_i, \ldots),$$

as the underlying fields $\varphi(x_i)$ and $\varphi(x_{i+1})$ do commute at spacelike distances:

$$[\varphi(x_1), \varphi(x_2)] = 0. \tag{2.1.1}$$

Covariance The correlation functions are covariant under the action of $g \in P$;

$$W(x_1, \ldots, x_n) = W(g \circ x_1, \ldots, g \circ x_n).$$

As the translations are part of P, the correlation functions depend only on relative coordinates, since they are Lorentz and translation invariant. As a result, one can write

$$W(x_1, \ldots, x_n) = W'(x_2 - x_1, \ldots, x_n - x_{n-1}).$$

Spectrum Condition The Fourier transformations of $W'(x_2 - x_1, \ldots, x_n - x_{n-1})$ are tempered distributions and have their support only on the future light cone. This feature admits only positive energy solutions.

Positivity Let φ be any field of the QFT and $\{f_i\}_{i \in I}$ a set of test functions, then

$$\sum_{i,j \in I} \int \overline{f_i(x_1, \ldots, x_i)} W_{\varphi^\star \ldots \varphi^\star, \varphi, \ldots, \varphi}(x_i, \ldots, x_1, y_1, \ldots, y_j) \cdot f_j(y_1, \ldots, y_j) \mathrm{d}x_1 \ldots \mathrm{d}x_i \mathrm{d}y_1 \ldots \mathrm{d}y_j \;\geq\; 0 \tag{2.1.2}$$

must hold to ensure positivity of the reconstructed inner product. To verify this property for certain correlation functions will be the main concern of this thesis.

Cluster Decomposition Property For spacelike $a \in \mathbb{M}$ holds

$$\lim_{\lambda \to \infty} W(x_1, \ldots, x_i, x_{i+1} + \lambda a, \ldots, x_n + \lambda a) = W(x_1, \ldots, x_i) W(x_{i+1}, \ldots, x_n),$$

which is another aspect of locality ensuring operators to behave independently at spacelike separated distances.

To shorten the notation, we will write for the vacuum expectation value

$$\langle \Omega, A\Omega \rangle \equiv \langle A \rangle$$

from now on.

2.2 Polarization vectors

In the following, we will often have to consider tensor fields $T^{\mu_1 \cdots \mu_n}(x)$. In order to simplify the calculations, i.e. avoid Lorentz indices in computer algebra systems (CAS), we use so called polarization vectors $v_1 \dots v_n \in \mathbb{M}$ and consider the quasi scalar field

$$T(\mathbf{v}, x) := v_{1,\mu_1} \dots v_{n,\mu_n} T^{\mu_1 \cdots \mu_n}(x)$$

instead. The original field can be recovered by

$$T^{\mu_1 \cdots \mu_n}(x) = \partial_{v_1}^{\mu_1} \dots \partial_{v_n}^{\mu_n} T(\mathbf{v}, x).$$

If $T^{\mu_1 \cdots \mu_n}(x)$ is totally symmetric, a single polarization vector suffices and from

$$T(v, x) := v_{\mu_1} \dots v_{\mu_n} T^{\mu_1 \cdots \mu_n}(x)$$

T can be recovered by

$$T^{\mu_1 \cdots \mu_n}(x) = \frac{1}{n!} \partial_v^{\mu_1} \dots \partial_v^{\mu_n} T(v, x).$$

A trace of a rank 2 tensor can be written as

$$T_\mu^\mu = \frac{1}{2} \partial_v^2 T(v, x). \tag{2.2.1}$$

2.3 Conformal invariance

To make decent predictions on the admissible theories, the Poincaré group is enlarged to the conformal group, which consists of all transformations that leave orientations

and angles invariant, including dilations; in two space time dimensions one could think of Möbius transformations to visualize the conformal group. Even though we know, due to e.g. masses of particles, that reality is by no means invariant under such transformations, the conformal field theories (CFT) are useful for applications in statistical mechanics due to scale invariance at phase transitions and in string theory, where the AdS/CFT correspondence is a prominent example.

In order to answer the question whether the Wightman axioms are too strict to allow interacting theories, the study of CFTs is indeed useful due to the easier structure they imply on the correlation functions.

In terms of the metric tensor, the conformal transformations can be expressed as all mappings $x \mapsto x'$ which leave the tensor invariant up to a scale $\Lambda(x)$ [5]. Those mappings can be classified as [1]

- Poincaré transformations $x^\mu \mapsto \Lambda^\mu_\nu x^\nu$ and $x^\mu \mapsto x^\mu + a^\mu$ for any Lorentz matrix Λ and $a \in \mathbb{M}$,
- Dilations $x^\mu \mapsto \lambda x^\mu$ with $\lambda \in \mathbb{R}^+$,
- Special conformal transformations $x^\mu \mapsto \frac{x^\mu - x^2 b^\mu}{1 - 2b\cdot x + b^2 x^2}$ with $b^\mu \in \mathbb{M}$.

Additionally the fields $\varphi(x)$ (scalar or tensor fields) will transform, e.g.

- $\varphi(x) \mapsto L_\Lambda \varphi(\Lambda^{-1} x)$ for Lorentz transformations with a matrix L_Λ if φ is a tensor field, $\varphi(x) \mapsto \varphi(x - a)$ for translations,
- $\varphi(x) \mapsto \lambda^d \varphi(\lambda x)$ for dilations where d denotes the scaling dimension of φ, an important quantum number in later considerations.

From these one can derive the generators $M_{\mu\nu}$ of the Lorentz group, D of the dilations, P_μ of the translations and K_μ of the special conformal transformations, their commutation relations with the (contracted symmetric, traceless tensor) fields $\varphi(v, x)$, given by [5, 14]

$$
\begin{aligned}
i[P_\mu, \varphi(v,x)] &= \partial_\mu \varphi(v,x), && (2.3.1)\\
i[D, \varphi(v,x)] &= (x\cdot\partial + d)\varphi(v,x), && (2.3.2)\\
i[M_{\mu\nu}, \varphi(v,x)] &= (x_\mu\partial_\nu - x_\nu\partial_\mu + v_\mu\partial^v_\nu - v_\nu\partial^v_\mu)\varphi(v,x)\\
&\equiv (x\wedge\partial + v\wedge\partial^v)\varphi(v,x), && (2.3.3)\\
i[K_\mu, \varphi(v,x)] &= (2x_\mu(x\cdot\partial) - x^2\partial_\mu + 2dx_\mu\\
&\quad +2v_\mu(x\cdot\partial^v) - 2(x\cdot v)\partial^v_\mu)\varphi(v,x), && (2.3.4)
\end{aligned}
$$

and the commutation relations among the generators which define the conformal Lie algebra [5]:

$$
\begin{aligned}
i[P_\mu, D] &= P_\mu, && (2.3.5)\\
i[K_\mu, D] &= -K_\mu, && (2.3.6)\\
i[K_\mu, P_\nu] &= 2(M_{\mu\nu} - \eta_{\mu\nu}D), && (2.3.7)\\
i[K_\rho, M_{\mu\nu}] &= \eta_{\rho\nu}K_\mu - \eta_{\rho\mu}K_\nu, && (2.3.8)\\
i[P_\rho, M_{\mu\nu}] &= \eta_{\rho\nu}P_\mu - \eta_{\rho\mu}P_\nu, && (2.3.9)\\
i[M_{\mu\nu}, M_{\rho\sigma}] &= \eta_{\mu\rho}M_{\nu\sigma} + \eta_{\nu\sigma}M_{\mu\rho} - \eta_{\nu\rho}M_{\mu\sigma} - \eta_{\mu\sigma}M_{\nu\rho}. && (2.3.10)
\end{aligned}
$$

Vanishing commutators are not listed. Here, $x\wedge y$ means $x_\mu y_\nu - x_\nu y_\mu$.

Note that the special conformal transformations give rise to singularities, i.e. they map certain spacetime points to infinity. One therefore has to treat the compactified Minkowski space

$$\overline{\mathbb{M}} = \left\{ z = (z_1, z_2, z_3, z_4) \in \mathbb{C}^4 \,|\, z = \frac{\bar{z}}{\bar{z}^2} \right\}$$

instead to include these points, leading to globally conformal invariant theories. A detailed definition and analysis is given in [7] and [8].

Definition 3. A Wightman QFT on Minkowski space is called *globally conformal invariant* (GCI) if for any conformal transformation g and all sets of points $(x_1, \ldots, x_N)$ with $x_i \in \mathbb{M}$ and $g \circ x_i \in \mathbb{M}$ holds that all correlation functions $W(x_1, \ldots, x_N)$ are invariant under g [7].
In fact, any well defined GCI QFT can be extended to $\overline{\mathbb{M}}$ [14].

GCI gives rise to more restrictions on the fields and their correlation functions. One of them is the so called Huygens principle

$$[\varphi(x), \varphi(y)] = 0$$

for $(x-y)^2 \neq 0$, as special conformal transformations can map spacelike points to timelike points and vice versa and fields commute at spacelike distances by (2.1.1). This means that information is only transferred on the light cone and all masses are zero.

The scaling dimension d together with the two spin quantum numbers j_1 and j_2 of the Lorentz representation characterize the irreducible unitary representations. It follows from global conformal invariance, that d and $j_1 + j_2 = L$, the rank of a tensor field, have to be integer [7].

A result given in [3] states that all irreducible unitary positive weights are lowest weight representations, giving restrictions on admissible quantum numbers, called the *unitarity bound*. They can be classified as follows:

$$\begin{aligned} &\bullet \quad d = j_1 = j_2 = 0, \\ &\bullet \quad j_1 \neq 0,\ j_2 \neq 0,\ d \geq j_1 + j_2 + 2, \\ &\bullet \quad j_1 \cdot j_2 = 0,\ d \geq j_1 + j_2 + 1. \end{aligned} \tag{2.3.11}$$

Representations below this bound manifestly violate positivity.

2.4 Correlation functions in CFT

The high symmetry of the conformal group gives rise to many constraints on admissible correlation functions in a conformal field theory and fixes the 1-point, 2-point and 3-point functions of scalar fields up to a normalization constant [1].

1-point functions

Due to translation invariance, for all $a \in \mathbb{M}$ holds $W_\varphi(x) = W_\varphi(x+a)$ and therefore all 1-point-functions have to be constant. As the conformal group also contains dilations, W_φ has to be homogeneous in x: The only scale invariant constant is 0

and therefore all 1-point functions vanish:

$$W_\varphi(x) = 0.$$

2-point functions

The 2-point functions of fields φ and φ' are zero, unless $(d, j_1, j_2) = (d', j_2', j_1')$. In case of scalar fields ($j_1 = j_2 = 0$) the function is fixed by conformal invariance and is given in by

$$W_{\varphi,\varphi'}(x_1, x_2) = \delta_{d,d'} \frac{C}{\rho_{12}^d}, \tag{2.4.1}$$

with the common abbreviation $\rho_{ij} := (x_i - x_j - 0ie_0)^2$ where $e_0 = (1, 0, 0, 0) \in \mathbb{M}$.

For general symmetric, traceless tensor fields $\varphi(v, x_1)$, $\varphi'(v', x_2)$ holds [2]

$$W_{\varphi,\varphi'}(x_1, x_2) = \delta_{d,d'}\delta_{j_1,j_2'}\delta_{j_2,j_1'} C \left[(v_1^\mu v_2^\nu R_{\mu\nu}(x_{12}))^L\right]_{00} \left(\frac{-1}{\rho_{12}}\right)^{d-L} \tag{2.4.2}$$

where

$$R_{\mu\nu}(x_{12}) := \frac{1}{\rho_{12}^2} \left(\eta_{\mu\nu}\rho_{12} - 2x_{12,\mu}x_{12,\nu}\right) \tag{2.4.3}$$

is the primitive covariant and

$$[A(v_1, v_2)]_{00}$$

denotes the traceless part, i.e. the harmonic part of A w.r.t. to v_1 and v_2, that is (c.f. (2.2.1)),

$$\partial_{v_1}^2 \left[A(x_1, x_2)\right]_{00} \equiv \Box_{v_1} \left[A(x_1, x_2)\right]_{00} = 0 = \Box_{v_2} \left[A(x_1, x_2)\right]_{00}.$$

Note that the harmonic decomposition of a complex power series is unique [9].

The two-point function (2.4.2) is conserved iff for scaling dimension d and rank L hold

$$d - L = 2.$$

Therefore it is useful to define the *twist* of a field:

Definition 4. The *twist* 2κ of a tensor field of rank L and scaling dimension d is defined as

$$2\kappa := d - L.$$

The Reeh-Schlieder theorem then implies that correlation functions are conserved in the spacetime variables of twist-2 fields [9].

3-point functions

The 3-points function of scalar fields are determined by the scaling dimensions d_i of the involved fields φ_i and read [14]

$$W_{\varphi_1,\varphi_2,\varphi_3}(x_1,x_2,x_3) = \frac{C}{\rho_{12}^{\frac{1}{2}(d_1+d_2-d_3)}\rho_{13}^{\frac{1}{2}(d_1-d_2+d_3)}\rho_{23}^{\frac{1}{2}(-d_1+d_2+d_3)}}.$$

For general fields there is no known explicit form. However, GCI gives further restrictions on the twists as only fields with even twist give rational three-point functions and are therefore admissible [6].

n-point functions

Due to globally conformal invariance, n-point functions for arbitrary n are rational [7], for scalar fields φ_i of scaling dimension d_i it follows that [7]

$$W_{\varphi_1\dots\varphi_n}(x_1,\dots,x_n) = \sum_\mu C_\mu \prod_{i<j} \rho_{ij}^{\mu_{ij}}, \tag{2.4.4}$$

where μ denotes the multi-index $\mu = \{\mu_{ij}\}_{i<j}$, obeying the sum rule

$$\sum_i \mu_{ij} = -d_j$$

in order to ensure the scaling properties.

The unitarity bounds (2.3.11) can be expressed in terms of the coefficients $\{\mu_{ij}\}_{i<j}$ as [7]

$$\mu_{ij} \geq -\left\lfloor \frac{d_i + j_{i,1} + j_{i,2} + d_j + j_{j,1} + j_{j,2} - (1 - \delta_{j_{i,1}j_{j,2}}\delta_{j_{i,2}j_{j,1}}\delta_{d_i d_j})}{2} \right\rfloor, \tag{2.4.5}$$

which will be one of the positivity tests to be conducted later. $\lfloor a \rfloor$ denotes the integer part of a. These bounds do also imply that the set of multi-indices is finite, thus the sum in (2.4.4) has a finite number of terms.

For non-scalar fields, an additional tensor valued polynomial occurs.

2.5 Positivity problem and Operator Product Expansion

An important and complicated task is the determination whether the inner product of the reconstructed Hilbert space from the known correlation functions is admissible. While all other properties are easy to check, the positive-definiteness is complicated and almost impossible to accomplish: One has to make sure, that for all vectors

$$\varphi(f) = \int \mathrm{d}^4x_1 \ldots \mathrm{d}^4x_n \, f(x_1, \ldots, x_n)\varphi_1(x_1) \ldots \varphi_n(x_n)\Omega$$

of fields φ_i the resulting norm $\|\varphi(f)\|^2$, which coincides with the axiom (2.1.2), is positive for all test functions and vectors [8]. Positivity is therefore a feature of the correlation functions. This and the fact that they are determined for small number of fields in the GCI case as shown above is the reason why we chose the correlation function description of the Wightman axioms in the first place.

An important tool to analyze the positivity of a correlation function is the operator product expansion of two fields. It exploits the proposition that products of fields are reducible representations, while fields themselves are irreducible. Assuming that the product is expandable in a Laurent series, one may define:

Definition 5. The operator product expansion (OPE) of two local (for now scalar) quantum fields $\varphi_1(x)$ and $\varphi_2(y)$ (operator valued distributions) of scaling dimension d is the expansion of the product of the fields in a series around $(x - y)$ in terms of a complete set of local fields $\{\varphi_i\}$, i.e. [15]

$$\varphi_k(x)\varphi_l(y) = \sum_i C_i(x - y)\varphi_i(y),$$

with functions $C_i(x-y)$ depending on k and l. In GCI, $C(x-y)$ is a Laurent series in $x - y$.

These expansion in general only exist in small regions around y. In GCI, the set of fields consists of fields carrying an irreducible representation of the conformal group. One may rearrange the power series by the twist of the involved fields. Considering the general two-point function (2.4.2), it can be seen that the most singular contribution is the twist-0 (vacuum) contribution, proportional to ρ_{12}^{-d}, which can easily be identified within the correlation function and therefore can be neglected. Hence it is useful to consider the OPE without vacuum contribution and make it regular

by multiplying with ρ_{12}^{d-1} to write it as

$$\begin{aligned} U(x,y) &:= \rho_{12}^{d-1}(\varphi_1(x)\varphi_2(y) - \langle\varphi_1(x)\varphi_2(y)\rangle) && (2.5.1)\\ &= V_1(x,y) + \rho_{12}V_2(x,y) + \dots. && (2.5.2)\end{aligned}$$

where V_κ is the twist-2κ contribution to the OPE.

It can be shown using the conservation laws for twist-2 fields and explicit forms of the OPE that the harmonic part of (2.5.1) is the twist-2 contribution V_1 [9], i.e.

$$\Box_x V_1(x,y) = \Box_y V_1(x,y) = 0.$$

Since the harmonic decomposition of a polynomial is unique, there are two rules to obtain V_1 from U. This condition can be refactored to a PDE that the leading contribution to V_1 has to fulfill. All solutions to this condition can be classified according to the poles of V_1:

- Either V_1 has *single poles* in ρ_{1k} with $k \in \{3,\dots,n\}$, that is, $V_1 \propto \frac{1}{\rho_{1k}^a}$ with $a > 0$ for just one k
- or V_1 has *double poles* in ρ_{1k} and ρ_{1l}, $k \in \{3,\dots,n\}$, $l \in \{3,\dots,n\}$, $l \neq k$ and no other poles ρ_{1m} with $m \neq k$ and $m \neq l$ may occur

and the same for x_2. This gives restrictions on admissible correlation functions of all kinds of fields in GCI.

In fact, the twist-2 contributions to currents of free fields, which we will study in 2.8, do never exhibit double poles. Consider for example the free Bose current

$$\begin{aligned} J_\mu(x) &= i{:}\,\varphi^\star(x)\overset{\leftrightarrow}{\partial}_\mu\varphi(x){:} = \iota^x_{x,x'} \circ (\partial^\mu - \partial'^\mu) i{:}\,\varphi^\star(x')\varphi(x){:}\\ &\equiv D_x{:}\,\varphi^\star(x')\varphi(x){:}\end{aligned}$$

with $D_x = \iota^x_{x,x'} \circ (\partial^\mu - \partial'^\mu)i$ and the operator product $J_\mu(x_1)J_\nu(x_2) \equiv J_1 J_2$. Using Wick's theorem, one gets

$$\begin{aligned} J_1J_2 &= \langle J_1J_2\rangle + D_{x_1}D_{x_2}\left(\langle\varphi(x_1)\varphi^\star(x_2')\rangle : \varphi^\star(x_1')\varphi(x_2){:}\right.\\ &\qquad \left. + \langle\varphi^\star(x_1')\varphi(x_2)\rangle : \varphi(x_1)\varphi^\star(x_2'){:}\ \right) + \dots\end{aligned}$$

where the middle terms are the twist-2 contributions and $\langle\varphi^\star(x_1')\varphi(x_2)\rangle \propto \rho_{12}^{-1}$.

All further contractions of the twist-2 part with other fields will exhibit only one additional two-point function of the form $\langle\varphi(x_1)\varphi(x_k)\rangle$ and $\langle\varphi(x_2)\varphi(x_l)\rangle$ with $k, l > 2$, thus no double poles can occur. The same argument also holds for Dirac currents. Any correlation function exhibiting double poles can therefore not arise in free theories and indicates, provided it obeys the Wightman axioms, an interacting theory.

2.6 Partial Wave Analysis

In order to test for positivity, the operator product expansion can be used to write improper vectors of the form

$$\varphi_1\varphi_2\Omega$$

within correlation functions as a linear combination of single quantum fields of different representations λ, using projectors Π_λ, which satisfy $\sum_\lambda \Pi_\lambda = 1$:

$$\varphi_1(x)\varphi_2(y)\Omega = \sum_\lambda \Pi_\lambda\varphi_1(x)\varphi_2(y)\Omega \equiv \sum_\lambda \int K(\lambda, x, y, z)\varphi_\lambda(z)\Omega \mathrm{d}z,$$

where $K(\lambda)$ denotes a suitable integration kernel.

Thus the n-point correlation functions can be reduced to $n-1$-point correlation functions in the corresponding vector norm by projecting with Π_λ to the partial wave of the representation λ of the correlation functions. Each of these partial waves has to be positive on its own.

The canonical approach on determining the partial waves is by using Casimir operators C [14]. The Lemma of Schur tells us that every irreducible representation space is an eigenspace of a Casimir operator, i.e.

$$C\Pi_\lambda\varphi_1\varphi_2\Omega = c_\lambda\Pi_\lambda\varphi_1\varphi_2\Omega, \tag{2.6.1}$$

holds with $c_\lambda \in \mathbb{C}$. As C commutes with the projector and can be expressed as a linear combination of the generators of the algebra, the commutation relations with the fields are known (eqn. (2.3.1)-(2.3.4)) and C can be commuted past the fields. Using that Ω is invariant under the action of the group, one can write

$$C\Pi_\lambda\varphi_1\varphi_2\Omega = D_{12}\Pi_\lambda\varphi_1\varphi_2\Omega, \tag{2.6.2}$$

where D_{12} is a differential operator acting on the fields.

Identifying (2.6.1) with (2.6.2), an eigenvalue equation is obtained, which one might be able to solve. However, the conformal Lie algebra in four spacetime dimensions possesses three Casimir operators, where the simplest is [14]

$$C = \frac{1}{2}\left(K \cdot P + P \cdot K\right) - D^2 + \frac{1}{2} M_{\mu\nu} M^{\mu\nu},$$

giving eigenvalue equations that are almost impossible to solve. The other two operators are even cubic and quartic in the generators. As a consequence we need to use another method to determine the partial waves.

2.7 The exotic four-point structure

While analyzing the general four-point function of conserved currents J ($\kappa = 1$, $L = 1$) in [12], Yassen S. Stanev found, apart from the two free field functions of Bose and Dirac currents, a third admissible correlation, which might be part of an interacting theory, provided that it is (or in a certain combination with the free field functions) positive.

Recalling the definition of the primitive covariant R given in (2.4.3), this four-point structure can be expressed as

$$\begin{aligned}\langle J_1 J_2 J_3 J_4 \rangle \;=\; & v_{1,\alpha_1} v_{2,\alpha_2} v_{3,\alpha_3} v_{4,\alpha_4} \varepsilon^{\alpha_1\mu_1\nu_1\rho_1} \varepsilon^{\alpha_2\mu_2\nu_2\rho_2} \varepsilon^{\alpha_3\mu_3\nu_3\rho_3} \varepsilon^{\alpha_4\mu_4\nu_4\rho_4} \\ & \times R_{\mu_1\mu_2}(x_{12}) R_{\nu_1\nu_3}(x_{13}) R_{\rho_1\rho_4}(x_{14}) R_{\rho_2\rho_3}(x_{23}) R_{\nu_2\nu_4}(x_{24}) R_{\mu_3\mu_4}(x_{34}). \quad (2.7.1)\end{aligned}$$

Here, the v_i correspond to the polarization vectors of the four resp. currents J_i and ε denotes the totally antisymmetric tensor. The structure is by construction conserved in all four currents.

The graphical representation in fig. 2.1 reveals the high symmetry of this structure, showing that it is invariant under the exchange of any two x_i, where

- every vertex i corresponds to one $\varepsilon^{\alpha_i \cdots}$-tensor and the spacetime point x_i,
- every line ending in that vertex corresponds to one of the Lorentz indices of ε,
- an internal line connecting vertices i and j corresponds to a primitive covariant $R_{\ldots}(x_{ij})$ with Lorentz indices given by the line endings,

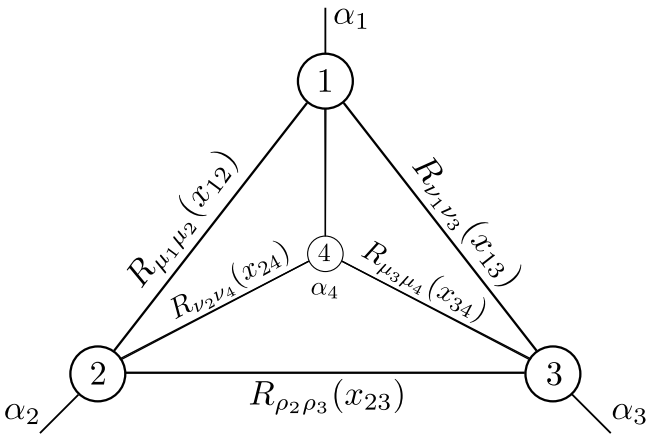

Figure 2.1: A graphical representation of the exotic four-point structure. Every vertex represents an ε-tensor and every line a primitive covariant.

- an external line connected to vertex i stands for v_{i,α_i}.

Of great interest is the pole structure of (2.7.1), given by

$$\frac{1}{\rho_{12}^2\rho_{13}^2\rho_{14}^2\rho_{23}^2\rho_{24}^2\rho_{34}^2}.$$

Note that the exponents of ρ_{13}^2 and ρ_{14}^2 are both positive, thus this correlation function cannot be realized by currents of free fields as seen in chapt. 2.5, motivating the term *exotic*.

2.8 The free Bose and Dirac currents in CFT

We will later need to combine the exotic structure with the free Dirac and Bose current four-point structure to compensate for occurring positivity violations and therefore need to determine them.

Consider primarily the free Bose current J_μ of a complex scalar field $\varphi_i = \varphi(x_i)$ of scaling dimension $d = 1$. By conformal invariance the two-point function is fixed as (c.f. (2.4.1))

$$\langle \varphi_1 \varphi_2^\star \rangle = \langle \varphi_1^\star \varphi_2 \rangle = \frac{-1}{\rho_{12}}.$$

The current is given by

$$J_\mu = i \colon \varphi^\star \overset{\leftrightarrow}{\partial}_\mu \varphi \colon$$

which can be rewritten to make Wick's theorem applicable as

$$J^\mu = \iota_{x,x'} \circ (\partial^\mu - \partial'^\mu) i{:}\, \varphi^\star(x')\varphi(x) {:}\, .$$

Using the abbreviation $\langle A'B\rangle = \langle \varphi^\star(x'_A)\varphi(x_B)\rangle = \frac{-1}{\rho_{A'B}}$ one can then calculate using Wick's theorem

$$\begin{aligned}\langle J_1^{\mu_1} J_2^{\mu_2} J_3^{\mu_3} J_4^{\mu_4}\rangle_{\text{Bose}} &= V^{\mu_1\mu_2\mu_3\mu_4}(\langle 1'3\rangle\langle 13'\rangle\langle 2'4\rangle\langle 24'\rangle + \langle 1'3\rangle\langle 14'\rangle\langle 2'4\rangle\langle 23'\rangle \\ &\quad +(6 \text{ similar terms}) + \langle 1'2\rangle\langle 12'\rangle\langle 3'4\rangle\langle 34'\rangle). \qquad (2.8.1)\end{aligned}$$

Here, $V^{\mu_1\mu_2\mu_3\mu_4} = i^4 \iota_{x_1,x'_1} \circ \iota_{x_2,x'_2} \circ \iota_{x_3,x'_3} \circ \iota_{x_4,x'_4} \circ (\partial_1^{\mu_1} - \partial_1'^{\mu_1}) \dots (\partial_4^{\mu_4} - \partial_4'^{\mu_4})$.

The last term in (2.8.1) corresponds to the vacuum contribution with highest singularities and will be neglected. The highest singularity in x_{12} of the remaining expression is proportional to $\frac{1}{\rho_{12}^3}$.

Consider now free Dirac fields ψ_i and their two-point functions

$$\left\langle \psi_{1,a}\bar\psi_{2,b}\right\rangle = i\not\partial_{ab}\frac{-1}{\rho_{12}} = 2i\frac{\not x_{12,ab}}{\rho_{12}^2} =: \left\langle 1_a\bar 2_b\right\rangle$$

and

$$\left\langle \bar\psi_{1,a}\psi_{2,b}\right\rangle = 2i\frac{x_{12,\mu}\gamma^\mu_{ba}}{\rho_{12}^2} =: \left\langle \bar 12\right\rangle .$$

The current is defined as $J^\mu =: \bar\psi\gamma^\mu\psi: \; =: \bar\psi_A\psi_a {:}\, \gamma^\mu_{Aa}$. Using again Wick's theorem for fermionic fields, one gets

$$\begin{aligned}\langle J_1^{\mu_1} J_2^{\mu_2} J_3^{\mu_3} J_4^{\mu_4}\rangle_{\text{Dirac}} &= \left\langle {:}\, \bar 1_A 1_a {:}\, {:}\, \bar 2_B 2_b {:}\, {:}\, \bar 3_C 3_c {:}\, {:}\, \bar 4_D 4_d {:}\, \right\rangle \gamma^{\mu_1}_{Aa}\gamma^{\mu_2}_{Bb}\gamma^{\mu_3}_{Cc}\gamma^{\mu_4}_{Dd} \\ &= (\left\langle \bar 1_A 3_c\right\rangle\left\langle 1_a\bar 3_C\right\rangle\left\langle \bar 2_B 4_d\right\rangle\left\langle 2_b\bar 4_D\right\rangle \pm (7 \text{ similar terms}) \\ &\quad + \left\langle \bar 1_A 2_b\right\rangle\left\langle 1_a\bar 2_B\right\rangle\left\langle \bar 3_C 4_d\right\rangle\left\langle 3_c\bar 4_D\right\rangle)\gamma^{\mu_1}_{Aa}\gamma^{\mu_2}_{Bb}\gamma^{\mu_3}_{Cc}\gamma^{\mu_4}_{Dd}.\end{aligned}$$

Again, the last term is the neglected vacuum contribution. Note that all terms give rise to a trace of either four or eight γ-matrices which can be calculated by the recursion formula for even n:

$$\operatorname{tr}(\gamma^{\mu_1}\dots\gamma^{\mu_n}) = \sum_{k=2}^{n}(-1)^k\eta^{\mu_1\mu_k}\operatorname{tr}(\gamma^{\mu_2}\dots\gamma^{\mu_{k-1}}\gamma^{\mu_{k+1}}\dots\gamma^{\mu_n}).$$

3 Reduction of n-point functions

The differential equations obtained in the last chapter are too complicated to be solved in 4D. We will instead use special differential operators to project n-point functions to contributions to the OPE of two fields, yielding $n-1$-point functions. All contributing lower correlation functions must then be positive on their own in sense of (2.1.2).

Furthermore, we restrict ourselves in the analysis of correlation functions to the modified functions by subtracting the most singular twist-0 vacuum contribution, as it can always be identified as the most singular contribution.

3.1 Intertwining property

This new approach on determining the partial waves in a correlation function is given in [6] and uses operators D_λ on vectors $\varphi_1\varphi_2\Omega$, which eliminate any contribution of representations $\lambda' \neq \lambda$, and thus within correlation functions

$$\iota^x_{x_1,x_2} \circ D_\lambda \varphi_1(x_1)\varphi_2(x_2)\Omega = \varphi_\lambda(x)\Omega, \tag{3.1.1}$$

where $\iota^x_{x_1,x_2} \equiv \iota^x$ denotes equating $x_1 = x_2 = x$. The operators are therefore similar to inserting a projection Π_λ. It is a very recent result [6] that such operators are differential operators for integer scaling dimensions as it is the case in GCI theories. In terms of representation theory, the statement (3.1.1) can be expressed as follows.

Definition 6. A linear operator $T : V_1 \to V_2$ is called *intertwining operator* for two representations $\alpha_1 : \mathcal{A} \to \mathrm{End}(V_1)$ and $\alpha_2 : \mathcal{A} \to \mathrm{End}(V_2)$ of an algebra $\mathcal{A}$, if for all $a \in \mathcal{A}$ holds

$$T \circ \alpha_1(a) = \alpha_2(a) \circ T,$$

thus it leaves the action of $\mathcal{A}$ invariant.

Postulating the intertwining property for D_λ allows to derive partial differential equations which the operators have to satisfy, as it implies that T commutes with

all generators X of $\mathcal{A}$ and thus $T[X,\varphi] = [X, T\varphi]$. Using this, one gets

$$\iota^x \circ D_\lambda \left\langle [X, \varphi_1\varphi_2]\varphi_\lambda \right\rangle = \left\langle [X, \iota^x \circ D_\lambda \varphi_1\varphi_2]\varphi_\lambda \right\rangle = \left\langle [X, \varphi_\lambda]\varphi_\lambda \right\rangle$$

for any generator X. The commutators are known and depend on the representations λ_1 of φ_1, λ_2 of φ_2 and λ of φ_λ , therefore we can write

$$i[X, \varphi_1\varphi_2] = (\Delta_{\lambda_1} + \Delta_{\lambda_2})\varphi_1\varphi_2,$$

where Δ_{λ_i} denotes the corresponding differential operator from eq. (2.3.4), acting only on φ_i.

3.2 Derivation of a partial differential equation for the operators

We are now using the quantum numbers $\lambda = (\kappa, L)$, as defined above. Thus we restrict ourselves to contributions of weights $j_1 = j_2 = \frac{L}{2}$, which corresponds to symmetric and traceless tensor fields. It is known that these are the only fields contributing to the OPE of two scalar fields [4]. As we will examine tensor fields, we will probably only see part of the contributions.

Demanding the intertwining property of D_λ

$$\iota^x \circ D_\lambda \circ (\Delta_{\lambda_1} + \Delta_{\lambda_2}) \left\langle \varphi_1\varphi_2\varphi_\lambda \right\rangle \stackrel{!}{=} \Delta_\lambda \circ \iota^x \circ D_\lambda \left\langle \varphi_1\varphi_2\varphi_\lambda \right\rangle$$

not just for this three-point function, but also for the operators, i.e.

$$\iota^x \circ D_\lambda \circ (\Delta_{\lambda_1} + \Delta_{\lambda_2}) \stackrel{!}{=} \Delta_\lambda \circ \iota^x \circ D_\lambda, \tag{3.2.1}$$

one can determine PDEs for the ansatz $D_\lambda = E_\lambda(v, y_1, y_2, w_1, w_2, x_1 + x_2) \circ \rho_{12}^n$. Here, $y_i \equiv \partial_i$ and $w_i \equiv \partial^{v_i}$, where v_i is the polarization vector of φ_i and v the polarization vector of φ_λ. n has to be chosen large enough to eliminate poles in x_{12} and make the correlation function regular; in the case of the exotic and the free Dirac structure, $n \geq 2$, the free Bose structure requires $n \geq 3$.

Additionally, there is no dependence on $x_1 - x_2$ because D_λ is always followed by the evaluation $x_1 = x_2$. Note that due to GCI $n \leq \frac{d_1+d_2}{2}$. Our derivation follows that of Wallenhorst in [14].

We start by commuting the factor ρ^n past $(\Delta_{\lambda_1} + \Delta_{\lambda_2})$ on the left hand side of (3.2.1) by using

$$[\Delta_\lambda(\partial_i), x_i] \circ P(x_i) = (\nabla_i \Delta_\lambda(\partial_i)) \circ P(x_i)$$

where ∇_i denotes the derivative w.r.t. $\frac{\partial}{\partial(\partial_i)}$ and $P(x_i)$ is any function depending on x_i.

1. For $X = P_\mu$ we obtain $\Delta_{\lambda_i} = \partial_{i,\mu}$, hence

$$\begin{aligned}
& \iota^x \circ E_\lambda \circ \rho_{12}^n \circ (\partial_{1,\mu} + \partial_{2,\mu}) \\
= \;& \iota^x \circ \Big(E_\lambda \circ (\partial_{1,\mu} + \partial_{2,\mu}) \circ \rho_{12}^n - E_\lambda \circ (2n\rho_{12}^{n-1} x_{12,\mu} - 2n\rho_{12}^{n-1} x_{12,\mu}) \Big) \\
= \;& \iota^x \circ ((\partial_{1,\mu} + \partial_{2,\mu}) \circ E_\lambda \circ \rho_{12}^n - (\partial_{1,\mu} E_\lambda + \partial_{2,\mu} E_\lambda) \circ \rho_{12}^n) \\
\stackrel{!}{=} \;& \partial_\mu \circ \iota^x \circ E_\lambda \circ \rho_{12}^n \\
\Rightarrow \;& (\partial_{1,\mu} E_\lambda + \partial_{2,\mu} E_\lambda) = 0.
\end{aligned}$$

   In the second line we commuted $(\partial_{1,\mu} + \partial_{2,\mu})$ past E_λ and in the third line we compare the LHS with the RHS of (3.2.1), exploiting that

$$\iota^x \circ (\partial_{1,\mu} + \partial_{2,\mu}) = \partial_{x,\mu} \circ \iota^x.$$

   We can conclude that E_λ does not depend on $x_1 + x_2$ thence neither on x_1 nor on x_2.

For further calculations, we need the following lemma:

Lemma. *Let $E(\partial_1, \ldots, \partial_n)$ be a (formal) power series in the derivatives $\partial_1, \ldots, \partial_n$ and $P(x_1, \ldots, x_n)$ a sufficiently nice behaved[1] power series in every $x_1, \ldots, x_n$. Then*

$$E(\partial_1, \ldots) \circ P(x_1, \ldots) = P(x_1 + \nabla_1, \ldots, x_n + \nabla_n) \circ E(\partial_1, \ldots) \tag{3.2.2}$$

where $\nabla_i = \frac{\partial}{\partial(\partial_i)}$ is the derivative w.r.t. the derivative ∂_i.

Proof. We choose w.l.o.g. $n = 1$, for greater n we could use the Taylor expansion in every variable separately. Let $E(\partial_1) = \sum_m b_m \partial_1^m$. Then

$$\begin{aligned}
E(\partial_1) \circ x_1 &= \sum_m b_m \partial_1^m \circ x_1 = \sum_m b_m \left(\partial_1^{m-1} x_1 \partial_1 + \partial_1^{m-1} \right) \\
&= \ldots = \sum_m b_m \left(x_1 \partial_1^m + m \partial_1^{m-1} \right)
\end{aligned}$$

[1] We ignore potential difficulties due to convergence.

$$= \quad x_1 \circ E(\partial_1) + \nabla_1 E(\partial_1).$$

Using this repeatedly yields

$$E(\partial_1) \circ x_1^m = \sum_{k=0}^{m} \binom{m}{k} x_1^k \nabla_1^{n-k} E = (x_1 + \nabla_1)^m E.$$

Hence writing $P(x_1) = \sum_m a_m x_1^m$, we can compute

$$\begin{aligned} E(\partial_1) \circ P(x_1) &= \sum_m a_m E \circ x_1^m \\ &= \sum_m a_m (x_1 + \nabla_1)^m E \\ &= P(x_1 + \nabla_1) \circ E. \end{aligned}$$

Note that for linear $P = a_1 x_1 + a_0$, the statement simplifies to

$$E \circ P(x_1) = P(x_1) E + a_1 \nabla_1 E.$$

□

2. If $X = D$ we have $\Delta_\lambda = (x \cdot \partial + d)$, yielding

$$\begin{aligned} & \iota^x \circ E_\lambda \circ \rho_{12}^n \circ (x_1 \cdot \partial_1 + x_2 \cdot \partial_2 + d_1 + d_2) \\ = \quad & \iota^x \circ E_\lambda \circ (x_1 \cdot \partial_1 + x_2 \cdot \partial_2 + d_1 + d_2 - 2n) \circ \rho_{12}^n \\ = \quad & \iota^x \circ ((x_1 \cdot \partial_1 + \partial_1 \cdot \nabla_1) + (x_2 \cdot \partial_2 + \partial_2 \cdot \nabla_2)) \circ E_\lambda \circ \rho_{12}^n \\ & + \iota^x \circ E_\lambda \circ (d_1 + d_2 - 2n) \rho_{12}^n \\ \stackrel{!}{=} \quad & (x \cdot \partial + d) \circ \iota^x \circ E_\lambda \circ \rho_{12}^n \\ \Rightarrow \quad & (\partial_1 \cdot \nabla_1 + \partial_2 \cdot \nabla_2) E_\lambda = (d + 2n - d_1 - d_2) E_\lambda, \end{aligned}$$

   where in the second line we used the lemma from above, respecting that the derivative ∇_i is acting on the partial derivative within the operator, not the polynomial. Thus, E_λ is homogeneous of degree $d + 2n - d_1 - d_2$ in ∂_1 and ∂_2.

3. If $X = M_{\mu\nu}$, the resulting condition on E_λ is

$$(\partial_1 \wedge \nabla_1 + \partial_2 \wedge \nabla_2 + v \wedge \partial^v + \partial^{v_1} \wedge \nabla_{v_1} + \partial^{v_2} \wedge \nabla_{v_2}) E_\lambda = 0,$$

where again $x \wedge y = x_\mu y_\nu - x_\nu y_\mu$. This implies that E_λ is either a Lorentz pseudo scalar or a Lorentz scalar, as free indices a_μ would exhibit nonzero contributions under application of $a \wedge \partial^a$.

4. To avoid confusion regarding the derivatives, we write y_i for ∂_i and w_i for ∂^{v_i} as variables of E_λ. Considering $X = K_\mu$ and combining the condition on E_λ with the previous results gives the final PDE [14]

$$[2(y_1 \cdot \nabla_1)\nabla_{1,\mu} - y_{1,\mu}\nabla_1^2 + 2\nabla_1^\nu(w_{1,\nu}\nabla_{v_1,\mu} - w_{1,\mu}\nabla_{v_1,\nu}) + 2(d_1 - n)\nabla_{1,\mu}]_{(1,2)} E_\lambda = 0, \tag{3.2.3}$$

where ∇_i means the derivative w.r.t. y_i, ∇_{v_i} the derivative w.r.t. w_i, d_i the scaling dimension of the field φ_i and $[\cdot]_{(i,j)}$ the symmetrization of $[\cdot]$ under exchange of i and j. Apart of solving eq. (3.2.3), E_λ must be homogeneous of degree L (the rank of φ_λ) in v to contract φ_λ completely, hom. of deg. L_i in w_i, in order to get the unpolarized field φ_i and hom. of deg. $2\kappa + L + 2n - d_1 - d_2 = d + 2n - d_1 - d_2$ in y_1 and y_2 simultaneously. Moreover, we consider only contributions of traceless tensor fields, which means that E_λ has to be harmonic in v. These homogeneities add up to an even number if the twist 2κ is even, as it is required by GCI. Otherwise, the constituents of the operator could not be contracted to form a scalar or pseudo scalar.

To summarize: if we find an admissible differential operator E_λ that complies with these restrictions, we can reduce the n-point function $\langle \varphi_1 \varphi_2 \varphi_3 \dots \varphi_n \rangle$ to the contributing $n-1$-point function $\langle \varphi_\lambda \varphi_3 \dots \varphi_n \rangle$ by applying $E_\lambda \circ \rho_{12}^n$ and substituting $x_1 = x_2 = x$, i.e.

$$\langle \varphi_\lambda \varphi_3 \dots \varphi_n \rangle = \iota_{x_1,x_2}^x \circ E_\lambda \circ \rho_{12}^n \langle \varphi_1 \varphi_2 \varphi_3 \dots \varphi_n \rangle .$$

3.3 Positivity tests

Finally we summarize the possible positivity tests and how we will apply them by briefly outlining the procedure to analyze the exotic four-point structure.

1. Pick a representation (κ, L) and compute the intertwining operator $E^{(\kappa,L)}$ by writing down an ansatz consisting of all possible contractions of vectors y_i, w_i and v according to the homogeneity conditions and solving the PDE.

2. Reduce $\langle J(x_1)J(x_2)J(x_3)J(x_4)\rangle_{\text{exotic}}$ to the contribution $\left\langle \phi^{(\kappa,L)}(x_1)J(x_3)J(x_4)\right\rangle$ by applying the operator and setting $x_1 := x_1 = x_2$.

3. Check for pole bound violations (eq. (2.4.5)).

4. If $\kappa = 1$ (twist-2), check for violation of conservation.

5. Reduce further to $\left\langle \phi^{(\kappa,L)}(x_1)\phi^{(\kappa,L)}(x_4)\right\rangle$, check the sign.

If the sign in the last step turns out to be negative for all representations, the four-point function $-\langle JJJJ\rangle$ is positive.

If the sign is indefinite, the four-point structure has to be combined with multiples of the free Bose and Dirac structures to derive restrictions on these coefficients.

4 Development and description of used CAS tools

Writing down an ansatz and solving the PDE proves to be challenging even for small quantum numbers κ and L. We therefore automated the process to form a software library for the computer algebra system software *Maple*[1]. The software consists of two parts; a tool to derive the intertwining operators for given source and target representations and to translate the result into a valid operator to be used in the second part, which does the actual reduction.

We will first describe the used techniques of each component and then give examples on how to load and use the library in *Maple*. The whole package can be found at `http://www.theorie.physik.uni-goettingen.de/~nikolai.wyderka`.

4.1 Operator derivation

To avoid differentiations with indices, we rearrange the terms of the differential equation (3.2.3) such that operations with the uncontracted index μ are conducted last by commuting them to the left:

$$\begin{aligned}[2(\nabla_{1,\mu}(y_1 \cdot \nabla_1) - \nabla_{1,\mu}) - y_{1,\mu}\nabla_1^2 + 2\nabla_{v_1,\mu}(w_1 \cdot \nabla_1) \\ -\nabla_{1,\mu} - 2w_{1,\mu}(\nabla_1 \cdot \nabla_{v_1})) + 2(d_1 - n)\nabla_{1,\mu}]_{(1,2)} E_\lambda &= 0.\end{aligned}$$

Within *Maple*, we will represent contracted terms like $y_i \cdot y_j$ in the ansatz by the variable $yy[i,j]$ and similarly combinations of the variables v_i, w_i and y_i, therefore, starting with an ansatz $E(vv, vw, vy, ww, wy, yy)$, we can write the differential

[1] `http://www.maplesoft.com`

operator $y_1 \cdot \nabla_1$ in the following way:

$$\begin{aligned} y_a \cdot \nabla_a \circ E &= \sum_{i=1}^{m} vy[i,a]\partial_{vy[i,a]}E + \sum_{i=1}^{m} wy[i,a]\partial_{wy[i,a]}E \\ &\quad + \sum_{i=1}^{m} yy[i,a]\partial_{yy[i,a]}E + \sum_{i=1}^{m} yy[a,i]\partial_{yy[a,i]}E. \end{aligned} \tag{4.1.1}$$

Here, m is the number of affected space time variables; $m = 2$ for scalar intertwiners reducing fields $\varphi(x_1)$ and $\varphi(x_2)$ and $m = 4$ for pseudo scalar intertwiners reducing in all four fields simultaneously due to restrictions covered in 4.2.2.

For higher derivatives and easy identification of equal terms it is convenient to introduce the following arrangement convention: all occurring factors $vv[i,j]$, $ww[i,j]$ and $yy[i,j]$ must obey $i \leq j$. We can then write for $a = 1, 2$

$$\begin{aligned} \nabla_a \cdot \nabla_{v_a} \circ E &= 4\partial_{wy[a,a]}E + \sum_{i=1}^{m}\sum_{j=1}^{m} vw[j,i]\partial_{ww[a,i]}\partial_{vy[j,a]}E \\ &= +\sum_{i=1}^{m}\sum_{j=1}^{m} vw[j,i]\partial_{ww[i,a]}\partial_{vy[j,a]}E \\ &= +\sum_{i=1}^{m}\sum_{j=1}^{m} ww[i,j]\partial_{ww[a,i]}\partial_{wy[j,a]}E + \dots. \end{aligned}$$

After applying such operators, the arrangement convention has to be restored.

Similarly, the derivatives $y_i \cdot \nabla_i$, ∇_i^2, $w_i \cdot \nabla_i$ have to be defined. The set of single derivatives involving indices μ adds a pseudo index μ to the expression which is only relevant to identify linearly dependent terms. The full expressions for these derivatives are listed within the sources in appendix B.2.

The resulting set of conditions on the coefficients of E making the result vanish has then to be extracted and solved. This is done by an external[2] script to which we will refer to as *solver*. It takes the result of the PDE applied to an operator and forms the coefficients of linearly independent terms to a set of equations that are solvable by *Maple*. The resulting conditions are then applied to the ansatz.

In order to get trace free operators, the same process has to be repeated to make the ansatz solve

$$\partial_{v_1}^2 E = 0.$$

[2]We distinguish between internal scripts which are written in the *Maple* script language and external scripts which are executed by *Maple* but written in *Python* as *Maple* lacks important programming features like regular expressions and efficient string manipulation.

The function *buildop* located in the file *diffsolver.mpl* generates the ansatz, solves the PDE and returns the intertwining operator in a raw form. In order to generate the ansatz, it uses the tool described in the next chapter.

4.2 Generating the ansatz

4.2.1 Scalar intertwiners

Writing down an ansatz is applicable for small quantum numbers but tedious for large numbers that give rise to hundreds of possible combinations of the involved differentiations and polarization vectors. Consider for example the scalar operator $E_{JJ}^{T_2}$ in the case $n = 2$ for a reduction of two currents ($(\kappa_{1,2}, L_{1,2}) = (1, 1)$) to the contribution of the twist 2 tensor of $(\kappa, L) = (1, 2)$. According to section 3.2, $E_{JJ}^{T_2}$ consists of two v_1, one w_1, one w_2 and $2\kappa + L + 2n - d_1 - d_2 = 2$ derivatives y_i. The ansatz for $E_{JJ}^{T_2}$ would then be

$$\begin{aligned} E_{JJ}^{T_2} &= A_{i,j}(v_1 \cdot v_1)(w_1 \cdot w_2)(y_i \cdot y_j) + B_{i,j}(v_1 \cdot w_1)(v_1 \cdot w_2)(y_i \cdot y_j) \\ &+ C_{i,j}(v_1 \cdot v_1)(w_1 \cdot y_i)(w_2 \cdot y_j) + D_{i,j}(v_1 \cdot w_1)(v_1 \cdot y_i)(w_2 \cdot y_j) \\ &+ E_{i,j}(v_1 \cdot w_2)(v_1 \cdot y_i)(w_1 \cdot y_j) + F_{i,j}(w_1 \cdot w_2)(v_1 \cdot y_i)(v_1 \cdot y_j) \end{aligned}$$

where $i, j \in \{1, 2\}$ are summed over, skipping identical terms as $(y_i \cdot y_j) = (y_j \cdot y_i)$ and similar identities, giving a total of 22 terms. To automate the generation of this ansatz, we use the following procedure to combine a list of vectors to all possible contractions. In the case above, the initial list of vectors would consist of v_1, v_1, w_1, w_2, y, y. The assignment of the derivation targets 1 and 2 to the y vectors will be conducted later. We then take the first entry and iterate over all possibilities to contract it with the latter entries. For each such contraction[3], we clone the list, removing the two contraction partners and contract the rest recursively. The result is sorted by some arbitrary condition to identify equal terms.

Finally, we assign derivation targets to the vectors y and assign a coefficient to each term. The relevant algorithms can be found in the file *combinator.py*.

[3]By convention, a contraction $w_i v_j$ for example will be written as $v_j w_i$ and similarly for $y_i v_j$ and $y_i w_j$, and the arrangement condition from above applies.

4.2.2 Pseudo scalar intertwiners

The generation of pseudo scalar intertwiners is similar to that of the scalar intertwiners, except for an additional ε-tensor in each term of the ansatz, contracting four of the vectors. It has to be considered that the contraction of two identical vectors with the full antisymmetric tensor gives zero and thus only different vectors may be contracted.

To avoid the calculation with ε-terms and the resulting uncontracted vectors, we restrict ourselves in the case of pseudo scalars to combined reductions in both variable pairs x_1, x_2 and x_3, x_4 simultaneously by multiplying both operators and contract the product of the two asymmetric tensors by using the identity

$$\varepsilon^{\mu_1\mu_2\mu_3\mu_4} \cdot \varepsilon^{\nu_1\nu_2\nu_3\nu_4} = -\begin{vmatrix} \eta^{\mu_1\nu_1} & \dots & \eta^{\mu_1\nu_4} \\ \vdots & \ddots & \vdots \\ \eta^{\mu_4\nu_1} & \dots & \eta^{\mu_4\nu_4} \end{vmatrix}. \tag{4.2.1}$$

The minus sign is due to the Minkowski metric.

The resulting expression of Lorentz vectors and η-terms can be contracted using the algorithm in *pacman.py*. The coefficients of this pseudo scalar are then determined to solve the PDE in both pairs x_1, x_2 and x_3, x_4 and to be traceless in v_1 and v_2.

The ansatz is generated by the script *combinator2.py*.

4.3 Representation and differentiation of the correlation functions

All correlation functions can be written in contracted form using the contractions $v_i \cdot v_j$, $v_i \cdot x_{ij}$ and ρ_{ij} where $x_{ij} = x_i - x_j$. Therefore, occurrences of $x_{ij} \cdot x_{kl}$ are rewritten in terms of ρ by using

$$\rho_{ij} = x_{ij}^2 = x_i^2 + x_j^2 + 2x_i \cdot x_j$$

and

$$x_{ij} \cdot x_{kl} = x_i \cdot x_k - x_i \cdot x_l - x_j \cdot x_k - x_j \cdot x_l = \frac{1}{2}(\rho_{il} + \rho_{jk} - \rho_{ik} - \rho_{jl}).$$

The intertwining operator derived beforehand can then be written similarly to (4.1.1), $v_c \cdot y_i$ translates to

$$\begin{aligned} v_c \cdot y_i \circ F(vv, vx, \rho) &= \sum_{k=1}^{m} \left(\sum_{j=i+1}^{n} vv[k,c] \cdot \partial_{vx[k,i,j]} - \sum_{j=1}^{i-1} vv[k,c] \cdot \partial_{vx[k,j,i]} \right) \\ &\quad + 2 \sum_{j=i+1}^{n} vx[c,i,j] \cdot \partial_{\rho[i,j]} - 2 \sum_{j=1}^{i-1} vx[c,j,i] \cdot \partial_{\rho[j,i]}. \end{aligned}$$

For technical reasons, vector products $a_i \cdot b_j$ are again written as $ab[i,j]$. The full set of derivatives and how they translate to *Maple* syntax can be found in appendix B.3.

The functions *translate_op_12*, *translate_op_34* and *translate_op_56* in the file *diffsolver.mpl* take the result of a *buildop* call and return the corresponding applicable operator which then can be used to reduce the correlation functions.

To avoid problems due to derivatives w.r.t. v_1 and v_2 within the operator, the polarization vector v of the target field is given a higher index, which can be controlled by the second parameter of the *translate_op* functions. A derivation of a four-point correlation function would therefore require the assignment of v_5 to a target field.

After any reduction, we evaluate the involved spacetime points x_i and x_j at equal position $x_i = x_j$. This implies that we are left in the four-point case reducing in x_1 and x_2 case with an expression where the polarization vector v_5 is associated with the position x_1.

As linear dependencies among the x_{ij} are not known to *Maple* and might therefore disguise vanishing contributions, we test this case by rewriting them as $x_i - x_j$ using another function.

4.4 Documentation and manual

Due to lack of space, the documentation of the software together with a listing of the *Maple* functions and the external scripts written can be found in appendix A. A manual of usage, as well as the scripts themselves, can be found at `http://www.theorie.physik.uni-goettingen.de/~nikolai.wyderka`.

5 Applications and results

Having collected the necessary tools we will now apply them to the exotic four-point structure and test for positivity. We will show that the structure is not positive on its own, but can be made positive by particular combinations of the exotic structure with the free bosonic and fermionic four-point functions.

We begin by giving a detailed example of the reduction of the exotic function to show the general scheme of our proceeding. We will then state general results derived for the exotic correlation function and proceed to the calculation of two-point contributions to the free and exotic structures.

Unless stated otherwise, we are using only scalar operators. To get further restrictions, we will later consider certain pseudo scalar contributions.

5.1 Treating the exotic structure

While the form of the exotic structure given in (2.7.1) is rather compact, it is not very useful for CAS calculations due to the four ε-tensors. One can either express them as $24 \cdot 24$ η-products by (4.2.1) and contract them, giving rise to an expression of a total of 9216 terms as a function of $v_i \cdot v_j$, $v_i \cdot x_{ij}$ and ρ_{ij} as demanded above. Another possibility is the usage of the compact form and operators for single derivatives, with space time indices like $\partial_{1,\mu}$. The advantage is that they can be written in a very compact form and the calculations are a lot faster, while the resulting expression consist of many more terms, which do require much more space in the computer's memory.

However, it turned out that memory size rather than computation time is the main concern, and therefore we did indeed use the contracted, 9216 terms long expression for further calculations. The other framework has been used to cross check some of the results we obtained, but has been omitted from the package linked above.

5.2 Contributions of representations $\kappa = 3$, $L = 0$ and $\kappa = 4$, $L = 0$

To calculate the contribution of a field $\phi^{(3,0)}$ of representation $(\kappa, L) = (3, 0)$ to STANEV's function, we start by determining the operator $E_{JJ}^{(3,0)}$ for $n = 2$, as the highest pole is of degree 2. The ansatz, which we can generate painlessly using our automated ansatz generator, consists of all possible contractions of four differentiations ∂_i and the differentiations ∂_1^v and ∂_2^v, according to section 3.2. Remember that these differentiations will be written as y_i and w_i respectively. The ansatz gives

$$E_{JJ}^{(3,0)} = A_{i,j,k,l} \cdot (y_i \cdot y_j)(y_k \cdot y_l)(w_1 \cdot w_2) + B_{i,j,k,l} \cdot (y_i \cdot y_j)(w_1 \cdot y_k)(w_2 \cdot y_l)$$

where $i, j, k, l \in \{1, 2\}$ are again summed over skipping identical terms, giving 18 different summands.

Applying the PDE yields many restrictions, leaving only three independent coefficients A, B and C:

$$\begin{aligned}
E_{JJ}^{(3,0)} \;=\; & A\Big((y_1 \cdot y_1)^2(w_1 \cdot w_2) + 2(y_1 \cdot y_1)(y_1 \cdot y_2)(w_1 \cdot w_2) \\
& -8(y_1 \cdot y_1)(y_2 \cdot y_2)(w_1 \cdot w_2) - 4(y_1 \cdot y_1)(w_1 \cdot y_1)(w_2 \cdot y_1) \\
& -6(y_1 \cdot y_1)(w_1 \cdot y_1)(w_2 \cdot y_2) - 6(y_1 \cdot y_1)(w_1 \cdot y_2)(w_2 \cdot y_1) \\
& +4(y_1 \cdot y_2)(w_1 \cdot y_1)(w_2 \cdot y_1) + 8(y_2 \cdot y_2)(w_1 \cdot y_1)(w_2 \cdot y_1)\Big) \\
& +B\,(-8(y_1 \cdot y_1)(y_2 \cdot y_2)(w_1 \cdot w_2) + 2(y_1 \cdot y_2)(y_2 \cdot y_2)(w_1 \cdot w_2) \\
& +(y_2 \cdot y_2)^2(w_1 \cdot w_2) + 8(y_1 \cdot y_1)(w_1 \cdot y_2)(w_2 \cdot y_2) \\
& +4(y_1 \cdot y_2)(w_1 \cdot y_2)(w_2 \cdot y_2) - 6(y_2 \cdot y_2)(w_1 \cdot y_1)(w_2 \cdot y_2) \\
& -6(y_2 \cdot y_2)(w_1 \cdot y_1)(w_2 \cdot y_2) - 6(y_2 \cdot y_2)(w_1 \cdot y_2)(w_2 \cdot y_1) \\
& -4(y_2 \cdot y_2)(w_1 \cdot y_2)(w_2 \cdot y_2)) \\
& +C\,((y_1 \cdot y_1)(y_2 \cdot y_2)(w_1 \cdot w_2) - (y_1 \cdot y_1)(w_1 \cdot y_2)(w_2 \cdot y_2) \\
& +(y_1 \cdot y_2)(w_1 \cdot y_1)(w_2 \cdot y_2) - (y_2 \cdot y_2)(w_1 \cdot y_1)(w_2 \cdot y_1))\,.
\end{aligned}$$

Note that due to the symmetry of the PDE, each operator can be written as a linear combination of a set of (in this case three) symmetric and antisymmetric operators under exchange of $y_1 \leftrightarrow y_2$, $w_1 \leftrightarrow w_2$. As the exotic structure is invariant under exchange of any two spacetime points, antisymmetric operators applied to it vanish.

A classification of the solution space of these operators is therefore useful but hard, especially for operators with a greater solution space. We therefore did not conduct the classification. Nevertheless, the number of symmetric constituents can later be recovered from the number of non-vanishing eigenvalues in the coefficient matrix.

Applying this operator to the exotic four-point structure and setting $x_1 := x_1 = x_2$ yields a long and difficult three-point contribution, where the denominator is

$$\left\langle \phi_1^{(3,0)} J_3 J_4 \right\rangle \propto \frac{1}{\rho_{13}^4 \rho_{14}^4 \rho_{34}^2},$$

thus the pole bounds (2.4.5) are not violated. Further reduction in x_3 and x_4 and renaming the coefficients to A', B' and C' yields the two-point contribution

$$\left\langle \phi_1^{(3,0)} \phi_4^{(3,0)} \right\rangle \propto -\frac{1}{\rho_{14}^6}(8A + 8B - C)(8A' + 8B' - C').$$

The resulting coefficient matrix

$$\begin{pmatrix} 64 & 64 & -8 \\ 64 & 64 & -8 \\ -8 & -8 & 1 \end{pmatrix}$$

is positive semi definite with the only non vanishing eigenvalue $\left\| \begin{pmatrix} 8 & 8 & -1 \end{pmatrix} \right\|^2$ due to the fact that the coefficients of both applications factorize as it is the case.[1] The minus sign, however, requires that the structure is either indefinite or negative. If the latter would be the case, we could still use the negation of the structure as a candidate for a positive theory.

However, doing the same derivations for $(\kappa, L) = (4, 0)$ yields a two parametric operator and thus

$$\left\langle \phi_1^{(4,0)} J_3 J_4 \right\rangle \propto \frac{1}{\rho_{13}^5 \rho_{14}^5 \rho_{34}^2},$$

where again no pole bound violation occurs and the two-point function is the positive function

$$\left\langle \phi_1^{(4,0)} \phi_4^{(4,0)} \right\rangle \propto \frac{1}{\rho_{14}^8} AA'.$$

It is therefore not sufficient to change the sign.

[1] A proof for this uses basic linear algebra.

5.3 Combining the exotic with the free bosonic and fermionic structure

We now switch to $n = 3$ as the most divergent pole in the free bosonic structure is of degree 3. There are two strategies to compare the results of $n = 3$-derivations with those of a $n = 2$-operator; one can either use lemma (3.2.2) and apply $(\nabla_1 - \nabla_2)^2$ to the $n = 3$-operator. Comparing the coefficients gives then relations between them that allow to compare results obtained with both operators.

However, as computation time of the considered representations is rather low compared to the memory consumption, we will use the other possibility and use $n = 3$-operators also for the free Dirac currents as well as the exotic structure. For some higher representations, where computations were only possible with $n = 2$, we switched back to $n = 2$ to check solely for pole bound violations and violations of conservation, but did not compare to the free Bose structure.

Additionally, representations of $\kappa = 0$ have not been considered, as they lie below the unitarity bound. We begin by summarizing some general results obtained during the process:

- All regarded twist-2 contributions to the exotic and the Dirac correlation function are identical on the level of two-point functions up to a factor of 2^7. We rescale the exotic structure by this factor in future computations. Albeit, on the three point level these contributions were not the same.
- No violation of conservation of these three-point functions has been observed.
- No pole bound violation could be found.
- There are no contributions of odd scaling dimension to any of the three structures in all considered representations.
- Whenever the two-point function vanished, also the intermediate three-point contribution had been zero. A different result here would indicate a violation of positivity, too.

As a consequence of these results we concentrate on restrictions on the coefficients of the expression

$$\pm 2^7 \langle J_1 J_2 J_3 J_4 \rangle_{\text{exotic}} + A_D \langle J_1 J_2 J_3 J_4 \rangle_{\text{Dirac}} + A_B \langle J_1 J_2 J_3 J_4 \rangle_{\text{Bose}}$$

by examining the signs of the two-point functions. As the exotic structure is indefinite, we have to work with both assumptions for its sign.

The following tables show the eigenvalues of the coefficient matrices of the different representations, sorted by twist-2κ. Whenever there appears just one eigenvalue, we write the coefficient matrix as a product of the corresponding eigenvector. If there are more, we list the eigenvalues as a diagonalized matrix. The expression $(\kappa, \{L_1, L_2, \ldots\})$ is to be understood as summarized results for representations (κ, L_1), $(\kappa, L_2), \ldots$. Common prefactors have been neglected.

Representation (κ, L)	$(1,1)$	$(1,\{3,5,7\})$	$(1,2)$	$(1,4)$	$(1,6)$
Solution space dim.	4	5	5	5	5
$2^7\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{exotic}}$	0	0	$2^4\begin{pmatrix}56\\42\\19\\-12\end{pmatrix}\cdot(\dots)^\top$	$2^4\begin{pmatrix}564\\117\\-324\\94\\540\end{pmatrix}(\dots)^\top$	$2^2\begin{pmatrix}840\\-2560\\120\\-61\\1264\end{pmatrix}(\dots)^\top$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Dirac}}$	0	0	$2^4\begin{pmatrix}56\\42\\19\\-12\end{pmatrix}\cdot(\dots)^\top$	$2^4\begin{pmatrix}564\\117\\-324\\94\\540\end{pmatrix}(\dots)^\top$	$2^2\begin{pmatrix}840\\-2560\\120\\-61\\1264\end{pmatrix}(\dots)^\top$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Bose}}$	0	0	$3\begin{pmatrix}56\\42\\19\\-12\end{pmatrix}\cdot(\dots)^\top$	$5\begin{pmatrix}102\\-9\\-324\\17\\-216\end{pmatrix}(\dots)^\top$	$3\cdot 7\begin{pmatrix}300\\-640\\30\\-19\\496\end{pmatrix}(\dots)^\top$
Conditions (+-case)	-	-	$A_D > \frac{1}{16}(-16-3A_B)$	$A_B > 0 \wedge A_D > -1$	$A_B > 0 \wedge A_D > -1$
Conditions (−-case)	-	-	$A_D > \frac{1}{16}(16-3A_B)$	$A_B > 0 \wedge A_D > 1$	$A_B > 0 \wedge A_D > 1$

Table 5.1: Contributions of twist-2 to the four-point structures and resulting conditions.

Representation (κ, L)	$(2,0)$	$(2,\{1,3,5\})$	$(2,2)$	$(2,4)$
Solution space dim.	3	$\{6,8,8\}$	8	8
$2^7\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{exotic}}$	0	0	$2^3U^{-1}\text{diag}(1417135 \pm \sqrt{2047730865265}, 0, \dots)U$	$2^7U^{-1}\text{diag}(45018143 \pm \sqrt{2354911856676754}, 0, \dots)U$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Dirac}}$	0	0	$2^4 \begin{pmatrix} 154 \\ -1032 \\ 420 \\ 22 \\ -12 \\ -86 \\ 21 \\ 35 \end{pmatrix} (\dots)^\top$	$2^3 \cdot 3 \cdot 19 \begin{pmatrix} 1792 \\ 272 \\ -7616 \\ -2520 \\ -1120 \\ 672 \\ -525 \\ 84 \end{pmatrix} (\dots)^\top$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Bose}}$	0	0	$\begin{pmatrix} 154 \\ -1032 \\ 420 \\ 22 \\ -12 \\ -86 \\ 21 \\ 35 \end{pmatrix} (\dots)^\top$	$3^2 \cdot 5 \begin{pmatrix} 1792 \\ 272 \\ -7616 \\ -2520 \\ -1120 \\ 672 \\ -525 \\ 84 \end{pmatrix} (\dots)^\top$
Conditions (+-case)	-	-	$A_D > \frac{1}{16}(32 - 3A_B)$	$A_D > \frac{1}{152}(16 - 15A_B)$
Conditions (−-case)	-	-	Not satisfiable	Not satisfiable

Table 5.2: Contributions of twist-4 to the four-point structures and resulting conditions.

Representation (κ, L)	$(3,0)$	$(3,\{1,3\})$	$(3,2)$
Solution space dim.	4	8	9
$2^7\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{exotic}}$	$-2^{31}\cdot 3^4\begin{pmatrix}2\\3\\3\\2\end{pmatrix}(\dots)^\top$	0	$2^2\cdot 5\cdot U^{-1}\text{diag}(857628096803\pm\sqrt{735526214821012495865149},0,\dots)U$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Dirac}}$	$2^{30}\cdot 3^5\begin{pmatrix}2\\3\\3\\2\end{pmatrix}(\dots)^\top$	0	$2^2U^{-1}\text{diag}(24360385918238\pm\sqrt{611816502453968665129512869},0,\dots)U$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Bose}}$	$2^{26}\cdot 3^5\begin{pmatrix}2\\3\\3\\2\end{pmatrix}(\dots)^\top$	0	$2U^{-1}\text{diag}(1040616284717\pm\sqrt{746355033556650246523941 14},0,\dots)U$
Conditions (+-case)	$A_D > \frac{1}{48}(32-3A_B)$	-	$A_D > \frac{1}{1744}\left(332-89A_B+\sqrt{472976-67816A_B+2689A_B^2}\right)$
Conditions (−-case)	$A_D > \frac{1}{48}(-32-3A_B)$	-	$A_D > \frac{1}{1744}\left(-332-89A_B+\sqrt{472976+67816A_B+2689A_B^2}\right)$

Table 5.3: Contributions of twist-6 to the four-point structures and resulting conditions.

Representation (κ, L)	$(4,0)$	$(4,1)$	$(4,2)$
Solution space dim.	3	5	6
$2^7\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{exotic}}$	$2^{42}\cdot 3\cdot 5\cdot 7$	0	$2^6U^{-1}\text{diag}(-3065047283\pm 2\sqrt{2381535705795974291},0,\ldots)U$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Dirac}}$	$2^{42}\cdot 3\cdot 5\cdot 7$	0	$2^4\cdot 5U^{-1}\text{diag}(13714412287\pm\sqrt{18714914032330875964 9},0,\ldots)U$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Bose}}$	$2^{40}\cdot 3^2\cdot 5$	0	$3\cdot 5\cdot U^{-1}\text{diag}(7798161087\pm\sqrt{6068590435126675148 9},0,\ldots)U$
Conditions (+-case)	$A_D>\frac{1}{28}(-28-3A_B)$	-	$A_D>\frac{1}{291360}\left(-36844-21975A_B+\sqrt{8998104976-1623105960A_B+83009025A_B^2}\right)$
Conditions (−-case)	$A_D>\frac{1}{28}(28-3A_B)$	-	$A_D>\frac{1}{291360}\left(36844-21975A_B+\sqrt{8998104976+1623105960A_B+83009025A_B^2}\right)$

Representation (κ, L)	$(5,0)$	$(6,0)$	$(7,0)$
Solution space dim.	2	2	2
$2^7\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{exotic}}$	$-2^3\cdot 7$	$2^2\cdot 5\cdot 23$	$-2^5\cdot 17$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Dirac}}$	$2^4\cdot 3\cdot 7$	$2^2\cdot 3\cdot 5\cdot 13$	$2^4\cdot 71$
$\langle\phi^{(\kappa,L)}\phi^{(\kappa,L)}\rangle_{\text{Bose}}$	3^2	$3\cdot 5^2$	$3\cdot 13$
Conditions (+-case)	$A_D>\frac{1}{336}(224-9A_B)$	$A_D>\frac{1}{156}(-92-15A_B)$	$A_D>\frac{1}{1136}(544-39A_B)$
Conditions (−-case)	$A_D>\frac{1}{336}(-224-9A_B)$	$A_D>\frac{1}{156}(92-15A_B)$	$A_D>\frac{1}{1136}(-544-39A_B)$

Table 5.4: Contributions of twist-8 and higher twists to the four-point structures and resulting conditions.

Representation (κ, L)	$(1,1)$	$(\{2,3,4,5,6\},0)$	$(2,1)$	$(1,2)^\star$
Solution space dim.	1	0	2	3
$2^7 \langle \phi^{(\kappa,L)} \phi^{(\kappa,L)} \rangle_{\text{exotic}}$	0	-	$\begin{pmatrix} 1 \\ -1 \end{pmatrix} \begin{pmatrix} 1 & -1 \end{pmatrix}$	0
$\langle \phi^{(\kappa,L)} \phi^{(\kappa,L)} \rangle_{\text{Dirac}}$	0	-	0	0
$\langle \phi^{(\kappa,L)} \phi^{(\kappa,L)} \rangle_{\text{Bose}}$	0	-	0	-
Conditions (+-case)	-	-	-	
Conditions (−-case)	-	-	Not satisfiable	

Table 5.5: Contributions of various pseudo scalars to the four-point structures and resulting conditions. Contributions marked with $\star$ have been calculated using $n = 2$ and therefore can not be compared with the free Bose field.

5.4 Conclusions

Due to the restrictions, we have to reject the assumption that the exotic structure might have a negative sign.

For the positive case, fig. 5.1 shows the excluded area in the configuration space of A_D and A_B. Considering more representations would possibly give more restrictions. However, as the conditions seem to get weaker with higher quantum numbers, it seems unlikely to get a manifest violation of positivity pursuing this method. Nevertheless, further investigation of the pseudo scalar contributions might give more interesting insights but, however, proves to be complicated due to the resource intensive process of deriving higher operators.

It is rather interesting that none of the structures has contributions of fields of odd scaling dimension.

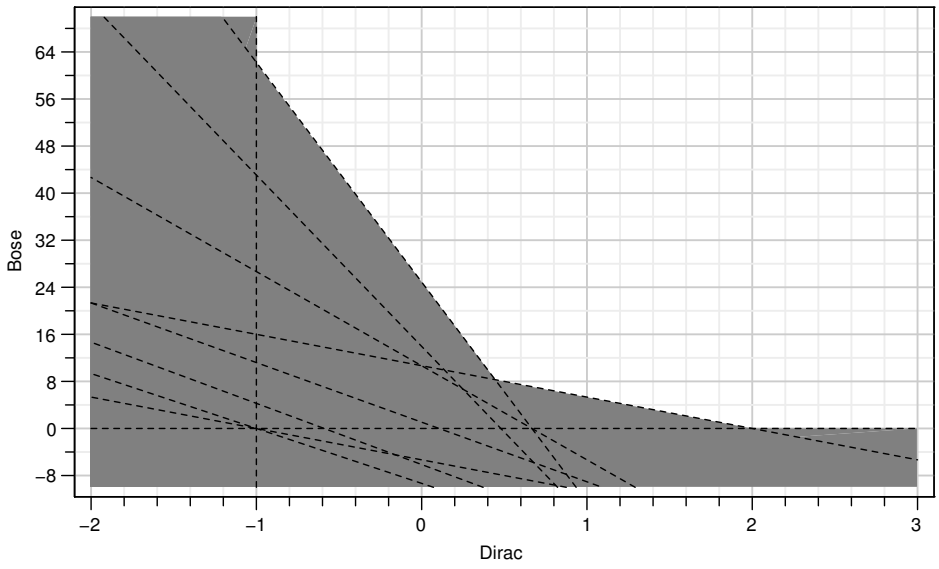

Figure 5.1: The allowed (white) and rejected (gray) combinations of coefficients A_D (Dirac) and A_B (Bose) in $2^7 \langle J_1 J_2 J_3 J_4 \rangle_{\text{exotic}} + A_D \langle J_1 J_2 J_3 J_4 \rangle_{\text{Dirac}} + A_B \langle J_1 J_2 J_3 J_4 \rangle_{\text{Bose}}$.

6 Summary and outlook

The positivity of correlation functions is, as we have seen, a nontrivial and interesting aspect of quantum field theories in the axiomatic framework. Especially in conformal invariant theories, where all correlation functions are rational, the description of field theories through the correlation functions rather than the fields, feasible through the reconstruction theorem, provides insight to admissible theories that fulfill the Wightman axioms. However, until today, no non free theory could be constructed within this setting, mainly due to the fact that positivity is hard to ensure.

We summarized the implications of global conformal invariance on the theories, especially on the correlation functions, and described the partial wave expansion as a tool to investigate positivity by decomposition of n-point correlation functions into two-point correlation functions, where positivity is trivial to verify. We presented an approach using Casimir operators to carry out these decompositions but concluded that it is too complicated in four-dimensional spacetime. Instead, we introduced and elucidated the recently developed method of intertwining operators.

We then developed the necessary CAS tools to apply that procedure to an exotic four-point correlation function found by Yassen Stanev which cannot occur in free field theories and is therefore of great interest. Using intertwining operators we finally showed that this structure is not positive on its own, but can be combined with certain combinations of the free Bose and Dirac structures to form a possibly positive theory. Particularly, the sign of the exotic structure could be fixed to be positive. However, as only a finite set of contributing representations was considered, it is still unknown whether there is a combination of the three structures that is positive.

Taking into account more representations, especially using pseudo scalar operators, might reveal more information. Additionally, we did not calculate contributions of non-symmetric tensor fields, as there has not yet been derived a PDE for such operators, which could be done in future work.

A great issue of using CAS systems to derive operators for higher quantum numbers is the exponentially increasing complexity, consuming both computer memory and CPU time. As a consequence, we could not derive non zero pseudo scalar operators of representation other than $(\kappa, L) = (1, 1)$ and $(\kappa, L) = (2, 1)$.

As all calculations involve only derivations of polynomials, it would be achievable to carry out these calculations completely writing custom software using regular expressions rather than using the software *Maple*. This would presumably increase computation speed and reduce memory usage significantly.

Sticking to CAS software, packages for treating ε-tensors could be used to derive pseudo scalar operators for single rather than simultaneous reductions to also verify pole bounds in these calculations.

Another possibility is to consider multiple bosonic and fermionic free fields, resulting in more parameters to derive restrictions for and would allow to give bounds on the field content of a theory involving the exotic correlation function.

A Documentation

This section briefly describes the different tools written for this thesis.

A.1 *Maple* scripts

A.1.1 utils.mpl

The file *utils.mpl* is included by any other *Maple* file and increases the stack limit and creates necessary temporary folders which are used by other parts of the system. The incrementation of the stack limit is crucial to allow collecting for a great amount of variables and a low stack size limits the number of recursive function calls. No subfunctions are defined within this file.

A.1.2 diffsolver.mpl

diffsovler.mpl contains all functions to derive scalar and pseudo scalar intertwining operators. The important functions are

buildop(kappa1,L1,kappa2,L2,n,kappa,L) Generates the raw version of intertwining operator E to reduce two fields of representations (κ_1, L_1) and (κ_2, L_2) to (κ, L) within a correlation function of maximum pole degree n.

buildpseudoop(kappa1,L1,kappa2,L2,n,kappa,L) Same as *buildop*, but for pseudo scalars. Note that *buildpseudoop* generates the raw version of a combined intertwiner for two reductions at once.

translate_op_12(F,voffset) Takes as argument F the raw version of an operator and translates it to an actual operator which acts on x_1 and x_2. To use the operator, the file *diffops.mpl* must be loaded. *voffset* denotes the index minus one(!) of the polarization vector the resulting field is assigned to. As the returned operator sets $x_1 = x_2$ and calls the result x_1 again, the result of

the operator acting on a four-point structure will be confusing by containing the variables x_1,x_3 and x_4 and the polarization vectors v_3,v_4 and $v_{\text{voffset}+1}$, where the latter is the polarization of the field x_1. This is due to technical difficulties concerning the differentiation targets. There are similar functions called *translate_op_34* and *translate_op_56*, which reduce in other variable pairs.

translate_pseudo_op(F,voffset) The same as *translate_op*, but for pseudo scalar operators. At the moment this function is only applicable for the simultaneous reduction in x_1 to x_4, assigning new polarization vectors $v_{\text{voffset}+1}$ to x_1 and $v_{\text{voffset}+2}$ to x_4.

DoDiff(E,d1,d2,n,kappa,L,a,b,mu) Applies the PDE (3.2.3) to expression E in x_a and x_b. μ is a pseudo index assigned to not contracted derivatives within the PDE and is used internally to identify linearly dependent terms.

diffcollect(F) Collects within expression F all identical derivative terms to reduce the size of an operator. For example, the expression $A \cdot vv[a,b] \cdot vw[c,d] + B \cdot vv[a,b] \cdot vw[c,d]$ would be translated to $(A+B) \cdot vv[a,b] \cdot vw[c,d]$.

epsilontoeta(F) Takes expression F and rewrites products of ε-tensors in terms of 24 η-tensors, which can then be contracted using

pacman(F) Contracts all indices in expression F if possible. This expression might contain vectors of the form $x[mu[i]]$, $v[mu[j]]$, $w[mu[k]]$ or $y[mu[l]]$ with arbitrary integers $i,\ldots,l$ and ε-tensors of the form $eta[mu[a],mu[b],mu[c],mu[d]]$.

diffmagic(E) Rearranges terms $vv[i,j]$, $ww[i,j]$ and $yy[i,j]$ to ensure that $i \leq j$.

A.1.3 diffops.mpl

This file contains the basic differential operators that are used by the operators returned by the function *tranlsate_op*. In order to work properly, the variables m and n have to be set correctly. n is the number of spacetime variables $x_1,\ldots,x_n$ within a target correlation function, m is the number of polarization vectors.

Reducing a four-point structure, it is generally a good idea to set $n = 4$ and $m = 6$, as the operators are likely to introduce polarization vectors v_5 and v_6.

Apart of the basic operators themselves, the file contains the following functions:

magic(E) Rearranges terms $vv[ij]$, $vx[k,i,j]$ to ensure that $i \leq j$ and sets $vx[k,i,i] = 0$ and $rho[i,i] = 0$ for all i.

magic2(E) Replaces $vx[k,i,j]$ and $rho[i,j]$ by auxiliary variables $x[i]$ and $x[j]$ to check whether the expression E vanishes. This is necessary due to linear dependencies among the $x_{ij} = x_i - x_j$.

setequal(F,a,b) Replaces all occurrences of x_b by x_a.

swap_indices(F,a,b) Swaps all occurrences of x_a with x_b and v_a with v_b.

A.2 External scripts

Many functions of the *Maple* scripts above call external scripts, as *Maple* lacks many basic programming features as well as speed. The following scripts are used:

collect.py Renames the operators within a raw operator expression such that *Maple* can collect them. This script is used within the method *diffcollect*.

collect2.py Inverts the renaming from *collect.py* after the collection is done. This script is used as well within the method *diffcollect*.

combinator.py Generates an ansatz for a scalar operator for given representations and saves it to a file. Used within the *buildop* method.

combinator2.py Generates an ansatz for a pseudo scalar operator for given representations and saves it to a file. Used within the *buildpseudoop* method.

epsilontoeta.py Searches in an expression for a product of two ε-tensors and rewrites them as 24 η-tensors. Used by *buildpseudoop*.

gammatrace.py Calculates the trace of a product of $2n$ γ-matrices with Lorentz indices μ_1 to μ_{2n}.

optranslate_files.py Translates the raw version of an operator to a real *Maple* operator. Used within the *translate_op* methods and to calculate the free Dirac structure.

pacman2.py Script that contracts Lorentz vectors and η-tensors. Used to contract the exotic structure and within *buildpseudoop*.

solve.py Rewrites the result of the application of the PDE to a set of linear equations of the coefficients that can be solved by *Maple*. Used by *buildop* and *buildpseudoop*.

B Sources

B.1 utils.mpl

```
1  if not assigned(INTERTWINER_PATH) then
2          error "Please␣set␣the␣variable␣INTERTWINER_PATH␣to␣the␣correct␣path!␣Thank␣you.";
3  end if;
4
5  kernelopts(stacklimit=2^20);
6
7  with(FileTools);
8
9  HOMEDIR := kernelopts(homedir);
10 SCRIPTPATH := "" || INTERTWINER_PATH || "/scripts";
11 USERNAME := kernelopts(username);
12 TMPPATH := "/tmp/" || USERNAME || "_intertwiners/" ;
13
14 if not Exists(TMPPATH) then
15         MakeDirectory(TMPPATH);
16 end if;
```

B.2 diffsolver.mpl

```
1  if not assigned(INTERTWINER_PATH) then
2          error "Please␣set␣the␣variable␣INTERTWINER_PATH␣to␣the␣correct␣path!␣Thank␣you.";
3  end if;
4
5  read "" || INTERTWINER_PATH || "/utils.mpl";
6  with(SolveTools);
7
8  m := 4;
9
10 #the diffop Nabla_x_a,mu
11 Nablaamu := (E, mu, a) ->
12 add( diff(E, vy[i,a]) * v[i][mu], i=1..m )
13 + add( diff(E, wy[i,a] ) * w[i][mu], i=1..m)
14 + add( diff(E, yy[a,i]) * y[i][mu], i=1..m)
15 + add( diff(E, yy[i,a]) * y[i][mu], i=1..m);
16
17 #the diffop Nabla_v_a,mu
18 Nablavamu := (E, mu, a) ->
19 add( diff(E, vw[i,a]) * v[i][mu], i=1..m )
20 + add( diff(E, wy[a,i] ) * y[i][mu], i=1..m)
21 + add( diff(E, ww[a,i]) * w[i][mu], i=1..m)
22 + add( diff(E, ww[i,a]) * w[i][mu], i=1..m);
23
24 #the diffop Nabla_xa Nabla_xa
25 NablaaNablaa := (E, a) ->
26 add(  add( diff(E, vy[i,a], vy[j,a]) * vv[i,j], i=1..m)
27      + add( diff(E, wy[i,a], vy[j,a]) * vw[j,i], i=1..m)
28      + add( diff(E, yy[a,i], vy[j,a]) * vy[j,i], i=1..m)
29      + add( diff(E, yy[i,a], vy[j,a]) * vy[j,i], i=1..m)
30 , j=1..m)
31 + add( add( diff(E, vy[i,a], wy[j,a]) * vw[i,j], i=1..m)
32      + add( diff(E, wy[i,a], wy[j,a]) * alb(i,j) * ww[i,j], i=1..m)
```

```
33      + add( diff(E, wy[i,a], wy[j,a]) * agb(i,j) * ww[j,i], i=1..m)
34      + add( diff(E, wy[i,a], wy[j,a]) * kdel(i,j) * ww[i,i], i=1..m)
35      + add( diff(E, yy[a,i], wy[j,a]) * wy[j,i], i=1..m)
36      + add( diff(E, yy[i,a], wy[j,a]) * wy[j,i], i=1..m)
37  , j=1..m)
38  + add( add( diff(E, vy[i,a], yy[a,j]) * vy[i,j], i=1..m), j=1..m)
39  + add( add( diff(E, vy[i,a], yy[j,a]) * vy[i,j], i=1..m), j=1..m)
40  + add( add( diff(E, wy[i,a], yy[a,j]) * wy[i,j], i=1..m), j=1..m)
41  + add( add( diff(E, wy[i,a], yy[j,a]) * wy[i,j], i=1..m), j=1..m)
42  + add( add( diff(E, yy[a,i], yy[a,j]) * yy[i,j], i=1..m), j=1..m)
43  + add( add( diff(E, yy[i,a], yy[a,j]) * yy[i,j], i=1..m), j=1..m)
44  + add( add( diff(E, yy[i,a], yy[j,a]) * yy[i,j], i=1..m), j=1..m)
45  + add( add( diff(E, yy[a,i], yy[j,a]) * yy[i,j], i=1..m), j=1..m)
46  + 8*diff(E, yy[a,a]);
47
48  #the diffop Nabla_xa Nabla_va
49  NablaaNablava := (E, a) ->
50  4*diff(E, wy[a,a])
51  + add( add( diff(E, ww[a,i], vy[j,a]) * vw[j,i], i=1..m), j=1..m)
52  + add( add( diff(E, ww[i,a], vy[j,a]) * vw[j,i], i=1..m), j=1..m)
53  + add( add( diff(E, ww[a,i], wy[j,a]) * ww[i,j], i=1..m), j=1..m)
54  + add( add( diff(E, ww[i,a], wy[j,a]) * ww[i,j], i=1..m), j=1..m)
55  + add( add( diff(E, vw[i,a], vy[j,a]) * vv[i,j], i=1..m), j=1..m)
56  + add( add( diff(E, vw[i,a], wy[j,a]) * vw[i,j], i=1..m), j=1..m)
57  + add( add( diff(E, vw[i,a], yy[a,j]) * vy[i,j], i=1..m), j=1..m)
58  + add( add( diff(E, vw[i,a], yy[j,a]) * vy[i,j], i=1..m), j=1..m)
59  + add( add( diff(E, wy[a,i], vy[j,a]) * vy[j,i], i=1..m), j=1..m)
60  + add( add( diff(E, wy[a,i], wy[j,a]) * wy[j,i], j=1..m), i=1..m)
61  + add( add( diff(E, wy[a,i], yy[a,j]) * yy[i,j], i=1..m), j=1..m)
62  + add( add( diff(E, wy[a,i], yy[j,a]) * yy[i,j], i=1..m), j=1..m)
63  + add( add( diff(E, ww[a,i], yy[a,j]) * wy[i,j], i=1..m), j=1..m)
64  + add( add( diff(E, ww[i,a], yy[a,j]) * wy[i,j], i=1..m), j=1..m)
65  + add( add( diff(E, ww[i,a], yy[j,a]) * wy[i,j], i=1..m), j=1..m)
66  + add( add( diff(E, ww[a,i], yy[j,a]) * wy[i,j], i=1..m), j=1..m);
67
68  #the diffop y_a Nabla_x_a
69  yaNablaa := (E, a) ->
70  add( diff(E, vy[i,a]) * vy[i,a], i=1..m )
71  + add( diff(E, wy[i,a] ) * wy[i,a], i=1..m)
72  + add( diff(E, yy[a,i]) * yy[a,i], i=1..m)
73  + add( diff(E, yy[i,a]) * yy[i,a], i=1..m);
74
75  #the diffop w_a Nabla_x_a
76  waNablaa := (E, a) ->
77  add( diff(E, vy[i,a]) * vw[i,a], i=1..m )
78  + add( diff(E, wy[i,a] ) * alb(a,i) * ww[a,i], i=1..m)
79  + add( diff(E, wy[i,a] ) * agb(a,i) * ww[i,a], i=1..m)
80  + add( diff(E, wy[i,a] ) * kdel(a,i) * ww[a,a], i=1..m)
81  + add( diff(E, yy[a,i]) * wy[a,i], i=1..m)
82  + add( diff(E, yy[i,a]) * wy[a,i], i=1..m);
83
84  #auxiliary functions
85  kdel := (r,s)-> 1-abs(signum(r-s));
86  agb := (r,s)->(signum(r-s)+1)/2 * abs(signum(r-s));
87  alb := (r,s)->(signum(s-r)+1)/2 * abs(signum(r-s));
88
89  #diffop d/dv_e d/dv_c, used for guaranteeing tracelessness
90  OpPartvPartv := (F, e, c) ->
91  add( add( diff(F, vv[c,d], vv[e,f]) * vv[f,d], d=c..m), f=e..m)
92  + add( add( diff(F, vv[c,d], vv[f,e]) * vv[f,d], d=c..m), f=1..e)
93  +4*alb(c,e)*diff(F, vv[c,e]) + 4*agb(c,e)*diff(F, vv[e,c]) + 8*kdel(c,e)*diff(F, vv[c,e])
94  +add( add( diff(F, vv[d,c], vv[e,f]) * vv[f,d], d=1..c), f=e..m)
95  + add( add( diff(F, vv[d,c], vv[f,e]) * vv[f,d], d=1..c), f=1..e)
96  +add( add( diff(F, vw[c,i], vv[e,f]) * vw[f,i], i=1..4), f=e..m)
97  +add( add( diff(F, vw[c,i], vv[f,e]) * vw[f,i], i=1..4), f=1..e)
98  +add( add( diff(F, vw[c,i], vw[e,j]) * ww[i,j], i=1..4), j=1..4)
99  +add( add( diff(F, vw[c,i], vy[e,j]) * wy[i,j], i=1..4), j=1..4)
100 +add( add( diff(F, vy[c,i], vv[e,f]) * vy[f,i], i=1..4), f=e..m)
101 +add( add( diff(F, vy[c,i], vv[f,e]) * vy[f,i], i=1..4), f=1..e)
102 +add( add( diff(F, vy[c,i], vw[e,j]) * wy[j,i], i=1..4), j=1..4)
103 +add( add( diff(F, vy[c,i], vy[e,j]) * yy[i,j], i=1..4), j=1..4)
104 +add( add( diff(F, vv[c,i], vw[e,j]) * vw[i,j], i=c..m), j=1..4)
```

```
105 +add( add( diff(F, vv[c,i], vy[e,j]) * vy[i,j], i=c..m), j=1..4)
106 +add( add( diff(F, vv[i,c], vw[e,j]) * vw[i,j], i=1..c), j=1..4)
107 +add( add( diff(F, vv[i,c], vy[e,j]) * vy[i,j], i=1..c), j=1..4);
108
109 #non-pseudo ops only, used to reduce diffops from n to n-1
110 Nabla1Nabla2 := E ->
111 add(  add( diff(E, vy[i,2], vy[j,1]) * vv[i,j], i=1..m)
112      + add( diff(E, wy[i,2], vy[j,1]) * vw[j,i], i=1..2)
113      + diff(E, yy[1,2], vy[j,1]) * vy[j,1]
114      + 2*diff(E, yy[2,2], vy[j,1]) * vy[j,2]
115  , j=1..m)
116 + add( add( diff(E, vy[i,2], wy[j,1]) * vw[i,j], i=1..m)
117      + add( diff(E, wy[i,2], wy[j,1]) * ww[j,i], i=1..2)
118      + diff(E, yy[1,2], wy[j,1]) * wy[j,1]
119      + 2*diff(E, yy[2,2], wy[j,1]) * wy[j,2]
120  , j=1..2)
121 + add( diff(E, vy[i,2], yy[1,2]) * vy[i,2], i=1..m)
122 + add( diff(E, wy[i,2], yy[1,2]) * wy[i,2], i=1..2)
123 + diff(E, yy[1,2], yy[1,2]) * yy[1,2]
124 + 2*diff(E, yy[2,2], yy[1,2]) * yy[2,2]
125 + 2*add( diff(E, vy[i,2], yy[1,1]) * vy[i,1], i=1..m)
126 + 2*add( diff(E, wy[i,2], yy[1,1]) * wy[i,1], i=1..2)
127 + 2*diff(E, yy[1,2], yy[1,1]) * yy[1,1]
128 + 4*diff(E, yy[2,2], yy[1,1]) * yy[1,2]
129 + 4*diff(E, yy[1,2]);
130
131 #brings terms in right order, to be called after every diffop from above
132 diffmagic := proc(E)
133   local tmp, i, j;
134
135   tmp := E;
136   for i from 1 to m do
137     for j from 1 to i-1 do
138       tmp := subs(vv[i,j] = vv[j,i], tmp);
139     end do;
140   end do;
141
142     for i from 1 to m do
143     for j from 1 to i-1 do
144       tmp := subs(ww[i,j] = ww[j,i], tmp);
145     end do;
146   end do;
147
148    for i from 1 to m do
149     for j from 1 to i-1 do
150       tmp := subs(yy[i,j] = yy[j,i], tmp);
151     end do;
152   end do;
153
154   tmp;
155 end proc;
156
157 #short for the function above
158 dmv := F -> diffmagic(F);
159
160 #contracts all vectors with indices, if possible
161 pacman := proc(F)
162   local tmp;
163   tmp := expand(F);
164
165   save tmp, "" || TMPPATH || "maplecontract.txt";
166   ssystem("python " || SCRIPTPATH || "/pacman2.py " || TMPPATH || "maplecontract.txt " ||
          TMPPATH || "maplecontracted.txt");
167   read "" || TMPPATH || "maplecontracted.txt";
168
169   tmp := %;
170   tmp;
171 end proc;
172
173 #collects for same operator terms to reduce the size of operators
174 diffcollect := proc(F)
175   local tmp;
```

```
176     tmp := expand(F);
177     save tmp, "" || TMPPATH || "maplecollect.txt";
178     print("python " || SCRIPTPATH || "/collect.py " || TMPPATH || "maplecollect.txt " || TMPPATH
            || "maplecollected.txt");
179     ssystem("python " || SCRIPTPATH || "/collect.py " || TMPPATH || "maplecollect.txt " || TMPPATH
            || "maplecollected.txt");
180     print("done");
181     read "" || TMPPATH || "maplecollected.txt";
182     tmp := %;
183     print("collecting...");
184     tmp := collect(expand(tmp), maplecollectterms);
185     print("collected!");
186
187     save tmp, "" || TMPPATH || "maplecollect2.txt";
188     print("python " || SCRIPTPATH || "/collect2.py " || TMPPATH || "maplecollect2.txt " || TMPPATH
            || "maplecollected2.txt");
189     ssystem("python " || SCRIPTPATH || "/collect2.py " || TMPPATH || "maplecollect2.txt " ||
            TMPPATH || "maplecollected2.txt");
190     print("done");
191     read "" || TMPPATH || "maplecollected2.txt";
192     tmp := %;
193
194     tmp;
195   end proc;
196
197   #reduces a non-pseudo op from n to n-1
198   DoReduce := F -> dmv(Nabla1Nabla1(F)-2*Nabla1Nabla2(F)+Nabla2Nabla2(F));
199
200   #the actual application of the PDE to an expression
201   DoDiff := (E, d1, d2, n, kappa, L, a, b, mu) -> 2 * (Nablaamu(diffmagic(value(yaNablaa(E, a))),
          mu, a) - Nablaamu(E, mu, a)) - y[a][mu] * NablaaNablaa(E, a) + 2*(Nablavamu( diffmagic(value
          (waNablaa(E, a))), mu, a) - Nablaamu(E, mu, a) - w[a][mu] * NablaaNablava(E, a)) + (2*d1 -
          2*n) * Nablaamu(E, mu, a)
202   + 2 * (Nablaamu(diffmagic(value(yaNablaa(E, b))), mu, b) - Nablaamu(E, mu, b)) - y[b][mu] *
          NablaaNablaa(E, b) + 2*(Nablavamu( diffmagic(value(waNablaa(E, b)))), mu, b) - Nablaamu(E, mu
          , b) - w[b][mu] * NablaaNablava(E, b)) + (2*d2 - 2*n) * Nablaamu(E, mu, b);
203
204   #rewrites products of two epsilon-tensors as sum of products of eta-tensors
205   epsilontoeta := proc(F)
206     local tmp;
207     tmp := expand(F);
208     save tmp, "" || TMPPATH || "mapleepsilontoeta.txt";
209     ssystem("python " || SCRIPTPATH || "/epsilontoeta.py " || TMPPATH || "mapleepsilontoeta.txt "
            || TMPPATH || "mapleepsilontoeta2.txt");
210     read "" || TMPPATH || "mapleepsilontoeta2.txt";
211     tmp := %;
212
213     tmp;
214   end proc;
215
216   #builds the actual non-pseudo operator to reduce from representation kappa1, L1 and kappa2, L2
          to kappa and L for singularities up to degree of n
217   buildop := proc(kappa1, L1, kappa2, L2, n, kappa, L)
218     local numv, numw1, numw2, numy, tmp, subsme, ansatz2, ansatz3, ansatz4;
219     #build ansatz
220     numv := L;
221     numw1 := L1;
222     numw2 := L2;
223     numy := 2*kappa + L + 2*n - 2*kappa1 - L1 - 2*kappa2 - L2;
224     print("python " || SCRIPTPATH || "/combinator.py " || numv || " " || numw1 || " " || numw2 ||
            " " || numy || " " || TMPPATH || "maplecombinate.txt");
225     ssystem("python " || SCRIPTPATH || "/combinator.py " || numv || " " || numw1 || " " || numw2
            || " " || numy || " " || TMPPATH || "maplecombinate.txt");
226     read "" || TMPPATH || "maplecombinate.txt";
227
228     print("Ansatz and factors:");
229     print(used_factors);
230     print(ansatz);
231
232     tmp := collect(simplify(expand(dmv(DoDiff(ansatz, 2*kappa1+L1, 2*kappa2+L2, n, kappa, L, 1, 2,
            mu)))), used_factors);
233     save tmp, "" || TMPPATH || "mapleops.txt";
```

```
234     print("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
            mapleops2.txt");
235     ssystem("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
            mapleops2.txt");
236     read "" || TMPPATH || "mapleops2.txt";
237
238     subsme := Linear(solveme, indets(solveme));
239     #print(subsme);
240     ansatz2 := simplify(expand(subs(subsme, ansatz)));
241     print("Solved PDE...");
242     #print(ansatz2);
243     tmp := collect(simplify(expand(dmv(OpPartvPartv(ansatz2, 1, 1)))), used_factors);
244     if tmp <> 0 then
245       save tmp, "" || TMPPATH || "mapleops.txt";
246       print("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
              mapleops2.txt");
247       ssystem("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH ||
              "mapleops2.txt");
248       read "" || TMPPATH || "mapleops2.txt";
249
250       subsme := Linear(solveme, indets(solveme));
251       print("Made traceless...");
252       ansatz3 := simplify(expand(subs(subsme, ansatz2)));
253     else
254       ansatz3 := ansatz2;
255     end if;
256
257     ansatz4 := expand(ansatz3);
258     #uncomment to make operator smaller
259     #ansatz4 := diffcollect(expand(ansatz3));
260
261
262     tmp := simplify(expand(dmv(DoDiff(ansatz4, 2*kappa1+L1, 2*kappa2+L2, n, kappa, L, 1, 2, mu))))
            ;
263     if tmp <> 0 then
264       error "Consistency check failed!";
265     end if;
266
267     tmp := simplify(expand(dmv(OpPartvPartv(ansatz4, 1, 1))));
268     if tmp <> 0 then
269       error "Consistency check failed!";
270     end if;
271
272     ansatz4;
273   end proc;
274
275   #builds the actual pseudo operator to reduce from representation kappa1, L1 and kappa2, L2 to
          kappa and L for singularities up to degree of n
276   buildpseudoop := proc(kappa1, L1, kappa2, L2, n, kappa, L)
277     local numv, numw1, numw2, numy, tmp, subsme, ansatz2, ansatz3, ansatz4, ansatz5, ansatz6,
            ansatza, ansatzb;
278     #build ansatz
279     numv := L;
280     numw1 := L1;
281     numw2 := L2;
282     numy := 2*kappa + L + 2*n - 2*kappa1 - L1 - 2*kappa2 - L2;
283     print("python " || SCRIPTPATH || "/combinator2.py " || numv || " " || numw1 || " " || numw2 ||
            " " || numy || " " || TMPPATH || "maplecombinate.txt");
284     ssystem("python " || SCRIPTPATH || "/combinator2.py " || numv || " " || numw1 || " " || numw2
            || " " || numy || " " || TMPPATH || "maplecombinate.txt");
285     read "" || TMPPATH || "maplecombinate.txt";
286
287     print("Ansatz and factors:");
288     print(used_factors);
289     print(ansatz);
290
291     ansatza := epsilontoeta(ansatz);
292     #print(ansatza);
293     print("Rewritten epsilon-products...");
294     ansatzb := pacman(ansatza);
295     #print(ansatzb);
296
```

```
297   print("Contracted epsilons...");
298
299   tmp := collect(simplify(expand(dmv(DoDiff(ansatzb, 2*kappa1+L1, 2*kappa2+L2, n, kappa, L, 1,
          2, mu)))), used_factors);
300   save tmp, "" || TMPPATH || "mapleops.txt";
301   print("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
          mapleops2.txt");
302   ssystem("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
          mapleops2.txt");
303   read "" || TMPPATH || "mapleops2.txt";
304
305   subsme := Linear(solveme, indets(solveme));
306
307   ansatz2 := simplify(expand(subs(subsme, ansatzb)));
308   print("Solved PDE for x1 and x2...");
309
310   tmp := collect(simplify(expand(dmv(OpPartvPartv(ansatz2, 1, 1)))), used_factors);
311   if tmp <> 0 then
312     save tmp, "" || TMPPATH || "mapleops.txt";
313     print("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
            mapleops2.txt");
314     ssystem("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH ||
            "mapleops2.txt");
315     read "" || TMPPATH || "mapleops2.txt";
316
317     subsme := Linear(solveme, indets(solveme));
318     print("Made traceless in v1...");
319     ansatz3 := simplify(expand(subs(subsme, ansatz2)));
320   else
321     ansatz3 := ansatz2;
322   end if;
323
324   tmp := collect(simplify(expand(dmv(DoDiff(ansatz3, 2*kappa1+L1, 2*kappa2+L2, n, kappa, L, 4,
          3, mu)))), used_factors);
325   if tmp <> 0 then
326     save tmp, "" || TMPPATH || "mapleops.txt";
327     print("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
            mapleops2.txt");
328     ssystem("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH ||
            "mapleops2.txt");
329     read "" || TMPPATH || "mapleops2.txt";
330
331     subsme := Linear(solveme, indets(solveme));
332     print("Solved PDE for x3 and x4...");
333     ansatz4 := simplify(expand(subs(subsme, ansatz3)));
334   else
335     ansatz4 := ansatz3;
336   end if;
337
338   tmp := collect(simplify(expand(dmv(OpPartvPartv(ansatz4, 2, 2)))), used_factors);
339   if tmp <> 0 then
340     save tmp, "" || TMPPATH || "mapleops.txt";
341     print("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH || "
            mapleops2.txt");
342     ssystem("python " || SCRIPTPATH || "/solve.py " || TMPPATH || "mapleops.txt " || TMPPATH ||
            "mapleops2.txt");
343     read "" || TMPPATH || "mapleops2.txt";
344
345     subsme := Linear(solveme, indets(solveme));
346     print("Made traceless in v2...");
347     ansatz5 := simplify(expand(subs(subsme, ansatz4)));
348   else
349     ansatz5 := ansatz4;
350   end if;
351
352   ansatz6 := expand(ansatz5);
353   #uncomment for shorter operators
354   #ansatz6 := diffcollect(expand(ansatz5));
355
356   tmp := simplify(expand(dmv(DoDiff(ansatz6, 2*kappa1+L1, 2*kappa2+L2, n, kappa, L, 1, 2, mu))))
          ;
357   if tmp <> 0 then
```

```
358       error "Consistency check failed!";
359     end if;
360
361     tmp := simplify(expand(dmv(OpPartvPartv(ansatz6, 1, 1))));
362     if tmp <> 0 then
363       error "Consistency check failed!";
364     end if;
365
366     tmp := simplify(expand(dmv(DoDiff(ansatz6, 2*kappa1+L1, 2*kappa2+L2, n, kappa, L, 4, 3, mu))))
          ;
367     if tmp <> 0 then
368       error "Consistency check failed!";
369     end if;
370
371     tmp := simplify(expand(dmv(OpPartvPartv(ansatz6, 2, 2))));
372     if tmp <> 0 then
373       error "Consistency check failed!";
374     end if;
375
376     ansatz6;
377   end proc;
378
379
380   translate_op_12 := proc(F, voffset)
381     local tmp;
382     global igel;
383     tmp := F:
384     save tmp, "" || TMPPATH || "mapleoptranslate.txt";
385     ssystem("python " || SCRIPTPATH || "/optranslate_files.py " || voffset || " 1 2 " || TMPPATH
          || "mapleoptranslate.txt " || TMPPATH || "mapleoptranslated.txt");
386     read "" || TMPPATH || "mapleoptranslated.txt";
387
388     tmp := %;
389
390     return tmp;
391   end proc;
392
393   translate_op_34 := proc(F, voffset)
394     local tmp;
395     global igel;
396     tmp := F:
397     save tmp, "" || TMPPATH || "mapleoptranslate.txt";
398     ssystem("python " || SCRIPTPATH || "/optranslate_files.py " || voffset || " 4 3 " || TMPPATH
          || "mapleoptranslate.txt " || TMPPATH || "mapleoptranslated.txt");
399     read "" || TMPPATH || "mapleoptranslated.txt";
400
401     tmp := %;
402
403     return tmp;
404   end proc;
405
406   translate_op_56 := proc(F, voffset)
407     local tmp;
408     global igel;
409     tmp := F:
410     save tmp, "" || TMPPATH || "mapleoptranslate.txt";
411     ssystem("python " || SCRIPTPATH || "/optranslate_files.py " || voffset || " 6 5 " || TMPPATH
          || "mapleoptranslate.txt " || TMPPATH || "mapleoptranslated.txt");
412     read "" || TMPPATH || "mapleoptranslated.txt";
413
414     tmp := %;
415
416     return tmp;
417   end proc;
418
419   translate_pseudo_op := proc(F, voffset)
420     local tmp;
421     global igel;
422     tmp := F:
423     save tmp, "" || TMPPATH || "mapleoptranslate.txt";
424     ssystem("python " || SCRIPTPATH || "/optranslate_files.py " || voffset || " 1 2 " || TMPPATH
          || "mapleoptranslate.txt " || TMPPATH || "mapleoptranslated.txt");
```

```
425   read "" || TMPPATH || "mapleoptranslated.txt";
426
427   tmp := %;
428
429   return tmp;
430 end proc;
```

B.3 diffops.mpl

```
1  if not assigned(n) then
2          error "Please␣set␣the␣variable␣n␣to␣the␣correct␣value␣(number␣of␣variables␣x,␣including␣
               new␣ones).␣Thank␣you.";
3  end if;
4
5  if not assigned(m) then
6          error "Please␣set␣the␣variable␣m␣to␣the␣correct␣value␣(number␣of␣polarization␣vectors,␣
               including␣new␣ones).␣Thank␣you.";
7  end if;
8
9  #auxiliary functions
10 kdel := (a,b)-> 1-abs(signum(a-b));
11 agb := (a,b)->(signum(a-b)+1)/2 * abs(signum(a-b));
12 alb := (a,b)->(signum(b-a)+1)/2 * abs(signum(a-b));
13
14 #to be called after every application of a diffop below. Brings terms in right order and does
       basic simplifications
15 mv := F-> expand(magic(F));
16
17 #diffop w_c y_i
18 PartvParti := (F, c, i) ->
19 add(# c <= d
20   add( # k
21     add( diff(F, vx[k,i,j], vv[c,d]) * vv[k,d], j=i+1..n)
22     - add( diff(F, vx[k,j,i], vv[c,d]) * vv[k,d], j=1..i-1),
23   k=1..m)
24   + 2*add( diff(F, rho[i,j], vv[c,d]) * vx[d,i,j], j=i+1..n)
25   - 2*add( diff(F, rho[j,i], vv[c,d]) * vx[d,j,i], j=1..i-1),
26 d=c..m)
27 +add( # c >= d
28   add ( # k
29     add( diff(F, vx[k,i,j], vv[d,c]) * vv[k,d], j=i+1..n)
30     - add( diff(F, vx[k,j,i], vv[d,c]) * vv[k,d], j=1..i-1),
31   k=1..m)
32   + 2*add( diff(F, rho[i,j], vv[d,c]) * vx[d,i,j], j=i+1..n)
33   - 2*add( diff(F, rho[j,i], vv[d,c]) * vx[d,j,i], j=1..i-1),
34 d=1..c)
35 +add( add( #a < b
36   add ( # k
37    add( diff(F, vx[k,i,j], vx[c,a,b]) * vx[k,a,b], j=i+1..n)
38     - add( diff(F, vx[k,j,i], vx[c,a,b]) * vx[k,a,b], j=1..i-1),
39   k=1..m)
40   + 2*add( diff(F, rho[i,j], vx[c,a,b]) * 1/2*(rho[a,j] + rho[b,i] - rho[a,i] - rho[b,j]), j=i
         +1..n)
41   - 2*add( diff(F, rho[j,i], vx[c,a,b]) * 1/2*(rho[a,i] + rho[b,j] - rho[a,j] - rho[b,i]), j=1..
         i-1),
42 b=a+1..n), a=1..n)
43 +4*add( #i < j
44   diff(F, vx[c,i,j]),
45 j=i+1..n)
46 -4*add( #i > j
47   diff(F, vx[c,j,i]),
48 j=1..i-1);
49
50 #diffop y_a y_i
51 PartiPartj := (F, a, i) ->
52 add( # c
53   add( #a < b
54     add( # k
55       add( diff(F, vx[k,i,j], vx[c,a,b]) * vv[k,c], j=i+1..n)
56       - add( diff(F, vx[k,j,i], vx[c,a,b]) * vv[k,c], j=1..i-1),
```

```
57        k=1..m)
58        + 2*add( diff(F, rho[i,j], vx[c,a,b]) * vx[c,i,j], j=i+1..n)
59        - 2*add( diff(F, rho[j,i], vx[c,a,b]) * vx[c,j,i], j=1..i-1),
60      b=a+1..n)
61      - add( #a > b
62        add( # k
63          add( diff(F, vx[k,i,j], vx[c,b,a]) * vv[k,c], j=i+1..n)
64          - add( diff(F, vx[k,j,i], vx[c,b,a]) * vv[k,c], j=1..i-1),
65        k=1..m)
66        + 2*add( diff(F, rho[i,j], vx[c,b,a]) * vx[c,i,j], j=i+1..n)
67        - 2*add( diff(F, rho[j,i], vx[c,b,a]) * vx[c,j,i], j=1..i-1),
68      b=1..a-1),
69    c=1..m)
70    +2*add( #a < b
71      add( # k
72        add( diff(F, vx[k,i,j], rho[a,b]) * vx[k,a,b], j=i+1..n)
73        - add( diff(F, vx[k,j,i], rho[a,b]) * vx[k,a,b], j=1..i-1),
74      k=1..m)
75      + 2*add( diff(F, rho[i,j], rho[a,b]) * 1/2*(rho[a,j] + rho[b,i] - rho[a,i] - rho[b,j]), j=i
            +1..n)
76      - 2*add( diff(F, rho[j,i], rho[a,b]) * 1/2*(rho[a,i] + rho[b,j] - rho[a,j] - rho[b,i]), j=1..i
            -1),
77    b=a+1..n)
78    -2*add( #a > b
79      add( # k
80        add( diff(F, vx[k,i,j], rho[b,a]) * vx[k,b,a], j=i+1..n)
81        - add( diff(F, vx[k,j,i], rho[b,a]) * vx[k,b,a], j=1..i-1),
82      k=1..m)
83      + 2*add( diff(F, rho[i,j], rho[b,a]) * 1/2*(rho[b,j] + rho[a,i] - rho[b,i] - rho[a,j]), j=i
            +1..n)
84      - 2*add( diff(F, rho[j,i], rho[b,a]) * 1/2*(rho[b,i] + rho[a,j] - rho[b,j] - rho[a,i]), j=1..i
            -1),
85    b=1..a-1)
86    +kdel(a,i)*8*add( diff(F, rho[i,k]), k=i+1..n) + kdel(a,i)*8*add( diff(F, rho[k,i]), k=1..i-1)
87    -alb(a,i)*8*diff(F, rho[a,i]) - agb(a,i)*8*diff(F, rho[i,a]);
88
89    #diffop v_c y_i
90    vParti := (F, c, i) ->
91    add( # k
92      add( diff(F, vx[k,i,j]) * vv[k,c], j=i+1..n)
93      - add( diff(F, vx[k,j,i]) * vv[k,c], j=1..i-1),
94    k=1..m)
95    + 2*add( diff(F, rho[i,j]) * vx[c,i,j], j=i+1..n)
96    - 2*add( diff(F, rho[j,i]) * vx[c,j,i], j=1..i-1);
97
98    #diffop v_k w_c
99    vPartv := (F, k, c) ->
100   add( diff(F, vv[c,d]) * vv[d,k], d=c..m)
101   + add( diff(F, vv[d,c]) * vv[d,k], d=1..c)
102   + add( add( diff(F, vx[c,a,b]) * vx[k,a,b], b=a+1..n), a=1..n);
103
104   #diffop w_e w_c
105   PartvPartv := (F, e, c) ->
106   add( add( diff(F, vv[c,d], vv[e,f]) * vv[f,d], d=c..m), f=e..m)
107   + add( add( diff(F, vv[c,d], vv[f,e]) * vv[f,d], d=c..m), f=1..e)
108   + add( add( add( diff(F, vv[c,d], vx[e,a,b]) * vx[d,a,b], d=c..m), b=a+1..n), a=1..n)
109   +4*alb(c,e)*diff(F, vv[c,e]) + 4*agb(c,e)*diff(F, vv[e,c]) + 8*kdel(c,e)*diff(F, vv[c,e])
110   +add( add( diff(F, vv[d,c], vv[e,f]) * vv[f,d], d=1..c), f=e..m)
111   + add( add( diff(F, vv[d,c], vv[f,e]) * vv[f,d], d=1..c), f=1..e)
112   + add( add( add( diff(F, vv[d,c], vx[e,a,b]) * vx[d,a,b], d=1..c), b=a+1..n), a=1..n)
113   +add( add( add( diff(F, vx[c,a,b], vv[e,f]) * vx[f,a,b], b=a+1..n), a=1..n), f=e..m)
114   +add( add( add( diff(F, vx[c,a,b], vv[f,e]) * vx[f,a,b], b=a+1..n), a=1..n), f=1..e)
115   +add( add( add( add( diff(F, vx[c,a,b], vx[e,g,h]) * 1/2 * (rho[g,b] + rho[a,h] - rho[g,a] - rho
          [h,b]), b=a+1..n), a=1..n), h=g+1..n), g=1..n);
116
117
118   #brings terms in right order
119   magic := proc(F)
120     local tmp, i, j, k, l;
121
122     tmp := F;
123
```

```
124     for i from 1 to n do
125       for j from 1 to i-1 do
126         tmp := subs(rho[i,j] = rho[j,i], tmp);
127         for k from 1 to m do
128           tmp := subs(vx[k,i,j] = -vx[k,j,i], tmp);
129         end do;
130       end do;
131     end do;
132
133     for k from 1 to m do
134       for l from 1 to k-1 do
135         tmp := subs(vv[k,l] = vv[l,k], tmp);
136       end do;
137     end do;
138
139     for i from 1 to n do
140       tmp := subs(rho[i,i] = 0, tmp);
141       for k from 1 to m do
142         tmp := subs(vx[k,i,i] = 0, tmp);
143       end do;
144     end do;
145     tmp;
146   end proc;
147
148   #replaces rho and vx by the underlying identities to check for vanishing contributions. Use this
        followed by 'simplify' to check whether an expression is zero.
149   magic2 := proc(F)
150     local tmp, i, j, k;
151
152     tmp := F;
153     for i from 1 to n do
154       for j from i+1 to n do
155         tmp := subs(rho[i,j] = x[i]^2 - 2*xx[i,j] + x[j]^2, tmp);
156         for k from 1 to m do
157           tmp := subs(vx[k,i,j] = vx[k,i]-vx[k,j], tmp);
158         end do;
159       end do;
160     end do;
161
162     tmp;
163   end proc;
164
165   #sets x_a = x_b and calls the result again x_b. To be used in most circumstances after the
        application of a intertwining operator.
166   setequal := proc(F, a, b)
167     local i, k, tmp;
168
169     tmp := F;
170     for i from 1 to a-1 do
171       tmp := subs(rho[i,a] = rho[i,b], tmp);
172       for k from 1 to m do
173         tmp := subs(vx[k,i,a] = vx[k,i,b], tmp);
174       end do;
175     end do;
176
177     for i from a+1 to n do
178       tmp := subs(rho[a,i] = rho[b,i], tmp);
179       for k from 1 to m do
180         tmp := subs(vx[k,a,i] = vx[k,b,i], tmp);
181       end do;
182     end do;
183
184     magic(tmp);
185   end proc;
186
187   #swaps x_a and x_b in the expression F
188   swap_indices := proc(F, a, b)
189     local i, j, k, tmp, aprime, bprime, difference, before;
190
191     aprime := a + n;
192     bprime := b + n;
193     tmp := F;
```

```
194     for i from 1 to 2*n do
195       tmp := subs(rho[i,a] = rho[i,bprime], tmp);
196       for k from 1 to m do
197         tmp := subs(vx[k,i,a] = vx[k,i,bprime], tmp);
198       end do;
199     end do;
200
201     #print(tmp);
202
203     for i from 1 to 2*n do
204       tmp := subs(rho[a,i] = rho[bprime,i], tmp);
205       for k from 1 to m do
206         tmp := subs(vx[k,a,i] = vx[k,bprime,i], tmp);
207       end do;
208     end do;
209
210     #print(tmp);
211
212     for i from 1 to 2*n do
213       tmp := subs(rho[i,b] = rho[i,aprime], tmp);
214       for k from 1 to m do
215         tmp := subs(vx[k,i,b] = vx[k,i,aprime], tmp);
216       end do;
217     end do;
218
219     #print(tmp);
220
221     for i from 1 to 2*n do
222       tmp := subs(rho[b,i] = rho[aprime,i], tmp);
223       for k from 1 to m do
224         tmp := subs(vx[k,b,i] = vx[k,aprime,i], tmp);
225       end do;
226     end do;
227
228     #print(tmp);
229
230     for i from 1 to 2*n do
231       for j from 1 to 2*n do
232         tmp := subs(vx[a, i, j] = vx[bprime, i, j], tmp);
233         tmp := subs(vx[b, i, j] = vx[aprime, i, j], tmp);
234       end do;
235     end do;
236
237     #print(tmp);
238
239     for i from 1 to 2*m do
240       tmp := subs(vv[i,a] = vv[i,bprime], tmp);
241     end do;
242     for i from 1 to 2*m do
243       tmp := subs(vv[a,i] = vv[bprime,i], tmp);
244     end do;
245     for i from 1 to 2*m do
246       tmp := subs(vv[i,b] = vv[i,aprime], tmp);
247     end do;
248     for i from 1 to 2*m do
249       tmp := subs(vv[b,i] = vv[aprime,i], tmp);
250     end do;
251
252
253     difference :=1;
254     while difference <> 0 do
255       before := tmp;
256       for i from 1 to 2*n do
257         tmp := subs(rho[i,aprime] = rho[i,a], tmp);
258         tmp := subs(rho[aprime,i] = rho[a,i], tmp);
259         tmp := subs(rho[i,bprime] = rho[i,b], tmp);
260         tmp := subs(rho[bprime,i] = rho[b,i], tmp);
261         for k from 1 to 2*m do
262           tmp := subs(vx[k,i,aprime] = vx[k,i,a], tmp);
263           tmp := subs(vx[k,aprime,i] = vx[k,a,i], tmp);
264           tmp := subs(vx[k,i,bprime] = vx[k,i,b], tmp);
265           tmp := subs(vx[k,bprime,i] = vx[k,b,i], tmp);
```

```
266         end do;
267         for j from 1 to 2*n do
268           tmp := subs(vx[aprime,i,j] = vx[a,i,j], tmp);
269           tmp := subs(vx[bprime,i,j] = vx[b,i,j], tmp);
270         end do;
271       end do;
272       for i from 1 to 2*m do
273         tmp := subs(vv[i,aprime] = vv[i,a], tmp);
274         tmp := subs(vv[aprime,i] = vv[a,i], tmp);
275         tmp := subs(vv[i,bprime] = vv[i,b], tmp);
276         tmp := subs(vv[bprime,i] = vv[b,i], tmp);
277       end do;
278       difference := before - tmp;
279     end do;
280     #print(tmp);
281
282     magic(tmp);
283   end proc;
284
285   #renames index 3 to 2, 4 to 3 and 6 to 4
286   rename := proc (F)
287     local tmp;
288
289     tmp := F;
290     tmp := subs(rho[1, 3] = rho[1, 2], rho[1, 4] = rho[1, 3], rho[1, 6] = rho[1, 4], rho[3, 4] =
           rho[2, 3], rho[3, 6] = rho[2, 4], rho[4, 6] = rho[3, 4], tmp);
291     tmp := subs(vx[1, 1, 3] = vx[1, 1, 2], vx[1, 1, 4] = vx[1, 1, 3], vx[1, 1, 6] = vx[1, 1, 4],
           vx[1, 3, 4] = vx[1, 2, 3], vx[1, 3, 6] = vx[1, 2, 4], vx[1, 4, 6] = vx[1, 3, 4], vx[6, 1,
           3] = vx[4, 1, 2], vx[6, 1, 4] = vx[4, 1, 3], vx[6, 1, 6] = vx[4, 1, 4], vx[6, 3, 4] = vx
           [4, 2, 3], vx[6, 3, 6] = vx[4, 2, 4], vx[6, 4, 6] = vx[4, 3, 4], vv[1, 6] = vv[1, 4], vv
           [6,6]=vv[4,4], tmp);
292
293     magic(tmp)
294
295   end proc;
296
297   #builds a matrix from coefficient factors. Sample: Give me 3*A*AA -7*A*BB - 7*B*AA + 3*B*BB and
         as factors the list [A, B], and i will give you the matrix ( (3, -7) (-7, 3) )
298   build_matrix := proc(A, factors)
299     local n, res, i, j, da, db, wupps;
300     n := nops(factors);
301     res := Matrix(n, n);
302     for i from 1 to n do:
303       for j from 1 to n do:
304         da := factors[i];
305         db := factors[j];
306         db := `` || db || db;
307         wupps := coeff( coeff( A, da), db);
308         print("" || da, "" || db, wupps);
309         res[i,j] := wupps;
310       end do:
311     end do:
312     res;
313   end proc:
```

Bibliography

[1] Ralph Blumenhagen and Erik Plauschinn. Basics in conformal field theory. In *Introduction to Conformal Field Theory*. Springer, 2009.

[2] Francis A. Dolan and Hugh Osborn. Conformal four point functions and the operator product expansion. *Nuclear Physics B*, 599(1):459–496, 2001.

[3] Gerhard Mack. All unitary ray representations of the conformal group su (2, 2) with positive energy. *Communications in Mathematical Physics*, 55(1):1–28, 1977.

[4] Gerhard Mack. Convergence of operator product expansions on the vacuum in conformal invariant quantum field theory. *Communications in Mathematical Physics*, 53(2):155–184, 1977.

[5] Pierre Mathieu, David Sénéchal, and Phillipe Di Francesco. *Conformal field theory*. Springer, 1997.

[6] Christoph Neumann, Karl-Henning Rehren, and Lena Wallenhorst. New methods in conformal partial wave analysis. In *Lie Theory and Its Applications in Physics*, pages 109–125. Springer, 2013.

[7] Nikolay M. Nikolov and Ivan T. Todorov. Rationality of conformally invariant local correlation functions on compactified Minkowski space. *Communications in Mathematical Physics*, 218(2):417–436, 2001.

[8] Nikolay M. Nikolov, Karl-Henning Rehren, and Ivan T. Todorov. Partial wave expansion and wightman positivity in conformal field theory. *Nuclear Physics B*, 722(3):266–296, 2005.

[9] Nikolay M. Nikolov, Karl-Henning Rehren, and Ivan T. Todorov. Harmonic bilocal fields generated by globally conformal invariant scalar fields. *Communications in Mathematical Physics*, 279(1):225–250, 2008.

[10] Karl-Henning Rehren. Konforme Quantenfeldtheorie. *Lecture Notes*, 1997.

[11] Martin Schottenloher. *A mathematical introduction to conformal field theory.* Springer, 2008.

[12] Yassen S. Stanev. Correlation functions of conserved currents in four dimensional conformal field theory. *Nuclear Physics B*, 2012.

[13] Raymond F. Streater and Arthur S. Wightman. *PCT, spin and statistics, and all that.* Princeton University Press, 1964.

[14] Lena Marie Wallenhorst. Higher order conformal partial wave analysis in 4-dimensional quantum field theory. *Master thesis*, 2012.

[15] Kenneth G. Wilson. Non-lagrangian models of current algebra. *Physical Review*, 179(5):1499, 1969.

Acknowledgements

I am grateful to Prof. Karl-Henning Rehren for guidance and support during the preparation of my thesis and the time he spent on helpful explanations and his patience. I would like to thank Prof. Laura Covi for being second referee and Christopher Eckner for many helpful and constructive discussions at any time of day. Moreover I thank all those who proofread this thesis, in particular Magdalena Albrecht, Christopher Eckner, Johannes Frey and Georg Roth-Kleyer.

Erklärung nach §13(8) der Prüfungsordnung für den Bachelor-Studiengang Physik und den Master-Studiengang Physik an der Universität Göttingen:

Hiermit erkläre ich, dass ich diese Abschlussarbeit selbständig verfasst habe, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe.
Darüberhinaus erkläre ich, dass diese Abschlussarbeit nicht, auch nicht auszugsweise, im Rahmen einer nichtbestandenen Prüfung an dieser oder einer anderen Hochschule eingereicht wurde.

Göttingen, den 5. August 2013

(Nikolai Wyderka)