Variational Calculus and Conservation Laws with Maple

von

Gehrt Hartjen

Diplomarbeit im Fach Mathematik
vorgelegt der
Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinisch-Westfälischen Technischen Hochschule
Aachen

Dezember 2001

angefertigt am
Lehrstuhl B für Mathematik
bei Prof. Dr. Wilhelm Plesken

Abstract:

One of the basic symmetry results in mathematical physics is Emmy Noether's celebrated theorem establishing a correspondence between the symmetries of a variational problem and its conservation laws. For instance, homogenity in time yields conservation of energy, homogenity of space leads to conservation of linear momentum and isotropy of the system implies conserved angular momentum; the Kepler laws can be deduced in a similar way.
To compute these conservation laws explicitly for any variational symmetries (also Bessel-Hagen symmetries), one has to make use of so-called higher Euler operators and thus define homotopy operators, which can be used to compute an "inverse" for the divergence of a current. The theory on these complex and hard-to-use tools was developed just in the last two decades by P.J. Olver and others.
To make these theories accessible for practical computation, their concepts and operators have been implemented in the computer algebra system Maple, as a part of the package `jets`. As the name of the package already indicates, we have used the jet notation to describe differential expressions as elements of the jet space.
On the basis of these implementations, we are now able to compute the conservation laws of a given variational problem explicitly, which will be demonstrated in two examples: First, we are going to discuss conservation laws in elastostatics on a two-dimensional elastic body, leading to Eshelby's energy momentum tensor and as a by-product several divergence identities. Then, we are going to discuss the symmetries (up to third order) of the three-dimensional wave equation and give a complete set of its 94 conservation laws. Similar results have been computed for the four-dimensional wave equation.
Furthermore, we are able to determine all independent second and third order symmetries of the three- and four-dimensional wave equation generated by successive products of recursion operators, verifying and correcting results of Nikitin and others.

Preface

First of all, I want to thank Professor Plesken, under whose guidance this work was produced, for his advice, his helpful comments and his patience. Further, I want to thank the whole staff of Lehrstuhl B for their support and patience, especially Mohamed Barakat for all the long and fruitful discussions and his good cooperation in developing and testing the JETS package. Finally, I want to thank my mother for her great moral support throughout my studies.

Introduction

This diploma thesis arose in a project to make the jet theory and notation of differential equations wider usable by formulating its concepts in the computer algebra system MAPLE, leading to the construction of the package JETS (see [BaHa]), originating from several extensions of the DESOLV package [VuCa], but soon becoming independent and growing in several directions.
The main target of this work has been to make conservation laws available for practical computations, such that the one-to-one correspondence between symmetries of a given variational problem and conservation laws may be found in both directions, especially finding the conservation laws of a variational problem for given symmetries.
With this aim, there have been two parallel parts of this work: On the one hand, the theoretical part, in which, following the book of [Olv], we have retraced the way from Noether's theorem to the explicit conservation laws, which based mainly on the definition and use of homotopy operators. So this concept of variational complexes and homotopy was adapted to our needs in this context (i.e. explicit formulas for horizontal p- and $(p-1)$-forms), where we could add a singularity-respecting case using a variable integration path. All this has been done using jet notation throughout the work and its applications, which makes several notions much clearer and offers useful concepts for practical applications.
On the other hand, a great part of this work consisted in implementing all occurring concepts and algorithms in the computer algebra system MAPLE, as a part of the package JETS that has been developed together with Mohamed Barakat (see [Bar]). Thus, in this joined project, the whole calculus based on jet notation was made accessible for electronic computations, including Euler operator, manipulation of symmetries as vector fields and, of course as well higher Euler operators, currents and homotopy operators, which finally lead to functions that will return the explicit conservation laws for given variational problems. So, an important part of this implementation work has been to prepare a set of universal tools in jet notation to have a full environment for these calculations completely in jet notation, which was not prepared in the MAPLE programming language so far.
These techniques and implementations have been tested on several examples, the most important amidst being an analysis on elastostatics in two dimension and

the example of the three and four dimensional wave equation, which has been used as a kind of standard example throughout this work, mainly because there are references about several properties of this equation (like symmetry operators and conservation laws), which could be used to check the implemented algorithms. These examples have been presented in the form of MAPLE worksheets that can be found in the appendix.
Thus, the main target of this work was to first review and prepare the underlying theory, which had been developed in the 70s and 80s of the twentieth century by Peter Olver and Ian Anderson, using the higher Euler operator first used by Kruskal in 1970, and where some minor misprints in [Olv] could be found and corrected. Secondly, we wanted to make the whole concept accessible for practical computations, which means a useful implementation in computer algebra, forming a part of the JETS package. Although JETS soon became an independent package, it had been necessary to use several functions from [VuCa] in special cases.
As a by-product of these algorithmic developments, we were also able to produce an explicit formula for adjoint operators for differential operators and implement the celebrated Helmholtz operator that can be used to identify Euler-Lagrange systems. A complete list of these functions and procedures with the according notational conventions can be found in the appendix in the form of MAPLE help pages.
Another main interest of this work lay in the analysis of higher order symmetries, especially for the three- and four-dimensional wave equation. An algorithm was formulated and implemented to find the independent symmetry operators of a given order, whose variational subset leads to further conservation laws, i.e. it has been possible to compute the whole set of conservation laws up to order three for the three- and four-dimensional wave equation.[1] Furthermore, the numbers of these independent symmetries resulting form these computations in practical examples, could be used to verify and correct theoretical predictions made by [Nik] and others.
Finally, it is obvious that there may be further applications of the theory and algorithms described in this work, which might include the generalization of some concepts like the homotopy operator and some algorithms to larger cases with more variables, as well as applications of the present tools to other problems, stemming from different directions. Thus, we have not only the theoretic concepts and their accessibility through functions in computer algebra, but as well some immediate algorithms or sequences of algorithms (e.g. the computation of conservation laws), which even could be used without deeper insight in the theory beyond, provided the conditions for the input are satisfied.
All implementations and program parts have been written in the computer alge-

[1]In the four-dimensional case, the third order symmetries had to be restricted to the Poincaré group, as further analysis exceeds the momentarily available computer power.

bra system Maple using a combination of MapleV5.1 and Maple6 on Linux for development. Thus, all procedures are tested on both versions and should be compatible to later versions, as well. The source code and explanations of the functions can be found in electronic form in the appendix of this work.

Contents

Chapter 1

Noether's Theorem

1.1 Jet Spaces

The basic idea of introducing the jet space is to rewrite a system of partial differential equations (PDEs) as a geometric object determined through the vanishing of certain (algebraic) functions. Therefore, we have to "enhance" the basic space (Euclidian space) $X \times U$, where $X \cong \mathbb{R}^p$ with coordinates $x = (x^1, \dots, x^p)$ denotes the independent variables, and $U \cong \mathbb{R}^q$ with coordinates $u = (u^1, \dots, u^q)$ the dependent variables, such that all occurring partial derivatives can be expressed.

1.1.1 Definition (Jet coordinates)
For a given smooth real-valued function $f : X \to U$ of p independent variables and with $u = f(x) = (f^1(x), \dots, f^q(x))$, we can define a jet coordinate of order k, according to the k-th order partial derivatives of f, by

$$u_J^\alpha \hat{=} \partial_J f^\alpha(x) = \frac{\partial^k f(x)}{\partial x^{j_1} \partial x^{j_2} \cdots \partial x^{j_k}}, \tag{1.1}$$

with $J = (j_1, \dots, j_k)$, $|J| = k$ denoting a multiindex of order k. Note that the jet coordinates are motivated by the partial derivative of the given function and are transformed similarly, but do not coincide. (More precisely, their connection is expressed through the *contact of order k* with according contact forms). Thus, these jet coordinates shall be treated just as further coordinates to the basic space.

As the number of different possible k-th order partial derivatives of the above function f is given by

$$p_k = \binom{p+k-1}{k},$$

we need $q \cdot p_k$ jet coordinates u_J^α to represent any k-th order derivatives of all components of f in any point. So $U_k = \mathbb{R}^{q \cdot p_k}$ denotes the Euclidian space of that

dimension with coordinates u^α_J for $\alpha = 1, \ldots, q$ and all $J = (j_1, \ldots, j_k)$, $|J| = k$ multi-indices of order k.
Further, we consider the Cartesian product space

$$U^{(n)} = U_1 \times U_2 \times \ldots \times U_n,$$

whose coordinates represent all partial derivatives of order 0 to n.
Obviously, $U^{(n)}$ is an Euclidian space of dimension

$$q + qp_1 + \ldots + qp_n = q\binom{p+n}{n} \equiv qp^{(n)}. \tag{1.2}$$

Therefore, denoting a point in $U^{(n)}$ by $u^{(n)}$, it has $qp^{(n)}$ components u^α_J.

1.1.2 Definition (Jet space)
The Euclidian space $X \times U^{(n)}$ constructed above with coordinates representing the independent variables, dependent variables and all jet variables up to order n, is called the n-th order **jet space** of the basic space $X \times U$.

1.1.3 Definition (prolongation)
For a smooth function $f : X \to U \quad : \quad x \mapsto f(x)$ consider its graph $\{(x, u)|u = f(x)\}$. In the following, we are going to identify the function and its graph through the notation $u = f(x)$ for smooth f. Thus we can definean induced function $\mathsf{pr}^{(n)} f : X \to U^{(n)}$, given as $u^{(n)} = \mathsf{pr}^{(n)} f(x)$, called the n-th **prolongation** of f.
It is defined through the set of equations

$$u^\alpha_J = \partial_J f^\alpha(x) \tag{1.3}$$

for all $\alpha = 1, \ldots, q$ and all multi-indices $J = (j_1, \ldots, j_k$ with $1 \leq j_k \leq p$ and $0 \leq k \leq n$ of order up to n. Thus, for each point x in X, $\mathsf{pr}^{(n)} f(x)$ is a list of $q \cdot p^{(n)}$ entries representing the values of f and all its derivatives up to order n at that given point.

1.1.4 Example
Let us now apply the above notation to the case $p = 2$ and $q = 1$. Then, $X = \mathbb{R}^2$ with coordinates $(x^1, x^2$, often written as (x, y) and $U = \mathbb{R}$ with the single coordinate u. Constructing the jet spaces, we get $U_1 \cong \mathbb{R}^2$ with coordinates (u_x, u_y) and $U_2 \cong \mathbb{R}^3$ with (u_{xx}, u_{xy}, u_{yy}). Thus, we have $U^{(2)} = U \times U_1 \times U_2 \cong \mathbb{R}^6$, which has coordinates $u^{(2)} = (u; u_x, u_y; u_{xx}, u_{xy}, u_{yy})$ representing the derivatives of u up to order 2.
Similarly, for a smooth function given as $u = f(x, y)$, the second prolongation $u^{(2)} = \mathsf{pr}^{(2)} f(x, y)$ takes the form

$$\mathsf{pr}^{(2)} f(x, y) = \left(f; \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial^2 f}{\partial x^2}, \frac{\partial^2 f}{\partial x \partial y}, \frac{\partial^2 f}{\partial y^2}\right).$$

Notational convention: In lists describing prolongations, also called n-jets, jet coordinates resp. partial derivatives of the same order are usually separated by commas, while those of different order are separated by semicolons.

1.2 Variational Calculus

As before, let us consider the Euclidian space $X \times U$ with corresponding coordinates; further be $\Omega \subset X$ open and connected with a smooth boundary $\partial\Omega$.

1.2.1 Definition (variational problem)
Find all extrema (maximals and minimals) in a class of functions $u = f(x)$ defined over Ω of a given functional

$$\mathfrak{L}[u] = \int_\Omega L(x, u^{(n)}) dx. \tag{1.4}$$

The function $L(x, u^{(n)})$ is called the *Lagrangian* or *Lagrange density* of the variational problem and is usually deduced from the physical properties of the considered system. In many cases, a standard approach to find the Lagrangian would be to take the difference of kinetic energy and potential of a mechanical system. The Lagrangian is a smooth (C^∞) function (horizontal p-form) depending on independent, dependent variables and jet coordinates up to order n. The precise class of functions on which $\mathfrak{L}$ shall be extremalized, depends on the specific conditions of the physical problem, such as boundary conditions or differentiability.

1.2.2 Example (curve length)
A simple example of the above notation would be to determine the shortest curve between two points (a, c) and (b, d) in a plane. So, if that curve is given as graph of a function $u = f(x)$, our problem is described by the functional

$$\mathfrak{L}[u] = \int_a^b \sqrt{1 + u_x^2} dx, \tag{1.5}$$

which would be minimized over the space of C^1-functions with the properties $f(a) = c$ and $f(b) = d$.

If we want to solve such variational problems, we need an approach giving us a way to determine extremals of a given functional. For one-dimensional problems and constant functions, this leads to the vast class of exercises known as "mini-max-problems" or "extremal value problems" from school, being solved by a simple differentiation.
More general, in finite dimensions, the extrema of a real-valued function $f : \mathbb{R}^n \to \mathbb{R}$ can be determined by describing those points where the gradient vanishes. So,

the classical approach to find the gradient (locally) is to examine f under "small"[1] changes in x. Using the standard inner product $\langle x, y\rangle$ on $\mathbb{R}^n$, it leads to

$$\langle \nabla f(x), y\rangle = \left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} f(x+\varepsilon y). \tag{1.6}$$

Transferring this concept to functionals $\mathfrak{L}[u]$ and replacing the standard inner product by the L^2 inner product for vector-valued functions, this leads to the following definition:

1.2.3 Definition (variational derivative)

The **variational derivative** $\delta\mathfrak{L}[u]$ (a list of q entries of the variational problem $\mathfrak{L}[u]$ is uniquely defined by

$$\left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} \mathfrak{L}[f+\varepsilon\eta] = \int_\Omega \delta\mathfrak{L}[f(x)]\cdot\eta(x)dx, \tag{1.7}$$

where $u = f(x)$ is a smooth function on Ω and η is smooth with compact support in Ω.

On the basis of this definition, we get the following proposition:

1.2.4 Proposition

If $u = f(x)$ is an extremal of the variational problem $\mathfrak{L}[u]$, it follows

$$\delta\mathfrak{L}[f(x)] = 0 \tag{1.8}$$

for all x in Ω. If we want to stress the properties of f as a function, we could also write the above equation like $(\delta\mathfrak{L}[f])(x)$ as a function in x, whicle the first notion is referring to the graph.

PROOF. For any η with compact support in Ω and f as above, $f+\varepsilon\eta$ is in the same function space as f. Thus, as f is an extremal of $\mathfrak{L}$, the functional $\mathfrak{L}[f+\varepsilon\eta]$, considered as an expression in ε, has an extremum at $\varepsilon = 0$. Hence, 1.7 has to vanish for any η of compact support, such that the claim holds for any $x \in \Omega$. The same arguments also leads to the uniqueness of the variational derivative $\delta\mathfrak{L}$. □

On the way to a general formula for the variational derivative, we begin with exchanging integration and differentiation of the given expression (which is possible due to the smoothness of all functions) and have

$$\left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} \mathfrak{L}[f+\varepsilon\eta] = \int_\Omega \left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} L(x, \mathsf{pr}^{(n)}(f+\varepsilon\eta)(x))dx. \tag{1.9}$$

[1] As such terms like "big" and "small" always depend on the actual frame of reference and are not invariant under rescaling, this has to be understood strictly infinitesimally.

Using the Notation of u^α_J as partial derivatives of u^α with respect to the multiindex J (and analogue $\partial_J\eta^\alpha$ for the partial derivatives of η^α), this equation can be rewritten as

$$\left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0}\mathfrak{L}[f+\varepsilon\eta]=\int_\Omega\left\{\sum_{\alpha,J}\frac{\partial L}{\partial u^\alpha_J}(x,\mathsf{pr}^{(n)}f(x))\cdot\partial_J\eta^\alpha(x)\right\}dx. \tag{1.10}$$

So this equation can be integrated by parts using the divergence theorem with the boundary terms vanishing.
Before proceeding, we need the following definition:

1.2.5 Definition (Total derivative)
(Throughout this work, the sum is always to be taken over equal upper and lower indices (summation convention).) The expression

$$D_i=D_{x^i}:=\frac{\partial}{\partial x^i}+u^\alpha_{J+1_i}\frac{\partial}{\partial u^\alpha_J} \tag{1.11}$$

denotes the *total derivative* with respect to x^i, $i=1,\dots,p$, where $J+1_i:=(j_1,\dots,j_i+1,\dots,j_p)$. Further define

$$D_J:=(D_1)^{J_1}\cdots(D_p)^{J_p}. \tag{1.12}$$

So, one gets for example

$$u_{xxy}=D_xu_{xy}=D_yu_{xx}=D_{xx}u_y=D_{xxy}u.$$

1.2.6 Remark
In the JETS package [BaHa], the total derivative of a differential expression is available as `totalder`, while the partial derivative is implemented as `partder`.

Further, as L depends on the x through the function $u=f(x)$, we get the identity

$$\frac{\partial}{\partial x^i}\left(\frac{\partial L}{\partial u^\alpha_J}\right)\equiv D_i,$$

where D_i denotes the total derivative as defined above. Therefore, the result can be finally rewritten as

$$\left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0}\mathfrak{L}[f+\varepsilon\eta]=\int_\Omega\left\{\sum_{\alpha=1}^{q}\left[\sum_J(-D)_J\frac{\partial L}{\partial u^\alpha_J}(x,\mathsf{pr}^{(n)}f(x))\right]\eta^\alpha(x)\right\}dx. \tag{1.13}$$

Comparing the latter equation with 1.7, this implies the following useful definition:

1.2.7 Definition (Euler operator)
The *Euler operator* E is defined as a differential operator with q entries given as

$$E_\alpha = \sum_J (-D)_J \frac{\partial}{\partial u_J^\alpha}, \tag{1.14}$$

where the sum is taken over all multiindices $J = (j_1, \dots, j_k)$ with $k \geq 0$, $1 \leq j_i \leq p$ and $(-D)_J = (-1)^k D_J$.

1.2.8 Remark
Although the Euler operator is written out as a sum over infinitely many terms, its application to a Lagrangian L will only have finitely many summands, because L depends on only finitely many jet coordinates, so most partial derivatives will vanish. This principle of "sorting out" those derivatives which might yield a non-vanishing contribution to the result was also used in the implementation of this operator in the JETS package [BaHa]. That function is (obviously) called EULER and was designed to handle as well higher Euler operators which shall be treated later.

Thus, comparing the equations 1.7 and 1.13, we easily see the identity $\delta\mathfrak{L}[u] = E(L)$ with $E(L) = (E_1(L), \dots, E_q(L))$. So, the above proposition 1.2.4 can be simply rewritten as

1.2.9 Theorem (Euler-Lagrange equations)
Any smooth extremal $u = f(x)$ of the variational problem $\mathfrak{L}[u]$ is necessarily a solution of the q Euler-Lagrange equations

$$E(L) = 0 \qquad \textit{i.e.} \qquad E_\alpha(L) = 0 \quad \textit{for all} \qquad 1 \leq \alpha \leq q. \tag{1.15}$$

Note that the vanishing of Euler-Lagrange equations is a necessary, but not a sufficient criterion for extremals, as in practice the critical points of the functional and special solutions from the boundary conditions have to be taken into account which might yield others than the previous smooth solutions.

1.2.10 Example
As a simple example, consider the case $p = q = 1$ with a single function $u = f(x)$. For a n-th order variational problem, the Euler-Lagrange equation takes the form

$$0 = E(L) = \frac{\partial L}{\partial u} - D_x \frac{\partial L}{\partial u_x} + D_x^2 \frac{\partial L}{\partial u_{xx}} - \cdots + (-1)^n D_x^n \frac{\partial L}{\partial u_{x^n}}.$$

If this case is further restricted to a first order variational problem with $L = L(x, u, u_x)$, the Euler-Lagrange equation takes the form usually known from physics courses:

$$0 = \frac{\partial L}{\partial u} - D_x \frac{\partial L}{\partial u_x}$$

Finally, while the Lagrangian L was the equation describing the properties of a variational problem, the Euler-Lagrange equations $E(L) = 0$ shall be that set of differential equations which allow further studies of the problem and its solutions.

1.3 Variational Symmetries

As a next step, we are going to infere some group methods into the calculus of variations. Therefore, we need a useful notion to describe the symmetry of a functional $\mathfrak{L}$, defined as above, where the considered groups will be some local transformation groups acting on an open subset of the domain, on which the functional is described. The purpose of this concept will be, roughly speaking, to transform smooth functions $u = f(x)$ over an appropriate subdomain of Ω into other smooth functions $\tilde{u} = \tilde{f}(\tilde{x}) = g \cdot f(\tilde{x})$ for some g in the local transformation group G. So we are going to define such symmetry groups G in a way that functionals $\mathfrak{L}$ are preserved for above f on a certain subdomain. So, more precisely, we can state:

1.3.1 Definition (variational symmetry)
A local group of transformations G, which acts on a subset $M \subset \Omega_0 \times U \subset X \times U$ is called a group of *variational symmetry* of a functional $\mathfrak{L}[u]$, if for any subdomain Ω with closure $\bar{\Omega} \subset \Omega_0$, $u = f(x)$ is a smooth function on Ω with its graph in M, such that $\tilde{u} = \tilde{f}(\tilde{x}) = g \cdot f(\tilde{x})$ for an $g \in G$ is a well-defined function defined over Ω, and, consequently

$$\int_{\tilde{\Omega}} L(\tilde{x}, \mathsf{pr}^{(n)} \tilde{f}(\tilde{x})) d\tilde{x} = \int_{\Omega} L(x, \mathsf{pr}^{(n)} f(x)) dx. \tag{1.16}$$

A simple example for a variational symmetry, which holds in many of the considered problems, will always be a translation in one coordinate:

1.3.2 Example
So, if we consider the one-dimensional first-order variational problem

$$\mathfrak{L}[u] = \int_b^a L(x, u, u_x) dx,$$

for $X = \mathbb{R}$, the group of translations defined through $(x, u) \mapsto (x + \varepsilon, u)$ will be a variational symmetry group of $\mathfrak{L}$, if L does not (explicitly) depend on x. Thus, with $\tilde{x} = x + \varepsilon$ and $\tilde{u} = u$, any smooth function $u = f(x)$ defined over an interval $[c, d] \subset (a, b)$ leads to $\tilde{u} = \tilde{f}(\tilde{x}) = f(\tilde{x} - \varepsilon)$, which is defined on $[\tilde{c}, \tilde{d}] = [c+\varepsilon, d+\varepsilon]$, still being a subinterval of (a, b) for a sufficiently small ε. Finally, we can verify equation 1.16 using a simple shift of variables:

$$\int_{\tilde{c}}^{\tilde{d}} L(\tilde{f}(\tilde{x}), \tilde{f}'(\tilde{x})) d\tilde{x} = \int_{\tilde{c}}^{\tilde{d}} L(f(\tilde{x} - \varepsilon), f'(\tilde{x} - \varepsilon)) d\tilde{x} = \int_c^d L(f(x), f'(x)) dx$$

Before deducing a handy criterion to check the variational properties of a given symmetry, we first need some notation to describe the symmetries themselves.

For this reason, we are going to shift our view from the transformation groups themselves to the corresponding vector fields, i.e. let us consider the vector field

$$\mathbf{v} = \xi^i(x,u)\frac{\partial}{\partial x^i} + \phi^\alpha(x,u)\frac{\partial}{\partial u^\alpha}, \tag{1.17}$$

defined on an open subset $M \subset X \times U$. Thus, the corresponding one-parameter group is given through $g_\varepsilon = \exp(\varepsilon\mathbf{v})$ and $\mathbf{v}$ is called the *infinitesimal generator* of the symmetry. Note that the functions ξ^i and ϕ^α for $i = 1,\dots,p$ and $\alpha = 1,\dots,q$ may depend on the dependent and independent variables, but not on higher order jet coordinates. The latter would produce generalized vector fields and corresponding higher order symmetries which shall be treated later.

1.3.3 Remark

This concept of using vector fields has also been implemented in the JETS package, where the notation for the infinitesimal generator 1.17 has to be entered like this

$$[[\xi^1,[x^1]],\dots,[\xi^p,[x^p]],[\phi^1,[u^1]],\dots,[\phi^q,[u^q]]].$$

So, for example, the infinitesimal generator of a simple rotation in the (x,y)-plane,

$$\mathbf{v} = x\frac{\partial}{\partial y} - y\frac{\partial}{\partial x}$$

will, in JETS notation, have the form `[[x,[y]],[-y,[x]]]`.

If we want to use this concept of symmetries for differential equations, we also need a prolongation of the vector fields on the n-th jet space, which shall be the generators of the prolonged group action the corresponding jet space. Thus, we can define:

1.3.4 Definition (prolonged vector field)

Let $\mathbf{v}$ be a vector field on an an open subset $M \subset X \times U$ with corresponding local one-parameter group $\exp(\varepsilon\mathbf{v})$. The infinitesimal generator of the prolonged one-parameter group $\mathsf{pr}^{(n)}[\exp(\varepsilon\mathbf{v})]$, being a vector field on the n-jet space $M^(n)$, is called the *n-th prolongation of* $\mathbf{v}$ and will be denoted by $\mathsf{pr}^{(n)}\mathbf{v}$.

So, the prolongation of a given vector field can be explicitly calculated with the following formula:

1.3.5 Theorem (General Prolongation Formula)

The n-the prolongation of a vector field $\mathbf{v}$*, defined as in 1.17 on an open subset* $M \subset X \times U$ *is defined on the jet space* $M^{(n)} \subset X \times U^{(n)}$ *and given by*

$$\mathsf{pr}^{(n)}\mathbf{v} = \mathbf{v} + \sum_{\alpha=1}^{q}\sum_J \phi^{J,\alpha}(x,u^{(n)})\frac{\partial}{\partial u^\alpha_J}, \tag{1.18}$$

where the sum is taken over all multi-indices $J = (j_1, \dots, j_k$ with $1 \le j_i \le p$ and $1 \le k \le n$, while the terms $\phi^{J,\alpha}$ are defined as follows:

$$\phi^{J,\alpha}(x, u^{(n)}) = D_J \left(\phi^\alpha - \sum_{i=1}^p \xi^i u_i^\alpha \right) + \sum_{i=1}^p \xi^i u_{J,i}^\alpha, \tag{1.19}$$

where $u_{J,i}^\alpha$ denotes the jet coordinate obtained by $\partial u_J^\alpha / \partial x^i$, i.e., the appropriate variable is attached to the multi-index J and D_J is the total derivative as defined earlier.

The proof of the general prolongation formula is more or less an infinitesimal version of the chain rule applied to vector fields. The detailed and complete version can be found in [Olv], page 113ff.
Hence, a closer look at equation 1.19 leads to the following definition which will be important in practical dealing with vector fields and symmetries:

1.3.6 Definition (characteristic)
The *characteristic* of a vector field $\mathbf{v}$, given as in equation 1.17, is defined as the q-tuple $Q(x, u^{(1)}) = (Q^1(x, u^{(1)}), \dots, Q^q(x, u^{(1)})$, where

$$Q^\alpha(x, u^{(1)}) := \phi^\alpha(x, u) - \sum_{i=1}^p \xi^i(x, u) u_i^\alpha. \tag{1.20}$$

Using this definition, equation 1.19 takes the form

$$\phi^{J,\alpha}(x, u^{(n)}) = D_J Q^\alpha + \sum_{i=1}^p \xi^i u_{J,i}^\alpha. \tag{1.21}$$

Thus, the general prolongation formula 1.18 can be rewritten as

$$\mathsf{pr}^{(n)}\mathbf{v} = \sum_{\alpha=1}^q \sum_J D_J Q^\alpha \frac{\partial}{\partial u_J^\alpha} + \sum_{i=1}^p \xi^i \left(\frac{\partial}{\partial x^i} + \sum_{\alpha=1}^q \sum_J u_{J,i}^\alpha \frac{\partial}{\partial u_J^\alpha} \right). \tag{1.22}$$

As we easily recognize the definition of the total derivative in the latter bracket, we may now write the prolongation as

$$\mathsf{pr}^{(n)}\mathbf{v} = \sum_{\alpha=1}^q \sum_J D_J Q^\alpha \frac{\partial}{\partial u_J^\alpha} + \sum_{i=1}^p \xi^i D_i, \tag{1.23}$$

and, even further, write down the prolongation in its compact, characteristic form:

$$\mathsf{pr}^{(n)}\mathbf{v} = \mathsf{pr}^{(n)}\mathbf{v}_Q + \sum_{i=1}^p \xi^i D_i, \tag{1.24}$$

where

$$\mathbf{v}_Q := \sum_{\alpha=1}^{q} Q^\alpha(x, u^{(1)}) \frac{\partial}{\partial u^\alpha} \tag{1.25}$$

and, consequently

$$\mathsf{pr}^{(n)} \mathbf{v}_Q = \sum_{\alpha=1}^{q} \sum_J D_J Q^\alpha \frac{\partial}{\partial u_J^\alpha}. \tag{1.26}$$

These latter concepts already show, in how far the definition of the characteristic is useful and practical for writing out prolongations of vector fields.

1.3.7 Remark

Of course, prolongations and characteristics of vector fields (and also generalized vector fields) are also available in the JETS package for practical computations. The prolongation can be accessed through the command `prolvec`, while a characteristic is calculated with `vec2char`.

After this introduction on ideas about vector fields, we shall now return to the concept of variational symmetries and have another look at the criterion for variational symmetries, 1.16. This equation has to be understood as an identity of volume forms (functional p-forms) with the transformation

$$d\tilde{x} = \det \left(\frac{\partial \tilde{x}}{\partial x} \right) \cdot dx,$$

according to the transformation theorem. Thus, the trace of the above transformation matrix can be rewritten as a total divergence, which leads to the following criterion for variational symmetries, after a small, but very important definition:

1.3.8 Definition (Total Divergence)

The *total divergence* of a p-tuple P of smooth differential functions $P^i(x, u^{(n)})$ is defined as

$$\mathsf{Div}(P) := \sum_{i=1}^{p} D_i P^i(x, u^{(n)}), \tag{1.27}$$

where D_i is the total derivative with respect to x^i.

In the JETS package, this function is implemented as `Div`.
As a simple exmaple, consider the 2-tuple $P = (uu_y, uu_x)$. Thus, the total derivative is

$$\mathsf{Div}(P) = D_x(uu_y) + D_y(uu_x) = 2uu_x + 2u_x u_y.$$

1.3.9 Theorem (Infinitesimal Criterion of Invariance)

A connected transformation group G, acting on an open subset $M \subset \Omega_0 \times U$ is a variational symmetry group of a given functional $\mathcal{L}[u]$, if and only if

$$\mathsf{pr}^{(n)} \mathbf{v}(L) + L \mathsf{Div} \xi = 0 \tag{1.28}$$

holds for all infinitesimal generators $\mathbf{v}$ *in* G*, defined as in 1.17 and for all* $(x, u^{(n)}) \in M^{(n)}$*, where* $\mathsf{Div}\xi$ *is the total derivative* $D_1\xi^1 + \ldots + D_p\xi^p$*.*

A detailed proof of this theorem can be found in [Olv] page 257f, which is mainly based on the above transformation.
Further, it is obvious that any symmetry group G of a given functional is also a symmetry group of the according Euler-Lagrange equations $E(L) = 0$ (but not the reverse), which will be important for the practical use.
Another obvious property is the fact that each linear combination of variational symmetries remains variational, as a closer look at the infinitesimal criterion of invariance and at the prolongation formula immediately shows.
On the basis of the infinitesimal criterion of invariance, the term of a variational symmetry can be slightly generalized (or, in some ways, relaxed) with the following definition:

1.3.10 Definition (divergence symmetry)
A vector field $\mathbf{v}$ as in 1.17, acting on $M \subset X \times U$ is called an infinitesimal *divergence symmetry* or *Bessel-Hagen symmetry* or *generalized variational symmetry* of a functional $\mathcal{L}[u]$, if there is any p-tuple of functions $B = (B^1, \ldots, B^p)$ with the property

$$\mathsf{pr}^{(n)}\mathbf{v}(L) + L\mathsf{Div}\xi = \mathsf{Div}(B) \tag{1.29}$$

for all x, u as above.

In how far this generalization is useful and practical will soon become obvious.
For practical testing, we still need the following property, which can be proved by some technical, but still elementary computation:

1.3.11 Remark
An expression (functional p-form) is the divergence of some p-tuple if and only if it is annihilated by the Euler operator, i.e.

$$L = \mathsf{Div}(B) \qquad \Leftrightarrow \qquad E(L) = 0.$$

A detailed proof of this remark can be found in [Olv], page 252f.
Before giving the detailed algorithm to find those variational (and divergence) symmetries from a given list, we shall first explain the purpose of this step and see their further relevance.

1.4 Conservation Laws

As one might already know from physical analysis of, e.g. mechanical, problems, a good characterization and useful piece of information can be the knowledge of those physical properties, which remain stable under changes of time or place, which are called conservation laws. In physics, usual examples for such conserved

quantities are in many examples terms like energy, linear momentum or angular momentum. While the classical approach to these conservation laws normally goes through observation and physical intuition, we are going to use mathematical concepts instead of experimental insight.
Therefore, we first need a reasonable definition for such conservation laws:

1.4.1 Definition (Conservation Law)
For a given system of differential equations $\Delta(x, u^{(n)}) = 0$ (e.g. the Euler-Lagrange equations $E(L) = 0$), any p-tuple of smooth, differential functions $P = (P^1(x, u^{(n)}), \ldots, P^p(x, u^{(n)}))$ with the property

$$\mathsf{Div}(P) = 0$$

on all smooth solutions $u = f(x)$ of $\Delta = 0$ is called a *conservation law* of those differential equations.

In dynamical problems, where the independent variables can be separated in a time-variable (usually called t) and some spatial variables (usually called $x, y, \ldots$), the above term can be split up into $D_t T + \mathsf{Div} X = 0$, where T is called the conserved density and X the conserved flux. In many examples, especially one of these will be of special interest, but not so much the whole conservation law itself.
The above criterion for conservation laws can now be rewritten in the following way:
As one can prove by some technical computations and substitutions, for any conservation law P of a system of differential equations $\Delta(x, u^{(n)}) = 0$, $\mathsf{Div}(P)$ vanishes on all solutions if and only if

$$\mathsf{Div}(P) = \sum_{\nu, J} Q^{\nu, J} D_J \Delta_\nu \tag{1.30}$$

holds for all x, u and for some functions $Q^{\nu, J}(x, u^{(n)})$. The right hand side of this equation may now be integrated by parts, which is equivalent to interchanging the order of Qs and Ds in the above equation, i.e. for each $1 \leq j \leq p$, we have $Q^{j,\nu} D_j \Delta_j = D_j(Q^{j,\nu}\Delta_\nu) - D_j(Q^{j,\nu})\Delta_\nu$. Thus, equation 1.30 can be rewritten as

$$\mathsf{Div}(P) = \mathsf{Div}(R) + \sum_{\nu=1}^{l} Q^\nu \Delta_\nu \equiv \mathsf{Div}(R) + Q \cdot \Delta,$$

where $Q^\nu = \sum_J (-D)_J Q^{J,\nu}$ and R is a p-tuple, which is not needed explicitly here.
Hence, we can now replace P by a (different) conservation law $P - R$ and get

$$\mathsf{Div}(\tilde{P}) = Q \cdot \Delta, \tag{1.31}$$

which is called the *characteristic form* of a conservation law, and Q is called the *characteristic* of the conservation law $\tilde{P}$.
From these last steps, it is obvious that conservation laws are usually not uniquely determined, but only up to a certain degree of additional "trivial" conservation laws. Such "triviality" of conservation laws occurs when an expression fulfills the requirements of definition 1.4.1, but is not a really "useful" conservation law. Firstly, this may happen, if the divergence vanishes on all solutions, just because P itself vanishes for all solutions of the given system. A second type of such trivial conservation laws are the so-called *null divergences*, expressions, whose divergence vanishes for all smooth functions $u = f(x)$, not only on solutions of the given differential equations, but identically. These may lead to some interesting identical, but have obviously no further significance as specific conservation laws, hence the term "trivial".
So, we have to keep in mind that conservation laws are always only uniquely determined up to trivial conservation laws of the two types mentioned above.
Finally, we are now able to link the two concepts of variational symmetries and conservation laws by using the following important theorem which was formulated by Emmy Noether in 1918. According to the symmetries described above, the theorem will first be proved for classical symmetries and later be adapted to symmetries of higher order:

1.4.2 Theorem (Noether's Theorem)
Consider a variational problem $\mathfrak{L}[u] = \int L(x, u^{(n)})$ with a local one-parameter group of (variational) symmetries G with infinitesimal generator

$$\mathbf{v} = \xi^i(x,u)\frac{\partial}{\partial x^i} + \phi^\alpha(x,u)\frac{\partial}{\partial u^\alpha}$$

and corresponding characteristic of $\mathbf{v}$

$$Q^\alpha(x, u^{(1)}) := \phi^\alpha(x,u) - \sum_{i=1}^{p} \xi^i(x,u) u_i^\alpha.$$

Then $Q = (Q^1, \ldots, Q^q)$ is also the characteristic of a conservation law of the Euler-Lagrange equations $E(L) = 0$ of the above variational problem.
Thus, there exists a tuple $P(x, u^{(m)}) = (P_1, \ldots, P_p)$ such that

$$\mathsf{Div}(P) = Q \cdot E(L) \tag{1.32}$$

is the characteristic form of a conservation law for the Euler-Lagrange equations.

Proof. In the present version, the theorem is not difficult to prove. So we start by substituting the general prolongation formula in characteristic form, equation 1.24 into the infinitesimal criterion of invariance, equation 1.28. Thus, the expression

$$0 = \mathsf{pr}^{(n)}\mathbf{v}(L) + L\mathsf{Div}\xi \tag{1.33}$$

takes the form

$$0 = \mathsf{pr}^{(n)}\mathbf{v}_Q(L) + \sum_{i=1}^{p} \xi^i D_i L + L \sum_{i=1}^{p} D_i \xi^i. \tag{1.34}$$

The two latter summands can now be combined using the Leibniz rule (product rule), yielding

$$0 = \mathsf{pr}^{(n)}\mathbf{v}_Q(L) + \mathsf{Div}(L\xi). \tag{1.35}$$

Using 1.26 the first term of this equation can now be integrated by parts and we get

$$\mathsf{pr}^{(n)}\mathbf{v}_Q(L) = \sum_{\alpha=1}^{q} \sum_{J} D_J Q^\alpha \frac{\partial L}{\partial u_J^\alpha} = \sum_{\alpha,J} Q^\alpha \cdot (-D)_J \frac{\partial L}{\partial u_J^\alpha} + \mathsf{Div}(A) \tag{1.36}$$

for some tuple of functions $A = (A^1, \ldots, A^p)$ depending on Q, L and their derivatives.
As we recognize the definition of the Euler operator in the last equation, it follows

$$\mathsf{pr}^{(n)}\mathbf{v}_Q(L) = Q \cdot E(L) + \mathsf{Div}(A), \tag{1.37}$$

and hence, returning to the whole expression, equation 1.35 can be written as

$$0 = Q \cdot E(L) + \mathsf{Div}(A + L\xi), \tag{1.38}$$

which obviously satisfies 1.32 for

$$P = -(A + L\xi). \tag{1.39}$$

□

As one can easily see, the above proof works also for divergence symmetries (Bessel-Hagen symmetries) as defined in 1.3.10. In the beginning, the infinitesimal invariance criterion 1.33 is replaced by equation 1.29, which simply means that the whole proof goes through with $\mathsf{Div}(B)$ instead of 0 as left hand side for some tuple of functions B. Consequently, 1.38 takes the form

$$\mathsf{Div}(B) = Q \cdot E(L) + \mathsf{Div}(A + L\xi), \tag{1.40}$$

and this satisfies the requirements for a conservation law

$$P = B - (A + L\xi). \tag{1.41}$$

So, if one wants to find the explicit conservation laws of a given variational problem, one has to compute the values for A and B in the above formula. This calculation mainly consists of finding the "anti-divergence" of given terms, which is done using homotopy operators and will be discussed in the next chapter.
But before applying Noether's theorem itself, it is necessary to find the relevant

symmetries. So, for a given problem, i.e. all we know about the variational problem is the Lagrangian L and the boundary conditions of the functional $\mathfrak{L}$, we should first determine all symmetries. This is, in our practical examples, usually done with the DESOLV package [VuCa], providing a PDE-solver for our purposes. More precisely, we first determine the defining equations of the point symmetries of the Euler-Lagrange equations given in infinitesimal functions (with `gendef`. As a next step, it is tried to solve the Euler-Lagrange equations (using `pdesolv`), which is usually not possible in general, so a set of solutions in terms of the above infinitesimal defining equations is returned. Finally, we can determine the Lie vectors (infinitesimal generators of the point symmetries) (with `genvec`) from the solutions of the infinitesimal functions. These Lie vectors now have the format which can be further used in the commands of JETS.
Another approach to find the complete set of symmetries can be the direct use of the JETS command `gengen` with appropriate parameters.
After determining all occurring point symmetries, we have to find those which are relevant for finding conservation laws, i.e. we need the variational and divergence symmetries. For this problem, we are not only going to look at the symmetry generators themselves, but it is necessary to consider the vector space generated by all these symmetries and split it up into the subspace generated by the variational resp. the divergence symmetries. This step is necessary, as the input list of symmetries is not uniquely defined, but only up to the generated vector space, i.e. we also have to examine linear combinations of the given vectors to find a maximal set of independent variational (and divergence) symmetries. This operation is done with the following algorithm implemented as `symsplit` in the JETS package:

1.4.1 Algorithm (symsplit)

input: Lagrangian L, list of symmetries in vector field notation
output: (a) a basis of the variational symmetries, (b) a completion of (a) to a basis of the divergence symmetries, (c) the remaining symmetry generators.
algorithm:

1. Produce a list of test terms applying the left hand side of 1.28 (infinitesimal criterion of invariance) to all given vector fields.

2. Filter out the variational symmetries in given form and remove them from the remaining list (by checking the test list for 0 entries).

3. find non-trivial linear combinations of variational symmetries (i.e. solve a linear system of equations on the test terms in the list from step 1 with a Gaussian elimination).

4. make a suitable basis change to separate the linear combinations from the last step and remove the variational symmetries from the test list.

5. Apply the Euler operator to the remaining test entries.
6. repeat steps 2, 3 and 4 to determine the divergence symmetries.
7. return the basis of the variational symmetries and the contributions for divergence and further symmetries in linear independent form.

To make the above rough description of the algorithm somewhat clearer, we need to state a few remarks:

1.4.3 Remark
In step 1 of the algorithm, the original invariance criterion 1.28 is slightly altered to reduce the computation effort for large lists of symmetries: The left hand side of the original criterion

$$\mathsf{pr}^{(n)}\mathbf{v}(L) + L\mathsf{Div}\xi = 0$$

can be rewritten, using the characteristic form of the vector field, as

$$\mathsf{pr}\,\mathbf{v}_Q(L) + \sum_{i=1}^{p} D_i(\xi^i L)$$

and, with a further identity, as

$$D_L(Q) + \mathsf{Div}(L\xi),$$

where $D_L(Q)$ denotes the Fréchet derivative of L applied to the characteristic of $\mathbf{v}$. The detailed definition of the Fréchet operator shall be given in chapter 3. The advantage of this notion, comparing with the original criterion, is here that we only have to compute only one explicit prolongation (the Fréchet derivative) for the whole input list, compared with a prolongation of each single vector field. Thus, the operations which have to be done for any generator can now be reduced to an application of the operator to a characteristic and the calculation of a divergence. On larger examples, this may reduce the computation effort notably.

The list of values produced in step 1 using the above prolongation formula obviously consists of entries, which are either 0 or some differential expressions in expanded form (i.e. written as sum, not as product), while the correspondence between these entries and the symmetries they stem from has to be preserved throughout the computation. Therefore, step 2 makes it very easy to filter out those symmetries that are obviously variational, but there usually remain further ones hidden in linear combinations. To solve this problem, we have to look for nontrivial linear combinations of such list entries to produce further variational symmetries. To achieve this, the differential expressions of our test list are transcribed as rows of a coefficient matrix in the following way: The occurring jet

variables (or products of those) are considered basis elements, in which the coefficient matrix is dynamically generated. This means, as the algorithm scans through the list entries one by one, the basis of occurring variables is also dynamically augmented with any new jet variables that had not appeared in the list entries before. Through this technique, the generated coefficient matrix will, roughly speaking, resemble a lower triangular matrix which is typically also very sparse.
After producing this coefficient matrix, we are not only looking for linear dependent rows, but we also want to parameterize them in terms of the original basis of vector fields (which were corresponding to the list entries and consequently to the rows of the matrix). So we will also need the basis transformation to reproduce the variational symmetries. This is done by transforming the coefficient matrix simultaneously with an identity matrix:

$$\left(\begin{array}{c|c} CM & I_n \end{array}\right), \tag{1.42}$$

where CM denotes the coefficient matrix of the above list of differential expressions, and I_n the identity matrix with as many rows as the coefficient matrix. On this matrix, whose entries should be numerical expressions, typically even small integers, we shall now perform a simple Gaussian elimination, leading to a matrix of the following shape:

$$\left(\begin{array}{c|c} A & B \\ \hline \begin{matrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{matrix} & C \end{array}\right). \tag{1.43}$$

In this gauss-reduced matrix, the submatrix A is an upper triangular denoting the test prolongated values of the non-variational symmetries, which is not of further interest. The submatrix B, however, gives the composition of the non-variational symmetries in terms of the original symmetries, expressed in the rows of the input matrix CM. Therefore, we will need B for reducing the basis by the variational compounds. The lower left 0-matrix indicates by its rows the newly found variational symmetries as linear combinations of given symmetries, while the rows of the submatrix C specify these linear combinations in the original terms.
Thus, the next step will be to add these linear combinations from the rows of C (which are linearly independent due to the Gaussian elimination) with the basis of the vector fields to the list of variational symmetries, which, together with the direct variational symmetries from the first step, yields us a complete basis of the variational subspace. For the remaining vectors, we now have to reduce the

basis by as many vectors as we have added to the variational list. As we want to preserve as many basis vectors as possible in their original form (to keep the output as readable and similar to the input notion as we can), we shall find the zero columns in the submatrix B and remove the corresponding vectors from the remaining basis. It is obvious that those elements don't give any further contribution to the subspace spanned by non-variational symmetries, and from some concepts of simple linear Algebra like Steinitz' basis exchange theorem, it follows that we will get a proper basis by these means. Because of some dimension estimations we easily see that the total number of vectors shall also be preserved.
So, after reducing the number of basis vectors for further examination and having separated the variational symmetries in notation of the given symmetries, we shall now check the differential test expressions for divergence properties, which is done with remark 1.3.11 (step 5 of the algorithm) and repeat the above calculations for the new, smaller coefficient matrix, to find the variational symmetries and the remaining subspace.
In most practical examples, it will be the case that a majority of the given symmetries shall turn out to be variational, so that the original basis is dramatically shortened and the second step (for the divergence symmetries) takes just a fraction of the effort for the first one. Thus, the main consumption of computer resources in practical examples will be the construction of the first coefficient matrix and its Gaussian elimination.
A similar, but a little more sophisticated algorithm will occur when dealing with higher order symmetries and shall be treated in a later chapter. As an example consider the three-dimensional wave equation, which can be found at [ex2]
. Another approach to examine the behaviour of a problem can be to impose certain symmetries (e.g. homogenity, isotropy of some material) and model the Lagrangian in an appropriate way, as it is shown in the example about the deformation of an elastic body [ex1].
So we are now able to find the variational symmetries of a given problem and can relate them to the conservation laws according to Noether's theorem. In the next chapter, we shall develop a general concept to compute the anti-divergence of a given expression, which allows us to compute the conservation laws for given symmetries and Lagrangians explicitly.

Chapter 2

The Homotopy Operator

In this chapter, we are going to show how to compute the conservation laws determined by Noether's theorem explicitly, using homotopy operators, which make available something like an "anti-divergence" to a known divergence expression. These algorithms have been implemented in the functions `classcons` for classical symmetries (i.e. variational symmetries) and `conservation` in the general, but also more complex case, which also works for divergence and generalized symmetries.
Thus, before defining the total homotopy operator itself, we need a few preparations:

2.0.4 Definition (sublists and differences of multi-indices)
Let I and J be multi-indices as defined earlier. Then J is called a *sublist* of I (denoted $J \subset I$), if and only if all indices in J also appear in I with at least the same multiplicity. (Of course this is also true for $I = J$.) In that case, the *difference* of the two multi-indices, $J \setminus I$, is defined as that multi-index obtained by removing each single indices in J from I, respecting the multiplicity.

2.0.5 Example
Consider the multi-index $I = (x, x, x, y, z, z)$. Then $J = (x, x, z)$ is a sublist of I, but not $K = (x, y, y)$, because in the latter case, the multiplicity of y is higher in K than in I (2 vs. 1). Further, the difference is $I \setminus J = (x, y, z)$.

2.0.6 Definition (multinomial coefficient)
For multi-indices I and J with the property $J \subset I$, the *multinomial coefficient* is defined in analogy to the binomial coefficient for integers as follows:

$$\binom{I}{J} := \frac{I!}{J!(I \setminus J)!}, \tag{2.1}$$

where $I!$ is defined as the product of factorials of the occurrence of the single variables in the multi-index I. If J is not a subset of I, the multinomial coefficient is defined as 0.

2.0.7 Example
Consider the multi-indices $I = (x, x, x, y, z, z)$ and $J = (x, x, z)$ as in the last example. Then we get $I! = 3! \cdot 1! \cdot 2! = 12$, since there are three x, one y and two z in I, and analogue $J! = 2! \cdot 1! = 2$, $(I \setminus J)! = 1$. Thus, the multinomial coefficient is $\frac{12}{2 \cdot 1} = 6$.

These two definitions may seem a bit unmotivated and technical at the moment, but will soon prove to be a useful and necessary tool for some differential operators:

2.1 Adjoint Operators

First, we need the concept of adjoints for given differential operators of arbitrary order, which are usually written as matrix operators:

2.1.1 Definition (adjoint operator)
Consider the differential operator given as

$$\mathfrak{D} = \sum_J P_J[u] D_J$$

for differential functions P_J and the sum taken over all multi-indices J. Then the differential operator $\mathfrak{D}^*$ is called its (formal) *adjoint operator*, if it satisfies the equation

$$\int_\Omega P \cdot \mathfrak{D} Q dx = \int_\Omega Q \cdot \mathfrak{D}^* P dx, \tag{2.2}$$

for all differential functions P and Q vanishing for $u \equiv 0$, for each domain $\Omega \subset \mathbb{R}^p$ and every smooth function $u = f(x)$ with compact support on Ω.

To make this definition applicable in practice, we have to write the adjoint operator in an explicit, algorithmic form that makes it possible to calculate it for a given matrix operator $\mathfrak{D}$. The general formula for the adjoint operator of $\mathfrak{D}$, defined as above, is given in the following proposition:

2.1.2 Proposition (Calculating adjoint operators)
For a differential operator $\mathfrak{D} = \sum_J P_J[u] D_J$, the according adjoint operator can be calculated with the following formula:

$$\mathfrak{D}^* = \sum_J (-1)^{|J|} \sum_{J \subset I} \binom{I}{J} D_{I \setminus J} P_J \cdot D_J. \tag{2.3}$$

In this formula, the first sum is taken over all multi-indices for which the differential functions P in the above definition do not vanish, while the second sum is taken over all sublists J of the multi-index I. Furthermore, we use the multinomial coefficient and difference of multi-indices as defined before.

This general formula has also been used for the implementation in the JETS package in the function `Adjoint`. PROOF. An integration by parts of the defining property for adjoint operators, equation 2.2 leads to the formula

$$\mathcal{D}^* = \sum_J (-D)_J \cdot P_J \tag{2.4}$$

for all multi-indices J. Thus, the problem of computing the adjoint operator is reduced to determining the inner product of a total derivative and an arbitrary (differential) expression. So, it follows from the product rule that

$$D_x \cdot f(x, u^{(n)}) = f(x, u^{(n)}) D_x + D_x f(x, u^{(n)}), \tag{2.5}$$

which can be generalized through induction over n to the following expression for successive total differentiation:

$$D_x{}^n \cdot f(x, u^{(n)}) = \sum_{i=0}^{n} \binom{n}{i} D_x{}^i f(x, u^{(n)}) \cdot D_x{}^{n-i} \tag{2.6}$$

for any positive integer n. To generalize the above for multi-indices containing different variables, we can rewrite 2.5 in the way that we apply only one entry of the multi-index, leaving the rest untouched:

$$D_I \cdot f(x, u^{(n)}) = D_{I\setminus i} \cdot \left(f(x, u^{(n)}) D_i + D_i f(x, u^{(n)}) \right), \tag{2.7}$$

which follows directly from the product rule for any element i of the multi-index I. Hence, we are now able to combine the latter two results, equation 2.6 and 2.7, to the following expression, using an induction over the split up multi-index, to obtain

$$D_I \cdot f(x, u^{(n)}) = \sum_{J \subset I} \binom{I}{J} D_{I\setminus J} f(x, u^{(n)}) \cdot D_J, \tag{2.8}$$

containing the multinomial coefficient defined above and summation over all subsets of the given multiindex. Substituting the last result into equation 2.4 immediately yields the general formula. □

2.1.3 Remark
Note that for general differential operators in matrix notation the adjoint is calculated entry-wise on the transposed matrix, i.e. we have $\mathcal{D}^*_{ij} = (\mathcal{D}_{ji})^*$. A differential operator shall be called *self-adjoint*, if $\mathcal{D} = \mathcal{D}^*$, and it is called *skew-adjoint*, if $\mathcal{D} = -\mathcal{D}^*$.

An important application of the adjoint operator is the adjoint Fréchet derivative: For a given p-tuple of differential functions, $P \in \mathcal{A}^p$, the adjoint of its Fréchet derivative has the form

$$(\mathsf{D}^*_P)_{\nu\mu} = \sum_J (-D)_J \cdot \frac{\partial P_\mu}{\partial u^\nu_J}, \tag{2.9}$$

for $1 \leq \mu \leq q$ and $1 \leq \nu \leq q$. In the above formula, we notice the resemblance to the Euler operator, leading to the following identity, which is a useful way to calculate Euler expressions in several cases:

$$E(P) = \mathsf{D}_P^*(1). \tag{2.10}$$

Another important application of the adjoint operator is the inverse problem of variational calculus, i.e. to determine if a given system of differential equations is an Euler-Lagrange system of equations or not. This can be checked with the following operator, implemented in the JETS package as `Helmholtz`:

2.1.4 Definition (Helmholtz operator)
For a given q-tuple of differential expressions Δ^α, stemming form the system of differential equations $\Delta = 0$, the Helmholtz operator is given by the difference of its Fréchet derivative and its adjoint, i.e.

$$\mathcal{H}(\Delta) := \mathsf{D}_\Delta - \mathsf{D}_\Delta^*. \tag{2.11}$$

So we can decide with the following criterion, if a given set of equations are the Euler-Lagrange equations for some variational problem:

2.1.5 Remark (Helmholtz conditions)
A given set of differential equations $\Delta = 0$ is a set of Euler-Lagrange equations of some variational problem, if the Helmholtz operator vanishes identically, $\mathcal{H}(\Delta) \equiv 0$, i.e. if the Fréchet derivative of Δ is self-adjoint.

Note that the other direction of the above statement does not hold, because the Helmholtz operator is not invariant under minor changes of the system Δ, although they preserve the solutions, e.g. exchanging the order of the single equations. Thus it might be difficult to find a strong criterion to identify such variational structures in a given system of equations.

2.2 Currents

On our way to an explicit expression for computing conservation laws, we shall now have another look at the proof of Noether's theorem, especially at the partial integration of formula 1.37. To ressolve the divergence term of this formula, we shall first try to express the whole right hand side of this equation in some divergence form, i.e. we want to the left hand side of that equation with some divergence term of $Q \cdot E(L)$ or something similar. Thus, we need something like the Euler operator which is defined as follows:

2.2.1 Definition (Higher Euler operator)
For any multi-index J and α taking values form 1 to q, define the *higher Euler operator* E_α^J on differential functions L, such that the following condition is

satisfied for any $\mathbf{v}_Q$:

$$\operatorname{pr}\mathbf{v}_Q(L) = \sum_{\alpha=1}^{q}\sum_{J}\left(Q^\alpha E_\alpha^J(P)\right), \tag{2.12}$$

where the second sum is taken over only finitely many terms due to the construction of E_α^J.

A direct evaluation of this condition for higher Euler operator leads to the following explicit formula:

2.2.2 Proposition (Higher Euler operator)
The higher Euler operator, defined as above, is explicitly given through the following formula for all $1 \le \alpha \le q$ and all multi-indices J:

$$E_\alpha^J = \sum_{I\supset J}\binom{I}{J}(-D)_{I\setminus J}\frac{\partial}{\partial u_I^\alpha} \tag{2.13}$$

The sum is taken over all multi-indices which contain J by the meaning of definition 2.0.4, but in practice, there will be only finitely many terms to consider due to the partial derivative applied to some differential function, which will vanish for almost all values of I.

Proof. To prove that the explicit formula for the higher Euler operator satisfies the condition 2.12, we start with the common Leibniz rule

$$R\cdot D_iQ = D_i(QR) - QD_iR$$

for some differential expressions Q and R, which can be generalized by induction to

$$R\cdot D_IQ = \sum_{J\subset I}\binom{I}{J}D_J\left(Q\cdot(-D)_{I\setminus J}R\right).$$

Thus, we can now evaluate the left hand side of equation 2.12 and get

$$\operatorname{pr}\mathbf{v}_Q(P) = \sum_{\alpha,I}D_IQ^\alpha\frac{\partial P}{\partial u_I^\alpha} = \sum_{\alpha,I}\sum_{J\subset I}D_J\left(Q^\alpha\binom{I}{J}(-D)_{I\setminus J}\frac{\partial P}{\partial u_I^\alpha}\right). \tag{2.14}$$

Interchanging the order of the two summations finally leads to the higher Euler operator as stated in 2.13, which also shows the uniqueness of that expression due to the direct consequence of the latter formula from 2.12. □

As we easily see, this above concept is indeed a generalization of the classical Euler operator defined in 1.2.7. For $J = 0$, which denotes the empty multi-index containing no variables, the two definitions coincide by $E_\alpha^0 = E^\alpha$. For this reason, the higher Euler operator has also been implemented in the function `Euler`, and we notice that its evaluation for given P and J can be a quite complex issue.

At this point, we are now able to solve the first part of the problem with calculating conservation laws by stating an explicit formula for the current A from equation 1.37, which has been implemented in the JETS package as `current`:

2.2.3 Proposition (currents)
For a given q-tuple of differential functions Q and a differential function (Lagrangian) L, there is some p-tuple A of differential functions depending on Q and L such that

$$\mathsf{pr}^{(n)}\mathbf{v}_Q(L) = Q \cdot E(L) + \mathsf{Div}(A) \tag{2.15}$$

(see 1.37), where A takes the form

$$A_k = \sum_{\alpha=1}^{q} \sum_{|I|\geq 0} \frac{i_k+1}{|I|+1} D_I \left(Q^\alpha E_\alpha^{J,k}(L)\right) \tag{2.16}$$

for $k = 1, \dots, p$. In this expression, i_k denotes the multiplicity of the variable k in multi-index I and I, k is the multi-index obtained by attaching variable k to that multi-index. Further, the second sum is formally taken over all multi-indices of arbitrary length (starting with the empty list), but this reduces to those (finitely many) terms for which the higher Euler operator does not vanish.

PROOF. The first part of this proposition has already been proved as a part of Noether's theorem 1.4.2 so that here only the explicit form of A remains to be shown. Thus, we simply try to prove it backwards by computing

$$\mathsf{Div}(A) = \sum_{\alpha=1}^{q} \sum_{|I|\geq 0} \sum_{k=1}^{p} \frac{i_k+1}{|I|+1} D_{I,k} \left(Q^\alpha E_\alpha^{I,k}(L)\right) \tag{2.17}$$

and the following substitution of indices: Let $J := (I, k)$, so we get $j_k = i_k + 1$ and $|J| = |I| + 1$, and since trivially $|J| = \sum_k j_k$, the fraction in the above formula simplifies to 1. (Note that this index shift from I to J has also been used in the implementation of `current` for convenience in programming).
Thus, comparing the result of our substitution with the definition of higher Euler operators, equation 2.12, the two expressions coincide up to the summation index for multi-indices, i.e. the arguments $Q^\alpha E_\alpha(L)$ related to multi-indices with $|J| = 0$ differ, immediately proving 1.37. □

Using this proposition, we are now able to compute conservation laws, according to Noether's theorem that only involve classical variational symmetries, while the case for divergence symmetries remains more complicated, as we shall see in the following.
Thus, we can now make use of equation 1.39 and compute conservation laws for given variational symmetries (with characteristics Q) and a given Lagrangian L of a variational problem as $P = -A - L\xi$ with the above formula for the current A. This has been implemented in the JETS package as `classcons` and allows to

find explicit formulas for conservation laws of this classical forms, as it is shown in the example worksheet [ex1]. In the following, we shall now generalize this concept to treat also divergence and generalized Bessel-Hagen symmetries using total homotopy operators.

2.3 Total Homotopy Operators

On our way to the total homotopy operator, it is necessary to note that all operators acting on differential functions, like vector fields, Euler operators or total derivatives, can be understood to act coefficient-wise on differential forms. Thus, we transcribe some differential functions P_J into a total *differential r-form*

$$\omega = \sum_J P_J[u] dx^J, \tag{2.18}$$

where the $dx^J = dx^{j_1} \wedge \ldots \wedge dx^{j_r}$ are elements from the standard basis of $\bigwedge^r T^*X$. Thus, we can write the prolongation as

$$\mathsf{pr}\,\mathbf{v}_Q(\omega) = \sum_J \mathsf{pr}\,\mathbf{v}_Q(P_J) dx^J, \tag{2.19}$$

and the total derivative of such r-form takes the form

$$\mathsf{D}\,\omega = \sum_{i=1}^p D_i(dx^i \wedge \omega) = \sum_{i=1}^p dx^i \wedge D_i\omega, \tag{2.20}$$

where the D_i act only on the coefficients of the differential forms. Hence, we see that it is practically just another notation for the total derivative treated before:

$$\mathsf{D}\,\omega = \sum_{i=1}^p \sum_J D_i P_J dx^i \wedge dx^J. \tag{2.21}$$

As our problem is to find some p-tuple of differential functions B for a given $\mathsf{Div}(B)$, this translates in our new notation into transforming a given p-form into a $(p-1)$-form with the appropriate derivative. This means we need some kind of *interior product*, i.e. an operator

$$\mathsf{I}_Q : \bigwedge^k \to \bigwedge^{k-1}$$

for some q-tuple of differential forms and $1 \leq k \leq p$, while the cases $k = p$ and $k = p - 1$ will be sufficient for our present applications.
This interior product should have the following property for a given r-form ω,

which can be understood as some kind of product rule, and which will soon show to be necessary for our purposes:

$$\mathsf{pr}\,\mathbf{v}_Q(\omega) = \mathsf{DI}_Q(\omega) + \mathsf{I}_Q(\mathsf{D}\omega). \tag{2.22}$$

An operator with these properties can indeed be formulated, mainly making use of higher Euler operators:

2.3.1 Theorem (Interior product)
The interior product I_Q *of a given* $\omega \in \bigwedge^r$ *for* $0 < r \leq p$*, satisfying the required property 2.22 is determined through the general formula*

$$\mathsf{I}_Q(\omega) = \sum_{\alpha=1}^{q} \sum_{|I|\geq 0} \sum_{k=1}^{p} \frac{i_k + 1}{p - r + |I| + 1} D_I \left(Q^\alpha E_\alpha^{I,k} \left(\frac{\partial}{\partial x^k} \lrcorner\, \omega \right) \right), \tag{2.23}$$

where the interior product $\lrcorner$ *for basis elements is defined as*

$$\frac{\partial}{\partial x^i} \lrcorner \left(dx^{j_1} \wedge \ldots \wedge dx^{j_k} \right) = (-1)^{l-1} dx^{j_1} \wedge \ldots \wedge dx^{j_{l-1}} \wedge dx^{j_{l+1}} \wedge \ldots \wedge dx^{j_k} \tag{2.24}$$

for some $j_l = i$*,* 0 *otherwise.*

The complete proof of this theorem for general r-forms is quite complex and technical, but for our present needs it will be sufficient to consider the cases $r = p$ and $r = p - 1$.
For $r = p$, the p-form will look like $\omega = L\, dx^1 \wedge \ldots \wedge dx^n$ and the coefficient L is a differential function, e.g. a Lagrangian. Hence, equation 2.23 reduces to

$$\mathsf{I}_Q(\omega) = \sum_{k=1}^{p} (-1)^{k-1} A_k dx^{\hat{k}}, \tag{2.25}$$

where the coefficients A_k are exactly the components of the current computed in 2.2.3 and $dx^{\hat{k}}$ denotes that basis element of $(p-1)$-forms in which dx^k is left out for a $1 \leq k \leq p$.
Thus, we can use this result to write equation 2.22 as

$$\mathsf{pr}\,\mathbf{v}_Q(\omega) = \mathsf{D}(\mathsf{I}_Q(\omega)) + Q \cdot E(\omega), \tag{2.26}$$

which is obviously the integration by part formula form Noether's theorem, formula 1.37, in homotopy form. This observation immediately yields an interpretation of the current calculated above in the context of interior products and homotopy.
In the case of $r = p - 1$, we have a differential form

$$\omega = \sum_{k=1}^{p} (-1)^{k-1} P_k dx^{\hat{k}}, \tag{2.27}$$

where the coefficients describe a p-tuple of differential functions. Further, the general formula for the interior product, equation 2.23 shall now take the form

$$\mathsf{I}_Q(\omega) = \sum_{j<k} (-1)^{j+k-1} R_{jk} dx^{\hat{jk}}, \tag{2.28}$$

where the coefficients are determined through a skew-symmetric matrix of differential functions R and $dx^{\hat{jk}}$ is a basis element of $(p-2)$ forms constructed in analogy to $dx^{\hat{k}}$ above by leaving out the two differentials referred to.
Thus, it is now easy to see from the general formula 2.23 that the matrix entries R_{jk} are determined by the following equation:

$$R_{jk} = \sum_{\alpha=1}^{q} \sum_{|I|\geq 0} D_I \left(Q^\alpha \left(\frac{i_j+1}{|I|+2} E_\alpha^{I,j}(P_k) - \frac{i_k+1}{|I|+2} E_\alpha^{I,k}(P_j) \right) \right) \tag{2.29}$$

Hence for this special case the prolongation formula 2.22 can be written as

$$\mathsf{pr}\,\mathbf{v}_Q(P_k) = \sum_{j=1}^{p} D_j R_{jk} + A_k, \tag{2.30}$$

with R_{jk} as above and A_k as determined in 2.16. This is again an expression fully in terms of coefficients of those differential forms, or rather of differential expressions in terms of higher Euler operators.
For practical reasons, the implementation of this interior product I_Q in JETS as `interprod` has been limited to these two special cases of $r = p$ and $r = p-1$, which is sufficient for the applications we are going to consider. Furthermore, this implementation could also concentrate on the coefficient structures and needs not involve explicit differential forms.
As a next step, we are going to specialize equation 2.22, the products rule for the interior product for scaling vector fields: For an evolutionary vector field

$$\mathbf{v}_u = \sum_\alpha u^\alpha \frac{\partial}{\partial u^\alpha},$$

its infinite prolongation

$$\mathsf{pr}\,\mathbf{v}_u = \sum_J u_J^\alpha \frac{\partial}{\partial u_J^\alpha}$$

is called the *basic scaling vector field.*
Further, for some smooth differential function $P[u] \equiv P(x, u^{(n)})$, defined on a vertically star-shaped domain, we can conclude the following differentiation with respect to some path parameterized by λ

$$\frac{d}{d\lambda} P[\lambda u] = \sum_{\alpha,J} u_J^\alpha \frac{\partial P}{\partial u_J^\alpha}[\lambda u] = \frac{1}{\lambda} \mathsf{pr}\,\mathbf{v}_u(P)[\lambda u], \tag{2.31}$$

with the notational convention that the term in square brackets is substituted for the dependent variables (and accordingly for their derivatives) for evaluation.
An integration of the latter over a path parameterized by λ (which need not be the standard path, but can also be substituted by some integration path transformation to avoid singularities in the examined function) leads to the expression

$$P[u] - P[0] = \int_0^1 \frac{1}{\lambda} \mathsf{pr}\,\mathbf{v}_u(P)[\lambda u] d\lambda, \tag{2.32}$$

where, as above, the notation $P[0]$ means that all dependent variables are set zero, hence it remains a function depending solely on the independent variables x^i. Recalling that prolongated vector fields act coefficient-wise on total differential forms, as used above, the latter equation is immediately analogue to

$$\omega[u] - \omega[0] = \int_0^1 \frac{1}{\lambda} \mathsf{pr}\,\mathbf{v}_u(\omega)[\lambda u] d\lambda \tag{2.33}$$

with $\omega[0] = \omega(x, 0)$ as above only depending on the basis space X.
Applying this specialization to the scaling group to the interior product defined above, formula 2.22 leads, in analogy to 2.23 to the following (differing only through the substitution $Q = u$):

$$\mathsf{I}_Q(\omega) = \sum_{\alpha=1}^{q} \sum_{|I|\geq 0} \sum_{k=1}^{p} \frac{i_k + 1}{p - r + |I| + 1} D_I \left(u^\alpha E_\alpha^{I,k} \left(\frac{\partial}{\partial x^k} \lrcorner\, \omega \right) \right). \tag{2.34}$$

Thus, using 2.22 and 2.33, we can now formulate the following *homotopy formula* through integration as above:

$$\omega[u] - \omega[0] = \mathsf{DH}(\omega) + \mathsf{H}(\mathsf{D}\,\omega), \tag{2.35}$$

where H denotes the *homotopy operator* defined as

$$\mathsf{H}(\omega) = \int_0^1 \frac{1}{\lambda} \mathsf{I}_u(\omega)[\lambda u] d\lambda, \tag{2.36}$$

showing directly the correspondence between 2.35 and 2.22. The terms in square brackets denote again a substitution of λu for u in all dependent variables and their derivatives. Analogue, $\omega[0]$ is only depending on the independent variables x^i. Thus, $\omega[0]$ is still an ordinary differential form on Ω. So, provided Ω is also star-shaped, we can use the basic identity for k-forms from the de Rham-complex,

$$\omega = dh(\omega) + h(d\omega), \tag{2.37}$$

with the appropriate homotopy operator given by

$$h(\omega) = \int_0^1 \frac{1}{\lambda} (\mathbf{v}_0 \lrcorner\, \omega)[\lambda x] d\lambda. \tag{2.38}$$

Hence, 2.37 can be written as

$$\omega[0] - \omega_0 = dh(\omega[0]) + h(d\omega[0]), \tag{2.39}$$

where $\omega_0 = 0$ for $r > 0$ and $\omega_0 = f(0)$ for $r = 0$, i.e. if $\omega[0] = f(x)$ is a function. As partial derivatives and total derivatives coincide for these forms that do not depend on dependent variables, we can obviously replace the d's in the last equation with the total differential D.
Thus, we obtain the *total homotopy formula* for r-forms ω with $0 \leq r \leq p$ as a summation of equation 2.35 and 2.39 in the form

$$\omega - \omega_0 = \mathsf{D}\mathsf{H}^*(\omega) + \mathsf{H}^*(\mathsf{D}\omega), \tag{2.40}$$

where the *total homotopy operator* is a combination of the two homotopy operators treated before:

$$\mathsf{H}^*(\omega) = \mathsf{H}(\omega) + h(\omega[0]) \tag{2.41}$$

So, if we intend to give the explicit form of this total homotopy operator in analogy to equation 2.36, we can even go one step further and introduce some integration path transformation Q instead of the simple substitution in $[\lambda u]$, which makes it possible to avoid singularities, such that the given form shall be practical usable on all star-shaped domains for an appropriate path.
Thus, consider an integration path transformation $Q(\lambda, x, u)$ depending on all variables, then the case of $\omega[0]$ means that all u's are set equal zero, so that for $\lambda = 0$, Q only depends on x, so we can write $Q(0, x, u) =: g(x)$ as a function of x, and consequently for $\omega_{\lambda=0} = \omega(x, g(x)) =: f(x)$ as another function. Therefore, in analogy to 2.36 the total homotopy operator with a variable integration path can be written as

$$\mathsf{H}^*(\omega) = \int_0^1 \mathsf{I}_{\frac{\partial Q}{\partial \lambda}(\lambda, x, Q^{-1}(\lambda, x, u))}(\omega)[Q(\lambda, x, u)]d\lambda + h\left(\omega(x, Q(0, x, u))\right) \tag{2.42}$$

for some appropriate path transformation Q, taking the place of the λ-substitution.
As a next step towards our practical applications, we are going to extend the total homotopy formula 2.35 to total p forms of the type $\omega = L[u]\, dx^1 \wedge \ldots \wedge dx^n$, using 2.26 for the total derivatives, thus using the prolongation form form Noether's theorem in homotopy form. Thus, we can refer to the results obtained when considering the case $r = p$ for the interior product above, therefore, if the coefficient of the p-form, $L[u] = L(x, u^{(n)})$ is defined on a totally star-shaped domain, equation 2.26 resp. 2.15 resp. 1.37 can be written as

$$L[u] = \mathsf{Div}(B^*[u]) + \int_0^1 u \cdot E(L)[\lambda u]d\lambda, \tag{2.43}$$

where B^* is the sum of two p-tuples $B^*[u] = B[u] + b(x)$, according to the total homotopy operator, while the form of the second component is obvious due to the integration and the components of the first one are just the currents as determined in 2.2.3 (as a consequence of this special case for the interior product, as treated above). Thus, the components of this current B^* can be calculated explicitly as follows for $1 \leq k \leq p$:

$$B_k[u] = \int_0^1 \sum_{\alpha=1}^{q} \sum_I \frac{i_k+1}{|I|+1} D_I \left(u^\alpha E_\alpha^{I,k}(L)[\lambda u]\right) d\lambda \tag{2.44}$$

and

$$b_k(x) = \int_0^1 \lambda^{p-1} x^k L(\lambda x, 0) d\lambda. \tag{2.45}$$

Further, for Euler-Lagrange equations with $E(L) = 0$, formula 2.43 will reduce to $L = \mathsf{Div}(B^*)$ so that we have achieved an explicit formula to compute a divergence form of any given null-Lagrangian.
Finally, we can now manage the case of total $(p-1)$-forms, leading to the solution of our problems, following as a direct consequence in analogy to the above formula from the case $r = p - 1$ for the interior product, as treated above:

2.3.2 Theorem (total homotopy operator)
For a q-tuple of differential functions P with $L = \mathsf{Div}(P)$, there is the general formula

$$P_k = \sum_{j=1}^{p} D_j Q^*_{jk} + B^*_k \tag{2.46}$$

for $1 \leq k \leq p$, where $B^*_k = B_k + b_k$ are given by equations 2.44 and 2.45 and the matrix entries $Q^*_{jk} = Q_{jk} + q_{jk}$ are given as

$$Q_{jk}[u] = \int_0^1 \sum_{\alpha=1}^{q} \sum_I D_I \left(u^\alpha \left(\frac{i_j+1}{|I|+2} E_\alpha^{I,j}(P_k)[\lambda u] - \frac{i_k+1}{|I|+2} E_\alpha^{I,k}(P_j)[\lambda u]\right)\right) d\lambda \tag{2.47}$$

and

$$q_{jk}(x) = \int_0^1 \lambda^{p-2} \left(x^j P_k(\lambda x, 0) - x^k P_j(\lambda x, 0)\right) d\lambda. \tag{2.48}$$

As we see, equation 2.47 resembles 2.29 from the case $r = p - 1$ of the interior product, and all integrations can be performed with an integration path transformation $Q(\lambda, x, u)$ as presented in 2.42.
An important application of this total homotopy operator or, more precisely, the reason why we have treated it at this place, is the property that it allows us to write a null-Lagrangian as a divergence, yielding some "anti-divergence" of a given p-form, and to write null divergences as total curls of some matrix operator given by Q^*_{jk}. The total curl was defined as $P_k = \sum D_j Q^*_{jk}$ for some skew-symmetric matrix operator Q^*_{jk}.
This general homotopy operator for p and $(p-1)$-forms has been implemented

in the JETS package as `homotopy`. As the equations above already show, the computations may get really complex and lengthy even for small examples, but it shows an algorithmic way, which is, through the help of computers, also practically usable, to compute an anti-divergence or anti-curl for a given expression. This implementation also covers the case of non-standard integration paths, as mentioned above, which may be determined algorithmically or assigned by hand. Finally, our main application of this homotopy operator shall now be obvious: With the help of the anti-divergence we are now able to compute the explicit conservation laws of a given variational problem. In Noether's theorem, we had the conservation law P expressed by

$$Q \cdot E(L) = \mathsf{Div}(P) \tag{2.49}$$

for given characteristics of variational (or divergence or generalized variational) symmetries Q and a given Lagrangian L. Thus, the conservation law can be computed in this general case by applying the total homotopy operator to $Q \cdot E(L)$. This has been implemented in the JETS function `conservation` which returns a conservation law for given Q and L.

2.4 Simplifying Conservation Laws

As we have seen before, the conservation laws of a given variational problem are only determined up trivial conservation laws and expressions with vanishing divergences. Nevertheless, especially if concentrating on special components of the conservation laws, e.g. the conserved density, it is very practical and almost crucial for useful results to have a ways to simplify the calculated expressions as far as possible, i.e. cancelling out terms where possible and shorten the resulting terms.
Although there is no normal form for such expressions and no general algorithm for an optimal simplification, there are some useful tactics that have a good chance to simplify differential expressions and transform a good share of examples into the expected form which is understandable to the user.
One way to do this, which is frequently used in the considered examples and has a high potential in simplifying results, is balancing out jet coordinates in products by shifting over indices, using the following proposition, which performs an integration by part to reduce jet expressions with respect to their order:

2.4.1 Proposition (balancing of conserved densities)
differential jet expressions remain invariant up to trivial divergence terms under the following transformation

$$u_I u_J f(x, u^{(n)}) \mapsto u_{I\setminus K}(-1)^{|K|} D_K(u_J f(x, u^{(n)}), \tag{2.50}$$

where the multiindex K is a sublist of J, i.e. $K \subset J$ and f is an arbitrary, smooth function depending on jet variables.

This proposition follows immediately from the product rule for integration by parts and is used to simplify products of jet coordinates by shifting over derivatives.

2.4.2 Example
Under the transformation of the above proposition, we get the following transformation

$$uu_{xy} \mapsto -u_x u_y$$

and the conserved density of energy

$$\frac{1}{2}uu_{xx} + \frac{1}{2}uu_{yy} - \frac{1}{2}{u_t}^2$$

would take the form known from the physics books:

$$-\frac{1}{2}({u_x}^2 + {u_y}^2 + {u_t}^2)$$

(cf. example [ex2]).

In the following algorithm that has been implemented as `intpart` in the JETS package, such transformations are applied whenever the multi-indices denoting jet derivatives can be balanced in a product:

2.4.1 Algorithm: intpart

input:differential expression, protected variables
output: simplified differential expression, transformation **algorithm:**

1. Write the differential expression in expanded form.

2. Scan the expression for summands in which the jet order of one factor (a jet coordinate) is at leat by two higher than the jet order of the remaining factor, not counting the protected variables from above.

3. If a summand is found in step 2, apply formula 2.50 and replace it with the result.

4. Continue scanning and substituting until then end of the expression.

5. Put the whole expression into expanded form, if necessary, and repeat steps 2..5 until no further changes can be made.

Looking at the above scheme, one has to make clear that the algorithm really does terminate after finitely many steps, which is due to the fact that shifting over of derivatives is performed, only if the according jet variables have orders that differ by at least two, while the index is shifted from the higher to the

lower order. With this restriction, we make sure that the order of a jet variable can never be increased to more than the highest order in that summand, and secondly, there can be no mutual exchange of variables. Thus, the algorithm will terminate as soon as all appropriate jet variables are balanced up to differences of one or protected variables. In practice, this procedure will often also make use of cancelling out terms during the intermediate simplifications such that the expression is usually only scanned very few times.
Nevertheless we have to point out that the result of this algorithm is no unique normal form and that not all possible simplifications and cancellations of terms can be found in this way, but `intpart` has shown to be a very useful tool when dealing with conserved densities.
Another way to normalize conservation laws or conserved densities is reducing them by trivial conservation laws of the type $P = P' + \tilde{P}$ where $\mathsf{Div}(\tilde{P}) \equiv 0$, which is done by an algorithm which has been implemented as `divnorm` in the JETS package, by removing such trivial divergence terms and reducing it up to divergences. This is done by scanning a differential expression for terms with vanishing divergence and removing them from the sum to reduce it as far as possible, while those expressions are stored in a different list to keep control on the applied changes. As above, this algorithm will simplify a given expression in most cases, sometimes even drastically, but it does not return a normal form in strict definition.

2.5 Applications to Conservation Laws

With the help of the above homotopy operators, we are now able to compute conservation laws explicitly by using the formulas 1.39 and 1.41. This means, for a given variational problem and given symmetries we are able to render a complete set of conservation laws stemming form those symmetries according to Noether's theorem.
For variational symmetries (i.e. using only `classcons` which is more stable for general terms with several arbitrary parameters, this has been done on the example form elastostatics, referring to the deformation of an elastic body, which can be found in example [ex1]. In that case, the strategy has been to impose special symmetries motivated through physical modelations, e.g. homogenity or isotropy of the considered body, formulate an appropriate Lagrangian to preserve these properties and compute the conservation laws stemming from these assumptions. These calculations are done on general equations for a given number of coordinates (here: two-dimensional) which yield interesting results like Eshelby's celebrated energy-momentum tensor or Euler equations in divergence form. For further details, see [ex1].
A second example, which also includes dealing with divergence symmetries, is the analysis of the three-dimensional wave equation, which has mainly been cho-

sen because its (first order) conservation laws are already explicitly known from physics and have been completely listed in literature (cf. [Olv] p. 285 and [Ibr] p.97). In this case, we use a different approach from above, as we impose the variational problem (i.e. the Lagrangian) and then compute the symmetries of the Euler-Lagrange equation with the help of a PDE solver from [VuCa], which produces a complete list of symmetries in vector field notation. Thus, can now filter out the variational and divergence symmetries, which are seven and three for the three-dimensional wave equation, and finally compute the according conservation laws with the function `conservation`. Finally, this leads to the following result for the wave equation (see MAPLE worksheet [ex2]), only writing out the conserved densities, as those are of main physical interest:

	Characteristic :	**ConservedDensity** :
translations :	u_x	$P_x = u_x u_t$
	u_y	$P_y = u_y u_t$
	u_t	$E = \frac{1}{2}({u_x}^2 + {u_y}^2 + {u_t}^2)$
rotations:	$xu_y - yu_x$	$A = xP_y - yP_x$
	$xu_t + tu_x$	$M_x = xE + tP_x$
	$yu_t + tu_y$	$M_y = yE + tP_y$
dilatation:	$xu_x + yu_y + tu_t + \frac{1}{2}u$	$D = xP_x + yP_y +$ $+\frac{1}{2}uu_t + tE$
inversions:	$(x^2 - y^2 + t^2)u_x +$ $+2xyu_y + 2xtu_t + xu$	$I_x = xD + yA +$ $\frac{1}{2}xuu_t + tM_x$
	$2xyu_x + (y^2 - x^2 + t^2)u_y +$ $+2ytu_t + yu$	$I_y = yD - xA +$ $+\frac{1}{2}yuu_t + tM_y$
	$2xtu_x + 2ytu_y + (x^2 + y^2 + t^2)u_t + tu$	$I_t = (x^2 + y^2)E -$ $\frac{1}{2}u^2 + 2tD - t^2E$

(2.51)

In this table, the first two conservation laws can easily be recognized as conserved momentum, and the third is obviously the expression of conserved energy. The fourth conserved density belongs to the angular momentum, while the physical interpretation of the remaining expressions is not so obvious.
In the same way, we also have computed all 15 conservation laws of the four-dimensional wave equation in the above manner, which can be found in example [ex4]. as above, these conservation laws (resp. their conserved densities) can be

expressed as (with spatial variables x, y, z and time-coordinate t)

Characteristic :	**ConservedDensity** :	
u_t	$E = \frac{1}{2}\left({u_x}^2 + {u_y}^2 + {u_z}^2 + {u_t}^2\right)$	
u_i	$P_i = u_t u_i$	$i \in \{x, y, z\}$
$u_t i + u_i t$	$M_i = tP_i + Ei$	$i \in \{x, y, z\}$
$-u_i j + u_j i$	$A_k = -u_i u_t j + u_j u_t i$	$(i, j, k) = \sigma(x, y, z)$
	$= iP_j - jP_i$	
$u + u_t t + u_x x + u_y z + u_z z$	$D = xP_x + yP_y + zP_z +$	
	$+tE + uu_t$	
$\frac{1}{2}u_t\left(t^2 + x^2 + y^2 + z^2\right) +$	$I_t = 2tD +$	
$+ut + u_x tx + u_y ty + u_z tz$	$+\left(x^2 + y^2 + z^2 - t^2\right)E - u^2$	
$\frac{1}{2}u_i\left(-i^2 + j^2 + k^2 - t^2\right)$	$I_i = 2iD - \left(x^2 + y^2 + z^2 - t^2\right)P_i$	
$-ui - u_t ti - u_j ij - u_k ik$	$= iD + jA_k - kA_j +$	$(i, j, k) = \sigma(x, y, z)$
	$+tM_i + iuu_t$	

(2.52)

In this table, i, j, k are used for cyclic permutations of the spatial variables x, y, z such that several lines yield three conserved densities at once. As before, we recognize conservation of energy, linear and angular momentum in the first lines.

Chapter 3

Higher Order Symmetries

In this chapter, we are going to apply our earlier transformations to generalized symmetries, i.e. after introducing the notation of higher order symmetries and differential operators, we are going to look at the corresponding conservation laws and finally give some kind of classification of these symmetries for wave equations, which shall be compared to theoretical predictions.

3.1 Recursion Operators

As an approach to higher order symmetries, i.e. such symmetries that also depend on jet coordinates of order larger than 1, we shall start by generalizing the vector field notion from an earlier chapter:

3.1.1 Definition (generalized vector field)
A *generalized vector field* [1] $\mathbf{v}$ is a vector field which may depend on higher jet variables, in the following form, with smooth functions ξ^i and ϕ^α depending on all jet coordinates

$$\mathbf{v} = \xi^i[u]\frac{\partial}{\partial x^i} + \phi^\alpha[u]\frac{\partial}{\partial u^\alpha}. \tag{3.1}$$

So, as an example, the expression

$$\mathbf{v} = xu_x\frac{\partial}{\partial x} + u_{xx}\frac{\partial}{\partial u}$$

is a generalized vector field in one dependent and one independent variable, but obviously, it is not vector field in the classical sense of definition 1.17.
The prolongation of such a generalized vector field can be defined exactly as in the classical case, i.e. the general prolongation formula 1.18 can be applied as

[1] In the following we shall always consider generalized vector fields, so the term "generalized" may be omitted.

well, and so the characteristic of such vector field is defined as in 1.3.6. In analogy to the characteristic form of a classical vector field, we are now able to make the following definition:

3.1.2 Definition (evolutionary vector field)
For any q-tuple $Q = (Q^1, \dots, Q^q)$ of differential functions, the generalized vector field defined by

$$\mathbf{v}_Q = \sum_{\alpha=1}^{q} Q^\alpha \frac{\partial}{\partial u^\alpha} \tag{3.2}$$

is called an *evolutionary vector field*, whereas Q is named its characteristic.

Obviously, the prolongation of such a vector field takes the simple form

$$\mathsf{pr}\,\mathbf{v}_Q = \sum_{\alpha,J} D_J Q^\alpha \frac{\partial}{\partial u^\alpha}, \tag{3.3}$$

and we further see that each generalized vector field defined as in 3.1 has an *evolutionary representative* $\mathbf{v}_Q$ to a characteristic Q with

$$Q^\alpha = \phi^\alpha - \sum_{i=1}^{p} \xi^i u_i^\alpha, \tag{3.4}$$

for any $1 \leq \alpha \leq q$. So, if we compare these two vector fields, we shall see that they determine the same symmetries on the basis of the following definition:

3.1.3 Definition (generalized infinitesimal symmetry)
A generalized vector field $\mathbf{v}$ is called *generalized infinitesimal symmetry* of a system of differential equations $\Delta_n u(x, u^{(n)}) = 0$ for any $1 \leq \nu \leq n$, if and only if, for any smooth function $u = f(x)$, it is $\mathsf{pr}\,\mathbf{v}[|delta_\nu] = 0$ for all ν.

Note that we also have to impose some nondegeneracy conditions both on the system Δ itself as well as on its prolongations, which we shall assume implicitly. On the basis of these two definitions, we can state the following:

3.1.4 Proposition
A generalized vector field $\mathbf{v}$ *is a symmetry of a system of differential equations, if and only if its evolutionary representative is.*

Proof. In analogy to the characteristic version of the prolongation formula, equation 1.24, we can write the criterion for generalized infinitesimal symmetries as

$$\mathsf{pr}\,\mathbf{v}[\Delta_\nu] = \mathsf{pr}\,\mathbf{v}_Q[\Delta_\nu] + \sum_{i=1}^{p} \xi^i D_i \Delta_\nu, \tag{3.5}$$

where the first term on the right hand side vanishes on all solutions of the system Δ, and so the rest follows directly from definition 3.1.3.

As a next step, we are now going to get from characteristics of a vector field to differential operators to describe the symmetries of a given variational problem, which will make it possible to construct such generalized vector fields that make sense as symmetries.

3.1.5 Definition (Recursion operator)
For a system of differential equations Δ, a linear differential operator $\mathcal{R} : \mathcal{A}^q \to \mathcal{A}^q$ on the space of q-tuples of differential functions id called a *recursion operator* for Δ, if for each evolutionary symmetry $\mathbf{v}_Q$ of Δ follows that $\mathbf{v}_{\tilde{Q}}$ is an evolution symmetry of Δ as well, where $\tilde{Q} = \mathcal{R}Q$ is the application of the differential operator to the characteristic Q.

It is obvious that this definition can now be used to produce further symmetries of Δ from any known symmetries $\mathbf{v}_Q$ by applying the operator $\mathcal{R}$ recursively, leading to an infinite family of symmetries (though we will later see that these need not be "new" symmetries). So this makes clear where the name *recursion operator* stems from. Furthermore, we are now able to come up with a one-to-one correspondence between characteristics of generalized symmetries and recursion operators in the following way:

3.1.6 Proposition (recursion operators and characteristics)
Let $\mathcal{R} : \mathcal{A}^q \to \mathcal{A}^q$ be a differential operator not depending on the dependent variables u^α or their derivatives and $\Delta[u] = 0$ a linear system of differential equations. Then $\mathcal{R}$ is a recursion operator for Δ (understood as differential operator), if and only if $Q = \mathcal{R}[u]$ is the characteristic of a "linear" generalized symmetry of Δ, where $\mathcal{R}[u]$ is the application of the differential operator to the q-tuple of dependent variables, yielding a characteristic as a q-tuple of differential expressions.

PROOF. To prove this identity, let us first assume that $\mathcal{R}$ is a recursion operator. It follows immediately that $Q = \mathcal{R}[u]$ is a symmetry since $Q_0 = u$ is the characteristic of the trivial scaling group $(x, u) \mapsto (x, \lambda u)$ coming form the imposed linearity of the system. On the other hand, if $\mathbf{v}_Q$ is a symmetry, it follows from the prolongation criterion 3.1.3 and the linearity of Δ that

$$\mathsf{pr}\,\mathbf{v}_q(\Delta[u]) = \Delta[Q] = \Delta\mathcal{R}[u]$$

holds on all solutions. Therefore there exists a differential operator $\tilde{\mathcal{R}}$ with $\Delta\mathcal{R}[u] = \tilde{\mathcal{R}}\Delta[u]$ for all u due to the nondegeneracy of Δ. As Δ and $\mathcal{R}$ are independent of u, it is obvious that we can also choose $\tilde{\mathcal{R}}$ to be independent of u and hence we get the identity of differential operators $\Delta\mathcal{R} = \tilde{\mathcal{R}}\Delta$. Thus, for $\tilde{Q} = \mathcal{R}Q$ and Q the characteristic of a symmetry, it is $\Delta[Q] = 0$ and consequently $\Delta[\tilde{Q}] = \tilde{\mathcal{R}}\Delta[Q] = 0$ is true for all symmetries, and so $\tilde{Q}$ defines another symmetry.
□

To make this correspondence of recursion operators and characteristics accessible for practical use, we need an algorithmic way to find the corresponding recursion operator to a given characteristic. In many (linear) examples, it might look evident and intuitively clear how to transcribe characteristics of given symmetries into the according recursion operators for that $Q = \mathfrak{R}[u]$ is satisfied. Therefore, one will easily see that the differential operator according to the characteristic for a dilatation, u_x, can only be D_x, and hence, a rotation like $xu_y - yu_x$ can be rewritten as $xD_y - yD_x$. Though this may be quite easy for manual computations, we need a more formal way for implementations on the computer.

3.2 Fréchet Derivatives

To make generalized symmetries accessible in the form or recursion operators, we need an algorithmic way to formally rewrite the characteristic of a given symmetry as a differential operator. This can be done with the (formal) Fréchet derivative of differential functions:

3.2.1 Definition (Fréchet derivative)
For an r-tuple of differential functions $P[u] = P(x, u^{(n)}) \in \mathcal{A}^r$, the *Fréchet derivative* of P is defined as the differential operator $\mathsf{D}_P : \mathcal{A}^q \to \mathcal{A}^r$ with the property

$$\mathsf{D}_P(Q) = \left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} P[u + \varepsilon Q[u]] \tag{3.6}$$

for any q-tuple of differential functions $\mathcal{A}^q$, where the expression $P[u + \varepsilon Q[u]]$ means that all dependent variables u^α and their jet derivatives are replaced by $u^\alpha + \varepsilon Q^\alpha$ and their corresponding derivatives. (Note the similarity of this definition with the variational derivative given in equation 1.7).

How this application of the Fréchet operator to a differential function is computed, shows the following example:

3.2.2 Example
Thus, in contrast to the above (linear) example of a rotation, let us now consider $P[u] = u_x u_{xx}$. According to the above definition, this leads to

$$\mathsf{D}_P(Q) = \left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} (u_x + \varepsilon D_x Q)(u_{xx} + \varepsilon D_x^2 Q) = u_x D_x^2 Q + u_{xx} D_x Q.$$

Thus, the corresponding differential operator is $\mathsf{D}_P = u_x D_x^2 + u_{xx} D_x$.

In comparison with the linear example above, we can see that this result is less obvious at first sight, due to the fact that taking of total derivatives produces inner derivatives as a consequence of the chain rule. Hence, we can give a general formula for the Fréchet operator on the basis of the chain rule in several dimensions. The general Fréchet derivative of a r-tuple P is given by the differential matrix operator with entries

$$(\mathsf{D}_P)_{\mu\nu} = \sum_J \frac{\partial P^\mu}{\partial u^\nu_J} D_J, \tag{3.7}$$

with $1 \leq \mu \leq r$ and $1 \leq \nu \leq q$, where the sum is taken over all multi-indices J. Note that the sum consists of only finitely many non-vanishing terms due to the partial derivative (cf. note to the Euler operator 1.2.7).
Thus, according to our simple linear example above, we can now conclude that for linear differential polynomial expressions $P = \Delta[u]$, we get the convenient identity $\mathsf{D}_P \equiv \Delta$. This notion requires that the strict form of Δ is preserved throughout the operation, although the system of differential equations stayed invariant under possible changes, like e.g. permutation of equations.
If we compare the above general formula 3.7 with the prolongation of an evolutionary vector field as given in 3.3, we immediately get to the following proposition:

3.2.3 Proposition
For a tuple of differential functions $P \in \mathcal{A}^r$ and $Q \in \mathcal{A}^q$, it is

$$\mathsf{D}_P(Q) = \mathsf{pr}\,\mathbf{v}_Q(P). \tag{3.8}$$

This proposition offers a useful way to compute prolongations of (generalized) vector fields, especially in those cases where the prolongation of several different vector fields applied to the same differential function (e.g. a Lagrangian) has to be calculated, where the same differential operator (the Fréchet derivative) can be applied to all relevant characteristics. This has been done in the implementation of the symmetry check `symsplit`, see 1.4.3.

3.3 Variational symmetries and conservation laws

On our way to use the generalized symmetries constructed above with the help of recursion operators to obtain further conservation laws, we have to generalized the concept of variational symmetries, which were linked to conservation laws through Noether's theorem. To generalize this theorem for higher order symmetries, we make the following definition:

3.3.1 Definition (variational generalized symmetries)
A generalized vector field $\mathbf{v}$ as defined in 3.1 given by

$$\mathbf{v} = \xi^i[u]\frac{\partial}{\partial x^i} + \phi^\alpha[u]\frac{\partial}{\partial u^\alpha}$$

is called a *variational symmetry* of the functional $\mathcal{L}[u] = \int L(c, u^{(n)})$ if and only if there is a p-tuple of differential function $B \in \mathcal{A}^p$ such that the criterion

$$\mathsf{pr}\,\mathbf{v}(L) + L\mathsf{Div}(\xi) = \mathsf{Div}(B) \tag{3.9}$$

is satisfied for all variables x, u.

As we see, this latter criterion resembles the infinitesimal criterion of invariance for point symmetries, just with the difference that, for generalized symmetries, the divergence symmetries are included in the term *variational*, because the distinction between those types, as in the classical case, would not make any sense here. So, strictly speaking, the variational generalized symmetries are a generalization of the Bessel-Hagen symmetries (divergence symmetries).
As a next step, we have to show that it is sufficient to consider evolutionary symmetries, when searching a complete set of variational symmetries of a given problem. This restriction of symmetries is crucial in so far as we are constructing generalized symmetries through recursion operators, which always leads to evolutionary symmetries. Thus, we have to make sure that no relevant information is lost through this restriction.

3.3.2 Proposition

A generalized symmetry $\mathbf{v}$ is a variational symmetry of a given functional $\mathcal{L}[u]$, if and only if its associated evolutionary vector field $\mathbf{v}_Q$ is variational.

Proof. If we substitute the prolongation formula as given in 3.5 into the invariance criterion 3.9, we get

$$\mathsf{pr}\,\mathbf{v}(L) + L\mathsf{Div}(\xi) = \mathsf{pr}\,\mathbf{v}_Q(L) + \sum_{i=1}^{p} \xi^i D_i L + L \sum_{i=1}^{p} D_i \xi^i,$$

which can be simplified through the product rule to

$$= \mathsf{pr}\,\mathbf{v}_Q(L) + \sum_{i=1}^{p} D_i(\xi^i L) = \mathsf{pr}\,\mathbf{v}_Q(L) + \mathsf{Div}(L\xi). \tag{3.10}$$

Comparing the latter to the right hand side of equation 3.9, we conclude

$$\mathsf{pr}\,\mathbf{v}_Q(L) = \mathsf{Div}(\tilde{B}) \tag{3.11}$$

for $\tilde{B} = B - L\xi$. □

This proof shows immediately that a restriction to $\mathsf{Div}(B) = 0$, according to variational point symmetries, would make the above proposition fail, so that it was indeed necessary to define variational symmetries as above in this wider way. In analogy to the case of point symmetries, we now can also conclude that any variational symmetry of a given variational problem is also a variational symmetry of the according Euler-Lagrange equations $E(L) = 0$. Hence, this leads us to a generalized version of Noether's theorem (see 1.4.2):

3.3.3 Theorem (Noether's Theorem)
A generalized vector field $\mathbf{v}$*, defined as in 3.1.3 with characteristic* Q *is a variational symmetry of a given variational problem* $\mathfrak{L}[u]$ *with corresponding Euler-Lagrange equations* $E(L) = 0$*, if and only if* Q *is also the characteristic of a conservation law* P *for* $E(L) = 0$*, i.e if the equation*

$$\mathsf{Div}(P) = Q \cdot E(L) \tag{3.12}$$

is satisfied. This means that there is a one-to-one correspondence between the equivalence classes of (nontrivial) conservation laws of the Euler-Lagrange equations and the equivalence classes of variational symmetries for the variational problem.

3.3.4 Remark
Thus we see that not only the conservation laws are only determined up to additional trivial conservation laws which vanish on all solutions, as already found in the classical case, but the same is true for variational symmetries. This means that two variational symmetries are obviously equivalent if their characteristics are the same for all solutions of the Euler-Lagrange equations and they may differ by a symmetry whose characteristic vanishes on all solutions. Such symmetry would be called a *trivial symmetry*.

As we have a version of Noether's theorem for generalized symmetries, we are now able to apply it to practical examples, leading to conservation laws stemming from higher order symmetries.

3.4 Further conservation laws of the wave equation

Returning to our example of the wave equation with two spatial coordinates and one time-coordinate, from the last chapter, we are now able to compute further conservation laws than the ten mentioned in the last chapter. To achieve this, we first have to construct higher order symmetries with the techniques described at the beginning of this chapter, using recursion operators.
Before explicitly computing these generalized symmetries, we need the following definition:

3.4.1 Definition (order of symmetries)
The *order* of a generalized symmetry is defined as the order of the highest jet coordinate occurring in its characteristic.

With this definition, it is clear that classical point symmetries always have order one and that the order of a generalized symmetry generated as a successive

product of symmetry operators has at most an order that is equal to the sum of the orders corresponding to the generators. We shall later see that the order has not to be the exact sum of orders, due to relations between such differential operators.
Thus, before applying successive products of these recursion operators, we first have to transform the characteristics of variational (and divergence) symmetries calculated before into differential operators, which can be done either with Fréchet derivatives as described above or simply with the remark for linear expressions. So we get the following ten recursion operators:

$$\begin{array}{ll} D_x, D_y, D_t & \text{(translations)} \\ \mathcal{R}_{xy} = xD_y - yD_x, \mathcal{R}_{xt} = tD_x + xD_t, \mathcal{R}_{yt} = tD_y + yD_t & \text{(rotations)} \\ \mathcal{D} = xD_x + yD_y + tD_t + \frac{1}{2} & \text{(dilatation)} \\ \mathcal{I}_x = (x^2 - y^2 + t^2)D_x + 2xyD_y + 2xtD_t + x & \\ \mathcal{I}_y = 2xyD_x + (y^2 - x^2 + t^2)D_y + 2ytD_t + y & \text{(inversions)} \\ \mathcal{I}_t = 2xtD_x + 2ytD_y + (x^2 + y^2 + t^2)D_t + t & \end{array} \qquad (3.13)$$

On the basis of this table, we can state the following remark resp. definition that we will need later on:

3.4.2 Remark (Poincaré group and conformal group)
The symmetry generators of the wave equation stemming from translations and rotations (in our above table the first six operators) generate a Lie group which is called the *Poincaré group.*
The group generated by all variational[2] symmetries, is called the *(full) conformal group* of the given Euler-Lagrange equations.

As we have seen before, successive products of these operators lead to generalized symmetries of the same variational problem, for example the combination of two rotations leads to the following characteristic of a second order symmetry:

$$\mathcal{R}_{xy}\mathcal{R}_{xt}(u) = -yu_t - yxu_{tx} - ytu_{xx} + x^2u_{ty} + xtu_{xy}$$

With this knowledge, we could principally now compute a complete set of conservation laws for any given order, but before facing this problem, we will first obtain those examples for such conservation laws given by Olver [Olv] on page 334 and compare it with our results. We have to point out at this place that, unlike the results form Olver, our conservation laws have been computed in a strict algorithmic way that does not require any physical intuition or theoretic reasoning related to that specific example. Thus, with the calculations shown in

[2]i.e. generalized variational symmetries (Bessel-Hagen symmetries) of first order, including divergence symmetries

the MAPLE worksheet [ex2], we get the following table:

Recursion operator	Characteristic	Conserved density
D_x^3	u_{xxx}	$u_{xx}u_{tt}$
$D_x^2 D_t$	u_{xxt}	$\frac{1}{2}(u_{xt}^2 + u_{xx}^2 + u_{xy}^2)$
D_t^3	u_{ttt}	$\frac{1}{2}(u_{tt}^2 + u_{xt}^2 + u_{yt}^2)$
$D_x \mathcal{R}_{xy} D_x$	$-yu_{xxx} + xu_{xxy} + u_{xy}$	$u_{xt}(xu_{xy} - yu_{xx})$
$D_x \mathcal{R}_{xy} D_y - \frac{1}{2}D_x^2 - \frac{1}{2}D_y^2$	$-yu_{xxy} + xu_{xyy} - \frac{1}{2}u_{xx} + \frac{1}{2}u_{yy}$	$-u_{xx}(yu_{yt} + \frac{1}{2}u_t)$ $+u_{yy}(xu_{xt} + \frac{1}{2}u_t)$
$D_x \mathcal{R}_{xt} D_x$	$xu_{xxt} + tu_{xxx} + u_{xt}$	$\frac{1}{2}x(u_{xt}^2 + u_{xx}^2 + u_{xy}^2)$ $+tu_{xx}u_{xt}$
$D_x \mathcal{D} D_x$	$xu_{xxx} + yu_{xxy} + tu_{xxt} + \frac{3}{2}u_{xx}$	$\frac{1}{2}t(u_{xt}^2 + u_{xx}^2 + u_{xy}^2)$ $+xu_{xx}u_{xt} + yu_{xx}u_{yt}$ $+\frac{1}{2}u_{xx}u_t$

(3.14)

A few slight differences of the table given here and the results printed in [Olv] stem from a couple of mistypings in the literature which we could find and correct with the algorithm of a direct computation of conservation laws. As before, we have only compared the conserved densities, which are of greater physical interest in many cases. For the full results, see the corresponding MAPLE worksheet.
In analogy to the case of first order symmetries, these are only unique up to trivial conservation laws so that the calculated results were simplified and in a certain way partially normalized with the JETS algorithms `intpart` and `divnorm`. But, according to remark 3.3.4, these conservation laws are also only determined up to conservation laws to trivial symmetries, and so the results may be reduced by terms which vanish for all solutions of the wave equation, i.e. the Euler-Lagrange equations themselves may be substituted into the conserved densities to identify equivalent results.
So we are now able to compute, in principal, infinitely many conservation laws for a given variational problem, e.g. the wave equation. In practice, this is only limited by computational power due to the complex calculations involved in computing the homotopy operators. In fact, this is often still less time-consuming than finding the relevant variational symmetries up to a given order. Thus, the complete set of conservation laws for the three-dimensional wave equation has been calculated in the MAPLE worksheet [ex3] up to the order three, leading to 94 conservation laws, which include then ten well-known of first order and 84 new ones of third order.
A similar calculation was done for the 4-dimensional wave equation (three spatial coordinates and one time-coordinate), but due to practical limitations of computer memory, the explicit calculation of third order conservation laws has been limited to those stemming from the Poincaré group, yielding in total 175 conservation laws, while the full conformal group had to be postponed. These latter

results can be found in the example worksheet [ex4].
As a next step, we shall now see how to find the complete set of variational symmetries of a given order.

3.5 Filtering generalized symmetries

As in the classical case of point symmetries treated in 1.4.1 we will again start with the whole space that is generated by all symmetries and filter out those which are variational. As we have seen before, all variational symmetries have also variational evolutionary representatives $\mathbf{v}_Q$, which can be generated as a successive product of recursion operators.
Therefore, a list of all generalized evolutionary symmetries up to a given order can be obtained by taking the recursion operators corresponding from the first order variational symmetries (through a Fréchet derivative) and determining all successive products of those operators up to the given order, including the identity operator as product of order zero.
But in this vast list of generalized symmetries, it is obvious that we have not produced a system of independent symmetries, because there may be relations and identities between those operators. For example, in the three-dimensional wave equation, the following linear combination of second order symmetries is identical to zero, which can also be understood through the geometric representation of the symmetries involved:

$$\mathcal{R}_{xy}D_t - \mathcal{R}_{xt}D_y + \mathcal{R}_{yt}D_x \equiv 0$$

For convenience, such identities are not checked by calculating with the differential operators, but on the basis of characteristics, because it is obvious that such non-trivial zero-operators like the example above will also have the characteristic 0 and that the other direction is also true.
Thus, we will first have to reduce our system of symmetry operators to a maximal linear independent subsystem, i.e. a basis of the vector space spanned by these operators. Another type of relations can occur through trivial symmetries, where the corresponding differential operator is not identical to zero, but nevertheless vanishes on all solutions of the Euler-Lagrange equations. Therefore, the symmetries also have to be reduced up to equivalence with the equation itself.
Before going into the algorithm itself, first a little definition:

3.5.1 Definition (Relations)
A linear combination of recursion operators that is equivalent to the zero operator (i.e. leads to a characteristic identical to zero), i.e. that vanishes for all values, is called an *absolute relation*, while a linear combination of recursion operators that vanishes only on solutions of the Euler-Lagrange equations is called a *relation modulo reduction* or *relation on solutions.*

So this leads to the following algorithm to filter generalized symmetries which has been implemented in the JETS package as `symtestgen`:

3.5.1 Algorithm: (symtestgen)

input: Euler-Lagrange equations as differential operators, a list of recursion operators **output:** A list with the following entries:

1. absolute relations
2. (further) relations on solutions
3. variational symmetries
4. variational symmetries with reduction expressed in old basis, if possible
5. remaining (i.e. non-variational) symmetries
6. remaining symmetries with reduction expressed in old basis, if possible
7. characteristics of variational symmetries (for test reasons)

algorithm

1. determine a list of characteristics corresponding to the recursion operators from the input.
2. Filter trivial relations from the symmetry list (i.e. those with characteristic 0).
3. Find all absolute relations between the operators (by solving a system of linear equation in the characteristics from step 1 with a Gaussian elimination).
4. Write these relations as linear combination of recursion operators from the input and remove them from the list of characteristics.
5. Reduce the remaining symmetries up to equivalence on solutions of the Euler-Lagrange equations.
6. Where possible, express the reduced operators in the original basis.
7. Find all further relations (after reduction) in the remaining symmetries (in analogy to step 2).
8. Write those new relations as linear combination of the differential operators and remove their characteristics from the list.

9. Find all variational symmetries by applying the Euler operator to the characteristics.

10. Search for relations between the variational symmetries and return a list of independent variational symmetries, both in the original basis and in the new basis after substitution.

11. Return the remaining independent symmetries in both bases.

To make the above scheme a bit more understandable, we have to explain some of the steps:
In general, the algorithm always refers to the symmetries in a parallel way by their recursion operators given in the input and in their characteristics, sometimes also with a different notion of their operators. The operator notation is used for the output to give, for instance, a relation between symmetries as linear combination of operators (a characteristic simplified to 0 would not give the user much information about the specific relation). On the other hand, the characteristics are crucial for the actual calculations, as they are much easier to handle and to simplify expressions, which would be a much higher effort in operator notion. In the input, also the Euler-Lagrange equations are required in operator notation in expanded and simplified form, which is necessary to perform the reduction up to trivial symmetries. This operator equation can be obtained by simply determining the Fréchet derivatives of both hand sides from the original Euler-Lagrange equations and eliminating it to one non-trivial variable, i.e. for instance, the 3-dimensional wave equation would be written as $D_t^2 = D_x^2 + D_y^2$.
The fact that some results occur twice in the output has mainly been implemented for test reasons, though are several cases where the basis change after the reduction of trivial symmetries shows its results here. The characteristics of variational symmetries, returned as last argument, can be used to test the given operators, though no deviation has been noticed there so far.
The reduction modulo trivial symmetries means that, to eliminate such symmetries which are equivalent on all solutions of the Euler-Lagrange equations (but not elsewhere), the equations, which can be seen as operator relations themselves, are substituted in all recursion operators, where appliable. Thus, treating the act of substitution as a linear differential operator (which is obviously allowed to do), we get a list of changed operators, where many summands will remain unchanged. Now it is theoretically not clear if each of these substituted recursion operators (or, more precisely, linear combination of substituted operators, which have been shown to be linear independent before) can be expressed as a (different) linear combination of the original basis operators, i.e. if the operation of substituting the Euler-Lagrange equations is closed on the subspace generated by those symmetry operators. This rewriting of linear combinations is done by a Gauss algorithm on the corresponding characteristics. In practical examples, we could see that there are examples where the reduction leads to a linear combination of

those operators (e.g. all trivial and semi-trivial cases), but also those where no representation in the original basis could be found. In those cases, we have used the notation of an operator $\mathfrak{R}$ to indicate that a reduction/substitution has taken place.
Three times during the course of the algorithm, the list of symmetries is scanned for relations, i.e. linear combinations between the given operators, which is done in the following way (in analogy to algorithm 1.4.1): To find linear dependencies, the list of characteristics (resp. the Euler operator applied to those dependencies in case of variational symmetries) in expanded form is considered as system of linear equations. So these characteristics generate a vector space whose kernel parametrizes the occurring relations and a basis of this vector space yields a complete system of independent symmetries. Thus, the characteristics are first rewritten as a coefficient matrix of that homogeneous system, each element of the list forming one row. So the differential expressions are written as coordinate rows where the relevant jet coordinates or products thereof form a basis. To avoid a matrix that is larger than necessary, this basis of jet coordinates is generated dynamically, i.e. the characteristics are scanned one by one and the new jet variables are added dynamically to the present basis. This also means that the coordinate rows transcribed before might have to be adapted by adding zero columns. Thus, the coefficient matrix grows both in columns and rows while being determined, and through this algorithm, it get roughly somewhat like a lower triangular shape, which can, in combination with the fact that the matrix is also sparse, be an advantage in the following Gaussian elimination. So this matrix has as many rows as there have been recursion operators in the lists to be checked and an undetermined, possibly larger or smaller, number of columns. To obtain a record on the basis transformation and the linear combinations used, the following gauss algorithm is simultaneously performed on an identity matrix (I_n) with as many rows and columns as the coefficient matrix (CM) has rows. Thus we start with a matrix of the following shape:

$$\left(\begin{array}{c|c} \\ CM & I_n \\ \\ \end{array}\right) \tag{3.15}$$

As a next step, the matrix is processed with the Gauss algorithm, which is stopped after the last column belonging to the original coefficient matrix, if necessary. This makes sure the linear combinations given in the basis transformation matrix (right hand side of the joint matrix) are changed as little as possible. Thus, we have achieved a matrix of the following shape:

$$\left(\begin{array}{c|c} A & B \\ \hline \begin{matrix}0 & \dots & 0\\ \vdots & \ddots & \vdots \\ 0 & \dots & 0\end{matrix} & C \end{array}\right) \tag{3.16}$$

This matrix consists of submatrices with the following functions: The upper triangular matrix A gives a basis of the (linear) independent characteristics resp. symmetry operators, with the basis transformation being given in the matrix B. Matrix A will furthermore only be used to determine its rank (which is a very easy calculation for a fully gauss-reduced upper triangular matrix), which yields the number of independent symmetries. The rows of matrix B could now be used to write down a basis for these symmetries, but we have chosen to take a basis which preserves as many of the original symmetry operators as possible. Thus we can, by using Steinitz' basis change theorem, choose such elements form the original list which also contribute to the basis, which is technically done by removing the others, which are indicated in the matrix B as zero columns. So this basis length not only corresponds to the (column) rank of the basis transformation matrix, but also to the zero columns themselves due to the construction of this matrix. Hence, we get a basis of the independent symmetries by removing those operators form the original list which corresponding to zero columns in matrix B.
To write down the relations in explicit form, we need a parameterization of the Kernel of matrix A, which is given in submatrix C. Thus, by applying its rows to the original generating system of the vector space, we are able to give a list of the relations in terms of the original symmetry operators. Obviously, these relations are only unique up to linear (and principally also algebraic) combinations.
So this step of the algorithm, which is performed three times with a different system of operators (which decreases dramatically from step to step), separates the list of given operators into two lists, one consisting linear independent operators and one relations between those differential operators. Note that in practical examples, the matrices can become extremely large (in our case up to about 1600 rows and maybe even much more columns, including the identity matrix), which makes the Gaussian elimination over the integers to a serious problem for MAPLE, both in time as also in memory consumption terms, as the matrix entries can become quite unpleasant, even if the original coefficients were small integers.
For finding the variational symmetries, we use the following remark:

3.5.2 Remark (variational symmetries)
According to the definition of a variational generalized symmetry, 3.3.1, the fol-

lowing criterion has to be satisfied:

$$\mathsf{pr}\,\mathbf{v}(L) + L\mathsf{Div}(\xi) = \mathsf{Div}(B)$$

This can be rewritten using equation 3.10 and leads to an equivalent criterion based on equation 3.11, taking into account the remark form an earlier chapter that a divergence term can be identified by its vanishing Euler-expression:

$$E(\mathsf{pr}\,\mathbf{v}_Q(L)) = 0 \tag{3.17}$$

This latter can be rewritten as

$$E(\mathsf{D}_L(Q)) = 0,$$

using equation 3.8, which is finally rewritten in characteristic form and leads to

$$E(Q \cdot E(L) + \mathsf{Div}(A)) = E(Q \cdot E(L)) = 0, \tag{3.18}$$

the final criterion which is actually checked for the given symmetries.

The advantage of the criterion formulated in this remark is the fact that we don't have to compute the prolongation for each single vector field, but we need the Euler-Lagrange equations, which are already known form the beginning, so that the computational effort reduces to one inner product and taking the Euler operator of one expression for each symmetry operator. So this is a much more efficient way than to determine the actual prolongation.

3.5.3 Remark (present limitations)
Though the algorithm formulated above works for arbitrary numbers of variables, the implementation as `symtestgen` has been limited to only one dependent variable (for the present), which is sufficient for many examples, while several dependent variables would increase the size of the matrices used in the Gauss algorithm dramatically, which might exceed the facilities of MAPLE's gauss algorithm even for smaller examples.

So, the above algorithm allows us to split a list of given symmetry operators up into relations and independent operators, also delivering a complete system of variational symmetries, which makes it possible to compute all conservation laws stemming form higher order symmetries. Hence, we have to point out that a usable result depends on a correct input for the list of differential operators to consider. It is obvious that all symmetries of the considered group up to a certain order have to be specified, as otherwise relations of mixed order could not be determined, leading to a wrong result in independent symmetries.

3.6 Counting symmetries of the wave equation

Besides explicitly computing conservation laws of first and higher order, the algorithm shown above can also be used to check theoretical predictions on the number of symmetries:
Considering the wave equation (in three and in four dimensions), it can be interesting to find out how many independent symmetries of a given order n exist, which include both variational and non-variational symmetries. Thus, before looking at the experimental results obtained with the above algorithm and using the JETS package, we shall first see what is predicted by theory: For the Poincaré group (i.e. we only consider those symmetries stemming from translations and rotations, but not dilatations and inversions), the number of independent symmetries can be estimated through combinatoric reasoning and calculations, as shown in [Nik]. If we consider the solutions of the wave equation of order m as polynomials of order j in the symmetry operators, we get a system of linear homogeneous algebraic equations where the number of equations of order k is given by

$$N_{eq}^k = \binom{j+m}{j+1}\binom{k+m-1}{k}, \tag{3.19}$$

and the number of unknowns in these equations are

$$N_{uk}^k = \binom{j+m-1}{j}\binom{m+k}{k+l}. \tag{3.20}$$

To determine the number of independent solutions, we now need the number of arbitrary parameters, which can be calculated with the following formula through elementary combinatorics, which differs from the one printed out in [Nik]:

$$\hat{N}_j^m = \sum_{k=0}^{j-1}(N_{uk}^k - N_{eq}^k) + \binom{j+m-1}{j}. \tag{3.21}$$

This formula can be simplified to the following expression of binomial coefficients:

$$\hat{N}_j^m = \frac{1}{m}\binom{j+m-1}{m-1}\binom{j+m}{m-1}. \tag{3.22}$$

For $m = 3$, i.e. the three-dimensional wave equation treated before, this expression simplifies to

$$\hat{N}_j^3 = \frac{1}{3}(j+1)(2j+1)(2j+3), \tag{3.23}$$

which coincides with the formula given in [Olv] p.317. In this case, the evaluations for symmetries of order $0, ..4$ would therefore be $1, 10, 35, 84, 165$.
If we consider the four-dimensional wave equation, i.e. $m = 4$, we get the expression

$$\hat{N}_j^4 = \frac{1}{12}(j+1)^2(j+2)^2(2j+3) \tag{3.24}$$

with first values for low orders $1, 15, 84, 300, 825$.
Hence, this now enables us to determine the number of linear independent symmetry operators of a given order. As we see, this number is equal to the number of independent solutions of order $j = n - 1$ and those of order $j = n$. Thus, the number of independent symmetry operators of order n is given by

$$N(n,m) = \hat{N}_n^m + \hat{N}_{n-1}^m, \tag{3.25}$$

which can be simplified to

$$N(n,m) = frac2n^2 = 2nm + m(m-1)m(m-1)^2 \binom{n+m-2}{m-2} \binom{n+m-1}{m-2}. \tag{3.26}$$

Note that the square in the denominator is missing in [Nik]. In the case $m = 3$, the above formula can be written as

$$\frac{1}{6}(n+2)(n+1)(n^2+3n+3), \tag{3.27}$$

and the number of independent symmetries in the orders from 0 to 5 can be expected as $1, 7, 26, 70, 155, 301$.
For $m = 4$, equation 3.26 becomes

$$\frac{1}{72}(n+3)(n+1)(n+2)^2(n^2+4n+6), \tag{3.28}$$

with symmetries of low orders in numbers of $1, 11, 60, 225, 665, 1666$.
Turning form the Poincaré group to the conformal group of the wave equations, i.e. considering as generators also dilatations and inversions besides translations and rotations, we are not yet able to state a general formula corresponding to 3.26 for the Poincaré group, but we have expressions in analogy to equations 3.27 and 3.28:
For the three-dimensional wave equation, that is

$$N(n,3) = \frac{1}{3}\sum_{j=0}^{n}(2j+1)(j+1)(2j+3) = \frac{1}{6}(n+1)(n+2)(2n^2+6n+3), \tag{3.29}$$

where the numbers of independent symmetry operators for orders 0..5 would thus be $1, 11, 46, 130, 295, 581$.
In the case of the four-dimensional wave equation, the corresponding formula is

$$\frac{1}{12}\sum_{j=0}^{n}(j+1)^2(j+2)^2(2j+3) = \frac{1}{36}(n+3)^2(n+2)^2(n+1)^2, \tag{3.30}$$

with first numerical evaluations $1, 16, 100, 400, 1225, 3136$.
Especially the last sequence shows that the numbers of symmetries rises strongly

with the order of the symmetries, as we would also expect from general intuition. On the other hand, these values also show that the number of independent symmetries is still really small compared to the number of relations (that can be obtained as difference of the total number of operators and the above values). In these estimations, we have not yet distinguished between variational and non-variational independent symmetry operators and not between absolute and reduced relations, either.
So we are now able to check the above theoretical values experimentally by determining the symmetries of the wave equation for given orders in several cases. This is done by filtering the complete set of symmetry operators with the help of the algorithm `symtestgen`, where only the number of results is of special interest. (It can be considered some by-product on the way to calculating all conservation laws.) These calculations, which will become quite challenging for the computer and the implementation of the Gauss algorithm used during our algorithm if the examples get larger, can be seen in the worksheets [ex3] and [ex4]. For a subsummation of the resulting numbers see the following tables:

3-dimensional Wave Equation

Poincaré Group			
	1st order	2nd order	3rd order
absolute relations	-	16	182
relations after reduction	-	1	7
variational symmetries	6	6	50
independent symmetries	-	20	20
total	6	43	259

Conformal Group			
	1st order	2nd order	3rd order
absolute relations	-	51	886
relations after reduction	-	14	95
variational symmetries	7+3	10	94
independent symmetries	1	36	36
total	11	111	1111

4-dimensional Wave Equation

Poincaré Group			
	1st order	2nd order	3rd order
absolute relations	-	50	875
relations after reduction	-	1	11
variational symmetries	10	10	175
independent symmetries	-	50	50
total	10	111	1111

Conformal Group			
	1st order	2nd order	3rd order
absolute relations	-	121	?[1]
relations after reduction	-	20	?[1]
variational symmetries	11+4	15	*315*[1]
independent symmetries	1	85	*85*[1]
total	16	241	3616

So these experimental results could be used to correct some equations given in [Nik] and check the outcome of those estimations, as well as similar formulas in [BSSS]. Furthermore, we can produce a complete system of independent symmetry operators, corresponding to the ones given by [Ibr].
Using a more sophisticated Gauss algorithm than the one implemented in MAPLE or a less memory-consuming fashion of storing matrices, it should be easily possible to exceed the above results to higher orders and higher dimensions of the wave equation, as well as to other variational problems with more than one dependent variable.

[1]As this example exceeds the computer resources on the used system, the indicated values have been taken from theoretical predictions.

Appendix A

Notational Conventions

In the following examples, we are going to use the *jet notation* for differential expressions.

```
> with(jets):
```

The coordinates of this jet space are the independent variables x^1,.., x^p...

```
> ivar:=[x,y];
```

$$ivar := [x,\, y]$$

...the dependent variables u^1,.., u^q ...

```
> dvar:=[u,v];
```

$$dvar := [u,\, v]$$

...and the jet coordinates of certain degrees:

```
> jetcoor(1,ivar,dvar);
```

$$[u_x,\, u_y,\, v_x,\, v_y]$$

which altogether are the coordinates of the jet space:

```
> alljets(2,ivar,dvar);
```

$$[x,\, y,\, u,\, v,\, u_x,\, u_y,\, v_x,\, v_y,\, u_{x,x},\, u_{x,y},\, u_{y,y},\, v_{x,x},\, v_{x,y},\, v_{y,y}]$$

These jet coordinates correspond to the (total) derivatives of functions:

```
> totalder(u,[x],ivar,dvar);
```

$$u_x$$

and with prolongations:

```
> totalder(u[x],[y],ivar,dvar);
```

$$u_{x,y}$$

To describe symmetries, we are going to use infinitesimal generators of vector fields of the form $v = \left(\sum_{i=1}^{p} \xi_i(x,\, u)\, (\frac{\partial}{\partial x_i}\ )\right) + \left(\sum_{\alpha=1}^{q} \phi_\alpha(x,\, u)\, (\frac{\partial}{\partial u_\alpha}\ )\right)$ where, e.g. a vector field

```
> vf:=[[x,[y]],[-y,[x]]];
```

$$vf := [[x,\, [y]],\, [-y,\, [x]]]$$

denotes the infinitesimal symmetry generator $x\,(\frac{\partial}{\partial y}\ ) - y\,(\frac{\partial}{\partial x}\ )$ of a rotation. A prolongation of the above vector field yields:

```
>  pv:=prolvec(vf,1,ivar,dvar);
```

$$pv := [[-y,\ [x]],\ [x,\ [y]],\ [-u_y,\ [u_x]],\ [u_x,\ [u_y]],\ [-v_y,\ [v_x]],\ [v_x,\ [v_y]]]$$

with the characteristic of this prolonged vector field

```
>  vec2char(pv,ivar,dvar);
```

$$[y\,u_x - x\,u_y,\ v_x\,y - v_y\,x]$$

Appendix B

Conservation Laws of the Three-dimensional Wave Equation

```
> restart;
> libname:=libname,"/usb/gehrt/maple":
```

(Specify the path where the packages jets and Desolv are saved)

```
> with(Desolv):
> with(jets):

Warning, the names colterm, comtab and genvec have been redefined
```

Consider the propagation of a wave in one time-coordinate t and two spatial coordinates x, y (independent variables) with elongation u (dependent variable), e.g. a surface wave on a water bassin.

```
> ivar:=[t,x,y]; dvar:=[u];
```

$$ivar := [t,\, x,\, y]$$

$$dvar := [u]$$

From the ansatz for the Lagrangian

```
> L:=1/2*(u[t]^2-u[x]^2-u[y]^2);
```

$$L := \frac{1}{2}\,{u_t}^2 - \frac{1}{2}\,{u_x}^2 - \frac{1}{2}\,{u_y}^2$$

we obtain the Euler-Lagrange equations from the relation $\mathrm{E}(L) = 0$ in the following way:

```
> EL:=op(Euler(L,ivar,dvar));
```

$$EL := -u_{t,\,t} + u_{x,\,x} + u_{y,\,y}$$

```
> DE:=-EL=0;
```

$$DE := u_{t,\,t} - u_{x,\,x} - u_{y,\,y} = 0$$

We can solve the resulting equation, known as the three-dimensional wave equation, in so far that we can determine its infinitesimal symmetries explicitly:

```
> de :=[ ind2eqn(DE,ivar,dvar) ]:
```

```
> le:=gendef(de,dvar,ivar):
> sol:=pdesolv(le[1],le[2],le[3]):
```

The infinitesimal generators of symmetries are written down as vector fields using the following notation:

```
> lv:=genvec(sol[3],sol[4],le[3]):
> nops(lv);
```

$$12$$

```
We remove the term depending on a general function from the list of
symmetries and get:
```

```
> lv:=subsop(1=NULL,lv):
> lv := [[[y, [t]], [t, [y]]], [[-2*y*t, [t]], [-2*y*x, [x]],
> [-y^2+x^2-t^2, [y]], [u*y, [u]]], [[u, [u]]], [[-t^2-y^2-x^2,
[t]],
> [-2*x*t, [x]], [-2*y*t, [y]], [u*t, [u]]], [[y, [x]], [-x, [y]]],
[[1,
> [t]]], [[1, [y]]], [[-x, [t]], [-t, [x]]], [[-x*t, [t]],
> [1/2*y^2-1/2*t^2-1/2*x^2, [x]], [-y*x, [y]], [1/2*u*x, [u]]],
[[-t,
> [t]], [-x, [x]], [-y, [y]]], [[1, [x]]]]:
> nops(lv);
```

$$11$$

The notation of the vector field means, for instance, an element

```
> lv[1];
```

$$[[y,\ [t]],\ [t,\ [y]]]$$

denotes the vector field generated by $y\,(\frac{\partial}{\partial t}\ ) - t\,(\frac{\partial}{\partial y}\ )$.

So we now have determined the infinitesimal generators for all 11 symmetries of the wave equation, which we are now going to check for their properties:

```
> sl:=symsplit(L,lv,ivar,dvar):
```

As follows, these symmetries split up into six variational symmetries and one that can be made variational through a linear combination of the given ones:

```
> sl[1];
```

$$[[[y,\ [t]],\ [t,\ [y]]],\ [[y,\ [x]],\ [-x,\ [y]]],\ [[1,\ [t]]],\ [[1,\ [y]]],\ [[-x,\ [t]],\ [-t,\ [x]]],\ [[1,\ [x]]],$$
$$[[-t,\ [t]],\ [-x,\ [x]],\ [-y,\ [y]],\ [\frac{1}{2}\,u,\ [u]]]]$$

```
> nops(sl[1]);
```

$$7$$

Further, there are three divergence symmetries:

```
> sl[2];
```

$$[[[-2\,t\,y,\,[t]],\,[-2\,y\,x,\,[x]],\,[-t^2+x^2-y^2,\,[y]],\,[u\,y,\,[u]]],$$
$$[[-x^2-y^2-t^2,\,[t]],\,[-2\,t\,x,\,[x]],\,[-2\,t\,y,\,[y]],\,[u\,t,\,[u]]],$$
$$[[-t\,x,\,[t]],\,[\frac{1}{2}\,y^2-\frac{1}{2}\,t^2-\frac{1}{2}\,x^2,\,[x]],\,[-y\,x,\,[y]],\,[\frac{1}{2}\,u\,x,\,[u]]]]$$

```
> nops(sl[2]);
```

$$3$$

...and one remaining symmetry:

```
> sl[3];
```

$$[[[u,\,[u]]]]$$

Corresponding to the variational and divergence symmetries, we get the following characteristics:

```
> char:=map(x->op(vec2char(x,ivar,dvar)),[op(sl[1]),op(sl[2])]):
```

After rearranging and changing the sign, if necessary...

```
> char := [u[t],u[x],u[y],x*u[y]-y*u[x],x*u[t]+t*u[x],y*u[t]+t*u[y],
> 1/2*u+t*u[t]+x*u[x]+y*u[y],
> 2*(1/2*u*x+x*t*u[t]-1/2*u[x]*y^2+1/2*u[x]*t^2+1/2*u[x]*x^2+y*x*u[y]),
> u*y+2*y*t*u[t]+2*y*x*u[x]+u[y]*y^2+u[y]*t^2-u[y]*x^2,
> u*t+u[t]*y^2+u[t]*t^2+u[t]*x^2+2*x*t*u[x]+2*y*t*u[y]];
```

$$\mathit{char} := [u_t,\,u_x,\,u_y,\,-u_x\,y+u_y\,x,\,u_t\,x+u_x\,t,\,u_t\,y+u_y\,t,\,\frac{1}{2}\,u+u_t\,t+u_x\,x+u_y\,y,$$
$$u\,x+2\,u_t\,t\,x-u_x\,y^2+u_x\,t^2+u_x\,x^2+2\,u_y\,y\,x,$$
$$u\,y+2\,u_t\,t\,y+2\,u_x\,y\,x+u_y\,t^2-u_y\,x^2+u_y\,y^2,$$
$$u\,t+u_t\,x^2+u_t\,y^2+u_t\,t^2+2\,u_x\,t\,x+2\,u_y\,t\,y]$$

```
> nops(char);
```

$$10$$

So now, we are able to compute the complete conservation laws of the wave equation explicitly:

```
> cons:=map(c->conservation(c,L,ivar,dvar),char):
```

In this case, we are mainly interested in conserved densities (the time-component of the conservation law). In the same step, the results are slightly simplified by shifting over derivatives between the jet variables.

```
> CL:=map(x->intpart(x[1],ivar,dvar),cons):
```

First, we are going to check the conservation laws for the criterion $\mathit{Div}\,P = Q\,\mathrm{E}(L)$ (Noether's theorem):

```
> map(i->simplify(Div(cons[i],ivar,dvar)-char[i]*EL),[$1..nops(char)]);
```

$$[0,\,0,\,0,\,0,\,0,\,0,\,0,\,0,\,0,\,0]$$

Comparing our results with those stated in literature [Olver, p.280], we have found:

From translation in direction t (time)

```
>  char[1];
```

$$u_t$$

we obtain the well-known conservation of energy.

```
>  E:=-CL[1];
```

$$E := \frac{1}{2}{u_x}^2 + \frac{1}{2}{u_y}^2 + \frac{1}{2}{u_t}^2$$

Translation in the spatial directions x, y

```
>  char[2..3];
```

$$[u_x,\ u_y]$$

yields conservation of momentum.

```
>  Px:=-CL[2]; Py:=-CL[3];
```

$$Px := u_t\,u_x$$
$$Py := u_t\,u_y$$

From these rotations

```
>  char[4..6];
```

$$[-u_x\,y + u_y\,x,\ u_t\,x + u_x\,t,\ u_t\,y + u_y\,t]$$

we get the following expressions:

```
>  A:=-CL[4];
```

$$A := -u_t\,u_x\,y + u_t\,u_y\,x$$

```
>  Mx:=-CL[5];
```

$$Mx := \frac{1}{2}x\,{u_x}^2 + \frac{1}{2}x\,{u_y}^2 + u_t\,u_x\,t + \frac{1}{2}x\,{u_t}^2$$

```
>  My:=-CL[6];
```

$$My := \frac{1}{2}y\,{u_x}^2 + \frac{1}{2}y\,{u_y}^2 + u_t\,u_y\,t + \frac{1}{2}y\,{u_t}^2$$

Hence we recognize the conservation of $A = x\,P_y - y\,P_x$, $M_x = x\,E + t\,P_x$ and $M_y = y\,E + t\,P_y$.

```
>  A-x*Py+y*Px;
```

$$0$$

```
>  expand(Mx-x*E-t*Px);
```

$$0$$

```
>  expand(My-y*E-t*Py);
```

$$0$$

The dilatation (made variational)

```
>  char[7];
```

$$\frac{1}{2}u + u_t\,t + u_x\,x + u_y\,y$$

leads to the following expression

```
>  DD:=-CL[7];
```

$$DD := \frac{1}{2}\,t\,{u_x}^2+\frac{1}{2}\,t\,{u_y}^2+\frac{1}{2}\,u_t\,u+u_t\,u_x\,x+u_t\,u_y\,y+\frac{1}{2}\,t\,{u_t}^2$$

and we can conclude the conservation law $\mathrm{D}=x\,P_x+y\,P_y+\frac{1\,u\,u_t}{2}+t\,E$.

```
>  expand(DD-x*Px-y*Py-1/2*u*u[t]-t*E);
```

$$0$$

Finally, from the inversions...

```
>  char[8..10];
```

$$[u\,x+2\,u_t\,t\,x-u_x\,y^2+u_x\,t^2+u_x\,x^2+2\,u_y\,y\,x,\ u\,y+2\,u_t\,t\,y+2\,u_x\,y\,x+u_y\,t^2-u_y\,x^2+u_y\,y^2,\\ u\,t+u_t\,x^2+u_t\,y^2+u_t\,t^2+2\,u_x\,t\,x+2\,u_y\,t\,y]$$

or, in a more familiar notation

```
>  map(i->collect(i,[u[x],u[y],u[t]]),char[8..10]);
```

$$[(-y^2+t^2+x^2)\,u_x+u\,x+2\,u_t\,t\,x+2\,u_y\,y\,x,\ 2\,u_x\,y\,x+(t^2-x^2+y^2)\,u_y+u\,y+2\,u_t\,t\,y,\\ 2\,u_x\,t\,x+2\,u_y\,t\,y+(x^2+y^2+t^2)\,u_t+u\,t]$$

arise these conservation laws:

```
>  Ix:=-CL[8];
```

$$Ix := {u_x}^2\,t\,x+{u_y}^2\,t\,x+u\,u_t\,x-u_t\,u_x\,y^2+u_t\,u_x\,t^2+u_t\,u_x\,x^2+2\,u_t\,u_y\,y\,x+{u_t}^2\,t\,x$$

```
>  Iy:=-CL[9];
```

$$Iy := {u_x}^2\,t\,y+{u_y}^2\,t\,y+u\,u_t\,y+2\,u_t\,u_x\,y\,x+u_t\,u_y\,t^2-u_t\,u_y\,x^2+u_t\,u_y\,y^2+{u_t}^2\,t\,y$$

```
>  It:=-CL[10];
```

$$It := \frac{1}{2}\,{u_t}^2\,x^2+\frac{1}{2}\,{u_t}^2\,y^2+\frac{1}{2}\,{u_t}^2\,t^2+2\,u_y\,u_t\,t\,y+u\,u_t\,t+\frac{1}{2}\,{u_x}^2\,t^2+\frac{1}{2}\,{u_x}^2\,x^2+\frac{1}{2}\,{u_x}^2\,y^2\\ +2\,u_x\,u_t\,t\,x+\frac{1}{2}\,{u_y}^2\,y^2+\frac{1}{2}\,{u_y}^2\,t^2+\frac{1}{2}\,{u_y}^2\,x^2-\frac{1}{2}\,u^2$$

and we recognize again the expressions $Ix=x\,\mathrm{D}+y\,A+\frac{1\,x\,u\,u_t}{2}+t\,Mx$, $Iy=y\,\mathrm{D}-x\,A+\frac{1\,y\,u\,u_t}{2}+$ and $It=(x^2+y^2)\,E-\frac{1\,u^2}{2}+2\,t\,DD-t^2\,E$ as conserved densities.

```
>  expand(Ix-x*DD-y*A-1/2*x*u*u[t]-t*Mx);
```

$$0$$

```
>  expand(Iy-y*DD+x*A-1/2*y*u*u[t]-t*My);
```

$$0$$

```
>  expand(It-(x^2+y^2)*E+1/2*u^2-2*t*DD+t^2*E);
```

$$0$$

Thus, we now have reproduced and reviewed the data from [Olver p.280].

Higher Order Symmetries

To consider higher order symmetries now, we first have to transform the characteristics of the first order symmetries into differential operators (recursion operators).

```
>  RO:=map(x->frechetc([x],ivar,dvar)[1,1],char):
```

```
> REC:=map(x->list2op(x,ivar,dvar),RO):
> Dt:=REC[1]: Dx:=REC[2]: Dy:=REC[3]: Rxy:=REC[4]; Rxt:=REC[5]:
> Ryt:=REC[6]: Dd:=REC[7]: Jx:=REC[8]: Jy:=REC[9]: Jt:=REC[10]:
```

$Rxy := a \to -y\,\mathrm{totalder}(a,\,[x],\,[t,\,x,\,y],\,[u]) + x\,\mathrm{totalder}(a,\,[y],\,[t,\,x,\,y],\,[u])$

A higher order symmetry results from successive products of such operators, e.g.

```
> expand((Rxy @ Rxt) (u));
```

$$-u_{t,\,x}\,y\,x - u_t\,y - t\,y\,u_{x,\,x} + u_{t,\,y}\,x^2 + u_{x,\,y}\,t\,x$$

Second order symmetries do not produce any new conservation laws, but those of third order lead to 84 new ones. Some of them are treated in literature [Olver p.340].

So we look at the following selection of third order symmetries (according to Olver):

```
> SO:=[Dx@@3,Dx@Dx@Dt,Dt@@3,Dx@Rxy@Dx,Dx@Rxy@Dy-1/2*Dx@Dx-1/2*Dy@Dy,Dx@
> Rxt@Dx,Dx@Dd@Dx]:
```

with corresponding characteristics:

```
> SC:=map(a->a(u),SO);
```

$$SC := [u_{x,\,x,\,x},\ u_{t,\,x,\,x},\ u_{t,\,t,\,t},\ -y\,u_{x,\,x,\,x} + u_{x,\,y} + x\,u_{x,\,x,\,y},\ -y\,u_{x,\,x,\,y} + \frac{1}{2}\,u_{y,\,y} + x\,u_{x,\,y,\,y} - \frac{1}{2}\,u_{x,\,x},$$
$$u_{t,\,x,\,x}\,x + u_{t,\,x} + u_{x,\,x,\,x}\,t,\ t\,u_{t,\,x,\,x} + \frac{3}{2}\,u_{x,\,x} + x\,u_{x,\,x,\,x} + y\,u_{x,\,x,\,y}]$$

Thus, the resulting conservation laws are:

```
> COS:=map(c->conservation(c,L,ivar,dvar),SC):
```

These results are checked using the criterion $Div\,P = Q\,\mathrm{E}(L)$ (Noether's theorem):

```
> map(i->simplify(Div(COS[i],ivar,dvar)-SC[i]*EL),[$1..nops(SC)]);
```

$$[0,\,0,\,0,\,0,\,0,\,0,\,0]$$

Now, we consider the conserved densities and simplify the expressions by shifting over derivatives of jet variables:

```
> COL:=map(x->intpart(x[1],ivar,dvar),COS):
> nops(COL);
```

$$7$$

So we are able to compare our results with the conserved densities given in the table [Olver, p.340]:

```
> COL[1];
```

$$u_{t,\,x}\,u_{x,\,x}$$

corresponds exactly to the given term...

```
> COL[2];
```

$$\frac{1}{2}\,{u_{x,x}}^2+\frac{1}{3}\,u_{x,x}\,u_{y,y}+\frac{1}{2}\,{u_{t,x}}^2+\frac{1}{6}\,{u_{x,y}}^2$$

```
> divnorm(expand(COL[2]-1/2*(u[t,x]^2+u[x,x]^2+u[x,y]^2)),ivar,dvar);
```

$$0$$

corresponds to the literature expression up to trivial conservation laws....

```
> COL[3];
```

$$u_{x,x}\,u_{t,t}+u_{t,t}\,u_{y,y}+\frac{1}{2}\,{u_{t,x}}^2+\frac{1}{2}\,{u_{t,y}}^2-\frac{1}{2}\,{u_{t,t}}^2$$

```
> expand(COL[3]-1/2*(u[t,t]^2+u[t,x]^2+u[t,y]^2));
```

$$u_{x,x}\,u_{t,t}+u_{t,t}\,u_{y,y}-{u_{t,t}}^2$$

```
> simplify(subs(isolate(DE,u[t,t]),%));
```

$$0$$

is equivalent to the given term for all solutions of the wave equation.

```
> COL[4];
```

$$-u_{t,x}\,u_{x,x}\,y+\frac{1}{2}\,u_{t,x}\,u_{x,y}\,x-\frac{1}{2}\,u_t\,u_{x,y}+\frac{1}{2}\,u_{x,x}\,u_{t,y}\,x$$

```
> divnorm(expand(COL[4]-u[t,x]*(x*u[x,y]-y*u[x,x])),ivar,dvar);
```

$$0$$

holds up to trivial conservation laws.

```
> COL[5];
```

$$-\frac{1}{2}\,u_{t,x}\,u_{x,y}\,y-\frac{1}{4}\,u_x\,u_{t,x}-\frac{1}{4}\,u_{t,y}\,u_y+\frac{1}{2}\,u_{t,x}\,x\,u_{y,y}-\frac{1}{2}\,u_{x,x}\,u_{t,y}\,y-\frac{1}{4}\,u_t\,u_{x,x}-\frac{1}{4}\,u_t\,u_{y,y}$$
$$+\frac{1}{2}\,u_{x,y}\,u_{t,y}\,x$$

```
> divnorm(expand(COL[5]+(u[x,x]*(y*u[t,y]+1/2*u[t])-u[y,y]*(x*u[t,x]+1/
> 2*u[t]))),ivar,dvar);
```

$$0$$

corresponds to the literature value up to normalization.

```
> COL[6];
```

$$\frac{1}{12}\,u_x\,u_{y,y}+\frac{1}{2}\,x\,{u_{x,x}}^2-\frac{1}{12}\,u_y\,u_{x,y}+\frac{1}{3}\,u_{x,x}\,x\,u_{y,y}+\frac{1}{6}\,{u_{x,y}}^2\,x+u_{t,x}\,u_{x,x}\,t+\frac{1}{2}\,{u_{t,x}}^2\,x$$

```
> Th:=1/2*(u[t,x]^2+u[x,x]^2+u[x,y]^2);
```

$$Th:=\frac{1}{2}\,{u_{t,x}}^2+\frac{1}{2}\,{u_{x,x}}^2+\frac{1}{2}\,{u_{x,y}}^2$$

```
> divnorm(expand(COL[6]-(x*Th+t*u[x,x]*u[t,x])),ivar,dvar);
```

$$0$$

..confirms the expression from the table

```
> COL[7];
```

$$\frac{1}{4} u_t\, u_{x,x} + \frac{1}{4} u_x\, u_{t,x} + \frac{1}{2} u_{t,x}\, u_{x,y}\, y + \frac{1}{2} u_{x,x}\, u_{t,y}\, y + \frac{1}{2} {u_{x,x}}^2\, t + \frac{1}{3} u_{x,x}\, t\, u_{y,y} + \frac{1}{6} {u_{x,y}}^2\, t$$
$$+ u_{t,x}\, x\, u_{x,x} + \frac{1}{2} {u_{t,x}}^2\, t$$

```
>   Ts:=x*u[x,x]*u[t,x]+y*u[x,x]*u[t,y]+1/2*u[x,x]*u[t];
```

$$Ts := u_{t,x}\, x\, u_{x,x} + u_{x,x}\, u_{t,y}\, y + \frac{1}{2} u_t\, u_{x,x}$$

```
>   divnorm(expand(COL[7]-(Ts+t*Th)),ivar,dvar);
```

$$0$$

and finally also this expression corresponds to the one given in the table [Olv] p.340.

So the conservation laws which have been computed here are able to check the examples given in the table and even correct some mistyping in some cases.

For a more detailed version of this example, see MAPLE worksheet [ex2] on the CD ROM.

Appendix C

List of JETS functions

The following list gives the names and a short description of the functions which were implemented in the MAPLE package JETS [BaHa] as a part of this work. A detailed description for each procedure, its input parameters and output, including examples of its application, can be found on the enclosed CD ROM. There, these documentations are available both as (executable) MAPLE worksheets and as MAPLE help pages as a part of the JETS package.

command	**description**
`conservation`	Compute the conservation laws according to Noether's theorem
`classcons`	Determine the classical conservation laws (variational case) by Noether's theorem
`symsplit`	Split a vector space of symmetries into subspaces of variational, divergence and other symmetries
`symtestgen`	Filter independent generalized symmetries and variational symmetries from a list of recursion operators
`Euler`	(Higher) Euler operator applied to an expression
`Ordeuler`	(Ordinary) Euler operator applied to an expression
`homotopy`	Total homotopy operator of Lagrangians and currents
`Helmholtz`	Helmholtz operator applied to a source form
`Adjoint`	Compute the adjoint of a differential operator
`gcollect`	Collect the terms of a vector field or differential operator
`intpart`	Normalize jet expressions with respect to their order
`divnorm`	Normalize a jet expression up to divergences
`depcheck`	Check a list of expressions for linear dependencies
`getbas`	Get a basis and linear relations from a list of expressions
`Div`	Divergence of a current
`Curl`	Generalized curl of a $(p-2)$-form

`current`	Divergence part (current) of a prolonged evolutionary vector field
`interprod`	Total interior product for Lagrangians (p-forms) and currents ($(p-1)$-forms)
`opdot`	Dot product of a differential operator with an expression
`eulprol`	Prolongation of an evolutionary vector field using the Euler operator
`jsubs`	Substitute jet variables and their derivatives into a (jet) expression
`Qsubs`	Substitute variables and their (jet) derivatives into an expression
`Qcheck`	Check the properties of an integration path transformation
`Qinverse`	Compute the inverse of an integration path transformation
`coeffmatrix`	Transform a system of (linear) expressions into a coefficient matrix
`linmatrix`	Transform a system of (linear) expressions into a coefficient matrix for a given basis
`grank`	Compute the rank of a Gauss-reduced matrix
`zerocol`	Find the positions of zero columns in a matrix
`zerorow`	Find the positions of zero rows in a matrix
`lorder`	Order function for differential operators and vector fields (ascending and descending order)
`oporder`	Order function for differential operators (ascending order)
`roporder`	Order function for differential operators (descending order)
`vforder`	Order function for vector fields (ascending order)
`rvforder`	Order function for vector fields (descending order)
`iorder`	Order function for lists
`list2op`	Transform a differential operator from list notation to operator notation
`mat2op`	Transform a list or matrix of differential operators from list notation to operator notation
`list2vf`	Transform a vector field from list notation to operator notation
`multinom`	Compute the multinomial coefficient of multiindices
`ind2mult`	Transform a list of variables into multiindex notation
`remain`	Determine the difference of two sets or lists
`sublist`	Check if a list is contained in another one
`ilint`	Integrate a Lagrangian with λ-substituted independent variables
`dlint`	Integrate a Lagrangian with λ-substituted dependent variables
`sdlint`	Integrate a Lagrangian with λ-substituted dependent variables, singularity-respecting case
`dlfac`	Partial derivative of integration path
`const`	Evaluate a function at 0 in all variables

Appendix D

CD ROM

The enclosed CD ROM contains all MAPLE worksheets and packages used in this work.
The software documentation (help files) are stored in the directory `helpfiles`, the example worksheets can be found in the directory `worksheets`. The maple packages JETS and DESOLV are in the directory `lib`, with subdirectories for MAPLE V R5 (`maple5`) and MAPLE 6 (`maple6`). When running any examples in MAPLE, the correct path of the appropriate packages has to be specified with the command `libname`.
The enclosed worksheets are suitable for both MAPLE V RELEASE 5.X and MAPLE 6 and all operating systems.

Bibliography

[BaHa] Mohamed Barakat, Gehrt Hartjen, *JETS for Maple*, Lehrstuhl B für Mathematik der RWTH Aachen, Aachen 1999..2001

[VuCa] Khai T. Vu, John Carminati, *DESOLV for Maple V Release 5*, School of Computing and Mathematics, Deakin University, Geelong, Victoria, Australia, released June 2000

[Bar] Mohamed Barakat, *Functional Spaces. A Direct Approach*, Ph.D thesis, Aachen 2001

[Olv] Peter J. Olver, *Applications of Lie groups to differential equations, 2nd edition*, Springer, New York, 1993

[Ibr] Nail. H. Ibragimov (ed.), *CRC handbook of Lie group analysis of differential equations, vol.2: Applications in engineering and physical sciences*, CRC Press, Boca Raton, 1995

[Nik] A. G. Nikitin, *Generalized Killing tensors of arbitrary rank and order [in Russian]*, Ukrainskii Mathematicheskii Zhurnal, vol.43, No.6, 786..195 (1991)

[Nik.e] A. G. Nikitin, *Generalized Killing tensors of arbitrary rank and order [English translation]*, Ukr. Math. J. 43, No.6, 734..743 (1991)

[BSSS] V. G. Bagrov, B. F. Samsonov, A. V. Shapovalov, I. V. Shirokov, *Identities on solutions of the wave equation in the enveloping algebra of the conformal group [in Russian]*, Teoreticheskaya i Matematicheskaya Fizika, vol.83, No.1, 14..22 (1990)

[BSSS.e] V. G. Bagrov, B. F. Samsonov, A. V. Shapovalov, I. V. Shirokov, *Identities on solutions of the wave equation in the enveloping algebra of the conformal group [English translation]*, Theor. Math. Phys. vol.83, No.1, 347..353 (1990)

[SSS] A. V. Shapovalov, I. V. Shirokov, *Symmetry algebras of linear differential equations [English translation]*, Theor. Math. Phys. vol.92, 697..705 (1992)

[ex0] Notational Conventions in the JETS package (Maple worksheet, see Appendix)

[ex1] Conservation Laws in Elastostatics (Maple worksheet, see Appendix)

[ex2] Conservation Laws of the Three-dimensional Wave Equation (Maple worksheet, see Appendix)

[ex3] Symmetries of the Three-dimensional wave Equation (Maple worksheet, see Appendix)

[ex4] The Wave Equation in Four Dimensions (Maple worksheet, see Appendix)

Statement

Hiermit versichere ich, daß ich diese Diplomarbeit selbständig verfasst und keine außer den angegebenen Hilfsmitteln verwendet habe.

Aachen, den 7. Dezember 2001,

Gehrt Hartjen