Functional Differential Geometry

Functional Differential Geometry

Gerald Jay Sussman and Jack Wisdom
with Will Farr

This book was set by the authors using the LaTeX typesetting system and was printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Sussman, Gerald Jay.
Functional Differential Geometry / Gerald Jay Sussman and Jack Wisdom with Will Farr.
p. cm.
Includes bibliographical references and index.
ISBN x-xxx-xxxxxx-x (hc.: alk. paper)

“The author has spared himself no pains in his endeavour to present the main ideas in the simplest and most intelligible form, and on the whole, in the sequence and connection in which they actually originated. In the interest of clearness, it appeared to me inevitable that I should repeat myself frequently, without paying the slightest attention to the elegance of the presentation. I adhered scrupulously to the precept of that brilliant theoretical physicist L. Boltzmann, according to whom matters of elegance ought be left to the tailor and to the cobbler.”

Albert Einstein, in *Relativity, the Special and General Theory*, (1961), p. v

Contents

Preface

Learning physics is hard. Part of the problem is that physics is naturally expressed in mathematical language. When we teach we use the language of mathematics in the same way that we use our natural language. We depend upon a vast amount of shared knowledge and culture, and we only sketch an idea using mathematical idioms. We are insufficiently precise to convey an idea to a person who does not share our culture. Our problem is that since we share the culture we find it difficult to notice that what we say is too imprecise to be clearly understood by a student new to the subject. A student must simultaneously learn the mathematical language and the content that is expressed in that language. This is like trying to read *Les Misérables* while struggling with French grammar.

This book is an effort to ameliorate this problem for learning the differential geometry needed as a foundation for a deep understanding of general relativity or quantum field theory. Our approach differs from the traditional one in several ways. Our coverage is unusual. We do not prove the general Stokes's theorem—this is well covered in many other books—instead, we show how it works in two dimensions. Because our target is relativity, we put lots of emphasis on the development of the covariant derivative, and we erect a common context for understanding both the Lie derivative and the covariant derivative. Most treatments of differential geometry aimed at relativity assume that there is a metric (or pseudometric). By contrast, we develop as much material as possible independent of the assumption of a metric. This allows us to see what results depend on the metric when we introduce it. We also try to avoid the use of traditional index notation for tensors. Although one can become very adept at "index gymnastics," that leads to much mindless (though useful) manipulation without much thought to meaning. Instead, we use a semantically richer language of vector fields and differential forms.

But the single biggest difference between our treatment and others is that we integrate computer programming into our explanations. By programming a computer to interpret our formulas we soon learn whether or not a formula is correct. If a formula is not clear, it will not be interpretable. If it is wrong, we will get a wrong answer. In either case we are led to improve our program and as a result improve our understanding. We have been teaching advanced classical mechanics at MIT for many years using this strategy. We use precise functional notation and we have students program in a functional language. The students enjoy this approach and we have learned a lot ourselves. It is the experience of writing software for expressing the mathematical content and the insights that we gain from doing it that we feel is revolutionary. We want others to have a similar experience.

Acknowledgments

We thank the people who helped us develop this material, and especially the students who have over the years worked through the material with us. In particular, Mark Tobenkin, William Throwe, Leo Stein, Peter Iannucci, and Micah Brodsky have suffered through bad explanations and have contributed better ones.

Edmund Bertschinger, Norman Margolus, Tom Knight, Rebecca Frankel, Alexey Radul, Edwin Taylor, Joel Moses, Kenneth Yip, and Hal Abelson helped us with many thoughtful discussions and advice about physics and its relation to mathematics.

We also thank Chris Hanson, Taylor Campbell, and the community of Scheme programmers for providing support and advice for the elegant language that we use. In particular, Gerald Jay Sussman wants to thank Guy Lewis Steele and Alexey Radul for many fun days of programming together—we learned much from each other's style.

Matthew Halfant started us on the development of the Scmutils system. He encouraged us to get into scientific computation, using Scheme and functional style as an active way to explain the ideas, without the distractions of imperative languages such as C. In the 1980s he wrote some of the early Scheme procedures for numerical computation that we still use.

Dan Zuras helped us with the invention of the unique organization of the Scmutils system. It is because of his insight that the system is organized around a generic extension of the chain rule for taking derivatives. He also helped in the heavy lifting that was required to make a really good polynomial GCD algorithm, based on ideas we learned from Richard Zippel.

A special contribution that cannot be sufficiently acknowledged is from Seymour Papert and Marvin Minsky, who taught us that

the practice of programming is a powerful way to develop a deeper understanding of any subject. Indeed, by the act of debugging we learn about our misconceptions, and by reflecting on our bugs and their resolutions we learn ways to learn more effectively.

We acknowledge the generous support of the Computer Science and Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. The laboratory provides a stimulating environment for efforts to formalize knowledge with computational methods. We also acknowledge the Panasonic Corporation (formerly the Matsushita Electric Industrial Corporation) for support of Gerald Jay Sussman through an endowed chair.

Jack Wisdom thanks his wife, Cecile, for her love and support. Julie Sussman, PPA, provided careful reading and serious criticism that forced us to reorganize and rewrite major parts of the text. She also developed and maintained Gerald Jay Sussman over these many years.

Gerald Jay Sussman & Jack Wisdom
Cambridge, Massachusetts, USA
August 2012

Prologue

Programming and Understanding

One way to become aware of the precision required to unambiguously communicate a mathematical idea is to program it for a computer. Rather than using canned programs purely as an aid to visualization or numerical computation, we use computer programming in a functional style to encourage clear thinking. Programming forces us to be precise and unambiguous, without forcing us to be excessively rigorous. The computer does not tolerate vague descriptions or incomplete constructions. Thus the act of programming makes us keenly aware of our errors of reasoning or unsupported conclusions.[1]

Although this book is about differential geometry, we can show how thinking about programming can help in understanding in a more elementary context. The traditional use of Leibnitz's notation and Newton's notation is convenient in simple situations, but in more complicated situations it can be a serious handicap to clear reasoning.

A mechanical system is described by a Lagrangian function of the system state (time, coordinates, and velocities). A motion of the system is described by a path that gives the coordinates for each moment of time. A path is allowed if and only if it satisfies the Lagrange equations. Traditionally, the Lagrange equations are written

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = 0.$$

What could this expression possibly mean?

Let's try to write a program that implements Lagrange equations. What are Lagrange equations for? Our program must take a proposed path and give a result that allows us to decide if the path is allowed. This is already a problem; the equation shown above does not have a slot for a path to be tested.

[1]The idea of using computer programming to develop skills of clear thinking was originally advocated by Seymour Papert. An extensive discussion of this idea, applied to the education of young children, can be found in Papert [12].

So we have to figure out how to insert the path to be tested. The partial derivatives do not depend on the path; they are derivatives of the Lagrangian function and thus they are functions with the same arguments as the Lagrangian. But the time derivative d/dt makes sense only for a function of time. Thus we must be intending to substitute the path (a function of time) and its derivative (also a function of time) into the coordinate and velocity arguments of the partial derivative functions.

So probably we meant something like the following (assume that w is a path through the coordinate configuration space, and so $w(t)$ specifies the configuration coordinates at time t):

$$\frac{d}{dt}\left(\left. \frac{\partial L(t,q,\dot{q})}{\partial \dot{q}} \right|_{\substack{q = w(t) \\ \dot{q} = \frac{dw(t)}{dt}}} \right) - \left. \frac{\partial L(t,q,\dot{q})}{\partial q} \right|_{\substack{q = w(t) \\ \dot{q} = \frac{dw(t)}{dt}}} = 0.$$

In this equation we see that the partial derivatives of the Lagrangian function are taken, then the path and its derivative are substituted for the position and velocity arguments of the Lagrangian, resulting in an expression in terms of the time.

This equation is complete. It has meaning independent of the context and there is nothing left to the imagination. The earlier equations require the reader to fill in lots of detail that is implicit in the context. They do not have a clear meaning independent of the context.

By thinking computationally we have reformulated the Lagrange equations into a form that is explicit enough to specify a computation. We could convert it into a program for any symbolic manipulation program because it tells us *how* to manipulate expressions to compute the residuals of Lagrange's equations for a purported solution path.[2]

[2]The *residuals* of equations are the expressions whose value must be zero if the equations are satisfied. For example, if we know that for an unknown x, $x^3 - x = 0$ then the residual is $x^3 - x$. We can try $x = -1$ and find a residual of 0, indicating that our purported solution satisfies the equation. A residual may provide information. For example, if we have the differential equation $df(x)/dx - af(x) = 0$ and we plug in a test solution $f(x) = Ae^{bx}$ we obtain the residual $(b-a)Ae^{bx}$, which can be zero only if $b = a$.

Functional Abstraction

But this corrected use of Leibnitz notation is ugly. We had to introduce extraneous symbols (q and $\dot{q}$) in order to indicate the argument position specifying the partial derivative. Nothing would change here if we replaced q and $\dot{q}$ by a and b.[3] We can simplify the notation by admitting that the partial derivatives of the Lagrangian are themselves new functions, and by specifying the particular partial derivative by the position of the argument that is varied

$$\frac{d}{dt}((\partial_2 L)(t, w(t), \frac{d}{dt}w(t))) - (\partial_1 L)(t, w(t), \frac{d}{dt}w(t)) = 0,$$

where $\partial_i L$ is the function which is the partial derivative of the function L with respect to the ith argument.[4]

Two different notions of derivative appear in this expression. The functions $\partial_2 L$ and $\partial_1 L$, constructed from the Lagrangian L, have the same arguments as L. The derivative d/dt is an expression derivative. It applies to an expression that involves the variable t and it gives the rate of change of the value of the expression as the value of the variable t is varied.

These are both useful interpretations of the idea of a derivative. But functions give us more power. There are many equivalent ways to write expressions that compute the same value. For example $1/(1/r_1 + 1/r_2) = (r_1 r_2)/(r_1 + r_2)$. These expressions compute the same function of the two variables r_1 and r_2. The first expression fails if $r_1 = 0$ but the second one gives the right value of the function. If we abstract the function, say as $\Pi(r_1, r_2)$, we can ignore the details of how it is computed. The ideas become clearer because they do not depend on the detailed shape of the expressions.

[3]That the symbols q and $\dot{q}$ can be replaced by other arbitrarily chosen non-conflicting symbols without changing the meaning of the expression tells us that the partial derivative symbol is a logical quantifier, like forall and exists ($\forall$ and $\exists$).

[4]The argument positions of the Lagrangian are indicated by indices starting with zero for the time argument.

So let's get rid of the expression derivative d/dt and replace it with an appropriate functional derivative. If f is a function then we will write Df as the new function that is the derivative of f:[5]

$$(Df)(t) = \frac{d}{dx} f(x) \bigg|_{x=t}$$

To do this for the Lagrange equation we need to construct a function to take the derivative of.

Given a configuration-space path w, there is a standard way to make the state-space path. We can abstract this method as a mathematical function Γ:

$$\Gamma[w](t) = (t, w(t), \frac{d}{dt} w(t)).$$

Using Γ we can write:

$$\frac{d}{dt}((\partial_2 L)(\Gamma[w](t))) - (\partial_1 L)(\Gamma[w](t)) = 0.$$

If we now define composition of functions $(f \circ g)(x) = f(g(x))$, we can express the Lagrange equations entirely in terms of functions:

$$D((\partial_2 L) \circ (\Gamma[w])) - (\partial_1 L) \circ (\Gamma[w]) = 0.$$

The functions $\partial_1 L$ and $\partial_2 L$ are partial derivatives of the function L. Composition with $\Gamma[w]$ evaluates these partials with coordinates and velocites appropriate for the path w, making functions of time. Applying D takes the time derivative. The Lagrange equation states that the difference of the resulting functions of time must be zero. This statement of the Lagrange equation is complete, unambiguous, and functional. It is not encumbered with the particular choices made in expressing the Lagrangian. For example, it doesn't matter if the time is named t or τ, and it has an explicit place for the path to be tested.

This expression is equivalent to a computer program:[6]

[5]An explanation of functional derivatives is in Appendix B, page 204.

[6]The programs in this book are written in Scheme, a dialect of Lisp. The details of the language are not germane to the points being made. What is important is that it is mechanically interpretable, and thus unambiguous. In this book we require that the mathematical expressions be explicit enough

```
(define ((Lagrange-equations Lagrangian) w)
  (- (D (compose ((partial 2) Lagrangian) (Gamma w)))
     (compose ((partial 1) Lagrangian) (Gamma w))))
```

In the Lagrange equations procedure the parameter `Lagrangian` is a procedure that implements the Lagrangian. The derivatives of the Lagrangian, for example `((partial 2) Lagrangian)`, are also procedures. The state-space path procedure `(Gamma w)` is constructed from the configuration-space path procedure `w` by the procedure `Gamma`:

```
(define ((Gamma w) t)
  (up t (w t) ((D w) t)))
```

where `up` is a constructor for a data structure that represents a state of the dynamical system (time, coordinates, velocities).

The result of applying the `Lagrange-equations` procedure to a procedure `Lagrangian` that implements a Lagrangian function is a procedure that takes a configuration-space path procedure `w` and returns a procedure that gives the residual of the Lagrange equations for that path at a time.

For example, consider the harmonic oscillator, with Lagrangian.

$$L(t, q, v) = \tfrac{1}{2}mv^2 - \tfrac{1}{2}kq^2,$$

for mass m and spring constant k. This Lagrangian is implemented by

```
(define ((L-harmonic m k) local)
  (let ((q (coordinate local))
        (v (velocity local)))
    (- (* 1/2 m (square v))
       (* 1/2 k (square q)))))
```

We know that the motion of a harmonic oscillator is a sinusoid with a given amplitude a, frequency ω, and phase φ:

$$x(t) = a\cos(\omega t + \varphi).$$

that they can be expressed as computer programs. Scheme is chosen because it is easy to write programs that manipulate representations of mathematical functions. An informal description of Scheme can be found in Appendix A. The use of Scheme to represent mathematical objects can be found in Appendix B. A formal description of Scheme can be obtained in [9]. You can get the software from [20].

Suppose we have forgotten how the constants in the solution relate to the physical parameters of the oscillator. Let's plug in the proposed solution and look at the residual:

```
(define (proposed-solution t)
  (* 'a (cos (+ (* 'omega t) 'phi))))

(show-expression
 (((Lagrange-equations (L-harmonic 'm 'k))
   proposed-solution)
  't))
```

$$\boxed{\cos\left(\omega t+\varphi\right)a\left(k-m\omega^2\right)}$$

The residual here shows that for nonzero amplitude, the only solutions allowed are ones where $(k - m\omega^2) = 0$ or $\omega = \sqrt{k/m}$.

But, suppose we had no idea what the solution looks like. We could propose a literal function for the path:

```
(show-expression
 (((Lagrange-equations (L-harmonic 'm 'k))
   (literal-function 'x))
  't))
```

$$\boxed{kx\left(t\right)+mD^2x\left(t\right)}$$

If this residual is zero we have the Lagrange equation for the harmonic oscillator.

Note that we can flexibly manipulate representations of mathematical functions. (See Appendices A and B.)

We started out thinking that the original statement of Lagrange's equations accurately captured the idea. But we really don't know until we try to teach it to a naive student. If the student is sufficiently ignorant, but is willing to ask questions, we are led to clarify the equations in the way that we did. There is no dumber but more insistent student than a computer. A computer will absolutely refuse to accept a partial statement, with missing parameters or a type error. In fact, the original statement of Lagrange's equations contained an obvious type error: the Lagrangian is a function of multiple variables, but the d/dt is applicable only to functions of one variable.

Functional Differential Geometry

1 Introduction

Philosophy is written in that great book which ever lies before our eyes—I mean the Universe—but we cannot understand it if we do not learn the language and grasp the symbols in which it is written. This book is written in the mathematical language, and the symbols are triangles, circles, and other geometrical figures without whose help it is impossible to comprehend a single word of it, without which one wanders in vain through a dark labyrinth.

Galileo Galilei [7]

Differential geometry is a mathematical language that can be used to express physical concepts. In this introduction we show a typical use of this language. Do not panic! At this point we do not expect you to understand the details of what we are showing. All will be explained as needed in the text. The purpose is to get the flavor of this material.

At the North Pole inscribe a line in the ice perpendicular to the Greenwich Meridian. Hold a stick parallel to that line and walk down the Greenwich Meridian keeping the stick parallel to itself as you walk. (The phrase "parallel to itself" is a way of saying that as you walk you keep its orientation unchanged. The stick will be aligned East-West, perpendicular to your direction of travel.) When you get to the Equator the stick will be parallel to the Equator. Turn East, and walk along the Equator, keeping the stick parallel to the Equator. Continue walking until you get to the 90°E meridian. When you reach the 90°E meridian turn North and walk back to the North Pole keeping the stick parallel to itself. Note that the stick is perpendicular to your direction of travel. When you get to the Pole note that the stick is perpendicular to the line you inscribed in the ice. But you started with that stick parallel to that line and you kept the stick pointing in the same direction on the Earth throughout your walk—how did it change orientation?

The answer is that you walked a closed loop on a curved surface. As seen in three dimensions the stick was actually turning as you walked along the Equator, because you always kept the stick parallel to the curving surface of the Earth. But as a denizen of a 2-dimensional surface, it seemed to you that you kept the stick parallel to itself as you walked, even when making a turn. Even if you had no idea that the surface of the Earth was embedded in a 3-dimensional space you could use this experiment to conclude that the Earth was not flat. This is a small example of intrinsic geometry. It shows that the idea of parallel transport is not simple. For a general surface it is necessary to explicitly define what we mean by parallel.

If you walked a smaller loop, the angle between the starting orientation and the ending orientation of the stick would be smaller. For small loops it would be proportional to the area of the loop you walked. This constant of proportionality is a measure of the curvature. The result does not depend on how fast you walked, so this is not a dynamical phenomenon.

Denizens of the surface may play ball games. The balls are constrained to the surface; otherwise they are free particles. The paths of the balls are governed by dynamical laws. This motion is a solution of the Euler-Lagrange equations[1] for the free-particle Lagrangian with coordinates that incorporate the constraint of living in the surface. There are coefficients of terms in the Euler-Lagrange equations that arise naturally in the description of the behavior of the stick when walking loops on the surface, connecting the static shape of the surface with the dynamical behavior of the balls. It turns out that the dynamical evolution of the balls may be viewed as parallel transport of the ball's velocity vector in the direction of the velocity vector. This motion by parallel transport of the velocity is called *geodesic motion*.

So there are deep connections between the dynamics of particles and the geometry of the space that the particles move in. If we understand this connection we can learn about dynamics by studying geometry and we can learn about geometry by studying dynamics. We enter dynamics with a Lagrangian and the associated Lagrange equations. Although this formulation exposes many important features of the system, such as how symmetries relate to

[1]It is customary to shorten ”Euler-Lagrange equations” to ”Lagrange equations.” We hope Leonhard Euler is not disturbed.

conserved quantities, the geometry is not apparent. But when we express the Lagrangian and the Lagrange equations in differential geometry language, geometric properties become apparent. In the case of systems with no potential energy the Euler-Lagrange equations are equivalent to the geodesic equations on the configuration manifold. In fact, the coefficients of terms in the Lagrange equations are Christoffel coefficients, which define parallel transport on the manifold. Let's look into this a bit.

Lagrange Equations

We write the Lagrange equations in functional notation[2] as follows:

$$D(\partial_2 L \circ \Gamma[q]) - \partial_1 L \circ \Gamma[q] = 0.$$

In SICM [18], Section 1.6.3, we showed that a Lagrangian describing the free motion of a particle subject to a coordinate-dependent constraint can be obtained by composing a free-particle Lagrangian with a function that describes how dynamical states transform given the coordinate transformation that describes the constraints.

A Lagrangian for a free particle of mass m and velocity v is just its kinetic energy, $mv^2/2$. The procedure `Lfree` implements the free Lagrangian:[3]

```
(define ((Lfree mass) state)
  (* 1/2 mass (square (velocity state))))
```

For us the dynamical state of a system of particles is a tuple of time, coordinates, and velocities. The free-particle Lagrangian depends only on the velocity part of the state.

For motion of a point constrained to move on the surface of a sphere the configuration space has two dimensions. We can describe the position of the point with the generalized coordinates colatitude and longitude. If the sphere is embedded in 3-dimensional space the position of the point in that space can be

[2]A short introduction to our functional notation, and why we have chosen it, is given in the prologue: Programming and Understanding. More details can be found in Appendix B.

[3]An informal description of the Scheme programming language can be found in Appendix A.

given by a coordinate transformation from colatitude and longitude to three rectangular coordinates.

For a sphere of radius R the procedure `sphere->R3` implements the transformation of coordinates from colatitude θ and longitude ϕ on the surface of the sphere to rectangular coordinates in the embedding space. (The $\hat{z}$ axis goes through the North Pole, and the Equator is in the plane $z = 0$.)

```
(define ((sphere->R3 R) state)
  (let ((q (coordinate state)))
    (let ((theta (ref q 0)) (phi (ref q 1)))
      (up (* R (sin theta) (cos phi))     ; x
          (* R (sin theta) (sin phi))     ; y
          (* R (cos theta))))))           ; z
```

The coordinate transformation maps the generalized coordinates on the sphere to the 3-dimensional rectangular coordinates. Given this coordinate transformation we construct a corresponding transformation of velocities; these make up the state transformation. The procedure `F->C` implements the derivation of a transformation of states from a coordinate transformation:

```
(define ((F->C F) state)
  (up (time state)
      (F state)
      (+ (((partial 0) F) state)
         (* (((partial 1) F) state)
            (velocity state)))))
```

A Lagrangian governing free motion on a sphere of radius R is then the composition of the free Lagrangian with the transformation of states.

```
(define (Lsphere m R)
  (compose (Lfree m) (F->C (sphere->R3 R))))
```

So, the value of the Lagrangian, at an arbitrary dynamical state, is:

```
((Lsphere 'm 'R)
 (up 't (up 'theta 'phi) (up 'thetadot 'phidot)))
(+ (* 1/2 m (expt R 2) (expt thetadot 2))
   (* 1/2 m (expt R 2) (expt (sin theta) 2) (expt phidot 2)))
```

or, in infix notation:

$$\frac{1}{2}mR^2\dot{\theta}^2 + \frac{1}{2}mR^2\left(\sin\left(\theta\right)\right)^2\dot{\phi}^2 \tag{1.1}$$

The Metric

Let's now take a step into the geometry. A surface has a metric which tells us how to measure sizes and angles at every point on the surface. (Metrics are introduced in Chapter 9.)

The metric is a symmetric function of two vector fields that gives a number for every point on the manifold. (Vector fields are introduced in Chapter 3). Metrics may be used to compute the length of a vector field at each point, or alternatively to compute the inner product of two vector fields at each point. For example, the metric for the sphere of radius R is

$$\mathsf{g}(\mathsf{u},\mathsf{v}) = R^2\mathsf{d}\theta(\mathsf{u})\mathsf{d}\theta(\mathsf{v}) + R^2(\sin\theta)^2\mathsf{d}\phi(\mathsf{u})\mathsf{d}\phi(\mathsf{v}), \tag{1.2}$$

where u and v are vector fields, and $\mathsf{d}\theta$ and $\mathsf{d}\phi$ are one-form fields that extract the named components of the vector-field argument. (One-form fields are introduced in Chapter 3.) We can think of $\mathsf{d}\theta(\mathsf{u})$ as a function of a point that gives the size of the vector field u in the θ direction at the point. Notice that $\mathsf{g}(\mathsf{u},\mathsf{u})$ is a weighted sum of the squares of the components of u. In fact, if we identify

$$\mathsf{d}\theta(\mathsf{v}) = \dot{\theta}$$
$$\mathsf{d}\phi(\mathsf{v}) = \dot{\phi},$$

then the coefficients in the metric are the same as the coefficients in the value of the Lagrangian, equation (1.1), apart from a factor of $m/2$.

We can generalize this result and write a Lagrangian for free motion of a particle of mass m on a manifold with metric g:

$$L_2(x,v) = \sum_{ij} \tfrac{1}{2}mg_{ij}(x)\,v^iv^j. \tag{1.3}$$

This is written using indexed variables to indicate components of the geometric objects expressed with respect to an unspecified coordinate system. The metric coefficients g_{ij} are, in general, a function of the position coordinates x, because the properties of the space may vary from place to place.

We can capture this geometric statement as a program:

```
(define ((L2 mass metric) place velocity)
  (* 1/2 mass ((metric velocity velocity) place)))
```

This program gives the Lagrangian in a coordinate-independent, geometric way. It is entirely in terms of geometric objects, such as a place on the configuration manifold, the velocity at that place, and the metric that describes the local shape of the manifold. But to compute we need a coordinate system. We express the dynamical state in terms of coordinates and velocity components in the coordinate system. For each coordinate system there is a natural vector basis and the geometric velocity vectors can be constructed by contracting the basis with the components of the velocity. Thus, we can form a coordinate representation of the Lagrangian.

```
(define ((Lc mass metric coordsys) state)
  (let ((x (coordinates state)) (v (velocities state))
        (e (coordinate-system->vector-basis coordsys)))
    ((L2 mass metric) ((point coordsys) x) (* e v))))
```

The manifold point `m` represented by the coordinates x is given by `(define m ((point coordsys) x))`. The coordinates of `m` in a different coordinate system are given by `((chart coordsys2) m)`. The manifold point `m` is a geometric object that is the same point independent of how it is specified. Similarly, the velocity vector `e`v is a geometric object, even though it is specified using components v with respect to the basis `e`. Both v and `e` have as many components as the dimension of the space so their product is interpreted as a contraction.

Let's make a general metric on a 2-dimensional real manifold:[4]

```
(define the-metric (literal-metric 'g R2-rect))
```

[4]The procedure `literal-metric` provides a metric. It is a general symmetric function of two vector fields, with literal functions of the coordinates of the manifold points for its coefficients in the given coordinate system. The quoted symbol `'g` is used to make the names of the literal coefficient functions. Literal functions are discussed in Appendix B.

The metric is expressed in rectangular coordinates, so the coordinate system is `R2-rect`.[5] The component functions will be labeled as subscripted gs.

We can now make the Lagrangian for the system:

```
(define L (Lc 'm the-metric R2-rect))
```

And we can apply our Lagrangian to an arbitrary state:

```
(L (up 't (up 'x 'y) (up 'vx 'vy)))
(+ (* 1/2 m (g_00 (up x y)) (expt vx 2))
   (* m (g_01 (up x y)) vx vy)
   (* 1/2 m (g_11 (up x y)) (expt vy 2)))
```

Compare this result with equation (1.3).

Euler-Lagrange Residuals

The Euler-Lagrange equations are satisfied on realizable paths. Let γ be a path on the manifold of configurations. (A path is a map from the 1-dimensional real line to the configuration manifold. We introduce maps between manifolds in Chapter 6.) Consider an arbitrary path:[6]

```
(define gamma (literal-manifold-map 'q R1-rect R2-rect))
```

The values of γ are points on the manifold, not a coordinate representation of the points. We may evaluate `gamma` only on points of the real-line manifold; `gamma` produces points on the $\mathbf{R}^2$ manifold. So to go from the literal real-number coordinate `'t` to a point on the real line we use `((point R1-rect) 't)` and to go from a point `m` in $\mathbf{R}^2$ to its coordinate representation we use `((chart R2-rect) m)`. (The procedures `point` and `chart` are introduced in Chapter 2.) Thus

[5] `R2-rect` is the usual rectangular coordinate system on the 2-dimensional real manifold. (See section 2.1, page 13.) We supply common coordinate systems for n-dimensional real manifolds. For example, `R2-polar` is a polar coordinate system on the same manifold.

[6] The procedure `literal-manifold-map` makes a map from the manifold implied by its second argument to the manifold implied by the third argument. These arguments must be coordinate systems. The quoted symbol that is the first argument is used to name the literal coordinate functions that define the map.

```
((chart R2-rect) (gamma ((point R1-rect) 't)))
(up (q^0 t) (q^1 t))
```

So, to work with coordinates we write:

```
(define coordinate-path
  (compose (chart R2-rect) gamma (point R1-rect)))

(coordinate-path 't)
(up (q^0 t) (q^1 t))
```

Now we can compute the residuals of the Euler-Lagrange equations, but we get a large messy expression that we will not show.[7] However, we will save it to compare with the residuals of the geodesic equations.

```
(define Lagrange-residuals
  (((Lagrange-equations L) coordinate-path) 't))
```

Geodesic Equations

Now we get deeper into the geometry. The traditional way to write the geodesic equations is

$$\nabla_{\mathsf{v}} \mathsf{v} = 0 \tag{1.4}$$

where ∇ is a covariant derivative operator. Roughly, $\nabla_{\mathsf{v}} \mathsf{w}$ is a directional derivative. It gives a measure of the variation of the vector field w as you walk along the manifold in the direction of v. (We will explain this in depth in Chapter 7.) $\nabla_{\mathsf{v}} \mathsf{v} = 0$ is intended to convey that the velocity vector is parallel-transported by itself. When you walked East on the Equator you had to hold the stick so that it was parallel to the Equator. But the stick is constrained to the surface of the Earth, so moving it along the Equator required turning it in three dimensions. The ∇ thus must incorporate the 3-dimensional shape of the Earth to provide a notion of "parallel" appropriate for the denizens of the surface of the Earth. This information will appear as the "Christoffel coefficients" in the coordinate representation of the geodesic equations.

The trouble with the traditional way to write the geodesic equations (1.4) is that the arguments to the covariant derivative are

[7]For an explanation of equation residuals see page xvi.

vector fields and the velocity along the path is not a vector field. A more precise way of stating this relation is:

$$\nabla^{\gamma}_{\partial/\partial \mathsf{t}} d\gamma(\partial/\partial \mathsf{t}) = 0. \tag{1.5}$$

(We know that this may be unfamiliar notation, but we will explain it in Chapter 7.)

In coordinates, the geodesic equations are expressed

$$D^2 q^i(t) + \sum_{jk} \Gamma^i_{jk}(\gamma(t)) Dq^j(t) Dq^k(t) = 0, \tag{1.6}$$

where $q(t)$ is the coordinate path corresponding to the manifold path γ, and $\Gamma^i_{jk}(\mathsf{m})$ are Christoffel coefficients. The $\Gamma^i_{jk}(\mathsf{m})$ describe the "shape" of the manifold close to the manifold point m. They can be derived from the metric g.

We can get and save the geodesic equation residuals by:

```
(define geodesic-equation-residuals
  (((((covariant-derivative Cartan gamma) d/dt)
     ((differential gamma) d/dt))
    (chart R2-rect))
   ((point R1-rect) 't)))
```

where `d/dt` is a vector field on the real line[8] and `Cartan` is a way of encapsulating the geometry, as specified by the Christoffel coefficients. The Christoffel coefficients are computed from the metric:

```
(define Cartan
  (Christoffel->Cartan
   (metric->Christoffel-2 the-metric
         (coordinate-system->basis R2-rect))))
```

The two messy residual results that we did not show are related by the metric. If we change the representation of the geodesic equations by "lowering" them using the mass and the metric, we see that the residuals are equal:

[8]We established `t` as a coordinate function on the rectangular coordinates of the real line by

```
(define-coordinates t R1-rect)
```

This had the effect of also defining `d/dt` as a coordinate vector field and `dt` as a one-form field on the real line.

```
(define metric-components
  (metric->components the-metric
                      (coordinate-system->basis R2-rect)))

(- Lagrange-residuals
   (* (* 'm (metric-components (gamma ((point R1-rect) 't))))
      geodesic-equation-residuals))
(down 0 0)
```

This establishes that for a 2-dimensional space the Euler-Lagrange equations are equivalent to the geodesic equations. The Christoffel coefficients that appear in the geodesic equation correspond to coefficients of terms in the Euler-Lagrange equations. This analysis will work for any number of dimensions (but will take your computer longer in higher dimensions, because the complexity increases).

Exercise 1.1: Motion on a Sphere

The metric for a unit sphere, expressed in colatitude θ and longitude ϕ, is

$$\mathsf{g}(\mathsf{u},\mathsf{v}) = \mathsf{d}\theta(\mathsf{u})\mathsf{d}\theta(\mathsf{v}) + (\sin\theta)^2\mathsf{d}\phi(\mathsf{u})\mathsf{d}\phi(\mathsf{v}).$$

Compute the Lagrange equations for motion of a free particle on the sphere and convince yourself that they describe great circles. For example, consider motion on the equator ($\theta = \pi/2$) and motion on a line of longitude (ϕ is constant).

2
Manifolds

A *manifold* is a generalization of our idea of a smooth surface embedded in Euclidean space. For an n-dimensional manifold, around every point there is a simply-connected open set, the *coordinate patch*, and a one-to-one continuous function, the *coordinate function* or *chart*, mapping every point in that open set to a tuple of n real numbers, the *coordinates*. In general, several charts are needed to label all points on a manifold. It is required that if a region is in more than one coordinate patch then the coordinates are consistent in that the function mapping one set of coordinates to another is continuous (and perhaps differentiable to some degree). A consistent system of coordinate patches and coordinate functions that covers the entire manifold is called an *atlas*.

An example of a 2-dimensional manifold is the surface of a sphere or of a coffee cup. The space of all configurations of a planar double pendulum is a more abstract example of a 2-dimensional manifold. A manifold that looks locally Euclidean may not look like Euclidean space globally: for example, it may not be simply connected. The surface of the coffee cup is not simply connected, because there is a hole in the handle for your fingers.

An example of a coordinate function is the function that maps points in a simply-connected open neighborhood of the surface of a sphere to the tuple of latitude and longitude.[1] If we want to talk about motion on the Earth, we can identify the space of configurations to a 2-sphere (the surface of a 3-dimensional ball). The map from the 2-sphere to the 3-dimensional coordinates of a point on the surface of the Earth captures the shape of the Earth.

Two angles specify the configuration of the planar double pendulum. The manifold of configurations is a torus, where each point on the torus corresponds to a configuration of the double pendulum. The constraints, such as the lengths of the pendulum rods, are built into the map between the generalized coordi-

[1]The open set for a latitude-longitude coordinate system cannot include either pole (because longitude is not defined at the poles) or the 180° meridian (where the longitude is discontinuous). Other coordinate systems are needed to cover these places.

nates of points on the torus and the arrangements of masses in 3-dimensional space.

There are computational objects that we can use to model manifolds. For example, we can make an object that represents the plane[2]

```
(define R2 (make-manifold R^n 2))
```

and give it the name R2. One useful patch of the plane is the one that contains the origin and covers the entire plane.[3]

```
(define U (patch 'origin R2))
```

2.1 Coordinate Functions

A coordinate function χ maps points in a coordinate patch of a manifold to a coordinate tuple:[4]

$$x = \chi(\mathsf{m}), \tag{2.1}$$

where x may have a convenient tuple structure. Usually, the coordinates are arranged as an "up structure"; the coordinates are selected with superscripts:

$$x^i = \chi^i(\mathsf{m}). \tag{2.2}$$

The number of independent components of x is the dimension of the manifold.

Assume we have two coordinate functions χ and χ'. The coordinate transformation from χ' coordinates to χ coordinates is just the composition $\chi \circ \chi'^{-1}$, where χ'^{-1} is the functional inverse of χ' (see figure 2.1). We assume that the coordinate transformation is continuous and differentiable to any degree we require.

[2] The expression `R^n` gives only one kind of manifold. We also have spheres `S^n` and `SO3`.

[3] The word `origin` is an arbitrary symbol here. It labels a predefined patch in `R^n` manifolds.

[4] In the text that follows we will use sans-serif names, such as f, v, m, to refer to objects defined on the manifold. Objects that are defined on coordinates (tuples of real numbers) will be named with symbols like f, v, x.

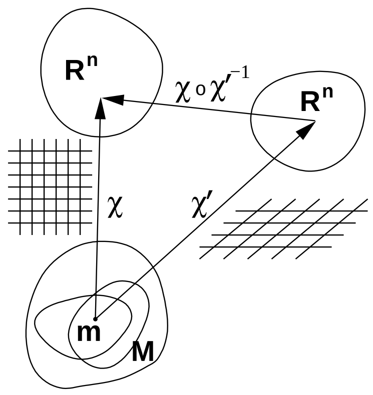

Figure 2.1 Here there are two overlapping coordinate patches that are the domains of the two coordinate functions χ and χ'. It is possible to represent manifold points in the overlap using either coordinate system. The coordinate transformation from χ' coordinates to χ coordinates is just the composition $\chi \circ \chi'^{-1}$.

Given a coordinate system `coordsys` for a patch on a manifold the procedure that implements the function χ that gives coordinates for a point is `(chart coordsys)`. The procedure that implements the inverse map that gives a point for coordinates is `(point coordsys)`.

We can have both rectangular and polar coordinates on a patch of the plane identified by the origin:[5,6]

```
;; Some charts on the patch U
(define R2-rect (coordinate-system 'rectangular U))
(define R2-polar (coordinate-system 'polar/cylindrical U))
```

For each of the coordinate systems above we obtain the coordinate functions and their inverses:

[5]The rectangular coordinates are good for the entire plane, but the polar coordinates are singular at the origin because the angle is not defined. Also, the patch for polar coordinates must exclude one ray from the origin, because of the angle variable.

[6]We can avoid explicitly naming the patch:

```
(define R2-rect (coordinate-system-at 'rectangular 'origin R2))
```

```
(define R2-rect-chi (chart R2-rect))
(define R2-rect-chi-inverse (point R2-rect))
(define R2-polar-chi (chart R2-polar))
(define R2-polar-chi-inverse (point R2-polar))
```

The coordinate transformations are then just compositions. The polar coordinates of a rectangular point are:

```
((compose R2-polar-chi R2-rect-chi-inverse)
 (up 'x0 'y0))
(up (sqrt (+ (expt x0 2) (expt y0 2))) (atan y0 x0))
```

And the rectangular coordinates of a polar point are:

```
((compose R2-rect-chi R2-polar-chi-inverse)
 (up 'r0 'theta0))
(up (* r0 (cos theta0)) (* r0 (sin theta0)))
```

And we can obtain the Jacobian of the polar-to-rectangular transformation by taking its derivative[7]

```
((D (compose R2-rect-chi R2-polar-chi-inverse))
 (up 'r0 'theta0))
(down (up (cos theta0) (sin theta0))
      (up (* -1 r0 (sin theta0)) (* r0 (cos theta0))))
```

2.2 Manifold Functions

Let f be a real-valued function on a manifold M: this function maps points m on the manifold to real numbers.

This function has a coordinate representation f_χ with respect to the coordinate function χ:

$$f_\chi = \mathsf{f} \circ \chi^{-1}. \tag{2.3}$$

Both the coordinate representation f_χ and the tuple x depend on the coordinate system, but the value $f_\chi(x)$ is independent of coordinates:

$$f_\chi(x) = (\mathsf{f} \circ \chi^{-1})(\chi(\mathsf{m})) = \mathsf{f}(\mathsf{m}). \tag{2.4}$$

[7]See Appendix B for an introduction to tuple arithmetic and a discussion of derivatives of functions with structured input or output.

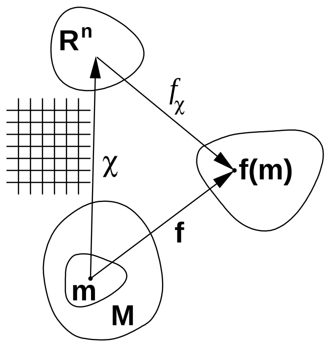

Figure 2.2 The coordinate function χ maps points on the manifold in the coordinate patch to a tuple of coordinates. A function f on the manifold M can be represented in coordinates by a function $f_\chi = \mathsf{f} \circ \chi^{-1}$.

The subscript χ may be dropped when it is unambiguous.

For example, in a 2-dimensional real manifold the coordinates of a manifold point m are a pair of real numbers,

$$(x, y) = \chi(\mathsf{m}), \tag{2.5}$$

and the manifold function f is represented in coordinates by a function f that takes a pair of real numbers and produces a real number

$$\begin{aligned} f &: \mathsf{R}^2 \to \mathsf{R} \\ f &: (x, y) \mapsto f(x, y). \end{aligned} \tag{2.6}$$

We define our manifold function

$$\begin{aligned} \mathsf{f} &: \mathsf{M} \to \mathsf{R} \\ \mathsf{f} &: \mathsf{m} \mapsto (f \circ \chi)(\mathsf{m}). \end{aligned} \tag{2.7}$$

Manifold Functions are Coordinate Independent

We can illustrate the coordinate independence with a program. We will show that an arbitrary manifold function f, when defined by its coordinate representation in rectangular coordinates, has the same behavior when applied to a manifold point independent of whether the point is specified in rectangular or polar coordinates.

We define a manifold function by specifying its behavior in rectangular coordinates:[8]

```
(define f
  (compose (literal-function 'f-rect R2->R) R2-rect-chi))
```

where `R2->R` is a signature for functions that map an up structure of two reals to a real:

```
(define R2->R (-> (UP Real Real) Real))
```

We can specify a typical manifold point using its rectangular coordinates:

```
(define R2-rect-point (R2-rect-chi-inverse (up 'x0 'y0)))
```

We can describe the *same point* using its polar coordinates:

```
(define corresponding-polar-point
  (R2-polar-chi-inverse
   (up (sqrt (+ (square 'x0) (square 'y0)))
       (atan 'y0 'x0))))
```

`(f R2-rect-point)` and `(f corresponding-polar-point)` agree, even though the point has been specified in two different coordinate systems:

```
(f R2-rect-point)
(f-rect (up x0 y0))

(f corresponding-polar-point)
(f-rect (up x0 y0))
```

Naming Coordinate Functions

To make things a bit easier, we can give names to the individual coordinate functions associated with a coordinate system. Here we name the coordinate functions for the `R2-rect` coordinate system `x` and `y` and for the `R2-polar` coordinate system `r` and `theta`.

```
(define-coordinates (up x y) R2-rect)
(define-coordinates (up r theta) R2-polar)
```

[8] Alternatively, we can define the same function in a shorthand

```
(define f (literal-manifold-function 'f-rect R2-rect))
```

This allows us to extract the coordinates from a point, independent of the coordinate system used to specify the point.

```
(x (R2-rect-chi-inverse (up 'x0 'y0)))
x0

(x (R2-polar-chi-inverse (up 'r0 'theta0)))
(* r0 (cos theta0))

(r (R2-polar-chi-inverse (up 'r0 'theta0)))
r0

(r (R2-rect-chi-inverse (up 'x0 'y0)))
(sqrt (+ (expt x0 2) (expt y0 2)))

(theta (R2-rect-chi-inverse (up 'x0 'y0)))
(atan y0 x0)
```

We can work with the coordinate functions in a natural manner, defining new manifold functions in terms of them:[9]

```
(define h (+ (* x (square r)) (cube y)))

(h R2-rect-point)
(+ (expt x0 3) (* x0 (expt y0 2))
   (expt y0 3))
```

We can also apply `h` to a point defined in terms of its polar coordinates:

```
(h (R2-polar-chi-inverse (up 'r0 'theta0)))
(+ (* (expt r0 3) (expt (sin theta0) 3))
   (* (expt r0 3) (cos theta0)))
```

Exercise 2.1: Curves

A curve may be specified in different coordinate systems. For example, a cardioid constructed by rolling a circle of radius a around another circle of the same radius is described in polar coordinates by the equation

$r = 2a(1 + \cos(\theta))$.

[9]This is actually a nasty, but traditional, abuse of notation. An expression like $\cos(r)$ can either mean the cosine of the angle r (if r is a number), or the composition $\cos \circ r$ (if r is a function). In our system `(cos r)` behaves in this way—either computing the cosine of `r` or being treated as `(compose cos r)` depending on what `r` is.

We can convert this to rectangular coordinates by evaluating the residual in rectangular coordinates.

```
(define-coordinates (up r theta) R2-polar)

((- r (* 2 'a (+ 1 (cos theta))))
 ((point R2-rect) (up 'x 'y)))

(/ (+ (* -2 a x)
      (* -2 a (sqrt (+ (expt x 2) (expt y 2))))
      (expt x 2) (expt y 2))
   (sqrt (+ (expt x 2) (expt y 2))))
```

The numerator of this expression is the equivalent residual in rectangular coordinates. If we rearrange terms and square it we get the traditional formula for the cardioid

$$\left(x^2 + y^2 - 2ax\right)^2 = 4a^2 \left(x^2 + y^2\right).$$

a. The rectangular coordinate equation for the Lemniscate of Bernoulli is

$$(x^2 + y^2)^2 = 2a^2(x^2 - y^2).$$

Find the expression in polar coordinates.

b. Describe a helix space curve in both rectangular and cylindrical coordinates. Use the computer to show the correspondence. Note that we provide a cylindrical coordinate system on the manifold $\mathbf{R}^3$ for you to use. It is called `R3-cyl`; with coordinates `(r, theta, z)`.

Exercise 2.2: Stereographic Projection

A stereographic projection is a correspondence between points on the unit sphere and points on the plane cutting the sphere at its equator. (See figure 2.3.)

The coordinate system for points on the sphere in terms of rectangular coordinates of corresponding points on the plane is `S2-Riemann`.[10] The procedure `(chart S2-Riemann)` gives the rectangular coordinates on the plane for every point on the sphere, except for the North Pole. The procedure `(point S2-Riemann)` gives the point on the sphere given rectangular coordinates on the plane. The usual spherical coordinate system on the sphere is `S2-spherical`.

We can compute the colatitude and longitude of a point on the sphere corresponding to a point on the plane with the following incantation:

[10] The plane with the addition of a point at infinity is conformally equivalent to the sphere by this correspondence. This correspondence is called the Riemann sphere, in honor of the great mathematician Bernard Riemann (1826–1866), who made major contributions to geometry.

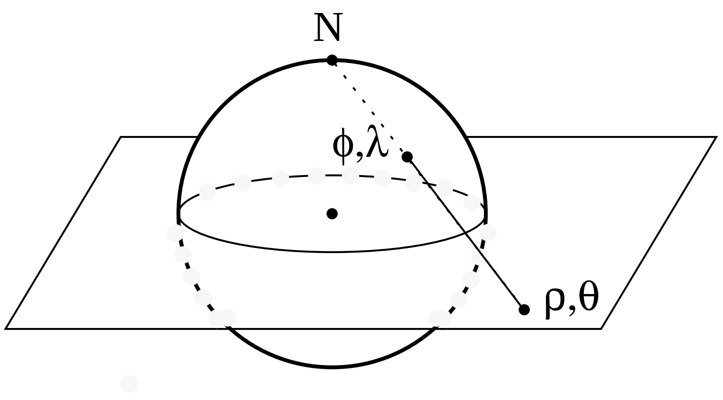

Figure 2.3 For each point on the sphere (except for its north pole) a line is drawn from the north pole through the point and extending to the equatorial plane. The corresponding point on the plane is where the line intersects the plane. The rectangular coordinates of this point on the plane are the Riemann coordinates of the point on the sphere. The points on the plane can also be specified with polar coordinates (ρ, θ) and the points on the sphere are specified both by Riemann coordinates and the traditional colatitude and longitude (ϕ, λ).

```
((compose
  (chart S2-spherical)
  (point S2-Riemann)
  (chart R2-rect)
  (point R2-polar))
 (up 'rho 'theta))
(up (acos (/ (+ -1 (expt rho 2))
             (+ +1 (expt rho 2))))
    theta)
```

Perform an analogous computation to get the polar coordinates of the point on the plane corresponding to a point on the sphere given by its colatitude and longitude.

3 Vector Fields and One-Form Fields

We want a way to think about how a function varies on a manifold. Suppose we have some complex linkage, such as a multiple pendulum. The potential energy is an important function on the multi-dimensional configuration manifold of the linkage. To understand the dynamics of the linkage we need to know how the potential energy changes as the configuration changes. The change in potential energy for a step of a certain size in a particular direction in the configuration space is a real physical quantity; it does not depend on how we measure the direction or the step size. What exactly this means is to be determined: What is a step size? What is a direction? We cannot subtract two configurations to determine the distance between them. It is our job here to make sense of this idea.

So we would like something like a derivative, but there are problems. Since we cannot subtract two manifold points, we cannot take the derivative of a manifold function in the way described in elementary calculus. But we can take the derivative of a coordinate representation of a manifold function, because it takes real-number coordinates as its arguments. This is a start, but it is not independent of coordinate system. Let's see what we can build out of this.

3.1 Vector Fields

In multiple dimensions the derivative of a function is the multiplier for the best linear approximation of the function at each argument point:[1]

$$f(x + \Delta x) \approx f(x) + (Df(x))\Delta x \tag{3.1}$$

The derivative $Df(x)$ is independent of Δx. Although the derivative depends on the coordinates, the product $(Df(x))\Delta x$ is in-

[1]In multiple dimensions the derivative $Df(x)$ is a down tuple structure of the partial derivatives and the increment Δx is an up tuple structure, so the indicated product is to be interpreted as a contraction. (See equation B.8.)

variant under change of coordinates in the following sense. Let $\phi = \chi \circ \chi'^{-1}$ be a coordinate transformation, and $x = \phi(y)$. Then $\Delta x = D\phi(y)\Delta y$ is the linear approximation to the change in x when y changes by Δy. If f and g are the representations of a manifold function in the two coordinate systems, $g(y) = f(\phi(y)) = f(x)$, then the linear approximations to the increments in f and g are equal:

$$Dg(y)\Delta y = Df(\phi(y))\,(D\phi(y)\Delta y) = Df(x)\Delta x.$$

The invariant product $(Df(x))\Delta x$ is the *directional derivative* of f at x with respect to the vector specified by the tuple of components Δx in the coordinate system. We can generalize this idea to allow the vector at each point to depend on the point, making a *vector field*. Let b be a function of coordinates. We then have a directional derivative of f at each point x, determined by b

$$D_b(f)(x) = (Df(x))b(x). \tag{3.2}$$

Now we bring this back to the manifold and develop a useful generalization of the idea of directional derivative for functions on a manifold, rather than functions on R^n. A *vector field on a manifold* is an assignment of a vector to each point on the manifold. In elementary geometry, a vector is an arrow anchored at a point on the manifold with a magnitude and a direction. In differential geometry, a vector is an operator that takes directional derivatives of manifold functions at its anchor point. The direction and magnitude of the vector are the direction and scale factor of the directional derivative.

Let m be a point on a manifold, v be a vector field on the manifold, and f be a real-valued function on the manifold. Then $\mathsf{v(f)}$ is the directional derivative of the function f and $\mathsf{v(f)(m)}$ is the directional derivative of the function f at the point m. The vector field is an operator that takes a real-valued manifold function and a manifold point and produces a number. The order of arguments is chosen to make $\mathsf{v(f)}$ be a new manifold function that can be manipulated further. Directional derivative operators, unlike ordinary derivative operators, produce a result of the same type as their argument. Note that there is no mention here of any coordinate system. The vector field specifies a direction and magnitude at each manifold point that is independent of how it is described using any coordinate system.

A useful way to characterize a vector field in a particular coordinate system is by applying it to the coordinate functions. The resulting functions $b^i_{\chi,\mathsf{v}}$ are called the *coordinate component functions* or *coefficient functions* of the vector field; they measure how quickly the coordinate functions change in the direction of the vector field, scaled by the magnitude of the vector field:

$$b^i_{\chi,\mathsf{v}} = \mathsf{v}\left(\chi^i\right) \circ \chi^{-1}. \tag{3.3}$$

Note that we have chosen the coordinate components to be functions of the coordinate tuple, not of a manifold point.

A vector with coordinate components $b_{\chi,\mathsf{v}}$ applies to a manifold function f via

$$\begin{aligned}
\mathsf{v}(\mathsf{f})(\mathsf{m}) &= ((D(\mathsf{f} \circ \chi^{-1})\ b_{\chi,\mathsf{v}}) \circ \chi)(\mathsf{m}) && (3.4)\\
&= D(\mathsf{f} \circ \chi^{-1})(\chi(\mathsf{m}))\ b_{\chi,\mathsf{v}}(\chi(\mathsf{m})) && (3.5)\\
&= \sum_i \partial_i(\mathsf{f} \circ \chi^{-1})(\chi(\mathsf{m}))\ b^i_{\chi,\mathsf{v}}(\chi(\mathsf{m})). && (3.6)
\end{aligned}$$

In equation (3.4), the quantity $\mathsf{f}\circ\chi^{-1}$ is the coordinate representation of the manifold function f. We take its derivative, and weight the components of the derivative with the coordinate components $b_{\chi,\mathsf{v}}$ of the vector field that specify its direction and magnitude. Since this product is a function of coordinates we use χ to extract the coordinates from the manifold point m. In equation (3.5), the composition of the product with the coordinate chart χ is replaced by function evaluation. In equation (3.6) the tuple multiplication is expressed explicitly as a sum of products of corresponding components. So the application of the vector is a linear combination of the partial derivatives of f in the coordinate directions weighted by the vector components. This computes the rate of change of f in the direction specified by the vector.

Equations (3.3) and (3.5) are consistent:

$$\begin{aligned}
\mathsf{v}(\chi)(\chi^{-1}(x)) &= D(\chi \circ \chi^{-1})(x)\ b_{\chi,\mathsf{v}}(x)\\
&= D(I)(x)\ b_{\chi,\mathsf{v}}(x)\\
&= b_{\chi,\mathsf{v}}(x) && (3.7)
\end{aligned}$$

The coefficient tuple $b_{\chi,\mathsf{v}}(x)$ is an up structure compatible for addition to the coordinates. Note that for any vector field v the coefficients $b_{\chi,\mathsf{v}}(x)$ are different for different coordinate functions χ.

In the text that follows we will usually drop the subscripts on b, understanding that it is dependent on the coordinate system and the vector field.

We implement the definition of a vector field (3.4) as:

```
(define (components->vector-field components coordsys)
  (define (v f)
    (compose (* (D (compose f (point coordsys)))
                components)
             (chart coordsys)))
  (procedure->vector-field v))
```

The vector field is an operator, like derivative.[2]

Given a coordinate system and coefficient functions that map coordinates to real values, we can make a vector field. For example, a general vector field can be defined by giving components relative to the coordinate system `R2-rect` by

```
(define v
  (components->vector-field
   (up (literal-function 'b^0 R2->R)
       (literal-function 'b^1 R2->R))
   R2-rect))
```

To make it convenient to define literal vector fields we provide a shorthand: `(define v (literal-vector-field 'b R2-rect))` This makes a vector field with component functions named `b^0` and `b^1` and names the result `v`. When this vector field is applied to an arbitrary manifold function it gives the directional derivative of that manifold function in the direction specified by the components `b^0` and `b^1`:

```
((v (literal-manifold-function 'f-rect R2-rect)) R2-rect-point)
(+ (* (((partial 0) f-rect) (up x0 y0)) (b^0 (up x0 y0)))
   (* (((partial 1) f-rect) (up x0 y0)) (b^1 (up x0 y0))))
```

This result is what we expect from equation (3.6).

We can recover the coordinate components of the vector field by applying the vector field to the coordinate chart:

[2]An operator is just like a procedure except that multiplication is interpreted as composition. For example, the derivative procedure is made into an operator `D` so that we can say `(expt D 2)` and expect it to compute the second derivative. The procedure `procedure->vector-field` makes a vector-field operator.

```
((v (chart R2-rect)) R2-rect-point)
(up (b^0 (up x y)) (b^1 (up x y)))
```

Coordinate Representation

The vector field v has a coordinate representation v:

$$\begin{aligned} \mathsf{v}(\mathsf{f})(\mathsf{m}) &= D(\mathsf{f} \circ \chi^{-1})(\chi(\mathsf{m}))\ b(\chi(\mathsf{m})) \\ &= Df(x)\ b(x) \\ &= v(f)(x), \end{aligned} \tag{3.8}$$

with the definitions $f = \mathsf{f} \circ \chi^{-1}$ and $x = \chi(\mathsf{m})$. The function b is the coefficient function for the vector field v. It provides a scale factor for the component in each coordinate direction. However, v is the coordinate representation of the vector field v in that it takes directional derivatives of coordinate representations of manifold functions.

Given a vector field v and a coordinate system coordsys we can construct the coordinate representation of the vector field.[3]

```
(define (coordinatize v coordsys)
  (define ((coordinatized-v f) x)
    (let ((b (compose (v (chart coordsys))
                      (point coordsys))))
      (* ((D f) x) (b x))))
  (make-operator coordinatized-v))
```

We can apply a coordinatized vector field to a function of coordinates to get the same answer as before.

```
(((coordinatize v R2-rect) (literal-function 'f-rect R2->R))
 (up 'x0 'y0))
(+ (* (((partial 0) f-rect) (up x0 y0)) (b^0 (up x0 y0)))
   (* (((partial 1) f-rect) (up x0 y0)) (b^1 (up x0 y0))))
```

Vector Field Properties

The vector fields on a manifold form a vector space over the field of real numbers and a module over the ring of real-valued manifold functions. A module is like a vector space except that there is no multiplicative inverse operation on the scalars of a module. Manifold functions that are not the zero function do not necessarily

[3]The make-operator procedure takes a procedure and returns an operator.

have multiplicative inverses, because they can have isolated zeros. So the manifold functions form a ring, not a field, and vector fields must be a module over the ring of manifold functions rather than a vector space.

Vector fields have the following properties. Let u and v be vector fields and let α be a real-valued manifold function. Then

$$(\mathsf{u}+\mathsf{v})(\mathsf{f}) = \mathsf{u}(\mathsf{f}) + \mathsf{v}(\mathsf{f}) \tag{3.9}$$

$$(\alpha\mathsf{u})(\mathsf{f}) = \alpha(\mathsf{u}(\mathsf{f})). \tag{3.10}$$

Vector fields are linear operators. Assume f and g are functions on the manifold, a and b are real constants.[4] The constants a and b are not manifold functions, because vector fields take derivatives. See equation (3.13).

$$\mathsf{v}(a\mathsf{f} + b\mathsf{g})(\mathsf{m}) = a\mathsf{v}(\mathsf{f})(\mathsf{m}) + b\mathsf{v}(\mathsf{g})(\mathsf{m}) \tag{3.11}$$

$$\mathsf{v}(a\mathsf{f})(\mathsf{m}) = a\mathsf{v}(\mathsf{f})(\mathsf{m}) \tag{3.12}$$

Vector fields satisfy the product rule (Leibniz rule).

$$\mathsf{v}(\mathsf{fg})(\mathsf{m}) = \mathsf{v}(\mathsf{f})(\mathsf{m})\,\mathsf{g}(\mathsf{m}) + \mathsf{f}(\mathsf{m})\,\mathsf{v}(\mathsf{g})(\mathsf{m}) \tag{3.13}$$

Vector fields satisfy the chain rule. Let F be a function on the range of f.

$$\mathsf{v}(F \circ \mathsf{f})(\mathsf{m}) = DF(\mathsf{f}(\mathsf{m}))\,\mathsf{v}(\mathsf{f})(\mathsf{m}) \tag{3.14}$$

3.2 Coordinate-Basis Vector Fields

For an n-dimensional manifold any set of n linearly independent vector fields[5] form a *basis* in that any vector field can be expressed as a linear combination of the basis fields with manifold-function

[4]If f has structured output then $\mathsf{v}(\mathsf{f})$ is the structure resulting from v being applied to each component of f.

[5] A set of vector fields, $\{\mathsf{v}_i\}$, is linearly independent with respect to manifold functions if we cannot find nonzero manifold functions, $\{\mathsf{a}_i\}$, such that

$$\sum_i \mathsf{a}_i \mathsf{v}_i(\mathsf{f}) = \mathsf{0}(\mathsf{f}),$$

where $\mathsf{0}$ is the vector field such that $\mathsf{0}(\mathsf{f})(\mathsf{m}) = 0$ for all f and m.

coefficients. Given a coordinate system we can construct a basis as follows: we choose the component tuple $b_i(x)$ (see equation 3.5) to be the ith unit tuple $u_i(x)$—an up tuple with one in the ith position and zeros in all other positions—selecting the partial derivative in that direction. Here u_i is a constant function. Like b, it formally takes coordinates of a point as an argument, but it ignores them. We then define the basis vector field X_i by

$$\begin{aligned}\mathsf{X}_i(\mathsf{f})(\mathsf{m}) &= D(\mathsf{f} \circ \chi^{-1})(\chi(\mathsf{m}))\ u_i(\chi(m)) \\ &= \partial_i(\mathsf{f} \circ \chi^{-1})(\chi(\mathsf{m})).\end{aligned} \tag{3.15}$$

In terms of X_i the vector field of equation (3.6) is

$$\mathsf{v}(\mathsf{f})(\mathsf{m}) = \sum_i \mathsf{X}_i(\mathsf{f})(\mathsf{m})\ b^i(\chi(\mathsf{m})). \tag{3.16}$$

We can also write

$$\mathsf{v}(\mathsf{f})(\mathsf{m}) = \mathsf{X}(\mathsf{f})(\mathsf{m})\ b(\chi(\mathsf{m})), \tag{3.17}$$

letting the tuple algebra do its job.

The basis vector field is often written

$$\frac{\partial}{\partial \mathsf{x}^i} = \mathsf{X}_i, \tag{3.18}$$

to call to mind that it is an operator that computes the directional derivative in the ith coordinate direction.

In addition to making the coordinate functions, the procedure `define-coordinates` also makes the traditional named basis vectors. Using these we can examine the application of a rectangular basis vector to a polar coordinate function:

```
(define-coordinates (up x y) R2-rect)
(define-coordinates (up r theta) R2-polar)

((d/dx (square r)) R2-rect-point)
(* 2 x0)
```

More general functions and vectors can be made as combinations of these simple pieces:

```
(((+ d/dx (* 2 d/dy)) (+ (square r) (* 3 x))) R2-rect-point)
(+ 3 (* 2 x0) (* 4 y0))
```

Coordinate Transformations

Consider a coordinate change from the chart χ to the chart χ'.

$$\begin{aligned}\mathsf{X}(\mathsf{f})(\mathsf{m}) &= D(\mathsf{f} \circ \chi^{-1})(\chi(\mathsf{m})) \\ &= D(\mathsf{f} \circ (\chi')^{-1} \circ \chi' \circ \chi^{-1})(\chi(\mathsf{m})) \\ &= D(\mathsf{f} \circ (\chi')^{-1})(\chi'(\mathsf{m}))(D(\chi' \circ \chi^{-1}))(\chi(\mathsf{m})) \\ &= \mathsf{X}'(\mathsf{f})(\mathsf{m})(D(\chi' \circ \chi^{-1}))(\chi(\mathsf{m})). \end{aligned} \tag{3.19}$$

This is the rule for the transformation of basis vector fields. The second factor can be recognized as "$\partial x'/\partial x$", the Jacobian.[6]

The vector field does not depend on coordinates. So, from equation (3.17), we have

$$\mathsf{v}(\mathsf{f})(\mathsf{m}) = \mathsf{X}(\mathsf{f})(\mathsf{m})\ b(\chi(\mathsf{m})) = \mathsf{X}'(\mathsf{f})(\mathsf{m})\ b'(\chi'(\mathsf{m})). \tag{3.20}$$

Using equation (3.19) with $x = \chi(\mathsf{m})$ and $x' = \chi'(\mathsf{m})$, we deduce

$$D(\chi' \circ \chi^{-1})(x)\ b(x) = b'(x'). \tag{3.21}$$

Because $\chi' \circ \chi^{-1}$ is the inverse function of $\chi \circ (\chi')^{-1}$, their derivatives are multiplicative inverses,

$$D(\chi' \circ \chi^{-1})(x) = (D(\chi \circ (\chi')^{-1})(x'))^{-1}, \tag{3.22}$$

and so

$$b(x) = D(\chi \circ (\chi')^{-1})(x')\ b'(x'), \tag{3.23}$$

as expected.[7]

It is traditional to express this rule by saying that the basis elements transform *covariantly* and the coefficients of a vector in

[6]This notation helps one remember the transformation rule:

$$\frac{\partial f}{\partial x^i} = \sum_j \frac{\partial f}{\partial x'^j} \frac{\partial x'^j}{\partial x^i},$$

which is the relation in the usual Leibniz notation. As Spivak pointed out in *Calculus on Manifolds*, p.45, f means something different on each side of the equation.

[7]For coordinate paths q and q' related by $q(t) = (\chi \circ (\chi')^{-1})(q'(t))$ the velocities are related by $Dq(t) = D(\chi \circ (\chi')^{-1})(q'(t))Dq'(t)$. Abstracting off paths, we get $v = D(\chi \circ (\chi')^{-1})(x')v'$.

terms of a basis transform *contravariantly*; their product is invariant under the transformation.

3.3 Integral Curves

A vector field gives a direction and rate for every point on a manifold. We can start at any point and go in the direction specified by the vector field, tracing out a parametric curve on the manifold. This curve is an *integral curve* of the vector field.

More formally, let v be a vector field on the manifold M. An integral curve $\gamma_{\mathsf{m}}^{\mathsf{v}} : \mathsf{R} \to \mathsf{M}$ of v is a parametric path on M satisfying

$$D(\mathsf{f} \circ \gamma_{\mathsf{m}}^{\mathsf{v}})(t) = \mathsf{v}(\mathsf{f})(\gamma_{\mathsf{m}}^{\mathsf{v}}(t)) = (\mathsf{v}(\mathsf{f}) \circ \gamma_{\mathsf{m}}^{\mathsf{v}})(t) \tag{3.24}$$

$$\gamma_{\mathsf{m}}^{\mathsf{v}}(0) = \mathsf{m}, \tag{3.25}$$

for arbitrary functions f on the manifold, with real values or structured real values. The rate of change of a function along an integral curve is the vector field applied to the function evaluated at the appropriate place along the curve. Often we will simply write γ, rather than $\gamma_{\mathsf{m}}^{\mathsf{v}}$. Another useful variation is $\phi_t^{\mathsf{v}}(\mathsf{m}) = \gamma_{\mathsf{m}}^{\mathsf{v}}(t)$.

We can recover the differential equations satisfied by a coordinate representation of the integral curve by letting $\mathsf{f} = \chi$, the coordinate function, and letting $\sigma = \chi \circ \gamma$ be the coordinate path corresponding to the curve γ. Then the derivative of the coordinate path σ is

$$\begin{aligned} D\sigma(t) &= D(\chi \circ \gamma)(t) \\ &= (\mathsf{v}(\chi) \circ \gamma)(t) \\ &= (\mathsf{v}(\chi) \circ \chi^{-1} \circ \chi \circ \gamma)(t) \\ &= (b \circ \sigma)(t), \end{aligned} \tag{3.26}$$

where $b = \mathsf{v}(\chi) \circ \chi^{-1}$ is the coefficient function for the vector field v for coordinates χ (see equation 3.7). So the coordinate path σ satisfies the differential equations

$$D\sigma = b \circ \sigma. \tag{3.27}$$

Differential equations for the integral curve can be expressed only in a coordinate representation, because we cannot go from one point on the manifold to another by addition of an increment.

We can do this only by adding the coordinates to an increment of coordinates and then finding the corresponding point on the manifold.

Iterating the process described by equation (3.24) we can compute higher-order derivatives of functions along the integral curve:

$$\begin{aligned} D(\mathsf{f} \circ \gamma) &= \mathsf{v}(\mathsf{f}) \circ \gamma \\ D^2(\mathsf{f} \circ \gamma) &= D(\mathsf{v}(\mathsf{f}) \circ \gamma) = \mathsf{v}(\mathsf{v}(\mathsf{f})) \circ \gamma \\ &\;\;... \\ D^n(\mathsf{f} \circ \gamma) &= \mathsf{v}^n(\mathsf{f}) \circ \gamma \end{aligned} \tag{3.28}$$

Thus, the evolution of $\mathsf{f} \circ \gamma$ can be written formally as a Taylor series in the parameter:

$$\begin{aligned} &(\mathsf{f} \circ \gamma)(t) \\ &\quad = (\mathsf{f} \circ \gamma)(0) + t\, D(\mathsf{f} \circ \gamma)(0) + \frac{1}{2} t^2\, D^2(\mathsf{f} \circ \gamma)(0) + \cdots \\ &\quad = (e^{tD}(\mathsf{f} \circ \gamma))(0) \\ &\quad = (e^{t\mathsf{v}}\mathsf{f})(\gamma(0)). \end{aligned} \tag{3.29}$$

Using ϕ rather than γ

$$(\mathsf{f} \circ \gamma_{\mathsf{m}}^{\mathsf{v}})(t) = (\mathsf{f} \circ \phi_t^{\mathsf{v}})(\mathsf{m}), \tag{3.30}$$

so, when the series converges,

$$(e^{t\mathsf{v}}\mathsf{f})(\mathsf{m}) = (\mathsf{f} \circ \phi_t^{\mathsf{v}})(\mathsf{m}). \tag{3.31}$$

In particular, let $\mathsf{f} = \chi$, then

$$\sigma(t) = (\chi \circ \gamma)(t) = (e^{tD}(\chi \circ \gamma))(0) = (e^{t\mathsf{v}}\chi)(\gamma(0)), \tag{3.32}$$

a Taylor series representation of the solution to the differential equation (3.27).

For example, a vector field circular that generates a rotation about the origin is:[8]

[8]In this expression `d/dx` and `d/dy` are vector fields that take directional derivatives of manifold functions and evaluate them at manifold points; `x` and `y` are manifold functions. `define-coordinates` was used to create these operators and functions, see page 27.

Note that `circular` is an operator—a property inherited from `d/dx` and `d/dy`.

```
(define circular (- (* x d/dy) (* y d/dx)))
```

We can exponentiate the `circular` vector field, to generate an evolution in a circle around the origin starting at `(1, 0)`:

```
(series:for-each print-expression
                 (((exp (* 't circular)) (chart R2-rect))
                  ((point R2-rect) (up 1 0)))
                 6)
(up 1 0)
(up 0 t)
(up (* -1/2 (expt t 2)) 0)
(up 0 (* -1/6 (expt t 3)))
(up (* 1/24 (expt t 4)) 0)
(up 0 (* 1/120 (expt t 5)))
```

These are the first six terms of the series expansion of the coordinates of the position for parameter `t`.

We can define an evolution operator $\mathsf{E}_{\Delta t,\mathsf{v}}$ using equation (3.31)

$$(\mathsf{E}_{\Delta t,\mathsf{v}}\mathsf{f})(\mathsf{m}) = (e^{\Delta t \mathsf{v}}\mathsf{f})(\mathsf{m}) = (\mathsf{f} \circ \phi^{\mathsf{v}}_{\Delta t})(\mathsf{m}). \tag{3.33}$$

We can approximate the evolution operator by summing the series up to a given order:

```
(define ((((evolution order) delta-t v) f) m)
  (series:sum
    (((exp (* delta-t v)) f) m)
    order))
```

We can evolve `circular` from the initial point up to the parameter `t`, and accumulate the first six terms as follows:

```
((((evolution 6) 'delta-t circular) (chart R2-rect))
 ((point R2-rect) (up 1 0)))
(up (+ (* -1/720 (expt delta-t 6))
       (* 1/24 (expt delta-t 4))
       (* -1/2 (expt delta-t 2))
       1)
    (+ (* 1/120 (expt delta-t 5))
       (* -1/6 (expt delta-t 3))
       delta-t))
```

Note that these are just the series for $\cos \Delta t$ and $\sin \Delta t$, so the coordinate tuple of the evolved point is $(\cos \Delta t, \sin \Delta t)$.

For functions whose series expansions have finite radius of convergence, evolution can progress beyond the point at which the Taylor series converges because evolution is well defined whenever the integral curve is defined.

Exercise 3.1: State Derivatives

Newton's equations for the motion of a particle in a plane, subject to a force that depends only on the position in the plane, are a system of second-order differential equations for the rectangular coordinates (X, Y) of the particle:

$$D^2X(t) = A_x(X(t), Y(t)) \quad \text{and} \quad D^2Y(t) = A_y(X(t), Y(t)),$$

where A is the acceleration of the particle.

These are equivalent to a system of first-order equations for the coordinate path $\sigma = \chi \circ \gamma$, where $\chi = (\mathsf{t}, \mathsf{x}, \mathsf{y}, \mathsf{v}_x, \mathsf{v}_y)$ is a coordinate system on the manifold $\mathbf{R}^5$. Then our equations are:

$$\begin{aligned}
D(\mathsf{t} \circ \gamma) &= 1 \\
D(\mathsf{x} \circ \gamma) &= \mathsf{v}_x \circ \gamma \\
D(\mathsf{y} \circ \gamma) &= \mathsf{v}_y \circ \gamma \\
D(\mathsf{v}_x \circ \gamma) &= A_x(\mathsf{x} \circ \gamma, \mathsf{y} \circ \gamma) \\
D(\mathsf{v}_y \circ \gamma) &= A_y(\mathsf{x} \circ \gamma, \mathsf{y} \circ \gamma)
\end{aligned}$$

Construct a vector field on $\mathbf{R}^5$ corresponding to this system of differential equations. Derive the first few terms in the series solution of this problem by exponentiation.

3.4 One-form Fields

A vector field that gives a velocity for each point on a topographic map of the surface of the Earth can be applied to a function, such as one that gives the height for each point on the topographic map, or a map that gives the temperature for each point. The vector field then provides the rate of change of the height or temperature as one moves in the way described by the vector field. Alternatively, we can think of a topographic map, which gives the height at each point, as measuring a velocity field at each point. For example, we may be interested in the velocity of the wind or the trajectories of migrating birds. The topographic map gives the rate of change of height at each point for each velocity vector field. The rate of change of height can be thought of as the number of equally-spaced (in height) contours that are pierced by each velocity vector in the vector field.

Differential of a Function

For example, consider the *differential*[9] df of a manifold function f, defined as follows. If df is applied to a vector field v we obtain

$$\mathsf{df}(\mathsf{v}) = \mathsf{v}(\mathsf{f}), \tag{3.34}$$

which is a function of a manifold point.

The differential of the height function on the topographic map is a function that gives the rate of change of height at each point for a velocity vector field. This gives the same answer as the velocity vector field applied to the height function.

The differential of a function is linear in the vector fields. The differential is also a linear operator on functions: if f_1 and f_2 are manifold functions, and if c is a real constant, then

$$\mathsf{d}(\mathsf{f}_1 + \mathsf{f}_2) = \mathsf{df}_1 + \mathsf{df}_2$$

and

$$\mathsf{d}(c\mathsf{f}) = c\mathsf{df}.$$

Note that c is not a manifold function.

One-Form Fields

A one-form field is a generalization of this idea; it is something that measures a vector field at each point.

One-form fields are linear functions of vector fields that produce real-valued functions on the manifold. A one-form field is linear in vector fields: if $\boldsymbol{\omega}$ is a one-form field, v and w are vector fields, and c is a manifold function, then

$$\boldsymbol{\omega}(\mathsf{v} + \mathsf{w}) = \boldsymbol{\omega}(\mathsf{v}) + \boldsymbol{\omega}(\mathsf{w}) \tag{3.35}$$

and

$$\boldsymbol{\omega}(\mathsf{cv}) = \mathsf{c}\boldsymbol{\omega}(\mathsf{v}). \tag{3.36}$$

[9] The differential of a manifold function will turn out to be a special case of the exterior derivative, which will be introduced later.

Sums and scalar products of one-form fields on a manifold have the following properties. If $\boldsymbol{\omega}$ and $\boldsymbol{\theta}$ are one-form fields, and if f is a real-valued manifold function, then:

$$(\boldsymbol{\omega}+\boldsymbol{\theta})(\mathsf{v}) = \boldsymbol{\omega}(\mathsf{v}) + \boldsymbol{\theta}(\mathsf{v}) \tag{3.37}$$

$$(\mathsf{f}\,\boldsymbol{\omega})(\mathsf{v}) = \mathsf{f}\,\boldsymbol{\omega}(\mathsf{v}) \tag{3.38}$$

3.5 Coordinate-Basis One-Form Fields

Given a coordinate function χ, we define the coordinate-basis one-form fields $\widetilde{\mathsf{X}}^i$ by

$$\widetilde{\mathsf{X}}^i(\mathsf{v})(\mathsf{m}) = \mathsf{v}(\chi^i)(\mathsf{m}) \tag{3.39}$$

or collectively

$$\widetilde{\mathsf{X}}(\mathsf{v})(\mathsf{m}) = \mathsf{v}(\chi)(\mathsf{m}). \tag{3.40}$$

With this definition the coordinate-basis one-form fields are dual to the coordinate-basis vector fields in the following sense (see equation 3.15):[10]

$$\widetilde{\mathsf{X}}^i(\mathsf{X}_j)(\mathsf{m}) = \mathsf{X}_j(\chi^i)(\mathsf{m}) = \partial_j(\chi^i \circ \chi^{-1})(\chi(\mathsf{m})) = \delta^i_j. \tag{3.41}$$

The tuple of basis one-form fields $\widetilde{\mathsf{X}}(\mathsf{v})(\mathsf{m})$ is an up structure like that of χ.

The general one-form field $\boldsymbol{\omega}$ is a linear combination of coordinate-basis one-form fields:

$$\boldsymbol{\omega}(\mathsf{v})(\mathsf{m}) = a(\chi(\mathsf{m}))\,\widetilde{\mathsf{X}}(\mathsf{v})(\mathsf{m}) = \sum_i a_i(\chi(\mathsf{m}))\,\widetilde{\mathsf{X}}^i(\mathsf{v})(\mathsf{m}), \tag{3.42}$$

with coefficient tuple $a(x)$, for $x = \chi(\mathsf{m})$. We can write this more simply as

$$\boldsymbol{\omega}(\mathsf{v}) = (a \circ \chi)\,\widetilde{\mathsf{X}}(\mathsf{v}), \tag{3.43}$$

because everything is evaluated at m.

[10]The Kronecker delta δ^i_j is one if $i = j$ and zero otherwise.

The coefficient tuple can be recovered from the one-form field:[11]

$$a_i(x) = \boldsymbol{\omega}(\mathsf{X}_i)(\chi^{-1}(x)). \tag{3.44}$$

This follows from the dual relationship (3.41). We can see this as a program:[12]

```
(define omega
  (components->1form-field
   (down (literal-function 'a_0 R2->R)
         (literal-function 'a_1 R2->R))
   R2-rect))

((omega (down d/dx d/dy)) R2-rect-point)
(down (a_0 (up x0 y0)) (a_1 (up x0 y0)))
```

We provide a shortcut for this construction:

```
(define omega (literal-1form-field 'a R2-rect))
```

A differential can be expanded in a coordinate basis:

$$\mathsf{df}(\mathsf{v}) = \sum_i \mathsf{c}_i \tilde{\mathsf{X}}^i(\mathsf{v}) \tag{3.45}$$

The coefficients $\mathsf{c}_i = \mathsf{df}(\mathsf{X}_i) = \mathsf{X}_i(\mathsf{f}) = \partial_i(\mathsf{f} \circ \chi^{-1}) \circ \chi$ are the partial derivatives of the coordinate representation of f in the coordinate system of the basis:

```
(((d (literal-manifold-function 'f-rect R2-rect))
  (coordinate-system->vector-basis R2-rect))
 R2-rect-point)
(down (((partial 0) f-rect) (up x0 y0))
      (((partial 1) f-rect) (up x0 y0)))
```

However, if the coordinate system of the basis differs from the coordinates of the representation of the function, the result is complicated by the chain rule:

[11]The analogous recovery of coefficient tuples from vector fields is equation (3.3): $b^i_{\chi,\mathsf{v}} = \mathsf{v}\,(\chi^i) \circ \chi^{-1}$.

[12]The procedure `components->1form-field` is analogous to the procedure `components->vector-field` introduced earlier.

```
(((d (literal-manifold-function 'f-polar R2-polar))
  (coordinate-system->vector-basis R2-rect))
 ((point R2-polar) (up 'r 'theta)))
(down (- (* (((partial 0) f-polar) (up r theta)) (cos theta))
         (/ (* (((partial 1) f-polar) (up r theta))
               (sin theta))
            r))
      (+ (* (((partial 0) f-polar) (up r theta)) (sin theta))
         (/ (* (((partial 1) f-polar) (up r theta))
               (cos theta))
            r)))
```

The coordinate-basis one-form fields can be used to find the coefficients of vector fields in the corresponding coordinate vector-field basis:

$$\widetilde{\mathsf{X}}^i(\mathsf{v}) = \mathsf{v}(\chi^i) = b^i \circ \chi \tag{3.46}$$

or collectively,

$$\widetilde{\mathsf{X}}(\mathsf{v}) = \mathsf{v}(\chi) = b \circ \chi \tag{3.47}$$

A coordinate-basis one-form field is often written dx^i This traditional notation for the coordinate-basis one-form fields is justified by the relation:

$$\mathsf{dx}^i = \widetilde{\mathsf{X}}^i = \mathsf{d}(\chi^i) \tag{3.48}$$

The `define-coordinates` procedure also makes the basis one-form fields with these traditional names inherited from the coordinates.

We can illustrate the duality of the coordinate-basis vector fields and the coordinate-basis one-form fields:

```
(define-coordinates (up x y) R2-rect)

((dx d/dy) R2-rect-point)
0

((dx d/dx) R2-rect-point)
1
```

We can use the coordinate-basis one-form fields to extract the coefficients of `circular` on the rectangular vector basis:

```
((dx circular) R2-rect-point)
(* -1 y0)

((dy circular) R2-rect-point)
x0
```

But we can also find the coefficients on the polar vector basis:

```
((dr circular) R2-rect-point)
0

((dtheta circular) R2-rect-point)
1
```

So `circular` is the same as `d/dtheta`, as we can see by applying them both to the general function `f`:

```
(define f (literal-manifold-function 'f-rect R2-rect))
(((- circular d/dtheta) f) R2-rect-point)
0
```

Not All One-Form Fields are Differentials

Although all one-form fields can be constructed as linear combinations of basis one-form fields, not all one-form fields are differentials of functions.

The coefficients of a differential are (see equation 3.45):

$$\mathsf{c}_i = \mathsf{X}_i(\mathsf{f}) = \mathsf{df}(\mathsf{X}_i) \tag{3.49}$$

and partial derivatives of functions commute

$$\mathsf{X}_i(\mathsf{X}_j(\mathsf{f})) = \mathsf{X}_j(\mathsf{X}_i(\mathsf{f})). \tag{3.50}$$

As a consequence, the coefficients of a differential are constrained

$$\mathsf{X}_i(\mathsf{c}_j) = \mathsf{X}_j(\mathsf{c}_i), \tag{3.51}$$

but a one-form field can be constructed with arbitrary coefficient functions. For example:

$$\mathsf{xdx} + \mathsf{xdy} \tag{3.52}$$

is not a differential of any function. This is why we started with the basis one-form fields and built the general one-form fields in terms of them.

Coordinate Transformations

Consider a coordinate change from the chart χ to the chart χ'.

$$\begin{aligned}\widetilde{\mathsf{X}}(\mathsf{v}) &= \mathsf{v}(\chi)\\ &= \mathsf{v}(\chi \circ (\chi')^{-1} \circ \chi')\\ &= (D(\chi \circ (\chi')^{-1}) \circ \chi')\ \mathsf{v}(\chi')\\ &= (D(\chi \circ (\chi')^{-1}) \circ \chi')\ \widetilde{\mathsf{X}}'(\mathsf{v}),\end{aligned} \tag{3.53}$$

where the third line follows from the chain rule for vector fields.

One-form fields are independent of coordinates. So,

$$\boldsymbol{\omega}(\mathsf{v}) = (a \circ \chi)\ \widetilde{\mathsf{X}}(\mathsf{v}) = (a' \circ \chi')\ \widetilde{\mathsf{X}}'(\mathsf{v}). \tag{3.54}$$

Eqs. (3.54) and (3.53) require that the coefficients transform under coordinate transformations as follows:

$$a(\chi(\mathsf{m}))\ D(\chi \circ (\chi')^{-1})(\chi'(\mathsf{m})) = a'(\chi'(\mathsf{m})), \tag{3.55}$$

or

$$a(\chi(\mathsf{m})) = a'(\chi'(\mathsf{m}))\ (D(\chi \circ (\chi')^{-1})(\chi'(\mathsf{m})))^{-1} \tag{3.56}$$

The coefficient tuple $a(x)$ is a down structure compatible for contraction with $b(x)$. Let v be the vector with coefficient tuple $b(x)$, and $\boldsymbol{\omega}$ be the one-form with coefficient tuple $a(x)$. Then, by equation (3.43),

$$\boldsymbol{\omega}(\mathsf{v}) = (a \circ \chi)\ (b \circ \chi). \tag{3.57}$$

As a program:

```
(define omega (literal-1form-field 'a R2-rect))

(define v (literal-vector-field 'b R2-rect))

((omega v) R2-rect-point)
(+ (* (b^0 (up x y)) (a_0 (up x0 y0)))
   (* (b^1 (up x y)) (a_1 (up x0 y0))))
```

Comparing equation (3.56) with equation (3.23) we see that one-form components and vector components transform oppositely, so that

$$a(x)\ b(x) = a'(x')\ b'(x'), \tag{3.58}$$

as expected because $\boldsymbol{\omega}(\mathsf{v})(\mathsf{m})$ is independent of coordinates.

Exercise 3.2: Verification

Verify that the coefficients of a one-form field transform as described in equation (3.56). You should use equation (3.44) in your derivation.

Exercise 3.3: Hill Climbing

The topography of a region on the Earth can be specified by a manifold function h that gives the altitude at each point on the manifold. Let v be a vector field on the manifold, perhaps specifying a direction and rate of walking at every point on the manifold.

a. Form an expression that gives the power that must be expended to follow the vector field at each point.

b. Write this as a computational expression.

4 Basis Fields

A vector field may be written as a linear combination of basis vector fields. If n is the dimension, then any set of n linearly independent vector fields may be used as a basis. The coordinate basis X is an example of a basis.[1] We will see later that not every basis is a coordinate basis: in order to be a coordinate basis, there must be a coordinate system such that each basis element is the directional derivative operator in a corresponding coordinate direction.

Let e be a tuple of basis vector fields, such as the coordinate basis X. The general vector field v applied to an arbitrary manifold function f can be expressed as a linear combination

$$\mathsf{v}(\mathsf{f})(\mathsf{m}) = \mathsf{e}(\mathsf{f})(\mathsf{m})\ \mathsf{b}(\mathsf{m}) = \sum_i \mathsf{e}_i(\mathsf{f})(\mathsf{m})\, \mathsf{b}^i(\mathsf{m}), \tag{4.1}$$

where b is a tuple-valued coefficient function on the manifold. When expressed in a coordinate basis, the coefficients that specify the direction of the vector are naturally expressed as functions b^i of the coordinates of the manifold point. Here, the coefficient function b is more naturally expressed as a tuple-valued function on the manifold. If b is the coefficient function expressed as a function of coordinates, then $\mathsf{b} = b \circ \chi$ is the coefficient function as a function on the manifold.

The coordinate-basis forms have a simple definition in terms of the coordinate-basis vectors and the coordinates (equation 3.40). With this choice, the dual property, equation (3.41), holds without further fuss. More generally, we can define a basis of one-forms $\tilde{\mathsf{e}}$ that is dual to e in that the property

$$\tilde{\mathsf{e}}^i(\mathsf{e}_j)(\mathsf{m}) = \delta^i_j \tag{4.2}$$

is satisfied, analogous to property (3.41). Figure 4.1 illustrates the duality of basis fields.

[1] We cannot say if the basis vectors are orthogonal or normalized until we introduce a metric.

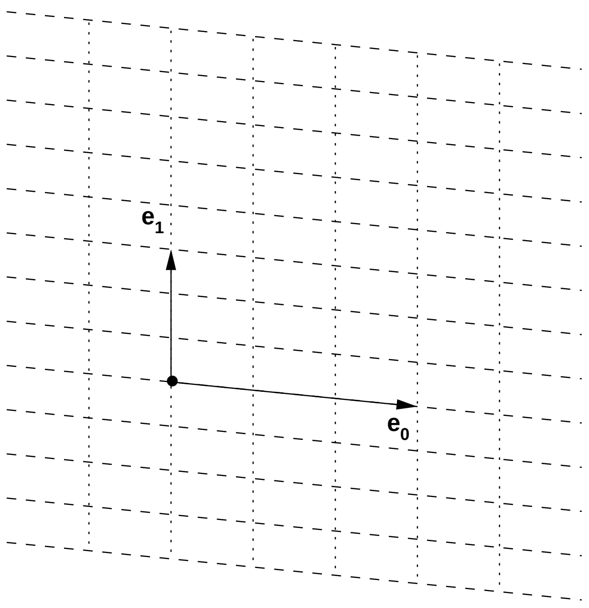

Figure 4.1 Let arrows e_0 and e_1 depict the vectors of a basis vector field at a particular point. Then the foliations shown by the parallel lines depict the dual basis one-form fields at that point. The dotted lines represent the field $\tilde{\mathsf{e}}^0$ and the dashed lines represent the field $\tilde{\mathsf{e}}^1$. The spacings of the lines are 1/3 unit. That the vectors pierce three of the lines representing their duals and do not pierce any of the lines representing the other basis elements is one way to see the relationship $\tilde{\mathsf{e}}^i(\mathsf{e}_j)(\mathsf{m}) = \delta^i_j$.

To solve for the dual basis $\tilde{\mathsf{e}}$ given the basis e, we express the basis vectors e in terms of a coordinate basis[2]

$$\mathsf{e}_j(\mathsf{f}) = \sum_k \mathsf{X}_k(\mathsf{f})\, \mathsf{c}^k_j \tag{4.3}$$

and the dual one-forms $\tilde{\mathsf{e}}$ in terms of the dual coordinate one-forms

$$\tilde{\mathsf{e}}^i(\mathsf{v}) = \sum_l \mathsf{d}^i_l\, \widetilde{\mathsf{X}}^l(\mathsf{v}), \tag{4.4}$$

[2]We write the vector components on the right and the tuple of basis vectors on the left because if we think of the basis vectors as organized as a row and the components as organized as a column then the formula is just a matrix multiplication.

then

$$\begin{aligned}\tilde{\mathsf{e}}^i(\mathsf{e}_j) &= \sum_l \mathsf{d}^i_l \widetilde{\mathsf{X}}^l(\mathsf{e}_j) \\ &= \sum_l \mathsf{d}^i_l \mathsf{e}_j(\chi^l) \\ &= \sum_l \mathsf{d}^i_l \sum_k \mathsf{X}_k(\chi^l)\mathsf{c}^k_j \\ &= \sum_{kl} \mathsf{d}^i_l \delta^l_k \mathsf{c}^k_j \\ &= \sum_k \mathsf{d}^i_k \mathsf{c}^k_j. \end{aligned} \tag{4.5}$$

Applying this at m we get

$$\tilde{\mathsf{e}}^i(\mathsf{e}_j)(\mathsf{m}) = \delta^i_j = \sum_k \mathsf{d}^i_k(\mathsf{m})\mathsf{c}^k_j(\mathsf{m}). \tag{4.6}$$

So the d coefficients can be determined from the c coefficents (essentially by matrix inversion).

A set of vector fields $\{\mathsf{e}_i\}$ may be linearly independent in the sense that a weighted sum of them may not be identically zero over a region, yet it may not be a basis in that region. The problem is that there may be some places in the region where the vectors are not independent. For example, two of the vectors may be parallel at a point but not parallel elsewhere in the region. At such a point m the determinant of the matrix $\mathsf{c}(\mathsf{m})$ is zero. So at these points we cannot define the dual basis forms.[3]

The dual form fields can be used to determine the coefficients b of a vector field v relative to a basis e, by applying the dual basis form fields $\tilde{\mathsf{e}}$ to the vector field. Let

$$\mathsf{v}(\mathsf{f}) = \sum_i \mathsf{e}_i(\mathsf{f})\,\mathsf{b}^i. \tag{4.7}$$

Then

$$\tilde{\mathsf{e}}^j(\mathsf{v}) = \mathsf{b}^j. \tag{4.8}$$

[3]This is why the set of vector fields and the set of one-form fields are modules rather than vector spaces.

Define two general vector fields:

```
(define e0
  (+ (* (literal-manifold-function 'e0x R2-rect) d/dx)
     (* (literal-manifold-function 'e0y R2-rect) d/dy)))

(define e1
  (+ (* (literal-manifold-function 'e1x R2-rect) d/dx)
     (* (literal-manifold-function 'e1y R2-rect) d/dy)))
```

We use these as a vector basis and compute the dual:

```
(define e-vector-basis (down e0 e1))
(define e-dual-basis
  (vector-basis->dual e-vector-basis R2-polar))
```

The procedure `vector-basis->dual` requires an auxiliary coordinate system (here `R2-polar`) to get the c_j^k coefficient functions from which we compute the d_k^i coefficient functions. However, the final result is independent of this coordinate system. Then we can verify that the bases e and $\tilde{\mathsf{e}}$ satisfy the dual relationship (equation 3.41) by applying the dual basis to the vector basis:

```
((e-dual-basis e-vector-basis) R2-rect-point)
(up (down 1 0) (down 0 1))
```

Note that the dual basis was computed relative to the polar coordinate system: the resulting objects are independent of the coordinates in which they were expressed!

Or we can make a general vector field with this basis and then pick out the coefficients by applying the dual basis:

```
(define v
  (* (up (literal-manifold-function 'b^0 R2-rect)
         (literal-manifold-function 'b^1 R2-rect))
     e-vector-basis))

((e-dual-basis v) R2-rect-point)
(up (b^0 (up x0 y0)) (b^1 (up x0 y0)))
```

4.1 Change of Basis

Suppose that we have a vector field v expressed in terms of one basis e and we want to reexpress it in terms of another basis e'. We have

$$\mathsf{v}(\mathsf{f}) = \sum_i \mathsf{e}_i(\mathsf{f})\mathsf{b}^i = \sum_j \mathsf{e}'_j(\mathsf{f})\mathsf{b}'^j. \tag{4.9}$$

The coefficients b' can be obtained from v by applying the dual basis

$$\mathsf{b}'^j = \tilde{\mathsf{e}}'^j(\mathsf{v}) = \sum_i \tilde{\mathsf{e}}'^j(\mathsf{e}_i)\mathsf{b}^i. \tag{4.10}$$

Let

$$\mathsf{J}^j_i = \tilde{\mathsf{e}}'^j(\mathsf{e}_i), \tag{4.11}$$

then

$$\mathsf{b}'^j = \sum_i \mathsf{J}^j_i \mathsf{b}^i, \tag{4.12}$$

and

$$\mathsf{e}_i(\mathsf{f}) = \sum_j \mathsf{e}'_j(\mathsf{f})\mathsf{J}^j_i \tag{4.13}$$

The Jacobian J is a structure of manifold functions. Using tuple arithmetic, we can write

$$\mathsf{b}' = \mathsf{J}\mathsf{b} \tag{4.14}$$

and

$$\mathsf{e}(\mathsf{f}) = \mathsf{e}'(\mathsf{f})\mathsf{J}. \tag{4.15}$$

We can write

```
(define (Jacobian to-basis from-basis)
  (s:map/r (basis->1form-basis to-basis)
           (basis->vector-basis from-basis)))
```

These are the rectangular components of a vector field:

```
(define b-rect
  ((coordinate-system->1form-basis R2-rect)
   (literal-vector-field 'b R2-rect)))
```

The polar components are:

```
(define b-polar
  (* (Jacobian (coordinate-system->basis R2-polar)
               (coordinate-system->basis R2-rect))
     b-rect))

(b-polar ((point R2-rect) (up 'x0 'y0)))
(up
 (/ (+ (* x0 (b^0 (up x0 y0))) (* y0 (b^1 (up x0 y0))))
    (sqrt (+ (expt x0 2) (expt y0 2))))
 (/ (+ (* x0 (b^1 (up x0 y0))) (* -1 y0 (b^0 (up x0 y0))))
    (+ (expt x0 2) (expt y0 2))))
```

We can also get the polar components directly:

```
(((coordinate-system->1form-basis R2-polar)
  (literal-vector-field 'b R2-rect))
 ((point R2-rect) (up 'x0 'y0)))
(up
 (/ (+ (* x0 (b^0 (up x0 y0))) (* y0 (b^1 (up x0 y0))))
    (sqrt (+ (expt x0 2) (expt y0 2))))
 (/ (+ (* x0 (b^1 (up x0 y0))) (* -1 y0 (b^0 (up x0 y0))))
    (+ (expt x0 2) (expt y0 2))))
```

We see that they are the same.

If K is the Jacobian that relates the basis vectors in the other direction

$$\mathsf{e}'(\mathsf{f}) = \mathsf{e}(\mathsf{f})\mathsf{K} \tag{4.16}$$

then

$$\mathsf{KJ} = \mathsf{I} = \mathsf{JK} \tag{4.17}$$

where I is a manifold function that returns the multiplicative identity.

The dual basis transforms oppositely. Let

$$\boldsymbol{\omega} = \sum_i \mathsf{a}_i \tilde{\mathsf{e}}^i = \sum_i \mathsf{a}'_i \tilde{\mathsf{e}}'^i \tag{4.18}$$

The coefficients are[4]

$$\mathsf{a}_i = \boldsymbol{\omega}(\mathsf{e}_i) = \sum_j \mathsf{a}'_j \tilde{\mathsf{e}}'^j(\mathsf{e}_i) = \sum_j \mathsf{a}'_j \mathsf{J}^j{}_i \tag{4.19}$$

or, in tuple arithmetic,

$$\mathsf{a} = \mathsf{a}'\mathsf{J} \tag{4.20}$$

Because of equation (4.18) we can deduce

$$\tilde{\mathsf{e}} = \mathsf{K}\tilde{\mathsf{e}}'. \tag{4.21}$$

4.2 Rotation Basis

One interesting basis for rotations in 3-dimensional space is not a coordinate basis.

Rotations are the actions of the special orthogonal group SO(3), which is a 3-dimensional manifold. The elements of this group may be represented by the set of 3×3 orthogonal matrices with determinant $+1$.

We can use a coordinate patch on this manifold with Euler angle coordinates: each element has three coordinates, θ, ϕ, ψ. A manifold point may be represented by a rotation matrix. The rotation matrix for Euler angles is a product of three simple rotations: $M(\theta, \phi, \psi) = R_z(\phi)R_x(\theta)R_z(\psi)$, where R_x and R_z are functions that take an angle and produce the matrices representing rotations about the x and z axes, respectively. We can visualize θ as the colatitude of the pole from the $\hat{z}$-axis, ϕ as the longitude, and ψ as the rotation around the pole.

Given a rotation specified by Euler angles, how do we change the Euler angle to correspond to an incremental rotation of size ϵ about the $\hat{x}$-axis? The direction (a, b, c) is constrained by the equation

$$R_x(\epsilon)M(\theta, \phi, \psi) = M(\theta + a\epsilon, \phi + b\epsilon, \psi + c\epsilon). \tag{4.22}$$

[4]We see from equations (4.15) and (4.16) that J and K are inverses. We can obtain their coefficients by: $\mathsf{J}^j_i = \tilde{\mathsf{e}}'^j(\mathsf{e}_i)$ and $\mathsf{K}^j_i = \tilde{\mathsf{e}}^j(\mathsf{e}'_i)$.

Linear equations for (a, b, c) can be found by taking the derivative of this equation with respect to ϵ. We find

$$0 = c\cos\theta + b \tag{4.23}$$
$$0 = a\sin\phi - c\cos\phi\sin\theta \tag{4.24}$$
$$1 = c\sin\phi\sin\theta + a\cos\phi, \tag{4.25}$$

with the solution

$$a = \cos\phi \tag{4.26}$$
$$b = -\frac{\sin\phi\cos\theta}{\sin\theta} \tag{4.27}$$
$$c = \frac{\sin\phi}{\sin\theta}. \tag{4.28}$$

Therefore, we can write the basis vector field that takes directional derivatives in the direction of incremental x rotations as

$$\begin{aligned}\mathsf{e}_x &= a\frac{\partial}{\partial\theta} + b\frac{\partial}{\partial\phi} + c\frac{\partial}{\partial\psi} \\ &= \cos\phi\frac{\partial}{\partial\theta} - \frac{\sin\phi\cos\theta}{\sin\theta}\frac{\partial}{\partial\phi} + \frac{\sin\phi}{\sin\theta}\frac{\partial}{\partial\psi}.\end{aligned} \tag{4.29}$$

Similarly, vector fields for the incremental y and z rotations are

$$\mathsf{e}_y = \frac{\cos\phi\cos\theta}{\sin\theta}\frac{\partial}{\partial\phi} + \sin\phi\frac{\partial}{\partial\theta} - \frac{\cos\phi}{\sin\theta}\frac{\partial}{\partial\psi} \tag{4.30}$$
$$\mathsf{e}_z = \frac{\partial}{\partial\phi} \tag{4.31}$$

4.3 Commutators

The commutator of two vector fields is defined as

$$[\mathsf{v}, \mathsf{w}](\mathsf{f}) = \mathsf{v}(\mathsf{w}(\mathsf{f})) - \mathsf{w}(\mathsf{v}(\mathsf{f})). \tag{4.32}$$

In the special case that the two vector fields are coordinate basis fields, the commutator is zero:

$$\begin{aligned}[\mathsf{X}_i, \mathsf{X}_j](\mathsf{f}) &= \mathsf{X}_i(\mathsf{X}_j(\mathsf{f})) - \mathsf{X}_j(\mathsf{X}_i(\mathsf{f})) \\ &= \partial_i\partial_j(\mathsf{f}\circ\chi^{-1})\circ\chi - \partial_j\partial_i(\mathsf{f}\circ\chi^{-1})\circ\chi \\ &= 0,\end{aligned} \tag{4.33}$$

because the individual partial derivatives commute. The vanishing commutator is telling us that we get to the same manifold point by integrating from a point along first one basis vector field and then another as from integrating in the other order. If the commutator is zero we can use the integral curves of the basis vector fields to form a coordinate mesh.

More generally, the commutator of two vector fields is a vector field. Let v be a vector field with coefficient function $\mathsf{c} = c \circ \chi$, and u be a vector field with coefficient function $\mathsf{b} = b \circ \chi$, both with respect to the coordinate basis X. Then

$$\begin{aligned}
[\mathsf{u},\mathsf{v}](\mathsf{f}) &= \mathsf{u}(\mathsf{v}(\mathsf{f})) - \mathsf{v}(\mathsf{u}(\mathsf{f})) \\
&= \mathsf{u}(\sum_i \mathsf{X}_i(\mathsf{f})\mathsf{c}^i) - \mathsf{v}(\sum_j \mathsf{X}_j(\mathsf{f})\mathsf{b}^j) \\
&= \sum_j \mathsf{X}_j(\sum_i \mathsf{X}_i(\mathsf{f})\mathsf{c}^i)\mathsf{b}^j - \sum_i \mathsf{X}_i(\sum_j \mathsf{X}_j(\mathsf{f})\mathsf{b}^j)\mathsf{c}^i \\
&= \sum_{ij} [\mathsf{X}_j, \mathsf{X}_i](\mathsf{f})\mathsf{c}^i\mathsf{b}^j \\
&\quad + \sum_i \mathsf{X}_i(\mathsf{f}) \sum_j (\mathsf{X}_j(\mathsf{c}^i)\mathsf{b}^j - \mathsf{X}_j(\mathsf{b}^i)\mathsf{c}^j) \\
&= \sum_i \mathsf{X}_i(\mathsf{f})\mathsf{a}^i, \qquad (4.34)
\end{aligned}$$

where the coefficient function a of the commutator vector field is

$$\begin{aligned}
\mathsf{a}^i &= \sum_j \left(\mathsf{X}_j(\mathsf{c}^i)\mathsf{b}^j - \mathsf{X}_j(\mathsf{b}^i)\mathsf{c}^j\right) \\
&= \mathsf{u}(\mathsf{c}^i) - \mathsf{v}(\mathsf{b}^i). \qquad (4.35)
\end{aligned}$$

We used the fact, shown above, that the commutator of two coordinate basis fields is zero.

We can check this formula for the commutator for the general vector fields `e0` and `e1` in polar coordinates:

```
(let* ((polar-basis (coordinate-system->basis R2-polar))
       (polar-vector-basis (basis->vector-basis polar-basis))
       (polar-dual-basis (basis->1form-basis polar-basis))
       (f (literal-manifold-function 'f-rect R2-rect)))
  ((- ((commutator e0 e1) f)
      (* (- (e0 (polar-dual-basis e1))
            (e1 (polar-dual-basis e0)))
         (polar-vector-basis f)))
   R2-rect-point))
0
```

Let e be a tuple of basis vector fields. The commutator of two basis fields can be expressed in terms of the basis vector fields:

$$[\mathsf{e}_i, \mathsf{e}_j](\mathsf{f}) = \sum_k \mathsf{d}^k_{ij} \mathsf{e}_k(\mathsf{f}), \tag{4.36}$$

where d^k_{ij} are functions of m, called the *structure constants* for the basis vector fields. The coefficients are

$$\mathsf{d}^k_{ij} = \tilde{\mathsf{e}}^k([\mathsf{e}_i, \mathsf{e}_j]). \tag{4.37}$$

The commutator $[\mathsf{u}, \mathsf{v}]$ with respect to a non-coordinate basis e_i is

$$[\mathsf{u}, \mathsf{v}](\mathsf{f}) = \sum_k \mathsf{e}_k(\mathsf{f}) \left(\mathsf{u}(\mathsf{c}^k) - \mathsf{v}(\mathsf{b}^k) + \sum_{ij} \mathsf{c}^i \mathsf{b}^j \mathsf{d}^k_{ji} \right). \tag{4.38}$$

Define the vector fields Jx, Jy, and Jz that generate rotations about the three rectangular axes in three dimensions:[5]

```
(define Jz (- (* x d/dy) (* y d/dx)))
(define Jx (- (* y d/dz) (* z d/dy)))
(define Jy (- (* z d/dx) (* x d/dz)))
```

[5]Using

```
(define R3-rect (coordinate-system-at 'rectangular 'origin R3))
(define-coordinates (up x y z) R3-rect)
(define R3-rect-point ((point R3-rect) (up 'x0 'y0 'z0)))
(define g (literal-manifold-function 'g-rect R3-rect))
```

```
(((+ (commutator Jx Jy) Jz) g) R3-rect-point)
0
(((+ (commutator Jy Jz) Jx) g) R3-rect-point)
0
(((+ (commutator Jz Jx) Jy) g) R3-rect-point)
0
```

We see that

$$\begin{aligned}
[\mathsf{J}_x, \mathsf{J}_y] &= -\mathsf{J}_z \\
[\mathsf{J}_y, \mathsf{J}_z] &= -\mathsf{J}_x \\
[\mathsf{J}_z, \mathsf{J}_x] &= -\mathsf{J}_y. \qquad (4.39)
\end{aligned}$$

We can also compute the commutators for the basis vector fields e_x, e_y, and e_z in the SO(3) manifold (see equations 4.29–4.31) that correspond to rotations about the x, y, and z axes, respectively:[6]

```
(((+ (commutator e_x e_y) e_z) f) SO3-point)
0
(((+ (commutator e_y e_z) e_x) f) SO3-point)
0
(((+ (commutator e_z e_x) e_y) f) SO3-point)
0
```

You can tell if a set of basis vector fields is a coordinate basis by calculating the commutators. If they are nonzero, then the basis is not a coordinate basis. If they are zero then the basis vector fields can be integrated to give the coordinate system.

Recall equation (3.31)

$$(e^{t\mathsf{v}}\mathsf{f})(\mathsf{m}) = (\mathsf{f} \circ \phi_t^{\mathsf{v}})(\mathsf{m}). \qquad (4.40)$$

Iterating this equation, we find

$$(e^{s\mathsf{w}}e^{t\mathsf{v}}\mathsf{f})(\mathsf{m}) = (\mathsf{f} \circ \phi_t^{\mathsf{v}} \circ \phi_s^{\mathsf{w}})(\mathsf{m}). \qquad (4.41)$$

[6]Using

```
(define Euler-angles (coordinate-system-at 'Euler 'Euler-patch SO3))
(define Euler-angles-chi-inverse (point Euler-angles))
(define-coordinates (up theta phi psi) Euler-angles)
(define SO3-point ((point Euler-angles) (up 'theta 'phi 'psi)))
(define f (literal-manifold-function 'f-Euler Euler-angles))
```

Notice that the evolution under w occurs before the evolution under v.

To illustrate the meaning of the commutator, consider the evolution around a small loop with sides made from the integral curves of two vector fields v and w. We will first follow v, then w, then $-\mathsf{v}$, and then $-\mathsf{w}$:

$$(e^{\epsilon \mathsf{v}} e^{\epsilon \mathsf{w}} e^{-\epsilon \mathsf{v}} e^{-\epsilon \mathsf{w}} \mathsf{f})(\mathsf{m}). \tag{4.42}$$

To second order in ϵ the result is[7]

$$(e^{\epsilon^2 [\mathsf{v},\mathsf{w}]} \mathsf{f})(\mathsf{m}). \tag{4.43}$$

This result is illustrated in Figure 4.2.

Take a point $\mathsf{0}$ in M as the origin. Then, presuming $[\mathsf{e}_i, \mathsf{e}_j] = 0$, the coordinates x of the point m in the coordinate system corresponding to the e basis satisfy[8]

$$\mathsf{m} = \phi_1^{x\mathsf{e}}(\mathsf{0}) = \chi^{-1}(x), \tag{4.44}$$

where χ is the coordinate function being defined. Because the elements of e commute, we can translate separately along the integral curves in any order and reach the same point; the terms in the exponential can be factored into separate exponentials if needed.

Exercise 4.1: Alternate Angles

Note that the Euler angles are singular at $\theta = 0$ (where ϕ and ψ become degenerate), so the representations of e_x, e_y, and e_z (defined in equa-

[7] For non-commuting operators A and B,

$$\begin{aligned} &e^A e^B e^{-A} e^{-B} \\ &\quad = \left(1 + A + \frac{A^2}{2} + \cdots\right)\left(1 + B + \frac{B^2}{2} + \cdots\right) \\ &\qquad\quad \times \left(1 - A + \frac{A^2}{2} + \cdots\right)\left(1 - B + \frac{B^2}{2} + \cdots\right) \\ &\quad = 1 + [A, B] + \cdots, \end{aligned}$$

to second order in A and B. All higher-order terms can be written in terms of higher-order commutators of A and B. An example of a higher-order commutator is $[A, [A, B]]$.

[8] Here x is an up-tuple structure of components, and e is down-tuple structure of basis vectors. The product of the two contracts to make a scaled vector, along which we translate by one unit.

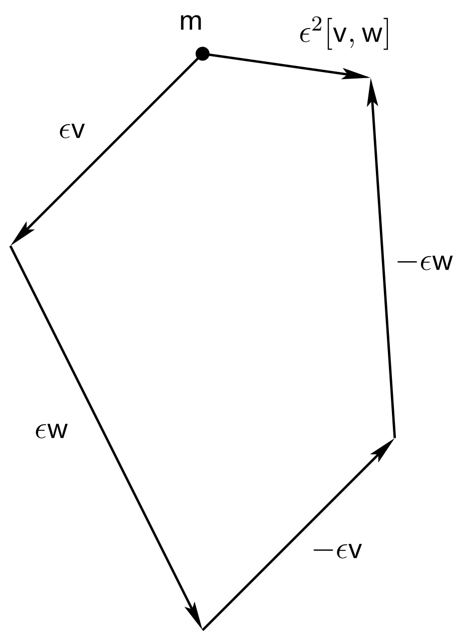

Figure 4.2 The commutator of two vector fields computes the residual of a small loop following their integral curves.

tions 4.29–4.31) have problems there. An alternate coordinate system avoids this problem, while introducing a similar problem elsewhere in the manifold.

Consider the "alternate angles" $(\theta_a, \phi_a, \psi_a)$ which define a rotation matrix via $M(\theta_a, \phi_a, \psi_a) = R_z(\phi_a) R_x(\theta_a) R_y(\psi_a)$.

a. Where does the singularity appear in these alternate coordinates? Do you think you could define a coordinate system for rotations that has no singularities?

b. What do the e_x, e_y, and e_z basis vector fields look like in this coordinate system?

Exercise 4.2: General Commutators

Verify equation (4.38).

Exercise 4.3: SO(3) Basis and Angular Momentum Basis

How are J_x, J_y, and J_z related to e_x, e_y, and e_z in Equations (4.29–4.31)?

5 Integration

We know how to integrate real-valued functions of a real variable. We want to extend this idea to manifolds, in such a way that the integral is independent of the coordinate systems used to compute it.

The integral of a real-valued function of a real variable is the limit of a sum of products of the values of the function on subintervals and the lengths of the increments of the independent variable in those subintervals:

$$\int_a^b f = \int_a^b f(x)\,dx = \lim_{\Delta x_i \to 0} \sum_i f\left(x_i\right) \Delta x_i. \tag{5.1}$$

If we change variables ($x = g(y)$), then the form of the integral changes:

$$\begin{aligned}\int_a^b f &= \int_a^b f(x)\,dx \\ &= \int_{g^{-1}(a)}^{g^{-1}(b)} f(g(y)) Dg(y) dy \\ &= \int_{g^{-1}(a)}^{g^{-1}(b)} (f \circ g) Dg.\end{aligned} \tag{5.2}$$

We can make a coordinate-independent notion of integration in the following way. An interval of the real line is a 1-dimensional manifold with boundary. We can assign a coordinate chart χ to this manifold. Let $x = \chi(\mathsf{m})$. The coordinate basis is associated with a coordinate-basis vector field, here $\partial/\partial\mathsf{x}$. Let $\boldsymbol{\omega}$ be a one-form on this manifold. The application of $\boldsymbol{\omega}$ to $\partial/\partial\mathsf{x}$ is a real-valued function on the manifold. If we compose this with the inverse chart, we get a real-valued function of a real variable. We can then write the usual integral of this function

$$I = \int_a^b \boldsymbol{\omega}(\partial/\partial\mathsf{x}) \circ \chi^{-1}. \tag{5.3}$$

It turns out that the value of this integral is independent of the coordinate chart used in its definition. Consider a different coordinate chart $x' = \chi'(\mathsf{m})$, with associated basis vector field $\partial/\partial\mathsf{x}'$. Let $g = \chi' \circ \chi^{-1}$ We have

$$\begin{aligned}
&\int_{a'}^{b'} \boldsymbol{\omega}\left(\partial/\partial\mathsf{x}'\right) \circ \chi'^{-1} \\
&\quad = \int_{a'}^{b'} \boldsymbol{\omega}\left(\partial/\partial\mathsf{x}\left(D\left(\chi \circ \chi'^{-1}\right) \circ \chi'\right)\right) \circ \chi'^{-1} \\
&\quad = \int_{a'}^{b'} \left(\boldsymbol{\omega}(\partial/\partial\mathsf{x}) D\left(\chi \circ \chi'^{-1}\right) \circ \chi'\right) \circ \chi'^{-1} \\
&\quad = \int_{a'}^{b'} \left(\boldsymbol{\omega}(\partial/\partial\mathsf{x}) \circ \chi'^{-1}\right) D\left(\chi \circ \chi'^{-1}\right) \\
&\quad = \int_{a}^{b} \left(\left(\left(\boldsymbol{\omega}(\partial/\partial\mathsf{x}) \circ \chi^{-1}\right) D\left(\chi \circ \chi'^{-1}\right)\right) \circ g\right) Dg \\
&\quad = \int_{a}^{b} \boldsymbol{\omega}(\partial/\partial\mathsf{x}) \circ \chi^{-1}, \qquad (5.4)
\end{aligned}$$

where we have used the rule for coordinate transformations of basis vectors (equation 3.19), linearity of forms in the first two lines, and the rule for change-of-variables under an integral in the last line.[1]

Because the integral is independent of coordinate chart, we can write simply

$$I = \int_{\mathsf{M}} \boldsymbol{\omega}, \qquad (5.5)$$

where M is the 1-dimensional manifold with boundary corresponding to the interval.

We are exploiting the fact that coordinate basis vectors in different coordinate systems are related by a Jacobian (see equation 3.19), which cancels the Jacobian that appears in the change-of-variables formula for integration (see equation 5.2).

[1] Note $(D\left(\chi \circ \chi'^{-1}\right) \circ (\chi' \circ \chi^{-1}))\, D(\chi' \circ \chi^{-1}) = 1$. With $g = \chi' \circ \chi^{-1}$ this is $(D(g^{-1}) \circ g)\,(Dg) = 1$.

5.1 Higher Dimensions

We have seen that we can integrate one-forms on 1-dimensional manifolds. We need higher-rank forms that we can integrate on higher-dimensional manifolds in a coordinate-independent manner.

Consider the integral of a real-valued function, $\mathsf{f} : \mathsf{R}^n \to \mathsf{R}$, over a region U in R^n. Under a coordinate transformation $g : \mathsf{R}^n \to \mathsf{R}^n$, we have[2]

$$\int_{\mathsf{U}} \mathsf{f} = \int_{g^{-1}(\mathsf{U})} (\mathsf{f} \circ g) \det (Dg) . \tag{5.6}$$

A rank n form field takes n vector field arguments and produces a real-valued manifold function: $\boldsymbol{\omega}(\mathsf{v}, \mathsf{w}, \ldots, \mathsf{u})(\mathsf{m})$. By analogy with the 1-dimensional case, higher-rank forms are linear in each argument. Higher-rank forms must also be antisymmetric under interchange of any two arguments in order to make a coordinate-free definition of integration analogous to equation (5.3).

Consider an integral in the coordinate system χ

$$\int_{\chi(\mathsf{U})} \boldsymbol{\omega}(\mathsf{X}_0, \mathsf{X}_1, \ldots) \circ \chi^{-1}. \tag{5.7}$$

Under coordinate transformations $g = \chi \circ \chi'^{-1}$, the integral becomes

$$\int_{\chi'(\mathsf{U})} \boldsymbol{\omega}(\mathsf{X}_0, \mathsf{X}_1, \ldots) \circ \chi'^{-1} \; \det (Dg) . \tag{5.8}$$

Using the change-of-basis formula, equation (3.19):

$$\mathsf{X}(\mathsf{f}) = \mathsf{X}'(\mathsf{f})(D(\chi' \circ \chi^{-1})) \circ \chi = \mathsf{X}'(\mathsf{f}) \left(D\left(g^{-1}\right)\right) \circ \chi. \tag{5.9}$$

If we let $M = (D\,(g^{-1})) \circ \chi$ then

$$\begin{aligned}
&(\boldsymbol{\omega}(\mathsf{X}_0, \mathsf{X}_1, \ldots) \circ \chi'^{-1}) \; \det (Dg) \\
&\qquad = (\boldsymbol{\omega}(\mathsf{X}' M_0, \mathsf{X}' M_1, \ldots) \circ \chi'^{-1}) \; \det (Dg) \\
&\qquad = (\boldsymbol{\omega}(\mathsf{X}'_0, \mathsf{X}'_1, \ldots) \circ \chi'^{-1}) \, \alpha (M_0, M_1, \ldots) \; \det (Dg) ,
\end{aligned} \tag{5.10}$$

[2]The *determinant* is the unique function of the rows of its argument that i) is linear in each row, ii) changes sign under any interchange of rows, and iii) is one when applied to the identity multiplier.

using the multilinearity of $\boldsymbol{\omega}$, where M_i is the i^{th} column of M. The function α is multilinear in the columns of M. To make a coordinate-independent integration we want the expression (5.10) to be the same as the integrand in

$$I' = \int_{\chi'(\mathsf{U})} \boldsymbol{\omega}\left(\mathsf{X}'_0, \mathsf{X}'_1, \ldots\right) \circ \chi'^{-1}. \tag{5.11}$$

For this to be the case, $\alpha\left(M_0, M_1, \ldots\right)$ must be $(\det(Dg))^{-1} = \det(M)$. So α is an antisymmetric function, and thus so is $\boldsymbol{\omega}$.

Thus higher-rank form fields must be antisymmetric multilinear functions from vector fields to manifold functions. So we have a coordinate-independent definition of integration of form fields on a manifold and we can write

$$I = I' = \int_{\mathsf{U}} \boldsymbol{\omega}. \tag{5.12}$$

Wedge Product

There are several ways we can construct antisymmetric higher-rank forms. Given two one-form fields $\boldsymbol{\omega}$ and $\boldsymbol{\tau}$ we can form a two-form field $\boldsymbol{\omega} \wedge \boldsymbol{\tau}$ as follows:

$$(\boldsymbol{\omega} \wedge \boldsymbol{\tau})(\mathsf{v}, \mathsf{w}) = \boldsymbol{\omega}(\mathsf{v})\boldsymbol{\tau}(\mathsf{w}) - \boldsymbol{\omega}(\mathsf{w})\boldsymbol{\tau}(\mathsf{v}). \tag{5.13}$$

More generally we can form the wedge of higher-rank forms. Let $\boldsymbol{\omega}$ be a k-form field and $\boldsymbol{\tau}$ be an l-form field. We can form a $(k+l)$-form field $\omega \wedge \tau$ as follows:

$$\boldsymbol{\omega} \wedge \boldsymbol{\tau} = \frac{(k+l)!}{k!\, l!} \mathrm{Alt}(\boldsymbol{\omega} \otimes \boldsymbol{\tau}) \tag{5.14}$$

where, if η is a function on m vectors,

$$\begin{aligned} &\mathrm{Alt}(\eta)(\mathsf{v}_0, \ldots, \mathsf{v}_{m-1}) \\ &\quad = \frac{1}{m!} \sum_{\sigma \in \mathrm{Perm}(m)} \mathrm{Parity}(\sigma)\eta(\mathsf{v}_{\sigma(0)}, \ldots, \mathsf{v}_{\sigma(m-1)}) \end{aligned} \tag{5.15}$$

and where

$$\begin{aligned} &\boldsymbol{\omega} \otimes \boldsymbol{\tau}(\mathsf{v}_0, \ldots, \mathsf{v}_{k-1}, \mathsf{v}_k, \ldots, \mathsf{v}_{k+l-1}) \\ &\quad = \boldsymbol{\omega}(\mathsf{v}_0, \ldots, \mathsf{v}_{k-1})\boldsymbol{\tau}(\mathsf{v}_k, \ldots, \mathsf{v}_{k+l-1}). \end{aligned} \tag{5.16}$$

The wedge product is associative, and thus we need not specify the order of a multiple application. The factorial coefficients of these formulas are chosen so that

$$(\mathsf{dx} \wedge \mathsf{dy} \wedge \ldots)\,(\partial/\partial\mathsf{x}, \partial/\partial\mathsf{y}, \ldots) = 1. \tag{5.17}$$

This is true independent of the coordinate system.

Equation (5.17) gives us

$$\int_{\mathsf{U}} \mathsf{dx} \wedge \mathsf{dy} \wedge \ldots = \text{Volume}(\mathsf{U}) \tag{5.18}$$

where Volume(U) is the ordinary volume of the region corresponding to U in the Euclidean space of R^n with the orthonormal coordinate system $(x, y, \ldots)$.[3]

An example two-form (see figure 5.1) is the oriented area of a parallelogram in the (x, y) coordinate plane at the point m spanned by two vectors $\mathsf{u} = \mathsf{u}^0\partial/\partial\mathsf{x} + \mathsf{u}^1\partial/\partial\mathsf{y}$ and $\mathsf{v} = \mathsf{v}^0\partial/\partial\mathsf{x} + \mathsf{v}^1\partial/\partial\mathsf{y}$, which is given by

$$\mathsf{A}\,(\mathsf{u}, \mathsf{v})\,(\mathsf{m}) = \mathsf{u}^0\,(\mathsf{m})\,\mathsf{v}^1\,(\mathsf{m}) - \mathsf{v}^0\,(\mathsf{m})\,\mathsf{u}^1\,(\mathsf{m})\,. \tag{5.19}$$

Note that this is the area of the parallelogram in the coordinate plane, which is the range of the coordinate function. It is not the area on the manifold. To define that we need more structure—the metric. We will put a metric on the manifold in Chapter 9.

3-dimensional Euclidean Space

Let's specialize to 3-dimensional Euclidean space. Following equation (5.18) we can write the coordinate-area two-form in another way: $\mathsf{A} = \mathsf{dx} \wedge \mathsf{dy}$. As code:

[3]By using the word "orthonormal" here we are assuming that the range of the coordinate chart is an ordinary Euclidean space with the usual Euclidean metric. The coordinate basis in that chart is orthonormal. Under these conditions we can usefully use words like "length," "area," and "volume" in the coordinate space.

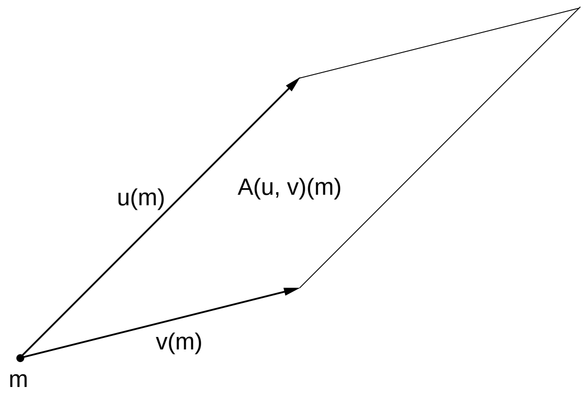

Figure 5.1 The area of the parallelogram in the (x, y) coordinate plane is given by $A(\mathsf{u}, \mathsf{v})(\mathsf{m})$.

```
(define-coordinates (up x y z) R3-rect)

(define u (+ (* 'u^0 d/dx) (* 'u^1 d/dy)))
(define v (+ (* 'v^0 d/dx) (* 'v^1 d/dy)))

(((wedge dx dy) u v) R3-rect-point)
(+ (* u^0 v^1) (* -1 u^1 v^0))
```

If we use cylindrical coordinates and define cylindrical vector fields we get the analogous answer in cylindrical coordinates:

```
(define-coordinates (up r theta z) R3-cyl)

(define a (+ (* 'a^0 d/dr) (* 'a^1 d/dtheta)))
(define b (+ (* 'b^0 d/dr) (* 'b^1 d/dtheta)))

(((wedge dr dtheta) a b) ((point R3-cyl) (up 'r0 'theta0 'z0)))
(+ (* a^0 b^1) (* -1 a^1 b^0))
```

The moral of this story is that this is the area of the parallelogram in the coordinate plane. It is not the area on the manifold!

There is a similar story with volumes. The wedge product of the elements of the coordinate basis is a three-form that measures our usual idea of coordinate volumes in R^3 with a Euclidean metric:

```
(define u (+ (* 'u^0 d/dx) (* 'u^1 d/dy) (* 'u^2 d/dz)))
(define v (+ (* 'v^0 d/dx) (* 'v^1 d/dy) (* 'v^2 d/dz)))
(define w (+ (* 'w^0 d/dx) (* 'w^1 d/dy) (* 'w^2 d/dz)))

(((wedge dx dy dz) u v w) R3-rect-point)
(+ (* u^0 v^1 w^2)
   (* -1 u^0 v^2 w^1)
   (* -1 u^1 v^0 w^2)
   (* u^1 v^2 w^0)
   (* u^2 v^0 w^1)
   (* -1 u^2 v^1 w^0))
```

This last expression is the determinant of a 3×3 matrix:

```
(- (((wedge dx dy dz) u v w) R3-rect-point)
   (determinant
    (matrix-by-rows (list 'u^0 'u^1 'u^2)
                    (list 'v^0 'v^1 'v^2)
                    (list 'w^0 'w^1 'w^2))))
0
```

If we did the same operations in cylindrical coordinates we would get the analogous formula, showing that what we are computing is volume in the coordinate space, not volume on the manifold.

Because of antisymmetry, if the rank of a form is greater than the dimension of the manifold then the form is identically zero. The k-forms on an n-dimensional manifold form a module of dimension $\binom{n}{k}$. We can write a coordinate-basis expression for a k-form as

$$\boldsymbol{\omega} = \sum_{i_0,\ldots,i_{k-1}=0}^{n} \omega_{i_0,\ldots,i_{k-1}} \mathsf{dx}^{i_0} \wedge \ldots \wedge \mathsf{dx}^{\mathsf{i}_{k-1}}. \tag{5.20}$$

The antisymmetry of the wedge product implies that

$$\omega_{i_{\sigma(0)},\ldots,i_{\sigma(k-1)}} = \text{Parity}(\sigma)\omega_{i_0,\ldots,i_{k-1}}, \tag{5.21}$$

from which we see that there are only $\binom{n}{k}$ independent components of $\boldsymbol{\omega}$.

Exercise 5.1: Wedge Product

Pick a coordinate system and use the computer to verify that

a. the wedge product is associative for forms in your coordinate system;

b. formula (5.17) is true in your coordinate system.

5.2 Exterior Derivative

The intention of introducing the exterior derivative is to capture all of the classical theorems of "vector analysis" into one unified Stokes's Theorem, which asserts that the integral of a form on the boundary of a manifold is the integral of the exterior derivative of the form on the interior of the manifold:[4]

$$\int_{\partial \mathsf{M}} \boldsymbol{\omega} = \int_{\mathsf{M}} \mathsf{d}\boldsymbol{\omega}. \tag{5.22}$$

As we have seen in equation (3.34), the differential of a function on a manifold is a one-form field. If a function on a manifold is considered to be a form field of rank zero,[5] then the differential operator increases the rank of the form by one. We can generalize this to k-form fields with the exterior derivative operation.

Consider a one-form $\boldsymbol{\omega}$. We define[6]

$$\mathsf{d}\boldsymbol{\omega}(\mathsf{v}_1, \mathsf{v}_2) = \mathsf{v}_1(\boldsymbol{\omega}(\mathsf{v}_2)) - \mathsf{v}_2(\boldsymbol{\omega}(\mathsf{v}_1)) - \boldsymbol{\omega}([\mathsf{v}_1, \mathsf{v}_2]). \tag{5.23}$$

More generally, the exterior derivative of a k-form field is a $k+1$-form field, given by:[7]

$$\begin{aligned} &\mathsf{d}\boldsymbol{\omega}(\mathsf{v}_0, \ldots, \mathsf{v}_k) = \\ &\quad \sum_{i=0}^{k} \Big\{ ((-1)^i \mathsf{v}_i(\boldsymbol{\omega}(\mathsf{v}_0, \ldots, \mathsf{v}_{i-1}, \mathsf{v}_{i+1}, \ldots, \mathsf{v}_k)) + \\ &\quad \sum_{j=i+1}^{k} (-1)^{i+j}\, \boldsymbol{\omega}([\mathsf{v}_i, \mathsf{v}_j], \mathsf{v}_0, \ldots, \mathsf{v}_{i-1}, \mathsf{v}_{i+1}, \ldots, \mathsf{v}_{j-1}, \mathsf{v}_{j+1}, \ldots, \mathsf{v}_k)) \Big\}. \end{aligned} \tag{5.24}$$

This formula is coordinate-system independent. This is the way we compute the exterior derivative in our software.

[4]This is a generalization of the Fundamental Theorem of Calculus.

[5]A manifold function f induces a form field $\hat{\mathsf{f}}$ of rank 0 as follows:

$\hat{\mathsf{f}}()(\mathsf{m}) = \mathsf{f}(\mathsf{m})$.

[6]The definition is chosen to make Stokes's Theorem pretty.

[7]See Spivak, *Differential Geometry*, Volume 1, p.289.

If the form field $\boldsymbol{\omega}$ is represented in a coordinate basis

$$\boldsymbol{\omega} = \sum_{i_0=0,\ldots,i_{k-1}=0}^{n-1} \mathsf{a}_{i_0,\ldots,i_{k-1}} \mathsf{dx}^{i_0} \wedge \cdots \wedge \mathsf{dx}^{i_{k-1}} \tag{5.25}$$

then the exterior derivative can be expressed as

$$\mathsf{d}\boldsymbol{\omega} = \sum_{i_0=0,\ldots,i_{k-1}=0}^{n-1} \mathsf{da}_{i_0,\ldots,i_{k-1}} \wedge \mathsf{dx}^{i_0} \wedge \cdots \wedge \mathsf{dx}^{i_{k-1}}. \tag{5.26}$$

Though this formula is expressed in terms of a coordinate basis, the result is independent of the choice of coordinate system.

We can test that the computation indicated by equation (5.24) is equivalent to the computation indicated by equation (5.26) in three dimensions with a general one-form field:

```
(define a (literal-manifold-function 'alpha R3-rect))
(define b (literal-manifold-function 'beta R3-rect))
(define c (literal-manifold-function 'gamma R3-rect))

(define theta (+ (* a dx) (* b dy) (* c dz)))
```

The test will require two arbitrary vector fields

```
(define X (literal-vector-field 'X-rect R3-rect))
(define Y (literal-vector-field 'Y-rect R3-rect))

(((- (d theta)
     (+ (wedge (d a) dx)
        (wedge (d b) dy)
        (wedge (d c) dz)))
  X Y)
 R3-rect-point)
0
```

We can also try a general two-form field in 3-dimensional space: Let

$$\boldsymbol{\omega} = a\mathsf{dy} \wedge \mathsf{dz} + b\mathsf{dz} \wedge \mathsf{dx} + c\mathsf{dx} \wedge \mathsf{dy}, \tag{5.27}$$

where $a = \alpha \circ \chi$, $b = \beta \circ \chi$, $c = \gamma \circ \chi$, and α, β, and γ are real-valued functions of three real arguments. As a program,

```
(define omega
  (+ (* a (wedge dy dz))
     (* b (wedge dz dx))
     (* c (wedge dx dy))))
```

Here we need another vector field because our result will be a three-form field.

```
(define Z (literal-vector-field 'Z-rect R3-rect))

(((- (d omega)
     (+ (wedge (d a) dy dz)
        (wedge (d b) dz dx)
        (wedge (d c) dx dy)))
  X Y Z)
 R3-rect-point)
0
```

The exterior derivative of the wedge of two form fields obeys the graded Leibniz rule. It can be written in terms of the exterior derivatives of the component form fields:

$$\mathsf{d}(\boldsymbol{\omega} \wedge \boldsymbol{\tau}) = \mathsf{d}\boldsymbol{\omega} \wedge \boldsymbol{\tau} + (-1)^k \boldsymbol{\omega} \wedge \mathsf{d}\boldsymbol{\tau}, \tag{5.28}$$

where k is the rank of $\boldsymbol{\omega}$.

A form field $\boldsymbol{\omega}$ that is the exterior derivative of another form field $\boldsymbol{\omega} = \mathsf{d}\boldsymbol{\theta}$ is called *exact*. A form field whose exterior derivative is zero is called *closed*.

Every exact form field is a closed form field: applying the exterior derivative operator twice always yields zero:

$$\mathsf{d}^2\boldsymbol{\omega} = \mathbf{0}. \tag{5.29}$$

This is equivalent to the statement that partial derivatives with respect to different variables commute.[8]

It is easy to show equation (5.29) for manifold functions:

$$\begin{aligned}\mathsf{d}^2\mathsf{f}(\mathsf{u},\mathsf{v}) &= \mathsf{d}(\mathsf{df})(\mathsf{u},\mathsf{v})\\ &= \mathsf{u}(\mathsf{df}(\mathsf{v})) - \mathsf{v}(\mathsf{df}(\mathsf{u})) - \mathsf{df}([\mathsf{u},\mathsf{v}])\\ &= \mathsf{u}(\mathsf{v}(\mathsf{f})) - \mathsf{v}(\mathsf{u}(\mathsf{f})) - [\mathsf{u},\mathsf{v}](\mathsf{f})\\ &= 0\end{aligned} \tag{5.30}$$

[8]See Spivak, *Calculus on Manifolds*, p.92

Consider the general one-form field $\boldsymbol{\theta}$ defined on 3-dimensional rectangular space. Taking two exterior derivatives of $\boldsymbol{\theta}$ yields a three-form field. It is zero:

```
(((d (d theta)) X Y Z) R3-rect-point)
0
```

Not every closed form field is an exact form field. Whether a closed form field is exact depends on the topology of a manifold.

5.3 Stokes's Theorem

The proof of the general Stokes's Theorem for n-dimensional orientable manifolds is quite complicated, but it is easy to see how it works for a 2-dimensional region M that can be covered with a single coordinate patch.[9]

Given a coordinate chart $\chi(\mathsf{m}) = (\mathsf{x}(\mathsf{m}), \mathsf{y}(\mathsf{m}))$ we can obtain a pair of coordinate-basis vectors $\partial/\partial\mathsf{x} = X_0$ and $\partial/\partial\mathsf{y} = X_1$.

The coordinate image of M can be divided into small rectangular areas in the (x, y) coordinate plane. The union of the rectangular areas gives the coordinate image of M. The clockwise integrals around the boundaries of the rectangles cancel on neighboring rectangles, because the boundary is traversed in opposite directions. But on the boundary of the coordinate image of M the boundary integrals do not cancel, yielding an integral on the boundary of M. Area integrals over the rectangular areas add to produce an integral over the entire coordinate image of M.

So, consider Stokes's theorem on a small patch P of the manifold for which the coordinates form a rectangular region ($x_{min} < x < x_{max}$ and $y_{min} < y < y_{max}$). Stokes's theorem on P states

$$\int_{\partial\mathsf{P}} \boldsymbol{\omega} = \int_{\mathsf{P}} \mathsf{d}\boldsymbol{\omega}. \tag{5.31}$$

The area integral on the right can be written as an ordinary multidimensional integral using the coordinate basis vectors (remember that the value of the integral is independent of the choice of coor-

[9]We do not develop the machinery for integration on chains that is usually needed for a full proof of Stokes's Theorem. This is adequately done in other books. A beautiful treatment can be found in Spivak, *Calculus on Manifolds* [16].

dinates):

$$\int_{\chi(\mathsf{P})} \mathsf{d}\boldsymbol{\omega}\,(\partial/\partial\mathsf{x}, \partial/\partial\mathsf{y}) \circ \chi^{-1} \\ = \int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} (\partial/\partial\mathsf{x}(\boldsymbol{\omega}(\partial/\partial\mathsf{y})) - \partial/\partial\mathsf{y}(\boldsymbol{\omega}(\partial/\partial\mathsf{x}))) \circ \chi^{-1}. \tag{5.32}$$

We have used equation (5.23) to expand the exterior derivative.

Consider just the first term of the right-hand side of equation (5.32). Then using the definition of basis vector field $\partial/\partial\mathsf{x}$ we obtain

$$\int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} \left(\partial/\partial\mathsf{x}(\boldsymbol{\omega}(\partial/\partial\mathsf{y})) \circ \chi^{-1}\right) \\ = \int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} \left(X_0(\boldsymbol{\omega}(\partial/\partial\mathsf{y})) \circ \chi^{-1}\right) \\ = \int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} \partial_0 \left((\boldsymbol{\omega}(\partial/\partial\mathsf{y})) \circ \chi^{-1}\right). \tag{5.33}$$

This integral can now be evaluated using the Fundamental Theorem of Calculus. Accumulating the results for both integrals

$$\int_{\chi(\mathsf{P})} \mathsf{d}\boldsymbol{\omega}\,(\partial/\partial\mathsf{x}, \partial/\partial\mathsf{y}) \circ \chi^{-1} \\ = \int_{x_{min}}^{x_{max}} \left((\boldsymbol{\omega}(\partial/\partial\mathsf{x})) \circ \chi^{-1}\right)(x, y_{min})dx \\ \int_{y_{min}}^{y_{max}} \left((\boldsymbol{\omega}(\partial/\partial\mathsf{y})) \circ \chi^{-1}\right)(x_{max}, y)dy \\ - \int_{x_{min}}^{x_{max}} \left((\boldsymbol{\omega}(\partial/\partial\mathsf{x})) \circ \chi^{-1}\right)(x, y_{max})dx \\ - \int_{y_{min}}^{y_{max}} \left((\boldsymbol{\omega}(\partial/\partial\mathsf{y})) \circ \chi^{-1}\right)(x_{min}, y)dy \\ = \int_{\partial\mathsf{P}} \boldsymbol{\omega}, \tag{5.34}$$

as was to be shown.

5.4 Vector Integral Theorems

Green's Theorem states that for an arbitrary compact set $M \subset \mathbf{R}^2$, a 2-dimensional Euclidean space:

$$\int_{\partial M} ((\alpha \circ \chi)\, \mathsf{dx} + (\beta \circ \chi)\, \mathsf{dy}) = \int_M ((\partial_0 \beta - \partial_1 \alpha) \circ \chi)\, \mathsf{dx} \wedge \mathsf{dy}. \tag{5.35}$$

We can test this. By Stokes's theorem, the integrands are related by an exterior derivative. We need some vectors to test our forms:

```
(define v (literal-vector-field 'v-rect R2-rect))
(define w (literal-vector-field 'w-rect R2-rect))
```

We can now test our integrands:[10]

```
(define alpha (literal-function 'alpha R2->R))
(define beta (literal-function 'beta R2->R))

(let ((dx (ref (basis->1form-basis R2-rect-basis) 0))
      (dy (ref (basis->1form-basis R2-rect-basis) 1)))
  (((- (d (+ (* (compose alpha (chart R2-rect)) dx)
             (* (compose beta (chart R2-rect)) dy)))
       (* (compose (- ((partial 0) beta)
                      ((partial 1) alpha))
                   (chart R2-rect))
          (wedge dx dy)))
    v w)
   R2-rect-point))
0
```

We can also compute the integrands for the Divergence Theorem: For an arbitrary compact set $M \subset \mathbf{R}^3$ and a vector field w

$$\int_M \text{div}(\mathsf{w})\, dV = \int_{\partial M} \mathsf{w} \cdot \mathsf{n}\, dA \tag{5.36}$$

where n is the outward-pointing normal to the surface ∂M. Again, the integrands should be related by an exterior derivative, if this is an instance of Stokes's Theorem.

Note that even the statement of this theorem cannot be made with the machinery we have developed at this point. The con-

[10] Using `(define R2-rect-basis (coordinate-system->basis R2-rect))`.
Here we extract `dx` and `dy` from `R2-rect-basis` to avoid globally installing coordinates.

cepts "outward-pointing normal," area A, and volume V on the manifold are not definable without using a metric (see chapter 9). However, for orthonormal rectangular coordinates in $\mathbf{R}^3$ we can interpret the integands in terms of forms.

Let the vector field describing the flow of stuff be

$$\mathsf{w} = \mathsf{a}\frac{\partial}{\partial \mathsf{x}} + \mathsf{b}\frac{\partial}{\partial \mathsf{y}} + \mathsf{c}\frac{\partial}{\partial \mathsf{z}} \tag{5.37}$$

The rate of leakage of stuff through each element of the boundary is $\mathsf{w} \cdot \mathsf{n}\, dA$. We interpret this as the two-form

$$\mathsf{a}\, \mathsf{dy} \wedge \mathsf{dz} + \mathsf{b}\, \mathsf{dz} \wedge \mathsf{dx} + \mathsf{c}\, \mathsf{dx} \wedge \mathsf{dy}, \tag{5.38}$$

because any part of the boundary will have y-z, z-x, and x-y components, and each such component will pick up contributions from the normal component of the flux w. Formalizing this as code we have

```
(define a (literal-manifold-function 'a-rect R3-rect))
(define b (literal-manifold-function 'b-rect R3-rect))
(define c (literal-manifold-function 'c-rect R3-rect))

(define flux-through-boundary-element
  (+ (* a (wedge dy dz))
     (* b (wedge dz dx))
     (* c (wedge dx dy))))
```

The rate of production of stuff in each element of volume is $\operatorname{div}(\mathsf{w})\, dV$. We interpret this as the three-form

$$\left(\frac{\partial}{\partial \mathsf{x}}\mathsf{a} + \frac{\partial}{\partial \mathsf{y}}\mathsf{b} + \frac{\partial}{\partial \mathsf{z}}\mathsf{c}\right) \mathsf{dx} \wedge \mathsf{dy} \wedge \mathsf{dz}. \tag{5.39}$$

or:

```
(define production-in-volume-element
  (* (+ (d/dx a) (d/dy b) (d/dz c))
     (wedge dx dy dz)))
```

Assuming Stokes's theorem, the exterior derivative of the leakage of stuff per unit area through the boundary must be the rate of production of stuff per unit volume in the interior. We check this by applying the difference to arbitrary vector fields at an arbitrary point:

```
(define X (literal-vector-field 'X-rect R3-rect))
(define Y (literal-vector-field 'Y-rect R3-rect))
(define Z (literal-vector-field 'Z-rect R3-rect))

(((- production-in-volume-element
     (d flux-through-boundary-element))
  X Y Z)
 R3-rect-point)
0
```

as expected.

Exercise 5.2: Graded Formula

Derive equation (5.28).

Exercise 5.3: Iterated Exterior Derivative

We have shown that the equation (5.29) is true for manifold functions. Show that it is true for any form field.

6 Over a Map

To deal with motion on manifolds we need to think about paths on manifolds and vectors along these paths. Tangent vectors along paths are not vector fields on the manifold because they are defined only on the path. And the path may even cross itself, which would give more than one vector at a point. Here we introduce the concept of a *vector field over a map*.[1] A vector field over a map assigns a vector to each image point of the map. In general the map may be a function from one manifold to another. If the domain of the map is the manifold of the real line, the range of the map is a 1-dimensional path on the target manifold. One possible way to define a vector field over a map is to assign a tangent vector to each image point of a path, allowing us to work with tangent vectors to paths. A *one-form field over the map* allows us to extract the components of a vector field over the map.

6.1 Vector Fields Over a Map

Let μ be a map from points n in the manifold N to points m in the manifold M. A vector over the map μ takes directional derivatives of functions on M at points $\mathsf{m} = \mu(\mathsf{n})$. The vector over the map applied to the function on M is a function on N.

Restricted Vector Fields

One way to make a vector field over a map is to restrict a vector field on M to the image of N over μ, as illustrated in Figure 6.1.

Let v be a vector field on M, and f a function on M. Then

$$\mathsf{v}_\mu(\mathsf{f}) = \mathsf{v}(\mathsf{f}) \circ \mu, \tag{6.1}$$

is a vector over the map μ. Note that $\mathsf{v}_\mu(\mathsf{f})$ is a function on N, not M:

$$\mathsf{v}_\mu(\mathsf{f})(\mathsf{n}) = \mathsf{v}(\mathsf{f})(\mu(\mathsf{n})). \tag{6.2}$$

[1]See Bishop and Goldberg, *Tensor Analysis on Manifolds* [2].

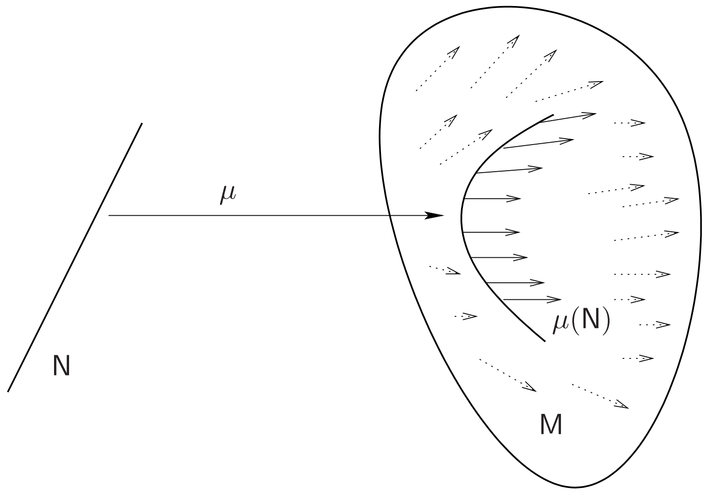

Figure 6.1 The vector field v on M is indicated by arrows. The solid arrows are v_μ, the restricted vector field over the map μ. The vector field over the map is restricted to the image of N in M.

We can implement this definition as:

```
(define ((vector-field->vector-field-over-map mu:N->M) v-on-M)
  (procedure->vector-field
   (lambda (f-on-M)
     (compose (v-on-M f-on-M) mu:N->M))))
```

Differential of a Map

Another way to construct a vector field over a map μ is to transport a vector field from the source manifold N to the target manifold M with the *differential* of the map

$$d\mu(\mathsf{v})(\mathsf{f})(\mathsf{n}) = \mathsf{v}(\mathsf{f} \circ \mu)(\mathsf{n}), \tag{6.3}$$

which takes its argument in the source manifold N. The differential of a map μ applied to a vector field v on N is a vector field over the map. A procedure to compute the differential is:

```
(define (((differential mu) v) f)
  (v (compose f mu)))
```

The nomenclature of this subject is confused. The "differential of a map between manifolds," $d\mu$, takes one more argument than the "differential of a real-valued function on a manifold," df, but when the target manifold of μ is the reals and I is the identity function on the reals,

$$d\mu(\mathsf{v})(I)(\mathsf{n}) = (\mathsf{v}(I \circ \mu))(\mathsf{n}) = (\mathsf{v}(\mu))(\mathsf{n}) = \mathsf{d}\mu(\mathsf{v})(\mathsf{n}). \tag{6.4}$$

We avoid this problem in our notation by distinguishing d and d. In our programs we encode d as `differential` and d as `d`.

Velocity at a Time

Let μ be the map from the time line to the manifold M, and $\partial/\partial\mathsf{t}$ be a basis vector on the time line. Then $d\mu(\partial/\partial\mathsf{t})$ is the vector over the map μ that computes the rate of change of functions on M along the path that is the image of μ. This is the velocity vector. We can use the differential to assign a velocity vector to each moment, solving the problem of multiple vectors at a point if the path crosses itself.

6.2 One-Form Fields Over a Map

Given a one-form $\boldsymbol{\omega}$ on the manifold M, the one-form over the map $\mu : \mathsf{N} \to \mathsf{M}$ is constructed as follows:

$$\boldsymbol{\omega}^{\mu}(\mathsf{v}_{\mu})(\mathsf{n}) = \boldsymbol{\omega}(\mathsf{u})(\mu(\mathsf{n})), \text{ where } \mathsf{u}(\mathsf{f})(\mathsf{m}) = \mathsf{v}_{\mu}(\mathsf{f})(\mathsf{n}). \tag{6.5}$$

The object u is not really a vector field on M even though we have given it that shape so that the dual vector can apply to it; $\mathsf{u}(\mathsf{f})$ is evaluated only at images $\mathsf{m} = \mu(\mathsf{n})$ of points n in N. If we were defining u as a vector field we would need the inverse of μ to find the point $\mathsf{n} = \mu^{-1}(\mathsf{m})$, but this is not required to define the object u in a context where there is already an m associated with the n of interest. To extend this idea to k-forms, we carry each vector argument over the map.

The procedure that constructs a k-form over the map from a k-form is:

```
(define ((form-field->form-field-over-map mu:N->M) w-on-M)
  (define (make-fake-vector-field V-over-mu n)
    (define ((u f) m)
      ((V-over-mu f) n))
    (procedure->vector-field u))
  (procedure->nform-field
   (lambda vectors-over-map
     (lambda (n)
       ((apply w-on-M
               (map (lambda (V-over-mu)
                      (make-fake-vector-field V-over-mu n))
                    vectors-over-map))
        (mu:N->M n))))
   (get-rank w-on-M)))
```

The internal procedure `make-fake-vector-field` counterfeits a vector field u on M from the vector field over the map $\mu : \mathsf{N} \to \mathsf{M}$. This works here because the only value that is ever passed as `m` is `(mu:N->M n)`.

6.3 Basis Fields Over a Map

Let e be a tuple of basis vector fields, and $\tilde{\mathsf{e}}$ be the tuple of basis one-forms that is dual to e:

$$\tilde{\mathsf{e}}^i(\mathsf{e}_j)(\mathsf{m}) = \delta^i_j. \tag{6.6}$$

The *basis vectors over the map*, e^μ, are particular cases of vectors over a map:

$$\mathsf{e}^\mu(\mathsf{f}) = \mathsf{e}(\mathsf{f}) \circ \mu. \tag{6.7}$$

And the elements of the *dual basis over the map*, $\tilde{\mathsf{e}}_\mu$, are particular cases of one-forms over the map. The basis and dual basis over the map satisfy

$$\tilde{\mathsf{e}}^i_\mu(\mathsf{e}^\mu_j)(\mathsf{n}) = \delta^i_j. \tag{6.8}$$

Walking on a Sphere

For example, let μ map the time line to the unit sphere.[2] We use colatitude θ and longitude ϕ as coordinates on the sphere:

```
(define S2 (make-manifold S^2 2 3))
(define S2-spherical
  (coordinate-system-at 'spherical 'north-pole S2))
(define-coordinates (up theta phi) S2-spherical)
(define S2-basis (coordinate-system->basis S2-spherical))
```

A general path on the sphere is:[3]

```
(define mu
  (compose (point S2-spherical)
           (up (literal-function 'theta)
               (literal-function 'phi))
           (chart R1-rect)))
```

The basis over the map is constructed from the basis on the sphere:

```
(define S2-basis-over-mu
  (basis->basis-over-map mu S2-basis))

(define h
  (literal-manifold-function 'h-spherical S2-spherical))

(((basis->vector-basis S2-basis-over-mu) h)
 ((point R1-rect) 't0))
(down
 (((partial 0) h-spherical) (up (theta t0) (phi t0)))
 (((partial 1) h-spherical) (up (theta t0) (phi t0))))
```

The basis vectors over the map compute derivatives of the function h evaluated on the path at the given time.

We can check that the dual basis over the map does the correct thing:

[2]We execute `(define-coordinates t R1-rect)` to make `t` the coordinate function of the real line.

[3]We provide a shortcut to make literal manifold maps:

```
(define mu (literal-manifold-map 'mu R1-rect S2-spherical))
```

But if we used this shortcut, the component functions would be named `mu^0` and `mu^1`. Here we wanted to use more mnemonic names for the component functions.

```
(((basis->1form-basis S2-basis-over-mu)
  (basis->vector-basis S2-basis-over-mu))
 ((point R1-rect) 't0))
(up (down 1 0) (down 0 1))
```

Components of the Velocity
Let χ be a tuple of coordinates on M, with associated basis vectors X_i, and dual basis elements dx^i. The vector basis and dual basis over the map μ are X_i^μ and dx_μ^i. The components of the velocity (rates of change of coordinates along the path μ) are obtained by applying the dual basis over the map to the velocity

$$v^i(t) = \mathsf{dx}_\mu^i(d\mu(\partial/\partial\mathsf{t}))(\mathsf{t}), \tag{6.9}$$

where t is the coordinate for the point t.

For example, the coordinate velocities on a sphere are

```
(((basis->1form-basis S2-basis-over-mu)
  ((differential mu) d/dt))
 ((point R1-rect) 't0))
(up ((D theta) t0) ((D phi) t0)))
```

as expected.

6.4 Pullbacks and Pushforwards

Maps from one manifold to another can also be used to relate the vector fields and one-form fields on one manifold to those on the other. We have introduced two such relations: restricted vector fields and the differential of a function. However, there are other ways to relate the vector fields and form fields on different manifolds that are connected by a map.

Pullback and Pushforward of a Function

The *pullback* of a function f on M over the map μ is defined as

$$\mu^*\mathsf{f} = \mathsf{f} \circ \mu. \tag{6.10}$$

This allows us to take a function defined on M and use it to define a new function on N.

For example, the integral curve of v evolved for time t as a function of the initial manifold point m generates a map ϕ_t^{v} of the manifold onto itself. This is a simple currying[4] of the integral curve of v from m as a function of time: $\phi_t^{\mathsf{v}}(\mathsf{m}) = \gamma_{\mathsf{m}}^{\mathsf{v}}(t)$. The evolution of the function f along an integral curve, equation (3.33), can be written in terms of the pullback over ϕ_t^{v}:

$$(\mathsf{E}_{t,\mathsf{v}}\mathsf{f})(\mathsf{m}) = \mathsf{f}(\phi_t^{\mathsf{v}}(\mathsf{m})) = ((\phi_t^{\mathsf{v}})^*\mathsf{f})(\mathsf{m}). \tag{6.11}$$

This is implemented as:

```
(define ((pullback-function mu:N->M) f-on-M)
  (compose f-on-M mu:N->M))
```

A vector field over the map that was constructed by restriction (equation 6.1) can be seen as the pullback of the function constructed by application of the vector field to a function:

$$\mathsf{v}_\mu(\mathsf{f}) = \mathsf{v}(\mathsf{f}) \circ \mu = \mu^*(\mathsf{v}(\mathsf{f})). \tag{6.12}$$

A vector field over the map that was constructed by a differential (equation 6.3) can be seen as the vector field applied to the pullback of the function:

$$d\mu(\mathsf{v})(\mathsf{f})(\mathsf{n}) = \mathsf{v}(\mathsf{f} \circ \mu)(\mathsf{n}) = \mathsf{v}(\mu^*\mathsf{f})(\mathsf{n}). \tag{6.13}$$

If we have an inverse for the map μ we can also define a *pushforward* of the function g, defined on the source manifold of the map:[5]

$$\mu_*\mathsf{g} = \mathsf{g} \circ \mu^{-1}. \tag{6.14}$$

Pushforward of a Vector Field

We can also define the *pushforward* of a vector field over the map μ. The pushforward takes a vector field v defined on N. The result

[4]A function of two arguments may be seen as a function of one argument whose value is a function of the other argument. This can be done in two different ways, depending on which argument is supplied first. The general process of specifying a subset of the arguments to produce a new function of the others is called *currying* the function, in honor of the logician Haskell Curry (1900–1982) who, with Moses Schönfinkel (1889–1942), developed combinatory logic.

[5]Notation note: superscript asterisk indicates pullback, subscript asterisk indicates pushforward. Pullbacks and pushforwards are tightly binding operators, so, for example $\mu^* f(\mathsf{n}) = (\mu^* f)(\mathsf{n})$.

takes directional derivatives of functions on M at a place determined by a point in M:

$$\mu_*\mathsf{v}(\mathsf{f})(\mathsf{m}) = \mathsf{v}(\mu^*\mathsf{f})(\mu^{-1}(\mathsf{m})) = \mathsf{v}(\mathsf{f} \circ \mu)(\mu^{-1}(\mathsf{m})), \tag{6.15}$$

or

$$\mu_*\mathsf{v}(\mathsf{f}) = \mu_*(\mathsf{v}(\mu^*\mathsf{f})). \tag{6.16}$$

Here we expressed the pushforward of the vector field in terms of pullbacks and pushforwards of functions. Note that the pushforward requires the inverse of the map.

If the map is from time to some configuration manifold and represents the time evolution of a process, we can think of the pushforward of a vector field as a velocity measured at a point on the trajectory in the configuration manifold. By contrast, the differential of the map applied to the vector field gives us the velocity vector at each moment in time. Because a trajectory may cross itself, the pushforward is not defined at any point where the crossing occurs, but the differential is always defined.

Pushforward Along Integral Curves

We can push a vector field forward over the map generated by an integral curve of a vector field w, because the inverse is always available.[6]

$$((\phi_t^{\mathsf{w}})_*\mathsf{v})(\mathsf{f})(\mathsf{m}) = \mathsf{v}((\phi_t^{\mathsf{w}})^*\mathsf{f})(\phi_{-t}^{\mathsf{w}}(\mathsf{m})) = \mathsf{v}(\mathsf{f} \circ \phi_t^{\mathsf{w}})(\phi_{-t}^{\mathsf{w}}(\mathsf{m})). \tag{6.17}$$

This is implemented as:

```
(define ((pushforward-vector mu:N->M mu^-1:M->N) v-on-N)
  (procedure->vector-field
   (lambda (f)
     (compose (v-on-N (compose f mu:N->M)) mu^-1:M->N))))
```

[6]The map ϕ_t^{w} is always invertible: $(\phi_t^{\mathsf{w}})^{-1} = \phi_{-t}^{\mathsf{w}}$ because of the uniqueness of the solutions of the initial-value problem for ordinary differential equations.

Pullback of a Vector Field

Given a vector field v on manifold M we can pull the vector field back through the map $\mu : \mathsf{N} \to \mathsf{M}$ as follows:

$$\mu^* \mathsf{v}(\mathsf{f})(\mathsf{n}) = (\mathsf{v}(\mathsf{f} \circ \mu^{-1}))(\mu(\mathsf{n})) \tag{6.18}$$

or

$$\mu^* \mathsf{v}(\mathsf{f}) = \mu^*(\mathsf{v}(\mu_* \mathsf{f})). \tag{6.19}$$

This may be useful when the map is invertible, as in the flow generated by a vector field.

This is implemented as:

```
(define (pullback-vector-field mu:N->M mu^-1:M->N)
  (pushforward-vector mu^-1:M->N mu:N->M))
```

Pullback of a Form Field

We can also pull back a one-form field $\boldsymbol{\omega}$ defined on M, but an honest definition is rarely written. The pullback of a one-form field applied to a vector field is intended to be the same as the one-form field applied to the pushforward of the vector field.

The pullback of a one-form field is often described by the relation

$$\mu^* \boldsymbol{\omega}(\mathsf{v}) = \boldsymbol{\omega}(\mu_* \mathsf{v}), \tag{6.20}$$

but this is wrong, because the two sides are not functions of points in the same manifold. The one-form field $\boldsymbol{\omega}$ applies to a vector field on the manifold M, which takes a directional derivative of a function defined on M and is evaluated at a point on M, but the left-hand side is evaluated at a point on the manifold N.

A more precise description would be

$$\mu^* \boldsymbol{\omega}(\mathsf{v})(\mathsf{n}) = \boldsymbol{\omega}(\mu_* \mathsf{v})(\mu(\mathsf{n})) \tag{6.21}$$

or

$$\mu^* \boldsymbol{\omega}(\mathsf{v}) = \mu^*(\boldsymbol{\omega}(\mu_* \mathsf{v})). \tag{6.22}$$

Although this is accurate, it may not be effective, because computing the pushforward requires the inverse of the map μ. But the inverse is available when the map is the flow generated by a vector field.

In fact it is possible to compute the pullback of a one-form field without having the inverse of the map. Instead we can use `form-field->form-field-over-map` to avoid needing the inverse:

$$\mu^* \boldsymbol{\omega}(\mathsf{v})(\mathsf{n}) = \boldsymbol{\omega}^\mu(d\mu(\mathsf{v}))(\mathsf{n}). \tag{6.23}$$

The pullback of a k-form generalizes equation 6.21:

$$\mu^* \boldsymbol{\omega}(\mathsf{u}, \mathsf{v}, \ldots)(\mathsf{n}) = \boldsymbol{\omega}(\mu_* \mathsf{u}, \mu_* \mathsf{v}, \ldots)(\mu(\mathsf{n})). \tag{6.24}$$

This is implemented as follows:[7]

```
(define ((pullback-form mu:N->M) omega-on-M)
  (let ((k (get-rank omega-on-M)))
    (if (= k 0)
        ((pullback-function mu:N->M) omega-on-M)
        (procedure->nform-field
         (lambda vectors-on-N
           (apply ((form-field->form-field-over-map mu:N->M)
                   omega-on-M)
                  (map (differential mu:N->M) vectors-on-N)))
         k))))
```

Properties of Pullback

The pullback through a map has many nice properties: it distributes through addition and through wedge product:

$$\mu^*(\boldsymbol{\theta} + \boldsymbol{\phi}) = \mu^* \boldsymbol{\theta} + \mu^* \boldsymbol{\phi} \tag{6.25}$$

$$\mu^*(\boldsymbol{\theta} \wedge \boldsymbol{\phi}) = \mu^* \boldsymbol{\theta} \wedge \mu^* \boldsymbol{\phi} \tag{6.26}$$

The pullback also commutes with the exterior derivative:

$$\mathsf{d}(\mu^* \boldsymbol{\theta}) = \mu^*(\mathsf{d}\boldsymbol{\theta}), \tag{6.27}$$

for $\boldsymbol{\theta}$ a function or k-form field.

We can verify this by computing an example. Let μ map the rectangular plane to rectangular 3-space:

[7] There is a generic `pullback` procedure that operates on any kind of manifold object. However, to pull a vector field back requires providing the inverse map.

```
(define mu (literal-manifold-map 'MU R2-rect R3-rect))
```

First, let's compare the pullback of the exterior derivative of a function with the exterior derivative of the pullback of the function:

```
(define f (literal-manifold-function 'f-rect R3-rect))
(define X (literal-vector-field 'X-rect R2-rect))

(((- ((pullback mu) (d f)) (d ((pullback mu) f))) X)
 ((point R2-rect) (up 'x0 'y0)))
0
```

More generally, we can consider what happens to a form field. For a one-form field the result is as expected:

```
(define theta (literal-1form-field 'THETA R3-rect))
(define Y (literal-vector-field 'Y-rect R2-rect))

(((- ((pullback mu) (d theta)) (d ((pullback mu) theta))) X Y)
 ((point R2-rect) (up 'x0 'y0)))
0
```

Pushforward of a Form Field

By symmetry, it is possible to define the pushforward of a one-form field as

$$\mu_* \boldsymbol{\omega}(\mathsf{v}) = \mu_*(\boldsymbol{\omega}(\mu^* v)), \tag{6.28}$$

but this is rarely useful.

Exercise 6.1: Velocities on a Globe

We can use manifold functions, vector fields, and one-forms over a map to understand how paths behave.

a. Suppose that a vehicle is traveling east on the Earth at a given rate of change of longitude. What is the actual ground speed of the vehicle?

b. Stereographic projection is useful for navigation because it is conformal (it preserves angles). For the situation of part **a**, what is the speed measured on a stereographic map? Remember that the stereographic projection is implemented with `S2-Riemann`.

7 Directional Derivatives

The vector field was a generalization of the directional derivative to functions on a manifold. When we want to generalize the directional derivative idea to operate on other manifold objects, such as directional derivatives of vector fields or of form fields there are several useful choices. In the same way that a vector field applies to a function to produce a function, we will build directional derivatives so that when applied to any object it will produce another object of the same kind. All directional derivatives require a vector field to give the direction and scale factor.

We will have a choice of directional derivative operators that give different results for the rate of change of vector and form fields along integral curves. But all directional derivative operators must agree when computing rates of change of functions along integral curves. When applied to functions, all directional derivative operators give:

$$\mathcal{D}_{\mathsf{v}}(\mathsf{f}) = \mathsf{v}(\mathsf{f}). \tag{7.1}$$

Next we specify the directional derivative of a vector field u with respect to a vector field v. Let an integral curve of the vector field v be γ, parameterized by t, and let $\mathsf{m} = \gamma(t)$. Let u' be a vector field that results from transporting the vector field u along γ for a parameter increment δ. How u is transported to make u' determines the type of derivative. We formulate the method of transport by:

$$\mathsf{u}' = F_\delta^{\mathsf{v}}\mathsf{u}. \tag{7.2}$$

We can assume without loss of generality that $F_\delta^{\mathsf{v}}\mathsf{u}$ is a linear transformation over the reals on u, because we care about its behavior only in an incremental region around $\delta = 0$.

Let g be the comparison of the original vector field at a point with the transported vector field at that point:

$$g(\delta) = \mathsf{u}(\mathsf{f})(\mathsf{m}) - \left(F_\delta^{\mathsf{v}}\mathsf{u}\right)(\mathsf{f})(\mathsf{m}). \tag{7.3}$$

So we can compute the directional derivative operator using only ordinary derivatives:

$$\mathcal{D}_{\mathsf{v}}\mathsf{u}(\mathsf{f})(\mathsf{m}) = Dg(0). \tag{7.4}$$

The result $\mathcal{D}_{\mathsf{v}}\mathsf{u}$ is of type vector field.

The general pattern of constructing a directional derivative operator from a transport operator is given by the following schema:[1]

```
(define (((((F->directional-derivative F) v) u) f) m)
  (define (g delta)
    (- ((u f) m) (((((F v) delta) u) f) m)))
  ((D g) 0))
```

The linearity of tranport implies that

$$\mathcal{D}_{\mathsf{v}}(\alpha\mathsf{O} + \beta\mathsf{P}) = \alpha\mathcal{D}_{\mathsf{v}}\mathsf{O} + \beta\mathcal{D}_{\mathsf{v}}\mathsf{P}, \tag{7.5}$$

for any real α and β and manifold objects O and P.

The directional derivative obeys superposition in its vector-field argument:

$$\mathcal{D}_{\mathsf{v}+\mathsf{w}} = \mathcal{D}_{\mathsf{v}} + \mathcal{D}_{\mathsf{w}}. \tag{7.6}$$

The directional derivative is homogeneous over the reals in its vector-field argument:

$$\mathcal{D}_{\alpha\mathsf{v}} = \alpha\mathcal{D}_{\mathsf{v}}, \tag{7.7}$$

for any real α.[2] This follows from the fact that for evolution along integral curves: when α is a real number,

$$\phi_t^{\alpha\mathsf{v}}(\mathsf{m}) = \phi_{\alpha t}^{\mathsf{v}}(\mathsf{m}). \tag{7.8}$$

When applied to products of functions, directional derivative operators satisfy Leibniz's rule:

$$\mathcal{D}_{\mathsf{v}}(\mathsf{fg}) = \mathsf{f}\,(\mathcal{D}_{\mathsf{v}}\mathsf{g}) + (\mathcal{D}_{\mathsf{v}}\mathsf{f})\,\mathsf{g}. \tag{7.9}$$

[1]The directional derivative of a vector field must itself be a vector field. Thus the real program for this must make the function of f into a vector field. However, we leave out this detail here to make the structure clear.

[2]For some derivative operators α can be a real-valued manifold function.

The Leibniz rule is extended to applications of one-form fields to vector fields:

$$\mathcal{D}_{\mathsf{v}}(\boldsymbol{\omega}(\mathsf{y})) = \boldsymbol{\omega}\,(\mathcal{D}_{\mathsf{v}}\mathsf{y}) + (\mathcal{D}_{\mathsf{v}}\boldsymbol{\omega})\,(\mathsf{y}). \tag{7.10}$$

The extension of the Leibniz rule, combined with the choice of transport of a vector field, determines the action of the directional derivative on form fields.[3]

7.1 Lie Derivative

The Lie derivative is one kind of directional derivative operator. We write the Lie derivative operator with respect to a vector field v as $\mathcal{L}_{\mathsf{v}}$.

Functions

The Lie derivative of the function f with respect to the vector field v is given by

$$\mathcal{L}_{\mathsf{v}}\mathsf{f} = \mathsf{v}(\mathsf{f}). \tag{7.11}$$

The tangent vector v measures the rate of change of f along integral curves.

Vector Fields

For the Lie derivative of a vector field y with respect to a vector field v we choose the transport operator $F^{\mathsf{v}}_{\delta}\mathsf{y}$ to be the pushforward of y along the integral curves of v. Recall equation (6.15). So the Lie derivative of y with respect to v at the point m is

$$(\mathcal{L}_{\mathsf{v}}\mathsf{y})\,(\mathsf{f})(\mathsf{m}) = Dg(0), \tag{7.12}$$

where

$$g(\delta) = \mathsf{y}(\mathsf{f})(\mathsf{m}) - ((\phi^{\mathsf{v}}_{\delta})_{*}\,\mathsf{y})(\mathsf{f})(\mathsf{m}). \tag{7.13}$$

We can construct a procedure that computes the Lie derivative of a vector field by supplying an appropriate transport operator

[3]The action on functions, vector fields, and one-form fields suffices to define the action on all tensor fields. See Appendix C.

`(F-Lie phi)` for `F` in our schema `F->directional-derivative`. In this first stab at the Lie derivative, we introduce a coordinate system and we expand the integral curve to a given order. Because in the schema we evaluate the derivative of g at 0, the dependence on the order and the coordinate system disappears. They will not be needed in the final version.

```
(define (Lie-directional coordsys order)
  (let ((Phi (phi coordsys order)))
    (F->directional-derivative (F-Lie Phi))))

(define (((F-Lie phi) v) delta)
  (pushforward-vector ((phi v) delta) ((phi v) (- delta))))

(define ((((phi coordsys order) v) delta) m)
  ((point coordsys)
   (series:sum (((exp (* delta v)) (chart coordsys)) m)
               order)))
```

Expand the quantities in equation (7.13) to first order in δ:

$$\begin{aligned}
g(\delta) &= \mathsf{y}(\mathsf{f})(\mathsf{m}) - (\phi^{\mathsf{v}}_{\delta *}\mathsf{y})(\mathsf{f})(\mathsf{m}) \\
&= \mathsf{y}(\mathsf{f})(\mathsf{m}) - \mathsf{y}(\mathsf{f}\circ\phi^{\mathsf{v}}_{\delta})(\phi^{\mathsf{v}}_{-\delta}(\mathsf{m})) \\
&= (\mathsf{y}(\mathsf{f}) - \mathsf{y}(\mathsf{f}+\delta\mathsf{v}(\mathsf{f})+\cdots) + \delta\mathsf{v}(\mathsf{y}(\mathsf{f}+\delta\mathsf{v}(\mathsf{f})+\cdots)))(\mathsf{m}) + \cdots \\
&= (-\delta\mathsf{y}(\mathsf{v}(\mathsf{f})) + \delta\mathsf{v}(\mathsf{y}(\mathsf{f})))(\mathsf{m}) + \cdots \\
&= \delta\left[\mathsf{v},\mathsf{y}\right](\mathsf{f})(\mathsf{m}) + \mathcal{O}\left(\delta^2\right). \qquad (7.14)
\end{aligned}$$

So the Lie derivative of a vector field y with respect to a vector field v is a vector field that is defined by its behavior when applied to an arbitrary manifold function f:

$$\left(\mathcal{L}_{\mathsf{v}}\mathsf{y}\right)(\mathsf{f}) = \left[\mathsf{v},\mathsf{y}\right](\mathsf{f}) \qquad (7.15)$$

Verifying this computation

```
(let ((v (literal-vector-field 'v-rect R3-rect))
      (w (literal-vector-field 'w-rect R3-rect))
      (f (literal-manifold-function 'f-rect R3-rect)))
  ((- ((((Lie-directional R3-rect 2) v) w) f)
      ((commutator v w) f))
   ((point R3-rect) (up 'x0 'y0 'z0))))
0
```

Although this is tested to second order, evaluating the derivative at zero ensures that first order is enough. So we can safely define:

```
(define ((Lie-derivative-vector V) Y)
  (commutator V Y))
```

We can think of the Lie derivative as the rate of change of the manifold function $\mathsf{y}(\mathsf{f})$ as we move in the v direction, adjusted to take into account that some of the variation is due to the variation of f:

$$\begin{aligned}(\mathcal{L}_\mathsf{v}\mathsf{y})(\mathsf{f}) &= [\mathsf{v},\mathsf{y}](\mathsf{f})\\ &= \mathsf{v}(\mathsf{y}(\mathsf{f})) - \mathsf{y}(\mathsf{v}(\mathsf{f}))\\ &= \mathsf{v}(\mathsf{y}(\mathsf{f})) - \mathsf{y}\left(\mathcal{L}_\mathsf{v}(\mathsf{f})\right). \end{aligned} \tag{7.16}$$

The first term in the commutator, $\mathsf{v}(\mathsf{y}(\mathsf{f}))$, measures the rate of change of the combination $\mathsf{y}(\mathsf{f})$ along the integral curves of v. The change in $\mathsf{y}(\mathsf{f})$ is due to both the intrinsic change in y along the curve and the change in f along the curve; the second term in the commutator subtracts this latter quantity. The result is the intrinsic change in y along the integral curves of v.

Additionally, we can extend the product rule, for any manifold function g and any vector field u:

$$\begin{aligned}\mathcal{L}_\mathsf{v}(\mathsf{gu})(\mathsf{f}) &= [\mathsf{v},\mathsf{gu}](\mathsf{f})\\ &= \mathsf{v}(\mathsf{g})\mathsf{u}(\mathsf{f}) + \mathsf{g}[\mathsf{v},\mathsf{u}](\mathsf{f})\\ &= (\mathcal{L}_\mathsf{v}\mathsf{g})\mathsf{u}(\mathsf{f}) + \mathsf{g}(\mathcal{L}_\mathsf{v}\mathsf{u})(\mathsf{f}). \end{aligned} \tag{7.17}$$

An Alternate View

We can write the vector field

$$\mathsf{y}(\mathsf{f}) = \sum_i y^i\, \mathsf{e}_i(\mathsf{f}). \tag{7.18}$$

By the extended product rule (equation 7.17) we get

$$\mathcal{L}_\mathsf{v}\mathsf{y}(\mathsf{f}) = \sum_i (\mathsf{v}(\mathsf{y}^i)\mathsf{e}_i(\mathsf{f}) + \mathsf{y}^i\mathcal{L}_\mathsf{v}\mathsf{e}_i(\mathsf{f})). \tag{7.19}$$

Because the Lie derivative of a vector field is a vector field, we can extract the components of $\mathcal{L}_{\mathsf{v}}\mathsf{e}_i$ using the dual basis. We define $\Delta^i_j(\mathsf{v})$ to be those components:

$$\Delta^i_j(\mathsf{v}) = \tilde{\mathsf{e}}^i\left(\mathcal{L}_{\mathsf{v}}\mathsf{e}_j\right) = \tilde{\mathsf{e}}^i([\mathsf{v},\mathsf{e}_j]). \tag{7.20}$$

So the Lie derivative can be written

$$\left(\mathcal{L}_{\mathsf{v}}\mathsf{y}\right)(\mathsf{f}) = \sum_i \left(\mathsf{v}(\mathsf{y}^i) + \sum_j \Delta^i_j(\mathsf{v})\mathsf{y}^j\right)\mathsf{e}_i(f). \tag{7.21}$$

The components of the Lie derivatives of the basis vector fields are the structure constants for the basis vector fields. (See equation 4.37.) The structure constants are antisymmetric in the lower indices:

$$\tilde{\mathsf{e}}^i\left(\mathcal{L}_{\mathsf{e}_k}\mathsf{e}_j\right) = \tilde{\mathsf{e}}^i([\mathsf{e}_k,\mathsf{e}_j]) = \mathsf{d}^i_{kj}. \tag{7.22}$$

Resolving v into components and applying the product rule, we get

$$\left(\mathcal{L}_{\mathsf{v}}\mathsf{y}\right)(\mathsf{f}) = \sum_k \left(\mathsf{v}^k[\mathsf{e}_k,\mathsf{y}](\mathsf{f}) - \mathsf{y}(\mathsf{v}^k)\mathsf{e}_k(\mathsf{f})\right). \tag{7.23}$$

So Δ^i_j is related to the structure constants by

$$\begin{aligned}\Delta^i_j(\mathsf{v}) &= \tilde{\mathsf{e}}^i\left(\mathcal{L}_{\mathsf{v}}\mathsf{e}_j\right)\\ &= \sum_k \left(\mathsf{v}^k\tilde{\mathsf{e}}^i([\mathsf{e}_k,\mathsf{e}_j]) - \mathsf{e}_j(\mathsf{v}^k)\tilde{\mathsf{e}}^i(\mathsf{e}_k)\right)\\ &= \sum_k \left(\mathsf{v}^k\mathsf{d}^i_{kj} - \mathsf{e}_j(\mathsf{v}^k)\delta^i_k\right)\\ &= \sum_k \mathsf{v}^k\mathsf{d}^i_{kj} - \mathsf{e}_j(\mathsf{v}^i).\end{aligned} \tag{7.24}$$

Note: Despite their appearance, the Δ^i_j are not form fields because $\Delta^i_j(\mathsf{fv}) \neq \mathsf{f}\Delta^i_j(\mathsf{v})$.

Form Fields

We can also define the Lie derivative of a form field $\boldsymbol{\omega}$ with respect to the vector field v by its action on an arbitrary vector field y, using the extended Leibniz rule (see equation 7.10):

$$(\mathcal{L}_{\mathsf{v}}(\boldsymbol{\omega}))(\mathsf{y}) \equiv \mathsf{v}(\boldsymbol{\omega}(\mathsf{y})) - \boldsymbol{\omega}(\mathcal{L}_{\mathsf{v}}\mathsf{y}). \tag{7.25}$$

The first term computes the rate of change of the combination $\boldsymbol{\omega}(\mathsf{y})$ along the integral curve of v, while the second subtracts $\boldsymbol{\omega}$ applied to the change in y. The result is the change in $\boldsymbol{\omega}$ along the curve.

The Lie derivative of a k-form field $\boldsymbol{\omega}$ with respect to a vector field v is a k-form field that is defined by its behavior when applied to k arbitrary vector fields $\mathsf{w}_0, \ldots, \mathsf{w}_{k-1}$. We generalize equation (7.25):

$$\begin{aligned}&\mathcal{L}_{\mathsf{v}}\boldsymbol{\omega}(\mathsf{w}_0, \ldots, \mathsf{w}_{k-1}) \\ &\quad = \mathsf{v}(\boldsymbol{\omega}(\mathsf{w}_0, \ldots, \mathsf{w}_{k-1})) - \sum_{i=0}^{k-1} \boldsymbol{\omega}(\mathsf{w}_0, \ldots, \mathcal{L}_{\mathsf{v}}\mathsf{w}_i, \ldots, \mathsf{w}_{k-1}).\end{aligned} \tag{7.26}$$

Uniform Interpretation

Consider abstracting the equations (7.16), (7.25), and (7.27). The Lie derivative of an object, a, that can apply to other objects, b, to produce manifold functions, $\mathsf{a}(\mathsf{b}) : \mathsf{M} \to \mathsf{R}^n$, is

$$(\mathcal{L}_{\mathsf{v}}\mathsf{a})(\mathsf{b}) = \mathsf{v}(\mathsf{a}(\mathsf{b})) - \mathsf{a}(\mathcal{L}_{\mathsf{v}}\mathsf{b}). \tag{7.27}$$

The first term in this expression computes the rate of change of the compound object $\mathsf{a}(\mathsf{b})$ along integral curves of v, while the second subtracts the change in a due to the change in b along the curves. The result is a measure of the “intrinsic” change in a along integral curves of v, with b held “fixed.”

Properties of the Lie Derivative

As required by properties 7.7–7.5, the Lie derivative is linear in its arguments:

$$\mathcal{L}_{\alpha\mathsf{v}+\beta\mathsf{w}} = \alpha\mathcal{L}_{\mathsf{v}} + \beta\mathcal{L}_{\mathsf{w}}, \tag{7.28}$$

and

$$\mathcal{L}_{\mathsf{v}}\left(\alpha\mathsf{a}+\beta\mathsf{b}\right)=\alpha\mathcal{L}_{\mathsf{v}}\mathsf{a}+\beta\mathcal{L}_{\mathsf{v}}\mathsf{b}, \tag{7.29}$$

with $\alpha, \beta \in \mathsf{R}$ and vector fields or one-form fields a and b.

For any k-form field $\boldsymbol{\omega}$ and any vector field v the exterior derivative commutes with the Lie derivative with respect to the vector field:

$$\mathcal{L}_{\mathsf{v}}(\mathsf{d}\boldsymbol{\omega})=\mathsf{d}(\mathcal{L}_{\mathsf{v}}\boldsymbol{\omega}) \tag{7.30}$$

If $\boldsymbol{\omega}$ is an element of surface then $\mathsf{d}\boldsymbol{\omega}$ is an element of volume. The Lie derivative computes the rate of change of its argument under a deformation described by the vector field. The answer is the same whether we deform the surface before computing the volume or compute the volume and then deform it.

We can verify this in 3-dimensional rectangular space for a general one-form field:[4]

```
(((- ((Lie-derivative V) (d theta))
     (d ((Lie-derivative V) theta)))
  X Y)
 R3-rect-point)
0
```

and for the general two-form field:

[4]In these experiments we need some setup.

```
(define a (literal-manifold-function 'alpha R3-rect))
(define b (literal-manifold-function 'beta R3-rect))
(define c (literal-manifold-function 'gamma R3-rect))

(define-coordinates (up x y z) R3-rect)

(define theta (+ (* a dx) (* b dy) (* c dz)))

(define omega
  (+ (* a (wedge dy dz))
     (* b (wedge dz dx))
     (* c (wedge dx dy))))

(define X (literal-vector-field 'X-rect R3-rect))
(define Y (literal-vector-field 'Y-rect R3-rect))
(define Z (literal-vector-field 'Z-rect R3-rect))
(define V (literal-vector-field 'V-rect R3-rect))
(define R3-rect-point
  ((point R3-rect) (up 'x0 'y0 'z0)))
```

```
(((- ((Lie-derivative V) (d omega))
     (d ((Lie-derivative V) omega)))
  X Y Z)
 R3-rect-point)
0
```

The Lie derivative satisfies a another nice elementary relationship. If v and w are two vector fields, then

$$[\mathcal{L}_{\mathsf{v}}, \mathcal{L}_{\mathsf{w}}] = \mathcal{L}_{[\mathsf{v},\mathsf{w}]}. \tag{7.31}$$

Again, for our general one-form field $\boldsymbol{\theta}$:

```
((((- (commutator (Lie-derivative X) (Lie-derivative Y))
      (Lie-derivative (commutator X Y)))
   theta)
  Z)
 R3-rect-point)
0
```

and for the two-form field $\boldsymbol{\omega}$:

```
((((- (commutator (Lie-derivative X) (Lie-derivative Y))
      (Lie-derivative (commutator X Y)))
   omega)
  Z V)
 R3-rect-point)
0
```

Exponentiating Lie Derivatives

The Lie derivative computes the rate of change of objects as they are advanced along integral curves. The Lie derivative of an object produces another object of the same type, so we can iterate Lie derivatives. This gives us Taylor series for objects along the curve.

The operator $e^{t\mathcal{L}_{\mathsf{v}}} = 1 + t\mathcal{L}_v + \frac{t^2}{2!}\mathcal{L}_{\mathsf{v}}^2 + \ldots$ evolves objects along the curve by parameter t. For example, the exponential of a Lie derivative applied to a vector field is

$$\begin{aligned} e^{t\mathcal{L}_{\mathsf{v}}}\mathsf{y} &= \mathsf{y} + t\mathcal{L}_{\mathsf{v}}\mathsf{y} + \frac{t^2}{2}{\mathcal{L}_{\mathsf{v}}}^2\mathsf{y} + \cdots \\ &= \mathsf{y} + t[\mathsf{v},\mathsf{y}] + \frac{t^2}{2}[\mathsf{v},[\mathsf{v},\mathsf{y}]] + \cdots \end{aligned} \tag{7.32}$$

Consider a simple case. We advance the coordinate-basis vector field $\partial/\partial\mathsf{y}$ by an angle a around the circle. Let $\mathsf{J}_z = x\,\partial/\partial\mathsf{y} - y\,\partial/\partial\mathsf{x}$, the circular vector field. We recall

```
(define Jz (- (* x d/dy) (* y d/dx)))
```

We can apply the exponential of the Lie derivative with respect to J_z to $\partial/\partial\mathsf{y}$. We examine how the result affects a general function on the manifold:

```
(series:for-each print-expression
  ((((exp (* 'a (Lie-derivative Jz))) d/dy)
    (literal-manifold-function 'f-rect R3-rect))
   ((point R3-rect) (up 1 0 0)))
  5)
(((partial 0) f-rect) (up 1 0))
(* -1 a (((partial 1) f-rect) (up 1 0)))
(* -1/2 (expt a 2) (((partial 0) f-rect) (up 1 0)))
(* 1/6 (expt a 3) (((partial 1) f-rect) (up 1 0)))
(* 1/24 (expt a 4) (((partial 0) f-rect) (up 1 0)))
;Value: ...
```

Apparently the result is

$$\exp\left(a\mathcal{L}_{(\mathsf{x}\,\partial/\partial\mathsf{y}-\mathsf{y}\,\partial/\partial\mathsf{x})}\right)\frac{\partial}{\partial\mathsf{y}} = -\sin(a)\frac{\partial}{\partial\mathsf{x}} + \cos(a)\frac{\partial}{\partial\mathsf{y}}. \tag{7.33}$$

Interior Product

There is a simple but useful operation available between vector fields and form fields called *interior product*. This is the substitution of a vector field v into the first argument of a p-form field $\boldsymbol{\omega}$ to produce a $p-1$-form field:

$$(i_{\mathsf{v}}\boldsymbol{\omega})(\mathsf{v}_1,\ldots\mathsf{v}_{\mathsf{p}-1}) = \boldsymbol{\omega}(\mathsf{v},\mathsf{v}_1,\ldots\mathsf{v}_{\mathsf{p}-1}) \tag{7.34}$$

There is a mundane identity corresponding to the product rule for the Lie derivative of an interior product:

$$\mathcal{L}_{\mathsf{v}}\left(i_{\mathsf{y}}\boldsymbol{\omega}\right) = i_{\mathcal{L}_{\mathsf{v}}\mathsf{y}}\boldsymbol{\omega} + i_{\mathsf{y}}\left(\mathcal{L}_{\mathsf{v}}\boldsymbol{\omega}\right). \tag{7.35}$$

And there is a rather nice identity for the Lie derivative in terms of the interior product called *Cartan's formula*:

$$\mathcal{L}_{\mathsf{v}}\boldsymbol{\omega} = i_{\mathsf{v}}(\mathsf{d}\boldsymbol{\omega}) + \mathsf{d}(i_{\mathsf{v}}\boldsymbol{\omega}) \tag{7.36}$$

We can verify Cartan's formula in a simple case with a program:

```
(define X (literal-vector-field 'X-rect R3-rect))
(define Y (literal-vector-field 'Y-rect R3-rect))
(define Z (literal-vector-field 'Z-rect R3-rect))

(define a (literal-manifold-function 'alpha R3-rect))
(define b (literal-manifold-function 'beta R3-rect))
(define c (literal-manifold-function 'gamma R3-rect))

(define omega
  (+ (* a (wedge dx dy))
     (* b (wedge dy dz))
     (* c (wedge dz dx))))

(define ((L1 X) omega)
  (+ ((interior-product X) (d omega))
     (d ((interior-product X) omega))))

((- (((Lie-derivative X) omega) Y Z)
    (((L1 X) omega) Y Z))
 ((point R3-rect) (up 'x0 'y0 'z0)))
0
```

Note that $i_{\mathsf{v}} \circ i_{\mathsf{u}} + i_{\mathsf{u}} \circ i_{\mathsf{v}} = 0$. One consequence of this is that $i_{\mathsf{v}} \circ i_{\mathsf{v}} = 0$.

7.2 Covariant Derivative

The covariant derivative is another kind of directional derivative operator. We write the covariant derivative operator with respect to a vector field v as ∇_{v}. This is pronounced "covariant derivative with respect to v" or "nabla v."

Covariant Derivative of Vector Fields

We may also choose our $F_\delta^{\mathsf{v}}\mathsf{u}$ to define what we mean by "parallel" transport of the vector field u along an integral curve of the vector field v. This may correspond to our usual understanding of parallel in situations where we have intuitive insight.

The notion of parallel transport is path dependent. Remember our example from the Introduction, page 1: Start at the North Pole carrying a stick along a line of longitude to the Equator, always pointing it south, parallel to the surface of the Earth. Then proceed eastward for some distance, still pointing the stick south. Finally, return to the North Pole along this new line of longitude, keeping the stick pointing south all the time. At the pole the stick will not point in the same direction as it did at the beginning of the trip, and the discrepancy will depend on the amount of eastward motion.[5]

So if we try to carry a stick parallel to itself and tangent to the sphere, around a closed path, the stick generally does not end up pointing in the same direction as it started. The result of carrying the stick from one point on the sphere to another depends on the path taken. However, the direction of the stick at the endpoint of a path does not depend on the rate of transport, just on the particular path on which it is carried. Parallel transport over a zero-length path is the identity.

A vector may be resolved as a linear combination of other vectors. If we parallel-transport each component, and form the same linear combination, we get the transported original vector. Thus parallel transport on a particular path for a particular distance is a linear operation.

So the transport function F_δ^{v} is a linear operator on the components of its argument, and thus

$$F_\delta^{\mathsf{v}}\mathsf{u}(\mathsf{f})(\mathsf{m}) = \sum_{i,j} (A_j^i(\delta)(\mathsf{u}^j \circ \phi_{-\delta}^{\mathsf{v}})\mathsf{e}_i(\mathsf{f}))(\mathsf{m}) \tag{7.37}$$

for some functions A_j^i that depend on the particular path (hence its tangent vector v) and the initial point. We reach back along the integral curve to pick up the components of u and then parallel-transport them forward by the matrix $A_j^i(\delta)$ to form the components of the parallel-transported vector at the advanced point.

As before, we compute

$$\nabla_{\mathsf{v}}\mathsf{u}(\mathsf{f})(\mathsf{m}) = Dg(0), \tag{7.38}$$

where

$$g(\delta) = \mathsf{u}(\mathsf{f})(\mathsf{m}) - (F_\delta^{\mathsf{v}}\mathsf{u})(\mathsf{f})(\mathsf{m}). \tag{7.39}$$

[5]In the introduction the stick was always kept East-West rather than pointing South, but the phenomenon is the same!

Expanding with respect to a basis $\{\mathsf{e}_i\}$ we get

$$g(\delta) = \sum_i \left(\mathsf{u}^i \mathsf{e}_i(\mathsf{f}) - \sum_j A^i_j(\delta)(\mathsf{u}^j \circ \phi^{\mathsf{v}}_{-\delta})\mathsf{e}_i(\mathsf{f}) \right)(\mathsf{m}). \tag{7.40}$$

By the product rule for derivatives,

$$Dg(\delta) = \sum_{ij} \left(A^i_j(\delta)((\mathsf{v}(\mathsf{u}^j)) \circ \phi^{\mathsf{v}}_{-\delta})\mathsf{e}_i(\mathsf{f}) - DA^i_j(\delta)(\mathsf{u}^j \circ \phi^{\mathsf{v}}_{-\delta})\mathsf{e}_i(\mathsf{f}) \right)(\mathsf{m}). \tag{7.41}$$

So, since $A^i_j(0)(\mathsf{m})$ is the identity multiplier, and ϕ^{v}_0 is the identity function,

$$Dg(0) = \sum_i \left(\mathsf{v}(\mathsf{u}^i)(\mathsf{m})\mathsf{e}_i(\mathsf{f}) - \sum_j DA^i_j(0)\mathsf{u}^j(\mathsf{m})\mathsf{e}_i(\mathsf{f}) \right)(\mathsf{m}). \tag{7.42}$$

We need $DA^i_j(0)$. Parallel transport depends on the path, but not on the parameterization of the path. From this we can deduce that $DA^i_j(0)$ can be written as one-form fields applied to the vector field v, as follows.

Introduce B to make the dependence of As on v explicit:

$$A^i_j(\delta) = B^i_j(\mathsf{v})(\delta). \tag{7.43}$$

Parallel transport depends on the path but not on the rate along the path. Incrementally, if we scale the vector field v by ξ

$$\frac{d}{d\delta}(B(\mathsf{v})(\delta)) = \frac{d}{d\delta}(B(\xi\mathsf{v})(\delta/\xi)). \tag{7.44}$$

Using the chain rule

$$D(B(\mathsf{v}))(\delta) = \frac{1}{\xi} D(B(\xi\mathsf{v}))(\frac{\delta}{\xi}), \tag{7.45}$$

so, for $\delta = 0$

$$\xi D(B(\mathsf{v}))(0) = D(B(\xi\mathsf{v}))(0). \tag{7.46}$$

The scale factor ξ can vary from place to place. So $DA^i_j(0)$ is homogeneous in v over manifold functions. This is stronger than the homogeneity required by equation (7.7).

The superposition property (equation (7.6)) is true of the ordinary directional derivative of manifold functions. By analogy we require it to be true of directional derivatives of vector fields.

These two properties imply that $DA^i_j(0)$ is a one-form field:

$$DA^i_j(0) = -\boldsymbol{\varpi}^i_j(\mathsf{v}), \tag{7.47}$$

where the minus sign is a matter of convention.

As before, we can take a stab at computing the covariant derivative of a vector field by supplying an appropriate transport operator for `F` in `F->directional-derivative`. Again, this is expanded to a given order with a given coordinate system. These will be unnecessary in the final version.

```
(define (covariant-derivative-vector omega coordsys order)
  (let ((Phi (phi coordsys order)))
    (F->directional-derivative
      (F-parallel omega Phi coordsys))))

(define ((((((F-parallel omega phi coordsys) v) delta) u) f) m)
  (let ((basis (coordinate-system->basis coordsys)))
    (let ((etilde (basis->1form-basis basis))
          (e (basis->vector-basis basis)))
      (let ((m0 (((phi v) (- delta)) m)))
        (let ((Aij (+ (identity-like ((omega v) m0))
                      (* delta (- ((omega v) m0)))))
              (ui ((etilde u) m0)))
          (* ((e f) m) (* Aij ui)))))))
```

So

$$Dg(0) = \sum_i \left(\mathsf{v}(\mathsf{u}^i)(\mathsf{m}) + \sum_j \boldsymbol{\varpi}^i_j(\mathsf{v})(\mathsf{m})\mathsf{u}^j(\mathsf{m}) \right) \mathsf{e}_i(\mathsf{f})(\mathsf{m}). \tag{7.48}$$

Thus the covariant derivative is

$$\nabla_{\mathsf{v}}\mathsf{u}(\mathsf{f}) = \sum_i \left(\mathsf{v}(\mathsf{u}^i) + \sum_j \boldsymbol{\varpi}^i_j(\mathsf{v})\mathsf{u}^j \right) \mathsf{e}_i(\mathsf{f}). \tag{7.49}$$

The one-form fields $\boldsymbol{\varpi}^i_j$ are called the *Cartan one-forms*, or the *connection one-forms*. They are defined with respect to the basis e.

As a program, the covariant derivative is:[6]

```
(define ((((covariant-derivative-vector Cartan) V) U) f)
  (let ((basis (Cartan->basis Cartan))
        (Cartan-forms (Cartan->forms Cartan)))
    (let ((vector-basis (basis->vector-basis basis))
          (1form-basis (basis->1form-basis basis)))
      (let ((u-components (1form-basis U)))
        (* (vector-basis f)
           (+ (V u-components)
              (* (Cartan-forms V) u-components)))))))
```

An important property of $\nabla_{\mathsf{v}}\mathsf{u}$ is that it is linear over manifold functions g in the first argument

$$\nabla_{\mathsf{gv}}\mathsf{u}(\mathsf{f}) = \mathsf{g}\nabla_{\mathsf{v}}\mathsf{u}(\mathsf{f}), \tag{7.50}$$

consistent with the fact that the Cartan forms ϖ^i_j share the same property.

Additionally, we can extend the product rule, for any manifold function g and any vector field u:

$$\begin{aligned}\nabla_{\mathsf{v}}(\mathsf{gu})(\mathsf{f}) &= \sum_i \left(\mathsf{v}(\mathsf{gu}^i) + \sum_j \varpi^i_j(\mathsf{v})\,\mathsf{gu}^j\right)\mathsf{e}_i(\mathsf{f}) \\ &= \sum_i \mathsf{v}(\mathsf{g})\mathsf{u}^i\mathsf{e}_i(\mathsf{f}) + \mathsf{g}\nabla_{\mathsf{v}}(\mathsf{u})(\mathsf{f}) \\ &= (\nabla_{\mathsf{v}}\mathsf{g})\mathsf{u}(\mathsf{f}) + \mathsf{g}\nabla_{\mathsf{v}}(\mathsf{u})(\mathsf{f}). \end{aligned} \tag{7.51}$$

An Alternate View

As we did with the Lie derivative (equations 7.18–7.21), we can write the vector field

$$\mathsf{u}(\mathsf{f})(\mathsf{m}) = \sum_i \mathsf{u}^i(\mathsf{m})\mathsf{e}_i(\mathsf{f})(\mathsf{m}). \tag{7.52}$$

By the extended product rule, equation (7.51), we get:

$$\nabla_{\mathsf{v}}\mathsf{u}(\mathsf{f}) = \sum_i (\mathsf{v}(\mathsf{u}^i)\mathsf{e}_i(\mathsf{f}) + \mathsf{u}^i\nabla_{\mathsf{v}}\mathsf{e}_i(\mathsf{f})). \tag{7.53}$$

[6]This program is incomplete. It must construct a vector field, it must make a differential operator, and it does not apply to functions or forms.

Because the covariant derivative of a vector field is a vector field we can extract the components of $\nabla_{\mathsf{v}}\mathsf{e}_i$ using the dual basis:

$$\varpi^i_j(\mathsf{v}) = \tilde{\mathsf{e}}^i\left(\nabla_{\mathsf{v}}\mathsf{e}_j\right). \tag{7.54}$$

This gives an alternate expression for the Cartan one forms. So

$$\nabla_{\mathsf{v}}\mathsf{u}(\mathsf{f}) = \sum_i \left(\mathsf{v}(\mathsf{u}^i) + \sum_j \varpi^i_j(\mathsf{v})\mathsf{u}^j\right)\mathsf{e}_i(\mathsf{f}). \tag{7.55}$$

This analysis is parallel to the analysis of the Lie derivative, except that here we have the Cartan form fields ϖ^i_j and there we had Δ^i_j, which are not form fields.

Notice that the Cartan forms appear here (equation 7.53) in terms of the covariant derivatives of the basis vectors. By contrast, in the first derivation (see equation 7.42) the Cartan forms appear as the derivatives of the linear forms that accomplish the parallel transport of the coefficients.

The Cartan forms can be constructed from the dual basis one-forms:

$$\varpi^i_j(\mathsf{v})(\mathsf{m}) = \sum_k \Gamma^i_{jk}(\mathsf{m})\,\tilde{\mathsf{e}}^k(\mathsf{v})(\mathsf{m}). \tag{7.56}$$

The connection coefficient functions Γ^i_{jk} are called the *Christoffel coefficients* (traditionally called *Christoffel symbols*).[7] Making use of the structures,[8] the Cartan forms are

$$\varpi(\mathsf{v}) = \Gamma\,\tilde{\mathsf{e}}(\mathsf{v}). \tag{7.57}$$

Conversely, the Christoffel coefficients may be obtained from the Cartan forms

$$\Gamma^i_{jk} = \varpi^i_j(\mathsf{e}_k). \tag{7.58}$$

[7]This terminology may be restricted to the case in which the basis is a coordinate basis.

[8]The structure of the Cartan forms ϖ together with this equation forces the shape of the Christoffel coefficient structure.

Covariant Derivative of One-Form Fields

The covariant derivative of a vector field induces a compatible covariant derivative for a one-form field. Because the application of a one-form field to a vector field yields a manifold function, we can evaluate the covariant derivative of such an application. Let $\boldsymbol{\tau}$ be a one-form field and w be a vector field. Then

$$\begin{aligned}
\nabla_{\mathsf{v}}(\boldsymbol{\tau}(\mathsf{w})) &= \mathsf{v}\left(\sum_j \boldsymbol{\tau}_j \mathsf{w}^j\right) \\
&= \sum_j \left(\mathsf{v}(\boldsymbol{\tau}_j)\mathsf{w}^j + \boldsymbol{\tau}_j \mathsf{v}(\mathsf{w}^j)\right) \\
&= \sum_j \left(\mathsf{v}(\boldsymbol{\tau}_j)\mathsf{w}^j + \boldsymbol{\tau}_j \left(\tilde{\mathsf{e}}^j(\nabla_{\mathsf{v}}\mathsf{w}) - \sum_k \boldsymbol{\varpi}^j_k(\mathsf{v})\mathsf{w}^k\right)\right) \\
&= \sum_j \left(\mathsf{v}(\boldsymbol{\tau}_j)\mathsf{w}^j - \boldsymbol{\tau}_j \sum_k \boldsymbol{\varpi}^j_k(\mathsf{v})\mathsf{w}^k\right) + \boldsymbol{\tau}(\nabla_{\mathsf{v}}\mathsf{w}) \\
&= \sum_j \left(\mathsf{v}(\boldsymbol{\tau}_j)\tilde{\mathsf{e}}^j - \boldsymbol{\tau}_j \sum_k \boldsymbol{\varpi}^j_k(\mathsf{v})\tilde{\mathsf{e}}^k\right)(\mathsf{w}) + \boldsymbol{\tau}(\nabla_{\mathsf{v}}\mathsf{w})
\end{aligned}$$

So if we define the covariant derivative of a one-form field to be

$$\nabla_{\mathsf{v}}(\boldsymbol{\tau}) = \sum_k \left(\mathsf{v}(\boldsymbol{\tau}_k) - \sum_j \boldsymbol{\tau}_j \boldsymbol{\varpi}^j_k(\mathsf{v})\right)\tilde{\mathsf{e}}^k, \tag{7.59}$$

then the generalized product rule holds:

$$\nabla_{\mathsf{v}}\left(\boldsymbol{\tau}(\mathsf{u})\right) = \left(\nabla_{\mathsf{v}}\boldsymbol{\tau}\right)(\mathsf{u}) + \boldsymbol{\tau}\left(\nabla_{\mathsf{v}}\mathsf{u}\right). \tag{7.60}$$

Alternatively, assuming the generalized product rule forces the definition of covariant derivative of a one-form field.

As a program this is

```
(define ((((covariant-derivative-1form Cartan) V) tau) U)
  (let ((nabla_V ((covariant-derivative-vector Cartan) V)))
    (- (V (tau U)) (tau (nabla_V U)))))
```

This program extends naturally to higher-rank form fields:

```
(define ((((covariant-derivative-form Cartan) V) tau) vs)
  (let ((k (get-rank tau))
        (nabla_V ((covariant-derivative-vector Cartan) V)))
    (- (V (apply tau vs))
       (sigma (lambda (i)
                (apply tau
                       (list-with-substituted-coord vs i
                           (nabla_V (list-ref vs i)))))
              0 (- k 1)))))
```

Change of Basis

The basis-independence of the covariant derivative implies a relationship between the Cartan forms in one basis and the equivalent Cartan forms in another basis. Recall (equation 4.13) that the basis vector fields of two bases are always related by a linear transformation. Let J be the matrix of coefficient functions and let e and e' be down tuples of basis vector fields. Then

$$\mathsf{e}(\mathsf{f}) = \mathsf{e}'(\mathsf{f})\mathsf{J}. \tag{7.61}$$

We want the covariant derivative to be independent of basis. This will determine how the connection transforms with a change of basis:

$$\begin{aligned}
\nabla_{\mathsf{v}}\mathsf{u}(\mathsf{f}) &= \sum_i \mathsf{e}_i(\mathsf{f})\left(\mathsf{v}(\mathsf{u}^i) + \sum_j \boldsymbol{\varpi}^i_j(\mathsf{v})\mathbf{u}^j\right) \\
&= \sum_{ijk} \mathsf{e}'_i(\mathsf{f})\mathsf{J}^i_j\left(\mathsf{v}\left((\mathsf{J}^{-1})^j_k(\mathsf{u}')^k\right) + \sum_l \boldsymbol{\varpi}^j_k(\mathsf{v})(\mathsf{J}^{-1})^k_l(\mathsf{u}')^l\right) \\
&= \sum_i \mathsf{e}'_i(\mathsf{f})\left(\mathsf{v}((\mathsf{u}')^i) + \sum_{jk} \mathsf{J}^i_j\mathsf{v}\left((\mathsf{J}^{-1})^j_k\right)(\mathsf{u}')^k\right. \\
&\qquad\qquad \left. + \sum_{jkl} \mathsf{J}^i_j\boldsymbol{\varpi}^j_k(\mathsf{v})(\mathsf{J}^{-1})^k_l(\mathsf{u}')^l\right) \\
&= \sum_i \mathsf{e}'_i(\mathsf{f})\left(\mathsf{v}((\mathsf{u}')^i) + \sum_j (\boldsymbol{\varpi}')^i_j(\mathsf{v})(\mathsf{u}')^j\right).
\end{aligned} \tag{7.62}$$

The last line of equation (7.62) gives the formula for the covariant derivative we would have written down naturally in the primed coordinates; comparing with the next-to-last line, we see that

$$\varpi'(\mathsf{v}) = \mathsf{J}\mathsf{v}\left(\mathsf{J}^{-1}\right) + \mathsf{J}\varpi(\mathsf{v})\mathsf{J}^{-1}. \tag{7.63}$$

This transformation rule is weird. It is not a linear transformation of ϖ because the first term is an offset that depends on v. So it is not required that $\varpi' = 0$ when $\varpi = 0$. Thus ϖ is not a tensor field. See Appendix C.

We can write equation (7.61) in terms of components

$$\mathbf{e}_i(\mathsf{f}) = \sum_j \mathbf{e}'_j(\mathsf{f})\mathsf{J}^j{}_i \tag{7.64}$$

Let $\mathsf{K} = \mathsf{J}^{-1}$, so $\sum_j \mathsf{K}^i_j(\mathsf{m})\mathsf{J}^j{}_k(\mathsf{m}) = \delta^i_k$. Then

$$\varpi'^i_l(\mathsf{v}) = \sum_j \mathsf{J}^i_j\, \mathsf{v}(\mathsf{K}^j{}_l) + \sum_{jk} \mathsf{J}^i_j\, \varpi^j_k(\mathsf{v})\, \mathsf{K}^k{}_l \tag{7.65}$$

The transformation rule for ϖ is implemented in the following program:

```
(define (Cartan-transform Cartan basis-prime)
  (let ((basis (Cartan->basis Cartan))
        (forms (Cartan->forms Cartan))
        (prime-dual-basis (basis->1form-basis basis-prime))
        (prime-vector-basis (basis->vector-basis basis-prime)))
    (let ((vector-basis (basis->vector-basis basis))
          (1form-basis (basis->1form-basis basis)))
      (let ((J-inv (s:map/r 1form-basis prime-vector-basis))
            (J (s:map/r prime-dual-basis vector-basis)))
        (let ((omega-prime-forms
               (procedure->1form-field
                (lambda (v)
                  (+ (* J (v J-inv))
                     (* J (* (forms v) J-inv)))))))
          (make-Cartan omega-prime-forms basis-prime))))))
```

The `s:map/r` procedure constructs a tuple of the same shape as its second argument whose elements are the result of applying the first argument to the corresponding elements of the second argument.

We can illustrate that the covariant derivative is independent of the coordinate system in a simple case, using rectangular and polar coordinates in the plane.[9] We can choose Christoffel coefficients for rectangular coordinates that are all zero:[10]

```
(define R2-rect-Christoffel
  (make-Christoffel
   (let ((zero (lambda (m) 0)))
     (down (down (up zero zero)
                 (up zero zero))
           (down (up zero zero)
                 (up zero zero))))
   R2-rect-basis))
```

With these Christoffel coefficients, parallel transport preserves the components relative to the rectangular basis. This corresponds to our usual notion of parallel in the plane. We will see later in Chapter 9 that these Christoffel coefficients are a natural choice for the plane. From these we obtain the Cartan form:[11]

```
(define R2-rect-Cartan
  (Christoffel->Cartan R2-rect-Christoffel))
```

And from equation (7.63) we can get the corresponding Cartan form for polar coordinates:

```
(define R2-polar-Cartan
  (Cartan-transform R2-rect-Cartan R2-polar-basis))
```

[9]We will need a few definitions:

```
(define R2-rect-basis (coordinate-system->basis R2-rect))
(define R2-polar-basis (coordinate-system->basis R2-polar))
(define-coordinates (up x y) R2-rect)
(define-coordinates (up r theta) R2-polar)
```

[10]Since the Christoffel coefficients are basis-dependent they are packaged with the basis.

[11]The code for making the Cartan forms is as follows:

```
(define (Christoffel->Cartan Christoffel)
  (let ((basis (Christoffel->basis Christoffel))
        (Christoffel-symbols (Christoffel->symbols Christoffel)))
    (make-Cartan
     (* Christoffel-symbols (basis->1form-basis basis))
     basis)))
```

The vector field $\partial/\partial\theta$ generates a rotation in the plane (the same as `circular`). The covariant derivative with respect to $\partial/\partial\mathsf{x}$ of $\partial/\partial\theta$ applied to an arbitrary manifold function is:

```
(define circular (- (* x d/dy) (* y d/dx)))

(define f (literal-manifold-function 'f-rect R2-rect))
(define R2-rect-point ((point R2-rect) (up 'x0 'y0)))

(((((covariant-derivative R2-rect-Cartan) d/dx)
   circular)
  f)
 R2-rect-point)
(((partial 1) f-rect) (up x0 y0))
```

Note that this is the same thing as $\partial/\partial\mathsf{y}$ applied to the function:

```
((d/dy f) R2-rect-point)
(((partial 1) f-rect) (up x0 y0))
```

In rectangular coordinates, where the Christoffel coefficients are zero, the covariant derivative $\nabla_{\mathsf{u}}\mathsf{v}$ is the vector whose coefficents are obtained by applying u to the coefficients of v. Here, only one coefficient of $\partial/\partial\theta$ depends on x, the coefficient of $\partial/\partial\mathsf{y}$, and it depends linearly on x. So $\nabla_{\partial/\partial\mathsf{x}}\partial/\partial\theta = \partial/\partial\mathsf{y}$. (See Figure 7.1.)

Note that we get the same answer if we use polar coordinates to compute the covariant derivative:

```
(((((covariant-derivative R2-polar-Cartan) d/dx) J) f)
 R2-rect-point)
(((partial 1) f-rect) (up x0 y0))
```

In rectangular coordinates the Christoffel coefficients are all zero; in polar coordinates there are nonzero coefficients, but the value of the covariant derivative is the same. In polar coordinates the basis elements vary with position, and the Christoffel coefficients compensate for this.

Of course, this is a pretty special situation. Let's try something more general:

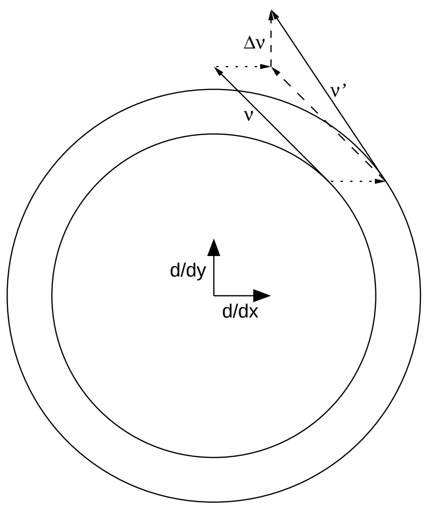

Figure 7.1 If v and v' are "arrow" representations of vectors in the circular field and we parallel-transport v in the $\partial/\partial\mathsf{x}$ direction, then the difference between v' and the parallel transport of v is in the $\partial/\partial\mathsf{y}$ direction.

```
(define V (literal-vector-field 'V-rect R2-rect))
(define W (literal-vector-field 'W-rect R2-rect))

(((((- (covariant-derivative R2-rect-Cartan)
       (covariant-derivative R2-polar-Cartan))
    V)
   W)
  f)
 R2-rect-point)
0
```

7.3 Parallel Transport

We have defined parallel transport of a vector field along integral curves of another vector field. But not all paths are integral curves of a vector field. For example, paths that cross themselves are not integral curves of any vector field.

Here we extend the idea of parallel transport of a stick to make sense for arbitrary paths on the manifold. Any path can be written as a map γ from the real-line manifold to the manifold M. We construct a vector field over the map u_γ by parallel-transporting the stick to all points on the path γ.

For any path γ there are locally directional derivatives of functions on M defined by tangent vectors to the curve. The vector over the map $\mathsf{w}_\gamma = d\gamma(\partial/\partial\mathsf{t})$ is a directional derivative of functions on the manifold M along the path γ.

Our goal is to determine the equations satisfied by the vector field over the map u_γ. Consider the parallel-transport $F_\delta^{\mathsf{w}_\gamma}\mathsf{u}_\gamma$.[12] So a vector field u_γ is parallel-transported to itself if and only if $\mathsf{u}_\gamma = F_\delta^{\mathsf{w}_\gamma}\mathsf{u}_\gamma$. Restricted to a path, the equation analogous to equation (7.40) is

$$g(\delta) = \sum_i \left(u^i(t) - \sum_j A^i_j(\delta) u^j(t-\delta) \right) \mathsf{e}^\gamma_i(\mathsf{f})(\mathsf{t}), \tag{7.66}$$

where the coefficient function u^i is now a function on the real-line parameter manifold and where we have rewritten the basis as a basis over the map γ.[13] Here $g(\delta) = 0$ if u_γ is parallel-transported into itself.

Taking the derivative and setting $\delta = 0$ we find

$$0 = \sum_i \left(Du^i(t) + \sum_j {}^\gamma\boldsymbol{\varpi}^i_j(\mathsf{w}_\gamma)(t) u^j(t) \right) \mathsf{e}^\gamma_i(\mathsf{f})(\mathsf{t}). \tag{7.67}$$

But this implies that

$$0 = Du^i(t) + \sum_j {}^\gamma\boldsymbol{\varpi}^i_j(\mathsf{w}_\gamma)(\mathsf{t}) u^j(t), \tag{7.68}$$

an ordinary differential equation in the coefficients of u_γ.

[12] The argument w_γ makes sense because our parallel-transport operator never depended on the vector field tangent to the integral curve existing off of the curve. Because the connection is a form field (see equation 7.47), it does not depend on the value of its vector argument anywhere except at the point where it is being evaluated.

The argument u_γ is more difficult. We must modify equation (7.37):

$$F_\delta^{\mathsf{w}_\gamma}\mathsf{u}_\gamma(\mathsf{f})(t) = \sum_{i,j} A^i_j(\delta) u^j(t-\delta) \mathsf{e}^\gamma_i(\mathsf{f})(t).$$

[13] You may have noticed that t and t appear here. The real-line manifold point t has coordinate t.

We can abstract these equations of parallel transport by inventing a covariant derivative over a map. We also generalize the time line to a source manifold N.

$$\nabla^{\gamma}_{\mathsf{v}}\mathsf{u}_{\gamma}(\mathsf{f})(\mathsf{n}) = \sum_i \left(\mathsf{v}(u^i)(\mathsf{n}) + \sum_j {}^{\gamma}\boldsymbol{\varpi}^i_j(d\gamma(\mathsf{v}))(\mathsf{n})u^j(\mathsf{n}) \right) \mathsf{e}^{\gamma}_i(\mathsf{f})(\mathsf{n}), \quad (7.69)$$

where the map $\gamma : \mathsf{N} \to \mathsf{M}$, v is a vector on N, u_γ is a vector over the map γ, f is a function on M, and n is a point in N. Indeed, if w is a vector field on M, f is a manifold function on M, and if $d\gamma(\mathsf{v}) = \mathsf{w}_\gamma$ then

$$\nabla^{\gamma}_{\mathsf{v}}\mathsf{u}_{\gamma}(\mathsf{f})(\mathsf{n}) = \nabla_{\mathsf{w}}\mathsf{u}(\mathsf{f})(\gamma(\mathsf{n})). \quad (7.70)$$

This is why we are justified in calling $\nabla^{\gamma}_{\mathsf{v}}$ a covariant derivative.

Respecializing the source manifold to the real line, we can write the equations governing the parallel transport of u_γ as

$$\nabla^{\gamma}_{\partial/\partial\mathsf{t}}\mathsf{u}_{\gamma} = 0. \quad (7.71)$$

We obtain the set of differential equations (7.68) for the coordinates of u_γ, the vector over the map γ, that is parallel-transported along the curve γ:

$$Du^i(t) + \sum_j {}^{\gamma}\boldsymbol{\varpi}^i_j(d\gamma(\partial/\partial t))(\mathsf{t})u^j(t) = 0. \quad (7.72)$$

Expressing the Cartan forms in terms of the Christoffel coefficients we obtain

$$Du^i(t) + \sum_{j,k} \Gamma^i_{jk}(\gamma(\mathsf{t}))D\sigma^k(t)u^j(t) = 0 \quad (7.73)$$

where $\sigma = \chi_{\mathsf{M}} \circ \gamma \circ \chi_{\mathsf{R}}^{-1}$ are the coordinates of the path (χ_{M} and χ_{R} are the coordinate functions for M and the real line).

On a sphere

Let's figure out what the equations of parallel transport of u_γ, an arbitrary vector over the map γ, along an arbitrary path γ on a sphere are. We start by constructing the necessary manifold.

```
(define sphere (make-manifold S^2 2 3))
(define S2-spherical
  (coordinate-system-at 'spherical 'north-pole sphere))
(define S2-basis
  (coordinate-system->basis S2-spherical))
```

We need the path γ, which we represent as a map from the real line to M, and w, the parallel-transported vector over the map:

```
(define gamma
  (compose (point S2-spherical)
           (up (literal-function 'alpha)
               (literal-function 'beta))
           (chart R1-rect)))
```

where `alpha` is the colatitude and `beta` is the longitude.

We also need an arbitrary vector field `u_gamma` over the map `gamma`. To make this we multiply the structure of literal component functions by the vector basis structure.

```
(define basis-over-gamma
  (basis->basis-over-map gamma S2-basis))

(define u_gamma
  (* (up (compose (literal-function 'u^0)
                  (chart R1-rect))
         (compose (literal-function 'u^1)
                  (chart R1-rect)))
     (basis->vector-basis basis-over-gamma)))
```

We specify a connection by giving the Christoffel coefficients.[14]

```
(define S2-Christoffel
  (make-Christoffel
   (let ((zero  (lambda (point) 0)))
     (down (down (up zero zero)
                 (up zero (/ 1 (tan theta))))
           (down (up zero (/ 1 (tan theta)))
                 (up (- (* (sin theta) (cos theta))) zero))))
   S2-basis))

(define sphere-Cartan (Christoffel->Cartan S2-Christoffel))
```

[14]We will show later that these Christoffel coefficients are a natural choice for the sphere.

Finally, we compute the residual of the equation (7.71) that governs parallel transport for this situation:[15]

```
(define-coordinates t R1-rect)

(s:map/r
 (lambda (omega)
   ((omega
     (((covariant-derivative sphere-Cartan gamma)
       d/dt)
      u_gamma))
    ((point R1-rect) 'tau)))
 (basis->1form-basis basis-over-gamma))
(up (+ (* -1
          (sin (alpha tau))
          (cos (alpha tau))
          ((D beta) tau)
          (u^1 tau))
       ((D u^0) tau))
    (/ (+ (* (u^0 tau) (cos (alpha tau)) ((D beta) tau))
          (* ((D alpha) tau) (cos (alpha tau)) (u^1 tau))
          (* ((D u^1) tau) (sin (alpha tau))))
       (sin (alpha tau))))
```

Thus the equations governing the evolution of the components of the transported vector are:

$$\begin{aligned} Du^0(\tau) &= \sin(\alpha(\tau))\cos(\alpha(\tau))D\beta(\tau)u^1(\tau) \\ Du^1(\tau) &= -\frac{\cos(\alpha(\tau))}{\sin(\alpha(\tau))}\left(D\beta(\tau)u^0(\tau) + D\alpha(\tau)u^1(\tau)\right) \end{aligned} \tag{7.74}$$

These equations describe the transport on a sphere, but more generally they look like

$$Du(\tau) = f(\sigma(\tau), D\sigma(\tau))\, u(\tau), \tag{7.75}$$

where σ is the tuple of the coordinates of the path on the manifold and u is the tuple of the components of the vector. The equation is linear in u and is driven by the path σ, as in a variational equation.

[15] If we give `covariant-derivative` an extra argument, in addition to the Cartan form, the covariant derivative treats the extra argument as a map and transforms the Cartan form to work over the map.

We now set this up for numerical integration. Let $s(t) = (t, u(t))$ be a state tuple, combining the time and the coordinates of u_γ at that time. Then we define g:

$$g(s(t)) = Ds(t) = (1, Du(t)), \tag{7.76}$$

where $Du(t)$ is the tuple of right-hand sides of equation (7.72).

On a Great Circle

We illustrate parallel transport in a case where we should know the answer: we carry a vector along a great circle of a sphere. Given a path and Cartan forms for the manifold we can produce a state derivative suitable for numerical integration. Such a state derivative takes a state and produces the derivative of the state.

```
(define (g gamma Cartan)
  (let ((omega
	 ((Cartan->forms
	    (Cartan->Cartan-over-map Cartan gamma))
	  ((differential gamma) d/dt))))
    (define ((the-state-derivative) state)
      (let ((t ((point R1-rect) (ref state 0)))
	    (u (ref state 1)))
	(up 1 (*  -1 (omega t) u))))
    the-state-derivative))
```

The path on the sphere will be the target of a map from the real line. We choose one that starts at the origin of longitudes on the equator and follows the great circle that makes a given tilt angle with the equator.

```
(define ((transform tilt) coords)
  (let ((colat (ref coords 0))
	(long (ref coords 1)))
    (let ((x (* (sin colat) (cos long)))
	  (y (* (sin colat) (sin long)))
	  (z (cos colat)))
      (let ((vp ((rotate-x tilt) (up x y z))))
	(let ((colatp (acos (ref vp 2)))
	      (longp (atan (ref vp 1) (ref vp 0))))
	  (up colatp longp))))))
```

```
(define (tilted-path tilt)
  (define (coords t)
    ((transform tilt) (up :pi/2 t)))
  (compose (point S2-spherical)
           coords
           (chart R1-rect)))
```

A southward pointing vector, with components `(up 1 0)`, is transformed to an initial vector for the tilted path by multiplying by the derivative of the tilt transform at the initial point. We then parallel transport this vector by numerically integrating the differential equations. In this example we tilt by 1 radian, and we advance for $\pi/2$ radians. In this case we know the answer: by advancing by $\pi/2$ we walk around the circle a quarter of the way and at that point the transported vector points south:

```
((state-advancer (g (tilted-path 1) sphere-Cartan))
 (up 0 (* ((D (transform 1)) (up :pi/2 0)) (up 1 0)))
 pi/2)
(up 1.5707963267948957
    (up .9999999999997626 7.376378522558262e-13))
```

However, if we transport by 1 radian rather than $\pi/2$, the numbers are not so pleasant, and the transported vector no longer points south:

```
((state-advancer (g (tilted-path 1) sphere-Cartan))
 (up 0 (* ((D (transform 1)) (up :pi/2 0)) (up 1 0)))
 1)
(up 1. (up .7651502649360408 .9117920272006472))
```

But the transported vector can be obtained by tilting the original southward-pointing vector after parallel-transporting along the equator:[16]

```
(* ((D (transform 1)) (up :pi/2 1)) (up 1 0))
(up .7651502649370375 .9117920272004736)
```

[16] A southward-pointing vector remains southward-pointing when it is parallel-transported along the equator. To do this we do not have to integrate the differential equations, because we know the answer.

7.4 Geodesic Motion

In geodesic motion the velocity vector is parallel-transported by itself. Recall (equation 6.9) that the velocity is the differential of the vector $\partial/\partial \mathsf{t}$ over the map γ. The equation of geodesic motion is[17]

$$\nabla^{\gamma}_{\partial/\partial \mathsf{t}} d\gamma(\partial/\partial \mathsf{t}) = 0. \tag{7.78}$$

In coordinates, this is

$$D^2\sigma^i(t) + \sum_{jk} \Gamma^i_{jk}(\gamma(t)) D\sigma^j(t) D\sigma^k(t) = 0, \tag{7.79}$$

where $\sigma(t)$ is the coordinate path corresponding to the manifold path γ.

For example, let's consider geodesic motion on the surface of a unit sphere. We let `gamma` be a map from the real line to the sphere, with colatitude `alpha` and longitude `beta`, as before. The geodesic equation is:

```
(show-expression
 (((((covariant-derivative sphere-Cartan gamma)
     d/dt)
    ((differential gamma) d/dt))
   (chart S2-spherical))
  ((point R1-rect) 't0)))
```

$$\boxed{\begin{pmatrix} -\cos(\alpha(t0))\sin(\alpha(t0))(D\beta(t0))^2 + D^2\alpha(t0) \\ \dfrac{2D\beta(t0)\cos(\alpha(t0))D\alpha(t0)}{\sin(\alpha(t))} + D^2\beta(t0) \end{pmatrix}}$$

The geodesic equation is the same as the Lagrange equation for free motion constrained to the surface of the unit sphere. The

[17]The equation of a geodesic path is often said to be

$$\nabla_{\mathsf{v}}\mathsf{v} = 0, \tag{7.77}$$

but this is nonsense. The geodesic equation is a constraint on the path, but the path does not appear in this equation. Further, the velocity along a path is not a vector field, so it cannot appear in either argument to the covariant derivative.

What is true is that a vector field v all of whose integral curves are geodesics satisfies equation (7.77).

Lagrangian for motion on the sphere is the composition of the free-particle Lagrangian and the state transformation induced by the coordinate constraint:[18]

```
(define (Lfree s)
  (* 1/2 (square (velocity s))))

(define (sphere->R3 s)
  (let ((q (coordinate s)))
    (let ((theta (ref q 0)) (phi (ref q 1)))
      (up (* (sin theta) (cos phi))
          (* (sin theta) (sin phi))
          (cos theta)))))

(define Lsphere
  (compose Lfree (F->C sphere->R3)))
```

Then the Lagrange equations are:

```
(show-expression
 (((Lagrange-equations Lsphere)
   (up (literal-function 'alpha)
       (literal-function 'beta)))
  't))
```

$$\boxed{\begin{bmatrix} -\left(D\beta\left(t\right)\right)^2\sin\left(\alpha\left(t\right)\right)\cos\left(\alpha\left(t\right)\right)+D^2\alpha\left(t\right) \\ \\ 2D\alpha\left(t\right)D\beta\left(t\right)\sin\left(\alpha\left(t\right)\right)\cos\left(\alpha\left(t\right)\right)+D^2\beta\left(t\right)\left(\sin\left(\alpha\left(t\right)\right)\right)^2 \end{bmatrix}}$$

The Lagrange equations are true of the same paths as the geodesic equations. The second Lagrange equation is the second geodesic equation multiplied by $(\sin(\alpha(t)))^2$, and the Lagrange equations are arranged in a down tuple, whereas the geodesic equations are arranged in an up tuple.[19] The two systems are equivalent unless $\alpha(t) = 0$, where the coordinate system is singular.

[18]The method of formulating a system with constraints by composing a free system with the state-space coordinate transformation that represents the constraints can be found in [18], Section 1.6.3. The procedure `F->C` takes a coordinate transformation and produces a corresponding transformation of Lagrangian state.

[19]The geodesic equations and the Lagrange equations are related by a contraction with the metric.

Exercise 7.1: Hamiltonian Evolution

We have just seen that the Lagrange equations for the motion of a free particle constrained to the surface of a sphere determine the geodesics on the sphere. We can investigate this phenomenon in the Hamiltonian formulation. The Hamiltonian is obtained from the Lagrangian by a Legendre transformation:

```
(define Hsphere
  (Lagrangian->Hamiltonian Lsphere))
```

We can get the coordinate representation of the Hamiltonian vector field as follows:

```
((phase-space-derivative Hsphere)
 (up 't (up 'theta 'phi) (down 'p_theta 'p_phi)))
(up 1
    (up p_theta
        (/ p_phi (expt (sin theta) 2)))
    (down (/ (* (expt p_phi 2) (cos theta))
             (expt (sin theta) 3))
          0))
```

The state space for Hamiltonian evolution has five dimensions: time, two dimensions of position on the sphere, and two dimensions of momentum:

```
(define state-space
  (make-manifold R^n 5))
(define states
  (coordinate-system-at 'rectangular 'origin state-space))
(define-coordinates
  (up t (up theta phi) (down p_theta p_phi))
  states)
```

So now we have coordinate functions and the coordinate-basis vector fields and coordinate-basis one-form fields.

a. Define the Hamiltonian vector field as a linear combination of these fields.

b. Obtain the first few terms of the Taylor series for the evolution of the coordinates (θ, ϕ) by exponentiating the Lie derivative of the Hamiltonian vector field.

Exercise 7.2: Lie Derivative and Covariant Derivative

How are the Lie derivative and the covariant derivative related?

a. Prove that for every vector field there exists a connection such that the covariant derivative for that connection and the given vector field is equivalent to the Lie derivative with respect to that vector field.

b. Show that there is no connection that for every vector field makes the Lie derivative the same as the covariant derivative with the chosen connection.

8
Curvature

If the intrinsic curvature of a manifold is not zero, a vector parallel-transported around a small loop will end up different from the vector that started. We saw the consequence of this before, on page 1 and on page 93. The Riemann tensor encapsulates this idea.

The Riemann curvature operator is

$$\mathcal{R}(\mathsf{w},\mathsf{v}) = [\nabla_{\mathsf{w}}, \nabla_{\mathsf{v}}] - \nabla_{[\mathsf{w},\mathsf{v}]} \tag{8.1}$$

The traditional Riemann tensor is[1]

$$\mathsf{R}(\boldsymbol{\omega},\mathsf{u},\mathsf{w},\mathsf{v}) = \boldsymbol{\omega}((\mathcal{R}(\mathsf{w},\mathsf{v}))(\mathsf{u})), \tag{8.2}$$

where $\boldsymbol{\omega}$ is a one-form field that measures the incremental change in the vector field u caused by parallel-transporting it around the loop defined by the vector fields w and v. R allows us to compute the *intrinsic curvature* of a manifold at a point.

The Riemann curvature is computed by

```
(define ((Riemann-curvature nabla) w v)
  (- (commutator (nabla w) (nabla v))
     (nabla (commutator w v))))
```

The `Riemann-curvature` procedure is parameterized by the relevant covariant-derivative operator `nabla`, which implements ∇. The `nabla` is itself dependent on the connection, which provides the details of the local geometry. The same `Riemann-curvature` procedure works for ordinary covariant derivatives and for covariant derivatives over a map. Given two vector fields, the result of `((Riemann-curvature nabla) w v)` is a procedure that takes a vector field and produces a vector field so we can implement the Riemann tensor as

[1] [10], [3], and [13] use our definition. [19] uses a different convention for the order of arguments and a different sign. See Appendix C for a definition of tensors.

```
(define ((Riemann nabla) omega u w v)
  (omega (((Riemann-curvature nabla) w v) u)))
```

So, for example,[2]

```
(((Riemann (covariant-derivative sphere-Cartan))
  dphi d/dtheta d/dphi d/dtheta)
 ((point S2-spherical) (up 'theta0 'phi0)))
1
```

Here we have computed the ϕ component of the result of carrying a $\partial/\partial\theta$ basis vector around the parallelogram defined by $\partial/\partial\phi$ and $\partial/\partial\theta$. The result shows a net rotation in the ϕ direction.

Most of the sixteen coefficients of the Riemann tensor for the sphere are zero. The following are the nonzero coefficients:

$$\begin{aligned}
&\mathsf{R}\left(\mathsf{d}\theta, \frac{\partial}{\partial\phi}, \frac{\partial}{\partial\theta}, \frac{\partial}{\partial\phi}\right)(\chi^{-1}(q^\theta, q^\phi)) = \left(\sin(q^\theta)\right)^2\\
&\mathsf{R}\left(\mathsf{d}\theta, \frac{\partial}{\partial\phi}, \frac{\partial}{\partial\phi}, \frac{\partial}{\partial\theta}\right)(\chi^{-1}(q^\theta, q^\phi)) = -\left(\sin(q^\theta)\right)^2\\
&\mathsf{R}\left(\mathsf{d}\phi, \frac{\partial}{\partial\theta}, \frac{\partial}{\partial\theta}, \frac{\partial}{\partial\phi}\right)(\chi^{-1}(q^\theta, q^\phi)) = -1\\
&\mathsf{R}\left(\mathsf{d}\phi, \frac{\partial}{\partial\theta}, \frac{\partial}{\partial\phi}, \frac{\partial}{\partial\theta}\right)(\chi^{-1}(q^\theta, q^\phi)) = 1
\end{aligned} \tag{8.3}$$

8.1 Explicit Transport

We will show that the result of the Riemann calculation of the change in a vector, as we traverse a loop, is what we get by explicitly calculating the transport. The coordinates of the vector to be transported are governed by the differential equations (see equation 7.72)

$$Du^i(t) = -\sum_j \boldsymbol{\varpi}^i_j(\mathsf{v})(\chi^{-1}(\sigma(t)))u^j(t) \tag{8.4}$$

[2]The connection specified by `sphere-Cartan` is defined on page 107.

and the coordinates as a function of time, $\sigma = \chi \circ \gamma \circ \chi_{\mathsf{R}}^{-1}$, of the path γ, are governed by the differential equations[3]

$$D\sigma(t) = \mathsf{v}(\chi)(\chi^{-1}(\sigma(t))). \tag{8.5}$$

We have to integrate these equations (8.4, 8.5) together to transport the vector over the map u_γ a finite distance along the vector field v.

Let $s(t) = (\sigma(t), u(t))$ be a state tuple, combining σ the coordinates of γ, and u the coordinates of u_γ. Then

$$Ds(t) = (D\sigma(t), Du(t)) = g(s(t)), \tag{8.6}$$

where g is the tuple of right-hand sides of equations (8.4, 8.5).

The differential equations describing the evolution of a function h of state s along the state path are

$$D(h \circ s) = (Dh \circ s)(g \circ s) = L_g h \circ s, \tag{8.7}$$

defining the operator L_g.

Exponentiation gives a finite evolution:[4]

$$h(s(t+\epsilon)) = (e^{\epsilon L_g} h)(s(t)). \tag{8.8}$$

The finite parallel transport of the vector with components u is

$$u(t+\epsilon) = (e^{\epsilon L_g} U)(s(t)), \tag{8.9}$$

where the selector $U(\sigma, u) = u$, and the initial state is $s(t) = (\sigma(t), u(t))$.

Consider parallel-transporting a vector u around a parallelogram defined by two coordinate-basis vector fields w and v. The vector u is really a vector over a map, where the map is the parametric curve describing our parallelogram. This map is implicitly defined in terms of the vector fields w and v. Let g_w and g_v be the right-hand sides of the differential equations for parallel transport

[3]The map γ takes points on the real line to points on the target manifold. The chart χ gives coordinates of points on the target manifold while χ_{R} gives a time coordinate on the real line.

[4]The series may not converge for large increments in the independent variable. In this case it is appropriate to numerically integrate the differential equations directly.

along w and v respectively. Then evolution along w for interval ϵ, then along v for interval ϵ, then reversing w, and reversing v, brings σ back to where it started to second order in ϵ.

The state $s = (\sigma, u)$ after transporting s_0 around the loop is[5]

$$\begin{aligned}(e^{-\epsilon L_{g_v}} I) \circ (e^{-\epsilon L_{g_w}} I) \circ (e^{\epsilon L_{g_v}} I) \circ (e^{\epsilon L_{g_w}} I)(s_0) \\ = (e^{\epsilon L_{g_w}} e^{\epsilon L_{g_v}} e^{-\epsilon L_{g_w}} e^{-\epsilon L_{g_v}} I)(s_0) \\ = (e^{\epsilon^2 [L_{g_w}, L_{g_v}] + \cdots} I)(s_0).\end{aligned} \tag{8.10}$$

So the lowest-order change in the transported vector is

$$\epsilon^2 U(([L_{g_w}, L_{g_v}] I)(s_0)), \tag{8.11}$$

where $U(\sigma, u) = u$.

However, if w and v do not commute the indicated loop does not bring σ back to the starting point, to second order in ϵ. We must account for the commutator. (See figure 4.2.) In the general case the lowest order change in the transported vector is

$$\epsilon^2 U((([L_{g_w}, L_{g_v}] - L_{g_{[w,v]}}) I)(s_0)). \tag{8.12}$$

This is what the Riemann tensor computation gives, scaled by ϵ^2.

Verification in Two Dimensions

We can verify this in two dimensions. We need to make the structure representing a state:

```
(define (make-state sigma u) (vector sigma u))

(define (Sigma state) (ref state 0))

(define (U-select state) (ref state 1))
```

[5] The parallel-transport operators are evolution operators, and therefore descend into composition:

$$e^A (F \circ G) = F \circ (e^A G),$$

for any state function G and any compatible F. As a consequence, we have the following identity:

$$e^A e^B I = e^A ((e^B I) \circ I) = (e^B I) \circ (e^A I),$$

where I is the identity function on states.

And now we get to the meat of the matter: First we find the rate of change of the components of the vector u as we carry it along the vector field v.[6]

```
(define ((Du v) state)
  (let ((CF (Cartan->forms general-Cartan-2)))
    (* -1
       ((CF v) (Chi-inverse (Sigma state)))
       (U-select state))))
```

We also need to determine the rate of change of the coordinates of the integral curve of v.

```
(define ((Dsigma v) state)
  ((v Chi) (Chi-inverse (Sigma state))))
```

Putting these together to make the derivative of the state vector

```
(define ((g v) state)
  (make-state ((Dsigma v) state) ((Du v) state)))
```

gives us just what we need to construct the differential operator for evolution of the combined state:

```
(define (L v)
  (define ((l h) state)
    (* ((D h) state) ((g v) state)))
  (make-operator l))
```

So now we can demonstrate that the lowest-order change resulting from explicit parallel transport of a vector around an infinitesimal loop is what is computed by the Riemann curvature.

[6] The setup for this experiment is a bit complicated. We need to make a manifold with a general connection.

```
(define Chi-inverse (point R2-rect))
(define Chi (chart R2-rect))
```

We now make the Cartan forms from the most general 2-dimensional Christoffel coefficient structure:

```
(define general-Cartan-2
 (Christoffel->Cartan
   (literal-Christoffel-2 'Gamma R2-rect)))
```

```
(let ((U (literal-vector-field 'U-rect R2-rect))
      (W (literal-vector-field 'W-rect R2-rect))
      (V (literal-vector-field 'V-rect R2-rect))
      (sigma (up 'sigma0 'sigma1)))
  (let ((nabla (covariant-derivative general-Cartan-2))
        (m (Chi-inverse sigma)))
    (let ((s (make-state sigma ((U Chi) m))))
      (- (((- (commutator (L V) (L W))
              (L (commutator V W)))
           U-select)
          s)
         (((((Riemann-curvature nabla) W V) U) Chi) m)))))
(up 0 0)
```

Geometrically

The explicit transport above was done with differential equations operating on a state consisting of coordinates and components of the vector being transported. We can simplify this so that it is entirely built on manifold objects, eliminating the state. After a long algebraic story we find that

$$\begin{aligned}((\mathcal{R}(\mathsf{w},\mathsf{v}))(\mathsf{u}))(\mathsf{f}) &= \mathsf{e}(\mathsf{f})\,\{(\mathsf{w}(\boldsymbol{\varpi}(\mathsf{v})) - \mathsf{v}(\boldsymbol{\varpi}(\mathsf{w})) - \boldsymbol{\varpi}([\mathsf{w},\mathsf{v}]))\tilde{\mathsf{e}}(\mathsf{u}) \\ &\qquad + \boldsymbol{\varpi}(\mathsf{w})\boldsymbol{\varpi}(\mathsf{v})\tilde{\mathsf{e}}(\mathsf{u}) - \boldsymbol{\varpi}(\mathsf{v})\boldsymbol{\varpi}(\mathsf{w})\tilde{\mathsf{e}}(\mathsf{u})\} \qquad (8.13)\end{aligned}$$

or as a program:

```
(define ((((curvature-from-transport Cartan) w v) u) f)
  (let* ((CF (Cartan->forms Cartan))
         (basis (Cartan->basis Cartan))
         (fi (basis->1form-basis basis))
         (ei (basis->vector-basis basis)))
    (* (ei f)
       (+ (* (- (- (w (CF v)) (v (CF w)))
                (CF (commutator w v)))
             (fi u))
          (- (* (CF w) (* (CF v) (fi u)))
             (* (CF v) (* (CF w) (fi u))))))))
```

This computes the same operator as the traditional Riemann curvature operator:

```
(define (test coordsys Cartan)
  (let ((m (typical-point coordsys))
        (u (literal-vector-field 'u-coord coordsys))
        (w (literal-vector-field 'w-coord coordsys))
        (v (literal-vector-field 'v-coord coordsys))
        (f (literal-manifold-function 'f-coord coordsys)))
   (let ((nabla (covariant-derivative Cartan)))
     (- (((((curvature-from-transport Cartan) w v) u) f) m)
        (((((Riemann-curvature nabla) w v) u) f) m)))))

(test R2-rect general-Cartan-2)
0

(test R2-polar general-Cartan-2)
0
```

Terms of the Riemann Curvature

Since the Riemann curvature is defined as in equation(8.1):

$$\mathcal{R}(\mathsf{w},\mathsf{v}) = [\nabla_{\mathsf{w}}, \nabla_{\mathsf{v}}] - \nabla_{[\mathsf{w},\mathsf{v}]} \tag{8.14}$$

it is natural[7] to identify these terms with the corresponding terms in

$$(([L_{g_w}, L_{g_v}] - L_{g_{[w,v]}})U)(s_0). \tag{8.15}$$

Unfortunately, this does not work, as demonstrated below:

```
(let ((U (literal-vector-field 'U-rect R2-rect))
      (V (literal-vector-field 'V-rect R2-rect))
      (W (literal-vector-field 'W-rect R2-rect))
      (nabla (covariant-derivative general-Cartan-2))
      (sigma (up 'sigma0 'sigma1)))
  (let ((m (Chi-inverse sigma)))
    (let ((s (make-state sigma ((U Chi) m))))
      (- (((commutator (L W) (L V)) U-select) s)
         ((((commutator (nabla W) (nabla V)) U) Chi)
          m)))))
a nonzero mess
```

The obvious identification does not work, but neither does the other one!

[7]People often say "Geodesic evolution is exponentiation of the covariant derivative." But this is wrong. The evolution is by exponentiation of L_g.

```
(let ((U (literal-vector-field 'U-rect R2-rect))
      (V (literal-vector-field 'V-rect R2-rect))
      (W (literal-vector-field 'W-rect R2-rect))
      (nabla (covariant-derivative general-Cartan-2))
      (sigma (up 'sigma0 'sigma1)))
  (let ((m (Chi-inverse sigma)))
    (let ((s (make-state sigma ((U Chi) m))))
      (- (((commutator (L W) (L V)) U-select) s)
         ((((nabla (commutator W V)) U) Chi)
          m)))))
a nonzero mess
```

Let's compute the two parts of the Riemann curvature operator and see how this works out. First, recall

$$\begin{aligned} \nabla_{\mathsf{v}}\mathsf{u}(\mathsf{f}) &= \sum_i \mathsf{e}_i(\mathsf{f}) \left(\mathsf{v}(\tilde{\mathsf{e}}^i(\mathsf{u})) + \sum_j \boldsymbol{\varpi}^i_j(\mathsf{v})\tilde{\mathsf{e}}^j(\mathsf{u}) \right) && (8.16) \\ &= \mathsf{e}(\mathsf{f})\left(\mathsf{v}(\tilde{\mathsf{e}}(\mathsf{u})) + \boldsymbol{\varpi}(\mathsf{v})\tilde{\mathsf{e}}(\mathsf{u})\right), && (8.17) \end{aligned}$$

where the second form uses tuple arithmetic. Now let's consider the first part of the Riemann curvature operator:

$$\begin{aligned} &[\nabla_{\mathsf{w}}, \nabla_{\mathsf{v}}]\,\mathsf{u} \\ &\quad = \nabla_{\mathsf{w}}\nabla_{\mathsf{v}}\mathsf{u} - \nabla_{\mathsf{v}}\nabla_{\mathsf{w}}\mathsf{u} \\ &\quad = \mathsf{e}\left\{\mathsf{w}(\mathsf{v}(\tilde{\mathsf{e}}(\mathsf{u})) + \boldsymbol{\varpi}(\mathsf{v})\tilde{\mathsf{e}}(\mathsf{u})) + \boldsymbol{\varpi}(\mathsf{w})(\mathsf{v}(\tilde{\mathsf{e}}(\mathsf{u})) + \boldsymbol{\varpi}(\mathsf{v})\tilde{\mathsf{e}}(\mathsf{u}))\right\} \\ &\qquad - \mathsf{e}\left\{\mathsf{v}(\mathsf{w}(\tilde{\mathsf{e}}(\mathsf{u})) + \boldsymbol{\varpi}(\mathsf{w})\tilde{\mathsf{e}}(\mathsf{u})) + \boldsymbol{\varpi}(\mathsf{v})(\mathsf{w}(\tilde{\mathsf{e}}(\mathsf{u})) + \boldsymbol{\varpi}(\mathsf{w})\tilde{\mathsf{e}}(\mathsf{u}))\right\} \\ &\quad = \mathsf{e}\{[\mathsf{w}, \mathsf{v}]\tilde{\mathsf{e}}(\mathsf{u}) \\ &\qquad\qquad + \mathsf{w}(\boldsymbol{\varpi}(\mathsf{v}))\tilde{\mathsf{e}}(\mathsf{u}) - \mathsf{v}(\boldsymbol{\varpi}(\mathsf{w}))\tilde{\mathsf{e}}(\mathsf{u}) \\ &\qquad\qquad + \boldsymbol{\varpi}(\mathsf{w})\boldsymbol{\varpi}(\mathsf{v})\tilde{\mathsf{e}}(\mathsf{u}) - \boldsymbol{\varpi}(\mathsf{v})\boldsymbol{\varpi}(\mathsf{w})\tilde{\mathsf{e}}(\mathsf{u})\}. \qquad (8.18) \end{aligned}$$

The second term of the Riemann curvature operator is

$$\nabla_{[\mathsf{w},\mathsf{v}]}\mathsf{u} = \mathsf{e}\left\{[\mathsf{w}, \mathsf{v}]\tilde{\mathsf{e}}(\mathsf{u}) + \boldsymbol{\varpi}([\mathsf{w}, \mathsf{v}])\tilde{\mathsf{e}}(\mathsf{u})\right\}. \qquad (8.19)$$

The difference of these is the Riemann curvature operator. Notice that the first term in each cancels, and the rest gives equation (8.13).

Ricci Curvature

One measure of the curvature is the Ricci tensor, which is computed from the Riemann tensor by

$$R(\mathsf{u},\mathsf{v}) = \sum_i \mathsf{R}(\tilde{\mathsf{e}}^i, \mathsf{u}, \mathsf{e}_i, \mathsf{v}) \tag{8.20}$$

Expressed as a program:

```
(define ((Ricci nabla basis) u v)
  (contract
   (lambda (ei wi) ((Riemann nabla) wi u ei v))
   basis))
```

Einstein's field equation (9.27) for gravity, which we will encounter later, is expressed in terms of the Ricci tensor.

Exercise 8.1: Ricci of a Sphere

Compute the components of the Ricci tensor of the surface of a sphere.

Exercise 8.2: Pseudosphere

A pseudosphere is a surface in 3-dimensional space. It is a surface of revolution of a tractrix about its asymptote (along the $\hat{z}$-axis). We can make coordinates for the surface (t, θ) where t is the coordinate along the asymptote and θ is the angle of revolution. We embed the pseudosphere in rectangular 3-dimensional space with

```
(define (pseudosphere q)
  (let ((t (ref q 0)) (theta (ref q 1)))
    (up (* (sech t) (cos theta))
        (* (sech t) (sin theta))
        (- t (tanh t)))))
```

The structure of Christoffel coefficients for the pseudosphere is

```
(down
 (down (up (/ (+ (* 2 (expt (cosh t) 2) (expt (sinh t) 2))
                 (* -2 (expt (sinh t) 4)) (expt (cosh t) 2)
                 (* -2 (expt (sinh t) 2)))
              (+ (* (cosh t) (expt (sinh t) 3))
                 (* (cosh t) (sinh t))))
           0)
       (up 0
           (/ (* -1 (sinh t)) (cosh t))))
 (down (up 0
           (/ (* -1 (sinh t)) (cosh t)))
       (up (/ (cosh t) (+ (expt (sinh t) 3) (sinh t)))
           0)))
```

Note that this is independent of θ.

Compute the components of the Ricci tensor.

8.2 Torsion

There are many connections that describe the local properties of any particular manifold. A connection has a property called *torsion*, which is computed as follows:

$$\mathcal{T}(\mathsf{u},\mathsf{v}) = \nabla_{\mathsf{u}}\mathsf{v} - \nabla_{\mathsf{v}}\mathsf{u} - [\mathsf{u},\mathsf{v}] \tag{8.21}$$

The torsion takes two vector fields and produces a vector field. The torsion depends on the covariant derivative, which is constructed from the connection.

We account for this dependency by parameterizing the program by `nabla`.

```
(define ((torsion-vector nabla) u v)
  (- (- ((nabla u) v) ((nabla v) u))
     (commutator u v)))

(define ((torsion nabla) omega u v)
  (omega ((torsion-vector nabla) u v)))
```

The torsion for the connection for the 2-sphere specified by the Christoffel coefficients `S2-Christoffel` above is zero. We demonstrate this by applying the torsion to the basis vector fields:

```
(for-each
 (lambda (x)
   (for-each
    (lambda (y)
      (print-expression
      ((((torsion-vector (covariant-derivative sphere-Cartan))
         x y)
        (literal-manifold-function 'f S2-spherical))
       ((point S2-spherical) (up 'theta0 'phi0)))))
    (list  d/dtheta d/dphi)))
 (list  d/dtheta d/dphi))
0
0
0
0
```

Torsion Doesn't Affect Geodesics

There are multiple connections that give the same geodesic curves. Among these connections there is always one with zero torsion. Thus, if you care about only geodesics, it is appropriate to use a torsion-free connection.

Consider a basis e and its dual $\tilde{\mathsf{e}}$. The components of the torsion are

$$\tilde{\mathsf{e}}^k(\mathsf{T}(\mathsf{e}_i, \mathsf{e}_j)) = \Gamma^k_{ij} - \Gamma^k_{ji} + \mathsf{d}^k_{ij}, \tag{8.22}$$

where d^k_{ij} are the structure constants of the basis. See equations (4.37, 4.38). For a commuting basis the structure constants are zero, and the components of the torsion are the antisymmetric part of Γ with respect to the lower indices.

Recall the geodesic equation (7.79):

$$D^2\sigma^i(t) + \sum_{jk} \Gamma^i_{jk}(\gamma(t)) D\sigma^j(t) D\sigma^k(t) = 0, \tag{8.23}$$

Observe that the lower indices of Γ are contracted with two copies of the velocity. Because the use of Γ is symmetrical here, any asymmetry of Γ in its lower indices is irrelevant to the geodesics. Thus one can study the geodesics of any connection by first symmetrizing the connection, eliminating torsion. The resulting equations will be simpler.

8.3 Geodesic Deviation

Geodesics may converge and intersect (as in the lines of longitude on a sphere) or they may diverge (for example, on a saddle). To capture this notion requires some measure of the convergence or divergence, but this requires metrics (see Chapter 9). But even in the absence of a metric we can define a quantity, the *geodesic deviation*, that can be interpreted in terms of relative acceleration of neighboring geodesics from a reference geodesic.

Let there be a one-parameter family of geodesics, with parameter s, and let T be the vector field of tangent vectors to those geodesics:

$$\nabla_{\mathsf{T}}\mathsf{T} = 0. \tag{8.24}$$

We can parameterize travel along the geodesics with parameter t: a geodesic curve $\gamma_s(t) = \phi_t^{\mathsf{T}}(\mathsf{m}_s)$ where

$$\mathsf{f} \circ \phi_t^{\mathsf{T}}(\mathsf{m}_s) = (e^{tT}\mathsf{f})(\mathsf{m}_s). \tag{8.25}$$

Let $\mathsf{U} = \partial/\partial s$ be the vector field corresponding to the displacement of neighboring geodesics.

Locally, (t, s) is a coordinate system on the 2-dimensional submanifold formed by the family of geodesics. The vector fields T and U are a coordinate basis for this coordinate system, so $[\mathsf{T}, \mathsf{U}] = 0$.

The geodesic deviation vector field is defined as:

$$\nabla_{\mathsf{T}}(\nabla_{\mathsf{T}}\mathsf{U}). \tag{8.26}$$

If the connection has zero torsion, the geodesic deviation can be related to the Riemann curvature:

$$\nabla_{\mathsf{T}}(\nabla_{\mathsf{T}}\mathsf{U}) = -\mathcal{R}(\mathsf{U}, \mathsf{T})(\mathsf{T}), \tag{8.27}$$

as follows, using equation (8.21),

$$\nabla_{\mathsf{T}}(\nabla_{\mathsf{T}}\mathsf{U}) = \nabla_{\mathsf{T}}(\nabla_{\mathsf{U}}\mathsf{T}), \tag{8.28}$$

because both the torsion is zero and $[\mathsf{T}, \mathsf{U}] = 0$. Continuing

$$\begin{aligned}\nabla_{\mathsf{T}}(\nabla_{\mathsf{T}}\mathsf{U}) &= \nabla_{\mathsf{T}}(\nabla_{\mathsf{U}}\mathsf{T}) \\ &= \nabla_{\mathsf{T}}(\nabla_{\mathsf{U}}\mathsf{T}) + \nabla_{\mathsf{U}}(\nabla_{\mathsf{T}}\mathsf{T}) - \nabla_{\mathsf{U}}(\nabla_{\mathsf{T}}\mathsf{T}) \\ &= \nabla_{\mathsf{U}}(\nabla_{\mathsf{T}}\mathsf{T}) - \mathcal{R}(\mathsf{U}, \mathsf{T})(\mathsf{T}) \\ &= -\mathcal{R}(\mathsf{U}, \mathsf{T})(\mathsf{T}).\end{aligned} \tag{8.29}$$

In the last line the first term was dropped because T satisfies the geodesic equation (8.24).

The geodesic deviation is defined without using a metric, but it helps to have a metric (see Chapter 9) to interpret the geodesic deviation. Consider two neighboring geodesics, with parameters s and $s + \Delta s$. Given a metric we can assume that t is proportional to path length along each geodesic, and we can define a distance $\delta(s, t, \Delta s)$ between the geodesics at the same value of the parameter t. So the velocity of separation of the two geodesics is

$$(\nabla_{\mathsf{T}}\mathsf{U})\Delta s = \partial_1\delta(s, t, \Delta s)\hat{s} \tag{8.30}$$

where $\hat{s}$ is a unit vector in the direction of increasing s. So $\nabla_{\mathsf{T}}\mathsf{U}$ is the factor of increase of velocity with increase of separation. Similarly, the geodesic deviation can be interpreted as the factor of increase of acceleration with increase of separation:

$$\nabla_{\mathsf{T}}(\nabla_{\mathsf{T}}\mathsf{U})\Delta s = \partial_1\partial_1\delta(s, t, \Delta s)\hat{s}. \tag{8.31}$$

Longitude Lines on a Sphere

Consider longitude lines on the unit sphere.[8] Let `theta` be co-latitude and `phi` be longitude. These are the parameters s and t, respectively. Then let `T` be the vector field `d/dtheta` that is tangent to the longitude lines.

We can verify that every longitude line is a geodesic:

```
((omega (((covariant-derivative Cartan) T) T)) m)
0
```

where `omega` is an arbitrary one-form field.
Now let `U` be `d/dphi`, then `U` commutes with `T`:

```
(((commutator U T) f) m)
0
```

The torsion for the usual connection for the sphere is zero:

```
(let ((X (literal-vector-field 'X-sphere S2-spherical))
      (Y (literal-vector-field 'Y-sphere S2-spherical)))
  ((((torsion-vector nabla) X Y) f) m))
0
```

So we can compute the geodesic deviation using `Riemann`

```
((+ (omega ((nabla T) ((nabla T) U)))
    ((Riemann nabla) omega T U T))
 m)
0
```

confirming equation (8.29).

[8]The setup for this example is:

```
(define-coordinates (up theta phi) S2-spherical)
(define T d/dtheta)
(define U d/dphi)
(define m ((point S2-spherical) (up 'theta0 'phi0)))
(define Cartan (Christoffel->Cartan S2-Christoffel))
(define nabla (covariant-derivative Cartan))
```

Lines of longitude are geodesics. How do the lines of longitude behave? As we proceed from the North Pole, the lines of constant longitude diverge. At the Equator they are parallel and they converge towards the South Pole.

Let's compute $\nabla_{\mathsf{T}}\mathsf{U}$ and $\nabla_{\mathsf{T}}(\nabla_{\mathsf{T}}\mathsf{U})$. We know that the distance is purely in the ϕ direction, so

```
((dphi ((nabla T) U)) m)
(/ (cos theta0) (sin theta0))

((dphi ((nabla T) ((nabla T) U))) m)
-1
```

Let's interpret these results. On a sphere of radius R the distance at colatitude θ between two geodesics separated by $\Delta\phi$ is $d(\phi, \theta, \Delta\phi) = R\sin(\theta)\Delta\phi$. Assuming that θ is uniformly increasing with time, the magnitude of the velocity is just the θ-derivative of this distance:

```
(define ((delta R) phi theta Delta-phi)
  (* R (sin theta) Delta-phi))

(((partial 1) (delta 'R)) 'phi0 'theta0 'Delta-phi)
(* Delta-phi R (cos theta0))
```

The direction of the velocity is the unit vector in the ϕ direction:

```
(define phi-hat
  (* (/ 1 (sin theta)) d/dphi))
```

This comes from the fact that the separation of lines of longitude is proportional to the sine of the colatitude. So the velocity vector field is the product.

We can measure the ϕ component with $d\phi$:

```
((dphi (* (((partial 1) (delta 'R))
            'phi0 'theta0 'Delta-phi)
           phi-hat))
 m)
(/ (* Delta-phi R (cos theta0)) (sin theta0))
```

This agrees with $\nabla_{\mathsf{T}}\mathsf{U}\,\Delta\phi$ for the unit sphere. Indeed, the lines of longitude diverge until they reach the Equator and then they converge.

Similarly, the magnitude of the acceleration is

```
(((partial 1) ((partial 1) (delta 'R)))
 'phi0 'theta0 'Delta-phi)
(* -1 Delta-phi R (sin theta0))
```

and the acceleration vector is the product of this result with $\hat{\phi}$. Measuring this with $d\phi$ we get:

```
((dphi (* (((partial 1) ((partial 1) (delta 'R)))
           'phi0 'theta0 'Delta-phi)
          phi-hat))
 m)
(* -1 Delta-phi R)
```

And this agrees with the calculation of $\nabla_{\mathsf{T}}\nabla_{\mathsf{T}}\mathsf{U}\,\Delta\phi$ for the unit sphere. We see that the separation of the lines of longitude are uniformly decelerated as they progress from pole to pole.

8.4 Bianchi Identities

There are some important mathematical properties of the Riemann curvature. These identities will be used to constrain the possible geometries that can occur.

A system with a symmetric connection, $\Gamma^i_{jk} = \Gamma^i_{kj}$, is torsion free.[9]

```
(define nabla
  (covariant-derivative
    (Christoffel->Cartan
      (symmetrize-Christoffel
        (literal-Christoffel-2 'C R4-rect)))))

(((torsion nabla) omega X Y)
 (typical-point R4-rect))
0
```

[9]Setup for this section:

```
(define omega (literal-1form-field 'omega-rect R4-rect))
(define X (literal-vector-field 'X-rect R4-rect))
(define Y (literal-vector-field 'Y-rect R4-rect))
(define Z (literal-vector-field 'Z-rect R4-rect))
(define V (literal-vector-field 'V-rect R4-rect))
```

The Bianchi identities are defined in terms of a cyclic-summation operator, which is most easily described as a Scheme procedure:

```
(define ((cyclic-sum f) x y z)
  (+ (f x y z) (f y z x) (f z x y)))
```

The first Bianchi identity is

$$\mathsf{R}(\omega, \mathsf{x}, \mathsf{y}, \mathsf{z}) + \mathsf{R}(\omega, \mathsf{y}, \mathsf{z}, \mathsf{x}) + \mathsf{R}(\omega, \mathsf{z}, \mathsf{x}, \mathsf{y}) = 0, \tag{8.32}$$

or, as a program:

```
(((cyclic-sum
   (lambda (x y z) ((Riemann nabla) omega x y z)))
  X Y Z)
 (typical-point R4-rect))
0
```

The second Bianchi identity is

$$\nabla_{\mathsf{x}}\mathsf{R}(\omega, \mathsf{v}, \mathsf{y}, \mathsf{z}) + \nabla_{\mathsf{y}}\mathsf{R}(\omega, \mathsf{v}, \mathsf{z}, \mathsf{x}) + \nabla_{\mathsf{z}}\mathsf{R}(\omega, \mathsf{v}, \mathsf{x}, \mathsf{y}) = 0 \tag{8.33}$$

or, as a program:

```
(((cyclic-sum
   (lambda (x y z)
     (((nabla x) (Riemann nabla)) omega V y z)))
  X Y Z)
 (typical-point R4-rect))
0
```

Things get more complicated when there is torsion. We can make a general connection, which has torsion:

```
(define nabla
  (covariant-derivative
    (Christoffel->Cartan
      (literal-Christoffel-2 'C R4-rect))))

(define R (Riemann nabla))
(define T (torsion-vector nabla))

(define (TT omega x y)
  (omega (T x y)))
```

The first Bianchi identity is now:[10]

```
(((cyclic-sum
    (lambda (x y z)
      (- (R omega x y z)
         (+ (omega (T (T x y) z))
            (((nabla x) TT) omega y z)))))
  X Y Z)
 (typical-point R4-rect))
0
```

and the second Bianchi identity for a general connection is

```
(((cyclic-sum
    (lambda (x y z)
      (+ (((nabla x) R) omega V y z)
         (R omega V (T x y) z))))
  X Y Z)
 (typical-point R4-rect))
0
```

[10]The Bianchi identities are much nastier to write in traditional mathematical notation than as Scheme programs.

9 Metrics

We often want to impose further structure on a manifold to allow us to define lengths and angles. This is done by generalizing the idea of the Euclidean dot product, which allows us to compute lengths of vectors and angles between vectors in traditional vector algebra.

For vectors $\vec{u} = u^x\hat{x} + u^y\hat{y} + u^z\hat{z}$ and $\vec{v} = v^x\hat{x} + v^y\hat{y} + v^z\hat{z}$ the dot product is $\vec{u} \cdot \vec{v} = u^x v^x + u^y v^y + u^z v^z$. The generalization is to provide coefficients for these terms and to include cross terms, consistent with the requirement that the function of two vectors is symmetric. This symmetric, bilinear, real-valued function of two vector fields is called a metric field.

For example, the natural metric on a sphere of radius R is

$$\mathsf{g}(\mathsf{u},\mathsf{v}) = R^2(\mathsf{d}\theta(\mathsf{u})\mathsf{d}\theta(\mathsf{v}) + (\sin\theta)^2\mathsf{d}\phi(\mathsf{u})\mathsf{d}\phi(\mathsf{v})), \tag{9.1}$$

and the Minkowski metric on the 4-dimensional space of special relativity is

$$\mathsf{g}(\mathsf{u},\mathsf{v}) = \mathsf{d}x(\mathsf{u})\mathsf{d}x(\mathsf{v}) + \mathsf{d}y(\mathsf{u})\mathsf{d}y(\mathsf{v}) + \mathsf{d}z(\mathsf{u})\mathsf{d}z(\mathsf{v}) - c^2\mathsf{d}t(\mathsf{u})\mathsf{d}t(\mathsf{v}). \tag{9.2}$$

Although these examples are expressed in terms of a coordinate basis, the value of the metric on vector fields does not depend on the coordinate system that is used to specify the metric.

Given a metric field g and a vector field v the scalar field $\mathsf{g}(\mathsf{v},\mathsf{v})$ is the squared length of the vector at each point of the manifold.

Metric Music

The metric can be used to construct a one-form field $\boldsymbol{\omega}_\mathsf{u}$ from a vector field u, such that for any vector field v we have

$$\omega_\mathsf{u}(\mathsf{v}) = \mathsf{g}(\mathsf{v},\mathsf{u}). \tag{9.3}$$

The operation of constructing a one-form field from a vector field using a metric is called "lowering" the vector field. It is sometimes notated

$$\boldsymbol{\omega}_\mathsf{u} = \mathsf{g}^\flat(\mathsf{u}). \tag{9.4}$$

There is also an inverse metric that takes two one-form fields. It is defined by the relation

$$\delta^i_k = \sum_j \mathsf{g}^{-1}(\tilde{\mathsf{e}}^i, \tilde{\mathsf{e}}^j)\mathsf{g}(\mathsf{e}_j, \mathsf{e}_k), \tag{9.5}$$

where e and $\tilde{\mathsf{e}}$ are any basis and its dual basis.

The inverse metric can be used to construct a vector field v_ω from a one-form field $\boldsymbol{\omega}$, such that for any one-form field $\boldsymbol{\tau}$ we have

$$\boldsymbol{\tau}(\mathsf{v}_\omega) = \mathsf{g}^{-1}(\boldsymbol{\omega}, \boldsymbol{\tau}). \tag{9.6}$$

This definition is implicit, but the vector field can be explicitly computed from the one-form field with respect to a basis as follows:

$$\mathsf{v}_\omega = \sum_i \mathsf{g}^{-1}(\boldsymbol{\omega}, \tilde{\mathsf{e}}^i)\mathsf{e}_i \tag{9.7}$$

The operation of constructing a vector field from a one-form field using a metric is called "raising" the one-form field. It is sometimes notated

$$\mathsf{v}_\omega = \mathsf{g}^\sharp(\boldsymbol{\omega}). \tag{9.8}$$

The raising and lowering operations allow one to interchange the vector fields and the one-form fields. However they should not be confused with the dual operation that allows one to construct a dual one-form basis from a vector basis or construct a vector basis from a one-form basis. The dual operation that interchanges bases is defined without assigning a metric structure on the space.

Lowering a vector field with respect to a metric is a simple program:

```
(define ((lower metric) u)
  (define (omega v) (metric v u))
  (procedure->1form-field omega))
```

But raising a one-form field to make a vector field is a bit more complicated:

```
(define (raise metric basis)
  (let ((gi (metric:invert metric basis)))
    (lambda (omega)
      (contract (lambda (e_i w^i)
                  (* (gi omega w^i) e_i))
                basis))))
```

where `contract` is the trace over a basis of a two-argument function that takes a vector field and a one-form field as its arguments.[1]

```
(define (contract proc basis)
  (let ((vector-basis (basis->vector-basis basis))
        (1form-basis (basis->1form-basis basis)))
    (s:sigma/r proc
               vector-basis
               1form-basis)))
```

9.1 Metric Compatibility

A connection is said to be compatible with a metric g if the covariant derivative for that connection obeys the “product rule”:

$$\nabla_{\mathsf{X}}(\mathsf{g}(\mathsf{Y},\mathsf{Z})) = \mathsf{g}(\nabla_{\mathsf{X}}(\mathsf{Y}),\mathsf{Z}) + \mathsf{g}(\mathsf{Y},\nabla_{\mathsf{X}}(\mathsf{Z})) \tag{9.9}$$

For a metric there is a unique torsion-free connection that is compatible with it. The Christoffel coefficients of the first kind are computed from the metric by the following

$$\bar{\Gamma}_{ijk} = \frac{1}{2}(\mathsf{e}_k(\mathsf{g}(\mathsf{e}_i,\mathsf{e}_j)) + \mathsf{e}_j(\mathsf{g}(\mathsf{e}_i,\mathsf{e}_k)) - \mathsf{e}_i(\mathsf{g}(\mathsf{e}_j,\mathsf{e}_k))), \tag{9.10}$$

for the coordinate basis e. We can then construct the Christoffel coefficients of the second kind (the ones used previously to define a connection) by “raising the first index.” To do this we define a function of three vectors, with a wierd currying:

$$\tilde{\Gamma}(\mathsf{v},\mathsf{w})(\mathsf{u}) = \sum_{ijk} \bar{\Gamma}_{ijk}\tilde{\mathsf{e}}^i(\mathsf{u})\tilde{\mathsf{e}}^j(\mathsf{v})\tilde{\mathsf{e}}^k(\mathsf{w}) \tag{9.11}$$

[1]Notice that `raise` and `lower` are not symmetrical. This is because vector fields and form fields are not symmetrical: a vector field takes a manifold function as its argument, whereas a form field takes a vector field as its argument. This asymmetry is not apparent in traditional treatments based on index notation.

This function takes two vector fields and produces a one-form field. We can use it with equation (9.7) to construct a new function that takes two vector fields and produces a vector field:

$$\hat{\Gamma}(\mathsf{v},\mathsf{w}) = \sum_i \mathsf{g}^{-1}(\tilde{\Gamma}(\mathsf{v},\mathsf{w}), \tilde{\mathsf{e}}^i)\mathsf{e}_i \tag{9.12}$$

We can now construct the Christoffel coefficients of the second kind:

$$\Gamma^i_{jk} = \tilde{\mathsf{e}}^i(\hat{\Gamma}(\mathsf{e}_j,\mathsf{e}_k)) = \sum_m \bar{\Gamma}_{mjk}\mathsf{g}^{-1}(\tilde{\mathsf{e}}^m, \tilde{\mathsf{e}}^i) \tag{9.13}$$

The Cartan forms are then just

$$\boldsymbol{\varpi}^i_j = \sum_k \Gamma^i_{jk}\tilde{\mathsf{e}}^k = \sum_k \tilde{\mathsf{e}}^i(\hat{\Gamma}(\mathsf{e}_j,\mathsf{e}_k))\tilde{\mathsf{e}}^k \tag{9.14}$$

So, for example, we can compute the Christoffel coefficients for the sphere from the metric for the sphere. First, we need the metric

```
(define ((g-sphere R) u v)
  (* (square R)
     (+ (* (dtheta u) (dtheta v))
        (* (compose (square sin) theta)
           (dphi u)
           (dphi v)))))
```

The Christoffel coefficients of the first kind are a complex structure with all three indices down:

```
((Christoffel->symbols
  (metric->Christoffel-1 (g-sphere 'R) S2-basis))
 ((point S2-spherical) (up 'theta0 'phi0)))
(down
 (down (down 0 0)
       (down 0 (* (* (cos theta0) (sin theta0)) (expt R 2))))
 (down (down 0 (* (* (cos theta0) (sin theta0)) (expt R 2)))
       (down (* (* -1 (cos theta0) (sin theta0)) (expt R 2))
             0)))
```

And the Christoffel coefficients of the second kind have the innermost index up:

```
((Christoffel->symbols
  (metric->Christoffel-2 (g-sphere 'R) S2-basis))
 ((point S2-spherical) (up 'theta0 'phi0)))
(down (down (up 0 0)
            (up 0 (/ (cos theta0) (sin theta0))))
      (down (up 0 (/ (cos theta0) (sin theta0)))
            (up (* -1 (cos theta0) (sin theta0)) 0)))
```

Exercise 9.1: Metric Compatibility

The connections constructed from a metric by equation (9.13) are "metric compatible," as described in equation (9.9). Demonstrate that this is true for a literal metric, as described on page 6, in $\mathbf{R}^4$. Your program should produce a zero.

9.2 Metrics and Lagrange Equations

In the Introduction (Chapter 1) we showed that the Lagrange equations for a free particle constrained to a 2-dimensional surface are equivalent to the geodesic equations for motion on that surface. We illustrated that in detail in Section 7.4 for motion on a sphere.

Here we expand this understanding to show that the Christoffel symbols can be derived from the Lagrange equations. Specifically, if we solve the Lagrange equations for the acceleration (the highest-order derivatives) we find that the Christoffel symbols are the symmetrized coefficients of the quadratic velocity terms.

Consider the Lagrange equations for a free particle, with Lagrangian

$$L_2(t, x, v) = \tfrac{1}{2} g(x)(v, v). \tag{9.15}$$

If we solve the Lagrange equations for the accelerations, the accelerations can be expressed with the geodesic equations (7.79):

$$D^2 q^i + \sum_{jk} (\Gamma^i_{jk} \circ \chi^{-1} \circ q) Dq^j Dq^k = 0. \tag{9.16}$$

We can verify this computationally. Given a metric, we can construct a Lagrangian where the kinetic energy is the metric applied to the velocity twice: The kinetic energy is proportional to the squared length of the velocity vector.

```
(define (metric->Lagrangian metric coordsys)
  (define (L state)
    (let ((q (ref state 1)) (qd (ref state 2)))
      (define v
        (components->vector-field (lambda (m) qd) coordsys))
      ((* 1/2 (metric v v)) ((point coordsys) q))))
  L)
```

The following code compares the Christoffel symbols with the coefficients of the terms of second order in velocity appearing in the accelerations, determined by solving the Lagrange equations for the highest-order derivative.[2] We extract these terms by taking two partials with respect to the structure of velocities. Because the elementary partials commute we get two copies of each coefficient, requiring a factor of 1/2.

```
(let* ((metric (literal-metric 'g R3-rect))
       (q (typical-coords R3-rect))
       (L2 (metric->Lagrangian metric R3-rect)))
  (+ (* 1/2
        (((expt (partial 2) 2) (Lagrange-explicit L2))
         (up 't q (corresponding-velocities q))))
     ((Christoffel->symbols
       (metric->Christoffel-2 metric
         (coordinate-system->basis R3-rect)))
      ((point R3-rect) q))))
(down (down (up 0 0 0) (up 0 0 0) (up 0 0 0))
      (down (up 0 0 0) (up 0 0 0) (up 0 0 0))
      (down (up 0 0 0) (up 0 0 0) (up 0 0 0)))
```

We get a structure of zeros, demonstrating the correspondence between Christoffel symbols and coefficients of the Lagrange equations.

Thus, if we have a metric specifying an inner product, the geodesic equations are equivalent to the Lagrange equations for

[2] The procedure `Lagrange-explicit` produces the accelerations of the coordinates. In this code the division operator (`/`) multiplies its first argument on the left by the inverse of its second argument.

```
(define (Lagrange-explicit L)
  (let ((P ((partial 2) L))
        (F ((partial 1) L)))
    (/ (- F (+ ((partial 0) P) (* ((partial 1) P) velocity)))
       ((partial 2) P))))
```

the Lagrangian that is equal to the inner product of the generalized velocities with themselves.

Kinetic Energy or Arc Length

A geodesic is a path of stationary length with respect to variations in the path that keep the endpoints fixed. On the other hand, the solutions of the Lagrange equations are paths of stationary action that keep the endpoints fixed. How are these solutions related?

The integrand of the traditional action is the Lagrangian, which is in this case the Lagrangian L_2, the kinetic energy. The integrand of the arc length is

$$L_1(t, x, v) = \sqrt{g(x)(v, v)} = \sqrt{2L_2(t, x, v)} \tag{9.17}$$

and the path length is

$$\tau = \int_{t_1}^{t_2} L_1(t, q(t), Dq(t))dt. \tag{9.18}$$

If we compute the Lagrange equations for L_2 we get the Lagrange equations for L_1 with a correction term. Since

$$L_2(t, x, v) = \tfrac{1}{2}(L_1(t, x, v))^2 \tag{9.19}$$

And the Lagrange operator for L_2 is[3]

$$\mathbf{E}[L_2] = D_t\partial_2 L_2 - \partial_1 L_2.$$

we find

$$\mathbf{E}[L_2] = L_1\mathbf{E}[L_1] + \partial_2 L_1 D_t L_1. \tag{9.20}$$

L_2 is the kinetic energy. It is conserved along solution paths, since there is no explicit time dependence. Because of the relation between L_1 and L_2, L_1 is also a conserved quantity. Let L_1 take the constant value a on the geodesic coordinate path q we are

[3]$\mathbf{E}$ is the Euler-Lagrange operator, which gives the residuals of the Lagrange equations for a Lagrangian. $\mathbf{\Gamma}$ extends a configuration-space path q to make a state-space path, with as many terms as needed: $\mathbf{\Gamma}[q](t) = (t, q(t), Dq(t), \cdots)$ The total time derivative D_t is defined by $D_t F \circ \mathbf{\Gamma}[q] = D(F \circ \mathbf{\Gamma}[q])$ for any state function F and path q. The Lagrange equations are $\mathbf{E}[L] \circ \Gamma[q] = 0$. See [18] for more details.

considering. Then $\tau = a(t_2 - t_1)$. Since L_1 is conserved, $(D_t L_1) \circ \mathbf{\Gamma}[q] = 0$ on the geodesic path q, and both $\mathbf{E}[L_1] \circ \mathbf{\Gamma}[q] = 0$ and $\mathbf{E}[L_2] \circ \mathbf{\Gamma}[q] = 0$, as required by equation (9.20).

Since L_2 is homogeneous of degree 2 in the velocities, L_1 is homogeneous of degree 1. So we cannot solve for the highest-order derivative in the Lagrange-Euler equations derived from L_1: The Lagrange equations of the Lagrangian L_1 are dependent. But although they do not uniquely specify the evolution, they do specify the geodesic path.

On the other hand, we can solve for the highest-order derivative in $\mathbf{E}[L_2]$. This is because $L_1 \mathbf{E}[L_1]$ is homogeneous of degree 2. So the equations derived from L_2 uniquely determine the time evolution along the geodesic path.

For 2-dimensions

We can show this is true for a 2-dimensional system with a general metric. We define the Lagrangians in terms of this metric:

```
(define L2
  (metric->Lagrangian (literal-metric 'm R2-rect)
                      R2-rect))

(define (L1 state)
  (sqrt (* 2 (L2 state))))
```

Although the mass matrix of L_2 is nonsingular

```
(determinant
 (((partial 2) ((partial 2) L2))
  (up 't (up 'x 'y) (up 'vx 'vy))))
(+ (* (m_00 (up x y)) (m_11 (up x y)))
   (* -1 (expt (m_01 (up x y)) 2)))
```

the mass matrix of L_1 has determinant zero

```
(determinant
 (((partial 2) ((partial 2) L1))
  (up 't (up 'x 'y) (up 'vx 'vy))))
0
```

showing that these Lagrange equations are dependent.

We can show this dependence explicitly, for a simple system. Consider the simplest possible system, a geodesic (straight line) in a plane:

```
(define (L1 state)
  (sqrt (square (velocity state))))

(((Lagrange-equations L1)
  (up (literal-function 'x) (literal-function 'y)))
 't)
(down
 (/ (+ (* (((expt D 2) x) t) (expt ((D y) t) 2))
       (* -1 ((D x) t) ((D y) t) (((expt D 2) y) t)))
    (expt (+ (expt ((D x) t) 2) (expt ((D y) t) 2)) 3/2))
 (/ (+ (* -1 (((expt D 2) x) t) ((D x) t) ((D y) t))
       (* (expt ((D x) t) 2) (((expt D 2) y) t)))
    (expt (+ (expt ((D x) t) 2) (expt ((D y) t) 2)) 3/2)))
```

These residuals must be zero; so the numerators must be zero.[4] They are:

$$D^2x\,(Dy)^2 = Dx\,Dy\,D^2y$$
$$D^2x\,Dx\,Dy = (Dx)^2\,D^2y$$

Note that the only constraint is $D^2x\,Dy = Dx\,D^2y$, so the resulting Lagrange equations are dependent.

This is enough to determine that the result is a straight line, without specifying the rate along the line. Suppose $y = f(x)$, for path $(x(t), y(t))$. Then

$$Dy = Df(x)\,Dx \text{ and } D^2y = D^2f(x)\,Dx + Df(x)\,D^2x.$$

Substituting, we get

$$Df(x)\,Dx\,D^2x = Dx(D^2f(x)\,Dx + Df(x)\,D^2x)$$

or

$$Df(x)\,D^2x = D^2f(x)\,Dx + Df(x)\,D^2x,$$

so $D^2f(x) = 0$. Thus f is a straight line, as required.

Reparameterization

More generally, a differential equation system $F[q](t) = 0$ is said to be *reparameterized* if the coordinate path q is replaced with a

[4]We cheated: We hand-simplified the denominator to make the result more obvious.

new coordinate path $q \circ f$. For example, we may change the scale of the independent variable. The system $F[q \circ f] = 0$ is said to be independent of the parameterization if and only if $F[q] \circ f = 0$. So the differential equation system is satisfied by $q \circ f$ if and only if it is satisfied by q.

The Lagrangian L_1 is homogeneous of degree 1 in the velocities; so

$$\mathbf{E}[L_1] \circ \Gamma[q \circ f] - (\mathbf{E}[L_1] \circ \Gamma[q] \circ f) Df = 0. \tag{9.21}$$

We can check this in a simple case. For two dimensions $q = (x, y)$, the condition under which a reparameterization f of the geodesic paths with coordinates q satisfies the Lagrange equations for L_1 is:

```
(let ((x (literal-function 'x))
      (y (literal-function 'y))
      (f (literal-function 'f))
      (E1 (Euler-Lagrange-operator L1)))
  ((- (compose E1 (Gamma (up (compose x f) (compose y f)) 4))
      (* (compose E1 (Gamma (up x y) 4) f) (D f)))
   't))
(down 0 0)
```

This residual is identically satisfied, showing that the Lagrange equations for L_1 are independent of the parameterization of the independent variable.

The Lagrangian L_2 is homogeneous of degree 2 in the velocities; so

$$\mathbf{E}[L_2][q \circ f] - (\mathbf{E}[L_2][q] \circ f)(Df)^2 = (\partial_2 L_2 \circ \Gamma[q] \circ f)(D^2 f). \tag{9.22}$$

Although the Euler-Lagrange equations for L_1 are invariant under an arbitrary reparameterization ($Df \neq 0$), the Euler-Lagrange equations for L_2 are invariant only for a restricted set of f. The conditions under which a reparameterization f of geodesic paths with coordinates q satisfies the Lagrange equations for L_2 are:

```
(let ((q (up (literal-function 'x) (literal-function 'y)))
      (f (literal-function 'f)))
  ((- (compose (Euler-Lagrange-operator L2)
               (Gamma (compose q f) 4))
      (* (compose (Euler-Lagrange-operator L2) (Gamma q 4) f)
         (expt (D f) 2)))
   't))
(down
 (* (+ (* ((D x) (f t)) (m_00 (up (x (f t)) (y (f t)))))
       (* ((D y) (f t)) (m_01 (up (x (f t)) (y (f t))))))
    (((expt D 2) f) t))
 (* (+ (* ((D x) (f t)) (m_01 (up (x (f t)) (y (f t)))))
       (* ((D y) (f t)) (m_11 (up (x (f t)) (y (f t))))))
    (((expt D 2) f) t)))
```

We see that if these expressions must be zero, then $D^2 f = 0$. This tells us that f is at most affine in t: $f(t) = at + b$.

Exercise 9.2: SO(3) Geodesics

We have derived a basis for SO(3) in terms of incremental rotations around the rectangular axes. See equations (4.29, 4.30, 4.31). We can use the dual basis to define a metric on SO(3).

```
(define (SO3-metric v1 v2)
  (+ (* (e^x v1) (e^x v2))
     (* (e^y v1) (e^y v2))
     (* (e^z v1) (e^z v2))))
```

This metric determines a connection. Show that uniform rotation about an arbitrary axis traces a geodesic on SO(3).

Exercise 9.3: Curvature of a Spherical Surface

The 2-dimensional surface of a 3-dimensional sphere can be embedded in three dimensions with a metric that depends on the radius:

```
(define M (make-manifold S^2-type 2 3))
(define spherical
  (coordinate-system-at 'spherical 'north-pole M))
(define-coordinates (up theta phi) spherical)
(define spherical-basis (coordinate-system->basis spherical))

(define ((spherical-metric r) v1 v2)
  (* (square r)
     (+ (* (dtheta v1) (dtheta v2))
        (* (square (sin theta))
           (dphi v1) (dphi v2)))))
```

If we raise one index of the Ricci tensor (see equation 8.20) by contracting it with the inverse of the metric tensor we can further contract it to obtain a scalar manifold function.

$$R = \sum_{ij} \mathsf{g}(\tilde{\mathsf{e}}^i, \tilde{\mathsf{e}}^j) R(\mathsf{e}^i, \mathsf{e}^j) \tag{9.23}$$

The `trace2down` procedure converts a tensor that takes two vector fields into a tensor that takes a vector field and a one-form field, and then it contracts the result over a basis to make a trace. It is useful for getting the Ricci scalar from the Ricci tensor, given a metric and a basis.

```
(define ((trace2down metric basis) tensor)
  (let ((inverse-metric-tensor
         (metric:invert metric-tensor basis)))
    (contract
     (lambda (v1 w1)
       (contract
        (lambda (v w)
          (* (inverse-metric-tensor w1 w)
             (tensor v v1)))
        basis))
     basis)))
```

Evaluate the Ricci scalar for a sphere of radius r to obtain a measure of its intrinsic curvature. You should obtain the answer $2/r^2$.

Exercise 9.4: Curvature of a Pseudosphere

Compute the scalar curvature of the pseudosphere (see exercise 8.2). You should obtain the value -2.

9.3 Application to Relativity

By analogy to Newtonian mechanics, relativistic mechanics has two parts. There are equations of motion that describe how particles move under the influence of "forces" and there are field equations that describe how the forces arise. In general relativity the only force considered is gravity. However, gravity is not treated as a force. Instead, gravity arises from curvature in the spacetime, and the equations of motion are motion along geodesics of that space.

The geodesic equations for a spacetime with the metric

$$
\begin{aligned}
\mathsf{g}(\mathsf{v}_1, \mathsf{v}_2) = &- c^2 \left(1 + \frac{2V}{c^2}\right) \mathsf{dt}(\mathsf{v}_1)\mathsf{dt}(\mathsf{v}_2) \\
&+ \mathsf{dx}(\mathsf{v}_1)\mathsf{dx}(\mathsf{v}_2) \\
&+ \mathsf{dy}(\mathsf{v}_1)\mathsf{dy}(\mathsf{v}_2) \\
&+ \mathsf{dz}(\mathsf{v}_1)\mathsf{dz}(\mathsf{v}_2)
\end{aligned}
\tag{9.24}
$$

are Newton's equations to lowest order in V/c^2

$$D^2\vec{x}(t) = -\text{grad}V(\vec{x}(t)) \tag{9.25}$$

Exercise 9.5: Newton's Equations

Verify that Newton's equations (9.25) are indeed the lowest-order terms of the geodesic equations for the metric (9.24).

Einstein's field equations tell how the local energy-momentum distribution determines the local shape of the spacetime, as described by the metric tensor g. The equations are traditionally written

$$R_{\mu\nu} - \frac{1}{2}Rg_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4}T_{\mu\nu} \tag{9.26}$$

where $R_{\mu\nu}$ are the components of the Ricci tensor (equation 8.20), R is the Ricci scalar (equation 9.23).[5] and Λ is the cosmological constant.

$T_{\mu\nu}$ are the components of the stress-energy tensor describing the energy-momentum distribution. Equivalently, one can write

$$R_{\mu\nu} = \frac{8\pi G}{c^4}\left(T_{\mu\nu} - \frac{1}{2}Tg_{\mu\nu}\right) - \Lambda g_{\mu\nu} \tag{9.27}$$

where $T = T_{\mu\nu}g^{\mu\nu}$.[6]

[5]The tensor with components $G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2}Rg_{\mu\nu}$ is called the *Einstein tensor*. In his search for an appropriate field equation for gravity, Einstein demanded *general covariance* (independence of coordinate system) and local Lorentz invariance (at each point transformations must preserve the line element). These considerations led Einstein to look for a tensor equation (see Appendix C).

[6]Start with equation (9.26). Raise one index of both sides, and then contract. Notice that the trace $g^\mu_\mu = 4$, the dimension of spacetime. This gets $R = -(8\pi G/c^4)T$, from which we can deduce equation (9.27).

Einstein's field equations arise from a heuristic derivation by analogy to the Poisson equation for a Newtonian gravitational field:

$$\mathrm{Lap}(V) = 4\pi G\rho \tag{9.28}$$

where V is the gravitational potential field at a point, ρ is the mass density at that point, and Lap is the Laplacian operator.

The time-time component of the Ricci tensor derived from the metric (9.24) is the Laplacian of the potential, to lowest order.

```
(define (Newton-metric M G c V)
  (let ((a
         (+ 1 (* (/ 2 (square c))
                 (compose V (up x y z))))))
    (define (g v1 v2)
      (+ (*  -1 (square c) a (dt v1) (dt v2))
         (* (dx v1) (dx v2))
         (* (dy v1) (dy v2))
         (* (dz v1) (dz v2))))
    g))

(define (Newton-connection M G c V)
  (Christoffel->Cartan
   (metric->Christoffel-2 (Newton-metric M G c V)
                          spacetime-rect-basis)))

(define nabla
  (covariant-derivative
    (Newton-connection 'M 'G ':c
      (literal-function 'V (-> (UP Real Real Real) Real)))))

(((Ricci nabla (coordinate-system->basis spacetime-rect))
  d/dt d/dt)
 ((point spacetime-rect) (up 't 'x 'y 'z)))
mess
```

The leading terms of the mess are

```
(+ (((partial 0) ((partial 0) V)) (up x y z))
   (((partial 1) ((partial 1) V)) (up x y z))
   (((partial 2) ((partial 2) V)) (up x y z)))
```

which is the Laplacian of V. The other terms are smaller by V/c^2.

Now consider the right-hand side of equation (9.27). In the Poisson equation the source of the gravitational potential is the density of matter. Let the time-time component of the stress-energy tensor T^{00} be the matter density ρ. Here is a program for the stress-energy tensor:

```
(define (Tdust rho)
  (define (T w1 w2)
    (* rho (w1 d/dt) (w2 d/dt)))
  T)
```

If we evaluate the right-hand side expression we obtain[7]

```
(let ((g (Newton-metric 'M 'G ':c V)))
  (let ((T_ij ((drop2 g spacetime-rect-basis) (Tdust 'rho))))
    (let ((T ((trace2down g spacetime-rect-basis) T_ij)))
      ((- (T_ij d/dt d/dt) (* 1/2 T (g d/dt d/dt)))
       ((point spacetime-rect) (up 't 'x 'y 'z))))))
(* 1/2 (expt :c 4) rho)
```

So, to make the Poisson analogy we get

$$R_{\mu\nu} = \frac{8\pi G}{c^4}\left(T_{\mu\nu} - \frac{1}{2}T g_{\mu\nu}\right) - \Lambda g_{\mu\nu} \tag{9.29}$$

as required.

Exercise 9.6: Curvature of Schwarzschild Spacetime

In spherical coordinates around a nonrotating gravitating body the metric of Schwarzschild spacetime is given as:[8]

[7]The procedure `trace2down` is defined on page 144. This expression also uses `drop2`, which converts a tensor field that takes two one-form fields into a tensor field that takes two vector fields. Its definition is

```
(define ((drop2 metric-tensor basis) tensor)
  (lambda (v1 v2)
    (contract
     (lambda (e1 w1)
       (contract
        (lambda (e2 w2)
          (* (metric-tensor v1 e1)
             (tensor w1 w2)
             (metric-tensor e2 v2)))
        basis))
     basis)))
```

[8]The spacetime manifold is built from $\mathbf{R}^4$ with the addition of appropriate coordinate systems:

```
(define-coordinates (up t r theta phi) spacetime-sphere)

(define (Schwarzschild-metric M G c)
  (let ((a (- 1 (/ (* 2 G M) (* (square c) r)))))
    (lambda (v1 v2)
      (+ (*  -1 (square c) a (dt v1) (dt v2))
         (* (/ 1 a) (dr v1) (dr v2))
         (* (square r)
            (+ (* (dtheta v1) (dtheta v2))
               (* (square (sin theta))
                  (dphi v1) (dphi v2))))))))
```

Show that the Ricci curvature of the Schwarzschild spacetime is zero. Use the definition of the Ricci tensor in equation (8.20).

Exercise 9.7: Circular Orbits in Schwarzschild Spacetime

Test particles move along geodesics in spacetime. Now that we have a metric for Schwarzschild spacetime (page 147) we can use it to construct the geodesic equations and determine how test particles move. Consider circular orbits. For example, the circular orbit along a line of constant longitude is a geodesic, so it should satisfy the geodesic equations. Here is the equation of a circular path along the zero longitude line.

```
(define (prime-meridian r omega)
  (compose
   (point spacetime-sphere)
   (lambda (t) (up t r (* omega t) 0))
   (chart R1-rect)))
```

This equation will satisfy the geodesic equations for compatible values of the radius `r` and the angular velocity `omega`. If you substitute this into the geodesic equation and set the residual to zero you will obtain a constraint relating `r` and `omega`. Do it.

Surprise: You should find out that $\omega^2 r^3 = GM$—Kepler's law!

Exercise 9.8: Stability of Circular Orbits

In Schwarzschild spacetime there are stable circular orbits if the coordinate r is large enough, but below that value all orbits are unstable. The critical value of r is larger than the Schwarzschild horizon radius. Let's find that value.

```
(define spacetime (make-manifold R^n 4))
(define spacetime-rect
  (coordinate-system-at 'rectangular 'origin spacetime))
(define spacetime-sphere
  (coordinate-system-at 'spacetime-spherical 'origin spacetime))
```

For example, we can consider a perturbation of the orbit of constant longitude. Here is the result of adding an exponential variation of size `epsilon`:

```
(define (prime-meridian+X r epsilon X)
  (compose
   (point spacetime-sphere)
   (lambda (t)
     (up (+ t (* epsilon (* (ref X 0) (exp (* 'lambda t)))))
         (+ r (* epsilon (* (ref X 1) (exp (* 'lambda t)))))
         (+ (* (sqrt (/ (* 'G 'M) (expt r 3))) t)
            (* epsilon (* (ref X 2) (exp (* 'lambda t)))))
         0))
   (chart R1-rect)))
```

Plugging this into the geodesic equation yields a structure of residuals:

```
(define (geodesic-equation+X-residuals eps X)
  (let ((gamma (prime-meridian+X 'r eps X)))
    (((((covariant-derivative Cartan gamma) d/dtau)
       ((differential gamma) d/dtau))
      (chart spacetime-sphere))
     ((point R1-rect) 't))))
```

The characteristic equation in the eigenvalue `lambda` can be obtained as the numerator of the expression:

```
(determinant
 (submatrix (((* (partial 1) (partial 0))
              geodesic-equation+X-residuals)
             0
             (up 0 0 0))
            0 3 0 3))
```

Show that the orbits are unstable if $r < 6GM/c^2$.

Exercise 9.9: Friedmann-Lemaître-Robertson-Walker

The Einstein tensor $G_{\mu\nu}$ (see footnote 5) can be expressed as a program:

```
(define (Einstein coordinate-system metric-tensor)
  (let* ((basis (coordinate-system->basis coordinate-system))
         (connection
          (Christoffel->Cartan
           (metric->Christoffel-2 metric-tensor basis)))
         (nabla (covariant-derivative connection))
         (Ricci-tensor (Ricci nabla basis))
         (Ricci-scalar
          ((trace2down metric-tensor basis) Ricci-tensor)) )
    (define (Einstein-tensor v1 v2)
      (- (Ricci-tensor v1 v2)
         (* 1/2 Ricci-scalar (metric-tensor v1 v2))))
    Einstein-tensor))
```

```
(define (Einstein-field-equation
          coordinate-system metric-tensor Lambda stress-energy-tensor)
  (let ((Einstein-tensor
         (Einstein coordinate-system metric-tensor)))
    (define EFE-residuals
      (- (+ Einstein-tensor (* Lambda metric-tensor))
         (* (/ (* 8 :pi :G) (expt :c 4))
            stress-energy-tensor)))
    EFE-residuals))
```

One exact solution to the Einstein equations was found by Alexander Friedmann in 1922. He showed that a metric for an isotropic and homogeneous spacetime was consistent with a similarly isotropic and homogeneous stress-energy tensor in Einstein's equations. In this case the residuals of the Einstein equations gave ordinary differential equations for the time-dependent scale of the universe. These are called the Robertson-Walker equations. Friedmann's metric is:

```
(define (FLRW-metric c k R)
  (define-coordinates (up t r theta phi) spacetime-sphere)
  (let ((a (/ (square (compose R t)) (- 1 (* k (square r)))))
        (b (square (* (compose R t) r))))
    (define (g v1 v2)
      (+ (*  -1 (square c) (dt v1) (dt v2))
         (* a (dr v1) (dr v2))
         (* b (+ (* (dtheta v1) (dtheta v2))
                 (* (square (sin theta))
                    (dphi v1) (dphi v2))))))
    g))
```

Here `c` is the speed of light, `k` is the intrinsic curvature, and `R` is a length scale that is a function of time.

The associated stress-energy tensor is

```
(define (Tperfect-fluid rho p c metric)
  (define-coordinates (up t r theta phi) spacetime-sphere)
  (let* ((basis (coordinate-system->basis spacetime-sphere))
         (inverse-metric (metric:invert metric basis)))
    (define (T w1 w2)
      (+ (* (+ (compose rho t)
               (/ (compose p t) (square c)))
            (w1 d/dt) (w2 d/dt))
         (* (compose p t) (inverse-metric w1 w2))))
    T))
```

where `rho` is the energy density, and `p` is the pressure in an ideal fluid model.

The Robertson-Walker equations are:

$$\left(\frac{DR(t)}{R(t)}\right)^2 + \frac{kc^2}{(R(t))^2} - \frac{\Lambda c^2}{3} = \frac{8\pi G}{3}\rho(t)$$
$$2\frac{D^2R(t)}{R(t)} - \frac{2}{3}\Lambda c^2 = -8\pi G\left(\frac{\rho(t)}{3} + \frac{p(t)}{c^2}\right) \tag{9.30}$$

Use the programs supplied to derive the Robertson-Walker equations.

Exercise 9.10: Cosmology

For energy to be conserved, the stress-energy tensor must be constrained so that its covariant divergence is zero

$$\sum_\mu \nabla_{\mathsf{e}_\mu} \mathsf{T}(\tilde{\mathsf{e}}^\mu, \omega) = 0 \tag{9.31}$$

for every one-form ω.

a. Show that for the perfect fluid stress-energy tensor and the FLRW metric this constraint is equivalent to the differential equation

$$D(c^2\rho R^3) + pD(R^3) = 0. \tag{9.32}$$

b. Assume that in a “matter-dominated universe” radiation pressure is negligible, so $p = 0$. Using the Robertson-Walker equations (9.30) and the energy conservation equation (9.32) show that the observation of an expanding universe is compatible with a negative curvature universe, a flat universe, or a positive curvature universe: $k \in \{-1, 0, +1\}$.

10
Hodge Star and Electrodynamics

The vector space of p-form fields on an n-dimensional manifold has dimension $n!/((n-p)!p!)$. This is the same dimension as the space of $(n-p)$-form fields. So these vector spaces are isomorphic. If we have a metric there is a natural isomorphism; for each p-form field $\boldsymbol{\omega}$ on an n-dimensional manifold with a metric there is an $(n-p)$-form field $\mathsf{g}^*\boldsymbol{\omega}$, called its *Hodge dual*.[1] The Hodge dual should not be confused with the duality of vector bases and one-form bases, which is defined without reference to a metric. The Hodge dual is useful for the elegant formalization of electrodynamics.

In Euclidean 3-space, if we think of a one-form as a foliation of the space, then the dual is a two-form, which can be thought of as a pack of square tubes, whose axes are perpendicular to the leaves of the foliation. The original one-form divides these tubes up into volume elements. For example, the dual of the basis one-form $\mathsf{d}x$ is the two-form $\mathsf{g}^*\mathsf{d}x = \mathsf{d}y \wedge \mathsf{d}z$. We may think of $\mathsf{d}x$ as a set of planes perpendicular to the $\hat{x}$-axis. Then $\mathsf{g}^*\mathsf{d}x$ is a set of tubes parallel to the $\hat{x}$-axis. In higher-dimensional spaces the visualization is more complicated, but the basic idea is the same. The Hodge dual of a two-form in four dimensions is a two-form that is perpendicular to the given two-form. However, if the metric is indefinite (e.g., the Lorentz metric) there is an added complication with the signs.

The Hodge dual is a linear operator, so it can be defined by its action on the basis elements. Let $\{\partial/\partial\mathsf{x}^0, \ldots, \partial/\partial\mathsf{x}^{n-1}\}$ be an orthonormal basis of vector fields[2] and let $\{\mathsf{d}x^0, \ldots, \mathsf{d}x^{n-1}\}$ be the ordinary dual basis for the one-forms. Then the $(n-p)$-form $\mathsf{g}^*\boldsymbol{\omega}$ that is the Hodge dual of the p-form $\boldsymbol{\omega}$ can be defined by its coefficients with respect to the basis, using indices, as

$$(\mathsf{g}^*\boldsymbol{\omega})_{j_p \ldots j_{n-1}} = \sum_{i_0 \ldots i_{p-1} j_0 \ldots j_{p-1}} \frac{1}{p!} \omega_{i_0 \ldots i_{p-1}} g^{i_0 j_0} \ldots g^{i_{p-1} j_{p-1}} \epsilon_{j_0 \ldots j_{n-1}}, \tag{10.1}$$

[1]The traditional notion is to just use an asterisk; we use g^* to emphasize that this duality depends on the choice of metric g.

[2]We have a metric, so we can define "orthonormal" and use it to construct an orthonormal basis given any basis. The Gram-Schmidt procedure does the job.

where g^{ij} are the coefficients of the inverse metric and $\epsilon_{j_0 \ldots j_{n-1}}$ is either -1 or $+1$ if the permutation $\{0 \ldots n-1\} \to \{j_0 \ldots j_{n-1}\}$ is odd or even, respectively.

Relationship to Vector Calculus

In 3-dimensional Euclidean space the traditional vector derivative operations are gradient, curl, and divergence. If $\hat{\mathrm{x}}, \hat{\mathrm{y}}, \hat{\mathrm{z}}$ are the usual orthonormal rectangular vector basis, f a function on the space, and $\vec{\mathrm{v}}$ a vector field on the space, then

$$\begin{aligned}
\mathrm{grad}(f) &= \frac{\partial f}{\partial x}\hat{\mathrm{x}} + \frac{\partial f}{\partial y}\hat{\mathrm{y}} + \frac{\partial f}{\partial z}\hat{\mathrm{z}} \\
\mathrm{curl}(\vec{\mathrm{v}}) &= \left(\frac{\partial v_z}{\partial y} - \frac{\partial v_y}{\partial z}\right)\hat{\mathrm{x}} + \left(\frac{\partial v_x}{\partial z} - \frac{\partial v_z}{\partial x}\right)\hat{\mathrm{y}} + \left(\frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y}\right)\hat{\mathrm{x}} \\
\mathrm{div}(\vec{\mathrm{v}}) &= \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}
\end{aligned}$$

Recall the meaning of the traditional vector operations. Traditionally we assume that there is a metric that allows us to determine distances between locations and angles between vectors. Such a metric establishes local scale factors relating coordinate increments to actual distances. The vector gradient, grad(f), points in the direction of steepest increase in the function with respect to actual distances. By contrast, the gradient one-form, df, does not depend on a metric, so there is no concept of distance built in to it. Nevertheless, the concepts are related. The gradient one-form is given by

$$\mathsf{df} = \left(\frac{\partial}{\partial \mathsf{x}}\mathsf{f}\right)\mathsf{dx} + \left(\frac{\partial}{\partial \mathsf{y}}\mathsf{f}\right)\mathsf{dy} + \left(\frac{\partial}{\partial \mathsf{z}}\mathsf{f}\right)\mathsf{dz}. \tag{10.2}$$

The traditional gradient vector field is then just the raised gradient one-form (see equation 9.8). So

$$\mathrm{grad}(\mathsf{f}) = \mathsf{g}^\sharp(\mathsf{df}) \tag{10.3}$$

computed by

```
(define (gradient metric basis)
  (compose (raise metric basis) d))
```

Let θ be a one-form field:

$$\theta = \theta_x \mathsf{dx} + \theta_y \mathsf{dy} + \theta_z \mathsf{dz} \tag{10.4}$$

We compute

$$\begin{aligned}\mathsf{d}\theta = &\left(\frac{\partial\theta_z}{\partial \mathsf{y}} - \frac{\partial\theta_y}{\partial \mathsf{z}}\right)\mathsf{dy}\wedge\mathsf{dz} + \left(\frac{\partial\theta_x}{\partial \mathsf{z}} - \frac{\partial\theta_z}{\partial \mathsf{x}}\right)\mathsf{dz}\wedge\mathsf{dx} \\ &+ \left(\frac{\partial\theta_y}{\partial \mathsf{x}} - \frac{\partial\theta_x}{\partial \mathsf{y}}\right)\mathsf{dx}\wedge\mathsf{dy}.\end{aligned} \tag{10.5}$$

So the exterior-derivative expression corresponding to the vector-calculus curl is:

$$\begin{aligned}\mathsf{g}^*(\mathsf{d}\theta) = &\left(\frac{\partial\theta_z}{\partial \mathsf{y}} - \frac{\partial\theta_y}{\partial \mathsf{z}}\right)\mathsf{dx} + \left(\frac{\partial\theta_x}{\partial \mathsf{z}} - \frac{\partial\theta_z}{\partial \mathsf{x}}\right)\mathsf{dy} \\ &+ \left(\frac{\partial\theta_y}{\partial \mathsf{x}} - \frac{\partial\theta_x}{\partial \mathsf{y}}\right)\mathsf{dz}\end{aligned} \tag{10.6}$$

Thus, the curl of a vector field v is

$$\mathrm{curl}(\mathsf{v}) = \mathsf{g}^\sharp(\mathsf{g}^*(\mathsf{d}(\mathsf{g}^\flat(\mathsf{v})))), \tag{10.7}$$

which can be computed with

```
(define (curl metric orthonormal-basis)
  (let ((star (Hodge-star metric orthonormal-basis))
        (sharp (raise metric orthonormal-basis))
        (flat (lower metric)))
    (compose sharp star d flat)))
```

Also, we compute

$$\mathsf{d}(\mathsf{g}^*\theta) = \left(\frac{\partial\theta_x}{\partial \mathsf{x}} + \frac{\partial\theta_y}{\partial \mathsf{y}} + \frac{\partial\theta_z}{\partial \mathsf{z}}\right)\mathsf{dx}\wedge\mathsf{dy}\wedge\mathsf{dz}. \tag{10.8}$$

So the exterior-derivative expression corresponding to the vector-calculus div is

$$\mathsf{g}^*\mathsf{d}(\mathsf{g}^*\theta) = \frac{\partial\theta_x}{\partial \mathsf{x}} + \frac{\partial\theta_y}{\partial \mathsf{y}} + \frac{\partial\theta_z}{\partial \mathsf{z}}. \tag{10.9}$$

Thus, the divergence of a vector field v is

$$\mathrm{div}(\mathsf{v}) = \mathsf{g}^*(\mathsf{d}(\mathsf{g}^*(\mathsf{g}^\flat(\mathsf{v})))). \tag{10.10}$$

It is easily computed:

```
(define (divergence metric orthonormal-basis)
  (let ((star (Hodge-star metric orthonormal-basis))
        (flat (lower metric)))
    (compose star d star flat)))
```

The divergence is defined even if we don't have a metric, but have only a connection. In that case the divergence can be computed with

```
(define (((divergence Cartan) v) point)
  (let ((basis (Cartan->basis Cartan))
        (nabla (covariant-derivative Cartan)))
    (contract
     (lambda (ei wi)
       ((wi ((nabla ei) v)) point))
     basis)))
```

If the Cartan form is derived from a metric these programs yield the same answer.

The Laplacian is, as expected, the composition of the divergence and the gradient:

```
(define (Laplacian metric orthonormal-basis)
  (compose (divergence metric orthonormal-basis)
           (gradient metric orthonormal-basis)))
```

Spherical Coordinates

We can illustrate these by computing the formulas for the vector-calculus operators in spherical coordinates. We start with a 3-dimensional manifold, and we set up the conditions for spherical coordinates.

```
(define spherical R3-rect)
(define-coordinates (up r theta phi) spherical)

(define R3-spherical-point
  ((point spherical) (up 'r0 'theta0 'phi0)))
```

The geometry is specified by the metric:

```
(define (spherical-metric v1 v2)
  (+ (* (dr v1) (dr v2))
     (* (square r)
        (+ (* (dtheta v1) (dtheta v2))
           (* (expt (sin theta) 2)
              (dphi v1) (dphi v2))))))
```

We also need an orthonormal basis for the spherical coordinates. The coordinate basis is orthogonal but not normalized.

```
(define e_0 d/dr)
(define e_1 (* (/ 1 r) d/dtheta))
(define e_2 (* (/ 1 (* r (sin theta))) d/dphi))

(define orthonormal-spherical-vector-basis
  (down e_0 e_1 e_2))

(define orthonormal-spherical-1form-basis
  (vector-basis->dual orthonormal-spherical-vector-basis
                      spherical))

(define orthonormal-spherical-basis
  (make-basis orthonormal-spherical-vector-basis
              orthonormal-spherical-1form-basis))
```

The components of the gradient of a scalar field are obtained using the dual basis:

```
((orthonormal-spherical-1form-basis
  ((gradient spherical-metric orthonormal-spherical-basis)
   (literal-manifold-function 'f spherical)))
 R3-spherical-point)
(up (((partial 0) f) (up r0 theta0 phi0))
    (/ (((partial 1) f) (up r0 theta0 phi0))
       r0)
    (/ (((partial 2) f) (up r0 theta0 phi0))
       (* r0 (sin theta0))))
```

To get the formulas for curl and divergence we need a vector field with components with respect to the normalized basis.

```
(define v
  (+ (* (literal-manifold-function 'v^0 spherical) e_0)
     (* (literal-manifold-function 'v^1 spherical) e_1)
     (* (literal-manifold-function 'v^2 spherical) e_2)))
```

The curl is a bit complicated:

```
((orthonormal-spherical-1form-basis
  ((curl spherical-metric orthonormal-spherical-basis) v))
 R3-spherical-point)
(up
 (/ (+ (* (sin theta0)
          (((partial 1) v^2) (up r0 theta0 phi0)))
       (* (cos theta0) (v^2 (up r0 theta0 phi0)))
       (* -1 (((partial 2) v^1) (up r0 theta0 phi0))))
    (* r0 (sin theta0)))
 (/ (+ (* -1 r0 (sin theta0)
          (((partial 0) v^2) (up r0 theta0 phi0)))
       (* -1 (sin theta0) (v^2 (up r0 theta0 phi0)))
       (((partial 2) v^0) (up r0 theta0 phi0)))
    (* r0 (sin theta0)))
 (/ (+ (* r0 (((partial 0) v^1) (up r0 theta0 phi0)))
       (v^1 (up r0 theta0 phi0))
       (* -1 (((partial 1) v^0) (up r0 theta0 phi0))))
    r0))
```

But the divergence and Laplacian are simpler

```
(((divergence spherical-metric orthonormal-spherical-basis) v)
 R3-spherical-point)
(+ (((partial 0) v^0) (up r0 theta0 phi0))
   (/ (* 2 (v^0 (up r0 theta0 phi0))) r0)
   (/ (((partial 1) v^1) (up r0 theta0 phi0)) r0)
   (/ (* (v^1 (up r0 theta0 phi0)) (cos theta0))
      (* r0 (sin theta0)))
   (/ (((partial 2) v^2) (up r0 theta0 phi0))
      (* r0 (sin theta0))))
```

```
(((Laplacian spherical-metric orthonormal-spherical-basis)
  (literal-manifold-function 'f spherical))
 R3-spherical-point)
(+ (((partial 0) ((partial 0) f)) (up r0 theta0 phi0))
   (/ (* 2 (((partial 0) f) (up r0 theta0 phi0)))
      r0)
   (/ (((partial 1) ((partial 1) f)) (up r0 theta0 phi0))
      (expt r0 2))
   (/ (* (cos theta0) (((partial 1) f) (up r0 theta0 phi0)))
      (* (expt r0 2) (sin theta0)))
   (/ (((partial 2) ((partial 2) f)) (up r0 theta0 phi0))
      (* (expt r0 2) (expt (sin theta0) 2))))
```

10.1 The Wave Equation

The kinematics of special relativity can be formulated on a flat 4-dimensional spacetime manifold.

```
(define SR R4-rect)
(define-coordinates (up ct x y z) SR)
(define an-event ((point SR) (up 'ct0 'x0 'y0 'z0)))

(define a-vector
  (+ (* (literal-manifold-function 'v^t SR) d/dct)
     (* (literal-manifold-function 'v^x SR) d/dx)
     (* (literal-manifold-function 'v^y SR) d/dy)
     (* (literal-manifold-function 'v^z SR) d/dz)))
```

The Minkowski metric is[3]

$$\mathsf{g}(\mathsf{u},\mathsf{v}) = -c^2 \mathsf{dt}(\mathsf{u})\,\mathsf{dt}(\mathsf{v}) + \mathsf{dx}(\mathsf{u})\,\mathsf{dx}(\mathsf{v}) + \mathsf{dy}(\mathsf{u})\,\mathsf{dy}(\mathsf{v}) + \mathsf{dz}(\mathsf{u})\,\mathsf{dz}(\mathsf{v}). \tag{10.11}$$

As a program:

```
(define (g-Minkowski u v)
  (+ (* -1 (dct u) (dct v))
     (* (dx u) (dx v))
     (* (dy u) (dy v))
     (* (dz u) (dz v))))
```

The length of a vector is described in terms of the metric:

$$\sigma = \mathsf{g}(\mathsf{v},\mathsf{v}). \tag{10.12}$$

If σ is positive the vector is *spacelike* and its square root is the *proper length* of the vector. If σ is negative the vector is *timelike* and the square root of its negation is the *proper time* of the vector. If σ is zero the vector is *lightlike* or *null*.

```
((g-Minkowski a-vector a-vector) an-event)
(+ (* -1 (expt (v^t (up ct0 x0 y0 z0)) 2))
   (expt (v^x (up ct0 x0 y0 z0)) 2)
   (expt (v^y (up ct0 x0 y0 z0)) 2)
   (expt (v^z (up ct0 x0 y0 z0)) 2))
```

[3]The metric in relativity is not positive definite, so nonzero vectors can have zero length.

As an example of vector calculus in four dimensions, we can compute the wave equation for a scalar field in 4-dimensional spacetime.

We need an orthonormal basis for the spacetime:

```
(define SR-vector-basis (coordinate-system->vector-basis SR))
```

We check that it is orthonormal with respect to the metric:

```
((g-Minkowski SR-vector-basis SR-vector-basis) an-event)
(down (down -1 0 0 0)
      (down  0 1 0 0)
      (down  0 0 1 0)
      (down  0 0 0 1))
```

So, the Laplacian of a scalar field is the wave equation!

```
(define p (literal-manifold-function 'phi SR))

(((Laplacian g-Minkowski SR-basis) p) an-event)
(+ (((partial 0) ((partial 0) phi)) (up ct0 x0 y0 z0))
   (* -1 (((partial 1) ((partial 1) phi)) (up ct0 x0 y0 z0)))
   (* -1 (((partial 2) ((partial 2) phi)) (up ct0 x0 y0 z0)))
   (* -1 (((partial 3) ((partial 3) phi)) (up ct0 x0 y0 z0))))
```

10.2 Electrodynamics

Using Hodge duals we can represent electrodynamics in an elegant way. Maxwell's electrodynamics is invariant under Lorentz transformations. We use 4-dimensional rectangular coordinates for the flat spacetime of special relativity.

In this formulation of electrodynamics the electric and magnetic fields are represented together as a two-form field, the *Faraday tensor*. Under Lorentz transformations the individual components are mixed. The Faraday tensor is:[4]

```
(define (Faraday Ex Ey Ez Bx By Bz)
  (+ (* Ex (wedge dx dct))
     (* Ey (wedge dy dct))
     (* Ez (wedge dz dct))
     (* Bx (wedge dy dz))
     (* By (wedge dz dx))
     (* Bz (wedge dx dy))))
```

[4]This representation is from Misner, Thorne, and Wheeler, *Gravitation*, p.108.

The Hodge dual of the Faraday tensor exchanges the electric and magnetic fields, negating the components that will involve time. The result is called the *Maxwell tensor*:

```
(define (Maxwell Ex Ey Ez Bx By Bz)
  (+ (* -1 Bx (wedge dx dct))
     (* -1 By (wedge dy dct))
     (* -1 Bz (wedge dz dct))
     (* Ex (wedge dy dz))
     (* Ey (wedge dz dx))
     (* Ez (wedge dx dy))))
```

We make a Hodge dual operator for this situation

```
(define SR-star (Hodge-star g-Minkowski SR-basis))
```

and indeed, it transforms the Faraday tensor into the Maxwell tensor:

```
(((- (SR-star (Faraday 'Ex 'Ey 'Ez 'Bx 'By 'Bz))
     (Maxwell 'Ex 'Ey 'Ez 'Bx 'By 'Bz))
  (literal-vector-field 'u SR)
  (literal-vector-field 'v SR))
 an-event)
0
```

One way to get electric fields is to have charges; magnetic fields can arise from motion of charges. In this formulation we combine the charge density and the current to make a one-form field:

```
(define (J charge-density Ix Iy Iz)
  (- (* (/ 1 :c) (+ (* Ix dx) (* Iy dy) (* Iz dz)))
     (* charge-density dct)))
```

The coefficient `(/ 1 :c)` makes the components of the one-form uniform with respect to units.

To develop Maxwell's equations we need a general Faraday field and a general current-density field:

```
(define F
  (Faraday (literal-manifold-function 'Ex SR)
           (literal-manifold-function 'Ey SR)
           (literal-manifold-function 'Ez SR)
           (literal-manifold-function 'Bx SR)
           (literal-manifold-function 'By SR)
           (literal-manifold-function 'Bz SR)))
```

```
(define 4-current
  (J (literal-manifold-function 'rho SR)
     (literal-manifold-function 'Ix SR)
     (literal-manifold-function 'Iy SR)
     (literal-manifold-function 'Iz SR)))
```

Maxwell's Equations

Maxwell's equations in the language of differential forms are

$$\mathsf{dF} = 0 \tag{10.13}$$

$$\mathsf{d}(\mathsf{g}^* \, \mathsf{F}) = 4\pi \, \mathsf{g}^* \, \mathsf{J}. \tag{10.14}$$

The first equation gives us what would be written in vector notation as

$$\mathrm{div}\vec{B} = 0, \tag{10.15}$$

and

$$\mathrm{curl}\vec{E} = -\frac{1}{c}\frac{d\vec{B}}{dt}. \tag{10.16}$$

The second equation gives us what would be written in vector notation as

$$\mathrm{div}\vec{E} = 4\pi\rho, \tag{10.17}$$

and

$$\mathrm{curl}\vec{B} = \frac{1}{c}\frac{d\vec{E}}{dt} + \frac{4\pi}{c}\vec{I}. \tag{10.18}$$

To see how these work out, we evaluate each component of dF and $\mathsf{d}\,(\mathsf{g}^*\mathsf{F}) - 4\pi\,\mathsf{g}^*\,\mathsf{J}$. Since these are both two-form fields, their exterior derivatives are three-form fields, so we have to provide three basis vectors to get each component. Each component equation will yield one of Maxwell's equations, written in coordinates, without vector notation. So, the purely spatial component $\mathsf{dF}(\partial/\partial x, \partial/\partial y, \partial/\partial z)$ of equation 10.13 is equation 10.15:

```
(((d F) d/dx d/dy d/dz) an-event)
(+ (((partial 1) Bx) (up ct0 x0 y0 z0))
   (((partial 2) By) (up ct0 x0 y0 z0))
   (((partial 3) Bz) (up ct0 x0 y0 z0)))
```

$$\frac{\partial B_x}{\partial x} + \frac{\partial B_y}{\partial y} + \frac{\partial B_z}{\partial z} = 0 \tag{10.19}$$

The three mixed space and time components of equation 10.13 are equation 10.16:

```
(((d F) d/dct d/dy d/dz) an-event)
(+ (((partial 0) Bx) (up ct0 x0 y0 z0))
   (((partial 2) Ez) (up ct0 x0 y0 z0))
   (* -1 (((partial 3) Ey) (up ct0 x0 y0 z0))))
```

$$\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = \frac{1}{c}\frac{\partial B_x}{\partial t} \tag{10.20}$$

```
(((d F) d/dct d/dz d/dx) an-event)
(+ (((partial 0) By) (up ct0 x0 y0 z0))
   (((partial 3) Ex) (up ct0 x0 y0 z0))
   (* -1 (((partial 1) Ez) (up ct0 x0 y0 z0))))
```

$$\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} = \frac{1}{c}\frac{\partial B_y}{\partial t} \tag{10.21}$$

```
(((d F) d/dct d/dx d/dy) an-event)
(+ (((partial 0) Bz) (up ct0 x0 y0 z0))
   (((partial 1) Ey) (up ct0 x0 y0 z0))
   (* -1 (((partial 2) Ex) (up ct0 x0 y0 z0))))
```

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = \frac{1}{c}\frac{\partial B_z}{\partial t} \tag{10.22}$$

The purely spatial component of equation 10.14 is equation 10.17:

```
(((- (d (SR-star F)) (* 4 :pi (SR-star 4-current)))
  d/dx d/dy d/dz)
 an-event)
(+ (* -4 :pi (rho (up ct0 x0 y0 z0)))
   (((partial 1) Ex) (up ct0 x0 y0 z0))
   (((partial 2) Ey) (up ct0 x0 y0 z0))
   (((partial 3) Ez) (up ct0 x0 y0 z0)))
```

$$\frac{\partial E_x}{\partial x} + \frac{\partial E_y}{\partial y} + \frac{\partial E_z}{\partial z} = 4\pi\rho \tag{10.23}$$

And finally, the three mixed time and space components of equation 10.14 are equation 10.18:

```
(((- (d (SR-star F)) (* 4 :pi (SR-star 4-current)))
  d/dct d/dy d/dz)
 an-event)
(+ (((partial 0) Ex) (up ct0 x0 y0 z0))
   (* -1 (((partial 2) Bz) (up ct0 x0 y0 z0)))
   (((partial 3) By) (up ct0 x0 y0 z0))
   (/ (* 4 :pi (Ix (up ct0 x0 y0 z0))) :c))
```

$$\frac{\partial B_y}{\partial z} - \frac{\partial B_z}{\partial y} = -\frac{1}{c}\frac{\partial E_x}{\partial t} - \frac{4\pi}{c} I_x \tag{10.24}$$

```
(((- (d (SR-star F)) (* 4 :pi (SR-star 4-current)))
  d/dct d/dz d/dx)
 an-event)
(+ (((partial 0) Ey) (up ct0 x0 y0 z0))
   (* -1 (((partial 3) Bx) (up ct0 x0 y0 z0)))
   (((partial 1) Bz) (up ct0 x0 y0 z0))
   (/ (* 4 :pi (Iy (up ct0 x0 y0 z0))) :c))
```

$$\frac{\partial B_z}{\partial x} - \frac{\partial B_x}{\partial z} = -\frac{1}{c}\frac{\partial E_y}{\partial t} - \frac{4\pi}{c} I_y \tag{10.25}$$

```
(((- (d (SR-star F)) (* 4 :pi (SR-star 4-current)))
  d/dct d/dx d/dy)
 an-event)
(+ (((partial 0) Ez) (up ct0 x0 y0 z0))
   (* -1 (((partial 1) By) (up ct0 x0 y0 z0)))
   (((partial 2) Bx) (up ct0 x0 y0 z0))
   (/ (* 4 :pi (Iz (up ct0 x0 y0 z0))) :c))
```

$$\frac{\partial B_x}{\partial y} - \frac{\partial B_y}{\partial x} = -\frac{1}{c}\frac{\partial E_z}{\partial t} - \frac{4\pi}{c} I_z \tag{10.26}$$

Lorentz Force

The classical force on a charged particle moving in a electromagnetic field is

$$\vec{f} = q\left(\vec{E} + \frac{1}{c}\vec{v} \times \vec{B}\right) \tag{10.27}$$

We can compute this in coordinates. We construct arbitrary $\vec{E}$ and $\vec{B}$ vector fields and an arbitrary velocity:

```
(define E
  (up (literal-manifold-function 'Ex SR)
      (literal-manifold-function 'Ey SR)
      (literal-manifold-function 'Ez SR)))

(define B
  (up (literal-manifold-function 'Bx SR)
      (literal-manifold-function 'By SR)
      (literal-manifold-function 'Bz SR)))

(define V (up 'V_x 'V_y 'V_z))
```

The 3-space force that results is a mess:

```
(* 'q (+ (E an-event) (cross-product V (B an-event))))
(up (+ (* q (Ex (up ct0 x0 y0 z0)))
       (* q V_y (Bz (up ct0 x0 y0 z0)))
       (* -1 q V_z (By (up ct0 x0 y0 z0))))
    (+ (* q (Ey (up ct0 x0 y0 z0)))
       (* -1 q V_x (Bz (up ct0 x0 y0 z0)))
       (* q V_z (Bx (up ct0 x0 y0 z0))))
    (+ (* q (Ez (up ct0 x0 y0 z0)))
       (* q V_x (By (up ct0 x0 y0 z0)))
       (* -1 q V_y (Bx (up ct0 x0 y0 z0)))))
```

The relativistic Lorentz 4-force is usually written in coordinates as

$$f^\nu = -qU^\mu F_{\mu\alpha}\eta^{\alpha\nu} \tag{10.28}$$

where U is the 4-velocity of the charged particle, F is the Faraday tensor, and $\eta^{\alpha\nu}$ are the components of the inverse of the Minkowski metric. Here is a program that computes a component of the force in terms of the Faraday tensor. The desired component is specified by a one-form.

```
(define (Force charge F 4velocity component)
  (* -1 charge
     (contract
      (lambda (a b)
        (contract
         (lambda (e w)
           (* (w 4velocity)
              (F e a)
              (eta-inverse b component)))
         SR-basis))
      SR-basis)))
```

So, for example, the force in the $\hat{x}$ direction for a stationary particle is

```
((Force 'q F d/dct dx) an-event)
(* q (Ex (up ct0 x0 y0 z0)))
```

Notice that the 4-velocity $\partial/\partial ct$ is the 4-velocity of a stationary particle!

If we give a particle a more general timelike 4-velocity in the $\hat{x}$ direction we can see how the $\hat{y}$ component of the force involves both the electric and magnetic field:

```
(define (Ux beta)
  (+ (* (/ 1 (sqrt (- 1 (square beta)))) d/dct)
     (* (/ beta (sqrt (- 1 (square beta)))) d/dx)))

((Force 'q F (Ux 'v/c) dy) an-event)
(/ (+ (* -1 q v/c (Bz (up ct0 x0 y0 z0)))
      (* q (Ey (up ct0 x0 y0 z0))))
   (sqrt (+ 1 (* -1 (expt v/c 2)))))
```

Exercise 10.1: Relativistic Lorentz Force

Compute all components of the 4-force for a general timelike 4-velocity.

a. Compare these components to the components of the nonrelativistic force given above. Interpret the differences.

b. What is the meaning of the time component? For example, consider:

```
((Force 'q F (Ux 'v/c) dct) an-event)
(/ (* q v/c (Ex (up ct0 x0 y0 z0)))
   (sqrt (+ 1 (* -1 (expt v/c 2)))))
```

c. Subtract the structure of components of the relativistic 3-space force from the structure of the spatial components of the 4-space force to show that they are equal.

11
Special Relativity

Although the usual treatments of special relativity begin with the Michelson-Morley experiment this is not how Einstein began. In fact, Einstein was impressed with Maxwell's work and he was emulating Maxwell's breakthrough.

Maxwell was preceded by Faraday, Ampere, Oersted, Coulomb, Gauss, and Franklin. These giants discovered electromagnetism and worked out empirical equations that described the phenomena. They understood the existence of conserved charges and fields. Faraday invented the idea of lines of force by which fields can be visualized.

Maxwell's great insight was noticing and resolving the contradiction between the empirically-derived laws of electromagnetism and conservation of charge. He did this by introducing the then experimentally undetectable displacement-current term into one of the empirical equations. The modified equations implied a wave equation and the propagation speed of the wave predicted by the new equation turned out to be the speed of light, as measured by the eclipses of the Galilean satellites of Jupiter. The experimental confirmation by Hertz of the existence of electromagnetic radiation that obeyed Maxwell's equations capped the discovery.

By analogy, Einsten noticed that Maxwell's equations were inconsistent with Galilean relativity. In free space, where electromagnetic waves propagate, Maxwell's equations say that the vector source of electric fields is the time rate of change of the magnetic field and the vector source of magnetic field is the time rate of change of the electric field. The combination of these ideas yields the wave equation. The wave equation itself is not invariant under the Galilean transformation: As Einstein noted, if you run with the propagation speed of the wave there is no time variation in the field you observe, so there is no space variation either, contradicting the wave equation. But the Maxwell theory is beautiful, and it can be verified to a high degree of accuracy, so there must be something wrong with Galilean relativity. Einstein resolved the contradiction by generalizing the meaning of the Lorentz transformation, which was invented to explain the failure of the Michelson-Morley experiment. Lorentz and his colleagues decided that the problem with the Michelson-Morley experiment was that matter interacting with the luminiferous ether contracts in the direction of motion. To make this consistent he had to invent a "local time" which had no clear interpretation. Einstein took the Lorentz transformation to be a fundamental replacement for the Galilean transformation in all of mechanics.

Now to the details. Before Maxwell the empirical laws of electromagnetism were as follows. Electric fields arise from charges, with the inverse square law of Coulomb. This is Carl Friedrich Gauss's law for electrostatics:

$$\operatorname{div} \vec{E} = 4\pi\rho. \tag{11.1}$$

Magnetic fields do not have a scalar source. This is Gauss's law for magnetostatics:

$$\operatorname{div} \vec{B} = 0. \tag{11.2}$$

Magnetic fields are produced by electric currents, as discovered by Hans Christian Oersted and quantified by André-Marie Ampère:

$$\operatorname{curl} \vec{B} = \frac{4\pi}{c}\vec{I}. \tag{11.3}$$

Michael Faraday (and Joseph Henry) discovered that electric fields are produced by moving magnetic fields:

$$\operatorname{curl} \vec{E} = \frac{-1}{c}\frac{\partial \vec{B}}{\partial t}. \tag{11.4}$$

Benjamin Franklin was the first to understand that electrical charges are conserved:

$$\operatorname{div} \vec{I} + \frac{\partial \rho}{\partial t} = 0 \tag{11.5}$$

Although these equations are written in terms of the speed of light c, these laws were originally written in terms of electrical permittivity and magnetic permeability of free space, which could be determined by measurement of the forces for given currents and charges.

It is easy to see that these equations are mutually contradictory. Indeed, if we take the divergence of equation (11.3) we get

$$\text{div curl } \vec{B} = 0 = \frac{4\pi}{c}\text{div } \vec{I}, \tag{11.6}$$

which directly contradicts conservation of charge (11.5).

Maxwell patched this bug by adding in the displacement current, changing equation (11.3) to read

$$\text{curl } \vec{B} = \frac{1}{c}\frac{\partial \vec{E}}{\partial t} + \frac{4\pi}{c}\vec{I}. \tag{11.7}$$

Maxwell proceeded by taking the curl of equation (11.4) to get

$$\text{curl curl } \vec{E} = \frac{-1}{c}\frac{\partial}{\partial t} \text{ curl } \vec{B}. \tag{11.8}$$

Expanding the left-hand side

$$\text{grad div } \vec{E} - \text{Lap } \vec{E} = \frac{-1}{c}\frac{\partial}{\partial t} \text{ curl } \vec{B}, \tag{11.9}$$

substituting from equations (11.7) and (11.1), and rearranging the terms we get the inhomogeneous wave equation:

$$\text{Lap } \vec{E} - \frac{1}{c^2}\frac{\partial^2 \vec{E}}{\partial t^2} = 4\pi\left(\text{grad } \rho + \frac{1}{c^2}\vec{I}\right). \tag{11.10}$$

We see that in free space (in the absence of any charges or currents) we have the familiar homogeneous linear wave equation. A similar equation can be derived for the magnetic field.

Lorentz, whom Einstein also greatly respected, developed a general formula to describe the force on a particle with charge q moving with velocity $\vec{v}$ in an electromagnetic field:

$$\vec{F} = q\vec{E} + \frac{q}{c}\vec{v} \times \vec{B} \tag{11.11}$$

A crucial point in Einstein's inspiration for relativity is, quoting Einstein (in English translation), "During that year [1895–1896] in Aarau the question came to me: If one runs after a light wave with light velocity, then one would encounter a time-independent wavefield. However, something like that does not seem to exist!"[1] This was the observation of the inconsistency.

Let's be more precise about this. Consider a plane sinusoidal wave moving in the $\hat{x}$ direction with velocity c in free space ($\rho = 0$ and $\vec{I} = 0$). This is a perfectly good solution of the wave equation. Now suppose that an observer is moving with the wave in the $\hat{x}$ direction with velocity c. Such an observer will see no time variation of the field. So the wave equation reduces to Laplace's equation. But a sinusoidal variation in space is not a solution of Laplace's equation.

Einstein believed that the Maxwell-Lorentz electromagnetic theory was fundamentally correct, though he was unhappy with an apparent asymmetry in the formulation. Consider a system consisting of a conductor and a magnet. If the conductor is moved and the magnet is held stationary (a stationary magnetic field) then the charge carriers in the conductor are subject to the Lorentz force (11.11), causing them to move. However, if the magnet is moved past a stationary conductor then the changing magnetic field induces an electric field in the conductor by equation (11.4), which causes the charge carriers in the conductor to move. The actual current which results is identical for both explanations if the relative velocity of the magnet and the conductor are the same. To Einstein there should not be two explanations for the same phenomenon.

Invariance of the Wave Equation

Let $u = (t, x, y, z)$ be a tuple of time and space coordinates that specify a point in spacetime.[2] If $\phi(t, x, y, z)$ is a scalar field over time and space, the homogeneous linear wave equation is

$$\frac{\partial^2 \phi(u)}{\partial x^2} + \frac{\partial^2 \phi(u)}{\partial y^2} + \frac{\partial^2 \phi(u)}{\partial z^2} - \frac{1}{c^2}\frac{\partial^2 \phi(u)}{\partial t^2} = 0. \tag{11.12}$$

The characteristics for this equation are the "light cones." If we define a function of spacetime points and increments, length,

[1]The quote is from Pais [11], p. 131.

[2]Points in spacetime are often called *events*.

such that for an incremental tuple in position and time $\xi = (\Delta t, \Delta x, \Delta y, \Delta z)$ we have[3]

$$\mathrm{length}_u(\xi) = \sqrt{(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2 - (c\,\Delta t)^2}, \tag{11.13}$$

then the light cones are the hypersurfaces, for which

$$\mathrm{length}_u(\Delta t, \Delta x, \Delta y, \Delta z) = 0. \tag{11.14}$$

This "length" is called the *interval*.

What is the class of transformations of time and space coordinates that leave the Maxwell-Lorentz theory invariant? The transformations that preserve the wave equation are exactly those that leave its characteristics invariant. We consider a transformation $u = A(u')$ of time and space coordinates:

$$t = A^0(t', x', y', z') \tag{11.15}$$
$$x = A^1(t', x', y', z') \tag{11.16}$$
$$y = A^2(t', x', y', z') \tag{11.17}$$
$$z = A^3(t', x', y', z'). \tag{11.18}$$

If we define a new field $\psi(t', x', y', z')$ such that $\psi = \phi \circ A$, or

$$\psi(t', x', y', z') = \phi(A(t', x', y', z')), \tag{11.19}$$

then ψ will satisfy the wave equation

$$\frac{\partial^2 \psi(u')}{\partial x'^2} + \frac{\partial^2 \psi(u')}{\partial y'^2} + \frac{\partial^2 \psi(u')}{\partial z'^2} - \frac{1}{c^2}\frac{\partial^2 \psi(u')}{\partial t'^2} = 0. \tag{11.20}$$

if and only if

$$\mathrm{length}_{u'}(\xi') = \mathrm{length}_{A(u')}(DA\ \xi') = \mathrm{length}_u(\xi). \tag{11.21}$$

But this is just a statement that the velocity of light is invariant under change of the coordinate system. The class of transformations that satisfy equation (11.21) are the Poincaré transformations.

[3]Here the length is independent of the spacetime point specified by u. In General Relativity we find that the metric, and thus the length function needs to vary with the point in spacetime.

11.1 Lorentz Transformations

Special relativity is usually presented in terms of global Lorentz frames, with rectangular spatial coordinates. In this context the Lorentz transformations (and, more generally, the Poincaré transformations) can be characterized as the set of affine transformations (linear transformations plus shift) of the coordinate tuple (time and spatial rectangular coordinates) that preserve the length of incremental spacetime intervals as measured by

$$f(\xi) = -(\xi^0)^2 + (\xi^1)^2 + (\xi^2)^2 + (\xi^3)^2, \tag{11.22}$$

where ξ is an incremental 4-tuple that could be added to the coordinate 4-tuple (ct, x, y, z).[4] The Poincaré-Lorentz transformations are of the form

$$x = \Lambda x' + a, \tag{11.23}$$

where Λ is the tuple representation of a linear transformation and a is a 4-tuple shift. Because the 4-tuple includes the time, these transformations include transformations to a uniformly moving frame. A transformation that does not rotate or shift, but just introduces relative velocity, is sometimes called a *boost*.

In general relativity, global Lorentz frames do not exist, and so global affine transformations are irrelevant. In general relativity Lorentz invariance is a local property of incremental 4-tuples at a point.

Incremental 4-tuples transform as

$$\xi = \Lambda \xi'. \tag{11.24}$$

This places a constraint on the allowed Λ

$$f(\xi') = f(\Lambda \xi'), \tag{11.25}$$

for arbitrary ξ'.

The possible Λ that are consistent with the preservation of the interval can be completely specified and conveniently parameterized.

[4]Incrementally, $\xi = \xi^0 \partial/\partial ct + \xi^1 \partial/\partial x + \xi^2 \partial/\partial y + \xi^3 \partial/\partial z$. The length of this vector, using the Minkowski metric (see equation 10.11), is the Lorentz interval, the right-hand side of equation (11.22).

Simple Lorentz Transformations

Consider the linear transformation, in the first two coordinates,

$$\begin{aligned}\xi^0 &= p(\xi')^0 + q(\xi')^1\\ \xi^1 &= r(\xi')^0 + s(\xi')^1.\end{aligned} \tag{11.26}$$

The requirement that the interval be preserved gives the constraints

$$\begin{aligned}p^2 - r^2 &= 1\\ pq - rs &= 0\\ q^2 - s^2 &= -1.\end{aligned} \tag{11.27}$$

There are four parameters to determine, and only three equations, so the solutions have a free parameter. It turns out that a good choice is $\beta = q/p$. Solve to find

$$p = \frac{1}{\sqrt{1-\beta^2}} = \gamma(\beta), \tag{11.28}$$

and also $p = s$ and $q = r = \beta p$. This defines γ. Written out, the transformation is

$$\begin{aligned}\xi^0 &= \gamma(\beta)((\xi')^0 + \beta(\xi')^1)\\ \xi^1 &= \gamma(\beta)(\beta(\xi')^0 + (\xi')^1).\end{aligned} \tag{11.29}$$

Simple physical arguments[5] show that this mathematical result relates the time and space coordinates for two systems in uniform relative motion. The parameter β is related to the relative velocity.

Consider incremental vectors as spacetime vectors relative to an origin in a global inertial frame. So, for example, $\xi = (ct, x)$, ignoring y and z for a moment. The unprimed coordinate origin $x = 0$ corresponds, in primed coordinates, to (using equations 11.29)

$$x = 0 = \gamma(\beta)(x' + \beta ct'), \tag{11.30}$$

so

$$\beta = -\frac{x'}{ct'} = -\frac{v'}{c}, \tag{11.31}$$

[5]See, for instance, Mermin, "Space and Time in Special Relativity."

with the definition $v' = x'/t'$. We see that β is minus $1/c$ times the velocity (v') of the unprimed system (which moves with its origin) as "seen" in the primed coordinates.

To check the consistency of our interpretation, we can find the velocity of the origin of the primed system ($x' = 0$) as seen by the unprimed system. Using both of equations (11.29), we find

$$\beta = \frac{x}{ct} = \frac{v}{c}. \tag{11.32}$$

So $v' = -v$.

A consistent interpretation is that the origin of the primed system moves with velocity $v = \beta c$ along the $\hat{x}$-axis of the unprimed system. And the unprimed system moves with the same velocity in the other direction, when viewed in terms of the primed system.

What happened to the other coordinates: y and z? We did not need them to find this one-parameter family of Lorentz transformations. They are left alone. This mathematical result has a physical interpretation: Lengths are not affected by perpendicular boosts. Think about two observers on a collision course, each carrying a meter stick perpendicular to their relative velocity. At the moment of impact, the meter sticks must coincide. The symmetry of the situation does not permit one observer to conclude that one meter stick is shorter than the other, because the other observer must come to the same conclusion. Both observers can put their conclusions to the test upon impact.

We can fill in the components of this simple boost:

$$\begin{aligned}
\xi^0 &= \gamma(\beta)((\xi')^0 + \beta(\xi')^1) \\
\xi^1 &= \gamma(\beta)(\beta(\xi')^0 + (\xi')^1) \\
\xi^2 &= (\xi')^2 \\
\xi^3 &= (\xi')^3.
\end{aligned} \tag{11.33}$$

More General Lorentz Transformations

One direction was special in our consideration of simple boosts. We can make use of this fact to find boosts in any direction.

Let $c\boldsymbol{\beta} = (v^0, v^1, v^2)$ be the tuple of components of the relative velocity of the origin of the primed system in the unprimed system. The components are with respect to the same rectangular basis used to define the spatial components of any incremental vector.

An incremental vector can be decomposed into vectors parallel and perpendicular to the velocity. Let $\boldsymbol{\xi}$ be the tuple of spatial components of ξ, and ξ^0 be the time component. Then,

$$\boldsymbol{\xi} = \boldsymbol{\xi}^{\perp} + \boldsymbol{\xi}^{\|}, \tag{11.34}$$

where $\boldsymbol{\beta} \cdot \boldsymbol{\xi}^{\perp} = 0$. (This is the ordinary dot product in three dimensions.) Explicitly,

$$\boldsymbol{\xi}^{\|} = \frac{\boldsymbol{\beta}}{\beta}(\frac{\boldsymbol{\beta}}{\beta} \cdot \boldsymbol{\xi}), \tag{11.35}$$

where $\beta = \|\boldsymbol{\beta}\|$, the magnitude of $\boldsymbol{\beta}$, and

$$\boldsymbol{\xi}^{\perp} = \boldsymbol{\xi} - \boldsymbol{\xi}^{\|}. \tag{11.36}$$

In the simple boost of equation (11.33) we can identify ξ^1 with the magnitude $|\boldsymbol{\xi}^{\|}|$ of the parallel component. The perpendicular component is unchanged.

$$\begin{aligned} \xi^0 &= \gamma(\beta)((\xi')^0 + \beta|(\boldsymbol{\xi}')^{\|}|) \\ |\boldsymbol{\xi}^{\|}| &= \gamma(\beta)(\beta(\xi')^0 + |(\boldsymbol{\xi}')^{\|}|) \\ \boldsymbol{\xi}^{\perp} &= (\boldsymbol{\xi}')^{\perp} \end{aligned} \tag{11.37}$$

Putting the components back together, this leads to

$$\begin{aligned} \xi^0 &= \gamma(\beta)\left((\xi')^0 + \boldsymbol{\beta} \cdot \boldsymbol{\xi}'\right) \\ \boldsymbol{\xi} &= \gamma(\beta)\boldsymbol{\beta}(\xi')^0 + \boldsymbol{\xi}' + \frac{\gamma(\beta) - 1}{\beta^2}\boldsymbol{\beta}(\boldsymbol{\beta} \cdot \boldsymbol{\xi}'), \end{aligned} \tag{11.38}$$

which gives the components of the general boost B along velocity $c\boldsymbol{\beta}$:

$$\xi = B(\boldsymbol{\beta})(\xi'). \tag{11.39}$$

Implementation

We represent a 4-tuple as a flat up-tuple of components.

```
(define (make-4tuple ct space)
  (up ct (ref space 0) (ref space 1) (ref space 2)))
```

```
(define (4tuple->ct v) (ref v 0))
(define (4tuple->space v)
  (up (ref v 1) (ref v 2) (ref v 3)))
```

The invariant interval is then

```
(define (proper-space-interval 4tuple)
  (sqrt (- (square (4tuple->space 4tuple))
           (square (4tuple->ct 4tuple)))))
```

This is a real number for space-like intervals. A space-like interval is one where spatial distance is larger than can be traversed by light in the time interval.

It is often convenient for the interval to be real for time-like intervals, where light can traverse the spatial distance in less than the time interval.

```
(define (proper-time-interval 4tuple)
  (sqrt (- (square (4tuple->ct 4tuple))
           (square (4tuple->space 4tuple)))))
```

The general boost B is

```
(define ((general-boost beta) xi-p)
  (let ((gamma (expt (- 1 (square beta)) -1/2)))
    (let ((factor (/ (- gamma 1) (square beta))))
      (let ((xi-p-time (4tuple->ct xi-p))
            (xi-p-space (4tuple->space xi-p)))
        (let ((beta-dot-xi-p (dot-product beta xi-p-space)))
          (make-4-tuple
           (* gamma (+ xi-p-time beta-dot-xi-p))
           (+ (* gamma beta xi-p-time)
              xi-p-space
              (* factor beta beta-dot-xi-p))))))))
```

We can check that the interval is invariant:

```
(- (proper-space-interval
     ((general-boost (up 'vx 'vy 'vz))
      (make-4tuple 'ct (up 'x 'y 'z))))
   (proper-space-interval
     (make-4tuple 'ct (up 'x 'y 'z))))
0
```

It is inconvenient that the general boost as just defined does not work if $\boldsymbol{\beta}$ is zero. An alternate way to specify a boost is through the magnitude of v/c and a direction:

```
(define ((general-boost2 direction v/c) 4tuple-prime)
  (let ((delta-ct-prime (4tuple->ct 4tuple-prime))
        (delta-x-prime (4tuple->space 4tuple-prime)))
    (let ((betasq (square v/c)))
      (let ((bx (dot-product direction delta-x-prime))
            (gamma (/ 1 (sqrt (- 1 betasq)))))
        (let ((alpha (- gamma 1)))
          (let ((delta-ct
                 (* gamma (+ delta-ct-prime (* bx v/c))))
                (delta-x
                 (+ (* gamma v/c direction delta-ct-prime)
                    delta-x-prime
                    (* alpha direction bx))))
            (make-4tuple delta-ct delta-x)))))))
```

This is well behaved as v/c goes to zero.

Rotations

A linear transformation that does not change the magnitude of the spatial and time components, individually, leaves the interval invariant. So a transformation that rotates the spatial coordinates and leaves the time component unchanged is also a Lorentz transformation. Let R be a 3-dimensional rotation. Then the extension to a Lorentz transformation $\mathcal{R}$ is defined by

$$(\xi^0, \boldsymbol{\xi}) = \mathcal{R}(R)((\xi')^0, \boldsymbol{\xi}') = ((\xi')^0, R(\boldsymbol{\xi}')). \tag{11.40}$$

Examining the expression for the general boost, equation (11.38), we see that the boost transforms simply as the arguments are rotated. Indeed,

$$B(\boldsymbol{\beta}) = (\mathcal{R}(R))^{-1} \circ B(R(\boldsymbol{\beta})) \circ \mathcal{R}(R). \tag{11.41}$$

Note that $(\mathcal{R}(R))^{-1} = \mathcal{R}(R^{-1})$. The functional inverse of the extended rotation is the extension of the inverse rotation. We could use this property of boosts to think of the general boost as a combination of a rotation and a simple boost along some special direction.

The extended rotation can be implemented:

```
(define ((extended-rotation R) xi)
  (make-4tuple
   (4tuple->ct xi)
   (R (4tuple->space xi))))
```

In terms of this we can check the relation between boosts and rotations:

```
(let ((beta (up 'bx 'by 'bz))
      (xi (make-4tuple 'ct (up 'x 'y 'z)))
      (R (compose
          (rotate-x 'theta)
          (rotate-y 'phi)
          (rotate-z 'psi)))
      (R-inverse (compose
                  (rotate-z (- 'psi))
                  (rotate-y (- 'phi))
                  (rotate-x (- 'theta)))))
  (- ((general-boost beta) xi)
     ((compose (extended-rotation R-inverse)
               (general-boost (R beta))
               (extended-rotation R))
        xi)))
(up 0 0 0 0)
```

General Lorentz Transformations

A Lorentz transformation carries an incremental 4-tuple to another 4-tuple. A general linear transformation on 4-tuples has sixteen free parameters. The interval is a symmetric quadratic form, so the requirement that the interval be preserved places only ten constraints on these parameters. Evidently there are six free parameters to the general Lorentz transformation. We already have three parameters that specify boosts (the three components of the boost velocity). And we have three more parameters in the extended rotations. The general Lorentz transformation can be constructed by combining generalized rotations and boosts.

Any Lorentz transformation has a unique decomposition as a generalized rotation followed by a general boost. Any Λ that preserves the interval can be written uniquely:

$$\Lambda = B(\boldsymbol{\beta})\mathcal{R}. \tag{11.42}$$

We can use property (11.41) to see this. Suppose we follow a general boost by a rotation. A new boost can be defined to absorb this rotation, but only if the boost is preceded by a suitable rotation:

$$\mathcal{R}(R) \circ B(\boldsymbol{\beta}) = B(R(\boldsymbol{\beta})) \circ \mathcal{R}(R). \tag{11.43}$$

Exercise 11.1: Lorentz Decomposition

The counting of free parameters supports the conclusion that the general Lorentz transformation can be constructed by combining generalized rotations and boosts. Then the decomposition (11.42) follows from property (11.41). Find a more convincing proof.

11.2 Special Relativity Frames

A new frame is defined by a Poincaré transformation from a given frame (see equation 11.23). The transformation is specified by a boost magnitude and a unit-vector boost direction, relative to the given frame, and the position of the origin of the frame being defined in the given frame.

Points in spacetime are called events. It must be possible to compare two events to determine if they are the same. This is accomplished in any particular experiment by building all frames involved in that experiment from a base frame, and representing the events as coordinates in that base frame.

When one frame is built upon another, to determine the event from frame-specific coordinates or to determine the frame-specific coordinates for an event requires composition of the boosts that relate the frames to each other. The two procedures that are required to implement this strategy are[6]

```
(define ((coordinates->event ancestor-frame this-frame
                             boost-direction v/c origin)
         coords)
  ((point ancestor-frame)
   (make-SR-coordinates ancestor-frame
     (+ ((general-boost2 boost-direction v/c) coords)
        origin))))
```

[6]The procedure `make-SR-coordinates` labels the given coordinates with the given frame. The procedures that manipulate coordinates, such as `(point ancestor-frame)`, check that the coordinates they are given are in the appropriate frame. This error checking makes it easier to debug relativity procedures.

```
(define ((event->coordinates ancestor-frame this-frame
                             boost-direction v/c origin)
         event)
  (make-SR-coordinates this-frame
    ((general-boost2 (- boost-direction) v/c)
     (- ((chart ancestor-frame) event) origin))))
```

With these two procedures, the procedure `make-SR-frame` constructs a new relativistic frame by a Poincaré transformation from a given frame.

```
(define make-SR-frame
  (frame-maker coordinates->event event->coordinates))
```

Velocity Addition Formula

For example, we can derive the traditional velocity addition formula. Assume that we have a base frame called `home`. We can make a frame `A` by a boost from `home` in the $\hat{x}$ direction, with components $(1, 0, 0)$, and with a dimensionless measure of the speed v_a/c. We also specify that the 4-tuple origin of this new frame coincides with the origin of `home`.

```
(define A
   (make-SR-frame 'A home
                  (up 1 0 0)
                  'va/c
                  (make-SR-coordinates home (up 0 0 0 0))))
```

Frame `B` is built on frame `A` similarly, boosted by v_b/c.

```
(define B
   (make-SR-frame 'B A
                  (up 1 0 0)
                  'vb/c
                  (make-SR-coordinates A (up 0 0 0 0))))
```

So any point at rest in frame `B` will have a speed relative to `home`. For the spatial origin of frame `B`, with `B` coordinates `(up 'ct 0 0 0)`, we have

```
(let ((B-origin-home-coords
       ((chart home)
        ((point B)
         (make-SR-coordinates B (up 'ct 0 0 0))))))
  (/ (ref B-origin-home-coords 1)
     (ref B-origin-home-coords 0)))
(/ (+ va/c vb/c) (+ 1 (* va/c vb/c)))
```

obtaining the traditional velocity-addition formula. (Note that the resulting velocity is represented as a fraction of the speed of light.) This is a useful result, so:

```
(define (add-v/cs va/c vb/c)
   (/ (+ va/c vb/c)
      (+ 1 (* va/c vb/c))))
```

11.3 Twin Paradox

Special relativity engenders a traditional conundrum: consider two twins, one of whom travels and the other stays at home. When the traveller returns it is discovered that the traveller has aged less than the twin who stayed at home. How is this possible?

The experiment begins at the start event, which we arbitrarily place at the origin of the home frame.

```
(define start-event
  ((point home)
   (make-SR-coordinates home (up 0 0 0 0))))
```

There is a homebody and a traveller. The traveller leaves home at the start event and proceeds at 24/25 of the speed of light in the $\hat{x}$ direction. We define a frame for the traveller, by boosting from the home frame.

```
(define outgoing
  (make-SR-frame 'outgoing       ; for debugging
                 home            ; base frame
                 (up 1 0 0)      ; x direction
                 24/25           ; velocity as fraction of c
                 ((chart home)
                  start-event)))
```

After 25 years of home time the traveller is 24 light-years out. We define that event using the coordinates in the home frame.

Here we scale the time coordinate by the speed of light so that the units of ct slot in the 4-vector are the same as the units in the spatial slots. Since $v/c = 24/25$ we must multiply that by the speed of light to get the velocity. This is multiplied by 25 years to get the $\hat{x}$ coordinate of the traveller in the home frame at the turning point.

```
(define traveller-at-turning-point-event
  ((point home)
   (make-SR-coordinates home
     (up (* :c 25) (* 25 24/25 :c) 0 0))))
```

Note that the first component of the coordinates of an event is the speed of light multiplied by time. The other components are distances. For example, the second component (the $\hat{x}$ component) is the distance travelled in 25 years at 24/25 the speed of light. This is 24 light-years.

If we examine the displacement of the traveller in his own frame we see that the traveller has aged 7 years and he has not moved from his spatial origin.

```
(- ((chart outgoing) traveller-at-turning-point-event)
   ((chart outgoing) start-event))
(up (* 7 :c) 0 0 0)
```

But in the frame of the homebody we see that the time has advanced by 25 years.

```
(- ((chart home) traveller-at-turning-point-event)
   ((chart home) start-event))
(up (* 25 :c) (* 24 :c) 0 0)
```

The proper time interval is 7 years, as seen in any frame

```
(proper-time-interval
 (- ((chart outgoing) traveller-at-turning-point-event)
    ((chart outgoing) start-event)))
(* 7 :c)

(proper-time-interval
 (- ((chart home) traveller-at-turning-point-event)
    ((chart home) start-event)))
(* 7 :c)
```

because it measures the aging of the traveller.

When the traveller is at the turning point, the event of the homebody is:

```
(define halfway-at-home-event
  ((point home)
   (make-SR-coordinates home (up (* :c 25) 0 0 0))))
```

and the homebody has aged

```
(proper-time-interval
 (- ((chart home) halfway-at-home-event)
    ((chart home) start-event)))
(* 25 :c)

(proper-time-interval
 (- ((chart outgoing) halfway-at-home-event)
    ((chart outgoing) start-event)))
(* 25 :c)
```

as seen from either frame.

As seen by the traveller, home is moving in the $-\hat{x}$ direction at 24/25 of the velocity of light. At the turning point (7 years by his time) home is at:

```
(define home-at-outgoing-turning-point-event
  ((point outgoing)
   (make-SR-coordinates outgoing
     (up (* 7 :c) (* 7 -24/25 :c) 0 0))))
```

Since home is speeding away from the traveller, the twin at home has aged less than the traveller. This may seem weird, but it is OK because this event is different from the halfway event in the home frame.

```
(proper-time-interval
 (- ((chart home) home-at-outgoing-turning-point-event)
    ((chart home) start-event)))
(* 49/25 :c)
```

The traveller turns around abruptly at this point (painful!) and begins the return trip. The incoming trip is the reverse of the outgoing trip, with origin at the turning-point event:

```
(define incoming
  (make-SR-frame 'incoming home
                 (up -1 0 0) 24/25
                 ((chart home)
                  traveller-at-turning-point-event)))
```

After 50 years of home time the traveller reunites with the homebody:

```
(define end-event
  ((point home)
   (make-SR-coordinates home (up (* :c 50) 0 0 0))))
```

Indeed, the traveller comes home after 7 more years in the incoming frame:

```
(- ((chart incoming) end-event)
   (make-SR-coordinates incoming
     (up (* :c 7) 0 0 0)))
(up 0 0 0 0)

(- ((chart home) end-event)
   ((chart home)
    ((point incoming)
     (make-SR-coordinates incoming
       (up (* :c 7) 0 0 0)))))
(up 0 0 0 0)
```

The traveller ages only 7 years on the return segment, so his total aging is 14 years:

```
(+ (proper-time-interval
  (- ((chart outgoing) traveller-at-turning-point-event)
     ((chart outgoing) start-event)))
 (proper-time-interval
  (- ((chart incoming) end-event)
     ((chart incoming) traveller-at-turning-point-event))))
(* 14 :c)
```

But the homebody ages 50 years:

```
(proper-time-interval
 (- ((chart home) end-event)
    ((chart home) start-event)))
(* 50 :c)
```

At the turning point of the traveller the homebody is at

```
(define home-at-incoming-turning-point-event
  ((point incoming)
   (make-SR-coordinates incoming
     (up 0 (* 7 -24/25 :c) 0 0))))
```

The time elapsed for the homebody between the reunion and the turning point of the homebody, as viewed by the incoming traveller, is about 2 years.

```
(proper-time-interval
 (- ((chart home) end-event)
    ((chart home) home-at-incoming-turning-point-event)))
(* 49/25 :c)
```

Thus the aging of the homebody occurs at the turnaround, from the point of view of the traveller.

A
Scheme

Programming languages should be designed not by piling feature on top of feature, but by removing the weaknesses and restrictions that make additional features appear necessary. Scheme demonstrates that a very small number of rules for forming expressions, with no restrictions on how they are composed, suffice to form a practical and efficient programming language that is flexible enough to support most of the major programming paradigms in use today.

IEEE Standard for the Scheme Programming Language [9], p. 3

Here we give an elementary introduction to Scheme.[1] For a more precise explanation of the language see the IEEE standard [9]; for a longer introduction see the textbook [1].

Scheme is a simple programming language based on expressions. An expression names a value. For example, the numeral `3.14` names an approximation to a familiar number. There are primitive expressions, such as a numeral, that we directly recognize, and there are compound expressions of several kinds.

Procedure Calls

A *procedure call* is a kind of compound expression. A procedure call is a sequence of expressions delimited by parentheses. The first subexpression in a procedure call is taken to name a procedure, and the rest of the subexpressions are taken to name the arguments to that procedure. The value produced by the procedure when applied to the given arguments is the value named by the procedure call. For example,

[1]Many of the statements here are valid only assuming that no assignments are used.

```
(+ 1 2.14)
3.14

(+ 1 (* 2 1.07))
3.14
```

are both compound expressions that name the same number as the numeral 3.14.[2] In these cases the symbols `+` and `*` name procedures that add and multiply, respectively. If we replace any subexpression of any expression with an expression that names the same thing as the original subexpression, the thing named by the overall expression remains unchanged. In general, a procedure call is written

(*operator* *operand-1* ... *operand-n*)

where *operator* names a procedure and *operand-i* names the ith argument.[3]

Lambda Expressions

Just as we use numerals to name numbers, we use λ-expressions to name procedures.[4] For example, the procedure that squares its input can be written:

```
(lambda (x) (* x x))
```

This expression can be read: "The procedure of one argument, x, that multiplies x by x." Of course, we can use this expression in any context where a procedure is needed. For example,

```
((lambda (x) (* x x)) 4)
16
```

The general form of a λ-expression is

[2]In examples we show the value that would be printed by the Scheme system using slanted characters following the input expression.

[3]In Scheme every parenthesis is essential: you cannot add extra parentheses or remove any.

[4]The logician Alonzo Church [4] invented λ-notation to allow the specification of an anonymous function of a named parameter: $\boldsymbol{\lambda}x$[expression in x]. This is read, "That function of one argument that is obtained by substituting the argument for x in the indicated expression."

(`lambda` *formal-parameters body*)

where *formal-parameters* is a list of symbols that will be the names of the arguments to the procedure and *body* is an expression that may refer to the formal parameters. The value of a procedure call is the value of the body of the procedure with the arguments substituted for the formal parameters.

Definitions

We can use the `define` construct to give a name to any object. For example, if we make the definitions[5]

```
(define pi 3.141592653589793)

(define square (lambda (x) (* x x)))
```

we can then use the symbols `pi` and `square` wherever the numeral or the λ-expression could appear. For example, the area of the surface of a sphere of radius 5 meters is

```
(* 4 pi (square 5))
314.1592653589793
```

Procedure definitions may be expressed more conveniently using "syntactic sugar." The squaring procedure may be defined

```
(define (square x) (* x x))
```

which we may read: "To square x multiply x by x."

In Scheme, procedures may be passed as arguments and returned as values. For example, it is possible to make a procedure that implements the mathematical notion of the composition of two functions:[6]

[5]The definition of `square` given here is not the definition of `square` in the Scmutils system. In Scmutils, `square` is extended for tuples to mean the sum of the squares of the components of the tuple. However, for arguments that are not tuples the Scmutils `square` does multiply the argument by itself.

[6]The examples are indented to help with readability. Scheme does not care about extra white space, so we may add as much as we please to make things easier to read.

```
(define compose
  (lambda (f g)
    (lambda (x)
      (f (g x)))))

((compose square sin) 2)
.826821810431806

(square (sin 2))
.826821810431806
```

Using the syntactic sugar shown above, we can write the definition more conveniently. The following are both equivalent to the definition above:

```
(define (compose f g)
  (lambda (x)
    (f (g x))))

(define ((compose f g) x)
  (f (g x)))
```

Conditionals

Conditional expressions may be used to choose among several expressions to produce a value. For example, a procedure that implements the absolute value function may be written:

```
(define (abs x)
  (cond ((< x 0) (- x))
        ((= x 0) x)
        ((> x 0) x)))
```

The conditional `cond` takes a number of clauses. Each clause has a predicate expression, which may be either true or false, and a consequent expression. The value of the `cond` expression is the value of the consequent expression of the first clause for which the corresponding predicate expression is true. The general form of a conditional expression is

```
(cond ( predicate-1   consequent-1)
        ...
      ( predicate-n   consequent-n))
```

For convenience there is a special predicate expression `else` that can be used as the predicate in the last clause of a `cond`. The `if` construct provides another way to make a conditional when there is only a binary choice to be made. For example, because we have to do something special only when the argument is negative, we could have defined `abs` as:

```
(define (abs x)
  (if (< x 0)
      (- x)
      x))
```

The general form of an `if` expression is

(`if` *predicate* *consequent* *alternative*)

If the *predicate* is true the value of the `if` expression is the value of the *consequent*, otherwise it is the value of the *alternative*.

Recursive Procedures

Given conditionals and definitions, we can write recursive procedures. For example, to compute the nth factorial number we may write:

```
(define (factorial n)
  (if (= n 0)
      1
      (* n (factorial (- n 1)))))

(factorial 6)
720

(factorial 40)
815915283247897734345611269596115894272000000000
```

Local Names

The `let` expression is used to give names to objects in a local context. For example,

```
(define (f radius)
  (let ((area (* 4 pi (square radius)))
        (volume (* 4/3 pi (cube radius))))
    (/ volume area)))

(f 3)
1
```

The general form of a `let` expression is

```
(let (( variable-1   expression-1)
      ...
      ( variable-n   expression-n))
   body)
```

The value of the `let` expression is the value of the *body* expression in the context where the variables *variable-i* have the values of the expressions *expression-i*. The expressions *expression-i* may not refer to any of the variables.

A slight variant of the `let` expression provides a convenient way to express looping constructs. We can write a procedure that implements an alternative algorithm for computing factorials as follows:

```
(define (factorial n)
  (let factlp ((count 1) (answer 1))
    (if (> count n)
        answer
        (factlp (+ count 1) (* count answer)))))

(factorial 6)
720
```

Here, the symbol `factlp` following the `let` is locally defined to be a procedure that has the variables `count` and `answer` as its formal parameters. It is called the first time with the expressions 1 and 1, initializing the loop. Whenever the procedure named `factlp` is called later, these variables get new values that are the values of the operand expressions `(+ count 1)` and `(* count answer)`.

Compound Data—Lists and Vectors

Data can be glued together to form compound data structures. A list is a data structure in which the elements are linked sequentially. A Scheme vector is a data structure in which the elements are packed in a linear array. New elements can be added to lists, but to access the *n*th element of a list takes computing time proportional to *n*. By contrast a Scheme vector is of fixed length, and its elements can be accessed in constant time. All data structures in this book are implemented as combinations of lists and Scheme vectors. Compound data objects are constructed from components by procedures called constructors and the components are accessed by selectors.

The procedure `list` is the constructor for lists. The selector `list-ref` gets an element of the list. All selectors in Scheme are zero-based. For example,

```
(define a-list (list 6 946 8 356 12 620))

a-list
(6 946 8 356 12 620)

(list-ref a-list 3)
356

(list-ref a-list 0)
6
```

Lists are built from pairs. A pair is made using the constructor `cons`. The selectors for the two components of the pair are `car` and `cdr` (pronounced "could-er").[7] A list is a chain of pairs, such that the `car` of each pair is the list element and the `cdr` of each pair is the next pair, except for the last `cdr`, which is a distinguishable value called the empty list and is written `()`. Thus,

```
(car a-list)
6

(cdr a-list)
(946 8 356 12 620)

(car (cdr a-list))
946

(define another-list
  (cons 32 (cdr a-list)))

another-list
(32 946 8 356 12 620)

(car (cdr another-list))
946
```

[7]These names are accidents of history. They stand for "Contents of the Address part of Register" and "Contents of the Decrement part of Register" of the IBM 704 computer, which was used for the first implementation of Lisp in the late 1950s. Scheme is a dialect of Lisp.

Both `a-list` and `another-list` share the same tail (their `cdr`).

There is a predicate `pair?` that is true of pairs and false on all other types of data.

Vectors are simpler than lists. There is a constructor `vector` that can be used to make vectors and a selector `vector-ref` for accessing the elements of a vector:

```
(define a-vector
  (vector 37 63 49 21 88 56))

a-vector
#(37 63 49 21 88 56)

(vector-ref a-vector 3)
21

(vector-ref a-vector 0)
37
```

Notice that a vector is distinguished from a list on printout by the character # appearing before the initial parenthesis.

There is a predicate `vector?` that is true of vectors and false for all other types of data.

The elements of lists and vectors may be any kind of data, including numbers, procedures, lists, and vectors. Numerous other procedures for manipulating list-structured data and vector-structured data can be found in the Scheme online documentation.

Symbols

Symbols are a very important kind of primitive data type that we use to make programs and algebraic expressions. You probably have noticed that Scheme programs look just like lists. In fact, they are lists. Some of the elements of the lists that make up programs are symbols, such as `+` and `vector`.[8] If we are to make programs that can manipulate programs, we need to be able to write an expression that names such a symbol. This is accomplished by the mechanism of *quotation*. The name of the symbol `+` is the expression `'+`, and in general the name of an expression

[8]Symbols may have any number of characters. A symbol may not contain whitespace or a delimiter character, such as parentheses, brackets, quotation marks, comma, or #.

is the expression preceded by a single quote character. Thus the name of the expression `(+ 3 a)` is `'(+ 3 a)`.

We can test if two symbols are identical by using the predicate `eq?`. For example, we can write a program to determine if an expression is a sum:

```
(define (sum? expression)
  (and (pair? expression)
       (eq? (car expression) '+)))

(sum? '(+ 3 a))
#t

(sum? '(* 3 a))
#f
```

Here `#t` and `#f` are the printed representations of the boolean values true and false.

Consider what would happen if we were to leave out the quote in the expression `(sum? '(+ 3 a))`. If the variable `a` had the value 4 we would be asking if 7 is a sum. But what we wanted to know was whether the expression `(+ 3 a)` is a sum. That is why we need the quote.

B
Our Notation

An adequate notation should be understood by at least two people, one of whom may be the author.

Abdus Salam (1950).

We adopt a *functional mathematical notation* that is close to that used by Spivak in his *Calculus on Manifolds* [16]. The use of functional notation avoids many of the ambiguities of traditional mathematical notation that can impede clear reasoning. Functional notation carefully distinguishes the function from the value of the function when applied to particular arguments. In functional notation mathematical expressions are unambiguous and self-contained.

We adopt a *generic arithmetic* in which the basic arithmetic operations, such as addition and multiplication, are extended to a wide variety of mathematical types. Thus, for example, the addition operator $+$ can be applied to numbers, tuples of numbers, matrices, functions, etc. Generic arithmetic formalizes the common informal practice used to manipulate mathematical objects.

We often want to manipulate aggregate quantities, such as the collection of all of the rectangular coordinates of a collection of particles, without explicitly manipulating the component parts. Tensor arithmetic provides a traditional way of manipulating aggregate objects: Indices label the parts; conventions, such as the summation convention, are introduced to manipulate the indices. We introduce a *tuple arithmetic* as an alternative way of manipulating aggregate quantities that usually lets us avoid labeling the parts with indices. Tuple arithmetic is inspired by tensor arithmetic but it is more general: not all of the components of a tuple need to be of the same size or type.

The mathematical notation is in one-to-one correspondence with expressions of the computer language *Scheme* [9]. Scheme is based on the λ-calculus [4] and directly supports the manipulation of functions. We augment Scheme with symbolic, numerical, and generic features to support our applications. For a simple introduction to Scheme, see Appendix A. The correspondence between the mathematical notation and Scheme requires that mathematical expressions be unambiguous and self-contained. Scheme provides immediate feedback in verification of mathematical deductions and facilitates the exploration of the behavior of systems.

Functions

The expression $f(x)$ denotes the value of the function f at the given argument x; when we wish to denote the function we write just f. Functions may take several arguments. For example, we may have the function that gives the Euclidean distance between two points in the plane given by their rectangular coordinates:

$$d(x_1, y_1, x_2, y_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{B.1}$$

In Scheme we can write this as:

```
(define (d x1 y1 x2 y2)
  (sqrt (+ (square (- x2 x1)) (square (- y2 y1)))))
```

Functions may be composed if the range of one overlaps the domain of the other. The composition of functions is constructed by passing the output of one to the input of the other. We write the composition of two functions using the $\circ$ operator:

$$(f \circ g) : x \mapsto (f \circ g)(x) = f(g(x)). \tag{B.2}$$

A procedure `h` that computes the cube of the sine of its argument may be defined by composing the procedures `cube` and `sin`:

```
(define h (compose cube sin))

(h 2)
.7518269446689928
```

which is the same as

```
(cube (sin 2))
.7518269446689928
```

Arithmetic is extended to the manipulation of functions: the usual mathematical operations may be applied to functions. Examples are addition and multiplication; we may add or multiply two functions if they take the same kinds of arguments and if their values can be added or multiplied:

$$\begin{aligned}(f+g)(x) &= f(x)+g(x),\\ (fg)(x) &= f(x)g(x).\end{aligned} \tag{B.3}$$

A procedure `g` that multiplies the cube of its argument by the sine of its argument is

```
(define g (* cube sin))

(g 2)
7.274379414605454

(* (cube 2) (sin 2))
7.274379414605454
```

Symbolic Values

As in usual mathematical notation, arithmetic is extended to allow the use of symbols that represent unknown or incompletely specified mathematical objects. These symbols are manipulated as if they had values of a known type. By default, a Scheme symbol is assumed to represent a real number. So the expression `'a` is a literal Scheme symbol that represents an unspecified real number:

```
((compose cube sin) 'a)
(expt (sin a) 3)
```

The default printer simplifies the expression,[1] and displays it in a readable form. We can use the simplifier to verify a trigonometric identity:

```
((- (+ (square sin) (square cos)) 1) 'a)
0
```

Just as it is useful to be able to manipulate symbolic numbers, it is useful to be able to manipulate symbolic functions. The procedure `literal-function` makes a procedure that acts as a function having no properties other than its name. By default, a literal function is defined to take one real argument and produce

[1]The procedure `print-expression` can be used in a program to print a simplified version of an expression. The default printer in the user interface incorporates the simplifier.

one real value. For example, we may want to work with a function $f : \mathbf{R} \to \mathbf{R}$:

```
((literal-function 'f) 'x)
(f x)

((compose (literal-function 'f) (literal-function 'g)) 'x)
(f (g x))
```

We can also make literal functions of multiple, possibly structured arguments that return structured values. For example, to denote a literal function named `g` that takes two real arguments and returns a real value ($g : \mathbf{R} \times \mathbf{R} \to \mathbf{R}$) we may write:

```
(define g (literal-function 'g (-> (X Real Real) Real)))

(g 'x 'y)
(g x y)
```

We may use such a literal function anywhere that an explicit function of the same type may be used.

There is a whole language for describing the type of a literal function in terms of the number of arguments, the types of the arguments, and the types of the values. Here we describe a function that maps pairs of real numbers to real numbers with the expression `(-> (X Real Real) Real)`. Later we will introduce structured arguments and values and show extensions of literal functions to handle these.

Tuples

There are two kinds of tuples: *up* tuples and *down* tuples. We write tuples as ordered lists of their components; a tuple is delimited by parentheses if it is an up tuple and by square brackets if it is a down tuple. For example, the up tuple v of velocity components v^0, v^1, and v^2 is

$$v = \left(v^0, v^1, v^2\right). \tag{B.4}$$

The down tuple p of momentum components p_0, p_1, and p_2 is

$$p = [p_0, p_1, p_2]. \tag{B.5}$$

A component of an up tuple is usually identified by a superscript. A component of a down tuple is usually identified by a subscript.

We use zero-based indexing when referring to tuple elements. This notation follows the usual convention in tensor arithmetic.

We make tuples with the constructors `up` and `down`:

```
(define v (up 'v^0 'v^1 'v^2))

v
(up v^0 v^1 v^2)

(define p (down 'p_0 'p_1 'p_2))

p
(down p_0 p_1 p_2)
```

Note that `v^0` and `p_2` are just symbols. The caret and underline characters are symbol constituents, so there is no meaning other than mnemonic to the structure of these symbols. However, our software can also display expressions using TeX, and then these decorations turn into superscripts and subscripts.

Tuple arithmetic is different from the usual tensor arithmetic in that the components of a tuple may also be tuples and different components need not have the same structure. For example, a tuple structure s of phase-space states is

$$s = (t, (x, y), [p_x, p_y]) . \tag{B.6}$$

It is an up tuple of the time, the coordinates, and the momenta. The time t has no substructure. The coordinates are an up tuple of the coordinate components x and y. The momentum is a down tuple of the momentum components p_x and p_y. In Scheme this is written:

```
(define s (up 't (up 'x 'y) (down 'p_x 'p_y)))
```

In order to reference components of tuple structures there are selector functions, for example:

$$\begin{aligned} I(s) &= s \\ I_0(s) &= t \\ I_1(s) &= (x, y) \\ I_2(s) &= [p_x, p_y] \\ I_{1,0}(s) &= x \\ &\cdots \end{aligned}$$

$$I_{2,1}(s) = p_y. \tag{B.7}$$

The sequence of integer subscripts on the selector describes the access chain to the desired component.

The procedure `component` is the general selector procedure that implements the selector function I_z:

```
((component 0 1) (up (up 'a 'b) (up 'c 'd)))
b
```

To access a component of a tuple we may also use the selector procedure `ref`, which takes a tuple and an index and returns the indicated element of the tuple:

```
(ref (up 'a 'b 'c) 1)
b
```

We use zero-based indexing everywhere. The procedure `ref` can be used to access any substructure of a tree of tuples:

```
(ref (up (up 'a 'b) (up 'c 'd)) 0 1)
b
```

Two up tuples of the same length may be added or subtracted, elementwise, to produce an up tuple, if the components are compatible for addition. Similarly, two down tuples of the same length may be added or subtracted, elementwise, to produce a down tuple, if the components are compatible for addition.

Any tuple may be multiplied by a number by multiplying each component by the number. Numbers may, of course, be multiplied. Tuples that are compatible for addition form a vector space.

For convenience we define the square of a tuple to be the sum of the squares of the components of the tuple. Tuples can be multiplied, as described below, but the square of a tuple is not the product of the tuple with itself.

The meaning of multiplication of tuples depends on the structure of the tuples. Two tuples are compatible for contraction if they are of opposite types, they are of the same length, and corresponding elements have the following property: either they are both tuples and are compatible for contraction, or at least one is not a tuple. If two tuples are compatible for contraction then generic multiplication is interpreted as contraction: the result is the sum of the products of corresponding components of the tuples. For example, p and v introduced in equations (B.4) and (B.5) above are compatible for contraction; the product is

$$pv = p_0 v^0 + p_1 v^1 + p_2 v^2. \tag{B.8}$$

So the product of tuples that are compatible for contraction is an inner product. Using the tuples `p` and `v` defined above gives us

```
(* p v)
(+ (* p_0 v^0) (* p_1 v^1) (* p_2 v^2))
```

Contraction of tuples is commutative: $pv = vp$. Caution: Multiplication of tuples that are compatible for contraction is, in general, not associative. For example, let $u = (5, 2)$, $v = (11, 13)$, and $g = [[3, 5], [7, 9]]$. Then $u(gv) = 964$, but $(ug)v = 878$. The expression ugv is ambiguous. An expression that has this ambiguity does not occur in this book.

The rule for multiplying two structures that are not compatible for contraction is simple. If A and B are not compatible for contraction, the product AB is a tuple of type B whose components are the products of A and the components of B. The same rule is applied recursively in multiplying the components. So if $B = (B^0, B^1, B^2)$, the product of A and B is

$$AB = \left(AB^0, AB^1, AB^2\right). \tag{B.9}$$

If A and C are not compatible for contraction and $C = [C_0, C_1, C_2]$, the product is

$$AC = [AC_0, AC_1, AC_2]. \tag{B.10}$$

Tuple structures can be made to represent linear transformations. For example, the rotation commonly represented by the matrix

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{B.11}$$

can be represented as a tuple structure:[2]

$$\left[\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix} \right]. \tag{B.12}$$

Such a tuple is compatible for contraction with an up tuple that represents a vector. So, for example:

$$\left[\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix} \right] \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{pmatrix}. \tag{B.13}$$

The product of two tuples that represent linear transformations—which are not compatible for contraction—represents the composition of the linear transformations. For example, the product of the tuples representing two rotations is

$$\begin{aligned} &\left[\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix} \right] \left[\begin{pmatrix} \cos\varphi \\ \sin\varphi \end{pmatrix} \begin{pmatrix} -\sin\varphi \\ \cos\varphi \end{pmatrix} \right] \\ &\qquad = \left[\begin{pmatrix} \cos(\theta+\varphi) \\ \sin(\theta+\varphi) \end{pmatrix} \begin{pmatrix} -\sin(\theta+\varphi) \\ \cos(\theta+\varphi) \end{pmatrix} \right]. \end{aligned} \tag{B.14}$$

Multiplication of tuples that represent linear transformations is associative but generally not commutative, just as the composition of the transformations is associative but not generally commutative.

Derivatives

The derivative of a function f is a function, denoted by Df. Our notational convention is that D is a high-precedence operator. Thus D operates on the adjacent function before any other application occurs: $Df(x)$ is the same as $(Df)(x)$. Higher-order derivatives are described by exponentiating the derivative operator. Thus the nth derivative of a function f is notated as $D^n f$.

The Scheme procedure for producing the derivative of a function is named `D`. The derivative of the `sin` procedure is a procedure that computes `cos`:

[2] To emphasize the relationship of simple tuple structures to matrix notation we often format `up` tuples as vertical arrangements of components and `down` tuples as horizontal arrangements of components. However, we could just as well have written this tuple as $[(\cos\theta, \sin\theta), (-\sin\theta, \cos\theta)]$.

```
(define derivative-of-sine (D sin))

(derivative-of-sine 'x)
(cos x)
```

The derivative of a function f is the function Df whose value for a particular argument is something that can be multiplied by an increment Δx in the argument to get a linear approximation to the increment in the value of f:

$$f(x + \Delta x) \approx f(x) + Df(x)\Delta x. \tag{B.15}$$

For example, let f be the function that cubes its argument ($f(x) = x^3$); then Df is the function that yields three times the square of its argument ($Df(y) = 3y^2$). So $f(5) = 125$ and $Df(5) = 75$. The value of f with argument $x + \Delta x$ is

$$f(x + \Delta x) = (x + \Delta x)^3 = x^3 + 3x^2\Delta x + 3x\Delta x^2 + \Delta x^3 \tag{B.16}$$

and

$$Df(x)\Delta x = 3x^2\Delta x. \tag{B.17}$$

So $Df(x)$ multiplied by Δx gives us the term in $f(x+\Delta x)$ that is linear in Δx, providing a good approximation to $f(x+\Delta x)-f(x)$ when Δx is small.

Derivatives of compositions obey the chain rule:

$$D(f \circ g) = ((Df) \circ g) \cdot Dg. \tag{B.18}$$

So at x,

$$(D(f \circ g))(x) = Df(g(x)) \cdot Dg(x). \tag{B.19}$$

`D` is an example of an *operator*. An operator is like a function except that multiplication of operators is interpreted as composition, whereas multiplication of functions is multiplication of the values (see equation B.3). If D were an ordinary function, then the rule for multiplication would imply that D^2f would just be the product of Df with itself, which is not what is intended. A product of a number and an operator scales the operator. So, for example

```
(((* 5 D) cos) 'x)
(* -5 (sin x))
```

Arithmetic is extended to allow manipulation of operators. A typical operator is $(D+I)(D-I) = D^2-I$, where I is the identity operator, subtracts a function from its second derivative. Such an operator can be constructed and used in Scheme as follows:

```
(((* (+ D I) (- D I)) (literal-function 'f)) 'x)
(+ (((expt D 2) f) x) (* -1 (f x)))
```

Derivatives of Functions of Multiple Arguments

The derivative generalizes to functions that take multiple arguments. The derivative of a real-valued function of multiple arguments is an object whose contraction with the tuple of increments in the arguments gives a linear approximation to the increment in the function's value.

A function of multiple arguments can be thought of as a function of an up tuple of those arguments. Thus an incremental argument tuple is an up tuple of components, one for each argument position. The derivative of such a function is a down tuple of the partial derivatives of the function with respect to each argument position.

Suppose we have a real-valued function g of two real-valued arguments, and we want to approximate the increment in the value of g from its value at x, y. If the arguments are incremented by the tuple $(\Delta x, \Delta y)$ we compute:

$$\begin{aligned} Dg(x,y) \cdot (\Delta x, \Delta y) &= [\partial_0 g(x,y), \partial_1 g(x,y)] \cdot (\Delta x, \Delta y) \\ &= \partial_0 g(x,y)\Delta x + \partial_1 g(x,y)\Delta y. \end{aligned} \tag{B.20}$$

Using the two-argument literal function `g` defined on page 200, we have:

```
((D g) 'x 'y)
(down (((partial 0) g) x y) (((partial 1) g) x y))
```

In general, partial derivatives are just the components of the derivative of a function that takes multiple arguments (or structured arguments or both; see below). So a partial derivative of a function is a composition of a component selector and the derivative of that function.[3] Indeed:

$$\partial_0 g = I_0 \circ Dg \tag{B.21}$$
$$\partial_1 g = I_1 \circ Dg. \tag{B.22}$$

Concretely, if

$$g(x, y) = x^3 y^5 \tag{B.23}$$

then

$$Dg(x, y) = \left[3x^2y^5, 5x^3y^4\right] \tag{B.24}$$

and the first-order approximation of the increment for changing the arguments by Δx and Δy is

$$\begin{aligned} g(x + \Delta x, y + \Delta y) - g(x, y) &\approx \left[3x^2y^5, 5x^3y^4\right] \cdot (\Delta x, \Delta y) \\ &= 3x^2y^5\Delta x + 5x^3y^4\Delta y. \end{aligned} \tag{B.25}$$

Partial derivatives of compositions also obey a chain rule:

$$\partial_i(f \circ g) = ((Df) \circ g) \cdot \partial_i g. \tag{B.26}$$

So if x is a tuple of arguments, then

$$(\partial_i(f \circ g))(x) = Df(g(x)) \cdot \partial_i g(x). \tag{B.27}$$

Mathematical notation usually does not distinguish functions of multiple arguments and functions of the tuple of arguments. Let $h((x, y)) = g(x, y)$. The function h, which takes a tuple of arguments x and y, is not distinguished from the function g that takes arguments x and y. We use both ways of defining functions of multiple arguments. The derivatives of both kinds of functions are compatible for contraction with a tuple of increments to the arguments. Scheme comes in handy here:

[3]Partial derivative operators such as `(partial 2)` are operators, so `(expt (partial 1) 2)` is a second partial derivative.

```
(define (h s)
  (g (ref s 0) (ref s 1)))

(h (up 'x 'y))
(g x y)

((D g) 'x 'y)
(down (((partial 0) g) x y) (((partial 1) g) x y))

((D h) (up 'x 'y))
(down (((partial 0) g) x y) (((partial 1) g) x y))
```

A phase-space state function is a function of time, coordinates, and momenta. Let H be such a function. The value of H is $H(t, (x, y), [p_x, p_y])$ for time t, coordinates (x, y), and momenta $[p_x, p_y]$. Let s be the phase-space state tuple as in (B.6):

$$s = (t, (x, y), [p_x, p_y]) . \tag{B.28}$$

The value of H for argument tuple s is $H(s)$. We use both ways of writing the value of H.

We often show a function of multiple arguments that include tuples by indicating the boundaries of the argument tuples with semicolons and separating their components with commas. If H is a function of phase-space states with arguments t, (x, y), and $[p_x, p_y]$, we may write $H(t; x, y; p_x, p_y)$. This notation loses the up/down distinction, but our semicolon-and-comma notation is convenient and reasonably unambiguous.

The derivative of H is a function that produces an object that can be contracted with an increment in the argument structure to produce an increment in the function's value. The derivative is a down tuple of three partial derivatives. The first partial derivative is the partial derivative with respect to the numerical argument. The second partial derivative is a down tuple of partial derivatives with respect to each component of the up-tuple argument. The third partial derivative is an up tuple of partial derivatives with respect to each component of the down-tuple argument:

$$\begin{aligned} DH(s) &= [\partial_0 H(s), \partial_1 H(s), \partial_2 H(s)] \\ &= [\partial_0 H(s), [\partial_{1,0} H(s), \partial_{1,1} H(s)], (\partial_{2,0} H(s), \partial_{2,1} H(s))] , \end{aligned} \tag{B.29}$$

where $\partial_{1,0}$ indicates the partial derivative with respect to the first component (index 0) of the second argument (index 1) of the function, and so on. Indeed, $\partial_z F = I_z \circ DF$ for any function F and access chain z. So, if we let Δs be an incremental phase-space state tuple,

$$\Delta s = \left(\Delta t, \left(\Delta x, \Delta y\right), \left[\Delta p_x, \Delta p_y\right]\right), \tag{B.30}$$

then

$$\begin{aligned} DH(s)\Delta s = {} & \partial_0 H(s)\Delta t \\ & + \partial_{1,0} H(s)\Delta x + \partial_{1,1} H(s)\Delta y \\ & + \partial_{2,0} H(s)\Delta p_x + \partial_{2,1} H(s)\Delta p_y. \end{aligned} \tag{B.31}$$

Caution: Partial derivative operators with respect to different structured arguments generally do not commute.

In Scheme we must make explicit choices. We usually assume phase-space state functions are functions of the tuple. For example,

```
(define H
  (literal-function 'H
    (-> (UP Real (UP Real Real) (DOWN Real Real)) Real)))

(H s)
(H (up t (up x y) (down p_x p_y)))

((D H) s)
(down
 (((partial 0) H) (up t (up x y) (down p_x p_y)))
 (down (((partial 1 0) H) (up t (up x y) (down p_x p_y)))
       (((partial 1 1) H) (up t (up x y) (down p_x p_y))))
 (up (((partial 2 0) H) (up t (up x y) (down p_x p_y)))
     (((partial 2 1) H) (up t (up x y) (down p_x p_y)))))
```

Structured Results

Some functions produce structured outputs. A function whose output is a tuple is equivalent to a tuple of component functions each of which produces one component of the output tuple.

For example, a function that takes one numerical argument and produces a structure of outputs may be used to describe a curve through space. The following function describes a helical path around the $\hat{z}$-axis in 3-dimensional space:

$$h(t) = (\cos t, \sin t, t) = (\cos, \sin, I)(t). \tag{B.32}$$

The derivative is just the up tuple of the derivatives of each component of the function:

$$Dh(t) = (-\sin t, \cos t, 1). \tag{B.33}$$

In Scheme we can write

```
(define (helix t)
  (up (cos t) (sin t) t))
```

or just

```
(define helix (up cos sin identity))
```

Its derivative is just the up tuple of the derivatives of each component of the function:

```
((D helix) 't)
(up (* -1 (sin t)) (cos t) 1)
```

In general, a function that produces structured outputs is just treated as a structure of functions, one for each of the components. The derivative of a function of structured inputs that produces structured outputs is an object that when contracted with an incremental input structure produces a linear approximation to the incremental output. Thus, if we define function g by

$$g(x, y) = ((x + y)^2, (y - x)^3, e^{x+y}), \tag{B.34}$$

then the derivative of g is

$$Dg(x, y) = \left[\begin{pmatrix} 2(x+y) \\ -3(y-x)^2 \\ e^{x+y} \end{pmatrix}, \begin{pmatrix} 2(x+y) \\ 3(y-x)^2 \\ e^{x+y} \end{pmatrix} \right]. \tag{B.35}$$

In Scheme:

```
(define (g x y)
  (up (square (+ x y)) (cube (- y x)) (exp (+ x y))))
```

```
((D g) 'x 'y)
(down (up (+ (* 2 x) (* 2 y))
          (+ (* -3 (expt x 2)) (* 6 x y) (* -3 (expt y 2)))
          (* (exp y) (exp x)))
      (up (+ (* 2 x) (* 2 y))
          (+ (* 3 (expt x 2)) (* -6 x y) (* 3 (expt y 2)))
          (* (exp y) (exp x))))
```

Exercise B.1: Chain Rule

Let $F(x, y) = x^2y^3$, $G(x, y) = (F(x, y), y)$, and $H(x, y) = F(F(x, y), y)$, so that $H = F \circ G$.

a. Compute $\partial_0 F(x, y)$ and $\partial_1 F(x, y)$.

b. Compute $\partial_0 F(F(x, y), y)$ and $\partial_1 F(F(x, y), y)$.

c. Compute $\partial_0 G(x, y)$ and $\partial_1 G(x, y)$.

d. Compute $DF(a, b)$, $DG(3, 5)$ and $DH(3a^2, 5b^3)$.

Exercise B.2: Computing Derivatives

We can represent functions of multiple arguments as procedures in several ways, depending upon how we wish to use them. The simplest idea is to identify the procedure arguments with the function's arguments.

For example, we could write implementations of the functions that occur in exercise B.1 as follows:

```
(define (f x y)
  (* (square x) (cube y)))

(define (g x y)
  (up (f x y) y))

(define (h x y)
  (f (f x y) y))
```

With this choice it is awkward to compose a function that takes multiple arguments, such as f, with a function that produces a tuple of those arguments, such as g. Alternatively, we can represent the function arguments as slots of a tuple data structure, and then composition with a function that produces such a data structure is easy. However, this choice requires the procedures to build and take apart structures.

For example, we may define procedures that implement the functions above as follows:

```
(define (f v)
  (let ((x (ref v 0))
        (y (ref v 1)))
    (* (square x) (cube y))))
```

```
(define (g v)
  (let ((x (ref v 0))
        (y (ref v 1)))
    (up (f v) y)))

(define h (compose f g))
```

Repeat exercise B.1 using the computer. Explore both implementations of multiple-argument functions.

C
Tensors

There are a variety of objects that have meaning independent of any particular basis. Examples are form fields, vector fields, covariant derivative, and so on. We call objects that are independent of basis *geometric objects.* Some of these are functions that take other geometric objects, such as vector fields and form fields, as arguments and produce further geometric objects. We refer to such functions as *geometric functions.* We want the laws of physics to be independent of the coordinate systems. How we describe an experiment should not affect the result. If we use only geometric objects in our descriptions then this is automatic.

A geometric function of vector fields and form fields that is linear in each argument with functions as multipliers is called a *tensor.* For example, let T be a geometric function of a vector field and form field that gives a real-number result at the manifold point m. Then

$$\mathsf{T}(\mathsf{f}\mathsf{u} + \mathsf{g}\mathsf{v}, \boldsymbol{\omega}) = \mathsf{f}\,\mathsf{T}(\mathsf{u}, \boldsymbol{\omega}) + \mathsf{g}\mathsf{T}(\mathsf{v}, \boldsymbol{\omega}) \tag{C.1}$$

$$\mathsf{T}(\mathsf{u}, \mathsf{f}\boldsymbol{\omega} + \mathsf{g}\boldsymbol{\theta}) = \mathsf{f}\,\mathsf{T}(\mathsf{u}, \boldsymbol{\omega}) + \mathsf{g}\mathsf{T}(\mathsf{u}, \boldsymbol{\theta}), \tag{C.2}$$

where u and v are vector fields, $\boldsymbol{\omega}$ and $\boldsymbol{\theta}$ are form fields, and f and g are manifold functions. That a tensor is linear over functions and not just constants is important.

The multilinearity over functions implies that the components of the tensor transform in a particularly simple way as the basis is changed. The components of a real-valued geometric function of vector fields and form fields are obtained by evaluating the function on a set of basis vectors and their dual form basis. In our example,

$$\mathsf{T}^i_j = \mathsf{T}(\mathsf{e}_j, \tilde{\mathsf{e}}^i), \tag{C.3}$$

for basis vector fields e_j and dual form fields $\tilde{\mathsf{e}}^i$. On the left, T^i_j is a function of place (manifold point); on the right, T is a function of a vector field and a form field that returns a function of place.

Now we consider a change of basis, $\mathsf{e}(\mathsf{f}) = \mathsf{e}'(\mathsf{f})\,\mathsf{J}$ or

$$\mathsf{e}_i(\mathsf{f}) = \sum_j \mathsf{e}'_j(\mathsf{f})\,\mathsf{J}^j_i, \tag{C.4}$$

where J typically depends on place. The corresponding dual basis transforms as

$$\tilde{\mathsf{e}}^i(\mathsf{v}) = \sum_j \mathsf{K}^i_j\,\tilde{\mathsf{e}}'^j(\mathsf{v}), \tag{C.5}$$

where $\mathsf{K} = \mathsf{J}^{-1}$ or $\sum_j \mathsf{K}^i_j(\mathsf{m})\mathsf{J}^j_k(\mathsf{m}) = \delta^i_k$.

Because the tensor is multilinear over functions, we can deduce that the tensor components in the two bases are related by, in our example,

$$\mathsf{T}^i_j = \sum_{kl} \mathsf{K}^i_k \mathsf{T}'^k_l \mathsf{J}^l_j. \tag{C.6}$$

or

$$\mathsf{T}'^i_j = \sum_{kl} \mathsf{J}^i_k \mathsf{T}^k_l \mathsf{K}^l_j. \tag{C.7}$$

Tensors are a restricted set of mathematical objects that are geometric, so if we restict our descriptions to tensor expressions they are *prima facie* independent of the coordinates used to represent them. So if we can represent the physical laws in terms of tensors we have built in the coordinate-system independence.

Let's test whether the geometric function R, which we have called the Riemann tensor (see equation 8.2), is indeed a tensor field. A real-valued geometric function is a tensor if it is linear (over the functions) in each of its arguments. We can try it for 3-dimensional rectangular coordinates:

```
(let ((cs R3-rect))
  (let ((u (literal-vector-field 'u-coord cs))
        (v (literal-vector-field 'v-coord cs))
        (w (literal-vector-field 'w-coord cs))
        (x (literal-vector-field 'x-coord cs))
        (omega (literal-1form-field 'omega-coord cs))
        (nu (literal-1form-field 'nu-coord cs))
        (f (literal-manifold-function 'f-coord cs))
        (g (literal-manifold-function 'g-coord cs))
        (nabla (covariant-derivative (literal-Cartan 'G cs)))
        (m (typical-point cs)))
    (let ((F (Riemann nabla)))
      ((up (- (F (+ (* f omega) (* g nu)) u v w)
              (+ (* f (F omega u v w)) (* g (F nu u v w))))
           (- (F omega (+ (* f u) (* g x)) v w)
              (+ (* f (F omega u v w)) (* g (F omega x v w))))
           (- (F omega v (+ (* f u) (* g x)) w)
              (+ (* f (F omega v u w)) (* g (F omega v x w))))
           (- (F omega v w (+ (* f u) (* g x)))
              (+ (* f (F omega v w u)) (* g (F omega v w x)))))
       m))))
(up 0 0 0 0)
```

Now that we are convinced that the Riemann tensor is indeed a tensor, we know how its components change under a change of basis. Let

$$\mathsf{R}^i_{jkl} = \mathsf{R}(\tilde{\mathsf{e}}^i, \mathsf{e}_j, \mathsf{e}_k, \mathsf{e}_l), \tag{C.8}$$

then

$$\mathsf{R}^i_{jkl} = \sum_{mnpq} \mathsf{K}^i_m \mathsf{R}'^m_{npq} \mathsf{J}^n_j \mathsf{J}^p_k \mathsf{J}^q_l. \tag{C.9}$$

or

$$\mathsf{R}'^i_{jkl} = \sum_{mnpq} \mathsf{J}^i_m \mathsf{R}^m_{npq} \mathsf{K}^n_j \mathsf{K}^p_k \mathsf{K}^q_l. \tag{C.10}$$

Whew!

It is easy to generalize these formulas to tensors with general arguments. We have formulated the general tensor test as a program `tensor-test` that takes the procedure `T` to be tested, a list of argument types, and a coordinate system to be used. It tests each argument for linearity (over functions). If the function passed as `T` is a tensor, the result will be a list of zeros.

So, for example, `Riemann` proves to be a tensor

```
(tensor-test
 (Riemann (covariant-derivative (literal-Cartan 'G R3-rect)))
 '(1form vector vector vector)
 R3-rect)
(0 0 0 0)
```

and so is the torsion (see equation 8.21):

```
(tensor-test
 (torsion (covariant-derivative (literal-Cartan 'G R3-rect)))
 '(1form vector vector)
 R3-rect)
(up 0 0 0)
```

But not all geometric functions are tensors. The covariant derivative is an interesting and important case. The function F, defined by

$$\mathsf{F}(\boldsymbol{\omega}, \mathsf{u}, \mathsf{v}) = \boldsymbol{\omega}(\nabla_{\mathsf{u}}\mathsf{v}), \tag{C.11}$$

is a geometric object, since the result is independent of the coordinate system used to represent the ∇. For example

```
(define ((F nabla) omega u v)
  (omega ((nabla u) v)))

(((- (F (covariant-derivative
         (Christoffel->Cartan
          (metric->Christoffel-2
           (coordinate-system->metric S2-spherical)
           (coordinate-system->basis S2-spherical)))))
     (F (covariant-derivative
         (Christoffel->Cartan
          (metric->Christoffel-2
           (coordinate-system->metric S2-stereographic)
           (coordinate-system->basis S2-stereographic))))))
  (literal-1form-field 'omega S2-spherical)
  (literal-vector-field 'u S2-spherical)
  (literal-vector-field 'v S2-spherical))
 ((point S2-spherical) (up 'theta 'phi)))
0
```

But it is not a tensor field:

```
(tensor-test
 (F (covariant-derivative (literal-Cartan 'G R3-rect)))
 '(1form vector vector)
 R3-rect)
(0 0 MESS)
```

This result tells us that the function F is linear in its first two arguments but not in its third argument.

That the covariant derivative is not linear over functions in the second vector argument is easy to understand. The first vector argument takes derivatives of the coefficients of the second vector argument, so multiplying these coefficients by a manifold function changes the derivative.

References

[1] Harold Abelson and Gerald Jay Sussman with Julie Sussman, *Structure and Interpretation of Computer Programs*, MIT Press, Cambridge, MA, 1996.

[2] R. L. Bishop and S. I. Goldberg, *Tensor Analysis on Manifolds*, MacMillan, New York, 1968.

[3] S. Carroll, *Spacetime and Geometry: An Introduction to General Relativity*, Benjamin Cummings, 2003.

[4] Alonzo Church, *The Calculi of Lambda-Conversion*, Princeton University Press, 1941.

[5] Harley Flanders, *Differential Forms with Applications to the Physical Sciences*, Academic Press, New York, 1963, Dover, New York, 1989.

[6] Theodore Frankel, *The Geometry of Physics*, Cambridge University Press, 1997.

[7] Galileo Galilei, *Il Saggiatore (The Assayer)*, 1623.

[8] S. W. Hawking and G. .F. .R. Ellis, *The Large Scale Structure of Space–Time*, Cambridge University Press, 1973.

[9] IEEE Std 1178-1990, *IEEE Standard for the Scheme Programming Language*, Institute of Electrical and Electronic Engineers, Inc., 1991.

[10] Charles W. Misner, Kip S. Thorne, and John Archibald Wheeler, *Gravitation*, W. H. Freeman and Company, San Francisco, 1973.

[11] Abraham Pais, *Subtle is the Lord: The Science and the Life of Albert Einstein*, Oxford University Press, Oxford UK, 1982.

[12] Seymour A. Papert, *Mindstorms: Children, Computers, and Powerful Ideas*, Basic Books, 1980.

[13] B. Schutz *A First Course in General Relativity*, Cambridge University Press, 1985.

[14] I. M. Singer and John A. Thorpe, *Lecture Notes on Elementary Topolgy and Geometry*, Scott, Foresman and Company, Glenview, Illinois, 1967.

[15] Michael Spivak, *A Comprehensive Introduction to Differential Geometry*, Publish or Perish, Houston, Texas, 1970.

[16] Michael Spivak, *Calculus on Manifolds*, W. A. Benjamin, New York, NY, 1965.

[17] Gerald Jay Sussman and Jack Wisdom, *The Role of Programming in the Formulation of Ideas*, Artificial Intelligence Laboratory memo AIM-2002-018, November 2002.

[18] Gerald Jay Sussman and Jack Wisdom with Meinhard E. Mayer, *Structure and Interpretation of Classical Mechanics*, MIT Press, Cambridge, MA, 2001.

[19] Robert M. Wald, *General Relativity*, University of Chicago Press, 1984.

[20] Free software is available at:
`groups.csail.mit.edu/mac/users/gjs/6946/linux-install.htm`.

Index

Any inaccuracies in this index may be explained by the fact that it has been prepared with the help of a computer.

Donald E. Knuth, *Fundamental Algorithms* (Volume 1 of *The Art of Computer Programming*)

Page numbers for Scheme procedure definitions are in italics.
Page numbers followed by *n* indicate footnotes.