Functional Differential Geometry

Gerald Jay Sussman and Jack Wisdom

May 2, 2022

Contents

> 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

1 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 Mis´erables 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.

1.1 Acknowledgements

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. Indeed, *Turtle Geometry* [2], a beautiful book about discrete differential geometry at a more elementary level, was inspired by Papert's work on education. [13]

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 inspired us to reorganize and rewrite major parts of the text. She has also developed and maintained Gerald Jay Sussman over these many years.

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

2 Prologue

2.1 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 Leibniz'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?

[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 [13].

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.

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 ω 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 $b = a$.

2.2 Functional Abstraction

But this corrected use of Leibniz 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

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

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 $\partial_2 L$ $\partial_1 L$, 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.

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]

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

[3] That the symbols q and $\dot{q}$ can be replaced by other arbitrarily chosen nonconflicting 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.

[5] An explanation of functional derivatives is in Appendix B, page 202.

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:

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

Using Γ we can write:

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

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:

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

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]

```
(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,

[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 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 [10]. You can get the software from [21].

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

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

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:

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

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))

;; should produce \cos(\omega t + \varphi) a (k-m\omega^2)
```

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))
;; should produce $$kx(t)+mD^2 x(t)$$
```

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.

3 Chapter 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 [8]

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[7] for the free-particle Lagrangian with coordinates that incorporate the constraint of living in the surface. There are coefficients of

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

terms in the EulerLagrange 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 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.

3.1 Lagrange Equations

We write the Lagrange equations in functional notation[8] as follows:

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

In SICM [19], 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:[9]

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

[8] 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

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

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 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)))
```

or, in infix notation:

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

3.2 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

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

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

$$\begin{aligned}\mathsf{d}\theta(\mathsf{v}) &= \dot{\theta}\\ \mathsf{d}\phi(\mathsf{v}) &= \dot{\phi},\end{aligned}$$

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:

$$L_2(x,v) = \sum_{ij}\frac{1}{2}mg_{ij}(x)v^iv^j \tag{3}$$

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:[10]

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

The metric is expressed in rectangular coordinates, so the coordinate system is `R2-rect`.[11] The component functions will be labeled as subscripted =g=s.

[10] 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.

[11] `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.

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).

3.3 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:[12]

```
(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 $\mathbb{R}^2$ 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 $\mathbb{R}^2$ to its coordinate representation we use `((chart R2-rect) m)`. (The procedures point and chart are introduced in Chapter 2.) Thus

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

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

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

[12] 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.

Now we can compute the residuals of the Euler-Lagrange equations, but we get a large messy expression that we will not show.[13] However, we will save it to compare with the residuals of the geodesic equations.

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

3.4 Geodesic Equations

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

$$\nabla_{\mathsf{v}}\mathsf{v} = 0 \tag{4}$$

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

$$\nabla^{\gamma}_{\partial/\partial\mathsf{t}} d\gamma\,(\partial/\partial\mathsf{t}) = 0. \tag{5}$$

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

In coordinates, the geodesic equations are expressed

$$D^2 q^i(t) + \sum_{jk} \Gamma^i_{jk}(\gamma(t)) Dq^j(t) Dq^k(t) = 0, \tag{6}$$

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

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[14] 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:

```
(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)
```

[14] 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.

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).

3.4.1 Exercise 1.1: Motion on a Sphere

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

$$\mathsf{g}(\mathsf{u},\mathsf{v}) = \mathsf{d}\theta(\mathsf{u})\mathsf{d}\theta(\mathsf{v}) + (\sin\theta)^2\mathsf{d}\phi(\mathsf{u})\mathsf{d}\phi(\mathsf{v}). \tag{7}$$

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).