

Lecture 12: Multivariable Calculus

Reading:

Kreyszig Sections: 9.5, 9.6, 9.7 (pages 389–398, 400–403, 403–409)

The Calculus of Curves

In the last lecture, the derivatives of a vector that varied continuously with a parameter, $\vec{r}(t)$, were considered. It is natural to think of $\vec{r}(t)$ as a curve in whatever space the vector $\vec{r}$ is defined. The most familiar example is a curve in the plane: the two values $(x(t), y(t))$ are mapped onto the plane through values as t sweeps through its range $t_{\text{initial}} \leq t \leq t_{\text{final}}$. A curve in three-dimensional cartesian space is the mapping of three values $(x(t), y(t), z(t))$; in cylindrical coordinates: $(r(t), \theta(t), z(t))$. In general, a curve is represented by N coordinates as a *single parameter* (i.e., t) takes on a range of numbers—the N coordinates form the embedding space.

Objects that have more dimensions than curves need more parameters. The number of parameters is the dimensionality of the object and the number of coordinates is the dimensionality of the embedding space. What we naturally call a *surface* is a two dimensional object embedded in a three-dimensional space—for example, in cartesian coordinates $(x(u, v), y(u, v), z(u, v))$ is a surface.

The two-dimensional surface $(x(u, v), y(u, v), z(u, v))$ can itself become an embedding space for lower dimensional objects; for example, the curve $(u(t), v(t))$ is embedded in the surface (u, v) which itself embedded in (x, y, z) . In other words, the curve $(x(u(t), v(t)), y(u(t), v(t)), z(u(t), v(t)))$ can be considered to be embedded in (u, v) , or embedded in (x, y, z) and constrained to the surface $(x(u, v), y(u, v), z(u, v))$.

In higher dimensions, there are many more possibilities and we can make a few introductory remarks about the language that is used to describe them. For application to physical problems, these considerations indicate the number of degrees-of-freedom that are available and the conditions that a system is overconstrained. An N -dimensional surface (sometimes called a *hyper-surface*) embedded in an M -dimensional space is said to have *codimension* $M - N$. Some objects cannot be embedded in

[3.016 Home](#)[Full Screen](#)[Close](#)[Quit](#)

a higher dimensional space; these are called *non-embeddable*, examples include the Klein bottle which cannot be embedded in our three-dimensional space.

Lecture 12 MATHEMATICA® Example 1

Curves in Three Dimensions

notebook (non-evaluated)

pdf (evaluated)

html (evaluated)

Two examples of parametric curves are presented with a visualization technique that animates the vector as it sweeps out a curve.

- 2: This is the second of two example functions that take an argument and return a vector at the argument. It is a vector function that takes a scalar argument.
- 3: The function *showcurve* takes two arguments. The first argument is the name of a vector function that takes a single argument, as in the above examples. The second argument is the upper limit of time to be plotted; i.e., it will sweep out a curve from $t = 0$ to $t = \text{upper-limit}$. The option **DisplayFunction** is a replacement to **Identity**, so the graphics will not be displayed.
- 4: The function *showline* takes the same two arguments, and it creates a graphics-object for a line drawn from the origin to the point indicated by the second argument.
- 5: The function *showcurveline* takes the same two arguments and calls the previously defined functions *showcurve* and *showline* with the graphics displayed by using **DisplayFunction** $\rightarrow$ **\$DisplayFunction**. The result should be a line pointing to a curve that is swept out from the time $t = 0$.
- 6: *CurveLineSequence* just calls *showcurveline* for a fixed interval.
- 8: This is the second example of an animation showing the vector as it sweeps out a curve.

```

1 PrettyFlower[t_] :=
  {1/4 + 3/4 Cos[3 t], Cos[t]^3, Sin[t] Cos[t]^2}
2 Bendy[t_] := {Cos[t], Sin[t], Sin[t] Cos[t]}
Display Functions
3 showcurve[VecFunc_, tl_] := ParametricPlot3D[
  Evaluate[VecFunc[tval]], {tval, 0, tl}, Compiled -> False,
  DisplayFunction -> Identity, PlotRange ->
  {{-1, 1}, {-1, 1}, {-1, 1}}, BoxRatios -> {1, 1, 1}]
4 showline[VecFunc_, tl_] := Graphics3D[
  {Thickness[0.01], Hue[1], Line[{0, 0, 0}, VecFunc[tl]]}]
5 showcurveline[VecFunc_, tl_] :=
  Show[showcurve[VecFunc, tl], showline[VecFunc, tl],
  DisplayFunction -> $DisplayFunction]
6 CurveLineSequence[VecFunc_] :=
  Table[showcurveline[VecFunc, i], {i, 1, 3 Pi, .1}]
Animating the Curves with Their Parameter
7 CurveLineSequence[PrettyFlower];
8 CurveLineSequence[Bendy];

```

3.016 Home



Full Screen

Close

Quit

Embedding Curves in Surfaces

notebook (non-evaluated)

pdf (evaluated)

html (evaluated)

An example is constructed that visualizes a two-dimensional surface in three dimensions and then visualizes a one-dimensional curve constrained to the surface.

- 1: *FlowerPot* takes two arguments and returns a vector. As the arguments sweep through domains, the vector will trace out a surface.
- 3: Using the `ParametricPlot3D` that is in the `Graphics`ParametricPlot`` package, the surface defined by `FlowerPlot` can be visualized.
- 4: *Vines* takes a single argument and then calls `FlowerPot` with arguments that are functions of that single argument—the result must be a curve embedded in the surface. In this case, the function is scaled a little, so the curves will be visible.
- 5: This is an example that makes the curve fat and green colored.
- 6: Here, both the embedded curve and the surface are shown together.

```

1 FlowerPot[u_, v_] := {(3 + Cos[v]) Cos[u],
   Sin[u] + (3 + Cos[v]) Sin[u], (3/2 + Cos[u + v]) Sin[v]}
2 << Graphics`ParametricPlot3D`
3 Flowerplot = ParametricPlot3D[FlowerPot[u, v], {u, 0, 2 Pi},
   {v, 0, 2 Pi}, ViewPoint -> {0.141, 1.653, 1.117},
   PlotPoints -> {120, 40}]
4 Vines[t_] := 1.025 * FlowerPot[t Cos[t], -t^2 Sin[t]]
   vineplot = ParametricPlot3D[Vines[t], {t, 0, 2 Pi},
   ViewPoint -> {0.141, 1.653, 1.117}, PlotPoints -> 500]
5 thickvineplot = Show[Graphics3D[Thickness[0.02]],
   Graphics3D[Hue[0.333, 0.5, 0.5]], vineplot]
6 Show[thickvineplot, Flowerplot]
7 Show[Flowerplot, thickvineplot]

```

3.016 Home



Full Screen

Close

Quit

Because the derivative of a curve with respect to its parameter is a tangent vector, the unit tangent can be defined immediately:

$$\hat{u} = \frac{\frac{d\vec{r}}{dt}}{\left\| \frac{d\vec{r}}{dt} \right\|} \quad (12-1)$$

It is convenient to find a new parameter, $s(t)$, that would make the denominator in Eq. 12-1 equal to one. This parameter, $s(t)$, is the arc-length:

$$\begin{aligned} s(t) &= \int_{t_o}^t ds \\ &= \int_{t_o}^t \sqrt{dx^2 + dy^2 + dz^2} \\ &= \int_{t_o}^t \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2} dt \\ &= \int_{t_o}^t \sqrt{\left(\frac{d\vec{r}}{dt}\right) \cdot \left(\frac{d\vec{r}}{dt}\right)} dt \end{aligned} \quad (12-2)$$

and with s instead of t ,

$$\hat{u}(s) = \frac{d\vec{r}}{ds} \quad (12-3)$$

This is natural because $\|\vec{r}\|$ and s have the same units (i.e., meters and meters, fots and feet, etc) instead of, for instance, time, t , that makes $d\vec{r}/dt$ a velocity and involving two different kinds of units (e.g., furlongs and hours).

With the arc-length s , the magnitude of the curvature is particularly simple,

$$\kappa(s) = \left\| \frac{d\hat{u}}{ds} \right\| = \left\| \frac{d^2\vec{r}}{ds^2} \right\| \quad (12-4)$$

as is its interpretation—the curvature is a measure of how rapidly the unit tangent is changing direction.

Furthermore, the rate at which the unit tangent changes direction is a vector that must be normal to the tangent (because $d(\hat{u} \cdot \hat{u}) = 0$) and therefore the unit normal is defined by:

$$\hat{p}(s) = \frac{1}{\kappa(s)} \frac{d\hat{u}}{ds} \quad (12-5)$$

There two unit vectors that are locally normal to the unit tangent vector $\hat{u}'(s)$ and the curve unit normal $\hat{p}(s) \times \hat{u}$ and $\hat{u}(s) \times \hat{p}$. This last choice is called the unit binormal, $\hat{b} \equiv \hat{u}(s) \times \hat{p}$ and the three vectors together form a nice little moving orthogonal axis pinned to the curve. Furthermore, there are convenient relations between the vectors and differential geometric quantities called the Frenet equations.

Using Arc-Length as a Curve's Parameter

However, it should be pointed out that—although re-parameterizing a curve in terms of its arc-length makes for simple analysis of a curve—achieving this re-parameterization is not necessarily simple.

Consider the steps required to represent a curve $\vec{r}(t)$ in terms of its arc-length:

[3.016 Home](#)

integration The integral in Eq. 12-2 may or may not have a closed form for $s(t)$.

If it does then we can perform the following operation:

inversion $s(t)$ is typically a complicated function that is not easy to invert, i.e., solve for t in terms of s to get a $t(s)$ that can be substituted into $\vec{r}(t(s)) = \vec{r}(s)$.



These difficulties usually result in treating the problem symbolically and the resorting to numerical methods of achieving the integration and inversion steps.

[Full Screen](#)
[Close](#)
[Quit](#)

Calculating arclength

notebook (non-evaluated)

pdf (evaluated)

html (evaluated)

Examples of computing a curve's arc-length s

- 1: Here, the tangent vector for the function, *PrettyFlower* defined above, is computed.
- 2: This is an attempt to find a closed-form solution for arclength $s(\tau) - s(0) = \int_0^\tau \sqrt{\left(\frac{dc}{dt}\right)^2} dt$. A closed-form doesn't exist.
- 4: However, a closed-form solution does exist for the *Bendy*-function defined earlier. If the closed-form $s(t)$ could be inverted (i.e., $t(s)$) then the curve $c(t)$ could be expressed in terms of its natural variable $c(s) = c(t(s))$.
- 5: The plot, $s(t)$ is monotonically increasing and therefore, the function could always be inverted numerically.
- 6: Even for the arc-length that could not be evaluated in closed-form (i.e., *PrettyFlower*), a numerical integration could be used to perform the inversion.

```

1 dFlowerDt = Simplify[D[PrettyFlower[t], t]]
2 sFlower = Integrate[Sqrt[Simplify[dFlowerDt.dFlowerDt]], t]
3 dBendyDt = D[Bendy[t], t]
4 sBendy = Integrate[Sqrt[dBendyDt.dBendyDt], t]
5 Plot[sBendy, {t, 0, 2 Pi}]
6 Plot[Evaluate[NIntegrate[
  Sqrt[dFlowerDt.dFlowerDt], {t, 0, uplim}]], {uplim, 0, 6.4}]

```

3.016 Home



Full Screen

Close

Quit

Scalar Functions with Vector Argument

In materials science and engineering, the concept of a spatially varying function arises frequently:

For example:

Concentration $c_i(x, y, z) = c_i(\vec{x})$ is the number (or moles) of chemical species of type i *per unit volume* located at the point $\vec{x}$.

Density $\rho(x, y, z) = \rho(\vec{x})$ mass *per unit volume* located at the point $\vec{x}$. a point $\rho(x, y, z) = \rho(\vec{x})$.

Energy Density $u(x, y, z) = u(\vec{x})$ energy *per unit volume* located at the point $\vec{x}$.

The examples above are spatially dependent densities of “extensive quantities.”

There are also spatially variable intensive quantities:

Temperature $T(x, y, z) = T(\vec{x})$ is the temperature which would be measured at the point $\vec{x}$.

Pressure $P(x, y, z) = P(\vec{x})$ is the pressure which would be measured at the point $\vec{x}$.

Chemical Potential $\mu_i(x, y, z) = \mu_i(\vec{x})$ is the chemical potential of the species i which would be measured at the point $\vec{x}$.

Each example is a scalar function of space—that is, the function associates a scalar with each point in space.

A topographical map is a familiar example of a graphical illustration of a scalar function (altitude) as a function of location (latitude and longitude).

How Confusion Can Develop in Thermodynamics

However, there are many other types of scalar functions of several arguments, such as the state function: $U = U(S, V, N_i)$ or $P = P(V, T, N_i)$. It is sometimes useful to think of these types of functions a scalar functions of a “point” in a thermodynamics space.

However, this is often a source of confusion: notice that the internal energy is used in two different contexts above. One context is the value of the energy, say 128.2 Joules. The other context is the function $U(S, V, N_i)$. While the two symbols are identical, their meanings are quite different.

[3.016 Home](#)

[Full Screen](#)
[Close](#)
[Quit](#)

The root of the confusion lurks in the question, “What are the variables of U ?” Suppose that there is only one (independent) chemical species, then $U(\cdot)$ has three variables, such as $U(S, V, N)$. “But what if $S(T, P, \mu)$, $V(T, P, \mu)$, and $N(T, P, \mu)$ are known functions, what are the variables of U ?” The answer is, they are any three independent variables, one could write $U(T, P, \mu) = U(S(T, P, \mu), V(T, P, \mu), N(T, P, \mu))$ and there are six other natural choices.

The trouble arises when variations of a function like U are queried—then the variables that are varying must be specified.

For this reason, it is either a good idea to keep the functional form explicit in thermodynamics, i.e.,

$$\begin{aligned} dU(S, V, N) &= \frac{\partial U(S, V, N)}{\partial S} dS + \frac{\partial U(S, V, N)}{\partial V} dV + \frac{\partial U(S, V, N)}{\partial N} dN \\ dU(T, P, \mu) &= \frac{\partial U(T, P, \mu)}{\partial T} dT + \frac{\partial U(T, P, \mu)}{\partial V} dV + \frac{\partial U(T, P, \mu)}{\partial \mu} d\mu \end{aligned} \quad (12-6)$$

or use, the common thermodynamic notation,

$$\begin{aligned} dU &= \left(\frac{\partial U}{\partial S} \right)_{V, N} dS + \left(\frac{\partial U}{\partial V} \right)_{S, N} dV + \left(\frac{\partial U}{\partial N} \right)_{S, V} dN \\ dU &= \left(\frac{\partial U}{\partial T} \right)_{P, \mu} dT + \left(\frac{\partial U}{\partial P} \right)_{T, \mu} dP + \left(\frac{\partial U}{\partial \mu} \right)_{T, P} d\mu \end{aligned} \quad (12-7)$$

Total and Partial Derivatives, Chain Rule

There is no doubt that a great deal confusion arises from the following question, “What are the variables of my function?”

For example, suppose we have a three-dimensional space (x, y, z) , in which there is an embedded surface $(x(w, v), y(w, v), z(w, v))$ $\vec{x}(w, v) = \vec{x}(\vec{u})$ where $\vec{u} = (v, w)$ is a vector that lies in the surface, and an embedded curve $(x(s), y(s), z(s)) = \vec{x}(s)$. Furthermore, suppose there is a curve that lies within the surface $(w(t), v(t)) = \vec{u}(t)$.

Suppose that $\mathcal{E} = f(x, y, z)$ is a scalar function of (x, y, z) .

Here are some questions that arise in different applications:

[3.016 Home](#)

[Full Screen](#)
[Close](#)
[Quit](#)

1. How does $\mathcal{E}$ vary as a function of position?
2. How does $\mathcal{E}$ vary along the surface?
3. How does $\mathcal{E}$ vary along the curve?
4. How does $\mathcal{E}$ vary along the curve embedded in the surface?

3.016 Home



Full Screen

Close

Quit

Total Derivatives and Partial Derivatives: A Mathematica Review

notebook (non-evaluated)

pdf (evaluated)

html (evaluated)

Demonstrations of 1) the three spatial derivatives of $F(x, y, z)$; 2) the two independent derivatives on a two-dimensional surface embedded in $x-y-z$; 3) the complete derivative of $F(x, y, z)$ along a curve $(x(t), y(t), z(t))$.

- 1: *AScalarFunction* is a symbolic representation of a function—it will be a place-holder for examples of partial derivatives.
- 3: This will print Mathematica's representation of derivatives with respect to one of several arguments—e.g., $\partial F(x, y, z)/\partial y$ is written as $F^{(0,1,0)}[x,y,z]$.
- 4: *AScalarFunction* becomes a function of two-variables when x , y , and z are restricted to a surface parameterized by (u, v) : $(x(w, v), y(w, v), z(w, v))$
- 5: Caution: the distinction between the symbol x and the symbol $x[w, v]$ is important; the following two examples show how the derivatives should appear.
- 7: This and the previous example show how the chain rule is computed, these two terms are the components of the gradient in the surface.
- 8: In this case, the previous *AScalarFunction* becomes a function of a single variable by specifying a curve in the surface with $(w(t), v(t))$.
- 9: Now, a total derivative can be calculated with the chain rule. This is equivalent to...
- 10: The total derivative along a specific curve $(x(t), y(t), z(t))$.

```

1 AScalarFunction[x_, y_, z_] := SomeFunction[x, y, z]
2 AScalarFunction[x, y, z]

Print["derivative w/r to first argument is "];
dFuncX = D[AScalarFunction[x, y, z], x]
Print["derivative w/r to second argument is "];
dFuncY = D[AScalarFunction[x, y, z], y]
Print["derivative w/r to third argument is "];
dFuncZ = D[AScalarFunction[x, y, z], z]

4 AScalarFunction[x[w, v], y[w, v], z[w, v]]
5 D[AScalarFunction[x[w, v], y[w, v], z[w, v]], x]
6 dFuncW = D[AScalarFunction[x[w, v], y[w, v], z[w, v]], w]
7 dFuncV = D[AScalarFunction[x[w, v], y[w, v], z[w, v]], v]
8 AScalarFunction[x[w[t], v[t]], y[w[t], v[t]], z[w[t], v[t]]]
9 dFuncT =
D[AScalarFunction[x[w[t], v[t]], y[w[t], v[t]], z[w[t], v[t]], t]
10 dFuncT = D[AScalarFunction[x[t], y[t], z[t]], t]

```

3.016 Home



Full Screen

Close

Quit

Taylor Series

The behavior of a function near a point is something that arises frequently in physical models. When the function has lower-order continuous partial derivatives (generally, a “smooth” function near the point in question), the stock method to model local behavior is Taylor’s series expansions around a fixed point.

Taylor’s expansion for a scalar function of n variables, $f(x_1, x_2, \dots, x_n)$ which has continuous first and second partial derivatives near the point $\vec{\xi} = (\xi_1, \xi_2, \dots, \xi_n)$ is:

$$\begin{aligned}
 f(\xi_1, \xi_2, \dots, \xi_n) &= f(x_1, x_2, \dots, x_n) \\
 &+ \left. \frac{\partial f}{\partial x_1} \right|_{\vec{\xi}} (\xi_1 - x_1) + \left. \frac{\partial f}{\partial x_2} \right|_{\vec{\xi}} (\xi_2 - x_2) + \dots + \left. \frac{\partial f}{\partial x_n} \right|_{\vec{\xi}} (\xi_n - x_n) \\
 &+ \frac{1}{2} \left[\right. \\
 &\quad \left. \frac{\partial^2 f}{\partial x_1^2} \right|_{\vec{\xi}} (\xi_1 - x_1)^2 + \frac{\partial^2 f}{\partial x_1 \partial x_2} \Big|_{\vec{\xi}} (\xi_1 - x_1)(\xi_2 - x_2) + \dots + \frac{\partial^2 f}{\partial x_1 \partial x_n} \Big|_{\vec{\xi}} (\xi_1 - x_1)(\xi_n - x_n) \\
 &\quad + \frac{\partial^2 f}{\partial x_2 \partial x_1} \Big|_{\vec{\xi}} (\xi_2 - x_2)(\xi_1 - x_1) + \frac{\partial^2 f}{\partial x_2^2} \Big|_{\vec{\xi}} (\xi_2 - x_2)^2 + \dots + \frac{\partial^2 f}{\partial x_2 \partial x_n} \Big|_{\vec{\xi}} (\xi_2 - x_2)(\xi_n - x_n) \quad (12-8) \\
 &\quad \vdots \qquad \qquad \qquad \vdots \\
 &\quad + \frac{\partial^2 f}{\partial x_n \partial x_1} \Big|_{\vec{\xi}} (\xi_n - x_n)(\xi_1 - x_1) + \frac{\partial^2 f}{\partial x_n \partial x_2} \Big|_{\vec{\xi}} (\xi_n - x_n)(\xi_2 - x_2) + \dots + \frac{\partial^2 f}{\partial x_n^2} \Big|_{\vec{\xi}} (\xi_n - x_n)^2 \\
 &\quad \left. \right] \\
 &+ \mathcal{O}[(\xi_1 - x_1)^3] + \mathcal{O}[(\xi_1 - x_1)^2(\xi_2 - x_2)] + \mathcal{O}[(\xi_1 - x_1)(\xi_2 - x_2)^2] + \mathcal{O}[(\xi_2 - x_2)^3] \\
 &+ \dots + \mathcal{O}[(\xi_1 - x_1)^2(\xi_n - x_n)] + \mathcal{O}[(\xi_1 - x_1)(\xi_2 - x_2)(\xi_n - x_n)] + \dots + \mathcal{O}[(\xi_n - x_n)^3]
 \end{aligned}$$

or in a vector shorthand:

$$f(\vec{x}) = f(\vec{\xi}) + \nabla_{\vec{x}} f|_{\vec{x}=\vec{\xi}} \cdot (\vec{\xi} - \vec{x}) + (\vec{\xi} - \vec{x}) \cdot (\nabla_{\vec{x}} \nabla_{\vec{x}} f)|_{\vec{x}=\vec{\xi}} \cdot (\vec{x} - \vec{\xi}) + \mathcal{O}[\|\vec{\xi} - \vec{x}\|^3] \quad (12-9)$$

In the following example, visualization of local approximations will be obtained for a scalar function of two variables, $f(x, y)$. This will be extended into a an approximating function of four variables by

[3.016 Home](#)
[◀◀](#)
[◀](#)
[▶](#)
[▶▶](#)
[Full Screen](#)
[Close](#)
[Quit](#)

expanding it about a point (ξ, η) to second order. The expansion is now a function of four variables—the first two variables are the point the function is expanded around (x and y), and the second two are the variable of the parabolic approximation at that point (ξ and η): $f_{\text{appx}}(\xi, \eta; x, y) = f(x, y) + \left. \frac{\partial f}{\partial x} \right|_{x,y} (\xi - x) + \left. \frac{\partial f}{\partial y} \right|_{x,y} (\eta - y) + \mathcal{Q}$ where $\mathcal{Q} \equiv \frac{1}{2} \left. \frac{\partial^2 f}{\partial x^2} \right|_{x,y} (\xi - x)(\xi - x) + \left. \frac{\partial^2 f}{\partial x \partial y} \right|_{x,y} (\xi - x)(\eta - y) + \frac{1}{2} \left. \frac{\partial^2 f}{\partial y^2} \right|_{x,y} (\eta - y)(\eta - y)$ or $f_{\text{appx}}(\xi, \eta, x, y) = f(x, y) + \nabla f \cdot \begin{pmatrix} \xi - x \\ \eta - y \end{pmatrix} + \frac{1}{2} \mathcal{Q}_{\text{form}}$ where $\mathcal{Q}_{\text{form}} \equiv (\xi - x, \eta - y) \begin{pmatrix} \left. \frac{\partial^2 f}{\partial x^2} \right|_{x,y} & \left. \frac{\partial^2 f}{\partial x \partial y} \right|_{x,y} \\ \left. \frac{\partial^2 f}{\partial y \partial x} \right|_{x,y} & \left. \frac{\partial^2 f}{\partial y^2} \right|_{x,y} \end{pmatrix} \begin{pmatrix} \xi - x \\ \eta - y \end{pmatrix}$

The function $f_{\text{appx}}(\xi, \eta, x, y)$ will be plotted as a function of ξ and η for $|\xi - x| < \delta$ and $|\eta - y| < \delta$ for a selected number of points (x, y) .

[3.016 Home](#)

[Full Screen](#)
[Close](#)
[Quit](#)

Approximating Surfaces at Points

notebook (non-evaluated)

pdf (evaluated)

html (evaluated)

Visualization of a quadratic approximations to a surface at points on that surface

- 1: *CrazyFun* is an example function of two variables.
- 3: Using `Normal` to convert the Taylor Expansion obtained by `Series` at an point x_o, y_o produces a function *Approxfunction* of four variables.
- 5: This illustrates how the local quadratic approximation fits the surface locally at a *particular* point.
- 6: Generate a list of random points at which to visualize the local approximation.
- 7: *ApproxPlot* is an example that will plot the local approximation for any indexed random point. The surface is colored by using the value of the point as an indicator. Visualization is delayed: only a graphics object is produced.
- 8: This is an example of producing a stack of graphics with a *recursive graphics function*. It iteratively adds a new approximating surface graphics object to the set of the previous ones.
- 9: `GraphicsArray` allows plots to be drawn in rows and columns. Here, intermediate output is produced and displayed, and then the approximating surfaces are plotted on the left of the surfaces with the original surface.

```

1 CrazyFun[x_, y_] := Sin[5 π x] Sin[5 π y] / (x y) +
  Sin[5 π (x - 1)] Sin[5 π (y - 1)] / ((x - 1) (y - 1))
2 theplot = Plot3D[CrazyFun[x, y], {x, 0, 1}, {y, 0, 1},
  {y, 0, 1}, PlotRange → All, Mesh → False]
3 Approxfunction[x_, y_, xo_, yo_] :=
  Series[CrazyFun[x, y], {x, xo, 2}, {y, yo, 2}] // Normal
4 anapprox = Plot3D[Evaluate[Approxfunction[x, y, .7, .1]],
  {x, .7 - .1, .7 + .1}, {y, .1 - .1, .1 + .1}]
5 Show[anapprox, theplot]
6 Table[{xol[i] = Random[], yol[i] = Random[]}, {i, 1, 100}];
7 ApproxPlot[i_] :=
  Plot3D[Evaluate[Approxfunction[x, y, xol[i], yol[i]],
  {x, xol[i] - .1, xol[i] + .1},
  {y, yol[i] - .1, yol[i] + .1}, PlotPoints → 6,
  ColorFunction → (RGBColor[0.9 xol[i], 0.9 yol[i], #] &),
  DisplayFunction → Identity]
8 GraphicsStack[0] =
  Show[ApproxPlot[1], DisplayFunction → Identity]
  GraphicsStack[i_] := GraphicsStack[i] =
  Show[GraphicsStack[i - 1], ApproxPlot[i + 1]]
9 Show[GraphicsArray[
  (GraphicsStack[10], Show[theplot, GraphicsStack[10]]),
  DisplayFunction → $DisplayFunction]

```

3.016 Home



Full Screen

Close

Quit

Just a few of many examples of instances where Taylor's expansions are used are:

linearization Examining the behavior of a model near a point where the model is understood. Even if the model is wildly non-linear, a useful approximation is to make it linear by evaluating near a fixed point.

approximation If a model has a complicated representation in terms of unfamiliar functions, a Taylor expansion can be used to characterize the 'local' model in terms of a simple polynomials.

asymptotics Even when a system has singular behavior (e.g, the value of a function becomes infinite as some variable approaches a particular value), *how* the system becomes singular is very important. At singular points, the Taylor expansion will have leading order terms that are singular, for example near $x = 0$,

$$\frac{\sin(x)}{x^2} = \frac{1}{x} - \frac{x}{6} + \mathcal{O}(x^3) \quad (12-10)$$

The singularity can be subtracted off and it can be said that this function approaches ∞ "linearly" from below with slope -1/6. Comparing this to the behavior of another function that is singular near zero:

$$\frac{\exp(x)}{x} = \frac{1}{x} + 1 + \frac{x}{2} + \frac{x^2}{6} + \mathcal{O}(x^3) \quad (12-11)$$

shows that the e^x/x behavior is "more singular."

3.016 Home



Full Screen

Close

Quit

```
Plot[{Sin[x]/x^2 - 1/x, Exp[x]/x - 1/x}, {x, .001, 2.5},
PlotStyle -> {{Thickness[0.02], Hue[1]}, {Thickness[0.01], Hue[0.5]}}]
```

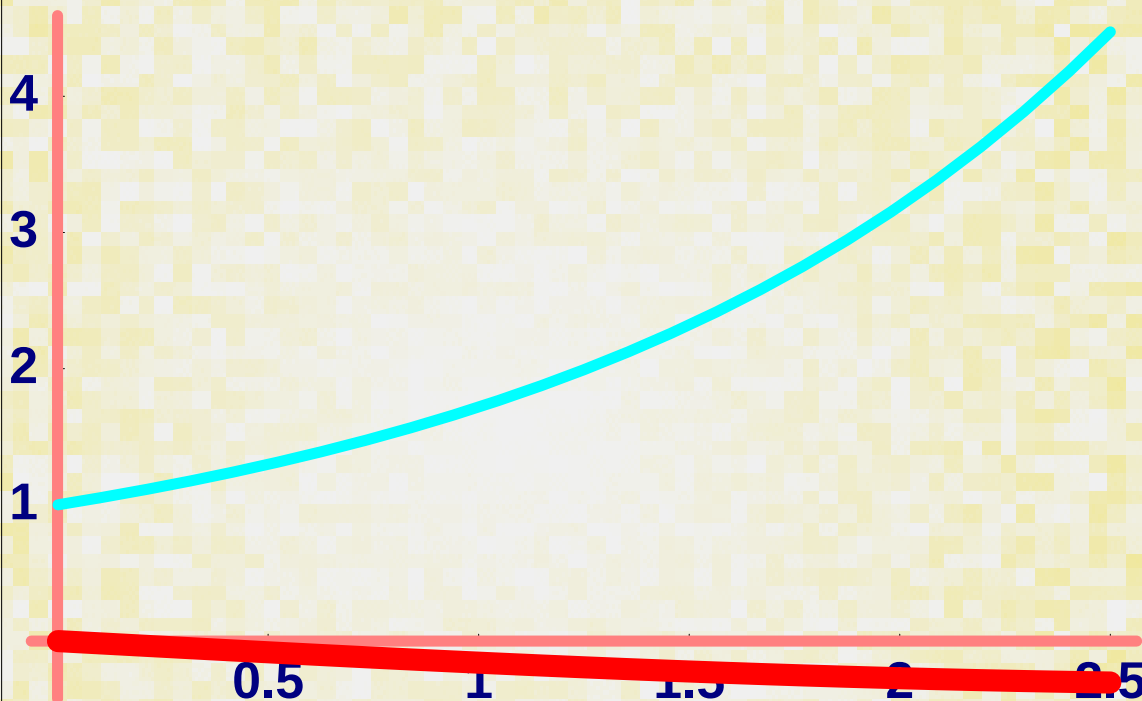


Figure 12-9: Behavior of two singular functions near their singular points.

stability In the expansion of energy about a point is obtained, then the various orders of expansion can be interpreted:

zero-order The zeroth-order term in a local expansion is the energy of the system at the point ©W. Craig Carter

evaluated. Unless this term is to be compared to another point, it has no particular meaning (if it is not infinite) as energy is always arbitrarily defined up to a constant.

first-order The first-order is related to the driving force to change the state of the system. Consider:

$$\Delta E = \nabla E \cdot \delta \vec{x} = -\vec{F} \cdot \delta \vec{x} \quad (12-12)$$

If force exists, the system can decrease its energy linearly by picking a particular change $\delta \vec{x}$ that is anti-parallel to the force.

For a system to be stable, it is a **necessary first condition that the forces (or first order expansion coefficients) vanish**.

second order If a system has no forces on it (therefore satisfying the necessary condition of stability), then where the system is stable or unstable depends on whether a small $\delta \vec{x}$ can be found that decreases the energy:

$$\begin{aligned} \Delta E &= \frac{1}{2} \delta \vec{x} \cdot \nabla (-\vec{F} \cdot \delta \vec{x}) \\ &= \frac{1}{2} \cdot \nabla \nabla E \cdot \delta \vec{x} \\ &= \frac{1}{2} \frac{\partial^2 E}{\partial x_i \partial x_j} \bigg|_{x_1, x_2, \dots, x_n} \delta x_i \delta x_j \end{aligned} \quad (12-13)$$

where the summation convention is used above and the point $(x_1, x_2, \dots, x_n)$ is one for which ∇E is zero.

numerics Derivatives are often obtained numerically in numerical simulations. The Taylor expansion provides a formula to approximate numerical derivatives—and provides an estimate of the numerical error as a function of quantities like numerical mesh size.

Gradients and Directional Derivatives

Scalar functions $F(x, y, z) = F(\vec{x})$ have a natural vector field associated with them—at each point $\vec{x}$ there is a direction $\hat{n}(\vec{x})$ pointing in the direction of the most rapid increase of F . Associating the

magnitude of a vector in the direction of steepest increase with the *rate of increase* of F defines the gradient.

The gradient is therefore a vector function with a vector argument ($\vec{x}$ in this case) and it is commonly written as ∇F .

Here are some natural examples:

Meteorology The “high pressure regions” are commonly displayed with weather reports—as are the “isobars” or curves of constant barometric pressure. Thus displayed, pressure is a scalar function of latitude and longitude.

At any point on the map, there is a direction that points to local high pressure center—this is the direction of the gradient. The rate at which the pressure is increasing gives the magnitude of the gradient.

The gradient of pressure should be a vector that is related to the direction and the speed of wind.

Mosquitoes It is known that hungry mosquitoes tend to fly towards sources (or local maxima) of carbon dioxide. Therefore, it can be hypothesized that mosquitoes are able to determine the gradient in the concentration of carbon dioxide.

Heat In an isolated system, heat flows from high-temperature ($T(\vec{x})$) regions to low-temperature regions.

The Fourier empirical law of heat flow states that the rate of heat flows is proportional to the local *decrease* in temperature.

Therefore, the local rate of heat flow should be a proportional to the vector which is *minus* the gradient of $T(\vec{x})$: $-\nabla T$

Finding the Gradient

Potentials and Force Fields

Force is a vector. Force projected onto a displacement vector $\vec{dx}$ is the rate at which work, dW , is done on an object $dW = -\vec{F} \cdot \vec{dx}$.

[3.016 Home](#)

[Full Screen](#)
[Close](#)
[Quit](#)

If the work is being supplied by an external agent (e.g., a charged sphere, a black hole, a magnet, etc.), then it may be possible to ascribe a potential energy ($E(\vec{x})$, a scalar function with vector argument) to the agent associated with the position at which the force is being applied.⁵ This $E(\vec{x})$ is the potential for the agent and the force field due to the agent is $\vec{F}(\vec{x}) = -\nabla E(\vec{x})$.

Sometimes the force (and therefore the energy) scale with the “size” of the object (i.e., the object’s total charge in an electric potential due to a fixed set of charges, the mass of an object in the gravitational potential of a black hole, the magnetization of the object in a magnetic potential, etc.). In these cases, the potential field can be defined in terms of a unit size (per unit charge, per unit mass, etc.). One can determine whether such a scaling is applied by checking the units.

[3.016 Home](#)

[Full Screen](#)
[Close](#)
[Quit](#)

⁵As with any energy, there is always an arbitrary constant associated with the position (or state) at which the energy is taken to be zero. There is no such ambiguity with force. Forces are, in a sense, more fundamental than energies. Energy appears to be fundamental because all observations of the first law of thermodynamics demonstrate that there is a conserved quantity which is a state function and is called energy.

Index

Approxfunction, [106](#)
ApproxPlot, [106](#)
 arclength, [97](#)
 as a parameter, [98](#)
AScalarFunction, [103](#)

Bendy, [99](#)

 calculus of many variables, [101](#)
 chain rule
 for several variables, [103](#)
 codimension, [94](#)
CrazyFun, [106](#)
 curvature
 formula in terms of arclength, [97](#)
 curvature vector, [97](#)
 curve
 local orthonormal frame, [98](#)
CurveLineSequence, [95](#)
 curves and surfaces, [94](#)
 displaying together
 example, [96](#)

 delayed graphics
 example, [95](#)
 density fields of extensive quantities, [100](#)
 derivatives of scalar functions, [100](#)
DisplayFunction, [95](#)
DisplayFunction $\rightarrow$ $\$DisplayFunction$, [95](#)

embedded curve, [101](#)
 embedded curves in surfaces
 visualization example, [96](#)
 embedded surface, [101](#)
 embedding space, [94](#)
 Example function
 AScalarFunction, [103](#)
 ApproxPlot, [106](#)
 Approxfunction, [106](#)
 Bendy, [99](#)
 CrazyFun, [106](#)
 CurveLineSequence, [95](#)
 FlowerPot, [96](#)
 PrettyFlower, [99](#)
 Vines, [96](#)
 showcurveline, [95](#)
 showcurve, [95](#)
 showline, [95](#)
 extensive quantities
 density fields, [100](#)

 fields of intensive quantities, [100](#)
FlowerPot, [96](#)
 Frenet equations, [98](#)

 gradients, [103](#), [109](#)
 Graphics‘ParametricPlot‘, [96](#)
 GraphicsArray, [106](#)

 heat flux and temperature gradients, [110](#)

hyper-surface, [94](#)

Identity, [95](#)

integration along curve

using arclength, [98](#)

intensive fields

chemical potential, [100](#)

pressure, [100](#)

temperature, [100](#)

inverting parametric form of curve, [98](#)

isobars and the weather, [110](#)

line integration, [98](#)

linearization, [107](#)

local orthonormal frame on curve, [98](#)

Mathematica function

DisplayFunction $\rightarrow$ `$DisplayFunction`, [95](#)

DisplayFunction, [95](#)

GraphicsArray, [106](#)

Identity, [95](#)

Normal, [106](#)

ParametricPlot3D, [96](#)

Series, [106](#)

Mathematica package

Graphics`ParametricPlot`, [96](#)

mosquitoes, [110](#)

non-embeddable, [95](#)

Normal, [106](#)

ParametricPlot3D, [96](#)

partial derivatives, [103](#)

potentials and force fields, [110](#)

PrettyFlower, [99](#)

recursive graphics function, [106](#)

scalar function of positions

example

concentration, [100](#)

density, [100](#)

energy density, [100](#)

Series, [106](#)

showcurve, [95](#)

showcurveline, [95](#)

showline, [95](#)

stability of a system, [109](#)

surface gradients, [103](#)

Taylor series, [104](#)

vector form, [104](#)

thermodynamics, [100](#)

thermodynamic notation, [101](#)

topographical map, [100](#)

total derivatives, [103](#)

unit binormal, [98](#)

unit tangent to curve, [97](#)

vector functions

example plots, [95](#)

Vines, [96](#)