

微分形式の引き戻し 1

Joh @物理のかぎプロジェクト

2006-11-13

この記事では、引き戻しと呼ばれる概念を説明します。順を追って理解していけば、それほど難しいことではないと思いますが、ここで躓く人が多いのも事実ですので、ゆっくり慌てずに消化していった下さい。この記事の内容には、微分形式はほとんど出てきません。写像の話ばかりですので、線形代数が得意な人は、この記事は飛ばしても大丈夫だと思います。微分形式の引き戻し 2 で、引き戻しを微分形式に適用することを考えます。

座標変換の一般論

まず、『全微分は座標系によらない』という話から始めましょう。このことは [外微分](#) で勉強しましたが、例えば関数 f を xyz デカルト座標系で表現しても、 $r\theta\phi$ 球座標系で表現しても全微分は同じでした。

$$\begin{aligned} df &= \frac{\partial f}{\partial x} dx + \frac{\partial f}{\partial y} dy + \frac{\partial f}{\partial z} dz \\ &= \frac{\partial f}{\partial r} dr + \frac{\partial f}{\partial \theta} d\theta + \frac{\partial f}{\partial \phi} d\phi \end{aligned} \quad (1)$$

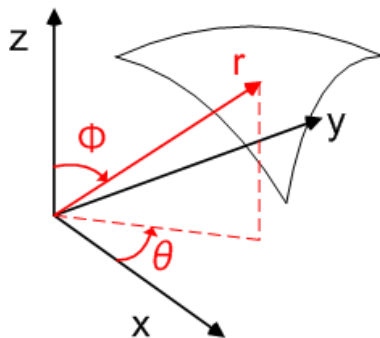
ただし、このような二種類の表現が出来るというのは、 (x, y, z) と (r, θ, ϕ) の間に滑らかな座標変換が定義されているからです。普通のデカルト座標と球座標ならば、次式の変換を考えることが出来ます。

$$x = r \cos \theta \sin \phi \quad (2-1)$$

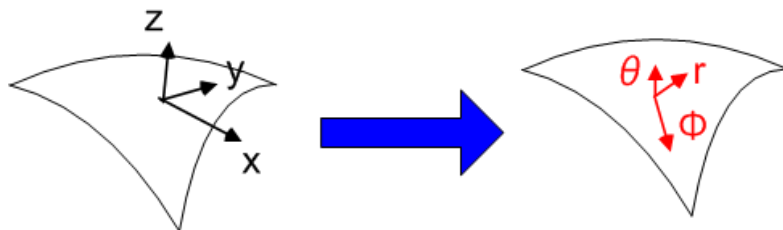
$$y = r \sin \theta \sin \phi \quad (2-2)$$

$$z = r \cos \phi \quad (2-3)$$

*1 先ほど、『滑らかな座標変換』と書きましたが、変換のヤコビアンが退化しないためには、少なくとも C^1 級の関数が必要です。普通は、何回でも微分できる座標変換が望まれます。『え！？無限回も微分しちゃっていいの？』と思う人がいるかも知れませんが、もし微分の回数に制限があったら、式 (2-1)(2-2)(2-3) のような座標変換を考えられません。(三角関数は何回でも微分できますよね。) 無限回微分できるような関数を、数学では C^∞ 級関数と呼びます。無限という言葉にどうしても抵抗を感じる人は、『必要なだけ微分できる』とおけば良いです。実際に無限回も微分することはあまりないと思います。



さて、この座標変換を、上図のように、一つの空間上に二つの座標系を重ねて考える人がいると思いますが（デカルト座標系を黒で、球座標系を赤で書きました）、ここでは次図のように『座標変換とは (x, y, z) 系の世界から、 (r, θ, ϕ) 系の世界への写像である』と考えることにします。（本当は全く同じことなんですが、このように見たほうが、より現代数学的です。あくまでも直観的イメージの問題です。写像という考え方に馴れて欲しいのです。）



次元を変えても良い

前セクションでは、デカルト座標系 (x, y, z) から、球座標系 (r, θ, ϕ) への座標変換という、物理でも馴染みの例を考えました。ここでは、特定の座標系の形は忘れて、最初の座標系を (x_1, x_2, x_3) と書くことにします。これを、異なる三次元の座標系 (x'_1, x'_2, x'_3) に移したのでは面白くないので、二次元の座標系 (u, v) に写像することを考えて見ましょう。『え！？違う次元にしちゃっていいの？』と思う人がいるかも知れませんが、 (x_1, x_2, x_3) を次式のように表現できれば、ちゃんと写像があるという意味ですから、いいんです。

$$x_1 = x_1(u, v) \quad (3-1)$$

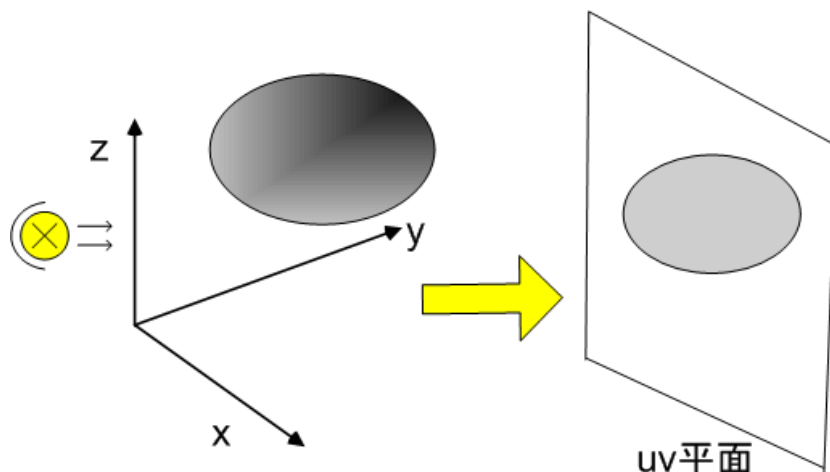
$$x_2 = x_2(u, v) \quad (3-2)$$

$$x_3 = x_3(u, v) \quad (3-3)$$

この座標変換のヤコビ行列は 2×3 行列の形になります。

$$\begin{pmatrix} du \\ dv \end{pmatrix} = \begin{pmatrix} \frac{\partial u}{\partial x_1} & \frac{\partial u}{\partial x_2} & \frac{\partial u}{\partial x_3} \\ \frac{\partial v}{\partial x_1} & \frac{\partial v}{\partial x_2} & \frac{\partial v}{\partial x_3} \end{pmatrix} \begin{pmatrix} dx_1 \\ dx_2 \\ dx_3 \end{pmatrix} \quad (4)$$

三次元を二次元に写像する場合というのは、例えば (x, y, z) 空間内の立体図形に光を当て、 uv 平面に影を映す様子を想像すれば了解できると思います。



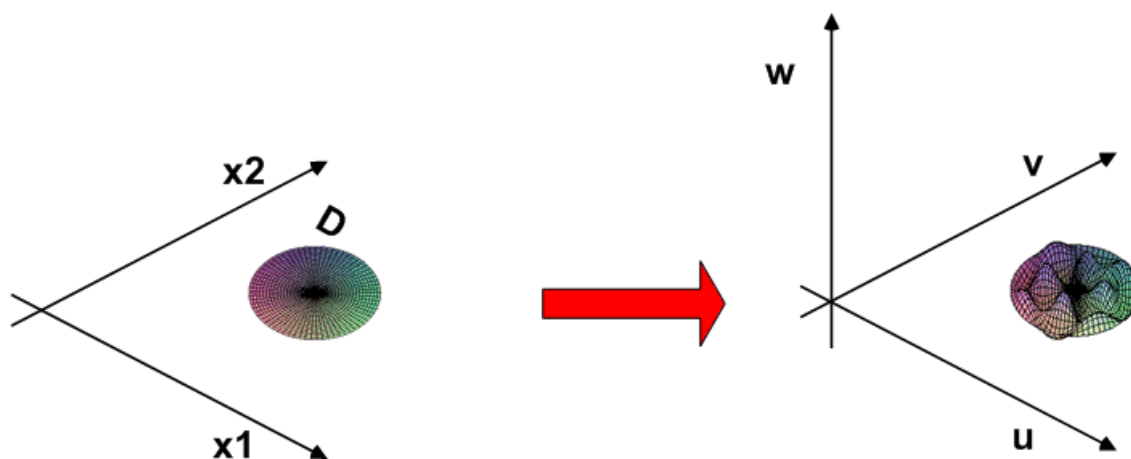
次元を上げてみてもいい

前セクションの逆で、今度は、二次元 (x_1, x_2) で考えていたものを、三次元 (u, v, w) に移すことだって出来ます。もちろん、十分に滑らかな（できれば C^∞ 級の）写像があることを想定しています。この座標変換のヤコビ行列は 3×2 行列の形になります。

$$\begin{pmatrix} du \\ dv \\ dw \end{pmatrix} = \begin{pmatrix} \frac{\partial u}{\partial x_1} & \frac{\partial u}{\partial x_2} \\ \frac{\partial v}{\partial x_1} & \frac{\partial v}{\partial x_2} \\ \frac{\partial w}{\partial x_1} & \frac{\partial w}{\partial x_2} \end{pmatrix} \begin{pmatrix} dx_1 \\ dx_2 \end{pmatrix} \quad (5)$$

このような写像がある、と言っているのですから、安心して写像していれば良いのですが、二次元の物をいきなり次元を上げるのには抵抗があるかも知れません。しかし、写像が十分に滑らかであるというのは、このように像の次元を上げて、『膨らませて』も良いということなんですね。この場合も、式 (5) のヤコビアン^{*}の階数が退化せず、2 であることが必要（そして重要）です。

^{*}2 ただし、上図のようなイメージだけでは、ちょうど光線と平行になった直線が、 uv 上で一点になってしまったり、光線と平行な平面内の図形が全て線分になってしまったりする場合も含まれることに注意して下さい。あくまで、滑らかで連続な写像があるという事、言い換えれば、式 (4) のヤコビアン^{*}の階数が 2 である（退化していない）ことが、写像の条件になっています。直観的な理解のために、乱暴な絵を描きましたが、数式の要点も、しっかり理解して下さい。



関数の写像

さて、今まで変数の写像ばかり話をしてきましたが、関数の写像ということを考えます。(ここから混乱する人が多いので、ゆっくりしっかり考えて下さい。) 前の二つのセクションの内容を統合し、一般に、変数 $(x_1, x_2, \dots, x_m)$ を $(y_1, y_2, \dots, y_n)$ に変換する (m 次元 $\rightarrow$ n 次元) 場合を考えます。この座標変換を表わす写像を ϕ と書きましょう。(さっきの例の球面座標の角度 ϕ とは関係ありません。念のため。)

$$y_1 = \phi_1(x_1, x_2, \dots, x_m)$$

$$y_2 = \phi_2(x_1, x_2, \dots, x_m)$$

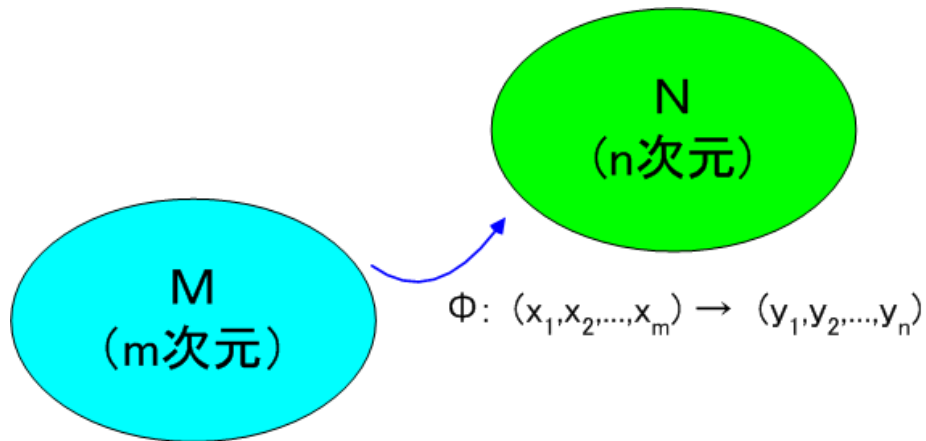
.....

$$y_n = \phi_n(x_1, x_2, \dots, x_m) \tag{6}$$

ここで、『変数 $(x_1, x_2, \dots, x_m)$ の世界』と『変数 $(y_1, y_2, \dots, y_n)$ の世界』は、簡単のためそれぞれ M と N で表わすことにします。 ϕ は、この二つの世界を結ぶ写像です。

*³ もし私が二次元生物だとして、『本当はお前は三次元生物の影なのだ』と言われたら、かなりショックだと思います。しかし、そんな物なのかも知れません。きっと、私達は四次元生物の影なのでしょう。近視眼的だというのは、悲しいことですね。ルルルララ♪

*⁴ もし多様体の概念を知っている人は、この M や N は多様体だと思ってください。ここで考えている話題は、多様体間の写像です。

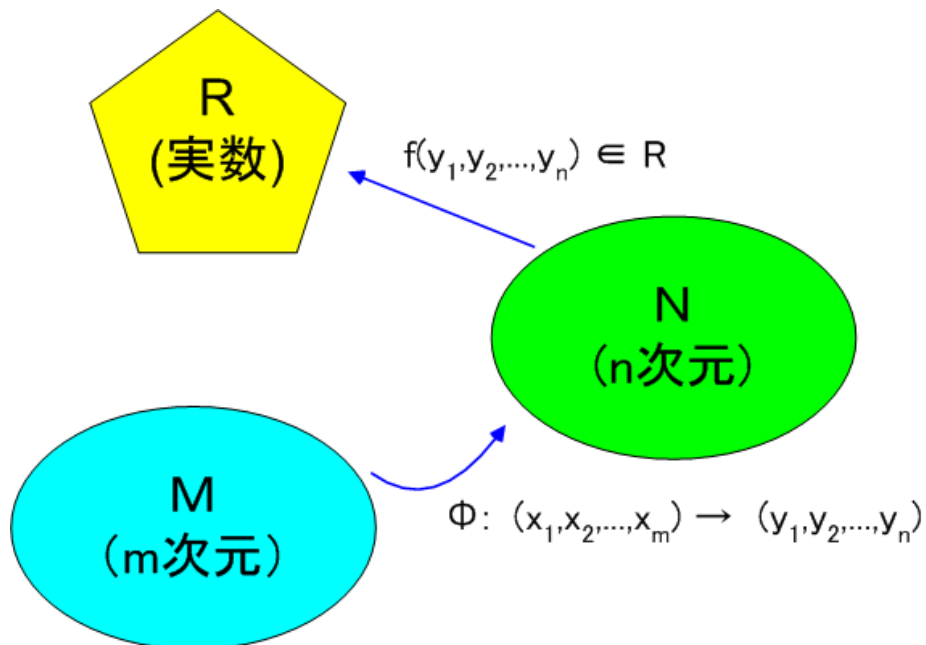


さて、いきなりですが、 N 上で定義される、ある関数 f を考えましょう。 f によって実数が一つ決まるとすると、 f は次のような写像であると考えられます。

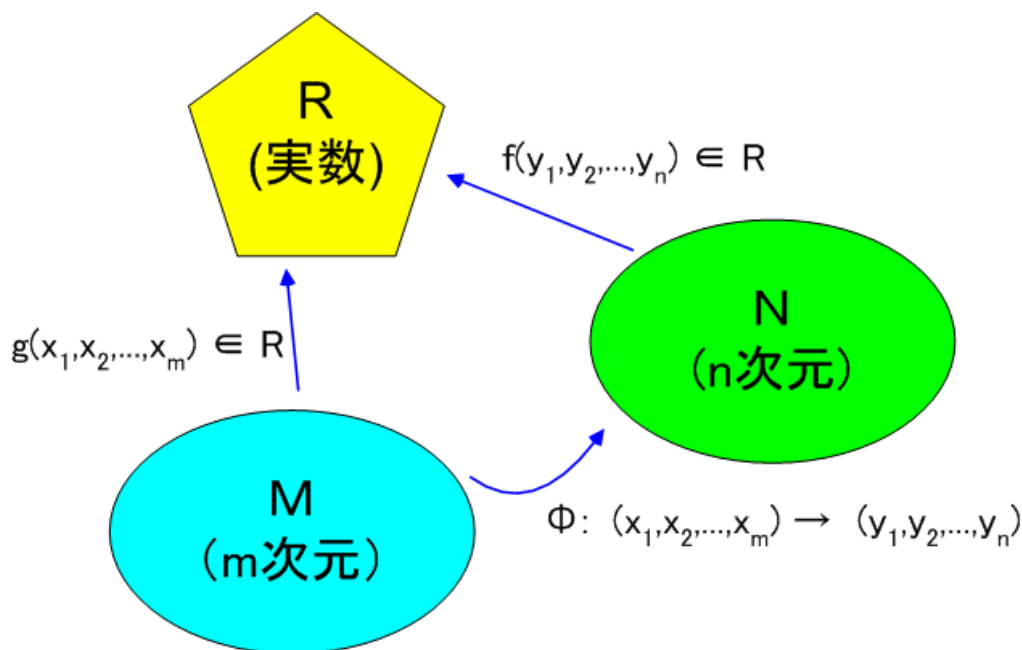
$$f(y_1, y_2, \dots, y_n) \in R$$

$$f: N \rightarrow R$$

さっきの図に、この写像を書き足してみます。



こう書いてみると、 M から R へ直接行く写像も欲しくなりますね♪ そんな写像だってあるはずですから、これを関数 g と書きましょう。 g は、 $g: M \rightarrow R$ という写像で、 $g(x_1, x_2, \dots, x_m) \in R$ となるはずです。



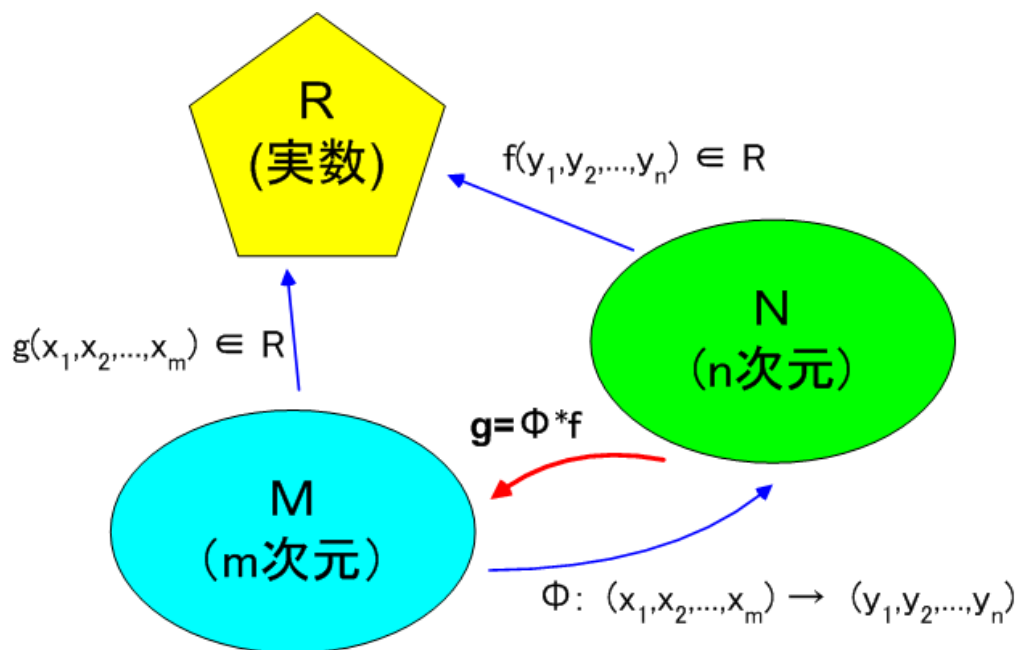
さて、基本的に $g(x_1, x_2, \dots, x_m)$ と $f(y_1, y_2, \dots, y_n)$ は写像 ϕ を介して、同じ値になるはずですね。(回り道しても、行き先は一緒ですね。) つまり、次式が言えます。

$$\begin{aligned} g(x_1, x_2, \dots, x_m) &= f(y_1, y_2, \dots, y_n) \\ &= f(\phi_1(x_i), \phi_2(x_i), \dots, \phi_n(x_i)) \quad (i = 1, 2, \dots, m) \end{aligned} \quad (7)$$

両辺を見比べて、関数 g は、 ϕ と f の合成写像であることが分かります。合成写像は、記号 $\circ$ を使って示すことにします。

$$g = f \circ \phi \quad (8)$$

この g を、 f の引き戻し (pull back) と呼びます。変な名前ですが、元来、 N 上で (変数 y_j のために) 定義されていた関数 f が、 x_i を y_j に移す写像 ϕ を使うことで、 M 上の関数 g に移されたということです。ここで本質的に働いているのは、 ϕ だけです。恐らく、『写像 $\phi: M \rightarrow N$ が、逆に N 上の関数 f を M 上に引っ張ってきた』というイメージで、引き戻しと命名されているのだと思います。引き戻しを決めるのは ϕ だけですから、 ϕ によって一意に決まる写像 ϕ^* を使って、 $g = \phi^* f$ のように書いても良いでしょう。つまり、 ϕ^* は N 上の関数を M 上の関数に移す写像です。



theorem

変数の写像 $\phi: M \rightarrow N$ によって, N 上の関数は M 上に引き戻されます.

⁵ 注意しておきますが, ϕ^ は ϕ の逆写像ではありません. なんだか紛らわしいですが, 別の写像です. 『 ϕ によって決まるよ』という意味を込めて, 似た顔つきをしています. 変数の写像と関数の写像は少し違うものだからです. 既にはっきりと書いたのですが, ϕ^* の正体は式 (8) で表わされる合成写像です.

*⁶ 式 (7) を見れば明らかなように, 写像 $\phi: M \rightarrow N$ をそのまま N 上の関数 f に代入することで, g を得ています. 変数と関数の変換の向きが一見逆 ($M \rightarrow N$ か $N \rightarrow M$ か) なので, これを気持ち悪く感じている人がいると思いますが, ではそんな人のために, 変数の写像 $\phi: M \rightarrow N$ によって, M 上の関数もやはり N 上に定義する (仮に『押しやり』 (push forward) と呼びましょう) ことが出来るか考えてみましょう. 式 (7) を見れば明かですが, 押しやりを定義するには, ϕ の逆写像 ϕ^{-1} が存在することが必要十分条件になります. (ϕ^{-1} さえ存在していれば, g に $\phi^{-1}(x_i)$ を代入して $f(y_j)$ を得ることが出来ますね.) ここまでの議論で, ϕ の逆写像については何も触れませんでした. 実は, 引き戻しを考える醍醐味は, 逆写像が無いような写像を考える際でも, 押しやりは無理だけれども, 引き戻しはいつでも考えられるという点にあるのです. 引き戻しは, 押しやりに比べてより一般的な写像に使える概念だと言えます. 古典的なテンソル解析では, 直交変換のように, 逆写像のある同次元の写像を考えることが多かったと思います. このような制約があれば, 変数も関数も変換は自由自在ですが, 私達はいま, より制約の少ない変換, 正確に言えば, 逆写像が存在しない (ヤコビアンに逆行列が存在しない) ような写像をも考えようとしています. そのような意図により, 押しやりではなくて引き戻しが重要になってくるという点を押さえておいて下さい.