高校生のための MuPAD

- MuPAD 3.0 入門 -

生越 茂樹（おごせ しげき）

2004 年 10 月 15 日

目 次

第1章　MuPAD Pro 3.0 とは

1.1　MuPAD Pro 3.0 の入手

「MuPAD Pro 3.0」は 2004 年 1 月に発表され, 現在 (2004 年 10 月) では SciFace 社 (http://www.mupad.com/) または TAN Server(http://www.mupad.org/muptan.html) へ行き, 無料でダウンロードできます．ただし, 無料で使用できるのは 1ヶ月間だけで, それ以上使用したい場合は,TAN Server へ行ってライセンスを購入しないと動かなくなります. ただ, ライセンスは非常に安く購入できます．(学生の場合は,65EUR　¥8800, 一般で非商用利用の場合は, 90EUR　¥12000, 2004 年 10 月現在) さらに, もし日本語によるサポートが欲しい方は, 日本の販売窓口のライトストーン社 (http://www.lightstone.co.jp/) から購入することもできます. 「MuPAD Pro 3.0」 は, 今のところ Windows 用 しかありません. したがって, ここでは 「Mupad Pro 3.0 for Windows」 について述べます.

1.2　Version3.0 と Version2.5 の違い (概観)

- Graphics library(グラフ描写) の改良. Graphics のプログラミングが, 一から書き直され, 空間図形の描写も劇的に改良されました. 描ける基本図形の数も 50 種類以上となり, 円, 四角形, , 直方体, 円柱, 円錐, 回転楕円体, 正多面体などもすぐ描く事ができます. Attribute(2.5 では option と呼んでいたもの) の数も 300 種類ぐらいに増えました. アニメーション (動く図形) も簡単に描けます．しかも AVI 形式で保存することもできるので，Windows Media Player などで見ることもできます.

- simplify() が改良され, Simplify() となりました.(以前の simplify() も使えます.) Simplify() は，simplify() だと簡単にできないような式も簡単にします．また内部で MuPAD は自動的に式を簡単にしている (internal simplication) のですが, それも改良されました. それに伴い LIST,SET,TABLE などの要素が並ぶ順序も多少変化しました.

- ２つの式が等しいかどうかを判定できる testeq() が導入されました. これは, 内部で Simplify() を使っているので, 今までより正確です.

- 常微分方程式のライブラリが改良されました.

このほか, 実にたくさんの改良が行われ, 新しいデータ形式もたくさん導入されました．詳しくは,「News and Changes」をご覧ください. これは，ヘルプ から容易に見れますが，なんと，30 ページもあります. これからも変化の大きさを, 見て取れます.

1.3　Light と Pro の違い

無料で登録できる MuPAD もあり，Light(Windows 版) といいます．ただし，現在のところ 「MuPAD Light 2.53」までしかなく,「MuPAD Light 3.0」は，まだ出ていません．(2004 年 10 月現在) しかし，出るのは時間の問題と思われます．そして,「Pro」 と 「Light」 は，出力がかなり異なるので，Light を使っ

ている方が, MuPAD Pro の本を読むと最初は戸惑われると思います．そこで, この本では,「MuPAD Pro 2.53」と「MuPAD Light 2.53」の違いから類推して「MuPAD Light 3.0 」の機能を**推定**して書いてあります．あくまで推定ですので，実際には違うこともあろうと思いますが，ご容赦ください．

さて，Light と Pro の違いは, なんといっても「Pro では, ノートブックが使える」 ということでしょう．ノートブックの特徴としては

- 数式が画像として出力されるので, きれいで見やすい.
- いろいろなコマンド (diff,int,plotfunc2d など 20 種類ぐらい) をマウスクリックで，入力できます.
- 入力補助が付いています. これは，コマンドを途中まで入力して Ctrl+Space を押すと, 残りのコマンドを補完してくれるという優れものです. また，F2 を押すと，小さなヘルプ が現われます．ヘルプにはそのスペルから始まる言葉が並ぶので，入力補助としても使えます.
- Light と違って, 一度入力したコマンドでも編集できます. また出力された数式は, そのままでは編集できませんが, マウスで範囲を選択し右クリックして,Text(通常文) に変更できます. Text に変更した後では, 自由に編集できます.
- Text をコマンドの間に書き入れられます. しかも,Text には日本語が使えます.
- Light だと長い計算になったとき途中でやめさせることはできず終了するしかありませんが, Pro の場合は メニューバーから「notebook」を選択して「Stop Calculation」を選ぶと, 計算を途中で停止させて, ノートブックに戻ることができます.
- 描いたグラフは, Light のように 別画面が立ち上がるのではなく, ノートブックの中に埋め込まれます. またマウスを使って，グラフをドラッグしてサイズを変えたり, 回転, 拡大させることもできます.
- Light では, メニューバーからは text 形式でしか出力できません. コマンド (`write`) を使えば セッション全体を保存できるはずですが, 実際には書いたファイルが読み込めないこともしばしば (恐怖!) あります. 一方, Pro では メニューバーから，ノートブック全体を保存することができ, これは確実性が非常に高いです. さらに,HTML,RTF,DOC 型式で保存することもできますから,「Microsoft word」や「Internet Explorer」などで読んだり, 加工することも簡単にできます. (しかもどの形式で保存しても, 外見はほとんど同じです．)
- 画像, サウンドファイル,「Word の DOC ファイル」や「Excel の表」などを，ノートブックに埋め込むこともできます. (ただ, 画像, サウンドファイルはアイコンで埋め込まれるだけで, 見たり聞いたりするには, そのアイコンをダブルクリックして開くことになります.)
- 逆に 「MicrosoftWord AddOn」(SciFace のサイトからダウンロードできます) を使えば,「MicroSoft Word」を，ノートブックのように使うこともできます. すなわち, 通常の文章入力の他に，数式を MuPAD の数式出力 (画像ファイルなので「Word」の出力よりずーときれい) で書くことができます．しかも,「Word」から MuPAD を呼び出して，数式を計算させたり，グラフを表示させ，それらを「Word」に埋め込むこともできます. (ただし,「Word」は「Word2000」以降のバージョンである必要があります．)

このようにとても便利な ノートブック ですが, このようなユーザーインターフェース上の違いを除けば Kernel(核) は全く同一です.

1.4 機能紹介

MuPAD を立ち上げ，新しい ノートブック（Light では，立ち上がった画面）に，次のコマンドを入力し，Enter を押すと，次のような結果が得られます．(ここは, 単なる機能紹介ですので, 詳しいことは, 後のページをご覧ください．)

1.4.1 微分, 積分

いろいろな関数の微分, 積分がそれぞれ，`diff`, `int` でできます.

- `diff(cos(x), x)` >> $-\sin(x)$
- `int(sin(x), x)` >> $-\cos(x)$

$(\cos x)' = -\sin x$, $\int \sin x dx = -\cos x$ を表しています．定積分もできます.

- `int(sin(x), x = 0..PI)` >> 2

π は，`PI` と入力します．したがって，$\pi\int_0^\pi \sin x dx = [-\cos x]_0^\pi = 2$ を表しています．もちろん, もっと複雑な微積分もできます.

- `diff(sin(x)^3, x)` >> $3 \cdot \cos(x) \cdot \sin(x)^2$
- `int(sin(x)^4, x = 0..PI)` >> $\frac{3 \cdot \pi}{8}$

ここで「^」は,「累乗」を表します．したがって，上の式は，それぞれ，$(\sin^3 x)' = 3\sin^2 x \cos x$, $\int_0^\pi \sin^4 x dx = \frac{3\pi}{8}$ を表しています.

1.4.2 方程式を解く

いろいろな方程式 (整式の不等式なども) を解くことができます．$f(x) = 0$ の解を求めるには，`solve(f, x)`, または，`solve(f = 0, x)`, `solve(f)` などとします.

- `solve(x^2 - 3 * x + 2, x)` >> $\{1, 2\}$

`x^2` は, x^2 を，`3 * x` は, $3x$ を表します．すなわち，$x^2 - 3x - 2 = 0$ の解が $\{1, 2\}$. ということです．連立方程式も解けます.

- `solve({x^2 - x * y = 6, y^2 - y * x = -2}, [x, y])` >> $\{[x = -3, y = -1], [x = 3, y = 1]\}$

$\begin{cases} x^2 - xy = 6 \\ y^2 - yx = -2 \end{cases}$ の解が，$(x, y) = (-3, -1), (3, 1)$ ということです.

1.4.3 因数分解

式の因数分解は factor でできます.

- eq1 := (a − b)^3 + (b − c)^3 + (c − a)^3 >> $(a-b)^3-(a-c)^3+(b-c)^3$

長くなるので，$(a-b)^3+(b-c)^3+(c-a)^3$ を, eq1 と名前をつけました.

- factor(eq1) >> $-3\cdot(b-c)\cdot(a-c)\cdot(a-b)$

MuPAD は，$(a-b)^3+(b-c)^3+(c-a)^3=3(a-b)(b-c)(c-a)$ と因数分解しました.

1.4.4 式の簡略化

いろいろな式を簡単にすることができます．これには, いろいろその用途 (展開, 因数分解, 通分, 無理数の有理化など）に特化したコマンドがありますが，とりあえず simplify, または Simplify で簡単にすることができます．(しかし，無理数の有理化は radsimp を使わないとうまく行かないことが多いです.)

- eq2 := (x^2)/(x^2 − 1) − (4 ∗ x − 3)/(x^2 − 1) >> $\frac{x^2}{x^2-1}-\frac{4\cdot x-3}{x^2-1}$

長くなるので，右辺の式に，eq2 という名前をつけました．これを simplify で簡単にしてみます.

- simplify(eq1) >> $\frac{x-3}{x+1}$

$\frac{x^2}{x^2-1}-\frac{4x-3}{x^2-1}=\frac{x-3}{x+1}$ ということです．今度は, 対数の式を簡単にして見ます．$\log_a x$ は,log(a, x) と入力します.

- eq3 := log(5, 7) ∗ log(7, 9) ∗ (log(3, 5) + log(9, 5)) >> $(\log_3(5)+\log_9(5))\cdot\log_7(9)\cdot\log_5(7)$

$\log_5 7\cdot\log_7 9\cdot(\log_3 5+\log_9 5)$ を eq3 と名前をつけました．これを，simplify で簡単にしてみます.

- simplify(eq3) >> $(\log_3(5)+\log_9(5))\cdot\log_7(9)\cdot\log_5(7)$

ぜんぜん簡単になりません．しかし，Simplify なら簡単にできます.

- Simplify(eq3) >> 3

この Simplify は，MuPAD 3.0 になって新しく導入されました．simplify より時間はかかりますが, より正確な結果がでます.

1.4.5 グラフの描写 (plot)

[$y = f(x), z = f(x,y)$ のグラフ]

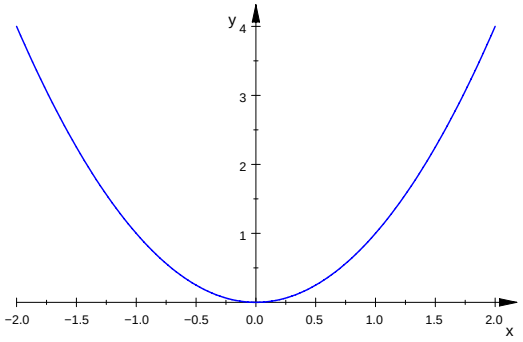
MuPAD は, 2 次元（平面）, 3 次元（空間）のグラフを簡単に描くことができます.

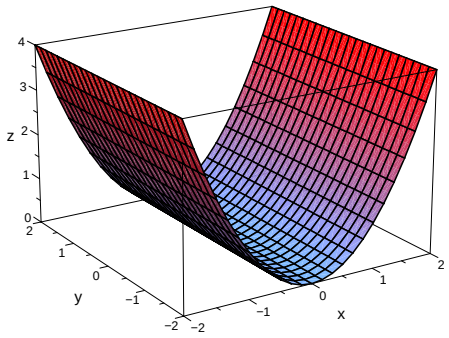
$$\begin{cases} \texttt{plotfunc2d(f, x = a..b)} & \cdots y = f(x)\ (a \leqq x \leqq b) \text{ のグラフ} \\ \texttt{plotfunc3d(f, x = a..b, y = c..d)} & \cdots z = f(x,y)\ (a \leqq x \leqq b, c \leqq y \leqq d) \text{ のグラフ} \end{cases}$$

$y = x^2\ (-2 \leqq x \leqq 2)$ のグラフと，$z = x^2\ (-2 \leqq x \leqq 2, -2 \leqq y \leqq 2)$ のグラフを描いてみます.

- `plotfunc2d(x^2, x = -2..2)`
- `plotfunc3d(x^2, x = -2..2, y = -2..2)`

これで, 次のようになります.





[パラメータ表示された曲線 (Curve2d,Curve3d)]

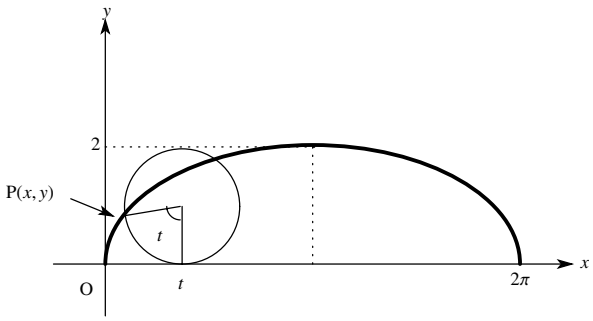
パラメータ表示された曲線も書くことができます．これには，plot library のコマンドを使うので，コマンド名は, `plot :: ···` となります．[注1)]

$$\begin{cases} \texttt{plot :: Curve2d([f, g], t = a..b)} & \cdots x = f(t), y = g(t)\ (a \leqq t \leqq b) \text{ のグラフ} \\ \texttt{plot :: Curve3d([f, g, h], t = a..b)} & \cdots x = f(t), y = g(t), z = h(t)\ (a \leqq t \leqq b) \text{ のグラフ} \end{cases}$$

注1) `plotfunc2d`, `plotfunc3d` も実は，内部で `plot :: Function2d`, `plot :: Function3d` という plot library のコマンドを使っています.

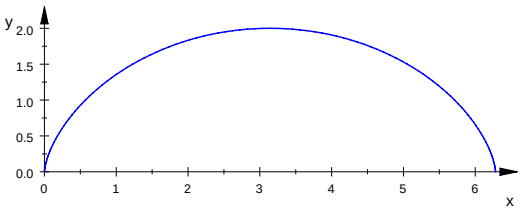
例:サイクロイド (*cycloid*) の静止画

サイクロイド:$x = t - \sin(t), y = 1 - \cos(t)$ $(0 \leqq t \leqq 2\pi)$ を描いてみます．これは半径 1 の円を x 軸の上に転がしたときに出来る図形です.



これを MuPAD でかいて見ます.

- `mycycloid := plot :: Curve2d([t - sin(t), 1 - cos(t)], t = 0..2 * PI) :`
- `plot(mycycloid, Scaling = Constrained)`



`plot :: Curve2d` のように，`plotfunc2d`, `plotfunc3d` 以外のコマンドは，`plot` と一緒に使います.
「`Scaling = Constrained`」 というのは,「x, y 軸方向の拡大率 (scaling) を等しくする」という意味です.
このように, コマンド中で（付けなくとも, 一応は大丈夫な) 部分を,option(または *Attribute*) といいます.
このような *Attribute* が，MuPAD 3.0 では, 300 種類ほどに増えました．しかも, そのほとんど全てを，対話的 (interactive) に, 変更することができるので, 名前を正確に覚えていなくとも大丈夫です.

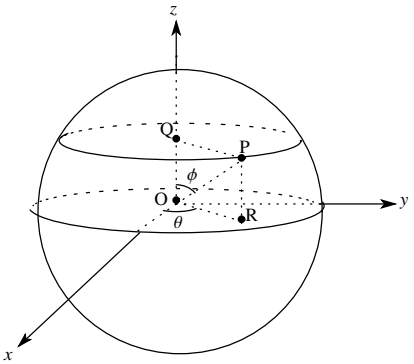
[パラメータ表示された曲面 (Surface)]

曲面も描くことができます.
曲面:$x = f(u, v), y = g(u, v)$ $(a \leqq u \leqq b, c \leqq v \leqq d)$ は次のように入力します.

`plot :: Surface([f(u, v), g(u, v)], u = a..b, v = c..d)`

例えば, 原点中心で半径 r の球面上の点を P とすると,

$$\begin{cases} \mathrm{OR} = \mathrm{PQ} = r\sin\phi \\ \mathrm{OQ} = r\cos\phi \end{cases}$$



よって，

$$\begin{cases} x = \mathrm{OR}\cos\theta = r\sin\phi\cos\theta \\ y = \mathrm{OR}\sin\theta = r\sin\phi\sin\theta \\ z = \mathrm{OQ} \qquad\quad = r\cos\phi \end{cases} \qquad \cdots(*)$$

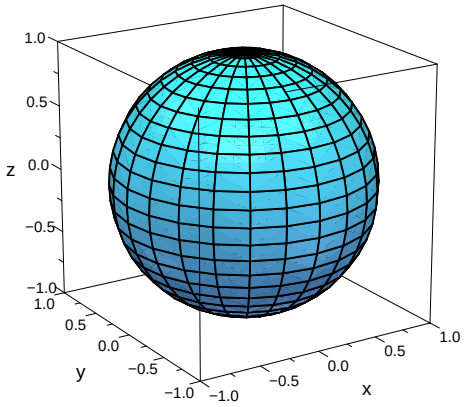
したがって原点中心，半径 1 の球のパラメータ表示は次のようになります。

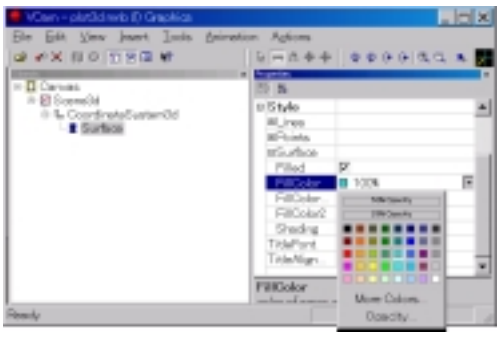
$$\begin{cases} x = \sin\phi\cos\theta \\ y = \sin\phi\sin\theta \quad (0 \leqq \phi \leqq \pi,\ 0 \leqq \theta < 2\pi) \\ z = \cos\phi \end{cases}$$

したがって，MuPAD では，次のように入力します。

- `mysphere := plot::Surface([sin(u)*cos(v), sin(u)*sin(v), cos(u)], u = 0..PI, v = 0..2*PI, FillColor = RGB::Aqua):`
- `plot(mysphere, Scaling = Constrained)`

ここで，`FillColor = RGB::Aqua` というのは，「塗りつぶしの色を水色 (aqua) にする」という意味です．このように `RGB::Black`, `RGB::Blue`, `RGB::Yellow` のようにして色を指定できます．このような色の名前を覚えていなくとも，右のような property inspector から，マウスでクリックして変更できます．





[幾何図形]

球のような幾何図形は，low-level primitive と呼ばれるコマンドを使って，すぐ作ることもできます．例えば，半径 r, 中心が $C(a,b,c)$ の球は，次のようにします．

`plot::Sphere(r, [a, b, c])`

また，底面の円の半径 r, 中心が $C(a,b,c)$, 頂点が (p_x, p_y, p_z) で表される円錐は，次のようになります．

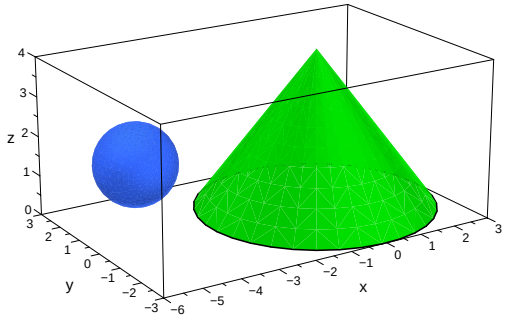
`plot::Cone(r, [a, b, c], [p_x, p_y, p_z])`

先ず, 球と円錐を作成します.

- `sphere := plot :: Sphere(1, [-5, 0, 2]) :` ··· 中心 $(-5, 0, 2)$, 半径 1 の球.
- `cone := plot :: Cone(3, [0, 0, 0], [0, 0, 4], FillColor = RGB :: Green) :`
  ··· 中心が $(0, 0, 0)$, 半径が 1 の円を底面とし，頂点が $(0, 0, 4)$ の円錐.

このように, 末尾に「:」を付けると, 計算はしますが，MuPAD は答えを表示しません．次に，これらを一緒に描きます.

- `plot(cone, sphere)`



この他, 円，平行四辺形，円柱, 直方体, 正多面体（正 6 面体, 正 8 面体,···, 正 20 面体) などもすぐ描くことができます.
さらに, 描いた静止画像ファイルは，eps,jpg,bmp,tif,png,gif,bif,pcx,tga,jvx,wmf 形式で保存 (export) できますから,「MicroSoft Word」などに貼り付けたり，対応ソフト (PaintShop ,paint など) で編集することもできます.

[アニメーション]

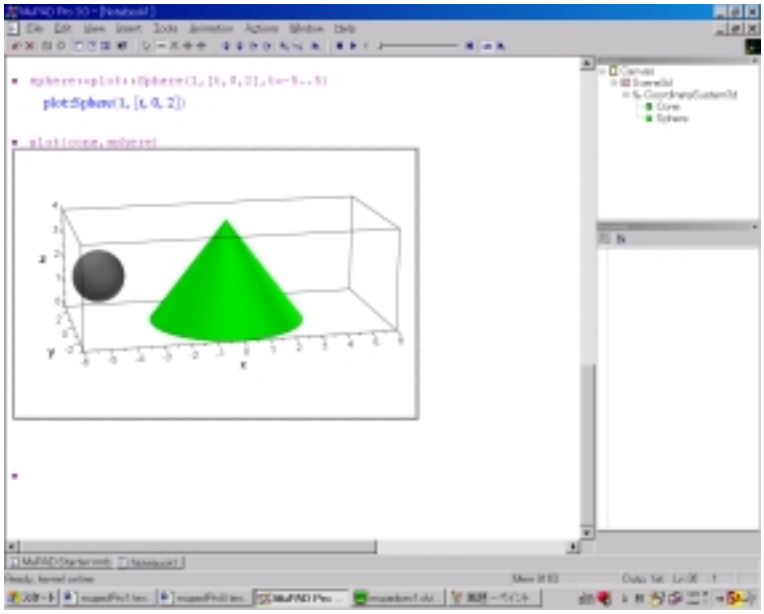
MuPAD 3.0 では, アニメーションを，実に簡単に描くことができます．特別のコマンドは，必要ありません．パラメーターを, 一個だけ増やし, その変域を一緒に指定するだけです．先の球と円錐の図形をアニメーションにして，球が円錐に衝突する様子を見てみましょう．球の中心を $(-5, 0, 2)$ から，$(t, 0, 2)$ にかえ，t の範囲を $-5 \leqq t \leqq 5$ で動かします.

- `sphere := plot :: Sphere(1, [t, 0, 2], t = -5..5) :`
  ··· 中心 $(t, 0, 2)$ $(-5 \leqq t \leqq 5)$, 半径 1 の動いている球.
- `cone := plot :: Cone(3, [0, 0, 0], [0, 0, 4], FillColor = RGB :: Green) :`
  ··· 静止している円錐 (先と同じ円錐)

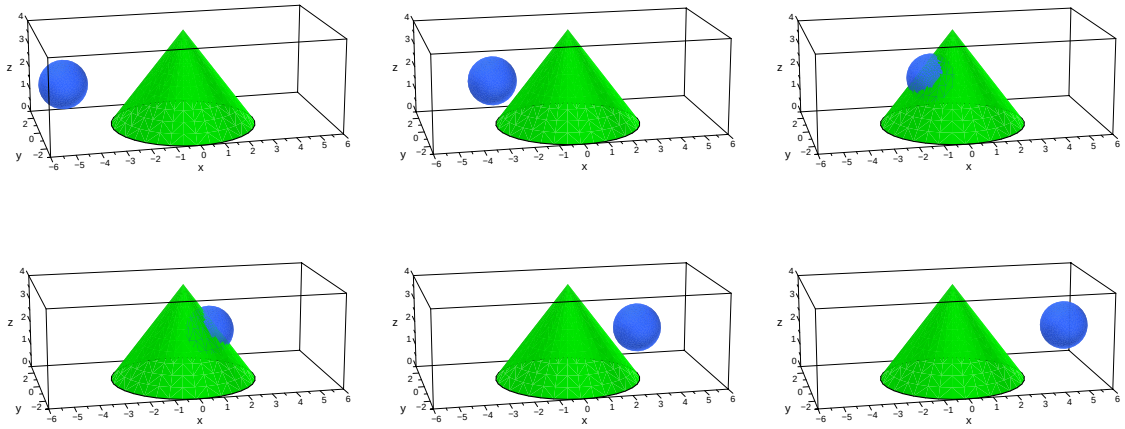
これらを, 先と同様に描きます.

- `plot(cone, sphere)`

一見するとさっきと変わりませんが，ノートブックでは, 画像をダブルクリックすると，メニューバーに下のようなプレイヤーが現われます.

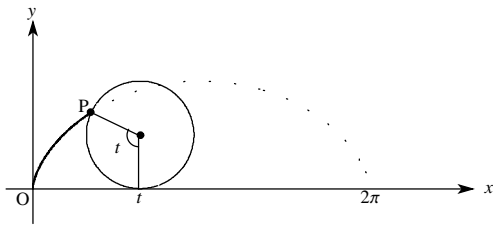


このプレイヤーの▶を,「ダブルクリック」すると再生が始まります.



ノートブックの通常の画面に戻るには, もう一度,「ダブルクリック」します. Light の場合は, ノートブックではなく, Vcam(別画面) のメニューバーに同じくプレイヤーが見えるはずです. また, Pro の場合でも, 画像を「右クリック」して,「Graphics オブジェクト」→ *Open* で, Vcam を立ち上げることができます.

今度は，もっと実際的なアニメーションを描いてみます．半径 1 の円と，その上の点 P，さらに，P の軌跡であるサイクロイドをアニメーションで見てみます．



半径 1 の円上の点 P が, はじめに原点にあったとし，その円が毎秒 1 の速度で，x 軸正方向に動いているとします．時刻 t において，円の中心も $(t, 1)$ になるので，上の図で,

$$\begin{cases} \text{円}: & (x-t)^2+(y-1)^2=1 \Longleftrightarrow \text{中心 } (t,1), \text{半径 1 の円} \\ \text{サイクロイドの一部（}O\text{ から }P\text{ までの曲線）}: & x=p-\sin p, y=1-\cos p\ (0 \leqq p \leqq t) \\ \text{点 } P: & x=t-\sin t, y=1-\cos t \end{cases}$$

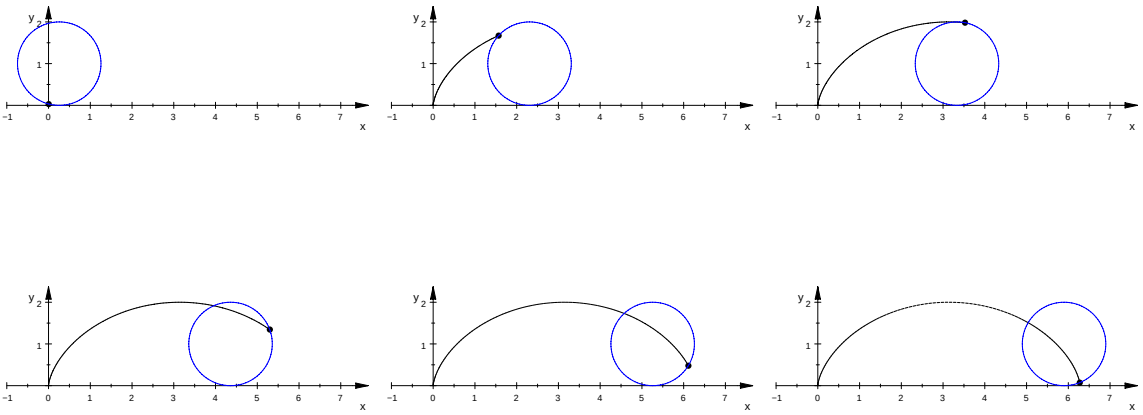
となります．したがって, 次のようにすれば, アニメーションが作れます．(詳しい説明は，本文を見てください.)

- `mycircle := plot :: Circle2d(1, [t, 1], t = 0..2 * PI) :`
- `mycycloid := plot :: Curve2d([p - sin(p), 1 - cos(p)], p = 0..t, t = 0..2 * PI,`
  `Color = RGB :: Black) :`
- `mypoint := plot :: Point2d([t - sin(t), 1 - cos(t)], t = 0..2 * PI,`
  `Color = RGB :: Black, PointSize = 2) :`

これを，`plot` を使って一度に描写 (plot) します．

- `plot(mycircle,mycycloid,mypoint)`

先ほどと同じようにして, プレイヤーを表示し, ▶ をクリックすると, 下のアニメーションが見えます．



アニメーションは，avi ファイルで保存 (export) できますから,「Windows Media Player」などでも見ることができます．このように，Graphics の章は, 根本的に書き直されました．

1.5 行列, ベクトル

1.5.1 行列, ベクトルの定義

MuPAD では, 行列やベクトルの計算をすることもできます．最も簡単に, 行列やベクトルを定義するには，matrix を使います.

- u := matrix([[1, 2]]) >> $(1,2)$ $\cdots \vec{u}$ の定義
- v := matrix([[2], [3]]) >> $\begin{pmatrix}2\\3\end{pmatrix}$ $\cdots \vec{v}$ の定義
- A := matrix([[1, 2], [3, 4]) >> $\begin{pmatrix}1 & 2\\3 & 4\end{pmatrix}$ $\cdots$ A の定義
- B := matrix([[1, 2], [−1, −2]) >> $\begin{pmatrix}1 & 2\\-1 & -2\end{pmatrix}$ $\cdots$ B の定義

1.5.2 行列, ベクトルの四則計算

行列の加減乗除は実数と同じ様に計算できます．例として 2 × 2 行列でやってみますが，どんな行列でも同様です.

- 2 ∗ A + 3 ∗ B >> $\begin{pmatrix}5 & 10\\3 & 2\end{pmatrix}$ $\cdots$ 2A+3B
- A ∗ B >> $\begin{pmatrix}-1 & -2\\-1 & -2\end{pmatrix}$ $\cdots$ AB
- B ∗ A >> $\begin{pmatrix}7 & 10\\-7 & -10\end{pmatrix}$ $\cdots$ BA
- A^(−1) >> $\begin{pmatrix}-2 & 1\\\frac{3}{2} & -\frac{1}{2}\end{pmatrix}$ $\cdots \mathrm{A}^{-1}$
- 1/A >> $\begin{pmatrix}-2 & 1\\\frac{3}{2} & -\frac{1}{2}\end{pmatrix}$ $\cdots \mathrm{A}^{-1}$
- A^2 >> $\begin{pmatrix}7 & 10\\15 & 22\end{pmatrix}$ $\cdots \mathrm{A}^2$
- A^2 ∗ B >> $\begin{pmatrix}-3 & -6\\-7 & -14\end{pmatrix}$ $\cdots \mathrm{A}^2 B$

1.5.3　linalg (線形代数のライブラリ)

行列の対角化 (一般には Jordan 標準形への変形) も 簡単にできます．高度な行列やベクトルの演算は, linalg (library for linear algebra) を使います．ごく一例を挙げます.

- `linalg :: rank(A)` ⋯ A の階数 (*rank*)
- `linalg :: charpoly(A, x)` ⋯ A の (変数を x とする) 固有方程式
- `linalg :: eigenvalues(A)` ⋯ A の固有値
- `linalg :: eigenvectors(A)` ⋯ A の固有ベクトル
- `linalg :: jordanForm(A)` ⋯ A の *jordan* 標準形 (対角化)

例えば，2×2 行列でやってみます．(一般の行列でも同様です.)

- `A := matrix([[2, 1], [3, 4]])` >> $\begin{pmatrix} 2 & 1 \\ 3 & 4 \end{pmatrix}$
- `linalg :: rank(A)` >> 2
- `linalg :: charpoly(A, x)` >> $x^2 - 6\cdot x + 5$
- `linalg :: eigenvalues(A)` >> $\{1, 5\}$
- `linalg :: eigenvectors(A)` >> $\left[\left[1, 1, \begin{pmatrix} -1 \\ 1 \end{pmatrix}\right], \left[5, 1, \begin{pmatrix} \frac{1}{3} \\ 1 \end{pmatrix}\right]\right]$
- `linalg :: jordanForm(A)` >> $\begin{pmatrix} 1 & 0 \\ 0 & 5 \end{pmatrix}$

固有値, 固有ベクトルなどの用語については, 本文で詳しく説明してあります．この他, ベクトルの内積, 外積，発散 (divergence)，回転 (curl) などのベクトル解析のコマンドもあり, 全部で 80 種類ほどのコマンドが，linalg には，入っています.

第2章　数字(number)の操作

2.1　最初の一歩

2.1.1　MuPAD Pro の場合

さて, それでは早速 notebook を使ってみましょう. まず MuPAD Pro 3.0 を立ち上げると「An Introduction to MuPAD」という画面が立ち上がります. (これも notebook で, 恐ろしいことに自分で加工することもできるので書き換えてしまわないように注意が必要です.) [注1)] メニューバーから「File」を選択し「New Notebook」をクリックすると新しい notebook が開きます.'•' が点滅している右横に数式を入力します. これが *input region* です.[注2)] '•' の横に

$$\bullet\ 2/3 + 1/2$$

と入力して Enter key を押します. すると結果が

$$\frac{7}{6}$$

と出力されます.[注3)] これを *output region* といいます. つぎに MenuBar から「Insert」⟶「Text Below」を選択すると'|' だけが'•' なしで現われます. これが *text region* で, いろいろな文章を書くことができます. 日本語もここでは使えます. すなわち notebook の領域は,

$$\begin{cases} \textit{input region} & \cdots \text{コマンドを入力する部分} \\ \textit{output region} & \cdots \text{MuPAD が計算結果を出力する部分} \\ \textit{text region} & \cdots \text{コメントを書き込む部分} \end{cases}$$

の 3 つの部分に分けられます. このうち, *input region* と *text region* は自由に編集できますが *output region* では, 一度出力されたものは変更できません. [注4)] しかし, メニューバー (MenuBar) の「NoteBook」⟶「Change to Text」を選択すると *text region* に変えることができ, その後は自由に変更できます. また逆に「Change to Input」を選んで, *text region* を *input region* に変更することもできます. また notebook では任意のところに *text region* や *input region* を挿入することもできます. MenuBar の「Insert」をクリックしてください. *text region* には日本語も使えますが *input region* では, コメント以外は日本語は使えず, **半角の**アルファベットや数字しか入力できません. このとき間に (半角の) 空白を入れても大丈夫ですが, 途中で改行したいときは shift を押しながら, Enter を押してください. また MuPAD では大文字と小文字を区別します. 例えば, simplify と Simplify は別のコマンドです.

注1) 最初に現われる画面はメニューバーの「View」⟶「「Options」⟶「MuPAD」から変更できます.

注2) 最初の設定だと, 赤色の丸に, 赤色の文字になっているはずです. 色を変えたいときは 窓の上の Menu bar から, View ⟶ Options ⟶ 「Font」 ⟶ 「InputRegion」とたどっていって, 色を変えることが出来ます. さらに *output region*, *text region* の色やフォントを変えることもできます.

注3) これは画像として出力されています.

注4) MenuBar⟶「Notebook」⟶「Delete output」で *output region* を一度に消去することはできます.

2.1.2　MuPAD Light の場合

立ち上げるといきなり notebook と同じような画面が現われます.'•' が点滅している右横に

• 2/3 + 1/2

と入力して Enter を押します. すると結果が

7/6

と出力されます.Light では Windows の標準のフォントのみを使って表示されます. しかも残念ながら Light では 1 度 Enter を押してしまった入力は変更できません. どうしても変更したいときは，*cut&paste* で新たに次の行に入力するしかありません. また *text region* もなく,text を入力することもできません. しかし前章で述べたように基本的な機能は全く同じです.

2.2　加減乗除

$a+b$	a + b
$a-b$	a − b
$a\times b$	a ∗ b
$\frac{a}{b}$	a/b
a^n	a^n

では，まず簡単な計算からやってみましょう.
+, −, ×, ÷, 累乗はそれぞれ +, −, ∗, /, ^ を使います. [注5)] 式を入力したら，Enter を押してください. すると計算した結果が, その下に表示されます. また式の最後に「;」と打ってから Enter を押しても同じです. 最後を 「:」で終わらせ Enter を押すと，MuPAD は計算はしますが表示はしません. また, 複数の式を同時に実行することもできます. その場合は, 各々の式の間に「;」または「:」を打って分割してください.
以下，計算したい数式を一番左に "?" をつけて表し，入力する式を • の横に書きます. Enter を押すと >> の右横の式が表示されるはずですが，MuPAD Light は Windows の標準フォントしか使わないので実際に画面に表示される式は異なって見えるはずです.
【例】

計算したい数式	入力	出力
$3+\frac{4}{3}$は?	• 3 + 4/3	>> $\frac{13}{3}$
$3-(2\times 3)$?	• 3 − (2 ∗ 3)	>> −3
2^3は?	• 2^3	>> 8
2^{50}は?	• 2^50	>> 1125899906842624
$\left(\frac{2}{3}\right)^{-2}$ は?	• (2/3)^(−2)	>> $\frac{9}{4}$

[注5)] +, −, ∗, /, ^ の代わりにそれぞれ _plus( ),_subtract( ), _mult( ),_devide( ),_power( ) を使うこともできますが，こちらは主にプログラミングの際に使用します.

(–2) のように括弧がいる事を忘れないでください.

$(2^3)^2$は?	• (2^3)^2	>> 64
2^{3^2}は?	• 2^(3^2)	>> 512
$2^4 \times 3$ は?	• 2^4 * 3	>> 48
$3 \div 2^3$は?	• 3/2^3	>> $\frac{3}{8}$

このように，累乗は積や商より優先します. MuPAD では，文字式の計算も出来ます. このとき，**乗法記号 (*) は省略できない**ことに注意してください.

$3x + 2 - (2x - 3)$?	• 3 * x + 2 - (2 * x - 3)	>> $x + 5$
$\frac{x+2}{2} - \frac{2x}{3}$は?	• (x + 2)/2 - (2 * x)/3 は?	>> $\frac{-x+6}{6}$

このように `simplify` を働かせなくとも，式は簡単になっています. [注6)]

2.3 小数表示

小数表示	`float()`
表示桁数をnへ変更	`DIGITS := n`

いま見たように MuPAD は電卓と違い $3 + \frac{4}{3} = 4.3333333$ としません. ちゃんと正確に答えを出してくれます. でも「MuPAD を電卓のように使いたい」というときは `float` というコマンドを使います. 例えば次のようにします.

$\frac{4}{3}$の小数表示は?	• `float(4/3)`	>> 1.333333333
$\frac{40}{3}$の小数表示は?	• `float(40/3)`	>> 13.33333333
$\frac{400}{3}$の小数表示は?	• `float(400/3)`	>> 133.3333333

全部の数字の数は，10 個で同じですが，小数点の位置が移動しているのがわかりますね？実はこのような小数点表示は (有効数字が 10 桁の)「浮動小数」と呼ばれます. [注7)] MuPAD では最初の設定 (Default) では有効数字桁数が 10 に設定されています. これを 例えば 20 に変えるには DIGITS:=20 とします.[注8)] このとき，'=' ではなく':=' であることに注意してください. ':=' は**代入や定義**をするときに使います. したがって DIGITS:=20; は「DIGITS という変数に 20 を代入せよ.」という意味です.[注9)]

注6) これは，*internal simplication* と呼ばれ，MuPAD の Kernel（核）で実行されています．これによって, メモリーを節約しています.

注7) 私は, これが float() の名前の由来ではないか？と思っています.

注8) digit は数字という意味です.digital の語源です.

注9) これに対し単なる'=' はプログラミングするときに，**判断**のとき使ったり, 方程式や TABLE(MuPAD の Data type の一つ) で,**(左辺)=(右辺)** とする時 に使います. 例えば,「•if x = 20 then y := 0」 ,「•solve(x^2-1=0)」,「•table(year=2004,month=10)」などの時に使います.

有効桁数を 20 桁にするには?	• `DIGITS := 20 :`	
$\frac{4}{3}$の小数表示は?	• `float(4/3)`	>> 1.3333333333333333333

2.4　履歴 (history table)

直前の履歴	%
n 個前の履歴	%n
直前の履歴	`last(1)`
n 個前の履歴	`last(n)`
$x_1, x_2, \cdots$ の定義のクリア	`delete` $\mathtt{x}_1, \mathtt{x}_2, \cdots$
全ての履歴のクリア	`reset( )`

MuPAD では直前の結果の式を, '%' または, `last` を使って表します. 例えば

有効桁数を 20 桁に設定	• `DIGITS := 20`	>> 20
$1 + \frac{4}{3}$を計算すると?	• `1 + 4/3`	>> $\frac{7}{3}$
$1 + \frac{4}{3}$の小数表示は?	• `float(%)`	>> 2.3333333333333333333

% は直前の output の式:$\frac{7}{3}$ を表しています. また, 以前に DIGITS:=20 と入力したのはまだ有効です. 1 つ直前でなく,2 つ以上前の式を表すには %2,%3 などを使用し, それぞれ 2 つ前,3 つ前に計算した式を表します. このとき「:」を使って画面に出力させていなくとも同じです. また% の代わりに `last()` を使うこともできます. 例えば,

2 個前に計算した式は?	• `last(2)`	>> $\frac{7}{3}$
有効桁数を 30 桁に変更	• `DIGITS := 30 :`	>>
$\frac{7}{3}$の小数表示は?	• `float(%2)`	>> 2.33333333333333333333333333333

•DIGITS:=30:と「:」で終わっているので, MuPAD の画面上では「30」は出力されません. しかし MuPAD の記憶庫 (history table) では「30」は記憶されているので •`float(%2)` と打つと, 2 つ前に計算された $\frac{7}{3}$ の小数表示を計算します. このように, % は画面上での順序とは関係なく,MuPAD の記憶庫における順序を表すことに注意してください. これは特に MuPAD Pro で *input region* を再編集したときに重要です. また変数の定義[注10)] は「DIGITS:=20 」などだけでなく,MuPAD ですでに定義されている変数を除けば, 自分で自由に設定できます. (ただし変数名は数字から始まってはいけない. 使ってよい記号は「_」だけ などの制限があります.) 設定した変数に関しても一度設定した定義は, 何もしなければ, ずーと有効です. これらを元の状態に戻すには delete を使います. (なお, `delete` には, 括弧は不要です. また, いくつかの変数をまとめて, `delete` することもできます.

有効桁数を初期値に戻す	• `delete DIGITS`	>>
$\frac{7}{3}$の小数表示は?	• `float(7/3)`	>> 2.333333333

このように元に戻ります．また, 自分で値を設定した変数の値を取り消したいときも delete を使います. さらに，全ての履歴を一度に消去するには, reset( ) を使います.

$a = \frac{4}{3}$と代入	• a := 4/3	>> $\frac{4}{3}$
$b = \frac{1}{2}$と代入	• b := 1/2	>> $\frac{1}{2}$
$a + b$ を計算	• a + b	>> $\frac{11}{6}$
b を消去	• delete b	>>
$a + b$ を計算	• a + b	>> $b + \frac{4}{3}$
全ての履歴を消去	• reset()	>>
$a + b$ を計算	• a + b	>> $a + b$

2.5 数学定数

MuPAD では特に大切な定数の値は，次のように定まっています. [注11)]

π(円周率)	PI
e(自然対数の底)	E
i(虚数単位)	I
∞(無限大)	infinity
真	TRUE
偽	FALSE
真偽不明	UNKNOWN

float() は数学定数にも使えます. 以下，DIGITS:=10; を入力したとします.

πの小数点表示は?	• float(PI)	>> 3.141592654
e の小数点表示は?	• float(E)	>> 2.718281829

このほかにも数学定数はあります. 詳しくは HELP⟶ QuickReference を見てください. [注12)]

2.6 平方根 ($\sqrt{}$) の計算

平方根	sqrt( )
平方根の単純化 (特に分母の有理化など)	radsimp( )
単純化	simplify( )
(複雑な式の) 単純化	Simplify( )
平方根の単純化 (radsimp と同じ)	simplify(,sqrt)

注11) 大文字と小文字の区別に注意してください.

注12) ほかに主なものとしては, Z␣,Q␣,R␣,C␣(それぞれ整数, 有理数, 実数, 複素数の集合．EULER(オイラー数), CATALAN(カタラン数) などがある.

注13) 平方根は,sqrt( ) を使います.

$\sqrt{5}\,\sqrt{5}$は?	• `sqrt(5)*sqrt(5);`	>> 5
$2\sqrt{18}+\sqrt{50}$は?	• `2*sqrt(18)+sqrt(50);`	>> $11\,2^{\frac{1}{2}}$

上の出力は MuPAD Light の場合です．MuPAD Pro では $11\cdot\sqrt{2}$ と表示されますが, このように，MuPAD Light では，結果が累乗根の形で出ます. ちなみに $a>0, n, m$ が自然数のとき,

$$a^{\frac{1}{n}} = \sqrt[n]{a}, a^{\frac{m}{n}} = (\sqrt[n]{a})^m \text{ (定義)}$$

ですから $a^{\frac{1}{2}} = \sqrt{a}$ となります. したがって確かに $11\,2^{\frac{1}{2}} = 11\sqrt{2}$ となります.(11 と $2^{\frac{1}{2}}$ の間の空白に注意してください. 以下, この章では MuPAD Pro の方の出力だけを表示します．) 次は積です.

$\sqrt{8}\,\sqrt{14}$は?	• `sqrt(8)*sqrt(14)`	>> $2\cdot\sqrt{2}\cdot\sqrt{14}$

`simplify` を使って簡単にしてみます.

上の式を簡単にすると?	• `simplify(%)`	>> $4\sqrt{7}$

うまく簡単になりました. さて, 今度は分母の有理化をやってみます.

$\frac{1}{2+\sqrt{3}}$は?	• `1/(2+sqrt(3))`	>> $\frac{1}{\sqrt{3}+2}$

まず, `simplify` を使ってみます.

上の式を簡単にすると?	• `simplify(%)`	>> $\frac{1}{\sqrt{3}+2}$

$\frac{1}{2+\sqrt{3}} = \frac{2-\sqrt{3}}{(2+\sqrt{3})(2-\sqrt{3})} = 2-\sqrt{3}$ となっても良いはずですが，ぜんぜん変化しません．今度は，`radsimp` を使ってみます.

2 つ前の式を簡単にすると?	• `radsimp(%2)`	>> $2-\sqrt{3}$

今度はうまく行きました. 分母の有理化には `radsimp` の方が良いみたいです．なお `radsimp` の代わりに,`simplify( ,sqrt)` を使っても，全く同じです.

3 つ前の式を簡単にすると?	• `simplify(%3,sqrt)`	>> $2-\sqrt{3}$

ここで，2 番目の引数の *sqrt* というのは *target* と呼ばれ, この他にも `simplify` では *sin,cos,exp,ln* などが使え，それぞれ, その対応する種類の関数のみを簡単にします．今度は, 二重根号をはずしてみましょう.

$\sqrt{8+2\sqrt{15}}$は?	• `sqrt(8+2*sqrt(15))`	>> $\sqrt{\sqrt{15}+4}\cdot\sqrt{2}$
簡単にすると?	• `radsimp(%)`	>> $\sqrt{3}+\sqrt{5}$

注13) sqrt は "*square root*" の略で二重根号の意味です．（ちなみに square は 2 乗を意味します.）simplify( ) は文字どうり「( ) 内を simple にせよ」という意味で平方根のみならず様々な変形に使えます.radsimp( ) は "simplify radicals" の略で，radical はここでは'根' という意味です. ' 過激な' という意味ではありません.

$\sqrt{8+2\sqrt{15}} = \sqrt{(\sqrt{3}+\sqrt{5})^2} = \sqrt{3}+\sqrt{5}$ なので合っています. 今度は simplify でやってみましょう.

2 つ前の式を簡単にすると? • `simplify(%2)` >> $\sqrt{3}+\sqrt{5}$

今度はどちらでもうまく行きました. 次の例はどうでしょう.(今度は, 変数を使っています)

$eq = \dfrac{1}{1+\sqrt{2}+\sqrt{3}}$ と定義 • `eq := 1/(1 + sqrt(2) + sqrt(3))` >> $\dfrac{1}{\sqrt{2}+\sqrt{3}+1}$

eq を簡単にすると? • `simplify(eq)` >> $\dfrac{1}{\sqrt{2}+\sqrt{3}+1}$

やはり simplify では簡単になりません. つぎは radsimp でやってみます.

eq を簡単にすると? • `radsimp(eq)` >> $\dfrac{\sqrt{2}}{4} - \dfrac{\sqrt{3}\cdot\sqrt{2}}{4} + \dfrac{1}{2}$

やはり, 分母の有理化は,`radsimp` を使った方が良さそうです. しかし, ここではさらに, $\sqrt{2}\cdot\sqrt{3} \longrightarrow \sqrt{6}$ となって欲しいですね. このような時は `combine` を使います.

2 つの指数をまとめると? • `combine(%);` >> $\dfrac{\sqrt{2}}{4} - \dfrac{\sqrt{6}}{4} + \dfrac{1}{2}$

combine は $2^{\frac{1}{2}}\cdot 3^{\frac{1}{2}} = (2\cdot 3)^{\frac{1}{2}}$ のように指数をまとめたり, より一般には関数の数を減らすときに使います. (左の例では $2^{\frac{1}{2}}$ と $3^{\frac{1}{2}}$ の二つの関数の代わりに $6^{\frac{1}{2}}$ の一つに減っています)

このような例外もありますが, 無理数を簡単にするには, ほとんどの場合, `radsimp` を使えばよいでしょう.
他に `simplify` を発展させた `Simplify` も使えます. しかし実は, `Simplify` でも, 分母の有理化 はやってくれません. [注14)]
また `Simplify` は MuPAD 3.0 になって初めて導入されましたが, `simplify` の方も大きく改善されています. MuPAD 2.5 で無理数の計算をやらせると, なかなか思った結果が得られず, とても実用的とはいえませんでした.
[注15)]

2.7 2 つの式の比較 (検算の方法)

2 つの式を比較するには, 次のようなコマンドがあります. simplify を除き, 出力は TRUE(真), FALSE(偽), UNKNOWN(不明) のいずれかです.

$eq1 - eq2 = 0$?	`simplify(eq1 - eq2)`
$eq1 = eq2$?	`bool(eq1 = eq2)`
$eq1 = 0$?	`iszero(eq1)`
$eq1 = eq2$?	`testeq(eq1,eq2)`

[注14)] `Simplify` には,*All* option があり, `Simplify(eq, All)` とすれば eq を簡単にした式**全て**を出力してくれるのですが, この中にも入っていません. よってこれは MuPAD のポリシーなのでしょう.
[注15)] 例えば •simplify(sqrt(8)*sqrt(14))>> $2\sqrt{28}$
となったり, • radsimp(sqrt(8+2*sqrt(15)) >> $\sqrt{2}(\sqrt{\sqrt{15}+\sqrt{4}})$,
• simplify(sqrt(8+2*sqrt(15)) >> $\sqrt{2}(\frac{\sqrt{6}}{2}+\frac{\sqrt{10}}{2})$ となったりしました.

前節で見たように，MuPAD では (他の数学ソフトも同じですが) 必ずしも思った通りの式が，すぐ得られるとは限りません. これは一言で simplify(簡略化) といっても，個人の好みやいろいろなケースがあり「$\frac{\sqrt{2}}{4} - \frac{\sqrt{3}\cdot\sqrt{2}}{4} + \frac{1}{2}$ の方が $\frac{1}{1+\sqrt{2}+\sqrt{3}}$ よりも簡単」 とは一概に言えないからです．しかし自分で計算した式の結果を検算するのならば,「(左辺)⟶(右辺)」を導く代わりに「(左辺)–(右辺)⟶0 」を示せば大丈夫です．例えば 前節の $eq = \frac{1}{1+\sqrt{2}+\sqrt{3}}$ を自分で計算して $ans = \frac{\sqrt{2}}{4} - \frac{\sqrt{6}}{4} + \frac{1}{2}$ になってこれが正しいかどうか知りたいとします．このとき次のようにして検算できます.

$eq = \frac{1}{1+\sqrt{2}+\sqrt{3}}$ と定義	• eq := 1/(1 + sqrt(2) + sqrt(3))	>> $\frac{1}{\sqrt{2}+\sqrt{3}+1}$
$ans = \frac{\sqrt{2}}{4} - \frac{\sqrt{6}}{4} + \frac{1}{2}$ と定義	• ans := sqrt(2)/4 − sqrt(6)/4 + 1/2	>> $\frac{\sqrt{2}}{4} - \frac{\sqrt{6}}{4} + \frac{1}{2}$
$eq - ans = 0$?	• simplify(eq − ans)	>> 0

うまく行きました．simplify(eq) ではうんともすんとも変わりませんが, simplify(eq − ans) は簡単になります．また 2 つの式 ($eq1$ と $eq2$) が等しいかどうかを調べるには bool(eq1 = eq2), iszero(eq1 − eq2) や testeq(eq1, eq2) が使えます．testeq は,MuPAD 3.0 で初めて導入されたコマンドで，内部で Simplify を利用しています．そのせいか精度も 他のコマンドより良いみたいで, bool や iszero だと間違うときも testeq は正解を出してきます.

$eq = ans$?	• bool(eq = ans)	>> $FALSE$
$eq - ans = 0$?	• iszero(eq − ans)	>> $FALSE$
$eq = ans$?	• testeq(eq, ans)	>> $TRUE$

これを見ると iszero や bool は使いたくなくなりますね．[注16)]

2.8　複素数の計算

虚数単位	I
$a + bi$(a, b 実数) の形に直す	rectform( )
実数部分	Re( )
虚数部分	Im( )

注17)

注16) iszero, bool は, スピードが速いので プログラミングの中で使います．$eq1, eq2$ にどのような式が入るか分かっていれば, iszero や bool でも間に合います．さらに bool よりは iszero の方が精度が良いので，プログラミング中で 2 つの式を比較するときは if bool(eq1 = eq2) ⋯ とやるよりも, if iszero(eq1 − eq2) ⋯ とした方が良いと, manual に書いてあります. なお，実際のプログラミングでは bool は省略して, if eq1 = eq2 ⋯ とすると,MuPAD の方で自動的に bool を補います.

注17) rectform は, rectangular form の略で , rectangular というのは' 長方形の' とか' 四角張った' という感じの意味です.

2.8.1 複素数の計算

複素数の計算は，虚数単位 i を I と打つだけです. このとき必ず大文字の I を使ってください.

i^2は?	• I^2	>> -1
$(2+3i)^2$は?	• (2 + 3 * I)^2	>> $-5+12I$
$\frac{-i}{2+i}$は?	• (−I)/(2 + I)	>> $-1/5-2/5I$

このように $a+bi(a,b$ は実数) の形に直してくれます. ただし，無理数や文字を含んでいる場合は，簡単には直してくれません. そのような時，$a+bi(a,b$ は実数) の形に直すには Re,Im,rectform を使います.

$x=\frac{1}{\sqrt{3}+i}$と定義	• x := 1/(sqrt(3) + I)	>> $\frac{1}{\sqrt{3}+i}$
x を簡単にすると?	• simplify(x)	>> $\frac{1}{\sqrt{3}+i}$
x を変換すると?	• rectform(x)	>> $\frac{\sqrt{3}}{4}-\frac{i}{4}$
x の実数部分は?	• Re(x)	>> $\frac{\sqrt{3}}{4}$
x の虚数部分は?	• Im(x)	>> $-\frac{1}{4}$

無理数の有理化と同じく, simplify では簡単になりません．実は, radsimp でも簡単になりますが, これが一番遅いです．一番早いのは Re, Im でその次は rectform です．次は文字の場合をやって見ます.

$a>0$ と設定	• assume(a > 0)	>> $(0,\infty)$
$\frac{1}{a+i}$を簡単にすると?	• rectform(1/(a + I))	>> $\frac{a}{a^2+1}-\frac{i}{a^2+1}$

ここで assume(a > 0) というのは,「変数 a を正の数に設定する」という意味です．もし a が純虚数だったりすると答えが違ってきますから, これは必要です.

2.8.2 $\sqrt{-a}\ (a>0)$ の表し方.

$\sqrt{-3}$ は, sqrt(−3) のように入力します. MuPAD Light では, $\sqrt{-3}$ は， $\sqrt{3}i$ でなく，$I\,3^{\frac{1}{2}}$ のように出力されます.

[*Light*]	$\sqrt{-3}$は?	• sqrt(−3)	>> $3^{\frac{1}{2}}I$
[*Pro*]	$\sqrt{-3}$は?	• sqrt(−3)	>> $i\cdot\sqrt{3}$
[*Light*&*Pro*]	$\sqrt{-4}$は	• sqrt(−4)	>> $2I$

2.9　その他の基本演算 (一部のみ)

x の切り上げ	`ceil(x)`
x の切り捨て	`floor(x)`
x の四捨五入	`round(x)`
x の絶対値	`abs(x)`
$\{x_1, x_2, \cdots\}$ の最大値	`max(x1,x2,...)`
$\{x_1, x_2, \cdots\}$ の最小値	`min(x1,x2,...)`
nの階乗	`n!`または `fact(n)`
${}_m\mathrm{C}_n$	`binomial(m,n)`
n の素因数分解	`ifactor(n)`
n が素数か否か	`isprime(n)`
n 番目の素数	`ithprime(n)`
n 以上の素数	`nextprime(n)`
$\{n_1, n_2, \cdots\}$ の最大公約数	`igcd(n1,n2,...)`
$\{n_1, n_2, \cdots\}$ の最小公倍数	`ilcm(n1,n2,...)`
$m \div n$ の商	`m div n` または`_div(m,n)`
$m \div n$ の余り	`m mod n` または`_mod(m,n)`

注18) 上の表で m,n などは整数で，x, x_1 などは実数です. (abs は複素数にも使えます.)

3.5 の切り上げは?	• `ceil(3.5)`	>> 4
3.5 の切り捨ては?	• `floor(3.5)`	>> 3
3.5 の四捨五入は?	• `round(3.5)`	>> 4
-5 の絶対値は?	• `abs(-5)`	>> 5
$\left\{\frac{5}{2}, 2.4, 3\right\}$ の最大値は?	• `max(5/2,2.4,3)`	>> 3
$\left\{\frac{5}{2}, 2.4, 3\right\}$ の最小値は?	• `min(5/2,2.4,3)`	>> 2.4
5! は?	• `5!`	>> 120
${}_5\mathrm{C}_3$ は?	• `binomial(5,3)`	>> 10
247 は素数?	• `isprime(247)`	>> $FALSE$
247 の素因数分解は?	• `ifactor(247)`	>> $13 \cdot 19$
100 番目の素数は?	• `ithprime(100)`	>> 541
541 の次の素数は?	• `nextprime(542)`	>> 547
{12, 30, 48} の最大公約数は?	• `igcd(12,30,48)`	>> 6
{12, 30, 48} の最小公倍数は?	• `ilcm(12,30,48)`	>> 240

注18) ceil は' 天井'，floor は' 床',round は' 丸める' ですからそれぞれ，切り上げ, 切り捨て，四捨五入の意味を持ってもおかしくないです. abs は'absolute value(絶対値)' の略で fact は'factorial(階乗)' の略で,binomial は 2 項係数の意味. integer は整数という意味なので，ifactor で整数の因数分解ということです. prime は素数の意味．igcd は'greatest common divisor of integers',ilcm は'least common multiple of integers' の略です. div は divide の，mod は modulus の略です.

mod と div のみ使い方が異なります.

30 を 8 で割ったときの商は?	• `30 div 8`	>> 3
30 を 8 で割ったときの余りは?	• `30 mod 8`	>> 6

しかし次のようにしても大丈夫です.

30 を 8 で割ったときの商は?	• `_div(30,8)`	>> 3
30 を 8 で割ったときの余りは?	• `_mod(30,8)`	>> 6

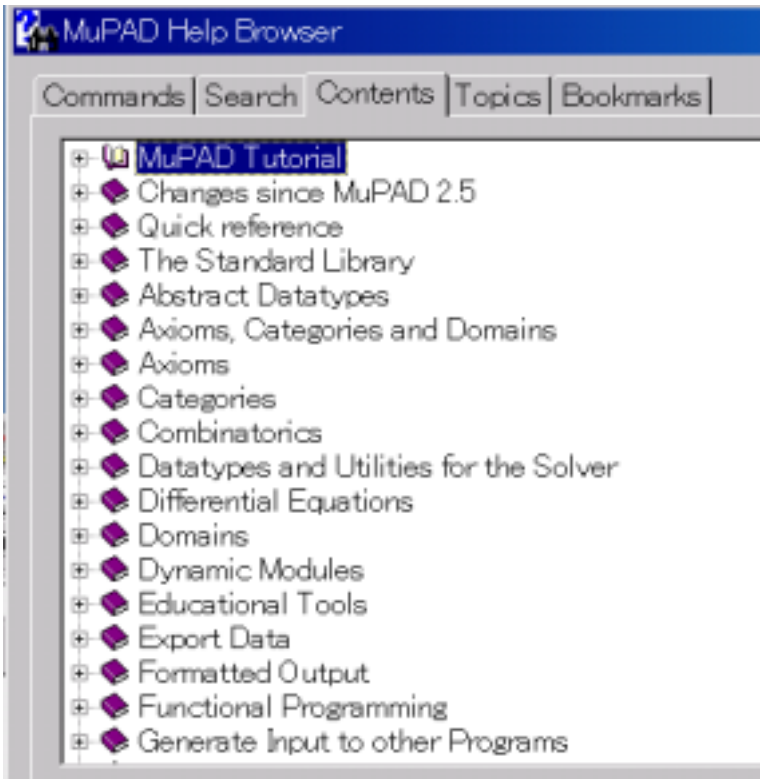
このように,「_」が前についているコマンドは，プログラムの時などに有効です.

第3章　基本操作

3.1　ヘルプの見方

3.1.1　MenuBar から

Menubar の一番右の「Help」から「Brouse Help」を選択すると, ポップアップメニュー (Help Brouser) が開きます.



「Help Brouser」からは，MuPAD の Manual 全体を見ることができます．しかし「Tutorial」や「Graphics Library」を見たいならば,「Open Help」からの方が速いです．一般に言って,

全く初めてのときは Tuorial から，
ある程度捜す言葉が分っているときは，「Help Browser」 の「Commands」や「Search」から

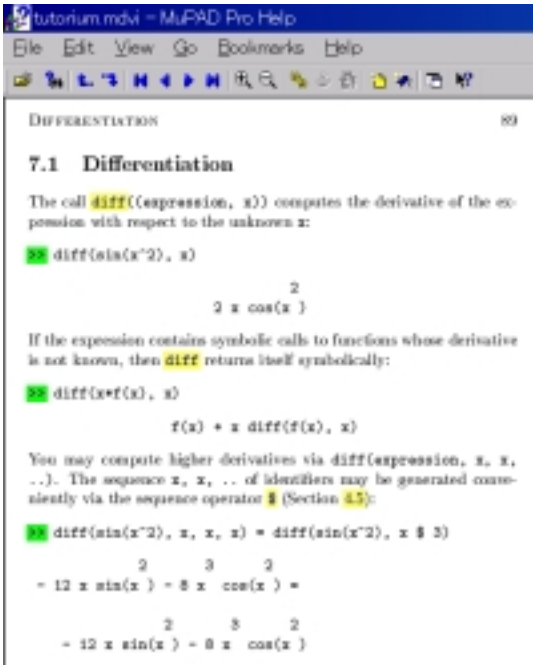
捜せば良いと思います.

例えば「微分」のコマンドについて全然解らないとします．このときは，和英辞書などで[注1)]「微分£ を調べると *differentiation* となることが解ります．そこで「Tutorial」をクリックすると「tutorium.mdvi-MuPAD Help」というファイルが開きます．これは TEX の DVI ファイルを機能拡張したもので，黄色の部分をクリックすると，ハイパージャンプします．すなわち，普通の本と同様に，Manual の前後のページや,

注1) ちなみに，私は「数 · 数式と図形の英語」（日興企画） を愛用しています.

Chapter の先頭や最後のページに移動することもできますし，ブラウザと同様に，自由にページを移動することもできます.

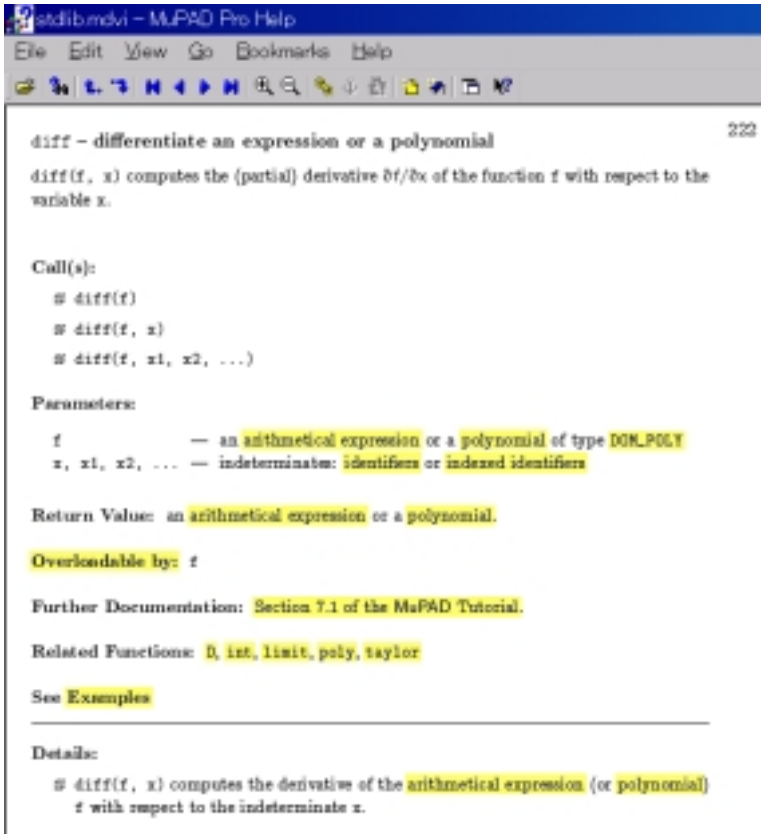
▶ を順にクリックして，目次 (index) を見ていくと,「Chapter7: differentiation and integration」が見つかります．この黄色で「differentiation」と書かれた部分をクリックすると「7·1 differentiation」 に移動します．(次図)



”The call `diff(expression,x)` computes the derivative of the expression with respect to the unknown `x`”（`diff(expression,x)` は，expression(式) を，未定義変数 `x` に関して微分します．）

とあるので, 微分は `diff` と入力すれば良いことが解ります.

緑で >> と囲ってあるのは**例**です．この部分をマウスで *input region* に *Drag&Drop* すると，**例**の式を入力できます．また，黄色で diff と囲ってある部分をクリックすると今度は「Standard library」内の `diff` の項目が開き, より詳しい説明を見ることができます.（次図)



Call は呼び出し方で, **parameter** は ( ) 内に許される parameter の種類です．下の **Examples** をクリックすると (または，▶をクリックして，ページを進めると），たくさんの例を見ることができます．(例は簡単な順に並んでいます.) さらに詳しいことは, **Details:** に載っています.

3.1.2 Help をすばやく見るには?

Help Brouser は，F2 でも開けます．これは, 入力中でも有効です．すなわち，*input region* でコマンド (またはその一部分) を入力して, F2 を押すと「help Brouser」が開き，その言葉から始まるコマンドのリストが出ます．これは，コマンドを忘れたときにも役立ちます.
さらに, ? を使って，Help Brouser を開くこともできます．また，`info(x)` で，x に関する簡単な情報を手に入れることもできます．例えば,

- `info(diff)` >> diff – a function of the system kernel [try ?diff for help]

このように, とても簡単な情報が手に入ります．最後の方に，[try ?diff for help] と書いてあるので,

- `?diff`

とすると, 今度は Help Brouser が立ち上がります.

注2)

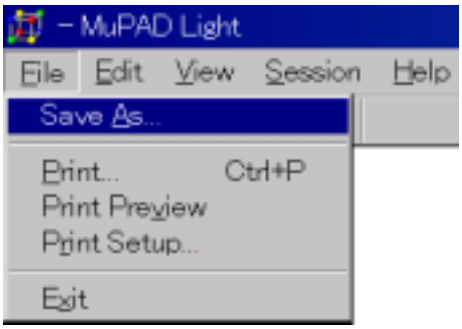
注2) 「help brouser」を開くには，MenuBar の右にある**双眼鏡**をクリックしてもできます．また「Tutorial」を見るには，MenuBar の**本**のマークをクリックしても大丈夫です.

3.2　ファイルの保存 (save& export)

3.2.1　MuPAD Light の場合

全体を保存 (binary)	write(”foo.mb”)
変数 $a, b \cdots$ だけを保存 (binary)	write(”foo.mb”, a, b, $\cdots$)
ソース全体を保存 (text)	write(Text,”foo.mu”)
変数 $a, b \cdots$ だけを保存 (text)	write(Text,”foo.mu”, a, b, $\cdots$)
ファイルを読む	read(”foo.mb”) または read(”foo.mu”)

注3)



MuPAD Light では Menu 上からは *foo*.txt としてしか保存できません. しかしこれだと，次回，同じところから再開するのは面倒です．次回も同じところからスタートしたいときは write と read を使います. テキスト形式で保存したい時は，Text を付けて保存します. (このとき, 拡張子は, 慣習上, mu とします．)

$a = 5$ と定義	• $a := 5$	>> 5
$b = 3$ と定義	• $b := 3$	>> 3
test_txt.mu として text で保存	• write(Text,”test_txt.mu”)	>>

このようにファイル名は, クォーテーションマーク (” ”) が必要です．ここで, いったん MuPAD を閉じて, また立ち上げます．そしてファイルを読み込んでみます.

test_txt.mu を読む	• read(”test_txt.mu”)	>>
a, b の値は?	• $a; b$	>> 5 >> 3

このようにデータが引き継がれました．一方, Text を付けないと，バイナリー形式で保存することになります. (このとき, 拡張子は, 慣習上, mb とします．)

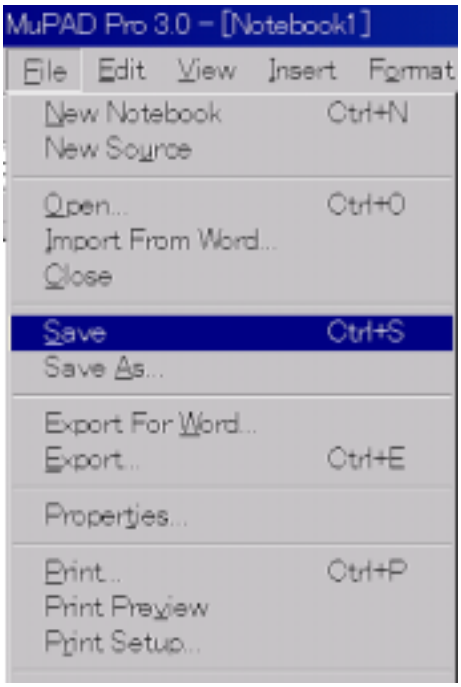
test_bin.mb として binary で保存	• write(”test_bin.mb”)	>>
test_bin.mb を読む	• read(”test_bin.mb”)	>>

注4) ”test.mu”とした場合は，Windows の場合は MuPAD のインストールフォルダー内に保存されます．これを例えば c:¥myDocument¥myMuPAD に保存する場合は次のようにします.

• write(”c : /myDocument/myMuPAD/test.mu”)

注3) 1. text というのは, editor などの様々なソフトで読めるデータ形式で，binary というのは, MuPAD 独自のデータ形式になります.
2. *foo* には, 適当な名前を自分でつけます.
注4) Manual によると，これでうまく行くはずなのですが，実は，私の環境では，binary ではうまく読めませんでした.

3.2.2 MuPAD Pro の場合



notebook の保存，読み込み

セッションの読み書きは MuPAD Light と同様に，コマンドによっても行うことができます．[注5)]
しかし, MuPAD Pro の場合は, GUI だけで, notebook の保存 (save) と, 読み込み (open) ができます．それには *File* から，*SaveAs* (名前をつけて保存) または *Save*(上書き保存) をクリックしてファイル名を指定するだけです. 保存したファイルは *foo*.mnb となります．また，*Export* をクリックすると html,doc,rtf,mu,text 形式で保存することもできます．このうち html,doc,rtf は notebook をほとんどそのままの外見で export できます．さらに html は Brouser(Internet Explorer など) で，doc と rtf は Microsoft Word など多くのソフトで開いて加工することができます.
保存した notebook を読むのも GUI だけでできます．MenuBar の *Open* を使うだけです．しかし，そのままではデータは MuPAD 本体 (Kernel) に引き継がれません．MenuBar の notebook⟶Evaluate⟶ Any Input で MuPAD Kernel に読ませます．または，引き継がせたいデータ (identifier) の部分の上で Enter を押します.

Source file の読み込み

notebook から *foo*.mu ファイルを読むことも Notebook⟶Read でできます．これで *foo*.mu のデータが引き継がれます.

[注5)] Pro では, 既存の notebook を開いた場合, NOTEBOOKPATH に notebook のパスが記憶されます．そして notebook と同じフォルダーに”foo.mu”を読み書きするには
- write(Text,NOTEBOOKPATH.”foo.mu”)
- read(NOTEBOOKPATH.”foo.mu”) とすれば良いです.

3.3　他の基本操作

3.3.1　Kernel と Library

Help brouser をみると Standard library,Graphics library という項目が見えます．MuPAD は Kernel(核) と Library(書庫) の 2 つに別れています. Kernel は C,C++で書かれ, その中では MuPAD の構文解析, 式の簡略化[注6)] が行われ, その中で MuPAD 独自のプログラム言語を定義しています．Library は その MuPAD の言語で書かれています．[注7)] library の最も簡単な情報は，`info()` で得られます.

```
•info()    >> − − Libraries:
Ax,        Cat,        Dom,      Graph,   RGB,      Series,    Type,
adt,       combinat,   detools,  fp,      generate,  groebner,  import,
intlib,    linalg,     linopt,   listlib, matchlib,  module,    numeric,
numlib,    ode,        orthpoly, output,  plot,      polylib,   prog,
property,  solvelib,   specfunc, stats,   stdlib,   stringlib, student,
transform
```

その中で最も基本的で, よく使うのが Stdlib(Standard library) です．Standard library のコマンドは，`diff` のように，単純に入力すればよいですが，他のライブラリーのコマンドは，そのライブラリー名を付けて，呼び出します．例えば，第 1 章で見たように，曲線を描くコマンド (`Curve2d`, `Curve3d`) は，plot lib(library for plot) に入っているので，`plot :: Curve2d` のように入力します．[注8)] この本で扱うコマンドのほとんどは,「Standard library」 の範囲ですから，ほとんどの場合は 「help brouser」⟶「Contents」⟶「Standard library」からも情報が得られます.

3.3.2　Kernel との切断

通常,MuPAD Pro の notebook は Kernel と接続されていますが, 計算の実行中に (エラーなどによって) Kernel との接続が切れることがあります. この場合は MenuBar の Notebook⟶Connect to で Kernel と再接続することができます.

3.3.3　入力補助

MuPAD Pro の場合は 入力補助もあります．途中までコマンドを入力して Crtl + Space を押すと (またはメニューバーの「NoteBook」から 「complete Word」 を選択すると) 残りのコマンドを補完してくれます. コマンドの候補が 2 つ以上になる時は PopupMenu がでて, 候補を選択できます．[注9)] また,「Manual」中の**例**において，緑で >> と囲ってある所は，マウスで *input region* に *Drag&Drop* すると，その式が貼り付けられます.

注6) internal simplication

注7) 一部のコマンドは Kernel 内で， C,C++を使って書かれています．例えば fact(階乗) を Manual で見ると, 最後に ”fact is a function of the system kernel” と書いてあります．(他には _plus,_minus なども kernel function です) MuPAD 言語で書かれている部分は expose でその内容を見ることができます．例えば
• expose(fact)
で fact のプログラムを見ることができます.

注8) 実は，`export(plot)` のようにすれば，いちいち `plot` と入力しなくとも，`Curve3d` だけで呼び出せるのですが，名前の衝突の問題（名前空間の問題) があって，いろいろ面倒なので，この本では，`export` は使いません.

注9) そのはず ··· ですが，候補が 2 つ以上のときは，私の環境ではなぜかうまく働きません. そのうえ,MuPAD の動作もおかしくなります.

3.3.4 計算の停止

ithprime(n) は, n 番目の素数を求めるコマンドですが，例えば

- ithprime(10000000000000000000)

とした場合, 計算に非常に時間がかかり，砂時計をジーと見ることになります．このような場合には, MuPAD Pro では,MenuBar の Notebook ⟶Stop Calculation で計算を終了させることができます．しかし, MuPAD Light の場合は，MuPAD を終了させるしかありません.

第4章　文字式(expression)の操作

4.1　文字式の入力

4.1.1　初等関数

文字式も，数字と同じように，+, −, ∗, /, ^ で, 和, 差, 積, 商, 累乗ができます．さらに，初等関数 (三角関数, 指数関数など) と，π, e の入力は，下の表のように入力します．

$\sin x, \cos x, \tan x$	`sin(x), cos(x), tan(x)`
$\sin^{-1} x, \cos^{-1} x, \tan^{-1} x$	`arcsin(x), arccos(x), arctan(x)`
$\sinh x, \cosh x, \tanh x$	`sinh(x), cosh(x), tanh(x)`
a^x	`_power(a, x)` または `a^x`
e^x	`exp(x)` または `E^x`
$\log_a x$	`log(a, x)`
$\log x$	`ln(x)` または `log(E, x)`
x の絶対値 $\lvert x \rvert$	`abs(x)`
x の実数成分 (*real part*)	`Re(x)`
x の虚数成分 (*imaginary part*)	`Im(x)`
$x \longrightarrow a + bi (a, b$ 実数$)$	`rectform(x)`
e, π	`E, PI`

注1)

関数の入力は，( ) の中に入れるだけです．

- `sin(PI); cos(PI/2); tan(PI/4)`　　>> 0　>> 0　>> 1

$\sin\pi = 0,\ \cos\left(\frac{\pi}{2}\right) = 0,\ \tan\left(\frac{\pi}{4}\right) = 1$ を表しています．

- `exp(1); E; exp(2); E^2`　　>> e　>> e　>> e^2　>> e^2

$\exp(x) = e^x$ ですから，$\exp(1) = e^1 = e,\ \exp(2) = e^2$ です．

- `ln(E); ln(E^2); ln(3)`　　>> 1　>> 2　>> ln(3)

注1) 1. 三角関数 ($\sin x, \cos x, \tan x$) の単位は，ラジアン (*rad*) です．
2. `ln(x)` の先頭の文字は，エル (小文字の L)．logarithm natural (自然対数) の略です．
3. $\sin^{-1} x, \cos^{-1} x, \tan^{-1} x$ は，それぞれ $\sin x, \cos x, \tan x$ の逆関数です．$\sinh x, \cosh x, \tanh x$ は双曲線関数と呼ばれ, 次の式で定義されます．

$$\sinh x = \frac{e^x - e^{-x}}{2},\ \cosh x = \frac{e^x + e^{-x}}{2},\ \tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

これ以外にも様々な関数が MuPAD では使えます．例えば *quick reference* の Special Mathmatical functions を見てください．
4. x を `assume(x, Type :: Real)` などを使って実数に制限していないときは,`exp(x), ln(x), sin(x), cos(x), ···` などは，複素数値を返します．そのため $\log x + \log y = \log(xy)$ などの基本公式は一般には成り立ちません．

$\ln(x) = \log x$ ですから，$\ln(E) = \log e = 1,\ \ln(E^2) = \log e^2 = 2$ です．また，$\log 3$ は, 簡単にできないので, このままです.

- `log(3,3); log(3,9); log(3,1/3)`　　>> 1　>> 2　>> −1

`log(a,x)` $= \log_a x$ ですから，$\log_3 3 = 1,\ \log_3 9 = 2,\ \log_3 \frac{1}{3} = -1$ を表しています.

- `_power(3,4); _power(x,3); x^3`　　>> 81　>> x^3　>> x^3

`_power(a,x)` $= a^x$ を表すので，`_power(3,4)` $= 3^4 = 81$ です.

- `abs(−3); abs(5)`　　>> 3　>> 5

`abs(x)` $= |x|$ ですから，$|-3| = 3,\ |5| = 5$ を表しています.

( ) のなかには，文字を入れても大丈夫です．文字を入れると，文字式 (expression) になります．この文字式をさらに，( ) の中に入れることが出来ます.

- `sin(2*x); exp(x^2); ln(abs(x))`　　>> $\sin 2 \cdot x$　>> $\exp(x^2)$　>> $\ln(|x|)$

このように，x に値が設定されていないときは，これ以上簡単に出来ないので, そのままですが，x に値が設定されていれば，その値を代入して, 計算した値が出力されます．x に値を設定するには，等号でなく「:=」を使用します.

- `x := PI; sin(x); cos(2*x)`　　>> 0　>> 1

$\sin\pi = 0,\ \cos 2\pi = 1$ と簡単になります．これは，入れ子になっていても, 同様です．(再帰的に計算されます.) また，x の値は，`delete x` で定義を消去するか，`x := PI/2` などと新しく値を設定するまで有効です.

- `ln(sin(x/2)); exp(cos(x))`　　>> 0　>> e^{-1}

$\log\left(\sin\frac{\pi}{2}\right) = \log 1 = 0,\ e^{\cos\pi} = e^{-1}$ を表しています.

- `delete x; sin(x)`　　>>　>> $\sin(x)$

`delete x` で定義を消去したので，もはや，`sin(x)` は簡単になりません.「:=」で文字だけでなく, 式を定義することも出来ます.

- `a := log(2,x)`　　>> $\log_2 x$
- `x := 4`　　>> 4
- `a; sin(a*PI)`　　>> 2　>> 0

`a` $= \log_2 4 = 2,\ \sin(a \times \pi) = \sin(2\pi) = 0$ と計算されます.

4.1.2 文字式の操作

文字式の操作が，MuPAD の中心となります．このあとは，興味のある章からお読みください．以上の知識で, ほとんど問題なく各章が読めると思います．ただし，やや高度な演算では，assume を使って，property を設定したり，op を使って，operand をいじることが必要になります. これらについては，必要のある毎に簡単に説明しますが，詳しい説明は，第 6 章，第 7 章もお読みください.

- 数式の同値変形 (因数分解, 式の展開, 整理など) – この章の後半
- 関数の定義, 式への代入 – 第 5 章
- 方程式, 不等式を解く – 第 8 章
- 微分, 積分 – 第 9 章, 第 10 章
- 平面のグラフの描写 – 第 11 章
- 空間のグラフの描写 – 第 12 章
- 行列, ベクトル – 第 13 章

4.2 数式の同値変形 (manipulation of expressions)

以下のコマンドを使うと文字式の変形ができます.

f を x に関し整理	`collect(f, x)`
f を $x, y, \cdots$ に関し整理	`collect(f, [x, y, ···])`
f を展開	`expand(f)`
$g_1, g_2, \cdots$ は展開しないで, f を展開	`expand(f, g1, g2, ···)`
f を 因数分解	`factor(f)`
f の指数をまとめる ($a^x * a^y = a^{x+y}$)	`combine(f)`
f を 既約分数にする (結果は展開)	`normal(f)`
f を部分分数に展開 (変数 x に関して)	`partfrac(f, x)`
f を *target* の関数を使って書き直す	`rewrite(f, target)`
f の単純化	`simplify(f)` または `Simplify(f)`

注2)

expand, combine, simplify, Simplify には option をつけることもできます．また rewrite には，target が必須です.

- combine の option – *sincos*, *exp*, *ln*, *log*, *sinhcosh*, *arctan*
- rewrite の target – *sin*, *cos*, *tan*, *sincos*, *sinh*, *cosh*, *tanh*, *arcsin*, *arccos*, *arctan*, *exp*, *ln* *cot*, *coth*, *andor*, *diff*, *D*, *fact*, *gamma*, *heaviside*, *sign*, *sinhcosh*, *piecewise*
- simplify の option – *sin*, *cos*, *exp*, *ln*, *sqrt*, *logic*, *relation*
- Simplify の option – *All*, Steps = *n*, ApplyRule = *applyFunction*, Discard = *discardFunction*, Goal = *a*, MaxInfoDepth = *depth*, OutputType = *output*, Remember = *b*, RuleBase = *base*, SelectRules = *selFunction*, Strategy = *strat*, Valuation = *valFunction*

simplify と Simplify は，式を「簡単にする」コマンドです．これが, 一番お手軽ですが,「簡単さ」の定義は, 人により，また状況により異なるので，思った結果が得られないこともあります．その場合は, 用途別のコマンド (例えば, 因数分解なら factor) をお使いください.

注2) factor でも normal のように既約分数にできますが, factor は分母, 分子を因数分解します.

4.2.1 simplify と Simplify

Simplify は MuPAD 3.0 から導入されました．Simplify は simplify より遅いですが，simplify では簡単にできない式も簡単にします．

- `a := log(8, 2)` >> $\log_8 2$
- `simplify(a)` >> $\log_8 2$
- `Simplify(a)` >> $\frac{1}{3}$

`simplify` には

$$sin, cos, exp, ln, sqrt, logic, relation$$

の option をつけることができます．option をつけると その option の種類の関数**だけ** を簡単にします．

- `b := sin(x)^2 + cos(x)^2 + ln(8)/ln(2)` >> $\frac{\ln(8)}{\ln(2)} + \cos(x)^2 + \sin(x)^2$
- `simplify(b, ln)` >> $\cos(x)^2 + \sin(x)^2 + \frac{1}{3}$
- `simplify(b, sin)` >> $\frac{\ln(2) + \ln(8)}{\ln(2)}$
- `simplify(b)` >> $\frac{4}{3}$

`Simplify` には

$$\text{Steps} = n\ (n \text{ は自然数}), All$$

などの option があります．例えば，`Simplify(f, Steps = 100)` とすると， f を簡単にするステップを 100 ステップまで実行して，結果を返します．(default では 50 ステップまで実行します.) また `Simplify(f, All)` とすると 50 ステップまでにでてきた候補を全て表示します．複数の option を指定するときは 「,」で区切って指定します．例えば `Simplify(f, 100, All)` とすると, 100 ステップまでにでてきた候補を全て表示します．[注3)]

- `a := log(8, 2)` >> $\log_8 2$
- `Simplify(a, All)` >> $\left[\frac{1}{3}, \log_8(2), \frac{\ln(2)}{\ln(8)}, \ln\left(2^{\frac{1}{\ln(8)}}\right)\right]$
- `Simplify(a, All, Steps = 20)` >> $\left[\log_8(2), \frac{\ln(2)}{\ln(8)}, \ln\left(2^{\frac{1}{\ln(8)}}\right)\right]$
- `Simplify(a, Steps = 20)` >> $\log_8(2)$
- `Simplify(a, Steps = 30)` >> $\frac{1}{3}$

[注3)] `Simplify` の option は,

ApplyRule = *applyFunction*, Discard = *discardFunction*, Goal = *a*, MaxInfoDepth = *depth*, OutputType = *output*, Remember = *b*, RuleBase = *base*, SelectRules = *selFunction*, Strategy = *strat*, Valuation = *valFunction*　があります.

これから, $a \longrightarrow \frac{1}{3}$ という変形は, 21 ステップから 30 ステップの間に行われたということが，分ります.

しかし，Simplify も万能ではありません．combine, rewrite や simplify の方が良いときもあります.

- a := log(10, 5) + log(10, 20) >> $\log_{10}(5) + \log_{10}(20)$
- combine(a, log) >> 2
- Simplify(a) >> $\log_{10}(5) + \log_{10}(20)$
- Simplify(a, Steps = 200) >> $\log_{10}(5) + \log_{10}(20)$
- Simplify(a, Steps = 300) >> 2

この場合は combine の方が良かったです．Simplify は, Steps= 300 でやっと簡単にできました.

分母の有理化も Simplify は苦手です．(というか，全くその気が無いようです.) 分母の有理化は simplify(f, sqrt) または radsimp(f) の方が得意です.

- b := 1/(sqrt(3) + 2) >> $\dfrac{1}{\sqrt{3}+2}$
- Simplify(b) >> $\dfrac{1}{\sqrt{3}+2}$
- Simplify(b, Steps = 10000) >> $\dfrac{1}{\sqrt{3}+2}$
- simplify(b, sqrt) >> $2 - \sqrt{3}$
- radsimp(b) >> $2 - \sqrt{3}$

4.2.2 expand, factor, collect (整式を例にして)

[expand]

整式を展開するには，expand を使います.
まず $(x^2 - 4x + 5)(x^2 - 3)$ を展開してみます.

- expand((x^2 − 4 ∗ x + 5) ∗ (x^2 − 3)) >> $12 \cdot x + 2 \cdot x^2 - 4 \cdot x^3 + x^4 - 15$

$(a + b + c)^2$の展開は? • expand((a + b + c)^2) >> $2 \cdot a \cdot b + 2 \cdot a \cdot c + 2 \cdot b \cdot c + a^2 + b^2 + c^2$

f を展開するときに，g を展開しないで，f のみを展開したいときは，expand(f, g) とします.

- eq := (t + 2) ∗ (x − t) + t^2 + 2 ∗ t >> $2 \cdot t - (t - x) \cdot (t + 2) + t^2$
- expand(eq, t + 2) >> $2 \cdot t - t \cdot (t + 2) + x \cdot (t + 2) + t^2$
- expand(eq) >> $2 \cdot x + t \cdot x$

[collect]

x に関し整理したいときは collect(f, x) を使います.

- f := x^2 + 3 ∗ x^2 ∗ y + 2 ∗ y^2 − 2 ∗ x − y + 1　　>> $3 \cdot x^2 \cdot y - y - 2 \cdot x + x^2 + 2 \cdot y^2 + 1$

x に関し整理すると?

- collect(f, x)　　>> $x^2 \cdot (3 \cdot y + 1) - y - 2 \cdot x + 2 \cdot y^2 + 1$

x, x^2 の項に分けられています. しかし, 項べきの順に並んでいるわけではありません. 次は, y に関して整理してみます.

- collect(f, y)　　>> $y \cdot (3 \cdot x^2 - 1) - 2 \cdot x + x^2 + 2 \cdot y^2 + 1$

[factor]

因数分解には, factor を使います.

- factor(x^2 + 3 ∗ x + 2)　　>> $(x + 2) \cdot (x + 1)$
- expand(%)　　>> $3 \cdot x + x^2 + 2$

展開すると, 元に戻ります. いろいろ因数分解してみます.

◇$x^4 - 8x^2 - 9$ の因数分解は?

- factor(x^4 − 8 ∗ x^2 − 9)　　>> $(x - 3) \cdot (x + 3) \cdot (x^2 + 1)$

◇$x^2 + 4xy + 3y^2 + x + 5y - 2$ の因数分解は?

- factor(x^2 + 4 ∗ x ∗ y + 3 ∗ y^2 + x + 5 ∗ y − 2)　　>> $(x + y + 2) \cdot (x + 3 \cdot y - 1)$

◇$(y + z)(z + x)(x + y) + xyz$ の因数分解は?

- siki := (y + z) ∗ (z + x) ∗ (x + y) + x ∗ y ∗ z　　>> $(x + y) \cdot (x + z) \cdot (y + z) + x \cdot y \cdot z$
- factor(siki)　　>> $(x \cdot y + x \cdot z) \cdot (x + y + z)$

【ちょっと寄り道】 整式の除法

整式の展開・因数分解を扱ったので, ついでに割り算も見てみます.

整式 $f(x)$ を整式 $g(x)$ で割ったときの商と剰余	`divide(f(x), g(x))`
多項式 f(x,y) を多項式 g(x,y) で割ったときの商と剰余を x についての整式と見て求める.	`divide(f(x), g(x), [x])`

整式 $f(x)$ を整式 $g(x)$ で割ったときの商と余りは, `divide(f(x), g(x))` で求めます.

◇$2x^3 - 12x + 9$ を $(x + 3)$ で割ったときの商と余りは?

- `divide(2 * x^3 - 12 * x + 9, x + 3)` >> $2x^2 - 6x + 6,\ -9$

これは商が $2x^2 - 6x + 6$, 余りが -9 ということです.

$$
\begin{array}{r}
2x^2\ -6x\ \ +6 \\
\hline
x+3\ \big)\ 2x^3\ \ \ \ -12x\ +9 \\
\underline{\ 2x^3+6x^2\ \ \ \ \ \ \ \ } \\
\ \ \ -6x^2-12x\ \ \\
\ \ \ \underline{-6x^2-18x\ \ \ } \\
\ \ \ \ \ \ \ 6x\ +9 \\
\ \ \ \ \ \ \ \underline{6x+18} \\
\ \ \ \ \ \ \ \ \ -9
\end{array}
$$

2 変数の場合は, どの文字に関する式と考えるかによって結果が変わってきます. このような時は divide(f,g,[x]) のように変数を指定します.

◇$8x^3 + y^3$ を $x^2 + y^2$ で割ったときの商と余りを x の整式と見て計算すると?

- `divide(8 * x^3 + y^3, x^2 + y^2, [x])` >> $8 \cdot x,\quad y^3 - 8 \cdot x \cdot y^2$

◇$8x^3 + y^3$ を $x^2 + y^2$ で割ったときの商と余りを y の整式と見て計算すると?

- `divide(8 * x^3 + y^3, x^2 + y^2, [y])` >> $y,\quad 8 \cdot x^3 - x^2 \cdot y$

$$
\begin{array}{r}
8x \\
\hline
x^2+y^2\ \big)\ 8x^3\ \ \ \ \ \ +y^3 \\
\underline{\ 8x^3\ \ +8xy^2\ \ \ } \\
\ \ \ \ \ -8xy^2\ \ +y^3
\end{array}
\qquad\qquad
\begin{array}{r}
y \\
\hline
y^2+x^2\ \big)\ y^3\ \ \ \ \ +8x^3 \\
\underline{\ y^3\ \ \ \ \ +x^2y\ } \\
\ \ \ \ \ 8x^3 - x^2y
\end{array}
$$

4.2.3　normal,factor,partfrac(有理式を例にして)

[normal,factor]

有理式を約分，通分したいときは，normal または factor をつかいます．このとき，normal を使うと，結果の式の分母，分子は展開されます．factor を使うと，結果の式の分母，分子は因数分解されます．

◇$\frac{x^2+3x-4}{2x^2-4x+2}$ を約分してみましょう．

- (x^2 + 3 * x - 4)/(2 * x^2 - 4 * x + 2)　　>> $\frac{3x+x^2-4}{2x^2-4x+2}$
- normal(%)　　>> $\frac{x+4}{2x-2}$

$\frac{3x+x^2-4}{2x^2-4x+2} = \frac{(x+4)(x-1)}{2(x-1)^2} = \frac{x+4}{2(x-1)}$ を表しています．今度は通分です．

◇$\frac{2x-3}{x^2-3x+2} - \frac{3x-2}{x^2-4}$ を通分してみましょう．

- y := (2 * x - 3)/(x^2 - 3 * x + 2) - (3 * x - 2)/(x^2 - 4)　　>> $\frac{2\cdot x-3}{x^2-3\cdot x+2} - \frac{3\cdot x-2}{x^2-4}$
- normal(y)　　>> $-\frac{x-4}{x+x^2-2}$
- factor(y)　　>> $-\frac{x-4}{(x+2)\cdot(x-1)}$

このように，normal は因数分解しませんが，factor は因数分解します．なお, この場合は simplify でも簡単にできます．

- simplify(y)　　>> $-\frac{x-4}{x+x^2-2}$

検算してみると，合っていることが判ります．

$$\frac{2x-3}{x^2-3x+2} - \frac{3x-2}{x^2-4} = \frac{(2x-3)(x+2)-(3x-2)(x-1)}{(x-1)(x-2)(x+2)} = -\frac{x^2-6x+8}{(x-1)(x-2)(x+2)}$$
$$= -\frac{\cancel{(x-2)}(x-4)}{(x-1)\cancel{(x-2)}(x+2)} = -\frac{x-4}{(x-1)(x+2)}$$

factor, expand, normal は，他の関数でも使えます．

- factor(sin(x)^2 - 1)　　>> $(\sin(x)-1)\cdot(\sin(x)+1)$
- expand(%)　　>> $\sin(x)^2 - 1$
- normal((sin(x) + 1)/(sin(x)^2 - 1))　　>> $\frac{1}{\sin(x)-1}$

内部的には, $\sin(x)$ を置き換えて, factor, expand, normal した後に, 元に戻しています．

[partfrac]

partfrac(f, x) は，f を，x に関して，部分分数に直すときに使います．変数を x しか含まないときは，x は省略できます．

- partfrac(partfrac(1/(x^2 − 1))) >> $\frac{1}{2\cdot(x-1)} - \frac{1}{2\cdot(x+1)}$

normal で元に戻ります．

- normal(%) >> $\frac{1}{x^2-1}$

変数が 2 つ以上のときは, 変数の指定が必要です．

- partfrac(partfrac(a/(x^2 − 1), x)) >> $\frac{a}{2\cdot(x-1)} - \frac{a}{2\cdot(x+1)}$
- partfrac(partfrac(a/(x^2 − 1), a)) >> $\frac{a}{x^2-1}$

x に関しては, 部分分数に変形できますが，a に関しては, これ以上変形できないので，そのままです．したがって，変数を省略すると, エラーになります．

- partfrac(partfrac(a/(x^2 − 1)))

  >> Error: you should supply a variable as second argument [partfrac]

4.2.4 combine, expand, rewrite(三角関数, 指数 · 対数関数を例にして)

[combine]

combine は default では, $a^x \cdot a^y = a^{x+y}, (a^x)^y = a^{xy}, a^x \cdot b^x = (ab)^x$ のように，指数を一つにまとめるとき**だけ**働きます．[注4] これに option を付けると その option で指定した種類の関数も同様の変形をします．option は *sincos*, *exp*, *ln*, *log*, *sinhcosh*, *arctan* が使えます．

- sqrt(3) ∗ sqrt(6); combine(%) >> $\sqrt{6}\cdot\sqrt{3}$ >> $3\cdot\sqrt{2}$
- 2^(1/3) ∗ 4^(1/3); combine(%) >> $4^{\frac{1}{3}}\cdot 2^{\frac{1}{3}}$ >> 2

文字のときでも, 大丈夫です．

- a^x ∗ a^y; combine(%) >> $a^x \cdot a^y$ >> a^{x+y}

ところが，底が自然対数 e のときは，うまく行きません．

- a := sqrt(E) ∗ sqrt(E^3); combine(a) >> $\sqrt{e^3}\cdot\sqrt{e}$ >> $e^{\frac{3}{2}}\cdot\sqrt{e}$
- b := (E^2)^(1/3) ∗ E^(1/3); combine(b) >> $\sqrt[3]{e^2}\cdot\sqrt[3]{e}$ >> $e^{\frac{2}{3}}\cdot e^{\frac{1}{3}}$

combine では簡単になりません．このような時は option で *exp* を指定します．

- combine(a, exp) >> e^2
- combine(b, exp) >> e

[注4] ただし, 後の 2 つは y が整数とかある条件の下で成り立ちます．

同様に

- `E^x * E^y; combine(%, exp);` >> $e^x \cdot e^y$ >> e^{x+y}

このように，option を指定しないと，うまくまとめてくれません．

対数の場合は，*ln*(自然対数のとき), *log*(一般の対数のとき) を指定します．すると，適当な条件の下で，combine は $\log_a x + \log_a y \longrightarrow \log_a xy$, $p \log_a x \longrightarrow \log_a x^p$ と変形します．

- `a := log(10, 2) + log(10, 5)` >> $\log_{10} 2 + \log_{10} 5$
- `b := 3 * log(10, 2)` >> $3 \cdot \log_{10} 2$
- `combine(a); combine(b)` >> $\log_{10} 2 + \log_{10} 5$ >> $3 \cdot \log_{10} 2$

このように，ただの combine では, 対数は変形できません．しかし，option で *log* を付けるとうまく行きます．

- `combine(a, log); combine(b, log)` >> 1 >> $\log_{10} 8$

底が *e* のときは option で *ln* を指定します．ただし option で *log* と指定しても (MuPAD の内部で *ln* に直すので) 大丈夫です．

- `delete a, b : a := ln(2) + ln(5)` >> ln(2) + ln(5)
- `b := 3 * ln(2)` >> $3 \cdot \ln(2)$
- `combine(a); combine(b)` >> ln(2) + ln(5) >> $3 \cdot \ln(2)$

このように，ただの combine では, 自然対数は変形できません．しかし，option で *ln*, または, *log* を付けるとうまく行きます．

- `combine(a, ln); combine(b, ln)` >> ln(10) >> ln(8)
- `combine(a, log); combine(b, log)` >> ln(10) >> ln(8)

三角関数では *sincos* をつけます．すると，combine(f, sincos) はいわゆる**次数下げ**に働きます．つまり**積**を**和**に直します．

- `t1 := sin(x) * cos(y)` >> $\cos(y) \cdot \sin(x)$
- `combine(t1, sincos)` >> $\frac{\sin(x-y)}{2} + \frac{\sin(x+y)}{2}$
- `t2 := sin(x)^2; combine(t2, sincos)` >> $\sin(x)^2$ >> $\frac{1}{2} - \frac{\cos(2 \cdot x)}{2}$
- `t3 := cos(x)^2 combine(t3, sincos)` >> $\cos(x)^2$ >> $\frac{1}{2} + \frac{\cos(2 \cdot x)}{2}$

注5)

注5) combine(f, sinhcosh) も combine(f, sincos) と非常に似た変形をします

[expand]

三角関数や指数・対数関数では expand は combine の逆の変形を行います.

- x1 := a^(x + y) : expand(x1) >> $a^x \cdot a^y$
- x2 := ln(10 * x) : expand(x2) >> $\ln(10) + \ln(x)$
- x3 := sin(x + y) : expand(x3) >> $\cos(x) \cdot \sin(y) + \cos(y) \cdot \sin(x)$
- x4 := sin(2 * x) : expand(x4) >> $2 \cdot \cos(x) \cdot \sin(x)$
- x5 := cos(2 * x) : expand(x5) >> $\cos(x)^2 - \sin(x)^2$

注6)

[rewrite]

rewrite(f, target) は , f を *target* の関数に書き直します. *target* は

$$sin, cos, tan, sincos, sinh, cosh, tanh, arcsin, arccos, arctan, exp, ln$$

などが使えます. 注7)

sincos は $\sin x$ と $\cos x$ で書き直します. *sin* は *sincos* で書き直した後, $\cos^2 x = 1 - \sin^2 x$ を使って $\sin x$ にできるだけ直します. *cos* も同様です. どちらも $\sqrt{\ }$ は使わないようです.

- f1 := sin(x)^2 + cos(x) >> $\sin(x)^2 + \cos(x)$
- rewrite(f1, sincos) >> $\sin(x)^2 + \cos(x)$
- rewrite(f1, cos) >> $\cos(x) + 1 - \cos(x)^2$
- rewrite(f1, sin) >> $\sin(x)^2 - 2\sin\left(\frac{x}{2}\right)^2 + 1$

最後の例では, 倍角公式:$\cos 2\theta = 1 - 2\sin^2\theta$ を使っています. option に *tan* をつけると

$$\sin(x) = \frac{2\tan\frac{x}{2}}{1 + \tan^2\frac{x}{2}}, \cos(x) = \frac{1 - \tan^2\frac{x}{2}}{1 + \tan^2\frac{x}{2}}$$

を使って tan に直します.

- rewrite(sin(x), tan) >> $\frac{2 \cdot \tan\frac{x}{2}}{\tan^2\frac{x}{2} + 1}$
- rewrite(cos(x), tan) >> $-\frac{\tan^2\frac{x}{2} - 1}{\tan^2\frac{x}{2} + 1}$

注6) このように同じコマンドを繰り返し働かせるときは map([f, g, ···], command) で [*command*(*f*), *command*(*g*), ···] が出力されます. 例えば, 上の場合は

•map([x1, x2, x3, x4, x5], expand) >> $[a^x \cdot a^y, ln(10) + ln(x), \cdots, \cos(x)^2 - \sin(x)^2]$

となります. この map も良く使われるコマンドです. (第 6 章)

注7) rewrite の target は他に,

$$cot, coth, andor, diff, D, fact, gamma, heaviside, sign, sinhcosh, piecewise$$

があります. なお, *sin*, *cos*, *sinh*, *cosh*, *arccos*, *arccot*, *arcsin*, *arctan* は, MuPAD 3.0 で初めて導入されました.

注8)

option で *exp*, *ln* をつけると $\exp(x) = e^x, \ln(x) = \log x$ を使って書き直します.

- `log(3, 5)` >> $log_3 5$
- `rewrite(%, ln)` >> $\frac{\ln(5)}{\ln(3)}$
- `3^x` >> 3^x
- `rewrite(%, exp)` >> $e^{\ln(3)\cdot x}$

$e^{\log_e 3} = 3$ なので, $e^{\ln(3)\cdot x} = 3^x$ です.

【参考】実は MuPAD は f を同値変形するとき，f を複素関数とみて変換しています．複素数の範囲まで考えると，三角関数は指数関数で, 逆に，指数関数は三角関数で，書き直すことが出来ます.

- `rewrite(sin(x), exp)` >> $\frac{i}{2}\cdot e^{(-i)\cdot x} - \frac{i}{2}\cdot e^{i\cdot x}$
- `rewrite(cos(x), exp)` >> $\frac{e^{(-i)\cdot x}}{2} + \frac{e^{i\cdot x}}{2}$
- `rewrite(exp(I * x), sincos)` >> $\cos(x) + i\cdot\sin(x)$

注9)

注8)

$$\cos x = \cos 2\cdot\left(\frac{x}{2}\right) = \cos^2\frac{x}{2} - \sin^2\frac{x}{2} = \frac{\cos^2\frac{x}{2} - \sin^2\frac{x}{2}}{\cos^2\frac{x}{2} + \sin^2\frac{x}{2}}$$

よって，分母・分子を $\cos^2\frac{x}{2}$ で割ると

$$\cos x = \frac{1 - \frac{\sin^2\frac{x}{2}}{\cos^2\frac{x}{2}}}{1 + \frac{\sin^2\frac{x}{2}}{\cos^2\frac{x}{2}}} = \frac{1 - \tan^2\frac{x}{2}}{1 + \tan^2\frac{x}{2}}$$

$\sin x$ も同様にして,

$$\sin x = \sin 2\cdot\left(\frac{x}{2}\right) = 2\sin\frac{x}{2}\cos\frac{x}{2} = \frac{2\sin\frac{x}{2}\cos\frac{x}{2}}{\cos^2\frac{x}{2} + \sin^2\frac{x}{2}} = \frac{2\frac{\sin\frac{x}{2}}{\cos\frac{x}{2}}}{1 + \frac{\sin^2\frac{x}{2}}{\cos^2\frac{x}{2}}} = \frac{2\tan\frac{x}{2}}{1 + \tan^2\frac{x}{2}}$$

注9) 一般に x が複素数のとき

$$e^{ix} = \cos x + i\sin x \qquad \cdots ①$$

① で x に $-x$ を代入して

$$e^{-ix} = \cos x - i\sin x \qquad \cdots ②$$

①,② より

$$\cos x = \frac{e^{ix} + e^{-ix}}{2}, \sin x = \frac{e^{ix} - e^{-ix}}{2i}$$

4.3 2 つの式の比較 (検算など)

$f-g=0$?	simplify(f − g)
$f=g$?	bool(f = g)
$f=0$?	iszero(f)
$f=g$?	testeq(f, g)

注10) 第 2 章でも扱いましたが，2 つの式が等しいか否かを調べるには上のいずれかを使います．普通は testeq(f, g) を使うのが良いと思います．返り値は TRUE(真), FALSE(偽), UNKNOWN(不明) のいずれかです．

Simplify, simplify, combine, rewrite などでも, f から g に変形できないときが，しばしばあります．このような時, 手計算で f⟶g を導き，それを MuPAD で検算することができます．例えば,

三角関数の合成公式:

$$a\sin\theta+b\cos\theta=\sqrt{a^2+b^2}\sin(\theta+\alpha)\left(\text{ただし }\sin\alpha=\frac{b}{\sqrt{a^2+b^2}},\cos\alpha=\frac{a}{\sqrt{a^2+b^2}}\right)$$

を利用した変形を考えて見ます．合成公式によると

$$\sin x+\cos x=\sqrt{2}\sin\left(x+\frac{\pi}{4}\right)$$

となるはずです．しかし，合成公式は， MuPAD は不得意です．

- f := sin(x) + cos(x) > cos(x) + sin(x)
- g := sqrt(2) ∗ sin(x + PI/4) >> $\sqrt{2}\cdot\sin\left(\frac{\pi}{4}+x\right)$
- Simplify(f) >> cos(x) + sin(x)
- Simplify(g) >> $\sqrt{2}\cdot\sin\left(\frac{\pi}{4}+x\right)$

このように，f と g は Simplify では，まったく簡単になりません．もうすでに簡単だと，MuPAD が考えているためです．しかし，$(f-g)$ は簡単になります．

- Simplify(f − g) >> 0

または testeq を使っても同じです.

- testeq(f, g) >> $TRUE$

testeq は Simplify を利用しています．そのせいか精度も 他のコマンドより良いみたいで bool や iszero だと間違うときも testeq は正解を出してきます.

- bool(f = g) >> $FALSE$
- iszero(f − g) >> $FALSE$

注10) testeq は Simplify と同じく，Steps= n の option が使える．その他 testeq には, Seconds= t,RuleBase= $base$ の option がある．

testeq を使った方が, 安全です．iszero, bool はスピードが速いので プログラミングの中で使います.（その場合でも,Manual では iszero の方を使うことを薦めています.）

なお，合成公式の問題は，expand を使っても，解決できます.

- expand(sqrt(2) ∗ sin(x + PI/4))　　>> cos(*x*) + sin(*x*)

このように，合成した式を，展開することは出来ます．しかし, 逆は出来ません.

- combine(%, sincos)　　>> cos(*x*) + sin(*x*)

4.4 変数の範囲の指定 (assume の利用)

いくつかの変形は，適当な範囲を指定しないとうまく行きません.

4.4.1 簡単にならない式

[x の絶対値]

abs(x) = |x|(x の絶対値) は,x と原点からの距離を表すので,

x が実数のとき,

$$|x| = \begin{cases} x & (x \geqq 0) \\ -x & (x < 0) \end{cases}$$

しかし, 上の公式は x が複素数のときは, 成り立ちません．一般には，$x = a + bi(a, b$ 実数) とすると,

$$|x| = \sqrt{a^2 + b^2}$$

[x^2 の平方根]

同様のことは， $\sqrt{x^2}$ にも成り立ちます.

$$\sqrt{3^2} = \sqrt{9} = 3,\ \sqrt{(-3)^2} = \sqrt{9} = 3 = -(-3),\ \sqrt{(-4)^2} = \sqrt{16} = 4 = -(-4), \cdots$$

ですから，x が実数のとき,

$$\sqrt{x^2} = |x| = \begin{cases} x & (x \geqq 0) \\ -x & (x < 0) \end{cases}$$

[assume の利用]

MuPAD は， default では, 変数を複素数とみなすので abs(x) や sqrt(x^2) は，全く，簡単になりません. このような時，assume を使って x の範囲を指定します.

$\|x\|$?	• abs(x)	>> $\|x\|$
$\sqrt{x^2}$は?	• sqrt(x^2)	>> $\sqrt{x^2}$
$x \geqq 0$ と仮定	• assume(x >= 0)	>> $[0, \infty)$

assume(x >= 0) で,「$x \geqq 0$」と設定しました.「$(0, \infty)$」は，「$0 < x < \infty$」の意味です.

このとき, $\|x\|$?	• abs(x)	>> x
このとき, $\sqrt{x^2}$は?	• sqrt(x^2)	>> x
$x < 0$ と仮定	• assume(x < 0)	>> $(-\infty, 0)$

assume(x < 0) で,「$x < 0$」と設定しました.「$(-\infty, 0)$」は,「$-\infty < x < 0$」の意味です.

このとき, $\lvert x\rvert$?	• abs(x)	>> $-x$
このとき, $\sqrt{x^2}$は?	• sqrt(x^2)	>> $-x$

このように x の範囲を指定しないと簡単になりません．また，assume の設定は，改めて assume または，delete するまで有効です.

4.4.2　成立しない公式

$\log_a x + \log_a y = \log_a xy$ でない?

x,y が複素数のとき，$\log xy$ と $(\log x + \log y)$ には $2\pi ni$(n は整数) の違いがあっても良いので

$$\begin{cases} \log xy = \log x + \log y \\ \log x^p = p \log x \end{cases} \quad \cdots(*)$$

の公式は成り立ちません．したがって，x, y の範囲を適当に指定しないと, 例えば,

$$\log x + \log \frac{e}{x} = \log\left(x \cdot \frac{e}{x}\right) = \log e = 1$$

とは**なりません**.[注11)]

$expr := \log x + \log \frac{e}{x}$と定義	• expr := ln(x) + ln(E/x)	>> $\ln\left(\frac{e}{x}\right) + \ln(x)$
1 つにまとめると?	• combine(expr, ln)	>> $\ln\left(\frac{e}{x}\right) + \ln(x)$

x を正の数に指定してから, もう一度やってみます.

x を正の数に指定	• assume(x > 0)	>> $(0, \infty)$
1 つにまとめると?	• combine(expr, ln)	>> 1

注11) 一般に z が複素数で,$z \neq 0$ のとき

$$\log z = \log|z| + i \arg(z) \quad \cdots(**)$$

(ただし $\arg z$ は $2\pi i$ の整数倍を加えてよく，かつ $\log|z|$ は, 通常の, 実数の対数関数)

例えば，n を整数とすると

$$\begin{cases} \log(-1) & = \log|-1| + i\arg(-1) = \log 1 + \pi i + 2\pi ni = (2n+1)\pi i \\ \log(i) & = \log|i| + i\arg(i) = \log 1 + \frac{\pi}{2} i + 2\pi ni = \left(2n + \frac{1}{2}\right)\pi i \\ \log(1+i) & = \log|1+i| + i\arg(1+i) = \log\sqrt{2} + \frac{\pi}{4} i + 2\pi ni = \log\sqrt{2} + \left(2n + \frac{1}{4}\right)\pi i \end{cases}$$

実際には，MuPAD では, 偏角 $\arg z$ の範囲を, $-\pi < \arg z \leqq \pi$ にとるので, 次のようになります.

•ln(1); ln(−1); ln(I); ln(−I)　　>> 0　>> $i \cdot \pi$　>> $\frac{i}{2} \cdot \pi$　>> $\left(-\frac{i}{2}\right) \cdot \pi$

したがって,

$$\begin{cases} \log(-1) + \log(-1) & = \pi i + \pi i = 2\pi i & \neq 0 = \log 1 = \log(-1)^2 \\ 4\log(-1) & = 4 \times \pi i = 4\pi i & \neq 0 = \log 1 = \log(-1)^4 \end{cases}$$

すなわち，$(*)$ の公式は成り立ちません．($x = y = -1, p = 4$ の場合)

このように x, y が正の数で, p が実数のときは, MuPAD は ,$\log x$ を実数値関数と考えて計算します．すなわち,$(*)$ の公式が成り立ちます.
逆に言うと,「x, y が正の数で, p が実数」と指定しないと $(*)$ は成り立ちません．その指定には，assume を使います．MuPAD で「x, y が正の数で, p が実数」と指定して, 確かめてみます.
今度は，assume(x, Type :: $\cdots$) を使って，設定してみます．assume(x, Type :: Positive) で正の数, assume(x, Type :: Real) で実数に設定できます．[注12)]

- assume(x, Type :: Positive) >> $(0, \infty)$
- assume(y, Type :: Positive) >> $(0, \infty)$
- assume(p, Type :: Real) >> $\mathbb{R}$

これで，x, y を正の数，p を実数と設定できています．なお，$\mathbb{R}$ は 実数集合 の意味です．これで, 展開してみます.

- expand(ln(x $*$ y)) >> $\ln(x) + \ln(y)$
- expand(ln(x^p)) >> $p \cdot \ln(x)$

確かに, 公式は成り立ちました．ところが,x, y, p が複素数のときは，これは成り立ちません．x, y, p の設定を消去してやってみます.

- delete x, y, p >>
- expand(ln(x $*$ y)) >> $\ln(x \cdot y)$
- expand(ln(x^p)) >> $\ln(x^p)$

このように**変数の範囲をうまく設定しないと, 多くの指数 · 対数, 三角関数の公式は成り立ちません．**
assume の詳しい使い方については，?assume, または，第 7 章をご覧ください.

[注12)] もちろん，assume(x, Type :: Positive) の代わりに，assume(x, x > 0) でも同じです.

第5章　関数の定義,文字式への代入

(式) に y という名前をつける	y := (式)
x の関数 y を定義する	y := x -> f(x)
x, y の関数 z を定義する	z := (x, y) -> f(x, y)
x の継ぎはぎ関数 f を定義する	f := x -> piecewise([condition$_1$, object$_1$], $\cdots$)
g と f の合成関数 : $f \circ g$ を, h と定義する	h := f@g
f を n 回合成した関数 : $f \circ f \cdots \circ f$ を, h と定義する	h := f@@n
f において, Old を New で置換 (主に，変数の置換)	subs(f, Old = New)
Old$_1$, Old$_2$, $\cdots$ を, New$_1$, New$_2$, $\cdots$ で,**順に置換**	subs(f, Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$)
Old$_1$, Old$_2$, $\cdots$ を, New$_1$, New$_2$, $\cdots$ で,**同時に置換**	subs(f, [Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$])
f において, Old を New で置換 (主に，式の置換)	subsex(f, Old = New)
Old$_1$, Old$_2$, $\cdots$ を, New$_1$, New$_2$, $\cdots$ で,**順に置換**	subsex(f, Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$)
Old$_1$, Old$_2$, $\cdots$ を, New$_1$, New$_2$, $\cdots$ で,**同時に置換**	subsex(f, [Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$])
$i_1, i_2, \cdots$ 番目の *operand* を, New$_1$, New$_2$, $\cdots$ で**順に置換**	subsop(f, i$_1$ = New$_1$, i$_2$ = New$_2$, $\cdots$)
f を評価 (*evaluate*)	eval(f)

注1)

5.1　式や関数の定義の仕方

5.1.1　式 (expression) に名前をつける.

第 3 章でも, 何度も使っていますが，式に名前をつけるのは':=' を使います．例えば x+y に f という名前をつけたいならば，f := x+y とします．式に名前をつけるのは，同じ式を繰り返し用いたいときです．例えば，$\frac{3x+x^2-4}{2x^2-4x+2}$ を normal と, simplify の両方を使って約分してみて，その動作を比べたいとします．このような時は適当な名前[注2)] を使って，次のようにすると楽です．

$\frac{3x+x^2-4}{2x^2-4x+2}$ に bunsu という名前をつける．

- bunsu := (3 * x + x^2 - 4)/(2 * x^2 - 4 * x + 2)　　>> $\dfrac{3x+x^2-4}{2x^2-4x+2}$

注1) subs(f, [Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$]) は，subs(f, {Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$}) でも大丈夫です．同様，subsex(f, [Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$]) は，subsex(f, {Old$_1$ = New$_1$, Old$_2$ = New$_2$, $\cdots$}) でも大丈夫です．
subs は substitute(代入する) の略です．eval は evaluate(評価する) の略です．
subsex は，substitute expression の略です．subsop は，substitute operand の略です．

注2) MuPAD により予約されている名前 (PI,simplify など) と，最初に数字が来る名前 (1x など),「_」以外の記号などを使った名前は定義できません．数字が後に来る場合 (x1 など) は大丈夫です．

式の名前は出力されません．これを normal で約分してみましょう．

- normal(bunsu)　　>> $\dfrac{x+4}{2x-2}$

今度は simplify() を使って簡単にしてみましょう．

- simplify(bunsu)　　>> $\dfrac{x+4}{2x-2}$

二つは，同じ結果となりました．この場合は，simplify と normal はどちらも同じような働きをする事がわかりました．もしここで simplify(%) では (直前の式)= $\frac{x+4}{2x-2}$ を simplify する事になりますので，比較したことにはなりません．(simplify(%2) とすれば大丈夫ですが $\cdots$．)

5.1.2　関数 (function) に名前をつける．

関数に名前をつけて，定義しておくと，いろいろな数字を代入したい時に便利です．関数の定義は矢印 '− >' を使います．[注3)] 例えば $x \mapsto x^2$ という関数を nijo という名前を付けて定義したいならば

- nijo := x −> x^2　　>> $x \longrightarrow x^2$

2 つ以上の変数のときも同様です．x と y の関数: $(x,y) \mapsto \frac{x+y}{2}$ を heikin と名をつけて定義するのは，次のようにします．

- heikin := (x, y) −> (x + y)/2　　>> $(x,y) \longrightarrow \dfrac{x+y}{2}$

5.1.3　関数と数式の定義の違い

MuPAD では, 関数 (function) と数式 (expression) を, 内部では, はっきりと区別しています．MuPAD の内部のデータ型のことを, ドメインタイプ (domain type) といいます．domain type は，コマンド domtype で見ることが出来ます．(詳しくは次章) 例えば,

- domtype(3); domtype(2/3)　　>> DOM_INT　>> DOM_RAT
- domtype(0.5); domtype(2 + 3 ∗ I)　　>> DOM_FLOAT　>> DOM_COMPLEX

「3」の domain type は，DOM_INT(整数),「$\frac{2}{3}$」の domain type は，DOM_RAT(有理数),「0.5」の domain type は，DOM_FLOAT(浮動小数）,「$2+3i$」の domain type は，DOM_COMPLEX(複素数) となります．

- domtype(x + 3)　　>> DOM_EXPR
- delete y : domtype(y)　　>> DOM_IDENT

「x+3」などの文字式は, DOM_EXPR(expression),「y」などのように値がない文字は，DOM_IDENT(identifier) となります．

- y := x + 3 : domtype(y)　　>> DOM_EXPR

注3) 関数の定義で'− >' は,(マイナス記号)+(不等号) で，矢印 (⟶) を表しています．例えば，'$x \longrightarrow x^2$' で，'x に x^2 を対応させる関数' という感じです．

しかし，値が与えられた文字は，その値の domain type を持ちます．(上の場合は，$x+3$ の domain type である DOM_EXPR になっています.)

以上は準備でした．ここで，関数の domain type を見てみます.

- `f := x -> x^2` >> x^2
- `domtype(f)` >> DOM_PROC
- `domtype(f(x))` >> DOM_EXPR

f は関数なので，その domain type は，DOM_PROC (procedure-手続き，関数の意味) になりますが，$f(x)$ の domain type は，DOM_EXPR になっています．つまり，MuPAD では，例えば $\sin(x)$ の `sin` の部分だけが**関数**(DOM_PROC) であり，$\sin(x)$ は**数式**(DOM_EXPR) というように厳密に区別しています.

- `domtype(sin)` >> DOM_PROC
- `domtype(sin(x))` >> DOM_EXPR

すなわち, MuPAD で関数というのは,

$$x \mapsto x^2$$

のような x を x^2 に対応させる**写像 (対応関係)** だけを表しています.

5.1.4 継ぎはぎ関数の定義

$f(x) = \begin{cases} x<0 \text{ のとき} & -x \\ x \geqq 0 \text{ のとき} & x^2 \end{cases}$ のような，いわゆる継ぎはぎ関数の定義には `piecewise` を次のように使います.

$$\texttt{piecewise}([\texttt{condition}_1,\ \texttt{f}_1],[\texttt{condition}_2,\ \texttt{f}_2],\cdots)$$

ここで，$\texttt{condition}_1, \texttt{condition}_2, \cdots$ は，"$x<0,\ x>=0$" のような条件式を表し，$f_1, f_2, \cdots$ は，"2*x, x^2" のような式を表します.

- `f := x -> piecewise([x < 0, -x], [0 <= x, x^2])` >> $x \to \text{piecewise}([x<0, -x], [0 \leqq x, x^2])$

代入もできます.

- `f(-5); f(4)` >> 5 >> 16

矢印 (- >) を忘れると, 関数でなく, 式になるので代入できません.

- `f := piecewise([x < 0, -x], [0 <= x, x^2])` >> $\begin{cases} -x & if \quad x<0 \\ x^2 & if \quad 0 \leqq x \end{cases}$
- `f(-5)` >> $\begin{cases} -x & if \quad x<0 \\ x^2 & if \quad 0 \leqq x \end{cases}$

5.1.5　合成関数の定義

g と f の合成関数: $f \circ g$ は，`f@g` と定義します．また，f を n 回合成した関数: $\underbrace{f \circ f \circ \cdots \circ f}_{n\text{ 個}}$ は，`f@@n` と定義します．

- `f := x -> x^2`　　>> $x \longrightarrow x^2$
- `g := x -> x + 2`　　>> $x \longrightarrow x + 2$
- `p := f@g`　　>> $(x \longrightarrow x^2) \circ (x \longrightarrow x + 2)$
- `p(x)`　　>> $(x + 2)^2$

$f \circ g(x) = f(g(x)) = f(x + 2) = (x + 2)^2$ を表しています．

- `q := g@f`　　$(x \longrightarrow x + 2) \circ (x \longrightarrow x^2)$
- `q(x)`　　>> $x^2 + 2$

$g \circ f(x) = g(f(x)) = g(x^2) = x^2 + 2$ を表しています．

f や g は，MuPAD の関数でも大丈夫です．

- `r := f@sin`　　>> $(x \longrightarrow x^2) \circ \sin$
- `r(x)`　　>> $\sin(x)^2$

$f \circ \sin(x) = f(\sin(x)) = \sin(x)^2$ を表しています．

何個でも合成できます．

- `h := f@f@f`　　>> $(x \longrightarrow x^2) \circ (x \longrightarrow x^2) \circ (x \longrightarrow x^2)$
- `h(x)`　　>> x^8

$f \circ f \circ f(x) = f(f(f(x))) = f(f((x^2))) = f(x^4) = (x^4)^2 = x^8$ を表しています．

同じ関数を合成するときは，`f@@n` が使えます．

- `rep := f@@3`　　>> $(x \longrightarrow x^2) \circ (x \longrightarrow x^2) \circ (x \longrightarrow x^2)$
- `rep(x)`　　>> x^8

`rep` は，`f` を 3 回合成した関数なので，`h` と一致します．

5.1.6 【参考】プログラミングを利用した関数の定義

MuPAD のプログラミング言語を用いて, 関数を定義することも出来ます．ただ，MuPAD のプログラミング言語の詳しい解説は, 長くなるのでこの本ではやりません．Tuorial (16,17,18 章) をご覧ください．
例えば，先の節で見た heikin という関数: $(x,y) \mapsto \frac{x+y}{2}$ を, プログラミングを利用して定義するには, 次のようにします.

```
•heikin := proc(x, y)
 begin
 return((x + y)/2)
 end_proc;                                  >> proc heikin(x,y) ... end
```

このように，MuPAD のプログラミング (procedure) では,「proc(x,$\cdots$) begin *Body*(本体) end_proc; 」の形になります.
同様に，piecewise を使った関数：

$$x \mapsto \begin{cases} -x & if \quad x < 0 \\ x^2 & if \quad 0 \leqq x \end{cases}$$

も，if 文 (if $\cdots$ then $\cdots$ else $\cdots$ end_if) を使って, 次のようにプログラミングできます.

```
•f := proc(x)
 begin
 if x < 0 then return(-x)
 else return(x^2)
 end_if
 end_proc;                                  >> proc f(x)...end
```

ちなみに procedure の, ドメインタイプも DOM_PROC です.

```
•domtype(f)                                 >> DOM_PROC
```

すなわち, 内部的には プログラミング言語を使って定義しても，x − > x^2 のように定義しても同じです.

5.2 代入

式への代入は，subs(f, x = a), 複数の式をまとめて代入したいときは subs(f, x = a, y = b, $\cdots$)(逐次に代入するとき), subs(f, [x = a, y = b, $\cdots$])(平行に代入するとき) を用います．逐次代入と，平行代入は普通は同じになります．[注4)]
一方，関数 f(x) で $x = a$ と代入したい時は，単に f(a) とするだけです．[注5)]

注4) subs(f, [x = a, y = b, $\cdots$]) は subs(f, {x = a, y = b, $\cdots$}) としても大丈夫です.
注5) $f(a)$ の方は，MuPAD では,「代入」とは呼んでいないようです.

5.2.1　式への代入

$\frac{x+y}{x-y}$ を, 式と考え, f と名前をつけて，いろいろな値を代入してみましょう.

- `f := (x + y)/(x - y);` >> $\frac{x+y}{x-y}$

まず x=1 を代入してみましょう.

- `subs(f, x = 1)` >> $\frac{1+y}{1-y}$

次は x= 0 を代入してみましょう.

- `subs(f, x = 0)` >> -1

$x = 0$ のとき, $\frac{x+y}{x-y} = \frac{y}{-y} = -1$ ですから合っています．次は,x=1,y=2 を代入してみます.

- `subs(f, x = 1, y = 2)` >> -3

文字式を代入することも出来ます．x に a^2, y に a を代入してみましょう.

- `subs(f, x = a^2, y = a)` >> $\frac{a+a^2}{a^2-a}$

「$x = 0$」や「$x = 1, y = 2$」 のとき，f が簡単になったのは MuPAD の Kernel で行っている簡略化 (internal simplication) のためです．**しかし，一般には，式に代入しても, 結果を評価 (evaluate) しないので簡単になりません．** そのような時は，`eval` を使って，MuPAD に値を評価 (evaluate) させます.

- `expr1 := log(2, x); subs(expr1, x = 2)` >> $\log_2(x)$ >> $\log_2(2)$
- `expr2 := sin(2 * x); subs(expr2, x = PI)` >> $\sin(2 * x)$ >> $\sin(2 \cdot \pi)$

このように結果を評価しません．そこで，`eval` を使って，結果を評価します.

- `eval(expr1); eval(expr2)` >> 1 >> 0

これは，$\log_2 2 = 1$, $\sin 2\pi = 0$ を表しています.

5.2.2　関数への代入

これに対し，関数として名前をつけると，代入は普通のコマンドとまったく同様にできます．[注6)] 先ほど定義した nijo := x -> x^2; と heikin := (x,y) -> (x+y)/2; を使ってやってみましょう.

- `nijo(4)` >> 16
- `heikin(4, 8)` >> 6

[注6)] MuPAD では, 関数への代入は，代入 (substitution) とは呼んでいない.

$4^2 = 16, \frac{4+8}{2} = 6$ です．文字式を代入することも出来ます．

- heikin(2 ∗ a + 3, 6 ∗ a + 1) >> $4a + 2$

確かに，$\frac{(2a+3)+(6a+1)}{2} = 4a + 2$ です

また，関数の場合は, 自動的に評価 (evaluate) されます．

- func := x− > log(2, x) >> $x-\ > \log_2(x)$
- func(2) >> 1

5.3 関数にするか，式にするか?

5.3.1 選択の基準

コマンドによっては, どちらか片方しか受け付けないもの (normal, factor などは，DOM_EXPR のみ. select, split などは DOM_PROC のみ) もありますし，関数と文字式でコマンドが微妙に違うもの (diff と D など)，どちらも受け付けるもの (plotfunc2d, plotfunc3d など) もあります．私見では，ほとんどの場合は 文字式 (DOM_EXPR) にすればよいと思います．ただ，頻繁に代入したり，非常に複雑でプログラミングを利用したい場合は, 関数 (DOM_PROC) の方が良いでしょう．

[式として定義しないといけない場合]

まず，絶対に式として定義しないといけない場合を見てみます．関数には factor, normal 等の変形はできません．それに diff, int などもできません．

- kansu := (x, y) − > (x ∗ y + x^2)/x >> $\frac{x \cdot y + x^2}{x}$
- normal(kansu) >> Error: expecting an arithmetical expression [normal]
- diff(kansu, x) >> Error: Illegal argument [diff]

どうしても kansu に normal や diff を働かせたいときは，次のように，式に直してから変形すると良いでしょう．

- g := kansu(x, y) >> $\frac{x \cdot y + x^2}{x}$
- normal(g) >> $x + y$

前節で述べたように，f は関数 (function) ですが，g は関数ではなく，式 (expression) になっています．すなわち, **関数を式に直すのは，とても簡単です．**

[式と関数で異なるコマンドを使う場合]

関数と式で, 同じような操作なのに，違なるコマンドを使う例も見てみましょう．式の微分には `diff` を使いましたが，関数の微分には `D` を使います.

- `diff(sin(x), x)` >> $\cos(x)$
- `D(sin)` >> cos
- `f := x -> x^2; D(f)` >> $x \longrightarrow x^2$ >> $x \longrightarrow 2 \cdot x$
- `diff(f(x), x)` >> $2 \cdot x$

[式としても，関数としても同じコマンドが使える場合]

この他, どちらも受け付けるものもあります．`plotfunc2d` は, 平面のグラフを描くコマンドですが，どちらも受け付けます.

- `plotfunc2d(sin(x), x = 0..PI)` >> グラフ (省略)
- `plotfunc2d(sin, x = 0..PI)` >> グラフ (省略)

どちらでも, 同じグラフとなります.

5.3.2　式と関数の変換

関数を式に直すのは, 前節でも見たようにとても簡単ですが，式を関数にするのは思ったほど簡単ではありません.

- `g := x^2` >> x^2
- `f := x-> g` $x \longrightarrow g$

しかし, これは x の関数になっていません.

- `f(3); f(4)` >> x^2 >> x^2

$f(x)$ の x と，x^2 の x は，MuPAD には無関係と見えている様です．こういうときは，fp lib(library for functional programming) の `fp :: unapply` を使います.

- `myfunc := fp :: unapply(g, x)` >> $x \longrightarrow x^2$
- `myfunc(3); myfunc(4)` >> 9 >> 16

このように，fp :: unapply(expr, x, y, ···) を使うと，式 (*expression*) を，$x, y, \cdots$ の関数 (function) に直すことができます.

- ex := x^2 + y　　>> $x^2 + y$
- myfunc := fp :: unapply(ex, x, y)　　>> $(x, y) \longrightarrow y + x^2$

このとき，x, y は省略できますが, 順序は，MuPAD 任せです.

- myfunc := fp :: unapply(ex)　　>> $(y, x) \longrightarrow y + x^2$

5.4 高度な代入 (substitution)

5.4.1 逐次代入 (sequential substitution) と平行代入 (parallel substitution)

2 つ以上の変数への代入には，2 種類あります．subs(f, x = a, y = b, ···) は $x = a, y = b, \cdots$ を順番に代入します．これに対し subs(f, [x = a, y = b, ···]) は $x = a, y = b, \cdots$ を平行に (同時に) 代入します．[注7)] 次のような場合は，2 種類の代入で結果が異なります.

- ex := x^2 + y　　>> $x^2 + y$
- subs(ex, x = y, y = x)　　>> $x^2 + x$
- subs(ex, [x = y, y = x])　　>> $y^2 + x$

subs(ex, x = y, y = x) の場合はまず x に y を代入します.

$$x^2 + y \longrightarrow y^2 + y$$

次に y に x を代入します.

$$y^2 + y \longrightarrow x^2 + x$$

一方，subs(ex, [x = y, y = x]) の場合は，x に y を，y に x を，平行に代入します．すなわち x と y を入れ替えます.

$$x^2 + y \xrightarrow{x,y \text{ を交換}} y^2 + x$$

しかし, 普通の代入 (x と y を交換しない代入) では結果は同じになるでしょう.

5.4.2 式 (expression) の置換

subs は変数のみならず，式を置き換えることもできます．このときは subs よりも subsex を使ったほうが良いことが多いです.

注7) subs(f, [x = a, y = b, ···]) の代わりに subs(f, {x = a, y = b, ···}) でも同じです.

- sum3 := x + y + z + (x + y)^2　　>> $x+y+z+(x+y)^2$
- subs(sum3, x + y = a)　　>> $x+y+z+a^2$
- subsex(sum3, x + y = a)　　>> $a+z+a^2$

このように，subs では，$(x+y)^2$ の方の $x+y$ は置換されましたが，$x+y+z$ の方の $x+y$ は置換されません．一方，subsex の方では，どちらも置換しています．すなわち, 部分的な式の置換は subs では出来ません．これに対して，subsex は, 部分的な式の置換も行います.
しかし，subsex でもうまく行かないことが，しばしばあります．例えば,

- ex := 1/(x * y)　　>> $\frac{1}{x \cdot y}$
- subs(ex, x * y = 1)　　>> $\frac{1}{x \cdot y}$

部分どころか, 式全体さえ置換されていません．これは MuPAD の内部では

$$\frac{1}{x \cdot y} \longrightarrow \frac{1}{x} \cdot \frac{1}{y}$$

と処理されていて,MuPAD が $x * y$ を見つけられないためです．よって，次のようにすればうまく行きます.

- subsex(ex, 1/x * 1/y = 1)　　>> 1

一般に，和, 差, 積の場合は, 部分式も置き換えられますが，商の場合は subsex をもってしても，置き換えるのが難しいようです．なお, subsex は subs よりずーと遅くなりますから，subs で済む置き換えに関しては，subs を使った方が良いでしょう.

subsex の他に，MuPAD の operand(MuPAD の building block のようなもの) を直接置き換えることの出来る subsop がありますが，これの理解のためには，operand に関する十分な知識が必要になり，また, 普通はそんなに使わないと思うので, ここでは省略します．なお，速度は subsex が一番遅く，subsop が一番速くなります.

第6章　MuPAD の オブジェクト (object)

6.1　まだある重要なデータ型 (sequence,list,set) の基本

オブジェクト (object) という言葉は,「Java は，オブジェクト指向言語である」などで使われる，あの「オブジェクト」ですが，いくつかのデータ（数字など) をまとめて,「1 つの物」(オブジェクト) として扱う時に使われます．そして，MuPAD では，オブジェクト は，いくつかのデータタイプに分類されていて，それを ドメインタイプ (domain type) といいます．いままで，数 (number), 文字式 (expression), 関数 (function) を扱ってきましたが，それ以外にも，重要なデータ型はあります．その中でも特に重要なのが，列 (sequence), リスト (list), 集合 (set) です．この節では，この 3 つのデータ型の基本についてのみ説明します．なお，この後の節で，さらに詳しいことも説明しますが，この本の後の章を読むには, この節の内容だけでも大丈夫です.

6.1.1　列（sequence）

`sq := a, b, c, ···`	列： *sq* の定義
`sq[i]`	*sq* の第 *i* 番目の項
`sq1 , sq2`	2 つの列を結びつける
`a(k) $k = i..j` または `_seqgen(a(k), k, i..j)`	$a(i), a(i+1), \cdots, a(j)$
`a $n` (`n` は自然数) または `_seqgen(a, n)`	$a, a, \cdots a$ (n 個の a)

列 (sequence) は，単に object を, コンマで区切って並べたものです.

- `sq := sin(x), x^2, 3/4`　　>> $\sin(x), x^2, \frac{3}{4}$

`sq` の *i* 番目の要素を取り出すには，`sq[i]` とします.

- `sq[1]; sq[2]; sq[3]`　　>> $\sin(x)$　>> x^2　>> $\frac{3}{4}$

２つの，sequence は,「,」(コンマ) で結びつけることができます.

- `sq2 := 1, 2, 3`　　>> $1, 2, 3$
- `sq, sq2`　　>> $\sin(x), x^2, \frac{3}{4}, 1, 2, 3$

「$ 」(sequence generator) で，sequence を作成することもできます.

- `k^2 $k = 2..5`　　>> $4, 9, 16, 25$

k^2 $k = 2..5 で，$2^2, 3^2, 4^2, 5^2$ と同じになります．このとき，***k^2 と$の間に, コンマを入れない*** ように注意してください．一般に,

object(k) $k = i..j　　>> object(i), object(i + 1), ··· , object(j)

となります．また，$の後に，数字だけを書くと,「繰り返し」を表します．例えば,

- object $3　　>> object, object, object
- x $5　　>> x, x, x, x, x

これは,n 階微分をするときに便利です．$f(x)$ を,x に関し 3 回微分するのは，diff(f(x), x, x, x) ですが,これを，diff(f(x), x $3) と書くことができます．例えば,

- diff(sin(x), x $3)　　>> $-\cos(x)$

確かに，$(\sin x)''' = (\cos x)'' = (-\sin x)' = -\cos x$ です.

6.1.2　リスト (list)

list := [a, b, c, ···]	リスト： *list* の定義
list := [seq]	リスト： *list* の定義 (列 : *seq* から)
list[i]	*list* の第 *i* 番目の項
list1.list2 または _concat(list1,list2)	2 つのリストを結びつける

list というのは，[ ] で囲ったデータ型で，順序が問題になるときに使います.

- list := [x, x^2, 3]　　>> $x, x^2, 3$

このとき, 入力した順番通りに出力されることに注意してください．list[i] で i 番目の要素を取り出すことができます.

- list[1]; list[2]; list[3]　　>> x >> x^2 >> 3

2 つのリストを結びつけるのは，sequence と異なり,「.」（ピリオド) を使います.

- list2 := [x, 1, 2, 3]　　>> $[x, 1, 2, 3]$
- list.list2　　>> $[x, x^2, 3, x, 1, 2, 3]$

このように, 順序を保存しています．また同じ要素があっても，ダブって表示します.

list は，グラフを描くときにも使われています．例えば，$x = \cos t, y = \sin t$ $(0 \leqq t \leqq \pi)$ の曲線は,

- plot :: Curve2d([cos(t), sin(t)], t = 0..PI)

ですが，これが {cos(*t*), sin(*t*)} だと，エラーになります．ここでは，*x* と *y* の順番が大切なので, list にしないといけません.

6.1.3 集合 (set)

`set := {a, b, c, ···}`	集合： *set* の定義
`set := [seq]`	集合： *set* の定義 (列 : *seq* から)
`set[i]`	*set* の第 *i* 番目の項 (画面表示の順序)
`set1 union set2`	*set*1 と *set*2 の和集合
`set1 intersect set2`	*set*1 と *set*2 の共通集合
`set1 minus set2`	*set*1 と *set*2 の差集合

集合 (set) は { } で囲ったデータ型で，順序は問題にしません．list と set の関係は,「順列」と¢組み合わせ£の関係に似ています.

- `set := {x, x^2, 3}` >> $\{3, x, x^2\}$

このように, 入力した順番と, 表示された順番は異なります．`set[i]` で，set の *i* 番目の要素を取り出すことができますが，これは, 入力でなく，出力画面上の順番であることに注意してください.

- `set[1]; set[2]; set[3]` >> 3 >> x >> x^2

set の場合, 重複した要素があると,MuPAD は，自動的に消去します.

- `{x, x^2, 3, x, 1, 2, 3}` >> $\{1, 2, 3, x, x^2\}$

2 つの set を「.」や「,」で結びつけるのは，できません．そのかわり，`union`, `intersect`, `minus` を使います.

`set1 union set2`(和集合)， `set1 intersect set2`(共通集合)， `set1 minus set2`(差集合)

例えば,

- `set1 := {x, x^2, 3}` >> $\{3, x, x^2\}$
- `set2 := {x, 1, 2, 3}` >> $\{1, 2, 3, x\}$
- `set1 union set2` >> $\{1, 2, 3, x, x^2\}$
- `set1 intersect set2` >> $\{3, x\}$
- `set1 minus set2` >> $\{x^2\}$
- `set2 minus set1` >> $\{1, 2\}$

set は，solve を使って, 方程式を解くときなどに使われます．例えば

• solve(x^2 − 3 ∗ x + 2, x)　　>> {1, 2}

この場合,2 つの解の順番は考えないので，集合が使われます.

6.2　ドメインタイプ (domain type)

domtype(object)	object の domain type
nops(object)	object の operand の数
op(object)	object の operand を全て取り出す
op(object, i)	object の i 番目の operand を取り出す
op(object, i..j)	object の i 番目から j 番目の operand を取り出す

この後の節では，さらに詳しく MuPAD のオブジェクト (object) について見ていきます．ここで述べた知識が無くとも，後の章は問題なく解ると思うので，必要な時に, 参照してください.
MuPAD では，オブジェクト は，いくつかのデータタイプに分類されていて，それを ドメインタイプ (domain type) といいます．object が，どのドメインに属しているかを知るには

domtype(object)

を使います.

• domtype(3); domtype(0.5)　　>> DOM_INT, >> DOM_FLOAT
• domtype(2/3); domtype(3 + I)　　>> DOM_RAT, >> DOM_COMPLEX

このように, 数 (number) には,

DOM_INT(整数), DOM_FLOAT(小数), DOM_RAT(有理数), DOM_COMPLEX(複素数)

の 4 つの domain type があります．さらに,

• domtype(x^2 + y)　　>> DOM_EXPR
• f := (x, y) −> x^2 + y　　>> $(x, y) \longrightarrow x^2 + y$
• domtype(f)　　>> DOM_PROC
• domtype(f(x, y))　　>> DOM_EXPR

式は，DOM_EXPR に，関数 f は, DOM_PROC に属します．また，f(x,y) は, 式となります．(第 5 章参照)

• domtype(a)　　>> DOM_IDENT
• a := 5 :　domtype(a)　　>> DOM_INT

DOM_IDENT というのは,「値を持っていない文字」(identifer) です．ここで，a := 5 のように値を代入すると，その値 (ここでは 5) の domain type である DOM_INT が表示されます．次は, 先ほどの 列 (sequence), リスト (list), 集合 (set) について見てみます.

- domtype(1, x, x^2) >> Error: Wrong number of arguments [domtype]

argument は引数ともよばれ，( ) の中の文字のことです.「引数の数が違っています」という意味です. domtype(object) ですから，引数の数は 1 個でないといけないみたいです.

- seq := 1, x, x^2 : domtype(seq) >> DOM_EXPR

sequence は，DOM_SEQ とかではなく (DOM_SEQ などはない) ，DOM_EXPR になります．また，list は DOM_LIST, set は DOM_SET となります.

- domtype([1, x, x^2]) >> DOM_LIST
- domtype({1, x, x^2}) >> DOM_SET

iszero(f) ($f = 0$ か?) などで出力される「TRUE,FALSE」などの domain type は，DOM_BOOL です.

- domtype(TRUE); domtype(FALSE) >> DOM_BOOL >> DOM_BOOL

ファイルを読み込む時に，read(”foo.mu”) としますが，このダブルクォーテーション (” ”) で囲まれるのは，DOM_STRING(文字列) です.

- domtype(”Hello World”) >> DOM_STRING

注1)

Tutorial に載っている主な domain type は次表のとおりです.

注1) MuPAD の DOM_EXPR は非常に広く，いわゆる文字式だけでなく，等式, 不等式なども含みます.

- domtype(x + y = 3) >> DOM_EXPR

ドメインタイプ (domain type)	意味 (meaning)
DOM_INT	整数 (integers) 例:−3, 10^5
DOM_RAT	有理数 (rational numbers) 例:7/11
DOM_FLOAT	浮動小数 (floating-point numbers) 例:0.123
DOM_COMPLEX	複素数 (complex numbers) 例:$1 + 2/3 * I$
DOM_IDENT	識別子 (symbolic identifiers) 例: x, y, f
DOM_INTERVAL	小数の区間 (floating-point intervals) 例: 2.1...3.2
DOM_EXPR	文字式 (symbolic expressions) 例: $x + y$
Series::Puiseux	級数展開 (symbolic series expansions) 例:$1 + x + x^2 + O(x^3)$
DOM_LIST	リスト (lists) 例:[1,2,3]
DOM_SET	集合 (sets) 例:{1,2,3}
DOM_ARRAY	配列 (arrays)
DOM_TABLE	表 (tables)
DOM_BOOL	真偽値 (Boolean values) TRUE,FALSE,UNKNOWN
DOM_STRING	文字列 (strings) 例:"I am a string"
Dom::Matrix(R)	R 上の行列, ベクトル (matrices and vectors over the ring R)
DOM_POLY	多項式 (polynomials) 例:$poly(x^2 + x + 1, [x])$
DOM_PROC	関数 (functions and procedures) 例:$x -> x^2$

注2)

大文字の DOM_··· のタイプは，Kernel(核) 内で，C,C++ を使って定義されている domain type です．小文字（Dom::Matrix や Series::Puiseux など) は，それぞれ，ライブラリ内で，MuPAD のプログラミング言語を使って定義されている domain type です.
注3)

6.3　オペランド (operand)

6.3.1　基本的な使い方

オペランド (operand) というのは，MuPAD の object を作っている一つ一つのブロック (building block) のようなものです.

object の i 番目の operand を取り出すには,

```
op(object,i)
```

i 番目から，j 番目の operand を取り出すこともできます.

```
op(object,i..j)
```

注2) これは，MuPAD 3.0 Tutorial にあるものを, 私が写したものです．(日本語訳は適当につけました.)
注3) MuPAD は Kernel(核) と，library の二つの部分に分かれていて，Kernel は，C,C++言語で書かれ, その中で, 入力された式の解析, 簡略化などと並び，MuPAD 独自のプログラミング言語を定義しています．このプログラミング言語を使って書かれているのが，ライブラリです．info() でその例を見ることができます．ライブラリの中のコマンドは, 普通そのライブラリ名を付けて, plot :: Curve2d のように呼び出しますが，std lib(standard library) だけは例外で，simplify のように，単独で呼び出すことができます.

object の operand を全て取り出すには,

```
op(object)
```

operand の個数を調べるには,

```
nops(object)
```

とします. (nops は, number of operands の略です.)

- `list := [A,B,C,D]` >> $[A, B, C, D]$
- `op(list,1); op(list,2); op(list,3); op(list,4)` >> A >> B >> C >> D
- `op(list,2..4)` >> B, C, D
- `op(list)` >> A , B , C, D
- `nops(list)` >> 4

list のときは, `op(list,i)` は, `list[i]` と同じになります. しかし, **set のときは, 画面上の順番とさえ異なります.**

- `set := {x,1,x^2,y}` >> $\{1, x, y, x^2\}$
- `op(set)` >> $x, 1, x^2, y$
- `op(set,1); op(set,2); op(set,3); op(set,4)` >> x >> 1 >> x^2 >> y

これに対して, `set[i]` は, 画面上の順番を守ります.

- `set[1]; set[2]; set[3]; set[4]` >> 1 >> x >> y >> x^2

文字式 (expression) に対しても, op, nops は使えます.

- `a := x + y + 1` >> $x + y + 1$
- `op(a,1); op(a,2); op(a,3)` >> x >> y >> 1
- `op(a)` >> $x, y, 1$
- `nops(a)` >> 3

分数のときは, 分子 → 分母の順に取り出されます.

- `rat := 3/5` >> $\frac{3}{5}$
- `op(rat,1); op(rat,2); op(rat)` >> 3 >> 5 >> 3, 5
- `nops(rat)` >> 2

実は,x の分子, 分母は, それぞれ `numer(x)`, `denom(x)` で取り出すこともできますが, op で取り出せば, いちいちコマンドを覚える必要がありません.

6.3.2　文字式 (expression) における，0 次オペランド (operand)

文字式 (expression) のとき，op(expression, 0) で，expression の operand を**直接**結び付けている演算子 (Kernel コマンド) が解ります.

- op(x + y); op(x + y, 0)　　>> x, y　>> _plus

x, y という operand が，_plus によって結ばれています．ここで _plus(a, b, ···) は，Kernel のコマンドで，a + b + ··· と同じ働きをします．そして，Kernel では，_plus に直して処理しています.

- info(_plus)　　>> _plus – a function of the **system kernel** [try ?_plus for help]
- _plus(3, 4, 5)　　>> 12

掛け算や累乗もやってみます.

- op(x ∗ y); op(x ∗ y, 0)　　>> x, y　>> _mult
- op(x/y); op(x/y, 0)　　>> $x, \frac{1}{y}$　>> _mult　　···①
- op(x^2); op(x^2, 0)　　>> $x, 2$　>> _power

_mult(a, b, ···) は，a ∗ b ∗ ··· と，また，_power(x, y) は，x^y と同じです．例えば,

- _mult(2, 3, x); _power(x, 3)　　>> $6 \cdot x$　>> x^3

したがって，① より，MuPAD の内部では，x/y = x × $\frac{1}{y}$ と処理されていることが解ります.

6.3.3　op を繰り返す

op(object, i) は, 何度でも入れ子にすることができます．それによって，どんどん細かく砕いて, もう細かくすることができない, いわば，**原子**まで還元することができます.

- a := 1 + 2 ∗ x ∗ y + x^2　　>> $2 \cdot x \cdot y + x^2 + 1$
- op(a, 0); op(a, 1); op(a, 2); op(a, 3)　　>> _plus　>> $2 \cdot x \cdot y$　>> x^2　>> 1

ここで，例えば，op(a, 1) をもっと砕いてみます.

- op(op(a, 1), 0); op(op(a, 1), 1); op(op(a, 1), 2); op(op(a, 1), 3)　>> _mult >> x　>> y　>> 2

これは, これ以上もう砕くことはできません.

- op(_mult); op(x); op(y); op(2)　　>> _mult　>> x　>> y　>> 2

また，op(op(a, 1), 0), op(op(a, 1), 1), op(op(a, 1), 2), op(op(a, 1), 3) などは,[ ] を用いて，次のように省略して入力する事もできます.

- op(a, [1, 0]); op(a, [1, 1]); op(a, [1, 2]); op(a, [1, 3])　　>> _mult　>> x　>> y　>> 2

6.3.4 オペランド (operand) に分解する事のメリット

オペランド (operand) に分解すると，ある部分だけ取り出して変形することができます.

solve(f, x) は, 方程式 $f = 0$ を，x に関して解きますが，通常は 3 次方程式の解の公式は使いません．しかし，*option* として，MaxDegree= 3 と指定すると 3 次方程式の解の公式を用いて解きます.

- ans := solve(x^3 − 2 ∗ x − 3, x, MaxDegree = 3)　　>> 超長〜い解

しかし，解は長すぎて，MuPAD Pro の通常の 出力さえできません．このようなとき，op(ans, i) を用いて，i 番目の解を取り出すことができます.

- op(ans, 1)　　$>> \dfrac{2}{3 \cdot \sqrt[3]{\frac{\sqrt{211}\cdot\sqrt{108}}{108} + \frac{2}{3}}} + \sqrt[3]{\dfrac{\sqrt{211}\cdot\sqrt{108}}{108} + \dfrac{2}{3}}$

これだけならば，ans[1] でも出来ますが，op ならば，さらに細かく分解して, 見ることができます.

- op(op(ans, 1), 2)　　$>> \sqrt[3]{\dfrac{\sqrt{211}\cdot\sqrt{108}}{108} + \dfrac{2}{3}}$
- op(op(op(ans, 1), 2), 1)　　$>> \dfrac{\sqrt{211}\cdot\sqrt{108}}{108} + \dfrac{2}{3}$
- float(%)　　>> 2.897749514

MuPAD を 対話的 (interactive) に使っている場合は，面倒ですが,「超長〜い解」を自分で入力することも出来ます．しかし，MuPAD でプログラムするときなどには, そういうことは出来ないので，よく使われます. (行列の章参照)

6.4 列, リスト, 集合のやや高度なコマンド

map(object, f)	f を object に次々作用させる
map(object, f, $p_1, p_2, \cdots$)	f を object に次々作用させる ($p_1, \cdots$ は，f のパラメータ)
select(object, f)	object のうち，f が TRUE(真) を返すものを選択
select(object, f, $p_1, p_2, \cdots$)	object のうち，f が TRUE(真) を返すものを選択 ($p_1, p_2 \cdots$は，f のパラメータ)
split(object, f)	object を，f が TRUE,FALSE,UNKNOWN を返すものに分別
select(object, f, $p_1, p_2, \cdots$)	object を，f が TRUE,FALSE,UNKNOWN を返すものに分別 ($p_1, p_2 \cdots$は，f のパラメータ)

この際ですので，少し進んだコマンドも見ておきます.

6.4.1 map

map(object, f) は，object が 列，リスト, 集合などのとき，f を順番に，その要素に作用させます.

注4)

- `map({x, y, z}, f)` >> $\{f(x), f(y), f(z)\}$

このように，$[x, y, z]$ をまとめて，f に代入します.

- `map([0, PI/4, PI/2], sin)` >> $\left[0, \frac{\sqrt{2}}{2}, 1\right]$

$\left[\sin 0, \sin\frac{\pi}{4}, \sin\frac{\pi}{2}\right]$ を表しています.

- `list := [sqrt(3)/(sqrt(2) + 1), (sqrt(3) + sqrt(2))/(sqrt(3) - sqrt(2))]`
  >> $\left[\frac{\sqrt{3}}{\sqrt{2}+1}, -\frac{\sqrt{2}+\sqrt{3}}{\sqrt{2}-\sqrt{3}}\right]$

map を使って，radsimp(無理数の簡略化), combine(指数をまとめる) を list の要素全てに働かせて見ます.

- `map(list, radsimp)` >> $\sqrt{3}\cdot\sqrt{2}-\sqrt{3},\ 2\cdot\sqrt{3}\cdot\sqrt{2}+5$
- `map(list, combine)` >> $\sqrt{6}-\sqrt{3},\ 2\cdot\sqrt{6}+5$

これで，[radsimp(list[1]), radsimp(list[2])], と [combine(list[1]), combine(list[2])] を表しています.

先の ans の全ての小数近似値を，一度に求めることも出来ます. 注5)

- `ans := solve(x^3 - 2 * x - 3, x, MaxDegree = 3) :`　- `map(ans, float)`
  >> $\{1.893289196, -0.9466445982 - 0.8297035529I, -0.9466445982 + 0.8297035529I\}$

map(ans, float) で，{float(ans[1]), float(ans[2]), float(ans[3])} と同じになります.

f がオプション (option) やパラメータ (parameter) を持つときは，それを f の後に, コンマで区切って入れます.

- `map([x, y, z], f, p)` >> $[f(x, p), f(y, p), f(z, p)]$
- `map([x, y, z], f, p, q)` >> $[f(x, p, q), f(y, p, q), f(z, p, q)]$

$x^2 - 6x + 8 = 0$ と $8x^2 - 6x + 1 = 0$ の二つの方程式を，x について解いてみます. もちろん，別々に，solve(x^2 − 6 ∗ x + 8, x); solve(8 ∗ x^2 − 6 ∗ x + 1, x) でも良いですが，次のように 1 つにまとめることが出来ます.

- `map([x^2 - 6 * x + 8, 8 * x^2 - 6 * x + 1], solve, x)` >> $\left[\{2, 4\}, \left\{\frac{1}{4}, \frac{1}{2}\right\}\right]$

注4) map は，列, リスト, 集合以外にも使えますが，働き方が直感的でありません.
注5) 実は，MuPAD 3.0 からは, 簡単に float(ans) でも求まるようになりましたが... (^_^;)

($ax^2+bx+c=0$ の解と, $cx^2+bx+a=0$ の解は互いに逆数になっていることが解ります.) [注6)]
f は, Kernel のコマンドでも大丈夫です. Kernel では, _plus(a,b) で a+b を, _mult(a,b) で a*b を, _power(a,n) で a^n を, 表しますから,

- map([3, x, x^2], _plus, 2) >> [5, $x+2$, x^2+2]
- map([3, x, x^2], _mult, 2) >> [6, $2\cdot x$, $2\cdot x^2$]
- map([3, x, x^2], _power, 3) >> [27, x^3, x^6]

diff(f,x) で x に関する微分を表します. 例えば,

- diff(x^2,x); diff(x^3,x) >> $2\cdot x$ >> $3\cdot x^2$

これをまとめて, map で行うことが出来ます.

- diff([x,x^2,x^3],diff,x) $[1, 2\cdot x, 3\cdot x^2]$

パラメータの個数は何個でも同じです. 例えば, igcd(a,b,$\cdots$) で, 整数 $a, b, \cdots$ の最大公約数を求められます.

- igcd(2,8,12,16); igcd(3,8,12,16); igcd(4,8,12,16) >> 2 >> 1 >> 4

これをまとめてすると, 次のようになります.

- map([2,3,4],igcd,8,12,16) >> [2,1,4]

6.4.2 select,split

select(object,f), split(object,f) は, f が真偽値 (TRUE,FALSE) を返す関数で, object が列, リスト, 集合などのとき, object を選択 (select) したり, 分類 (split) します. [注7)]
map(object,f) で, f が 真偽値 (TRUE,FALSE) を返す場合を考えて見ます. 例えば, isprime(n) は, n が素数なら, TRUE を, n が素数でないなら, FALSE を返します. 例えば,

- isprime(3); isprime(4) >> TRUE >> FALSE

これを使って, mapping してみます.

- map([1,2,3,4,5,6,7,8,9,10],isprime)
  >> FALSE,TRUE,TRUE,FALSE,TRUE,FALSE,TRUE,FALSE,FALSE,FALSE

注6) $c\neq 0$ のとき,

$$ax^2+bx+c=0 \Longleftrightarrow a+b\left(\frac{1}{x}\right)+c\left(\frac{1}{x}\right)^2=0 \Longleftrightarrow c\left(\frac{1}{x}\right)^2+b\left(\frac{1}{x}\right)+a=0$$

よって, $\frac{1}{x}$ は, $cx^2+bx+a=0$ の解となる.

注7) 他に, object は, expression でも大丈夫です. しかし, やや解りづらいです.

列生成子 (sequence generator)「$」 を使って, 次のようにしても同じです.

- map([k $k = 1..10], isprime)

  >> FALSE,TRUE,TRUE,FALSE,TRUE,FALSE,TRUE,FALSE,FALSE,FALSE

ここで，リスト $[1, 2, 3, \cdots]$ から，TRUE の値だけを返す要素を取り出すことができます．それには，select(object, f) を使います．これで，object から，f が TRUE の値を返すものだけを選びます.

- select([k $k = 1..10], isprime)　　>> [2, 3, 5, 7]

さらに，リストを，f が「TRUE の答えを返すもの」,「FALSE の答えを返すもの」,「UNKNOWN の答えを返すもの」の 3 種類に分けることも出来ます．それには，split(object, f) を使います.

- select([k $k = 1..10], isprime)　　>> [[2, 3, 5, 7], [1, 4, 6, 8, 9, 10], [ ]]

順に，[TRUE の答えを返すもの],[FALSE の答えを返すもの]，[UNKNOWN の答えを返すもの] となっています.

注8)

注8) select, split を使うとき，適当な 関数 f がないときは，MuPAD でプログラミングすればよいです．このとき，FALSE か，TRUE か，UNKNOWN が返ってくるようにプログラミングします．例えば，リストから「3 で割った余りが 1」になるものだけを選択したいとします．このときは, 次のようにプログラミングします.

```
•myselect := proc(n)
 begin
 if _mod(n, 3) = 1 then return(TRUE)
 else return(FALSE)
 end_if
 end_proc;
      >> proc myselect(n)...end
```

これで,「n を 3 で割った余りが 1」のときだけ，myselect(n) は TRUE を返します.

- myselect(5); myselect(6); myselect(7)　　>> FALSE　>> FALSE　>> TRUE

これを使えば，リストから,「3 で割った余りが 1」になるものだけを選択できます.

- select([k $k = 1..10], myselect)　　>> [1, 4, 7, 10]

この場合は，実は，MuPAD のコマンド is を使っても出来ます．(is については，次章参照) is(n, Type :: Residue(1, 3)) で,「n を 3 で割った余りが 1」になるときだけ TRUE を返します.

- is(6, Type :: Residue(1, 3)); is(7, Type :: Residue(1, 3))　　>> FALSE　>> TRUE

したがって次のようにしても大丈夫です.

- select([k $k = 1..10], is, Type :: Residue(1, 3))　　>> [1, 4, 7, 10]

6.4.3 リスト (list), 集合 (set) の，その他のコマンド (contains,has,zip,append,sort)

contains(list, object)	object の list における順番 (index) を返す. list が object を含んでいないときは「0」を返す.
contains(set, object)	set が，object を含んでいれば *TRUE*, 含んでなければ *FALSE* を返す
has(list, object), has(set, object)	list,set の operands が object を含んでいれば *TRUE*, 含んでなければ *FALSE* を返す.
zip(list1, list2, f)	f(list1[i], list2[i]) を要素にするリストを，新たに作る
append(list, $obj_1, obj_2, \cdots$)	list に, $obj_1, obj_2, \cdots$の要素を追加する
sort(list)	要素を，小さい順 (文字列はアルファベット順) に並べ替える

[contains]

contains の例を挙げます.

- list := [x^2, 3 * y, x + y + z, x^4] >> $[x^2, 3 \cdot y, x + y + z, x^4]$
- set := {x^2, 3 * y, x + y + z, x^4} >> $\{3 \cdot y, x^2, x^4, x + y + z\}$
- contains(list, 3 * y); contains(list, x + y + z) >> 2 >> 3

$3 * y$, $x + y + z$ は，それぞれ list の 2 番目,3 番目の要素なので,「>> 2 >> 3」となります．list の場合, 要素の中に無いときは，¢0£を返します.

- contains(list, x); contains(list, x + y) >> 0 >> 0

一方，set の場合は，*TRUE* か *FALSE* を返します.

- contains(set, x + y + z); contains(set, x) >> *TRUE* >> *FALSE*

[has]

次に，has をみます.

- has(list, x + y + z); has(list, x); has(list, 4) >> *TRUE* >> *TRUE* >> *TRUE*

has(list, x + y + z) が *TRUE* となるのは，contains と同様ですが，has(list,x) や has(list, 4) も *TRUE* となっています．これは，has が， list の 全ての operands をチェックしているからです.

- op(list, 1); op(%, 1) >> x^2 >> x
- op(list, 4); op(%, 2) >> x^4 >> 4

このように，x, 4 は，list の operand の 1 つになっています．さらに,0 次オペランドさえチェックします.

- has(list, _mult) >> *TRUE*
- op(list, 2); op(%, 0) >> $3 \cdot y$ >> _mult

確かに，list は _mult を operand に持っています．しかし，has(list,x + y) は，*FALSE* となります．

- has(list,x + y)　　>> *FALSE*

これは，list は，x + y を，operand に持たないからです．

- term := op(list, 3)　　>> x, y, z
- op(term, 1); op(term, 2); op(term, 3)　　>> x　>> y　>> z

[zip]

zip は 2 つのリストをつなぐとき使います．[注9)]

- zip([a, b, c], [1, 2, 3], _plus)　　>> $[a+1, b+2, c+3]$

これは，[_plus(a, 1), _plus(b, 2), _plus(c, 3)] と同じです．

- zip([a, b, c], [1, 2, 3], _mult)　　>> $[a, 2\cdot b, 3\cdot c]$
- zip([a, b, c], [1, 2, 3], _power)　　>> $[a, b^2, c^3]$

このように 2 つのリストを結び付けますが，map の一般化としても役立ちます．すなわち，map(list, f, a) だと，[f(list[1], a), f(list[2], a), ···] となり，f のパラメーターは一定です．このパラメータを変えたいときにも使えます．

diff(f, x) は，f を x で微分することを表します．zip を使って，x,y の両方で f を微分してみます．

- zip([x^2 * y, x^2 * y], [x, y], diff)　　>> $[2\cdot x\cdot y,\ x^2]$

[diff(x^2 * y, x), diff(x^2 * y, y)] と同じです．$\frac{d}{dx}(x^2y) = 2xy$，$\frac{d}{dy}(x^2y) = x^2$ を表しています．

6.5　その他の object(table,array,polynomial)

数，文字式，関数, リスト, 集合などの他にも，まだまだ ドメインタイプはあります．ここでは，表 (table-DOM_TABLE), 配列 (array-DOM_ARRAY) , 整式 (polynomial-DOM_POLY) について，超簡単に説明します．(実は，私は, この 3 つのドメインタイプは，ほとんど使ったことがありません．しかし，Manual を読むときには, 必要なこともあると思います．)

6.5.1　表 (table)

表 (table) は，table((インデックス) = (値), ···) のように定義できます．「インデックス (index) と値 (value) を等号で結んだ組」 を集めたものですが，index が整数でなくともよく，文字でも良い点がリストとの

[注9)] zipper(ジッパー）は，洋服などにも使われています．

違いです.

- present := table(year = 2004, month = 11, 11 = month) >>

$$\begin{bmatrix} \text{year=2004} \\ \text{month=11} \\ \text{11=month} \end{bmatrix}$$

list と同じように, table[index] で, その index の値 (value) にアクセスできます.

- present[year]; present[month]; present[11] >> 2004 >> 11 >> month

6.5.2 配列 (array)

配列は, array を使って定義します. 配列の index は自然数です. 文字の index は使えません.

- a := array(1..2, 1..3) >>

```
+   -                              -   +
|       ?[1,1]   ?[1,2]   ?[1,3]      |
|                                      |
|       ?[2,1]   ?[2,2]   ?[2,3]      |
+   -                              -   +
```

array(1..2, 1..3) で, 2×3 型行列のようなものが定義されました. これが, array です. ?[1,1] とあるのは, [1,1] 成分がまだ定まっていないという意味です. これは, 後から定義することも出来ます.

- a[1, 1] = x; a[1, 2] := 1, a[1, 3] := [1, 2, 3] >> x >> 1 >> [1, 2, 3]
- a >>

$$\begin{pmatrix} x & 1 & [1,2,3] \\ a_{2,1} & a_{2,2} & a_{2,3} \end{pmatrix}$$

成分を, サイズと同時に定義することも出来ます. そのときは, 行列 (Dom::Matrix) と同様, 各行をリストにして, さらに外から [ ] で挟みます.

- myarray := array(1..2, 1..3, [[1, 2, 3], [4, 5, 6]]) >>

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

ただし, 行列と違い, 行の範囲と列の範囲を省略することは出来ません.

- b := array([[1, 2, 3], [4, 5, 6]]) >> Error: Array dimension not a positive integer [array]

要素をとりだすのは, 行列と同じです.

- b[1, 1]; b[1, 2]; b[1, 3]; b[2, 1] >> 1 >> 2 >> 3 >> 4

行列との最大の違いは, 和, 差, 積などが定義されていないことです.

- 2 * a >> Error: Illegal operand [_mult]

6.5.3 整式 (polynomial)

変数が x の整式は,poly(f, [x]) で作成します．例えば,

- p := poly(2 ∗ x^2 + 3 ∗ x + 1, [x]) >> $\mathrm{poly}(2\cdot x^2+3\cdot x+1,[x])$
- q := poly(3 ∗ x^2 + 3 ∗ x, [x]) >> $\mathrm{poly}(3\cdot x^2+3\cdot x,[x])$

p を，q で割った商と余りを求めるには，divide(p, q) を使います.

- divide(p, q) >> $\mathrm{poly}\left(\frac{2}{3},[x]\right),\ \mathrm{poly}(x+1,[x])$

これは，商が $\frac{2}{3}$, 余りが，$x+1$ ということを表しています.

$$\begin{array}{r} \frac{2}{3} \\ 3x^2+3x\,\overline{)\,2x^2+3x+1} \\ \underline{2x^2+2x} \\ x+1 \end{array}$$

これだけでは，普通の式 (DOM_EXPR) と変わりませんが，poly は, 係数をいろいろな環 (ring) に指定することが出来ます．例えば，Dom::Integer を付けると, 整数係数の整式に制限できます.

- pZ := poly(2 ∗ x^2 + 3 ∗ x + 1, [x], Dom :: Integer) >> $\mathrm{poly}(2\cdot x^2+3\cdot x+1,[x],\mathrm{Dom}::\mathrm{Integer})$
- qZ := poly(3 ∗ x^2 + 3 ∗ x, [x], Dom :: Integer) >> $\mathrm{poly}(3\cdot x^2+3\cdot x,[x],\mathrm{Dom}::\mathrm{Integer})$
- r := poly(1/2 ∗ x^2, [x], Dom :: Integer) >> FAIL

この通り，$1/2*x^2$ の係数は整数でないので，Dom::Integer を付けると定義できません．今度は，pZ を qZ で割ってみます.

- divide(pZ, qZ) >> FAIL

商が整数係数の整式にならないので，先ほどと違い，割ることは出来ません．係数が整数になるときは, 割ることが出来ます.

- rZ := poly(x^2 + 3 ∗ x, [x], Dom :: Integer) >> $\mathrm{poly}(x^2+3\cdot x,[x],\mathrm{Dom}::\mathrm{Integer})$

- divide(pZ, rZ) >> $\mathrm{poly}(2,[x],\mathrm{Dom}::\mathrm{Integer}),\ \mathrm{poly}(1-3\cdot x,[x],\mathrm{Dom}::\mathrm{Integer})$

$$\begin{array}{r} 2 \\ x^2+3x\,\overline{)\,2x^2+3x+1} \\ \underline{2x^2+6x} \\ -3x+1 \end{array}$$

Dom::Integer(整数) の他にも，Dom::Rational(有理数),Dom::Real(実数),Dom::Complex(複素数), Dom::IntegerMod(n)(n を法とする剰余系) などを使って，いろいろな種類の整式が作れます．しかし，係数が実数の整式を考えるだけならば，poly は, 余り使う必要が無いと思います.

第7章　プロパティ(property)の設定

7.1　変数のプロパティの設定は, なぜ必要か?

いくつかの変形は，適当な範囲を指定しないとうまく行きません.

7.1.1　範囲を指定しないと, 公式が成り立たない例

MuPAD は, 式の同値変形をするとき, 複素関数と見て変形するので, 多くのお馴染みの公式が成り立ちません.

- `ex := sqrt(x) * sqrt(1/x)`　　>> $\sqrt{x}\cdot\sqrt{\frac{1}{x}}$
- `combine(ex); Simplify(ex, Steps = 10000)`　　>> $\sqrt{x}\cdot\sqrt{\frac{1}{x}}$　　>> $\sqrt{x}\cdot\sqrt{\frac{1}{x}}$

$\sqrt{x}\cdot\sqrt{\frac{1}{x}} = \sqrt{x\cdot\frac{1}{x}} = \sqrt{1} = 1$ と簡単になりそうですが,「a, b が 0 以上」という条件がないと,

$$\sqrt{a}\cdot\sqrt{b} \neq \sqrt{ab}$$

となる場合があります. [注1)] したがって，これ以上は簡単に出来ません. ほかにも, `ln(x) + ln(1/x)`, `sqrt(x^2)` なども簡単に出来ません. (第 4 章参照)

7.1.2　方程式の解が，たくさん求まる場合

MuPAD では，`solve` を使って方程式を解くときにも，複素関数とみなして解くので, 複素数の解も求まることがあります. (第 8 章参照)

- `solve(exp(x) = 1, x)`　　>> $(2\cdot i)\cdot\pi k | k\ in \mathbb{Z}$

[注1)] 例えば，$a = b = -1$ のとき,

$$\sqrt{a}\cdot\sqrt{b} = \sqrt{-1}\times\sqrt{-1} = i\times i = -1$$

一方

$$\sqrt{ab} = \sqrt{(-1)\cdot(-1)} = \sqrt{1} = 1$$

すなわち,

$$\sqrt{a}\times\sqrt{b} \neq \sqrt{ab}$$

$\mathbb{Z}$ は整数の集合を表す記号なので，$x = 2\pi i\,k$(k は整数) ということです．MuPAD では，exp(x) は，x が複素数のときも定義されている (第 4 章参照) ので，無数の解が出て来てしまいました．

7.1.3　積分が出来ない場合

int(f(x), x = a..b) で，$\int_a^b f(x)dx$ を表します．a, b が実定数ならば，MuPAD は f(x) を普通の実数値関数と見て積分するので問題ないですが，a, b が文字の場合は，a, b を複素数と見てしまうので，積分できないときがあります．(第 10 章参照)

- int(1/x^2, x = 1/2..1)　　>> 1
- int(1/x^2, x = a..1)　　>> undefined

7.2　assume の利用による プロパティの設定

7.2.1　最初の設定

このような場合，assume を用いて，変数を実数に制限したり，適当な範囲 (例えば，$1 \leqq x \leqq 5$ など) に制限することによって，望みの結果を手に入れることが出来ます．x のプロパティの設定には, 大きく分けて 3 つあります．

- assume(x, Type ::*name*)　– Type ::*name* を指定する方法
- assume(x の等式, 不等式) – 範囲を指定する方法
- assume(x in *Set*)　– 集合を指定する方法

x を Global と入力すると，全ての変数を, 同時にそのタイプに設定することができます．(Global property の設定) また，Type::*name* で設定するときは, x を抜かして，assume(Type ::*name*) のようにしても，Global 設定ができます．

設定の解除は，delete, または, unassume で行います．しかし，全体設定 (Global property) の場合は，unassume でないといけません．

x の設定解除	delete x または unassume(x)
$x, y\cdots$ の設定解除	delete x, y, $\cdots$ または unassume([x, y, $\cdots$])
全体設定 (*global property*) の設定解除	unassume() または unassume(Global)

注2) assume を使って設定された，Type とか, 範囲とか *set* をまとめてプロパティ(mathmatical property) と呼びます．

[Type の設定]

先ずタイプ (Type) を指定する方法があります．次のように指定します．

x を, Type :: *name* というタイプに設定	assume(x, Type ::*name*)
全変数を, Type :: *name* というタイプに設定	assume(Global, Type ::*name*) または assume(Type ::*name*)

注2) unassume([x, y, $\cdots$]) は，unassume({x, y, $\cdots$}) でも OK です．

プロパティ(property) として，assume で使える主な Type は，以下のとおりです．(下表以外にも様々な Type が, property として設定可能です．詳しくは, 章末の表をご覧ください．)

タイプ	Type ::*type* 名 (*name*)
実数	Type :: Real
有理数	Type :: Rational
複素数	Type :: Complex
純虚数	Type :: Imaginary
整数	Type :: Integer
偶数	Type :: Even
奇数	Type :: Odd
$b \cdot$ 整数 $+ a$	Type :: Residue(a, b)(a, b は整数)
正の実数	Type :: Positive
正の有理数	Type :: PosRat
正の整数	Type :: PosInt
負でない実数	Type :: NonNegative
負でない有理数	Type :: NonNegRat
負でない整数	Type :: NonNegInt
負の実数	Type :: Negative
負の有理数	Type :: NegRat
負の整数	Type :: NegInt

例えば，x を正の数 (Type :: Positive) に設定してみます．

- assume(x, Type :: Positive); sqrt(x) * sqrt(1/x) >> $(0, \infty)$ >> 1

$(0, \infty)$ というのは，$0 < x < \infty \longleftrightarrow 0 < x$ ということです．$x > 0$ と設定されたので，$\sqrt{x} \cdot \sqrt{\frac{1}{x}} = 1$ となります．x の設定をはずすと, 元の木阿弥です．

- delete x : sqrt(x) * sqrt(1/x) >> $\sqrt{x} \cdot \sqrt{\frac{1}{x}}$

今度は，x を実数 (Type :: Real) に制限してみます．

- assume(x, Type :: Real); solve(exp(x) = 1, x) >> $\mathbb{R}$ >> {0}

$\mathbb{R}$ というのは, 実数の集合を表します．x が実数に制限されたので，$\exp(x) = 1 \longleftrightarrow e^x = 1$ の解も，$x = 0$ だけとなりました．今度は，x を整数 (Type :: Integer) に制限してみます．

- assume(x, Type :: Integer); solve(x^2 = 1/4, x) >> $\mathbb{Z}$ >> ϕ

$\mathbb{Z}$ というのは, 整数の集合です．また，ϕ は空集合を表します．$x^2 = 1/4$ の解は,$x = \pm\frac{1}{2}$ ですから, 整数の解は持ちません．今度は，x を有理数 (Type :: Rational) に制限してみます．

- assume(x, Type :: Rational); >> $\mathbb{Q}$
- solve(x^2 = 1/4, x); solve(x^2 = −4) >> $\left\{-\frac{1}{2}, \frac{1}{2}\right\}$ >> ϕ

$\mathbb{Q}$ は有理数の集合を表します．$\pm\frac{1}{2}$ は有理数なので，$x^2 = \frac{1}{4}$ は，解を持ちました．しかし，$x^2 = -4$ は解を持ちません．今度は，x を複素数 (Type :: Complex) に制限してみます．

- assume(x, Type :: Complex); solve(x^2 = −4, x)　　$\{-2\cdot i, 2\cdot i\}$

$\mathbb{C}$ は複素数の集合を表します．$\pm 2i$ は複素数ですから, 解を持ちました．今度は,unassume を使って，設定を消去してみます．

- unassume(x); solve(x^2 = −4, x)　　$\{-2\cdot i, 2\cdot i\}$

solve の default は,「複素数の範囲で解を求めること」なので，制限を消去しても，同じです．x の代わりに，*Global* を使うと，全ての変数が, そのタイプになります．全ての変数を正の数に制限してみます．

- assume(Global, Type :: Positive)　　>> $(0,\infty)$
- solve(y^2 = 4, y)　　>> {2}
- sqrt(z) ∗ sqrt(1/z); simplify(%)　　>> $\sqrt{z}\cdot\sqrt{\frac{1}{z}}$　>> 1
- ln(p) + ln(1/p); simplify(%)　　>> $\ln(p)+\ln(1/p)$　>> 1

$y^2 = 4$ の正の解は $y = 2$ だけです．また，z や p も正の数に設定されたので，$\sqrt{a}\cdot\sqrt{b} = \sqrt{ab}, \log x + \log y = \log xy$ の公式も成立しています．Global 設定を解除するには，delete でなく，unassume を使います．

- unassume(Global) : simplify(sqrt(z) ∗ sqrt(1/z))　　>> $\sqrt{z}\cdot\sqrt{\frac{1}{z}}$

このように，もはや，$\sqrt{a}\cdot\sqrt{b} = \sqrt{ab}$ は成り立ちません．

[範囲の設定]

変数の範囲を設定することも，できます．[注3)]

$x = a$ と条件設定	assume(x = a)
$x \neq a$ と条件設定	assume(x <> a)
$x > a$ と条件設定	assume(x > a)
$x \geqq a$ と条件設定	assume(x >= a)
$x < a$ と条件設定	assume(x < a)
$x \leqq a$ と条件設定	assume(x <= a)
$a < x < b$ と条件設定	assume(a < x < b)
$a \leqq x < b$ と条件設定	assume(a <= x < b)
$a < x \leqq b$ と条件設定	assume(a < x <= b)
$a \leqq x \leqq b$ と条件設定	assume(a <= x <= b)
全変数を a より大きく設定	assume(Global > a)
全変数を a より小さく設定	assume(Global < a)

[注3)] > と = の順序は大切です．例えば，$x \geqq 0$ は x>=0 とします．x=>0 と順序を逆にするとエラーになります．

x の範囲を設定するには，assume(x > 0) のように, 不等式を使うことができます.

- assume(x > 0); sqrt(x) * sqrt(1/x) $(0,\infty)$ >> 1

assume(x > 0) で，assume(x, Type :: Positive) と同じです．次は，$0 < x \leqq 1$ に設定してみます.

- assume(0 < x <= 1); solve(x * (x − 1) = 0, x) >> (0, 1] >> {1}

「≦」は「<=」と表します．(0, 1] は,「$0 < x \leqq 1$」を表します．このように，MuPAD は,「≦」や「≧」 の場合は,「[」や「] 」を使って,「<」や「>」の場合は,「(」や「) 」を使って表します．今度は, $0 \leqq x \leqq 1$ に設定してみます.

- assume(0 <= x <= 1); solve(x * (x − 1) = 0, x) >> [0, 1] >> {0, 1}

[集合 (set) による設定]

assume(x in *Set*) [*Set* は, 複素数の部分集合, または, 適当なプロパティ(property)] でも，x の設定が出来ます．*Set* として，Type ::··· を使うときは，assme(x, Type :: ···) と全く同じです.

- assume(x in Type :: Positive); assume(x in Type :: Integer) >> $(0,\infty)$ >> $\mathbb{Z}$
- assume(x, Type :: Positive); assume(x, Type :: Integer) >> $(0,\infty)$ >> $\mathbb{Z}$

このように全く同じになります．違いは, 自分でも簡単に集合 (set) を設定できることです.

- myset := {k $k = 1..10} >> {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
- assume(x in myset) >> $[1, 10] \cap \mathbb{Z}$
- solve(x − 11 = 0, x); solve(x − 10 = 0, x) >> ϕ >> 10

$x = 11$ は，myset の中に入っていないので,「$x - 11 = 0$」は「解なし」となりましたが,「$x - 10 = 0$」は, myset の中に解を持ちます.

7.2.2 追加条件の設定

すでに設定した条件 A に，条件 B を追加することも出来ます.

条件 B を追加し, A と B との共通集合に変更	assume(B, _and)
条件 B を追加し, A と B との和集合に変更	assume(B, _or)

条件を追加するときは _and と _or のオプションを指定します．それぞれ前の条件との共通集合, 和集合を作ります.

- assume(x > 0) : assume(x < 5, _and) >> (0, 5)

(0, 5) で $0 < x < 5$ を表します．$x > 0$ と $x < 5$ の共通部分ですから, $0 < x < 5$ となります．これは次のように設定したのと同じです.

- `assume(0 < x < 5)` >> $(0,\ 5)$

今度は, 和集合を見て見ます．上に続けて，次のように打ち込みます.

- `assume(x > 10, _or)` >> $(0,\ 5) \vee (10,\ \infty)$

これは $x > 10$, または $0 < x < 5$ という事です.

- `assume(x < 20, _and)` >> $(0,\ 5) \vee (10,\ 20)$

これは, $0 < x < 5$ または $10 < x < 20$ という事です.

_and や _or が無いと，新しい範囲に置き換えられます.

- `assume(x < 0)` >> $(-\infty,\ 0)$

7.2.3　assuming によるプロパティの設定

assume は，別の条件を assume するか，または，delete, unassume するまで, その条件は残りますが, 一時的にのみ設定したい場合は，assuming を使います．使い方は，assume とほとんど同じです．[注4)]

- `solve(x^2 = 9, x)  assuming(x, Type :: Positive)` >> {3}
- `solve(x^2 = 9, x)` >> {−3, 3}

1 行目の方では，正の解しか求まっていませんが，2 行目では，負の解も求めています．このように, assuming の方の設定は一時的で，それが付いたコマンドにしか効いていません．不等式や，集合を使って設定するときは, 括弧を省略することも出来ます．(つけても構いません．)

- `solve(x^2 = 9, x)  assuming  x in Type :: Positive` >> {3}
- `solve(x^2 = 9, x)  assuming  0 < x < 3` >> ϕ

7.3　プロパティ(property) のチェック

x は Type :: *name* に入っているか?	`is(x, Type :: name)`
$x > a$ か?	`is(x > a)`
$x < a$ か?	`is(x < a)`
x は, Set の要素か?	`is(x in Set)`
x のプロパティは?	`getprop(x)`

[注4)] assuming は，MuPAD 3.0 より導入されました.

x があるプロパティを持つかどうかは，is(···) でチェック出来ます．··· の中には，assume(···) で使える式が全て使えます．返り値は，*TRUE*(真) か *FALSE*(偽) か *UNKNOWN*(不明) です．

- assume(x, Type :: Integer) >> $\mathbb{Z}$
- is(x, Type :: Integer) >> TRUE
- is(x, Type :: Rational) >> TRUE
- is(x, Type :: Real) >> TRUE
- is(x, Type :: Positive) >> UNKNOWN
- is(x = 1/2) >> FALSE

まず，x を整数 (integer) に設定しました．整数は，実数 (real) や有理数 (rational) でもあるので，is(x, Type :: Real) や is(x, Type :: Rational) は「真」となります．一方，正の整数も, 負の整数もあるので is(x, Type :: Positive) は「不明」となります．ところが，x は整数なので，is(x = 1/2) は¢偽£となります．もう 1 つやってみます．

- assume(x > 0) >> $(0, \infty)$
- is(x > −1) >> TRUE
- is(x < −1) >> FALSE
- is(x > 5) >> UNKNOWN
- is(x in Type :: Real) >> TRUE

さらに複雑な等式, 不等式に対しても is は使えます．

- assume(0 < a < 1) >> $(0, 1)$
- is(a ∗ (2 − a) > 0) >> TRUE
- is(abs(a) = a) >> TRUE

しかし複雑すぎると, うまく行きません．

- is(sin(a ∗ PI) > 0) >> UNKNOWN

$0 < a < 1$ のとき，$0 < a\pi < \pi$ ですから，「$\sin(a\pi) > 0$ は 真」の筈ですが, そこまでは MuPAD は判定できません．しかし,UNKNOWN ですから，少なくとも**嘘は言ってません．**「MuPAD は**正直**だ」とは言えるでしょう．

is(x···) は,「x がある Type や範囲に入っているか？」をチェックするコマンドでした．それに対し, 直接 x のプロパティを得るコマンドが，getprop(x) [注5] です．しかし，getprop(x) は，しばしば，大雑把な範囲を返してきます．

- assume(0 < x < 2) >> $(0, 2)$
- getprop(x) >> $(0, 2)$

[注5] get property

これは正確です．しかし，

• `getprop(x^2 − 2 ∗ x)` >> $(-4, 4)$

実際には，$0 < x < 2$ のとき，$y = x^2 - 2x = (x-1)^2 - 1$ のグラフを考えると $-1 \leqq x^2 - 2x < 0$ となるはずです.「$-4 < x < 4$」 は 「$-1 \leqq x < 0$」を含んでいるので，**間違いではない**ですが，かなり大雑把な答えといえます.

7.4　ドメインタイプとプロパティ

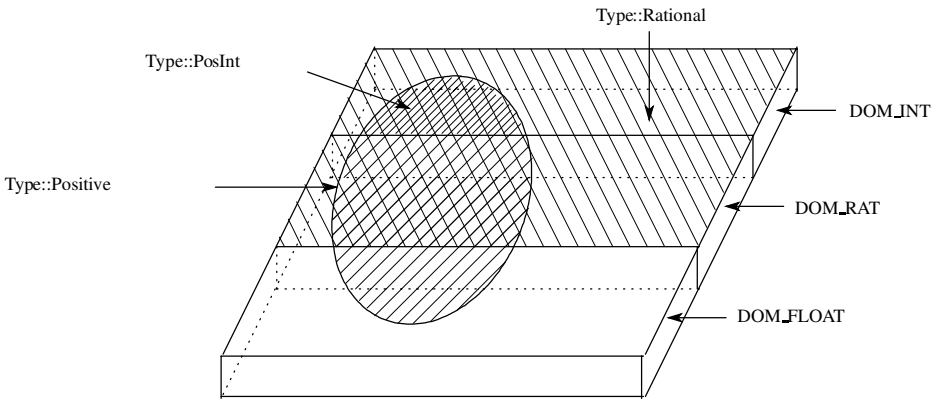
第 6 章で, 基本データ型の ドメインタイプ（domain type）に関して触れました．それとプロパティ(property) の関係はどうなっているのでしょうか？

• `domtype(3); is(3, Type :: Integer)` >> DOM_INT >> TRUE

「３」の domain type は，DOM_INT で,「３」は Type::Integer にも入っています．しかし，

• `is(3, Type :: Rational); is(3, Type :: Positive)` >> TRUE >> TRUE

すなわち，各オブジェクトは 1 つの domain type しか持てませんが，property は，たくさん持つことが出来ます．数学的には，¢3£は「整数」でもあり,「有理数」でもあり，また「正の数」でもあるわけですから，オブジェクトの，数学的な意味に，より対応しているのがプロパティ(mathmatical property) となります．一方，MuPAD 内部で計算を実行するためのデータ型が，ドメインタイプ (domain type) になります.（MuPAD の方では，それぞれ，semantic と syntactic と呼んでいます.）



Type::Rational は，DOM_INT と DOM_RAT の和集合．また,
Type:: Positive は，DOM_INT, DOM_RAT, DOM_FLOAT 全てにまたがる.
さらに，Type::PosInt は，Type::Positive と DOM_INT の共通集合.

乱暴にたとえると，ドメインタイプは，木材，竹，ブロック，コンクリートなどの**建築素材**で，プロパ

ティの方は，別荘，数寄屋造り，校倉造り，京の町屋，モダン建築，高層ビル，洋風建築, 和風建築，マンション, アパート，長屋，一戸建てなどの**建物の種類**にあたると思います．[注6)]

[注6)] 以上のことは，`testtype(x, T)` (T は Type または，domain type) を使うと, さらによく解ります．`testtype` は，property の他, domain type などの’ タイプ’ も, 調べることが出来ます.

- `testtype(3, DOM_INT)` >> TRUE
- `testtype(3, DOM_RAT)` >> FALSE
- `testtype(3, Type :: Rational)` >> TRUE
- `testtype(3, Type :: Real)` >> TRUE
- `testtype(3, Type :: Positive)` >> TRUE

御覧のように，「３」は，DOM_RAT(有理数の domain type) には属しませんが，Type::Rational(有理数), Type::Real(実数), Type::Positive(正の数) など複数のプロパティを持ちます.

7.4.1　プロパティ(property) の表

以下に，MuPAD で利用可能な property のリストを挙げます.. [注7)]

basic	`Type :: Complex`	$\mathbb{C}$
number	`Type :: Even`	$2\mathbb{Z}$
domains	`Type :: Imaginary`	$\mathbb{R}\,i$
	`Type :: Integer`	$\mathbb{Z}$
	`Type :: Negative`	$\mathbb{R}_{<0}$
	`Type :: NegInt`	$\mathbb{Z}_{<0}$
	`Type :: NegRat`	$\mathbb{Q}_{<0}$
	`Type :: NonNegative`	$\mathbb{R}_{\geqq 0}$
	`Type :: NonNegInt`	$\mathbb{N}$
	`Type :: NonNegRat`	$\mathbb{Q}_{\geqq 0}$
	`Type :: NonZero`	$\mathbb{C} \setminus \{0\}$
	`Type :: Odd`	$2\mathbb{Z}+1$
	`Type :: PosInt`	$\mathbb{N}_{>0}$
	`Type :: Positive`	$\mathbb{R}_{>0}$
	`Type :: PosRat`	$\mathbb{Q}_{>0}$
	`Type :: Prime`	prime numbers
	`Type :: Rational`	$\mathbb{Q}$
	`Type :: Real`	$\mathbb{R}$
	`Type :: Zero`	$\{0\}$
intervals	`Type :: Interval(a, b, T)`	$\{x \in T : a < x < b\}$
	`Type :: Interval([a], b, T)`	$\{x \in T : a \leqq x < b\}$
	`Type :: Interval(a, [b], T)`	$\{x \in T : a < x \leqq b\}$
	`Type :: Interval([a], [b], T)`	$\{x \in T : a \leqq x \leqq b\}$
		a,b: expressions
		T: basic number domain
residue	`Type :: Residue(a, b)` or	$b\mathbb{Z}+a$
classes	`b * Type :: Integer + a`	a, b : integers
relations	= b	$\{b\}$
	<> b	$\mathbb{C} \setminus \{b\}$
	< b	$\mathbb{R}_{<b}$
	<= b	$\mathbb{R}_{\leqq b}$
	> b	$\mathbb{R}_{>b}$
	>= b	$\mathbb{R}_{\geqq b}$
		b : expression

Types of mathmatical properties available in MuPAD.

これから,「x を，0 より大きく 100 より小さい整数に指定」するのは,

- `assume(x, Type :: Interval(0, 100, Type :: Integer))`　　>> $[1,99] \cap \mathbb{Z}$

注7) これは，MuPAD Pro 3.0 Tutorial に載っていたものを, 私が書き写したものです. `assume(x, Type :: Prime)` は使えませんが, `is(x, Type :: Prime)` は使えます.

とすればよいことが解ります．つまり，次のように指定したのと同じです．

- assume(0 < x < 100); assume(x, Type :: Integer, and) >> (0, 100) >> [1, 99] ∩ ℤ

また,assume(1 < x < 10) の代わりに

- assume(x, Type :: Interval(1, 10, Type :: Real)) >> (1, 10)

と指定できることも解ります．(その表にはありませんが，Type :: Real を省略することもできます)

注8)

注8) おそらく，MuPAD の内部では, 不等式による設定なども，Type :*name* による設定に書き直されて保存されているものと思われます．

- myset := {1, 4} >> {1, 4}
- assume(x in myset) >> [1, 1] ∩ ℤ ∨ [4, 4] ∩ ℤ

これは，次のようにしたのと同じです．

- assume(x, Type :: Interval([1], [1], Type :: Integer)) >> [1, 1] ∩ ℤ
- assume(x, Type :: Interval([4], [4], Type :: Integer), _or) >> [1, 1] ∩ ℤ ∨ [4, 4] ∩ ℤ

第8章　方程式·不等式を解く(solve)

方程式 $f=0$ を x に関し解く	`solve(f = 0, x)`
方程式 $f=0$ を x に関し解く	`solve(f, x)`
不等式 $f>0$ を x に関し解く	`solve(f > 0, x)`
不等式 $f \geqq 0$ を x に関し解く	`solve(f >= 0, x)`
連立方程式 $f=0, g=0, \cdots$ を $x, y, \cdots$ に関し解く	`solve( {f = 0, g = 0, ···}, [x, y, ···] )`
連立不等式 $f>0, g>0, \cdots$ を $x, y, \cdots$ に関し解く	`solve( {f > 0, g > 0, ···}, [x, y, ···] )`
$f=0$ の解の小数近似	`numeric :: solve(f = 0, x)`
$f=0$ の解の小数近似 ($a \leqq x \leqq b$ の解のみを求める)	`numeric :: solve(f = 0, x = a..b)`
ans の小数近似	`float(ans)`

注1)

- `solve` の *option* – MaxDegree = n(n は自然数), Domain = d, IgnoreSpecialCases, BackSubstitution = b, DontRewriteBySystem,Multiple, PrincipalValue, IgnorProperties
- Domain = d で指定できる主な値 – Dom::Complex(複素数), Dom::Real(実数), Dom::Rational(有理数), Dom::Integer(整数), Dom::Interval(a, b)

`solve` はいろいろな方程式 · 不等式を解くことができます．default では, `solve` は複素数の範囲で解を求めます．これを実数全体に制限したいときは,`assume(x, Type :: Real)` のように x に *property* を与えるか，または option の Domain= d を使って `solve(f, x, Domain = Dom :: Real)` のように制限します．(この本では, 原則として，`assume` を使って, 範囲を指定します．) また Default では 解の公式は 2 次までの解の公式しか使用しません．これを 3 次,4 次の解の公式を使用させる場合は `MaxDegree = 3, Maxdegree = 4` のように指定します．さらに `solve(f, x)` は,$f(x)=0$ が解けない場合, `RootOf(f(x1), x1)` のように，x は f(x)=0 の解である ことを表現します．注2)

8.1　整式の方程式，不等式

整式の方程式 · 不等式は MuPAD は正確に解くようです.

8.1.1　整式の方程式

例を見て見ましょう.

注1) 連立方程式，不等式は { } を使っても，[ ] を使っても大丈夫です．例えば，`solve( {f = 0, g = 0, ···}, [x, y, ···] )` は，`solve( {f = 0, g = 0, ···}, {x, y, ···} )` や，`solve( [f = 0, g = 0, ···], [x, y, ···] )` でも大丈夫です.

注2) 変数の指定は省略できますが, その場合は $\{[x=1], [x=2]\}$ のように 未知数 = 値 で示されます．一方，変数を指定したときは $\{1, 2\}$ のように, 表されます.

$x^2 - 3x + 2 = 0$ の解は?

- `solve(x^2 − 3 ∗ x + 2 = 0, x)` >> $\{1, 2\}$

$x^2 - 3x + 1 = 0$ を x に関して解くと解が $\{1, 2\}$ ということです．ここで，{ } は集合を表す記号です．
根号が出てくるような方程式も解けます.(ただし、MuPAD Light では，$\sqrt{a}$ は $a^{\frac{1}{2}}$ のように出力されます.)

$2x^2 - 5x + 1 = 0$ の解は?

[Light] • `solve(2 ∗ x^2 − 5 ∗ x + 1 = 0, x)` >> $\left\{5/4 - \frac{17^{\frac{1}{2}}}{4}, \frac{17^{\frac{1}{2}}}{4} + 5/4\right\}$

[Pro] • `solve(2 ∗ x^2 − 5 ∗ x + 1 = 0, x)` >> $\left\{\frac{5}{4} - \frac{\sqrt{17}}{4}, \frac{\sqrt{17}}{4} + \frac{5}{4}\right\}$

虚数解も出せます．虚数単位は I と示されます．$2x^2 + 3x + 4 = 0$ の解は?

[Light] • `solve(2 ∗ x^2 + 3 ∗ x + 4 = 0, x)` >> $\left\{-\frac{1}{4} I\, 23^{\frac{1}{2}} - 3/4, \frac{1}{4} I\, 23^{\frac{1}{2}} - 3/4\right\}$

[Pro] • `solve(2 ∗ x^2 + 3 ∗ x + 4 = 0, x)` >> $\left\{-\frac{i}{4}\sqrt{23} - \frac{3}{4}, \frac{i}{4}\sqrt{23} - \frac{3}{4}\right\}$

普通の書き方では，それぞれ，$\frac{5 \pm \sqrt{17}}{4}, \frac{-3 \pm \sqrt{23}i}{4}$ となります．Light の方はちょっと答えは見づらいです．
以下，特に断りが無ければ，MuPAD Pro の方のみ書きます.

$f = 0$ の「0」は省略できます.

- `solve(2 ∗ x^2 + 3 ∗ x + 4, x)` >> $\left\{-\frac{i}{4}\sqrt{23} - \frac{3}{4}, \frac{i}{4}\sqrt{23} - \frac{3}{4}\right\}$

さらに，文字が一個のときは，未知数の指定は省略できます．このとき $[x = \cdots]$ のように, 未知数とセットで表されます.

- `solve(2 ∗ x^2 + 3 ∗ x + 4)` >> $\left\{\left[x = -\frac{i}{4}\sqrt{23} - \frac{3}{4}\right], \left[x = \frac{i}{4}\sqrt{23} - \frac{3}{4}\right]\right\}$

2 つ以上の変数があるときは，省略するとどちらかの変数に関して解かれます．指定すると，指定した文字に関し解かれます.

- `solve(x + 2 ∗ y = 1)` >> $\{[x = 1 - 2 \cdot y]\}$

$2x + y = 1$ を x に関し解くには?

- `solve(x + 2 ∗ y = 1, x)` ; >> $\{1 - 2y\}$

$2x + y = 1$ を y に関し解くには?

- `solve(x + 2 ∗ y = 1, y)` ; >> $\left\{\frac{1}{2} - \frac{x}{2}\right\}$

３次以上の方程式も同様に解けます．$x^4 - x^3 - 2x^2 + x + 1 = 0$ の解は?

• solve(x^4 − x^3 − 2 ∗ x^2 + x + 1 = 0, x)　　>> $\left\{-1, 1, \frac{1}{2} - \frac{\sqrt{5}}{2}, \frac{\sqrt{5}}{2} + \frac{1}{2}\right\}$

ただし，3 次,4 次方程式の解の公式を使って解く必要がある場合は，MaxDegree=3, MaxDegree=4 とします．(後述) [注3)]
さらに, 文字定数を含み，場合わけが必要な場合は，ちゃんと場合わけをして表示します．

• solve(a ∗ x = b, x)　　>> $\begin{cases} \left\{\frac{b}{a}\right\} & if \quad a \neq 0 \\ \mathbb{C} & if \quad a = 0 \vee b = 0 \\ \phi & if \quad b \neq 0 \vee a = 0 \end{cases}$

$a = 0$ のときは，$\frac{b}{a}$ は存在しないので，場合分けしています．このような例外を無視したいときは, option で IgnoreSpecialCases を指定します．

• solve(a ∗ x = b, x, IgnoreSpecialCases)　　>> $\left\{\frac{b}{a}\right\}$

8.1.2 整式の不等式

不等式も, 方程式と同じように解けます．ただ，≦, ≧ などはキーボードにないので <=, >= のように入力します．また, 不等式では実数解だけを出します．[注4)]

$x^2 - 3x + 2 < 0$ の解は?

[*Pro*]• solve(x^2 − 3 ∗ x + 2 < 0, x)　　>> (1, 2)
[*Light*]• solve(x^2 − 3 ∗ x + 2 < 0, x)　　>>]1, 2[

$x^2 - 3x + 2 \leqq 0$ の解は?

• solve(x^2 − 3 ∗ x + 2 <= 0, x)　　>> [1, 2]

ここで，　”]1,2[や (1,2)” は $1 < x < 2$, ”[1,2]” は $1 \leqq x \leqq 2$ を表します．

区間が無限に広い場合は infinitity(∞) と, −infinity(−∞) を使って示されます．

注3) 検算をして見ます．$f(x) = x^4 - x^3 - 2x^2 + x + 1$ とおくと,

$$f(1) = 1 - 1 - 2 + 1 + 1 = 0,\ f(-1) = 1 + 1 - 2 - 1 + 1 = 0$$

よって，$f(x)$ は $(x-1)$ と $(x+1)$ を因数にもちます．$f(x)$ を $(x-1)(x+1)$ で割って,

$$f(x) = (x-1)(x+1)(x^2 - x - 1)$$

ゆえに，$f(x) = 0$ とおくと

$$x = \pm 1,\ \text{または},\ x^2 - x - 1 = 0 \Longleftrightarrow x = \pm 1,\ x = \frac{1 \pm \sqrt{5}}{2}$$

確かに結果は一致します．
注4) まちがって =< のように入力するとエラーがでます．

$x^2 > 4$ の解は?

[*Light*]• `solve(x^2 > 4, x)` >>]2, infinity[union]-infinity, -2[

[*Pro*]• `solve(x^2 > 4, x)` >> $(2, \infty) \cup (-\infty, -2)$

union(∪) というのは和集合の意味です．ですから，$-\infty < x < -2$ または $2 < x < \infty$ ということです.

不等式では実数解だけ求まります.

◇$x^2 + 1 < 0$ の解は?

[Light]• `solve(x^2 + 1 < 0, x)` >> { }

[Pro]• `solve(x^2 + 1 < 0, x)` >> ϕ

{ } も,ϕ も「空集合」を表していて,「解なし」ということです．他の例も見て見ます.

◇$x^2 - 2x + 1 \leqq 0$ の解は?

• `solve(x^2 - 2 * x + 1 <= 0, x)` >> { 1}

$x^2 - 2x + 1 \leqq 0 \Longleftrightarrow (x-1)^2 \leqq 0 \Longleftrightarrow x = 1$ ですから，正解です.

8.1.3　整式の連立方程式 · 不等式

連立方程式も, 同様に解くことが出来ます．方程式と未知数を，それぞれ { } や [ ] の中にまとめて入れるだけです．また未知数の指定を省略すると適当な文字に関して解きます．[注5)]

$\begin{cases} x + y = 2 \\ x - y = 0 \end{cases}$ の解は?

• `solve({x + y = 2, x - y = 0}, [x, y])` >> {[x = 1, y = 1]}

$\begin{cases} x + y + z = 2 & \cdots ① \\ x^2 + y^2 + y^2 = 14 & \cdots ② \\ x^3 + y^3 + z^3 = 20 & \cdots ③ \end{cases}$ の解は?

• `eq := {x + y + z = 2, x^2 + y^2 + z^2 = 14, x^3 + y^3 + z^3 = 20}`

>>$\{x + y + z = 2,\ x^2 + y^2 + z^2 = 14,\ x^3 + y^3 + z^3 = 20\}$

[注5)] { } は集合 (set) を表します．集合では要素の順番を考えません．一方 [ ] はリスト (list) で，要素の順番を考えます．方程式や未知数の指定は, セットでもリストでも，大丈夫です.

•solve(eq, [x, y, z])

$$>>\{[x = 1, y = -2, z = 3], [x = 1, y = 3, z = -2], [x = -2, y = 1, z = 3],$$
$$[x = -2, y = 3, z = 1], [x = 3, y = 1, z = -2], [x = 3, y = -2, z = 1]\}$$

注6) 連立不等式も解けます.

$$\diamond \begin{cases} x^2 + x - 2 < 0 & \cdots ① \\ x^2 - x - 3 > 0 & \cdots ② \end{cases} \text{ の解は?}$$

•ineq := {x^2 + x − 2 < 0, x^2 − x − 3 > 0}

$$>> \text{ineq} := \{x^2 + x - 2 < 0,\ x^2 - x - 3 > 0\}$$

•solve(ineq, x)

$$>> \left(-2,\ \frac{1}{2} - \frac{\sqrt{13}}{2}\right)$$

これは

$$-2 < x < \frac{1}{2} - \frac{\sqrt{13}}{2}$$

を表しています. 注7)

8.1.4 整式の方程式が解けないとき (MaxDegree と RootOf)

solve(f, x) は,$f(x) = 0$ が解けない場合, RootOf(f(x1), x1) のように，x は f(x)=0 の解である ことを表現します. また Default では, 整式の方程式を解くとき，2 次までの解の公式しか使用しません. もし，3 次, 4 次の解の公式を使用させる場合は MaxDegree = 3, Maxdegree = 4 のように指定します. 注8)

$\diamond x^3 + 6x - 2 = 0$ を解いてみましょう.

• eq := x^3 + 6 ∗ x − 2	$>> 6 \cdot x + x^3 - 2$
• solve(eq, x)	>> RootOf($\boldsymbol{X18}^3 + 6\boldsymbol{X18} - 2, \boldsymbol{X18}$)

注6) 手計算で，検算してみます. $(x + y + z)^2 = x^2 + y^2 + z^2 + 2(xy + yz + zx)$ だから，①,② より,

$$2^2 = 14 + 2(xy + yz + zx) \Longleftrightarrow xy + yz + zx = -5 \qquad \cdots ④$$

$x^3 + y^3 + z^3 - 3xyz = (x + y + z)(x^2 + y^2 + z^2 - xy - yz - zx)$ だから①,②,③,④ より,

$$20 - 3xyz = 2 \cdot \{14 - (-5)\} \Longleftrightarrow xyz = -6 \qquad \cdots ⑤$$

①,④,⑤より x, y, z を解に持つ方程式は

$$t^3 - 2t^2 - 5t + 6 = 0 \Longleftrightarrow (t - 1)(t - 3)(t + 2) = 0 \Longleftrightarrow t = 1, 3, -2$$

よって方程式の解の集合は

$$\{x, y, z\} = \{1, 3, -2\}$$

確かに, 一致します.

注7) ① $\Longleftrightarrow -2 < x < 1$, ② $\Longleftrightarrow x < \frac{1-\sqrt{13}}{2}, x > \frac{1+\sqrt{13}}{2}$ なので確かに合っています.

注8) 5 次以上の解の公式は無いので,MaxDegree=5 と指定するのは無意味です.

RootOf($\boldsymbol{X18}^3 + 6\,\boldsymbol{X18} - 2, \boldsymbol{X18}$) というのは x の解集合が，$\boldsymbol{X18}^3 + 6\,\boldsymbol{X18} - 2 = 0$ をみたす $\boldsymbol{X18}$ と一致している事を表しています．すなわち, 全く簡単になっていません．(ここで $\boldsymbol{X18}$ というのは MuPAD が勝手に付けてくる変数名です．もしかしたら, その日に使った 18 番目の変数という意味なのかもしれません.) これを MaxDegree=3 として, もう一度，解いてみます.

- `ans := solve(eq, x, MaxDegree = 3)`

>> $\left\{\sqrt[3]{4} - \frac{4^{\frac{2}{3}}}{2},\ \frac{4^{\frac{2}{3}}}{4} - \frac{\sqrt[3]{4}}{2} - \frac{i}{2}\cdot\left(\sqrt[3]{4} + \frac{4^{\frac{2}{3}}}{2}\right)\cdot\sqrt{3},\ \frac{i}{2}\cdot\left(\sqrt[3]{4} + \frac{4^{\frac{2}{3}}}{2}\right)\cdot\sqrt{3} - \frac{\sqrt[3]{4}}{2} + \frac{4^{\frac{2}{3}}}{4}\right\}$

3 次方程式の解の公式を使ったので, うまく解けました.

念のため subs を使って検算してみます．このとき op というコマンドを使うと便利です．[注9)]

解には ans, 方程式には eq という名前をつけていたので, 次のようにやれます.

- `x1 := op(ans, 1)`　　>> $\sqrt[3]{4} - \frac{4^{\frac{2}{3}}}{2}$

ans の最初の要素が出てきました．eq に代入してみます.

- `subs(eq, x1)`　　>> $6\cdot\sqrt[3]{4} - \left(\frac{4^{\frac{2}{3}}}{2} - \sqrt[3]{4}\right)^3 - 3\cdot 4^{\frac{2}{3}} - 2$
- `simplify(%)`　　>> 0

確かにうまく行きました．残りの解も同様にして，検算することができます.

8.2　解の小数近似 (数値解)

先のように，超なが〜い解が出てきた時は, 小数近似をすると解の見当がつきます．`solve` した解の近似値を求めるには,`float` を使います．一方，厳密解は必要なく，近似値だけが欲しいときは，numeric library の `numeric :: solve` を使った方が早いです.

[注10)] 先の方程式 $x^3 + 6x - 2 = 0$ の近似解を求めてみます.

- `ans := solve(x^3 + 6 * x - 2, x)`　　>> RootOf($\boldsymbol{X19}^3 + 6\boldsymbol{X19} - 2, \boldsymbol{X19}$)
- `float(%)`　　>> {−1.699628148, 0.2391232783, 2.46050487}

[注9)] MuPAD の数式は operand とよばれる 部品 (building block) でできています．(第 6 章参照) `op(f)` で f の operand を全て，`op(f, 1)` で f の 1 番目の operand，`op(f, 2)` で f の 2 番目の operand，⋯ というように取り出すことができます．(ただ operand の順番は, 画面に見えているのとは必ずしも一致しないので注意が必要です.)

例えば，`f := x + 2` のとき, f の 1,2 番目の operand と，operand 全てを取り出してみます.

- `f := x + 2 : op(f, 1); op(f, 2); op(f)`　　>> x,　>> 2,　>> x, 2

[注10)] MuPAD 2.0 では，解 (ans) が求まっているときは float(ans) で，RootOf() の形でしか求まっていないときは, map(ans,float) を使わないといけませんでした．MuPAD 2.5 からはどちらの場合でも float(ans) で大丈夫です．また `numeric :: solve(f, x)` は `float(hold(solve)(f, x))` と同じです.

最初から, 近似値のみが欲しいときは numeric :: solve の方が早いです.

- numeric :: solve(x^3 + 6 ∗ x − 2, x) >> {−1.699628148, 0.2391232783, 2.46050487}

さらに, f が整式でないときは, 例えば, numeric :: solve(f, x = 1..5) のようにすると, $1 \leqq x \leqq 5$ の範囲の解だけを求めることができます. しかし, f が整式のときは, 範囲指定は, 働きません.

8.3 解の範囲の指定

default では, solve は複素数の範囲で解を求めます. これを実数全体に制限したいときは, assume(x, Type :: Real) のように x に *property* を与えるか, または option の Domain= d を使って solve(f, x, Domain = Dom :: Real) のようにして制限します. (しかし, numeric :: solve を使う場合は, assume や *Domain* ではなく, numeric :: solve(f, x = 1..5) のようにして指定します.)

◇$x^2 + 1 = 0$ の解は?

- solve(x^2 + 1, x) >> $\{-i, i\}$

範囲を実数に制限して, 同じ方程式を解いてみます.

- assume(x, Type :: Real); solve(x^2 + 1, x) >> $\mathbb{R}$ >> ϕ

$\mathbb{R}$ は実数という意味です. ϕ は空集合, すなわち「解なし」を表します. $x^2 + 1 = 0$ は実数解を持たないので, 解がなくなってしまいました. また, assume の代わりに option の *Domain* を使っても大丈夫です. ただしこのとき Type::Real でなく Dom::Real を使わないといけません.

- delete x; solve(x^2 + 1, x, Domain = Dom :: Real) >> >> ϕ

注11)

しかし, assume, assuming を使っても同様の指定ができますし, Domain による指定は, やや難しく, さらに, この本では, 前章で property をやや詳しく扱ったことなどから, この本では, 変域は, assume によって設定します. assume による指定の詳しい説明は, 第 7 章をご覧ください.

8.4 整式以外 (三角関数, 指数 · 対数など) の方程式 · 不等式

三角関数や, 指数 · 対数の方程式も, 同様に解けます. しかし, 整式以外の方程式では, うまく解が求まらないことも, また間違った解が出てくることもあります. また, 整式と違い float を使っても, 全ての解が求まらないこともあります. (解が無数にあることもあるので, 全ての解を小数で近似するわけには行きません. 簡単に小数で表せないときは, **1 個だけ**を求めて, 仕事を終わってしまいます.) このような場合は, numeric :: solve(f, x = a..b) を使って, 適当に x の範囲を制限して求めます.

注11) Domain の値は, 適切な Dom::· · · を指定しますが, この Dom というライブラリは古くからある重要なライブラリです. 一方, property による指定は, わりと新しい方法です. 現在,2 つの方法がミックスしているので, やや複雑になっています.
Domain=d で指定できる主な値は, 下のようになります.

Dom::Complex(複素数), Dom::Real(実数),Dom::Rational(有理数), Dom::Integer(整数), Dom::Interval(a,b)(区間 $a < x < b$)

8.4.1　三角関数の方程式 · 不等式

単位はラジアン (*rad*) になります.

◇ $\sin x = 1$ の解は?

- [*Light*]`solve(sin(x), x)`　　>> `1/2 * PI + 2 * X1 * PI | X1 in Z_`
- [*Pro*]`solve(sin(x), x)`　　>> $\left\{\frac{\pi}{2} + 2\pi k \middle| k \in \mathbf{Z}\right\}$

$\mathbb{Z}$ は整数の集合を表します. したがって $\sin(x) = 1$ の解が $\frac{\pi}{2} + 2n\pi$(n は整数) ということです. x の範囲を指定したいときは `assume` を使って先に指定します.

◇ $\sin(x) = 1$ $(0 \leqq x < 2\pi)$ の解は？

- `assume(0 <= x < 2 * PI)`　　>> `[0, 2 * PI)`
- `solve(sin(x) = 1, x)`　　>> $\left\{\frac{\pi}{2}\right\}$

予想どおり，解は $\frac{\pi}{2}$ だけになりました.

MuPAD は $\sin^2 x + \cos^2 x = 1$ を利用した方程式も解きます.

◇ $2\cos^2 x + 5\sin x + 1 = 0$ $(0 \leqq x < 2\pi)$ を解いてみましょう．変域を設定してから解きます.

- `assume(0 <= x < 2 * PI)`　　>> $[0, 2 \cdot \pi)$
- `solve(2 * cos(x)^2 + 5 * sin(x) + 1, x)`　　>> $\left\{\frac{7 \cdot \pi}{6}, \frac{11 \cdot \pi}{6}\right\}$

注12)

しかし合成公式を使う方程式や, 少し複雑な方程式は解けません．また連立方程式は解けないみたいです．さらに，三角不等式は，まったく解けないみたいです．例えば, MuPAD は

$$\sin x + \sqrt{3}\cos x = 1 \quad \text{や} \quad \begin{cases} \sin x + \sin y = 1 \\ \cos x + \cos y = \sqrt{3} \end{cases} \quad \text{や} \quad \sin x \leqq \frac{1}{2}$$

は解けません．そういう時は，`numeric :: solve` を使って数値解を求めます.

注12) 手計算で確認してみましょう．$\cos^2 x = 1 - \sin^2 x$ ですから,

$$2\cos^2 x + 5\sin x + 1 = 0 \Longleftrightarrow 2(1 - \sin^2 x) + 5\sin x + 1 = 0 \Longleftrightarrow 2\sin^2 x - 5\sin x - 3 = 0$$
$$\Longleftrightarrow (2\sin x + 1)(\sin x - 3) = 0 \Longleftrightarrow \sin x = -\frac{1}{2},\ \sin x = 3 \Longleftrightarrow x = \frac{7\pi}{6},\ x = \frac{11\pi}{6}$$

となり一致しました.

[数値解 (numeric::solve の利用)]

例として，MuPAD で，$\sin x + \sqrt{3}\cos x = 1$ $(0 \leqq x < 2\pi)$ を解いてみます.

- `assume(0 <= x < 2 * PI)` >> $[0, 2\cdot\pi)$
- `eq := sin(x) + sqrt(3) * cos(x) - 1` >> $\sin(x) + \sqrt{3} * \cos(x) - 1$
- `ans := solve(eq, x)` >> 超長〜い解
- `Simplify(ans)` >> 砂時計

私は，12 時間計算させ続けましたが, 砂時計が見えるだけでした．実は, 手計算では, 簡単にこの方程式が，$x = \frac{\pi}{2}, \frac{11\pi}{6}$ を解に持つことがいえます．[注13)]
そこで今度は, 数値解を求めてみます．まずは `float` でやってみます.

- `float(ans)` >> 超長〜い解

実用になりません．そこで今度は `numeric :: solve` を使ってみます.

- `numeric :: solve(eq, x)` >> $\{-226.7182698\}$

このように **整式以外のときは，numeric::solve では, 無数の解のうち一つしか出しません．** 範囲を $0 \leqq x < 2\pi$ に制限してみましょう．それには 単なる `x` の代わりに，`x = 0..2 * PI` とすれば良いです.

- `numeric :: solve(eq, x = 0..2 * PI)` >> $\{1.576796327\}$
- `float(PI/2)` >> 1.576796327

これは，およそ $x = \frac{\pi}{2}$ になることが解ります．ところが, この方程式は $x = \frac{11\pi}{6}$ も解に持つ筈です．そこで範囲を変えてみます.

- `numeric :: solve(eq, x = PI..2 * PI)` >> $\{5.759586532\}$
- `float(11 * PI/6)` >> 5.759586532

こちらは，およそ $x = \frac{11\pi}{6}$ になることが解ります．このように，`numeric :: solve` で区間を制限した場合でも, 答えは一個しか出ないので，注意が必要です.

注13) 手計算で $\sin x + \sqrt{3}\cos x = 1$ を解いてみます．合成公式より,

$$\sin x + \sqrt{3}\cos x = 2\sin\left(x + \frac{\pi}{3}\right)$$

よって, 与式より

$$\begin{aligned}\sin x + \sqrt{3}\cos x = 1 &\Longleftrightarrow 2\sin\left(x + \frac{\pi}{3}\right) = 1 \Longleftrightarrow \sin\left(x + \frac{\pi}{3}\right) = \frac{1}{2}\\ &\Longleftrightarrow x + \frac{\pi}{3} = \frac{\pi}{6} + 2n\pi,\ x + \frac{\pi}{3} = \frac{5\pi}{6} + 2n\pi\\ &\Longleftrightarrow x = -\frac{\pi}{6} + 2n\pi,\ x = \frac{\pi}{2} + 2n\pi\end{aligned}$$

よって，$0 \leqq x < 2\pi$ の解は

$$x = \frac{\pi}{2}, \frac{11\pi}{6}$$

となります.

8.4.2　指数 · 対数の方程式

MuPAD では，Default では, 原則として，関数は複素関数となっています．この結果，範囲を指定しないとき，指数方程式は，無数の解を持ちます.

[Light] • `solve(exp(x) = 1, x)` >> $\left\{2 * I * \text{X1*PI} \,\middle|\, \text{X1 } in \text{ Z\_}\right\}$

[Pro] • `solve(exp(x) = 1, x)` >> $\left\{(2 \cdot i) \cdot \pi \cdot k \middle| k \in \mathbb{Z}\right\}$

すなわち，$e^x = 1$ の解は $x = 2n\pi i$(n は任意の整数) ということです．これは

x が複素数のとき

$$e^{ix} = exp(ix) = \cos x + i \sin x$$

から来ています．実際，$x = 2n\pi$ のとき,

$$e^{2\pi ni} = \cos(2n\pi) + i \sin(2n\pi) = 1$$

となります．すなわち，$z = 2n\pi i$(n は整数) は，$e^z = 1$ の解となっています．このように x が複素数のとき，$e^x = a$(a は定数) の解は周期が $2\pi i$ の周期解になります．したがって，実数の解だけが欲しいときは,x の範囲を制限する必要があります.

•`assume(x, Type :: Real)`	>> $\mathbb{Z}$
•`solve(exp(x) = 1, x)`	>> {0}

x が実数のときは, 確かに，$e^x = 1$ の解は $x = 0$ だけ になりました.

注 14)

今度は対数方程式 · 不等式を解いてみましょう.

◇ $\log_{\frac{1}{2}} \geqq -2$ の解は

• `solve(log(1/2, x) >= -2, x)` $(0, 4]$

$\log_{\frac{1}{2}} x \geqq -2 \Longleftrightarrow \log_{\frac{1}{2}} x \geqq \log_{\frac{1}{2}} \left(\frac{1}{2}\right)^{-2} \Longleftrightarrow \log_{\frac{1}{2}} x \geqq \log_{\frac{1}{2}} 4 \Longleftrightarrow 0 < x \leqq 4$ なので合っています．不等式の場合は，MuPAD は実数と考えて解くので，x を実数に `assume` する必要がありません．このように左辺または右辺のみに未知数があるときは，MuPAD は不等式でも解けます．ところが両辺に未知数があると, 方程式でも厳密解は求まりません.

注14) 三角関数も複素関数として扱われていますが，$\sin x = k$(ただし, k は $-1 \leqq k \leqq 1$ の実数) のときは，x は実数解しか持たないので, 複素関数であることを, 特に意識しません.

◇ $\log_2 x = \log_4 x$ の実数解は?

- `assume(x > 0); solve(log(2, x) = log(4, x), x)`

  >> $(0, \infty)$ >> $(0, \infty) \cap$ solve(log(2,x)-log(4,x)=0, x, IgnoreProperties=TRUE)

"IgnoreProperty=TRUE " というのは,「$x > 0$ という property を無視して解く」という事で, その前に $(0, \infty)\cap$ があることから,「とりあえず解いた**後**で共通部分をとろう」としていることが分ります. いずれにせよ, 解けないので, 数値解を求めてみます.

- `float(%)` >> Error: unknown option [numeric::solve]

うまく行きません. そこで `numeric :: solve` を使うことにします.

- `numeric :: solve(log(2, x) = log(4, x), x)` >> {1.0}

$\log_2 x = \log_4 x \iff \log_2 x = \frac{\log_2 x}{\log_2 4} \iff 2\log_2 x = \log_2 x \iff \log_2 x = 0 \iff x = 1$ なので合っています. しかしこれは数値解です.

8.4.3 MuPAD が苦手な方程式

人間にはすぐ解が求まるが, MuPAD だとなかなか求められない (または数値解しか求まらない) 例を, いろいろ見てみます. $\log x = \frac{x}{e}$ の実数解は $x = e$ です. (代入してみると明らか. これ以外に実数解を持たないことは, 微分を利用するか, グラフを描くとすぐに解ります.) これを MuPAD で解いて見ましょう.

- `eq := ln(x) - x/E` >> $\ln(x) - \frac{x}{e}$
- `ans := solve(eq, x)` >> $\left\{ e^{-W_k(-\frac{1}{e})} \middle| \, k \in \mathbb{Z} \right\}$

ここで $W_k(x)$ というのは MuPAD 3.0 から導入された *lambertW*(k, x) という関数で $y = xe^x$ の逆関数です. 初等関数ではないので解けた気はしません. そこで近似値を求めてみます.

- `float(ans)` >> $\left\{ e^{-1.0 \cdot W_k(-\frac{1}{e})} \middle| \, k \in \mathbb{Z} \right\}$

ぜんぜん求まっていません. 今度は `numeric :: solve` を使ってみます.

- `numeric :: solve(eq, x)` >> {2.718281829 + 0.000000000375722512i}

`numeric :: solve(f, x)` の範囲を指定するには, x を, x= *a..b* に代えて `numeric :: solve(f, x = a..b)` と指定します. `numeric :: solve` には, Domain= *d* や `assume` は効きません. x を $0 \leqq x \leqq 5$ に制限してみます.

- `numeric :: solve(eq, x = 0..5)` >> {2.718281829}
- `float(E)` >> 2.718281828

およそ e です．やっとできました．

次は，MuPAD が偽りの解を solve する例を考えて見ます．$\sin x = x$ の実数解は, 明らかに $x = 0$ だけです．これを MuPAD で解いて見ましょう．

- eq := sin(x) − x　　>> $\sin(x) - x$
- solve(eq, x)　　>> ϕ
- numeric :: solve(eq, x)　　>> ϕ

このように「解なし」になってしまいます．ところが x の範囲を, 指定すると正解が出ます．

- numeric :: solve(eq, x = −1..1)　　>> {0.0}

$e^x = x + 1$ も同じようなことになります．ご自分でどうぞ．

8.5　まとめ

整式の方程式, 不等式は，MuPAD は正確に解くようですが，それ以外は 正確に解けることは余りありません．ans := solve(f, x) ⟶ float(ans) でやっても，間違った解が出たり, 解けないことさえ多いです．こういう際は，numeric :: solve を使い，かつ x の範囲も適切に指定した方が，良いと思われます．範囲を適切に指定さえすれば，正解を求めることが出来そうです．もちろん，大切な問題では subs して検算したり，実数解ならば, グラフなども併用した方が良いと思われます．(ちなみに不等式では numeric :: solve は使えません．方程式の解と, グラフなどを併用することになります.)

第9章　初歩的な微積分(整式を例にとって)

極限 $\lim_{x\to a} f(x)$	`limit(f(x), x = a)`
数式 f を x に関し微分 : $f'(x)$	`diff(f(x), x)`
数式 f を $x_1, x_2, \cdots$ に関し，続けて微分	`diff(f(x), x1, x2, ···)`
数式 f を x に関し n 回続けて微分 (f の n 次導関数)	`diff(f(x), x $n)`
関数 f の微分 : $D(f)$	`D(f)` または `f'`
不定積分 $\int f(x)dx$	int(f(x),x)
定積分 $\int_a^b f(x)dx$	int(f(x),x=a .. b)

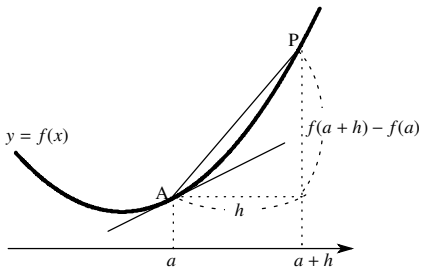
注1) `solve` と異なり, 微分と不定積分に関しては, MuPAD は，f を実数値関数と考えて計算します．積分区間の上限, 下限が実数の場合も，積分変数をその区間の実数と考えて積分します．ただし，被積分関数が，パラメーターを含んでいる場合，default では，そのパラメータは複素数となります．したがって適当に範囲を制限しないと思う結果が得られません.

9.1　極限

$\lim_{x\to a} f(x)$ は `limit(f(x), x = a)` と書きます．x は省略できません．微分の定義を復習してみましょう.

$y = f(x)$ 上に定点 A$(a, f(a))$ と動点 P$(a+h, f(a+h))$ をとると, 直線 AP の傾きは

$$\frac{f(a+h)-f(a)}{(a+h)-a} = \frac{f(a+h)-f(a)}{h}$$



ここで, 点 P を限りなく点 A に近づけると, 直線 AP の傾きは, 点 A における接線の傾きに限りなく近づくから, 点 A における接線の傾きは

$$f'(a) = \lim_{h\to 0} \frac{f(a+h)-f(a)}{h}$$

これを，$y = f(x)$ の, $x = a$ における微分係数:$f'(a)$ と呼ぶのでした．$f(x) = x^2$ のとき，$f'(a)$ を求めてみましょう.

$$f'(a) = \lim_{h\to 0} \frac{f(a+h)-f(a)}{h} = \lim_{h\to 0} \frac{(a+h)^2-a^2}{h} \qquad \cdots ①$$
$$= \lim_{h\to 0} \frac{2ah+h^2}{h} = \lim_{h\to 0}(2a+h) = 2a$$

注1) 1．diff は 'differentiate(微分する)' の略で int は 'integral(積分)' の略です.
2．diff(f,x) は f が数式 (expression) のときに使います．これに対し f が関数 (function) の時は,「D」または「 ' 」を使います.

では，MuPAD で求めてみましょう．

$f(x) = x^2$ を微分してみます．① から $f'(a)$ を求めるには, 次のように入力すればいいはずです．

- `limit(((a + h)^2 − a^2)/h, h = 0)`　　>> $2a$

確かに一致します．

9.2　微分

$f(x)$ を x に関し微分するのは，diff(f(x),x) とするだけです．

$\frac{d}{dx}(x^3 + 3x^2)$ は?　　• `diff(x^3 + 3 ∗ x^2, x)`　　>> $6x + 3x^2$

$(x^3 + 3x^2)' = 3x^2 + 6x$ ですから正しいですね．今度は，変数が 2 つ以上の場合をやってみます．

$\frac{\partial}{\partial x}(ax^3 + bx)$ は?　　• `diff(a ∗ x^3 + b ∗ x, x)`　　>> $b + 3ax^2$

$\frac{\partial}{\partial a}(ax^3 + bx)$ は?　　• `diff(a ∗ x^3 + b ∗ x, a)`　　>> x^3

2 番目の式は x, 3 番目の式は a に関し微分したので、結果が異なります．ここで $\frac{\partial}{\partial x}, \frac{\partial}{\partial a}$ は偏微分といって，それぞれ「x に関しての微分」,「a に関しての微分」を表します．2 つ以上変数があって, どちらに関して微分するかをはっきりさせたい時に使います．なお,1 変数の時でも `diff(f(x), x)` の x は省略できません．

- `diff(x^3)`　　>> x^3

注2)

注2) 【参考】diff(f,x) は f が数式 (expression) のときに使います．これに対し f が関数 (function) の時は,「D」または「 ’ 」を使います．

- `f := x− > x^3 + 3x^2`　　>> $x \longrightarrow x^3 + 3 \cdot x^2$
- `D(f)`　　>> $x \longrightarrow 6 \cdot x + 3 \cdot x^2$

このように関数になっています．`D(f)` の代わりに `f'` を使うこともできます．

- `f'`　　>> $x \longrightarrow 6 \cdot x + 3 \cdot x^2$

関数なので代入は，`g(a)` のようにできます．

`f'(1)` は?

- `D(f)(1)`　　>> 9
- `f'(1)`　　>> 9

x を代入すると x の数式 (expression) が得られます．

- `D(f)(x)`　　>> $6 \cdot x + 3 \cdot x^2$
- `f'(x)`　　>> $6 \cdot x + 3 \cdot x^2$

D を使っても，diff を使っても同じことができますが，以下の章では，`D` よりも `diff` のほうを使用します．なお，`rewrite` を使うと，`D` と `diff` を書き換えることもできます．

9.3 不定積分

$f(x)$ を x に関し不定積分するのは `int(f(x), x)` とします．微分と同様, 'x'(積分変数) は省略できません．

$\int (x^2+3)dx$ は? • `int(x^2 + 3, x)` >> $3x + \frac{x^3}{3}$

$\int (x^2+3)dx = \frac{x^3}{3} + 3x + C$ ですから正しいですね．MuPAD では、積分定数 C は出力されません．

$\int t^2 dt$ は? • `int(t^2, t)` >> $\frac{t^3}{3}$

$\int t^2 dx$ は? • `int(t^2, x)` >> $t^2\, x$

t^2 を t に関し積分すると $\frac{t^3}{3}$, x に関し積分すると $t^2\, x$ です．このように積分変数によって積分値は変わってきます.

9.4 定積分

9.4.1 基本的な定積分

$f(x)$ を x に関し $x = a$ から $x = b$ まで定積分するのは `int(f(x), x = a..b)` とします.

$\int_1^2 (x^2+3)dx$ の値は? • `int(x^2 + 3, x = 1..2)` >> $\frac{16}{3}$

実際、

$$\int_1^2 (x^2+3)dx = \left[\frac{x^3}{3} + 3x\right]_1^2 = \left(\frac{8}{3} + 6\right) - \left(\frac{1}{3} + 3\right) = \frac{16}{3}$$

なので正しいです.

積分の上限や下限は, 文字を使うこともできます.

$\int_1^x (t^2+3)dt$ の値は? • `int(t^2 + 3, t = 1..x)` >> $3 \cdot x + \frac{x^3}{3} - \frac{10}{3}$

続けて x に関し微分してみましょう.

• `diff(%, x)` >> $x^2 + 3$

実際,

$$\int_1^x (t^2+3)dt = \left[\frac{t^3}{3} + 3t\right]_1^x = \left(\frac{x^3}{3} + 3x\right) - \left(\frac{1}{3} + 3\right) = \frac{x^3}{3} + 3x - \frac{10}{3}$$

これを x で微分すると

$$\frac{d}{dx}\left(\frac{x^3}{3} + 3x - \frac{10}{3}\right) = x^2 + 3$$

となり一致します．これは

$$\text{(定理)} \qquad \frac{d}{dx}\int_a^x f(t)dt = f(x)$$

の例です．MuPAD で，この定理を確かめることも出来ます．

- `delete x, f : F := int(f(t), t = a..x)`　　>> $\int_a^x f(t)dt$
- `diff(F, x)`　　>> $f(x)$

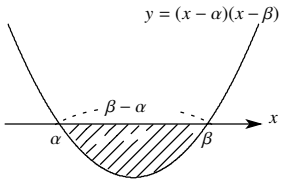
9.5　高校の問題から

9.5.1　[$\frac{1}{6}$ 公式]

公式

$$\int_\alpha^\beta (x-\alpha)(x-\beta)dx = -\frac{1}{6}(\beta-\alpha)^3$$

いわゆる $\frac{1}{6}$ 公式です．これを使うと 2 次関数と直線で囲まれた面積がすぐ求まるので重宝します．例えば, $y=(x-\alpha)(x-\beta)$ と x 軸で囲まれた面積を S とすると，



$$S = \int_\alpha^\beta \{-(x-\alpha)(x-\beta)\}\,dx = \frac{(\beta-\alpha)^3}{6}$$

となります．

まずこの公式を MuPAD を使って出してみましょう．

- `int((x − a) ∗ (x − b), x = a..b);`　　>> $a\cdot b^2 - a^2\cdot b + \frac{a^2\cdot(a+b)}{2} - \frac{b^2\cdot(a+b)}{2} - \frac{a^3}{3} + \frac{b^3}{3}$

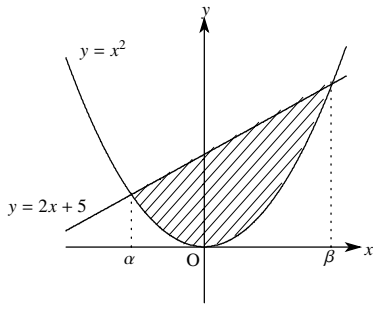
続けて因数分解します．

- factor(%);　　>> $\frac{1}{6}(a-b)^3$

確かに, 一致しました．次はこの公式の応用です．

例題

$y = x^2$ と $y = 2x + 5$ で囲まれる面積 S を求めよ.



【解答】 $x^2 = 2x + 5$ とおくと

$$x^2 - 2x - 5 = 0 \Longleftrightarrow x = 1 \pm \sqrt{6}$$

$\alpha = 1 - \sqrt{6}, \beta = 1 + \sqrt{6}$ とおくと，$x^2 - 2x - 5 = (x - \alpha)(x - \beta)$ と因数分解されるから,

$$S = \int_\alpha^\beta (2x + 5 - x^2)dx = -\int_\alpha^\beta (x^2 - 2x - 5)dx = -\int_\alpha^\beta (x - \alpha)(x - \beta)dx$$
$$= \frac{1}{6}(\beta - \alpha)^3 = \frac{1}{6}\left\{(1 + \sqrt{6}) - (1 - \sqrt{6})\right\}^3 = \frac{1}{6}(2\sqrt{6})^3 = 8\sqrt{6}$$

となります．これを公式を使わずに，MuPAD で計算してみます.

- `integrands := 2 * x + 5 - x^2` >> $2 \cdot x - x^2 + 5$
- `X := solve(integrands,x)` >> $\{\sqrt{6} + 1, 1 - \sqrt{6}\}$
- `ans := int(integrands, x = X[2]..X[1])`

  >> $(\sqrt{6} + 1)^2 - (\sqrt{6} - 1)^2 - \frac{(\sqrt{6} - 1)^3}{3} - \frac{(\sqrt{6} + 1)^3}{3} + 10 \cdot \sqrt{6}$

X は集合なので，`X[1], X[2]` でそれぞれ，X の第 1 成分 $(1 + \sqrt{6})$ と第 2 成分 $(1 - \sqrt{6})$ を取り出しています．続けて，展開します.

- `expand(ans)` >> $8 \cdot \sqrt{6}$

9.5.2 絶対値のついた積分

MuPAD では, `abs(x)` で x の絶対値: $|x|$ が求まります. [注3)]

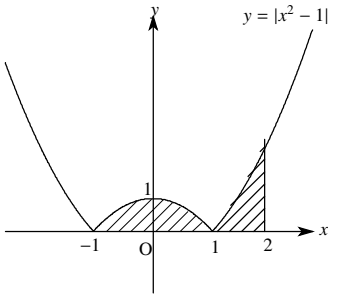
$I = \int_{-1}^{2} |x^2 - 1| dx$ を求めてみましょう.

- `int(abs(x^2 - 1)) , x = -1..2` >> $\frac{8}{3}$

注3) 例えば，• `abs(-5)` >> 5 となります.

実際，$|x^2-1| \geqq 0$ だから，I は右図の面積と一致して，

$$\begin{aligned} I &= \int_{-1}^{1} \{-(x^2-1)\}dx + \int_{1}^{2} (x^2-1)dx \\ &= \Big[-\frac{x^3}{3}+x\Big]_{-1}^{1} + \Big[\frac{x^3}{3}-x\Big]_{1}^{2} \\ &= \Big(-\frac{1}{3}+1\Big) - \Big(\frac{1}{3}-1\Big) + \Big(\frac{8}{3}-2\Big) - \Big(\frac{1}{3}-1\Big) \\ &= \frac{8}{3} \qquad \text{です.} \end{aligned}$$

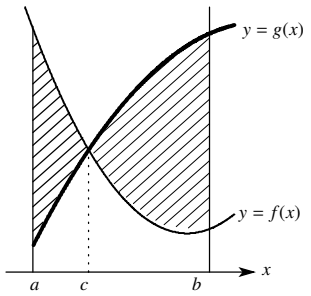


9.5.3　面積

$f(x), g(x)$ が連続関数で，$a \leqq x \leqq c < b$ で $g(x) \leqq f(x)$, $a < c \leqq x \leqq b$ で $f(x) \leqq g(x)$ のとき，$x=a, x=b, y=f(x), y=g(x)$ で囲まれた領域の面積の和を S とすると，

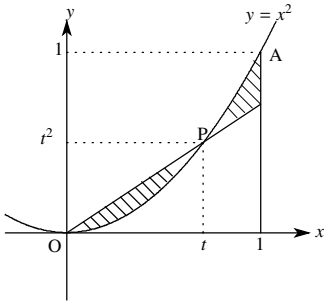
$$\begin{aligned} S &= \int_a^c \{f(x)-g(x)\}dx + \int_c^b \{g(x)-f(x)\}dx \\ &= \int_a^c |f(x)-g(x)|dx + \int_c^b |f(x)-g(x)|dx \\ &= \int_a^b |f(x)-g(x)|dx \end{aligned}$$

したがって，面積は $|f(x)-g(x)|$ の積分で与えられます.



例題

点 P は 曲線 $y=x^2$ 上を原点 O から点 A(1, 1) まで動く．このとき，直線 OP, 曲線 $y=x^2$ および直線 $x=1$ で囲まれる部分の面積 S が最小となるのは P がどの位置にあるときか. その点 P の座標, およびその最小値を求めよ.



まずは MuPAD を使わないでやってみます.

【解答】

仮定より，$P(t, t^2)$ $(0 \leqq t \leqq 1)$ とおける．このとき，直線 OP の式は $y=tx$ となるから，

$$\begin{aligned} S &= \int_0^1 |tx-x^2|dx \\ &= \int_0^t (tx-x^2)dx + \int_t^1 (x^2-tx)dx \\ &= \Big[\frac{tx^2}{2}-\frac{x^3}{3}\Big]_0^t + \Big[\frac{x^3}{3}-\frac{tx^2}{2}\Big]_t^1 \\ &= \Big(\frac{t^3}{2}-\frac{t^3}{3}\Big) - 0 + \Big(\frac{1}{3}-\frac{t}{2}\Big) - \Big(\frac{t^3}{3}-\frac{t^3}{2}\Big) \\ &= \frac{t^3}{3}-\frac{t}{2}+\frac{1}{3} \end{aligned}$$

よって

$$\frac{d}{dt}S = t^2 - \frac{1}{2} = \left(t - \frac{1}{\sqrt{2}}\right)\left(t + \frac{1}{\sqrt{2}}\right).$$

t	0	$\cdots$	$\frac{1}{\sqrt{2}}$	$\cdots$	1
$S'(t)$		−	0	+	
$S(t)$	$\frac{1}{3}$	↘	極小	↗	$\frac{1}{6}$

表より，S が最小になる t の値は

$$t = \frac{1}{\sqrt{2}}$$

このとき，点 P の座標は

$$P\left(\frac{1}{\sqrt{2}}, \frac{1}{2}\right) \quad \cdots(答)$$

また最小値は

$$S\left(\frac{\sqrt{2}}{2}\right) = \frac{1}{3} - \frac{\sqrt{2}}{6} \quad \cdots(答)$$

次は MuPAD で解いてみます．パラメータを含む絶対値付きの関数の積分では，先ずパラメータの範囲を制限します．さもないと式は簡単になりません．(後述）

まず，t の変域を $0 < t < 1$ にします.

- `assume(0 < t < 1)` >> $(0, 1)$

絶対値をつけて積分します.

- `S := int(abs(t * x - x^2)), x = 0..1)` >> $\frac{t^3}{3} - \frac{t}{2} + \frac{1}{3}$

前の $S(t)$ の結果と一致します．t に関して微分して

- diff(S,t) >> $t^2 - \frac{1}{2}$

$S'(t) = 0$ の解 ans を求めます.

- `ans := solve(%, t)` >> $\left\{\frac{\sqrt{2}}{2}\right\}$

次は最小値を求めます．まず ans(これは { } で囲まれているので，集合です) から, その 1 番めの要素 (operand) を `op(ans, 1)` で取り出して，$t0$ とおきます.

- `t0 := op(ans, 1)` >> $\frac{\sqrt{2}}{2}$

次に，S に $t = t0$ を代入 (`subs`) します.

- `subs(S, t = t0)` >> $\frac{1}{3} - \frac{\sqrt{2}}{6}$

注4) 本当は，増減表を描かないと，$t = \frac{\sqrt{2}}{2}$ で最小値をとるとはいえません．しかし, 検算には使えます. このように，MuPAD は微積分に関しては, 非常に強力な武器になります.

9.6　パラメーターを含んだ積分

9.6.1　被積分関数がパラメーターを含む時

被積分関数が，パラメーターを含んでいる場合，Default では，そのパラメータは複素数となります．したがって適当に範囲を制限しないと思う結果が得られません.

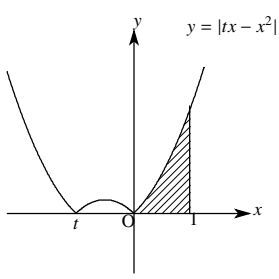
例題

t を実数とするとき

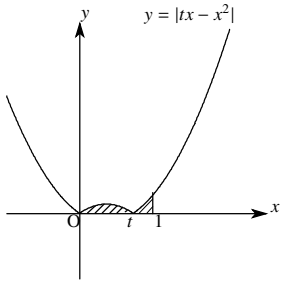
$$f(t) = \int_0^1 |tx - x^2| dx$$

を簡単にせよ.

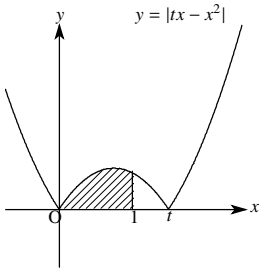
まず，MuPAD を使わずにやってみます.



$t \leqq 0$ のとき



$0 < t < 1$ のとき



$t \geqq 1$ のとき

(i) $t \leqq 0$ のとき,

$$f(t) = \int_0^1 |tx - x^2| dx = \int_0^1 \left\{-(tx - x^2)\right\} dx = \left[-\frac{tx^2}{2} + \frac{x^3}{3}\right]_0^1 = -\frac{t}{2} + \frac{1}{3}$$

注4) MuPAD の数式は，いくつかの要素 (building block) から成っていてそれを *operand* といいます．op(f) は，f の *operand* を全て取り出すコマンドで,op(f, i) は i 番目の要素を取り出すコマンドです．例えば

- a := {1, sqrt(2), 1/2}　　>> $\left\{1, \frac{1}{2}, \sqrt{2}\right\}$
- op(a)　　>> $1, \sqrt{2}, \frac{1}{2}$
- op(a, 1); op(a, 2); op(a, 3)　　>> 1　>> $\sqrt{2}$　>> $\frac{1}{2}$

となります.

(ii) $0 < t < 1$ のとき,

$$\begin{aligned} f(t) &= \int_0^1 |tx - x^2| dx \\ &= \int_0^t (tx - x^2) dx + \int_t^1 (x^2 - tx) dx \\ &= \left[\frac{tx^2}{2} - \frac{x^3}{3}\right]_0^t + \left[\frac{x^3}{3} - \frac{tx^2}{2}\right]_t^1 \\ &= \left(\frac{t^3}{2} - \frac{t^3}{3}\right) - 0 + \left(\frac{1}{3} - \frac{t}{2}\right) - \left(\frac{t^3}{3} - \frac{t^3}{2}\right) \\ &= \frac{t^3}{3} - \frac{t}{2} + \frac{1}{3} \end{aligned}$$

(iii) $t \geqq 1$ のとき,

$$f(t) = \int_0^1 |tx - x^2| dx = \int_0^1 (tx - x^2) dx = \left[\frac{tx^2}{2} - \frac{x^3}{3}\right]_0^1 = \frac{t}{2} - \frac{1}{3}$$

次に，MuPAD でやってみます.

(i) $t \leqq 0$ のとき

- `assume(t < 0)` >> $(-\infty, 0)$
- `int(abs(t * x - x^2), x = 0..1)` >> $\frac{1}{3} - \frac{t}{2}$

(ii) $0 < t < 1$ のとき

- `assume(0 < t < 1)` >> $(0, 1)$
- `int(abs(t * x - x^2), x = 0..1)` >> $\frac{t^3}{3} - \frac{t}{2} + \frac{1}{3}$

(iii) $t \geqq 1$ のとき

- `assume(t > 1)` >> $(1, \infty)$
- `int(abs(t * x - x^2), x = 0..1)` >> $-\frac{1}{3} + \frac{t}{2}$

うまく行きました. しかし, `assume(t < 0)` や `assume(t > 1)` のかわりに,`assume(t <= 0)` や `assume(t >= 1)` とすると，うまく行きません.

- `assume(t <= 0)` >> $(-\infty, 0]$
- `int(abs(t * x - x^2), x = 0..1)` >> $\int_0^1 |t \cdot x - x^2| dx$

ご覧のように，結果は，積分記号のままです. MuPAD は，積分ができない時には $\int$ 記号のままを返してきます. 本当は $t \leqq 0$ の時でも, 積分はできるはずなのですが，MuPAD は, うまくやってくれません. したがって，このような時は,「<=」や「>=」の代わりに「<」や「>」を使った方が良いでしょう.

9.6.2　積分区間がパラメーターを含む時

積分区間の方にパラメーターが入っているも，範囲を制限しないと思う結果が得られないことが良くあります.

$a \geqq 1$ のときは

$$\int_1^a |x^2-1|dx = \int_1^a (x^2-1)dx = \frac{a^3}{3} - a + \frac{2}{3}$$

ですが，$a < 1$ のときは, 違った式になります．したがって MuPAD でやるときは，a の範囲を指定しないとうまく行きません.

- `assume(a > 1)` >> $(1, \infty)$
- `int(abs(x^2 - 1), x = 1..a)` >> $\frac{a^3}{3} - a - \frac{2}{3}$

ここで a の範囲を指定しないと, 答えが出ません.

- `delete a` >>
- `int(abs(x^2 - 1), x = 1..a)` >> $\frac{a^3 \cdot \mathtt{sign}(a^2-1)}{3} - a \cdot \mathtt{sign}(a^2-1) - \frac{2}{3}$

ここで，`sign(x)` は，次のような関数です.

$$\mathtt{sign(x)} = \begin{cases} \frac{x}{|x|} & (x \text{ が } 0 \text{ 以外の複素数のとき}) \\ 0 & (x = 0 \text{ のとき}) \end{cases}$$

特に，x が実数のときは

$$\mathtt{sign(x)} = \begin{cases} 1 & (x > 0 \text{ のとき}) \\ 0 & (x = 0 \text{ のとき}) \\ -1 & (x < 0 \text{ のとき}) \end{cases}$$

となります．よって，$a > 1$ のときは，$\mathtt{sign(a^2 - 1)} = 1$ となるので,

$$\frac{a^3 \cdot \mathtt{sign}(a^2-1)}{3} - a \cdot \mathtt{sign}(a^2-1) - \frac{2}{3} = \frac{a^3 \cdot 1}{3} - a \cdot 1 - \frac{2}{3} = \frac{a^3}{3} - a - \frac{2}{3}$$

と簡単にできます．しかし `delete a` とすると,$\mathtt{sign(a^2 - 1)}$ の値が決まらないので，そのままです.

第10章　微積分(応用編)

微積分

数式 f を x に関し微分 $f'(x)$	diff(f(x), x)
数式 f を $x_1, x_2, \cdots$ に関し，続けて微分	diff(f(x), x_1, x_2, $\cdots$)
数式 f を x に関し n 回続けて微分 (f の n 次導関数)	diff(f(x), x $n)
関数 f の微分 $D(f)$	D(f) または f′
不定積分 $\int f(x)dx$	int(f(x),x)
定積分 $\int_a^b f(x)dx$	int(f(x),x=a .. b)

注1)

関数表 (再掲)

$\sin x$	sin(x)
$\cos x$	cos(x)
$\tan x$	tan(x)
a^x	_power(a, x) または a^x
e^x	exp(x) または E^x
$\log_a x$	log(a, x)
$\log x$(自然対数)	ln(x)
π(円周率)	PI
e(自然定数の底)	E
∞(無限大)	infinity

極限

$\lim_{n\to\infty} a_n$	limit(a(n), n = infinity)
$\lim_{x\to a} f(x)$	limit(f(x), x = a)
$\lim_{x\to a+0} f(x)$	limit(f(x), x = a, Right)
$\lim_{x\to a-0} f(x)$	limit(f(x), x = a, Left)

注2)

式の変形 (再掲)

f を x に関し整理	collect(f, x)
f を展開	expand(f)
因数分解	factor(f)
指数をまとめる	combine(f)
既約分数にする (結果は展開する)	normal(f)
f を部分分数に展開	partfrac(f)
f を *target* の関数を使って書き直す	rewrite(f, target)
単純化	simplify(f) または Simplify(f)

solve と異なり，default では，不定積分に関しては,f を実数値関数とみなします．定積分では，上端, 下端が実数のとき，やはり，f を実数値関数とみなします．しかし，パラメーターを含んでいる場合，default

注1) int の *option* は,*PrincipalValue*(Caushy の主値) と *Continuous*(連続と仮定して積分) の 2 つがあります.
なお，diff には，*option* はありません.
注2) a には, infinity, −infinity も使える.

では，そのパラメータは複素数となります．したがって適当に範囲を制限しないと思う結果が得られません．これは，整式の微積分と同じです．

10.1　極限

10.1.1　数列の極限

∞(無限大) は infinity, −∞(無限小) は −infinity です．

$\lim_{n\to\infty}\frac{2n-1}{3n-2}$ は?	• limit((2 * n − 1)/(3 * n − 2), n = infinity)	>> $\frac{2}{3}$
$\lim_{n\to\infty}\frac{2n^2-1}{3n-2}$ は?	• limit((2 * n^2 − 1)/(3 * n − 2), n = infinity)	>> ∞
$\lim_{n\to\infty}\frac{-2n^2-1}{3n-2}$ は?	• limit((−2 * n^2 − 1)/(3 * n − 2), n = infinity)	>> −∞
$\lim_{n\to\infty}\left(n+\frac{1}{n}\right)^n$ は?	[*Pro*] • limit((1 + 1/n)^n, n = infinity)	>> e
	[*Light*] • limit((1 + 1/n)^n, n = infinity)	>> exp(1)

最後の結果は e の定義式;

$$e=\lim_{n\to\infty}\left(n+\frac{1}{n}\right)^n$$

を表しています．

10.1.2　無限級数の和

無限級数の和も数列の極限と同様です．部分和を S_n とするとき無限級数の和は $\lim_{n\to\infty} S_n$ なので，部分和を求めた後，極限をとります．$\{a_k\}$ の第 n 項までの和は sum(a(k), k = 1..n) です．[注3] sum を使って部分和を求めてから, 極限をとります．例えば、

$$S=1+\frac{1}{2}+\left(\frac{1}{2}\right)^2+\cdots+\left(\frac{1}{2}\right)^{n-1}+\cdots \qquad \cdots(*)$$

を求めてみます．

n 項までの和: $S_n=\sum_{k=1}^{n}\left(\frac{1}{2}\right)^{k-1}$ だから, S_n は

• sum((1/2)^(k − 1), k = 1..n)　　>> $2-2\left(\frac{1}{2}\right)^n$

この極限をとります

• limit(%, n = infinity)　　>> 2

注4)

[注3] 例えば sum(k^2,k=1..3) は $\sum_{k=1}^{3}k^2=1^2+2^2+3^2=14$ を表します．

[注4] 実際、(∗) は初項 1, 公比 $\frac{1}{2}$ の等比数列の和だから n 項までの和 S_n は,

$$S_n=\frac{1\left\{1-\left(\frac{1}{2}\right)^n\right\}}{1-\frac{1}{2}}=2-2\left(\frac{1}{2}\right)^n$$

10.1.3 関数の極限

関数の極限も, 数列の極限と全く同じです. $\lim_{x\to a} f(x)$ は `limit(f(x), x = a)` と入力します.

三角関数の極限です.

$\lim_{x\to 0} \frac{\sin x}{x}$ は? • `limit(sin(x)/x, x = 0)` >> 1

続けて少し複雑なのをやってみます.

$\lim_{x\to \frac{\pi}{2}} \frac{\cos^2(x)}{1-\sin x}$ は? • `limit(cos(x)^2/(1 - sin(x)), x = PI/2)` >> 2

注5)

次は指数・対数関数の極限です.

$\lim_{h\to 0} \frac{e^h - 1}{h}$ は? • `limit((E^h - 1)/h, h = 0)` >> 1

$\lim_{t\to 0} \frac{\log(1+t)}{t}$ は? • `limit(ln(1 + t)/t, t = 0)` >> 1

$\lim_{t\to 0} (1+t)^{\frac{1}{t}}$ は? • `limit((1 + t)^(1/t), t = 0)` >> e

$\lim_{x\to \infty} \left(x + \frac{1}{x}\right)^x$ は? • `limit((1 + 1/x)^x, x = infinity)` >> e

$\lim_{x\to -\infty} \left(x + \frac{1}{x}\right)^x$ は? • `limit((1 + 1/x)^x, x = -infinity)` >> e

これは

$$\lim_{h\to 0} \frac{e^h - 1}{h} = 1 \Longleftrightarrow \lim_{t\to 0} \frac{\log(1+t)}{t} = 1 \Longleftrightarrow \lim_{t\to 0} (1+t)^{\frac{1}{t}} = e \Longleftrightarrow \lim_{x\to \pm\infty} \left(x + \frac{1}{x}\right)^x = e$$

を表しています.

したがって,(∗) の和は,

$$\lim_{n\to\infty} S_n = 2$$

で一致します.

注5) 実際、$\frac{\pi}{2} - x = t$ とおくと, $x \to \frac{\pi}{2}$ のとき $t \to 0$ だから、

$$\lim_{x\to \frac{\pi}{2}} \frac{\cos^2(x)}{1-\sin x} = \lim_{t\to 0} \frac{\cos^2\left(\frac{\pi}{2} - t\right)}{1 - \sin\left(\frac{\pi}{2} - t\right)} = \lim_{t\to 0} \frac{\sin^2 t}{1-\cos t} = \lim_{t\to 0} \frac{1-\cos^2 t}{1-\cos t} = \lim_{t\to 0} (1+\cos t) = 2$$

10.1.4　右極限・左極限

右極限, 左極限はそれぞれ $\lim_{x\to a+0} f(x)$, $\lim_{x\to a-0} f(x)$ とかかれ，それぞれ,「x が a より大きいほうから a に近づく (右から近づく) ときの極限」と「x が a より小さいほうから a に近づく (左から近づく) ときの極限」を表します．MuPAD では， option で，それぞれ，Right と Left を指定します．
では，$f(x) = \frac{x}{x-1}$ の右極限と左極限を見てみます．

$\lim_{x\to 1+0} \frac{x}{x-1}$ は?	• `limit(x/(x - 1), x = 1, Right)`	∞
$\lim_{x\to 1-0} \frac{x}{x-1}$ は?	• `limit(x/(x - 1), x = 1, Left)`	$-\infty$
$\lim_{x\to 1} \frac{x}{x-1}$ は?	• `limit(x/(x - 1), x = 1)`	undefined

実際, $x > 1$ のとき, $\frac{x}{x-1} > 0$,　$x < 1$ のとき, $\frac{x}{x-1} < 0$ で, かつ $x \to 1$ のとき, 分母の絶対値は限りなく 0 に近づくからから

$$\lim_{x\to 1+0} \frac{x}{x-1} = \infty, \qquad \lim_{x\to 1-0} \frac{x}{x-1} = -\infty$$

しかし左極限と、右極限が異なるので $\lim_{x\to 1} \frac{x}{x-1}$ は定義できません．

10.2　微分

10.2.1　微分, 偏微分

関数 f を x に関し微分するのは，`diff(f,x)` とするだけです．

$\frac{d}{dx}\sin x = (\sin x)'$は?	•`diff(sin(x), x)`	>> $\cos(x)$
$\frac{d}{dx}\cos x = (\cos x)'$は?	•`diff(cos(x), x)`	>> $-\sin(x)$
$\frac{d}{dx}e^x = (e^x)'$は?	•`diff(exp(x), x)`	>> e^x
$\frac{d}{dx}\log x = (\log x)'$は?	•`diff(ln(x), x)`	>> $\frac{1}{x}$
$\frac{d}{dx}a^x = (a^x)'$は?	•`diff(a^x, x)`	>> $a^x \cdot \mathtt{ln}(a)$
$\frac{d}{dx}\sqrt{x} = (\sqrt{x})'$は?	•`diff(sqrt(x), x)`	>> $\frac{1}{2\cdot\sqrt{x}}$

2 つ以上変数があるときも同じです．

$\frac{\partial}{\partial x}(x^y)$ は?	•`diff(x^y)`	>> $x^{y-1}\cdot y$
$\frac{\partial}{\partial y}(x^y)$ は?	•`diff(x^y)`	>> $x^y \cdot \mathtt{ln}(x)$

上の式は x, 下の式は y に関し微分したので、結果が異なります．上の式は，$\frac{d}{dx}(x^n) = nx^{n-1}$ と，下の式は，$\frac{d}{dy}(a^y) = a^y \log y$ と同じです．ここで $\frac{\partial}{\partial x}, \frac{\partial}{\partial y}$ は偏微分といって，それぞれ「x に関しての微分」,「y に関しての微分」を表します．2 つ以上変数があって, どちらに関して微分するかをはっきりさせたい時に使います．

$x_1, x_2, \cdots$ に関し続けて微分する時は diff(f, x1, x2, ⋯) とします.

$\frac{\partial^2}{\partial y \partial x}(x^y) = \frac{\partial}{\partial y}\left\{\frac{\partial}{\partial x}(x^y)\right\}$ は?

- diff(x^y, x, y) >> $x^{y-1} + x^{y-1} \cdot y \cdot \ln(x)$

$\frac{\partial^2}{\partial x \partial y}(x^y) = \frac{\partial}{\partial x}\left\{\frac{\partial}{\partial y}(x^y)\right\}$ は?

- diff(x^y, y, x) >> $\frac{x^y}{x} + x^{y-1} \cdot y \cdot \ln(x)$

このように $f(x, y) = x^y$ のときは, $\frac{\partial^2 f}{\partial y \partial x} = \frac{\partial^2 f}{\partial x \partial y}$ が成り立っています.

10.2.2 高階微分

$x_1, x_2, \cdots$ に関し続けて微分する時は diff(f, x1, x2, ⋯) とするので, $f(x)$ を, x に関して 2 回微分するには, diff(f, x, x), 3 回微分するには, diff(f, x, x, x) とします. また 列生成子「$」 を使うと, 2 階微分, 3 階微分はそれぞれ diff(f, x $2), diff(f, x $3) とすることもできます. [注6)]

$f(x) = x^3$ の 2 階微分,3 階微分を求めてみます.

$f''(x)$ は?	• diff(x^3, x $2)	$6x$
$f'''(x)$ は?	• diff(x^3, x $3)	6

4 階以上の微分も同様です.

10.2.3 いろいろな関数の微分

いろいろな関数を微分してみます. 微分そのものは簡単ですが, その後の式をきれいにするほうが大変です. 出てきた式を変形するには, factor(因数分解), expand(展開), normal(通分), combine(指数をまとめる), rewrite(書き換え), simplify(簡略化), Simplify(簡略化) 等を使います. (第 4 章参照)

◇ $\left(\frac{7x-6}{x^2+1}\right)'$ を求めてみます.

- diff((7 * x − 6)/(x^2 + 1), x) >> $\frac{7}{x^2+1} - \frac{2 \cdot x(7 \cdot x - 6)}{(x^2+1)^2}$

注6) $ は列生成子 (sequence generator) と呼ばれ, 数列を簡単に表すことができます. 例えば

- k^2 $k = 1..4 >> 1, 4, 9, 16
- k^2 $k = 2..5 >> 4, 9, 16, 25
- k^2 $i = 1..3 >> k^2, k^2, k^2

変数を省略すると, 3 番目の式と同じになります.

- k^2 $3 >> k^2, k^2, k^2

通分しましょう．続けて、normal(%) と打ちます.

- normal(%) >> $\dfrac{12\cdot x-7\cdot x^2+7}{2\cdot x^2+x^4+1}$

因数分解された形が欲しい時は，factor を利用します.

- factor(%) >> $-\dfrac{-12\cdot x+7\cdot x^2-7}{(x^2+1)^2}$

このように factor は, 通分と因数分解をともに実行します．一方，normal は, 通分はしますが, 因数分解はしません (約分はどちらもします．)

◇ $\left(\frac{1-\sin x}{1+\sin x}\right)'$ を求めてみます.

- diff((1 + sin(x))/(1 − sin(x)), x) >> $\dfrac{\cos(x)\cdot(\sin(x)-1)}{(\sin(x)+1)^2}-\dfrac{\cos(x)}{\sin(x)+1}$

factor で, 通分と因数分解を同時にします.

- factor(%) >> $-\dfrac{2\cdot\cos(x)}{(\sin(x)+1)^2}$

注7)

10.3　積分

10.3.1　不定積分

f を x に関し不定積分するのは int(f, x) とします．ただし，MuPAD では積分定数 C は表示されません．また，default では，積分変数は実数とみなされます．ただし, パラメーターは複素数とみなされるので, 原則として,assume を使って，自分で範囲を指定する必要があります.

$\int e^x dx$ は?	• int(exp(x), x)	>> e^x
$\int \frac{1}{x} dx$ は?	• int(1/x, x)	>> ln(x)
$\int \sin x dx$ は?	• int(sin(x), x)	>> $-\cos(x)$
$\int \cos x dx$ は?	• int(cos(x), x)	>> sin(x)
$\int \frac{1}{\cos^2 x} dx$ は?	• int(1/cos(x)^2, x)	>> $\dfrac{\sin(2\cdot x)}{\cos(2\cdot x)+1}$

注7) 結果を確かめて見ます.

$$\begin{aligned}\left(\frac{1-\sin x}{1+\sin x}\right)' &= \frac{(1-\sin x)'(1+\sin x)-(1-\sin x)(1+\sin x)'}{(1+\sin x)^2}\\ &= \frac{-\cos x(1+\sin x)-(1-\sin x)\cos x}{(1+\sin x)^2}\\ &= -\frac{2\cos x}{(\sin x+1)^2}\end{aligned}$$

最後の式は, $\int \frac{1}{\cos^2 x} dx = \tan x$ となる筈なので、$\tan x$ を使って書き直します．このように tan の式に書き直すには，`rewrite(f, tan)` を使います．

- `rewrite(%, tan)` >> $-\dfrac{2 \cdot \tan(x)}{(\tan(x)^2+1) \cdot \left(\dfrac{\tan(x)^2-1}{\tan(x)^2+1} - 1\right)}$

`normal` を使って約分します．

- `normal(%)` >> $\tan(x)$

やっと一致しました．不定積分も微分と同様，それ自体は簡単ですが，出てきた式をきれいにする方が大変です．次はやや複雑な式の不定積分をやってみます．

$\int \cos^2 x dx$ は?	• `int(cos(x)^2, x)`	>> $\dfrac{x}{2} + \dfrac{\sin(2x)}{4}$
$\int \sin 2x \cos x dx$ は?	• `int(sin(2 * x) * cos(x), x)`	>> $-\dfrac{\cos(x)}{2} - \dfrac{\cos(3x)}{6}$

注8)

10.3.2 定積分

これは答えを簡単にする必要が余りないので, むしろ不定積分より簡単です．$\int_a^b f(x)dx$ は，`int(f, x = a .. b)` のように指定します．

$\int_0^{\pi} \sin x dx$ は?	• `int(sin(x), x = 0 ..PI)`	>> 2
$\int_0^{\frac{\pi}{2}} \cos x dx$ は?	• `int(cos(x), x = 0 ..PI/2)`	>> 1
$\int_0^1 e^x dx$ は?	• `int(exp(x), x = 0 ..1)`	>> $e - 1$
$\int_1^2 \dfrac{1}{x} dx$ は?	• `int(1/x, x = 1 ..2)`	>> `ln(2)`
$\int_6^8 \dfrac{x}{x^2 - 6x + 8} dx$ は?	• `int(x/(x^2 - 6 * x + 8), x = 6..8)`	>> `3ln(4) - 2ln(2) - ln(6)`
1 つにまとめると?	• `combine(%, ln)`	>> $-\texttt{ln}\dfrac{3}{8}$

注8) 実際、$\cos^2 x = \frac{1+\cos 2x}{2}$ だから

$$\int \cos^2 x dx = \int \frac{1+\cos 2x}{2} dx = \frac{x}{2} + \frac{\sin 2x}{4} + C$$

また、積和の公式より $\sin 2x \cos x = \frac{1}{2}\{\sin(2x+x) + \sin(2x-x)\} = \frac{1}{2}(\sin 3x + \sin x)$ だから

$$\int \sin 2x \cos x dx = \int \frac{1}{2}(\sin 3x + \sin x) dx = -\frac{\cos 3x}{6} - \frac{\cos x}{2} + C$$

です．

このように log の式を, 一つにまとめるには，`combine(f, ln)` を使います. [注9)]

MuPAD は 置換積分，部分積分が必要な積分も簡単に解きます.

$\int_1^e \frac{\sin(\pi \log x)}{x} dx$ は?　　• `int(sin(PI * ln(x))/x, x = 1..E)`　　>> $\frac{2}{\pi}$

$\int_1^e x^2 \log x dx$ は?　　• `int(x^2 * ln(x), x = 1..E)`　　>> $\frac{2 \cdot e^3}{9} + \frac{1}{9}$

注10)

10.3.3　広義積分

MuPAD はいわゆる広義積分もこなします.

$\int_1^\infty \frac{1}{x^2}$ は?

• `int(1/x^2, x = 1..infinity)`　　>> 1

$\int_0^1 \frac{1}{\sqrt{x}}$ は?

• `int(1/sqrt(x), x = 0..1)`　　>> 2

実際，$a > 0$ のとき

$$\int_1^a \frac{1}{x^2} dx = \left[-\frac{1}{x} \right]_1^a = -\frac{1}{a} + 1$$
$$\int_a^1 \frac{1}{\sqrt{x}} = \left[2\sqrt{x} \right]_a^1 = 2 - 2\sqrt{a}$$

よって

$$\int_1^\infty \frac{1}{x^2} = \lim_{a \to \infty} \left(-\frac{1}{a} + 1 \right) = 1$$
$$\int_0^1 \frac{1}{\sqrt{x}} = \lim_{a \to +0} \left(2 - 2\sqrt{a} \right) = 2$$

注9)

$$\begin{aligned} \int_6^8 \frac{x}{x^2 - 6x + 8} dx &= \int_6^8 \left(\frac{2}{x-4} - \frac{1}{x-2} \right) dx = 2\Big[\log|x-4| \Big]_6^8 - \Big[\log|x-2| \Big]_6^8 \\ &= 2(\log 4 - \log 2) - (\log 6 - \log 4) \\ &= 2\log 2 - (\log 2 + \log 3) + 2\log 2 = 3\log 2 - \log 3 = \log \tfrac{8}{3} = -\log \tfrac{3}{8} \end{aligned}$$

注10)

確かめてみます.　$\pi \log x = t$ とおくと, $\frac{\pi}{x} dx = dt$, $\begin{array}{c|ccc} x & 1 & \to & e \\ \hline t & 0 & \to & \pi \end{array}$

$$\int_1^e \frac{\sin(\pi \log x)}{x} dx = \frac{1}{\pi} \int_0^\pi \sin t dt = \frac{1}{\pi} \Big[-\cos t \Big]_0^\pi = \frac{2}{\pi}$$

また,

$$\int_1^e x^2 \log x dx = \int_1^e \left(\frac{x^3}{3} \right)' \log x dx = \Big[\frac{x^3}{3} \log x \Big]_1^e - \int_1^e \frac{x^3}{3} \cdot \frac{1}{x} dx = \frac{e^3}{3} - \Big[\frac{x^3}{9} \Big]_1^e = \frac{2e^3}{9} + \frac{1}{9}$$

一般の広義積分が求まらなくとも，いわゆる *Caushy* の主値 (v.p.) は求まることがあります．それには option で *PrincipalValue* を指定します．

$\int_{-1}^{1} \frac{1}{x} dx$ は?

- `int(1/x, x = -1..1)` >> undifined

v.p. $\int_{-1}^{1} \frac{1}{x} dx$ は?

- `int(1/x, x = -1..1, PrincipalValue)` >> 0

注11)

10.3.4 絶対値のついた積分

x の絶対値：$|x|$ は MuPAD では `abs(x)` と表します．しかし，MuPAD は整式の場合と異なり, いつでも正しい答えを与えるわけではありません．

$\int_0^{\pi} |\cos x| dx$ は? • `int(abs(cos(x)), x = 0..PI)` >> 2

$\int_0^{2\pi} |\sin x + \cos x| dx$ は? • `int(abs(sin(x) + cos(x)), x = 0..2 * PI)` >> $4 \cdot \sqrt{2}$

$\int_0^{2\pi} |3\sin x + 2\cos x| dx$ は? • `int(abs(3 * sin(x) + 2 * cos(x)), x = 0..2 * PI)` >> 0

実際やってみると、

$$\int_0^{\pi} |\cos x| dx = \int_0^{\frac{\pi}{2}} \cos x dx + \int_{\frac{\pi}{2}}^{\pi} (-\cos x) dx = \Big[\sin x \Big]_0^{\frac{\pi}{2}} - \Big[\sin x \Big]_{\frac{\pi}{2}}^{\pi} = 2$$

2 番目の積分です．$\sin x + \cos x = \sqrt{2} \sin\left(x + \frac{\pi}{4}\right)$ だから, $x + \frac{\pi}{4} = t$ とおくと，$dx = dt$, $\begin{array}{c|ccc} x & 0 & \to & 2\pi \\ \hline t & \frac{\pi}{4} & \to & 2\pi + \frac{\pi}{4} \end{array}$

$$\therefore \quad \int_0^{2\pi} |\sin x + \cos x| dx = \int_{\frac{\pi}{4}}^{2\pi + \frac{\pi}{4}} \sqrt{2} |\sin t| dt$$

注11)

$a > 0, b < 0$ のとき

$$\int_a^1 \frac{1}{x} dx = [\log |x|]_a^1 = -\log a, \quad \int_{-1}^b \frac{1}{x} dx = [\log |x|]_{-1}^b = \log |b| = \log(-b)$$

よって,

$$\int_{-1}^1 \frac{1}{x} dx = \int_{-1}^0 \frac{1}{x} dx + \int_0^1 \frac{1}{x} dx = \lim_{b \to -0} \log(-b) + \lim_{a \to +0} (-\log a) = -\infty + \infty$$

となり，$\int_{-1}^1 \frac{1}{x} dx$ は存在しません．しかし $a = -b$ を保ちながら，a, b が 0 に近づくときは，値 (v.p.) が存在します．

$$\int_{-1}^{-a} \frac{1}{x} dx + \int_a^1 \frac{1}{x} dx = [\log |x|]_{-1}^{-a} + [\log |x|]_a^1 = \log a - \log 1 + \log 1 - \log a = 0$$

ところが $\sin t$ の周期は 2π だから

$$\int_{\frac{\pi}{4}}^{2\pi+\frac{\pi}{4}} \sqrt{2}|\sin t|dt = \sqrt{2}\int_0^{2\pi} |\sin t|dt = 2\sqrt{2}\int_0^{\pi} \sin t dt = 2\sqrt{2}\Big[-\cos t\Big]_0^{\pi} = 4\sqrt{2}$$

となり，始めの二つは一致しますが，明らかに最後の積分は間違っています．(2 番目の積分と同様に計算すると, $4\sqrt{13}$ になるはずです．) [注12)] したがって, 絶対値がついている積分においては，結果を確かめたほうが良いと思われます.

10.3.5　数値積分

積分できない (厳密な値の求まらない) 場合は，`int` のかわりに，`numeric :: int` を使った方が早く, また, 大きく外れることはないようです．[注13)] 先の，$I = \int_0^{2\pi} |3\sin x + 2\cos x|dx$ を求めてみます.

- `numeric :: int(abs(3 * sin(x) + 2 * cos(x)), x = 0..2 * PI)`　　>> 14.4222051
- `float(4 * sqrt(13))`　　>> 14.4222051

`solve` だと,「$I = 0$」 という大幅に外れた答えになりましたが，`numeric :: solve` を使うと，数値解ながら，正確な値（$4\sqrt{13}$）に近い値が求まります．このように，MuPAD では, 正確な値が出ないときに, `numeric :: int` はとても有効です.

$I = \int_0^{\frac{\pi}{2}} \frac{\sin x}{\sin x + \cos x}dx$ の値を求めてみます.

- `int(sin(x)/(sin(x) + cos(x)), x = 0..PI/2)`　　>> $\int_0^{\frac{\pi}{2}} \frac{\sin(x)}{\sin(x) + \cos(x)}dx$

このように，MuPAD では計算できません．そこで `float` で近似解を求めてみます.

- `float(%)`　　>> 0.7853981634

このように小数近似が求まりました．しかし，最初から近似解だけが欲しい時は `numeric :: int` を使った方が早いです.

- `numeric :: int(sin(x)/(sin(x) + cos(x)), x = 0..PI/2)`　　>> 0.7853981634

実は，$I = \int_0^{\frac{\pi}{2}} \frac{\sin x}{\sin x + \cos x}dx$ の値は厳密に求まります.

$J = \int_0^{\frac{\pi}{2}} \frac{\cos x}{\sin x + \cos x}dx$ とおくと

$$I + J = \int_0^{\frac{\pi}{2}} \frac{\sin x + \cos x}{\sin x + \cos x}dx = \int_0^{\frac{\pi}{2}} 1\, dx = \frac{\pi}{2} \qquad \cdots ①$$

[注12)] これは $3\sin x + 2\cos x = 0$ の解が，簡単に表せないことと関係があると思われます.
[注13)] `numeric :: int` は，`numeric :: solve` と同様，numeric lib のコマンドです.

$\frac{\pi}{2} - x = t$ と置換すると，$-dx = dt$, $\begin{array}{c|ccc} x & 0 & \to & \frac{\pi}{2} \\ \hline t & \frac{\pi}{2} & \to & 0 \end{array}$, $\left\{ \begin{array}{ll} \sin x & = \sin\left(\frac{\pi}{2} - t\right) = \cos t \\ \cos x & = \cos\left(\frac{\pi}{2} - t\right) = \sin t \end{array} \right.$ ．よって，置換積分の公式より，

$$I = \int_{\frac{\pi}{2}}^{0} \frac{\cos t}{\sin t + \cos t}(-dt) = \int_{0}^{\frac{\pi}{2}} \frac{\cos t}{\sin t + \cos t} dt = \int_{0}^{\frac{\pi}{2}} \frac{\cos x}{\sin x + \cos x} dx = J \qquad \cdots ②$$

①,② より

$$I = J = \frac{\pi}{4}$$

MuPAD で計算してみます.

- `float(PI/4)` >> 0.7853981634

確かに一致します.

10.3.6 定積分の工夫 (不定積分から求める)

先の積分:

$$I = \int_{0}^{\frac{\pi}{2}} \frac{\sin(x)}{\sin(x) + \cos(x)} dx$$

は MuPAD では厳密解は求まりませんでした．しかし不定積分を利用すると厳密解を求められます.

$f(x) = \int \frac{\sin x}{\sin x + \cos x} dx$ は?

- `f := int(sin(x)/(sin(x) + cos(x)), x)` >> $\frac{x}{2} - \frac{\ln(2 \cdot \sin(2 \cdot x) + 2)}{4}$

$I = f|_{x=\frac{\pi}{2}} - f|_{x=0}$ ですから,

- `subs(%, x = PI/2) - subs(%, x = 0)` >> $\frac{\pi}{4} + \frac{\ln(2 \cdot \sin(0) + 2)}{4} - \frac{\ln(2 \cdot \sin(\pi) + 2)}{4}$

これを，評価 (evaluate) すると?

- `eval(%)` >> $\frac{\pi}{4}$

うまく行きました．このように定積分は求まらなくとも, 不定積分が求まり, それを利用して定積分が求まられる場合もあります．[注14)] こういう例は，三角関数には，ちょくちょくあります.

$\int_{\frac{\pi}{4}}^{\frac{\pi}{3}} \frac{1}{\tan x} dx$ は?

- `int(1/tan(x), x = PI/4..PI/3)` >> $\int_{\frac{\pi}{4}}^{\frac{\pi}{3}} \frac{1}{\tan x} dx$
- `int(1/tan(x), x)` >> $\frac{\ln(2 - 2 \cdot \cos(2 \cdot x))}{2}$

注14) ここで，`subs(f, x = a)` は，x に a を代入するコマンドです．ただ, 結果はごらんのように $\sin(0) \to 0$ と, なっていません．そこで `eval` を使って, 評価します.

$\sin x$ で書き直すと

- `rewrite(%, sin)` >> $\frac{\mathtt{ln}(4 \cdot \sin(x)^2)}{2}$

実際,

$$\int \frac{1}{\tan x} dx = \int \frac{\cos x}{\sin x} dx = \int \frac{(\sin x)'}{\sin x} dx = \log|\sin x| + C = \frac{\log \sin^2 x}{2} + C$$

ですから, 不定積分はあっています．しかし, 定積分は求まりません.

同様の積分はまだあります.

- `int(1/cos(x), x = 0..PI/4)` >> $\int_0^{\frac{\pi}{4}} \frac{1}{\cos x} dx$
- `int(1/cos(x), x)` >> $\frac{\mathtt{ln}(2 \cdot \sin(x) + 2)}{2} - \frac{\mathtt{ln}(2 - 2 \cdot \sin(x))}{2}$ $\cdots$①
- `int(sin(x)^3/(sin(x) + cos(x)), x = 0..PI/2)` >> $\int_0^{\frac{\pi}{2}} \frac{\sin(x)^3}{\sin(x) + \cos(x)} dx$
- `int(sin(x)^3/(sin(x) + cos(x)), x)` >> $\frac{x}{2} - \frac{\mathtt{ln}(2 \cdot \sin(2 \cdot x) + 2)}{8} + \frac{\cos(2 \cdot x)}{8} - \frac{\sin(2 \cdot x)}{8}$ $\cdots$②

定積分：$\int_0^{\frac{\pi}{4}} \frac{1}{\cos x} dx, \int_0^{\frac{\pi}{2}} \frac{\sin(x)^3}{\sin(x)+\cos(x)} dx$ は求まりませんが，不定積分の ①,② は求まります．ちなみに, 微分すれば検算はすぐできます．例えば，② の式の検算は, ② に続けて，次のようにすればできます.

- `diff(%, x)` >> $\frac{1}{2} - \frac{\sin(2 \cdot x)}{4} - \frac{\cos(2 \cdot x)}{2 \cdot (2 \cdot \sin(2 \cdot x) + 2)} - \frac{\cos(2 \cdot x)}{4}$

これが被積分関数と一致するかのチェックは, `testeq(f, g)` で行います.

- `testeq(%, sin(x)^3/(sin(x) + cos(x)))` >> $TRUE$

ごらんのように, 合っています．また，以上の定積分も, 全て `numeric` :: `int` を使っても，かなり正確な数値解を得ることができます.

10.3.7　MuPAD の苦手な積分の計算例

手計算では，すぐ求まるが，MuPAD が正確な解をなかなか求められない例もあります．やはり, そういうのは三角関数に多いです.

- `int(cos(2 * x)/(sin(x) + cos(x)), x)` >> $\int \frac{\cos(2 \cdot x)}{\sin(x) + \cos(x)} dx$

MuPAD は積分できませんが，手計算では, 簡単に不定積分できます.

$$\int \frac{\cos 2x}{\sin x + \cos x} dx = \int \frac{\cos^2 x - \sin^2 x}{\sin x + \cos x} dx = \int (\cos x - \sin x) dx = \sin x + \cos x + C$$

これほど簡単ではないですが, 同様の例に

$$\int e^{\sqrt{x}} dx$$

があります. $\sqrt{x} = u$ とおくと, $x = u^2, dx = 2udu$

$$\therefore \int e^{\sqrt{x}}dx = \int e^u 2u\,du = \int (e^u)'2u\,du = 2ue^u - \int (2u)'\,e^u du = 2ue^u - 2e^u = 2\sqrt{x}e^{\sqrt{x}} - 2e^{\sqrt{x}}$$

しかし，MuPAD ではこれも積分できません.

•`int(exp(sqrt(x)), x)` >> $\int e^{\sqrt{x}}dx$

このような例外はあるものの, 実は非常に稀で,(以上の例は, 私が, 一日かけて探し回ったものです.) ほとんどの場合は無事に積分 (定積分, 不定積分とも）できます. どうしようもないときでも，`numeric :: int` を使えば, 大丈夫です.

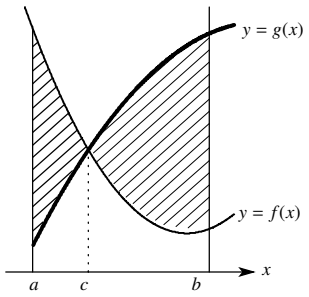
10.4 高校の数学から

10.4.1 面積

右図のように $f(x), g(x)$ が連続関数で，$a \leqq x \leqq c$ で $g(x) \leqq f(x)$, $c \leqq x \leqq b$ で $f(x) \leqq g(x)$ のとき，$x = a, x = b, y = f(x), y = g(x)$ で囲まれた領域の面積の和を S とすると,

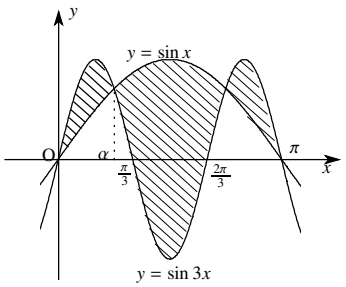
$$\begin{aligned} S &= \int_a^c \{f(x) - g(x)\}dx + \int_c^b \{g(x) - f(x)\}dx \\ &= \int_a^c |f(x) - g(x)|dx + \int_c^b |f(x) - g(x)|dx \\ &= \int_a^b |f(x) - g(x)|dx \end{aligned}$$

したがって，面積は $|f(x) - g(x)|$ の積分で与えられる.



例題 1

二つの曲線 $C_1 : y = \sin x (0 \leqq x \leqq \pi), C_2 : y = \sin 3x (0 \leqq x \leqq \pi)$ によって囲まれる面積 S を求めよ.



まず,MuPAD でやってみます. $S = \int_0^\pi |\sin 3x - \sin x|dx$ ですから次のように入力します.

• `int(abs(sin(3 * x) - sin(x)), x = 0..PI)` >> $\frac{8 \cdot \sqrt{2}}{3} - \frac{4}{3}$

次は手計算でやって見ます.

【解答】

$0 < x < \frac{\pi}{2}$ における交点の x 成分を $\alpha\ (0 < \alpha < \frac{\pi}{2})$ とおくと，3 倍角の公式より

$$\sin\alpha = \sin 3\alpha \Longleftrightarrow \sin\alpha = 3\sin\alpha - 4\sin^3\alpha \Longleftrightarrow 4\sin^3\alpha - 2\sin\alpha = 0$$
$$\Longleftrightarrow \sin\alpha\left(\sin\alpha - \frac{1}{\sqrt{2}}\right)\left(\sin\alpha + \frac{1}{\sqrt{2}}\right) = 0$$

$0 < \alpha < \frac{\pi}{2}$ だから

$$\sin\alpha = \frac{1}{\sqrt{2}} \Longleftrightarrow \alpha = \frac{\pi}{4}$$

C_1, C_2 は直線 $x = \frac{\pi}{2}$ に関し対称だから

$$\begin{aligned}\frac{S}{2} &= \int_0^{\alpha}(\sin 3x - \sin x)dx + \int_{\alpha}^{\frac{\pi}{2}}(\sin x - \sin 3x)dx \\ &= \left[-\frac{1}{3}\cos 3x + \cos x\right]_0^{\alpha} + \left[-\cos x + \frac{1}{3}\cos 3x\right]_{\alpha}^{\frac{\pi}{2}} \\ &= \left(-\frac{1}{3}\cos 3\alpha + \cos\alpha\right) - \left(-\frac{1}{3} + 1\right) + 0 - \left(\frac{1}{3}\cos 3\alpha - \cos\alpha\right) \\ &= -\frac{2}{3}\cos 3\alpha + 2\cos\alpha - \frac{2}{3} = -\frac{2}{3}\cos\frac{3\pi}{4} + 2\cos\frac{\pi}{4} - \frac{2}{3} \\ &= \frac{4\sqrt{2} - 2}{3}\end{aligned}$$

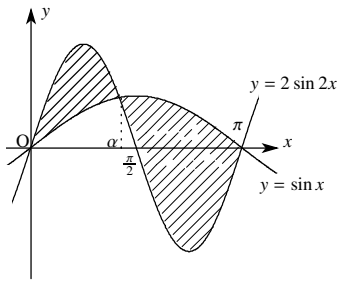
よって

$$S = 2\left(\frac{4\sqrt{2} - 2}{3}\right) = \frac{8\sqrt{2} - 4}{3}$$

一致しました．次はどうでしょう?

例題 2

二つの曲線 $C_1 : y = \sin x (0 \leqq x \leqq \pi), C_2 : y = 2\sin 2x (0 \leqq x \leqq \pi)$ によって囲まれる面積 S を求めよ．



まず,MuPAD でやってみます．$S = \int_0^{\pi}|\sin x - 2\sin 2x|dx$ ですから次のように入力します.

- `int(abs(sin(x) - 2 * sin(2 * x)), x = 0..PI)` >> 2

次は手計算でやって見ます.

【解答】 $0 < x < \pi$ における交点の x 成分を $\alpha\ (0 < \alpha < \pi)$ とおくと，倍角公式より

$$\sin\alpha = 2\sin 2\alpha \Longleftrightarrow \sin\alpha = 4\sin\alpha\cos\alpha \Longleftrightarrow \sin\alpha(1 - 4\cos\alpha) = 0 \Longleftrightarrow \cos\alpha = \frac{1}{4}$$

$$\begin{aligned}S &= \int_0^{\alpha}(2\sin 2x - \sin x)dx + \int_{\alpha}^{\pi}(\sin x - 2\sin 2x)dx \\ &= \left[-\cos 2x + \cos x\right]_0^{\alpha} + \left[-\cos x + \cos 2x\right]_{\alpha}^{\pi} \\ &= (-\cos 2\alpha + \cos\alpha) - 0 + 2 - (\cos 2\alpha - \cos\alpha) \\ &= -2\cos 2\alpha + 2\cos\alpha + 2 = -2(2\cos^2\alpha - 1) + 2\cos\alpha + 2\end{aligned}$$

$\cos\alpha = \frac{1}{4}$ だから

$$S = -4\cos^2\alpha + 2\cos\alpha + 4 = -\frac{1}{4} + \frac{1}{2} + 4 = \frac{17}{4}$$

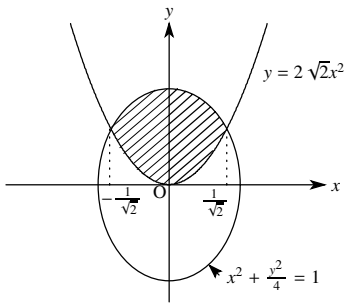
一致しません! 絶対値の付いているときは, 要注意でした．そこで，numeric :: int を使ってみます.

• numeric :: int(abs(sin(x) − 2 ∗ sin(2 ∗ x)), x = 0..PI) >> 4.25

一致しました.

例題 3

二つの不等式 $x^2 + \frac{y^2}{4} \leqq 1, y \geqq 2\sqrt{2}x^2$ で
表される領域の面積 S を求めよ.



片方が陰関数になっているので，まず，y に関して解きます.

• g := solve(x^2 + y^2/4 − 1, y) >> $\{-2\cdot\sqrt{-(x-1)\cdot(x+1)},\ 2\cdot\sqrt{-(x-1)\cdot(x+1)}\}$

この第 2 成分を g[2] で取り出し，$y = g[2] = 2\cdot\sqrt{-(x-1)\cdot(x+1)}$ と，放物線の交点を求めます.

• X := solve({y = g[2], y = 2 ∗ sqrt(2) ∗ x^2}, [x, y])

$$>> \left\{\left[x = -\frac{\sqrt{2}}{2}, y = \sqrt{2}\right], \left[x = \frac{\sqrt{2}}{2}, y = \sqrt{2}\right]\right\}$$

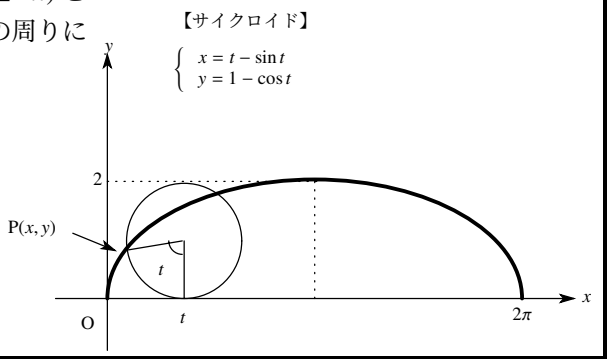
交点の x 成分が，$\pm\frac{\sqrt{2}}{2}$ と分かったので，面積は，次の積分で求まります.

• int(g[2] − 2 ∗ sqrt(2) ∗ x^2, x = −sqrt(2)/2 .. sqrt(2)/2) >> $\frac{\pi}{2} + \frac{1}{3}$

MuPAD で解く場合，solve では, 解けない方程式が多いので，交点を求める方が, 積分それ自体よりも難しいです．(この場合は簡単に求まりましたが)

例題 4

サイクロイド:$x = t - \sin(t), y = 1 - \cos(t)\ (0 \leqq t \leqq 2\pi)$ と x 軸で囲まれた領域 D の面積, および，D を x 軸の周りに回転した立体の体積を求めよ.



次はサイクロイド; $x = t - \sin(t), y = 1 - \cos(t)\ (0 \leqq t \leqq 2\pi)$ です．これは半径 1 の円を x 軸の上に転がしたときに出来る図形です．図より，サイクロイドのパラメータ表示は次のようになります.

$$\begin{cases} x = t - \sin t \\ y = 1 - \cos t \end{cases}$$

【解答】

x 軸との間で囲まれた面積を S，回転体の体積を V とします.

$x = t - \sin t$ とおくと $\frac{dx}{dt} = 1 - \cos t$, $\begin{array}{c|ccc} x & 0 & \to & 2\pi \\ \hline t & 0 & \to & 2\pi \end{array}$ ですから

$$\begin{aligned} S &= \int_0^{2\pi} y\,dx = \int_0^{2\pi} y\,\frac{dx}{dt}dt = \int_0^{2\pi} (1-\cos t)^2 dt \\ &= \int_0^{2\pi} (1 - 2\cos t + \cos^2 t)dt \\ &= \int_0^{2\pi} \left(1 - 2\cos t + \frac{1+\cos 2t}{2}\right)dt \\ &= \left[\frac{3}{2}t - 2\sin t + \frac{1}{4}\sin 2t\right]_0^{2\pi} \\ &= 3\pi \\ V &= \int_0^{2\pi} \pi y^2\,dx = \int_0^{2\pi} \pi y^2\,\frac{dx}{dt}dt = \int_0^{2\pi} \pi(1-\cos t)^3 dt \\ &= \pi\int_0^{2\pi} \left\{1 - 3\cos t + \frac{3}{2}(1+\cos 2t) - \frac{1}{4}(\cos 3t + 3\cos t)\right\}dt \\ &= \pi\left[\frac{5}{2}t - \frac{15}{4}\sin t + \frac{3}{4}\sin 2t - \frac{1}{12}\sin 3t\right]_0^{2\pi} \\ &= 5\pi^2 \end{aligned}$$

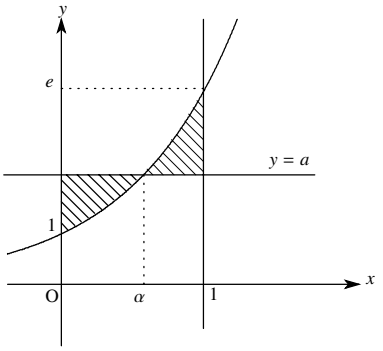
となるはずです．実際、MuPAD でやると，

[面積] • `int((1 − cos(t)) ∗ diff(t − sin(t), t), t = 0..2 ∗ PI)`　　>> 3π

[体積] • `int(PI ∗ (1 − cos(t))^2 ∗ diff(t − sin(t), t), t = 0..2 ∗ PI)`　　>> $5\pi^2$

例題 5

曲線 $C:\ y = e^x$, 直線 $l: y = a\ (1 < a < e)$ と，直線 $x = 1$, y 軸 によって囲まれる 2 つの領域の面積の和を S とする．このとき，S の最小値と，S を最小にする a の値を求めよ．



まず，MuPAD でやってみます．
はじめに，a の変域を `assume` によって制限します．

- `assume(1 < a < E)` >> $(1, e)$

次に，絶対値をつけて積分します．

- `S := int(abs(exp(x) − a), x = 0..1)` >> $e - 3\cdot a + 2\cdot a\cdot \ln(a) + 1$

S を a で微分して，S が極小値を取る a の値を求めます．

- `diff(S, a)` >> $2\cdot \ln(a) - 1$
- `amin := solve(%, a)` >> $\{\sqrt{e}\}$

次に, S の式の a に $\sqrt{e}$ を代入します．

- `subs(S, a = amin[1])` >> $e - 3\cdot \sqrt{e} + 2\cdot \ln(\sqrt{e})\cdot \sqrt{e} + 1$

この値を，評価 (evaluate) します．

- `eval(%)` >> $e - 2\cdot \sqrt{e} + 1$

ここで, 最初に `assume` を使って，a の範囲を定めないとうまく行きません．

- `delete a: S := int(abs(exp(x) − a), x = 0..1)` >> $\displaystyle\int_0^1 |e^x - a|dx$

手計算でやると次のようになります．

【解答】

$e^x = a$ の解を α とおくと，

$$\begin{aligned} S &= \int_0^{\alpha}(a - e^x)dx + \int_{\alpha}^{1}(e^x - a)dx \\ &= [ax - e^x]_0^{\alpha} + [e^x - ax]_{\alpha}^{1} \\ &= (a\alpha - e^{\alpha}) - (-1) + (e - a) - (e^{\alpha} - a\alpha) \end{aligned}$$

$e^{\alpha} = a,\ \alpha = \log a$ だから

$$\begin{aligned} S &= a\log a - a + 1 + e - a - a + a\log a \\ &= 2a\log a - 3a + e + 1 \text{ (略)} \end{aligned}$$

10.5　パラメーターを含む積分

整式の場合と同様，被積分関数や積分区域がパラメーターを含む場合は，パラメーターは default では，複素数になっています．よって，パラメータの範囲を assume を利用して，制限しないとうまく行かないことがあります．(前節の 例題 5 参照) ただ，整式と違い，被積分関数が不連続点を持つようなときは，さらに注意が必要です．例えば，「広義積分」の節で触れたように，区間が $x = 0$ を含むときは，$\frac{1}{x^2}$ は積分不能で，$\int_{-1}^{1} \frac{1}{x^2} dx$ は存在しません．よって，

- `int(1/x^2, x = -1..a)`　　>> $\int_{-1}^{a} \frac{1}{x^2} dx$

この積分は，簡単になりません．同様に

- `int(1/(x - a), x = 0..1)`　　>> $\int_{0}^{1} \left(-\frac{1}{a-x}\right) dx$

も簡単になりません．しかし，a の範囲を設定すると，どちらも簡単にすることができます．

- `assume(-1 < a < 0)`　　>> $(-1, 0)$
- `int(1/x^2, x = -1..a)`　　>> $-\frac{1}{a} - 1$
- `int(1/(x - a), x = 0..1)`　　>> $\ln(1 - a) - \ln(-a)$

注 15)

注15) ところが状況によっては, MuPAD の方で適当に範囲を制限しているように見える場合があります．例えば

- `delete a : int(1/x^2, x = 1..a)`　　>> $1 - \frac{1}{a}$

となります. この積分は $a \leqq 0$ のときは簡単にならないので，MuPAD の方で， a の範囲を，$a > 0$(ひょっとすると $a \geqq 1$) に 制限しているように見えます.

- `int(abs(x), x = 1..a)`　　>> $\frac{a^2}{2} - \frac{1}{2}$

も $a \geqq 1$ を感じさせます．ところが

- `int(abs(x^2 - 1), x = 2..a)`　　>> $\frac{a^3 \cdot \mathrm{sign}(a^2 - 1)}{3} - a \cdot \mathrm{sign}(a^2 - 1) - \frac{2}{3}$

と簡単になりません．$a \geqq 2$ なら `sign(a^2 - 1) = 1` の筈なのですが... とにかく, 最初に設定して置くに越したことはありません.

10.6 級数展開

f を $x=0$ の周りに taylor 展開	`taylor(f,x)`
f を $x=a$ の周りに taylor 展開	`taylor(f,x = a)`
f を $x=a$ の周りに 第 n 項まで, taylor 展開	`taylor(f,x = a,n)`
f を $x=0$ の周りに 級数展開	`series(f,x)`
f を $x=a$ の周りに 級数展開	`series(f,x = a)`
f を $x=a$ の周りに 第 n 項まで，級数展開	`series(f,x = a,n)`

注16) x の関数 $f(x)$ は, 適当な条件の下で級数に展開されます.

$f(x)$ が $x=a$ を含む領域で正則なとき,

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \cdots$$

これは Taylor 展開と呼ばれます. 例えば,

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + \cdots , \qquad \cdots ①$$

右辺は，初項 1, 公比 x の等比数列の和の公式で，$|x| < 1$ において収束します. これは，$x \quad 0$ のときの近似式を与えます. 例えば，$|x|$ が十分小さいとき，$x^2, x^3, \cdots$ は x に比べて非常に小さくなるので

$$\frac{1}{1-x} \quad 1 + x$$

実際, これを利用して MuPAD は，$x \to 0$ のときの極限を求めています. これ以外にも, いろいろな関数が Taylor 展開されます. 例えば,

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!} + \cdots , \qquad \cdots ②$$

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots , \qquad \cdots ③$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots , \qquad \cdots ④$$

Taylor 展開は，$x \quad 0$ の近似式を与えますが，一方，$x \quad \infty$ の近似式 (漸近式) を得ることもできます. 例えば，① の式で $x \to \frac{1}{x}$ とおいて,

$$\frac{1}{1-\frac{1}{x}} = \frac{x}{x-1} = 1 + \frac{1}{x} + \frac{1}{x^2} + \frac{1}{x^3} + \cdots \qquad \cdots ⑤$$

注16) 1.`taylor` は，内部で `series` を呼んでいるので，速度が `series` より早いわけではありません.
2.`taylor` も `series` も, 環境変数 *ORDER* で定まっている項数だけを表示します. Default では, *ORDER* = 6 です. これを変更しなくとも，*order(option* の一つ*)* で (一時的に) 指定できます.
3.`series` の *option* は, 他に dir=*Left*,*Right*,*Real*,*Undirected* のいずれか*)* と *NoWarning* があります. `taylor` には，*order* 以外の *option* はありません.

右辺は初項 1, 公比 $\frac{1}{x}$ の等比数列の和の公式で，$|x| > 1$ において収束します．x が非常に大きいときは，$\frac{1}{x^2}, \frac{1}{x^3}, \cdots$ は，$\frac{1}{x}$ に比べて十分小さくなるので，

$$\frac{x}{x-1} \quad 1 + \frac{1}{x}$$

とみなして良いことを表しています．また② の式からも同様にして

$$e^{\frac{1}{x}} = 1 + \frac{1}{x} + \frac{1}{2!x^2} + \frac{1}{3!x^3} + \cdots + \frac{1}{n!x^n} + \cdots \qquad \cdots ⑥$$

これは $|x| > 0$ において収束します．

①,②,③,④ の展開を，MuPAD でやってみます．

- `taylor(1/(1 - x), x)` >> $1 + x + x^2 + x^3 + x^4 + x^5 + O(x^6)$
- `taylor(exp(x), x)` >> $1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + O(x^6)$
- `taylor(sin(x), x)` >> $x - \frac{x^3}{6} + \frac{x^5}{120} + O(x^7)$
- `taylor(cos(x), x)` >> $1 - \frac{x^2}{2} + \frac{x^4}{24} + O(x^6)$

ここで $O(x^6)$ というのは,「x^6 以上の無限小」という意味で，$a_6x^6 + a_7x^7 + \cdots$ の省略です．項数を増やすには, *order* を指定します．order= 10 で，x^9 の項まで，表示します．

- `taylor(1/(1 - x), x, 10)` >> $1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + O(x^{10})$

または, 環境変数 ORDER (default では ORDER=6) を変えても良いです．しかし，環境変数を変えると，`delete` するまで，その設定は有効のままです．

- `ORDER := 5; taylor(1/(1 - x), x)` >> 5 >> $1 + x + x^2 + x^3 + x^4 + O(x^5)$
- `taylor(exp(x), x)` >> $1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + O(x^5)$

元に戻すには，`delete` します．

- `delete ORDER; taylor(exp(x), x)` >> $1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + O(x^6)$

MuPAD は，極限も，Taylor 展開を利用してやっているので，ORDER を変えるとできなくなる事があります．

- `ORDER := 1 : taylor(cos(x), x)` >> $1 + O(x^3)$
- `limit((cos(x) - 1 + x^2/2)/x^4, x = 0)` >> FAIL
- `delete ORDER : taylor(cos(x), x)` >> $1 - \frac{x^2}{2} + \frac{x^4}{24} + O(x^6)$
- `limit((cos(x) - 1 + x^2/2)/x^4, x = 0)` >> $\frac{1}{24}$

$x \quad 0$ のとき，$\cos x \quad 1 - \frac{x^2}{2} + \frac{x^4}{24}$ ですから，x が十分小さいとき，

$$\frac{\cos x - 1 + \frac{x^2}{2}}{x^4} \quad \frac{\left(1 - \frac{x^2}{2} + \frac{x^4}{24}\right) - 1 + \frac{x^2}{2}}{x^4} = \frac{1}{24}$$

を利用して,MuPAD は極限を求めています.

$x = a$ の周りに展開するには,「$x = a$」とします．省略すると $x = 0$ の周りに展開します．① の左辺の式を，$x = 2$ の周りに展開してみます.

- `taylor(1/(1 - x), x = 2)`

  >> $-1 + x - 2 - (x-2)^2 + (x-2)^3 - (x-2)^4 + (x-2)^5 + O((x-2)^6)$

ここで，右辺は初項 (-1), 公比 $-(x-2)$ の等比数列の和なので, $|-(x-2)| < 1 \Longleftrightarrow 1 < x < 3$ のとき,

$$(\text{右辺}) = \frac{-1}{1 - \{-(x-2)\}} = \frac{1}{1-x} = (\text{左辺})$$

となります.

以上の展開は，`taylor` の代わりに，`series` でも全く同じ結果が得られます．しかも，スピードも同じなので, いつも, こちらを使っても大丈夫です.

- `series(1/(1 - x), x)` >> $1 + x + x^2 + x^3 + x^4 + x^5 + O(x^6)$
- `series(1/(1 - x), x = 2)`

  >> $-1 + x - 2 - (x-2)^2 + (x-2)^3 - (x-2)^4 + (x-2)^5 + O((x-2)^6)$

MuPAD で ⑤,⑥ の展開を得るには `x = infinity` とします．こちらは，`series` の方を使わないといけません.

- `series(x/(x - 1), x = infinity)` >> $1 + \frac{1}{x} + \frac{1}{x^2} + \frac{1}{x^3} + \frac{1}{x^4} + \frac{1}{x^5} + O\left(\frac{1}{x^6}\right)$
- `series(exp(1/x), x = infinity)` >> $1 + \frac{1}{x} + \frac{1}{2 \cdot x^2} + \frac{1}{6 \cdot x^3} + \frac{1}{24 \cdot x^4} + \frac{1}{120 \cdot x^5} + O\left(\frac{1}{x^6}\right)$

これは，x が非常に大きいときの近似式を与えます．ところが

- `series(exp(x), x = infinity)` >> e^x

これは，e^x には，$x \to \infty$ のときの式は, $\frac{1}{x}$ の展開式では近似できないからです.（もし, そんなことができたら，$x \to \infty$ のときは，e^x は定数に近づいてしまう!) そういう時は，MuPAD はそのままを返してきます.

第11章　平面のグラフ

MuPAD 3.0 になって一番変わったのが，Graphics だと言えます．Graphics のプログラミングは, 一から書き直され, 空間図形の描写も改良されました. 描ける基本図形の数も 50 種類以上となり, 円, 四角形, , 直方体, 円柱, 円錐, 回転楕円体, 正多面体などもすぐ描く事ができます.*Attribute* (MuPAD 2.5 では *option* と呼んでいたもの) の数も 300 種類ぐらいに増えました. さらに，その *attribute* のほとんど全てを, 対話的 (interactive) に操作することができます．アニメーション (動く図形) も簡単に描け,AVI 形式で export することもできます.

11.1　y=f(x) のグラフ (plotfunc2d)

$y = f(x)$ のグラフ ($a \leqq x \leqq b$)	plotfunc2d(f(x), x = a ..b)
$y = f(x), y = g(x), \cdots$ のグラフ ($a \leqq x \leqq b$)	plotfunc2d(f(x), g(x), $\cdots$, x = a ..b)
$y = f(x)$ のアニメーション	plotfunc2d(f(x, t), x = a ..b, t = t_1..t_2)
$y = f(x), y = g(x), \cdots$ のアニメーション	plotfunc2d(f(x, t), g(x, t), $\cdots$, x = a ..b, t = t_1..t_2)

$y = f(x)$ のグラフを描くには plotfunc2d を使うのがもっとも簡単で，たくさんのグラフを一緒に描くことが出来ます．また plotfunc2d では，関数 $f(x)$ の値が実数でない時は, 無視します．(全く描きません)

y の範囲も $c \leqq y \leqq d$ のように指定したいときは次のようにします.

plotfunc2d(f(x), x = a ..b, ViewingBoxYRange = c ..d)

注1)

11.1.1　最初のグラフ (ViewingBox,Scaling,Grid の設定)

多項式のグラフを描きながら，ViewingBox,Scaling,Grid *Attribute* の設定の仕方も見てみます.

$y = x^3 - 3x + 1 (-3 \leqq x \leqq 3)$ のグラフは？

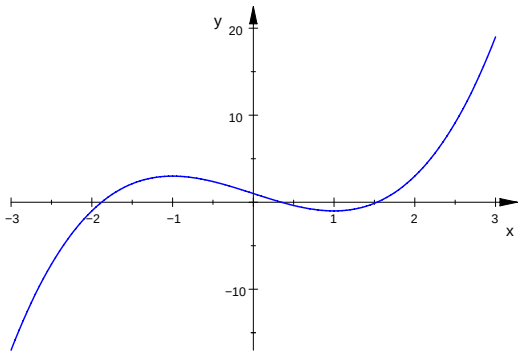
- plotfunc2d(x^3 − 3 ∗ x + 1, x = −3..3)

と打ちます．すると,Pro の場合は notebook 上に，Light の場合は，別の window 上 (Vcam という) に，次のようなグラフが見えるはずです.

注1) MuPAD 2.5 のときは ,y の範囲を制限するには

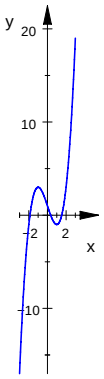
plotfunc2d(f, x = a..b, y = c..d)

のようにしました．ところが MuPAD 3.0 では，2 番目の変数は，アニメーションの変数と取られるので，もはや, これは使えません.



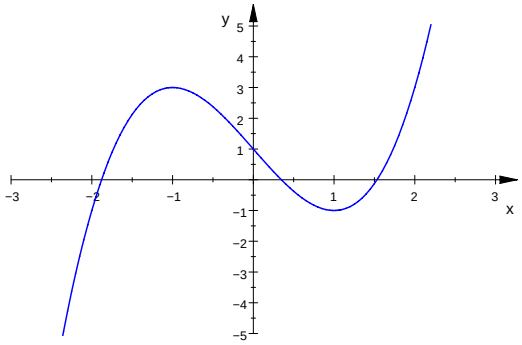
このようにやや y 軸方向につぶれて見えます．これは MuPAD が自動的に y 軸方向の拡大率 (Scale) を変更して書いているためです．(y 軸の「20」という目盛りで，分かります．)
Attribute で Scaling= *Constrained* と指定すると x, y 軸方向の拡大率 (scale) のとり方が同じになります．(初期値では *Scaling=Unconstrained* になっていて，MuPAD が自動的に拡大率を変更します.)

- `plotfunc2d(x^3 − 3 ∗ x + 1, x = −3..3, Scaling = Constrained)`



x,y 軸方向の拡大率が同じになっているのが分かります．今度は，y 軸方向の変域を $-5 \leqq y \leqq 5$ に指定してみます．これは ViewingBoxYRange または YRange で指定します．[注2)]

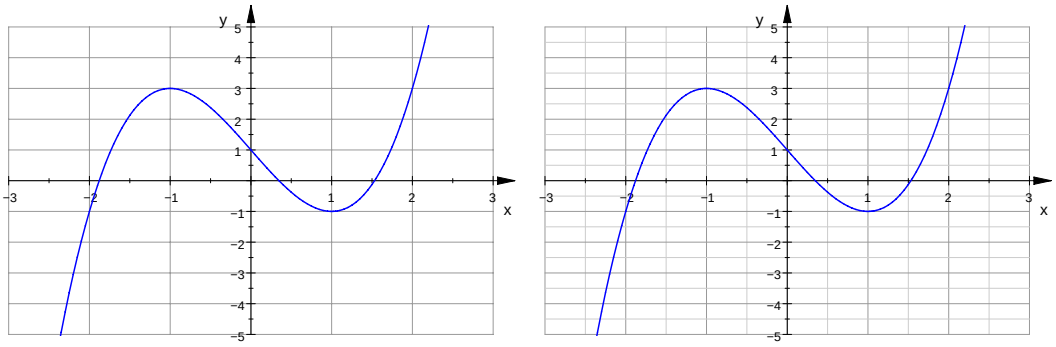
- `plotfunc2d(x^3 − 3 ∗ x + 1, x = −3..3, ViewingBoxYRange = −5..5)`



[注2)] 後述の `Function2d` では, ViewingBoxYRange しか効きません．なお，ViewingBox の *Attribute* には, 他に ViewingBoxXMin, ViewingBoxXMax,ViewingBoxYMin, ViewingBoxYMax などがあります.
$x = -3..3$ の代わりに，ViewingBoxXMin= −3, ViewingBoxXMax= 3. ViewingBoxYRange= −5..5 の代わりに，ViewingBoxYMin= −5, ViewingBoxYMax= 5 としても同じです.

このように，$-3 \leqq x \leqq 3$ かつ $-5 \leqq y \leqq 5$ の範囲しか表示されません．**y 軸方向に拡大したいときはこのように y の範囲も指定するといいです．**さらに，極値の値を見るために，座標系の格子 (grid) を表示させます．grid の表示には，GridVisible= *True* と指定します．さらに，比較のため，SubgridVisible= *True* で，subgrid も表示させて見ます．

- `plotfunc2d(x^3 - 3 * x + 1, x = -3..3, ViewingBoxYRange = -5..5, GridVisible = TRUE)`
- `plotfunc2d(x^3 - 3 * x + 1, x = -3..3, ViewingBoxYRange = -5..5,`
  `GridVisible = TRUE, SubgridVisible = TRUE)`

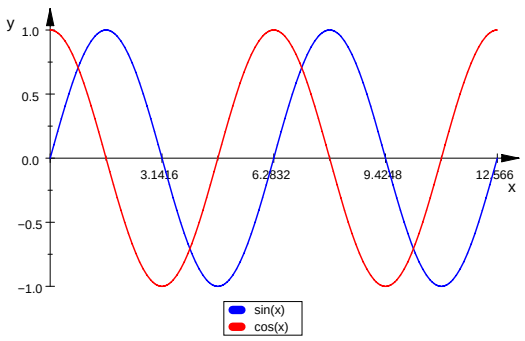


座標軸に目盛り (ticks) と数字 (label) が振ってあります．このラベルが振ってある点を通る格子を *grid*，ラベルの振っていない点を通る格子を *subgrid* といいます．[注3)] この後，しばらくはいろいろな *Attribute* について説明するので，多少難しく感じるかもしれません．しかし，MuPAD 3.0 では，ほとんど全ての *Attribute* は，Vcam または notebook から対話的に操作，編集できますから，名前を正確に覚える必要はありません．**うろ覚え**でも全く問題ありません．それでも面倒だと思われる方は，第 11.1.7 節の「いろいろな関数」まで飛ばしてください．

11.1.2 三角関数のグラフ (ticks の設定)

MuPAD では，$\sin x, \cos x$ の角度の単位はラジアン (*rad*) です．早速，$y = \sin x, y = \cos x$ $(0 \leqq x \leqq 4\pi)$ のグラフを書いてみます．2 つのグラフを一緒に描くときは，コンマで区切るだけです．

- `plotfunc2d(sin(x), cos(x), x = 0..4 * PI, XTicksAnchor = 0, XTicksDistance = PI)`



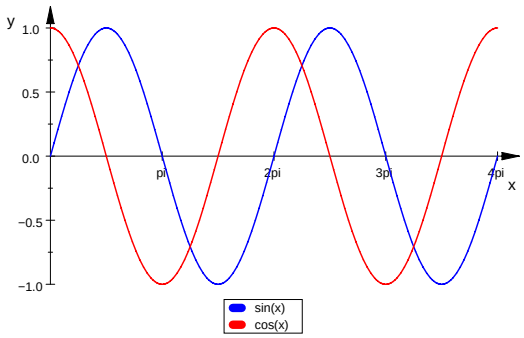
[注3)] この他の grid 関係の *Attribute* は，`GridLineColor`, `GridLineWidth`, `GridLineStyle`, `GridlnFront` などがあります．

このように, 2 つのグラフが色分けされて現われます．XTicksAnchor = 0, XTicksDistance = PI というのは，x 軸方向に ticks(目盛り) を，$x = 0$ からスタートして，π 刻みに，左右に付けていくことを意味しています．(一般には XTicksAnchor からスタートして，XTicksDistance 刻みに目盛りを付けていきます．)

また,2 つのグラフの下にある一覧表 (legend) は plotfunc2d では自動的に表示されます.

次は，ticks に $3.14, \cdots$ のような**数字**ではなく，$pi, 2pi, \cdots$ のような**文字**を表示させます．そのためには, 任意の点に, 任意のラベルを付けられる TicksAt = $[x_1 = label_1, x_2 = label_2, \cdots]$ を使います．ここで $x_1, x_2, \cdots$ は数字で，$label_1, label_2, \cdots$ は文字です．文字 (string) は，「” ”」に挟んで表します．また, TicksAt は, 普通 XTicksNumber = None とセットで使います.（もし XTicksNumber = None を指定しないと，$pi, 2pi \cdots$ という目盛りの他に，標準の目盛りを付けられてしまいます.）[注4)]

```
•plotfunc2d(sin(x), cos(x), x = 0..4 * PI, XTicksNumber = None,
 XTicksAt = [PI = ”pi”, 2 * PI = ”2pi”, 3 * PI = ”3pi”, 4 * PI = ”4pi”])
```

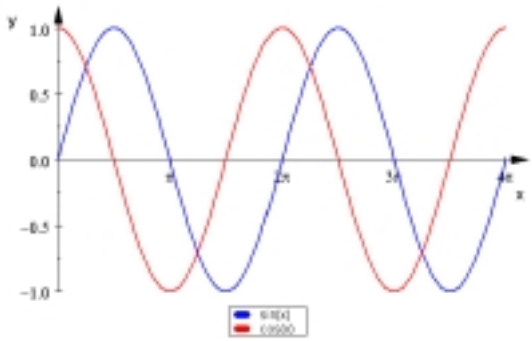


このように，π の位置に pi, 2π の位置に 2pi, $\cdots$ という目盛りが付きました.

さらに $\pi, 2\pi \cdots$ というラベルにするには，TicksLabelFont = [”symbol”] を利用します．これで，Font を *symbol* というフォントグループにすることができます．(フォント名は, リスト [ ] に入れて表します．これは，[”sans-serif”,10,Bold] や [”TimesNewRoman”,12,Italic] のように，フォントファミリーとサイズ, ボールド, イタリックなどをまとめて指定するためです．) そして, *symbol* というフォントグループでは, π は 「p」で表されるので，次のようにすれば，うまく行きます.

```
•plotfunc2d(sin(x), cos(x), x = 0..4 * PI, XTicksNumber = None, YTicksNumber = Low,
 TicksLabelFont = [”symbol”], XTicksAt = [PI = ”p”, 2 * PI = ”2p”, 3 * PI = ”3p”, 4 * PI = ”4p”])
```

注4) XTicksNumber の値は，*High*, *Normal*, *Low*, *None* の 4 つあり，この順で，目盛り (ticks) の数が減っていきます.



[./section1eps/example2/symbol_pi2.eps の，目盛りが π になっていたら, そちらを使用してください．] 注5)

11.1.3 ぎざぎざグラフ (Mesh の設定)

グラフが滑らかにならずに,’ ぎざぎざ’ になることがあります．このような時は Mesh *Attribute* を使います．注6) Mesh *Attribute* は, 細かく分けると，`Mesh`, `Submesh`, `AdaptiveMesh` の 3 種類あります.

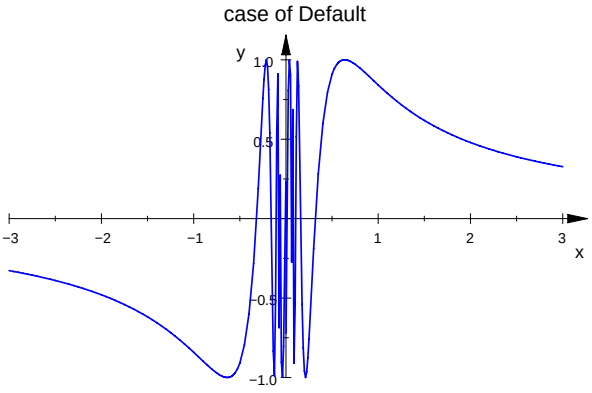
ここでは，有名な $\sin\frac{1}{x}$ を例に取り上げます．この関数は $x=0$ の近くでは無数に -1 と 1 の値をとり振動します．実際，$\sin\frac{1}{x}=1$ とおいてみると，

$$\sin\frac{1}{x}=\pm1 \Longleftrightarrow \frac{1}{x}=(2n\pm\frac{1}{2})\pi \Longleftrightarrow x=\frac{1}{(2n\pm\frac{1}{2})\pi}\ (n \text{ は整数})$$

すなわち，$x=\frac{1}{(2+\frac{1}{2})\pi}, \frac{1}{(4+\frac{1}{2})\pi}, \frac{1}{(6+\frac{1}{2})\pi}, \cdots$ のとき，$\sin\frac{1}{x}=1$ となり, $x=0$ の近くで振動します.

まず，$y=\sin\frac{1}{x}$ $(-3\leqq x\leqq 3, x\neq 0)$ のグラフを描いてみます.

- `plotfunc2d(sin(1/x), x = -3..3, Header = "case of Default")`



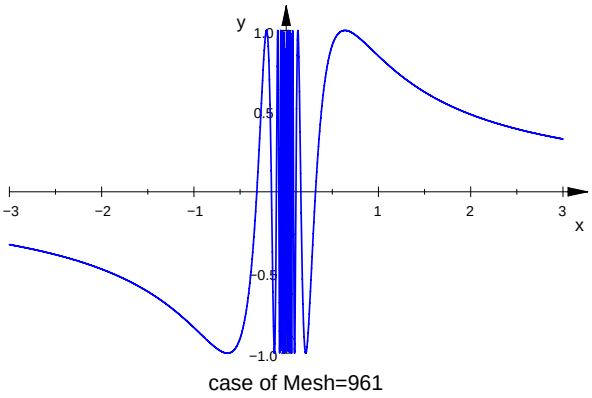
このように，**ぎざぎざ** になっています．また Header は，グラフの**上**に付きます．Header=”· · · ” のように，” ” マークの中に, 文を入れます.

注5) この他，ticks 関係では,`TicksBetween` =n (ラベルの付いた ticks の間に n 個の，ラベルの付かない ticks を入れる), `TicksLabelStyle`(値は `Horizontal`, `Vertical`, `Diagonal`, `Shifted`), `TicksLength` などが，あります.

注6) MuPAD 2.5 では，*grid option* と呼ばれていました．ちなみに，grid は「格子」，mesh は「網の目」という意味です.

次は, Mesh= 961 と指定してやって見ます.

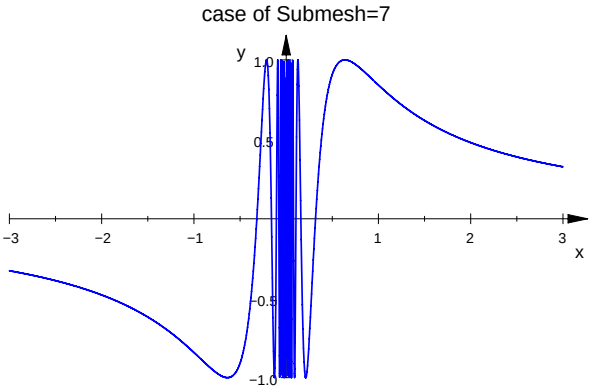
- `plotfunc2d(sin(1/x), x = −3..3, Mesh = 961, Footer = "case of Mesh = 961")`



だいぶ滑らかになりました．Mesh= 961 というのは， 指定区間: $-3 \leqq x \leqq 3$ の間に，$x = -3, x = 3$ を含め，等間隔に 961 個の x 成分をとって，$f(x)$ の値を計算して, グラフを描くという事です．[注7] default では，`plotfunc2d` では，Mesh= 121 になっています．したがって, ほぼ 8 倍の数の点を取って，計算した事になります．また Footer は，図のように, グラフの**下**に付きます.

これに対し，Submesh を指定して, 同じ効果を得る事ができます．Submesh= 7 と指定してやって見ます.

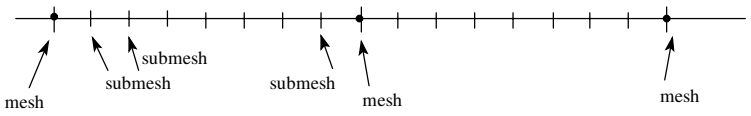
- `plotfunc2d(sin(1/x), x = −3..3, Submesh = 7, Header = "case of Submesh = 7")`



さて，Submesh= 7 というのは，もともとの Mesh(121 個) の各々の間に，7 個の点を取って計算すると

[注7] この場合では，$x_k = -3 + \frac{3-(-3)}{960} * k, (k = 0, 1, 2, \cdots, 960)$ に対し，$(x_k, f(x_k))(k = 0, 1, 2, \cdots, 960)$ という 961 個の点ができ，その点を利用して，グラフを描くことになります.

いう事です.



Mesh= 3, submesh= 7 のとき

したがって，もともと 120 個の区間に分かれていた部分をさらに 8 等分する事になり，ほぼ 8 倍の個数の点を取った事になります. 注8)

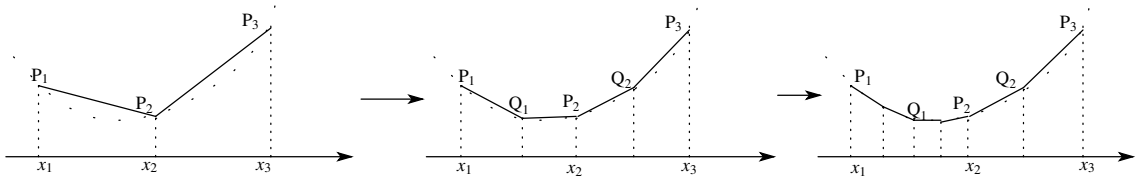
Submesh と Mesh の違いは，$y = f(x)$ のグラフでは現われませんが，3 次元 (空間) のグラフや，曲面を描くときは, 細かな違いがあります．すなわち，空間の曲面などでは, ユーザーが見やすいように, 曲線を曲面上に描きます．(地球儀の緯度線や経度線のようなものです.）この線は Mesh= n だけに反応します．すなわち，Mesh を大きくすると, 緯度, 経度線も細かくなりますが，Submesh を大きくしても, (曲面は滑らかになりますが）緯度, 経度線には, 変化がありません.

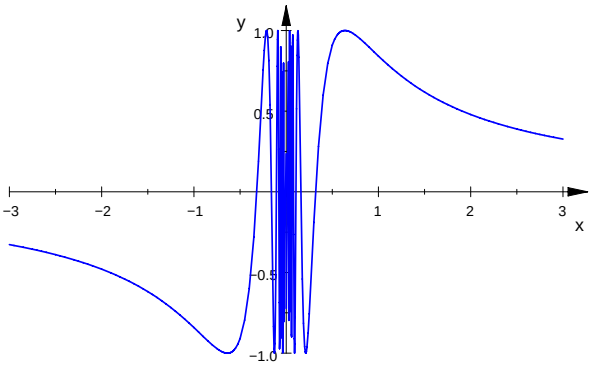
さらに MuPAD 3.0 になって新しく導入された `AdaptiveMesh` もあります．これは `Mesh` や `Submesh` のように, 区間を均等に割るのではなく，**ぎざぎざになっている区間だけ** を, 滑らかに見えるように, 再帰的に，半分に分割します．その再帰 (繰り返し）の最大回数を `AdaptiveMesh` = n で指定します. `AdaptiveMesh` = n のとき, 最大の場合，2^n 程度の新しい点が作られるので，Manual では $n = 1, 2, 3$ 程度の大きさにしておく事を薦めています．が，ここでは，敢えて，`AdaptiveMesh` = 4 で描いてみます. 注9)

- `plotfunc2d(sin(1/x), x = -3..3, AdaptiveMesh = 4, Footer = "AdaptiveMesh = 4")`

注8) 一般には，`Mesh` = n, `Submesh` = m のとき，点の個数は $(n-1)(m+1)+1$ です.
`Mesh` = 121, `Submesh` = 7 は，`Mesh` = $(121-1)\cdot(7+1)+1 = 961$, `Submesh` = 0 と同じです.

注9) 次の図で x_1, x_2, x_3 が Mesh や Submesh で与えられている点とします．このとき 線分 P_1P_2, 線分 P_2P_3 のなす角が 10 度より大きいと，区間 $[x_1, x_2], [x_2, x_3]$ を半分に分割します．そして, 線分 P_1Q_1, 線分 $Q_1P_2, \cdots$, 線分 Q_2P_3 において, 隣接する線分のなす角が, 全て 10 度以下だと, そこで終了します．しかし AdaptiveMesh $\geqq 2$ で，かつ線分のなす角が 10 度より大きい場合は, さらに同じ事を繰り返して，新しい点を追加します．この最大再帰（繰り返し）の回数が AdaptiveMesh です．AdaptiveMesh= n のとき，最大で $2^n - 1$ の点を，P_2 の両側に追加します．(図では AdaptiveMesh= 2 の場合で，P_2 の左側に $2^2 - 1 = 3$ 個, 右側には $2^1 - 1 = 1$ 個の点を追加しています.）

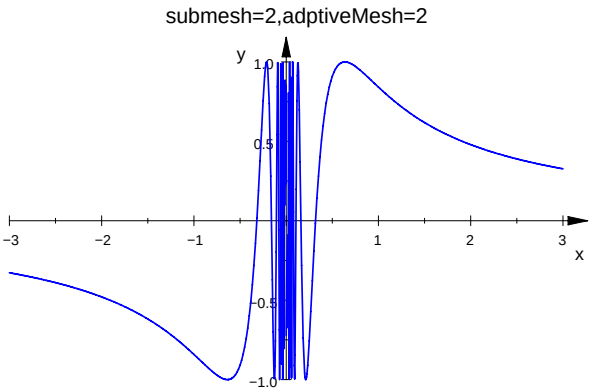




adptiveMesh=4

ご覧のように, Submesh = 7 の方がきれいです．関数によっては，AdaptiveMesh = n は，Submesh = 2^n-1 とほぼ同様になることもありますが，余りにも**ぎざぎざ**した関数では, AdaptiveMesh だけではうまく行かない時があります．このような場合は, **あらかじめ，**Submesh や Mesh **で, ある程度，網の目を小さくしてから，**AdaptiveMesh **を使った方がよい**ようです．[注10)]
次は，Submesh と AdaptiveMesh を併用して描いてみます．

•plotfunc2d(sin(1/x), x = −3..3, Submesh = 2, AdaptiveMesh = 2,
Header = ”submesh = 2, adptiveMesh = 2”)



Adaptivemesh の値は減らして，Submesh の値を大きくしました．こちらの方が，Adaptivemesh だけより，きれいです．

11.1.4　アニメーション (動画）

MuPAD 3.0 は, 非常に簡単にアニメーション (動画）を描く事ができます．$f(x)$ のグラフであれば，適当なパラメータ (ここでは t とします）をとって

•plotfunc2d(f(x, t), x = a..b, t = t_1..t_2)

のようにすれば，$y = f(x,t)$ $(t_1 \leqq t \leqq t_2)$ のアニメーションが描かれます．また，2 つ以上の関数をコンマで区切って, 同時にアニメーションすることもできます．さらに，(f(x) の方にパラメータを含んでいない時は，Warning は出るものの) 区間の方に, パラメータを含んでいても大丈夫です．

注10) なお，空間の曲面などでは,「AdaptiveMesh よりも，Submesh を使用した方が良い」とマニュアルには書いてあります．しかし, 余りにも**一部分だけが，ぎざぎざ**している場合は，空間でも AdaptiveMesh を併用した方が良いみたいです．(後述)

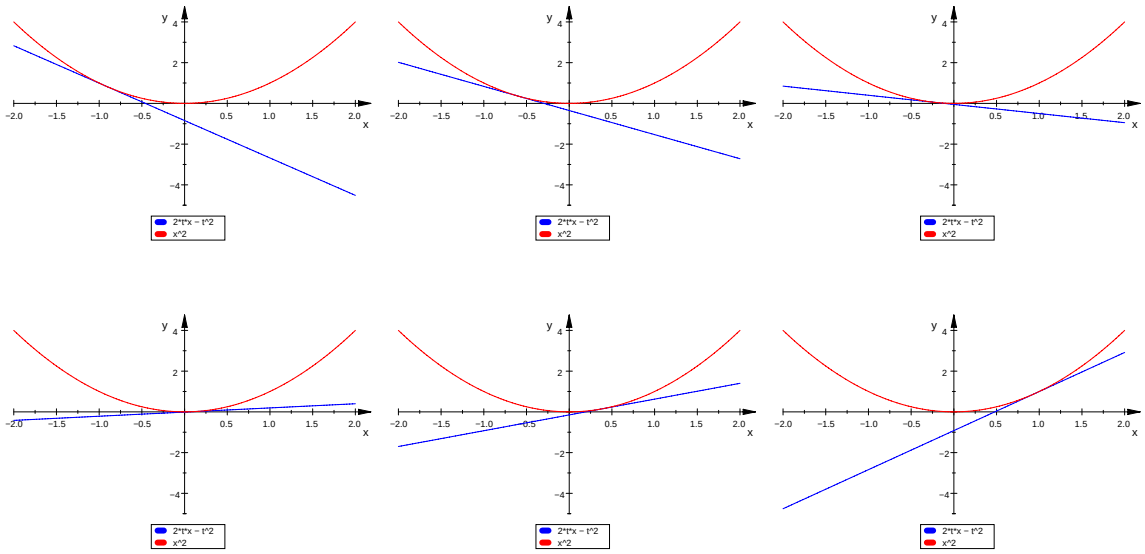
$y = f(x) = x^2$ に対し，$f'(x) = 2x$ だから，$y = f(x)$ 上の点 $(t, f(t))$ における接線は

$$y - t^2 = 2t(x - t) \Longleftrightarrow y = 2tx - t^2$$

そこで $y = 2tx - t^2(-2 \leqq x \leqq 2)$ のグラフと，$y = x^2(-2 \leqq x \leqq 2)$ のグラフを同時に描かせてアニメーションさせてみます．パラメータ：t は，$-1 \leqq t \leqq 1$ で動かしてみます．

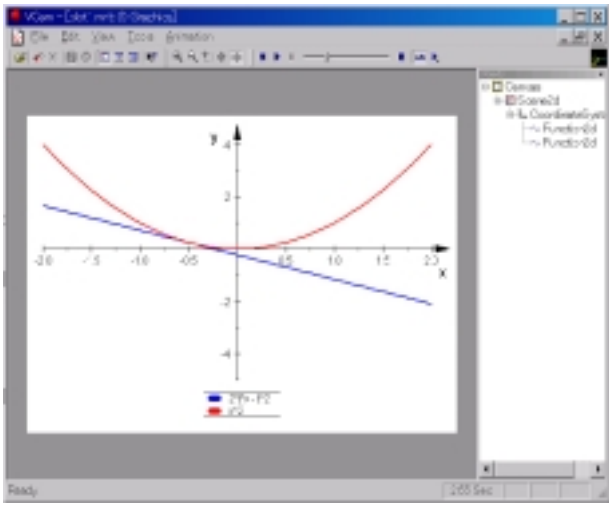
●`plotfunc2d(2 * t * x - t^2, x^2, x = -2..2, t = -1..1)`

これで次のようなアニメーションが見えます．



しかし，実は, すぐこのようにアニメーションする訳ではありません．最初は一番左上の画面のみが見えます．(これは，$t = -1$ の時の画像です. 接点の x 成分で分かります．) 一見すると全く普通の, グラフに見えます．しかし，MuPAD Pro の場合であれば, 画像をダブルクリックすれば,notebook の上の方のツールバー内に，CDplayer のようなマークが見えるはずです．または，画像の上で右クリックして「Graphics オブジェクト」から *Open* を選択すれば，Vcam が立ち上がり，そのツールバーに，同じく，CDplayer のようなマークが見えるはずです．

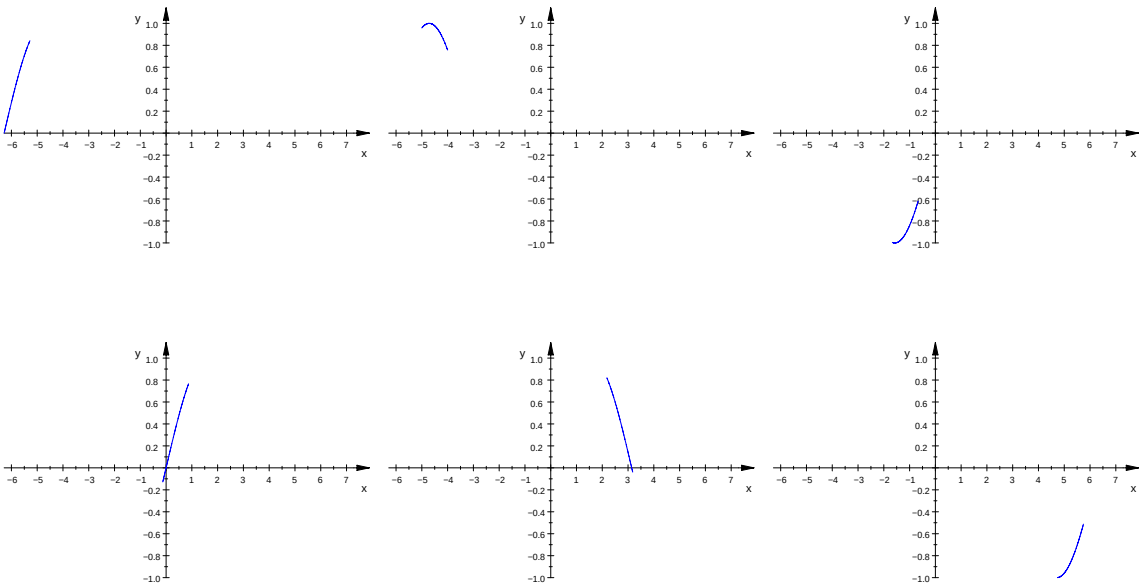
いずれにせよ，この CDPlayer もどきで，▶マーク をクリックすれば，アニメーションが始まります．



アニメーションを終了するには，notebook に埋め込まれている場合は，ダブルクリックで notebook に戻ります．notebook から，Vcam を開いている場合は，Vcam を閉じると notebook に戻ります．

次は,3 次関数 $y = \sin x (-2\pi \leqq x \leqq 2\pi)$ 上を動く，x 軸方向の幅が 1 の曲線のアニメを描いて見ます．名づけて,「芋虫すいすい」です．

```
●plotfunc2d(sin(x), x = a..a + 1, a = −2 ∗ PI..2 ∗ PI)
```



このように，アニメーションを描くには，x 座標 (座標系の横軸の変数) 以外にパラメータを一つ設定すれば良いです．しかも, そのパラメータは, 変域についていても大丈夫です．(ただし, $f(x)$ の方にパラメータを含んでいない，上のような場合は，警告が出ます.)

アニメーションの *Attribute* は，Frames(アニメーションのコマ数), TimeRange(アニメーションの動作時間, 単位は秒) などいろいろあります．default では, Frames= 50,TimeRange= 0..10 で 1 秒あたりのコマ数は，$\frac{50\text{ コマ}}{10\text{ 秒}} = 5$ コマ/秒 となっています．[注11)]

11.1.5　色の変更，注釈の付け方

最後に色の変更の仕方と，注釈 (Header,Footer,Title,Legend) の変更の仕方を見ます．色を変えるには, 関数が 1 個だけならば `Color` = ⋯ で, 関数が 2 個以上ならば `Colors` = [$Color_1, Color_2, \cdots$] と指定します．このとき Color 変数を用いて `Color` = `RGB` :: *Blue* のように指定するか，R(red),G(green),B(blue) の強さを, 0 以上 1 以下の数で表し，`Color` = [0, 0, 1] のように指定します．(RGB::Red=[1,0,0], RGB::Green=[0,1,0], RGB::Blue=[0,0,1] です．) RGB::Blue 以外の，Color 変数としては，RGB::Black, RGB::White, RGB::Green, RGB::Red, RGB::Blue, RGB::Yellow, RGB::Gray, RGB::Magenta,

---

注11) TimeRange= 0..20 は，TimeBegin= 0,TimeEnd= 20 としても同じです．また，plotfunc2d では,TimeRange= 10..30 としても同じです．(つまり, 時間の差だけが問題です.)
この他，アニメーションの *Attribute* としては，VisibleBefore,VisibleAfter,VisibleFromTo,VisibleBeforeBegin,VisibleAfterEnd などがあります．しかし，普通は余り使うことはないと思われます.

RGB::Olive, RGB::LightGray, など ”色々”あります．これらは, 全部で 287 種類あり, RGB :: ColorNames で, その全てを見ることができます.

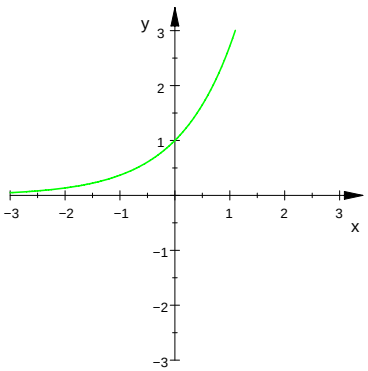
•RGB :: ColorNames()

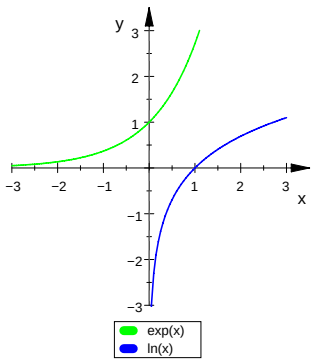
>> [AliceBlue, AlizarinCrimson, Antique, Aqua, Aquamarine, AquamarineMedium, AureolineYellow, Azure, Banana, Beige, Bisque, Black, Black0, Black10, Black100, Black15, ···

しかし，MuPAD 3.0 では，マウスで設定できるので, 覚える必要はありません．(後述) [注12)]

•plotfunc2d(exp(x), x = −3..3, YRange = −3..3, Scaling = Constrained, Color = RGB :: Green) :

•plotfunc2d(exp(x), ln(x), x = −3..3, YRange = −3..3, Scaling = Constrained, Colors = [RGB :: Green, RGB :: Blue])

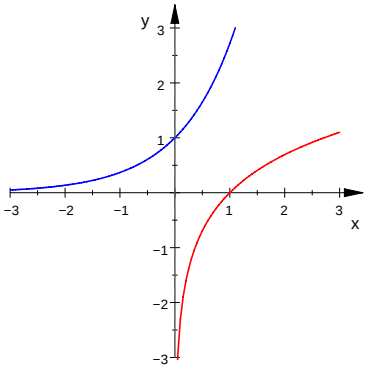


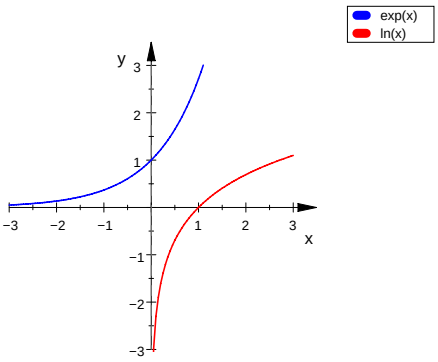


注釈 (annotation) は，ヘッダー (header), フッター (footer)，タイトル (title)，一覧表の 4 種類あります．先ず, 一覧表を消すには，LegendVisible = *FALSE* とします．一覧表の位置を変えるには，LegendAlignment(値は *Center*, *Left*, *Right*) と LegendPlacement(値は *Top*, *Bottom*) を使います.

•plotfunc2d(exp(x), ln(x), Scaling = Constrained, YRange = −3..3, x = −3..3, LegendVisible = FALSE) :

•plotfunc2d(exp(x), ln(x), Scaling = Constrained, YRange = −3..3, x = −3..3, LegendAlignment = Right, LegendPlacement = Top) :

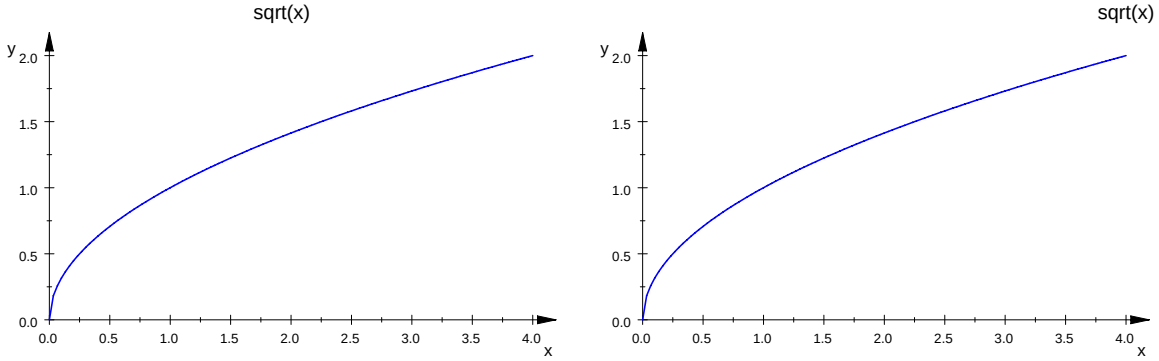




[注12)] この他, 透過率も指定できます．[1,0,0,0.3]=RGB::Red.[0.3] で, 透過率は 1 – 0.3 = 0.7 です．default では, 透過率は 0 になっています．すなわち，RGB::Red=[1,0,0,1] です．さらに，HSV 形式で指定することもできます．RGB :: fromHSV([h, s, v]) で，HSV 形式の [h,s,v] を，RGB 形式の [r,g,b] に直します．逆に，RGB :: toHSV([r, g, b]) で，RGB 形式を，HSV 形式に直すこともできます.

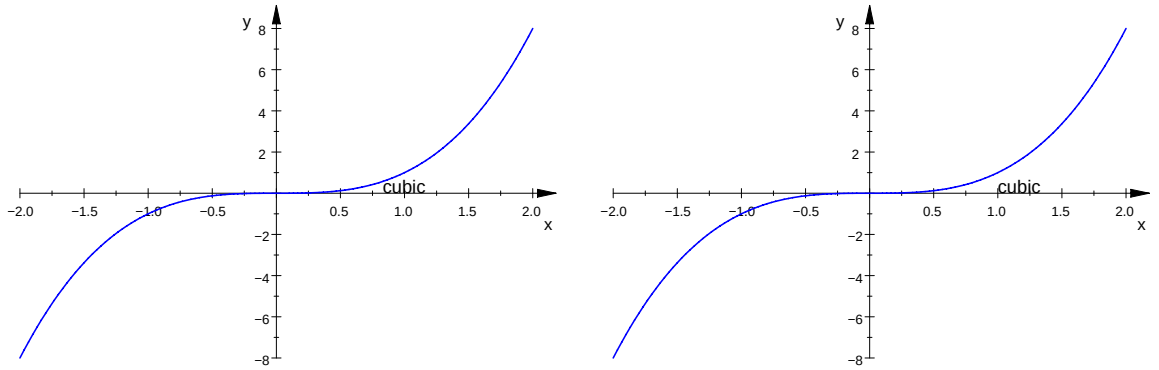
Header や Footer は default では，見えません．Header の位置は HeaderAlignment で, Footer の位置も FooterAlignment で変えられます．値は, どちらも，*Right*, *Center*, *Left* です．

- plotfunc2d(sqrt(x), x = 0..4, Header = "sqrt(x)")
- plotfunc2d(sqrt(x), x = 0..4, Header = "sqrt(x)", HeaderAlignment = Right)



最後に，タイトル (title) ですが，Title と Header や Footer との違いは，Title は関数ごとに何個も付けられるが，Header や Footer は, 1 つのキャンバスに一つしか付けられないという事です．[注13] Title の位置は，TitlePosition = [*x* 座標, *y* 座標] で指定します．さらに，TitleAlignment(値は *Left*, *Right*, *Center*) で，「左端位置合わせ」,「右端位置合わせ」,「中央位置合わせ」 を選ぶこともできます．

- plotfunc2d(x^3, x = −2..2, Title = "cubic", TitlePosition = [1, 0])
- plotfunc2d(x^3, x = −2..2, Title = "cubic", TitlePosition = [1, 0], TitleAlignment = Left)



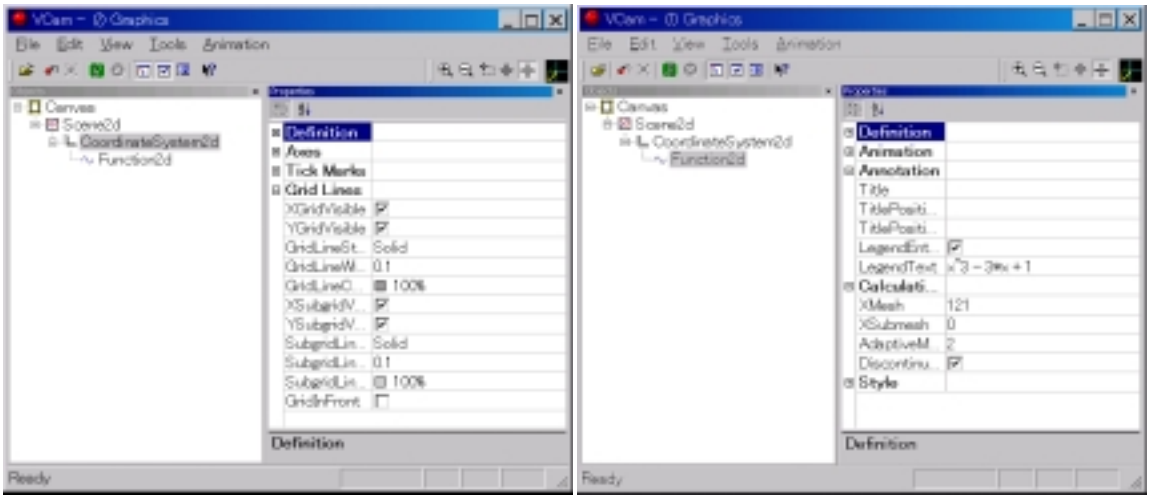
その他にも，たくさん *Attribute* はありますが，何しろ 300 種類もあるので, それらをいちいち説明することはできません．しかも，その中には余り必要ないものも多々ありますし，また, そのほとんどはマウスを使って, 対話的に変更, 設定できるので，ここでは，このくらいにしておきます．

11.1.6　対話的な *Attribute* の設定

ほとんど全ての *Attribute* は，Objects Brouser と Property Inspector を利用して，対話的 (interactive) に操作できます．Light の場合は, Vcam の メニューバーから *View* を選択し, *Objects* と *Properties* にチェックを付けてください．すると，Objects Brouser と Property Inspector を見ることができます．

[注13] ただし，plotfunc2d の場合は，Title も一個しか付けられません．何個も付ける場合は，後述する Function2d を使う必要があります．

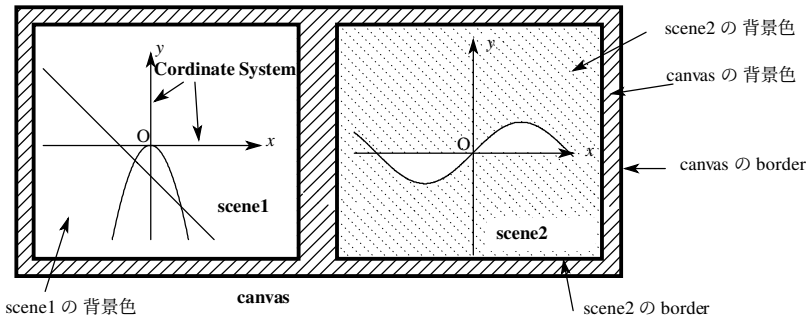
Pro の場合は, グラフをダブルクリックすると，notebook に埋め込まれた形で，Objects Brouser と Property Inspector が現われます.（もし, 見えない場合は，Vcam と同じく，メニューバーから *View* を選択し, *Objects* と *Properties* にチェックを付けてください.）または，canvas 上で右クリックして 「Graphics オブジェクト」 から，*Open* を選択すると，Vcam が立ち上がり，Light と同様に，Objects Brouser と Property Inspector を見ることができます. 注14)



左側の Objects Brouser で，⊞ マークを，適当にクリックしていくと，オブジェクトは，（plotfunc2d では)

Canvas(キャンバス), Scene(シーン), CoordinateSystem(座標系), Function(関数)

の 4 つに分類されていることが分かります.
MuPAD では,「キャンバス」の中に, いくつかの「シーン」が埋め込まれ，その中に,「座標系」を設定し,「グラフ」を (何個か）描く」という仕組みになっています．これが Canvas,Scene,CoordinateSystem,Function のカテゴリーです.



適当に選んで，クリックすると，今度は，右側の，Property Inspector が, それに応じて，下のような項目を表示します.

- Canvas の *Attribute* ⟶ Annotation,Layout,Style
- Scene2d の *Attribute* ⟶ Annotation,Layout,Style
- CoordinateSystem2d の *Attribute* ⟶ Definition,Axes,Tick Marks, Grid Lines
- Function2d の *Attribute* ⟶ Definition,Animation,Annotation,Calculation,Style

注14) 個人的には，右クリックして，Vcam を開く方が好きです．ダブルクリックしてグラフを見るのは楽ですが，私のコンピュータでは, よくフリーズします.

ここで，*Annotation* は「注釈」，*Axes* は「座標軸」,*Layout* は「配置」，*Tick Marks* は「座標軸についているマーク」，*Grid Line* は,「座標系の格子」，*Definition* は「定義」という意味です．したがって，例えば，`Grid` の *Attribute* なら，`CoordinateSystem2d`(座標系) を探せばよいし，`Mesh` の *Attribute* なら，`Function2d`(関数) の `Calculation`(計算) を探せば見つかります．このようにして, ほとんど全ての *Attribute* を，対話的に変更できます.

object brouser や property inspector を閉じるのは，notebook に埋め込まれている場合は, ダブルクリックするだけです．notebook から，Vcam を開いた場合は，Vcam を閉じれば，notebook に戻ります.

11.1.7　様々なグラフの例 – 高校の数学から

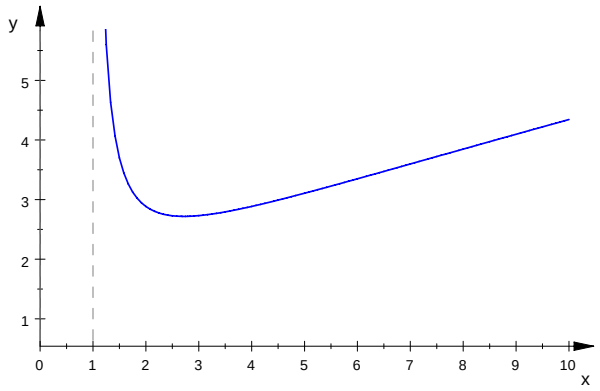
この節では, 高校の数学を例にとり，どんどんグラフを描いていきます．あわせて，Objects Brouser, Property Inspector をつかって，対話的に *Attribute* を変えてみます．ただし，単にグラフを描くだけならば，ViewingBox 関係を除き，*Attribute* は，ほとんど使う必要はありません．(きれいなグラフを描きたいならば別ですが $\cdots$) また，`plotfunc2d` では, $f(x)$ が実数にならない時は無視します.

例題 1

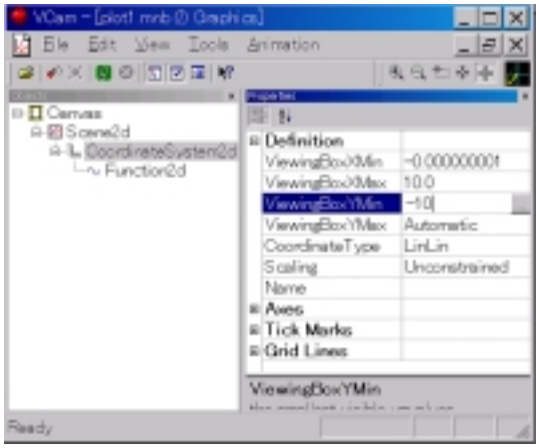
$y = \frac{x}{\log x} (0 < x \leqq 10)$ のグラフの概略を描け.

とりあえず, 普通に書いて見ます.

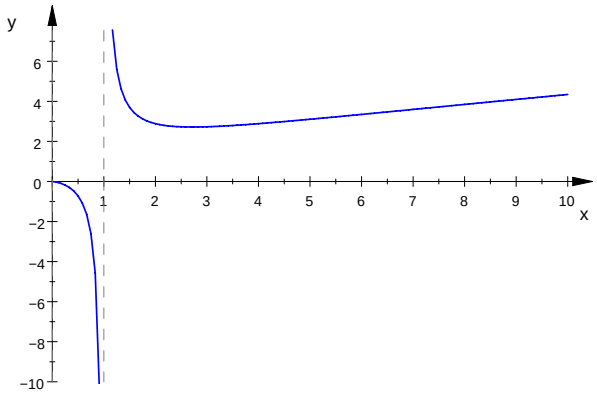
- `plotfunc2d(x/ln(x), x = 0..10)`



$0 < x < 1$ においては, グラフが表示されていません．$0 < x < 1$ においても $\frac{x}{\log x}$ は存在するはずなので，これは ViewingBox の問題と考えられます．ViewingBox は「座標系」の問題なので，*CoordinateSystem2d* をクリックします．ViewingBox は，とても基本的なことですから，Definition の問題です．クリックして，ViewingBoxYMin=−10, ViewingBoxYMax= *Automatic* と変えてみます.

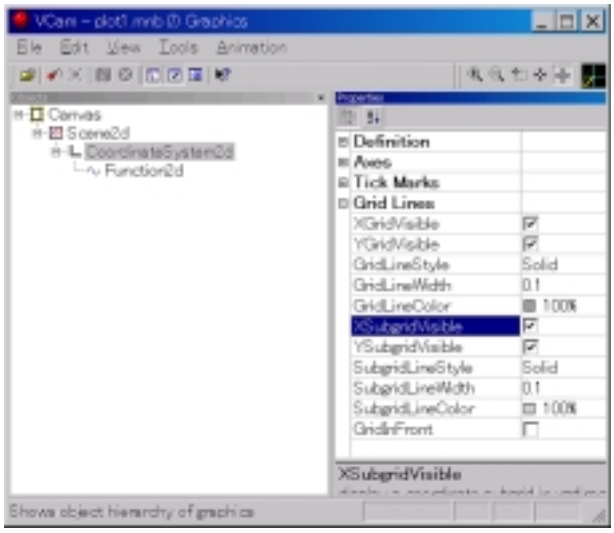


すると, 次のようになります.

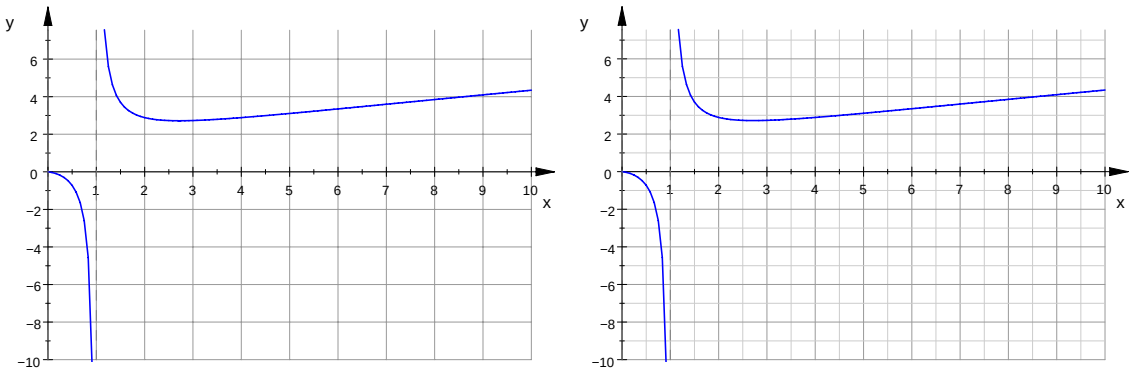


もしならない場合は，メニューバーから，*View* を選択し，*Recalculate*(再計算) をクリックします．notebook の場合は，このとき，Object Inspector が開いていることを確認してください．(notebook のメニューバーは，Object Inspector が開いているか, 否かで変わります.)

さらに，grid(格子) を表示させて見ましょう．これも，座標系の問題なので *CoordinateS ystem2d* をクリックします．今度は，Grid Lines をクリックして，`XGridVisible`,`YGridVisible` にマークを付けます．さらに subgrid(ラベルの付いていない ticks を通る格子) を表示するには，XSubGridVisible,YSubGridVisible にもマークを付けます.



すると, 次のようになります.　(左側が grid のみ表示, 右が grid, subgrid ともに表示.)

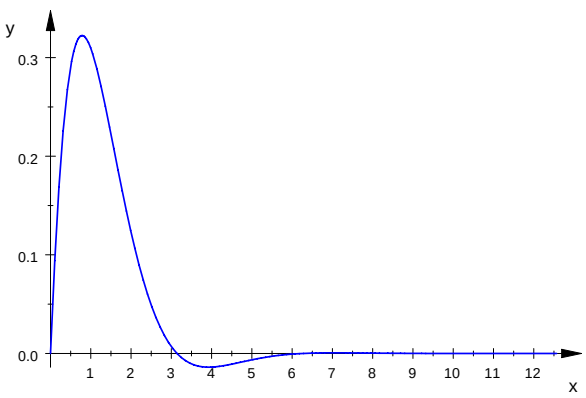


例題 2

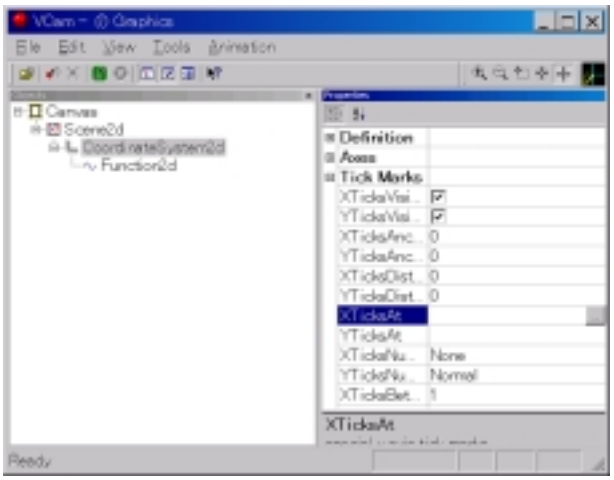
$y = e^{-x} \sin x \ (0 \leqq x \leqq 4\pi)$ のグラフの概略を描け.

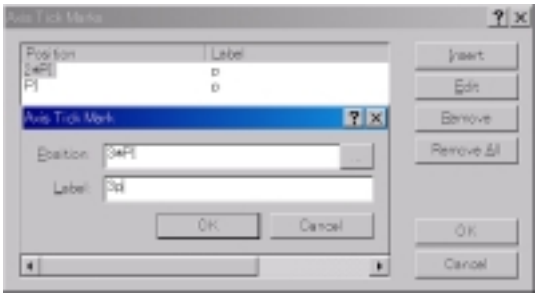
$y = e^{-x} \sin x \ (0 \leqq x \leqq 4\pi)$ のグラフを描いてみましょう. 横軸の単位は，$\pi, 2\pi, \cdots$ としたいですが，とりあえず, 普通に書いて見ます.

- `plotfunc2d(exp(-x) * sin(x), x = 0..4 * PI)`

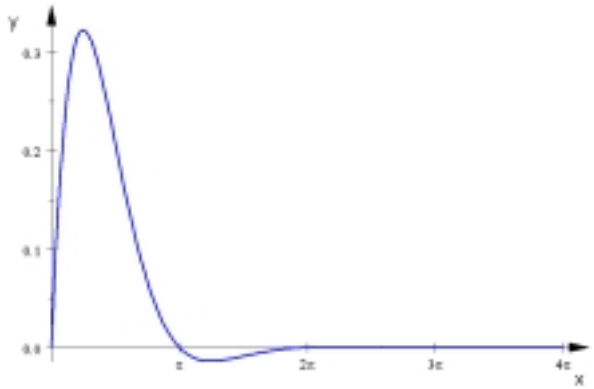


横軸の単位を, $\pi, 2\pi, \cdots$ にするには, `TicksAt` を指定します. これは, 座標系の問題ですから, *CoordinateSystem2d* から, `TickAt`(の右にある[$\cdots$]) をクリックします. [注15] さらに, [Insert]をクリックすると, 新たな画面が開きます.





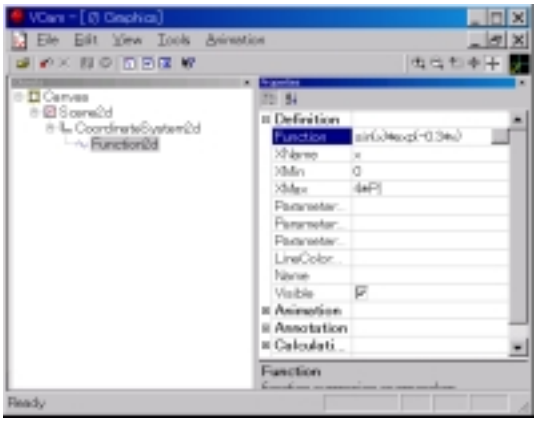
ここに 1 つずつ, Position と Label に,「PI」と「p」を書き入れ, [OK]を押します. 同様にして, 再び, [Insert]をクリックして,「2*PI」と「2p」を書き入れ, [OK]を押します. これを続けます. `TickAt` だけだと, $1, 2, \cdots$ という, 通常の ticks も残るので, 次は `XTicksNumber` をクリックして, *None* を選択します. 最後に `TicksLabelFont` をクリックして, フォントを *symbol* にします.
すると, *symbol* というフォントでは, p= π なので, うまくいきます. もしうまく行かない場合は, Recalculate(再計算) させます. すると, 次のような画面になるはずです.



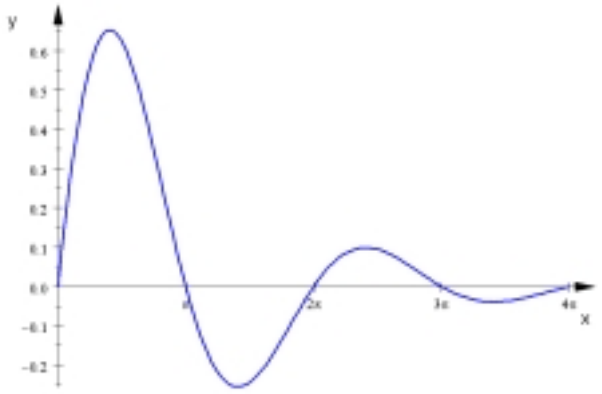
[exmaple2_2a.eps のラベルが $\pi, 2\pi, \cdots$ になっていれば, そちらを使用.]

上のグラフは, つぶれる速度が速すぎてよくわからないので, $y = e^{-x} \sin x$ の代わりに, $y = e^{-0.3x} \sin x$ のグラフを描いてみます. このときは, 座標系とは無関係で, 関数の定義 (difinition) ですから, *Function2d* と辿り, [Difinition]をクリックします. その一番上の [Function] をクリックして, `sin(x)*exp(-0.3*x)` と変更します.

[注15] [$\cdots$]が見えない場合は, 画面をつまんで, 右側に, 大きくしてください.



先と同様に，Reculculate してやると, 下のグラフになります.



[exmaple2_3a.eps のラベルが $\pi, 2\pi, \cdots$ になっていれば，そちらを使用．]
指数が $0.3x$ となっているので、$x = 4\pi$　12.5 の近くでもつぶれていません．教科書などでは，解りやすいようにデフォルメしてあることが多いです.

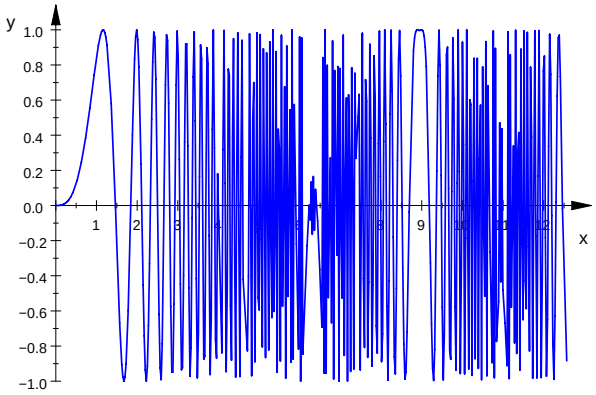
次は，**ぎざぎざ**グラフです．

例題 3

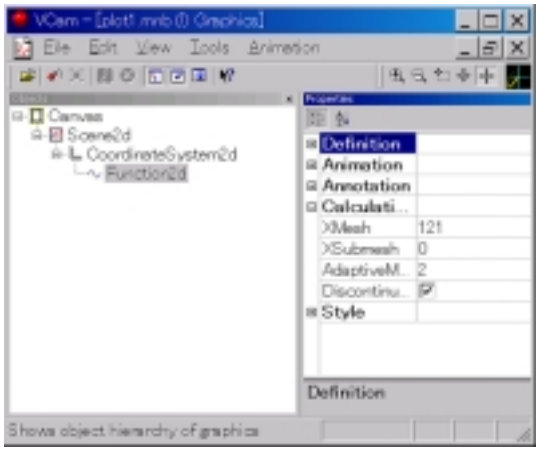
$y = f(x) = \sin x^3$ $(0 \leqq x \leqq 4\pi)$ の増減をしらべてグラフを描け．

とりあえず，描いてみます．

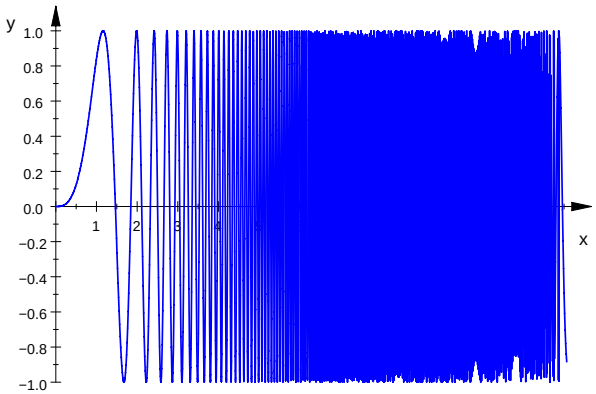
- `plotfunc2d(sin(x^3), x = 0..4 * PI)`



このように**ぎざぎざ**になっています．そこで Mesh をいじってみます．`Mesh` = n というのは, 与えられた区間に均等に n 個の点 x_k をとって，$f(x_k)$ の値を**計算**するということなので，*Function2d* の Calculate を開きます．すると , XMesh=121,SubMesh=0,AdaptiveMesh=2 というのが見えます．



これを，例えば，XMesh=121,SubMesh=7,AdaptiveMesh=2 に代えると, 次のようになります．



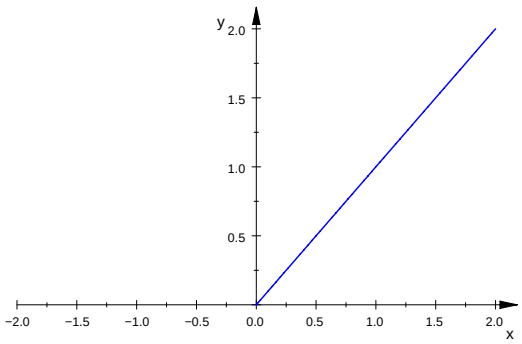
次は，$f(x)$ の値が実数にならないときです．plotfunc2d は, そのような場合を無視します.

例題 4

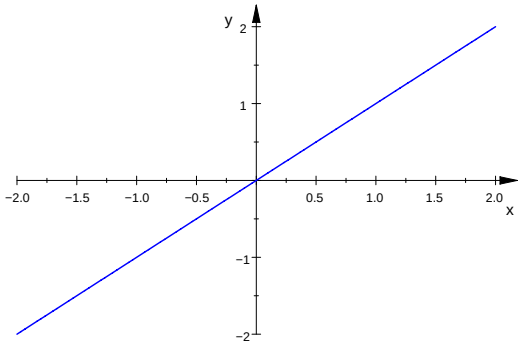
$y = f(x) = \sqrt[3]{x^3}$ $(-2 \leqq x \leqq 2)$ のグラフを MuPAD で描け.

次のようにするとどうでしょう？

- `plotfunc2d((x^3)^(1/3), x = -2..2)`



これだと上図のように $x \geqq 0$ の範囲でしか図示されません．しかし, 実際は x が実数のとき $\sqrt{x^3} = x$ が成り立つので, 下図のようになるはずです．なぜでしょう?



MuPAD では，$(-8)^{\frac{1}{3}} = -2$ となりません．$a < 0$ のときは $a^{\frac{1}{3}}$ は a の虚数 3 乗根を現してしまいます．実際， rectform(z) は，z を $x + iy(x, y$ 実数) の形に直すコマンドですが,

- `rectform((-8)power(1/3))`　　>> $1 + i \cdot \sqrt{3}$

となって, $(-8)^{\frac{1}{3}} \neq -2$ です．そして plotfunc2d では，虚数の値は無視するので，$x < 0$ のとき，$y = x^{\frac{1}{3}}$ は, プロットできません.

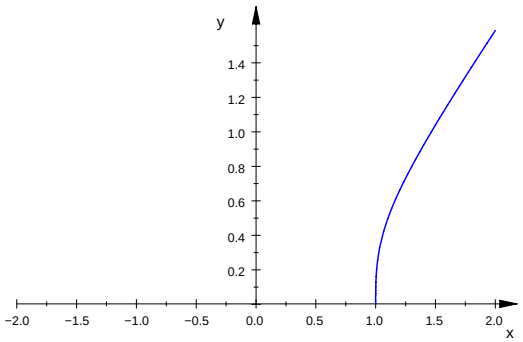
例題 5

$y = f(x) = \sqrt[3]{x^3 - x^2}$ $(-2 \leqq x \leqq 2)$ の増減をしらべてグラフを描け．また漸近線も一緒に描け.

とりあえず，$x \geqq 0$ のとき, $\sqrt[3]{x} = x^{\frac{1}{3}}$ を使って，MuPAD で,$y = f(x)$ のグラフを描いてみます.

- `plotfunc2d((x^3 - x^2)^(1/3), x = -2..2)`

しかし, うまく行きません.



$$\begin{cases} x^3 - x^2 = x^2(x-1) \geqq 0 & \Longleftrightarrow x \geqq 1 \\ x^3 - x^2 = x^2(x-1) < 0 & \Longleftrightarrow x < 1 \end{cases}$$

ですから，MuPAD は, $x < 1$ の区間をプロットしません．このような時は, 継ぎはぎ関数で定義すると良いです.

$$\sqrt[3]{x^3 - x^2} = \begin{cases} (x^3 - x^2)^{\frac{1}{3}} & (x \geqq 1 \text{ のとき}) \\ -(x^2 - x^3)^{\frac{1}{3}} & (x < 1 \text{ のとき}) \end{cases}$$

注16)

MuPAD では, `piecewise` を使います. 注17)

- `f := piecewise([ x >= 1,(x^3 - x^2)^(1/3) ], [ x < 1, -(x^2 - x^3)^(1/3) ])`

  >> $\begin{cases} \sqrt[3]{x^3 - x^2} & if \quad 1 \leqq x \\ -\sqrt[3]{x^2 - x^3} & if \quad x < 1 \end{cases}$

注16) これは $\sqrt[3]{-8} = -\sqrt[3]{8} = -8^{\frac{1}{3}}$ のように，$a > 0$ のとき，$\sqrt[3]{-a} = -\sqrt[3]{a} = -a^{\frac{1}{3}}$ となることを利用しています.

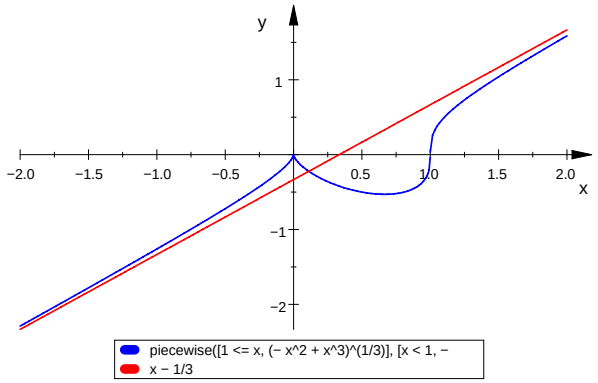
注17) MuPAD のプログラミングを使って, 次のようにしても同じです．このとき，g は関数になりますが，`plotfunc2d` は, 式でも関数でも大丈夫なので,`plotfunc2d(g, x = −2..2)` で描写できます.

```
• g := proc(x)
begin
if x >= 1 then (x^3 − x^2)^(1/3)
else −(x^2 − x^3)^(1/3)
end_if
end_proc
```

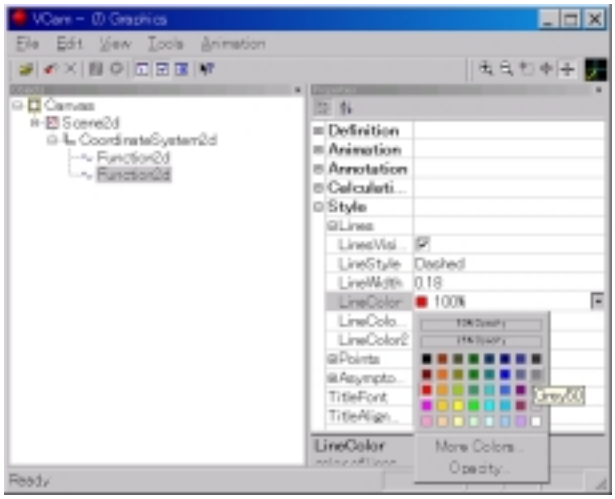
漸近線は，$y = x - \frac{1}{3}$ となる[注18)] ので，一緒にプロットします．

- `plotfunc2d(f(x), x - 1/3, x = -2..2)`

次のようなグラフになります．



欲を言えば，$y = x - \frac{1}{3}$ は，$y = f(x)$ の漸近線 (asymptote) なので，灰色の破線にしたいところです．これは，関数の**スタイル**の問題なので，2 つ目の `Function2d` をクリックし，`Style` から `Lines` と辿り，`LineStyle` = *Dashed*, `LineWidth` = 0.18 と変更します．さらに，その下の `LineColor` をクリックして，Grey50 を選択します．



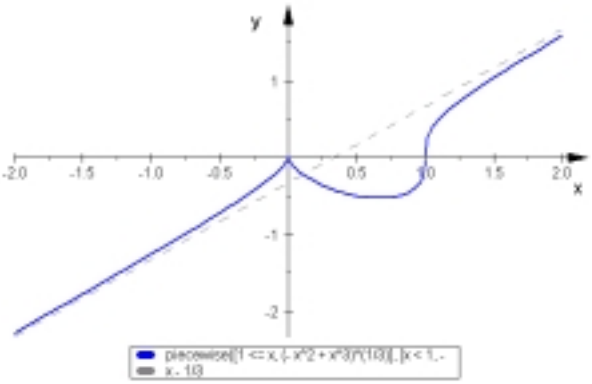
[注18)] 手計算してみます．$x < 0$ のときでも，$x^{\frac{1}{3}} = \sqrt[3]{x}$ と定義すると，0 以外の実数に対して $(x^{\frac{1}{3}})' = \frac{1}{3}x^{-\frac{2}{3}}$ が成り立ちますから，$y = (x^3 - x^2)^{\frac{1}{3}}$ とおくと，$x^3 - x^2 \neq 0 \iff x \neq 0, x \neq 1$ のとき，

$$y' = \frac{1}{3}(x^3 - x^2)^{-\frac{2}{3}} \cdot (3x^2 - 2x) = \frac{x(3x-2)}{3\sqrt[3]{(x^3-x^2)^2}} = \frac{3x-2}{3\sqrt[3]{x(x-1)^2}}$$

x	$\cdots$	0	$\cdots$	$\frac{2}{3}$	$\cdots$	1	$\cdots$
y'	+	＼	−	0	+	＼	+
y	↗	0	↘	$-\frac{\sqrt[3]{4}}{3}$	↗	0	↗

また $\sqrt[3]{x^3 - x^2} = \sqrt[3]{(x - \frac{1}{3})^3 - \frac{1}{3}x + \frac{1}{27}} = \left(x - \frac{1}{3}\right)\sqrt[3]{1 + \frac{-\frac{1}{3}x + \frac{1}{27}}{(x-\frac{1}{3})^3}}$ だから $x \to \pm\infty$ のとき，$y = \sqrt[3]{x^3 - x^2}$ は限りなく $y = x - \frac{1}{3}$ に近づきます．すなわち，漸近線は $y = x - \frac{1}{3}$．

これで, 必要ならば, 再計算 (Recalculate) させると, 次のようになります.



[example5_3a の方で,$y = x - \frac{1}{3}$ が点線になっていれば, そちらを使ってください]
このようにして，簡単に, 曲線の色や, 線種を変えることができます．また，関数が 2 つあるときは，オブジェクト：Function2d も 2 つできます.

例題 6

m を実数の定数とする．2 直線

$$l_1 : \ mx - y = 0 \qquad l_2 : \ x + my - m - 2 = 0$$

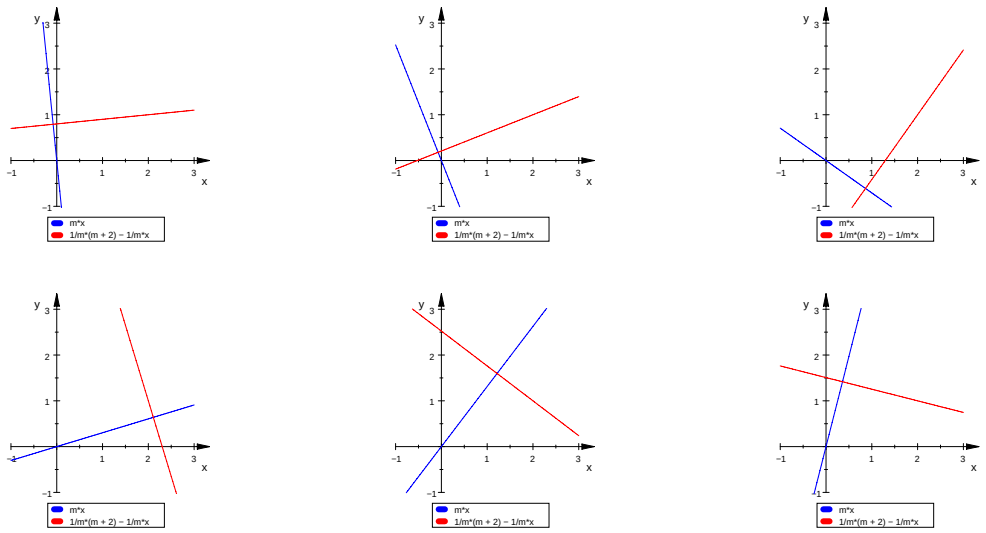
の交点の軌跡は，どのような図形になるか？

アニメーションで描いてみます． $m \neq 0$ のとき,

$$x + my - m - 2 = 0 \Longleftrightarrow y = -\frac{1}{m}x + \frac{m+2}{m}$$

よって次のようにしてみます.

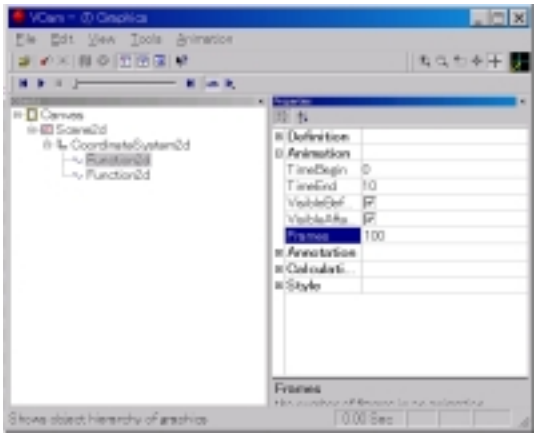
```
●plotfunc2d(m * x, -1/m * x + (m + 2)/m, x = -1..3, m = -10..10
,ViewingBoxYRange = -1..3, Scaling = Constrained)
```



どうやら, 交点は円周になりそうです．実際,

$$x + my - m - 2 = 0 \Longleftrightarrow x - 2 + m(y - 1) = 0$$

ですから，$m \neq 0$ のとき，l_2 は, 定点 A(2, 1) を通り, 傾き $-\frac{1}{m}$ の直線を表します．一方，l_1 は, 原点 O を通り，傾き m の直線ですから, 2 本は直交します．よって，その交点は，線分 OA を直径とする円周上にあります．[注19] また，$m = 0$ のとき，$y = -\frac{1}{m}x + \frac{m+2}{m}$ は定義できませんが，plotfunc2d は，このような例外は，無視して描写します．少しギクシャクしているので，アニメーションのコマ数を変更します．コマ数は，Function2d の Animation から変更できます．



ここで，Frames = 100 に変更します．両方の関数で, 同じように変更します．すると滑らかに動くようになります．(図は省略)
このように，アニメーションのコマ数は，それぞれの関数の *Attribute* になっています．したがって, 別々に設定することができます.

最後に, 注釈の付け方を見てみます.

例題 7

3 次関数

$$f(x) = x^3 + ax^2 + x$$

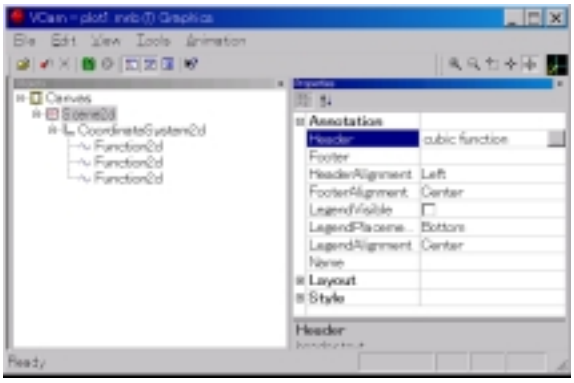
がある．$a = 1$, $a = 3$, $a = 0$ のときの $y = f(x)$ のグラフを描いて，タイトルも付けよ．

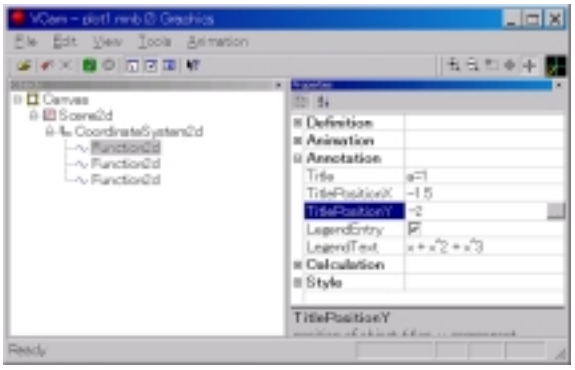
とりあえず，plotfunc2d で描いてみます.

●plotfunc2d(x^3 + x^2 + x, x^3 + 3 ∗ x^2 + x, x^3 + x, YRange = −3..5, x = −3..3)

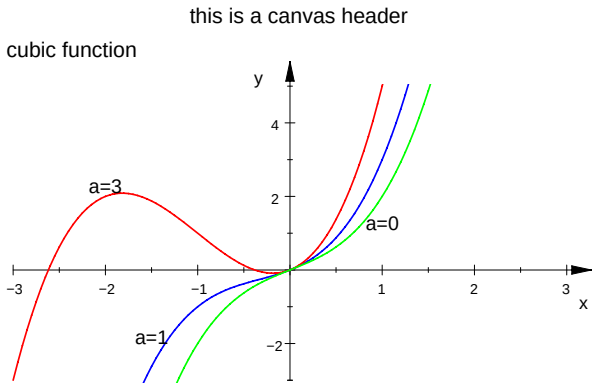
注釈 (annotation) は,「キャンバス」,「シーン」,「関数」 のそれぞれで，付けることができます．まず，一覧表 (legend) を消します．これは,「シーン」の *Attribute* ですから，Scene2d の Annotation にあります．ここで，LegendVisible = *FALSE* にします．次に，一覧表の代わりに，ヘッダーを指定しましょう．Header = *cubic function*(三次関数) とし， HeaderAlignment = *Left* にします (左図). 次に，関数にタイトルをつけます．Function2d の Annotation で，Title, TitlePositionX, TitlePositionY を指定します (右図)．仮に，TitlePositionX = −1.5, TitlePositionY = −2 とすると，Title のベースラインの中心が，(−1.5, −2) に来ます.

[注19] 詳しく言うと，l_1 は，原点 O を通る直線のうち，$x = 0$ を，l_2 は，点 A を通る直線のうち，$y = 1$ を表さないので，その交点 (0, 1) は，軌跡に含みません.





タイトルは関数ごとに指定できるので，3 つとも同様にして指定します．最後に,「キャンバス」のヘッダーも付けます．`Canvas` から，同じく，`Annotation` を選んで,「シーン」のヘッダーと同様に指定できます．これで次のようになります．



薔薇(ばら) の漢字が書けなくとも, 読むことはできるのと同様に，見て意味が推定できれば，*Attribute* は，自由に変更できます．

11.2 平面の幾何図形 (low-level primitives)

MuPAD 3.0 では, 円, 楕円, 長方形, 円錐, 正多面体などの複雑な幾何図形をすぐ描くことができます．このような幾何図形は”low-level primitive” と呼びます．一方，`plotfunc2d` のように, 関数式を指定していろいろなグラフを描かせることもできます．こちらは，”high-level primitive” と呼びます．この節では,

low-level primitive を取り上げます．2 次元 (平面) の low-level primitive は, 以下のとおりです.

plot :: Arc2d	円弧 (円の一部)	Arc2d(r, [C_x, C_y], α..β)
plot :: Arrow2d	矢印	Arrow2d([x_1, y_1], [x_2, y_2])
plot :: Circle2d	円弧	Circle2d(r, [C_x, C_y])
plot :: Ellipse2d	楕円	Ellipse2d(r_x, r_y, [C_x, C_y])
plot :: Line2d	線分	Line2d([x_1, y_1], [x_2, y_2])
plot :: Parallelogram2d	平行四辺形	Parallelogram([C_x, C_y], [U_x, U_y], [V_x, V_y])
plot :: Point2d	点	Point2d(x, y)
plot :: PointList2d	多くの点	PointList2d([[x_1, y_1], [x_2, y_2], $\cdots$])
plot :: Polygon2d	多角形	Polygon2d([[x_1, y_1], [x_2, y_2], $\cdots$])
plot :: Rectangle	長方形	Rectangle(x_{min}..x_{max}, y_{min}..y_{max})
plot :: Text2d	文	Text2d("$\cdots$", [C_x, C_y])

注20)

これらは, 全て,

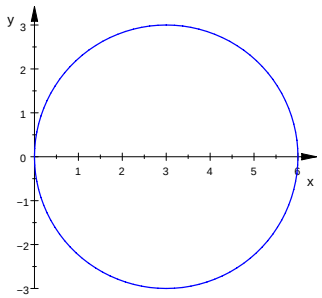
- plot(plot :: Circle2d(3, [3, 0]))　　$\cdots$ 中心が (3, 0) で, 半径 3 の円

のように plot() コマンドと一緒に使います．しかし，次のように分けることもできます.

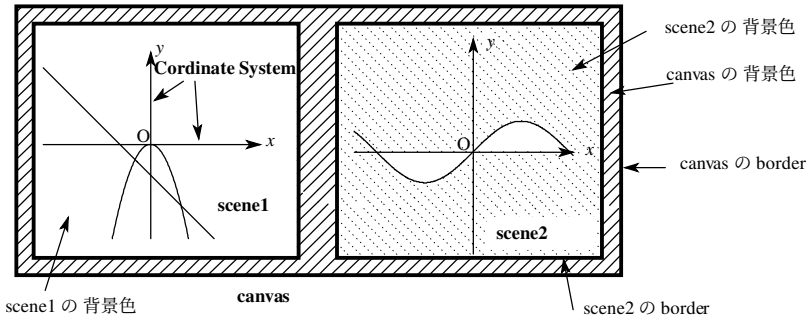
- mycircle := plot :: Circle2d(3, [3, 0])　　>> plot::Circle2d(3, [3, 0])
- plot(mycircle)

これは, 最近流行の「オブジェクト指向」の流れで，plot :: CIrcle(3, [0, 3]) で「半径 3 で, 中心が (3, 0) の円」という**物 (object) を作り**，plot(object) で, その **物 (object) を描く**という感じです.

いずれにせよ，次のようになります.



注20) 主な *Attribute* としては，領域を囲むような図形 (円, 楕円, 平行四辺形, 長方形など）に対しては，Filled(値は *TRUE*, *FALSE*), FillColor(内部の色), LineColor(境界の色) などが，点や点列に関しては, PointSize, FillColor などが，文 (text) に関しては, TextFont, TextRotation などが，また, 円弧, 多角形に関しては，Closed(値は *TRUE*, *FALSE*) などもあります．これらの *Attribute* も，property inspector から，編集することができます.



オブジェクトの間には,

$$\texttt{Canvas} \supset \texttt{Scene2d} \supset \texttt{CoordinateSystem2d} \supset \texttt{plot :: Circle2d}$$

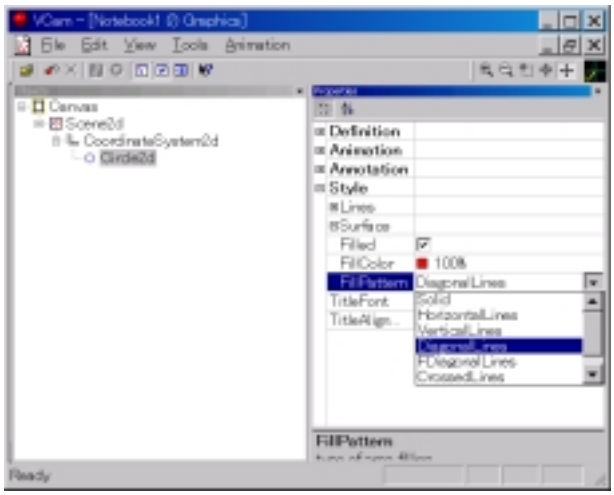
の関係が成り立つので

- `plot ( plot :: Canvas ( plot :: Scene2d ( plot :: CoordinateSystem2d(`

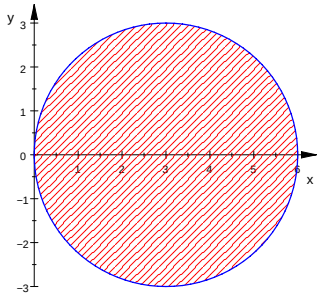
  `plot :: Circle2d(3,[3,0]) ))))`

としても大丈夫です．実際は，MuPAD は，内部で，上のような形に直して描写しています．

Attribute も，property inspector から設定できます．

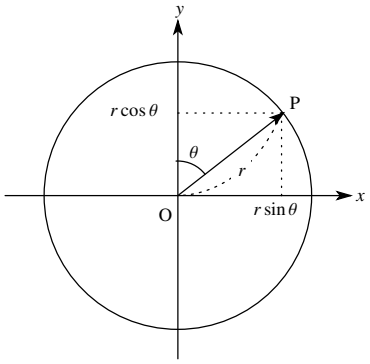


`plotfunc2d` と違い，`Function2d` が `Circle2d` に変わっています．斜線で塗りつぶしてみます．これは，「円」の スタイル の問題なので，`Circle2d` の `Style` を開けます．`Filled` = *TRUE*, `FillColor` を Red にし，`FillPatern` = *DiagonalLines*(斜線) にします．すると，次のようになります．



[アニメーション]

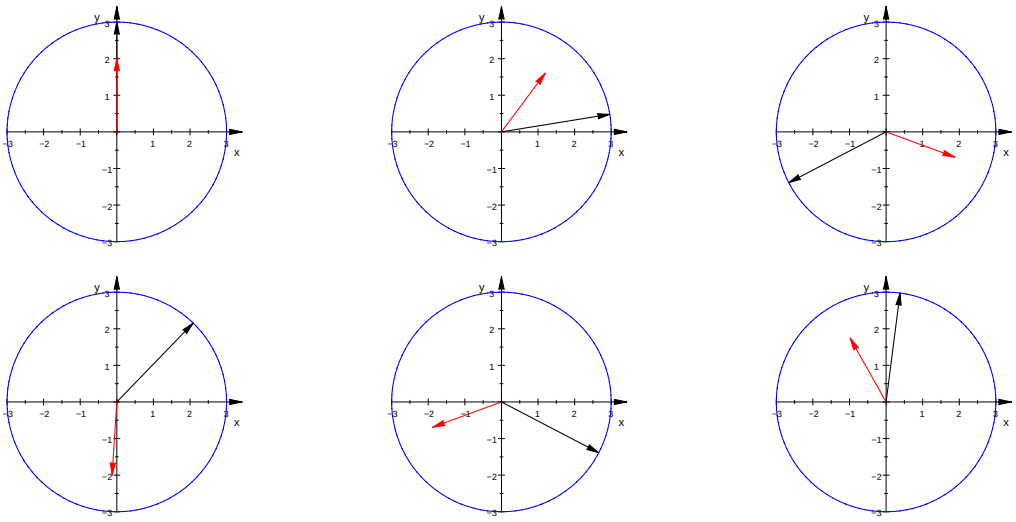
low-lebel primitive もアニメーションできます．パラメータを 1 つ増やすだけです．例として,「アナログ時計」を描いて見ます．先ず,body(本体),minutehand(分針), timehand(長針) だけからなるシンプルな時計を作ります.



- `body := plot :: Circle2d(3, [0, 0])` : ⋯ 中心 (0,0), 半径 3 の円
- `minutehand := plot :: Arrow2d([0, 0], [3 * sin(t), 3 * cos(t)], t = 0..24 * PI, Color = RGB :: Black)` : ⋯ 始点が $(0,0)$, 終点が $(3\sin t, 3\cos t)$ の矢印
- `timehand := plot :: Arrow2d([0, 0], [2 * sin(t), 2 * cos(t)], t = 0..2 * PI, Color = RGB :: Red)` : ⋯ 始点が $(0,0)$, 終点が $(2\sin t, 2\cos t)$ の矢印

`Arrow2d([a_x, a_y], [b_x, b_y])` で，始点が (a_x, a_y), 終点が (b_x, b_y) の矢印を作ります．minutehand は，$0 \leqq t \leqq 24\pi$ と変域を指定したので，1 アニメーション周期あたり，原点の周りを 12 周します．timehand は，$0 \leqq t \leqq 2\pi$ と変域を指定したので，1 アニメーション周期あたり，原点の周りを 1 周します．2 つのアニメーションの変数は独立なので，t 以外の文字を使って，例えば，`s = 0..2 * PI` としても大丈夫です．これをまとめて，plot します.

- `plot(minutehand, timehand, body)`



しかしこのままでは，10 秒で,1 回のアニメーションが終わってしまいます．また, コマ数が少なくて, ギクシャクしてます．そこで，`TimeRange = 0..60`, `Frames = 300` にします. これは，**timehand,minutehand**

に共通なので，次のように plot コマンドの中に入れます.

$$\bullet \texttt{plot}(\texttt{minutehand}, \texttt{timehand}, \texttt{body}, \texttt{TimeRange} = \texttt{0..60}, \texttt{Frames} = \texttt{300})$$

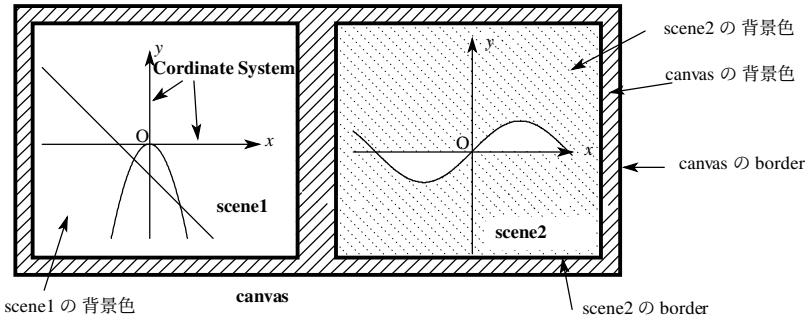
これで,

- minutehand := plot :: Arrow2d([0, 0], [3 ∗ sin(t), 3 ∗ cos(t)], TimeRange = 0..60, Frames = 300, ···)
- timehand := plot :: Arrow2d([0, 0], [2 ∗ sin(t), 2 ∗ cos(t)], TimeRange = 0..60, Frames = 300, ···)

としたのと同じになります．これは,

- plot(plot :: CoordinateSystem2d(minutehand, timehand, body, TimeRange = 0..60, Frames = 300))
- plot(plot :: Scene2d(minutehand, timehand, body, TimeRange = 0..60, Frames = 300))
- plot(plot :: Canvas(minutehand, timehand, body, TimeRange = 0..60, Frames = 300)) ···

などとしても同じです.



すなわち，Canvas, Scene2d, CoordinateSystem2d は，minutehand, timehand のオブジェクトを含んでいるので， その中で指定した *Attribute* は，全て minutehand, timehand にも反映されます．このように**各オブジェクトに共通の *Attribute* を指定したいときは，それらのオブジェクトを含んでいるオブジェクトの中で, まとめて *Attribute* を指定することができます．**

最後に仕上げとして, base(時計の枠)，center(中心) や label(数字) も追加して見ます．数字は, 別々に

- plot :: Text2d(”1”, [3 ∗ sin(PI/6), 3 ∗ cos(PI/6)]),
- plot :: Text2d(”2”, [3 ∗ sin(2 ∗ PI/6), 3 ∗ cos(2 ∗ PI/6)]), ···
- plot :: Text2d(”12”, [3 ∗ sin(12 ∗ PI/6), 3 ∗ cos(12 ∗ PI/6)]) ···(∗)

と指定してもよいですが, あまりにも面倒です．このような時，列生成子「$」を使うと便利です．(第 6 章参照)

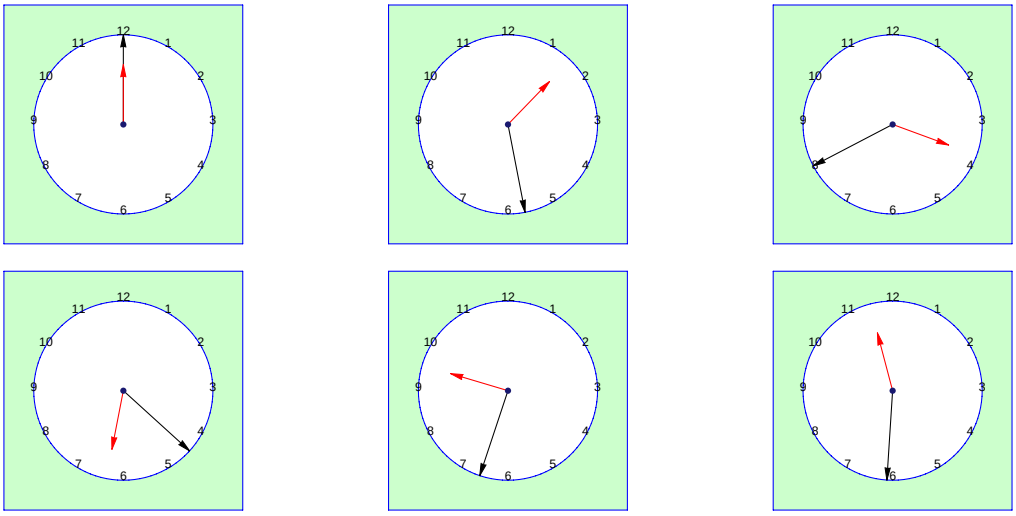
- labels := plot :: Text2d(expr2text(k), [3 ∗ sin(k ∗ PI/6), 3 ∗ cos(k ∗ PI/6)]) $k = 1..12 :

ここで，expr2text は，数式 (expression) を文字列 (text) に変換するコマンドです．これで，(∗) と同じになります．さらに,「本体，枠，中心」を新しく定義します．

- body := plot :: Circle2d(3, [0, 0], Filled = TRUE, FillColor = RGB :: White, FillPattern = Solid) :
- base := plot :: Rectangle(−4..4, −4..4,
  Filled = TRUE, FillColor = RGB :: LightGreen, FillPattern = Solid) :
- center := plot :: Point2d([0, 0], PointSize = 2) :

先の, 分針, 長針と合わせて plot します．

- plot(base, body, timehand, minutehand, center, labels, Axes = None)



ここで，Axes = *None* というのは,「座標軸を描かない」という option で，objects inspector を使って，*CoordinateSystem2d* からも指定することができます．

[進んだアニメーション]

いままでのアニメーションでは,「ある図形だけを，ある時間だけ表示させること」はできませんでした．これは，VisibleBeforeStart, VisibleAfterEnd を使えばできます．
今度は，デジタル時計を作ります．アニメーションパラメータを k に取って，

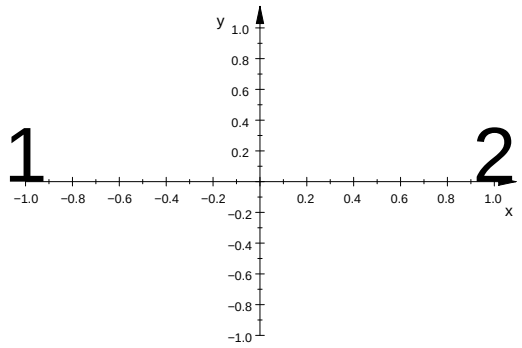
●plot :: Text2d(expr2text(k), [0, 0], k = 0..59)

でうまく行きそうな気がしますが，残念ながら，k の文字が，原点に表示されるだけです．そこで，簡単のため,「1」と「2」の文字を交互に表示するプログラムから作ります．

- label1 = plot :: Text2d("1", [−1, 0], TimeRange = 0..2) :　　···"1" の文字を (−1, 0) に表示
- label2 = plot :: Text2d("2", [1, 0], TimeRange = 2..4) :　　···"2" の文字を (1, 0) に表示

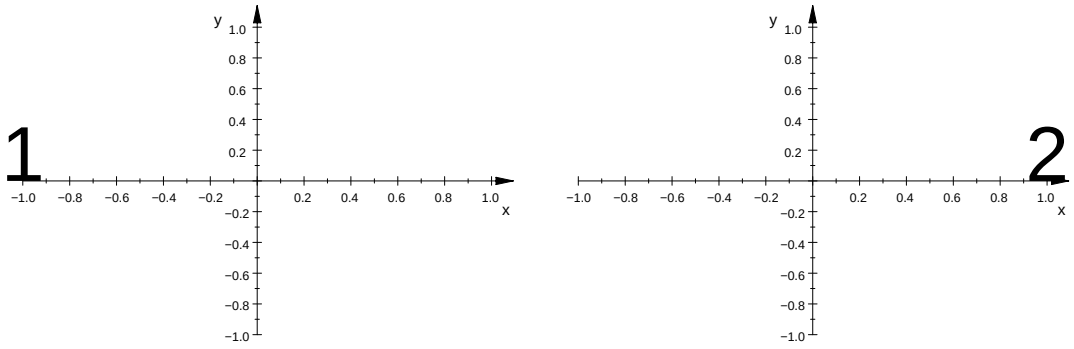
Text の "1"や"2"は定数なので，アニメーションパラメータは必要ありません．スタートからの時間を t 秒とすると，label1 は，$0 \leqq t \leqq 2$ のとき表示し，label2 は，$2 \leqq t \leqq 4$ のとき表示するはずです．これを TextFont= [50] の大きいサイズにして plot してみます．

- plot(label1, label2, TextFont = [50])



ご覧のように，”1” と ”2” の両方の文字が見えてしまいます．これは，default では，VisibleBeforeBegin (TimeRange に入る前にも表示) と VisibleAfterEnd(TimeRange から出た後も表示) が，共に *TRUE* になっているためです．そのため，TimeRange に入っていなくとも，図形が図示されてしまいます．したがって，これらを, *FALSE* にすると, うまく行きます.

•plot(label1, label2, TextFont = [50], VisibleBeforeBegin = FALSE, VisibleAfterEnd = FALSE)



これで最初の 2 秒間は ”1” の数字だけ, 後の 2 秒間は ”2” の数字だけ見えます．これを 60 個やれば, 秒針の出来上がりですが，ちょっと大変なので 列生成子「$」を使います.

まず，*x*(expression) を, ”x”(string) に直す関数: f を作ります.

•f := x −> expr2text(x) >> $x \to$ expr2text(x)

これで，例えば，• f(3) >> ”3” のように，数字の 3 が文字の ”3” になります．次に秒針の列を作ります.

• seconds := plot :: Text2d(f(k), [0, 0], TimeRange = k..k + 1) $k = 0..59

>> plot::Text2d(”0”, [0, 0]), plot::Text2d(”1”, [0, 0]), plot::Text2d(”2”, [0, 0]), ···

上には表示されていませんが，順に TimeRange = 0..1, TimeRange = 1..2, ··· となっているはずです.

• plot(seconds, VisibleBeforeBegin = FALSE, VisibleAfterEnd = FALSE,
TextFont = [50], Axes = None)

これで, 次のように，デジタル秒時計ができます．これは正確な時計です.

0　　7　　33　　59

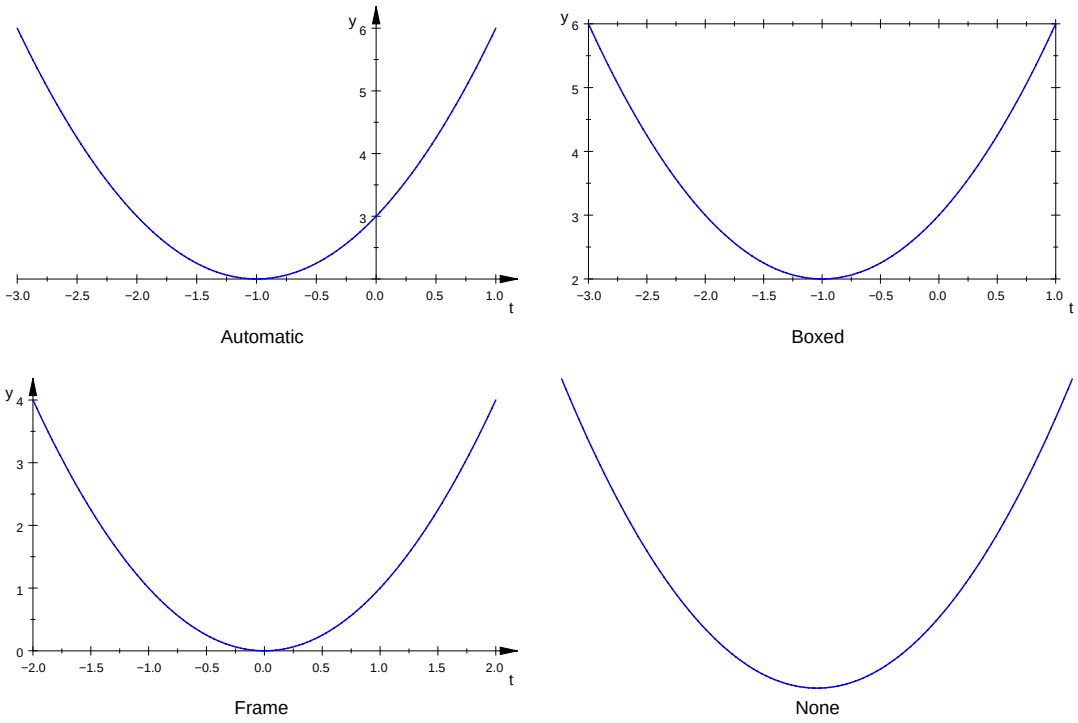
同様にして，時針, 分針も作れますが, 省略します.

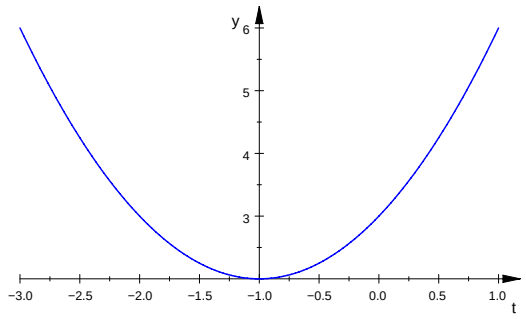
[座標軸 (axes) の *Attribute*]

CoordinateSystem2d では，`Axes` の表示の仕方は, *Automatic*, *Boxed*, *Frame*, *Origin*, *None* の 5 種類があります (default では Axes= *Automatic*)．この違いを `plotfunc2d` を使って，見てみます.

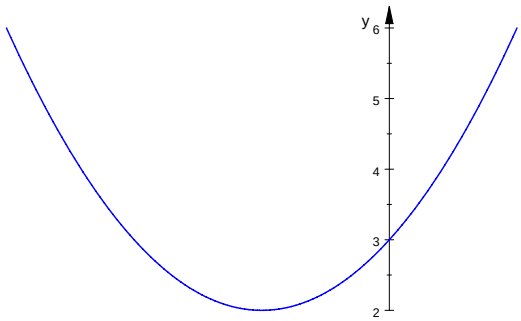
- `plotfunc2d(t^2 + 2 * t + 3, t = -3..1, Axes = Automatic, Footer = "Automatic")`
- `plotfunc2d(t^2 + 2 * t + 3, t = -3..1, Axes = Boxed, Footer = "Boxed")`
- `plotfunc2d(t^2 + 2 * t + 3, t = -3..1, Axes = Frame, Footer = "Frame")`
- `plotfunc2d(t^2 + 2 * t + 3, t = -3..1, Axes = None, Footer = "None")`
- `plotfunc2d(t^2 + 2 * t + 3, t = -3..1, Axes = Origin, AxesOrigin = [-1, 2], Footer = "AxesOrigin = [-1, 2]")`
- `plotfunc2d(t^2 + 2 * t + 3, t = -3..1, Axes = Origin, Footer = "Origin")`

順に下の図のようになります．Origin は, 普通は AxesOrigin= $[x_0, y_0]$ とともに使用されると思われます. これで座標軸の交点が $[x_0, y_0]$ になります．(origin は,「原点」の意味)





AxesOrigin=[−1,2]



Origin

11.3 様々なグラフ (high-level primitives)

幾何図形のような”low-level primitive” と異なり，plotfunc2d のように, 関数式を指定していろいろなグラフを描かせることもできます．こちらは，”high-level primitive” と呼びます．この節では，high-level primitive を取り上げます．下が，2 次元 (平面) の比較的基本的と思われる high-level primitive です．

plot :: Function2d	$y = f(x)$ のグラフ (陽関数表示)	Function2d(f(x), x = x_1..x_2)
plot :: Implicit2d	$f(x, y) = 0$ のグラフ (陰関数表示)	Implicit2d(f(x, y), x = x_1..x_2, y = y_1..y_2)
plot :: Curve2d	曲線のパラメータ表示	Curve2d([x(t), y(t)], t = t_1..t_2)
plot :: Hatch	fとgで囲まれた領域表示	Hatch(f, g)
plot :: Hatch	閉曲線Cで囲まれた領域表示	Hatch(C)

注21)

11.3.1 $y = f(x)$ のグラフ (plot::Function2d)

Function2d は，$y = f(x)$ のグラフを描く時に使います．例えば，$y = x^2(-2 \leqq x \leqq 2)$ のグラフは，次のようにします.

- f := plot :: Function2d(x^2, x = −2..2)
- plot(f)

これは，plotfunc2d を用いて, 次のようにしても同じです．(property inspector の表示も同じです.)

- plotfunc2d(x^2, x = −2..2)

注21) この他,

統計用グラフ plot::Bars2d,plot::Boxplot,plot::Histogram2d, plot::Piechart2d
密度グラフ plot::Density
不等式の領域表示 plot::Inequality
漸化式のグラフ表現 plot::Iteration
微分方程式の解の表示 plot::Ode2d
写真の表示 plot::Raster
ベクトル場 plot::VectorField2d
複素関数の表示 plot::Conformal
その他 plot::Lsys(Lindenmayer system), plot::Turtle(turtle plot) があります.

すなわち，plotfunc2d(f, x = x_1..x_2) は，plot(plot :: Function2d(f, x = x_1..x_2) の簡易表現です．Function2d は, 他の primitive と組み合わせる場合に用います.

11.3.2　$f(x, y) = 0$ のグラフ (plot::Implicit2d)

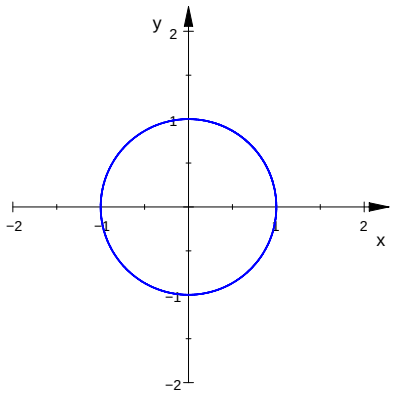
円の方程式;$x^2+y^2-r^2 = 0$ のように $f(x, y) = 0$ の形をした関数は，$y = f(x)$ の形に直して plotfunc2d, Function2d をつかうか，または plot :: Implicit2d を使って描きます．[注22)]
$f(x, y) = 0$ $(a \leqq x \leqq b, c \leqq y \leqq d)$ のグラフのオブジェクトは,

plot :: Implicit2d(f(x, y), x = a..b, y = c..d)

です. 例えば，円を描いてみます．$x^2 + y^2 = 1 \Longleftrightarrow x^2 + y^2 - 1 = 0$ ですから次のようにします．変域は $-1 \leqq x \leqq 1, -1 \leqq y \leqq 1$ を含んでいれば大丈夫です.

- mycircle := plot :: Implicit2d(x^2 + y^2 − 1, x = −2..2, y = −2..2, Scaling = Constrained) :
- plot(mycircle)

[注23)]



Implicit2d では，$f(x, y) = 0$ がどのような範囲に表示されようが，指定した領域を表示します．また，Scaling = *Constrained* としないと, 円がつぶれてしまいます．[注24)]

11.3.3　パラメータ表示された曲線:$x = f(t), y = g(t)$ のグラフ (plot :: Curve2d)

$x = f(t), y = g(t)(a \leqq t \leqq b)$ とパラメータ表示されたグラフのオブジェクトは

plot :: Curve2d([f(t), g(t)], t = a..b))

です.

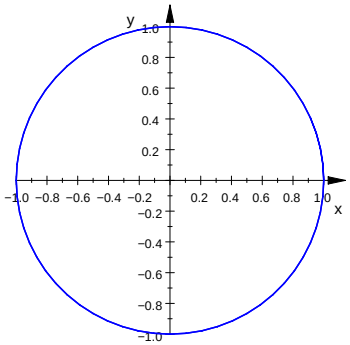
[注22)] $f(x, y) = 0$ で表される関数を陰関数 (implicit function) といいます. これに対し $y = f(x)$ で表される関数を陽関数 (explicit function) といいます.
[注23)] このとき、x^2 + y^2 = 1 とうっかりやってしまいがちです．御注意ください.
[注24)] properties inspector を開いて，*CoordinateSystem2d* の Definition から Scaling = *Constrained* と変更することもできます.

[円のパラメータ表示]

円 $(x^2 + y^2 = 1)$ をパラメータ表示： $x = \cos t, y = \sin t$ $(0 \leqq t \leqq 2\pi)$ を利用して，描いてみます.

- `mycircle := plot::Curve2d([cos(t), sin(t)], t = 0..2 * PI):`
- `plot(mycircle, Scaling = Constrained)`

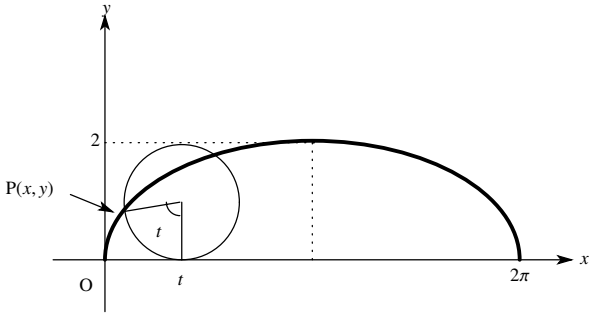
これで, 次のようになります.



[サイクロイドの静止画]

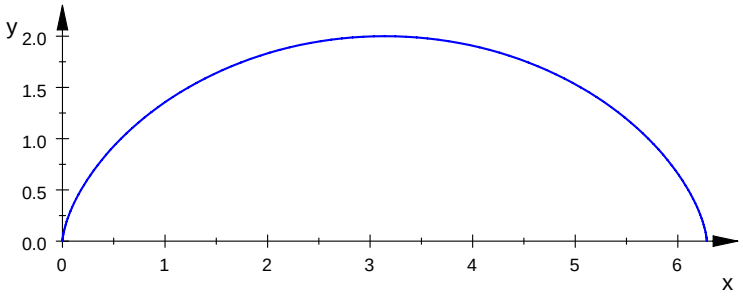
次はサイクロイドです．これは半径 1 の円を直線上に転がしたときに出来る図形です．右図から, サイクロイドのパラメータ表示は次のようになります.

$$\begin{cases} x = t - \sin t \\ y = 1 - \cos t \end{cases}$$

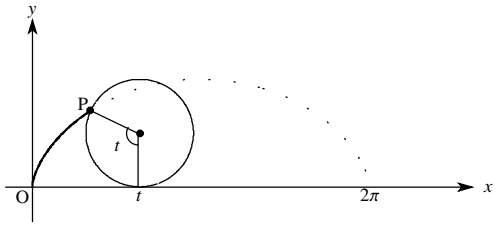


これを MuPAD でかいて見ます.

- `mycycloid := plot::Curve2d([t − sin(t), 1 − cos(t)], t = 0..2 * PI):`
- `plot(mycycloid, Scaling = Constrained)`



11.3.4　[サイクロイドのアニメーション]



半径 1 の円上の点 P が, はじめに原点にあったとし，その円が毎秒 1 の速度で，x 軸正方向に動いているとします．時刻 t において，円の中心も $(t, 1)$ になるので,

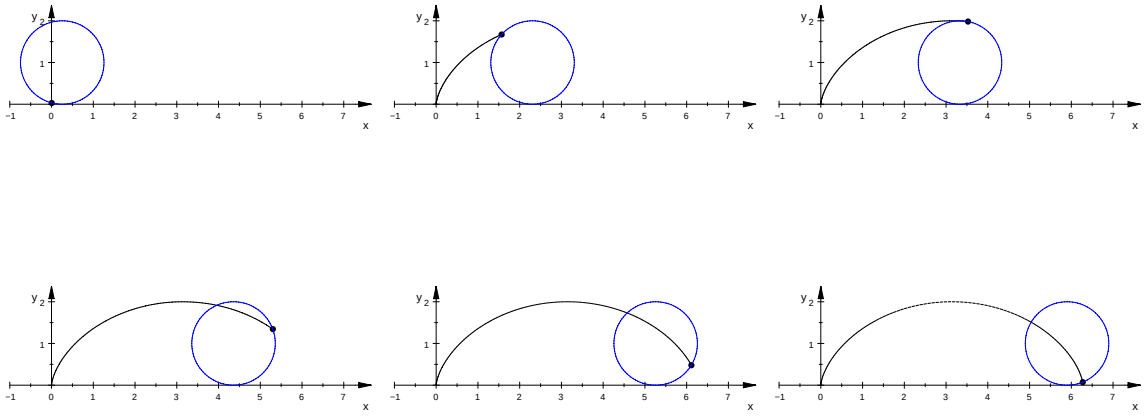
$$\begin{cases} \text{中心 } (t,1), \text{半径 1 の円}: & (x-t)^2+(y-1)^2=1 \\ \text{サイクロイドの}\textbf{一部}(O \text{ から } P \text{ まで}): & x=u-\sin u, y=1-\cos u\ (0 \leqq u \leqq t) \\ \text{点 } P: & x=t-\sin t, y=1-\cos t \end{cases}$$

となります．したがって, 次のようにすれば,$0 \leqq t \leqq 2\pi$ におけるそれぞれの図形のアニメーションが表されます.

- `mycircle := plot :: Circle2d(1,[t,1],t = 0..2 * PI,Color = RGB :: Blue) :`
- `mycycloid := plot :: Curve2d([u - sin(u),1 - cos(u)],u = 0..t,t = 0..2 * PI)`
- `mypoint := plot :: Point2d([t - sin(t),1 - cos(t)],t = 0..2 * PI)`

これを, まとめて `plot` します.

- `plot(mycircle,mycycloid,mypoint,Color = RGB :: Black)`



原則として，$\texttt{plot}(\texttt{object}_1, \texttt{object}_2, \cdots, \texttt{Color} = \texttt{RGB :: Black})$ とすれば, 全てのオブジェクトが黒になりますが，円だけが黒色になっていないのは，$\texttt{mycircle} := \texttt{plot :: Circle2d}(\cdots, \texttt{Color} = \texttt{RGB :: Blue})$ と個別に指定したためです．すなわち，ある「オブジェクト a 」を含んでいる「オブジェクト A」で指定した *Attribute* は,「オブジェクト a 」に対しても有効ですが，**「オブジェクト a 」 で個別に定めた内容を上書きすることはありません．**

11.4 平面の一次変換

MuPAD では, 平面の一次変換：

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

によって, 様々な 図形, グラフ (primitive) を一次変換できます．また一度に複数の primitive を変換することもできます．[注25)]

plot :: Translate2d	$\overrightarrow{d} = (d_x, d_y)$ だけ平行移動 Translate2d([dx, dy], primitive$_1$, primitive$_2$, $\cdots$)
plot :: Scale2d	x, y 軸方向に，それぞれ s_x, s_y倍に拡大 Scale2d([sx, sy], primitive$_1$, primitive$_2$, $\cdots$)
plot :: Rotate2d	点 C の周りにθ回転 Rotate2d(θ, [Cx, Cy], primitive$_1$, primitive$_2$, $\cdots$)
plot :: Transform2d	行列 A による一次変換 Transform2d(A, primitive$_1$, primitive$_2$, $\cdots$)
	$\overrightarrow{d} = (d_x, d_y)$ 平行移動させたあと，行列 A による一次変換 Transform2d([dx, dy], A, primitive$_1$, primitive$_2$, $\cdots$)

注26)

11.4.1 最初の例

plot :: Ellipse2d では，長軸, 短軸が x, y 軸と平行な楕円しか描けませんが，plot :: Rotate2d を使うと，傾いた楕円なども描けます.

まず，長軸半径の長さが 4, 短軸半径の長さが 2, 中心が原点の楕円 original を, 作ります．合わせて, 色を変えただけの楕円も作っておきます.

- original := plot :: Ellipse2d(2, 1, [0, 0], LineColor = RGB :: Black, LineStyle = Dashed) :
- originBlue := plot :: Ellipse2d(2, 1, [0, 0], LineColor = RGB :: Blue) :
- originRed := plot :: Ellipse2d(2, 1, [0, 0], LineColor = RGB :: Red) :
- originGreen := plot :: Ellipse2d(2, 1, [0, 0], LineColor = RGB :: Green) :

注25) 空間の一次変換もできます．次章を見てください.

注26) 1. plot :: Rotate2d で，$[C_x, C_y]$ を省略すると，$C(0, 0)$ となります.
2. 行列は, MuPAD の 2×2 matrix (行列), または，2×2 array , または，4 個の要素からなる list で表します．すなわち,

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

の行列を，MuPAD で作成するには,

$$A := \text{matrix}([[a_{11}, a_{12}], [a_{21}, a_{22}]])$$

のように, 各行をリスト ([] で挟む) にして, それをさらにリストの中に入れ, matrix を使って作ります．しかし，plot :: transform で使う場合は，単に，リストでも良く,

$$A := [[a_{11}, a_{12}], [a_{21}, a_{22}]] \quad \text{や} \quad A := [a_{11}, a_{12}, a_{21}, a_{22}]$$

でも大丈夫です.

青色の楕円 originBlue を，原点の周りに，$\frac{\pi}{4}$ 回転した楕円 rotated を作ります.

- rotated := plot :: Rotate2d(PI/4, originBlue) :　　… originBlue を $\frac{\pi}{4}$ 回転した楕円

次は，$\frac{\pi}{6}$ の回転を表す行列 $A = \begin{pmatrix} \cos\frac{\pi}{6} & -\sin\frac{\pi}{6} \\ \sin\frac{\pi}{6} & \cos\frac{\pi}{6} \end{pmatrix}$ を，リストで表します.

- A := [[cos(PI/6), −sin(PI/6)], [sin(PI/6), cos(PI/6)]] :

A を使って，赤色の楕円 originRed を，一次変換します.

- transformed := plot :: Transform2d(A, originRed) :　　… originRed を A で一次変換した楕円

緑色の楕円 originGreen を，$\overrightarrow{d} = (5, 0)$ 平行移動した楕円 translated を作ります.

- translated := plot :: Translate2d([5, 0], originGreen) :
　　… originGreen を (5, 0 平行移動した楕円

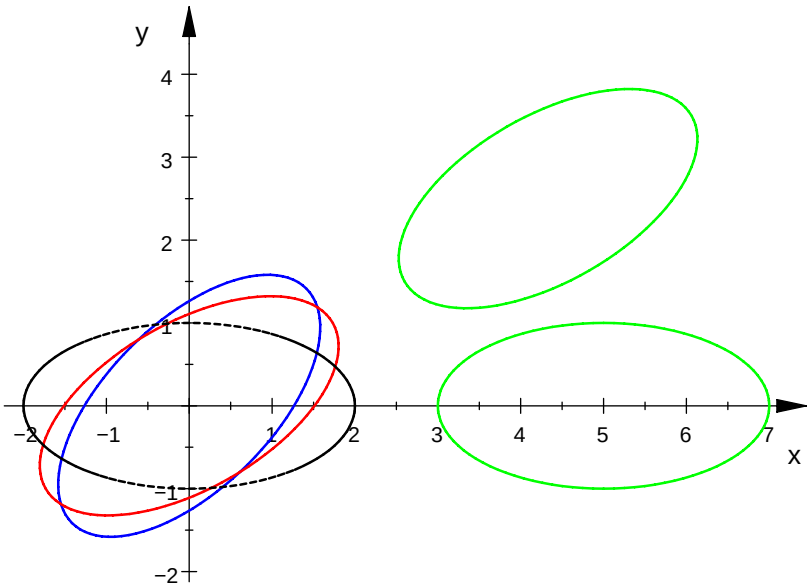
translated を原点の周りに $\frac{\pi}{6}$ 回転した楕円 composit を作ります.

- composit := plot :: Rotate2d(PI/6, translated) :　　… translated を，$\frac{\pi}{6}$ 回転した楕円

以上の図形を，まとめて描写します.

- plot(original, rotated, transformed, translated, composit)

これで次のようになります.



11.5 特殊なプロット

11.5.1 領域の表示 (plot :: Hatch)

plot :: Hatch	2 曲線で囲まれた領域表示	Hatch(f, g)
plot :: Hatch	曲線 f とy=k(k は定数)で囲まれた領域表示	Hatch(f, k)
plot :: Hatch	曲線 f とy=0(x軸) で囲まれた領域表示	Hatch(f)
plot :: Hatch	パラメータ表示曲線Cで囲まれた領域表示	Hatch(C)

plot :: Hatch は,MuPAD 3.0 になって新しく導入された primitive です．曲線によって囲まれる領域を, 塗りつぶします．なお，上の表で，f, g は plot :: Function2d の object, C は plot :: Curve2d の object です．C が閉曲線でないときは，MuPAD は，勝手に閉曲線に直して, 塗りつぶします．また，plot :: Hatch は, 境界線は図示しません．[注27)]

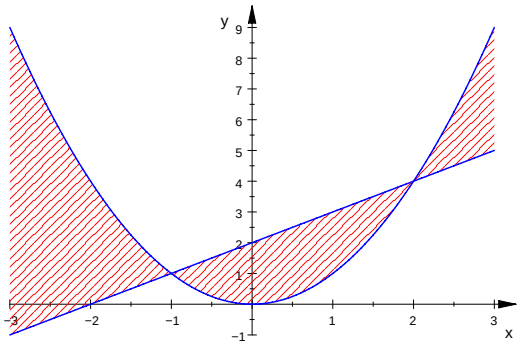
例として，2 曲線: $y = x^2$ と $y = x + 2$ で囲まれた領域を図示します．まず，2 つのグラフの オブジェクト を作成します.

- f := plot :: Function2d(x^2, x = −3..3) :
- g := plot :: Function2d(x + 2, x = −3..3) :

つぎに，plot を使って，描写します. このとき，f, g も plot しないと境界が描かれません.

- plot(plot :: Hatch(f, g), f, g)

これで図のようになります.

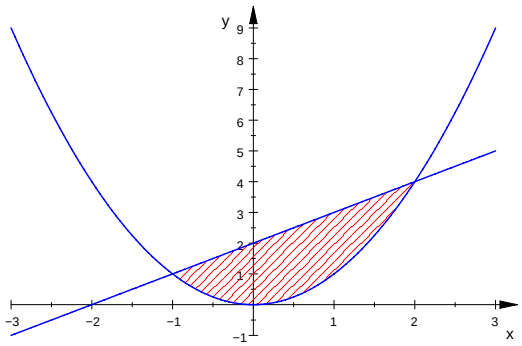


さらに,閉曲線で囲まれている領域のみを表示したい時は，別に交点の x 成分を求めて，plot :: Hatch(f, g, x_1..x_2) のようにします．上の例では，交点の x 成分は, $x = -1, x = 2$ ですから, 次のようにします.

- plot(plot :: Hatch(f, g, −1..2), f, g)

これで図のようになります.

注27) hatch は「斜線」という意味です．また，plot :: Hatch は，high-level primitive の一つですが, すこし，特殊だと思ったので, 他の high-level primitive とは別に，説明します.

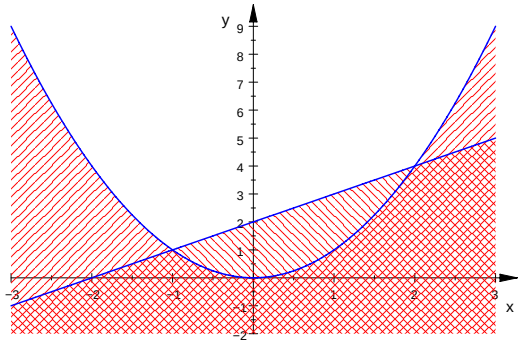


$y = f(x)$ の上側, 下側のような領域を図示するには，$y = k(k$ は定数) と $y = f(x)$ で囲まれた領域と考えて図示します．例えば，領域:$y \leqq x + 2$, かつ, $y \leqq x^2$ を図示するには, 次のようにします.

- `plot(plot :: Hatch(f, −2), plot :: Hatch(g, −2, FillPattern = FDiagonalLines), f, g)`

ここで,FillPattern= *FDiagonalLines* というのは, 斜線のタイプを指示しています．これらも property inspector から, 変更できます.

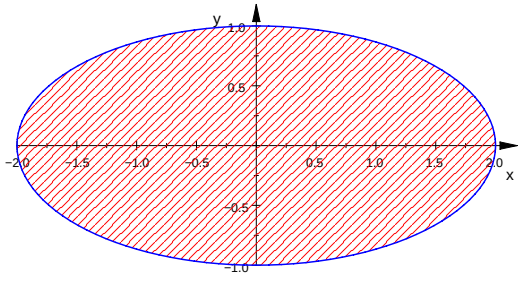
これで図のようになります.



さらに，`plot :: Curve2d` で描かれた図形で囲まれた領域も図示できます．このとき, もし曲線が閉じてない場合は,MuPAD の方で，始点と終点を結んで, 閉じた図形に直してから, 図示します.
楕円：$x = 2\cos t, y = \sin t\ (0 \leqq t \leqq 2\pi)$ で囲まれた領域を図示して見ます.

- `c := plot :: Curve2d([2 * cos(t), sin(t)], t = 0..2 * PI) :`
- `plot(plot :: Hatch(c), c, Scaling = Constrained)`

これで図のようになります.



注28)

注28) これは，low-level primitive の `Ellipse2d` を使って, 次のようにしても同じです.

- `plot(plot :: Ellipse2d(2, 1, [0, 0], Filled = TRUE))`

また，`plot :: Inequality` を使っても, 不等式で表された領域を表示できますが, 目が荒くなります.

第12章　空間のグラフ

12.1　$y = f(x,y)$ のグラフ (plotfunc3d)

$z = f(x,y)$ のグラフ ($a \leqq x \leqq b$, $c \leqq y \leqq d$)	plotfunc3d(f(x,y), x = a ..b, y = c..d)
$z = f(x,y)$ のアニメーション	plotfunc2d(f(x,y,t), x = a ..b, y = c..d , t = t_1..t_2)

$z = f(x,y)$ のグラフを描くのは plotfunc3d を使うのが, 最も簡単です. [注1)]

また，MuPAD 3.0 になってから,z の範囲も,ViewingBoxZRange = z_1..z_2 で指定できるようになりました. (plotfunc3d では,ZRange = z_1..z_2 でも大丈夫です.)

plotfunc3d(f(x), g(x) , x = x_1 ..x_2, y = y_1 ..y_2, ViewingBoxZRange = z_1..z_2)

またこれらの *Attribute* も，plotfunc2d と同じく，property inspector から編集できます．plotfunc2d と plotfunc3d は, 次数が変わることを除けばほとんど同じですが，plotfunc2d は,$f(x)$ の値が実数値にならないときは, 単に無視するだけなのに対し，plotfunc3d では,$f(x,y)$ の値が（ある領域で）実数にならないときは，エラーになるという違いがあります.

[$z = f(x,y)$ のグラフとは何か？]

$z = f(x,y)$ のグラフに馴染みがない方は，次の高校生の問題を考えてみてください.

実数 x, y が独立にすべての実数値をとり変化するとき、

$$z = x^2 - 2xy + 2y^2 - 4y + 5 \qquad \cdots (*)$$

の最小値を求めよ.

[注1)] plot(plot :: Function3d(f, x = x_1..x_2, y = y_1..y_2)) の省略が，plotfunc3d(f, x = x_1..x_2, y = y_1..y_2) です.
さらに，plot(plot :: Function3d(f, $\cdots$)) は,

plot(plot :: Canvas(plot :: Scene3d(plot :: CoordinateSystem3d(
plot :: Function3d(f, $\cdots$)))))

の省略です.

【解答】 x に関し平方完成して

$$z = (x-y)^2 + y^2 - 4y + 5$$

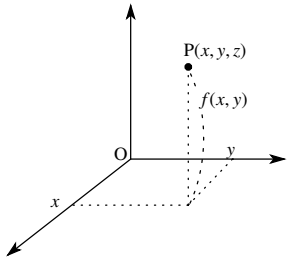
さらに y に関し平方完成して

$$z = (x-y)^2 + (y-2)^2 + 1 \geqq 1$$

等号は $x = y$, かつ $y = 2 \Longleftrightarrow x = y = 2$ のとき成立するので、最小値は

$$2 \qquad \cdots \text{(答)}$$

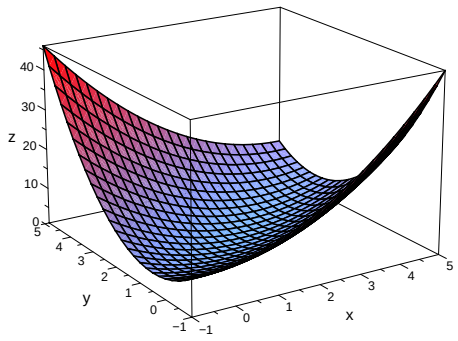
このような時,$z = f(x,y) = x^2 - 2xy + 2y^2 - 4y + 5$ のグラフが頭にあれば問題の意味がよく解ります．一般に $z = f(x,y)$ の式が与えられたとき、x, y が決まれば $\mathrm{P}(x, y, f(x,y))$ という点が定まり, (x,y) が xy 平面上を動くとき，点 P はある曲面を動きます．このグラフで z 成分の最小値が，$f(x,y)$ の最小値になるので，グラフの一番低い点を見ればよいことが解ります.



12.1.1　[初めての plotfunc3d(ViewingBox の設定)]

さっそく
$z = f(x,y) = x^2 - 2xy + 2y^2 - 4y + 5$ のグラフを描いてみます.

- `plotfunc3d(x^2 − 2 ∗ x ∗ y + 2 ∗ y^2 − 4 ∗ y + 5, x = −1..5, y = −1..5)`

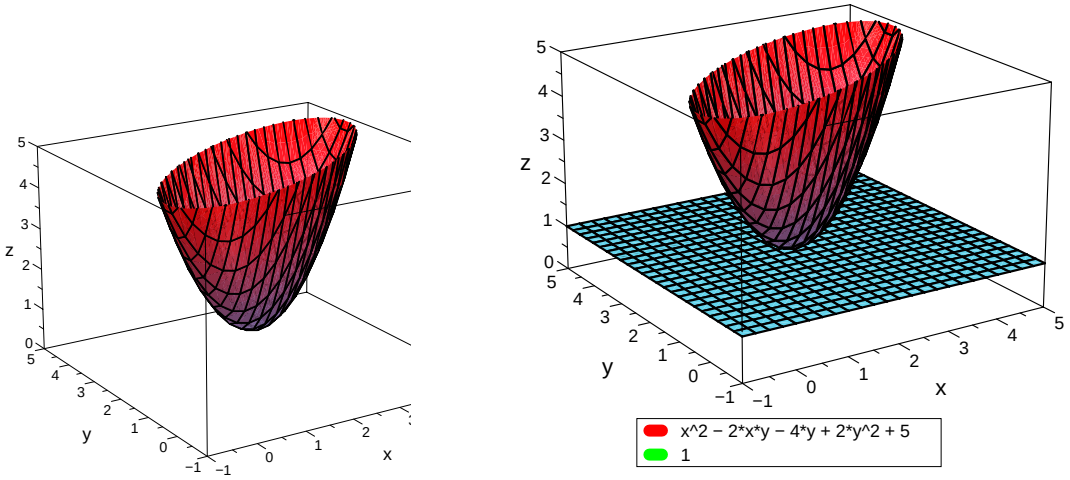


$x = y = 2$ の前後で z 成分が最小となるはずですが，グラフからは良くわかりません．そこで，ViewingBoxZRange(または，ZRange) の範囲を変えて見ます.

- `plotfunc3d(x^2 − 2 ∗ x ∗ y + 2 ∗ y^2 − 4 ∗ y + 5, x = −1..5, y = −1..5, ZRange = 0..5)`

さらに,$z = f(x,y)$ の最小値は 1 になるはずなので，$z = 1$ のグラフも重ねて描いてみます．コンマで区切って入れるだけです.

- `plotfunc3d(x^2 − 2 ∗ x ∗ y + 2 ∗ y^2 − 4 ∗ y + 5, 1, x = −1..5, y = −1..5, ZRange = 0..5)`



どうやら, 接しているみたいです.

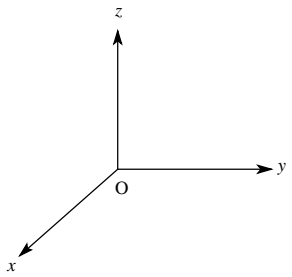
12.1.2 対話的な *Attribute* の変更 (CameraPosition，Grid, Axes, Mesh)

`plotfunc2d` と同様，`plotfunc3d` も, 対話的 (interactive) に, *Attribute* を変更することができます．特に, マウスを使って自由に回転や, 拡大ができます.

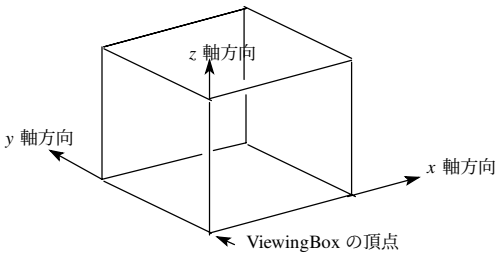
まず，notebook ではグラフ上で右クリックして,「Graphics オブジェクト」⟶ *Open* で, Vcam が表示されます．その上部にメニューバーと，object brouser, と property inspector が見えます.（見えない場合は, メニューバーの *View* → *Objects*, *Properties* と辿り, チェックを付けてください．) あるいは，グラフをダブルクリックすれば，notebook に埋め込まれて，property inspector と，interactive viewer ，その上に（変更された）メニューバーがみえます.

左から，Select(2 つ以上の object があるときに，それぞれの object を選択), Rotate(回転), Zooming(ズーム), Moving(移動), Query(座標表示), RotateLeft(左回転), RotateRight(右回転), RotateDown(下方回転), RotateUp(上方回転), ZoomIn(ズームイン), ZoomOut(ズームアウト), ActionSpeed(動作速度) です．最初は, Rotate になっているので，canvas をマウスでドラッグすれば，グラフは上下左右に自由に回転します. その横の Zooming をクリックし，canvas 上でマウスで上 ⟷ 下すると，ZoomIn⟷Out します．こちらは手動の Zooming です．右の方にも, 回転のボタンや，ZoomIn⟷Out のボタンがありますが, こちらは, 自動で動き続けます．止めるには, もう一度押します．ActionSpeed は, その速度を制御します．[注2)] はじめに表示される画面は、高校の教科書とは違う視点から見ています．(x, y 軸をそれぞれ一塁・三塁方向とすると,MuPAD はバックネットの裏から見た感じです．これに対し，教科書は，ライト方向から見ている感じです．)

注2) このような動きは，MuPAD の内部では，MuPAD のカメラ相当品である `plot :: camera([px, py, pz], [fx, fy, fz], angle)` を変更して実現しています．とくに `plotfunc3d` では,「カメラの位置」を表すパラメータ: $[p_x, p_y, p_z]$ を変更して実現されています.

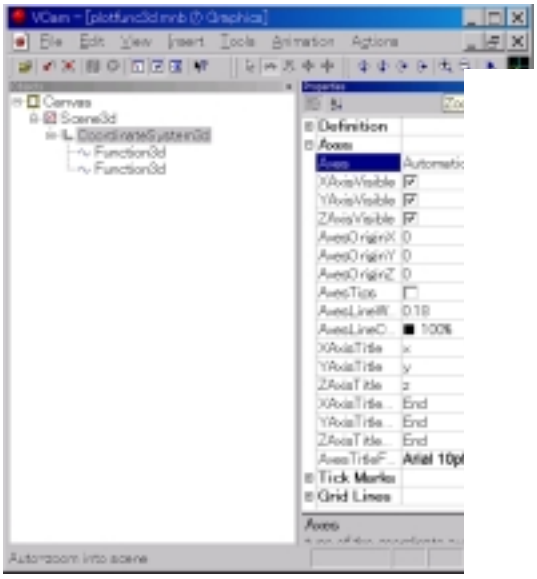


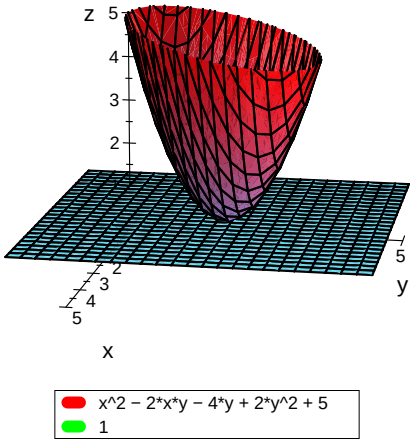
【高校の教科書の視点】



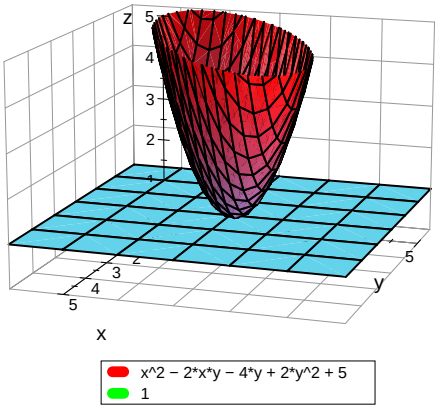
【MuPAD の視点 (最初)】

これを回転させて, 教科書のような向きに変えます．さらに，CoordinateSystem3d の Axes で，Axes = *Automatic*,XAxisTitleAlignment = *End*, YAxisTitleAlignment, ZAxisTitleAlignment も同様 *End* に設定します．これで，教科書のようなグラフが得られます.





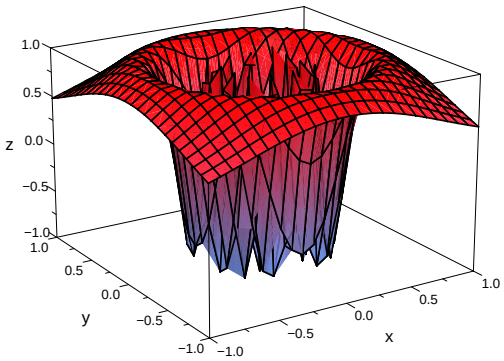
さらに, 同じく CordinateSystem3d の Grid Lines を選択し，XGridVisible = *TRUE*,YGridVisible = *TRUE*, ZGridVisible = *TRUE* にします．あわせて，平面の 網の目 (mesh) が細かすぎるので，幅を 1 にします．そのためには，指定した区間 ($-1 \leqq x \leqq 5, -1 \leqq y \leqq 5$) を 6×6 の区間に分割すれば良いので，2 番目の Function3d の Calculate で，XMesh = YMesh = 7 に変更します.
すると図のようになります．図からも，$(x, y) = (2, 2)$ の近くで最小値をとりそうなことが分ります．(さらに確認したいならば, グラフを ”ひっくり返して, 底を見ると” もっとよく分かります．また，メニューバーの *Query* をクリックして，マウスを移動し，点の座標を得ることもできます．)



12.1.3 ぎざぎざ関数の表示 (Mesh の設定)

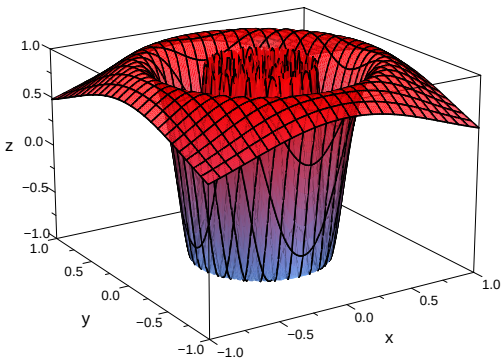
plotfunc2d の $y = \sin\frac{1}{x}$ のグラフと同様に,$z = \sin\frac{1}{x^2+y^2}$ も, **ぎざぎざ**のグラフとなります.

- plotfunc3d(sin(1/(x^2 + y^2)), x = −1..1, y = −1..1)



これを, 滑らかにするには, XMesh,YMesh を増やすか, XSubmesh,YSubmesh の値を増やします. [注3)] まずは, XSubmesh,YSubmesh を増やしてみます.

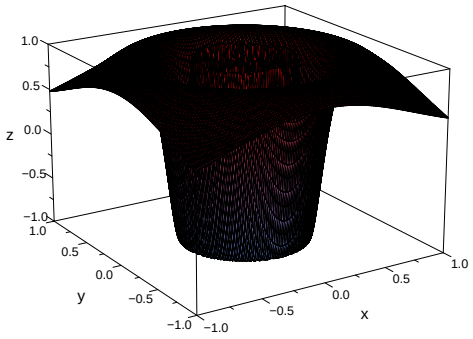
- plotfunc3d(sin(1/(x^2 + y^2)), x = −1..1, y = −1..1, XSubmesh = 5, YSubmesh = 5)



[注3)] AdaptiveMesh を増やしても大丈夫ですが, 空間では, Mesh,Submesh を設定する方が推薦されています.

XSubmesh= 5,YSubmesh= 5 によって，$-1 \leqq x \leqq 1, -1 \leqq y \leqq 1$ 内の点の数は, それぞれほぼ 6 倍になるので，XMesh= 150,YMesh= 150 としても同じようなグラフが得られます．[注4)]

- `plotfunc3d(sin(1/(x^2 + y^2)), x = -1..1, y = -1..1, XMesh = 150, YMesh = 150)`



しかし，XMesh, YMesh を増やすと，実際に, 網の目 (mesh) が, グラフ上に表示されます．一方，XSubmesh, YSubmesh を増やしても，内部で計算を細かくするだけで，グラフ上の網の目は，変化ありません．

12.1.4　北半球（$f(x, y)$ の値が虚数になる場合）

最後に, 半球 $(x^2 + y^2 + z^2 = 1, z \geqq 0)$ を描いて見ます．実は，low-level primitive を使えばすぐ描けるのですが，ここでは，z に関し解いて，plotfunc3d を使って描いてみます．z に関し解くと，

$$x^2 + y^2 + z^2 = 1 \Longleftrightarrow z^2 = 1 - x^2 - y^2 \Longleftrightarrow z = \pm\sqrt{1 - x^2 - y^2}$$

ですから, 球の上半分 (北半球) を描くために，次のように入力します．

- `plotfunc3d(sqrt(1 − x^2 − y^2), x = −1..1, y = −1..1)`

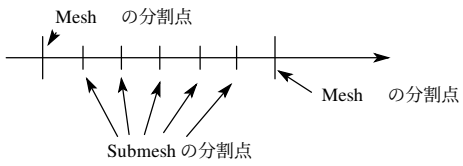
しかし，次のような表示が出ます．

>> Error: cannot evaluate to a real numerical value near the point (−1.0, −1.0) [plot::surfaceEval]

つまり $\sqrt{\ }$ のなかが負になってしまい，$f(x, y)$ の値が虚数になってしまったのです．plotfunc2d の場合は，$f(x)$ が虚数になる区間は無視して，残りの区間だけを描写しますが，plotfunc3d の場合は, ある**領域**で $f(x, y)$ が虚数になるときは，Error を出して, 全く描写しません．[注5)]
そこで, piecewise[注6)] を使って，g を定義します．

- `g := piecewise([1 − x^2 − y^2 >= 0, sqrt(1 − x^2 − y^2)], [1 − x^2 − y^2 < 0, 0])`

$$>> \begin{cases} \sqrt{1 - y^2 - x^2} & if \quad 0 \leqq 1 - y^2 - x^2 \\ 0 & if \quad 1 - y^2 - x^2 < 0 \end{cases}$$



注4) XSubmesh = YSubmesh = 5 のとき，XMesh, YMesh で設定されている点の間に, 新しく，5 個の点が加わります．
一般に，XMesh = YMesh = n, XSubmesh = YSubmesh = m は，XMesh = YMesh = $(n-1)\cdot(m+1)+1$, XSubmesh = YSubmesh = 0 と同等です．

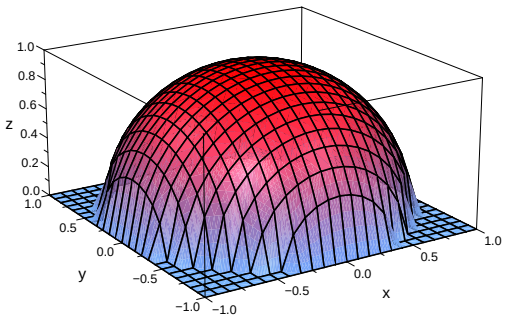
注5) $z = \frac{1}{x^2+y^2}$ も，$x = y = 0$ において z が虚数になりますが，1 点でのみ虚数になるだけなので, 問題ありません．

注6) ([不等式$_1$, 関数$_1$], [不等式$_2$, 関数$_2$], $\cdots$) の形で使います．

これを，Scaling= *Constrained* をつけて，表示させます.

- `plotfunc3d(g, x = −1..1, y = −1..1, Scaling = Constrained)`

すると、次の図のようになります.



当たり前ですが $z = 0$ も描かれてしまいます．球を描くには low-level primitive を使うと早くかけます．この他，`plotfunc3d` も，`plotfunc2d` と同様，Footer,Header,Title をつけたり，グラフの色の変更などができます．その方法も, `plotfunc2d` と同様なので省略します.

12.2 空間の幾何図形 (3d low-level primitive)

MuPAD 3.0 では, 円, 楕円, 長方形, 円錐, 正多面体などの複雑な幾何図形をすぐ描くことができます．このような幾何図形は”low-level primitive” と呼びます．3 次元 (空間) の low-level primitive の主なものは, 以下のとおりです.

`plot :: Arrow3d`	矢印	`Arrow3d`$([x_1, y_1, z_1], [x_2, y_2, z_2])$
`plot :: Box`	直方体	`Box`$(x_1..x_2, y_1..y_2, z_1..z_2)$
`plot :: Circle3d`	円	`Circle3d`$(r, [C_x, C_y, C_z], [n_x, n_y, n_z])$
`plot :: Cone`	円錐	`Cone`$(r_{base}, [C_x, C_y, C_z], [D_x, D_y, D_z])$
`plot :: Cone`	円錐台	`Cone`$(r_{base}, [C_x, C_y, C_z], r_{top}, [D_x, D_y, D_z])$
`plot :: Cylinder`	円柱	`Cylinder`$(r, [C_x, C_y, C_z], [D_x, D_y, D_z])$
`plot :: Ellipsoid`	楕円面	`Ellipsoid`$(r_x, r_y, r_z, [C_x, C_y, C_z])$
`plot :: Line3d`	線分	`Line3d`$([x_1, y_1, z_1], [x_2, y_2, z_2])$
`plot :: Parallelogram3d`	平行四辺形	`Parallelogram`$([C_x, C_y, C_z], [U_x, U_y, U_z], [V_x, V_y, V_z])$
`plot :: Point3d`	点	`Point3d`(x, y, z)
`plot :: PointList3d`	多くの点	`PointList2d`$([[x_1, y_1, z_1], [x_2, y_2, z_2], \cdots])$
`plot :: Polygon3d`	多角形	`Polygon3d`$([[x_1, y_1, z_1], [x_2, y_2, z_2], \cdots])$
`plot :: Sphere`	球	`Sphere`$(r, [C_x, C_y, C_z])$
`plot :: Text3d`	文	`Text3d`$(”\cdots”, [C_x, C_y, C_z])$

注7)

注7) 1. この他，`plot :: SurfaceSet`(三角形の集合で面を作る), `plot :: SurfaceSTL`(STL ファイルで表された図形を表示) があります．しかし, この 2 つは，他の low-level primitive と比べ, かなり複雑なので省略しました.
2. 主な *Attribute* としては，領域を囲むような図形 (球, 楕円面, 平行四辺形, 直方体, 円錐, 円柱など）に対しては，Filled(値は *TRUE*, *FALSE*),FillColor(表面の色),LineColor(境界の色) などが，点や点列に関しては, PointSize,Color などが，文 (text) に関しては，TextFont,Billboarding(Default では *TRUE*), TextOrientation などがあります.

12.2.1　球

半径が r, 中心が (C_x, C_y, C_z), の球 (sphere) は

$$\texttt{plot} :: \texttt{Sphere}(\texttt{r}, [\texttt{C}_\texttt{x}, \texttt{C}_\texttt{y}, \texttt{C}_\texttt{z}])$$

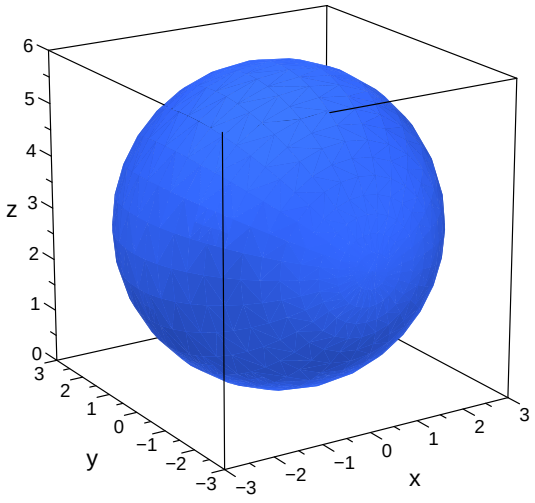
で表されます．また，primitive は, 全て

- `plot(plot :: Sphere(3,[0,0,3]))` ⋯ 中心が $(0,0,3)$ で, 半径 3 の球

のように `plot()` コマンドと一緒に使います．しかし，次のように分けても大丈夫です.

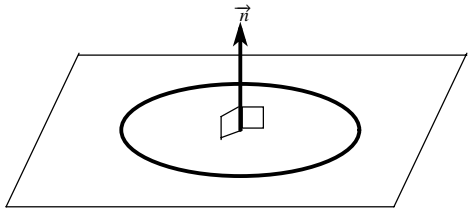
- `mysphere := plot :: Sphere(3,[0,0,3])`　　>> plot::Sphere(3, [0, 0, 3])
- `plot(mysphere)`

これで次のようになります.



12.2.2　円

空間の円は, 平面の円と違って法線ベクトル $\vec{n}$ も指定します.



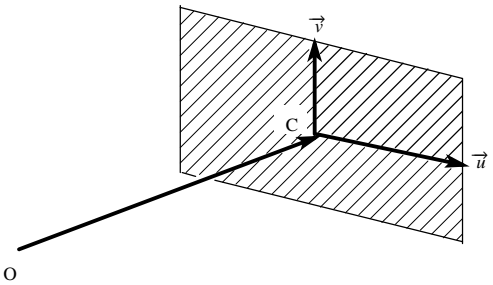
半径が r, 中心が (C_x, C_y, C_z), 法線ベクトル $\vec{n} = (n_x, n_y, n_z)$ の円は

$$\texttt{plot} :: \texttt{Circle3d}(\texttt{r}, [\texttt{C}_\texttt{x}, \texttt{C}_\texttt{y}, \texttt{C}_\texttt{z}], [\texttt{n}_\texttt{x}, \texttt{n}_\texttt{y}, \texttt{n}_\texttt{z}])$$

で表されます．[注8] Default では, 塗りつぶしはないので, 円盤を描きたいときは，Filled= *TRUE* とします．塗りつぶしの色は，FilledColor(Color でも大丈夫)，円周の色は，LineColor で指定します.

[注8] $[n_x, n_y, n_z]$ を省略すると，$\vec{n} = [0, 0, 1]$ になります.

12.2.3 平行四辺形



上図のように,

$$\overrightarrow{\mathrm{OP}} = \overrightarrow{\mathrm{OC}} + s\vec{u} + t\vec{v} \Longleftrightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} C_x \\ C_y \\ C_z \end{pmatrix} + s\begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} + t\begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} (-1 \leqq s \leqq 1, -1 \leqq t \leqq 1)$$

の平行四辺形 (parallelogram) は,

$$\texttt{plot :: Parallelogram3d([C}_\texttt{x}\texttt{, C}_\texttt{y}\texttt{, C}_\texttt{z}\texttt{], [u}_\texttt{x}\texttt{, u}_\texttt{y}\texttt{, u}_\texttt{z}\texttt{], [v}_\texttt{x}\texttt{, v}_\texttt{y}\texttt{, v}_\texttt{z}\texttt{])}$$

で表されます. Default では, 塗りつぶしになっているので, 境界だけが欲しい時は Filled= *FALSE* で指定します.

[球, 円, 平行四辺形の例]

平行四辺形, 円, 球の例として, 2つの平面：$z = 0$ と $z = 6$ に挟まれた球：$x^2 + y^2 + (z-3)^2 = 9$ と, 球の赤道となる円を一緒に描いてみます. 先ずは, それぞれの オブジェクト を作ります.

- `sphere := plot :: Sphere(3, [0, 0, 3])` : ··· 半径が 3, 中心が $(0, 0, 3)$ の球
- `plane1 := plot :: Parallelogram3d([0, 0, 0], [3, 0, 0], [0, 3, 0], Color = RGB :: Green)` :
  ··· 中心が $(0, 0, 0)$, $\vec{u} = (3, 0, 0)$, $\vec{v} = (0, 3, 0)$ の平行四辺形
- `plane2 := plot :: Parallelogram3d([0, 0, 6], [3, 0, 0], [0, 3, 0], Color = RGB :: Green.[0.7])` :
  ··· 中心が $(0, 0, 6)$, $\vec{u} = (3, 0, 0)$, $\vec{v} = (0, 3, 0)$ の平行四辺形
- `circle := plot :: Circle3d(3, [0, 0, 3], [0, 0, 1], Color = RGB :: Red, LineWidth = 1)` :
  ··· 半径が 3, 中心が $(0, 0, 3)$, 法線ベクトルが $(0, 0, 1)$ の円周

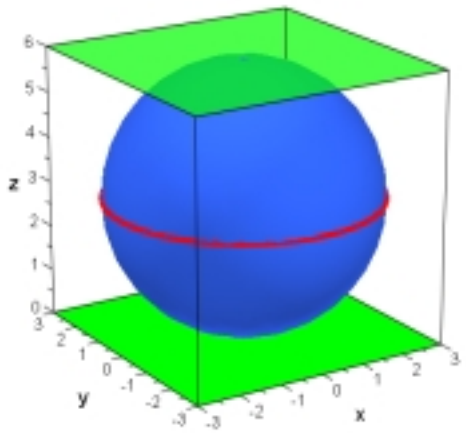
まとめて `plot` します.

- `plot(sphere, plane1, plane2, circle)`

これで次のようになります. [注9)]

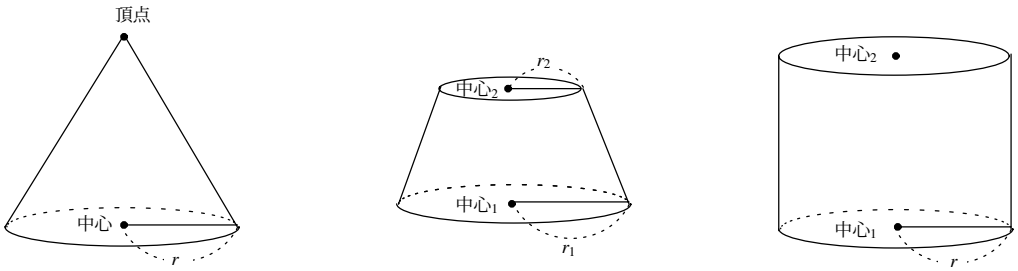
注9) `plane2` の定義で, `Color = RGB :: Green.[0.7]` の [0.7] というのは, 色の濃さ (1− 透過率) を表していて, 0 以上 1 以下の数字で表します. `RGB :: Green.[1]` というのは, 緑色で, 背景色をまったく透過しません. (`RGB :: Green` は, `RGB :: Green.[1]` の省略です.) 一方, `RGB :: Green.[0]` というのは, 緑色ですが, 背景色を完全に透過します. (したがって画面上では, 全く見えません.) `RGB :: Green.[0.5]` は, 背景色を半分透過します. また `RGB :: Green = [0, 1, 0]` ですから, `RGB :: Green.[0.5] = [0, 1, 0, 0.5]` としても同じです.
なお,「.」は, 一般に 2 つのリストを結びつけることができます. 例えば,

`[1, 2, 3].[4, 5]` >> `[1, 2, 3, 4, 5]`



[./lowlevel_primitives/sphere2.eps で, 上の面が透過色になっていれば, そちらを使用してください]

12.2.4　円錐, 円柱



底面の円の半径が r, 中心の座標が (C_x, C_y, C_z), 頂点 D の成分が (D_x, D_y, D_z) の円錐 (cone) は，次のように表されます.

$$\texttt{plot} :: \texttt{Cone}(\texttt{r}, [\texttt{C}_\texttt{x}, \texttt{C}_\texttt{y}, \texttt{C}_\texttt{z}], [\texttt{D}_\texttt{x}, \texttt{D}_\texttt{y}, \texttt{D}_\texttt{z}])$$

底面の円の半径が r_1, 中心 C の座標が (C_x, C_y, C_z), 上面の円の半径が r_2, 中心 D の座標が (D_x, D_y, D_z) の円錐台 (conial frustum) は，次のように表されます.

$$\texttt{plot} :: \texttt{Cone}(\texttt{r}_1, [\texttt{C}_\texttt{x}, \texttt{C}_\texttt{y}, \texttt{C}_\texttt{z}], \texttt{r}_2, [\texttt{D}_\texttt{x}, \texttt{D}_\texttt{y}, \texttt{D}_\texttt{z}])$$

底面の円の半径が r, 中心 C の座標が (C_x, C_y, C_z), 上面の円の中心が D，その座標が (D_x, D_y, D_z) の円柱 (cylinder) は，次のように表されます.

$$\texttt{plot} :: \texttt{Cylinder}(\texttt{r}, [\texttt{C}_\texttt{x}, \texttt{C}_\texttt{y}, \texttt{C}_\texttt{z}], [\texttt{D}_\texttt{x}, \texttt{D}_\texttt{y}, \texttt{D}_\texttt{z}])$$

円錐, 円柱, 円錐台では，Default で, 表面を塗りつぶしますから，境界だけが欲しい時は，Filled= $FALSE$ とします. ただし，円錐台では，側面しか塗りつぶしませんので，底面や上面も欲しい場合は，`plot :: Circle3d` と `Filled = TRUE` を組み合わせて，底面や上面を描きます.

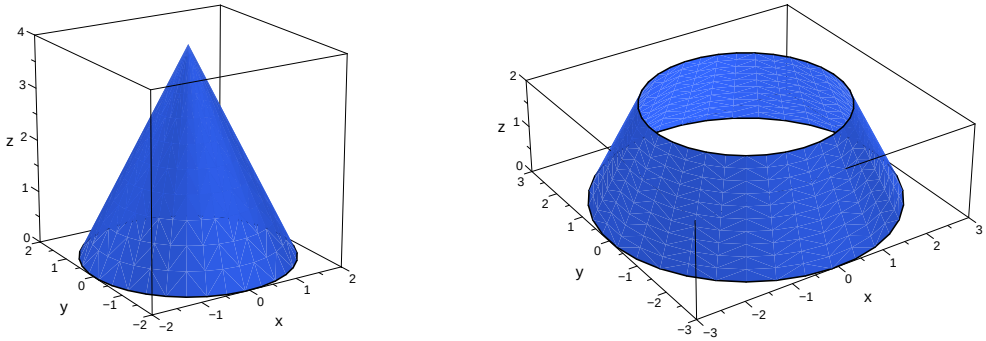
[円錐, 円錐台の例]

先ず オブジェクト を作ります.

- `cone1 := plot :: Cone(2, [0, 0, 0], [0, 0, 4]) :`
  ⋯ 底面の円の半径は 2, 中心は $(0, 0, 0)$, 頂点は $(0, 0, 4)$ の円錐
- `frustum := plot :: Cone(3, [0, 0, 0], 2, [0, 0, 2]) :`
  ⋯ 底面の円の半径は 3, 中心は $(0, 0, 0)$, 上面の円の半径は 2, 中心は $(0, 0, 2)$ の円錐台

これらを `plot` します.

- `plot(cone1); plot(frustum)`

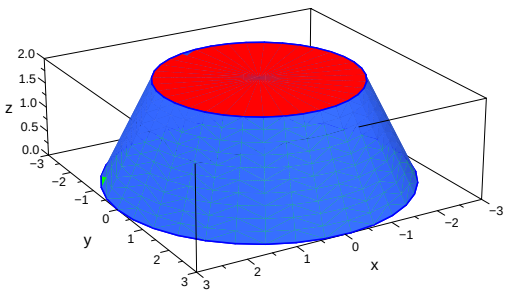


ご覧のように, 円錐台は側面だけ描かれます. `plot :: Circle3d` を使って, 底面と, 上面を作ります.

- `dish1 := plot :: Circle3d(3, [0, 0, 0], Filled = TRUE, FillColor = RGB :: Green) :`
  ⋯ 半径が 3, 中心が $(0, 0, 0)$ の円盤 (緑色)
- `dish2 := plot :: Circle3d(2, [0, 0, 2], Filled = TRUE, FillColor = RGB :: Red) :`
  ⋯ 半径が 2, 中心が $(0, 0, 2)$ の円盤 (赤色)

そして, 先の円錐台に付け加えます.

- `plot(frustum, dish1, dish2)`



[円柱の例]

半径が 1，軸が x 軸に平行で, 長さが 4 の円柱 (薄緑) を作ってみます.

- `cy1 := plot :: Cylinder(1, [-2, 0, 0], [2, 0, 0], Color = RGB :: LightGreen) :`
  ⋯ 底面の半径が 1，底面の中心が $(-2, 0, 0)$, 上面の中心が $(2, 0, 0)$ の円柱

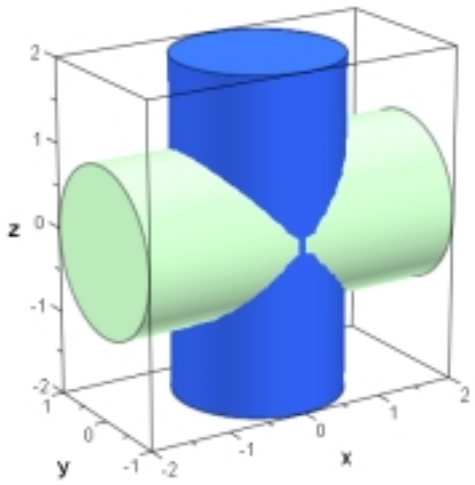
次は，半径が 1，軸が z 軸に平行で, 長さが 4 の円柱 (水色) を作ってみます.

- `cy2 := plot :: Cylinder(1, [0, 0, -2], [0, 0, 2]) :`
  ⋯ 底面の半径が 1，底面の中心が $(0, 0, -2)$, 上面の中心が $(0, 0, 2)$ の円柱

2 つを一緒に描いてみます.

- `plot(cy1, cy2)`

次のようになります.



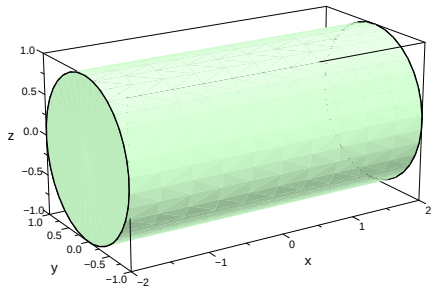
[./lowlevel_primitives/cylinder1.eps の方がきれいなら, そちらを使ってください．]

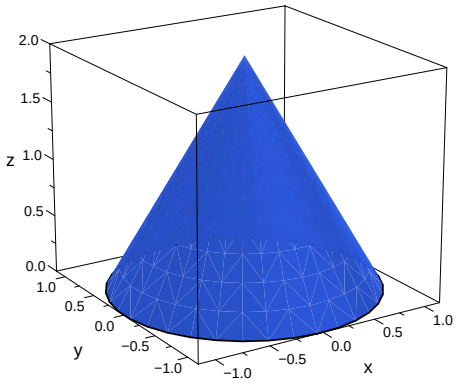
今度は, 円錐と, 円柱を一緒に描いて見ます．まず，底面の半径が $\frac{2}{\sqrt{3}}$, 中心が $(0, 0, 0)$, 頂点が $(0, 0, 2)$ の円錐を作ります.

- `cone2 := plot :: Cone(2/sqrt(3), [0, 0, 0], [0, 0, 2]) :`
  ⋯ 底面の半径が $\frac{2}{\sqrt{3}}$，底面の中心が $(0, 0, 0)$, 頂点が $(0, 0, 2)$ の円柱

まず，この円錐と, 先の円柱 (cy1) を別々に描いてみます.

- `plot(cy1); plot(cone2)`

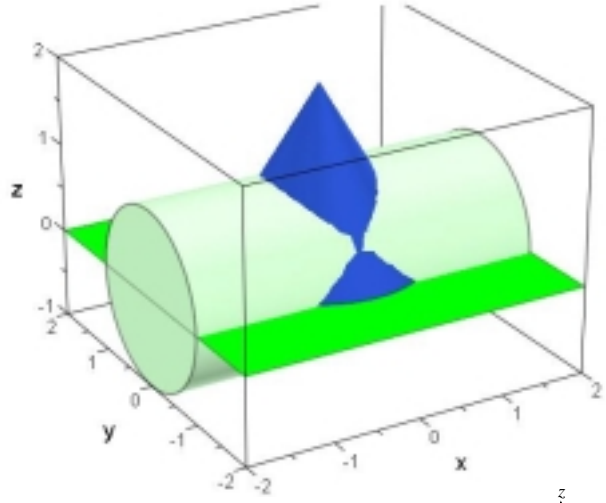




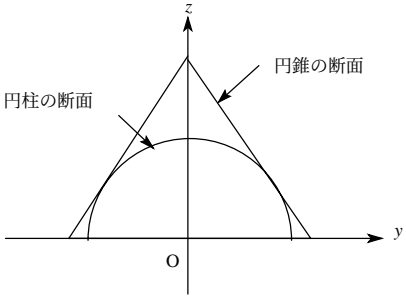
次は, この 2 つを一緒に描いてみます．見やすくするために，xy 平面上の平行四辺形を追加します．

- `plane := plot::Parallelogram3d([0,0,0],[2,0,0],[0,2,0],Color = RGB::Green)` :
  $\cdots$ 中心が $(0,0,0)$,$\overrightarrow{u} = (2,0,0)$,$\overrightarrow{v} = (0,2,0)$ の平行四辺形
- `plot(cy1,cone2,plane)`

[./lowlevel_primitives/cone_cylinder2.eps の方がきれいなら, そちらを使ってください．]



実は yz 平面の断面図を考えると 2 つの図形は接しています．いまいち, 接しているのが見えづらいですが,, MuPAD の実際の画面ではもっときれいに見えます．また，cone や cylinder には，Mesh という *Attribute* は無いので, これ以上は仕方ありません．



平面 $x = 0$ 上の断面図

注10)

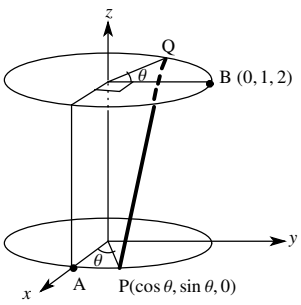
注10) 1. low-level primitive では, 小さい三角形を集めて, 図形を描きます.
2. ちなみに, 円錐と円柱の交線のうち, 閉曲線を C とすると，円柱の側面のうち C で囲まれた面積 S_1 や，円錐の側面のうち C で

12.2.5　アニメーション

low-level primitive もアニメーションできます.

例題 1

2 点 A(1, 0, 0), B(0, 1, 2) を結ぶ線分 AB を, z 軸の周りに回転して得られる曲面はどのような曲面か?



点 A,B を z 軸の周りに θ 回転した点を, それぞれ P,Q とすると,

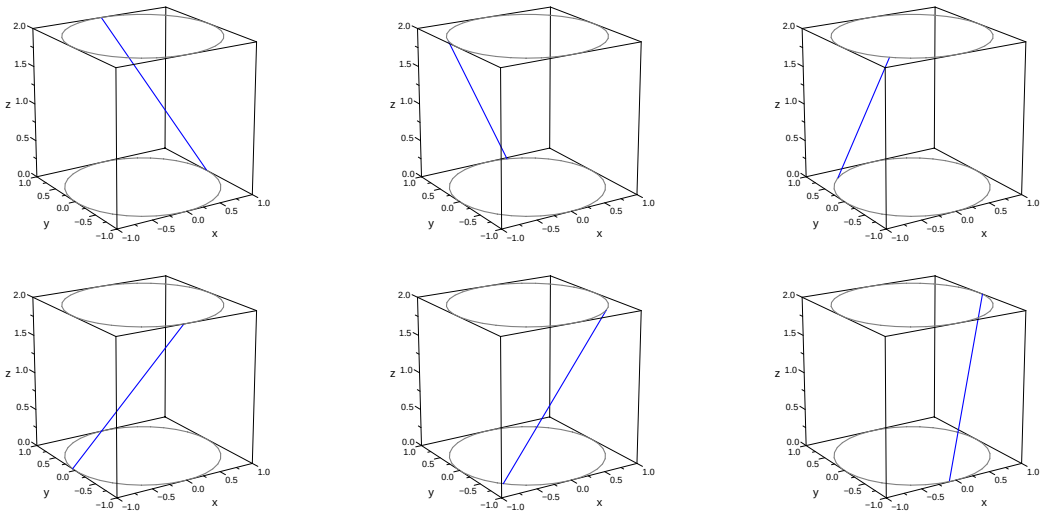
$$\mathrm{P}(\cos\theta, \sin\theta, 0),\ Q\left(\cos\left(\frac{\pi}{2}+\theta\right), \sin\left(\frac{\pi}{2}+\theta\right), 2\right)$$

よって次のようにすれば，線分 PQ の回転をアニメーションにできます.

```
segment := plot::Line3d([cos(t), sin(t), 0], [cos(t + PI/2), sin(t + PI/2), 2], t = 0..2 * PI):
```

これに 2 つの円をつけて plot します.

```
upperCircle := plot::Circle3d(1, [0, 0, 2], Color = RGB::Gray50):
lowerCircle := plot::Circle3d(1, [0, 0, 0], Color = RGB::Gray50):
plot(segment, upperCircle, lowerCircle)
```



囲まれた面積 S_2，さらに，円錐を中が詰まっていると考えて，そのうち円柱の側面の上側にある部分の体積 V なども求まります.

$$S_1 = 4 - \frac{4\sqrt{3}}{9}\pi,\ S_2 = \frac{8\sqrt{3}\pi}{9} - 2,\ V = \frac{4\sqrt{3}\pi}{9} - 2$$

接している場合以外は, 側面積や体積は, なかなか求まりません．ちなみに,「S_1 を求めよ.」というのは，私がある雑誌に出題した問題なのですが，そのときの読者のコメントに「概形が分らないので，ナスをくり抜いて実験した！」というのもありました. MUPAD を使うと，ナスを無駄にしなくて済みます.

しかし，これでは，どんな曲面になるか分かりづらいので，今度は，パラメータを変えた曲面を一度に描いてみます．

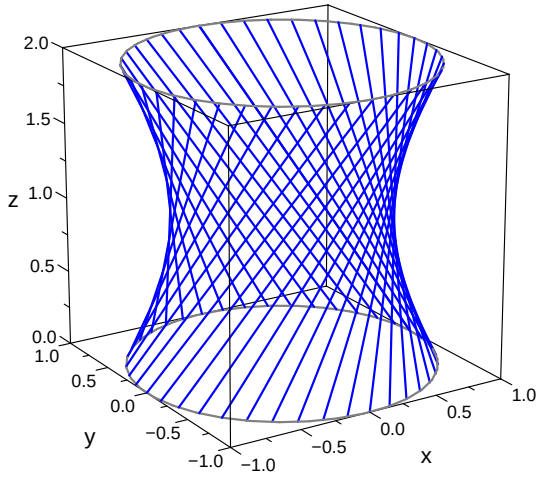
```
plot :: Line3d([cos(0), sin(0), 0], [cos(PI/2), sin(PI/2), 2]),
plot :: Line3d([cos(PI/20), sin(PI/20), 0], [cos(PI/20 + PI/2), sin(PI/20 + PI/2), 2])···
plot :: Line3d([cos(2 * PI), sin(2 * PI), 0], [cos(2 * PI + PI/2), sin(2 * PI + PI/2), 2])
```

これを，列生成子「$」を使って，一度に 定義します．

```
•segments := plot :: Line3d([cos(PI/20 * i), sin(PI/20 * i), 0],
 [cos(PI/20 * i + PI/2), sin(PI/20 * i + PI/2), 2]) $i = 0..40 :
```
···①

まとめて，plot します．

```
•plot(segments,upperCircle,lowerCircle)
```



これはこれで良いですが，さらに，描く線分の数が，次第に増えていくような，アニメーションにします．

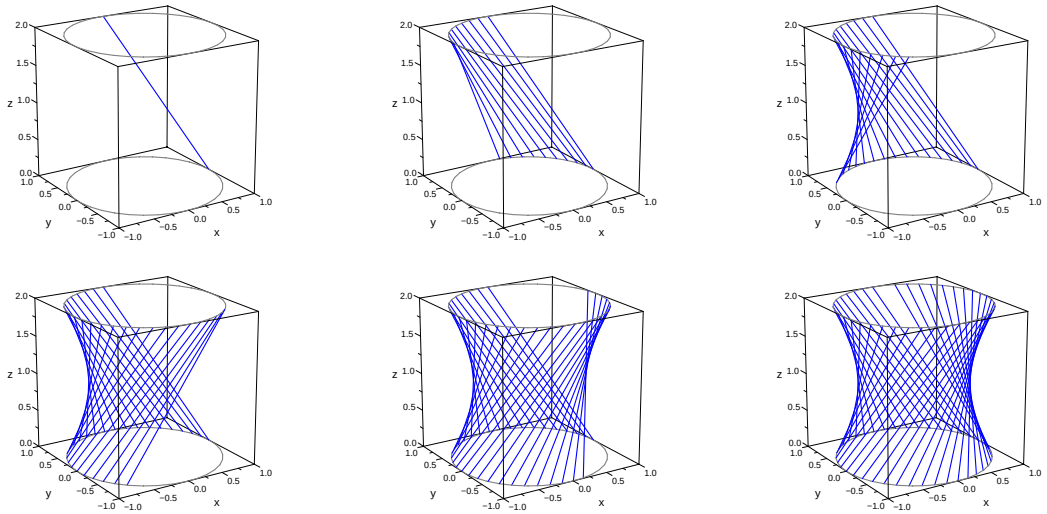
```
•Segments := plot :: Line3d([cos(PI/20 * i), sin(PI/20 * i), 0],
 [cos(PI/20 * i + PI/2), sin(PI/20 * i + PI/2), 2], TimeRange = 0.25 * i..10 $i = 0..40 :
```
···②

TimeRange = 0.25 * i..10 だけを，① へ追加しました．これによって

```
 plot :: Line3d([cos(0), sin(0), 0], [cos(PI/2), sin(PI/2), 2], TimeRange = 0..10),
 plot :: Line3d([cos(PI/20), sin(PI/20), 0], [cos(PI/20 + PI/2), sin(PI/20 + PI/2), 2], TimeRange = 0.25..10)···
 plot :: Line3d([cos(2 * PI), sin(2 * PI), 0], [cos(2 * PI + PI/2), sin(2 * PI + PI/2), 2], TimeRange = 10..10)
```

が生成されたことになります．これを，VisibleBeforeBegin = *FALSE* のオプションをつけて，plot すれば，その TimeRange に入るまでは見えないが，入った後は見え続けるので，線分の数が次第に増えて行きます．

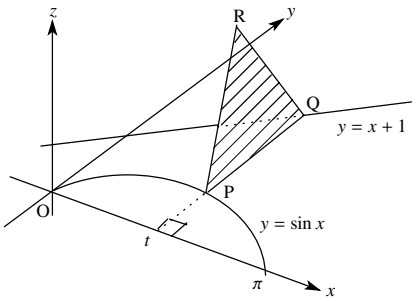
```
•plot(Segments,upperCircle,lowerCircle,VisibleBeforeBegin = FALSE)
```



これで，全部で 10 秒間のアニメーションになります．このように通過領域はアニメーションと相性が良いです．

例題 2

点 P は，曲線 $y = \sin x$ $(0 \leqq x \leqq \pi)$ 上を動くとする．点 P を通り y 軸に平行な直線と $y = x + 1$ の交点を Q とする．線分 PQ を含み xy 平面に垂直な平面上に点 R をとり，正三角形 PQR を作る．点 P が，原点から点 $(\pi, 0)$ まで動くとき，△PQR が通過してできる立体を図示せよ．ただし，R の z 成分は 負にならないものとする．



P の x 成分を t とすると，

$$\mathrm{P}(t, \sin t, 0),\ \mathrm{Q}(t, t+1, 0)$$

R から xy 平面に下ろした垂線の足を H とすると，H は，線分 PQ の中点．HR= $\frac{\sqrt{3}}{2}$PQ だから

$$R\left(t, \frac{\sin t + t + 1}{2}, \frac{\sqrt{3}}{2}(t + 1 - \sin t)\right)$$

三角形は，plot :: Polygon3d([[x_1, y_1, z_1], [x_2, y_2, z_2], [x_3, y_3, z_3]]) [注11] を使って作れます．ただそのままでは，開いた三角形になってしまうので，Closed = $TRUE$ とします．よって，△PQR は，

- plot :: Polygon3d([[t, sin(t), 0], [t, t + 1, 0], [t, (sin(t) + t + 1)/2, sqrt(3)/2 ∗ (t + 1 − sin(t))]], Closed = TRUE) :

[注11] ポリゴンは多角形の意味. 4 角形，5 角形 ··· も描ける．外側のリストを忘れないこと．

となります．よって，次のようにすると,「例題１」の最後の例のようなアニメーションになります.

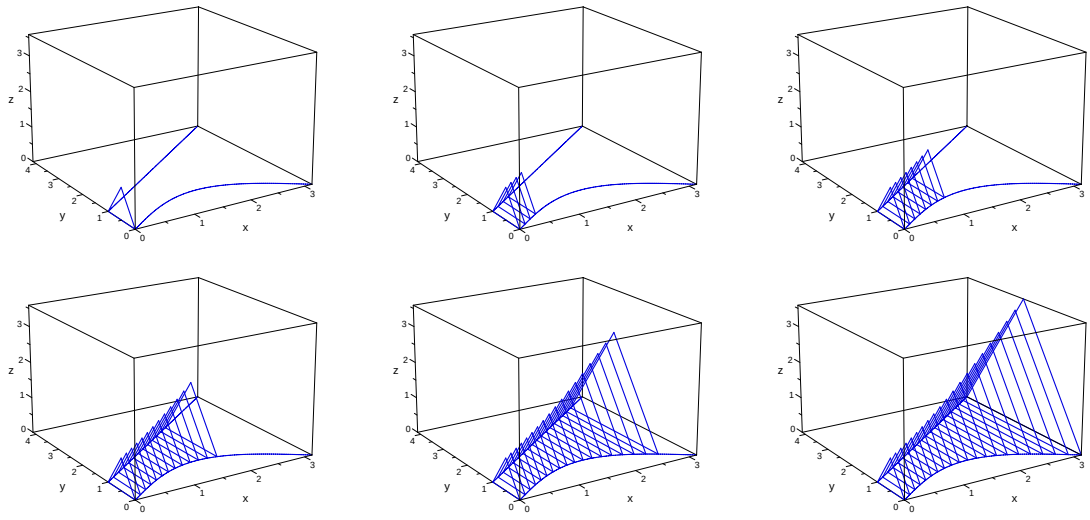
```
•body := plot::Polygon3d([[i/20*PI, i/20*PI+1, 0], [i/20*PI, sin(i/20*PI), 0],
 [i/20*PI, (i/20*PI+1+sin(i/20*PI))/2, sqrt(3)/2*(i/20*PI+1-sin(i/20*PI))]],
 Closed=TRUE, TimeRange=i/2..(i+1)/2) $i=0..20:
```

これに，*xy* 平面上の曲線を加えます．空間の図形と併用する場合は，plot::Curve2d ではなく，後述する plot::Curve3d を使います.

```
•curve1 := plot::Curve3d([t, sin(t), 0], t=0..PI):
•curve2 := plot::Curve3d([t, t+1, 0], t=0..PI):
```

VisibleBeforeBegin=FALSE のオプションを付けて，plot します.

```
•plot(body, curve1, curve2, VisibleBeforeBegin=FALSE)
```



12.2.6 正多面体 (polyhedron)

正多面体は，正 4 面体, 正 6 面体, 正 8 面体, 正 12 面体, 正 20 面体 の 5 種類だけが存在することが知られています．この正多面体を簡単に書くことができます.

plot::Tetrahedron()	正 4 面体
plot::Hexahedron()	正 6 面体
plot::Octahedron()	正 8 面体
plot::Dodecahedron()	正 12 面体
plot::Icosahedron()	正 20 面体

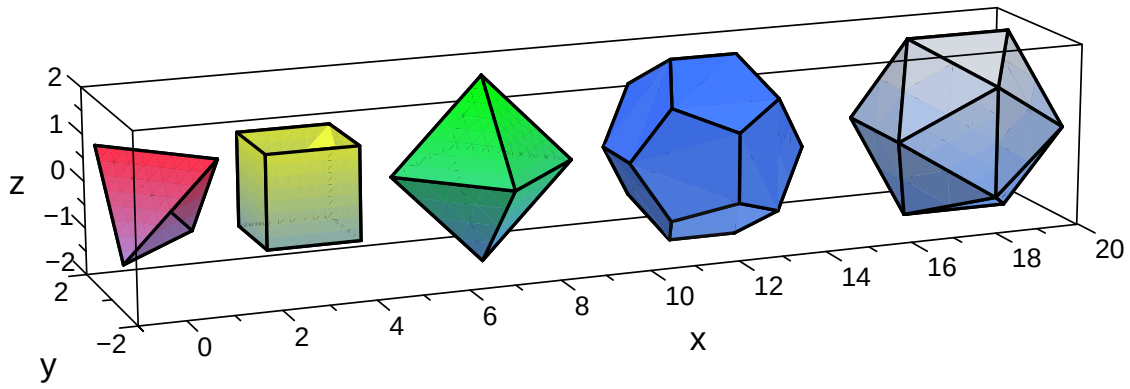
Attribute の，Center=$[C_x, C_y, C_z]$ で, 中心の座標を指定できます．Radius=r で，外接球の半径を r にします.（ただし，正 6 面体の場合は，内接円の半径を r にします.）default では，Center=$[0,0,0]$, Radius=1 となっています．球などと違って，中心の座標はオプションなので，plot::Tetrahedron([1,2,3]) な

どとは，できません．plot :: Tetrahedron(Center = [1, 2, 3]) とします．

- hedron4 := plot :: Tetrahedron() : ··· 中心 (0, 0, 0),Radius=1 の正 4 面体
- hedron6 := plot :: Hexahedron(Center = [3, 0, 0]) : ··· 中心 (3, 0, 0),Radius=1 の正 6 面体
- hedron8 := plot :: Octahedron(Center = [7, 0, 0], Radius = 2) : ··· 中心 (7, 0, 0),Radius=2 の正 8 面体
- hedron12 := plot :: Dodecahedron(Center = [12, 0, 0], Radius = 2) : ··· 中心 (12, 0, 0),Radius=2 の正 12 面体
- hedron20 := plot :: Icosahedron(Center = [18, 0, 0], Radius = 2) : ··· 中心 (18, 0, 0),Radius=2 の正 20 面体

まとめて一緒に描きます．

- plot(hedron4, hedron6, hedron8, hedron12, hedron20)



向きを変更したい時は plot :: Rotate3d, plot :: Transform3d (後述) を利用します．

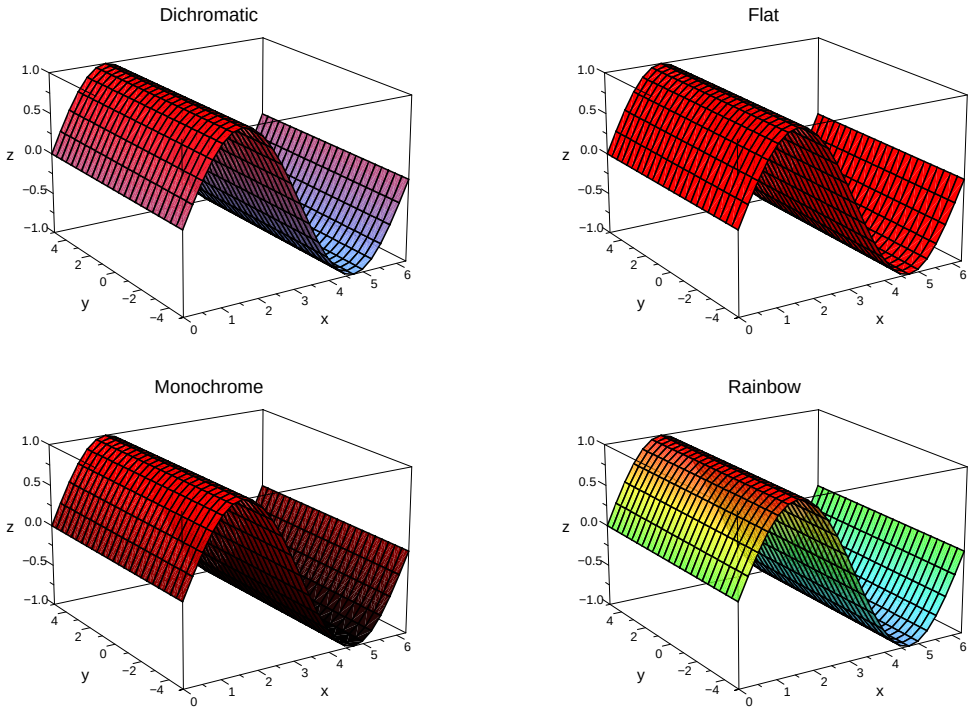
12.3 曲面の色

plotfunc3d と low – level primitives とでは，色の塗り方が全く異なることに，気付かれたと思います．これは，FillColorType の default が，異なるためです．ここで，FillColorType というのは, 面の色の塗り方を指定する *Attribute* であり，その値は, 以下の 5 種類です.

- Dichromatic– 2 色塗り (plotfunc3d, high-level primitives の default)
- Flat –ベタ塗り (low-level primitives の default)
- Monochrome –モノクローム (明度だけ変化)
- Rainbow – 2 色塗り（虹色の効果あり）
- Functional –FillColorFunction(自分で定義した関数) に従って描写

まず，Dichromatic,Flat,Monochrome,Rainbow を見てみます.

- plotfunc3d(sin(x), x = 0..2 ∗ PI, FillColorType = Dichromatic, Header = ”Dichromatic”)
- plotfunc3d(sin(x), x = 0..2 ∗ PI, FillColorType = Flat, Header = ”Flat”)
- plotfunc3d(sin(x), x = 0..2 ∗ PI, FillColorType = Monochrome, Header = ”Monochrome”)
- plotfunc3d(sin(x), x = 0..2 ∗ PI, FillColorType = Rainbow, Header = ”Rainbow”)



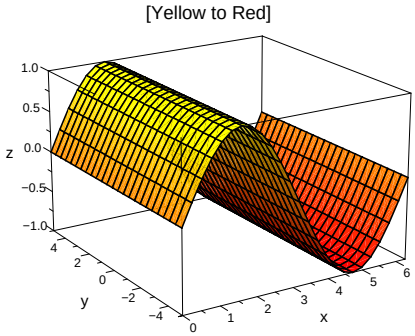
Dichromatic と Rainbow は，FillColor と FillColor2 という 2 種類の色を使って描きます. (plotfunc3d の場合，default では，FillColor = RGB :: Red , FillColor2 = RGB :: Blue です．) Dichromatic の場合，z 成分にしたがって，上からだんだんと，FillColor から FillColor2 に変化します．Rainbow も原則として同じですが，虹色の効果が出るように, 途中に多くの中間色を挟んでいます.

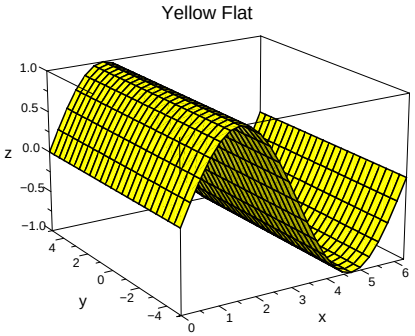
FillColorType = *Flat* の場合は，FillColor だけで塗りつぶします．FillColor2 の値は関係ありません.

FillColorType = *Monochrome* の場合も，FillColor2 の値は関係ありません．FillColor の色を，上方から, だんだんと，暗度を増しながら塗ります.

FillColor, FillColor2 も 変更できます．また，plotfunc3d では，Color でも FillColor を指定できます．(object inspector を使って，Style → Surface からもできます．)

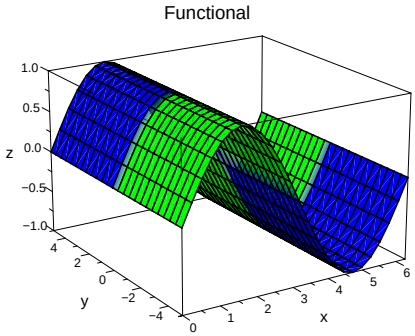
- plotfunc3d(sin(x), x = 0..2 ∗ PI, Color = RGB :: Yellow, FillColor2 = RGB :: Red, Header = "[Yellow to Red]"
- plotfunc3d(sin(x), x = 0..2 ∗ PI, FillColor = RGB :: Yellow, FillColorType = Flat, Header = "Yellow Flat")

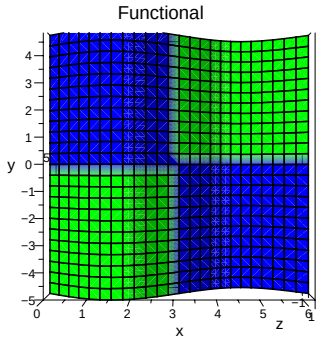




最後に FillColorType = *Functional* の場合は，FillColor や FillColor2 の値は関係ありません．ユーザーの定義した関数（FillColorFunction で指定する関数）に従って，塗っていきます．この関数は，plotfunc3d の場合は，(x, y) の関数で，RGB 色 (または，[r,g,b], [r,g,b,p] のリスト) を返す関数です.

- mycolor := (x, y) −> piecewise([(x − 3) ∗ y > 0, RGB :: Green], [(x − 3) ∗ y <= 0, RGB :: Blue]) :
- plotfunc3d(sin(x), x = 0..2 ∗ PI, FillColorType = Functional, FillColorFunction = mycolor, Header = "Functional")





ご覧のように，$(x-3)y > 0$ の部分は，緑に，それ以外は，青色に塗られました．なお，LineColorFunction というのもあり，それを使って，細かく曲線の色を指定することも出来ます.

12.4 様々な空間のグラフ (high-level primitives)

幾何図形のような”low-level primitive” と異なり，plotfunc3d のように, 関数式を指定していろいろなグラフを描かせることもできます．こちらは，”high-level primitive” と呼びます．この節では，high-level primitive を取り上げます．以下は，空間の (比較的基本的と思われる) high-level primitive です.

plot :: Function3d	$z = f(x, y)$ のグラフ (陽関数表示) Function3d(f(x, y), x = x_1..x_2, y = y_1..y_2)
plot :: Implicit3d	$f(x, y, z) = 0$ のグラフ (陰関数表示) Implicit3d(f(x, y, z), x = x_1..x_2, y = y_1..y_2, z = z_1..z_2)
plot :: Curve3d	曲線のパラメータ表示 Curve3d([x(t), y(t), z(t)], t = t_1..t_2)
plot :: Surface	曲面のパラメータ表示 Surface([x(u, v), y(u, v), z(u, v)], u = u_1..u_2, v = v_1..v_2)

注12)

12.4.1 $y = f(x, y)$ のグラフ (plot::Function3d)

Function3d は，$y = f(x, y)$ のグラフを描く時に使います．例えば，$y = x^2 (-2 \leqq x \leqq 2, -2 \leqq y \leqq 2)$ のグラフは，次のようにします.

- f := plot :: Function3d(x^2, x = −2..2, y = −2..2)
- plot(f)

これは，plotfunc3d を用いて, 次のようにしても同じです．(property inspector の表示も同じです.)

- plotfunc3d(x^2, x = −2..2, y = −2..2)

すなわち，plotfunc3d(f, x = x_1..x_2, y = y_1..y_2) は，plot(plot :: Function3d(f, x = x_1..x_2, y = y_1..y_2) の簡易表現です．Function3d は, 他の primitive（球など) と組み合わせる場合に用います.

注12) この他,

統計用グラフ plot::Bars3d,plot::Histogram3d, plot::Piechart3d

微分方程式の解の表示 plot::Ode3d

ベクトル場 plot::VectorField3d

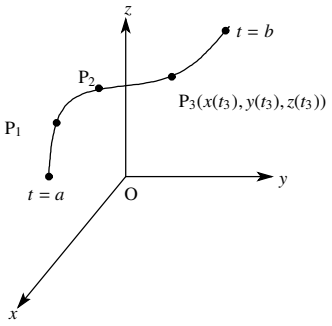
行列の空間表示 plot::Matrixplot

***x*, *y* 軸周りの回転体の表示** plot::XRotate,plot::ZRotate

その他 plot::HOrbital(原子軌道表示)

があります．plot :: XRotate, plot :: ZRotate に関しては, 後で取り上げます.

12.4.2　空間の曲線



[空間の曲線とは何か?]

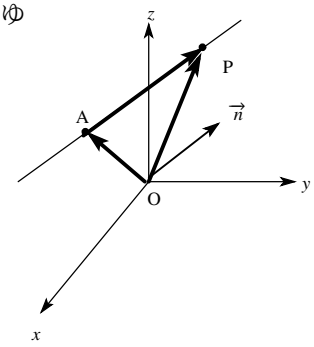
$x = x(t), y = y(t), z = z(t)$ $(a \leqq t \leqq b)$ は媒介変数表示された 3 次元の曲線 を表します．なぜなら直線上の点の空間座標は 1 つのパラメーター t の値で決まってしまうからです．また, この式の意味は t が変化するにつれ点 P $(x(t), y(t), z(t))$ も動き、それは曲線を作るという意味です.

[空間の曲線のプロット]

空間の曲線: $x = x(t), y = y(t), z = z(t)$ $(a \leqq t \leqq b)$ は,

$$\texttt{plot :: Curve3d([x(t), y(t), z(t)], t = a..b)}$$

です．plot :: Curve2d とほとんど同じです．最も基本的な曲線は，直線です.



点 A を通りベクトル $\vec{n}$ と平行な直線上の点を点 P とすると $\overrightarrow{AP} // \vec{n}$. ゆえに，$\overrightarrow{AP} = t\vec{n}$ (t は実数) とおけて,

$$\overrightarrow{OP} = \overrightarrow{OA} + \overrightarrow{AP} = \overrightarrow{OA} + t\vec{n}$$

です．よって P(x, y, z), A(x_0, y_0, z_0), $\vec{n} = (a, b, c)$ とすると

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + t \begin{pmatrix} a \\ b \\ c \end{pmatrix} \Longleftrightarrow \begin{cases} x = x_0 + at \\ y = y_0 + bt \\ z = z_0 + ct \end{cases}$$

たしかに x, y, z 座標がパラメータ t の関数になっています.

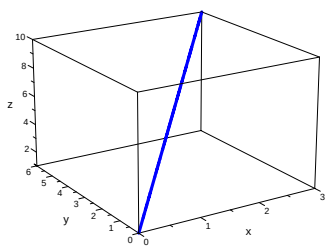
点 A$(0, 0, 1)$ を通り,$\vec{n} = (1, 2, 3)$ と平行な直線を描いてみます.

$$(x, y, z) = (0, 0, 1) + t(1, 2, 3) = (t, 2t, 3t + 1)$$

よって，次のようにします.

- line := plot :: Curve3d([t, t ∗ 2, 3 ∗ t + 1], t = 0..3, LineWidth = 1) :
- plot(line)

線の太さを 1∗unit::mm にしました．これで次のようになります．[注13)]



[注13)] unit::mm は,unit lib の object です．他に unit::cm,unit::km,unit::kg などもあります．実際に印刷される太さとは異なります.

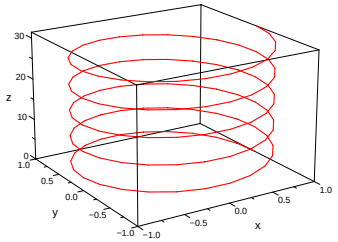
さて次の曲線は何を表しているでしょう？

$$x = \cos t, y = \sin t, z = t \ (0 \leqq t \leqq 10\pi)$$

$z = 0$ ならもちろん円周 $(x = \cos t, y = \sin t, z = 0)$ 上を 5 周します．でも t が増えるにつれて z 成分も増えるのだから…そうです螺旋になるはずです．やってみます．

- `spiral := plot :: Curve3d([cost, sint, t], t = 0..10 * PI) :`
- `plot(spiral)`

次のようになりました．



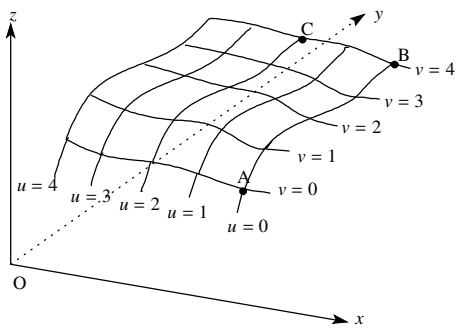
さらに, 前節の **12.2.5** 例題 2 で，$y = \sin(x)$ $(0 \leqq x \leqq \pi)$ を，`plot :: Curve3d([t, sin(t), 0], t = 0..PI)` としました．xy 平面上の曲線: $x = f(t), y = g(t)$ $(a \leqq t \leqq b)$ は，空間の図形と一緒に使う場合は，`plot :: Curve3d` を使って，`plot :: Curve3d([f(t), g(t), 0], t = a..b)` としないといけません．

12.4.3 空間の曲面 (plot::Surface)

[曲面とは何か？]

$x = f(u, v), y = g(u, v), z = h(u, v)$ $(a \leqq u \leqq b, c \leqq v \leqq d)$ は媒介変数表示された 3 次元の曲面を表します．なぜなら曲面上の点の位置 (曲面座標) は 2 つの数字の組 (u, v) で表されるからです．

例えば’ 仮に’ 図の点 A の空間座標が A$(2, 3, 4)$ とすると $[f(0,0), g(0,0), h(0,0)] = [2, 3, 4]$ となります．同様、図の点 B, 点 C の空間座標を B(x_2, y_2, z_2), C(x_3, y_3, z_3), とすると $[f(0,4), g(0,4), h(0,4)] = [x_2, y_2, z_2]$, $[f(2,4), g(2,4), h(2,4)] = [x_3, y_3, z_3]$ となります．
または，”u を一定にして v のみを動かすと P$(f(u,v), g(u,v), h(u,v))$ は曲線を描き，次に v を動かすと曲線が動き曲面を作る” と考えても良いと思います．



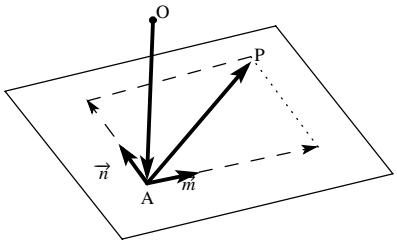
[平面のプロット]

plot :: Surface を使って曲面を描くのは次のようにします.

- surface := plot :: Surface([x(u, v), y(u, v), z(u, v)], u = a..b, v = c ..d) :
- plot(surface)

最も基本的な曲面は，平面です. 平面と平行で 1 次独立[注14] な 2 つのベクトルを $\vec{n}$, $\vec{m}$. 平面上の 1 点を点 A(x_0, y_0, z_0), 平面上の点を点 P(x, y, z) , パラメーターを u, v (u,v は実数) とすると，

$$\overrightarrow{\mathrm{AP}} = u\vec{n} + v\vec{m} \iff \overrightarrow{\mathrm{OP}} = \overrightarrow{\mathrm{OA}} + \overrightarrow{\mathrm{AP}} = \overrightarrow{\mathrm{OA}} + u\vec{n} + v\vec{m}$$



さらに，$\vec{n} = (n_x, n_y, n_z), \vec{m} = (m_x, m_y, m_z)$ とすると

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + u \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} + v \begin{pmatrix} m_x \\ m_y \\ m_z \end{pmatrix} \Longleftrightarrow \begin{cases} x &= x_0 + u\,n_x + v\,m_x \\ y &= y_0 + u\,n_y + v\,m_y \\ z &= z_0 + u\,n_z + v\,m_z \end{cases}$$

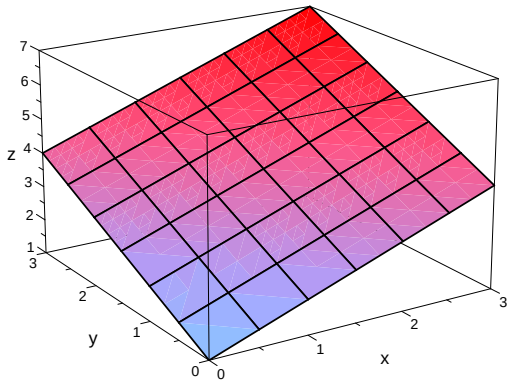
たしかに, x, y, z 座標がパラメータ u, v の関数になっています．(もちろんパラメータは, どんな文字を使っても O.K. です．)

点 A$(0, 0, 1)$ を通り、$\vec{n} = (1, 0, 1)$, $\vec{m} = (0, 1, 1)$ と平行な平面を描いてみます．このとき

$$(x, y, z) = (0, 0, 1) + u(1, 0, 1) + v(0, 1, 1) = (u, v, u + v + 1)$$

したがって，MuPAD では，次のようにすれば良いです.

- mysurface := plot :: Surface([u, v, u + v + 1], u = 0..3, v = 0..3, UMesh = 7, VMesh = 7)
- plot(mysurface)



UMesh,VMesh は，plot :: Surface で， 網の目（mesh）を調整するときの *Attribute* です. (Mesh= [7, 7] のようにしても同じです．) UMesh=7,VMesh=7 で mesh が, 各々7 本ずつ描かれます．Default では，UMesh=25,VMesh=25 になっています．(UMesh は, plot :: Curve3d でも使えます．しかし，もちろん，mesh は表示されません．)

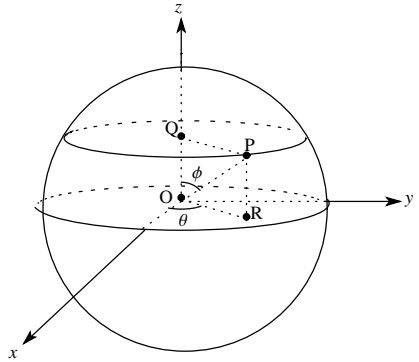
注14) $\vec{m}$, と$\vec{n}$ が 1 次独立というのは互いに平行でなく，またどちらも $\vec{0}$ でないという事です．1 次独立でないと $\vec{m}$, と$\vec{n}$ が平行四辺形を作れません.

[球のプロット]

注15) 球 $(x^2+y^2+z^2=r^2)$ をパラメータ表示します．そのため，図の $\theta\,(0 \leqq \theta < 2\pi)$, と $\phi\,(0 \leqq \phi \leqq \pi)$ を利用します.

球面上の点 P(x,y,z) から z 軸におろした垂線の足を Q, xy 平面におろした垂線の足を R とすると,

$$\begin{cases} \mathrm{OR} = \mathrm{PQ} = r\sin\phi \\ \mathrm{OQ} = r\cos\phi \end{cases}$$



よって,

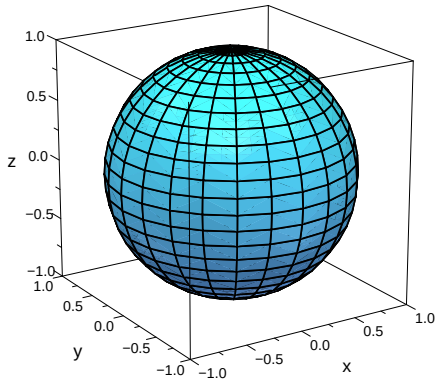
$$\begin{cases} x = \mathrm{OR}\cos\theta = r\sin\phi\cos\theta \\ y = \mathrm{OR}\sin\theta = r\sin\phi\sin\theta \\ z = \mathrm{OQ} \qquad\quad = r\cos\phi \end{cases} \qquad \cdots(*)$$

したがって原点中心，半径 1 の球のパラメータ表示は次のようになります.

$$\begin{cases} x = \sin\phi\cos\theta \\ y = \sin\phi\sin\theta \quad (0 \leqq \phi \leqq \pi,\ 0 \leqq \theta < 2\pi) \\ z = \cos\phi \end{cases}$$

したがって，MuPAD では，次のように入力します.

- `mysphere := plot::Surface([sin(u)*cos(v), sin(u)*sin(v), cos(u)], u = 0..PI, v = 0..2*PI, FillColor = RGB::Aqua):`
- `plot(mysphere, Scaling = Constrained)`



Surface3d() を使ったとき，曲面に描かれている曲線は u = 一定, v = 一定 となる曲線を表しています．$(*)$ において θ = 一定 ということは，球の北極から南極を結ぶ線 (子午線) を表しています．また ϕ = 一定 ということは，北緯 45 度線のような緯線にあたります．つまり $(*)$ の表示は地球上の点を東経 135 度, 北緯 45 度などと表すのと同じようなものです．注16)

注15) 球は，`plot::Spherical` を利用しても描くことができます．例えば, 原点中心で半径 1 の球は, 次のようになります.

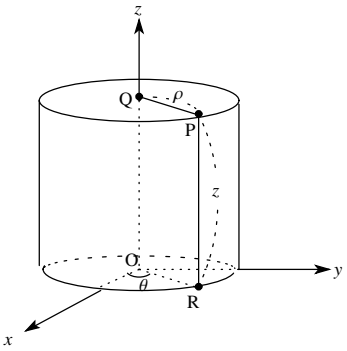
`plot::Spherical([1, u, v], u = 0..2*PI, v = 0..PI)`

注16) Default では，UMesh=VMesh=25 ですから，$u = 0(0 \leqq v \leqq 2\pi)$ と $u = \pi(0 \leqq v \leqq 2\pi)$ はそれぞれ, 北極と南極になります．さらに，$v = 0(0 \leqq u \leqq \pi)$ と $v = 2\pi(0 \leqq u \leqq \pi)$ は同じ線になることも考慮すると，子午線は 24 本，緯線は 23 本になります.

[円柱のプロット]

z 軸を中心軸, 底面の半径が ρ の円柱の側面上の点を P, 点 P(x, y, z) と z 軸との距離を ρ, P から z 軸におろした垂線の足を Q, xy 平面におろした垂線の足を R, OR と x 軸正方向との角を θ とすると図より

$$\begin{cases} x = \rho\cos\theta \\ y = \rho\sin\theta \\ z = z \end{cases} \quad \cdots(***)$$



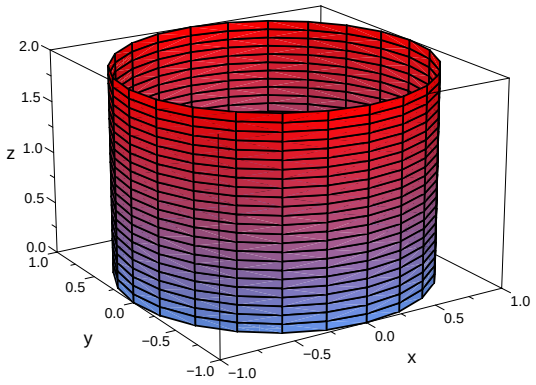
注 17)

したがって，例えば, 中心軸が z 軸で，半径が 1，高さが 2 の円柱の側面上の点は, 次のようにパラメータ表示されます.

$$(x, y, z) = (\cos\theta, \sin\theta, z)\ (0 \leqq \theta < 2\pi, 0 \leqq z \leqq 2)$$

よって，MuPAD では，次のようにします.

- `cylinderZ := plot :: Surface([cos(u), sin(u), z], u = 0..2 * PI, z = 0..2) :`
- `plot(cylinderZ)`

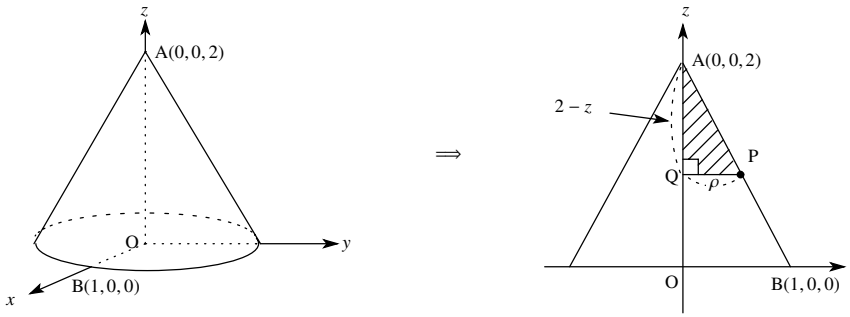


注 17) 円柱は，`plot :: Cylindrical` を使っても描けます．例えば，軸が z 軸で, 半径が 1，高さが 2 の円柱は,

`plot :: Cylindrical([1, phi, z], phi = 0..2 * PI, z = 0..2)`

[円錐のプロット]

注18) 頂点が A(0, 0, 2), 底面が $x^2 + y^2 \leqq 1, z = 0$ である円錐の側面を表して見ます.



円錐の側面上の点を P, z 軸上で 点 P と z 成分が等しい点を Q, さらに底面上の一点として B(1, 0, 0) をとります. 点 P が, 線分 AB 上にあるとき, PQ= ρ, P(X, Y, Z) とすると, $\triangle$APQ $\backsim$ $\triangle$ABO より

$$(2 - Z) : \rho = 2 : 1 \Longleftrightarrow 2\rho = 2 - Z \Longleftrightarrow \rho = \frac{2 - Z}{2}$$

よって $\overrightarrow{\mathrm{QP}}$ と x 軸正方向とのなす角を θ , P(x, y, z) とすると,

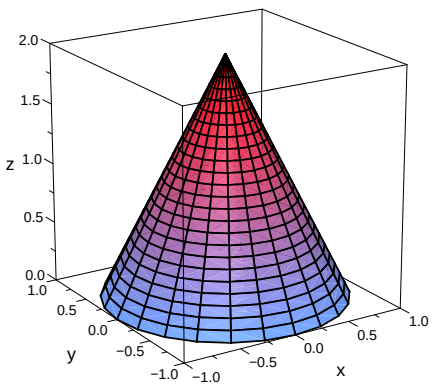
$$\begin{cases} x &= \rho\cos\theta &= \frac{2-Z}{2}\cos\theta \\ y &= \rho\sin\theta &= \frac{2-Z}{2}\sin\theta \quad (0 \leqq \theta < 2\pi, 0 \leqq Z \leqq 2) \\ z &= Z \end{cases}$$

したがって,

$$(x, y, z) = \left(\frac{2 - Z}{2}\cos\theta, \frac{2 - Z}{2}\sin\theta, Z\right) \ (0 \leqq \theta < 2\pi, 0 \leqq Z \leqq 2)$$

とパラメータ表示できるので, 次のように入力します.

- `coneZ := plot :: Surface([(2 − z)/2 ∗ cos(u), (2 − z)/2 ∗ sin(u), z], u = 0..2 ∗ PI, z = 0..2) :`
- `plot(coneZ, Scaling = Constrained)`



注18) 円錐も `plot :: Cylindrical` を使っても描けます. 例えば, 頂点が (0, 0, 2) , 底円の半径が 1 で, 中心が原点の円錐は

`plot :: Cylindrical([(2 − z)/2, phi, z], phi = 0..2 ∗ PI, z = 0..2)`

[円柱と円錐の交線]

以前，low-level primitive の，plot :: Cone, plot :: Cylinder を用いて，円柱と円錐を重ねて描きました．同じ図形を, 今度は plot :: Surface を用いて描いてみます．まず，底面の半径が $\frac{2}{\sqrt{3}}$, 中心が $(0,0,0)$, 頂点が $(0,0,2)$ の円錐を作ります．円錐上の点は, 前節と同様にして,

$$\left(\frac{2-z}{\sqrt{3}}\cos\theta, \frac{2-z}{\sqrt{3}}\sin\theta, z\right)\ (0 \leqq\theta\leqq 2\pi, 0 \leqq z \leqq 2)$$

で表されるので,

```
•mycone := plot :: Surface([(2 − z)/sqrt(3) ∗ cos(u), (2 − z)/sqrt(3) ∗ sin(u), z],
 u = 0..2 ∗ PI, z = 0..2, Mesh = [50, 50], AdaptiveMesh = 3, FillColor = RGB :: Blue) :
```

一方, 軸が x 軸で，半径が 1 ，平面 $x = -1$ と $x = 1$ に挟まれた円柱の側面のうち，$z \geqq 0$ の部分は, $(x, \cos\theta, \sin\theta)$ $(-1 \leqq x \leqq 1, 0 \leqq\theta\leqq\pi)$ と表されるので,

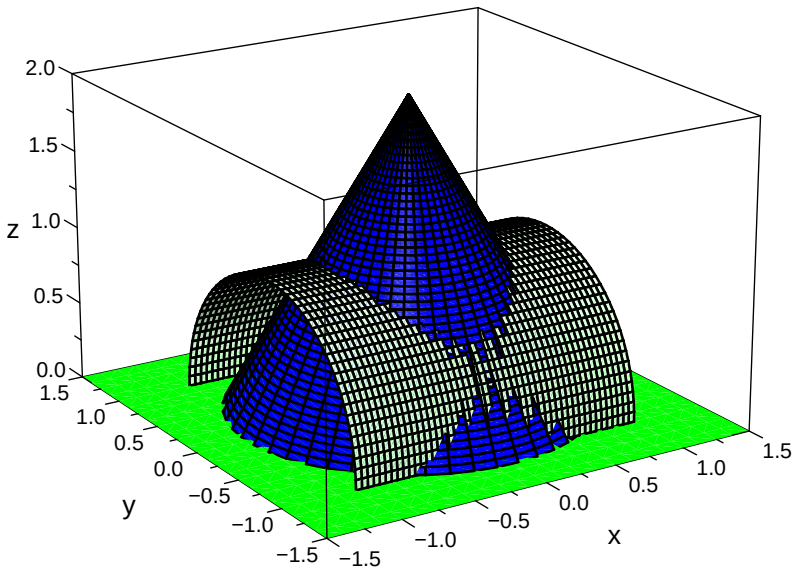
```
•mycylinder := plot :: Surface([x, cos(u), sin(u)], x = −1..1, u = 0..PI,
 Mesh = [50, 50], AdaptiveMesh = 3, FillColor = RGB :: LightGreen) :
```

さらに，xy 平面も描きます.

```
•myplane := plot :: Surface([x, y, 0], x = −1.5..1.5, y = −1.5..1.5,
 ULinesVisible = FALSE, VLinesVisible = FALSE, FillColor = RGB :: Green) :
```

最後に，まとめて plot します．このとき，FillColorType = *Flat* にして,1 つのオブジェクトは，1 つの色で塗るようにしています.

```
• plot(mycone, mycylinder, myplane, Scaling = Constrained, FillColorType = Flat)
```



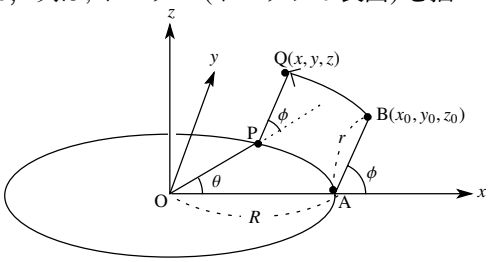
plot :: Surface では，UMesh,VMesh の値を設定できるので，UMesh = 50, VMesh = 50 として，網の目 (mesh) を細かくしました．さらに，AdaptiveMesh の値も, 増加させて，交線の付近の mesh をさらに細

かくしてあります．以前に low-level primitive を使って描いたものより，plot :: Surface を使って描いた図形の方がきれいな気がしますが，いかがでしょうか？[注19)]

[トーラス]

円錐, 円柱は low – level primitive を使っても描けるので，次は, トーラス (ドーナツの表面) を描いてみます．[注20)]

右の図で，A,B は xz 平面上にあり，$\overrightarrow{AB}$ と x 軸正方向とのなす角を ϕ, OA= R, AB= r, B(x_0, y_0, z_0) とすると，



$$\begin{cases} x_0 = R + r\cos\phi \\ y_0 = 0 \\ z_0 = r\sin\phi \end{cases}$$

トーラス上の点を Q(x, y, z) とすると，点 Q は，点 B を z 軸の周りに θ 回転した点だから

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}\begin{pmatrix} R + r\cos\phi \\ 0 \end{pmatrix} = \begin{pmatrix} (R + r\cos\phi)\cos\theta \\ (R + r\cos\phi)\sin\theta \end{pmatrix}$$

Q の z 成分は，B の z 成分と等しいから

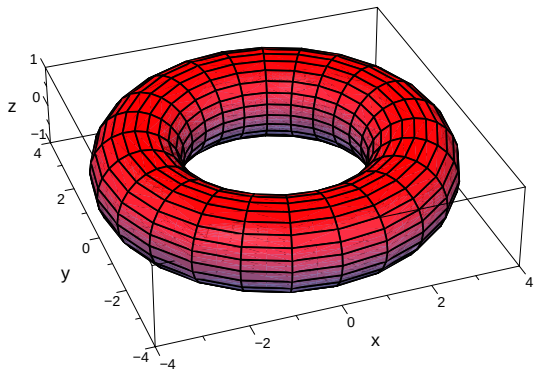
$$z = z_0 = r\sin\phi$$

以上から, トーラス上の点 Q は, 次のようにパラメータ表示されます.

$$[x, y, z] = [\,(R + r\cos\phi)\cos\theta,\ (R + r\cos\phi)\sin\theta,\ r\sin\phi\,] \qquad \cdots (*)$$

$R = 3, r = 1$ として , MuPAD で描いてみます.

```
torus := plot :: Surface([(3 + cos(v)) * cos(u), (3 + cos(v)) * sin(u), sin(v)],
 u = 0..2 * PI, v = 0..2 * PI, Scaling = Constrained)
```



[注19)] plot :: Surface の default では，UMesh= 25,VMesh= 25，AdaptiveMesh= 0 です．また，Mesh= [50,50] は，UMesh = 50, VMesh = 50 と同じ働きをします．さらに，Manual では,「空間では，AdaptiveMesh より，Submesh を使った方が良い」となっていますが，この場合は，AdaptiveMesh を使った方が，はるかに早く描写できました.

[注20)] トーラスは，plot :: Tube を使っても描けます．例えば，

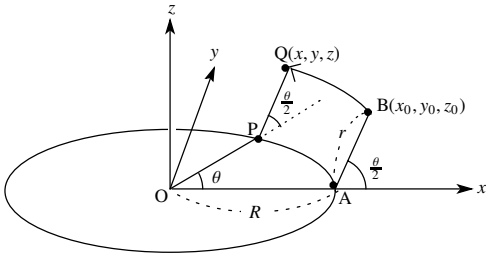
```
plot :: Tube([3 * cos(u), 3 * sin(u), 0], 1, u = 0..2 * PI)
```

で曲線 $(x, y, z) = (3\cos t, 3\sin t, 0)$ $(0 \leqq t \leqq 2\pi)$ を膨らまし，半径 1 としたチューブを描きます.

[メビウスの帯]

メビウスの帯とは，テープを切って, ねじって, 張り合わせたようなものです.

今度はこれを描いてみます．さて，トーラスでは，θ と ϕ は独立でした．ここで，$\phi = \frac{\theta}{2}$ としてみます．すると，R と r が一定で，θ が，$0 \leqq \theta < 2\pi$ の範囲を動くとき，点 P は，地球が太陽の周りを回るように，原点 O の回りを一周します．一方，点 Q は，月が地球の周りを回るように, 点 P の周りを回ります．ただし，月とは異なり，地球 (点 P) の公転面とは垂直に，点 P の周りを**半周**します.

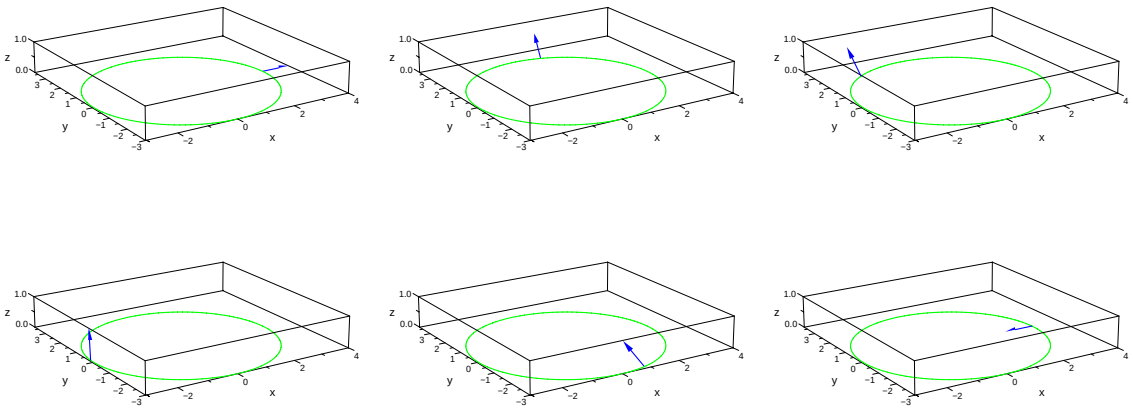


$R = 3, r = 1$ としたときの点 Q を，点 Q_0 とし，$\overrightarrow{PQ_0}$ の動きをアニメーションして見ます．$R = 3, r = 1$ のとき，前節の $(*)$ から,

$$P(3\cos\theta, 3\sin\theta, 0),\ Q_0\left(\left(3+\cos\frac{\theta}{2}\right)\cos\theta,\ \left(3+\cos\frac{\theta}{2}\right)\sin\theta,\ \sin\frac{\theta}{2}\right) \qquad \cdots (**)$$

よって, 次のようにします.

```
•vecPQ := plot :: Arrow3d([(3 + cos(u/2)) * cos(u), (3 + cos(u/2)) * sin(u), sin(u/2)],
 [3 * cos(u), 3 * sin(u), 0], u = 0..2 * PI, Scaling = Constrained) :
•circle := plot :: Curve3d([3 * cos(u), 3 * sin(u), 0], u = 0..2 * PI, Color = RGB :: Green)
•plot(circle, vecPQ)
```



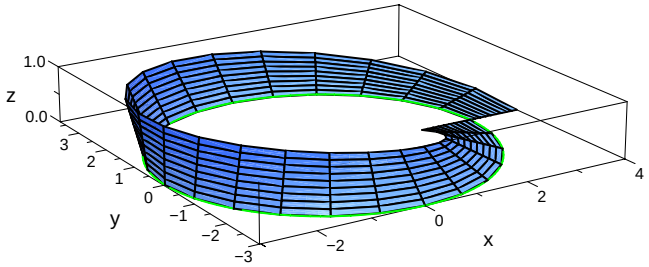
上の図で，最初と最後で $\overrightarrow{PQ_0}$ の向きが逆になっている事にご注意ください.

ここで，r を,$0 \leqq r \leqq 1$ の範囲で変えて，

$$\mathrm{Q}\left(\left(3 + r\cos\frac{\theta}{2}\right)\cos\theta,\ \left(3 + r\cos\frac{\theta}{2}\right)\sin\theta,\ r\sin\frac{\theta}{2}\right) \quad (0 \leqq \theta < 2\pi,\ 0 \leqq r \leqq 1)$$

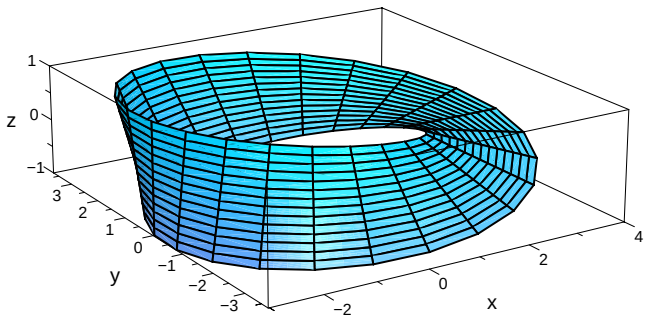
とすると，今度はパラメータが，θ, r の二つあるので曲面になります．これは，$\overrightarrow{\mathrm{PQ}_0}$ の通過領域と一致します．

```
•half := plot :: Surface([(3 + r * cos(u/2)) * cos(u), (3 + r * cos(u/2)) * sin(u), r * sin(u/2)],
  u = 0..2 * PI, r = 0..1, VMesh = 10, Color = RGB :: SkyBlue, Scaling = Constrained) :
•plot(half,circle)
```



さらに，r を $-1 \leqq r \leqq 1$ の範囲で動かします．例えば，$r = -1$ のとき，(**) より，$\overrightarrow{\mathrm{PQ}} = -\overrightarrow{\mathrm{PQ}_0}$ ですから，$\overrightarrow{\mathrm{PQ}_0}$ と $\left(-\overrightarrow{\mathrm{PQ}_0}\right)$ 両方の通過領域が得られます．

```
•mobius := plot :: Surface([(3 + r * cos(u/2)) * cos(u), (3 + r * cos(u/2)) * sin(u), r * sin(u/2)],
  u = 0..2 * PI, r = −1..1, VMesh = 15, Color = RGB :: SkyBlue, Scaling = Constrained) :
•plot(mobius)
```



やっとできました．メビウスの帯は，方向がない曲面として有名です．また，MuPAD の Help には，内側と外側の区別のないことで有名な「クラインの壷」が載っています．ご自分でご覧ください．

12.5　一次変換

MuPAD では, 空間の一次変換：

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

によって, 様々な 図形, グラフを一次変換できます．また，複数の図形の変換もできます.

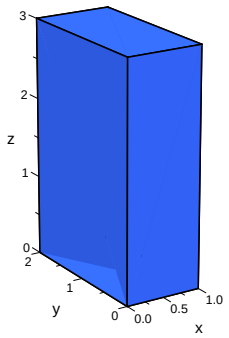
plot :: Translate3d	$\overrightarrow{d} = (d_x, d_y, d_z)$ だけ平行移動 Translate([dx, dy, dz], primitive1, primitive2, ···)
plot :: Scale3d	x, y, z 軸方向に，それぞれ s_x, s_y, s_z 倍に拡大 Scale3d([sx, sy, sz], primitive1, primitive2, ···)
plot :: Rotate3d	点 C を通り, $\overrightarrow{n}$ と平行な直線の周りに θ 回転 Rotate3d(θ, [Cx, Cy, Cz], [nx, ny, nz], primitive1, primitive2, ···)
plot :: Transform3d	行列 A による一次変換 Transform3d(A, primitive1, primitive2, ···)
	$\overrightarrow{d} = (d_x, d_y, d_z)$ 平行移動させたあと，行列 A による一次変換 Transform3d([dx, dy, dz], A, primitive1, primitive2, ···)

注21)

12.5.1　最初の例

$0 \leqq x \leqq 1$, $0 \leqq y \leqq 2$, $0 \leqq z \leqq 3$ の直方体を, low-level primitive の plot :: Box を使って描き, それをいろいろ一次変換してみます.

- box := plot :: Box(0..1, 0..2, 0..3) :　　···$0 \leqq x \leqq 1$, $0 \leqq y \leqq 2$, $0 \leqq z \leqq 3$ の直方体
- plot(box, Scaling = Constrained)



注21) 1. plot :: Rotate3d で，$[C_x, C_y, C_z]$ を省略すると，$[C_x, C_y, C_z] = [0,0,0]$ となります．また，$[n_x, n_y, n_z]$ を省略すると，$[n_x, n_y, n_z] = [0,0,1]$ となります.
2. 行列は, MuPAD の 3×3 matrix (行列), または，3×3 array , または，9 個の要素からなる list で表します.

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

のとき，これを MuPAD の行列で表すには，matrix を使って,

$$A := \text{matrix}([[a_{11}, a_{12}, a_{13}], [a_{21}, a_{22}, a23], [a_{31}, a_{32}, a_{33}]])$$

のようにしますが，plot :: transform の場合は，matrix を省略して，$A := [[a_{11}, a_{12}, a_{13}], [[a_{21}, a_{22}, a_{23}], [a_{31}, a_{32}, a_{33}]]$ とすることができます．さらに，$A := [a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a23, a_{31}, a_{32}, a_{33}]$ のように，1 行のリストにすることもできます.

次にこれを，plot :: Scale3d を用いて，x 軸方向に 2 倍，y 軸方向に 1 倍，z 軸方向に $\frac{2}{3}$ 倍拡大して, 一辺が 2 の立方体 cubic を作ります.

- cubic := plot :: Scale3d([2, 1, 2/3], box) :

この cubic を，plot :: Translate を用いて，$(4, 0, 3)$ 平行移動します.

- box1 := plot :: Translate3d([4, 0, 3], cubic) :

今度は，plot :: Rotate3d を使って，もとの box を $\frac{\pi}{4}$ 回転させます．回転軸は, 点 $C(2, 0, 0)$ を通り, $\overrightarrow{n} = (0, 1, 0)$ と平行な直線とします.

- box2 := plot :: Rotate3d(PI/2, [2, 0, 0], [0, 1, 0], box)

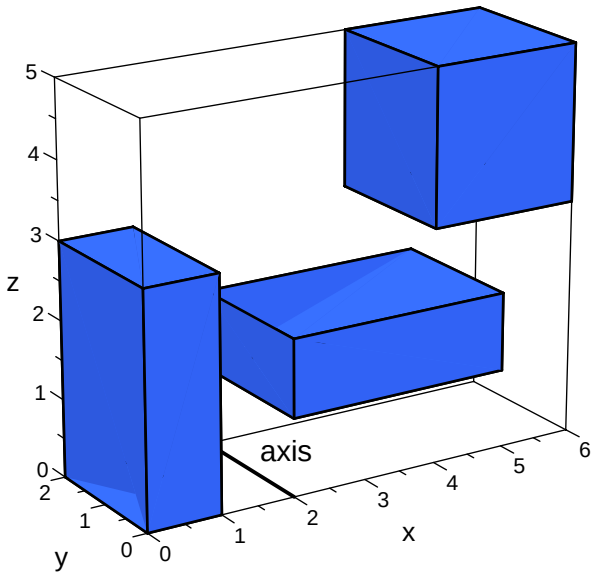
回転軸やタイトルも作っておきます.

- axis := plot :: Line3d([2, 0, 0], [2, 2, 0], LineColor = RGB :: Black, LineWidth = 0.5) :
- title := plot :: Text3d("axis", [2.5, 1, 0]) :

最後に，box, box1, box2, axis, title を, plot します.

- plot(box, box1, box2, axis, title)

これで次のようになります．box2 は，box を，回転軸 (axis) の周りに，$\frac{\pi}{2}$ 回転しているのが分かります．ここで，回転の向きは, いわゆる「右手の法則」に従います．すなわち，右手の親指を，方向ベクトル と同じ向きにとった時に, 残りの指の指す方向が回転方向です.



12.5.2　アニメーション

plot :: Rotate3d を使って, 月と地球のアニメーションを描いてみます．ただ, 描写の都合上, 月は 3 日で地球を一周するものとします．(^_^;)
まずは, 地球を，原点中心, 半径 1 の球とします.

- earth := plot :: Surface([cos(u) ∗ sin(v), sin(u) ∗ sin(v), cos(v)], u = 0..2 ∗ PI, v = 0..PI, FillColor = RGB :: Aqua) :

地球の上に, 点も付け加えます.

- japan := plot :: Point3d([sin(PI/3), 0, cos(PI/3)], PointSize = 2)

次に月をつくります．とりあえず，(3, 0, 0) を中心とし，半径は 0.3 の球とします.

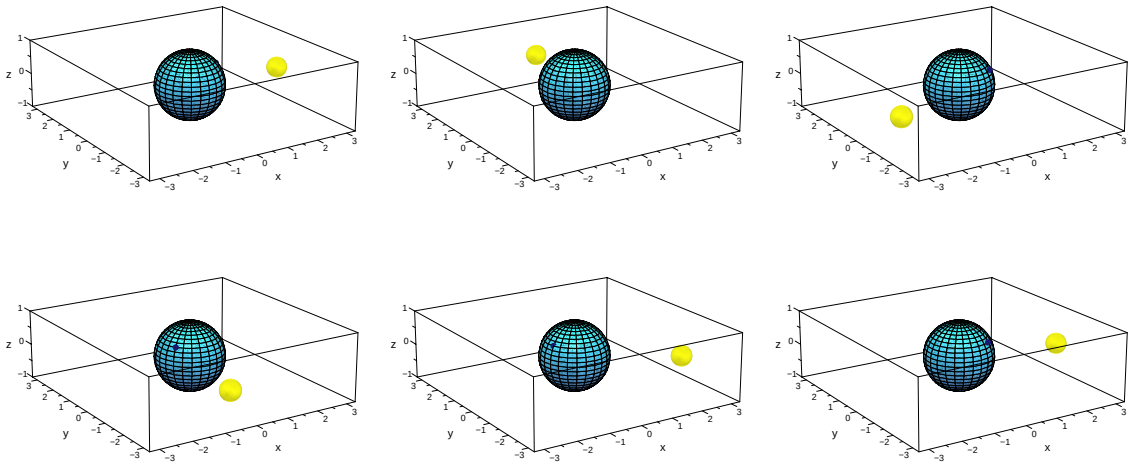
- moon := plot :: Sphere(0.3, [3, 0, 0], FillColor = RGB :: Yellow) :

もちろん，このままでは, 月も地球も動きません．plot :: Rotate3d を使って，アニメーションにします．z 軸周りの a (rad) の 回転は，plot :: Rotate3d(a, [0, 0, 0], [0, 0, 1], object$_1$, $\cdots$) ですが，これは plot :: Rotate3d(a, object$_1$, $\cdots$) と省略できます．よって，次のようにします.

- rotatingEarth := plot :: Rotate3d(a, earth, japan, a = 0..6 ∗ PI) :
- rotatingMoon := plot :: Rotate3d(a, moon, a = 0..2 ∗ PI) :

これで，1 回のアニメーションで，地球 (earth と japan) は, 6π 回転し，月は，2π 回転します．最後に，地球と月をまとめて plot します.

- plot(rotatingEarth, rotatingMoon, Scaling = Constrained)



このように,「一次変換」 でもアニメーションが使えます.

12.6 特殊な primitive

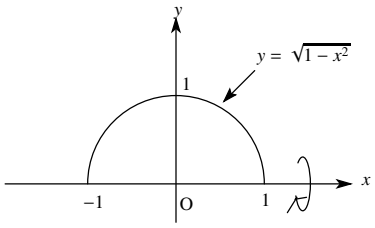
12.6.1 x, z 軸周りの回転体 (plot::XRotate, plot::ZRotate)

plot :: XRotate	y = f(x) (a ≦ x ≦ b) を, x 軸の周りに回転した回転体 XRotate(f(x), x = a..b)
plot :: ZRotate	z = f(x) (a ≦ x ≦ b) を, z 軸の周りに回転した回転体 ZRotate(f(x), x = a..b)

[注22] $y = f(x)$ を，x 軸の周りに回転した回転体や，$z = f(x)$ を，z 軸の周りに回転した回転体を簡単に描く事ができます．さらに，default では，回転角 θ は，$0 \leqq \theta < 2\pi$ ですが，その範囲を AngleRange = $\theta_1..\theta_2$ で指定できます．[注23]

[x 軸の周りの回転]

円の上半分:$y = \sqrt{1-x^2}$ を x 軸の周りに回転してみます．



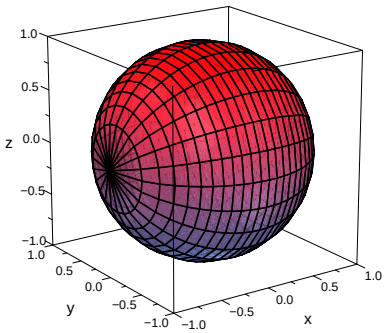
- sphere := XRotate(sqrt(1 − x^2), x = −1..1) :

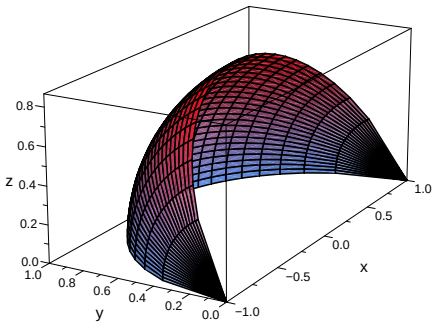
こんどは，$0 \leqq \theta \leqq \frac{\pi}{3}$ だけ回転させて見ます．

- feather := XRotate(sqrt(1 − x^2), x = −1..1, AngleRange = 0..PI/3) :

これを描いてみます．

- plot(sphere); plot(feather)





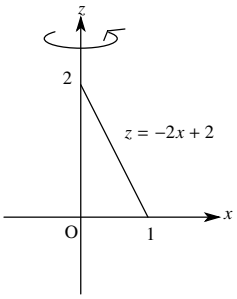
回転の向きは，¢右手の法則」に従っています．すなわち，右手の親指を回転軸に向けた時, 他の指の向く方向が正の方向です．

[注22] この primitive は，MuPAD では，high-lebel primitive として分類されていますが，plot :: Surface などと比べると, かなり特殊と思われるので, ここに分類しました．

[注23] plot :: XRotate, plot :: ZRotate は, 内部で plot :: Surface を使用していて，x 成分，回転角の順に, パラメータ u, v と解釈されます．そして，plot :: Surface の *Attribute* はほとんど全て受け付けます．よって，ULineVisible，VLineVisible などを使って，網の目 (mesh) を見えなくしたり，UMesh, VMesh などで，mesh を細かくすることができます．
また，plot :: XRotate は,plot :: Function3d と同様，( ) 内に primitive ではなく，**式** が入ることに注意してください．

[z 軸の周りの回転]

xz 平面の線分: $z = 2 - 2x$ を z 軸の周りに回転してみます.



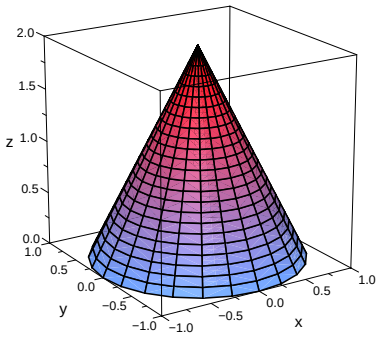
• cone := plot :: ZRotate(2 − 2 ∗ x, x = 0..1) :

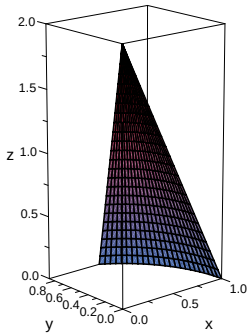
こんどは，$0 \leqq \theta \leqq \frac{\pi}{3}$ だけ回転させて見ます.

• feather := plot :: ZRotate(2 − 2 ∗ x, x = 0..1, AngleEnd = PI/3) :

これを描いてみます.

• plot(cone); plot(feather)





このように，XRotate,ZRotate を使っても，球，円錐，円柱などが簡単に描けます. [注24)]

12.6.2　その他の primitives

その他，特別な primitive がまだまだあります.

円柱座標, 球面座標, Tube 座標による描写

plot :: Cylindrical, plot :: Spherical, plot :: Tube

これらは，それぞれ, 球面, 円柱, チューブ (トーラス) などの描写をやり易くしますが，全て plot :: Surface を用いてもできます.

シーン, 座標系の追加

plot :: Scene3d, plot :: CoordinateSystem3d

plot :: Scene3d, plot :: CoordinateSystem3d は，1 つの Canvas の中に，2 つ以上の Scene3d を入れたり，1 つの Scene3d の中に,2 つ以上の異なる座標系を入れたりするとき使います.

[注24)] 円柱は,•plot(plot :: XRotate(1, x = 0..2)) でかけます.

直接光，間接光，光る点，スポットライトの設定

plot :: DistantLight, plot :: AmbientLight, plot :: PointLight, plot :: SpotLight

この 4 つは, 光の当て具合を調整する時に使います．順に, 直接光, 間接光，ポイントライト (電球のような一点だけの明かり)，スポットライトを表します.

Group 化

plot :: Group3d

plot :: Group3d は, いくつかの 3d の 図形をまとめて扱いたいときに使います.

カメラ

plot :: Camera

視点 (カメラ) の位置を調整する時に使います．これらは空間においてのみ設定可能です．また，自動的に設定されるので，自分では調整する必要はありません．特別なことをやりたい時のみ使用します.

注25)

ここでは， plot :: Group3d, plot :: Camera のみ少し取り上げます.

[グループ化 (plot::Group3d)]

plot :: Group3d は，次のようにして使います.

$$\texttt{plot :: Group3d}(\texttt{object}_1, \texttt{object}_2, \cdots)$$

$\texttt{object}_1, \texttt{object}_2, \cdots$ は, 任意の空間の オブジェクト です．これを使うと，プログラムが解りやすくなるだけでなく，*Attribute* を，まとめて指定できるので便利です．茶色のテーブルを作って見ます.

先ず, 4 本の足と，天板を作ります.

- leg1 := plot :: Line3d([5, 0, 0], [5, 0, 2]) :
- leg2 := plot :: Line3d([5, 2, 0], [5, 2, 2]) :
- leg3 := plot :: Line3d([6, 0, 0], [6, 0, 2]) :
- leg4 := plot :: Line3d([6, 2, 0], [6, 2, 2]) :
- tableboard := plot :: Box(5..6, 0..2, 2..2.1, FillColor = RGB :: Brown) :

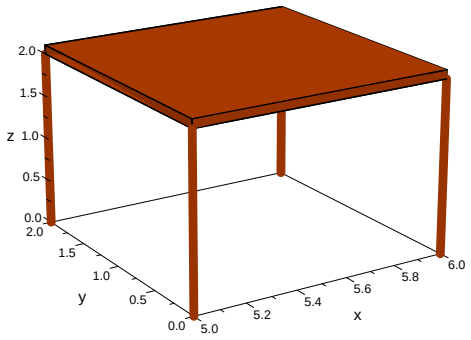
注25) 平面の時は，Scene3d, CoordinateSystem3d, Group3d の代わりに，それぞれ，Scene2d, CoordinateSystem2d, Group2d となります.

これを，plot :: Group3d を使って，一つにまとめます．先ず, 4 本の足をまとめます.

- legs := plot :: Group3d(leg1, leg2, leg3, leg4, LineWidth = 2, LineColor = RGB :: Brown) :

これで，leg1, leg2, leg3, leg4 の全てが，LineWidth = 2, LineColor = RGB :: Brown になります．これに，天板をくっつけて，plot します.

- mytable := plot :: Group3d(legs, tableboard) : plot(mytable)

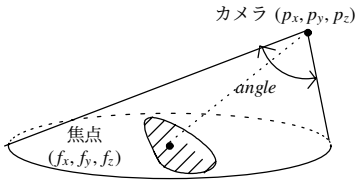


[視点の位置の変更 (plot::Camera)]

plot :: Camera は，次のようにして使います.

plot :: Camera([p_x, p_y, p_z], [f_x, f_y, f_z], angle)

ここで，(p_x, p_y, p_z) は, カメラの位置の x, y, z 成分を示し，(f_x, f_y, f_z) は焦点の位置，*angle* は，カメラの**視界の**角度を表しています．したがって，*angle* が大きくなると, カメラが見る範囲は広くなり, 逆に図形の大きさは小さくなります．また *angle* の大きさは，π を超えない正の数で指定します．(angle　π の場合は, いわゆる, 魚眼 になってしまいます.)



焦点 (focal point) は, 単に画面の中心の点を表していて，焦点がずれると「**暈 (ぼ) ける**」という訳ではなさそうです．先ほどの mytable を, カメラの高さを変えながら, 真上から見てみます．まず，camera を作ります.

- camera1 := plot :: Camera([5.5, 1, 4], [5.5, 1, 2], PI/2) :
- camera2 := plot :: Camera([5.5, 1, 4], [5.5, 1, 3], PI/2) :
- camera3 := plot :: Camera([5.5, 1, 4], [5.5, 1, 3.9], PI/2) :
- camera4 := plot :: Camera([5.5, 1, 4], [5.5, 1, 4], PI/2) :
- camera5 := plot :: Camera([5.5, 1, 4], [5.5, 1, 5], PI/2) :

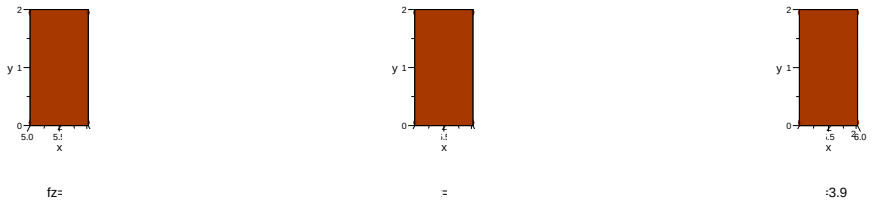
mytable を, 上のカメラを使いながら, 描写してみます.

- plot(mytable, camera1, Footer = "fz = 2", Scaling = Constrained)

⋮

- plot(mytable, camera5, Footer = "fz = 5" Scaling = Constrained)

これで, 次のようになります. 最後は何も見えません. これはカメラの位置より, 焦点が高い位置にあるので, 天井しか見えていないからです.



このように, $[f_x, f_y, f_z]$ は, $[p_x, p_y, p_z]$ からの方向だけが, 問題です.

[アニメーション]

カメラ (plot :: Camera) を, アニメーションにすることもできます.

テーブルの他に, ベッドを作ります. 先ず, ベッドの床と, 頭, そして布団を作ります.

- bedbase := plot :: Box(4.1..6, 3..4, 0..1, Color = RGB :: Blue) :
- bedhead := plot :: Box(4..4.1, 3..4, 0..2, Color = RGB :: Blue) :
- futon := plot :: Box(4.1..6, 3..4, 1..1.2, Color = RGB :: White) :

これらをまとめて, ベッドを作ります.

- bed := plot :: Group3d(bedbase, bedhead, futon) :

次に, 本箱とタンスを作ります.

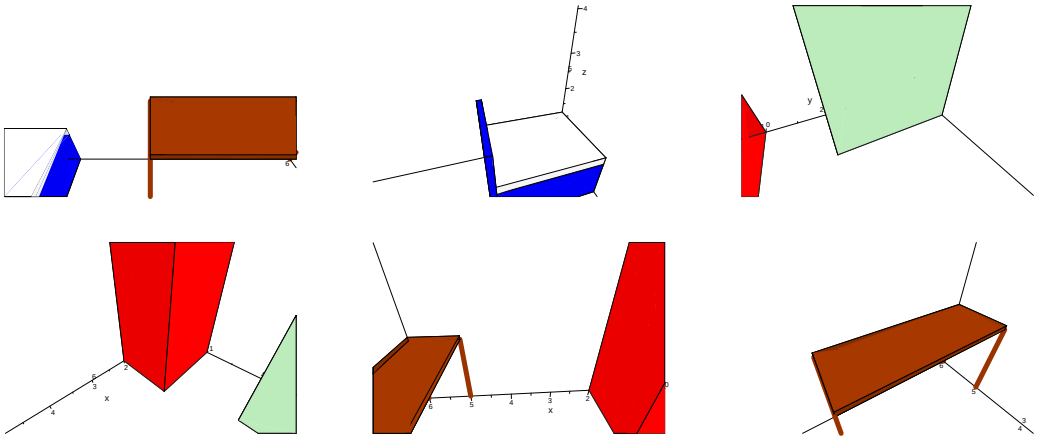
- bookshelf := plot :: Box(0..1.5, 2..4, 0..5, Color = RGB :: LightGreen) :
- closet := plot :: Box(0..2, 0..1, 0..5, Color = RGB :: Red) :

次に, カメラを作ります. 位置は部屋の中央付近に固定して, 焦点を, ぐるぐる回転させます.

- camera := plot :: Camera([4, 2, 4], [cos(a) + 4, sin(a) + 2, 3], PI/3, a = 0..2 ∗ PI) :

最後に, まとめて plot します.

- plot(bed, mytable, bookshelf, closet, camera)



自分で作ってジーと見ると, 目が回ります. このように, ほとんど全てのオブジェクトは, アニメーションになります.

第13章　行列, ベクトル

13.1　行列, ベクトルの基本

行列, ベクトルの定義	matrix または Dom :: Matrix(R)
行列, ベクトルの和, 差, 行列の積, 累乗	+, −, ∗, ^
A の逆行列	A^(−1) または 1/A
A の (i, j) 成分	A[i, j]

この他の (stlib の) いろいろな演算が, 文字式同様，行列, ベクトルに対しても使えます.

diff(A, x), int(A, x)	各要素を微分, 積分
expand(A)	各要素を展開
factor(A)	各要素を因数分解
normal(A)	各要素を約分
simplify(A), Simplify(A)	簡略化
exp(A)	$E + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots$ を計算
subs(A, x = a)	A の各要素で, x に a を代入
iszero(A)	$A = O$ かチェックする
float(A)	A の各要素に float を働かせる
conjugate(A)	A の各要素の共役成分をとる
map(A, function)	A の各要素に function を働かせる

13.1.1　行列, ベクトルの定義

行列, ベクトルの定義は matrix([[第一行], [第 2 行], ⋯ , [第 n 行]]) のように, リスト [ ] を利用して, 定義するのが，手っ取り早いです.

- matrix([[2, 5, 7]])　　>> (2, 5, 7)
- matrix([[3], [4], [5]])　　>> $\begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix}$

matrix([[2, 5, 7]] は要素がひとつ ([2,5,7] だけ) なので, 行ベクトルに，一方，matrix([[3], [4], [5]]) は要素が 3 つなので, 列ベクトルになります．なお，列ベクトルは, 内側の [ ] を省略しても, 大丈夫です．このように，ベクトルは，行ベクトル, 列ベクトルとして定義できます.

- V := matrix([3, 4, 5])　　>> $\begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix}$

行列も同様に定義できます.

- `A := matrix([[1, 4], [7, 10]])` >> $\begin{pmatrix} 1 & 4 \\ 7 & 10 \end{pmatrix}$

行列 A の (i, j) 成分は,A[i, j] で取り出せます.

- `A[1, 1]; A[1, 2]; A[2, 1]; A[2, 2]` >> 1　>> 4　>> 7　>> 10

ベクトル V の第 i 成分は,V[i] でも，取り出せます.

- `V[1]; V[2]; V[3]` >> 3　>> 4　>> 5

明示的に，行列の型を指定することもできます．$m \times n$ 型行列の場合，`matrix(m,n,`$\cdots$`)` とします.

- `matrix(1, 3, [5, 6, 7])` >> (5, 6, 7)
- `matrix(3, 1, [5, 6, 7])` >> $\begin{pmatrix} 5 \\ 6 \\ 7 \end{pmatrix}$

上のほうは 1×3 行列，下の方は 3×1 行列になっています．一般の行列でも同じです.

- `matrix(2, 3, [[a, b, c], [d, e, f]])` >> $\begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix}$

リストが不足したときは,「0」で空きを埋めます.

- `matrix(2, 3, [[a, b]])` >> $\begin{pmatrix} a & b & 0 \\ 0 & 0 & 0 \end{pmatrix}$
- `matrix(2, 3, [a, b])` >> $\begin{pmatrix} a & 0 & 0 \\ b & 0 & 0 \end{pmatrix}$

オプションで `Diagonal`(対角的) をつけると，リストの成分を対角線 $(a_{11}, a_{22}, \cdots)$ に並べます.

- `matrix(2, 3, [a, b], Diagonal)` >> $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \end{pmatrix}$

普通は, 型を指定する必要はありませんが，零行列や単位行列を作るときは，役立ちます.

- `o := matrix(3, 3)` >> $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$
- `e := matrix(3, 3, [1, 1, 1], Diagonal)` >> $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

このとき，大文字の O や，E や, I は，MuPAD が内部で使用しているので，使えません．なお，n 次の単位行列は，`matrix :: identity(n)` でも定義できます.

- `e := matrix :: identity(3)` >> $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

13.1.2 基本的な計算

行列の加減乗除は実数と同じ様に計算できます．A の逆行列は 1/A でも A^(−1) でも計算できます．また逆行列が存在しないときは *FAIL* というエラーがでます．

- `u := matrix([[1,2]])` >> $(1,2)$ $\cdots\overrightarrow{u}$ の定義
- `v := matrix([2,3])` >> $\begin{pmatrix}2\\3\end{pmatrix}$ $\cdots\overrightarrow{v}$ の定義
- `u * v` >> 8 $\cdots\begin{pmatrix}1&2\end{pmatrix}\begin{pmatrix}2\\3\end{pmatrix}=8$
- `v * u` >> $\begin{pmatrix}2&4\\3&6\end{pmatrix}$ $\cdots\begin{pmatrix}2\\3\end{pmatrix}\begin{pmatrix}1&2\end{pmatrix}=\begin{pmatrix}2&4\\3&6\end{pmatrix}$
- `A := matrix([[1,2],[3,4]])` >> $\begin{pmatrix}1&2\\3&4\end{pmatrix}$ $\cdots$A の定義
- `B := matrix([[1,2],[-1,-2]])` >> $\begin{pmatrix}1&2\\-1&-2\end{pmatrix}$ $\cdots$B の定義
- `2 * A + 3 * B` >> $\begin{pmatrix}5&10\\3&2\end{pmatrix}$ $\cdots$ 2A+3B
- `A * B` >> $\begin{pmatrix}-1&-2\\-1&-2\end{pmatrix}$ $\cdots$AB
- `B * A` >> $\begin{pmatrix}7&10\\-7&-10\end{pmatrix}$ $\cdots$BA
- `A^(-1)` >> $\begin{pmatrix}-2&1\\\frac{3}{2}&-\frac{1}{2}\end{pmatrix}$ $\cdots\mathrm{A}^{-1}$
- `1/A` >> $\begin{pmatrix}-2&1\\\frac{3}{2}&-\frac{1}{2}\end{pmatrix}$ $\cdots\mathrm{A}^{-1}$
- `B^(-1)` >> *FAIL* $\cdots\mathrm{B}^{-1}$ はない
- `A * A^(-1);` >> $\begin{pmatrix}1&0\\0&1\end{pmatrix}$ $\cdots\mathrm{AA}^{-1}=\mathrm{E}$
- `A^2` >> $\begin{pmatrix}7&10\\15&22\end{pmatrix}$ $\cdots\mathrm{A}^2$
- `A^3` >> $\begin{pmatrix}-17&-34\\-37&-74\end{pmatrix}$ $\cdots\mathrm{A}^3$

- `assume(n, Type :: PosInt) : A^n` >> Error: Wrong type of 2. argument (type 'DOM_INT' expected, got argument 'n'); during evaluation of '(Dom::Matrix (Dom::ExpressionField()))::_power'

$det\ B = 1\times(-2)-2\times(-1)=0$ ですから,B の逆行列は存在しません．[注1)] 残念ながら，行列の 2 乗，3 乗は計算できても，n 乗は計算できないみたいです．成分が文字の行列も, 同様に計算できます. ただし,

[注1)] $A=\begin{pmatrix}a&b\\c&d\end{pmatrix}$ のとき, $det(A)=ad-bc$ と定義して, A の行列式 (*determinant*) といいます. $det(A)=0$ のとき,A^{-1} は存在しません. $det(A)\neq 0$ のとき,$A^{-1}=\frac{1}{ad-bc}\begin{pmatrix}d&-b\\-c&a\end{pmatrix}$ です.

このとき，例えば，a := matrix([1, 2]) としたのを忘れて，matrix(a, b) などとするミスが, 起こりがちです．(私の場合だけかもしれませんが...) 忘れずに定義を消去しておきます．

- delete a, b, c, d　　>>　　$\cdots a, b, c, d$ の定義の消去
- A := matrix([[a, b], [c, d])　　>> $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$　　$\cdots$ A の定義
- e := matrix([[1, 0], [0, 1]])　　>> $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$　　$\cdots$ e(単位行列) の定義
- A * e　　>> $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$　　$\cdots AE = A$
- 1/A　　>> $\begin{pmatrix} \frac{d}{a\cdot d - b\cdot c} & -\frac{b}{a\cdot d - b\cdot c} \\ -\frac{c}{a\cdot d - b\cdot c} & \frac{a}{a\cdot d - b\cdot c} \end{pmatrix}$　　$\cdots A^{-1}$

$ad - bc = 0$ のときは逆行列は存在しないはずですが, とりあえずこの式を出してきます．お次は「ケーリー・ハミルトンの定理」の検証です．

- A^2 − (a + d) * A + (a * d − b * c) * e　　>> $\begin{pmatrix} -a\cdot(a+d) + a\cdot d + a^2 & -b\cdot(a+d) + a\cdot b + b\cdot d \\ -c\cdot(a+d) + a\cdot c + c\cdot d & -d\cdot(a+d) + a\cdot d + d^2 \end{pmatrix}$
- simplify(%)　　>> $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

たしかに，$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ のとき，

$$A^2 - (a+d)A + (ad - bc)E = O$$

となります．

13.1.3　行列の一般的な定義

特別な行列が欲しい時は，もっと一般的な定義：Dom :: Matrix(R) を使います．ここで，R には,Dom :: Real, Dom :: Complex, Dom :: Integral, Dom :: ExpressionField() のような環 (ring) を指定します．例えば, 整数だけを要素に持つ行列を作りたければ，R として，Dom :: Integer を指定します．

- intmatrix := Dom :: Matrix(Dom :: Integer)　　>> Dom::Matrix(Dom::Integer)
- intmatrix([[1, 2], [3, 4]])　　>> $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

intmatrix は，生成子 (コンストラクタ) のような働きをしています．ここで，もし, 要素を整数以外にすると，エラーがでます．

- intmatrix([[0.5, 2], [3, 4]])

  >> Error: unable to define matrix over Dom::Integer [(Dom::Matrix(Dom::Integer))::new]

matrix は，Dom :: Matrix(Dom :: ExpressionField()) の省略で，R が Dom :: ExpressionField() (数式の環) の場合です．よって，数式も要素に持つことができます．

- matrix　　>> Dom::Matrix()
- mymatrix := Dom :: Matrix(Dom :: ExpressionField())　　>> Dom::Matrix()

全く同じ出力です．ちなみに，Dom :: Matrix() も Dom :: Matrix(Dom :: ExpressionField()) の略です．

- A := mymatrix([[cos(x), −sin(x)], [sin(x), cos(x)]]) >> $\begin{pmatrix} \cos(x) & -\sin(x) \\ \sin(x) & \cos(x) \end{pmatrix}$
- 1/A >> $\begin{pmatrix} \frac{\cos(x)}{\cos(x)^2+\sin(x)^2} & \frac{\sin(x)}{\cos(x)^2+\sin(x)^2} \\ -\frac{\sin(x)}{\cos(x)^2+\sin(x)^2} & \frac{\cos(x)}{\cos(x)^2+\sin(x)^2} \end{pmatrix}$
- simplify(%) >> $\begin{pmatrix} \cos(x) & \sin(x) \\ -\sin(x) & \cos(x) \end{pmatrix}$

たしかに，

$$\begin{pmatrix} \cos(x) & -\sin(x) \\ \sin(x) & \cos(x) \end{pmatrix}^{-1} = \begin{pmatrix} \cos(x) & \sin(x) \\ -\sin(x) & \cos(x) \end{pmatrix}$$

が成り立っています．以上のことは，mymatrix の代わりに，matrix を使っても同じです．

13.2 高校の数学から

この節は, 高校の数学を例にして，2×2 行列について見ていきます．

13.2.1 2 次正方行列の n 乗 (固有多項式の利用)

行列の n 乗は，高校の教科書や大学入試に, 非常に多く見ることができます．しかし，MuPAD では，n が文字のとき，A^n は計算できません．そこで,「整式の割り算」を利用して，手動で求めてみます．

例題 1

$A = \begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix}$, $E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $f(x) = x^n$ (n は自然数) について次の問いに答えよ．

(1) $A^2 - 7A + 10E = O \cdots (*)$ が成り立つことを示せ．

(2) $f(x)$ を $x^2 - 7x + 10$ で割ったときの余りを求めよ．

(3) A^n を求めよ．

【解答】
(1)(略) ケーリー・ハミルトンの定理より明らか.
(2)$x^2-7x+10$ は 2 次式なので, 余りは $px+q(p,q$ 定数) とおける．さらに商を $g(x)$ とおくと

$$f(x)=(x^2-7x+10)g(x)+px+q=(x-2)(x-5)g(x)+px+q \quad \cdots①$$

① に $x=2, x=5$ を代入して

$$\begin{cases} f(2)=2^n=2p+q \\ f(5)=5^n=5p+q \end{cases} \Longleftrightarrow \begin{cases} p=\frac{5^n-2^n}{3} \\ q=\frac{5\cdot 2^n-2\cdot 5^n}{3} \end{cases}$$

よって求める余りは

$$\frac{5^n-2^n}{3}x+\frac{5\cdot 2^n-2\cdot 5^n}{3} \quad \cdots(答)$$

(3) ① で x に A を代入して

$$f(A)=A^n=(A^2-7A+10E)g(A)+pA+qE \quad \cdots②$$

(1) より $A^2-7A+10E=O$ だから

$$\begin{aligned} A^n &= pA+qE \\ &= \frac{5^n-2^n}{3}A+\frac{5\cdot 2^n-2\cdot 5^n}{3}E \\ &= \frac{5^n-2^n}{3}\begin{pmatrix}3 & 1\\ 2 & 4\end{pmatrix}+\frac{5\cdot 2^n-2\cdot 5^n}{3}\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} \qquad = \frac{1}{3}\begin{pmatrix}5^n+2\cdot 2^n & 5^n-2^n\\ 2\cdot 5^n-2\cdot 2^n & 2\cdot 5^n+2^n\end{pmatrix} \end{aligned} \quad \cdots(答)$$

注2)

これを MuPAD でやらせるにはどうすればよいでしょう? 実は，MuPAD には，ケーリー・ハミルトンの多項式 (固有多項式) を作るコマンドはあります．しかし n が自然数のとき x^n を多項式で割った余りを求めることは出来ません．[注3)] そこで我々のほうで解を求める手順を作る必要があります．いま $A=\begin{pmatrix}a & b\\ c & d\end{pmatrix}$ の固有方程式:

$$x^2-(a+d)x+(ad-bc)=0 \quad \cdots(*)$$

の 2 解を α,β とします．$f(x)=x^n$ を $(*)$ で割ったときの商を $g(x)$, 余りを $px+q$ とおくと,

$$f(x)=\{x^2-(a+d)x+(ad-bc)\}g(x)+px+q$$

(i) **$\alpha\neq\beta$のとき,** 上の式に α,β を代入して

$$\begin{cases} f(\alpha)=\alpha^n=p\alpha+q & \cdots① \\ f(\beta)=\beta^n=p\beta+q & \cdots② \end{cases}$$

注2) ① の x に A を代入するときに 実数の 1(単位元) を単位行列 E に変えるのを注意しましょう．このように 1 変数 x のみの整式には行列 A を代入してもかまいません．例えば,

$$(x+2)+(3x+1)=4x+2 \Longrightarrow (A+2E)+(3A+E)=4A+3E$$
$$x(x+2)+1=x^2+2x+1 \Longrightarrow A(A+2E)+E=A^2+2A+E$$

しかし一般に，2 変数 x,y の整式に，行列 A,B を代入することはできません.

$$(x+y)^2=x^2+2xy+y^2 \Longrightarrow (A+B)^2=A^2+2AB+B^2 \text{ (これは誤り)}$$

一般に $AB\neq BA$ ですから，$(A+B)^2=(A+B)(A+B)=A^2+AB+BA+B^2$ から簡単に出来ません.
注3) 行列 A の固有多項式 (characteristic polynomial) は，`linalg :: charpoly(A)` で求まります.

①,② より

$$p = \frac{\alpha^n - \beta^n}{\alpha - \beta}, \quad q = \frac{\alpha\cdot\beta^n - \beta\cdot\alpha^n}{\alpha - \beta}$$

ここで $\alpha - \beta \neq 0$ ☜ を使ったことに注意.

ゆえに

$$f(x) = \{x^2 - (a+d)x + (ad-bc)\}g(x) + \frac{\alpha^n - \beta^n}{\alpha - \beta}x + \frac{\alpha\cdot\beta^n - \beta\cdot\alpha^n}{\alpha - \beta} \quad \cdots ③$$

$A^2 - (a+d)A + (ad-bc)E = O$ だから③ に A を代入すると

$$A^n = \frac{\alpha^n - \beta^n}{\alpha - \beta}A + \frac{\alpha\cdot\beta^n - \beta\cdot\alpha^n}{\alpha - \beta}E$$

(ii) **$\alpha = \beta$のとき,** $x^2 - (a+d)x + (ad-bc) = (x-\alpha)^2$ となるから, ③ は

$$f(x) = x^n = (x-\alpha)^2 g(x) + px + q \quad \cdots ④$$

両辺を x で微分して

$$f'(x) = nx^{n-1} = 2(x-\alpha)g(x) + (x-\alpha)^2 g'(x) + p \quad \cdots ⑤$$

④,⑤ へ α を代入して

$$\begin{cases} \alpha^n = p\alpha + q \\ n\alpha^{n-1} = p \end{cases} \Longleftrightarrow \begin{cases} p = n\alpha^{n-1}, \\ q = (1-n)\alpha^n \end{cases}$$

④ へ代入して

$$f(x) = (x-\alpha)^2 g(x) + n\alpha^{n-1}x + (1-n)\alpha^n \quad \cdots ⑥$$

$(A - \alpha E)^2 = O$ だから, ⑥ に A を代入すると

$$A^n = n\alpha^{n-1}A + (1-n)\alpha^n E$$

以上まとめると

$$\begin{cases} \text{(i) } \alpha \neq \beta \text{のとき}, & A^n = \dfrac{\alpha^n - \beta^n}{\alpha - \beta}A + \dfrac{\alpha\cdot\beta^n - \beta\cdot\alpha^n}{\alpha - \beta}E \\ \text{(ii) } \alpha = \beta \text{のとき}, & A^n = n\alpha^{n-1}A + (1-n)\alpha^n E \end{cases} \quad \cdots (**)$$

ここで α, β は

$$x^2 - (a+d)x + (ad-bc) = 0 \quad \cdots (*)$$

の 2 解でした. $(*)$ を固有方程式, $(*)$ の解を固有値といいます. 幸いにして MuPAD には固有値を求めるコマンドがあります. `linalg :: eigenvalues(A)` です.
注4)

注4) linalg は linear algebra(線形代数) の略で, eigen はドイツ語で ' 固有の' という意味です. 固有値は普通 2 個以上あるので複数形になっています.

それでは (**) の結果を使って 例題 1(3) を解いてみます．次のようにします．

- `A := matrix([[3,1],[2,4]]);` >> $\begin{pmatrix}3 & 1\\ 2 & 4\end{pmatrix}$ ⋯ A の定義
- `e := matrix([[1,0],[0,1]]);` >> $\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$ ⋯ 単位行列の定義
- `linalg :: eigenvalues(A);` >> {2, 5} ⋯ これが固有値

そこで，$\alpha = 5, \beta = 2$ として (**) に代入すると，

- `(5^n - 2^n)/(5 - 2) * A + (5 * 2^n - 2 * 5^n)/(5 - 2) * e` >> $\begin{pmatrix}\frac{2\cdot 2^n}{3}+\frac{5^n}{3} & -\frac{2^n}{3}+\frac{5^n}{3}\\ -\frac{2\cdot 2^n}{3}+\frac{2\cdot 5^n}{3} & \frac{2^n}{3}+\frac{2\cdot 5^n}{3}\end{pmatrix}$

例題 1 の解答と同じにするには, `factor` を使います．

- `factor(%)` >> $\dfrac{\begin{pmatrix}2\cdot 2^n+5^n & -2^n+5^n\\ -2\cdot 2^n+2\cdot 5^n & 2^n+2\cdot 5^n\end{pmatrix}}{3}$

確かに，同じになりました．こんどは $\alpha = \beta$ となる場合をやって見ます．

例題 2

$A = \begin{pmatrix}1 & -1\\ 4 & 5\end{pmatrix}$ とするとき, A^n (n は自然数) を求めよ．

これを公式 (**) を使って MuPAD でやってみます．

- `A := matrix([[1,-1],[4,5]])` >> $\begin{pmatrix}1 & -1\\ 4 & 5\end{pmatrix}$ ⋯ A の定義
- `e := matrix([[1,0],[0,1]])` >> $\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$ ⋯ E の定義
- `linalg :: eigenvalues(A)` >> {3} ⋯ これが固有値

そこで，$\alpha = 3$ として (**) に代入すると，

- `n * 3^(n - 1) * A + (1 - n) * 3^n * e` >> $\begin{pmatrix}3^{n-1}\cdot n - 3^n\cdot(n-1) & -3^{n-1}\cdot n\\ 4\cdot 3^{n-1}\cdot n & 5\cdot 3^{n-1}\cdot n - 3^n\cdot(n-1)\end{pmatrix}$
- `Simplify(%)` >> $\begin{pmatrix}-\frac{3^n\cdot(2\cdot n-3)}{3} & -\frac{3^n\cdot n}{3}\\ \frac{4\cdot 3^n\cdot n}{3} & \frac{3^n\cdot(2\cdot n+3)}{3}\end{pmatrix}$
- `factor(%)` >> $\dfrac{3^n}{3}\cdot\begin{pmatrix}-2\cdot n+3 & -n\\ 4\cdot n & 2\cdot n+3\end{pmatrix}$

13.2.2　行列の n 乗計算のプログラミング

MuPAD を使って A^n が求まることは解りましたが，やや面倒です．このように同じ手続きを繰り返し利用したい際に，プログラミングというのをやります．プログラミングの基礎知識に関しては, 長くなるので，ここでは省略しますが，BASIC と，そんなに変わりません．[注5)]

注5) MuPAD のプログラミングについては,「Tutorial」 の，p123 〜 p142 をご覧ください．

さて，とりあえず次のようにプログラミングすると njo_matrix(A) で A の n 乗を計算することが出来ます．入力はちょっと面倒です．1 行目には ';' をつけません．代入は'=' でなく':=' を使います．また改行するときは，Shift キーを押しながら Return キーを押さないと，Error になります．

```
•njo_matrix := proc(x)
 local a,b,e, koyuuti;
 begin
 e := matrix([[1,0],[0,1]]);
 koyuuti := linalg :: eigenvalues(x);
   if nops(koyuuti) = 2
     then a := koyuuti[1]; b := koyuuti[2];
     return(factor(simplify((a^n − b^n)/(a − b) ∗ x + (a ∗ b^n − b ∗ a^n)/(a − b) ∗ e)));
   else  a := koyuuti[1]; return(factor(simplify(n ∗ a^(n − 1) ∗ x + (1 − n) ∗ a^n ∗ e)));
   end_if
 end_proc;
```

簡単に，説明します．

```
njo_matrix := proc(x)
```

njo_matrix という名前で 1 変数の関数を定義します．次の

```
local a,b,e,koyuuti;
```

で a,b,e,koyuuti を,「ローカル変数」として宣言しておきます.「local 変数」というのは，他のプログラムから，参照できない変数という意味です．この宣言をしないと，$a, b, \cdots$ などに，値が設定されたままになります．

begin から end_proc までの間が *Body* といってプログラムの本体です．ここにはいくつかの制御文 (if 文，for 文, while 文, repeat 文などのように,「場合分け」と「繰り返し」を制御する文) と関数を書きます．

```
e := matrix([[1,0],[0,1]]);
koyuuti := linalg :: eigenvalues(x);
```

単位行列を e とし，koyuuti に固有値を代入します．

```
if nops(koyuuti) = 2 then a := koyuuti[1]; b := koyuuti[2];
return(factor(simplify((a^n − b^n)/(a − b) ∗ x + (a ∗ b^n − b ∗ a^n)/(a − b) ∗ e)));
```

これは「もし固有値が 2 つあるなら,a,b に固有値の 2 個の値を代入して A^n を表示せよ.」という意味です．このときの A^n の式は，もちろん，前節 (∗∗) の $\alpha \neq \beta$ の場合の式に代入します．なお，nops(objects) というのは number of operands の略で objects の要素の数を返します．linalg :: eigenvalues(A) は，A の固有値の集合 koyuuti を返してくるので，koyuuti[1], koyuuti[2] で，その要素を取り出してます．

```
else a := koyuuti[1]; return(factor(simplify(n ∗ a^(n − 1) ∗ x + (1 − n) ∗ a^n ∗ e)));
end_if
```

もし固有値が 1 つしかないときは，else 以下 end_if までを実行します．すなわち，a に固有値の値を代入して, 前節 (**) の $\alpha = \beta$ の場合の式に代入し，A^n を表示してます.

ここで使った制御文を「if 文」といって

if　条件　then　命令$_{真}$　else　命令$_{偽}$　end_if

「もし, 条件 が真ならば，命令$_{真}$ を実行して終わる．もし真でないならば，命令$_{偽}$ を実行して終わる」という風に使います.

これで，njo_matrix(A) とやると 2×2 行列: A の n 乗が計算できます．まずは 例題 1 の行列からです.

- A := matrix([[3, 1], [2, 4]])　　>> $\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix}$
- njo_matrix(A);　　>> $\dfrac{\begin{pmatrix} 2\cdot 2^n + 5^n & -2^n + 5^n \\ -2\cdot 2^n + 2\cdot 5^n & 2^n + 2\cdot 5^n \end{pmatrix}}{3}$

確かに同じになりました．次は 例題 2 の行列です.

- A := matrix([1, −1], [4, 5])　　>> $\begin{pmatrix} 1 & -1 \\ 4 & 5 \end{pmatrix}$
- njo_matrix(A)　　>> $\dfrac{3^n}{3}\cdot\begin{pmatrix} -2\cdot n + 3 & -n \\ 4\cdot n & 2\cdot n + 3 \end{pmatrix}$

これも，同じになりました.

このようにプログラムすると計算は楽ですが，最初に入力するのが大変です．是非，保存して再利用したいものです．MuPAD Pro の場合は，メニューバー の *File* から *New Source* をクリックし, 新しくソースファイルを開きます．そこに，プログラムの部分だけを，コピーして，それを *foo*.mu の名前 (*foo* の部分には, 適当に名前をつけます) で, 保存します．一方，これを開く時は，メニューバー の *NoteBook* から *Read* をクリックし，*foo*.file を開くと, そのファイルの内容が，MuPAD に実行され記憶されます. Light の場合は，エディターで，新しく text ファイルを開き，そこに，プログラムの部分だけを，コピーして，それを *foo*.mu の名前で, 保存します．それを読み込むには, セッション中に, コマンドを使い

•read("foo.mu")

とします．[注6)]

注6) Pro，Light ともに，コマンドを使い •write(Text, "foo.mu") とすると，text file でそのセッションの内容が，保存されます．しかし, まれにうまく行かないことがあるので，手動でコピーして，保存した方が安全だと思います．また，default では,「*foo.mu* file は, MuPAD がインストールされているフォルダの中に作られる」ので，他のフォルダーに保存したい場合は，フルパスで指定します．例えば「C:/myDocument/myMuPAD」の中に保存したい場合は,

•write(Text, "C : /myDocument/myMuPAD/ foo.mu")

「C:/myDocument/myMuPAD」の中のファイルを読むには,

•read("C : /myDocument/myMuPAD/ foo.mu")

とします．さらに詳しいことは, 第 3 章をお読みください.

13.2.3 2 次正方行列の，固有値と固有ベクトル

実は行列の n 乗の求め方は色々あります．次の例題を見てください．

例題 3

2 次の正方行列 A は，

$$A\begin{pmatrix}1\\2\end{pmatrix}=2\begin{pmatrix}1\\2\end{pmatrix}\cdots①,\quad A\begin{pmatrix}3\\4\end{pmatrix}=-2\begin{pmatrix}3\\4\end{pmatrix}\cdots②$$

をみたしている．
n を自然数とするとき，A^n を求めよ．

【解答】

① の両辺に A をかけて，

$$A^2\begin{pmatrix}1\\2\end{pmatrix}=2A\begin{pmatrix}1\\2\end{pmatrix}=2\cdot 2\begin{pmatrix}1\\2\end{pmatrix}=2^2\begin{pmatrix}1\\2\end{pmatrix}$$

さらにこの両辺に A をかけて

$$A^3\begin{pmatrix}1\\2\end{pmatrix}=2^2A\begin{pmatrix}1\\2\end{pmatrix}=2^2\cdot 2\begin{pmatrix}1\\2\end{pmatrix}=2^3\begin{pmatrix}1\\2\end{pmatrix}$$

同様にして，$n=1,2,3\cdots$ のとき

$$A^n\begin{pmatrix}1\\2\end{pmatrix}=2^n\begin{pmatrix}1\\2\end{pmatrix}=\begin{pmatrix}2^n\\2^{n+1}\end{pmatrix}\quad\cdots③$$

同様にして，② より

$$A^n\begin{pmatrix}3\\4\end{pmatrix}=(-2)^n\begin{pmatrix}3\\4\end{pmatrix}=\begin{pmatrix}3(-2)^n\\4(-2)^n\end{pmatrix}\quad\cdots④$$

③,④ をまとめて書くと

$$A^n\begin{pmatrix}1&\vdots&3\\2&\vdots&4\end{pmatrix}=\begin{pmatrix}2^n&\vdots&3(-2)^n\\2^{n+1}&\vdots&4(-2)^n\end{pmatrix}$$

$$\therefore A^n=\begin{pmatrix}2^n&3(-2)^n\\2^{n+1}&4(-2)^n\end{pmatrix}\begin{pmatrix}1&3\\2&4\end{pmatrix}^{-1}=\begin{pmatrix}2^n&3(-2)^n\\2^{n+1}&4(-2)^n\end{pmatrix}\frac{1}{2}\begin{pmatrix}-4&3\\2&-1\end{pmatrix}$$

$$=\frac{1}{2}\begin{pmatrix}-4\cdot 2^n+6(-2)^n&3\cdot 2^n-3(-2)^n\\-4\cdot 2^{n+1}+8(-2)^n&3\cdot 2^{n+1}-4(-2)^n\end{pmatrix}\quad\cdots(答)$$

一般に行列 A，列ベクトル $\vec{x}$ にたいし

$$A\vec{x}=k\vec{x},\ \text{かつ}\vec{x}\neq\vec{0}$$

が成り立つとき，k を固有値，$\vec{x}$ を A の固有値 k に対する固有ベクトルといいます．上の例では，A の

固有値は $\{2,-2\}$，　固有値 2 に対する固有ベクトルは $\begin{pmatrix}1\\2\end{pmatrix}$，　固有値 (-2) に対する固有ベクトルは $\begin{pmatrix}3\\4\end{pmatrix}$

となります．1 次独立な固有ベクトルが 2 つ求まれば, A^n はすぐに求まります.

さて，$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ の固有値と固有ベクトルを求めるには次のようにします.

いま $A\vec{x} = k\vec{x}$ とすると

$$(A - kE)\vec{x} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \cdots ①$$

ここで $(A - kE)$ が逆行列を持つとすると，それを両辺にかけて

$$\vec{x} = (A - kE)^{-1} \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

となり $\vec{x} \neq \vec{0}$ をみたさない．よって $(A - kE)$ は逆行列を持たないことが必要で

$$det(A - kE) = det \begin{pmatrix} a - k & b \\ c & d - k \end{pmatrix} = (a - k)(d - k) - bc = 0 \Longleftrightarrow k^2 - (a + d)k + (ad - bc) = 0$$

すなわち固有値 k は

$$x^2 - (a + d)x + (ad - bc) = 0 \quad \cdots ②$$

の解となることが必要です．② を固有方程式といいます．逆に k が② の解のとき，① をみたすベクトル $\vec{x}$ が少なくとも 1 つ存在することも，すぐ言えます.

では，実際に，固有値と固有ベクトルを求めてみます.

例題 4

$A = \begin{pmatrix} -10 & 6 \\ -16 & 10 \end{pmatrix}$ のとき,

$$A\vec{x} = k\vec{x} \quad \cdots ①$$

をみたす k の値と，そのときの $\vec{x}$ を求めよ.

【解答】
固有方程式は

$$x^2 - 4 = 0$$

となるので固有値は

$$k = -2, 2$$

(i) $k = -2$ のとき, ① より

$$(A - 2E)\vec{x} = \vec{0} \Longleftrightarrow \begin{pmatrix} -8 & 6 \\ -16 & 12 \end{pmatrix}\vec{x} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

よって

$$\vec{x_1} = c_1 \begin{pmatrix} 3 \\ 4 \end{pmatrix} (c_1 \neq 0)$$

(ii) $k = 2$ のとき, ① より

$$(A - 2E)\vec{x} = \vec{0} \Longleftrightarrow \begin{pmatrix} -12 & 6 \\ -16 & 8 \end{pmatrix}\vec{x} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

よって

$$\vec{x_2} = c_2 \begin{pmatrix} 1 \\ 2 \end{pmatrix} (c_2 \neq 0)$$

したがって求める解は,c_1, c_2 を任意の 0 でない定数として,

$$k = -2 \text{ のとき}, \vec{x} = c_1 \begin{pmatrix} 3 \\ 4 \end{pmatrix},\ k = 2 \text{ のとき}, \vec{x} = c_2 \begin{pmatrix} 1 \\ 2 \end{pmatrix} \quad \cdots(\text{答})$$

普通，固有ベクトルといえば c_1, c_2 をつけないで良いことになっています．したがって A の固有値は ± 2, 固有値 -2 に対する固有ベクトルは $\begin{pmatrix} 3 \\ 4 \end{pmatrix}$, 固有値 2 に対する固有ベクトルは $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$ といっても構いません．さてそれではこれを MuPAD で求めて見ましょう．次のコマンドを使います．

linalg :: charpoly(A, x)	A の固有方程式 (変数 x) を求める．
linalg :: eigenvalues(A)	A の固有値を求める．
linalg :: eigenvectors(A)	A の固有ベクトルを求める．
linalg :: jordanForm(A)	A を対角行列 (一般には *Jordan* 標準形) にする．

(linalg :: jordanForm(A, All) とすると，変換行列 P も得られます.)

それではやってみます．

- A := matrix([[−10, 6][−16, 10]]) >> $\begin{pmatrix} -10 & 6 \\ -16 & 10 \end{pmatrix}$
- linalg :: charpoly(A, x) >> $x^2 - 4$
- linalg :: eigenvalues(A) >> $\{-2, 2\}$
- linalg :: eigenvectors(A) >> $\left[\left[-2, 1, \left[\begin{pmatrix} \frac{3}{4} \\ 1 \end{pmatrix}\right]\right], \left[2, 1, \left[\begin{pmatrix} \frac{1}{2} \\ 1 \end{pmatrix}\right]\right]\right]$

最後の式は
「固有値 (−2) の重複度は 1 で, その固有ベクトルの 1 つは $\begin{pmatrix}\frac{3}{4}\\1\end{pmatrix}$, 固有値 2 の重複度は 1 で, その固有ベクトルの 1 つは $\begin{pmatrix}\frac{1}{2}\\1\end{pmatrix}$」を表します. $\begin{pmatrix}\frac{3}{4}\\1\end{pmatrix}/\!/\begin{pmatrix}3\\4\end{pmatrix}, \begin{pmatrix}\frac{1}{2}\\1\end{pmatrix}/\!/\begin{pmatrix}1\\2\end{pmatrix}$ ですから，確かに計算で求めた結果と一致します.

13.2.4　2 次正方行列の対角化

[1 次独立な固有ベクトルが 2 つあるとき]

固有値, 固有ベクトルは次のような形でも出題されます.

例題 5

$A=\begin{pmatrix}6&6\\-2&1\end{pmatrix}, P=\begin{pmatrix}a&2\\2&b\end{pmatrix}$ とし, A, P が $P^{-1}AP=\begin{pmatrix}2&0\\0&c\end{pmatrix}$ をみたすとき, 次の問いに答えよ.
(1)a, b, c の値を求めよ.
(2) A^n $(n=1,2,3,\cdots)$ を求めよ.

【解答】
(1)$P^{-1}AP=\begin{pmatrix}2&0\\0&c\end{pmatrix}$ より

$$AP=P\begin{pmatrix}2&0\\0&c\end{pmatrix} \Longleftrightarrow \begin{pmatrix}6&6\\-2&-1\end{pmatrix}\begin{pmatrix}a&2\\2&b\end{pmatrix}=\begin{pmatrix}a&2\\2&b\end{pmatrix}\begin{pmatrix}2&0\\0&c\end{pmatrix}$$

$$\Longleftrightarrow \begin{pmatrix}6a+12&12+6b\\-2a-2&-4-b\end{pmatrix}=\begin{pmatrix}2a&2c\\4&bc\end{pmatrix}$$

$$\Longleftrightarrow \begin{cases}6a+12=2a\\12+6b=2c\\-2a-2=4\\-4-b=bc\end{cases}$$

$$\Longleftrightarrow a=-3, b=-1, c=3 \quad \cdots(\text{答})$$

(2) (1) より $P=\begin{pmatrix}-3&2\\2&-1\end{pmatrix}$ とすると

$$P^{-1}AP=\begin{pmatrix}2&0\\0&3\end{pmatrix} \quad \cdots①$$

① の両辺を n 乗すると,

$$(P^{-1}AP)^n=\begin{pmatrix}2&0\\0&3\end{pmatrix}^n$$

一般に, $(P^{-1}AP)^n=P^{-1}A^nP$, また $\begin{pmatrix}2&0\\0&3\end{pmatrix}$ は対角行列だから $\begin{pmatrix}2&0\\0&3\end{pmatrix}^n=\begin{pmatrix}2^n&0\\0&3^n\end{pmatrix}$. したがって,

$$(P^{-1}AP)^n=\begin{pmatrix}2&0\\0&3\end{pmatrix}^n \Longleftrightarrow P^{-1}A^nP=\begin{pmatrix}2^n&0\\0&3^n\end{pmatrix}$$

よって

$$\begin{aligned}A^n&=P\begin{pmatrix}2^n&0\\0&3^n\end{pmatrix}P^{-1}\\&=\begin{pmatrix}-3&2\\2&-1\end{pmatrix}\begin{pmatrix}2^n&0\\0&3^n\end{pmatrix}\begin{pmatrix}1&2\\2&3\end{pmatrix}\\&=\begin{pmatrix}-3\cdot2^n+4\cdot3^n&-6\cdot2^n+6\cdot3^n\\2\cdot2^n-2\cdot3^n&4\cdot2^n-3\cdot3^n\end{pmatrix} \quad \cdots(\text{答})\end{aligned}$$

一般に, 次の定理が成り立ちます.

定理

$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ が固有値 k_1, k_2 とそれぞれに対する固有ベクトル $\overrightarrow{x_1} = \begin{pmatrix} p \\ q \end{pmatrix}, \overrightarrow{x_2} = \begin{pmatrix} r \\ s \end{pmatrix}$ をもち，$\overrightarrow{x_1}$と$\overrightarrow{x_2}$ が 1 次独立なとき，$P = \begin{pmatrix} p & r \\ q & s \end{pmatrix} = (\overrightarrow{x_1}, \overrightarrow{x_2})$ とすると，$ps - qr \neq 0$ より P^{-1} が存在し,

$$P^{-1}AP = \begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix}$$

【証明】
仮定より,

$$A\begin{pmatrix} p \\ q \end{pmatrix} = k_1\begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} pk_1 \\ qk_1 \end{pmatrix}, \text{ かつ } A\begin{pmatrix} r \\ s \end{pmatrix} = k_2\begin{pmatrix} r \\ s \end{pmatrix} = \begin{pmatrix} rk_2 \\ sk_2 \end{pmatrix}$$

まとめて書くと

$$A\begin{pmatrix} p & \vdots & r \\ q & \vdots & s \end{pmatrix} = \begin{pmatrix} pk_1 & \vdots & rk_2 \\ qk_1 & \vdots & sk_2 \end{pmatrix} = \begin{pmatrix} p & r \\ q & s \end{pmatrix}\begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix}$$

よって $P = \begin{pmatrix} p & r \\ q & s \end{pmatrix}$ とおくと

$$AP = P\begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix}$$

$$\therefore P^{-1}AP = \begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix} \quad \cdots \text{【証明終】}$$

すなわち 1 次独立な固有ベクトル $\overrightarrow{x_1} = \begin{pmatrix} p \\ q \end{pmatrix}, \overrightarrow{x_2} = \begin{pmatrix} r \\ s \end{pmatrix}$ があるとき，$P = \begin{pmatrix} p & r \\ q & s \end{pmatrix} = (\overrightarrow{x_1}, \overrightarrow{x_2})$ とすれば, $P^{-1}AP = \begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix}$ (x_1, x_2 は それぞれ $\overrightarrow{x_1}, \overrightarrow{x_2}$ に対する固有値) が成り立ちます．このように $\begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$ の形の行列に変えることを対角化といいます．MuPAD では, A の対角化は，次のようにします.

```
linalg :: jordanForm(A)
```

さっそく，やって見ましょう.

- `A := matrix([[6, 6], [-2, -1]])` >> $\begin{pmatrix} 6 & 6 \\ -2 & -1 \end{pmatrix}$
- `linalg :: eigenvectors(A)` >> $\left[\left[2, 1, \left[\begin{pmatrix} -\frac{3}{2} \\ 1 \end{pmatrix}\right]\right], \left[3, 1, \left[\begin{pmatrix} -2 \\ 1 \end{pmatrix}\right]\right]\right]$

どうやら固有ベクトルは，$\overrightarrow{x_1} = 2\begin{pmatrix} -\frac{3}{2} \\ 1 \end{pmatrix} = \begin{pmatrix} -3 \\ 2 \end{pmatrix}$ と, $\overrightarrow{x_2} = \begin{pmatrix} -2 \\ 1 \end{pmatrix}$ となりますから, P を次のように定義します.

- `P := matrix([[-3, -2], [2, 1]])` >> $\begin{pmatrix} -3 & -2 \\ 2 & 1 \end{pmatrix}$

$P^{-1}AP$ を計算してみます.

- `P^(-1) * A * P;` >> $\begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}$

確かに成分を固有値とした対角行列になりました．次はこれを一気にやってみます.

- linalg :: jordanForm(A)　　>> $\begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}$

オプションで, All を指定すると，変換行列 P' も得られます.

- linalg :: jordanForm(A, All)　　>> $\left[\begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}, \begin{pmatrix} -3 & 4 \\ 2 & -2 \end{pmatrix}\right]$

$\begin{pmatrix} 4 \\ -2 \end{pmatrix} = -2\begin{pmatrix} -2 \\ 1 \end{pmatrix} = -2\overrightarrow{x_2}$ ですから，P' も, 固有ベクトルを並べた行列になっています.

[1 次独立な固有ベクトルが１つしかないとき]

実は，1 次独立な固有ベクトルを 2 つ取れない場合でも, P を適当に選ぶと, $P^{-1}AP = \begin{pmatrix} \alpha & 1 \\ 0 & \alpha \end{pmatrix}$ の形に変形できることがわかっています．(2 次の場合の *Jordan* 標準形 といいます．)

定理

$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ が固有値 α(2 重解) をもち，かつ $A \neq \alpha E$ のときは,
α に対する固有ベクトルを $\overrightarrow{x_1}\,(= \begin{pmatrix} p \\ q \end{pmatrix})$，$(A - \alpha E)\overrightarrow{x_2} = \overrightarrow{x_1}$ をみたすベクトルを $\overrightarrow{x_2}\,(= \begin{pmatrix} r \\ s \end{pmatrix})$ としたとき

$$P = (\overrightarrow{x_1}, \overrightarrow{x_2}) = \begin{pmatrix} p & r \\ q & s \end{pmatrix} \text{とすると，} P^{-1}AP = \begin{pmatrix} \alpha & 1 \\ 0 & \alpha \end{pmatrix}$$

$A = \begin{pmatrix} 1 & -1 \\ 4 & 5 \end{pmatrix}$ のとき，MuPAD で やってみましょう.

- A := matrix([[1, −1], [4, 5]])　　>> $\begin{pmatrix} 1 & -1 \\ 4 & 5 \end{pmatrix}$
- linalg :: eigenvectors(A)　　>> $\left[\left[3, 2, \left[\begin{pmatrix} -\frac{1}{2} \\ 1 \end{pmatrix}\right]\right]\right]$
- linalg :: jordanForm(A, All)　　>> $\left[\begin{pmatrix} 3 & 1 \\ 0 & 3 \end{pmatrix}, \begin{pmatrix} -2 & 1 \\ 4 & 0 \end{pmatrix}\right]$

これは固有値が 3 (2 重解) で, それに対する固有ベクトルが $\begin{pmatrix} -\frac{1}{2} \\ 1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} -1 \\ 2 \end{pmatrix}$ であることを表しています.
そして，$P = \begin{pmatrix} -2 & 1 \\ 4 & 0 \end{pmatrix}$ とすると，$P^{-1}AP = \begin{pmatrix} \alpha & 1 \\ 0 & \alpha \end{pmatrix}$ の形になります.

- P := op(%, 2)　　>> $\begin{pmatrix} -2 & 1 \\ 4 & 0 \end{pmatrix}$
- P^(−1) ∗ A ∗ P　　>> $\begin{pmatrix} 3 & 1 \\ 0 & 3 \end{pmatrix}$

実際，$\alpha = 3$ のとき，$A - 3E = \left(\begin{smallmatrix} -2 & -1 \\ 4 & 2 \end{smallmatrix}\right)$ だから，

$$(A - 3E)\begin{pmatrix} -2 \\ 4 \end{pmatrix} = \begin{pmatrix} -2 & -1 \\ 4 & 2 \end{pmatrix}\begin{pmatrix} -2 \\ 4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

$$(A - 3E)\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -2 & -1 \\ 4 & 2 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -2 \\ 4 \end{pmatrix}$$

となり，この P は, 定理の条件をみたしています.

【参考】定理の「証明」
いま A の固有値を α (2 重解), その固有ベクトルを $\vec{x_1}$ とすると,

$$(A-\alpha E)\vec{x_1}=\vec{0} \quad \cdots ①$$

ここで

$$(A-\alpha E)\vec{x_2}=\vec{x_1} \text{ (ただし,} \vec{x_1}\text{と} \vec{x_2}\text{は 1 次独立)} \quad \cdots ②$$

をみたす ベクトル $\vec{x_2}$ が見つかったとすると，② $\Longleftrightarrow A\vec{x_2}=\vec{x_1}+\alpha\vec{x_2}$ であるから，$P=(\vec{x_1},\vec{x_2})$ とすると

$$AP=A(\vec{x_1},\vec{x_2})=(A\vec{x_1},A\vec{x_2})=(\alpha\vec{x_1},\vec{x_1}+\alpha\vec{x_2})=(\vec{x_1},\vec{x_2})\begin{pmatrix}\alpha & 1\\ 0 & \alpha\end{pmatrix}=P\begin{pmatrix}\alpha & 1\\ 0 & \alpha\end{pmatrix}$$

$\vec{x_1}$と$\vec{x_2}$ が 1 次独立のとき，P^{-1} は存在するから

$$P^{-1}AP=\begin{pmatrix}\alpha & 1\\ 0 & \alpha\end{pmatrix}$$

となり確かに変形された．よって，あとは② をみたす ベクトル $\vec{x_2}$ が存在することを言えばよい．まず α が重解となるので固有方程式は

$$(x-\alpha)^2=x^2-2\alpha x+\alpha^2=0$$

したがってケーリー・ハミルトンの定理より

$$A^2-2\alpha A+\alpha^2E=(A-\alpha E)^2=O \quad \cdots ③$$

仮定より，$A-\alpha E\neq O$ だから，$A-\alpha E=B$ とおくと，①,②,③ より

$B^2=O, B\neq O$ のとき,

$$\begin{cases} B\vec{x_1}=\vec{0} \\ B\vec{x_2}=\vec{x_1} \end{cases} \text{（ただし,} \vec{x_1}\text{と} \vec{x_2}\text{は 1 次独立)} \quad \cdots ④$$

をみたす $\vec{x_1},\vec{x_2}$ が存在することを証明すればよい．まず，$B^2=O$ より $det(B)=0$ だから $B\vec{x_1}=\vec{0}$ をみたす $\vec{x_1}(\vec{x_1}\neq\vec{0})$ は存在する．つぎに，$B\neq O$ より，$\vec{x_1}$ と 1 次独立な**あるベクトル**$\vec{x_3}$ をとって，$B\vec{x_3}=\vec{y}$ $(\vec{y}\neq\vec{0})$ とすることができる．このとき， $B^2=O$ より

$$B\vec{y}=B^2\vec{x_3}=\vec{0}$$

ところが $B\vec{x_1}=\vec{0}$ でもあるから，$\vec{y}$と$\vec{x_1}$ が 1 次独立とすると,$B=O$ となり矛盾．よって $\vec{y}=k\vec{x_1}$ $(k\neq 0)$. ゆえに, $\vec{x_2}=\frac{1}{k}\vec{x_3}$ とすると,

$$B\vec{x_2}=\frac{1}{k}B\vec{x_3}=\frac{1}{k}\vec{y}=\frac{1}{k}\cdot k\vec{x_1}=\vec{x_1}$$

ここで，$\vec{x_1}$と$\vec{x_3}$ は 1 次独立なので，$\vec{x_1}$と$\vec{x_2}$ も 1 次独立である．よって定理は証明された．

注7)

注7) 一般に 2×2 行列 B に対し $B\vec{x_1}=\vec{0}$かつ $B\vec{x_2}=\vec{0}$ ならば，$B(\vec{x_1},\vec{x_2})=\begin{pmatrix}0&0\\0&0\end{pmatrix}$. ここで，さらに $\vec{x_1}$と$\vec{x_2}$ が 1 次独立ならば，$(\vec{x_1},\vec{x_2})^{-1}$ が存在するので，$B=O$ となります.

13.3 その他の linalg のコマンド

linalg(library of linear algebra) に含まれるコマンドは, 全部で 80 種類ほどもあるので, ほんの一部のみを紹介します.

行列 A に関するコマンド (一部)

A の第 i 行を取り出す	linalg :: row(A, i)
A の第 j 列を取り出す	linalg :: col(A, j)
A の第 j 行の a 倍を，第 i 行に加える	linalg :: addRow(A, j, i, a)
A の第 j 列の a 倍を，第 i 列に加える	linalg :: addCol(A, j, i, a)
A の第 i 行を a 倍する	linalg :: multRow(A, i, a)
A の第 i 列を a 倍する	linalg :: multCol(A, i, a)
A の第 i 行を,$\vec{u}$で置き換える	linalg :: setRow(A, i, $\vec{u}$)
A の第 i 列を,$\vec{v}$で置き換える	linalg :: setCol(A, i, $\vec{v}$)
A の第 i 行と，第 j 行を交換する	linalg :: swapRow(A, i, j)
A の第 i 列と，第 j 列を交換する	linalg :: swapCol(A, i, j)
A の行の数	nrows(A)
A の列の数	ncols(A)
A の行と列の数	matdim(A)
A の零でない要素の数	nonZeros(A)
A の転置行列	linalg :: transpose(A)
A の LU 分解	linalg :: factorLU(A)
A の QR 分解	linalg :: factorQR(A)
$Gauss$ の消去法	linalg :: gaussJordan(A)
A の階数 ($rank$)	linalg :: rank(A)
A のトレース ($trace$)	linalg :: tr(A)
A の行列式	linalg :: det(A)
A の最小多項式 (変数 x のとき)	linalg :: minpoly(A, x)
A の固有方程式 (変数 x のとき)	linalg :: charpoly(A, x)
A の固有値	linalg :: eigenvalues(A)
A の固有ベクトル	linalg :: eigenvectors(A)
ジョルダン標準形 (A の対角化)	linalg :: jordanForm(A)
ジョルダン標準形 (変換行列も表示)	linalg :: jordanForm(A, All)

線形代数

S の基底	linalg :: basis(S)
$S_1, S_2, \cdots$ の共通集合の基底	linalg :: intBasis($S_1, S_2, \cdots$)
$S_1, S_2, \cdots$ の和集合の基底	linalg :: sumBasis($S_1, S_2, \cdots$)
S の基底の直交化 (*Gram* − *Schmidt* の直交化)	linalg :: orthog(S)

ベクトルの基本的な演算

$\vec{u} \cdot \vec{v}$	linalg :: scalarProduct($\vec{u}, \vec{v}$)
$\vec{u} \times \vec{v}$	linalg :: crossProduct($\vec{u}, \vec{v}$)
$\vec{u}$ と $\vec{v}$ のなす角	linalg :: angle($\vec{u}, \vec{v}$)
$\vec{v}$ の正規化 (長さ 1)	linalg :: normalize($\vec{v}$)
$\vec{v}$ の要素の数	linalg :: vecdim($\vec{v}$)

ベクトル解析

$\nabla \cdot \vec{v}$	linalg :: divergence($\vec{v}$, [x, y, z])
$\nabla \times \vec{v}$	linalg :: curl($\vec{v}$, [x, y, z])
∇f	linalg :: grad(f, [x, y, z])
Δf	linalg :: laplacian(f, [x, y, z])
$\vec{v}$ のスカラーポテンシャル	linalg :: potential($\vec{v}$, [x, y, z])
$\vec{v}$ のベクトルポテンシャル	linalg :: vectorPotential($\vec{v}$, [x, y, z])
f のヘッセ行列	linalg :: hessian(f, [x, y, z])
$\vec{v}$ のヤコビ行列	linalg :: jacobian($\vec{v}$, [x, y, z])

上の表で，$\vec{u}, \vec{v}$ はベクトル，f はスカラー を表します．このとき，行ベクトルや列ベクトルが混ざっていても大丈夫です．$S_1, S_2, \cdots$ は，同次元のベクトルの, リストまたは集合．すなわち，$[\vec{v_1}, \vec{v_2}, \cdots]$ または $\{\vec{v_1}, \vec{v_2}, \cdots\}$ です．また，簡単のため，ベクトル解析では, 3 次元のベクトルを例にしましたが, crossProduct, curl, vectorPotential を除き, 何次元でも大丈夫です．[注8)]
少しだけ例を挙げます.

注8) ベクトル解析では, 極座標, 円柱座標なども使えますが，このときは option で，scaling factor を指定しないといけません．scaling factor は自分で作成することもできますが，よく使われる場合は，簡単に言葉で指定できます．それには，

Spherical(球面座標), *Cylindrical*(円柱座標), *EllipticCylindrical*, *ParabolicCylindrical*, *Torus*, *Cartesian*

があります．例えば，球面座標, 円柱座標での発散は次のようになります.

[球面座標]linalg :: divergence($\vec{v}$, [r, theta, phi], Spherical)

[円柱座標]linalg :: divergence($\vec{v}$, [r, phi, z], Cylindrical)

先ず 3 次の行列の演算です.

•A := matrix([[1, 1, −2], [−1, 3, −2], [2, −2, 6]]) >> $\begin{pmatrix} 1 & 1 & -2 \\ -1 & 3 & -2 \\ 2 & -2 & 6 \end{pmatrix}$

•linalg :: rank(A) >> 3

•linalg :: charpoly(A, x) >> $x^3 + 10 \cdot x^2 + 28 \cdot x - 24$

•factor(%) >> $(x-6) \cdot (x-2)^2$

•linalg :: minpoly(A, x) >> $x^2 - 8 \cdot x + 12$

•linalg :: eigenvalues(A) >> {2, 6}

•linalg :: eigenvectors(A) >> $\left[\left[2, 2, \left[\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}\right]\right], \left[6, 1, \left[\begin{pmatrix} -\frac{1}{2} \\ -\frac{1}{2} \\ 1 \end{pmatrix}\right]\right]\right]$

•linalg :: jordanForm(A, All) >> $\left[\begin{pmatrix} 2 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 2 \end{pmatrix}, \begin{pmatrix} \frac{4}{5} & -\frac{1}{4} & -2 \\ \frac{1}{4} & -\frac{1}{4} & 0 \\ -\frac{1}{2} & \frac{1}{2} & 1 \end{pmatrix}\right]$

次は, 内積と外積です.

•u := matrix([a, b, c]) >> $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$

•v := matrix([1, 2, 3]) >> $\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$

•linalg :: scalarProduct(u, v) >> $a + 2 \cdot b + 3 \cdot c$

•linalg :: crossProduct(u, v) >> $\begin{pmatrix} 3 \cdot b - 2 \cdot c \\ -3 \cdot a + c \\ 2 \cdot a - b \end{pmatrix}$

次は，ベクトル解析です.

•v := matrix([x, y^2, z^3]) >> $\begin{pmatrix} x \\ y^2 \\ z^3 \end{pmatrix}$

•linalg :: divergence(v, [x, y, z]) >> $2 \cdot y + 3 \cdot z^2 + 1$

linalg :: divergence(v, [x1, x2, x3]) で，$\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3}$ を表すので，通常は，$[x_1, x_2, x_3] = [x, y, z]$ にします．以下の，ベクトル解析のコマンドも同様です．

- linalg :: curl(v, [x, y, z])　　>> $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
- f := x + y^2 + z^3　　>> $x + y^2 + z^3$
- linalg :: grad(f, [x, y, z])　　>> $\begin{pmatrix} 1 \\ 2 \cdot y \\ 3 \cdot z^2 \end{pmatrix}$
- linalg :: laplacian(f, [x, y, z])　　>> $6 \cdot z + 2$

最後は, 線形代数です.

- v1 := matrix([1, 2]); v2 := matrix([3, 1]); v3 := matrix([1, 0])　　>> $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$　>> $\begin{pmatrix} 3 \\ 1 \end{pmatrix}$　>> $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
- linalg :: basis([v1, v2, v3])　　>> $\left[\begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 3 \\ 1 \end{pmatrix}\right]$
- linalg :: orthog([v1, v2, v3])　　>> $\left[\begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 2 \\ -1 \end{pmatrix}\right]$

第14章　微分方程式,漸化式,数列の和

14.1　微分方程式

微分方程式の定義

$f(x)$ の微分方程式	ode(方程式, f(x))
$f(x)$ の微分方程式 (初期条件あり)	ode({ 方程式, 初期条件 }, f(x))
$f(x), g(x), \cdots$ の連立微分方程式	ode({ 方程式$_1$, 方程式$_2$, $\cdots$ }, {f(x), g(x), $\cdots$})

注1) 結局，**ode の引数は常に 2 つになります**．微分は，diff(f(x), x), D(f), y′ が全て使えます.

例 1(変数分離形)

(1) $x\frac{dy}{dx} = -y$ の一般解を求めよ.
(2) $x\frac{dy}{dx} = -y$ を, 初期条件 ($x = 1$ のとき $y = 2$) のもとで解け.

先ず, 微分方程式を定義します.

- ode1 := ode(x ∗ f′(x) = −f(x), f(x))　　>> $\mathrm{ode}\left(f(x) + x \cdot \frac{\partial}{\partial x} f(x), f(x)\right)$

$f(x)$ のかわりに，y を使ってもよいですが，$x * y'(x) = -y$ というミスが起こりやすいです．正しくは, $x * y'(x) = -y(x)$ のように，x を入れます．さて，これを解くのは，solve するだけです.

- solve(ode1)　　>> $\left\{\frac{C51}{x}\right\}$

一般解が $f(x) = \frac{C}{x}$ ということです．"51" は，MuPAD が勝手につけてくる番号です． 次は, 初期条件として，($x = 1$ のとき $y = 2$) をつけます．今度は y でやってみます．x を付け忘れないようにします．また, 初期条件と方程式は { } で,1 つにまとめます.

- ode({x ∗ y′(x) + y(x), y(1) = 2}, y(x))　　>> $\mathrm{ode}\left(\left\{y(x) + x \cdot \frac{\partial}{\partial x} y(x), y(1) = 2\right\}, y(x)\right)$
- solve(%)　　>> $\left\{\frac{2}{x}\right\}$

$y = \frac{2}{x}$ が特殊解ということです．注2) また,「x ∗ y′(x) + y(x) = 0」 の 「0」は省略できます.

注1) ode は, ordinary differential equation の略です.
注2) 検算してみます．与式を x, y に関して分離して,

$$\frac{1}{y}dy = -\frac{1}{x}dx$$

両辺を積分して

$$\therefore \int \frac{1}{y}dy = -\int \frac{1}{x}dx \iff \log|y| = -\log|x| + C_1 \iff |y| = \frac{e^{C_1}}{x} \iff y = \pm\frac{e^{C_1}}{x}$$

いくつか 例を挙げます.

例 2 (1 階線形)

$x\frac{dy}{dx} + y = \log x$　($x = 1$ のとき $y = 0$) を解け.

•ode2 := ode({x ∗ y′(x) + y(x) = ln(x), y(1) = 0}, y(x))

>> ode$\left(\left\{y(1) = 0, -\mathtt{ln}(x) + y(x) + x \cdot \frac{\partial}{\partial x}y(x)\right\}, y(x)\right)$

•solve(ode2)　　>> $\left\{\frac{x \cdot \mathtt{ln}(x) - x + 1}{x}\right\}$

例 3 (2 階線形)

(1)$y'' + 4y' + 13y = 0$ の一般解を求めよ.
(2)$y'' + 4y' + 13y = 0$ を初期条件 ($x = 0$ のとき $y = 0$, $x = \frac{\pi}{2}$のとき $y = -1$) のもとで解け.

y''は, $y''(x)$ と入力します.

•ode(y′′(x) − 4 ∗ y′(x) + 13 ∗ y(x), y(x))　　>> ode$\left(13 \cdot y(x) - 4 \cdot \frac{\partial}{\partial x}y(x) + \frac{\partial^2}{\partial x^2}y(x),\ y(x)\right)$

•solve(%)　　>> $C72 \cdot \cos(3 \cdot x) \cdot e^{2 \cdot x} - C73 \cdot e^{2 \cdot x} \cdot \sin(3 \cdot x)$

初期条件が 2 つ以上あるときも，そのまま, 方程式と一緒に，{ } の中に並べるだけです.

•ode({y′′(x) − 4 ∗ y′(x) + 13 ∗ y(x), y(0) = 0, y(PI/2) = −1}, y(x))

>> ode$\left(\left\{y\left(\frac{\pi}{2}\right) = -1,\ y(0) = 0,\ 13 \cdot y(x) - 4 \cdot \frac{\partial}{\partial x}y(x) + \frac{\partial^2}{\partial x^2}y(x)\right\},\ y(x)\right)$

•solve(%)　　>> $\left\{\frac{(e^x)^2 \cdot \sin(3 \cdot x)}{e^{\pi}}\right\}$

例 4 (連立方程式)

t の関数 $x(t), y(t)$ が次の方程式をみたしている.

$$\begin{cases} \frac{dx(t)}{dt} &= x(t) + y(t) \\ \frac{dy(t)}{dt} &= -2x(t) + 4y(t) \end{cases} \quad \cdots (*)$$

(∗) を初期条件 ($x(0) = 1, y(0) = -1$) のもとで解け.

方程式と初期条件，関数はそれぞれまとめて 1 つにします．すなわち, ode **の引数は，常に 2 つです．**

•ode({x′(t) = x(t) + y(t), y′(t) = −2 ∗ x(t) + 4 ∗ y(t), x(0) = 1, y(0) = −1}, {x(t), y(t)})

>> ode$\left(\left\{x(0) = 1, y(0) = -1, -x(t) - y(t) + \frac{\partial}{\partial t}x(t), 2 \cdot x(t) - 4 \cdot y(t) + \frac{\partial}{\partial t}y(t)\right\}, \{x(t), y(t)\}\right)$

•solve(%)　　>> $\{[x(t) = 3 \cdot e^{2 \cdot t} - 2 \cdot e^{3 \cdot t}, y(t) = 3 \cdot e^{2 \cdot t} - 4 \cdot e^{3 \cdot t}]\}$

14.2 漸化式

漸化式の定義

a_nの漸化式	rec(方程式, a(n))
a_nの漸化式 (初期条件あり)	rec(方程式, a(n), 初期条件)

注3) **微分方程式と異なり，初期条件をつけるときは，それが 3 つ目の引数になります．**

例 1 (2 項間漸化式)

$$a_{n+1} = 2a_n - 1 \ (n = 1, 2, 3 \cdots), \ a_1 = 3$$

で定まる数列の一般項を求めよ.

a_n は，$a(n)$ のように書きます.

- rec1 := rec(a(n + 1) = 2 ∗ a(n) − 1, a(n), a(1) = 3) >> rec($a(n+1) - 2 \cdot a(n) + 1, a(n), \{a(1) = 3\}$)

解くのは，solve を使うだけです.

- solve(rec1) >> $\{2^n + 1\}$

ここで初期条件を省略してみます.

- rec2 := rec(a(n + 1) = 2 ∗ a(n) − 1, a(n)) >> rec($a(n+1) - 2 \cdot a(n) + 1, a(n), \phi$)
- solve(rec2) >> $\{2^n \cdot C95 + 1\}$

これが一般解です．漸化式の一般解は, 余り聞かないので, 漸化式を手動で解いてみます.

$$a_{n+1} = 2a_n - 1$$

$\alpha = 2\alpha - 1$ の解は $\alpha = 1$ だから，両辺から 1 を引くと

$$a_{n+1} - 1 = 2(a_n - 1)$$

よって，$\{a_n - 1\}$ は, 初項 $(a_1 - 1)$, 公比 2 の等比数列となるので

$$a_n - 1 = (a_1 - 1)2^{n-1} \Longleftrightarrow a_n = (a_1 - 1)2^{n-1} + 1 = \frac{a_1 - 1}{2}2^n + 1$$

$\frac{a_1-1}{2} = C$ とおくと，$a_n = C \cdot 2^n + 1$ となる．これが一般解です．a_1 の値によらず，a_n は，この形をしています.

注3) rec は, recurrence equation の略です．MuPAD は，線形の漸化式しか解けません.
初項に文字を使ったり，a_0 や a_2 を初項にすることもできます．さらに初項を省略すると一般解が求まります.
漸化式は $a(n + 1) = 2 * a(n)$ の代わりに，$a(n + 2) = 2 * a(n + 1), a(n + 3) = 2 * a(n + 2) \cdots$ のように書いても良いですが，真ん中の $a(n)$ は $a(n + 1)$ などとはできません．また，n の代わりに k, i など別の文字を使っても大丈夫です.

例 2 (3 項間漸化式)

$$a_{n+1} - 3a_{n+1} + 2a_n = 0\ (n = 1, 2, 3\cdots),\ a_1 = 1, a_2 = 2$$

で定まる数列の一般項を求めよ.

先ず一般解を求めて見ます.「$a_{n+1} - 3a_{n+1} + 2a_n = 0$」 の「0」は省略できます.

•reset()　　>>

•rec(a(n + 2) − 3 ∗ a(n + 1) + 2 ∗ a(n), a(n))

>> rec($2 \cdot a(n) - 3 \cdot a(n+1) + a(n+2), a(n)$)

•solve(%)　　>> $\{C1 + 2^n \cdot C2\}$

最初に reset をしたので，定数の名前が，$C1, C2$ になりました．さて，次は，初期条件を加えて解いてみます．初期条件が 2 つ以上あるときは, { } で，1 つにまとめます.

•rec(a(n + 2) − 3 ∗ a(n + 1) + 2 ∗ a(n), a(n), {a(1) = 1, a(2) = 2})

>> rec($2 \cdot a(n) - 3 \cdot a(n+1) + a(n+2), a(n), \{a(1) = 1, a(2) = 2\}$)

•solve(%)　　>> $\left\{\frac{2^n}{2}\right\}$

しかし，残念ながら，MuPAD は, 漸化式は非常に簡単なものしか解けません．$a_{n+1} = 2a_n + n$ を解いてみましょう.

•rec(a(n + 1) = 2 ∗ a(n) + n, a(n), a(1) = 1)　　>> rec($a(n+1) - 2 \cdot a(n) - n, a(n), \{a(1) = 1\}$)

•solve(%)　　>> solve(rec($a(n+1) - 2 \cdot a(n) - n, a(n), \{a(1) = 1\}$))

このように全く解けません．[注4)] その他，$a_{n+1} = 2a_n + 2^n$, $a_{n+1} = \frac{a_n}{a_n+3}$ なども解けません．また連立漸化式は, 定義すらできません.

[注4)] そのうえ，初項を省略すると, 誤った一般解も出してきます.

•rec(a(n + 1) = 2 ∗ a(n) + n, a(n))　　>> rec($a(n+1) - 2 \cdot a(n) - n, a(n), \phi$)

•solve(%)　　>> $\{-n - 1\}$

14.3 数列の和

sum(a(k), k = 1..n)	$\sum_{k=1}^{n} a_k$
sum(a(k), k)	$f(k+1)-f(k)=a(k)$ となる $f(k)$

14.3.1 数列の和，階差数列の利用

今度は，数列の和を調べます．sum(a(k), k = 1..n) は，$\sum_{k=1}^{n} a_k$ を表します．

$\sum_{k=1}^{3} k^2$ は?

•sum(k^2, k = 1..3) >> 14

注5)

$\sum_{k=1}^{n} k^2$ は?

•sum(k^2, k = 1..n) >> $\dfrac{n\cdot(2\cdot n+1)\cdot(n+1)}{6}$

sum(a(k), k) は，$f(k+1)-f(k)=a(k)$ となる $f(k)$ を求めます．

•sum(k^2, k) >> $\dfrac{k}{6}-\dfrac{k^2}{2}+\dfrac{k^3}{3}$

$f(k)=\frac{k}{6}-\frac{k^2}{2}+\frac{k^3}{3}$ とおき，確かめてみます．

$$f(k+1)-f(k)=\frac{(k+1)-k}{6}-\frac{(k+1)^2-k^2}{2}+\frac{(k+1)^3-k^3}{3}=\frac{1}{6}-\frac{2k+1}{2}+\frac{3k^2+3k+1}{3}=k^2$$

確かに，$f(k+1)-f(k)=k^2$ となっています．このような $f(k)$ が求まると，和はすぐ求まります．

一般に，$a(k)=f(k+1)-f(k)$ とかけるとき，

$$\sum_{k=1}^{n} a(k)=\sum_{k=1}^{n}\{f(k+1)-f(k)\}$$
$$=\{\cancel{f(2)}-f(1)\}+\{\cancel{f(3)}-\cancel{f(2)}\}+\{\cancel{f(4)}-\cancel{f(3)}\}+\cdots+\{f(n+1)-\cancel{f(n)}\}=f(n+1)-f(1)$$

実際，$f(k)=\frac{k}{6}-\frac{k^2}{2}+\frac{k^3}{3}$, $k^2=f(k+1)-f(k)$ だから

$$\sum_{k=1}^{n} k^2=f(n+1)-f(1)=\left\{\frac{n+1}{6}-\frac{(n+1)^2}{2}+\frac{(n+1)^3}{3}\right\}-\left(\frac{1}{6}-\frac{1}{2}+\frac{1}{3}\right)$$
$$=\frac{n+1}{6}\left(1-3(n+1)+2(n+1)^2\right)=\frac{n+1}{6}(2n^2+n)=\frac{n(n+1)(2n+1)}{6}$$

注5) 有限個の和の場合は，列生成子「$」と _plus を使ってもできます．

•_plus(k^2 $k = 1..3) >> 14

今度は，$f(n+1)-f(n)=3^n$ となる $f(n)$ を求めてみます.

$$\bullet\texttt{sum(3\^{}n,n)} \qquad >> \left\{\frac{3^n}{2}\right\}$$

実際，$\frac{3^{n+1}}{2}-\frac{3^n}{2}=\frac{3\cdot3^n-3^n}{2}=3^n$ が成り立ちます．よって，

$$\sum_{k=1}^{n}3^k=\sum_{k=1}^{n}\left(\frac{3^{k+1}}{2}-\frac{3^k}{2}\right)=\left(\frac{3^2}{2}-\frac{3^1}{2}\right)+\left(\frac{3^3}{2}-\frac{3^2}{2}\right)+\cdots+\left(\frac{3^{n+1}}{2}-\frac{3^n}{2}\right)=\frac{3^{n+1}}{2}-\frac{3}{2}$$

これは, 初項 3, 公比 3, 項数 n の等比数列の和の公式: $S_n=\frac{a(r^n-1)}{r-1}=\frac{3(3^n-1)}{3-1}$ と一致します. 次は, 分数の和: $\displaystyle\sum_{k=1}^{n}\frac{1}{4k^2-1}$ を求めてみます.

$$\bullet\texttt{sum(1/(4*k\^{}2-1),k)} \qquad >> -\frac{1}{2\cdot(2\cdot k-1)}$$
$$\bullet\texttt{sum(1/(4*k\^{}2-1),k=1..n)} \qquad >> \frac{n}{2\cdot n+1}$$

実際，$f(k)=-\frac{1}{2(2k-1)}$ とおくと，$f(k+1)-f(k)=-\frac{1}{2(2k+1)}+\frac{1}{2(2k-1)}=\frac{1}{4k^2-1}$ ですから

$$\begin{aligned}&\sum_{k=1}^{n}\frac{1}{4k^2-1}=\frac{1}{2}\sum_{k=1}^{n}\left(\frac{1}{2k-1}-\frac{1}{2k+1}\right)\\&=\frac{1}{2}\left\{\left(\frac{1}{1}-\frac{1}{3}\right)+\left(\frac{1}{3}-\frac{1}{5}\right)+\cdots+\left(\frac{1}{2n-1}-\frac{1}{2n+1}\right)\right\}=\frac{1}{2}\left(1-\frac{1}{2n+1}\right)=\frac{n}{2n+1}\end{aligned}$$

14.3.2　sum と int の関係

sum と int には，密接な関係があります．まず，sum(a(k),k) と sum(a(k),k=1..n) の関係です.

$$\begin{cases}\texttt{sum(a(k),k)=F(k)} & \Longleftrightarrow \quad F(k+1)-F(k)=a(k)\\ F(k+1)-F(k)=a(k) & \longrightarrow \quad \texttt{sum(a(k),k=1..n)}=\displaystyle\sum_{k=1}^{n}a(k)=F(n+1)-F(1)\end{cases}$$

一方, 積分にも似た関係があります.

$$\begin{cases}\texttt{int(f(x),x)=F(x)} & \Longleftrightarrow \quad \displaystyle\lim_{h\to0}\frac{F(x+h)-F(x)}{h}=f(x)\\ \displaystyle\lim_{h\to0}\frac{F(x+h)-F(x)}{h}=f(x) & \longrightarrow \quad \texttt{int(f(x),x=a..b)}=\int_a^b f(x)dx=F(b)-F(a)\end{cases}$$

$a(k)=F(k+1)-F(k)$ をみたす $F(k)$ は，積分における「不定積分」のような働きをしています.