

appendix A1

たとえば $y = xe^{-x}$ のグラフなら `\easydraw[x e^{(-x)}](-1.4, 5.5)`、 $y = x + \frac{1}{x}$ なら `\easydraw[x+1/x](-5.5, 5.5)`、... のように記述して描画できれば申し分ないのですが、さすがにそれは私の手に負えるものではありません（そもそも \TeX だけで可能なのか?）。そこで、グラフ 1 つにつき、以下に示す `drawfunc` 環境、`drawfuncp` 環境を用いて関数式を定義することにしました。これらの見本は、文書のプリアンブルに `\usepackage{tmtmath}` を宣言して出力したものです。tmtmath.sty ファイルは、所定のディレクトリか文書と同じディレクトリに置いてください。また `pict2e` を使うことがあるので、プリアンブルに `\usepackage[ドライバ名]{pict2e}` も宣言しておくとういでしょう。出来が今ひとつ使い勝手も良くないので、合成関数が描画できるオマケ機能と考えてください（⇒appendix A3 参照）。

[`drawfunc` 環境] ($y = f(x)$ のグラフを描画するための環境)

```
\begin{drawfunc} (#2, #3) % 補 1, 2]
\@steps=0.01\p@ % 必要に応じて記述する描画精度の値 [補 3]
\def\calcdimen{ % 関数の定義
  (ここに  $f(x)$  に相当する計算式を記述)
}\end{drawfunc}
```

x の区間—(#2, #3) の範囲—で、`\calcdimen{...}` で定義した関数を描画する。描画精度の値は記述しなくてもよい（そのときは、規定値の `0.05\p@` が使われる）。`drawpict` 環境か `picture` 環境で使用することが前提だが、文章中で使用しても問題ない。

`\calcdimen{...}` で関数を定義すると書いたが、`drawfunc` 環境が実際にしていることは、区間開始の x の値 #2 を変数（レジスタ）である `\dimen@` に代入し、計算された y の値を再び `\dimen@` に代入するというものである。このため、 y の値を計算するには少々の工夫と注意が要る。この計算を `\@steps` 刻みで #3 まで繰り返し、`\qbezier` でつなぐことで関数のグラフが描画される。

つまり、`drawfunc` 環境で区間の数値が与えられれば、座標 (x, y) が必ず計算され線がつながれることになる。そのため、関数を定義しないまま

```
\begin{drawfunc}(-5, 10)%
\def\calcdimen{
}\end{drawfunc}
```

のように使っても、 $y = x$ の線が描画される。それは、 $(-5, -5)$, $(-4.95, -4.95)$, ..., $(10, 10)$ がつながった結果である。

計算式については追って説明するが、たとえば

```
\begin{drawfunc}(-5, 6)%
\def\calcdimen{
  \dimenmul(\strip@pt\dimen@, \strip@pt\dimen@)
}\end{drawfunc}
```

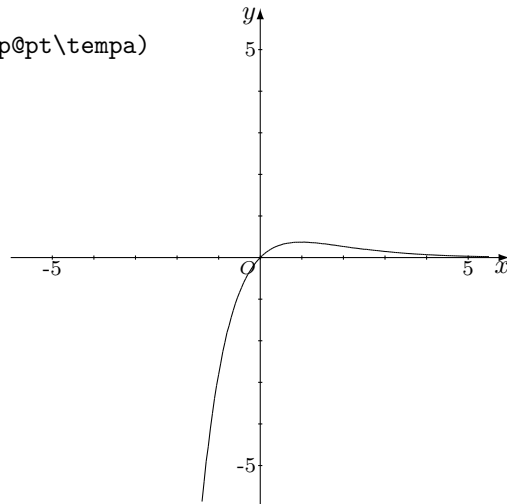
のようにすると、このように $y = x^2$ の線が描画される。それは、 $(-5, 25)$, $(-4.95, 24.5025)$, ..., $(6, 36)$ がつながった結果である。

ちなみに、文章中の描画単位は `pt` である。

▽ $y = xe^{-x}$ のグラフを x - y 実数平面に描画

```
\begin{drawpict}[.55cm](12, 12)(-6, -6)
  \baseskip6
  \coordinate[R](-6, 6)(-6, 6)

  \begin{drawfunc}(-1.4, 5.5)%
  \def\calcdimen{ %  $xe^{-x}$  を表す計算式
    \tempa=\dimen@
    \dimenmul(\strip@pt\dimen@, -1)
    \dimenexp
    \dimenmul(\strip@pt\dimen@, \strip@pt\tempa)
  }\end{drawfunc}
\end{drawpict}
```



xe^{-x} の計算 描画範囲は-1.4 から 5.5

```
\tempa=\dimen@
```

>> $\dimen@$ が x の値である。計算中に $\dimen@$ の値が変わるので臨時変数 \tempa に退避させる [補 4]。

```
\dimenmul(\strip@pt\dimen@, -1)
```

>> $\dimen@$ (すなわち x) を -1 倍する。この時点で $\dimen@ (x)$ の値は $-x$ になる。

```
\dimenexp
```

>> e^x の計算 (いまの場合、 x は $-x$ である)。これにより計算結果が $\dimen@$ に代入される。

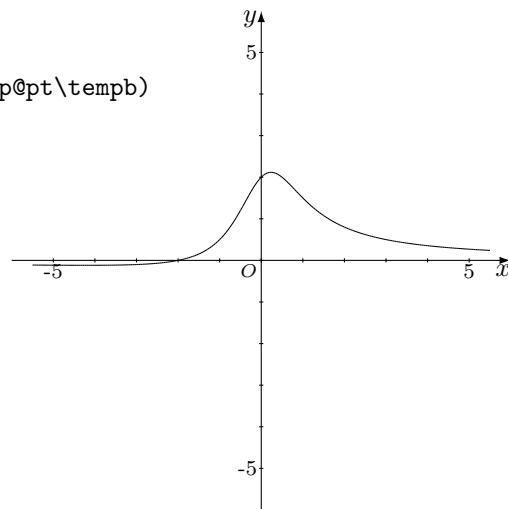
```
\dimenmul(\strip@pt\dimen@, \strip@pt\tempa)
```

>> $\dimen@$ を \tempa 倍する (すなわち正規の x の値を掛ける)。これで $\dimen@$ の値が y の値となる。

$\nabla y = \frac{x+2}{x^2+1}$ のグラフを x - y 実数平面に描画

```
\begin{drawpict}[.55cm](12, 12)(-6, -6)
  \baseskip6
  \coordinate[R](-6, 6)(-6, 6)

  \begin{drawfunc}(-5.5, 5.5)%
  \def\calcdimen{ % (x+2)/(x^2+1) を表す計算式
    \tempa=\dimen@
    \dimenmul(\strip@pt\dimen@, \strip@pt\dimen@)
    \dimensum(\strip@pt\dimen@, 1)
    \tempb=\dimen@
    \dimensum(\strip@pt\tempa, 2)
    \dimendiv(\strip@pt\dimen@, \strip@pt\tempb)
  }\end{drawfunc}
\end{drawpict}
```



$y = \frac{x+2}{x^2+1}$ の計算式	描画範囲は-5.5 から 5.5
------------------------------	------------------

```
\tempa=\dimen@
```

>> $\dimen@$ が x の値である。計算中に $\dimen@$ の値が変わるので臨時変数 \tempa に退避させる。

```
\dimenmul(\strip@pt\dimen@, \strip@pt\dimen@)
```

>> 分母の x^2 の計算。 $\dimen@$ に $\dimen@$ を掛ける。この時点で $\dimen@ (x)$ の値は x^2 に変わる。

```
\dimensum(\strip@pt\dimen@, 1)
```

>> x^2 に 1 を加える。これで分母の値 $x^2 + 1$ が $\dimen@$ に代入される。

```
\tempb=\dimen@
```

>> 分子の計算中に $\dimen@$ の値が変わるので臨時変数 \tempb に退避させる。

```
\dimensum(\strip@pt\tempa, 2)
```

>> 正規の x (退避させた \tempa) に 2 を加える。この時点で $\dimen@$ の値は $x+2$ である。

```
\dimendiv(\strip@pt\dimen@, \strip@pt\tempb)
```

>> $\dimen@$ を \tempb で割る (すなわち $(x+2)/(x^2+1)$)。これで $\dimen@$ の値が y の値となる。

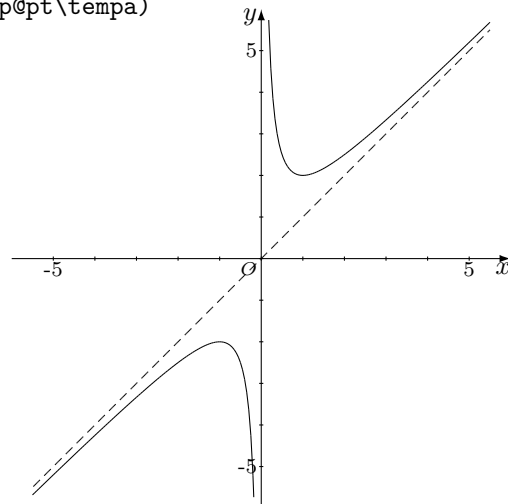
▽ $y = x + \frac{1}{x}$ のグラフと漸近線 $y = x$ を x - y 実数平面に描画

```
\begin{drawpict}[.55cm](12, 12)(-6, -6)
  \baseskip6
  \coordinate[R](-6, 6)(-6, 6)

  \begin{drawfunc}(-5.5, -.18)% (x < -0.18)
  \def\calcdimen{ % y = x+1/x を表す計算式
    \tempa=\dimen@
    \dimendiv(1, \strip@pt\dimen@)
    \dimensum(\strip@pt\dimen@, \strip@pt\tempa)
  }\end{drawfunc}

  \begin{drawfunc}(.18, 5.5)% (x > 0.18)
  \def\calcdimen{ % y = x+1/x を表す計算式
    \tempa=\dimen@
    \dimendiv(1, \strip@pt\dimen@)
    \dimensum(\strip@pt\dimen@, \strip@pt\tempa)
  }\end{drawfunc}

  \abline[1, 0]:(-5.5, 5.5) % 漸近線
\end{drawpict}
```



$y = x + \frac{1}{x}$ の計算式 描画範囲は-5.5 から-0.18 と 0.18 から 5.5

```
\tempa=\dimen@
```

>> $\dimen@$ が x の値である。計算中に $\dimen@$ の値が変わるので臨時変数 \tempa に退避させる。

```
\dimendiv(1, \strip@pt\dimen@)
```

>> 1を $\dimen@$ で割る。この時点で $\dimen@$ の値は $\frac{1}{x}$ になっている。

```
\dimensum(\strip@pt\dimen@, \strip@pt\tempa)
```

>> $\dimen@$ に \tempa を加える。これで $\dimen@$ の値が y の値となる。

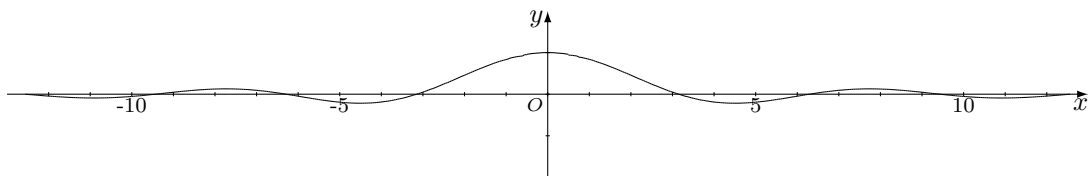
$\nabla y = \frac{\sin x}{x}$ のグラフを x - y 実数平面に描画

```
\begin{drawpict}[.55cm](26, 4)(-13, -2)
  \coordinate[R](-13, 13)(-2, 2)

  \begin{drawfunc}(-12.56, -.5)% (x < -0.5)
  \def\calcdimen{ % y = (sin x)/x を表す計算式
    \tempa=\dimen@
    \dimensin
    \dimendiv(\strip@pt\dimen@, \strip@pt\tempa)
  }\end{drawfunc}

  \qbezier(-.5, .9588)(-.2468, 1)(0, 1) % (-0.5 < x < 0.5)
  \qbezier(0, 1)(.2468, 1)(.5, .9588)

  \begin{drawfunc}(.5, 12.56)% (x > 0.5)
  \def\calcdimen{ % y = (sin x)/x を表す計算式
    \tempa=\dimen@
    \dimensin
    \dimendiv(\strip@pt\dimen@, \strip@pt\tempa)
  }\end{drawfunc}
\end{drawpict}
```



$y = \frac{\sin x}{x}$ の計算式 描画範囲は-12.56 から 12.56
(ただし-0.5 から 0.5 は\qbezierによる)

```
\tempa=\dimen@
```

>> \dimen@が x の値である。計算中に\dimen@の値が変わるので臨時変数\tempaに退避させる。

```
\dimensin
```

>> $\sin x$ の計算 [補 5]。これにより計算結果が\dimen@に代入される。

```
\dimendiv(\strip@pt\dimen@, \strip@pt\tempa)
```

>> \dimen@を\tempaで割る。これで\dimen@の値が y の値となる。

```
\qbezier(-.5, .9588)(-.2468, 1)(0, 1)
```

```
\qbezier(0, 1)(.2468, 1)(.5, .9588)
```

※ $x = 0$ 付近では線が乱れるので\qbezierで代用している。

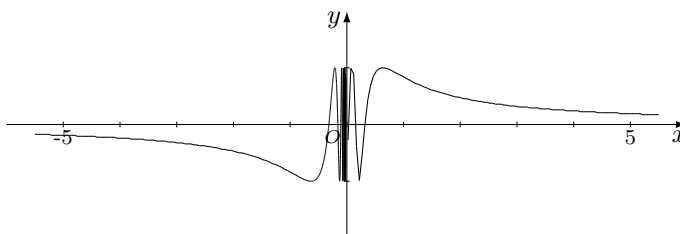
$\nabla y = \sin \frac{1}{x}$ のグラフを x - y 実数平面に描画
 (描画精度は、 $-0.6366 < x < -0.02$ の範囲だけ 0.001 に指定して描画した)

```
\begin{drawpict}[.75cm](12, 4)(-6, -2)
  \baseskip3
  \coordinate[R](-6, 6)(-2, 2)

  \begin{drawfunc}(-5.5, -.6366)% (x < -0.6366)
  \def\calcdimen{ % y = sin(1/x) を表す計算式
    \dimendiv(1, \strip@pt\dimen@)
    \dimensin
  }\end{drawfunc}

  \begin{drawfunc}(-.6366, -.02)% (x < -0.02)
  \@steps=.001\p@ % 描画精度の変更
  \def\calcdimen{ % y = sin(1/x) を表す計算式
    \dimendiv(1, \strip@pt\dimen@)
    \dimensin
  }\end{drawfunc}

  \begin{drawfunc}(.02, 5.5)% (x > 0.02)
  \def\calcdimen{ % y = sin(1/x) を表す計算式
    \dimendiv(1, \strip@pt\dimen@)
    \dimensin
  }\end{drawfunc}
\end{drawpict}
```



$y = \sin \frac{1}{x}$ の計算式 描画範囲は-5.5 から-0.02 と 0.02 から 5.5

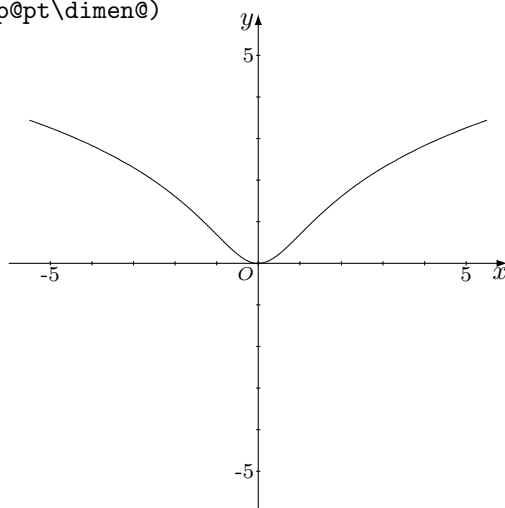
```
\@steps=.001\p@
  >> 描画精度を細かく設定。
\dimendiv(1, \strip@pt\dimen@)
  >> \dimen@が x の値である。 $\frac{1}{x}$  の計算。臨時変数に退避させる必要はない。
\dimensin
  >>  $\sin x$  の計算。これで \dimen@ の値が y の値となる。
```

※ 描画精度の違いによるグラフの形状に注意。この場合は精度が細かい方が、原点付近はそれっぽい。

▽ $y = \log(x^2 + 1)$ のグラフを x - y 実数平面に描画

```
\begin{drawpict}[.55cm](12, 12)(-6, -6)
  \baseskip6
  \coordinate[R](-6, 6)(-6, 6)

  \begin{drawfunc}(-5.5, 5.5)%
  \def\calcdimen{ % y = log(x^2+1) を表す計算式
    \dimenmul(\strip@pt\dimen@, \strip@pt\dimen@)
    \dimensum(\strip@pt\dimen@, 1)
    \dimenlog
  }\end{drawfunc}
\end{drawpict}
```



$y = \log(x^2 + 1)$ の計算式 描画範囲は-5.5 から 5.5

```
\dimenmul(\strip@pt\dimen@, \strip@pt\dimen@)
  >> \dimen@が  $x$  の値である。  $x^2$  の計算。 臨時変数に退避させる必要はない。
\dimensum(\strip@pt\dimen@, 1)
  >>  $x^2 + 1$  の計算。 これにより計算結果が\dimen@に代入される。
\dimenlog
  >>  $\log x$  の計算 [補 5]。 これで\dimen@の値が  $y$  の値となる。
```

[**drawfuncp 環境**] ($x = f(t)$, $y = g(t)$ のグラフを描画するための環境)

```
\begin{drawfuncp}(#2,#3)%[補 2]
\@steps=0.01\p@ % 必要に応じて記述する描画精度の値 [補 3]
\def\calcdimenx{ % 関数の定義
(ここに  $f(t)$  に相当する計算式を記述)
}
\def\calcdimeny{ % 関数の定義
(ここに  $g(t)$  に相当する計算式を記述)
}\end{drawfuncp}
```

パラメータ t の区間—(#2, #3) の範囲—で、`\calcdimenx{...}` と `\calcdimeny{...}` で定義した関数を描画する。描画精度の値は記述しなくてもよい（そのときは、規定値の 0.05p@ が使われる）。`drawpict` 環境か `picture` 環境で使うことが前提だが、文章中で使用しても問題ない。

これは `\drawfuncp` 環境のパラメータ版である。**環境名が `\drawfuncp` となっていること、関数の定義式が `\calcdimenx` と `\calcdimeny` の 2 つがあることに注意**。この場合は、`\dimen@` がパラメータ t として扱われる。たとえば、 t をパラメータ θ として

```
\begin{drawfuncp}(0, 6)%
\def\calcdimenx{
\dimencos
}
\def\calcdimeny{
\dimensin
}\end{drawfuncp}
```

のようにすると、このようになるのは、文章中の描画単位が `pt` だからである。大きな円を描くなら

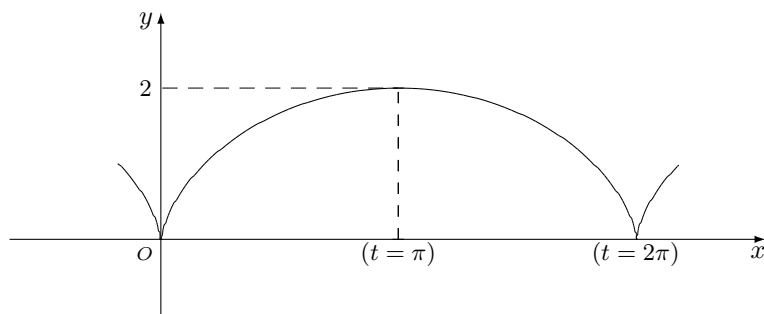
```
\begin{drawfuncp}(0, 6)%
\def\calcdimenx{
\dimencos
\dimenmul(\strip@pt\dimen@, 10)
}
\def\calcdimeny{
\dimensin
\dimenmul(\strip@pt\dimen@, 10)
}\end{drawfuncp}
```

のようにする（円が欠けているのはパラメータの範囲が 6 までだからである。円は一周 $2\pi \approx 6.28$ のパラメータ範囲が必要）。

$\nabla x = t - \sin t, y = 1 - \cos t$ のグラフ (サイクロイド) を x - y 実数平面に描画

```
\begin{drawpict}[1cm](10, 4)(-2, -1)
  \coordinate[A](-2, 8)(-1, 3)
  \locate[x](3.14, 2)
  \apeex(3.14, -.2){$(t=\pi)$}(6.28, -.2){$(t=2\pi)$}(-.2, 2){$2$}

  \begin{drawfuncp}(-1.57, 7.85)%  $-\pi/2 < t < (5/2)\pi$ 
  \def\calcdimenx{ %  $x = t - \sin(t)$  を表す計算式
    \tempa=\dimen@
    \dimensin
    \dimensum(\strip@pt\tempa, -\strip@pt\dimen@)
  }
  \def\calcdimeny{ %  $y = 1 - \cos(t)$  を表す計算式
    \dimencos
    \dimensum(1, -\strip@pt\dimen@)
  }\end{drawfuncp}
\end{drawpict}
```



$x = t - \sin t$ の計算式

```
\tempa=\dimen@
```

>> $\dimen@$ が t の値である。計算中に $\dimen@$ の値が変わるので臨時変数 \tempa に退避させる。

```
\dimensin
```

>> $\sin t$ の計算。これにより計算結果が $\dimen@$ に代入される。

```
\dimensum(\strip@pt\tempa, -\strip@pt\dimen@)
```

>> 正規の t (退避させた \tempa) から $\sin t$ を引く。これで $\dimen@$ の値が x の値となる。

$y = 1 - \cos t$ の計算式

```
\dimencos
```

>> $\cos t$ の計算。臨時変数に退避させる必要はない。

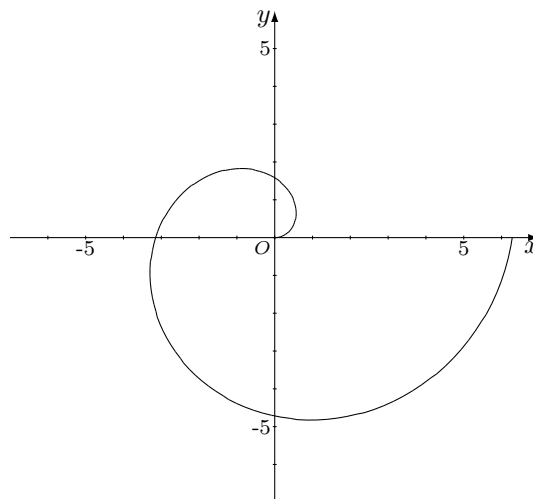
```
\dimensum(1, -\strip@pt\dimen@)
```

>> $1 - \cos t$ の計算。これにより計算結果が $\dimen@$ に代入され、 y の値となる。

$\nabla x = t \cos t, y = t \sin t$ のグラフを x - y 実数平面に描画

```
\begin{drawpict}[.5cm](14, 13)(-7, -7)
  \coordinate[R](-7, 7)(-7, 6)

  \begin{drawfuncp}(0, 6.28318)% 0 < t < 2π
  \def\calcdimenx{ % x = t cos(t) を表す計算式
    \tempa=\dimen@
    \dimencos
    \dimenmul(\strip@pt\tempa, \strip@pt\dimen@)
  }
  \def\calcdimeny{ % y = t sin(t) を表す計算式
    \tempa=\dimen@
    \dimensin
    \dimenmul(\strip@pt\tempa, \strip@pt\dimen@)
  }\end{drawfuncp}
\end{drawpict}
```



$x = t \cos t$ の計算式

```
\tempa=\dimen@
```

>> $\dimen@$ が t の値である。計算中に $\dimen@$ の値が変わるので臨時変数 \tempa に退避させる。

```
\dimencos
```

>> $\cos t$ の計算。これにより計算結果が $\dimen@$ に代入される。

```
\dimensum(\strip@pt\tempa, \strip@pt\dimen@)
```

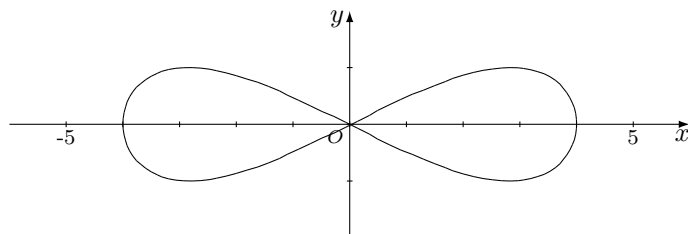
>> 正規の t (退避させた \tempa) に $\cos t$ の値 ($\dimen@$) を掛ける。これで $\dimen@$ の値が x の値となる。

※ $y = t \sin t$ の計算は $t \cos t$ の計算に同じ。

$\nabla x = 4 \cos t, y = \sin 2t$ のグラフを x - y 実数平面に描画

```
\begin{drawpict}[.75cm](12, 4)(-6, -2)
  \coordinate[R](-6, 6)(-2, 2)

  \begin{drawfuncp}(0, 6.28)% 0 < t < 2π
  \def\calcdimenx{ % x = 4cos(t) を表す計算式
    \dimencos
    \dimenmul(\strip@pt\dimen@, 4)
  }
  \def\calcdimeny{ % y = sin(2t) を表す計算式
    \dimenmul(\strip@pt\dimen@, 2)
    \dimensin
  }\end{drawfuncp}
\end{drawpict}
```



$x = 4 \cos t$ の計算式

```
\dimencos
```

>> $\cos t$ の計算。臨時変数に退避させる必要はない。

```
\dimenmul(\strip@pt\dimen@, 4)
```

>> $\cos t$ を 4 倍する。これにより計算結果が $\dimen@$ に代入される。

$y = \sin 2t$ の計算式

```
\dimenmul(\strip@pt\dimen@, 2)
```

>> $2t$ の計算。この時点で $\dimen@ (t)$ の値は $2t$ に変わる。

```
\dimensin
```

>> $\sin t$ の計算。これにより計算結果が $\dimen@$ に代入される。

[**drawfinv 環境**] ($y = f^{-1}(x)$ のグラフを描画するための環境) [補 6]

```
\begin{drawfinv}(\#2,\#3)%
\@steps=0.01\p@ % 必要に応じて記述する描画精度の値
\def\calcdimen{ % 関数の定義
(ここに  $f(x)$  に相当する計算式を記述)
}\end{drawfinv}
```

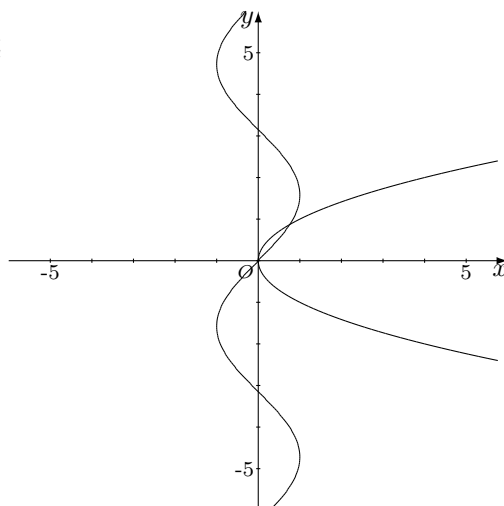
x の区間—($\#2, \#3$) の範囲—で、`\calcdimen{...}` で定義した関数の逆関数を描画する。つまりこの環境で、 $y = x^2$ を定義すれば $y = \pm\sqrt{x}$ のグラフが、 $y = \sin x$ を定義すれば $y = \arcsin x$ のグラフが描かれる (いずれも、($\#2, \#3$) の範囲が y の範囲になることに注意)。

▽ $y = x^2$ の逆関数と $y = \sin x$ の逆関数のグラフを x - y 実数平面に描画

```
\begin{drawpict}[.55cm](12, 12)(-6, -6)
\baseskip6
\coordinate[R](-6, 6)(-6, 6)

\begin{drawfinv}(-2.4, 2.4)%
\def\calcdimen{ %  $x^2$  を表す計算式
\dimenmul(\strip@pt\dimen@, \strip@pt\dimen@)
}\end{drawfinv}

\begin{drawfinv}(-6, 6)%
\def\calcdimen{ %  $\sin(x)$  を表す計算式
\dimensin
}\end{drawfinv}
\end{drawpict}
```



[変数 (レジスタ)]

- `\dimen@` 最初は $y = f(x)$ に代入する x の値 (描画区間の左辺値) が代入される。
`\dimen@` の値は計算の都度更新され、最後に代入された値が y の値になる。
- `\tempa` `\dimen@` の値を一時退避させる。
他に `\tempb`、`\tempc` が使用可能。
- `\@steps` 描画精度を代入する。[規定値] は 0.05p@ 。

[計算式 (マクロ)]

- `\dimensum(#1,#2)` #1 に#2 を加えた値が `\dimen@` に代入される。
#1、#2 が変数の場合は、必ず `\strip@pt` を変数の前につける。
#1、#2 が数値の場合は、そのまま数値を記述する。
引き算の場合は `\dimensum(#1,-#2)` とする。
- `\dimenmul(#1,#2)` #1 に#2 を掛けた値が `\dimen@` に代入される。
#1、#2 が変数の場合は、必ず `\strip@pt` を変数の前につける。
#1、#2 が数値の場合は、そのまま数値を記述する。
- `\dimendiv(#1,#2)` #1 を#2 で割った値が `\dimen@` に代入される。
#1、#2 が変数の場合は、必ず `\strip@pt` を変数の前につける。
#1、#2 が数値の場合は、そのまま数値を記述する。
- `\dimensqrt{#1}` #1 の平方根が `\dimen@` に代入される。
`\tempa=\dimen@ \dimensqrt{\strip@pt\tempa}` のように使う。
- `\dimencbrt{#1}` #1 の立方根が `\dimen@` に代入される。
`\tempa=\dimen@ \dimencbrt{\strip@pt\tempa}` のように使う。
- (注意) `\dimensqrt` と `\dimencbrt` は、引数#1 に `\strip@pt\dimen@` を使用できない。
理由は、`\dimen@` が `\dmsqrt` と `\dimencbrt` のマクロ計算中でも使われているため。
- `\dimenexp` $e(\dimen@)$ の値が `\dimen@` に代入される。
- `\dimenlog` $\log_e(\dimen@)$ の値が `\dimen@` に代入される。
- `\dimensin` $\sin(\dimen@)$ の値が `\dimen@` に代入される。
- `\dimencos` $\cos(\dimen@)$ の値が `\dimen@` に代入される。

- [補 1] `drawfunc` 環境で指定する描画範囲と描画精度の組み合わせによっては、描画できない場合がある。微妙な誤差処理のためと思われる。その場合は、描画範囲の開始の数値を細かくすると描画できる。具体的には、描画精度 0.01p@ 、範囲 $(-0.6, -0.02)$ ではだめでも、範囲を $(-0.636, -0.02)$ にすれば大丈夫、というように。
- [補 2] `drawfunc` 環境、および `drawfuncp` 環境でもグラフを破線で描画できる。一般のグラフ同様、描画範囲 $(\#2, \#3)$ の前に `:` を入れればよい。こちらも変化量が大きく変化する区間では、破線の間隔が少々乱れたり、描画に時間がかかったりと難点が多いが、ぎりぎり実用になると思われる。
- [補 3] 曲線は微小区間 (区間幅の規定値は 0.05p@) を `\qbezier` による直線で描いている。精度の変更は、それぞれのグラフ描画において、`\begin{drawfunc}(\#2,\#3)` の後に `\@steps=0.01\text{p@}` のように指定する。`\begin{drawfunc}(\#2,\#3)` の場合も、その後に `\@steps=0.01\text{p@}` のように一度指定すればよい。ただし、精度を細かくしても必ずしもきれいに描画できるわけではない。
- [補 4] 臨時変数は `drawfunc` および `drawfuncp` 環境内で `\tempa`, `\tempb`, `\tempc` まで用意したが、必要なら `\tempd`, ... 以降の変数を増やしてかまわない。
- [補 5] `\dimensin`, `\dimencos` は 0° から 360° まで 1° 刻みで、`\dimenlog` は 1 から 999 まで 1 刻みで、数表から値を読んで計算している。
- また、平方根と立方根の計算はニュートンの近似式で求めている。計算時間を考慮して、計算回数を `\Tms<6` に押さえている。計算回数を上げればより正確な値になるが、だからといってきれいに描画できるわけではない。
- [補 6] `drawfinv` 環境でも、描画範囲 $(\#2, \#3)$ の前に `:` を入れて破線で描画できる。
- また、**ここだけの秘密だが** `drawfinv` 環境のパラメータ版である、`drawfinvp` 環境も定義してある。
- [補 7] 計算式の記述などは忘れやすいので、忘備用に `\dman` マクロを組み入れてある。`\dman` と書いて処理すれば、簡易マニュアルとして以下の記述が表示される。

<code>\#1, \#2</code>	<code>\strip@pt\dimen@</code> <code>\strip@pt\tempa</code> , <code>\strip@pt\tempb</code> , <code>\strip@pt\tempc</code> 数値
<code>\dimensum(\#1,\#2)</code>	$(\#1) + (\#2) \rightarrow \dimen@$
<code>\dimensum(\#1,-\#2)</code>	$(\#1) - (\#2) \rightarrow \dimen@$
<code>\dimenmul(\#1,\#2)</code>	$(\#1) * (\#2) \rightarrow \dimen@$
<code>\dimendiv(\#1,\#2)</code>	$(\#1)/(\#2) \rightarrow \dimen@$
<code>\dimensqrt{\#1}</code>	$\sqrt{\#1} \rightarrow \dimen@$
<code>\dimencbrt{\#1}</code>	必ず <code>\tempa=\dimen@\dimensqrt{\strip@pt\tempa}</code> $\sqrt[3]{\#1} \rightarrow \dimen@$
<code>\dimenexp</code>	必ず <code>\tempa=\dimen@\dimencbrt{\strip@pt\tempa}</code> $e^{(\dimen@)} \rightarrow \dimen@$
<code>\dimenlog</code>	$\log_e(\dimen@) \rightarrow \dimen@$
<code>\dimensin</code>	$\sin(\dimen@) \rightarrow \dimen@$
<code>\dimencos</code>	$\cos(\dimen@) \rightarrow \dimen@$

appendix A2

`drawfunc` 環境と `drawfuncp` 環境を使えば様々な関数のグラフが描けますが、計算の定義式を記述するのが大変だと思われます。私は、自分で定義したものを好きで使っているわけですから、このような記述であっても苦になりません。しかし、`tmtmath.sty` を手軽に使用しようと考える人には、定義式の記述は苦行でしかないはず。そこで、計算の定義式を簡略化して再定義してみました。ただし、簡略化するほどマクロの種類が多くなるので、簡略化とマクロの種類のトレードオフが問題となります。トレードオフの落としどころは人それぞれでしょうから、この再定義が不満なら自分にあったものになるよう、`tmtmath.sty` を修正するとよいでしょう。

簡易計算式として以下のように再定義（二重定義）した（`\dimenexp`, `\dimenlog`, `\dimensin`, `\dimencos` は再定義していない）。

<code>\pusha</code>	→	<code>\dimen@</code> の値を <code>\tempa</code> に退避させる
<code>\pushb</code>	→	<code>\dimen@</code> の値を <code>\tempb</code> に退避させる
<code>\pushc</code>	→	<code>\dimen@</code> の値を <code>\tempc</code> に退避させる
<code>\popa</code>	→	<code>\tempa</code> に退避させた値を <code>\dimen@</code> に代入する
<code>\popb</code>	→	<code>\tempb</code> に退避させた値を <code>\dimen@</code> に代入する
<code>\popc</code>	→	<code>\tempc</code> に退避させた値を <code>\dimen@</code> に代入する
<code>\dplus#1</code>	→	$(\dimen@) + (\#1)$ の値を <code>\dimen@</code> に代入する（ $\#1 < 0$ なら引き算）
<code>\dplusa</code>	→	$(\dimen@) + (\tempa)$ の値を <code>\dimen@</code> に代入する
<code>\dplusb</code>	→	$(\dimen@) + (\tempb)$ の値を <code>\dimen@</code> に代入する
<code>\dplusc</code>	→	$(\dimen@) + (\tempc)$ の値を <code>\dimen@</code> に代入する
<code>\dminusa</code>	→	$(\dimen@) - (\tempa)$ の値を <code>\dimen@</code> に代入する
<code>\dminusb</code>	→	$(\dimen@) - (\tempb)$ の値を <code>\dimen@</code> に代入する
<code>\dminusc</code>	→	$(\dimen@) - (\tempc)$ の値を <code>\dimen@</code> に代入する
<code>\dtimes#1</code>	→	$(\dimen@) \times (\#1)$ の値を <code>\dimen@</code> に代入する
<code>\dtimesa</code>	→	$(\dimen@) \times (\tempa)$ の値を <code>\dimen@</code> に代入する
<code>\dtimesb</code>	→	$(\dimen@) \times (\tempb)$ の値を <code>\dimen@</code> に代入する
<code>\dtimesc</code>	→	$(\dimen@) \times (\tempc)$ の値を <code>\dimen@</code> に代入する
<code>\dsquare</code>	→	$(\dimen@) \times (\dimen@)$ の値を <code>\dimen@</code> に代入する（すなわち2乗）
<code>\dover#1</code>	→	$(\dimen@) / (\#1)$ の値を <code>\dimen@</code> に代入する
<code>\dovera</code>	→	$(\dimen@) / (\tempa)$ の値を <code>\dimen@</code> に代入する
<code>\doverb</code>	→	$(\dimen@) / (\tempb)$ の値を <code>\dimen@</code> に代入する
<code>\doverc</code>	→	$(\dimen@) / (\tempc)$ の値を <code>\dimen@</code> に代入する
<code>\dunder#1</code>	→	$(\#1) / (\dimen@)$ の値を <code>\dimen@</code> に代入する
<code>\dundera</code>	→	$(\tempa) / (\dimen@)$ の値を <code>\dimen@</code> に代入する
<code>\dunderb</code>	→	$(\tempb) / (\dimen@)$ の値を <code>\dimen@</code> に代入する
<code>\dunderc</code>	→	$(\tempc) / (\dimen@)$ の値を <code>\dimen@</code> に代入する
<code>\dsqrt</code>	→	$\sqrt{\dimen@}$ の値を <code>\dimen@</code> に代入する

再定義に従い appendix A1 の計算式部分だけを書き直すと以下のような。ただし、再定義した計算式はすべて `\dimen@` に対するものなので、直接 `\tempa` に 1 を加えるようなことはできない。その場合は `\popa` などを使うか、`\tempa` に対する計算式を定義する（以下では `\popa` で対処している）。

$$\nabla y = xe^{-x}$$

```
\def\calcdimen{
  \pusha
  \dtimes{-1}
  \dimenexp
  \dtimesa }
```

$$\nabla y = \log(x^2 + 1)$$

```
\def\calcdimen{
  \dsquare
  \dplus1
  \dimenlog }
```

$$\nabla y = \frac{x+2}{x^2+1}$$

```
\def\calcdimen{
  \pusha
  \dsquare
  \dplus1
  \pushb
  \popa
  \dplus2
  \doverb }
```

$$\nabla x = t - \sin t, \quad y = 1 - \cos t$$

```
\def\calcdimenx{
  \pusha
  \dimensin
  \dtimes{-1}
  \dplusa }
\def\calcdimeny{
  \dimencos
  \dtimes{-1}
  \dplus1 }
```

$$\nabla y = x + \frac{1}{x}$$

```
\def\calcdimen{
  \pusha
  \dunder1
  \dplusa }
```

$$\nabla x = t \cos t, \quad y = t \sin t$$

```
\def\calcdimenx{
  \pusha
  \dimencos
  \dtimesa }
\def\calcdimeny{
  \pusha
  \dimensin
  \dtimesa }
```

$$\nabla y = \frac{\sin x}{x}$$

```
\def\calcdimen{
  \pusha
  \dimensin
  \dovera }
```

$$\nabla x = 4 \cos t, \quad y = \sin 2t$$

```
\def\calcdimenx{
  \dimencos
  \dtimes4 }
\def\calcdimeny{
  \dtimes2
  \dimensin }
```

$$\nabla y = \sin \frac{1}{x}$$

```
\def\calcdimen{
  \dunder1
  \dimensin }
```

appendix A3

appendix A1 で、 $y = xe^{-x}$ のグラフなら `\easydraw[x e^{-x}](-1.4, 5.5)`、 $y = x + \frac{1}{x}$ なら `\easydraw[x+1/x](-5.5, 5.5)` のように記述して描画できれば申し分ないと書きましたが、ここでは代替マクロ `\easyfx` を定義しました。このマクロでは、 $y = xe^{-x}$ は `\easyfx[x{-1}x*e*](-1.4, 5.5)`、 $y = x + \frac{1}{x}$ は `\easyfx[x1x/+](-5.5, -.18)\easyfx[x1x/+] (.18, 5.5)` のように記述します。関数式の記述は、“逆ポーランド記法” に従い素のまま定義します。drawfunc 環境に比べて、使い勝手は格段によくなっていると思います。

`\easyfx[#1]#2(#3,#4)` `\easyfx[関数式]{op}(x_l, x_r)`

逆ポーランド記法で記述した #1 のグラフを区間 (#3, #4) で描画。#2 はオプションで : のみ。

◇関数式に使える演算子等について

[二項演算子 (二項後オペレーター)]

演算子	機能	記述例	通常の記述	
+	和 A+B を計算する	x2+	$x + 2$	[補 1]
-	差 A-B を計算する	x{10}-	$x - 10$	[補 2]
*	積 A*B を計算する	xx*	x^2	
/	商 A/B を計算する	1x/	$\frac{1}{x}$	

[単項演算子 (後置オペレーター)]

演算子	機能	記述例	通常の記述
r	\sqrt{A} を計算する	x2+r	$\sqrt{x + 2}$
e	e^A を計算する	{-1}x*e	e^{-x}
l	$\log_e A$ を計算する	x1	$\log_e x$ または $\ln x$
s	$\sin A$ を計算する	{3.14}s	$\sin \pi$
c	$\cos A$ を計算する	xc	$\cos x$
t	$\tan A$ を計算する	x2/t	$\tan \frac{x}{2}$
	$ A $ を計算する	x2+	$ x + 2 $

[変数と数値 (オペランド)]

変数・数値	機能	記述例	通常の記述	
x	変数	x, {-1}x*	$x, -x$	[補 3]
数値	数値	5, {-2}, {0.333}	$5, -2, \frac{1}{3}$	[補 4]

※ 通常の式を逆ポーランド記法の式にするには、 x に値を代入して計算するときの順番に従って、逐次 逆ポーランド記法の式に直して—新たな項にして—いくとよい。また、逆ポーランド記法の式を通常の式にするには、逆ポーランド記法の式を左から順に見て、演算子を見つけたら 逐次 直前の 2 項または 1 項を、通常の式に直して—新たな項にして—いくとよい。次ページに実際の例を示す¹。

¹逆ポーランド記法の詳細については、ネットワークなどで調べてもらいたい。

関数式 $f(x) = xe^{-x}$

関数式 $f(x) = \frac{x+2}{x^2+1}$

[$\times \div$ の式にする]

$$x \times e^{(-1) \times x}$$

[$A \times B \rightarrow AB^*$ より]

$$x \times e^{\{-1\}x^*}$$

[$e^A \rightarrow Ae$ より]

$$x \times \{-1\}x^*e$$

[$A \times B \rightarrow AB^*$ より]

$$x\{-1\}x^*e^*$$

[$\times \div$ の式にする]

$$(x+2) \div (x \times x+1)$$

[$A+B \rightarrow AB^+$ より]

$$(x2+) \div (x \times x+1)$$

[$A \times B \rightarrow AB^*$ より]

$$(x2+) \div ((xx^*)+1)$$

[$A+B \rightarrow AB^+$ より]

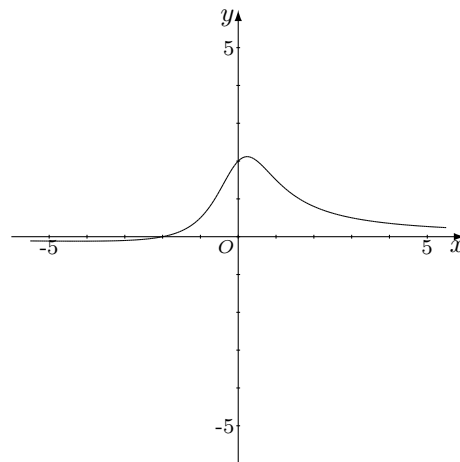
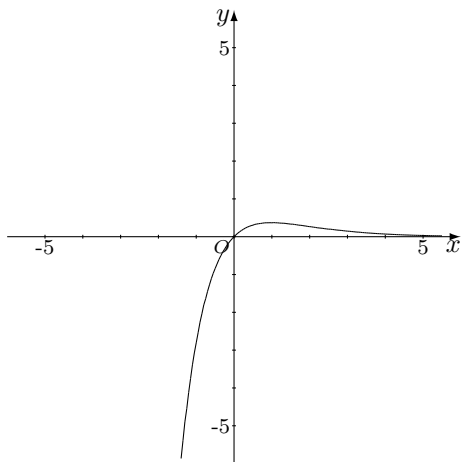
$$(x2+) \div (xx^*1+)$$

[$A \div B \rightarrow AB/$ より]

$$x2+xx^*1+ /$$

```
\begin{drawpict}[.5cm](26, 12)(-6, -6)
\coordinate[R](-6, 6)(-6, 6) %左のグラフ
\easyfx[x\{-1\}x^*e^*](-1.4, 5.5)

\baseskip{14}
\coordinate[R](-6, 6)(-6, 6) %右のグラフ
\easyfx[x2+xx*1+ /](-5.5, 5.5)
\end{drawpict}
```



$$\text{関数式 } f(x) = x + \frac{1}{x}$$

$$\text{関数式 } f(x) = \frac{\sin x}{x}$$

[×÷ の式にする]

$$x + 1 \div x$$

[×÷ の式にする]

$$\sin x \div x$$

[A ÷ B → AB/ より]

$$x + (1x/)$$

[sin A → As より]

$$(xs) \div x$$

[A + B → AB+ より]

$$x1x/+$$

[A ÷ B → AB/ より]

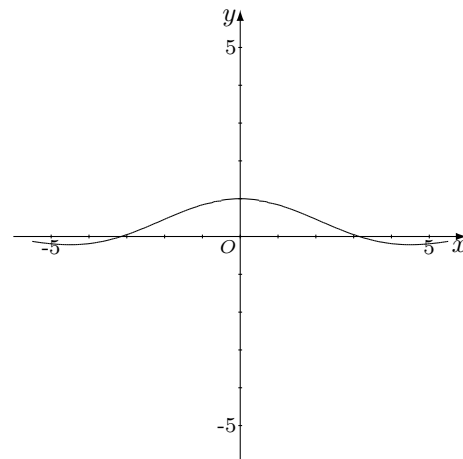
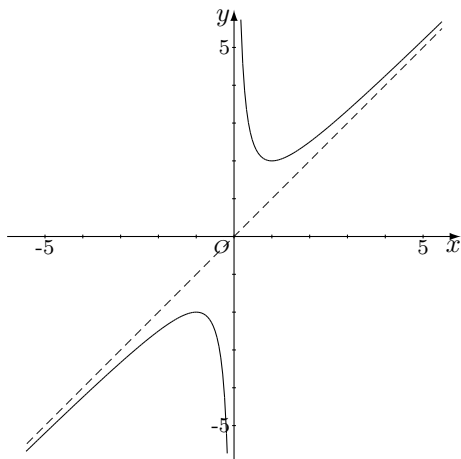
$$xsx/$$

```

\begin{drawpict}[.5cm](26, 12)(-6, -6)
  \coordinate[R](-6, 6)(-6, 6) %左のグラフ
  \easyfx[x1x/+](-5.5, -.18) \easyfx[x1x/+] (.18, 5.5)
  \easyfx[1x*]:(-5.5, 5.5) %漸近線 [補 3]

  \baseskip{14}
  \coordinate[R](-6, 6)(-6, 6) %右のグラフ
  \easyfx[xsx/]( -5.5, -.5)
  \qbezier(-.5, .9588)(-.2468, 1)(0, 1) %-0.5<x<0.5 の範囲を\qbezier で補正
  \qbezier(0, 1)(.2468, 1)(.5, .9588)
  \easyfx[xsx/](.5, 5.5)
\end{drawpict}

```



$$\text{関数式 } f(x) = \sin \frac{1}{x}$$

[$\times \div$ の式にする]

$$\sin(1 \div x)$$

[$A \div B \rightarrow AB/$ より]

$$\sin(1x/)$$

[$\sin A \rightarrow As$ より]

$$1x/s$$

$$\text{関数式 } f(x) = \log_e(x^2 + 1)$$

[$\times \div$ の式にする]

$$\log_e(x \times x + 1)$$

[$A \times B \rightarrow AB*$ より]

$$\log_e((xx*) + 1)$$

[$A + B \rightarrow AB+$ より]

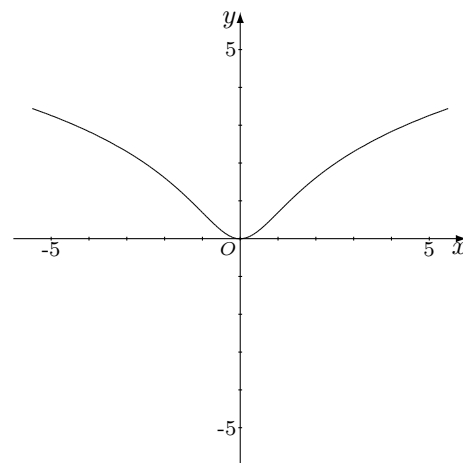
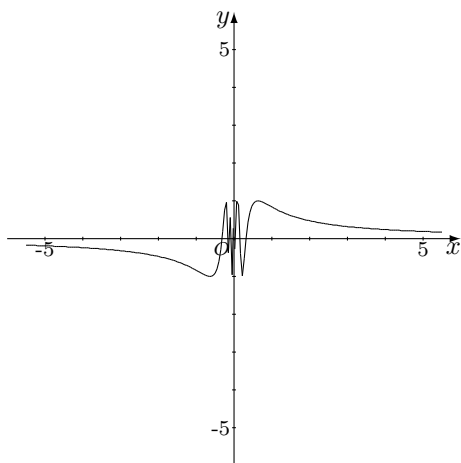
$$\log_e(xx*1+)$$

[$\log_e A \rightarrow A1$ より]

$$xx*1+1$$

```
\begin{drawpict}[.5cm](26, 12)(-6, -6)
\coordinate[R](-6, 6)(-6, 6) %左のグラフ
\easyfx[1x/s](-5.5, -.02) %(x<0)
\easyfx[1x/s](.02, 5.5) %(x>0)

\baseskip{14}
\coordinate[R](-6, 6)(-6, 6) %右のグラフ
\easyfx[xx*1+1](-5.5, 5.5)
\end{drawpict}
```



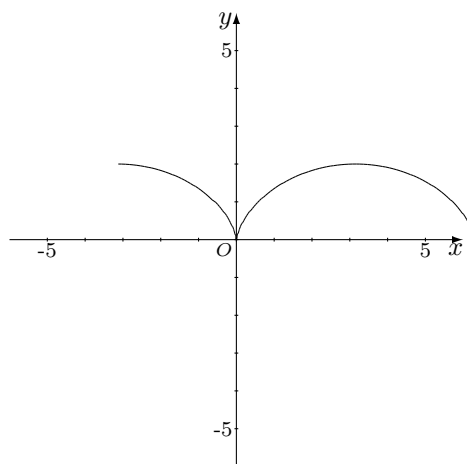
■ 媒介変数を用いた関数 $x = f(t)$, $y = g(t)$ のグラフは、逆ポーランド記法の計算ルーティン `\RPNfx[]`, `\RPNgx[]` と `drawfuncp` 環境を組み合わせで描画する。

```
\begin{drawfuncp}(#2,#3)
  \def\calcdimenx{\RPNfx[f(t)の関数式]}
  \def\calcdimeny{\RPNgx[g(t)の関数式]}
\end{drawfuncp}
```

$$\text{関数式 } f(t) = t - \sin t, g(t) = 1 - \cos t$$

(変数 t は x にして) `xxs-`, `1xc-`

```
\begin{drawpict}[.5cm](12,12)(-6,-6)
  \coordinate[R](-6,6)(-6,6)
  \begin{drawfuncp}(-3.14,6)
    \def\calcdimenx{\RPNfx[xxs-]}
    \def\calcdimeny{\RPNgx[1xc-]}
  \end{drawfuncp}
\end{drawpict}
```



■ 逆関数 $y = f^{-1}(x)$ のグラフは、逆ポーランド記法の計算ルーティン `\RPNfx[]` と `drawfinv` 環境を組み合わせで描画する。

```
\begin{drawfinv}(#2,#3)
  \def\calcdimenx{\RPNfx[f(x)の関数式]}
\end{drawfinv}
```

- [補 1] たとえば x^2+ は $x^2 +$ のように、英字空白で区切って記述してもよい。
- [補 2] 数値が 1 文字でないとき—小数や負の数など—は、必ず $\{10\}$ のように中括弧で囲む。1 文字の数値を中括弧で囲んでも問題ないので、見やすさのために囲むのもよいだろう。
- [補 3] `\easyfx`[関数式](..) は演算子を見つけて計算する仕組みなので、 x だけを単独で使うことはできない。たとえば、 $y = x$ の関数式は `\easyfx[x](..)` ではなく、`\easyfx[1x*](..)` とする。同様に定数関数 $y = 5$ なども `\easyfx[5](..)` ではなく、`\easyfx[50+]` のようにする。
- [補 4] 分数は (近似) 小数で表す。分数は数値ではなく計算式として $13/$ のように記述できるが、精度が高まるわけではなく、処理時間も余分にかかるので小数に直すべきである。
- [補 5] このマクロには、計算処理用のスタックが 6 個だけ用意されている。簡単な関数ならスタックは 6 個で十分だろう。たとえば $x + (x + (x + (x + (x + (x + x))))))$ は、`xxxxxxxx+++++` と書くとスタックを最大 7 個使うので計算できない。`xx+x+x+x+x+x+` と書けば問題ないが、そもそも $x + x + x + x + x + x + x = 7x$ であるから、`7x*` と書けばよい。どうしてもスタックが 7 個以上必要なら、`tmtmath.sty` ファイルを書き換えてほしい。書き換えるべき箇所—(if in need) と書かれた 4 箇所—は%でコメントアウトにしてある。
- [補 6] 結局のところ、たとえば $y = x^2$ のグラフを描画するには、`\parabola[1,0,0](..)` でも `\easyfx[xx*](..)` でも同じである。なぜなら最終的に使う計算ルーティンが同じだからである。しかし `\parabola` は律儀に係数 0 の計算もするのに対し、`\easyfx` は無駄な計算をしない。係数の計算だけが原因ではないが、計算量の点では `\easyfx` の方が数倍有利である。

appendix B

$\text{T}_{\text{E}}\text{X}$ の `picture` 環境だけでグラフや図を描くのは大変で、必ずしも思い通りの描画ができるわけではありません。しかし、ある程度それが実現すると、「他にこんな機能も…」という欲求が生じます。その場合は、他のマクロ集を使うべきでしょう。この `tmtmath.sty` は、あくまでもこれひとつで描画を支援するものですから、手書き以上に手間がかかることはしません。実際私は、 $\text{T}_{\text{E}}\text{X}$ での描画が厳しいときは手書きしています。領域を示す斜線などもその類いです。というものの、領域を示す斜線を描画するマクロも定義してみました。もちろん、出来が今ひとつで使い勝手も良くないので、不要なおマケ機能と考えてもらえばよいでしょう。

```
\neobliqs#1(#2,#3)(#4,#5)(#6,#7)(#8,#9)
```

```
\neobliqs{b}(x_{1l}, x_{1r})(x_{2l}, x_{2r})(x_{3l}, x_{3r})(x_{4l}, x_{4r})
```

#1 を y 切片にもつ右上がり (傾き 1) の斜線から始め、 y 切片を 0.5 ずつ下げながら、4 本の斜線を引く。

```
\seobliqs#1(#2,#3)(#4,#5)(#6,#7)(#8,#9)
```

```
\seobliqs{b}(x_{1l}, x_{1r})(x_{2l}, x_{2r})(x_{3l}, x_{3r})(x_{4l}, x_{4r})
```

#1 を y 切片にもつ右下がり (傾き -1) の斜線から始め、 y 切片を 0.5 ずつ上げながら、4 本の斜線を引く。

```
\NEobliqs#1(#2,#3)(#4,#5)(#6,#7)(#8,#9)
```

```
\NEobliqs{b}(x_{1l}, x_{1r})(x_{2l}, x_{2r})(x_{3l}, x_{3r})(x_{4l}, x_{4r})
```

#1 を y 切片にもつ右上がり (傾き 1) の斜線から始め、 y 切片を 20 ずつ下げながら、4 本の斜線を引く。

```
\SEobliqs#1(#2,#3)(#4,#5)(#6,#7)(#8,#9)
```

```
\SEobliqs{b}(x_{1l}, x_{1r})(x_{2l}, x_{2r})(x_{3l}, x_{3r})(x_{4l}, x_{4r})
```

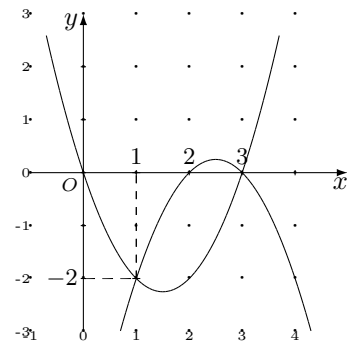
#1 を y 切片にもつ右下がり (傾き -1) の斜線から始め、 y 切片を 20 ずつ上げながら、4 本の斜線を引く。

※ 以上のマクロはすべて、左から右へ向けて斜線を引くことになる。斜線は順に、 x の区間 (#2, #3), (#4, #5), (#6, #7), (#8, #9) に引かれる。

$y = x^2 - 3x$ と $y = -x^2 + 5x - 6$ で囲まれる部分に斜線を引く場合を例にとる。まず、斜線を引く目安のために `\dotsgrid` で格子点を描画しておく（座標があるので必要ないかもしれないが、図形を描画する際は必要だろう）。

```
\begin{drawpict}[.7cm](6, 6)(-1, -3)
  \coordinate[R](-1, 5)(-3, 3)
  \parabola[1, -3, 0](-.7, 3.7)
  \parabola[-1, 5, -6](.7, 4.3)
  \locate[x](1, -2)
  \apeeex(1, .3){$1$}(2, .3){$2$}(3, .3){$3$}
  \apex(-.4, -2){$-2$}

  \dotsgrid1(-1, -3)(5, 4) % 格子点
\end{drawpict}
```



右下がりの斜線を引くとすると、最初の斜線の y 切片は -0.5 である。そこで `\seobliqs{-.5}` とし、 x の区間はとりあえず $(0, 0)$ としておく。斜線が4本では足りないので、`\seobliqs{1.5}` も加える（右下がりの場合、次の y 切片は2上である—右上がりの場合は2下）。

```
\begin{drawpict}[.7cm](6, 6)(-1, -3)}
  \coordinate[R](-1, 5)(-3, 3)
  \parabola[1, -3, 0](-.7, 3.7)
  \parabola[-1, 5, -6](.7, 4.3)
  \locate[x](1, -2)
  \apeeex(1, .3){$1$}(2, .3){$2$}(3, .3){$3$}
  \apex(-.4, -2){$-2$}

  \dotsgrid1(-1, -3)(5, 4)
  \seobliqs{-.5}(0, 0)(0, 0)(0, 0)(0, 0)
  \seobliqs{1.5}(0, 0)(0, 0)(0, 0)(0, 0)
\end{drawpict}
```

斜線を引く区間の値は、図を見ながらおよその目安で入れてもよいのだが、幸いグラフが2次関数なので、交点は楽に計算できる。斜線の左側は $y = -x^2 + 5x - 6$ と斜線の交点になるので、斜線を $y = -x + b$ とおいて $-x^2 + 5x - 6 = -x + b$ を解けばよい。解の公式より \pm の解が求まるが、左側の交点を採用するので $x = 3 - \sqrt{3 - b}$ が交点になる。この b に y 切片である $-0.5, 0, 0.5, 1, \dots, 3$ まで代入すれば、8本の斜線の始点 (x_l 座標) が分かる。計算は、電卓か表計算ソフトでするとよいだろう。その結果、以下のように左側の値が決まる（斜線は7本でよいので8本目の値は0のままにする）。

```
\seobliqs{-.5}(1.12, 0)(1.26, 0)(1.41, 0)(1.58, 0)
\seobliqs{1.5}(1.77, 0)(2, 0)(2.29, 0)(0, 0)
```

次は斜線の終点 (x_r 座標) である。こちらは、 $y = x^2 - 3x$ と $y = -x + b$ の右側の交点を採用し $x = 1 + \sqrt{1+b}$ が交点になる。同様に、 b に $-0.5, 0, 0.5, 1, \dots, 3$ まで代入する。

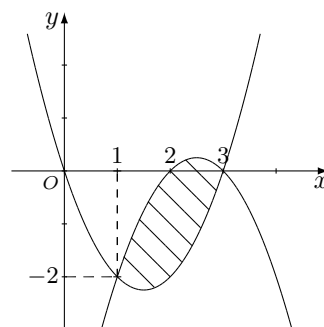
```
\seobliqs{-.5}(1.12, 1.7)(1.26, 2)(1.41, 2.22)(1.58, 2.41)
```

```
\seobliqs{1.5}(1.77, 2.58)(2, 2.73)(2.29, 2.87)(0, 0)
```

これで `\dotsgrid` を削除すれば出来上がりである。

```
\begin{drawpict}[.7cm](6, 6)(-1, -3)
\baseskip7
\coordinate[R](-1, 5)(-3, 3)
\parabola[1, -3, 0](-.7, 3.7)
\parabola[-1, 5, -6](.7, 4.3)
\locate[x](1, -2)
\apeeex(1, .3){$1$}(2, .3){$2$}(3, .3){$3$}
\apex(-.4, -2){$-2$}

\seobliqs{-.5}(1.12, 1.7)(1.26, 2)(1.41, 2.22)(1.58, 2.41)
\seobliqs{1.5}(1.77, 2.58)(2, 2.73)(2.29, 2.87)(0, 0)
\end{drawpict}
```



斜線の始点と終点の x 座標がすぐに分かれば実用になるだろうが、明らかに手書きするほうが速やかである。斜線の間隔は、描画単位が `cm` なら `\neobliqs` か `\seobliqs` を使って `0.5` に、描画単位が `pt` なら `\NEobliqs` か `\SEobliqs` を使って `20` になるように決めてある。変更は `tmtmath.sty` ファイルで行ってもらいたい。