Explore コマンドの更新

Explore コマンドでは、現在のドキュメントやワークシート内に対話型のアプリケーションを作成するための使いやすいツールを提供します。Maple 18 では、次のような大幅な更新と追加が行われています。

- コントロールの配置のカスタマイズ
- ダイアル、ゲージなどを含む範囲パラメータのコントロールタイプ
- コンボボックスコントロールを使用したパラメータの選択
- 表示コンポーネントサイズのカスタマイズ
- 多重パラメータのアニメーション
- 2-D プロットでの対話型マーカーコントロール
- 画像の対話的な調査

上記のリストに加え、Explore コマンドの対話型ポップアップダイアログも強化されています。このダイアログウィンドウは、parameters オプションが Explore コマンドのコールシーケンスで提供されない場合に表示されます。これには、式または未評価の関数コールを右クリックすると表示されるコンテキストメニューを使用して Explore を呼び出すケースも含まれます。

追加の詳細と実例は、Explore のワークシート例を参照してください。

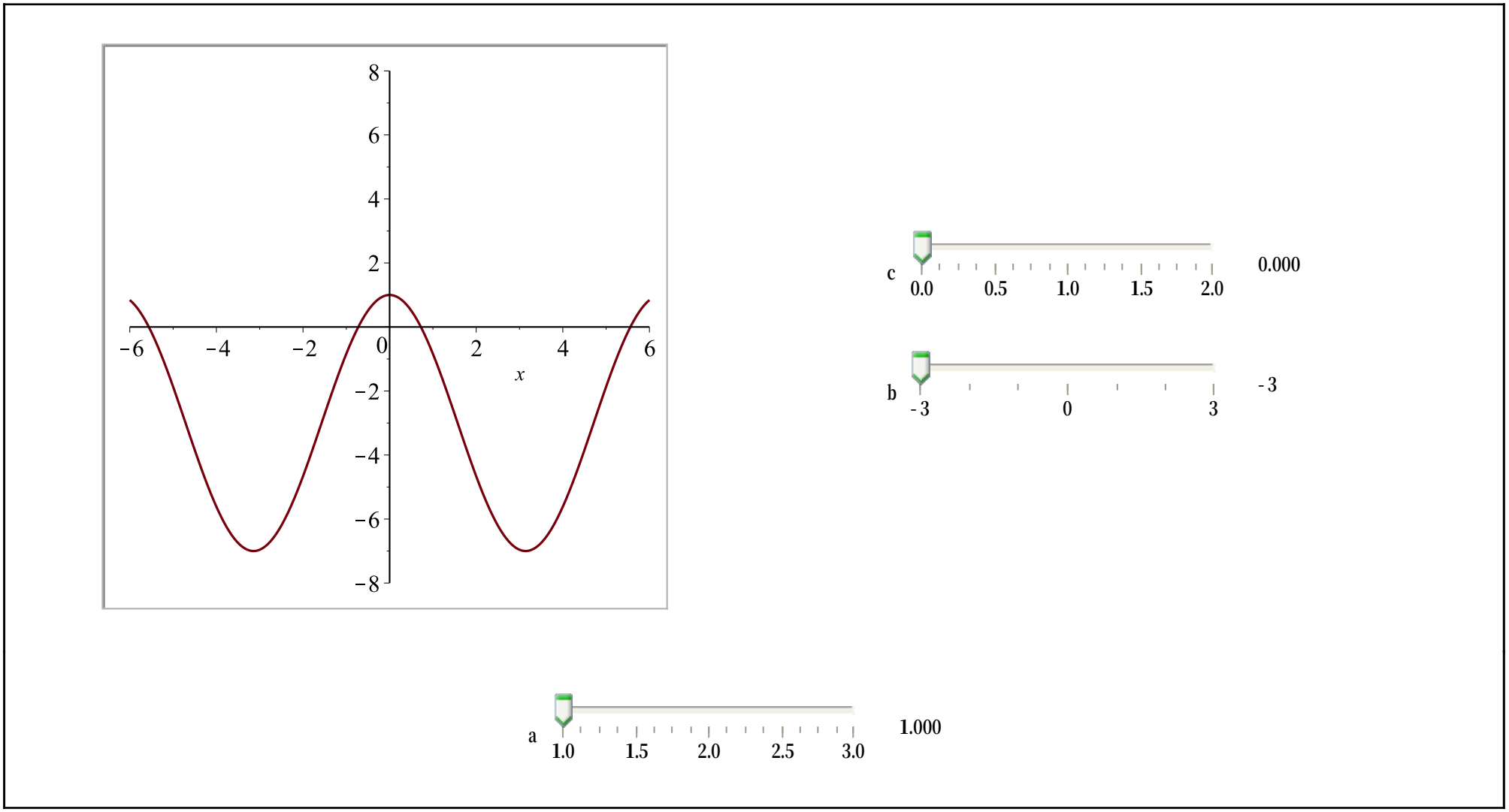
コントロールの配置

パラメータのコントロールは、表示コンポーネントの下、左、または右側に配置できます。カスタマイズされたデフォルトとして、単一のオプションの引数 placement によってすべてのパラメータのコントロールを Explore コマンドに配置できます。

パラメータのコントロールの配置は、デフォルト設定をオーバーライドすることで個別に指定することもできます。

```
> Explore( plot(b + |(c - 2) (c + 2)| cos(a x), x = -6..6, view = -8..8 ),
                parameters = [c = 0.0..2,
```

```
    b =-3 ..3,
    [a = 1 ..3.0, placement = bottom ]],

placement = right)
```



範囲パラメータのコントロールタイプ

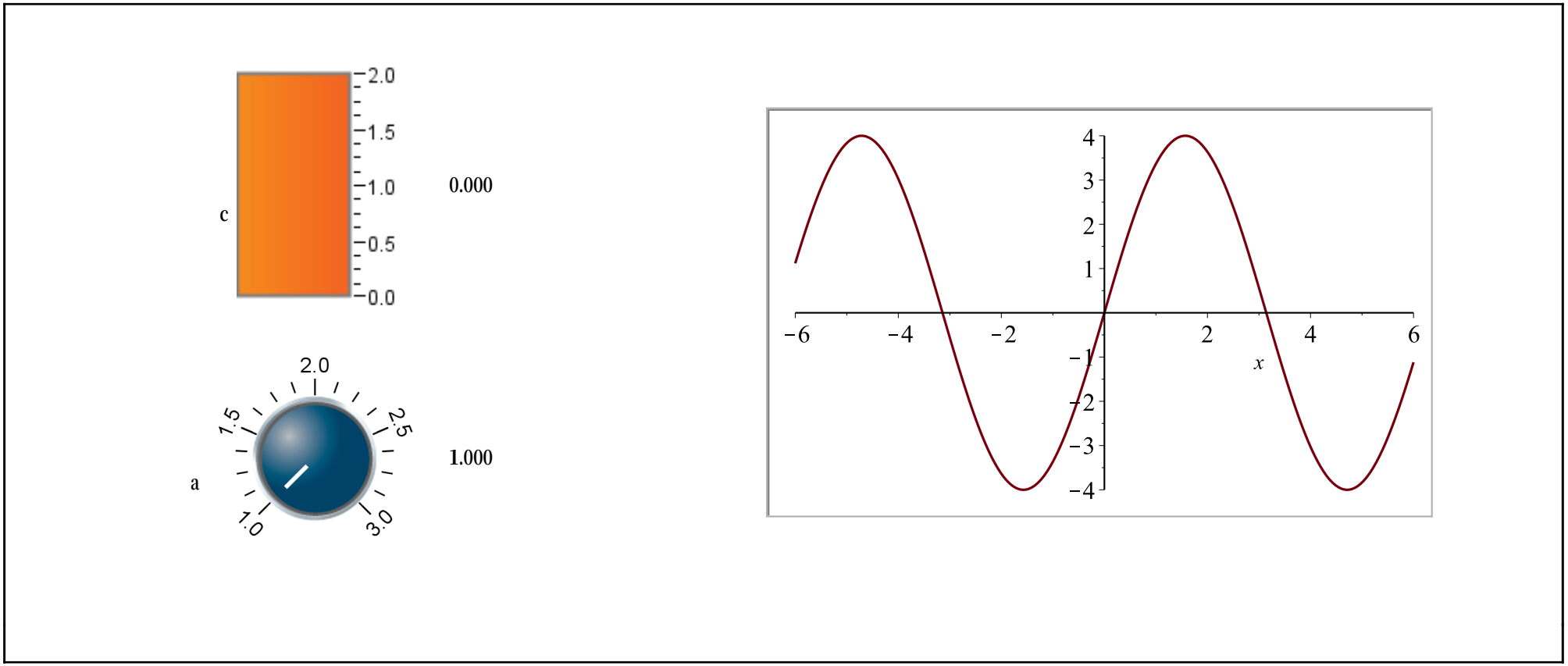
値が範囲として指定されたパラメータを、変数コントロールのいくつかのタイプの内のひとつとして指定できます。

変数コントローラの選択肢として slider、dial、volumegauge、meter、およびrotarygauge があります。

プロットするコンポーネントのサイズは、オプションで指定できます。デフォルトでは、サイズは最初のプロットのサイズで決められます。

```
> Explore(plot(|(c − 2) (c + 2)| sin(a x), x = −6 ..6, view = −4 ..4),
```

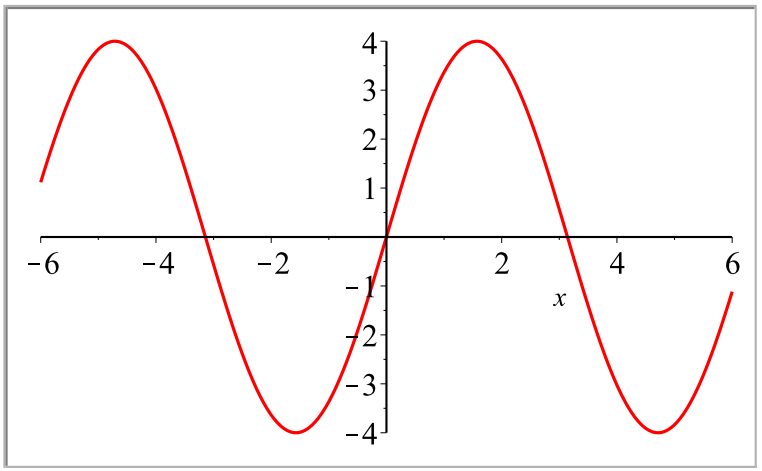
```
    parameters = [ [c = 0.0 ..2, controller = volumegauge],
                   [a = 1 ..3.0, controller = dial] ],

    size = [400, 246],
    placement = left)
```



コンボボックスコントロール

値がリストとして指定されたパラメータは、コンボボックスコントロールに組み込まれます。

```
> Explore(plot(|(c − 2) (c + 2)| sin(a x), x = −6 ..6, view = −4 ..4, color = r, style = s),

                parameters = [ [r = [red, blue, green], label = color],
                               [s = [line, point], label = style],
                               c = 0.0 ..2,
                               a = 1 ..3.0 ],

               size = [400, 246],
               placement = right)
```



color red

style line

c 0.0 0.5 1.0 1.5 2.0 0.000

a 1.0 1.5 2.0 2.5 3.0 1.000

アニメーション

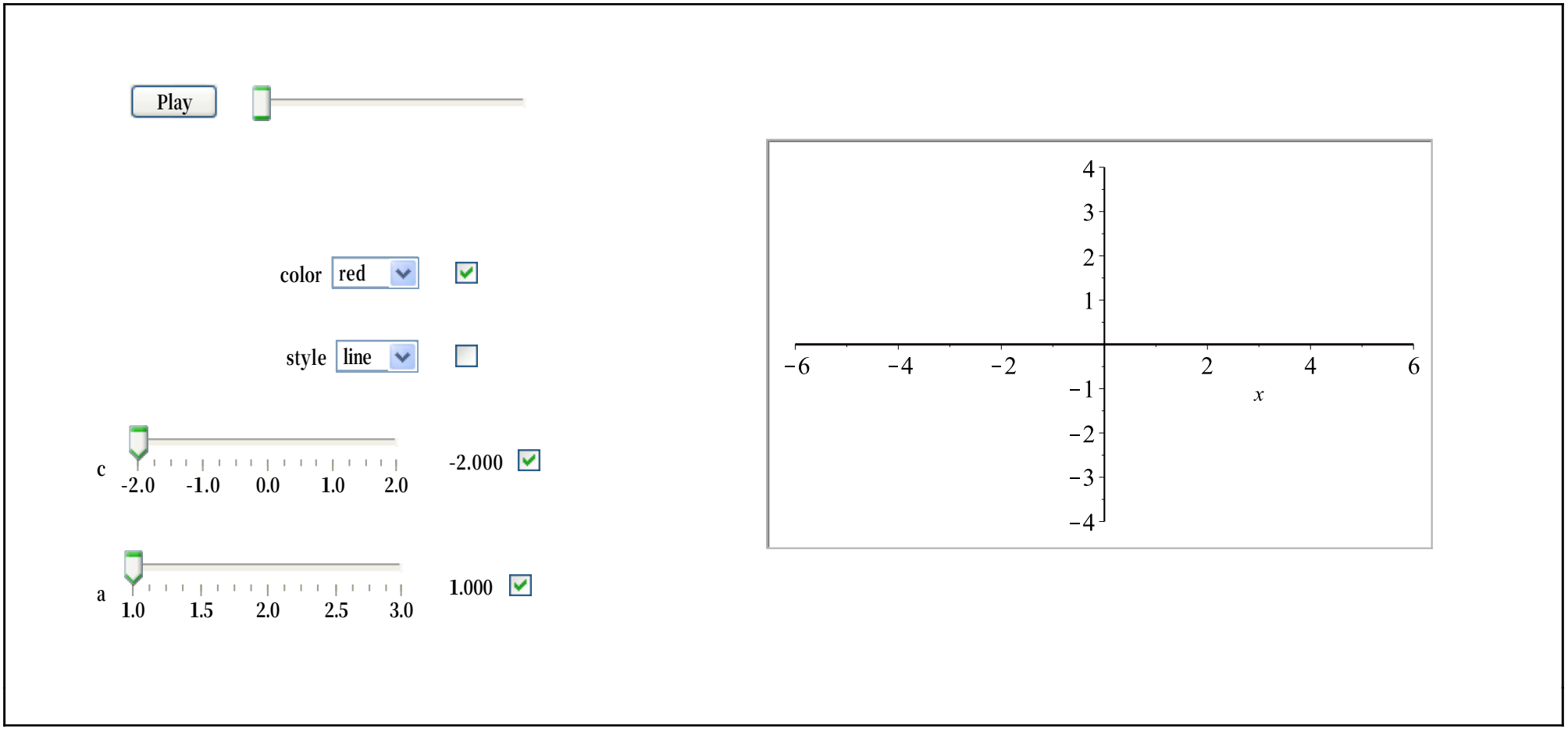
パラメータ値の各セットで生成されたフレームの表示は、アニメーションとして表示することができます。各パラメータのコントロールに、アニメーションとして再生するかどうか決めるためのチェックボックスを配置できます。このチェックボックスは、アニメーション再生中に選択解除することができます。

アニメーション再生中に選択解除した場合でも、パラメータは引き続き対話的にコントロールできる点に注意してください。

アニメーションのチェックボックスは、単一の新規デフォルトオプションとして指定できます。また、これは、ほかのパラメータで個別に指定することでオーバーライドされることがあります。アニメーションのチェックボックスの初期状態として、選択状態で表示、非選択状態で非表示、または非表示から選択できます。

デフォルトでは、パラメータで個別に指定していない限り、離散値パラメータのコンボボックスコントロールにはアニメーションチェックボックスはありません。

```
> Explore(plot(|(c − 2) (c + 2)| sin(a x), x = −6 ..6, view = −4 ..4, color = r, style = s),

                parameters = [ [r = [red, blue, green], label = color, animate = true],
                               [s = [line, point, line], label = style, animate = false],
                                c = −2.0 ..2,
                               [a = 1 ..3.0 ]],

                size = [400, 246],
                animate,
                placement = left)
```



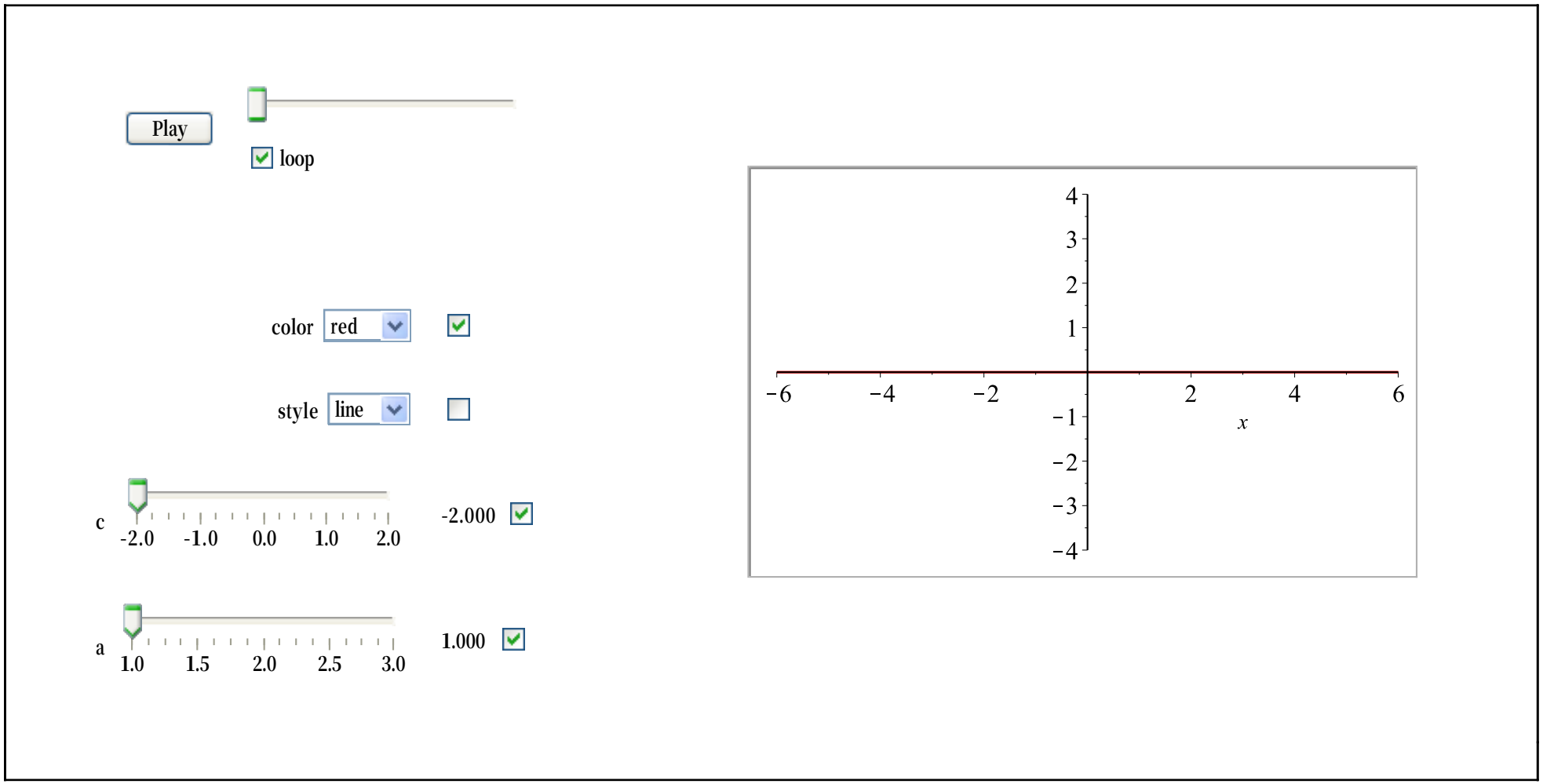
アニメーションのループ

loop オプションによって、連続のアニメーションのループ再生を行うことができます。

[loop] コントロールチェックボックスの初期状態として、完全に省略 (デフォルト)、選択状態、または非選択状態から選択できます。

```
> Explore(plot(|(c-2)(c+2)| sin(a x), x = -6..6, view = -4..4, color = r, style = s),

                parameters = [[r = [red, blue, green], label = color, animate = true],
                              [s = [line, point, line], label = style, animate = false],
```

```
                    c = -2.0 ..2,
                    [a = 1 ..3.0 ]],

 size = [400, 246 ],
animate,
loop,
placement = left)
```



対話型マーカー

範囲コントロールのペア (例 : スライダコントロール) を 2-D プロット上で対話型マーカーとして統合できます。各マーカーはプロット上で十字記号で表示されます。これをマウスポインタで対話的にドラッグできます。

デフォルトでは、マーカーパラメータと関連付けられた個別のスライダは非表示になっていますが、オプションで同時または個別に表示することができます。

DEplot および 初期条件

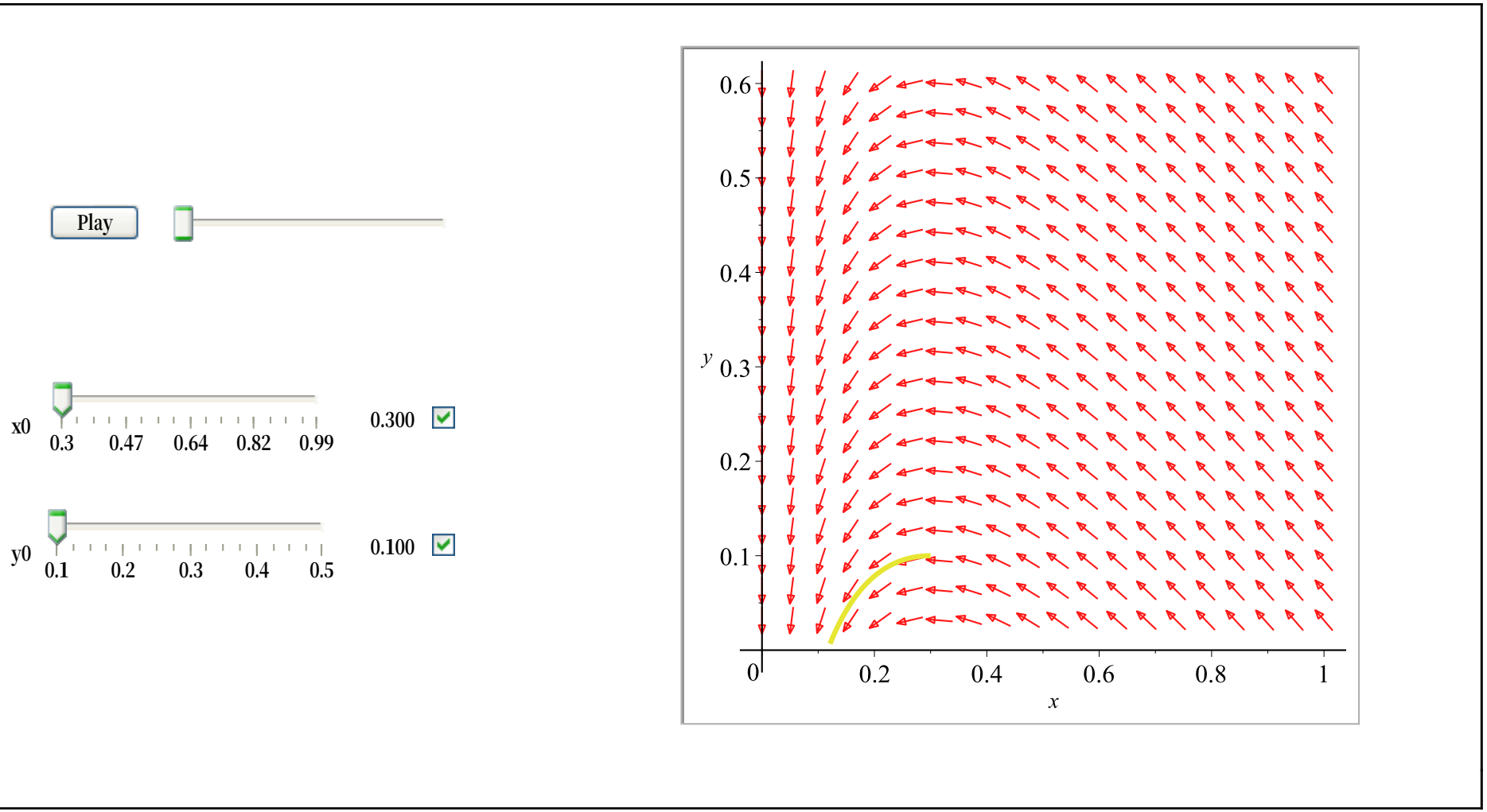
> $deS := \frac{d}{dt} x(t) = -0.5\, x(t)\, y(t)$

$$deS := \frac{d}{dt} x(t) = -0.5\, x(t)\, y(t) \qquad \textbf{(5.1.1}$$

> $deI := \frac{d}{dt} y(t) = 0.5\, x(t)\, y(t) - 0.15\, y(t)$

$$deI := \frac{d}{dt} y(t) = 0.5\, x(t)\, y(t) - 0.15\, y(t) \qquad \textbf{(5.1.2}$$

```
> Explore(DEtools[DEplot]([deS, deI], [x(t), y(t)],
                          t = 0..40, x = 0..1, y = 0..0.6,
                          [[x(0) = x0, y(0) = y0]], arrows = medium),

   parameters = [[x0 = 0.3..0.99], [y0 = 0.1..0.5]],

   showmarkercontrols,
   markers = [[x0, y0]],

   placement = left,
   animate)
```



画像

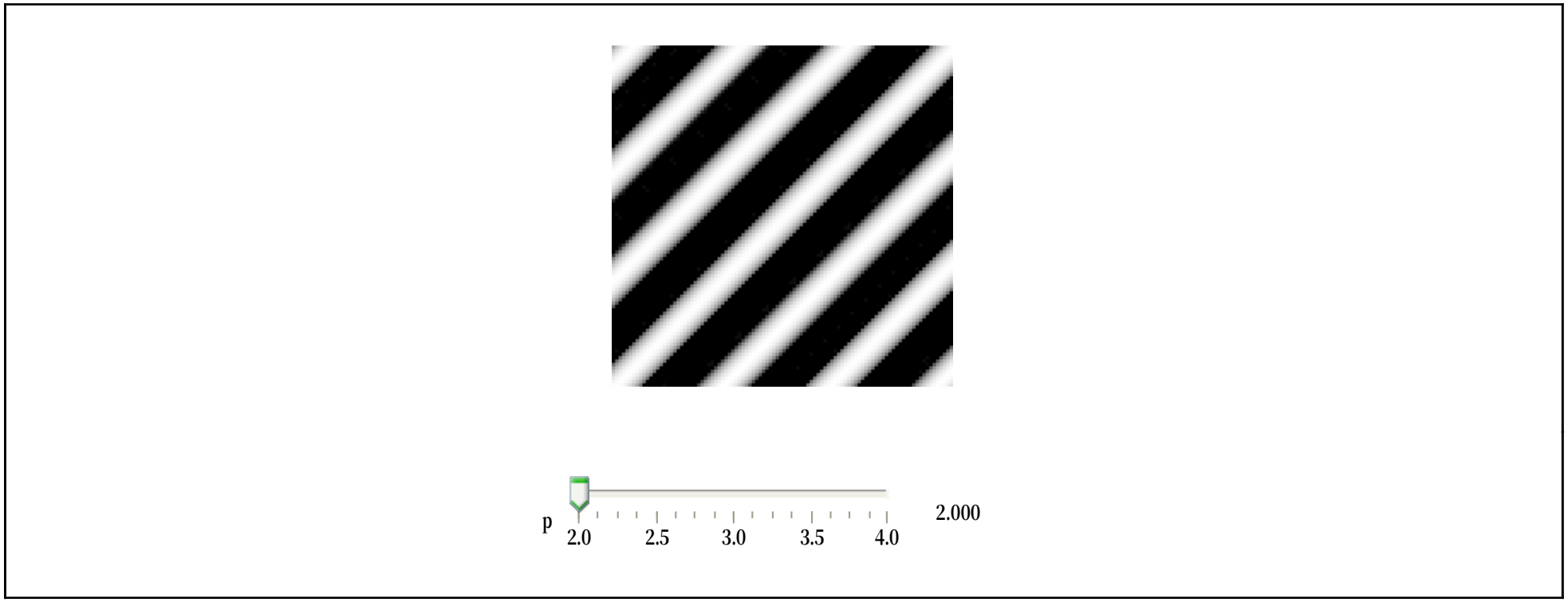
Maple 18 では、画像を返すプロシージャを対話的に調査できます。

基本的な例：float[8] Array

ImageTools パッケージでサポートされている配列は、画像として解釈することができます。

以下は、関数コールにかかわる基本的な例で、2-D datatype=float[8] 配列に評価されます。グレースケールとして解釈されます。

```
> Explore(Array(1..100, 1..100, (i,j)->evalf(sin(p*(i+j)/10)), datatype=float[8]),
                parameters=[p=2.0..4.0], size=[200,200])
```



画像を返すプロシージャ

画像を返すコマンドまたはプロシージャの関数コールを調査することもできます。

```
> img := ImageTools:-Read(cat(kernelopts(datadir), "/images/excavator.jpg")):
> Explore(ImageTools:-Threshold(img, threshold),
                parameters=[threshold=0.2..0.8], placement=right, size=[320,210])
```

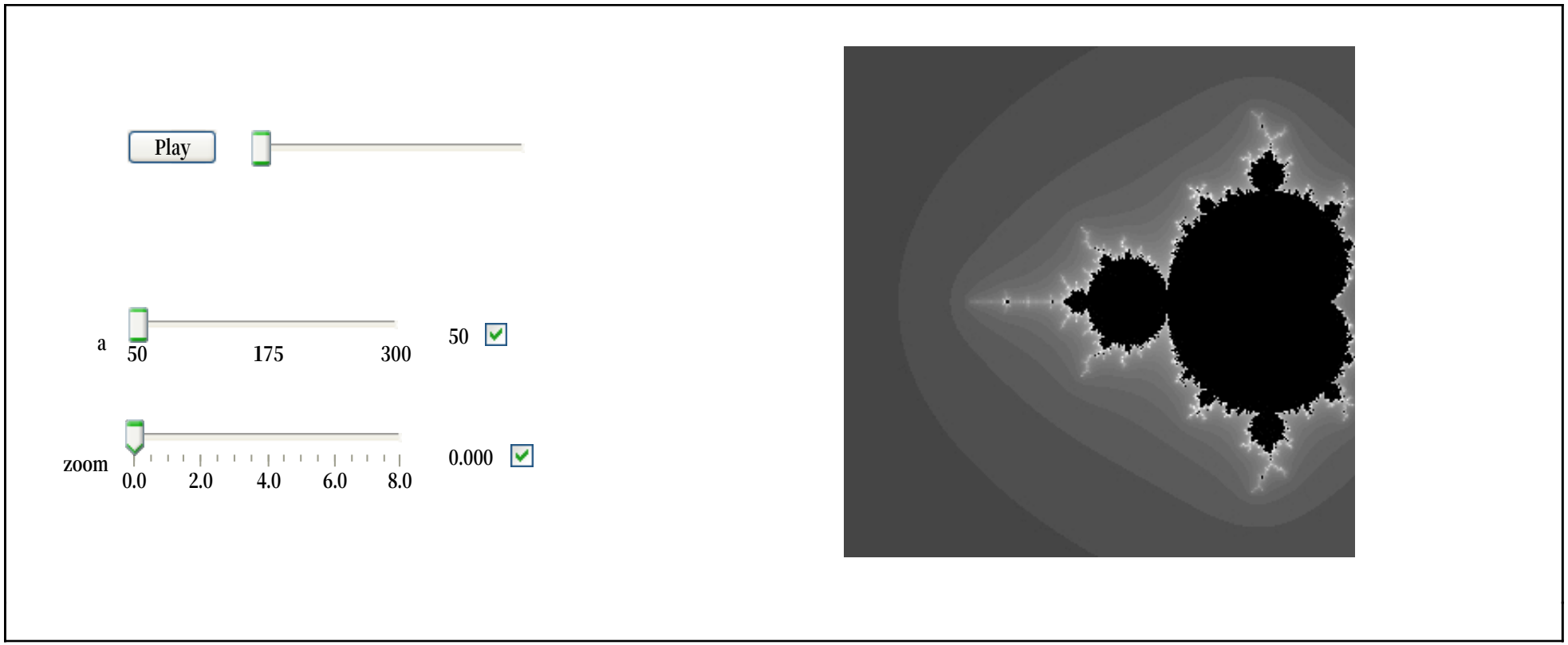
threshold 0.2 0.35 0.5 0.65 0.8 0.357

アニメーション

画像の調査をアニメーションで表示することもできます。

```
> with(Fractals:-EscapeTime):
```

```
> Explore(Mandelbrot(300, -1.78 + (-1 - 1.5*I)/exp(zoom), -1.78 + (2.2 + 1.5*I)/exp(zoom),
                     iterationlimit = a, output = layer1),
          parameters = [a = 50..300, zoom = 0.0..8.0],
          animate,
          placement = left)
```



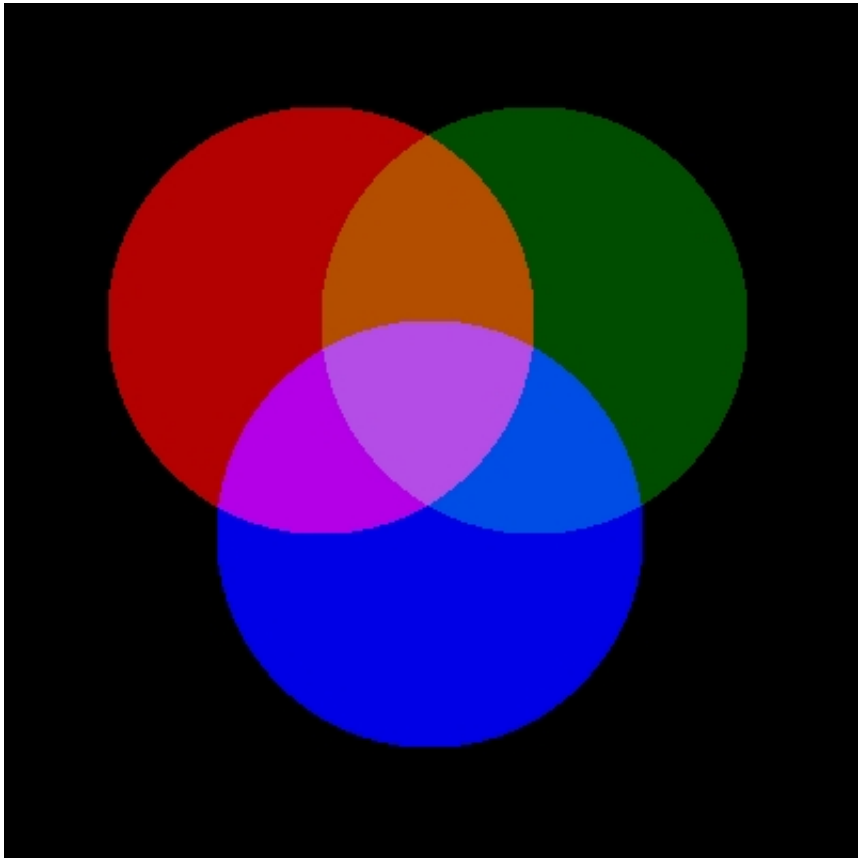
RGB 色空間：アプリケーション

次の例では、パラメータ値の変更に伴う配列の画像の効率的な直接更新を行うために、独自の初期化モジュールを使用しています。コードエディタを右クリックし、プロシージャを初期化するために [コードを実行する] を選択します。

```
#Example Code
F:=module()
    local ModuleApply,init,f,sr,sg,sb,n,n1,n2,c,tc,R;
    init:=proc()
        n:=200;  n1,n2:=trunc(n/4),trunc(n/2);
        c:=Matrix(n,'datatype'='float[8]',
                  subs(d2=n2,
                       (i,j)->`if`((d2-i)^2+(d2-j)^2<(d2)^2,1,0)));
        tc:=Matrix(n,'datatype'='float[8]');
        R:=Array(1..2*n,1..2*n,1..3,'datatype'='float[8]');
    end proc;
    f:=proc(x)
        LinearAlgebra:-MatrixAdd(tc,c,.0,x,'inplace');
        NULL;
    end proc;
    ModuleApply:=proc(tr,tg,tb)
        if not n::posint then init();  end if;
        if sr<>tr then
            f(tr);
            sr,R[n1..n1+n,n1..n1+n,1]:=tr,tc;
        end if;
        if sg<>tg then
            f(tg);
            sg,R[n1..n1+n,2*n-5*n1..2*n-n1,2]:=tg,tc;
        end if;
        if sb<>tb then
            f(tb);
            sb,R[2*n-5*n1..2*n-n1,1+n2..n+n2,3]:=tb,tc;
        end if;
        R;
    end proc;
end module:

```

F に対する単一のコールによってカラー画像が出力として生成されます。これは、現在のワークシートに埋め込むことで可視化できます。

```
> ImageTools:-Embed(F(0.7, 0.3, 0.9))
```



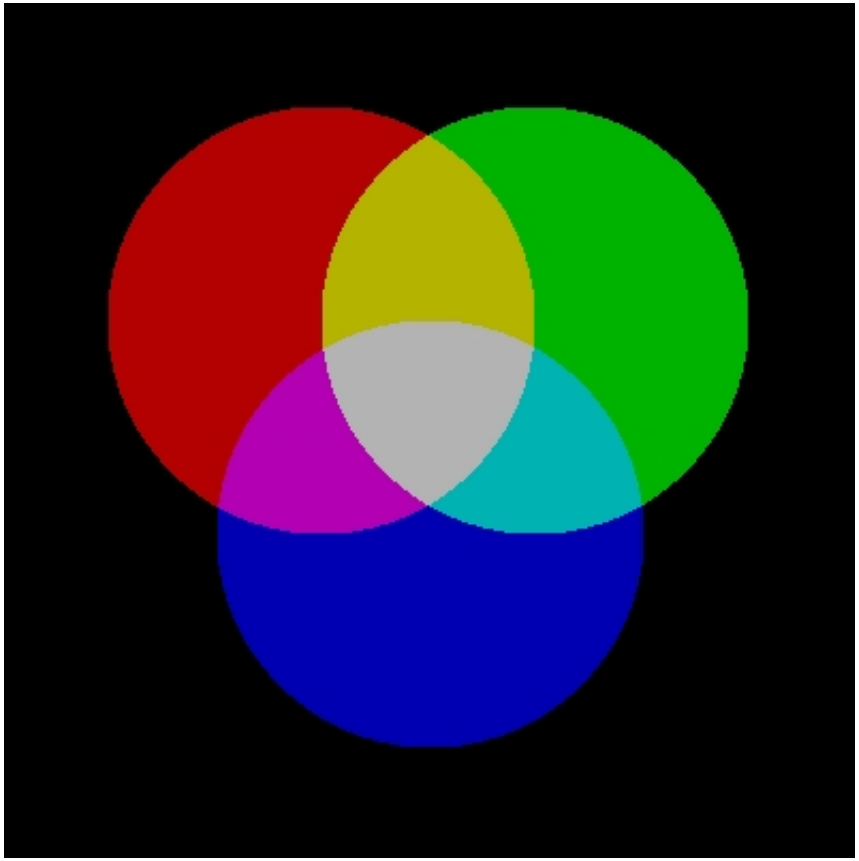
これにより、対話型のアプリケーションを簡単に構築することができます。

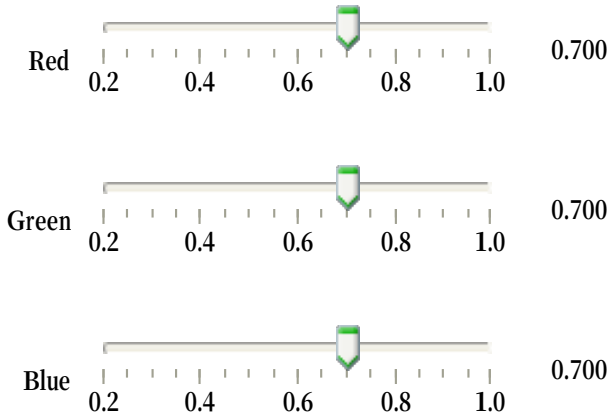
```
> Explore(F(Red, Green, Blue),

                parameters = [Red = 0.2 ..1, Green = 0.2 ..1, Blue = 0.2 ..1],

                initialvalues = [Red = 0.7, Green = 0.7, Blue = 0.7],
```

placement = *right*,

title = "The Additive RGB Color Space")

加法 RGB 色空間





参照

Explore、Explore の例、The Mobius Project

▶ Pages That Link to This Page