理化学研究所　御中

平成３０年度
InSitu 処理向け二次元可視化ライブラリの
プロトタイプ整備

システム設計書

富士通株式会社

2018 年 7 月

改定履歴

リリース	版数	備考
2018/07	1.0	初版

目次

1. はじめに

本書は、理化学研究所様向け「平成３０年度 InSitu 処理向け二次元可視化ライブラリのプロトタイプ整備」における、ソフトウエアの詳細設計書です。

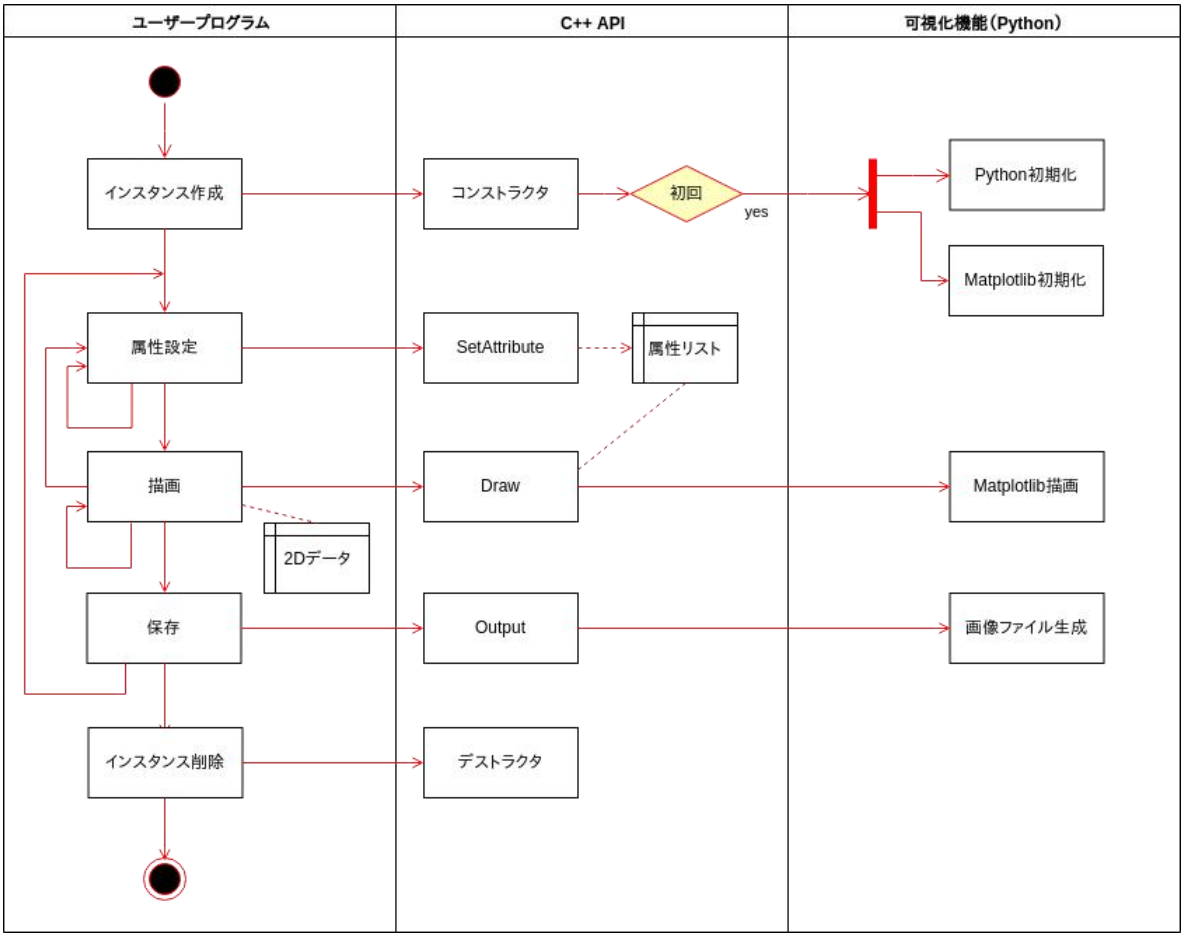
本作業では、数値シミュレーションプログラムの InSitu 可視化機能評価向けに、2D の可視化ライブラリのプロトタイプを実装します。「2D」は 3D のシミュレーションデータの格子断面または任意断面を想定したものであり、本ソフトウエアでは 2D の配列データを入力として受け取り、可視化画像ファイルを生成する機能を実装します。

2. プログラムの機能構成

本システムは、以下に示すプログラムで構成されます。

・C++ API
　ユーザープログラム(数値シミュレータ)向けアプリケーションインターフェース(C++)
・可視化機能
　2D データ配列に対する可視化画像生成機能(Python)

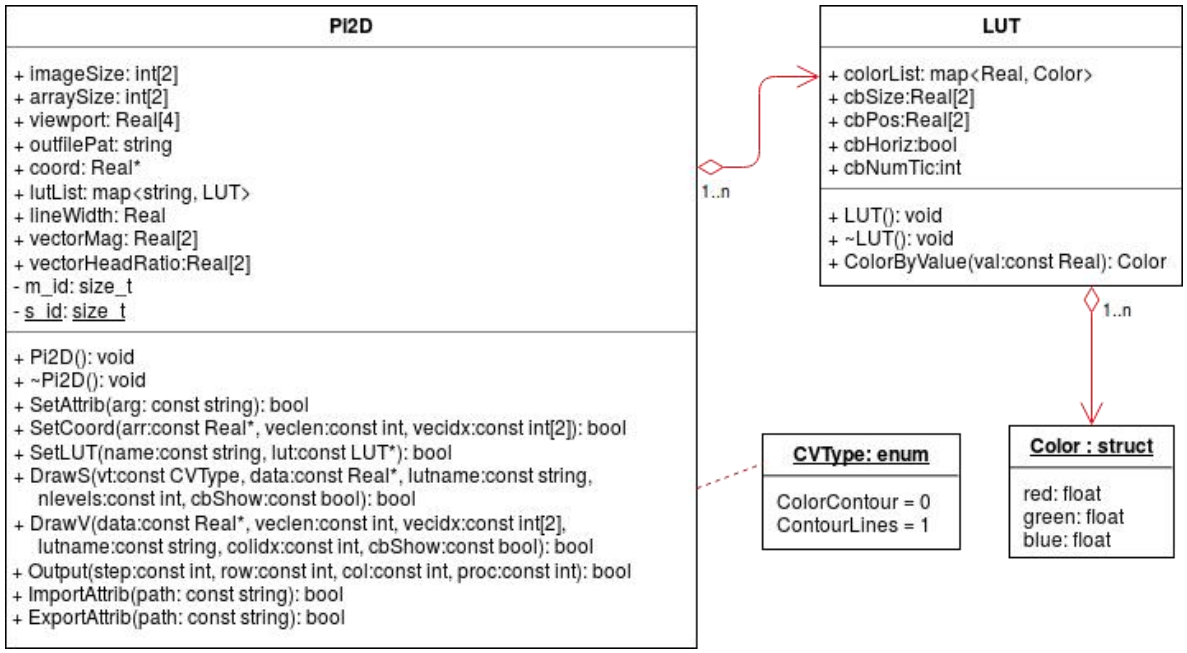
これらのプログラムはライブラリとして構成され、ユーザープログラムに結合されることで、
ユーザープログラムから機能を呼び出すことが可能となります。



3. C++ API

本機能は、ユーザープログラム(数値シミュレータ)から2D 可視化機能を利用するためのアプリケーションインターフェース(API)であり、C++クラスとして実装されます。

3.1. クラス構成



＊ 実数型の Real は、コンパイル時に double または float のいずれかに定義されます。

3.2. Pi2D クラス

Pi2D は、ユーザープログラム(数値シミュレータ)から2D 可視化機能を利用するための API 機能を実装する C++クラスです。
2D 可視化に必要な各種情報を「属性」として保持し、属性の更新と可視化処理のインターフェースを持ちます。

(1) メンバー変数

項番	名前	型	説明
1	imageSize	int[2]	生成する可視化画像のサイズ。初期値：(600, 400)
2	arraySize	int[2]	データ配列のインデックスサイズ。

			初期値：(-1, -1) ... 未設定
3	viewport	Real[4]	可視化画像の表示領域(x0,y0, x1,y1)。 初期値：(0,0,0,0) ... 設定なし(表示データに合わせる)
4	outfilePat	string	生成する可視化画像ファイルのパス名パターン。 初期値：./outimage_%S6.png 文字列中に以下の文字列が含まれる場合、数字に置き換えられる。ここで *n* は 1 個の正整数(0〜9) 「%S*n*」 タイムステップ番号。*n* は 0 埋めされる桁数 「%R*n*」 画像の Row 番号。*n* は 0 埋めされる桁数 「%C*n*」 画像の Col 番号。*n* は 0 埋めされる桁数 「%P*n*」 MPI プロセス番号。*n* は 0 埋めされる桁数 各番号の値が *n* 桁を超えた場合はその桁数となる。
5	coord	Real*	座標値属性。初期値：null … 設定なし 設定なしの場合、orig(0,0)、pitch(1,1)の座標値が仮定される。
6	lutList	map< string,LUT>	LUT(ルックアップテーブル)と名前のマップ。 初期値：空リスト
7	lineWidth	Real	ContourLines, VectorArrow 描画時の線幅。 初期値：1.0
8	vectorMag	Real	VectorArrow 描画時の倍率。初期値：1.0
9	vectorHea dRatio	Real[2]	VectorArrow 描画時のヘッド部分の割合(幅、長さ)。 初期値：(-1, -1) ... 設定なし(Matplotlib のデフォルト)
10	m_id	size_t	インスタンスの ID 番号(protected)。
11	s_id	size_t	インスタンス ID 番号のカウンタ(protected, static)。

(2) 公開メソッド

1	Pi2D	コンストラクタ	
処理	各属性の初期値を設定する。 m_id を s_id の値に設定し、s_id はインクリメントする。 最初のインスタンス作成の場合(m_id が 0)、Python/Matplotlib の初期化機能を実行する。		
引数	なし		
戻値	なし		
2	~Pi2D	デストラクタ	
処理	coord を(null でなければ)解放する。		
引数	なし		
戻値	なし		
3	SetAttrib	属性設定	
処理	座標値、LUT 以外の属性値を設定する。		
引数	arg	const string	属性設定指示を以下の形式の文字列で指定する。

			「属性名」=「値 [, 値, 値, ...]」
戻値	bool	true：成功、false：失敗	
4	SetCoord	座標値設定	
処理	座標値属性を設定する。 coord を arraySize でリアロケートし、引数 arr の値をコピーする。 arr に null が指定された場合は、coord を(null でなければ)解放し、null に設定する。 このメソッド呼び出し前に、SetAttribute で arraySize 属性を設定しておく必要がある。		
引数	arr	const Real*	座標値を格納した Real 型の配列または null を指定する。配列の長さは arraySize[0]XarraySize[1]Xveclen である必要がある。
	veclen	const int	引数 arr の成分の長さを指定する(2 以上)。省略時：2
	vecidx	const int[2]	引数 arr の成分のうち、何番目を座標値として使用するかをインデックス番号で指定する。0 以上、veclen より小さい値でなければならない。省略時：(0, 1)
戻値	bool	true：成功、false：失敗	
5	SetLUT	LUT 設定	
処理	LUT 属性を設定する。 引数 lut が null でない場合、引数 name に紐づけて(*lut)を登録する。既に name が存在している場合は上書きする。lut が null の場合、既に name が存在している場合は削除し、存在していない場合は何もしない。		
引数	name	const string	登録(または削除)する LUT の名前を指定する。
	lut	const LUT*	登録する LUT へのポインタまたは null を指定する。
戻値	bool	true：成功、false：失敗	
6	DrawS	スカラーデータの可視化描画	
処理	スカラーデータについて、カラーコンターまたはコンターラインの描画を行う。 どちらの描画を行うかは引数 vt で指定する。		
引数	vt	const CVType	可視化種別を指定する。 ColorContour：カラーコンター描画 ContourLines：コンターライン描画
	data	const Real*	可視化するスカラーデータを格納した Real 型の配列を指定する。配列の長さは arraySize[0]XarraySize[1]である必要がある。
	lutname	const string	参照する LUT の名前を指定する。省略時：””
	nlevels	const int	コンターのレベル数を指定する。省略時：10
	cbShow	bool	カラーバーを表示するかを指定する。省略時：false
戻値	bool	true：成功、false：失敗	
7	DrawV	ベクトルデータの可視化描画	
処理	ベクトルデータについて、ベクトル図の描画を行う。		

引数	data	const Real*	可視化するベクトルデータを格納した Real 型の配列を指定する。配列の長さは arraySize[0]X arraySize[1]X veclen である必要がある。
	veclen	const int	引数 data の成分の長さを指定する(2 以上)。省略時：2
	vecidx	const int[2]	引数 data の成分のうち、何番目をベクトル成分として使用するかをインデックス番号で指定する。0 以上、veclen より小さい値でなければならない。 省略時：(0, 1)
	lutname	const string	参照する LUT の名前を指定する。省略時：””
	colidx	const int	引数 data の成分のうち、何番目を色成分として使用するかをインデックス番号で指定する。-2 以上、veclen より小さい値でなければならない。-1 はベクトル成分の長さ、-2 は色参照なし(白)を意味する。省略時：-1
	cbShow	bool	カラーバーを表示するかを指定する。省略時：false
戻値	bool	true：成功、false：失敗	
8	Output	可視化画像出力	
処理	可視化画像を出力する。 outfilePat の設定と、引数 step, row, col の値に基づいてファイル名を決定し、前回の Output 実行以降に行われた Draw 結果を PNG 形式の画像ファイルとして作成する。		
引数	step	const int	outfilePat 中に「%S*n*」が含まれている場合に置き換えられる番号。省略時：0
	row	const int	outfilePat 中に「%R*n*」が含まれている場合に置き換えられる番号。省略時：0
	col	const int	outfilePat 中に「%C*n*」が含まれている場合に置き換えられる番号。省略時：0
	proc	const int	outfilePat 中に「%P*n*」が含まれている場合に置き換えられる番号。省略時：0
戻値	bool	true：成功、false：失敗	
9	ImportAttrib	ファイルからの属性値読込み	
処理	設定されている属性値のセットを引数 path で指定された JSON ファイルから読み込む。		
引数	path	const string	属性セットを読み込む JSON ファイルのパス。 絶対パスまたはカレントディレクトリからの相対パスで指定する。
戻値	bool	true：成功、false：失敗	
10	ExportAttrib	ファイルへの属性値出力	

処理	設定されている属性値のセットを引数 path で指定された JSON ファイルに出力する。ファイルが既に存在している場合は上書きする。		
引数	path	const string	属性セットを出力する JSON ファイルのパス。 絶対パスまたはカレントディレクトリからの相対パスで指定する。
戻値	bool	true：成功、false：失敗	

(3) 外部インターフェース

■ 属性リストファイル
属性リストファイルは、Pi2D の各属性値を記述する JSON 形式のファイルです。
以下の形式で記述します。

```
{
    “Pi2DAttr”: {
            # “属性名” : “属性値” [, “属性値”, ...]
            “imageSize”: [1000, 1000],
            “arraySize”: [256, 256],
            “viewport”: [-100, -100, 100, 100],
            “outfilePat”: “./outimage_%S6.png”,
            “lineWidth”: 1.0,
            “vectorMag”: 1.0
    }
}
```

3.3. LUT クラス

LUT は、ユーザープログラム(数値シミュレータ)から2D 可視化機能を利用する際に、LUT (Look Up Table)を定義するための API 機能を実装する C++クラスです。
データの値域に対して複数の{データ値：色}のペアを登録することで「カラーマップ」を定義する他、単一の{データ値：色}ペアのみを登録することで単一色の定義にも用います。

(1) メンバー変数

項番	名前	型	説明
1	colorList	map<Real, Color>	{データ値：色}ペアのマップ。Key であるデータ値でソートされる。初期値：{0.0, (1.0,1.0,1.0)}のみ登録
2	cbSize	Real[2]	カラーバーのサイズ(幅、高さ)。初期値：(0.05, 0.5)
3	cbPos	Real[2]	カラーバーの位置。初期値：(0, 0)
4	cbHoriz	bool	横方向のカラーバーかのフラグ。初期値：false
5	cbNumTic	size_t	カラーバーの目盛の数。初期値：2(最小値と最大値)

(2) 公開メソッド

1	LUT	コンストラクタ	
処理	各メンバー変数に初期値を設定する。		
引数	なし		
戻値	なし		
2	~LUT	デストラクタ	
処理	特になし		
引数	なし		
戻値	なし		
3	ColorByValue	データ値に対応する色を返す	
処理	引数 val で与えられたデータ値に対応する Color を返す。 colorList の先頭ペアのデータ値以下の値に対しては先頭ペアの Color を返し、 colorList の末尾のペアのデータ値以上の値に対しては末尾ペアの Color を返す。		
引数	val	const Real	Color を求めるデータ値
戻値	Color	val に対応する色	

(3) 注意事項

・単一色の定義

colorList に単一の{データ値：色}ペアのみしか登録されていない場合、先頭ペア==末尾のペアであるので、どのようなデータ値に対しても同一の Color が返され、結果として単一色を表すことになります。

・ Color の定義

Color は、float 型のメンバー{red, green, blue}を持つ構造体として定義します。値域はいずれも[0.0, 1.0]であり、0.0 以下の値は 0.0、1.0 以上の値は 1.0 として扱います。

・カラーバーの定義

カラーバーは、Pi2D クラスの DrawS または DrawV メソッドの cbShow 引数が true の場合に、参照する LUT(引数 lutname で指定)の設定に基づいて描画されます。

・属性リストファイルでの記述

属性リストファイル中では、Pi2DAttr 配下に以下の形式で記述します。

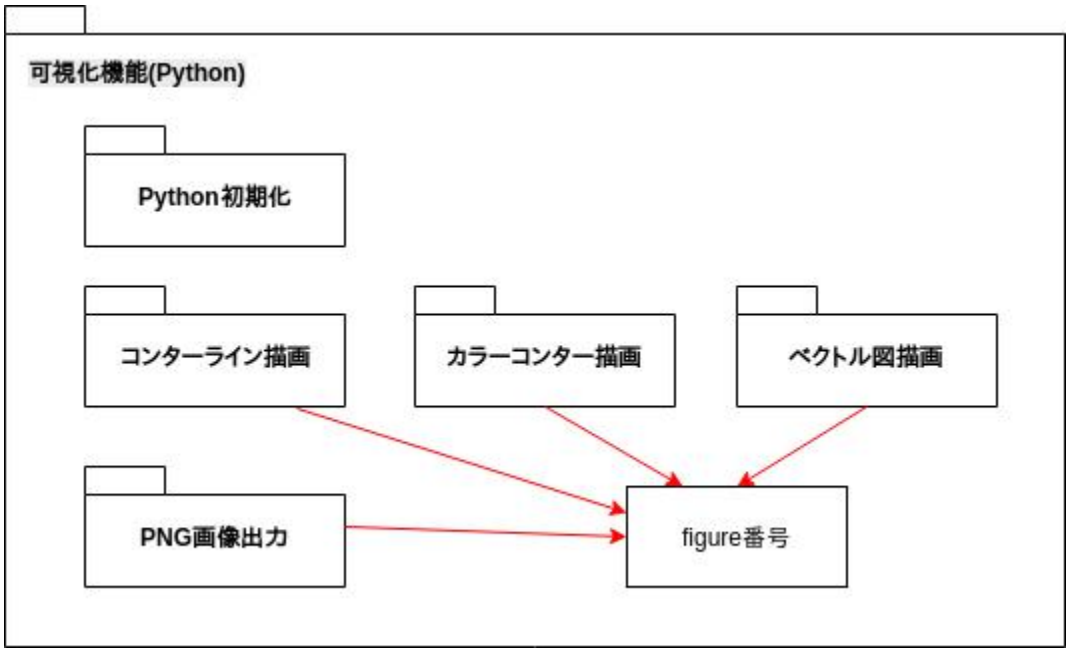
```
{
    “Pi2DAttr”: {
        “LUT”: {
            “lutName”: {
                “colorList”: [
                        [0.0, [1.0, 0.0, 0.0]],
                        [0.5, [0.0, 1.0, 0.0]],
                        [1.0, [0.0, 0.0, 1.0]] ],
                “cbSize”: [0.05, 0.5],
                “cbPos”: [0.0, 0.0],
                “cbHoriz”: false,
                “cbNumTic”: 2
            },
            “lutName2”: {    ...
            },
            ...
        },
        # 他の属性の記述
    }
}
```

”LUT”キーの値としてオブジェクトを持ち、ここでは LUT の名前をキーにして LUT クラスの属性値が記述されます。

4. 可視化画像生成機能

本機能は、与えられた２次元スカラーデータ／ベクトルデータについて、Python の Matplotlib モジュールを使用してカラーコンター、コンターラインおよびベクトル図の可視化を行うものです。
本機能は Python コードとして実装され、C++ API より libpython を介して実行されます。

4.1. 機能構成



4.2. Python 初期化

本機能は、Python および Matplotlib を Pi２D ライブラリから使用するための初期化処理を行うもので、Pi2D クラスのコンストラクタから呼出されます。以下に示す処理を実行します。

(1) Libpython の初期化(C++)

```
Py_Initialize();
```

(2) Matplotlib をヘッドレス環境で使用するための初期設定 (Python)

```
import matplotlib
matplotlib.use('Agg')
```

(3) その他、Python コードが必要とするモジュール群の import (Python)

```
import os, sys
import numpy as np
...
```

尚、本機能は Pi2D をリンクしたプログラム内で１度だけ実行されれば良いため、Pi2D クラスのコンストラクタ内で、スタティックメンバー変数 s_id を使用した実行判定を行います。

4.3. コンターライン描画

本機能は、Python および Matplotlib を使用して２D のコンターライン図を描画するもので、Pi2D クラスの DrawS メソッドから呼出されます。
本機能では、Pi2D クラスの m_id メンバーの値を Matplotlib の figure 番号として使用し、Matplotlib の pyplot.contour および pyplot.colorbar メソッドを使用して描画を行います。

4.4. カラーコンター描画

本機能は、Python および Matplotlib を使用して２D のカラーコンター図を描画するもので、Pi2D クラスの DrawS メソッドから呼出されます。
本機能では、Pi2D クラスの m_id メンバーの値を Matplotlib の figure 番号として使用し、Matplotlib の pyplot.contourf および pyplot.colorbar メソッドを使用して描画を行います。

4.5. ベクトル図描画

本機能は、Python および Matplotlib を使用して２D のベクトル図を描画するもので、Pi2D クラスの DrawV メソッドから呼出されます。
本機能では、Pi2D クラスの m_id メンバーの値を Matplotlib の figure 番号として使用し、Matplotlib の pyplot.quiver および pyplot.colorbar メソッドを使用して描画を行います。

4.6. PNG 画像出力

本機能は、Python および Matplotlib を使用して描画した figure を、PNG 形式の画像ファイルに出力するもので、Pi2D クラスの Output メソッドから呼出されます。
本機能では、Pi2D クラスの m_id メンバーの値を Matplotlib の figure 番号として使用し、Matplotlib の pyplot.figure.savefig メソッドを使用してファイル出力を行います。

5. C/FORTRAN プログラム用 API

ユーザープログラムが C または FORTRAN で記述されている場合、Pi2D クラスを直接使用することは出来ません。
本ライブラリでは、C または FORTRAN で記述されたユーザプログラムから Pi2D の C++ API を使用するためのラッパー関数を API として提供します。

5.1. 名前空間

本機能は C++で実装され、C++の「extern "C"」名前空間内で定義されます。
API の関数名は、一部の FORTRAN 処理系ではシンボル名の大文字小文字を区別しないことを考慮し、全て小文字で定義します。
また、C コンパイラではシンボル名の前に、FORTRAN コンパイラではシンボル名の前後にアンダースコアが付加されることを前提とし、関数名の末尾にアンダースコアを付加し、値参照の引数をアドレス参照に変更した関数も同時に定義します。

5.2. API

1	pi2d_init	初期化	
処理	C/FORTRAN API 用の Pi2D クラスインスタンスを作成する。 既に作成済みの場合は何もしない。		
引数	なし		
戻値	int	0: 成功, 0 以外: 失敗	
2	pi2d_finalize	使用終了	
処理	C/FORTRAN API 用の Pi2D クラスインスタンスを削除する。		
引数	なし		
戻値	int	0: 成功, 0 以外: 失敗	
3	pi2d_setatrib	属性設定	
処理	座標値、LUT 以外の属性値を設定する。		
引数	arg	char*	属性設定指示を以下の形式の文字列で指定する。 「属性名」=「値 [, 値, 値, ...]」
	arglen	int	引数 arg の長さ。
戻値	int	0: 成功, 0 以外: 失敗	

4	pi2d_setcoord	座標値設定	
処理	座標値属性を設定する。 指定された引数を全て Pi2D クラスの SetCoord メソッドに渡して呼び出す。		
引数	arr	Real*	座標値を格納した Real 型の配列または null を指定する。配列の長さは arraySize[0]XarraySize[1]Xveclen である必要がある。
	veclen	int	引数 arr の成分の長さを指定する(2 以上)。省略時：2
	vecidx	int[2]	引数 arr の成分のうち、何番目を座標値として使用するかをインデックス番号で指定する。0 以上、veclen より小さい値でなければならない。省略時：(0, 1)
戻値	int	0: 成功, 0 以外: 失敗	
5	pi2d_createlut	LUT 作成	
処理	指定された名前で LUT を作成する。 指定された名前の LUT が既に存在している場合は初期化する。		
引数	name	char*	LUT の名前を指定する。
	namelen	int	引数 name の長さ
戻値	int	0: 成功, 0 以外: 失敗	
6	pi2d_setlutcolor	LUT の色設定	
処理	指定された名前の LUT の色リストを設定する。 引数 numcolor で指定された個数の物理量と色のリストから、LUT クラスの colorList メンバーの設定を行う。		
引数	name	char*	LUT の名前を指定する。
	namelen	int	引数 name の長さを指定する。
	ncolor	int	設定する色数を指定する。
	valuelist	Real*	設定する色に対応する物理量のリストを指定する。 リストの長さは ncolor である必要がある。
	colorlist	Real*	設定する色のリストを指定する。 リストは R, G, B, R, G, B, ...の順で格納し、リストのサイズは ncolorx3 である必要がある。
戻値	int	0: 成功, 0 以外: 失敗	
7	pi2d_setlutattrib	LUT の属性設定	
処理	指定された名前の LUT の属性を設定する。 引数 numcolor で指定された個数の物理量と色のリストから、LUT クラスの colorList メンバーの設定を行う。		
引数	name	char*	LUT の名前を指定する。

	namelen	int	引数 name の長さを指定する。
	arg	char*	属性設定指示を以下の形式の文字列で指定する。 「属性名」=「値 [, 値, 値, ...]」
	arglen	int	引数 arg の長さ。
戻値	int	0: 成功, 0 以外: 失敗	
6	pi2d_draws	スカラーデータの可視化描画	
処理	スカラーデータについて、カラーコンターまたはコンターラインの描画を行う。 どちらの描画を行うかは引数 vt で指定する。		
引数	vt	int	可視化種別を指定する。 0 : カラーコンター描画、1 : コンターライン描画
	data	Real*	可視化するスカラーデータを格納した Real 型の配列を指定する。配列の長さは arraySize[0]XarraySize[1]である必要がある。
	lutname	char*	参照する LUT の名前を指定する。
	lutnamelen	int	引数 lutname の長さを指定する。
	nlevels	int	コンターのレベル数を指定する。
	cbshow	int	カラーバーを表示するかを指定する。 0 : 表示しない、1 : 表示する
戻値	int	0: 成功, 0 以外: 失敗	
7	pi2d_drawv	ベクトルデータの可視化描画	
処理	ベクトルデータについて、ベクトル図の描画を行う。		
引数	data	Real*	可視化するベクトルデータを格納した Real 型の配列を指定する。配列の長さは arraySize[0]X arraySize[1]X veclen である必要がある。
	veclen	int	引数 data の成分の長さを指定する(2 以上)。
	vecidx	int[2]	引数 data の成分のうち、何番目をベクトル成分として使用するかをインデックス番号で指定する。0 以上、veclen より小さい値でなければならない。
	lutname	char*	参照する LUT の名前を指定する。
	lutnamelen	int	引数 lutname の長さを指定する。
	colidx	int	引数 data の成分のうち、何番目を色成分として使用するかをインデックス番号で指定する。-2 以上、veclen より小さい値でなければならない。-1 はベクトル成分の長さ、-2 は色参照なし(白)を意味する。
	cbshow	int	カラーバーを表示するかを指定する。 0 : 表示しない、1 : 表示する
戻値	int	0: 成功, 0 以外: 失敗	

8	pi2d_output	可視化画像出力	
処理	可視化画像を出力する。 outfilePat の設定と、引数 step, row, col の値に基づいてファイル名を決定し、前回の pi2d_output 実行以降に行われた描画結果を PNG 形式の画像ファイルとして作成する。		
引数	step	int	outfilePat 中に「%S*n*」が含まれている場合に置き換えられる番号。
	row	int	outfilePat 中に「%R*n*」が含まれている場合に置き換えられる番号。
	col	int	outfilePat 中に「%C*n*」が含まれている場合に置き換えられる番号。
	proc	int	outfilePat 中に「%P*n*」が含まれている場合に置き換えられる番号。
戻値	int	0: 成功, 0 以外: 失敗	
9	pi2d_importattrib	ファイルからの属性値読込み	
処理	設定されている属性値のセットを引数 path で指定された JSON ファイルから読み込む。		
引数	path	char*	属性セットを読み込む JSON ファイルのパス。 絶対パスまたはカレントディレクトリからの相対パスで指定する。
	pathlen	int	引数 path の長さを指定する。
戻値	int	0: 成功, 0 以外: 失敗	
10	pi2d_exportattrib	ファイルへの属性値出力	
処理	設定されている属性値のセットを引数 path で指定された JSON ファイルに出力する。ファイルが既に存在している場合は上書きする。		
引数	path	char*	属性セットを出力する JSON ファイルのパス。 絶対パスまたはカレントディレクトリからの相対パスで指定する。
	pathlen	int	引数 path の長さを指定する。
戻値	int	0: 成功, 0 以外: 失敗	

5.3. 制限事項

本機能では、C/FORTRAN API 用の Pi2D クラスインスタンスは 1 個しか作成することが出来ません。そのため、1 つのユーザープログラムから複数の時系列可視化画像群を生成することは出来ません。