改訂日：2025/04/04

マルチフィジクスプラットフォーム JAMPAN 簡易マニュアル

原子力機構は中性子輸送計算コードや熱流動計算コードなどのシングルフィジックスコードを結合するため、マルチフィジックス・シミュレーション・プラットフォーム「JAMPAN」を開発している。JAMPAN は、HDF5 形式のデータコンテナとシングルフィジックスコードの入力作成及び出力読み取りのためのモジュールから構成されている。ユーザーは結合する計算コードに適合した入出力アクセサモジュールを使用することで、簡単に計算コードを追加・変更することができる。JAMPAN の最初の計算対象は、炉心設計コードの参照解を提供するための核熱結合計算である。現行の JAMPAN では、中性子輸送計算コード MVP と熱流動計算コード JUPITER、ACE-3D、NASCA に対応した取り扱いモジュールを実装し、結合を可能としている。ユーザーは計算規模や計算性能などに応じて熱流動計算コードを選択することが可能である。また、燃料棒の物性値の算出として FEMAXI を利用することが可能である。本報告書では JAMPAN の概要と利用方法について説明する。

1. 序論

近年の計算機性能の向上により大規模で高忠実な計算が可能になっている。マサチューセッツ工科大学(MIT)の Kord Smith 教授は 2003 年に全炉心モンテカルロ計算を提案した。このベンチマーク計算の対象は加圧水型軽水炉(PWR)の全炉心解析であり、軸方向に 100 分割、径方向に 10 分割している。それぞれの局所出力の標準偏差は 1%以下にする必要があるとしている。Smith 教授は、このような大規模な計算は 2030 年にはワークステーションを用いて実現すると予想した。Smith 教授の予想通り、計算機性能の向上によりこのような大規模な計算が近い将来実現するであろう。

このように、核計算などのシングルフィジックス計算の計算精度は、計算機の進歩に伴ってこの数十年で大幅に向上している。そこで本研究では、これらの高精度なシングルフィジックス計算を組み合わせた、大規模で高忠実なマルチフィジックス計算の実用化に焦点を当てた。炉心設計コードの妥当性確認のため、複数の物理現象の相互作用の取り扱いを可能とするマルチフィジックス・シミュレーション・プラットフォーム **J**AEA **A**dvanced **M**ulti-**P**hysics **A**nalysis platform for **N**uclear systems (JAMPAN)を開発した。JAMPAN は原子炉内の中性子、熱流動、燃料解析といったそれぞれの物理現象を取り扱うシングルフィジックスの解析コードを連結したプラットフォームである。そのため、JAMPAN 自体はシングルフィジックスやマルチフィジックスの計算機能を持たず、各シングルフィジックス解析コードを連携するための入出力読み取り機能やデータ変換機能などを備えている。

JAMPAN は Python で実装されている。JAMPAN は、HDF5 形式のデータコンテナとシングルフィジクス解析コードの入力作成及び出力読み取りのためのモジュールから構成されている。JAMPAN のコンセプトは米国のアイダホ国立研究所(INL)の MOOSE やフランスの原子力・代替エネルギー庁(CEA)の SALOME など、他のマルチフィジックス・シミュレーション・プラットフォームと同様である。

JAMPAN 開発の最初の計算対象は、炉心設計コードの参照解を提供するための核熱結合計算である。燃料集合体の複雑な体系を取り扱うことによって、より正確な流速や反応率分布を得ることが期待できる。例えば部分長燃料棒の上端部や制御棒の挿入部の複雑な熱流動挙動も考慮できるようになる。JAMPAN を用いたマルチフィジックス解析の最終目標は炉心設計コードの妥当性確認のためのモックアップテストの実施コストを削減することである。この目標を達成するため、経験式を可能な限り含まない高忠実なマルチフィジックス解析を目指している。

2. JAMPAN の特徴とインストール方法

2.1. JAMPAN で取り扱うことのできる体系

JAMPAN の最初の計算対象は核熱結合計算である。現在このプラットフォームは１つの中性子輸送計算コード(連続エネルギーモンテカルロ計算コード MVP)と３つの熱流動計算コード(多相多成分熱流動計算コード JUPITER、三次元二流体モデル解析コード ACE-3D、サブチャンネル解析コード NASCA)の連結を実装している。

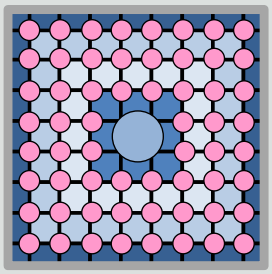
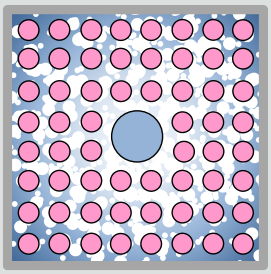
表 2.1 に核熱結合計算対象の例を示す。連結する熱流動計算コードは、計算体系の大きさや求める計算精度によって選択することが可能となっている。どのような計算体系に対しても高忠実な中性子輸送計算コードと熱流動計算コードを用いたマルチフィジックス解析を実施することが理想である。しかし、全炉心体系を JUPITER のような詳細な Computational Fluid Dynamics (CFD)計算コードで計算するには、現在のコンピューターの演算性能では十分ではない。そのため、全炉心体系のような大きな体系では、計算コストを削減するため、NASCA や ACE-3D のような平均化処理を行った熱流動計算コードを用いることを想定している。NASCA や ACE-3D はこれまでの炉心設計コードで用いられている熱流動計算コードよりも高精度な熱流動計算コードであるため、MVP/NASCA、MVP/ACE-3D の核熱連結計算によって、炉心設計コードの参照解を与えることが期待できる。なお、現バージョンの JAMPAN で取り扱える中性子輸送計算コードは MVP のみである。今後は、時間依存の解析などを対象とする予定で、時間依存の解析では多群の決定論コードが必要になってくることが予想される。そのため、将来的には GENESIS や CBZ などの多群の決定論コードを JAMPAN で連結することを計画している。

JAMPAN は HDF5 フォーマットの JAMPAN 独自のデータコンテナとアクセサモジュールをもつ。アクセサモジュールはシングルフィジックスコードの入力ファイルの生成、出力ファイルの読み取りを行うことができる。図 2.1 に JAMPAN のシステム構造を示す。 ユーザーは結合するシングルフィジックスコードを指定できる。また、入力生成、出力読込モジュールを調整すれば別の計算コードを追加することができる。コード連結時の入力生成に関しては JAMPAN が自動的に管理する。連結するコードの依存性を低減するため、出力読込モジュールがコード依存の出力フォーマットを JAMPAN 独自のフォーマットに変換する。入力生成モジュールは JAMPAN 独自のフォーマットを参照し、入力コードのデータ形式に合わせて JAMAPAN のデータフォーマットを適切に変換するため、連結するコードの出力フォーマットを気にする必要はない。例えば、ACE-3D はサブチャンネル単位の減速材の密度を出力する。JUPITER では小さな立方体のセルに分割して計算を実行して、セルごとの減速材の密度を出力する。入出力データの単位は温度(K or ℃)、重さ(g or kg)、圧力(bar, or N/m^2)、体積(cc, litter, or m^3)のようにコードに依存している。 データセットクラスはフォーマットや単位を区別するための識別子を持っている。JAMPAN はそれらのフォーマットや

単位を変換する関数を持っており、連結するコードに合わせて単位を適切に変換できる。そのためユーザーは連結するコードのフォーマットや単位を気にせずに、容易に入力生成、出力読込モジュールを導入できる。

JUPITER で単一集合体を解析する際、計算コストを鑑みると立方体のセルの 1 辺はおよそ 1mm 程度が妥当である。しかし、MVP でコード上 1mm の立方体のセルを取り扱うことは可能であるが、計算コストが大きすぎる。この問題への対応として JAMPAN では計算コスト低減のための減速材密度を平均化する関数を用意している。この平均化関数により、コードによって平均化する領域のサイズを自由に改変することができる。JAMPAN では、スクリプトによって MVP と JUPITER それぞれのセルサイズをユーザーが任意に指定することができる。

表 2.1 JAMPAN の核熱結合計算の計算対象と計算規模

	Whole core geometry	**Single assembly geometry**
Image of calculation geometry (single assembly)		
Range of statistical averages	Subchannel geometry	Faithfully reproduces the flow states
Calculation scale (The number of mesh divisions in space)	1,000,000 to 10,000,000 (Using workstation)	100,000,000 to 1,000,000,000 (Using supercomputer)
Thermal-hydraulics code	NASCA, ACE-3D	JUPITER
Neutronics code	MVP	MVP

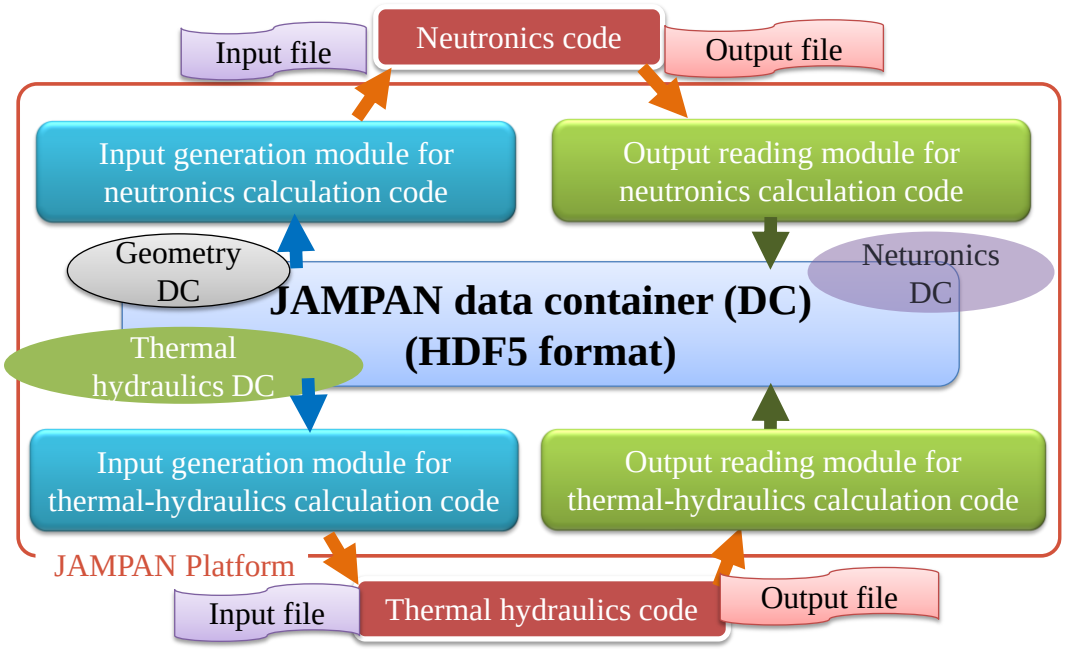


図 2.1 JAMPAN のシステム構造

2.2. JAMPAN のデータコンテナの構造

JAMPAN のデータコンテナは HDF5 フォーマットの計算グループ、時間グループ、データセットの 3 つの層で構成されている。図 2.2 に JAMPAN のデータコンテナの構成を示す。データコンテナは中性子輸送計算(核計算)、熱流動計算、燃料挙動計算といった各シングルフィジックス層のクラスを持ち、それぞれのフィジックス層は時間グループのクラスを持っている。全てのタイムステップの計算結果をコンテナに格納すると JAMPAN のデータコンテナはファイルサイズが巨大になってしまう。核熱計算では、全てのタイムステップの計算結果をデータコンテナに格納する必要はない。そこで JAMPAN では、使用するデータ容量を抑えるために、核計算で使用するタイムステップの分だけデータコンテナに格納するようにしている。タイムステップはユーザーが設定することができる。時間グループは連結計算用のインプットパラメーターに使用されるデータセットである。現バージョンでは核熱計算のために使用されるデータセットのみ含んでいる。近いうちにユーザーは JAMPAN の入力ファイルを使用してデータセットの選択ができるようになる予定である。

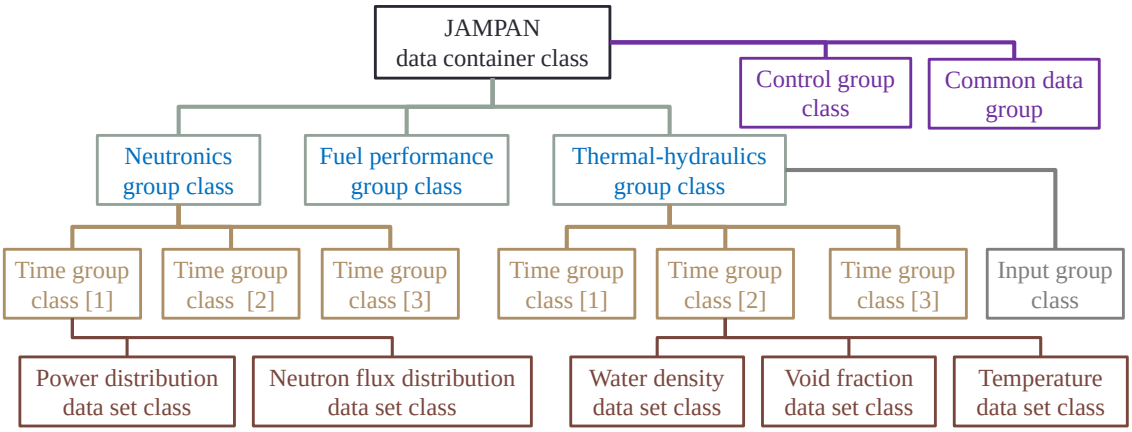


図 2.2 JAMPAN のデータコンテナの構造

2.3. JAMPAN の実行環境

JAMPAN は Python3 で実装されている。サンプルの JAMPAN 実行スクリプトが jampan/platform/interface/scripts/に用意されている。計算コードのパスの設定をすればジョブを実行することができるようになっている。別途、実行に必要な python パッケージは以下の通りである。

- Python 3.6.8 以上
- PyYAML
- NumPy (2.0 以上は未対応)
- Lark
- h5py
- Paramiko (遠隔 job 投入の場合)
- Nose

また、必要に応じて次の中性子輸送計算・熱流動計算関連の解析コードを準備する必要がある。

中性子輸送計算

- MVP

熱流動計算

- ACE-3D
- JUPITER
- NASCA

燃料挙動計算コード

- FEMAXI

なお、NASCA 関連の機能については公開版には含まれていない。NASCA 関連の機能については、ユーザーが NASCA を別途入手した後に個別に配布する。

2.4. JAMPAN のインストール方法

JAMPAN は、Python モジュールとして開発されているので、標準的な Python モジュールと同様に、ライブラリの検索パス（Python の sys.path）に 含まれていればそのまま利用することができる。

JAMPAN では、パッケージファイルを展開してできるディレクトリの直下にある"jampan"というディレクトリが Python モジュールになるようになっている。この jampan ディレクトリがある場所を Python に認識させるために、環境変数 PYTHONPATH を設定すればよい。

PYTHONPATH(必須設定)

Bourne シェル系列のシェル（例えば sh、bash 等）でホームディレクトリ直下に JAMPAN パッケージファイルを設置した場合は、ターミナル上で次のコマンドを実行すればよい。

```
export PYTHONPATH=$PYTHONPATH:$HOME
```

ログイン時に自動反映させるためには上記コマンドを次のファイルに追記すればよい

```
$HOME/.bashrc
```

C シェル系列のシェル（例えば csh、tcsh 等）では、次のコマンドを実行すればよい。

```
setenv PYTHONPATH $PYTHONPATH:$HOME
```

ログイン時に自動反映させるためには上記コマンドを次のファイルに追記すればよい

```
$HOME/.cshrc
```

Python3 を実行して import jampan が成功すればインストール完了である。

また、JAMPAN が使用する環境変数は次の 2 つである。

JAMPAN_CODE_PATH(推奨設定)

JAMPAN_CODE_PATH は設定していなくても JAMPAN は機能するが、パスの簡易化と設計方針より、上記の解析コードやライブラリ等のパスを一つにまとめることを推奨している。

Bourne シェル系列のシェルの設定例は次の通りである。

```
export JAMPAN_CODE_PATH=$HOME/local/mvp-3.0/bin/:$HOME/local/JUPITER/build/bin/
```

C シェル系列のシェルの設定例は次のとおりである。

```
setenv JAMPAN_CODE_PATH $HOME/local/mvp-3.0/bin/:$HOME/local/JUPITER/build/bin/
```

JAMPAN_WORK_PATH(自動設定)

テストケース(現状ごく一部の関数)を実行した際に利用する一時ファイルのための作業ディレクトリである。未設定の場合、JAMPAN の存在するディレクトリと同階層に大文字の JAMPAN ディレクトリが自動で生成される。

2.5. JAMPAN のディレクトリ構造

JAMPAN のディレクトリ構造を図 2.3 に示す。JAMPAN 開発ユーザーでなければ、通常はユーザーが直接アクセスする可能性があるディレクトリは主に次の 3 か所だと想定している。

- data/
- doc/
- platform/interface/scripts/

これらの内、data には基本的なインプットファイルが参考として格納されている。また、Doc には JAMPAN のマニュアルなどが格納されており、その詳細は 2.6 節に記載されている。platform/interface/scripts/は JAMPAN 実行スクリプトのサンプルが格納されている。jampan ディレクトリが PYTHONPATH で指定されていれば、JAMPAN 実行 scripts は別の場所にコピーしても実行することができる。

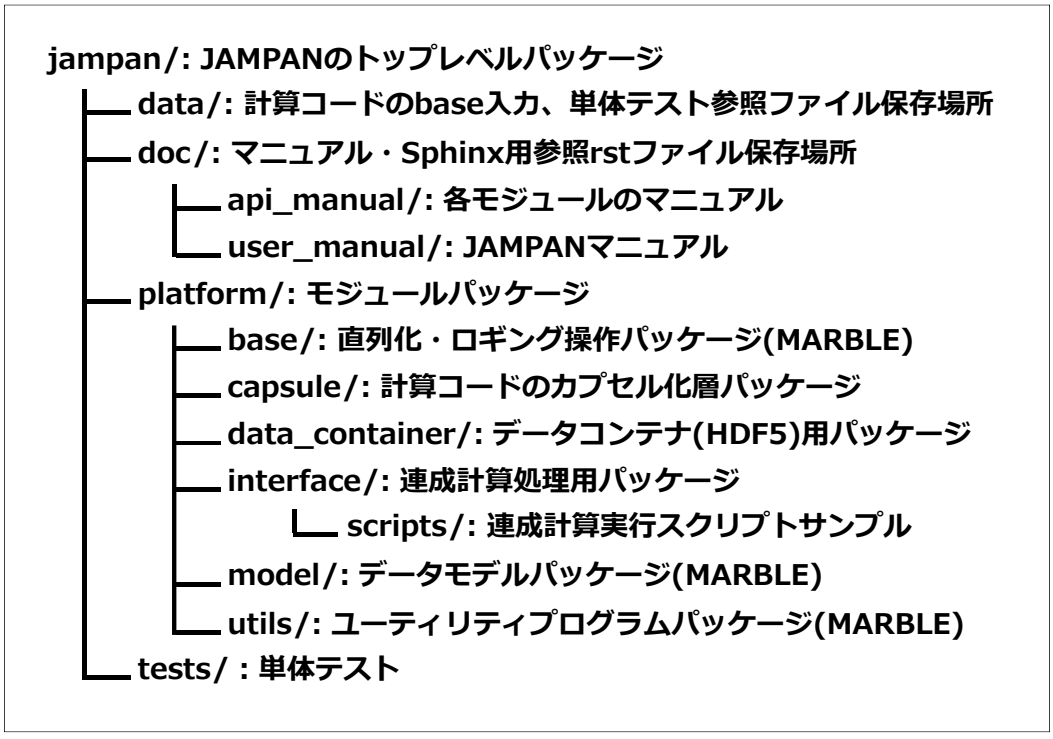


図 2.3 JAMPAN のディレクトリ構造

2.6. JAMPAN のドキュメント

JAMPAN はユーザーマニュアル(jampan/doc/user_manual/_build/html/index.html)と API マニュアル(jampan/doc/api_manual/_build/html/index.html)を用意している。マニュアルは python パッケージの Sphinx で作成しており、Web ブラウザで閲覧可能である。Web ブラウザで開いたドキュメントのサンプル画像を図 2.4 に示す。

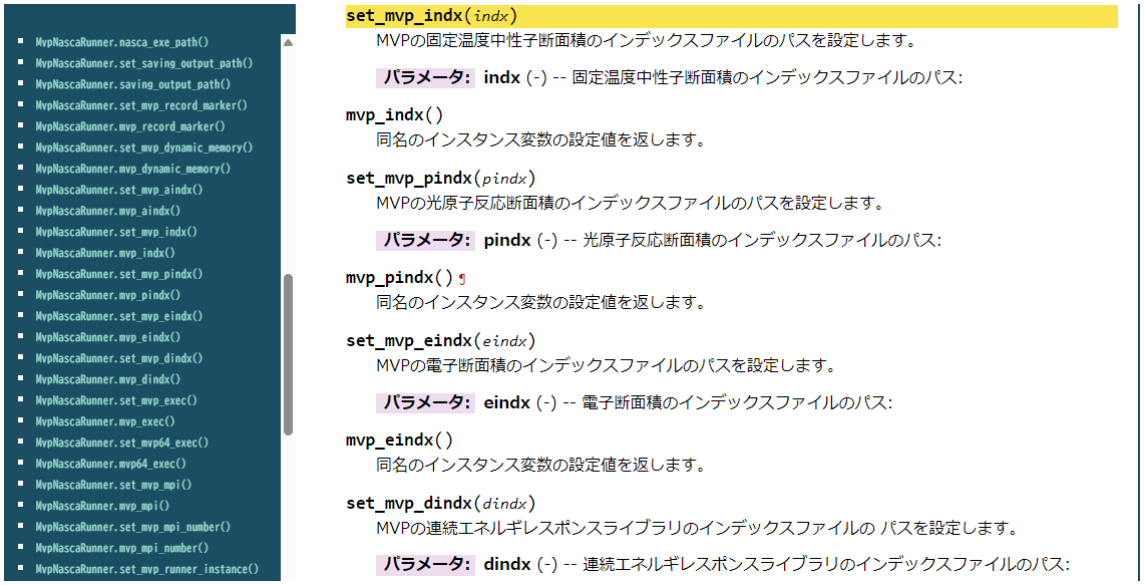


図 2.4 JAMPAN のドキュメントのブラウザ表示例