JNC TJ9400 2003-012

燃焼感度解析コードのシステム化整備

（核燃料サイクル開発機構　契約業務報告書）

２００４年２月

原子燃料工業株式会社

〒319－1184　茨城県那珂郡東海村村松４番地４９
核燃料サイクル開発機構
技術展開部　技術協力課
電話：029-282-1122（代表）
ファックス：029-282-7980
電子メール：jserv@jnc.go.jp

JNC TJ9400 2003-012
2004年2月

燃焼感度解析コードのシステム化整備

（核燃料サイクル開発機構　契約業務報告書）

巽　雅洋※，兵頭　秀昭※

要　旨

高速炉の実用化に向けて、高速炉実機炉心の核特性予測精度を向上させることは、合理的で高性能な炉心を設計してプラントの経済性の向上を図る上でも、信頼性および安全性の裕度をより高める上でも、極めて重要な研究課題となっている。

これまでの研究では、炉定数調整法を適用することにより、JUPITER等の臨界実験の成果を最大限有効に反映した統合炉定数を開発し、核設計精度の大幅な向上を達成している。一方、高速炉の炉心設計に於いては、臨界性、反応率、制御棒価値等のいわゆる静核特性だけでなく、燃焼反応度損失、増殖比といった燃焼核特性の精度良い評価も重要である。このためには、高速実験炉「常陽」等の豊富な実機燃焼データを有効に活用して、燃焼核特性の精度向上を図る必要がある。

炉定数調整法により、実機燃焼データを活用するためには、燃焼核特性の感度解析（以下、燃焼感度解析と呼ぶ）を行う必要がある。これまでに、燃焼感度解析を実施するためのコード（SAGEP-BURN）の開発が行われ、その有効性が確認されている。しかしながら、この燃焼感度の理論の複雑さと、システム上の制限から、ユーザへの負担が大きく解析作業が極めて非効率的であるという問題があった。また、システムの巨大化により機能の拡張が難しくなっているため、今後の機能拡張のために整理・統合が必要となっている。

一方、解析対象によって計算ステップが変わることや、物理的意味を分析する際には計算ステップを分解する必要があること等から、各計算機能を単純に統合するだけでは不十分である。各計算ステップは部品として保持したまま、必要に応じて部品を組み立てたり分解したりできるようにして、現在の燃焼感度解析コードをシステム化する必要がある。このため、オブジェクト指向とスクリプト言語の技術を利用して、燃焼感度解析コードのシステム化作業を実施した。

本研究では、既存のSAGEP-BURNシステムの改良すべき点を、解析プロセスの分析を通じて明らかにした。その上で、新SAGEP-BURNシステムの設計を実施し、その第一段階のプロトタイプをPythonにて実装して動作確認を行った。また、さらなる改良に向け、オブジェクト指向分析とそれに基づく設計を実施した。

※原子燃料工業（株）

JNC TJ9400 2003-012
February, 2004

Systemization of Burnup Sensitivity Analysis Code

(Document Prepared by Other Organization, Based on the Contract)

Masahiro Tatsumi※, Hideaki Hyoudou※

Abstract

To practical use of fact reactors, it is a very important subject to improve prediction accuracy for neutronic properties in LMFBR cores from the viewpoints of improvements on plant efficiency with rationally high performance cores and that on reliability and safety margins.

A distinct improvement on accuracy in nuclear core design has been accomplished by development of adjusted nuclear library using the cross-section adjustment method, in which the results of critical experiments of JUPITER and so on are reflected. In the design of large LMFBR cores, however, it is important to accurately estimate not only neutronic characteristics, for example, reaction rate distribution and control rod worth but also burnup characteristics, for example, burnup reactivity loss, breeding ratio and so on. For this purpose, it is desired to improve prediction accuracy of burnup characteristics using the data widely obtained in actual core such as the experimental fast reactor core "JOYO".

The analysis of burnup characteristics is needed to effectively use burnup characteristics data in the actual cores based on the cross-section adjustment method. So far, development of a analysis code for burnup sensitivity, SAGEP-BURN, has been done and confirmed its effectiveness. However, there is a problem that analysis sequence become inefficient because of a big burden to user due to complexity of the theory of burnup sensitivity and limitation of the system. It is also desired to rearrange the system for future revision since it is becoming difficult to implement new functionalities in the existing large system

It is not sufficient to unify each computational component for some reasons; computational sequence may be changed for each item being analyzed or for purpose such as interpretation of physical meaning. Therefore it is needed to systemize the current code for burnup sensitivity analysis with component blocks of functionality that can be divided or constructed on occasion. For this purpose, systemization of the burnup sensitivity analysis code has been done with an object-oriented scripting language.

In this study, points that should be improved in the current SAGEP-BURN were clarified through analysis of a computational process. Then a prototype of new SAGEP-BURN system was designed and implemented in Python. It is confirmed that the new system gives the identical results with that by the conventional system. For further implementation, analysis and design based on an object-oriented approach have been done.

※ Nuclear Fuel Industries, Ltd.

目　次

表リスト

図リスト

1. はじめに

高速炉の実用化に向けて、高速炉実機炉心の核特性予測精度を向上させることは、合理的で高性能な炉心を設計してプラントの経済性の向上を図る上でも、信頼性および安全性の裕度をより高める上でも、極めて重要な研究課題となっている。

これまでの研究では、炉定数調整法を適用することにより、JUPITER 等の臨界実験の成果を最大限有効に反映した統合炉定数を開発し、核設計精度の大幅な向上を達成している。しかし、高速炉の炉心設計に於いて、臨界性、反応率、制御棒価値等のいわゆる静核特性だけでなく、燃焼反応度損失、増殖比といった燃焼核特性の精度良い評価も重要である。このためには、高速実験炉「常陽」等の豊富な実機燃焼データを有効に活用して、燃焼核特性の精度向上を図る必要がある。

炉定数調整法により、実機燃焼データを活用するためには、燃焼核特性の感度解析（以下、燃焼感度解析と呼ぶ）を行う必要がある。これまでに、燃焼感度解析を実施するためのコード[1,2]（SAGEP-BURN）の開発が行われ、その有効性が確認されている。しかしながら、この燃焼感度の理論では、燃焼に伴う時間ステップを逆の方向に解いたり、物理的意味を持つ複数の項毎に計算したりする必要があるため、現在の燃焼感度解析コードシステムは、理論に合わせてユーザが計算ステップをひとつずつ追いかけながら使うようになっており、ユーザへの負担が大きく解析作業が極めて非効率的である。また、システムの巨大化により機能の拡張が難しくなっているため、今後の機能拡張のために整理・統合が必要となっていた。

一方、解析対象によって計算ステップが変わることや、物理的意味を分析する際には計算ステップを分解する必要があること等から、各計算機能を単純に統合するだけでは不十分である。各計算ステップは部品として保持したまま、必要に応じて部品を組み立てたり分解したりできるようにして、現在の燃焼感度解析コードをシステム化する必要がある。このため、オブジェクト指向とスクリプト言語の技術を利用して、燃焼感度解析コードをシステム化する作業が本契約作業である。

2. SAGEP-BURN システムの分析

本章では、SAGEP-BURN システムの概要について述べる。また、代表的な解析を対象としたプロセス分析結果に基づいて問題点を指摘し、新システムに向けた改善案を示す。

2.1 概要

SAGEP-BURN システムは、汎用性を高める目的から、複数のシステムモジュールから構成されている。提供される 17 個のシステムモジュールは、総計約 33,000 行からなる Fortran コードで記述されており、全体として巨大なシステムを構成している。表 2-1 に SAGEP-BURN のシステムモジュールの一覧を示す。原子数密度、実効増倍率、燃焼反応度損失、増殖比、制御棒反応度価値、反応率等の燃焼特性は、これらのシステムモジュールを組み合わせることにより解析することが出来る。

各システムモジュールは独立しており、それぞれに入力ファイルを指定する必要がある。また、これらのモジュールはシェルスクリプトからコントロールされる。モジュール間の情報のやりとりは、ファイル I/O を介して行われる。なお、対象とする燃焼特性ごとに計算フローも異なってくるため、必然的にシェルスクリプト内でのモジュールの制御方法も異なる。一例として、図 2-1 に増殖比に関する計算フローの一部を示す。また、付録 1 に各システムモジュールのファイル入出力の詳細について添付する。

2.2 解析プロセスの分析

問題点の抽出と改善策の立案のために、SAGEP-BURN システムを用いた解析を追試し、解析プロセスの分析を実施した。例として、増殖比に関する燃焼特性解析を採用した。

2.2.1 解析計算の環境

図 2-2 に解析環境のディレクトリ構成を示す。最上位ディレクトリ BR 以下には、3 つのディレクトリが含まれている。MakeXsec ディレクトリには、実効ミクロ定数の計算を行うための入力及びシェルスクリプトが用意される。Burn ディレクトリには燃焼感度解析に必要な実行シェルスクリプト及び入力ファイルが用意される。WORK ディレクトリでは、計算の途中で生成される中間ファイルを保存される。

2.2.2 問題点の抽出

解析プロセスの分析を通じて、以下の問題点を抽出した。

(1) システム構造に関する問題

・ システムが非常に複雑で、全体像を把握しにくい。

システムモジュールが数多くあり、それぞれが特化した機能を持っている。従って、全容を把握するためにはかなりの時間を要する。

・ モジュールが数多くあり、モジュール間の関係がわかりにくい。

解析項目によってシステムモジュールの接続関係が異なるが、なぜそのようにしなければいけないのかが理解しにくい。物理的なモデルとの関連づけが難しい。

・ 解析自体とはあまり関係のないシステムの細部についても注意する必要がある。

システム側の要求が強く、ユーザが個別に指定しなければならない入力等が多い。このような設定は、解析内容とは直接的な関連は無く、ユーザには理解されにくい。

・ モジュール間を接続する場合、ファイルの入出力関係がわかりにくい。

あるモジュールからの出力ファイルは、別のモジュールの入力ファイルとなる場合が多い。このファイルの接続関係（接続するモジュールやユニット番号）が非常に複雑である。

・ エラーメッセージが無い、あるいは分かりにくい。

入力やファイル接続設定に問題があった場合に、その原因がわかりにくい。

(2) 入出力内容に関する問題

・ ある解析計算を実施したい場合、その処理の雛形が存在しない、もしくは非常に複雑。

過去に実施したことのない新しい評価項目について解析を実施する場合、過去の知見を生かすことが出来ないため、計算環境や入力ファイル群を準備することは非常に困難であることが予想される。また、過去に解析を実施した例がある場合でも、その環境や入力を把握するには多くの努力が必要となる。

・ モジュール毎に入力を用意する必要がある。

・ 内容がほとんど同じ入力を複数用意する必要がある。

システムモジュールは独立したものであり、それぞれに入力をファイルで与える必要がある。モジュールによっては入力内容がほとんど同一の部分があり、全体的にとらえると情報の重複が見られる。計算体系を表すマテリアルマップの指定はその一例である。

同一のシステムモジュールに対しても、複数の入力ファイルを指定する必要がある場合もある。というのも、燃焼特性を評価するためには、数サイクルにわたって解析をする必要があるからである。従って、炉心サイクル毎（あるいは、さらにサイクル初期及びサイクル末期の両

方）を対象とした入力ファイルを用意することが多い。このとき、各入力のほとんどが共通で、一部分のみ（例えばサイクル名）のみが異なるという場合が多い。

・ 入力形式が固定されていて融通が利かない。

各システムモジュールに対する入力は、いわゆるフリーフォーマット形式ではなく、入力フォーマットが指定されている。従って、これらに則して入力ファイルを作成する必要がある。入力の指定方法を誤った際に計算が実行できない場合は対応が容易であるが、意図した物とは異なった計算条件で実行された場合は、その原因追及が困難である。

・ 得られた結果の取り出し方が統一されていない。
・ 結果をどの様に判断すればよいか分からない。

システムモジュールからは共通の場所から、共通のフォーマットで計算結果が取り出せることが望ましいが、必ずしもそうなっていない。例えば、あるモジュールでは6番ファイルから計算リストとして情報が与えられ、また、別のモジュールでは30番ファイルにて与えられる。

2.2.3 改善案の検討

SAGEP-BURNシステムの使いやすくするためには、2.2.2節で抽出した問題点を改善することが必要である。以下では、目指す方向性について述べるにとどめ、詳細は次章にて示す。

・ エンドユーザが操作するモジュール（エンドモジュール）を極少数あるいは一つにする。

システムモジュールが複数に分割されているのは、歴史的経緯やシステム側の制限にすぎない。エンドモジュールが少ない方がユーザへの負担が小さいことは明らかである。従ってエンドモジュールはごく少数あるいは一つであることが望ましい。

・ エンドモジュールに与える入力は一つないしは、データベースから自動的に取得する。

エンドモジュールに与える入力は少ない方がユーザへの負担は小さい。従って、エンドモジュールに与える入力は最小限とし、可能な限り一つとする。また、炉心形状等のプロジェクト間で共有できる基礎データは、データベースから必要な情報が取り出せるようにする。

・ 各炉心特性評価の解析に関する雛形をシステムに内蔵する。特別な理由が内限り、各処理の詳細はユーザが意識しなくても良い構造にする。

増殖比や燃焼反応度損失に関する感度解析など、型にはまった解析をする場合には、解析手順の雛形がシステムに内蔵されていると便利である。雛形を利用することにより、ユーザが指定すべきパラメータは最小限ですむ。

・ 各処理においてデフォルト挙動の設定を行い、すべてのパラメータの必要性をなくす。

現状ではサイクル間で共通するパラメータも毎回設定する必要があるが、共通化できるものは省略可能とする。また、システムで用意されたデフォルト値から変更がない場合は、入力自体を省略できるようにする。

・ わかりやすいエラーメッセージを提供する。

ユーザの指定した入力等に誤りがある場合には、適切なメッセージを表示するとともに必要であれば確実に実行を停止する。なお、メッセージには、"Note", "Warning", "Error"等の区分を設ける。

・ 計算結果をユーザ利用しやすい形で統一的に提供する。

解析セッションの途中あるいは終了した後に、ユーザが利用しやすい形式で情報が取り出せるようにする。また、取り出す形式もユーザ指定できるようにする。

表 2-1　SAGEP-BURN におけるシステムモジュールの一覧

ｼｽﾃﾑﾓｼﾞｭｰﾙ名	機　　能
ＳＴＡＲＴ－ＵＰ	マクロ断面積を作成する。
ＣＩＴＡＴＩＯＮ	拡散計算コード 中性子束、および随伴中性子束を求める。
ＦＩＲＥ－１	燃焼計算コード 中性子束の出力規格化因子を計算する。また、燃焼方程式を解き、原子数密度の時間変化を求めること、随伴燃焼方程式を時間的に逆に解き、随伴原子数密度を求めることが出来る。
ＦＩＲＥ－２	燃焼計算コード ＦＩＲＥ－１で計算される出力規格化因子を入力して、燃焼方程式を解き着目集合体の原子数密度を求める。
ＦＩＲＥ－３	燃焼計算コード 原子数密度の感度係数を求める時に、ＦＩＲＥ－１で計算される出力規格化因子を入力して、随伴燃焼方程式から随伴原子数密度を求める。
ＮＳ－ＩＮＩ	原子数密度の感度係数を求める時に随伴原子数密度の初期値を求める。
ＮＳ－ＩＮＩ２	実効増倍率の感度係数を求める時に随伴原子数密度の初期値を求める。
ＮＳ－ＩＮＩ３	燃焼反応度損失の感度係数を求める時に随伴原子数密度の初期値を求める。また、燃料交換による不連続性を考慮する時にも使用する。
ＮＳ－ＩＮＩ４	増殖比の感度係数を求める時に随伴原子数密度の初期値を求める。
ＮＳ－ＩＮＩ５	制御棒価値の感度係数を求める時に随伴原子数密度の初期値を求める。
ＮＳ－ＩＮＩ６	反応率の感度係数を求める時に随伴原子数密度の初期値を求める。
ＮＳ－ＪＵＭＰ	時間ステップの前後での随伴原子数密度の不連続性の計算を行い、次のステップの随伴原子数密度の値を求める。
ＲＥＦＵＥＬ	燃料交換を含めた原子数密度の管理を行う。
ＲＥＦＵＥＬ２	燃料交換を含めた随伴原子数密度の管理を行う。
ＲＥＦＵＥＬ３	燃焼反応度損失の場合に随伴原子数密度の不連続性を考慮するとともに、燃料交換を含めた随伴原子数密度の管理を行う。
ＳＣＧＩＶＥ	随伴出力Ｐ*を計算し感度係数の第２、５項（原子数密度項、出力項）を計算する。また、随伴一般化中性子束を計算するためのソース項の計算も行う。
ＳＡＧＥＰ９３	感度係数の第１、３項（直接項、中性子束項）を計算する。また増殖比と反応率の感度係数を求める場合、随伴出力、ソース項の計算を行う。

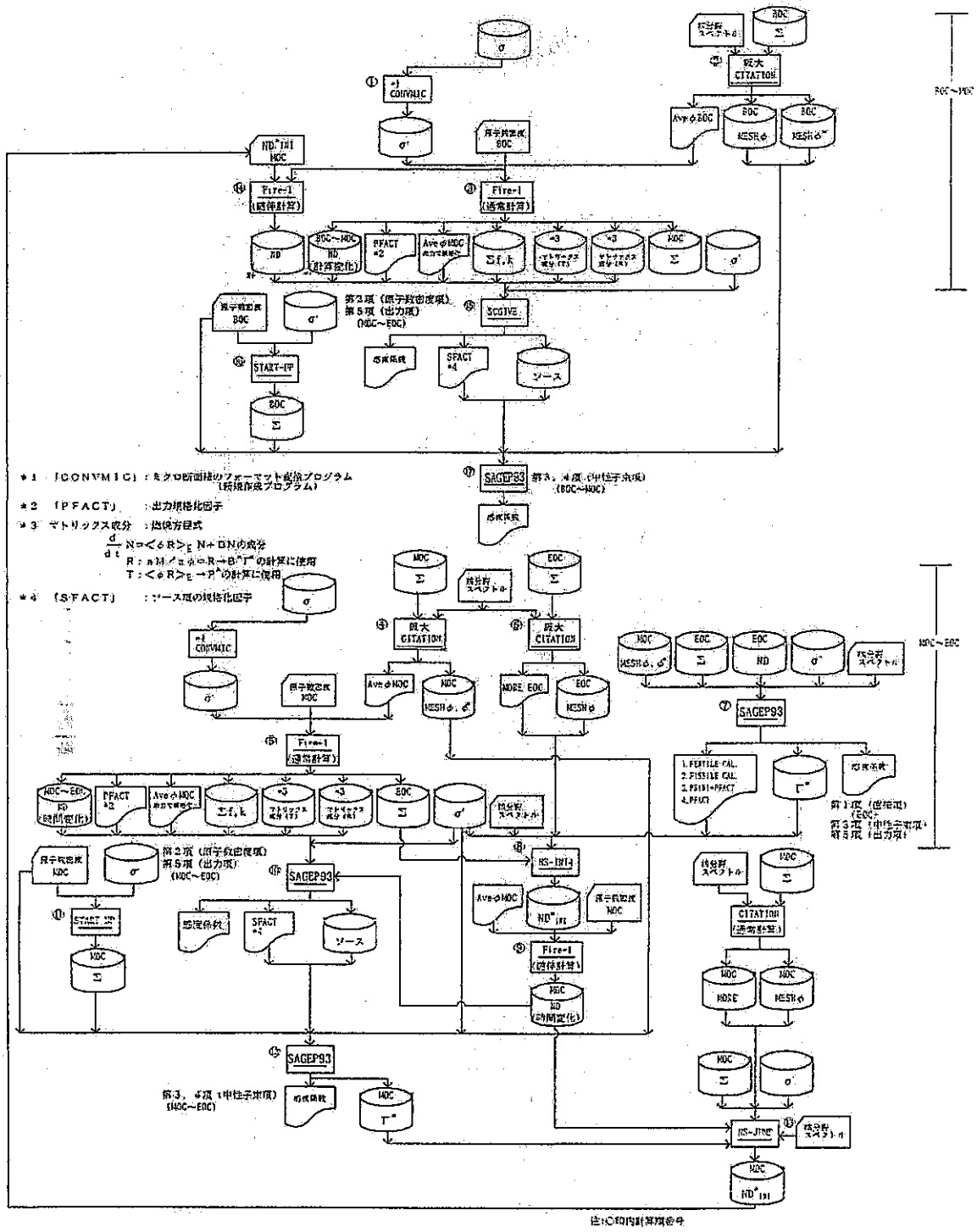


図 2-1　増殖比に関する感度係数の計算フロー（抜粋）

```
BR/
    make_Xsec
        COL18g  FLX  MICRO18g  RZ18g  RZ70g  SLMHOM  cit-burn.sh  cit.sh  jnt18g.sh  jntMKX.sh  jntcol.sh  jntflx.sh  mk_micro.sh  slarom.sh
    Burn/
        Burn-adj-debug.sh  CIT-cyc03-EOC      FIRE1_cyc04      Refuel-cyc01BOC   Refuel2-cyc04BOC      STARTUP-cyc01-EOC  ns-ini4.sh
        Burn-adj.sh        CIT-cyc03-EOC-adj  FIRE1_cyc04-adj  Refuel-cyc01EOC   Refuel2-cyc04EOC      STARTUP-cyc02-BOC  ns-jump.sh
        Burn-normal.sh     CIT-cyc04-BOC      FIRE2_cyc01      Refuel-cyc02BOC   SAGEP-snslev          STARTUP-cyc02-EOC  org.sh
        CIT-cyc01-BOC      CIT-cyc04-BOC-adj  FIRE2_cyc02      Refuel-cyc02EOC   SCG-cyc01             STARTUP-cyc03-BOC  pdsedit.sh
        CIT-cyc01-BOC-adj  CIT-cyc04-EOC      FIRE2_cyc03      Refuel-cyc03BOC   SCG-cyc02             STARTUP-cyc03-EOC  refuel.sh
        CIT-cyc01-EOC      CIT-cyc04-EOC-adj  FIRE2_cyc04      Refuel-cyc03EOC   SCG-cyc03             STARTUP-cyc04-BOC  refuel2.sh
        CIT-cyc01-EOC-adj  CONVMIC            FLXDBL           Refuel-cyc04BOC   SCG-cyc04             STARTUP-cyc04-EOC  refuel2x.sh
        CIT-cyc02-BOC      FIRE1_cyc01        NSINI4-cyc04EOC  Refuel-cyc04EOC   SGP-cyc01-flx         cit-adj.sh         refuel2y.sh
        CIT-cyc02-BOC-adj  FIRE1_cyc01-adj    NSJUMP-cyc02     Refuel2-cyc01EOC  SGP-cyc02-flx         citation.sh        refuel3.sh
        CIT-cyc02-EOC      FIRE1_cyc02        NSJUMP-cyc03     Refuel2-cyc02BOC  SGP-cyc03-flx         convmic.sh         sagep93-direct.sh
        CIT-cyc02-EOC-adj  FIRE1_cyc02-adj    NSJUMP-cyc04     Refuel2-cyc02EOC  SGP-cyc04-EOC-direct  fire1-adj.sh       sagep93-flx.sh
        CIT-cyc03-BOC      FIRE1_cyc03        PDSEDT-DATA      Refuel2-cyc03BOC  SGP-cyc04-flx         fire1.sh           scgive.sh
        CIT-cyc03-BOC-adj  FIRE1_cyc03-adj    PDSEDT-NUCL      Refuel2-cyc03EOC  STARTUP-cyc01-BOC     fire2.sh           start-up.sh
    WORK/
        DataBase         RZ70g.FT07       fire1.FT12/  fire2.FT12/  mac18g/           mic18g-cyc04-EOC/  refuel.FT13/  sagep93.FT52/
        DataBase.old     RZ70g.joint.log  fire1.FT30/  fire2.FT30/  mac70g/           micro              refuel.FT14/  scgive.FT42/
        FLUX.direct      aaa              fire1.FT31/  fire2.FT34/  mic18g-cyc01-BOC/ nsini4.FT20/       refuel2.FT11/ scgive.FT52/
        MICRO18g         citation.FT02/   fire1.FT34/  fire2.FT35/  mic18g-cyc02-BOC/ nsjump.FT20/       refuel2.FT12/ startup.FT03/
        RZ18g.FT08       citation.FT09/   fire1.FT35/  fire2.FT36/  mic18g-cyc03-BOC/ refuel.FT11/       sagep93.FT12/
        RZ18g.joint.log  citation.FT20/   fire1.FT36/  flx70g/      mic18g-cyc04-BOC/ refuel.FT12/       sagep93.FT20
```

（注） ファイル名が .sh, スラッシュ(/)で終わるものは、それぞれシェルスクリプト及びディレクトリを示す。なお、各ディレクトリには複数個のファイルが保存される。

図 2-2　解析環境のディレクトリ構成　（増殖比に関する燃焼感度解析の例）

3. 新 SAGEP-BURN システムの作成

前章において検討した改善策を実現する、新しい SAGEP-BURN システムの設計を実施し、その第一段階のプロトタイプを作成した。本章では、新システムに関する開発方針、設計及び実装について述べる。

3.1 開発方針

新システムの開発では、短期間で有効なシステム開発を行う方法として、オブジェクト指向アプローチを採用した。なお、オブジェクト指向アプローチの有用性は、これまでの検討[3,4]を通じて十分に示されているため、ここではこれ以上議論しない。

開発方法としては、効率性や保守性を高めるために、いわゆる「アジャイル*開発」を採用した。これは、反復的でインクリメントな開発サイクル[4]であり、近年注目されている手法である。この方法は、従来主流であったウォーターフォール型開発のように、要求定義、分析、実装、テストを分離されたステージとは考えずオーバーラップさせて同時進行的に実施する。開発の全期間をいくつかの区分（イテレーションと呼ぶ）に分割し、その区分内で実動するプロトタイプを作成する。そして、そのプロトタイプから得られたフィードバックを、次のイテレーションでの開発に反映する。

また、開発速度を速めると同時にソフトウェアの信頼性・保守性を高める方法として、代表的なアジャイル開発法であるエクストリーム・プログラミングのテスト戦略[5]を採用した。これは「テストファーストプログラミング」(Test First Programming)と呼ばれるもので、作成対象コードのテストプログラムを先に作成するという方法である。詳細は 3.6 節に譲るが、本手法により開発効率が高まるとともに、コードの最適化に寄与した。

3.2 記述言語の選定

SAGEP-BURN システムをオブジェクト指向アプローチですべて再実装することは非現実的である。従って、既存のシステムを活用する方式を採用する必要がある。本開発では、図 3-1 に示すように、各システムモジュールをオブジェクト化し、オブジェクト指向言語から自由に制御する方式を採用した。なお、このようにオブジェクトによるカプセル化をする際に、用いられるクラスをラッパー**・クラス(Wrapper class)と呼ぶ。なおこの手法は、デザインパターン[6,7]の観点からは、Adaptor パターンに分類される。

---

*Agile: 「俊敏な」「機敏な」の意

**包み込む意味からこう呼ぶ

図 3-1 におけるカプセル化層と制御層の記述は、単一あるいは複数のオブジェクト指向言語で記述することが出来る。例えば、カプセル化層を C++等のオブジェクト指向システム言語で記述し、制御層を Python 等のオブジェクト指向スクリプト言語で記述するといったことが可能である*。表 3-1 にカプセル化層と制御層の記述言語の組み合わせを示す。この組み合わせのうち、今回のシステム化整備で有効な方法を検討した。この際に、以下の要件を考慮した。

① カプセル化するべきシステムモジュールの数が非常に多い。
② 各システムモジュールによる実行時間は比較的長い。
③ 短期間で実働するシステムを構築する。

表 3-1 における 1 番目の方法は、すべての要件に対して不利である。3 番目の方法は、要件①に対して不利であり、かつ要件②に対してオーバースペックとなっている。また、4 番目の方法は、今回のシステム化整備には適合しない。したがって、本開発では、2 番目の方法を採用し、カプセル化層および制御層の両方をオブジェクト指向スクリプト言語にて記述することとした。

オブジェクト指向スクリプト言語の具体例としては Python[8]や Ruby[9]が挙げられるが、サイクル機構におけるこれまでの検討実績[3,4]から、Python を採用した。

3.3 開発期間

SAGEP-BURN システム全体を再構築し、単一のコードとすることが最終目標であるが、本年度中にすべての部分を完成することは困難である。したがって、下記のように 2 段階に分割して開発を実施することとした。

フェーズ 1 ： オブジェクト指向スクリプト言語によるカプセル化層の設計・実装
フェーズ 2 ： オブジェクト指向スクリプト言語による制御層の設計・実装

本年度では、フェーズ 1 における Python によるカプセル化（ラッパー）クラスライブラリの構築を実施した。これらのライブラリを用いることにより、既存のシステムモジュールを Python レベルで自由に取り扱うことが可能となる。

* オブジェクト指向システム言語とオブジェクト指向スクリプト言語の結合には、SWIG 等のスタブ生成ツールを用いることで容易に実現可能である。

3.4 設計

3.4.1 システムモジュールの制御方法

カプセル化クラスの挙動で重要なことは、内部の処理を完全に外部から隠蔽することである。言い換えると、重要なことはオブジェクトに対する入出力が重要であり、その内部挙動の実装には特に制限はない。このことから、システムモジュールの制御方法には自由度が存在する。結合度の観点から次の二通りに分類し、比較検討を行った。

(1) システムモジュールとの結合が強い場合（方法１：図 3-2(a)）

この方法では、システムモジュールの内部とカプセル化ライブラリが強く結びつけられている。システム言語とスクリプト言語の結合にはスタブを用意する必要があるが、高速のデータ転送が可能となる。換言すると、スクリプト言語からシステムモジュール内の COMMON ブロックの設定・参照が可能となる。また、システムモジュールの呼び出しも関数コールと同様になり、起動時間の観点からも有利である。しかしながら、システムモジュール部とのスタブ作成には、システムモジュール内部のデータ構造を正確に把握する必要がある。従って、システムモジュール内部のデータ構造が複雑である場合、システムモジュールの数が多い場合にはデメリットが強調される。

(2) システムモジュールとの結合が弱い場合（方法２：図 3-2(b)）

この方法では、システムモジュールを外部コマンドと同様に取り扱う。この方法の最大のメリットは、システムモジュールを修正する必要がないことである。従って、方法１より短期間での開発が可能である。システムモジュールの入出力関係は変化しないため、結果の検証も容易である。速度的には方法１には劣るが、SAGEP-BURN における各システムモジュールの実行時間は比較的長く、また実行頻度はそれほど多くない。従って、速度的な観点からはデメリットはないと判断した。

以上の検討結果から、本開発では方法２を採用することとした。

3.4.2 Python による実行イメージ

前節の方法２では、システムモジュール制御は図 3-2(b)のように、入力ファイルの準備、実行、出力ファイルの収集、結果の加工の４ステップに分割される。代表的な例として、Convmic コードを実行する場合を考えてみる。

本段階で想定した Python による実行イメージを図 3-3 に示す。最初に①において Convmic オブジェクトを生成している。これは、Convmic モジュールに対するカプセル化（ラッパー）オブジェクトである。同モジュールに対する入力ファイル(いわゆる５番ユニット)は、CardFile オブジェクトにより表現されている。（②および③） また、MicroFile オブジェクトとして抽象化さ

れた CITATION 形式のミクロ定数ファイルも指定されている（④）。これらの入力ファイルは、抽象化されたファイルオブジェクトとして Convmic に入力される（⑤）。注目すべきは、いわゆる Fortran でいうファイルユニット番号は全く指定していない点である。これらのファイルユニット番号と実際のファイルの対応関係は、自動的にシステムで管理・設定される。⑥では、与えられた入力ファイルを用いて計算を実行する。この際に、ケース名を入力している。このケース名は、ファイルの入出力関係の設定時に、補助情報として使用される。システムモジュールの実行後、得られた出力ファイル群は⑦のようにして取り出すことができる。出力ファイルは、入力ファイルの場合と同様に、オブジェクトにより抽象化されている。ファイルが複数ある場合には、ファイルオブジェクトのリストとして取り出される。また、ここで得られた出力ファイルが、異なる計算コードに対する入力ファイルとすることもできる。最後に、計算リストがある場合は⑧のように計算リストオブジェクトを取り出すことができる。

図 3-3 で示したような実行イメージを実現するためには、オブジェクトの抽象化の仕方を工夫する必要がある。具体的に述べると、図 3-4 に示すような入出力関係を適切に取り扱い、かつユーザからはファイルユニット番号等を隠蔽する必要がある。このためには、ファイル入出力関係の依存関係を自動的に解決する仕組みが重要となる。次節では、このファイルの接続関係の自動化に関する仕組みの設計について述べる。

3.4.3 ファイル入出力の管理方法

各モジュール間のファイル入出力を自動的に管理するためには、次の条件が必要となる。

- ファイルがその内容や使用目的によって分類され、ファイルタイプ名として個別に判別可能であること。
- 各モジュールのユニット番号とファイルタイプ名が対応すること
- 同一のファイルタイプ名を持つ複数のファイルが同時に与えられた場合に、その選別のための方法があること。

これらについて、図 3-5 を例に具体的に説明する。

Startup モジュールは、5 番、8 番、11 番ユニットからそれぞれユーザーカード入力、Sagep-Burn 形式ミクロ断面積、領域毎の数密度情報を読み込む。また、また実行時にマクロ断面積及び計算リストを、それぞれ 3 番及び 6 番ユニットに出力する。従って、入出力のファイル依存性の自動的な解決は、次頁の表のように、ユニット番号とファイルタイプ名をモジュール内で管理することにより実現可能である。

ユニット番号	モード	タイプ	コメント
5	Input	USER_CARD	ユーザーカード入力
8	Input	SAGEP-BURN_MICRO_XS	Sagep-Burn 形式ミクロ断面積
11	Input	NUMBER_DENSITY_REGION	領域毎の数密度情報
3	Output	MACRO_XS	マクロ断面積
6	Output	EDIT_LIST	計算リスト

上記の例は、ファイルタイプ名とユニット番号が一意に決定できる場合であるが、同一のファイルタイプ名を持つ複数のファイルが与えられる場合がある。図 3-6 では、CITATION コードの出力であるメッシュ毎の中性子束情報を含んだ二つのファイルが SAGEP コードに与えられている。これらは、通常の中性子束と随伴系での中性子束であるが、ファイルタイプ名のみでは判別できない。このような場合が発生する場合には、ユニット番号にヒント情報を付与しておき、依存関係の解決に利用する。

3.5 実装

上記設計案に基づき、フェーズ 1 の実装を実施した。これにより、SAGEP-BURN システムを Python から取り扱えるようなった。本節では、その実装内容について述べる。なお、実装は下記の環境にて実施した。

OS	Linux 2.4 (RedHat Linux)
Python	Ver 2.3.2
Fortran	Absoft Fortran 8.2

3.5.1 モジュール・パッケージ

図 3-7 にフェーズ 1 において実装した本システムの Python モジュールパッケージを示す。パッケージには、1. 設定ファイル及び汎用関数の定義、2. ファイル及びシステムモジュールのラッパークラス定義(本体部)、3. 各種テスト、の 3 つに分類される。表 3-2 に、テストクラス以外の実装済みクラス一覧を示す。

3.5.2 クラス構造

図 3-8 に本システムに関するクラス図を示す。図中における最下層に、SAGEP-BURN の各システムモジュールに対応するラッパークラスが定義されている。付録 2 に、本体部分のソースリストを添付する。

すべてのオブジェクトは、SerializableAndVisible クラスを継承している。同クラスでは、Python

のイントロスペクション機能を用いて、オブジェクト内部構造を表示する show()メソッドを定義している。

GeneralFile クラス及び DataFile クラスでは、ファイルを抽象化するためのクラスであり、ファイル内容、絶対パス、ファイルタイプ、その他付加情報の管理と、ファイル生成、読み込み、書き出し、削除等の動作を行う。

CardFile は、システムモジュールへの入力カード(5 番ユニットへの入力ファイル)を作成するためのクラスであり、内部的に FortranFormat 及び FortranLine クラスを使用している。

RefFile は、基本的に DataFile と同一であるが、書き出しやファイル削除等は制限されている。ファイルを読み込み専用で取り扱いたい場合に便利である。

SagepMicroFile および MicroFile においては、対応するファイルタイプ名が定義されている。これらのクラスを用いることにより、定数ファイルを抽象化して取り扱うことが出来る。

DataBase クラスは、CITATION や SAGEP の 51 番ユニットで使用されるデータベースファイルを抽象化したものである。データベース内には、図 3-9 に示すようにエントリー名とデータの組が格納されている。DataBase クラスは、データベースからの値の取り出しや、データベースへの値の設定を実施する。通常、CITATION 等のシステムモジュールは、計算を実行した後にデータをファイル末尾に追加する。しかしながら、この際にすでに同一エントリーブロックが存在するかどうかは考慮しない。従って、異なる値を持つ同一エントリー名のデータセットが複数存在することを許容している。データの一貫性の観点からは、このような動作は望ましくないため、計算終了時にデータベースに追加されたエントリーブロックがすでに存在する場合、ブロック内容を上書きするようにしている。なお、この仕組みは次に述べる ExecutableFile において実装されている。

ExecutableFile オブジェクトは、Fortran で記述されたシステムモジュールのラッパークラスの基底クラスである。ファイルオブジェクトの入力指定、実行、出力ファイルや計算リストに関するオブジェクトの取り出しに必要なメソッドが定義されている。また、ユニット番号とファイルタイプを関連づける defineio()メソッドも定義されている。図 3-10 に、Sagep クラスにおける defineio()メソッドの使用例を示す。

3.6 検証

3.1 節において述べたように、本システムは「テストファーストプログラミング」(Test First Programming)の考え方に基づき開発を行った。すなわち、まずテスト対象のクラスオブジェクトの挙動を定義したテストコードを記述し、そのテストが通るように本体を実装した。従って、テストコードは、すべてのクラスに対して用意されている。なお、これらのテストコードの記述にはテスティングフレームワークである PyUnit を使用した。付録 3 に、テストコードのソースリストを添付する。

テストコードは、ユニットテスト及び結合テストの２種類に分類される。前者は、いわゆる単体テストであり、当該クラスにおける各メソッドの挙動をテストするものである。後者は、ファイル受け渡し等のオブジェクト間の関係をテストする物である。

これらのテストにおいて、動作確認用データとして、実施済みである増殖比に関する燃焼感度解析における入力ファイル、出力ファイル、計算リストを利用した。

3.6.1 ユニットテスト

表 3-3 にユニットテストの一覧を示す。各ユニットテストのためのクラスは、テスト対象クラスの名前の後に"Test"が追加されている。各システムモジュールのテストの際には、入出力ファイル及び計算リストが参照計算のものと一致することを確認した。

3.6.2 結合テスト

表 3-4 に結合テストの一覧を示す。これらは、増殖比の感度解析のための Forward 計算及び Adjoint 計算に関するものであり、ファイルの接続関係をテストしている。表 3-5 に、結合テストにおける計算内容について示す。

これらの結合テストの各計算段階で、本システムによる計算結果は、参照計算のものと完全に一致することを確認した。

表 3-1　カプセル化層と制御層におけるオブジェクト指向言語選択の組み合わせ

ケース	記述言語		特徴
	カプセル化層	制御層	
1	システム言語	システム言語	高速な動作が期待できる反面、データ構造が貧弱なため、自由度が少ない。高速性が最重要視される場合に選択する。Fortran との結合部分の実装が必要。
2	スクリプト言語	スクリプト言語	スクリプト言語によるカプセル化部オーバーヘッドを十分無視できる場合に有利。スクリプト言語の豊かなデータ構造を活用することができる。
3	システム言語	スクリプト言語	1 と 2 のハイブリッドで、ある程度の高速性が必要な場合に選択する。システム言語とスクリプト言語部分の結合部分の実装が必要。
4	スクリプト言語	システム言語	採用はそれほどないが、既存のシステムの中にマクロ言語としてインタープリタを組み込む場合などに利用される。

システム言語：Smalltalk, C++ 等

スクリプト言語：Python, Ruby 等

表 3-2　今回実装したクラスの一覧　（テスト向けクラスを除く）

クラス名	機能
SerializableAndVisible	オブジェクトの内部状態の表示
FortranFormat, FortranLine	フォートラン形式書式によるプレーンテキストの読み書き（ScientificPython[10]より引用）
GeneralFile	ファイル操作に関する基底クラス
DataFile	データファイルの取り扱い
CardFile	ユーザカード(fort.5)の取り扱い
ExecutableFile	ロードモジュール・ラッパーのための基底クラス
Convmic	Convmic モジュールのラッパー
Refuel	Refuel モジュールのラッパー
Citation	Citation モジュールのラッパー
Fire1	Fire1 モジュールのラッパー
Fire2	Fire2 モジュールのラッパー
PDSHandler	PDS を取り扱うモジュールのための基底クラス
PDSEdit	PDSEdit モジュールのラッパー
Flux	Flux モジュールのラッパー
Sagep	Sagep93 モジュールのラッパー
_Nsini	Ns 系モジュールのための基底クラス
Nsini4	Nsini4 モジュールのラッパー
Nsjump	Nsjump モジュールのラッパー
Refuel2	Refuel2 モジュールのラッパー
Scgive	Scgive(scgivebr)モジュールのラッパー
Startup	Startup モジュールのラッパー

表 3-3　ユニットテストのためのクラス一覧

ファイル名	テスト対象クラス
AllTest.py	下記のテストをすべて実行
CardFileTest.py	CardFile
CitationTest.py	Citation
ConvmicTest.py	Convmic
DataBaseTest.py	DataBase
DataFileTest.py	DataFile
ExecutableFileTest.py	ExecutableFile
Fire1Test.py	Fire1
Fire2Test.py	Fire2
Nsini4Test.py	Nsini4
NsjumpTest.py	Nsjump
PDSEditTest.py	PDSEdit
Refuel2Test.py	Refuel2
RefuelTest.py	Refuel
SagepTest.py	Sagep
ScgivebrTest.py	Scgivebr
StartupTest.py	Startup

表 3-4　結合テストのためのクラス一覧

ファイル名	テスト内容
BR-foward-cycle1.py	増殖比に関する Forward 計算
BR-adjoint-cycle4.py	増殖比に関する Adjoint 計算

表 3-5　増殖比の感度解析のための Forward 計算時の実行ケース（サイクル 1 のみ）

実行モジュール	ケース
CONVMIC	第 1 サイクル、BOC
REFUEL	第 1 サイクル、BOC
STARTUP	第 1 サイクル、BOC
CITATION	第 1 サイクル、BOC、Forward 計算
CITATION	第 1 サイクル、BOC、Adjoint 計算
FIRE1	第 1 サイクル
FIRE2	第 1 サイクル
REFUEL	第 1 サイクル、EOC
STARTUP	第 1 サイクル、EOC
CITATION	第 1 サイクル、EOC、Forward 計算
CITATION	第 1 サイクル、EOC、Adjoint 計算

表 3-6　増殖比の感度解析のための Adjoint 計算時の実行ケース（サイクル 4 のみ）

実行モジュール	ケース
PDSEDIT	第 4 サイクル、EOC
SAGEP	第 4 サイクル、EOC
NSINI4	第 4 サイクル、EOC
REFUEL2	第 4 サイクル、EOC
FIRE1	第 4 サイクル、EOC
SCGIVEBR	第 4 サイクル、EOC
PDSEDIT	第 4 サイクル、BOC
SAGEP	第 4 サイクル、BOC
REFUEL	第 4 サイクル、BOC
NSJUMP	第 4 サイクル、BOC

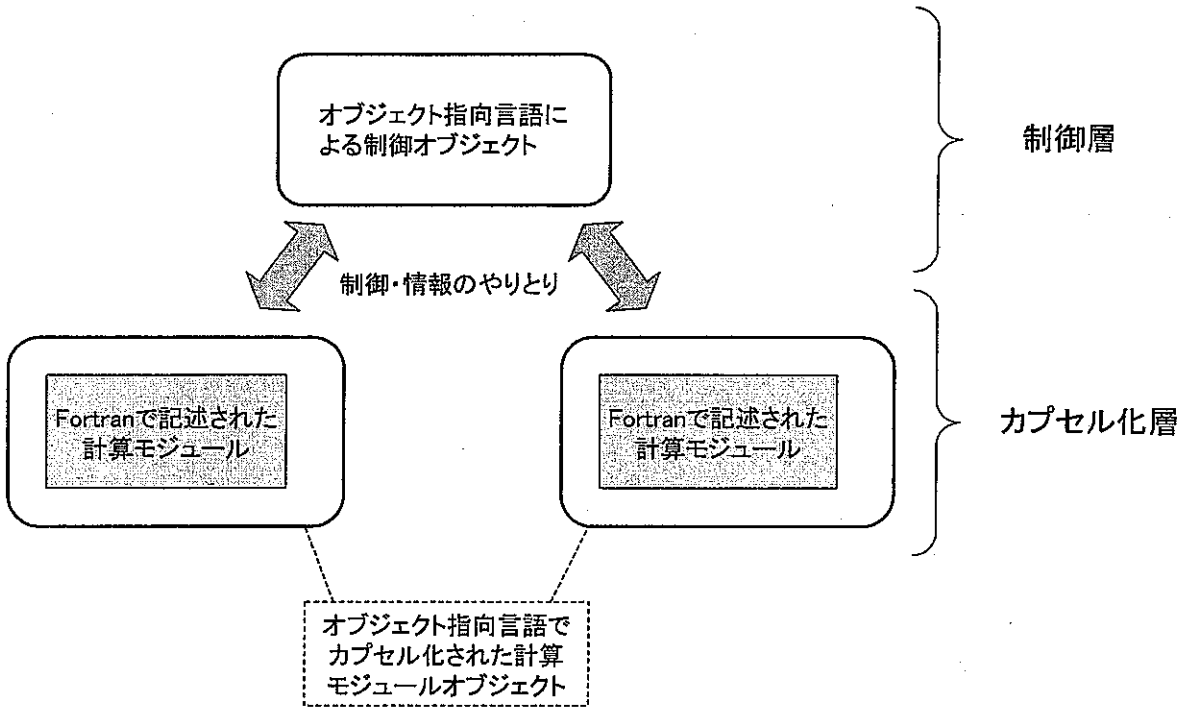
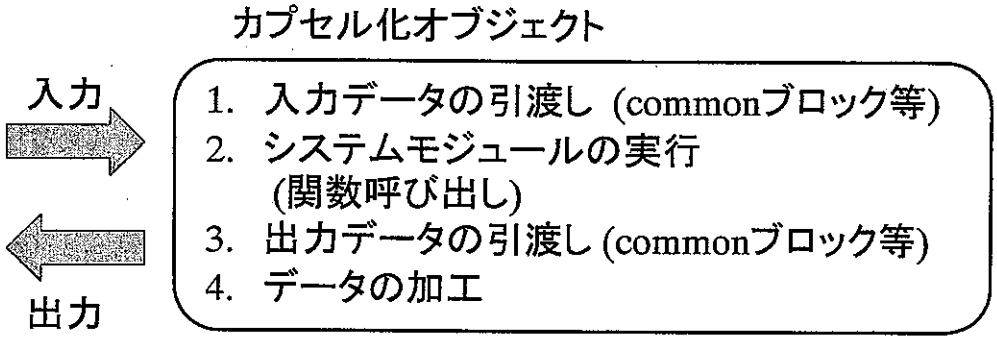
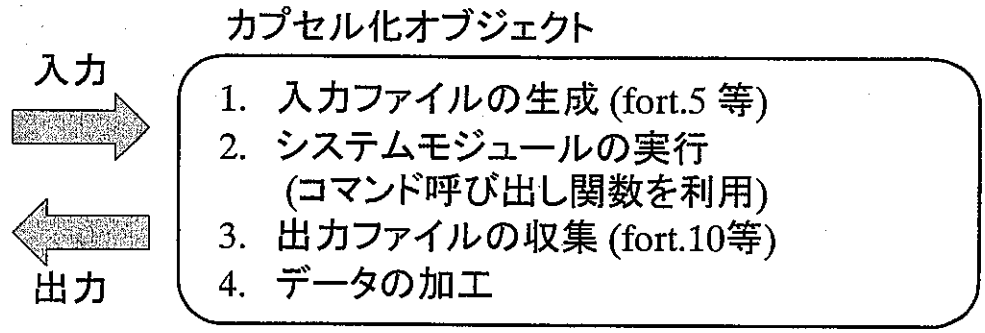


図 3-1　オブジェクト指向言語による Fortran システムのカプセル化と制御



(a) システムモジュールとの結合が強い場合 (方法 1)



(b) システムモジュールとの結合が弱い場合 (方法 2)

図 3-2 システムモジュールの制御方法に関する分類

```
convmic = Convmic()                                                    # ① Convmic オブジェクトの生成

convmic_card = [                                                       # ② Convmic への入力カードイメージ
      "    33",
      "  925  926  937  928  939  948  949" +
      "  940  941  942  951  950  953  962",
      "  963  964  965  854  884  894  814" +
       "     8   11   24   25   26   28   42",
      "  603   74  105  115    6"
      ]
convmic_inp = CardFile( convmic_card )                                 # ③ Convmic への入力ファイルオブジェクトの生成

cit_micro_file = MicroFile( micro_dir_prefix + "/MICRO18g" )           # ④ Citation 形式のミクロ定数ファイルオブジェクトの生成

convmic.inputs( [convmic_inp, cit_micro_file] )                        # ⑤ 入力ファイルの指定
convmic.run( "cyc01-BOC" )                                             # ⑥ 計算の実行
icro_file = convmic.outfiles()                                         # ⑦ 出力ファイルオブジェクトの取得
list_file = convmic.list()                                             # ⑧ 計算リストオブジェクトの取得
```

図 3-3　Python による実行イメージ

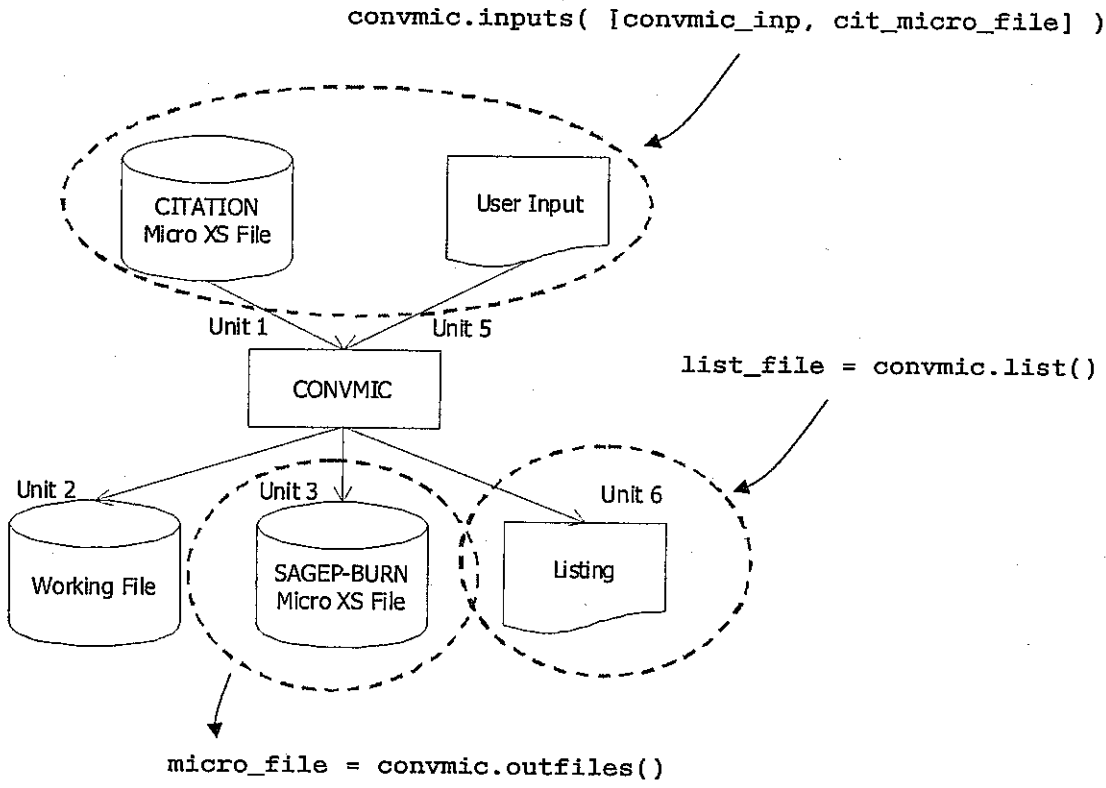


図 3-4　Fortran モジュールに対するファイル入出力と Python コードの対応

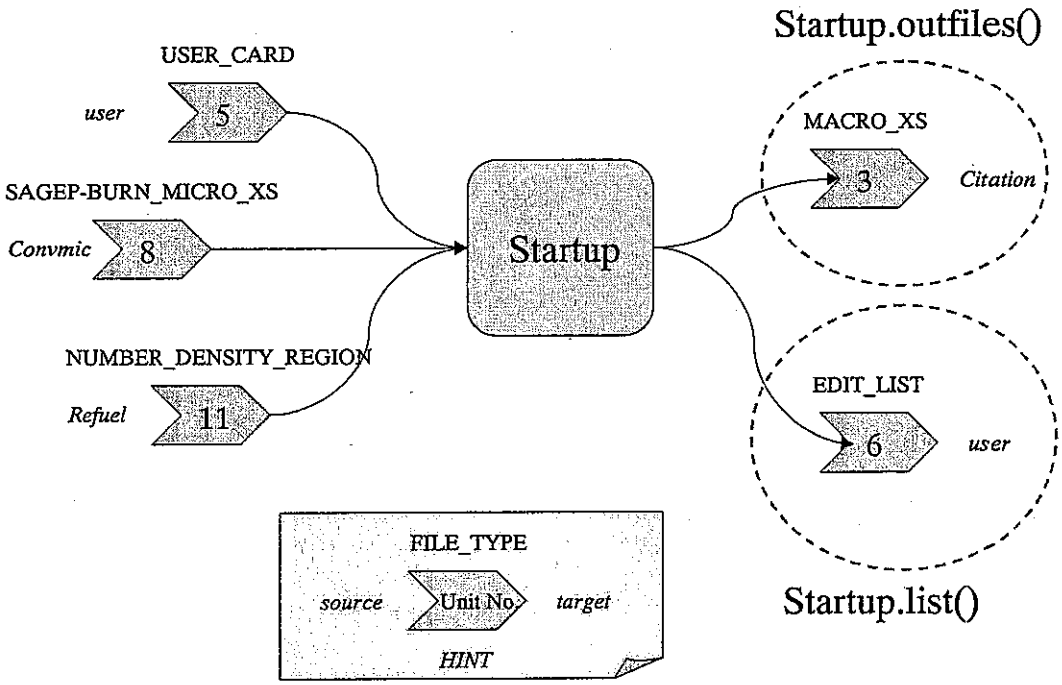


図 3-5　ファイル入出力の管理方法の例（１対１の対応関係）

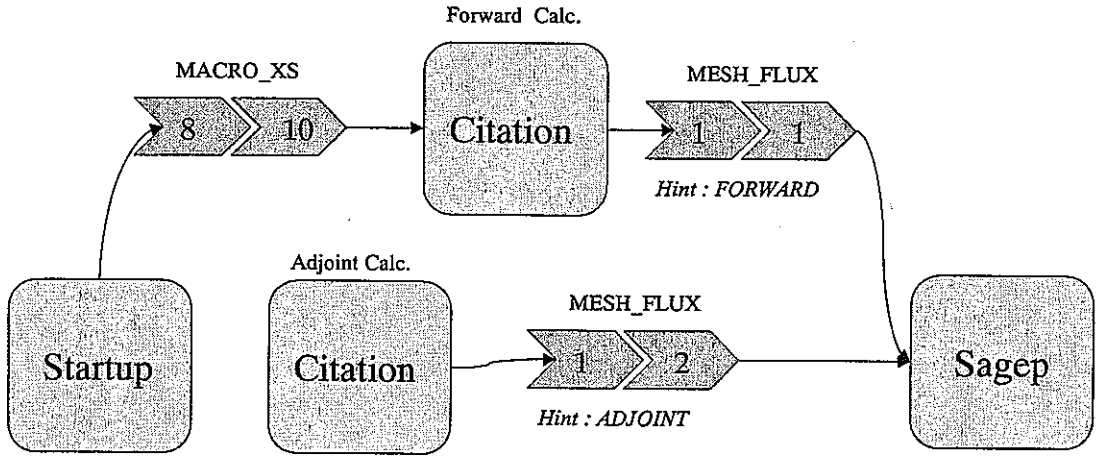


図 3-6　ファイル入出力の管理方法の例（ヒント情報の利用）

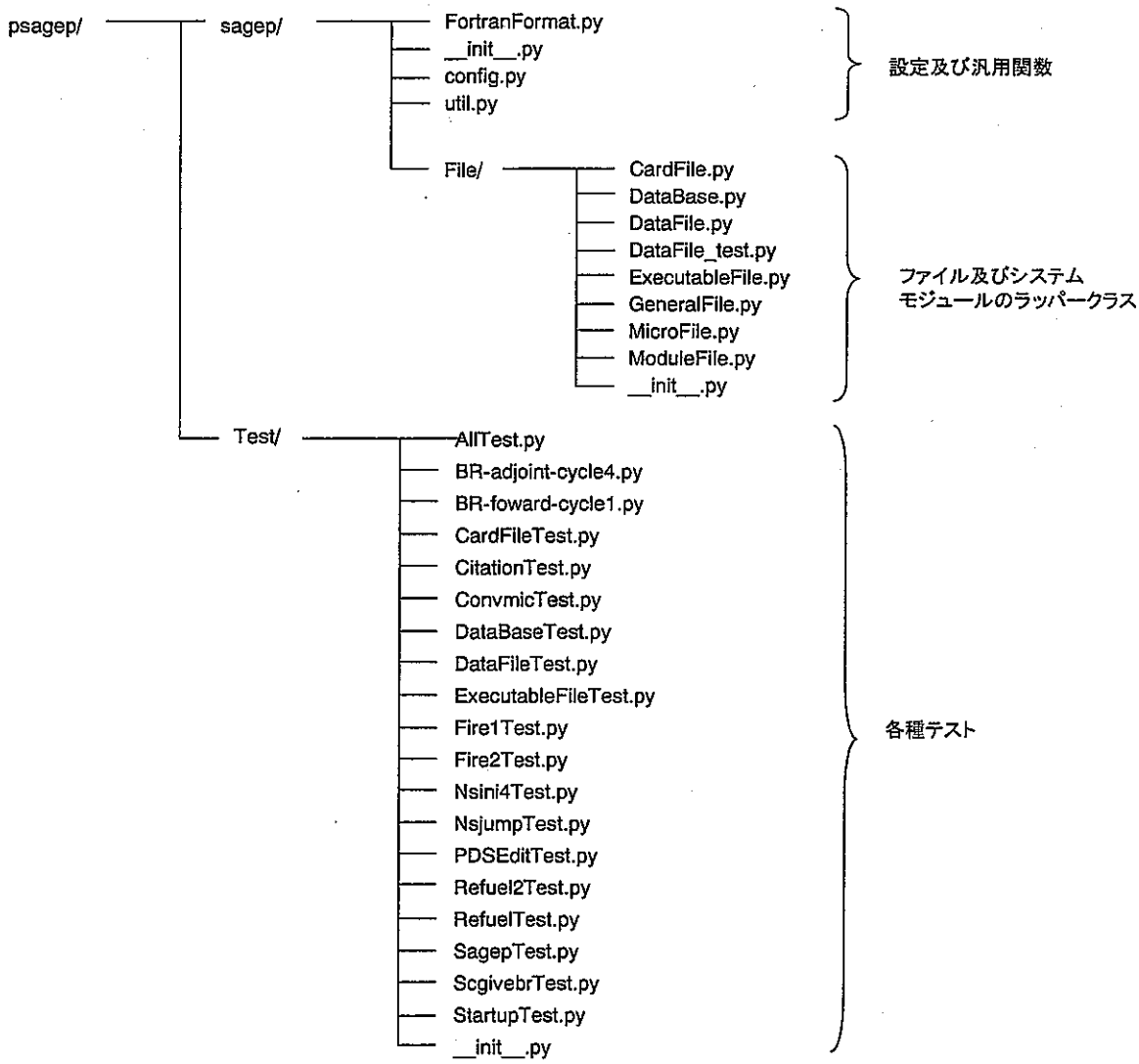


図 3-7　フェーズ１において実装したモジュールパッケージ

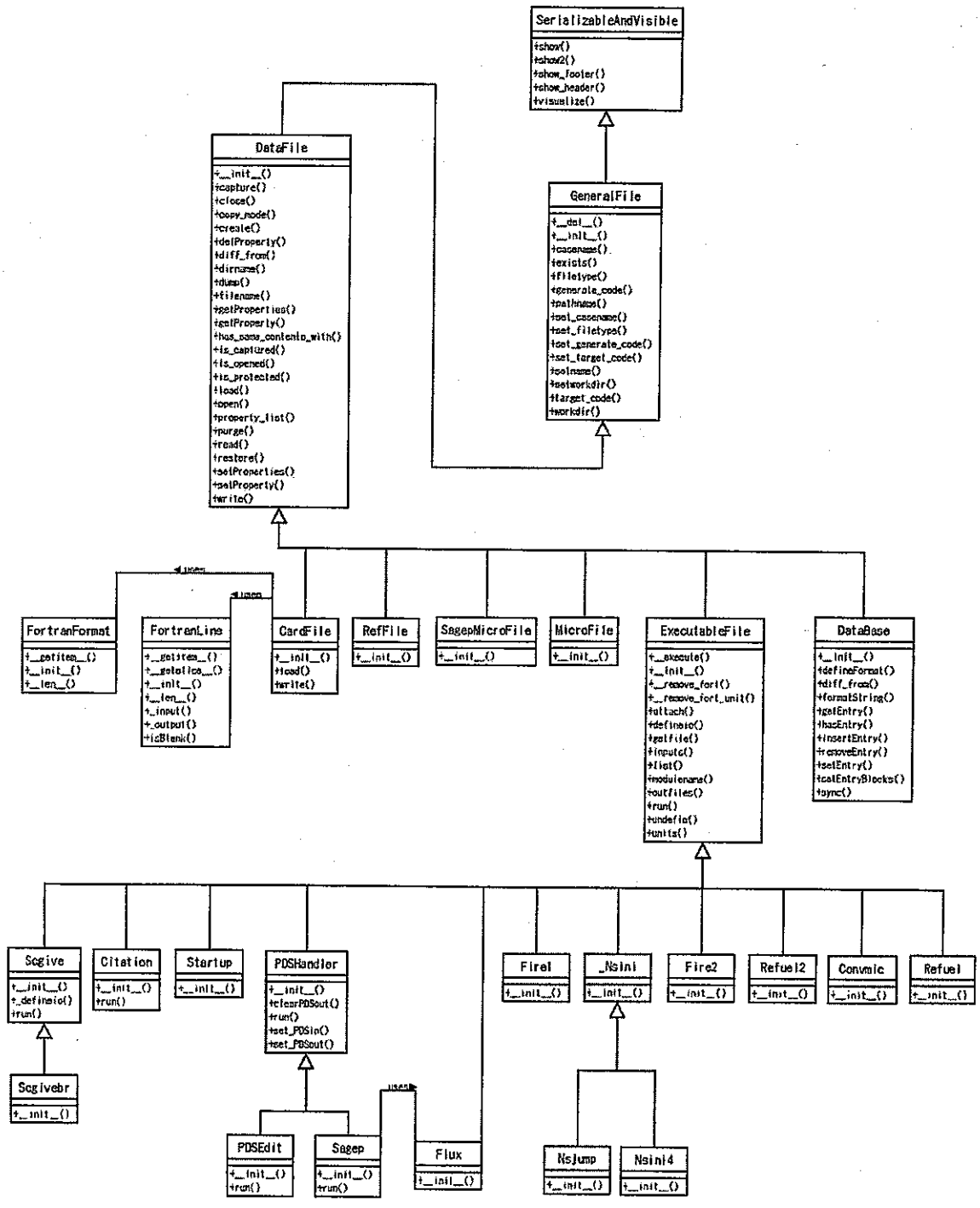


図 3-8　各システムモジュールのラッパークラスに関するクラス図

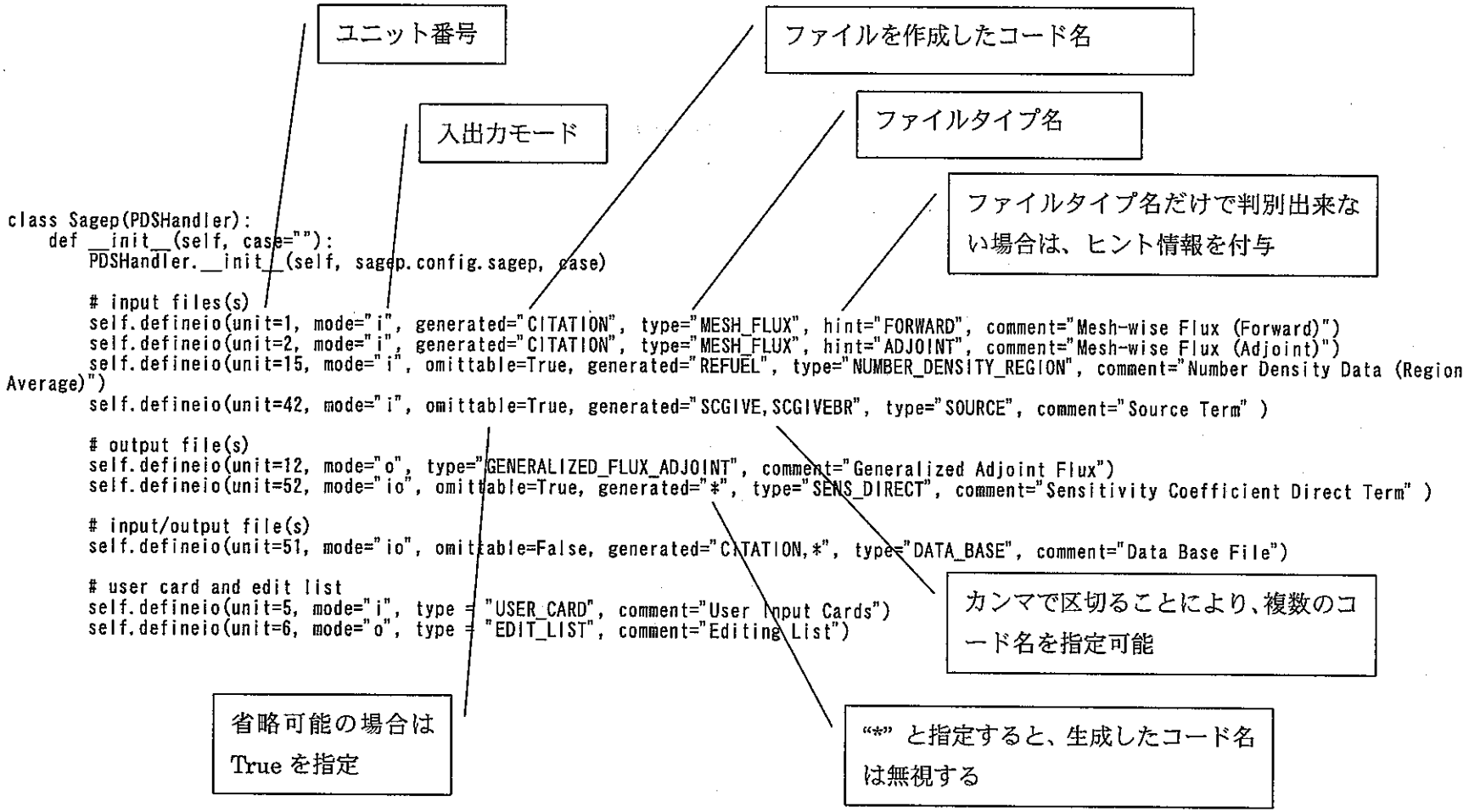
```
FISSPEC
 2.74940E-02 1.20238E-01 2.22775E-01 2.34681E-01 1.73057E-01 1.40719E-01
 5.36364E-02 1.83373E-02 6.11115E-03 1.99959E-03 6.49155E-04 2.12088E-04
 6.88247E-05 2.22676E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
SFACT
  -1.0000000000E+00
KEFF 01B
          1.065947
AVFX 01B
 3.04200E-02 1.44101E-01 3.58623E-01 5.23008E-01 6.18121E-01 1.60867E+00
 1.91446E+00 2.00639E+00 1.68933E+00 1.28785E+00 8.93881E-01 4.51927E-01
 1.65946E-01 2.49575E-01 1.03763E-01 2.37773E-02 4.07610E-03 6.29563E-04
 2.74463E-02 1.29918E-01 3.23330E-01 4.70445E-01 5.56431E-01 1.44936E+00
 1.72037E+00 1.79636E+00 1.50701E+00 1.14349E+00 7.89252E-01 3.96448E-01
 1.45751E-01 2.15194E-01 8.64037E-02 1.92418E-02 3.15150E-03 4.54734E-04
 2.22242E-02 1.05613E-01 2.62398E-01 3.83781E-01 4.49489E-01 1.14570E+00
 1.34929E+00 1.39827E+00 1.15724E+00 8.73401E-01 5.98673E-01 3.02061E-01
 1.08051E-01 1.68363E-01 7.05404E-02 1.58751E-02 2.69464E-03 4.16673E-04

 (中略)

7.57614E-04 7.37394E-04 6.68744E-04 4.73941E-04 4.65821E-04 3.62800E-04
 7.06484E-02 6.84528E-02 7.02560E-02 6.66350E-02 5.99318E-02 5.80394E-02
 4.51133E-02 3.47111E-02 2.65683E-02 1.99498E-02 1.71788E-02 1.60026E-02
 1.57534E-02 1.57593E-02 1.68318E-02 1.51365E-02 1.62361E-02 1.52291E-02
 7.71321E-03 7.58190E-03 7.14783E-03 6.08114E-03 4.85422E-03 3.19648E-03
 1.76421E-03 8.95893E-04 4.46295E-04 1.80649E-04 9.41721E-05 5.74910E-05
 3.68760E-05 3.56486E-05 2.27635E-05 9.32739E-06 7.38502E-06 2.62000E-06
AVFXa04E
 3.02593E+06 4.21798E+06 2.24645E+06 4.12625E+06 5.95727E+06 5.95727E+06
 9.62792E+05 2.20067E+06                 E+06 2.21270E+07 3.57609E+06
 5.93148E+06 8.10636E+06                 5E+07
END
```

エントリー名

データ本体は、行頭にスペースを一つ以上配置する。

最後は、"END"

図 3-9　データベースファイルの構造




```
class Sagep(PDSHandler):
    def __init__(self, case=""):
        PDSHandler.__init__(self, sagep.config.sagep, case)

        # input files(s)
        self.defineio(unit=1, mode="i", generated="CITATION", type="MESH_FLUX", hint="FORWARD", comment="Mesh-wise Flux (Forward)")
        self.defineio(unit=2, mode="i", generated="CITATION", type="MESH_FLUX", hint="ADJOINT", comment="Mesh-wise Flux (Adjoint)")
        self.defineio(unit=15, mode="i", omittable=True, generated="REFUEL", type="NUMBER_DENSITY_REGION", comment="Number Density Data (Region
Average)")
        self.defineio(unit=42, mode="i", omittable=True, generated="SCGIVE,SCGIVEBR", type="SOURCE", comment="Source Term" )

        # output file(s)
        self.defineio(unit=12, mode="o", type="GENERALIZED_FLUX_ADJOINT", comment="Generalized Adjoint Flux")
        self.defineio(unit=52, mode="io", omittable=True, generated="*", type="SENS_DIRECT", comment="Sensitivity Coefficient Direct Term" )

        # input/output file(s)
        self.defineio(unit=51, mode="io", omittable=False, generated="CITATION,*", type="DATA_BASE", comment="Data Base File")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")
```

図 3-10　ラッパークラスにおけるファイル入出力関係の定義 (Sagep クラスにおける defineio()メソッドの使用例)

4. 改良のためのオブジェクト指向分析および設計

本章では、3.3 節で述べたフェーズ 2 における SAGEP-BURN システムの再構築を目指し、オブジェクト指向分析を実施した。また、一例として UML[11,12]を用いた設計を実施した。

4.1 分析

第 2.2.3 節で述べたように、解析者が操作する解析モジュール（エンドモジュール）は少ない方が望ましい。従って、フェーズ 2 実装では、エンドモジュールは 1 つと設定する。解析者からエンドモジュールに対する指示はユースケース図（図 4-1）に示す次の 3 つに分解することが出来る。これらを解析者からシステムへのメッセージと見なし、メッセージの受け手をオブジェクトとして設定する。さらに、オブジェクト間での作業分担及びメッセージの内容や手順について検討を行い、図 4-2 に示すコラボレーション図を作成した。以下では、図 4-2 の内容について説明する。

解析者から最初のメッセージは、「解析条件の設定」である。これにより、「解析条件管理オブジェクト」は、解析者から与えられた入力ファイルから、計算を実施するためのデータベースを構築する。

解析者からの 2 番目のメッセージは、「計算実行」である。「計算実行オブジェクト」は、「解析条件管理オブジェクト」に、計算体系や断面積情報等の計算条件を取得する(2a)。次に、システムモジュールが必要とする入力ファイル(5 番ファイル)を、先ほどの計算条件から生成する(2b)。さらに、システムモジュールへの入力ファイルを「データファイル管理オブジェクト」から取得し(2c)、ユーザカードとともに「システムモジュールラッパー」に渡して計算を実行する(2d, 2e)。「計算実行オブジェクト」は、「システムモジュールラッパー」から得た出力ファイルを「データファイル管理オブジェクト」に登録し(2f)、シーケンスが終了するまで自立的に制御を行う(2g)。

最後に、解析者は「計算結果取得オブジェクト」に対して計算結果の取得に関するメッセージを送る。「解析結果取得オブジェクト」は、「データファイル管理オブジェクト」から必要なファイルを取り出し(3a)、ユーザの希望に添った形式にデータを加工する(3b)。

以上のような手順で一連の解析が可能であると考えられる。なお、「計算実行オブジェクト」は解析項目によって変更することにより、増殖比や燃焼反応度損失等の異なる評価項目に対応できると予想される。

4.2 設計

前節のオブジェクト指向分析を元に、フェーズ2システムの設計を行った。図4-3に設計の一例としてクラス図を示す。なお、図中において灰色で描かれたクラスはフェーズ1において実装したものである。

図4-2における「解析条件管理オブジェクト」、「データファイル管理オブジェクト」、「計算オブジェクト」は、それぞれ InfoSetManager、DataFileManger, Scenario オブジェクトに対応する。なお、Scenarioオブジェクトは、「解析結果取得オブジェクト」の役割も兼ねるとした。これは、どのような計算結果を取得するべきかは、計算シーケンスと密接な関係があるため、これらの役割は分離しない方が良いと考えたためである。

InfoSetMangerは、InfoSetと呼ばれる解析条件情報が保管されているデータベースである。情報の保持形態やアクセス手法については現時点では未定である。

DataFileManagerは、ExecutableFileから出力されるデータファイルを保管するデータベースである。DataFileManagerは、諸条件（計算サイクル情報やファイルタイプ）をキーにして、該当するファイルを抽出することが出来る。これにより、これから計算に必要なデータファイルを取り出すことが出来る。

Scenarioオブジェクトは計算シーケンスを制御するものであり、これらのオブジェクトのうち最も重要な役割を果たす。Scenarioオブジェクトにおける計算シーケンスの制御には、デザインパターン[6,7]におけるCommandパターンないしはInterpreterパターンを採用する。なお、実際に制御が渡させるのは、各評価対象項目に対して用意されるScenarioオブジェクトの派生クラスである。

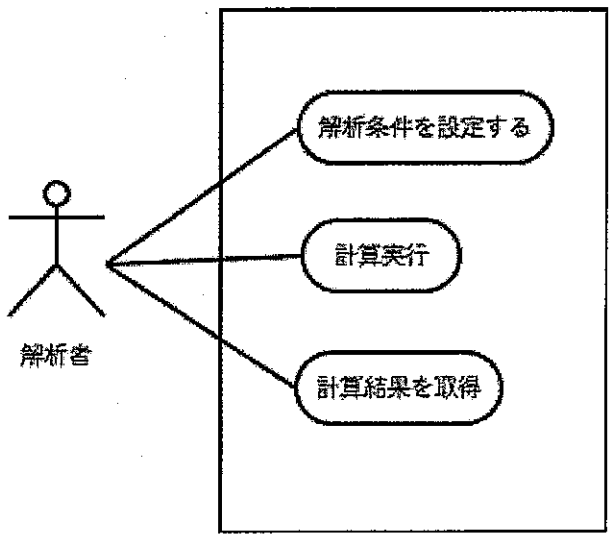


図 4-1　解析者とエンドモジュールの関係（ユースケース図）

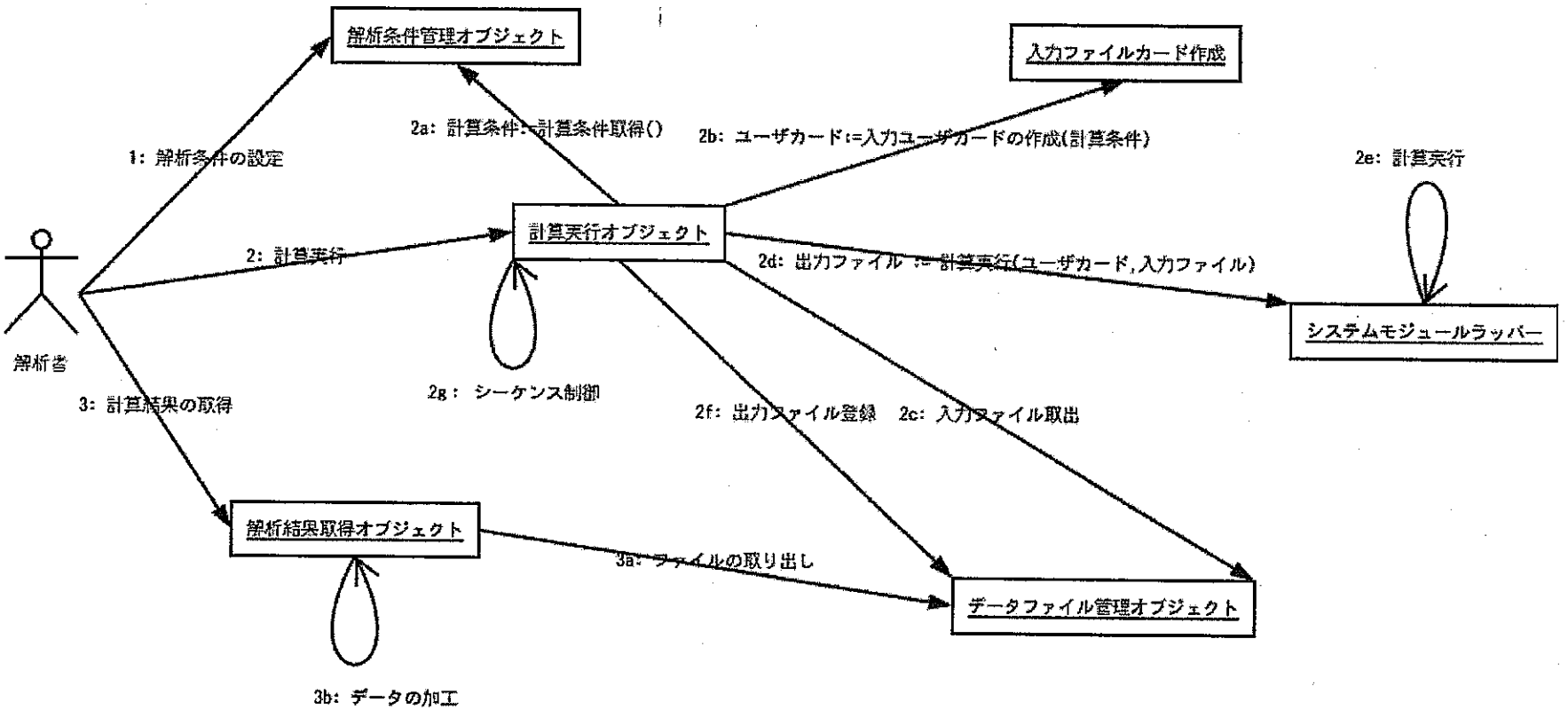
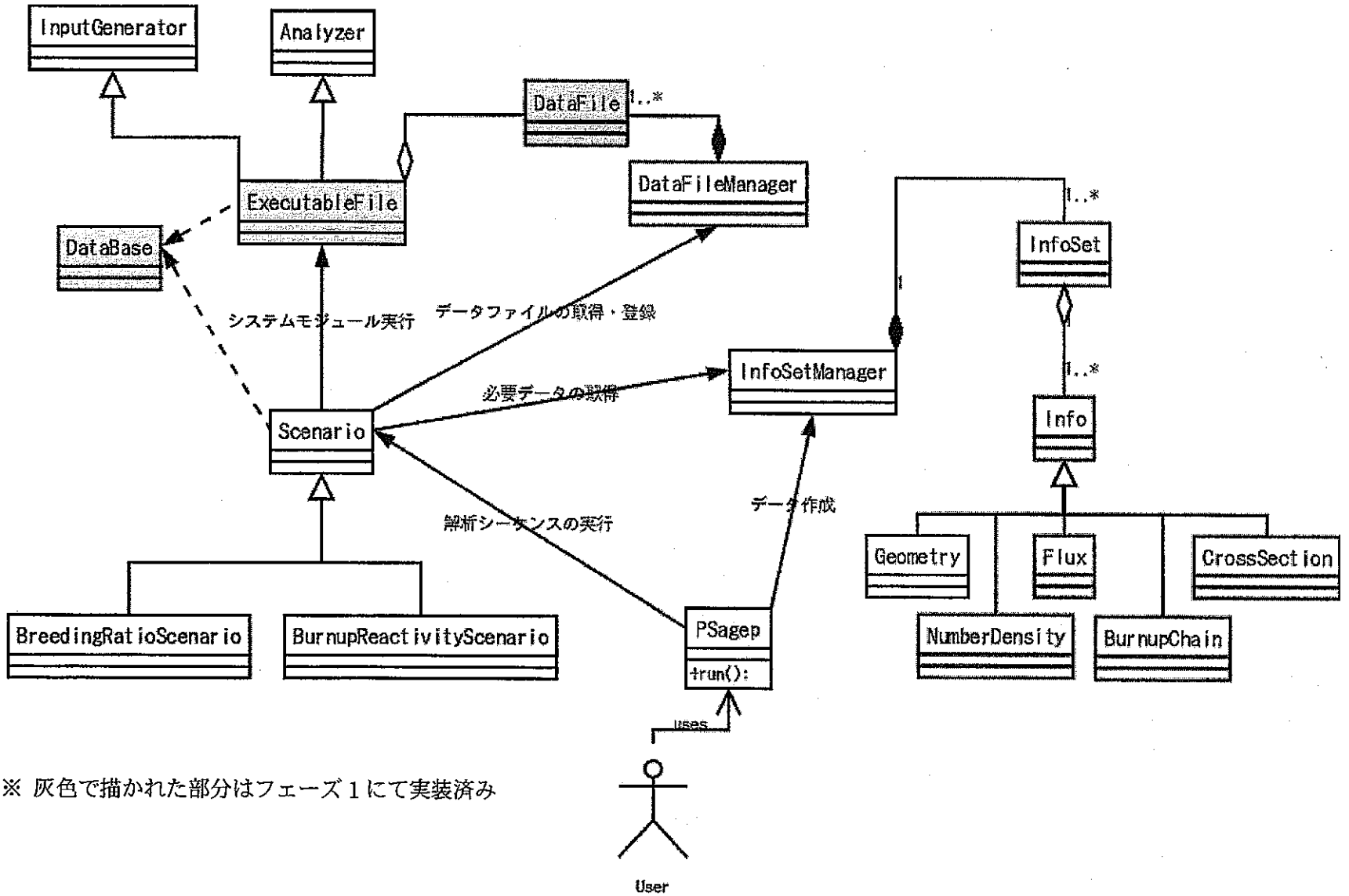


図 4-2　オブジェクト指向分析により得たオブジェクト間の関連（コラボレーション図）



※ 灰色で描かれた部分はフェーズ１にて実装済み

図 4-3　フェーズ２システムの設計例　(クラス図)

5. おわりに

高速炉の実用化に向けて、燃焼感度解析が果たす役割はきわめて重要であり、同解析を行うための解析コード SAGEP-BURN の有効性が確認されてきた。しかしながら、この燃焼感度の解析は非常に複雑であり、SAGEP-BURN を用いた解析では理論に合わせてユーザが計算ステップをひとつずつ追いかける必要があるため、ユーザへの負担が大きく解析作業が極めて非効率的という問題があった。また、解析対象によって計算ステップが変わることや、物理的意味を分析する際には計算ステップを分解する必要があることから、現在の燃焼感度解析コードをより使いやすくシステム化する必要があった。このため、本研究では、オブジェクト指向とスクリプト言語の技術を利用して、燃焼感度解析コードのシステム化作業を実施した。

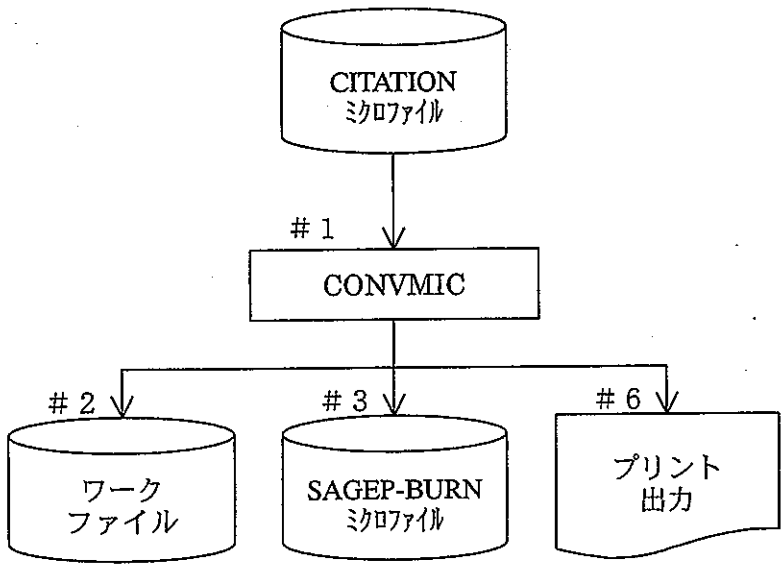
まず、既存の SAGEP-BURN システムのどこに問題があるのかをプロセス分析を通じて明らかにし、新しいシステムではどのような形態が望ましいかを検討した。その結果をふまえて、オブジェクト指向スクリプト言語 Python を用いてプロトタイプを実装した。この際、Fortran で記述されたシステムモジュールは変更せずに、カプセル化することにより Python 言語と結合した。これにより、既存の SAGEP-BURN システムの全機能は Python 言語レベルで制御可能となり、特に入出力に関わる部分の柔軟性が大幅に向上した。なお、システムモジュールを含め、計算ロジックには変更は施しておらず、今回のシステムを用いて従来の解析結果と同一の計算結果を得ることを確認した。

ユーザがより使いやすくするための今後の改良に向けて、オブジェクト指向分析を実施した。これをもとに、今年度の作業により実装した部分を基本ライブラリとして組み込んだ改良システムの基本設計を実施した。

6. 参考文献

1. 花木　洋, 澤田 周作, 三田 敏男,「燃焼核特性に対する感度解析コードの整備」, JNC TJ9124 93-009 (1993).
2. 花木　洋, 三田 敏男, 大橋 正久,「燃焼核特性に対する感度解析コードの整備(II)」, JNC TJ9124 94-007 (1994).
3. 横山 賢治, 細貝 広視 他,「工学系モデリング言語としての次世代解析システムの開発(I) 課題及び要素技術の調査」, JNC TN9420 2002-004 (2002).
4. 横山 賢治, 細貝 広視 他,「工学系モデリング言語としての次世代解析システムの開発(II) プロトタイプ作成による検討」, JNC TN9400 2003-021 (2003).
5. 日本 XP ユーザグループ,「eXtereme Programming テスト技法　xUnit ではじめる実践 XP プログラミング」, 翔泳社 (2001).
6. Eric Gamma, Richard Helm 他, 本位田 真一, 吉田 和樹 監訳,「オブジェクト指向における再利用のためのデザインパターン (改訂版)」, ソフトバンク パブリッシング (1999).
7. 結城　浩,「Java 言語で学ぶデザインパターン入門」, ソフトバンク パブリッシング, (2001).
8. http://www.python.org
9. http://www.ruby-lang.org
10. ScientificPython, http://starship.python.net/~hinsen/ScientificPython/
11. 竹政 昭利,「はじめて学ぶ UML」, ナツメ社 (2003).
12. Hans-Erik Eriksson, Magnus Penker, 杉本 宣男 他監訳,「UML ガイドブック」, トッパン (1998).

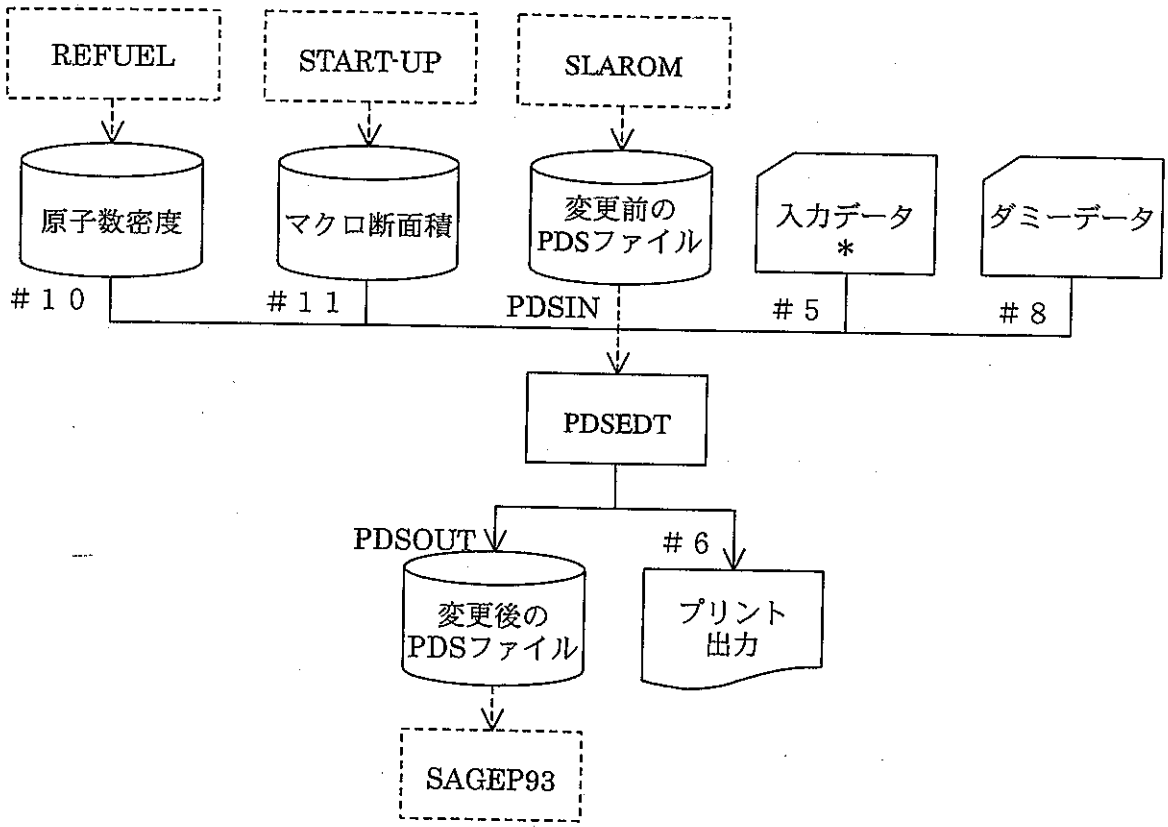
付録１　SAGEP-BURN の各モジュールにおけるファイル入出力の詳細



入力			出力	
入力を作成したﾌﾟﾛｸﾞﾗﾑ	JOINT	装置	＃２	＃３
装置	＃１	出力先ﾌﾟﾛｸﾞﾗﾑ	ﾜｰｸﾌｧｲﾙ	「SAGEP-BURN」ｺｰﾄﾞｼｽﾃﾑ
ﾐｸﾛ断面積ﾌｧｲﾙ形式	CITATION	ﾐｸﾛ断面積ﾌｧｲﾙ形式	----------	「SAGEP-BURN」

図A1-1　ＣＯＮＶＭＩＣの入出力ファイル

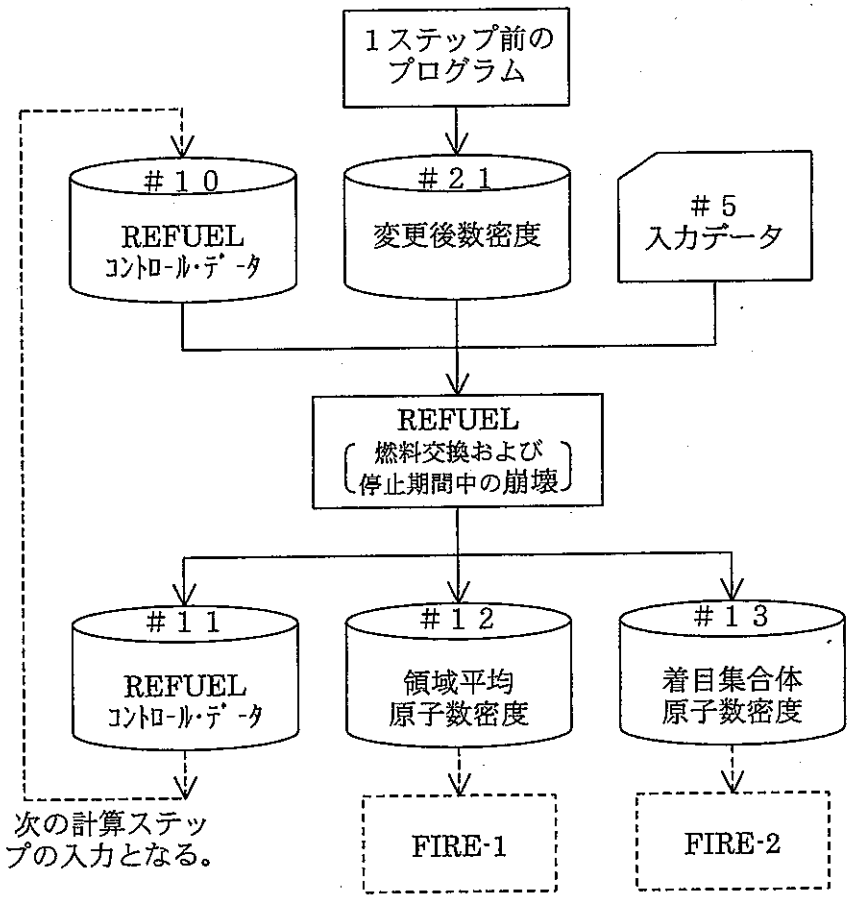
入　　力					
入力を作成したﾌﾟﾛｸﾞﾗﾑ	― ― ―	― ― ―	REFUEL	START-UP	SLAROM
装置	＃５	＃８	＃１０	＃１１	PDSIN
当該サイクル燃焼時期	入力ﾃﾞｰﾀ (ｵﾌﾟｼｮﾝ=0)	ﾀﾞﾐｰﾃﾞｰﾀ	原子数密度	マクロ断面積	変更前 PDS ﾌｧｲﾙ



＊５番入力データの編集オプション＝０を設定

出　　力			
装置	PDSOUT	＃２	＃６
出力先ﾌﾟﾛｸﾞﾗﾑ	SAGEP93	― ― ―	― ― ―
当該ｻｲｸﾙ燃焼時期	変更後 PDS ﾌｧｲﾙ	ｶｰﾄﾞ出力	ﾌﾟﾘﾝﾄ出力

図A1-2　ＰＤＳＥＤＴの入出力ファイル



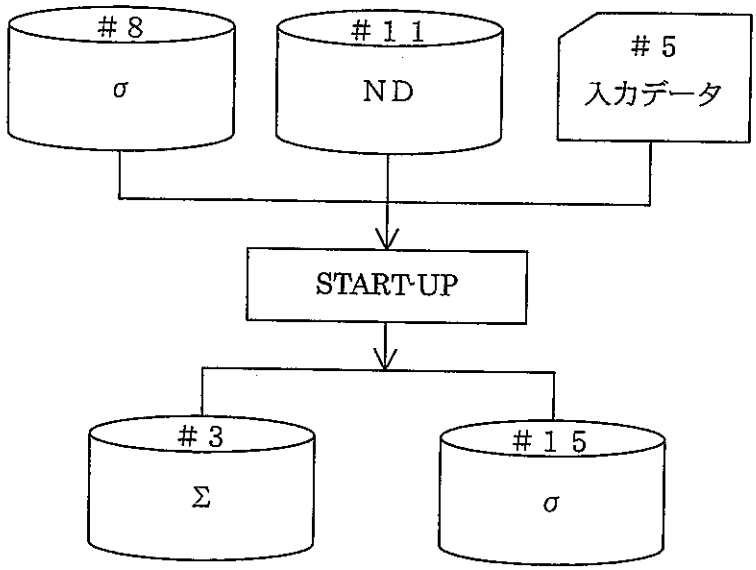
＊当該するステップのプログラム

１．燃交ステップ　　：＃２１装置は未使用

２．燃焼ステップ　　：ＦＩＲＥ－２

図 A1-3　ＲＥＦＵＥＬの入出力ファイル

出　　力			
入力を作成したﾌﾟﾛｸﾞﾗﾑ	----------	fort.12 REFUEL	----------
装置	＃８	＃１１	＃５
当該燃焼*3 サイクル	σ	ＮＤ	入力データ

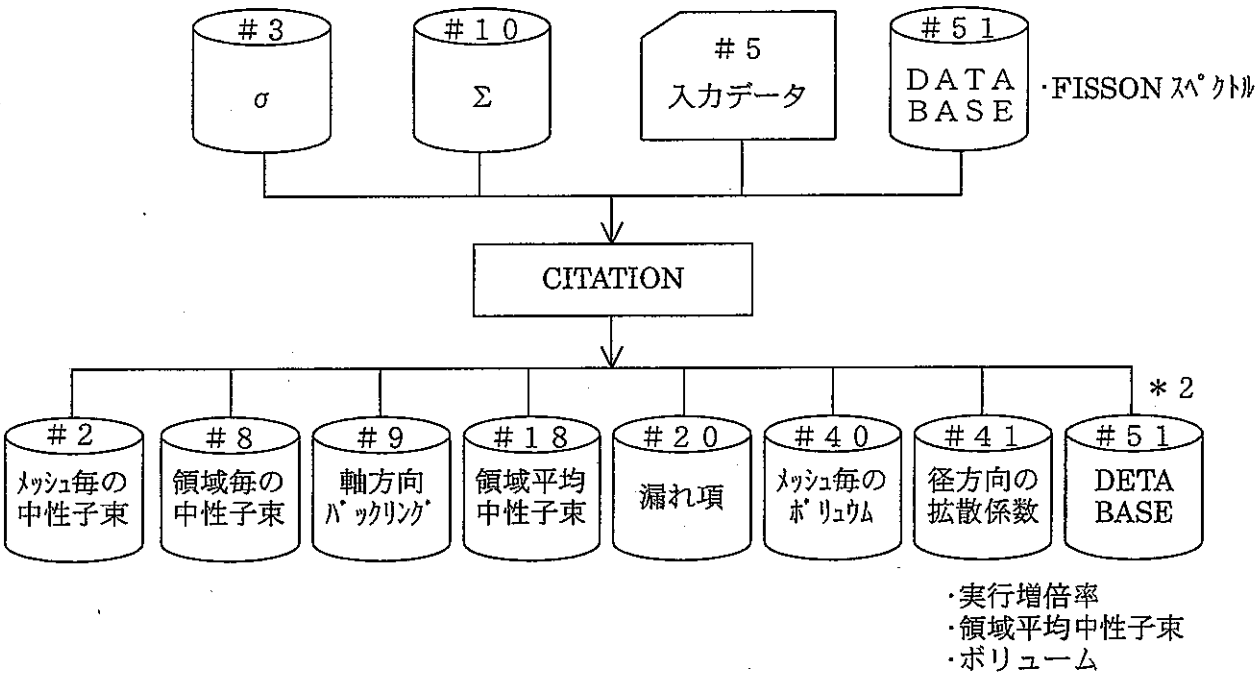


出　　力		
装置	＃３	＃１５
出力先 プログラム	CITATION	未使用のため ﾜｰｸﾌｧｲとする
当該燃焼*3 サイクル	Σ	σ

＊　３　当該サイクルを１ステップで計算した場合。

図 A1-4　ＳＴＡＲＴ－ＵＰの入出力ファイル

入　　力				
入力を作成したﾌﾟﾛｸﾞﾗﾑ	CITATION	START-UP	----------	----------
装置	＃３	＃１０	＃５１	＃５
当該燃焼*3 サイクル	初期中性子束分布設定値 10PT(1)=1 の時	Σ	FISSION ｽﾍﾟｸﾄﾙ	入力データ

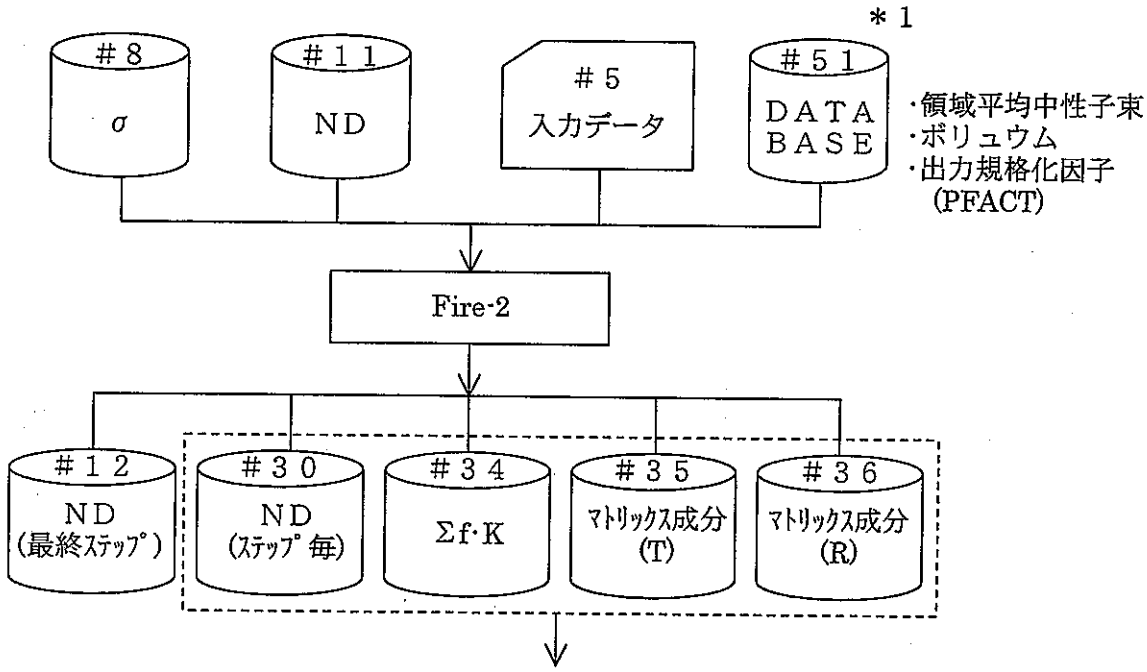


出　　力								
装置	＃２	＃８	＃９	＃１８	＃２０	＃４０	＃４１	＃５１
出力先 ﾌﾟﾛｸﾞﾗﾑ	SAGEP93	未使用の為 ﾜｰｸﾌｧｲﾙ	CITATION	未使用の為 ﾜｰｸﾌｧｲﾙ	NS-INIn NS-JUMP	未使用の為 ﾜｰｸﾌｧｲﾙ	未使用の為 ﾜｰｸﾌｧｲﾙ	FIRE-1 FIRE-2
当該燃焼*3 サイクル	ﾒｯｼｭ毎の中性子束	領域毎の中性子束 10PT(4)=0 の時	軸方向ﾊﾞｯｸﾘﾝｸﾞ 10PT(6)=0 の時	領域平均中性子束	漏れ項	ﾒｯｼｭ毎のﾎﾞﾘｭｳﾑ 10PT(5)=0 の時	径方向の拡散係数 10PT(5)=0 の時	＊２

＊３　当該サイクルを１ステップで計算した場合。

図 A1-5　ＣＩＴＡＴＩＯＮの入出力ファイル

入　　力				
入力を作成したﾌﾟﾛｸﾞﾗﾑ	----------	REFUEL	CITATION FIRE-1	----------
装置	＃８	＃１１	＃５１	＃５
当該燃焼*2 サイクル	σ	初期 ND	＊１	入力データ

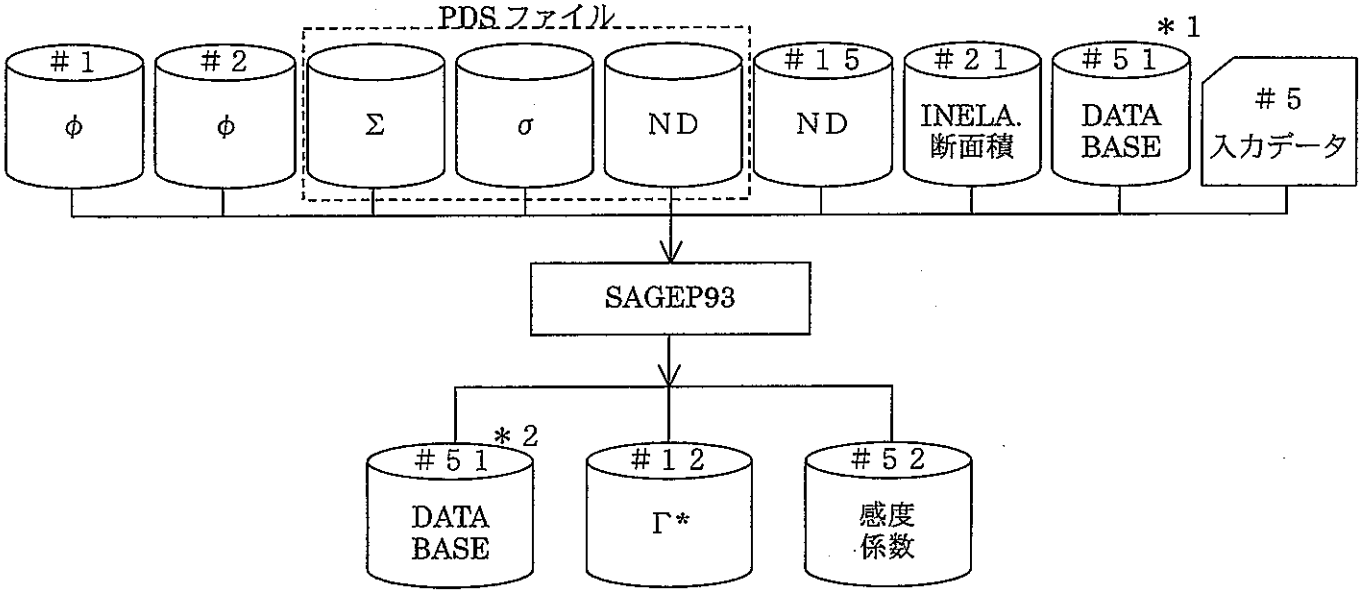


出　　力					
装置	＃１２	＃３０	＃３４	＃３５	＃３６
出力先 ﾌﾟﾛｸﾞﾗﾑ	REFUEL	SCGIVE	SCGIVE	SCGIVE	SCGIVE
当該燃焼*2 サイクル	ND (最終ｻﾌﾞｽﾃｯﾌﾟ	ND (ｻﾌﾞｽﾃｯﾌﾟ毎)	Σf･K	ﾏﾄﾘｯｸｽ 成分(T)	ﾏﾄﾘｯｸｽ 成分(R)

＊２　当該サイクルを１ステップで計算した場合。□内ファイルは随伴計算で使用する。
・原子数密度

図 A1-5　ＦＩＲＥ－２の入出力ファイル

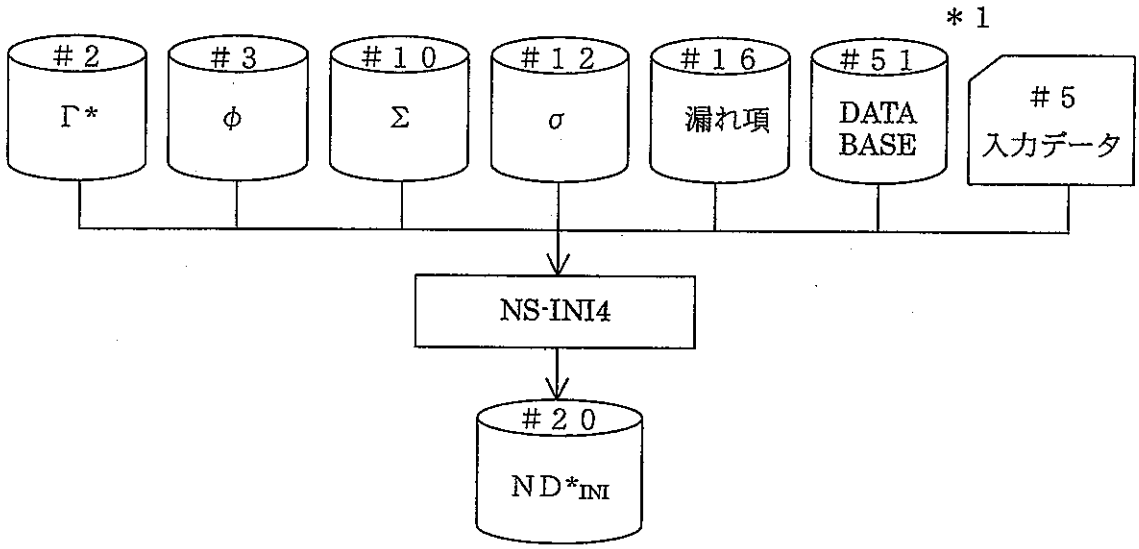
入　　力									
入力を作成したﾌﾟﾛｸﾞﾗﾑ	CITETION	CITETION	START-UP	----------	REFUEL	REFUEL	----------	CITETION	----------
装置	＃３	＃３	PDS ﾌｧｲﾙ	PDS ﾌｧｲﾙ	PDS ﾌｧｲﾙ	＃１５	＃２１	＃５１	＃５
当該燃焼*3サイクル	末期 φ	末期 φ*	末期 Σ	σ	末期 ND	末期 ND	非弾性錯乱断面積	＊１	入力ﾃﾞｰﾀ



出　　力			
装置	＃１２	＃５１	＃５２
出力先ﾌﾟﾛｸﾞﾗﾑ	NS-INI4	NS-INI4	----------
当該燃焼*3サイクル	Γ*	＊２	感度係数 直接項 中性子束項 出力項

＊１　・核分裂スペクトル（あらかじめ#51 番装置に設定しておく。）
　　・ＳＦＡＣＴ（あらかじめ#51 番装置に-1.0 を設定しておく。）
　　・当該サイクルの末期固有値（CITATION で作成）
＊２　・出力規格化因子（PFACT）、NS-INI4 で使用する。
　　・FERTILE CALCULAION、NS-INI4 で使用する。
　　・FISSILE CALCULAION、NS-INI4 で使用する。
　　・PSINI*PFACT、NS-INI4 で使用する。
＊３　・当該サイクルを１ステップで計算した場合。

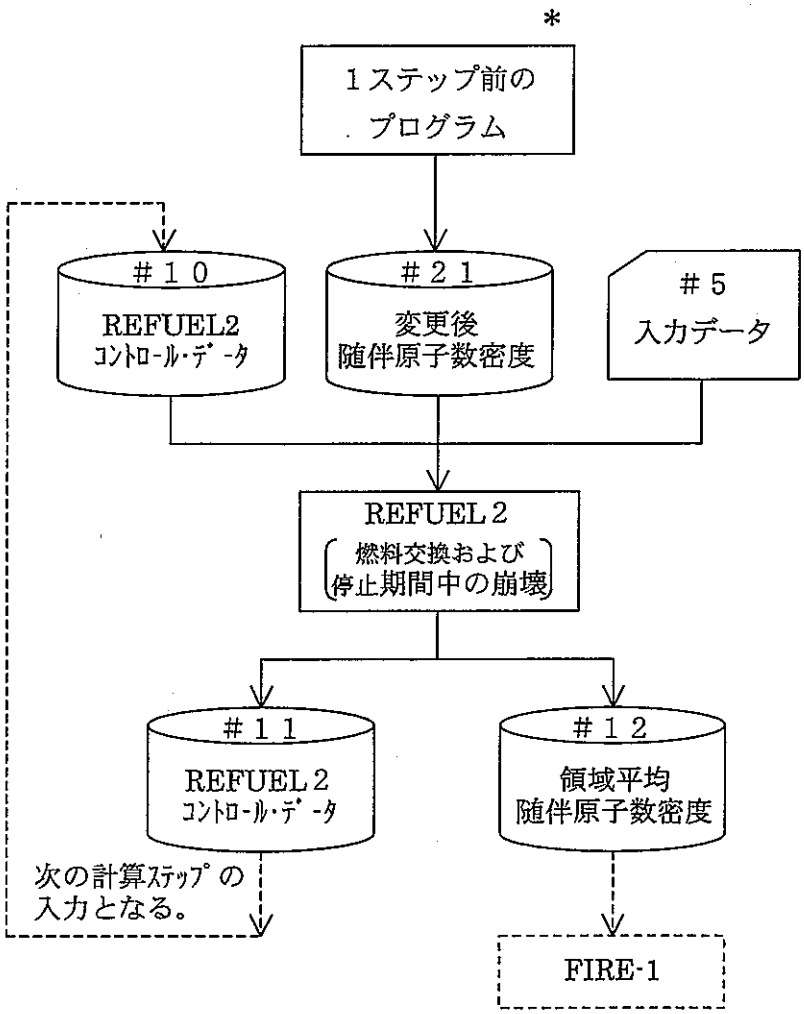
図 A1-6　　ＳＡＧＥＰ９３の入出力ファイル
（直接項および i=1+1 における中性子束項を計算する場合）



入力							出力	
SAGEP93	CITATION	START-UP	----------	CITATION	CITATION SAGEP93	----------	装置	＃20
＃2	＃3	＃10	＃12	＃16	＃51	＃5	出力先 ﾌﾟﾛｸﾞﾗﾑ	REFUEL2
Γ*	末期 φ	末期 Σ	σ	末期 漏れ項	＊1	入力 データ	当該燃焼*2 サイクル	末期 ND*INI

＊1　・出力規格化因子（PFACT）
　　・核分裂スペクトル，固有値，PSTAR，DNORM
＊2　・当該サイクルを１ステップで計算した場合。

図 A1-7　ＮＳ－ＩＮＩ４の入出力ファイル



＊当該するステップのプログラム

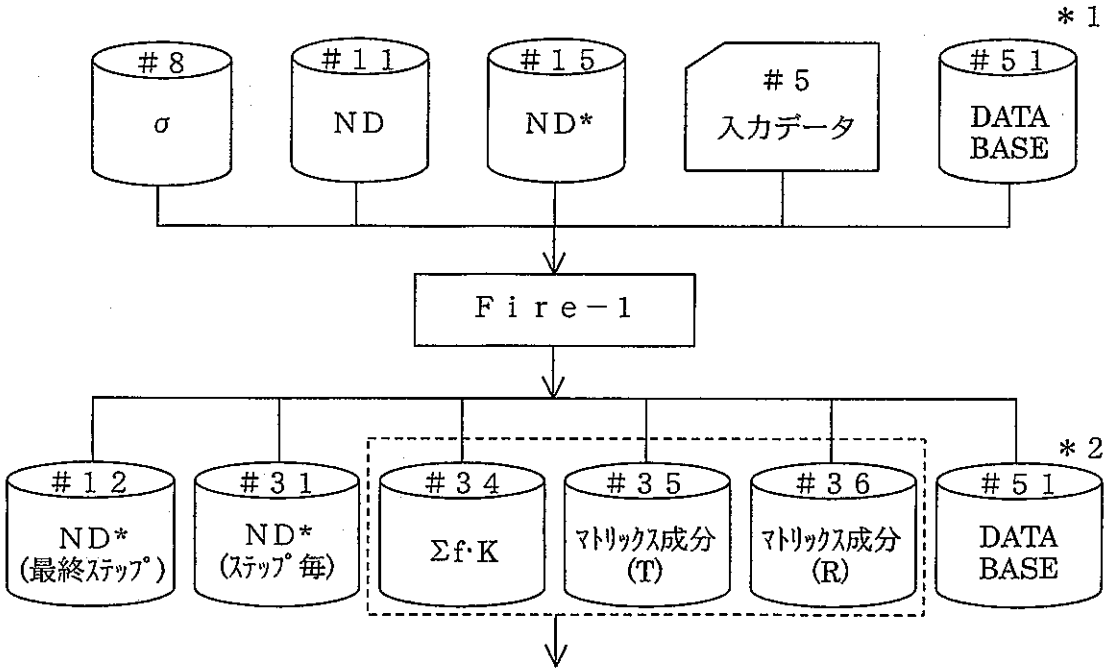
１．随伴数密度初期値設定ステップ　　：ＮＳ－ＩＮＩ４

２．燃焼ステップ　　：ＦＩＲＥ－１

３．不連続ステップ　　：ＮＳ－ＪＵＭＰ

図 A1-8　　ＲＥＦＵＥＬ－２の入出力ファイル

入　　力					
入力を作成したﾌﾟﾛｸﾞﾗﾑ	----------	REFUEL	REFUEL-2	CITATION	----------
装置	＃８	＃１１	＃１５	＃５１	＃５
当該燃焼*3 サイクル	σ	初期 ND	末期 ND*	＊１	入力データ



＊１　・領域平均中性子束、ボリューム

＊２　・出力規格化因子（PFACT)、出力で規格化した中性子束

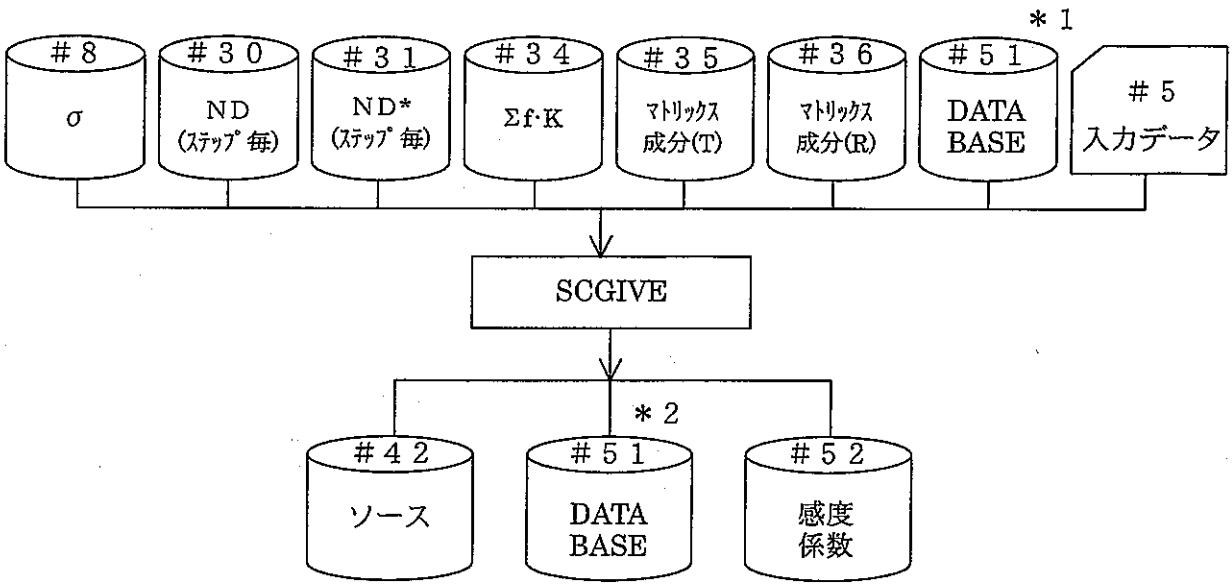
＊３　・当該サイクルを１ステップで計算した場合。

□ 内は随伴計算では不用のためワークファイルとする。

出　　力						
装置	＃１２	＃３１	＃３４	＃３５	＃３６	＃５１
出力先 ﾌﾟﾛｸﾞﾗﾑ	REFUEL-2	SCGIVE	----------	----------	----------	SCGIVE
当該燃焼*3 サイクル	ND* (最終ｻﾌﾞｽﾃｯﾌﾟ)	ND (ｻﾌﾞｽﾃｯﾌﾟ毎)	Σf･K	ﾏﾄﾘｯｸｽ 成分(T)	ﾏﾄﾘｯｸｽ 成分(R)	＊２

図 A1-9　ＦＩＲＥ－１の入出力ファイル

入　　力								
入力を作成したﾌﾟﾛｸﾞﾗﾑ	----------	FIRE-1	FIRE-1	FIRE-1	FIRE-1	FIRE-1	FIRE-1	----------
装置	＃８	＃３０	＃３１	＃３４	＃３５	＃３６	＃５１	＃５
当該燃焼*3 サイクル	σ	ND (ｻﾌﾞｽﾃｯﾌﾟ毎)	ND* (ｻﾌﾞｽﾃｯﾌﾟ毎)	Σf･K	ﾏﾄﾘｯｸｽ 成分(T)	ﾏﾄﾘｯｸｽ 成分(R)	＊１	入力データ

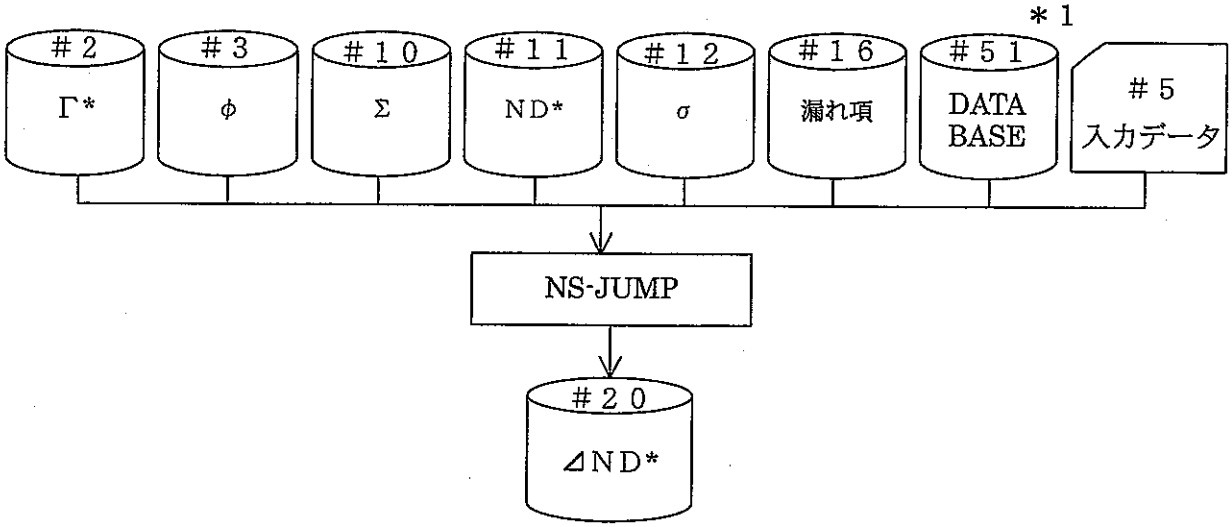


出　　力			
装置	＃４２	＃５１	＃５２
出力先ﾌﾟﾛｸﾞﾗﾑ	SAGEP93	SAGEP93 NS-JUMP	----------
当該燃焼*3 サイクル	ソース	＊２	感度係数 数密度項 出力項

＊１　・出力規格化因子（PFACT）
　　・出力で規格化した中性子束
　　・ボリューム
＊２　・SFACT
　　・PSTAR
　　・DNORM
＊３　・当該サイクルを１ステップで計算した場合

□ 内のファイルは通常燃焼計算のＦＩＲＥ－１で算出。

図 A1-10　ＳＣＧＩＶＥの入出力ファイル



	入力								出力	
入力を作成したﾌﾟﾛｸﾞﾗﾑ	SAGEP93	CITATION	START-UP	REFUEL2	----------	CITATION	CITATION FIRE-1 SCGIVE	----------	装置	＃20
装置	＃2	＃3	＃10	＃11	＃12	＃16	＃51	＃5	出力先ﾌﾟﾛｸﾞﾗﾑ	REFUEL2
当該燃焼*2サイクル	Γ*	初期 φ	初期 Σ	初期 ND*	σ	初期 漏れ項	＊1	入力データ	当該燃焼*2サイクル	末期 ⊿ND*

＊1　CITATIONで作成　・当該燃焼サイクル初期の固有値
・核分裂スペクトル
FIRE-1で作成　・出力規格化因子（PFACT）
SCGIVEで作成　・PSTAR
・DNORM

＊2　当該サイクルを1ステップで計算した場合。

図A1-11　ＮＳ－ＪＵＭＰの入出力ファイル

付録 2　新 SAGEP-BURN のための Python クラスライブラリの詳細

図 A2-1 ExecutableFile.py

```
from sagep.File.DataFile import *
from sagep.util import *
import string
import os
import copy
import sagep.config
from pprint import *

class ExecutableFile(DataFile):
    def __init__(self, filename, case = ""):
        DataFile.__init__(self, filename)
        self.setworkdir( sagep.config.work_dir )
        self.files = {}
        self.mode = {}
        self.omittable = {}
        self.target = {}
        self.generated = {}
        self.type = {}
        self.hint = {}
        self.comment = {}
        self.set_casename(case)
        mod_tmp = string.split( str( self.__class__ ), '.' )
        self.mod_name = string.upper( mod_tmp[-1] )

    def __execute(self, command):
        debug_print( 1, command )
        result =  os.system( command )
        if result != 0:
            print "Error in executing: " + command
        return result

    def __remove_fort_unit(self, unit):
        fort = "fort.%d" % unit
        fortpath = os.path.join(self.workdir(), fort)
        command = "rm  %s" %  fortpath
        self.__execute( command )

    def __remove_fort(self):
        for unit in self.files.keys():
            self.__remove_fort_unit(unit)

    def modulename(self):
        return self.mod_name

    def defineio(self, unit, mode, omittable=False, generated="", target="*", type="", hint="", comment=""):
        mode = string.lower(mode)
        if mode == "o" or mode == "i" or mode == "s" or mode == "io":
            self.mode[unit] = mode
            self.omittable[unit] = omittable
            self.generated[unit] = generated
            self.target[unit] = target
            self.type[unit] = type
            self.hint[unit] = hint
            self.comment[unit] = comment
        else:
            raise TypeError, "Mode should be one of the following mode: i, o, io, s "

    def undefio(self, unit):
        del self.mode[unit]
        del self.omittable[unit]
        del self.generated[unit]
        del self.target[unit]
        del self.type[unit]
        del self.hint[unit]
        del self.comment[unit]

    def attach(self, unit, sgp_file):
        # check unit ids
        if self.files.has_key(unit):
            raise IOError, "Unit %d is already attached." % unit

        if not self.mode.has_key(unit):
            raise IOError, "Unit %d is not defined." % unit

        # input unit ?
        if self.mode[unit] == "i" or self.mode[unit] == "io":
            if sgp_file.is_captured():
                sgp_file.restore()
```

```
            if not sgp_file.exists():
                raise IOError, "File not found: " + sgp_file.pathname()
        else:
            path = os.path.split( sgp_file.pathname() )
            if path[0] != "":
                if not os.path.isdir( sgp_file.dirname() ):
                    raise IOError, "Directory not found: " + sgp_file.dirname()

        fort = "fort.%d" % unit
        fortpath = os.path.join(self.workdir(), fort)
        command = "ln -sf %s  %s" % (sgp_file.pathname(), fortpath)
        self.__execute( command )
        self.files[unit] = sgp_file

    def getfile(self,unit):
        try:
            return copy.copy(self.files[unit])
        except KeyError:
            raise IOError, "file (unit %d) is not yet generated or prepared!" % unit


    def units(self, mode="" ):
        if mode == "":
            return self.mode.keys()
        else:
            dat = []
            for unit in self.mode.keys():
                if unit != 5 and unit != 6:
                    if mode == self.mode[unit]:
                        dat.append(unit)
            return dat

    def list(self):
        return self.getfile(6)


    def outfiles(self):
        dat = []
        all_units = self.mode.keys()
        all_units.sort()
        outfile_units = []
        debug_print( 2, all_units )
        for unit in all_units:
            if unit != 6 and self.mode[unit] == "o":  # or self.mode[unit] == "io":
                ofile = self.getfile(unit)
                dat.append(ofile)
                outfile_units.append(unit)

        debug_print( 2, "Outfile units: ", str(outfile_units) )
        return dat


    def inputs(self, _list):
        if type(_list) == type([]):
            file_list = flatten(_list)
        else:
            file_list = [ _list ]

        # delete all file obj for input
        self.files = {}

        # defined I/O units for the system modules
        io_units = self.mode.keys()
        io_units.sort()


        debug_print( 1, "Inputed Files:" )
        for file in file_list:
            debug_print( 1, "¥t", "  Gen:", file.generate_code(), " Type:", file.filetype(), " Case:", file.casename())

        debug_print( 1, "In_units:", io_units )

        # reorder non_omittable units first, then omittable
        non_omittable_list = []
        omittable_list = []
        for unit in io_units:
            if self.omittable[unit]:
                omittable_list.append(unit)
            else:
                non_omittable_list.append(unit)

        debug_print( 1, "Non_omittable", non_omittable_list, "  Omittable", omittable_list )
```

```
        io_units = non_omittable_list + omittable_list

        # matching between files and Fortran units
        for unit in io_units:
            if self.mode[unit] == "i" or self.mode[unit] == "io":
                match_file = []
                for fileobj in file_list:
                    if self.type[unit] == fileobj.filetype() or self.type[unit] == '*':
                        for code in string.split(self.generated[unit], ','):
                            if code == fileobj.generate_code() or code == '*':
                                match_file.append( fileobj )
                                break

                if len(match_file) > 1:
                    candidates = copy.copy(match_file)
                    match_file=[]
                    for fileobj in candidates:
                        case = fileobj.casename()
                        if case.find( self.hint[unit] ) > 0:  #if case has a hint string (eg. "BOC" or "EOC" )
                            match_file.append(fileobj)

                if len(match_file) == 0:
                    if not self.omittable[unit]:
                        raise IOError, "No matched files found for Unit " + str(unit) + ": gencode=" + self.generated[unit]
+ " and type=" + self.type[unit]
                else:
                    if len(match_file) == 1:
                        debug_print( 1, "Unit: ", unit, "(", self.mode[unit], " )  ===>",
                                    "  Gen:", match_file[0].generate_code(),
                                    " Type:", match_file[0].filetype(),
                                    " Case:", match_file[0].casename() )
                        self.attach(unit, match_file[0])
                        file_list.remove(match_file[0])
                    else:
                        raise IOError, "Multiple files are being assigned to Unit " + str(unit)


    def run(self, case=""):
        # go into working directory
        command = self.full_path_name

        # set case name
        if case != "":
            self.set_casename( case )

        # input files
        for unit in  self.mode.keys():
            m = self.mode[unit]
            if m == "i" or m == "io":
                if not self.files.has_key(unit) and self.omittable[unit] == False:
                    raise IOError, "Unit %d is not yet specified" % unit

        if self.files.has_key(5):
            command += " < fort.5"

        # output files
        for unit in  self.mode.keys():
            m = self.mode[unit]
            if m == "o" or m == "s" or m == "io":
                if not self.files.has_key(unit):
                    if self.type[unit] == "MESH_FLUX":   # tempo
                        binary_flag = True
                    else:
                        binary_flag = False

                    self.attach(unit, DataFile(protect=False, binary=binary_flag))

                self.files[unit].set_generate_code( self.modulename() )
                self.files[unit].set_target_code( self.target[unit] )
                self.files[unit].set_filetype( self.type[unit] )
                self.files[unit].set_casename( self.casename() )

        if self.files.has_key(6):
            command += " > fort.6"


        # debug
        units = self.files.keys()
        units.sort()
        for u in units:
            debug_print(2, "Unit: ", u, " Type:", self.mode[u], " fort.", u, "---> ", self.files[u].pathname() )
```

```
# save the original DataBase object
db_org = {}
for unit in self.type.keys():
    if self.type[unit] == "DATA_BASE":   # this implementation shoule be refactored
        db_org[unit] = copy.deepcopy(self.files[unit])
        db_org[unit].capture()
        db_org[unit].copy_mode()

# execute
print "Executing ", self.mod_name,
debug_print( 1, "=== EXECUTING", self.mod_name, "===" )
os.chdir( self.workdir() )
ret_code =  self.__execute( command)


# remove fort.?
self.__remove_fort()

# capture all files
for unit in self.files.keys():
    m = self.mode[unit]
    if m == "s":
        self.files[unit].purge()
    else:
        debug_print( 2, "Unit: ", unit )
        self.files[unit].capture(delete=True)

# update and replace DataBase
for unit in self.type.keys():
    if self.type[unit] == "DATA_BASE":    # this implementation shoule be refactored

        if sagep.config.debug_level >= 2:
            db_org[unit].show()
            self.files[unit].show()

        diff = self.files[unit].diff_from( db_org[unit] )
        debug_print( 2, "Difference in two DataBase files" )
        if sagep.config.debug_level >= 2:
            pprint(diff)

        test = copy.deepcopy( db_org[unit] )
        test.restore(path="/tmp/database.before")

        db_org[unit].setEntryBlocks(diff) # update the original DataBase obj.
        self.files[unit] = db_org[unit]   # replace the old DataBase with new one

        test = copy.deepcopy( self.files[unit] )
        test.restore(path="/tmp/database")

        self.files[unit].sync()          # syncronize

print " ... done"
return ret_code
```

図 A2-2 CardFile.py

```
from sagep.File.DataFile import *

class CardFile(DataFile):

    def __init__(self, text=""):
        DataFile.__init__(self, protect=False)
        if text != "":
            self.load(text)
        self.set_filetype( "USER_CARD" )


    def write(self, data, binary_mode=False):
        for line in range(len(data)):
            # print "%d: %s" % (line, data[line] )
            if data[line][-1] != '¥n':
                data[line] = data[line] + '¥n'

        try:
            self.file.writelines(data)
        except IOError:
            errout( "Error occured in writing to " + self.full_path_name )

    def load(self, data):
        if not self.is_opened():
            self.create()
        self.write(data)
        self.close()
```

図 A2-3 Config.py

```
# Debug Level ( 0: production   1: simple   2: detail )
debug_level = 0

# Langulage ( "Japanese" or "English" )
language = "Japanese"

# reference directory
refi_dir = "/project/sagep-burn/"

# Working directory
work_dir = "/tmp"

# Location of executables
prefix = "/project/sagep-burn/H13-Na-MOX.C3/CODE/"
convmic   = prefix + "sagep-burn/LM/convmic.lm"
citation  = prefix + "sagep-burn/LM/citation.lm"
fire1     = prefix + "sagep-burn/LM/fire1.lm"
fire2     = prefix + "sagep-burn/LM/fire2.lm"
fire3     = prefix + "sagep-burn/LM/fire3.lm"
nsini     = prefix + "sagep-burn/LM/nsini.lm"
nsini2    = prefix + "sagep-burn/LM/nsini2.lm"
nsini3    = prefix + "sagep-burn/LM/nsini3.lm"
nsini4    = prefix + "sagep-burn/LM/nsini4.lm"
nsini5    = prefix + "sagep-burn/LM/nsini5.lm"
nsini6    = prefix + "sagep-burn/LM/nsini6.lm"
nsjump    = prefix + "sagep-burn/LM/nsjump.lm"
pdsedit   = prefix + "sagep-burn/LM/pdsedt.lm"
refuel    = prefix + "sagep-burn/LM/refuel.lm"
refuel2   = prefix + "sagep-burn/LM/refuel2.lm"
refuel3   = prefix + "sagep-burn/LM/refuel3.lm"
sagep     = prefix + "sagep-burn/LM/sagep93.lm"
scgive    = prefix + "sagep-burn/LM/scgive.lm"
scgive_br = prefix + "sagep-burn/LM/scgive_br.lm"
startup   = prefix + "sagep-burn/LM/startup.lm"
flux     = prefix + "sagep-burn/LM/flxdbl.lm"
```

図 A2-4 DataBase.py

```
from sagep.FortranFormat import *
from sagep.File.DataFile import *
from string import rstrip

class DataBase(DataFile):
    """
    DataBase class manupulates the database that is used by CITATION, SAGEP, SCGIVE
    """

    def __init__(self, full_path_name=""):
        DataFile.__init__(self, full_path_name, protect=False, capture=True, binary=False, fixed_path=True)
        self.set_generate_code( "CITATION" )
        self.set_filetype( "DATA_BASE" )

        # format
        self.format_table = {}
        self.columns_table = {}


    def defineFormat(self, entry, columns, format):
        self.columns_table[entry] = columns
        self.format_table[entry] = format


    def formatString(self, format, columns, data):
        str_data = []
        lines = len(data) / columns

        frt_format = FortranFormat( str(columns) + format )
        i=0
        for i in range(lines):
            line = FortranLine( data[(columns*i):(columns*(i+1))], frt_format )
            str_data.append( line.text )

        if len(data) % columns > 0:
            frt_format = FortranFormat( str( len(data) % columns) + format )
            line = FortranLine( data[(columns*lines):], frt_format )
            str_data.append( line.text )

        return map(lambda x: rstrip(x) + '¥n', str_data )


    def hasEntry(self, entry ):
        "Check whether the database has the specified block or not."
        entry_stripped = rstrip(entry)
        n = len(entry_stripped)
        for i in range( len(self.data) ):
            if self.data[i][:n] == entry_stripped:
                break

        if i != len(self.data)-1:
            return i  # found
        else:
            return -1


    def getEntry(self, entry, convert=True):
        """
        Get data set for the specified entry block. If size of data is only one,
        it returns a float value. If the size is greater than one, returns array of float values.
        """
        pos = self.hasEntry(entry)
        debug_print( 2, 'pos', pos )
        if pos >= 0:   # entry found
            data = []
            i = 1
            while True:
                debug_print( 2, "***",self.data[pos+i].split()  )
                if self.data[pos+i][0] == ' ':
                    data += self.data[pos+i].split()
                    i += 1
                else:
                    break

            if convert:
                final_data =  map(float, data)
            else:
                final_data = data
            if len(final_data) == 1:
```

```
            return final_data[0]
        else:
            return final_data

    else:  # entry not found
        return ""


def removeEntry(self, entry):
    """Remove a specified entry block from the database. If specified entry exists, the position
    of the deleted entry will be returned. If not, -1 will be returned."""
    pos = self.hasEntry(entry)
    if pos >= 0:   # has entry
        del self.data[pos]
        while True:
            if self.data[pos][0] == ' ':
                del self.data[pos]
            else:
                break
    return pos


def insertEntry(self, entry, data, columns, format, pos=-1):
    "Insert entry data at the specified position. "
    str_data_list = self.formatString( format, columns, data )
    debug_print( 2, "str_data_list", str_data_list )
    self.data[pos:] = [ entry ] + str_data_list + self.data[pos:]


def setEntry(self, entry, data):
    """Set entry data into the database. If database already has the entry, it will
    replace data with the new one. If not, the specified data is added at the
    end of the database.
    """
    if not self.format_table.has_key(entry):
        format = "A80"
        columns = 1
    else:
        format = self.format_table[entry]
        columns = self.columns_table[entry]

    pos = self.hasEntry(entry)
    if pos >= 0:  # entry already exists
        self.removeEntry(entry)
    self.insertEntry(entry, data, columns, format, pos)


def setEntryBlocks(self, entry_block):
    """Set entry blocks which are pairs of entry and its data.
    A entry_block is a list of string lines
    """
    i = 0
    debug_print( 2, entry_block )
    while True:
        debug_print( 2, i, entry_block[i][0] )
        if  entry_block[i][0] !=' ':  # entry
            entry = rstrip(entry_block[i]) + '¥n'
            debug_print( 2, entry )
            if entry == 'END¥n':
                break
            i += 1

            data = []
            for j in range( len(entry_block) - i ):
                if entry_block[i+j][0] == ' ':
                    data.append( rstrip(entry_block[i+j]) )
                else:
                    break;

            debug_print( 2, "entry:", entry )
            debug_print( 2, "data", data )
            self.setEntry(entry, data)
            i += j
        if i > len(entry_block):
            break

def sync(self):
    self.restore()
    self.capture()


def diff_from(self, other, filter='+' ):
```

```
        data = DataFile.diff_from(self, other, filter)
        data.append( 'END¥n' )
        return data
```

図 A2-5 DataFile.py

```
from sagep.File.GeneralFile import *
from  difflib import *
import string
from difflib import *
from pprint import *

class DataFile(GeneralFile):
    "DataFile manages general data file and it encapsulates any kinds of files into object."

    def __init__(self, full_path_name="", capture=False, protect=False, binary=False, fixed_path=False):
        GeneralFile.__init__(self,full_path_name)
        if full_path_name != "":
            protect = True
        self.property = {}
        self.open_flag = False
        self.capture_flag = False
        self.protect_flag = protect
        self.binary_flag  = binary
        self.restore_path = ""
        self.fixed_path_flag = False
        if fixed_path and full_path_name != "":
            self.fixed_path_flag = True
            self.restore_path = full_path_name
        self.data = ""
        if capture:
            self.capture()


    def filename(self):
        "returns file name"
        return os.path.basename(self.full_path_name)

    def dirname(self):
        "returns directory name of the file encapsulated"
        return os.path.dirname(self.full_path_name)

    def create(self):
        debug_print( 1, "Creating ", self.full_path_name )
        try:
            self.file = open( self.full_path_name, 'w' )
            self.open_flag = True
            return True
        except IOError:
            print "Cannot create file " + self.full_path_name
            return False

    def purge(self, force=False):
        if self.is_protected():
            if not force:
                return False
        try:
            debug_print( 1, "removing ", self.full_path_name )
            if self.is_opened():
                self.close()
#           os.remove( self.full_path_name )
            if not self.fixed_path_flag:
                self.full_path_name = ""
            return True
        except IOError:
            print "Cannot purge file " +  self.full_path_name
            return False


    def open(self):
        try:
            self.file = open( self.full_path_name, 'r' )
            self.open_flag = True
            return True
        except IOError:
            print "Cannot open file " + self.full_path_name
            return False

    def is_opened(self):
        return self.open_flag

    def close(self):
        try:
            if self.is_opened():
                self.file.close()
```

```
            self.open_flag = False
        return True
    except IOError:
        print  "Close error occured"
        return False

def read(self, binary_mode=False):
    debug_print( 2, "opened?: ", self.is_opened() )
    if not self.is_opened():
        self.open()
    try:
        if self.binary_flag or binary_mode:
            data = self.file.read()
        else:
            data = self.file.readlines()  # read whole data
        self.close()
        return data

    except IOError:
        raise "Error occured in reading from " + self.full_path_name
        self.close()
        return ""


def write(self, data, binary_mode=False):
    if self.is_protected():
        print "Write failed since ", self.full_path_name, " is protected."
        return False

    if self.binary_flag or binary_mode:
        try:
            self.file.write(data)
        except IOError:
            raise  "Error occured in writing to " + self.full_path_name + " in binary mode"

    else:
        for line in range(len(data)):
            debug_print( 2, line, data[line] )
            if data[line][-1] != '¥n':
                data[line] = data[line] + '¥n'
        try:
            self.file.writelines(data)
        except IOError:
            raise "Error occured in writing to " + self.full_path_name

    self.close()
    return True


def has_same_contents_with(self, other, exception=""):
    data = []

    self_captured = self.is_captured()
    other_captured = other.is_captured()
    if self_captured:
        self.restore(temporary=True)
    if other_captured:
        other.restore(temporary=True)

    if exception != "":
        binary_flag = False
    else:
        binary_flag = True

    data.append( self.read(binary_flag) )
    data.append( other.read(binary_flag) )

    if exception != "":
        for d in data:
            for i in range(len(d)):
                if d[i].find( exception ) > 0: # found
                    d[i] = ""

    if self_captured:
        self.capture(delete=True)
    if other_captured:
        other.capture(delete=True)

    return data[0] == data[1]


def dump(self, filename, save_contents=True):
```

```
        debug_print( 1, "Dumping", self.__class__.__name__, "object to Pickled file..." )

        self.contents = self.read()
        del self.file

        fh = open(filename, "w")
        cPickle.dump(self, fh, 1)


    def load(self, filename):
        debug_print( 1, "Loading", self.__class__.__name__, "object from Pickled file..." )

        fh = open( filename, "rb" )
        tmp =  cPickle.load(fh)
        self = copy.copy(tmp)
        create_flag = True

        if self.exists():
            file_contents = self.read()
            if self.contents == file_contents:
                create_flag = False

        if create_flag:
            tmpname = tempfile.mkstemp(prefix="sagep", dir=self.workdir())
            self.full_path_name = os.path.join(self.workdir(), tmpname[1] )
            self.create()
            self.file.writelines( self.contents )
            self.close()
            del self.file

        # clear contents
        self.contents = ""

        return self

    def setProperty(self, key, value):
        self.property[string.upper(key)] = value

    def getProperty(self,key):
        return self.property[string.upper(key)]

    def delProperty(self,key):
        del self.property[string.upper(key)]

    def property_list(self):
        list = self.property.keys()
        list.sort()
        return list

    def getProperties(self):
        "This method returns the internal dictionary of properties."
        return self.property

    def setProperties(self, prop):
        "This method register prop into the internal dictionary."
        self.property = prop

    def is_captured(self):
        "If a file has been captured, it returns True."
        return self.capture_flag

    def is_protected(self):
        "If true, the file indicated by the file pointer will not be removed."
        return self.protect_flag

    def capture(self, delete=False):
        """
        This method captures whole contents in the file that is currently indicated with the file pointer.
        If delete flag is enabled, the actual file will be deleted. After capturing, the file pointer is reset.
        """
        if self.is_captured():
            return

        debug_print( 1, "Capturing from " + self.filename() )

        self.data = self.read(binary_mode=self.binary_flag)

        self.protect_flag = False                    # After capture, a temporary file can be removed

        self.capture_flag = True
        if delete and self.protect_flag == False:
            self.purge()
```

```
        self.close()

        self.full_path_name = ""      # clear file name pointer


    def restore(self, path="", temporary=False):

        if temporary:
            GeneralFile.__init__(self, "", init=False )  # generate temporary file
        else:
            GeneralFile.__init__(self, self.restore_path, init=False )

        if path != "":
            GeneralFile.__init__(self, path, init=False )


        self.create()
        self.write(self.data, binary_mode=self.binary_flag)              # write down to the file in binary mode
        self.close()
        self.capture_flag = False
        debug_print( 1, "Restored to " + self.full_path_name )

    def copy_mode(self):
        self.full_path_name = ""
        self.fixed_path_flag = False
###         self.protect_flag = False


    def diff_from(self, other, filter='' ):

        self_captured = self.is_captured()
        other_captured = other.is_captured()
        debug_print( 2, self_captured, other_captured )

        debug_print( 2, "diff_from:  self, other : ", self_captured, other_captured )
        debug_print( 2, "  self:", self.full_path_name )
        debug_print( 2, "  other:", other.full_path_name )
        if self_captured:
            self.restore(temporary=True)
        if other_captured:
            other.restore(temporary=True)

        d1 =  other.read(self.binary_flag)
        d2 =  self.read(self.binary_flag)

        diff = list(ndiff(d1, d2))
        if sagep.config.debug_level >= 2:
            pprint(diff)

        if self_captured:
            self.capture(delete=True)
        if other_captured:
            other.capture(delete=True)

        data = []
        for i in diff:
            if i[0] == '+' or i[0] == '-':
                if filter != '':
                    if i[0] == filter:
                        data.append( i[2:] )
                else:
                    data.append(i[2:] )

        return data


# aliases

    save = dump



class RefFile(DataFile):
    "RefFile is a write-protected DataFile object for reference purpose"

    def __init__(self, full_path_name, capture=True, protect=True, binary=False):
        DataFile.__init__(self,full_path_name, capture, protect, binary)


ListFile = RefFile
```

図 A2-6 FortranFormat.py

```
# This module defines a class that handles I/O using
# Fortran-compatible format specifications.
#
#
# Warning: Fortran formatting is a complex business and I don't
# claim that this module works for anything complicated. It knows
# only the most frequent formatting options. Known limitations:
#
# 1) Only A, D, E, F, G, I, and X formats are supported (plus string constants
#    for output).
# 2) No direct support for complex numbers. You have to split them into
#    real and imaginary parts before output, and for input you get
#    two float numbers anyway.
#
#
# Written by Konrad Hinsen <hinsen@ibs.ibs.fr>
# last revision: 1996-9-20
#

"""Fortran-compatible input/output

This module provides two classes that aid in reading and writing
Fortran-formatted text files. Only a subset of formatting options
is supported: A, D, E, F, G, I, and X formats, plus string constants
for output. Repetition (e.g. 4I5 or 3(1X,A4)) is supported. Complex
numbers are not supported; you have to treat real and imaginary parts
separately.

Examples:
=========

Input:
------
>> s = '   59999'
>> format = FortranFormat('2I4')
>> line = FortranLine(s, format)
>> line[0]
5
>> line[1]
9999

Output:
-------
>> format = FortranFormat('2D15.5')
>> line = FortranLine([3.1415926, 2.71828], format)
>> line.text
'    3.14159D+00    2.71828D+00'

The second argumet to FortranLine can be a format object or a
string (that is then converted into a format object). If the
same format is to be used several times, it is more efficient
to convert it into a format object once - parsing the format
string is a relatively expensive operation.
"""

import string

#
# The class FortranLine represents a single line of input/output,
# which can be accessed as text or as a list of items.
#
class FortranLine:

    def __init__(self, line, format, length = 80):
        if type(line) == type(''):
            self.text = line
            self.data = None
        else:
            self.text = None
            self.data = line
        if type(format) == type(''):
            self.format = FortranFormat(format)
        else:
            self.format = format
        self.length = length
        if self.text is None:
            self._output()
        if self.data is None:
            self._input()
```

```
    def __len__(self):
        return len(self.data)

    def __getitem__(self, i):
        return self.data[i]

    def __getslice__(self, i, j):
        return self.data[i:j]

    def isBlank(self):
        return len(string.strip(self.text)) == 0

    def _input(self):
        text = self.text
        if len(text) < self.length: text = text + (self.length-len(text))*' '
        self.data = []
        for field in self.format:
            i = field[1]
            s = text[:i]
            text = text[i:]
            type = field[0]
            value = None
            if type == 'A':
                value = s
            elif type == 'I':
                s = string.strip(s)
                if len(s) == 0:
                    value = 0
                else:
                    value = string.atoi(s)
            elif type == 'D' or type == 'E' or type == 'F' or type == 'G':
                s = string.strip(s)
                if len(s) == 0:
                    value = 0.
                else:
                    value = string.atof(s)
            if value is not None:
                self.data.append(value)

    def _output(self):
        data = self.data
        self.text = ''
        for field in self.format:
            type = field[0]
            if type == "'":
                self.text = self.text + field[1]
            elif type == 'X':
                self.text = self.text + field[1]*' '
            else: # fields that use input data
                length = field[1]
                if len(field) > 2: fraction = field[2]
                value = data[0]
                data = data[1:]
                if type == 'A':
                    self.text = self.text + (value+length*' ')[:length]
                else: # numeric fields
                    if type == 'I':
                        s = `value`
                    elif type == 'D':
                        s = ('%'+`length`+'.'+`fraction`+'e') % value
                        n = string.find(s, 'e')
                        s = s[:n] + 'D' + s[n+1:]
                    elif type == 'E':
                        s = ('%'+`length`+'.'+`fraction`+'e') % value
                    elif type == 'F':
                        s = ('%'+`length`+'.'+`fraction`+'f') % value
                    elif type == 'G':
                        s = ('%'+`length`+'.'+`fraction`+'g') % value
                    else:
                        raise ValueError, 'Not yet implemented'
                    s = string.upper(s)
                    self.text = self.text + ((length*' ')+s)[-length:]

#
# The class FortranFormat represents a format specification.
# It ought to work for correct specifications, but there is
# little error checking.
#
class FortranFormat:

    def __init__(self, format, nested = 0):
```

```
        fields = []
        format = string.strip(format)
        while format and format[0] != ')':
            n = 0
            while format[0] in string.digits:
                n = 10*n + string.atoi(format[0])
                format = format[1:]
            if n == 0: n = 1
            type = string.upper(format[0])
            if type == "'":
                eof = string.find(format, "'", 1)
                text = format[1:eof]
                format = format[eof+1:]
            else:
                format = string.strip(format[1:])
            if type == '(':
                subformat = FortranFormat(format, 1)
                fields = fields + n*subformat.fields
                format = subformat.rest
                eof = string.find(format, ',')
                if eof >= 0:
                    format = format[eof+1:]
            else:
                eof = string.find(format, ',')
                if eof >= 0:
                    field = format[:eof]
                    format = format[eof+1:]
                else:
                    eof = string.find(format, ')')
                    if eof >= 0:
                        field = format[:eof]
                        format = format[eof+1:]
                    else:
                        field = format
                        format = ''
                if type == "'":
                    field = (type, text)
                else:
                    dot = string.find(field, '.')
                    if dot > 0:
                        length = string.atoi(field[:dot])
                        fraction = string.atoi(field[dot+1:])
                        field = (type, length, fraction)
                    else:
                        if field:
                            length = string.atoi(field)
                        else:
                            length = 1
                        field = (type, length)
                fields = fields + n*[field]
        self.fields = fields
        if nested:
            self.rest = format

    def __len__(self):
        return len(self.fields)

    def __getitem__(self, i):
        return self.fields[i]


if __name__ == '__main__':
    f = FortranFormat("'!!',D10.3,F10.3,G10.3,'!!'")
    l = FortranLine([1.5707963, 3.14159265358, 2.71828], f)
    print l.text
```

図 A2-7 GeneralFile.py

```
from  sagep.util import *
import tempfile
import os
import os.path
import sys
import copy

class GeneralFile(SerializableAndVisible):

    def __init__(self, full_path_name="", init=True):
        self.full_path_name = full_path_name
        self.is_temporary = False
        if self.full_path_name == "":
            tmpname = tempfile.mkstemp(prefix="sagep", dir=self.workdir())
            self.full_path_name = os.path.join(self.workdir(), tmpname[1] )
            self.is_temporary = True

        if init:
            self.openflag = False
            self.gen_code = ""
            self.type = ""
            self.target_code = ""
            self.case = ""
            self.contents = ""

    def __del__(self):
        if self.is_opened():
            self.close()
#        if self.is_temporary:
#            os.remove( self.full_path_name )


    def setname(self, full_path_name):
        self.full_path_name = full_path_name

    def set_generate_code(self, generate_code ):
        self.gen_code = generate_code

    def set_target_code(self, target_code ):
        self.target_code = target_code

    def set_filetype(self, file_type):
        self.type = file_type

    def generate_code(self):
        return self.gen_code

    def target_code(self):
        return self.target_code

    def filetype(self):
        return self.type

    def set_casename(self, name):
        self.case = name

    def casename(self):
        return self.case

    def setworkdir(self, workdir):
        GeneralFile.work_dir = workdir

    def workdir(self):
        try:
            return GeneralFile.work_dir
        except AttributeError:
            debug_print( 2, "Working directory is not defined"  )
            return "/tmp"


    def pathname(self):
        return self.full_path_name

    def exists(self):
        return os.path.exists(self.full_path_name)
```

図 A2-8 MicroFile.py

```
from sagep.File.DataFile import *

class MicroFile(DataFile):

    def __init__(self, full_path_name=""):
        DataFile.__init__(self, full_path_name, protect=True, capture=True, binary=True)
        self.set_generate_code( "JOINT" )
        self.set_filetype( "CITATION_MICRO_XS" )


class SagepMicroFile(DataFile):

    def __init__(self, full_path_name=""):
        DataFile.__init__(self, full_path_name, protect=True, capture=True, binary=True)
        self.set_generate_code( "CONVMIC" )
        self.set_filetype( "SAGEP-BURN_MICRO_XS" )
        return os.path.exists(self.full_path_name)
```

図 A2-9 ModuleFile.py

```
from sagep.File.ExecutableFile import *
from sagep.File.CardFile import *
import sagep.config
from os.path import *
import string

class Convmic(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.convmic, case )

        # input file(s)
        self.defineio(unit=1, mode="i", generated="JOINT", type="CITATION_MICRO_XS", comment="CITATION Microscopic XS
File")

        # output file(s)
        self.defineio(unit=3, mode="o", type = "SAGEP-BURN_MICRO_XS", comment="SAGEP-BURN Microscopic XS File")

        # scratch files(s)
        self.defineio(unit=2, mode="s", type = "UNKNOWN",  comment="Working File (Unknown)")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


class Refuel(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.refuel, case )

        # input file(s)
        self.defineio(unit=10, mode="i", omittable=True, generated="REFUEL", type="REFUEL_CONTROL_DATA", comment="REFUEL
Control Data")
        self.defineio(unit=21, mode="i", omittable=True, generated="FIRE2", type="NUMBER_DENSITY_FINAL", comment="Number
Density Data (Final)")

        # output file(s)
        self.defineio(unit=11, mode="o", target="REFUEL", type="REFUEL_CONTROL_DATA", comment="REFUEL Control Data")
        self.defineio(unit=12, mode="o", target="FIRE1",   type="NUMBER_DENSITY_REGION", comment="Number Density Data
(Region Average)")
        self.defineio(unit=13, mode="o", target="FIRE2",   type="NUMBER_DENSITY_ASSEMBLY", comment="Number Density Data
(Assembly Average)")
        self.defineio(unit=14, mode="o", target="*",       type="NUMBER_DENSITY", comment="Number Density Data (Unknown)")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


class Startup(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.startup, case )

        # input file(s)
        self.defineio(unit=   8,   mode="i",   omittable=False,   generated="CONVMIC",   type="SAGEP-BURN_MICRO_XS",
comment="SAGEP-BURN Microscopic XS File")
        self.defineio(unit=11,   mode="i",   omittable=False,   generated="REFUEL",       type="NUMBER_DENSITY_REGION",
comment="Number Density (Region Average)")

        # output file(s)
        self.defineio(unit= 3, mode="o", target="CITATION", type="MACRO_XS", comment="Macroscopic XS File")

        # scratch files(s)
        self.defineio(unit=15, mode="s", target="*",        type="MICRO_XS", comment="Wrling File")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


class Citation(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.citation, case )

        # input file(s)
        self.defineio(unit= 3, mode="i", omittable=True,  generated="CITATION", type="MESH_FLUX",  comment="Mesh-wise
Flux")
        self.defineio(unit=10, mode="i", omittable=False, generated="STARTUP",  type="MACRO_XS", comment="Macroscopic XS
File")
```

```
        # output file(s)
        self.defineio(unit= 2, mode="o", target="SAGEP93", type="MESH_FLUX", comment="Mesh-wise Flux")
        self.defineio(unit=109, mode="o", target="CITATION", type="AXIAL_BUCKLING", comment="Axial Buckling")
        self.defineio(unit=20, mode="o", target="NSINI?,NSJUNP", type="LEAKAGE_TERM", comment="Leakage Term")

        # input/output file(s)
        self.defineio(unit=51, mode="io", omittable=False, generated="CITATION,*", type="DATA_BASE", comment="Data Base
File")

        # scratch files(s)
        self.defineio(unit= 8, mode="s", type="REGION_FLUX", comment="Region-wise Flux")
        self.defineio(unit=18, mode="s", type="REGION_FLUX_AVE", comment="Region-wise Averaged Flux")
        self.defineio(unit=40, mode="s", type="MESH_VOLUME", comment="Mesh-wise Volume")
        self.defineio(unit=41, mode="s", type="RADIAL_DIFF_COEF", comment="Radial Diffusion Coefficient")

       # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


    def run(self, case=""):
        inp = self.files[5].read()
        val = string.split(inp[1])  # second line
        debug_print( 1, val )
        mode = "_FORWARD"
        if val[8] == '1':
            mode = "_ADJOINT"

        debug_print( 1, "casename = ", case + mode )
        ExecutableFile.run(self,case + mode)

        dim = string.split(inp[7])  #  NX, NY, NZ, KG
        for outunit in 2,109,20:
            self.files[outunit].setProperty( "NX", dim[0] )
            self.files[outunit].setProperty( "NY", dim[1] )
            self.files[outunit].setProperty( "NZ", dim[2] )
            self.files[outunit].setProperty( "NG", dim[3] )


class Fire1(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.fire1, case )

        # input file(s)
        self.defineio(unit=   8,   mode="i",   omittable=False,   generated="CONVMIC",   type="SAGEP-BURN_MICRO_XS",
comment="SAGEP-BURN Microscopic XS File")
        self.defineio(unit=11, mode="i", generated="REFUEL", type="NUMBER_DENSITY_REGION", hint="BOC", comment="Number
Density Data (Region Average)")
        self.defineio(unit=15,  mode="i",  omittable=True,  generated="REFUEL2",  type="NUMBER_DENSITY_REGION_ADJOINT",
comment="Number Density Data (Region Average, Adjoint)" )

        # output file(s)
        self.defineio(unit=12, mode="o", target="REFUEL2", type="NUMBER_DENSITY_ADJOINT_FINAL", comment="Number Density
for Final Step (Adjoint)") # for adjoint calc.
        self.defineio(unit=30, mode="o", target="SCGIVE", type="NUMBER_DENSITY_EACH", comment="Number Density for Each
Substep")
        self.defineio(unit=31, mode="o", target="SCGIVE", type="NUMBER_DENSITY_ADJOINT_EACH", comment="Number Density for
Each Substep (Adjoint)")
        self.defineio(unit=34, mode="o", target="SCGIVE", type="KAPPA_SIGF_XS", comment="Kappa-Sigf Cross Sections")
        self.defineio(unit=35, mode="o", target="SCGIVE", type="MATRIX_T", comment="Matrix T Conponents")
        self.defineio(unit=36, mode="o", target="SCGIVE", type="MATRIX_R", comment="Matrix R Conponents")

        # input/output file(s)
        self.defineio(unit=51,    mode="io",    omittable=False,    generated="CITATION,*",    target="SCGIVE,FIRE22",
type="DATA_BASE", comment="Data Base File")


       # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


class Fire2(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.fire2, case )

        # input file(s)
        self.defineio(unit=   8,   mode="i",   omittable=False,   generated="CONVMIC",   type="SAGEP-BURN_MICRO_XS",
comment="SAGEP-BURN Microscopic XS File")
        self.defineio(unit=11, mode="i", generated="REFUEL", type="NUMBER_DENSITY_ASSEMBLY", comment="Number Density in
Selected Assembly")
        self.defineio(unit=42, mode="i", generated="REFUEL", type="REFUEL_CONTROL_DATA", comment="REFUEL Control Data")
```

```
        # output file(s)
        self.defineio(unit=12, mode="o",    generated="REFUEL2",  type="NUMBER_DENSITY_FINAL",  comment="Forward/Adjoint
Number Density")
        self.defineio(unit=30, mode="o", target="SCGIVE", type="NUMBER_DENSITY_EACH", comment="Number Density for Each
Substep")
        self.defineio(unit=34, mode="o", target="SCGIVE", type="KAPPA_SIGF_XS", comment="Kappa-Sigf Cross Sections")
        self.defineio(unit=35, mode="o", target="SCGIVE", type="MATRIX_T", comment="Matrix T Conponents")
        self.defineio(unit=36, mode="o", target="SCGIVE", type="MATRIX_R", comment="Matrix R Conponents")

        # input/output file(s)
        self.defineio(unit=51,     mode="io",     omittable=False,     generated="CITATION,*",     target="SCGIVE,FIRE2",
type="DATA_BASE", comment="Data Base File")


       # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")



class PDSHandler(ExecutableFile):
    def __init__(self, code, case):
        self.pds_in_dir = ""
        self.pds_out_dir = ""
        ExecutableFile.__init__(self, code, case)


    def set_PDSin(self, dir):
        if not ( os.path.exists(dir) and os.path.isdir(dir) ):
            raise IOError, "PDS directory not found"
        self.pds_in_dir = os.path.join(dir, "")
        os.environ['PDSIN'] = self.pds_in_dir    # for pdsedit
        os.environ['MACREF'] = self.pds_in_dir   # for sagep
        os.environ['MACPERT'] = self.pds_in_dir  # for sagep


    def set_PDSout(self,dir):
        full_path_dir = os.path.join(self.workdir(), dir, "" )
        debug_print(2, "set_PDSout: full_path_dir = ", full_path_dir )
        if not os.path.exists(full_path_dir):
            os.mkdir(full_path_dir)
        if not os.path.isdir(full_path_dir):
            raise IOError, "%s is not directory" % full_pah_dir

        self.pds_out_dir = full_path_dir
        os.environ['PDSOUT'] = self.pds_out_dir


    def clearPDSout(self):
        if self.pds_out_dir:
            debug_print( 2, "clear PDS out dir" )
            if os.path.exists(self.pds_out_dir) and os.path.isdir(self.pds_out_dir):
                os.system( "rm -rf " + self.pds_out_dir )
            os.mkdir(self.pds_out_dir)


    def run(self, case=""):
        if not self.pds_out_dir:
            self.set_PDSout( case )

        if not self.pds_in_dir:
            self.set_PDSin( self.pds_out_dir )

        debug_print( 2, "PDS_in_dir = ", self.pds_in_dir )
        debug_print( 2, "PDS_out_dir = ", self.pds_out_dir )

        ExecutableFile.run(self, case)

        # clear directory settings
        self.pds_out_dir = ""
        self.pds_in_dir = ""



class PDSEdit(PDSHandler):
    def __init__(self, case=""):
        PDSHandler.__init__(self, sagep.config.pdsedit, case)

        # input files(s)
        self.defineio(unit= 8, mode="i", type="DUMMY", comment="Dummy Data")
```

```
        self.defineio(unit=10, mode="i", generated="REFUEL", type="NUMBER_DENSITY_REGION", comment="Number Density Data
(Region Average)")
        self.defineio(unit=11, mode="i", generated="STARTUP", type="MACRO_XS", comment="Macroscopic XS File")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")

    def run(self, case=""):
        if not self.pds_out_dir:
            self.set_PDSout( case )
        self.clearPDSout()
        PDSHandler.run(self,case)


class Flux(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.flux, case )

        # input files(s)
        self.defineio(unit=1, mode="i", generated="CITATION", type="MESH_FLUX", hint="FORWARD", comment="Mesh-wise Flux
(Forward)" )
        self.defineio(unit=2, mode="i", generated="CITATION", type="MESH_FLUX", hint="ADJOINT", comment="Mesh-wise Flux
(Adjoint)" )

        # output file(s)
        self.defineio(unit=11, mode="o", generated="CITATION", type="MESH_FLUX", comment="Mesh-wise Flux (Double Presicion
/ Forward" )
        self.defineio(unit=12, mode="o", generated="CITATION", type="MESH_FLUX", comment="Mesh-wise Flux (Double Presicion
/ Adjoint)" )

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


class Sagep(PDSHandler):
    def __init__(self, case=""):
        PDSHandler.__init__(self, sagep.config.sagep, case)

        # input files(s)
        self.defineio(unit=1, mode="i", generated="CITATION", type="MESH_FLUX", hint="FORWARD", comment="Mesh-wise Flux
(Forward)")
        self.defineio(unit=2, mode="i", generated="CITATION", type="MESH_FLUX", hint="ADJOINT", comment="Mesh-wise Flux
(Adjoint)")
        self.defineio(unit=15,    mode="i",    omittable=True,    generated="REFUEL",    type="NUMBER_DENSITY_REGION",
comment="Number Density Data (Region Average)")
        self.defineio(unit=42, mode="i", omittable=True, generated="SCGIVE,SCGIVEBR", type="SOURCE", comment="Source
Term" )

        # output file(s)
        self.defineio(unit=12, mode="o", type="GENERALIZED_FLUX_ADJOINT", comment="Generalized Adjoint Flux")
        self.defineio(unit=52,  mode="io",  omittable=True,  generated="*",  type="SENS_DIRECT",  comment="Sensitivity
Coefficient Direct Term" )

        # input/output file(s)
        self.defineio(unit=51, mode="io", omittable=False, generated="CITATION,*", type="DATA_BASE", comment="Data Base
File")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


    def run(self,case=""):

        # execute 'Flux'
        flux = Flux(case)

        flux1 = self.files[1]
        flux2 = self.files[2]

        flux1_prop = self.files[1].getProperties()
        flux2_prop = self.files[2].getProperties()

        if flux1_prop != flux2_prop:
            raise TypeError, "Dirrent dimesnsions between 2 Flux files"

        debug_print( 1, flux1_prop )
```

```
        flux_card = "%5s%5s%5s%5s¥n" % ( flux1_prop["NX"], flux1_prop["NY"], flux1_prop["NG"], flux1_prop["NZ"] )
        debug_print( 1, flux_card )

        flux_inp = CardFile( [flux_card] )

        flux.inputs( [flux_inp, flux1, flux2] )
        flux.run(case)
        dbl_flux = flux.outfiles()
        dbl_flux[0].restore()
        dbl_flux[1].restore()


        flux_mix = DataFile()
        flux_mix.set_generate_code( "CITATION" )
        flux_mix.set_filetype( "MESH_FLUX" )
        flux_mix.setProperties( dbl_flux[0].getProperties() )

        command = "cat %s %s > %s" % ( dbl_flux[0].pathname(), dbl_flux[1].pathname(), flux_mix.pathname() )
        debug_print( 1, command )
        os.system( command )

        # debug
        if sagep.config.debug_level >= 1:
            command = "ls -l  %s %s"  % ( dbl_flux[1].pathname(), flux_mix.pathname() )
            os.system( command )

        # debug
        #flux_mix = DataFile( "/project/sagep-burn/H13-Na-MOX.C3/BR/WORK/FLUX.flx" )
        #flux_mix.set_generate_code( "CITATION" )
        #flux_mix.set_filetype( "MESH_FLUX" )


        debug_print( 1, "flux_mix:", flux_mix.pathname() )
        self.defineio(unit=3, mode="i", generated="CITATION", type="MESH_FLUX", comment="Mesh-wise Flux (Double Presicion
/ Forward+Adjoint)" )
        self.attach(3, flux_mix)

        # re-attach unit 5
        my_inp = self.files[5]
        del self.files[5]
        self.attach(5, my_inp)

        # fake and create unit 52, then write "END" card
        file52 = DataFile()
        file52.create()
        file52.set_filetype("SENS_DIRECT")
        file52.write(["END¥n"])
        file52.close()
        self.attach(52, file52)


        # run sagep
        PDSHandler.run(self, case)

        # undefine Unit 3
        self.undefio(3)


class _Nsini(ExecutableFile):
    def __init__(self, case=""):
        # input file(s)
        self.defineio(unit= 2, mode="i", generated="SAGEP", type="GENERALIZED_FLUX_ADJOINT",  comment="Generalized Flux
(Adjoint)")
        self.defineio(unit= 3, mode="i", generated="CITATION",  type="MESH_FLUX", comment="Mesh-wise Flux (Forward)")
        self.defineio(unit=10, mode="i", generated="STARTUP",  type="MACRO_XS", comment="Macroscopic XS File")
        self.defineio(unit=11,  mode="i",  omittable=True,  generated="REFUEL2",    type="NUMBER_DENSITY_REGION_ADJOINT",
comment="Adjoint Number Density Data (Region Average)")
        self.defineio(unit=12,   mode="i",   generated="CONVMIC",      type="SAGEP-BURN_MICRO_XS",   comment="SAGEP-BURN
Microscopic XS File")
        self.defineio(unit=16, mode="i", generated="CITATION",  type="LEAKAGE_TERM", comment="Leakage Term" )
        self.defineio(unit=51, mode="i", generated="CITATION,*", type="DATA_BASE", comment="Data Base File")

        # output file(s)
        self.defineio(unit=20, mode="o", target="REFUEL2", type="NUMBER_DENSITY_ADJOINT_FINAL", comment="Initial Number
Density" )

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


class Nsini4(_Nsini):
```

```
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.nsini4, case )
        _Nsini.__init__(self, case)


class Nsjump(_Nsini):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.nsjump, case )
        _Nsini.__init__(self, case)


class Refuel2(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.refuel2, case )

        # input file(s)
        self.defineio(unit=10,    mode="i",    omittable=True,    generated="REFUEL2",    type="REFUEL2_CONTROL_DATA",
comment="REFUEL2 Control Data")
        self.defineio(unit=21,        mode="i",        omittable=True,        generated="NSINI4,FIRE1,FIRE2,NSJUMP",
type="NUMBER_DENSITY_ADJOINT_FINAL", comment="Adjoint Number Density Data (Final)")

        # output file(s)
        self.defineio(unit=11, mode="o", target="REFUEL2", type="REFUEL2_CONTROL_DATA", comment="REFUEL2 Control Data")
        self.defineio(unit=12, mode="o", target="FIRE1",  type="NUMBER_DENSITY_REGION_ADJOINT", comment="Adjoint Number
Density Data (Region Average)")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


class Scgive(ExecutableFile):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.scgive, case )
        self._defineio()

    def _defineio(self):
        # input file(s)
        self.defineio(unit=8, mode="i", generated="CONVMIC",  type="SAGEP-BURN_MICRO_XS", comment="SAGEP-BURN Microscopic
XS File")
        self.defineio(unit=30,  mode="i",  generated="FIRE1",  type="NUMBER_DENSITY_EACH",  hint="BOC",  comment="Number
Density for Each Substep")
        self.defineio(unit=31,    mode="i",    generated="FIRE1",    type="NUMBER_DENSITY_ADJOINT_EACH",    hint="EOC",
comment="Number Density for Each Substep (Adjoint)")
        self.defineio(unit=34, mode="i", generated="FIRE1", type="KAPPA_SIGF_XS", hint="BOC", comment="Kappa-Sigf Cross
Sections")
        self.defineio(unit=35, mode="i", generated="FIRE1", type="MATRIX_T", hint="BOC", comment="Matrix T Conponents")
        self.defineio(unit=36, mode="i", generated="FIRE1", type="MATRIX_R", hint="BOC", comment="Matrix R Conponents")

        # input/output file(s)
        self.defineio(unit=51,    mode="io",    omittable=False,    generated="CITATION,*",    target="SCGIVE,FIRE22",
type="DATA_BASE", comment="Data Base File")

        # output file(s)
        self.defineio(unit=42, mode="o", target="SAGEP", type="SOURCE", comment="Source Term used in Sagep")
        self.defineio(unit=52, mode="o", target="*",  type="SENS_ND_AND_POW", comment="Sensitivity Coefficients for 2nd
(N.D) and 5th (Pow) Terms")

        # user card and edit list
        self.defineio(unit=5, mode="i", type = "USER_CARD", comment="User Input Cards")
        self.defineio(unit=6, mode="o", type = "EDIT_LIST", comment="Editing List")


    def run(self,case=""):

        # fake and create unit 52, thenwrite "END" card
        file52 = DataFile()
        file52.create()
        file52.set_filetype("SENS_DIRECT")
        file52.write(["END¥n"])
        file52.close()
        self.attach(52, file52)


        # run sagep
        ExecutableFile.run(self, case)


class Scgivebr(Scgive):
    def __init__(self, case=""):
        ExecutableFile.__init__(self, sagep.config.scgive_br, case )
        self._defineio()
```

図 A2-10 ExecutableFile.py

```
import sagep.config
import sys
import cPickle

class SerializableAndVisible:

    def show_header(self):
        print "=====", self.__class__.__name__, " (BEGIN) ====="

    def show_footer(self):
        print "=====", self.__class__.__name__, " (END) ======="

    def show(self, verbose=0, header=1, footer=1):
        if header: self.show_header()
        keys = self.__dict__.keys()
        keys.sort()
        for key in keys:
            if type(self.__dict__[key]) == type([]):
                if len(self.__dict__[key]) > 10 :
                    print key, "=", self.__dict__[key][:5], " .... (omitted) .... ", self.__dict__[key][-5:]
                else:
                    print key, "=", self.__dict__[key]
            else:
                if type(self.__dict__[key]) == type("string"):
                    if len(self.__dict__[key]) > 1000:
                        #print key, "=", self.__dict__[key][:500], "¥n .... (omitted) .... ¥n", self.__dict__[key][-500:]
                        pass
                    else:
                        print key, "=", self.__dict__[key]
                else:
                    print key, type(self.__dict__[key]),  "=", self.__dict__[key]

        if footer: self.show_footer()

    def show2(self, verbose=0):
        self.show_header()
        keys = self.__dict__.keys()
        keys.sort()
        for key in keys:
            if hasattr(self.__dict__[key], "show" ):
                print key, "=", self.__dict__[key]
                print self.__dict__[key].show(header=0, footer=0)
            else:
                print key, "=", slef.__dict__[key]
        self.show_footer()

    def visualize(self):
        raise NotImplementedError


# global function

def flatten(seq):
    flat = []
    for item in seq:  # simple member
        if not (isinstance(item,list) or isinstance(item,tuple)):
            flat.append(item)
        else:     # sequence, so flatten this one too
            flat.extend(flatten(item))
    return flat  # or if you want: return tuple(flat)


def debug_print(level, *message):
    if level <= sagep.config.debug_level:
        for item in message:
            print item,
        print


if __name__ == "__main__":
    print flatten( [ [1,2,3], [4,5,6], 7 ] )
```

付録 3 新 SAGEP-BURN のためのテストクラスの詳細

図 A3-1 BR-adjoint-cycle4.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.MicroFile import *

from sagep.File.ModuleFile import *


class BR_Forward_Cycle1_Test(unittest.TestCase):

    def testRun(self):
        # initialization
        pdsedit = PDSEdit()
        sagep = Sagep()
        nsini4 = Nsini4()


        # Data file definitions
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )
        cit_micro_file = MicroFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICR018g" )


        # --- pdsedit (Cy04-EOC) ------------------------------------------------------------------
        card_text = [
            "16 1 0 0 2                                                                    ",
            " 0 18                                                                         ",
            " 0.0          0.0          0.0          0.0          0.0                      ",
            " 0.0          0.0          0.0          0.0          0.0                      ",
            " 0.0          0.0          0.0          0.0          0.0                      ",
            " 0.0          0.0          0.0          0.0          0.0                      ",
            "       0.01296      0.03111      0.13320      0.34764      1.43147",
            "       3.81048      0.0          0.0          0.0          0.0",
            "ICO1    SLAROM  ICO1  ",

            "ICO2    SLAROM  ICO2 ",
            "OTO1    SLAROM  OTO1 ",
            "OTO2    SLAROM  OTO2 ",
            "ABLU    SLAROM  ABLU ",
            "ABLL    SLAROM  ABLL ",
            "RBLU    SLAROM  RBLU ",
            "RBLM    SLAROM  RBLM ",
            "RBLL    SLAROM  RBLL ",
            "NAFO    SLAROM  NAFO ",
            "GPL1    SLAROM  GPL1 ",
            "GPL2    SLAROM  GPL2 ",
            "AXSU    SLAROM  AXSU ",
            "AXSL    SLAROM  AXSL  ",
            "RDS1    SLAROM  RDS1 ",
            "RDS2    SLAROM  RDS2 ",
            ]
        pdsedit_inp = CardFile( card_text )

        dummy_card = [
            "  33                                                                          ",
            "PU-239          949   1 239.05215     3.1800-11      1.0461-12      9.1581-13 ",
            "    0.0082E-2   0.0605E-2   0.0467E-2   0.0709E-2   0.0223E-2   0.0076E-2     ",
            "PU-240          940   2 240.05388     3.1543-11      8.3945-13      3.3404-12 ",
            "    0.0081E-2   0.0787E-2   0.0554E-2   0.1010E-2   0.0369E-2   0.0084E-2     ",
            "PU-241          941   3 241.05674     3.2088-11      1.0109-12      1.4950-09 ",
            "    0.0053E-2   0.1214E-2   0.0917E-2   0.2067E-2   0.0964E-2   0.0085E-2     ",
            "PU-242          942   4 242.05873     3.2040-11      8.3945-13      5.7994-14 ",
            "    0.0         0.0         0.0         0.0         0.0                       ",
            "U-235           925   5 235.04527     3.0903-11      1.0477-12      3.0957-17 ",
            "    0.0254E-2   0.1423E-2   0.1256E-2   0.2719E-2   0.0855E-2   0.0174E-2     ",
            "U-236           926   6 236.04607     3.0662-11      8.7469-13      9.1965-16 ",
            "    0.0         0.0         0.0         0.0         0.0                       ",
            "U-238           928   7 238.04951     3.1063-11      9.1154-13      4.8735-18 ",
            "    0.0206E-2   0.2169E-2   0.2564E-2   0.6142E-2   0.3562E-2   0.1187E-2     ",
            "FP-U235         854   8 235.0         0.0            1.2976-12      0.0       ",
            "    0.0         0.0         0.0         0.0         0.0                       ",
            "FP-PU239        894   9 239.0         0.0            1.2976-12      0.0       ",
            "    0.0         0.0         0.0         0.0         0.0                       ",
            "O-16              8  10  15.99950     0.0            5.4148-13      0.0       ",
            "    0.0         0.0         0.0         0.0         0.0                       ",
            "NA               11  11  22.98979     0.0            2.0025-12      0.0       ",
            "    0.0         0.0         0.0         0.0         0.0                       ",
```

```
    "CR                     24  12  51.99763      0.0             1.4883-12     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "NI                     28  13  58.72843      0.0             1.3665-12     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "MO                     42  14  95.88961      0.0             1.2928-12     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "FE                     26  15  55.84731      0.0             1.2496-12     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "MN                     25  16  54.94000      0.0             8.5841-13     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "B-10                  105  17  10.01610      0.0             4.4701-13     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "B-11                  115  18  11.00930      0.0             1.3827-12     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "C                       6  20  12.01100      0.0             9.7255-13     0.0       ",
    "   0.0          0.0             0.0          0.0        0.0             0.0          ",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  20  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  20  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  20  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  20  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  20  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  20  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0",
    "C                       6  19  12.01100      0.0             9.7255-13     0.0",
    "   0.0          0.0             0.0          0.0        0.0             0.0"
    ]
pdsedit_dummy_inp = CardFile( dummy_card )
pdsedit_dummy_inp.set_filetype( "DUMMY" )

nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc04-EOC" )
nd_file.set_generate_code( "REFUEL" )
nd_file.set_filetype( "NUMBER_DENSITY_REGION" )

macro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc04-EOC" )
macro_file.set_generate_code( "STARTUP" )
macro_file.set_filetype( "MACRO_XS" )

pdsedit.set_PDSin( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/mac18g" )  # given by make_Xsec/mk_micro.sh
pdsedit.inputs([pdsedit_inp, pdsedit_dummy_inp, macro_file, nd_file])
pdsedit.run( "cy04-EOC" )
list_file = pdsedit.list()

ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/PDSEDT-DATA.cyc04-EOC.dmp" )
assert list_file.has_same_contents_with( ref_list )


# --- sagep-direct ---------------------------------------------------------------------
card_text = [
    "FISSPEC ",
    "KEFF 04E",
    "SFACT   ",
    "PFACT04E",
    "BRFET04E",
    "BRFIS04E",
    "BRFI204E",
    "PSINI04E",
    "SGPFX04E",
    "H14 NA-MOX Breeding Ratio  cyc04 EOC DIRECT                                    ",
    "   0  1  0  0 18 16  0",
    "   1  0  0  0",
    "004",
    "   2  9.55043  3 15.71764  3 16.36130  3 16.46364  2  5.97324  4 18.64292",
    "   4 17.91088  3 12.78505  2  4.72722  5 21.25165  4 17.83599  4 18.10268",
```

```
"   2  4.62130  3 13.20161  2 10.12392  2  3.55860  4 18.97454  4 17.49924",
"   4 17.37424  5 32.60152  3 17.22164",
"   5 30.00000  6 30.00000  4 20.00000  3 15.00000  3 15.00000  3 15.00000",
"   4 20.00000  3 15.00000  3 15.00000  3 15.00000  4 20.00000 16 100.0000",
"   5 30.00000",
"005",
" 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
" 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
" 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
" 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
" 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
" 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
" 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
"   1",
"ICO1       SLAROM",
"   2",
"ICO2       SLAROM",
"   3",
"OTO1       SLAROM",
"   4",
"OTO2       SLAROM",
"   5",
"ABLU       SLAROM",
"   6",
"ABLL       SLAROM",
"   7",
"RBLU       SLAROM",
"   8",
"RBLM       SLAROM",
"   9",
"RBLL       SLAROM",
"  10",
"NAFO       SLAROM",
"  11",
"GPL1       SLAROM",
"  12",
"GPL2       SLAROM",
"  13",
"AXSU       SLAROM",
"  14",
"AXSL       SLAROM",
"  15",
"RDS1       SLAROM",
"  16",
"RDS2       SLAROM",
"H13 NA-MOX Breeding Ratio  cyc04 EOC DIRECT ",
"      4     0     0",
"   3",
" 928 0.00000E+00 0.35000E-10",
" 940 3.34600E-12 0.32440E-10",
" 948 0.00000E+00 0.31550E-10",
"   2",
" 949 9.11000E-13 0.32220E-10",
" 941 1.52640E-09 0.32480E-10",
"   9",
" 925 0.00000E+00 0.31270E-10",
" 926 0.00000E+00 0.33400E-10",
" 942 0.00000E+00 0.33010E-10",
" 951 5.08200E-11 0.35380E-10",
" 950 1.56000E-10 0.34490E-10",
" 953 0.00000E+00 0.34430E-10",
" 937 0.00000E+00 0.33360E-10",
" 939 3.40440E-06 0.33310E-10",
" 962 4.92820E-08 0.35100E-10",
" 3570.00E+06",
" 1.0000E-4 5.0000E-2 ",
"  1000   200",
"  2000"
]

self.footer_text = [
    "FISSION SP          8              -1.00",
    "FISSION SP          8              -0.90",
    "FISSION SP          8              -0.80",
    "FISSION SP          8              -0.70",
```

```
"FISSION SP          8              -0.60",
"FISSION SP          8              -0.50",
"FISSION SP          8              -0.40",
"FISSION SP          8              -0.30",
"FISSION SP          8              -0.20",
"FISSION SP          8              -0.10",
"FISSION SP          8              -0.05",
"FISSION SP          8               0.05",
"FISSION SP          8               0.10",
"FISSION SP          8               0.20",
"FISSION SP          8               0.30",
"FISSION SP          8               0.40",
"FISSION SP          8               0.50",
"FISSION SP          8               0.60",
"FISSION SP          8               0.70",
"FISSION SP          8               0.80",
"FISSION SP          8               0.90",
"FISSION SP          8               1.00",
" U-235        925    1",
" U-235        925    2",
" U-235        925    4",
" U-235        925    6",
" U-235        925    7",
" U-235        925   10",
" U-235        925    9",
" U-236        926    1",
" U-236        926    2",
" U-236        926    4",
" U-236        926    6",
" U-236        926    7",
" U-236        926   10",
" U-236        926    9",
" U-237        927    1",
" U-237        927    2",
" U-237        927    4",
" U-237        927    6",
" U-237        927    7",
" U-237        927   10",
" U-237        927    9",
" U-238        928    1",
" U-238        928    2",
" U-238        928    4",
" U-238        928    6",
" U-238        928    7",
" U-238        928   10",
" U-238        928    9",
"PU-238        948    1",
"PU-238        948    2",
"PU-238        948    4",
"PU-238        948    6",
"PU-238        948    7",
"PU-238        948   10",
"PU-238        948    9",
"PU-239        949    1",
"PU-239        949    2",
"PU-239        949    4",
"PU-239        949    6",
"PU-239        949    7",
"PU-239        949   10",
"PU-239        949    9",
"PU-240        940    1",
"PU-240        940    2",
"PU-240        940    4",
"PU-240        940    6",
"PU-240        940    7",
"PU-240        940   10",
"PU-240        940    9",
"PU-241        941    1",
"PU-241        941    2",
"PU-241        941    4",
"PU-241        941    6",
"PU-241        941    7",
"PU-241        941   10",
"PU-241        941    9",
"PU-242        942    1",
"PU-242        942    2",
"PU-242        942    4",
"PU-242        942    6",
"PU-242        942    7",
"PU-242        942   10",
"PU-242        942    9",
"AM-241        951    1",
```

```
"AM-241      951    2",
"AM-241      951    4",
"AM-241      951    6",
"AM-241      951    7",
"AM-241      951   10",
"AM-241      951    9",
"AM-242M     950    1",
"AM-242M     950    2",
"AM-242M     950    4",
"AM-242M     950    6",
"AM-242M     950    7",
"AM-242M     950   10",
"AM-242M     950    9",
"AM-243      953    1",
"AM-243      953    2",
"AM-243      953    4",
"AM-243      953    6",
"AM-243      953    7",
"AM-243      953   10",
"AM-243      953    9",
"NP-237      937    1                                   00800007",
"NP-237      937    2                                   00810007",
"NP-237      937    4                                   00820007",
"NP-237      937    6                                   00830007",
"NP-237      937    7                                   00820007",
"NP-237      937    9                                   00830007",
"NP-237      937   10                                   00830007",
"NP-239      939    1                                   00800007",
"NP-239      939    2                                   00810007",
"NP-239      939    4                                   00820007",
"NP-239      939    6                                   00830007",
"NP-239      939    7                                   00820007",
"NP-239      939    9                                   00830007",
"NP-239      939   10 ",
"CM-242      962    1                                   00720007",
"CM-242      962    2                                   00730007",
"CM-242      962    4                                   00740007",
"CM-242      962    6                                   00750007",
"CM-242      962    7                                   00730007",
"CM-242      962    9                                   00750007",
"CM-242      962   10                                   00740007",
"CM-243      963    1                                   00720007",
"CM-243      963    2                                   00730007",
"CM-243      963    4                                   00740007",
"CM-243      963    6                                   00750007",
"CM-243      963    7                                   00730007",
"CM-243      963    9                                   00750007",
"CM-243      963   10                                   00740007",
"CM-244      964    1                                   00720007",
"CM-244      964    2                                   00730007",
"CM-244      964    4                                   00740007",
"CM-244      964    6                                   00750007",
"CM-244      964    7                                   00730007",
"CM-244      964    9                                   00750007",
"CM-244      964   10    ",
"O             8    1",
"O             8    6",
"O             8    7",
"O             8   10",
"O             8    9",
"NA           11    1",
"NA           11    6",
"NA           11    7",
"NA           11   10",
"NA           11    9",
"CR           24    1",
"CR           24    6",
"CR           24    7",
"CR           24   10",
"CR           24    9",
"MN-55        25    1                                   00860007",
"MN-55        25    6                                   00860007",
"MN-55        25    7                                   00860007",
"MN-55        25   10                                   00860007",
"MN-55        25    9    ",
"FE           26    1",
"FE           26    6",
"FE           26    7",
"FE           26   10",
"FE           26    9",
"NI           28    1",
"NI           28    6",
```

```
    "NI              28     7",
    "NI              28    10",
    "NI              28     9",
    "ZR              40     1",
    "ZR              40     6",
    "ZR              40     7",
    "ZR              40    10",
    "ZR              40     9",
    "MO-NAT          42     1                                        00860007",
    "MO-NAT          42     6                                        00860007",
    "MO-NAT          42     7                                        00860007",
    "MO-NAT          42    10                                        00860007",
    "MO-NAT          42     9   ",
    "ND-143         603     1",
    "ND-143         603     6",
    "ND-143         603     7",
    "ND-143         603    10",
    "ND-143         603     9",
    "W               74     1                                        00860007",
    "W               74     6                                        00860007",
    "W               74     7                                        00860007",
    "W               74    10                                        00860007",
    "W               74     9",
    "PB              82     1",
    "PB              82     6",
    "PB              82     7",
    "PB              82    10",
    "PB              82     9",
    " B- 10         105     1",
    " B- 10         105     6",
    " B- 10         105     7",
    " B- 10         105    10",
    " B- 10         105     9",
    " B- 11         115     1",
    " B- 11         115     6",
    " B- 11         115     7",
    " B- 11         115    10",
    " B- 11         115     9",
    "C                6     1",
    "C                6     6",
    "C                6     7",
    "C                6    10",
    "C                6     9",
    "PU-241FP       814     1",
    "PU-241FP       814     6",
    "PU-241FP       814     7",
    "PU-241FP       814    10",
    "PU-241FP       814     9",
    "U-235FP        854     1",
    "U-235FP        854     6",
    "U-235FP        854     7",
    "U-235FP        854    10",
    "U-235FP        854     9",
    "U-238FP        884     1",
    "U-238FP        884     6",
    "U-238FP        884     7",
    "U-238FP        884    10",
    "U-238FP        884     9",
    "PU-239FP       894     1",
    "PU-239FP       894     6",
    "PU-239FP       894     7",
    "PU-239FP       894    10",
    "PU-239FP       894     9"
    ]

inp = CardFile( card_text + self.footer_text )

flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-EOC" )
flux_forward.set_generate_code( "CITATION" )
flux_forward.set_filetype( "MESH_FLUX" )
flux_forward.set_casename( "Cy04-EOC_FORWARD" )
flux_forward.setProperty( "NX", 68 )
flux_forward.setProperty( "NY", 62 )
flux_forward.setProperty( "NZ", 1 )
flux_forward.setProperty( "NG", 18 )

flux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-EOC-adj" )
flux_adjoint.set_generate_code( "CITATION" )
flux_adjoint.set_filetype( "MESH_FLUX" )
flux_adjoint.set_casename( "Cy04-EOC_ADJOINT" )
flux_adjoint.setProperty( "NX", 68 )
flux_adjoint.setProperty( "NY", 62 )
```

```
flux_adjoint.setProperty( "NZ", 1 )
flux_adjoint.setProperty( "NG", 18 )

nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc04-EOC" )
nd_file.set_generate_code( "REFUEL" )
nd_file.set_filetype( "NUMBER_DENSITY_REGION" )

sagep.inputs([inp,  nd_file, database, flux_forward, flux_adjoint])
sagep.run( "cy04-EOC" )
list_file = sagep.list()
sagep_direct_outfiles = sagep.outfiles()

ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SGP-cyc04-EOC-direct.dmp" )
assert list_file.has_same_contents_with( ref_list )


# --- nsini4 ------------------------------------------------------------------------------------
nsini4 = Nsini4()

card_text = [
    "KEFF 04E",
    "BRFET04E",
    "BRFIS04E",
    "PSINI04E",
    "PFACT04E",
    "FISSPEC",
    " 21",
    " 925 926 937 928 939 948 949",
    " 940 941 942 951 950 953 962",
    " 963 964 965 854 884 894 814",
    "  0  1  0",
    "         68         62          1         18         21         13          1",
    "      2     3     3     3     2     4     4     3     2     5     4     4",
    "      2     3     2     2     4     4     4     5     3",
    "  9.5504 15.7176 16.3613 16.4636  5.9732 18.6429 17.9109 12.7851",
    "  4.7272 21.2517 17.8360 18.1027  4.6213 13.2016 10.1232  3.5586",
    " 18.9745 17.4992 17.3742 32.6015 17.2216",
    "      5     6     4     3     3     3     4     3     3     3     4    16",
    "      5",
    " 30.0000 30.0000 20.0000 15.0000 15.0000 15.0000 20.0000 15.0000",
    " 15.0000 15.0000 20.0000 100.000 30.0000",
    "      1",
    "    1.00",
    " 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
    " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
    " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
    " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
    " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
    " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
    " 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
    "         17        17",
    " 925 0.31270E-10",
    " 926 0.33400E-10",
    " 937 0.33360E-10",
    " 928 0.35000E-10",
    " 939 0.33310E-10 3.4044E-06",
    " 948 0.31550E-10",
    " 949 0.32220E-10 9.1100E-13",
    " 940 0.32440E-10 3.3460E-12",
    " 941 0.32480E-10 1.5264E-09 ",
    " 942 0.33010E-10 ",
    " 951 0.35380E-10 5.0820E-11",
    " 950 0.34490E-10 1.5600E-10",
    " 953 0.34430E-10",
    " 962 0.35100E-10 4.9282E-08",
    " 963 0.35160E-10 7.5500E-10",
    " 964 0.34010E-10 1.2130E-09",
    " 965 0.35290E-10",
    " 854 0.0",
    " 884 0.0",
    " 894 0.0",
    " 814 0.0"
    ]
inp = CardFile( card_text )

# unit 2
```

```
#gflux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/sagep93.FT12/SGP-cyc04-EOC-direct" )
#gflux_adjoint.set_generate_code( "SAGEP" )
#gflux_adjoint.set_filetype( "GENERALIZED_FLUX_ADJOINT" )

# unit 3
flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-EOC" )
flux_forward.set_generate_code( "CITATION" )
flux_forward.set_filetype( "MESH_FLUX" )
flux_forward.set_casename( "Cy04-EOC_FORWARD" )
flux_forward.setProperty( "NX", 68 )
flux_forward.setProperty( "NY", 62 )
flux_forward.setProperty( "NZ", 1 )
flux_forward.setProperty( "NG", 18 )

# unit 10
macro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc04-EOC" )
macro_file.set_generate_code( "STARTUP" )
macro_file.set_filetype( "MACRO_XS" )

# unit 12
micro_file = SagepMicroFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )

# unit 16
leakage_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT20/cyc04-EOC" )
leakage_file.set_generate_code( "CITATION" )
leakage_file.set_filetype( "LEAKAGE_TERM" )

nsini4.inputs([inp, sagep_direct_outfiles, flux_forward, macro_file, micro_file, leakage_file, database])
nsini4.run("Cy04-EOC")
list_file = nsini4.list()

ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/NSINI4-cyc04EOC.dmp" )
assert list_file.has_same_contents_with( ref_list )


nsini4_outfiles = nsini4.outfiles()
assert len(nsini4_outfiles) == 1 , len(nsini4_outfiles)   # units: 20

# unit 20 (text)
assert nsini4_outfiles[0].generate_code() == "NSINI4", nsini4_outfiles[0].generate_code()
assert nsini4_outfiles[0].filetype() == "NUMBER_DENSITY_ADJOINT_FINAL", nsini4_outfiles[0].filetype()
assert nsini4_outfiles[0].has_same_contents_with( RefFile( ref1_dir +
"/H13-Na-MOX.C3/BR/WORK/nsini4.FT20/cyc04-EOC" ) )


# --- refuel2 (EOC) ------------------------------------------------------------------------
refuel2 = Refuel2()

card_text = [
    "    1    0",
    "   16    0",
    "    1    1    1    1    1    1    1    1    1    1    1    1    1    1           14",
    "    1    1    ",
    "   21",
    "  925  926  937  928  939  948  949  940  941  942  951  950  953  962",
    "  963  964  965  854  884  894  814",
    "    1  4.0                                                                      ",
    "    2  4.0                                                                      ",
    "    3  4.0                                                                      ",
    "    4  4.0                                                                      ",
    "    5  4.0                                                                      ",
    "    6  4.0                                                                      ",
    "    7  4.0                                                                      ",
    "    8  4.0                                                                      ",
    "    9  4.0                                                                      ",
    "   10  1.0                                                                      ",
    "   11  1.0                                                                      ",
    "   12  1.0                                                                      ",
    "   13  1.0                                                                      ",
    "   14  1.0                                                                      ",
    "   15  1.0                                                                      ",
    "   16  1.0                                                                      ",
    "    1    1  0.0",
    "    2    1  0.0",
    "    3    1  0.0",
    "    4    1  0.0",
    "    5    1  0.0",
    "    6    1  0.0",
    "    7    1  0.0",
    "    8    1  0.0",
    "    9    1  0.0",
    "   10    1  0.0",
```

```
    "   11    1  0.0",
    "   12    1  0.0",
    "   13    1  0.0",
    "   14    1  0.0",
    "   15    1  0.0",
    "   16    1  0.0"
    ]
inp = CardFile( card_text )

# unit 21
#nd_init = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/nsini4.FT20/cyc04-EOC" )
#nd_init.set_generate_code( "NSINI4" )
#nd_init.set_filetype( "NUMBER_DENSITY_ADJOINT_FINAL" )

refuel2.inputs( [inp, nsini4_outfiles] )
refuel2.run( "cyc04-EOC" )
list_file = refuel2.list()

refuel2_outfiles = refuel2.outfiles()   # unit 11, 12
assert len(refuel2_outfiles) == 2

# unit 11
assert refuel2_outfiles[0].filetype() == "REFUEL2_CONTROL_DATA"
assert refuel2_outfiles[0].generate_code() == "REFUEL2"
ref_unit_11 = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT11/cyc04-EOC" )
assert refuel2_outfiles[0].has_same_contents_with( ref_unit_11 )

# unit 12
assert refuel2_outfiles[1].filetype() == "NUMBER_DENSITY_REGION_ADJOINT"
assert refuel2_outfiles[1].generate_code() == "REFUEL2"
ref_unit_12 = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT12/cyc04-EOC" )
assert refuel2_outfiles[1].has_same_contents_with( ref_unit_12 )


# --- fire1 ------------------------------------------------------------------------------
fire1 = Fire1()

card_text = [
    "AVFX 04E",
    "VOL  04E",
    "PWFTa04E",
    "PWFXa04E",
    "16,21,18,10,1,17,1,1",
    "3570.00E+06,0.0,0.0",
    "47347200,20",
    "  5, 3.4044E-06,  1,  7, 1.0",
    "  7, 9.1100E-13,  1,  1, 1.0",
    "  8, 3.3460E-12,  1,  2, 1.0",
    "  9, 1.5264E-09,  1, 11, 1.0",
    " 11, 5.0820E-11,  1,  3, 1.0",
    " 12, 1.5600E-10,  1, 10, 0.173",
    " 12, 1.5600E-10,  1, 14, 0.827",
    " 14, 4.9282E-08,  1,  6, 1.0",
    " 15, 7.5500E-10,  1,  7, 1.0",
    " 16, 1.2130E-09,  1,  8, 1.0",
    "  1,  2, 1.0",
    "  2,  3, 1.0",
    "  3,  6, 1.0",
    "  4,  5, 1.0",
    "  6,  7, 1.0",
    "  7,  8, 1.0",
    "  8,  9, 1.0",
    "  9, 10, 1.0",
    " 10, 13, 1.0",
    " 11, 10, 0.1384",
    " 11, 12, 0.200",
    " 11, 14, 0.6616",
    " 12, 13, 1.0",
    " 13, 16, 1.0",
    " 14, 15, 1.0",
    " 15, 16, 1.0",
    " 16, 17, 1.0",
    "0,0,0.0",
    "  2,  1, 1.0",
    "  4,  3, 1.0",
    "  6,  3, 1.0",
    "  7,  6, 1.0",
    "  8,  7, 1.0",
    "  9,  8, 1.0",
    " 10,  9, 1.0",
    " 11,  8, 1.0",
    " 12, 11, 1.0",
```

```
            " 14, 11, 1.0",
            " 15, 14, 1.0",
            " 16, 15, 1.0",
            " 17, 16, 1.0",
            "0,0,0.0",
            "   U 235 925   U 236 926  NP-237 937   U 238 928  NP 239 939  PU 248 948",
            "  PU 239 949  PU 240 940  PU 241 941  PU 242 942  AM 241 951  AM 942 950",
            "  AM 243 953  CM 242 962  CM 243 963  CM 244 964  CM 245 965  FP 854 854",
            "  FP 884 884  FP 894 894  FP 814 814",
            "0,0",
            "   1, 0.312700E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
            "   2, 0.334000E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
            "   3, 0.333600E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
            "   4, 0.350000E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
            "   5, 0.333100E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            "   6, 0.315500E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "   7, 0.322200E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "   8, 0.324400E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "   9, 0.324800E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  10, 0.330100E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  11, 0.353800E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  12, 0.344900E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  13, 0.344300E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  14, 0.351000E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  15, 0.351600E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  16, 0.340100E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0",
            "  17, 0.352900E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0"
            ]

        inp = CardFile( card_text )


        # Unit 11
        nd_file_boc = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc04-BOC" )
        nd_file_boc.set_generate_code( "REFUEL" )
        nd_file_boc.set_filetype( "NUMBER_DENSITY_REGION" )
        nd_file_boc.set_casename( "Cy04-BOC" )


        fire1.inputs([inp, micro_file, nd_file_boc, refuel2_outfiles, database])
        fire1.run("Cy04-EOC")
        list_file = fire1.list()

        ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/FIRE1_cyc04-adj.dmp" )
        assert list_file.has_same_contents_with( ref_list )


        fire1_outfiles = fire1.outfiles()
        assert len(fire1_outfiles) == 6 , len(fire1_outfiles)   # units: 12, 30, 31, 34, 35, 36

        # unit 12 (text)
        assert fire1_outfiles[0].generate_code() == "FIRE1", fire1_outfiles[0].generate_code()
        assert fire1_outfiles[0].filetype() == "NUMBER_DENSITY_ADJOINT_FINAL", fire1_outfiles[0].filetype()
        assert           fire1_outfiles[0].has_same_contents_with(           RefFile(           ref1_dir           +
"/H13-Na-MOX.C3/BR/WORK/fire1.FT12/cyc04-EOC" ) )

        # unit 31 (text)
        assert fire1_outfiles[2].generate_code() == "FIRE1", fire1_outfiles[0].generate_code()
        assert fire1_outfiles[2].filetype() == "NUMBER_DENSITY_ADJOINT_EACH", fire1_outfiles[0].filetype()
        assert           fire1_outfiles[2].has_same_contents_with(           RefFile(           ref1_dir           +
"/H13-Na-MOX.C3/BR/WORK/fire1.FT31/cyc04-EOC" ) )


        # --- scgive ------------------------------------------------------------------------
```

```
scgivebr = Scgivebr()  #  for BR calculation, a special version of SCGIVE is used.
card_text = [
    "PWFTa04E",
    "PWFXa04E",
    "VOL   04B",
    "SFACT04B ",
    "PS    04B",
    "DNORM04B",
    "SCG   04B",
    "0,0,0,0,1.0,1.0",
    "16,21,18",
    "1",
    "47347200,20,3570.00E+06",
    "1",
    "21,1",
    "925,1",
    "1,1,1.0,  1,2,1.0,  1,18,1.0,  0,0,1.0",
    "926,1",
    "2,2,1.0,  2,3,1.0,  2,18,1.0,  2,1,1.0",
    "937,1",
    "3,3,1.0,  3,6,1.0,  3,19,1.0,  0,0,1.0",
    "928,1",
    "4,4,1.0,  4,5,1.0,  4,19,1.0,  4,3,1.0",
    "939,1",
    "5,5,1.0,  0,0,1.0,  5,20,1.0,  0,0,1.0",
    "948,1",
    "6,6,1.0,  6,7,1.0,  6,19,1.0,  6,3,1.0",
    "949,1",
    "7,7,1.0,  7,8,1.0,  7,20,1.0,  7,6,1.0",
    "940,1",
    "8,8,1.0,  8,9,1.0,  8,21,1.0,  8,7,1.0",
    "941,1",
    "9,9,1.0,  9,10,1.0,  9,21,1.0,  9,8,1.0",
    "942,1",
    "10,10,1.0,  10,13,1.0,  10,21,1.0,  10,9,1.0",
    "951,3",
    "11,11,1.0,  11,10,0.1384,  11,21,1.0,  11,  8,1.0",
    "  0,  0,1.0,  11,12,0.2000,   0,  0,1.0,   0,  0,1.0",
    "  0,  0,1.0,  11,14,0.6616,   0,  0,1.0,   0,  0,1.0",
    "950,1",
    "12,12,1.0,  12,13,1.0,  12,21,1.0,  12,11,1.0",
    "953,1",
    "13,13,1.0,  13,16,1.0,  13,21,1.0,   0,  0,1.0",
    "962,1",
    "14,14,1.0,  14,15,1.0,  14,21,1.0,  14,11,1.0",
    "963,1",
    "15,15,1.0,  15,16,1.0,  15,21,1.0,  15,14,1.0",
    "964,1",
    "16,16,1.0,  16,17,1.0,  16,21,1.0,  16,15,1.0",
    "964,1",
    "17,17,1.0,   0,  0,1.0,  17,21,1.0,  17,16,1.0",
    "854,1",
    "18,18,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
    "884,1",
    "19,19,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
    "894,1",
    "20,20,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
    "814,1",
    "21,21,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
    "21",
    "925,  0.312700E-10,  1",
    "926,  0.334000E-10,  2",
    "937,  0.333600E-10,  3",
    "928,  0.350000E-10,  4",
    "939,  0.333100E-10,  5",
    "948,  0.315500E-10,  6",
    "949,  0.322200E-10,  7",
    "940,  0.324400E-10,  8",
    "941,  0.324800E-10,  9",
    "942,  0.330100E-10,10",
    "951,  0.353800E-10,11",
    "950,  0.344900E-10,12",
    "953,  0.344300E-10,13",
    "962,  0.351000E-10,14",
    "963,  0.351600E-10,15",
    "964,  0.340100E-10,16",
    "965,  0.352900E-10,17",
    "854,  0.0          ,18",
    "884,  0.0          ,19",
    "894,  0.0          ,20",
    "814,  0.0          ,21",
    ]
```

```
inp = CardFile( card_text )

ref_inp = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SCG-cyc04" )
assert inp.has_same_contents_with( ref_inp )


# unit 30
nd_forward_boc = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT30/cyc04-BOC" )
nd_forward_boc.set_generate_code( "FIRE1" )
nd_forward_boc.set_filetype( "NUMBER_DENSITY_EACH" )
nd_forward_boc.set_casename( "Cy04-BOC" )

# unit 34
sigfk = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT34/cyc04-BOC" )
sigfk.set_generate_code( "FIRE1" )
sigfk.set_filetype( "KAPPA_SIGF_XS" )
sigfk.set_casename( "Cy04-BOC" )

# unit 35
mat_t = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT35/cyc04-BOC" )
mat_t.set_generate_code( "FIRE1" )
mat_t.set_filetype( "MATRIX_T" )
mat_t.set_casename( "Cy04-BOC" )

# unit 36  --> fire1_outfiles
mat_r = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT36/cyc04-BOC" )
mat_r.set_generate_code( "FIRE1" )
mat_r.set_filetype( "MATRIX_R" )
mat_r.set_casename( "Cy04-BOC" )

scgivebr.inputs( [inp, micro_file, fire1_outfiles, nd_forward_boc, sigfk, mat_t, mat_r, database] )
scgivebr.run("Cy04-EOC")
list_file = scgivebr.list()

scgive_outfiles = scgivebr.outfiles()   # unit 42, 52
assert len(scgive_outfiles) == 2

# unit 42
assert scgive_outfiles[0].filetype() == "SOURCE"
assert scgive_outfiles[0].generate_code() == "SCGIVEBR"
ref_unit_42 = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/scgive.FT42/SCG-cyc04" )
assert scgive_outfiles[0].has_same_contents_with( ref_unit_42 ), scgive_outfiles[0].diff_from( ref_unit_42 )

# unit 52
assert scgive_outfiles[1].filetype() == "SENS_ND_AND_POW"
assert scgive_outfiles[1].generate_code() == "SCGIVEBR"
ref_unit_52 = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/scgive.FT52/SCG-cyc04" )
assert scgive_outfiles[1].has_same_contents_with( ref_unit_52 )



# --- pdsedit (Cy04-BOC) ------------------------------------------------------------------------
nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc04-BOC" )
nd_file.set_generate_code( "REFUEL" )
nd_file.set_filetype( "NUMBER_DENSITY_REGION" )

macro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc04-BOC" )
macro_file.set_generate_code( "STARTUP" )
macro_file.set_filetype( "MACRO_XS" )

pdsedit.set_PDSin( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/mac18g" )  # given by make_Xsec/mk_micro.sh
pdsedit.inputs([pdsedit_inp, pdsedit_dummy_inp, macro_file, nd_file])
pdsedit.run( "cy04-BOC" )
list_file = pdsedit.list()

ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/PDSEDT-DATA.cyc04-BOC.dmp" )
assert list_file.has_same_contents_with( ref_list )


# --- SAGEP-flux ---------------------------------------------------------------------------------

card_text = [
    "FISSPEC ",
    "KEFF 04B",
    "SFACT04B ",
    "PFACT04B",
    "BRFET04B",
    "BRFISO4B",
    "BRFI204B",
    "PSINI04B",
    "SGPFX04B",
    "H14 NA-MOX Breeding Ratio  cyc04 flux                                    ",
```

```
"  0  1  0  0 18 16  0",
"  1  0  0  0",
"004",
"  2  9.55043  3 15.71764  3 16.36130  3 16.46364  2  5.97324  4 18.64292",
"  4 17.91088  3 12.78505  2  4.72722  5 21.25165  4 17.83599  4 18.10268",
"  2  4.62130  3 13.20161  2 10.12392  2  3.55860  4 18.97454  4 17.49924",
"  4 17.37424  5 32.60152  3 17.22164",
"  5 30.00000  6 30.00000  4 20.00000  3 15.00000  3 15.00000  3 15.00000",
"  4 20.00000  3 15.00000  3 15.00000  3 15.00000  4 20.00000 16 100.0000",
"  5 30.00000",
"005",
" 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
" 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
" 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
" 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
" 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
" 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
" 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
" 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
"  1",
"ICO1      SLAROM",
"  2",
"ICO2      SLAROM",
"  3",
"OTO1      SLAROM",
"  4",
"OTO2      SLAROM",
"  5",
"ABLU      SLAROM",
"  6",
"ABLL      SLAROM",
"  7",
"RBLU      SLAROM",
"  8",
"RBLM      SLAROM",
"  9",
"RBLL      SLAROM",
" 10",
"NAFO      SLAROM",
" 11",
"GPL1      SLAROM",
" 12",
"GPL2      SLAROM",
" 13",
"AXSU      SLAROM",
" 14",
"AXSL      SLAROM",
" 15",
"RDS1      SLAROM",
" 16",
"RDS2      SLAROM",
"H13 NA-MOX Breeding Ratio  cyc04 EOC DIRECT ",
"      3     0     0",
"   1.0E-04   5.0E-02",
" 99999 99999",
"  2000"
]

inp = CardFile( card_text + self.footer_text )

flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-BOC" )
flux_forward.set_generate_code( "CITATION" )
flux_forward.set_filetype( "MESH_FLUX" )
flux_forward.set_casename( "Cy04-BOC_FORWARD" )
flux_forward.setProperty( "NX", 68 )
flux_forward.setProperty( "NY", 62 )
flux_forward.setProperty( "NZ", 1 )
flux_forward.setProperty( "NG", 18 )

flux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-BOC-adj" )
flux_adjoint.set_generate_code( "CITATION" )
flux_adjoint.set_filetype( "MESH_FLUX" )
flux_adjoint.set_casename( "Cy04-BOC_ADJOINT" )
flux_adjoint.setProperty( "NX", 68 )
flux_adjoint.setProperty( "NY", 62 )
flux_adjoint.setProperty( "NZ", 1 )
flux_adjoint.setProperty( "NG", 18 )
```

```
#source = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/scgive.FT42/SCG-cyc04" )
#source.set_generate_code( "SCGIVE" )
#source.set_filetype( "SOURCE" )

database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

# sagep.set_PDSin( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/mic18g-cyc04-BOC" )
sagep.inputs([inp,  scgive_outfiles, database, flux_forward, flux_adjoint])
sagep.run( "cy04-BOC" )
sagep_flux_outfiles = sagep.outfiles()
list_file = sagep.list()

ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SGP-cyc04-flx.dmp" )
assert list_file.has_same_contents_with( ref_list )


# --- REFUEL2 ------------------------------------------------------------------------------
card_text = [
    "     1    1"
    ]
inp = CardFile( card_text )

# unit 21
#nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT12/cyc04-EOC" )
#nd_file.set_generate_code( "FIRE1" )
#nd_file.set_filetype( "NUMBER_DENSITY_ADJOINT_FINAL" )

# unit 10
#ctrl_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT11/cyc04-EOC" )
#ctrl_file.set_generate_code( "REFUEL2" )
#ctrl_file.set_filetype( "REFUEL2_CONTROL_DATA" )


refuel2.inputs( [inp, fire1_outfiles, refuel2_outfiles] )
refuel2.run( "cyc04-BOC" )
list_file = refuel2.list()

refuel2_outfiles = refuel2.outfiles()   # unit 11, 12
assert len(refuel2_outfiles) == 2

# unit 11
assert refuel2_outfiles[0].filetype() == "REFUEL2_CONTROL_DATA"
assert refuel2_outfiles[0].generate_code() == "REFUEL2"
ref_unit_11 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT11/cyc04-BOC" )
assert refuel2_outfiles[0].has_same_contents_with( ref_unit_11 )

# unit 12
assert refuel2_outfiles[1].filetype() == "NUMBER_DENSITY_REGION_ADJOINT"
assert refuel2_outfiles[1].generate_code() == "REFUEL2"
ref_unit_12 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT12/cyc04-BOC" )
assert refuel2_outfiles[1].has_same_contents_with( ref_unit_12 )


# --- NSJUMP -------------------------------------------------------------------------------
nsjump = Nsjump()
card_text = [
    "KEFF 04B",
    "PWFTa04E",
    "PS   04B",
    "DNORM04B",
    "FISSPEC",
    " 1.00000E+24",
    " 21",
    " 925 926 937 928 939 948 949",
    " 940 941 942 951 950 953 962",
    " 963 964 965 854 884 894 814",
    "  0  1  0",
    "        68        62         1        18        21        13         1",
    "     2     3     3     3     2     4     4     3     2     5     4     4",
    "     2     3     2     2     4     4     4     5     3",
    "  9.5504 15.7176 16.3613 16.4636  5.9732 18.6429 17.9109 12.7851",
    "  4.7272 21.2517 17.8360 18.1027  4.6213 13.2016 10.1232  3.5586",
    " 18.9745 17.4992 17.3742 32.6015 17.2216",
    "     5     6     4     3     3     3     4     3     3     3     4    16",
    "     5",
    " 30.0000 30.0000 20.0000 15.0000 15.0000 15.0000 20.0000 15.0000",
    " 15.0000 15.0000 20.0000 100.000 30.0000",
    "     1",
    "    1.00",
    " 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
    " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
```

```
        " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
        " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
        " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
        " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
        " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
        " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
        " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
        " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
        " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
        " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
        " 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
        "         17          17",
        " 925 0.31270E-10",
        " 926 0.33400E-10",
        " 937 0.33360E-10",
        " 928 0.35000E-10",
        " 939 0.33310E-10 ",
        " 948 0.31550E-10",
        " 949 0.32220E-10 ",
        " 940 0.32440E-10",
        " 941 0.32480E-10",
        " 942 0.33010E-10",
        " 951 0.35380E-10 ",
        " 950 0.34490E-10 ",
        " 953 0.34430E-10",
        " 962 0.35100E-10 ",
        " 963 0.35160E-10 ",
        " 964 0.34010E-10 ",
        " 965 0.35290E-10",
        " 854 0.0",
        " 884 0.0",
        " 894 0.0",
        " 814 0.0",
        "3570.00E+06"
        ]
    inp = CardFile( card_text )

    # unit 2
    #gflux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/sagep93.FT12/SGP-cyc04-flx" )
    #gflux_adjoint.set_generate_code( "SAGEP" )
    #gflux_adjoint.set_filetype( "GENERALIZED_FLUX_ADJOINT" )

    # unit 3
    flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-BOC" )
    flux_forward.set_generate_code( "CITATION" )
    flux_forward.set_filetype( "MESH_FLUX" )
    flux_forward.set_casename( "Cy04-EOC_FORWARD" )
    flux_forward.setProperty( "NX", 68 )
    flux_forward.setProperty( "NY", 62 )
    flux_forward.setProperty( "NZ", 1 )
    flux_forward.setProperty( "NG", 18 )

    # unit 10
    macro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc04-BOC" )
    macro_file.set_generate_code( "STARTUP" )
    macro_file.set_filetype( "MACRO_XS" )

    # unit 11
    #nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT12/cyc04-BOC" )
    #nd_file.set_generate_code( "REFUEL2" )
    #nd_file.set_filetype( "NUMBER_DENSITY_REGION_ADJOINT" )

    # unit 12
    #micro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
    #micro_file.set_generate_code( "CONVMIC" )
    #micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

    # unit 16
    leakage_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT20/cyc04-BOC" )
    leakage_file.set_generate_code( "CITATION" )
    leakage_file.set_filetype( "LEAKAGE_TERM" )

    # Unit 51
    #database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )


    nsjump.inputs([inp, sagep_flux_outfiles, flux_forward, macro_file, refuel2_outfiles, micro_file, leakage_file,
database])
    nsjump.run()
    list_file = nsjump.list()

    ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/NSJUMP-cyc04.dmp" )
```

```
        assert list_file.has_same_contents_with( ref_list )


        nsjump_outfiles = nsjump.outfiles()
        assert len(nsjump_outfiles) == 1 , len(nsjump_outfiles)   # units: 20

        # unit 20 (text)
        assert nsjump_outfiles[0].generate_code() == "NSJUMP", nsjump_outfiles[0].generate_code()
        assert nsjump_outfiles[0].filetype() == "NUMBER_DENSITY_ADJOINT_FINAL", nsjump_outfiles[0].filetype()
        assert          nsjump_outfiles[0].has_same_contents_with(          DataFile(          ref1_dir          +
"/H13-Na-MOX.C3/BR/WORK/nsjump.FT20/cyc04-BOC" ) )


def suite():
    return unittest.makeSuite( BR_Forward_Cycle1_Test, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-2 BR-forward-cycle1.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.MicroFile import *

from sagep.File.ModuleFile import *


class BR_Forward_Cycle1_Test(unittest.TestCase):

    def testRun(self):
        # initialization
        convmic = Convmic()
        refuel = Refuel()
        startup = Startup()
        citation = Citation()
        fire1 = Fire1()
        fire2 = Fire2()

        # Data file definitions
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

        cit_micro_file = MicroFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICRO18g" )

        # convmic
        convmic_card = [
            "    33",
            "  925  926  937  928  939  948  949  940  941  942  951  950  953  962",
            "  963  964  965  854  884  894  814    8   11   24   25   26   28   42",
            "  603   74  105  115    6"
            ]
        convmic_inp = CardFile( convmic_card )
        convmic.inputs( [convmic_inp, cit_micro_file] )
        convmic.run( "cyc01-BOC")
        micro_file = convmic.outfiles()

        list_file = convmic.list()
        assert list_file.has_same_contents_with( DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CONVMIC.dmp" ) )


        # refuel
        refuel_card = [
            "    1    0                                                                  ",
            "   16    0                                                                  ",
            "    4    4    4    4    4    4    4    4    4    1    1    1    1    1      14",
            "    1    1    ",
            "   33",
            "  925  926  937  928  939  948  949  940  941  942  951  950  953  962",
            "  963  964  965  854  884  894  814    8   11   24   25   26   28   42",
            "  603   74  105  115    6",
            "    1  1.0  1.0  1.0  1.0                                                  ",
            "    2  1.0  1.0  1.0  1.0                                                  ",
            "    3  1.0  1.0  1.0  1.0                                                  ",
            "    4  1.0  1.0  1.0  1.0                                                  ",
            "    5  1.0  1.0  1.0  1.0                                                  ",
            "    6  1.0  1.0  1.0  1.0                                                  ",
            "    7  1.0  1.0  1.0  1.0                                                  ",
            "    8  1.0  1.0  1.0  1.0                                                  ",
            "    9  1.0  1.0  1.0  1.0                                                  ",
            "   10  1.0  ",
            "   11  1.0  ",
            "   12  1.0  ",
            "   13  1.0  ",
            "   14  1.0  ",
            "   15  1.0  ",
            "   16  1.0  ",
            "   16                                                                      ",
            "    1",
            "  925 1.99109E-05   926 0.00000E+00   937 9.45050E-06   928 6.53346E-03",
            "  939 0.00000E+00   948 2.06825E-05   949 1.01294E-03   940 5.98514E-04",
            "  941 7.98413E-05   942 7.21147E-05   951 3.71706E-05   950 0.00000E+00",
            "  953 1.84334E-05   962 0.00000E+00   963 0.00000E+00   964 1.83577E-05",
            "  965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "  814 0.00000E+00     8 1.69491E-02    11 7.32332E-03    24 2.42060E-03",
            "   25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
            "  603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
            "    6 0.00000E+00",
```

```
"    2",
"   925 1.99109E-05   926 0.00000E+00   937 9.45050E-06   928 6.53346E-03",
"   939 0.00000E+00   948 2.06825E-05   949 1.01294E-03   940 5.98514E-04",
"   941 7.98413E-05   942 7.21147E-05   951 3.71706E-05   950 0.00000E+00",
"   953 1.84334E-05   962 0.00000E+00   963 0.00000E+00   964 1.83577E-05",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 1.69491E-02    11 7.32332E-03    24 2.42060E-03",
"    25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
"   603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    3",
"   925 1.92901E-05   926 0.00000E+00   937 1.05223E-05   928 6.32977E-03",
"   939 0.00000E+00   948 2.30282E-05   949 1.12782E-03   940 6.66396E-04",
"   941 8.88966E-05   942 8.02937E-05   951 4.13863E-05   950 0.00000E+00",
"   953 2.05241E-05   962 0.00000E+00   963 0.00000E+00   964 2.04398E-05",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 1.69639E-02    11 7.32332E-03    24 2.42060E-03",
"    25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
"   603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    4",
"   925 1.92901E-05   926 0.00000E+00   937 1.05223E-05   928 6.32977E-03",
"   939 0.00000E+00   948 2.30282E-05   949 1.12782E-03   940 6.66396E-04",
"   941 8.88966E-05   942 8.02937E-05   951 4.13863E-05   950 0.00000E+00",
"   953 2.05241E-05   962 0.00000E+00   963 0.00000E+00   964 2.04398E-05",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 1.69639E-02    11 7.32332E-03    24 2.42060E-03",
"    25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
"   603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    5",
"   925 2.46280E-05   926 0.00000E+00   937 0.00000E+00   928 8.08131E-03",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 1.62119E-02    11 7.32332E-03    24 2.42060E-03",
"    25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
"   603 0.00000E+00    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    6",
"   925 2.46280E-05   926 0.00000E+00   937 0.00000E+00   928 8.08131E-03",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 1.62119E-02    11 7.32332E-03    24 2.42060E-03",
"    25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
"   603 0.00000E+00    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    7",
"   925 3.33890E-05   926 0.00000E+00   937 0.00000E+00   928 1.09561E-02",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 2.19790E-02    11 6.61882E-03    24 2.01073E-03",
"    25 1.03803E-04    26 1.45512E-02    28 6.47780E-05    42 4.95338E-05",
"   603 0.00000E+00    74 1.03395E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    8",
"   925 3.33890E-05   926 0.00000E+00   937 0.00000E+00   928 1.09561E-02",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 2.19790E-02    11 6.61882E-03    24 2.01073E-03",
"    25 1.03803E-04    26 1.45512E-02    28 6.47780E-05    42 4.95338E-05",
"   603 0.00000E+00    74 1.03395E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    9",
"   925 3.33890E-05   926 0.00000E+00   937 0.00000E+00   928 1.09561E-02",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 2.19790E-02    11 6.61882E-03    24 2.01073E-03",
"    25 1.03803E-04    26 1.45512E-02    28 6.47780E-05    42 4.95338E-05",
"   603 0.00000E+00    74 1.03395E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"   10",
"   925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
```

```
"    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"    814 0.00000E+00     8 0.00000E+00    11 2.03057E-02    24 1.21654E-03",
"     25 1.34329E-04    26 4.58722E-03    28 1.43655E-03    42 1.09920E-04",
"    603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"      6 0.00000E+00",
"    11",
"    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"    814 0.00000E+00     8 0.00000E+00    11 7.22756E-03    24 3.18543E-03",
"     25 3.51733E-04    26 1.20114E-02    28 3.76282E-03    42 2.87732E-04",
"    603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"      6 0.00000E+00",
"    12",
"    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"    814 0.00000E+00     8 0.00000E+00    11 6.85516E-03    24 2.51265E-03",
"     25 2.77445E-04    26 9.47455E-03    28 2.96810E-03    42 2.26961E-04",
"    603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"      6 0.00000E+00",
"    13",
"    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"    814 0.00000E+00     8 0.00000E+00    11 7.22164E-03    24 2.28020E-03",
"     25 2.51777E-04    26 8.59800E-03    28 2.69258E-03    42 2.06026E-04",
"    603 0.00000E+00    74 0.00000E+00   105 1.05361E-02   115 4.21443E-02",
"      6 1.31701E-02",
"    14",
"    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"    814 0.00000E+00     8 0.00000E+00    11 5.55674E-03    24 1.04735E-02",
"     25 1.15647E-03    26 3.94927E-02    28 1.23677E-02    42 9.46329E-04",
"    603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"      6 0.00000E+00",
"    15",
"    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"    814 0.00000E+00     8 0.00000E+00    11 6.38300E-03    24 9.95491E-03",
"     25 1.09921E-03    26 3.75373E-02    28 1.17553E-02    42 8.99472E-04",
"    603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"      6 0.00000E+00",
"    16",
"    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"    814 0.00000E+00     8 0.00000E+00    11 4.28467E-03    24 2.89636E-03",
"     25 3.19813E-04    26 1.09214E-02    28 3.42018E-03    42 2.61699E-04",
"    603 0.00000E+00    74 0.00000E+00   105 1.22001E-02   115 4.88005E-02",
"      6 1.52502E-02",
"    0                                                                    "
]
refuel_inp = CardFile( refuel_card )
refuel.inputs(refuel_inp)
refuel.run( "cyc01_BOC" )
refuel_files_cyl_boc = refuel.outfiles()


# startup
startup_card = [
    "16,33,18",
    "18",
    "949,1,1,0.0",
    "949,1,2,0.0",
```

```
    "949,1,3,0.0",
    "949,1,4,0.0",
    "949,1,5,0.0",
    "949,1,6,0.0",
    "949,1,7,0.0",
    "949,1,8,0.0",
    "949,1,9,0.0",
    "949,1,10,0.0",
    "949,1,11,0.0",
    "949,1,12,0.0",
    "949,1,13,0.0",
    "949,1,14,0.0",
    "949,1,15,0.0",
    "949,1,16,0.0",
    "949,1,17,0.0",
    "949,1,18,0.0"
    ]
startup_inp = CardFile( startup_card )
startup.inputs( [startup_inp] + micro_file + refuel_files_cyl_boc )
startup.run( "cyc01-BOC")
macro_file = startup.outfiles()

list_file = startup.list()
assert list_file.has_same_contents_with( DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/STARTUP-cyc01-BOC.dmp" ) )


# citation
cit_card_forward = [
    "FISSPEC ",
    "  0  0  0  1  1  1  1  0  0  0"
 ]
cit_card_adjoint = [
    "FISSPEC ",
    "  0  0  0  1  1  1  1  0  1  0"
 ]

cit_card_body = [
    "  0  1  0",
    "  1.0",
    "       300   1200",
    " 1.0   E-5 1.0   E-5",
    "         1",
    "        68        62         1        18        21        13         1",
    "     2     3     3     3     2     4     4     3     2     5     4     4",
    "     2     3     2     2     4     4     4     5     3",
    "  9.5504 15.7176 16.3613 16.4636  5.9732 18.6429 17.9109 12.7851",
    "  4.7272 21.2517 17.8360 18.1027  4.6213 13.2016 10.1232  3.5586",
    " 18.9745 17.4992 17.3742 32.6015 17.2216",
    "     5     6     4     3     3     3     4     3     3     3     4    16",
    "     5",
    " 30.0000 30.0000 20.0000 15.0000 15.0000 15.0000 20.0000 15.0000",
    " 15.0000 15.0000 20.0000 100.000 30.0000",
    "     1",
    "    1.00",
    " 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
    " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
    " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
    " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
    " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
    " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
    " 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
    "       0.0    0.4692    0.4692    0.4692    0.0       0.0    0.0",
    "        17        17",
 ]
cit_card_region_cyl_forward = [
    "        16"
    "KEFF 01B ",
    "AVFX 01B",
    "VOL  01B"
    ]
cit_card_region_cyl_adjoint = [
    "KEFFa01B ",
    "AVFXa01B",
    "VOL a01B"
    ]
```

```
print "Macrofile(before fwd):", macro_file[0].generate_code()
print "Macrofile(before fwd):", macro_file[0].filetype()

cit_inp_cyl_boc = CardFile( cit_card_forward + cit_card_body + cit_card_region_cyl_forward )
citation.inputs([cit_inp_cyl_boc, database] + macro_file)
citation.run( "cyc01-BOC-forward" )

print "Macrofile(after fwd):", macro_file[0].generate_code()
print "Macrofile(after fwd):", macro_file[0].filetype()


list_file = citation.list()
ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CIT-cyc01-BOC.dmp" )
assert list_file.has_same_contents_with( ref_list, "ITERATION TIME"  )


cit_inp_cyl_boc_adj = CardFile( cit_card_adjoint + cit_card_body + cit_card_region_cyl_adjoint )
assert          cit_inp_cyl_boc_adj.has_same_contents_with(          DataFile(          ref1_dir          +
"/H13-Na-MOX.C3/BR/Burn/CIT-cyc01-BOC-adj" ) )

print "Macrofile(before adj):", macro_file[0].generate_code()
print "Macrofile(before adj):", macro_file[0].filetype()

citation.inputs([cit_inp_cyl_boc_adj, database] + macro_file)
citation.run( "cyc01-BOC-adjoint" )

list_file = citation.list()
ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CIT-cyc01-BOC-adj.dmp" )
assert list_file.has_same_contents_with( ref_list, "ITERATION TIME"  )


# fire1
fire1_head_cyl = [
    "AVFX 01B",
    "VOL  01B ",
    "PWFT 01B",
    "PWFX 01B"
    ]

fire_body = [
    "16,21,18,10,0,17,1,1",
    "3570.00E+06,0.0,0.0",
    "47347200,20",
    "  5, 3.4044E-06,  1,  7, 1.0",
    "  7, 9.1100E-13,  1,  1, 1.0",
    "  8, 3.3460E-12,  1,  2, 1.0",
    "  9, 1.5263E-09,  1, 11, 1.0",
    " 11, 5.0820E-11,  1,  3, 1.0",
    " 12, 1.5600E-10,  1, 10, 0.173",
    " 12, 1.5600E-10,  1, 14, 0.827",
    " 14, 4.9282E-08,  1,  6, 1.0",
    " 15, 7.5500E-10,  1,  7, 1.0",
    " 16, 1.2130E-09,  1,  8, 1.0",
    "  1,  2, 1.0",
    "  2,  3, 1.0",
    "  3,  6, 1.0",
    "  4,  5, 1.0",
    "  6,  7, 1.0",
    "  7,  8, 1.0",
    "  8,  9, 1.0",
    "  9, 10, 1.0",
    " 10, 13, 1.0",
    " 11, 10, 0.1384",
    " 11, 12, 0.200",
    " 11, 14, 0.6616",
    " 12, 13, 1.0",
    " 13, 16, 1.0",
    " 14, 15, 1.0",
    " 15, 16, 1.0",
    " 16, 17, 1.0",
    "0,0,0.0",
    "  2,  1, 1.0",
    "  4,  3, 1.0",
    "  6,  3, 1.0",
    "  7,  6, 1.0",
    "  8,  7, 1.0",
    "  9,  8, 1.0",
    " 10,  9, 1.0",
    " 11,  8, 1.0",
    " 12, 11, 1.0",
    " 14, 11, 1.0",
    " 15, 14, 1.0",
```

```
    " 16, 15, 1.0",
    " 17, 16, 1.0",
    "0,0,0.0",
    "   U 235 925   U 236 926  NP-237 937   U 238 928  NP 239 939  PU 248 948",
    "  PU 239 949  PU 240 940  PU 241 941  PU 242 942  AM 241 951  AM 942 950",
    "  AM 243 953  CM 242 962  CM 243 963  CM 244 964  CM 245 965  FP 854 854",
    "  FP 884 884  FP 894 894  FP 814 814   ",
    "0,0",
    "   1, 0.312700E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
    "  2, 0.334000E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
    "  3, 0.333600E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
    "  4, 0.350000E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
    "  5, 0.333100E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "  6, 0.315500E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
    "  7, 0.322200E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
    "  8, 0.324400E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "  9, 0.324800E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 10, 0.330100E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 11, 0.353800E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 12, 0.344900E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 13, 0.344300E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 14, 0.351000E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 15, 0.351600E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 16, 0.340100E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    " 17, 0.352900E-10",
    " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0"
    ]
fire1_inp = CardFile( fire1_head_cy1 + fire_body )

fire1.inputs([fire1_inp, micro_file, refuel_files_cy1_boc, database])
fire1.run()
list_file = fire1.list()
ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/FIRE1_cyc01.dmp" )
assert list_file.has_same_contents_with( ref_list )


# fire2
fire2_head_cy1 = [
    "AVFX 01B",
    "VOL  01B ",
    "PWFT 01B",
    ]

fire2_inp = CardFile( fire2_head_cy1 + fire_body )

fire2.inputs([fire2_inp, micro_file, refuel_files_cy1_boc, database])
fire2.run()
fire2_files_cy1_boc = fire2.outfiles()
list_file = fire2.list()
ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/FIRE2_cyc01.dmp" )
assert list_file.has_same_contents_with( ref_list )


# refuel cy01 EOC
refuel_card = [
    "    1    1                                                                  ",
    "    1   21                                                                  ",
    "    1   21                                                                  ",
    ]
refuel_inp = CardFile( refuel_card )
refuel.inputs( [refuel_inp, refuel_files_cy1_boc, fire2_files_cy1_boc] )
refuel.run( "cyc01_EOC" )
refuel_files_cy1_eoc = refuel.outfiles()

# startup cy1 EOC
startup_inp = CardFile( startup_card )
```

```
        startup.inputs( [startup_inp, micro_file, refuel_files_cyl_eoc ] )
        startup.run( "cyc01-EOC" )
        macro_file_cyl_eoc = startup.outfiles()

        list_file = startup.list()
        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/STARTUP-cyc01-EOC.dmp" )
        assert list_file.has_same_contents_with( ref_list )



        # citation cyl eoc forward
        cit_card_region_cyl_forward_eoc = [
             "          16"
            "KEFF 01E ",
            "AVFX 01E",
            "VOL  01E"
            ]
        cit_card_region_cyl_adjoint_eoc = [
            "KEFFa01E ",
            "AVFXa01E",
            "VOL a01E"
            ]

        cit_inp_cyl_eoc = CardFile( cit_card_forward + cit_card_body + cit_card_region_cyl_forward_eoc )
        citation.inputs([cit_inp_cyl_eoc, database, macro_file_cyl_eoc ])
        citation.run( "cyc01-EOC-forward" )

        list_file = citation.list()
        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CIT-cyc01-EOC.dmp" )
        assert list_file.has_same_contents_with( ref_list, "ITERATION TIME"  ), list_file.diff_from( ref_list, ''  )


        cit_inp_cyl_eoc_adj = CardFile( cit_card_adjoint + cit_card_body + cit_card_region_cyl_adjoint_eoc )
        citation.inputs([cit_inp_cyl_eoc_adj, database, macro_file_cyl_eoc ])
        citation.run( "cyc01-EOC-adjoint" )

        list_file = citation.list()
        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CIT-cyc01-EOC-adj.dmp" )
        assert list_file.has_same_contents_with( ref_list, "ITERATION TIME"  )


def suite():
    return unittest.makeSuite( BR_Forward_Cycle1_Test, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-3 CardFileTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *

class CardFileTest(unittest.TestCase):

    def testWriteAndRead(self):
        a = CardFile()
        b = CardFile()
        text = [ "12345", "abcdef" ]
        a.create()
        b.create()

        a.write( text )
        b.write( text )
        a.close()
        b.close()

        del a   # a.purge() is automatically called
        del b   # ditto

        d = CardFile()
        e = CardFile()

        read_d = d.read()
        read_e = e.read()

        assert read_d == read_e
        assert d.has_same_contents_with(e) == True


    def testLoad(self):
        text = ["0123456789","abcdefghij","ABCDEFGHIJ" ]

        a = CardFile()
        a.create()
        a.write(text)
        a.close()

        b = CardFile()
        print b.pathname()
        b.load(text)

        assert b.has_same_contents_with(a) == True


    def testInit(self):

        card_text = [
            "   33",
            "  925  926  937  928  939  948  949  940  941  942  951  950  953  962",
            "  963  964  965  854  884  894  814    8   11   24   25   26   28   42",
            "  603   74  105  115    6"
            ]

        inp = CardFile( card_text )

        inp_path = inp.pathname()
        a = DataFile( inp_path )

        a_text = a.read()

        assert a_text == card_text


def suite():
    return unittest.makeSuite( CardFileTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-4 CitationTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.ModuleFile import *
from string import *

class CitationTest(unittest.TestCase):

    def testForward(self):
        citation = Citation()

        card_text = [
            "FISSPEC ",
            "   0  0  0  1  1  1  1  0  0  0",
            "   0  1  0",
            "   1.0",
            "        300   1200",
            " 1.0   E-5 1.0   E-5",
            "          1",
            "         68       62        1       18       21       13        1",
            "      2     3     3     3     2     4     4     3     2     5     4     4",
            "      2     3     2     2     4     4     4     5     3",
            "   9.5504 15.7176 16.3613 16.4636  5.9732 18.6429 17.9109 12.7851",
            "   4.7272 21.2517 17.8360 18.1027  4.6213 13.2016 10.1232  3.5586",
            " 18.9745 17.4992 17.3742 32.6015 17.2216",
            "      5     6     4     3     3     3     4     3     3     3     4    16",
            "      5",
            " 30.0000 30.0000 20.0000 15.0000 15.0000 15.0000 20.0000 15.0000",
            " 15.0000 15.0000 20.0000 100.000 30.0000",
            "      1",
            "    1.00",
            " 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
            "        0.0    0.4692    0.4692    0.4692    0.0       0.0    0.0",
            "         17        17",
            "         16",
            "KEFF 01B",
            "AVFX 01B",
            "VOL  01B"
            ]
        inp = CardFile( card_text )

        # Unit 10
        macro_file = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc01-BOC" )
        macro_file.set_generate_code( "STARTUP" )
        macro_file.set_filetype( "MACRO_XS" )

        # Unit 51
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

        citation.inputs([inp, macro_file, database])
        citation.run( "Cy01-BOC" )
        list_file = citation.list()

        ref_list = ListFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CIT-cyc01-BOC.dmp" )
        assert list_file.has_same_contents_with( ref_list, "ITERATION TIME"  )

        out_files = citation.outfiles()
        assert len(out_files) == 3 , len(out_files)   # scratch file is ignored

        # unit 2 (binary)
        assert out_files[0].generate_code() == "CITATION", out_files[0].generate_code()
        assert out_files[0].filetype() == "MESH_FLUX", out_files[0].filetype()
        assert atoi(out_files[0].getProperty( "NX" ) ) == 68
        assert out_files[0].casename() == "Cy01-BOC_FORWARD", out_files[0].casename()
```

```
        # unit 20 (text)
        assert out_files[1].generate_code() == "CITATION", out_files[1].generate_code()
        assert out_files[1].filetype() == "LEAKAGE_TERM", out_files[1].filetype()
        assert             out_files[1].has_same_contents_with(             RefFile(             ref1_dir             +
"/H13-Na-MOX.C3/BR/WORK/citation.FT20/cyc01-BOC" ) )

        # unit 109 (text)
        assert out_files[2].generate_code() == "CITATION", out_files[3].generate_code()
        assert out_files[2].filetype() == "AXIAL_BUCKLING", out_files[3].filetype()
        assert             out_files[2].has_same_contents_with(             RefFile(             ref1_dir             +
"/H13-Na-MOX.C3/BR/WORK/citation.FT09/cyc01-BOC" ) )


    def testAdjoint(self):
        citation = Citation()

        card_text = [
            "FISSPEC ",
            "  0  0  0  1  1  1  1  0  1  0",
            "  0  1  0",
            "  1.0",
            "       300   1200",
            " 1.0   E-5 1.0   E-5",
            "         1",
            "        68       62        1       18       21       13        1",
            "     2     3     3     3     2     4     4     3     2     5     4     4",
            "     2     3     2     2     4     4     4     5     3",
            "  9.5504 15.7176 16.3613 16.4636  5.9732 18.6429 17.9109 12.7851",
            "  4.7272 21.2517 17.8360 18.1027  4.6213 13.2016 10.1232  3.5586",
            " 18.9745 17.4992 17.3742 32.6015 17.2216",
            "     5     6     4     3     3     3     4     3     3     3     4    16",
            "     5",
            " 30.0000 30.0000 20.0000 15.0000 15.0000 15.0000 20.0000 15.0000",
            " 15.0000 15.0000 20.0000 100.000 30.0000",
            "     1",
            "    1.00",
            " 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
            "        0.0    0.4692    0.4692    0.4692    0.0       0.0    0.0",
            "         17        17",
            "KEFFa01B",
            "AVFXa01B",
            "VOL a01B"
            ]
        inp = CardFile( card_text )

        # Unit 10
        macro_file = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc01-BOC" )
        macro_file.set_generate_code( "STARTUP" )
        macro_file.set_filetype( "MACRO_XS" )

        # Unit 51
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

        citation.inputs([inp, macro_file, database])
        citation.run( "Cy01-BOC-adj" )
        list_file = citation.list()

        ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CIT-cyc01-BOC-adj.dmp" )
        assert list_file.has_same_contents_with( ref_list, "ITERATION TIME"  )

        out_files = citation.outfiles()
        assert len(out_files) == 3 , len(out_files)   # scratch file is ignored

        # unit 2 (binary)
        assert out_files[0].generate_code() == "CITATION", out_files[0].generate_code()
        assert out_files[0].filetype() == "MESH_FLUX", out_files[0].filetype()
        assert atoi(out_files[0].getProperty( "NX" ) ) == 68
        assert out_files[0].casename() == "Cy01-BOC-adj_ADJOINT", out_files[0].casename()

        # unit 20 (text)
```

```
        assert out_files[1].generate_code() == "CITATION", out_files[1].generate_code()
        assert out_files[1].filetype() == "LEAKAGE_TERM", out_files[1].filetype()
        assert             out_files[1].has_same_contents_with(             RefFile(             ref1_dir             +
"/H13-Na-MOX.C3/BR/WORK/citation.FT20/cyc01-BOC-adj" ) )

        # unit 109 (text)
#        assert out_files[3].generate_code() == "CITATION", out_files[3].generate_code()
#        assert out_files[3].filetype() == "AXIAL_BUCKLING", out_files[3].filetype()
#                                          assert     out_files[3].has_same_contents_with(     RefFile(     ref1_dir     +
"/H13-Na-MOX.C3/BR/WORK/citation.FT09/cyc01-BOC-adj" ) )


def suite():
#     return unittest.makeSuite( CitationTest, 'test' )
      suite = unittest.TestSuite()
#      suite.addTest( CitationTest( "testForward" ))
      suite.addTest( CitationTest( "testAdjoint" ))
      return suite


if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-5 ConvmicTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.ModuleFile import *
from sagep.File.MicroFile import *

class ConvmicTest(unittest.TestCase):

    def testRun(self):
        convmic = Convmic()

        card_text = [
            "    33",
            "  925  926  937  928  939  948  949  940  941  942  951  950  953  962",
            "  963  964  965  854  884  894  814    8   11   24   25   26   28   42",
            "  603   74  105  115    6"
            ]
        inp = CardFile( card_text )
        micro_file = MicroFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICRO18g" )

        convmic.inputs([inp, micro_file])
        convmic.run()
        list_file = convmic.list()

        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CONVMIC.dmp" )
        assert list_file.has_same_contents_with( ref_list )

        out_files = convmic.outfiles()
        assert len(out_files) == 1
        assert out_files[0].filetype() == "SAGEP-BURN_MICRO_XS"
        assert out_files[0].generate_code() == "CONVMIC"


def suite():
    return unittest.makeSuite( ConvmicTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-6 DataBaseTest.py

```
import unittest
from  sagep.config import *
from sagep.File.DataBase import *
from copy import *

class DataBaseTest(unittest.TestCase):
    def setUp(self):
        pass

    def tearDown(self):
        pass

    def testHasEntry(self):
        db = DataBase( ref1_dir + "/H13-Na-MOX.C3.ref/BR/DataBase2" )
        assert db.hasEntry( "FISSPEC" ) != -1
        assert db.hasEntry( "NOT_HAVE_THIS_ENTRY" ) == -1


    def testGetEntry(self):
        db = DataBase( ref1_dir + "/H13-Na-MOX.C3.ref/BR/DataBase2" )
        fisspec = db.getEntry( "FISSPEC" )

        ref = [2.74940E-02, 1.20238E-01, 2.22775E-01, 2.34681E-01, 1.73057E-01, 1.40719E-01,
               5.36364E-02, 1.83373E-02, 6.11115E-03, 1.99959E-03, 6.49155E-04, 2.12088E-04,
               6.88247E-05, 2.22676E-05, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00
               ]

        assert fisspec == ref

        sfact = db.getEntry( "SFACT" )
        assert sfact == -1.0000000000E+00

        dummy = db.getEntry( "DUMMY_NOT_FOUND" )
        assert dummy == ""


    def testRemoveEntry(self):
        db = DataBase( ref1_dir + "/H13-Na-MOX.C3.ref/BR/DataBase2" )

        db.removeEntry( "FISSPEC" )
        assert db.hasEntry( "FISSPEC" ) == -1



    def testInsertEntry(self):
        data = [1,2,3,4,5,6,7,8,9,10]
        db = DataBase( ref1_dir + "/H13-Na-MOX.C3.ref/BR/DataBase" )
        db.insertEntry( "TEST", data, 6, "E12.5", pos=-1 )
        #db.show()
        data2 = db.getEntry( "TEST" )
        print data
        print data2
        assert data2 == data


    def testSetEntry(self):
        data = [1,2,3,4,5,6,7,8,9,10]
        db = DataBase( ref1_dir + "/H13-Na-MOX.C3.ref/BR/DataBase" )

        db.defineFormat("TEST", 6, "F8.4")
        db.defineFormat("FISSPEC", 6, "E10.2")

        db.setEntry( "TEST", data )
        db.setEntry( "FISSPEC", data )
        data2 = db.getEntry( "FISSPEC" )
        assert data2 == data

    def testSetEntryBlocks(self):
        entry_block = ['FISSPEC¥n',
                       '  1.00E+00  2.00E+00  3.00E+00  4.00E+00  5.00E+00  6.00E+00¥n',
                       '  7.00E+00  8.00E+00  9.00E+00  1.00E+01¥n'
                       ,'SFACT¥n',
                       '  -1.0000000000E+00¥n',
                       'TEST¥n',
                       '  1.0000  2.0000  3.0000  4.0000  5.0000  6.0000¥n',
                       '  7.0000  8.0000  9.0000 10.0000¥n',
                       'END¥n'
                       ]
        db = DataBase( ref1_dir + "/H13-Na-MOX.C3.ref/BR/DataBase" )
```

```
        db.setEntryBlocks(entry_block)
        #db.show()

        data2 = db.getEntry( "TEST" )
        data = [1,2,3,4,5,6,7,8,9,10]
        #print data
        #print data2
        assert data2 == data

    def testDiffAndSet(self):
        entry_block = ['FISSPEC¥n',
                       '   1.00E+00  2.00E+00  3.00E+00  4.00E+00  5.00E+00  6.00E+00¥n',
                       '   7.00E+00  8.00E+00  9.00E+00  1.00E+01¥n'
                       'SFACT¥n',
                       '  -1.0000000000E+00¥n',
                       'TEST¥n',
                       '   1.0000  2.0000  3.0000  4.0000  5.0000  6.0000¥n',
                       '   7.0000  8.0000  9.0000 10.0000¥n',
                       'END¥n'
                       ]
        db = DataBase( refl_dir + "/H13-Na-MOX.C3.ref/BR/DataBase" )
        db_org = deepcopy(db)

        print "****************************"
        db.setEntryBlocks(entry_block)
        print "============================"
        diff = db.diff_from( db_org )
        print "----------------------------"
        print diff

        db_org.setEntryBlocks(diff)

        diff = db.diff_from( db_org )


        assert len(diff) == 1
        assert diff[0] == 'END¥n'


    def testSetEntryBlocks2(self):
        entry_block = ['KEFF 01B¥n',
                                  1.065947¥n',
                       'AVFX 01B¥n',
                       ' 3.04200E-02 1.44101E-01 3.58623E-01 5.23008E-01 6.18121E-01 1.60867E+00¥n',
                       ' 1.91446E+00 2.00639E+00 1.68933E+00 1.28785E+00 8.93881E-01 4.51927E-01¥n',
                       ' 1.65946E-01 2.49575E-01 1.03763E-01 2.37773E-02 4.07610E-03 6.29563E-04¥n',
                       ' 2.74463E-02 1.29918E-01 3.23330E-01 4.70445E-01 5.56431E-01 1.44936E+00¥n',
                       ' 1.72037E+00 1.79636E+00 1.50701E+00 1.14349E+00 7.89252E-01 3.96448E-01¥n',
                       ' 1.45751E-01 2.15194E-01 8.64037E-02 1.92418E-02 3.15150E-03 4.54734E-04¥n',
                       ' 2.22242E-02 1.05613E-01 2.62398E-01 3.83781E-01 4.49489E-01 1.14570E+00¥n',
                       ' 1.34929E+00 1.39827E+00 1.15724E+00 8.73401E-01 5.98673E-01 3.02061E-01¥n',
                       ' 1.08051E-01 1.68363E-01 7.05404E-02 1.58751E-02 2.69464E-03 4.16673E-04¥n',
                       ' 1.38603E-02 6.57062E-02 1.62893E-01 2.35344E-01 2.76509E-01 7.07230E-01¥n',
                       ' 8.28969E-01 8.55787E-01 7.10540E-01 5.35772E-01 3.67715E-01 1.83449E-01¥n',
                       ' 6.70329E-02 9.82132E-02 3.85655E-02 8.67813E-03 1.50922E-03 2.88453E-04¥n',
                       ' 2.41499E-03 1.10696E-02 3.00564E-02 4.90950E-02 6.39754E-02 2.25371E-01¥n',
                       ' 3.30689E-01 4.18668E-01 4.18248E-01 3.69361E-01 2.93315E-01 1.67050E-01¥n',
                       ' 6.76429E-02 1.34138E-01 8.49535E-02 4.04630E-02 1.84904E-02 1.07157E-02¥n',
                       ' 2.42058E-03 1.10880E-02 3.01531E-02 4.92128E-02 6.41419E-02 2.26518E-01¥n',
                       ' 3.32665E-01 4.21249E-01 4.20938E-01 3.71960E-01 2.95402E-01 1.68321E-01¥n',
                       ' 6.81202E-02 1.35661E-01 8.67410E-02 4.15301E-02 1.92394E-02 1.30017E-02¥n',
                       ' 3.06688E-04 1.35810E-03 3.75119E-03 5.98597E-03 8.08542E-03 3.39257E-02¥n',
                       ' 5.55756E-02 7.72529E-02 8.56931E-02 8.06960E-02 6.82169E-02 4.13534E-02¥n',
                       ' 1.77408E-02 3.26851E-02 2.20068E-02 1.17249E-02 6.34711E-03 5.70724E-03¥n',
                       ' 2.13603E-03 9.67388E-03 2.55571E-02 3.84539E-02 4.90250E-02 1.69504E-01¥n',
                       ' 2.37069E-01 2.86740E-01 2.77671E-01 2.36506E-01 1.80338E-01 1.02325E-01¥n',
                       ' 4.24683E-02 7.14118E-02 4.15233E-02 1.92394E-02 9.32023E-03 8.17601E-03¥n',
                       ' 3.08264E-04 1.36440E-03 3.77266E-03 6.01366E-03 8.12260E-03 3.41558E-02¥n',
                       ' 5.60469E-02 7.79395E-02 8.65208E-02 8.15421E-02 6.88935E-02 4.17857E-02¥n',
                       ' 1.79209E-02 3.31597E-02 2.25585E-02 1.21251E-02 6.68674E-03 6.55826E-03¥n',
                       ' 6.37703E-03 3.02886E-02 7.71488E-02 1.19805E-01 1.40786E-01 3.96405E-01¥n',
                       ' 4.80726E-01 5.50939E-01 4.79824E-01 3.87960E-01 2.87133E-01 1.48376E-01¥n',
                       ' 4.25366E-02 1.41626E-01 7.72148E-02 3.36402E-02 1.97302E-02 2.80431E-02¥n',
                       ' 4.89434E-05 2.22769E-04 7.37923E-04 1.81355E-03 2.95379E-03 1.57773E-02¥n',
                       ' 2.86412E-02 4.77977E-02 5.71019E-02 5.36863E-02 4.34445E-02 3.05319E-02¥n',
                       ' 1.23744E-02 4.45157E-02 4.32859E-02 2.86900E-02 2.59089E-02 5.13112E-02¥n',
                       ' 1.02463E-05 4.61800E-05 1.53233E-04 3.83345E-04 6.76224E-04 3.94030E-03¥n',
                       ' 7.56008E-03 1.27936E-02 1.61199E-02 1.51741E-02 1.19533E-02 8.63090E-03¥n',
                       ' 3.50566E-03 1.31356E-02 1.28778E-02 8.71460E-03 8.41173E-03 1.89441E-02¥n',
                       ' 5.34470E-06 2.67395E-05 8.70625E-05 2.50905E-04 5.01231E-04 1.72005E-03¥n',
                       ' 3.19852E-03 4.82591E-03 5.87704E-03 5.26291E-03 3.86008E-03 2.28953E-03¥n',
                       ' 8.09302E-04 9.16353E-04 6.52689E-04 2.44386E-04 1.28194E-04 8.22534E-05¥n',
```

```
                    ' 2.40973E-08 1.23228E-07 7.32750E-07 5.21813E-06 2.00269E-05 2.30467E-04¥n',
                    ' 6.10642E-04 1.11402E-03 1.69407E-03 1.47671E-03 1.02718E-03 8.39914E-04¥n',
                    ' 5.56448E-04 1.36270E-03 1.52247E-03 1.15211E-03 1.42291E-03 5.51670E-03¥n',
                    ' 2.38135E-05 1.05019E-04 3.53323E-04 8.46776E-04 1.62422E-03 8.68307E-03¥n',
                    ' 1.43019E-02 1.98097E-02 2.37191E-02 1.91809E-02 1.32952E-02 1.04371E-02¥n',
                    ' 6.52548E-03 1.49436E-02 1.29188E-02 8.27047E-03 8.09534E-03 1.72820E-02¥n',
                    ' 1.12166E-07 5.64961E-07 2.41017E-06 1.09658E-05 4.18612E-05 2.29101E-04¥n',
                    ' 4.82011E-04 6.59444E-04 8.11682E-04 6.78666E-04 4.55081E-04 2.56630E-04¥n',
                    ' 1.02874E-04 9.27759E-05 6.10022E-05 2.28748E-05 1.59263E-05 1.79040E-05¥n',
                    'VOL  01B¥n',
                     '  1.00E+00  2.00E+00  3.00E+00  4.00E+00  5.00E+00  6.00E+00¥n',
                     '  7.00E+00  8.00E+00  9.00E+00  1.00E+01¥n',
                    'END¥n'
                    ]

        db = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )
        db.setEntryBlocks(entry_block)
        #db.show()

        data2 = db.getEntry( "VOL  01B" )
        data = [1,2,3,4,5,6,7,8,9,10]
        print data
        print data2
        assert data2 == data

        db.restore(path="/tmp/database")


def suite():
    return unittest.makeSuite( DataBaseTest, 'test' )
#     suite = unittest.TestSuite()
#     suite.addTest( DataBaseTest( "testHasEntry" ))
#     suite.addTest( DataBaseTest( "testGetEntry" ))
#     suite.addTest( DataBaseTest( "testHasEntry" ))
#     suite.addTest( DataBaseTest( "testRemoveEntry" ))
#     suite.addTest( DataBaseTest( "testInsertEntry" ))
#     suite.addTest( DataBaseTest( "testSetEntry" ))
#     suite.addTest( DataBaseTest( "testSetEntryBlocks" ))
#     suite.addTest( DataBaseTest( "testDiffAndSet" ))
#     suite.addTest( DataBaseTest( "testSetEntryBlocks2" ))
#     return suite

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-7 DataFileTest.py

```
import unittest
from  sagep.config import *

from sagep.File.DataFile import *

class DataFileTest(unittest.TestCase):
    def setUp(self):
        self.f = DataFile("/tmp/test.dat")
        self.f.setworkdir( "/tmp" )

    def tearDown(self):
        pass

    def testConstructor(self):
        pass

    def testExists(self):
        assert self.f.exists() == False

        b = DataFile( "/etc/inittab" )
        assert b.exists() == True

    def testWorkdir(self):
        assert self.f.workdir() == "/tmp"
        b = DataFile()
        assert b.workdir() == "/tmp"

    def testDirname(self):
        assert self.f.dirname() == "/tmp"

    def testFileName(self):
        assert self.f.filename() == "test.dat"

    def testPathName(self):
        assert self.f.pathname() == "/tmp/test.dat"
        c = DataFile()

    def testCreateAndPurge(self):
        a = DataFile( "/tmp/aaa.test" )
        a.create()
        assert a.exists() == True
        a.purge(force=True)
        assert a.exists() == False

    def testOpenAndClose(self):
        a = DataFile( "/etc/hosts" )
        assert a.open() == True
        c = a.read()
        assert a.close() == True


    def testGenerateCode(self):
        a = DataFile()

        assert a.generate_code() == ""
        assert a.filetype() == ""

        a.set_generate_code( "CITATION" )
        a.set_filetype( "FLUX" )

        assert a.generate_code() == "CITATION"
        assert a.filetype() == "FLUX"

    def testProperty(self):
        a = DataFile()
        a.setProperty( "NX", 10 )
        a.setProperty( "NY", 30 )
        a.setProperty( "NZ", 1 )
        a.setProperty( "KG", 18 )

        assert a.getProperty( "NX" ) == 10
        assert a.getProperty( "Ny" ) == 30
        assert a.getProperty( "nZ" ) == 1
        assert a.getProperty( "kg" ) == 18

        self.assertRaises( KeyError, a.getProperty, "A" )

        a.delProperty( "NX" )
        self.assertRaises( KeyError, a.getProperty, "NX" )
```

```
        assert a.property_list() == [ "KG", "NY", "NZ" ], a.property_list()


    def testWrite(self):
        text = [ "12345", "abcdef" ]
        a = DataFile()
        b = DataFile()

        a.create()
        b.create()

        a.write(text)
        b.write(text)

        del a   # a.purge() is automatically called
        del b   # ditto

        d = DataFile()
        e = DataFile()

        read_d = d.read()
        read_e = e.read()

        assert read_d == read_e
        assert d.has_same_contents_with(e) == True


    def testCaptureAndRestore(self):
        text = [ "12345", "abcdef" ]
        a = DataFile()
        b = DataFile()
        a.create()
        b.create()

        a.write(text)
        b.write(text)

        filename = a.filename()
        print "filename : ", filename
        assert a.filename() != ""
        assert a.dirname() != ""

        a.capture(delete=True)
        assert a.filename() == ""
        assert a.dirname() == ""
        assert a.is_captured() == True
        assert a.exists() == False

        a.restore()
        assert a.filename() != ""
        assert a.dirname() != ""
        assert a.is_captured() == False

        assert b.has_same_contents_with(a)


    def testCaptureAndRestore2(self):
        "Capture and restore test for binary file"
        a = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICRO18g", capture=False, binary=True )
        b = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICRO18g", capture=False, binary=True )

        a.capture(delete=False)
        a.show()
        assert a.filename() == ""
        assert a.dirname() == ""
        assert a.is_captured() == True
        assert a.exists() == False

        a.restore()
        assert a.filename() != ""
        assert a.dirname() != ""
        assert a.is_captured() == False

        assert b.has_same_contents_with(a)


    def testCaptureAndRestore3(self):
        "Capture and restore test for binary file (2)"
        a = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICRO18g", capture=True)
        assert a.is_captured() == True
        assert a.filename() == ""
```

```
    def testDiff(self):
        text1 = [ "12345", "abcdef" ]
        text2 = [ "12345", "abcdef", "ADDED LINE" ]
        a = DataFile()
        b = DataFile()

        a.create()
        b.create()

        a.write(text1)
        b.write(text2)

        diff1 = b.diff_from( a )
        print "diff1 : ", diff1
        assert diff1 == [ "ADDED LINE¥n" ]

        diff2 = a.diff_from( b, '-' )
        print "diff : ", diff2
        assert diff2 == [ "ADDED LINE¥n" ]


def suite():
    return unittest.makeSuite( DataFileTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-8 ExecutableFileTest.py

```
import unittest
from  sagep.config import *

from sagep.File.ExecutableFile import *
from sagep.File.DataFile import *

class ExecutableFileTest(unittest.TestCase):
    def setUp(self):
        self.exe = ExecutableFile(ref1_dir + "/H13-Na-MOX.C3/CODE/sagep-burn/LM/convmic.lm", "TestCase" )
        self.exe.setworkdir( "/tmp" )
        self.exe.defineio(unit=5, mode="i", comment="user input")
        self.exe.defineio(unit=6, mode="o", comment="output")
        self.exe.defineio(unit=1, mode="i", generated="JOINT", type="MICRO_XS", comment="CITATION Microscopic XS File")
        self.exe.defineio(unit=2, mode="s", comment="working file")
        self.exe.defineio(unit=3, mode="o", type="MICRO_XS", comment="Microscopic XS File")

    def tearDown(self):
        del self.exe

    def testFile1(self):
        inp = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CONVMIC" )
        self.exe.attach(5, inp)
        inp2 = DataFile( "/tmp/fort.5" )
        assert inp2.has_same_contents_with(inp) == True

        file = self.exe.getfile(unit=5)
        print file.pathname()
        #assert file.pathname() == ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CONVMIC"

        self.assertRaises( IOError ,  self.exe.getfile, unit=8 ) # self.exe.getfile(unit=8)


    def testRun(self):
        inp = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CONVMIC" )
        self.exe.attach(5, inp)

        micin = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICRO18g" )
        self.exe.attach(1, micin)

        micout = DataFile( "micout" )
        self.exe.attach(3, micout)

        self.exe.setworkdir( "/tmp" )
        result = self.exe.run()

        assert micout.generate_code() == "EXECUTABLEFILE"
        assert micout.filetype() == "MICRO_XS"

        list_file = self.exe.list()
        ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CONVMIC.dmp" )
        assert list_file.has_same_contents_with( ref_list )

        files = self.exe.outfiles()
        assert len(files) == 1
        assert files[0].filetype() == "MICRO_XS"   # unit 3


    def testInvalidPath(self):
        inp = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/CONVMICaaa" )
        self.assertRaises( IOError, self.exe.attach, 5, inp )

        micout = DataFile( "/ttt/micout" )
        self.assertRaises( IOError, self.exe.attach, 3, micout )

        self.assertRaises( TypeError, self.exe.defineio, unit=3, mode="p", comment="Microscopic XS File" )



    def testInput(self):
        micin = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/MICRO18g" )
        micin.set_generate_code( "JOINT" )
        micin.set_filetype( "MICRO_XS" )

        exe = ExecutableFile(ref1_dir + "/H13-Na-MOX.C3/CODE/sagep-burn/LM/convmic.lm")
        exe.setworkdir( "/tmp" )
        exe.defineio(unit=5, mode="i", comment="user input")
        exe.defineio(unit=6, mode="o", comment="output")
        exe.defineio(unit=1, mode="i", generated="DUMMY", type="MICRO_XS", comment="CITATION Microscopic XS File")
        exe.defineio(unit=2, mode="s", comment="working file")
```

```
        exe.defineio(unit=3, mode="o", comment="Microscopic XS File")

        self.assertRaises( IOError, exe.inputs, micin )
        self.assertRaises( IOError, exe.getfile, unit=10 )


    def testModname(self):
        # print self.exe.modulename()
        assert self.exe.modulename() == "EXECUTABLEFILE"

    def testCaseName(self):
        assert self.exe.casename() == "TestCase"

        self.exe.set_casename("Cy01")
        assert self.exe.casename() == "Cy01"

    def testUnits(self):
        assert self.exe.units() == [1,2,3,5,6]
        assert self.exe.units( "i" ) == [1]
        assert self.exe.units( "o" ) == [3]


    def testHint1(self):
        self.exe = ExecutableFile(ref1_dir + "/H13-Na-MOX.C3/CODE/sagep-burn/LM/convmic.lm", "TestCase" )
        self.exe.defineio(unit=1, mode="i", generated="JOINT", type="MICRO_XS", hint="BOC", comment="CITATION Microscopic
XS File")
        self.exe.defineio(unit=10,   mode="i",   omittable=True,   generated="JOINT",   type="MICRO_XS",   hint="EOC",
comment="CITATION Microscopic XS File")

        f1 = DataFile()
        f1.set_filetype( "MICRO_XS" )
        f1.set_generate_code( "JOINT" )
        f1.set_casename( "Cy-01 BOC" )

        f2 = DataFile()
        f2.set_filetype( "MICRO_XS" )
        f2.set_generate_code( "JOINT" )
        f2.set_casename( "Cy-02 EOC" )

        self.exe.inputs( [f1, f2] )

        of1 = self.exe.getfile(unit=1)
        of2 = self.exe.getfile(unit=10)

        assert of1.pathname() == f1.pathname()
        assert of2.pathname() == f2.pathname()



    def testHint2(self):
        self.exe = ExecutableFile(ref1_dir + "/H13-Na-MOX.C3/CODE/sagep-burn/LM/convmic.lm", "TestCase" )
        self.exe.defineio(unit=1, mode="i", generated="JOINT", type="MICRO_XS", hint="BOC", comment="CITATION Microscopic
XS File")
        self.exe.defineio(unit=10,   mode="i",   omittable=True,   generated="JOINT",   type="MICRO_XS",   hint="EOC",
comment="CITATION Microscopic XS File")

        f3 = DataFile()
        f3.set_filetype( "MICRO_XS" )
        f3.set_generate_code( "JOINT" )
        f3.set_casename( "Test" )
        self.exe.inputs(f3)
        of1 = self.exe.getfile(unit=1)
        self.assertRaises( IOError, self.exe.getfile, unit=10 )

        self.exe.defineio(unit=1,   mode="i",   omittable=True,   generated="JOINT",   type="MICRO_XS",   hint="BOC",
comment="CITATION Microscopic XS File")
        self.exe.defineio(unit=10,  mode="i",    generated="JOINT",  type="MICRO_XS",  hint="EOC",  comment="CITATION
Microscopic XS File")
        self.exe.inputs(f3)
        of1 = self.exe.getfile(unit=10)
        self.assertRaises( IOError, self.exe.getfile, unit=1 )


    def testHint3(self):
        self.exe = ExecutableFile(ref1_dir + "/H13-Na-MOX.C3/CODE/sagep-burn/LM/convmic.lm", "TestCase" )
        self.exe.defineio(unit=1, mode="i", generated="JOINT", type="MICRO_XS", hint="BOC", comment="CITATION Microscopic
XS File")
        self.exe.defineio(unit=10,   mode="i",   omittable=True,   generated="JOINT",   type="MICRO_XS",   hint="EOC",
comment="CITATION Microscopic XS File")

        f1 = DataFile()
        f1.set_filetype( "MICRO_XS" )
```

```
        f1.set_generate_code( "JOINT" )
        f1.set_casename( "Cy-01 BOC" )

        # ommit test
        self.exe.inputs( f1 )
        of1 = self.exe.getfile(unit=1)
        assert of1.pathname() == f1.pathname()


    def testHint4(self):
        self.exe = ExecutableFile(ref1_dir + "/H13-Na-MOX.C3/CODE/sagep-burn/LM/convmic.lm", "TestCase" )
        self.exe.defineio(unit=1, mode="i", generated="JOINT", type="MICRO_XS", hint="BOC", comment="CITATION Microscopic
XS File")
        self.exe.defineio(unit=10,   mode="i",   omittable=True,   generated="JOINT",   type="MICRO_XS",   hint="EOC",
comment="CITATION Microscopic XS File")

        f1 = DataFile()
        f1.set_filetype( "MICRO_XS" )
        f1.set_generate_code( "JOINT" )
        f1.set_casename( "Cy-01 BOC" )

        f4 = DataFile()
        f4.set_filetype( "MICRO_XS" )
        f4.set_generate_code( "JOINT" )
        f4.set_casename( "Cy-02 BOC" )

        # fails for a doubling definition
        self.assertRaises( IOError, self.exe.inputs, [f1, f4] )


def suite():
    return unittest.makeSuite( ExecutableFileTest, 'test' )
#     suite = unittest.TestSuite()
#     suite.addTest( ExecutableFileTest( "testInput" ))
#     return suite
#     suite.addTest( ExecutableFileTest( "testHint1" ))
#     suite.addTest( ExecutableFileTest( "testHint2" ))
#     suite.addTest( ExecutableFileTest( "testHint3" ))
#     suite.addTest( ExecutableFileTest( "testHint4" ))
#     return suite


if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-9 File1Test.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.ModuleFile import *
from sagep.File.DataBase import *
from sagep.File.MicroFile import *
from pprint import *

class File1Test(unittest.TestCase):

    def testForward(self):
        file1 = File1()

        card_text = [
            "AVFX 01B",
            "VOL  01B ",
            "PWFT 01B",
            "PWFX 01B",
            "16,21,18,10,0,17,1,1",
            "3570.00E+06,0.0,0.0",
            "47347200,20",
            "   5, 3.4044E-06,   1,   7, 1.0",
            "   7, 9.1100E-13,   1,   1, 1.0",
            "   8, 3.3460E-12,   1,   2, 1.0",
            "   9, 1.5263E-09,   1,  11, 1.0",
            "  11, 5.0820E-11,   1,   3, 1.0",
            "  12, 1.5600E-10,   1,  10, 0.173",
            "  12, 1.5600E-10,   1,  14, 0.827",
            "  14, 4.9282E-08,   1,   6, 1.0",
            "  15, 7.5500E-10,   1,   7, 1.0",
            "  16, 1.2130E-09,   1,   8, 1.0",
            "   1,   2, 1.0",
            "   2,   3, 1.0",
            "   3,   6, 1.0",
            "   4,   5, 1.0",
            "   6,   7, 1.0",
            "   7,   8, 1.0",
            "   8,   9, 1.0",
            "   9,  10, 1.0",
            "  10,  13, 1.0",
            "  11,  10, 0.1384",
            "  11,  12, 0.200",
            "  11,  14, 0.6616",
            "  12,  13, 1.0",
            "  13,  16, 1.0",
            "  14,  15, 1.0",
            "  15,  16, 1.0",
            "  16,  17, 1.0",
            "0,0,0.0",
            "   2,   1, 1.0",
            "   4,   3, 1.0",
            "   6,   3, 1.0",
            "   7,   6, 1.0",
            "   8,   7, 1.0",
            "   9,   8, 1.0",
            "  10,   9, 1.0",
            "  11,   8, 1.0",
            "  12,  11, 1.0",
            "  14,  11, 1.0",
            "  15,  14, 1.0",
            "  16,  15, 1.0",
            "  17,  16, 1.0",
            "0,0,0.0",
            "    U 235 925   U 236 926  NP-237 937   U 238 928  NP 239 939  PU 248 948",
            "   PU 239 949  PU 240 940  PU 241 941  PU 242 942  AM 241 951  AM 942 950",
            "   AM 243 953  CM 242 962  CM 243 963  CM 244 964  CM 245 965  FP 854 854",
            "   FP 884 884  FP 894 894  FP 814 814   ",
            "0,0",
            "   1, 0.312700E-10",
            "  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  1.0,  0.0,  0.0,  0.0",
            "   2, 0.334000E-10",
            "  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  1.0,  0.0,  0.0,  0.0",
            "   3, 0.333600E-10",
            "  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  1.0,  0.0,  0.0",
            "   4, 0.350000E-10",
            "  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  1.0,  0.0,  0.0",
            "   5, 0.333100E-10",
            "  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  1.0,  0.0",
```

```
        "  6, 0.315500E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
        "  7, 0.322200E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
        "  8, 0.324400E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        "  9, 0.324800E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 10, 0.330100E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 11, 0.353800E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 12, 0.344900E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 13, 0.344300E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 14, 0.351000E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 15, 0.351600E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 16, 0.340100E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
        " 17, 0.352900E-10",
        " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0"
    ]
    inp = CardFile( card_text )

    # unit 8
    micro_file = SagepMicroFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
    micro_file.set_generate_code( "CONVMIC" )
    micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

    # Unit 11
    nd_file = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc01-BOC" )
    nd_file.set_generate_code( "REFUEL" )
    nd_file.set_filetype( "NUMBER_DENSITY_REGION" )

    # Unit 51
    database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )
    database.set_generate_code( "CITATION" )


    fire1.inputs([inp, micro_file, nd_file, database])
    fire1.run()
    list_file = fire1.list()

    ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/FIRE1_cyc01.dmp" )


    #pprint( ref_list.diff_from( list_file, filter="" ) )

    assert list_file.has_same_contents_with( ref_list )


    out_files = fire1.outfiles()
    assert len(out_files) == 6 , len(out_files)   # units: 12, 30, 31, 34, 35, 36

    # unit 30 (text)
    assert out_files[1].generate_code() == "FIRE1", out_files[0].generate_code()
    assert out_files[1].filetype() == "NUMBER_DENSITY_EACH", out_files[0].filetype()
    assert out_files[1].has_same_contents_with( RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT30/cyc01-BOC" ) )

    # unit 34 (text)
    assert out_files[3].generate_code() == "FIRE1", out_files[1].generate_code()
    assert out_files[3].filetype() == "KAPPA_SIGF_XS", out_files[1].filetype()
    assert out_files[3].has_same_contents_with( RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT34/cyc01-BOC" ) )

    # unit 35 (text)
    assert out_files[4].generate_code() == "FIRE1", out_files[2].generate_code()
    assert out_files[4].filetype() == "MATRIX_T", out_files[2].filetype()
    assert out_files[4].has_same_contents_with( RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT35/cyc01-BOC" ) )

    # unit 36 (text)
    assert out_files[5].generate_code() == "FIRE1", out_files[3].generate_code()
    assert out_files[5].filetype() == "MATRIX_R", out_files[3].filetype()
    assert out_files[5].has_same_contents_with( RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT36/cyc01-BOC" ) )


def testAdjoint(self):
    fire1 = Fire1()
```

```
card_text = [
    "AVFX 04E",
    "VOL  04E",
    "PWFTa04E",
    "PWFXa04E",
    "16,21,18,10,1,17,1,1",
    "3570.00E+06,0.0,0.0",
    "47347200,20",
    "   5, 3.4044E-06,  1,  7, 1.0",
    "   7, 9.1100E-13,  1,  1, 1.0",
    "   8, 3.3460E-12,  1,  2, 1.0",
    "   9, 1.5264E-09,  1, 11, 1.0",
    "  11, 5.0820E-11,  1,  3, 1.0",
    "  12, 1.5600E-10,  1, 10, 0.173",
    "  12, 1.5600E-10,  1, 14, 0.827",
    "  14, 4.9282E-08,  1,  6, 1.0",
    "  15, 7.5500E-10,  1,  7, 1.0",
    "  16, 1.2130E-09,  1,  8, 1.0",
    "   1,  2, 1.0",
    "   2,  3, 1.0",
    "   3,  6, 1.0",
    "   4,  5, 1.0",
    "   6,  7, 1.0",
    "   7,  8, 1.0",
    "   8,  9, 1.0",
    "   9, 10, 1.0",
    "  10, 13, 1.0",
    "  11, 10, 0.1384",
    "  11, 12, 0.200",
    "  11, 14, 0.6616",
    "  12, 13, 1.0",
    "  13, 16, 1.0",
    "  14, 15, 1.0",
    "  15, 16, 1.0",
    "  16, 17, 1.0",
    "0,0,0.0",
    "   2,  1, 1.0",
    "   4,  3, 1.0",
    "   6,  3, 1.0",
    "   7,  6, 1.0",
    "   8,  7, 1.0",
    "   9,  8, 1.0",
    "  10,  9, 1.0",
    "  11,  8, 1.0",
    "  12, 11, 1.0",
    "  14, 11, 1.0",
    "  15, 14, 1.0",
    "  16, 15, 1.0",
    "  17, 16, 1.0",
    "0,0,0.0",
    "    U 235 925   U 236 926  NP-237 937   U 238 928  NP 239 939  PU 248 948",
    "   PU 239 949  PU 240 940  PU 241 941  PU 242 942  AM 241 951  AM 942 950",
    "   AM 243 953  CM 242 962  CM 243 963  CM 244 964  CM 245 965  FP 854 854",
    "   FP 884 884  FP 894 894  FP 814 814",
    "0,0",
    "   1, 0.312700E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
    "   2, 0.334000E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
    "   3, 0.333600E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
    "   4, 0.350000E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
    "   5, 0.333100E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
    "   6, 0.315500E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
    "   7, 0.322200E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "   8, 0.324400E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "   9, 0.324800E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "  10, 0.330100E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "  11, 0.353800E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "  12, 0.344900E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "  13, 0.344300E-10",
    "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
    "  14, 0.351000E-10",
```

```
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 15, 0.351600E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 16, 0.340100E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 17, 0.352900E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0"
            ]

        inp = CardFile( card_text )

        # unit 8
        micro_file = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
        micro_file.set_generate_code( "CONVMIC" )
        micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

        # Unit 11
        nd_file_boc = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc04-BOC" )
        nd_file_boc.set_generate_code( "REFUEL" )
        nd_file_boc.set_filetype( "NUMBER_DENSITY_REGION" )
        nd_file_boc.set_casename( "Cy04-BOC" )

        # Unit 15
        nd_file_eoc = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT12/cyc04-EOC" )
        nd_file_eoc.set_generate_code( "REFUEL2" )
        nd_file_eoc.set_filetype( "NUMBER_DENSITY_REGION_ADJOINT" )
        nd_file_eoc.set_casename( "Cy04-EOC" )

        # Unit 51
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )
        database.set_generate_code( "CITATION" )
        #database.set_filetype( "DATA_BASE" )


        fire1.inputs([inp, micro_file, nd_file_boc, nd_file_eoc, database])
        fire1.run()
        list_file = fire1.list()

        ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/FIRE1_cyc04-adj.dmp" )
        assert list_file.has_same_contents_with( ref_list )


        out_files = fire1.outfiles()
        assert len(out_files) == 6 , len(out_files)   # units: 12, 30, 31, 34, 35, 36

        # unit 12 (text)
        assert out_files[0].generate_code() == "FIRE1", out_files[0].generate_code()
        assert out_files[0].filetype() == "NUMBER_DENSITY_ADJOINT_FINAL", out_files[0].filetype()
        assert out_files[0].has_same_contents_with( RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT12/cyc04-EOC" ) )

        # unit 31 (text)
        assert out_files[2].generate_code() == "FIRE1", out_files[0].generate_code()
        assert out_files[2].filetype() == "NUMBER_DENSITY_ADJOINT_EACH", out_files[0].filetype()
        assert out_files[2].has_same_contents_with( RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT31/cyc04-EOC" ) )



def suite():
    return unittest.makeSuite( Fire1Test, 'test' )
#     suite = unittest.TestSuite()
#     suite.addTest( Fire1Test( "testForward" ))
#     suite.addTest( Fire1Test( "testAdjoint" ))
#     return suite


if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-10 Fire2Test.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.ModuleFile import *

class Fire2Test(unittest.TestCase):

    def testRun(self):
        fire2 = Fire2()

        card_text = [
            "AVFX 01B",
            "VOL  01B ",
            "PWFT 01B",
            "16,21,18,10,0,17,1,1",
            "3570.00E+06,0.0,0.0",
            "47347200,20",
            "   5, 3.4044E-06,   1,   7, 1.0",
            "   7, 9.1100E-13,   1,   1, 1.0",
            "   8, 3.3460E-12,   1,   2, 1.0",
            "   9, 1.5263E-09,   1,  11, 1.0",
            "  11, 5.0820E-11,   1,   3, 1.0",
            "  12, 1.5600E-10,   1,  10, 0.173",
            "  12, 1.5600E-10,   1,  14, 0.827",
            "  14, 4.9282E-08,   1,   6, 1.0",
            "  15, 7.5500E-10,   1,   7, 1.0",
            "  16, 1.2130E-09,   1,   8, 1.0",
            "   1,  2, 1.0",
            "   2,  3, 1.0",
            "   3,  6, 1.0",
            "   4,  5, 1.0",
            "   6,  7, 1.0",
            "   7,  8, 1.0",
            "   8,  9, 1.0",
            "   9, 10, 1.0",
            "  10, 13, 1.0",
            "  11, 10, 0.1384",
            "  11, 12, 0.200",
            "  11, 14, 0.6616",
            "  12, 13, 1.0",
            "  13, 16, 1.0",
            "  14, 15, 1.0",
            "  15, 16, 1.0",
            "  16, 17, 1.0",
            "0,0,0.0",
            "   2,  1, 1.0",
            "   4,  3, 1.0",
            "   6,  3, 1.0",
            "   7,  6, 1.0",
            "   8,  7, 1.0",
            "   9,  8, 1.0",
            "  10,  9, 1.0",
            "  11,  8, 1.0",
            "  12, 11, 1.0",
            "  14, 11, 1.0",
            "  15, 14, 1.0",
            "  16, 15, 1.0",
            "  17, 16, 1.0",
            "0,0,0.0",
            "    U 235 925   U 236 926  NP-237 937   U 238 928  NP 239 939  PU 248 948",
            "   PU 239 949  PU 240 940  PU 241 941  PU 242 942  AM 241 951  AM 942 950",
            "   AM 243 953  CM 242 962  CM 243 963  CM 244 964  CM 245 965  FP 854 854",
            "   FP 884 884  FP 894 894  FP 814 814   ",
            "0,0",
            "   1, 0.312700E-10",
            "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
            "   2, 0.334000E-10",
            "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0",
            "   3, 0.333600E-10",
            "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
            "   4, 0.350000E-10",
            "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
            "   5, 0.333100E-10",
            "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
            "   6, 0.315500E-10",
            "  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0",
            "   7, 0.322200E-10",
```

```
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0",
            "  8, 0.324400E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            "  9, 0.324800E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 10, 0.330100E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 11, 0.353800E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 12, 0.344900E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 13, 0.344300E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 14, 0.351000E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 15, 0.351600E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 16, 0.340100E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0",
            " 17, 0.352900E-10",
            " 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0"
        ]
        inp = CardFile( card_text )

        # unit 8
        micro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
        micro_file.set_generate_code( "CONVMIC" )
        micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

        # Unit 11
        nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT13/cyc01-BOC" )
        nd_file.set_generate_code( "REFUEL" )
        nd_file.set_filetype( "NUMBER_DENSITY_ASSEMBLY" )

        # Unit 42
        ctl_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT11/cyc01-BOC" )
        ctl_file.set_generate_code( "REFUEL" )
        ctl_file.set_filetype( "REFUEL_CONTROL_DATA" )

        # Unit 51
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

        fire2.inputs([inp, micro_file, nd_file, database, ctl_file])
        fire2.run()
        list_file = fire2.list()

        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/FIRE2_cyc01.dmp" )
        assert list_file.has_same_contents_with( ref_list )


        out_files = fire2.outfiles()
        assert len(out_files) == 5 , len(out_files)   # units: 12, 30, 34, 35, 36

        # unit 30 (text)
        assert out_files[1].generate_code() == "FIRE2", out_files[0].generate_code()
        assert out_files[1].filetype() == "NUMBER_DENSITY_EACH", out_files[1].filetype()
        assert out_files[1].has_same_contents_with(DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire2.FT30/cyc01-BOC" ) )

        # unit 34 (text)
        assert out_files[2].generate_code() == "FIRE2", out_files[1].generate_code()
        assert out_files[2].filetype() == "KAPPA_SIGF_XS", out_files[1].filetype()
        assert out_files[2].has_same_contents_with(DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire2.FT34/cyc01-BOC" ) )

        # unit 35 (text)
        assert out_files[3].generate_code() == "FIRE2", out_files[2].generate_code()
        assert out_files[3].filetype() == "MATRIX_T", out_files[2].filetype()
        assert out_files[3].has_same_contents_with(DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire2.FT35/cyc01-BOC" ) )

        # unit 36 (text)
        assert out_files[4].generate_code() == "FIRE2", out_files[3].generate_code()
        assert out_files[4].filetype() == "MATRIX_R", out_files[3].filetype()
        assert out_files[4].has_same_contents_with(DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire2.FT36/cyc01-BOC" ) )


def suite():
    return unittest.makeSuite( Fire2Test, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-11 Nsini4Test.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.ModuleFile import *

class Nsini4Test(unittest.TestCase):

    def testRun(self):
        nsini4 = Nsini4()

        card_text = [
            "KEFF 04E",
            "BRFET04E",
            "BRFIS04E",
            "PSINI04E",
            "PFACT04E",
            "FISSPEC",
            " 21",
            " 925 926 937 928 939 948 949",
            " 940 941 942 951 950 953 962",
            " 963 964 965 854 884 894 814",
            "  0  1  0",
            "         68         62          1         18         21         13          1",
            "      2      3      3      3      2      4      4      3      2      5      4      4",
            "      2      3      2      2      4      4      4      5      3",
            "  9.5504 15.7176 16.3613 16.4636  5.9732 18.6429 17.9109 12.7851",
            "  4.7272 21.2517 17.8360 18.1027  4.6213 13.2016 10.1232  3.5586",
            " 18.9745 17.4992 17.3742 32.6015 17.2216",
            "      5      6      4      3      3      3      4      3      3      3      4    16",
            "      5",
            " 30.0000 30.0000 20.0000 15.0000 15.0000 15.0000 20.0000 15.0000",
            " 15.0000 15.0000 20.0000 100.000 30.0000",
            "      1",
            "   1.00",
            " 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
            "        17        17",
            " 925 0.31270E-10",
            " 926 0.33400E-10",
            " 937 0.33360E-10",
            " 928 0.35000E-10",
            " 939 0.33310E-10 3.4044E-06",
            " 948 0.31550E-10",
            " 949 0.32220E-10 9.1100E-13",
            " 940 0.32440E-10 3.3460E-12",
            " 941 0.32480E-10 1.5264E-09 ",
            " 942 0.33010E-10 ",
            " 951 0.35380E-10 5.0820E-11",
            " 950 0.34490E-10 1.5600E-10",
            " 953 0.34430E-10",
            " 962 0.35100E-10 4.9282E-08",
            " 963 0.35160E-10 7.5500E-10",
            " 964 0.34010E-10 1.2130E-09",
            " 965 0.35290E-10",
            " 854 0.0",
            " 884 0.0",
            " 894 0.0",
            " 814 0.0"
            ]
        inp = CardFile( card_text )

        # unit 2
        gflux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/sagep93.FT12/SGP-cyc04-EOC-direct" )
        gflux_adjoint.set_generate_code( "SAGEP" )
        gflux_adjoint.set_filetype( "GENERALIZED_FLUX_ADJOINT" )
```

```
        # unit 3
        flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-EOC" )
        flux_forward.set_generate_code( "CITATION" )
        flux_forward.set_filetype( "MESH_FLUX" )
        flux_forward.set_casename( "Cy04-EOC_FORWARD" )
        flux_forward.setProperty( "NX", 68 )
        flux_forward.setProperty( "NY", 62 )
        flux_forward.setProperty( "NZ", 1 )
        flux_forward.setProperty( "NG", 18 )

        # unit 10
        macro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc04-EOC" )
        macro_file.set_generate_code( "STARTUP" )
        macro_file.set_filetype( "MACRO_XS" )

        # unit 12
        micro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
        micro_file.set_generate_code( "CONVMIC" )
        micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

        # unit 16
        leakage_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT20/cyc04-EOC" )
        leakage_file.set_generate_code( "CITATION" )
        leakage_file.set_filetype( "LEAKAGE_TERM" )

        # Unit 51
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )


        nsini4.inputs([inp, gflux_adjoint, flux_forward, macro_file, micro_file, leakage_file, database])
        nsini4.run()
        list_file = nsini4.list()

        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/NSINI4-cyc04EOC.dmp" )
        assert list_file.has_same_contents_with( ref_list )


        out_files = nsini4.outfiles()
        assert len(out_files) == 1 , len(out_files)   # units: 20

        # unit 20 (text)
        assert out_files[0].generate_code() == "NSINI4", out_files[0].generate_code()
        assert out_files[0].filetype() == "NUMBER_DENSITY_ADJOINT_FINAL", out_files[0].filetype()
        assert            out_files[0].has_same_contents_with(            DataFile(            ref1_dir            +
"/H13-Na-MOX.C3/BR/WORK/nsini4.FT20/cyc04-EOC" ) )


def suite():
    return unittest.makeSuite( Nsini4Test, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-12 NsjumpTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.ModuleFile import *

class NsjumpTest(unittest.TestCase):

    def testRun(self):
        nsjump = Nsjump()

        card_text = [
            "KEFF 04B",
            "PWFTa04E",
            "PS   04B",
            "DNORM04B",
            "FISSPEC",
            " 1.00000E+24",
            " 21",
            " 925 926 937 928 939 948 949",
            " 940 941 942 951 950 953 962",
            " 963 964 965 854 884 894 814",
            "  0  1  0",
            "         68         62          1         18         21         13          1",
            "      2     3     3     3     2     4     4     3     2     5     4     4",
            "      2     3     2     2     4     4     4     5     3",
            "  9.5504 15.7176 16.3613 16.4636  5.9732 18.6429 17.9109 12.7851",
            "  4.7272 21.2517 17.8360 18.1027  4.6213 13.2016 10.1232  3.5586",
            " 18.9745 17.4992 17.3742 32.6015 17.2216",
            "      5     6     4     3     3     3     4     3     3     3     4    16",
            "      5",
            " 30.0000 30.0000 20.0000 15.0000 15.0000 15.0000 20.0000 15.0000",
            " 15.0000 15.0000 20.0000 100.000 30.0000",
            "      1",
            "   1.00",
            " 10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            " 10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            " 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
            "        17        17",
            " 925 0.31270E-10",
            " 926 0.33400E-10",
            " 937 0.33360E-10",
            " 928 0.35000E-10",
            " 939 0.33310E-10 ",
            " 948 0.31550E-10",
            " 949 0.32220E-10 ",
            " 940 0.32440E-10",
            " 941 0.32480E-10",
            " 942 0.33010E-10",
            " 951 0.35380E-10 ",
            " 950 0.34490E-10 ",
            " 953 0.34430E-10",
            " 962 0.35100E-10 ",
            " 963 0.35160E-10 ",
            " 964 0.34010E-10 ",
            " 965 0.35290E-10",
            " 854 0.0",
            " 884 0.0",
            " 894 0.0",
            " 814 0.0",
            "3570.00E+06"
            ]
        inp = CardFile( card_text )

        # unit 2
        gflux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/sagep93.FT12/SGP-cyc04-flx" )
        gflux_adjoint.set_generate_code( "SAGEP" )
        gflux_adjoint.set_filetype( "GENERALIZED_FLUX_ADJOINT" )
```

```
        # unit 3
        flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-BOC" )
        flux_forward.set_generate_code( "CITATION" )
        flux_forward.set_filetype( "MESH_FLUX" )
        flux_forward.set_casename( "Cy04-EOC_FORWARD" )
        flux_forward.setProperty( "NX", 68 )
        flux_forward.setProperty( "NY", 62 )
        flux_forward.setProperty( "NZ", 1 )
        flux_forward.setProperty( "NG", 18 )

        # unit 10
        macro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc04-BOC" )
        macro_file.set_generate_code( "STARTUP" )
        macro_file.set_filetype( "MACRO_XS" )

        # unit 11
        nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT12/cyc04-BOC" )
        nd_file.set_generate_code( "REFUEL2" )
        nd_file.set_filetype( "NUMBER_DENSITY_REGION_ADJOINT" )

        # unit 12
        micro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
        micro_file.set_generate_code( "CONVMIC" )
        micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

        # unit 16
        leakage_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT20/cyc04-BOC" )
        leakage_file.set_generate_code( "CITATION" )
        leakage_file.set_filetype( "LEAKAGE_TERM" )

        # Unit 51
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )


        nsjump.inputs([inp, gflux_adjoint, flux_forward, macro_file, nd_file, micro_file, leakage_file, database])
        nsjump.run()
        list_file = nsjump.list()

        ref_list = RefFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/NSJUMP-cyc04.dmp" )
        assert list_file.has_same_contents_with( ref_list )


        out_files = nsjump.outfiles()
        assert len(out_files) == 1 , len(out_files)   # units: 20

        # unit 20 (text)
        assert out_files[0].generate_code() == "NSJUMP", out_files[0].generate_code()
        assert out_files[0].filetype() == "NUMBER_DENSITY_ADJOINT_FINAL", out_files[0].filetype()
        assert                out_files[0].has_same_contents_with(            DataFile(            ref1_dir            +
"/H13-Na-MOX.C3/BR/WORK/nsjump.FT20/cyc04-BOC" ) )


def suite():
    return unittest.makeSuite( NsjumpTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-13 PDSEditTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.ModuleFile import *

class PDSEditTest(unittest.TestCase):

    def testRun(self):
        pdsedit = PDSEdit()

        card_text = [
            "16 1 0 0 2                                                                    ",
            " 0 18                                                                         ",
            " 0.0          0.0          0.0          0.0          0.0                     ",
            " 0.0          0.0          0.0          0.0          0.0                     ",
            " 0.0          0.0          0.0          0.0          0.0                     ",
            " 0.0          0.0          0.0          0.0          0.0                     ",
            "       0.01296      0.03111      0.13320      0.34764      1.43147",
            "       3.81048      0.0          0.0          0.0          0.0",
            "ICO1    SLAROM  ICO1  ",
            "ICO2    SLAROM  ICO2 ",
            "OTO1    SLAROM  OTO1 ",
            "OTO2    SLAROM  OTO2 ",
            "ABLU    SLAROM  ABLU ",
            "ABLL    SLAROM  ABLL ",
            "RBLU    SLAROM  RBLU ",
            "RBLM    SLAROM  RBLM ",
            "RBLL    SLAROM  RBLL ",
            "NAFO    SLAROM  NAFO ",
            "GPL1    SLAROM  GPL1 ",
            "GPL2    SLAROM  GPL2 ",
            "AXSU    SLAROM  AXSU ",
            "AXSL    SLAROM  AXSL  ",
            "RDS1    SLAROM  RDS1 ",
            "RDS2    SLAROM  RDS2 ",
            ]
        inp = CardFile( card_text )

        dummy_card = [
            " 33                                                                           ",
            "PU-239          949   1 239.05215     3.1800-11      1.0461-12      9.1581-13 ",
            "    0.0082E-2  0.0605E-2   0.0467E-2  0.0709E-2   0.0223E-2   0.0076E-2      ",
            "PU-240          940   2 240.05388     3.1543-11      8.3945-13      3.3404-12 ",
            "    0.0081E-2  0.0787E-2   0.0554E-2  0.1010E-2   0.0369E-2   0.0084E-2      ",
            "PU-241          941   3 241.05674     3.2088-11      1.0109-12      1.4950-09 ",
            "    0.0053E-2  0.1214E-2   0.0917E-2  0.2067E-2   0.0964E-2   0.0085E-2      ",
            "PU-242          942   4 242.05873     3.2040-11      8.3945-13      5.7994-14 ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "U-235           925   5 235.04527     3.0903-11      1.0477-12      3.0957-17 ",
            "    0.0254E-2  0.1423E-2   0.1256E-2  0.2719E-2   0.0855E-2   0.0174E-2      ",
            "U-236           926   6 236.04607     3.0662-11      8.7469-13      9.1965-16 ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "U-238           928   7 238.04951     3.1063-11      9.1154-13      4.8735-18 ",
            "    0.0206E-2  0.2169E-2   0.2564E-2  0.6142E-2   0.3562E-2   0.1187E-2      ",
            "FP-U235         854   8 235.0         0.0            1.2976-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "FP-PU239        894   9 239.0         0.0            1.2976-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "O-16              8  10  15.99950     0.0            5.4148-13      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "NA               11  11  22.98979     0.0            2.0025-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "CR               24  12  51.99763     0.0            1.4883-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "NI               28  13  58.72843     0.0            1.3665-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "MO               42  14  95.88961     0.0            1.2928-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "FE               26  15  55.84731     0.0            1.2496-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "MN               25  16  54.94000     0.0            8.5841-13      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "B-10            105  17  10.01610     0.0            4.4701-13      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "B-11            115  18  11.00930     0.0            1.3827-12      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
            "C                 6  19  12.01100     0.0            9.7255-13      0.0       ",
            "    0.0        0.0         0.0        0.0         0.0         0.0            ",
```

```
    "C                       6  20  12.01100      0.0            9.7255-13      0.0          ",
    "   0.0          0.0             0.0        0.0         0.0         0.0                  ",
    "C                       6  19  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  20  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  19  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  20  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  19  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  20  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  19  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  20  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  19  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  20  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  19  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  20  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0",
    "C                       6  19  12.01100      0.0            9.7255-13      0.0",
    "   0.0          0.0             0.0        0.0         0.0         0.0"
    ]
    dummy_inp = CardFile( dummy_card )
    dummy_inp.set_filetype( "DUMMY" )

    nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc04-EOC" )
    nd_file.set_generate_code( "REFUEL" )
    nd_file.set_filetype( "NUMBER_DENSITY_REGION" )

    macro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/startup.FT03/cyc04-EOC" )
    macro_file.set_generate_code( "STARTUP" )
    macro_file.set_filetype( "MACRO_XS" )

    pdsedit.set_PDSin( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/mac18g" )
    pdsedit.inputs([inp, dummy_inp, macro_file, nd_file])
    pdsedit.run( "cy04-EOC" )
    list_file = pdsedit.list()

    ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/PDSEDT-DATA.cyc04-EOC.dmp" )
    assert list_file.has_same_contents_with( ref_list )


def suite():
    return unittest.makeSuite( PDSEditTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-14 Refuel2Test.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.ModuleFile import *

class Refuel2Test(unittest.TestCase):

    def testRun(self):
        refuel2 = Refuel2()

        card_text = [
            "     1    0",
            "    16    0",
            "     1    1    1    1    1    1    1    1    1    1    1    1    1    1           14",
            "     1    1    ",
            "    21",
            "   925  926  937  928  939  948  949  940  941  942  951  950  953  962",
            "   963  964  965  854  884  894  814",
            "     1  4.0                                                                      ",
            "     2  4.0                                                                      ",
            "     3  4.0                                                                      ",
            "     4  4.0                                                                      ",
            "     5  4.0                                                                      ",
            "     6  4.0                                                                      ",
            "     7  4.0                                                                      ",
            "     8  4.0                                                                      ",
            "     9  4.0                                                                      ",
            "    10  1.0                                                                      ",
            "    11  1.0                                                                      ",
            "    12  1.0                                                                      ",
            "    13  1.0                                                                      ",
            "    14  1.0                                                                      ",
            "    15  1.0                                                                      ",
            "    16  1.0                                                                      ",
            "     1    1  0.0",
            "     2    1  0.0",
            "     3    1  0.0",
            "     4    1  0.0",
            "     5    1  0.0",
            "     6    1  0.0",
            "     7    1  0.0",
            "     8    1  0.0",
            "     9    1  0.0",
            "    10    1  0.0",
            "    11    1  0.0",
            "    12    1  0.0",
            "    13    1  0.0",
            "    14    1  0.0",
            "    15    1  0.0",
            "    16    1  0.0"
        ]
        inp = CardFile( card_text )

        # unit 21
        nd_init = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/nsini4.FT20/cyc04-EOC" )
        nd_init.set_generate_code( "NSINI4" )
        nd_init.set_filetype( "NUMBER_DENSITY_ADJOINT_FINAL" )

        refuel2.inputs( [inp, nd_init] )
        refuel2.run( "cyc04-EOC" )
        list_file = refuel2.list()

        out_files = refuel2.outfiles()   # unit 11, 12
        assert len(out_files) == 2

        # unit 11
        assert out_files[0].filetype() == "REFUEL2_CONTROL_DATA"
        assert out_files[0].generate_code() == "REFUEL2"
        ref_unit_11 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT11/cyc04-EOC" )
        assert out_files[0].has_same_contents_with( ref_unit_11 )

        # unit 12
        assert out_files[1].filetype() == "NUMBER_DENSITY_REGION_ADJOINT"
        assert out_files[1].generate_code() == "REFUEL2"
        ref_unit_12 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT12/cyc04-EOC" )
        assert out_files[1].has_same_contents_with( ref_unit_12 )
```

```
    def testRunx(self):
        refuel2 = Refuel2()

        card_text = [
            "     1    1"
        ]
        inp = CardFile( card_text )

        # unit 21
        nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT12/cyc04-EOC" )
        nd_file.set_generate_code( "FIRE1" )
        nd_file.set_filetype( "NUMBER_DENSITY_ADJOINT_FINAL" )

        # unit 10
        ctrl_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT11/cyc04-EOC" )
        ctrl_file.set_generate_code( "REFUEL2" )
        ctrl_file.set_filetype( "REFUEL2_CONTROL_DATA" )


        refuel2.inputs( [inp, nd_file, ctrl_file] )
        refuel2.run( "cyc04-BOC" )
        list_file = refuel2.list()

        out_files = refuel2.outfiles()   # unit 11, 12
        assert len(out_files) == 2

        # unit 11
        assert out_files[0].filetype() == "REFUEL2_CONTROL_DATA"
        assert out_files[0].generate_code() == "REFUEL2"
        ref_unit_11 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT11/cyc04-BOC" )
        assert out_files[0].has_same_contents_with( ref_unit_11 )

        # unit 12
        assert out_files[1].filetype() == "NUMBER_DENSITY_REGION_ADJOINT"
        assert out_files[1].generate_code() == "REFUEL2"
        ref_unit_12 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel2.FT12/cyc04-BOC" )
        assert out_files[1].has_same_contents_with( ref_unit_12 )


def suite():
    return unittest.makeSuite( Refuel2Test, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-15 RefuelTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.ModuleFile import *

class RefuelTest(unittest.TestCase):

    def testRun(self):
        refuel = Refuel()

        card_text = [
            "    1    0                                                                      ",
            "   16    0                                                                      ",
            "    4    4    4    4    4    4    4    4    4    1    1    1    1    1          14",
            "    1    1    ",
            "   33",
            "  925  926  937  928  939  948  949  940  941  942  951  950  953  962",
            "  963  964  965  854  884  894  814    8   11   24   25   26   28   42",
            "  603   74  105  115    6",
            "    1  1.0  1.0  1.0  1.0                                                       ",
            "    2  1.0  1.0  1.0  1.0                                                       ",
            "    3  1.0  1.0  1.0  1.0                                                       ",
            "    4  1.0  1.0  1.0  1.0                                                       ",
            "    5  1.0  1.0  1.0  1.0                                                       ",
            "    6  1.0  1.0  1.0  1.0                                                       ",
            "    7  1.0  1.0  1.0  1.0                                                       ",
            "    8  1.0  1.0  1.0  1.0                                                       ",
            "    9  1.0  1.0  1.0  1.0                                                       ",
            "   10  1.0  ",
            "   11  1.0  ",
            "   12  1.0  ",
            "   13  1.0  ",
            "   14  1.0  ",
            "   15  1.0  ",
            "   16  1.0  ",
            "   16                                                                           ",
            "    1",
            "  925 1.99109E-05   926 0.00000E+00   937 9.45050E-06   928 6.53346E-03",
            "  939 0.00000E+00   948 2.06825E-05   949 1.01294E-03   940 5.98514E-04",
            "  941 7.98413E-05   942 7.21147E-05   951 3.71706E-05   950 0.00000E+00",
            "  953 1.84334E-05   962 0.00000E+00   963 0.00000E+00   964 1.83577E-05",
            "  965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "  814 0.00000E+00     8 1.69491E-02    11 7.32332E-03    24 2.42060E-03",
            "   25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
            "  603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
            "    6 0.00000E+00",
            "    2",
            "  925 1.99109E-05   926 0.00000E+00   937 9.45050E-06   928 6.53346E-03",
            "  939 0.00000E+00   948 2.06825E-05   949 1.01294E-03   940 5.98514E-04",
            "  941 7.98413E-05   942 7.21147E-05   951 3.71706E-05   950 0.00000E+00",
            "  953 1.84334E-05   962 0.00000E+00   963 0.00000E+00   964 1.83577E-05",
            "  965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "  814 0.00000E+00     8 1.69491E-02    11 7.32332E-03    24 2.42060E-03",
            "   25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
            "  603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
            "    6 0.00000E+00",
            "    3",
            "  925 1.92901E-05   926 0.00000E+00   937 1.05223E-05   928 6.32977E-03",
            "  939 0.00000E+00   948 2.30282E-05   949 1.12782E-03   940 6.66396E-04",
            "  941 8.88966E-05   942 8.02937E-05   951 4.13863E-05   950 0.00000E+00",
            "  953 2.05241E-05   962 0.00000E+00   963 0.00000E+00   964 2.04398E-05",
            "  965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "  814 0.00000E+00     8 1.69639E-02    11 7.32332E-03    24 2.42060E-03",
            "   25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
            "  603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
            "    6 0.00000E+00",
            "    4",
            "  925 1.92901E-05   926 0.00000E+00   937 1.05223E-05   928 6.32977E-03",
            "  939 0.00000E+00   948 2.30282E-05   949 1.12782E-03   940 6.66396E-04",
            "  941 8.88966E-05   942 8.02937E-05   951 4.13863E-05   950 0.00000E+00",
            "  953 2.05241E-05   962 0.00000E+00   963 0.00000E+00   964 2.04398E-05",
            "  965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "  814 0.00000E+00     8 1.69639E-02    11 7.32332E-03    24 2.42060E-03",
            "   25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
            "  603 2.75794E-04    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
            "    6 0.00000E+00",
            "    5",
            "  925 2.46280E-05   926 0.00000E+00   937 0.00000E+00   928 8.08131E-03",
```

```
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 1.62119E-02    11 7.32332E-03    24 2.42060E-03",
"    25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
"   603 0.00000E+00    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    6",
"   925 2.46280E-05   926 0.00000E+00   937 0.00000E+00   928 8.08131E-03",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 1.62119E-02    11 7.32332E-03    24 2.42060E-03",
"    25 1.24962E-04    26 1.75173E-02    28 7.79823E-05    42 5.96308E-05",
"   603 0.00000E+00    74 1.24471E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    7",
"   925 3.33890E-05   926 0.00000E+00   937 0.00000E+00   928 1.09561E-02",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 2.19790E-02    11 6.61882E-03    24 2.01073E-03",
"    25 1.03803E-04    26 1.45512E-02    28 6.47780E-05    42 4.95338E-05",
"   603 0.00000E+00    74 1.03395E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    8",
"   925 3.33890E-05   926 0.00000E+00   937 0.00000E+00   928 1.09561E-02",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 2.19790E-02    11 6.61882E-03    24 2.01073E-03",
"    25 1.03803E-04    26 1.45512E-02    28 6.47780E-05    42 4.95338E-05",
"   603 0.00000E+00    74 1.03395E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"    9",
"   925 3.33890E-05   926 0.00000E+00   937 0.00000E+00   928 1.09561E-02",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 2.19790E-02    11 6.61882E-03    24 2.01073E-03",
"    25 1.03803E-04    26 1.45512E-02    28 6.47780E-05    42 4.95338E-05",
"   603 0.00000E+00    74 1.03395E-04   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"   10",
"   925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 0.00000E+00    11 2.03057E-02    24 1.21654E-03",
"    25 1.34329E-04    26 4.58722E-03    28 1.43655E-03    42 1.09920E-04",
"   603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"   11",
"   925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 0.00000E+00    11 7.22756E-03    24 3.18543E-03",
"    25 3.51733E-04    26 1.20114E-02    28 3.76282E-03    42 2.87732E-04",
"   603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"   12",
"   925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
"   953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
"   965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
"   814 0.00000E+00     8 0.00000E+00    11 6.85516E-03    24 2.51265E-03",
"    25 2.77445E-04    26 9.47455E-03    28 2.96810E-03    42 2.26961E-04",
"   603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
"     6 0.00000E+00",
"   13",
"   925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
"   939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
"   941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
```

```
            "    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
            "    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "    814 0.00000E+00     8 0.00000E+00    11 7.22164E-03    24 2.28020E-03",
            "     25 2.51777E-04    26 8.59800E-03    28 2.69258E-03    42 2.06026E-04",
            "    603 0.00000E+00    74 0.00000E+00   105 1.05361E-02   115 4.21443E-02",
            "      6 1.31701E-02",
            "    14",
            "    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
            "    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
            "    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
            "    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
            "    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "    814 0.00000E+00     8 0.00000E+00    11 5.55674E-03    24 1.04735E-02",
            "     25 1.15647E-03    26 3.94927E-02    28 1.23677E-02    42 9.46329E-04",
            "    603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
            "      6 0.00000E+00",
            "    15",
            "    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
            "    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
            "    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
            "    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
            "    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "    814 0.00000E+00     8 0.00000E+00    11 6.38300E-03    24 9.95491E-03",
            "     25 1.09921E-03    26 3.75373E-02    28 1.17553E-02    42 8.99472E-04",
            "    603 0.00000E+00    74 0.00000E+00   105 0.00000E+00   115 0.00000E+00",
            "      6 0.00000E+00",
            "    16",
            "    925 0.00000E+00   926 0.00000E+00   937 0.00000E+00   928 0.00000E+00",
            "    939 0.00000E+00   948 0.00000E+00   949 0.00000E+00   940 0.00000E+00",
            "    941 0.00000E+00   942 0.00000E+00   951 0.00000E+00   950 0.00000E+00",
            "    953 0.00000E+00   962 0.00000E+00   963 0.00000E+00   964 0.00000E+00",
            "    965 0.00000E+00   854 0.00000E+00   884 0.00000E+00   894 0.00000E+00",
            "    814 0.00000E+00     8 0.00000E+00    11 4.28467E-03    24 2.89636E-03",
            "     25 3.19813E-04    26 1.09214E-02    28 3.42018E-03    42 2.61699E-04",
            "    603 0.00000E+00    74 0.00000E+00   105 1.22001E-02   115 4.88005E-02",
            "      6 1.52502E-02",
            "      0                                                                  "
        ]
    inp = CardFile( card_text )

    refuel.inputs(inp)
    refuel.run()
    list_file = refuel.list()

    out_files = refuel.outfiles()    # unit 11, 12, 13, 14
    assert len(out_files) == 4
    assert out_files[0].filetype() == "REFUEL_CONTROL_DATA"
    assert out_files[0].generate_code() == "REFUEL"

    ref_dat = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT11/cyc01-BOC" )
    assert out_files[0].has_same_contents_with( ref_dat )


def suite():
    return unittest.makeSuite( RefuelTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-16 SagepTest.py

```
import unittest
from  sagep.config import *
import os

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.ModuleFile import *

class SagepTest(unittest.TestCase):

    def setUp(self):
        self.footer_text = [
            "FISSION SP            8              -1.00",
            "FISSION SP            8              -0.90",
            "FISSION SP            8              -0.80",
            "FISSION SP            8              -0.70",
            "FISSION SP            8              -0.60",
            "FISSION SP            8              -0.50",
            "FISSION SP            8              -0.40",
            "FISSION SP            8              -0.30",
            "FISSION SP            8              -0.20",
            "FISSION SP            8              -0.10",
            "FISSION SP            8              -0.05",
            "FISSION SP            8               0.05",
            "FISSION SP            8               0.10",
            "FISSION SP            8               0.20",
            "FISSION SP            8               0.30",
            "FISSION SP            8               0.40",
            "FISSION SP            8               0.50",
            "FISSION SP            8               0.60",
            "FISSION SP            8               0.70",
            "FISSION SP            8               0.80",
            "FISSION SP            8               0.90",
            "FISSION SP            8               1.00",
            " U-235        925     1",
            " U-235        925     2",
            " U-235        925     4",
            " U-235        925     6",
            " U-235        925     7",
            " U-235        925    10",
            " U-235        925     9",
            " U-236        926     1",
            " U-236        926     2",
            " U-236        926     4",
            " U-236        926     6",
            " U-236        926     7",
            " U-236        926    10",
            " U-236        926     9",
            " U-237        927     1",
            " U-237        927     2",
            " U-237        927     4",
            " U-237        927     6",
            " U-237        927     7",
            " U-237        927    10",
            " U-237        927     9",
            " U-238        928     1",
            " U-238        928     2",
            " U-238        928     4",
            " U-238        928     6",
            " U-238        928     7",
            " U-238        928    10",
            " U-238        928     9",
            "PU-238        948     1",
            "PU-238        948     2",
            "PU-238        948     4",
            "PU-238        948     6",
            "PU-238        948     7",
            "PU-238        948    10",
            "PU-238        948     9",
            "PU-239        949     1",
            "PU-239        949     2",
            "PU-239        949     4",
            "PU-239        949     6",
            "PU-239        949     7",
            "PU-239        949    10",
            "PU-239        949     9",
            "PU-240        940     1",
            "PU-240        940     2",
            "PU-240        940     4",
```

```
"PU-240        940     6",
"PU-240        940     7",
"PU-240        940    10",
"PU-240        940     9",
"PU-241        941     1",
"PU-241        941     2",
"PU-241        941     4",
"PU-241        941     6",
"PU-241        941     7",
"PU-241        941    10",
"PU-241        941     9",
"PU-242        942     1",
"PU-242        942     2",
"PU-242        942     4",
"PU-242        942     6",
"PU-242        942     7",
"PU-242        942    10",
"PU-242        942     9",
"AM-241        951     1",
"AM-241        951     2",
"AM-241        951     4",
"AM-241        951     6",
"AM-241        951     7",
"AM-241        951    10",
"AM-241        951     9",
"AM-242M       950     1",
"AM-242M       950     2",
"AM-242M       950     4",
"AM-242M       950     6",
"AM-242M       950     7",
"AM-242M       950    10",
"AM-242M       950     9",
"AM-243        953     1",
"AM-243        953     2",
"AM-243        953     4",
"AM-243        953     6",
"AM-243        953     7",
"AM-243        953    10",
"AM-243        953     9",
"NP-237        937     1                                    00800007",
"NP-237        937     2                                    00810007",
"NP-237        937     4                                    00820007",
"NP-237        937     6                                    00830007",
"NP-237        937     7                                    00820007",
"NP-237        937     9                                    00830007",
"NP-237        937    10                                    00830007",
"NP-239        939     1                                    00800007",
"NP-239        939     2                                    00810007",
"NP-239        939     4                                    00820007",
"NP-239        939     6                                    00830007",
"NP-239        939     7                                    00820007",
"NP-239        939     9                                    00830007",
"NP-239        939    10 ",
"CM-242        962     1                                    00720007",
"CM-242        962     2                                    00730007",
"CM-242        962     4                                    00740007",
"CM-242        962     6                                    00750007",
"CM-242        962     7                                    00730007",
"CM-242        962     9                                    00750007",
"CM-242        962    10                                    00740007",
"CM-243        963     1                                    00720007",
"CM-243        963     2                                    00730007",
"CM-243        963     4                                    00740007",
"CM-243        963     6                                    00750007",
"CM-243        963     7                                    00730007",
"CM-243        963     9                                    00750007",
"CM-243        963    10                                    00740007",
"CM-244        964     1                                    00720007",
"CM-244        964     2                                    00730007",
"CM-244        964     4                                    00740007",
"CM-244        964     6                                    00750007",
"CM-244        964     7                                    00730007",
"CM-244        964     9                                    00750007",
"CM-244        964    10    ",
"O               8     1",
"O               8     6",
"O               8     7",
"O               8    10",
"O               8     9",
"NA             11     1",
"NA             11     6",
"NA             11     7",
```

```
"NA          11    10",
"NA          11     9",
"CR          24     1",
"CR          24     6",
"CR          24     7",
"CR          24    10",
"CR          24     9",
"MN-55       25     1                                    00860007",
"MN-55       25     6                                    00860007",
"MN-55       25     7                                    00860007",
"MN-55       25    10                                    00860007",
"MN-55       25     9   ",
"FE          26     1",
"FE          26     6",
"FE          26     7",
"FE          26    10",
"FE          26     9",
"NI          28     1",
"NI          28     6",
"NI          28     7",
"NI          28    10",
"NI          28     9",
"ZR          40     1",
"ZR          40     6",
"ZR          40     7",
"ZR          40    10",
"ZR          40     9",
"MO-NAT      42     1                                    00860007",
"MO-NAT      42     6                                    00860007",
"MO-NAT      42     7                                    00860007",
"MO-NAT      42    10                                    00860007",
"MO-NAT      42     9   ",
"ND-143     603     1",
"ND-143     603     6",
"ND-143     603     7",
"ND-143     603    10",
"ND-143     603     9",
"W           74     1                                    00860007",
"W           74     6                                    00860007",
"W           74     7                                    00860007",
"W           74    10                                    00860007",
"W           74     9",
"PB          82     1",
"PB          82     6",
"PB          82     7",
"PB          82    10",
"PB          82     9",
" B- 10     105     1",
" B- 10     105     6",
" B- 10     105     7",
" B- 10     105    10",
" B- 10     105     9",
" B- 11     115     1",
" B- 11     115     6",
" B- 11     115     7",
" B- 11     115    10",
" B- 11     115     9",
"C             6     1",
"C             6     6",
"C             6     7",
"C             6    10",
"C             6     9",
"PU-241FP    814     1",
"PU-241FP    814     6",
"PU-241FP    814     7",
"PU-241FP    814    10",
"PU-241FP    814     9",
"U-235FP     854     1",
"U-235FP     854     6",
"U-235FP     854     7",
"U-235FP     854    10",
"U-235FP     854     9",
"U-238FP     884     1",
"U-238FP     884     6",
"U-238FP     884     7",
"U-238FP     884    10",
"U-238FP     884     9",
"PU-239FP    894     1",
"PU-239FP    894     6",
"PU-239FP    894     7",
"PU-239FP    894    10",
"PU-239FP    894     9"
```

```
        ]

    def testDirect(self):
        sagep = Sagep()

        card_text = [
            "FISSPEC",
            "KEFF 04E",
            "SFACT  ",
            "PFACT04E",
            "BRFET04E",
            "BRFIS04E",
            "BRFI204E",
            "PSINI04E",
            "SGPFX04E",
            "H14 NA-MOX Breeding Ratio  cyc04 EOC DIRECT                              ",
            "   0   1   0   0 18 16  0",
            "   1   0   0   0",
            "004",
            "   2  9.55043   3 15.71764   3 16.36130   3 16.46364   2  5.97324   4 18.64292",
            "   4 17.91088   3 12.78505   2  4.72722   5 21.25165   4 17.83599   4 18.10268",
            "   2  4.62130   3 13.20161   2 10.12392   2  3.55860   4 18.97454   4 17.49924",
            "   4 17.37424   5 32.60152   3 17.22164",
            "   5 30.00000   6 30.00000   4 20.00000   3 15.00000   3 15.00000   3 15.00000",
            "   4 20.00000   3 15.00000   3 15.00000   3 15.00000   4 20.00000 16 100.0000",
            "   5 30.00000",
            "005",
            "  10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
            "  10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            "  10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            "  10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
            "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
            "  10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            "  10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
            "  10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
            "  14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
            "   1",
            "ICO1       SLAROM",
            "   2",
            "ICO2       SLAROM",
            "   3",
            "OTO1       SLAROM",
            "   4",
            "OTO2       SLAROM",
            "   5",
            "ABLU       SLAROM",
            "   6",
            "ABLL       SLAROM",
            "   7",
            "RBLU       SLAROM",
            "   8",
            "RBLM       SLAROM",
            "   9",
            "RBLL       SLAROM",
            "  10",
            "NAFO       SLAROM",
            "  11",
            "GPL1       SLAROM",
            "  12",
            "GPL2       SLAROM",
            "  13",
            "AXSU       SLAROM",
            "  14",
            "AXSL       SLAROM",
            "  15",
            "RDS1       SLAROM",
            "  16",
            "RDS2       SLAROM",
            "H13 NA-MOX Breeding Ratio  cyc04 EOC DIRECT ",
            "      4      0      0",
            "    3",
            "  928 0.00000E+00 0.35000E-10",
            "  940 3.34600E-12 0.32440E-10",
            "  948 0.00000E+00 0.31550E-10",
            "    2",
            "  949 9.11000E-13 0.32220E-10",
            "  941 1.52640E-09 0.32480E-10",
```

```
        "   9",
        " 925 0.00000E+00 0.31270E-10",
        " 926 0.00000E+00 0.33400E-10",
        " 942 0.00000E+00 0.33010E-10",
        " 951 5.08200E-11 0.35380E-10",
        " 950 1.56000E-10 0.34490E-10",
        " 953 0.00000E+00 0.34430E-10",
        " 937 0.00000E+00 0.33360E-10",
        " 939 3.40440E-06 0.33310E-10",
        " 962 4.92820E-08 0.35100E-10",
        " 3570.00E+06",
        " 1.0000E-4 5.0000E-2 ",
        "  1000   200",
        "  2000"
        ]

    inp = CardFile( card_text + self.footer_text )

    org_inp1 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SGP-cyc04-EOC-direct")
    org_inp2 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SAGEP-snslev")
    org_inp = DataFile()
    command = "cat %s %s > %s" % ( org_inp1.pathname(), org_inp2.pathname(), org_inp.pathname() )
    print command
    os.system( command )

    assert inp.has_same_contents_with( org_inp )


    flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-EOC" )
    flux_forward.set_generate_code( "CITATION" )
    flux_forward.set_filetype( "MESH_FLUX" )
    flux_forward.set_casename( "Cy04-EOC_FORWARD" )
    flux_forward.setProperty( "NX", 68 )
    flux_forward.setProperty( "NY", 62 )
    flux_forward.setProperty( "NZ", 1 )
    flux_forward.setProperty( "NG", 18 )

    flux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-EOC-adj" )
    flux_adjoint.set_generate_code( "CITATION" )
    flux_adjoint.set_filetype( "MESH_FLUX" )
    flux_adjoint.set_casename( "Cy04-EOC_ADJOINT" )
    flux_adjoint.setProperty( "NX", 68 )
    flux_adjoint.setProperty( "NY", 62 )
    flux_adjoint.setProperty( "NZ", 1 )
    flux_adjoint.setProperty( "NG", 18 )

    nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc04-EOC" )
    nd_file.set_generate_code( "REFUEL" )
    nd_file.set_filetype( "NUMBER_DENSITY_REGION" )

    database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

    sagep.set_PDSin( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/mic18g-cyc04-EOC" )
    sagep.inputs([inp,  nd_file, database, flux_forward, flux_adjoint])
    sagep.run( "cy04-EOC" )
    list_file = sagep.list()

    ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SGP-cyc04-EOC-direct.dmp" )
    assert list_file.has_same_contents_with( ref_list )


def testFlux(self):
    sagep = Sagep()

    card_text = [
        "FISSPEC ",
        "KEFF 04B",
        "SFACT04B ",
        "PFACT04B",
        "BRFET04B",
        "BRFIS04B",
        "BRFI204B",
        "PSINI04B",
        "SGPFX04B",
        "H14 NA-MOX Breeding Ratio  cyc04 flux                                    ",
        "  0  1  0  0 18 16  0",
        "  1  0  0  0",
        "004",
        "  2  9.55043  3 15.71764  3 16.36130  3 16.46364  2  5.97324  4 18.64292",
        "  4 17.91088  3 12.78505  2  4.72722  5 21.25165  4 17.83599  4 18.10268",
        "  2  4.62130  3 13.20161  2 10.12392  2  3.55860  4 18.97454  4 17.49924",
        "  4 17.37424  5 32.60152  3 17.22164",
```

```
    "   5 30.00000   6 30.00000   4 20.00000   3 15.00000   3 15.00000   3 15.00000",
    "   4 20.00000   3 15.00000   3 15.00000   3 15.00000   4 20.00000 16 100.0000",
    "   5 30.00000",
    "005",
    "  10 13 13 13 10 13 13 13 10 13 13 13 10 13 13 10 13 13 13 15 16",
    "  10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
    "  10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
    "  10  5  5  5 10  5  5  5 10  5  5  5 10  5  5 10  5  5  7 15 16",
    "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    "  10  1  1  1 10  1  1  1 10  2  2  2 10  3  3 10  4  4  8 15 16",
    "  10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
    "  10  6  6  6 10  6  6  6 10  6  6  6 10  6  6 10  6  6  9 15 16",
    "  10 11 11 11 10 11 11 11 10 11 11 11 10 11 11 10 11 11 12 15 16",
    "  14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 16",
    "   1",
    "ICO1       SLAROM",
    "   2",
    "ICO2       SLAROM",
    "   3",
    "OTO1       SLAROM",
    "   4",
    "OTO2       SLAROM",
    "   5",
    "ABLU       SLAROM",
    "   6",
    "ABLL       SLAROM",
    "   7",
    "RBLU       SLAROM",
    "   8",
    "RBLM       SLAROM",
    "   9",
    "RBLL       SLAROM",
    "  10",
    "NAFO       SLAROM",
    "  11",
    "GPL1       SLAROM",
    "  12",
    "GPL2       SLAROM",
    "  13",
    "AXSU       SLAROM",
    "  14",
    "AXSL       SLAROM",
    "  15",
    "RDS1       SLAROM",
    "  16",
    "RDS2       SLAROM",
    "H13 NA-MOX Breeding Ratio  cyc04 EOC DIRECT ",
    "     3      0      0",
    "    1.0E-04   5.0E-02",
    "  99999 99999",
    "   2000"
    ]

inp = CardFile( card_text + self.footer_text )

org_inp1 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SGP-cyc04-fix")
org_inp2 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SAGEP-snslev")
org_inp = DataFile()
command = "cat %s %s > %s" % ( org_inp1.pathname(), org_inp2.pathname(), org_inp.pathname() )
print command
os.system( command )

assert inp.has_same_contents_with( org_inp )


flux_forward = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-BOC" )
flux_forward.set_generate_code( "CITATION" )
flux_forward.set_filetype( "MESH_FLUX" )
flux_forward.set_casename( "Cy04-BOC_FORWARD" )
flux_forward.setProperty( "NX", 68 )
flux_forward.setProperty( "NY", 62 )
flux_forward.setProperty( "NZ", 1 )
flux_forward.setProperty( "NG", 18 )

flux_adjoint = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/citation.FT02/cyc04-BOC-adj" )
flux_adjoint.set_generate_code( "CITATION" )
flux_adjoint.set_filetype( "MESH_FLUX" )
flux_adjoint.set_casename( "Cy04-BOC_ADJOINT" )
flux_adjoint.setProperty( "NX", 68 )
```

```
        flux_adjoint.setProperty( "NY", 62 )
        flux_adjoint.setProperty( "NZ", 1 )
        flux_adjoint.setProperty( "NG", 18 )

        source = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/scgive.FT42/SCG-cyc04" )
        source.set_generate_code( "SCGIVE" )
        source.set_filetype( "SOURCE" )

        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

        sagep.set_PDSin( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/mic18g-cyc04-BOC" )
        sagep.inputs([inp,  source, database, flux_forward, flux_adjoint])
        sagep.run( "cy04-BOC" )
        list_file = sagep.list()

        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SGP-cyc04-flx.dmp" )
        assert list_file.has_same_contents_with( ref_list )


def suite():
    return unittest.makeSuite( SagepTest, 'test' )
#     suite = unittest.TestSuite()
#     suite.addTest( SagepTest( "testDirect" ))
#     suite.addTest( SagepTest( "testFlux" ))
#     return suite

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-17 ScgivebrTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.DataBase import *
from sagep.File.ModuleFile import *

class ScgivebrTest(unittest.TestCase):

    def testRun(self):
        scgivebr = Scgivebr()

        card_text = [
            "PWFTa04E",
            "PWFXa04E",
            "VOL  04B",
            "SFACT04B ",
            "PS   04B",
            "DNORM04B",
            "SCG  04B",
            "0,0,0,0,1.0,1.0",
            "16,21,18",
            "1",
            "47347200,20,3570.00E+06",
            "1",
            "21,1",
            "925,1",
            "1,1,1.0,  1,2,1.0,  1,18,1.0,  0,0,1.0",
            "926,1",
            "2,2,1.0,  2,3,1.0,  2,18,1.0,  2,1,1.0",
            "937,1",
            "3,3,1.0,  3,6,1.0,  3,19,1.0,  0,0,1.0",
            "928,1",
            "4,4,1.0,  4,5,1.0,  4,19,1.0,  4,3,1.0",
            "939,1",
            "5,5,1.0,  0,0,1.0,  5,20,1.0,  0,0,1.0",
            "948,1",
            "6,6,1.0,  6,7,1.0,  6,19,1.0,  6,3,1.0",
            "949,1",
            "7,7,1.0,  7,8,1.0,  7,20,1.0,  7,6,1.0",
            "940,1",
            "8,8,1.0,  8,9,1.0,  8,21,1.0,  8,7,1.0",
            "941,1",
            "9,9,1.0,  9,10,1.0,  9,21,1.0,  9,8,1.0",
            "942,1",
            "10,10,1.0,  10,13,1.0,  10,21,1.0,  10,9,1.0",
            "951,3",
            "11,11,1.0,  11,10,0.1384,  11,21,1.0,  11,  8,1.0",
            "  0,  0,1.0,  11,12,0.2000,   0,  0,1.0,   0,  0,1.0",
            "  0,  0,1.0,  11,14,0.6616,   0,  0,1.0,   0,  0,1.0",
            "950,1",
            "12,12,1.0,  12,13,1.0,  12,21,1.0,  12,11,1.0",
            "953,1",
            "13,13,1.0,  13,16,1.0,  13,21,1.0,   0,  0,1.0",
            "962,1",
            "14,14,1.0,  14,15,1.0,  14,21,1.0,  14,11,1.0",
            "963,1",
            "15,15,1.0,  15,16,1.0,  15,21,1.0,  15,14,1.0",
            "964,1",
            "16,16,1.0,  16,17,1.0,  16,21,1.0,  16,15,1.0",
            "964,1",
            "17,17,1.0,   0,  0,1.0,  17,21,1.0,  17,16,1.0",
            "854,1",
            "18,18,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
            "884,1",
            "19,19,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
            "894,1",
            "20,20,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
            "814,1",
            "21,21,1.0,   0,  0,1.0,   0,  0,1.0,   0,  0,1.0",
            "21",
            "925,  0.312700E-10,  1",
            "926,  0.334000E-10,  2",
            "937,  0.333600E-10,  3",
            "928,  0.350000E-10,  4",
            "939,  0.333100E-10,  5",
            "948,  0.315500E-10,  6",
            "949,  0.322200E-10,  7",
            "940,  0.324400E-10,  8",
```

```
            "941, 0.324800E-10, 9",
            "942, 0.330100E-10,10",
            "951, 0.353800E-10,11",
            "950, 0.344900E-10,12",
            "953, 0.344300E-10,13",
            "962, 0.351000E-10,14",
            "963, 0.351600E-10,15",
            "964, 0.340100E-10,16",
            "965, 0.352900E-10,17",
            "854, 0.0         ,18",
            "884, 0.0         ,19",
            "894, 0.0         ,20",
            "814, 0.0         ,21",
            ]
        inp = CardFile( card_text )

        ref_inp = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/SCG-cyc04" )
        assert inp.has_same_contents_with( ref_inp )


        # unit 8
        micro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
        micro_file.set_generate_code( "CONVMIC" )
        micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

        # unit 30
        nd_forward_boc = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT30/cyc04-BOC" )
        nd_forward_boc.set_generate_code( "FIRE1" )
        nd_forward_boc.set_filetype( "NUMBER_DENSITY_EACH" )

        # unit 31
        nd_adjoint_eoc = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT31/cyc04-EOC" )
        nd_adjoint_eoc.set_generate_code( "FIRE1" )
        nd_adjoint_eoc.set_filetype( "NUMBER_DENSITY_ADJOINT_EACH" )

        # unit 34
        sigfk = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT34/cyc04-BOC" )
        sigfk.set_generate_code( "FIRE1" )
        sigfk.set_filetype( "KAPPA_SIGF_XS" )

        # unit 35
        mat_t = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT35/cyc04-BOC" )
        mat_t.set_generate_code( "FIRE1" )
        mat_t.set_filetype( "MATRIX_T" )

        # unit 36
        mat_r = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/fire1.FT36/cyc04-BOC" )
        mat_r.set_generate_code( "FIRE1" )
        mat_r.set_filetype( "MATRIX_R" )

        # Unit 51
        database = DataBase( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/DataBase" )

        scgivebr.inputs( [inp, micro_file, nd_forward_boc, nd_adjoint_eoc, sigfk, mat_t, mat_r, database] )
        scgivebr.run()
        list_file = scgivebr.list()

        out_files = scgivebr.outfiles()   # unit 42, 52
        assert len(out_files) == 2

        # unit 42
        assert out_files[0].filetype() == "SOURCE"
        assert out_files[0].generate_code() == "SCGIVEBR"
        ref_unit_42 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/scgive.FT42/SCG-cyc04" )
        assert out_files[0].has_same_contents_with( ref_unit_42 )

        # unit 52
        assert out_files[1].filetype() == "SENS_ND_AND_POW"
        assert out_files[1].generate_code() == "SCGIVEBR"
        ref_unit_52 = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/scgive.FT52/SCG-cyc04" )
        assert out_files[1].has_same_contents_with( ref_unit_52 )


def suite():
    return unittest.makeSuite( ScgivebrTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```

図 A3-18 StartupTest.py

```
import unittest
from  sagep.config import *

from sagep.File.CardFile import *
from sagep.File.ModuleFile import *

class StartupTest(unittest.TestCase):

    def testRun(self):
        startup = Startup()

        card_text = [
            "16,33,18",
            "18",
            "949,1,1,0.0",
            "949,1,2,0.0",
            "949,1,3,0.0",
            "949,1,4,0.0",
            "949,1,5,0.0",
            "949,1,6,0.0",
            "949,1,7,0.0",
            "949,1,8,0.0",
            "949,1,9,0.0",
            "949,1,10,0.0",
            "949,1,11,0.0",
            "949,1,12,0.0",
            "949,1,13,0.0",
            "949,1,14,0.0",
            "949,1,15,0.0",
            "949,1,16,0.0",
            "949,1,17,0.0",
            "949,1,18,0.0"
            ]
        inp = CardFile( card_text )

        micro_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/micro" )
        micro_file.set_generate_code( "CONVMIC" )
        micro_file.set_filetype( "SAGEP-BURN_MICRO_XS" )

        nd_file = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/WORK/refuel.FT12/cyc01-BOC" )
        nd_file.set_generate_code( "REFUEL" )
        nd_file.set_filetype( "NUMBER_DENSITY_REGION" )

        startup.inputs([inp, micro_file, nd_file])
        startup.run()
        list_file = startup.list()

        ref_list = DataFile( ref1_dir + "/H13-Na-MOX.C3/BR/Burn/STARTUP-cyc01-BOC.dmp" )
        assert list_file.has_same_contents_with( ref_list )

        out_files = startup.outfiles()
        assert len(out_files) == 1  # scrach file is ignored
        assert out_files[0].filetype() == "MACRO_XS", out_files[0].filetype()
        assert out_files[0].generate_code() == "STARTUP"


def suite():
    return unittest.makeSuite( StartupTest, 'test' )

if __name__ == "__main__":
    runner = unittest.TextTestRunner()
    runner.run(suite())
```