JNC TN9400 2005-010

燃料集合体変形解析コードBAMBOOの高速化のための改良

（研究報告）

2005年4月

核燃料サイクル開発機構
大洗工学センター

〒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 TN9400 2005-010
2005 年 4 月

燃料集合体変形解析コードＢＡＭＢＯＯ の高速化のための改良

（研究報告）

伊藤　昌弘[*1]　、上羽　智之[*2]

要　　旨

燃料ピンバンドルの変形が生じた集合体の健全性評価を目的として、燃料集合体変形解析コード BAMBOO と熱流動解析コード ASFRE-IV を結合し、これらの挙動の相互作用を評価できるようにするための連成機能の整備を行なっている。この連成解析では、燃料ピンバンドルの変形と冷却材温度の変化に関して数値的な安定解を得るために収束計算を行うので、多くの解析時間を必要とする。このため、連成解析を効率的に行うために、BAMBOO コードを高速化するための改良を実施した。

BAMBOO コードは、有限要素法を用いているため、行列演算で多くのデータを格納する領域が必要であるが、このデータをハードディスクに格納していた。ハードディスクは、メモリーよりも低速なため、ハードディスクを用いたデータ入出力処理は、解析時間の増大に影響していた。このため、BAMBOO コードの改良では、ハードディスクへのデータ入出力処理を廃止し、メモリーを用いてデータ入出力処理を行うようにするとともに、高性能な計算機で実行できるように整備した。その結果、解析時間に対する CPU 利用率が上昇し、従来の約１／７の時間で解析を行うことができるようになった。

*1 株式会社 NESI
*2 大洗工学センター　システム技術開発部　核燃料工学グループ

JNC TN9400 2005-010
April, 2005

Improvement of the computing speed of the FBR fuel pin bundle deformation analysis code "BAMBOO "

Masahiro ITO [*1] , Tomoyuki UWABA [*2]

Abstract

JNC has developed a coupled analysis system of a fuel pin bundle deformation analysis code "BAMBOO" and a thermal hydraulics analysis code "ASFRE-IV" for the purpose of evaluating the integrity of a subassembly under the BDI condition. This coupled analysis took much computation time because it needs convergent calculations to obtain numerically stationary solutions for thermal and mechanical behaviors. We improved the computation time of the BAMBOO code analysis to make the coupled analysis practicable.

"BAMBOO" is a FEM code and as such its matrix calculations consume large memory area to temporarily stores intermediate results in the solution of simultaneous linear equations. The code used the Hard Disk Drive (HDD) for the virtual memory area to save Random Access Memory (RAM) of the computer. However, the use of the HDD increased the computation time because Input/Output (I/O) processing with the HDD took much time in data accesses. We improved the code in order that it could conduct I/O processing only with the RAM in matrix calculations and run with in high-performance computers. This improvement considerably increased the CPU occupation rate during the simulation and reduced the total simulation time of the BAMBOO code to about one-seventh of that before the improvement.

*1 NESI Inc.
*2 Nuclear Fuel Research Group, System Engineering Technology Division, OEC, JNC

JNC: Japan Nuclear Cycle Development Institute

目　　次

図・表リスト

１.はじめに

燃料ピンバンドルの変形と集合体内の温度分布変化の相互作用を考慮することにより、精度の高い温度場および変形を評価することを目的として、集合体バンドル変形解析コード(BAMBOO)と熱流力解析コード(ASFRE-IV)を連成解析させるための整備を進めている。連成アルゴリズムの構築および機能検証を目的として、平成１５年度に実施した「変形-熱流動」連成解析[1]では、バンドル変形と冷却材温度に関して数値的な安定解を得るために、収束計算が必要であることが明らかになった。これは、解析時間ステップ毎に収束解を得るまでASFRE-IVコードとBAMBOOコードの間で反復計算を実施することを意味する。この反復計算は、解析時間の増大につながるため、今回は、BAMBOO コードを高速化するための改良を実施した。

また、これまで手動で実施していた反復計算を自動化するにあたり、ASFRE-IV コードと BAMBOO コードの実行環境を統一させる必要があるので、BAMBOOコードをASFRE-IVコードが作動しているシステムへ移行する作業を行った。

本報告の構成は次のようになっている。第２章ではBAMBOOコードの改良内容と実行環境移行に関する作業内容を示す。第３章では改良に関する検証とこの改良で得られた効果について述べる。

２.改良内容

２.１　背景

平成１５年度に行った「変形-熱流動」連成解析[1]では、原型炉仕様の集合体について図 2-1 のように ASFRE-IV コードによる熱流動解析と BAMBOO コードによるバンドル変形解析（熱変形解析）を４回反復計算した。この時、ASFRE-IV コードの平均解析時間が約２時間４０分であったのに対し、BAMBOO コードの平均解析時間は約２０時間７分であった。両コードの解析で使用した計算機は、ほぼ同じ性能であるので、BAMBOO コードの解析は ASFRE-IV コードよりも多くの時間（約 7.5 倍）を要したことになる。また、BAMBOO コードを実行していたクラスタシステム（図 2-2）が、ディスクサーバへの過負荷によってシステムが停止するというトラブルが発生した。調査の結果、システム停止の原因は、BAMBOO コードで使用されている FFCP（Finas File Control Package）と呼ばれるファイルコントロールシステムの処理が主な要因であることがわかり、これは解析時間の増大にも影響していることがわかった。

FFCP は、図 2-3 に示すように BAMBOO コード開発初期に、主記憶装置（メモリー）資源枯渇対策として設計・開発されたファイルコントロールシステムであり、データの記憶域としてメモリーの代わりにハードディスクを用いて、ファイルの入出力処理を行う。障害が発生したクラスタシステムは、図 2-2 に示すように計算用サーバとディスクサーバを低速なネットワークで結ぶ構成になっており、FFCP によって大量のデータアクセスが発生し、ディスクサーバの負荷が許容限度を超えたことがシステム停止の原因となった。また、ハードディスクはメモリーと比較してアクセス速度が遅いため、FFCP によるファイルの入出力処理が完了するまでの間、CPU に待ち時間が発生し、エラップスタイム（CPU の待ち時間を含めた全解析時間）が長くなる原因となった。

一方、ASFRE-BAMBOO 反復計算を４回行ったが、冷却材流路形状と冷却材温度は反復計算の前回値に対して変化しており、バンドル変形と冷却材温度の平衡状態を解析するためには ASFRE-BAMBOO の反復計算回数を増やす必要があることがわかった。しかし、これまで ASFRE-IV コードと BAMBOO コードはそれぞれ異なる計算機で運用しており、反復計算に必要なデータ（冷却材流路形状と冷却材温度）の受け渡しを手動で行っていた。このため、ASFRE-BAMBOO 反復計算を効率良く行うためには、これまで異なる環境で実行していた両コードを同じ計算機上に移植し、データの受け渡し処理を自動化する必要がある。

これらの問題を解決するため、FFCP によるファイル入出力処理を廃止し、BAMBOO コードを ASFRE-IV コードと同じ計算機上で実行させるための改良を行うことにした。

２．２　BAMBOO コード概要

BAMBOO コードは、高速炉燃料集合体の BDI 発生下における燃料ピンバンドルの変形挙動を有限要素法によって解析するプログラムである。図 2-4 に解析モデルを示す。コードではバンドル内全ての燃料ピンを 1/12 ワイヤピッチの２節点梁要素の結合によりモデル化しており、これら節点で隣接ピン同士あるいはバンドル最外周ピンとラッパ管との接触解析を行い、全ての燃料ピンについて荷重と変位の力学的な平衡状態を求める。これらは次式により定式化している。

$$\begin{bmatrix} K & Bc \\ Bc^T & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta\mu \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} P \\ 0 \end{bmatrix} + \begin{bmatrix} Q \\ 0 \end{bmatrix} - \begin{bmatrix} R \\ 0 \end{bmatrix} + \begin{bmatrix} \lambda \\ \delta \end{bmatrix} \quad \cdots (1)$$

ここで、

K：ピンの剛性マトリクス
Bc：接触拘束マトリクス
P：外荷重ベクトル
R：応力に等価な荷重ベクトル
Q：照射クリープ擬似荷重ベクトル
λ：接触荷重ベクトル
Δμ：ピンの変位増分ベクトル
Δλ：接触荷重増分ベクトル
δ：オーバーラップベクトル

連立方程式の未知数が変位増分と接触力増分であるが、係数マトリクスの接触力増分に対応する対角成分がゼロとなっているので、このままでは方程式を解くことができない。このため、方程式を変位増分式と接触力増分式に分離して、収束繰り返し計算の中で変位増分と接触力増分を相互に参照しながら解を求める２段階解法を採用している。図 2-5 に BAMBOO コードのフローチャートを示す。

まず、接触荷重増分ベクトルをゼロとして、変位増分ベクトルを計算する。計算には、変位求解ソルバーを用いる。

$$\Delta\mu = K^{-1} \cdot \{P + Q - R + \lambda\} \quad \cdots (2)$$

変位増分により生じる燃料ピンバンドル内で接触判定を行い、接触拘束マトリクスおよびオーバーラップベクトルを計算する。接触力求解ソルバーでは、変位増分により生じたオーバーラップを打ち消す反力（接触荷重増分：Δλ）を計算する。

$$\Delta\lambda = \left(Bc^T \cdot K^{-1} \cdot Bc\right)^{-1} \cdot \delta \quad \cdots (3)$$

$$\lambda = \Delta\lambda + \lambda \qquad \cdots(4)$$

(3)式より、計算した接触荷重増分ベクトルを接触荷重ベクトルに加え、(2)式により、変位増分ベクトルを計算する。以降、オーバーラップがゼロになるまで、(2)～(4)式の処理を繰り返す。以上が、BAMBOO コードの主な処理である。

変位求解ソルバーおよび接触力求解ソルバー（(2),(3)式）では、連立方程式の解法に LU 分解を用いている。係数行列の次元の大きさは一般的な解析モデルで 1000～10000 元となるため、係数行列を格納する領域と係数行列を LU 分解するための作業域として 100 メガバイト～2 ギガバイトの記憶領域を必要とする。BAMBOO コードの開発は、昭和６３年度に開始したが、その当時の計算機環境下では、約 2 ギガバイトものデータを主記憶装置（メモリー）で管理することは事実上不可能であったため、FFCP を用いてハードディスクでデータの入出力処理を行うことによって連立方程式を解いていた。しかし、近年は大容量のメモリーを搭載した計算機が普及してきたため、FFCP でデータの入出力処理を行う必要はなく、逆に前述のように FFCP によるハードディスクへの入出力処理がクラスタシステムのトラブルや計算時間の増大などの原因となり問題になっていた。

２．３　改良内容

FFCP によるハードディスクへの入出力処理を廃止するにあたり、BAMBOO コードにおいて FFCP がどのようなデータを管理しているか調査した。その結果、連立方程式の係数行列の他に、剛性マトリクス、荷重ベクトル、温度データ、照射量データなど解析に必要なデータのほとんどを FFCP で管理しており、ハードディスクからメモリーへ移行する必要があるデータは膨大な量となることがわかった。また、図 2-5 のフローチャートに示すように FFCP による処理を行う箇所はコード全体に及ぶことがわかった。このため、調査結果を基にデータ項目を整理し、その用途に応じた管理手法を適用した。

●データ管理手法

今回の改良は、FFCP を廃止し、これまでハードディスクで管理していたデータをメモリーに移行することである。通常、FORTRAN プログラムにおいてデータをメモリーで管理する場合はコモン変数を用いる。しかし、今回の改良では FFCP からメモリーへ移行するデータが多いので、全てのデータにコモン変数を割り当てるのは効率が悪かった。そこで、FFCP から移行するデータを「計算実行中に保持または蓄積が必要なデータ（温度や照射量、ピン径などのデータ)」と「一時的に記憶域が必要なデータ（LU 分解のための作業域など)」に分け、前者のデータは新たに作成したコモン変数へ移行し、後者のデータはワーク領域に移行した。ワーク領域とは図 2-6 に示すように、プログラム起動時に確保する専用のメモリ

ー領域であり、データ毎にコモン変数を用意するよりも、プログラム全体のメモリー消費量を節約することができる。

改良内容の詳細については、プログラマーズマニュアル（改良ルーチン一覧およびFFCPファイル毎の改良内容）として、APPENDIXに記載した。

２．４　実行環境移植

これまで、ASFRE-IVコードとBAMBOOコードは、それぞれ異なる計算機で実行しており、「変形-熱流動」連成解析に必要なデータ（温度データ、流路形状データ）の受け渡しは、手動で行っていた。具体的には、ASFRE-IVコードの解析で得られた温度データをBAMBOOコードの実行環境にファイル転送し、それを入力データとしてBAMBOOコードを実行する。ASFRE-IVコード側も同様にBAMBOOコードの解析で得られた流路形状データをファイル転送で受け取り、それを入力データとして解析を実行する。これらの操作をそれぞれの解析コードの運用者が手動で実施していた。ASFRE-BAMBOO間の反復計算を効率良く行うためには、このデータの受け渡し処理を自動化する必要がある。データの受け渡し処理を自動化するには、ASFRE-IVコードとBAMBOOコードを同一の環境で実行させる必要があるため、BAMBOOコードの実行環境を移植するための作業を実施した。移植先の実行環境は、サイクル機構が所有する科学技術計算機（HPC2500）とAT互換機のLinuxワークステーションを対象とした。

BAMBOOコードのデータ入出力処理に使用されていたFFCPは、バイナリデータを扱うため、コンパイラやCPUに依存する処理があり、他の計算機へ移植するには、移植先の計算機に合わせプログラムを修正する必要があった。しかし、今回の改良でFFCP処理を廃止したので、プログラムの修正は不要になり、上記のそれぞれの計算機環境用にメイクファイルを作成することで実行環境を移植することができた。

３.機能検証

３.１　解析モデル

平成１５年度に行った「変形-熱流動」連成解析[1]で実施した原型炉仕様の集合体と同じ条件で BAMBOO コードによるバンドル熱変形解析を実施した。図 3-1 に解析モデル概略図を示す。解析条件を以下に示す。

●集合体形状

燃料ピン本数	:	169 本
燃料ピン外径	:	6.50［ mm ］
燃料ピン配列ピッチ	:	7.87［ mm ］
ピン全長	:	2813［ mm ］
ワイヤー外径	:	1.32［ mm ］
ワイヤー巻き付けピッチ	:	307［ mm ］
ワイヤー巻き始め位置	:	65.4167［ mm ］（ピン下端から）
ワイヤー巻き始め方向	:	0［ deg ］
ダクト内対面間距離	:	104.6［ mm ］
ダクト肉厚	:	3.0［ mm ］

●材料物性値（被覆管、ワイヤ、ダクト共通）

温度（℃）	ヤング率（GPa）	線熱膨張係数(/℃)
150	186.2	1.712×10^{-5}
250	180.3	1.829×10^{-5}
350	172.5	1.924×10^{-5}
450	163.7	2.000×10^{-5}
550	153.9	2.059×10^{-5}
650	144.1	2.104×10^{-5}

●温度データ

解析ケース	ASFRE コード条件（バンドル形状）	JNC レポート[1]での解析識別番号
解析ケース１	直立バンドル	手順２
解析ケース２	変形バンドル	手順４（１回目の反復計算）
解析ケース３	変形バンドル	手順６（２回目の反復計算）
解析ケース４	変形バンドル	手順８（３回目の反復計算）
解析ケース５	変形バンドル	手順１０（４回目の反復計算）

３．２ 検証結果

３．１章に示す５ケースのバンドル熱変形解析について改良前の解析結果と改良後の解析結果を流路形状変数で比較した。改良前の解析は、平成１５年度に行った「変形-熱流動」連成解析[1]の結果であり、従来の BAMBOO コード実行環境（VT-Alpha600）で行ったものである。改良後の解析は、サイクル機構所有の科学技術計算機（HPC2500）で行った。図 3-2～3-6 に解析結果の比較図を示す。図に示すように改良前後の解析結果はほぼ一致しており、その誤差は最大で 5.65×10^{-3}%以下である。よって、改良（FFCP 廃止）と実行環境移植による従来の解析結果との互換性への影響は殆どなく、機能要件を満たしていると判断できる。

一方、実行時間に対しては、表 3-1 に示すように、５～８倍（平均解析時間で約７倍）の速度向上効果があった。

尚、Linux 環境でも同様の検証を行っており、実行速度が向上し、解析結果の互換性に問題がなかったことを確認している。

４.おわりに

「変形-熱流動」連成解析における ASFRE-BAMBOO 反復計算を効率良く行うために、解析時間の高速化を目的として、BAMBOO コードの改良および実行環境の移植を行った。改良では、BAMBOO コードのファイルコントロールシステムである FFCP を廃止し、ハードディスクへの入出力処理を最小限にした。実行環境の移植は、サイクル機構所有の科学技術計算機(HPC2500)と AT 互換機の Linux ワークステーションを対象にした。その結果、従来の約 1/７の時間で解析を行うことが可能になった。今後、大型バンドルの照射変形挙動評価や熱流力コード”ASFRE”との連成解析機能整備を本格化させるにあたり、今回の改良による実行時間の短縮化は、業務の効率化に大きく寄与するものと期待できる。

参考文献

[1]伊藤昌弘、今井康友、上羽智之、大島宏之、「燃料集合体変形解析コードBAMBOOと熱流動解析コードASFRE-IVによるバンドル変形・熱流動連成解析（研究報告）」、JNCレポート TN9400 2004-003、2004.

表３－１　改良前後における実行時間の比較

	実行時間		速度比
	改良前	改良後	（改良前／改良後）
解析ケース１	４１時間３６分３６秒	４時間５３分１５秒	8.5倍
解析ケース２	１０時間３４分４１秒	１時間５３分１４秒	5.6倍
解析ケース３	１１時間２８分１５秒	１時間４６分１２秒	6.5倍
解析ケース４	２８時間３４分０５秒	４時間３７分４０秒	6.2倍
解析ケース５	８時間２１分４２秒	１時間１７分３１秒	6.5倍
平均解析時間	**２０時間０７分０４秒**	**２時間５３分３４秒**	**7.0倍**

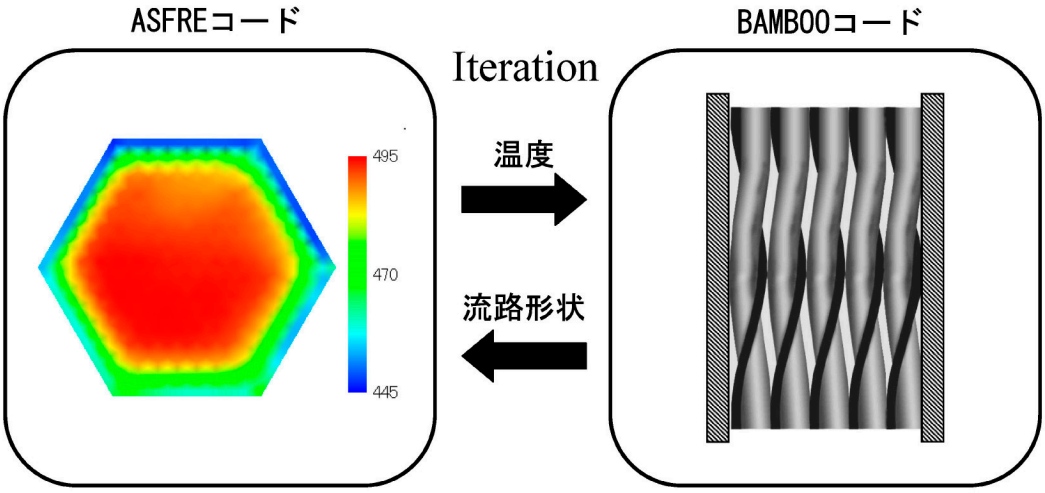


図 2-1　「変形-熱流動」連成解析イメージ

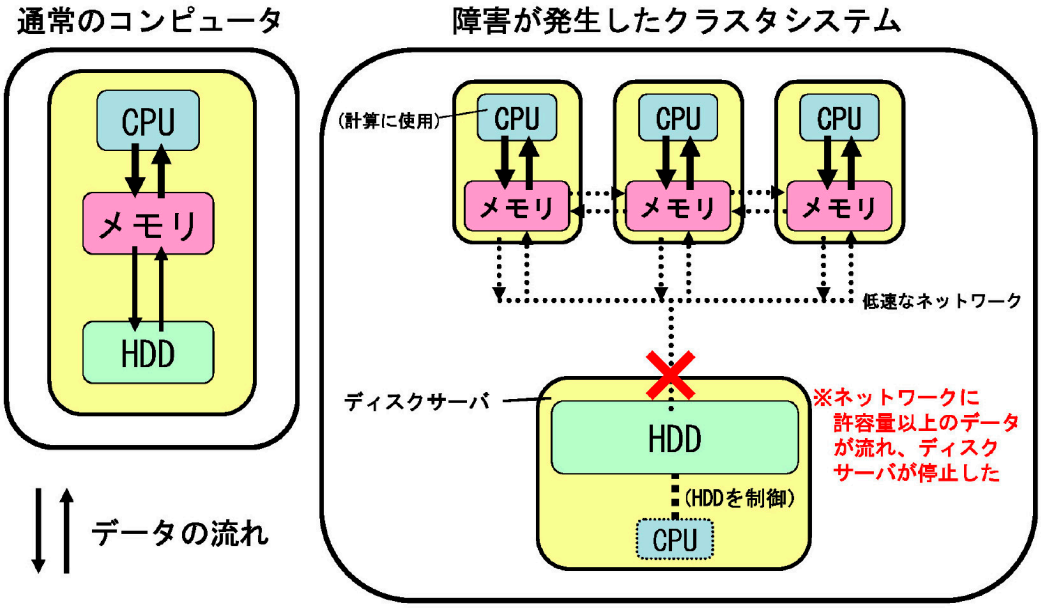


図 2-2　クラスタシステム概略図

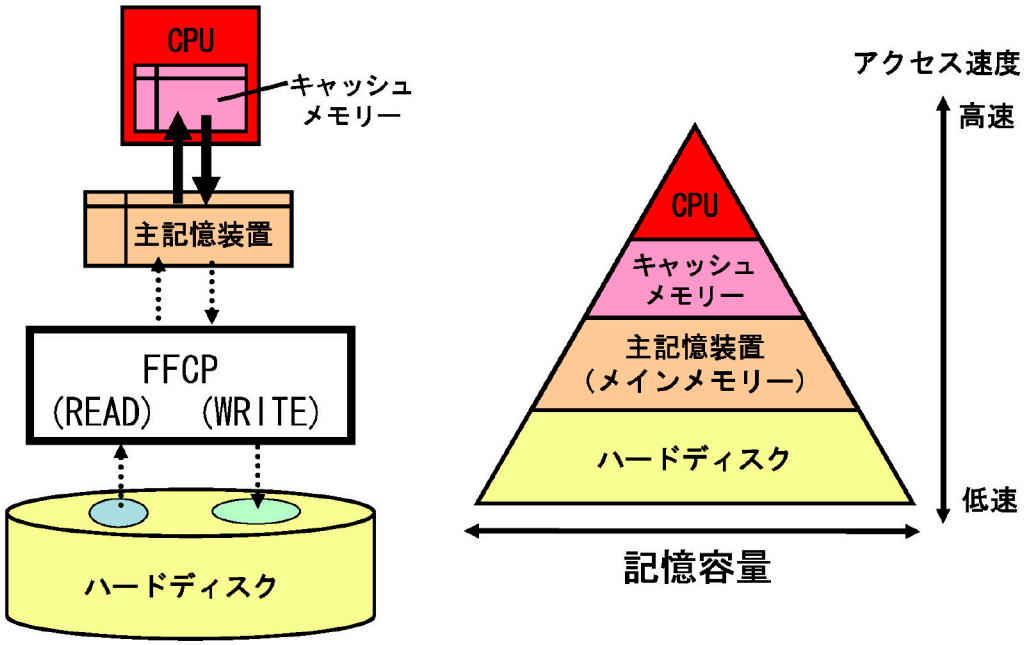


図 2-3　FFCP の機能概要

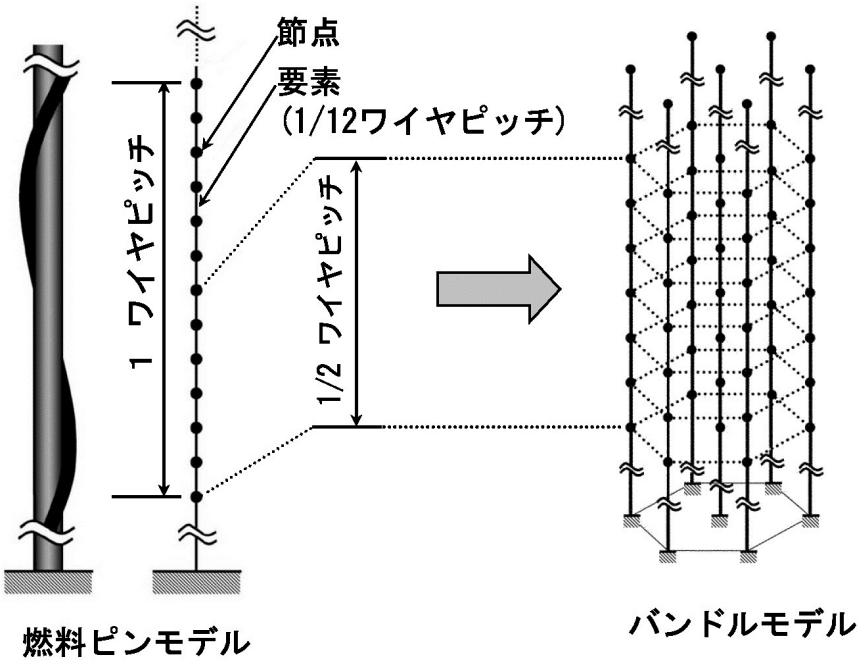


図 2-4　BAMBOO コード解析モデル

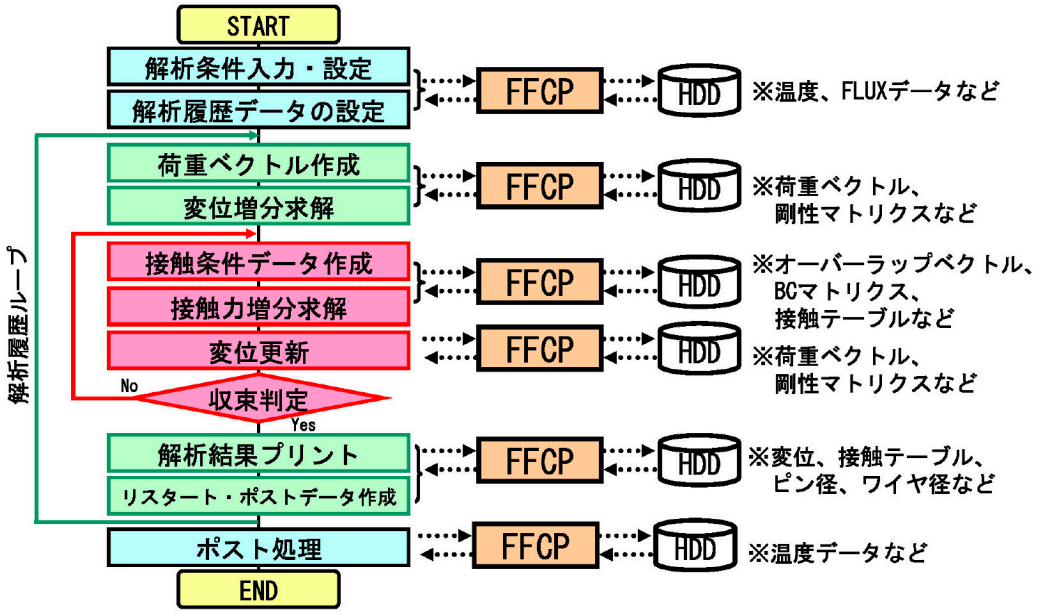


図 2-5　BAMBOO コードフローチャート

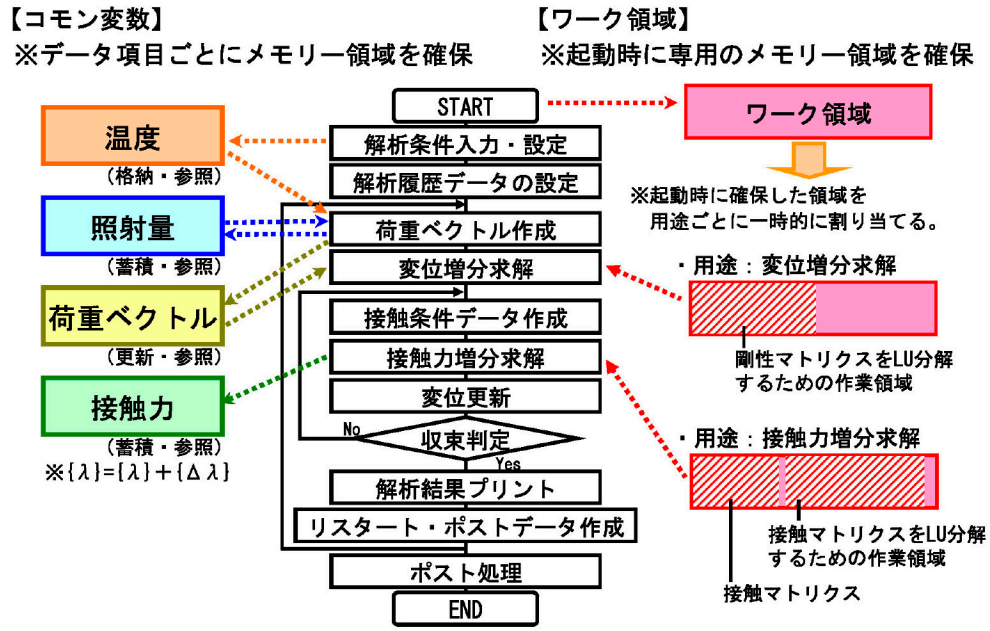


図 2-6　データ管理手法の比較

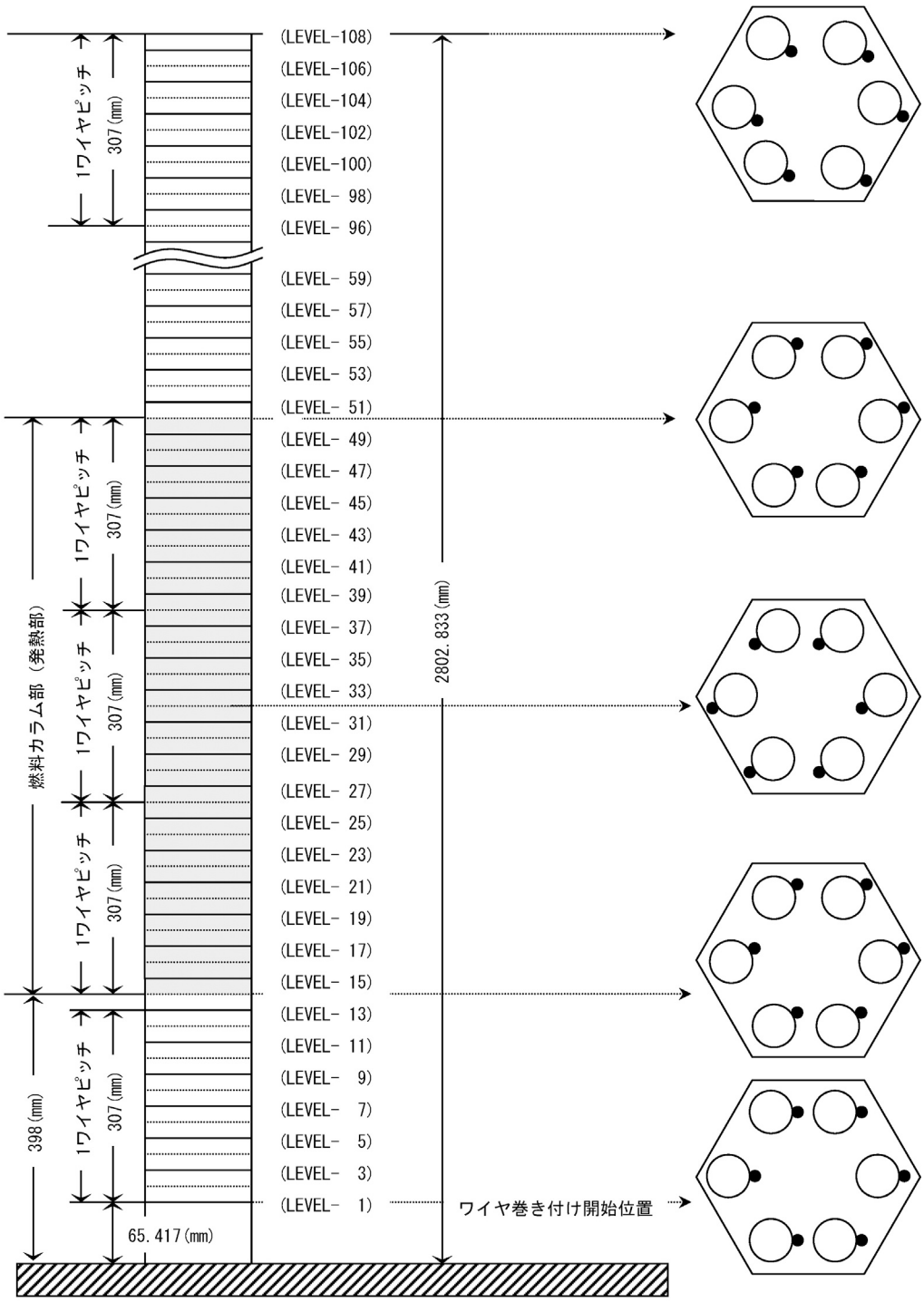


図 3-1　解析モデル概略図

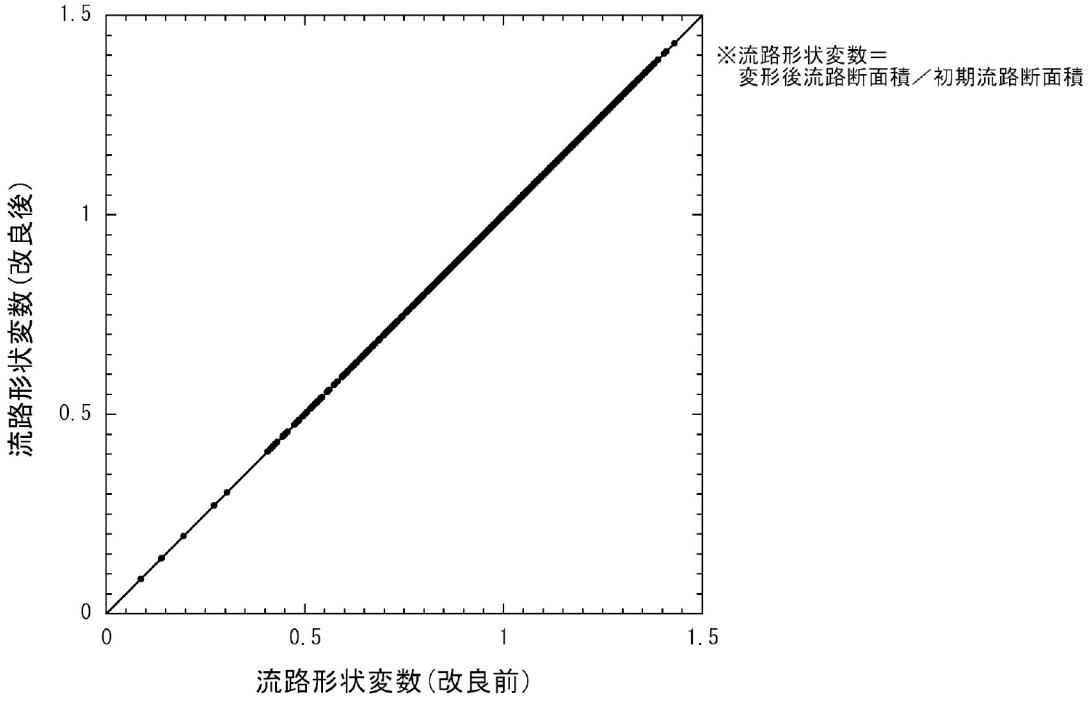


図 3-2　改良前後の解析結果比較（解析ケース１）

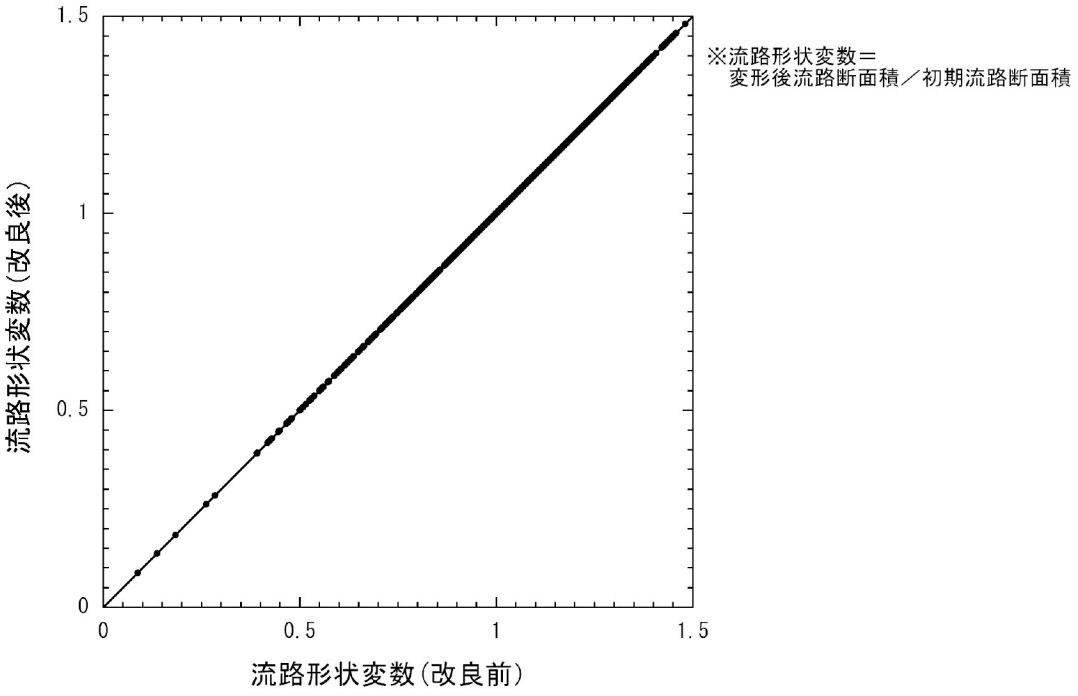


図 3-3　改良前後の解析結果比較（解析ケース２）

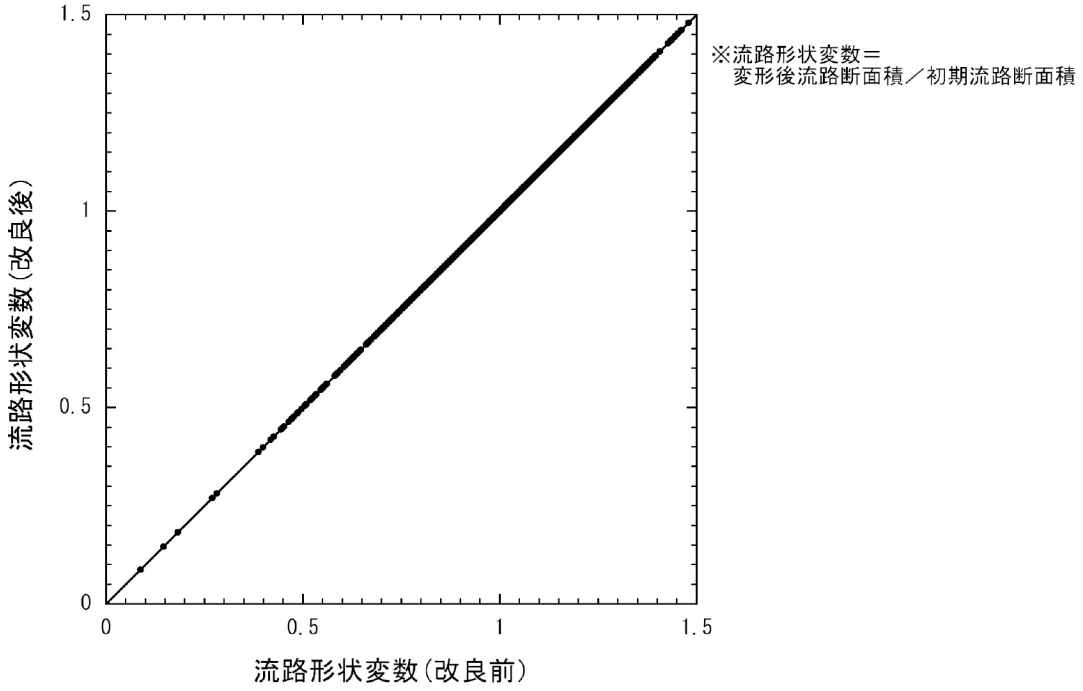


図 3-4　改良前後の解析結果比較（解析ケース３）

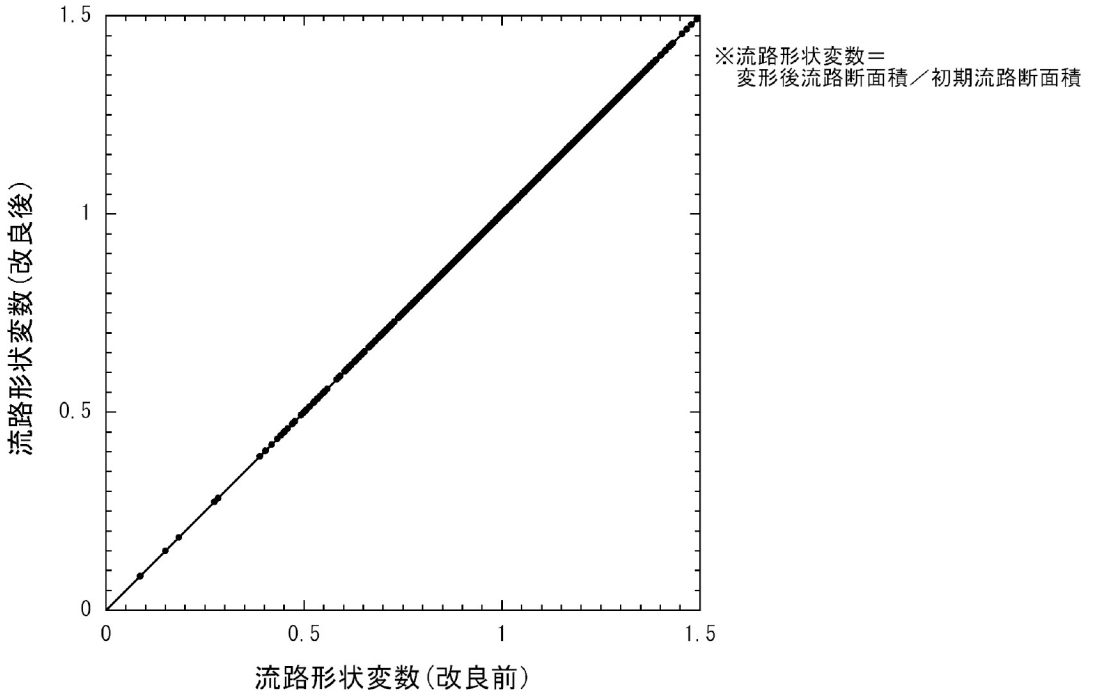


図 3-5　改良前後の解析結果比較（解析ケース４）

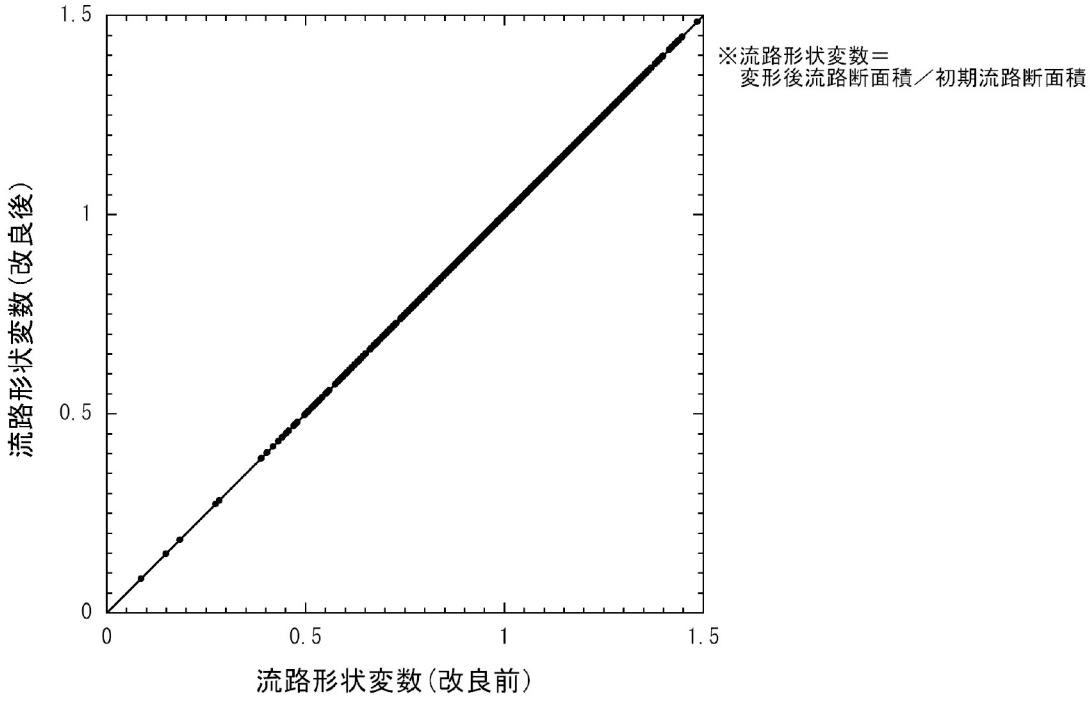


図 3-6　改良前後の解析結果比較（解析ケース５）

プログラマーズ・マニュアル

Ａ．改良ルーチン一覧表

	No	ルーチン名	改良の概要
ヘッダ	1	BMTEMP.h	温度データを格納するコモン変数（USR02.h から改名し、修正）
	2	CNTCT.h	接触テーブルの領域を拡張（CNTTBL(6,MXC)→CNTTBL(10,MXC)）
	3	CNTCTO.h	接触テーブル選択フラグを格納するためのコモン変数を新たに作成
	4	CNTINF.h	接触情報テーブルを格納するためのコモン変数を新たに作成
	5	COMLNG.h	コモン変数の領域拡張（LMODEL=30→LMODEL=33）
	6	DCTDAT.h	ダクトデータ（FLUX、照射量、スエリング量）を格納するためのコモン変数を新たに作成
	7	DCTDEF.h	ダクトデータ（変形量）を格納するためのコモン変数を新たに作成
	8	DCTHED.h	ダクトデータ（ヘッダ）を格納するためのコモン変数を新たに作成
	9	DISP.h	DDSP（変位増分）領域を拡張
	10	DPOINT.h	モデル情報ブロック（ノックバー膨張前の座標データ格納アドレスとデータ長）追加 変数追加（MQQQ(4)→MMSHNB,MQQQ(3)、LQQQ(4)→LMSHNB,LQQQ(3)）
	11	EQUIV.h	ノックバーオプションの追加に関する修正
	12	FORC.h	熱荷重ベクトル（TFRC）の追加に関する修正
	13	ITRCHK.h	収束計算の状態をチェックするためのコモン変数を新たに作成
	14	KCCPRM.h	接触マトリクス制御用コモン変数
	15	LEVTAB.h	ポスト処理でのレベル指定拡張
	16	LODFID.h	解析ステップ履歴データ（１）を格納するためのコモン変数を新たに作成
	17	LODFRD.h	解析ステップ履歴データ（２）を格納するためのコモン変数を新たに作成

	No	ルーチン名	改良の概要
ヘッダ	18	MODEL.h	ノックバーオプションに関する変数を追加
	19	MTBCF.h	BC マトリクスを格納するコモン変数を新たに作成
	20	MTBCN.h	BC マトリクスの管理変数を格納するコモン変数を新たに作成
	21	MTSTFF.h	剛性マトリクスを格納するコモン変数を新たに作成
	22	MTSTFN.h	剛性マトリクスの管理変数を格納するコモン変数を新たに作成
	23	MTTRIF.h	剛性マトリクスの[L][D]を格納するコモン変数を新たに作成
	24	MTTRIN.h	剛性マトリクスの[L][D]の管理変数を格納するコモン変数を新たに作成
	25	PARAMS.h	解析ステップ数最大値に関するパラメータ文
	26	PARAMW.h	ワーク領域のサイズを設定する
	27	PNFLUX.h	ピン FLUX データを格納するためのコモン変数を新たに作成
	28	RPDDAT.h	接触判定用ピン径、ワイヤ径を格納するコモン変数を新たに作成
	29	RPHDAT.h	RPHASE 用変数
	30	SPHDAT.h	シャットダウンモード変数（USR05.h を改名し、修正）
	31	STRCRP.h	クリープひずみ、応力データ
	32	STRSWL.h	スエリングひずみ、照射量データ（USR03.h を改名し、修正）
	33	SWAP.h	スワップ領域
	34	VCDELF.h	オーバーラップベクトルを格納するコモン変数を新たに作成
	35	VCDELN.h	オーバーラップベクトルの管理変数を格納するコモン変数を新たに作成

	No	ルーチン名	改良の概要
ヘッダ	36	WKDTEM.h	温度データ用ワーク領域
ルーチン	37	ADD	未使用ルーチンだったため削除
	38	ASFBDI	初期流路形状計算に関する修正
	39	ASFTMC	サブルーチンインタフェース変更に伴う修正、 温度データに関する FFCP 処理廃止に伴う修正
	40	ASFTMD	サブルーチンインタフェース変更に伴う修正
	41	ASFTMW	コモンブロック名変更に伴う修正
	42	BCOUT	BC マトリクスをコモン変数に格納、接触テーブル領域（6→10）を拡張
	43	BGNBDI	不要な処理を削除
	44	BITOFF	未使用ルーチンだったため削除
	45	BLKDT1	コモン変数を追加
	46	BNUMBO	未使用ルーチンだったため削除
	47	CHKBDI	サブルーチンインタフェース変更に伴う修正
	48	CHKTNS	サブルーチン BCOUT のインターフェース改良、接触テーブル変更に関する修正
	49	CNTDAT	サブルーチン BCOUT のインターフェース改良。強制変位に係わる処理削除。変位増分ベクトルの FFCP 処理廃止、ノックバーに関する Write 文追加
	50	CPYSTR	未使用ルーチンだったため削除
	51	CRCOND	インタフェース改良
	52	CREAD1	ノックバー、新接触テーブルに関する入力データを追加
	53	CREAD2	解析ステップ数最大値に関する修正、ノックバーオプションに関する修正

	No	ルーチン名	改良の概要
ルーチン	54	CTBADS	接触テーブル７～１０項のクリア
	55	CTBMDF	FFCP 処理を廃止するための修正
	56	CTBSET	FFCP 処理を廃止するための修正、接触テーブル７，８項のクリア
	57	DATSET	FFCP 処理を廃止するための修正
	58	DATSTR	FFCP 処理を廃止するための修正
	59	DCTDFB	FFCP 処理を廃止するための修正
	60	DCTDFM	FFCP 処理を廃止するための修正
	61	DCTDFT	FFCP 処理を廃止するための修正
	62	DFLTOT	初期流路形状計算に関する修正
	63	DSPINT	強制変位に係わる処理を削除。剛性マトリクス、荷重増分ベクトル、変位増分ベクトルの FFCP 処理廃止
	64	DUMMYQ	未使用ルーチンだったため削除
	65	EASCLD	サブルーチンインタフェース変更に伴う修正、熱荷重ベクトル（TFRC）の追加に関する修正
	66	EASCLR	サブルーチンインタフェース変更に伴う修正、熱荷重ベクトル（TFRC）の追加に関する修正
	67	EASLD	サブルーチンインタフェース変更に伴う修正、熱荷重ベクトル（TFRC）の追加に関する修正
	68	EASLD2	熱膨張ひずみ等価荷重作成に関する修正
	69	EPSRPH	照射クリープひずみ、応力に関する FFCP 処理廃止に伴う修正
	70	EPSSCR	温度、応力、照射クリープひずみ、スエリング、照射量、FLUX に関する FFCP 処理を廃止するための修正
	71	EPSTCR	温度、応力、熱クリープひずみに関する FFCP 処理廃止に伴う修正

	No	ルーチン名	改良の概要
ルーチン	72	FINTPD	未使用ルーチンだったため削除
	73	GAPSET	初期流路形状計算に関する修正
	74	GETDSP	未使用ルーチンだったため削除
	75	GETPRM	FFCP の GETSTR 機能の代用ルーチンを新たに作成。接触マトリクスを制御する。
	76	GMESH	初期座標のコピー、PDNB←PDLENG
	77	HEADCK	入力データに関する修正
	78	HPLT	初期流路形状計算に関する修正
	79	HSCOND	サブルーチンインタフェース変更に伴う修正
	80	IBFLG	ANDF 関数（C 言語ルーチン）を廃止し、FORTRAN 組み込み関数 IAND を適用
	81	INITAR	荷重ベクトル（FRC,CFC,RES)、スエリング、クリープひずみ、FLUX ファイルに関する FFCP 処理廃止に伴う修正、熱荷重ベクトル（TFRC）の追加に関する修正
	82	IPNUMB	未使用ルーチンだったため削除
	83	JGCNT2	コモン変数 ANGLE 削除、GETDSP 削除、CNTTBL(6,1)→CNTTBL(10,1)、PPITCH→PPCH、PDLENG→PDLNG
	84	JGCONT	コモン変数 ANGLE 削除、CNTTBL(6,1)→CNTTBL(10,1)、PPITCH→PPCH、PDLENG→PDLNG
	85	KPASMC	FFCP 処理を廃止するための修正
	86	KREAD2	接触テーブル領域拡張に関する修正
	87	KREADC	扁平化に関する READ 文を追加
	88	LBDI	WRITE 文の修正
	89	LCLRD	FORMAT 文の修正（I2→I3)

	No	ルーチン名	改良の概要
ルーチン	90	LCLRP	FORMAT 文の修正（I2→I3）
	91	LCONT	サブルーチン LCONTB と LCONTC を選択する処理を追加
	92	LCONTB	ワイヤを介さずにダクトと接触するピンの荷重を出力する機能を追加
	93	LCONTC	選択された節点の接触荷重を出力するサブルーチンを新たに作成
	94	LDASM	荷重増分ベクトルの FFCP 処理廃止に伴う修正、熱荷重ベクトル（TFRC）の追加に関する修正
	95	LDASM2	荷重増分ベクトルの FFCP 処理廃止に伴う修正、熱荷重ベクトル（TFRC）の追加に関する修正
	96	LDASMR	荷重増分ベクトルの FFCP 処理廃止に伴う修正、熱荷重ベクトル（TFRC）の追加に関する修正
	97	LDDTM2	未使用ルーチンだったため削除
	98	LDDTMP	FFCP 処理を廃止するための修正
	99	LDLOD	解析ステップ履歴データ（FFCP ファイル番号 109）の FFCP 処理廃止に伴う修正
	100	LDTMP	温度データの FFCP 処理廃止に伴う修正、コメント文追加
	101	LDTMP2	未使用ルーチンだったため削除
	102	LEVSET	初期流路形状計算に関する修正
	103	LISTOB	接触テーブル領域拡張に関する修正
	104	LPWD	ピン列別出力機能を追加
	105	LPWDB	扁平化弾性変形分計算、ワイヤずれ時の節点荷重分配機能に関する修正
	106	LSARE	FORMAT 文の修正（I2→I3）
	107	LSARE0	FORMAT 文の修正（I2→I3）

	No	ルーチン名	改良の概要
ルーチン	108	MADRS	未使用ルーチンだったため削除
	109	MAIN	FFCP 処理を廃止するための修正
	110	MAKPST	FFCP 処理を廃止するための修正、ポストファイル作成（接触テーブル、扁平化関連）
	111	MAKRST	FFCP 処理を廃止するための修正、リスタートファイル作成（接触テーブル、扁平化関連）、熱荷重ベクトル（TFRC）の追加に関する修正
	112	MESAGE	不要な処理を削除
	113	MKDAT	強制変位に係わる処理を削除（SETFXT を廃止）、サブルーチンインタフェース変更に伴う修正
	114	MKDATR	サブルーチンインタフェース変更に伴う修正、 初期座標読み込み処理追加
	115	NBCOND	ノックバーの熱膨張を計算するサブルーチン
	116	NEBTAS	未使用ルーチンだったため削除
	117	OUTPRE	ポスト処理でのレベル指定拡張
	118	OUTPUT	LCONT の呼び出し方法を変更
	119	OUTSTP	初期流路形状計算に関する修正
	120	PACK	接触マトリクスを圧縮し、ワーク領域に格納するための修正
	121	PAGE	WRITE 文の修正
	122	PCNTR	温度データに関する FFCP 処理廃止に伴う修正
	123	PDLNG	初期流路形状計算に関する修正
	124	PEXIT	FFCP 処理を廃止したことによる処置を追加
	125	PGAPE	ノックバー膨張機能に関する改良（PPCH←PPICH、PDLNG←PDLENG）

	No	ルーチン名	改良の概要
ルーチン	126	PL2SET	ノックバー膨張機能に関する改良（PPCH←PPICH）
	127	PL2STN	ノックバー膨張機能に関する改良（PDLNG←PDLENG）
	128	PLOT	DUMMY.f から独立。DUMMY.f を廃止。
	129	PNDC	FFCP 処理を廃止するための修正
	130	PNDT	FFCP 処理を廃止するための修正
	131	PPLT2N	未使用ルーチンだったため削除
	132	PRECH2	FFCP 処理を廃止するための修正、接触テーブルのバージョンをプリントするための修正
	133	PREPLT	FFCP 処理を廃止するための修正
	134	PRNCNT	接触テーブルの領域を拡張（CNTTBL(6,1)→CNTTBL(10,1)）
	135	PRNDIA	FFCP 処理を廃止するための修正
	136	PUTBCM	BC マトリクスをコモン変数へ格納するルーチンを新たに作成
	137	PUTPRM	FFCP の PUTSTR 機能の代用ルーチンを新たに作成。接触マトリクスを制御する。
	138	RCTR	未使用ルーチンだったため削除
	139	RDPOST	FFCP 処理を廃止するための修正、ポストファイル読み込み（接触テーブル、扁平化関連）
	140	RESID	残差荷重ベクトルの FFCP 処理廃止に伴う修正
	141	RNUMBC	BC マトリクスの FFCP 処理を廃止するための修正
	142	RPCOND	サブルーチンインタフェース変更に伴う修正
	143	RPHASE	コモンブロック追加（PARAMS.h）に伴う修正

	No	ルーチン名	改良の概要
ルーチン	144	RSREAD	FFCP 処理を廃止するための修正、リスタートファイル読み込み（接触テーブル、扁平化関連）、熱荷重ベクトル（TFRC）の追加に関する修正
	145	SAVLOD	解析ステップ履歴データ（FFCP ファイル番号 109）の FFCP 処理廃止に伴う修正
	146	SDCCNT	ワーク領域の接触マトリクスを参照するように改良（FFCP 処理廃止に伴う修正）
	147	SDCIN	ワーク領域の接触マトリクスを参照するように改良（FFCP 処理廃止に伴う修正）
	148	SDCOM1	接触マトリクスを LU 分解するためのサブルーチン
	149	SDCOMP	不要な処理の削除。インターフェース改良。
	150	SDCOUT	接触マトリクスの[L][D]をワーク領域に格納するための修正
	151	SETO	不要な処理を削除
	152	SETOR	不要な処理を削除
	153	SETBIT	未使用ルーチンだったため削除
	154	SETCNT	FFCP 処理を廃止するための修正、CNTTBL 領域の初期化（６→10）
	155	SETFD	FFCP 処理を廃止するための修正
	156	SETFDR	FFCP 処理を廃止するための修正
	157	SETFLX	FFCP 処理を廃止するための修正
	158	SETFXT	未使用ルーチンだったため削除
	159	SETPTP	サブルーチンインタフェース変更に伴う修正
	160	SETTMP	温度データに関する FFCP 処理廃止に伴う修正
	161	SLDS1A	強制変位に係わる処理を削除。FFCP 処理を廃止するための修正

	No	ルーチン名	改良の概要
ルーチン	162	SLDS1B	強制変位に係わる処理を削除。FFCP 処理を廃止するための修正
	163	SLDSPF	未使用ルーチンだったため削除
	164	SLST1A	FFCP 処理を廃止するための修正
	165	SLST1B	ワーク領域の接触マトリクスを参照するように改良（FFCP 処理廃止に伴う修正）
	166	SLSTPF	接触マトリクスの計算での FFCP 処理を廃止するための修正
	167	SOL1FB	ワーク領域の接触マトリクスを参照するように改良（FFCP 処理廃止に伴う修正）
	168	SOLCNT	強制変位に係わる処理を削除。CNTDAT、UPDDSP のインタフェース改良
	169	SOLDS1	強制変位に係わる処理を削除。FFCP 処理を廃止するための修正
	170	SOLDS3	FFCP 処理を廃止するための修正
	171	SOLDSP	強制変位に係わる処理を削除
	172	SOLRMD	強制変位に係わる処理を削除
	173	SOLST1	接触マトリクスおよび LU 分解作業領域をワーク領域に割り当てるための修正
	174	SRMLV	残差荷重ベクトルを計算するためのサブルーチン
	175	SRMR	未使用ルーチンだったため削除
	176	STCOND	剛性マトリクス、荷重増分ベクトルの FFCP 処理廃止、NBCOND の呼び出し、熱荷重ベクトル（TFRC）の追加、熱膨張ひずみ等価荷重作成に関する修正
	177	STRESS	クリープひずみ、温度、スエリングひずみの FFCP 処理廃止に伴う修正
	178	STRMR	未使用ルーチンだったため削除
	179	STWPDF	接触テーブルの領域を拡張（CNTTBL(6,1)→CNTTBL(10,1)）

	No	ルーチン名	改良の概要
ルーチン	180	SUBCHN	初期流路形状計算に関する修正
	181	SWPDAT	FFCP 処理を廃止するための修正
	182	T1XYP	初期流路形状計算に関する修正
	183	TA2ACN	BC マトリクスの FFCP 処理を廃止するための修正
	184	TA2APT	未使用ルーチンだったため削除
	185	TA2ASQ	FFCP 処理を廃止するための修正
	186	TA2AST	不要な処理を削除
	187	TMP1B	温度データの FFCP 処理廃止に伴う修正
	188	TMP2B	温度データの FFCP 処理廃止に伴う修正
	189	TPHASE	コモンブロック名変更に伴う修正
	190	UPDDSP	荷重増分ベクトルの FFCP 廃止、熱荷重ベクトル（TFRC）の追加に関する修正
	191	WIRTNS	コモンブロック名変更に伴う修正
	192	WRTBDI	FFCP 処理を廃止するための修正
	193	XSWEL2	未使用ルーチンだったため削除
	194	XYVALU	DSPINT からのインタフェース改良（DDSP 拡張による処置）
	195	XYWAKU	未使用ルーチンだったため削除

プログラマーズ・マニュアル

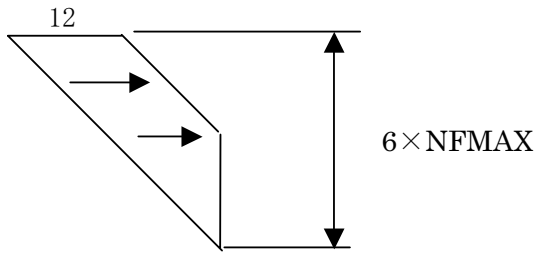
B. FFCP ファイル毎の改良内容

FFCP ファイル一覧

種別	FFCP 番号	識別子	ファイル名	改良後の記憶場所
データファイル （CRC 作業報告書のプログラマーズマニュアルに記載されている公開ファイル）	101	STF	全体剛性マトリクス	/MTSTFN/ /MTSTFF/
	102	EFC	荷重項ベクトル	/DISP/
	103	TRI	三角化マトリクス	/MTTRIN/ /MTTRIF/
	104	BCX	接触状態マトリクス	/MTBCN/ /MTBCF/
	105	DEL	オーバーラップベクトル	/VCDELN/ /VCDELF/
	106	CHS	接触情報	/CNTINF/
	107	FIX	強制変位	削除
	108	TMP	温度増分	/BMTEMP/
	109	LOD	荷重乗数	/LODFID/ /LODFRD/
	110	DDS	変位増分	/DISP/
	111,112	FRC	外荷重ベクトル	/ FORC/
	113	RES	残差荷重ベクトル	/ FORC/
	114	CFC	接触荷重ベクトル	/ FORC/
	115,116	PIN	接触判定用ピン径、ワイヤ径	/ RPDDAT/
	117,118	ECT	熱クリープひずみ	/ STRCRP/
	119,120	ECS	照射クリープひずみ	/ STRCRP/
	121,122	SWL	スエリング量	/ STRSWL /
	123	STR	応力	/ STRCRP/
	124,125	STP	評価点温度	/BMTEMP/
	126,127	FLX	中性子束	/ PNFLUX /
	128	DFX	中性子束分布（増分）	/ PNFLUX /
	129	DUCT	ダクト	/DCTDAT/ /DCTDEF/ /DCTHED/ /BMTEMP/
接触力求解ソルバー （接触マトリクス[kcc]の計算）	130	IWOKF	スワップデータ	/SWAP/
	301	N301	[kcc]計算用一時ファイル（[BC]インデックス）	ワーク領域（/BLKCOM/）
	302	N302	[kcc]マトリクス	
	303	N303	[kcc]計算用一時ファイル（[L1BC]マトリクス）	
	304	L304	リナンバー後[BC]マトリクス	/MTBCN/
	305	L305	リナンバー情報ベクトル	ワーク領域（/BLKCOM/）
接触力求解ソルバー （接触マトリクス[kcc]の LU 分解）	301	SCRTCH(1)	LU 分解に使用するメモリーが足りない場合のスワップ領域	削除
	303	SCRTCH(2)		
	304	LUNL	[kcc]の三角化マトリクス	ワーク領域（/BLKCOM/）
変位求解ソルバー	130	GSTF	強制変位に関するデータ	削除
	301	N301		
	302	ISCR(1)		
	303	ISCR(2)		

●FFCP 101 (STF) 全体剛性マトリクスファイル

NPINT	ピン本数	
NFMAX	ピンあたり節点数	
IFREE	節点あたり自由度（６）	
		ヘッダ
PSTF	ピン１に関する剛性マトリクス (R*8)	12×NFMAX×6
EOR		
PSTF	ピン２に関する剛性マトリクス (R*8)	12×NFMAX×6
EOR		
：		
：		
：		
：		12×NFMAX×6×NPINT



【改良】

以下のコモン変数に格納した。

/MTSTFN/ 全体剛性マトリクス管理変数データブロック

MTSTFH(6)	剛性マトリクスヘッダ 1:ピン本数 2:ピンあたり節点数 3:節点あたり自由度（６） 4:未使用 5:未使用 6:未使用

/MTSTFF/ 全体剛性マトリクスデータブロック

STFW(NPNMAX*NODMAX*72)	全体剛性マトリクス(R*8)

●FFCP 102 (EFC) 荷重項ベクトルファイル

NPINT	ピン本数	
NFMAX	ピンあたり節点数	
IFREE	節点あたり自由度（６）	
		ヘッダ
PLVEC	ピン１に関する荷重項 (R*8)	NFMAX×6
EOR		
PLVEC	ピン２に関する荷重項 (R*8)	NFMAX×6
EOR		
：		NFMAX×6×NPINT

荷重項 PLVEC は、ピンの変位増分を計算するための荷重ベクトル増分であり、以下の式で定義される。

{PLVEC}={FRC}+{CFC}-{RES}

{FRC}：外荷重ベクトル（熱荷重＋クリープ擬似荷重）
{CFC}：接触荷重ベクトル
{RES}：残差荷重ベクトル

【改良】

{PLVEC}は、ピンの変位増分を計算するために必要なデータである。

$\{\Delta u\}=[K]^{-1}\{PLVEC\}$

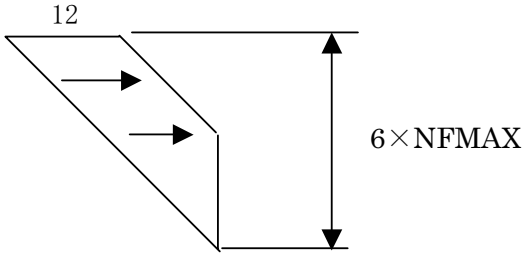
したがって変位増分計算後は不要であるため、新たなデータブロックは用意せず、既存の変位データブロックに格納した。また、FFCP ヘッダ領域に格納しているデータは未使用であったため、削除した。

/DISP/ 変位データブロック

DSP(MAXDOF*NODMAX*NPNMAX)	ピンの変位ベクトル(R*8)
DDSP(MAXDOF*NODMAX*NPNMAX)	ピンの変位増分ベクトル(R*8) （兼） 荷重項 PLVEC（ピンの荷重増分ベクトル）

●FFCP 103（TRI）三角化ファイル

NPINT	ピン本数	
NFMAX	ピンあたり節点数	
IFREE	節点あたり自由度（６）	
		ヘッダ
TRI	ピン１に関する三角化マトリクス （R*8）	12×NFMAX×6
EOR		
TRI	ピン２に関する三角化マトリクス （R*8）	12×NFMAX×6
EOR		
： ： ： ：		12×NFMAX×6×NPINT



剛性マトリクス[K]を修正コレスキー分解により、[L]（左下三角マトリクス）[D]対角マトリクス[U]（右上三角マトリクス）に分解し、このうちの[L]と[D]を格納したファイル。剛性マトリクス[K]は対称マトリクスであるため、[U]は不要（$[U]=[L]^T$）である。変位増分計算、接触力増分計算で用いる。

【改良】

以下のコモン変数に格納した。

/MTTRIN/ 三角化マトリクス管理変数データブロック

MTTRIH(6)	三角化マトリクスヘッダ 1:ピン本数 2:ピンあたり節点数 3:ワーク領域の大きさ 4:未使用 5:ワーク領域の大きさ 6:ワーク領域の大きさ
NTRITB(3,NPNMAX*NODMAX*6)	三角化マトリクス管理変数 (1,*):マトリクス列番号 (2,*):マトリクス位置情報 (3,*):対角成分位置情報
NTRILB(4,NPNMAX)	三角化マトリクス管理変数 (1,*):ピン番号 (2,*):6×節点数 (3,*):ピン番号ｉの管理変数先頭アドレス (4,*):ピン番号ｉの三角化マトリクス先頭アドレス

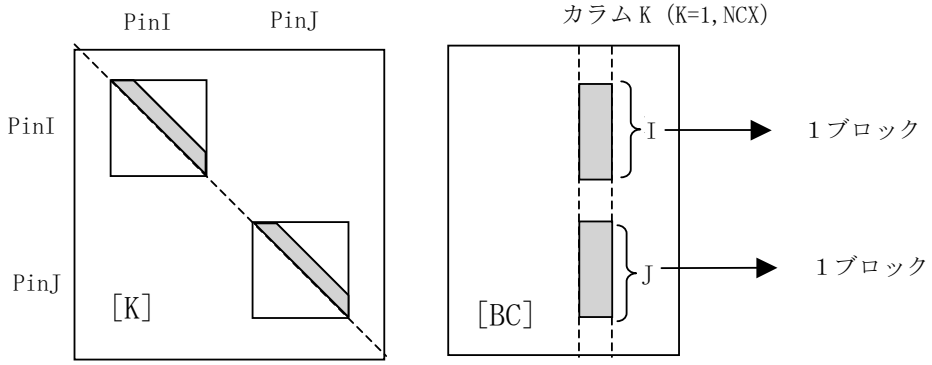
/MTTRIF/ 三角化マトリクスデータブロック

TRIW(NPNMAX*NODMAX*72)	三角化マトリクス（剛性マトリクスの[L][D]）(R*8)

●FFCP 104 (BCX) 接触状態マトリクスファイル

NPINT	ピン本数	
NFMAX	ピンあたり節点数	
IFREE	節点あたり自由度（６）	
NCX	接触点数	
		ヘッダ
IPN	ピン番号	
LEV	節点番号	１ブロック
BC(6)	BC 項 6 ワード (2I*4+6R*8)	
IPN	ピン番号	
LEV	節点番号	２ブロック
BC(6)	BC 項 6 ワード (2I*4+6R*8)	
：	：	1～4 ブロックで
：	：	１レコード
EOR	:	
：		
：		NCX×レコード
:		

接触状態マトリクス（[BC]マトリクス）は、接触しているピンの当該自由度以外は全てゼロ要素であるので、[BC]マトリクスの各列は下図のように各列のうち非ゼロを含むピンに対応したベクトルをピン番号、節点番号のタグとともに記録する。



１接触点につき１レコードとし、１レコード中１～４ブロック（可変）のデータが記録される。上記の例では PinI と PinJ が接触した場合であり、PinI 分と PinJ 分の２ブロックで１レコードのデータになる。

※Pin-Pin 接触でワイヤディスパージョン機能が有効な場合は、４ブロック
Pin-Duct 接触でワイヤディスパージョン機能が有効な場合は、２ブロック
Pin-Duct 接触でワイヤディスパージョン機能が無効な場合は、１ブロックとなる。

【改良】

１レコード中のブロック数が変化するため、BC マトリクスデータの格納領域の他に、BC マトリクスデータを管理する変数を用意した。以下に示すコモン変数に格納した。

/MTBCN/ BC マトリクス管理変数データブロック

NBCHED(6)	BC マトリクスヘッダ 1:ピン本数 2:ピンあたり節点数 3:節点あたり自由度（６） 4:接触点数 5:未使用 6:未使用
NBCLBL(2,MXC)	BC マトリクス管理変数 (1,*):BC マトリクス K の開始アドレス (2,*):BC マトリクス K の終了アドレス
NBCTAG(2,MXC*4)	BC マトリクスのタグ (1,*):ピン番号（タグ１） (2,*):節点番号（タグ２）

/MTBCF/ BC マトリクスデータブロック

BCMTX(6,MXC*4)	BC マトリクス(R*8) (1,*):BC 項１（接触割合） (2,*):BC 項２（接触方向 X 成分） (3,*):BC 項３（接触方向 Y 成分） (4,*):未使用 (5,*):未使用 (6,*):未使用

●FFCP 105 (DEL) オーバーラップベクトルファイル

NCX		接触点数	
			ヘッダ
DEL(1)		接触点 1 のオーバーラップ量 (R*8)	
DEL(2)		接触点 2 のオーバーラップ量 (R*8)	
⋮		⋮	
DEL(NCX)	EOL	接触点 NCX のオーバーラップ量(R*8)	NCX

各接触点で生じているオーバーラップ量を記録する。

【改良】

以下のコモン変数に格納した。

/VCDELN/ オーバーラップベクトル管理変数データブロック

NDELHD(6)	オーバーラップベクトルヘッダ 1:接触点数 2:未使用 3:未使用 4:未使用 5:未使用 6:未使用

/VCDELF/ オーバーラップベクトルデータブロック

DELVEC(MXC)	各接触点のオーバーラップ量(R*8)

●FFCP 106 (CHS) 接触情報ファイル

			ヘッダなし
⋮ ⋮ ⋮	EOR	⋮ ⋮ ⋮	
MCONT		該当レベルでの接触点数	
LEVL		該当レベル	2
IPN(1) IDRC(1) IADS(1)		ピン番号 接触方向番号 接触情報テーブルのアドレス	
⋮ ⋮		⋮ ⋮	
IPN(MCONT) IDRC(MCONT) IADS(MCONT)	EOR	ピン番号 接触方向番号 接触情報テーブルのアドレス	3×MCONT
⋮ ⋮ ⋮ ⋮		⋮ ⋮ ⋮ ⋮	NFMAX レコード

１節点につき１レコードとし、１レコード中該当レベルの接触情報（可変）が記録される。

【改良】

以下のコモン変数に格納した。

/CNTINF/ 接触情報データブロック

MCONT(NODMAX)	該当レベルでの接触点数
MCLEVL(NODMAX)	該当レベル
INFOC1(MXC)	ピン番号
INFOC2(MXC)	接触方向
INFOC3(MXC)	接触情報テーブルのアドレス

●FFCP 107 (FIX) 強制変位ファイル

			ヘッダなし
NOFIX		強制変位条件番号	
NFLX		強制変位点数	2
FNP(1)		ピン番号	
FND(1)		節点番号	
FDG(1)		θ方向拘束フラグ	
FIX(1)		強制変位量	
DEG(1)		強制変位角度 (rad)	5
⋮		⋮	
FNP(NFIX)		ピン番号	
FND(NFIX)		節点番号	
FDG(NFIX)		θ方向拘束フラグ	
FLX(NFIX)		強制変位量	
DEG(NFIX)	EOR	強制変位角度 (rad)	5

【改良】

廃止（未使用のため）

●FFCP 108 (TMP) 温度増分ファイル

		ヘッダなし

TEMP(1,1,1)	温度増分 (R*4) IP：評価ポイント番号 LEV：レベル (節点) 番号 IPN：ピン番号	NSND ×NFMAX1 ×NPINT
: :		
TEMP(IP,LEV,IPN)		
: :		
TEMP(NSND,NFMAX1,NPINT)		
EOR		

評価点での温度増分を保持する。

1. NSND：評価点の数
2. NFMAX1：ピンあたりの節点数＋１
3. NPINT：ピン本数
4. IP(1～6)はピンの評価点
5. READS3/WRITSQ を用いる

【改良】

以下のコモン変数 (BAMTDP) に格納した。倍精度で保持するよう改良した。

/BMTEMP/ 温度データブロック

BAMTDW(NPNMAX,NODMAX+1)	ワイヤ温度増分 (R*8)
BAMTMW(NPNMAX,NODMAX+1)	ワイヤ温度 (R*8)
BAMTDD(NODMAX+1,6)	ダクト温度増分 (R*8)
BAMTMD(NODMAX+1,6)	ダクト温度 (R*8)
BAMTDP(NPNMAX,NODMAX+1,6)	ピン温度増分 (R*8) **…該当変数！**
BAMTMP(NPNMAX,NODMAX+1,6)	ピン温度 (R*8)

●FFCP 109 (LOD) 荷重乗数ファイル

	ヘッダなし

LTITER	THERMAL PHASE ステップ数
KCITER	CREEP PHASE ステップ数
MXITRT	THERMAL 最大収束回数
MXITRC	CREEP 最大収束回数
KLDBD	境界条件識別番号
FLDBD	境界条件への乗数
KLDTP	温度条件識別番号
FLDTP	温度条件への乗数
KLDPP	ピン内圧条件識別番号
FLDPP	ピン内圧条件への乗数
KLDDP	ダクト内圧条件識別番号
FLDDP	ダクト内圧条件への乗数
KLDNT	中性子束条件識別番号
FLDNT	中性子束条件への乗数
IOFLG	出力制御フラグ
CRTIME	クリープ時間増分
IPHFLG	室温フリー解析スイッチ
TMPIPH	室温フリー解析時の材料温度
⋮ ⋮ ⋮ ⋮	

18

18×NLOAD

1./DATA/ MLOD
2.LTITER,KCITER により各解析フェーズのループが制御される。
3.READS3/WRITSQ を用いる。

【改良】

以下のコモン変数に格納した。

/LODFID/ 荷重乗数データ（整数）ブロック

IDLOD(1,I)	THERMAL PHASE ステップ数
IDLOD(2,I)	CREEP PHASE ステップ数
IDLOD(3,I)	THERMAL 最大収束回数
IDLOD(4,I)	CREEP 最大収束回数
IDLOD(5,I)	境界条件識別番号
IDLOD(6,I)	温度条件識別番号
IDLOD(7,I)	ピン内圧条件識別番号
IDLOD(8,I)	ダクト内圧条件識別番号
IDLOD(9,I)	中性子束条件識別番号
IDLOD(10,I)	出力制御フラグ
IDLOD(11,I)	室温フリー解析スイッチ

/LODFRD/ 荷重乗数データ（実数）ブロック

RDLOD(1,I)	境界条件への乗数
RDLOD(2,I)	温度条件への乗数
RDLOD(3,I)	ピン内圧条件への乗数
RDLOD(4,I)	ダクト内圧条件への乗数
RDLOD(5,I)	中性子束条件への乗数
RDLOD(6,I)	クリープ時間増分
RDLOD(7,I)	室温フリー解析時の材料温度

●FFCP 110 (DDS) 変位増分ファイル

			ヘッダ未使用
DDSP(1,1,1) : : : DDSP(6,NFMAX,1)	EOR	変位増分 (R*8) IF：自由度番号 LEV：節点番号 IPN：ピン番号	6×NFMAX 1 ピン分
: : : DDSP(IF,LEV,IP) : : : DDSP(6,NFMAX,NPINT)	EOR		6×NFMAX×NPINT

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. 自由度 (Z,X,Y,TZ,TX,TY) ただし、Z はピン扁平化自由度に使用。TZ は未使用。

【改良】

以下のコモン変数 (DDSP) に格納した。

/DISP/ 変位データブロック

DSP(MAXDOF*NODMAX*NPNMAX)	ピンの変位ベクトル(R*8) (R*8)
DDSP(MAXDOF*NODMAX*NPNMAX)	ピンの変位増分ベクトル (兼) 荷重項 PLVEC (ピンの荷重増分ベクトル)

●FFCP 111,112 (FRC) 外荷重ベクトルファイル

			ヘッダ未使用
FRC(1,1,1) : : : FRC(6,NFMAX,1)	EOR	外荷重ベクトル (R*8) IF：自由度番号 LEV：節点番号 IPN：ピン番号	6×NFMAX 1 ピン分
: : : FRC(IF,LEV,IP) : : : FRC(6,NFMAX,NPINT)	EOR		6×NFMAX×NPINT

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. 自由度 (Z,X,Y,TZ,TX,TY) ただし、Z はピン扁平化自由度に使用。TZ は未使用。

【改良】

以下のコモン変数 (FRC) に格納した。

/FORC/ 荷重ベクトルデータブロック

FRC(MAXDOF*NODMAX*NPNMAX)	外荷重ベクトル(R*8)	**…該当変数！**
RES(MAXDOF*NODMAX*NPNMAX)	残差荷重ベクトル(R*8)	
CFC(MAXDOF*NODMAX*NPNMAX)	接触荷重ベクトル(R*8)	
TFRC(MAXDOF*NODMAX*NPNMAX)	熱膨張歪み等価荷重ベクトル (R*8)	

●FFCP 113 (RES) 残差荷重ベクトルファイル

			ヘッダ未使用
RES(1,1,1) : : : RES(6,NFMAX,1)	EOR	残差荷重ベクトル (R*8) IF：自由度番号 LEV：節点番号 IPN：ピン番号	6×NFMAX 1 ピン分
: : : RES(IF,LEV,IP) : : : RES(6,NFMAX,NPINT)	EOR		6×NFMAX×NPINT

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. 自由度 (Z,X,Y,TZ,TX,TY) ただし、Z はピン扁平化自由度に使用。TZ は未使用。

【改良】

以下のコモン変数 (RES) に格納した。

/FORC/ 荷重ベクトルデータブロック

FRC(MAXDOF*NODMAX*NPNMAX)	外荷重ベクトル(R*8)
RES(MAXDOF*NODMAX*NPNMAX)	残差荷重ベクトル(R*8) **…該当変数！**
CFC(MAXDOF*NODMAX*NPNMAX)	接触荷重ベクトル(R*8)
TFRC(MAXDOF*NODMAX*NPNMAX)	熱膨張歪み等価荷重ベクトル (R*8)

●FFCP 114 (CFC) 接触荷重ベクトルファイル

			ヘッダ未使用
CFC(1,1,1) : : : CFC(6,NFMAX,1)	EOR	接触荷重ベクトル (R*8) IF：自由度番号 LEV：節点番号 IPN：ピン番号	6×NFMAX 1 ピン分
: : : CFC(IF,LEV,IP) : : : CFC(6,NFMAX,NPINT)	EOR		6×NFMAX×NPINT

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. 自由度 (Z,X,Y,TZ,TX,TY) ただし、Z はピン扁平化自由度に使用。TZ は未使用。

【改良】

以下のコモン変数 (RES) に格納した。

/FORC/ 荷重ベクトルデータブロック

FRC(MAXDOF*NODMAX*NPNMAX)		外荷重ベクトル(R*8)
RES(MAXDOF*NODMAX*NPNMAX)		残差荷重ベクトル(R*8)
CFC(MAXDOF*NODMAX*NPNMAX)		接触荷重ベクトル(R*8) **…該当変数！**
TFRC(MAXDOF*NODMAX*NPNMAX)		熱膨張歪み等価荷重ベクトル (R*8)

●FFCP 115,116 (PIN) 接触判定用ピン径、ワイヤ径ファイル

		ヘッダ未使用
RPD(1,1,1)		
: :		
RPD(1,LEV,IPN) RPD(2,LEV,IPN) RPD(3,LEV,IPN) RPD(4,LEV,IPN) RPD(5,LEV,IPN) RPD(6,LEV,IPN) RPD(7,LEV,IPN)		熱によるピン径増分 スエリングによるピン径増分 内圧によるピン径増分 熱によるワイヤ径増分 スエリングによるワイヤ径増分 ピン径計算時の節点温度 ワイヤ径計算用の温度
: :		: :
RPD(7,NFMAX,NPINT)	EOR	7×NFMAX×NPINT

1.NFMAX : ピンあたり節点数
2.NPINT : ピン本数
3.READS3/WRITSQ を用いる

【改良】

以下のコモン変数に格納した。

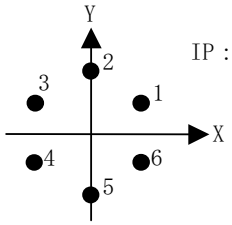
/RPDDAT/ 接触判定用ピン径、ワイヤ径データブロック

RPDWD(7*NODMAX*NPNMAX)	接触判定用ピン径、ワイヤ径データ(R*4) (1,*) : 熱によるピン径増分 (2,*) : スエリングによるピン径増分 (3,*) : 内圧によるピン径増分 (4,*) : 熱によるワイヤ径増分 (5,*) : スエリングによるワイヤ径増分 (6,*) : ピン径計算時の節点温度 (7,*) : ワイヤ径計算用の温度

●FFCP 117,118 (ECT) 熱クリープひずみファイル

			ヘッダ未使用

EPSTC(1,1,1,1,1) : : EPSTC(IC,IP,IBT,LEV,IPN) : : EPSTC(6,6,2,NFMAX,NPINT)	EOR	熱クリープひずみ (R*8) IC：コンポーネント番号 IP：評価ポイント番号 IBT：i,j 端番号 LEV：レベル (節点) 番号 IPN：ピン番号	6×6×2 ×NFMAX×NPINT



IP：評価点の位置

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. READS3/WRITSQ を用いる
4. IC：ひずみのコンポーネント
   1：熱クリープひずみの積算量
   2：0.0
   3：0.0
   4：熱クリープひずみ増分
   5：0.0
   6：0.0

【改良】

現在の BAMBOO コードでは、熱クリープひずみは取り扱っていないが、今後を見込み FFCP に代わる記憶域として、以下のコモン変数 (EPSTC) に格納した。

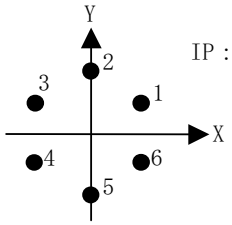
/STRCRP/ 応力、熱・照射クリープひずみデータブロック

EPSSC(6,2*6*NODMAX*NPNMAX)	照射クリープひずみデータ(R*8) (1,*)：照射クリープひずみの積算量 (2,*)：ひずみ増分 (STR(3)) (3,*)：0.0 (4,*)：照射クリープひずみ増分 (5,*)：応力 (SIGX) (6,*)：照射量 (FTN)
EPSTC(6,2*6*NODMAX*NPNMAX)	熱クリープひずみデータ(R*8)　　**…該当変数！** (1,*)：熱クリープひずみの積算量 (2,*)：0.0 (3,*)：0.0 (4,*)：熱クリープひずみ増分 (5,*)：0.0 (6,*)：0.0
STRES(4,2*6*NODMAX*NPNMAX)	応力データ(R*8) (1,*)：応力 (2,*)：0.0 (3,*)：全体ひずみ-初期ひずみ (4,*)：応力の絶対値

●FFCP 119,120 (ECS) 照射クリープひずみファイル

			ヘッダ未使用

EPSSC(1,1,1,1,1) : : EPSSC(IC,IP,IBT,LEV,IPN) : : EPSSC(6,6,2,NFMAX,NPINT)	EOR	照射クリープひずみ (R*8) IC：コンポーネント番号 IP：評価ポイント番号 IBT：i,j 端番号 LEV：レベル (節点) 番号 IPN：ピン番号	6×6×2 ×NFMAX×NPINT



IP：評価点の位置

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. READS3/WRITSQ を用いる
4. IC：ひずみのコンポーネント
   1：照射クリープひずみの積算量
   2：ひずみ増分 (STR(3))
   3：0.0
   4：照射クリープひずみ増分
   5：応力 (SIGX)
   6：照射量 (FTN)

【改良】

以下のコモン変数 (EPSSC) に格納した。

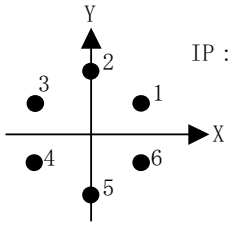
/STRCRP/ 応力、熱・照射クリープひずみデータブロック

EPSSC(6,2*6*NODMAX*NPNMAX)	照射クリープひずみデータ(R*8)　　**…該当変数！** (1,*)：照射クリープひずみの積算量 (2,*)：ひずみ増分 (STR(3)) (3,*)：0.0 (4,*)：照射クリープひずみ増分 (5,*)：応力 (SIGX) (6,*)：照射量 (FTN)
EPSTC(6,2*6*NODMAX*NPNMAX)	熱クリープひずみデータ(R*8) (1,*)：熱クリープひずみの積算量 (2,*)：0.0 (3,*)：0.0 (4,*)：熱クリープひずみ増分 (5,*)：0.0 (6,*)：0.0
STRES(4,2*6*NODMAX*NPNMAX)	応力データ(R*8) (1,*)：応力 (2,*)：0.0 (3,*)：全体ひずみ-初期ひずみ (4,*)：応力の絶対値

●FFCP 121,122 (SWL) スエリング量ファイル

	ヘッダ未使用

VSWLP(1,1,1,1) VFTN(1,1,1,1)		スエリング量 (R*8) 照射量 (R*8)
: :		
VSWLP(IP,IBT,LEV,IPN) VFTN(IP,IBT,LEV,IPN)		IP：評価ポイント番号 IBT：i,j 端番号 LEV：レベル (節点) 番号 IPN：ピン番号
: :		
VSWLP(7,2,NFMAX,NPINT) VFTN(7,2,NFMAX,NPINT)	EOR	2×7×2 ×NFMAX×NPINT



IP：評価点の位置

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. READS3/WRITSQ を用いる
4. IP(1～6)はピンの評価点、IP(7)はワイヤのスエリング量

【改良】

以下のコモン変数 (SWLPV, SWLWV, FTNP, FTNW) に格納した。

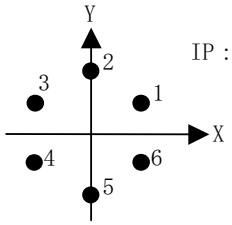
/ STRSWL / スエリング、照射量データブロック

DSWLPV(2*6*NODMAX*NPNMAX)	ピンスエリング量増分(R*8)	
SWLPV(2*6*NODMAX*NPNMAX)	ピンスエリング量 (R*8)	**…該当変数！**
SWLWV(2*NODMAX*NPNMAX)	ワイヤスエリング量(R*8)	**…該当変数！**
FTNP(2*6*NODMAX*NPNMAX)	ピン照射量(R*8)	**…該当変数！**
FTNW(2*NODMAX*NPNMAX)	ワイヤ照射量(R*8)	**…該当変数！**

●FFCP 123 (STR) 応力ファイル

	ヘッダ未使用

STRS(1,1,1,1,1) : : STRS(IC,IP,IBT,LEV,IPN) : : STRS(4,6,2,NFMAX,NPINT)	EOR	応力 (R*8) IC：コンポーネント番号 IP：評価ポイント番号 IBT：i,j 端番号 LEV：レベル (節点) 番号 IPN：ピン番号	4×6×2 ×NFMAX×NPINT



IP：評価点の位置

1. NFMAX：ピンあたり節点数
2. NPINT：ピン本数
3. READS3/WRITSQ を用いる
4. IC：ひずみのコンポーネント
   1：応力
   2：0.0
   3：全体ひずみ-初期ひずみ
   4：応力の絶対値

【改良】

以下のコモン変数 (STRES) に格納した。

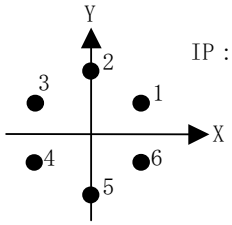
/STRCRP/ 応力、熱・照射クリープひずみデータブロック

EPSSC(6,2*6*NODMAX*NPNMAX)	照射クリープひずみデータ(R*8) (1,*)：照射クリープひずみの積算量 (2,*)：ひずみ増分 (STR(3)) (3,*)：0.0 (4,*)：照射クリープひずみ増分 (5,*)：応力 (SIGX) (6,*)：照射量 (FTN)
EPSTC(6,2*6*NODMAX*NPNMAX)	熱クリープひずみデータ(R*8) (1,*)：熱クリープひずみの積算量 (2,*)：0.0 (3,*)：0.0 (4,*)：熱クリープひずみ増分 (5,*)：0.0 (6,*)：0.0
STRES(4,2*6*NODMAX*NPNMAX)	応力データ(R*8)　　**…該当変数！** (1,*)：応力 (2,*)：0.0 (3,*)：全体ひずみ-初期ひずみ (4,*)：応力の絶対値

●FFCP 124,125 (STP) 評価点温度ファイル

		ヘッダなし
STEMP(1,1,1) : : STEMP(IP,LEV,IPN) : : STEMP(6,NFMAX1,NPINT)	温度 (R*4) IP：評価ポイント番号 LEV：レベル (節点) 番号 IPN：ピン番号	6×NFMAX1×NPINT

EOR



IP：評価点の位置

1. IP(1～6)はピンの評価点
2. NFMAX1：ピンあたりの節点数＋１
3. NPINT：ピン本数
4. STCOND/LDTMP でセットされる
5. READS3/WRITSQ を用いる

【改良】

以下のコモン変数 (BAMTMP) に格納した。倍精度で保持するよう改良した。

/BMTEMP/ 温度データブロック

BAMTDW(NPNMAX,NODMAX+1)	ワイヤ温度増分 (R*8)
BAMTMW(NPNMAX,NODMAX+1)	ワイヤ温度 (R*8)
BAMTDD(NODMAX+1,6)	ダクト温度増分 (R*8)
BAMTMD(NODMAX+1,6)	ダクト温度 (R*8)
BAMTDP(NPNMAX,NODMAX+1,6)	ピン温度増分 (R*8)
BAMTMP(NPNMAX,NODMAX+1,6)	ピン温度 (R*8) **…該当変数！**

●FFCP 126,127 (FLX) 中性子束ファイル

		ヘッダなし
FLX(1,1,1)	中性子束 φ (R*4)	2×NFMAX×NPINT
:		
:	IB : i,j 端番号	
FLX(IB,LEV,IPN)	LEV : レベル (節点) 番号	
:	IPN : ピン番号	
:		
FLX(2,NFMAX,NPINT) EOR		

1. 前ステップでの φ の分布を保持する。(要素節点位置)
2. NFMAX : ピンあたりの節点数
3. NPINT : ピン本数
4. READS3/WRITSQ を用いる

●FFCP 128 (DFX) 中性子束分布 (増分) ファイル

		ヘッダなし
FLUX(1,1,1)	中性子束分布 φ (R*4)	2×NFMAX×NPINT
:	(増分量)	
:		
FLUX(IB,LEV,IPN)	IB : i,j 端番号	
:	LEV : レベル (節点) 番号	
:	IPN : ピン番号	
FLUX(2,NFMAX,NPINT) EOR		

1. 中性子束分布は、相対的な分布値を設定する。(要素節点位置)
2. 実際の値は、φt=FLUX × FLXMAX × time で与えられる。FLXMAX は、INTERVAL で与える倍率。
3. READS3/WRITSQ を用いる。

【改良】

以下のコモン変数にそれぞれ格納した。

/ PNFLUX / ピン FLUX データブロック

FLUXP(2*NODMAX*NPNMAX)	FLUX データ (R*4)
DFLUXP(2*NODMAX*NPNMAX)	FLUX 増分データ (R*4)

●FFCP 129 (DUCT) ダクトファイル

NBS		ダクト１スパンの分割数	
KYDUCT		ダクトの扱いのオプション	
NFMAX		ピンあたりの節点数	
			ヘッダ
RLFI		初期内対面距離 (R*4)	
RTI	EOR	初期板厚 (R*4)	2
DFLX(1,1) : DFLX(LEX,IP) : DFLX(NFMAX,6)	EOR	中性子束分布データ (R*4) IP：ダクトパネル番号 LEV：レベル（節点）番号	6×NFMAX
FLX(1,1) : FLX(LEX,IP) : FLX(NFMAX,6)	EOR	中性子束データ (R*4) IP：ダクトパネル番号 LEV：レベル（節点）番号	6×NFMAX
FTN(1,1) : FTN(LEX,IP) : FTN(NFMAX,6)	EOR	照射量データ (R*4) IP：ダクトパネル番号 LEV：レベル（節点）番号	6×NFMAX
SWL(1,1) : SWL(LEX,IP) : SWL(NFMAX,6)	EOR	スエリング量（ΔV/V %）(R*4) IP：ダクトパネル番号 LEV：レベル（節点）番号	6×NFMAX
DDTMP(1,1) : DDTMP(LEX,IP) : DDTMP(NFMAX,6)	EOR	温度増分（℃）(R*4) IP：ダクトパネル番号 LEV：レベル（節点）番号	6×NFMAX
DTTMP(1,1) : DTTMP(LEX,IP) : DTTMP(NFMAX,6)	EOR	温度（℃）(R*4) IP：ダクトパネル番号 LEV：レベル（節点）番号	6×NFMAX
DDCT(1,1,1) : : : DDCT(ID,IB,IP) : : : DDCT(4,NBS+1,6)	EOR	ダクト変形量 (R*4) ID：ダクト変形要因 =1:スエリングによる変形 =2:内圧による変形 =3:熱膨張による変形 =4:BDI 荷重による変形 IB:1 スパンでの評価点 (NBS+1) IP:ダクトパネル番号	(1 レベルあたり) 4×(NBS+1) ×6
: : : :		（ダクト変形量） レベル NFMAX まで繰り返し	

1. FLX は、前ステップでの中性子束分布 ϕ を保持する。(ダクト評価レベル)
2. DFLX は、入力された中性子束分布。(ダクト評価レベル)

【改良】

以下のコモン変数に格納した。一部の変数（初期内対面距離、初期板厚、ダクト変形量、ダクト温度、ダクト温度増分）を倍精度で保持するよう改良した。

/DCTDAT/ ダクト照射条件データブロック

DCTDFX(6,NODMAX+1)	ダクト FLUX 増分データ (R*4)
DCTFLX(6,NODMAX+1)	ダクト FLUX データ (R*4)
DCTFTN(6,NODMAX+1)	ダクト照射量データ (R*4)
DCTSWL(6,NODMAX+1)	ダクトスエリングデータ (R*4)

/DCTDEF/ ダクト変形データブロック

RLFI	ダクト初期内対面距離 (R*8)
RTI	ダクト初期板厚 (R*8)
DDFM(4,NBSMAX,6,NODMAX+1)	ダクト変形量 (R*8) (1,*):スエリングによる変形 (2,*):内圧による変形 (3,*):熱膨張による変形 (4,*):BDI 荷重による変形

/DCTHED/ ダクトデータ管理変数ブロック

IHEDCT(6)	ダクトヘッダ (1):ダクト１スパンの分割数 (2):ダクトの扱いのオプション (3):ピンあたりの節点数 (4):未使用 (5):未使用 (6):未使用

/BMTEMP/ 温度データブロック

BAMTDW(NPNMAX,NODMAX+1)	ワイヤ温度増分 (R*8)
BAMTMW(NPNMAX,NODMAX+1)	ワイヤ温度 (R*8)
BAMTDD(NODMAX+1,6)	ダクト温度増分 (R*8) **…該当変数！**
BAMTMD(NODMAX+1,6)	ダクト温度 (R*8) **…該当変数！**
BAMTDP(NPNMAX,NODMAX+1,6)	ピン温度増分 (R*8)
BAMTMP(NPNMAX,NODMAX+1,6)	ピン温度 (R*8)

接触力求解ソルバー ～準備部～

接触力増分Δλは、以下の方程式より計算される。

$$[Kcc]=[BC]^T\cdot[K]^{-1}\cdot[BC] \quad \cdots(1)$$

$$\{\Delta\lambda\}=[Kcc]^{-1}\cdot\{\Delta\delta\} \quad \cdots(2)$$

ここで、[K]は剛性マトリクス、[BC]は接触状態マトリクス、{Δλ}は接触力増分ベクトル、{Δδ}はオーバーラップベクトルである。

準備部では、この演算のうち、リナンバーアルゴリズムによる[BC]マトリクス最適化処理と接触マトリクス[kcc]の計算（(1)式）までを行う。

また、下記の改良内容で［L1BC]として記述するマトリクスは、[Kcc]マトリクスを計算するための処理途中の行列であり、以下のように定義される。

$$[L1BC]=\left[\sqrt{D}\right]^{-1}\cdot[L]^{-1}\cdot[BC] \quad \cdots(3)$$

ここで、[L]は剛性マトリクスの左下三角マトリクス、[D]は剛性マトリクスの対角マトリクスである。

●FFCP 130（IWOKF）スワップデータファイル

		ヘッダなし
IDATA(1) : : IDATA(NWD)	接触点情報テーブル （ICNTTB,ICNHIS,CFX）	NWD (=LCTB+LCTH+LCFC)

接触力求解ソルバーで使用するワーク領域の空き領域を確保ため、ワーク領域に記録されているデータを一時的に退避するために用いる。

1.LCTB=6×MXCONT
2.LCTH=6×NPINT
3.LCFC=(6×NFMAX×NPINT)×2
4.MXCONT：最大接触点数（=60000）
5.NPINT：ピン本数
6.NFMAX：ピンあたり節点数

【改良】

以下のコモン変数に格納した。

/SWAP/ スワップデータブロック

ISWAP(NSWAP) (NSWAP=6*MXC+6*NPNMAX+2*6*NPNMAX*NODMAX)	接触点情報テーブル （ICNTTB,ICNHIS,CFX）退避用データ

●FFCP 301 (N301) 接触力求解ソルバーワークファイル

	ヘッダなし

NN	カラム１の active ピン本数
IAC(2)	カラム１の active ピン番号 3
:	:
:	:
:	:
NN	カラム n の active ピン本数
IAC(2)	カラム n の active ピン番号 3×NDOFC

[BC]マトリクスの非ゼロ要素（active Pin No.）をカラム毎に整理する。[kcc]マトリクスの計算に使用する。

1.NDOFC：接触点数

【改良】

ワーク領域に以下の領域を確保した。

$$[4byte]\times(3\times NDOFC)$$

●FFCP 305 (L305) リナンバー情報ベクトルファイル

	ヘッダなし

NEWSEQ(1)	1 番目に処理される[BC]番号
NEWSEQ(2)	2 番目に処理される[BC]番号
:	:
:	:
:	:
NEWSEQ(NDOFC)	NDOFC 番目に処理される[BC]番号 NDOFC

リナンバーアルゴリズムにより求まった情報（[BC]マトリクスの並べ替え順）を、FFCP 305 に書き込む。

1.NDOFC：接触点数

【改良】

ワーク領域に以下の領域を確保した。

$$[4byte]\times NDOFC$$

●FFCP 302 (N302) [kcc]マトリクスファイル

LU 分解をウェーブフロント法で行うために､PACK ルーチンを用いて[kcc]マトリクスの非ゼロ成分のみ記憶（圧縮）する。他の FFCP ファイルと異なり、FFCP の PUTSTR 機能および GETSTR 機能を用いてデータ管理を行っていた。

【改良】

ワーク領域に以下の領域を確保した。

[kcc]マトリクスデータ領域

$$[8byte]\times\left(\frac{NDOFC\times(NDOFC+1)}{2}\right)$$

[kcc]マトリクス管理変数領域

$$[4byte]\times\frac{3}{4}\times(NDOFC\times(NDOFC+1))$$

ここで NDOFC は接触点数である。

[kcc]マトリクスデータ領域は、圧縮前のマトリクスを記憶できる大きさを確保した。

[kcc]マトリクス管理変数は、圧縮後データが圧縮前のマトリクスのどの部分に相当するかなどの情報を記録する変数である。

【管理変数】

NKCC(1,I)：アクティブな成分の開始行番号
NKCC(2,I)：連続しているアクティブ成分数
NKCC(3,I)：NKCC(1,I)の成分が格納されている圧縮配列のアドレス
（各カラムの終端を表すフラグとしても用いる。-1 を入力）
（I=1～管理個数）

(例)右のような[kcc]マトリクスを配列 P に圧縮する。

ここで、”*”はアクティブ成分である。
カラム番号を i、行番号を j としたときの
[kcc]マトリクス成分を a(i,j)とおく。

１カラム目から順にアクティブ成分のみを配列 P に格納する。

P(1) = a(1,1)
P(2) = a(1,2)
P(3) = a(1,3)
P(4) = a(1,6)
P(5) = a(1,9)
P(6) = a(2,2)
P(7) = a(2,3)
P(8) = a(2,6)
P(9) = a(2,9)
:
:
P(N) = a(24,24)

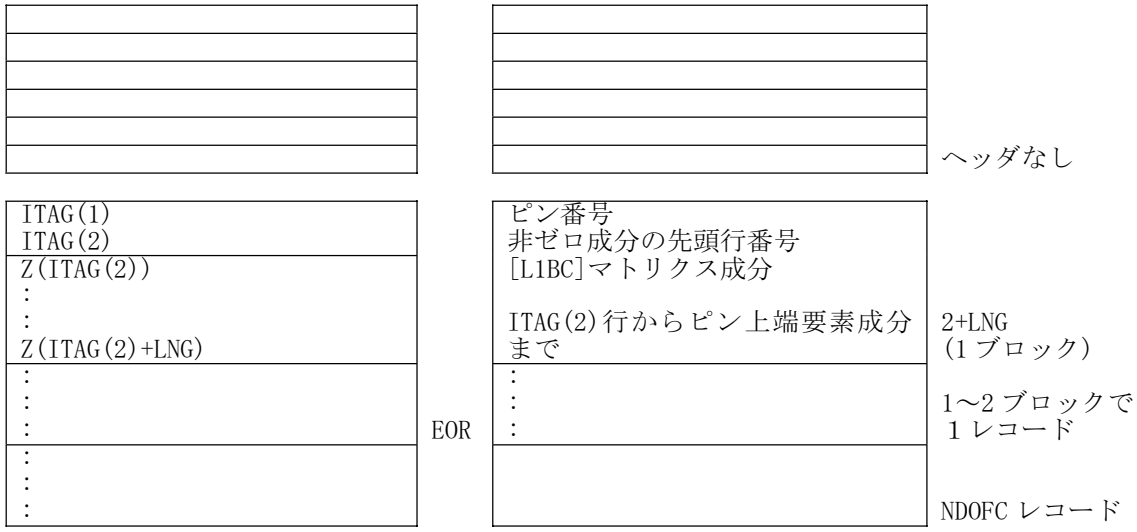
【カラム番号】→

【行番号】↓

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	*																							
2	*	*																						
3	*	*	*																					
4	.	.	*	*																				
5	.	.	*	*	*																			
6	*	*	*	.	.	*																		
7	.	.	.	.	.	*	*																	
8	.	.	.	.	.	*	*	*																
9	*	*	*	.	.	*	.	.	*															
10	.	.	*	*	*	.	.	.	*	*														
11	.	.	.	.	.	*	*	*	*	*	*													
12	.	.	*	*	*	.	.	.	*	.	*													
13	.	.	.	.	.	.	.	.	.	.	.	*	*											
14	.	.	.	.	.	.	.	.	.	.	.	*	*	*										
15	.	.	.	.	.	.	.	.	*	*	*	*	*	*	*									
16	.	.	.	.	*	*	*	.	.	*	.	.	.	.	.	*								
17	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	*	*							
18	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	*	*	*						
19	.	.	.	.	.	.	.	.	*	*	*	.	.	.	*	*	*	*	*					
20	.	.	.	.	.	.	.	.	.	.	.	*	*	*	*	.	.	.	.	*				
21	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	*	*			
22	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	*	*	*		
23	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	*	*	*	*	*	*	*	*	
24	.	.	.	.	.	.	.	.	*	*	*	.	.	.	*	.	.	.	*	*	*	*	*	*

管理番号I	NKCC(1, I)	NKCC(2, I)	NKCC(3, I)
1	1	3	1
2	6	1	4
3	9	1	5
4	0	0	-1
5	2	2	6
6	6	1	8
7	9	1	9
8	0	0	-1
:	:	:	:
:	:	:	:

●FFCP 303 (N303) 接触力求解ソルバーワークファイル



[L1BC]マトリクスを格納する領域である。

1. NDOFC：接触点数
2. LNG：0～6×NNOD
3. NNOD：ピンあたり節点数

【改良】

ワーク領域に以下の領域を確保した。

[L1BC]マトリクス用領域

$$[8byte]\times(6\times NNOD\times 2\times NDOFC)$$

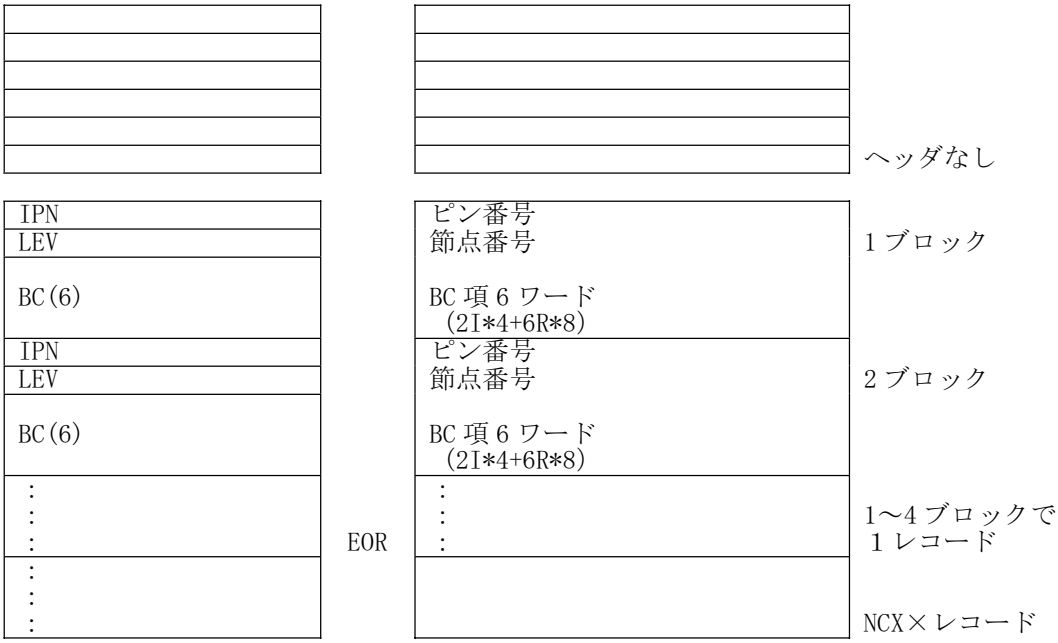
[L1BC]マトリクス管理変数領域

$$[4byte]\times(3\times 2\times NDOFC)$$

【管理変数】

NL1BC(1,I)：ピン番号
NL1BC(2,I)：非ゼロ成分の先頭行番号
NL1BC(3,I)：カラム番号

●FFCP 304 (L304) リナンバー後[BC]マトリクスファイル



FFCP 104(BCX)のデータをリナンバリングし、FFCP 304 に書き込む。

【改良】

[BC]マトリクスは、下記の「/MTBCN/」と「/MTBCF/」に記録している。このうち、[BC]マトリクス管理変数「NBCLBL」をリナンバリングすることにより最適化する。

/MTBCN/ BC マトリクス管理変数データブロック

NBCHED(6)	BC マトリクスヘッダ 1:ピン本数 2:ピンあたり節点数 3:節点あたり自由度（６） 4:接触点数 5:未使用 6:未使用
NBCLBL(2,MXC)	BC マトリクス管理変数　　**…該当変数！** (1,*):BC マトリクス K の開始アドレス (2,*):BC マトリクス K の終了アドレス
NBCTAG(2,MXC*4)	BC マトリクスのタグ (1,*):ピン番号（タグ１） (2,*):節点番号（タグ２）

接触力求解ソルバー ～求解部～

求解部では、準備部で計算した[kcc]マトリクス（(1)式）の LU 分解を行い、接触力増分{Δλ}を求解する。

●FFCP 301,303 (SCRTCH) スワップデータファイル

[Kcc]マトリクスは、接触点数によってその大きさが決まる。接触点数が N のとき、[Kcc]マトリクスは N×N の対称マトリクスとなる。[Kcc]マトリクスの LU 分解は、ウェーブフロント法で行うが、接触点数が N のとき、次に示す作業領域（メモリー）を必要とする。

- AC：処理対象カラムのアクティブな行番号
  → $[4byte]\times N$
- ACX：処理対象カラム以前にアクティブだった行番号
  → $[4byte]\times N$
- P：処理対象カラムの[Kcc]マトリクス成分
  → $[8byte]\times N$
- W：LU 分解に必要な一時変数
  → $[8byte]\times\left[\frac{N\times(N-1)}{2}\right]$

N=5000 で約 100MByte となる。この作業領域は、メモリーから確保するが、使用可能なメモリーが少ない場合、作業領域のデータを FFCP によりハードディスクに退避させていた。この退避させたデータがスワップデータファイルとなる。

【改良】

ワーク領域に十分なメモリーを確保できるようになったため、スワップデータファイルは廃止した。ワーク領域は、PARAMETER 文の MXS 値で決まる。MXS 値を変更するには「PARAMW.h」を編集し、再コンパイルが必要である。

[Kcc]マトリクスの LU 分解では、上記の作業領域の他に

- [Kcc]マトリクスデータ記憶領域
- [Kcc]マトリクス管理変数領域
- [Kcc]マトリクスの[L][D]マトリクスデータ記憶領域
- [Kcc]マトリクスの[L][D]マトリクス管理変数領域

が必要であるため、ワーク領域はこれらを全て格納できるメモリーが必要となる。
以下に、接触点数とメモリー容量の関係を示す。

消費メモリー	接触点数	MXS 値	備考
100 Mbyte	2000	25000000	
500 Mbyte	5000	125000000	
1 Gbyte	**7000**	**250000000**	**（現在設定値）**
2 Gbyte	10000	500000000	
3 Gbyte	13000	750000000	
4 Gbyte	15000	1000000000	

表より、7000 点の接触点が発生する場合、接触力を求解するためには約 1Gbyte のメモリーが必要であることがわかる。

●FFCP 304 (LUNL) [Kcc]の三角化マトリクスファイル

[Kcc]マトリクスの左下三角マトリクス[L]と対角マトリクス[D]を格納するための領域。[Kcc]マトリクスと同様に、FFCP の PUTSTR 機能および GETSTR 機能を用いてデータ管理を行っていた。

【改良】

ワーク領域に以下の領域を確保した。

[kcc]マトリクスの[L][D]データ領域

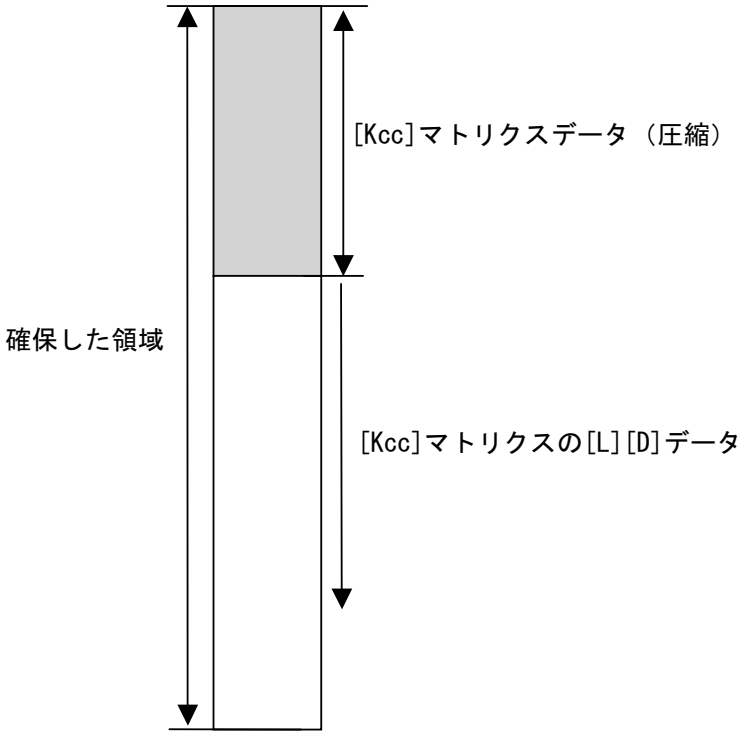
$$[8byte]\times\left(\frac{NDOFC\times(NDOFC+1)}{2}\right)$$

[kcc]および[kcc]の[L][D] 管理変数領域

$$[4byte]\times\frac{3}{4}\times(NDOFC\times(NDOFC+1))$$

ここで NDOFC は接触点数である。

これらの領域は、[kcc]マトリクスと共用する。



変位求解ソルバー

変位求解ソルバーでは、ピン剛性マトリクスと節点荷重ベクトルにより、ピンの変形を解析する。

$$[K]\cdot\{\Delta\mu\}=\{FRC\}+\{CFC\}-\{RES\}=\{\Delta P\}$$

ここで、[K]は剛性マトリクス、{Δμ}は変位増分ベクトル、{FRC}は外荷重ベクトル、{CFC}は接触荷重ベクトル、{RES}は残差荷重ベクトルである。

変位求解手順は、まず剛性マトリクス[K]を修正コレスキー分解により、[L]（左下三角マトリクス）[D]対角マトリクス[U]（右上三角マトリクス）に分解する。

$$[K]=[L]\cdot[D]\cdot[U]$$

このうちの[L]と[D]を格納したファイルが、三角化マトリクス（FFCP103）である。

次の手順では、[L]と[D]を用い、右辺ベクトル（{ΔP}）に対し、前進消去・後退代入を行って{Δμ}を得る。

【前進消去】

$$[L]\cdot\{y\}=\{\Delta P\}$$

【後退代入】

$$[D]\cdot[L]^T\cdot\{\Delta\mu\}=\{y\}$$

【改良】

●FFCP 130（GSTF）強制変位に関するデータ
●FFCP 301（N301）強制変位に関するデータ
●FFCP 302,303（ISCR）強制変位に関するデータ

これらのデータは、BAMBOO コード開発初期に導入された強制変位に関するデータと思われる。調査の結果、上記の変位増分求解過程では使用されないデータであったため削除した。

その他、剛性マトリクスファイル（FFCP 101）、荷重ベクトルファイル（FFCP 102）、三角化ファイル（FFCP 103）、変位増分ファイル（FFCP 110）が変位求解ソルバーの改良に関係する。