DESENVOLVIMENTO DE UMA INTERFACE AMIGÁVEL PARA OBTENÇÃO E ANÁLISE DE TERMO FONTE PARA REATORES PWR USANDO O ORIGEN2.1 E O SOFTWARE MATLAB

João Paulo Macena Muniz Vieira

Orientador: Prof. Dra. Maria Auxiliadora Fortini Veloso.
Co-Orientador: Prof. Dra. Cláubia Pereira Bezerra Lima.
Área de concentração: Engenharia Nuclear e da Energia.

UNIVERSIDADE FEDERAL DE MINAS GERAIS

ESCOLA DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA NUCLEAR

DESENVOLVIMENTO DE UMA INTERFACE AMIGÁVEL PARA OBTENÇÃO E ANÁLISE DE TERMO FONTE PARA REATORES PWR USANDO O ORIGEN2.1 E O SOFTWARE MATLAB.

Dissertação apresentada ao curso de Ciências e Técnicas Nucleares da Escola de Engenharia da Universidade Federal de Minas Gerais como requisitos parcial à obtenção do Título de mestre em Ciências e Técnicas Nucleares.

Área de concentração: Engenharia Nuclear e de Energia

Orientador: Profa. Dra. Maria Auxiliadora Fortini Veloso

Co-Orientadora: Profa. Dra. Cláubia Pereira Bezerra Lima

João Paulo Macena Muniz Vieira

Belo Horizonte

Agosto de 2012

V658d Vieira, João Paulo Macena Muniz
Desenvolvimento de uma interface amigável para obtenção e análise de termo fonte para reatores PWR usando o ORIGEN2.1 e o software MATLAB / João Paulo Macena Muniz Vieira .— 2012
ix, 60f., enc: il.

Orientadora: Maria Auxiliadora Fortini Veloso.

Coorientadora: Cláubia Pereira Bezerra Lima.

Dissertação (mestrado) – Universidade Federal de Minas Gerais, Escola de Engenharia.

Bibliografia: f. 59-60.

1. Engenharia nuclear - Teses. 2. Reatores nucleares - Teses. 3. MATLAB (Programa de computador) – Teses. I. Veloso, Maria Auxiliadora Fortini. II. Lima, Cláubia Pereira Bezerra III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 621.039.4(043)

UFMG
UNIVERSIDADE FEDERAL DE MINAS GERAIS

Programa de Pós-Graduação em Ciências e Técnicas Nucleares–PCTN
Departamento de Engenharia Nuclear – Bloco 4 - Escola de Engenharia
Av. Antônio Carlos, 6627 – Campus Pampulha
Belo Horizonte-MG – CEP: 31.270-901
E-mail: poscctn@nuclear.ufmg.br Fone: (31) 3409-6666

O resultado final foi comunicado publicamente ao candidato pela Presidente da Comissão. Nada mais havendo a tratar, a Presidente encerrou a reunião e lavrou a presente ATA, que será assinada por todos os membros participantes da Comissão Examinadora. Belo Horizonte, 31 de agosto de 2012.

Presidente da Comissão: Maria Auxiliadora Fortini Veloso

Profa. Dra. Maria Auxiliadora Fortini Veloso – orientadora

Profa. Dra. Cláubia Pereira Bezerra Lima – coorientadora

Profa. Dra. Antonella Lombardi Costa

Dr. Clarysson Alberto Mello da Silva

Dr. Rômulo Verdolin de Souza

AGRADECIMENTOS

Agradeço primeiro à minha esposa pelas longas horas dedicadas em paciência à duração deste mestrado. E por ter tornado isso possível com seu apoio incondicional.

À minha mãe Vera, a quem devo a existência e que ainda sente algumas reflexões pesarosas ao se lembrar de minha infância.

Ao meu pai Carlos que mesmo separado por alguns anos me põe a carregar o ensinamento do perdão e compreensão. Ao meu padrasto Carlinhos pela leveza que me ensinou, embora ele já não a possua.

Aos meus irmãos por escolha Dinamar e Rafael Amparo. Por estarem sempre à minha disposição em horas turbulentas e sombrias e em outras mais serenas e ensolaradas.

Aos meus somativos irmãos Flávia, Mariana, Vinícius, Júnior, Lucas, Lívia e Saulo pelas diversas saudades sentidas.

Aos meus tios Toninho, Lili e Maúrea pela receptibilidade e carinho.

Aos meus amigos Bruno e Fabi pelas horas de descontração, conselhos e incentivos.

Ao meu amigo Abraão por tolerar-me e também à Cidiney e Jésus pelo apoio técnico e consultoria em dúvidas frequentes.

À Profa. Dra. Cláubia Pereira Bezerra Lima pela co-orientação, confiança, apoio e estímulo.

À Profa. Dra. Maria Auxiliadora Fortini pela orientação, por ter sido paciente, encorajadora e operar com justiça.

Ao Departamento de Engenharia Nuclear da UFMG, Profª Ângela , Profª Antonella, Bernadete, Nanci,Vanderlei e aos amigos de trabalho Mario, Mauricio, Carla e Reginaldo.

Ao CNPq, CAPES e FAPEMIG pelo suporte financeiro.

À Deus por ter mantido as leis físicas constantes até o fim deste projeto e razoavelmente compreensíveis à nossa capacidade de resumi-las.

E finalmente eu agradeço aos problemas perpétuos pela glória de superá-los.

RESUMO

Na ocorrência de uma liberação acidental de material radioativo o termo fonte, ou seja, a composição isotópica disponibilizada para o ambiente, deve ser conhecida o quanto antes para suprir a necessidade de ações mitigadoras. Assim, um acesso rápido, claro e seguro à toda informação vinculada ao material liberado é necessário. Boa parte dos códigos e programas que calculam o termo fonte não o fazem em tempo hábil para a tomada de decisões em situação emergêncial.

Este trabalho descreve a iniciativa de desenvolver uma ferramenta em MATLAB que através do ORIGEN 2.1 calcule o termo fonte e suas características utilizando poucos recursos computacionais e consequentemente tempo de execução, essa ferramenta é chamada de TFORI. Para alcançar este objetivo o usuário tem a possibilidade de definir células em qualquer quantidade num espaço tridimensional e executá-las, sendo que cada célula terá sua própria saída. Isto é feito a fim de se obter maiores detalhes sobre uma região e reduzir os gastos computacionais envolvidos.

Para calcular o termo fonte o usuário deverá conhecer os fatores de distribuição de pico de potência radial e axial de cada célula, além de possuir a biblioteca apropriada para o caso estudado e o histórico de operação do reator desejado. A validação do programa TFORI foi feita por comparação entre os valores obtidos por ele num caso estudado e os valores obtidos pelo ORIGEN2.1 para o mesmo caso. Não apresentou-se perda na qualidade das informações fornecidas pelo TFORI. A sua saída ainda foi avaliada comparando o comportamento de nuclídeos já conhecidos com a literatura.

Por fim, um estudo de caso utilizando o programa TFORI foi realizado e seus dados comparados com uma simulação que utilizava outro método. A diferença entre os resultados se deve a uma aproximação feita para para o fator de distribuição de pico de potência axial por uma curva senoidal e naturalmente por serem métodos diferentes de cálculo.

Futuramente, espera-se a implementação da possibilidade dos resultados serem reaproveitados para que se faça uma extensão no período de decaimento já que o ORIGEN 2.1 é capaz de fazer essa realimentação de dados.

ABSTRACT

In the event of an accidental release of radioactive material the source term, i.e., the isotopic composition released to the environment, should be known as soon as possible to meet the need of mitigating actions. So, a quick, clear and safe to all information linked to material released is necessary. Much of codes and programs that calculate the source term do not do so in time for decision making in an emergency situation.

This paper describes the initiative to develop a tool in MATLAB that through ORIGEN 2.1 calculate the source term and its features using few computational resources and hence runtime, this tool is called TFORI. To accomplish this, the user has the possibility to define cells in any quantity in a three dimensional space and run them, each cell will have its own output. This is done in order to obtain more information about a region and reduce computational costs involved.

To calculate the source term the user must know the factors of distribution of power radial and axial of each cell, as well as having the appropriate library for the case study and the history of reactor operation desired. The validation of the program TFORI was made by comparing the values obtained by it in a case study and the values obtained by ORIGEN2.1 for the same case. There are not losses in the quality of information provided by TFORI. The output was also evaluated by comparing the behavior of nuclides well known from the literature.

Finally, a case study using the program TFORI was performed and compared their data with a simulation that used another method. The difference between the results are due to an approximation made to the distribution factor for the axial power by a sine curve and being naturally different methods of calculation.

In the future, it is expected the possibility of implementing the results are reused for to achieve an extension in the period of decay as the ORIGEN 2.1 is able to make this feedback data.

ÍNDICE

ÍNDICE DE TABELAS

ÍNDICE DE FIGURAS

ÍNDICE DE SIGLAS

ADAM - Accident Diagnostics, Analysis and Management

ASTEC - Accident Source Term Evaluation Code

ASTRID - Assessment of Source Term for emergency response based on Installation Data

BWR - Boiling Water Reactor

GRS - Gesellshaft für Anlagen und ReaktorSicherheit

GUI - Graphic User Interface

GWD/MTU - GigaWatts-day/MegaTon of Uranium

IRSN - Institut de Radioprotection et de Surête Nucleaire

LWR - Light Water Reactor

MARS - MAAP Accident Response System

MCNP - Monte Carlo N Particle

MELCOR - Methods for Estimation of Leakages and Consequences of Releases

ORNL - Oak Ridge National Laboratory

PWR - Pressure Water Reactor

TFORI - Termo Fonte pelo ORIGEN

1 INTRODUÇÃO

Um dos requisitos de segurança para a operação de um reator nuclear é o acompanhamento e a estocagem do material radioativo produzido durante o funcionamento da usina. A possível liberação desse material deve ser sempre considerada em uma situação de emergência. Portanto, conhecido o histórico de funcionamento da usina durante o ciclo em avaliação, a quantidade de radionuclídeos passível de ser liberada, para a contenção ou para o ambiente, é sempre o foco do projeto, operação e regulamentação das usinas nucleares. A quantidade de material radioativo que pode ser liberada para o meio ambiente, assim como sua composição isotópica, constitui um parâmetro definido como termo fonte. O conhecimento do termo fonte no momento de um acidente é um dos requisitos para o licenciamento de usinas termonucleares e dado essencial para o serviço de emergência. Esses dados são usados para a quantificação da dose a que alguma população estaria exposta em casos emergenciais [1].

O termo fonte é expresso em função do tempo e das taxas de liberação e remoção dos materiais radioativos dentro da contenção bem como dos tipos e quantidades das espécies liberadas. No caso de um PWR são considerados: gases nobres, halogênios, metais alcalinos, telúrio, bário e estrôncio, metais nobres, cério, lantanídeos e as formas químicas do iodo. No estudo de segurança de reatores a determinação da composição do combustível antes de algum acidente é fundamental para obter o termo fonte [2].

Para o cálculo do termo fonte de um reator em situação emergencial pode-se utilizar o código de evolução de combustível ORIGEN 2.1. ORIGEN é um sistema de código computacional para cálculo de queima, decaimento e processamento de materiais radioativos, desenvolvido pelo Oak Ridge National Laboratory (ORNL). Trata-se de um código computacional unidimensional e monoenergético, que utiliza as equações de Bateman para descrever a evolução isotópica do combustível quando submetido a um fluxo de nêutrons bem como todo o processo radioativo associado. As características do termo fonte que podem ser calculadas pelo ORIGEN2.1 são, por exemplo, massa, radioatividade, toxicidade química, taxa de fissão etc . [3; 4]

ORIGEN2 é uma versão revisada do ORIGEN e incorpora atualizações de modelos de reatores, seções de choques, rendimentos de produtos de fissão, dados de decaimento de isótopos e biblioteca de fótons, sendo que o próprio código fonte também foi revisado nesta versão. ORIGEN2.1 substituiu o ORIGEN2 e incluiu bibliotecas adicionais para cálculo de queima estendida dos modelos de reatores tipo PWR e BWR, conforme documentado em ORNL/TM-11018 [3]. Podendo ainda ser acoplado a outros programas tais como MCNP5[5], MCNPX [6], KENO [7].

1.1 Justificativa

As liberações de radionuclídeos para o meio ambiente podem significar efeitos potencialmente danosos para o homem e para a natureza. O parâmetro mais importante na análise de consequências dessas liberações é a magnitude da liberação radioativa, assim como o comportamento dessa liberação em função do tempo. Para a determinação do termo-fonte é essencial que seja conhecido o inventário dos radionuclídeos presentes no núcleo do reator, no instante correspondente ao inicio de um acidente. Existe uma variedade de códigos computacionais disponíveis para o cálculo da composição dos combustíveis dos reatores nucleares durante a queima. No entanto, estes mesmos códigos apesar de disponibilizarem informações detalhadas sobre um acidente, não estão hábeis a fazer essa análise em um curto intervalo de tempo, menos de trinta minutos, devido ao alto custo computacional envolvido nos cálculos.

Pretende-se com este trabalho possibilitar a um usuário o cálculo da evolução isotópica de qualquer célula em alguma coordenada 3D do núcleo do reator, a fim de se obter o termo fonte para cálculos de dose numa população, por exemplo. As diferenças de potência, bem como as diferentes composições iniciais de cada região promovem termos fontes diversos em áreas distintas deste reator.

Numa análise de emergência é necessário que as decisões que mitiguem os danos possíveis sejam tomadas tão cedo quanto possível. Sob esta perspectiva é necessária uma ferramenta que seja capaz de dar informações sobre o termo fonte e suas características de maneira rápida e organizada.

Códigos de evolução isotópica que têm baixo custo computacional são os códigos da família do ORIGEN. O ORIGEN é capaz de simular a evolução de um combustível, mas não realiza cálculos através da distribuição de potência, o que seria interessante já que poderíamos calcular apenas uma região afetada poupando mais recursos e consequentemente tempo computacional. A família do ORIGEN também não possui uma interface gráfica de usuário (GUI), que seria um agente facilitador na experiência homem-máquina, reduzindo o tempo necessário de programação e aumentando ainda a confiabilidade dos dados, uma vez, que reduz a possibilidade de propagação de erros.

Visto que o código ORIGEN tem afinidades com o objetivo do projeto, embora careça de alguns recursos interessantes, um programa poderia ser criado acoplado ao ORIGEN para suprir suas carências e servir como ferramenta para a análise do termo fonte e suas características.

1.2 Objetivos gerais e específicos

Este trabalho se propõe à criação de um programa, denominado TFORI (Termo Fonte pelo ORIgen) que, para qualquer célula numa coordenada 3D de um núcleo de reator, levando em consideração a quantidade de nuclídeos, histórico e fatores de pico de potência radial e axial dessa região, seja capaz de simular a evolução do combustível, através do ORIGEN 2.1. Isso é feito para possibilitar uma ação mais efetiva em situações emergenciais, beneficiando o usuário com informações relevantes do termo fonte a fim de que através delas possa-se calcular as doses às quais uma população pudesse estar exposta.

Além de gerar entradas e processá-las através do ORIGEN automaticamente, o programa deverá permitir ao usuário visualizar graficamente as respostas, possibilitando a visão imediata do termo fonte e todos os parâmetros importantes para uma análise em caso de emergência, como radiotoxicidade, toxicidade química, concentração isotópica, atividade etc. Para isso foram definidos os seguintes objetivos específicos:

- Estudar o programa ORIGEN2.1 e suas especificidades de entrada bem como as características e padrões encontrados em sua saída;
- Criar uma interface gráfica de usuário (GUI) no MATLAB onde o usuário inclua dados relevantes para a análise como: a potência do reator, os fatores de pico axiais e radiais, a biblioteca desse reator, os identificadores das células e, se houver, ler também o histórico do fator de carga etc;
- Usar esses dados para gerar entradas adequadas do ORIGEN 2.1 para qualquer região determinada pelo usuário ou para todo o reator;
- Executar as entradas no ORIGEN2.1 automaticamente;
- Criar uma GUI no MATLAB capaz de interpretar as saídas do ORIGEN2.1 produzidas de forma objetiva e segura na forma de gráficos 2D;
- Validar o programa.

1.3 Apresentação da dissertação

No capítulo 2 é apresentado o código ORIGEN2.1 e os cálculos necessários para obter os dados sobre composição isotópica de um reator durante a operação. Em seguida, é descrita a evolução do ORIGEN desde sua primeira versão até as mais atuais. Ainda neste capítulo é feita uma explanação sobre os usos acadêmicos do MATLAB e suas versões.

No capítulo 3 é descrita a metodologia empregada para a escolha do ORIGEN2.1 como o código de evolução isotópica, a metodologia utilizada para preparar entradas que sejam capazes de fornecer o termo fonte e escolha da forma correta de apresentar o termo fonte e suas características, tais como atividade total e alfa, radiotoxicidade de inalação e ingestão, toxicidade etc.

No capítulo 4 é apresentado o programa TFORI explicitando as suas funcionalidades e ainda mostrando o tempo de processamento versus o número de casos rodados.

No capítulo 5 é feita a validação do programa TFORI através de comparações com arquivos padrões distribuídos pelo Oak Ridge Laboratory.

No capítulo 6 é apresentado um estudo de caso executado a partir da ferramenta desenvolvida.

Por fim, no capítulo 7, são apresentadas as conclusões gerais deste trabalho e as perspectivas futuras para a aplicação e aperfeiçoamento deste programa.

2 REVISÃO BIBLIOGRÁFICA

2.1 ORIGEN

2.1.1 Sobre o código e suas versões

A primeira versão do ORIGEN data do princípio da década de 70. Trata-se de um código computacional que foi amplamente usado para o cálculo de queima, decaimento e processamento de materiais radioativos. Durante o início da década de 80 um esforço foi feito pelo ORNL para atualizar o código ORIGEN original e suas bases associadas. O resultado deste esforço foram atualizações dos modelos dos reatores, sessões de choque, rendimento dos produtos de fissão, dados de decaimento, e do próprio código [3], Tais atualizações geraram a segunda versão do programa ORIGEN2 em 1983.

O objetivo primário do ORIGEN é calcular a evolução isotópica de algum material radioativo com especificações simples de entrada e informações, compiladas na forma de uma biblioteca dos dados de seção de choque. Algumas bibliotecas estão disponíveis no pacote fornecido pelo Oak Ridge Laboratory, mas também podem ser geradas pelo usuário.

O ORIGEN2 foi atualizado para o ORIGEN2.1 em agosto de 1991 e incluiu bibliotecas adicionais para os cálculos das queimas padrão e estendida dos reatores tipo PWR e BWR. Esta versão foi substituída quase dez anos depois, em junho de 2002, pelo ORIGEN2.2, estimulada pela descoberta de um usuário de uma discrepância nas massas dos produtos de fissão calculados usando o ORIGEN2.1. Nessa atualização houve modificações no código e uma limitação na etapa de irradiação para não mais que 100 dias/etapa o que reduziu a discrepância de aproximadamente 10% para 0,16%. O erro não afetava significativamente a massa dos produtos de fissão em cálculos típicos do ORIGEN2.1 envolvendo combustíveis de reatores, porque essencialmente todas as fissões vinham de actinídeos que têm bibliotecas de rendimento de produtos de fissão explícitas. Assim, a maioria dos cálculos feitos em versões anteriores não seria afetada. [8]

2.1.2 ORIGEN-S

Às mudanças nas seções de choque provocadas pelo histórico de operação bem como pelo projeto de um reator foram delegadas pouca importância baseado na ampla natureza dos estudos do ciclo de combustível durante o desenvolvimento do ORIGEN2. Consequentemente, a confiança no ORIGEN2 para resultados que eram altamente sensíveis às variações na seção de choque se tornavam um problema recorrente. ORIGEN-S foi desenvolvido com menos atenção à simplificação às entradas feitas pelo usuário, mas com foco em fornecer uma interface flexível com códigos de neutrônica que forneceriam seções de choque dependentes da queima baseado nas informações de projeto. [9]

2.1.3 ORIGEN-ARP

Para favorecer as necessidades da comunidade de usuários por velocidade computacional, facilidade de uso e precisão para uma ampla gama de condições de reatores e projeto foi desenvolvida a metodologia ORIGEN-ARP. ARP produz bibliotecas ORIGEN-S em qualquer enriquecimento, queima e densidade de moderadores para um projeto específico de arranjo de elementos combustível. Além disso, o sistema ORIGEN-ARP fornece um programa, ORIGNARP, para rápido ajuste e execução de uma entrada para um cenário específico de histórico e execução de uma simulação de um reator. [10].

2.2 Funcionalidade do ORIGEN

O ORIGEN usa um método de matriz exponencial para solucionar um grande sistema de equações diferenciais de primeira ordem, lineares, ordinárias, com coeficientes constantes [3] da seguinte forma:

$$\frac{dN_i}{dt} = P_i - L_i \tag{2.1}$$

onde

P_i é a quantidade do isótopo i produzido;

L_i é a quantidade de isótopo i transformado.

sendo

$$P_i = \sum_{x \to i} \Sigma_x^y \phi + \sum_{j \to i} \lambda_j N_j \phi \tag{2.2}$$

e

$$L_i = \Sigma_a^i \phi + \lambda_i N_i \tag{2.3}$$

onde

ϕ é o fluxo escalar a um grupo,

N_x é a quantidade do isótopo x,

$\sum_x^y$ é a seção de choque macroscópica do isótopo x para a reação y,

λ_i constante de decaimento do isótopo i,

λ_j constante de decaimento do isótopo j.

O código ORIGEN depende de uma biblioteca de seções de choque microscópica específica para efetuar os cálculos de composição isotópica aplicando esta biblioteca a um grupo e espectro de nêutrons.

2.3 Equações de Bateman

Harry Bateman derivou o sistema de equações diferenciais existentes na teoria das transformações radioativas [11]. Com a solução das equações de Bateman é possível calcular a quantidade de nuclídeos existentes em materiais expostos aos processos de irradiação e/ou em decaimento radioativo.

A equação de Bateman para a concentração N_i do isótopo i de um material é dada, por

$$\frac{dN_i}{dt} = -\sum_{j\neq i}\left[\lambda_{ji}^{d} + \int \phi(E,t)\sigma_{ji}^{tr}(E)dE\right]N_i + \sum_{j\neq i}\left[\lambda_{ij}^{d} + \int \phi(E,t)\sigma_{ij}^{tr}(E)dE\right]N_j, \quad (2.4)$$

onde, λ^d são as constantes de decaimento, $\phi(E,t)$é o fluxo de partículas com energia E no intervalo de tempo t e σ_{ji}^{tr} é a seção de choque microscópica média para uma dada reação e energia, neste caso para a transmutação do isótopo j no isótopo i.

O somatório é aplicado em todos os nuclídeos j do material. Uma vez que o fluxo de partículas depende da composição do material resolve-se a equação de Bateman para um fluxo constante em um reator. Este fluxo constante pode ser considerado assumindo condições de baixa fluência e pequenas seções de choque para um intervalo Δt. Sendo a seção de choque microscópica média para uma dada reação e espectro definida por:

$$\bar{\sigma}_{ij}^{tr} \equiv \frac{1}{\bar{\phi}}\int \phi(E)\sigma_{ij}^{tr}(E)dE \quad (2.5)$$

onde

$$\bar{\phi} = \int \phi(E)dE \quad (2.6)$$

reduzindo as equações de Bateman para

$$\frac{dN_i}{dt} = -\sum_{j\neq i}\left[\lambda_{ij}^{d} + \bar{\sigma}_{ji}^{tr}\bar{\phi}\right]N_i + \sum_{j\neq i}\left[\lambda_{ij}^{d} + \bar{\sigma}_{ij}^{tr}\bar{\phi}\right]N_j \quad (2.7)$$

que podem ser escritas no sistema vetorial como

$$\frac{d\vec{N}}{dt} = \hat{\Lambda}\vec{N} \quad (2.8)$$

onde

$\vec{N}$ é o vetor concentração de nuclídeos e

$\hat{\Lambda}$ é a matriz de transição em que cada termo dessa matriz é definido por:

$$\Lambda_{ij} \equiv \lambda_{ij}^{d} + \bar{\sigma}_{ij}^{tr}\bar{\varphi} \tag{2.9}$$

A solução formal dessa matriz é:

$$\vec{N}(t) = e^{\vec{\Lambda}t}\vec{N}(0) \tag{2.10}$$

onde a exponencial da matriz é expandida por série de Taylor válida para pequenos valores de t

$$e^{\Lambda t} \equiv 1 + \Lambda t + \frac{1}{2}\Lambda^2 t + \cdots \tag{2.11}$$

Esta solução é a base para a simulação de queima no ORIGEN 2.1 e é conhecida como Método da Matriz Exponencial.

2.4 MATLAB uso acadêmico e suas versões

MATLAB, Matrix Laboratory, é uma linguagem de computação de alto nível e um ambiente de computação numérica que permite manipulações de matriz, projeção de funções e dados além de desenvolvimento e implementação de algoritmos numéricos ou simbólicos para aplicação nas mais diversas áreas das ciências exatas. Através da escrita de funções próprias com extensão '.m' o MATLAB ainda permite a criação de ferramentas reutilizáveis.

Este sistema interativo tem como elemento básico uma matriz sem dimensionamento e permite a solução de problemas numéricos em uma fração do tempo de programação quando comparado a uma linguagem clássica.

O MATLAB fornece ferramentas para a construção de interfaces gráficas de usuário (GUI) que são de fácil acesso além de confiáveis. Um GUI é uma interface ilustrada para um programa. As GUI desenvolvidas no MATLAB podem tornar alguns programas mais fáceis de serem usados provendo a eles uma aparência consistente e com controles intuitivos como botões, caixas de listas, *sliders*, menus e assim por diante. A GUI deve se comportar de forma compreensível e previsível, a fim de que um usuário saiba o que esperar quando uma ação é realizada. Esta ferramenta será utilizada no desenvolvimento deste trabalho [12].

2.5 Termo Fonte e Códigos de Análise de Acidentes

Como dito anteriormente entende-se por termo fonte a composição isotópica de um material radioativo a partir de sua liberação no ambiente. Um dos objetivos desse dado é determinar ações mitigantes para os efeitos de uma eventual liberação. Sendo também um dos requisitos para o próprio licenciamento das usinas nucleares.

Durante o processo de queima do combustível em um reator nuclear ocorrem o decaimento e transmutação dos nuclídeos deste material e de seus filhos em novos nuclídeos com propriedades radioativas e físico-químicas diferentes da composição inicial. Mesmo após a queima deste combustível esse processo continua. Dentre os fatores que influenciam esse proceso podemos citar:

- geometria do reator;
- composição anterior ao evento mais recente;
- histórico de operação do reator;
- composição dos materiais estruturais do núcleo do reator;
- arranjo dos elementos combustíveis

Desta forma, em todos os reatores nucleares, a grande complexidade do comportamento da composição isotópica e termo fonte exige a monitoração, avaliação e previsão de acidentes para garantir a segurança da população. Além disso, quaisquer modificações realizadas na configuração do núcleo ou na rotina operacional do reator podem levar a mudanças no termo fonte.

Portanto, a utilização de códigos computacionais capazes de determinar a composição isotópica, o termo fonte e suas características radioativas e físico-químicas com dependência temporal são essenciais para a análise de segurança e ações mitigantes em caso de acidentes. A seguir são apresentadas o estado da arte de alguns códigos e programas que lidam com o termo fonte.

2.5.1 Ferramentas utilizadas em acidente para a avaliação do código fonte.

Esta seção fornece uma descrição das ferramentas computarizadas cujo objetivo principal seja a aplicação em situações de emergência para uma determinação rápida do termo fonte.

2.5.1.1 ASTEC

O código para cálculo de termo fonte ASTEC [13] (Accident Source Term Evaluation Code) foi desenvolvido em conjunto pelo IRSN (Institut de Radioprotection et de Surête Nucleaire, France) e GRS (Gesellshaft für Anlagen und ReaktorSicherheit, Germany). O objetivo deste código é computar a progressão de diversos acidentes em usinas nucleares desde o início de um evento até a liberação dos produtos de fissão no ambiente cobrindo todos os fenômenos importantes dentro e fora do vaso de pressão. O Código ASTEC é principalmente usado como uma ferramenta de suporte para perícias de segurança de usinas nucleares de 2ª e 3ª geração, como:

- cálculo de sequências de acidente;
- análise de procedimentos de gerência de acidentes;
- cálculo do termo fonte;
- análise de risco probabilístico (PSA nível 2).

ASTEC é composto por um conjunto de módulos que operam em simulações específicas e que podem ser usados tanto sozinhos quanto acoplados. Esta modularização do ASTEC é utilizada para otimizar os cálculos sendo que a simulação executada por cada módulo está descrita a seguir:

- CESAR: simula a termo-hidraúlica dos circuitos primários e secundários.
- ICARE: simula o decaimento do núcleo.
- ELSA: liberação dos produtos de fissão e dos materiais estruturais vindos do núcleo.
- SOPHAEROS: transporte dos produtos de fissão (aerosol e vapor) no sistema de refrigeração do reator.
- RUPUICUV: despressurização do vaso de pressão do reator e subsequente ejeção de metal fundido.
- CORIUM: comportamento das gotas de detritos aquecidos na contenção.
- CPA: termo-hidraúlica na contenção e comportamento de aerosois e outros produtos de fissão.
- COVI: combustão do H_2 e CO e o risco de explosão na contenção.
- DOSE: taxa de dose de inalação nos compartimentos da contenção.
- IODE: química do iodo e rutênio dentro da contenção.
- ISODOP: simula o destino e distribuição de todos os isótopos no reator de modo que forneça a outros módulos o calor de decaimento e as atividades devido às radiações alfa, beta e gama.
- MEDDICIS: simula interação do núcleo fundido e o concreto dentro da cavidade.
- SYSINT: caracterização de eventos que controlam o sistema de gerência de riscos.

2.5.1.2 ASTRID

O sistema ASTRID (Assessment of Source Term for emergency response based on Installation Data) foi desenvolvido através de um projeto da comunidade européia dividido em duas partes: uma em que a metodologia para análise de um acidente em LWR (Light Water Reactor) fosse analisada e outra onde uma ferramenta que suporte essa metodologia fosse desenvolvida.

O objetivo é realizar uma análise aprofundada da situação da usina. Sendo baseada no conceito de barreiras de segurança (por exemplo, combustível/revestimento, filtros/contenção). Funções de segurança crítica são definidas para cada barreira que assegura a integridade da mesma. O próximo passo é identificar quais sistemas de segurança são usados para estas funções. Durante um acidente as condições das barreiras são consideradas intactas, degradadas, perdidas ou desconhecidas. No princípio do trabalho é importante determinar à qual extensão as barreiras de segurança foram degradadas. Então, os prováveis termos fontes podem ser estimados através da predição de possíveis cenários.

Depois da conclusão do projeto algum esforço foi feito para desenvolver ainda mais o código, entretanto isto não ocorreu. O IRSN, um dos institutos envolvidos no desenvolvimento, chegou à conclusão que o ASTRID não estava maduro o suficiente para ser aplicado. [14]

2.5.1.3 ADAM

ADAM (Accident Diagnostics, Analysis and Management) é uma ferramenta para diagnóstico tanto de acidentes *online* quanto para acidentes *offline.* Foi desenvolvida pelo Energy Research, Inc (ERI) consistindo de quatro módulos:

- ADAM-PI (Pikett ingenieur) que apresenta parâmetros de processo importantes e informações relacionadas às condições do reator e contenção.
- ADAM-D (Diagnósticos *online*) que fornece métodos avançados de diagnóstico do cenário de acidente.
- ADAM-A (Análise e gerenciamento de acidentes *offline*) onde é feita uma prognose das sequências de evento e do termo fonte através da simulação de diferentes cenários para diferentes condições de contorno.
- ADAM-STEP (Predição de termo fonte) onde é feito o cálculo rápido do termo fonte baseado nos parâmetros das usinas e entradas fornecidas pelo usuário.

Atualmente é usado pelo ENSI (Swiss Federal *Nuclear* Safety Inspectorate) e está sendo implementado também, pelas autoridades húngaras e eslovacas. [15]

2.5.1.4 MARS

MARS (MAAP Accident Response System) é desenvolvido pela companhia norte-americana FAI que também desenvolveu o código de acidentes variados integrado MAAP. MARS é usado para fiscalização contínua e *online* das condições da usina através da interpretação dos valores dos parâmetros para detectar desvios dos modos de operação desejados. No caso de um desvio, MARS é supostamente capaz de diagnosticar a resposta da usina para este evento e seguir as condições do evento de maneira a determinar a evolução e possível piora dos eventos. Também é capaz de dinamicamente iniciar o MAAP que simula as condições futuras dessa usina.

O sistema MARS é usado somente no Consejo de Seguridad Nuclear (CSN) na Espanha, embora haja planos da implantação dele pelas autoridades suecas. [16]

2.5.1.5 MELCOR

MELCOR é um código computacional completamente integrado que simula a progressão de vários acidentes em usinas nucleares com reatores tipo LWRs. MELCOR foi desenvolvido pelo Sandia National Laboratories pela U.S. Nuclear Regulatory Commission como uma ferramenta de análise de risco para usinas de segunda geração. Um número maior de fenômenos de acidentes é contemplado para reatores PWRs e BWRs pelo MELCOR. Isto inclui resposta termo-hidraúlica no sistema de refrigeração do reator, cavidade do reator, ataque núcleo-concreto, produção de hidrogênio, transporte e combustão, liberação de produtos de fissão e comportamento de transporte. Usos atuais do MELCOR incluem a estimativa de vários termos fontes de acidente e suas sensibilidades e incertezas numa variedade de aplicações. [17]

O MELCOR é empregado em mais de 50 entidades pelo mundo, incluindo a US NRC. MELCOR-H2 é uma versão extendida do MELCOR projetada para simular dinamicamente dinamicamente modelos detalhados de reatores nucleares que estão

completamente acoplados com os componentes do sistema secundário modular e ciclos termoquímicos para a produção de hidrogênio e eletricidade. [18]

2.6 Aproximação para os fatores axiais de potência

Caso o usuário não possua os dados dos fatores de potência axial eles serão calculados segundo se segue. A potência linear pode ser expressa como o produto dos fatores axial e radial de potência de uma vareta, um elemento combustível ou área na seção radial pela densidade média de potência no núcleo, ou seja,

$$q'(r,z)_{área} = f_{radial} f_{axial} \overline{q'}_{núcleo} \tag{2.12}$$

onde,

$q'_{área}$ é a densidade linear local de potência;

f_{radial} é o fator de distribuição radial;

f_{axial} é o fator de distribuição axial;

$\overline{q'}_{núcleo}$ é a potência linear média do núcleo.

sendo,

$$f_{radial} = \frac{\overline{q'}_{área}}{\overline{q'}_{nucleo}} \tag{2.13}$$

onde,

$\overline{q'}_{área}$ é a potência linear média de uma área.

sendo,

$$f_{axial} = \frac{\overline{q'}_{vareta}(z)}{\overline{q'}_{vareta}} \tag{2.14}$$

onde,

$\overline{q'}_{vareta}(z)$ é a potência linear média da vareta na coordenada axial Z.

Supondo-se que a distribuição axial do fluxo de nêutrons da vareta combustível possa ser descrita por uma função senoidal e considerando-se que a densidade de potência seja proporcional ao fluxo de nêutrons, então

$$q'(z') = q'_{max}\, sen\left(\frac{\pi z'}{L}\right) \tag{2.15}$$

em que q'_{max} é a densidade linear máxima em $z' = L'/2$, e L' é comprimento do semi-ciclo do seno. A função $q'(z')$ encontra-se representada na Figura 2.1. Os símbolos que não foram definidos acima têm os significados seguintes: L_0, comprimento ativo da vareta; ℓ, distância de extrapolação do fluxo de nêutrons; L, comprimento total da vareta; z, coordenada axial. [19]

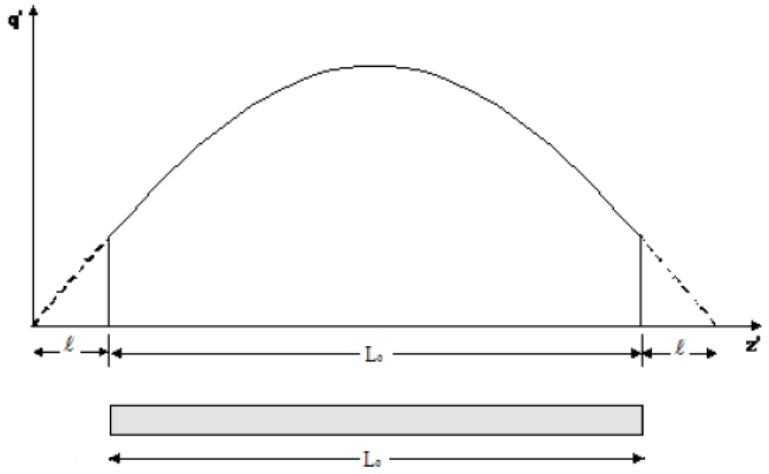

Figura 2.1: Distribuição axial de potência

3 METODOLOGIA

Entende-se por termo fonte a composição de um material passível de ser liberado durante um acidente. Quando há um acidente ou incidente nuclear é essencial conhecermos o termo fonte o quanto antes. No caso de um reator nuclear é possível calcular a composição através de códigos de evolução isotópicas, como GB, MCB, TRITON etc. Entretanto, alguns destes códigos exigem um alto recurso computacional. E portanto não são adequados para fazer uma análise de emergência na qual se tem um curto período para serem tomadas decisões. Uma método para obter o termo fonte em um curto intervalo de tempo é além de utilizar um código capaz de calcular a evolução isotópica num breve intervalo é restringir a porção calculada deste núcleo em células. Desta maneira poderemos calcular apenas uma porção atingida no menor intervalo de tempo possível.

O ORIGEN é um código de evolução isotópica que possui um baixo gasto computacional, mas não há a possibilidade de calcular células ou regiões específicas do reator, o que reduziria ainda mais o tempo de processamento ao focar numa área, além de proporcionar mais detalhes.

Um programa que se acople ao ORIGEN preenchendo essas lacunas é o objetivo deste trabalho a fim de que no futuro ele possa ser utilizado como uma ferramenta para análise emergencial de termos fontes.

Nesta seção é discutida a metodologia desenvolvida para abordar este problema, mantendo as características básicas do ORIGEN, ou seja, sem perda de recursos ou qualidade de informação para qualquer elemento combustível, área, região, em um número variável de células ou em uma coordenada tridimensional arbitrária do reator PWR.

3.1 Requisitos para uma solução

Para solucionar o problema apresentado acima propõe-se neste trabalho a criação de um programa chamado TFORI (Termo Fonte - ORIgen), que com dados dos

fatores radiais e axiais de pico de potência será capaz de simular a evolução isotópica de qualquer porção do núcleo de um reator tipo PWR, ou dele como um todo, durante um período a ser determinado pelo usuário. Esta evolução deverá ser feita através de um código de evolução isotópica capaz de:

- Calcular o termo fonte;
- Ter baixo gasto computacional;
- Fornecer informações sobre um grande número de nuclídeos.

Caso não haja dados sobre a composição isotópica imediatamente antes do acidente o usuário poderá inserir dados prévios de qualquer instante para que o código de evolução isotópica simule a queima deste combustível até a data desejada. Este histórico conterá dados do fator de carga, ou seja, da razão entre a potência real e a potência nominal, nos dias em que estes são alterados até a data mais recente.

Seguindo o desligamento do reator o código ORIGEN2 simulará o decaimento do combustível e o acompanhamento do termo fonte durante um período determinado pelo usuário em intervalos também controlados por ele.

Além disto, o programa deverá construir gráficos temporais baseado nas escolhas do usuário para qualquer nuclídeo, elemento, ou soma de grupos. Estes gráficos deverão ser confiáveis e precisos.

3.1.1 Pré-requisitos funcionais para o programa TFORI

O programa deve ser automatizado o suficiente para que o usuário insira apenas informações sobre as características do reator, potência térmica nominal, composição isotópica em um dado instante. Caso não haja dados sobre uma composição recente também é necessário informações sobre um histórico de potência deste reator, para o cálculo da queima até o evento de emergência. Para o cálculo de porções específicas da composição do núcleo do reator também são necessários os fatores de distribuição radial e axial de potência. Caso não existam dados sobre os fatores axiais o

programa deverá fazer uma aproximação senoidal da distribuição de potência no sentido axial da vareta combustível. Em suma deverão ser fornecidos:

- Os ‘labels’ ou rótulos do combustível que está sendo simulado, ou seja, a região onde se encontra o combustível e um identificador genérico;
- Biblioteca de seção de choque do PWR a ser simulado e o conjunto de actinídeos com produto de fissão direto, podendo ser usadas tanto as bibliotecas padrões distribuídas junto ao ORIGEN 2.1 quanto quaisquer outra biblioteca;
- Número de regiões com a mesma massa de metal pesado e composições isotópicas mais recentes
- Composição isotópica de cada uma dessas regiões, bem como uma nomenclatura para as mesmas;
- Número de células no núcleo do reator, nomenclatura de cada célula;
- Fator de pico de potência radial e axial de cada célula;
- Histórico de operação até a data de parada informando o fator de carga e o período de operação. Caso não haja dados recentes, os dados faltantes serão calculados pelo programa;

O programa deve ser capaz de calcular a potência local de cada célula, caso os fatores de distribuição axial não sejam fornecidos pelo usuário, e com esses dados construir entradas para serem executadas no código de evolução isotópica ORIGEN 2.1.

3.1.2 Pré-requisitos não funcionais

Conforme informado o TFORI é um programa escrito em MATLAB, um programa distribuído pela MATHWORKS. Não sendo um software livre é necessário uma licença inclusive para produções acadêmicas, que visa incluir interfaces para cálculo de distribuição de potência através do ORIGEN 2.1. Portanto, é necessária a instalação do ORIGEN 2.1 e do MATLAB versão r2010b ou superior. Os pré-requisitos computacionais são ditados pelo MATLAB. O tempo computacional gasto é definido por ambos programas.

O executável do ORIGEN utilizado neste trabalho foi gerado através da compilação do ORIGEN 2.1 utilizando o compilador de Fortran 77 MinGW. [20].

3.2 Definindo o código de evolução isotópica

Como dito anteriormente, o código de evolução isotópica precisa atender a três requisitos: (a) Calcular o termo fonte e suas características como por exemplo massa, radiotoxicidade, radioatividade, toxicidade química, etc. (b) Apresentar uma ampla gama de nuclídeos e elementos, pois ao lidarmos com uma situação de emergência o conhecimento de todos os nuclídeos é determinante para uma ação mitigante (c) usar poucos recursos computacionais, pois ao se tratar de uma situação de emergência as informações têm de ser dadas em tempo hábil.

Para satisfazer os critérios citados acima, o Departamento de Engenharia Nuclear da UFMG (DEN - UFMG) disponibilizou uma cópia do código ORIGEN 2.1, que conforme demonstrado na sessão 2.1 atende perfeitamente a estes três critérios e, ainda que existam versões mais modernas, estas não apresentam diferenças que interfiram significativamente nos resultados apresentados.

3.3 Definição do tempo de decaimento

Para definir o tempo de decaimento do combustível após uma parada forçada do reator, toma-se como referência o relatório NUREG-1465, que descreve os termos fontes de acidentes em usinas nucleares operadas à água leve.

Este documento apresenta uma descrição das fases de um acidente, bem como a duração da liberação de radionuclídeos nessas fases e a porcentagem de alguns grupos de radionuclídeos liberada em cada uma delas. As fases de um acidente são discutidas brevemente na Tabela 3.1, a duração e a porcentagem dessas liberações é apresentada na Tabela 3.2.

Fase de liberação	Região de liberação de Radionúclideos
Para o gap	Região entre a pastilha e revestimento da vareta combustível.
Precoce in-vessel	Primeira liberação no interior do vaso de pressão.
Ex-vessel	Região externa ao vaso de pressão.
Tardia in-vessel	Segunda liberação no interior do vaso de pressão.

Tabela 3.1: Fases de liberação de radionuclídeos no evento de um acidente [3]

Observando a Tabela 3.2 podemos perceber que grande parte da liberação acontece nas duas primeiras horas (93% aproximadamente). Se estendermos esse tempo a seis horas temos 98% das liberações realizadas. Portanto, seis horas de decaimento com intervalos de trinta minutos é uma boa estimativa para uma análise emergencial.

	Liberação no gap	Precoce in-vessel	Ex-vessel	Tardia in-vessel
Duração (horas)	0,5	1,3	2,0	10,0
Gases Nobres	5%	95%	0	0
Halogênios	5%	35%	25%	10%
Metais Alcalinos	5%	25%	35%	10%
Grupo do telúrio	0	5%	25%	0,5%
Bário e estrôncio	0	2%	10%	0
Metais Nobres	0	0,25%	0,2%	0
Grupo do cério	0	0,05%	0,5%	0
Lantanídeos	0	0,02%	0,5%	0

Tabela 3.2: Tempo de liberação por grupo de nuclídeo e fase de liberação [3]

4 DESENVOLVIMENTO DO PROJETO

4.1 TFORI

TFORI é um programa escrito na linguagem MATLAB. Através de uma interface simples utiliza o código ORIGEN2.1 para calcular a evolução isotópica do combustível em todas as coordenadas 3D de um reator PWR. Para isso os fatores radiais e axiais de distribuição de potência devem ser conhecidos, bem como a composição isotópica mais recente do núcleo do reator.

Uma vez calculado o termo fonte e suas características importantes para o cálculo (por exemplo, radioatividade, radiotoxicidade etc.) o programa iniciará sua segunda fase que é a análise dos dados gerados. Nesta fase, o programa rastreia os arquivos gerados e os disponibiliza em uma listagem simples para que o usuário escolha uma opção. Cada arquivo gerado pelo TFORI contém os dados da entrada de uma célula individual que representa uma porção do núcleo do reator. Os dados dessa célula ficarão disponibilizados para projeção em gráficos bidimensionais.

O programa TFORI pode ser subdividido em duas partes: a primeira seria responsável pela geração de sucessivas entradas válidas para compor todas as células desejadas por um usuário, executar estas entradas no código ORIGEN 2.1 e armazenar os resultados apropriadamente.

A segunda parte é responsável por interpretar os dados de saída, armazenar estes dados como variáveis associadas a vetores numéricos e disponibilizá-los para a projeção gráfica a partir da escolha do usuário. Um esquema do funcionamento do TFORI é fornecido na Figura 4.1.

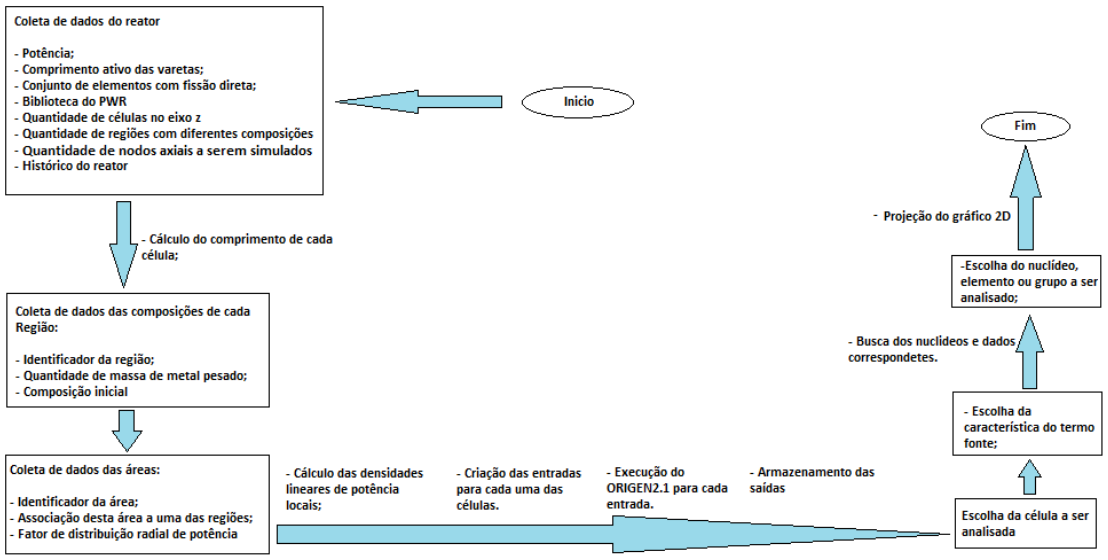

Figura 4.1: Breve esquema de uma operação típica do TFORI.

4.2 Entradas padrão

O TFORI irá construir entradas padrão para o ORIGEN 2.1. Esta entrada padrão deverá conter dados para queima do combustível, caso não haja informações recentes de sua composição isotópica. Após a irradiação esse combustível deverá decair por um tempo e intervalo a serem determinado pelo usuário. Em nosso trabalho estes dados foram fixados em 6 horas de decaimento em intervalos de trinta minutos. O escopo desta entrada padrão está descrito na Figura 4.2. E as razões para esse padrão são dicutidas aqui.

Os dados dos comandos podem ser vistos individualmente no manual de usuário do ORIGEN. Entretanto, um breve resumo sobre os comandos usados nesta entrada são apresentados na Tabela 4.1.

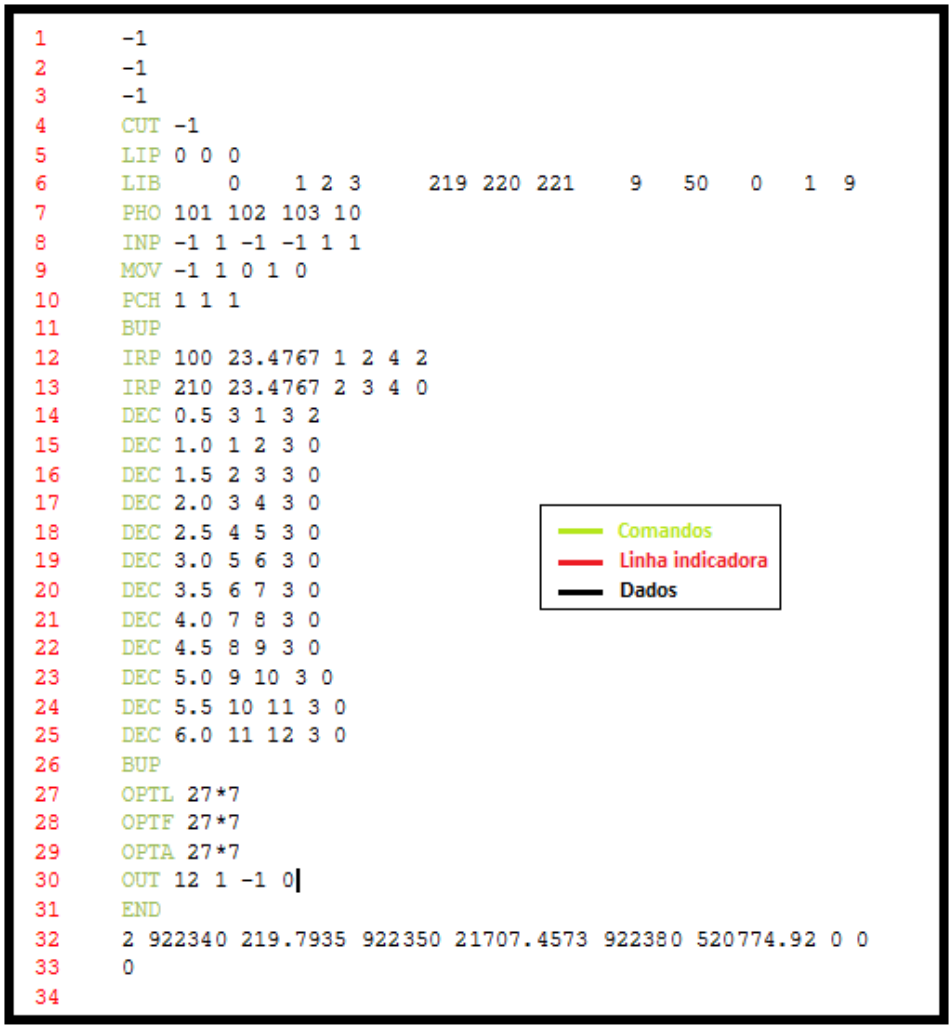

Figura 4.2: Uma entrada padrão do ORIGEN 2.1 produzida pelo TFORI.

O arquivo com as bibliotecas padrões distribuídas no ORIGEN2.1, armazenadas tradicionalmente no TAPE9.INP é composto por dados dos produtos de ativação, de actinídeos e filhos, e dados dos produtos de fissão. Outras bibliotecas podem ser geradas para este programa. Os produtos de ativação consistem de aproximadamente todos os nuclídeos naturais, seus produtos de absorção de nêutrons e os filhos do decaimento destes produtos sendo usado principalmente para lidar com material estrutural e impureza dos combustíveis. Os actinídeos compreendem os isótopos do tório ao einstênio (número atômico entre 90 e 99) que aparecem em quantidades significativas em combustíveis de descarga de reatores nucleares. Por fim, os produtos de fissão consistem dos nuclídeos produzidos pela fissão dos actinídeos incluindo seus produtos de captura e decaimento.

Comando	Descrição
CUT	Frações de corte para as tabelas de resumo
LIP	Controle de impressão da biblioteca
RDA	Comentários sobre o caso da entrada
LIB	Controle da biblioteca de reatores
PHO	Controle da biblioteca de fótons
TIT	Título do caso
INP	Leitura da composição de entrada
MOV	Mover composição nuclídica de um vetor para outro
BUP	Cálculo de queima (BURNUP)
IRP	Especificar potência e período de operação
IRF	Especificar fluxo e período de operação
DEC	Especificar tempo de decaimento
PCH	Imprime um vetor de saída
OPTL	Controle das tabelas de saída do grupo dos produtos de ativação
OPTA	Controle das tabelas de saída do grupo dos actinídeos
OPTF	Controle das tabelas de saída do grupo dos produtos de fissão
OUT	Impressão dos resultados calculados
END	Fim da execução do programa

Tabela 4.1: Lista dos comandos do ORIGEN2.1 usados pelo TFORI

Para a especificação da composição do material inicial o ORIGEN2.1 controla os parâmetros da entrada através do comando INP, que é mostrado por um exemplo na Tabela 4.2, e armazena os nuclídeos e suas quantidades ao fim da entrada em uma sequência numérica que ditará a qual segmento os dados de composição correspondem, o identificador dos nuclídeos e a sua representação mássica. O comando INP chama pela composição isotópica e a armazena em um vetor.

Entrada	Descrição
INP	Palavra-chave do comando
-1	A composição isotópica inicial será armazenada no vetor -1
1	A composição isotópica será lida em g/unidade base no arquivo TAPE5.INP
-1	Não será lida a taxa de alimentação isotópica contínua
-1	Não será lida a taxa de remoção isotópica contínua
1	Grandeza temporal para alimentação isotópica contínua são 'segundos'
1	Grandeza temporal para remoção isotópica contínua são 'segundos'

Tabela 4.2: Configuração do comando INP conforme inserido pelo programa TFORI.

A especificação da composição inicial do material é escrita para este caso no arquivo TAPE5.INP do ORIGEN2.1 nas linhas posteriores ao comando END. O identificador de seis dígitos é um inteiro que define unicamente um nuclídeo. Este identificador é definido como se segue:

$$NUCLIDEO = 10{,}000Z + 10A + IS \tag{4.1}$$

onde,

$NUCLIDEO$ é o identificador de seis dígitos;

Z é o número atômico do nuclídeo;

A é o número de massa do nuclídeo;

IS é o indicador do estado isomérico;

0 = estado fundamental

1 = estado excitado

O identificador para um elemento é dado analogamente por:

$$ELEMENTO = 10{,}000Z \tag{4.2}$$

Os dados são inseridos de forma a se ajustar ao padrão reconhecido pelo ORIGEN2.1. Um exemplo típico dos valores da composição inicial a serem inseridos pelo código ORIGEN em um vetor é apresentado na Figura 4.3.

```
2 922340  290.0  922350 32000. 922380 967710.     0   0.0 FUEL 3.2%
4 030000  1.0    050000 1.0  060000   89.4   070000  25.0 FUEL IMPU
4 080000 134454. 090000 10.7 110000   15.0   120000   2.0 FUEL IMPU
4 130000  16.7   140000 12.1 150000   35.0   170000   5.3 FUEL IMPU
4 200000  2.0    220000 1.0  230000   3.0    240000   4.0 FUEL IMPU
4 250000  1.7    260000 18.0 270000   1.0    280000  24.0 FUEL IMPU
4 290000  1.0    300000 40.3 420000   10.0   470000  0.1  FUEL IMPU
4 480000  25.0   490000 2.0  500000   4.0    640000  2.5  FUEL IMPU
4 740000  2.0    820000 1.0  830000   0.4    0       0.0  FUEL IMPU
 0
4 400000  979.11 500000 16.0    260000   2.25   240000  1.25 ZIRC-4
4 280000  0.02 130000 0.024  050000  0.00033 480000  0.00025 ZIRC-4
4 060000  0.120  270000 0.010  290000   0.020  720000  0.078 ZIRC-4
4 010000  0.013  250000 0.020  070000   0.080  080000  0.950 ZIRC-4
4 160000  0.035  220000 0.020  740000   0.020  230000  0.020 ZIRC-4
5 920000  0.0002      0 0.0                                  ZIRC-4
 0
4 260000  688.44 240000 190.0  280000   89.2   250000  20.0  SS-304
4 060000  0.8    150000 0.45   160000   0.3    140000  10.0  SS-304
4 070000  1.3    270000 0.8         0   0.0                  SS-304
 0
```

Figura 4.3: Entrada típica da composição inicial no ORIGEN 2.1.

4.3 Aquisição de dados pelo TFORI

Para executar o TFORI é necessário a inserção de dados a respeito do núcleo do reator a saber:

- Potência térmica nominal do reator;
- Comprimento das varetas;
- Número de células na região axial;
- Número de células na região radial. Esta célula pode ser, por exemplo, todo reator, uma região, um elemento combustível ou uma vareta sendo esta descrição feita pelo próprio usuário;
- Número de regiões com a mesma composição isotópica;
- Biblioteca a ser utilizada;
- Conjunto de actinídeos a serem utilizados;
- Histórico do fator de carga;
- Composição para cada célula (r,z);
- Fator de pico radial e axial para cada célula.

A Figura 4.4 mostra a primeira tela do TFORI onde esses dados são inseridos. Nesta sessão do programa os dados inseridos pelo usuário serão armazenados para identificar a quantidade de células que terá a composição isotópica calculada e o número de composições de combustível diferentes que existe no núcleo do reator. A posição de cada célula é conhecida através da nomenclatura dada pelo próprio usuário.

Figura 4.4: Primeira tela do TFORI.

Na Tabela 4.3 são apresentados os actinídeos que podem ter produtos de fissão diretos e como ele são apresentados nas opções do programa TFORI. Uma vez inseridos os dados, o programa os armazena em variáveis e chama a segunda tela do programa que está disposta na Figura 4.5.

Dado no ORIGEN	Descrição	Opção no TFORI
1	$^{235,\ 238}U,\quad ^{239,\ 241}Pu$	U, Pu
2	$^{232}Th,\quad ^{233,\ 235}U,\quad ^{239}Pu$	Th, U, Pu
3	$^{232}Th,\quad ^{233,\ 235,\ 238}U,\quad ^{239,\ 241}Pu$	$Th, U, Pu +$
4	$^{232}Th,\quad ^{233,\ 235,\ 238}U,\quad ^{239,\ 241}Pu,\ ^{245}Cm,\quad ^{252}Cf$	$Th, U, Pu, C \ldots$

Tabela 4.3: Opções do ORIGEN 2.1 para actinídeos com produtos de fissão diretos.

Figura 4.5: Segunda tela do TFORI

Como pode ser visto pela Figura 4.5 essa sessão do programa é responsável basicamente pela captura dos dados relacionados à composição isotópica. O que foi chamado nessa tela de região se refere às regiões no núcleo do reator que tenham a mesma massa e composição isotópica no momento em que o combustível é inserido. Lembrando que cada célula terá sua própria nomenclatura na saída. Esta definição é bem maleável podendo caracterizar desde elementos combustíveis, como foi o caso deste trabalho, ou ignorar esses limites e definir como uma região toda a composição do núcleo do reator em um único dado, ou ainda, criar sessões dependentes apenas do interesse de simulação do usuário.

Os dados coletados nessa sessão são:

- O identificador da região: será usado para nomear os arquivos na saída;

- A massa de metal pesado inicial, ou seja, a massa do combustível contida na região;
- O grupo ao qual pertencem aqueles dados, conforme ilustrados na Figura 4.5;
- A composição do combustível através da inserção da quantidade de massa, do número atômico e do número de massa do nuclídeos.

Esses dados são escritos em variáveis e serão usados para calcular e construir a composição isotópica de cada região. Cada composição será armazenada para ser usada na última sessão desta primeira parte do programa que está representada pela Figura 4.6

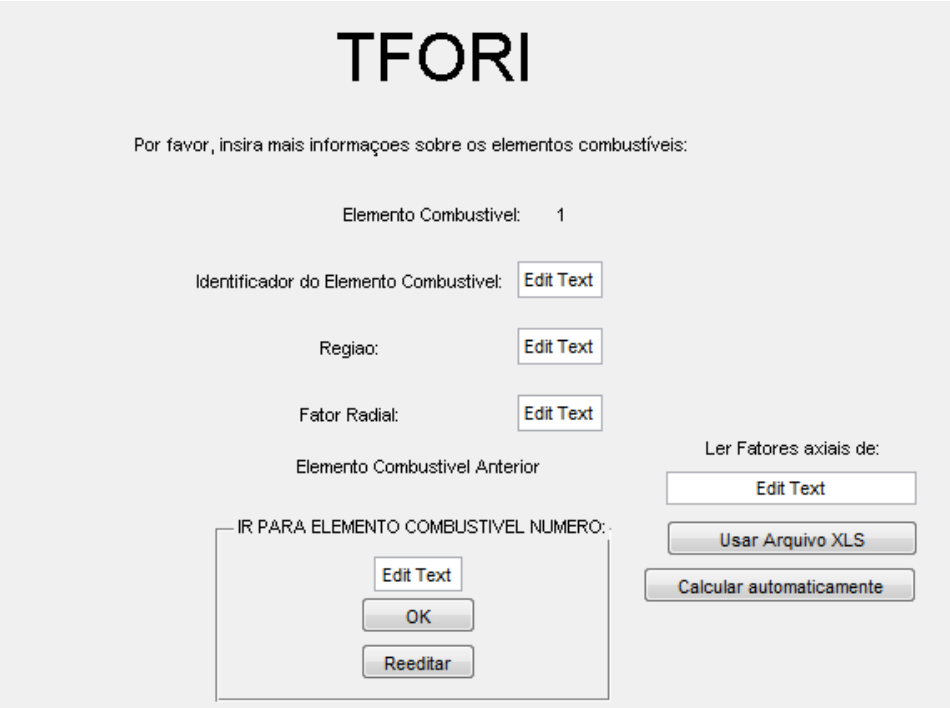

Figura 4.6: Terceira entrada do programa TFORI

4.4 Terceira sessão do programa TFORI

Nesta última sessão do programa são inseridos os dados radiais da célula, por exemplo, estas células podem ser consideradas como sendo um elemento combustível cada. A inserção do fator de distribuição radial de potência é feita nesta

seção do programa. No caso apresentado neste trabalho não havia informações dos fatores de distribuição de pico de potência axiais, por isso considerou-se que esta distribuição fosse descrita por uma função senoidal. As potências axiais podem ser inseridas manualmente por um usuário.

Uma vez capturados todos os dados, o programa comporá um arquivo de entrada para o ORIGEN 2.1 e usará os identificadores fornecidos pelo usuário para nomear esse arquivo. As posições das células podem ser reconhecidas através desses identificadores. Será gerado um arquivo de entrada para o ORIGEN2 para cada célula do núcleo do reator. Esses arquivos ficarão salvos e disponíveis para uso na pasta "INP" criada na instalação do programa.

Terminada essa sessão, terá início então a próxima etapa do programa, que é a execução do ORIGEN 2.1 para cada uma das células. Essa execução é feita automaticamente. Os resultados gerados pelo ORIGEN 2.1 são salvos em arquivos com o mesmo nome, porém, com uma extensão diferente. Esse arquivo é salvo numa pasta denominada "OUT" criada na instalação do programa. Os arquivos salvos ficam disponíveis para qualquer alteração que o usuário queira fazer diretamente. Como os arquivos são gerados pelo ORIGEN 2.1 o seu padrão é mantido podendo ser utilizado em programas que já trabalhem com este padrão.

4.5 Análise das saídas

O termo fonte é a quantidade de material radioativo disponível para o ambiente em um acidente. Conhecer o termo fonte e suas características é vital em qualquer análise para gerar ações mitigadoras que contornem, eliminem ou atenuem os danos prováveis. Essas características podem ser encontradas nas tabelas disponíveis após uma simulação no ORIGEN 2.1 para cada nuclídeo, elemento e soma de grupos. Uma saída típica do ORIGEN 2.1 é apresentada na Figura 4.7. Nesta figura, o marco zero é o evento de parada de emergência de um reator.

As características oferecidas pelo ORIGEN após uma simulação estão apresentadas na Tabela 4.4.

```
+                                                                                              FISSION PRODUCTS
    POWER= 1.00000E+00 MW,  BURNUP= 1.00000E+00 MWD,  FLUX= 1.00E+00 N/CM**2-SEC
0                     3 SUMMARY TABLE:  CONCENTRATIONS, GRAM-ATOMS

                0.5HR     1.0HR     1.5HR     2.0HR     2.5HR     3.0HR     3.5HR     4.0HR     4.5HR     5.0HR     5.5HR     6.0HR
 CE142      2.681E-01 2.681E-01 2.681E-01 2.681E-01 2.681E-01 2.681E-01 2.681E-01 2.681E-01 2.682E-01 2.682E-01 2.682E-01 2.682E-01
 PR143      2.434E-02 2.434E-02 2.434E-02 2.434E-02 2.434E-02 2.433E-02 2.433E-02 2.433E-02 2.433E-02 2.433E-02 2.433E-02 2.432E-02
 ND143      2.373E-01 2.373E-01 2.374E-01 2.374E-01 2.374E-01 2.374E-01 2.375E-01 2.375E-01 2.375E-01 2.375E-01 2.376E-01 2.376E-01
 CE144      1.920E-01 1.920E-01 1.920E-01 1.919E-01 1.919E-01 1.919E-01 1.919E-01 1.919E-01 1.919E-01 1.919E-01 1.919E-01 1.919E-01
 ND144      5.739E-02 5.740E-02 5.741E-02 5.742E-02 5.743E-02 5.744E-02 5.745E-02 5.746E-02 5.747E-02 5.748E-02 5.749E-02 5.750E-02
 ND145      1.779E-01 1.779E-01 1.780E-01 1.780E-01 1.780E-01 1.780E-01 1.780E-01 1.780E-01 1.780E-01 1.780E-01 1.780E-01 1.781E-01
 ND146      1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01 1.390E-01
 PM147      8.626E-02 8.627E-02 8.627E-02 8.628E-02 8.629E-02 8.630E-02 8.631E-02 8.632E-02 8.633E-02 8.634E-02 8.635E-02 8.636E-02
 ND148      7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02 7.960E-02
 ND150      3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02 3.227E-02
 SM150      4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02 4.507E-02
 SM151      1.479E-02 1.479E-02 1.480E-02 1.480E-02 1.480E-02 1.480E-02 1.480E-02 1.481E-02 1.481E-02 1.481E-02 1.481E-02 1.481E-02
 SM152      1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02 1.979E-02
 SUMTOT     9.125E+00 9.126E+00 9.126E+00 9.127E+00 9.127E+00 9.127E+00 9.128E+00 9.128E+00 9.128E+00 9.128E+00 9.129E+00 9.129E+00
0TOTAL      9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00 9.291E+00
1                                                                           OUTPUT UNIT =  6                    PAGE    2
```

Figura 4.7: Concentração de nuclídeos típica do ORIGEN2.1.

Tabela	Descrição	Grandeza
1	Composição isotópica de cada elemento	Fração atômica
2	Composição isotópica de cada elemento	Fração mássica
3	Composição	Átomos. grama
4	Composição	Fração atômica
5	Composição	Gramas
6	Composição	Fração mássica
7	Radioatividade (total)	Ci
8	Radioatividade (total)	Fracional
9	Potência térmica	Watts
10	Potência térmica	Fracional
13	Risco de inalação radioativa	m³ de ar
14	Risco de inalação radioativa	Fracional
15	Risco de ingestão radioativa	m³ de água
16	Risco de ingestão radioativa	Fracional
17	Risco de ingestão química	m³ de água
18	Risco de ingestão química	Fracional
19	Risco de absorção de nêutrons	Neutrons
20	Risco de absorção de nêutrons	Fracional
21	Taxa de fissão induzida por nêutrons	Fissões
22	Taxa de fissão induzida por nêutrons	Fracional
23	Radioatividade (alfa)	Ci
24	Radioatividade (alfa)	Fracional

Tabela 4.4: Tabelas de saída padrão do ORIGEN 2.1

Estas tabelas contêm dados fundamentais para a análise de termo fonte em caso de emergência, entretanto cabem algumas atualizações, como por exemplo, a radioatividade deveria ser apresentada em Becquerel e não em Curie. Esta atualização é feita pelo programa TFORI. Também análises visuais são preferidas a dados dispostos em tabelas, portanto uma apresentação gráfica dos dados é mais adequada.

A segunda parte do programa é responsável por fazer o tratamento gráfico das saídas calculadas pelo ORIGEN 2.1. As saídas do ORIGEN 2.1 são compostas por uma grande série de tabelas e outras informações convenientemente agrupadas de acordo com certo padrão. O programa TFORI é capaz de rastrear estes padrões oferecendo ao usuário a possibilidade de escolher uma tabela dentre todas as outras e, por procedimentos internos acumular o histórico de cada nuclídeo e elemento em variáveis individuais nomeadas de acordo com a nomenclatura original destes nuclídeos e elementos. Por exemplo: H, H3, AG, AG106, AG108, AG108M, são nomes de variavéis geradas pelo TFORI.

Estas variáveis estão dispostas em ordem alfabética para facilitar seu uso e outras variáveis, como a soma dos produtos de ativação, actinídeos e filhos e produtos de fissão, também estão disponíveis para serem projetadas em gráficos. Os gráficos gerados têm todos os recursos básicos necessários para fazer um tratamento de imagem além de poder ser salvo em vários formatos.

Para fazer o que foi descrito acima e para melhorar a resposta do programa além da intuitividade do uso, a interface foi dividida em três sessões. (a) seleção da característica e grupo de nuclídeo, (b) projetor de varíaveis com seleção de variáveis e (c) resposta gráfica para o pedido requerido. A Figura 4.8 mostra um esquema do funcionamento desta parte do programa.

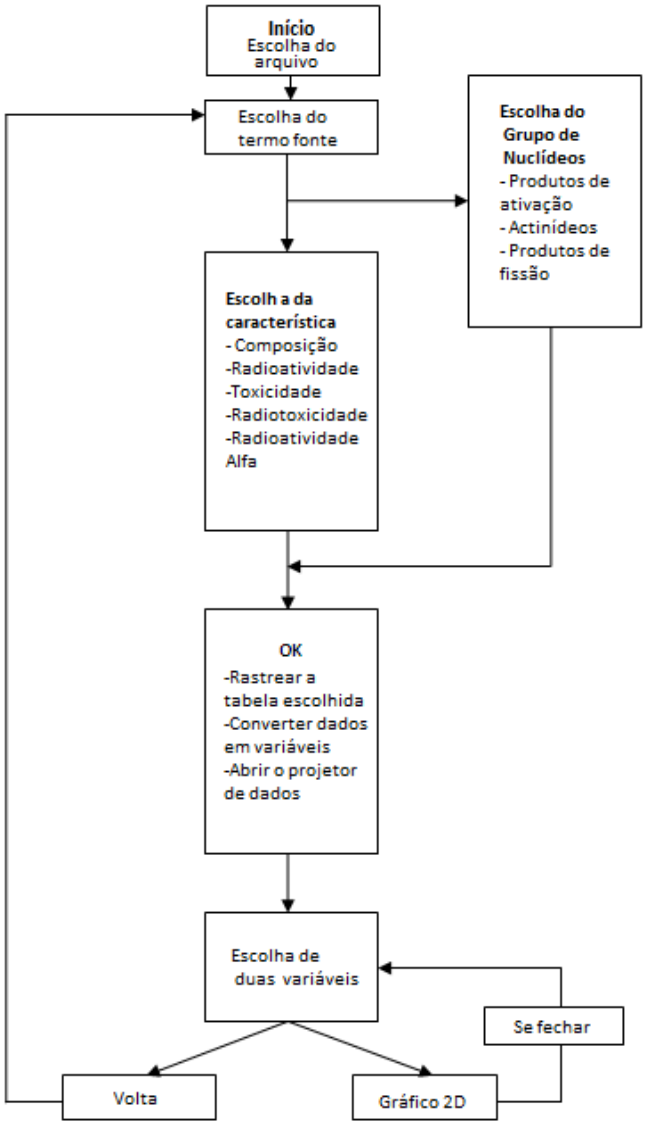

Figura 4.8: Esquema de funcionamento da segunda parte do TFORI.

Um exemplo da resposta final do programa é apresentado na Figura 4.9 onde é projetada a radioatividade total da Prata -109 metaestável versus radioatividade total da Prata-108 metaestável. Esta figura tem como objetivo mostrar que além de projetar os dados de um nuclídeo versus uma linha temporal o programa pode projetar quaisquer outras duas variáveis dadas.

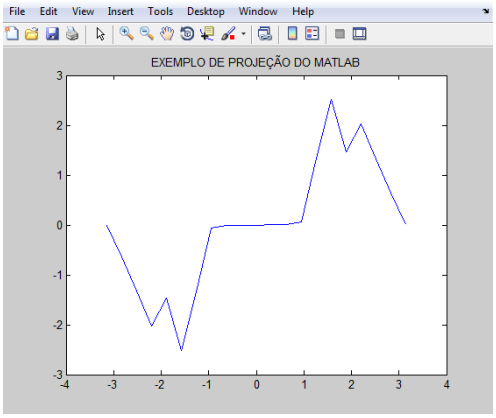

Figura 4.9: Exemplo de projeção 2D do programa MATLAB.

4.6 Tempo de processamento

Para verificar o tempo de processamento em função do número de células, foram executados várias células onde foi alterada apenas esta quantidade mantendo inalteradas a composição e a potência. Tratam-se de células do núcleo de um reator tipo PWR com 3765MW de potência. Os dados dessas células são fornecidos na Tabela 4.5 e todos foram executados no mesmo computador que possui processador Intel Core i5-2410M e 4 Gb de memória RAM. A Figura 4.10 e a Tabela 4.5 apresentam este resultado.

Dados das células			
Potência térmica		9,1414 MW	
Tempo de decaimento após emergência		6 horas	
Intervalo de tempo		0,5 horas	
Composição			
Nuclídeo	Quantidade (gramas)	Nuclídeo	Quantidade (gramas)
U-238	519470,184	U-235	20429,816

Tabela 4.5: Dados das células executadas no TFORI

Figura 4.10: Número de células pelo tempo de execução em segundos

Número de célula	Tempo (s)
10	8,8
50	49,3
100	93,4
200	249,3
560	567,7
800	1032,3

Tabela 4.6: Número de células e tempo de execução em segundos.

Conforme esperado o tempo de processamento cresce linearmente com a quantidade de casos. É importante notar que para análises mais rápidas é necessário restringir o número de células e selecionar alguma região do reator.

5 VALIDAÇÃO

5.1 Validação do programa TFORI

Neste capítulo serão apresentados testes que validam o programa TFORI. Essa validação será feita através da comparação entre as entradas exemplos para reatores tipo PWR distribuídas no pacote oficial do ORIGEN2.1 e entradas geradas pelo próprio TFORI. Também será verificado se o tratamento dado às saídas do ORIGEN2 pelo TFORI as modifica.

Serão executados dois testes e todos para reatores do tipo PWR.

- Reator tipo PWR operando com Urânio enriquecido com U^{235} e queima de 50 GWD/MTU. (PWRUE.INP)
- Reator tipo PWR operando com Urânio enriquecido com U^{235} e queima de 33 GWD/MTU. (PWRUS.INP)

5.2 Arquivo PWRUE.INP

A entrada padrão fornecida pelo pacote de distribuição original do ORIGEN2.1 está representado na Figura 5.1. Nesta entrada um reator tipo PWR queimará o dióxido de urânio enriquecido a 4,2% durante três ciclos num total de 1545 dias de operação com uma queima de 50 GWD/MTU. Os dados dessa entrada foram inseridos no TFORI a fim de produzir uma entrada funcionalmente idêntica a do exemplo PWRUE.INP. O resultado desse processo está apresentado na Figura 5.2.

```
-1
-1
-1
RDA  *  BURNUP OF PWR 4.2% UO2 FUEL & ASSY HDWARE, 50,000 MWD/MT
RDA     **  CROSS SECTION LIBRARY = PWRUE, 3 CYCLE
RDA     ***  SCOTT B. LUDWIG, OAK RIDGE NATIONAL LABORATORY
RDA     *****  (615) 574-7916, FTS 624-7916
RDA     -1 = FRESH PWR FUEL WITH IMPURITIES (1 MT = 1000 KG)
RDA     WARNING:  VECTORS ARE CHANGED WITH RESPECT TO CONTENT.
RDA               THESE CHANGES WILL BE NOTED ON RDA CARDS.
CUT     5 1.0E-10 7 1.0E-10 9 1.0E-10 -1
LIP     0 0 0
RDA        DECAY LIB    XSECT LIB                        VAR. XSECT
LIB     0    1 2 3     604 605 606    9   50   0   1        39
RDA      PHOTON LIB
PHO     101 102 103   10
TIT    INITIAL COMP. OF UNIT AMOUNTS OF FUEL AND STRUCTURAL MAT'LS
RDA     READ FUEL COMPOSITION INCLUDING IMPURITIES (1000 KG)
INP     -1   1  -1  -1  1  1
TIT     IRRADIATION OF ONE METRIC TON OF PWRU FUEL
MOV     -1   1   0   1.0
PCH      1   1   1
BUP
IRP       66.7     37.5   1   2   4  2   BURNUP= 2,500 MWD/MTIHM
IRP      133.3     37.5   2   3   4  0   BURNUP= 5,000 MWD/MTIHM
IRP      266.7     37.5   3   4   4  0   BURNUP=10,000 MWD/MTIHM
IRP      360.0     37.5   4   5   4  0   BURNUP=13,500 MWD/MTIHM
IRP      444.4     37.5   5   6   4  0   BURNUP=16,665 MWD/MTIHM
DEC      550.4            6   7   4  0    DECAY FOR 106.0 DAYS
IRP      650.0     37.5   7   8   4  0   BURNUP=20,400 MWD/MTIHM
IRP      780.0     37.5   8   9   4  0   BURNUP=25,275 MWD/MTIHM
IRP      890.0     37.5   9  10   4  0   BURNUP=29,400 MWD/MTIHM
IRP      994.8     37.5  10  11   4  0   BURNUP=33,330 MWD/MTIHM
DEC     1100.8           11  12   4  0    DECAY FOR 106.0 DAYS
IRP     1220.0     37.5  12   1   4  0   BURNUP=37,830 MWD/MTIHM
IRP     1320.0     37.5   1   2   4  0   BURNUP=41,550 MWD/MTIHM
IRP     1430.0     37.5   2   3   4  0   BURNUP=45,675 MWD/MTIHM
IRP     1545.33    37.5   3   4   4  0   BURNUP=50,000 MWD/MTIHM
BUP
OPTL      6*8  7  8  7  8  14*8
OPTA      6*8  7  8  7  8  14*8
OPTF      6*8  7  8  7  8  14*8
OUT      4    1     -1    0
END
2 922340  379.0 922350 42000. 922380 957621.   0    0.0  FUEL 4.2%
4 030000  1.0    050000 1.0    060000   89.4 070000 25.0 FUEL IMPU
4 080000 134454. 090000 10.7   110000   15.0 120000  2.0 FUEL IMPU
4 130000  16.7   140000 12.1   150000   35.0 170000  5.3 FUEL IMPU
4 200000  2.0    220000 1.0    230000   3.0  240000  4.0 FUEL IMPU
4 250000  1.7    260000 18.0   270000   1.0  280000 24.0 FUEL IMPU
4 290000  1.0    300000 40.3   420000   10.0 470000 0.1  FUEL IMPU
4 480000  25.0   490000 2.0    500000   4.0  640000 2.5  FUEL IMPU
4 740000  2.0    820000 1.0    830000   0.4  0      0.0  FUEL IMPU
0
```

Figura 5.1: Entrada exemplo PWRE.INP

```
-1
-1
-1
CUT      -1
LIP     0 0 0
LIB     0 1 2 3 604 605 606 9 50 0 1 39
PHO     101 102 103   10
INP     -1   1  -1  -1  1  1
MOV     -1   1   0   1.0
PCH      1   1   1
BUP
IRP      66.7    37.5   1   2   4  2
IRP     133.3    37.5   2   3   4  0
IRP     266.7    37.5   3   4   4  0
IRP     360.0    37.5   4   5   4  0
IRP     444.4    37.5   5   6   4  0
DEC     550.4           6   7   4  0
IRP     650.0    37.5   7   8   4  0
IRP     780.0    37.5   8   9   4  0
IRP     890.0    37.5   9  10   4  0
IRP     994.8    37.5  10  11   4  0
DEC    1100.8          11  12   4  0
IRP    1220.0    37.5  12   1   4  0
IRP    1320.0    37.5   1   2   4  0
IRP    1430.0    37.5   2   3   4  0
IRP    1545.33   37.5   3   4   4  0
BUP
OPTL   28*8
OPTA   28*8
OPTF   28*8
OUT      4    1     -1    0
END
2 922340  379.0 922350 42000. 922380 957621.   0    0.0
4 030000  1.0    050000 1.0    060000   89.4 070000 25.0
4 080000 134454. 090000 10.7   110000   15.0 120000  2.0
4 130000  16.7   140000 12.1   150000   35.0 170000  5.3
4 200000  2.0    220000 1.0    230000   3.0  240000  4.0
4 250000  1.7    260000 18.0   270000   1.0  280000 24.0
4 290000  1.0    300000 40.3   420000   10.0 470000 0.1
4 480000  25.0   490000 2.0    500000   4.0  640000 2.5
4 740000  2.0    820000 1.0    830000   0.4  0      0.0
0
```

Figura 5.2: Entrada gerada no TFORI baseada nos dados de PWRUE.INP.

Embora esta entrada não seja simetricamente idêntica a do exemplo, as operações foram mantidas intactas. As únicas diferenças foram a exclusão de títulos, comentários, frações de corte usadafoi a padrão e as tabelas a serem apresentadas pela entrada gerada pelo TFORI são todas as existentes no código ORIGEN2.1 e não uma seleção de tabelas como feito no caso exemplo. As diferenças não impactam no resultado. Nas Figuras 5.3 e 5.4 foi escolhida uma tabela para comparação dos valores entre as saídas geradas pela entrada produzida pelo TFORI e as produzidas pelo caso exemplo PWRU.INP e exposição de que essas entradas são funcionalmente identicas.

```
ORIGEN2 V2.1 (8-1-91), Run on Àà[illegible] at Èà[illegible]*CB[illegible]

     *  IRRADIATION OF ONE METRIC TON OF PWRU FUEL
+
     POWER= 3.75000E+01 MW,  BURNUP= 4.99999E+04 MWD,  FLUX= 3.44E+14 N/CM**2-SEC
0                      7 SUMMARY TABLE:  RADIOACTIVITY,  CURIES
```

	1220.0D	1320.0D	1430.0D	1545.3D
CU 80	4.065E-01	4.136E-01	4.269E-01	4.316E-01
ZN 80	2.044E+02	2.012E+02	1.997E+02	1.948E+02
GA 80	4.354E+03	4.192E+03	4.038E+03	3.832E+03
GE 80	2.469E+04	2.417E+04	2.353E+04	2.279E+04
AS 80	3.175E+04	3.144E+04	3.094E+04	3.037E+04
BR 80	8.723E-01	9.317E-01	9.829E-01	1.045E+00
BR 80M	5.565E-01	5.938E-01	6.230E-01	6.598E-01
CU 81	2.980E-02	3.056E-02	3.183E-02	3.244E-02
ZN 81	4.639E+01	4.631E+01	4.678E+01	4.635E+01
GA 81	2.542E+03	2.462E+03	2.396E+03	2.293E+03
GE 81	2.961E+04	2.867E+04	2.770E+04	2.648E+04
AS 81	4.980E+04	4.911E+04	4.818E+04	4.709E+04
SE 81	5.347E+04	5.287E+04	5.200E+04	5.098E+04
SE 81M	1.463E+03	1.427E+03	1.383E+03	1.336E+03
KR 81M	3.416E-02	3.826E-02	4.278E-02	4.774E-02
ZN 82	5.996E+00	6.054E+00	6.191E+00	6.208E+00
GA 82	9.364E+02	9.155E+02	9.010E+02	8.714E+02
GE 82	2.571E+04	2.468E+04	2.368E+04	2.238E+04
AS 82	4.374E+04	4.232E+04	4.082E+04	3.897E+04
AS 82M	1.798E+04	1.759E+04	1.711E+04	1.656E+04
BR 82	6.228E+03	7.046E+03	8.084E+03	9.027E+03
BR 82M	2.403E+03	2.723E+03	3.133E+03	3.504E+03
ZN 83	5.574E-01	5.661E-01	5.832E-01	5.884E-01
GA 83	2.576E+02	2.535E+02	2.516E+02	2.452E+02
GE 83	1.849E+04	1.769E+04	1.695E+04	1.596E+04
AS 83	7.786E+04	7.459E+04	7.123E+04	6.715E+04
SE 83	4.718E+04	4.644E+04	4.544E+04	4.429E+04
SE 83M	7.073E+04	6.887E+04	6.673E+04	6.422E+04
BR 83	1.204E+05	1.180E+05	1.149E+05	1.113E+05
KR 83M	1.207E+05	1.183E+05	1.154E+05	1.119E+05

Figura 5.3: Radioatividade (Ci) de alguns produtos de fissão do exemplo PWRUE.INP.

```
 ORIGEN2 V2.1 (8-1-91), Run on Àà(▯▯▯▯▯▯▯ at Èà(▯*CB▯

     *  IRRADIATION OF ONE METRIC TON OF PWRU FUEL
+
     POWER= 3.75000E+01 MW,  BURNUP= 4.99999E+04 MWD,  FLUX= 3.44E+14 N/CM**2-SEC
0                        7 SUMMARY TABLE:  RADIOACTIVITY,  CURIES
                              ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
               1220.0D   1320.0D   1430.0D   1545.3D
 CU 80        4.065E-01 4.136E-01 4.269E-01 4.316E-01
 ZN 80        2.044E+02 2.012E+02 1.997E+02 1.948E+02
 GA 80        4.354E+03 4.192E+03 4.038E+03 3.832E+03
 GE 80        2.469E+04 2.417E+04 2.353E+04 2.279E+04
 AS 80        3.175E+04 3.144E+04 3.094E+04 3.037E+04
 BR 80        8.723E-01 9.317E-01 9.829E-01 1.045E+00
 BR 80M       5.565E-01 5.938E-01 6.230E-01 6.598E-01
 CU 81        2.980E-02 3.056E-02 3.183E-02 3.244E-02
 ZN 81        4.639E+01 4.631E+01 4.678E+01 4.635E+01
 GA 81        2.542E+03 2.462E+03 2.396E+03 2.293E+03
 GE 81        2.961E+04 2.867E+04 2.770E+04 2.648E+04
 AS 81        4.980E+04 4.911E+04 4.818E+04 4.709E+04
 SE 81        5.347E+04 5.287E+04 5.200E+04 5.098E+04
 SE 81M       1.463E+03 1.427E+03 1.383E+03 1.336E+03
 KR 81M       3.416E-02 3.826E-02 4.278E-02 4.774E-02
 ZN 82        5.996E+00 6.054E+00 6.191E+00 6.208E+00
 GA 82        9.364E+02 9.155E+02 9.010E+02 8.714E+02
 GE 82        2.571E+04 2.468E+04 2.368E+04 2.238E+04
 AS 82        4.374E+04 4.232E+04 4.082E+04 3.897E+04
 AS 82M       1.798E+04 1.759E+04 1.711E+04 1.656E+04
 BR 82        6.228E+03 7.046E+03 8.084E+03 9.027E+03
 BR 82M       2.403E+03 2.723E+03 3.133E+03 3.504E+03
 ZN 83        5.574E-01 5.661E-01 5.832E-01 5.884E-01
 GA 83        2.576E+02 2.535E+02 2.516E+02 2.452E+02
 GE 83        1.849E+04 1.769E+04 1.695E+04 1.596E+04
 AS 83        7.786E+04 7.459E+04 7.123E+04 6.715E+04
 SE 83        4.718E+04 4.644E+04 4.544E+04 4.429E+04
 SE 83M       7.073E+04 6.887E+04 6.673E+04 6.422E+04
 BR 83        1.204E+05 1.180E+05 1.149E+05 1.113E+05
 KR 83M       1.207E+05 1.183E+05 1.154E+05 1.119E+05
```

Figura 5.4: Radioatividade (Ci) de alguns produtos de fissão da entrada produzida pelo programa TFORI.

Não há diferenças entre as tabelas, os valores são idênticos e também a sua apresentação. Na saída produzida pelo TFORI todas as tabelas disponíveis pelo ORIGEN2 são apresentadas, porque esses dados serão tratados mais tarde pelo programa segundo as escolhas do usuário. Outras tabelas estão disponíveis para comparação nas Figuras 5.4 e 5.5.

```
     *  IRRADIATION OF ONE METRIC TON OF PWRU FUEL
+                                                                                                 FISSION PRODUCTS
     POWER= 3.75000E+01 MW,  BURNUP= 4.99999E+04 MWD,  FLUX= 3.44E+14 N/CM**2-SEC
0                    9 SUMMARY TABLE:  THERMAL POWER,  WATTS
            1220.0D   1320.0D   1430.0D   1545.3D
  H        2.021E-02 2.235E-02 2.470E-02 2.718E-02
 CO        1.123E-01 1.168E-01 1.233E-01 1.274E-01
 NI        2.116E+00 2.186E+00 2.286E+00 2.344E+00
 CU        1.508E+01 1.521E+01 1.548E+01 1.550E+01
 ZN        8.678E+01 8.561E+01 8.484E+01 8.294E+01
 GA        5.438E+02 5.303E+02 5.176E+02 4.991E+02
 GE        2.000E+03 1.940E+03 1.880E+03 1.800E+03
 AS        8.456E+03 8.154E+03 7.843E+03 7.456E+03
 SE        1.739E+04 1.671E+04 1.601E+04 1.515E+04
 BR        4.944E+04 4.710E+04 4.471E+04 4.186E+04
 KR        6.590E+04 6.315E+04 6.026E+04 5.684E+04
 RB        1.319E+05 1.273E+05 1.222E+05 1.162E+05
 SR        1.156E+05 1.138E+05 1.111E+05 1.078E+05
  Y        1.834E+05 1.823E+05 1.793E+05 1.760E+05
 ZR        9.130E+04 9.283E+04 9.279E+04 9.258E+04
 NB        1.929E+05 1.974E+05 1.988E+05 2.004E+05
 MO        8.217E+04 8.555E+04 8.761E+04 9.037E+04
 TC        1.157E+05 1.223E+05 1.270E+05 1.329E+05
 RU        2.551E+04 2.773E+04 2.909E+04 3.073E+04
 RH        3.106E+04 3.408E+04 3.687E+04 3.978E+04
 PD        3.117E+03 3.347E+03 3.561E+03 3.788E+03
 AG        4.529E+03 4.952E+03 5.436E+03 5.904E+03
 CD        1.929E+03 2.012E+03 2.101E+03 2.172E+03
 IN        7.947E+03 8.228E+03 8.498E+03 8.727E+03
 SN        1.807E+04 1.838E+04 1.854E+04 1.870E+04
 SB        7.139E+04 7.227E+04 7.246E+04 7.274E+04
 TE        7.582E+04 7.594E+04 7.533E+04 7.474E+04
  I        1.743E+05 1.754E+05 1.748E+05 1.745E+05
 XE        8.374E+04 8.359E+04 8.262E+04 8.172E+04
 CS        1.527E+05 1.531E+05 1.522E+05 1.514E+05
 BA        8.988E+04 8.973E+04 8.876E+04 8.782E+04
```

Figura 5.4: Potência térmica (Watts) fornecida por alguns produtos de fissão do exemplo PWRUE.INP.

```
     *  IRRADIATION OF ONE METRIC TON OF PWRU FUEL
+                                                                                                 FISSION PRODUCTS
     POWER= 3.75000E+01 MW,  BURNUP= 4.99999E+04 MWD,  FLUX= 3.44E+14 N/CM**2-SEC
0                    9 SUMMARY TABLE:  THERMAL POWER,  WATTS
            1220.0D   1320.0D   1430.0D   1545.3D
  H        2.021E-02 2.235E-02 2.470E-02 2.718E-02
 CO        1.123E-01 1.168E-01 1.233E-01 1.274E-01
 NI        2.116E+00 2.186E+00 2.286E+00 2.344E+00
 CU        1.508E+01 1.521E+01 1.548E+01 1.550E+01
 ZN        8.678E+01 8.561E+01 8.484E+01 8.294E+01
 GA        5.438E+02 5.303E+02 5.176E+02 4.991E+02
 GE        2.000E+03 1.940E+03 1.880E+03 1.800E+03
 AS        8.456E+03 8.154E+03 7.843E+03 7.456E+03
 SE        1.739E+04 1.671E+04 1.601E+04 1.515E+04
 BR        4.944E+04 4.710E+04 4.471E+04 4.186E+04
 KR        6.590E+04 6.315E+04 6.026E+04 5.684E+04
 RB        1.319E+05 1.273E+05 1.222E+05 1.162E+05
 SR        1.156E+05 1.138E+05 1.111E+05 1.078E+05
  Y        1.834E+05 1.823E+05 1.793E+05 1.760E+05
 ZR        9.130E+04 9.283E+04 9.279E+04 9.258E+04
 NB        1.929E+05 1.974E+05 1.988E+05 2.004E+05
 MO        8.217E+04 8.555E+04 8.761E+04 9.037E+04
 TC        1.157E+05 1.223E+05 1.270E+05 1.329E+05
 RU        2.551E+04 2.773E+04 2.909E+04 3.073E+04
 RH        3.106E+04 3.408E+04 3.687E+04 3.978E+04
 PD        3.117E+03 3.347E+03 3.561E+03 3.788E+03
 AG        4.529E+03 4.952E+03 5.436E+03 5.904E+03
 CD        1.929E+03 2.012E+03 2.101E+03 2.172E+03
 IN        7.947E+03 8.228E+03 8.498E+03 8.727E+03
 SN        1.807E+04 1.838E+04 1.854E+04 1.870E+04
 SB        7.139E+04 7.227E+04 7.246E+04 7.274E+04
 TE        7.582E+04 7.594E+04 7.533E+04 7.474E+04
  I        1.743E+05 1.754E+05 1.748E+05 1.745E+05
 XE        8.374E+04 8.359E+04 8.262E+04 8.172E+04
 CS        1.527E+05 1.531E+05 1.522E+05 1.514E+05
 BA        8.988E+04 8.973E+04 8.876E+04 8.782E+04
```

Figura 5.5: Potência térmica (Watts) fornecidos por alguns produtos de fissão resultantes da entrada gerada pelo programa TFORI.

5.3 Arquivo PWRUS.INP

A entrada padrão fornecida pelo pacote de distribuição original do ORIGEN2.1 PWRUS.INP está representada na Figura 5.6.

```
-1
-1
-1
RDA  *  BURNUP OF PWR 3.2% UO2 FUEL & ASSY HARDWARE, 33,000 MWD/MT
RDA  **  CROSS SECTION LIBRARY = PWRUS, 3 CYCLE
RDA  ***  SCOTT B. LUDWIG, OAK RIDGE NATIONAL LABORATORY
RDA  ****  (615) 574-7916, FTS 624-7916
RDA     -1 = FRESH PWR FUEL WITH IMPURITIES (1 MT = 1000 KG)
RDA     WARNING:  VECTORS ARE CHANGED WITH RESPECT TO CONTENT.
RDA               THESE CHANGES WILL BE NOTED ON RDA CARDS.
CUT     5 1.0E-10 7 1.0E-10 9 1.0E-10 -1
LIP     0 0 0
RDA        DECAY LIB    XSECT LIB                    VAR. XSECT
LIB     0    1 2 3     601 602 603    9   50   0   1       38
RDA      PHOTON LIB
PHO     101 102 103  10
TIT   INITIAL COMP. OF UNIT AMOUNTS OF FUEL AND STRUCTURAL MAT'LS
RDA     READ FUEL COMPOSITION INCLUDING IMPURITIES (1000 KG)
INP     -1  1  -1  -1  1  1
TIT     IRRADIATION OF ONE METRIC TON OF PWR FUEL
MOV     -1 1 0 1.0
PCH     1  1  1
BUP
IRP       66.7    37.5    1   2   4 2  BURNUP=  2,500 MWD/MTIHM
IRP      133.3    37.5    2   3   4 0  BURNUP=  5,000 MWD/MTIHM
IRP      293.3    37.5    3   4   4 0  BURNUP= 11,000 MWD/MTIHM
DEC      399.3            4   5   4 0   DECAY FOR 106.0 DAYS
IRP      506.0    37.5    5   6   4 0  BURNUP= 15,000 MWD/MTIHM
IRP      639.3    37.5    6   7   4 0  BURNUP= 20,000 MWD/MTIHM
IRP      692.7    37.5    7   8   4 0  BURNUP= 22,000 MWD/MTIHM
DEC      798.7            8   9   4 0   DECAY FOR 106.0 DAYS
IRP      878.7    37.5    9  10   4 0  BURNUP= 25,000 MWD/MTIHM
IRP     1012.0    37.5   10  11   4 0  BURNUP= 30,000 MWD/MTIHM
IRP     1092.0    37.5   11  12   4 0  BURNUP= 33,000 MWD/MTIHM
BUP
OPTL      4*8  7  8  7  8  7  8  8*8  5*7  8
OPTA      4*8  7  8  7  8  7  8  8*8  5*7  8
OPTF      4*8  7  8  7  8  7  8  8*8  5*7  8
OUT      12    1     -1    0

END
2 922340  290.0  922350  32000. 922380   967710.     0  0.0 FUEL 3.2%
4 030000  1.0    050000  1.0    060000   89.4   070000 25.0 FUEL IMPU
4 080000 134454. 090000  10.7   110000   15.0   120000  2.0 FUEL IMPU
4 130000  16.7   140000  12.1   150000   35.0   170000  5.3 FUEL IMPU
4 200000  2.0    220000  1.0    230000   3.0    240000  4.0 FUEL IMPU
4 250000  1.7    260000  18.0   270000   1.0    280000 24.0 FUEL IMPU
4 290000  1.0    300000  40.3   420000   10.0   470000 0.1  FUEL IMPU
4 480000  25.0   490000  2.0    500000   4.0    640000 2.5  FUEL IMPU
4 740000  2.0    820000  1.0    830000   0.4    0      0.0  FUEL IMPU
0
```

Figura 5.6: Entrada exemplo PWRS.INP

Nesta entrada, um reator tipo PWR queimará o dióxido de urânio enriquecido a 3,2% durante três ciclos num total de 1092 dias de operação com uma queima de 33 GWD/MTIHM. Os dados dessa entrada foram inseridos no TFORI a fim de produzir uma entrada funcionalmente idêntica a do exemplo PWRUE.INP. O resultado desse processo está apresentado na Figura 5.7. A metodologia empregada para essa validação é a mesma da seção 5.2.

```
-1
-1
-1
CUT     -1
LIP     0 0 0
LIB     0    1 2 3 601 602 603   9   50   0   1   38
PHO     101 102 103  10
INP     -1  1  -1  -1  1  1
MOV     -1 1 0 1.0
PCH     1  1  1
BUP
IRP       66.7    37.5    1   2   4 2
IRP      133.3    37.5    2   3   4 0
IRP      293.3    37.5    3   4   4 0
DEC      399.3            4   5   4 0
IRP      506.0    37.5    5   6   4 0
IRP      639.3    37.5    6   7   4 0
IRP      692.7    37.5    7   8   4 0
DEC      798.7            8   9   4 0
IRP      878.7    37.5    9  10   4 0
IRP     1012.0    37.5   10  11   4 0
IRP     1092.0    37.5   11  12   4 0
BUP
OPTL      28*7
OPTA      28*7
OPTF      28*7
OUT      12    1     -1    0
END
2 922340  290.0  922350  32000. 922380   967710.     0  0.0
4 030000  1.0    050000  1.0    060000   89.4   070000 25.0
4 080000 134454. 090000  10.7   110000   15.0   120000  2.0
4 130000  16.7   140000  12.1   150000   35.0   170000  5.3
4 200000  2.0    220000  1.0    230000   3.0    240000  4.0
4 250000  1.7    260000  18.0   270000   1.0    280000 24.0
4 290000  1.0    300000  40.3   420000   10.0   470000 0.1
4 480000  25.0   490000  2.0    500000   4.0    640000 2.5
4 740000  2.0    820000  1.0    830000   0.4    0      0.0
0
```

Figura 5.7: Entrada gerada no TFORI baseada nos dados de PWRUS.INP.

Como pode ser notado as duas entradas têm a mesma função embora os comentários e outras funções ornamentais tenham sido removidas. Nas Figuras 5.8 e 5.9 são apresentadas a concentração em gramas de alguns produtos de fissão para os dados gerados a partir do caso exemplo PWRUS.INP e a partir do programa TFORI respectivamente. Como pode ser observado as tabelas são idênticas.

```
 ORIGEN2 V2.1 (8-1-91), Run on Ää(▯▯▯▯▯▯▯ at Èä(▯ªCB▯

     *  IRRADIATION OF ONE METRIC TON OF PWR FUEL
+                                                                                                              FISSION PRODUCTS
     POWER= 3.75000E+01 MW,  BURNUP= 3.30000E+04 MWD,  FLUX= 3.26E+14 N/CM**2-SEC
0                     5 SUMMARY TABLE:  CONCENTRATIONS, GRAMS
                               ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
               0.0D       66.7D     133.3D     293.3D     399.3D     506.0D     639.3D     692.7D     798.7D     878.7D    1012.0D    1092.0D
 PR153      0.000E+00 1.426E-06 1.469E-06 1.566E-06 0.000E+00 1.652E-06 1.765E-06 1.844E-06 0.000E+00 1.895E-06 1.989E-06 2.085E-06
 ND153      0.000E+00 5.059E-05 5.380E-05 5.935E-05 0.000E+00 6.298E-05 6.736E-05 7.028E-05 0.000E+00 7.224E-05 7.555E-05 7.918E-05
 PM153      0.000E+00 2.701E-04 2.894E-04 3.221E-04 0.000E+00 3.426E-04 3.671E-04 3.831E-04 0.000E+00 3.940E-04 4.124E-04 4.324E-04
 SM153      0.000E+00 1.783E-01 2.426E-01 4.073E-01 1.621E-17 5.281E-01 6.874E-01 7.706E-01 3.065E-17 8.610E-01 1.041E+00 1.158E+00
 EU153      0.000E+00 3.308E+00 7.799E+00 2.320E+01 2.360E+01 3.668E+01 5.668E+01 6.548E+01 6.625E+01 7.925E+01 1.037E+02 1.187E+02
 GD153      0.000E+00 2.100E-06 1.929E-05 1.835E-04 1.355E-04 4.581E-04 1.013E-03 1.268E-03 9.357E-04 1.505E-03 2.595E-03 3.259E-03
 ND154      0.000E+00 1.277E-05 1.415E-05 1.667E-05 0.000E+00 1.822E-05 2.010E-05 2.118E-05 0.000E+00 2.211E-05 2.355E-05 2.496E-05
 PM154      0.000E+00 6.380E-05 7.173E-05 8.573E-05 0.000E+00 9.405E-05 1.039E-04 1.095E-04 0.000E+00 1.144E-04 1.219E-04 1.293E-04
 PM154M     0.000E+00 5.944E-06 7.182E-06 9.193E-06 0.000E+00 1.023E-05 1.140E-05 1.203E-05 0.000E+00 1.260E-05 1.346E-05 1.429E-05
 SM154      0.000E+00 1.626E+00 3.534E+00 8.990E+00 8.991E+00 1.318E+01 1.901E+01 2.153E+01 2.153E+01 2.545E+01 3.246E+01 3.695E+01
 EU154      0.000E+00 1.333E-01 5.660E-01 3.076E+00 3.005E+00 6.041E+00 1.152E+01 1.425E+01 1.392E+01 1.857E+01 2.773E+01 3.379E+01
 GD154      0.000E+00 6.466E-04 5.368E-03 6.220E-02 1.333E-01 2.315E-01 4.699E-01 6.105E-01 9.399E-01 1.193E+00 1.788E+00 2.259E+00
 ND155      0.000E+00 2.985E-06 3.339E-06 4.019E-06 0.000E+00 4.455E-06 4.988E-06 5.286E-06 0.000E+00 5.558E-06 5.979E-06 6.374E-06
 PM155      0.000E+00 7.748E-06 9.025E-06 1.129E-05 0.000E+00 1.261E-05 1.416E-05 1.500E-05 0.000E+00 1.578E-05 1.698E-05 1.810E-05
 SM155      0.000E+00 3.091E-04 3.658E-04 4.666E-04 0.000E+00 5.250E-04 5.951E-04 6.327E-04 0.000E+00 6.693E-04 7.278E-04 7.804E-04
 EU155      0.000E+00 6.433E-01 1.192E+00 2.458E+00 2.361E+00 3.440E+00 5.203E+00 6.075E+00 5.834E+00 7.407E+00 1.047E+01 1.257E+01
 GD155      0.000E+00 3.184E-03 6.834E-03 1.603E-02 1.137E-01 2.079E-02 3.247E-02 3.422E-02 2.757E-01 4.384E-02 6.108E-02 6.787E-02
 ND156      0.000E+00 1.928E-06 2.255E-06 2.918E-06 0.000E+00 3.345E-06 3.873E-06 4.149E-06 0.000E+00 4.430E-06 4.856E-06 5.232E-06
 PM156      0.000E+00 1.231E-06 1.521E-06 2.057E-06 0.000E+00 2.370E-06 2.744E-06 2.936E-06 0.000E+00 3.132E-06 3.423E-06 3.685E-06
 SM156      0.000E+00 4.117E-03 5.208E-03 7.154E-03 0.000E+00 8.245E-03 9.526E-03 1.018E-02 0.000E+00 1.084E-02 1.183E-02 1.272E-02
 EU156      0.000E+00 2.163E-01 3.653E-01 7.315E-01 5.850E-03 9.149E-01 1.353E+00 1.599E+00 1.274E-02 1.900E+00 2.798E+00 3.440E+00
 GD156      0.000E+00 3.756E-01 1.293E+00 5.522E+00 6.255E+00 9.648E+00 1.670E+01 2.035E+01 2.195E+01 2.715E+01 4.176E+01 5.320E+01
 PM157      0.000E+00 2.784E-06 3.499E-06 4.897E-06 0.000E+00 5.748E-06 6.782E-06 7.302E-06 0.000E+00 7.859E-06 8.671E-06 9.392E-06
 SM157      0.000E+00 3.438E-05 4.478E-05 6.397E-05 0.000E+00 7.498E-05 8.807E-05 9.458E-05 0.000E+00 1.015E-04 1.116E-04 1.207E-04
 EU157      0.000E+00 4.329E-03 5.784E-03 8.636E-03 0.000E+00 1.029E-02 1.272E-02 1.408E-02 0.000E+00 1.555E-02 1.895E-02 2.173E-02
 GD157      0.000E+00 1.897E-02 2.555E-02 3.844E-02 4.714E-02 4.490E-02 5.396E-02 5.786E-02 7.205E-02 6.427E-02 7.645E-02 8.647E-02
 SM158      0.000E+00 9.462E-05 1.254E-04 1.845E-04 0.000E+00 2.194E-04 2.613E-04 2.819E-04 0.000E+00 3.046E-04 3.369E-04 3.658E-04
 EU158      0.000E+00 1.056E-04 1.404E-04 2.068E-04 0.000E+00 2.457E-04 2.924E-04 3.154E-04 0.000E+00 3.407E-04 3.768E-04 4.091E-04
 GD158      0.000E+00 3.640E-01 9.253E-01 2.889E+00 2.889E+00 4.599E+00 7.298E+00 8.566E+00 8.567E+00 1.060E+01 1.478E+01 1.777E+01
 SM159      0.000E+00 2.380E-06 3.289E-06 5.082E-06 0.000E+00 6.158E-06 7.463E-06 8.096E-06 0.000E+00 8.813E-06 9.825E-06 1.072E-05
```

Figura 5.8: Concentração (gramas) fornecida por alguns produtos de fissão do exemplo PWRUS.INP.

```
 ORIGEN2 V2.1 (8-1-91), Run on Ää(▯▯▯▯▯▯▯ at Èä(▯ªCB▯

     *  IRRADIATION OF ONE METRIC TON OF PWR FUEL
+                                                                                                              FISSION PRODUCTS
     POWER= 3.75000E+01 MW,  BURNUP= 3.30000E+04 MWD,  FLUX= 3.26E+14 N/CM**2-SEC
0                     5 SUMMARY TABLE:  CONCENTRATIONS, GRAMS
                               ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
               0.0D       66.7D     133.3D     293.3D     399.3D     506.0D     639.3D     692.7D     798.7D     878.7D    1012.0D    1092.0D
 PR153      0.000E+00 1.426E-06 1.469E-06 1.566E-06 0.000E+00 1.652E-06 1.765E-06 1.844E-06 0.000E+00 1.895E-06 1.989E-06 2.085E-06
 ND153      0.000E+00 5.059E-05 5.380E-05 5.935E-05 0.000E+00 6.298E-05 6.736E-05 7.028E-05 0.000E+00 7.224E-05 7.555E-05 7.918E-05
 PM153      0.000E+00 2.701E-04 2.894E-04 3.221E-04 0.000E+00 3.426E-04 3.671E-04 3.831E-04 0.000E+00 3.940E-04 4.124E-04 4.324E-04
 SM153      0.000E+00 1.783E-01 2.426E-01 4.073E-01 1.621E-17 5.281E-01 6.874E-01 7.706E-01 3.065E-17 8.610E-01 1.041E+00 1.158E+00
 EU153      0.000E+00 3.308E+00 7.799E+00 2.320E+01 2.360E+01 3.668E+01 5.668E+01 6.548E+01 6.625E+01 7.925E+01 1.037E+02 1.187E+02
 GD153      0.000E+00 2.100E-06 1.929E-05 1.835E-04 1.355E-04 4.581E-04 1.013E-03 1.268E-03 9.357E-04 1.505E-03 2.595E-03 3.259E-03
 ND154      0.000E+00 1.277E-05 1.415E-05 1.667E-05 0.000E+00 1.822E-05 2.010E-05 2.118E-05 0.000E+00 2.211E-05 2.355E-05 2.496E-05
 PM154      0.000E+00 6.380E-05 7.173E-05 8.573E-05 0.000E+00 9.405E-05 1.039E-04 1.095E-04 0.000E+00 1.144E-04 1.219E-04 1.293E-04
 PM154M     0.000E+00 5.944E-06 7.182E-06 9.193E-06 0.000E+00 1.023E-05 1.140E-05 1.203E-05 0.000E+00 1.260E-05 1.346E-05 1.429E-05
 SM154      0.000E+00 1.626E+00 3.534E+00 8.990E+00 8.991E+00 1.318E+01 1.901E+01 2.153E+01 2.153E+01 2.545E+01 3.246E+01 3.695E+01
 EU154      0.000E+00 1.333E-01 5.660E-01 3.076E+00 3.005E+00 6.041E+00 1.152E+01 1.425E+01 1.392E+01 1.857E+01 2.773E+01 3.379E+01
 GD154      0.000E+00 6.466E-04 5.368E-03 6.220E-02 1.333E-01 2.315E-01 4.699E-01 6.105E-01 9.399E-01 1.193E+00 1.788E+00 2.259E+00
 ND155      0.000E+00 2.985E-06 3.339E-06 4.019E-06 0.000E+00 4.455E-06 4.988E-06 5.286E-06 0.000E+00 5.558E-06 5.979E-06 6.374E-06
 PM155      0.000E+00 7.748E-06 9.025E-06 1.129E-05 0.000E+00 1.261E-05 1.416E-05 1.500E-05 0.000E+00 1.578E-05 1.698E-05 1.810E-05
 SM155      0.000E+00 3.091E-04 3.658E-04 4.666E-04 0.000E+00 5.250E-04 5.951E-04 6.327E-04 0.000E+00 6.693E-04 7.278E-04 7.804E-04
 EU155      0.000E+00 6.433E-01 1.192E+00 2.458E+00 2.361E+00 3.440E+00 5.203E+00 6.075E+00 5.834E+00 7.407E+00 1.047E+01 1.257E+01
 GD155      0.000E+00 3.184E-03 6.834E-03 1.603E-02 1.137E-01 2.079E-02 3.247E-02 3.422E-02 2.757E-01 4.384E-02 6.108E-02 6.787E-02
 ND156      0.000E+00 1.928E-06 2.255E-06 2.918E-06 0.000E+00 3.345E-06 3.873E-06 4.149E-06 0.000E+00 4.430E-06 4.856E-06 5.232E-06
 PM156      0.000E+00 1.231E-06 1.521E-06 2.057E-06 0.000E+00 2.370E-06 2.744E-06 2.936E-06 0.000E+00 3.132E-06 3.423E-06 3.685E-06
 SM156      0.000E+00 4.117E-03 5.208E-03 7.154E-03 0.000E+00 8.245E-03 9.526E-03 1.018E-02 0.000E+00 1.084E-02 1.183E-02 1.272E-02
 EU156      0.000E+00 2.163E-01 3.653E-01 7.315E-01 5.850E-03 9.149E-01 1.353E+00 1.599E+00 1.274E-02 1.900E+00 2.798E+00 3.440E+00
 GD156      0.000E+00 3.756E-01 1.293E+00 5.522E+00 6.255E+00 9.648E+00 1.670E+01 2.035E+01 2.195E+01 2.715E+01 4.176E+01 5.320E+01
 PM157      0.000E+00 2.784E-06 3.499E-06 4.897E-06 0.000E+00 5.748E-06 6.782E-06 7.302E-06 0.000E+00 7.859E-06 8.671E-06 9.392E-06
 SM157      0.000E+00 3.438E-05 4.478E-05 6.397E-05 0.000E+00 7.498E-05 8.807E-05 9.458E-05 0.000E+00 1.015E-04 1.116E-04 1.207E-04
 EU157      0.000E+00 4.329E-03 5.784E-03 8.636E-03 0.000E+00 1.029E-02 1.272E-02 1.408E-02 0.000E+00 1.555E-02 1.895E-02 2.173E-02
 GD157      0.000E+00 1.897E-02 2.555E-02 3.844E-02 4.714E-02 4.490E-02 5.396E-02 5.786E-02 7.205E-02 6.427E-02 7.645E-02 8.647E-02
 SM158      0.000E+00 9.462E-05 1.254E-04 1.845E-04 0.000E+00 2.194E-04 2.613E-04 2.819E-04 0.000E+00 3.046E-04 3.369E-04 3.658E-04
 EU158      0.000E+00 1.056E-04 1.404E-04 2.068E-04 0.000E+00 2.457E-04 2.924E-04 3.154E-04 0.000E+00 3.407E-04 3.768E-04 4.091E-04
 GD158      0.000E+00 3.640E-01 9.253E-01 2.889E+00 2.889E+00 4.599E+00 7.298E+00 8.566E+00 8.567E+00 1.060E+01 1.478E+01 1.777E+01
 SM159      0.000E+00 2.380E-06 3.289E-06 5.082E-06 0.000E+00 6.158E-06 7.463E-06 8.096E-06 0.000E+00 8.813E-06 9.825E-06 1.072E-05
```

Figura 5.9: Concentração (gramas) fornecida por alguns produtos de fissão resultantes da entrada gerada pelo programa TFORI.

Pela comparação das duas figuras fica claro novamente que os dados são idênticos em seus valores. Isso comprova que as entradas produzidas pelo TFORI são idênticas funcionalmente àquelas produzidas pelo usuário. Cabe dizer também que as saídas não foram disponibilizadas em sua totalidade por comodidade visual uma vez que elas têm mais de 7000 linhas no caso das geradas pelos casos exemplos e mais de 14000 linhas no caso das geradas pelo programa TFORI. Essa discrepância de valores do número de linhas surge porque o programa TFORI pede todas as saídas, enquanto em entradas produzidas por usuários, esse volume de dados pode ser controlado.

5.4 Validação da projeção de dados feito pelo do programa TFORI

Nesta seção será simulado um teste usando um dos exemplos padrões disponíveis no pacote do ORIGEN2.1, para satisfazer as condições necessárias para gerar um arquivo TAPE6.OUT com o termo fonte referente ao combustível e suas impurezas. Os dados do combustível são apresentados na Tabela 5.1.

Nesta fase usamos um caso exemplo do ORIGEN 2.1 disponível no pacote distribuído pelo Oak Ridge Laboratory: um reator tipo PWR operando com combustível dióxido de urânio enriquecido com 4,15%; BURNUP de 50.000 MWD/MTU, 5 ciclos. Em nosso caso é considerado apenas o combustível e suas impurezas para um período residente no núcleo de 280 dias. A composição é mostrada na Tabela 5.1. Este curto período é utilizado para colocar em evidência o comportamento de três nuclídeos analisados depois do desligamento. Para validar os resultados parciais nós iremos comparar a saída do TFORI com a saída do ORIGEN 2.1 mostrando que ambas operam independentemente. A saída do TFORI é um gráfico 2D. Dados extraídos do programa estão apresentados na Tabela 5.2.

Os três nuclídeos analisados aqui são o I^{135}, Xe^{135} e Sm^{149}. O Xe^{135} é o produto de fissão mais significante devido à alta seção de choque de absorção de nêutrons térmicos e relativamente alto rendimento de fissão. O I^{135} é o pai direto do Xe^{135} e assim é o responsável pelo aumento da concentração deste último mesmo durante o desligamento do reator. O Sm^{149} é outro produto de fissão associado com envenenamento do combustível devido ao seu grande rendimento de fissão e seção de

choque de absorção. Devido as características apresentadas por estes nuclídeos os dados referentes a eles são bem conhecidos e de fácil acesso na literatura. [22, 23]

Nuclídeo/ elemento	Composição (gramas)	Nuclídeo/ Elemento	Composição (gramas)
U^{234}	376	U^{235}	41500
B	1,0	C	89,4
F	10,7	Na	15,0
Si	12,1	P	35,0
Ca	2,0	Ti	1,0
Mn	1,7	Fe	18,0
Zn	40,3	Mo	10,0
In	2,0	Sn	4,0
Pb	1,0	Bi	0,4
U^{238}	958124,0	Li	1,0
N	25,0	O	134454,0
Mg	2,0	Al	16,7
Cl	5,3	Cr	4,0
V	3,0	Ni	24,0
Co	27,0	Cu	1,0
Ag	1,0	Cd	25,0
Gd	2,5	W	2,0

Tabela 5.1. Composição do combustível simulado e suas impurezas

As Figuras 5.10 e 5.11, mostram a composição em gramas para o I^{135} e o Xe^{135}, respectivamente. O período de 48 horas foi escolhido para compatibilizar com sete meias vidas do I^{135} e facilitar a comparação entre esses nuclídeos. O comportamento esperado foi completamente satisfeito. Observa-se a quantidade

decrescente do I^{135} sendo responsável pelo pequeno aumento na concentração de Xe^{135} durante as primeiras dez horas estando a transmutação e as curvas de acordo com a literatura.

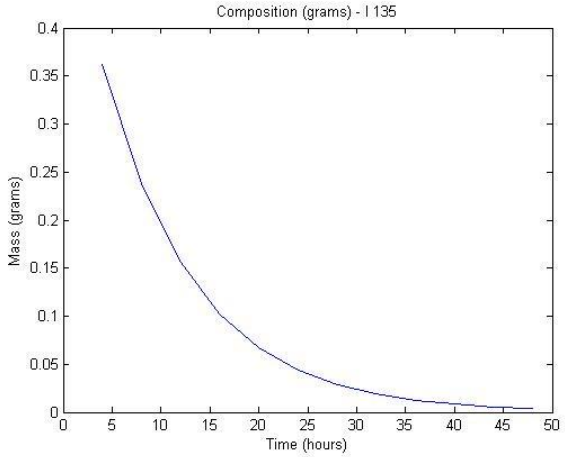

Figura 5.10: Composição do I^{135} após o desligamento do reator (gráfico gerado pelo TFORI).

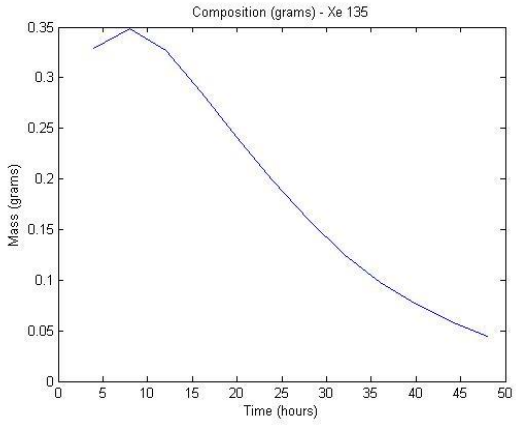

Figura 5.11: Composição do Xe^{135} após o desligamento do reator (gráfico gerado pelo TFORI).

A Figura 5.12 mostra a composição para o Sm^{149} durante um período de três horas quando a maior parte de seu valor de equilíbrio já foi atingida. Durante um desligamento do reator o decaimento dos produtos de fissão podem resultar em Nd^{149} que por sua vez decairá em Nd^{149} e finalmente em Sm^{149}. Esta corrente de decaimento é a responsável pelo comportamento observado, e, de novo concorda com a literatura.

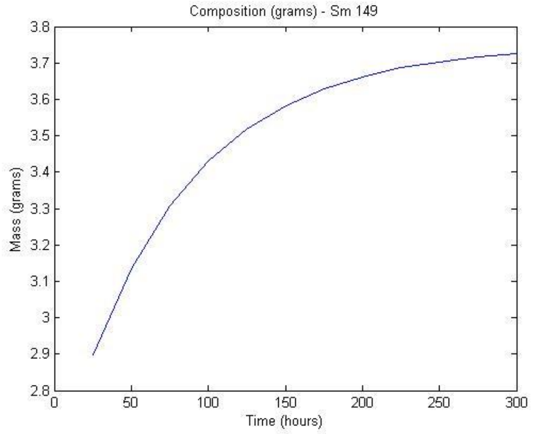

Figura 5.12: Composição do Sm^{149} durante o desligamento do reator.

Na Figura 5.13 é mostrado como a radioatividade de alguns nuclídeos do termo fonte seria apresentada em uma saída padrão do ORIGEN 2.1. Em destaque na figura os resultados para o I^{135} e o Xe^{135}. Para comparação mais imediata, os dados extraídos do programa TFORI são apresentados na Tabela 5.2. Na Figura 5.14 o resultado correspondente para o samário. Conforme pode ser percebido, uma saída em gráficos 2D é uma forma mais acessível para realizar analises rápidas e então mais apropriada a uma situação de emergência. Além disso, é mostrado que os dados são idênticos, ou seja, o TFORI não modifica os dados calculados pelo ORIGEN 2.1, apenas os lê.

* IRRADIATION OF ONE METRIC TON OF PWRU FUEL

\+ FISSION PRODUCTS

POWER= 1.00000E+00 MW, BURNUP= 1.00000E+00 MWD, FLUX= 1.00E+00 N/CM**2-SEC

0 7 SUMMARY TABLE: RADIOACTIVITY, CURIES

	4.0HR	8.0HR	12.0HR	16.0HR	20.0HR	24.0HR	28.0HR	32.0HR	36.0HR	40.0HR	44.0HR	48.0HR
TE131M	1.199E+05	1.093E+05	9.968E+04	9.088E+04	8.286E+04	7.554E+04	6.887E+04	6.279E+04	5.725E+04	5.220E+04	4.759E+04	4.339E+04
I131	9.553E+05	9.433E+05	9.313E+05	9.194E+05	9.075E+05	8.957E+05	8.840E+05	8.723E+05	8.607E+05	8.492E+05	8.378E+05	8.265E+05
XE131M	1.065E+04	1.065E+04	1.065E+04	1.065E+04	1.064E+04	1.064E+04	1.063E+04	1.062E+04	1.061E+04	1.060E+04	1.059E+04	1.057E+04
TE132	1.340E+06	1.293E+06	1.248E+06	1.205E+06	1.163E+06	1.122E+06	1.083E+06	1.046E+06	1.009E+06	9.740E+05	9.400E+05	9.073E+05
I132	1.372E+06	1.330E+06	1.285E+06	1.241E+06	1.198E+06	1.156E+06	1.116E+06	1.077E+06	1.040E+06	1.003E+06	9.685E+05	9.348E+05
CS132	7.294E+01	7.165E+01	7.039E+01	6.914E+01	6.792E+01	6.672E+01	6.554E+01	6.438E+01	6.324E+01	6.212E+01	6.102E+01	5.994E+01
TE133	6.923E+03	3.436E+02	1.706E+01	8.470E-01	4.205E-02	2.088E-03	1.037E-04	5.146E-06	2.555E-07	1.268E-08	6.298E-10	3.127E-11
TE133M	4.127E+04	2.049E+03	1.017E+02	5.051E+00	2.508E-01	1.245E-02	6.181E-04	3.069E-05	1.524E-06	7.564E-08	3.756E-09	1.865E-10
I133	1.862E+06	1.632E+06	1.428E+06	1.250E+06	1.094E+06	9.573E+05	8.378E+05	7.333E+05	6.418E+05	5.617E+05	4.916E+05	4.302E+05
XE133	2.073E+06	2.066E+06	2.055E+06	2.040E+06	2.021E+06	2.000E+06	1.977E+06	1.951E+06	1.925E+06	1.897E+06	1.867E+06	1.837E+06
XE133M	6.257E+04	6.193E+04	6.101E+04	5.986E+04	5.851E+04	5.702E+04	5.542E+04	5.373E+04	5.199E+04	5.020E+04	4.840E+04	4.660E+04
TE134	3.543E+04	6.621E+02	1.237E+01	2.313E-01	4.323E-03	8.079E-05	1.510E-06	2.822E-08	5.274E-10	9.857E-12	1.846E-13	3.562E-15
I134	2.716E+05	1.473E+04	6.839E+02	3.007E+01	1.294E+00	5.513E-02	2.340E-03	9.916E-05	4.198E-06	1.777E-07	7.520E-09	3.182E-10
CS134	1.593E+04	1.593E+04	1.593E+04	1.592E+04	1.592E+04	1.592E+04	1.592E+04	1.591E+04	1.591E+04	1.591E+04	1.591E+04	1.590E+04
CS134M	4.890E+03	1.880E+03	7.226E+02	2.778E+02	1.068E+02	4.105E+01	1.578E+01	6.065E+00	2.331E+00	8.962E-01	3.445E-01	1.324E-01
I135	1.271E+06	8.353E+05	5.492E+05	3.611E+05	2.374E+05	1.561E+05	1.026E+05	6.746E+04	4.435E+04	2.916E+04	1.917E+04	1.260E+04
XE135	8.399E+05	8.902E+05	8.344E+05	7.322E+05	6.167E+05	5.052E+05	4.057E+05	3.209E+05	2.509E+05	1.944E+05	1.495E+05	1.143E+05
XE135M	2.035E+05	1.338E+05	8.797E+04	5.784E+04	3.802E+04	2.500E+04	1.644E+04	1.081E+04	7.104E+03	4.671E+03	3.071E+03	2.019E+03
CS135	1.283E-01	1.285E-01	1.286E-01	1.287E-01	1.288E-01	1.289E-01	1.289E-01	1.290E-01	1.290E-01	1.290E-01	1.291E-01	1.291E-01
CS135M	1.061E+02	4.599E+00	1.993E-01	8.636E-03	3.742E-04	1.622E-05	7.028E-07	3.046E-08	1.320E-09	5.720E-11	2.479E-12	1.072E-13
BA135M	1.898E+00	1.723E+00	1.564E+00	1.420E+00	1.289E+00	1.170E+00	1.063E+00	9.648E-01	8.759E-01	7.953E-01	7.220E-01	6.555E-01
CS136	1.622E+04	1.608E+04	1.594E+04	1.580E+04	1.566E+04	1.552E+04	1.539E+04	1.525E+04	1.512E+04	1.498E+04	1.485E+04	1.472E+04

1 OUTPUT UNIT = 6 PAGE 2

Figura 5.13: Radioatividade para I^{135} e o Xe^{135} em uma saída típica do ORIGEN 2.1.

Nuclídeo/Tempo	4h	8h	12h	16h	20h	24h
I^{135}	1,271e6	8,353e5	5,492e5	3,611e5	2,374e5	1,561e5
Xe^{135}	8,399e5	8,902e5	8,344e5	7,322e5	6,167e5	5,052e5
Nuclídeo/Tempo	28h	32h	36h	40h	44h	48h
I^{135}	1,026e5	6,746e4	4,435e4	2,916e4	1,917e4	1,26e4
Xe^{135}	4,057e5	3,209e5	2,509e5	1,944e5	1,495e5	1,143e5

Tabela 5.2: Radioatividade para I^{135} e o Xe^{135} dados retirados diretamente do TFORI.

```
1                                                                              OUTPUT UNIT =  6                          PAGE    2
 ORIGEN2 V2.1 (8-1-91), Run on ôä"□□□□□□□□ at øä"□ªCB□

     *  IRRADIATION OF ONE METRIC TON OF PWRU FUEL
+                                                                                                  FISSION PRODUCTS
     POWER= 1.00000E+00 MW,  BURNUP= 1.00000E+00 MWD,  FLUX= 1.00E+00 N/CM**2-SEC
0                     5 SUMMARY TABLE:  CONCENTRATIONS, GRAMS
                              □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
              25.0HR    50.0HR    75.0HR   100.0HR   125.0HR   150.0HR   175.0HR   200.0HR   225.0HR   250.0HR   275.0HR   300.0HR
 BA140     2.409E+01 2.277E+01 2.152E+01 2.034E+01 1.922E+01 1.816E+01 1.717E+01 1.622E+01 1.533E+01 1.449E+01 1.370E+01 1.295E+01
 LA140     3.331E+00 3.237E+00 3.117E+00 2.984E+00 2.844E+00 2.704E+00 2.566E+00 2.432E+00 2.303E+00 2.179E+00 2.061E+00 1.950E+00
 CE140     3.615E+02 3.629E+02 3.643E+02 3.656E+02 3.668E+02 3.680E+02 3.692E+02 3.702E+02 3.713E+02 3.722E+02 3.731E+02 3.740E+02
 LA141     4.120E-03 5.015E-05 6.103E-07 7.429E-09 9.042E-11 1.100E-12 1.339E-14 1.630E-16 1.984E-18 2.414E-20 2.987E-22 1.227E-23
 CE141     6.118E+01 5.984E+01 5.852E+01 5.724E+01 5.598E+01 5.475E+01 5.355E+01 5.237E+01 5.122E+01 5.010E+01 4.900E+01 4.792E+01
 PR141     2.970E+02 2.983E+02 2.996E+02 3.009E+02 3.022E+02 3.034E+02 3.046E+02 3.058E+02 3.069E+02 3.080E+02 3.091E+02 3.102E+02
 LA142     1.840E-06 2.477E-11 3.333E-16 4.485E-21 6.036E-26 8.123E-31 1.093E-35 1.470E-40 0.000E+00 0.000E+00 0.000E+00 0.000E+00
 CE142     3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02 3.590E+02
 PR142     5.589E-03 2.259E-03 9.131E-04 3.691E-04 1.492E-04 6.030E-05 2.438E-05 9.853E-06 3.983E-06 1.610E-06 6.507E-07 2.630E-07
 ND142     1.378E+00 1.381E+00 1.382E+00 1.383E+00 1.383E+00 1.383E+00 1.383E+00 1.383E+00 1.383E+00 1.383E+00 1.383E+00 1.383E+00
 CE143     1.528E+00 9.039E-01 5.347E-01 3.162E-01 1.871E-01 1.106E-01 6.544E-02 3.871E-02 2.290E-02 1.354E-02 8.010E-03 4.738E-03
 PR143     2.451E+01 2.385E+01 2.297E+01 2.199E+01 2.098E+01 1.996E+01 1.897E+01 1.802E+01 1.710E+01 1.622E+01 1.539E+01 1.459E+01
 ND143     3.088E+02 3.101E+02 3.113E+02 3.125E+02 3.136E+02 3.147E+02 3.158E+02 3.168E+02 3.177E+02 3.186E+02 3.194E+02 3.202E+02
 CE144     2.400E+02 2.394E+02 2.388E+02 2.381E+02 2.375E+02 2.369E+02 2.363E+02 2.357E+02 2.351E+02 2.345E+02 2.339E+02 2.334E+02
 PR144     1.013E-02 1.011E-02 1.008E-02 1.006E-02 1.003E-02 1.000E-02 9.979E-03 9.954E-03 9.929E-03 9.904E-03 9.878E-03 9.853E-03
 PR144M    5.065E-05 5.052E-05 5.040E-05 5.027E-05 5.014E-05 5.001E-05 4.989E-05 4.976E-05 4.963E-05 4.951E-05 4.938E-05 4.926E-05
 ND144     1.131E+02 1.137E+02 1.143E+02 1.149E+02 1.155E+02 1.161E+02 1.167E+02 1.173E+02 1.179E+02 1.185E+02 1.191E+02 1.197E+02
 PR145     1.766E-02 9.740E-04 5.373E-05 2.964E-06 1.635E-07 9.017E-09 4.974E-10 2.744E-11 1.513E-12 8.348E-14 4.605E-15 2.540E-16
 ND145     2.367E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02 2.368E+02
 ND146     1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02 1.960E+02
 PM146     9.308E-04 9.305E-04 9.301E-04 9.298E-04 9.295E-04 9.291E-04 9.288E-04 9.285E-04 9.281E-04 9.278E-04 9.275E-04 9.271E-04
 SM146     1.488E-04 1.489E-04 1.490E-04 1.491E-04 1.493E-04 1.494E-04 1.495E-04 1.496E-04 1.497E-04 1.499E-04 1.500E-04 1.501E-04
 ND147     8.070E+00 7.560E+00 7.082E+00 6.635E+00 6.215E+00 5.823E+00 5.455E+00 5.110E+00 4.787E+00 4.484E+00 4.201E+00 3.936E+00
 PM147     9.798E+01 9.842E+01 9.882E+01 9.919E+01 9.954E+01 9.986E+01 1.001E+02 1.004E+02 1.007E+02 1.009E+02 1.011E+02 1.013E+02
 SM147     9.601E+00 9.675E+00 9.749E+00 9.824E+00 9.898E+00 9.974E+00 1.005E+01 1.012E+01 1.020E+01 1.028E+01 1.035E+01 1.043E+01
 ND148     1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02 1.127E+02
 PM148     8.255E-01 7.224E-01 6.322E-01 5.534E-01 4.844E-01 4.241E-01 3.714E-01 3.253E-01 2.850E-01 2.498E-01 2.190E-01 1.920E-01
 PM148M    8.866E-01 8.712E-01 8.561E-01 8.413E-01 8.267E-01 8.123E-01 7.983E-01 7.844E-01 7.708E-01 7.575E-01 7.443E-01 7.314E-01
 SM148     1.902E+01 1.914E+01 1.924E+01 1.934E+01 1.942E+01 1.950E+01 1.956E+01 1.962E+01 1.968E+01 1.973E+01 1.977E+01 1.981E+01
 ND149     1.346E-06 6.009E-11 2.683E-15 1.198E-19 5.350E-24 2.389E-28 1.067E-32 4.763E-37 2.130E-41 0.000E+00 0.000E+00 0.000E+00
 PM149     8.494E-01 6.128E-01 4.422E-01 3.190E-01 2.302E-01 1.661E-01 1.198E-01 8.644E-02 6.237E-02 4.500E-02 3.246E-02 2.342E-02
 SM149     2.899E+00 3.135E+00 3.306E+00 3.429E+00 3.518E+00 3.582E+00 3.628E+00 3.662E+00 3.686E+00 3.703E+00 3.716E+00 3.725E+00
```

Figura 5.14: Composição em gramas para o Sm^{149} em uma saída típica do ORIGEN.

Nuclídeo/Tempo	4h	8h	12h	16h	20h	24h
Sm^{149}	2,899	3,135	3,306	3,429	3,518	3,582
Nuclídeo/Tempo	28h	32h	36h	40h	44h	48h
Sm^{149}	3,628	3,662	3,686	3,703	3,716	3,725

Tabela 5.3: Composição em gramas para o Sm^{149} dados retirados diretamente do TFORI.

6 APLICAÇÃO DA METODOLOGIA

Nesta seção será feita uma aplicação do programa desenvolvido para um caso em que os dados são conhecidos, entretanto haverá um desligamento anormal simulando uma parada de emergência. Além disso, uma aproximação será feita para os fatores de pico axiais, uma vez que não são conhecidos esses valores.

6.1 Descrição do reator

O reator simulado é um PWR típico com 193 elementos combustíveis arranjados num sistema de baixa fuga de nêutrons com nove regiões de composições similares de composição isotópica inicial, descritas na Tabela 6.1. As características gerais deste reator estão apresentadas na Tabela 6.2. Entretanto, pela sua simetria serão simulados apenas 56 elementos cada um dividido em 10 células resultando em 560 células.

Região	A	B	C	D	E	F	G	H	I
Enriquecimento inicial [w/o ^{235}U]	3,75	1,91	3,78	4,02	3,96	3,99	3,94	4,00	3,94
Massa de metal pesado inicial por Elemento combustível	540,8	544,4	539,9	542,0	541,5	543,0	540,4	542,7	539,9
Número de elementos combustíveis	24	1	4	20	32	20	36	20	36

Tabela 6.1: Descrição das regiões do reator simulado.

Para este reator será simulado um histórico de 210 dias operando com um fator de carga 100% . Após este dia será simulado um desligamento de emergência e será acompanhada a evolução dos nuclídeos durante as 6 horas subsequentes à este desligamento em intervalos de trinta minutos.

Característica	Módulo	Grandeza
Potência térmica	3765	MW
Quantidade de elementos combustíveis	193	
Comprimento ativo da vareta	390	cm
Tamanho do ciclo	366	Dias

Tabela 6.2: Características gerais do reator

6.1.1 Distribuições de potência

Para que sejam calculadas coordenadas especificas no espaço tridimensional de um reator é necessário levar em consideração a densidade linear local de potência (potência local por unidade de comprimento). No relatório disponibilizado para este trabalho não há informações sobre os fatores de distribuição de potência axiais, sendo assim, o programa irá calculá-los.

A distribuição radial de potência dos elementos combustíveis no núcleo do reator estudado foi dada previamente pelo usuário para um quarto deste reator devido à simetria, conforme apresentado na Figura 6.1.

6.2 Estudo de caso

O exemplo utilizado, como dito na seção anterior, trata de um reator PWR com BURNUP de 50,000 MWD/MTU, 193 elementos combustíveis, 9 regiões de mesma composição de massa de metal pesado, comprimento ativo da vareta de 390 cm, potência térmica nominal de 3765 MW, 396 dias efetivos de queima, operando com urânio levemente enriquecido com variação de 3 a 5% de U^{235}.

	1	2	3	4	5	6	7	8
P					A 0.26	E 0.26	A 0.40	C 0.44
O			A 0.24	E 0.50	H 1.13	I 1.08	H 1.37	I 1.19
N		A 0.24	H 0.99	I 1.10	F 1.18	E 0.98	E 1.01	E 1.21
M		E 0.49	I 1.09	D 0.99	G 1.30	G 1.30	I 1.35	G 1.26
L	A 0.26	H 1.13	F 1.18	G 1.30	G 1.32	F 1.36	D 1.11	G 1.25
K	E 0.38	I 1.08	E 0.98	G 1.30	F 1.36	D 1.11	I 1.35	G 1.28
J	A 0.40	H 1.37	E 1.01	I 1.35	D 1.11	I 1.35	F 1.25	D 0.93
H	C 0.44	I 1.19	G 1.21	G 1.26	G 1.25	G 1.28	D 0.93	B 0.26

X
Y

X = Região

Y = Fator Radial

Figura 6.1: Distribuição radial de potência normalizada do núcleo.

Os dados de fatores de distribuição axial foram aproximados por uma curva senoidal conforme descrito na seção 3.2.5.

Essas regiões foram escolhidas porque possuem elementos combustíveis que foram recém inseridos no núcleo. Os dados analisados conforme dito acima são apenas os dos actinídeos U-235, U-236, U-238, Pu-238, Pu-239, Pu-240, Pu-241 e Pu-242, pois estes são os únicos dados encontrados na referência.

Como o TFORI realizou o cálculo para cada elemento combustível de duas regiões serão apresentados os dados desses actinídeos para os elementos combustíveis. Estes dados totalizam 40 células oriundas de 4 elementos combustíveis.

A nomenclatura dessas células são representadas por uma fórmula geral *Região_Elemento_Célula*. Por exemplo, "10G6_G035_5" refere-se à região "10G6", elemento combustível "G035"e à célula "5". Esses dados são apresentados na Tabela 6.3, Tabela 6.4, Tabela 6.5 e Tabela 6.6.

Célula	10_C011_1	10_C011_2	10_C011_3	10_C011_4	10_C011_5	10_C011_6	10_C011_7	10_C011_8	10_C011_9	10_C011_10
U235	0,056	23,309	18,537	15,823	14,951	15,823	18,537	23,309	30,348	40,004
U236	1,795	3,035	3,820	4,242	4,371	4,242	3,820	3,035	1,795	0,000
U238	954,118	948,959	944,537	941,404	940,299	941,404	944,537	948,959	954,118	959,646
Pu239	3,197	4,459	4,959	5,159	5,209	5,159	4,959	4,459	3,197	0,000
Pu240	0,000	1,048	1,488	1,743	1,818	1,743	1,488	1,048	0,000	0,000
Pu241	0,000	0,000	0,000	1,132	0,121	1,132	0,000	0,000	0,000	0,000

Tabela 6.3: Composição isotópica em g/kg do elemento C011 calculados pelo TFORI.

Célula	10_C019_1	10_C019_2	10_C019_3	10_C019_4	10_C019_5	10_C019_6	10_C019_7	10_C019_8	10_C019_9	10_C019_10
U235	28,561	21,006	15,288	12,464	11,524	12,430	15,288	21,006	28,561	40,004
U236	2,114	3,499	4,321	4,723	4,842	4,723	4,321	3,499	2,114	0,000
U238	953,013	946,379	940,851	936,797	935,323	936,797	940,851	946,379	953,013	959,646
Pu239	3,580	4,780	5,191	5,316	5,353	5,316	5,191	4,780	3,580	0,000
Pu240	0,000	1,305	1,789	2,027	2,106	2,027	1,789	1,305	0,000	0,000
Pu241	0,000	0,000	1,179	1,436	1,511	1,466	1,179	0,000	0,000	0,000

Tabela 6.4: Composição isotópica em g/kg do elemento C019 calculados pelo TFORI

Célula	10G6_C035_1	10G6_C035_2	10G6_C035_3	10G6_C035_4	10G6_C035_5	10G6_C035_6	10G6_C035_7	10G6_C035_8	10G6_C035_9	10G6_C035_10
U235	30,117	23,319	18,726	16,079	15,231	16,079	18,726	23,319	30,117	39,396
U236	1,727	2,925	3,688	4,101	4,229	4,101	3,688	2,925	1,727	0,000
U238	954,806	949,806	945,545	942,582	941,656	942,582	945,545	949,806	954,806	960,178
Pu239	3,143	4,410	4,921	5,129	5,182	5,129	4,921	4,410	3,143	0,000
Pu240	0,000	1,012	1,445	1,698	1,806	1,698	1,445	1,012	0,000	0,000
Pu241	0,000	0,000	0,000	1,089	1,166	1,089	0,000	0,000	0,000	0,000

Tabela 6.5: Composição isotópica em g/kg do elemento C035 calculados pelo TFORI.

Célula	10G6_C043_1	10G6_C043_2	10G6_C043_3	10G6_C043_4	10G6_C043_5	10G6_C043_6	10G6_C043_7	10G6_C043_8	10G6_C043_9	10G6_C043_10
U235	28,116	20,189	15,038	12,228	12,228	12,228	15,038	20,189	28,116	39,396
U236	2,087	3,449	4,256	4,653	4,769	4,653	4,256	3,449	2,087	0,000
U238	953,510	947,027	941,285	937,396	935,914	937,396	941,285	947,027	953,510	960,178
Pu239	3,590	4,786	5,194	5,320	5,355	5,320	5,194	4,786	3,590	0,000
Pu240	0,000	1,307	1,790	2,026	2,102	2,026	1,790	1,307	0,000	0,000
Pu241	0,000	0,000	1,183	1,440	1,515	1,440	1,183	0,000	0,000	0,000

Tabela 6.6: Composição isotópica em g/kg do elemento C043 calculados pelo TFORI.

Embora os dados apresentados sejam de apenas quatro elementos foram feitos os cálculos para todas as 560 células resultando em 5 ou 6 minutos de processamento. O estudo de caso apresentado aqui apenas ilustra uma aplicação dada ao TFORI.

7. CONCLUSÃO E PERSPECTIVAS FUTURAS

A fim de contribuir com as ferramentas disponíveis para o acompanhamento de situações emergenciais passíveis de acontecer em reatores tipo PWR foi desenvolvida uma ferramenta que utiliza o código ORIGEN2.1 para suportar tridimensionalmente o termo fonte e suas características como radioatividade total e alfa, ratiotoxicidade de inalação e ingestão, toxicidade química etc. Tal programa, denominado TFORI, tem como objetivo operar como uma interface que calcula a evolução isotópica de combustíveis durante qualquer período seguinte ao desligamento de um reator tipo PWR sendo que este cálculo pode ser feito para o reator como um todo ou para regiões deste reator, dada a composição isotópica mais recente das células. Caso não haja o histórico recente deste reator e os respectivos fatores de distribuição axial e radial, o programa é capaz de calcular a distribuição axial através de uma aproximação com uma curva senoidal e efetuar a queima desde o momento do último registro do histórico até a data da parada. O cálculo é realizado através do ORIGEN 2.1 já que este apresenta as características necessárias para uma avaliação em situação emergencial: (a) ter baixo gasto computacional, (b) calcular o termo fonte e suas características e (c) fornecer dados de uma ampla variedade de nuclídeos. As características deste termo fonte poderiam ser usadas para o cálculo de dose em uma população afetada.

A metodologia para o desenvolvimento do TFORI atendeu aos requisitos necessários para executar o ORIGEN 2.1 e avaliar as suas saídas por representações gráficas. Testes de verificação foram feitos comparando as entradas produzidas pelo TFORI com entradas exemplos distribuídas pelo OAK RIDGE LABORATORY no pacote do ORIGEN2.1. Em seguida, foi comprovado que não houve perda de qualidade nas saídas baseadas na entrada construída pelo TFORI.

Por fim, foi simulado um caso exemplo no TFORI e os resultados apresentaram uma boa concordância com a proposta de reduzir o tempo de processamento, uma vez que foram necessários apenas 6 minutos para concluir a execução do TFORI. Esse valor é condizente com uma rápida tomada de decisões. E como as saídas do ORIGEN para cada célula são preservadas elas podem ser utilizadas por outros programas relacionados à segurança do ambiente.

Citamos algumas recomendações que podem contribuir para o aprimoramento do programa:

- Implementação da possibilidade dos resultados serem reaproveitados para fazer uma extensão no período de decaimento. Sendo que o ORIGEN 2.1 é capaz de fazer essa realimentação de dados;
- O tempo de decaimento e seus intervalos poderiam ser controlados pelo usuário, isso possibilitaria a análise da composição em condições mais adequadas ao ambiente de trabalho desejado por ele;
- Oferecer ao usuário a possibilidade de fazer análises gráficas de qualquer saída produzida pelo ORIGEN2.1 e não somente daquelas produzidas com o auxílio do TFORI;
- Possibilitar ao TFORI armazenar os fatores de distribuição de potência axial e radial em arquivos a fim de serem utilizados em outras simulações sem a necessidade de um usuário redigitá-las;
- Ao lidar com as saídas possibilitar que várias curvas de diversas células sejam projetadas em um único gráfico.
- Permirtir que as saídas produzidas pelo TFORI sejam utilizadas em outros códigos como, por exemplo, códigos de dispersão de pluma.
- Executar o TFORI em paralelo à múltiplos computadores.

REFERÊNCIAS

1. Bennett, G.L., 1991. The safety review and approval process for space nuclear-power sources. Nuclear Safety 32 (1), 1–18.
2. U.S. Nuclear Regulatory commission; “Accident Source Terms for Light-Waters Nuclear Power Plants” NUREG-1465, (1995).
3. Croff, A. G., A User's Manual for the ORIGEN2 Computer Code, Oak Ridge Laboratory, Oak Ridge Tenessee (1980).
4. Croff, A. G., "ORIGEN2: A Versatile Computer Code for Calculating the Nuclide Compositions and Characteristics of Nuclear Materials", Nuclear Technology, Vol. 62, pp. 335- 352, 1983.
5. X-5 Monte Carlo Team, “MCNP – A General N-Particle Transport Code, Version 5” Volume I: Overview and Theory, LA-UR-03-1987 (2003).
6. L.S. Waters, Ed., "MCNPX User's Manual, Version 2.3.0", LA-UR-02-2607 (2002).
7. SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluations, ORNL/TM-2005/39, Version 5, Vols. I-III, April 2005.
8. Ludwig, S. “Revision to ORIGEN2 – Version 2.2”, memorando de transmissão 23 de Maio de 2002.
9. Hermann, O.W. and R.M. Westfall. ORIGEN-S – SCALE System Module to Calculate Fuel Depletion, Actinide Transmutation, Fission Product Buildup and Decay, and Associated Radiation Source Terms, in SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluations, NUREG/CR-0200, Rev. 4 (ORNL/NUREG/CSD-2/R4), Vol. II, PART I, 1995.
10. Bowman et al, “ORIGEN-ARP A Fast and Easy-to-Use Source Term Generation Tool” ICRS-9 Ninth International Conference on radiation Shielding, (1999).
11. BATEMAN H. “The solution of a system of differential equations occurring in the theory of radio-active transformations.” Proc. Cambridge Phil. Soc. 16, 423, 1910.

12. Hahn, Brian and Valentine, Daniel. “Essential MATLAB For Engineers and Scientits”Butherworth-Heinemann, Elsevier, Oxford, terceira edição, 2007.
13. Cantrel, L. et al, “Source term computation with ASTEC code”, Institut de Radioprotection et Süreté Nucléaire, Major Accident Prevetion Division (DPMA), SEMIC, LETR.
14. K. Herviou et al, “Development of a methodology and of a computer tool for source term estimation in case of nuclear emergency in a light water reactor (ASTRID),”, IRSN, France, 2005.
15. M. Khatib-Rahbar, M. Zavisca, H. Esmaili, C. E. G, U. Schmocker, G. Schoen and R. Schulz, “Accident diagnostic, analysis and management (ADAM) system applications to severe accident management” Severe accident management (SAM) on operator training and instrumentation capabilities, Lyon, 2001.
16. J. R. Alonso, S. Aleza, M. Alonso and G. Carmen, “Consejo de seguridad nuclear use and experience with the MARS software” in the 11th international topical meeting on nuclear reactor thermal-hydraulics (NURETH-11), Avignon, 2005
17. Gauntt, R., et al., “MELCOR Computer Code Manuals, NUREG/CR-6119”, SAND2005 - 5713, US Nuclear Regulatory Commission, Washington, D.C. 2005.
18. Rodriguez, Sal B. et al., “Addition of Secondary System Modules and a Graphical User Interface into MELCOR-H2—Phase 1”, SAND2006-4157 C, US Nuclear Regulatory Commission, Washington, D.C. 2007
19. Duderstadt, A. J., Hamilton, J. L. Nuclear Reactor Analysis, Department of Nuclear engineering, The University of Michigan, Ann Arbor, Michigan. (1976).
20. Site oficial do MinGw. Disponível em: <http://www.mingw.org/> Acesso em 13 jun. 2011.

Apêndice 1 – Programa TFORI

Parte – 1:

```
function varargout = ORIG3N_1(varargin)
% ORIG3N_1 M-file for ORIG3N_1.fig
%      ORIG3N_1, by itself, creates a new ORIG3N_1 or raises the
existing
%      singleton*.
%
%      H = ORIG3N_1 returns the handle to a new ORIG3N_1 or the handle
to
%      the existing singleton*.
%
%      ORIG3N_1('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in ORIG3N_1.M with the given input
arguments.
%
%      ORIG3N_1('Property','Value',...) creates a new ORIG3N_1 or
raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before ORIG3N_1_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to ORIG3N_1_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help ORIG3N_1

% Last Modified by GUIDE v2.5 28-May-2012 10:49:09

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ORIG3N_1_OpeningFcn, ...
                   'gui_OutputFcn',  @ORIG3N_1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before ORIG3N_1 is made visible.
function ORIG3N_1_OpeningFcn(hObject, eventdata, handles, varargin)
assignin('base', 'kh', 1);
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ORIG3N_1 (see VARARGIN)

% Choose default command line output for ORIG3N_1
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ORIG3N_1 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ORIG3N_1_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function Potencia_Callback(hObject, eventdata, handles)
% hObject    handle to Potencia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Potencia as text
%        str2double(get(hObject,'String')) returns contents of
Potencia as a double


% --- Executes during object creation, after setting all properties.
function Potencia_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Potencia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function compVar_Callback(hObject, eventdata, handles)
% hObject    handle to compVar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of compVar as text
%        str2double(get(hObject,'String')) returns contents of compVar
as a double


% --- Executes during object creation, after setting all properties.
function compVar_CreateFcn(hObject, eventdata, handles)
% hObject    handle to compVar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function SegZ_Callback(hObject, eventdata, handles)
% hObject    handle to SegZ (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of SegZ as text
%        str2double(get(hObject,'String')) returns contents of SegZ as
a double


% --- Executes during object creation, after setting all properties.
function SegZ_CreateFcn(hObject, eventdata, handles)
% hObject    handle to SegZ (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function QteElem_Callback(hObject, eventdata, handles)
% hObject    handle to QteElem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of QteElem as text
%        str2double(get(hObject,'String')) returns contents of QteElem
as a double


% --- Executes during object creation, after setting all properties.
function QteElem_CreateFcn(hObject, eventdata, handles)
% hObject    handle to QteElem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function QteReg_Callback(hObject, eventdata, handles)
% hObject    handle to QteReg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of QteReg as text
%        str2double(get(hObject,'String')) returns contents of QteReg
as a double


% --- Executes during object creation, after setting all properties.
function QteReg_CreateFcn(hObject, eventdata, handles)
% hObject    handle to QteReg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
ASeguinte


function ciclo_Callback(hObject, eventdata, handles)
% hObject    handle to ciclo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of ciclo as text
%        str2double(get(hObject,'String')) returns contents of ciclo
as a double


% --- Executes during object creation, after setting all properties.
function ciclo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ciclo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listbox1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
listbox1


% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes when selected object is changed in uipanel2.
function uipanel2_SelectionChangeFcn(hObject, eventdata, handles)
switch get(eventdata.NewValue,'Tag') % Get Tag of selected object.
    case 'radiobutton1'
        Th=1;
        assignin('base','Th',Th);
    case 'radiobutton2'
        Th=2;
        assignin('base','Th',Th);
    case 'radiobutton3'
        Th=3;
```

```
        assignin('base','Th',Th);
    case 'radiobutton4'
        Th=4;
        assignin('base','Th',Th);
end
% hObject    handle to the selected object in uipanel2
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if
none was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function uipanel2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to uipanel2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called



function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of edit12
as a double


% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as text
%        str2double(get(hObject,'String')) returns contents of edit13
as a double
```

```
% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
AproxVal
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function load_Callback(hObject, eventdata, handles)
% hObject    handle to load (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of load as text
%        str2double(get(hObject,'String')) returns contents of load as
a double


% --- Executes during object creation, after setting all properties.
function load_CreateFcn(hObject, eventdata, handles)
% hObject    handle to load (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function dia_Callback(hObject, eventdata, handles)
% hObject    handle to dia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of dia as text
```

```
%        str2double(get(hObject,'String')) returns contents of dia as
a double


% --- Executes during object creation, after setting all properties.
function dia_CreateFcn(hObject, eventdata, handles)
% hObject    handle to dia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Parte – 2:

```
function varargout = ORIG3N_2(varargin)
% ORIG3N_2 M-file for ORIG3N_2.fig
%      ORIG3N_2, by itself, creates a new ORIG3N_2 or raises the
existing
%      singleton*.
%
%      H = ORIG3N_2 returns the handle to a new ORIG3N_2 or the handle
to
%      the existing singleton*.
%
%      ORIG3N_2('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in ORIG3N_2.M with the given input
arguments.
%
%      ORIG3N_2('Property','Value',...) creates a new ORIG3N_2 or
raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before ORIG3N_2_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to ORIG3N_2_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help ORIG3N_2

% Last Modified by GUIDE v2.5 30-Aug-2012 01:05:01

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
```

```
                   'gui_OpeningFcn', @ORIG3N_2_OpeningFcn, ...
                   'gui_OutputFcn',  @ORIG3N_2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before ORIG3N_2 is made visible.
function ORIG3N_2_OpeningFcn(hObject, eventdata, handles, varargin)

nome='RegId'; %Raiz do nome das variáveis.
qteReg=evalin('base','qteReg'); %resgata o valor da quantidade de
regiões.
assignin('base','ko',1);
assignin('base','kn',1);
assignin('base','kReg',1);

for i=1:qteReg
    b=num2str(i); %transforma o contador numa palavra
    name=[nome,b];%une a raiz do nome ao sufixo da variável
    ncell{i}=name;%armazena os nomes numa célula
end
assignin('base', 'ncell',ncell);


% Choose default command line output for ORIG3N_2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ORIG3N_2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ORIG3N_2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function regId_Callback(hObject, eventdata, handles)
% hObject    handle to regId (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of regId as text
%        str2double(get(hObject,'String')) returns contents of regId
as a double


% --- Executes during object creation, after setting all properties.
function regId_CreateFcn(hObject, eventdata, handles)
% hObject    handle to regId (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function MMP_Callback(hObject, eventdata, handles)
% hObject    handle to MMP (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of MMP as text
%        str2double(get(hObject,'String')) returns contents of MMP as
a double


% --- Executes during object creation, after setting all properties.
function MMP_CreateFcn(hObject, eventdata, handles)
% hObject    handle to MMP (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in seguinte.
function seguinte_Callback(hObject, eventdata, handles)
ORIG3N_3
close ORIG3N_2
% hObject    handle to seguinte (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on button press in okGo.
function okGo_Callback(hObject, eventdata, handles)
ko=num2double(get(handles.edit9, 'String'));
assingnin('base','ko',ko);
set(handles.text4,'string', ko);
% hObject    handle to okGo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9
as a double


% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in reedit.
function reedit_Callback(hObject, eventdata, handles)
% hObject    handle to reedit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function ZIso_Callback(hObject, eventdata, handles)
% hObject    handle to ZIso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ZIso as text
%        str2double(get(hObject,'String')) returns contents of ZIso as
a double


% --- Executes during object creation, after setting all properties.
function ZIso_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ZIso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function AIso_Callback(hObject, eventdata, handles)
% hObject    handle to AIso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AIso as text
%        str2double(get(hObject,'String')) returns contents of AIso as
a double


% --- Executes during object creation, after setting all properties.
function AIso_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AIso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function QIso_Callback(hObject, eventdata, handles)
% hObject    handle to QIso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of QIso as text
%        str2double(get(hObject,'String')) returns contents of QIso as
a double


% --- Executes during object creation, after setting all properties.
function QIso_CreateFcn(hObject, eventdata, handles)
% hObject    handle to QIso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listbox1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
listbox1


% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in delComp.
function delComp_Callback(hObject, eventdata, handles)
assignin('base','kReg',1);
% hObject    handle to delComp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in proxNucl.
function proxNucl_Callback(hObject, eventdata, handles)
BProxNucl
% hObject    handle to proxNucl (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in edIso.
function edIso_Callback(hObject, eventdata, handles)
BFimEdIso
% hObject    handle to edIso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
function ZIso2_Callback(hObject, eventdata, handles)
% hObject    handle to ZIso2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ZIso2 as text
%        str2double(get(hObject,'String')) returns contents of ZIso2
as a double


% --- Executes during object creation, after setting all properties.
function ZIso2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ZIso2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function AIso2_Callback(hObject, eventdata, handles)
% hObject    handle to AIso2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AIso2 as text
%        str2double(get(hObject,'String')) returns contents of AIso2
as a double


% --- Executes during object creation, after setting all properties.
function AIso2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AIso2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function QIso2_Callback(hObject, eventdata, handles)
% hObject    handle to QIso2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of QIso2 as text
```

```
%        str2double(get(hObject,'String')) returns contents of QIso2
as a double


% --- Executes during object creation, after setting all properties.
function QIso2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to QIso2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function ZIso3_Callback(hObject, eventdata, handles)
% hObject    handle to ZIso3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ZIso3 as text
%        str2double(get(hObject,'String')) returns contents of ZIso3
as a double


% --- Executes during object creation, after setting all properties.
function ZIso3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ZIso3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function AIso3_Callback(hObject, eventdata, handles)
% hObject    handle to AIso3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AIso3 as text
%        str2double(get(hObject,'String')) returns contents of AIso3
as a double


% --- Executes during object creation, after setting all properties.
function AIso3_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to AIso3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function QIso3_Callback(hObject, eventdata, handles)
% hObject    handle to QIso3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of QIso3 as text
%        str2double(get(hObject,'String')) returns contents of QIso3
as a double


% --- Executes during object creation, after setting all properties.
function QIso3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to QIso3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function ZIso4_Callback(hObject, eventdata, handles)
% hObject    handle to ZIso4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ZIso4 as text
%        str2double(get(hObject,'String')) returns contents of ZIso4
as a double


% --- Executes during object creation, after setting all properties.
function ZIso4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ZIso4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function AIso4_Callback(hObject, eventdata, handles)
% hObject    handle to AIso4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AIso4 as text
%        str2double(get(hObject,'String')) returns contents of AIso4
as a double


% --- Executes during object creation, after setting all properties.
function AIso4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AIso4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function QIso4_Callback(hObject, eventdata, handles)
% hObject    handle to QIso4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of QIso4 as text
%        str2double(get(hObject,'String')) returns contents of QIso4
as a double


% --- Executes during object creation, after setting all properties.
function QIso4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to QIso4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit19_Callback(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit19 as text
%        str2double(get(hObject,'String')) returns contents of edit19
as a double


% --- Executes during object creation, after setting all properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
BFimEdIso_XLS



function nomeXLS_Callback(hObject, eventdata, handles)
% hObject    handle to nomeXLS (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of nomeXLS as text
%        str2double(get(hObject,'String')) returns contents of nomeXLS
as a double


% --- Executes during object creation, after setting all properties.
function nomeXLS_CreateFcn(hObject, eventdata, handles)
% hObject    handle to nomeXLS (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


% --- Executes on button press in pushbutton12.
function pushbutton12_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
BFimEdIso_XLS
```

Parte – 3:

```
function varargout = ORIG3N_3(varargin)
% ORIG3N_3 M-file for ORIG3N_3.fig
%      ORIG3N_3, by itself, creates a new ORIG3N_3 or raises the
existing
%      singleton*.
%
%      H = ORIG3N_3 returns the handle to a new ORIG3N_3 or the handle
to
%      the existing singleton*.
%
%      ORIG3N_3('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in ORIG3N_3.M with the given input
arguments.
%
%      ORIG3N_3('Property','Value',...) creates a new ORIG3N_3 or
raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before ORIG3N_3_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to ORIG3N_3_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help ORIG3N_3

% Last Modified by GUIDE v2.5 30-Aug-2012 23:46:05

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ORIG3N_3_OpeningFcn, ...
                   'gui_OutputFcn',  @ORIG3N_3_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before ORIG3N_3 is made visible.
function ORIG3N_3_OpeningFcn(hObject, eventdata, handles, varargin)
assignin('base','ks',1);
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ORIG3N_3 (see VARARGIN)

% Choose default command line output for ORIG3N_3
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ORIG3N_3 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ORIG3N_3_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function ElId_Callback(hObject, eventdata, handles)
% hObject    handle to ElId (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ElId as text
%        str2double(get(hObject,'String')) returns contents of ElId as
a double


% --- Executes during object creation, after setting all properties.
function ElId_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ElId (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function regId_Callback(hObject, eventdata, handles)
% hObject    handle to regId (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of regId as text
%        str2double(get(hObject,'String')) returns contents of regId
as a double


% --- Executes during object creation, after setting all properties.
function regId_CreateFcn(hObject, eventdata, handles)
% hObject    handle to regId (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Ffa_Callback(hObject, eventdata, handles)
% hObject    handle to Ffa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Ffa as text
%        str2double(get(hObject,'String')) returns contents of Ffa as
a double


% --- Executes during object creation, after setting all properties.
function Ffa_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ffa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
CSeguinte
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in ProxElem.
function ProxElem_Callback(hObject, eventdata, handles)
CproxElem
% hObject    handle to ProxElem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4
as a double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
function FatorAxial_Callback(hObject, eventdata, handles)
% hObject    handle to FatorAxial (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of FatorAxial as text
%        str2double(get(hObject,'String')) returns contents of
FatorAxial as a double


% --- Executes during object creation, after setting all properties.
function FatorAxial_CreateFcn(hObject, eventdata, handles)
% hObject    handle to FatorAxial (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
CproxElem_XLS
```

Parte – 4:

```
function varargout = Pure(varargin)

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Pure_OpeningFcn, ...
                   'gui_OutputFcn',  @Pure_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Pure is made visible.
```

```
function Pure_OpeningFcn(hObject, eventdata, handles, varargin)

% Choose default command line output for Pure
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);


% --- Outputs from this function are returned to the command line.
function varargout = Pure_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;


% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
if hObject == handles.T1
    set (handles.saida,'String','T1');
    handles.teste=10;%tem de usar essa estrutura para reter a
informação.
    guidata(hObject, handles)%ao fim de cada informação retida é
necessário usar essa estrutura para poder usar a informação em outra
função.

elseif hObject == handles.T2
    set (handles.saida,'String','T2');
    handles.teste=20;
    guidata(hObject, handles)
elseif hObject == handles.T3
    set (handles.saida,'String','T3');
    handles.teste=30;
    guidata(hObject, handles)
elseif hObject == handles.T4
    set (handles.saida,'String','T4');
    handles.teste=40;
    guidata(hObject, handles)
elseif hObject == handles.T5
    set (handles.saida,'String','T5');
    handles.teste=50;
    guidata(hObject, handles)
elseif hObject == handles.T6
    set (handles.saida,'String','T6');
    handles.teste=60;
    guidata(hObject, handles)
elseif hObject == handles.T7
    set (handles.saida,'String','T7');
    handles.teste=70;
    guidata(hObject, handles)
elseif hObject == handles.T8
    set (handles.saida,'String','T8');
    handles.teste=80;
    guidata(hObject, handles)
elseif hObject == handles.T9
    set (handles.saida,'String','T9');
    handles.teste=90;
    guidata(hObject, handles)
elseif hObject == handles.T10
    set (handles.saida,'String','T10');
    handles.teste=100;
    guidata(hObject, handles)
```

```
elseif hObject == handles.T11
    set (handles.saida,'String','T11');
    handles.teste=110;
    guidata(hObject, handles)
elseif hObject == handles.T12
    set (handles.saida,'String','T12');
    handles.teste=120;
    guidata(hObject, handles)
elseif hObject == handles.T13
    set (handles.saida,'String','T13');
    handles.teste=130;
    guidata(hObject, handles)
elseif hObject == handles.T14
    set (handles.saida,'String','T14');
    handles.teste=140;
    guidata(hObject, handles)
elseif hObject == handles.T15
    set (handles.saida,'String','T15');
    handles.teste=150;
    guidata(hObject, handles)
elseif hObject == handles.T16
    set (handles.saida,'String','T16');
    handles.teste=160;
    guidata(hObject, handles)
elseif hObject == handles.T17
    set (handles.saida,'String','T17');
    handles.teste=170;
    guidata(hObject, handles)
elseif hObject == handles.T18
    set (handles.saida,'String','T18');
    handles.teste=180;
    guidata(hObject, handles)
elseif hObject == handles.T19
    set (handles.saida,'String','T19');
    handles.teste=190;
    guidata(hObject, handles)
elseif hObject == handles.T20
    set (handles.saida,'String','T20');
    handles.teste=200;
    guidata(hObject, handles)
elseif hObject == handles.T21
    set (handles.saida,'String','T21');
    handles.teste=210;
    guidata(hObject, handles)
elseif hObject == handles.T22
    set (handles.saida,'String','T22');
    handles.teste=220;
    guidata(hObject, handles)
elseif hObject == handles.T23
    set (handles.saida,'String','T23');
    handles.teste=230;
    guidata(hObject, handles)
elseif hObject == handles.T24
    set (handles.saida,'String','T24');
    handles.teste=240;
    guidata(hObject, handles)
else
    set (handles.saida,'String','ERROR');
    handles.teste=0;
    guidata(hObject, handles)
end
```

```
% --- Executes on button press in done.
function done_Callback(hObject, eventdata, handles)
if handles.teste==10
    set (handles.saida,'String','ok');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')

    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==20
        set (handles.saida,'String','ok2');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
        AcumVaria1_otimizado
        Plotter
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==30
        set (handles.saida,'String','ok3');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==40
        set (handles.saida,'String','ok4');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==50
        set (handles.saida,'String','ok5');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
```

```
    end
elseif handles.teste==60
        set (handles.saida,'String','ok6');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==70
        set (handles.saida,'String','ok7');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==80
        set (handles.saida,'String','ok8');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==90
        set (handles.saida,'String','ok9');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==100
        set (handles.saida,'String','ok10');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==110
        set (handles.saida,'String','ok11');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
```

```
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==120
        set (handles.saida,'String','ok12');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==130
        set (handles.saida,'String','ok13');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==140
        set (handles.saida,'String','ok14');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==150
        set (handles.saida,'String','ok15');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==160
        set (handles.saida,'String','ok16');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
```

```
elseif handles.teste==170
        set (handles.saida,'String','ok17');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==180
        set (handles.saida,'String','ok18');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==190
        set (handles.saida,'String','ok19');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==200
        set (handles.saida,'String','ok20');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==210
        set (handles.saida,'String','ok21');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==220
        set (handles.saida,'String','ok22');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
```

```
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==230
        set (handles.saida,'String','ok23');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
elseif handles.teste==240
        set (handles.saida,'String','ok24');
    if handles.grupoG == 1
        set (handles.saida2,'String','ok')
    elseif handles.grupoG == 2
        set (handles.saida2,'String','ok2')
    elseif handles.grupoG == 3
        set (handles.saida2, 'String', 'ok3')
    else
        set (handles.saida3, 'String', 'ERROR')
    end
else
        set (handles.saida,'String','ERROR');
        set (handles.saida2,'String','ERROR');
end


% --- Executes when selected object is changed in uipanel2.
function uipanel2_SelectionChangeFcn(hObject, eventdata, handles)
if hObject==handles.AP
    set (handles.saida2,'String','AP');
    handles.grupoG=1;
    guidata(hObject,handles)
elseif hObject==handles.AD
    set (handles.saida2,'String','AD');
    handles.grupoG=2;
    guidata(hObject,handles)
elseif hObject==handles.FP
    set (handles.saida2,'String','FP');
    handles.grupoG=3;
    guidata(hObject,handles)
else
    set (handles.saida2,'String','ERROR');
    handles.grupoG=0;
    guidata(hObject,handles)

end
```

Parte – 5:

```
s = A(n); %Armazena o elemento de A numa variável

label=cell2mat(s);% transforma a célula numa matriz de strings
```

```
if ~isempty(label)%verifica se existe algum valor
    nome=label(1:12);% separa o nome
    number = label (12:end);% separa os valores
else % evitar valores nulos.
    nome='0';
    number = '0';
end
```

Parte – 6 - AProxVal:

```
kh=evalin('base', 'kh');

potTok=['load' num2str(kh)];
load=get(handles.load, 'String');
assignin('base', potTok, eval(load));

diaTok=['dia' num2str(kh)];
dia=get(handles.dia, 'String');
assignin('base', diaTok, eval(dia));

set(handles.dia,'String','');

kh=kh+1;
assignin('base', 'kh', kh);
```

Parte – 7: - Aseguinte

```
potencia=str2double(get(handles.Potencia, 'String'));%2
assignin('base','potencia',potencia);

compVar=str2double(get(handles.compVar, 'String'));%3
assignin('base','compVar',compVar);

SegZ=str2double(get(handles.SegZ, 'String'));%4
passoZ=compVar/SegZ;
assignin('base','passoZ',passoZ);
assignin('base','SegZ', SegZ);

qteElem=str2double(get(handles.QteElem, 'String'));%5
assignin('base','qteElem',qteElem);

qteReg=str2double(get(handles.QteReg, 'String'));%6
assignin('base','qteReg',qteReg);

library=get(handles.listbox1, 'Value');
if library==1%PWRU
    PWR= 'LIB     0    1 2 3     219 220 221    9   50   0   ';
    last='1';
elseif library==2%PWRPUU
    PWR= 'LIB     0    1 2 3     207 208 209    9   50   0   ';
    last='2';
elseif library==3%PWRPUPU
    PWR= 'LIB     0    1 2 3     210 211 212    9   50   0   ';
    last='3';
elseif library==4%PWRDU3TH
    PWR= 'LIB     0    1 2 3     213 214 215    9   50   0   ';
    last='7';
elseif library==5%PWRPUTH
```

```
    PWR= 'LIB     0    1 2 3     216 217 218    9   50   0   ';
    last='8';
elseif library==6%PWRU50
    PWR= 'LIB     0    1 2 3     219 220 221    9   50   0   ';
    last='9';
elseif library==7%PWRDUD35
    PWR= 'LIB     0    1 2 3     222 223 224    9   50   0   ';
    last='10';
elseif library==8%PWRDUD33
    PWR= 'LIB     0    1 2 3     225 226 227    9   50   0   ';
    last='11';
elseif library==9%PWRUS
    PWR= 'LIB     0    1 2 3     604 605 606    9   50   0   ';
    last='38';
elseif library==10%PWRUE
    PWR= 'LIB     0    1 2 3     601 602 603    9   50   0   ';
    last='39';
else
    fprintf(fid, 'ERROR! library not found!');
end

NLIB = evalin('base', 'Th');

if NLIB == 1 %Como o valor é variável tive de usar essa estrutura
    NUC = '1       ';%U, PU
elseif NLIB == 2
    NUC = '2       ';%Th, U, PU
elseif NLIB == 3
    NUC = '3       ';%Th, U, PU +
elseif NLIB == 4
    NUC = '4       ';%Th, U, PU, Cm, Cf
end
LIB=[PWR,NUC,last];
assignin('base','LIB',LIB);
ORIG3N_2
close ORIG3N_1
```

Parte – 8:

```
% Lendo dados
nomeXLS = get(handles.nomeXLS, 'String');
[lixao, leDados] = xlsread(nomeXLS{1});

maxIterLeDados = size(leDados,1);
kReg = maxIterLeDados;

for iterLeDados=1:maxIterLeDados
    auxLeDados = '';
    for iterColunasLeDados=1:size(leDados,2)
        auxLeDados = [auxLeDados, ' ', leDados{iterColunasLeDados}];
    end
    compAtual{iterLeDados} = auxLeDados;
end

% salvando compAtual na base
assignin('base', 'compAtual', compAtual);
assignin('base', 'kReg', kReg);
```

```
regToken=get(handles.regId, 'String');
ko=evalin('base','ko');
ncell=evalin('base','ncell');

eval([sprintf(genvarname(ncell{ko})) '=regToken;']); %transforma as
palavras em variáveis.
assignin('base', ncell{ko}, eval(char(ncell(ko))))%transfere os
valores para serem usados em dados futuros.

MMP=str2double(get(handles.MMP, 'String'));

ko=ko+1;
assignin('base','ko',ko);

set(handles.text4, 'String', ko);

qteReg=evalin('base','qteReg');
if ko == qteReg
    set(handles.seguinte,'Visible','on');
end

if ko==2
    raiz = cell2mat(['Comp' get(handles.regId,'String')]);
else
    raiz = ['Comp' get(handles.regId,'String')];
end
compAtual=evalin('base','compAtual');
eval([raiz '=compAtual']);
assignin('base', raiz, eval(char(raiz)));
set(handles.regId, 'String','');
assignin('base','kReg', 1);
```

Parte – 9:

```
ElId=get(handles.ElId, 'String');
RegId=get(handles.regId, 'String');
Ffa=str2double(get(handles.Ffa, 'String'));
vazio=' ';
%_____________________________________________________________OK
SegZ=evalin('base','SegZ');
qteElem=evalin('base','qteElem');
passoZ=evalin('base','passoZ');
compVar=evalin('base','compVar');
potencia=evalin('base','potencia');
ks=evalin('base','ks');

%ler os dados do caminho:
FatorXLS = get(handles.FatorAxial, 'String');
[lixao, FatorAxial] = xlsread(FatorXLS);


for i=1:SegZ


    PotReg(i)=((Ffa/qteElem)*potencia)*num2double(FatorAxial{i});
```

```
end
assignin('base','PotReg',PotReg);

%_____________________________________________________________OK

ArqLab=[ get(handles.regId,'String') '_' get(handles.ElId, 'String')];
assignin('base','ArqLab',ArqLab);
LIB=evalin('base', 'LIB');

for i=1:SegZ
    if ks==1
        ArqName=cell2mat([ '.\INP\' ArqLab '_' num2str(i) '.INP']);
%OK
    else
        ArqName=[ '.\INP\' ArqLab '_' num2str(i) '.INP'];
    end
    fid=fopen(ArqName, 'w');
    fprintf(fid,'-1');
    fprintf(fid,'\r\n');
    fprintf(fid,'-1');
    fprintf(fid,'\r\n');
    fprintf(fid,'-1');
    fprintf(fid,'\r\n');
    fprintf(fid,'CUT -1');
    fprintf(fid,'\r\n');
    fprintf(fid,'LIP 0 0 0');
    fprintf(fid,'\r\n');
    fprintf(fid,LIB);
    fprintf(fid,'\r\n');
    fprintf(fid,'PHO 101 102 103 10');
    fprintf(fid,'\r\n');
    fprintf(fid,'INP -1 1 -1 -1 1 1');
    fprintf(fid,'\r\n');
    fprintf(fid,'MOV -1 1 0 1 0');
    fprintf(fid,'\r\n');
    fprintf(fid,'PCH 1 1 1');
    fprintf(fid,'\r\n');
    fprintf(fid,'BUP');
    fprintf(fid,'\r\n');
    kh=evalin('base', 'kh');
    for j=1:kh-1 %ok
        loadTkn = evalin('base', ['load', num2str(j)]); %ok
        PotTot=PotReg(i)*loadTkn;%ok
        diaTkn = evalin('base', ['dia', num2str(j)]);
        if j==1
            hLinha=['IRP' vazio num2str(diaTkn) vazio num2str(PotTot)
vazio '1 2 4 2' ];
            fprintf(fid, hLinha);
            fprintf(fid,'\r\n');
        else
            hLinha=['IRP' vazio num2str(diaTkn) vazio num2str(PotTot)
vazio num2str(j) vazio num2str(j+1) vazio '4 0'];
            fprintf(fid, hLinha);
            fprintf(fid,'\r\n');
        end
    end
    decFst=['DEC 0.5' vazio num2str(kh) vazio '1 3 2'];
    fprintf(fid, decFst);
    fprintf(fid,'\r\n');
    fprintf(fid, 'DEC 1.0 1 2 3 0');
    fprintf(fid,'\r\n');
```

```
        fprintf(fid, 'DEC 1.5 2 3 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 2.0 3 4 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 2.5 4 5 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 3.0 5 6 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 3.5 6 7 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 4.0 7 8 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 4.5 8 9 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 5.0 9 10 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 5.5 10 11 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid, 'DEC 6.0 11 12 3 0');
        fprintf(fid,'\r\n');
        fprintf(fid,'BUP');
        fprintf(fid,'\r\n');
        fprintf(fid,'OPTL 27*7');
        fprintf(fid,'\r\n');
        fprintf(fid,'OPTF 27*7');
        fprintf(fid,'\r\n');
        fprintf(fid,'OPTA 27*7');
        fprintf(fid,'\r\n');
        fprintf(fid,'OUT 12 1 -1 0');
        fprintf(fid,'\r\n');
        fprintf(fid,'END');
        fprintf(fid,'\r\n');
        if ks==1
            cTkn=cell2mat(['Comp' get(handles.regId,'String')]);
        else
            cTkn=['Comp' get(handles.regId,'String')];
        end
        cLinha=evalin('base', cTkn);
        med=size(cLinha,2);
        for j=1:med
            aux=cLinha{j}
            fprintf(fid,aux);
            fprintf(fid,'\r\n');
        end
        fprintf(fid,'0');
        fprintf(fid,'\r\n');
        fclose(fid);
    end
    set(handles.ElId, 'String','');
    set(handles.regId, 'String','');
    set(handles.Ffa, 'String','');

    qteReg=evalin('base','qteReg');
    if ks==qteReg
        set(handles.pushbutton3, 'Visible', 'on');
    end
    ks=ks+1;
    set(handles.text4, 'String', num2str(ks));
    assignin('base', 'ks', ks);
```

Parte – 10:

```
function varargout = Plotter(varargin)
% PLOTTER Application M-file for Plotter.fig
%   PLOTTER, by itself, creates a new PLOTTER or raises the existing
%   singleton*.
%
%   H = PLOTTER returns the handle to a new PLOTTER or the handle to
%   the existing singleton*.
%
%   PLOTTER('CALLBACK',hObject,eventData,handles,...) calls the local
%   function named CALLBACK in PLOTTER.M with the given input
arguments.
%
%   PLOTTER('Property','Value',...) creates a new PLOTTER or raises
the
%   existing singleton*.  Starting from the left, property value pairs
are
%   applied to the GUI before lb_OpeningFunction gets called.  An
%   unrecognized property name or invalid value makes property
application
%   stop.  All inputs are passed to Plotter_OpeningFcn via varargin.
%
%   *See GUI Options - GUI allows only one instance to run
(singleton).
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Plotter

% Copyright 2000-2006 The MathWorks, Inc.

% Last Modified by GUIDE v2.5 12-Jul-2011 17:41:44

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',          mfilename, ...
                   'gui_Singleton',     gui_Singleton, ...
                   'gui_OpeningFcn',    @Plotter_OpeningFcn, ...
                   'gui_OutputFcn',     @Plotter_OutputFcn, ...
                   'gui_LayoutFcn',     [], ...
                   'gui_Callback',      []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Plotter is made visible.
function Plotter_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% varargin   command line arguments to Plotter (see VARARGIN)

% Choose default command line output for Plotter
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% Populate the listbox
update_listbox(handles)
set(handles.listbox1,'Value',[])

% UIWAIT makes Plotter wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Plotter_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


function update_button_Callback(hObject, eventdata, handles, varargin)
% hObject    handle to update_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

update_listbox(handles)

function update_listbox(handles)
% hObject    handle to update (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns listbox1 contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from
listbox1

% Updates the listbox to match the current workspace
vars = evalin('base','who');
set(handles.listbox1,'String',vars)

function [var1,var2] = get_var_names(handles)
% Returns the names of the two variables to plot
list_entries = get(handles.listbox1,'String');
index_selected = get(handles.listbox1,'Value');
if length(index_selected) ~= 2
    errordlg('You must select two variables','Incorrect
Selection','modal')
else
    var1 = list_entries{index_selected(1)};
    var2 = list_entries{index_selected(2)};
end
```

```
function plot_button_Callback(hObject, eventdata, handles, varargin)
% hObject    handle to plot_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

[x,y] = get_var_names(handles);
figure(gcf)
try
    evalin('base',['plot(',x,',',y,')'])
catch ex
    errordlg(...
      ex.getReport('basic'),'Error generating linear plot','modal')
end


function semilogx_button_Callback(hObject, eventdata, handles,
varargin)
% hObject    handle to semilogx_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

[x,y] = get_var_names(handles);
figure(gcf)
try
    evalin('base',['semilogx(',x,',',y,')'])
catch ex
    errordlg(...
      ex.getReport('basic'),'Error generating semilogx plot','modal')
end


function semilogy_button_Callback(hObject, eventdata, handles,
varargin)
% hObject    handle to semilogy_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

[x,y] = get_var_names(handles);
figure(gcf)
try
    evalin('base',['semilogy(',x,',',y,')'])
catch ex
    errordlg(...
      ex.getReport('basic'),'Error generating semilogy plot','modal')
end


% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background, change
%       'usewhitebg' to 0 to use default.  See ISPC and COMPUTER.
usewhitebg = ispc;
if usewhitebg
```

```
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor')
);
end
```

Parte – 11:

```
regToken=get(handles.regId, 'String');
ko=evalin('base','ko');
ncell=evalin('base','ncell');

eval([sprintf(genvarname(ncell{ko})) '=regToken;']); %transforma as
palavras em variáveis.
assignin('base', ncell{ko}, eval(char(ncell(ko))))%transfere os
valores para serem usados em dados futuros.

MMP=str2double(get(handles.MMP, 'String'));

ko=ko+1;
assignin('base','ko',ko);

set(handles.text4, 'String', ko);

qteReg=evalin('base','qteReg');
if ko == qteReg
    set(handles.seguinte,'Visible','on');
end

if ko==2
    raiz = cell2mat(['Comp' get(handles.regId,'String')]);
else
    raiz = ['Comp' get(handles.regId,'String')];
end
compAtual=evalin('base','compAtual');
eval([raiz '=compAtual']);
assignin('base', raiz, eval(char(raiz)));
set(handles.regId, 'String','');
assignin('base','kReg', 1);
```

Parte – 12:

```
gBox=num2str(get(handles.listbox1, 'value'));%pega o valor da caixa
vazio=' ';%cria um espaço em branco para as strings
ZIso=str2double(get(handles.ZIso,'String'));
AIso=str2double(get(handles.AIso,'String'));
QIso=str2double(get(handles.QIso,'String'));

NUCLID=num2str(10000*ZIso+10*AIso);
QteNuc=num2str(QIso*str2double(get(handles.MMP, 'String')));

CompNuc=[gBox vazio NUCLID vazio QteNuc];%gera a string com a
composição para o ORIGEN

%_____________________________________________________________________

ZIso2=str2double(get(handles.ZIso2,'String'));
```

```
AIso2=str2double(get(handles.AIso2,'String'));
QIso2=str2double(get(handles.QIso2,'String'));

T2=isnan(ZIso2);

if T2==0

NUCLID2=num2str(10000*ZIso2+10*AIso2);
QteNuc2=num2str(QIso2*str2double(get(handles.MMP, 'String')));

CompNuc=[CompNuc vazio NUCLID2 vazio QteNuc2];

end
%_____________________________________________________________________

ZIso3=str2double(get(handles.ZIso3,'String'));
AIso3=str2double(get(handles.AIso3,'String'));
QIso3=str2double(get(handles.QIso3,'String'));

T3=isnan(ZIso3);

if T3==0

NUCLID3=num2str(10000*ZIso3+10*AIso3);
QteNuc3=num2str(QIso3*str2double(get(handles.MMP, 'String')));

CompNuc=[CompNuc vazio NUCLID3 vazio QteNuc3];

end

%_____________________________________________________________________

ZIso4=str2double(get(handles.ZIso4,'String'));
AIso4=str2double(get(handles.AIso4,'String'));
QIso4=str2double(get(handles.QIso4,'String'));

T4=isnan(ZIso4);

if T4==0

NUCLID4=num2str(10000*ZIso4+10*AIso4);
QteNuc4=num2str(QIso4*str2double(get(handles.MMP, 'String')));

CompNuc=[CompNuc vazio NUCLID4 vazio QteNuc4];

end

%_____________________________________________________________________
kReg = evalin('base','kReg');
if kReg==1
    compAtual{kReg}=CompNuc;
    assignin('base','compAtual',compAtual);
else
    compAtual=evalin('base', 'compAtual');
    compAtual{kReg}=CompNuc;
    assignin('base','compAtual',compAtual);
end
kReg=kReg+1;
```

```
assignin('base','kReg',kReg);
%_____________________________________________________________________
set(handles.ZIso, 'String','');
set(handles.ZIso2, 'String','');
set(handles.ZIso3, 'String','');
set(handles.ZIso4, 'String','');
%_____________________________________________________________________
set(handles.AIso, 'String','');
set(handles.AIso2, 'String','');
set(handles.AIso3, 'String','');
set(handles.AIso4, 'String','');
%_____________________________________________________________________
set(handles.QIso, 'String','');
set(handles.QIso2, 'String','');
set(handles.QIso3, 'String','');
set(handles.QIso4, 'String','');
```

Parte – 13 - AchaAD:

```
%Evita que os valores de FP e AD sejam lidos

produtofissao='FISSION PRODUCTS';
activationproducts='ACTIVATION PRODUCTS';
achaPF=strfind(tline, produtofissao);
achaAP=strfind(tline, activationproducts);
if ~isempty (achaAP)
    tline = fgetl(arqOrigem);
    tline = fgetl(arqOrigem);
    tline = fgetl(arqOrigem);
    tline = fgetl(arqOrigem);




elseif ~isempty (achaPF)
    tline = fgetl(arqOrigem);
    tline = fgetl(arqOrigem);
    tline = fgetl(arqOrigem);
    tline = fgetl(arqOrigem);



end
```

Parte – 14:

```
clear
clc
close all
tic %Qual o tempo que está sendo levado para processar daqui...

arqOrigem = fopen('TAPE6.OUT'); %Abre o arquivo de saída do ORIGEN
```

```
strInicio = '3 SUMMARY TABLE:  CONCENTRATIONS, GRAM-ATOMS'; %Busca a
linha que determina o início de uma tabela
strFim =    'OUTPUT UNIT ='; %Busca a linha que determina o fim de uma
tabela
strTabela = '0                                          CUMULATIVE TABLE
TOTALS';

n = 0; %Início do contador
salvar = 0; %Indicador para o caso do arquivo salvar ou não


disp(' ') % Saltar linha
disp('------> iniciando processamento, tire as crianças da
sala')%Marcar início do processo



tline = fgetl(arqOrigem);%Armazenar a próxima linha em tline

while ischar(tline)% Rotina ativa enquanto o arquivo não for
inteiramente lido

    achaAD
    procuraInicio = strfind(tline,strInicio);%Busca pelo início pré
definido
    if ~isempty(procuraInicio)%verifica se o início foi encontrado
        tline = fgetl(arqOrigem);

        tline = fgetl(arqOrigem);

        tline = fgetl(arqOrigem);


        salvar=1; %Indicador para salvar
    end
    procuraFim = strfind(tline,strFim);%Busca pelo fim definido
    if ~isempty(procuraFim)%Verifica se o indicador de fim foi
encontrado

        salvar = 0;%Indicador para não salvar
    end

    procuraTabela = strfind (tline, strTabela);
    if ~isempty (procuraTabela)
        tline = fgetl(arqOrigem);

        tline = fgetl(arqOrigem);
    end

    if salvar%Define o que acontecerá caso seja para salvar
        n=n+1;%Indicador para criar as células
        A(n) = cellstr(tline);%Armazena os strings lidos dentro de uma
celula de n elementos
        ConvertendoStrings%Chama o arquivo que converterás as strings
em números
        b{n} = str2num(number);
        c(n) = cellstr(nome);
```

```
    end


    tline = fgetl(arqOrigem);%Ler próxima linha

end
c=c';
b=cell2mat(b');
a = regexprep(c, ' ', '', 'ignorecase');
a = regexprep(a, '+', '_', 'ignorecase');
a = regexprep(a, '0TOTAL', 'TOTAL', 'ignorecase');

for i=1:n
    eval([sprintf(genvarname(a{i})) '=b(i,:);']);
    assignin('base', a{i}, eval(char(a(i))))
end

ATime=[ 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5 5.5 6];
assignin('base', 'ATime', ATime)



fclose(arqOrigem); %Fechar arquivo
disp(' ')
disp('-----> Processamento terminado')%Indicar que o programa foi lido
até o fim.
toc%...até aqui
```