UTILIZAÇÃO E VERIFICAÇÃO DO CÓDIGO GEANT4 EM APLICAÇÕES MÉDICAS

Estudo de Caso para Elétrons

SANDRO ROGER BOSCHETTI

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA

UTILIZAÇÃO E VERIFICAÇÃO DO CÓDIGO GEANT4 EM APLICAÇÕES MÉDICAS

Estudo de Caso para Elétrons

SANDRO ROGER BOSCHETTI

Dissertação apresentada ao Programa de Pós-graduação em Ciências e Técnicas Nucleares da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito parcial para obtenção do título de Mestre em Ciências e Técnicas Nucleares
Área de Concentração: Ciências das Radiações

Orientadora: Dr.ª Antonella Lombardi Costa
Coorientador: Dr. Tarcísio Passos Ribeiro de Campos

BELO HORIZONTE
2013

Boschetti, Sandro Roger
B742u Utilização e verificação do código GEANT4 em aplicações médicas [manuscrito]: estudo de caso para eletróns / Sandro Roger Boschetti. — 2013.
89 f., enc.: il.

Orientadora : Antonella Lombardi Costa.
Coorientador:Tarcísio Passos Ribeiro de Campos.

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

Inclui apêndices.

.

Inclui bibliografia.

1. Engenharia nuclear - Teses. 2. Física médica – Teses. 3. Radiação ionizante - Teses. I. Costa, Antonella Lombardi. II. Campos, Tarcísio Passos Ribeiro de. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU:621.039(043)

Resumo

O aumento cada vez mais acelerado da utilização de radiação ionizante nos vários setores da medicina é uma realidade. Neste sentido, vários códigos inspirados no Método Monte Carlo têm sido desenvolvidos e adaptados para atender à crescente demanda no uso da radiação ionizante em aplicações médicas. Entre os códigos mais utilizados para transporte de radiação atualmente podemos incluir o ***GE****ometry* ***AN****d* ***T****racking* ***4*** (GEANT4) disponibilizado pelo *Conseil Européen pour la Recherche Nucléaire* (CERN). O GEANT4 é um conjunto de ferramentas computacionais, chamado de *toolkit*. O código utiliza programação orientada a objeto em `C++` e atualmente é utilizado em várias áreas com uma ampla faixa de energia. Diversos processos físicos e partículas têm sido incorporados no toolkit do GEANT4 permitindo seu uso em física médica. Este trabalho apresenta uma metodologia para instalação e utilização do código GEANT4 bem como a verificação do mesmo em aplicações com elétrons com o objetivo de aumentar a difusão e aperfeiçoamento do código entre um número cada vez maior de usuários.

Abstract

The increase ever accelerating of the use of ionizing radiation in several sectors of medicine is a reality. In this sense, codes inspired in the Monte Carlo method have been developed and adapted to meet the growing demand of the ionizing radiation application in medical utilizations. Among the most frequently used codes for radiation transport is included the ***GE****ometry* ***AN****d* ***T****racking* ***4*** (GEANT4) provided by *Conseil Européen pour la Recherche Nucléaire* (CERN). The GEANT4 is a set of software tools named toolkit. The code uses object oriented programming in `C++` and it is currently used in several areas with wide range of energy. Several physical processes and particles have been incorporated into the GEANT4 toolkit allowing its use in medical physics. This work presents a methodology for installing and using the GEANT4 code as well as to perform its verification for some applications with electons in order to stimulate its use, improve the code and increase its number of users.

Agradecimentos

À minha esposa, pela compreensão e amor.

Aos meus pais por tudo que me ensinaram e pelo amor.

À minha orientadora Antonella, por sua extraordinária orientação e por toda ajuda.

Ao meu coorientador Tarcísio pelas orientações valiosas.

Aos membros da banca, um agradecimento especial pela valiosa contribuição.

Aos meus amigos José Maria, Carla Flávia e Ely Amorim.

Aos professores e colaboradores do DEN.

À CAPES pelo suporte financeiro.

Dedico as minhas filhas e a minha esposa, grandes amores da minha vida.

Sumário

Siglas

R_0 *Alcance máximo dos elétrons*. 43, 44

CERN *Conseil Européen pour la Recherche Nucléaire*. iii, iv, 13, 14, 16, 23, 26–28, 59

CLHEP *Class Library for High Energy Physics*. 40, 78

CSDA *Continuous Slowing Down Approximation Range*. 34, 35, 43–45

EGS *Electron Gamma Shower*. 14

EGS4 *Electron Gamma Shower 4*. 32

EGSnrc *Electron Gamma Shower do NRC*. 32, 34, 36, 49

ENIAC *Electronic Numerical Integrator and Computer*. 17

FORTRAN *Formula Translating System*. 23

gcc *compilador da linguagem `C` com sua versão `C++ g++`*. 39, 40, 60, 68

GEANT3 ***GE****ometry* ***AN****d* ***T****racking* ***3***. 23

GEANT4 ***GE****ometry* ***AN****d* ***T****racking* ***4***. iii, iv, 14–16, 23–28, 30, 32, 33, 35, 38–42, 48, 49, 56–60, 67, 68, 72, 75, 76

GNA *Gerador de Número Aleatório*. 20, 21

HEP *High Energy Physics*. 13

ICRU *International Commission on Radiation Units & Measurements*. 35

IDE *Integrated Development Environment*. 39

KEK *High Energy Accelerator Research Organization*. 23

LCRNG *Linear Congruential Random Number Generator*. 20–22

MCNP *Monte Carlo N-Particle*. 14, 32, 34, 36, 46, 49, 51, 57

MCRNG *Multiplicative Congruential Random Number Generator*. 21, 22

OS *Operating System*. 38

PCTN *Programa de Pós-graduação em Ciências e Técnicas Nucleares*. 15

PENELOPE *Penetration and Energy Loss of Positrons and Electrons*. 14

POO *Programação Orientada a Objetos*. 31

RNG *Random Number Generator*. 20

SLC *Scientific Linux CERN*. 59, 60

SO *Sistema Operacional*. 38, 39

SSD *Source to Surface Distance*. 35

Glossário

Max Step Length comprimento máximo do passo. Equivale ao Δx no MCNP. Também chamado de Max Step Limit, é um dos parâmetros que o usuário pode definir no GEANT4. É o tamanho máximo permitido para o passo. 34, 46

Max Step Limit comprimento máximo do passo. Equivale ao Δx no MCNP. Também chamado de Max Step Length. 46, 49

`C++` Linguagem orientada ao objeto oriunda da linguagem `C`. iii, iv, 16, 23–25, 38, 39, 57, 58, 60, 68

ATLAS é um experimento na área de física de partículas no LHC do CERN. Também dá nome ao detector usado no experimento. 26–28

MacOS X sistema operacional da empresa Apple Inc.. 38, 39, 48, 60

Scientific Linux CERN 6.3 Sistema operacional do CERN baseando em Linux, versão 6.3. 38, 39, 48, 56, 59

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A simulação computacional, e em especial o método Monte Carlo, mostra-se um método muito interessante ao estudo de problemas nos quais o número de partículas não é tão pequeno em que a mecânica clássica seja adequada e nem tão grande em que a mecânica estatística seja a opção óbvia. Para esses tipos de problemas, métodos numéricos como o Monte Carlo geralmente se mostram a melhor opção.

O método Monte Carlo encontra aplicação nas mais variadas áreas do conhecimento humano como, por exemplo, ciências, engenharias, economia. É utilizado na simulação desde fenômenos físicos mais simples em problemas acadêmicos de física estatística, passando pela modelagem de problemas em economia e engenharia até simulações de grandes proporções como os relacionados aos experimentos realizados no *Conseil Européen pour la Recherche Nucléaire* (CERN) ou Centro Europeu para a Pesquisa Nuclear [Agostinelli et al., 2003]. Nesses experimentos, emprega-se o método Monte Carlo para a simulação de modelos de imensos detectores de radiação, em especial os de alta energia ou *High Energy Physics* (HEP).

Dentre as aplicações do método Monte Carlo, uma das mais nobres é a da área médica, mais especificamente da física médica. Nessas áreas o interesse no método Monte Carlo é em utlilizá-lo como ferramenta para a simulação do transporte de radiação. Com isso, é possível usar essa ferramenta em atividades como pesquisa e desenvolvimento de tecnologias em equipamentos médicos como, por exemplo, novas geometrias e detectores em radiodiagnóstico. Outra grande aplicação em física médica é na determinação de

doses de radiação depositadas em procedimentos de radioterapia.

Segundo [Yoriyaz, 2009], o documento mais antigo que se tem notícia a respeito do método Monte Carlo é o artigo [Metropolis e Ulam, 1949]. Desde então, no método em sí não há grandes mudanças, mas desenvolvimentos tecnológicos como computadores cada vez mais rápidos e novas linguagens de programação tem feito essa técnica ter uma aplicabilidade cada vez maior.

Vários códigos inspirados no Método Monte Carlo têm sido desenvolvidos e adaptados para análise e simulação do comportamento da radiação ionizante em aplicações médicas. Entre os códigos mais utilizados para transporte de radiação atualmente podemos incluir o *Electron Gamma Shower* (EGS), o *Monte Carlo N-Particle* (MCNP), o *Penetration and Energy Loss of Positrons and Electrons* (PENELOPE) e o ***GE****ometry* ***AN****d* ***T****racking* ***4*** (GEANT4), esse, disponibilizado pelo CERN.

Sendo o GEANT4 um conjunto de ferramentas de distribuição livre, aberto e relativamente novo, desenvolvido por pesquisadores de vários países da Europa, Japão e América do Norte, é importante a realização de pesquisas e testes desse código para validação dos mais diversos modelos. Devido à modularidade do código, seus usuários podem instalar, usar e modificar os componentes de interesse de acordo com o problema a ser tratado. Dessa forma, com o intuito de introduzir a utilização do código GEANT4 de uma maneira mais geral no meio acadêmico e científico, este trabalho apresenta uma metodologia padrão para instalação, utilização e verificação desse programa. A verificação do programa, depois de instalado, foi realizada tomando como base um documento *benchmark* [Carrier et al., 2004] no qual várias geometrias são consideradas para simular fantomas reproduzindo situações similares às aplicações médicas com utilização de fontes de elétrons como será descrito nas próximas sessões.

Pretende-se então, com este trabalho, contribuir para aumentar o número de usuários que detêm conhecimento dessa ferramenta, fortalecendo assim a difusão e o aperfeiçoamento do código GEANT4. Para tanto, será apresentada uma metodologia de instalação e utilização do código de forma mais prática através de um tutorial passo à passo.

A apresentação da presente dissertação está organizada da seguinte forma:

No Capítulo 2 apresenta-se a Revisão Bibliográfica trazendo uma breve descrição sobre o Método de Monte Carlo e as principais características do código GEANT4. Há também uma descrição detalhada do *benchmark* utilizado com base para verificação do código. Vide seção 2.4.

No Capítulo 3 descreve-se, em detalhes, a metodologia de simulação dos casos selecionados do *benchmark*. São também feitas algumas considerações sobre o descarregamento, configuração, instalação de programas e sistemas operacionais utilizados nas simulações. O restante da metodologia relacionada à essa parte encontra-se nos apêndices A, B, C e D.

No Capítulo 4 apresentam-se os resultados relacionados aos casos descritos no Capítulo 3.

O Capítulo 5 apresenta as conclusões, as possíveis contribuições desse trabalho para o *Programa de Pós-graduação em Ciências e Técnicas Nucleares* (PCTN) e propõe atividades a serem desenvolvidas futuramente.

Capítulo 2

Revisão Bibliográfica

Vários códigos para análise do transporte de radiação inspirados no Método Monte Carlo têm sido desenvolvidos e adaptados para atender à crescente demanda no uso da radiação ionizante em aplicações médicas. Entre os códigos mais utilizados atualmente podemos incluir o GEANT4 disponibilizado pelo CERN. O GEANT4 é um conjunto de ferramentas computacionais, chamado de *toolkit* por [Agostinelli et al., 2003]. O código utiliza a linguagem `C++` e é atualmente utilizado em várias áreas com uma ampla faixa de energia. A seguir, será apresentada uma breve descrição sobre o Método de Monte Carlo e, especificamente, sobre o código GEANT4.

2.1 O Método Monte Carlo

No século XIX havia dois modos distintos de tratar problemas físicos. Os problemas relacionados a sistemas físicos de poucas partículas eram objetos da mecânica clássica cujo tratamento dado era por meio de equações diferenciais ordinárias. Por outro lado, os sistemas físicos com muitas partículas eram objeto de estudo da mecânica estatística que é parte da física que não se atem às partículas individuais, mas estuda o comportamento coletivo e suas propriedades. Para problemas físicos envolvendo somente algumas partículas, ambos métodos não são apropriados.

Um exemplo desse tipo de sistema, segundo [Metropolis, 1987] e [Metropolis e Ulam, 1949], é a cadeia de eventos que se sucede à entrada na atmosfera de uma partícula

portando grande quantidade de energia. A partícula pode produzir, num primeiro evento, outras partículas dos mais variados tipos que, por sua vez, podem produzir outras e assim sucessivamente. A cada interação há certo número de fenômenos físicos de natureza probabilística a serem considerados, tanto com relação a que partícula que é produzida quanto à direção de espalhamento. Do ponto de vista matemático, é um problema complicado e serve como exemplo de um sistema matemático chamado **Cadeia de Markov**, cuja solução passa pela multiplicação de um grande número de matrizes $n \times n$ onde n é bem grande.

Uma maneira de tratar esse problema é pela execução de um número finito de experimentos que não são realizados praticamente, mas sim de forma teórica. Sendo conhecidas as probabilidades envolvidas nos processos de interação das partículas com a matéria, então é possível realizar esses experimentos de forma a obter o comportamento médio que ocorre com as partículas e seus históricos. Esse tipo de experimento denomina-se **Método Monte Carlo**, cuja descrição matemática, conforme Neumann e Ulam [Neumann e Ulam, 1945] "é o estudo do fluxo que consiste de uma mistura de processos estocásticos e determinísticos".

O procedimento acima, para levar a resultados com algum significado, deve ser obtido pelo acúmulo de um número suficientemente grande de experimentos. Isso é impraticável a não ser pelo uso do computador.

2.1.1 Breve Histórico

Em 1946, enquanto o matemático Stanislaw Ulam jogava paciência, ele tentava calcular as probabilidades no jogo usando análise combinatória. Após empregar bastante tempo fazendo cálculos, percebeu que uma alternativa mais prática seria simplesmente realizar inúmeras jogadas (por exemplo, cem, ou mil) e contar quantas vezes cada resultado ocorria [Eckhard, 1987].

Nessa época surgiu o primeiro computador eletrônico, desenvolvido durante a II guerra mundial, o *Electronic Numerical Integrator and Computer* (ENIAC); antes dele eram utilizados dispositivos mecânicos para fazer cálculos. A versatilidade e rapidez do ENIAC impressionaram Ulam, que sugeriu o uso de métodos de amostragem estatística

para solucionar o problema da difusão de nêutrons em material sujeito a fissão nuclear. Nicholas Metropolis sugeriu o nome Monte Carlo para o método estatístico, inspirado em um tio de Ulam que sempre pegava dinheiro emprestado com parentes para ir até Monte Carlo jogar [Metropolis, 1987].

O método Monte Carlo implica que o comportamento de uma partícula não é determinístico. Dessa forma, as informações a respeito de uma partícula (energia, posição, direção e movimento) e as propriedades do meio na qual ela interage, não possibilitam determinar o futuro de tal partícula. Uma vez que a colisão ocorra, o resultado é um fenômeno estocástico. Por exemplo, quando há uma interação com nêutrons, existe a probabilidade que ele seja espalhado ou capturado. Então, se existirem meios para acompanhar uma partícula com parâmetros iniciais idênticos, a "história"para cada partícula será diferente. Entretanto, ao calcular a média dos resultados que se tem interesse, como por exemplo fluxo ou dose de radiação, baseado nos resultados das histórias de muitas partículas, este cálculo mostra que as médias concordam quantitativamente com os valores correspondentes medidos [Lamarsh e Baratta, 2001].

2.1.2 Fundamento Matemático

Um requisito fundamental para o cálculo de Monte Carlo é a distribuição das probabilidades associadas a possíveis eventos, bem como todas as variáveis envolvidas com cada evento. A decisão de um determinado evento ocorrer ou não, e de uma variável assumir um valor, está diretamente associada às distribuições de probabilidade. Considerando uma variável x e definindo $f(x)\,dx$ como a probabilidade de que x esteja entre x e $x + dx$, a função $f(x)$ é definida como a **função distribuição de probabilidade** (*pdf*) da variável x, normalizada para um. Uma **função de probabilidade acumulativa** (*cdf*), $F(x)$ é definida como [Cochran e Tsoulfanidis, 1999]:

$$F(x) = \int_{x_1}^{x} f(x)dx, \tag{2.1}$$

onde $F(x)$ é a probabilidade que o valor da variável x seja menor ou igual a x. De acordo com essa definição, $F(x)$ está delimitada entre 0 e 1.

Para selecionar valores adequados de x para o cálculo de Monte Carlo, a distribuição destes valores deve seguir a função $f(x)$. Isto é feito selecionando um valor RN (do acrônimo em inglês *Random Number*) e ajustando:

$$RN = F(x) \tag{2.2}$$

e invertendo-o para obter

$$x = F^{-1}(RN). \tag{2.3}$$

Pode ser demonstrado que se os valores de x forem selecionados conforme as Equações 2.2 e 2.3, baseado nos valores de RN uniformemente distribuídos entre 0 e 1, sua distribuição é a de $f(x)$. Computadores são capazes de gerar uma série de números aleatórios entre 0 e 1, distribuídos uniformemente nesta faixa, ou seja, com igual probabilidade de encontrar um número entre 0 e 1.

A geração de números randômicos é essencial para um cálculo baseado no método Monte Carlo e, sendo assim existem algoritmos específicos para gerá-los [Stacey, 2007].

Existem casos onde a *pdf* é uma função discreta. Então, se o número de casos é N, tem-se [Cochran e Tsoulfanidis, 1999]:

$$\sum_{i=1}^{N} f(x_i) = 1 \tag{2.4}$$

e a *cdf* é dada por:

$$F(x_i) = \sum_{j \leq i} f(x_j). \tag{2.5}$$

As seguintes características são comuns nos cálculos de Monte Carlo:

1. descrição do processo físico;
2. formulação do modelo de probabilidade;
3. definição da base estimando uma variável aleatória;

4. construção de distribuição de amostragem, usando números aleatórios; e
5. processamento de amostras e análise estatística de dados.

2.1.3 Geradores de Números Aleatórios

O chamado *Gerador de Número Aleatório* (GNA), em inglês *Random Number Generator* (RNG), ou na verdade pseudo-aleatórios, está no cerne de qualquer simulação Monte Carlo. É com eles que imitamos o comportamento estocástico dos fenômenos que estudamos por meios dessas simulações. Devido à sua importância, muito empenho tem sido feito no estudo desse assunto que é objeto de pesquisa intensa. Como consequência, as informações desse campo de pesquisa mudam continuamente.

Existe uma categoria de GNA considerada geradores de números aleatórios reais. Exemplo desses geradores são fontes externas de ruído de circuitos elétricos e o intervalo de tempo entre decaimentos sucessivos de uma substância radioativa. Para que se possa usar um GNA desse tipo, seria necessário ou ter um dispositivo eletrônico acoplado ao computador da simulação ou ter uma grande sequência de números aleatórios armazenados no computador. Ambas as situações são, em muitos casos, indesejáveis ou impraticáveis.

O estudo de geradores de números aleatórios e suas aplicações mostram que a medida que um código evolui, menos erros são encontrados nesses códigos. Outro ponto a ser considerado é que a sequência de números gerados por geradores de números pseudo-aleatórios se repete depois de um determinado número de números ter sido gerado. Segundo [Bielajew, 2001], esses dois fatos bastam para dizer que essas técnicas devem ser usadas com extrema cautela, ou erros na simulação devido a isso podem surgir.

Geradores de Números Aleatórios Linear Congruente (*LCRNG – Linear Congruential Random Number Generator*)

Em computadores cuja arquitetura suporta a aritmética de complemento de 2, pode-se empregar a equação 2.6 que descreve um *Linear Congruential Random Number Generator* (LCRNG) para gerar números pseudo-aleatórios:

$$X_{n+1} = \mathrm{mod}(a\,X_n + c, 2^{32}) \tag{2.6}$$

Onde a é o chamado número "mágico", c é um número "impar", X_{n+1} é o número aleatório no passo $n+1$ e X_n é o número aleatório no passo n e todos são números inteiros de 32 bits. Para o caso em que $c = 0$ então LCRNG torna-se o método *Multiplicative Congruential Random Number Generator* (MCRNG). X_0 é chamado de "**semente**" e pode ser qualquer número inteiro para o caso do LCRNG. É prática comum usar um número impar grande ou um grande número primo. Enquanto o LCRNG pode gerar uma sequência de números de até 4 bilhões, o MCRNG gera uma sequência de no máximo 1 bilhão de números.

Por convenção, usam-se números aleatórios entre 0 e 1. A conversão para os números gerados pela equação 2.6 pode ser feita pela seguinte equação:

$$r_n = \frac{1}{2} + \frac{X_n}{2^{32}} \tag{2.7}$$

Isso produz números no intervalo $0 \leqslant r_n < 1$ por conta da representação assimétrica desses números que vai de -2^{31} à $2^{31} - 1$.

Esse GNA gera cadeias aleatórias de 32 bits X_{n+1} por meio da representação aleatória do passo anterior X_n. Por meio das adições e multiplicações, os bits de mais alta ordem (à esquerda), são perdidos deixando os bits de mais baixa ordem da cadeia embaralhados de uma forma pseudo aleatória.

Geradores de Números Aleatórios de Sequência Longa (Long Sequence Random Number Generators)

Para algumas aplicações, a quantidade de números gerados pelos métodos acima pode não ser suficiente. Por isso, muitas vezes é necessário o emprego de outros métodos ou de outra técnica como por exemplo o **Gerador de Números Aleatórios de Sequência Longa**. Nesse caso o algoritmo é o mesmo, mas agora com um número inteiro representado em 64 bits. Números de 64 bits podem gerar uma cadeia de até 2^{64} números aleatórios, ou aproximadamente $1,84 \times 10^{19}$ para o caso do LCRNG e de até

2^{62} ou aproximadamente $4,61 \times 10^{18}$ para o MCRNG. Uma boa escolha para o número a nesse caso é 6364136223846793005.

O Algoritmo "*Subtract-with-borrow*"

Esse método foi desenvolvido por George Marsaglia e fora implementado usando ponto flutuante ao invés de números inteiros [Marsaglia et al., 1990] apud [Bielajew, 2001] e [Marsaglia e Zaman, 1991] apud [Bielajew, 2001]. Outra característica desse método é sua portabilidade para diferentes arquiteturas. O algoritmo desse método é equivalente ao LCRNG com um número inteiro bem grande, o que reduz significativamente as anomalias espectrais. Sua longa sequência de números e velocidade o faz muito atraente, no entanto, o fato de ainda ser pouco conhecido e estudado faz com que o LCRNG seja mais usado pois inspira maior segurança.

Segue abaixo a listagem de um código escrito pelo autor dessa dissertação como forma de exercício do algoritmo LCRNG.

```
#include <stdio.h>
#include <math.h>  // pow(x,y)
#include <time.h>  // para time() para a geracao da semente.

/*
 * INT_MAX = pow(2,31) para signed int em 32 bits = 2147483647
 * de - 2147483648 a 2147483647 = 4.294.967.295
 * O unsigned int de 32 bits vai de 0 a 4.294.967.295
 */
#include <limits.h>


int main(int argc, char **argv){

	unsigned int seed = (unsigned) time(NULL); // 987654321 ou (unsigned) time(NULL)
	unsigned int m = (unsigned)pow(2,32);      // pow(2,32) para 32 bits = UINT_MAX

/*
 * Good numbers for the "magic number" a are 663608941 and 69069.
 * The latter has been suggested by Donald Knuth as the "best" 32-bit multiplier
 */
	unsigned int a = 69069;

	unsigned int c = 13;
	unsigned int x0 = seed;
	unsigned int x1;

	double rand;

	// Declaracao fora do loop por sugestao do compilador
	unsigned int i;

	// Sao gerados no maximo m numeros aleatorios. O numero m+1 eh igual ao primeiro.
	for (i = 0; i < m; i++){
		x1 = (a * x0 + c) % m;
		rand = x1 / (double)m;
		printf("%10u = %12u => %1.5f\n", i+2, x1, rand);
		x0 = x1;
	}

	return 0;
}
```

2.2 O Código GEANT4

Como foi descrito anteriormente, diversos códigos inspirados no Método Monte Carlo têm sido desenvolvidos e adaptados para uso em aplicações médicas, entre os quais o GEANT4 disponibilizado pelo CERN [Agostinelli et al., 2003; Allison et al., 2006].

Vários trabalhos têm demonstrado a eficácia de aplicação do código GEANT4 e têm investigado vários aspectos do mesmo como, por exemplo, na simulação de espectros de energia produzidos por equipamento de Raio-X [Bonifácio et al., 2007], no estudo da dependência da energia depositada com parâmetros não físicos do algoritmo de transporte de elétrons e influência do número de eventos nos resultados simulados [Malthez, 2011], na caracterização de máquinas e fontes tais como aceleradores lineares [Carrier et al., 2004] e fontes de braquiterapia [Almansa et al., 2007], entre muitos outros.

Vários trabalhos têm sido publicados referentes ao processo de validação do código GEANT4. Grande parte deles encontra-se no próprio sítio do código, `http://geant4.web.cern.ch/geant4`, e podem ser livremente acessados através do link "*Results & Publications related to Geant4*".

2.2.1 Breve Histórico

O desenvolvimento do GEANT4 ocorreu independentemente através de dois grupos de estudo do CERN e do *High Energy Accelerator Research Organization* (KEK) em 1993. Ambos pesquisavam técnicas computacionais modernas para melhorar o já existente ***GE****ometry* ***AN****d* ***T****racking* ***3*** (GEANT3) o qual era programado na linguagem *Formula Translating System* (FORTRAN). Os estudos avançaram e culminaram na construção de um programa baseado na tecnologia de programação orientada a objetos. O projeto resultante incluiu os esforços de cerca de 100 cientistas e engenheiros de vários países que conseguiram adaptar a metodologia de programação orientada a objetos e decidiram pelo uso de uma linguagem de programação mais prática, o `C++` [Agostinelli et al., 2003].

A fase de *R&D* (Pesquisa e Desenvolvimento) foi completada em 1998 com a liberação

da primeira versão. Foi organizado o "*Geant4 Collaboration*" em 1999 para continuar o desenvolvimento e refinamento do conjunto de ferramentas para prover a manutenção e suporte ao usuário.

2.2.2 Características Gerais

Vários processos físicos estão implementados no código GEANT4 tais como o efeito Rayleigh, efeito fotoelétrico, efeito Compton, produção de pares, efeito Auger, ionização, bremsstrahlung, aniquilação de pósitrons, múltiplos espalhamentos, entre outros. Entre algumas vantagens desse código, destacam-se:

- Possibilidade de lidar com todos os tipos de partículas;
- É aplicável à geometrias bastante complexas;
- Está na linguagem `C++`.

O GEANT4 é um conjunto de ferramentas computacionais, chamado de *toolkit*. O código utiliza programação orientada a objeto em `C++` e atualmente é utilizado em várias áreas com uma ampla faixa de energia. Diversos processos físicos e partículas têm sido incorporados ao *toolkit* do GEANT4 permitindo seu uso em física médica. Esses processos estão bem descritos em "*Physics Reference Manual*" [CERN, 2012b].

Sendo então o código GEANT4 um conjunto de ferramentas de distribuição livre e relativamente novo, desenvolvido por pesquisadores de vários países, é importante a realização de pesquisas e testes desse código para validação dos mais diversos modelos. Devido à modularidade do código seus usuários podem instalar, usar e modificar os componentes de interesse de acordo com o problema a ser tratado.

É possível utilizar o GEANT4 em diversas aplicações como pode ser verificado através dos exemplos fornecidos no conjunto de ferramentas do mesmo [Allison et al., 2006]. Os exemplos incluem simulação de detectores de partículas, telescópios espaciais para raio-X e raio gama astronômicos, detectores para investigação de matéria escura, um experimento com raios cósmicos, entre outros. Duas simulações tratam situações críticas tais como uma blindagem e proteção contra nêutrons em missões espaciais. Outros

exemplos tratam de várias técnicas em radioterapia, incluindo braquiterapia e hadronterapia.

Apesar de o código GEANT4 já não ser exatamente uma novidade, pois já está sendo usado há um bom tempo, ainda não é um código amplamente difundido e utilizado, apesar de possuir grandes vantagens como ser um código fonte aberto e gratuito e ser baseado em uma linguagem orientada ao objeto, o que lhe confere grande flexibilidade e reuso, dentre outras vantagens. Justamente o item que confere a esse código várias vantagens é também uma desvantagem em sua adesão. O fato de esse código ser baseado em `C++`, uma linguagem de programação orientada ao objeto, faz com que seja naturalmente limitado o número de pessoas que podem utilizá-lo de forma imediata. Mesmo para os usuários mais familiarizados com programação, o processo de aprendizagem da linguagem `C++`, os conceitos de orientação a objetos, a instalação e configuração do ambiente de desenvolvimento e da utilização do GEANT4 propriamente pode não ser tão óbvio e demandar bastante tempo. O diagrama de blocos da figura 2.1 nos dá uma ideia dos módulos dessse conjunto de ferramentas.

Todo o conjunto de ferrametas GEANT4 é muito bem modularizado. Essa modularização pode ser visualizada de uma forma bastante abrangente por meio do diagrama de blocos da figura 2.1. Neste diagrama, cada bloco representa um conjunto de benfeitorias com responsabilidades muito bem definidas e suas interdependências. Por exemplo, os blocos *Material* e *Geometry* têm um conjunto de ferramentas (classes e métodos) que podem ser usados para definir a geometria dos detectores assim como sua constituição. Já *Particle* usamos para definir as partículas, suas energias, direção de emissão, etc da fonte. Os blocos *Run*, *Event*, *Traking*, *Digits+Hits*, *Processes* e *Track* são, de cima para baixo, pacotes de programas mais abrangentes ao mais granular. *Run* é responsável por uma dada simulação com um determinado número de eventos. *Event* é um pacote usado por *Run* e é reponsável por apenas um evento, ou seja, tudo aquilo que ocorre durante o processamento de um evento que vai desde a emissão da partícula primária da fonte até essa partícula perder toda energia ou sair do volume que circunscreve toda a geometria (*World Volume*) levando-se em conta todas as radiações segundárias. *Processes* é o pacote de ferramentas que utiliza-se para registrar fenômenos físicos às partículas. Diferentemente de outros códigos, o GEANT4

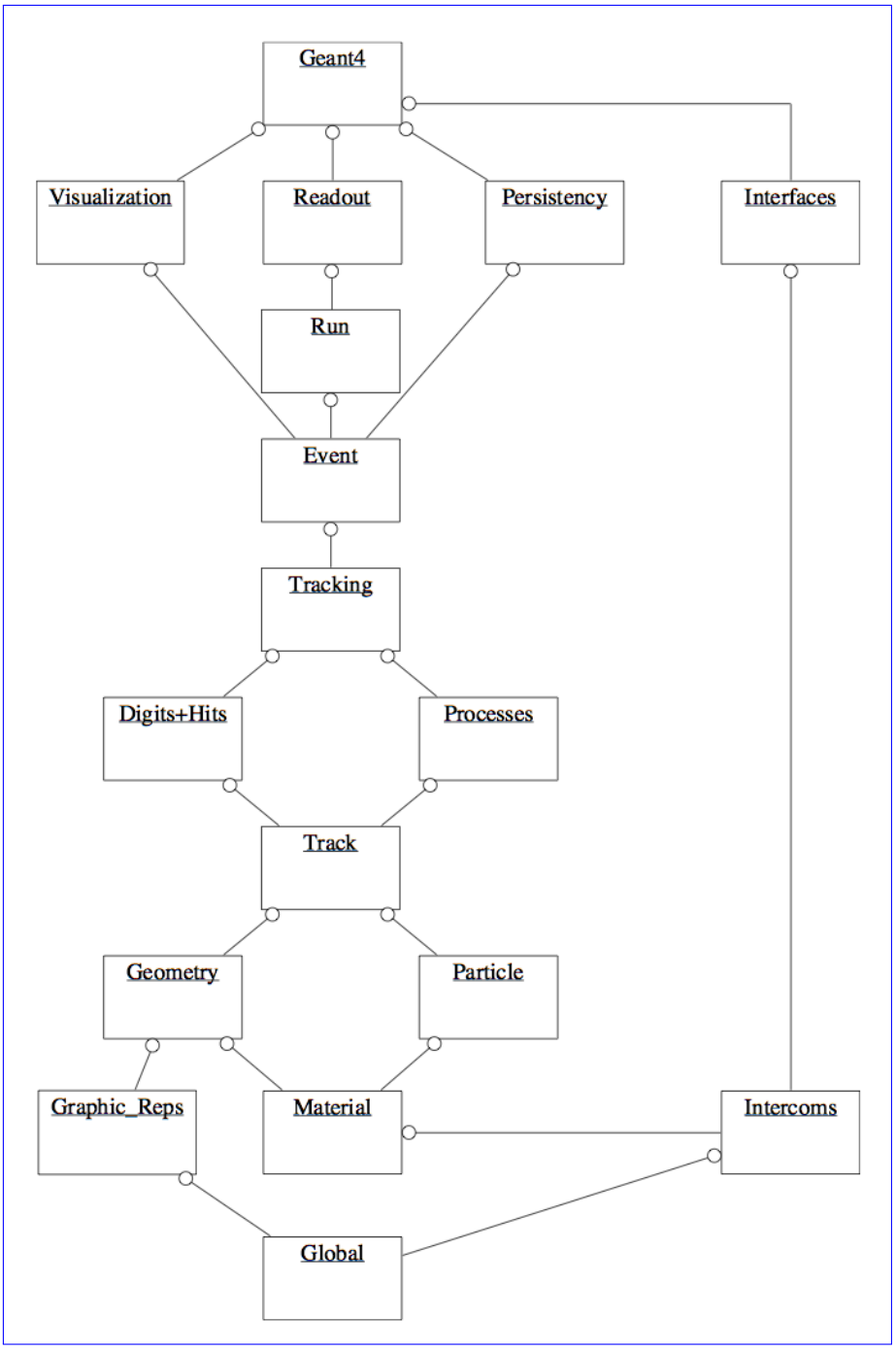

Figura 2.1: Diagrama de blocos do conjunto de ferrametas GEANT4 segundo [Agostinelli et al., 2003].

não disponibiliza um conjunto *default* de processos às partículas de nosso código. É necessário que o usuário registe na simulação as partículas de que precisa, não somente as primárias, mas também as que podem ser originadas nos processos de interação da radiação com a matéria. É preciso também que o usuário registre para cada partícula,

os fenômenos de interação da radiação com a matéria que cada partícula pode ter.

A figura 2.2 mostra a foto do complexo detector ATLAS do CERN e a figura 2.3 mostra a simulação desse detector no código GEANT4. Essas figuras são emblemáticas e mostram como o GEANT4 é capaz de modelar detectores extremamente complexos como esses.

Pretende-se então, com este trabalho, contribuir para aumentar o número de usuários que detêm conhecimento dessa ferramenta, fortalecendo assim a difusão e o aperfeiçoamento do código GEANT4. Para tanto, será apresentada uma metodologia de instalação e utilização do código de forma mais prática através de um tutorial passo a passo descrito nos apêndices A, B, C e D. Além disso, há o material em vídeo que está disponibilizado no sítio do Youtube na internet, cujo endereço é `http://www.youtube.com/playlist?list=PLKorLi2cUF3vOYrUBOa4_aWNDsy4c6TuS`.

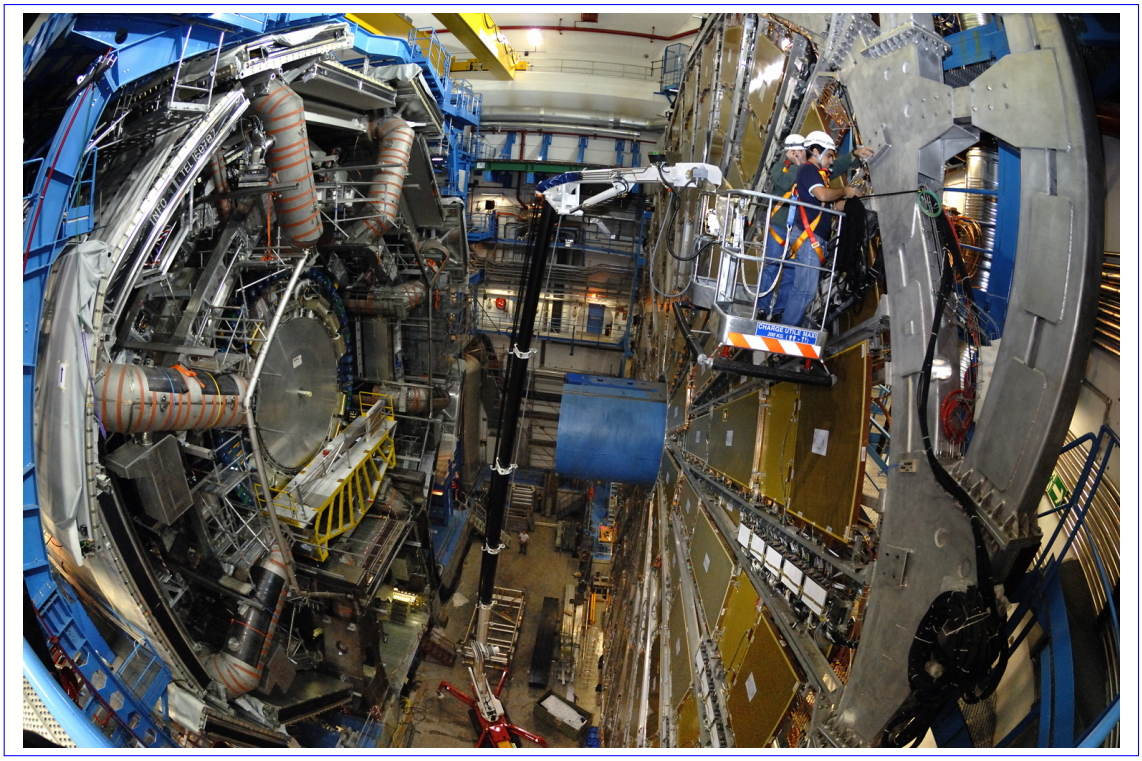

Figura 2.2: Foto do detector ATLAS no CERN.

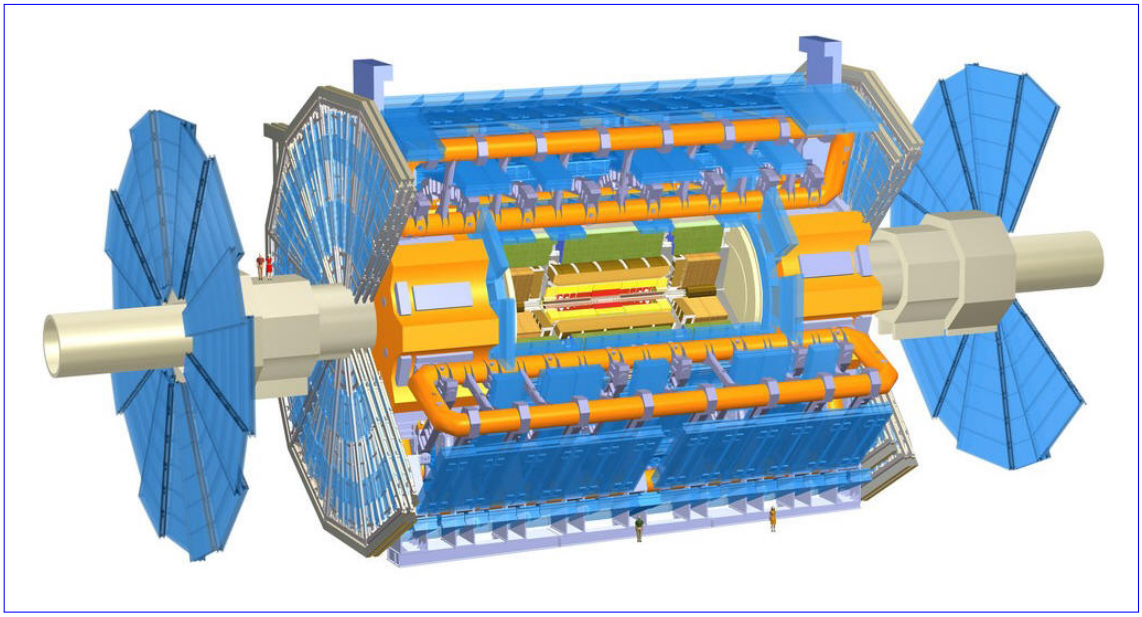

Figura 2.3: Detector ATLAS do CERN mostrado na figura 2.2 simulado usando o GEANT4.

2.2.3 Transporte de Partículas

O cálculo do livre caminho médio de uma partícula em um meio é realizado no GEANT4 usando a seção de choque da mesma em um processo físico e a densidade atômica [CERN, 2012b]. Em um material, o número de átomos por unidade de volume, N, é dado por:

$$N = \frac{N_0 \rho}{A} \tag{2.8}$$

Sendo, N_0 o número de Avogadro, ρ a densidade do meio e A é a massa em mol.

Em um material composto, o número de átomos por unidade de volume do i-ésimo elemento é dado por:

$$N_i = \frac{N_0 \rho w_i}{A_i} \tag{2.9}$$

Sendo, w_i a proporção de massa do i-ésimo elemento e A_i a massa em mol do i-ésimo elemento.

O livre caminho médio de um processo, λ, também chamado de comprimento de interação, pode ser dado em termos da seção de choque total:

$$\lambda(E) = \left(\sum_i \left[N_i \cdot \sigma(Z_i, E) \right] \right)^{-1} \tag{2.10}$$

Sendo, $\sigma(Z, E)$ a seção de choque microscópica total, dada em cm^2, por átomo do processo e $\sum_i \left[N_i \cdot \sigma(Z_i, E) \right]$ a seção de choque macroscópica total.

Os valores de seção de choque e de livre caminho médio podem ser tabelados no início da simulação para elevar o desempenho.

O livre caminho médio, λ, de uma partícula para um determinado processo depende do meio e por isso não pode ser usado diretamente para amostrar a probabilidade de interação em um detector heterogêneo.

O número de livres caminhos médios que uma partícula atravessa, dado por [CERN, 2012b] é:

$$n_\lambda = \int_{x_1}^{x_2} \frac{dx}{\lambda(x)} \tag{2.11}$$

Definindo n_r como uma variável randômica denotando o número de livres caminhos médios de um dado ponto ao ponto de interação, pode-se demonstrar que n_r tem a seguinte função de distribuição:

$$P(n_r < n_\lambda) = 1 - e^{-n_\lambda} \tag{2.12}$$

O número total de livres caminhos médios que a partícula atravessa antes de alcançar o ponto de interação, n_λ, é amostrado no início da trajetória como:

$$n_\lambda = -\log \eta \tag{2.13}$$

Sendo η um número randômico uniformemente distribuído no intervalo [0, 1]. O valor n_λ é atualizado a cada passo Δx de acordo com a fórmula:

$$n'_\lambda = n_\lambda - \frac{\Delta x}{\lambda(x)} \tag{2.14}$$

O ponto de intereação é então estimado como sendo:

$$s(x) = n_\lambda \, \lambda(x) \tag{2.15}$$

Até que s seja muito pequeno e desencadeie o processo específico.

2.2.4 Propagação de Erro na Trajetória

O pacote de propagação de erro no GEANT4 relaciona a trajetória da partícula com o erro da sua trajetória até que a mesma atinja o alvo (uma superfície, um volume, um dado comprimento da trajetória, etc.). Dessa forma, esse pacote calcula a trajetória média de uma partícula que a mesma poderia seguir [CERN, 2012a]. Os seguintes aspectos físicos são considerados:

- Não há múltiplos espalhamentos;
- Não há perda de energia por flutuações randômicas;
- Não há criação de trajetórias secundárias;
- Não há processo randômico.

O código considera o efeito do retroespalhamento (espalhamento na direção oposta à trajetória original) no ganho de energia. O processo `G4EnergyLossForExtrapolator` do GEANT4 calcula a perda média de energia da partícula na propagação normal e no retroespalhamento.

O usuário tem que fornecer o estado inicial da partícula no ponto de partida. Para fazer isso, ele deve criar um objeto usando o cartão `G4ErrorTrajState` identificando:

- Tipo de partícula;
- Posição;
- Momento;
- Matriz de erro da trajetória.

A trajetória da partícula é caracterizada por cindo variáveis independentes como função de um parâmetro (por ex., o comprimento da trajetória). Entre as cinco variáveis, uma está relacionada à curvatura da trajetória, duas estão ligadas à direção de propagação da partícula e outras duas à localização espacial da mesma. Há duas possíveis representações desses cinco parâmetros no pacote propagador de erro: como uma classe de

estado de trajetória livre (`G4ErrorTrajStateFree`) ou uma classe de estado de trajetória em uma superfície (`G4ErrorTrajStateonSurface`).

A matriz-erro 5x5 poderia também ser fornecida no momento da criação do estado de trajetória como um objeto `G4ErrorTrajErr`. Se não é dada, um objeto default será criado e preenchido com valores nulos. A matriz-erro é dada em unidades de energia e comprimento.

2.3 Programação Orientada a Objetos

O termo *Programação Orientada a Objetos* (POO) foi criado por Alan Kay, autor da linguagem de programação *Smalltalk*. O objetivo da criação da POO foi tentar aproximar o mundo real do mundo virtual. Dessa forma, na POO, o programador modela o mundo dos objetos os quais devem interagir entre si. Os objetos "conversam" uns com os outros através do envio de mensagens, e o papel principal do programador é especificar quais serão as mensagens que cada objeto pode receber, e também qual a ação que aquele objeto deve realizar ao receber aquela mensagem em específico [David, 2007].

Um dos grandes diferenciais da programação orientada a objetos em relação a outros paradigmas de programação está no conceito de herança, mecanismo através do qual definições existentes podem ser facilmente estendidas. Juntamente com a herança deve ser enfatizada a importância do polimorfismo, que permite selecionar funcionalidades que um programa irá utilizar de forma dinâmica, durante sua execução [Ricarte, 2001].

Uma classe é um gabarito para a definição de objetos. Através da definição de uma classe, descreve-se que propriedades o objeto deverá ter. O conjunto de atributos descreve as propriedades da classe. Na prática, a maior parte das linguagens de programação orientada a objetos oferecem um grupo de tipos primitivos que podem ser usados na descrição de atributos. O atributo pode ainda ter um valor *default* opcional, que especifica um valor inicial para o atributo.

O que torna a orientação a objetos única é o conceito de herança. Herança é um me-

canismo que permite que características comuns a diversas classes sejam fatoradas em uma classe base, a partir da qual outras classes podem ser especificadas. Cada classe derivada ou subclasse apresenta as características (estrutura e métodos) da classe base e pode-se acrescentar à elas particularidades que a classe base não possui [Ricarte, 2001].

Um dos conceitos mais complicados de se entender, e também um dos mais importantes, é o polimorfismo. Na orientação a objetos, isso significa que um mesmo tipo de objeto, sob certas condições, pode realizar ações diferentes ao receber uma mesma mensagem. Ou seja, apenas olhando o código fonte não sabemos exatamente qual será a ação tomada pelo sistema, sendo que o próprio sistema é quem decide qual método será executado, dependendo do contexto durante a execução do programa [Ricarte, 2001].

2.4 O Documento *Benchmark* Usado na Verificação do GEANT4

O artigo escolhido para o papel de *benchmark* foi o [Carrier et al., 2004], cujo título é "*Validation of GEANT4, an object-oriented Monte Carlo toolkit, for simulations in medical physics*", ou seja, "Validação do GEANT4, um conjunto de ferramentas orientado ao objeto, para simulações em física médica". Esse artigo foi publicado em 2004, sendo portanto razoavelmente recente. Foi publicado por quatro pesquisadores da Universidade de Laval em Quebec no Canadá. Três deles, J. F. Carrier, L. Archambault e L. Beaulieu, são do Departamento de Radio-oncologia e Centro de Pesquisa em Cancerologia, enquanto o quarto, R.Roy, é do Departamento de Física.

Esse documento foi escolhido para a verificação do código após instalado por ser razoavelmente recente e por possuir uma grande quantidade de combinações de partícula, energia, fantoma e geometria de interesse em física médica. Além disso, os autores fazem, nesse artigo, simulações usando o GEANT4 e comparam os resultados dessas simulações com simulações de vários outros códigos populares, além de as compararem com dados experimentais.

O *benchmark*, como seu próprio título já esclarece, tem por objetivo fazer a validação

do conjunto de ferramentas GEANT4 para aplicações em física médica. Para isso, os autores escolheram comparar os resultados de suas simulações com o GEANT4 com resultados da literatura de dois códigos bastante populares e de relevância em física médica. São eles: MCNP, *Electron Gamma Shower 4* (EGS4) e o *Electron Gamma Shower do NRC* (EGSnrc). Além da comparação com códigos já exaustivamente testados e validados, os autores fazem também comparações com dados experimentais baseado no artigo [Lockwood et al., 1987] do laboratório estadunidense Sandia. Para isso, os autores utilizam várias combinações de fantomas (homogêneos e multicamadas), fontes (pontual e feixe paralelo) e partículas (elétrons e fótons) à diferentes energias. Muitas dessas combinações têm relação direta com situações clínicas em física médica, embora algumas nem tanto. As geometrias simuladas no *benchmark* foram conforme a lista a seguir:

1. Feixe Largo Paralelo em um Meio Homogêneo (elétrons)
2. Feixe Monoenergético em um Fantoma Multicamadas (elétrons)
3. Dose Radial de uma Fonte Pontual (elétrons)
4. Feixe de um Acelerador Clínico Realístico (elétrons)
5. Feixe Monoenergético em um Fantoma Multicamadas (fótons)
6. Feixe de um Acelerador Clínico Realístico (fótons)

Nas 4 primeiras geometrias foram utilizados elétrons e nas duas últimas, fótons. Embora sejam listadas 6 geometrias, o número de casos é maior pois cada geometria pode apresentar uma ou mais situações diferentes como, por exemplo, a composição do fantoma, que para os primeiros casos foram utilizados o Berílio e o Molibdênio, e energias das partículas.

As simulações utilizaram a versão 5.2 do GEANT4 com a utilização do pacote de baixa energia. A versão do compilador gcc foi a 2.96. Para a avaliação das incertezas, os casos foram simulados de 5 a 10 vezes sendo que a incerteza relativa para cada voxel foi ajustada para ser igual ao desvio padrão.

Para as simulações dessa dissertação foram escolhidos os primeiros casos da primeira geometria considerada no *benchmark*, vide seção 2.4.1. Esses casos apresentavam, pelo

menos inicialmente, um grau de dificuldade menor. Assim poder-se-ia, após essa primeira parte, aumentar o grau de dificuldade em direção àqueles casos de maior relevância para a física médica clínica.

2.4.1 Feixe Largo Paralelo em um Meio Homogêneo (e^-)

Essa geometria é a primeira de um total de seis e é também a primeira das quatro nas quais a partícula emitida pela fonte é o elétron. A primeira parte consta de um fantoma semi-infinito irradiado por um feixe largo e paralelo incidindo perpendicularmente ao plano do fantoma. Dois fantomas são considerados, um de Berílio e outro de Molibdênio. O fantoma de Berílio é irradiado por elétrons de 0,521 MeV e 1,033 MeV, e o de Molibdênio é irradiado por elétrons de 0,5 MeV e 1,0 MeV, dando um total de 4 simulações diferentes. O voxel de registro das doses são, como o fantoma, lateralmente infinito, mas com espessura igual à 0,02 x R_0, sendo o parâmetro R_0 o *Continuous Slowing Down Approximation Range* (CSDA).

Os resultados dessas 4 simulações foram comparadas com resultados de simulações obtidos com os códigos MCNP e EGSnrc da literatura e com dados experimentais [Lockwood et al., 1987]. As comparações são feitas por meio de gráficos nos quais os resultados são plotados juntos e em que as abcissas são $^{z}/_{R_0}$ e as ordenadas são $^{MeV}/_{(g/cm^2)}$.

Para cada um dos quatro casos são simulados 1.000.000 elétrons primários. O limiar para a produção de gamas foi ajustado para 10 keV para elétrons e o parâmetro *Max Step Length* (Comprimento máximo do passo. Equivale ao Δx no MCNP) foi ajustado para 1% do CSDA inicial.

Na segunda parte, o fantoma é uma camada de água também semi-infinita irradiada por feixes largos e paralelos de elétrons monoenergéticos. São consideradas três diferentes energias, 100 keV, 1 MeV e 10 MeV. Como nos casos anteriores, os voxels são camadas lateralmente infinitas de espessura igual à 0,02 x R_0.

Para todos os casos a distribuição de doses são dadas pela grandeza adimencional B dada por [Chibani e Li, 2002] como segue:

$$B(E, z/R_0) = \frac{\Delta E}{E}\frac{R_0}{\Delta z} \tag{2.16}$$

Onde ΔE é a energia média depositada por elétron emitido da fonte com energia inicial E e R_0 correspondente à camada de espessura Δz na profundidade z.

A simulações com o GEANT4 tiveram o limiar de produção para gamas ajustados em 1 keV para gamas e 10 keV para elétrons. Para essas simulações o "comprimento máximo do passo" também foi ajustado para 1% do CSDA inicial e o número de elétrons iniciais foi de 1.000.000 para cada simulação.

2.4.2 Feixe Monoenergético em um Fantoma Multicamadas (e^-)

Nessa geometria o fantoma é multicamadas e é irradiado por elétrons de 20 MeV. A largura do fantoma é de 30,5 x 39,5 x 30 cm. São quatro camadas sendo a primeira de 0 à 2 cm de água, a segunda de 2 a 3 cm de alumínio, a terceira de 3 à 6 cm de pulmão (*International Commission on Radiation Units & Measurements* (ICRU), ρ = 0,26 g/cm^2) e a quarta de 6 à 30 cm de água. O feixe de elétrons é advindo de uma fonte pontual a 10 cm da superfície do fantoma e irradia esse perpendicularmente com um campo de 1,5 cm x 1,5 cm. Os voxels de registro da dose são de 5mm x 5mm de largura com 2mm de espessura e são centralizados no eixo central do fantoma.

2.4.3 Dose Radial de uma Fonte Pontual (e^-)

Para essa geometria a fonte é pontual e localizada no centro de um fantoma cúbico de água. As fontes são pontuais sendo uma de ^{90}Sr/^{90}Y sem encapsulamento e a outra é também de ^{90}Sr/^{90}Y com encapsulamento. Os voxels são camadas esféricas concêntricas de 0,1mm de espessura centralizadas na fonte.

2.4.4 Feixe de um Acelerador Clínico Realístico (e^-)

Nessa configuração, o acelerador linear de 6 MeV Varian Clinac 2100C irradia um fantoma de água. Os voxels são discos de 2 mm de espessura e 4 cm de raio centrados no meio do campo de irraiação de 10 x 10 cm^2. A distância da fonte à superfície, *Source to Surface Distance* (SSD), é de 100 cm.

A seguir, os casos que são simulados no *benchmark* usando-se fontes de fótons.

É importante lembrar que, nesse trabalho de dissertação, foram simuladas somente as situações descritas na primeira parte da subseção 2.4.1.

2.4.5 Feixe Monoenergético em um Fantoma Multicamadas (fótons)

Nesse caso, a fonte é pontual monoenergética de 10 e 20 MeV de fótons irradiando um fantoma multicamadas. A distância da fonte à superfície é de 100 cm e o campo irradiado na superfície é de 5 x 5 cm^2. O fantoma possui 5 camadas sendo a primeira (superior) de 6 cm de água, seguida por 3,5 de ar e, após, 1 cm de aço, outra de 3,5 cm de ar e então 6 cm de água. Os voxels são cilindros de 0,5 cm de espessura com 1 cm de raio ao longo do eixo central. As comparações são feitas com o EGSnrc e o MCNP.

2.4.6 Feixe de um Acelerador Clínico Realístico (fótons)

E por último, é simulada uma configuração clínica realística de um feixe de 18 MeV de fótons irradiando um fantoma de várias camadas. As dimensões do fantomas são 30,5 cm x 39,5 cm x 30 cm. As camadas são de 3 cm de água, 2 cm de alumínio, 7 cm de pulmão e, por último, 18 cm de água. Os voxels são de cubos de 1,5 cm x 1,5 cm.

2.5 Erro na Determinação da Dose em Radioterapia

Segundo [Thwaites, 2013], nos últimos 30 anos um número razoável de publicações tem tratado a questão da acurácia em radioterapia. Um dos trabalhos mais importantes nesse aspecto é o [International Commission on Radiation Units e Measurements, 1976] (*ICRU Report 24*). Nesse trabalho, à luz dos dados da época, é aceitável uma acurácia de ± 5% na dose entregue ao volume alvo, muito embora, em situações específicas, pode-se requerer uma acurácia tão boa quanto ± 2%. Adicionalmente, segundo [Souza et al., 2001], para que se tenha sucesso em um tratamento radioterápico, a dose dada ao volume inteiro do tumor não deve variar mais do que ± 5% da dose prescrita.

Capítulo 3

Metodologia

A metodologia apresentada nessa dissertação consiste na implementação de um programa de computador e na sua verificação através de dados de várias simulações disponíveis no *benchmark* [Carrier et al., 2004]. Os resultados das simulações são comparados com os resultados das simulações constantes no *benchmark* realizadas com o GEANT4 e com outros dados desse mesmo documento. O desenvolvimento do programa de computador para a simulação dos casos é levada a cabo por meio do uso do conjunto de ferramentas GEANT4 e da linguagem de programação `C++`.

Além da implementação do programa de simulação para os casos escolhidos, apresentamos um roteiro para a implantação do sistema de simulação GEANT4. Inicialmente, foi escolhida a versão 4.9.3 patch 02 em um *Sistema Operacional* (SO), ou *Operating System* (OS), MacOS X, que era a mais recente à época. No entanto, algum tempo depois houve alguns problemas técnicos e não foi mais possível dar continuidade aos trabalhos com essa versão do GEANT4. Então, optou-se por mudar tanto de versão do GEANT4 quanto a do sistema operacional. Foi utilizada a versão 9.6 do GEANT4 em o SO Scientific Linux CERN 6.3, ambos eram os mais recentes à época. Optou-se também por virtualizar o sistema operacional pois assim ter-se-ia uma facilidade de se fazer cópia de segurança de todo o sistema e também de sua replicação sem a necessidade de passar pelo enfadonho processo de instalação. Para esse último sistema, foi não só criado um roteiro em forma de texto com também em vídeo.

Como foi dito, inicialmente foi escolhida a versão 4.9.3 patch 02 do GEANT4 (`http:`

`//geant4.cern.ch`). O sistema operacional escolhido foi o MacOS X versão 10.6.8 com o compilador `C++` *compilador da linguagem* `C` *com sua versão* `C++` `g++` (gcc) versão 4.2.1 da Apple Inc, pois era a máquina mais acesível. Essa máquina é dotada de um processador Intel Core 2 Duo de 2,4 GHz com 4 GB de memória RAM de 1067 MHz DDR3. Além disso, foram utilizados inicialmente para o desenvolvimento do programa de simulação a *Integrated Development Environment* (IDE) o Eclipse Juno para `C++`, o programa doxygen versão 1.82 para documentação e o sistema GitHub (`https://github.com/sandrorb/Geant4MasterDissertationSimulation`) como sistema de controle de versões. A metodologia de implantação desse sistema é mostrada em detalhes nos apêndices C e D.

Devido aos problemas técnicos mencionados, houve a mudança da versão do sistema operacional MacOS X que passou de 10.6 para 10.9. Houve algumas tentativas frustradas de reinstalação do GEANT4 9.3.p02 quando então optou-se por mudar de versão do GEANT4 e do sistema operacional. Além disso, decidiu-se implantar o sistema de forma virtualizada. Para isso utilizou-se o VMware Fusion para MacOS X. Dessa forma temos um sistema para simulação Monte Carlo com um GEANT4 portável, facilmente replicável, multi-plataforma e que pode ser armazenado e transportado facilmente em dispositivos como discos externos portáteis, *pendrives*, e também em nuvem (`http://l.bitcasa.com/4tq3kp44`).

O sistema virtualizado foi criado com o GEANT4 9.6 e o sistema operacional Scientific Linux CERN 6.3 de 64 bits com o compilador `C++` `g++` versão 4.4.6. O *hardware* continuou o mesmo.

Em linhas gerais, o processo de instalação é complicado e enfadonho, especialmente se o usuário não tem familiaridade com o sistema operacional Linux ou o sistema escolhido não for compatível. Mesmo escolhendo-se um SO compatível, isso não significa que tudo fluirá sem percalços. Uma série de procedimentos deve ser feito ao sistema para compatibilizá-lo com o perfil de instalação que se pretende fazer do GEANT4.

Para o caso das versões 4.9.0 até a versão 4.9.3 o processo envolve, além da instalação dos pré-requisitos, a instalação do GEANT4 por meio da execução de um *script* cha-

mado "`Configure`". A partir da versão 4.9.5, a instalação pode ser realizada por meio do uso do programa "`cmake`". Outra inovação foi que para uma instalação básica dessa última versão não é necessário a instalação do pacote *Class Library for High Energy Physics* (CLHEP) e dos dados para a simulação. Uma versão básica da CLHEP já é distribuída juntamente com o GEANT4 e o conjunto de dados são baixados da internet no momento da instalação do GEANT4.

O GEANT4.9.3.p02 fora obtido na parte de *download* do sítio `http://geant4.cern.ch`, assim como os arquivos dos dados. Já o código fonte da CLHEP fora obtido do sítio `http://proj-clhep.web.cern.ch/proj-clhep/DISTRIBUTION`. No referido sítio, é possível descarregar versões mais antigas. Optou-se pelo programa fonte ao invés de instalar os arquivos já compilados. A versão obtida da CLHEP foi a 2.0.4.7.

Embora a documentação da versão 4.9.0 do GEANT4 diga que o compilador deva ser da versão 3.2.3 ou 3.4.5, sendo que alguns usuários tenham reportado sucesso também com a versão 4.1.1, nessa instalação foi usado o gcc 4.2.1 sem problemas, que é a versão distribuída junto com o DVD do sistema operacional.

O GEANT4 dá nomes próprios às suas bibliotecas de dados. Abaixo são listadas as bibliotecas usadas pelo GEANT4 e suas origens. À frente é escrito o nome da biblioteca segundo o GEANT4 e na descrição, sua origem e outros comentários. Também há, em alguns casos, ligação para o local de origem das bibliotecas.

1. Photon Evaporation e Radioactive Decay: *Evaluated Nuclear Structure Data File* (**ENSDF**) mantida pelo *National Nuclear Data Center* do *Brookhaven National Laboratory*. `http://www.nndc.bnl.gov/nudat2`.
2. G4NDL (com ou sem nêutrons térmicos): Os dados dessa biblioteca vêm principalmente da biblioteca **ENDF/B-VI** que é desenvolvida e mantida pelo *Cross Section Evaluation Working Group* (CSEWG). `http://www.nndc.bnl.gov/csewg`. Os dados originais podem ser obtidos de `http://www.nndc.bnl.gov`.
3. G4SAID: dados do banco de dados **SAID** para próton, nêutron, pion inelástico, elástico e seção de choque troca de carga de núcleos abaixo de 3 GeV.
4. G4NEUTRONXS: dados avaliados são produzidos usando o conjunto de dados **G4NDL** por meio de procedimento de média para nêutrons com energias menores que 10 MeV. Para energias maiores que 20 MeV, as seções de choque são calculadas pelo GEANT4 por meio de suas classes de seção de choque chamadas `G4BGGNucleonInelasticXS` e `G4BGGNucleonElasticXS`. Para o intervalo de energias entre 10 MeV e 20 Mev, é feito uma interpolação linear.

5. G4EMLOW (Baixa energia eletromagnética): **EPDL97** – *Evaluated Photon Data Library*, **EEDL** – *Evaluated Electron Data Library*, **EADL** – *Evaluated Atom Data Library*. `https://wci.llnl.gov/codes/tart`.
6. G4ENSDFSTATE (nuclide state properties database): conjunto de dados de propriedades de estados de radionuclídeos derivados de *Evaluated Nuclear Structure Data File*. `http://www.nndc.bnl.gov/ensdf`.

As bibliotecas de dados do GEANT4 instaladas com a versão 9.6 foram: `G4EMLOW6.32`, `G4NDL4.2`, `G4NEUTRONXS1.2`, `G4PII1.3`, `G4SAIDDATA1.1`, `PhotonEvaporation2.3`, `RadioactiveDecay3.6` e `RealSurface1.0`. Dessas, a biblioteca efetivamente utilizada nessa dissertação foi a `G4EMLOW6.32` (versão 6.32), ou seja, a biblioteca de baixa energia eletromagnética para os seguintes processos físicos: efeito Rayleigh, efeito fotoelétrico, efeito Compton, produção de pares, efeito Auger, ionização, *bremsstrahlung*, aniquilação de pósitron, relaxação atômica e espalhamentos múltiplos. Essa biblioteca do GEANT4 é formada pelas seguintes bibliotecas de dados: **EPDL97**, **EEDL** e **EADL**.

3.1 Simulações

Dos vários casos constantes no *benchmark* escolheu-se um subconjunto para se implementar e simular. Inicialmente foram selecionados os primeiros casos, especialmente porque parecem ser mais simples para usuários inciantes do GEANT4. Essa escolha se deve ao fato de que, além do próprio autor desse trabalho ser um iniciante, o programa de simulação gerado é parte integrante da lista de resultados destinados ao uso dos futuros usuários do sistema que assumimos aqui serem iniciantes. O programa será disponibilizado na internet para descarregamento.

São simulados os 4 primeiros casos pertencentes à primeira parte de duas na qual os autores do *benchmark* consideram "Feixe Largo Paralelo em um Meio Homogêneo", ou seja, a primeira parte da seção 2.4.1.

Nos 4 casos, a fonte é plana e emite elétrons unidirecionalmente em um feixe paralelo de cima para baixo em direção ao fantoma. O fantoma é lateralmente infinito comparado a dimensão lateral da fonte e está 100 cm abaixo da fonte, enquanto a fonte tem dimensões laterais de 5 cm x 5 cm, o fantoma tem dimensões laterais de 100 cm x 100 cm. A listagem 3.1 mostra o trecho de código da definição da fonte.

Lista 3.1: Trecho de código em que se modela a fonte

```
G4double widthZ = 5.0*cm;
G4double posX = widthX * G4UniformRand() - widthX/2.0;
G4double posY = 100*cm;
G4double posZ = widthZ * G4UniformRand() - widthZ/2.0;
G4double px = 0;
G4double py = -1;
G4double pz = 0;
```

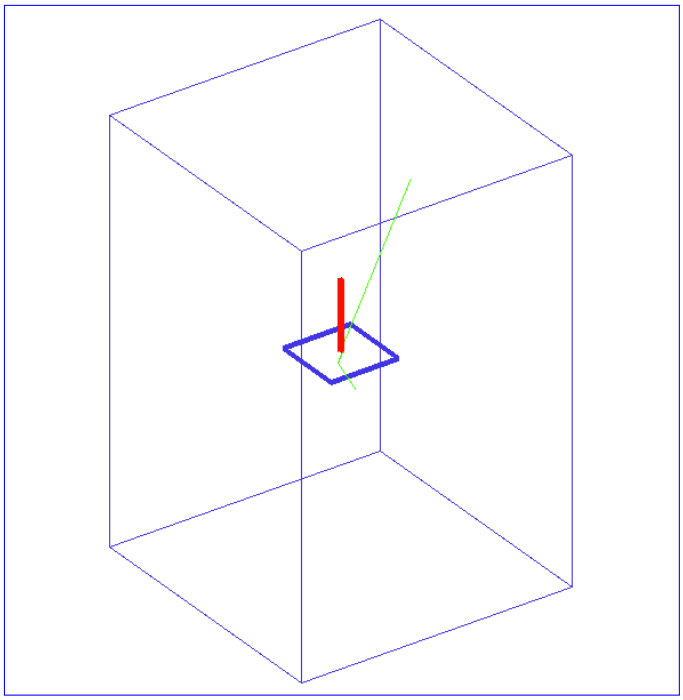

Figura 3.1: Geometria geral para os 4 casos simulados sem ampliação. Simulação realizada com apenas 100 elétrons somente para ilustrar a geometria.

A imagem da figura 3.1 mostra a geometria relativa aos quatro casos simulados. O paralelepípedo azul mostra o que se chama, no jargão GEANT4, de volume mundo ou *World Volume*. Nada é simulado fora desse volume, ou seja, se a partícula sair dele, o GEANT4 não a considera mais. O pequeno quadrado azul no centro refere-se às bordas das camadas lógicas do fantoma que, nesse caso, é de Berílio. Como não há ampliação nessa imagem, não é possível ver as camadas individualmente. Com a finalidade de se obter uma imagem ilustrativa do caminho dos elétrons na simulação, fez-se uma simulação com apenas 100 eventos (100 elétrons). Esses elétrons, ou melhor, seus caminhos, são mostrados em vermelho na figura 3.1. O ponto de partida dos elétrons é escolhido aleatoriamente em uma área quadrada de 5 cm x 5 cm, conforme e mostrado pelo fragmento de código da listagem 3.1. As linhas verdes referem-se ao caminho de fótons originados nos processos de interação dos elétrons com o fantoma.

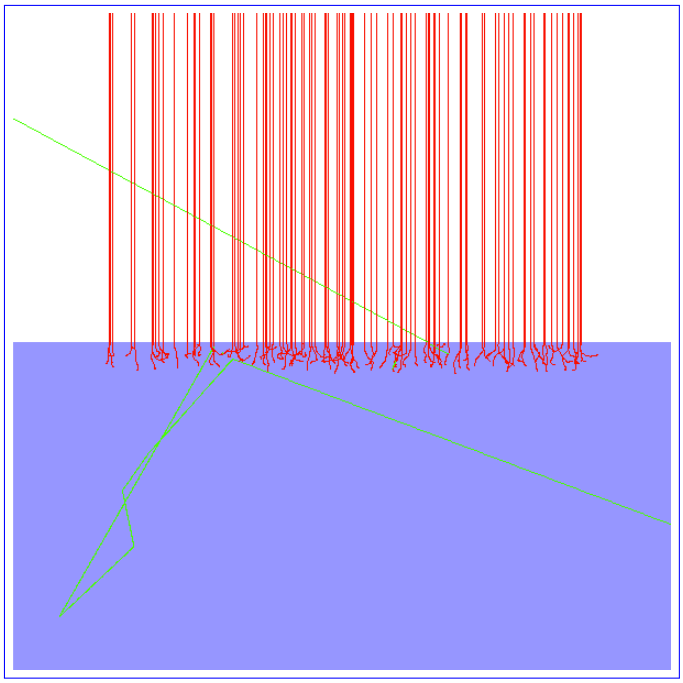

Figura 3.2: Fantoma de Berílio irradiado por 100 elétrons de 1,033 MeV. Essa é a mesma imagem da figura 3.1, mas com uma ampliação de 120 vezes. Para essa ilustração foram simulados apenas 100 elétrons para que fosse possível visualizar seus caminhos.

A imagem da figura 3.2 é uma ampliação de 120 vezes da imagem da figura 3.1. Para a formação dessa imagem, foi feita uma simulação com apenas 100 elétrons para que fosse possível visualizar os caminhos deles dentro do volume de simulação. Da mesma forma, as imagens das figuras 3.2, 3.3 e 3.4. Todas são oriundas de uma mesma simulação realizada meramente para obtenção dessas imagens para ilustrar a geometria e o caminho errático dos elétrons.

Já na imagem da figura 3.3 pode-se ver ainda melhor os elétrons interagindo com o fantoma.

E por último, uma grande ampliação de 1200 vezes foi feita na figura 3.4 para mostrar com bastante clareza o que ocorre dentro do fantoma com os elétrons. Conforme esperado, os elétrons que conseguem maior alcance chegam a atingir uma profundidade aproximada de 50 camadas, o que dá *Alcance máximo dos elétrons* (R_0) ou *Continuous Slowing Down Approximation Range* (CSDA).

O fantoma é dividido em camadas com o propósito de se calcular a dose depositada

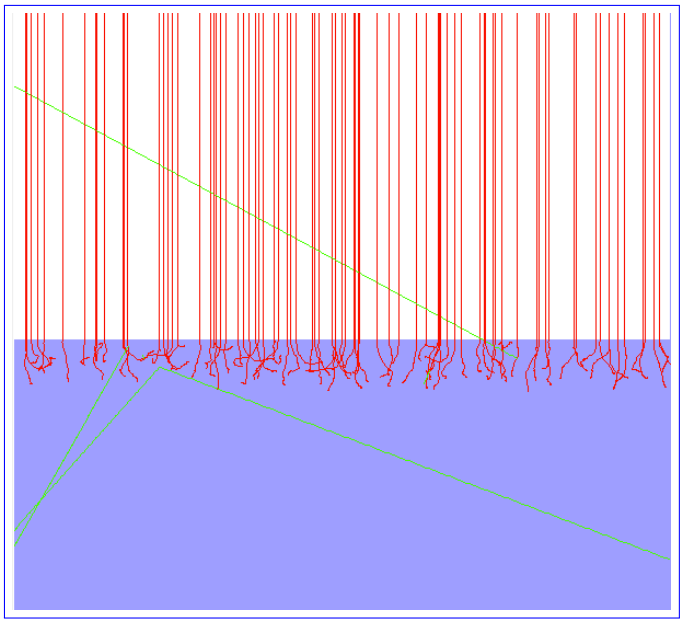

Figura 3.3: Fantoma de Berílio irradiado por 100 elétrons de 1,033 MeV. Essa é a mesma imagem da figura 3.2, mas agora com uma ampliação de 200 vezes.

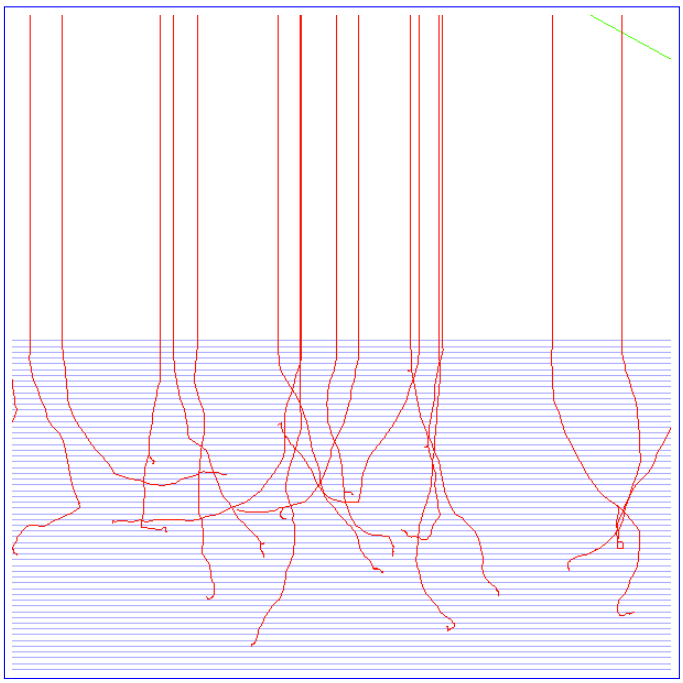

Figura 3.4: Fantoma de Berílio irradiado por elétrons de 1,033 MeV. Essa é a mesma imagem da figura 3.3, mas agora com uma ampliação de 1200 vezes.

em cada uma. A espessura total do fantoma é de R_0 (ou CSDA). Os valores de R_0 são mostrados na tabela 3.1. A espessura da camada é dada por $0,02 \times R_0$, ou seja, o fantoma é dividido em 50 camadas horizontais empilhadas umas sobre as outras. O eixo imaginário que passa pelo centro da fonte é perpendicular à ela é também perpen-

dicular ao plano do fantoma passando pelo seu centro.

Casos	**Material**	**Energia (MeV)**	**R_0 ($\frac{g}{cm^2}$)**
Caso 1	Berílio	0,521	0,224
Caso 2	Berílio	1,033	0,554
Caso 3	Molibdênio	0,5	0,281
Caso 4	Molibdênio	1,0	0,673

Tabela 3.1: Dados de R_0 para as diferentes combinações de material e energia dos elétrons utilizados nos 4 casos simulados.

Os gráficos de dose por profundidade são construídos mostrando-se os resultados das simulações de dose por camada em função da profundidade (camada). O eixo das abscissas refere-se à profundidade dentro do fantoma. Essa profundidade é adimensional dada por z/R_0, onde z é a profundidade e R_0 é o CSDA. O eixo das ordenadas relaciona-se com a dose de radiação depositada em cada camada, e sua unidade é dada por $MeV \cdot g^{-1} \cdot cm^2$. Como é contabilizada a dose depositada em cada camada, logo os gráficos possuem 50 pontos, um para cada profundidade considerada. A energia depositada é proporcional ao fator B dado pela equação 2.16.

3.1.1 Casos Simulados

No caso 1, o fantoma é de Berílio e os elétrons têm energia de 0,521 MeV. No caso 2, o fantoma é de Berílio e os elétrons têm energia de 1,033 MeV. No caso 3, o fantoma é de Molibdênio e os elétrons têm energia de 0,5 MeV. Por último, o caso 4 é de elétrons com 1,0 MeV e o fantoma é de Molibdênio. Essa é, na verdade, a descrição dos 4 primeiros casos apresentados na seção 2.4 subseção 2.4.1 referente ao *benchmark*, e resumidos na tabela 3.1.

Em todas as simulações de cada um dos casos, utilizou-se 1.000.000 de eventos primários, ou seja, foram lançados da fonte em direção ao fantoma 1 milhão de elétrons para cada simulação.

Os autores do *benchmark* fizeram 5 simulações para cada um desses casos. Nesse tra-

balho, são realizadas 50 simulações para o caso 1 afim de se estudar o erro em função do número de simulações. Conforme é mostrado na capítulo 4 dos resultados, o erro que se comete em 5 simulações é suficientemente pequeno comparado ao erro máximo admitido em aplicações médicas, que é de $\pm$ 5%. Vide seção 2.5 sobre erro de dose em radioterapia.

Para se obter os resultados das simulações em um tempo menor, os casos 2, 3 e 4 foram executados fazendo-se 5 simulações para cada caso.

É importante mencionar que nos casos 3 e 4 se fez necessário configurar manualmente o parâmetro *Max Step Length* ou *Max Step Limit* que nada mais é que o comprimento máximo do passo da simulação (equivale ao Δx no MCNP). Esses são os casos nos quais o fantoma é constituído de Molibdênio, cuja densidade é de 10,28 g·cm^{-3}, contra 1,848 g·cm^{-3} do Berílio. Tanto seu número atômico quanto sua densidade são maiores que o número atômico e a densidade do Berílio. Isso confere ao Molibdêniob maior seção de choque macroscópica total para elétrons do que o Berílio. Assim, o comprimento máximo do passo foi ajustado para ser 20% da espessura da camada para os casos 3 e 4.

Capítulo 4

Resultados

Nesse capítulo mostramos os resultados das simulações computacionais como forma de verificação do sistema instalado.

Os gráficos das figuras 4.1, 4.2, 4.5, 4.7 e 4.9 mostram em suas abscissas a profundidade dentro do fantoma. Suas unidades são dadas em termos z/R_0, onde z é a profundidade e R_0 é o alcance máximo dos elétrons da energia e meio considerados. Já nas ordenadas, são plotadas as doses depositadas em cada camada e suas unidades são $MeV \cdot g^{-1} \cdot cm^2$. As mensurações das doses depositadas são tomadas para cada camada, logo, os gráficos possuem 50 pontos, um para cada profundidade considerada. Nesses gráficos, os resultados das simulações são mostrados juntamente com seus correlatos do *benchmark*.

Os gráficos das figuras 4.3, 4.4, 4.6, 4.8 e 4.10 monstram os erros relativos, ou desvios percentuais, dos resultados das simulações em relação aos dados da literatura via *benchmark*.

Também são mostrados, nos gráficos das figuras 4.11 e 4.12, a influência negativa que o comprimento máximo do passo tem quando não é corretamente ajustado.

Foram simulados 4 casos do *benchmark* para a fonte de elétrons plana com feixe paralelo unidirecional e perpendicular ao plano do fantoma semi-infinito. Os casos de 1 à 4 são respectivamente simulações dessa geometria com elétrons e fantomas, conforme descrito na tabela 3.1.

O único resultado obtido com o sistema montado com a versão 9.3 do GEANT4 sobre o MacOS X 10.6 é mostrado no gráfico da figura 4.1 (quadrados azuis). Todos os demais resultados a seguir são oriundos de simulações realizadas no sistema virtualizado com o Scientific Linux CERN 6.3 e o GEANT4 9.6, cuja montagem é descrita nos apêndices A e B.

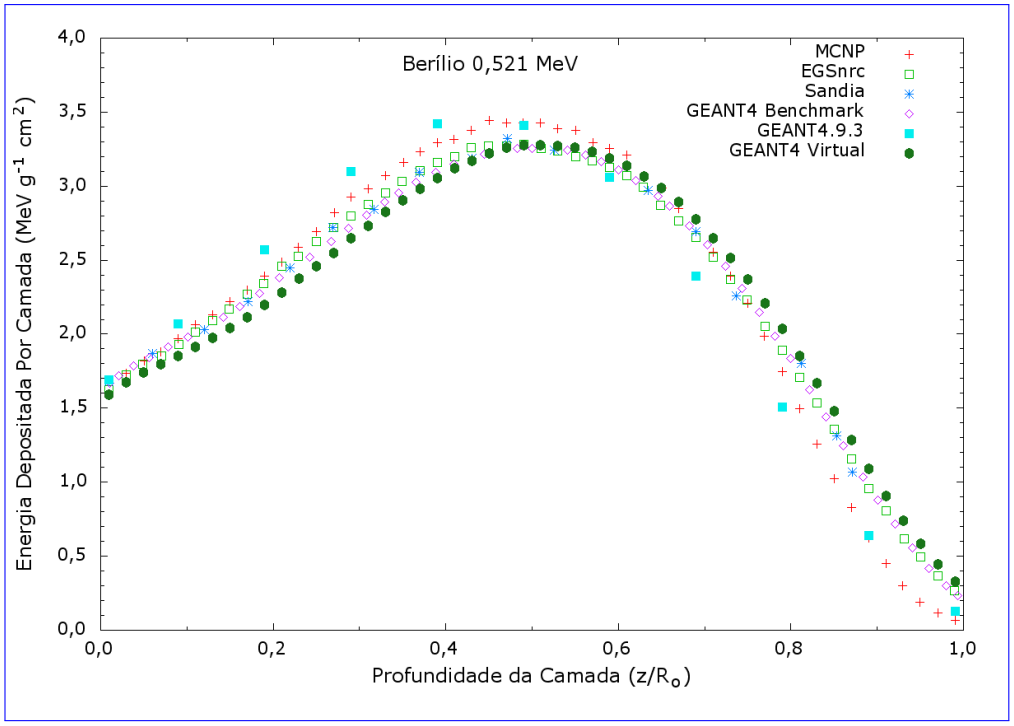

Figura 4.1: Simulação de uma fonte plana unidirecional de elétrons de 0,521 MeV irradiando um fantoma semi-infinito de múltiplas camadas de Berílio. Pontos quadrados em azul referem-se a simulação com a versão 9.3.p02 do GEANT4 instalado no MacOS X enquanto os pontos cujo rótulo é "GEANT4 Virtual", foram simulados com a versão 9.6 do GEANT4 no sistema virtualizado do Scientific Linux CERN 6.3.

O gráfico da figura 4.2, nos mostra o resultado da simulação do caso 1, ou seja, Berílio irradiado por elétrons de 0,521 MeV. Os pontos em verde representam a simulação realizada e as outras são dados obtidos do *benchmark*. Esse caso foi simulado 50 vezes para que fosse possível realizar um estudo do erro para se ter uma ideia de quantas simulações seriam necessárias para o presente estudo.

A figura 4.3 nos mostra o erro relativo (coeficiente de variação) para cada uma das camadas em função do número de simulações. Nota-se que, de um modo geral, quanto mais profunda a camada, maior o erro relativo. O maior erro relativo não supera 0,5%,

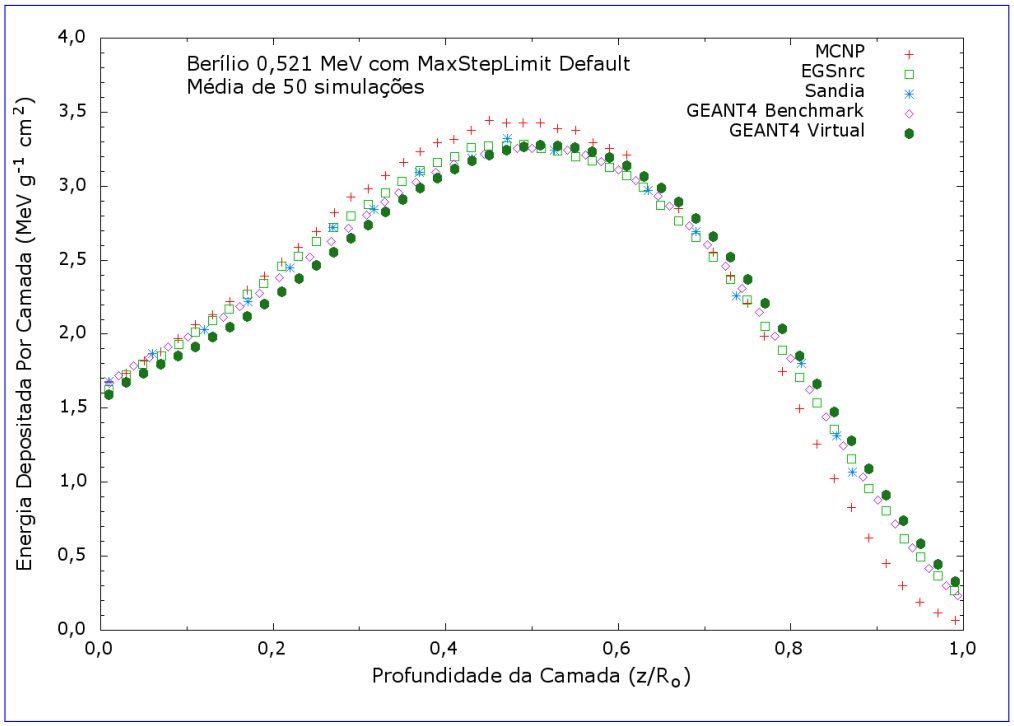

Figura 4.2: Simulação com elétrons de 0,521 MeV irradiando Be no GEANT4 virtual. Os pontos triangulares em azul claro são os resultados médios de 50 simulações. Simulações realizadas com o comprimento do passo (*Max Step Limit*) default do sistema.

valor bem abaixo do que se exige como erro máximo em aplicações de dosimetria em radioterapia. Baseado nesse estudo, decidiu-se simular os casos restantes com repetição de 5 vezes, da mesma forma que foi feito no *benchmark*.

De forma geral, o resultado da simulação tem boa concordância com os dados da literatura, pelo menos até aproximadamente a profundidade de 0,8. Essa concordândia está dentro de ± 10% de erro. Para profundidades maiores de 0,8 os dados divergem.

O gráfico da figura 4.4 mostra então o erro relativo das 50 simulações para o caso 1 juntamente com os repectivos dados da literatura. A simulação concorda em ±5% com Sandia, EGSnrc e as simulações do *benchmark* para profundidades de até $0,8$. Não há uma boa concordância com o MCNP.

O gráfico da figura 4.5 mostra os dados pertinentes ao caso 2, tanto a simulação quanto os dados da literatura obtidos no *benchmark*. Os pontos da simulação são resultados da média de 5 simulações.

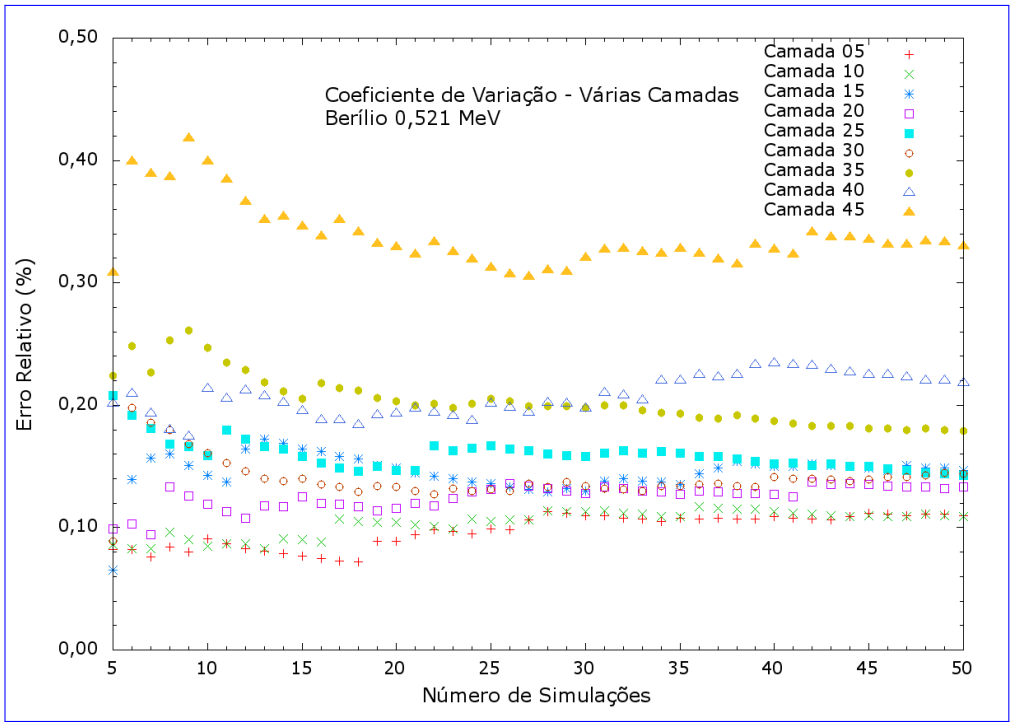

Figura 4.3: Erros relativos (coeficiente de variação) de algumas camadas em profundidades selecionadas em função do número de simulações. Nenhum erro supera $0,5\%$.

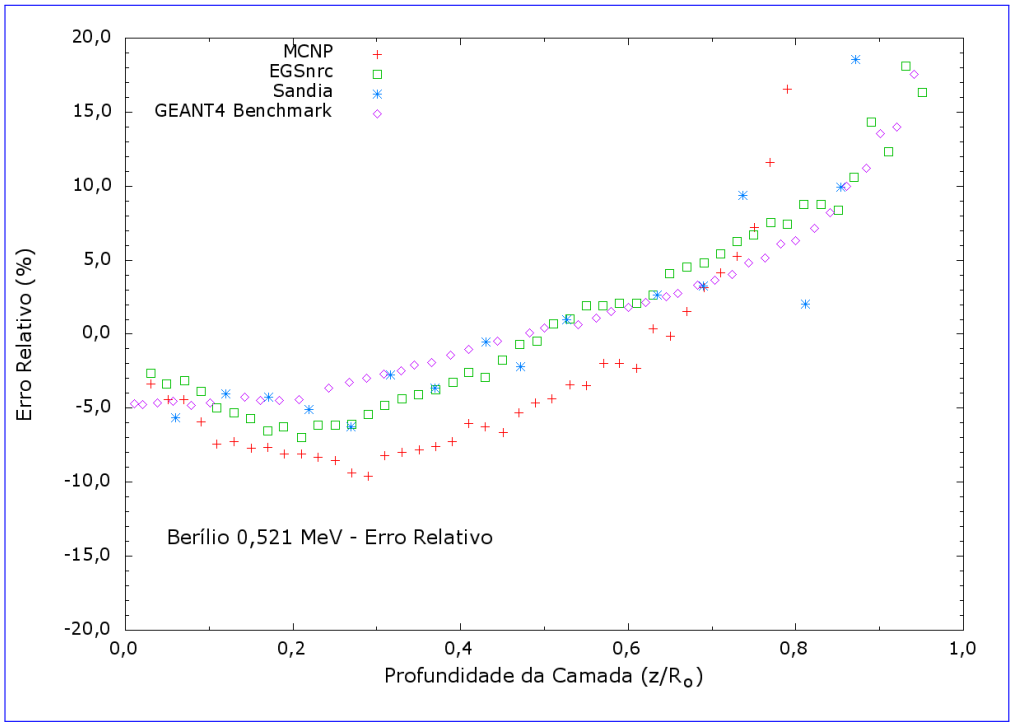

Figura 4.4: Erro relativo entre os dados médios das simulações do caso 1 e os dados do *benchmark*. Foram 50 simulações com 1.000.000 de eventos cada.

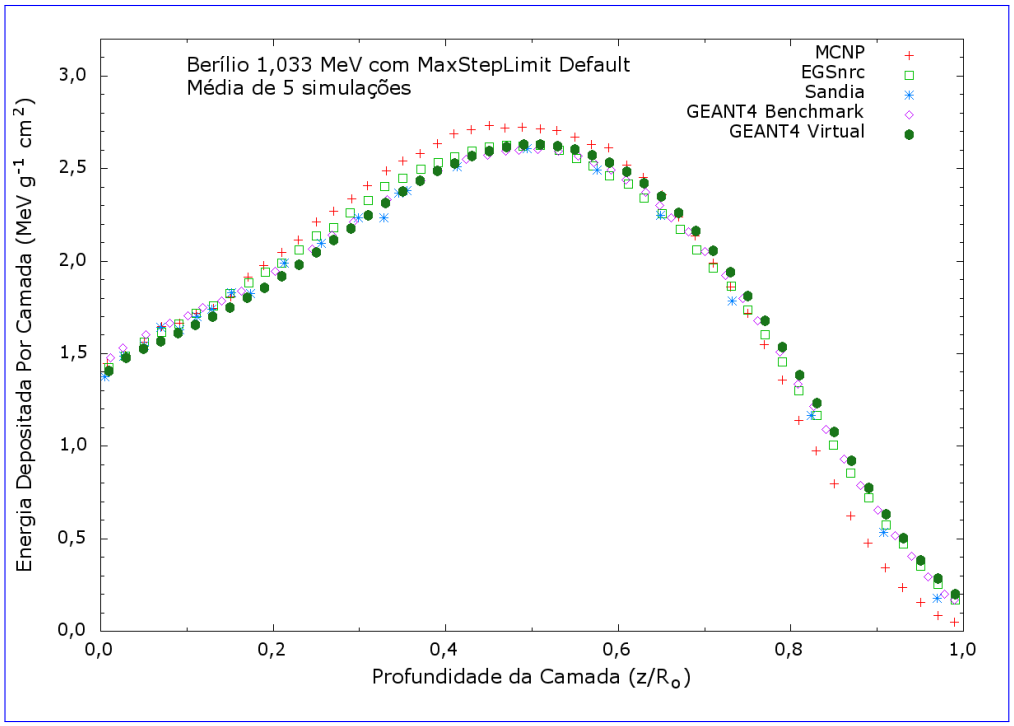

Figura 4.5: Simulação de uma fonte plana unidirecional de elétrons de 1,033 MeV irradiando um fantoma semi-infinito de múltiplas camadas de Berílio. Os pontos referentes a simulação são médias de 5 simulações de 1.000.000 de eventos cada.

O gráfico da figura 4.6 mostra o desvio relativo entre o resultado da média de 5 simulações do caso 2 e os dados obtidos do *benchmark*.

Como pode ser visto no gráfico da figura 4.6, os dados da simulação do caso 2 estão em melhor acordo com a literatura do que os do caso 1, pelo menos até a profundidade de 0,8. A exceção é o código MCNP ao qual a simulação tem um desvio maior, mesmo assim, menor do que 10%. De uma forma geral, até a profundidade de 0,8, a simulação concorda com os dados da literatura dentro de uma margem de ± 5%.

O gráfico da figura 4.7 mostra o resultado da média de 5 simulações do caso 3, cada uma com 1.000.000 de eventos. O resultado é mostrado juntamente com os dados da literatura obtidos pelo *benchmark*.
Conforme mostra o gráfico da figura 4.8, há uma boa concordância entre os dados da simulação do caso 3 com a literatura até a profundidade de 0,4. Até essa profundidade, os resultados concordam dentro de ± 5%.

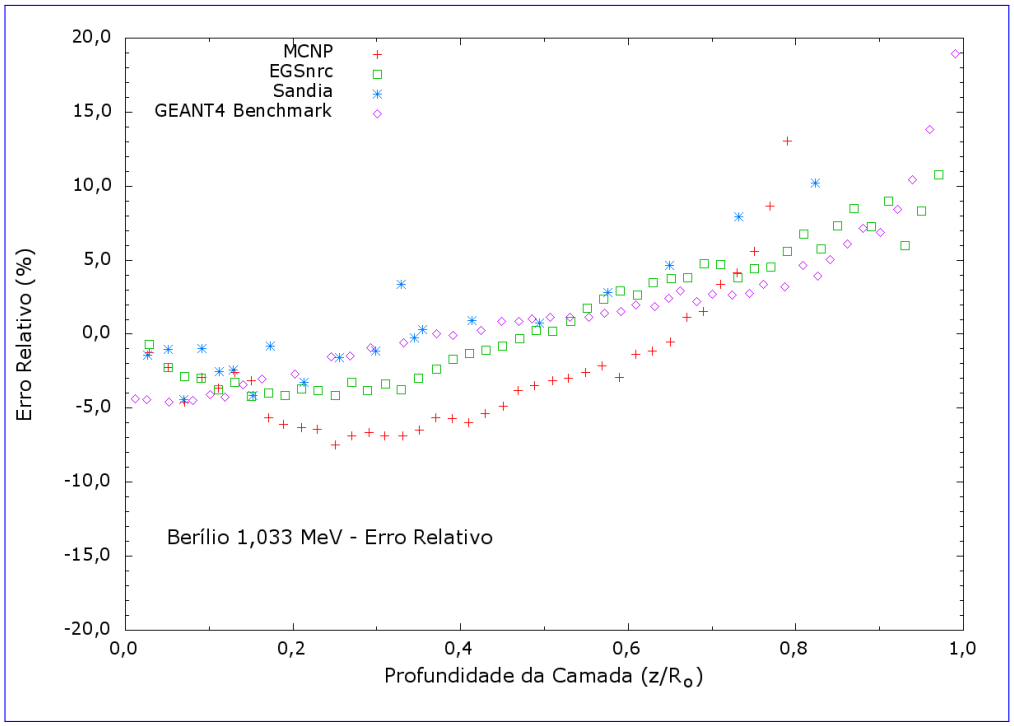

Figura 4.6: Erro relativo entre os dados médios das simulações do caso 2 e os dados do *benchmark*. Foram 5 simulações com 1.000.000 de eventos cada.

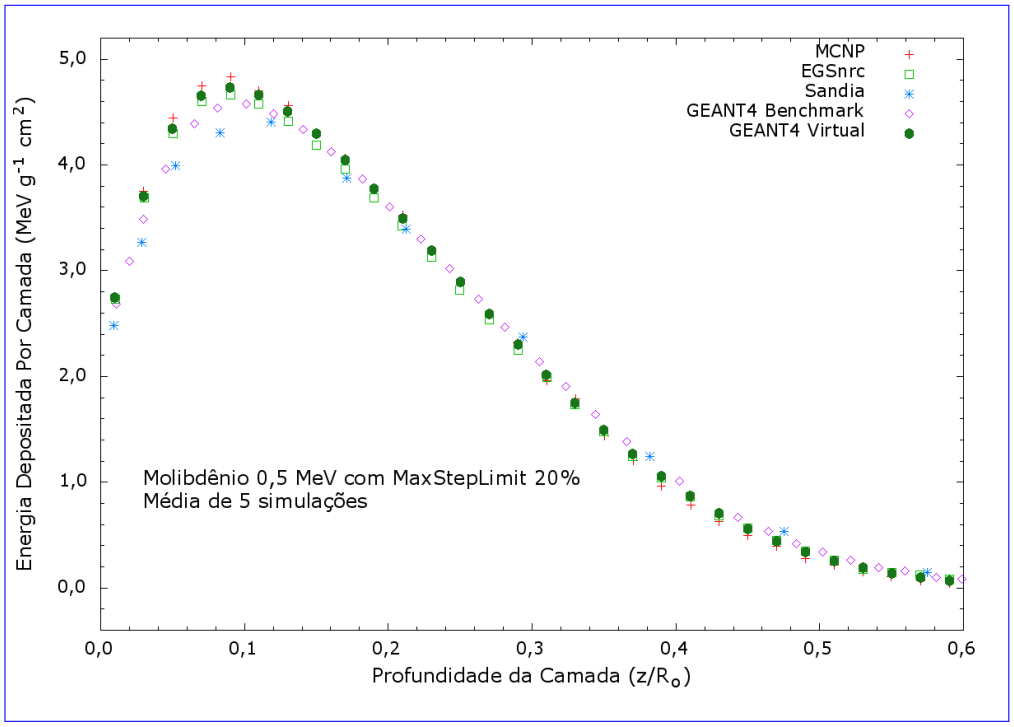

Figura 4.7: Simulação de uma fonte plana unidirecional de elétrons de 0,5 MeV irradiando um fantoma semi-infinito de múltiplas camadas de Molibdênio. Os pontos referentes a simulação são médias de 5 simulações de 1.000.000 de eventos cada.

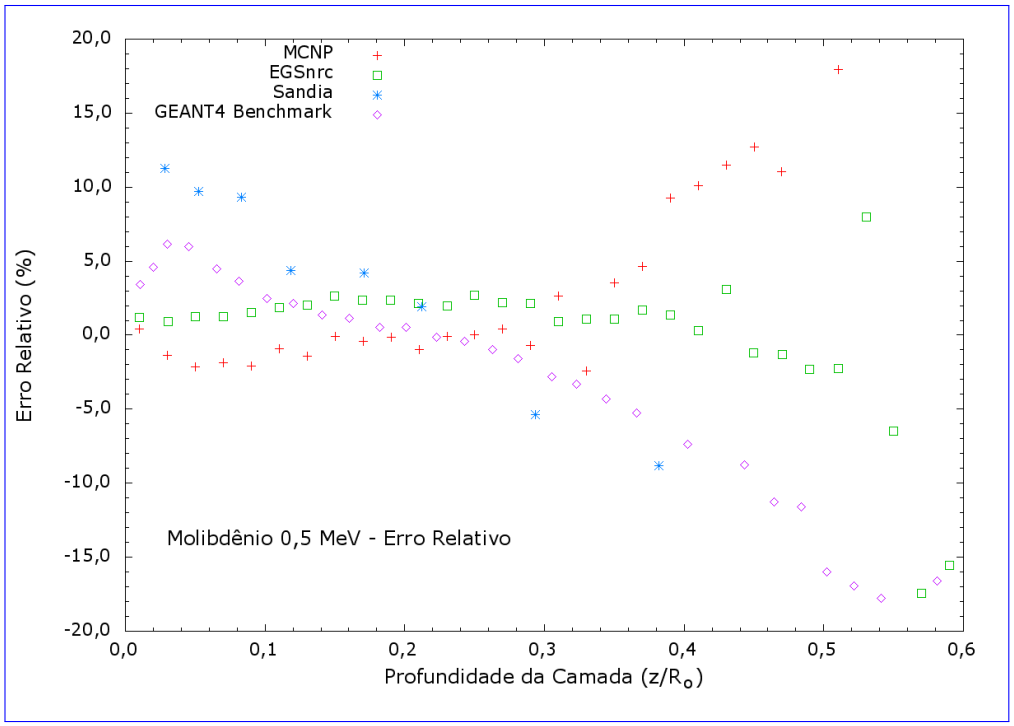

Figura 4.8: Erro relativo entre os dados médios das simulações do caso 3 e os dados do *benchmark*. Foram 5 simulações com 1.000.000 de eventos cada.

O gráfico da figura 4.9 mostra as médias de 5 simulações de 1.000.000 de eventos cada do caso 4. Para efeito de comparação o gráfico mostra também os dados da literatura obtidos do *benchmark*.

Os erros relativos entre os dados médios das simulações do caso 4 e os dados da literatura são mostrados no gráfico da figura 4.10. Novamente, como no caso 4, o erro entre a simulação e a literatura fica dentro de $\pm$ 5% somente até a profundidade de 0,4.

Conforme explicado na seção 3.1.1, o comprimento máximo do passo teve um papel fundamental nas simulações dos casos 3 e 4. Esse parâmetro teve que ser ajustado manualmente ao invés de se usar o valor padrão ou *default* do sistema. Como a seção de choque macroscópica total do Molibdênio para elétrons é maior do que a do Berílio para elétrons, o comprimento máximo do passo foi ajustado para ser 20% da espessura da camada para os casos 3 e 4 nos quais o fantoma é formado de Molibdênio. Nos casos 1 e 2 não houve diferença significativa ao se usar esse parâmetro ou o valor padrão.

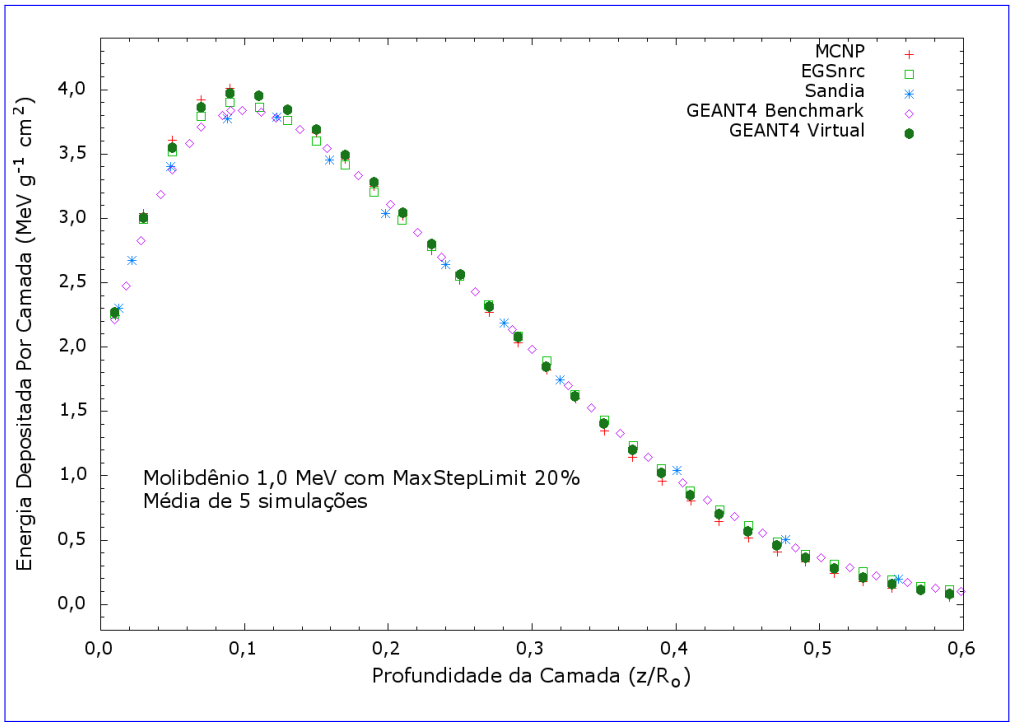

Figura 4.9: Simulação de uma fonte plana unidirecional de elétrons de 1,0 MeV irradiando um fantoma semi-infinito de múltiplas camadas de Molibdênio. Os pontos referentes a simulação são médias de 5 simulações de 1.000.000 de eventos cada.

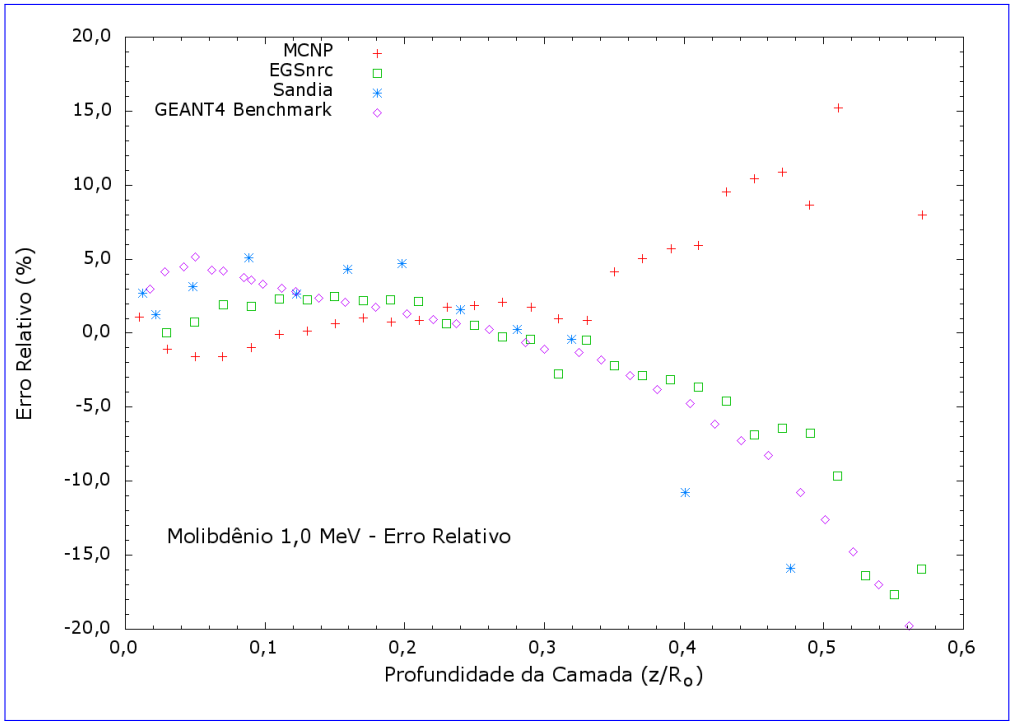

Figura 4.10: Erro relativo entre os dados médios das simulações do caso 4 e os dados do *benchmark*. Foram 5 simulações com 1.000.000 de eventos cada.

Os gráficos das figuras 4.11 e 4.12 mostram como fica a simulação dos casos 3 e 4, respectivamente, quando se usa o comprimento máximo do passo padrão do sistema.

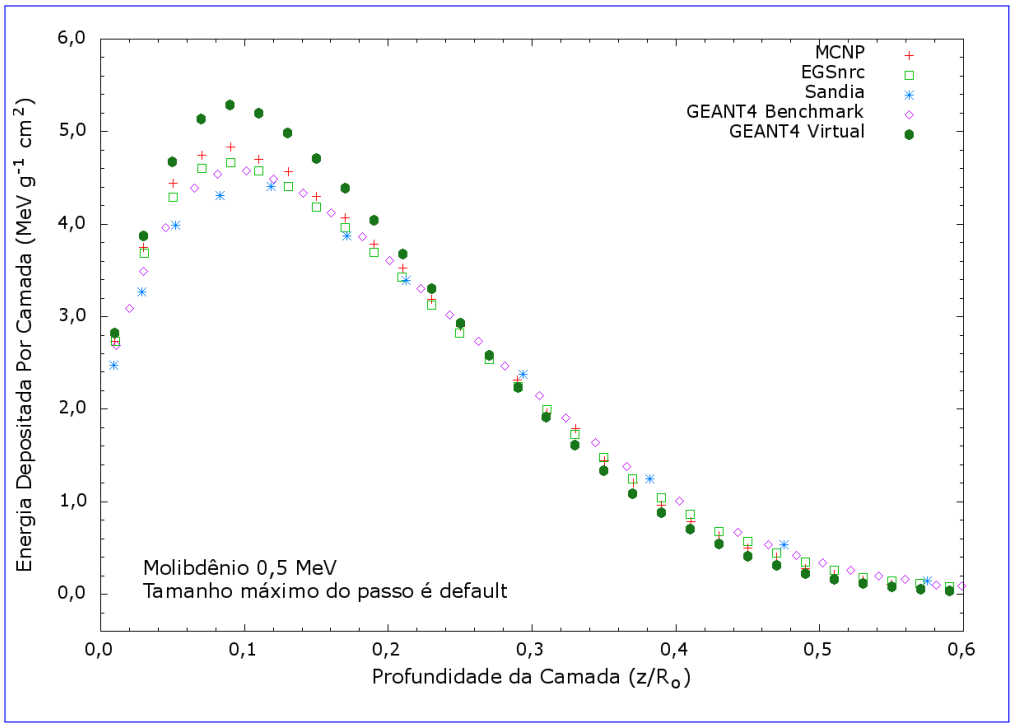

Figura 4.11: Simulação única de 1 milhão de eventos do caso 3 com o tamanho máximo do passo ajustado pelo sistema (padrão).

Todos os gráficos possuem um comportamento em comum. Na superfície do fantoma a dose é menor vai aumentando até que, em uma dada profundidade, há um pico e então começa a cair numa forma monotonamente decrescente. Esse é um comportamento típico desse tipo de sistema. A explicação para isso é que há uma concorrência de alguns efeitos, tais como: o *buildup*, a blindagem que as camadas superiores excercem sobre as inferiores. Há também efeitos do "endurecimento" do feixe e, para o caso de fontes pontuais, o inverso do quadrado da distância. O *buildup* está relacionado com o equilíbrio eletrônico. Nas camadas superiores, há mais elétrons sendo arrancados e poucos depositando energia localmente. A medida que se aprofunda no fantoma, o equilíbrio eletrônico vai sendo atingido, ou seja, a quantidade de elétrons portando energia que absorveu da radiação que o ionizou e que deixa um certo volume é a mesma quantidade dos elétrons que vêm de camadas mais acima e que deposita energia localmente. A medida que se aprofunda no fantoma, além do pico, o equilíbrio eletrônico existe, mas o efeito de blindagem que as camadas acima têm é cada vez maior e a dose

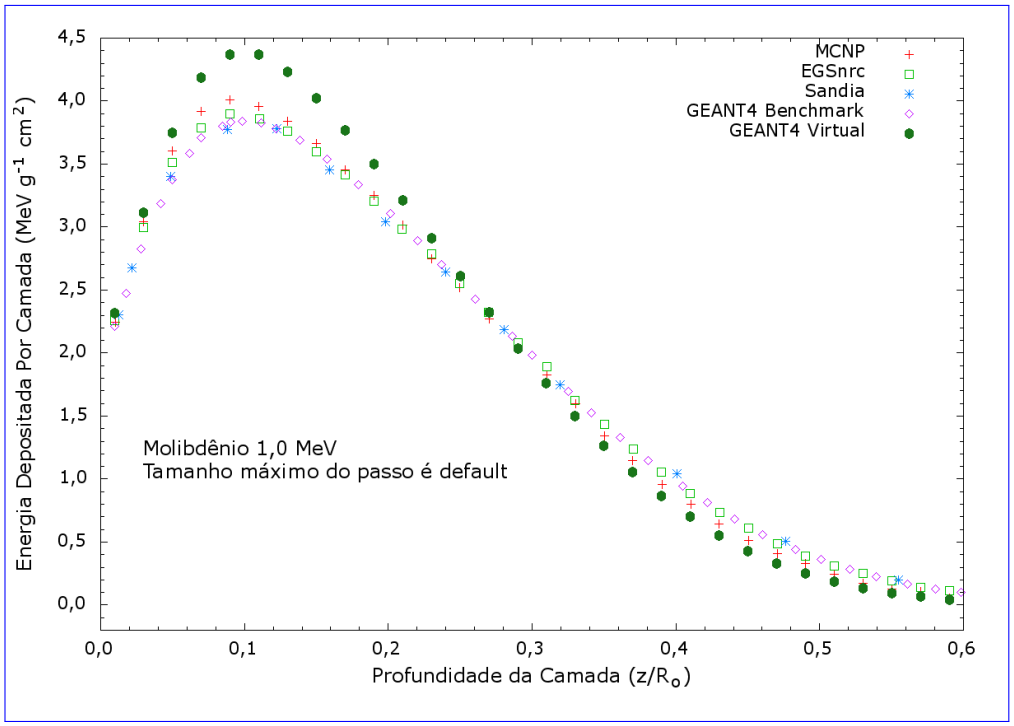

Figura 4.12: Simulação única de 1 milhão de eventos do caso 4 com o comprimento máximo do passo ajustado pelo sistema (padrão).

vai diminuindo com a profundidade.

Considera-se, como parte dos resultados, os procedimentos para a obtenção de um sistema operacional Scientific Linux CERN 6.3 completo para o GEANT4 descrito no apêndice A e os procedimentos para obtenção de um sistema GEANT4 completamente funcional descrito no apêndice B e nos apêndices C e D nos quais são mostrados os procedimentos para se ter um sistema GEANT4 9.3.p02 e o 9.6.

Além dos dos resultados acima, foram produzidos os vídeos tutoriais que podem ser encontrados no sítio do *Youtube* na internet, cuja ligação à eles é `http://www.youtube.com/playlist?list=PLKorLi2cUF3vOYrUBOa4_aWNDsy4c6TuS`.

Capítulo 5

Conclusões e Atividades Futuras

Apesar de o código GEANT4 ser usado há um bom tempo, ainda não é um código amplamente difundido e utilizado, embora possua grandes vantagens como, por exemplo, ser um código fonte aberto e gratuito. O fato de esse código ser baseado em `C++` faz com que seja naturalmente limitado o número de pessoas que possam utilizá-lo de forma imediata. Adicionalmente, uma das coisas que mais tem limitado o uso do GEANT4 é a complexidade de todo o processo relacionado à sua implantação, ou seja, configuração do sistema operacional para recebê-lo com todos os pré-requisitos: configuração, compilação e instalação.

Dessa forma, este trabalho apresentou uma metodologia para a implantação de um sistema de simulação Monte Carlo com o código GEANT4 através de um tutorial passo a passo com o intuito de aumentar o número de usuários que detêm conhecimento dessa ferramenta, fortalecendo assim a difusão e o aperfeiçoamento do código GEANT4.

Para os casos 1 e 2, os erros relativos entre as simulações realizadas nesse trabalho e os dados constantes no *benchmark* então em boa concordância (dentro de $\pm 5\%$). A exceção é a comparação com o MCNP em que os desvios podem chegar à $\pm 10\%$, ou mesmo superá-lo, como é o caso 1 à profundidades maiores que $0,9$.

Para os casos 3 e 4, os erros relativos entre as simulações estão acima de $\pm 10\%$ para profundidades superiores à $0,4$. Abaixo dessa profundidade, os erros não superam os $\pm 5\%$ para o caso 4, mas para o caso 3, há desvios consideráveis quando a comparação

é entre a simulação e o Sandia.

De forma geral, as simulações exibiram resultados aceitáveis em comparação aos dados presentes no *benchmark* mostrando que o sistema instalado pode ser uma ferramenta muito interessante para simulações com elétrons em meios densos. Isso pode indicar que simulações ná área de física médica em radioterapia pode se beneficiar sobremaneira dessa ferramenta, inclusive para simulações nas quais haja meios densos como ossos.

Indica-se como objeto de futuros estudos o efeito de alguns parâmetros de configuração da simulação já que o presente estudo claramente verificou que o parâmetro comprimento máximo do passo infuencia nos resultados, assim como outros parâmetros podem fazê-lo. Adicionalmente, estudos de simulações envolvendo fótons em tecidos moles pode ser também muito interessante, ou seja, estudos envolvendo casos de maior relevância clínica.

Outra possibilidade de sequência a esse trabalho é a realização de simulações envolvendo geometrias mais complexas ou mesmo elaborar um tutorial envolvendo a linguagem de programação `C++`, programação orientada ao objeto em associação ao uso das bibliotecas do conjunto de ferramentas GEANT4.

Referências Bibliográficas

Agostinelli, S. et al. GEANT4 – A Simulation Toolkit. Nuclear Instruments and Methods in Physics Research A, 506:250–303, 2003.

Allison, J. et al. GEANT4 Developments and Applications. IEEE Transactions on Nuclear Science, 53 No. 1:270–278, 2006.

Almansa, J. F., Guerrero, R., Al-Dweri, F. M. O., Anguiano, M. e Lallena, A. M. Dose Distribution in Water for Monoenergetic Photon Point Sources in the Energy Range of Interest in Brachytherapy: Monte Carlo Simulations with PENELOPE and GEANT4. Radiation Physics and Chemistry, 76:766–773, 2007.

Bielajew, A. F. Fundamentals of the Monte Carlo Method for Neutral and Charged Particle Transport. setembro 2001. University of Michigan, Department of Nuclear Engineering and Radiological Sciences.

Bonifácio, D. A. et al. Validação do Código GEANT4 para a Produção e Detecção de RX na Faixa de Energia de Radiodiagnóstico. Dissertação de Mestrado em Ciências, Instituto de Física, Universidade de São Paulo, 2007.

Carrier, J. F., Archambault, L., Beaulieu, L. e Roy, R. Validation of GEANT4, an Object-Oriented Monte Carlo Toolkit, for Simulations in Medical Physics. Medical Physics, 31(3):484–492, March 2004.

CERN. Geant4 User's Guide for Application Developers, Novembro 2012a. URL `http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/fo/BookForAppliDev.pdf`. Acesso em 15 de maio de 2013.

CERN. Physics Reference Manual, 2012b. URL `http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/PhysicsReferenceManual/fo/PhysicsReferenceManual.pdf`. Acesso em 30 de janeiro de 2013.

Chibani, O. e Li, X. A. Monte Carlo Dose Calculations in Homogeneous Media and at Interfaces: A Comparison Between GEPTS, EGSnrc, MCNP and Measurements. Med. Phys., 29:835–847, 2002.

Cochran, R. G. e Tsoulfanidis, N. The Nuclear Fuel Cycle: Analysis and Management. 1999. American Nuclear Society – La Grange Park, Illinois, USA, 2a. edição, 1999.

David, M. F. Programação Orientada a Objetos: Uma Introdução, janeiro 2007. URL `http://www.hardware.com.br/artigos/programacao-orientada-objetos`. Acesso em 30 de janeiro de 2013.

Eckhard, R. Stan Ulam, John von Neumann and the Monte Carlo Method. Los Alamos Special Issue, pages 131–141, 1987.

International Commission on Radiation Units e Measurements. Determination of absorbed dose in a patient irradiated by beams of x or gamma rays in radiotherapy procedures. ICRU Report 24, 1976.

Lamarsh, J. R. e Baratta, A. J. Introduction to Nuclear Engineering. Prentice Hall, 2001. 3^a edição.

Lockwood, G. J., Ruggles, L. E., Miller, G. H. e Halbieib, J. A. Calorimetric Measurement of Electron Energy Deposition in Extended Media – Theory vs Experiment. SANDIA Report, SAND79–0414 l UC–34a, Sandia National Laboratories, 1987.

Malthez, A. L. M. C. Aplicabilidade e Validação do GEANT4 para Fótons e Elétrons em Radioterapia. Faculdade de Engenharia Elétrica e de Computação – FEEC, UNICAMP, Dissetação de Mestrado, 2011.

Marsaglia, G. e Zaman, A. A new class of random number generators. Annals of Applied Probability, 1:462–480, 1991.

Marsaglia, G., Zaman, A. e Tsang, W. W. Toward a universal random number generator. Statistics and Probability Letters, 8:35–39, 1990.

Metropolis, N. The Beginning of the Monte Carlo Method. Los Alamos Science Special Issue, pages 125–130, 1987.

Metropolis, N. e Ulam, S. The Monte Carlo Method. Journal of the American Statistical Association, 247 (44):335–341, 1949.

Neumann, J. V. e Ulam, S. Random Ergodic Theorems. Bulletin A. M. S., Abstract 51-9-165, 1945.

Ricarte, I. L. M. Programação Orientada a Objetos: Uma Abordagem com Java. Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, 2001.

Souza, C. N. d., Monti, C. R. e Sibata, C. H. Recomendações para se Evitar Grandes Erros de Dose em Tratamentos Radioterápicos. Radiol Bras, 34:29–37, 2001.

Stacey, W. M. Nuclear Reactor Physics. 2007. Wiley-VCH, Weinheim, Alemanha, 2a. edição.

Thwaites, D. Accuracy Required and Achievable in Radiotherapy Dosimetry: have modern technology and techniques changed our views? Journal of Physics, 444: 1–17, 2013.

Yoriyaz, H. Método de Monte Carlo: princípios e aplicações em Física Médica. Revista Brasileira de Física Médica, 3:141–149, 2009.

Apêndice A

Instalação do Scientific Linux CERN 6.3

Obtenção e Instalação do Scientific Linux CERN 6.3

Damos a seguir um roteiro de como obter e instalar o Scientific Linux CERN 6.3, ou simplesmente SLC6.3, que é uma distro (distribuição) Linux elaborada pelo Fermilab e pelo CERN, além de outros laboratórios e universidades ao redor do mundo. O principal objetivo dessa distro é reduzir esforços e evitar trabalho sobreposto para se ter uma distro para diversos trabalhos e experimentos. Essa distro pode ser obtida a partir de `https://www.scientificlinux.org`.

Como dito acima, devido sua origem, o *Scientific Linux CERN* (SLC) foi escolhido aqui por ser a distro Linux que tem a maior "vocação" para o GEANT4, ou seja, tem menos probabilidade de exibir incompatibilidades durante os processos de configuração, compilação, instalação e uso do GEANT4. Na verdade, essa é uma das distros oficialmente suportadas pelo GEANT4.

A versão do SLC escolhida foi a 6.3 que é a mais recente no momento da elaboração desse documento. Instruções em vídeo foram elaboradas e podem ser encontradas no sítio *Youtube* da empresa *Google* no endereço da internet: `http://www.youtube.com/playlist?list=PLKorLi2cUF3vOYrUBOa4_aWNDsy4c6TuS`. Esse endereço leva à uma lista de vídeos na qual os três primeiros estão relacionados à esse tópico. Essa mesma lista pode ser acessada por meio dessa ligação curta: `http://bit.ly/15H1It0`.

A versão 6.3 do SLC foi escolhida por ser a mais recente versão dessa distro que é oficialmente suportada pela versão mais recente do GEANT4 no momento que esse documento estava em elaboração, a saber, versão 9.6.

Dentre as plataformas suportadas e testadas estão para o GEANT4 6.0 estão:

1. Scientific Linux CERN 5 com o gcc 4.1.2 ou 4.3.x para processadores de 32 ou 64 bits;
2. Scientific Linux CERN 6 com gcc 4.6.X, 64bit;
3. MacOS X 10.7 (Lion) e 10.8 (Mountain Lion) com gcc 4.2.1 (Apple), 64bit;
4. Windows 7 com Visual Studio 10 (VS2010).

GEANT4 também foi compilado com sucesso em distribuições Linux tais como, Debian, Ubuntu e openSUSE. No entando, o manual diz que distros outras que não a SLC não são s uportadas oficialmente. Além dessas distros Linux não oficiais, o GEANT4 também foi compilado com sucesso no MacOS X 10.6.8 (Snow Leopard) with gcc 4.2.1 (Apple) que também não tem suporte oficial. Além desses, outros sistemas também foram testados, mas não tem suporte. São eles: Scientific Linux CERN 5 with gcc 4.7.X, 64bit; Scientific Linux CERN 5 with Intel `C++` 13, 64bit; MacOS X 10.7 (Lion) and 10.8 (Mountain Lion) with Clang 3.1 e Windows 7 with Visual Studio 9 (VS2008).

Embora as simulações dessa dissertação tenham sido feitas com o GEANT4 versão 9.3 pack 02 em um MacOS X 10.6.8 com o gcc 4.2.1, vide capítulo metodologia dessa dissertação, esse documento detalha, para os futuros usuários, a implantação do sistema de simulação com as mais recentes versões do GEANT4 9.6 com SLC 6.3 e do compilador gcc, a saber, SLC 6.3 kernel 2.6.x 64bits com gcc 4.4.6

A.1 Obtendo o Scientific Linux CERN 6.3

Esse passo a passo foi escrito considerando os acessos ao sítio do Scientific Linux na data de 11 de dezembro de 2012. Mesmo assim, a forma de obtenção dos DVDs tenha mudado, esse passo a passo pode dar um norte em como obtê-los e como testá-los quanto a sua integridade.

1. Vá ao site `http://www.scientificlinux.org`;

2. Clique em download no menu à esquerda;

3. Escolha a opção "Public Torrent"para descarregar o "link torrent" para ser usado com o programa BitTorrent. No sistema de "torrent" o arquivo a ser baixado usualmente possui uma cópia em diversos computadores na internet. O arquivo baixado no sítio do Scientific Linux é usado por um programa de descamento de torrent como o BitTorrent para localizar os lugares na internet onde estão as cópias do arquivo a ser baixado. O programa de descarregar torrents então começa a descarregar dessas diversas cópias ao mesmo tempo. Isso confere a esse sistema uma maior velocidade de descarregamento. Se preferir, pode-se baixá-lo da forma tradicional, ou seja, descarregar o arquivo "iso" diretamente do sítio do Scientific Linux. Pode-se obter o programa de descarregamento de torrents nesse lugar: `http://www.bittorrent.com`.

   Se desejar fazer o descarregamento dos ISOs dos DVDs da forma tradicional, veja ao final dessa lista como fazê-lo.

4. Escolha a versão 6.3;

5. Escolha a versão x86_64, que é a versão para processadores 64 bits;

6. Clique em"iso";

7. Descarregue para o seu computador os dois arquivos de torrents dos DVDs 1 e 2 x86_64 "Everything". É também interessante baixar os arquivos "message digest" para checar se os arquivos foram baixados sem erros. Os arquivos que contém as "messages digest" são arquivos onde foram gravados as "assinaturas" dos arquivos (DVDs) a serem descarregados. Depois de descarragar os DVDs de instalação do SLC6.3, pode-se usar um programa de cálculo dessas "mensages" para comprar com a que foi calculada antes do descarregamento. Qualquer erro na descarga de um bit se quer, dará uma "message digest" diferente. Veja o conteúdo do arquivo SHA256SUM. A cadeia de caracteres na frente do nome do arquivo é uma "assinatura" única relativa ao conteúdo do arquivo. Qualquer que seja a mudança em ao menos um bit, essa cadeia de caracteres muda;

8. Descarregue então os arquivos com os "message digest";

9. É sempre bom também ter o arquivo "readme";

10. Com seu programa de descarregamento de torrents favorito, e usando os arquivos de ligação dos torrents, descarregue os DVDs de instalação do SLC6.3; O DVD1 tem tamanho 4,1GB. O DVD2 é de 1,1GB.
    O sistema de torrents baixa diferentes partes do arquivo de diferentes fontes ao mesmo tempo, diferentemente da forma tradicional que baixa o arquivo sequenciamente de apenas uma fonte. Isso confere a esse tipo de sistema um descarregamento muito mais rápido;

11. Depois de realizados o descarregamento de ambos os DVDs, pode-se calcular o "message digest" para cada DVD afim de comparar o valor obtido aqui com o original. Para isso, em sistemas MacOSX ou Linux, pode-se usar o programa openssl. Na linha de comando (console) e vá para o diretório (pasta) onde se encontram os arquivos ISO dos DVDs descarregados, que nesse caso, o nome do DVD1 é "`SL-63-x86_64-2012-08-02-Everything-DVD1.iso`". Então para calcular dê o seguinte comando:
    ```
    openssl -sha1 SL-63-x86_64-2012-08-02-Everything-DVD1.iso
    ```

    Pode demorar alguns minutos pois o programa irá ler byte por byte para o cálculo. O resultado será mostrado no console. Compare esse resultado com aquele que consta no arquivo de SHA descarregado ou no mesmo arquivo no prórpio sítio. Se o resultado tiver qualquer diferença com o original, então houve erro no descarregamento.

    Repita o procedimento para o DVD2.

    Uma forma alternativa de se testar os DVDs é o teste que nos é oferecido no início da instalação do sistema. Esse teste é igualmente interessante e pode substituir o teste proposto acima com o openssl.

12. Pode-se também baixar os ISOs dos DVDs da forma tradicional;

13. É só ir a página de downloads do site do Scientific Linux e clicar em um dos links sob o rótulo "Scientific Linux 6.x";

14. Há duas opções de servidores, ftp1 e ftp. Qualquer um deve servir. Atenção para a escolha do ISO. Escolha a opção 64 bits (x86_64);

15. Como no caso anterior com os links dos torrents, baixe os dois DVDs "Everything" de 64 bits. Se quiser checar se os arquivos baixados estão íntegros, então baixe também o arquivo de "message digest"correspondente. Veja acima como executar o programa openssl contra o arquivo descarregado para comparação com a cadeia de caracteres original;

16. Para a instalação do SLC6.3 em uma máquina real, utilize seu programa favorido para gravas os arquivos ISOs em suas respectivas mídias de DVD.

A.2 Instalando o Scientific Linux CERN 6.3

Para a instalação do SLC6.3 via DVDs, será suposto que o computador tenha um hardware recente sem nenhuma parte muito fora do padrão. Sendo assim, não necessitaremos obter separadamente nenhum driver de dispositivo e o programa de instalação do Linux detectará todo o hardware sem problemas. Vou supor também que o computador tem apenas um disco rígido (HD). Além disso, como o SLC6.3 que instalaremos é para um processador de 64 bits, então o processador do computador deve ser de 64 bits. É também importante que o computador tenha driver de DVD para usarmos para as mídias DVD de instalação.

Embora a experiência de instalação descrita abaixo tenha sido feita em uma máquina virtual, vide videos do Youtube no endereço `http://bit.ly/15H1It0`, acredito que não haverá problemas num hardware real, desde que aquilo que fora dito no parágrafo anterior seja obedecido.

A instalação é bem simples e na dúvida escolha a opção ou configuração padrão oferecida pelo programa de instalação.

Para o caso dos vídeos, no qual instalamos o SLC6.3 via arquivos ISO dos DVDs, aqui são necessários as mídias físicas.

1. Inicie o computador com a mídia 1 no drive de DVD. Caso o computador não inicie pelo DVD, pode ser necessário entrar no setup do computador e mudar a configuração de como o computador inicia. É pouco provável que isso seja necessário;

2. O sistema começar a ler a "mídia" e o processo de instalação começa efetivamente;

3. O ambiente gráfico é iniciado;

4. Escolha a primeira opção de instalação: "Install or upgrade an existing system";

5. Depois do sistema ser executado, ele executará o programa de instalação. No início do programa de instalação é possível testar a integridade dos dados na mídia. Essa é uma forma alternativa àquela que fizemos via openssl e a "message digest" descrito anteriormente. Caso não queira executar os testes das mídias, selecione "Skip", caso contrário, selecione "Ok". Aqui, como fizemos os testes via openssl, não executaremos o teste de mídia;

6. A primeira tela do assistente de instalação é apenas uma identificação do sistema. Clique em "Next" para dar continuidade;

7. Escolha a língua de sua preferência e dê continuidade. Em nosso caso escolhemos "English";

8. Após a escolha da língua de instalação, vem a escolha da língua do teclado. Essa é importante para mapear adequadamente as teclas para que se possa usá-lo sem grandes dificuldades. Se seu teclado for "brasileiro" com "ç", talvez ele seja ABNT2. No caso da instalação feita para esse manual, o teclado era inglês e portanto, foi escolhido "U. S. International" já que esse tipo de mapeamento permite digitar acentos via execução de duas teclas seguidas como " ' "+ "c" para o "ç";

9. Para a escolha do sistema de arquivos, escolhemos o sistema de arquivos ordinário (Basic Storage Devices);

10. Pressione “Yes” para seguir adiante. Essa é apenas uma mensagem de advertência de que o disco será escrito. Como não temos nada no disco rígido, então podemos escrever no disco sem preocupação;

11. Na tela de configuração do “hostname”, aceitaremos a configuração padrão no momento. Essa configuração, como a maioria da configurações aqui descritas, pode ser feitas ou mudadas a posteriori. O nome da máquina (“hostname”) e o dominio onde ela se encontra (o nome depois no ponto após o “hostname”, ou seja, “domainname” ou “localdomain)”, são configurações que podem ser importantes quando temos a máquina ligada em rede com outras, o que não é o caso nesse momento;

12. Para a configuração do “Time Zone” escolheremos “São Paulo” para ajustar o relógio. Essa configuração pode ser feita/mudada após a instalação;

13. IMPORTANTE: Escolha uma senha para o usuário “root” do sistema. ANOTE para não esquecer. O usuário root tem acesso a TUDO no sistema;

14. Caso você digite como senha uma sequência de caracteres e dígitos que sejam consideradas como fraca pelo sistema, então receberá uma mensagem lhe alertando para esse fato. Caso esse sistema já seja o sistema definitivo e não de teste, sugiro repetir o processo até você ter uma senha forte aceita pelo sistema;

15. Para o tipo de instalação, escolheremos a opção padrão (“Replace Existing Linux System(s)”). A primeira opção também pode ser usada, especialmente se o disco de sua máquina for usado exclusivamente para o Linux. As outras opções devem ser usadas caso queira instalar o Linux em um disco que tenha outras partições e/ou outros sistemas operacionais, como o Windows;

16. É exibido uma mensagem de advertência de que o disco será escrito. Sem problemas, afinal você está executando essa instalaçã em uma máquina na qual só há HD(s) que pode(m) ser usado(s) pela instalação, certo?!!

17. Escolheremos o perfil Desktop para o tipo de instalação para essa máquina por acreditar que é a que deixará a máquina com o maior número de ferramentas possível para o GEANT4 sem muita coisa que não precisaríamos como banco de dados e outros servidores;

18. Abaixo, na tela de escolha do tipo de instalação, há como configurar os locais da internet que nosso sistema usará no futuro para obter programas (pacotes). Usaremos a opção default que é o site do Scientific Linux como repositório de programas. Isso será importante depois quando fizermos algumas configurações pós instalação por meio das quais instalaremos programas desse site;

19. Ao dar continuidade a procedimento acima, uma mensagem sobre checagem de dependências será exibido. Isso se dá porque as configurações que fizemos eventualmente (sempre na prática) podem precisar de outros programas para satisfazer nossas configurações;

20. Após a checagem das dependências, o sistema lhe informa sobre as mídias que serão necessárias. Como temos nossas mídias em "mãos", daremos continuidade sem reiniciar. Clique em "Continue";

21. Inicia-se o processo de cópia dos arquivos do DVD para o HD. Isso pode demorar um pouco;

22. Em algum momento, em nosso caso, aproximadamente aos 75% da cópia dos arquivos, o sistema lhe solicitará o segundo DVD. É só trocar dar continuidade;

23. Ao final desse processo de cópia dos arquivos, o sistema exibirá uma mensagem sobre a instalação do "Boot Loader", que é um programa especial responsável por carregar o sistema;

24. Ao final da instalação do "bootloader", a instalação estará completa. Então clique em "Reboot", para reiniciar o sistema para mais algumas configurações;

25. O sistema é então reiniciado e agora o sistema entra pelo Linux instalado no disco rígido. Esse é o papel do "Boot Loader" que foi configurado automaticamente pelo programa assistente de instalação para iniciar o sistema via disco rígido. Bem no inicio de sua execução, pode-se ver as possibilidades de "boot" pressionando qualquer tecla. Mas esse não é o caso. Só temos o Linux aqui. O "bootloader" está configurado para aguardar uns 3 segundos. Caso não haja nenhuma ação do usuário, ele dá continuidade a inicialização do sistema pelo dispositivo padrão, que nesse caso é o disco rígido;

26. Ao final dessa primeira inicialização do sistema, um programa é acionado automaticamente para que possamos fazer algumas configurações finais. Uma tela de boas vindas. Clique em "Forward";

27. Devemos criar ao menos um usuário. Vamos criar um usuário chamado "geant4" com o qual trabalharemos com o GEANT4. Então, digite geant4 no campo "Username". No campo "Full Name" deve-se digitar algo significativo à esse usuário, aqui usaremos o termo "Geant4 by CERN". Então, digite uma senha. No nosso caso digitamos "geant", que é uma senha fraca e portanto o sistema exibe uma advertência ao clicarmos em "Forward";

28. Depois de criado o usuário, o sistema lhe oferece a oportunidade de ajustar data e hora;

29. Se o programa "kdump" exibir uma mensagem de erro dizendo que não pode ser configurado por falta de memória, não se preocupe e vá em frente. Isso não deverá ser um problema para nossos propósitos;

30. Depois disso, o sistema reinicia e então exibe a tela de "login" pela qual pode-se entrar no sistema com o nome de usuário e senha cadastrados anteriormente, no nosso caso, "geant4" para o login e "geant" para a senha;

31. Para sair do sistema após logado, vá para o menu da barra superior e clique em "System" e então em "Shutdown". Uma tela de diálogo será exibida pela qual pode-se escolher opções tais como: "Hibernate", "Restart" e "Shutdown";

A.3 Algumas Configurações do Scientific Linux CERN 6.3 Pós Instalação

Antes de iniciarmos o processo de instalação do GEANT4 propriamente dito, é importantes fazer algumas configurações no sistema e verificar e instalar alguns programas que são pré-requisitos para o GEANT4. Alguns deles são o driver OpenGL que auxilia na visualização da simulação e é muito útil para o "debugging" do programa de simulação, o programa cmake, que a partir da versão 9.5 se faz necessário, necessário para a configuração da compilação dos programas de simulação implementados pelo usuário e para

a configuração da compilação do próprio GEANT4. É necessário também o programa compilador `C++` e suas bibliotecas. No nosso caso usaremos como compilador o "gcc".

Para realizarmos uma série de tarefas, será necessário dar privilégios administrativos ao usuário geant4 criado previamente. A forma default com a qual criamos o usuário geant4 não muitos privilégios a ele, por motivos de segurança. Sem alguns privilégios administrativos, não seria possível realizar as configurações e instalações de alguns programas que serão necessários para o GEANT4. Assim, os seguintes passos são sugeridos para dar o usuário geant4 privilégios para as tarefas seguintes:

1. Vá até o ícone "Terminal" na barra de menu na parte superior da tela e clique nele para abrir uma janela de terminal, ou também chamada de console;

2. Para fazermos as alterações dos privilégios do usuário geant4, precisamos fazê-las como usuário que tenha tais privilégios. Como só há um usuário, que é o próprio geant4 e o usuário "master", que é o usuário root, então utilizaremos o usuário root. Root é o usuário criado pelo próprio sistema para o qual demos uma senha no momento da instalação so sistema operacional, lembra?!!!

   Para entrarmos como "root" é só digitar o comando "`su`" seguido de "enter". Será solicitada a senha do root. Digite-a e pressione "enter". Veja que o "promp" muda de "`[geant4@localhost ~]$`" para "`[root@localhost geant4]#`". O símbolo de "`#`" é usualmente usado para designar que se está em modo "super user" e o símbolo "`$`" usualmente sinaliza um usuário ordinário. O símbolo "~" no primeiro prompt significa que se está no diretório "home" do usuário, que nesse caso significa "`/home/geant4`". Quando se muda para o usuário root, fica indicado no segundo prompt que se está no diretório "`geant4`". O comando "`pwd`" lhe mostra o caminho completo;

3. O arquivo que temos que mudar chama-se "sudoers" e encontra-se no diretório "`/etc`". Portanto, devemos digitar "`cd /etc`" seguido de "enter" para mudar de diretório. "`cd`" significa "change directory".

4. Uma vez no diretório onde encontra-se o arquivo sudoers, podemos executar o comando "`ls`" (de list) para ver o conteúdo do diretório. Nesse caso, pode-se também executar "`ls -l sudoers`" que diz para listar em detalhes os arquivos

com nome `sudoers`, ou seja, listá-lo se ele existir. Dessa forma pode-se visualizar quem é o dono ("owner") do arquivo e vemos que é o root. Vemos também que ele está configurado com permissões de somente leitura. Isso pode ser visto pela cadeia de caracteres no início da entrada lista com o comando "`ls -l sudoers`". Essa cadeia de caracteres é "`-r--r-----`". O primeiro caractere refere-se ao tipo de arquivo. Se for "`d`" trata-se de um diretório, mas se for um "`-`" trata-se de uma arquivo comum. Os próximos 3 caracteres dizem respeito as permissões que o próprio usuário dono tem sendo `rwx` para leitura, escrita e execução. Os próximos 3 são as permissões que um determinado grupo de usuário pode ser sobre esse arquivo e os últimos 3 caracteres são as permissões de qualquer um ou "all" ou "Other" no jargão Linux.

Para mudar a permissão de somente leitura para que possamos alterá-lo, executamos o seguinte comando: "`chmod u+w sudoers`" que mudará ("`+`") o atributo de escrita de "`-`" para "`w`" do usuário "`u`" do arquivo "`sudoers`". A efetivação da mudança pode ser verificada com um novo comando "`ls -l sudoers`".

5. Para editar usaremos o programa nano que é um programa de console, mas pode-se utilizar o `gedit` que é um programa para o ambiente gráfico. Para usar o nano contra o arquivo a ser editado executamos o comando "`nano sudoers`". Vá até a linha contendo "`root ALL=(ALL) ALL`". Digite a mesma configuração para o usuário geant4, ou seja, adiciona uma linha abaixo com o seguinte conteúdo: "`geant4 ALL=(ALL) (ALL)`". Caso queira alguma configuração de segurança mais adequada, vide manual do sudoers para refinar essa configuração. Salve a alteração com as teclas "`CTRL+O`" e então "enter". Para sair do nano, digite "`CTRL-X`";

6. Retorne as configurações de acesso ao arquivo sudoers para o original: "`chmod u-w sudoers`";

7. Pode-se sair do modo super usuário digitando exit e então enter;

8. Agora, instalaremos o compilador `C++` para Linux chamado `gcc`. Tente verificar se ele está instalado tentando executar o comando "`gcc`" ou "`gcc -v`". A mensagem "`bash: gcc: command not found`" será exibida. Utilizaremos o pro-

grama "yum" para instalar o gcc. Primeiro executaremos o comando "yum info gcc" para checar informações a respeito da instalação do gcc nesse sistema. Esse comando retornar uma série de informações, mas umas delas na segunda linha é "Available Packages" mostrando que o gcc está disponível, mas não está instalado. Podemos instalá-lo via programa gráfico gerenciador de pacotes. Clique no menu "System" da barra de menu na parte superior da tela e então clique em "Administration" e por último clique em "Add/Remove Software". Digite gcc na caixa de pesquisa. Aparecerá uma lista de programas relacionados ao compilador C++ GNU à direita da janela do programa. Selecione os seguintes programas para serem instalados: "C++ support for gcc" e "Various compilers (C, C++, Objective-C, Java, ...)". Não será possível selecionar o pacote "GCC version 4.4 shared support library" pois já está instalado no sistema operacional, assim como "GNU Standard C++ Library". Clique em "Apply" para executar a instalação dos programas selecionados. O gerenciador de pacotes poderá instalar outros programas que possam ser pré-requisitos. O gerenciador lhe mostrará esses programas e lhe pedirá permissão para prosseguir. É necessário a senha do usuário root. Veja na janela do programa do lado inferior esquerdo que há uma mensagem mostrando que o gerenciador está fazendo a obtenção dos programas na internet. Pode ser que vc seja perguntado pelo programa instalador se você confia ou não na fonte. Clique "Yes".

Após a instalação do gcc, pode-se verificar o novo status desse pacote por digitar "yum info gcc" na linha de comando e ver que agora ele está como "Installed Package". Pode-se verificar também agora a versão do gcc instalada no sistema digitando-se "gcc --version";

9. Para instalar o programa cmake, comecemos por verificar, via linha de comando, algumas informações com o comando "yum info cmake". Veja que é um pacote "Available", mas não está instalado. O instalaremos via console, mas pode-se também instalado via gerenciador gráfico de pacotes conforme foi feito para o gcc. Para proceder a instalação digite "yum install cmake", mas com o comando "sudo" antes do comando principal. Isso é necessário pois para instalar programas no sistema é necessário privilégios de administrador. Embora o geant4 seja um usuário com privilégios de administrador, por segurança devemos informar ao sistema quando o usuário efetivamente usará esses privilégios. Para isso, usamos

o comando "`sudo`" antes do comando a ser executado com privilégios especiais. Assim o comando de instalação do `cmake` é: "`sudo yum install cmake`". A senha do usuário geant4 será perguntada. Nesse caso, o programa de instalação lhe mostrará o que deve ser baixado da internet e o seu respectivo tamanho e então he perguntará se você quer realmente instalá-lo. Responda de acordo. Após a instalação pode-se testar o programa por digitar "`cmake`" ou "`cmake -v`". Informações sobre a instalação do cmake também podem ser obtidas via "`yum info cmake`";

10. Por último temos que verificar e/ou instalar o driver `opengl`. Para testar podemos executar o comando "`yum info opengl`", ou melhor ainda o comando "`yum info opengl*`". O caractere "`*`" siginifica que queremos informação sobre qualquer pacote que se inicie com "`opengl`". No caso desse sistema, o driver `opengl` é provida via implementação mesa e por isso o nome do driver começa com "`mesa`". Para progcurarmos por algum programa (ou driver) que se inicie com "`mesa`" executaremos o seguinte comando: "`yum info mesa*`". Ao procedermos dessa forma, teremos como resultado uma série de programas (pacotes) relacionado com a implementação mesa para o driver openGL sob a forma de pacote disponível, mas não instalado.

    Outra forma de se listar os programas instalado é via comando "`yum list`" que lista todos. O comando "`yum list | grep -R "opengl|mesa"`", nesse caso, inclua as aspas antes e depois de `opendl|mesa`. O primeiro caractere "`|`" indica que a saida do comando "`yum list`" seja direcionado para a entrada do comando "`grep`". Isso seja chama "pipeline"e é ferramenta muito importante e poderosa dos sistemas tipo Unix como o Linux. Como o primeiro comando lista tudo, passamos toda a lista para o programa grep que com seus argumento "`-E`" e "`opengl|mesa`" filtra e só mostra as linhas que possuem ou os caracteres "`opengl`" ou os caracteres "`mesa`". Esse "ou" é implementado pelo caractere "`|`" entre as duas palavras. Como mostram os comandos, o driver openGL encontra-se instalado.

Apêndice B

Instalação do GEANT4 9.6

Obtenção, Configuração, Compilação, Instalação e Teste do GEANT4 9.6

B.1 Obtenção e Configuração do GEANT4 9.6 para a Compilação

1. Vá ao sítio do GEANT4 na internet, `http://geant4.cern.ch`, e clique em download no canto superior direito da página. Na sessão "Source Files" clique no ícone para baixar a versão GNU. Existe no canto superior direito da página uma sessão chamada "Related Links" onde há uma ligação chamada "Previous Realeases of Geant4" por meio da qual pode-se obter versões mais antigas do GEANT4. Salve o arquivo no "Desktop" do usuário geant4. Usaremos o diretório Desktop pois fica mais fácil acesso e visualização do que está acontecendo durante nossas ações. O arquivo baixado chama-se "`geant4.9.6.tar.gz`". Pode ocorrer do nome mudar numa data futura como por exemplo para "`geant4.9.6.p01.tar.gz`", caso seja lançado algum "patch" ao programa, o que é comum;

2. Para descompactar o arquivo recém baixado, pode-se proceder o comando "`tar zxvf geant4.9.6.tar.gz`" via console ou um duplo clique no arquivo o abrirá com o gerenciador de arquivos exibindo o seu conteúdo. Para extrair o conteúdo é só clicar no botão "Extract" no canto inferior direito da janela do programa gerenciador de arquivos. A extração se dará no mesmo diretório onde se encontra

o arquivo compactado, nesse caso, Desktop (`/home/geant4/Desktop` é o caminho completo) e a pasta tem o nome de "`geant4.9.6`";

3. Abra um terminal (console) para os procedimentos iniciais de configuração para compilação do GEANT4. Mude para o diretório onde se encontra a pasta descompactada do GEANT4, que é `/home/geant4/Desktop`. Ao abrir o terminal, o prompt nos põe automaticamente no diretório raiz do usuário com o qual estamos logado, logo `/home/geant4`. Para ir para a pasta Desktop, é só executar o comando "`cd ./Desktop`". Aqui o "`.`" É o caminho para o diretório atual e é o mesmo que "`/home/geant4`" nesse caso. Portanto, esse comando diz para trocar de diretório e entrar no diretório "`Desktop`" que existe dentro do diretório atual "`/home/geant4`". Não confundir com o diretório "∼" que significa o diretório "`home`" do usuário, que nesse caso é também "`/home/geant4`". Em caso de dúvida em qual diretório você s e encontra, execute o comando "`pwd`" que ele lhe mostra o diretório atual;

4. Estando agora no diretório `/home/geant4/Desktop`, crie uma pasta na qual será configurado a instalação do GEANT4. Essa pasta pode ter qualquer nome, mas nesse caso a chamaremos de "`geant4.9.6-build`". Para criá-la, execute o comando "`mkdir geant4.9.6-build`". Entre na pasta com o comando "`cd ./geant4.9.6-build`" (lembre-se, o "`.`" é o caminho para o diretório atual, que nesse caso é "`/home/geant4/Desktop`").

5. Para configurar a compilação, usaremos o programa `cmake` com uma série de argumentos como segue:

```
cmake -DCMAKE_INSTAL_PREFIX=/usr/local \
-DGEANT4_INSTALL_DATA=ON \
-DGEANT4_USE_OPENGL_X11=ON \
-DGEANT4_INSTALL_EXAMPLES=ON \
-DBUILD_SHARED_LIBS=ON \
-DBUILD_STATIC_LIBS=ON \
/home/geant4/Desktop/geant4.9.6
```

Aqui o comando foi desmembrado em linhas separadas. Para informar ao SO que as diferentes linhas fazem parte de um único comando, colaca-se a barra invertida

"\" ao final de cada linha. Para saber sobre todas as opções do `cmake`, consulte manual. Aqui descrevemos algumas dessas opções:

`DCMAKE_INSTAL_PREFIX`: é o diretório destino da instalação, ou seja, onde serão postos os arquivos resultantes do processo de compilação e após ser executado o comando "`make install`" mais adiante.

`DGEANT4_INSTALL_DATA`: se "`ON`", o processo de compilação fará o download dos arquivos de dados necessários. Nas versões do GEANT4 anteriores a 9.5, era necessário fazer a obtenção dos arquivos de dados separadamente e manualmente.

`DGEANT4_USE_OPENGL_X11`: se "`ON`", significa que a compilação do GEANT4 será feita considerando o driver openGL instalado. Se o driver não existir, ocorrerá um erro. Se existir e essa opção for "`OFF`", então o GEANT4 não será capaz de usar o driver para apresentar certos resultados da simulação na forma gráfica utilizando o openGL.

`DGEANT4_INSTALL_EXAMPLES`: se "`ON`", os programas de exemplo serão baixados da internet. Isso é particularmente interessante para quem está iniciando.

`DBUILD_SHARED_LIBS`: se "`ON`", a compilação criará as bibliotecas dinâmica ou compartilhadas. No caso do Linux, são arquivos com extensão "`.so`" nos quais se encerram funções que podem ser importadas dinamicamente em tempo de execução por programas escritos pelo usuário.

`DBUILD_STATIC_LIBS`: se "`ON`", a compilação criará as bibliotecas estáticas, que no Linux são arquivos com a extensão "`.a`". Essas bibliotecas, como as dinâmicas acima, são arquivos que encapsulam programas ou funções do GEANT4 e que podemos utilizar para importar ("`#include`") em nossos programas de simulação. A diferenças das bibliotecas estáticas com relação as dinâmicas é que essas são importadas para nossos programas em tempo de compilação e "colocadas" dentro do programa executável criados por nós. Isso dá um programa executável em um

arquivo maior, mas tem a vantagem de se poder executar o programa em outra máquina, mesmo que a outra máquina não tenha o GEANT4.

Por último, vem o diretório onde se encontra o código que se deseja a compilação, no nosso caso "`/home/geant4/Desktop/geant4.9.6`", onde estão os arquivos fontes do GEANT4 versão 9.6.

6. Durante a tentativa de executar o comando do item anterior, ocorreu um erro e foi preciso executar o seguinte comando para instalar um programa que estava faltando (o "expat-devel"): "`sudo yum install expat*`";

7. Foi também necessário instalar ou reinstalar o driver mesa (openGL) com o seguinte comando: "`sudo yum install mesa*`";

8. Depois dessas duas instalações de programas acima, o procedimento de configuração para compilação deve ser executado novamente;

B.2 Compilação do GEANT4 9.6

Para compilar o GEANT4 após ter executado o procedimento de configuração para compilação com o `cmake` mostrado anteriormente, siga os seguintes passos:

1. Para compilar o GEANT4, abra uma tela de terminal e vá ao diretório onde o GEANT4 foi configurado para a compilação, ou seja, no nosso caso: "`cd /home/geant4/Desktop/geant4.9.6-build`", ou "`cd ~/Desktop/geant4.9.6-build`";

2. Para compilar execute o comando: "`make -j2 VERBOSE=1`", onde o argumento "`-j2`" informa ao compilador que é para compilar para um sistema com dois processadores. Se no seu caso houver apenas um processador, use "`-j1`" ou apenas não informe nada, pois "`-j1`" é o default. A opção "`VERBOSE`" está relacionada a quantidade de informações mostradas no processo de compilação, quanto maior o número, maior a quantidade de informação mostrada.
   A compilação pode demorar. No caso realizado para a construção desse manual, a compilação demorou cerca de uma hora;

B.3 Instalação do GEANT4 9.6

O processo de instalação do GEANT4 é praticamente um roteiro de cópia de alguns arquivos que estão no diretório "`~/Desktop/geant4.9.6-build`", especialmente aqueles resultantes da compilação. Em especial, os diretórios "`/usr/local/include`"e "`/usr/local/lib64`"são destinos dos arquivos de include "`.h` e `.hh`" e dos arquivos das bibliotecas ("`.a` e `.so`") respectivamente. Vários arquivos também são postos em "`/usr/local/share/Geant4-9.6.0`", em particular, os arquivos de dados no diretório "`data`" e os exemplos no diretório "`examples`". Adicionalmente, o diretório "`/usr/local/bin`" também será destino de alguns arquivos interessantes.

1. Para compilar o GEANT4, abra uma tela de terminal e vá ao diretório onde o GEANT4 foi configurado para a compilação, ou seja, no nosso caso: "`cd /home/geant4/Desktop/geant4.9.6`", ou "`cd ~/Desktop/geant4.9.6-build`";

2. O comando que deve ser executado para a instalação é: "`sudo make install`". Repare que aqui o `sudo` é importante, já que deverá ser acessado para escrita alguns diretórios que são do sistema e o GEANT4 não tem acesso direto à eles.

B.4 Teste do GEANT4 9.6

Faremos um teste da instalação do GEANT4 9.6 por executar o programa B3 de exemplo contido em sua instalação.

1. Use o programa File Browser e vá ao diretório "`/usr/local/share/Geant4-9.6.0/examples`". Copie a pasta `B3` que está dentro da pasta "`basic`" para o `Desktop`;

2. Abra um terminal e vá para o `Desktop` dessa forma: "`cd ~/Desktop`". Crie uma pasta dessa forma: "`mkdir B3-build`". Entre na pasta: "`cd ./B3-build`". De dentro desse diretório faremos a configuração para a compilação do programa de exemplo que está na pasta "`~/Desktop/B3`";

3. Como dissemos anteriormente, o diretório "`/usr/local/bin`" são postos arquivos interessantes pelo processo de instalação do GEANT4. Um é particularmente útil e chama-se "`geant4.sh`". Ele é um "*script*" que configura várias variáveis

de ambiente para o processo de configuração e compilação de programas escritos por nós para uso do GEANT4. As variáveis de ambiente do GEANT4 tem seus nomes iniciados por `G4`. Como até o momento nenhuma variável de ambiente foi configurada. Isso quer disser que se executarmos o comando "`env | grep G4`" nada deve ser mostrado na tela. O comando "`env`" mostra todas as variáveis de ambiente configuradas no sistema. O comando "`env | grep G4`" faz toda a saida do comando "`env`" ser enviada para a entrada do programa "`grep`" que filtra (mostra) somente o que tem os caracteres `G4`. Execute o comando "`source /usr/local/bin/geant4.sh`", para configurar as variáveis de ambiente. Depois de configuradas, o comando "`env | grep G4`" mostrará várias variáveis do GEANT4 configuradas;

4. Para configurarmos a compilação execute o comando:
   `cmake -DGeant4_DIR=/usr/local/share/Gean4-9.6 /home/geant4/Desktop/B3`. Aqui o argumento "`-DGeant4_DIR`" deve apontar para o dietório onde está instalado o GEANT4 9.6 e o último argumento é o caminho completo para o diretório onde se encontra o programa a ser configurado;

5. Para compilá-lo, execute: "`make -j2`", onde "`-j2`" é para o caso de seu processador ser duplo ou haver dois processadores em sua máquina;

6. Para executar o programa de exemplo compilado, execute: "`./exampleB3`". O programa executará e no console será exibido o prompt do programa. Para ver alguma simulação na janela gráfica, execute o seguinte comando: "`/run/beamOn 100`" e `100` eventos serão simulados;

Apêndice C

Instalação da CLHEP 2.0.4.7

Para a instalação da CLHEP 2.0.4.7 deve-se proceder da seguinte maneira:

1. Descarregar e descompactar o arquivo clhep-2.0.4.7.tgz através do comando "`tar zxvf clhep-2.0.4.7.tgz`". Isso criará uma pasta de nome 2.0.4.7 no diretório local de onde o comando fora executado. Essa pasta contém todo o conteúdo descompactado do arquivo baixado.

2. Entrar no diretório com o comando "`cd ./2.0.4.7`" e então dar o comando "`cd ./CLHEP`" para entrar na pasta CLHEP para, a partir de lá, começar o processo de configuração, compilação e instalação.

3. Para configurar a CLHEP para a instalação deve-se dar o comando "`./configure --prefix /Applications/CLHEP_2.0.4.7`" de dentro da pasta "`2.0.4.7/CLHEP`". O argumento de "`--prefix`"é o diretório destino de instalação da CLHEP e é de escolha do usuário. Esse argumento deve ser um caminho absoluto, ou seja, deve ser um caminho a partir do diretório raiz que em sistemas Linux é o "`/`" e no Windows é "`c:\`". Caso não seja utilizada a opção "`--prefix`", a instalação se dará no diretório default que é "`/usr/local`". A vantagem de indicar um local de instalação personalizado é de ter múltiplas versões da CLHEP instaladas.

A compilação é feita simplesmente pela execução do programa "`make`" do pacote gcc. Esse procedimento pode demorar alguns minutos. A instalação ocorre de forma similar,

ou seja, basta executar o comando make com install como argumento: "`make install`". Após esse procedimento, deverá existir os diretórios "`lib`" e "`include`" dentro de "`/Applications/CLHEP_2.0.4.7`"

Apêndice D

Instalação do GEANT4.9.3.p02

1. Estando no diretório "`/Users/you`" e tendo descarregado o arquivo "`geant4.9.3.p02.tar.gz`" para o mesmo, proceder à descompactação por meio de "`tar -zxvf geant4.9.0.tar.gz`". Isso cria a pasta "`geant4.9.3.p02`" em "`/Users/you`".

2. Entrar no diretório recém criado "`cd ./geant4.9.3.p02`" e criar uma pasta chamada "`data`" dentro dele com o comando "`mkdir data`".

3. Entrar nesse diretório ("`cd data`") e copiar todos os arquivos de dados descarregados do sítio do GEANT4 para dentro dessa pasta.

4. Descompactar os arquivos dentro dessa mesma pasta. Isso pode ser feito da seguinte forma: "`tar zxvf G4EMLOW.4.3.tar.gz`", utilizando o arquivo de dado "`G4EMLOW.4.3.tar.gz`" como exemplo.

5. Para a compilação do GEANT4, deveremos estar na pasta "`/Users/you/geant4.9.3.p02`". De dentro desse diretório digite "`./Configure -build`". Lembrar sempre que os sistemas tipo Unix diferenciam letras maiúsculas de minúsculas. O "`C`" nesse caso é maiúsculo. Após o comando, uma série de perguntas deve ser respondida para configurar a compilação adequadamente. É crucial que as respostas sejam dadas de forma correta ou, caso contrário, a compilação não terá sucesso ou o GEANT4 poderá exibir problemas.

6. Após executar o comando, a primeira parada é informativa. Simplesmente deve-se digitar "enter" para continuar. A segunda parada é também informativa: digitar "enter" para continuar. Idem para a terceira parada. Então, tem-se a

primeira pergunta. O script de configuração detecta automaticamente o sistema e o compilador, nesse caso "`Darwin`" (MacOSX) e "`g++`" (`gcc/g++`). Tem-se a opção de modificar caso o script tenha obtido essas informações incorretamente. Como nesse caso está correto, é só digitar "enter" para continuar. A segunda pergunta é sobre rodar uma máquina apenas ou múltiplas máquinas. Aceite o valor default "`[n]`" digitando "enter" para dizer ao script que os programas serão rodados em apenas uma máquina.

7. A terceira pergunta é sobre a localização dos arquivos fontes do GEANT4. Se ele apontar corretamente para "`/Users/you/geant4.9.3.p02`", então basta digitar "enter" para continuar.

8. A quarta pergunta: "Onde o GEANT4 deverá ser instalado?". A sugestão do script é "`/Users/you/geant4.9.3.p02`". Pode-se configurar outro destino, mas nesse caso será acatada a sugestão.

9. Quinta pergunta: "Você quer instalar todos os cabeçalhos do GEANT4 em um único diretório?". Nesse caso, será aceito o valor default, ou seja, não. Porém, talvez isso possa ser interessante ao se configurar uma IDE de desenvolvimento como o Eclipse.

10. Sexta pergunta: "Você quer compilar/construir bibliotecas compartilhadas?". Aceitar o default "`[y]`".

11. Sétima pergunta: "Você quer compilar bibliotecas estáticas também?". Aceitar o default "`[n]`".

12. Oitava pergunta: "Você quer compilar bibliotecas globais?". Aceitar o default "`[y]`".

13. Nona pergunta: "Você quer compilar bibliotecas granulares também?". Aceitar o default "`[n]`".

14. Décima pergunta: "Você quer compilar as bibliotecas com informações para debugging?". Aceitar o default "`[n]`".

15. 11a. Pergunta: Refere-se ao local onde estão os dados. Como os dados foram colocados em "`/Users/you/geant4.9.3.p02/data`", então basta continuar. Caso

o script não encontre alguns diretórios de dados, isso pode ocorrer devido a uma versão diferente dos dados como, por exemplo, ter "`PhotonEvaporation2.2`" e o script espera por "`PhotonEvaporation2.0`". Solucionar isso baixando a versão correta dos arquivos de dados.

16. 12a. Pergunta: relacionada à definição do caminho para a biblioteca CLHEP. Nesse caso, como a instalação foi realizada em "`/Applications/CLHEP_2.0.4.7`", então esse é o caminho que deverá ser informado ao script de configuração.

17. 13a. Pergunta: se todo o procedimento estiver correto, o script deve exibir os caminhos completos dos diretórios "`lib`" e "`include`". Basta continuar.

18. 14a. Pergunta: relativa a habilitar ou não os módulos de UI (interface do usuário). Aceitar o valor default "`[y]`".

19. Para as perguntas sobre a interface do usuário, responder usando os valores default:

```
Enable building of User Interface (UI) modules?  [y]
Enable building of the XAW (X11 Athena Widget set) UI module?  [n]
Enable building of the X11-Motif (Xm) UI module?  [n]
Enable building of the Qt UI module?  [n]
```

20. Para as perguntas relacionadas às opções de drivers de visualização a sugestão é habilitar o driver OpenGL e desabilitar os outros:

```
Enable building of visualization drivers?  [y]
Enable building of the X11 OpenGL visualization driver?  [n] y
Enable building of the X11-Motif OpenGL visualization driver?  [n]
Enable building of the FukuiRenderer/DAWN visualization driver?  [n]
Enable building of the X11 OpenInventor visualization driver?  [n]
Enable building of the X11 RayTracer visualization driver?  [n]
Enable building of the VRML visualization driver?  [n]
```

21. Para as opções restantes (módulos de extensão opcionais do GEANT4) a sugestão é responder "não" para todas:

```
Enable the Geometry Description Markup Language (GDML) module?  [n]
```

```
Enable build of the g3tog4 utility module?  [n]
Enable internal zlib compression for HepRep visualization?  [n]
```

Assim, termina a parte de configuração. Então, é exibida uma informação de onde se encontra o *script* de configuração no qual estão registradas todas as informações que foram dadas ao *script* de construção do *script* de configuração. O *script* de configuração deve estar em:

`/Users/you/geant4.9.3.p02/.config/bin/Darwin-g++/config.sh`. Antes da compilação começar, pode-se editar esse arquivo para alguma configuração.

Digitar "enter" para iniciar a compilação e instalação do GEANT4 9.3 patch 02. A compilação pode demorar vários minutos.

Executar o *script* "`./Configure`" para criar o *script* de *setup* que ajusta várias variáveis de ambiente para o GEANT4. Depois desse comando, deverá aparecer o arquivo "`env.sh`" no diretório local "`/Users/you/geant4.9.3.p02`". Para usar esse *script* e ajustar as variáveis de ambiente basta executar "`source ./env.sh`".

É importante ajustar a variável de ambiente que informa ao GEANT4 qual é o diretório de trabalho onde são colocados os programas executáveis de simulação após sua respectiva compilação. Executar o comando "`export G4WORKDIR=/Users/you/geant4`". Dessa forma, o GEANT4 colocará os programas compilados em "`/Users/you/geant4/bin/Darwin-g++`"

E assim termina o processo de instalação do GEANT4 versão 9.3 patch 02 no sistema operacional MacOS X 10.6 (Snow Leopard).

Para que os futuros usuários tenham um metodologia de instalação da versão mais recente do GEANT4, versão 9.6 à época da elaboração desse documento, foi elaborado também uma metodologia para essa versão. Para essa versão foi utilizada o sistema operacional Scientific Linux CERN 6.3, a última versão desse S.O. à época da elaboração desse documento. Essa metodologia foi também feita em vídeos que estão disponi-

bilizados na rede mundial de computadores no endereço `http://www.youtube.com/playlist?list=PLKorLi2cUF3vOYrUBOa4_aWNDsy4c6TuS` ou `http://bit.ly/15H1It0`, além do texto que pode ser visto nos apêndices.

O apêndice A apresenta detalhes sobre a instalação do Scientific Linux CERN 6.3 e o apêndice B contém informações sobre a instalação do GEANT4 9.6.