ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ

ЛЕКЦИЯ №13

«Статистическое моделирование в пакете Matlab. Решение дифференциальных уравнений в частных производных в Matlab»

Отделение ядерно-топливного цикла

Лектор:
Зав. каф. - руководитель ОЯТЦ ИЯТШ
Горюнов А.Г.

2020

План лекции

13.1. Статистическое моделирование в Matlab

13.2. Решение дифференциальных уравнений в частных производных в Matlab.

13.3. Программные комплексы статистического моделирования

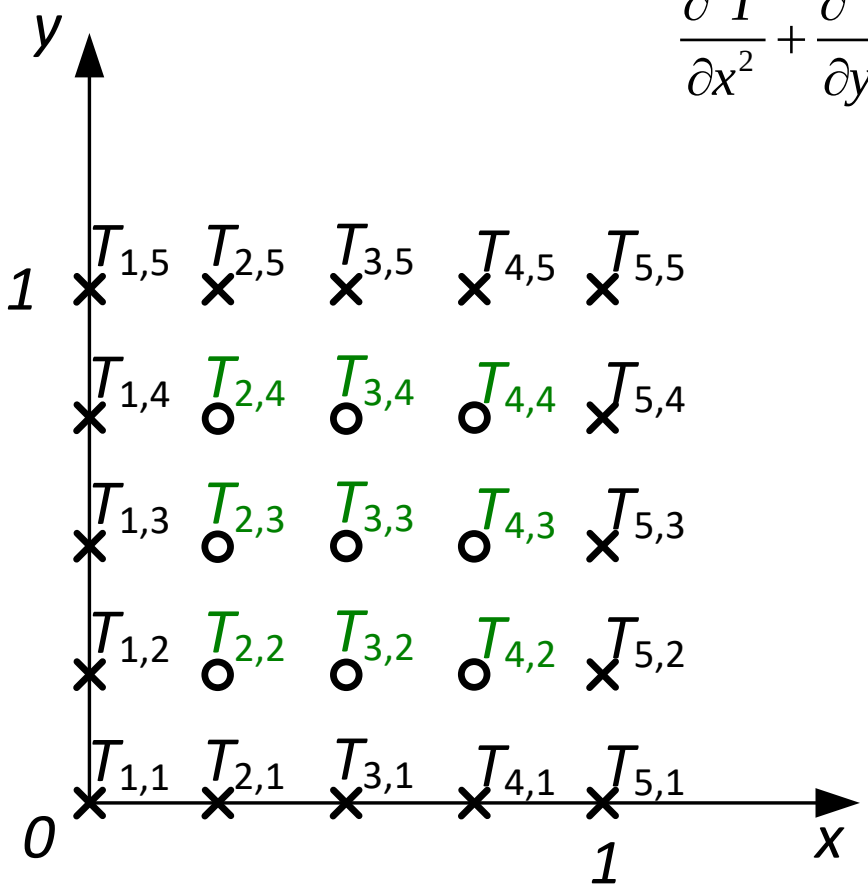
Информация по курсу:

https://portal.tpu.ru/SHARED/a/ALEX1479/study/Matmod/Tab

13.1 Статистическое моделирование в Matlab

13.1.1 Пример решения уравнения теплопроводности методом Монте-Карло

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \qquad (13.1)$$



$$T_{1,1} = T_{1,2} = T_{1,3} = T_{1,4} = T_{1,5} = 0$$

$$T_{5,1} = T_{5,2} = T_{5,3} = T_{5,4} = T_{5,5} = 100$$

$$T_{2,1} = 25;\ T_{3,1} = 50;\ T_{4,1} = 75$$

$$T_{2,5} = 6.25;\ T_{3,5} = 25;\ T_{4,5} = 56.25$$

$$h = h_x = h_y = 0.25$$

13.1.1 Пример решения уравнения теплопроводности методом Монте-Карло

$$T_{i,j} = P_{x+} \cdot T_{i+1,j} + P_{x-} \cdot T_{i-1,j} + P_{y+} \cdot T_{i,j+1} + P_{y-} \cdot T_{i,j-1},$$
$$P_{x+} = P_{x-} = P_{y+} = P_{y-} = \frac{1}{4}. \qquad (13.2)$$

Чтобы начать вычисление температуры в точке $T_{i,j}$, в этой точке приводится в движение случайно блуждающая частица. После этого частица начинает блуждать от одной узловой точки к другой. Когда частица достигает узловой точки на границе, блуждание прекращается и записывается известная в этой граничной точке температура.

Обозначим температуру в конце первого блуждания Tw_1. Затем из точки $T_{i,j}$ выпускается вторая, третья, …, N -я частицы и записываются соответствующие температуры в конечных точках блуждания Tw_2, Tw_3, …, Tw_N.

Температура внутренней точки определяется осреднением N температур граничных точек, достигнутых беспорядочно блуждающими частицами.

Следовательно, решение для температуры $T_{i,j}$ по методу Монте-Карло выражается в виде:

$$T_{i,j} = \frac{1}{N} \sum_{j=1}^{N} Tw_j \qquad (13.3)$$

13.1.1 Пример решения уравнения теплопроводности методом Монте-Карло


```
clc;
T = zeros(5,5);

T(1,1) = 0; T(1,2) = 0; T(1,3) = 0; T(1,4) = 0; T(1,5) = 0;
T(5,1) = 100; T(5,2) = 100; T(5,3) = 100; T(5,4) = 100; T(5,5) = 100;
T(2,1) = 25; T(3,1) = 50; T(4,1) = 75;
T(2,5)= 6.25; T(3,5) = 25; T(4,5) = 56.25;


imin = 2; imax = 4; jmin = 2; jmax = 4;
N = 1000;
Tw = []; stop = 0;
```

13.1.1 Пример решения уравнения теплопроводности методом Монте-Карло


```
for i=2:4
    for j=2:4
        k = 1;
        while (k < N)
            ii = i; jj = j;stop = 0;
            while (stop == 0)
                direction = randi(4);
                switch direction
                    case 1
                        ii = ii + 1;
                    case 2
                        ii = ii - 1;
                    case 3
                        jj = jj + 1;
                    case 4
                        jj = jj - 1;
                end
                if(ii < imin)
                    Tw(k) = T(imin-1,jj);
                    k = k + 1;
                    stop = 1;
                end
                if(ii > imax)
                    Tw(k) = T(imax+1,jj);
                    k = k + 1;
                    stop = 1;
                end
                if(jj < jmin)
                    Tw(k) = T(ii,jmin-1);
                    k = k + 1;
                    stop = 1;
                end
                if(jj > jmax)
                    Tw(k) = T(ii,jmax+1);
                    k = k + 1;
                    stop = 1;
                end
            end
        end
```

Command Window

```
ans =

         0   25.0000   50.0000   75.0000  100.0000
         0   22.7312   48.5688   72.3063  100.0000
         0   20.5625   44.3438   70.6813  100.0000
         0   16.9375   38.9125   66.1250  100.0000
         0    6.2500   25.0000   56.2500  100.0000
```

13.1.2 Моделирование процесса прохождения нейтронов через пластину

Необходимо вычислить:

1) вероятность прохождения нейтрона через пластину p^+ (случай а);
2) вероятность поглощения нейтрона в пластине p^0 (случай б);
3) вероятность отражения нейтрона пластиной p^- (случай в).

Взаимодействие нейтронов с веществом описывается двумя сечениями: Σ_s и Σ_c. Их сумма – полное сечение взаимодействия: $\Sigma = \Sigma_s + \Sigma_c$.

Таким образом, при столкновении нейтрона с атомом вероятность поглощения равна Σ_c/Σ, а вероятность рассеяния – Σ_s/Σ.

Для разыгрывания длины свободного пробега воспользуемся выражением:

$$\lambda = -\frac{1}{\Sigma}\ln\gamma, \qquad (13.4)$$

где γ - случайная величина с равномерным распределением.

Остается выбирать случайное направление нейтрона после рассеяния. Так как задача симметрична относительно *x*, то направление определяется одним углом ϕ между скоростью и осью *OX*. Требование равной вероятности любого направления равносильно требованию, чтобы косинус этого угла $\mu = \cos\phi$ был равномерно распределен в интервале (-1,1). Для равномерно распределенных величин на отрезке [-1,1] правило розыгрыша: $\gamma=(\mu-(-1))/(1-(-1))$, следовательно, $\mu = 2\gamma-1$.

Схема алгоритма

Пусть нейтрон испытал k-ое рассеяние внутри пластины в точке с абсциссой x_k и после этого начал двигаться в направлении μ_k.

Разыграем длину свободного пробега:

$$\lambda_k = -\frac{1}{\Sigma}\ln\gamma$$

Вычислим абсциссу следующего столкновения:

$$x_{k+1} = x_k + \lambda_k \mu_k$$

Проверим условие прохождения через пластинку: $x_{k+1} > h$?

Если это условие выполнено, то траектория заканчивается и добавляется 1 к счетчику прошедших частиц N_+. В случае, если условие не выполнено, проверяем условие отражения: $x_{k+1} < 0$. Если да, то добавляется единица к счетчику отраженных частиц N_-. Если нет, т.е. $0 \le x_{k+1} \le h$,то нейтрон испытал k+1 столкновение внутри пластины и надо разыграть судьбу нейтрона при столкновении.

Проверяем условие поглощения: $\gamma < \Sigma_c/\Sigma$. Если да, то добавляем единицу к счетчику поглощенных частиц N_0. Если нет, тогда разыгрываем рассеяние $\mu_{k+1} = 2\gamma$-1, и цикл повторяется снова (но уже с другими значениями γ).

Все γ написаны без индексов, так как имеется в виду, что каждое значение γ используется всего один раз.

Для расчета одного звена траектории нужны три значения γ. Начальные значения для каждой траектории $x_0 = 0$, $\mu_0 = 1$.

$$p^+ \approx N^+ / N;\ p^- \approx N^- / N;\ p^0 \approx N^0 / N.$$

Код в Matlab

lecture_13_2.m

```
clc;
sigma = 15; sigma_c = 5;
h = 0.002;
N = 1000;
Nplus = 0; Nnull = 0; Nminus = 0;
x = []; m = [];
j = 1;
while (j < N)
    k = 1; x(1) = 0; m(1) = 0; cycle = 1;
    while(cycle == 1)
        lambda = (-1/sigma)*log(rand(1));
        x(k+1) = x(k) + lambda * m(k);
        if(x(k+1) > h)
            Nplus = Nplus + 1;
            cycle = 0;
        else
            if(x(k+1) < 0)
                Nminus = Nminus + 1;
                cycle = 0;
            else
                if (rand(1) < (sigma_c/sigma))
                    Nnull = Nnull + 1;
                    cycle = 0;
                else
                    m(k+1) = 2*rand(1) - 1;
                    k = k + 1;
                end
            end
        end
    end
    j = j + 1;
end
p_plus = Nplus / N
p_minus = Nminus / N
p_null = Nnull / N
p = p_plus + p_minus + p_null
```

Command Window

```
p_plus =

    0.3300


p_minus =

    0.3510


p_null =

    0.3180


p =

    0.9990
```

13.2 Решение дифференциальных уравнений в частных производных в Matlab

Многие прикладные задачи сводятся к решению дифференциальных уравнений в частных производных (ДУЧП) и их систем. Распространенным приближенных методом их решения является **метод конечных элементов** (МКЭ).

В Matlab МКЭ реализован в пакет расширения Partial Differential Equations Toolbox (**PDE Toolbox**), который предназначен для решения краевых задач с дифференциальными уравнениями в частных производных в двумерных областях.
Пакет состоит из набора функций, автоматизирующих реализацию МКЭ для решения различного типа ДУЧП 2 – го порядка и их систем: эллиптических, параболических и гиперболических. Кроме того, в состав пакета входит приложение pdetool с графическим интерфейсом пользователя, использование которого не требует глубокого понимания метода конечных элементов и упрощает доступ к набору функций пакета.

13.2.1 Метод конечных элементов

Метод конечных элементов (МКЭ) – основной метод современной вычислительной механики, лежащий в основе подавляющего большинства современных программных комплексов, предназначенных для выполнения расчетов инженерных конструкций на ЭВМ. МКЭ используется для решения разнообразных задач как в области прочностных расчетов, так и во многих других сферах: гидродинамике, электромагнетизме, теплопроводности и др.[1]

Метод конечных элементов позволяет практически полностью автоматизировать расчет механических систем, хотя, как правило, требует выполнения значительно большего числа вычислительных операций по сравнению с классическими методами механики деформируемого твердого тела. Современный уровень развития вычислительной техники открывает широкие возможности для внедрения МКЭ в инженерную практику. Поэтому знание основ метода конечных элементов и современных программных средств, позволяющих на его основе решать разнообразные задачи, в наше время для инженера является абсолютно необходимым.[1]

1. Шимановский А. О. Применение метода конечных элементов в решении задач прикладной механики : учеб.-метод. пособие для студентов технических специальностей / А. О. Шимановский, А. В. Путято ; М-во образования Респ. Беларусь, Белорус. гос. ун-т трансп. – Гомель : БелГУТ, 2008. – 61 с.

13.2.1 Метод конечных элементов

В МКЭ исследуемая конструкция мысленно разбивается на отдельные части – **конечные элементы**, соединяющиеся между собой в узлах. Совокупность соединенных между собой и прикрепленных к основанию конечных элементов образует расчетную схему, называемую конечноэлементной схемой или **конечноэлементной моделью**.

Каждый отдельно конечный элемент должен быть достаточно простым, чтобы имелась возможность легко определить перемещения и напряжения в любой его части по заданным перемещениям узлов. Связь между перемещениями узлов элемента и силами, действующими на них, задается при помощи **матрицы жесткости элемента**. Количество перемещений узлов элемента, которые однозначно определяют положение данного элемента, называют **числом степеней свободы элемента**.

Аналогично, для всей конечноэлементной схемы вводятся **матрица жесткости системы К**, или **глобальная матрица жесткости**, устанавливающая связь между перемещениями узлов системы и силами, действующими на них, а также число степеней свободы системы, или глобальное число степеней свободы – количество координат узлов системы, которые достаточно знать, чтобы однозначно определить положение всей системы. Обычно, все степени свободы представляются в виде вектора U , **называемого вектором узловых перемещений**.

13.2.1 Метод конечных элементов

Так как матрица жесткости системы устанавливает связь между силами, приложенными к ее узлам, и перемещениями ее узлов, то, имея построенную матрицу жесткости системы и зная узловую нагрузку *F*, можно найти перемещения всех узлов конечноэлементной схемы. Для этого требуется решить систему линейных алгебраических уравнений вида:

$$K \cdot \overline{U} = \overline{F_{м}} + \overline{F_{0}} \qquad (13.5)$$

где $F_{М}$ – вектор внешних сил, а F_0 есть узловой вектор начальных сил, который имеет место, например, при учете начальных температурных напряжений. Порядок этой системы равен глобальному числу степеней свободы системы. По вычисленным таким образом перемещениям определяются напряжения и деформации. Физический смысл векторов U и F определяются областью применения МКЭ:

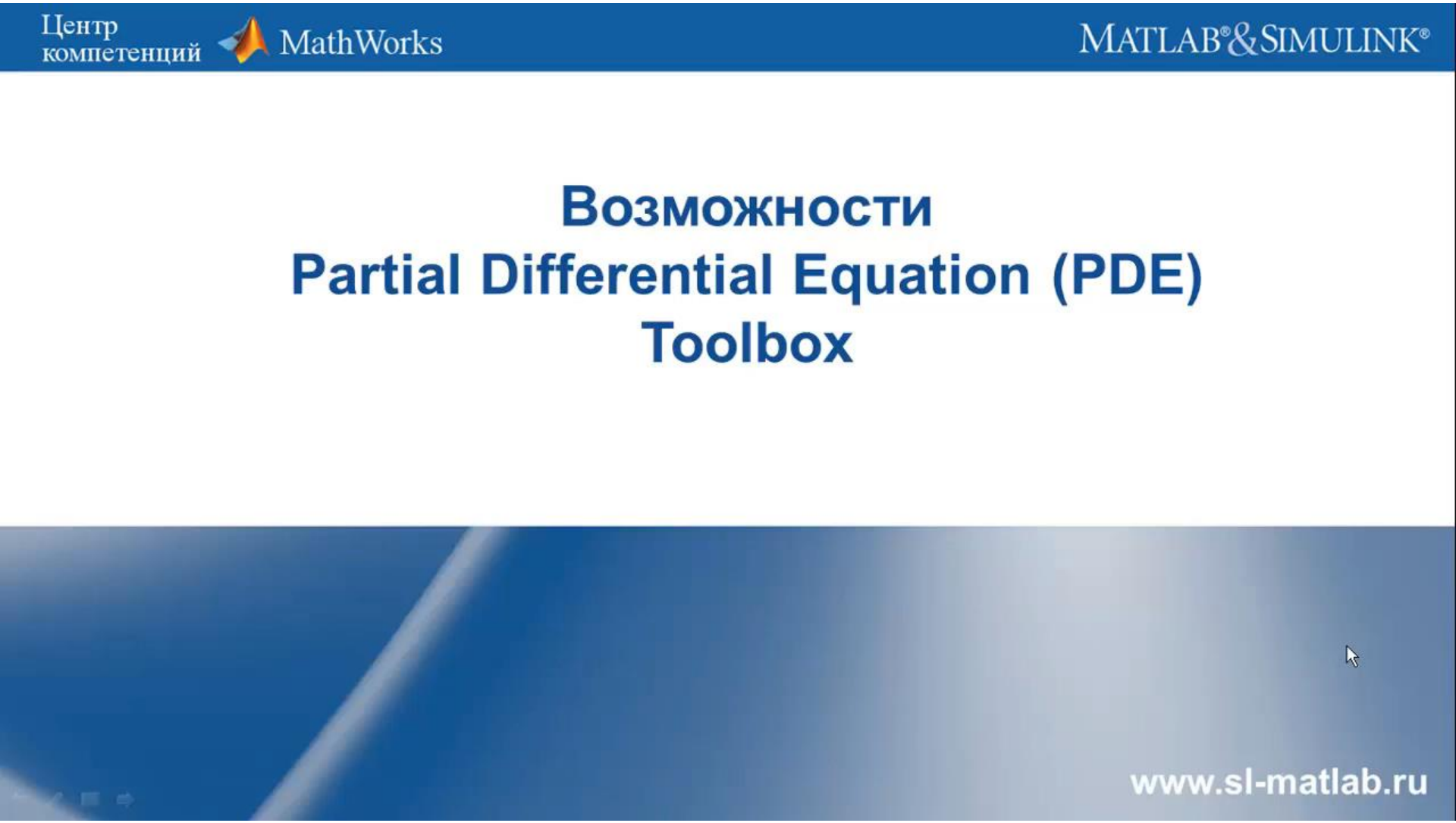
Область применения	$\overline{U}$	$\overline{F}$
Механика деформируемого твердого тела	Перемещение	Сила
Теплоперенос	Теплопроводность	Тепловой поток
Гидромеханика	Скорость	Поток
Электростатика	Электрический потенциал	Плотность заряда
Магнитостатика	Магнитный потенциал	Интенсивность магнитного поля

13.2.1 Метод конечных элементов

Основные этапы метода конечных элементов:

13.2.2 Возможности Partitional Differential Equation (PDE) Toolbox

https://www.youtube.com/watch?v=gFnFEsqXOwo



13.3 Программные комплексы статистического моделирования

Существует множество пакет имитационного и статистического моделирования, например:

GPSS (General Purpose Simulation System) – общецелевая система моделирования сложных систем, разработанная Джеффри Гордоном в 1960 г. в компании IBM. Первоначально разрабатывалась и поддерживалась компанией IBM. В настоящее время имеются версии различных разработчиков. В Интернете обширную информацию о системе предоставляют следующие русскоязычные источники:
http://www.gpss.ru – портал GPSS.RU , посвященный имитационному моделированию с использованием GPSS и других средств;
http://gpss-forum.narod.ru – сайт Юрия Носкова о GPSS.

Geant4 (**англ. GEometry ANd Tracking** — геометрия и трекинг) — инструментарий для моделирования прохождения элементарных частиц через вещество с использованием методов Монте-Карло. Разработана в CERN на объектно-ориентированном языке программирования C++.

13.3.1 Набор инструментов GEANT4

GEANT4 представляет собой набор программ для моделирования прохождения частиц через вещество

- Включает в себя инструменты для гибкого описания геометрии

- Содержит множество физических моделей взаимодействия частиц с веществом^

– Электромагнитные процессы

– Адронные процессы

– Фотон-адронные и лептон-адронные процессы

– Процессы с участием оптических фотонов

– Моделирование распадов

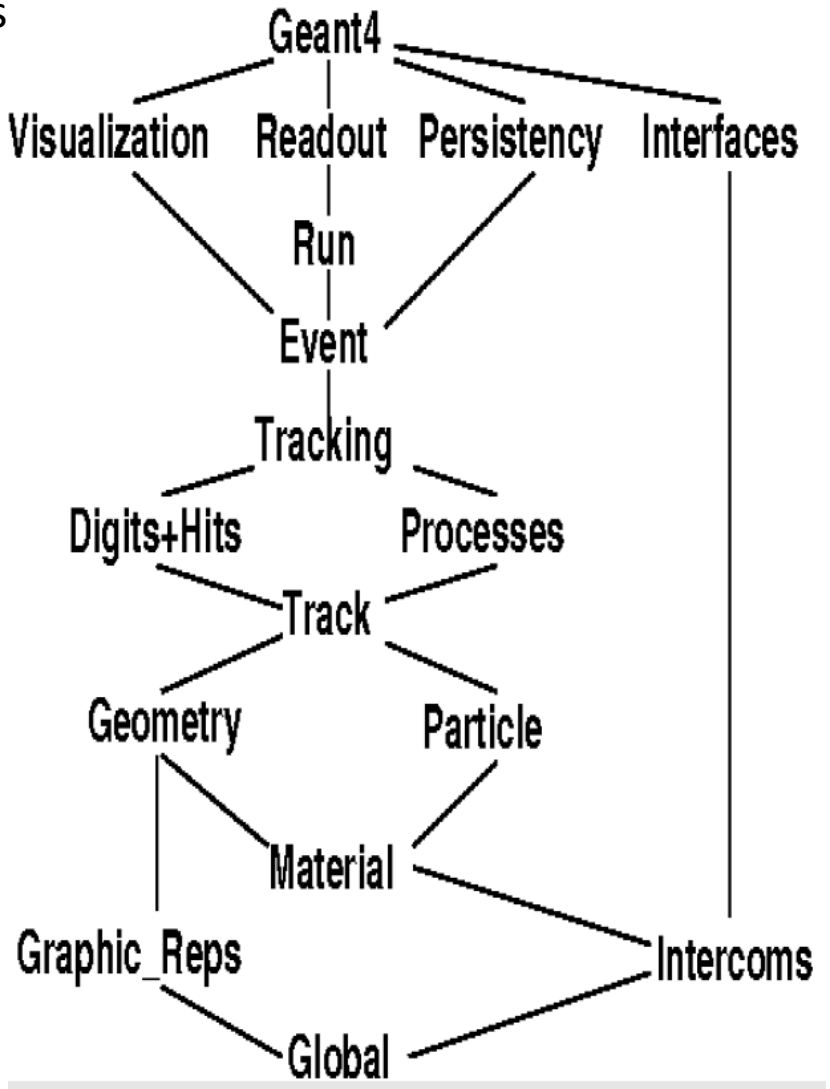
– Параметризация

– Методы использования статистических весов

13.3.1 Набор инструментов GEANT4

Документация Geant4

Geant4 User's Guide for Application Developers
Geant4 Physics Reference Manual
Geant4 User's Guide for Toolkit Developers
https://geant4.web.cern.ch/



Ядро GEANT4

Классы, описывающие основные понятия: сеанс, событие, трек, шаг срабатывание, траектория.

Базовый механизм:

- описания геометрии
- описания физических процессов
- визуализации и интерфейсов пользователя