SpdRoot: пакет моделирования и анализа для Spin Physics Detector (SPD).

Version 0.0.1

Ноябрь 2018, ОИЯИ, Дубна

Содержание

1 Введение

Измерения асимметрии в лептонной паре (процессы Дрелла-Яна) при столкновении неполяризованных, продольно и поперечно поляризованных протонов и пучков дейтронов предлагается выполнять на коллайдере NICA в ОИЯИ с использованием детектора спиновой физики SPD (Spin Physics Detector). Эти измерения помогут обеспечить подход ко всем коллинеарным и поперечно-импульсным зависимым функциям распределения кварков и антикварков в нуклонах. Измерения асимметрий в J/Ψ и прямых фотонов, которые предоставляют дополнительную информацию о структуре нуклона, будут выполняться одновременно с данными Дрелла-Яна с использованием специальных триггеров. Набор этих измерений позволит проверить кварк-партонную модель нуклонов на уровне twist-2 квантовой хромодинамики с минимальными систематическими ошибками.

2 Как установить

Перед компиляцией SpdRoot необходима установка следущих пакетов:

- FairSoft.
- FairRoot.

Подробную информацию о них можно получить на сайте: https://fairroot.gsi.de

2.1 FairSoft

Пакет FairSoft заботится об установке всех необходимых внешних пакетов (таких как ROOT, Geant4, Pythia, CLHEP и т. Д.) в правильном порядке и с правильными флагами компиляции. FairSoft также содержит конфигурационные скрипты, которые проверяют, установлены ли все необходимые системные пакеты. Если некоторые системные пакеты отсутствуют, конфигурационный скрипт остановится с подробным сообщением об ошибке. В итоге все дополнительное программное обеспечение будет установлено в одной директории.

Текущую версию пакета FairSoft можно загрузить по ссылке:

git clone https://github.com/FairRootGroup/FairSoft.git FairSoft

Предположим, что **/work/FairSoft** - полный путь к пакету FairSoft.

Тогда

1. Установка системных пакетов.
   Полный список необходимых системных пакетов можно найти в файле **/work/FairSoft/DEPENDENCIES**.
   Этот файл содержит также полные командные строки для установки необходимых пакетов в наиболее распространенных дистрибутивах Linux.

2. Создаем директорию для установки. Например:
   mkdir /work/fairsoft_install

3. cd /work/FairSoft
   ./configure

   Пример того, что вы можете увидеть в файле:
   /spdroot/doc/fairsoft_configure.txt

   Признаком того, что установка успешно завершена, является сообщение:

   ***** End installation of external packages without errors *****

   В результате установки, в директории fairsoft_install/ должны появиться (минимально) следующие папки:

   /work/fairsoft_install/
   bin/
   include/
   lib/
   share/

4. Создание ссылки fairsoft на директорию fairsoft_install в папке /work:
   cd /work
   ln -s fairsoft_install fairsoft

Полный список установленных пакетов и их версий можно увидеть в **/work/FairSoft/Readme.md**.

2.2 FairRoot

FairRoot представляет собой объектно-ориентированную для среду моделирования, реконструкции и анализа данных. Основная идея FairRoot заключается в предоставлении единого пакета с универсальными механизмами для решения наиболее часто используемых задач в физике высоких энергий. FairRoot позволяет сосредоточиться на деталях характеристики детекторов, избегая проблем с программным обеспечением, таких как хранение, извлечение, организация кода и т. д.

Текущую версию пакета FairRoot можно загрузить по ссылке:

git clone https://github.com/FairRootGroup/FairRoot.git FairRoot

Предположим, что **/work/FairRoot** - полный путь к пакету FairRoot
а **/work/fairsoft_install** - полный путь к директории установки FairSoft.

Тогда

1. Создаем директорию для установки FairRoot. Например:
   /work/fairroot_install

2. Задаем переменную окружения SIMPATH:
   cd /work
   export SIMPATH=/work/fairsoft_install

3. cd /work/FairRoot
   mkdir build
   cd build
   cmake -DCMAKE_INSTALL_PREFIX= "/work/fairroot_install"..
   make -j8
   make install

   В результате установки, в директории /fairroot_install должны появиться (минимально) следующие папки:

   /work/fairroot_install/
   bin/
   include/
   lib/
   share/

4. Создание ссылки fairroot на директорию fairroot_install в папке /work:
   cd /work
   ln -s fairroot_install fairroot

2.3 SpdRoot

SpdRoot - это пакет программ моделирования и анализа, который разрабатывается для будущей установки SPD (Spin Physics Detector). На данный момент SpdRoot можно рассматривать как расширенную адаптацию пакета FairRoot применительно к детектору SPD, которая, с учетом специфики установки, призвана решать аналогичные задачи. SpdRoot имеет ту же структуру кода и напрямую использует значительное количество технических возможностей, предоставляемых пакетом FairRoot.

Текущую версию пакета SpdRoot можно загрузить по ссылке:

git clone https://git.jinr.ru/Tkachenko/spdroot

Распаковываем проект в папку spdroot/

cd spdroot/

1. Проверяем содержимое файла SetEnv.sh:
   cat /spdroot/SetEnv.sh
   Необходимо исправить в файле пути, если они указаны неверно:

   ```
   #!/bin/bash
   export SIMPATH=/work/fairsoft_install
   export FAIRROOTPATH=/work/fairroot_install
   source build/config.sh
   ```

   (редактировать строку "source build/config.sh"не нужно)

2. cd spdroot/

   . SetEnv.sh

   (игнорируем предупреждение "build/config.sh: no such file or directory")

   mkdir build

   cd build

   cmake ..

   make -jN (N - число процессоров компьютера)

2.4 Создание HTML - документации

Имеется возможность создать html-документацию со списком классов SpdRoot.

Предварительные требования: Пакет Doxygen (debian, ubuntu: sudo apt-get install doxygen)

1. Включите опцию "BUILD_DOXYGEN"в конце файла spdroot/CMakeLists.txt :
   Option(BUILD_DOXYGEN
   “Whether to generate automatic documentation with Doxygen ”
   OFF)
2. Удаляем (или очищаем) build/ директорию:
   rm -rf build/
3. Собираем spdroot:
   mkdir build
   cd build/
   cmake ..
   make
4. Make html:
   make html-doc

Расположена html-документация в spdroot/build/doc/html/classes.html .

3 Общая структура

В SpdRoot доступны для моделирования три варианта установки (геометрии):

- Соленоид;
- Тороид;
- Гибрид (он похож на тороид, но существуют некоторые различия в конструкции магнитов и Tracking system).

Каждому типу геометрии соответствует свой тип магнитного поля.

Общая структура SpdRoot представлена в таблице 1 :

Директория	Поддиректория	Краткое описание
ts(s,t)/	ecps/ barrel/	Tracking system: endcaps and barrel (программный код)
ecal(s,t)/	ecps/ barrel/	Электромагнитный калориметр: endcaps and barrel (программный код)
rs(s,t)/	ecps/ barrel/	Range system: endcaps and barrel (программный код)
geometry/		Файлы с описанием материалов и геометрии
input/		Карты магнитного поля
gconfig/		Конфигурационные файлы и скрипты для MC-приложения
spdgenerators/		Первичные генераторы (программный код)
field/		Магнитное поле (программный код)
spddata/		Выходные параметры: данные, параметры, геометрия (программный код)
common/		общие классы, области геометрии (программный код)
passive/		Пассивные модули (программный код)
spddisplay/		Визуализация, просмотр событий (программный код)
test/		experimental (программный код)
doc/		Документация
tools/		Вспомогательные утилиты
scripts/		run scripts
build/		Папка установки

Таблица 1: Общая структура пакета SpdRoot

4 Как запустить

Прежде чем приступить к работе, необходимо зупустить bash-скрипт, который устанавливает необходимые переменные окружения и находится в папке spdroot/:

```
cd spdroot/
. SetEnv.sh
```

Важное замечание! Задавать переменные окружения с помощью SetEnv.sh необходимо каждый раз перед началом работы.

Далее переходим в папку macro/

```
cd macro
root -l
```

В папке macro/ содержится некоторое количество root-скриптов, с помощью которых можно смоделировать событие (или серию событий) в детекторе. Все моделирующие скрипты имеют в названии слово **Simu** и в качестве параметра могут принимать число событий. Например:

```
root [0] .x SimuHyb.C          (будет смоделировано одно событие)
root [0] .x SimuHyb.C(5)       (будет смоделировано 5 событий)
```

Присутствие в названии скрипта слова **Hyb**, **Tor** или **Sol** указывает на выбор для моделирования одной из доступных геметрических конфигураций установки: гибридной, тороидальной или соленоидальной и соответствующего магнитного поля.

Важное замечание! Фактически, **Simu**-скрипты представляют собой конфигурационные файлы, с помощью которых осуществляется выбор геометрии установки и ряд параметров моделирования. Некоторые параметры моделирования, которые могут быть заданы в **Simu**-скриптах описаны в разделе 5.

В результате работы моделирующего скрипта будет получено пара файлов: файл с данными **run*.root** (треки, хиты, геометрия установки и т. д.) и файл с парметрами, **param*.root** (геометрия, магнитное поле, ...):

```
run_tor.root    param_tor.root      (для гибридной и тороидальной геометрии)
или
run_sol.root    param_sol.root      (для геометрии соленоидального типа)
```

Для просмотра содержимого файла с данными можно воспользоваться одним из скриптов, запускающих специальную программу - viewer. В качестве параметра скрипты могут принимать номер события, которое должно быть показано после запуска. По умолчанию будет показано первое событие в файле (если была включена опция сохранения в файл геометрических треков, если нет - будет показана только геометрия установки). Нумерация событий начинается с нуля:

```
root [0] .x DisplayTorEvent.C       (первое событие из файла run_tor.root)
root [0] .x DisplaySolEvent.C(3)    (пятое событие из файла run_sol.root)
```

Более подробно о программах из директории macro/ можно узнать в разделе 7.

5 Входные параметры

Список основных настроек для моделирования, который доступен в **Simu**-скриптах (см macro/ directory):

- **Количество генерируемых событий**.

  Число событий указывается как аргумент главной функции и, соответственно, может быть задан при запуске макроса, см., например, раздел 4 (по умолчанию моделируется только одно событие).

- **Выходные файлы**.

  По завершению работы **Simu**-скрипта будут получены два выходных файла: с результатами (данными) и параметрами моделирования в формате root. Имена этих файлов могут быть выбраны в скрипте произвольно:

  $$\textbf{outFile} = \textbf{"run\_tor.root"}$$
  $$\textbf{parFile} = \textbf{"params\_tor.root"}$$

  Следует учитывать, что при запуске анализирующих скриптов и программ для визуализации из директории macro/ (например, Display*Event.C) имена входных файлов должны быть в них также исправлены соответствующим образом.

- **Media file**.

  По правилам FairRoot, файл с описанием материалов, необходимых для построения установки должен находиться в директории /spdroot/geometry. Стандартный файл называется **media.geo**. Передать в программу файл с описанием материалов можно с помощью метода:

  $$\textbf{FairRunSim::SetMaterials("media.geo")}$$

  Если в файле нет нужного материала, он может быть добавлен пользователем. Подробнее можно узнать по ссылке: https://fairroot.gsi.de/?q=node/34

- **Транспорт-генератор**.

  В качестве транспортного генератора на текущий момент может быть выбран TGeant4 или TGeant3. По умолчанию задан TGeant4:

  $$\textbf{FairRunSim::SetName("TGeant4")}$$

  Подробнее о настройках генератора (например, выбор physics list для Geant4) можно узнать в разделе 8.

- **Набор частей установки: beam pipe, магнит, setup frame, tracking system, электромагнитный калориметр, range system и др**.

  SpdRoot позволяет менять геометрическую конфигурацию установки напрямую из **Simu**-скрипта. Установка состоит из совокупности геометрических модулей (пассивных и активных) которые можно добавлять (или убирать просто закомментировав соответствующую строку в скрипте) по необходимости с помощью метода

  $$\textbf{FairRunSim::AddModule(module)}$$

  где **module** - это указатель на объект, описывающий данный модуль и который должен быть предварительно создан и настроен (если это необходимо).

  К **пассивным** частям (модулям) установки относятся магнит, пучковая трубка, элементы поддерживающей конструкции, а также **CAVE** - the top level geometry

module. **Активные** модули (трековая система, электромагнитный калориметр, range system и др), кроме геометрии, также содержит описание того какая физическеская информация о смоделированных событиях будет сохранена в выходном файле.

Более подробная информация о модулях представленна в разделах 11, 12 и 13.

- **Глобальные параметры геометрии**.

  Ряд глобальных геометрических параметров установки доступен через методы статического класса **SpdCommonGeoMapper** (см. раздел 11).

- **Первичный генератор**.

  Первичный генератор формирует список частиц (как правило, выходящих из области взаимодействия пучков - "вершины") для дальнейшей их транспортировки через установку. Одновременно может быть подключено несколько генераторов. Добавить генератор можно с помощью метода

  SpdPrimaryGenerator :: AddGenerator(generator) ,

  где **generator** указатель на объект, описывающий данный генератор и который должен быть предварительно создан и настроен соответствующим образом.

  SpdPrimaryGenerator является стандартным классом-контейнером для списка генераторов. В процессе формирования списка первичных частиц будут вызваны все генераторы в том же порядке, в котором они были добавлены в контейнер.

  Основным первичным вершинным генератором в SpdRoot на данный момент является **Pythia6**. Список доступных генераторов, а также их описание, представлены в секции 9.

- **Магнитное поле**.

  Магнитное поле задается независимо от геометрии установки, несмотря на то, что его конфигурация, очевидно, тесно связана с конструкцией магнита и распределением вещества в объеме установки. В SpdRoot магнитное поле может быть задано: a) аналитически; b) с помощью таблицы; c) в виде произвольной комбинации набора первых двух вариантов. Добавить поле можно с помощью метода:

  FairRunSim :: AddField(field) ,

  где **field** это указатель на объект описывающий данный генератор и который должен быть предварительно создан и настроен соответствующим образом.

  Также в **Simu**-скрипте может быть определен field region, т. е. геометрическая область, за пределами которой величина магнитного поля полагается равной нулю. Field region может быть определен как:

  SpdField :: CreateFieldRegion("region_type") ,

  где **"region_type"** один из параметров: "box "tube"or "physical".

  Для более подробной информации, см 10.

- **Другие настройки**.

  Некоторые другие настройки могут быть также достуны в **Simu**-скрипте. Например, создание отдельного файла с геометрией установки или запись в файл output

data геометрических треков частиц (через TGeoTrack) для последующей визуализации события с помощью Display*Event.C:

FairRunSim :: SetStoreTraj(true)

Важное замечание! Настоятельно рекомедуется сохранять геотреки только при запуске **Simu**-скрипта с небольшим числом событий из-за значительного объема получаемого на выходе файла с данными.

6 Выходные параметры

7 Root-скрипты для симуляции, визуализации, анализа и тестирования

Содержание директории **macro**/:

- ряд макросов для моделирования и визуализации (этот раздел);
- geom/ - настройка геометрии (и ее частей) (разделы 12, 13);
- primgen/ - примеры и тесты основных вершинных генераторов (раздел 9);
- field/ - магнитное поле (раздел 10);
- analysis/ - скрипты для анализа (раздел 14);
- testmisc/ - разнообразные программы.

7.1 Симуляция

Ряд root-скриптов для разных видов геометрии и магнитного поля (гибридное, тороидальное, соленоидальное) доступны в папке **macro**/:

Simu[Hyb,Tor,Sol].C - стандартные макросы для симуляции:

- Используемый вершинный генератор: Pythia6 (global option: 1, minimum bias: MSEL = 1, beam = p+p, энергия = 26 ГэВ в системе цента масс).

XSimu[Hyb,Tor,Sol].C - эти программы главным образом предназначены для тестирования геометрии и магнитного поля:

- Задействованы два первичных вершинных генератора изотропного типа;
- Без вещества;
- Сохраняет геометрию в отдельный root-файл.

Simu[Hyb,Tor,Sol]ExtDecayer.C - примеры моделирования с внешним декаером:

- Используемый вершинный генератор: Pythia6 (global option: 0, minimum bias: MSEL = 0, J/Ψ будет сгенерирован в первичной вершине, beam = p+p, энергия = 8 ТэВ в системе цента масс).

7.2 Визуализация

Программы визуализации выходных данных (настройка геометрии и треков частиц) доступны в папке **macro**/:

- DisplaySolEvents.C - для геометрии соленоида;
- DisplayTorEvents.C - для геометрии соленоида и тороида.

7.3 Тестирование

Некоторые root-скрипты для тестов.

8 Транспорт-генератор и некоторые настройки MC-приложения

Необходимы несколько классов для моделирования с VMC:

* TVirtualMC: обычно означает транспорт-генератор (TGeant4, TGeant3, и т.д)
* TVirtualMCStack: контейнер с транспортируемыми частицами (SpdStack).
* TVirtualMCApplication: Задачи MC (FairRunSim);

Глобальные настройки для транспорт-генератора и MC stack доступны в макросах g4Config.C и g3Config.C для Geant4 и Geant3 соответственно. Большинство настроек MC-приложения должны быть определены непосредственно в Simu-скриптах (см 5).

8.1 Geant4

1. Доступные настройки g4Config.C:

   - выбор physics list;
   - physical cuts (по энергии) for particles and processes selection in Geant3-like style through the SetCuts.C loading;
   - настройки пользователя для транспорт-генератора Geant4 при загрузке g4config.in;
   - VMC stack settings.

2. Пользовательские настройки decayer (при необходимости): DecayConfig.C .

3. Специальные пользовательские определения распадов (при необхлдимости): UserDecay.C .

8.2 Geant3

Geant3 доступен, но еще не настроен и не протестирован.

9 Первичные генераторы

9.1 Pythia6

9.2 Pythia8

9.3 IsotropicGenerator

10 Магнитное поле

10.1 Вид поля

Магнитное поле в SpdRoot может быть задано одним из следующих способов:

a) **Аналитически (функция)**. Поле в любой точке вычисляется заранее описанной функцией (или алгоритмом), оформленной в виде класса. Простейший случай аналитического описания - постоянное поле (класс **SpdConstField**).

b) **С помощью таблицы(карты)**. В этом случае поле задается в узлах кубической сетки, предварительно записанной в отдельный файл. Значения поля в произвольной точке определяются с помощью межузловой аппроксимации (линейной по умолчанию). Для загрузки из файла поля данного типа используется стандартный класс
SpdFieldMap.

c) **Набор полей**. Произвольная совокупность описанных выше двух типов полей (а также соответствующих им геометрических областей). В этом случае величина поля в точке по умолчанию определяется как сумма значений всех полей из набора. Для подрузки набора полей разных типов используется класс
SpdMultiField.

10.2 Область действия поля

Для полей типов a) и b) может быть задана область их действия, т. е. геометрическая область, за пределами которой значение поля будет считаться равным нулю. На данный момент возможны три варианта выбора field region:

- **Box**. Область имеет форму параллилепипеда. Для определения требуется задать шесть параметров - границы области вдоль координатных осей.
- **Tube**. Область имет вид цилиндра ось которого совпадает с координатной осью Z. Для определения требуется задать четыре параметра: минимальный и максимальный радиус цилиндра, а также границы области вдоль оси Z.
- **Physical**. Специальный случай - область определяется либо по заданному имени (точнее, по объему geopath) физического объема, либо по материалу.

10.3 Учебное пособие

Учебные скрипты с магнитным полем размещены в **spdroot/macro/field** .

11 Модули установки и глобальные параметры геометрии

Конфигурация установки определятся входящим в ее состав набором модулей. В узком смысле, setup module - это просто программный код, содержащий описание геометрии части установки, которая физически и функционально может рассматриваться как единое целое. Такое понимание, в принципе, справедливо для модулей, обозначенных как **Passive**. В широком смысле setup module - это программный код, состоящий, как правило, из набора классов (часто оформленных в процессе компилляции в отдельную библиотеку), которые кроме описания собственно геометрии, также выполняют сбор и некоторую первичную обработку полезной физической информации, получаемой в процессе моделирования, в формате пригодном для сохранения в файл. Модули имеющие широкую функциональную нагрузку определяются как **Active**. Активный модуль может также содержать средства для вторичной обработки данных - работе с файлами, полученными в результате моделирования. Setup module (как активный, так и пассивный) может в процессе моделирования формировать список своих параметров, которые потом будут добавлены в выходной файл.

Каждый модуль в SpdRoot имеет свой уникальный номер (целое число), который принимает положительные значения ($>$ или $= 0$) для активных модулей и отрицательные для пассивных.

Detailed description of passive and active modules, which are available for simulation and testings are presented in sections 12 and 13 respectively.

12 Пассивные модули

Полный список пассивных модулей - уникальный идентификатор модуля, соответствующий класс для добавления в **Simu** -скрипт и применимая геометрия представлены в таблице:

13 Активные модули

Полный список активных модулей - уникальный идентификатор модуля, соответствующий класс для добавления в **Simu** -скрипт и применимая геометрия представлены в таблице:

14 Анализ

Прямой доступ к данным для анализа (пример):

macro/analysis/

- CheckOutputData.C - поиск параметров моделирования, заголовка события, треков и branches with hits в выходных файлах.