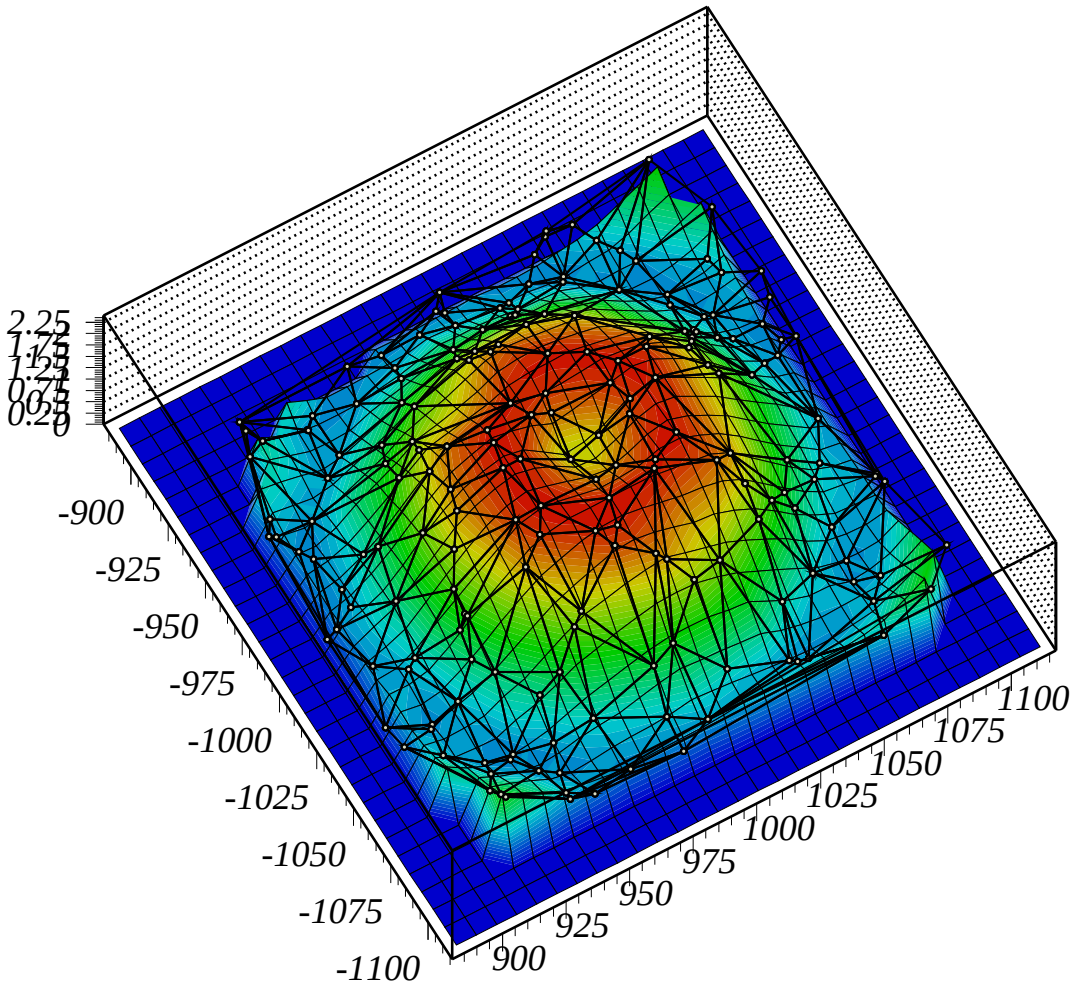
Physics Analysis Workstation (PAW)

© Е.М. Балдин*



Del Trirx ry rz 200

*e-mail: E.M.Baldin@inp.nsk.su

Скрипт, который создал картинку, взят с `http://paw.web.cern.ch/paw/contributions/`. Автор скрипта Luke Jones.

Оглавление

Эта статья была опубликована в августовском номере русскоязычного журнала Linux Format (`http://www.linuxformat.ru`) за 2006 год. Статья размещена с разрешения редакции журнала на сайте `http://www.inp.nsk.su/~baldin/` и до января месяца все вопросы с размещением статьи в других местах следует решать с редакцией Linux Format, а после все вопросы следует решать со мной.

Текст, представленный здесь, не является точной копией статьи в журнале. Текущий текст в отличии от журнального варианта корректор не просматривал. Все вопросы по содержанию, а так же замечания и предложения следует задавать мне по электронной почте `mailto:E.M.Baldin@inp.nsk.su`.

Текст на текущий момент является просто *текстом*, а не книгой. Поэтому результирующая доводка в целях улучшения восприятия текста не проводилась.

2 PAW tutorial

Система анализа данных PAW или Physics Analysis Workstation для работы не требует доскональных знаний всех подсистема и команд. Но чтобы действовать эффективно следует изучить логику построения команд и стандартные приёмы. Это позволит в дальнейшем легко получать навыки для выполнения более сложных задач.

Если Вы не планируете использования PAW, то в любом случае полезно присмотреться к этому инструменту, так как он достаточно прост и основные концепции, необходимые для анализа данных, там достаточно прозрачны для понимания. Создатели PAW действовали по принципу минимализма. Делалось только необходимое — никаких «рюшечек», зато просто. Жёсткая структура команд дополнена возможностью писать свои функции и скрипты.

2.1 Простейший анализ

Учиться лучше всего на примере реального анализа. Попробуем сделать нечто подобное.

Считаем, что программа `paw` уже запущена и мы находимся в рабочей директории. Вызов внешних команд обеспечивается с помощью инструкции SHELL (можно сократить до `sh`).

```
PAW > sh ls
ascii.png   lkravg.dat     paw.metafile     ee-ang.rz
th1.eps     last.kumac     last.kumacold    sin.dat
```

Необходимо провести предварительный анализ данных представленных в текстовом файле `lkravg.dat`:

```
1099279655   4119  0.8318  0.0014  1.13  5.99195
1099397693   4126  0.8404  0.0032  1.07  6.001685
  ...
```

Колонки соответствуют `time_t` — времени в секундах, номеру измерения, исследуемое значение, ошибки значения для текущего измерения и двум сторонним параметрам, от которых интересующее нас значение может зависеть. Задача: имея время и два сторонних параметра, попробовать предсказать исследуемое значение.

Анализировать можно и без модели явления. Но чтобы правильно подготовить данные для исследования, необходимо её иметь. Легенда для этих данных следующая: исследуемое значение представляет из себя степень «ухудшения» качества

жидкого криптона (LKr — Liquid Kripton) в LKr-калориметре для регистрации энергии элементарных частиц. Качество вычисляется в относительных единицах по амплитуде сигнала от космических мюонов (эти частицы хорошо выделяются и есть всегда). Два параметра от которых может зависеть качество: избыточное давление LKr и магнитное поле в котором калориметр находится. Избыточное давление примерно линейно связано с температурой LKr, что влияет на амплитуду сигнала. В магнитном поле прямолинейная траектория мюона искажается, что тоже может влиять на амплитуду сигнала.

Для начала следует прочитать данные из текстового файла, для этого воспользуемся командой VECTOR/READ (`help ve/re`):

```
#чтение текстового файла в вектора
PAW > ve/re time,run,avg,avg_er,P,H lkravg.dat
# меняем тип маркера
PAW > set mtyp 2
# рисуем зависимость исследуемого значения от времени
PAW > ve/pl avg%time
```

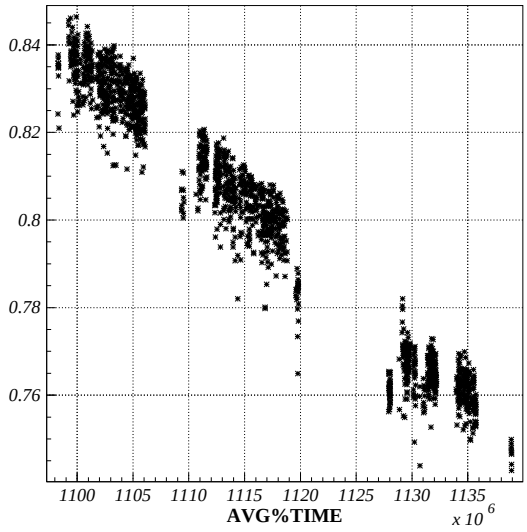


Рис. 2.1. Зависимость качества LKr от времени.

Из рисунка видно, что исследуемое значение в среднем уменьшается за большой промежуток времени (исследуемый интервал равен году и четырём месяцам) и в то же время у данных есть структура соответствующая более короткому интервалу.

Сначала исключим временную зависимость. Воспользуемся для этого стандартной процедурой подгонки для векторов VECTOR/FIT X Y EY FUNC (`help ve/fit`), где `X` соответствует оси времени, `Y` — исследуемому значению, `EY` — ошибки определения значения, а `FUNC` — подгоночная функция. Вместо `FUNC` можно передать любую функцию, написанную на FORTRAN, но в этом случае достаточно полинома

первой степени. Для полинома в PAW есть сокращённое обозначение `PN`, где `N` — степень полинома.

```
#подгоняем временную зависимость прямой
PAW > ve/fit time avg avg_er P1

      ********************************************
      *                                          *
      * Function minimization by SUBROUTINE HFITV *
      * Variable-metric method                   *
      * ID =          0  CHOPT =                 *
      *                                          *
      ********************************************
 Convergence when estimated distance to minimum (EDM) .LT. 0.10E+01

EXT PARAMETER                APPROXIMATE     STEP         FIRST
NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
1      P1       3.3620       0.50070E-03  0.85051E-02    -223.19
2      P2    -0.22939E-08    0.44872E-12  0.58032E-11  -0.27325E+12

 CHISQUARE = 0.1680E+02   NPFIT =   1644
```

При выполнении процедуры подгонки PAW выдаёт стандартную «портянку». Для пользователя основной интерес представляют результаты подгонки. В случае полинома первой степени подгоняются два параметра `P1` — константа и `P2` — коэффициент пропорциональности

Для исключения временной зависимости воспользуемся пакетом работы с векторами `SIGMA` (`help sigma`). С помощью команды `sigma` векторами можно манипулировать, как если бы это были обычные переменные:

```
#создаём новый вектор уже без временной зависимости
PAW > sigma avg1=avg-3.3620+0.22939E-08*time
```

Прежде чем действовать дальше оценим для начала какую точность в принципе можно ожидать. Вектору исследуемой величины `avg`, соответствует вектор ошибок `avg_er`. Посмотрим чему эти ошибки равны. Число измерений превосходит полторы тысячи, поэтому просмотреть все значения глазами не очень реально. Поэтому создадим гистограмму:

```
#создаём из значений вектора гистограмму №15
PAW > ve/pl avg_er 15
#включаем отображение статистики (число событий/среднее/разброс)
PAW > opt sta
#меняем цвет гистограммы на красный
PAW > set hcol 2
#вывод гистограммы №15 в диапазоне от 0. до 0.01
```

```
PAW > hi/plot 15(0.:0.01)
```

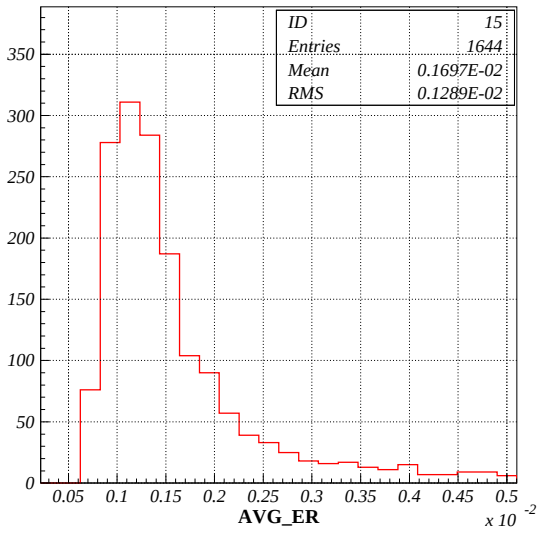


Рис. 2.2. Гистограмма ошибок, измеряемого параметра.

Из гистограммы видно, что большинство измерений имеют ошибку меньше $0.15 \cdot 10^{-2} \div 0.2 \cdot 10^{-2}$. То есть, даже при самом удачном раскладе точность предсказания не будет выше этих $0.15\% \div 0.2\%$.

После того как убрали основную (временную) зависимость можно проверить зависит ли изучаемая переменная от давления (P) и магнитного поля (H). Для этого воспользуемся способностью PAW создавать и отображать двумерные гистограмм:

```
#создаём двумерную гистограмму давление от avg1
PAW > ve/pl P%avg1 10
#рисуем двумерную гистограмму 10 в виде поверхности
PAW > SURF 10 65 -30 1
```

Обратите внимание на разделитель % между векторами в команде VECTOR/PLOT. Значениям первого вектора противопоставляется ось ординат (ось Y), а значениям второго ось абсцисс (ось X).

Команда `HISTOGRAM/2D_PLOT/SURFACE [ID THETA PHI CHOPT]` (`help surf`) позволяет отобразить двумерную гистограмму в виде поверхности. Здесь `ID` — номер гистограммы, `THETA` и `PHI` — углы поворота гистограммы θ и φ в сферической системе координат, `CHOPT` — опции изображения. Из картинки видно, что какая-то зависимость есть — избавимся от неё, как это сделали с временной зависимостью. В результате подгонки сначала для давления, а потом для поля были получены ещё две формулы:

```
#поправка на давление
```

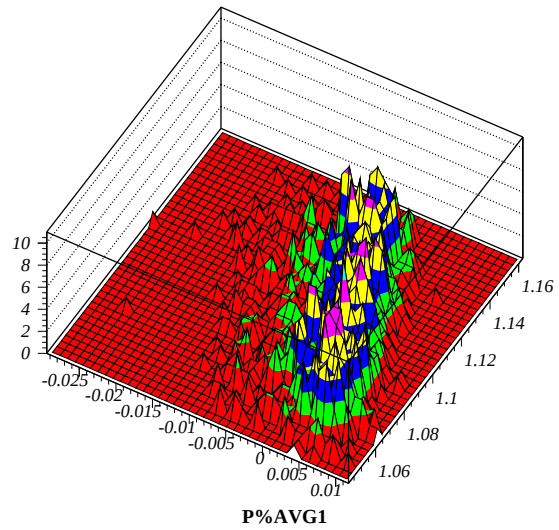


Рис. 2.3. Распределение зависимости от давления

```
PAW > sigma avg2=avg1-0.10901+0.99336E-01*P
#поправка на магнитное поле
PAW > sigma avg3=avg2+0.10844E-01-0.18258E-02*H
#сравниваем две гистограммы до и после поправок
#на давление и магнитное поле
#переключаем цвет на красный
PAW > set hcol 2
#рисуем гистограмму по значениям вектора
PAW > ve/plot avg3
#переключаем цвет на синий
PAW > set hcol 4
#рисуем вторую гистограмму поверх 's' - superimpose
PAW > ve/plot avg1 ! 's'
```

Обратите внимание на команда SET (`help GRAPHICS/SET`). Эта инструкция позволяет менять параметры графического представления данных. Если запустить её без опций, то будет выдан список переменных, которые устанавливаются с помощью этой команды.

На рис. 2.4 представлены две гистограммы. Красная — конечный результат, а синяя — то, что осталось после исключения временной зависимости. Очевидно, что после учёта давления и поля разброс уменьшился. То, что гистограммы не симметричны на самом деле указывает на то, что временная зависимость на самом деле не линейная. В реальности при подгонки использовалась экспоненциальная зависимость плюс некоторая константа. Выбор подгоночной функции был продиктован

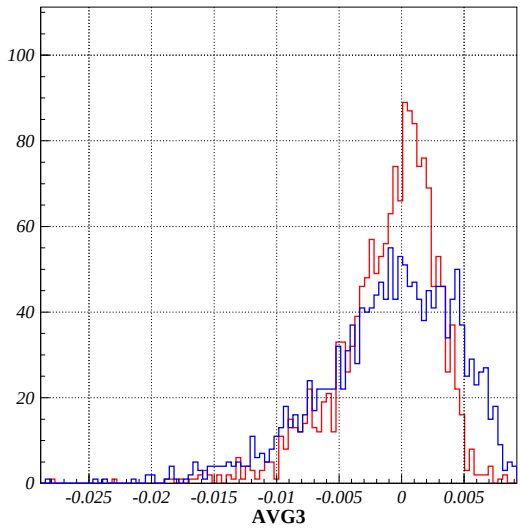


Рис. 2.4. Сравнение двух гистограмм с помощью наложения

физической моделью явления, но по большому счёту на этом временном интервале она не сильно отличается от обычной прямой.

В результате всех действий была получена зависимость:

$$\mathrm{AVG} = 3.46 + 2.29 \cdot 10^{-9} \times \mathrm{time} - 9.9 \cdot 10^{-2} \times \mathrm{P} + 1.8 \cdot 10^{-3} \times \mathrm{H} \qquad (2.1)$$

Эта функция позволяет предсказывать исследуемое значение в зависимости от времени, давления и магнитного поля. Хотелось бы понять точность этого предсказания. Для оценки точности достаточно получить распределение разницы значений между экспериментальными точками и этой зависимостью и взять его ширину. Можно просто оценить ширину распределения на глазок, посмотрев на рис. 2.4, а можно взять эту ширину из известной функции более-менее описывающей это распределение, например, из функции Гаусса:

```
#создаём из значений вектора avg1 гистограмму №11
PAW > ve/plot avg1 11
#подгоняем гистограмму №11 в диапазоне (-0.007,0.007)
#функций Гаусса (G - Gauss)
PAW > hi/fit 11(-0.007:0.007) G
                ...
 EXT    PARAMETER                                    ...
 NO.     NAME         VALUE            ERROR         ...
   1     Constant      51.608           1.9494       ...
   2     Mean        0.39292E-03     0.20076E-03     ...
   3     Sigma       0.51728E-02     0.27301E-03     ...
                ...
```

```
#создаём из значений вектора avg3 гистограмму №13
PAW > ve/plot avg3 13
PAW > hi/fit 13(-0.004:0.004) G
                    ...
 EXT    PARAMETER                                        ...
 NO.     NAME          VALUE            ERROR           ...
   1     Constant       75.655           3.0477         ...
   2     Mean        0.20614E-03     0.11857E-03       ...
   3     Sigma       0.29684E-02     0.16655E-03       ...
                    ...
#делим графическое окно на две зоны по оси ординат (help zone)
PAW > zone 1 2
#меняем цвет гистограммы на синий
PAW > set hcol 4
#рисуем гистограмму в диапазоне (-0.009,0.009)
PAW > hi/pl 11(-0.009:0.009)
#меняем цвет гистограммы на красный
PAW > set hcol 2
PAW > hi/pl 13(-0.009:0.009)
```

Из всех значений в данном случае интересно только значение ширины распределения, или `SIGMA`[1]. Если учитывать только временную зависимость, то точность

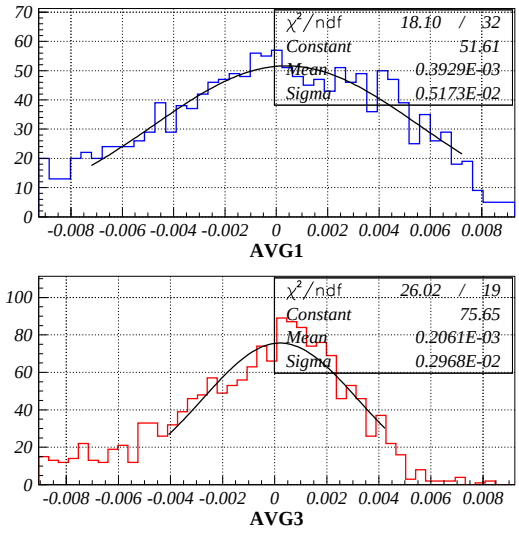


Рис. 2.5. Сравнение двух гистограмм. Подгонка функцией Гаусса

предсказания будет примерно 0.52%, если же учесть давление и магнитное поле,

[1] В случае гауссоподобного распределения в диапазоне $(\bar{x}-\sigma, \bar{x}+\sigma)$ лежит примерно 68% от всех событий. $\bar{x}$ — среднее значение или `MEAN`, σ соответствует `SIGMA`.

то точность улучшится до 0.30%, что существенно хуже идеальных $0.15\% \div 0.2\%$. Очевидно, что остались ещё какие-то неучтённые систематики.

Оценим какую систематику удалось выбрать, учтя зависимость от давления и магнитного поля. Для этого воспользуемся системой COMIS (`help comis`):

```
PAW > comis
 PAW
CS> type sqrt(0.52**2-0.30**2)
MND> end
 *T   SQRT(0.52**2-0.30**2) = 0.4247352
 PAW
CS> end
```

В общем, неплохо. Кстати, более тщательный анализ никаких кардинальных улучшений не дал — удалось только уменьшить «хвосты» и сделать итоговое распределение более симметричным за счёт выбора более сложной подгоночной функции.

Быстрый анализ позволяет оценить к какой точности имеет смысл стремиться. Это очень важно, так как сложность получения большей точности увеличивается от требуемой точности существенно нелинейным образом. Фактически на только что изложенный анализ по разным причинам ушёл примерно месяц реального времени. Правда, основные проблемы были вовсе не технические. В частности очень много времени ушло на осознание, что исследуемая величина зависит от времени — это не казалось очевидным.

2.2 Ntuple

То, что только что было сделано с помощью векторов можно проделать с помощью ntuple. Для этого надо сначала создать ntuple (NTUPLE/CREATE), а затем считать в него текстовый файл (NTUPLE/READ).

```
# создаём ntuple с ID=1
PAW > nt/cre 1 'LKr quality' 6 ! ! time run avg er P H
#читаем в ntuple с ID 1 текстовый файл
PAW > nt/read 1 lkravg.dat
 ==>    1644 events have been read
PAW > nt/print 1
*********************************************************
* NTUPLE ID=     1   ENTRIES=    1644    LKr quality
*********************************************************
*   Var numb   *    Name     *     Lower      *     Upper      *
*********************************************************
*       1      * TIME         * 0.109828E+10 * 0.113892E+10 *
*       2      * RUN          * 0.406400E+04 * 0.709400E+04 *
*       3      * AVG          * 0.742800E+00 * 0.846400E+00 *
*       4      * ER           * 0.700000E-03 * 0.201000E-01 *
```

```
*       5       * P             * 0.104400E+01 * 0.116000E+01 *
*       6       * H             * 0.000000E+00 * 0.704971E+01 *
*************************************************************
#включаем отображение статистики
PAW > opt stat
#отрисовываем разницу между экспериментом и предсказанием
#с различным ограничением на величину ошибки er
PAW > set hcol 1
PAW > nt/pl 1.avg-(3.456-2.29E-9*time-9.9E-2*P+1.8E-3*H)
PAW > set hcol 2
PAW > nt/pl 1.avg-(3.456-2.29E-9*time-9.9E-2*P+1.8E-3*H) er<0.0015 ! ! ! '
PAW > set hcol 3
PAW > nt/pl 1.avg-(3.456-2.29E-9*time-9.9E-2*P+1.8E-3*H) er>0.0015 ! ! ! '
```

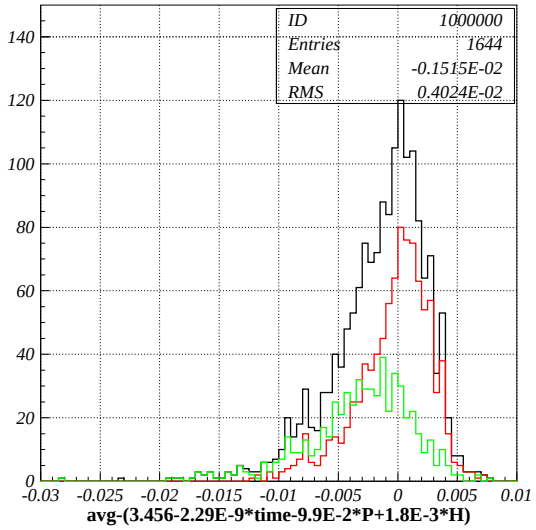


Рис. 2.6. Разность между экспериментом и предсказания в зависимости от наложенного условия на величину ошибки.

Преимущество при использовании ntuple заключается в том, что интерактивно можно накладывать условия для фильтрации данных. Обычно, ntuple создаются с помощью внешних программ, а PAW используется уже для интерактивного анализа. В стандартной документации к PAW очень подробно описывается как это делается.

2.3 Гистограммы

Гистограммы это базовые объекты PAW. Данные непосредственно перед отображением почти всегда преобразуются в одномерную или двумерную гистограмму. На

гистограмму можно просто смотреть, а можно подгонять какой-либо теоретической зависимостью (HISTOGRAM/FIT).

```
#поиск rz-файлов в базовой директории (файл создан внешней программой)
PAW > sh ls *.rz
ee-ang.rz
#чтение файла
#известно что в файле есть ntuple с ID=1
PAW > hi/fil 1 ee-ang.rz
#создаём гистограмм №20 из переменной E1 с некоторыми условиями
PAW > nt/plot 1.E1 f1=11&&f2=-11&&E1<2 ! ! ! ! 20
#подгоняем гистограмму распределением Гаусса
PAW > hi/fit 20 G
```

Из рис. 2.7 видно, что наблюдаемое распределение функцией Гаусса не подгоняется. Надо что-то делать. Очевидно, что теоретическая функция должна быть как минимум не симметрична. Для этой цели подойдёт, так называемый, логарифмический гаусс. Для подгонки надо создать файл `loggaus.for` примерно со следующим содержанием:

```
C Файл loggaus.for
      real function loggaus(x)
C   С помощью этого common-блока PAW получает доступ
C   к параметрам функции
      common/PAWPAR/PAR(4)
      sqrtln4=1.177410022515475
      A=PAR(1)
      pike=PAR(2)
      sigma=PAR(3)*pike/100.
      assim=PAR(4)
      loggaus=0.
      if (abs(assim).le.1.E-6) then
         assim=sign(1.E-6,assim)
      endif
      if (sigma.le.0.) goto 10
      xx=1.+sinh(assim*sqrtln4)/sqrtln4*(x-pike)/sigma
      if (xx<1.E-07) goto 10
      loggaus=A*exp(-((log(xx)/assim)**2+assim**2)/2.)
 10   continue
      end
```

Функция зависит от четырёх параметров: `A` — амплитуда, `pike` — местоположение пика, `sigma` — ширины распределения в процентах, `assim` — асимметрии.

```
#создаём вектор параметров с начальными значениями
PAW > ve/cre par(4) r 25. 1.4 5. 0.
```

```
# создаём вектор с минимально допустимыми значениями (на глазок)
PAW > ve/cre pmin(4) r 10. 1.3 1. -1.
# создаём вектор с максимально допустимыми значениями
PAW > ve/cre pmax(4) r 40. 1.5 10. 1.
# создаём вектор, для ошибок подгонки
PAW > ve/cre err(4) r
# просим PAW подогнать распределение теоретической функцией
# опции подгонки:
#      B - учитывать минимально/максимально допустимые значения
#      M - перейти интерактивную сессию Minuit
# M обычно не используется, так как действия PAW при подгонке
# по умолчанию вполне разумны
PAW > hi/fit 20 loggaus.for "BM" 4 par ! pmin pmax err
          ...
# задаём имена параметрам
 Minuit > name 1 A
 Minuit > name 2 pike
 Minuit > name 3 sigma
 Minuit > name 4 assim
# задаём метод минимизации (migrad, обычно, самый подходящий)
 Minuit > migrad
# просим попробовать улучшить подгонку (дольше, но чуть точнее)
 Minuit > improve
          ...
 Minuit > exit
          ...
#смотрим результаты подгонки
PAW > ve/print par
 PAR(1) = 35.4279
 PAR(2) = 1.4809
 PAR(3) = 4.30618
 PAR(4) = -0.999999
#смотрим ошибки
PAW > ve/print err
 ERR(1) = 3.61113
 ERR(2) = 0.00484378
 ERR(3) = 0.339507
 ERR(4) = 0.174973
#нарисовать гистограмму 20 ещё раз
# e - рисовать статистические ошибки в бинах
PAW > hi/plot 20 e
```

При подгонке этого распределения основной интерес представляло его ширина: sigma $= (4.3 \pm 0.3)\%$. Важно не только значение подгонки, но и оценка ошибки. На-

пример, результат для sigma отличается от того, что должно быть в идеале больше чем на десять ошибок — можно сделать вывод, что есть какая-то, даже не проблема, а «плюха».

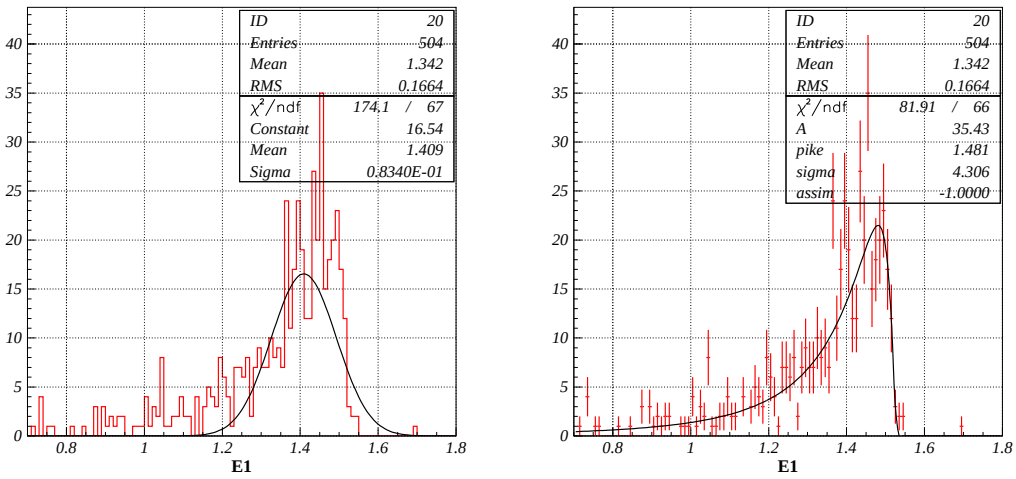


Рис. 2.7. Подгонка гистограммы

2.4 Функции

Функции так же являются базовым объектом для PAW. Для отрисовки одномерных функций используется команда FUNCTION/PLOT.

```
#включить логарифмический масштаб для оси Y
PAW > opt logy
#нарисовать одномерную функцию
PAW > fun/plot (sin(x)/x)**2+0.1 -10 10
#вернуться к линейному масштабу для оси Y
PAW > opt liny
```

Обратите внимание на инструкцию `opt` (`help GRAPHICS/OPTION`). Эта инструкция по своим функциям схожа с командой `set`, но в отличии от неё отвечает за организацию представления данных, а не за графическое оформление. Работу с двумерными функциями продемонстрируем на классическом фрактальном изображении имени Мандельброта. Создадим код на FORTRAN:

```
C Из официальной документации к PAW
C Файл mandel.for
      real function MANDEL(XP)
      dimension XP(2)
      data NMAX/30/
```

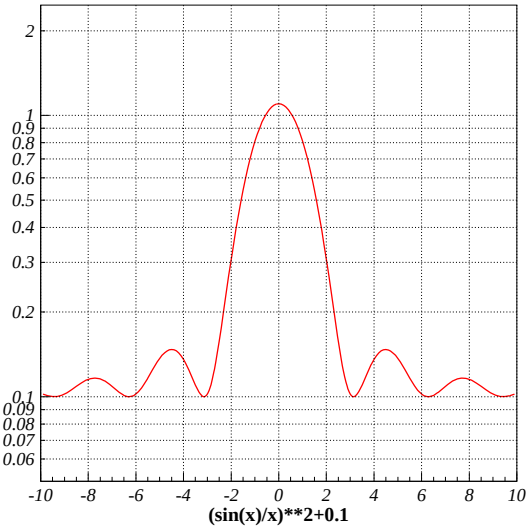


Рис. 2.8. Пример одномерной функции.

```
      x=XP(1)
      y=XP(2)
      xx=0.
      yy=0.
      do n=1,NMAX
         tt=xx*xx-yy*yy+x
         yy=2.*xx*yy+y
          xx=tt
         if (4..lt.xx*xx+yy*yy) goto 1
      enddo
1     MANDEL=FLOAT(n)/FLOAT(NMAX)
      end
```

В случае двумерных функций проблема отображения стоит гораздо острее чем у одномерных. Двумерные функции для отображения преобразуются в гистограммы (`help fun2`)

```
# По результатам вычисления mandel.for создаём гистограмму 10
PAW > fun2 10 mandel.for 100 -2.4 .8 100 -1.2 1.2 ' '
# Выводим гистограмму 10 как контур
PAW > hi/pl 10 cont3
# Выводим гистограмму 10 как поверхность
PAW > hi/pl 10 surf4
```

Если разрешение не удовлетворяет, то можно создать гистограмму не 100x100, как в примере, а 1000x1000.

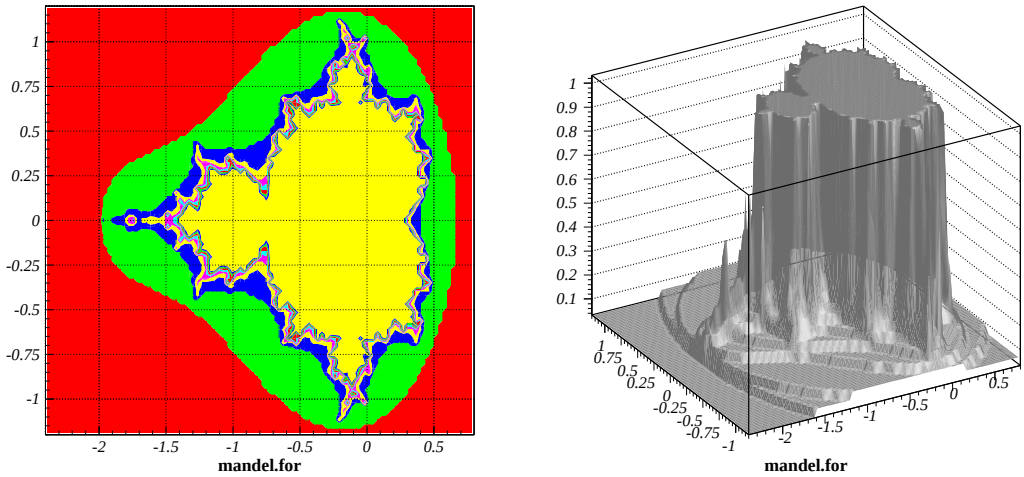


Рис. 2.9. Множество Мандельброта. Опции CONT3 и SURF4, соответственно.

2.5 Заключение

Объять необъятное совершенно не реально, особенно при лимите на объём текста. Официальная документация содержит около 500 страниц, причём, один алфавитный указатель занимает 17 страниц. Это матёрый программный продукт, которому уже двадцать лет. Этому пакету есть достойный приемник, правда, не лишённый недостатков, но об этом в следующий раз.