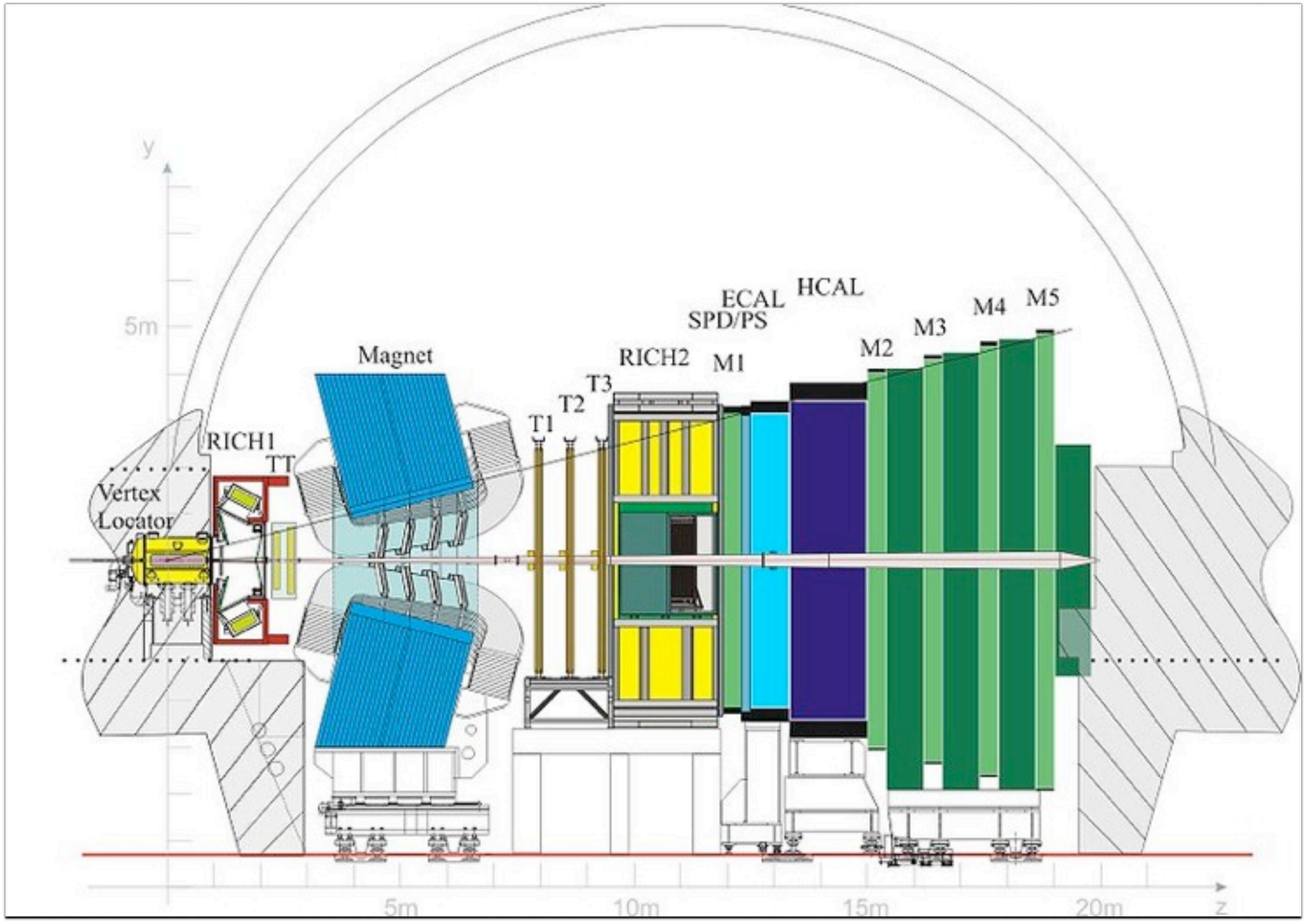
Dalitz plot Analisys of the $D^+ \rightarrow K^-K^+\pi^+$ decay in LHCb

- Rio+ -

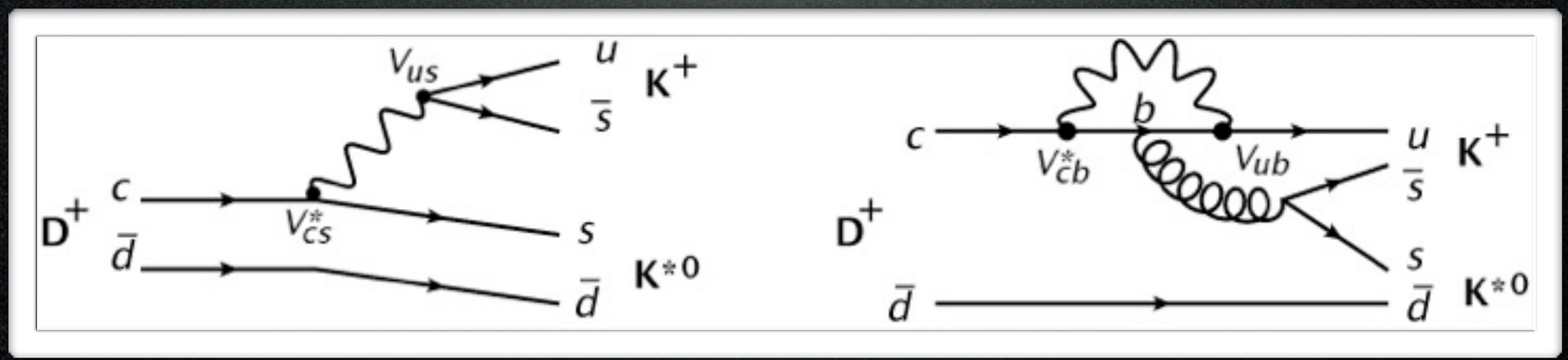
Sandra Amato, Carla Göbel, Josué Molina, Érica Polycarpo, Alberto Reis and Daniel Vieira

LHCb

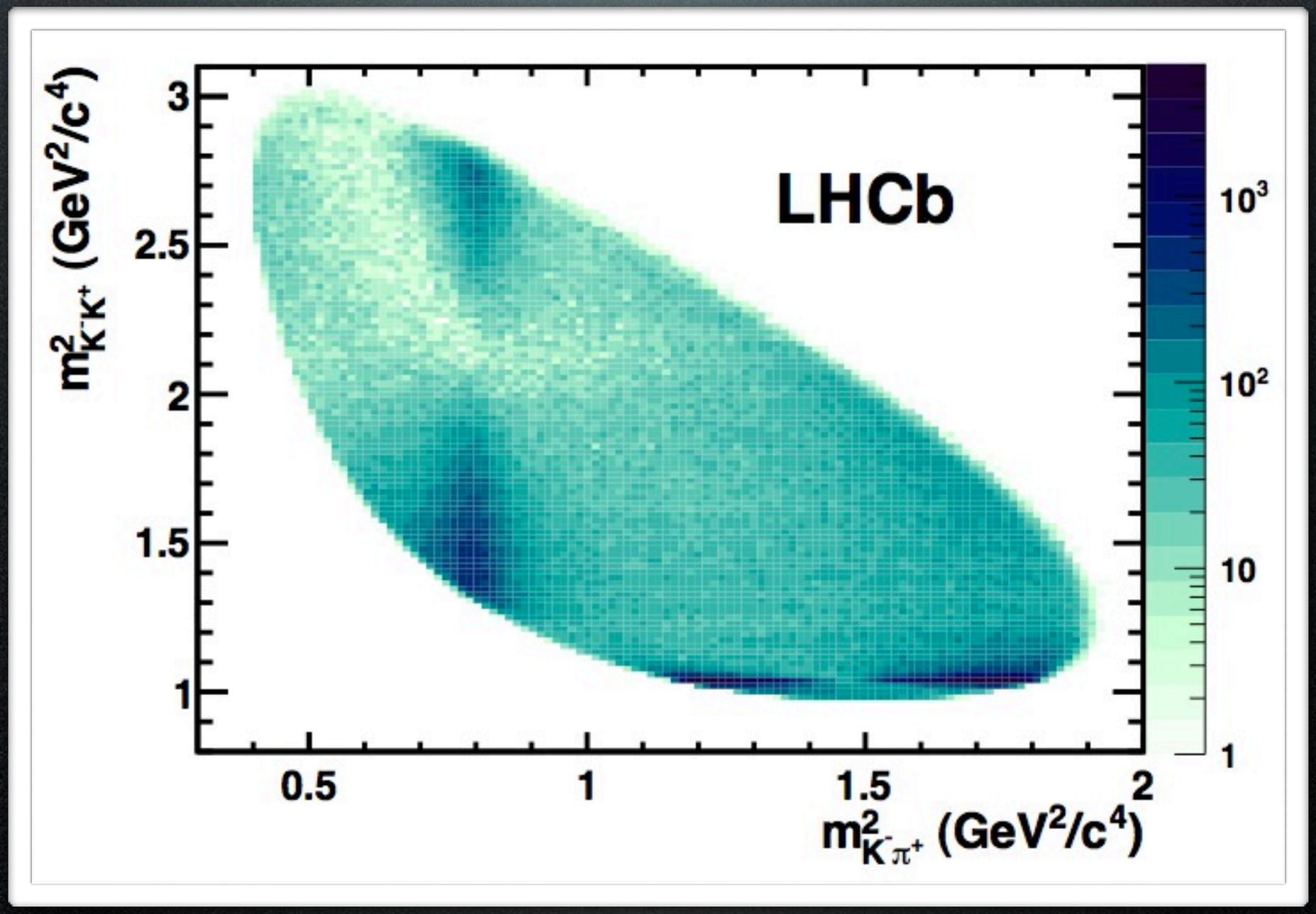

y
5m
Magnet
RICH1
TT
Vertex
Locator
T1
T2
T3
RICH2
M1
ECAL
HCAL
SPD/PS
M2
M3
M4
M5
5m
10m
15m
20m
z

CP Violation in charm decays

- SM only predicts significant CPV in Cabibbo suppressed decays
- In 3-body decays, CPV effects may arise in specific phase space regions

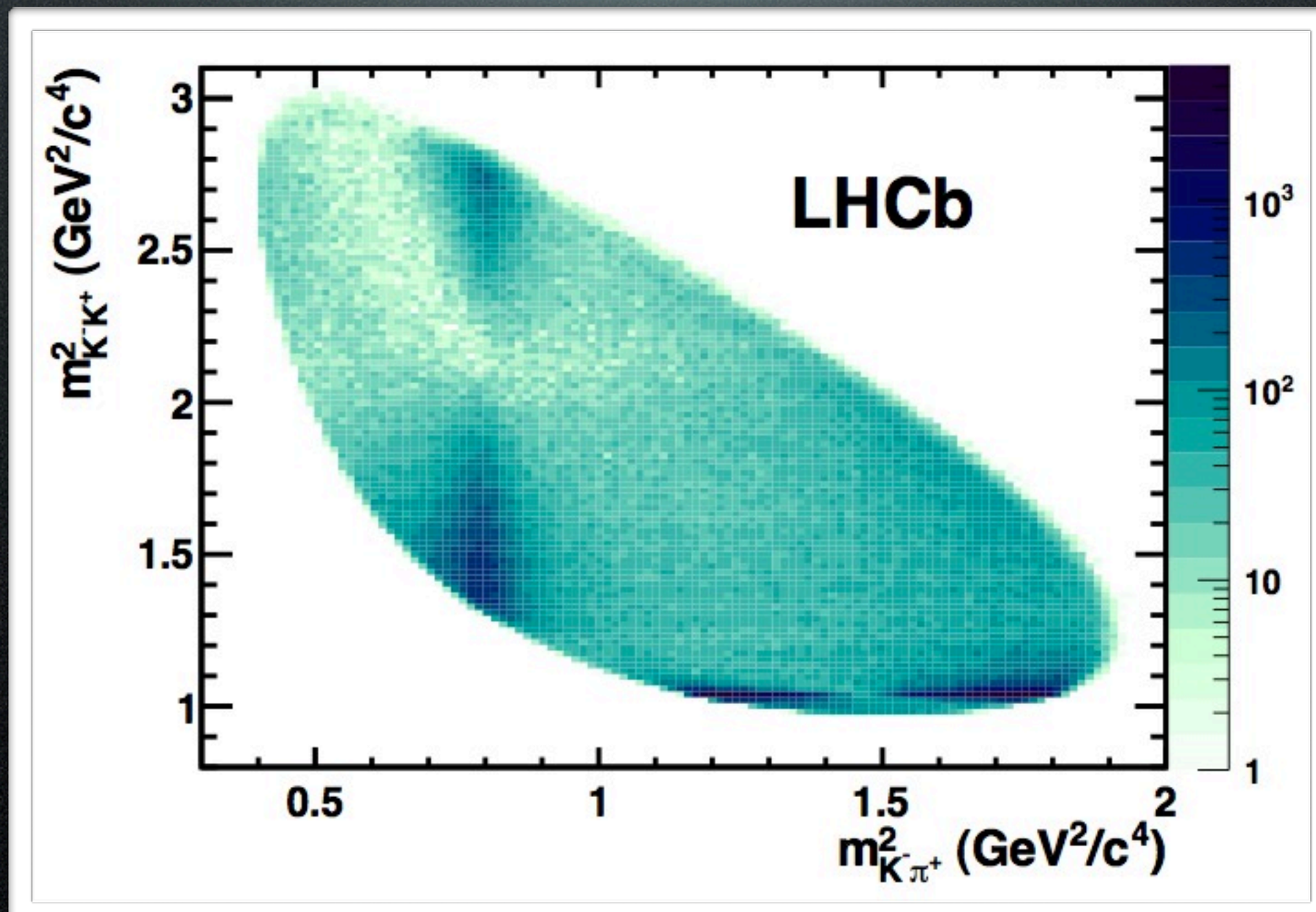


$D^+ \to K^-K^+\pi^+$ Dalitz plot

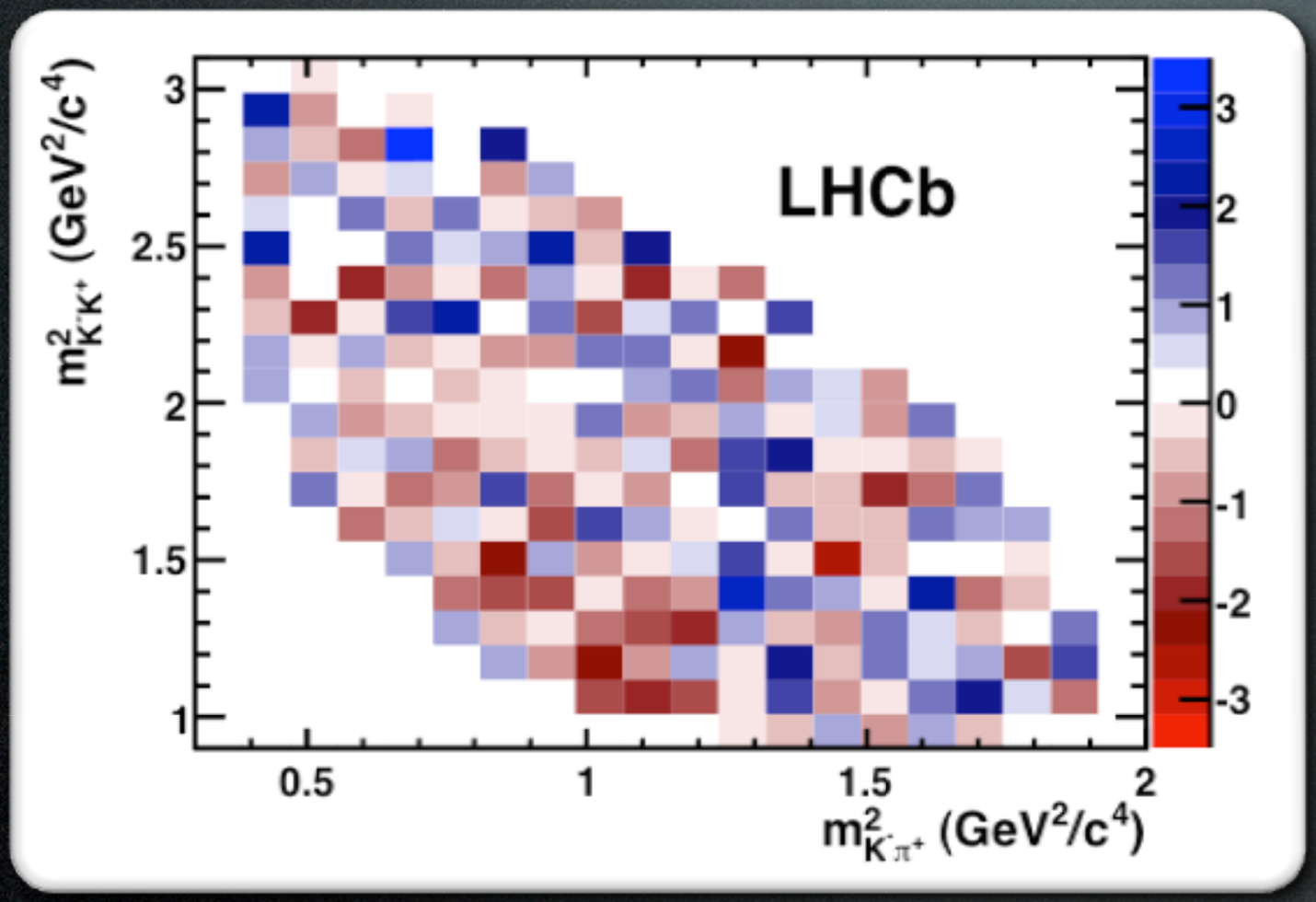


$D^+ \to K^-K^+\pi^+$ Dalitz plot

R. Aaij et al. [LHCb Collaboration], Phys. Rev. D. 84, 112008 (2011) [arXiv:1110.3970 [hep-ex]].

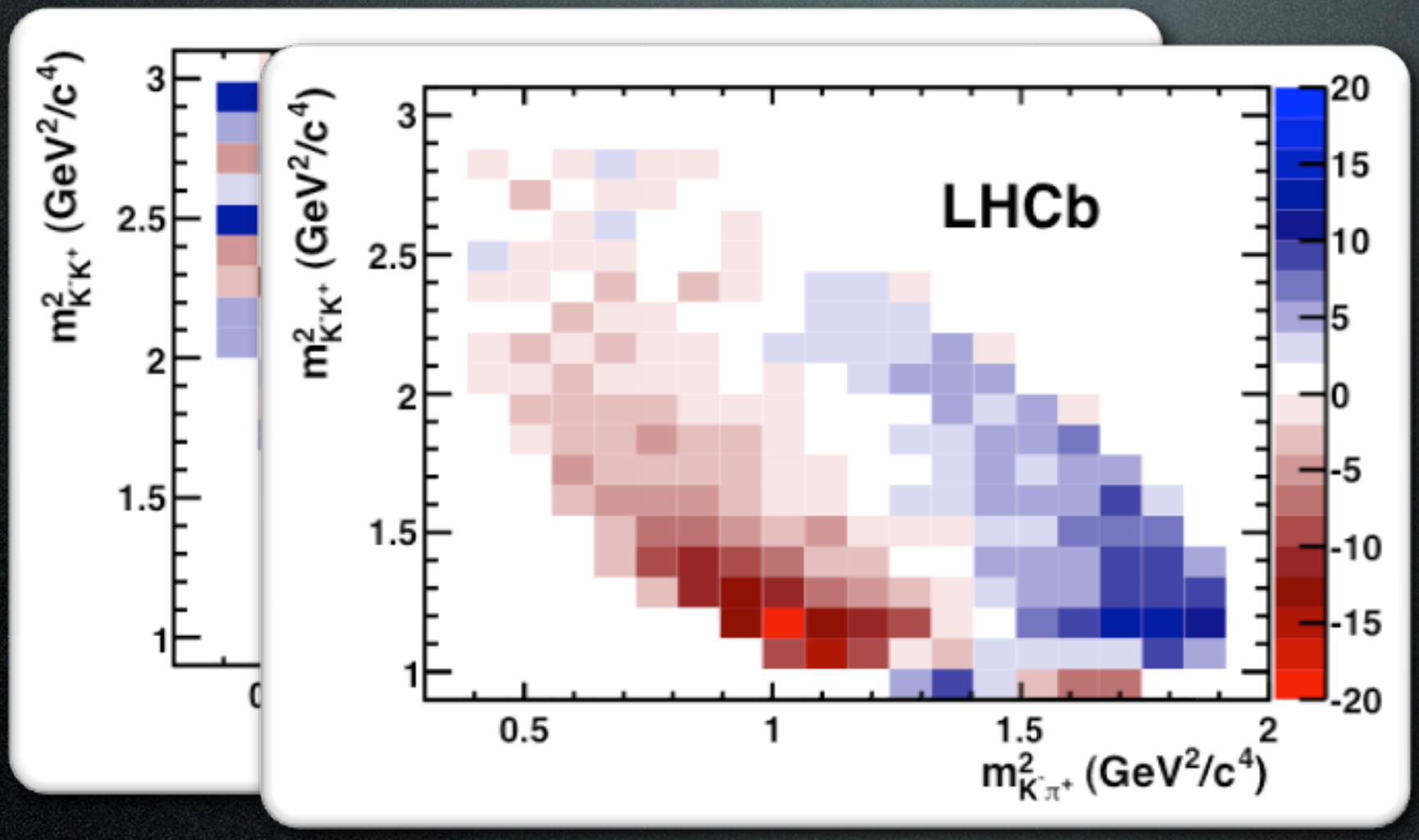


Anisotropy Method



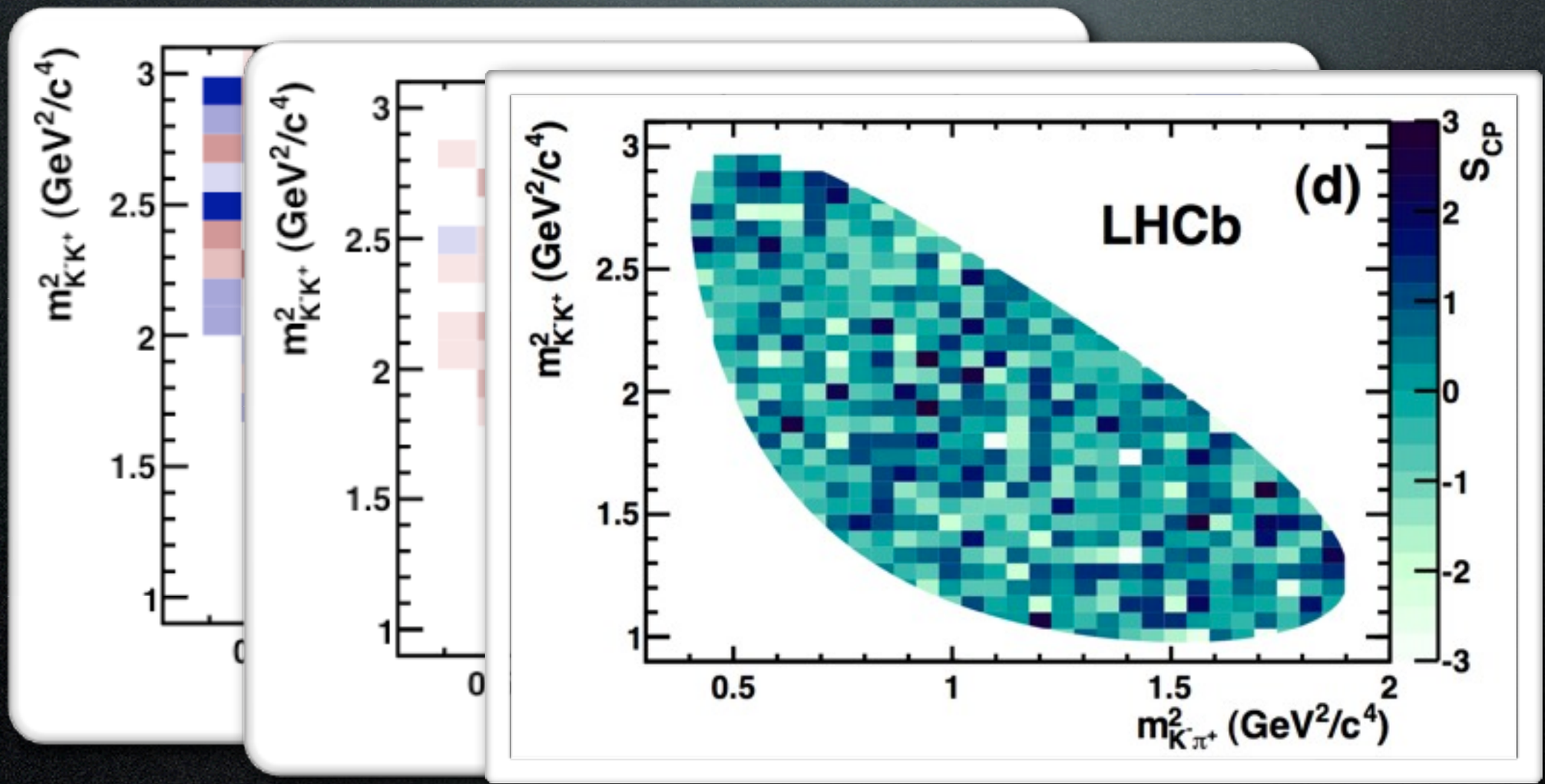
$$S_{CP}(i) = \frac{N_{D+}(i) - \alpha N_{D-}(i)}{\sqrt{N_{D+}(i) + \alpha^2 N_{D-}(i)}}$$

Anisotropy Method



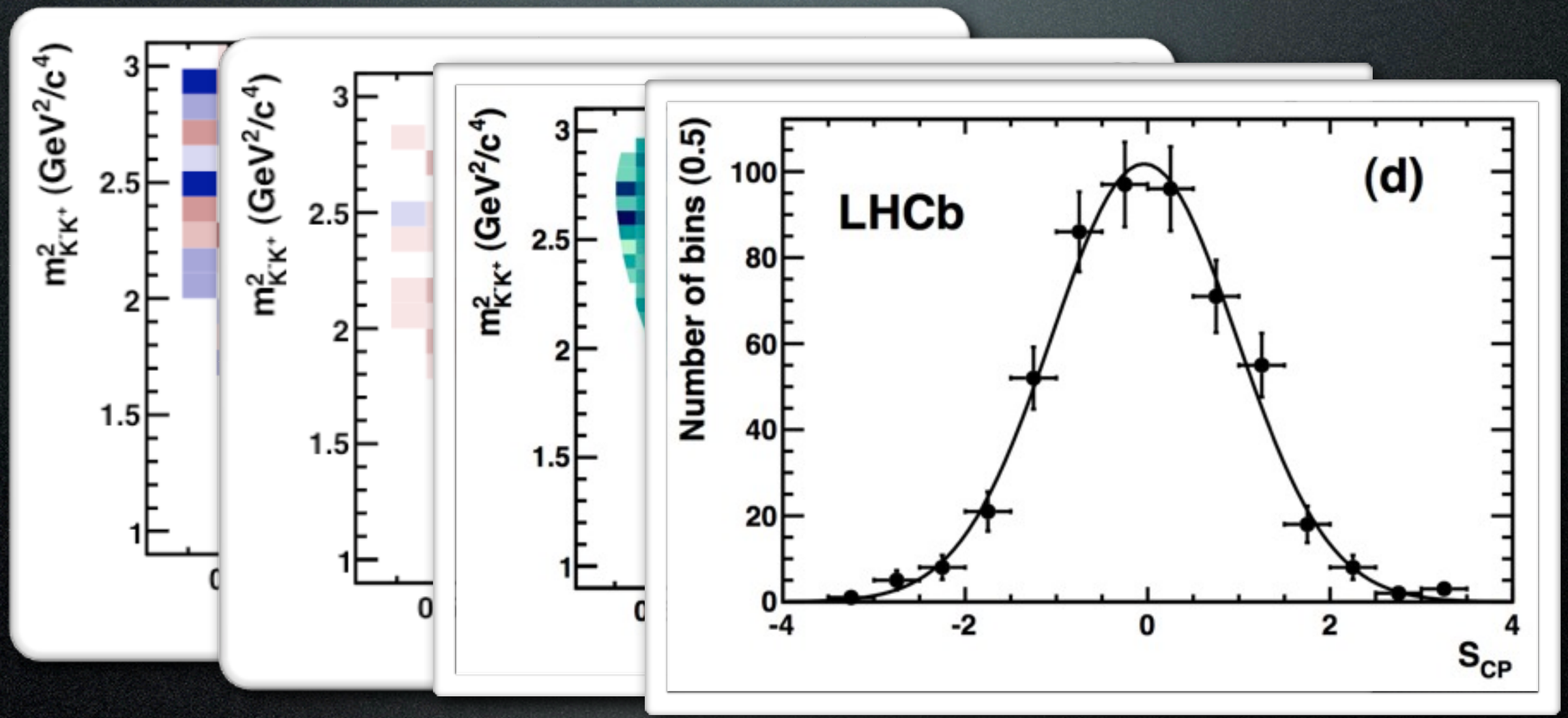
$$S_{CP}(i) = \frac{N_{D+}(i) - \alpha N_{D-}(i)}{\sqrt{N_{D+}(i) + \alpha^2 N_{D-}(i)}}$$

Anisotropy Method



$$S_{CP}(i) = \frac{N_{D^+}(i) - \alpha N_{D^-}(i)}{\sqrt{N_{D^+}(i) + \alpha^2 N_{D^-}(i)}}$$

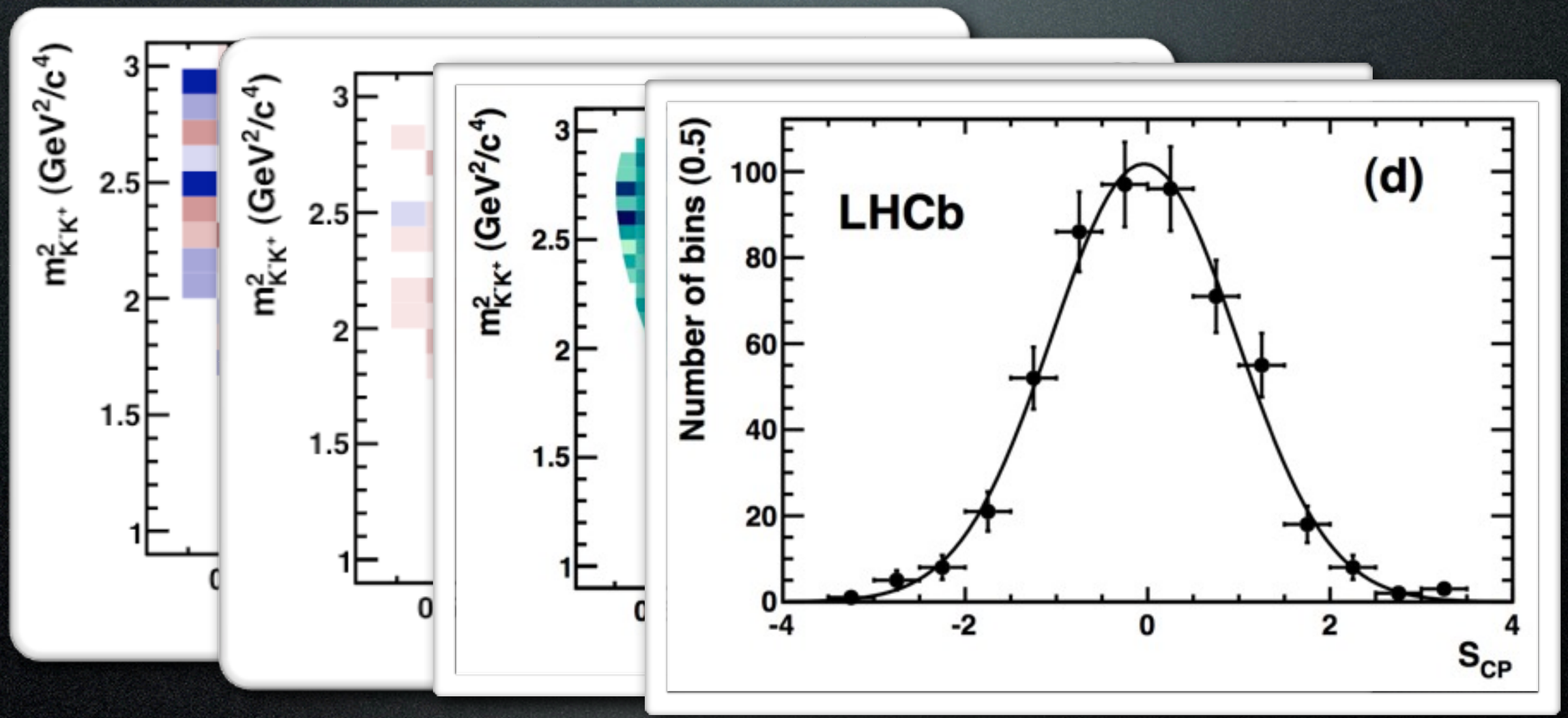
Anisotropy Method



$$S_{CP}(i) = \frac{N_{D^+}(i) - \alpha N_{D^-}(i)}{\sqrt{N_{D^+}(i) + \alpha^2 N_{D^-}(i)}}$$

Anisotropy Method

R. Aaij et al. [LHCb Collaboration], Phys. Rev. D. 84, 112008 (2011) [arXiv:1110.3970 [hep-ex]].



$$S_{CP}(i) = \frac{N_{D^+}(i) - \alpha N_{D^-}(i)}{\sqrt{N_{D^+}(i) + \alpha^2 N_{D^-}(i)}}$$

Dalitz plot analisys

- Most precise results obtained so far by CLEO-C collaboration
- Necessary to interpret eventual CPV signs
- Important to understand S-wave behaviour and low KK mass region contributions

Analisys tool

- We need a flexible tool which can accept different models.
- Fitting and toy MC genaration
- Easy to use and adapt (including and excluding resonances, choosing models, constants, etc)

Rio+

- Rio+ is based on the fortran code for Dalitz plot analysis used by FOCUS and E791, which was used in many publications.
- The code was built for D meson 3-body decays, but it can be used for any 3-body decay
- It can generate toy MC samples and make Dalitz plot fits.

Features

- Toy MC generation
- Fit - maximum likelihood method (ML)
- Fit - least squares
- Simulates also background
- Gaussian mass smearing
- Acceptance function
- Very simple and adaptable code

Features under construction

- PWA - binned amplitudes
- Masses and widths of resonances allowed as free parameters

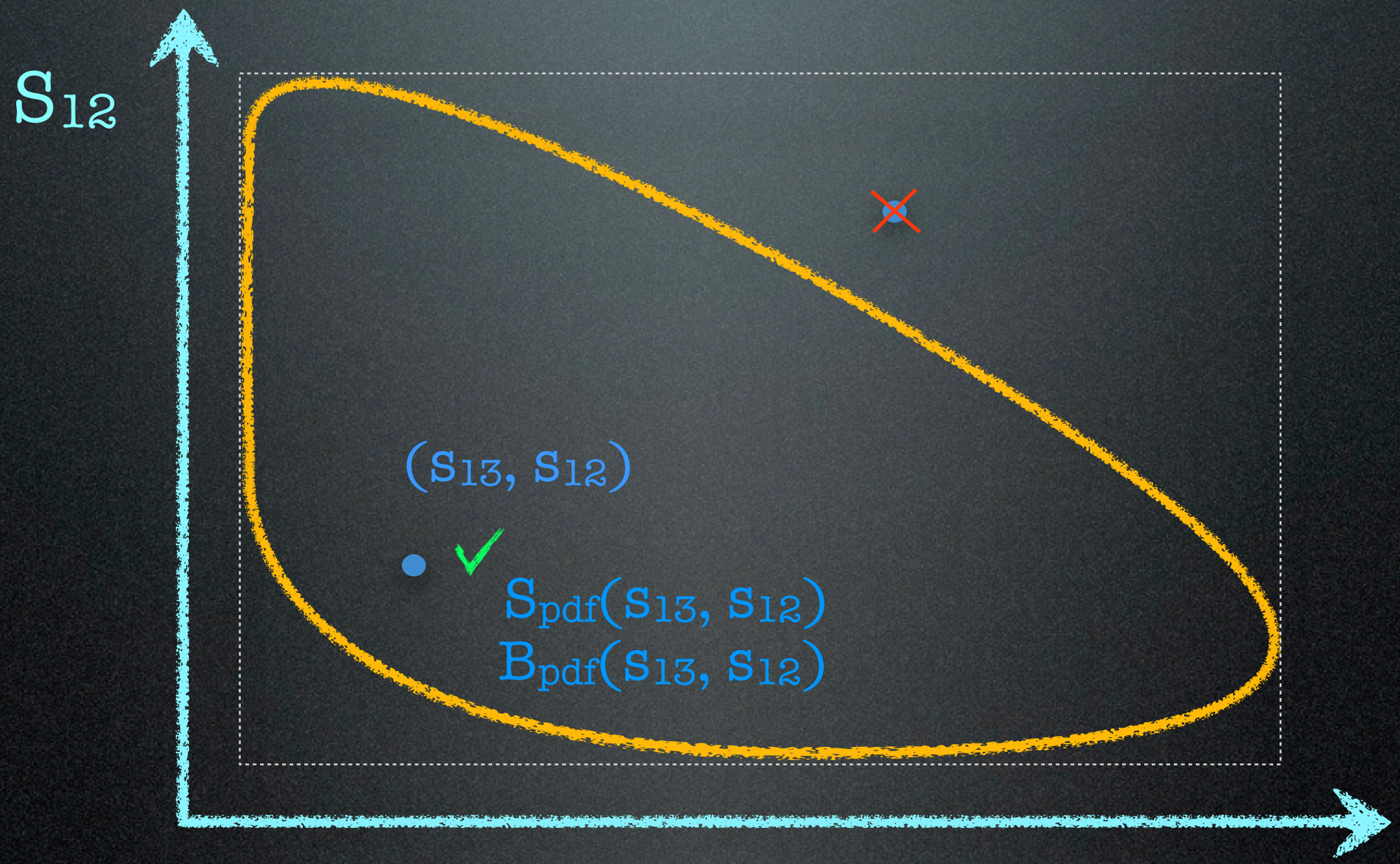
Integration Method

- Gauss Legendre quadrature

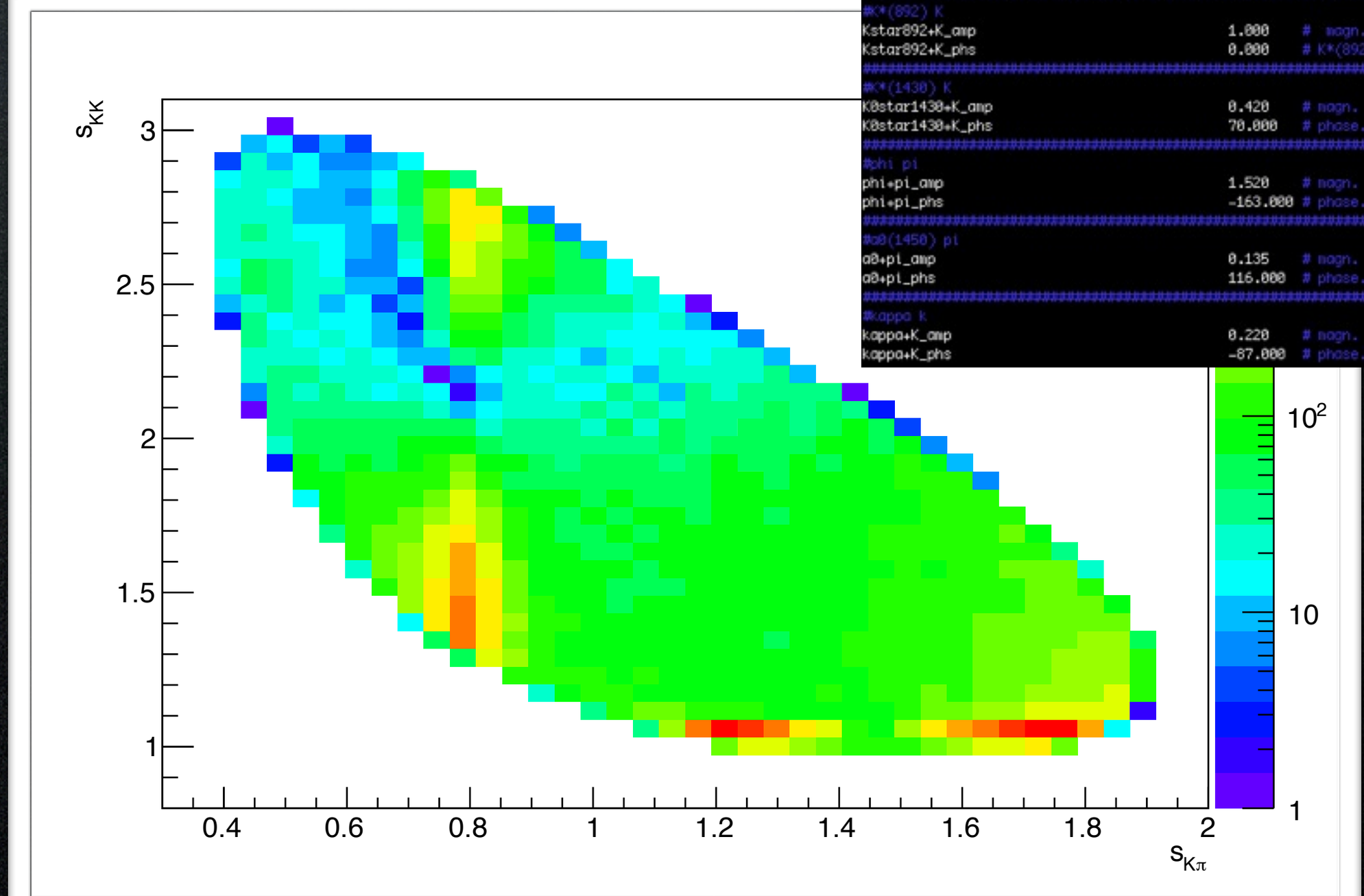
$$\int_a^b f(x)dx \approx \frac{b-a}{2}\sum_{i=1}^{n}\omega_i f(\frac{b-a}{2}z_i + \frac{a+b}{2})$$

$$\omega_i = \frac{2}{(1-x_i^2)\left[P_n'\left(x_i\right)\right]^2}$$

Toy MC Generator



Generated sample




```
#########################################################################
#K*(892) K
Kstar892+K_amp                    1.000     #  magn.
Kstar892+K_phs                    0.000     # K*(892) K phase.
#########################################################################
#K*(1430) K
K0star1430+K_amp                  0.420     # magn.
K0star1430+K_phs                  70.000    # phase.
#########################################################################
#phi pi
phi+pi_amp                        1.520     # magn.
phi+pi_phs                        -163.000  # phase.
#########################################################################
#a0(1450) pi
a0+pi_amp                         0.135     # magn.
a0+pi_phs                         116.000   # phase.
#########################################################################
#kappa k
kappa+K_amp                       0.220     # magn.
kappa+K_phs                       -87.000   # phase.
```

Fitter - ML

- Fit using maximum likelihood method

$$\mathcal{L} = \prod_{i=1}^{nevents} \left[\frac{S}{S+B} S_{PDF}(s_{13}^{(i)}, s_{12}^{(i)}) + \frac{B}{S+B} B_{PDF}(s_{13}^{(i)}, s_{12}^{(i)}) \right]$$

$$\int\int |\mathcal{A}(s_{12}, s_{13})|^2 \, \mathrm{d}s_{12}\mathrm{d}s_{13} = \int\int \sum_{i,j} a_i a_j e^{i\delta_i} e^{-i\delta_j} A_i(s_{12}, s_{13}) A_j^*(s_{12}, s_{13}) \mathrm{d}s_{12}\mathrm{d}s_{13}$$

$$= \sum_{i,j} a_i a_j e^{i\delta_i} e^{-i\delta_j} \int\int A_i(s_{12}, s_{13}) A_j^*(s_{12}, s_{13}) \mathrm{d}s_{12}\mathrm{d}s_{13}$$

Fitter - Least Squares

- A bit more complicated, but good for large samples
- Adaptive or uniform binning

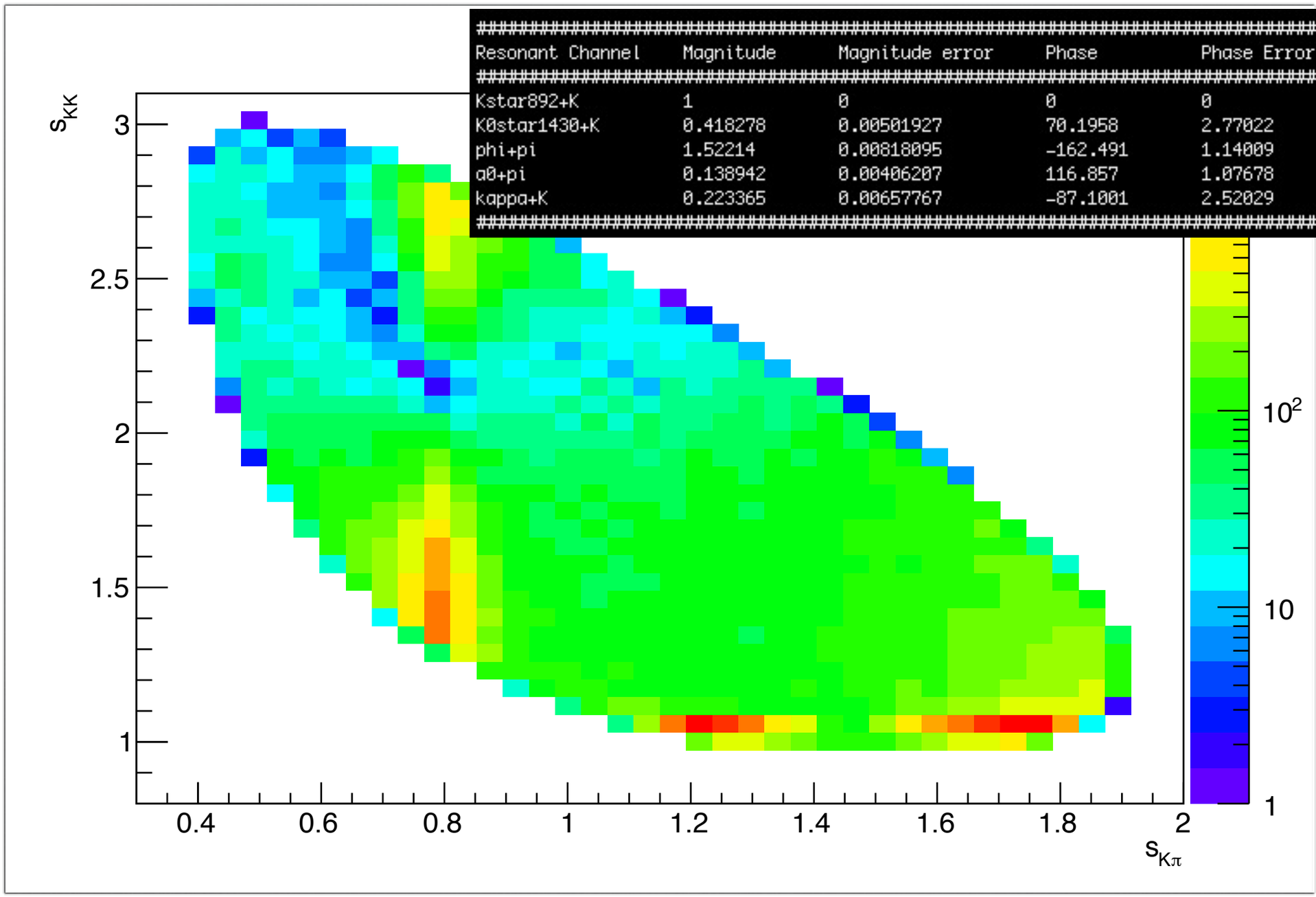
- Minimization:

$$\chi^2 = \sum_{i=1}^{nbins} \frac{(\text{observed}_i - \text{expected}_i)^2}{\sigma_i^2}$$

$$expected_i = N_{entries} \left(\frac{S}{S+B} \right) \frac{S_{PDF_i}}{Norm} + N_{entries} \left(\frac{B}{S+B} \right) B_{PDF_i}$$

Example Fit

Resonant Channel	Magnitude	Magnitude error	Phase	Phase Error
Kstar892+K	1	0	0	0
K0star1430+K	0.418278	0.00501927	70.1958	2.77022
phi+pi	1.52214	0.00818095	-162.491	1.14009
a0+pi	0.138942	0.00406207	116.857	1.07678
kappa+K	0.223365	0.00657767	-87.1001	2.52029



Usage

- Compilation through makefile in Log_Likelihood_Fitter folder: “make clean” to erase previous compilation and then “make”
- Runs with “./Log_Likelihood_Fitter”
- Alternatively, one can run it inside root
- Compilation: .L Log_Likelihood_Fitter.h+
- run: Log_Likelihood_Fitter(“Input_Parameters.txt”)
- Requires .txt as input

```
iMac-do-Daniel:log_likelihood_fitter dvieira$ make clean
rm -rf *o Log_Likelihood_Fitter
iMac-do-Daniel:log_likelihood_fitter dvieira$ make
g++ -c -g -Wall `root-config --cflags` Log_Likelihood_Fitter.C -o Log_Likelihood_Fitter.o
g++ `root-config --glibs` -lMinuit Log_Likelihood_Fitter.o -o Log_Likelihood_Fitter
iMac-do-Daniel:log_likelihood_fitter dvieira$ ./Log_Likelihood_Fitter

#####################################################
#                                                   #
#         Welcome to Log_Likelihood_Fitter          #
#                                                   #
#                                                   #
#       Authors: Sandra Amato, Carla Gobel,         #
#         Érica Polycarpo, Danielle Tostes,         #
#          Alberto Reis and Daniel Vieira           #
#                                                   #
#            contact: dvieira@if.ufrj.br            #
#                                                   #
#                     Have fun!                     #
#                                                   #
#####################################################

Insert the input file name:
Input_Parameters.txt
```

```
#Insert the input file name (with .root)
input_ntuple_name                    ntuple_complete_gaussian_10%bkg_kkpi_0.root
#Insert the final state here. (0 = kkpi, 1 = kpipi, 2 = pipipi).
final_state                          0
#What seed should be used in the generator? (insert 0 for random seed)
seed                                 1
#Would you like the mass distribution to be a delta or a gaussian? (1 for gaussian and 0 for delta)
is_gaussian                          1
#Background mass distribution represented by a linear equation. Insert Parameters. WARNING: these parameters will be ignored in case delta mass distribution was chosen.
Bkg_par1                             3000 # Background mass distribution parameter 1
Bkg_par2                             2000 # Background mass distribution parameter 2
```

Input ntuple name

Integer associated with final state

Seed

Delta or gaussian distribution

Background distribution parameters

```
#K*(892) K
Kstar892+K_amp                          1.000       # magn.
Kstar892+K_amp_lower_limit              0           # lower limit for magn.
Kstar892+K_amp_upper_limit              10          # upper limit for magn.
Kstar892+K_amp_fix                      1           # Set as fix? (1 if yes and 0 if not)
Kstar892+K_phs                          0.000       # phase.
Kstar892+K_phs_lower_limit              -180        # lower limit for phase.
Kstar892+K_phs_upper_limit              180         # upper limit for phase.
Kstar892+K_phs_fix                      1           # Set as fix? (1 if yes and 0 if not)
#K*(1430) K
K0star1430+K_amp                        0.420       # magn.
K0star1430+K_amp_lower_limit            0           # lower limit for magn.
K0star1430+K_amp_upper_limit            10          # upper limit for magn.
K0star1430+K_amp_fix                    0           # Set as fix? (1 if yes and 0 if not)
K0star1430+K_phs                        70.000      # phase.
K0star1430+K_phs_lower_limit            -180        # lower limit for phase.
K0star1430+K_phs_upper_limit            180         # upper limit for phase.
K0star1430+K_phs_fix                    0           # Set as fix? (1 if yes and 0 if not)
#phi pi
phi+pi_amp                              1.520       # magn.
phi+pi_amp_lower_limit                  0           # lower limit for magn.
phi+pi_amp_upper_limit                  10          # upper limit for magn.
phi+pi_amp_fix                          0           # Set as fix? (1 if yes and 0 if not)
phi+pi_phs                              -163.000    # phase.
phi+pi_phs_lower_limit                  -180        # lower limit for phase.
phi+pi_phs_upper_limit                  180         # upper limit for phase.
phi+pi_phs_fix                          0           # Set as fix? (1 if yes and 0 if not)
#a0(1450) pi
a0+pi_amp                               0.135       # magn.
a0+pi_amp_lower_limit                   0           # lower limit for magn.
a0+pi_amp_upper_limit                   10          # upper limit for magn.
a0+pi_amp_fix                           0           # Set as fix? (1 if yes and 0 if not)
a0+pi_phs                               116.000     # phase.
a0+pi_phs_lower_limit                   -180        # lower limit for phase.
a0+pi_phs_upper_limit                   180         # upper limit for phase.
a0+pi_phs_fix                           0           # Set as fix? (1 if yes and 0 if not)
#kappa k
kappa+K_amp                             0.220       # magn.
kappa+K_amp_lower_limit                 0           # lower limit for magn.
kappa+K_amp_upper_limit                 10          # upper limit for magn.
kappa+K_amp_fix                         0           # Set as fix? (1 if yes and 0 if not)
kappa+K_phs                             -87.000     # phase.
kappa+K_phs_lower_limit
kappa+K_phs_upper_limit
kappa+K_phs_fix
```

Amplitudes and phases of each resonant channel, plus parameter limits and boolean to set parameter as fix

```
#Insert the coefficients for each background component. Comment, erase or insert fraction fixed at 0 for the undesired background components.

#Random phi + pi
Random_phi_fraction                     0.2     # fraction
Random_phi_fix                          1       # Set as fix? (1 if yes and 0 if not)
#Random K* + K
Random_Kstar_fraction                   0.2     # fraction
Random_Kstar_fix                        1       # Set as fix? (1 if yes and 0 if not)
#Combinatorial
COMBINATORIAL_fraction                  0.6     # fraction
COMBINATORIAL_fix                       1       # Set as fix? (1 if yes and 0 if not)

#Inform the background fraction

bkg_fraction                            0.1
```

Background coefficients and boolean to set parameter as fix

B/(S+B) ratio

If you want to go further

- Add your resonant channels: Edit Amplitudes.h using predefined functions (lineshapes, form factors and spin amplitudes)
- Acceptance correction: just insert acceptance function inside GenericFunctions.h
- Changing constants: modify Constants.h (masses, widths, etc)

Example: How to include a new Amplitude

Amplitudes.h

```
if (resonances.at(12) == 1) {
    //define constants in Constants.h
    m_0 = mass of the resonance;
    w_0 = width of the resonance;
    spin = spin of the resonance;

    //Breit-Wigner example. For a decay into daughters a, b, c, defines a intermediate state with Breit-Wigner distribution in the ab system.

    // Blatt-Weisskopf form factors

    fD = Form_Factor_Mother_Decay(spin, M, sab, mcsq, m_0);
    fR = Form_Factor_Resonance_Decay(spin, m_0, sab, masq, mbsq);

    // Mass-dependent width

    gamma = Gamma(spin, m_0, w_0, mab, masq, mbsq);

    // Angular distribution

    f_theta = Angular_Distribution(spin, M, ma, mb, mc, sab, sbc, sac);

    // Complex amplitude for D -> rho(770) PI+

    Real_factor(fD*fR*f_theta, 0);

    sig.at(size_resonances) = Real_factor*BW(s13, m_0, gamma);
    size_resonances++;
}
```

Add constants

Write your amplitude. A few functions already written to help

Add your amplitude

Constants.h

```
#define AVAILABLE_RESONANCES 12
#define AVAILABLE_BKG_COMPONENTS 3

 // Defines identification strings for resonant channels and background components

string const resonant_channel_string[AVAILABLE_RESONANCES] = {"Kstar892+K","K0star1430+K","phi+pi","a0+pi","kappa+K","rho770+pi","f0980+pi",
                                                              "f21270+pi","rho1450+pi","f0X+pi","sigma+pi","NR"};
string const bkg_component_string[AVAILABLE_BKG_COMPONENTS] = {"Random_phi","Random_Kstar","COMBINATORIAL"};
```

Don't forget

Identification string must be added in the correct order

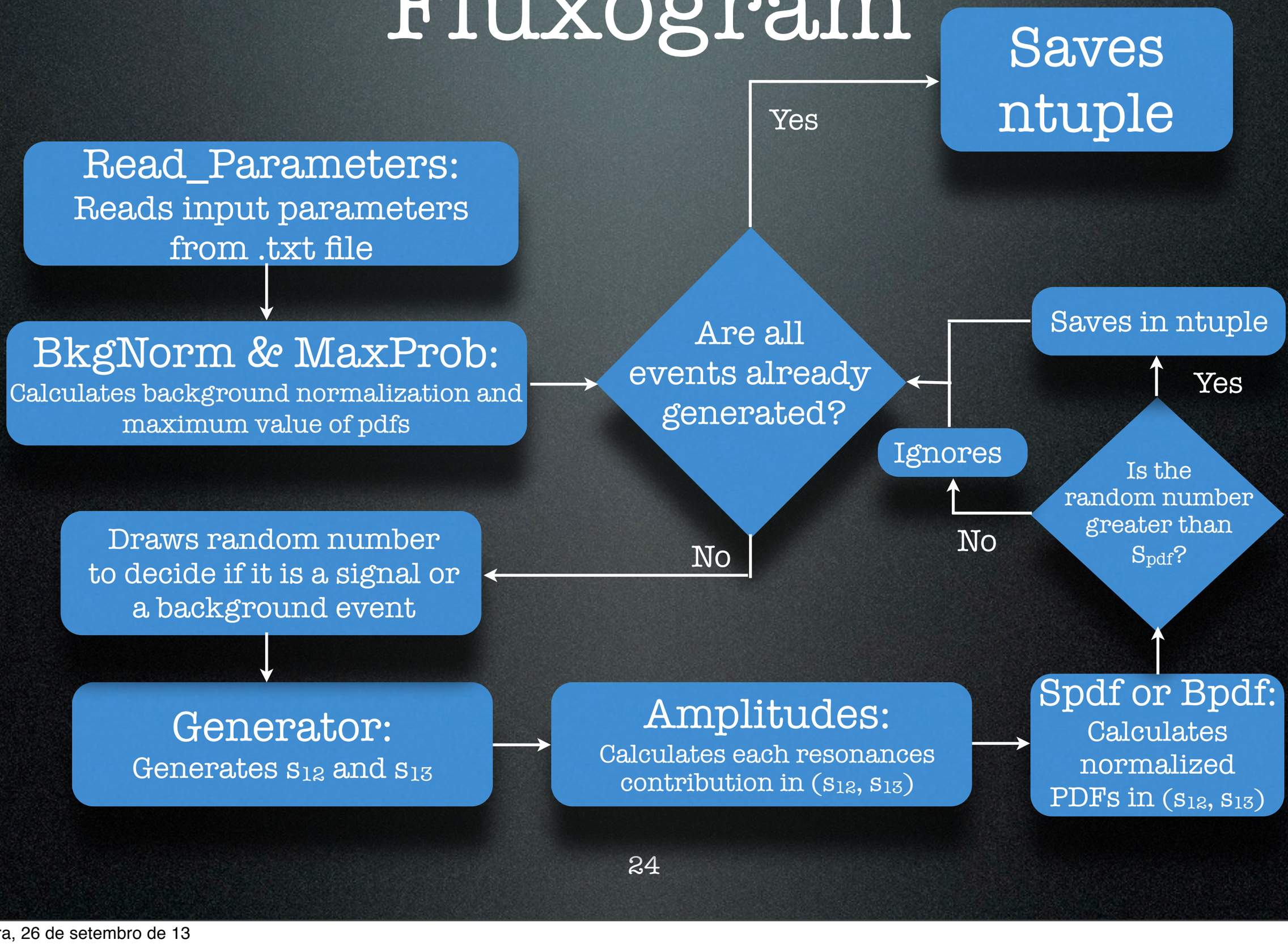
Conclusions

- Analisys in first steps. Rio+ development ongoing
- We are still adding some features to our tool (PWA binned amplitudes, free masses and widths of resonances, flexibilization)
- Easy to use, easy to modify, good place to start if you want to use something beyond the simple isobar model
- Soon officially released, available now at:

  http://www.if.ufrj.br/~dvieira/Rio+/

Backup

Toy MC Generator Fluxogram



Usage

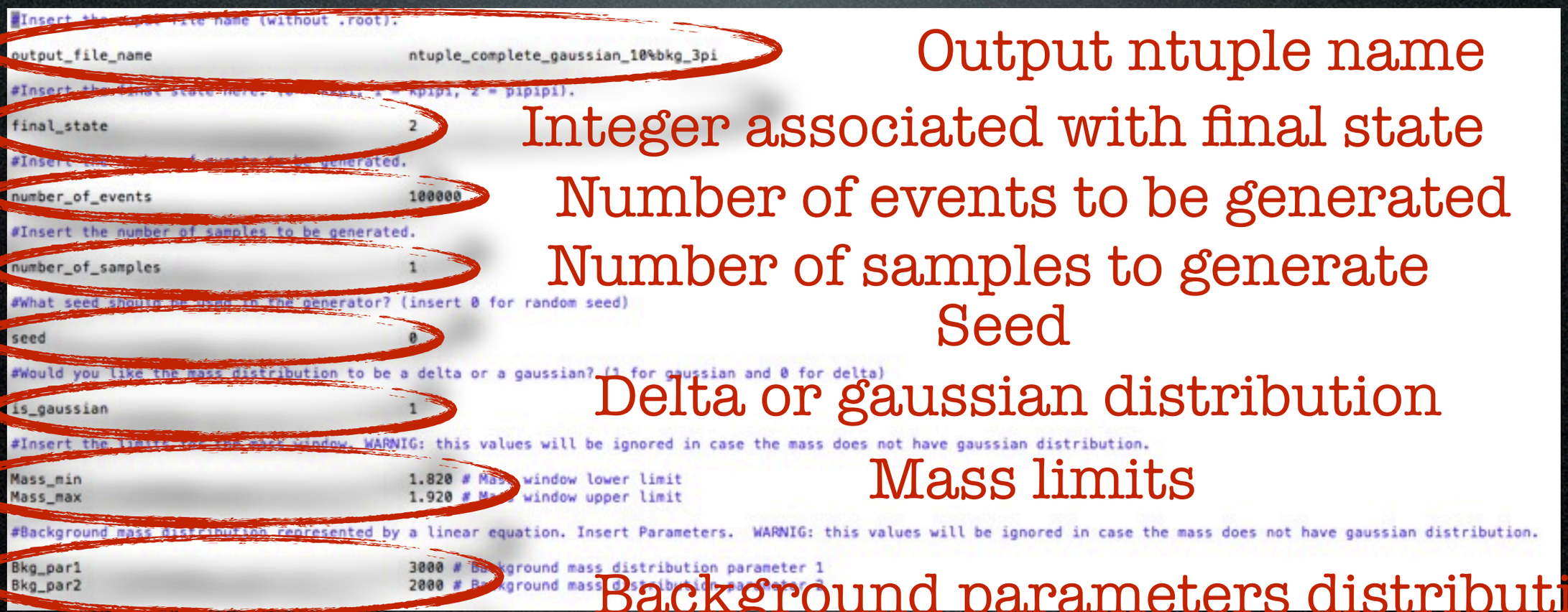
- Compilation through makefile in ToyMCGenerator folder: “make clean” to erase previous compilation and then “make”
- Runs with:

  “./ToyMCGenerator”
- Requires .txt as input

- Alternatively, one can run it inside root
- Compilation: .L ToyMCGenerator.h+
- run: ToyMCGenerator(“Input_Parameters.txt”)

```
iMac-do-Daniel:toyMCgenerator dvieira$ make clean
rm -rf *o ToyMCGenerator
iMac-do-Daniel:toyMCgenerator dvieira$ make
g++ -c -g -Wall `root-config --cflags` ToyMCGenerator.C -o ToyMCGenerator.o
g++ `root-config --glibs` ToyMCGenerator.o -o ToyMCGenerator
iMac-do-Daniel:toyMCgenerator dvieira$ ./ToyMCGenerator

#######################################################
#                                                     #
#             Welcome to ToyMCGenerator               #
#                                                     #
#                                                     #
#       Authors: Sandra Amato, Carla Gobel,           #
#         Érica Polycarpo, Danielle Tostes,           #
#          Alberto Reis and Daniel Vieira             #
#                                                     #
#            contact: dvieira@if.ufrj.br              #
#                                                     #
#                    Have fun!                        #
#                                                     #
#######################################################

Insert the input file name:
./Input_Parameters.txt
```

- Each parameter is identified by a string
- User may change parameters order and even omit the ones for which he/her doesn't care.
- Omitted parameters will receive a default value



Output ntuple name

Integer associated with final state

Number of events to be generated

Number of samples to generate

Seed

Delta or gaussian distribution

Mass limits

Background parameters distribution

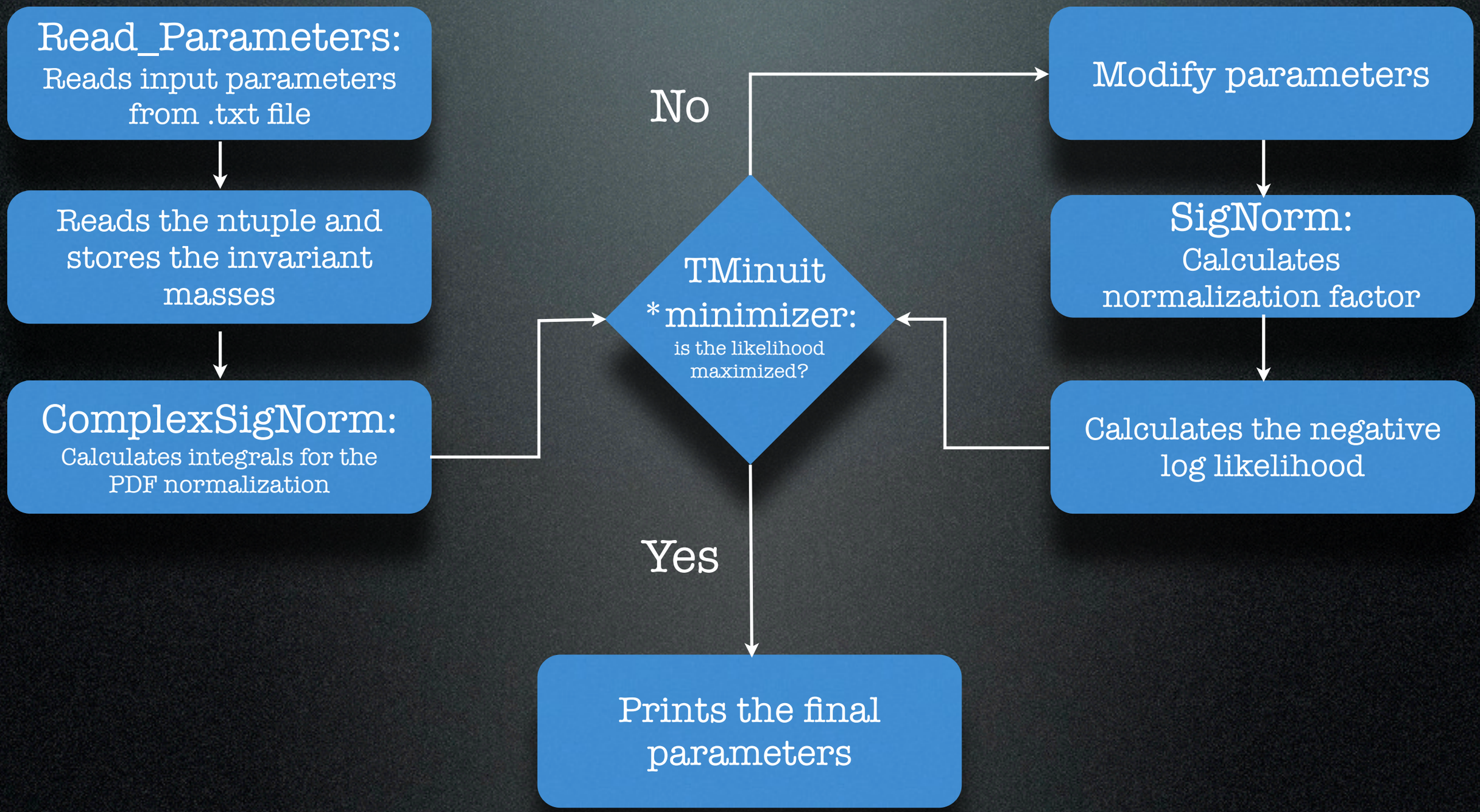
```
#rho(770) pi
rho770+pi_amp                                   0.800    # magn.
rho770+pi_phs                                   0.000    # phase
#f0(980) pi
f0980+pi_amp                                    0.080    # magn.
f0980+pi_phs                                    165.000  # phase
#f2(1270) pi
f21270+pi_amp                                   21.000   # magn.
f21270+pi_phs                                   57.000   # phase
#rho(1450) pi
rho1450+pi_amp                                  0.300    # magn.
rho1450+pi_phs                                  320.000  # phase
#f0(X) pi
f0X+pi_amp                                      0.080    # magn.
f0X+pi_phs                                      105.000  # phase
#sigma pi
sigma+pi_amp                                    0.300    # magn.
sigma+pi_phs                                    200.000  # phase
#NR
NR_amp                                          0.300    # magn.
NR_phs                                          57.000   # phase

#Insert the coefficients for each background component. Comment, erase or insert 0 fraction for the undesired background components.

#Random rho + pi
Random_rho_fraction                             0.4      # fraction
##Combinatorial
COMBINATORIAL_fraction                          0.6      # fraction

#Inform the background fraction*/

bkg_fraction                                    0.0
```

Amplitudes and phases of each resonant channel.

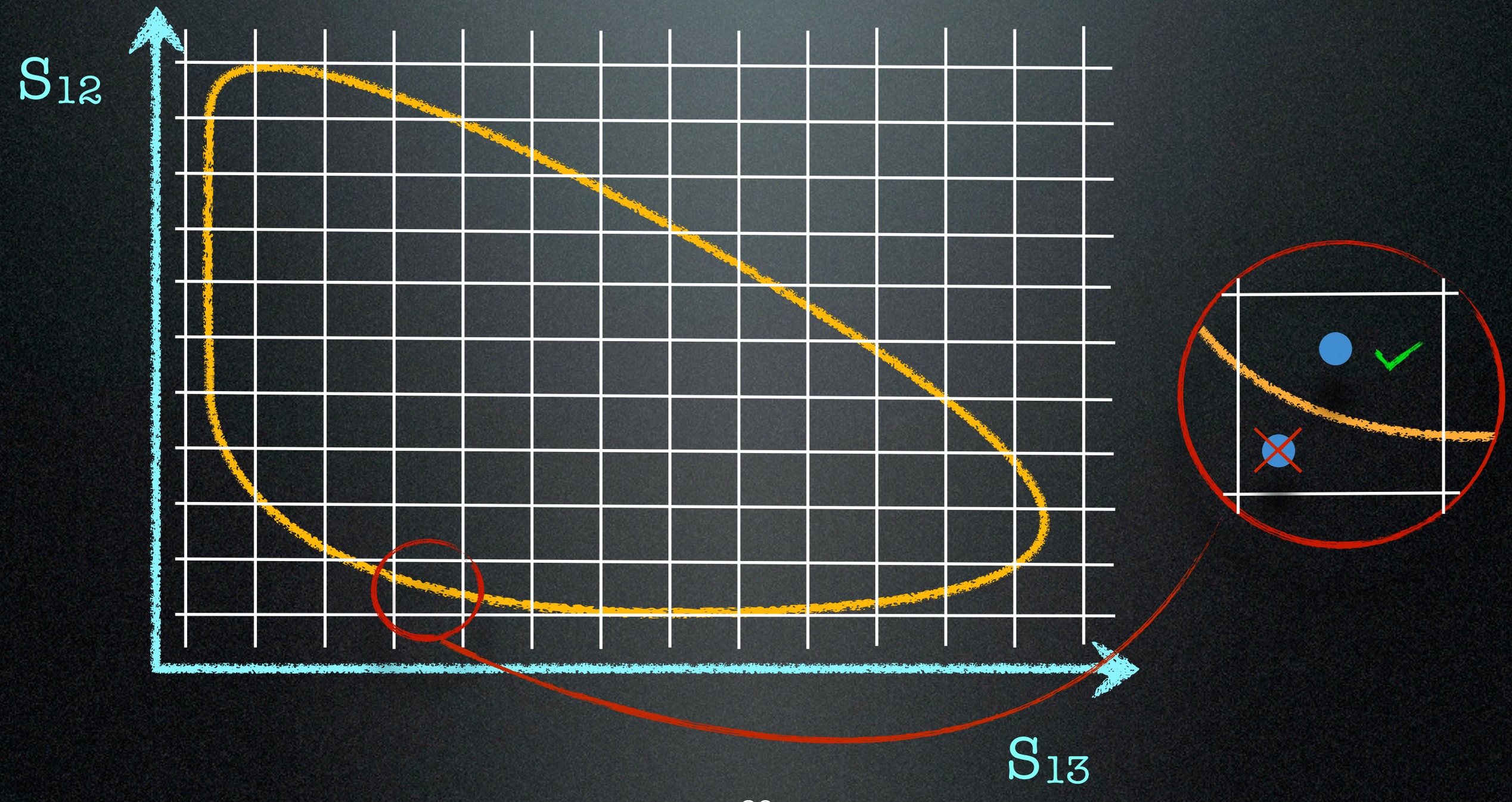
Background coefficients

B/(S+B) ratio

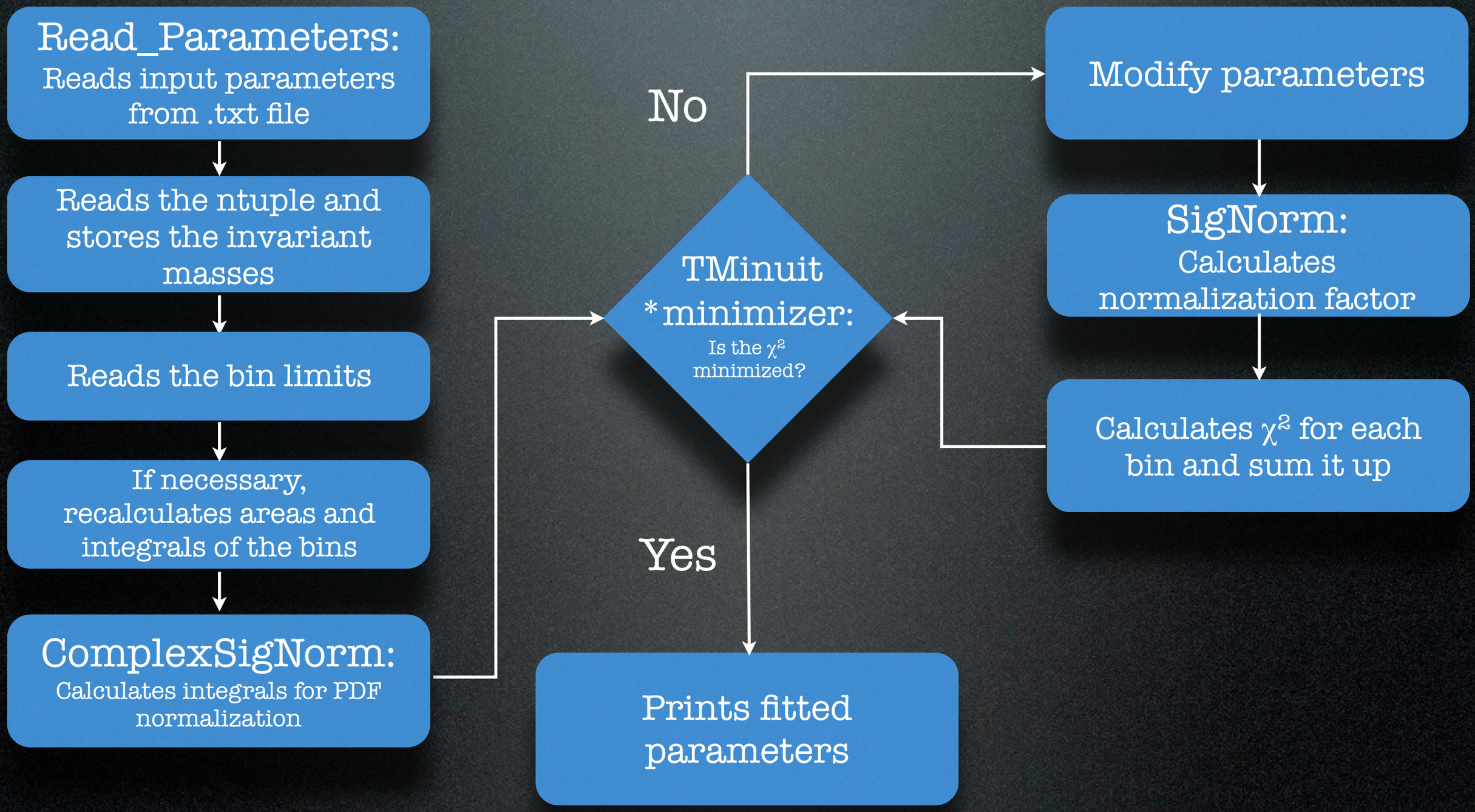
ML Fitter Fluxogram



Fitter - Least Squares



Least Squares Fitter Fluxogram



Usage

- Compilation through makefile in Chi2_Fitter folder: “make clean” to erase previous compilation and then “make”
- Runs with “./Chi2_Fitter”
- Requires .txt as input
- Alternatively, one can run it inside root
- Compilation: .L Chi2_Fitter.h+
- run: Chi2_Fitter(“Input_Parameters.txt”)

```
iMac-do-Daniel:chi2_fitter dvieira$ make clean
rm -rf *o Chi2_Fitter
iMac-do-Daniel:chi2_fitter dvieira$ make
g++ -c -g -Wall `root-config --cflags` Chi2_Fitter.C -o Chi2_Fitter.o
g++ `root-config --glibs` -lMinuit Chi2_Fitter.o -o Chi2_Fitter
iMac-do-Daniel:chi2_fitter dvieira$ ./Chi2_Fitter

#######################################################
#                                                     #
#              Welcome to Chi2_Fitter                 #
#                                                     #
#                                                     #
#       Authors: Sandra Amato, Carla Gobel,           #
#         Érica Polycarpo, Danielle Tostes,           #
#          Alberto Reis and Daniel Vieira             #
#                                                     #
#            contact: dvieira@if.ufrj.br              #
#                                                     #
#                    Have fun!                        #
#                                                     #
#######################################################

Insert the input file name:
Input_Parameters.txt
```

```
#Insert the input file name (with .root)

input_ntuple_name                       ntuple_complete_delta_0%bkg_kkpi_0.root

#Insert the final state (1 = kkpi, 2 = pipipi).

final_state                             0

#Do you want to use uniform or adaptive binning? (1 for adaptive and 0 for uniform)

adaptive                                0

##Insert the number of bins - WARNING: this will be ignored in case uniform binning was chosen

number_of_bins                          424

#Insert number of bins in s12 and s13 axis respectively - WARNING: this will be ignored in case adaptive binning was chosen

number_of_s12_Bins                      15   # Number of bins in s12 axis
number_of_s13_Bins                      15   # Number of bins in s13 axis

#Insert upper and lower limits for each axis - WARNING: this will be ignored in case adaptive binning was chosen

s12_lower_limit                         0.0  # s12 lower limit
s12_upper_limit                         3.5  # s12 upper limit
s13_lower_limit                         0.0  # s13 lower limit
s13_upper_limit                         2.5  # s13 upper limit

#What seed should be used in the generator? (insert 0 for random seed)

seed                                    0

#Is it necessary to recalculate the normalization integrals? (1 for yes and 0 for no)

ints_are_not_ready                      1

#Would you like the mass distribution to be a delta or a gaussian? (1 for gaussian and 0 for delta)

is_gaussian                             0

#Background mass distribution represented by a linear equation. Insert Parameters. WARNING: these parameters will be ignored in case delta mass distribution was chosen.

Bkg_par1                                3000 # Background mass distribution parameter 1
Bkg_par2                                2000 # Background mass distribution parameter 2
```

Input ntuple name

Integer associated with final state

Uniform or adaptive binning

Number of adaptive bins

Number of uniform bins per axis

Uniform bins histogram limits

Seed

Recalculate normalization?

Delta or gaussian distribution

Background distribution parameters

```
#K*(892) K
Kstar892+K_amp                          1.000       # magn.
Kstar892+K_amp_lower_limit              0           # lower limit for magn.
Kstar892+K_amp_upper_limit              10          # upper limit for magn.
Kstar892+K_amp_fix                      1           # Set as fix? (1 if yes and 0 if not)
Kstar892+K_phs                          0.000       # phase.
Kstar892+K_phs_lower_limit              -180        # lower limit for phase.
Kstar892+K_phs_upper_limit              180         # upper limit for phase.
Kstar892+K_phs_fix                      1           # Set as fix? (1 if yes and 0 if not)
#K*(1430) K
K0star1430+K_amp                        0.420       # magn.
K0star1430+K_amp_lower_limit            0           # lower limit for magn.
K0star1430+K_amp_upper_limit            10          # upper limit for magn.
K0star1430+K_amp_fix                    0           # Set as fix? (1 if yes and 0 if not)
K0star1430+K_phs                        70.000      # phase.
K0star1430+K_phs_lower_limit            -180        # lower limit for phase.
K0star1430+K_phs_upper_limit            180         # upper limit for phase.
K0star1430+K_phs_fix                    0           # Set as fix? (1 if yes and 0 if not)
#phi pi
phi+pi_amp                              1.520       # magn.
phi+pi_amp_lower_limit                  0           # lower limit for magn.
phi+pi_amp_upper_limit                  10          # upper limit for magn.
phi+pi_amp_fix                          0           # Set as fix? (1 if yes and 0 if not)
phi+pi_phs                              -163.000    # phase.
phi+pi_phs_lower_limit                  -180        # lower limit for phase.
phi+pi_phs_upper_limit                  180         # upper limit for phase.
phi+pi_phs_fix                          0           # Set as fix? (1 if yes and 0 if not)
#a0(1450) pi
a0+pi_amp                               0.135       # magn.
a0+pi_amp_lower_limit                   0           # lower limit for magn.
a0+pi_amp_upper_limit                   10          # upper limit for magn.
a0+pi_amp_fix                           0           # Set as fix? (1 if yes and 0 if not)
a0+pi_phs                               116.000     # phase.
a0+pi_phs_lower_limit                   -180        # lower limit for phase.
a0+pi_phs_upper_limit                   180         # upper limit for phase.
a0+pi_phs_fix                           0           # Set as fix? (1 if yes and 0 if not)
#kappa k
kappa+K_amp                             0.220       # magn.
kappa+K_amp_lower_limit                 0           # lower limit for magn.
kappa+K_amp_upper_limit                 10          # upper limit for magn.
kappa+K_amp_fix                         0           # Set as fix? (1 if yes and 0 if not)
kappa+K_phs                             -87.000     # phase.
kappa+K_phs_lower_limit
kappa+K_phs_fix
```

Amplitudes and phases of each resonant channel, plus parameter limits and boolean to set parameter as fix

```
#Insert the coefficients for each background component. Comment, erase or insert fraction fixed at 0 for the undesired background components.
#Random phi + pi
Random_phi_fraction                     0.2     # fraction
Random_phi_fix                          1       # Set as fix? (1 if yes and 0 if not)
#Random K* + K
Random_Kstar_fraction                   0.2     # fraction
Random_Kstar_fix                        1       # Set as fix? (1 if yes and 0 if not)
#Combinatorial
COMBINATORIAL_fraction                  0.6     # fraction
COMBINATORIAL_fix                       1       # Set as fix? (1 if yes and 0 if not)

#Inform the
bkg_fraction                            0.1
```

Background coefficients and boolean to set parameter as fix

B/(S+B) ratio