CERN-TH/99-98

How to Generate Four-Fermion Phase Space †

M. Skrzypek[a,b,‡] *and* **Z. Wąs**[a,b,‡]

[a] *Institute of Nuclear Physics, ul. Kawiory 26a, 30-055 Cracow, Poland,*
[b] *CERN, Theory Division, CH-1211 Geneva 23, Switzerland,*

Abstract

We present a scheme for integrating the matrix element of an arbitrary $e^+e^- \to f_1 f_2 \bar{f}_3 \bar{f}_4$ process over the complete four-fermion phase space, or its any part, by means of the Monte Carlo technique. The presented algorithm has been successfully implemented in the `KORALW` Monte Carlo code.

To be submitted to Comput. Phys. Commun.

---

† Work supported in part by Polish Government grants KBN 2P03B08414 and KBN 2P03B14715, and by the Maria Skłodowska-Curie Joint Fund II PAA/DOE-97-316.

‡ Part of this study was performed while the authors were at the *Inst. für Theoretische Physik, Karlsruhe Universität, Karlsruhe, Germany*, supported by a stipend within the EU grant no ERBCIPDCT940016.

CERN-TH/99-98
April 1999

1 Introduction

The measurement of the different observables related to the so-called four-fermion processes constitutes an important part of LEP2 physics. The most important group of observables is related to W-pair production and decay. It is not only the total cross section we are interested in, but also the wealth of different spin and angular asymmetries. This leads to rather sophisticated observables depending on multidimensional distributions that are also often strongly affected by experimental cut-offs. There are *nine* distinct W decay channels and, owing to small statistics of W-pair production, one is often bound to combine data of rather different experimental signatures to obtain statistically meaningful results. The situation of the forthcoming ZZ physics or so-called single-W physics is rather similar. In all cases, theoretical predictions, to be comparable with the data, have to be provided in the form of four-fermion Monte Carlo simulations.

Monte Carlo integration over four-body phase space[1] is a non-trivial problem. Why is it so? First of all, there is over a hundred different four-fermion final states allowed within the Standard Model. They differ by the flavours of final-state quarks and/or leptons. Each of these states, even at the Born level, is described by many, often over a hundred, Feynman graphs. For distinct four-fermion final states one finds thus diametrically different structures of singularities depending on the set of Feynman graphs describing the particular process. The singularities, or more precisely sharp peaks, have to be carefully tackled by the generators to assure their efficiency. They can affect the distribution in one or more variables of the particular parametrization of the phase space. As an example we can think of the t-channel singularities $1/t$, s-channel ones such as resonances, etc. These singularities occur in various parts of the phase space. Also, if they are present in more than one variable they can affect one another, either destructively or by increasing the degree of singularity of the second one. Finally one particular four-fermion process can have many distinct singular sub-processes, which need to be treated individually.

Of course, there are also some similarities and symmetries amongst processes as well as individual Feynman graphs. If properly used, they can somewhat reduce the size of the problem, but unfortunately its complexity remains intact. At the same time, if one departs from the Standard Model and introduces any new particles or interaction types, the size and complexity of the problem grows.

At the top of the above list of difficulties, there is also a strong cut-off dependence of the cross-section. This, in the case of electrons close to the beam directions, lead either to instability of the Monte Carlo integration or makes the generation ineffective since many events are generated outside the detector acceptance.

In the present paper we show a working solution to the above phase-space integration problem. It is based on the *multichannel* approach that is used in the `KORALW` Monte Carlo code [1,2]. Our aim was to enable the generation of all four-fermion $e^+e^- \to f_1 f_2 \bar{f}_3 \bar{f}_4$ final

[1] In reality the case is even more complicated, because of additional bremsstrahlung photons. In this paper, however, we restrict ourselves to pure Born-level phase space and *do not* discuss any issues related to photonic corrections. Let us note, only, that in the general case the Born-level phase space can form well-defined building-blocks of the whole algorithm.

states of the Standard Model over the full phase space or its sub-space, within reasonable CPU time.

At the time of the LEP2 Workshop [3], several dedicated four-fermion Monte Carlo codes were presented. As far as the integration method is concerned, some general strategies can be identified amongst these codes: the adaptive algorithms based on `VEGAS`-type routines [4–7], the `BASES-SPRING` algorithm [8, 9, 6], the importance-sampling technique [7,10,11] and the *multichannel* importance-sampling algorithms. In the latter group, one finds two approaches: of the `EXCALIBUR` code [12, 13], followed by `ERATO` [14], and of the `KORALW` [1, 2], the subject of this paper.

The major advantage of the *multichannel* approach is, in our opinion, that it gives almost complete control over the presampling process. The branches correspond to physical configurations in a transparent way. At the same time, since one can implement many branches, the structure of the generator can be very rich. No complicated dividing of the multidimensional phase space is necessary and no *potentially uncontrolled* automated optimization in the core of the algorithm needs to be performed (although some *auxiliary* optimization at the top levels can be useful).

The *multichannel* approach to few-body phase-space integration has been used already in the past. For instance, the `TAUOLA` [15–17] package for τ decays works in such a manner up to five-body decays. The `FERMISV` [18] code generates the NC-type four-fermion processes in that way also. As it happens with the Monte Carlo algorithms, it may be difficult to point to the exact source of the *multichannel* concept. Traces of it can be found in a number of other papers as well. Let us mention just a few of these: [19–23].

It is instructive to note that there are two "layers" in the structure of *multichannel* algorithms. The core of the algorithm is determined by the internal structure of the *channels* that contain the process-specific information on the actual generation algorithm. However, on the top of separate channels there is another, external layer, responsible for the "branching" process itself. Recently [24] this second step has been explained in detail.

The main advantage of the algorithm presented here is, in our opinion, the following. Out of various, already known as well as new, ideas, we show how to create in a systematic way, with simple building blocks, a universal and extendable presampler for four-body final states. Despite its simplicity, this presampler is shown to work effectively in an actual Monte Carlo code `KORALW`. The important advantage of the algorithm proposed here is its modularity and simplicity: all the branches are built out of two rather simple elementary pieces.

In this paper we present a complete description of the multichannel four-fermion Monte Carlo algorithm of the `KORALW` code. We start with the detailed description of the "external layer" – that is the branching process – which is quite universal. Using notation of ref. [24], we show how it works in the existing algorithm of `KORALW`. Starting from section 3, we discuss the "inner layer" – the actual structure of individual channels dedicated to generating the four-fermion phase space. We end the paper with a short summary.

2 "Multichannel" Master Formula

As a basis of the generation of the four-fermion phase space we take a multibranch type of Monte Carlo algorithm. In this section, let us summarize the details of the "external layer" of four-fermion phase space. Our goal is to calculate the following integral

$$\sigma = \int d\Phi(P,\underline{q})|M(P,\underline{q})|^2 \tag{1}$$

$$d\Phi(P,\underline{q}) = \frac{1}{2P^2}(2\pi)^4\delta^4\left(P-\sum_{k=1}^{r}q_k\right)\frac{1}{2^{r-1}}\prod_{k=1}^{r}\frac{1}{(2\pi)^3}\frac{d^3\vec{q_k}}{2E_k},\quad r=4, \tag{2}$$

where P denotes the sum of beam momenta, q_k denote the four-momenta of final particles, and $\underline{q}$ denotes collectively all q_k four-momenta. Note that eq. (2) includes all the normalization factors, including the flux factor $1/2P^2$.

Our strategy in solving eq. (1) is to simplify $|M|^2$ until it reaches the analytically calculable level. Then, we generate this simple distribution and reweight it to an exact $|M|^2$.

Let us then replace $|M|^2$ with the crude distribution $\tilde{\psi}_{CR}$, in which we introduce the structure of branches

$$d\Phi(P,\underline{q})\tilde{\psi}_{CR} = d\Phi(P,\underline{q})\sum_{i}^{N_{BR}}\tilde{p}_i\frac{\tilde{\psi}^i_{CR}(P,\underline{q})}{\mathcal{J}^i(P,\underline{q})}. \tag{3}$$

The summation goes over branches, with N_{BR} denoting the total number of branches. In this equation we introduced two set of functions: $\tilde{\psi}^i_{CR}$ and $\mathcal{J}^i$. The first represents the actual (crude) distributions to be generated over the phase space. It is in these functions that all the physical information is located. Their specific form is the main subject of this paper and will be analysed in detail in the next sections. Here we confine ourselves to the formal structure of the arrangement of branches. $\mathcal{J}^i$ is composed of the Jacobians of the change of variables from four-momenta to angular and invariant-mass variables (as defined in formulae below), which we anticipate in each channel i and of which we want to get rid (and reintroduce later by appropriate reweighting). They are isolated in eq. (3) for simplicity reasons. Finally, $\tilde{p}_i$ are some positive coefficients to be specified later. We change variables $(P,\underline{q})$ into angles and masses $(\cos\underline{\theta}^i,\underline{\phi}^i,\underline{s}^i)$ defined for each branch in different Lorentz frames. By $\cos\underline{\theta}^i,\underline{\phi}^i,\underline{s}^i$ we denote collectively all the angles and masses of a given branch i. The crude distribution becomes

$$\sum_{i}^{N_{BR}} d\Phi^i(\cos\underline{\theta}^i,\underline{\phi}^i,\underline{s}^i)\tilde{p}_i\tilde{\psi}^i_{CR}(\cos\underline{\theta}^i,\underline{\phi}^i,\underline{s}^i), \tag{4}$$

where we introduced the explicit form of the Jacobian $\mathcal{J}^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i)$ in new variables

$$\mathcal{J}^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i) = \frac{d\Phi(P, \underline{q})}{d\Phi^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i)} = \prod_{j=1}^{3} \lambda_j^i, \tag{5}$$

$$\lambda_j^i = \lambda(1, s_{2_j}^i/s_{1_j}^i, s_{3_j}^i/s_{1_j}^i) = \frac{1}{s_{1_j}^i}\sqrt{\left(s_{1_j}^i - s_{2_j}^i - s_{3_j}^i\right)^2 - 4s_{2_j}^i s_{3_j}^i}, \tag{6}$$

$$d\Phi^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i) = \frac{1}{4P^2}\frac{1}{(4\pi)^8} ds_1^i ds_2^i \prod_{j=1}^{3} d\cos\theta_j^i d\phi_j^i. \tag{7}$$

The mass-like variables s_1^i and s_2^i present in the integration element denote squares of the invariant masses of systems consisting of two or three outgoing fermions. They are present in the definition of the λ-factors of the $\mathcal{J}$ as well, but in this case $s_{j_k}^i$ can denote squares of masses of the outgoing fermions as well.

We have found that it is convenient to enlarge integration domains of the branches. This can be done for each branch in a different way. We do it by introducing Θ_i functions

$$\tilde{\psi}_{CR}^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i) = \psi_{CR}^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i)\Theta_i(\Omega^i) \tag{8}$$

and extending the integration areas $\Omega^i \to \Omega_{CR}^i$. The Θ_i enforce the exact phase-space limits. Now, by setting $\Theta_i \to 1$, we arrive at the final form of the crude distribution

$$\sum_{i}^{N_{BR}} d\Phi^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i)|_{\Omega_{CR}^i} \tilde{p}_i \psi_{CR}^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i). \tag{9}$$

We assume that eq. (9) is integrable analytically, with the result

$$\int\limits_{\Omega_{CR}^i} \tilde{p}_i d\Phi^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i)\psi_{CR}^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}^i) = \tilde{p}_i \mathcal{N}_{CR}^i. \tag{10}$$

The random choice of the branch to be used in the generation of a particular point in the phase space is performed with the help of probabilities P_i:

$$P_i = \frac{\tilde{p}_i \mathcal{N}_{CR}^i}{\sum\limits_{k}^{N_{BR}} \tilde{p}_k \mathcal{N}_{CR}^k}. \tag{11}$$

Later, following ref. [24], we generate points according to distributions ψ_{CR}^i of eq. (9). The last step to be done is to reintroduce all the simplifications in the form of the appropriate weights to complete the prescription on how to calculate the integral (1). For each channel i, the transition from ψ_{CR}^i of eq. (9) to $\tilde{\psi}_{CR}^i$ of eq. (4) can be achieved with the help of a simple step-like weight:

$$w^a(i) = \frac{\tilde{\psi}_{CR}^i}{\psi_{CR}^i} = \Theta_i. \tag{12}$$

The whole distribution of eq. (4) is then generated from ψ^i_{CR} with the help of the same weight $w^a(i)$, the argument i being the number of the chosen branch for a particular event. The distribution generated this way

$$\sum_i^{N_{BR}} P_i \frac{\psi^i_{CR}}{\tilde{p}_i \mathcal{N}^i_{CR}} w^a(i) = \frac{\sum_i^{N_{BR}} \tilde{p}_i \tilde{\psi}^i_{CR}}{\sum_i^{N_{BR}} \tilde{p}_i \mathcal{N}^i_{CR}} \tag{13}$$

is, up to a normalization factor, equal to that of eq. (4). The normalization (integral) of eq. (4) can be calculated as

$$\tilde{\mathcal{N}}_{CR} = \sum_i^{N_{BR}} \tilde{p}_i \tilde{\mathcal{N}}^i_{CR} = \sum_i^{N_{BR}} \tilde{p}_i \mathcal{N}^i_{CR} \langle w^a(i) \rangle_i \tag{14}$$

$$= \left(\sum_k^{N_{BR}} \tilde{p}_k \mathcal{N}^k_{CR} \right) \left(\sum_i^{N_{BR}} P_i \langle w^a(i) \rangle_i \right) = \left\langle {}_{P_i} \langle w^a(i) \rangle_i \right\rangle \sum_k^{N_{BR}} \tilde{p}_k \mathcal{N}^k_{CR}, \tag{15}$$

where we have made use of the fact that P_i defines a ratio of the number of events generated in channel i to the total number of events. With $\langle x \rangle_i$ we denote the average of x within the i-th channel and with $\langle {}_{P_i} \ldots \rangle$ we denote consecutive average with respect to the branches:

$$\left\langle {}_{P_i} \langle w(i, \ldots) \rangle_i \right\rangle = \sum_i^{N_{BR}} \frac{N_i}{N} \langle w(i, \ldots) \rangle_i \; ; \; \lim_{N \to \infty} \left\langle {}_{P_i} \langle w(i, \ldots) \rangle_i \right\rangle = \sum_i^{N_{BR}} P_i \langle w(i, \ldots) \rangle_i \,. \tag{16}$$

In practice, the external summation over channels is realized with the help of the Monte Carlo generation of the channel number with the probabilities P_i and, for every event, only that individual weight of the particular channel is calculated and used. Furthermore, we do not need to calculate $\left\langle {}_{P_i} \langle w(i, \ldots) \rangle_i \right\rangle$ as an explicit sum of averages $\langle w(i, \ldots) \rangle_i$, calculated and stored for each channel separately, as in eq. (16). Instead we calculate only *one* average over the whole sample of generated events of all channels:

$$\left\langle {}_{P_i} \langle w(i, \ldots) \rangle_i \right\rangle = \sum_i^{N_{BR}} \frac{N_i}{N} \frac{1}{N_i} \sum_{events \; in \; i}^{N_i} w(i, \ldots) = \frac{1}{N} \sum_{all \; events}^{N} w(i, \ldots) \equiv \langle w(i, \ldots) \rangle \,. \tag{17}$$

Here the argument i denotes that for the event generated with channel i we calculate a weight $w(i, \ldots)$. [2]

Including the fundamental matrix-element weight

$$w^b(P, \underline{q}) = \frac{|M(P, \underline{q})|^2}{\tilde{\psi}_{CR}(P, \underline{q})} \tag{18}$$

[2]Let us also note that in general, in our paper, we will omit the symbol $\lim_{N \to \infty}$.

and repeating the above steps we end up with the following formula for the integral (1)

$$\int d\Phi(P,\underline{q})|M(P,\underline{q})|^2 = \left\langle \tilde{w}(k,P,\underline{q})\right\rangle \sum_i^{N_{BR}} \tilde{p}_i \mathcal{N}^i_{CR}, \tag{19}$$

$$\tilde{w}(k,P,\underline{q}) = w^a(k)w^b(P,\underline{q}) = \frac{|M(P,\underline{q})|^2}{\tilde{\psi}_{CR}(P,\underline{q})}\Theta_k(\Omega_k). \tag{20}$$

Let us concentrate for a moment on the coefficients $\tilde{p}_i$. If we rewrite them in a more specific form as $\tilde{p}_i = p_i/\mathcal{N}^i_{CR}$, with the condition $\sum p_i = 1$, we find from eq. (11) that $P_i = p_i$, so that the p_i have a nice interpretation as the actual branching probabilities. The master equation (19) then evolves into

$$\int d\Phi(P,\underline{q})|M(P,\underline{q})|^2 = \left\langle w(k,P,\underline{q})\right\rangle, \tag{21}$$

$$w(k,P,\underline{q}) = |M(P,\underline{q})|^2 \left[\sum_i^{N_{BR}} \frac{p_i\tilde{\psi}^i_{CR}(P,\underline{q})}{\mathcal{N}^i_{CR}\mathcal{J}^i(P,\underline{q})}\right]^{-1} \Theta_k(\Omega_k). \tag{22}$$

Much as in ref. [24], we will now discuss a possible variation of the above algorithm. Turning back to eq. (3), we modify it by removing the Jacobians $\mathcal{J}_i$:

$$d\Phi(P,\underline{q})\tilde{\psi}_{CR} = d\Phi(P,\underline{q})\sum_i^{N_{BR}} \tilde{p}_i\tilde{\psi}^i_{CR}(P,\underline{q}) \tag{23}$$

and instead introduce them in eq. (4)

$$\sum_i^{N_{BR}} d\Phi^i(\cos\underline{\theta}^i,\underline{\phi}^i,\underline{s}^i)\tilde{p}_i\mathcal{J}^i(\cos\underline{\theta}^i,\underline{\phi}^i,\underline{s}^i)\tilde{\psi}^i_{CR}(\cos\underline{\theta}^i,\underline{\phi}^i,\underline{s}^i). \tag{24}$$

On the course to eq. (9) we simplify $\mathcal{J}^i_{CR} \to 1$ and, as a result, weights $w^a(i)$ get modified:

$$w^a(i) = \frac{\mathcal{J}^i\tilde{\psi}^i_{CR}}{\psi^i_{CR}} = \mathcal{J}^i\Theta_i. \tag{25}$$

Accordingly, eq. (21) becomes

$$\int d\Phi(P,\underline{q})|M(P,\underline{q})|^2 = \left\langle w(k,P,\underline{q})\right\rangle, \tag{26}$$

$$w(k,P,\underline{q}) = |M(P,\underline{q})|^2 \left[\sum_i^{N_{BR}} \frac{p_i\tilde{\psi}^i_{CR}(P,\underline{q})}{\mathcal{N}^i_{CR}}\right]^{-1} \mathcal{J}^k(P,\underline{q})\Theta_k(\Omega_k). \tag{27}$$

Note that here the Jacobians $\mathcal{J}$ are *not* summed over branches but calculated for the branch of a given event only. As the actual form of the Jacobians is rather simple, we should not expect a significant difference in the performance of the two algorithms.

Finally, let us remark that *both* of our algorithms are in fact of the "second" type in the classification of ref. [24]. In the first one (eq. (21)), we attribute zero weight to some events already inside branches in the form of trivial step functions Θ_i. In the second algorithm, weights $\Theta_i \mathcal{J}^i$ are continuous functions between zero and one.

In the language of measure theory the difference of the two algorithms (defined respectively by formulae (21) and (26)) lies in the choice of the basic measure on the phase-space manifold. In the second, more natural case, the Lorentz-invariant phase-space element takes this role.

3 The Branches

In the previous section we gave the complete general description of the "external layer" of the multichannel algorithm responsible for the alignment of separate branches. In this section we will present the construction of the actual branches in the case of four-fermion final states that we have developed for the `KORALW` code [2].

Let us begin with remark. Each branch ψ^i_{CR} of eqs. (4), (9), (24) is intended to describe a certain type of singularities that one encounters in the Feynman graphs, but there is no unique definition of the ψ^i_{CR} functions. They are dummy functions that cancel out in the final result. On the one hand, their role is to mimic as close as possible the complicated structure of singularities of the true matrix element, but on the other hand they should not be too complicated. First of all they *must* be analytically integrable over the generation area in order to be able to normalize the generator. Secondly, because of the complexity of the problem, it is, in our opinion, very useful for them to have a modular structure. This allows us to write the program and its documentation in a compact and transparent way, leaving room for easier modifications in the future. Finally, these simplifications obviously cannot go too far as the ψ^i_{CR} functions have to be quite universal and flexible to accommodate a variety of matrix-element singularities.

How do we fulfil in practice all these, partly contradictory, requirements? How do we construct the actual ψ^i_{CR} functions?

The crucial assumption that we make is the partial factorization of ψ^i_{CR} with respect to the angular and mass variables (we skip the branch index i for the next two sub-sections):

$$\psi_{CR}(\cos\underline{\theta}, \underline{\phi}, \underline{s}) = f(\underline{s}) g(\cos\underline{\theta}, \underline{\phi}, \underline{s}). \tag{28}$$

We request that $g(\cos\underline{\theta}, \underline{\phi}, \underline{s})$, upon integrating over angular variables, becomes independent of masses $\underline{s}$

$$\int d\cos\underline{\theta} d\underline{\phi} g(\cos\underline{\theta}, \underline{\phi}, \underline{s}) = \text{const}. \tag{29}$$

Finally, we assume that masses $\underline{s}$ will always be generated first, before the angles $\underline{\theta}, \underline{\phi}$.

With all these simplifications, one may worry that we are too restrictive. For example, it may happen that, for certain configurations of singularities, it would be natural to reverse the order of generation – angles first and then masses, or even, further, that one

should use some combination of these variables as more "physical". This approach would, however, betray to some extent the modularity of the program: it is always up to the author of an algorithm to decide where to draw the line. Our priority was set on simplicity and modularity of the algorithm. We believe we achieved those to a large extent. The solution presented here may look trivial and simplistic, but the point is that the efficiency of the generation is sufficient for our purposes, and we do not need to pay the price of formulas several pages long, which may be necessary e.g. in the algorithm of refs. [18,13].

Now we can proceed to the details of the mass and angular distributions f and g.

3.1 Mass Distributions

As explained above we intend to generate s_1, s_2 variables first, and then the angles. Generically there are two possible ways of aligning the two mass variables: a "chain"-like and a "fork"-like, see Fig. 1.

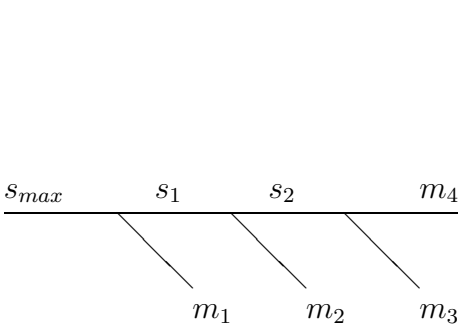

Fig. 1a: "Chain"

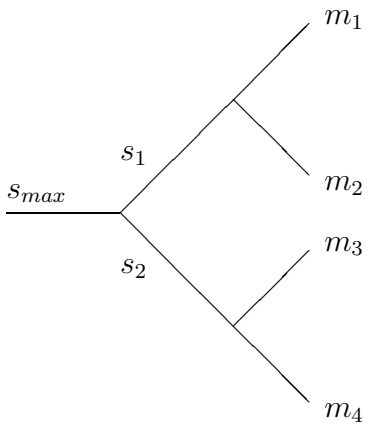

Fig. 1b: "Fork"

For each of these configurations we need to give the exact phase-space limits $\Theta(\Omega)$ corresponding to eq. (8). Then we define an extension $\Omega \to \Omega_{CR}$ of these limits that enable the analytic integration of the distributions to simple results, allowing numerically fast normalization to unity.

For the two cases defined above we continue independently:

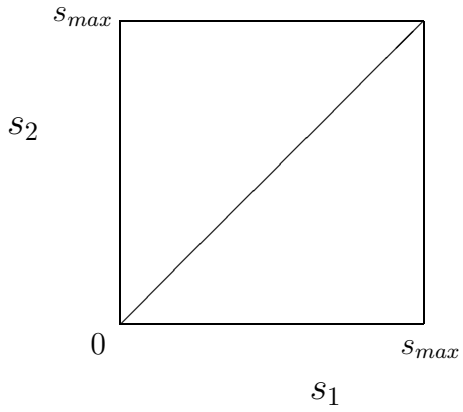

Fig. 2a: "Chain"-type phase space

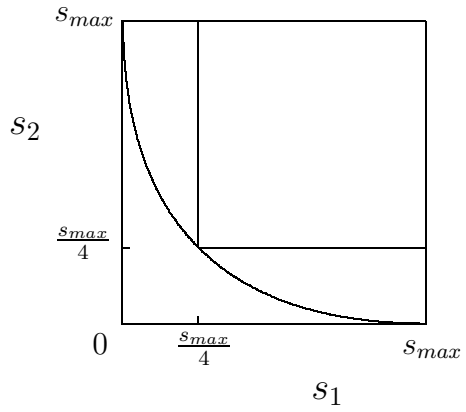

Fig. 2b: "Fork"-type phase space

- The "chain".
  The exact phase-space limits $\Theta(\Omega^C)$ of eq. (8) in this case are given by:

$$\sqrt{s_{max}} \geq \sqrt{s_1} + m_1, \quad \sqrt{s_1} \geq \sqrt{s_2} + m_2, \quad \sqrt{s_2} \geq m_3 + m_4. \tag{30}$$

  By dropping masses, the above domain Ω^C extends to the simpler one, Ω^C_{CR} of Fig 2a:

$$s_{max} \geq s_1 \geq s_2 \geq 0. \tag{31}$$

  Next, we assume that s_1 and s_2 distributions are identical and that the function $f(s_1, s_2)$ factorizes

$$f(s_1, s_2) = \frac{1}{\mathcal{F}(s_{max})} f_1(s_1) f_1(s_2). \tag{32}$$

  This is a somewhat restrictive assumption, but as there will be practically no restrictions on the form of a single distribution $f_1(s_1)$, we can always use a sufficiently general form of it. It costs some efficiency, but this is compensated by the (almost) perfect representation of the phase-space limits already at a crude level and by the speed of the code.

  With our assumptions, the integral $\mathcal{F}$ can be easily calculated:

$$\mathcal{F}(s_{max}) = \int_{\Omega^C_{CR}} f(s_1, s_2) ds_1 ds_2 = \int_0^{s_{max}} ds_1 f_1(s_1) \int_0^{s_1} ds_2 f_1(s_2) \tag{33}$$

$$= \frac{1}{2} \left[F_1(s_{max}) - F_1(0) \right]^2, \tag{34}$$

$$F_1(x) = \int^x f_1(s) ds \tag{35}$$

  and the distribution can be normalized, assuming one knows the F_1 function.

- The "fork".
  Also in this configuration we assume factorization of the function $f(s_1, s_2)$ into the one-dimensional functions $f_1(s_1)$ and $f_2(s_2)$:

$$f(s_1, s_2) = \frac{1}{\mathcal{F}(s_{max})} f_1(s_1) f_2(s_2). \tag{36}$$

  The phase-space limits ($\Theta(\Omega^F)$ of eq. (8)) are the following:

$$\sqrt{s_1} + \sqrt{s_2} \leq \sqrt{s_{max}}, \;\; \sqrt{s_1} \geq m_1 + m_2, \;\; \sqrt{s_2} \geq m_3 + m_4. \tag{37}$$

  This area can be extended to the polygonal Ω^F_{CR} domain, or explicitly to the surface defined by the difference between the two squares, as shown in Fig. 2b. The integration is easy:

$$\begin{aligned}
\mathcal{F}(s_{max}) &= \int_{\Omega^F_{CR}} f_1(s_1) f_2(s_2) ds_1 ds_2 && (38)\\
&= \left[\int_0^{s_{max}} - \int_{s_{max}/4}^{s_{max}} \right] ds_1 ds_2 f_1(s_1) f_2(s_2) && (39)\\
&= \left[F_1(s_{max}) - F_1(0) \right] \left[F_2(s_{max}) - F_2(0) \right] && (40)\\
&\quad - \left[F_1(s_{max}) - F_1(\frac{s_{max}}{4}) \right] \left[F_2(s_{max}) - F_2(\frac{s_{max}}{4}) \right], && (41)
\end{aligned}$$

$$F_j(x) = \int^x f_j(s) ds, \;\; j = 1, 2. \tag{42}$$

Finally we make a choice of a suitable form of f_j functions. We use a multi-branch form of it, each branch α being associated with a basic mass singularity type:

$$f_j(s) = \sum_\alpha^{N_{MAS}} a_{j\alpha} \frac{f_{j\alpha}(s)}{\int_0^{s_{max}} f_{j\alpha}(x) dx}, \;\; \sum_\alpha^{N_{MAS}} a_{j\alpha} = 1, \tag{43}$$

$$f_{j\alpha}(s) = \begin{cases} 1, \\ (s + m_R^2)^{-n}, \\ (s + m_R^2)^{-1} \log^n \left((s_{max} + m_R^2)/(s + m_R^2) \right), \\ \left((s - M_k^2)^2 + M_k^2 \Gamma_k^2 \right)^{-1}, \\ (s_{max} + m_R^2 - s)^{-1}. \end{cases} \tag{44}$$

The N_{MAS} denotes the total number of "mass branches". The parameter m_R^2 takes a role of a regulator, preventing distributions from blowing up to infinity at the phase-space limit. It should be stressed that m_R^2 does not play the role of a cut-off, and that the complete phase space is covered by the generation. The M_k and Γ_k are parameters of the k-th resonance. All the functions presented above are easily integrable into the elementary functions.

3.2 Angular Distributions

In the next step we generate the angular variables: $\cos\theta_j$ and ϕ_j (the index $j = 1, 2, 3$ denotes the number of angle within the same branch).

The role of the angular variables is to describe the class of t- and u-channel-type singularities of the Feynman graphs. There is a number of different transfers that can be built, based on two beams and four external final-state four-vectors. In principle each of them can develop a singularity in some of the Feynman graphs for some configurations of final-state fermions.

The general angular function $g(\cos\underline{\theta}, \underline{\phi}, \underline{s})$ of eq. (28) is further factorized with respect to the angles of each step of the generation. Unlike the mass distributions case, the factorization is not complete – each subsequent angle uses previously generated angles and masses (in the form of four-momenta). We write it symbolically as:

$$g(\cos\underline{\theta}, \underline{\phi}, \underline{s}) = g_1(\cos\theta_1, \phi_1; \underline{s}) g_2(\cos\theta_2, \phi_2; \theta_1, \phi_1, \underline{s}) g_3(\cos\theta_3, \phi_3; \theta_1, \phi_1, \theta_2, \phi_2, \underline{s}). \quad (45)$$

Much as with the mass distributions, we use branching structure to build individual g_j functions (we always take flat distributions in ϕ_j variables and thus their generation decouples):

$$g_j(\cos\theta_j, \phi_j; ..., \underline{s}) = \sum_{\alpha}^{N_{ANG}} b_{j\alpha} \frac{g_{j\alpha}(\cos\theta_j, \underline{s})}{\int_{-1}^{1} d\cos\theta \int_0^{2\pi} d\phi g_{j\alpha}(\cos\theta, \underline{s})}, \quad \sum_{\alpha}^{N_{ANG}} b_{j\alpha} = 1, \quad (46)$$

with N_{ANG} denoting the number of "angular branches". The basic angular distribution functions $g_{j\alpha}$ are:

$$g_{j\alpha}(\cos\theta_j, \underline{s}) = \begin{cases} 1, \\ (-t_j + m_A^2)^{-n}, \\ (-t_j + m_A^2)^{-1} \log^n \left((-t_{j,max} + m_A^2)/(-t_j + m_A^2)\right), \\ (-u_j + m_A^2)^{-n}, \\ (-u_j + m_A^2)^{-1} \log^n \left((-u_{j,max} + m_A^2)/(-u_j + m_A^2)\right), \end{cases} \quad (47)$$

where t_j and u_j are functions of $\cos\theta_j$ and $\underline{s}$-variables. The ϕ_j variables are independent and generated with a flat distribution. The m_A^2 is, as before, a small mass introduced to regularize the distributions at singularity points.

3.3 Event Construction

Now, we can continue step by step with the generation and construction of the explicit form of the event with the help of the variables generated as explained in the previous sections.

Let us first define the relation between angles and t_j and u_j transfers. For any two-to-two sub-process of our generation (see Fig. 1), the transfers t_j and u_j are defined as

$$t_j = (P_1 - q_{1_j})^2, \quad u_j = (P_1 - q_{2_j})^2, \tag{48}$$

where P_1, P_2 denote four-momena of the first and second beams, and q_{1_j}, q_{2_j} denote the four-momenta of the first and second outgoing objects. These outgoing objects can either be the final-state particles or groups of particles ("intermediate states"). Later on, we will use the masses of the objects to label them. Thus $[m_1^2]$, $[s_1]$ will denote respectively the first final-state fermion and the system consisting of second, third and fourth particles (case of fig. 1a) or the system consisting of first and second particles (case of fig. 1b). In every case, the invariant mass of the object is known, either from the input parameters or from the mass generation in the previous step of the algorithm. In this way one has an access to all possible transfers t_j and u_j.

Angles corresponding to the invariants are easiest to generate in the local CMS frames of outgoing four-vectors (q_{1_j}, q_{2_j}). In such a frame the $t_j(\cos\theta_j, \underline{s})$ and $u_j(\cos\theta_j, \underline{s})$ can be expressed in a standard way (we drop the index j here):

$$t = (P_1 - q_1)^2 = q_1^2 + P_1^2 - 2q_1^0 P_1^0 + 2|\vec{q}_1||\vec{P}_1|\cos\theta, \tag{49}$$

$$u = (P_1 - q_2)^2 = q_1^2 + P_2^2 - 2q_1^0 P_2^0 - 2|\vec{q}_1||\vec{P}_1|\cos\theta, \tag{50}$$

with

$$q_1^0 = \frac{1}{2\sqrt{s_{12}}}\left(s_{12} + q_1^2 - q_2^2\right), \tag{51}$$

$$P_1^0 = \frac{1}{2\sqrt{s_{12}}}\left(s_{12} + P_1^2 - P_2^2\right), \tag{52}$$

$$|\vec{q}_1|^2 = \frac{1}{4}s_{12}\lambda\left(1, q_1^2/s_{12}, q_2^2/s_{12}\right), \tag{53}$$

$$|\vec{P}_1|^2 = \frac{1}{4}s_{12}\lambda\left(1, P_1^2/s_{12}, P_2^2/s_{12}\right), \tag{54}$$

$$s_{12} = (P_1 + P_2)^2. \tag{55}$$

Finally we need to define a series of frames (i.e. four-momenta of group of particles) used to construct subsequent angles. This can be, as a matter of fact, decoded from figs. 1a and 1b.

For the configuration of fig. 1a we start by generating the angle $\angle([m_1^2], P_1)$ in the rest frame of the two beams P_1 and P_2. Next, we subtract the generated final four-momentum $[m_1^2]$ from the second beam P_2 and repeat the previous step for the angle $\angle([m_2^2], P_1)$ in the rest frame of the pair $([m_2^2], [s_2])$. Finally, in the third step, after subtracting again $[m_2^2]$ from $P_2 - [m_1^2]$, we build the angle $\angle([m_3^2], P_1)$ in the rest frame of the pair $([m_3^2], [m_4^2])$.

In the case of fig. 1b, the first and second angles, $\angle([s_1], P_1)$ in the rest frame of the pair $([s_1], [s_2])$ and $\angle([m_1^2], P_1)$ in the rest frame of the pair $([m_1^2], [m_2^2])$, are generated

as before. In the definition of the third angle, $\angle([m_3^2], P_2)$ in the rest frame of the pair $\angle([m_3^2], [m_4^2])$, the direction of the second beam P_2 is chosen as the z-axis instead of the first one.

3.4 Normalization

The last step is to check out the normalization $\mathcal{N}^i_{CR}$ of a given ψ^i_{CR} used in the definition of the branch i of eq. (10). We need to calculate the integral $\int d\Phi^i \psi^i_{CR}$:

$$\begin{aligned}\mathcal{N}^i_{CR} &= \int\limits_{\Omega^i_{CR}} d\Phi^i(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s})\psi^i_{CR}(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s}) \\ &= \int ds_1 ds_2 f(s_1, s_2) \int \prod_{j=1}^{3} d\cos\theta_j d\phi_j g_j(\cos\theta_1, \phi_1; \ldots, \theta_j, \phi_j, \underline{s}) = 1.\end{aligned} \tag{56}$$

The normalization to unity follows from the specific form of the f and g functions of our choice. The property that normalization factors $\mathcal{N}^i_{CR} = 1$, although not necessary for the correctness of the algorithm, simplifies the practical use. Let us elaborate more on this point. Thanks to the normalization, and if in formulae (44) and (47) only single branches are present, the product of functions f and g forms the Jacobian of the transformation from the Ω^i_{CR} domain to the eight-dimensional unit cube. The inverse transformation induces the measure on Ω^i_{CR}, which is the (crude) probability distribution. In the multi-channel case, when coefficients a_{j_α} and b_{j_α} can all be non-zero, the measure (probability distribution) on Ω^i_{CR} is defined as the weighted[3] sum of measures defined by individual transformations.

3.5 Optimization

Finally, we have to discuss the choice of the coefficients p_i, $a_{j\alpha}$ and $b_{j\alpha}$. It is these parameters that assure flexibility of the algorithm. The coefficients p_i, $a_{j\alpha}$ and $b_{j\alpha}$ are free parameters of the generator. Their choice depends on the final state chosen as well as on the external cuts used in the calculation of cross-section. An optimal choice of these parameters is a delicate and important part of the algorithm. We did it "manually", by looking at the physical structures they approximate. Our strategy was to eliminate the most overweighted events by iterative adjustments of these coefficients and, if needed, adding more branches to the master sum of eq. (4). One can easily imagine other, more sophisticated, optimization procedures, for example by assuming functional dependence of these coefficients on the type of the final fermions or by allowing for functional dependence on external cut-offs. This latter optimization may be especially interesting. In the case of strongly peaked cross-sections, strong cut-offs can downgrade the efficiency of the algorithms by causing excessive rejection, unless compensated by retuning of these

[3]More precisely, weighted with appropriate products of coefficients/probabilities a_{j_α} and b_{j_α}.

internal parameters. Yet another option would be a numerical optimization of some sort. One can mention in this context the approach of Ref. [25], for example.

To summarize, such a large number of free parameters makes it more complicated to find the optimal configuration of the coefficients, but on the other hand it gives a direct access to virtually *every* singular configuration and gives us a possibility for its direct modelling.

We also have to remember that p_i, $a_{j\alpha}$ and $b_{j\alpha}$ are dummy parameters. It means that, by varying them, one gets a strong tool for testing the correctness of the integration.

4 The Algorithm

At this point we are ready to put together all the pieces and write down the complete formula for the overall weight and normalization for our algorithm. To be able to calculate σ of eq. (1), the following steps have to be completed:

1. Generate the branch number i according to probabilities p_k.
2. Generate point $(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s})$ according to the ψ^i_{CR} distribution of the chosen i-th branch.
3. Construct the weight $w^a(i)$.
4. If $w^a(i) \neq 0$ construct the phase-space point $(\underline{q})$ out of $(\cos\underline{\theta}^i, \underline{\phi}^i, \underline{s})$.
5. Calculate the total weight $w(i, P, \underline{q})$ of eq. (21) (or (26)) in terms of four-vectors.
6. Calculate the cross-section as an average of the weight $w(i, P, \underline{q})$

$$\sigma = \left\langle w(i, P, \underline{q}) \right\rangle . \tag{57}$$

A few comments are in order here.

While constructing a set of all branches, one has to include all the necessary permutations of external momenta of beams and outgoing fermions. The generic two branches — "chain" and "fork", need to be applied to all configurations of external momenta to describe all the singularities properly. All together, this leads to over fifty branches in the case of four-fermion final states[4].

The algorithm as described in this note is the one with variable weights. By means of the standard rejection technique it can provide unweighted events as well.

[4]If one wanted to be even more precise, and counted separately each "basic" branch of eqs. (44) and (47), the total number of branches would easily exceed 10^6!

5 Summary

In this paper we gave a complete description of the Monte Carlo algorithm for the generation of the four-fermion phase space, as needed for LEP2 applications. It is based on the "multichannel" approach, as explained elsewhere e.g. in [24]. Strong emphasis is given to the construction of separate branches with the help of rather simple modular building blocks. This approach allowed for transparent writing of the code and testing, as well as for easy extensions and modifications in the future.

The major disadvantage of the present version, or rather the important place for future improvements, is optimization of internal coefficients. In particular, in the case of big changes of external cut-offs or centre-of-mass energy, the coefficients may need retuning. At present this must be done by hand.

The algorithm has been successfully implemented in the KORALW [2] Monte Carlo program. Generation can cover the full phase space, or any sub-region defined by cut-offs[5]. This is the case of all four-fermion final states at LEP2 centre-of-mass energies. Numerically stable results, with a statistical precision of few per mille, can be obtained from the runs easily attainable in a few hours of CPU time of any modern PC.

Acknowledgements

Authors are indebted to S. Jadach, W. Płaczek, A. Vallasi, M. Witek for stimulating discussions and cooperation at different steps of the algorithm development. One of the authors (MS) acknowledges support of the CERN ALEPH group.

References

[1] M. Skrzypek, S. Jadach, W. Płaczek and Z. Wąs, Comput. Phys. Commun. **94**, 216 (1996).

[2] S. Jadach et al., CERN preprint CERN-TH/98-242, to appear in Comput. Phys. Commun.

[3] *Physics at LEP2*, edited by G. Altarelli, T. Sjöstrand and F. Zwirner (CERN 96-01, Geneva, 1996), 2 vols.

[4] F. Caravaglios and M. Moretti, Z. Phys. **C74**, 291 (1997).

[5] E. Accomando and A. Ballestrero, Comput. Phys. Commun. **99**, 270 (1997).

[6] G. van Oldenborgh, P. Franzini and A. Borrelli, Comput. Phys. Commun. **83**, 14 (1994).

[5] In KORALW some of the cut-offs can be optionally implemented before lengthy calculation of the matrix element. This is essential, e.g. in the generation of $e^+e^- f\bar{f}$ final states with tagged electrons when the rejection rate is very high.

[7] D. Charlton, G. Montagna, O. Nicrosini and F. Piccinini, Comput. Phys. Commun. **99**, 355 (1997).

[8] J. Fujimoto et al., Comput. Phys. Commun. **100**, 128 (1997).

[9] E. E. Boos et al., Korean Physical Society Meeting, October 21, 1994, e-Print Archive: hep-ph/9503280 (unpublished).

[10] R. Miquel and M. Schmitt, Z. Phys. **C71**, 251 (1996).

[11] H. Anlauf, P. Manakos, T. Ohl and H. Dahmen, IKDA-96-15, e-Print Archive: hep-ph/9605457 (unpublished).

[12] F. A. Berends, R. Pittau and R. Kleiss, Nucl. Phys. **B424**, 308 (1994).

[13] F. A. Berends, R. Pittau and R. Kleiss, Comput. Phys. Commun. **85**, 437 (1995).

[14] C. Papadopoulos, Comput. Phys. Commun. **101**, 183 (1997).

[15] S. Jadach, J. H. Kühn and Z. Wąs, Comput. Phys. Commun. **64**, 275 (1990).

[16] M. Jeżabek, Z. Wąs, S. Jadach and J. H. Kühn, Comput. Phys. Commun. **70**, 69 (1992).

[17] R. Decker, S. Jadach, J. H. Kühn and Z. Wąs, Comput. Phys. Commun. **76**, 361 (1993).

[18] J. Hilgart, R. Kleiss and F. Le Diberder, Comput. Phys. Commun. **75**, 191 (1993).

[19] F. James, CERN Program Library W505, CERN, Geneva, 1972 (unpublished).

[20] F. Berends, R. Kleiss and S. Jadach, Comput. Phys. Commun. **29**, 185 (1983).

[21] S. Jadach, Acta Phys. Pol. **B16**, 1007 (1985).

[22] E. Richter-Wąs, Z. Phys. **C61**, 323 (1994).

[23] E. Richter-Wąs, Z. Phys. **C64**, 227 (1994).

[24] S. Jadach, Practical Guide to Monte Carlo, `http://home.cern.ch/~jadach`, CERN 1998 (unpublished).

[25] R. Kleiss and R. Pittau, Comput. Phys. Commun. **83**, 141 (1994).