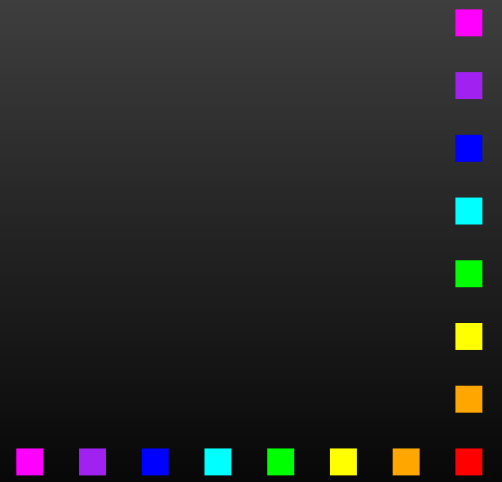
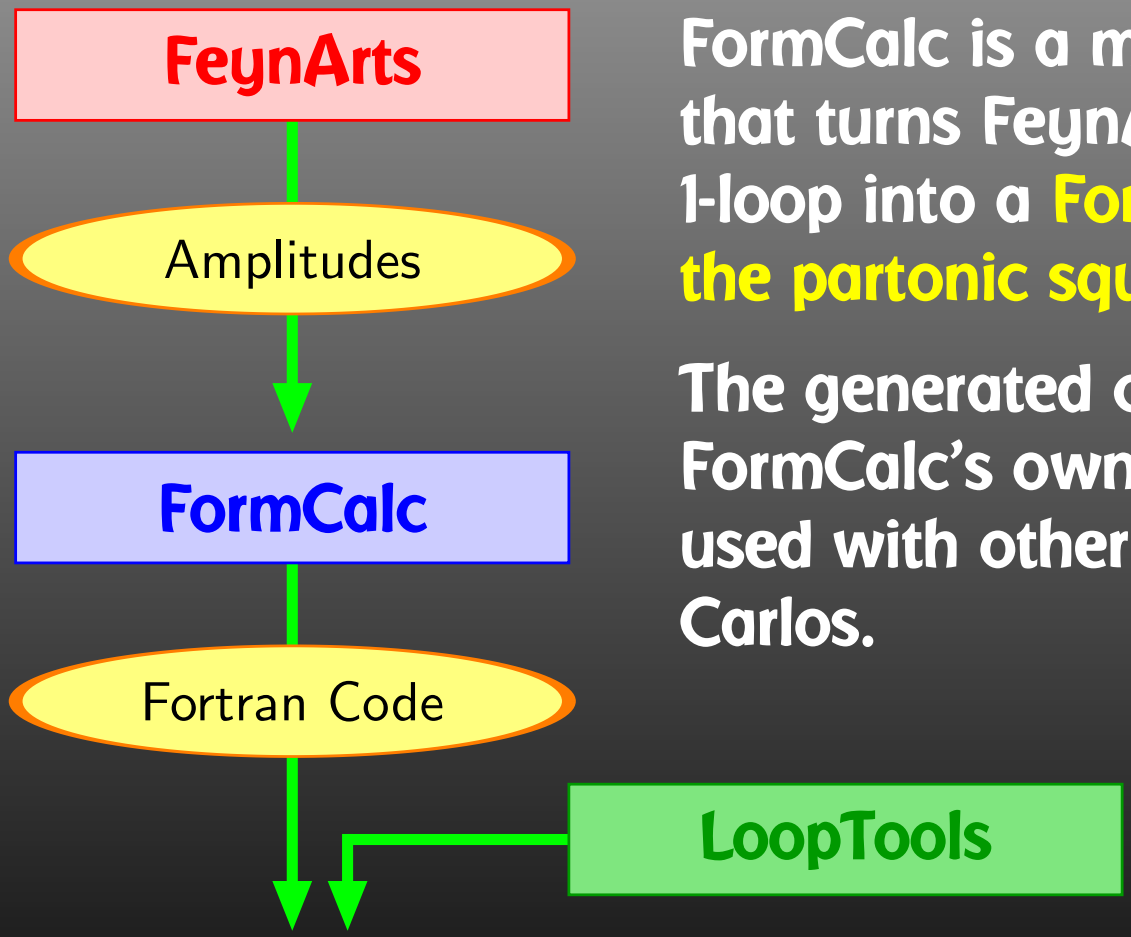
New Developments in FormCalc

Thomas Hahn

Max-Planck-Institut für Physik
München



What is FormCalc?

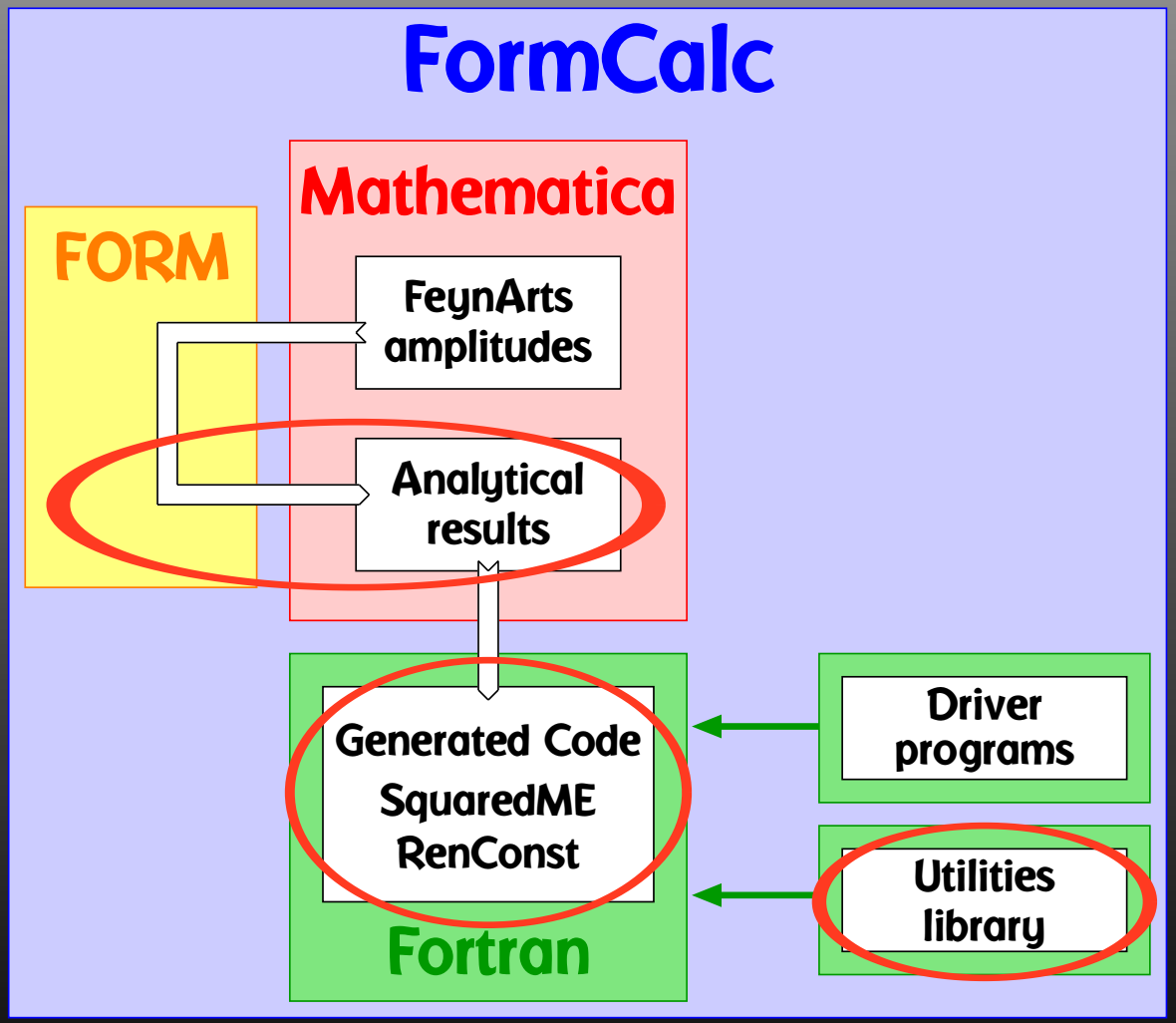


FormCalc is a matrix-element generator that turns FeynArts amplitudes up to 1-loop into a Fortran code for computing the partonic squared matrix element.

The generated code can be run with FormCalc's own driver programs, or used with other 'Frontends', e.g. Monte Carlos.

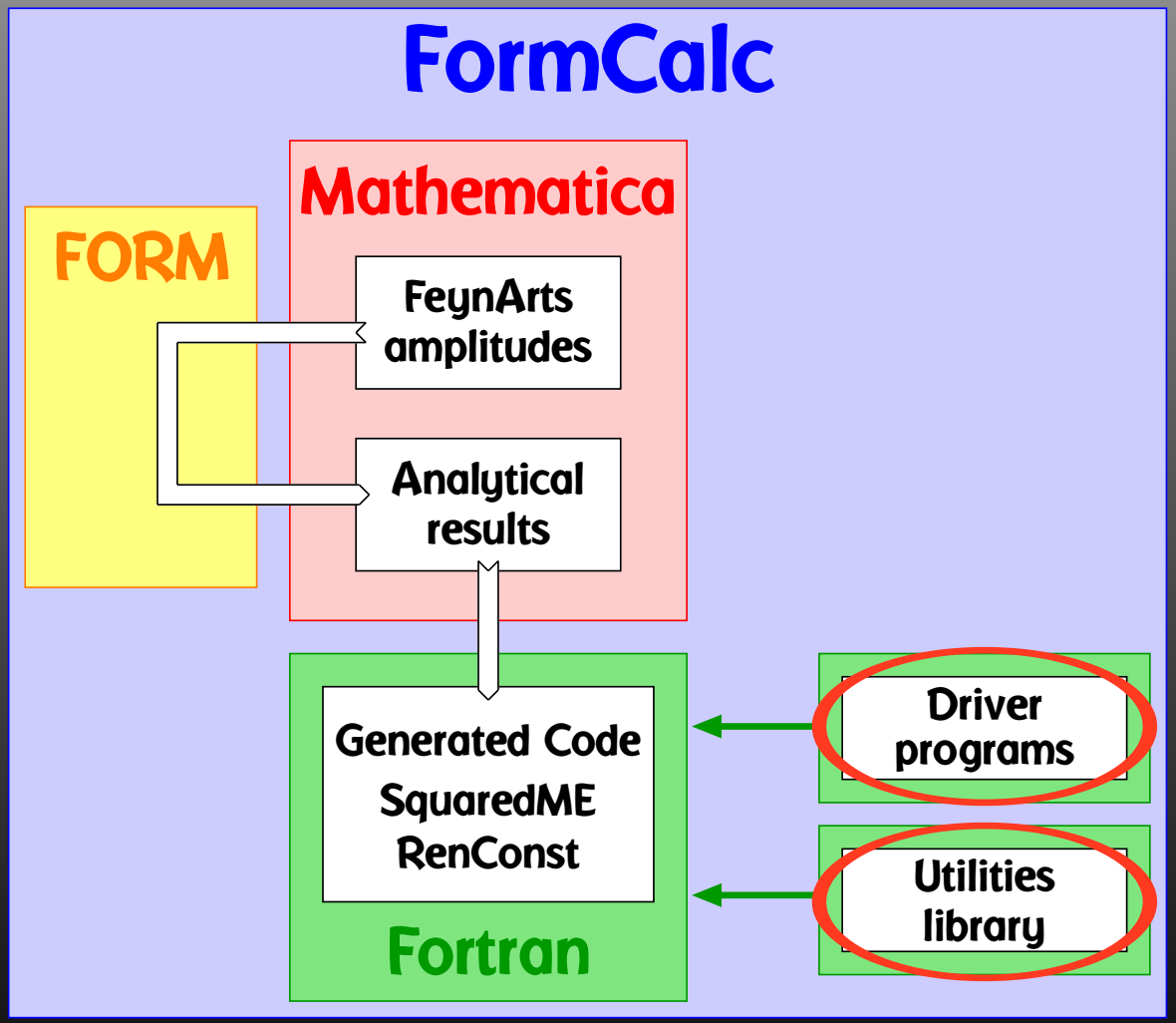
$|\mathcal{M}|^2 \longrightarrow$ Cross-sections, Decay rates, ...

New Stuff



Weyl-van der Waerden spinor formalism for external fermions

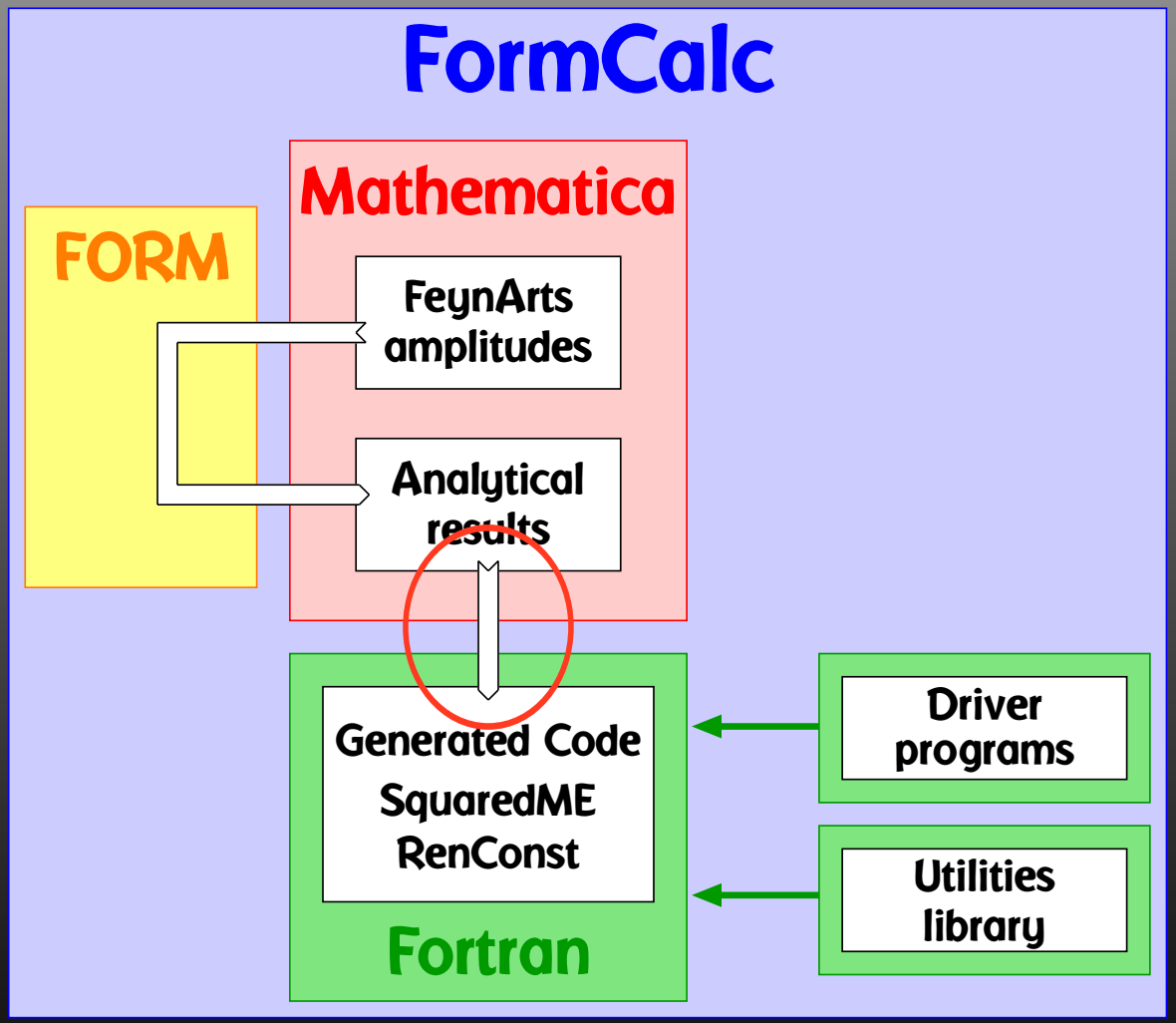
New Stuff



Weyl-van der Waerden spinor formalism for external fermions

Multidimensional integration with CUBA

New Stuff

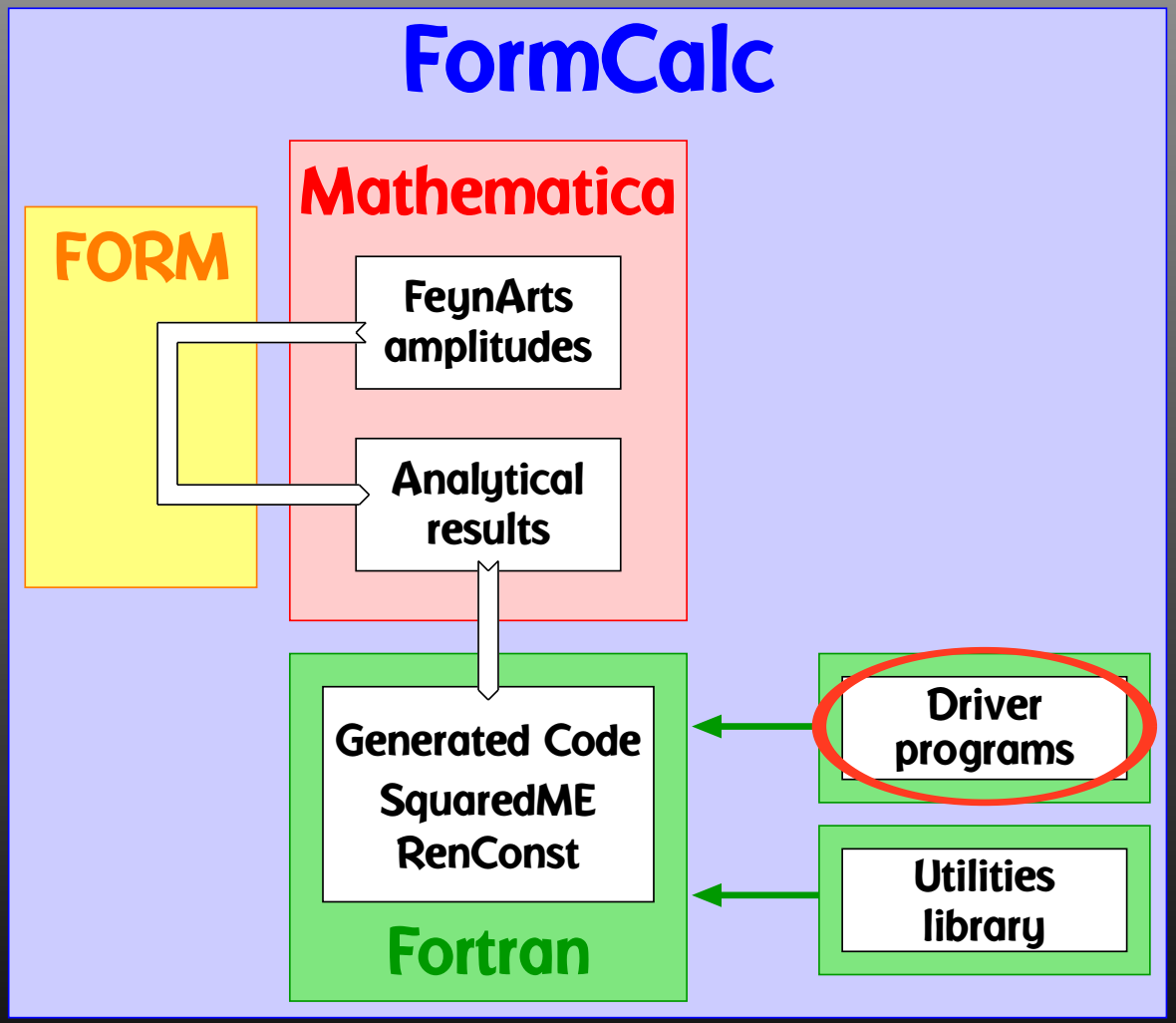


Weyl-van der Waerden spinor formalism for external fermions

Multidimensional integration with CUBA

Optimized public code-generation funcs

New Stuff



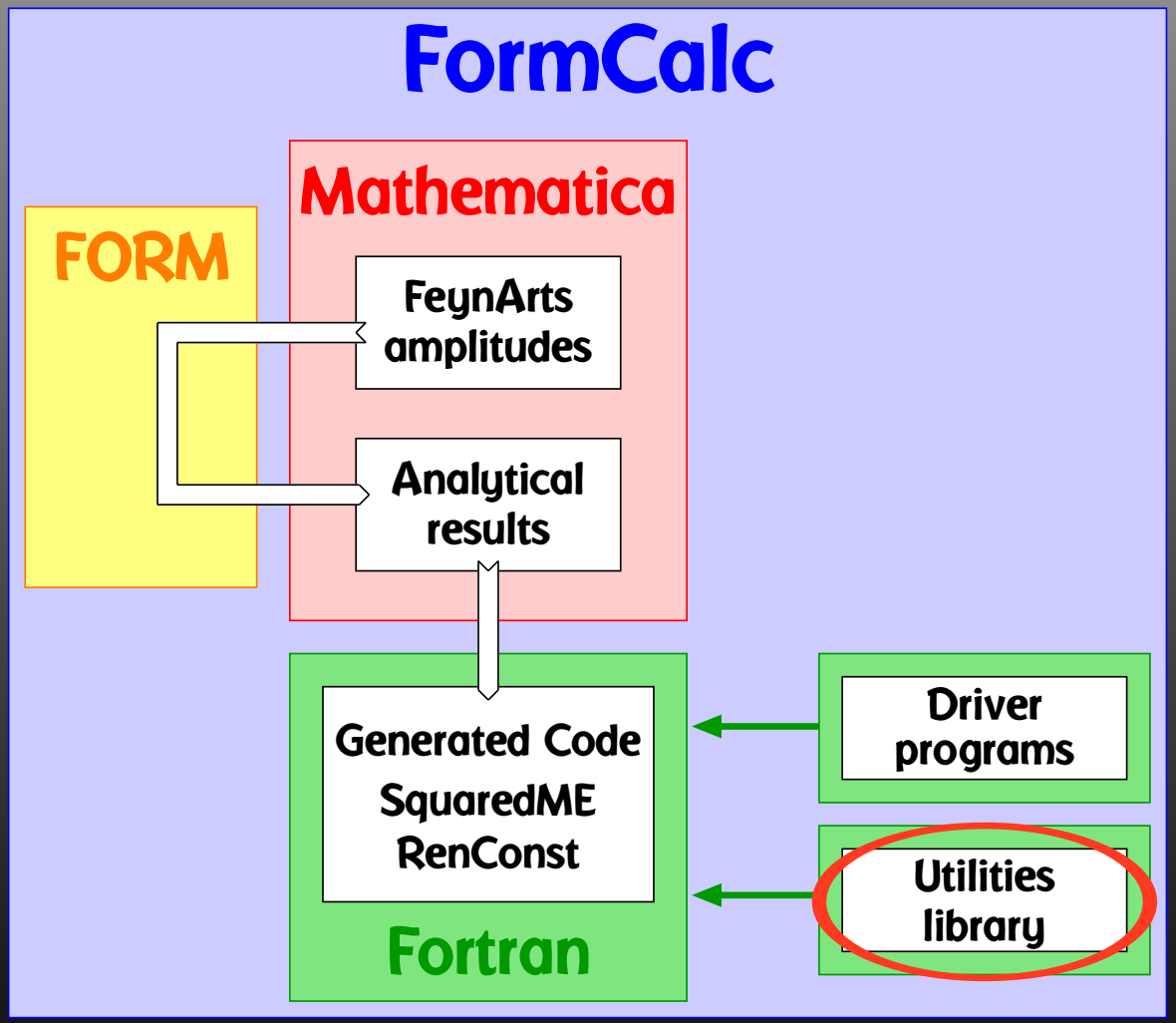
Weyl-van der Waerden spinor formalism for external fermions

Multidimensional integration with CUBA

Optimized public code-generation funcs

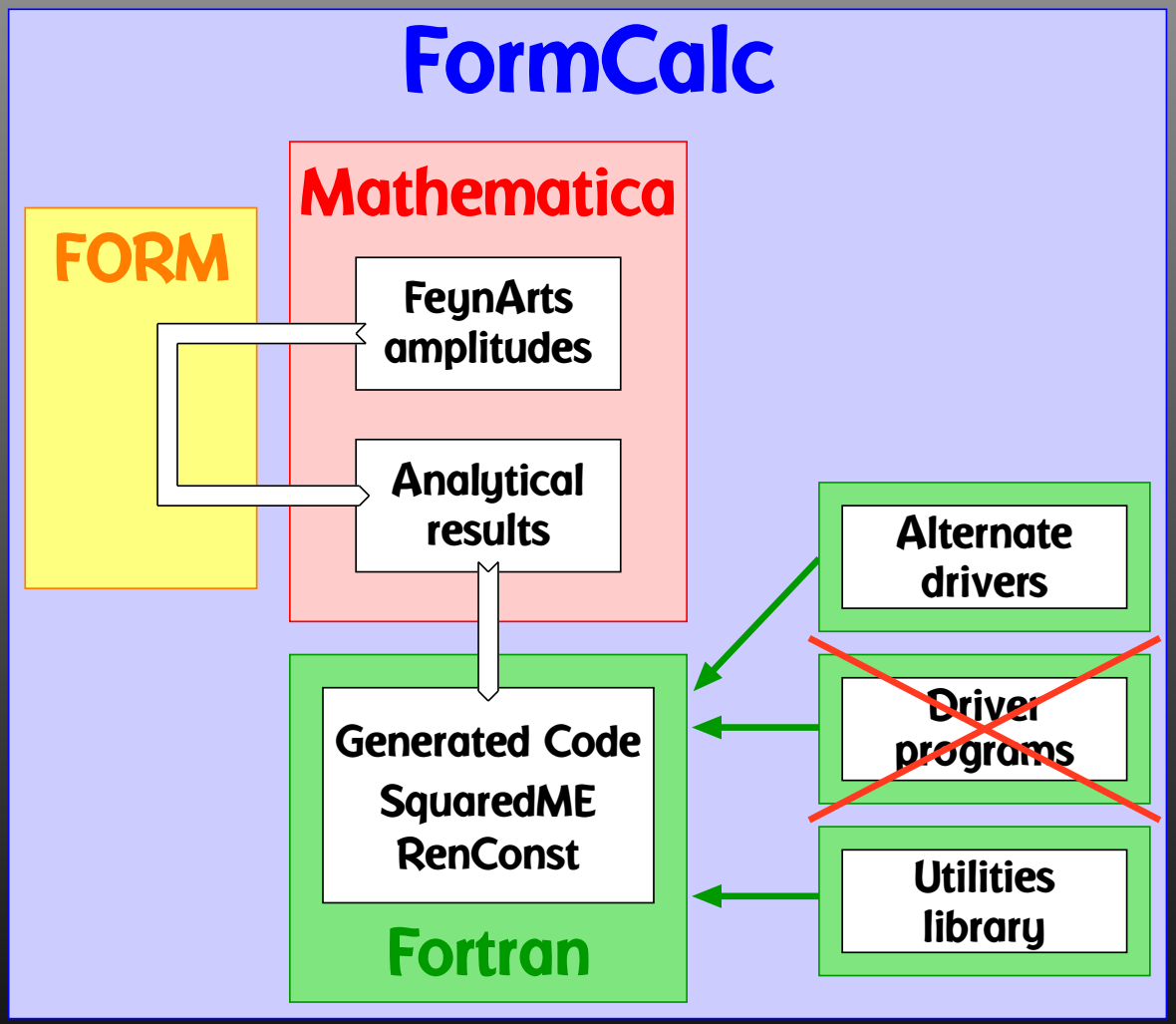
FeynHiggs interface and NMFV

New Stuff



- Weyl-van der Waerden spinor formalism for external fermions
- Multidimensional integration with CUBA
- Optimized public code-generation funcs
- FeynHiggs interface and NMFV
- Useful scripts

New Stuff



- Weyl-van der Waerden spinor formalism for external fermions
- Multidimensional integration with CUBA
- Optimized public code-generation funcs
- FeynHiggs interface and NMFV
- Useful scripts
- HadCalc by M. Rauch

External Fermion Lines

An amplitude containing external fermions has the form

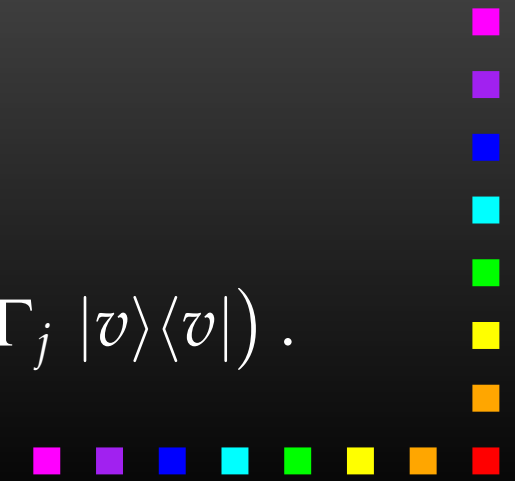
$$\mathcal{M} = \sum_{i=1}^{n_F} c_i F_i \quad \text{where} \quad F_i = \text{(Product of)} \langle u | \Gamma_i | v \rangle .$$

$n_F =$ number of fermionic structures.

Textbook procedure: Trace Technique

$$|\mathcal{M}|^2 = \sum_{i,j=1}^{n_F} c_i^* c_j F_i^* F_j$$

where $F_i^* F_j = \langle v | \bar{\Gamma}_i | u \rangle \langle u | \Gamma_j | v \rangle = \mathrm{Tr}\left(\bar{\Gamma}_i \, |u\rangle\langle u| \, \Gamma_j \, |v\rangle\langle v|\right) .$



Problems with the Trace Technique

PRO: Trace technique is independent of any representation.

CON: For n_F F_i's there are n_F^2 $F_i^* F_j$'s.

Things get worse the more vectors are in the game: multi-particle final states, polarization effects . . .

Essentially $n_F \sim$ (# of vectors)! because all combinations of vectors can appear in the Γ_i.

Solution: Use Weyl-van der Waerden spinor formalism to compute the F_i's directly.

Sigma Chains

Define **Sigma matrices** and **2-dim. Spinors** as

$$\sigma_\mu = (\mathbb{1}, -\vec{\sigma})\,, \qquad \overline{\sigma}_\mu = (\mathbb{1}, +\vec{\sigma})\,,$$

$$\langle u|_{4\mathrm{d}} \equiv \left(\langle u_+|_{2\mathrm{d}}\,, \langle u_-|_{2\mathrm{d}}\right), \qquad |v\rangle_{4\mathrm{d}} \equiv \begin{pmatrix} |v_-\rangle_{2\mathrm{d}} \\ |v_+\rangle_{2\mathrm{d}} \end{pmatrix}.$$

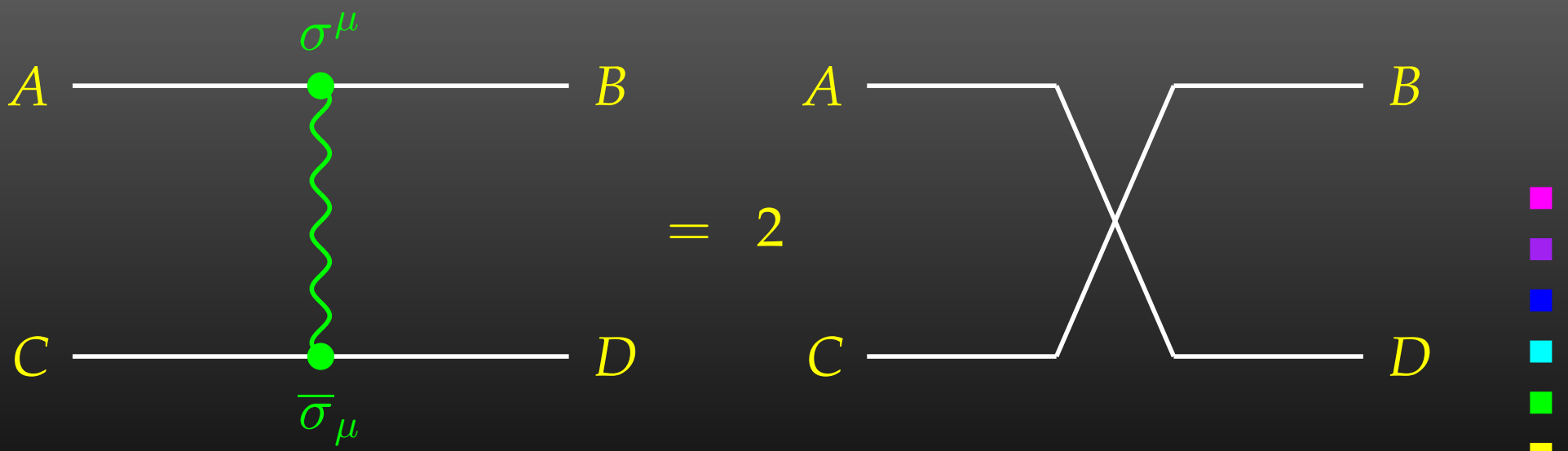
Using the chiral representation it is easy to show that **every chiral 4-dim. Dirac chain** can be converted to a ***single*** **2-dim. sigma chain:**

$$\langle u|\,\omega_-\gamma_\mu\gamma_\nu\cdots|v\rangle = \langle u_-|\,\overline{\sigma}_\mu\sigma_\nu\cdots|v_\pm\rangle\,,$$
$$\langle u|\,\omega_+\gamma_\mu\gamma_\nu\cdots|v\rangle = \langle u_+|\,\sigma_\mu\overline{\sigma}_\nu\cdots|v_\mp\rangle\,.$$

Fierz Identities

With the Fierz identities for sigma matrices it is possible to remove all Lorentz contractions between sigma chains, e.g.

$$\langle A|\,\sigma_\mu\,|B\rangle\,\langle C|\,\overline{\sigma}^\mu\,|D\rangle \;=\; 2\,\langle A|D\rangle\,\langle C|B\rangle$$



Implementation

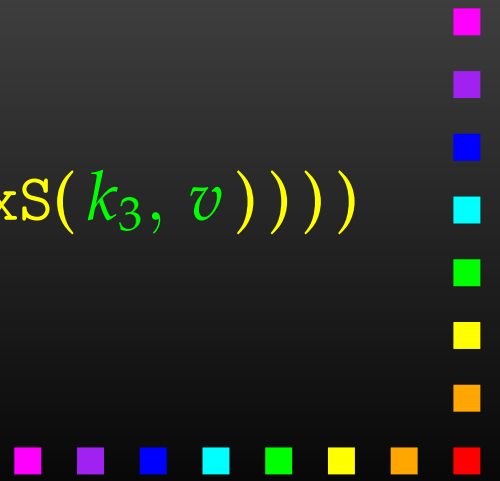
- **Objects (arrays):** $|u_\pm\rangle \sim \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \quad (\sigma \cdot k) \sim \begin{pmatrix} a & b \\ c & d \end{pmatrix}$
- **Operations (functions):**

$$\langle u|v\rangle \sim (u_1 \;\; u_2) \cdot \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \qquad \texttt{SxS}$$

$$(\overline{\sigma} \cdot k)\,|v\rangle \sim \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \qquad \texttt{VxS, BxS}$$

Sufficient to compute any sigma chain:

$$\langle u|\,\sigma_\mu \overline{\sigma}_\nu \sigma_\rho\,|v\rangle\, k_1^\mu k_2^\nu k_3^\rho = \texttt{SxS}(u, \texttt{VxS}(k_1, \texttt{BxS}(k_2, \texttt{VxS}(k_3, v))))$$



More Freebies

- Polarization does not 'cost' extra:
  = Get spin physics for free.
- Better numerical stability because components of k^μ are arranged as 'small' and 'large' matrix entries, viz.

$$\sigma_\mu k^\mu = \begin{pmatrix} k_0 + k_3 & k_1 - \mathrm{i}k_2 \\ k_1 + \mathrm{i}k_2 & k_0 - k_3 \end{pmatrix}$$

$$\downarrow$$

Large cancellations of the form $\sqrt{k^2 + m^2} - \sqrt{k^2}$ when $m \ll k$ are avoided: better precision for mass effects.

Routines in the CUBA Library

Routine	Basic method	Type	Variance reduction
Vegas	Sobol sample *or* MT sample	Monte Carlo Monte Carlo	importance sampling
Suave	Sobol sample *or* MT sample	Monte Carlo Monte Carlo	globally adaptive subdivision
Divonne	Korobov sample *or* Sobol sample *or* MT sample *or* cubature rules	Monte Carlo Monte Carlo Monte Carlo deterministic	stratified sampling, aided by methods from numerical optimization
Cuhre	cubature rules	deterministic	globally adaptive subdivision

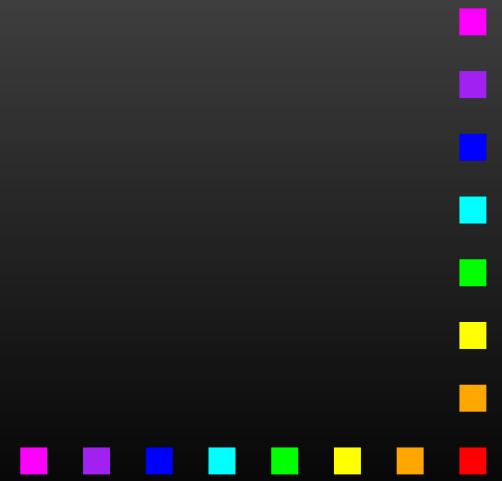
- Very similar invocation (easily interchangeable).
- Fortran, C/C++, Mathematica interface provided.
- Can integrate vector integrands.

CUBA 1.2

Many improvements in CUBA 1.2, e.g.

- Vegas can memorize its grid for subsequent invocations,
- Vegas can save its internal state such that the calculation can be resumed e.g. after a crash,
- One-stop invocation for CUBA routines, i.e. the user calls the single subroutine

```
subroutine Cuba(ndim, integrand, result, error)
```

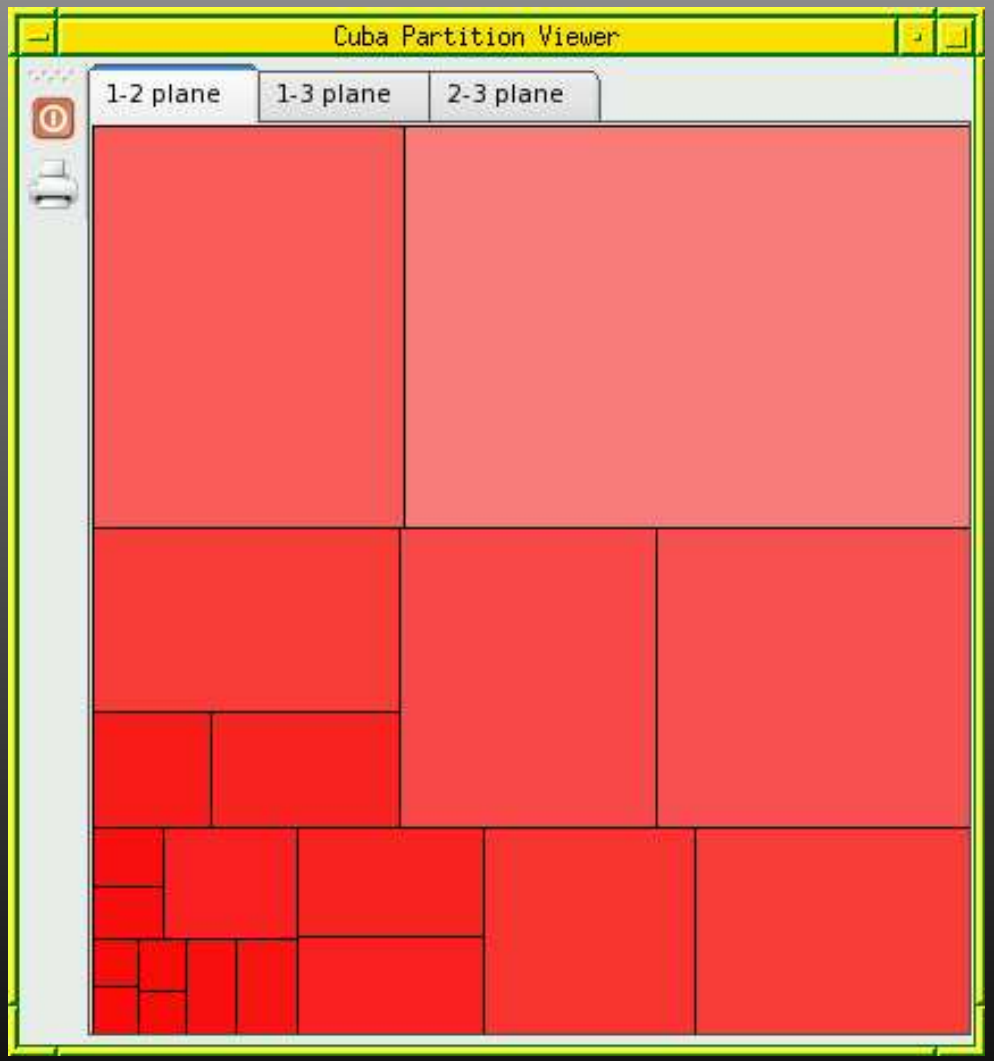


Partition Viewer

CUBA's Partition Viewer visualizes the partition taken by the integration algorithm.

Verbosity level 3 must be chosen and the output piped through `partview`:

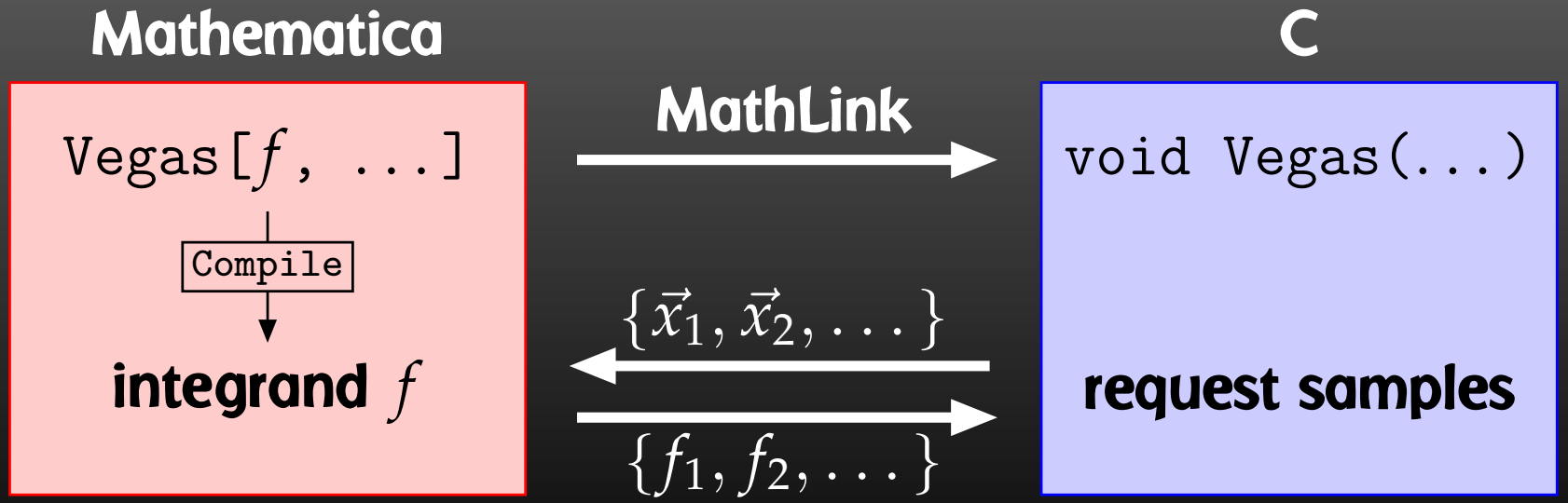
```
myprog | partview 1 2
```



Mathematica interface

- **Used almost like** `NIntegrate`.
- **The integrand is evaluated completely in Mathematica. Can do things like**

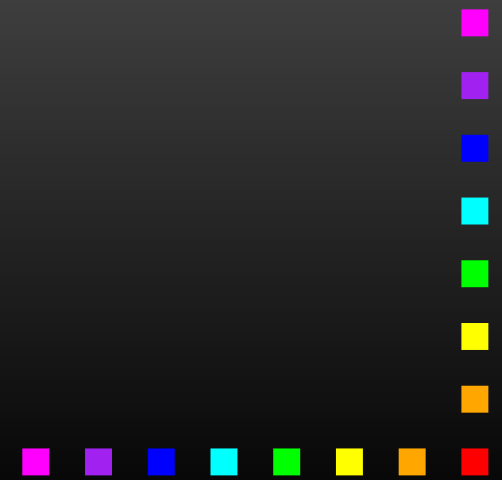
```
Cuhre[PolyLog[2, x y], {x,.2,.3}, {y,.4,.5}]
```



Code-generation Functions

FormCalc's code-generation functions are now public and disentangled from the rest of the code. They can be used to write out an arbitrary Mathematica expression as optimized Fortran code:

- *handle* `= OpenFortran["`*file.F*`"]`
  opens *file.F* as a Fortran file for writing,
- `WriteExpr[`*handle*`, {`*var* `->` *expr*`,...}]`
  writes out Fortran code which calculates *expr* and stores the result in *var*,
- `Close[`*handle*`]`
  closes the file again.



Code generation

- **Expressions too large** for Fortran are split into parts, as in

```
var = part1
var = var + part2
...
```

- **High level of optimization,** e.g. common subexpressions are pulled out and computed in temporary variables.
- **Many ancillary functions,** e.g.

  `PrepareExpr`, `OnePassOrder`, `SplitSums`, `$Prefix`, `CommonDecl`, `SubroutineDecl`, **etc.**

  make code generation versatile and highly automatable, such that the resulting code needs few or no changes by hand.

FeynHiggs Interface and NMFV

Various methods can be chosen to calculate the Higgs masses:

`#define HIGGS_MASSES TREE` - use tree-level Higgs masses,

`#define HIGGS_MASSES SIMPLE` - use simple 1-loop formula,

`#define HIGGS_MASSES FEYNHIGGS` - call FeynHiggs
(this is the most precise determination),

`HIGGS_MASSES` **undefined** - use FeynHiggsFast approximation
(quite precise, but valid only for real parameters).

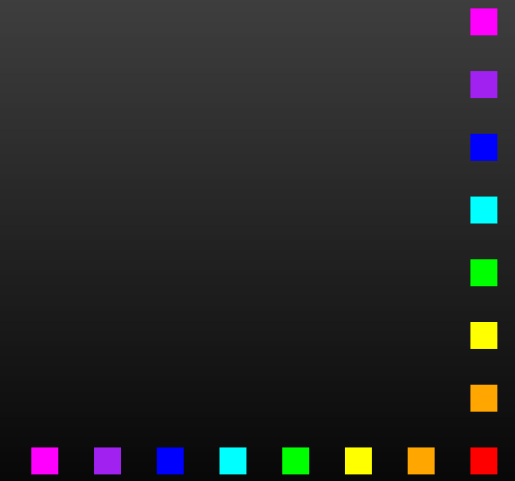
Non-minimal flavour violation, i.e. full 6×6 mixing among sfermions, now available both in FeynArts and FormCalc.

Useful Shell Scripts

- `sfx` packs all files into a mail-safe self-extracting archive.
- `turnoff` switches off/on the evaluation of modules, e.g.

```
./turnoff box      (turn off the boxes)
./turnoff          (restore)
```

- `pnuglot` makes a customizable high-quality plot from a data file.
- `submit` automatically distributes a job on a cluster.



Distributing a Job

Parameter scans can automatically be distributed on a cluster of computers:

- The machines are declared in a file `.submitrc`, e.g.

```
# Optional: Nice to start jobs with
nice 10
# Pentium 4 3000
pcl301
pcl301a
pcl305
# Dual Xeon 2660
pcl247b   2
pcl319a   2
pcl321    2
...
```

- The command line for distributing a job is *exactly the same* except that `submit` is prepended, e.g.

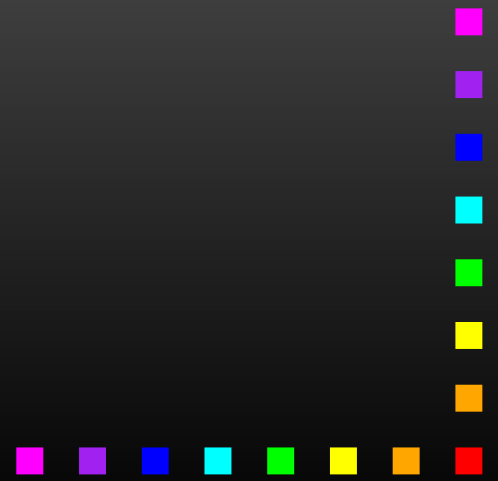
```
submit run uuuu 0,1000
```

HadCalc

HadCalc is a new front-end for FormCalc, i.e. it uses the generated Fortran code with a custom set of driver programs.

- Automates the calculation of hadronic cross-sections,
- Includes convolution with PDFs,
- Various cuts can be applied,
- Operates in interactive or batch mode.

HadCalc is not (yet) public. It can currently be obtained from Michael Rauch <mrauch@mppmu.mpg.de>.



Summary

New FormCalc Features:

- Weyl-van der Waerden spinor formalism radically simplifies calculations with external fermions.
- CUBA Library provides four independent algorithms for multidimensional numerical integration.
  Version 1.2 built into FormCalc or available separately at http://www.feynarts.de/cuba (LGPL).
- Public functions for generating optimized Fortran code from an arbitrary Mathematica expression.
- FeynHiggs Interface and NMFV available.
- Useful shell scripts, simple parallelization mechanism.
- HadCalc is an alternate front-end for FormCalc for hadronic reactions.