FeynHiggs 2.11.0

Application Programming Interface

FHSetFlags

Description

`FHSetFlags` sets the flags for FeynHiggs. It must be called before any other FeynHiggs function.

Synopsis – Fortran version

```
integer error
integer mssmpart, fieldren, tanbren, higgsmix, p2approx
integer looplevel, runningMT, botResum, tlCplxApprox

subroutine FHSetFlags(error,
  mssmpart, fieldren, tanbren, higgsmix, p2approx,
  looplevel, runningMT, botResum, tlCplxApprox)
```

Synopsis – Mathematica version

```
FHSetFlags[mssmpart, fieldren, tanbren, higgsmix, p2approx,
  looplevel, runningMT, botResum, tlCplxApprox]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SetFlags.F` from which the error message was emitted.

IN `integer mssmpart` = $0 \ldots 4$

specifies the scope of the calculation:

0: m_t^4-approximation,

1: top/stop-sector,

2: top/stop- + bottom/sbottom-sector,

3: full (s)quark/(s)lepton-sector,

4: full MSSM *(recommended)*.

IN `integer fieldren` = $0\ldots4$

determines the one-loop field-renormalization constants:

0: $\overline{\text{DR}}$ *(strongly recommended)*,

1: on-shell, Dabelstein's convention,

2: on-shell, 'Goldstone-pole' version,

3: on-shell, MF I,

4: on-shell, MF II.

IN `integer tanbren` = $0\ldots2$

determines the one-loop $\tan\beta$ counter term:

0: $\overline{\text{DR}}$ *(strongly recommended)*,

1: field renormalization part only (Dabelstein),

2: mixed field/on-shell-renormalization.

IN `integer higgsmix` = $1\ldots3$

determines the mixing in the Higgs sector:

1: all non-diagonal self-energies ($\Sigma_{hH}, \Sigma_{hA}, \Sigma_{HA}$) are set to zero,

2: the CP-violating non-diagonal self-energies (Σ_{hA}, Σ_{HA}) are set to zero, i.e. 2×2 mixing among CP-even states = evaluation in the rMSSM,

3: full 3×3 mixing in the neutral sector = evaluation in the cMSSM.

IN `integer p2approx` = $0\ldots4$

determines the approximation for the one-loop result:

0: none, i.e. full determination of the propagator matrices's poles, `UHiggs` is evaluated at $p^2 = m^2$,

1: $p^2 = 0$ approximation,

2: all self-energies are calculated at $p^2 = m^2_{\text{tree}}$,

3: imaginary parts of self-energies are discarded,

4: as 0, but with `UHiggs` evaluated at $p^2 = 0$ *(recommended)*.

IN `integer looplevel` = $0\ldots3$

determines the inclusion of higher-order corrections:

0: tree level,

1: one-loop contributions only,

2: also include various two-loop contributions *(recommended)*,

3: also include resummation of large logs.

IN `integer runningMT` = $0 \ldots 1$

determines which top mass shall be used in the 1-/2-loop corrections

0: use m_t^{pole},

1: use m_t^{run} *(recommended)*.

IN `integer botResum` = $0 \ldots 1$

determines whether the $O(\tan^n \beta)$ corrections shall be resummed:

0: no resummation,

1: resummation *(recommended)*.

IN `integer tlCplxApprox` = $0 \ldots 6$

determines how the two-loop corrections are treated in the presence of complex parameters (cMSSM):

0: all corrections ($\alpha_s\alpha_t$, $\alpha_s\alpha_b$, α_t^2, $\alpha_t\alpha_b$) are computed in the rMSSM *(recommended for evaluation in the rMSSM)*,

1: only the cMSSM $\alpha_s\alpha_t$ and α_t^2 corrections are used,

2: the cMSSM $\alpha_s\alpha_t$ and α_t^2 corrections are combined with the remaining corrections in the rMSSM,

3: the cMSSM $\alpha_s\alpha_t$ and α_t^2 corrections are combined with the remaining corrections, whose complex phases are interpolated in A_t, A_b, M_3, μ,

4: ditto, with interpolation in X_t, A_b, M_3, μ,

5: ditto, with interpolation in A_t, X_b, M_3, μ,

6: ditto, with interpolation in X_t, X_b, M_3, μ.

FHSetFlagsString

Description

`FHSetFlagsString` sets the flags for FeynHiggs. It works just like `FHSetFlags` except that it takes a 9-character string as argument rather than 9 integers.

Synopsis – Fortran version

```
integer error
character*9 flags

subroutine FHSetFlagsString(error, flags)
```

Synopsis – Mathematica version

```
FHSetFlagsString[flags]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SetFlags.F` from which the error message was emitted.

IN `character*9 flags`

the flags in the same order and with the same values as in `FHSetFlags`.

FHRetrieveFlags

Description

`FHRetrieveFlags` retrieves the flags from FeynHiggs.

Synopsis – Fortran version

```
integer error
integer mssmpart, fieldren, tanbren, higgsmix, p2approx
integer looplevel, runningMT, botResum, tlCplxApprox

subroutine FHRetrieveFlags(error,
  mssmpart, fieldren, tanbren, higgsmix, p2approx,
  looplevel, runningMT, botResum, tlCplxApprox)
```

Synopsis – Mathematica version

```
FHRetrieveFlags[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `RetrieveFlags.F` from which the error message was emitted.

OUT `integer mssmpart` = $0 \ldots 4$

specifies the scope of the calculation:

0: m_t^4-approximation,

1: top/stop-sector,

2: top/stop- + bottom/sbottom-sector,

3: full (s)quark/(s)lepton-sector,

4: full MSSM.

OUT `integer fieldren` = $0 \ldots 4$

determines the one-loop field-renormalization constants:

0: $\overline{\text{DR}}$,

1: on-shell, Dabelstein's convention,

2: on-shell, 'Goldstone-pole' version,

3: on-shell, MF I,

4: on-shell, MF II.

OUT `integer tanbren` = $0 \dots 2$

determines the one-loop $\tan\beta$ counter term:

0: $\overline{\text{DR}}$,

1: field renormalization part only (Dabelstein),

2: mixed field/on-shell-renormalization.

OUT `integer higgsmix` = $1 \dots 3$

determines the mixing in the Higgs sector:

1: all non-diagonal self-energies ($\Sigma_{hH}, \Sigma_{hA}, \Sigma_{HA}$) are set to zero,

2: the CP-violating non-diagonal self-energies (Σ_{hA}, Σ_{HA}) are set to zero, i.e. 2×2 mixing among CP-even states = evaluation in the rMSSM,

3: full 3×3 mixing in the neutral sector = evaluation in the cMSSM.

OUT `integer p2approx` = $0 \dots 2$

determines the approximation for the one-loop result:

0: none, i.e. full determination of the propagator matrices's poles,

1: $p^2 = 0$ approximation,

2: all self-energies are calculated at $p^2 = m^2_{\text{tree}}$,

3: imaginary parts of self-energies are discarded,

4: `UHiggs` is evaluated at $p^2 = 0$.

OUT `integer looplevel` = $0 \dots 2$

determines the inclusion of higher-order corrections:

0: tree level,

1: one-loop contributions only,

2: include various two-loop contributions.

$\boxed{\text{OUT}}$ `integer runningMT` $= 0 \dots 1$

determines which top mass shall be used in the 1-/2-loop corrections

0: use m_t^{pole},

1: use m_t^{run}.

$\boxed{\text{OUT}}$ `integer botResum` $= 0 \dots 1$

determines whether the $O(\tan^n \beta)$ corrections shall be resummed:

0: no resummation,

1: resummation.

$\boxed{\text{OUT}}$ `integer tlCplxApprox` $= 0 \dots 6$

determines how the two-loop corrections are treated in the presence of complex parameters (cMSSM):

0: all corrections ($\alpha_s\alpha_t$, $\alpha_s\alpha_b$, $\alpha_t\alpha_t$, $\alpha_t\alpha_b$) are computed in the rMSSM,

1: only the cMSSM $\alpha_s\alpha_t$ corrections are used,

2: the cMSSM $\alpha_s\alpha_t$ corrections are combined with the remaining corrections in the rMSSM,

3: the cMSSM $\alpha_s\alpha_t$ corrections are combined with the remaining corrections, whose complex phases are interpolated in A_t, A_b, M_3, μ,

4: ditto, with interpolation in X_t, A_b, M_3, μ,

5: ditto, with interpolation in A_t, X_b, M_3, μ,

6: ditto, with interpolation in X_t, X_b, M_3, μ.

FHRetrieveFlagsString

Description

`FHRetrieveFlagsString` retrieves the flags from FeynHiggs. It works just like `FHRetrieveFlags` except that it takes a 9-character string as argument rather than 9 integers.

Synopsis – Fortran version

```
integer error
character*9 flags

subroutine FHRetrieveFlagsString(error, flags)
```

Synopsis – Mathematica version

```
FHRetrieveFlagsString[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `RetrieveFlags.F` from which the error message was emitted.

OUT `character*9 flags`

the flags in the same order and with the same values as in `FHRetrieveFlags`.

FHSetDebug

Description

`FHSetDebug` sets the debugging level for FeynHiggs.

Synopsis – Fortran version

```
integer debuglevel

subroutine FHSetDebug(debuglevel)
```

Synopsis – Mathematica version

```
FHSetDebug[debuglevel]
```

Arguments

IN `integer debuglevel`

the new debugging level, where

0: no debugging messages,

1: dump `FHSetFlags` and `FHSetPara` values,

2: echo input parameters in detail, display the Higgs mass matrix at $p^2 = 0$ and the counter-terms,

3: display the search for zeros of the Higgs propagator matrix.

FHSetSMPara

Description

FHSetSMPara sets up the SM inputs for FeynHiggs. All of these parameters have default values so it is optional to call FHSetSMPara. If FHSetSMPara is called, it must be called before FHSetPara.

Substituting -1 for any argument uses its default value.

Synopsis – Fortran version

```
integer error
double precision invAlfa, AlfasMZ, GF
double precision ME, MU, MD, MM, MC, MS, ML, MB
double precision MW, MZ
double precision CKMlambda, CKMA, CKMrhobar, CKMetabar

subroutine FHSetSMPara(error,
  invAlfa, AlfasMZ, GF,
  ME, MU, MD, MM, MC, MS, ML, MB,
  MW, MZ,
  CKMlambda, CKMA, CKMrhobar, CKMetabar)
```

Synopsis – Mathematica version

```
FHSetSMPara[invAlfa, AlfasMZ, GF,
  ME, MU, MD, MM, MC, MS, ML, MB,
  MW, MZ,
  CKMlambda, CKMA, CKMrhobar, CKMetabar]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in SetSMPara.F from which the error message was emitted.

IN `double precision invAlfa`

the value of electromagnetic coupling constant α^{-1}, or -1 for the default value.

IN `double precision AlfasMZ`

the value of the strong coupling constant $\alpha_s(M_Z)$, or -1 for the default value.

IN `double precision GF`

the value of the Fermi constant G_F, or -1 for the default value.

IN `double precision ME, MM, ML`

the on-shell electron, muon, tauon mass, or -1 for the default values.

IN `double precision MU, MD, MS`

the up, down, and strange quark masses at 2 GeV, or -1 for the default values.

IN `double precision MC, MB`

the charm mass at m_c and bottom mass at m_b, or -1 for the default values.

IN `MW, MZ`

the W and Z masses, or -1 for the default values.

IN `double precision CKMlambda, CKMA, CKMrhobar, CKMetabar`

the CKM input parameters λ, A, $\bar{\rho}$, and $\bar{\eta}$ in Wolfenstein parameterization, or -1 for the default values. The CKM matrix is computed as

$$\text{CKM} = \begin{pmatrix} c_{12}c_{13} & s_{12}c_{13} & s_{13}^* \\ -s_{12}c_{23} - c_{12}s_{23}s_{13} & c_{12}c_{23} - s_{12}s_{23}s_{13} & s_{23}c_{13} \\ s_{12}s_{23} - c_{12}s_{23}s_{13} & -c_{12}s_{23} - s_{12}c_{23}s_{13} & c_{23}c_{13} \end{pmatrix}$$

where $s_{12} = \lambda$, $s_{23} = A\lambda^2$, $s_{13} = \dfrac{A\lambda^3(\bar{\rho} + \mathrm{i}\bar{\eta})\sqrt{1 - A^2\lambda^4}}{\sqrt{1 - \lambda^2}(1 - A^2\lambda^4(\bar{\rho} + \mathrm{i}\bar{\eta})}$ and $c_{ij} = \sqrt{1 - |s_{ij}|^2}$.

FHRetrieveSMPara

Description

`FHRetrieveSMPara` retrieves the SM input parameters from FeynHiggs.

Synopsis – Fortran version

```
integer error
double precision invAlfa, AlfasMZ, GF
double precision ME, MU, MD, MM, MC, MS, ML, MB
double precision MW, MZ
double precision CKMlambda, CKMA, CKMrhobar, CKMetabar

subroutine FHRetrieveSMPara(error,
  invAlfa, AlfasMZ, GF,
  ME, MU, MD, MM, MC, MS, ML, MB,
  MW, MZ,
  CKMlambda, CKMA, CKMrhobar, CKMetabar)
```

Synopsis – Mathematica version

```
FHRetrieveSMPara[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `RetrieveSMPara.F` from which the error message was emitted.

OUT `double precision invAlfa`

the value of electromagnetic coupling constant α^{-1}.

OUT `double precision AlfasMZ`

the value of the strong coupling constant $\alpha_s(M_Z)$.

OUT `double precision GF`

the value of the Fermi constant G_F.

$\boxed{\text{OUT}}$ `double precision ME, MM, ML`

the on-shell electron, muon, tauon mass.

$\boxed{\text{OUT}}$ `double precision MU, MD, MS`

the up, down, and strange quark masses at 2 GeV.

$\boxed{\text{OUT}}$ `double precision MC, MB`

the charm mass at m_c and bottom mass at m_b.

$\boxed{\text{OUT}}$ `double precision MW, MZ`

the W and Z masses.

$\boxed{\text{OUT}}$ `double precision CKMlambda, CKMA, CKMrhobar, CKMetabar`

the CKM input parameters λ, A, $\bar{\rho}$, $\bar{\eta}$ in Wolfenstein parameterization.

FHGetSMPara

Description

FHGetSMPara returns the SM parameters computed from the input values by FHSetSMPara. The flags and SM parameters must have been set before with FHSetFlags and FHSetSMPara.

Synopsis – Fortran version

```
integer error
double complex CKM(3,3)

subroutine FHGetSMPara(error, CKM)
```

Synopsis – Mathematica version

```
FHGetSMPara[]
```

Arguments

OUT integer error

zero if successful, otherwise the line number in GetSMPara.F from which the error message was emitted.

OUT double complex CKM(g_1,g_2)

the CKM matrix.

FHSetPara

Description

FHSetPara sets up the parameters for FeynHiggs. From the given input values it computes the remaining MSSM parameters (masses and mixing matrices). The flags must have been set before with FHSetFlags.

Synopsis – Fortran version

```
integer error
double precision scalefactor
double precision MT, TB, MA0, MHp
double precision M3SL, M3SE, M3SQ, M3SU, M3SD
double precision M2SL, M2SE, M2SQ, M2SU, M2SD
double precision M1SL, M1SE, M1SQ, M1SU, M1SD
double complex MUE, M_1, M_2, M_3
double complex At, Ab, Atau, Ac, As, Amu, Au, Ad, Ae
double precision Qtau, Qt, Qb

subroutine FHSetPara(error, scalefactor,
  MT, TB, MA0, MHp,
  M3SL, M3SE, M3SQ, M3SU, M3SD,
  M2SL, M2SE, M2SQ, M2SU, M2SD,
  M1SL, M1SE, M1SQ, M1SU, M1SD,
  MUE,
  Atau, At, Ab, Amu, Ac, As, Ae, Au, Ad,
  M_1, M_2, M_3,
  Qtau, Qt, Qb)
```

Synopsis – Mathematica version

```
FHSetPara[scalefactor,
  MT, TB, MA0, MHp,
  M3SL, M3SE, M3SQ, M3SU, M3SD,
  M2SL, M2SE, M2SQ, M2SU, M2SD,
  M1SL, M1SE, M1SQ, M1SU, M1SD,
  MUE,
  Atau, At, Ab, Amu, Ac, As, Ae, Au, Ad,
```

```
M_1, M_2, M_3,
Qtau, Qt, Qb]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SetPara.F` from which the error message was emitted.

IN `double precision scalefactor`

the renormalization scale is m_t times the `scalefactor`.

IN `double precision MT`

the top-quark mass.

IN `double precision TB`

the ratio of the Higgs vacuum expectation values, $\tan\beta$.

IN `double precision MA0, MHp`

the masses of the CP-odd and charged Higgs, respectively. Only one should be given: if `MA0` $\geqslant 0$, `MA0` is taken as input, otherwise `MHp` is used.

IN `double precision M`g`SL, M`g`SE, M`g`SQ, M`g`SU, M`g`SD,` $g = 1\dots 3$

the soft-SUSY breaking parameters for the g-th generation in the sfermion sector, specifically: MgSL for the slepton doublet, MgSE for the slepton singlet, MgSQ for the squark doublet, MgSU for the up-type squark singlet, and MgSD for the down-type squark singlet.

IN `double complex MUE`

the Higgs mixing parameter μ.

IN `double complex Ae, Amu, Atau, Au, Ac, At, Ad, As, Ab`

the soft-SUSY breaking parameters. To give an example (and thus fix the notation) the stop mass matrix is given by ($D_t^{1,2}$ are the D-terms):

$$\begin{pmatrix} \mathtt{M3SQ}^2 + \mathtt{MT}^2 + D_t^1 & \mathtt{MT}\,(\mathtt{At}^* - \mathtt{MUE}/\mathtt{TB}) \\ \mathtt{MT}\,(\mathtt{At} - \mathtt{MUE}^*/\mathtt{TB}) & \mathtt{M3SU}^2 + \mathtt{MT}^2 + D_t^2 \end{pmatrix}$$

IN `double complex M_1, M_2, M_3`

the gaugino mass parameters. If zero is passed for `M_1`, the GUT relation is used.

$\boxed{\text{IN}}$ `double precision Qtau, Qt, Qb`

the scales at which the sfermion input parameters `M3S{L,E,Q,U,D}` are given. There are two special cases:

- The value 0 indicates on-shell parameters.
- The value -1 selects the scale $\sqrt{\tilde{m}_t^1 \tilde{m}_t^2}$. The procedure is: compute the sfermion masses from the given input parameter, run them to the on-shell scale, and extract the on-shell input parameters from the latter.

`Qtau` is presently not used.

FHRetrievePara

Description

`FHRetrievePara` retrieves the input parameters from FeynHiggs.

Synopsis – Fortran version

```
integer error
double precision scalefactor
double precision MT, TB, MA0, MHp
double precision M3SL, M3SE, M3SQ, M3SU, M3SD
double precision M2SL, M2SE, M2SQ, M2SU, M2SD
double precision M1SL, M1SE, M1SQ, M1SU, M1SD
double complex MUE, M_1, M_2, M_3
double complex At, Ab, Atau, Ac, As, Amu, Au, Ad, Ae
double precision Qtau, Qt, Qb

subroutine FHRetrievePara(error, scalefactor,
  MT, TB, MA0, MHp,
  M3SL, M3SE, M3SQ, M3SU, M3SD,
  M2SL, M2SE, M2SQ, M2SU, M2SD,
  M1SL, M1SE, M1SQ, M1SU, M1SD,
  MUE,
  Atau, At, Ab, Amu, Ac, As, Ae, Au, Ad,
  M_1, M_2, M_3,
  Qtau, Qt, Qb)
```

Synopsis – Mathematica version

```
FHRetrievePara[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `RetrievePara.F` from which the error message was emitted.

OUT `double precision scalefactor`

the renormalization scale is m_t times the `scalefactor`.

OUT `double precision MT`

the top-quark mass.

OUT `double precision TB`

the ratio of the Higgs vacuum expectation values, $\tan\beta$.

OUT `double precision MA0, MHp`

the masses of the CP-odd and charged Higgs, respectively.

OUT `double precision M`g`SL, M`g`SE, M`g`SQ, M`g`SU, M`g`SD,` $g = 1 \ldots 3$

the soft-SUSY breaking parameters for the g-th generation in the sfermion sector, specifically: MgSL for the slepton doublet, MgSE for the slepton singlet, MgSQ for the squark doublet, MgSU for the up-type squark singlet, and MgSD for the down-type squark singlet.

OUT `double complex MUE`

the Higgs mixing parameter μ.

OUT `double complex Ae, Amu, Atau, Au, Ac, At, Ad, As, Ab`

the soft-SUSY breaking parameters.

OUT `double complex M_1, M_2, M_3`

the gaugino mass parameters.

OUT `double precision Qtau, Qt, Qb`

the scales at which the sfermion input parameters `M3S`{`L`,`E`,`Q`,`U`,`D`} are given.

FHRetrieveOSPara

Description

`FHRetrieveOSPara` retrieves the on-shell input parameters from FeynHiggs.

Synopsis – Fortran version

```
integer error
double precision scalefactor
double precision MT, TB, MA0, MHp
double precision M3SL, M3SE, M3SQ, M3SU, M3SD
double precision M2SL, M2SE, M2SQ, M2SU, M2SD
double precision M1SL, M1SE, M1SQ, M1SU, M1SD
double complex MUE, M_1, M_2, M_3
double complex At, Ab, Atau, Ac, As, Amu, Au, Ad, Ae

subroutine FHRetrieveOSPara(error, scalefactor,
  MT, TB, MA0, MHp,
  M3SL, M3SE, M3SQ, M3SU, M3SD,
  M2SL, M2SE, M2SQ, M2SU, M2SD,
  M1SL, M1SE, M1SQ, M1SU, M1SD,
  MUE,
  Atau, At, Ab, Amu, Ac, As, Ae, Au, Ad,
  M_1, M_2, M_3)
```

Synopsis – Mathematica version

```
FHRetrieveOSPara[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `RetrieveOSPara.F` from which the error message was emitted.

OUT `double precision scalefactor`

the renormalization scale is m_t times the `scalefactor`.

OUT `double precision MT`

the top-quark mass.

OUT `double precision TB`

the ratio of the Higgs vacuum expectation values, $\tan\beta$.

OUT `double precision MA0, MHp`

the masses of the CP-odd and charged Higgs, respectively.

OUT `double precision M`g`SL, M`g`SE, M`g`SQ, M`g`SU, M`g`SD,` $g = 1 \ldots 3$

the soft-SUSY breaking parameters for the g-th generation in the sfermion sector, specifically: MgSL for the slepton doublet, MgSE for the slepton singlet, MgSQ for the squark doublet, MgSU for the up-type squark singlet, and MgSD for the down-type squark singlet.

OUT `double complex MUE`

the Higgs mixing parameter μ.

OUT `double complex Ae, Amu, Atau, Au, Ac, At, Ad, As, Ab`

the soft-SUSY breaking parameters.

OUT `double complex M_1, M_2, M_3`

the gaugino mass parameters.

FHSetSLHA

Description

FHSetSLHA is the companion routine to FHSetPara. It extracts the parameters for FeynHiggs from SUSY Les Houches Accord (SLHA) data. In Fortran, it reads the data from the slhadata array used by the SLHA library, in Mathematica it reads the data from the SLHA file directly. As with FHSetPara, the flags must have been set before with FHSetFlags.

Synopsis – Fortran version

```
#include "SLHA.h"

integer error
double complex slhadata(nslhadata)

subroutine FHSetSLHA(error, slhadata)
```

Synopsis – Mathematica version

```
FHSetSLHA[slhafile]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in SetSLHA.F from which the error message was emitted.

IN `double complex slhadata(`i`)`

the SLHA data structure. The slhadata array should never be accessed directly, but only through the preprocessor macros defined in SLHADefs.h.

IN `String slhafile`

the name of the file from which to read the SLHA data.

FHSetNMFV

Description

`FHSetNMFV` sets the non-minimal flavour-violating parameters, i.e. the off-diagonal entries of the SUSY-breaking parameters $M_{\tilde Q}$, $M_{\tilde U}$ and $M_{\tilde D}$, and $A_{u,d}$.

The 6×6 sfermion mass matrices are

$$\begin{aligned}
M^2_{\tilde u} &= \begin{pmatrix} V M^2_{\tilde Q} V^\dagger & K_u \\ K_u^* & M^2_{\tilde U} \end{pmatrix} + \begin{pmatrix} (\frac{1}{2} - \frac{2}{3} s_W^2) m_Z^2 \cos 2\beta + m_{u_i}^2 & -m_{u_i} \mu^* \cot\beta \\ -m_{u_i} \mu \cot\beta & \frac{2}{3} s_W^2 m_Z^2 \cos 2\beta + m_{u_i}^2 \end{pmatrix}, \\
M^2_{\tilde d} &= \begin{pmatrix} M^2_{\tilde Q} & K_d \\ K_d^* & M^2_{\tilde D} \end{pmatrix} + \begin{pmatrix} (-\frac{1}{2} + \frac{1}{3} s_W^2) m_Z^2 \cos 2\beta + m_{d_i}^2 & -m_{d_i} \mu^* \tan\beta \\ -m_{d_i} \mu \tan\beta & -\frac{1}{3} s_W^2 m_Z^2 \cos 2\beta + m_{d_i}^2 \end{pmatrix},
\end{aligned}$$

where V is the CKM matrix and $(K_q)_{ij} = m_{q_i} (A_q)_{ij}$.

The NMFV parameters are given as dimensionless quantities δ_{ij}^{XY} which are scaled by diagonal entries of the SUSY-breaking masses $M_{\tilde Q, \tilde U, \tilde D}$:

$$\begin{aligned}
(M^2_{\tilde Q})_{ij} &= \sqrt{(M^2_{\tilde Q})_{ii} (M^2_{\tilde Q})_{jj}}\, \delta_{ij}^{\rm LL}\,, \qquad && i \neq j \\
(M^2_{\tilde R})_{ij} &= \sqrt{(M^2_{\tilde R})_{ii} (M^2_{\tilde R})_{jj}}\, \delta_{q_i q_j}^{\rm RR}\,, \qquad && R = U, D \\
(K_q)_{ij} &= \sqrt{(M^2_{\tilde Q})_{ii} (M^2_{\tilde R})_{jj}}\, \delta_{q_i q_j}^{\rm LR}\,.
\end{aligned}$$

Synopsis – Fortran version

```
integer error
double complex deltaQLL12, deltaQLL23, deltaQLL13
double complex deltaULR12, deltaULR23, deltaULR13
double complex deltaURL12, deltaURL23, deltaURL13
double complex deltaURR12, deltaURR23, deltaURR13
double complex deltaDLR12, deltaDLR23, deltaDLR13
double complex deltaDRL12, deltaDRL23, deltaDRL13
double complex deltaDRR12, deltaDRR23, deltaDRR13

subroutine FHSetNMFV(error,
  deltaQLL12, deltaQLL23, deltaQLL13,
  deltaULR12, deltaULR23, deltaULR13,
  deltaURL12, deltaURL23, deltaURL13,
  deltaURR12, deltaURR23, deltaURR13,
```

```
  deltaDLR12, deltaDLR23, deltaDLR13,
  deltaDRL12, deltaDRL23, deltaDRL13,
  deltaDRR12, deltaDRR23, deltaDRR13)
```

Synopsis – Mathematica version

```
FHSetNMFV[
  deltaQLL12, deltaQLL23, deltaQLL13,
  deltaULR12, deltaULR23, deltaULR13,
  deltaURL12, deltaURL23, deltaURL13,
  deltaURR12, deltaURR23, deltaURR13,
  deltaDLR12, deltaDLR23, deltaDLR13,
  deltaDRL12, deltaDRL23, deltaDRL13,
  deltaDRR12, deltaDRR23, deltaDRR13]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SetFV.F` from which the error message was emitted.

IN `double complex delta`$FXYij$, F = `Q,U,D`, XY = `LR,RL,RR`, ij = `12,23,13`

the dimensionless off-diagonal NMFV parameters $(\delta^F)_{ij}^{XY}$.

FHRetrieveNMFV

Description

FHRetrieveNMFV retrieves the non-minimal flavour-violating parameters, defined as in FHSetNMFV.

Synopsis – Fortran version

```
integer error
double complex deltaQLL12, deltaQLL23, deltaQLL13
double complex deltaULR12, deltaULR23, deltaULR13
double complex deltaURL12, deltaURL23, deltaURL13
double complex deltaURR12, deltaURR23, deltaURR13
double complex deltaDLR12, deltaDLR23, deltaDLR13
double complex deltaDRL12, deltaDRL23, deltaDRL13
double complex deltaDRR12, deltaDRR23, deltaDRR13

subroutine FHRetrieveNMFV(error,
  deltaQLL12, deltaQLL23, deltaQLL13,
  deltaULR12, deltaULR23, deltaULR13,
  deltaURL12, deltaURL23, deltaURL13,
  deltaURR12, deltaURR23, deltaURR13,
  deltaDLR12, deltaDLR23, deltaDLR13,
  deltaDRL12, deltaDRL23, deltaDRL13,
  deltaDRR12, deltaDRR23, deltaDRR13)
```

Synopsis – Mathematica version

```
FHRetrieveNMFV[]
```

Arguments

$\boxed{\text{OUT}}$ `integer error`

zero if successful, otherwise the line number in RetrieveFV.F from which the error message was emitted.

$\boxed{\text{OUT}}$ `double complex delta`$FXYij$, F = Q,U,D, XY = LR,RL,RR, ij = 12,23,13 the dimensionless off-diagonal NMFV parameters $(\delta^F)_{ij}^{XY}$.

FHSetLFV

Description

`FHSetLFV` sets the lepton-flavour-violating parameters, i.e. the off-diagonal entries of the SUSY-breaking parameters $M_{\tilde{L}}$, $M_{\tilde{E}}$, and A_e. These are defined analogously to the NMFV parameters (see `FHSetNMFV`).

Synopsis – Fortran version

```
integer error
double complex deltaLLL12, deltaLLL23, deltaLLL13
double complex deltaELR12, deltaELR23, deltaELR13
double complex deltaERL12, deltaERL23, deltaERL13
double complex deltaERR12, deltaERR23, deltaERR13

subroutine FHSetLFV(error,
  deltaLLL12, deltaLLL23, deltaLLL13,
  deltaELR12, deltaELR23, deltaELR13,
  deltaERL12, deltaERL23, deltaERL13,
  deltaERR12, deltaERR23, deltaERR13)
```

Synopsis – Mathematica version

```
FHSetLFV[
  deltaLLL12, deltaLLL23, deltaLLL13,
  deltaELR12, deltaELR23, deltaELR13,
  deltaERL12, deltaERL23, deltaERL13,
  deltaERR12, deltaERR23, deltaERR13]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SetFV.F` from which the error message was emitted.

IN `double complex delta`$FXYij$, F = `L,E`, XY = `LR,RL,RR`, ij = `12,23,13`

the dimensionless off-diagonal LFV parameters $(\delta^F)^{XY}_{ij}$.

FHRetrieveLFV

Description

`FHRetrieveLFV` retrieves the lepton-flavour-violating parameters, defined as in `FHSetLFV`.

Synopsis – Fortran version

```
integer error
double complex deltaLLL12, deltaLLL23, deltaLLL13
double complex deltaELR12, deltaELR23, deltaELR13
double complex deltaERL12, deltaERL23, deltaERL13
double complex deltaERR12, deltaERR23, deltaERR13

subroutine FHRetrieveLFV(error,
  deltaLLL12, deltaLLL23, deltaLLL13,
  deltaELR12, deltaELR23, deltaELR13,
  deltaERL12, deltaERL23, deltaERL13,
  deltaERR12, deltaERR23, deltaERR13)
```

Synopsis – Mathematica version

```
FHRetrieveLFV[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `RetrieveFV.F` from which the error message was emitted.

OUT `double complex delta`$FXYij$, F = `L,E`, XY = `LR,RL,RR`, ij = `12,23,13`

the dimensionless off-diagonal LFV parameters $(\delta^F)^{XY}_{ij}$.

FHGetPara

Description

`FHGetPara` returns the sfermion, chargino, and neutralino masses and mixing matrices, and the correction to m_b which were computed from the input parameters by `FHSetPara`. The flags and parameters must have been set before with `FHSetFlags` and `FHSetPara`.

Synopsis – Fortran version

```
integer error, nmfv
double precision MSf(2,5,3), MASf(6,5), MCha(2), MNeu(4)
double complex USf(2,2,5,3), UASf(6,6,5)
double complex UCha(2,2), VCha(2,2), ZNeu(4,4)
double complex Deltab
double precision MGl
double precision MHtree(4), SAtree

subroutine FHGetPara(error, nmfv, MSf, USf, MASf, UASf,
  MCha, UCha, VCha, MNeu, ZNeu, Deltab, MGl,
  MHtree, SAtree)
```

Synopsis – Mathematica version

```
FHGetPara[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `GetPara.F` from which the error message was emitted.

OUT `integer nmfv`

bit $t = 1 \ldots 4$ indicates whether non-minimal flavour violation is active for sfermion type t: ν, e, u, d.

OUT `double precision MSf(`s`,`t`,`g`)`

the MFV squark masses, with indices

$s = 1 \ldots 2$	sfermion index,
$t = 1 \ldots 4$	sfermion type: ν, e, u, d,
$g = 1 \ldots 3$	generation index.

OUT `double complex USf(`s_1`,`s_2`,`t`,`g`)`

the MFV squark mixing matrices, with indices

$s_1 = 1 \ldots 2$	sfermion index (enumerates mass eigenstates),
$s_2 = 1 \ldots 2$	sfermion index (enumerates gauge eigenstates, L/R),
$t = 1 \ldots 4$	sfermion type: ν, e, u, d,
$g = 1 \ldots 3$	generation index.

OUT `double precision MASf(`a`,`t`)`

the NMFV squark masses, with indices

$a = 1 \ldots 6$	extended sfermion index,
$t = 1 \ldots 4$	sfermion type: ν, e, u, d.

OUT `double complex UASf(`a_1`,`a_2`,`t`)`

the NMFV squark mixing matrices, with indices

$a_1 = 1 \ldots 6$	extended sfermion index (enumerates mass eigenstates),
$a_2 = 1 \ldots 6$	extended sfermion index (enumerates gauge eigenstates),
$t = 1 \ldots 4$	sfermion type: ν, e, u, d.

OUT `double precision MCha(`c`)`

the chargino masses, with index

$c = 1 \ldots 2$	chargino index.

OUT `double complex UCha(`c_1`,`c_2`), VCha(`c_1`,`c_2`)`

the chargino mixing matrices, with indices

$c_1 = 1 \ldots 2$	chargino index (enumerates mass eigenstates),
$c_2 = 1 \ldots 2$	chargino index (enumerates gauge eigenstates).

OUT `double precision MNeu(`n`)`

the neutralino masses, with index

$n = 1 \ldots 4$	neutralino index.

OUT `double complex ZNeu(`n_1`,`n_2`)`

the neutralino mixing matrix, with indices

$n_1 = 1 \ldots 4$	neutralino index (enumerates mass eigenstates),
$n_2 = 1 \ldots 4$	neutralino index (enumerates gauge eigenstates).

$\boxed{\text{OUT}}$ `double complex Deltab`

the correction to the bottom Yukawa coupling, Δ_b.

$\boxed{\text{OUT}}$ `double precision MGl`

the gluino mass.

$\boxed{\text{OUT}}$ `double precision MHtree(`h`)`

the tree-level Higgs masses, in the order m_h, m_H, m_A, $m_{H^\pm}$.

$\boxed{\text{OUT}}$ `double precision SAtree`

the tree-level Higgs mixing parameter $\sin\alpha$.

FHGetTLPara

Description

`FHGetTLPara` returns the sfermion parameters used in the one-loop computation of the neutral Higgs masses to be consistent with the two-loop parts.

Do not use this function unless you know precisely what you are doing.

Synopsis – Fortran version

```
integer error
double precision MSb(2), MbSL2
double complex USb(2,2), Deltab

subroutine FHGetTLPara(error, MSb, USb, MbSL2, Deltab)
```

Synopsis – Mathematica version

```
FHGetTLPara[]
```

Arguments

OUT `error`

zero if successful, otherwise the line number in `GetTLPara.F` from which the error message was emitted.

OUT `MSb`

the sbottom masses.

OUT `USb`

the sbottom mixing matrix.

OUT `MbSL2`

the doublet squark soft-breaking mass squared for the sbottom including corrections.

OUT `Deltab`

the correction to the bottom Yukawa coupling, Δ_b.

FHGetFV

Description

`FHGetFV` returns the FV SUSY breaking parameters computed from the deltas set by `FHSetNMFV` and `FHSetLFV`. In the MFV case, the matrices are returned in the same way only that they are diagonal.

Synopsis – Fortran version

```
integer error
double complex MSS2(3,3,5), Kf(3,3,2:4)

subroutine FHGetFV(error, MSS2, Kf)
```

Synopsis – Mathematica version

```
FHGetFV[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `GetFV.F` from which the error message was emitted.

OUT `double complex MSS2(`g_1`,`g_2`,`n`)`

the 3×3 sfermion mass parameter matrices, where the last index n runs over 1 = slepton doublet, 2 = slepton singlet, 3 = squark doublet, 4 = up-type squark singlet, 5 = down-type squark singlet.

OUT `double complex Kf(`g_1`,`g_2`,`t`)`

the 3×3 trilinear coupling matrices multiplied by the corresponding fermion mass, i.e. $K_f = m_f A_f$, where the last index t runs over 2 = slepton, 3 = up-type squark, 4 = down-type squark.

FHGetSelf

Description

FHGetSelf returns the renormalized Higgs self-energies at a given k^2. The flags and parameters must have been set before with FHSetFlags and FHSetPara.

Synopsis – Fortran version

```
integer error
double complex k2
integer key, dkey, ren
double complex sig(13), dsig(13)
integer h0h0, HHHH, A0A0, HmHp
integer h0HH, h0A0, HHA0
integer G0G0, h0G0, HHG0, A0G0
integer GmGp, HmGp
parameter (h0h0 = 1, HHHH = 2, A0A0 = 3, HmHp = 4)
parameter (h0HH = 5, h0A0 = 6, HHA0 = 7)
parameter (G0G0 = 8, h0G0 = 9, HHG0 = 10, A0G0 = 11)
parameter (GmGp = 12, HmGp = 13)

#define Key(se) 2**(se-1)

subroutine FHGetSelf(error, k2, key, sig, dkey, dsig, ren)
```

Synopsis – Mathematica version

```
FHGetSelf[k2, key, dkey, ren]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in GetSelf.F from which the error message was emitted.

IN `k2`

the k^2 at which the self-energies are evaluated.

IN `key`

a flag determining which of the self-energies are actually evaluated, e.g. to evaluate the h^0 self-energy, add `Key(h0h0)` to `key`.

OUT `sig(h0h0), sig(HHHH), sig(A0A0), sig(HmHp)`

the h^0, H^0, A^0, and H^+ self-energies at k^2 = k2.

OUT `sig(h0HH), sig(h0A0), sig(HHA0)`

the h^0-H^0, h^0-A^0, and H^0-A^0 mixing self-energies at k^2 = k2.

OUT `sig(G0G0), sig(h0G0), sig(HHG0), sig(A0G0)`

the neutral Goldstone self-energies at k^2 = k2.

OUT `sig(GmGp), sig(HmGp)`

the charged Goldstone self-energies at k^2 = k2.

IN `dkey`

a flag determining which of the derivatives of the self-energies are actually evaluated, e.g. to evaluate the derivative of the h^0 self-energy, add `Key(h0h0)` to `dkey`.

OUT `dsig(i)`

the derivatives of the self-energies with respect to k^2 at k^2 = k2, where the index `i` runs as for the `sig(i)`.

IN `ren`

whether the unrenormalized (0) or renormalized (1) self-energies are output.

FHAddSelf

Description

`FHAddSelf` allows the user to register shifts in the Higgs self-energies, to be used in the computation of the Higgs masses and mixings in `FHHiggsCorr`. The flags and parameters must have been set before with `FHSetFlags` and `FHSetPara`.

Synopsis – Fortran version

```
integer error, rotate
double complex sig(13)
integer h0h0, HHHH, A0A0, HmHp
integer h0HH, h0A0, HHA0
integer G0G0, h0G0, HHG0, A0G0
integer GmGp, HmGp
parameter (h0h0 = 1, HHHH = 2, A0A0 = 3, HmHp = 4)
parameter (h0HH = 5, h0A0 = 6, HHA0 = 7)
parameter (G0G0 = 8, h0G0 = 9, HHG0 = 10, A0G0 = 11)
parameter (GmGp = 12, HmGp = 13)

subroutine FHAddSelf(error, sig, rotate)
```

Synopsis – Mathematica version

```
FHAddSelf[{sig[h0h0], ...}, rotate]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `AddSelf.F` from which the error message was emitted.

IN `sig(h0h0), sig(HHHH), sig(A0A0), sig(HmHp)`

the h^0, H^0, A^0, and H^+ self-energy shifts.

IN `sig(h0HH), sig(h0A0), sig(HHA0)`

the h^0-H^0, h^0-A^0, and H^0-A^0 mixing self-energy shifts.

IN `sig(G0G0), sig(h0G0), sig(HHG0), sig(A0G0)`

the neutral Goldstone self-energy shifts.

IN `sig(GmGp), sig(HmGp)`

the charged Goldstone self-energy shifts.

IN `rotate`

a flag determining whether to rotate the CP-even self-energies with the (tree-level) angle α, i.e. if `rotate` $\neq 0$, the `h0h0`, `HHHH`, and `h0HH` elements of the input array `sig` are respectively assumed to contain the Φ_1-Φ_1, Φ_2-Φ_2, and Φ_1-Φ_2 shifts.

FHHiggsCorr

Description

`FHHiggsCorr` computes the values of the MSSM Higgs masses according to the given parameters and flags. These must have been set before with `FHSetFlags` and `FHSetPara`.

Synopsis – Fortran version

```
integer error
double precision MHiggs(4)
double complex SAeff, UHiggs(3,3), ZHiggs(3,3)

subroutine FHHiggsCorr(error, MHiggs, SAeff, UHiggs, ZHiggs)
```

Synopsis – Mathematica version

```
FHHiggsCorr[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `HiggsCorr.F` from which the error message was emitted.

OUT `double precision MHiggs(`i`)`

the Higgs masses, where
`MHiggs(1)` = m_1 (= m_h in the rMSSM),
`MHiggs(2)` = m_2 (= m_H in the rMSSM),
`MHiggs(3)` = m_3 (= m_A in the rMSSM),
`MHiggs(4)` = $m_{H^\pm}$,

OUT `double complex SAeff`

the sine of the effective Higgs mixing angle, α_{eff}. With the knowledge of the full mixing matrix `UHiggs`, this is of course a somewhat redundant output.

OUT `double complex UHiggs`

the matrix needed to rotate the Higgs mass matrix to its diagonal form.

OUT `double complex ZHiggs`

the matrix of Z-factors needed to combine amplitudes involving on-shell Higgs bosons.

FHUncertainties

Description

`FHUncertainties` computes estimates for the Higgs masses and mixings. Currently three effects are taken into account:

1. the variation of the renormalization scale from $m_t/2$ to $2m_t$,
2. the use of m_t^{pole} instead of m_t^{run} in the two-loop corrections (only if the `tl_running_mt` flag is set, of course), and
3. the exclusion of higher-order resummation effects in m_b.

Synopsis – Fortran version

```
integer error
double precision DeltaMHiggs(4)
double complex DeltaSAeff, DeltaUHiggs(3,3), DeltaZHiggs(3,3)

subroutine FHUncertainties(error,
  DeltaMHiggs, DeltaSAeff, DeltaUHiggs, DeltaZHiggs)
```

Synopsis – Mathematica version

```
FHUncertainties[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `.F` from which the error message was emitted.

OUT `double precision DeltaMHiggs(`i`)`

the uncertainties of the Higgs masses, where
`DeltaMHiggs(1)` = Δm_1 (= Δm_h in the rMSSM),
`DeltaMHiggs(2)` = Δm_2 (= Δm_H in the rMSSM),
`DeltaMHiggs(3)` = Δm_3 (= Δm_A in the rMSSM),
`DeltaMHiggs(4)` = $\Delta m_{H^\pm}$,

OUT `double complex DeltaSAeff`

the uncertainty of the sine of the effective Higgs mixing angle,

OUT `double complex DeltaUHiggs, DeltaZHiggs`

the (component-wise) uncertainties of the Higgs mixing matrices `UHiggs` and `ZHiggs`.

FHCouplings

Description

`FHCouplings` computes the Higgs couplings, decay widths, and branching ratios. It uses the Higgs masses and mixings computed during the last invocation of `FHHiggsCorr`. The flags and parameters must have been set before with `FHSetFlags` and `FHSetPara` or `FHSetSLHA`.

The arrays passed to FHCouplings should never be accessed directly, but only through the preprocessor macros defined in `FHCouplings.h`, which needs to be included once per file.

Synopsis – Fortran version

```
#include "FHCouplings.h"

integer error
double complex couplings(ncouplings), couplingsms(ncouplingsms)
double precision gammas(ngammas), gammasms(ngammasms)
integer fast

subroutine FHCouplings(error,
  couplings, couplingsms,
  gammas, gammasms, fast)
```

Synopsis – Mathematica version

```
FHCouplings[fast:1]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `Couplings.F` from which the error message was emitted.

OUT `double complex couplings(`i`)`

the MSSM Higgs couplings. This array is indexed with preprocessor macros (see below).

OUT `double complex couplingsms(`i`)`

the couplings of a Standard Model Higgs with the same mass as the respective MSSM Higgs. This array is indexed with preprocessor macros (see below).

OUT `double precision gammas(`i`)`

the Higgs decay widths and branching ratios. This array is indexed with preprocessor macros (see below).

OUT `double precision gammasms(`i`)`

the decay widths and branching ratios of a Standard Model Higgs with the same mass as the respective MSSM Higgs. This array is indexed with preprocessor macros (see below).

OUT `integer fast`

a flag indicating whether the off-diagonal fermion decays shall be computed. Unless the decays $h_i \to f_j f_k$ ($j \neq k$) are explicitly needed, they may safely be ignored, as they hardly contribute to the total cross-section.

Total Decay Widths

The following quantities are implemented as preprocessor macros and map onto the `gammas` and `gammasms` arrays. The latter arrays should never be used directly. The macros are defined in `FHCouplings.h`.

- `double precision GammaTot(`h`)`, $h = 1 \ldots 4$

  The total width of the MSSM Higgs boson, where h enumerates h_1 (h_0 in the rMSSM), h_2 (H_0 in the rMSSM), h_3 (A_0 in the rMSSM), and $H^\pm$.

- `double precision GammaSMTot(`h`)`, $h = 1 \ldots 3$

  The total width of a Standard Model Higgs boson with the same mass as the respective MSSM Higgs boson, where h enumerates h_1 (h_0 in the rMSSM), h_2 (H_0 in the rMSSM), and h_3 (A_0 in the rMSSM).

Couplings, Partial Decay Widths, and Branching Ratios

The following quantities are implemented as preprocessor macros and map onto the `couplings`, `couplingsms`, `gammas`, and `gammasms` arrays. The latter arrays should never be used directly. The macros are defined in `FHCouplings.h`.

- `double complex Coupling(`c`)`

  The coupling for the scalar or vector channel c.

- `double complex LCoupling(`c`), RCoupling(`c`)`

  The left- and right-handed couplings for the fermionic channel c, i.e. the coupling is

  $$\texttt{LCoupling}(c)\,\frac{1-\gamma_5}{2} + \texttt{RCoupling}(c)\,\frac{1+\gamma_5}{2}\,.$$

  Equating this to $S + \mathrm{i}\gamma_5 P$, the scalar and pseudo-scalar coefficients are trivially obtained as

  $$\begin{aligned} S &= \frac{1}{2}\left(\texttt{RCoupling}(c) + \texttt{LCoupling}(c)\right), \\ P &= \frac{1}{2\mathrm{i}}\left(\texttt{RCoupling}(c) - \texttt{LCoupling}(c)\right). \end{aligned}$$

  The couplings are given in the conventions of the MSSM model file of FeynArts. For couplings where the order is essential, the following rules apply:

  - A charged Higgs at position 1 is always the particle, i.e. H^-, this fixes all other particles by charge conservation.
  - For the `H0ChaCha` coupling, the first Chargino is the particle, χ^-, and the second one the antiparticle, χ^+.

- `double complex CouplingSM(`c`), LCouplingSM(`c`), RCouplingSM(`c`)`

  The coupling for the corresponding SM channel.

- `double precision Gamma(`c`), BR(`c`)`

  The width and branching ratio for channel c.

- `double precision GammaSM(`c`), BRSM(`c`)`

  The width and branching ratio of the corresponding SM channel.

Channels

The following quantities are implemented as preprocessor macros and evaluate to an integer which indexes the appropriate arrays. The macros are defined in `FHCouplings.h`. For example, `BR(H0FF(1,3,3,3))` extracts the $h_0 \to t\bar{t}$ branching ratio.

- `H0VV(`h`,`vv`)`

  Neutral Higgs to Vector + Vector, where

$h = 1 \ldots 3$	Higgs: h_1 (h_0), h_2 (H_0), h_3 (A_0),
$vv = 1 \ldots 5$	vector-boson pair: $\gamma\gamma$, γZ, ZZ, WW, gg.

- `H0FF(`h`,`t`,`g_1`,`g_2`)`

  Neutral Higgs to Fermion + Fermion, where

$h = 1 \ldots 3$	Higgs: h_1 (h_0), h_2 (H_0), h_3 (A_0),
$t = 1 \ldots 4$	fermion type: ν, e, u, d,
$g_1 = 1 \ldots 3$	generation of fermion 1.
$g_2 = 1 \ldots 3$	generation of fermion 2.

- `HpFF(`p`,`g_1`,`g_2`)`

  Charged Higgs to Fermion + Fermion, where

$p = 1 \ldots 2$	decay products: leptons (ν/e), quarks (u/d),
$g_1 = 1 \ldots 3$	up-type fermion 1 generation,
$g_2 = 1 \ldots 3$	down-type fermion 2 generation.

- `H0ChaCha(`h`,`c_1`,`c_2`)`

  Neutral Higgs to Chargino + Chargino, where

$h = 1 \ldots 3$	Higgs: h_1 (h_0), h_2 (H_0), h_3 (A_0),
$c_1 = 1 \ldots 2$	chargino 1,
$c_2 = 1 \ldots 2$	chargino 2.

- `H0NeuNeu(`h`,`n_1`,`n_2`)`

  Neutral Higgs to Neutralino + Neutralino, where

$h = 1 \ldots 3$	Higgs: h_1 (h_0), h_2 (H_0), h_3 (A_0),
$n_1 = 1 \ldots 4$	neutralino 1,
$n_2 = 1 \ldots 4$	neutralino 2.

- `HpNeuCha(`n_1`,`c_2`)`

  Charged Higgs to Neutralino + Chargino, where

$n_1 = 1 \ldots 4$	neutralino,
$c_2 = 1 \ldots 2$	chargino.

- `H0HV(`h`,`hv`)`

  Neutral Higgs to Higgs + Vector, where

$h = 1 \ldots 3$	decaying Higgs: h_1 (h_0), h_2 (H_0), h_3 (A_0),
$hv = 1 \ldots 3$	produced pair: h_1–Z (h_0–Z), h_2–Z (H_0–Z), h_3–Z (A_0–Z).

- `HpHV(`hv`)`

  Charged Higgs to Higgs + Vector, where

$hv = 1 \ldots 3$	produced pair: h_1–W (h_0–W), h_2–W (H_0–W), h_3–W (A_0–W).

- `H0HH(`h`,`h_1`,`h_2`)`

  Neutral Higgs to Higgs + Higgs, where

$h = 1 \ldots 3$	decaying Higgs: h_1 (h_0), h_2 (H_0), h_3 (A_0),
$h_1 = 1 \ldots 4$	produced Higgs 1: h_1 (h_0), h_2 (H_0), h_3 (A_0), $H_\pm$,
$h_2 = 1 \ldots 4$	produced Higgs 1: h_1 (h_0), h_2 (H_0), h_3 (A_0), $H_\pm$.

- `H0SfSf(`h`,`s_1`,`s_2`,`t`,`g`)`

  Neutral Higgs to Sfermion + Sfermion, where

$h = 1 \ldots 3$	Higgs: h_1 (h_0), h_2 (H_0), h_3 (A_0),
$s_1 = 1 \ldots 2$	sfermion 1,
$s_2 = 1 \ldots 2$	sfermion 2,
$t = 1 \ldots 4$	sfermion type: ν, e, u, d,
$g = 1 \ldots 3$	common sfermion generation.

- `HpSfSf(`s_1`,`s_2`,`p`,`g_1`,`g_2`)`

  Charged Higgs to Sfermion + Sfermion, where

$s_1 = 1 \ldots 2$	sfermion 1,
$s_2 = 1 \ldots 2$	sfermion 2,
$p = 1 \ldots 2$	decay products: sleptons ($\tilde{\nu}$/ẽ), squarks (ũ/$\tilde{\text{d}}$),
$g_1 = 1 \ldots 3$	up-type sfermion 1 generation,
$g_2 = 1 \ldots 3$	up-type sfermion 2 generation.

- `tBF(`i`)`

  Top quark to boson + fermion, where

$i = 1 \ldots 2$	W + bottom, charged Higgs + bottom

Mathematica Usage

Mathematica and Fortran share the same names for the channels, but due to the structure of the Mathematica output, the results have to be accessed in a slightly different way.

To access the $h_0 \to t\bar{t}$ decay, for example, one would in Mathematica use

```
couplings = FHCouplings[];
h0ff = Gamma[H0FF] /. couplings;
h0toptop = h0ff[[1,3,3]]
```

while in Fortran the same is done with

```
call FHCouplings(couplings, gammas, gammasms)
h0toptop = Gamma(H0FF(1,3,3))
```

FHSelectUZ

Description

FHSelectUZ chooses which Higgs mixing to use for internal and external Higgs bosons, i.e. in the couplings and the decay rates, and whether resummed masses should be used in the couplings.

Synopsis – Fortran version

```
integer error, uzint, uzext, mfeff

subroutine FHSelectUZ(error, uzint, uzext, mfeff)
```

Synopsis – Mathematica version

```
FHSelectUZ[uzint, uzext, mfeff]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SelectUZ.F` from which the error message was emitted.

IN `integer uzint`

whether to use no mixing (0), UHiggs (1), or ZHiggs (2) for internal Higgs bosons, i.e. in the couplings. Default: 1.

IN `integer uzext`

whether to use no mixing (0), UHiggs (1), or ZHiggs (2) for external Higgs bosons, i.e. in the decay rates. Default: 2.

IN `integer mfeff`

which effective bottom mass to use in the `H0FF`, `HpFF`, `H0SfSf`, `HpSfSf` couplings:

0: $m_b(m_b)$, 1: $\dfrac{m_b(m_b)}{|1+\Delta_b|}$ (default), 2: $\dfrac{m_b(m_H)}{|1+\Delta_b|}$.

FHHiggsProd

Description

`FHHiggsProd` computes approximate Higgs production cross-sections.

For neutral-Higgs production it uses the SM data from Fabio Maltoni's Web page `http://maltoni.home.cern.ch/maltoni/TeV4LHC`, fitted to a function, and multiplies them with the appropriate MSSM/SM ratio of the couplings involved. Specifically,

$$\left.\begin{array}{r}\texttt{bbh}\\ \texttt{btagbh}\end{array}\right\} = \frac{\Gamma^{\text{MSSM}}(h\to\bar bb)}{\Gamma^{\text{SM}}(h\to\bar bb)}\left\{\begin{array}{l}\texttt{bbhSM}\\ \texttt{btagbhSM}\end{array}\right.$$

$$\texttt{tth} = \frac{|c_L|^2+|c_R|^2}{2}\texttt{tthSM}\quad\text{where}\qquad c_{L,R} = \frac{C^{\text{MSSM}}_{L,R}(h,t,t)}{C^{\text{SM}}_{L,R}(h,t,t)}\,,$$

$$\begin{aligned}\texttt{ggh} = \frac{|A^{\text{MSSM}}|^2}{|A_{\text{SM}}|^2}\texttt{gghSM}\qquad\text{where}\qquad A^{\text{MSSM}} &= c_t^{\text{NLO}}c_t^{\text{NNLO}}A_t^{\text{MSSM,LO}}+\\ &\quad c_{b,r}^{\text{NLO}}\,\mathrm{Re}\,A_b^{\text{MSSM,LO}}+c_{b,i}^{\text{NLO}}\,\mathrm{Im}\,A_b^{\text{MSSM,LO}}+\\ &\quad c_{\tilde f}^{\text{NLO}}A_{\tilde f}^{\text{MSSM,LO}}+A_{\text{rest}}^{\text{MSSM,LO}},\\ A^{\text{SM}} &= c_t^{\text{NLO}}A_t^{\text{SM,LO}}+\\ &\quad c_{b,r}^{\text{NLO}}\,\mathrm{Re}\,A_b^{\text{SM,LO}}+c_{b,i}^{\text{NLO}}\,\mathrm{Im}\,A_b^{\text{SM,LO}}+\\ &\quad A_{\text{rest}}^{\text{SM,LO}},\end{aligned}$$

$$\begin{aligned}\texttt{ggh2} = \frac{|A^{\text{MSSM}}|^2}{|A_{\text{SM}}|^2}\texttt{gghSM}\qquad\text{where}\qquad |A^{\text{MSSM}}|^2 &= (k_t^{\text{NLO}}k_t^{\text{NNLO}}-k_{tb}^{\text{NLO}})|A_t^{\text{MSSM,LO}}|^2+\\ &\quad (k_b^{\text{NLO}}-k_{tb}^{\text{NLO}})|A_b^{\text{MSSM,LO}}|^2+\\ &\quad (k_{tb}^{\text{NLO}}-1)|A_t^{\text{MSSM,LO}}+A_b^{\text{MSSM,LO}}|^2+\\ &\quad |A^{\text{MSSM,LO}}|^2,\\ |A^{\text{SM}}|^2 &= (k_t^{\text{NLO}}-k_{tb}^{\text{NLO}})|A_t^{\text{SM,LO}}|^2+\\ &\quad (k_b^{\text{NLO}}-k_{tb}^{\text{NLO}})|A_b^{\text{SM,LO}}|^2+\\ &\quad k_{tb}^{\text{NLO}}|A_t^{\text{SM,LO}}+A_b^{\text{SM,LO}}|^2,\end{aligned}$$

$$\left.\begin{array}{r}\texttt{qqh}\\ \texttt{Wh}\end{array}\right\} = \frac{|C^{\text{MSSM}}(h,W,W)|^2}{|C^{\text{SM}}(h,W,W)|^2}\left\{\begin{array}{l}\texttt{qqhSM}\\ \texttt{WhSM}\end{array}\right.$$

$$\texttt{Zh} = \frac{|C^{\text{MSSM}}(h,Z,Z)|^2}{|C^{\text{SM}}(h,Z,Z)|^2}\texttt{ZhSM}$$

All production cross-sections are 4π cross-sections, with `btagbhTeV` and `btagbhLHC` being the only exceptions. Here $p_T(\text{jet}) > 15$ GeV and $|\eta(\text{jet})| < 2.5$ has been used. (The PDF for these two processes are CTEQ6M.)

For charged-Higgs production, a fit to Tilman Plehn's data, available at the Web site `http://www.ph.ed.ac.uk/~tplehn/charged_higgs`, is used to approximate the cross-section.

`FHHiggsProd` uses the Higgs masses and couplings computed during the last invocation of `FHHiggsCorr` and `FHCouplings`. The flags and parameters must have been set before with `FHSetFlags` and `FHSetPara`/`FHSetSLHA`.

Synopsis – Fortran version

```
#include "FHCouplings.h"

integer error
double precision sqrts, prodxs(nprodxs)

subroutine FHHiggsProd(error, sqrts, prodxs)
```

Synopsis – Mathematica version

```
FHHiggsProd[sqrts]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `HiggsProd.F` from which the error message was emitted.

IN `double precision sqrts`

the collider energy at which the cross-sections are to be computed. Note that not all cross-sections are currently implemented for energies other than 2 TeV (Tevatron) and 14 TeV (LHC).

OUT `double precision prodxs(i)`

the Higgs production cross-sections. This array is indexed with preprocessor macros (see below).

Cross-Sections

All cross-sections are fully inclusive and given in fb. They are available for the three neutral Higgs bosons: h_1 (h_0), h_2 (H_0), h_3 (A_0) correspond to $h = 1 \dots 3$ in the following macros which index the `prodxs` array.

- `bbh(`h`)`, `bbhSM(`h`)`

  The MSSM and SM bottom-fusion cross-sections, $bb \to h + X$.

- `btagbh(`h`)`, `btagbhSM(`h`)`

  The MSSM and SM bottom-fusion cross-sections with one tagged b, $b_{\text{tagged}}b \to h + X$.

- `ggh(`h`)`, `gghSM(`h`)`

  The MSSM and SM gluon-fusion cross-sections, $gg \to h + X$, using the c-factor (amplitude) method.

- `ggh2(`h`)`, `ggh2SM(`h`)`

  The MSSM and SM gluon-fusion cross-sections, $gg \to h + X$, using the k-factor (squared-amplitude) method.

- `qqh(`h`)`, `qqhSM(`h`)`

  The MSSM and SM vector-boson-fusion cross-sections, $qq \to qqh + X$.

- `tth(`h`)`, `tthSM(`h`)`

  The MSSM and SM associated top-pair production cross-sections, $qq, gg \to tth + X$.

- `Wh(`h`)`, `WhSM(`h`)`

  The MSSM and SM associated W production cross-sections, $qq \to Wh + X$.

- `Zh(`h`)`, `ZhSM(`h`)`

  The MSSM and SM associated Z production cross-sections, $qq \to Zh + X$.

- `StSth(`h`)`

  The MSSM production cross-section $pp \to \tilde{t}_1\tilde{t}_1 h$ (only $h = 1$).

- `tHm`

  The MSSM production cross-section $gb \to tH^-$.

Mathematica Usage

Mathematica and Fortran share the same names for the cross-sections, but due to the structure of the Mathematica output, the results have to be accessed in a slightly different way.

To access the `qqhLHC` mode, for example, one would use

```
{qqh0, qqHH, qqA0} = qqh /. FHHiggsProd[sqrts]
```

in Mathematica, while in Fortran the same is done with

```
call FHHiggsProd(error, prodxs, sqrts)
qqh0 = qqh(1)
qqHH = qqh(2)
qqA0 = qqh(3)
```

FHConstraints

Description

FHConstraints evaluates electroweak precision observables, currently $(g_\mu - 2)$ and $\Delta\rho$, which are used as further constraints on the MSSM parameter space. Furthermore, the electric dipole moments (EDMs) of the electron (i.e. thorium), the neutron, and mercury are evaluated to constrain the complex parameter space.

Synopsis – Fortran version

```
integer error
double precision gm2
double precision Deltarho, MWMSSM, MWSM, SW2MSSM, SW2SM
double precision edmeTh, edmn, edmHg

subroutine FHConstraints(error, gm2,
  Deltarho, MWMSSM, MWSM, SW2MSSM, SW2SM,
  edmeTh, edmn, edmHg)
```

Synopsis – Mathematica version

```
FHConstraints[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `Constraints.F` from which the error message was emitted.

OUT `double precision gm2`

the anomalous magnetic moment of the muon, $(g_\mu - 2)$.

OUT `double precision Deltarho`

the electroweak precision observable $\Delta\rho$.

OUT `double precision MWMSSM, MWSM, SW2MSSM, SW2SM`

the W mass and effective weak mixing angle in the MSSM and SM, OUT `double precision edmeTh, edmn, edmHg`

electric dipole moments of the electron (derived from Thorium), the neutron, and mercury.

FHFlavour

Description

`FHFlavour` evaluates flavour observables, currently $B \to X_s\gamma$, ΔM_s, and soon $B_s \to \mu^+\mu^-$, which are used as further constraints on the MSSM parameter space.

Synopsis – Fortran version

```
integer error
double precision bsgMSSM, bsgSM
double precision deltaMsMSSM, deltaMsSM
double precision bsmumuMSSM, bsmumuSM

subroutine FHFlavour(error,
  bsgMSSM, bsgSM,
  deltaMsMSSM, deltaMsSM,
  bsmumuMSSM, bsmumuSM)
```

Synopsis – Mathematica version

```
FHFlavour[]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `Constraints.F` from which the error message was emitted.

OUT `double precision bsgammaMSSM, bsgammaSM`

the value of $B \to X_s\gamma$ in the MSSM and SM.

OUT `double precision deltaMsMSSM, deltaMsSM`

the value of ΔM_s in the MSSM and SM.

OUT `double precision bsmumuMSSM, bsmumuSM`

the value of $B_s \to \mu^+\mu^-$ in the MSSM and SM.

FHOutput

Description

FHOutput writes user-specified FeynHiggs inputs and outputs to a file.

Synopsis – Fortran version

```
integer error, key
character*(*) filename
double precision sqrts

subroutine FHOutput(error, filename, key, sqrts)
```

Synopsis – Mathematica version

```
FHOutput[filename, key, sqrts]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in Output.F from which the error message was emitted.

IN `character*(*) filename`

the name of the file to which to write the output. "stdout" and "stderr" are recognized as special filenames.

IN `integer key`

a bit-wise encoding of which output to produce:
bit 0 (1) = input parameters (FHRetrievePara),
bit 1 (2) = derived parameters (FHGetPara),
bit 2 (4) = Higgs masses and mixings (FHHiggsCorr),
bit 3 (8) = uncertainties (FHUncertainties),
bit 4 (16) = decay widths and branching ratios (FHCouplings),
bit 5 (32) = the 'fast' flag of FHCouplings,
bit 6 (64) = electroweak constraints (FHConstraints),
bit 7 (128) = flavour observables (FHFlavour).

FHOutputSLHA

Description

`FHOutputSLHA` adds user-specified FeynHiggs inputs and outputs to an SLHA data structure.

Synopsis – Fortran version

```
#include "SLHA.h"

integer error, key
double complex slhadata(nslhadata)

subroutine FHOutputSLHA(error, slhadata, key)
```

Synopsis – Mathematica version

```
FHOutputSLHA[slhafile, key]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `OutputSLHA.F` from which the error message was emitted.

IN `double complex slhadata(`i`)`

the SLHA data structure. The `slhadata` array should never be accessed directly, but only through the preprocessor macros defined in `SLHADefs.h`.

IN `String slhafile`

the name of the file from which to read the SLHA data.

IN `integer key`

a bit-wise encoding of which parts to fill:
bit 0 (1) = input parameters (`FHRetrievePara`),
bit 1 (2) = derived parameters (`FHGetPara`),
bit 2 (4) = Higgs masses and mixings (`FHHiggsCorr`),

bit 3 (8) = uncertainties (`FHUncertainties`),
bit 4 (16) = decay widths and branching ratios (`FHCouplings`),
bit 5 (32) = the 'fast' flag of `FHCouplings`,
bit 6 (64) = electroweak constraints (`FHConstraints`),
bit 7 (128) = flavour observables (`FHFlavour`).

FHRecordIndex

Description

`FHRecordIndex` converts a parameter name into the corresponding index in a FeynHiggs Record, or returns zero if no such name is known. The record has four fields for every index i: `record(`i`,iVar)`, `record(`i`,iLower)`, `record(`i`,iUpper)`, and `record(`i`,iStep)`. These respectively denote a quantity's current, lower, upper, and step-size value, i.e. define a possible loop over the quantity.

Synopsis – Fortran version

```
integer ind
character*(*) para

subroutine FHRecordIndex(ind, para)
```

Synopsis – Mathematica version

```
ind = FHRecordIndex[para]
```

Arguments

OUT `integer ind`

the index into the FeynHiggs Record, or zero if the record contains no parameter of the given name.

IN `character*(*) para`

the parameter name.

FHClearRecord

Description

`FHClearRecord` sets the fields of a FeynHiggs Record to initial values. Possible pre-existing values are overwritten.

Synopsis – Fortran version

```
#include "FHRecord.h"

RecordDecl(record)

subroutine FHClearRecord(record)
```

Synopsis – Mathematica version

```
record = FHClearRecord[]
```

Arguments

$\boxed{\text{OUT}}$ `double precision record(`i`,`ℓ`)`
the initialized FeynHiggs Record.

FHReadRecord

Description

`FHReadRecord` reads a parameter file in FeynHiggs' native format into a FeynHiggs Record. Possible pre-existing values are overwritten.

Synopsis – Fortran version

```
#include "FHRecord.h"
#include "SLHA.h"

integer error
RecordDecl(record)
double complex slhadata(nslhadata)
character*(*) file

subroutine FHReadRecord(error, record, slhadata, file)
```

Synopsis – Mathematica version

```
record = FHReadRecord[file]
```

Arguments

OUT `integer error`

0 if successful (SLHA file, `slhadata` filled), 2 if successful (FH file, `slhadata` not filled), otherwise the line number in `ReadRecord.F` from which the error message was emitted.

OUT `double precision record(`i`,`ℓ`)`

the FeynHiggs Record.

OUT `double complex slhadata(`i`)`

the SLHA data,

IN `character*(*) file`

the parameter file name.

FHSLHARecord

Description

`FHSLHARecord` initializes a FeynHiggs record from SLHA data. Possible pre-existing values are overwritten.

Synopsis – Fortran version

```
#include "FHRecord.h"
#include "SLHA.h"

integer error
RecordDecl(record)
double complex slhadata(nslhadata)

subroutine FHSLHARecord(error, record, slhadata)
```

Synopsis – Mathematica version

```
record = FHSLHARecord[file]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SLHARecord.F` from which the error message was emitted.

OUT `double precision record(`i`,`ℓ`)`

the FeynHiggs Record.

IN `double complex slhadata(`i`)`

the SLHA data structure.

IN `String file`

the file containing the SLHA data.

FHLoopRecord

Description

`FHLoopRecord` advances the loops implied by a FeynHiggs Record or else signals that all loops have been done. This subroutine is meant to be called in a loop where it updates the record according to its internally defined loops.

To signal the end of the loop, the Fortran version returns a negative error code (positive codes correspond to true errors) and the Mathematica version returns a negative value instead of the record.

Looping over a record would thus look like

```
call FHLoopRecord(error, record)
do while( error .eq. 0 )
  ...
  call FHLoopRecord(error, record)
enddo
if( error .gt. 0 ) stop
```

or in Mathematica

```
While[ Head[record = FHLoopRecord[record]] =!= FHRecord,
  ...
]
```

Synopsis – Fortran version

```
#include "FHRecord.h"

integer error
RecordDecl(record)

subroutine FHLoopRecord(error, record)
```

Synopsis – Mathematica version

```
record = FHLoopRecord[record]
```

Arguments

OUT `integer error`

zero if successful, a negative number if all loops have been done, otherwise the line number in `LoopRecord.F` from which the error message was emitted.

I/O `double precision record(`i`,`ℓ`)`

the (updated) FeynHiggs record.

FHSetRecord

Description

`FHSetRecord` sets the FeynHiggs parameters from a FeynHiggs Record. This subroutine works like a combination of `FHSetPara`, `FHSetCKM`, and `FHSetNMFV` except that the input parameters are taken from the FeynHiggs Record.

Synopsis – Fortran version

```
#include "FHRecord.h"

integer error
RecordDecl(record)

subroutine FHSetRecord(error, record)
```

Synopsis – Mathematica version

```
FHSetRecord[record]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `SetRecord.F` from which the error message was emitted.

I/O `double precision record(`i`,`ℓ`)`

the FeynHiggs Record.

FHRetrieveRecord

Description

`FHRetrieveRecord` fills a FeynHiggs Record from the FeynHiggs parameters currently set.

Synopsis – Fortran version

```
#include "FHRecord.h"

integer error, iX
RecordDecl(record)

subroutine FHRetrieveRecord(error, record, iX)
```

Synopsis – Mathematica version

```
record = FHRetrieveRecord[record, iX]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `RetrieveRecord.F` from which the error message was emitted.

I/O `double precision record(`i,ℓ`)`

the FeynHiggs Record.

IN `integer iX`

the slot into which to fill the data: `iVar`, `iLower`, or `iUpper`.

FHLoadTable

Description

`FHLoadTable` loads a parameter table from a data file into internal storage. The first line of the file contains the column names, separated by whitespace, all following lines are then the corresponding data, similarly whitespace-separated.

Synopsis – Fortran version

```
integer error, unit
character*(*) file

subroutine FHLoadTable(error, file, unit)
```

Synopsis – Mathematica version

```
FHLoadTable[file]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `LoadTable.F` from which the error message was emitted.

IN `character*(*) file`

the name of the data file.

IN `integer unit`

the Fortran unit to read from, if the file is “`-`”.

FHTableRecord

Description

`FHTableRecord` associates a FeynHiggs Record with the internal table, i.e. the two given parameters (e.g. `iTB` and `iMA0`) are used as inputs for interpolating table data in the next `FHLoopRecord` cycle.

Synopsis – Fortran version

```
#include "FHRecord.h"

integer error, i1, i2
RecordDecl(record)

subroutine FHTableRecord(error, record, i1, i2)
```

Synopsis – Mathematica version

```
record = FHTableRecord[record, i1, i2]
```

Arguments

OUT `integer error`

zero if successful, otherwise the line number in `TableRecord.F` from which the error message was emitted.

I/O `double precision record(`i`,`ℓ`)`

the FeynHiggs Record.

IN `integer i1, i2`

the record entries used as inputs for interpolating the internal table, e.g. `i1` = `iTB` and `i2` = `iMA0`.