```
1  /*****************************************************************************
2   * Package: GSM                                                              *
3   * Class  : BosonPart                                                        *
4   *                                                                           *
5   * Description:                                                              *
6   *      Auxiliary Theory for bosonic part of self energies                   *
7   *      one loop core of ZFitter option                                      *
8   *                                                                           *
9   * Sources:                                                                  *
10  *      - Bardin, Degrassi, The Standard Model in the Making, Oxford, 1999   *
11  *      - Nucl. Phys. B197 (1982) 1-44 / first summary of one loop core      *
12  *      - Bardin et al., hep-ph/9709229,                                     *
13  *      - Bardin et al., CPC. 133 (2001) 229, hep-ph/9908433                 *
14  *      - Bardin et al.,ZFitter package dizet6_42.f                          *
15  *                                                                           *
16  * This class also contains code lines ported to C++ from the Fortran package *
17  * ZFITTER                                                                   *
18  *                                                                           *
19  *****************************************************************************/
20  #include <math.h>
21
22  #include "TMath.h"
23
24  #include "Gfitter/GMath.h"
25  #include "Gfitter/GConstants.h"
26  #include "Gfitter/GTheory.h"
27  #include "Gfitter/GTheoryRef.h"
28  #include "Gfitter/GParameterRef.h"
29  #include "Gfitter/GReference.h"
30  #include "Gfitter/GVariable.h"
31  #include "Gfitter/GStore.h"
32
33  #include "GSM/BosonPart.h"
34  #include "GSM/ZMath.h"
35
36  using std::complex;
37
38  using namespace Gfitter;
39
40  GSM::BosonPart::BosonPart()
41     : Gfitter::GAuxTheory(),
42       m_isUpToDate_Update( kFALSE )
43  {
```

Hinweis:

Kommentare mit Hinweisen auf ZFitter sind grün markiert

Übereinstimmungen sind gelb markiert.

Auffällige Stellen bzw. Anmerkungen sind violett markiert

Anhang 3

zum Gutachten DESY ZFitter_GFitter vom 17.03.2014

Lübeck, den 17. März 2014

Dr. Ulrich Obermöller
-Sachverständiger-

Von der Handelskammer Hamburg
Dr. Ulrich Obermöller
Sachverständiger für Systeme und Anwendungen der Informationsverarbeitung, insbesondere Individualsoftware und Software-Qualitätssicherung
öffentlich bestellt und vereidigt

```
44
45      SetTheoryName( GetName() );
46      SetExistDerivative( kFALSE );
47
48      const TString& logMH = gStore()->GetVariable( "GSMFlags::logMH" )->GetStringValue();
49      m_logger << kINFO << "Using logMH: \"" << logMH << "\"" << GEndl;
50
51      if      (logMH == "Yes" ) m_logMH  = kTRUE;
52      else if (logMH == "No" )  m_logMH  = kFALSE;
53      else {
54         m_logger << kFATAL << "unknown value for \"GSMFlags::logMH\": \"" << logMH << "\""
55                  << ". Possible are: \"Yes\" and \"No\"\""
56                  << GEndl;
57      }
58
59      BookParameter( "MZ"      , & p_MZ );
60      BookTheory   ( "GSM::WMass" , & t_MW );
61      BookTheory   ( "GSM::MH" , & t_MH );
62   }
63
64   void GSM::BosonPart::UpdateLocalFlags( GReference& /* ref */ )
65   {
66      m_isUpToDate_Update = kFALSE;
67   }
68
69   void GSM::BosonPart::Update()
70   {
71      if (m_isUpToDate_Update) return;
72
73      // now, it is uptodate (I mean... it will be)
74      m_isUpToDate_Update = kTRUE;
75
76      Double_t MH      = GetMH().GetValue(); //p_MH;
77      if( m_logMH ) MH = TMath::Exp(GetMH().GetValue() ); //p_MH );
78
79      Double_t MZ2 = p_MZ*p_MZ;
80      Double_t MW2 = GMath::IPow( GetMW(), 2 );
81      Double_t MH2 = MH*MH;
82
83      m_R         = MW2/MZ2;
84      m_rw        = MH2/MW2;
85      m_rz        = MH2/MZ2;
86
```

```
87      // Get L and J functions
88      // see dizet6_42.f line 1962-1967
89      m_L0 = ZMath::L( -MZ2, MW2, MW2 )/MZ2;
90      m_L1 = ZMath::L( -MW2, MH2, MW2 )/MW2;
91      m_L2 = ZMath::L( -MW2, MW2, MZ2 )/MW2;
92      m_L3 = ZMath::L( -MZ2, MH2, MZ2 )/MW2;
93      m_L4 = ZMath::L( -MZ2, MW2, MW2 )/MW2;
94      m_J1 = ZMath::J( -MW2, MH2, MW2 )*MH2;
95      m_J3 = ZMath::J( -MZ2, MH2, MZ2 )*MH2/m_R;
96
97      SetUpToDate();
98    }
99
100   // see to all equation for a deeper understanding
101   // The Standard Model in the Making page 192-195
102   // hep-ph/9908433v3 page 152-155 and entire Appendix A
103   // Nucl. Phys. B197 (1982) 1-44
104
105   // eq.(259) of  hep-ph/9709229v1
106   // Z boson self energy at MZ
107   // see dizet6_42.f line 1977
108   complex<Double_t> GSM::BosonPart::GetZbAtMZ()
109   {
110     Update();
111     return ( 35.0/(18.0*m_R) + 35.0/18.0 - 34.0/3.0*m_R - 8.0*m_R*m_R - m_rw/2.0
112            + m_rw*m_rw*m_R/12.0 + m_rw*(-3.0/4.0 + m_rz/4.0 - m_rz*m_rz/24.0)*TMath::Log(m_rz)
113            + 5.0/(6.0*m_R)*TMath::Log(m_R) + (0.5 -m_rz/6.0 + m_rz*m_rz/24.0)*m_L3
114            + (1.0/24.0 + 2.0/3.0*m_R -17/6.0*m_R*m_R - 2.0*m_R*m_R*m_R)*m_L4 );
115   }
116
117   // eq.(260) of  hep-ph/9709229v1
118   // derivative of Z boson self energy at MZ
119   // see dizet6_42.f line 1983
120   complex<Double_t> GSM::BosonPart::GetZbFAtMZ()
121   {
122     Update();
123     return ( -4.0*m_R*m_R + 17.0/3.0*m_R - 23.0/9.0 + 5.0/(18.0*m_R) - m_rw/2
124            + m_rw*m_rz/6.0 - TMath::Log(m_R)/(12.0*m_R)
125            + m_rw*(-3.0/4.0 + 3.0/8.0*m_rz - m_rz*m_rz/12.0)*TMath::Log(m_rz) + 0.5/m_R*TMath::Log(m_rz)
126            + (-m_R*m_R*m_R + 7.0/6.0*m_R*m_R - 17.0/12.0*m_R - 1.0/8.0)*m_L4
127            + (0.5 -5.0/24.0*m_rz + 1.0/12.0*m_rz*m_rz)*m_L3 + 0.5*m_J3 );
128   }
129
```

Match 1

```
1960  *     FILL BOSONIC PARTS
1961  *
1962        XL1=XL(-AMW2,AMH2,AMW2)/AMW2
1963        XJ1=XJ(-AMW2,AMH2,AMW2)*AMH2
1964        XL2=XL(-AMW2,AMW2,AMZ2)/AMW2
1965        XL3=XL(-AMZ2,AMH2,AMZ2)/AMW2
1966        XJ3=XJ(-AMZ2,AMH2,AMZ2)*AMH2/R
1967        XL4=XL(-AMZ2,AMW2,AMW2)/AMW2
1968        R3=R2*R
```

Match 2

```
1977        XZM1=35.D0/18.D0/R+35.D0/18.D0-34.D0/3.D0*R-8.D0*R2-RW/2.D0
1978  *     +RW2*R/12.D0+RW*(-3.D0/4.D0+RZ/4.D0-RZ2/24.D0)*ALRZ
1979  *     +5.D0/6.D0/R*ALR+(0.5D0-RZ/6.D0+RZ2/24.D0)*XL3
1980  *     +(1.D0/24.D0+2.D0/3.D0*R-17.D0/6.D0*R2-2.D0*R3)*XL4
```

Match 3

```
1983        XZFM1=-4.D0*R2+17.D0/3.D0*R-23.D0/9.D0+5.D0/18.D0/R-RW/2.D0
1984  *     +RW*RZ/6.D0-ALR/12.D0/R
1985  *     +RW*(-3.D0/4.D0+3.D0/8.D0*RZ-RZ2/12.D0)*ALRZ+0.5D0/R*ALRZ
1986  *     +(-R*R2+7.D0/6.D0*R2-17.D0/12.D0*R-1.D0/8.D0)*XL4
1987  *     +(0.5D0-5.D0/24.D0*RZ+1.D0/12.D0*RZ2)*XL3+0.5D0*XJ3
```

```
130  // eq.(257) of  hep-ph/9709229v1
131  // W boson self ennergy at 0 GeV
132  // see dizet6_42.f line 1969
133  complex<Double_t> GSM::BosonPart::GetWbAt0()
134  {
135     Update();
136     return ( 5.0/(8.0*m_R) - 17.0/4.0 + 5.0/8.0*m_R*(1.0+m_R) - m_rw/8.0
137              + 3.0/4.0*m_rw/(1-m_rw)*TMath::Log(m_rw) + (3.0/(4.0*m_R) + 9.0/4.0 - 3.0/(1-m_R))*TMath::Log(m_R) );
138  }
139
140
141  // eq.(258) of  hep-ph/9709229v1
142  // W boson self ennergy at MW
143  // see dizet6_42.f line 1972
144  complex<Double_t> GSM::BosonPart::GetWbAtMW()
145  {
146     Update();
147     return ( 1.0/(12.0*m_R*m_R) + 23.0/(12.0*m_R) - 157.0/9.0 - m_rw/2.0 + m_rw*m_rw/12.0
148              - m_rw*(3.0/4.0 - m_rw/4.0 + m_rw*m_rw/24.0)*TMath::Log(m_rw)
149              + (1.0/(24.0*m_R*m_R*m_R) + 7.0/(12.0*m_R*m_R) - 7.0/(2.0*m_R))*TMath::Log(m_R)
150              + (0.5 -m_rw/6.0 + m_rw*m_rw/24.0)*m_L1
151              + (1.0/(24.0*m_R*m_R) + 2.0/(3.0*m_R) - 17.0/6.0 - 2.0*m_R)*m_L2 );
152  }
153
154  // eq.(A.7) of Nucl. Phys. B197 (1982)
155  // fermionic and bosonic part were added together in that eq.
156  // photon Z mixing function
157  // see dizet6_42.f line 1988
158  complex<Double_t> GSM::BosonPart::GetMbPhoZAtMZ()
159  {
160     Update();
161     return ( 2.0/(9.0*m_R) + 35.0/18.0 - 34.0/3.0*m_R - 8.0*m_R*m_R
162              + (1.0/24.0 + 2.0/3.0*m_R - 17.0/6.0*m_R*m_R - 2.0*m_R*m_R*m_R)*m_L4 );
163  }
164
165  // eq.(A.3) of Nucl. Phys. B197 (1982)
166  // fermionic and bosonic part were added together in that eq.
167  // derivative of W boson self energy at MW
168  // see dizet6_42.f line 1990
169  complex<Double_t> GSM::BosonPart::GetWbFAtMW()
170  {
171     Update();
172     return ( m_R - 34/9.0 + 2/m_R + 1/(6.0*m_R*m_R) - m_rw/2.0 + m_rw*m_rw/6.0
```

Match 4

```
1969        W0=5.D0/8.D0/R-17.D0/4.D0+5.D0/8.D0*R*(1.D0+R)-RW/8.D0
1970     *  +3.D0/4.D0*RW/RW1*ALRW+(3.D0/4.D0/R+9.D0/4.D0-3.D0/R1)*ALR
```

Match 5

```
1972        XWM1=1.D0/12.D0/R2+23.D0/12.D0/R-157.D0/9.D0-RW/2.D0+RW2/12.D0
1973     *     -RW*(3.D0/4.D0-RW/4.D0+RW2/24.D0)*ALRW
1974     *     +(1.D0/24.D0/R3+7.D0/12.D0/R2-7.D0/2.D0/R)*ALR
1975     *     +(0.5D0-RW/6.D0+RW2/24.D0)*XL1
1976     *     +(1.D0/24.D0/R2+2.D0/3.D0/R-17.D0/6.D0-2.D0*R)*XL2
```

Match 6

```
1988        XAMM1=2.D0/9.D0/R+35.D0/18.D0-34.D0/3.D0*R-8.D0*R2
1989     *      +(1.D0/24.D0+2.D0/3.D0*R-17.D0/6.D0*R2-2.D0*R*R2)*XL4
```

Match 7

```
1990        XWFM1=R-34.D0/9.D0+2.D0/R+1.D0/6.D0/R2-RW/2.D0+RW**2/6.D0
1991     *      +(3.D0*R+5.D0/2.D0-17.D0/4.D0/R+7.D0/8.D0/R2+1.D0/12.D0/R3)
1992     *      *ALR+(0.5D0-3.D0*RW/4.D0+3.D0*RW2/8.D0-RW**3/12.D0)*ALRW
1993     *      +(-R/2.D0-2.D0+25.D0/24.D0/R+1.D0/12.D0/R2)*XL2
1994     *      +(0.5D0-5.D0*RW/24.D0+RW2/12.D0)*XL1+0.5D0*XJ1
```

```
173             + (3.0*m_R + 5/2.0 - 17/(4.0*m_R) + 7/(8.0*m_R*m_R) + 1/(12.0*m_R*m_R*m_R))*TMath::Log(m_R)
174             + (0.5 - 3.0*m_rw/4.0 + 3.0*m_rw*m_rw/8.0 - GMath::IPow(m_rw,3)/12.0)*TMath::Log(m_rw)
175             + (-m_R/2.0 - 2.0 + 25/(24.0*m_R) + 1/(12.0*m_R*m_R))*m_L2
176             + (0.5 - 5*m_rw/24.0 + m_rw*m_rw/12.0)*m_L1 + 0.5*m_J1 );
177     }
```