2.3 *OASDI Covered Employment and Earnings (MODSOLA)*

Total At-Any-Time Employment (Equations 1–52)

Ages 0 through 15, where s=sex;a=age 0,1,2,3,…15; i=calendar year) (1–10, 21, 22, 27–36, 47, 48)

```
he_m_sy(s,a,i) = (he_m_sy(s,a,histend)   / nsy_a(s,a,histend) &
        + he_m_sy(s,a,histend-1) / nsy_a(s,a,histend-1) &
        + he_m_sy(s,a,histend-2) / nsy_a(s,a,histend-2) ) / 3 &
        *   nsy_a(s,a,i)
```

HI covered workers age groups 10-13 and 14-15, by sex and calendar year (21, 22, 47, 48)

```
he_m_1013(s,i) = sum(he_m_sy(s,10:13,i))
he_m_1415(s,i) = sum(he_m_sy(s,14:15,i)
```

HI covered workers age group 15u, by sex and calendar year (23, 49)

```
he_m_15u(s,i) = sum(he_m_sy(s,0:9,i)) + he_m_1013(s,i) + he_m_1415(s,i)
```

OASDI covered = HI covered, by sex, single year of age, and calendar year (11–22, 37–48)

```
ce_m_sy(s,a,i)=he_m(s,a,i)
```

OASDI covered workers age groups 10-13 and 14-15, by sex and calendar year (24, 25, 50, 51)

```
ce_m_1013(s,i) = sum(ce_m_sy(s,10:13,i))
ce_m_1415(s,i) = sum(ce_m_sy(s,14:15,i)
```

OASDI covered workers age group 15u, by sex and calendar year (26, 52)

```
ce_m_15u(s,i) = sum(ce_m_sy(s,0:9,i)) +ce_m_1013(s,i) + ce_m_1415(s,i)
```

Male Disaggregates Aged 16 and Over

Preliminary

Average Weeks Worked

```
AWWM1617_P = K(1,1,1) * TREND_TE + K(1,1,2) * RM1617 + K(1,1,3);          (53)
AWWM1819_P = K(1,2,1) * TREND_TE + K(1,2,2) * RM1819 + K(1,2,3);          (58)
AWWM2024_P = K(1,3,1) * TREND_TE + K(1,3,2) * RM2024 + K(1,3,3);          (63)
AWWM2529_P = K(1,4,1) * TREND_TE + K(1,4,2) * RM2529 + K(1,4,3);          (68)
AWWM3034_P = K(1,5,1) * TREND_TE + K(1,5,2) * RM3034 + K(1,5,3);          (73)
AWWM3539_P = K(1,6,1) * TREND_TE + K(1,6,2) * RM3539 + K(1,6,3);          (78)
AWWM4044_P = K(1,7,1) * TREND_TE + K(1,7,2) * RM4044 + K(1,7,3);          (83)
AWWM4549_P = K(1,8,1) * TREND_TE + K(1,8,2) * RM4549 + K(1,8,3);          (88)
AWWM5054_P = K(1,9,1) * TREND_TE + K(1,9,2) * RM5054 + K(1,9,3);          (93)
AWWM5559_P = K(1,10,1) * TREND_TE + K(1,10,2) * RM5559 + K(1,10,3);               (98)
AWWM6064_P = K(1,11,1) * TREND_TE + K(1,11,2) * RM6064 + K(1,11,3);               (103)
AWWM6569_P = K(1,12,1) * TREND_TE + K(1,12,2) * RM6569 + K(1,12,3);               (108)
AWWM70O_P  = K(1,13,1) * TREND_TE + K(1,13,2) * RM70O  + K(1,13,3);               (113)

AWWM1617_PL = K(1,1,1) * TREND_TE.1 + K(1,1,2) * RM1617.1 + K(1,1,3);     (55)
AWWM1819_PL = K(1,2,1) * TREND_TE.1 + K(1,2,2) * RM1819.1 + K(1,2,3);     (60)
AWWM2024_PL = K(1,3,1) * TREND_TE.1 + K(1,3,2) * RM2024.1 + K(1,3,3);     (65)
AWWM2529_PL = K(1,4,1) * TREND_TE.1 + K(1,4,2) * RM2529.1 + K(1,4,3);     (70)
AWWM3034_PL = K(1,5,1) * TREND_TE.1 + K(1,5,2) * RM3034.1 + K(1,5,3);     (75)
AWWM3539_PL = K(1,6,1) * TREND_TE.1 + K(1,6,2) * RM3539.1 + K(1,6,3);     (80)
AWWM4044_PL = K(1,7,1) * TREND_TE.1 + K(1,7,2) * RM4044.1 + K(1,7,3);     (85)
AWWM4549_PL = K(1,8,1) * TREND_TE.1 + K(1,8,2) * RM4549.1 + K(1,8,3);     (90)
AWWM5054_PL = K(1,9,1) * TREND_TE.1 + K(1,9,2) * RM5054.1 + K(1,9,3);     (95)
AWWM5559_PL = K(1,10,1) * TREND_TE.1 + K(1,10,2) * RM5559.1 + K(1,10,3);          (100)
AWWM6064_PL = K(1,11,1) * TREND_TE.1 + K(1,11,2) * RM6064.1 + K(1,11,3);          (105)
AWWM6569_PL = K(1,12,1) * TREND_TE.1 + K(1,12,2) * RM6569.1 + K(1,12,3);          (110)
AWWM70O_PL  = K(1,13,1) * TREND_TE.1 + K(1,13,2) * RM70O.1  + K(1,13,3);          (115)
```

Where

K(1,1,1:3) = -0.10135, 0.10821, 38.87103
K(1,2,1:3) = -0.26079, -0.16399, 59.94278
K(1,3,1:3) = -0.10444, -0.20399, 52.93407
K(1,4,1:3) = -0.07062, -0.25689, 54.60044
K(1,5,1:3) = -0.11854, -0.26421, 60.49258
K(1,6,1:3) = -0.01961, -0.12194, 50.95524
K(1,7,1:3) = 0.00877, -0.17920, 48.77217
K(1,8,1:3) = 0.05920, -0.16050, 43.61904
K(1,9,1:3) = 0.06994, -0.20015, 42.40736
K(1,10,1:3) = 0.08423, -0.14701, 39.48020
K(1,11,1:3) = 0.16591, -0.04181, 25.83782
K(1,12,1:3) = 0.09235, -0.05071, 28.20550
K(1,13,1:3) = 0.20038, -0.46791, 16.99531

Work Experience

WEM1617_3_P = EM1617 * 52 / AWWM1617_P; (54)
WEM1819_3_P = EM1819 * 52 / AWWM1819_P; (59)
WEM2024_3_P = EM2024 * 52 / AWWM2024_P; (64)

WEM2529_3_P = EM2529 * 52 / AWWM2529_P; (69)
WEM3034_3_P = EM3034 * 52 / AWWM3034_P; (74)
WEM3539_3_P = EM3539 * 52 / AWWM3539_P; (79)
WEM4044_3_P = EM4044 * 52 / AWWM4044_P; (84)
WEM4549_3_P = EM4549 * 52 / AWWM4549_P; (89)
WEM5054_3_P = EM5054 * 52 / AWWM5054_P; (94)

WEM5559_3_P = EM5559 * 52 / AWWM5559_P; (99)
WEM6064_3_P = EM6064 * 52 / AWWM6064_P; (104)
WEM6569_3_P = EM6569 * 52 / AWWM6569_P; (109)
WEM70O_3_P = EM70O * 52 / AWWM70O_P ; (114)

WEM1617_3_PL = EM1617.1 * 52 / AWWM1617_PL; (56)
WEM1819_3_PL = EM1819.1 * 52 / AWWM1819_PL; (61)
WEM2024_3_PL = EM2024.1 * 52 / AWWM2024_PL; (66)

WEM2529_3_PL = EM2529.1 * 52 / AWWM2529_PL; (71)
WEM3034_3_PL = EM3034.1 * 52 / AWWM3034_PL; (76)
WEM3539_3_PL = EM3539.1 * 52 / AWWM3539_PL; (81)
WEM4044_3_PL = EM4044.1 * 52 / AWWM4044_PL; (86)
WEM4549_3_PL = EM4549.1 * 52 / AWWM4549_PL; (91)
WEM5054_3_PL = EM5054.1 * 52 / AWWM5054_PL; (96)

WEM5559_3_PL = EM5559.1 * 52 / AWWM5559_PL; (101)
WEM6064_3_PL = EM6064.1 * 52 / AWWM6064_PL; (106)
WEM6569_3_PL = EM6569.1 * 52 / AWWM6569_PL; (111)
WEM70O_3_PL = EM70O.1 * 52 / AWWM70O_PL ; (116)

Total Employed

TEM1617_P = ((WEM1617_3_P / WEM1617_3_PL) * (TEM1617.1 - NM1617M.1) + NM1617M) * MULT1_TEM1617 * MULT2_TEM1617; (57)
TEM1819_P = ((WEM1819_3_P / WEM1819_3_PL) * (TEM1819.1 - NM1819M.1) + NM1819M) * MULT1_TEM1819 * MULT2_TEM1819; (62)
TEM2024_P = ((WEM2024_3_P / WEM2024_3_PL) * (TEM2024.1 - NM2024M.1) + NM2024M) * MULT1_TEM2024 * MULT2_TEM2024; (67)

TEM2529_P = ((WEM2529_3_P / WEM2529_3_PL) * (TEM2529.1 - NM2529M.1) + NM2529M) * MULT1_TEM2529 * MULT2_TEM2529; (72)
TEM3034_P = ((WEM3034_3_P / WEM3034_3_PL) * (TEM3034.1 - NM3034M.1) + NM3034M) * MULT1_TEM3034 * MULT2_TEM3034; (77)
TEM3539_P = ((WEM3539_3_P / WEM3539_3_PL) * (TEM3539.1 - NM3539M.1) + NM3539M) * MULT1_TEM3539 * MULT2_TEM3539; (82)
TEM4044_P = ((WEM4044_3_P / WEM4044_3_PL) * (TEM4044.1 - NM4044M.1) + NM4044M) * MULT1_TEM4044 * MULT2_TEM4044; (87)
TEM4549_P = ((WEM4549_3_P / WEM4549_3_PL) * (TEM4549.1 - NM4549M.1) + NM4549M) * MULT1_TEM4549 * MULT2_TEM4549; (92)

TEM5054_P = ((WEM5054_3_P / WEM5054_3_PL) * (TEM5054.1 - NM5054M.1) + NM5054M) * MULT1_TEM5054 * MULT2_TEM5054;
(97)

TEM5559_P = ((WEM5559_3_P / WEM5559_3_PL) * (TEM5559.1 - NM5559M.1) + NM5559M) * MULT1_TEM5559 * MULT2_TEM5559;
(102)

TEM6064_P = ((WEM6064_3_P / WEM6064_3_PL) * (TEM6064.1)) * MULT1_TEM6064 * MULT2_TEM6064; (107)
TEM6569_P = ((WEM6569_3_P / WEM6569_3_PL) * (TEM6569.1)) * MULT1_TEM6569 * MULT2_TEM6569; (112)
TEM70O_P = ((WEM70O_3_P / WEM70O_3_PL) * (TEM70O.1)) * MULT1_TEM70O * MULT2_TEM70O; (117)

WEM16O_3_P = WEM1617_3_P + WEM1819_3_P + WEM2024_3_P + WEM2529_3_P + WEM3034_3_P + WEM3539_3_P +
WEM4044_3_P + WEM4549_3_P
+ WEM5054_3_P + WEM5559_3_P + WEM6064_3_P + WEM6569_3_P + WEM70O_3_P; (190)

AWWM16O_P = EM16O * 52 / WEM16O_3_P; (191)

TEM16O_P = TEM1617_P + TEM1819_P + TEM2024_P + TEM2529_P + TEM3034_P + TEM3539_P + TEM4044_P + TEM4549_P +
TEM5054_P + TEM5559_P + TEM6064_P + TEM6569_P + TEM70O_P; (118)

Final (Pre-TE.ADD) (192–230)

Average Weeks Worked

AWWM1617 = AWWM1617_P;
AWWM1819 = AWWM1819_P;
AWWM2024 = AWWM2024_P;

AWWM2529 = AWWM2529_P;
AWWM3034 = AWWM3034_P;
AWWM3539 = AWWM3539_P;
AWWM4044 = AWWM4044_P;
AWWM4549 = AWWM4549_P;
AWWM5054 = AWWM5054_P;

AWWM5559 = AWWM5559_P;
AWWM6064 = AWWM6064_P;
AWWM6569 = AWWM6569_P;
AWWM70O = AWWM70O_P ;

Work Experience

WEM1617_3 = WEM1617_3_P;
WEM1819_3 = WEM1819_3_P;
WEM2024_3 = WEM2024_3_P;

WEM2529_3 = WEM2529_3_P;
WEM3034_3 = WEM3034_3_P;
WEM3539_3 = WEM3539_3_P;
WEM4044_3 = WEM4044_3_P;
WEM4549_3 = WEM4549_3_P;
WEM5054_3 = WEM5054_3_P;

WEM5559_3 = WEM5559_3_P;
WEM6064_3 = WEM6064_3_P;
WEM6569_3 = WEM6569_3_P;
WEM70O_3 = WEM70O_3_P ;

Total Employed

TEM1617 = TEM1617_P;
TEM1819 = TEM1819_P;
TEM2024 = TEM2024_P;

TEM2529 = TEM2529_P;
TEM3034 = TEM3034_P;
TEM3539 = TEM3539_P;
TEM4044 = TEM4044_P;

TEM4549 = TEM4549_P;
TEM5054 = TEM5054_P;

TEM5559 = TEM5559_P;
TEM6064 = TEM6064_P;
TEM6569 = TEM6569_P;
TEM70O = TEM70O_P ;

WEM16O_3 = WEM16O_3_P;
AWWM16O = AWWM16O_P ;
TEM16O   = TEM16O_P ; (119)
TEM = TEM16O + HE_M_15U(1,YEAR) (120)

Female Disaggregates Aged 16 and Over

Preliminary

Average Weeks Worked

AWWF1617_P = K(2,1,1) * TREND_TE + K(2,1,2) * RF1617 + K(2,1,3); (121)
AWWF1819_P = K(2,2,1) * TREND_TE + K(2,2,2) * RF1819 + K(2,2,3); (126)
AWWF2024_P = K(2,3,1) * TREND_TE + K(2,3,2) * RF2024 + K(2,3,3); (131)

AWWF2529_P = K(2,4,1) * TREND_TE + K(2,4,2) * RF2529 + K(2,4,3); (136)
AWWF3034_P = K(2,5,1) * TREND_TE + K(2,5,2) * RF3034 + K(2,5,3); (141)
AWWF3539_P = K(2,6,1) * TREND_TE + K(2,6,2) * RF3539 + K(2,6,3); (146)
AWWF4044_P = K(2,7,1) * TREND_TE + K(2,7,2) * RF4044 + K(2,7,3); (151)
AWWF4549_P = K(2,8,1) * TREND_TE + K(2,8,2) * RF4549 + K(2,8,3); (156)
AWWF5054_P = K(2,9,1) * TREND_TE + K(2,9,2) * RF5054 + K(2,9,3); (161)

AWWF5559_P = K(2,10,1) * TREND_TE + K(2,10,2) * RF5559 + K(2,10,3); (166)
AWWF6064_P = K(2,11,1) * TREND_TE + K(2,11,2) * RF6064 + K(2,11,3); (171)
AWWF6569_P = K(2,12,1) * TREND_TE + K(2,12,2) * RF6569 + K(2,12,3); (176)
AWWF70O_P = K(2,13,1) * TREND_TE + K(2,13,2) * RF70O + K(2,13,3); (181)

AWWF1617_PL = K(2,1,1) * TREND_TE.1 + K(2,1,2) * RF1617.1 + K(2,1,3); (123)
AWWF1819_PL = K(2,2,1) * TREND_TE.1 + K(2,2,2) * RF1819.1 + K(2,2,3); (128)
AWWF2024_PL = K(2,3,1) * TREND_TE.1 + K(2,3,2) * RF2024.1 + K(2,3,3); (133)
AWWF2529_PL = K(2,4,1) * TREND_TE.1 + K(2,4,2) * RF2529.1 + K(2,4,3); (138)
AWWF3034_PL = K(2,5,1) * TREND_TE.1 + K(2,5,2) * RF3034.1 + K(2,5,3); (143)
AWWF3539_PL = K(2,6,1) * TREND_TE.1 + K(2,6,2) * RF3539.1 + K(2,6,3); (148)
AWWF4044_PL = K(2,7,1) * TREND_TE.1 + K(2,7,2) * RF4044.1 + K(2,7,3); (153)
AWWF4549_PL = K(2,8,1) * TREND_TE.1 + K(2,8,2) * RF4549.1 + K(2,8,3); (158)
AWWF5054_PL = K(2,9,1) * TREND_TE.1 + K(2,9,2) * RF5054.1 + K(2,9,3); (163)

AWWF5559_PL = K(2,10,1) * TREND_TE.1 + K(2,10,2) * RF5559.1 + K(2,10,3); (168)
AWWF6064_PL = K(2,11,1) * TREND_TE.1 + K(2,11,2) * RF6064.1 + K(2,11,3); (173)
AWWF6569_PL = K(2,12,1) * TREND_TE.1 + K(2,12,2) * RF6569.1 + K(2,12,3); (178)
AWWF70O_PL = K(2,13,1) * TREND_TE.1 + K(2,13,2) * RF70O.1 + K(2,13,3); (183)

Where

K(2,1,1:3) = -0.12302, 0.12897, 41.41885
K(1,2,1:3) = -0.29702, -0.13940, 61.87470
K(2,3,1:3) = -0.10879, -0.21126, 50.64397
K(2,4,1:3) = -0.10150, -0.22016, 53.73949
K(2,5,1:3) = -0.17025, -0.12264, 61.65237
K(2,6,1:3) = -0.14313, -0.06333, 60.50550
K(2,7,1:3) = -0.12561, -0.09314, 60.30694
K(2,8,1:3) = -0.07918, -0.11812, 56.25816
K(2,9,1:3) = -0.05666, -0.18944, 54.11270
K(2,10,1:3) = -0.08189, 0.03822, 55.14774
K(2,11,1:3) = 0.02421, 0.03288, 39.72131
K(2,12,1:3) = -0.02386, -0.24361, 40.86353
K(2,13,1:3) = 0.22062, -0.09578, 14.58432

Work Experience

WEF1617_3_P = EF1617 * 52 / AWWF1617_P; (122)
WEF1819_3_P = EF1819 * 52 / AWWF1819_P; (127)
WEF2024_3_P = EF2024 * 52 / AWWF2024_P; (132)

WEF2529_3_P = EF2529 * 52 / AWWF2529_P; (137)
WEF3034_3_P = EF3034 * 52 / AWWF3034_P; (142)
WEF3539_3_P = EF3539 * 52 / AWWF3539_P; (147)
WEF4044_3_P = EF4044 * 52 / AWWF4044_P; (152)
WEF4549_3_P = EF4549 * 52 / AWWF4549_P; (157)
WEF5054_3_P = EF5054 * 52 / AWWF5054_P; (162)

WEF5559_3_P = EF5559 * 52 / AWWF5559_P; (167)
WEF6064_3_P = EF6064 * 52 / AWWF6064_P; (172)
WEF6569_3_P = EF6569 * 52 / AWWF6569_P; (177)
WEF70O_3_P = EF70O * 52 / AWWF70O_P ; (182)

WEF1617_3_PL = EF1617.1 * 52 / AWWF1617_PL; (124)
WEF1819_3_PL = EF1819.1 * 52 / AWWF1819_PL; (129)
WEF2024_3_PL = EF2024.1 * 52 / AWWF2024_PL; (134)

WEF2529_3_PL = EF2529.1 * 52 / AWWF2529_PL; (139)
WEF3034_3_PL = EF3034.1 * 52 / AWWF3034_PL; (144)
WEF3539_3_PL = EF3539.1 * 52 / AWWF3539_PL; (149)
WEF4044_3_PL = EF4044.1 * 52 / AWWF4044_PL; (154)
WEF4549_3_PL = EF4549.1 * 52 / AWWF4549_PL; (159)
WEF5054_3_PL = EF5054.1 * 52 / AWWF5054_PL; (164)

WEF5559_3_PL = EF5559.1 * 52 / AWWF5559_PL; (169)
WEF6064_3_PL = EF6064.1 * 52 / AWWF6064_PL; (174)
WEF6569_3_PL = EF6569.1 * 52 / AWWF6569_PL; (179)
WEF70O_3_PL = EF70O.1 * 52 / AWWF70O_PL ; (184)

Total Employed

TEF1617_P = ((WEF1617_3_P / WEF1617_3_PL) * (TEF1617.1 - NF1617M.1) + NF1617M) * MULT1_TEF1617 * MULT2_TEF1617; (125)
TEF1819_P = ((WEF1819_3_P / WEF1819_3_PL) * (TEF1819.1 - NF1819M.1) + NF1819M) * MULT1_TEF1819 * MULT2_TEF1819; (130)
TEF2024_P = ((WEF2024_3_P / WEF2024_3_PL) * (TEF2024.1 - NF2024M.1) + NF2024M) * MULT1_TEF2024 * MULT2_TEF2024; (135)

TEF2529_P = ((WEF2529_3_P / WEF2529_3_PL) * (TEF2529.1 - NF2529M.1) + NF2529M) * MULT1_TEF2529 * MULT2_TEF2529; (140)
TEF3034_P = ((WEF3034_3_P / WEF3034_3_PL) * (TEF3034.1 - NF3034M.1) + NF3034M) * MULT1_TEF3034 * MULT2_TEF3034; (145)
TEF3539_P = ((WEF3539_3_P / WEF3539_3_PL) * (TEF3539.1 - NF3539M.1) + NF3539M) * MULT1_TEF3539 * MULT2_TEF3539; (150)
TEF4044_P = ((WEF4044_3_P / WEF4044_3_PL) * (TEF4044.1 - NF4044M.1) + NF4044M) * MULT1_TEF4044 * MULT2_TEF4044; (155)
TEF4549_P = ((WEF4549_3_P / WEF4549_3_PL) * (TEF4549.1 - NF4549M.1) + NF4549M) * MULT1_TEF4549 * MULT2_TEF4549; (160)
TEF5054_P = ((WEF5054_3_P / WEF5054_3_PL) * (TEF5054.1 - NF5054M.1) + NF5054M) * MULT1_TEF5054 * MULT2_TEF5054; (165)

TEF5559_P = ((WEF5559_3_P / WEF5559_3_PL) * (TEF5559.1 - NF5559M.1) + NF5559M) * MULT1_TEF5559 * MULT2_TEF5559; (170)
TEF6064_P = ((WEF6064_3_P / WEF6064_3_PL) * (TEF6064.1)) * MULT1_TEF6064 * MULT2_TEF6064; (175)
TEF6569_P = ((WEF6569_3_P / WEF6569_3_PL) * (TEF6569.1)) * MULT1_TEF6569 * MULT2_TEF6569; (180)
TEF70O_P = ((WEF70O_3_P / WEF70O_3_PL) * (TEF70O.1)) * MULT1_TEF70O * MULT2_TEF70O; (185)

WEF16O_3_P = WEF1617_3_P + WEF1819_3_P + WEF2024_3_P + WEF2529_3_P + WEF3034_3_P + WEF3539_3_P + WEF4044_3_P + WEF4549_3_P
+ WEF5054_3_P + WEF5559_3_P + WEF6064_3_P + WEF6569_3_P + WEF70O_3_P; (233)

AWWF16O_P = EF16O * 52 / WEF16O_3_P; (234)

TEF16O_P = TEF1617_P + TEF1819_P + TEF2024_P + TEF2529_P + TEF3034_P + TEF3539_P + TEF4044_P + TEF4549_P + TEF5054_P
+ TEF5559_P + TEF6064_P + TEF6569_P + TEF70O_P; (186)

Final (Pre-TE.ADD) (235–273)

Average Weeks Worked

AWWF1617 = AWWF1617_P;
AWWF1819 = AWWF1819_P;
AWWF2024 = AWWF2024_P;

AWWF2529 = AWWF2529_P;
AWWF3034 = AWWF3034_P;
AWWF3539 = AWWF3539_P;
AWWF4044 = AWWF4044_P;
AWWF4549 = AWWF4549_P;
AWWF5054 = AWWF5054_P;

AWWF5559 = AWWF5559_P;
AWWF6064 = AWWF6064_P;
AWWF6569 = AWWF6569_P;
AWWF70O = AWWF70O_P ;

Work Experience

WEF1617_3 = WEF1617_3_P;
WEF1819_3 = WEF1819_3_P;
WEF2024_3 = WEF2024_3_P;

WEF2529_3 = WEF2529_3_P;
WEF3034_3 = WEF3034_3_P;
WEF3539_3 = WEF3539_3_P;
WEF4044_3 = WEF4044_3_P;
WEF4549_3 = WEF4549_3_P;
WEF5054_3 = WEF5054_3_P;

WEF5559_3 = WEF5559_3_P;
WEF6064_3 = WEF6064_3_P;
WEF6569_3 = WEF6569_3_P;
WEF70O_3 = WEF70O_3_P ;

Total Employed

TEF1617 = TEF1617_P;
TEF1819 = TEF1819_P;
TEF2024 = TEF2024_P;

TEF2529 = TEF2529_P;
TEF3034 = TEF3034_P;
TEF3539 = TEF3539_P;
TEF4044 = TEF4044_P;
TEF4549 = TEF4549_P;
TEF5054 = TEF5054_P;

TEF5559 = TEF5559_P;
TEF6064 = TEF6064_P;
TEF6569 = TEF6569_P;
TEF70O = TEF70O_P ;

WEF16O_3 = WEF16O_3_P; (274)
AWWF16O = AWWF16O_P ; (275)
TEF16O = TEF16O_P ; (187)
TEF = TEF16O + HE_M_15U(2,YEAR) (188)

Combined, Age 16 and Over

WE16O_3_P = WEM16O_3_P + WEF16O_3_P; (276)
AWW16O_P = E16O * 52 / WE16O_3_P; (277)

WE16O_3 = WE16O_3_P ; (278)
AWW16O = AWW16O_P ; (279)

TE = TEM + TEF (189)

Self-Employed Only
SEOCMB = WSW_HIO_OTH_SE + TEFC_N_N_SE + TESL_N_N_HI_SE (308)
SEOCMBL1 = WSW_HIO_OTH_SE + TEFC_N_N_SE + TESL_N_N_HI_SE (309)
SEO = (SEO.1 * (EAS + ENAS) / (EAS.1 + ENAS.1) + (SEOCMB - SEOCMBL1)) * MULTSEO (310)

Combination Workers
CMB_TOT = ((-0.01468 + 0.06227 * RTP.1 -0.0008) * WSWA - SEOCMB) * MULTCMB (322)
CSW_TOT = SEO + CMB_TOT
AW_CMBTOT = 1.4953 * ACWA (337)
W_CMBTOT = AW_CMBTOT * CMB_TOT (338)
CMB_WRELMAX = TAXMAX/AW_CMBTOT (344)

CMB Wage Andover Curve (345–349)
CMB_WAO1 = IF (CMB_WRELMAX < 0.0543009)
THEN 1 - 0.722659 * CMB_WRELMAX$^{0.65}$ - 0.461913 * CMB_WRELMAX$^{0.8}$
ELSE IF (CMB_WRELMAX<0.1086018)
THEN -1.02884 * CMB_WRELMAX$^{0.6}$ + 0.324761 * CMB_WRELMAX$^{1.6}$ + 1.02015
ELSE IF (CMB_WRELMAX < 0.1629027)
THEN -0.906607 * CMB_WRELMAX$^{0.7}$ + 0.947662
ELSE IF (CMB_WRELMAX < 0.2172037)
THEN -0.813951 * CMB_WRELMAX$^{0.55}$ + 0.991722
ELSE IF (CMB_WRELMAX < 0.3258055)
THEN -0.755135 * CMB_WRELMAX$^{0.55}$ + 0.964593
ELSE 0

CMB_WAO2 = IF (CMB_WRELMAX < 0.5430091)
THEN -0.649755 * CMB_WRELMAX$^{0.6}$ + 0.886467
ELSE IF (CMB_WRELMAX < 0.7059119)
THEN -0.573205 * CMB_WRELMAX$^{0.7}$ + 0.810122
ELSE IF (CMB_WRELMAX < 0.9231155)
THEN - 5.22264 * CMB_WRELMAX$^{0.06}$ + 5.47514
ELSE IF (CMB_WRELMAX < 1.0860183)
THEN - 2.02619 * CMB_WRELMAX$^{0.15}$ + 2.27963
ELSE IF (CMB_WRELMAX < 1.5204256)
THEN 0.605192 * EXP (-0.2 * CMB_WRELMAX) - 0.827158 * EXP (-0.8 * CMB_WRELMAX)
+ 1.52918 * EXP (- 1.5 * CMB_WRELMAX) - 0.212269
ELSE 0

CMB_WAO3 = IF (CMB_WRELMAX < 1.8462311)
THEN 0.19139 * EXP (-0.6 * CMB_WRELMAX) + 0.764408 * EXP (- 1.8 * CMB_WRELMAX) + 0.0194903
ELSE IF (CMB_WRELMAX < 2.3077888)
THEN 0.12964 * EXP (-0.5 * CMB_WRELMAX) + 0.644861 * EXP (- 1.5 * CMB_WRELMAX) + 0.0183343
ELSE IF (CMB_WRELMAX < 2.9865502)
THEN 0.361318 * EXP (-0.8 * CMB_WRELMAX) + 0.0219491
ELSE IF (CMB_WRELMAX < 4.3440731)
THEN 0.193202 * EXP (-0.45 * CMB_WRELMAX) + 0.00425171
ELSE IF (CMB_WRELMAX<5.4300913)
THEN 0.0560412 * EXP (-0.25 * CMB_WRELMAX) +0.311286 * EXP (-0.8 * CMB_WRELMAX) +
0.00297316
ELSE 0

CMB_WAO4 = IF (CMB_WRELMAX < 13.5752283)
THEN 0.0995677 * EXP (-0.32 * CMB_WRELMAX) + 0.00355234
ELSE IF (CMB_WRELMAX < 21.7203653)
THEN 0.041159 * EXP (-0.19 * CMB_WRELMAX) + 0.00156765
ELSE IF (CMB_WRELMAX < 678.7614168)

THEN 0.265022 * CMB_WRELMAX $^{(-1.555)}$
ELSE 0

CMB_WAO = IF (CMB_WRELMAX < 0.3258055)
THEN CMB_WAO1
ELSE IF (CMB_WRELMAX < 1.5204256)
THEN CMB_WAO2
ELSE IF (CMB_WRELMAX < 5.4300913)
THEN CMB_WAO3
ELSE CMB_WAO4

CMB	=	(1 - (CMB_WAO - 0.019)) * CMB_TOT	(350)
CSW	=	SEO + CMB	(353)
SEOCMB	=	WSW_HIO_OTH_SE + TEFC_N_N_SE + TESL_N_N_HI_SE	(308)
SEO_HI	=	SEO - SEOCMB	(311)
CMB_HI	=	CMB_TOT + SEOCMB	(351)
CSW_TOT	=	SEO + CMB_TOT	(352)
CSW_HI	=	SEO_HI + CMB_HI	(354)

NIPA Wages

Private Residual Sector

WSDPB	=	WSDP – WSPH – WSPF – WSPRRB – TIPS_SR	(327)
TIPS_SR	=	(0.000508328 * RTP - 0.000481700) * GDP * 1.26393 + TIPS_SR_ADD	(326)

OASDI Wages

Covered Employment and Wages – Federal Civilian Government

TEFC	=	(TEFC.1 / EGGEFC.1) * EGGEFC	(362)
TEFC_N	=	IF (CSRS.1 > 0) THEN TEFC_N.1/CSRS.1 * CSRS - TEFC_N_SW ELSE 0	
TEFC_N_N	=	HE_WOF_M	(319)
TEFC_N_N_SE	=	HE_WOSF_M	(302)
TEFC_N_O	=	(TEFC_N - TEFC_N_N)	(365)
TEFC_O	=	(TEFC - TEFC_N)	(364)
WEFC	=	(WEFC.1 / WSGGEFC.1) * WSGGEFC	(331)
WEFC_O	=	(WEFC -WEFC_N) * ADJ_FSA_FC	(332)

Covered Employment and Wages - State and Local Govt.

TESL	=	(TESL.1/EGGESLMAX.1) * EGGESLMAX	(280)
TESL_O	=	(TESL_O.1/TESL.1) * TESL	(281)
TESL_N	=	(TESL-TESL_O)	(282)
TESL_N_O	=	(TESL_N_O_HI + TESL_N_O_NHI)	(293)
TESL_N_O_HI	=	(TESL_N - TESL_N_O_NHI - TESL_N_N_NHI) * CER_MQGE_O	(292)
TESL_N_O_NHI	=	(TESL_N_O_NHI_S + TESL_N_O_NHI_E + TESL_N_O_NHI_NS)	(288)
TESL_N_S	=	TESL_N_S.1 * (NF1819 + NF2024 + NM1819 + NM2024) / (NF1819.1 + NF2024.1 + NM1819.1 + NM2024.1)	(355)
TESL_N_E	=	TESL_N_E.1 * (TESL / TESL.1)	(285)
TESL_N_O_NHI_S	=	TESL_N_S * (TESL_N_O_NHI_S.1/TESL_N_S.1)	(284)
TESL_N_O_NHI_E	=	TESL_N_E * 0.6	(286)
TESL_N_O_NHI_NS	=	TESL_N_O_NHI_NS.1 * ESR_NS	(287)
TESL_N_N	=	(TESL_N - TESL_N_O)	(294)
TESL_N_N_HI	=	(TESL_N_N - TESL_N_N_NHI)	(295)
TESL_N_N_HI_SE	=	(TESL_N_N_HI_SE.1 / TESL_N_N_HI.1) * TESL_N_N_HI	(303)
TESL_N_N_NHI	=	(TESL_N_N_NHI_S + TESL_N_N_NHI_E + TESL_N_N_NHI_NS)	(291)
TESL_N_N_NHI_S	=	(TESL_N_S - TESL_N_O_NHI_S)	(283)
TESL_N_N_NHI_E	=	(TESL_N_E - TESL_N_O_NHI_E)	(289)
TESL_N_N_NHI_NS	=	TESL_N_N_NHI_NS.1 * ESR_NS	(290)
WESL	=	(WESL.1/WSGGESL.1) * WSGGESL	(297)
WESL_O	=	(WESL_O.1/WSGGESL.1) * WSGGESL	(298)
WESL_N	=	(WESL - WESL_O)	(299)
WESL_N_HI	=	(WESL_N - WESL_N_NHI)	(361)
WESL_N_NHI	=	(WESL_N_NHI_S + WESL_N_NHI_E + WESL_N_NHI_NS)	(360)
WESL_N_NHI_S	=	WESL_N_NHI_S.1 * (TESL_N_S/TESL_N_S.1) * (AWSGGESL/AWSGGESL.1)	(356)
WESL_N_NHI_E	=	WESL_N_NHI_E.1 * (TESL_N_E/TESL_N_E.1) * (AWSGGESL/AWSGGESL.1)	(357)
RAWR_NS	=	IF (AWR_NS = 0) THEN 0 ELSE AWR_NS/AWR_NS.1	(358)

WESL_N_NHI_NS = IF (ESR_NS = 0) THEN 0 ELSE WESL_N_NHI_NS.1 * (TESL_N_O_NHI_NS + TESL_N_N_NHI_NS) / (TESL_N_O_NHI_NS.1 + TESL_N_N_NHI_NS.1) * (AWSGGESL/AWSGGESL.1) * RAWR_NS (359)

WSW_HIO_OTH = 0 (296)
WSCA_HIO_OTH = WSW_HIO_OTH*(WESL_N/TESL_N) (300)
WSW_HIO_OTH_SE = 0 (301)

HE_WOL_M = TESL_N_N_HI (304)
HE_WOR_M = 0 (305)
HE_WOSL_M = (HE_WOSL_M.1 / HE_WOL_M.1) * HE_WOL_M (306)
HE_WOSR_M = 0 (307)

Underground Economy and the Earnings Suspense File

TE_U = TEO_UND (312)
TEL_SO = TEL_SO.1 * (TE – TEO_UND – TEO_ESF) / (TE.1 – TEO_UND.1 – TEO_ESF.1) (313)
TE_S = TEL_SO + TEO_ESF (314)
TE_SFO_LRP = TEL_SO (389)
WE_SFO_LRP = TE_SFO_LRP * ACWA (390)
TE_SFM_LRP = TE_SFO_LRP (391)
WE_SFM_LRP = TE_SFM_LRP * ACWA * 0.5 (392)
WE_SF_LRP = WE_SFO_LRP + WE_SFM_LRP (393)
WE_SF = WS_EO_ESF + WE_SF_LRP (394)
TE_SF_LRP = TE_SFO_LRP + TE_SFM_LRP (403)
TE_SF_TEO = TEO_NOL_S + TEO_NOI_S (405)

MEF

TE_M = TE – TE_U – TE_S (315)
TE_MN = TE_RRO_M + TE_SLOO_M + TE_SLOS_M + TE_SLOE_M + TE_PS_M + TE_PH_M + TEO_ASF1 + TEO_ASJ1 + TEO_AWJ + TEO_AWH (316)
HE_M = TE_M – TE_MN (317)
HEW_M = HE_M – SEO_HI (396)
WSW_MEF = HEW_M + TE_MN (397)
WS_MEF = WSCAHI – WE_SF + WSPRRB + WESL_N_NHI + (WESL_N_NHI_S /TESL_N_S) * TE_PS_M + (0.5 * 1.8 / 44.32167) * (AIW.1 / 1000 * PROD / PROD.1 * AHRS / AHRS.1 * PGDP / PGDP.1) * TE_PH_M + WS_EO_ASF1 + WS_EO_ASJ1 + WS_EO_AWJ + WS_EO_AWH (395)
CE_M = HE_M – (HE_WOF_M + HE_WOL_M + HE_WOR_M) – (HE_WOSF_M + HE_WOSL_M + HE_WOSR_M) (406)
CEW_M = CE_M – SEO (407)
CESO_M = SEO (408)
HESO_M = SEO_HI (409)

Self-Employed Earnings Sector

Covered SENE (366–374)
CSE_TOT = (YF +YNF) / (YF.1 + YNF.1) * CSE_TOT.1
CSE_CMB_N = (CSE_TOT / (CMB_TOT + SEO)) / (CSE_TOT.1 / (CMB_TOT.1 + SEO.1)) * (CSE_CMB_N.1 / (CMB_TOT.1 - CMB.1)) * (CMB_TOT - CMB)
CSE = CSE_TOT - CSE_CMB_N
ACSE_SEO = (CSE_TOT / (SEO + 0.416488 * CMB_TOT))
ACSE_CMB_TOT = 0.416488 * ACSE_SEO
CSE_SEO = ACSE_SEO * SEO
CSE_CMB_TOT = ACSE_CMB_TOT * CMB_TOT
CSE_CMB = CSE_CMB_TOT - CSE_CMB_N
ACSE_CMB = CSE_CMB/CMB

Present Law OASDI and HI Covered Wages and Earnings
WSGMLC = CML * WSGFM (334)
WSGFCA = WEFC_O (333)
CFCA = WSGFCA/WSGGEFC (375)
CSLHI = (WESL_O+WESL_N_HI)/WSGGESL (376)

WSGSLCA	=	WESL_O	(330)
WSPH_O	=	CPH * WSPH	(323)
WSPF_O	=	WSPF_O.1 * WSPF/WSPF.1	(324)
CPF	=	WSPF_O/WSPF	(377)
WSPRR_O	=	CPRR * WSPRRB	(325)
WSPC	=	WSPH_O + WSPF_O + WSPRR_O + TIPS_SR + WSPB_O	(329)
CP	=	WSPC/WSDP	(378)
WSCA	=	(WSPC + WSGSLCA + WSGFCA + WSGMLC)	(335)
COVERNA	=	(WSCA + CSE)	(379)
ACWA	=	WSCA/WSWA	(336)
ASE	=	CSE/CSW	(380)
ASEHI	=	CSE_TOT/CSW_HI	(381)
ACEA	=	COVERNA/TCEA	(382)
ACSLW	=	WESL_O/TESL_O * MULTACSLW	(383)
ACMW	=	ACMW.1 * AWSGFM/AWSGFM.1 * CML/CML.1	(384)
ACFCW	=	WEFC_O/TEFC_O	(385)
ACFMW	=	ACFMW.1 * (AIW.1/AIW.3)$^{0.5}$	(386)
TEPH_N	=	ENAWPH * (1 -CPH)	
TEP_N_N_S	=	TEP_N_N_S.1 * (NF1819 + NF2024 + NM1819 + NM2024) / (NF1819.1 + NF2024.1 + NM1819.1 + NM2024.1)	
WSWA	=	(TCEA-SEO)	(321)

Present Law HI Covered Wages and Earnings

WSCAHI_ADD	=	WSCA * WSCAHI_ADD.1/WSCA.1	
TCEAHI	=	HE_M + TE_S	(318)
TCEA	=	TCEAHI – ((TESL_N_N_HI + TEFC_N_N + WSW_HIO_OTH) – (TESL_N_N_HI_SE + TEFC_N_N_SE + WSW_HIO_OTH_SE))	(320)
WSWAHI	=	TCEAHI - SEO_HI	
WSCAHI	=	WSCA + WEFC_N + WESL_N_HI + WSCA_HIO_OTH	(388)
ACWAHI	=	WSCAHI/WSWAHI	
COVERNHI	=	WSCAHI + CSE_TOT	
ACEAHI	=	COVERNHI/TCEAHI	

Complete Coverage concepts

WSWC	=	(WSWAHI + TEPH_N + EPRRB + TEP_N_N_S + TEPO_N + TESL_N_N_NHI) + LOST_MF
ACWC	=	WSD/WSWC
AIW	=	IF AIW_GR_YR = 0 THEN AIW.1 *ACWC/ACWC.1 * MULTAIW ELSE AIW.1 * (1 +AIW_GR/100)

Taxable Maximums

RAIW	=	AIW.2/AIWBASE	(339)
TAXMAXB1	=	RAIW * TMAXBASE * 1000/300	(340)
TAXMAXB2	=	IF TAXMAXB1 - ROUND (TAXMAXB1) >= 0.5 THEN ROUND (TAXMAXB1) +1 ELSE ROUND (TAXMAXB1)	(341)
TAXMAXB3	=	IF TAXMAXB2 < TAXMAX.1 THEN TAXMAX.1 * 1000/300 ELSE TAXMAXB2	(342)
TAXMAX	=	IF BENINC.1 <= 0.001 THEN TAXMAX.1 ELSE 300 * TAXMAXB3/1000	(343)

Deemed Military Wage Credits

EDMILAF	=	EDMIL * 1.1
EDMILT	=	(2.00303 - 50.7517/YEAR) * EDMILAF
EDMILR	=	EDMILT – EDMILAF
MWC_ED_O	=	1.2 * EDMILAF * 0.997
MWC_ED_HI	=	1.2 * EDMILAF
AMWC_GO2	=	MIN (1.2, AWSGFM * (2/52) * (1/3))
MWC_EDR_O	=	AMWC_GO2 * EDMILR * (1 - 0.017)
MWC_EDR_HI=		MWC_EDR_O + ((1.2 +AMWC_GO2) * 0.5) * EDMILR * 0.017
MWC_O	=	MWC_ED_O + MWC_EDR_O
MWC_HI	=	MWC_ED_HI+ MWC_EDR_HI