```
In[72]:= (* ref: New precession expression, valid for long time intervals,
J Vondrak, DOI: 10.1051/0004-6361/201117274 *)
(* small differences due to approximation *)
ClearAll["Global`*"]

In[44]:= (* long term precession of the ecliptic *)
pecl[refDate_] :=
 Module[
  {daysInCentury, as2r, eps0, vPcos, vPsin, vQcos, vQsin, vPconst, vers, J2000, T,
   periodicP, sumP, periodicQ, sumQ, asP, asQ, z, s, c, epj},
  daysInCentury = 36525;
  (* Arcseconds to Radians *)
  as2r = 4.848136811095359935899141 × 10^-6;
  (* Obliquity at J2000 (radians) *)
  eps0 = 84381.406 as2r;
  J2000 = JulianDate[DateObject[{2000, 1, 1, 12, 0, 0}, TimeZone → "GMT"]];
  epj = JulianDate[refDate];
  T = (epj - J2000) / daysInCentury;
  (* Periodic terms *)
  vPcos = {-5486.751211, -17.127623, -617.517403,
    413.44294, 78.614193, -180.732815, -87.676083, 46.140315};
  vPsin = {667.6673, -2354.886252, -428.152441,
    376.202861, 184.778874, 335.321713, -185.138669, -120.97283};
  vQcos = {-684.66156, 2446.28388, 399.671049,
    -356.652376, -186.387003, -316.80007, 198.296071, 101.135679};
  vQsin = {-5523.863691, -549.74745, -310.998056,
    421.535876, -36.776172, -145.278396, -34.74445, 22.885731};
  vPconst = {708.15, 2309, 1620, 492.2, 1183, 622, 882, 547};
  vers = {1, 1, 1, 1, 1, 1, 1, 1};
  periodicP = vPcos Cos[2 π T / vPconst] + vPsin Sin[2 π T / vPconst];
  sumP = periodicP.vers; (* dot product *)
  periodicQ = vQcos Cos[2 π T / vPconst] + vQsin Sin[2 π T / vPconst];
  sumQ = periodicQ.vers; (* dot product *)
  asP = (5851.607687 - .1189 T - .00028913 T^2 + .000000101 T^3 + sumP) * as2r;
  asQ = (-1600.8863 + 1.1689818 T - .0000002 T^2 - .000000437 T^3 + sumQ) * as2r;
  (* ecliptic pole vector *)
  z = Sqrt[Max[1 - asP^2 - asQ^2, 0]];
  s = Sin[eps0];
  c = Cos[eps0];
  {asP, -asQ * c - z * s, -asQ * s + z * c}
 ]
```

In[45]:=
```
(* long precession of the equator *)
pequ[refDate_] :=
 Module[
  {daysInCentury, as2r, vXcos, vXsin, vYcos, vYsin, vPconst, vers, J2000, T,
   periodicX, sumX, periodicY, sumY, asX, asY, w, epj},
  daysInCentury = 36525;
  (* Arcseconds to Radians *)
  as2r = 4.848136811095359935899141 × 10^-6;
  J2000 = JulianDate[DateObject[{2000, 1, 1, 12, 0, 0}, TimeZone → "GMT"]];
  epj = JulianDate[refDate];
  T = (epj - J2000) / daysInCentury;
  (* Periodic terms *)
  vXcos = {-819.940624, -8444.676815,
    2600.009459, 2755.17563, -167.659835, 871.855056, 44.769698,
          -512.313065, -819.415595,
    -538.071099, -189.793622, -402.922932, 179.516345, -9.814756};
  vXsin = {81491.287984, 787.163481,
    1251.296102, -1257.950837, -2966.79973, 639.744522, 131.600209,
         -445.040117, 584.522874,
    -89.756563, 524.42963, -13.549067, -210.157124, -44.919798};
  vYcos = {75004.344875, 624.033993,
    1251.136893, -1102.212834, -2660.66498, 699.291817, 153.16722,
         -950.865637, 499.754645,
    -145.18821, 558.116553, -23.923029, -165.405086, 9.344131};
  vYsin = {1558.515853, 7774.939698, -2219.534038,
    -2523.969396, 247.850422, -846.485643, -1393.124055,
    368.526116, 749.045012, 444.704518,
    235.934465, 374.049623, -171.33018, -22.899655};
  vPconst = {256.75, 708.15, 274.2, 241.45, 2309, 492.2, 396.1,
           288.9, 231.1, 1610, 620, 157.87, 220.3, 1200};
  vers = {1, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1};
  periodicX = vXcos Cos[2 π T / vPconst] + vXsin Sin[2 π T / vPconst];
  sumX = periodicX.vers; (* dot product *)
  periodicY = vYcos Cos[2 π T / vPconst] + vYsin Sin[2 π T / vPconst];
  sumY = periodicY.vers; (* dot product *)
  (* X and Y direction cosine *)
  asX = (5453.282155 + .4252841 T - .00037173 T^2 - .000000152 T^3 + sumX) * as2r;
  asY = (-73750.93035 - .7675452 T - .00018725 T^2 - .000000231 T^3 + sumY) * as2r;
  (* equator pole vector *)
  w = Sqrt[Max[1 - asX^2 - asY^2, 0]];
  {asX, asY, w}
 ]
```

In[46]:=
```
(* long term precession matrix *)
pmat[refDate_] :=
 Module[
  {vPecl, vPequ, eqx, vCross},
  vPecl = pecl[refDate];
  vPequ = pequ[refDate];
  vCross = Cross[vPequ, vPecl];
  eqx = Normalize[vCross];
  vCross = Cross[vPequ, eqx];
  {eqx, vCross, vPequ}
 ]
```

In[47]:=
```
(* equatorial to rectangular with proper motion, if any *)
EquRect[equ_, propMotion_] :=
 Module[
  {α, δ, as2r, pRA, pDecl, pYrs, x, y, z},
  mas2r = 4.84813681109535993589914 × 10^-9; (* milliarcsec (mas) to radians *)
  α = FromDMS[equ[[1]]] * 15 * Degree; (* hours to degrees (1^h=15°) to radians *)
  δ = FromDMS[equ[[2]]] * Degree; (* radians *)
  (* add proper motion *)
  pRA = propMotion[[1]] * mas2r;
  pDecl = propMotion[[2]] * mas2r;
  pYrs = propMotion[[3]];
  (*Print[{α,δ}];*)
  α = α + pRA * pYrs;
  δ = δ + pDecl * pYrs;
  x = Cos[α] Cos[δ];
  y = Sin[α] Cos[δ];
  z = Sin[δ];
  (*Print[{α,δ,pRA,pDecl,pYrs}];*)
  {x, y, z}
 ]
```

In[48]:=
```
(* rectangular to equatorial *)
RectEqu[rect_] :=
 Module[
  {x, y, z, α, δ},
  x = rect[[1]];
  y = rect[[2]];
  z = rect[[3]];
  α = ArcTan[x, y] / (15 Degree); (* radians to degrees to hours (15°=1^h) *)
  α = If[α < 0, α + 24, α];
  δ = ArcSin[z] / Degree; (* degrees *)
  {α, δ}
 ]
```

```
(* Conversion from equinox date to J2000 *)
EqxToJ2000[dtRef_, α_, δ_, pmRA_, pmDecl_] :=
 Module[
  (* local variables *)
  {daysInCentury, dtJ2000, pmYrs, propMotion, rotMat, matFromEqxToJ2K,
   coords, rectEqx, rectJ2K, equCoords},
  (* constants *)
  daysInCentury = 36525;
  dtJ2000 = DateObject[{2000, 1, 1, 12, 0, 0}, TimeZone → "GMT"];
  (* J2000 *)
  (* proper motion *)
  pmYrs = (JulianDate[dtRef] - JulianDate[dtJ2000]) / daysInCentury;
  propMotion = {pmRA, pmDecl, pmYrs};
  (* compute transf. matrix *)
  rotMat = pmat[dtRef];
  matFromEqxToJ2K = Inverse[rotMat];
  (*Print[matFromEqxToJ2K//MatrixForm];*)
  (* assign object equatorial coordinates *)
  coords = {α, δ};
  (* convert to rectangular (cartesian) coordinates with proper motion *)
  rectEqx = EquRect[coords, propMotion];
  (* apply rotation matrix *)
  rectJ2K = matFromEqxToJ2K.rectEqx;
  (*convert to equatorial coordinates*)
  equCoords = RectEqu[rectJ2K];
  (*return in HMS form for RA=1, and DMS form for Decl=2 *)
  {UnitConvert[Quantity[equCoords[[1]], "Hours"],
    MixedUnit[{"Hours", "Minutes", "Seconds"}]], DMSString[equCoords[[2]]]}
 ]
```

```
(* Conversion from J2000 to equinox date *)
J2000ToEqx[dtRef_, α_, δ_, pmRA_, pmDecl_] :=
 Module[
  {daysInCentury, dtJ2000, pmYrs, propMotion, matFromJ2KToEqx,
   coords, rectEqx, rectJ2K, equCoords},
  daysInCentury = 36525;
  dtJ2000 = DateObject[{2000, 1, 1, 12, 0, 0}, TimeZone → "GMT"];
  (* J2000 *)
  (* proper motion *)
  pmYrs = (JulianDate[dtJ2000] - JulianDate[dtRef]) / daysInCentury;
  propMotion = {pmRA, pmDecl, pmYrs};
  (* compute transf. matrix *)
  matFromJ2KToEqx = pmat[dtRef];
  (*Print[matFromJ2KToEqx//MatrixForm];*)
  (* assign object equatorial coordinates *)
  coords = {α, δ};
  (* convert to rectangular (cartesian) coordinates with proper motion *)
  rectJ2K = EquRect[coords, propMotion];
  (* apply rotation matrix *)
  rectEqx = matFromJ2KToEqx.rectJ2K;
  (*convert to equatorial coordinates*)
  equCoords = RectEqu[rectEqx];
  (*return in HMS form for RA=1, and DMS form for Decl=2 *)
  {UnitConvert[Quantity[equCoords[[1]], "Hours"],
    MixedUnit[{"Hours", "Minutes", "Seconds"}]], DMSString[equCoords[[2]]]}
 ]
```

```
In[51]:= matJ2KToDateTest = pmat[DateObject[{-1375, 5, 3, 13, 52, 19.2}, TimeZone → "GMT"]];
matJ2KToDateTest // MatrixForm
matDateToJ2KTest = Inverse[matJ2KToDateTest];
matDateToJ2KTest // MatrixForm
```

```
In[70]:= (* Polaris *)
J2000ToEqx[DateObject[{-10950, 1, 1, 0, 0, 0}, TimeZone → "GMT"],
 FromDMS[{2, 32, 38.78}],
 FromDMS[{89, 16, 9.1}],
 44.48,
 -11.85]
```

```
In[71]:= (* Polaris *)
EqxToJ2000[DateObject[{-10950, 1, 1, 0, 0, 0}, TimeZone → "GMT"],
 FromDMS[{18, 6, 42.36}],
 FromDMS[{40, 28, 8.35}],
 44.48,
 -11.85]
```