Ridge-instantaneous-velocity.nb

 Written August 1, 2018, by Vince Cronin; revised 28 November 2018

A code to compute the meridional and zonal components of the instantaneous velocity of a point along a mid-ocean ridge

---

Introduction

The background of this analysis is explained in Cronin (1994).

This version contains data that is useful to investigate the North American-Nubian plate system.

Input

Angular velocities of the 2 - plate system

The following data describe the instantaneous motion of individual plates in an NNR reference frame (e.g., Argus, D.F., Gordon, R.G., and DeMets, C., 2011). A code to compute the meridional and zonal components of the instantaneous velocity of a point along a mid-o

NA refers to the North American plate, and NB refers to the Nubian plate.

```
latNA = -4.85; longNA = -80.64; angvelNA = 0.209;
```

```
latNB = 47.68; longNB = -68.44; angvelNB = 0.292;
```

The following data describe the instantaneous motion of one plate as observed from another plate (e.g., DeMets, C., Gordon, R.G., and Argus, D.F., 2010).

Angular velocities describe counter - clockwise (positive) rotation of the first listed plate relative to the plate listed second

```
latNBNA = 79.2; longNBNA = 40.2; angvelNBNA = 0.233;
```

Locations of the endpoints of the transform fault

Kane transform fault

faultName = "Kane";

latWestEnd = 23.842; longWestEnd = -46.345;

latEastEnd = 23.649; longEastEnd = -44.855;

Atlantis transform fault

faultName = "Atlantis";

latWestEnd = 30.135; longWestEnd = -42.693;

latEastEnd = 29.985; longEastEnd = -42.018;

Oceanographer transform fault

```
faultName = "Oceanographer";
latWestEnd = 35.287; longWestEnd = -36.327;
latEastEnd = 35.026; longEastEnd = -34.994;
```

User - defined functions

```
convert2cart[lat_, long_] := {Cos[lat Degree] Cos[long Degree],
    Cos[lat Degree] Sin[long Degree], Sin[lat Degree]};
```

```
findGeogCoord[vect_] := Module[{lat, long, a, b, c, d, e, f}, a = ArcSin[vect[[3]]];
    b = {vect[[1]], vect[[2]], 0};
    c = If[((Abs[vect[[1]]] < (1 × 10^-14)) && (Abs[vect[[2]]] < (1 × 10^-14))),
       {1, 1, 0}, {vect[[1]] / Norm[b], vect[[2]] / Norm[b], 0}];
    d = {1, 0, 0};
    e = VectorAngle[c, d];
    f = If[(vect[[2]] < 0), (-e), (e)];
    lat = a (180 / π);
    long = If[
       ((Abs[vect[[1]]] < (1 × 10^-14)) && (Abs[vect[[2]]] < (1 × 10^-14))), 0, (f (180 / π))];
    {lat, long}];
```

```
unitVect3D[vect_] :=
   {(vect[[1]] / Norm[vect]), (vect[[2]] / Norm[vect]), (vect[[3]] / Norm[vect])};
```

```
tangentialVelocityComputer[latPole_, longPole_, latPoint_, longPoint_, angVel_] :=
  Module[{a, b, c, d, e, answer}, a = {Cos[latPole Degree] Cos[longPole Degree],
     Cos[latPole Degree] Sin[longPole Degree], Sin[latPole Degree]};
   b = {Cos[latPoint Degree] Cos[longPoint Degree],
     Cos[latPoint Degree] Sin[longPoint Degree], Sin[latPoint Degree]};
   c = VectorAngle[a, b] / Degree;
   d = (angVel * 6371.01 * 2 * π * Sin[c Degree]) / 360;
   e = unitVect3D[Cross[a, b]] * d;
   answer = e;
   answer];
```

```
findTangentialVector[locVectPole_, locVectPoint_, angVel_] :=
  Module[{a, b, c, answer}, a = VectorAngle[locVectPole, locVectPoint] / Degree;
   b = (angVel * 6371.01 * 2 * π * Sin[a Degree]) / 360;
   c = unitVect3D[Cross[locVectPole, locVectPoint]] * b;
   answer = c;
   answer];
```

We now know the length of each side of a plane triangle (two plane triangles, in fact -- one for each end of the transform fault) and need to know the interior angles. Given a triangle whose sides have lengths a, b, and c (where the angle between sides a and b is C, etc.), the law of cosines holds that

$$c^2 = a^2 + b^2 - 2ab\,\mathrm{Cos}[C]$$

Re-arranging to isolate the unknown quantity, the angle C,

$$C = \mathrm{ArcCos}\left[\frac{a^2 + b^2 - c^2}{2\,ab}\right]$$

```
interiorAngles[a_, b_, c_] :=
  Module[{α, β, χ, result}, α = ArcCos[(b^2 + c^2 - a^2) / (2 * b * c)] / Degree;
   β = ArcCos[(a^2 + c^2 - b^2) / (2 * a * c)] / Degree;
   χ = ArcCos[(a^2 + b^2 - c^2) / (2 * a * b)] / Degree;
   result = {a, b, c, α, β, χ};
   result];
```

Computation

Convert the input data from geographic coordinates (lat, long) to cartesian coordinates

```
poleNA = convert2cart[latNA, longNA];
```

```
poleNB = convert2cart[latNB, longNB];
```

```
poleNBNA = convert2cart[latNBNA, longNBNA];
```

```
pointWestEnd = convert2cart[latWestEnd, longWestEnd];
pointEastEnd = convert2cart[latEastEnd, longEastEnd];
```

Combine unit location vector coordinates with the corresponding angular speed

```
angVelVectorNA = {poleNA[[1]], poleNA[[2]], poleNA[[3]], angvelNA};

angVelVectorNB = {poleNB[[1]], poleNB[[2]], poleNB[[3]], angvelNB};

angVelVectorNBNA = {poleNBNA[[1]], poleNBNA[[2]], poleNBNA[[3]], angvelNBNA};
```

Determine the tangential velocity vectors at both ends of the

```
tangVelVectWNA = findTangentialVector[poleNA, pointWestEnd, angVelVectorNA[[4]]];

tangSpeedWNA = Norm[%];

tangVelVectENA = findTangentialVector[poleNA, pointEastEnd, angVelVectorNA[[4]]];

tangSpeedENA = Norm[%];

tangVelVectWNB = findTangentialVector[poleNB, pointWestEnd, angVelVectorNB[[4]]];

tangSpeedWNB = Norm[%];

tangVelVectENB = findTangentialVector[poleNB, pointEastEnd, angVelVectorNB[[4]]];

tangSpeedENB = Norm[%];

tangRelVelVectWNBNA =
  findTangentialVector[poleNBNA, pointWestEnd, angVelVectorNBNA[[4]]];

tangSpeedWNBNA = Norm[%];

tangRelVelVectENBNA =
  findTangentialVector[poleNBNA, pointEastEnd, angVelVectorNBNA[[4]]];

tangSpeedENBNA = Norm[%];
```

Initial check for closure of vector triangle for the west end of the transform fault

```
firstStepW = interiorAngles[tangSpeedWNA, tangSpeedWNB, tangSpeedWNBNA];

test = firstStepW[[4]] + firstStepW[[5]] + firstStepW[[6]];

check1 = If[(Abs[test - 180] < (1 × 10^-10)), "OK", "We have a problem here"];
```

Instantaneous velocities of west end of the transform

The meridional velocity (mm/yr) is the velocity toward or away from the specified pole of instantaneous relative plate motion.

```
meridionalVelW1 = Abs[tangSpeedWNA Sin[firstStepW[[5]] Degree]];
```

```
meridionalVelW2 = Abs[tangSpeedWNB Sin[firstStepW[[4]] Degree]];

check2 = If[(Abs[Abs[meridionalVelW1] - Abs[meridionalVelW2]] < (1 × 10^-10)),
   "OK", "We have a problem here"];
```

The zonal velocity (mm/yr) is the velocity around the specified pole of instantaneous relative plate motion. A positive zonal velocity involves a counter - clockwise (positive) rotation around that pole.

```
zonalVelW1 = (tangSpeedWNA Cos[firstStepW[[5]] Degree]) - tangSpeedWNBNA/2;

zonalVelW2 = (tangSpeedWNB Cos[firstStepW[[4]] Degree]) - tangSpeedWNBNA/2;

check3 = If[(Abs[Abs[zonalVelW1] - Abs[zonalVelW2]] < (1 × 10^-10)),
   "OK", "We have a problem here"];
```

Initial check for closure of vector triangle for the east end of the transform fault

```
firstStepE = interiorAngles[tangSpeedENA, tangSpeedENB, tangSpeedENBNA];

test = firstStepE[[4]] + firstStepE[[5]] + firstStepE[[6]]

180.
```

Instantaneous velocities of east end of the transform

The meridional velocity (mm/yr) is the velocity toward or away from the specified pole of instantaneous relative plate motion.

```
meridionalVelE1 = tangSpeedENA Sin[firstStepE[[5]] Degree];

meridionalVelE2 = tangSpeedENB Sin[firstStepE[[4]] Degree];

check4 = If[(Abs[Abs[meridionalVelE1] - Abs[meridionalVelE2]] < (1 × 10^-10)),
   "OK", "We have a problem here"];
```

The zonal velocity (mm/yr) is the velocity around the specified pole of instantaneous relative plate motion.

```
zonalVelE1 = (tangSpeedENA Cos[firstStepE[[5]] Degree]) - tangSpeedENBNA/2;

zonalVelE2 = (tangSpeedENB Cos[firstStepE[[4]] Degree]) - tangSpeedENBNA/2;

check5 = If[(Abs[Abs[zonalVelE1] - Abs[zonalVelE2]] < (1 × 10^-10)),
   "OK", "We have a problem here"];

consolidatedChecks = {check1, check2, check3, check4, check5};
```

Output

Did the code generate expected results at five places in the process?

```
consolidatedChecks
{OK, OK, OK, OK, OK}
```

```
faultName
Oceanographer
```

The meridional velocity (mm/yr) is the velocity toward or away from the specified pole of instantaneous relative plate motion.

The meridional velocity of the west end of the fault is ...

```
Abs[meridionalVelW1]
13.2763
```

The meridional velocity of the east end of the fault is ...

```
Abs[meridionalVelE1]
13.7438
```

The zonal velocity (mm/yr) is the velocity around the specified pole of instantaneous relative plate motion.

The zonal velocity of the west end of the fault is ...

```
Abs[zonalVelW1]
4.1689
```

The zonal velocity of the east end of the fault is ...

```
Abs[zonalVelE1]
3.95168
```

Other results

```
tangVelVectWNA
{-14.1489, -3.46773, 13.2045}
```

```
tangSpeedWNA
19.6615
```

```
tangVelVectENA = findTangentialVector[poleNA, pointEastEnd, angVelVectorNA[[4]]]
{-14.0365, -3.47968, 13.559}

tangSpeedENA
19.8236

tangVelVectWNB = findTangentialVector[poleNB, pointWestEnd, angVelVectorNB[[4]]]
{-0.135767, 11.1474, 9.48569}

tangSpeedWNB
14.6376

tangVelVectENB = findTangentialVector[poleNB, pointEastEnd, angVelVectorNB[[4]]]
{-0.394352, 11.4947, 9.86632}

tangSpeedENB
15.1535

tangRelVelVectWNBNA =
 findTangentialVector[poleNBNA, pointWestEnd, angVelVectorNBNA[[4]]]
{14.1163, 14.5943, -3.85373}

tangSpeedWNBNA
20.6667

tangRelVelVectENBNA =
 findTangentialVector[poleNBNA, pointEastEnd, angVelVectorNBNA[[4]]]
{13.7503, 14.9445, -3.84352}

tangSpeedENBNA
20.6684
```

Tangential velocity triangle: a, b, c, α, β, χ

```
firstStepW
{19.6615, 14.6376, 20.6667, 65.0935, 42.473, 72.4335}

firstStepE
{19.8236, 15.1535, 20.6684, 65.0902, 43.892, 71.0178}
```

References

Argus, D.F., Gordon, R.G., and DeMets, C., 2011, Geologically current motion of 56 plates relative to the no-net-rotation reference frame: Geochemistry, Geophysics, Geosystems, v. 12(11), Q1101, doi:10.1029/2011GC003751.

Cronin, V.S., 1994, Instantaneous velocity of mid-ocean ridges: Tectonophysics, v. 230, p. 151-159.

DeMets, C., Gordon, R.G., and Argus, D.F., 2010, Geologically current plate motions: Geophysical Journal International, v. 181, p. 1-80, doi:10.1111/j.1365-246X.2010.04491.x.