Isochron-TF-Offset.nb

A method for locating the point of intersection between an isochron and the trace of an oceanic fracture zone

---

Introduction

The background of this analysis is explained in Cronin (1991). The general data source for this analysis is described by Seton et al. (2014) and is accessible online via http://www.soest.hawaii.edu/PT/GSFML/ (Wessel, and Müller, 2018)

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

Input Data

Explanation

The input code involves establishing a datafile name and filling it with data imported from an Excel (.xlsx) spreadsheet. One way to complete the "Import" line is to choose **File Path...** from the **Insert** menu. The first two to four records have location data from the isochron, and the last four records of the eight total records have location data along the fracture zone adjacent to that isochron.
Each record has the following cells: isochron name or FZ name, side of isochron or “c” for points on FZ, longitude in decimal degrees with west longitudes as negative values, latitude in decimal degrees with south longitudes as negative values, source of data, original record number, age of younger edge of isochron, age of older edge of isochron.
A screen capture of an example of a valid input datafile called **testFileKaneNOAM-N.xlsx** is shown below.

	A	B	C	D	E	F	G	H
1	C3n.4n	o	-46.9457	24.3822	MerkourievDeMets2013GJI	9468	4.997	5.235
2	C3n.4n	o	-46.9579	24.2827	MerkourievDeMets2013GJI	9468	4.997	5.235
3	C3n.4n	o	-46.9659	24.2395	MerkourievDeMets2013GJI	9468	4.997	5.235
4	C3n.4n	o	-46.955	24.0872	MerkourievDeMets2013GJI	9468	4.997	5.235
5	all	c	-47.25	23.9500	GeoMapApp	0	0	0
6	all	c	-47	23.9100	GeoMapApp	0	0	0
7	all	c	-46.749	23.8820	GeoMapApp	0	0	0
8	all	c	-46.499	23.8580	GeoMapApp	0	0	0

An example of a correct input line, used on Vince' s home iMac to open the file called **test-FileKaneNOAM - N.xlsx** is shown below.

```
initialData = Import["/Users/vince/Desktop/testFileKaneNOAM-N.xlsx"];
```

Input Code

Input the data associated with the isochron - FZ intersection on one side of the fracture zone

```
initialData1 = Import[
   "/Users/vince/Desktop/Kane-Atlantis-Oceanographer-Nov23/All-Oceanographer-
     July2018/Oceanographer_C5n.2n-o/Oceanographer_C5n.2n-o-NOAM-N.xlsx"];
```

```
modInputData1 = Flatten[initialData1, 1];
```

Input the data associated with the isochron - FZ intersection on the other side of the fracture zone

```
initialData2 = Import[
   "/Users/vince/Desktop/Kane-Atlantis-Oceanographer-Nov23/All-Oceanographer-
     July2018/Oceanographer_C5n.2n-o/Oceanographer_C5n.2n-o-NOAM-S.xlsx"];
```

```
modInputData2 = Flatten[initialData2, 1];
```

User-Defined Functions

```
geog2cart[lat_, long_] := {Cos[lat Degree] Cos[long Degree],
   Cos[lat Degree] Sin[long Degree], Sin[lat Degree]};
```

```
cart2geog[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}];
```

```
meanUnitVectNormal4[vect1_, vect2_, vect3_, vect4_] :=
  Module[{a, b, c, d, nAB, nAC, nAD, nBC, nBD, nCD, wtAB,
    wtAC, wtAD, wtBC, wtBD, wtCD, vectAB, vectAC, vectAD, vectBC,
    vectBD, vectCD, longVect, unitResultVect, answer}, a = vect1;
   b = vect2;
   c = vect3;
   d = vect4;
   nAB = Cross[a, b] / Norm[Cross[a, b]];
   wtAB = VectorAngle[a, b] / VectorAngle[a, d];
   vectAB = nAB * wtAB;
   nAC = Cross[a, c] / Norm[Cross[a, c]];
   wtAC = VectorAngle[a, c] / VectorAngle[a, d];
   vectAC = nAC * wtAC;
   nAD = Cross[a, d] / Norm[Cross[a, d]];
   wtAD = VectorAngle[a, d] / VectorAngle[a, d];
   vectAD = nAD * wtAD;
   nBC = Cross[b, c] / Norm[Cross[b, c]];
   wtBC = VectorAngle[b, c] / VectorAngle[a, d];
   vectBC = nBC * wtBC;
   nBD = Cross[b, d] / Norm[Cross[b, d]];
   wtBD = VectorAngle[b, d] / VectorAngle[a, d];
   vectBD = nBD * wtBD;
   nCD = Cross[c, d] / Norm[Cross[c, d]];
   wtCD = VectorAngle[c, d] / VectorAngle[a, d];
   vectCD = nCD * wtCD;
   longVect = vectAB + vectAC + vectAD + vectBC + vectBD + vectCD;
   unitResultVect = longVect / Norm[longVect];
   answer = unitResultVect;
   answer];
```

```
meanUnitVectNormal3[vect1_, vect2_, vect3_] :=
  Module[{a, b, c, nAB, nAC, nBC, wtAB, wtAC, wtBC, vectAB,
    vectAC, vectBC, longVect, unitResultVect, answer}, a = vect1;
   b = vect2;
   c = vect3;
   nAB = Cross[a, b] / Norm[Cross[a, b]];
   wtAB = VectorAngle[a, b] / VectorAngle[a, c];
   vectAB = nAB * wtAB;
   nAC = Cross[a, c] / Norm[Cross[a, c]];
   wtAC = VectorAngle[a, c] / VectorAngle[a, c];
   vectAC = nAC * wtAC;
   nBC = Cross[b, c] / Norm[Cross[b, c]];
   wtBC = VectorAngle[b, c] / VectorAngle[a, c];
   vectBC = nBC * wtBC;
   longVect = vectAB + vectAC + vectBC;
   unitResultVect = longVect / Norm[longVect];
   answer = unitResultVect;
   answer];
```

```
meanUnitVectNormal2[vect1_, vect2_] :=
  Module[{a, b, nAB, wtAB, vectAB, longVect, answer}, a = vect1;
   b = vect2;
   vectAB = Cross[a, b];
   answer = vectAB / Norm[vectAB];
   answer];
```

Computation

One side of the fracture zone

Determine how many rows are in the datafile

```
dim = Dimensions[modInputData1];
```

```
nRows = dim[[1]];
```

Make a table in which all of the location data are in cartesian coordinates, with unit location vectors in an Earth - centered geographic coordinate system

```
cartesianData =
  Table[geog2cart[modInputData1[[i, 4]], modInputData1[[i, 3]]], {i, nRows}];
```

Find the unit vector normal to the set of four points along the isochron

```
result1 =
  If[(nRows == 8), meanUnitVectNormal4[cartesianData[[1]], cartesianData[[2]],
    cartesianData[[3]], cartesianData[[4]]], If[nRows == 7, meanUnitVectNormal3[
     cartesianData[[1]], cartesianData[[2]], cartesianData[[3]]],
    meanUnitVectNormal2[cartesianData[[1]], cartesianData[[2]]]]];
```

Find the unit vector normal to the set of four points along the oceanic fracture zone

```
result2 = If[nRows == 8, meanUnitVectNormal4[cartesianData[[5]],
    cartesianData[[6]], cartesianData[[7]], cartesianData[[8]]],
   If[nRows == 7, meanUnitVectNormal4[cartesianData[[4]],
     cartesianData[[5]], cartesianData[[6]], cartesianData[[7]]],
    meanUnitVectNormal4[cartesianData[[3]], cartesianData[[4]],
     cartesianData[[5]], cartesianData[[6]]]]];
```

Find the point of intersection between the average trend of the fracture zone and the average trend of the isochron

```
intersectionPoint1 =
  If[VectorAngle[cartesianData[[6]], Cross[result1, result2]] > (π/2),
   Cross[result2, result1] / Norm[Cross[result2, result1]],
   Cross[result1, result2] / Norm[Cross[result1, result2]]];
```

```
intersectionPointGeog1 = cart2geog[intersectionPoint1];
```

The geographic coordinates (lat, long) of the point of intersection between the isochron and the fracture zone are ...

```
intersectionPointGeog1
```

```
{35.2805, -36.218}
```

The cartesian-geographic coordinates (x,y,z) of the point of intersection between the isochron and the fracture zone are ...

```
intersectionPoint1
```

```
{0.658598, -0.482338, 0.57758}
```

```
dim1 = dim;
```

```
ClearAll[dim, nRows, cartesianData, result1, result2]
```

Other side of the fracture zone

Determine how many rows are in the datafile

```
dim = Dimensions[modInputData2];
```

```
nRows = dim[[1]];
```

Make a table in which all of the location data are in cartesian coordinates, with unit location vectors in an Earth - centered geographic coordinate system

```
cartesianData =
  Table[geog2cart[modInputData2[[i, 4]], modInputData2[[i, 3]]], {i, nRows}];
```

Find the unit vector normal to the set of four points along the isochron

```
result1 =
  If[(nRows == 8), meanUnitVectNormal4[cartesianData[[1]], cartesianData[[2]],
    cartesianData[[3]], cartesianData[[4]]], If[nRows == 7, meanUnitVectNormal3[
     cartesianData[[1]], cartesianData[[2]], cartesianData[[3]]],
    meanUnitVectNormal2[cartesianData[[1]], cartesianData[[2]]]]];
```

Find the unit vector normal to the set of four points along the oceanic fracture zone

```
result2 = If[nRows == 8, meanUnitVectNormal4[cartesianData[[5]],
    cartesianData[[6]], cartesianData[[7]], cartesianData[[8]]],
   If[nRows == 7, meanUnitVectNormal4[cartesianData[[4]],
     cartesianData[[5]], cartesianData[[6]], cartesianData[[7]]],
    meanUnitVectNormal4[cartesianData[[3]], cartesianData[[4]],
     cartesianData[[5]], cartesianData[[6]]]]];
```

Find the point of intersection between the average trend of the fracture zone and the average trend of the isochron

```
intersectionPoint2 =
  If[VectorAngle[cartesianData[[6]], Cross[result1, result2]] > (π / 2),
   Cross[result2, result1] / Norm[Cross[result2, result1]],
   Cross[result1, result2] / Norm[Cross[result1, result2]]];
```

```
intersectionPointGeog2 = cart2geog[intersectionPoint2];
```

The geographic coordinates (lat, long) of the point of intersection between the isochron and the fracture zone are ...

```
intersectionPointGeog2
```

```
{35.5176, -37.2611}
```

The cartesian-geographic coordinates (x,y,z) of the point of intersection between the isochron and the fracture zone are ...

```
intersectionPoint2
```

```
{0.6478, -0.492797, 0.580953}
```

```
dim2 = dim;
```

```
ClearAll[dim, nRows, cartesianData, result1, result2]
```

Determine the circumferential distance across Earth’s surface between the two intersection points

Constants

```
earthMeanRadius = 6371.01;
```

```
earthMeanCircumference = earthMeanRadius 2 π;
```

```
kmPerDegree = earthMeanCircumference / 360;
```

```
kmPerRadian = earthMeanCircumference / (2 π);
```

Computation

```
distance = VectorAngle[intersectionPoint1, intersectionPoint2] * kmPerRadian;
```

Output

The number of records in the datafiles

```
dim1[[1]]
```

```
8
```

```
dim2[[1]]
```

```
8
```

The two points of isochron - fracture zone intersection used to model the length of the transform fault

```
intersectionPointGeog1
```

```
{35.2805, -36.218}
```

```
intersectionPointGeog2
```

```
{35.5176, -37.2611}
```

The length of the transform fault corresponding to this isochron, expressed in km, is ...

```
N[distance]
```

```
98.1557
```

References

Cronin, V.S., 1987a, Cycloid kinematics of relative plate motion : Geology, v.15, p.1006 - 1009.

Cronin, V.S., 1987b, Kinematics of transform plate - boundary faults re - evaluated[abs.], in Hilde, T.W.C., and Carlson, R.L., [editors], Proceedings of the 1987 Geodynamics Symposium : Geodynamics Research Institute, Texas A & M University, p.84 - 86.

Cronin, V.S., 1987c, Cycloid tectonics : Predicted changes in the position and morphology of mid - ocean ridge segments, transform faults, and oceanic fracture zones[abs.] : EOS (American Geophysical Union Transactions), v.68, no.16, p.408.

Cronin, V.S., 1987d, Cycloid tectonics : The kinematics of transform faulting re - evaluated[abs.] : American Association of Petroleum Geologists Bulletin, v.71, p.544.

Cronin, V.S., 1987e, Cycloid tectonics : Are transform plate - boundary faults "lines of pure slip" along which "crust is conserved"? : Geological Society of America, Abstracts with Programs, v.19, No.7, p.631.

Cronin, V.S., 1988, Cycloid tectonics : Fracture zones as flow lines of transform faults[abs.] : EOS (American Geophysical Union Transactions), v.69, p.1415.

Cronin, V.S., 1989, Cycloid tectonics : Is there a maximum stable length for a transform fault?Geological Society of America, Abstracts with Programs, v.21, no.4, p.8.

Cronin, V.S., 1991, The cycloid relative-motion model and the kinematics of transform faulting, in Hilde, T.W.C., and Carlson, R.L., [editors], Proceedings of the 1987 Geodynamics Symposium: Tectonophysics, v. 187, p. 215-249; corrections to publisher errors, 1991, Tectonophysics, v. 192, no. 3/4, p. 401.

Cronin, V.S., 1992, Instantaneous velocity of mid - ocean ridges[abs.] : EOS (American Geophysical Union Transactions), v.73, No.14, p.284.

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

Merkouriev, S.and DeMets, C., 2008, A high - resolution model for Eurasia - North America plate kinematics since 20 Ma : Geophysical Journal International, v.173 (3), p.1064 - 1083.

Merkouriev, S., and DeMets, C., 2014, High - resolution Quaternary and Neogene reconstructions of Eurasia - North America plate motion : Geophysical Journal International, v.198 (1), p.366 - 384, doi : 10.1093/gji/ggu142.

Merkouriev, S., and DeMets, C., 2014, High - resolution estimates of Nubia - North plate motion, 20 Ma to present : Geophys.J.Int., 196 (3), p.1281 - 1298, doi : 10.1093/gji/ggt463.

Schouten, H., Cande, S.C., and Klitgord, K.D., 1985, Magnetic anomaly profiles, south (22 °N to 28 °N), in Rabinowitz, P.D., and Schouten, H., [editors], Mid - Atlantic Ridge between 22 ° and 38 °N, Atlas 11, Ocean Margin Drilling Program, Regional Atlas Series : Woods Hole, Massachusetts, Marine Science

International.

Seton, M., Whittaker, J.M., Wessel, P., Müller, R.D., DeMets, C., Merkouriev, S., Cande, S., Gaina, C., Eagles, G., Granot, R., Stock, J., Wright, N., and Williams, S.E., 2014, Community infrastructure and repository for marine magnetic identifications : Geochemistry, Geophysics, Geosystems, v.15, p.1629 - 1641, doi : 10.1002/2013 GC005176.

Wessel, P., and Müller, R.D., 2018, The global seafloor fabric and magnetic lineation data base project : accessible via http : // www.soest.hawaii.edu/PT/GSFML/