Evaluating the Alignment of Astronomical Linear Polarization Data, Intermediate Level Software Version 2,
Richard Shurtleff

The Hub Tests are two polarization alignment tests, designed with polarized astronomical sources in mind. By analogy, if the polarization directions were magnetic needle compass directions, then one test finds the best virtual North Pole and determines how well the polarization directions correlate with Local North. That is the ‘alignment test’. The second test, the ‘avoidance’ test, asks how well the directions correlate with Local East. The tests are further described in an article, arXiv:1311.6118, where the tests are applied to an online data set measured, analyzed and published by others. This Mathematica notebook is meant to serve as a practical introduction to the Hub Tests. By replacing the DATA used here, a USER can evaluate other data sets.

July 2023

CONTENTS

Preface

The Hub Tests make use of the concepts of Local North and Local East directions that should be familiar to all. One hopes the Hub Test will appeal to those who seek correlations among the polarization directions of astronomical sources.

The theory is described in an article, “A Large Scale Pattern from Optical Quasar Polarization Vectors, Version 2”, Ref. 1

The computer program is written in Mathematica, Ref. 9. I do not know how to make the pdf file of this notebook run directly as a Mathematica Notebook. A ready-to-run .nb notebook file is currently available in Ref. 2. The pdf file will be (or was) uploaded to viXra and ResearchGate, (‘will be’: upload prediction and ‘was’: possible confirmation).

This notebook is set in the Galactic Coordinate System (ℓ,b). Occasionally, an RA or a dec may appear where an ℓ or a b should be. Both RA and ℓ are longitudes, one in the Equatorial Coordinate System, one in the Galactic Coordinate System. Likewise, dec and b are latitudes. (RA - Right Ascension, dec - declination, ℓ - Galactic longitude, b - Galactic latitude.)

The hubs have more than one name. $H_{\text{min}} \equiv H_{\text{align}}$ and $H_{\text{max}} \equiv H_{\text{avoid}}$. It is better to use H_{align} and H_{avoid} because the hubs are not themselves extreme values. They are, instead, places where the fundamental alignment function $\overline{\eta}(H)$ has extreme values, $\overline{\eta}_{\text{min}}$ and $\overline{\eta}_{\text{max}}$.

References

1. Shurtleff, R., A Large Scale Pattern from Optical Quasar Polarization Vectors. arXiv e-prints, arXiv:1311.6118 A copy is available on ResearchGate.
2. Shurtleff, R., Links to ready-to-run files of this notebook: (a) https://www.wolframcloud.com/obj/shurtleffr/Published/20230629IntermediateKitForHubTest5.nb and (b) https://www.dropbox.com/s/1yewu76s62ot79q/20230629IntermediateKitForHubTest5.nb?dl=0
3. Wikipedia contributors. “Aitoff projection.” Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 10 Jan. 2023. Web. 7 Apr. 2023.
4. Westmeier, T., Conversions of Coordinates, Link: https://www.atnf.csiro.au/people/Tobias.Westmeier/tools_coords.php
5. VizieR, J/A+A/441/915, Link: https://vizier.cds.unistra.fr/viz-bin/VizieR?-source=J/A+A/441/915
6. Hutsemekers, D., Cabanac, R., Lamy, H., Sluse, D. (2005). Mapping extreme-scale alignments of quasar polarization vectors, A\&A, 441, pp. 915-930. doi:10.1051/0004-6361:20053337
7. Shurtleff, R., Link to 20230702RunDataU2000.dat, https://www.dropbox.com/scl/fi/60xxod61lqro4qyn4vrff/20230702RunDataU2000.dat?rlkey=2eutngk2jvbm8sjrwo7ptcho4&dl=0
8. Shurtleff, R., Link to “20230703randRuns2000.dat”, https://www.dropbox.com/scl/fi/sgf6lexhixrpv14lb0i6d/20230703randRuns2000.dat?rlkey=bbk5ck6uayyepk185aerif608&dl=0
9. Wolfram Research, Inc., Mathematica, Version 13.2, Champaign, IL (2022).

A1. Preliminary, useful functions

The section has trig functions for use with angles in degrees, rectangular to spherical coordinate conversion functions, Aitoff Plot functions, arithmetic mean and standard deviation functions.

Definitions:

home directory - the directory containing the notebook. So saved files and uploaded files are most likely there, although they can move away.

sin, cos, tan, arcsin, arccos, arctan these take arguments in degrees, not radians. Mathematica has capitalized Sin, Cos, ... and takes radians

er, eN, eE[*l*,*b*] - unit vectors in a 3D Cartesian coordinate system:
er - from Origin to Source at [*l*,*b*], eN, eE Local North, East at Source
[*l*,*b*] - galactic longitude, latitude notation

*l*FROMr(er) - longitude determined for a radial unit vector er *b*FROMr(er) - latitude determined for a radial unit vector er

Aitoff Plot Functions, Ref. 3
αH , xH, yH, where xH, yH is centered on $\alpha = 0$.
xHGal, yHGal The plots of the sky in Galactic coordinates have the *l* axis running from +180° on the left to -180° on the right. Angles *l* and *b* are in degrees

mean - the arithmetic average of a set of numbers, $\frac{1}{N}\sum_{i=1}^{N} n_i$
stanDev - the standard deviation. Given a set of *N* numbers n_i with mean value *m*, the standard deviation is $\left(\frac{1}{N}\sum_{i=1}^{N} (n_i - m)^2\right)^{1/2}$, the square root of the average of the squares of the differences of the numbers with the mean. Note that we divide by *N* to get the average of the deviations squared.

In[1]:=
```
homeDirectory = NotebookDirectory[];
```

In[2]:=
```
(*We work with degrees, so define convenient functions.*)
cos[θ_] := cos[θ] = Cos[θ (2. π/360.)];
sin[θ_] := sin[θ] = Sin[θ (2. π/360.)];
tan[θ_] := tan[θ] = Tan[θ (2. π/360.)];
arccos[x_] := arccos[x] = ArcCos[x] (360./2. π);
arcsin[x_] := arcsin[x] = ArcSin[x] (360./2. π);
arctan[x_] := arctan[x] = ArcTan[x] (360./2. π)
```

In[5]:=
```
mean[data_] := mean[data] = (1 / Length[data]) Sum[data[[i4]], {i4, Length[data]}];
(* arithmetic average *)
stanDev[data_] :=
 stanDev[data] = ((1 / Length[data]) Sum[(data[[i5]] - mean[data])^2, {i5, Length[data]}])^(1/2)
  (*standard deviation*)
```

In[7]:=
```
(* For a Source at  (gLON,b): er, eN,
eE are unit vectors from Origin to Source, local North, local East, resp.  *)
er[ℓ_, b_] := er[ℓ, b] = {cos[ℓ] × cos[b], sin[ℓ] × cos[b], sin[b]}
eN[ℓ_, b_] := eN[ℓ, b] = {-cos[ℓ] × sin[b], -sin[ℓ] × sin[b], cos[b]}
eE[ℓ_, b_] := eE[ℓ, b] = {-sin[ℓ], cos[ℓ], 0}
{"Check er.er = 1, er.eN = 0, er.eE = 0, eN.eN
   = 1, eN.eE = 0,eE.eE = 1, erXeE = eN, eEXeN = er, eNXer = eE: ",
 {0} == Union[Flatten[Simplify[{er[ℓ, b].er[ℓ, b] - 1, er[ℓ, b].eN[ℓ, b], er[ℓ, b].eE[ℓ, b],
      eN[ℓ, b].eN[ℓ, b] - 1, eN[ℓ, b].eE[ℓ, b], eE[ℓ, b].eE[ℓ, b] - 1, Cross[er[ℓ, b], eE[ℓ, b]] -
       eN[ℓ, b], Cross[eE[ℓ, b], eN[ℓ, b]] - er[ℓ, b], Cross[eN[ℓ, b], er[ℓ, b]] - eE[ℓ, b]}]]]}
```

Out[10]=
```
{Check er.er = 1, er.eN = 0, er.eE = 0, eN.eN = 1,
   eN.eE = 0,eE.eE = 1, erXeE = eN, eEXeN = er, eNXer = eE: , True}
```

Get (ℓ,b) in degrees from radial vector r:

```
In[11]:= ℓFROMr[r_] := (N[arctan[Abs[r〚2〛/r〚1〛]]] /; (r〚2〛 ≥ 0 && r〚1〛 > 0))
ℓFROMr[r_] := (N[180. - arctan[Abs[r〚2〛/r〚1〛]]] /; (r〚2〛 ≥ 0 && r〚1〛 < 0))
ℓFROMr[r_] := (N[-180. + arctan[Abs[r〚2〛/r〚1〛]]] /; (r〚2〛 < 0 && r〚1〛 < 0))
ℓFROMr[r_] := (N[-arctan[Abs[r〚2〛/r〚1〛]]] /; (r〚2〛 < 0 && r〚1〛 > 0))
ℓFROMr[r_] := (90. /; (r〚2〛 ≥ 0 && r〚1〛 == 0))
ℓFROMr[r_] := (-90. /; (r〚2〛 < 0 && r〚1〛 == 0))
```

```
In[17]:= bFROMr[r_] := (N[arctan[r〚3〛/√(r〚1〛^2 + r〚2〛^2)]] /; (√(r〚1〛^2 + r〚2〛^2) > 0))
bFROMr[r_] := (Sign[r〚3〛] 90. /; (√(r〚1〛^2 + r〚2〛^2) == 0))
```

The following Aitoff Plot formulas can be found in, for example, Ref. 3.

```
In[19]:= αHA[ℓ_, b_] := αHA[ℓ, b] = arccos[cos[b] × cos[ℓ/2.]]
xH[ℓ_, b_] := xH[ℓ, b] = (2. cos[b] × sin[ℓ/2.]) / Sinc[αHA[ℓ, b]]
yH[ℓ_, b_] := yH[ℓ, b] = sin[b] / Sinc[((2. π)/360.) αHA[ℓ, b]]
```

```
In[22]:= (*The plots of the sky in Galactic coordinates have the ℓ axis running from +
 180° on the left to -180° on the right. Angles ℓ and b are in degrees*)
xHGal[ℓ_, b_] := (*xHGal[ℓ,b]= *) (2. cos[b] × sin[-ℓ/2.]) / Sinc[((2. π)/360.) αHA[-ℓ, b]]
yHGal[ℓ_, b_] := (*yHGal[ℓ,b]=*) sin[b] / Sinc[((2. π)/360.) αHA[-ℓ, b]]
```

For Aitoff Plots centered on (0°,0°). (Some adjustments may be needed. The plots are not used in this notebook.) gLON = ℓ and gLAT = b.

```
In[24]:= xH0[gLON_, gLAT_] := (*xH0[gLON,gLAT]=*)
 (2. Cos[((2. π)/360.) gLAT] Sin[((2. π)/360.) (gLON - 180.)/2.]) / Sinc[αHA[(gLON - 180.), gLAT]]
yH0[gLON_, gLAT_] := (*yH0[gLON,gLAT]=*) Sin[((2. π)/360.) gLAT] / Sinc[αHA[(gLON - 180.), gLAT]]
```

For Aitoff Plots centered on (180°,0°). (Some adjustments may be needed. The plots are not used in this notebook.) gLON = ℓ and gLAT = b.

```
In[26]:= xH180[gLON_, gLAT_] := (*xH180[gLON,gLAT]=*)
 (2. Cos[((2. π)/360.) gLAT] Sin[((2. π)/360.) (gLON - 180.)/2.]) / Sinc[αHA[(gLON - 180.), gLAT]]
yH180[gLON_, gLAT_] := (*yH180[gLON,gLAT]=*)
 Sin[((2. π)/360.) gLAT] / Sinc[αHA[(gLON - 180.), gLAT]]
```

```
After Sec. A1, time and memory used are 3.282 seconds and 138 801 472 bytes.
```

A2. Catalog Data

For each of the 355 QSOs, the needed DATA consists of the location of the source (ℓ,b), the position angle ψ, and the uncertainty in ψ. The DATA has been converted from B1950.0 coordinates to J2000.0 Galactic coordinates by Ref. 4. The conversion process is detailed in Ref. 1.

The DATA is derived from an online catalog produced and analyzed in Hutsemekers 2005 Ref. 5,6.

data	1. ℓ Galactic longitude 2. b latitude 3. ψ polarization position angle 4. $\sigma\psi$ uncertainty in ψ (experimental error)
nSrc	the number of sources (355)
ℓi, bi	longitude, latitude of i^{th} source
ψi	polarization position angle in Galactic coords.
$\sigma\psi$i	uncertainty in the polarization position angle
ri	unit radial vector to i^{th} source
vNi, vEi	Local North, East unit vectors at i^{th} source, referenced to the Galactic North Pole
vψi	unit vector along polarization direction

The table ‘data’ should be hidden in the following cell. If the cell is hidden, Open the cell to find:
data = {{93.209,-66.509,6.362,12.},{107.149,-45.383,128.596,7.}, ...,{98.441,-57.954,66.671,7.}}

In[32]:=
```
(*This cell would normally be hidden. Go to Cell - Cell Properties - Open to hide it.*)
data = {{93.20861155459032`, -66.50870391050688`, 6.362455192493305`, 12},
   {107.14948078363278`, -45.38309660790614`, 128.59643487190402`, 7},
   {100.7854284574113`, -58.9503431294937`, 141.62178343307275`, 6},
   {102.39608578101563`, -61.2340588520829`, 134.20568078378878`, 13},
   {103.76496680266416`, -61.651570012284125`, 132.00724321865718`, 10},
   {111.75932828800691`, -46.50212726559873`, 147.3410782650579`, 13},
   {107.58256166408503`, -60.60108751921669`, 39.65463388104803`, 7},
   {106.91577957699039`, -63.92432726573619`, 4.249847932020008`, 14},
   {115.46821128634694`, -39.83500572531703`, 97.230930992251`, 14},
   {109.33022103449977`, -63.76278098795589`, 121.11749654866682`, 13},
   {112.30654357217857`, -61.98506574893724`, 167.4820756092585`, 14},
   {120.05223665711517`, -34.53718677651223`, 105.9640291763511`, 3},
   {-48.13186214659006`, -85.848353656613`, 149.76287411691337`, 4},
   {122.46138875931678`, -34.79791350517866`, 49.540580694231586`, 9},
   {122.7385999031094`, -33.39949369030997`, 98.26585999076572`, 5},
   {123.38144293040958`, -50.19926117691099`, 7.653260899078058`, 3},
   {123.52390435292247`, -33.497970477619496`, 118.46767527894956`, 14},
   {125.11268019455972`, -46.87544608038943`, 13.051706153319623`, 13},
   {-143.84326912771718`, -87.77925165908769`, 78.99584950825067`, 5},
   {125.79114225387804`, -36.445931967723155`, 117.24062276358238`, 8},
   {127.12883959205149`, -49.51201084527512`, 108.23848481515354`, 10},
   {126.93891290209368`, -36.79316573217359`, 110.11914768613221`, 2},
   {127.9293925695893`, -40.947184535293665`, 114.29683624940134`, 5},
   {131.79331088256671`, -61.01013654904819`, 135.19767991944562`, 13},
   {134.28540529485122`, -63.56199731851372`, 65.99150960626953`, 6},
   {128.41845106049664`, -32.67950515870394`, 57.46151150938257`, 13},
   {158.90890473419003`, -78.60209776443712`, 159.80442103513758`, 10},
   {131.63069199422483`, -40.83145340471579`, 93.77863433022424`, 9},
```

```
{131.9405569811039`, -42.414601334037684`, 119.5727938076729`, 11},
{-147.11138405432047`, -83.5553321056786 3`, 64.44010570066436`, 1},
{137.09334542786553`, -57.67881705343723`, 46.48238062994784`, 6},
{132.50901176256485`, -36.27297847077511`, 130.63220771263278`, 14},
{134.79881973346366`, -37.461199158418644`, 98.51991380895926`, 9},
{136.62474524204436`, -40.73594532614322`, 36.08130808290733`, 6},
{149.21557356303902`, -61.938261180350956`, 37.89696643049302`, 8},
{148.56396736104153`, -61.20084903807302`, 131.46999939977897`, 14},
{158.45430623287405`, -68.86539000607998`, 136.50328714488097`, 11},
{148.30271475060246`, -55.63168927987494`, 108.63691084844366`, 3},
{150.52316827064402`, -57.829452394444985`, 116.76710868159768`, 5},
{146.49620476291327`, -50.895500648283374`, 117.99964077062437`, 13},
{144.46796772189273`, -42.98102618773851`, 46.14815179710926`, 9},
{172.6971690476706`, -67.28762149969`, 140.255269175467`, 13},
{-174.3852690251957`, -70.31094000109339`, 42.38852151033898`, 8},
{142.02966356423235`, -30.62067159093578 5`, 97.63282846770427`, 6},
{157.57731288437748`, -55.028618370884615`, 171.7155449945951`, 7},
{-83.72679258106223`, -61.9132035215285`, 127.58467706482779`, 1},
{154.20792076110888`, -46.598914533847754`, 93.07519177487329`, 6},
{-179.00024661911254`, -61.552352329491356`, 115.0499391752982`, 3},
{171.71585029517638`, -56.950103342122055`, 26.03967074876502`, 13},
{150.7548056675033`, -32.65959484533822`, 71.99972434723725`, 5},
{174.1999123365264`, -56.22328389581604 6`, 119.07067559395507`, 10},
{174.07535815643152`, -56.1512016199509 1`, 178.1741627861355 3`, 8},
{170.8810267884461`, -51.50289156017339`, 125.80859831850667`, 5},
{172.04071095088176`, -51.934577134422724`, 0.8973795530538601`, 6},
{-145.43207641972373`, -60.22655500065499`, 155.15588586233278`, 1},
{161.91652144228826`, -30.87405581863997`, 110.34508858167175`, 8},
{179.49914254227122`, -46.203002176559096`, 77.23898537512727`, 6},
{166.91332971679623`, -31.69717213690917`, 78.75927094126645`, 8},
{-115.2277295991651`, -54.15521499309595`, 14.749147486929601`, 3},
{-119.17133843207529`, -54.22590306418792`, 130.1055887511727`, 10},
{-172.2382716110451`, -42.629514817345935`, 148.53354658814516`, 4},
{175.79645497795423`, -31.2794123769249`, 22.54174573331285`, 9},
{-177.92175429091827`, -35.31286041912385`, 107.0069017418943`, 10},
{-163.49177861288405`, -42.722167252738984`, 134.87960397411777`, 4},
{-122.28757823668138`, -48.49132516420969`, 157.9203854763207`, 14},
{-154.35711520475093`, -42.67416149207217`, 105.25719981619335`, 4},
{-155.1693449505115`, -41.84561981245465`, 71.9137919914304`, 5},
{-161.02003251310452`, -36.92465318795607`, 85.98996253718462`, 8},
{-164.8841456398158`, -33.262288694541894`, 57.31006854761378`, 1},
{-119.43352726832848`, -44.58124939523184`, 85.86510722135486`, 14},
{-119.37310449495907`, -43.79443436853239`, 3.438661950576527 5`, 7},
{-111.65419474861393`, -41.6955205983824`, 117.1572641719386 4`, 6},
{-140.2345491979936`, -35.99886195659918`, 106.42129654332966`, 12},
{-117.32378659333293`, -39.60268478002701`, 144.50757422899642`, 10},
{-131.00859325118836`, -37.06761464502392`, 170.44074285455127`, 9},
```

```
{-136.47792612727528`, -35.07690511750923`, 110.98991188512856`, 1},
{-89.4782412520602`, -36.09260009438631`, 164.72540027207532`, 12},
{-109.94352201387913`, -31.149767888903906`, 55.38510843630977`, 1},
{151.25947514623354`, 31.99369311272642`, 25.728850530192666`, 3},
{169.25316070764632`, 32.40062185120499`, 94.30971823446107`, 2},
{143.6355159863652`, 34.41797872167939`, 175.954931176931`, 12},
{-152.66653987546937`, 32.4017468126744`, 31.137973059314923`, 9},
{-171.4160826076913`, 37.845201728207`, 125.70437897895899`, 6},
{-148.511287717752`, 32.791398856994014`, 83.86094612108252`, 8},
{-150.5052903462451`, 33.733576705467954`, 31.695947520972343`, 3},
{-149.0618566478866`, 33.50458275133709`, 139.42248311557796`, 11},
{-146.3049623826885`, 33.04552302321857`, 39.80459312424951`, 14},
{-153.16585766510167`, 35.57664311109231`, 85.8594868431179`, 1},
{-146.04907170760174`, 34.275871034144565`, 143.64084585142334`, 11},
{-149.1995962160857`, 35.61758573692134`, 111.76786829107792`, 10},
{-148.78077439575534`, 37.28091210613043`, 171.49481901145975`, 9},
{178.46239495241085`, 42.78808578679698`, 142.67184245319368`, 2},
{171.1264591170743`, 42.52240450833849`, 52.69743130627625`, 8},
{-159.3243407232255`, 40.92286243779495`, 24.455367183073896`, 9},
{-152.34183750509914`, 40.79589381930164`, 109.41944689408943`, 12},
{-152.27329909017848`, 41.26938206698193`, 82.28240884097622`, 0},
{-176.21219028198902`, 45.984724618530606`, 12.60364731102982`, 11},
{167.85476343510075`, 46.451267981640676`, 63.47617600906265`, 3},
{-162.1188218513057`, 50.06420980116766`, 33.82993819078142`, 5},
{-154.4247660907864`, 50.74186094165244`, 49.94673264493618`, 7},
{158.64793542179063`, 47.905910092063365`, 70.32583603395534`, 3},
{145.8248788005231`, 43.05120652531879`, 47.1561720892076`, 1},
{-148.75222627230733`, 50.97440282504874`, 56.84503984893335`, 13},
{-157.56996638112832`, 52.72904504360933`, 171.69422009552403`, 8},
{-125.90925621488162`, 44.43951549370859`, 17.69167575929302`, 8},
{-134.9268631181847`, 48.93017729032567`, 13.998954073356197`, 4},
{-115.36707694541279`, 41.05729768893662`, 128.81601649868378`, 9},
{-120.75807617083639`, 44.2689065151557`, 84.1597176459923`, 7},
{-128.49451701590345`, 48.523046043766335`, 77.49992242880843`, 4},
{-144.12329208909333`, 53.23944625762858`, 27.334319185458135`, 5},
{-118.46872447126421`, 43.937722798762366`, 60.7328288568869`, 7},
{-118.79878900802014`, 45.0852540247536`, 107.48506253588357`, 7},
{-130.57594366940876`, 52.99737620877864`, 80.27713068261237`, 4},
{-112.19829411471898`, 45.70839115781815`, 74.1779935185637`, 8},
{-119.01484024897655`, 52.497985667498575`, 96.91539566742722`, 11},
{-100.11017305746435`, 43.27440746401823`, 82.09899078086106`, 7},
{144.9056379535577`, 50.39949205061594`, 40.13899321511089`, 12},
{-107.12365072006864`, 50.088398114455934`, 47.04788486887204`, 3},
{-108.72267777607799`, 52.58896083367969`, 101.78935023804569`, 3},
{130.4561029859311`, 38.581557523301214`, 107.33332046592191`, 8},
{164.75075310891032`, 64.4416260938817`, 151.98237373768825`, 2},
{-110.29647741059188`, 60.4939773543791`, 0.0452172839301718`, 12},
```

```
{-94.09330534709612`, 50.3475173804481`, 58.4508867440824`, 12},
{-100.68205575665937`, 56.77395727724658`, 151.15610066025587`, 4},
{-87.23949292363754`, 44.24574410417551`, 81.66117684903068`, 3},
{-83.47627414388431`, 39.60169936002668`, 12.315235427934017`, 1},
{-84.92745193901027`, 43.6205306134327`, 177.56470680154962`, 10},
{-85.83802363920371`, 44.97157609444721`, 19.928743442088944`, 3},
{-164.97578032184904`, 71.9252262166772`, 74.0660208235283`, 10},
{-82.27628350596932`, 41.64994877640228`, 19.63321464061456 5`, 9},
{-94.70364915050286`, 57.84063555384333`, 162.10952608457467`, 8},
{-94.92546588559446`, 58.47999462862927`, 130.91584798391764`, 7},
{-83.49533796172906`, 52.1852731763217`, 28.475745656585847`, 6},
{-83.49533796172906`, 52.1852731763217`, 96.47574565658584`, 13},
{-88.43159768995477`, 59.17292853416558`, 128.41981721204618`, 4},
{-100.73542343522976`, 69.228191798108`, 61.18588342783312`, 7},
{-160.3407558640722`, 78.16943044857129`, 16.994318926553`, 4},
{-71.96082792272726`, 37.188492660802`, 80.48146880641232`, 4},
{-84.79725706395358`, 61.169034458703244`, 14.534120996762434`, 7},
{-146.66145247140668`, 79.08280746162269`, 92.93594856842611`, 7},
{-69.92431494337903`, 35.22063093526073`, 54.21402072248001`, 7},
{-101.23723664226463`, 74.01504669337591`, 170.0032823292999`, 4},
{-81.29452181226927`, 61.017014787682506`, 101.57448043892535`, 5},
{-97.87296918275007`, 73.61023647968544`, 124.10443071250333`, 6},
{-69.85383634719886`, 40.252000901611844`, 86.76110794918125`, 7},
{-78.86863224079116`, 60.70878585739614`, 147.6891101901244`, 7},
{-177.8370999656225`, 79.90001337481668`, 90.27160923872515`, 7},
{-92.99281574806406`, 74.65883936969074`, 171.35645787592205`, 6},
{-76.61208527338589`, 61.386674179393054`, 85.66264477550129`, 4},
{-76.35157000952745`, 62.68090147957848`, 65.88577143074157`, 7},
{-87.09196499179941`, 73.28643845224207`, 169.8510723706757`, 12},
{-74.8774257927899`, 61.217258517505655`, 75.18425173409656`, 1},
{-73.94586317200219`, 60.51240790637628`, 85.00058470608384`, 1},
{-79.40808028986301`, 68.05623427987507`, 67.06798148706001`, 6},
{-83.99884221090474`, 73.71080291205027`, 126.59702530212392`, 9},
{-75.59561076393179`, 65.91370956891748`, 101.50394117061595`, 1},
{-92.09549923329176`, 78.34991321475216`, 173.6014316618661`, 7},
{-70.70954827511625`, 60.281116677699096`, 106.84867545922484`, 8},
{-73.39069336707392`, 65.43930851833633`, 83.460225230889`, 2},
{-105.08956740567238`, 81.5556376606593 6`, 121.715886855594`, 3},
{-111.51032198999386`, 82.35260406308467`, 117.35805248034616`, 4},
{-70.43003676936272`, 62.04797308069775`, 148.10010466674112`, 14},
{-68.67615894969444`, 63.12411544827653`, 30.647522453947083`, 8},
{-91.19434679025993`, 81.64218972543549`, 85.83622293562549`, 6},
{-74.4953776359055`, 75.2866457793876`, 146.0821482963971`, 14},
{-67.10635483883986`, 63.4471149061162`, 173.03538838979506`, 5},
{-73.67383901407608`, 75.44819549806486`, 82.81931769746765`, 5},
{-61.64585640659924`, 44.27398298737632`, 166.65872556127243`, 5},
{-67.79059698528478`, 71.20682449988706`, 11.31434693505231`, 4},
```

```
{-65.1438525377298`, 72.34208249513063`, 153.6651421974114`, 6},
{-58.41433737449408`, 37.08630236979706`, 108.60881141953226`, 1},
{-59.59252552531368`, 61.17489009400840 4`, 57.69779479109526`, 10},
{-58.36637944112496`, 56.89206012241605 5`, 147.78090302359402`, 3},
{126.3144736585923`, 79.68073313496551`, 155.7196399012455`, 10},
{-54.25561367356009`, 74.48405096328476`, 131.48985305027992`, 6},
{-55.167928863635304`, 57.08580509427914`, 68.63400689180294`, 1},
{-53.774516055664215`, 67.26875752090483`, 167.87298241879415`, 3},
{-55.50979317585206`, 30.982680988256472`, 154.55336341330556`, 12},
{-36.58155048967095`, 86.0604104333611`, 124.7711341570924 3`, 3},
{-54.86375070367639`, 40.57103408964452`, 162.04550417941957`, 4},
{-54.636563830278035`, 45.067953224060105`, 73.20004011729009`, 6},
{-54.9054837039717`, 39.67160001185774`, 159.01861026368516`, 1},
{-46.691208898572654`, 79.2399296781029`, 52.55304603514898`, 2},
{-51.89769279029391`, 63.81102106711923 5`, 68.52980413499405`, 13},
{-51.634279784217874`, 62.1960271076670 6`, 39.76235574377862`, 4},
{-51.7687152693918 2`, 52.26822866774141`, 74.71777145648417`, 11},
{-49.84640878012549`, 62.919738062324505`, 40.34779982276936`, 10},
{81.86814752616986`, 85.42762428013403`, 127.16481988566053`, 14},
{-53.00137754095072`, 37.491545733554986`, 108.91690546280064`, 5},
{-51.2216118881693`, 50.532041875941566`, 78.24123900963644`, 6},
{-12.168450390419224`, 85.1347446530322`, 80.20704881741614`, 7},
{-48.331246476627825`, 62.43105515302908`, 158.69000233813645`, 9},
{-50.83893760656043`, 45.58508964943286`, 57.71220489982781`, 7},
{85.81684101418199`, 83.37156065305383`, 28.438372610031784`, 3},
{-50.85831632541075`, 40.76195951010184`, 165.86351264688125`, 2},
{-47.922443131693846`, 56.64795271042737`, 7.090870554534845`, 11},
{-6.722322038803324`, 84.03528414042512`, 34.1267038653369`, 4},
{48.75498323846038`, 83.41456522893188`, 153.32510179907138`, 13},
{-40.682936181286294`, 61.31649024902726`, 29.444223611162375`, 5},
{50.74531223173715`, 82.6860164454319`, 34.934949202842304`, 6},
{117.6396863807165`, 51.156001666276524`, 78.28074979880965`, 8},
{-37.51548069969856`, 62.03385268193416`, 83.2272053078762`, 6},
{-25.17676698095736`, 72.67872541141894`, 66.64145617962723`, 4},
{56.48812062312788`, 80.86443925597831`, 155.74307446765374`, 11},
{-35.78804498240237`, 59.7634000480924`, 47.74082159479501`, 5},
{42.61025137492201`, 80.2173231528068`, 75.43338410773727`, 1},
{-40.32820932482168`, 48.441573646601874`, 23.13487465252701`, 1},
{-31.356460622913353`, 62.59310540778855`, 41.61459441337612`, 8},
{-36.93533219323931`, 54.702577202184344`, 108.83799353222696`, 10},
{-40.64968048498507`, 43.010963207084195`, 35.23569795984417`, 5},
{43.58637818674378`, 78.6642254953048`, 139.55128658275916`, 12},
{14.214249568088459`, 76.0703748192437`, 170.20537713696953`, 2},
{105.0298243295973`, 61.434669424547714`, 133.2970074672622`, 13},
{112.01564054098883`, 52.06071516836354`, 168.30068302913097`, 4},
{-34.72165501625797`, 44.57719061182975`, 66.41259125484576`, 10},
{14.125094031193388`, 74.00638149362334`, 145.3780274293423`, 6},
```

```
{47.395937469850885`, 75.08320396040763`, 114.29278235091232`, 1},
{-27.3880669480026`, 52.73881889820267`, 127.16433769708794`, 7},
{85.51903158765735`, 68.11804810744312`, 164.7547507770336`, 1},
{-18.800705212778407`, 57.7879086721255`, 63.30580745497223`, 2},
{83.98939049267582`, 66.49314237869253`, 9.783107012312135`, 6},
{-1.4430955166325354`, 64.81636434849034`, 101.39207165681819`, 3},
{-26.56730195232397`, 44.51207364148873`, 71.47395130090219`, 3},
{-8.08996812126854`, 60.770608235802406`, 165.35548689393852`, 14},
{38.25032762532409`, 68.93692206594213`, 164.21426381765326`, 5},
{36.67316611657576`, 68.64687961221395`, 156.63040305814891`, 5},
{-12.461358572891712`, 52.95191608847454`, 47.51837993862195`, 9},
{-16.157078281265637`, 47.34258883800924`, 62.77563328563273`, 6},
{-5.506125790988126`, 52.44538359447726`, 23.029085718165017`, 5},
{14.125414437625977`, 58.65298780790525`, 133.0168776023074`, 11},
{-22.6742066453052`, 32.599634903909745`, 92.6034939601179`, 3},
{-14.670508657362499`, 41.3193311775902`, 96.50516242576394`, 9},
{109.99871580805316`, 42.19791731678867 6`, 68.2463571815832`, 12},
{32.973174786700625`, 60.50409261644156`, 49.66091368934505`, 4},
{7.680322469709102`, 53.697702255429505`, 154.21634475029228`, 9},
{11.253447593767046`, 54.5332262247396`, 37.02915003822727`, 5},
{-16.361381127206954`, 35.1663193104090 3`, 89.11152653324525`, 4},
{-6.123289541793467`, 43.00046214833899`, 110.78417970731212`, 9},
{-8.734782682147946`, 40.173160694101966`, 121.11278953091941`, 6},
{59.83163154010501`, 58.47304787758651`, 26.261400992049534`, 6},
{33.73246974483824`, 57.04748563122167`, 44.82375705926482`, 8},
{26.852177854821896`, 55.80650558528219`, 172.0887802783985`, 0},
{26.657981628487946`, 55.257364157508384`, 19.766301979532898`, 10},
{22.459973993385848`, 52.64033943387406`, 96.92831778193451`, 5},
{83.71973154014572`, 52.212139548070795`, 28.917166697897883`, 4},
{6.6141171388057245`, 43.248596842490514`, 3.71519159366028 2`, 2},
{24.172461587854873`, 48.88686146186828`, 30.206296118442253`, 1},
{76.36500156572967`, 51.50536497921368`, 171.45828767119573`, 4},
{31.30802895695041`, 50.25665768484199`, 112.49254582724932`, 3},
{33.804987760758685`, 49.56511348103394`, 89.96825314648429`, 3},
{14.191982532432682`, 42.253493804071944`, 71.61197612570408`, 7},
{35.008054081408126`, 49.081976368720525`, 155.6082381592921 7`, 12},
{18.261104539915344`, 42.90041637174238`, 135.24295566613577`, 3},
{53.49892149910834`, 49.576054169505056`, 155.75467765437224`, 10},
{22.950816999137274`, 40.866217997578296`, 16.794392172328887`, 12},
{55.07317693898785`, 46.53950553622367`, 38.83397364359449`, 11},
{44.20570265076692`, 45.09903052798443`, 157.90146275371922`, 13},
{32.7527777811809`, 41.27550660946753`, 152.57423201090427`, 6},
{61.02662393017077`, 42.43380561760152`, 2.754687355297306`, 11},
{40.553689819384616`, 39.15312597563563`, 174.16426839662438`, 4},
{28.4125358022574`, 35.05638446993925`, 59.74364450059625`, 13},
{86.60121186272173`, 40.407008783260736`, 92.08463825958192`, 9},
{63.44588164702946`, 41.004897424463465`, 9.104706945914302`, 2},
```

```
{100.67379229121102`, 36.65934908209057`, 117.7616420839314`, 3},
{85.65690797581497`, 37.9261960178107`, 148.49247070602377`, 7},
{38.35298427931641`, 32.807534537551476`, 50.85296370035088`, 3},
{41.3816892219913 8`, 33.74521830023025`, 179.51507409998968`, 2},
{45.3689079850817`, 34.60121546422472`, 155.65763708713985`, 5},
{48.09480192508495`, 31.999978515223766`, 136.65305562213507`, 12},
{50.506190337567226`, 32.15266437702678`, 79.7059482589632`, 6},
{58.04464043328841`, 32.28151189364568`, 39.821140883855115`, 6},
{79.53334136782792`, 31.820356564323497`, 77.19935596027952`, 2},
{100.51305100226183`, 30.726063614921618`, 25.963658257932856`, 1},
{43.66919435056407`, -33.04959908945096`, 28.41543377475217`, 6},
{15.774594625129097`, -43.518536887783384`, 146.7935052060096`, 3},
{-1.6107317771849623`, -44.981397437438716`, 45.42510413262782`, 9},
{57.61331699010018`, -30.29184442782771`, 122.11851259759209`, 6},
{40.46151593416346`, -40.74616219154144 4`, 128.2870649923049`, 6},
{44.52904821147003`, -39.1553378565913`, 52.67428850947174`, 14},
{46.48179251229769`, -38.59820408289387`, 104.48336378388773`, 5},
{52.30206436802545`, -36.37489508335759`, 149.97928023685972`, 1},
{53.507367112617985`, -36.033488477626754`, 169.27431088034226`, 9},
{38.492978278691794`, -43.2991659337089`, 166.01390004437266`, 10},
{46.58938435553205`, -41.39397334171703`, 40.7557155351948`, 8},
{-11.775420745679716`, -48.00233336241495`, 54.97099290376514`, 12},
{60.20451458428748`, -34.8836480504957`, 163.32906978940642`, 9},
{8.345963487764562`, -50.10476817258659`, 135.2166370110791`, 13},
{63.51492558140568`, -33.97455588008184`, 8.363728753401595`, 11},
{33.184097246654154`, -48.358593760845096`, 101.8719355482988`, 4},
{33.76996175389568`, -49.4683198751036`, 35.857876845086324`, 12},
{40.66189174961286`, -47.913057041832374`, 72.75341818037018`, 2},
{36.84199353412138`, -50.50716876076785`, 76.081704759741`, 10},
{36.6640512615494`, -51.056509951589256`, 100.37946071227037`, 7},
{32.5632975098656`, -51.93673250548431`, 119.5892191170102`, 9},
{-20.031295179351282`, -49.876143079359764`, 64.72201939145967`, 4},
{27.117742175292484`, -53.540970729656124`, 25.221680228907605`, 1},
{39.59620434710648`, -51.61046248322282`, 35.477182025165746`, 7},
{22.16946367343113`, -55.65661319627956`, 0.04329128853769253`, 8},
{-16.4822181471955`, -52.728097107733916`, 98.13072749477297`, 8},
{59.03600640446457`, -46.56733495403914`, 12.080104384535511`, 11},
{52.497124353359325`, -49.572351242167954`, 58.948800784978516`, 14},
{81.21568654042473`, -30.60820367937699`, 147.89994372820985`, 4},
{81.28859009543814`, -30.530358976689637`, 176.86286643594818`, 7},
{58.92390880636182`, -48.80291105292661`, 6.300335726283361`, 1},
{82.1468839369421`, -31.11612068427291`, 96.08596954748658`, 13},
{59.011064685185104`, -49.38133023569162`, 35.28108585249455`, 2},
{-1.4640761538786435`, -57.72419126246514`, 160.7438468610905 6`, 12},
{55.21980731203827`, -51.69109098064318`, 49.27132260616372`, 3},
{-7.604284721533364`, -56.93839760565353`, 127.08420753217436`, 3},
{69.1810151729185 2`, -45.1294746618859`, 164.83103921247928`, 14},
```

```
{77.34631186672435`, -38.617430761918435`, 158.3668794178 5212`, 1},
{-15.393019805162723`, -56.0624350546387`, 126.4920153869 0083`, 2},
{5.328690537642593`, -61.236184818490415`, 128.09450059499756`, 3},
{28.30622477529960 8`, -61.234662852724334`, 30.88993824263124`, 1},
{53.831729687823696`, -56.892517404752674`, 34.13262607082229`, 6},
{13.965403697885613`, -62.7441637565389 5`, 166.1812713472482 4`, 13},
{83.5996122927983 3`, -39.1174708309064 4`, 110.5054082878249 6`, 8},
{72.5931342469152 6`, -48.8212247753193 1`, 125.1865401402257`, 7},
{82.56795537953823`, -41.87913468597311`, 84.95653679539056`, 4},
{85.92806283522229`, -38.186056904507396`, 177.79375132123693`, 3},
{91.49066505866143`, -30.90315607318379`, 143.6314275608002`, 14},
{72.76998648085286`, -50.161541928036826`, 172.03374271622`, 9},
{57.5910640260935 4`, -58.5023319104317`, 5.470056909395044`, 8},
{75.49578766988297`, -49.32489237976959`, 42.933676556512296`, 13},
{24.464869087582382`, -64.8347093353817`, 17.02471115789224`, 6},
{94.13660980988917`, -31.03699455030239`, 168.32220432489396`, 7},
{80.9517344336499`, -47.61330234309482`, 19.71931212830302`, 2},
{25.51642387392157`, -66.34685423808361`, 93.89003531186252`, 7},
{86.16991374306608`, -45.47475215232268`, 137.68884268134022`, 4},
{79.67825266836365`, -55.28678062798501`, 21.52869544240123`, 5},
{77.4642188252492 6`, -58.024105691610316`, 129.40569035213582`, 1},
{-23.989778858896237`, -64.04414289959661`, 57.05548333733941`, 8},
{-27.850378776372096`, -62.251587824679035`, 121.50050187382787`, 2},
{84.06774916946604`, -58.370850255218784`, 125.92793736967093`, 1},
{73.3901210484218`, -65.11319120741547`, 23.38116283465095`, 10},
{84.19309572356345`, -59.61884435527171`, 143.84928618892076`, 2},
{85.29735527682382`, -61.05734358104395`, 162.95833767296403`, 8},
{43.47541250898864`, -74.17697214757003`, 14.171068869776063`, 9},
{98.39689471069248`, -47.382298241754654`, 149.2025421853356`, 6},
{102.21281256371392`, -41.568120444925206`, 107.39386028913763`, 10},
{65.56127214387064`, -71.74484060844905`, 121.37195890846243`, 8},
{91.25267597627146`, -58.409229169073946`, 161.8493518393685`, 10},
{-5.255302768013721`, -73.86694199986435`, 148.85640635132486`, 12},
{79.22485190580042`, -67.6781329167402`, 144.65853132995278`, 8},
{91.89839031103409`, -59.9358208713557`, 170.3017988051909`, 7},
{93.90010439787434`, -58.43525410143111`, 52.58290856841974`, 5},
{71.75626788715134`, -71.96802814973357`, 58.89917880177063`, 12},
{108.33678263277267`, -32.70448416419367`, 90.78819838904182`, 11},
{93.87112388646025`, -60.20224163281756`, 41.60783533054826`, 5},
{80.91099443109735`, -69.68159290227975`, 142.42448457906698`, 6},
{95.16320309324419`, -59.40211160000764`, 98.49236898999496`, 14},
{-39.666365139746375`, -62.149358129601836`, 99.66338954306339`, 4},
{95.41241600177047`, -60.33717642563692`, 2.278566453938273`, 7},
{80.4507825653529 1`, -71.07912659131706`, 9.01779388387706`, 1},
{98.44104343792776`, -57.95369999164699`, 66.67087010287912`, 7}};
```

In[33]:=
```
nSrc = Length[data];(* nSrc - the number of sources (355)*)
Print["The number of sources is ", nSrc, " sources (= number of records in 'data')."]
```

The number of sources is 355 sources (= number of records in 'data').

In[35]:=
```
(*Give names to the data. *)
ℓi[i_] := data⟦i, 1⟧ (*Galactic Longitude of i^th source*)
bi[i_] := data⟦i, 2⟧ (*Galactic Latitude of i^th source*)
ψi[i_] := data⟦i, 3⟧ (*Position angle for i^th source*)
σψi[i_] := data⟦i, 4⟧ (*Uncertainty of ψ for i^th source*)
```

In[39]:=
```
Print["The largest PPA uncertainty σψ is ", Sort[Table[σψi[i], {i, nSrc}]]⟦-1⟧, "°."]
```

The largest PPA uncertainty σψ is 14°.

In[40]:=
```
ri[i_] := ri[i] = er[ℓi[i], bi[i]]
(*unit vector from Origin to ith Source on Celestial Sphere*)
vNi[i_] := (*vNi[i]=*)eN[ℓi[i], bi[i]] (*North at i^th source *)
vEi[i_] := (*vEi[i]=*)eE[ℓi[i], bi[i]] (*East at i^th source*)
vψi[i_] := (*vψi[i]=*)
 cos[ψi[i]] × vNi[i] + sin[ψi[i]] × vEi[i] (*unit vector in direction of PPA*)
```

In[44]:=
```
Print["All vψ have a positive component along Local East: ",
 {1} == Union[Table[Sign[vψi[i].vEi[i]], {i, nSrc}]] ]
```

All v_{ψ} have a positive component along Local East: True

In[45]:=
```
lpPPAψGAL = ListPlot[Sort[Table[ψi[i], {i, nSrc}]], PlotRange → All,
   FrameLabel → {"Count", "ψ"}, PlotLabel → "PPA ψ in Galactic coords, Sorted",
   PlotTheme → {"Scientific", "Detailed", "Classic"}];
```

Out[46]=

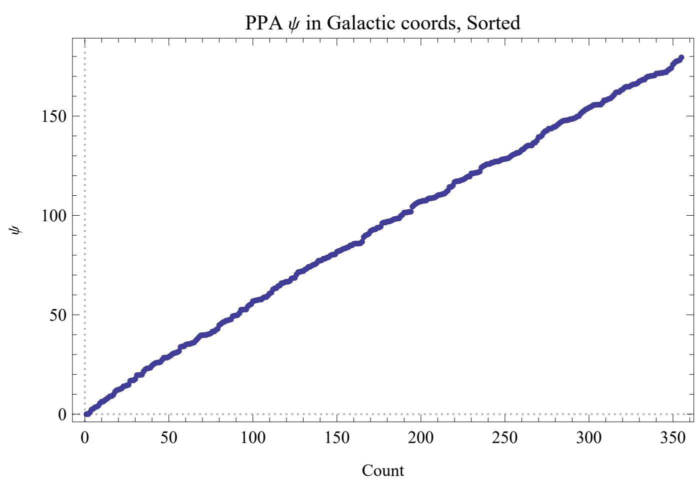


Figure A2.1: Polarization position angles ψ in Galactic coordinates.
None should be negative. The PPA ψ is measured from North toward East.

Section A2a. Northern and Southern hemisphere cohorts

We want to find the number of QSOs in the North Galactic Hemisphere and in the South Galactic Hemisphere.

Definitions:

lpPPAψGAL list plot of the ψs of the sources
lpGALb list plot of Galactic latitudes of the sources
idsForNorthGalQSOs list of sources in the Northern hemisphere
rAVEnorth355 unit radial vector to average location of the Northern cohort
ρRMSnorth355 RMS radius of the northern sources
idsForSouthGalQSOs list of sources in the Southern hemisphere
rAVEsouth355 unit radial vector to average location of the Southern cohort
ρRMSsouth355
ρRMSeffective355 effective rms radius combining northern & southern cohorts
lpGALsources plot of source locations

In[48]:=
```
lpGALb = ListPlot[Sort[Table[bi[i], {i, nSrc}]],
   FrameLabel → {"Count", "Gal. Lat. b, deg."}, PlotRange → {-90, 90},
   PlotLabel → "Galactic Latitudes", PlotTheme → {"Scientific", "Detailed", "Classic"}];
Table[{j, Sort[Table[bi[i], {i, nSrc}]][[j]]}, {j, 155, 165}];
```

In[50]:=
```
lpGALb
Print["Figure A2.2: There are no QSOs within 30° of the Galactic
   Equator, 195 in the north and 160 in the south galactic hemisphere."]
```

Out[50]=

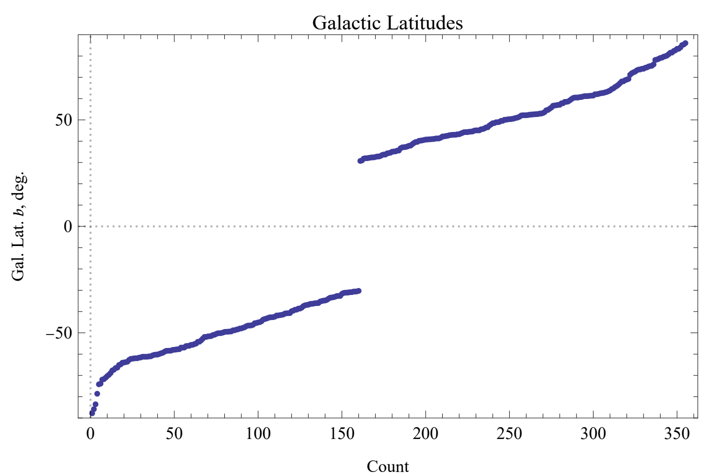


Figure A2.2: There are no QSOs within 30° of the Galactic
  Equator, 195 in the north and 160 in the south galactic hemisphere.

In[52]:=
```
idsForNorthGalQSOs = {};
idsForSouthGalQSOs = {};
For[i = 1, i ≤ nSrc, i++,
 If[bi[i] < 0., AppendTo[idsForSouthGalQSOs, i], AppendTo[idsForNorthGalQSOs, i]]]
Length[idsForNorthGalQSOs];
Length[idsForSouthGalQSOs];
```

In[57]:=
```
rAVEnorth0 = Sum[ri[ i], {i, idsForNorthGalQSOs}] / Length[idsForNorthGalQSOs];
rAVEnorth355 = rAVEnorth0 / (rAVEnorth0.rAVEnorth0)^(1/2) ;
(*unit radial vector to average location of the catalogued sources *)
ℓAVEnorth355 = ℓFROMr[rAVEnorth355]; (*longitudes, north cohort*)
bAVEnorth355 = bFROMr[rAVEnorth355]; (*latitudes, south cohort*)
```

In[61]:=
```
rAVEsouth0 = Sum[ri[ i], {i, idsForSouthGalQSOs}] / Length[idsForSouthGalQSOs];
rAVEsouth355 = rAVEsouth0 / (rAVEsouth0.rAVEsouth0)^(1/2) ;
(*unit radial vector to average location of the catalogued sources *)
ℓAVEsouth355 = ℓFROMr[rAVEsouth355];
bAVEsouth355 = bFROMr[rAVEsouth355];
```

In[65]:=
```
lpGALsources = ListPlot[{Table[{ℓi[j], bi[j]}, {j, idsForNorthGalQSOs}],
     Table[{ℓi[j], bi[j]}, {j, idsForSouthGalQSOs}]},
   PlotRange → {{-180., 180.}, {-90., 90.}}, PlotStyle → {Black, Purple},
   FrameLabel → {" ℓ (NOT REVERSED)", " b"},
   PlotLabel → "Sources (Galactic Coords. J2000.0)",
   PlotTheme → {"Scientific", "Detailed", "Classic"}];
```

Out[66]=

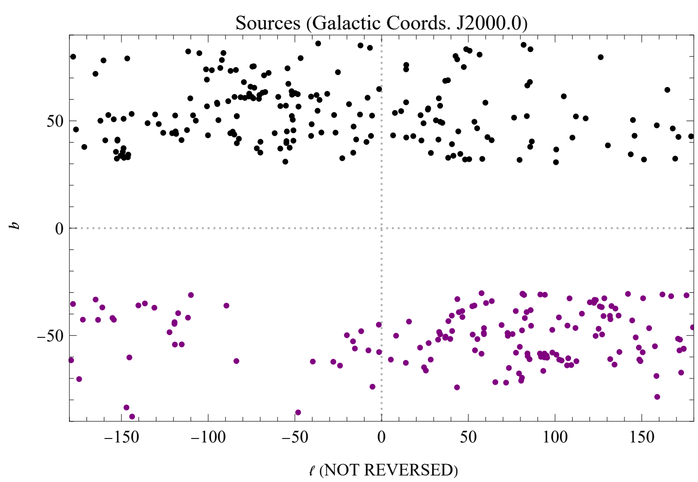


Figure A2.3: The locations of the sources on the Celestial Sphere.

In[68]:=
```
(*We can calculate the RMS radius of the sources in the North Galactic Hemisphere*)
ρSrcToAVEnorthi[i_] := arccos[ Abs[ ri[ i ].rAVEnorth355 ]]
ρRMSnorth355 =
  (1/Length[idsForNorthGalQSOs] Sum[ρSrcToAVEnorthi[i]^2, {i, idsForNorthGalQSOs}])^(1/2) ;
```

In[70]:=
```
(*We can calculate the RMS radius of the sources in the South Galactic Hemisphere*)
ρSrcToAVEsouthi[i_] := arccos[ Abs[ ri[ i ].rAVEsouth355 ]]
ρRMSsouth355 =
  (1/Length[idsForSouthGalQSOs] Sum[ρSrcToAVEsouthi[i]^2, {i, idsForSouthGalQSOs}])^(1/2) ;
```

If we add the squares of the RMS radii and multiply by π, then that is the effective area of a circular region with an area equal to the combined area of the North and South collections of QSOs. To get the RMS radius of that combined area, we divide by π and take the square root. One sees that ρRMSeffective = (ρRMSnorth355^2 + ρRMSsouth355^2)^1/2

```
In[72]:= ρRMSeffective355 = ((ρRMSnorth355^2 + ρRMSsouth355^2))^(1/2);
```

```
The rms radius of the sources in the Northern Galactic Hemisphere is 37.9539°, North.

The rms radius of the sources in the Southern Galactic Hemisphere is 36.9514°, South.

The effective rms radius of the sources, combining hemispheres, is 52.9708°, combined.

After Sec. A2, time and memory used are 3.969 seconds and 158 266 720 bytes.
```

A3. Derivation of a formula for the alignment angle η_{iH}.

The formula depends on the location on the sky $(\ell, b)_S$ of the ith source , on the location on the sky $(\ell, b)_H$ of point H , and on the polarization direction ψ for the ith source. Here, ℓ is longitude and b is latitude on the 2D Celestial sphere with unit radius.

When the source S and the point H either coincide on the sphere, or are antipodal, the angle η_{iH} is ill-defined. To avoid this problem, the angle η_{iH} is set to 45° whenever ±S
is "close" to ±H. One of the four arcs from ±S to ±H must be less than or equal to 0.001 radian = 0.0573° for ±S to be considered "close" to ±H.

Note. The derivation is copied from my notebook '20211030ReplaceClump1PaperFirstCopy.nb'. It is modified by replacing Cos, Sin, ... Mathematica functions with the corresponding functions for angles in degrees, which are defined above and have lower case first letters, cos, sin, ... That puts numerical conversion factors in some of Mathematica's outputs.

Definitions

vH = $\frac{\mathrm{rH} - (\mathrm{rH.rS})\,\mathrm{rS}}{[(\mathrm{rH} - (\mathrm{rH.rS})\,\mathrm{rS}).(\mathrm{rH} - (\mathrm{rH.rS})\,\mathrm{rS})]^{1/2}}$: unit vector in the 2D tangent plane at S, in the direction of H from S, vH.rS = 0, where er[ℓH,bH].er[ℓS,bS] = rH.rS is the inner product of the radial unit vectors rH and rS to point H and source S

Since vψ is also perpendicular to rS, it follows that vψ.rS = 0, and we have $\frac{\mathrm{rH}}{[(\mathrm{rH} - (\mathrm{rH.rS})\,\mathrm{rS}).(\mathrm{rH} - (\mathrm{rH.rS})\,\mathrm{rS})]^{1/2}}$ as the part of vH that contributes to the dot product cosη = vψ.vH . Therefore, define

vHperpS = $\frac{\mathrm{rH}}{[(\mathrm{rH} - (\mathrm{rH.rS})\,\mathrm{rS}).(\mathrm{rH} - (\mathrm{rH.rS})\,\mathrm{rS})]^{1/2}}$

denoSquared1 = deno of vHperpS^2
vψ unit vector in the tangent space to source S in the direction of polarization ψ

ηiH The alignment angle η is the acute angle between vψ and vH in the 2D tangent plane at S_i.
ηBarAtHwithAnyψ(ℓH, bH, ψ) formula for $\overline{\eta}(H)$, since we know the locations (ℓS, bS) for all the sources

```
In[80]:= denoSquared1 = FullSimplify[(er[ℓH, bH] - (er[ℓH, bH].er[ℓS, bS]) er[ℓS, bS]).
            (er[ℓH, bH] - (er[ℓH, bH].er[ℓS, bS]) er[ℓS, bS])];

(* denoSquared = [rH - (rH.rS) rS].[rH - (rH.rS) rS] =
     rH.rH - 2(rH.rS)^2 + (rH.rS)^2 rS.rS =
      1 - 2(rH.rS)^2 + (rH.rS)^2 = 1 - (rH.rS)^2 *)
```

In[81]:=
```
Print["Check some algebra, the formula for the denominator of vH⊥rS: ",
 0 == FullSimplify[denoSquared1 - (1 - (er[ℓH, bH].er[ℓS, bS])^2)]] (*check*)
```

```
Check some algebra, the formula for the denominator of vH⊥rS: True
```

Write the formula for the vector vHperpS, with a denominator of $(1 - (rH.rS)^2)^{1/2}$:

Note: The formula blows up when H = S.

In[82]:=
```
vHperpS[ℓS_, bS_, ℓH_, bH_] := (*vHperpS[ℓS,bS,ℓH,bH]=*)
 er[ℓH, bH] / (1 - (er[ℓH, bH].er[ℓS, bS])^2)^(1/2)
```

In[83]:=
```
(*This, vH⊥rS, blows up when H = S:*)
(*Simplify[vHperpS[ℓH,bH,ℓH,bH] ];*)
(* BANG, BOOM!! when H = S. *)
```

The other vector we need is vψ, the unit vector in the 2D tangent plane at S pointing in the direction of the polarization position angle ψ. By Fig. A23.1, one sees that

$$v\psi = \cos(\psi)\, N + \sin(\psi)\, E ,$$

where N and E are local north and east unit vectors in the 2D tangent plane at S.

In[84]:=
```
vψ[ℓS_, bS_, ℓH_, bH_, ψ_] := (*vψ[ℓS,bS,ℓH,bH,ψ]= *)
 cos[ψ] × eN[ℓS, bS] + sin[ψ] × eE[ℓS, bS]
(*vψ[ℓS,bS,ℓH,bH,ψ]*)
```

The alignment angle η is the acute angle between vψ and vH in the 2D tangent plane at S,

In[85]:=
```
ηiH0[ℓS_, bS_, ℓH_, bH_, ψ_] := (*ηiH0[ℓS,bS,ℓH,bH,ψ]=*)
 arccos[ Abs[vψ[ℓS, bS, ℓH, bH, ψ].vHperpS[ℓS, bS, ℓH, bH] ] ]
(*ηiH0[ℓS,bS,ℓH,bH,ψ]*)
FullSimplify[ηiH0[ℓS, bS, ℓH, bH, ψ]]
```

Out[86]=
```
57.2958 ArcCos[Abs[ (Cos[0.0174533 bS] Cos[0.0174533 ψ] Sin[0.0174533 bH] +
     Cos[0.0174533 bH] (-Cos[0.0174533 ℓH - 0.0174533 ℓS] Cos[0.0174533 ψ] Sin[0.0174533 bS] +
        Sin[0.0174533 ℓH - 0.0174533 ℓS] Sin[0.0174533 ψ])) /
   (√(1 - (Cos[0.0174533 bH] Cos[0.0174533 bS] Cos[0.0174533 ℓH - 0.0174533 ℓS] +
         Sin[0.0174533 bH] Sin[0.0174533 bS])^2))]]
```

Mathematica likes its own trig functions for angles in radians. Note the conversion factor appearing in all the angles. The factor converts degrees to radians, 1° = 0.0174533 radians and radians to degrees, 1 radian = 57.2958° , approximately.

In[87]:=
```
(*The following function is well-
 behaved everywhere except where ±H coincides with ±S.*)
ηiHwithIndeterminate[ℓS_, bS_, ℓH_, bH_, ψ_] :=
 (*ηiHwithIndeterminate[ℓS,bS,ℓH,bH,ψ]=*) 57.29577951308232`
  ArcCos[Min[1., Abs[(Cos[0.017453292519943295` bS] Cos[0.017453292519943295` ψ]
          Sin[0.017453292519943295` bH] + Cos[0.017453292519943295` bH]
          (-Cos[0.017453292519943295` ℓH - 0.017453292519943295` ℓS]
             Cos[0.017453292519943295` ψ] Sin[0.017453292519943295` bS] +
            Sin[0.017453292519943295` ℓH - 0.017453292519943295` ℓS]
             Sin[0.017453292519943295` ψ])) /
       (√(1 - (Cos[0.017453292519943295` ℓH - 0.017453292519943295` ℓS] Cos[
                 0.017453292519943295` bH] Cos[0.017453292519943295` bS] +
                Sin[0.017453292519943295` bH] Sin[0.017453292519943295` bS])^2))]]]
```

In[88]:=
```
(*Since η is an acute angle, let us take the middle value,
η = 45° in the neighborhood where H ≈ S.*)
ηiH[ℓS_, bS_, ℓH_, bH_, ψ_] :=
 ηiHwithIndeterminate[ℓS, bS, ℓH, bH, ψ] /; ((1 - (er[ℓH, bH].er[ℓS, bS])^2) ≥ 0.000001)
ηiH[ℓS_, bS_, ℓH_, bH_, ψ_] := 45. /; ((1 - (er[ℓH, bH].er[ℓS, bS])^2) < 0.000001)
Print["We choose ηiH = 45° wherever ±H
   is 'close' to ±S, with 'close' meaning within an angle of ",
 ArcSin[0.000001^(1/2)], " radians = ", arcsin[0.000001^(1/2)], "°."]
```

We choose η_{iH} = 45° wherever ±H is 'close' to ±S, with 'close' meaning within an angle of 0.001 radians = 0.0572958°.

In[91]:=
```
ηBarAtHwithAnyψ[ℓH_?NumberQ, bH_?NumberQ, ψ_] := (*ηBarAtHwithAnyψ[ℓH,bH,ψ]=*)
 1/nSrc Sum[ηiH[ℓi[i], bi[i], ℓH, bH, ψ[[i]] ], {i, nSrc}] /; VectorQ[ψ, NumberQ]
```

After Sec. A3, time and memory used are 84.468 seconds and 173 516 104 bytes.

A4. Build a Grid

To avoid programming issues that appeared when trying to find the continuous alignment function $\bar{\eta}(H)$, we instead find $\bar{\eta}$(H) at a finite number of grid points. Once the grid points nearest the extremes of $\bar{\eta}(H)$ are found, the continuous function is used to determine the function’s extreme values $\bar{\eta}$min and $\bar{\eta}$max and the hubs $\pm H_{\text{align}}$ and $\pm H_{\text{avoid}}$, which are the points where those extreme values occur.

The grid is meant to have equally spaced grid points. The decreasing radii of parallels approaching the North Pole is accounted for. However, any pairs of grid points that turn out to be closer together than planned add points to the evaluation of $\bar{\eta}(H)$ and are not a problem.

We calculate all the alignment angles η_{ij}, where η_{ij} is the alignment angle η between the polarization direction ψ_i for the i^{th} source S^i and the direction from S^i to the j^{th} grid point H_j. There are 355 sources and 10518 grid points, so there are 355(10,518) = 3,733,890 alignment angles η_{ij}.

Definitions:

gridSpacing   grid spacing in degrees

$d\theta 1$ = gridSpacing   separation in degrees between grid points on a constant latitude circle and separation of constant latitude circles. Vestigial duplicity noted.
gridN, gridS, grid North, South hemisphere, combined grid point tables, record items listed below
nGrid number of grid points
rHj(j) unit radial vector to jth grid point H_j
ℓHj[j], *b*Hj[j] longitude and latitude of the j^{th} grid point in degrees
vHij(i,j) unit vector tangent to the great circle connecting the ith source with Hj in tangent space of the ith source
ηiHj(i,j) alignment angle η between the PPA direction ψ and the great circle toward j^{th} grid point Hj in the tangent space at the ith source.
showSphereOfGridPoints 3D view of the grid points

gridN and ***gridS*** and ***grid***
1. sequential point # 2. ℓ index 3. *b* index 4. ℓ (range: -180° to +180°) 5. *b* (range: -90° - +90°) 6. Cartesian coordinates of the point

Let's get the grid. With "gridSpacing" = 2°, it is a 2°x2° grid.

In[96]:=
```
gridSpacing = 2 (*grid spacing in degrees*);
dθ1 = gridSpacing ;(* grid Spacing in degrees*)
```

In[98]:=
```
(*KEEP this cell - DO NOT DELETE*)
gridN = {{1, 0, 0, 90., 90., {0., 0., 1.}}}; idN = 2;
For[bj = 0., bj < 90./dθ1, bj++, bpointH = bj dθ1;
 For[ ai = 0., ai < Ceiling[360./dθ1 (cos[bpointH] + 0.01)],
  ai++, ℓpointH = ai dθ1 / (cos[bpointH] + 0.01);
  AppendTo[gridN, {idN, ai, bj, ℓpointH, bpointH, er[ℓpointH, bpointH]}];
  idN = idN + 1
 ]]
```

In[100]:=
```
(*KEEP this cell - DO NOT DELETE*)
gridS = {{1, 0, 0, 90., -90., {0., 0., -1.}}}; idN = 2;
For[bj = 1., bj < 90./dθ1, bj++, bpointH = -bj dθ1;
 For[ ai = 0., ai < Ceiling[360./dθ1 (cos[bpointH] + 0.01)],
  ai++, ℓpointH = ai dθ1 / (cos[bpointH] + 0.01);
  AppendTo[gridS, {idN, ai, bj, ℓpointH, bpointH, er[ℓpointH, bpointH]}];
  idN = idN + 1
 ]]
```

In[102]:=
```
(*KEEP this cell - DO NOT DELETE*)
grid = {}; j = 1;
For[jN = 1, jN ≤ Length[gridN], jN++,
 AppendTo[grid, {j, gridN⟦jN, 2⟧, gridN⟦jN, 3⟧, gridN⟦jN, 4⟧, gridN⟦jN, 5⟧, gridN⟦jN, 6⟧}];
 j = j + 1]
For[jS = 1, jS ≤ Length[gridS], jS++,
 AppendTo[grid, {j, gridS⟦jS, 2⟧, gridS⟦jS, 3⟧, gridS⟦jS, 4⟧, gridS⟦jS, 5⟧, gridS⟦jS, 6⟧}];
 j = j + 1]
```

In[105]:=
```
nGrid = Length[grid];
rHj[j_] := rHj[j] = grid⟦j, 6⟧ (*unit radial vector to jth grid point H*)
ℓHj[j_] := ℓHj[j] = ℓFROMr[rHj[j]]
bHj[j_] := bHj[j] = bFROMr[rHj[j]]
```

In[109]:=
```
(* iᵗʰ Source and jᵗʰ grid point*)
(*vHij: unit vector tangent to the great circle connecting
  the ith source with Hj in tangent space of the ith source*)
 (*ηiHj: alignment angle between the PPA direction ψ and the great circle
  toward Hj in the tangent space at the ith source. See Fig. A23.1.*)
Clear[vHij, ηiHj]
vHij[i_, j_] := vHij[i, j] = (rHj[j] - (rHj[j].ri[i]) ri[i]) /
   (√((rHj[j] - (rHj[j].ri[i]) ri[i]).(rHj[j] - (rHj[j].ri[i]) ri[i])))
Clear[ηiHj]
ηiHj[i_, j_] := ηiHj[i, j] = arccos[Abs[vψi[i].vHij[i, j]]]
```

In[113]:=
```
showSphereOfGridPoints =
  Show[{Graphics3D[{Sphere[{0, 0, 0}], Thick, Line[{{0, 0, -1.2}, {0, 0, 1.2}}],
      Text[Style["N", Bold], {0, 0, 1.25}]}, Boxed → False],
    ListPointPlot3D[Table[rHj[j], {j, nGrid}], PlotStyle → {PointSize[0.007]},
     PlotTheme → {"Scientific", "Detailed", "Classic"}]}, ImageSize → 72 × 4];
```

Out[114]=

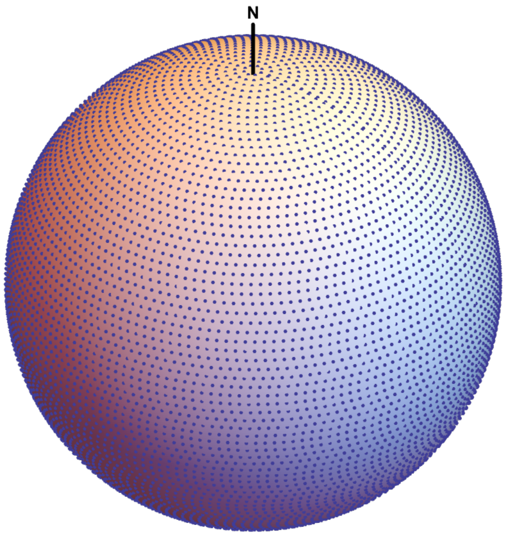


Figure A4.1. The grid. There are 10 520 grid points.

In[116]:= **MemoryInUse[]**

Out[116]= 186 243 944

After Sec. A4, time and memory used are 85.109 seconds and 186 250 208 bytes.

A5. Evaluate the alignment function $\overline{\eta}$(H), Eq. (1)

The alignment function $\overline{\eta}$(H) is evaluated first on the grid and then more carefully evaluated at the alignment and avoidance hubs. The locations of the hubs and the extreme values of the function $\overline{\eta}$(H) imply the results of the hub tests.

The quantities found in this section reflect the "best" values of the polarization direction ψ data, the values of the directions presented in the catalog. There are no error bars or uncertainties on the quantities calculated here. These are their "best" values.

In a section or sections below, the polarization directions will be chosen from a distribution of directions for each QSO source based on the uncertainty $\psi \pm \sigma\psi$ values from the catalog. Those results include the experimental uncertainties arising from the uncertainties $\sigma\psi$ in the directions ψ. But here, the "best" results are found, no error bars.

Definitions

etaij[[i, j]] = η_{ij}, the angle between the transverse direction ψ_i at the i^{th} source S^i with the j^{th} grid point H_j

ηBarHj[j] average alignment angle $\overline{\eta}$(H) at grid point Hj, averaged over all sources

lpSortEtaBarAtHj graph of the average alignment angles $\overline{\eta}$(Hj)

ηBarHjAlign 1. smallest average alignment angle $\overline{\eta}$(Hj) 2. grid point ID# j where min $\overline{\eta}$(Hj) occurs

jHalign, ℓHjAlign, bHjAlign j, longitude, latitude for grid point near alignment hub Halign

ηBarHjAvoid 1. largest average avoidance angle $\overline{\eta}$(Hj) 2. grid point ID# j where max $\overline{\eta}$(Hj) occurs

jHavoid, ℓHjAvoid, bHjAvoid j, longitude, latitude for grid point near avoidance hub Havoid

ψBestTable The 'best' values of ψ are the ones recorded in the catalog

findBestAlign starting at grid point j with min $\overline{\eta}$(Hj), use formulas to get minimum between grid points

ηBarHbestAlign, ℓHbestAlign, bHbestAlign minimum of alignment function $\overline{\eta}$(H) in degrees, longitude, latitude at Halign, degrees

findBestAvoid starting at grid point j with max $\overline{\eta}$(Hj), use formulas to get maximum $\overline{\eta}$(H) between grid points
ηBarHbestAvoid, ℓHbestAvoid, bHbestAvoid maximum of avoidance function $\overline{\eta}$(H) in degrees, longitude, latitude at Havoid, degrees
crossHxH unit vector perpendicular to Halign and Havoid radii, best value
ℓbestcrossHxH ,$b$bestcrossHxH coordinates of unit cross product of radial vectors to Halign and Havoid
bestθHToH angle between radial directions to Halign and Havoid

θNGPtoHalign, θNGPtoHavoid, θNGPtoHxH angles from NGP to hubs and HxH
ℓjbjηBarHjTable For each grid point Hj: 1. ℓ 2. b 3. alignment angle $\overline{\eta}$(H) at grid point j (all in degrees)
ηBarHjSmooth average alignment angle $\overline{\eta}$(ℓ,b), by interpolation of ℓjbjηBarHjTable

In[121]:=
```
(*Use the following cell to generate the table etaij = ηij, the angle between
 the tansverse direction ψi at the ith source Si with the jth grid point Hj. *)
etaij = Table[ηiHj[i, j], {i, nSrc}, {j, nGrid}];
```

In[122]:=
```
(*ηBarHkj: average alignment angle at Hj for the catalogued sources.*)
ηBarHj[j_] := ηBarHj[j] = Total[Table[ηiHj[i, j], {i, nSrc}]] / nSrc
```

In[123]:=
```
lpSortEtaBarAtHj =
  ListPlot[Sort[Table[ηBarHj[j], {j, nGrid}]], FrameLabel → {"Count", "η̄, deg."},
   PlotLabel → "Average alignment angle η̄(Hj) at grid points Hj, Sorted ",
   ImageSize → Medium, PlotTheme → "Detailed"];
```

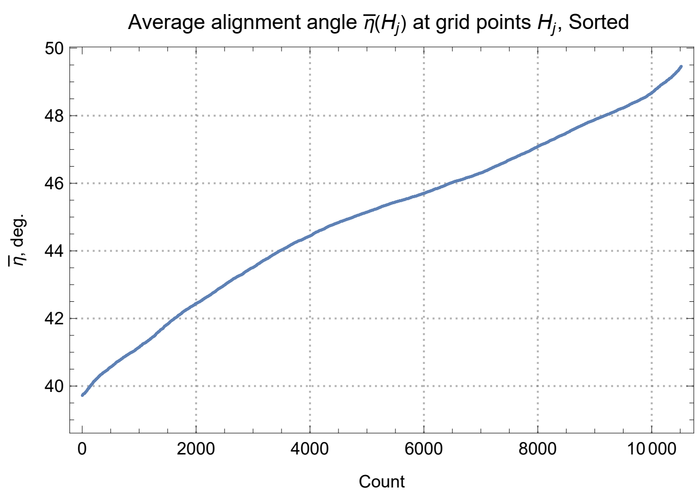
Out[124]=


Figure A5.1 The function $\overline{\eta}(H_j)$ evaluated at the 10 520 grid points H_j. The range of $\overline{\eta}(H_j)$ runs from 39.72° to 49.46°.

In[126]:=
```
ηBarHjAlign = Sort[Table[{ηBarHj[j], j}, {j, nGrid}]][[1]];
jHalign = ηBarHjAlign[[2]];
ℓHjAlign = ℓHj[ηBarHjAlign[[2]]];
bHjAlign = bHj[ηBarHjAlign[[2]]];
rHjAlign = er[ℓHjAlign, bHjAlign];
ηBarHjAvoid = Sort[Table[{ηBarHj[j], j}, {j, nGrid}]][[-1]];
jHavoid = ηBarHjAvoid[[2]];
ℓHjAvoid = ℓHj[ηBarHjAvoid[[2]]];
bHjAvoid = bHj[ηBarHjAvoid[[2]]];
rHjAvoid = er[ℓHjAvoid, bHjAvoid];
```

In[136]:=
```
ψBestTable = Table[ψi[i], {i, nSrc}];
(*The 'best' values of ψ are the ones listed in the catalog
 converted to Galactic coords, no plus/minus, no error bars. *)
findBestAlign =
  FindMinimum[ηBarAtHwithAnyψ[ℓH, bH, ψBestTable], {{ℓH, ℓHjAlign}, {bH, bHjAlign}}];
ηBarHbestAlign = findBestAlign[[1]];
ℓHbestAlign = (ℓH /. findBestAlign[[2, 1]]) - 180.;
bHbestAlign = -bH /. findBestAlign[[2, 2]];
rHbestAlign = er[ℓHbestAlign , bHbestAlign];
```

FindMinimum: The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances.

In[142]:=
```
findBestAvoid = FindMaximum[ηBarAtHwithAnyψ[ℓH, bH, ψBestTable],
   {{ℓH, ℓHjAvoid - 0.01}, {bH, bHjAvoid - 0.01}}];
ηBarHbestAvoid = findBestAvoid[[1]];
ℓHbestAvoid = (ℓH /. findBestAvoid[[2, 1]]) - 180.;
bHbestAvoid = -bH /. findBestAvoid[[2, 2]];
rHbestAvoid = er[ℓHbestAvoid , bHbestAvoid];
```

FindMaximum: The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient increase in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances.

In[147]:=
```
crossHxH0 = Cross[rHbestAlign, rHbestAvoid];
crossHxH = crossHxH0 / (crossHxH0.crossHxH0)^(1/2);
crossHxH.crossHxH;
{ℓbestcrossHxH , bbestcrossHxH } = {ℓFROMr[crossHxH ], bFROMr[crossHxH ]};
arccos[crossHxH.{0, 0, 1.}];
```

In[152]:=
```
bestΘHToH = arccos[rHbestAlign.rHbestAvoid];
```

The best alignment of the polarization directions with a point on the sky occurs at two hubs H_{align} and $-H_{align}$ with hub H_{align} located at $\{\ell, b\}$ = {-99.6, 15.2}, in degrees.

At H_{align}, the average alignment angle $\overline{\eta}_{align}$ has the value $\overline{\eta}_{align}$ = 39.71°.

The best avoidance of the polarization directions with a point on the sky occurs at two hubs H_{avoid} and $-H_{avoid}$ with hub H_{avoid} located at $\{\ell, b\}$ = {-12.3, -23.7}, in degrees.
At H_{avoid}, the average avoidance angle $\bar{\eta}_{avoid}$ has the value $\bar{\eta}_{avoid}$ = 49.47°.

The normal HxH to the plane of H_{align} and H_{avoid} is in the direction of $\{\ell, b\}$ = {21.4, 62.2}, in degrees.
The angle from the normal HxH to the North Galactic Pole is 27.82°.

In[161]:=
```
(*The following table ℓjbjηBarHjTable is created to be interpolated below,
yielding a smooth function ηBarHjSmooth of the alignment angle η̄(H) over the sphere.*)
(* Table ℓjbjηBarHjTable
 entries: 1.  ℓ  2. b  3. alignment angle ηBarHj at grid point j (all in degrees)*)
ℓjbjηBarHjTable = {};
For[j = 1, j ≤ nGrid, j++, AppendTo[ ℓjbjηBarHjTable,  {ℓHj[j], bHj[j], ηBarHj[j]}]  ;
  If[180. ≥ ℓHj[j] > 174., AppendTo[ ℓjbjηBarHjTable,  {ℓHj[j] - 360., bHj[j], ηBarHj[j]}]  ] ;
  If[ -174. > ℓHj[j] ≥ -180., AppendTo[ ℓjbjηBarHjTable,
    {ℓHj[j] + 360., bHj[j], ηBarHj[j]}]  ]       ];
```

In[163]:=
```
ℓjbjηBarHjTable[[1]];
ℓjbjηBarHjTable[[5000]];
ℓjbjηBarHjTable[[-1]];
```

In[166]:=
```
ηBarHjSmooth = Interpolation[ℓjbjηBarHjTable] (*The smooth alignment angle η̄(H).*)
```

Interpolation: Interpolation on unstructured grids is currently only supported for InterpolationOrder->1 or InterpolationOrder->All. Order will be reduced to 1.

Out[166]=
```
InterpolatingFunction[ Domain: {{-186., 186.}, {-90., 90.}} Output: scalar ]
```

After Sec. A5, time and memory used are 477.454 seconds and 1821673536 bytes.

A6. Uncertainty Runs

Creating and Storing Uncertainty Runs

For each “uncertainty run”, the polarization direction ψ for the i^{th} source is allowed to differ from the best value ψ_i by an amount $\delta\psi_i$ chosen according to a normal Gaussian distribution with mean (best) value ψ_i and half-width $\sigma\psi$, $\psi = \psi_i + \delta\psi$. The values ψ_i and $\sigma\psi_i$ are provided in the data catalog.

Given a set of values of ψ chosen as just described, one can assume that the extremes of $\bar{\eta}$(H) will not wander far from their best values found with the best ψ. We find the new extremes for the new set of ψs with the formula ηiH from Sec. 3. We get the locations of the alignment and avoidance hubs H_{align} and H_{avoid}, as well as the minimum and maximum values of the average alignment angle function $\bar{\eta}$(H).

By repeating with many sets of ψ, distributions of the calculated quantities are built up. One finds the most likely value of a quantity, the mean value. The experimental uncertainty of the quantity, denoted denoted with a “σ”, is the standard deviation of the distribution for that quantity. Or the distribution is fit by a Gaussian and the most likely value and experimental uncertainty are determined from the Gaussian fit.

Definitions:

runDataU 1. Run # 2. {$\overline{\eta}_{min}$, {ℓ,b} at grid H_{min}} 3. {$\overline{\eta}_{max}$, {ℓ,b} at grid H_{max}}
Length[runDataU] number of uncertainty runs

ηBarMinrunDataU, ηBarMaxrunDataU table of min, max $\overline{\eta}$(H)
HminℓrunDataU, HmaxℓrunDataU, ℓHxHrunDataU table of Halign, Havoid, HxH galactic longitude
Hmin$b$runDataU, Hmax$b$runDataU, bHxHrunDataU table of Halign, Havoid, HxH galactic latitude
HminℓbrunDataU, HmaxℓbrunDataU, ℓbHxHrunDataU table of Halign, Havoid, HxH both (ℓ,b)
rHminrunDataU,rHmaxrunDataU,rHxHrunDataU radial unit vectors to Hmin, Hmax, HxH
θHToHrunDataU angle from Hmin to Hmax, 0 <= θ <= 180°

```
(*Use the following cells to generate uncertainty runs*)

(*Definitions*)
(*μ   position angle in degrees, given catalog data
  σ   uncertainty in ψ, given catalog data
 nR   number of uncertainty runs to be generated, (USER)
  printInterval   number of runs between printing (USER)
  nRunPrint   counter to control printing
  ψSrc   set of ψ chosen from distribution ψ±σψ
  ηminαδH FindMinimum: {min η̄(H),ℓj,bj}
 ηmaxαδH FindMaximum: {max η̄(H),ℓj,bj} *)(*
ψdata  -  an optional table of ψs
   (Since RandomVariate is random, ψdata essential to repeat calculations.)
  rundataU - data saved from uncertainty runs
*)

(*nR=200; (*USER's choice*)
printInterval =Round[nR/10.];(*USER*)
μ=  Table[ψi[i],{i,nSrc}];
σ=  Table[σψi[i],{i,nSrc}];
runDataU={};ψData={};nRunPrint=0;
*)
```

```
(*For[nRun=1,nRun≤nR,nRun++,
 If[nRun>nRunPrint,Print["At the start of run ",nRun,", the time is ",
   TimeUsed[]," seconds and the memory in use is ",MemoryInUse[]," bytes."];
  nRunPrint=nRunPrint+printInterval];   ψSrc=Table[RandomVariate[
    NormalDistribution[μ⟦i⟧,σ⟦i⟧]],{i,nSrc}]; (*chosen from distribution ψ±σψ *)
 (*AppendTo[ψData,{nRun,ψSrc}];*)
 ηminℓbH=FindMinimum[ηBarAtHwithAnyψ[ℓH,bH,ψSrc],
   {{ℓH,ℓHbestAlign },{bH,bHbestAlign }},AccuracyGoal→5,PrecisionGoal→∞];
  ηmaxℓbH=
  FindMaximum[ηBarAtHwithAnyψ[ℓH,bH,ψSrc],
   {{ℓH,ℓHbestAvoid},{bH,bHbestAvoid}},AccuracyGoal→5,PrecisionGoal→∞];
 AppendTo[runDataU,{nRun,{ ηminℓbH⟦1⟧,{ℓH,bH}/.ηminℓbH⟦2⟧},
   {  ηmaxℓbH⟦1⟧,{ℓH,bH}/.ηmaxℓbH⟦2⟧}} ](*collect data*)    ]
*)
```

In[172]:=
```
(*A link to the data file "20230702RunDataU2000.dat" can be found in Ref. 7.*)
SetDirectory[NotebookDirectory[]]
(*Put[runDataU,"20230630RunDataU200.dat"]*)
runDataU = Get["20230702RunDataU2000.dat"];
```

Out[172]= C:\Users\momen\Dropbox\HOME_DESKTOP-0MRE5OJ\SendXXX_CJP_CEJPetc\SendViXra\20200715AlignmentMethod\20200715AlignmentMMAnotebooks\StarterKit

The number of uncertainty runs is 2000.

In[175]:=
```
runDataU⟦1⟧(*For a cursory data check.*)
runDataU⟦-1⟧
```

Out[175]= {1, {39.8634, {-97.2924, 21.8162}}, {49.1281, {-10.9386, -30.6322}}}

Out[176]= {2000, {40.1384, {-104.566, 13.5851}}, {49.1463, {-12.2615, -23.6118}}}

In[177]:=
```
(*name the quantities in runDataU.*)
ηBarMinrunDataU = Table[runDataU⟦i1, 2, 1⟧ , {i1, Length[runDataU]}];
ηBarMaxrunDataU = Table[runDataU⟦i1, 3, 1⟧ , {i1, Length[runDataU]}];
```

In[179]:=
```
(*Symmetry across a diameter means there are hubs diametrically opposed to each
 other. For statistical calculations the hubs must be collected together.*)
HminℓrunDataU = Table[If[
    180. < Mod[ runDataU⟦i1, 2, 2, 1⟧, 360.] < 360., Mod[ runDataU⟦i1, 2, 2, 1⟧, 360.] - 360.,
    Mod[180 +  runDataU⟦i1, 2, 2, 1⟧, 360.] - 360.] , {i1, Length[runDataU]}];
HminbrunDataU = Table[
   If[180. < Mod[ runDataU⟦i1, 2, 2, 1⟧, 360.] < 360., Mod[runDataU⟦i1, 2, 2, 2⟧, 180., -90.],
    -Mod[runDataU⟦i1, 2, 2, 2⟧, 180., -90.]], {i1, Length[runDataU]}];
HminℓbrunDataU = Table[{HminℓrunDataU⟦i1⟧, HminbrunDataU⟦i1⟧}, {i1, Length[runDataU]}];
```

In[182]:=
```
Hmaxℓrun DataU = Table[If[
    180. < Mod[ runDataU⟦i1, 3, 2, 1⟧, 360.] < 360., Mod[ runDataU⟦i1, 3, 2, 1⟧, 360.] - 360.,
    Mod[ runDataU⟦i1, 3, 2, 1⟧, 360.] - 180.] , {i1, Length[runDataU]}];
Hmaxbrun DataU = Table[
   If[180. < Mod[ runDataU⟦i1, 3, 2, 1⟧, 360.] < 360., Mod[runDataU⟦i1, 3, 2, 2⟧, 180., -90.],
    -Mod[runDataU⟦i1, 3, 2, 2⟧, 180., -90.]], {i1, Length[runDataU]}];
Hmaxℓbrun DataU = Table[{HmaxℓrunDataU⟦i1⟧, HmaxbrunDataU⟦i1⟧}, {i1, Length[runDataU]}];
```

In[185]:=
```
rHminrunDataU = Table[er[HminℓrunDataU⟦i1⟧, HminbrunDataU⟦i1⟧], {i1, Length[runDataU]}];
rHmaxrunDataU = Table[er[HmaxℓrunDataU⟦i1⟧, HmaxbrunDataU⟦i1⟧], {i1, Length[runDataU]}];
rHxHrunDataU0 =
  Table[Cross[rHminrunDataU⟦i1⟧, rHmaxrunDataU⟦i1⟧ ], {i1, Length[runDataU]}];
rHxHrunDataU = Table[
   rHxHrunDataU0⟦i1⟧ / (rHxHrunDataU0⟦i1⟧.rHxHrunDataU0⟦i1⟧)^(1/2), {i1, Length[runDataU]}];
```

In[189]:=
```
ℓHxHrunDataU = Table[ℓFROMr[rHxHrunDataU⟦i1⟧ ], {i1, Length[runDataU]}];
bHxHrunDataU = Table[bFROMr[rHxHrunDataU⟦i1⟧ ], {i1, Length[runDataU]}];
ℓbHxHrunDataU = Table[{ℓHxHrunDataU⟦i1⟧, bHxHrunDataU⟦i1⟧}, {i1, Length[runDataU]}];
θHToHrunDataU =
  Table[arccos[rHminrunDataU⟦i1⟧.rHmaxrunDataU⟦i1⟧], {i1, Length[runDataU]}];
```

In[193]:=
```
(*Check some named quantities.*)
i = 16;
{i, ηBarMinrunDataU⟦i⟧ , ηBarMaxrunDataU⟦i⟧, HminℓrunDataU⟦i⟧, HminbrunDataU⟦i⟧,
 HminℓbrunDataU⟦i⟧, HmaxℓrunDataU⟦i⟧, HmaxbrunDataU⟦i⟧, HmaxℓbrunDataU⟦i⟧}
runDataU⟦i⟧
Print["Compare named values for the ", i,
 "th uncertainty run (top) with the original record (bottom)."]
Clear[i]
```

Out[194]=
```
{16, 40.0097, 48.8704, -94.6364, 20.7513,
 {-94.6364, 20.7513}, -14.5964, -30.3025, {-14.5964, -30.3025}}
```

Out[195]=
```
{16, {40.0097, {-94.6364, 20.7513}}, {48.8704, {-14.5964, -30.3025}}}
```

```
Compare named values for the 16th uncertainty run (top) with the original record (bottom).
```

A6a. The Smallest Alignment Angle $\bar{\eta}_{\min}$

This section fits a Gaussian distribution to the $\bar{\eta}_{\min}$ from the uncertainty runs.

Definitions

sortηBarMinU, sortηBarMaxU   sort the list of $\bar{\eta}_{\min}$, $\bar{\eta}_{\max}$ from the uncertainty runs
η0minU, η0maxU   mean of the $\bar{\eta}_{\min}$,$\bar{\eta}_{\max}$ from the uncertainty runs
σminU,σmaxU   standard dev. of the $\bar{\eta}_{\min}$,$\bar{\eta}_{\max}$ from the uncertainty runs
hlminU0, hlminU, hlmaxU0, hlmaxU   histogram {quantity, bin height} tables, set up the NonlinearModelFit (nlm)
nlmminU,nlmmaxU   a Gaussian fit to the $\bar{\eta}_{\min}$ histogram
showNLMB,showNLMmaxU   plot of Gaussian and histogram
pTableNLMminU,pTableNLMmaxU   table of fit parameters
σηBarminUFit, ηBarminUFit,σηBarmaxUFit, ηBarmaxUFit   - half-width, and peak value of the fit

In[198]:=
```
sortηBarMinU = Sort[ηBarMinrunDataU];
η0minU = mean[ηBarMinrunDataU]; (*Guess the mean for the Gaussian. *)
σminU = stanDev[ηBarMinrunDataU];(*Guess the half-width.*)
hlminU0 = HistogramList[sortηBarMinU, {η0minU - 5 σminU, η0minU + 5 σminU, 0.4 σminU}];
hlminU = Table[{(1/2) (hlminU0[[1, i1]] + hlminU0[[1, i1 + 1]]), hlminU0[[2, i1]]},
  {i1, Length[ hlminU0[[2]] ]}];
nlmminU = NonlinearModelFit[hlminU, a Exp[- (1/2.) ((x - x0) / b)^2],
  {{a, Length[sortηBarMinU / 6]}, {b, σminU}, {x0, η0minU}}, x];(*x is ηBarMin*)
```

In[203]:=
```
pTableNLMminU = nlmminU["ParameterTable"] ;
{σηBarminUFit, ηBarminUFit} = {b, x0} /. nlmminU["BestFitParameters"];(*degrees*)
```

In[205]:=
```
showNLMB = Show[{Histogram[sortηBarMinU, {η0minU - 5 σminU, η0minU + 5 σminU, 0.4 σminU},
     PlotLabel → "Uncertainty runs, η̅min distribution",
     FrameLabel → {"η̅min, degrees", "ΔR"}, PlotTheme → "Detailed"],
    Plot[Normal[nlmminU], {x, η0minU - 5 σminU, η0minU + 5 σminU}, PlotLabel → "η̅min"],
    ListPlot[hlminU, PlotLabel → "η̅min"] }];
```

Out[206]=

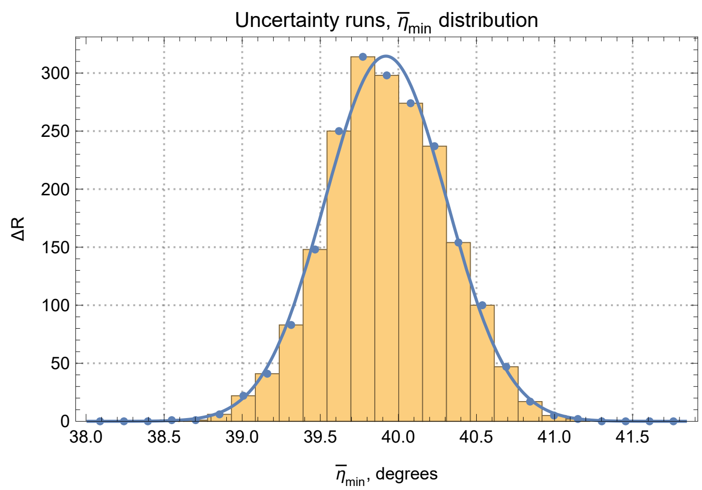


Fig. A6.1: The histogram for the uncertainty run values of the alignment angle $\overline{\eta}_{min}$ and its Gaussian fit.

The total number of uncertainty runs is R = Σ(ΔR) = 2000.

In[209]:=
```
Print["The uncertainty runs find the alignment angle η̅min to be η̅min = ",
 Round[100. ηBarminUFit] / 100., "°± ", Round[100. σηBarminUFit] / 100., "°."]
Print[
 "Compare that with the value found using the listed (best) values of ψ:  η̅min^best = ",
 Round[100. ηBarHbestAlign] / 100., "°."]
```

The uncertainty runs find the alignment angle $\overline{\eta}_{min}$ to be $\overline{\eta}_{min}$ = 39.92°± 0.39°.

Compare that with the value found using the listed (best) values of ψ: $\overline{\eta}_{min}^{best}$ = 39.71°.

A6b The Largest Avoidance Angle $\overline{\eta}_{max}$

Definitions: see A6a

In[211]:=
```
sortηBarMaxU = Sort[ηBarMaxrunDataU];
η0maxU = mean[ηBarMaxrunDataU ]; (*Guess the mean for the Gaussian. *)
σmaxU  = stanDev[ηBarMaxrunDataU ];(*Guess the half-width.*)
histogramrangemaxU = {η0maxU - 5 σmaxU, η0maxU + 5 σmaxU, 0.4 σmaxU};
hl0maxU = HistogramList[sortηBarMaxU, histogramrangemaxU];
hlmaxU = Table[{(1 / 2) (hl0maxU⟦1, i1⟧ + hl0maxU⟦1, i1 + 1⟧), hl0maxU⟦2, i1⟧},
  {i1, Length[ hl0maxU⟦2⟧ ]}];
nlmmaxU = NonlinearModelFit[hlmaxU, a Exp[- (1 / 2.) ((x - x0) / b)^2],
  {{a, 300.}, {b, σmaxU}, {x0, η0maxU}}, x];(*x is ηBarmaxU *)
nlmBmaxU = NonlinearModelFit[hlmaxU, {a Exp[- 1/2. ((x - x0)/b)^2] (*,b>0*)},
   {{a, Length[sortηBarMaxU / 6]}, {b, σmaxU }, {x0, η0maxU}}, x];
```

In[218]:=
```
pTableNLMmaxU  = nlmBmaxU["ParameterTable"] ;
{σηBarmaxFitU, ηBarmaxFitU}  =
  ParametersNLMmaxU = {b, x0} /. nlmBmaxU["BestFitParameters"];(*degrees*)
```

In[220]:=
```
showNLMmaxU = Show[{Histogram[sortηBarMaxU,
     histogramrangemaxU, PlotLabel → "Uncertainty runs, η̄max distribution",
     FrameLabel → {"η̄max, degrees", "ΔR"}, PlotTheme → "Detailed"],
    Plot[Normal[nlmBmaxU], {x, η0maxU - 5 σmaxU, η0maxU + 5 σmaxU}, PlotLabel → "η̄max"] ,
    ListPlot[hlmaxU, PlotLabel → "η̄max"]}];
```

Out[221]=

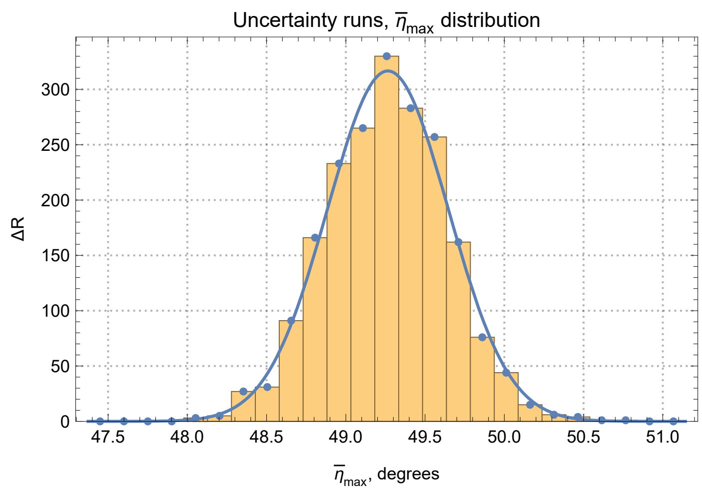


Fig. A6.2: The histogram for the uncertainty run values of the alignment angle $\overline{\eta}_{max}$ and its Gaussian fit

The total number of uncertainty runs is R = Σ(ΔR) = 2000.

The uncertainty runs find the avoidance angle $\overline{\eta}_{max}$ to be $\overline{\eta}_{max}$ = 49.27° ± 0.38°.

Compare that with the value found using the listed (best) values of ψ: $\overline{\eta}_{max}^{best}$ = 49.47°.

After Sec. A6, time and memory used are 480.594 seconds and 1 828 797 408 bytes.

A7. Uncertainty Runs, Alignment Hub H_{align} (= H_{min})

A7a. The location of the hub

Each uncertainty run returns an alignment hub H_{min}. In this section, we investigate the distribution of the locations the alignment Hubs H_{min}.

There are two hubs, H_{min} and $-H_{min}$ for each uncertainty run, by the symmetry across a diameter. So we collect the data together by moving the $-H_{min}$ hubs across a diameter to join the H_{min} hubs.

Definitions

Both RA and ℓ are longitudes, one in the Equatorial Coordinate System, one in the Galactic Coordinate System.
Likewise, dec and b are latitudes.
This notebook is set in the Galactic Coordinate System (ℓ,b). Occasionally, a RA or dec may appear where a ℓ or a b should be.

sortHminℓbrunDataU, sortHmaxℓbrunDataU, sortHxHℓbrunDataU   sorted tables of Halign, Havoid, HxH values of (ℓ,b) = (x,y), from all uncertainty runs, in degrees
lpHminU, lpHmaxU, lpHxHU   a plot of hubs from all the uncertainty runs

sortHminℓ, sortHmin$b$, sortHmaxℓ, sortHmax$b$, sortHxHℓ,sortHxHb sorted table of uncertainty run values for ℓ,b of Halign, Havoid, HxH
x0Hmin, x0Hmax,x0HxH   arithmetic average of $\ell$of Halign, Havoid, HxH uncertainty run values
y0Hmin, y0Hmax,y0HxH, y0   arithmetic average of b of Halign, Havoid, HxH uncertainty run values
dx0Hmin, dx0Hmax,dx0HxH   standard deviation of ℓ of Halign, Havoid,HxH uncertainty run values
dy0Hmin, dy0Hmax,dy0HxH   standard deviation of b of Halign, Havoid,HxH uncertainty run values
histogramrangeℓHminU, histogramrangeℓHmaxU, histogramrangeℓHxHU   parameters for histograms
histogramrange$b$HminU, histogramrange$b$HmaxU, histogramrange$b$HxHU   parameters for histograms
hl0xHmin, hlxHmin, hl0xHmax, hlxHmax,hl0xHxH, hlxHxH   histogram {quantity, bin height} tables needed to set up the NonlinearModelFit for hub ℓ(which is x)
hl0yHmin, hlyHmin, hl0yHmax, hlyHmax,hl0yHxH, hlyHxH   histogram {quantity, bin height} tables needed to set up the NonlinearModelFit for hub b(which is y)
pTablenlmxHmin, pTablenlmxHmax, pTablenlmxHxH,   parameter table for the Gaussian fit of hub ℓ = x
pTablenlmyHmin, pTablenlmyHmax, pTablenlmyHxH,   parameter table for the Gaussian fit of hub b = y
HminℓFit, HmaxℓFit, HxHℓFit,   most likely value, peak value of ℓ for hubs of Gaussian fit,
Hmin$b$Fit, Hmax$b$Fit, HxHbFit,   most likely value, peak value of b for hubs of Gaussian fit
σHminℓFit, σHmaxℓFit, σHxHℓFit,   half width of ℓ Gaussian fit for hubs
σHmin$b$Fit, σHmax$b$Fit, σHxHbFit,   half width of of b Gaussian fit for hubs

logOfnlmxHmin, logOfnlmxHmax, logOfnlmxHxH   Log of Gaussian fit expression ('exp' means exponential expression)
shownlmxHmin, shownlmxHmax, shownlmxHxH   Plot histogram and Gaussian fit for ℓ = x
shownlmyHmin, shownlmyHmax, shownlmyHxH   Plot histogram and Gaussian fit for b = y

Galactic longitude ℓ of H_{align}

```
In[230]:= sortHminℓbrunDataU = Sort[Union[HminℓbrunDataU]];
lpHminU =
  ListPlot[Union[HminℓbrunDataU], PlotRange → All, PlotStyle → {Blue, PointSize[0.01]},
   PlotLabel → "The alignment hubs from the uncertainty runs",
   FrameLabel → {"ℓ (deg)", "b (deg)"}, AspectRatio → 2 / 3, PlotTheme → "Detailed"];
```

```
In[232]:= sortHminℓ = Sort[HminℓrunDataU];
x0Hmin = mean[HminℓrunDataU ]; (*Guess the mean for the Gaussian. *)
dx0Hmin = stanDev[HminℓrunDataU ]; (*Guess the half-width.*)
histogramrangegLONHminU = {x0Hmin - 5 dx0Hmin, x0Hmin + 5 dx0Hmin, 0.4 dx0Hmin};
hl0xHmin = HistogramList[sortHminℓ, histogramrangegLONHminU];
hlxHmin = Table[{(1 / 2) (hl0xHmin[[1, i1]] + hl0xHmin[[1, i1 + 1]]), hl0xHmin[[2, i1]]},
  {i1, Length[ hl0xHmin[[2]] ]}];
nlmxHmin = NonlinearModelFit[hlxHmin, a Exp[- (1 / 2.) ((x - x0) / b)^2],
  {{a, Length[sortHminℓ / 6]}, {b, dx0Hmin}, {x0, x0Hmin}}, x]; (*x is Hminℓ*)
```

```
In[238]:= pTablenlmxHmin = nlmxHmin["ParameterTable"] ;
{σHminℓFit, HminℓFit} = ParametersnlmxHmin = {b, x0} /. nlmxHmin["BestFitParameters"];
(*degrees*)
Normal[nlmxHmin] /. {x → ℓHalign};
logOfnlmxHmin[x_] := - (1 / 2.) ((x - x0) / b)^2 /. nlmxHmin["BestFitParameters"]
logOfnlmxHmin[x];
```

```
In[243]:= shownlmxHmin = Show[{Histogram[sortHminℓ, histogramrangegLONHminU,
     PlotLabel → "Uncertainty run distribution of ℓ for H_align",
     FrameLabel → {"ℓ, degrees", "ΔR"}, PlotRange → All, PlotTheme → "Detailed"],
   Plot[Normal[nlmxHmin], {x, -108., -88.}, PlotRange → All, PlotLabel → "ℓHmin"],
   ListPlot[hlxHmin, PlotLabel → "ℓHmin"] }];
```

Out[244]=

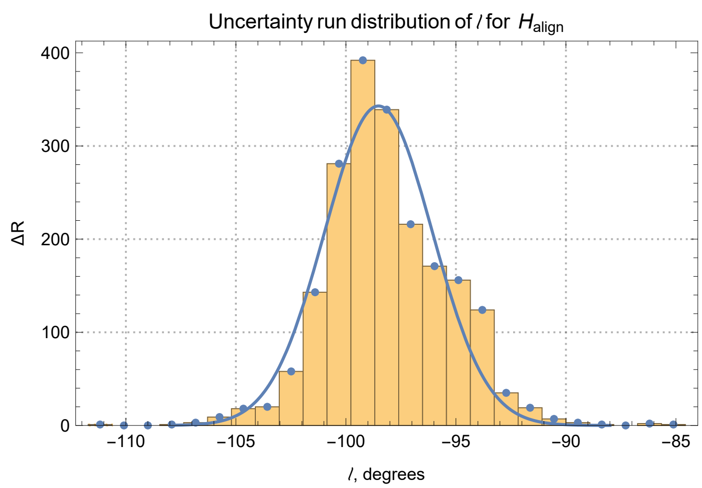


Figure A7.1. The histogram of uncertainty run values of alignment hub H_{align} ℓs. We display a single Gaussian fit with ℓ = −98.5194° ± 2.44653°.

The total number of runs is R = Σ(ΔR) = 2000.

Latitude b of H_{align}

In[247]:=
```
sortHminb = Sort[HminbrunDataU];
y0Hmin = mean[HminbrunDataU];(*Guess the mean for the Gaussian. *)
dy0Hmin = stanDev[HminbrunDataU];(*Guess the half-width.*)
histogramrangegLATHminU = {y0Hmin - 5 dy0Hmin, y0Hmin + 5 dy0Hmin, 0.4 dy0Hmin};
hl0yHmin = HistogramList[sortHminb, histogramrangegLATHminU];
hlyHmin = Table[{(1/2) (hl0yHmin[[1, i1]] + hl0yHmin[[1, i1 + 1]]), hl0yHmin[[2, i1]]},
   {i1, Length[ hl0yHmin[[2]] ]}];
nlmyHmin = NonlinearModelFit[hlyHmin, a Exp[-(1/2.) ((y - y0) / b)^2],
   {{a, Length[sortHminb / 6]}, {b, dy0Hmin}, {y0, y0Hmin}}, y];(*y is Hminb*)
```

In[253]:=
```
pTablenlmyHmin = nlmyHmin["ParameterTable"] ;
{σHminbFit, HminbFit} = ParametersnlmyHmin = {b, y0} /. nlmyHmin["BestFitParameters"];
(*degrees*)
Normal[nlmyHmin] /. {y → bHalign};
logOfnlmyHmin[y_] := -(1/2.) ((y - y0) / b)^2 /. nlmyHmin["BestFitParameters"]
logOfnlmyHmin[y];
```

In[258]:=
```
HminrFit = er[HminℓFit, HminbFit];
```

In[259]:=
```
shownlmyHmin = Show[{Histogram[sortHminb, histogramrangegLATHminU,
     PlotLabel → "Uncertainty run distribution of b for Halign",
     FrameLabel → {"b, degrees", "ΔR"}, PlotRange → All, PlotTheme → "Detailed"],
    Plot[Normal[nlmyHmin], {y, 3., 30.}, PlotRange → All, PlotLabel →
      "Galactic latitude b, for Hmin"], ListPlot[hlyHmin, PlotLabel → "bHmin"] }];
```

Out[260]=

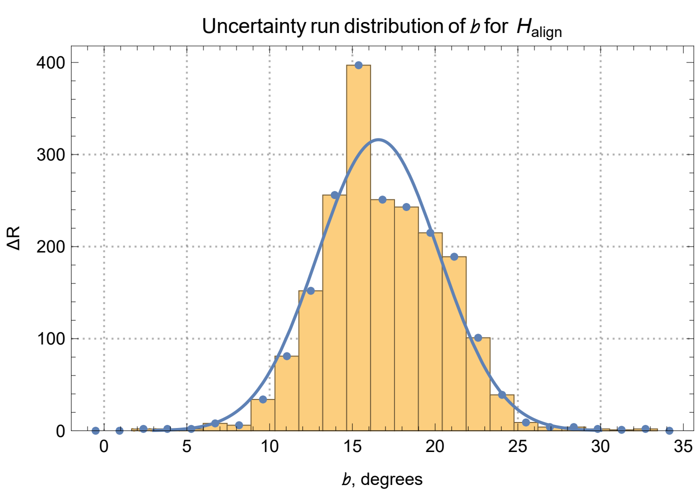


Figure A7.2. The histogram of uncertainty run values of alignment hub H_{align} bs.The fit gives b = 16.5773° ± 3.67628°.

The total number of runs is R = Σ(△R) = 2000.

A7b. Uncertainty ellipse surrounding the hub

Definitions

logofHminU, logofHmaxU, logofHxHU Log of the 2D (ℓ,b) Gaussian fit for the (α,δ) of H_{min}, H_{max}, HxH
frθHmin, frθHmax, frθHxH, equation for uncertainty ellipse r(θ) with H_{min} + (rcosθ,rsinθ), (H_{max}, HxH are similar) where likelihood down by $e^{-1/2}$
solverHminθ, solverHmaxθ, solverHxHθ, solution for radius of uncertainty ellipse r(θ)
rECSHminEllipse, rECSHmaxEllipse, , rECSHxHEllipse, max radial unit vector to uncertainty ellipse for H_{min}, H_{max}, HxH
aveHminErrorRadius, aveHmaxErrorRadius, aveHxHErrorRadius, average radius of the uncertainty ellipse for H_{min}, H_{max}, HxH

In[263]:=
```
logofHminU[x_, y_] := - (logOfnlmxHmin[x] + logOfnlmyHmin[y])
Print[
 "The negative log of the probability distribution for the coords (ℓ,δ) of Halign: ",
 logofHminU[ℓ, b], ", aside from a constant."]
```

The negative log of the probability distribution for the coords (ℓ,δ) of H_{align}: 0.0369958 $(-16.5773 + b)^2$ + 0.0835349 $(98.5194 + \ell)^2$, aside from a constant.

In[265]:=
```
plot3DLogHalign =
  Plot3D[{logofHminU[x, y], 0.5}, {x, x0 - 5., x0 + 5.} /. nlmxHmin["BestFitParameters"],
   {y, y0 - 5., y0 + 5.} /. nlmyHmin["BestFitParameters"],
   PlotLabel → "Negative log of the probability of (ℓ,b) for Halign",
   AxesLabel → {"ℓ (deg)", "b (deg)"}, PlotTheme → {"Scientific"}];
```

Out[266]=

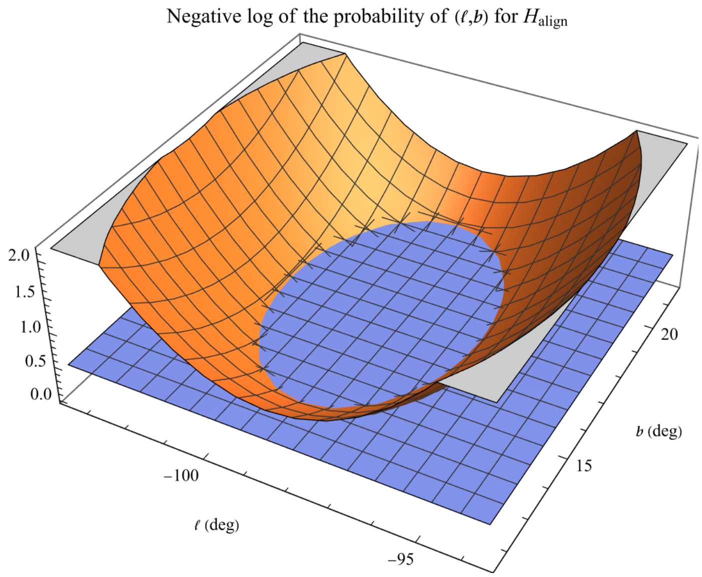


Figure A7.3: The negative log of the likelihood of (ℓ,b) for H_{align}, as a function of ℓ and b. Where the likelihood is down by a factor $e^{-1/2}$, the negative log is 0.5 and that defines the uncertainty ellipse at the half-width σ of the distribution.

In[268]:=
```
(*Find the curve for the intersection in Fig. A8.3.*)
frθHmin[r_, θ_] :=
 Simplify[(logofHminU[x, y]) - 0.5 /. {x → HminℓFit + r Cos[θ], y → HminbFit + r Sin[θ]}]
frθHmin[r, θ];
solverHminθ[θ_] := Solve[frθHmin[r, θ] == 0, r];
solverHminθ[θ];
rHminθ[θ_] := Abs[r /. solverHminθ[θ][[2]]]
rHminθ[θ];
rHminθ[0.8];
Plot[rHminθ[θ], {θ, 0, 2. π}];
aveHminErrorRadius = Integrate[rHminθ[θ], {θ, 0, 2 π}] / (2. π);
```

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

In[277]:=
```
showURunHalign =
  Show[{lpHminU, ParametricPlot[{HminℓFit + rHminθ[θ] Cos[θ], HminbFit + rHminθ[θ] Sin[θ]},
     {θ, 0, 2. π}, PlotStyle → Black, PlotRange → (*{{90.,180.},{-60.,0.}}*)All]}];
```

Out[278]=

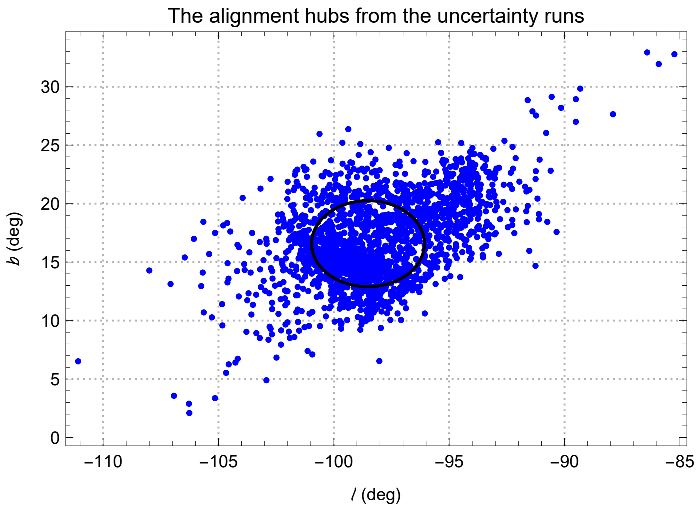


Figure A7.4: All of the alignment hubs H_{align} from uncertainty runs. The ellipse encloses the locations of the hubs within 1σ. Symmetry across diameters means there is another set diametrically opposite to those displayed here.

In[280]:=
```
ℓHminEllipse[θ_] := HminℓFit + rHminθ[θ] Cos[θ]
𝒷HminEllipse[θ_] := Hmin𝒷Fit + rHminθ[θ] Sin[θ]
rOfHAlignRegion[θ_] := er[ℓHminEllipse[θ], 𝒷HminEllipse[θ]]
```

In[283]:=
```
findMinℓHminEllipse = FindMinimum[ℓHminEllipse[θ], {θ, π - π / 8.}];
findMaxℓHminEllipse = FindMaximum[ℓHminEllipse[θ], {θ, -π / 8.}];
σℓHminEllipse = ((ℓHminEllipse[θ] /. findMaxℓHminEllipse⟦2, 1⟧) -
    (ℓHminEllipse[θ] /. findMinℓHminEllipse⟦2, 1⟧)) / 2.;
```

··· Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

··· Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

··· Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

··· Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

In[286]:=
```
findMin𝒷HminEllipse = FindMinimum[𝒷HminEllipse[θ], {θ, π}];
findMax𝒷HminEllipse = FindMaximum[𝒷HminEllipse[θ], {θ, 2 π}];
σ𝒷HminEllipse = ((𝒷HminEllipse[θ] /. findMax𝒷HminEllipse[[2, 1]]) -
    (𝒷HminEllipse[θ] /. findMin𝒷HminEllipse[[2, 1]])) / 2.;
```

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

FindMinimum: The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

FindMaximum: The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient increase in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

The location of the hub H_{align} can be written in the form $(\ell, b) \pm (\sigma\ell, \sigma b)$ = {-98.5194, 16.5773} ± {2.44653, 3.67628}, in degrees.

There is an uncertainty ellipse for use when plotting the hub.

After Sec. A7, time and memory used are 499.704 seconds and 1 880 872 992 bytes.

A8. Uncertainty Runs, Avoidance Hub H_{avoid} (= H_{max})

A8a. The location of the hub

Each uncertainty run returns an avoidance hub H_{max}. In this section, we investigate the distribution of the locations the avoidance hubs H_{max}.

There are two hubs, H_{max} and $-H_{max}$ for each uncertainty run, by the symmetry across a diameter. So we collect all the hubs together by moving the $-H_{max}$ hubs across a diameter to join the H_{max} hubs.

Definitions: See Sec. A7.

In[295]:=
```
sortHmaxℓ𝒷runDataU = Sort[Union[Hmaxℓ𝒷runDataU]];
lpHmaxU = ListPlot[Union[Hmaxℓ𝒷runDataU],
   PlotRange → {{-40., 20.}, {-60., 0.}}, PlotStyle → {Red, PointSize[0.01]},
   PlotLabel → "The avoidance hubs from the uncertainty runs",
   FrameLabel → {"ℓ (deg)", "𝒷 (deg)"}, AspectRatio → 1 / 2, PlotTheme → "Detailed"];
```

In[297]:=
```
sortHmaxℓ = Sort[HmaxℓrunDataU];
x0Hmax = mean[HmaxℓrunDataU]; (*Guess the mean for the Gaussian. *)
dx0Hmax = stanDev[HmaxℓrunDataU] / 2.5; (*Guess the half-width.*)
histogramrangeHmaxℓ = (*{257.,323.,2.5}*)
  {x0Hmax - 5 dx0Hmax, x0Hmax + 5 dx0Hmax, dx0Hmax};
hl0xHmax = HistogramList[sortHmaxℓ, histogramrangeHmaxℓ];
hlxHmax = Table[{(1 / 2) (hl0xHmax[[1, i1]] + hl0xHmax[[1, i1 + 1]]), hl0xHmax[[2, i1]]},
  {i1, Length[ hl0xHmax[[2]] ]}];
nlmxHmax = NonlinearModelFit[hlxHmax, a Exp[- (1 / 2.) ((x - x0) / b)^2],
  {{a, Length[sortHmaxℓ / 6]}, {b, dx0Hmax}, {x0, x0Hmax}}, x]; (*x is Hmaxℓ*)
```

In[303]:=
```
pTablenlmxHmax = nlmxHmax["ParameterTable"] ;
{σHmaxℓFit, HmaxℓFit} = ParametersnlmxHmax = {b, x0} /. nlmxHmax["BestFitParameters"];
(*degrees*)
Normal[nlmxHmax] /. {x → ℓHavoid};
logOfnlmxHmax[x_] := - (1 / 2.) ((x - x0) / b)^2 /. nlmxHmax["BestFitParameters"]
logOfnlmxHmax[x]
```

Out[307]= $-0.104567\ (12.726 + x)^2$

In[308]:=
```
shownlmxHmax = Show[{Histogram[sortHmaxℓ, histogramrangeHmaxℓ,
     PlotLabel → "Uncertainty run distribution of ℓ for Havoid",
     FrameLabel → {"ℓ, degrees", "ΔR"}, PlotRange → All, PlotTheme → "Detailed"],
   Plot[Normal[nlmxHmax], {x, -90., 60.}, PlotRange → All, PlotLabel → "ℓHmax"],
   ListPlot[hlxHmax, PlotLabel → "ℓHmax"] }];
```

Out[309]=

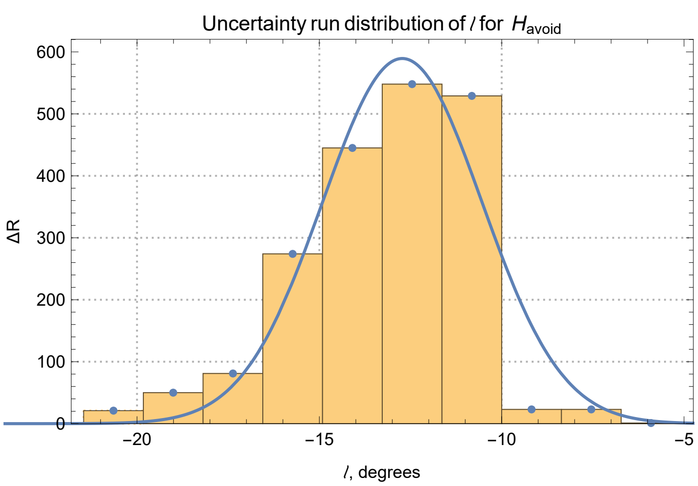


Figure A8.1. The histogram of uncertainty run values of H_{avoid} gLONs. The fit gives ℓ = −12.726° ± 2.18669°.

The total number of uncertainty runs is R = Σ(ΔR) = 2000.

In[312]:=
```
sortHmax𝒷 = Sort[Hmax𝒷runDataU];
y0Hmax = mean[Hmax𝒷runDataU];(*Guess the mean for the Gaussian. *)
dy0Hmax = stanDev[Hmax𝒷runDataU];(*Guess the half-width.*)
histogramrangeHmax𝒷 = {y0Hmax - 5 dy0Hmax, y0Hmax + 5 dy0Hmax, 0.4 dy0Hmax};
hl0yHmax = HistogramList[sortHmax𝒷, histogramrangeHmax𝒷];
hlyHmax = Table[{(1/2) (hl0yHmax[[1, i1]] + hl0yHmax[[1, i1 + 1]]), hl0yHmax[[2, i1]]},
  {i1, Length[ hl0yHmax[[2]] ]}];
nlmyHmax = NonlinearModelFit[hlyHmax, a Exp[- (1/2.) ((y - y0)/b)^2],
  {{a, Length[sortHmax𝒷/6]}, {b, dy0Hmax}, {y0, y0Hmax}}, y];(*y is Hmax𝒷*)
```

In[318]:=
```
pTablenlmyHmax = nlmyHmax["ParameterTable"] ;
{σHmax𝒷Fit, Hmax𝒷Fit} = ParametersnlmyHmax = {b, y0} /. nlmyHmax["BestFitParameters"];
(*degrees*)
Normal[nlmyHmax] /. {y → 𝒷Havoid};
logOfnlmyHmax[y_] := - (1/2.) ((y - y0)/b)^2 /. nlmyHmax["BestFitParameters"]
logOfnlmyHmax[y];
```

In[323]:=
```
HmaxrFit = er[HmaxℓFit, Hmax𝒷Fit];
```

In[324]:=
```
shownlmyHmax = Show[{Histogram[sortHmax𝒷, histogramrangeHmax𝒷,
     PlotLabel → "Uncertainty run distribution of 𝒷 for Havoid",
     FrameLabel → {"𝒷, degrees", "ΔR"}, PlotRange → All, PlotTheme → "Detailed"],
    Plot[Normal[nlmyHmax], {y, -60., 0.}, PlotRange → All, PlotLabel → "𝒷Hmax"],
    ListPlot[hlyHmax, PlotLabel → "𝒷Hmax"] }];
```

Out[325]=

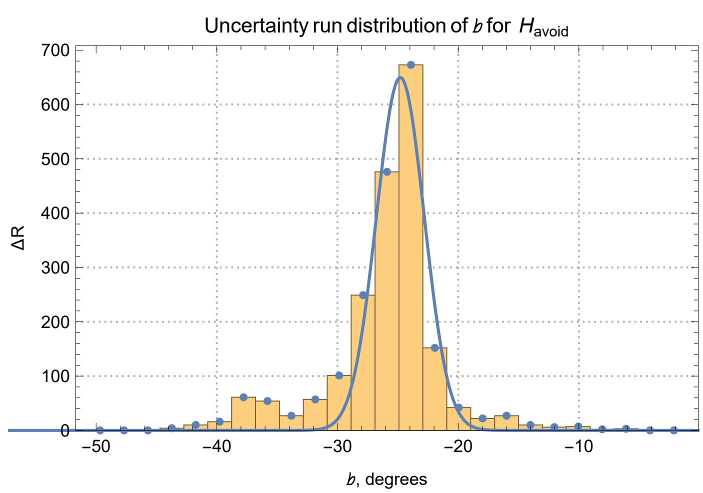


Figure A8.2. The histogram of uncertainty run values of H_{avoid} 𝒷s. A double Gaussian might fit better, with a primary peak at 𝒷 = ??°. The single Gaussian fit gives 𝒷 = −24.7924° ± 1.98282°.

The total number of uncertainty runs is R = Σ(ΔR) = 2000.

A8b. Uncertainty ellipse surrounding the hub

Definitions

logofHmaxU Log of the 2D (ℓ,*b*) Gaussian fit for the (α,δ) of H_{max}
frθHmax equation for uncertainty ellipse r(θ) with H_{max} + (rcosθ,rsinθ), where likelihood down by $e^{-1/2}$
solverHmaxθ solution for radius of uncertainty ellipse r(θ)
rECSHmaxEllipse radial unit vector to uncertainty ellipse for H_{max}
aveHmaxErrorRadius average radius of the uncertainty ellipse for H_{max}

In[328]:=
```
logofHmaxU[x_, y_] := - (logOfnlmxHmax[x] + logOfnlmyHmax[y])
Print["The negative log of the probability distribution for Hmax: ",
 logofHmaxU[ℓ, b], ", aside from a dropped constant."]
```

The negative log of the probability distribution for H_{max}: 0.127175 $(24.7924 + b)^2$ + 0.104567 $(12.726 + \ell)^2$, aside from a dropped constant.

In[330]:=
```
plot3DLogHavoid =
  Plot3D[{logofHmaxU[x, y], 0.5}, {x, x0 - 4., x0 + 4.} /. nlmxHmax["BestFitParameters"],
   {y, y0 - 3., y0 + 3.} /. nlmyHmax["BestFitParameters"],
   PlotLabel → "Negative log of the probability of (ℓ,b) for Hmax",
   AxesLabel → {"ℓ (deg)", "b (deg)"}, PlotTheme → {"Scientific"}];
```

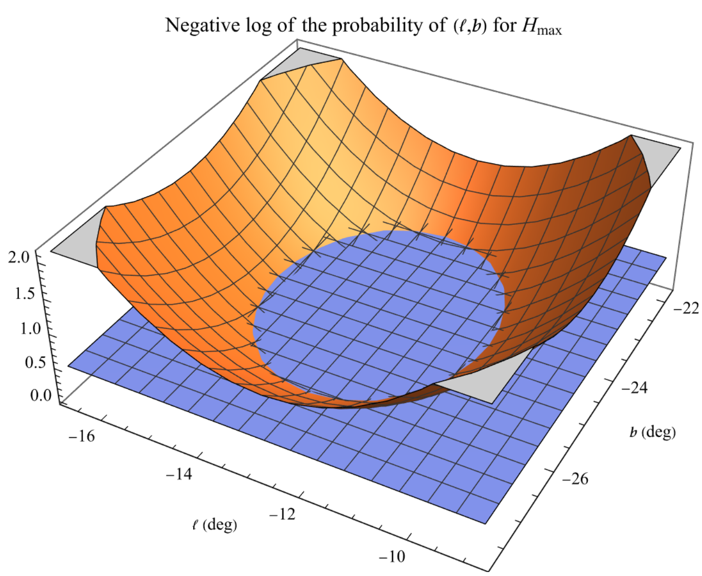
Out[331]=


Figure A8.3: The negative log of the likelihood of (ℓ,*b*) for H_{avoid}, as a function of ℓ and *b*. Where the likelihood is down by a factor $e^{-1/2}$, the negative log is 0.5 and that defines the uncertainty ellipse at the half-width σ of the distribution.

In[333]:=
```
(*Find the curve for the intersection in Fig. A9.3*)
frθHmax[r_, θ_] :=
 Simplify[(logofHmaxU[x, y]) - 0.5 /. {x → Hmaxℓ Fit + r Cos[θ], y → HmaxbFit + r Sin[θ]}]
frθHmax[r, θ];
solverHmaxθ[θ_] := Solve[frθHmax[r, θ] == 0, r];
solverHmaxθ[θ];
rHmaxθ[θ_] := Abs[r /. solverHmaxθ[θ]〚2〛]
rHmaxθ[θ];
rHmaxθ[0.8];
Plot[rHmaxθ[θ], {θ, 0, 2. π}];
aveHmaxErrorRadius = Integrate[rHmaxθ[θ], {θ, 0, 2 π}] / (2. π);
```

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

In[342]:=
```
showURunHavoid =
  Show[{lpHmaxU, ParametricPlot[{HmaxℓFit + rHmaxθ[θ] Cos[θ], HmaxbFit + rHmaxθ[θ] Sin[θ]},
     {θ, 0, 2. π}, PlotStyle → Black, PlotRange → All]}];
```

Out[343]=

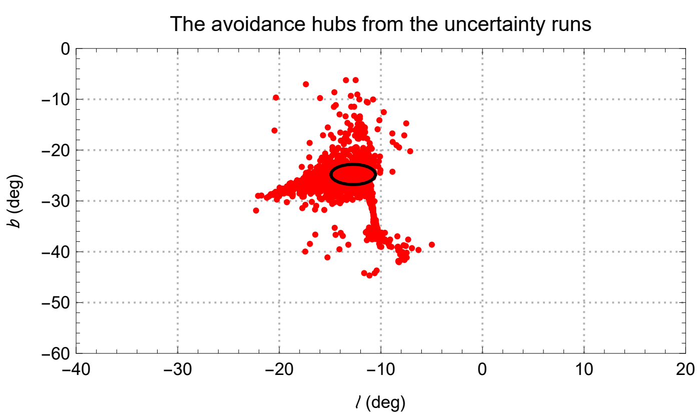


Figure A8.4: All of the alignment hubs H_{avoid} from uncertainty runs. The ellipse encloses the locations of the hubs within 1σ. Symmetry across diameters means there is another set diametrically opposite to those displayed here.

In[345]:=
```
ℓHmaxEllipse[θ_] := HmaxℓFit + rHmaxθ[θ] Cos[θ]
bHmaxEllipse[θ_] := HmaxbFit + rHmaxθ[θ] Sin[θ]
rOfHAvoidRegion[θ_] := er[ℓHmaxEllipse[θ], bHmaxEllipse[θ]]
```

```
In[348]:= findMinℓHmaxEllipse = FindMinimum[ℓHmaxEllipse[θ], {θ, 3 π / 2}];
findMaxℓHmaxEllipse = FindMaximum[ℓHmaxEllipse[θ], {θ, π / 2}];
σℓHmaxEllipse = ((ℓHmaxEllipse[θ] /. findMaxℓHmaxEllipse[[2, 1]]) -
    (ℓHmaxEllipse[θ] /. findMinℓHmaxEllipse[[2, 1]])) / 2.;
```

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

```
In[351]:= (*Plot[{ℓHmaxEllipse[θ],bHmaxEllipse[θ]},{θ,0.,2.π}]*)

In[352]:= findMinbHmaxEllipse = FindMinimum[{bHmaxEllipse[θ], 0 ≤ θ < 2. π}, {θ, -π}];
findMaxbHmaxEllipse = FindMaximum[{bHmaxEllipse[θ], 0 ≤ θ < 2. π}, {θ, 0.}];
σbHmaxEllipse = ((bHmaxEllipse[θ] /. findMaxbHmaxEllipse[[2, 1]]) -
    (bHmaxEllipse[θ] /. findMinbHmaxEllipse[[2, 1]])) / 2.;
```

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

The location of the hub H_{avoid} is (ℓ,b) ± (σℓ,σb) = {-12.726, -24.7924} ± {2.18669, 0.99141}, in degrees.

There is an uncertainty ellipse available for use when plotting the hub, {ℓHmaxEllipse[θ],bHmaxEllipse[θ]}.

After Sec. A8, time and memory used are 520.563 seconds and 1 885 196 472 bytes.

A9. Uncertainty Runs, normal $H_{align} \times H_{avoid}$ (= *HxH*)

Definitions: See Sec. A7.

A9a. The location of HxH

In[361]:=
```
sortHxHℓbrunDataU = Sort[Union[ℓbHxHrunDataU]];
lpHxHU = ListPlot[Union[ℓbHxHrunDataU],
   PlotRange → {{-20., 110.}, {0., 90.}}, PlotStyle → {Purple, PointSize[0.01]},
   PlotLabel → "The HxH direction from the uncertainty runs",
   FrameLabel → {"ℓ (deg)", "b (deg)"}, AspectRatio → 1.;
   PlotTheme → "Detailed"];
```

In[363]:=
```
sortHxHℓ = Sort[ℓHxHrunDataU];
x0HxH = mean[ℓHxHrunDataU]; (*Guess the mean for the Gaussian. *)
dx0HxH = stanDev[ℓHxHrunDataU]; (*Guess the half-width.*)
histogramrangeℓHxHU = {x0HxH - 5 dx0HxH, x0HxH + 5 dx0HxH, 0.4 dx0HxH};
hl0xHxH = HistogramList[sortHxHℓ, histogramrangeℓHxHU];
hlxHxH = Table[{(1/2) (hl0xHxH[[1, i1]] + hl0xHxH[[1, i1 + 1]]), hl0xHxH[[2, i1]]},
  {i1, Length[ hl0xHxH[[2]] ]}];
nlmxHxH = NonlinearModelFit[hlxHxH, a Exp[- (1/2.) ((x - x0) / b)^2],
  {{a, Length[sortHxHℓ / 6]}, {b, dx0HxH}, {x0, x0HxH}}, x]; (*x is HxHℓ*)
```

In[369]:=
```
pTablenlmxHxH = nlmxHxH["ParameterTable"] ;
{σHxHℓFit, HxHℓFit} = ParametersnlmxHxH = {b, x0} /. nlmxHxH["BestFitParameters"];
(*degrees*)
Normal[nlmxHxH] /. {x → ℓHxH};
logOfnlmxHxH[x_] := - (1/2.) ((x - x0) / b)^2 /. nlmxHxH["BestFitParameters"]
logOfnlmxHxH[x];
```

In[374]:=
```
shownlmxHxH = Show[{Histogram[sortHxHℓ,
     histogramrangeℓHxHU, PlotLabel → "Uncertainty run distribution of ℓ for HxH",
     FrameLabel → {"ℓ, degrees", "ΔR"}, PlotRange → All, PlotTheme → "Detailed"],
   Plot[Normal[nlmxHxH], {x, -15., 60.}, PlotRange → All, PlotLabel → "ℓHxH"],
   ListPlot[hlxHxH, PlotLabel → "ℓHxH"] }];
```

Out[375]=

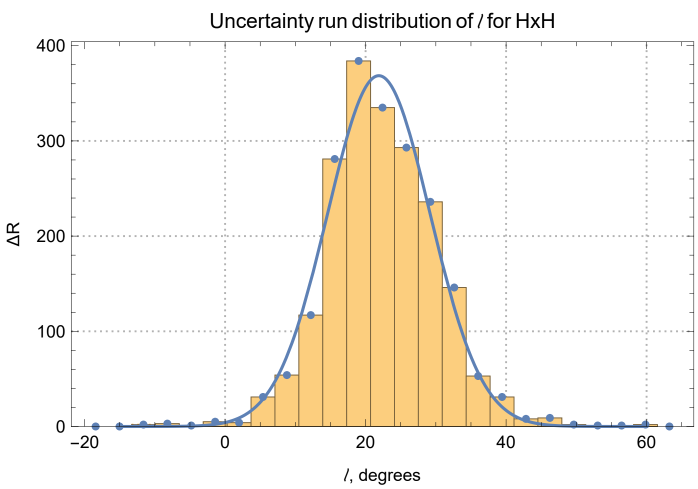


Figure A9.1. The histogram of uncertainty run values of HxH ℓs. The fit gives ℓ = 21.8861° ± 7.29949°.

The total number of uncertainty runs is R = Σ(ΔR) = 2000.

In[378]:=
```
sortHxHb = Sort[bHxHrunDataU];
y0HxH = mean[bHxHrunDataU];(*Guess the mean for the Gaussian. *)
dy0HxH = stanDev[bHxHrunDataU];(*Guess the half-width.*)
histogramrangebHxHU = {y0HxH - 5 dy0HxH, y0HxH + 5 dy0HxH, 0.4 dy0HxH};
hl0yHxH = HistogramList[sortHxHb, histogramrangebHxHU];
hlyHxH = Table[{(1 / 2) (hl0yHxH[[1, i1]] + hl0yHxH[[1, i1 + 1]]), hl0yHxH[[2, i1]]},
   {i1, Length[ hl0yHxH[[2]] ]}];
nlmyHxH = NonlinearModelFit[hlyHxH, a Exp[- (1 / 2.) ((y - y0) / b)^2],
   {{a, Length[sortHxHb / 6]}, {b, dy0HxH}, {y0, y0HxH}}, y];(*y is HxHb*)
```

In[384]:=
```
pTablenlmyHxH = nlmyHxH["ParameterTable"] ;
{σHxHbFit, HxHbFit} = ParametersnlmyHxH = {b, y0} /. nlmyHxH["BestFitParameters"];
Normal[nlmyHxH] /. {y → bHxH};
logOfnlmyHxH[y_] := - (1 / 2.) ((y - y0) / b)^2 /. nlmyHxH["BestFitParameters"]
logOfnlmyHxH[y];
```

In[389]:=
```
shownlmyHxH = Show[{Histogram[sortHxHb,
     histogramrangebHxHU, PlotLabel → "Uncertainty run distribution of b for HxH",
     FrameLabel → {"b, degrees", "ΔR"}, PlotRange → All, PlotTheme → "Detailed"],
    Plot[Normal[nlmyHxH], {y, 40., 80.}, PlotRange → All, PlotLabel → "bHxH"],
    ListPlot[hlyHxH, PlotLabel → "bHxH"] }];
```

Out[390]=

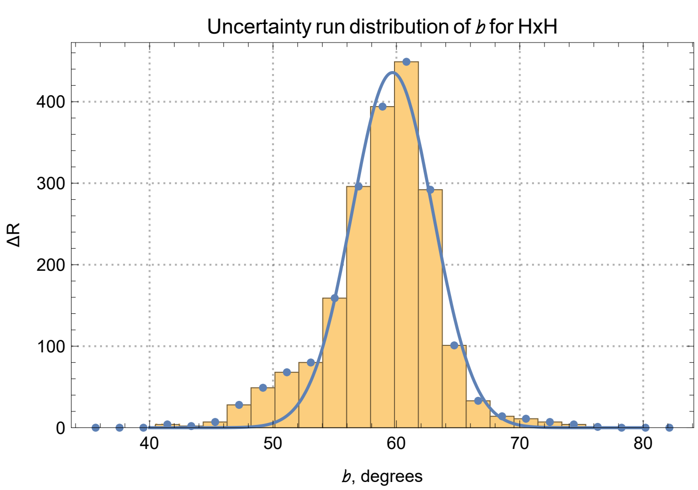


Figure A9.2. The histogram of uncertainty run values of HxH bs. The fit gives b = 59.6843° ± 3.28406°.

The total number of uncertainty runs is R = Σ(ΔR) = 2000.

A9b. Uncertainty ellipse surrounding HxH

Definitions: See Sec. A7b.

In[393]:=
```
logofHxHU[x_, y_] := - (logOfnlmxHxH[x] + logOfnlmyHxH[y])
Print["The negative log of the probability distribution for HxH: ",
 logofHxHU[ℓ, b], ", aside from a dropped constant."]
```

The negative log of the probability distribution for HxH:
0.0463604 (−59.6843 + b)^2 + 0.00938395 (−21.8861 + ℓ)^2, aside from a dropped constant.

In[395]:=
```
plot3DLogHxH =
  Plot3D[{logofHxHU[x, y], 0.5}, {x, x0 - 8., x0 + 8.} /. nlmxHxH["BestFitParameters"],
   {y, y0 - 4., y0 + 4.} /. nlmyHxH["BestFitParameters"],
   PlotLabel → "Negative log of the probability of (ℓ,b) for HxH",
   AxesLabel → {"ℓ (deg)", "b (deg)"}, PlotTheme → {"Scientific"}];
```

Out[396]=

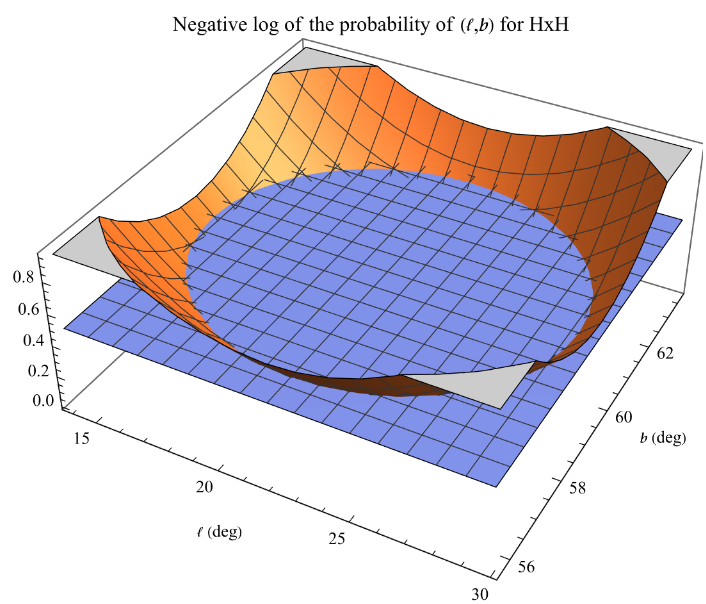


Figure A9.3: The negative log of the likelihood of (ℓ,b) for HxH, as a function of ℓ and b. Where the likelihood is down by a factor $e^{-1/2}$, the negative log is 0.5 and that defines the uncertainty ellipse at the half-width σ of the distribution.

In[398]:=
```
(*Find the curve for the intersection in Fig. A10.3*)
frθHxH[r_, θ_] :=
 Simplify[(logofHxHU[x, y]) - 0.5 /. {x → HxHℓFit + r Cos[θ], y → HxHbFit + r Sin[θ]}]
frθHxH[r, θ];
solverHxHθ[θ_] := Solve[frθHxH[r, θ] == 0, r];
solverHxHθ[θ];
rHxHθ[θ_] := Abs[r /. solverHxHθ[θ][[2]]]
rHxHθ[θ];
rHxHθ[0.8];
Plot[rHxHθ[θ], {θ, 0, 2. π}, PlotTheme → {"Detailed"}];
aveHminErrorRadius = Integrate[rHxHθ[θ], {θ, 0, 2 π}] / (2. π);
```

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

In[407]:=
```
showURunHxH =
  Show[{lpHxHU, ParametricPlot[{HxHℓFit + rHxHθ[θ] Cos[θ], HxHbFit + rHxHθ[θ] Sin[θ]},
     {θ, 0, 2. π}, PlotStyle → Black, PlotRange → {{90., 180.}, {-60., 0.}}]}];
```

Out[408]=

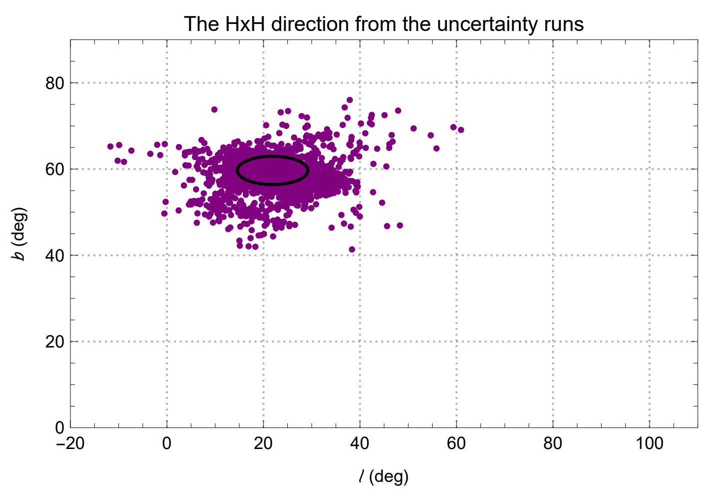


Figure A9.4: All of the alignment hubs HxH from uncertainty runs. The ellipse encloses the locations of the hubs within 1σ. Symmetry across diameters means there is another set diametrically opposite to those displayed here.

In[410]:=
```
ℓHxHEllipse[θ_] := HxHℓFit + rHxHθ[θ] Cos[θ]
bHxHEllipse[θ_] := HxHbFit + rHxHθ[θ] Sin[θ]
rOfHxHRegion[θ_] := er[ℓHmaxEllipse[θ], bHmaxEllipse[θ]]
```

In[413]:=
```
(*NGP and HxH are in the same general direction*)
rNGP = {0., 0., 1.};
rHxHFit = er[HxHℓFit, HxHbFit];
θNGPtoHxHFit = 90. - HxHbFit;
(*Table[er[HxHℓFit+rHxHθ[θ]Cos[θ],HxHbFit+rHxHθ[θ]Sin[θ]],{θ,0.,2.π,π/6.}];
Table[arccos[Abs[rNGP.er[HxHℓFit+rHxHθ[θ]Cos[θ],HxHbFit+rHxHθ[θ]Sin[θ]]]],
 {θ,0.,2.π,π/6.}];*)
bHxHFitFindMin = FindMinimum[HxHbFit + rHxHθ[θ] Sin[θ], {θ, 0.}];
bHxHFitSmall = bHxHFitFindMin[[1]];
bHxHFitFindMax = FindMaximum[HxHbFit + rHxHθ[θ] Sin[θ], {θ, 0.}];
bHxHFitBig = bHxHFitFindMax[[1]];
```

••• Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

••• Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

The location of the hub HxH is (ℓ,b) ± (σℓ,σb) = {21.8861, 59.6843} ± {7.29949, 3.28406}, in degrees.

There is an uncertainty ellipse available for use when plotting the hub, {ℓGalHxHEllipse[θ],bGalHxHEllipse[θ]}.

After Sec. A9, time and memory used are 540.533 seconds and 1 887 239 640 bytes.

A10. Uncertainty in the angle between Halign and Havoid

Definitions

θHalignHavoidU	angles θ from Halign and Havoid from fit, all uncertainty run values, deg.
lpθHalignHavoidU	a plot of angles θ from Halign and Havoid from fit,
sortθHalignHavoidU	sorted table of angles θ from Halign and Havoid from fit, deg.
θHalignHavoidUm	arithmetic average of angles θ from Halign and Havoid from fit
σθHalignHavoidUm	standard deviation of angles θ from Halign and Havoid from fit
histogramrangeθU	parameters for histograms
hl0θU, hlθU	histogram {quantity, bin height} tables needed to set up the NonlinearModelFit
nlmBθU	Gaussian fit to histogram θ distribution
pTableNLMθU	parameter table for the Gaussian fit
θHalignHavoidUlikely	peak value of Gaussian fit to θ distribution
σθHalignHavoidUlikely	half width of Gaussian fit to θ distribution
showNLMθU	Plot histogram and fit

In[426]:=
```
θHalignHavoidU =
  Table[arccos[rHminrunDataU[[i1]].rHmaxrunDataU[[i1]]], {i1, Length[runDataU]}];
```

In[427]:=
```
(*Plot the uncertainty run values for the angle θ between hubs Halign and Havoid.*)
(*lpθHalignHavoidU=
 ListPlot[Sort[θHalignHavoidU],FrameLabel→{"Count","θ, deg"},PlotLabel→
   "arc θ between hubs Halign and Havoid (uncertainty runs)",PlotTheme→"Detailed"]*)
```

In[428]:=
```
sortθHalignHavoidU = Sort[θHalignHavoidU];
θHalignHavoidUm = mean[θHalignHavoidU ]; (*Guess the mean for the Gaussian. *)
σθHalignHavoidU = stanDev[θHalignHavoidU ];(*Guess the half-width.*)
histogramrangeθU = {θHalignHavoidUm - 5 σθHalignHavoidU,
   θHalignHavoidUm + 5 σθHalignHavoidU, 0.4 σθHalignHavoidU};
hl0θU = HistogramList[sortθHalignHavoidU, histogramrangeθU];
hlθU =
  Table[{(1 / 2) (hl0θU[[1, i1]] + hl0θU[[1, i1 + 1]]), hl0θU[[2, i1]]}, {i1, Length[ hl0θU[[2]] ]}];
(*nlmθU=NonlinearModelFit[hlθU,a Exp[-(1/2.)((x-x0)/b)^2],
   {{a,300.},{b,σθHalignHavoidU},{x0,θHalignHavoidUm}},x];*)(*x is θ from H to H *)
nlmBθU = NonlinearModelFit[hlθU, {a Exp[-1/2. ((x - x0)/b)^2] (*,b>0*)},
   {{a, Length[sortθHalignHavoidU / 6]}, {b, σθHalignHavoidU }, {x0, θHalignHavoidUm}}, x];
```

In[435]:=
```
pTableNLMθU = nlmBθU["ParameterTable"] ;
{σθHalignHavoidUlikely, θHalignHavoidUlikely} =
  ParametersNLMθU = {b, x0} /. nlmBθU["BestFitParameters"];(*degrees*)
```

In[437]:=
```
showNLMθU = Show[{Histogram[sortθHalignHavoidU, histogramrangeθU,
     PlotLabel → "Uncertainty run distribution of θ between HAlign and HAvoid ",
     FrameLabel → {"θ, degrees", "ΔR"}, PlotTheme → "Detailed"], Plot[Normal[nlmBθU],
     {x, θHalignHavoidUm - 5 σθHalignHavoidU, θHalignHavoidUm + 5 σθHalignHavoidU},
     PlotLabel → "θ between HAlign and HAvoid ", PlotRange → All] ,
    ListPlot[hlθU, PlotLabel → "θ between HAlign and HAvoid "]}];
```

Out[438]=

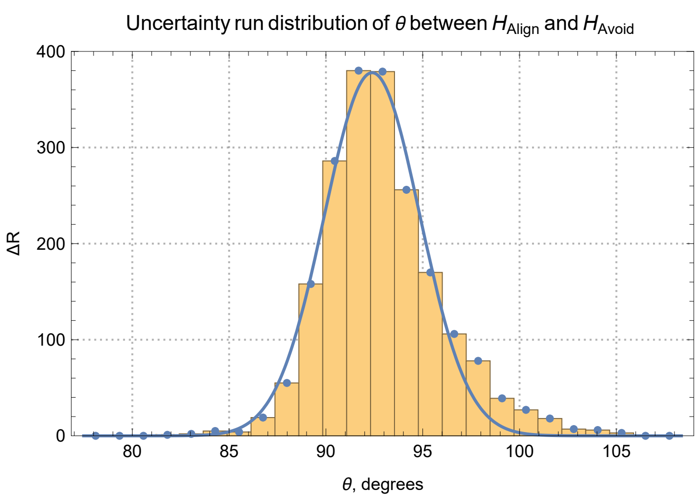


Figure A10.1: The histogram for the angle θ between H_{Align} and H_{Avoid} and its Gaussian fit. The fit gives θ = 92.3868° ± 2.46751°.

The angle Θ between H_{Align} and H_{Avoid}, Θ = 92.4° ± 2.5°.

Compare that with the value found using the listed (best) values of ψ, $\Theta_{best\psi}$ = 93.6°.

After Sec. A10, time and memory used are 542.158 seconds and 1 887 369 512 bytes.

A11. Random Runs

The problem of “significance” is to determine the likelihood that random polarizations directions ψ would have better alignment or avoidance results than the observed polarization directions have. The most reliable method of finding the significance of either value is to creates many copies of the QSO sample but assign the sources randomly directed polarizations. Collect the results, the angles $\overline{\eta}_{\min}$ and $\overline{\eta}_{\max}$ for each randomly directed copy. One does not expect the hub locations to be localized in any meaningful sense for random data, but let us collect the hubs anyway.

Given an observed result, one obtained with the catalog data, or another value that arises in some other way, the significance is found simply by counting how many better random results there are. Divide the number of better results by the total number of random runs give an estimate of the significance of the observed or the other result.

The QSO sample here has significances for alignment and avoidance that are about :p = 0.01 and p = 0.05, respectively. Thus with a batch of, say, 10,000 random runs, 100 and 500, resp. random runs will be better. There is no need to estimate the significances by fitting curves to distributions.

Fitting curves to the distributions of random run results becomes necessary when the significances are too small of a fraction. With a small fraction, the total number of random runs must be large in order to have an accurate estimate of the number of better runs. Here the fractions 0.01, one out of a hundred, and 0.05, one in twenty, allow us to be satisfied making tens of thousands of random runs. We can simply count the number of better random runs to find the significance of the observed results.

Getting error bars for the statistical process requires some effort. To get error bars on the process, we create a large number, say 25,000, random runs. Then we choose a large subset of the 25,000 total, say 10,000 random runs in a “batch”. For each batch, we get the number of random runs that are better than the observed data and determine the significance of the observed results according to the random runs in each batch.

Collecting a large number of similarly constructed batches, we get a large number of significances. That gives a distribution of significances with a peak and half-width. That is the way we get plus/minus values for the significances. The error bars on significance due to statistics are much smaller than the error bars from experimental uncertainty $\sigma\psi$ in the polarization directions. Perhaps we could have made do with fewer random runs. We keep what we have.

A11a. Generate new sets of random runs or retrieve old sets

Generating random ψ runs

To generate random runs use cells that are commented out: (* comments are not processed by Mathematica*).
Notes: 1. Rand, rand are short for “random” 2. Hmin = Halign 3. Hmax = Havoid

Definitions:

nRunTotal number of random runs to be generated

ηAlignR sorted values of smallest alignment angle $\overline{\eta}$(Hj) from random runs
ηAvoidR sorted largest alignment angles $\overline{\eta}$(Hj) from random runs
ηBarminR random run values of $\overline{\eta}_{\min}$, in order of random runs
ηBarmaxR random run values of $\overline{\eta}_{\max}$
(ℓHalignR, bHalignR) random run values of (ℓ,b) for hub H_{align}
(ℓHavoidR, bHavoidR) random run values of (ℓ,b) for hub H_{avoid}
rHalignR, rHavoidR, rCrossHHR radial unit vectors to hubs H_{align} and H_{avoid} and the HxH axis, one for each random run
ℓCrossHHR galactic longitude ℓ at HxH on Celestial Sphere

*b*CrossHHR galactic latitude *b* at HxH on Celestial Sphere
θHalignHavoidR random run values of the angle between hubs H_{align} and H_{avoid}

random run values converted to the Galactic Coordinate System:
randomRunsHub
1. run # 2. radial unit vector to H_{align} 3. longitude ℓ of H_{align} 4. latitude *b* of H_{align} 5. radial unit vector to H_{avoid} 6. longitude ℓ of H_{avoid} 7. latitude *b* of H_{avoid} 9. radial unit vector to HxH 10. longitude ℓ of HxH 11. latitude *b* of HxH 12. Total number of random runs
rank Given a set of random runs, 'rank' is the number of random runs with a better value of the quantity.

In[446]:=
```
(*Generate new sets of random runs*)

(*Definitions:

 rSrcxrGrid  unit vector perpendicular to the plane containing the origin,
the source Si and a grid point Hj
 nRunTotal   the number of random runs to be generated (USER)
 printIntervalR   number of runs between printing (USER)
  nRunPrint   counter to control printing
  dθ1radians    the grid spacing in radians
  ψSrcRand     random PPA angles ψ, in radians
  rSrcxψSrc    unit vector perpendicular to the plane containing the origin,
the source Si and the random PPA direction ψSrcRand
  ηψSrcToGrid    the alignment angle ηiH in radians between the direction
 ψSrcRand and the direction toward Hj in the tangent space of the source
  jηBarToGrid
 {1. grid point # for Hj 2. average alignment angle function η̄(Hj), in degrees}
   sortjηBarToGrid    sorted, smallest angle first
 ℓbηBarAlignRandψ    (ℓ,b) at Halign and smallest η̄(Hj)
  ℓbηBarAvoidRandψ   (ℓ,b) at Havoid and largest η̄(Hj)
  randRuns    output: 1. run # 2abc. ℓ,b,
η̄ at smallest η̄(Hj) 3abc. ℓ,b,η̄ at largest η̄(Hj)
*)
(*Calculate rSrcxrGrid table*)
(*
t3=TimeUsed[]
  rSrcxrGrid1 =Table[ Cross[  ri[i],rHj[j]  ] ,    {i,nSrc},{j,nGrid}]
   (*first step: raw cross product, not unit vectors*);
rSrcxrGrid=Table[ rSrcxrGrid1[[i,j]]/
   (rSrcxrGrid1[[i,j]].rSrcxrGrid1[[i,j]]+ 0.00000001)^1/2. ,    {i,nSrc},{j,nGrid}];
t4=TimeUsed[]
   t4-t3
*)
```

```
In[447]:= (*Generate new sets of random runs*)
(*
nRunTotal=200; (*USER sets the number of random runs*)
printIntervalR = 20; (*USER sets the printing interval*)
randRuns={};(*initially empty*)
nRunPrint=0;
dθ1radians=dθ1(2.π/360.)(*the grid spacing in radians*)

   t1=TimeUsed[]
   For[nRun=1,nRun≤nRunTotal,nRun++,
     If[nRun>nRunPrint,Print["At the start of run ",nRun,", the time is ",
      TimeUsed[]," seconds and the memory in use is ",MemoryInUse[]," bytes."];
     nRunPrint=nRunPrint+printIntervalR];
    (*Random PPA angles ψ in radians:*)
    ψSrcRand=Table[RandomReal[{0.001,π-0.001}],{i,nSrc}];
    (*the unit vector cross product of rSrc and the direction of ψSrcRand:*)
    rSrcxψSrc = Table[ Sin[ψSrcRand[[i]]]vNi[i]-Cos[ψSrcRand[[i]]]vEi[i],    {i,nSrc}];
    (*the alignment angle η_iH in radians:
    *)
    ηψSrcToGrid=Table[
      ArcCos[ Min[1., Abs[  rSrcxψSrc[[i]].rSrcxrGrid[[i,j]]  ]]   ],{j,nGrid} , {i,nSrc}];
    (*{ j, grid point #,  η̄(Hj) main alignment function at Hj, in degrees}:*)
    jηBarToGrid = Table[{j,(360./2.π)(1/nSrc)Total[ηψSrcToGrid[[j]]]},{j,nGrid}];
    (*OPTIONAL REPLACEMENT STEP:*)
    (*jηBarToGrid = Table[{j,(1/nSrc)Sum[ ArcCos[  Abs[  rSrcxψSrc[[i]].
              rSrcxrGrid[[i,j]]  ] - 0.000001  ],{i,nSrc}]},{j,nGrid}];*)
     (*same as jηBarToGrid, {j,η̄(Hj)},
    but sorted with the smallest alignment angles first:
    *)
    sortjηBarToGrid=Sort[jηBarToGrid,#1[[2]]<#2[[2]]&];
    (*  {ℓ,b,η̄} at smallest η̄(Hj)*)
    ℓbηBarAlignRandψ=
     {ℓHj[sortjηBarToGrid[[1,1]]],bHj[sortjηBarToGrid[[1,1]]],sortjηBarToGrid[[1,2]]};
    (*  {ℓ,b,η̄} at largest η̄(Hj)*)
    ℓbηBarAvoidRandψ=
     {ℓHj[sortjηBarToGrid[[-1,1]]],bHj[sortjηBarToGrid[[-1,1]]],sortjηBarToGrid[[-1,2]]};
    (*collect data: {run #, {2abc. ℓ,b,η̄ at smallest η̄(Hj)}
        {3abc. ℓ,b,η̄ at largest η̄(Hj)} }:*)
    AppendTo[randRuns,{nRun,ℓbηBarAlignRandψ,ℓbηBarAvoidRandψ} ]  ]  ;
t2=TimeUsed[]
   t2-t1
*)
```

In[448]:=
```
(*Save and/or retrieve random runs from a file.*)
(*A link to the data file "20230703randRuns2000.dat" can be found in Ref. 8.*)
SetDirectory[NotebookDirectory[]]
(*Put[randRuns,"20230701randRuns200.dat"]*)
randRuns = Get["20230703randRuns2000.dat"];
```

Out[448]=
```
C:\Users\momen\Dropbox\HOME_DESKTOP-0MRE5OJ\SendXXX_CJP_CEJPetc\
  SendViXra\20200715AlignmentMethod\20200715AlignmentMMAnotebooks\StarterKit
```

In[450]:=
```
ηAlignR = Sort[Table[randRuns[[i, 2, 3]], {i, Length[randRuns]}]];
ηAvoidR = Sort[Table[randRuns[[i, 3, 3]], {i, Length[randRuns]}]];
```

There are 2000 random runs.

The random run alignment angles ηAlignR range from 38.3337° to 43.7265°.

The random run avoidance angles ηAvoidR range from 46.1859° to 51.8967°.

A noteworthy significance is $p = 0.05 = 1/20$.
So we get some values that rank 1/20th among all random runs.

For 2000 random runs, the rank at 1/20th is 2000/20 = 100. random runs.

The alignment angle ηAlignR is 'significant', $p < 0.05$, for ηAlignR < 40.5817°.

The avoidance angle ηAvoidR is 'significant', $p < 0.05$, for ηAvoidR > 49.5534°.

In[459]:=
```
ηBarminR = Table[randRuns[[i1, 2, 3]], {i1, Length[randRuns]}];
ηBarmaxR = Table[randRuns[[i1, 3, 3]], {i1, Length[randRuns]}];
```

In[461]:=
```
ℓHalignR = Table[randRuns[[i1, 2, 1]], {i1, Length[randRuns]}];
bHalignR = Table[randRuns[[i1, 2, 2]], {i1, Length[randRuns]}];
ℓHavoidR = Table[randRuns[[i1, 3, 1]], {i1, Length[randRuns]}];
bHavoidR = Table[randRuns[[i1, 3, 2]], {i1, Length[randRuns]}];
```

In[465]:=
```
ListPlot[Sort[ℓHalignR], PlotRange → All];
ListPlot[Sort[bHalignR], PlotRange → All];
ListPlot[Sort[ℓHavoidR], PlotRange → All];
ListPlot[Sort[bHavoidR], PlotRange → All];
```

In[469]:=
```
rHalignR = Table[er[ℓHalignR[[i1]], bHalignR[[i1]]], {i1, Length[randRuns]}];
rHavoidR = Table[er[ℓHavoidR[[i1]], bHavoidR[[i1]]], {i1, Length[randRuns]}];
rCrossHH0 = Table[Cross[rHalignR[[i1]], rHavoidR[[i1]]], {i1, Length[randRuns]}];
rCrossHHR =
  Table[rCrossHH0[[i1]] / (rCrossHH0[[i1]].rCrossHH0[[i1]])^(1/2), {i1, Length[randRuns]}];
ℓCrossHHR = Table[ℓFROMr[rCrossHHR[[i1]]], {i1, Length[randRuns]}];
bCrossHHR = Table[bFROMr[rCrossHHR[[i1]]], {i1, Length[randRuns]}];
```

In[475]:=
```
θHalignHavoidR = Table[arccos[rHalignR[[i1]].rHavoidR[[i1]]], {i1, Length[randRuns]}];
```

```
In[476]:= randomRunsHub = Table[{i1, rHalignR[[i1]], ℓHalignR[[i1]],
      bHalignR[[i1]], rHavoidR[[i1]], ℓHavoidR[[i1]], bHavoidR[[i1]], rCrossHHR[[i1]],
      ℓCrossHHR[[i1]], bCrossHHR[[i1]], Length[randRuns]}, {i1, Length[randRuns]}];
   randomRunsHub[[1]];
   randomRunsHub[[-1]];
```

A11b. Significance of many observed quantities

There are many quantities that are calculated from the data which we call ‘observables’. The observables are organized in a list, ‘observedQuantities’, with a list of their values, ‘valuesOfObservedQuantities’. For each batch of random runs, described above, we find the ranks of the observables. The significance of the observable is the rank divided by the number of random runs in the batch. The significances are collected and displayed in a table, Table A11.1 *The significances of several quantities*.

Definitions:

observedQuantities a list of the quantities: alignment angle $\overline{\eta}_{\text{min}}$ using best ψ, ...
valuesOfObservedQuantities values of the quantities: $\overline{\eta}_{\text{min}}$ = 39.72°, ...
nRunTotal total number of all random runs
nRunsPerBatch the number of random runs in one batch
nBatchTotal the number of batches
IDsRunsThisBatch list of random run ID #s for a batch, chosen randomly for each
randRunsBatch the randRuns data for the batch
sortQ sort the randRuns data for the quantities Q = ηBarmin, ηBarmax, ...
rank Given a set of random runs, ‘rank’ is the number of random runs with a better value of the quantity.
idForRankOfQ the rank of Q in the batch
ranksOfQuantitiesBatch collect the ranks of observables in one batch of random runs
batchRunsOutput the set of ‘ranksOfQuantitiesBatch’ for all batches
rank[q] mean value of the rank for the q^{th} observable, averaged over all batches
signifOfηBarAlign[q] significance of the q^{th} observable = rank/runs per batch
stanDevOfSignifOfηBarAlign[q] statistical uncertainty of the significance of the q^{th} observable
sigTable display table of the significances

```
In[479]:= observedQuantities = {"using best ψ, best η̄min ", "uncertainty fit, most likely η̄min ",
      " uncertainty fit, η̄min - ση ", " uncertainty fit, η̄min + ση",
      "using best ψ, best η̄max ", "uncertainty fit, most likely η̄max ",
      " uncertainty fit, η̄max - ση ", " uncertainty fit, η̄max + ση",
      "best arc Halign to Havoid", "UFit Likely arc Halign to Havoid",
      "UFit Small arc Halign to Havoid", "UFit Big arc Halign to Havoid"};
   Length[%];

In[481]:= valuesOfObservedQuantities = {ηBarHbestAlign, ηBarminUFit, ηBarminUFit - σηBarminUFit,
      ηBarminUFit + σηBarminUFit, ηBarHbestAvoid, ηBarmaxFitU, ηBarmaxFitU - σηBarmaxFitU,
      ηBarmaxFitU + σηBarmaxFitU, Abs[bestθHToH], Abs[θHalignHavoidUlikely],
      90., Abs[(θHalignHavoidUlikely + σθHalignHavoidUlikely)]};
   Length[%];

In[483]:= nRunTotal = Length[randRuns];
   nRunsPerBatch = Round[0.6 nRunTotal];

   There are a total of 2000 random runs.

   The number of random runs in one batch is 1200 per batch.

In[487]:= (*Find the rank of observables in the random run results.*)
   nBatchTotal = Round[Length[randRuns]^(1/2)]; iBatchPrint = 0;
```

```
batchRunsOutput = {};
For[iBatch = 1, iBatch ≤ nBatchTotal, iBatch++,
 If[iBatch > iBatchPrint, (*Print["At the start of run ",iBatch,
    ", the time is ",TimeUsed[]," seconds and the memory in use is ",
    MemoryInUse[]," bytes."];*)iBatchPrint = iBatchPrint + 5];
 IDsRunsThisBatch = RandomSample[Range[nRunTotal], nRunsPerBatch];
 randRunsBatch =
  Table[randRuns⟦i⟧, {i, RandomSample[Range[nRunTotal], Round[nRunsPerBatch]]}];
 (*η̄minS*)
 sortηBarmin = Sort[Table[ηBarminR⟦i1⟧ , {i1, IDsRunsThisBatch}]];
 idForRankOfBestηAlignR = 1;
 For[i = 1, i ≤ Length[sortηBarmin], i++,
  If[sortηBarmin⟦i⟧ ≤ ηBarHbestAlign, idForRankOfBestηAlignR = i]];
 idForRankOfLikelyηAlignRUFit = 1;
 For[i = 1, i ≤ Length[sortηBarmin], i++,
  If[sortηBarmin⟦i⟧ ≤ ηBarminUFit, idForRankOfLikelyηAlignRUFit = i]];
 idForRankOfSmallηAlignRUFit = 1;
 For[i = 1, i ≤ Length[sortηBarmin], i++,
  If[sortηBarmin⟦i⟧ ≤ ηBarminUFit - σηBarminUFit, idForRankOfSmallηAlignRUFit = i]];
 idForRankOfBigηAlignRUFit = 1;
 For[i = 1, i ≤ Length[sortηBarmin], i++,
  If[sortηBarmin⟦i⟧ ≤ ηBarminUFit + σηBarminUFit, idForRankOfBigηAlignRUFit = i]];
 (*η̄maxS*)
 sortηBarmax = Sort[Table[ηBarmaxR⟦i1⟧ , {i1, IDsRunsThisBatch}], Greater];
 idForRankOfBestηAvoidR = 1;
 For[i = 1, i ≤ Length[sortηBarmax], i++,
  If[sortηBarmax⟦i⟧ ≥ ηBarHbestAvoid, idForRankOfBestηAvoidR = i, Continue[]]];
 idForRankOfLikelyηAvoidRUFit = 1;
 For[i = 1, i ≤ Length[sortηBarmax], i++,
  If[sortηBarmax⟦i⟧ ≥ ηBarmaxFitU, idForRankOfLikelyηAvoidRUFit = i, Continue[]]];
 idForRankOfSmallηAvoidRUFit = 1;
 For[i = 1, i ≤ Length[sortηBarmax], i++, If[sortηBarmax⟦i⟧ ≥ ηBarmaxFitU - σηBarmaxFitU,
   idForRankOfSmallηAvoidRUFit = i, Continue[]]];
 idForRankOfBigηAvoidRUFit = 1;
 For[i = 1, i ≤ Length[sortηBarmax], i++, If[sortηBarmax⟦i⟧ ≥ ηBarmaxFitU + σηBarmaxFitU,
   idForRankOfBigηAvoidRUFit = i, Continue[]]];
 (*arc Halign to Havoid*)
 arc90MinusθHToH = Table[Abs[90. - θHalignHavoidR⟦i1⟧ ], {i1, IDsRunsThisBatch}];
 sortarc90MinusθHToH = Sort[arc90MinusθHToH];
 idForRankOfBestarc90MinusθHToH = 1;
 For[i = 1, i ≤ Length[sortarc90MinusθHToH],
  i++, If[sortarc90MinusθHToH⟦i⟧ ≤ Abs[90. - bestθHToH],
   idForRankOfBestarc90MinusθHToH = i, Continue[]]];
 idForRankOfLikelyarc90MinusθHalignHavoidRUlikely = 1;
 For[i = 1, i ≤ Length[sortarc90MinusθHToH], i++,
  If[sortarc90MinusθHToH⟦i⟧ ≤ Abs[90. - θHalignHavoidUlikely],
   idForRankOfLikelyarc90MinusθHalignHavoidRUlikely = i, Continue[]]];
```

```
 (*With the most likely θ = 92.3° and σθ = 2.6°,
 a perfect θ = 90° is in range on the Small side. So take 0 for θSmall = θ - σθ.*)
 idForRankOfSmallarc90MinusθHalignHavoidRUlikely = 1;
 (*For[i=1,i≤Length[sortarc90MinusθHToH],i++,If[sortarc90MinusθHToH[[i]]≤Abs[0.],
    idForRankOfSmallarc90MinusθHalignHavoidRUlikely=i,Continue[]]];*)
 idForRankOfBigarc90MinusθHalignHavoidRUlikely = 1;
 For[i = 1, i ≤ Length[sortarc90MinusθHToH], i++,
  If[sortarc90MinusθHToH[[i]] ≤ Abs[90. - (θHalignHavoidUlikely + σθHalignHavoidUlikely)],
   idForRankOfBigarc90MinusθHalignHavoidRUlikely = i, Continue[]]];
 ranksOfQuantitiesBatch = Flatten[
   {{idForRankOfBestηAlignR, idForRankOfLikelyηAlignRUFit, idForRankOfSmallηAlignRUFit,
     idForRankOfBigηAlignRUFit}, {idForRankOfBestηAvoidR, idForRankOfLikelyηAvoidRUFit,
     idForRankOfSmallηAvoidRUFit, idForRankOfBigηAvoidRUFit},
    {idForRankOfBestarc90MinusθHToH, idForRankOfLikelyarc90MinusθHalignHavoidRUlikely,
     idForRankOfSmallarc90MinusθHalignHavoidRUlikely,
     idForRankOfBigarc90MinusθHalignHavoidRUlikely}}];
 AppendTo[batchRunsOutput, Flatten[{ranksOfQuantitiesBatch, iBatch}]]
]
```

The number of batches, nBatchTotal, is the square root of the number of random runs, nRunTotal.

We have $nRunTotal^{1/2}$ = 44.72, which gives nBatchTotal = 45.

In[492]:=
```
batchRunsOutput;
Length[batchRunsOutput];
Length[batchRunsOutput[[1]]];
batchRunsOutput[[1]];
```

There are 'nBatchTotal' = 45 batches.

The data for a batch is recorded in 'batchRunsOutput'
and consists of the ranks of observed quantities in the batch.

Each batch has 1200 random runs selected at random from a pool of 2000 total random runs.

Thus, the same random run likely appears in
many batches, so the ranks are not independent from batch to batch.

On average, a given random run should appear in a fraction, 1200/2000 = 0.6, of batches.

This should be borne in mind when considering
the associated uncertainties in significances found by this method.

In[502]:=
```
observedQuantities[[5]]; binWidth = 0.005;
N[Sort[Table[batchRunsOutput[[i, 5]], {i, Length[batchRunsOutput]}] / nRunsPerBatch]];
histEtaMaxRandom = Histogram[%, {binWidth}, PlotRange → {{0.0, 0.12}, Automatic},
   PlotLabel → " Random runs distribution of the significance of best η̄max ",
   FrameLabel → {"significance", "ΔR"}, PlotTheme → "Detailed"];
```

Out[505]=

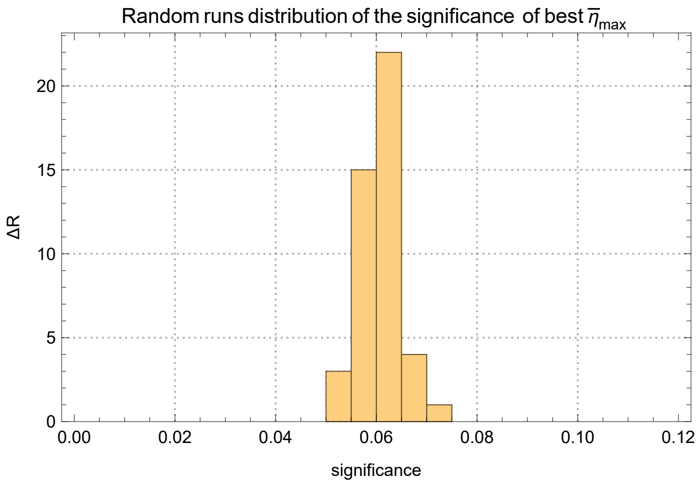


Figure 11.1. An example of a distribution of significance over batches of random runs. The number ΔR is the number of batches that have significance $\bar{\eta}_{max}$ in a bins of width $\Delta\bar{\eta}$ = 0.005. The total number of batches is R = ΣΔR = 45.

In[507]:=
```
rank[q_] := N[mean[Table[batchRunsOutput[[i, q ]], {i, Length[batchRunsOutput]}]]]
signifOfηBarAlign[q_] := rank[q] / nRunsPerBatch
stanDevOfSignifOfηBarAlign[q_] :=
 N[stanDev[Table[batchRunsOutput[[i, q ]], {i, Length[batchRunsOutput]}]]] / nRunsPerBatch
```

In[735]:=
```
sigTable = Text@Grid[
    Prepend[Table[{observedQuantities[[q]], Round[10^2 valuesOfObservedQuantities[[q]]] / 10.^2,
       rank[q], nRunsPerBatch, Round[10^4 signifOfηBarAlign[q]] / 10.^4,
       Round[10^4 stanDevOfSignifOfηBarAlign[q]] / 10.^4,
       Round[10^2 signifOfηBarAlign[q]^-1] / 10.^2}, {q, 12}],
     {"quantity", "value, deg", "average rank", "runs/batch", " p ", " ±σp ", "1/p"}],
    Spacings → {1, 1.}, Dividers → Center];
```

Out[736]=

quantity	value, deg	average rank	runs/batch	p	$\pm\sigma p$	1/p
using best ψ, best $\overline{\eta}_{\text{min}}$	39.71	11.1111	1200	0.0093	0.0013	108.
uncertainty fit, most likely $\overline{\eta}_{\text{min}}$	39.92	14.6444	1200	0.0122	0.0016	81.94
uncertainty fit, $\overline{\eta}_{\text{min}} - \sigma\eta$	39.53	6.04444	1200	0.005	0.0012	198.53
uncertainty fit, $\overline{\eta}_{\text{min}} + \sigma\eta$	40.31	32.8889	1200	0.0274	0.0029	36.49
using best ψ, best $\overline{\eta}_{\text{max}}$	49.47	72.4667	1200	0.0604	0.0042	16.56
uncertainty fit, most likely $\overline{\eta}_{\text{max}}$	49.27	103.6	1200	0.0863	0.0049	11.58
uncertainty fit, $\overline{\eta}_{\text{max}} - \sigma\eta$	48.88	195.333	1200	0.1628	0.0064	6.14
uncertainty fit, $\overline{\eta}_{\text{max}} + \sigma\eta$	49.65	48.6222	1200	0.0405	0.0038	24.68
best arc Halign to Havoid	93.58	106.867	1200	0.0891	0.006	11.23
UFit Likely arc Halign to Havoid	92.39	72.3111	1200	0.0603	0.005	16.59
UFit Small arc Halign to Havoid	90.	1.	1200	0.0008	0.	1200.
UFit Big arc Halign to Havoid	94.85	134.711	1200	0.1123	0.0063	8.91

Table A11.1: The significances of several quantities. The average rank of a value of the quantity Q is the average number of better results from random runs in the batches. The average rank is in column 3 and the value of Q is in column 2. The fraction runs/batch is the average rank divided by the number of random runs in a batch. That makes the significance of Q in column 5. The standard deviation σp is taken over the significances found for individual batches, making column 6. Column 7 has the inverse p^{-1}, the number of random runs in a set on average before one of the random runs has a better result than the value Q considered.

A11c. The measured ψ puts the angle θ from H_{align} to H_{avoid} at about 90°. Would random ψ do that?

Definitions:

sortθHToH sort the random run values of the angle θ from H_{align} to H_{avoid}

meanθHToH arithmetic average of the angle θ from H_{align} to H_{avoid} for all random runs

stanDevθHToH standard deviation of the values for θ from H_{align} to H_{avoid} for all random runs

histogramrangeθ parameters for histograms

hlθ histogram tables, {quantity, bin height}, needed to set up the NonlinearModelFit (nlm)

nlmθHToH NonlinearModelFit (nlm), fit to the random run values of the angle θ from H_{align} to H_{avoid}

pTableNLMθ parameter table for the fit

showθHtoHRandom histogram of quantity angle θ from H_{align} to H_{avoid} for all random runs with Gaussian fit using mean for the peak and standard deviation for the half-width

sortarc90minusθHToHrandom sort |90° - θ| for all random runs

idForRankOfLikelyarcθHalignHavoidUlikely rank of uncertainty run fit most likely value |90° - θ| in set of all random runs

In[513]:=
```
sortθHToH = Sort[θHalignHavoidR];
meanθHToH = mean[θHalignHavoidR ]; (*Guess the mean for the Gaussian. *)
stanDevθHToH = stanDev[θHalignHavoidR ];(*Guess the half-width.*)
histogramrangeθ =
  {meanθHToH - 5 stanDevθHToH, meanθHToH + 5 stanDevθHToH, 0.4 stanDevθHToH};
hl0θ = HistogramList[sortθHToH, histogramrangeθ];
hlθ =
  Table[{(1 / 2) (hl0θ[[1, i1]] + hl0θ[[1, i1 + 1]]), hl0θ[[2, i1]]}, {i1, Length[ hl0θ[[2]] ]}];
nlmθHToH = NonlinearModelFit[hlθ, {a Exp[- 1/2. ((x - x0)/b)^2] (*,b>0*)},
   {{a, Length[sortθHToH] / 6}, {b, stanDevθHToH}, {x0, meanθHToH}}, x];
```

In[520]:=
```
pTableNLMθ = nlmθHToH["ParameterTable"] ;
{σθHToHlikely, θHToHlikely} = ParametersNLMθ = {b, x0} /. nlmθHToH["BestFitParameters"];
(*degrees*)
```

In[522]:=
```
showθHtoHRandom = Show[{Histogram[sortθHToH, histogramrangeθ,
     PlotLabel → "Distribution of θ from H_Align to H_Avoid,  over all random runs ",
     FrameLabel → {"θ, degrees", "ΔR"}, PlotTheme → "Detailed"],
    Plot[Normal[nlmθHToH], {x, meanθHToH - 5 stanDevθHToH, meanθHToH + 5 stanDevθHToH},
     PlotLabel → "θ between H_Align and H_Avoid "] ,
    ListPlot[hlθ, PlotLabel → "θ between H_Align and H_Avoid "]}];
```

Out[523]=

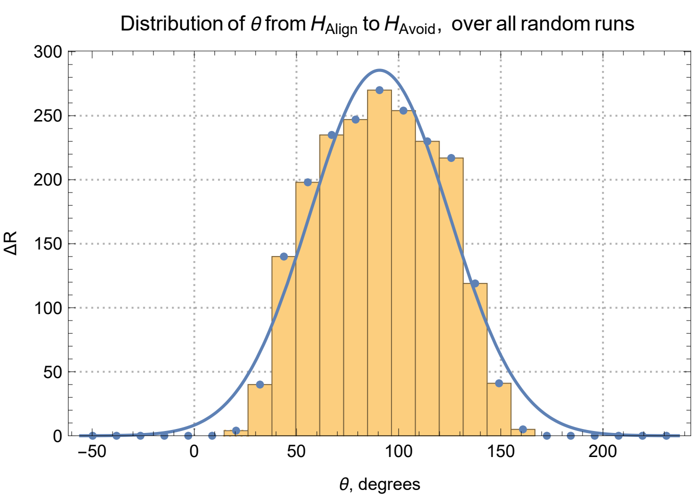


Figure A11.2: The histogram for the angle θ between H_{Align} and H_{Avoid} and its Gaussian fit. The fit for random data gives θ = 90.6713° ± 34.0871 °. Recall that, for the Uncertainty run distribution of θ in Fig. A12.1, we found θ = 92.3868° ± 2.46751 °, which has almost the same peak as the random run distribution. But the width due to experimental uncertainty is much narrower than the random runs produce.

In[525]:=
```
sortarc90minusθHToHrandom =
  Sort[Table[Abs[90 - θHalignHavoidR[[i1]] ], {i1, Length[randRuns]}]];
idForRankOfLikelyarcθHalignHavoidRUlikely = 1;
For[i = 1, i ≤ Length[sortarc90minusθHToHrandom], i++,
   If[sortarc90minusθHToHrandom[[i]] ≤ Abs[90. - θHalignHavoidUlikely],
    idForRankOfLikelyarcθHalignHavoidRUlikely = i, Continue[]]];
idForRankOfLikelyarcθHalignHavoidRUlikely;
N[idForRankOfLikelyarcθHalignHavoidRUlikely / Length[randRuns]];
N[(idForRankOfLikelyarcθHalignHavoidRUlikely / Length[randRuns])^-1];
```

The observed data gives the angle θ from H_{align} to H_{avoid}, θ = 92.3868°. The number of random values in Fig. A13.2 that are closer is 120 out of 2000 random runs, so the significance of the observed θ is p = 0.06.

That means that one of 16.67 random runs produce an arc θ from H_{align} to H_{avoid} that is closer to 90° than the observed value.

Random ψ do put the average angle θ from H_{align} to H_{avoid} at 90°, but random ψ are less likely to do so than the observed ψ.

After Sec. A11, time and memory used are 549.312 seconds and 1 891 441 464 bytes.

A12. Plot the data, the sources and their polarization directions
A12a. Plot a close-up of a portion of the data

Definitions

xyAitoffSources Coords of sources in Aitoff projection
xyAitoffSourcesShadedψ Coords of sources shaded with PPA ψ

line segments that represent polarization directions ψ:
rPlusψ endpoints of line segments
crossesOverPlus, *Minus, *noCrossing ID#s of sources with segments that cross the Aitoff plot edge or don’t cross it
polarLinesNoCrossing1[d],*CrossingPlus,*CrossingMinus the line segments
vψSrc, vψSrcBig, Small unit vectors, v(ψ), v($\psi \pm \sigma\psi$), large & small, the best and the one-sigma range of polarization directions ψ
polarLines, polarLinesBig, polarLinesSmall [d] Line segments of length d to represent polarization directions, ψ and $\psi \pm \sigma\psi$
rPlusψBig, Small radial unit vector to endpoints of polarLinesBig, polarLinesSmall [d]

hubs:
xyAitoffHminU, xyAitoffHmaxU, xyAitoffOppositeHminU, xyAitoffOppositeHmaxU Aitoff coords for hubs H_{align}, H_{avoid}, $-H_{align}$, $-H_{avoid}$
listCPlocalU Local plot of data, (i) sources with (ii) polarization directions $\psi \pm \sigma\psi$ drawn

In[537]:=
```
(*Coords of sources in Aitoff projection*)
xyAitoffSources = Table[{xHGal[ℓi[i], bi[i] ], yHGal[ ℓi[i], bi[i] ]}, {i, nSrc}];
```

In[538]:=
```
(*Coords of sources in Aitoff projection, shaded with PA ψ*)
xyAitoffSourcesShadedψ = Table[{ColorData["Rainbow"][(ψi[i]) / 180.],
    Point[{xHGal[ℓi[i], bi[i] ], yHGal[ ℓi[i], bi[i] ]}]}, {i, nSrc}];
```

In[539]:=
```
(*Prepare to plot polarization directions*)
rPlusψ[i_, d_] := (*rPlusψ[i,d]=*)
 (ri[i] + d vψi[i]) / ((ri[i] + d vψi[i]).(ri[i] + d vψi[i]))^1/2
crossesOverPlus = {}; crossesOverMinus = {};
For[i = 1, i ≤ nSrc, i++,
 If[ ℓFROMr[ rPlusψ[i, 0.08]] - ℓi[i] < -200., AppendTo[crossesOverPlus, i]];
 If[  ℓFROMr[ rPlusψ[i, -0.08]] - ℓi[i] > 20., AppendTo[crossesOverMinus, i]]]
noCrossing = Complement[Range[nSrc], Union[crossesOverPlus, crossesOverMinus]];
```

In[543]:=
```
(*Prepare to plot transverse directions*)
polarLinesNoCrossing1[d_] := (*polarLinesNoCrossing1[d]=*)
 Table[{ColorData["Rainbow"][(ψi[i]) / 180.], Line[
    {{xHGal[ℓFROMr[ rPlusψ[i, d]], bFROMr[ rPlusψ[i, d]] ], yHGal[ℓFROMr[ rPlusψ[i, d]],
       bFROMr[ rPlusψ[i, d]]]}, {xHGal[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]],
      yHGal[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]]}}]}, {i, noCrossing}]
```

In[544]:=
```
polarLinesCrossingPlus1[d_] := (*polarLinesCrossingPlus1[d]=*)Table[
  {ColorData["Rainbow"][(ψi[i]) / 180.], Line[{{xHGal[ℓi[i], bi[i]], yHGal[ℓi[i], bi[i]]},
     {xHGal[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]],
      yHGal[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]]}}]}, {i, crossesOverPlus}]
```

In[545]:=
```
polarLinesCrossingMinus1[d_] := (*polarLinesCrossingMinus1[d]=*)
 Table[{ColorData["Rainbow"][(ψi[i]) / 180.],
   Line[{{xHGal[ℓFROMr[ rPlusψ[i, d]], bFROMr[ rPlusψ[i, d]]],
      yHGal[ℓFROMr[ rPlusψ[i, d]], bFROMr[ rPlusψ[i, d]]]}, {xHGal[ℓi[i], bi[i]],
      yHGal[ℓi[i], bi[i]]}}]}, {i, crossesOverMinus(*noCrossing*)}]
```

In[546]:=
```
(*The Aitoff coordinates for the hubs H_align locations.*)
xyAitoffHminU = Table[{xHGal[ HminℓrunDataU[[n]], HminbrunDataU[[n]] ],
    yHGal[  HminℓrunDataU[[n]], HminbrunDataU[[n]] ]}, {n, Length[HminbrunDataU ]}];
```

In[547]:=
```
(*The Aitoff coordinates for the hubs H_avoid locations.*)
xyAitoffHmaxU = Table[{xHGal[  HmaxℓrunDataU[[n]], HmaxbrunDataU[[n]] ],
    yHGal[  HmaxℓrunDataU[[n]], HmaxbrunDataU[[n]] ]}, {n, Length[HmaxbrunDataU ]}];
```

In[548]:=
```
(*The Aitoff coordinates for the hubs -H_align locations.*)
xyAitoffOppositeHminU = Table[{xHGal[ If[0 ≤ HminℓrunDataU[[n]] < +180, HminℓrunDataU[[n]] - 180,
      If[0 > HminℓrunDataU[[n]] > -180, HminℓrunDataU[[n]] + 180]], -HminbrunDataU[[n]] ],
    yHGal[ If[0 ≤ HminℓrunDataU[[n]] < +180, HminℓrunDataU[[n]] - 180,
      If[0 > HminℓrunDataU[[n]] > -180, HminℓrunDataU[[n]] + 180]],
     -HminbrunDataU[[n]]]}, {n, Length[HminbrunDataU ]}];
```

In[549]:=
```
(*The Aitoff coordinates for the hubs -Havoid locations.*)
xyAitoffOppositeHmaxU =
  Table[{xHGal[ If[0 ≤ Hmaxℓ runDataU⟦n⟧ < +180, Hmaxℓ runDataU⟦n⟧ + 180,
      If[0 > Hmaxℓ runDataU⟦n⟧ > -180, Hmaxℓ runDataU⟦n⟧ + 180]], -Hmaxb runDataU⟦n⟧ ],
    yHGal[ If[0 ≤ Hmaxℓ runDataU⟦n⟧ < +180, Hmaxℓ runDataU⟦n⟧ + 180,
      If[0 > Hmaxℓ runDataU⟦n⟧ > -180, Hmaxℓ runDataU⟦n⟧ + 180]],
     -Hmaxb runDataU⟦n⟧ ]}, {n, Length[Hmaxb runDataU]}];
```

In[550]:=
```
(* vψ unit vectors pointing along the polarization direction ψ,
which has experimental uncertainty. This cell
 includes the plus/minus values of the vψ unit vectors. *)
vψSrc =
 Table[cos[ (ψi[i]) ] × eN[ ℓi[i], bi[i] ] + sin[ (ψi[i]) ] × eE[ ℓi[i], bi[i] ], {i, nSrc}];
vψSrcBig = Table[cos[ (ψi[i] + σψi[i]) ] × eN[ ℓi[i], bi[i] ] +
   sin[ (ψi[i] + σψi[i]) ] × eE[ ℓi[i], bi[i] ], {i, nSrc}];
vψSrcSmall = Table[cos[ (ψi[i] - σψi[i]) ] × eN[ ℓi[i], bi[i] ] +
    sin[ (ψi[i] - σψi[i]) ] × eE[ ℓi[i], bi[i] ], {i, nSrc}];
```

In[552]:=
```
(*Plot polarization directions*)
polarLines[d_] := Table[
  Line[{{xHGal[ℓFROMr[ rPlusψ[i, d]], bFROMr[ rPlusψ[i, d]]], yHGal[ℓFROMr[ rPlusψ[i, d]],
      bFROMr[ rPlusψ[i, d]]]}, {xHGal[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]],
     yHGal[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]]}}], {i, nSrc}]
```

In[553]:=
```
rPlusψBig[i_, d_] :=
 (ri[i] + d vψSrcBig⟦i⟧) / ((ri[i] + d vψSrcBig⟦i⟧).(ri[i] + d vψSrcBig⟦i⟧))^(1/2)
polarLinesBig[d_] := Table[{ColorData["Rainbow"][(ψi[i]) / 180.],
   Line[{{xHGal[ℓFROMr[ rPlusψBig[i, d]], bFROMr[ rPlusψBig[i, d]]],
      yHGal[ℓFROMr[ rPlusψBig[i, d]], bFROMr[ rPlusψBig[i, d]]]},
     {xHGal[ℓFROMr[ rPlusψBig[i, -d]], bFROMr[ rPlusψBig[i, -d]]],
      yHGal[ℓFROMr[ rPlusψBig[i, -d]], bFROMr[ rPlusψBig[i, -d]]]}}]}, {i, noCrossing}]
```

In[555]:=
```
rPlusψSmall[i_, d_] :=
 (ri[i] + d vψSrcSmall⟦i⟧) / ((ri[i] + d vψSrcSmall⟦i⟧).(ri[i] + d vψSrcSmall⟦i⟧))^(1/2)
polarLinesSmall[d_] := Table[{ColorData["Rainbow"][(ψi[i]) / 180.],
   Line[{{xHGal[ℓFROMr[ rPlusψSmall[i, d]], bFROMr[ rPlusψSmall[i, d]]],
      yHGal[ℓFROMr[ rPlusψSmall[i, d]], bFROMr[ rPlusψSmall[i, d]]]},
     {xHGal[ℓFROMr[ rPlusψSmall[i, -d]], bFROMr[ rPlusψSmall[i, -d]]], yHGal[
       ℓFROMr[ rPlusψSmall[i, -d]], bFROMr[ rPlusψSmall[i, -d]]]}}]}, {i, noCrossing}]
```

In[557]:=
```
xyAitoffSourcesForDataMap = Table[{ColorData["Rainbow"][(ψi[i]) / 180.],
    Point[{xHGal[ℓi[i], bi[i] ], yHGal[ ℓi[i], bi[i] ]}]}, {i, nSrc}];
```

In[558]:=
```
listCPlocalU = Show[ {Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {b, -90., 90.},
      PlotStyle → {Black, Thickness[0.002]}, PlotPoints → 60, PlotRange → {{-0.5, 1.}, {0.5, 1.5}},
      Axes → False, Frame → True, FrameLabel → {"ℓ", "b", "Close-Up View"}, FrameTicks → None],
     {ℓ, -180., 180., 30.}], Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {ℓ, -180., 180.},
      PlotStyle → {Black, Thickness[0.002]}, PlotPoints → 60], {b, -60, 60, 30}],
    Graphics[{PointSize[0.009], Gray, {Thick, polarLinesNoCrossing1[0.03]},
      {Thick, polarLinesCrossingPlus1[0.03]}, {Thick, polarLinesCrossingMinus1[0.03]},
      {Thick, polarLinesBig[0.03]}, {Thick, polarLinesSmall[0.03]}, Black, Text[
       StyleForm["ℓ = 0°", FontSize → 14, FontWeight → "Plain"], {xHGal[1., 32.], yHGal[1., 32.]}],
      Text[StyleForm["30°", FontSize → 14, FontWeight → "Plain"], {xHGal[28., 32.],
        yHGal[28., 32.]}], Text[StyleForm["-30°", FontSize → 14, FontWeight → "Plain"],
       {xHGal[-32., 32.], yHGal[-32., 32.]}], Text[StyleForm["-60°", FontSize → 14,
        FontWeight → "Plain"], {xHGal[-62., 32.], yHGal[-62., 32.]}], Text[StyleForm[
        "b = 60°", FontSize → 14, FontWeight → "Plain"], {xHGal[15., 62.], yHGal[15., 62.]}],
      (*Sources S:*)Green, PointSize[0.012], xyAitoffSourcesForDataMap, PointSize[0.01],
      Blue, (*Hmin:*)Point[ xyAitoffHminU ], Gray, PointSize[0.005]
        }],
    ParametricPlot[{xHGal[(HminℓFit + rHminθ[θ] Cos[θ]), (HminbFit + rHminθ[θ] Sin[θ])],
      yHGal[(HminℓFit + rHminθ[θ] Cos[θ]), (HminbFit + rHminθ[θ] Sin[θ])]},
     {θ, 0., 2. π}, PlotStyle → {Orange, Thickness[0.01]}],
    ParametricPlot[{xHGal[(HmaxℓFit + rHmaxθ[θ] Cos[θ]), (HmaxbFit + rHmaxθ[θ] Sin[θ])],
      yHGal[(HmaxℓFit + rHmaxθ[θ] Cos[θ]), (HmaxbFit + rHmaxθ[θ] Sin[θ])]},
     {θ, 0., 2. π}, PlotStyle → {Orange, Thickness[0.005]}, PlotLegends →
      BarLegend[{"Rainbow", {0, 180.}}, LegendLabel → "PPA ψ, deg."]]}, ImageSize → 0.9 × 432   ];
```

Out[559]=

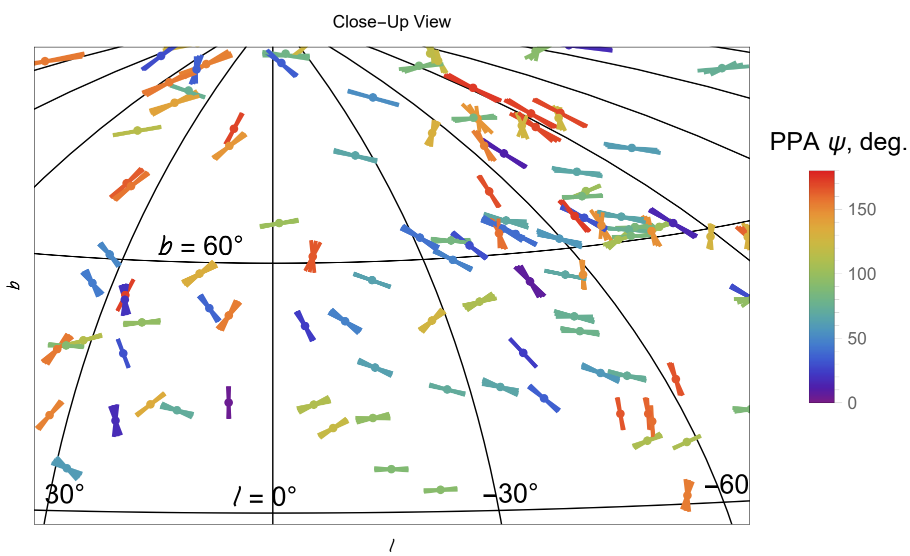


Figure A12.1: A close-up of several sources and their polarization directions with 1σ uncertainties in the Galactic Coordinate System. The polarization directions are shaded by the polarization position angle ψ. For position angles like ψ, ψ = 0 is the same direction as ψ = 180°, ■ = ■.

A12b. Plot all the sources and their polarization directions

Definitions

polarLinesBigToNGP, polarLinesSmallToNGP[d]   Line segments of length d to represent polarization directions, $\psi \pm \sigma\psi$. In Aitoff Galactic coords.

setUPplotForMap   preliminary plot with meridians and parallels and legend
mapOfSources   whole-sky Aitoff plot of sources and polarization directions

In[561]:=
```
polarLinesBigToNGP[d_] := Table[{ColorData["Rainbow"][(ψi[i]) / 180.],
    Line[{{xHGal[ℓFROMr[ rPlusψBig[i, d]], bFROMr[ rPlusψBig[i, d]]],
        yHGal[ℓFROMr[ rPlusψBig[i, d]], bFROMr[ rPlusψBig[i, d]]]},
       {xHGal[ℓFROMr[ rPlusψBig[i, -d]], bFROMr[ rPlusψBig[i, -d]]],
        yHGal[ℓFROMr[ rPlusψBig[i, -d]], bFROMr[ rPlusψBig[i, -d]]]}}]}, {i, noCrossing}]
```

In[562]:=
```
polarLinesSmallToNGP[d_] := Table[{ColorData["Rainbow"][(ψi[i]) / 180.],
    Line[{{xHGal[ℓFROMr[ rPlusψSmall[i, d]], bFROMr[ rPlusψSmall[i, d]]],
        yHGal[ℓFROMr[ rPlusψSmall[i, d]], bFROMr[ rPlusψSmall[i, d]]]},
       {xHGal[ℓFROMr[ rPlusψSmall[i, -d]], bFROMr[ rPlusψSmall[i, -d]]], yHGal[
         ℓFROMr[ rPlusψSmall[i, -d]], bFROMr[ rPlusψSmall[i, -d]]]}}]}, {i, noCrossing}]
```

In[563]:=
```
setUPplotForMap = Show[{ParametricPlot[{xHGal[-120, b], yHGal[-120, b]},
      {b, -90, 90}, PlotStyle → {Black, Thickness[0.002]},
      PlotLegends → BarLegend[{"Rainbow", {0, 180.}}, LegendLabel → "PPA ψ, deg."],
      PlotPoints → 60, PlotRange → {{-4.0, 3.5}, (7.5 / 11.0) {-3, 3}}, Axes → False,
      Frame → False], Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]},
       {b, -90, 90}, PlotStyle → {Black, Thickness[0.002]},
       PlotPoints → 60, PlotRange → {{-4.0, 3.5}, (7.5 / 11.0) {-3, 3}},
       Axes → False, Frame → False], {ℓ, -180., 180., 30}],
     Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {ℓ, -180., 180.},
       PlotStyle → {Black, Thickness[0.002]}, PlotPoints → 60], {b, -60, 60, 30}]}];
```

In[564]:=
```
mapOfSources =
  Show[{setUPplotForMap, Graphics[{PointSize[0.006], Black,
      Text[StyleForm["N", FontSize→14, FontWeight→"Plain"], {0, 1.85}], Black,
      Text[StyleForm["0°", FontSize→14, FontWeight→"Plain"], {xHGal[0., -7.], yHGal[0., -7.]}],
      Text[StyleForm["ℓ=90°", FontSize→14, FontWeight→"Plain"],
       {xHGal[90., -7.], yHGal[90., -7.]}], Text[StyleForm["180°", FontSize→14,
        FontWeight→"Plain"], {xHGal[180., -7.], yHGal[180., -7.]}],
      Text[StyleForm["-90°", FontSize→14, FontWeight→"Plain"], {xHGal[-90., -7.],
        yHGal[-90., -7.]}], Text[StyleForm["-180°", FontSize→14, FontWeight→"Plain"],
       {xHGal[-180., -7.], yHGal[-180., -7.]}], Text[StyleForm["b=+30°", FontSize→14,
        FontWeight→"Plain"], {xHGal[-180., 33.] + 0.2, yHGal[-180., 33.] + 0.1}],
      Text[StyleForm["b=-30°", FontSize→14, FontWeight→"Plain"],
       {xHGal[-180., -33.] + 0.2, yHGal[-180., -33.] - 0.1}],
      Text[StyleForm["b=+60°", FontSize→14, FontWeight→"Plain"],
       {xHGal[-180., 63.] + 0.2, yHGal[-180., 63.] + 0.1}],
      Text[StyleForm["b=-60°", FontSize→14, FontWeight→"Plain"],
       {xHGal[-180., -63.] + 0.2, yHGal[-180., -63.] - 0.1}],
      Text[StyleForm["Galactic Coordinate System", FontSize→14, FontWeight→"Plain"],
       {0, -1.85}], {Thickness[0.003], polarLinesBigToNGP[(*0.095*)0.075]},
      {Thickness[0.003], polarLinesSmallToNGP[(*0.095*)0.075]},
      {Thickness[0.003], polarLinesNoCrossing1[0.075]},
      {Thickness[0.003], polarLinesCrossingPlus1[0.075]}, {Thickness[0.003],
       polarLinesCrossingMinus1[0.075]}, (*Sources S:*) xyAitoffSourcesForDataMap
        }]}, ImageSize→ 1.2×432];
```

Out[565]=

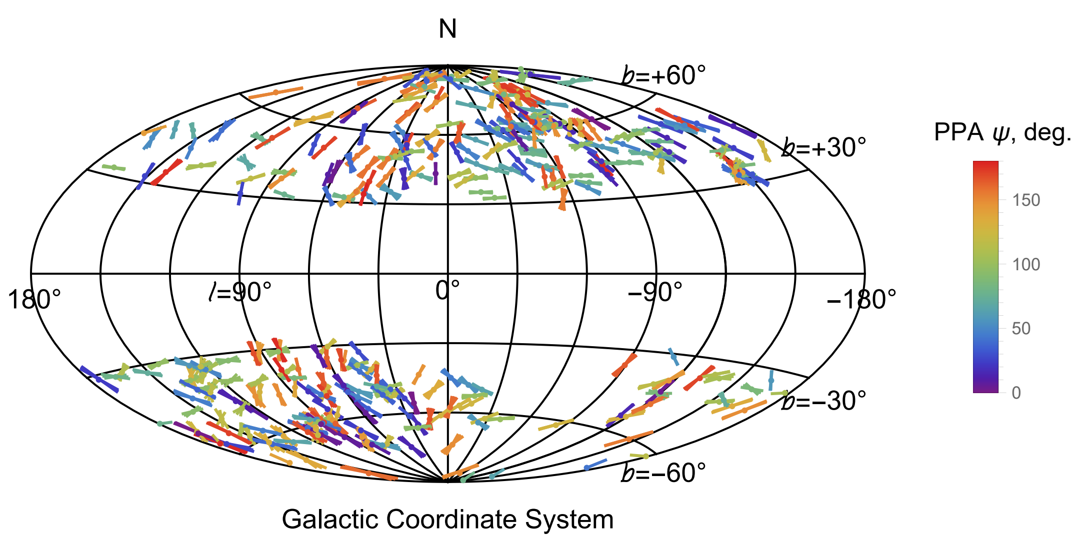


Figure A12.2. The data. The 355 QSOs and their polarization directions plotted in galactic coordinates. The shading colors indicate the polarization position angle ψ. The band of white space reflects the absence of sources within 30° of the Galactic Equator.

In[567]:=
```
SetDirectory[NotebookDirectory[]]
(*Export["mapOfDataGal.pdf",mapOfSources,ImageSize→1.2 432]*)
```

Out[567]=
```
C:\Users\momen\Dropbox\HOME_DESKTOP-0MRE5OJ\SendXXX_CJP_CEJPetc\
  SendViXra\20200715AlignmentMethod\20200715AlignmentMMAnotebooks\StarterKit
```

```
After Sec. A12, time and memory used are 565.813 seconds and 1913148648 bytes.
```

A13. Plot the alignment function $\overline{\eta}$(H)

The alignment function $\overline{\eta}$(H) is plotted on the sky. The alignment hubs H_{align} and the avoidance hubs H_{avoid} are displayed with their location error ellipses; the uncertainty is due to experimental uncertainty in the polarization directions ψ. The contour lines should be blurred by a similar uncertainty.

Definitions:

xyηBarAitoffTable table of the function $\overline{\eta}(x, y)$; {x(H),y(H), $\overline{\eta}(H)$} where x and y are Aitoff coordinates at H and $\overline{\eta}(H)$ is the alignment angle function at H
xyAitoffHjAlign, xyAitoffHjAvoid, xyAitoffOPPHjAlign, xyAitoffOPPHjAvoid Aitoff coords at the hubs
dηContourPlot angle between $\overline{\eta}$ contours in degrees
listCP basic list contour plot, to be dressed up later
ellAxisLabels Galactic longitude ticks for plot; longitude plotted right to left
beeAxisLabels Galactic latitude ticks for plot
mapOfηBar contour map of alignment function $\overline{\eta}$ (*H*)
mapOfηBarU hub uncertainty ellipses added to contour map mapOfηBar

In[572]:=
```
(*Transcribe the alignment function η̄(H), the location of the sources,
and the Celestial Equator onto an Aitoff plot.*)
xyηBarAitoffTable = Partition[Flatten[Table[
     {xHGal[ℓ, b], yHGal[ℓ, b], ηBarHjSmooth[ℓ, b]}, {ℓ, -180., 180., 2.}, {b, -88., 88., 2.}]], 3];
(*The Aitoff coordinates for the hubs*)
xyAitoffHjAlign = {xHGal[ℓHjAlign, bHjAlign], yHGal[ℓHjAlign, bHjAlign]};
xyAitoffHjAvoid = {xHGal[ℓHjAvoid, bHjAvoid], yHGal[ℓHjAvoid, bHjAvoid]};
xyAitoffOPPHjAlign =
  {xHGal[Mod[ℓHjAlign - 180., 360.], -bHjAlign], yHGal[Mod[ℓHjAlign - 180., 360.], -bHjAlign]};
xyAitoffOPPHjAvoid =
  {xHGal[Mod[ℓHjAvoid - 180., 360.], -bHjAvoid], yHGal[Mod[ℓHjAvoid - 180., 360.], -bHjAvoid]};
```

InterpolatingFunction: Input value {180., -88.} lies outside the range of data in the interpolating function.

InterpolatingFunction: Input value {180., -86.} lies outside the range of data in the interpolating function.

InterpolatingFunction: Input value {180., 86.} lies outside the range of data in the interpolating function.

General: Further output of InterpolatingFunction::femdmval will be suppressed during this calculation.

In[577]:=
```
(* Contour plot of the alignment function ηBarHjSmooth. *)
dηContourPlot = 1.;
(*, in degrees. *)listCP = ListContourPlot[Union[xyηBarAitoffTable], AspectRatio → 1/2,
  Contours → Table[η, {η, Floor[ηBarHjAlign[[1]]] + 1, Ceiling[ηBarHjAvoid[[1]]] - 1, dηContourPlot}],
  ColorFunction → "TemperatureMap", PlotRange → {{-3.5, 3.5}, {-2., 2.}},
  Axes → False, Frame → False, (*PlotLabel→"The alignment function η̅(H)",*)
  PlotLegends → Placed[BarLegend[Automatic, LegendMargins → {{0, 0}, {10, 5}},
     LegendLabel → "η̅(H), deg.", LabelStyle → {Plain, FontFamily → "Times"}], Right]];
```

In[578]:=
```
(*Axes tick labels*)
ellAxisLabels = Table[Text[StyleForm[ℓ "°", FontSize → 10, FontWeight → "Plain", Black],
    {xHGal[ℓ, -7], yHGal[ℓ, -7]}], {ℓ, -180, 180, 90}];
beeAxisLabels = {Text[StyleForm[-30 °, FontSize → 10, FontWeight → "Plain", Black],
    {xHGal[-180, -30] + 0.15, yHGal[-180, -30] - 0.1}],
   Text[StyleForm[+30 °, FontSize → 10, FontWeight → "Plain", Black],
    {xHGal[-180, 30] + 0.15, yHGal[-180, 30] + 0.1}]};
```

In[580]:=
```
(*Construct the map of η̅(H).*)
mapOfηBar =
  Show[{listCP, Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {b, -90, 90},
      PlotStyle → {Black, Thickness[0.002]}, (*Mesh→{11,5,0}(*{23,11,0}*),
      MeshStyle→Thick,*)PlotPoints → 60, PlotRange → All], {ℓ, -180., 180., 30}],
    Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {ℓ, -180., 180.},
      PlotStyle → {Black, Thickness[0.002]}, (*Mesh→{11,5,0}(*{23,11,0}*),MeshStyle→Thick,*)
      PlotPoints → 60], {b, -60, 60, 30}], Graphics[{ellAxisLabels, beeAxisLabels,
      Text[StyleForm["N", FontSize → 14, FontWeight → "Plain", Black], {0, 1.85}], Magenta,
      PointSize[0.009], Point[ {xHGal[(HminℓFit), (HminbFit)], yHGal[(HminℓFit), (HminbFit)]}],
      Point[ {xHGal[(HmaxℓFit), (HmaxbFit)], yHGal[(HmaxℓFit), (HmaxbFit)]}],
      Point[ {xHGal[HminℓFit + 180., -HminbFit], yHGal[HminℓFit + 180., -HminbFit]}],
      Point[ {xHGal[Mod[HmaxℓFit - 180., 360.], -HmaxbFit], yHGal[Mod[HmaxℓFit - 180., 360.],
         -HmaxbFit]}], PointSize[0.015], (*Point[ {xHGal[90.,30.],yHGal[90.,30.]}],*)
      Black, Text[StyleForm["H_align", FontSize → 12, FontWeight → "Bold"], {2.3, +1.5}],
      {Arrow[BezierCurve[{{2.3, 1.3}, {3., -0.3},
           {xHGal[HminℓFit, HminbFit] + 0.05, yHGal[HminℓFit, HminbFit] - 0.02}}]]} ,
      Text[StyleForm["-H_align", FontSize → 12, FontWeight → "Bold"], {-2.3, -1.5}],
      {Arrow[BezierCurve[{{-2.55, -1.3}, {-2.8, 0.7},
           {xHGal[HminℓFit + 180., -HminbFit] - 0.05, yHGal[HminℓFit + 180., -HminbFit] + 0.02}}]]},
      Text[StyleForm["H_avoid", FontSize → 12, FontWeight → "Bold"], {2.3, -1.5}],
      {Arrow[BezierCurve[{{2., -1.5}, {0.7, -1.6},
           {xHGal[HmaxℓFit, HmaxbFit] + 0.02, yHGal[HmaxℓFit, HmaxbFit] - 0.05}}]]},
      Text[StyleForm["-H_avoid", FontSize → 12, FontWeight → "Bold"], {-2.3, +1.5}],
      {Arrow[BezierCurve[{{-2.6, 1.3}, {-3.2, 1.4}, {xHGal[Mod[HmaxℓFit - 180., 360.], -HmaxbFit] -
             0.05, yHGal[Mod[HmaxℓFit - 180., 360.], -HmaxbFit] + 0.07}}]]}, Text[
       StyleForm["Galactic Coordinate System", FontSize → 14, FontWeight → "Plain"], {0, -1.85}]
      }]}, ImageSize → 432];
```

```
In[581]:= (*Construct the map of η̄(H).*)
mapOfηBarU =
  Show[{mapOfηBar, ParametricPlot[
     {xHGal[ℓHminEllipse[θ], bHminEllipse[θ]], yHGal[ℓHminEllipse[θ], bHminEllipse[θ]]},
     {θ, 0., 2. π}, PlotStyle → {Magenta, Thickness[0.002]}, PlotRange → {{-4., 4.}, {-4., 4.}}],
    ParametricPlot[{xHGal[ℓHmaxEllipse[θ], bHmaxEllipse[θ]], yHGal[ℓHmaxEllipse[θ],
       bHmaxEllipse[θ]]}, {θ, 0., 2. π}, PlotStyle → {Magenta, Thickness[0.002]}],
    ParametricPlot[{xHGal[ℓHminEllipse[θ] + 180., -bHminEllipse[θ]], yHGal[ℓHminEllipse[θ] + 180.,
       -bHminEllipse[θ]]}, {θ, 0., 2. π}, PlotStyle → {Magenta, Thickness[0.002]}],
    ParametricPlot[{xHGal[ℓHmaxEllipse[θ] + 180., -bHmaxEllipse[θ]],
      yHGal[ℓHmaxEllipse[θ] + 180., -bHmaxEllipse[θ]]}, {θ, 0., 2. π},
     PlotStyle → {Magenta, Thickness[0.002]}]}, ImageSize → 1.2 × 432];
```

Out[582]=

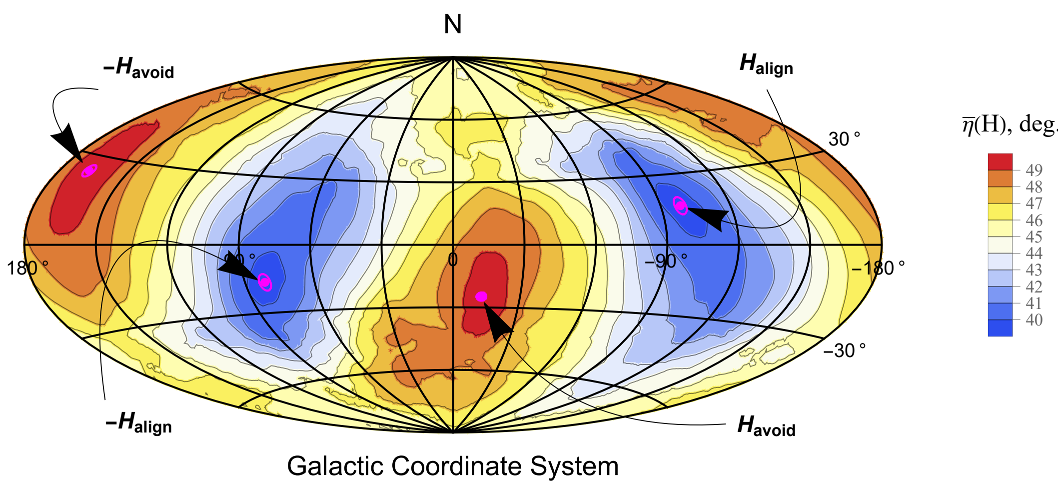


Figure A13.1. Map of the average alignment angle function $\bar{\eta}(H)$. The $\pm 1\sigma$ uncertainty ellipses are drawn at the locations of the alignment and avoidance hubs $\pm H_{align}$ and $\pm H_{avoid}$. The contour lines are expected to be uncertain by a similar amount, a couple of degrees.

```
In[584]:= SetDirectory[NotebookDirectory[]]
(*Export["mapOfEtaBarU.pdf",mapOfηBarU,ImageSize→1.2 432]*)
```

```
Out[584]= C:\Users\momen\Dropbox\HOME_DESKTOP-0MRE5OJ\SendXXX_CJP_CEJPetc\
  SendViXra\20200715AlignmentMethod\20200715AlignmentMMAnotebooks\StarterKit
```

After Sec. A13, time and memory used are 606.048 seconds and 1 926 721 928 bytes.

A14. Plot data with H_{align} as the (virtual) North Pole

The Hub Test of alignment seeks to find a location, the hub H_{align}, where the polarization vectors act most like Local Norths. Therefore, we want a rotation that takes the alignment hub H_{Align} to the point (0,0,1) where the z-axis intersects the Celestial sphere. That makes H_{align} look like a virtual North Pole and the tendency of the polarization vectors to point toward H_{align} may be more apparent. To fix the rotation, make the hub H_{avoid} move to the 0° meridian, $\ell_{Havoid} =$ 0°, very near the x-axis. The angle between hubs is a right angle, within experimental uncertainty, so having H_{avoid} in the x-direction is within experimental uncertainty.

A14a. Build the rotation, move quantities.

Definitions

rot4HAlignToZ0 rotation taking alignment hub H_{align} from uncertainty runs to the North Pole, the point (0,0,1)
rot4NegHavoid0 where the avoidance hub H_{avoid} ends up after rotating with rot4HAlignToZ0
rot4HAlignToZ rotation taking H_{align} to the North Pole and forcing H_{avoid} to have longitude $\ell = 0 = 360°$, near x-axis (ℓ, b) = (0,0).

rot4rNGP, rot4ℓNGP, rot4*b*NGP radial unit vector, ℓ,*b* of NGP in rotated coords
rot4rHavoid, rot4ℓHavoid, rot4*b*Havoid radial unit vector, ℓ,*b* of Havoid in rotated coords
rot4ri, rot4ℓi, rot4*b*i new unit radial vectors of sources, then new longitudes and latitudes
rot4xyAitoffSourcesToHalign the Aitoff coords of the sources
rot4rPlus*ψ*, rot4rPlus*ψ*Big, Small radial unit vectors for drawing polarization line segments

A14a. Build the rotation, move quantities.

Definitions

rot4HAlignToZ0 rotation taking most likely hub H_{align} from uncertainty runs to the North Pole, the point (0,0,1)
rot4NegHavoid0 the point where the avoidance hub H_{avoid} ends up after applying rot4HAlignToZ0
rot4HAlignToZ rotation taking most likely hub H_{align} from uncertainty runs to the North Pole, the point (0,0,1) and most likely hub H_{avoid} from uncertainty runs to the xz-plane, so H_{avoid} has galactic longitude $\ell = 0 = 360°$
rot4rNGP, rot4ℓNGP, rot4*b*NGP radial unit vector, ℓ,*b* of NGP in rotated coords
rot4rNEP, rot4ℓNEP, rot4*b*NEP radial unit vector, ℓ,*b* of NEP in rotated coords
rot4rHavoid, rot4ℓHavoid, rot4*b*Havoid radial unit vector, ℓ,*b* of Havoid in rotated coords
rot4ri, rot4ℓi, rot4*b*i rotate sources, these are the unit radial vector, then galactic longitude and latitude
rot4xyAitoffSourcesToHalign the appropriate Aitoff coords for sources
rot4rPlus*ψ*, rot4rPlus*ψ*Big, Small radial unit vectors for polarization line segments
Rotate the edge of the uncertainty region of the hub H_{align} and H_{avoid} :
rOfHAlignRegion[θ] radial unit vector for uncertainty one sigma curve for H_{align} Galactic coords
rot4rOfHAlignRegion, ℓOfrot4rOfHAlignRegion, *b*Ofrot4rOfHAlignRegion ℓ, *b*, radial unit vector for uncertainty one sigma curve for H_{align} in coords with Halign at North Pole
rOfHAvoidRegion[θ] radial unit vector for uncertainty one sigma curve for H_{avoid} Galactic coords
rot4rOfHAvoidRegion, ℓOfrot4rOfHAvoidRegion, *b*Ofrot4rOfHAvoidRegion ℓ, *b*, radial unit vector for uncertainty one sigma curve for H_{avoid} in coords with Halign at North Pole

rot4crossesOverPlus, rot4crossesOverMinus ids for sources crossing over boundaries
{rot4polarLinesNoCrossing1ToHalign, rot4polarLinesCrossingPlus1ToHalign, rot4polarLinesCrossingMinus1ToHalign, rot4polarLinesBig, Small } rotate the line segments representing the polarization vectors
rot4setUPplotForMapToHalign preliminary plot with meridians and parallels and legend
rot4mapOfSourcesToHalign map of sources and polarization directions with uncertainties in a coordinate system with H_{align} at the North Pole

In[589]:=
```
(*{HminℓFit,HminbFit}*)(*To the virtual North Pole*)
rot4HAlignToZ0 = RotationMatrix[{er[HminℓFit, HminbFit], {0, 0, 1.}}];
```

In[590]:=
```
rot4NegHavoid0 = rot4HAlignToZ0.er[Hmaxℓ Fit + 180., -Hmaxb Fit] ;
(*Note: er[Hmaxℓ Fit+180.,-Hmaxb Fit] = unit radial vector to -Havoid*)
{ℓ FROMr[rot4NegHavoid0], b FROMr[rot4NegHavoid0]};
```

In[592]:=
```
ℓ FROMr[RotationMatrix[
   -(ℓ FROMr[rot4NegHavoid0] + 180.) ((2. π) / 360.), {0, 0, 1.}].rot4NegHavoid0]
```

Out[592]= -180.

In[593]:=
```
rot4HAlignToZ = RotationMatrix[
    -(ℓ FROMr[rot4NegHavoid0] + 180.) ((2. π) / 360.), {0, 0, 1.}].rot4HAlignToZ0;
```

In[594]:=
```
Print["Check that rot4 takes Halign to the z-axis: rot4.Halign = ",
 Chop[rot4HAlignToZ.er[Hminℓ Fit, Hminb Fit]], "."]
Print["Check that rot4 takes Havoid close to the x-axis: rot4.Havoid = ",
 Chop[rot4HAlignToZ.er[Hmaxℓ Fit, Hmaxb Fit]], "."]
Print[
 "Since the radial direction to Halign is nearly perpendicular to the radius to Havoid,
   Havoid can, at best, be a little off the x-axis if Halign is on the z-axis. "]
```

Check that rot4 takes H_{align} to the z-axis: rot4.Halign = {0, 0, 1.}.

Check that rot4 takes H_{avoid} close to the x-axis: rot4.Havoid = {0.998441, 0, -0.0558134}.

Since the radial direction to H_{align} is nearly perpendicular to the radius to H_{avoid}, H_{avoid} can, at best, be a little off the x-axis if Halign is on the z-axis.

In[597]:=
```
rot4rNGP = rot4HAlignToZ.{0., 0., 1.} ;(*North Galactic Pole*)
rot4ℓ NGP = ℓ FROMr[rot4rNGP];
rot4b NGP = b FROMr[rot4rNGP];
```

In[600]:=
```
Print["Rot4 takes the NGP to (long.,lat.) = ", {rot4ℓ NGP, rot4b NGP},
 " in the new coord system. Long., lat. are in degrees."]
```

Rot4 takes the NGP to (long.,lat.) = {114.933, 16.5773}
 in the new coord system. Long., lat. are in degrees.

In[601]:=
```
rot4rHavoid = rot4HAlignToZ.er[Hmaxℓ Fit, Hmaxb Fit];(*Havoid*)
rot4ℓ Havoid = ℓ FROMr[rot4rHavoid];
rot4b Havoid = b FROMr[rot4rHavoid];
```

In[604]:=
```
Print["Rot4 takes the hub Havoid to (long.,lat.) = ", {Chop[rot4ℓ Havoid], rot4b Havoid},
 " in the new coord system. Long., lat. are in degrees."]
```

Rot4 takes the hub H_{avoid} to (long.,lat.) =
 {0, -3.19953} in the new coord system. Long., lat. are in degrees.

In[605]:=
```
rot4ri[i_] := rot4HAlignToZ.ri[i];(*Rotate the sources*)
```

In[606]:=
```
rot4ℓ i[i_] := ℓ FROMr[rot4ri[i]];
rot4b i[i_] := b FROMr[rot4ri[i]];
```

In[608]:=
```
(*Plot sources*)
rot4xyAitoffSourcesToHalign = Table[{ColorData["Rainbow"][(ηiHj[i, jHalign]) / 90.],
    Point[{xHGal[rot4ℓ i[i], rot4b i[i] ], yHGal[ rot4ℓ i[i], rot4b i[i] ]}]}, {i, nSrc}];
```

In[609]:=
```
rot4rPlusψ[i_, d_] := rot4HAlignToZ.rPlusψ[i, d]
rot4rPlusψBig[i_, d_] := rot4HAlignToZ.rPlusψBig[i, d]
rot4rPlusψSmall[i_, d_] :=
 rot4HAlignToZ.rPlusψSmall[i, d](*Rotate the polarization directions*)
```

A14b. Make plot with H_{align} as the (virtual) North Pole

Definitions

uncertainty ellipses:

rot4rOfHAlignRegion, ℓOfrot4rOfHAlignRegion, *b*Ofrot4rOfHAlignRegion radial unit vector for uncertainty ellipse and its ℓ, *b*
rot4rOfHAvoidRegion, ℓOfrot4rOfHAvoidRegion, *b*Ofrot4rOfHAvoidRegion radial unit vector for uncertainty ellipse and its ℓ, *b*

polarization line segments:

rot4crossesOverPlus, rot4crossesOverMinus ids for sources crossing over boundaries
{rot4polarLinesNoCrossing1ToHalign, rot4polarLinesCrossingPlus1ToHalign, rot4polarLinesCrossingMinus1ToHalign, rot4polar-LinesBig, Small } rotate the line segments representing the polarization vectors
rot4setUPplotForMapToHalign preliminary plot with meridians and parallels and legend
rot4mapOfSourcesToHalign map of sources and polarization directions, H_{align} at the North Pole
histoEtaToHalign histogram of alignment angles η_{iH} with H_{align} at the North Pole

rot4crossesOverPlus, rot4crossesOverMinus ids for sources crossing over boundaries
{rot4polarLinesNoCrossing1ToHalign, rot4polarLinesCrossingPlus1ToHalign, rot4polarLinesCrossingMinus1ToHalign, rot4polar-LinesBig, Small } rotate the line segments representing the polarization vectors
rot4setUPplotForMapToHalign preliminary plot with meridians and parallels and legend
rot4mapOfSourcesToHalign map of sources and polarization directions with uncertainties in a coordinate system with H_{align} at the North Pole

In[612]:=
```
(*Radial Unit Vector to the Uncertainty region perimeter.*)
rot4rOfHAlignRegion[θ_] := rot4HAlignToZ.rOfHAlignRegion[θ]
ℓOfrot4rOfHAlignRegion[θ_] := ℓFROMr[rot4rOfHAlignRegion[θ]]
bOfrot4rOfHAlignRegion[θ_] := bFROMr[rot4rOfHAlignRegion[θ]]
```

In[615]:=
```
rOfHAvoidRegion[θ_] := er[ℓHmaxEllipse[θ], bHmaxEllipse[θ]]
(*Radial Unit Vector to the Uncertainty ellipse.*)
rot4rOfHAvoidRegion[θ_] := rot4HAlignToZ.rOfHAvoidRegion[θ]
ℓOfrot4rOfHAvoidRegion[θ_] := ℓFROMr[rot4rOfHAvoidRegion[θ]]
bOfrot4rOfHAvoidRegion[θ_] := bFROMr[rot4rOfHAvoidRegion[θ]]
```

In[619]:=
```
(*Plot polarization directions*)
rot4crossesOverPlus = {}(*{19,324,337}*);
rot4crossesOverMinus = {82, 87, 85}(*{330,327,354}*);
For[i = 1, i ≤ nSrc, i++,
 If[ ℓFROMr[ rot4rPlusψ[i, 0.10]] - rot4ℓi[i] < -200, AppendTo[rot4crossesOverPlus, i]];
 If[ ℓFROMr[ rot4rPlusψ[i, -0.10]] - rot4ℓi[i] > 200, AppendTo[rot4crossesOverMinus, i]]]
rot4noCrossing =
  Complement[Range[nSrc], Union[rot4crossesOverPlus, rot4crossesOverMinus]];
```

```
In[622]:= (*Plot transverse directions*)
rot4polarLinesNoCrossing1ToHalign[d_] := (*rot4polarLinesNoCrossing1ToHalign[d]=*)
 Table[{ColorData["Rainbow"][(ηiHj[i, jHalign]) / 90.],
   Line[{{xHGal[ℓFROMr[ rot4rPlusψ[i, d]], bFROMr[ rot4rPlusψ[i, d]] ],
      yHGal[ℓFROMr[ rot4rPlusψ[i, d]], bFROMr[ rot4rPlusψ[i, d]]]},
     {xHGal[ℓFROMr[ rot4rPlusψ[i, -d]], bFROMr[ rot4rPlusψ[i, -d]]], yHGal[
       ℓFROMr[ rot4rPlusψ[i, -d]], bFROMr[ rot4rPlusψ[i, -d]]]}}]}, {i, rot4noCrossing}]

In[623]:= rot4polarLinesCrossingPlus1ToHalign[d_] := (*rot4polarLinesCrossingPlus1ToHalign[d]=*)
 Table[{ColorData["Rainbow"][(ηiHj[i, jHalign]) / 90.],
   Line[{{xHGal[rot4ℓi[i], rot4bi[i]], yHGal[rot4ℓi[i], rot4bi[i]]},
     {xHGal[ℓFROMr[ rot4rPlusψ[i, -d]], bFROMr[ rot4rPlusψ[i, -d]]], yHGal[ℓFROMr[
        rot4rPlusψ[i, -d]], bFROMr[ rot4rPlusψ[i, -d]]]}}]}, {i, rot4crossesOverPlus}]

In[624]:= rot4polarLinesCrossingMinus1ToHalign[d_] := (*rot4polarLinesCrossingMinus1ToHalign[d]=*)
 Table[{ColorData["Rainbow"][(ηiHj[i, jHalign]) / 90.],
   Line[{{xHGal[ℓFROMr[ rot4rPlusψ[i, d]], bFROMr[ rot4rPlusψ[i, d]]],
      yHGal[ℓFROMr[ rot4rPlusψ[i, d]], bFROMr[ rot4rPlusψ[i, d]]]},
     {xHGal[rot4ℓi[i], rot4bi[i]], yHGal[rot4ℓi[i], rot4bi[i]]}}]},
  {i, rot4crossesOverMinus(*noCrossing*)}]

In[625]:= rot4polarLinesBig[d_] := Table[{ColorData["Rainbow"][(ηiHj[i, jHalign]) / 90.],
   Line[{{xHGal[ℓFROMr[rot4rPlusψBig[i, d]], bFROMr[ rot4rPlusψBig[i, d]]],
      yHGal[ℓFROMr[ rot4rPlusψBig[i, d]], bFROMr[ rot4rPlusψBig[i, d]]]},
     {xHGal[ℓFROMr[ rot4rPlusψBig[i, -d]], bFROMr[ rot4rPlusψBig[i, -d]]], yHGal[ℓFROMr[
        rot4rPlusψBig[i, -d]], bFROMr[ rot4rPlusψBig[i, -d]]]}}]}, {i, rot4noCrossing}]

In[626]:= rot4polarLinesSmall[d_] := Table[{ColorData["Rainbow"][(ηiHj[i, jHalign]) / 90.],
   Line[{{xHGal[ℓFROMr[rot4rPlusψSmall[i, d]], bFROMr[ rot4rPlusψSmall[i, d]]],
      yHGal[ℓFROMr[ rot4rPlusψSmall[i, d]], bFROMr[ rot4rPlusψSmall[i, d]]]},
     {xHGal[ℓFROMr[ rot4rPlusψSmall[i, -d]], bFROMr[ rot4rPlusψSmall[i, -d]]],
      yHGal[ℓFROMr[ rot4rPlusψSmall[i, -d]],
       bFROMr[ rot4rPlusψSmall[i, -d]]]}}]}, {i, rot4noCrossing}]

In[627]:= rot4setUPplotForMapToHalign =
  Show[{ParametricPlot[{xHGal[-120, b], yHGal[-120, b]}, {b, -90, 90},
     PlotStyle → {Black, Thickness[0.002]}, PlotLegends →
      BarLegend[{"Rainbow", {0, 90.}}, LegendLabel → "η to H_align, deg."], PlotPoints → 60,
     PlotRange → {{-4.0, 3.5}, (7.5 / 11.0) {-3, 3}}, Axes → False, Frame → False],
    Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {b, -90, 90},
      PlotStyle → {Black, Thickness[0.002]}, PlotPoints → 60, PlotRange → {{-4.0, 3.5},
        (7.5 / 11.0) {-3, 3}}, Axes → False, Frame → False], {ℓ, -180., 180., 30}],
    Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {ℓ, -180., 180.},
      PlotStyle → {Black, Thickness[0.002]}, PlotPoints → 60], {b, -60, 60, 30}]}];
```

In[628]:=
```
(*map of sources and ψ with σψ uncertainties in a coordinate system with Halign at (0,0,1).*)
rot4mapOfSourcesToHalign =
  Show[{rot4setUPplotForMapToHalign, Graphics[
     {PointSize[0.006], Text[StyleForm["HAlign", FontSize → 14, FontWeight → "Plain"], {0, 1.85}],
      Text[StyleForm["0°", FontSize → 14, FontWeight → "Plain"], {xHGal[0., -5.], yHGal[0., -5.]}],
      Text[StyleForm["90°", FontSize → 14, FontWeight → "Plain"],
       {xHGal[90., -5.], yHGal[90., -5.]}], Text[StyleForm["180°", FontSize → 14,
        FontWeight → "Plain"], {xHGal[180., -5.], yHGal[180., -5.]}],
      Text[StyleForm["-180°", FontSize → 14, FontWeight → "Plain"],
       {xHGal[-180., -5.], yHGal[-180., -5.]}], Text[StyleForm["-90°", FontSize → 14,
        FontWeight → "Plain"], {xHGal[-90., -5.], yHGal[270., -5.]}],
      Text[
       StyleForm["90°", FontSize → 14, FontWeight → "Plain"], {xHGal[90., -5.], yHGal[90., -5.]}],
      (*{Black,PointSize[Large],Point[{xHGal[rot4ℓAE,rot4bAE],yHGal[rot4ℓAE,rot4bAE]}]},*)
      {Purple, PointSize[Large], Point[{xHGal[rot4ℓNGP, rot4bNGP], yHGal[rot4ℓNGP, rot4bNGP]}]},
      {Purple, Arrow[BezierCurve[{{-2.7, 1.0}, {-3.8, 0.5},
           {xHGal[rot4ℓNGP, rot4bNGP] - 0.04, yHGal[rot4ℓNGP, rot4bNGP]}}]]},
      Text[StyleForm["NGP", Purple, FontSize → 14, FontWeight → "Plain"], {-2.6, 1.1}],
      {Cyan, PointSize[Large], Point[{{xHGal[140., 90.], yHGal[140., 90.]},
          {xHGal[rot4ℓHavoid, rot4bHavoid], yHGal[rot4ℓHavoid, rot4bHavoid]}}]},
      Text[StyleForm["HAvoid", Black, FontSize → 14, FontWeight → "Plain"],
       {xHGal[rot4ℓHavoid, rot4bHavoid] + 0.05, yHGal[rot4ℓHavoid, rot4bHavoid] + 0.2}],
      {Cyan, PointSize[Large], Point[{xHGal[rot4ℓHavoid - 180., -rot4bHavoid],
          yHGal[rot4ℓHavoid - 180., -rot4bHavoid]}]},
      Text[StyleForm["-HAvoid", Black, FontSize → 14, FontWeight → "Plain"],
       {xHGal[rot4ℓHavoid - 172., -rot4bHavoid], yHGal[rot4ℓHavoid - 172., -rot4bHavoid] + 0.2}],
      Text[StyleForm["Rotated Coordinate System I", FontSize → 14, FontWeight → "Plain"],
       {0, -1.85}], {Thickness[0.003], rot4polarLinesBig[0.05]},
      {Thickness[0.003], rot4polarLinesSmall[0.095]},
      {Thickness[0.003], rot4polarLinesNoCrossing1ToHalign[0.08]},
      {Thickness[0.003], rot4polarLinesCrossingPlus1ToHalign[0.08]},
      {Thickness[0.003], rot4polarLinesCrossingMinus1ToHalign[0.08]},
      (*Sources S:*) rot4xyAitoffSourcesToHalign
         }]}, ImageSize → 1.2 × 432];
```

Out[629]=

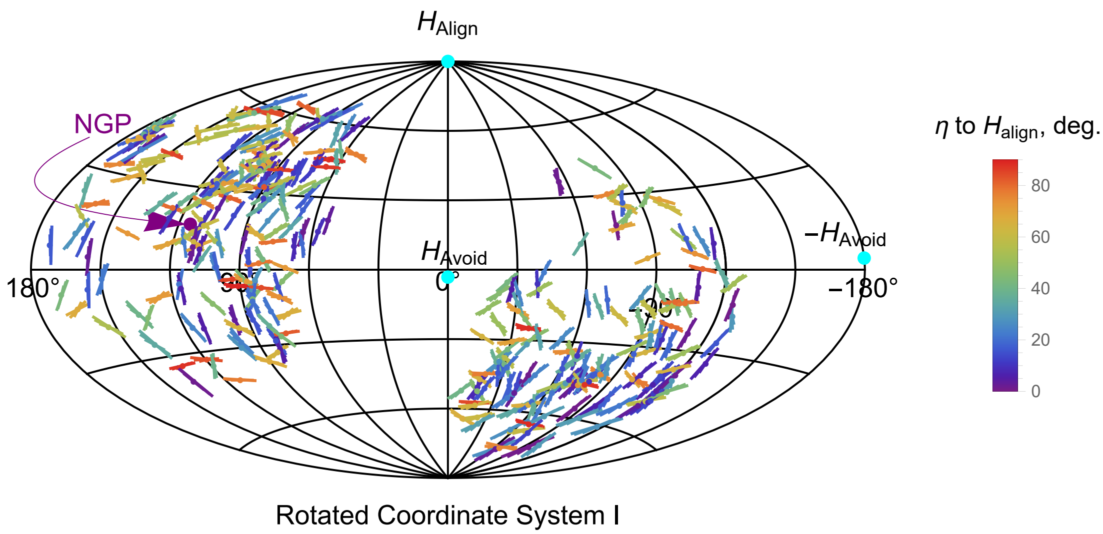


Figure A14.1 With the alignment hub H_{Align} at the virtual North Pole, a pine needle plot of the 355 QSO sources. The location of the North Galactic Pole NGP helps guide the eye. Many of the polarization vectors point in the same direction as nearby meridians; those vectors approximate Local Norths. Also note that the vectors aligned with H_{Align}, shaded blue and purple, occur throughout the sample, indicating the alignment is widespread, an extreme-scale phenomenon.

In[631]:=
```
Table[Histogram[Table[ηiHj[i, jHalign], {i, nSrc}], {dη}, PlotLabel → dη], {dη, 1, 12}];
```

In[632]:=
```
histoEtaToHalign = Histogram[Table[ηiHj[i, jHalign], {i, nSrc}], {9.0},
   PlotTheme → {"Scientific"}, FrameLabel → {"η_iH (to H = H_align), deg.", "ΔR"},
   PlotLabel → "Histogram of η_iH from i^th source ψ_i to H=H_align "];
```

Out[633]=

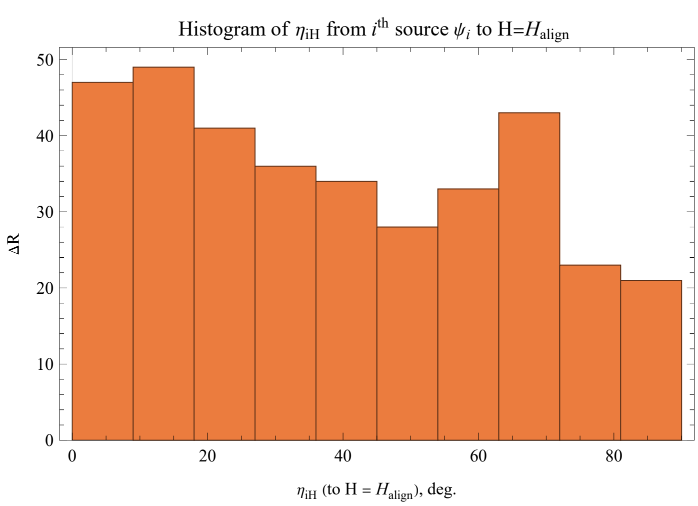


Figure A14.2 The distribution is somewhat bimodal, with peaks near η_{iH} = 0° and η_{iH} = 65°.

In[635]:=
```
SetDirectory[NotebookDirectory[]]
(*Export["rot4mapOfSourcesToHalignGal.pdf",rot4mapOfSourcesToHalign,ImageSize→1.2 432]*)
```

Out[635]=
```
C:\Users\momen\Dropbox\HOME_DESKTOP-0MRE5OJ\SendXXX_CJP_CEJPetc\
  SendViXra\20200715AlignmentMethod\20200715AlignmentMMAnotebooks\StarterKit
```

```
After Sec. A14, time and memory used are 609.392 seconds and 1931460728 bytes.
```

A15. Plot data with H_{avoid} as the (virtual) North Pole

Just as the Hub Test of alignment asks how well the polarization vectors act as Local Norths, the Hub Test of Avoidance asks how well the polarization vectors act as Local Easts. To make the avoidance more easily viewed, the avoidance hub H_{avoid} is rotated to make a virtual North Pole and this time we see how well the polarization vectors point in the East-West direction.

Since the hubs H_{avoid} and H_{align} are almost perpendicular, we again arrange the rotation to keep the hubs in *xz*-plane. Preserving the relative orientation means that the alignment hub H_{align} should point along the negative *x*-direction.

We want a rotation that takes the avoidance hub H_{Avoid} to the North Pole , (0,0,1) = z-axis, and the hub H_{align} to the 180° meridian, ℓ_{Havoid} = 180° = 12^h.

A15a. Build the rotation, move quantities.

Definitions

rot5HAvoidToZ0 rotation taking the avoidance hub H_{avoid} to the North Pole, the point (0,0,1)
rot5NegHalign0 where the alignment hub H_{align} ends up after applying rot5HAvoidToZ0
rot5HAvoidToZ rotation taking H_{avoid} to the North Pole, the point (0,0,1) and H_{align} to the xz-plane, so H_{align} is near negative x -axis (ℓ,b) = (180°,0).
rot5rNGP, rot5ℓNGP, rot5bNGP radial unit vector, ℓ,b of NGP in rotated coords

rot5rHalign, rot5ℓHalign, rot5bHalign radial unit vector, ℓ,b of Halign
rot5ri, rot5ℓi, rot5bi rotate sources: unit radial vector, then galactic longitude and latitude
rot5xyAitoffSourcesToHavoid Aitoff coords for the sources
rot5rPlusψ, rot5rPlusψBig, Small radial unit vectors for polarization line segments

In[640]:=
```
(*{Hmaxℓ Fit,HmaxbFit}*)(*To the virtual North Pole*)
rot5HAvoidToZ0 = RotationMatrix[{er[HmaxℓFit, HmaxbFit], {0, 0, 1.}}];
```

In[641]:=
```
rot5NegHalign0 = rot5HAvoidToZ0.er[HminℓFit, HminbFit] ;
(*er[HminℓFit,HminbFit] = unit radial vector to  Halign*)
{ℓFROMr[rot5NegHalign0], bFROMr[rot5NegHalign0]};
```

In[643]:=
```
ℓFROMr[RotationMatrix[
   - (ℓFROMr[rot5NegHalign0] - 180.) ((2. π) / 360.), {0, 0, 1.}].rot5NegHalign0];
```

In[644]:=
```
rot5HAvoidToZ = RotationMatrix[
   - (ℓFROMr[rot5NegHalign0] - 180.) ((2. π) / 360.), {0, 0, 1.}].rot5HAvoidToZ0;
```

Check that rot5 takes H_{avoid} to the z-axis: rot5.Havoid = {0, 0, 1.}.

Check that rot5 takes H_{align} to the negative x-axis: rot5.Halign = {-0.998441, 0, -0.0558134}.

Since the radial direction to H_{align} is not perpendicular to the radius
to H_{avoid}, H_{align} must be a little off the -x-axis if Havoid is on the z-axis.

```
In[648]:= rot5rNGP = rot5HAvoidToZ.{0., 0., 1.} ;(*North Galactic Pole*)
rot5ℓNGP = ℓFROMr[rot5rNGP];
rot5bNGP = bFROMr[rot5rNGP];
```

Rot5 takes the NGP to (long.,lat.) = {106.795, -24.7924}
in the new coord system. Long., lat. are in degrees.

```
In[652]:= rot5rHalign = rot5HAvoidToZ.er[HminℓFit, HminbFit];(*Halign*)
rot5ℓHalign = ℓFROMr[rot5rHalign];
rot5bHalign = bFROMr[rot5rHalign];
```

Rot5 takes the hub H_{align} to (long.,lat.) =
{-180., -3.19953} in the new coord system. Long., lat. are in degrees.

```
In[656]:= rot5ri[i_] := rot5HAvoidToZ.ri[i];(*Rotate sources*)

In[657]:= rot5ℓi[i_] := ℓFROMr[rot5ri[i]];
rot5bi[i_] := bFROMr[rot5ri[i]];

In[659]:= (*Plot sources*)
rot5xyAitoffSourcesToHavoid = Table[{ColorData["Rainbow"][(ηiHj[i, jHavoid]) / 90.],
    Point[{xHGal[rot5ℓi[i], rot5bi[i] ], yHGal[ rot5ℓi[i], rot5bi[i] ]}]}, {i, nSrc}];

In[660]:= rot5rPlusψ[i_, d_] := rot5HAvoidToZ.rPlusψ[i, d]
rot5rPlusψBig[i_, d_] := rot5HAvoidToZ.rPlusψBig[i, d]
rot5rPlusψSmall[i_, d_] :=
 rot5HAvoidToZ.rPlusψSmall[i, d](*Rotate the polarization directions*)
```

A15b. Make plot with H_{avoid} as the (virtual) North Pole

Definitions

uncertainty ellipses:

rot5rOfHAlignRegion, ℓOfrot5rOfHAlignRegion, bOfrot5rOfHAlignRegion ℓ, b, radial unit vector for H_{align} uncertainty ellipse in coords with Havoid at North Pole

rot5rOfHAvoidRegion, ℓOfrot5rOfHAvoidRegion, bOfrot5rOfHAvoidRegion ℓ, b, radial unit vector for H_{avoid} uncertainty ellipse in coords with Havoid at North Pole

polarization line segments:

rot5crossesOverPlus, rot5crossesOverMinus ids for sources crossing over boundaries

{rot5polarLinesNoCrossing1ToHalign, rot5polarLinesCrossingPlus1ToHalign, rot5polarLinesCrossingMinus1ToHalign, rot5polarLinesBig, Small } rotate the line segments representing the polarization vectors

rot5setUPplotForMapToHalign preliminary plot with meridians and parallels and legend

rot5mapOfSourcesToHavoid map of sources and polarization directions with uncertainties in a coordinate system with H_{align} at the North Pole histoEtaToHavoid histogram of alignment angles η_{iH} with H_{avoid} at the North Pole

In[663]:=
```
(*Radial Unit Vector to the Uncertainty region perimeter.*)
rot5rOfHAlignRegion[θ_] := rot5HAvoidToZ.rOfHAlignRegion[θ]
ℓOfrot5rOfHAlignRegion[θ_] := ℓFROMr[rot5rOfHAlignRegion[θ]]
bOfrot5rOfHAlignRegion[θ_] := bFROMr[rot5rOfHAlignRegion[θ]]
```

In[666]:=
```
(*Radial Unit Vector to the Havoid Uncertainty ellipse.*)
rot5rOfHAvoidRegion[θ_] := rot5HAvoidToZ.rOfHAvoidRegion[θ]
ℓOfrot5rOfHAvoidRegion[θ_] := ℓFROMr[rot5rOfHAvoidRegion[θ]]
bOfrot5rOfHAvoidRegion[θ_] := bFROMr[rot5rOfHAvoidRegion[θ]]
```

In[669]:=
```
(*Plot polarization directions*)
rot5crossesOverPlus = {87, 85, 82}(*{19,324,337}*);
rot5crossesOverMinus = {}(*{330,327,354}*);
For[i = 1, i ≤ nSrc, i++,
 If[ ℓFROMr[ rot5rPlusψ[i, 0.10]] - rot5ℓi[i] < -200, AppendTo[rot5crossesOverPlus, i]];
 If[ ℓFROMr[ rot5rPlusψ[i, -0.10]] - rot5ℓi[i] > 200, AppendTo[rot5crossesOverMinus, i]]]
rot5noCrossing =
  Complement[Range[nSrc], Union[rot5crossesOverPlus, rot5crossesOverMinus]];
```

In[672]:=
```
(*Plot transverse directions*)
rot5polarLinesNoCrossing1ToHvoid[d_] := (*rot5polarLinesNoCrossing1ToHvoid[d]=*)
 Table[{ColorData["Rainbow"][(ηiHj[i, jHavoid]) / 90.],
   Line[{{xHGal[ℓFROMr[ rot5rPlusψ[i, d]], bFROMr[ rot5rPlusψ[i, d]] ],
      yHGal[ℓFROMr[ rot5rPlusψ[i, d]], bFROMr[ rot5rPlusψ[i, d]]]},
     {xHGal[ℓFROMr[ rot5rPlusψ[i, -d]], bFROMr[ rot5rPlusψ[i, -d]]], yHGal[
       ℓFROMr[ rot5rPlusψ[i, -d]], bFROMr[ rot5rPlusψ[i, -d]]]}}]}, {i, rot5noCrossing}]
```

In[673]:=
```
rot5polarLinesCrossingPlus1ToHvoid[d_] := (*rot5polarLinesCrossingPlus1ToHvoid[d]=*)
 Table[{ColorData["Rainbow"][(ηiHj[i, jHavoid]) / 90.],
   Line[{{xHGal[rot5ℓi[i], rot5bi[i]], yHGal[rot5ℓi[i], rot5bi[i]]},
     {xHGal[ℓFROMr[ rot5rPlusψ[i, -d]], bFROMr[ rot5rPlusψ[i, -d]]], yHGal[ℓFROMr[
        rot5rPlusψ[i, -d]], bFROMr[ rot5rPlusψ[i, -d]]]}}]}, {i, rot5crossesOverPlus}]
```

In[674]:=
```
rot5polarLinesCrossingMinus1ToHvoid[d_] := (*rot5polarLinesCrossingMinus1ToHvoid[d]=*)
 Table[{ColorData["Rainbow"][(ηiHj[i, jHavoid]) / 90.],
   Line[{{xHGal[ℓFROMr[ rot5rPlusψ[i, d]], bFROMr[ rot5rPlusψ[i, d]]],
      yHGal[ℓFROMr[ rot5rPlusψ[i, d]], bFROMr[ rot5rPlusψ[i, d]]]},
     {xHGal[rot5ℓi[i], rot5bi[i]], yHGal[rot5ℓi[i], rot5bi[i]]}}]},
  {i, rot5crossesOverMinus(*noCrossing*)}]
```

In[675]:=
```
rot5polarLinesBig[d_] := Table[{ColorData["Rainbow"][(ηiHj[i, jHavoid]) / 90.],
   Line[{{xHGal[ℓFROMr[rot5rPlusψBig[i, d]], bFROMr[ rot5rPlusψBig[i, d]]],
      yHGal[ℓFROMr[ rot5rPlusψBig[i, d]], bFROMr[ rot5rPlusψBig[i, d]]]},
     {xHGal[ℓFROMr[ rot5rPlusψBig[i, -d]], bFROMr[ rot5rPlusψBig[i, -d]]], yHGal[ℓFROMr[
        rot5rPlusψBig[i, -d]], bFROMr[ rot5rPlusψBig[i, -d]]]}}]}, {i, rot5noCrossing}]
```

```
In[676]:= rot5polarLinesSmall[d_] := Table[{ColorData["Rainbow"][(ηiHj[i, jHavoid]) / 90.],
    Line[{{xHGal[ℓFROMr[rot5rPlusψSmall[i, d]], bFROMr[ rot5rPlusψSmall[i, d]]],
       yHGal[ℓFROMr[ rot5rPlusψSmall[i, d]], bFROMr[ rot5rPlusψSmall[i, d]]]},
      {xHGal[ℓFROMr[ rot5rPlusψSmall[i, -d]], bFROMr[ rot5rPlusψSmall[i, -d]]],
       yHGal[ℓFROMr[ rot5rPlusψSmall[i, -d]],
        bFROMr[ rot5rPlusψSmall[i, -d]]]}}]}, {i, rot5noCrossing}]

In[677]:= rot5setUPplotForMapToHavoid =
   Show[{ParametricPlot[{xHGal[-120., b], yHGal[-120., b]}, {b, -90, 90},
      PlotStyle → {Black, Thickness[0.002]}, PlotLegends →
       BarLegend[{"Rainbow", {0, 90.}}, LegendLabel → "η to H_avoid, deg."], PlotPoints → 60,
      PlotRange → {{-4.0, 3.5}, (7.5 / 11.0) {-3, 3}}, Axes → False, Frame → False],
     Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {b, -90, 90},
       PlotStyle → {Black, Thickness[0.002]}, PlotPoints → 60, PlotRange → {{-4.0, 3.5},
         (7.5 / 11.0) {-3, 3}}, Axes → False, Frame → False], {ℓ, -180., 180., 30}],
     Table[ParametricPlot[{xHGal[ℓ, b], yHGal[ℓ, b]}, {ℓ, -180., 180.},
       PlotStyle → {Black, Thickness[0.002]}, PlotPoints → 60], {b, -60, 60, 30}]}];

In[678]:= (*map of sources and ψ with σψ uncertainties in a coordinate system with H_avoid at (0,0,1).*)
rot5mapOfSourcesToHavoid =
  Show[{rot5setUPplotForMapToHavoid, Graphics[
     {PointSize[0.006], Text[StyleForm["H_Avoid", FontSize→14, FontWeight→"Plain"], {0, 1.85}],
      Text[StyleForm["0°", FontSize→14, FontWeight→"Plain"], {xHGal[0., -5.], yHGal[0., -5.]}],
      Text[StyleForm["90°", FontSize→14, FontWeight→"Plain"],
       {xHGal[90., -5.], yHGal[90., -5.]}], Text[StyleForm["180°", FontSize→14,
        FontWeight→"Plain"], {xHGal[180., -5.], yHGal[180., -5.]}],
      Text[StyleForm["-180°", FontSize→14, FontWeight→"Plain"],
       {xHGal[-180., -5.], yHGal[-180., -5.]}], Text[StyleForm["-90°", FontSize→14,
        FontWeight→"Plain"], {xHGal[-90., -5.], yHGal[-90., -5.]}],
      {Purple, PointSize[Large], Point[{xHGal[rot5ℓNGP, rot5bNGP], yHGal[rot5ℓNGP, rot5bNGP]}]},
      {Purple, Arrow[BezierCurve[{{-2.3, -1.5 + 0.15}, {-2.5, -1.0},
          {xHGal[rot5ℓNGP, rot5bNGP] - 0.04, yHGal[rot5ℓNGP, rot5bNGP]}}]]},
      Text[StyleForm["NGP", Purple, FontSize→14, FontWeight→"Plain"], {-2.3, -1.5}],
      {Cyan, PointSize[Large],
       Point[{xHGal[rot5ℓHalign + 360., rot5bHalign], yHGal[rot5ℓHalign + 360., rot5bHalign]}]},
      Text[StyleForm["H_Align", Black, FontSize→14, FontWeight→"Plain"],
       {xHGal[rot5ℓHalign + 360., rot5bHalign] - 0.15,
        yHGal[rot5ℓHalign + 360., rot5bHalign] + 0.2}], {Cyan, PointSize[Large],
       Point[{xHGal[rot5ℓHalign - 180., -rot5bHalign], yHGal[rot5ℓHalign - 180., -rot5bHalign]}]},
      Text[StyleForm["-H_Align", Black, FontSize→14, FontWeight→"Plain"],
       {xHGal[rot5ℓHalign - 180., -rot5bHalign] - 0.15,
        yHGal[rot5ℓHalign - 180., -rot5bHalign] + 0.20}],
      Text[StyleForm["Rotated Coordinate System II", FontSize→14, FontWeight→"Plain"],
       {0, -1.85}], {Thickness[0.003], rot5polarLinesBig[0.095]}, {Thickness[0.003],
       rot5polarLinesSmall[0.095]}, (*Sources S:*) rot5xyAitoffSourcesToHavoid
     }]}, ImageSize→ 1.2×432];
```

Out[679]=

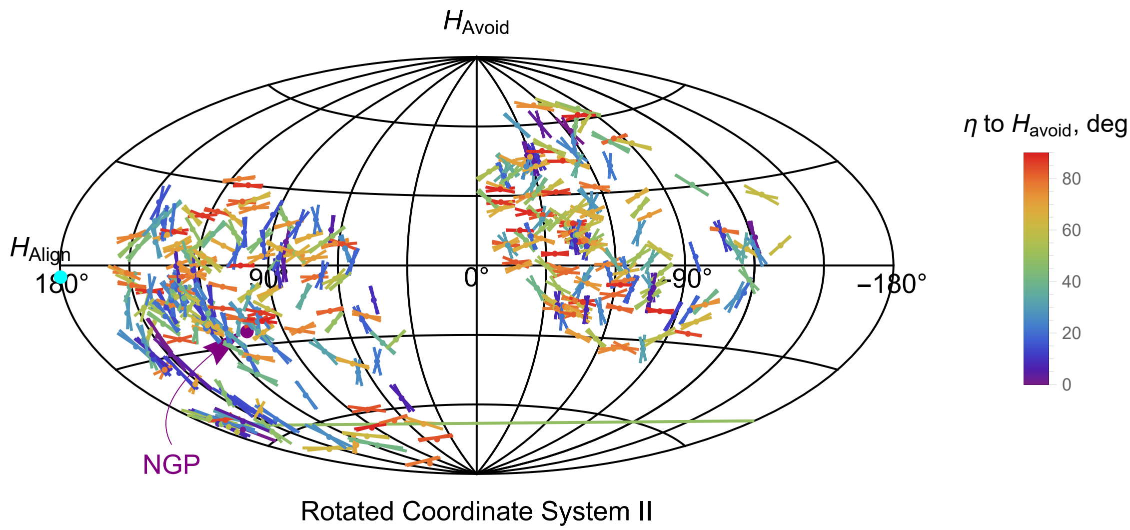


Figure A15.1 With the avoidance hub H_{Avoid} plotted at the virtual North Pole, a pine needle plot of the 355 QSO sources. Many of the polarization vectors point in the same direction as nearby parallels, so the vectors approximate Local Easts. Also note that the vectors that avoid the direction toward H_{Avoid}, shaded red and orange, occur throughout the sample, indicating the avoidance is widespread, an extreme-scale scale phenomenon. Many also point along meridians, so the distribution of polarization position angles appears bimodal, see Fig. A15.2.

In[681]:=
```
SetDirectory[NotebookDirectory[]]
(*Export["rot5mapOfSourcesToHavoid.pdf",rot5mapOfSourcesToHavoid,ImageSize→1.2 432]*)
```

Out[681]= C:\Users\momen\Dropbox\HOME_DESKTOP-0MRE5OJ\SendXXX_CJP_CEJPetc\SendViXra\20200715AlignmentMethod\20200715AlignmentMMAnotebooks\StarterKit

In[682]:=
```
Table[Histogram[Table[ηiHj[i, jHavoid], {i, nSrc}], {dη}, PlotLabel → dη], {dη, 1, 12}];
```

In[683]:=
```
histoEtaToHavoid = Histogram[Table[ηiHj[i, jHavoid], {i, nSrc}], {10.},
   PlotTheme → {"Scientific"}, FrameLabel → {"η_iH (to H = H_avoid), deg.", "ΔR"},
   PlotLabel → "Histogram of η_iH from i^th source ψ_i to H=H_avoid "];
```

Out[684]=

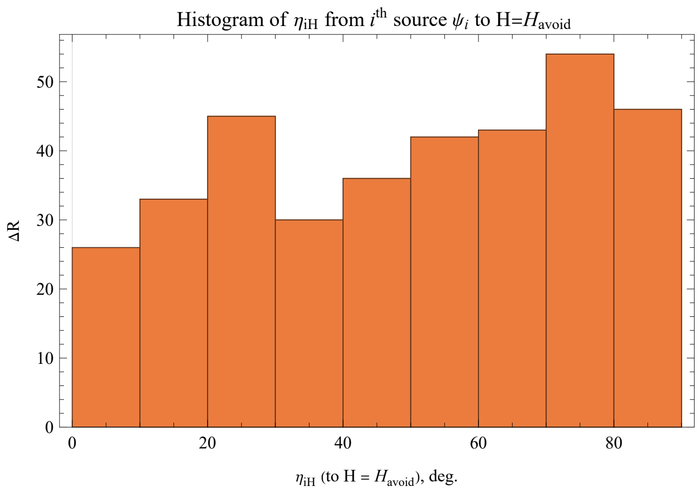


Figure A15.2 The distribution is somewhat bimodal, with peaks near η_{iH} = 25° and η_{iH} near 90°. Compare the secondary peak here at η_{iH} = 25° with the secondary peak in A14.2 at η_{iH} = 65° (= 90°−25°). Coincidence?

After Sec. A15, time and memory used are 612.065 seconds and 1 934 987 984 bytes.

A16. Plot the sources that are perfectly aligned or avoid perfectly

Plot two North poles, one with meridians for alignment with hub H_{align} as primary North Pole #1 and one with parallels for avoidance of hub H_{avoid} as alternate NorthPole #2.

Definitions

rNGP rNGP = (0,0,1), the radial unit vector to the North Galactic Pole
rHalignXHavoid radial unit vector to Halign×Havoid = HxH axis
ℓHalignXHavoid, *b*HalignXHavoid ℓ, *b* coords of HxH

idsForSalignsσψ ID #s for sources aligned with H = Halign within experimental error
rIDsForSalignsσψ unit radial vectors to the aligned sources in Galactic coords
rot4rIDsForSalignsσψ unit radial vectors to the aligned sources with Halign at the North pole

idsForSavoidsσψ ID #s for sources with η_{iH} ~ 90°, for H = Halign
rIDsForSavoidsσψ GCS unit radial vectors to the sources
rot4rIDsForSavoidsσψ unit radial vectors to the aligned sources with Halign at the North pole

ONLYaligns (BLUE) implies the source aligns perfectly, but does not perfectly avoid
ONLYavoids (ORANGE) implies the source avoids perfectly, but does not perfectly align

BOTH (GREEN) implies perfect alignment and avoidance for the source

polarLinesONLYalignsσψ, polarLinesONLYavoidsσψ, polarLinesBOTHalignAvoidσψ line segments to represent polarization directions
rotzTox a convenient rotation for drawing parallels to the avoidance hub direction, not used for anything else

preferredCoordsSeenFromNGP   the sphere with H_{align} at the North with a viewpoint above the NGP

In[690]:=
```
rHalignXHavoid0 = Cross[er[Hminℓ Fit, Hmin𝑏Fit], er[HmaxℓFit, Hmax𝑏Fit]];
rHalignXHavoid = rHalignXHavoid0 / (rHalignXHavoid0.rHalignXHavoid0)^(1/2);
ℓHalignXHavoid = ℓFROMr[rHalignXHavoid];
𝑏HalignXHavoid = 𝑏FROMr[rHalignXHavoid];
{ℓHalignXHavoid, 𝑏HalignXHavoid};
{ℓHalignXHavoid + 180., -𝑏HalignXHavoid};
```

In[696]:=
```
rNGP = {0., 0., 1.};
```

In[697]:=
```
idsForSalignsσψ = {};
For[i = 1, i ≤ nSrc, i++,
 If[ηiHj[i, jHalign] ≤ σψi[i] + aveHminErrorRadius, AppendTo[idsForSalignsσψ, i]]]
idsForSalignsσψ;
Length[%];
rIDsForSalignsσψ = Table[ri[i], {i, idsForSalignsσψ}];
rot4rIDsForSalignsσψ = Table[rot4HAlignToZ.ri[i], {i, idsForSalignsσψ}];
```

In[703]:=
```
idsForSavoidsσψ = {};
For[i = 1, i ≤ nSrc, i++,
 If[ηiHj[i, jHavoid] ≥ 90. - σψi[i] - aveHmaxErrorRadius, AppendTo[idsForSavoidsσψ, i]]]
idsForSavoidsσψ;
Length[%];
rIDsForSavoidsσψ = Table[ri[i], {i, idsForSavoidsσψ}];
rot4rIDsForSavoidsσψ = Table[rot4HAlignToZ.ri[i], {i, idsForSavoidsσψ}];
```

In[709]:=
```
idsForSBOTHalignAvoidσψ = Intersection[idsForSalignsσψ, idsForSavoidsσψ];
Length[%];
rIDsForSBOTHalignAvoidσψ = Table[ri[i], {i, idsForSBOTHalignAvoidσψ}];
rot4rIDsForSBOTHalignAvoidσψ = Table[rot4HAlignToZ.ri[i], {i, idsForSBOTHalignAvoidσψ}];
```

In[713]:=
```
idsForSONLYaligns = Complement[idsForSalignsσψ, idsForSBOTHalignAvoidσψ];
rot4rIDsForSONLYalignsσψ = Table[rot4HAlignToZ.ri[i], {i, idsForSONLYaligns }];
idsForSONLYavoids = Complement[idsForSavoidsσψ, idsForSBOTHalignAvoidσψ];
rot4rIDsForSONLYavoidsσψ = Table[rot4HAlignToZ.ri[i], {i, idsForSONLYavoids}];
```

In[717]:=
```
polarLinesONLYalignsσψ[d_] := Table[{(*ColorData["Rainbow"][(ηiHj[i,jHalign])/90.]*)
    {Thick, Line[1.02 {rot4HAlignToZ.er[ℓFROMr[ rPlusψ[i, d]], bFROMr[ rPlusψ[i, d]]],
        rot4HAlignToZ.er[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]]}]}},
  {i, Complement[idsForSalignsσψ, idsForSBOTHalignAvoidσψ]}]
polarLinesONLYavoidsσψ[d_] := Table[{(*ColorData["Rainbow"][(ηiHj[i,jHalign])/90.]*)
    {Thick, Line[1.02 {rot4HAlignToZ.er[ℓFROMr[ rPlusψ[i, d]], bFROMr[ rPlusψ[i, d]]],
        rot4HAlignToZ.er[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]]}]}},
  {i, Complement[idsForSavoidsσψ, idsForSBOTHalignAvoidσψ]}]
polarLinesBOTHalignAvoidσψ[d_] :=
 Table[{(*ColorData["Rainbow"][(ηiHj[i,jHalign])/90.]*)
    {Thick, Line[1.02 {rot4HAlignToZ.er[ℓFROMr[ rPlusψ[i, d]], bFROMr[ rPlusψ[i, d]]],
        rot4HAlignToZ.er[ℓFROMr[ rPlusψ[i, -d]], bFROMr[ rPlusψ[i, -d]]]}]}},
  {i, idsForSBOTHalignAvoidσψ}]
```

In[720]:=
```
rotzTox = RotationMatrix[{{0, 0, 1.}, rot4HAlignToZ.er[HmaxℓFit, HmaxbFit]}];
```

In[721]:=
```
preferredCoordsSeenFromNGP = Show[{
    ParametricPlot3D[Table[er[ℓ, b], {ℓ, 0., 340., 20.}], {b, -90., 90.},
     PlotRange → {{-1.35, 1.35}, {-1.35, 1.35}, {-1.35, 1.35}}, Boxed → False,
     Axes → False, PlotStyle → Directive[Purple, Thick], ImageSize → 0.8 × 432,
     ViewPoint → {3 cos[105.], 3 sin[105.], 1.}, PlotTheme → "Scientific"],
    ParametricPlot3D[Table[rotzTox.er[ℓ, b], {b, -80., 80., 20}], {ℓ, 0., 360.},
     PlotStyle → Directive[Red, Thick]], Graphics3D[{{Opacity[1.00], White, Sphere[]},
      PointSize[Large], {Black, Thick, Arrow[{{0, 0, 0}, {0, 0, 1.35}}]},
      {Black, Thick, Arrow[{{0, 0, 0}, 1.35 rotzTox.{0, 0, 1.}}]},
      {Black, Thick, Line[{{0, 0, 0}, -1.2 rotzTox.{0, 0, 1.}}]}, Green,
      Point[1.02 rot4rIDsForSBOTHalignAvoidσψ], polarLinesBOTHalignAvoidσψ[0.15],
      Blue, Point[1.02 rot4rIDsForSONLYalignsσψ], polarLinesONLYalignsσψ[0.15], Orange,
      Point[1.02 rot4rIDsForSONLYavoidsσψ], polarLinesONLYavoidsσψ[0.15], Black,
      Text[StyleForm["H_Align", FontSize → 14, FontWeight → "Plain"], {0.4, 0, 1.2}],
      Text[StyleForm["H_Avoid", FontSize → 14, FontWeight → "Plain"],
       rot4HAlignToZ.er[HmaxℓFit, HmaxbFit] + {0.3, 0, -0.1}],
      Text[StyleForm["H×H", FontSize → 14, FontWeight → "Plain"],
       1.02 rot4HAlignToZ.rHalignXHavoid + {0.2, 0.1, 0.05}],
      {Black, Thick, Arrow[{{0, 0, 0}, 1.35 rot4HAlignToZ.rHalignXHavoid}]},
      Point[-1.02 rot4HAlignToZ.rHalignXHavoid], Black,
      Text[StyleForm["NGP", FontSize → 14, FontWeight → "Plain"], {-1., 1.0, 1.2}],
      {Black, Arrow[BezierCurve[{{-1., 0.8, 1.1}, {-1.1, 1.2, +0.2},
          1.02 rot4HAlignToZ.rNGP + {-0.05, 0., 0.}}]]},
      Point[1.02 rot4HAlignToZ.rNGP], Red, Point[rot4HAlignToZ.er[HmaxℓFit, HmaxbFit]],
      Point[-rot4HAlignToZ.er[HmaxℓFit, HmaxbFit]]}]}];
```

Out[722]=

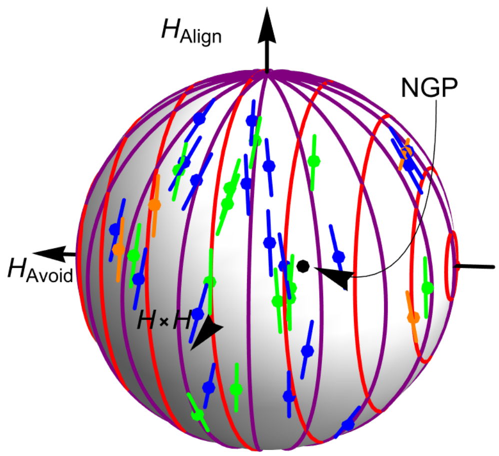


Figure A16.1: Perfect alignment and avoidance, within experimental error. The meridians have H_{Align} as the virtual North Pole, while the parallels have H_{Avoid} as their virtual North Pole. Blue indicates alignment with H_{align}, Orange means avoidance of H_{avoid}, and Green implies both.

In[724]:=
```
SetDirectory[NotebookDirectory[]]
(*Export["preferredCoordsSeenFromNGP.pdf",
 preferredCoordsSeenFromNGP,ImageSize→0.8 432]*)
```

Out[724]= C:\Users\momen\Dropbox\HOME_DESKTOP-0MRE5OJ\SendXXX_CJP_CEJPetc\SendViXra\20200715AlignmentMethod\20200715AlignmentMMAnotebooks\StarterKit

```
In[725]:= preferredCoordsSeenFromSGP = Show[{
    ParametricPlot3D[Table[er[ℓ, b], {ℓ, 0., 340., 20.}], {b, -90., 90.},
     PlotRange → {{-1.35, 1.35}, {-1.35, 1.35}, {-1.35, 1.35}}, Boxed → False,
     Axes → False, PlotStyle → Directive[Purple, Thick], ImageSize → 0.8 × 432,
     ViewPoint → -{3 cos[105.], 3 sin[105.], 1.}, PlotTheme → "Scientific"],
    ParametricPlot3D[Table[rotzTox.er[ℓ, b], {b, -80., 80., 20}], {ℓ, 0., 360.},
     PlotStyle → Directive[Red, Thick]], Graphics3D[{{Opacity[1.00], White, Sphere[]},
      PointSize[Large], {Black, Thick, Arrow[{{0, 0, 0}, {0, 0, 1.35}}]},
      {Black, Thick, Arrow[{{0, 0, 0}, 1.35 rotzTox.{0, 0, 1.}}]},
      {Black, Thick, Line[{{0, 0, 0}, -1.2 rotzTox.{0, 0, 1.}}]}, Green,
      Point[1.02 rot4rIDsForSBOTHalignAvoidσψ], polarLinesBOTHalignAvoidσψ[0.15],
      Blue, Point[1.02 rot4rIDsForSONLYalignsσψ], polarLinesONLYalignsσψ[0.15], Orange,
      Point[1.02 rot4rIDsForSONLYavoidsσψ], polarLinesONLYavoidsσψ[0.15], Black,
      Text[StyleForm["H_Align", FontSize → 14, FontWeight → "Plain"], {0.4, 0, 1.2}],
      Text[StyleForm["H_Avoid", FontSize → 14, FontWeight → "Plain"],
       rot4HAlignToZ.er[HmaxℓFit, HmaxbFit] + {0.3, 0, -0.1}],
      Black, Text[StyleForm["-H×H", FontSize → 14, FontWeight → "Plain"],
       -(1.02 rot4HAlignToZ.rHalignXHavoid + {0.2, 0.1, 0.05})],
      {Black, Thick, (*Arrow*)Line[{{0, 0, 0}, -4.45 rot4HAlignToZ.rHalignXHavoid}]},
      (*Point[-1.02rot4HAlignToZ.rHalignXHavoid]*), Black,
      Text[StyleForm["SGP", FontSize → 14, FontWeight → "Plain"], -{-1., 1.0, 1.2}],
      {Black, Arrow[BezierCurve[{-{-1., 0.8, 1.1}, -{-1.1, 1.2, +0.2},
          -(1.02 rot4HAlignToZ.rNGP + {-0.05, 0., 0.})}]]},
      Point[-1.02 rot4HAlignToZ.rNGP], Red, Point[rot4HAlignToZ.er[HmaxℓFit, HmaxbFit]],
      Point[-rot4HAlignToZ.er[HmaxℓFit, HmaxbFit]]}]}];
```

Out[726]=

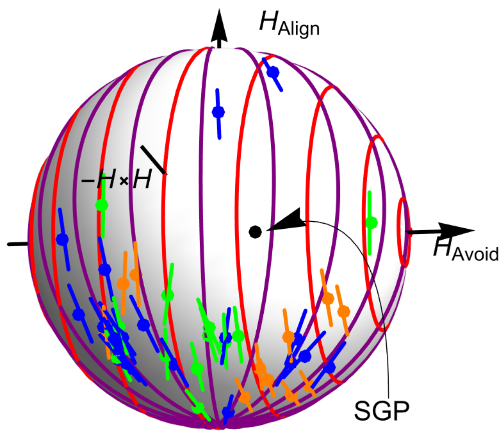


Figure A16.2: Same as Fig.16.1, except viewed from the South Galactic Pole.

After Sec. A16, time and memory used are 613.143 seconds and 1 937 721 264 bytes.

A17. Computer Memory and Time Usage

In[732]:=
```
lpComputerMemVsTime = ListPlot[
  Table[{sectionTime[i] / 60., sectionMemory[i] / 10.^9}, {i, 16}] → Table[i, {i, 16}],
  PlotLabel → "Time and space consumed, section-by-section", FrameLabel →
   {"computer time used, min.", "Memory in Use, Gbytes"}, PlotTheme → "Detailed"];
```

Out[733]=

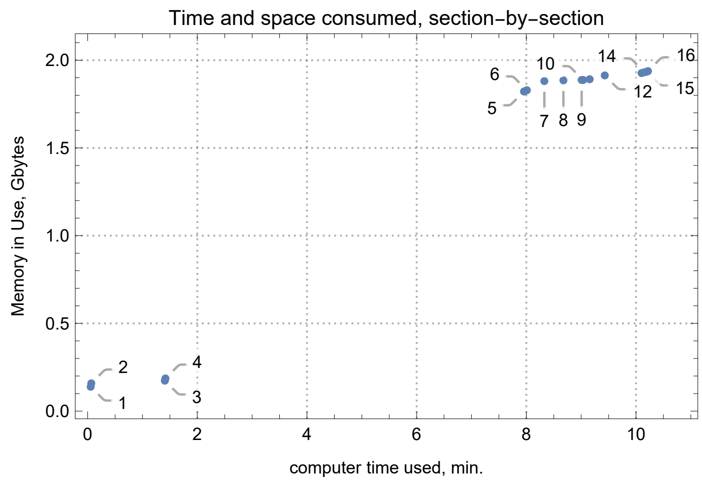


Figure 17.1: For this computing session, the computer memory in use as a function of computer time used, measured at the end of each section, all but this section. The points are labeled with Section numbers.