check-ga-on-icrf2

August 23, 2019

In this notebook I will look at the radio source catalogs (.sou file) from a VLBI solution denoted as `icrf2-ga-09` and `icrf2-nga-09`.

The solution `icrf2-nga-09` used sessions and analysis strategies that are similar to the ICRF2 solution.

The only difference between `icrf2-ga-09` and `icrf2-nga-09` is that we modeled the Galactic aberration effect in `icrf2-ga-09`.

Three comparisons need to do done:

- icrf2-nga-09 vs. ICRF2
- icrf2-nga-09 vs. Gaia DR2
- icrf2-ga-09 vs. Gaia DR2

The first comparison is to check if I can nearly reproduce the ICRF2 solution.

[1]:
```
from astropy.table import Table, Column
import astropy.units as u
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np

import bottleneck as bn
```

The two solutions are loaded from .sou files and there are **3581** sources in both solution.

[2]:
```
from my_progs.vlbi.read_sou import read_sou

solnga = read_sou("../data/icrf2-nga-09/icrf2-nga-09.sou")
solga = read_sou("../data/icrf2-ga-09/icrf2-ga-09.sou")
```

[3]:
```
solga
```

[3]:
```
<Table masked=True length=3581>
ivs_name icrf_name  iers_name class … used_sess total_sess beg_epoch end_epoch
                                    …                          MJD       MJD
  str8     str10       str8     str1 …   int64      int64     float64   float64
-------- ---------- --------- ----- … --------- ---------- --------- ---------
0000+212 J0003+2129  0000+212     C …         2          2   50084.0   50155.0
0000-160 J0003-1547  0000-160     - …         1          1   54817.0   54817.0
```

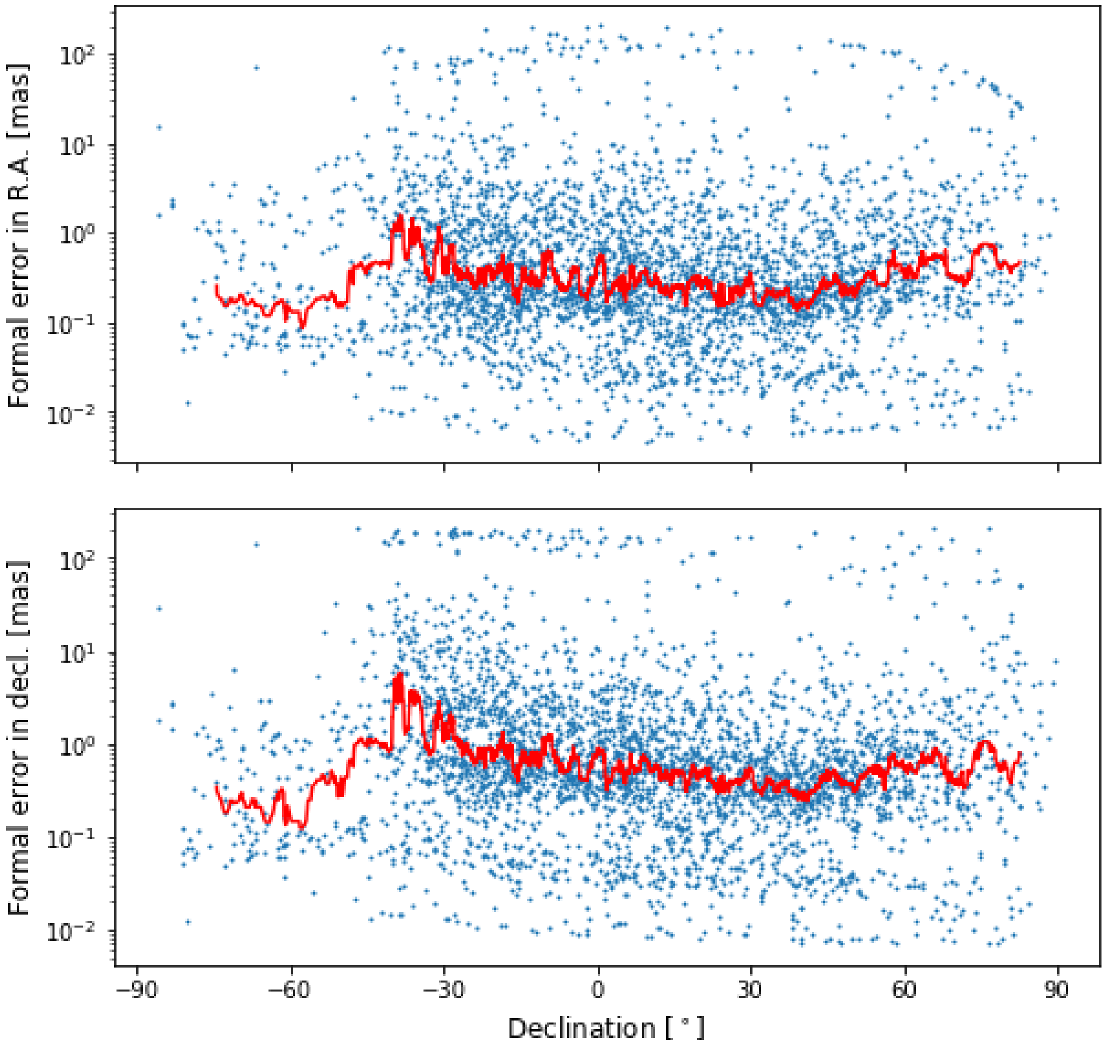
0000-197	J0003-1927	0000-197	C	…	2	2	50631.0	50687.0
0000-199	J0003-1941	0000-199	N	…	1	1	54087.0	54087.0
0001+459	J0004+4615	0001+459	C	…	1	1	50305.0	50305.0
0001+478	J0003+4807	0001+478	N	…	1	1	50305.0	50305.0
0001-120	J0004-1148	0001-120	C	…	3	3	50575.0	53133.0
0002+051	J0005+0524	0002+051	C	…	1	1	49913.0	49913.0
0002+200	J0004+2019	0002+200	C	…	3	4	52408.0	52982.0
…	…	…	…	…	…	…	…	…
OJ287	J0854+2006	0851+202	C	…	3446	3459	44202.0	54906.0
OP326	J1317+3425	1315+346	C	…	39	40	47945.0	54886.0
OQ172	J1445+0958	1442+101	C	…	52	54	47010.0	54900.0
OW-015	J2011-0644	2008-068	C	…	34	48	48345.0	54740.0
SN1993J	J0955+6901	0951+692	C	…	3	4	49224.0	49266.0
UG01841	J0223+4259	0220+427	C	…	2	2	51448.0	53067.0
UG03927	J0737+5941	0733+597	C	…	4	4	49576.0	54087.0
UGC02748	J0327+0233	0325+023	C	…	3	4	51491.0	53067.0
VELA-G	J0833-4441	0831-445	N	…	5	7	48042.0	49894.0
VR422201	J2202+4216	2200+420	C	…	1040	1047	44088.0	54893.0

A good habit is to check the dependency of the formal uncertainties on the declination. One can find that they show similar declination-dependence trend, but the dependency of the declination uncertainty is stronger.

[4]:

```python
fig, (ax0, ax1) = plt.subplots(figsize=(8, 8), nrows=2, sharex=True)

ax0.plot(solga["dec"], solga["ra_err"], ".", ms=2)
ax1.plot(solga["dec"], solga["dec_err"], ".", ms=2)

# Running median
temp = Table(solga)
temp.sort("dec")

window = 50
decmd = bn.move_median(temp["dec"], window=window)
raerrmd = bn.move_median(temp["ra_err"], window=window)
decerrmd = bn.move_median(temp["dec_err"], window=window)

ax0.plot(decmd, raerrmd, "r")
ax1.plot(decmd, decerrmd, "r")

ax0.set_yscale("log")
ax1.set_yscale("log")

ax1.set_xlabel("Declination [$^\circ$]", fontsize=12)
ax0.set_ylabel("Formal error in R.A. [mas]", fontsize=12)
ax1.set_ylabel("Formal error in decl. [mas]", fontsize=12)
```

```
ax1.set_xticks(np.arange(-90, 91, 30))

plt.subplots_adjust(hspace=0.1)
```



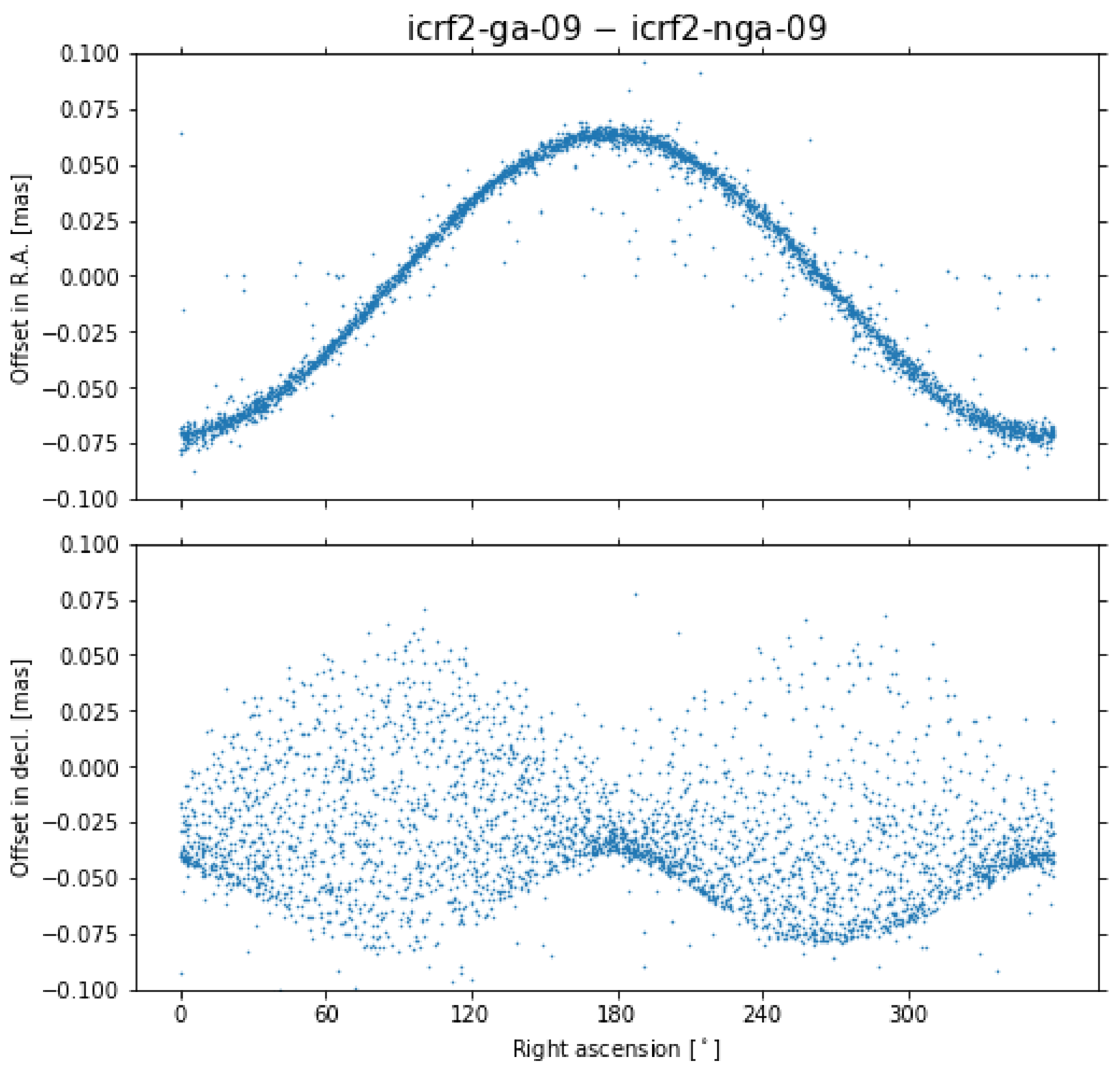
1 1) Compare icrf2-nga-09 with icrf2-ga-09

First we take a look at the positional offsets between these two solutions, which are purely caused by the Galactic aberration with the assumed direction and magnitude.

[5]:
```
from my_progs.catalog.pos_diff import radio_cat_diff_calc

soldif = radio_cat_diff_calc(solga, solnga, sou_name="iers_name")
```

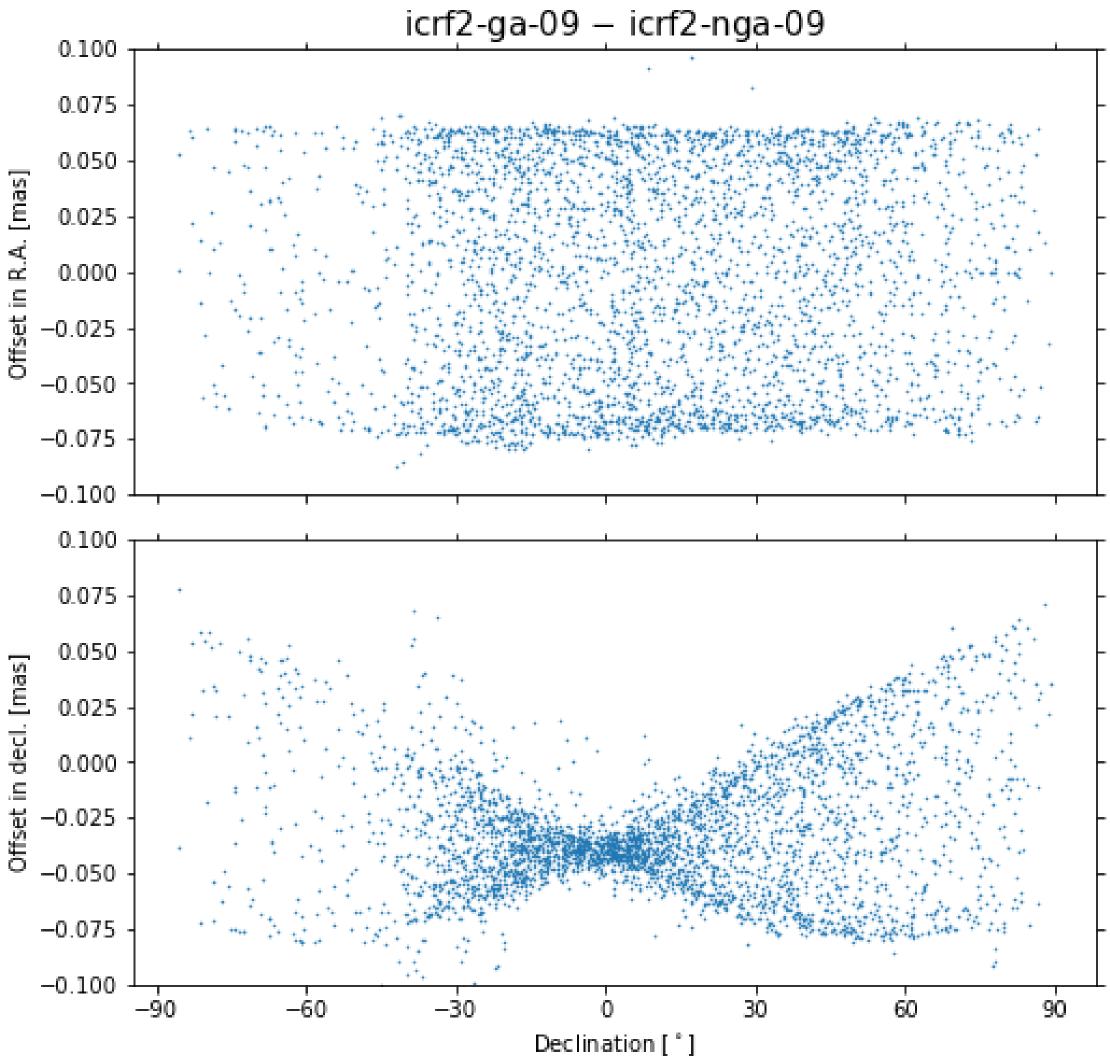
[6]:

```
fig, (ax0, ax1) = plt.subplots(figsize=(8, 8), nrows=2, sharex=True)

ax0.plot(soldif["ra"], soldif["dra"], ".", ms=1)
ax1.plot(soldif["ra"], soldif["ddec"], ".", ms=1)

ax0.set_ylabel("Offset in R.A. [mas]")

ax1.set_xlabel("Right ascension [$^\circ$]")
ax1.set_ylabel("Offset in decl. [mas]")

ax1.set_xticks(np.arange(0, 360, 60))

ax0.set_ylim([-0.1, 0.1])
ax1.set_ylim([-0.1, 0.1])

ax0.set_title("icrf2-ga-09 $-$ icrf2-nga-09", fontsize=15)

ax0.xaxis.set_ticks_position("both")
ax0.yaxis.set_ticks_position("both")
ax1.xaxis.set_ticks_position("both")
ax1.yaxis.set_ticks_position("both")

plt.subplots_adjust(hspace=0.1)
```



[7]:
```
fig, (ax0, ax1) = plt.subplots(figsize=(8, 8), nrows=2, sharex=True)

ax0.plot(soldif["dec"], soldif["dra"], ".", ms=1)
ax1.plot(soldif["dec"], soldif["ddec"], ".", ms=1)

ax0.set_ylabel("Offset in R.A. [mas]")

ax1.set_xlabel("Declination [$^\circ$]")
ax1.set_ylabel("Offset in decl. [mas]")

ax1.set_xticks(np.arange(-90, 91, 30))

ax0.set_ylim([-0.1, 0.1])
ax1.set_ylim([-0.1, 0.1])
```

```python
ax0.set_title("icrf2-ga-09 $-$ icrf2-nga-09", fontsize=15)

ax0.xaxis.set_ticks_position("both")
ax0.yaxis.set_ticks_position("both")
ax1.xaxis.set_ticks_position("both")
ax1.yaxis.set_ticks_position("both")

plt.subplots_adjust(hspace=0.1)
```



[8]:
```python
# My modules
from my_progs.catalog.vsh_deg2_cor import vsh_deg02_fitting, residual_calc02
from my_progs.catalog.write_output import print_vsh1_corr, print_vsh2_corr
```

```
[9]: # Try with all the sources
# Transform columns into np.array
dra = np.array(soldif["dra"])
ddec = np.array(soldif["ddec"])
dra_err = np.array(soldif["dra_err"])
ddec_err = np.array(soldif["ddec_err"])
ra_rad = np.array(soldif["ra"].to(u.radian))
dec_rad = np.array(soldif["dec"].to(u.radian))
dra_ddec_cov = np.array(soldif["dra_ddec_cov"])

# Transformation parameters
# l_max = 2
w2_all, sig2_all, corrcoef2_all, _, _, _ = vsh_deg02_fitting(
    dra, ddec, ra_rad, dec_rad, dra_err, ddec_err,
    cov=dra_ddec_cov, elim_flag="None")

# mas -> uas
w2 = w2_all * 1.e3
sig2 = sig2_all * 1.e3

# Print results
print("Estimates (%6d sources)\n"
      "----------------------------------------------"
      "----------------------------------------------\n"
      "              Rotation [uas]                  "
      "                 Glide [uas]               \n"
      "     x             y             z           "
      "     x             y             z\n"
      "----------------------------------------------"
      "----------------------------------------------\n"
      "%+4.0f +/- %3.0f  %+4.0f +/- %3.0f  %+4.0f +/- %3.0f   "
      "  %+4.0f +/- %3.0f  %+4.0f +/- %3.0f  %+4.0f +/- %3.0f\n"
      "----------------------------------------------"
      "----------------------------------------------\n" %
      (dra.size,
       w2[3], sig2[3], w2[4], sig2[4], w2[5], sig2[5],
       w2[0], sig2[0], w2[1], sig2[1], w2[2], sig2[2]))

quad_names = Column(["ER22", "EI22", "ER21", "EI21", "E20",
                     "MR22", "MI22", "MR21", "MI21", "M20"])
t_quad = Table([quad_names, w2[6:], sig2[6:]],
               names=["Quadrupolar term", "Estimate", "Error"])
t_quad["Estimate"].format = "%5.0f"
t_quad["Error"].format = "%5.0f"
print(t_quad)
```

```python
print("Correlation coefficient between parameters in 'l_max=2' fit")
print_vsh2_corr(corrcoef2_all, deci_digit=1, included_one=False)
```

```
Estimates (  3581 sources)
--------------------------------------------------------------------------------
------------
              Rotation [uas]                                   Glide [uas]
     x             y             z                x             y             z
--------------------------------------------------------------------------------
------------
  -1 +/-   0    +3 +/-   0    -4 +/-   0     -5 +/-   0   -68 +/-   0   -40 +/-
0
--------------------------------------------------------------------------------
------------

Quadrupolar term Estimate Error
---------------- -------- -----
            ER22        0     0
            EI22       -0     0
            ER21       -0     0
            EI21        0     0
             E20        1     0
            MR22        0     0
            MI22       -0     0
            MR21        1     0
            MI21       -1     0
             M20        0     0
Correlation coefficient between parameters in 'l_max=2' fit
          R1     R2     R3     D1     D2     D3   E22R   E22I   E21R   E21I    E20   M22R
M22I  M21R  M21I
    R2  +0.0
    R3  +0.0  +0.1
    D1  +0.0  +0.6  +0.0
    D2  -0.6  +0.0  -0.0  +0.0
    D3  -0.0  +0.0  -0.1  +0.0  +0.0
  E22R  -0.0  -0.1  -0.0  -0.1  +0.0  +0.1
  E22I  +0.0  -0.0  +0.0  -0.0  +0.0  -0.0  -0.0
  E21R  +0.5  +0.1  +0.0  +0.0  -0.1  -0.1  -0.1  +0.0
  E21I  -0.0  -0.5  -0.0  -0.1  +0.0  -0.0  +0.1  +0.0  -0.0
   E20  -0.1  -0.0  -0.5  -0.1  -0.0  +0.0  -0.0  -0.0  -0.0  -0.0
  M22R  +0.0  +0.0  -0.0  +0.1  -0.0  +0.0  +0.0  -0.3  +0.0  -0.0  +0.1
  M22I  -0.0  -0.0  -0.0  -0.0  +0.0  +0.0  +0.3  +0.0  +0.0  +0.0  +0.0  +0.0
  M21R  +0.0  +0.6  +0.0  +0.5  -0.0  +0.0  -0.1  -0.1  +0.0  -0.3  -0.0  -0.0
+0.0
  M21I  +0.6  -0.0  -0.0  -0.0  -0.5  -0.0  -0.0  -0.0  +0.3  -0.0  -0.1  +0.0
+0.0  +0.0
   M20  +0.0  -0.0  +0.0  -0.1  -0.0  -0.4  -0.1  +0.0  +0.0  +0.1  -0.0  -0.0
```

```
+0.0  -0.0  -0.0
```

The offset is generally on the order of 25 μas and we found an offset in declination of $-50\,\mu as$, which could explain part of the declination bias in the ICRF2.

Later we will make certain conclusion.

2 2) Compare icrf2-nga-09 versus ICRF2

[10]:
```python
from my_progs.catalog.read_icrf import read_icrf2
icrf2 = read_icrf2()
```

[11]:
```python
# Crossmatch between ICRF2 and solution
icrfdif1 = radio_cat_diff_calc(solnga, icrf2, sou_name="iers_name")
```

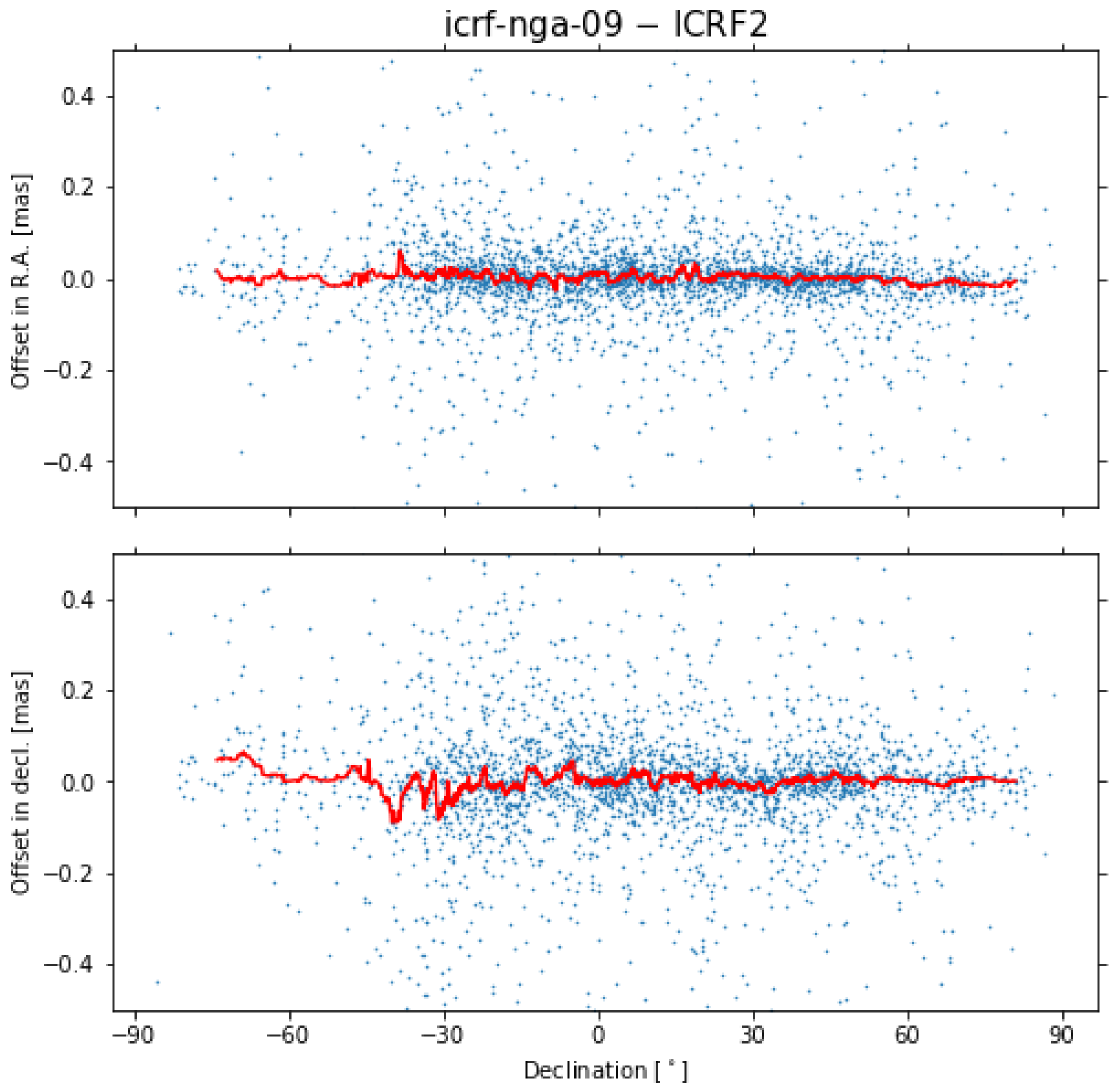
The cross-match gives a sample of 3362 rather than 3414 common sources.

Possibly these sources are VCS sources.

But it make no sense here.

[12]:
```python
fig, (ax0, ax1) = plt.subplots(figsize=(8, 8), nrows=2, sharex=True)

ax0.plot(icrfdif1["dec"], icrfdif1["dra"], ".", ms=1)
ax1.plot(icrfdif1["dec"], icrfdif1["ddec"], ".", ms=1)

ax0.set_ylabel("Offset in R.A. [mas]")

ax1.set_xlabel("Declination [$^\circ$]")
ax1.set_ylabel("Offset in decl. [mas]")

# Running median
temp1 = Table(icrfdif1)
temp1.sort("dec")

window = 50
decmd1 = bn.move_median(temp1["dec"], window=window)
dramd1 = bn.move_median(temp1["dra"], window=window)
ddecmd1 = bn.move_median(temp1["ddec"], window=window)

ax0.plot(decmd1, dramd1, "r")
ax1.plot(decmd1, ddecmd1, "r")

ax1.set_xticks(np.arange(-90, 91, 30))

ax0.set_ylim([-0.5, 0.5])
ax1.set_ylim([-0.5, 0.5])

ax0.xaxis.set_ticks_position("both")
```

```
ax0.yaxis.set_ticks_position("both")
ax1.xaxis.set_ticks_position("both")
ax1.yaxis.set_ticks_position("both")

ax0.set_title("icrf-nga-09 $-$ ICRF2", fontsize=15)

plt.subplots_adjust(hspace=0.1)
```



The offset scatter in the declination is obviously larger than in the RA.

For most sources, they are below in 0.1 mas, albeit some large offset over 1 mas.

[13]:
```
# Try with all the sources
# Transform columns into np.array
```

```python
dra = np.array(icrfdif1["dra"])
ddec = np.array(icrfdif1["ddec"])
dra_err = np.array(icrfdif1["dra_err"])
ddec_err = np.array(icrfdif1["ddec_err"])
ra_rad = np.array(icrfdif1["ra"].to(u.radian))
dec_rad = np.array(icrfdif1["dec"].to(u.radian))
dra_ddec_cov = np.array(icrfdif1["dra_ddec_cov"])

# Transformation parameters
# l_max = 2
w2_all, sig2_all, corrcoef2_all, _, _, _ = vsh_deg02_fitting(
    dra, ddec, ra_rad, dec_rad, dra_err, ddec_err,
    cov=dra_ddec_cov, elim_flag="None")

# mas -> uas
w2 = w2_all * 1.e3
sig2 = sig2_all * 1.e3

# Print results
print("Estimates (%6d sources)\n"
      "----------------------------------------------"
      "----------------------------------------------\n"
      "               Rotation [uas]                  "
      "                  Glide [uas]               \n"
      "     x             y             z           "
      "     x             y             z\n"
      "----------------------------------------------"
      "----------------------------------------------\n"
      "%+4.0f +/- %3.0f  %+4.0f +/- %3.0f  %+4.0f +/- %3.0f  "
      "  %+4.0f +/- %3.0f  %+4.0f +/- %3.0f  %+4.0f +/- %3.0f\n"
      "----------------------------------------------"
      "----------------------------------------------\n" %
      (dra.size,
       w2[3], sig2[3], w2[4], sig2[4], w2[5], sig2[5],
       w2[0], sig2[0], w2[1], sig2[1], w2[2], sig2[2]))

quad_names = Column(["ER22", "EI22", "ER21", "EI21", "E20",
                     "MR22", "MI22", "MR21", "MI21", "M20"])
t_quad = Table([quad_names, w2[6:], sig2[6:]],
               names=["Quadrupolar term", "Estimate", "Error"])
t_quad["Estimate"].format = "%5.0f"
t_quad["Error"].format = "%5.0f"
print(t_quad)


print("Correlation coefficient between parameters in 'l_max=2' fit")
print_vsh2_corr(corrcoef2_all, deci_digit=1, included_one=False)
```

```
Estimates (  3362 sources)
--------------------------------------------------------------------------------
------------
              Rotation [uas]                                  Glide [uas]
     x             y             z                x             y             z
--------------------------------------------------------------------------------
------------
  -2 +/-   2    +1 +/-   2    +1 +/-   1      +2 +/-   1    -1 +/-   1    -2 +/-
2
--------------------------------------------------------------------------------
------------

Quadrupolar term Estimate Error
---------------- -------- -----
            ER22        0     1
            EI22       -0     1
            ER21        3     2
            EI21        4     2
             E20        4     2
            MR22        1     1
            MI22        1     1
            MR21       -2     2
            MI21        1     2
             M20       -0     1
Correlation coefficient between parameters in 'l_max=2' fit
          R1    R2    R3    D1    D2    D3  E22R  E22I  E21R  E21I   E20  M22R
M22I  M21R  M21I
    R2  +0.0
    R3  -0.0  +0.0
    D1  +0.0  +0.5  +0.0
    D2  -0.5  +0.0  -0.0  +0.0
    D3  -0.0  +0.0  -0.1  -0.0  +0.0
  E22R  +0.0  -0.0  -0.0  -0.0  +0.0  +0.1
  E22I  +0.0  -0.0  +0.0  +0.0  +0.0  +0.1  +0.0
  E21R  +0.4  +0.1  -0.0  +0.0  -0.1  -0.0  -0.0  +0.0
  E21I  +0.0  -0.4  -0.0  -0.1  +0.0  +0.0  +0.1  +0.0  +0.0
   E20  -0.0  -0.0  -0.4  -0.0  -0.0  +0.0  -0.0  -0.0  -0.0  -0.0
  M22R  +0.0  -0.0  -0.1  +0.0  -0.0  -0.0  +0.1  -0.3  +0.0  +0.0  +0.1
  M22I  -0.0  -0.0  -0.0  +0.0  -0.0  +0.0  +0.3  +0.1  +0.0  +0.0  +0.1  +0.0
  M21R  -0.0  +0.4  +0.0  +0.4  -0.0  -0.0  -0.0  -0.0  -0.0  -0.2  -0.0  -0.0
+0.0
  M21I  +0.4  -0.0  -0.0  -0.0  -0.4  -0.0  -0.0  -0.0  +0.3  -0.0  -0.0  +0.1
+0.0  +0.0
   M20  +0.0  +0.0  +0.0  -0.0  -0.0  -0.3  -0.1  -0.1  +0.0  +0.0  -0.0  -0.0
-0.0  -0.0  -0.0
```

Basing on all the common sources between the two data sets and modelling the positional offsets to extract the global (long wavelength) differences, we found no significant terms.

As a result, I reproduced a catalog having similar global properties to the ICRF2.

3 2) icrf2-nga/ga-09 versus the Gaia DR2

[14]:
```
from my_progs.catalog.read_gaia import read_dr2_iers

gdr2 = read_dr2_iers()

# Crossmatch between Gaia DR2 and solution
gaiadif1 = radio_cat_diff_calc(solnga, gdr2, sou_name="iers_name")

gaiadif2 = radio_cat_diff_calc(solga, gdr2, sou_name="iers_name")

gaiadif1
```

[14]:
```
<Table masked=True length=2360>
iers_name       ra_err_1       …        nor_dec              nor_sep
                  mas          …
   str8         float64        …        float64              float64
--------- -------------------- … --------------------- -------------------
 0000-197 0.32661929193625466 …    1.3563059271796065   2.64482732802601
 0000-199  1.6302775548424904 …  -0.42022179627092204 0.9897627368510226
 0001+459 0.20743503589630327 …   0.11881064252786361 1.0301596172987877
 0001-120 0.09631485130652458 …    1.2343140113986901  4.007359259633918
 0002+051  1.3841229462433025 …   -1.2898815480439287 1.2928061090499665
 0002+541 0.49891369488195203 …    2.0951207510502625   2.10843079344123
 0002-170 0.26402690905814885 …    -1.910266784137314 2.0358973930097344
 0002-478 0.06337782678662841 …   -0.4055542805407914 0.6943509208727018
 0003+123  0.5777470442710799 …  -0.12080383315358702 0.7435491642109877
      …                  … …                     …                  …
 2355-106 0.01790467513317899 …    0.7383735724024061 2.4502080196773406
 2355-291   89.46571632534463 … -0.017558686407799295 0.7291986082896366
 2355-534 0.06836910385226462 …     1.905989062067475  6.794382789900244
 2356+196 0.11930573809938169 …    0.7438364016468374 1.8482592426170907
 2356+385 0.00872305978042032 …  -0.06863941236182419  1.820274647690552
 2356+390 0.18444153749895628 …   -0.5955968519074689  2.830558338598594
 2357-318 0.06816581467049355 …     2.117066104963246 2.2002967249724685
 2357-326  0.3018441821147953 …    1.4542046854474326  17.71630348461664
 2358-161 0.31119750453488076 …  -0.18949460847349578 1.5213055073471848
 2359-221  11.002615216965625 …    0.9534586969652366  2.643986465902531
```

We found **2360** common sources between icrf2-ga-09/icrf2-nga-09 and *Gaia* DR2.

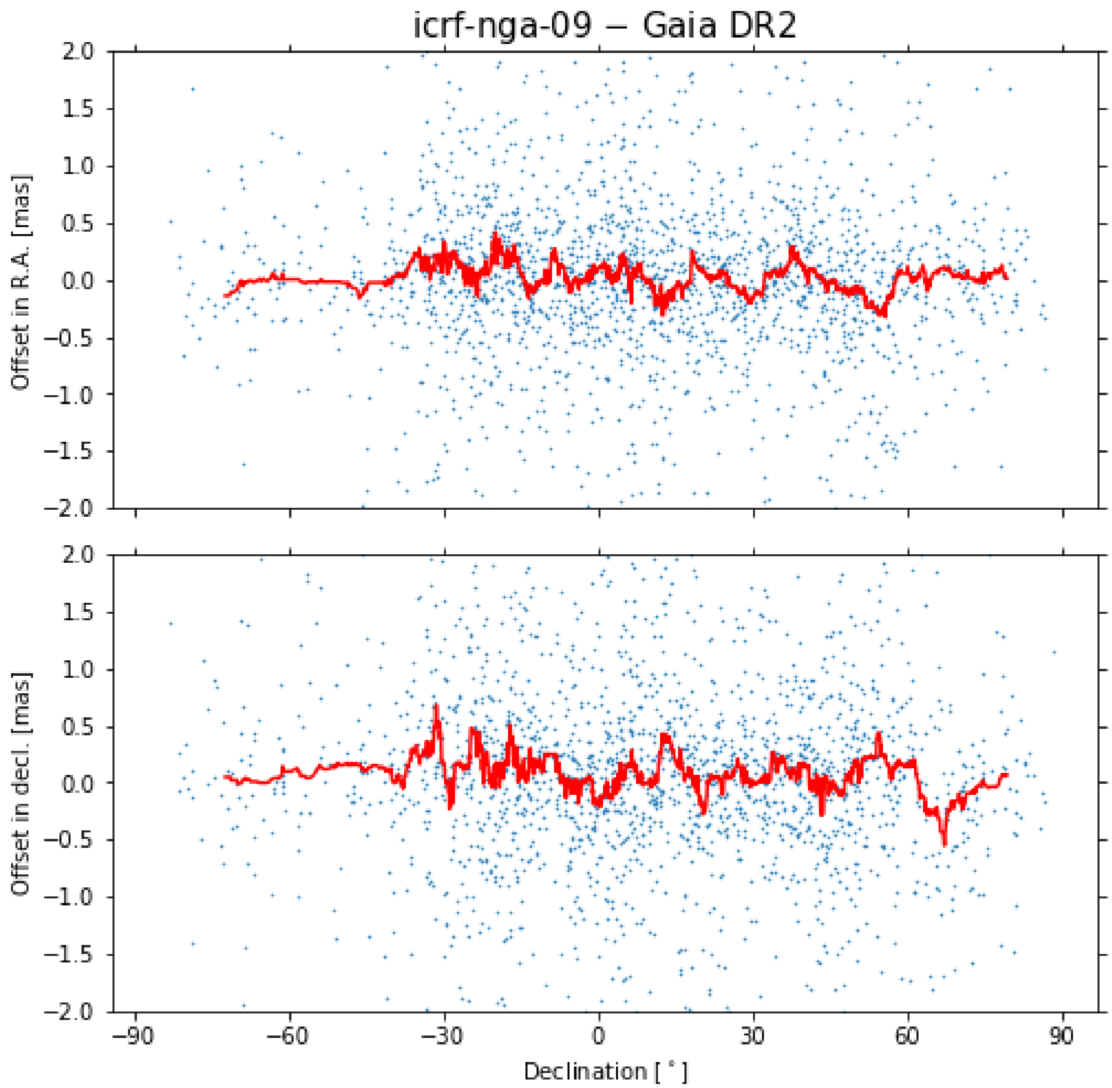
Their positional differences are on the order of 1 mas.

[15]:
```
fig, (ax0, ax1) = plt.subplots(figsize=(8, 8), nrows=2, sharex=True)
```

```
ax0.plot(gaiadif1["dec"], gaiadif1["dra"], ".", ms=1)
ax1.plot(gaiadif1["dec"], gaiadif1["ddec"], ".", ms=1)

ax0.set_ylabel("Offset in R.A. [mas]")

ax1.set_xlabel("Declination [$^\circ$]")
ax1.set_ylabel("Offset in decl. [mas]")

ax1.set_xticks(np.arange(-90, 91, 30))

# Running median
temp2 = Table(gaiadif1)
temp2.sort("dec")

window = 50
decmd2 = bn.move_median(temp2["dec"], window=window)
dramd2 = bn.move_median(temp2["dra"], window=window)
ddecmd2 = bn.move_median(temp2["ddec"], window=window)

ax0.plot(decmd2, dramd2, "r")
ax1.plot(decmd2, ddecmd2, "r")

ax0.set_ylim([-2, 2])
ax1.set_ylim([-2, 2])

ax0.xaxis.set_ticks_position("both")
ax0.yaxis.set_ticks_position("both")
ax1.xaxis.set_ticks_position("both")
ax1.yaxis.set_ticks_position("both")

ax0.set_title("icrf-nga-09 $-$ Gaia DR2", fontsize=15)

plt.subplots_adjust(hspace=0.1)
```

```
/usr/local/miniconda3/lib/python3.7/site-packages/bottleneck/slow/move.py:149:
FutureWarning: Using a non-tuple sequence for multidimensional indexing is
deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will
be interpreted as an array index, `arr[np.array(seq)]`, which will result either
in an error or a different result.
  nidx1 = n[idx1]
/usr/local/miniconda3/lib/python3.7/site-packages/bottleneck/slow/move.py:150:
FutureWarning: Using a non-tuple sequence for multidimensional indexing is
deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will
be interpreted as an array index, `arr[np.array(seq)]`, which will result either
in an error or a different result.
  nidx1 = nidx1 - n[idx2]
/usr/local/miniconda3/lib/python3.7/site-packages/bottleneck/slow/move.py:152:
```

```
FutureWarning: Using a non-tuple sequence for multidimensional indexing is
deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will
be interpreted as an array index, `arr[np.array(seq)]`, which will result either
in an error or a different result.
  idx[idx1] = nidx1 < min_count
/usr/local/miniconda3/lib/python3.7/site-packages/bottleneck/slow/move.py:153:
FutureWarning: Using a non-tuple sequence for multidimensional indexing is
deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will
be interpreted as an array index, `arr[np.array(seq)]`, which will result either
in an error or a different result.
  idx[idx3] = n[idx3] < min_count
```



Then the postional offsets are projected on the expensions of the vertor spherical harmonics.

[16]:
```
# icrf2-nga-09 - gaia dr2
# Transform columns into np.array
dra1 = np.array(gaiadif1["dra"])
ddec1 = np.array(gaiadif1["ddec"])
dra_err1 = np.array(gaiadif1["dra_err"])
ddec_err1 = np.array(gaiadif1["ddec_err"])
ra_rad1 = np.array(gaiadif1["ra"].to(u.radian))
dec_rad1 = np.array(gaiadif1["dec"].to(u.radian))
dra_ddec_cov1 = np.array(gaiadif1["dra_ddec_cov"])

# Transformation parameters
# l_max = 2
w21, sig21, corrcoef21, _, _, _ = vsh_deg02_fitting(
    dra1, ddec1, ra_rad1, dec_rad1, dra_err1, ddec_err1,
    cov=dra_ddec_cov1, elim_flag="None")

# mas -> uas
w21 = w21 * 1.e3
sig21 = sig21 * 1.e3

# icrf2-ga-09 - gaia dr2
# Transform columns into np.array
dra2 = np.array(gaiadif2["dra"])
ddec2 = np.array(gaiadif2["ddec"])
dra_err2 = np.array(gaiadif2["dra_err"])
ddec_err2 = np.array(gaiadif2["ddec_err"])
ra_rad2 = np.array(gaiadif2["ra"].to(u.radian))
dec_rad2 = np.array(gaiadif2["dec"].to(u.radian))
dra_ddec_cov2 = np.array(gaiadif2["dra_ddec_cov"])

# Transformation parameters
# l_max = 2
w22, sig22, corrcoef22, _, _, _ = vsh_deg02_fitting(
    dra2, ddec2, ra_rad2, dec_rad2, dra_err2, ddec_err2,
    cov=dra_ddec_cov2, elim_flag="None")

# mas -> uas
w22 = w22 * 1.e3
sig22 = sig22 * 1.e3
```

Use a histogram to show these terms clearly.

[17]:
```
from my_progs.catalog.vec_mod import vec_mod_calc

# icrf2-nga-09 - gaia dr2
gli1 = w21[:3]
rot1 = w21[3:6]
```

```
qua1 = w21[6:]

gerr1 = sig21[:3]
rerr1 = sig21[3:6]
qerr1 = sig21[6:]

glimod1, glierr1 = vec_mod_calc(gli1, gerr1)
rotmod1, roterr1 = vec_mod_calc(rot1, rerr1)

# icrf2-ga-09 - gaia dr2
gli2 = w22[:3]
rot2 = w22[3:6]
qua2 = w22[6:]

gerr2 = sig22[:3]
rerr2 = sig22[3:6]
qerr2 = sig22[6:]

glimod2, glierr2 = vec_mod_calc(gli2, gerr2)
rotmod2, roterr2 = vec_mod_calc(rot2, rerr2)
```

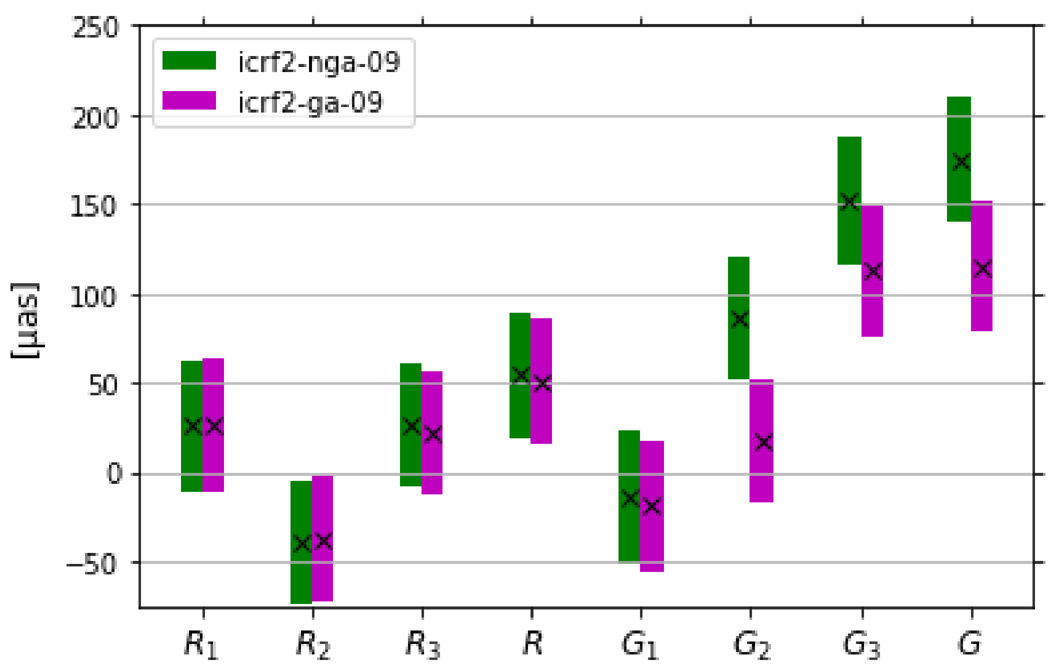
3.1 2.1) Rotation and Glide

[18]:
```
from my_progs.catalog.vec_mod import vec_mod_calc
```

[19]:
```
# Rotation
fig, ax = plt.subplots()

barwidth = 0.2
loc = 0.1

terms = ["$R_1$", "$R_2$", "$R_3$", "$R$",
         "$G_1$", "$G_2$", "$G_3$", "$G$"]

pos1 = np.arange(len(terms)) - 1 * loc
pos2 = np.arange(len(terms)) + 1 * loc

par1 = np.concatenate((rot1, [rotmod1], gli1, [glimod1]))
err1 = np.concatenate((rerr1, [roterr1], gerr1, [glierr1]))

par2 = np.concatenate((rot2, [rotmod2], gli2, [glimod2]))
err2 = np.concatenate((rerr2, [roterr2], gerr2, [glierr2]))


ax.bar(pos1, 2 * err1, bottom=par1-err1, width=barwidth,
       color="g", ecolor="black", label="icrf2-nga-09")
```

```
ax.bar(pos2, 2 * err2, bottom=par2-err2, width=barwidth,
       color="m", ecolor="black", label="icrf2-ga-09")

ax.plot(pos1, par1, "kx")
ax.plot(pos2, par2, "kx")

ax.set_xticks(range(len(terms)))
ax.set_xticklabels(terms, fontsize=12)
ax.set_ylabel("[$\\mathrm{\mu as}$]", fontsize=12)
ax.xaxis.set_ticks_position("both")
ax.yaxis.set_ticks_position("both")
ax.yaxis.grid()  # horizontal lines

ax.set_ylim(-75, 250)

ax.legend()
```

[19]:
```
<matplotlib.legend.Legend at 0x113cd0470>
```

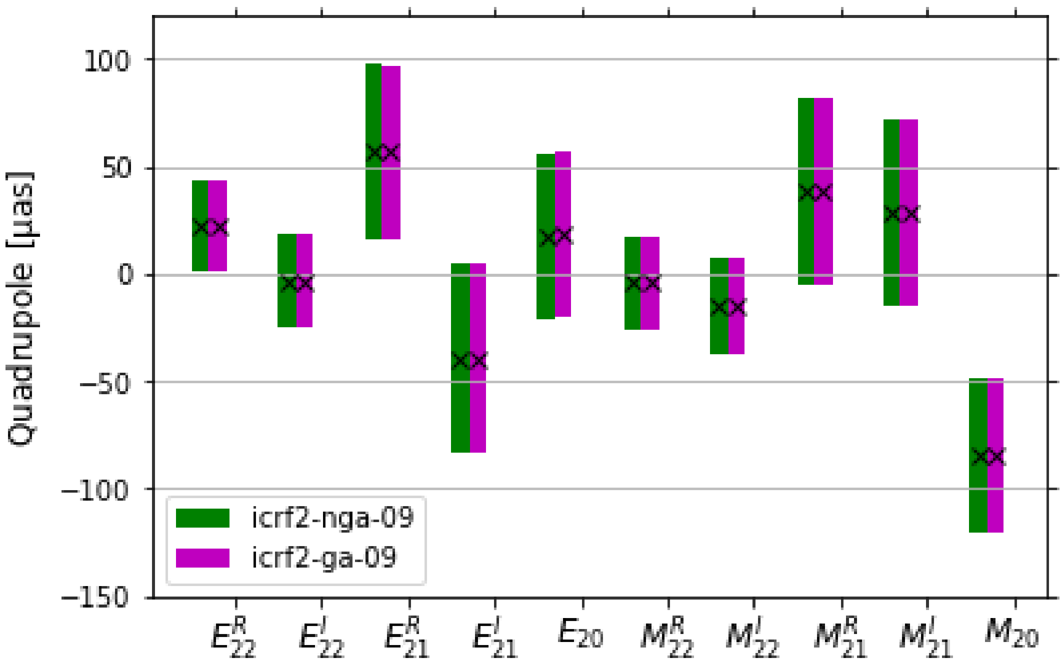


3.2 2.2) Quadruple terms

[20]:
```
# Quadruple terms

fig, ax = plt.subplots()
```

```python
barwidth = 0.2
loc = 0.2

terms = ["$E_{22}^R$", "$E_{22}^I$", "$E_{21}^R$", "$E_{21}^I$", "$E_{20}$",
         "$M_{22}^R$", "$M_{22}^I$", "$M_{21}^R$", "$M_{21}^I$", "$M_{20}$", ]

pos1 = np.arange(len(terms)) - 2 * loc
pos2 = np.arange(len(terms)) - 1 * loc

ax.bar(pos1, 2 * qerr1, bottom=qua1-qerr1, width=barwidth,
       color="g", ecolor="black", label="icrf2-nga-09")
ax.bar(pos2, 2 * qerr2, bottom=qua2-qerr2, width=barwidth,
       color="m", ecolor="black", label="icrf2-ga-09")

ax.plot(pos1, qua1, "kx")
ax.plot(pos2, qua2, "kx")

ax.set_xticks(range(len(terms)))
ax.set_xticklabels(terms, fontsize=12)

ax.set_ylabel("Quadrupole [$\\mathrm{\mu as}$]", fontsize=12)
ax.xaxis.set_ticks_position("both")
ax.yaxis.set_ticks_position("both")
ax.yaxis.grid()  # horizontal lines

ax.set_ylim(-150, 120)

ax.legend()
```

```
[20]: <matplotlib.legend.Legend at 0x114cc3710>
```



We can find

- 1) Modeling the GA effect can reduce the glide in Y- and Z-component, bringing the VLBI positions close to the *Gaia* position.
- 2) The glide in Z-component is still significant, even though the Galactic aberration effect is considered. Possibly it reflects the intrinsic errors in the VLBI astrometry.