ESM 232 Assignment 8

Madeline Oliver, Jennifer Truong, Alex Milward

5/19/2021

```r
sager = read.table("sager.txt", header=T)
head(sager)
```

```
##       model       obs month day year   wy wyd
## 1 0.4238063 0.3358678    10   1 1965 1966   1
## 2 0.4133587 0.3208737    10   2 1965 1966   2
## 3 0.4032640 0.3058796    10   3 1965 1966   3
## 4 0.3935287 0.2968832    10   4 1965 1966   4
## 5 0.3841480 0.2968832    10   5 1965 1966   5
## 6 0.3751000 0.2968832    10   6 1965 1966   6
```

```r
# add date from the existing columns of day, month, year
sager = sager %>% mutate(date=make_date(year=year, month=month, day=day))

# always start with plotting observed and model
# here's where you can catch "unrealistic" values
# plot
sagerl = sager %>% gather(key="source",value="streamflow",-date,-month,-day,-wy,-wyd,-year)

# apply some functions to measure performance

source("nse.R")
source("relerr.R")
source("cper.R")
nse(m=sager$model, o=sager$obs)
```

```
## [1] 0.6253416
```

```r
relerr(m=sager$model, o=sager$obs)*100
```

```
## [1] -18.9577
```

```r
cper(m=sager$model, o=sager$obs, weight.nse=0.8)
```

```
## [1] 0.5002733
```

```r
# try a different time step
# often evaluation changes when you change time scale
# choose the time scale most meaningful to your project
sager_wy = sager %>% group_by(wy) %>% summarize(model=sum(model), obs=sum(obs))

nse(sager_wy$model, sager_wy$obs)
```

```
## [1] 0.7702007
```

```r
cper(m=sager_wy$model, o=sager_wy$obs, weight.nse=0.8)
```

```
## [1] 0.6161606
```

```r
# just look at august flow
# imagine we are concerned about low flows causing high stream temperatures
# august might be the most important month
# first sum streamflow by month for each year
tmp = sager %>% group_by(month, year) %>% summarize(model=sum(model), obs=sum(obs))
```

```
## `summarise()` has grouped output by 'month'. You can override using the `.groups` argument.
```

```r
# now extract august
sager_aug = subset(tmp, month==8)
cor(sager_aug$model, sager_aug$obs)
```

```
## [1] 0.8248351
```

```r
# turn your evaluation metric into a function
# here's one for correlating annual minimum flow
source("check_minannual.R")
check_minannual(m=sager$model,o=sager$obs, month=sager$month, day=sager$day, year=sager$year, wy=sager$w
```

```
## [1] 0.8101656
```

Peformance evaluation may depend on what parameter set you use

Calibration is picking parameter sets based on performance evaluation

Apply metrics over multiple outputs (generated by running across many parameters sets) - like we've done in our sensitivity analysis work

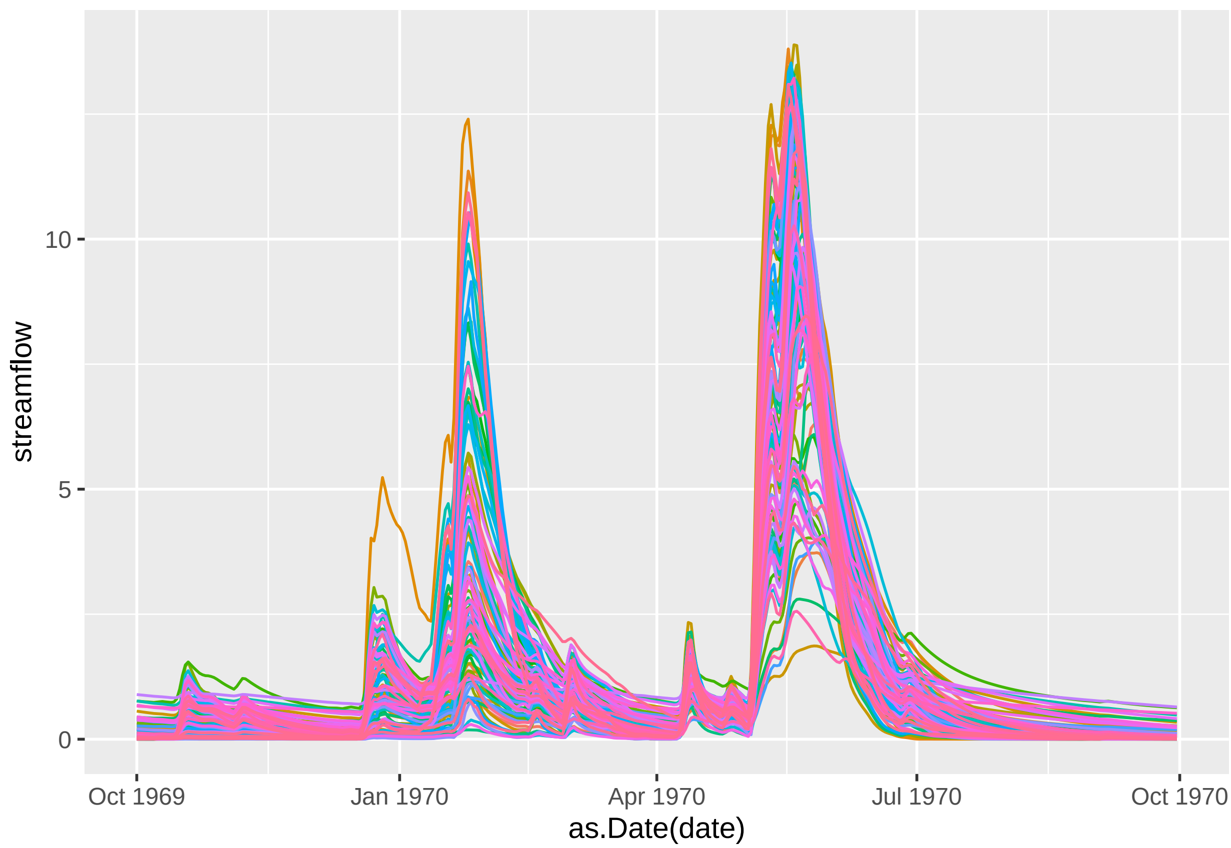
```r
# multiple results - lets say we've run the model for multiple years, each column
# is streamflow for a different parameter set
msage = read.table("sagerm.txt", header=T)

# lets say we know the start date from our earlier output
msage$date = sager$date
head(msage)
```

```
##        V99.1    V100.1       V101      V102       V103      V104      V105
## 1 0.07191767 0.3316747 0.04331200 0.1875757 0.07469700 0.2454343 0.1347037
## 2 0.06689267 0.3179167 0.04020500 0.1819137 0.06790767 0.2412470 0.1286780
## 3 0.06221900 0.3047440 0.03732067 0.1764227 0.06173567 0.2371983 0.1229220
## 4 0.05787167 0.2921237 0.03464333 0.1710973 0.05612433 0.2332663 0.1174237
## 5 0.05382833 0.2800427 0.03215800 0.1659330 0.05102333 0.2294617 0.1121710
## 6 0.05006733 0.2684613 0.02985100 0.1609243 0.04638600 0.2257630 0.1071530
##           V106      V107        V108      V109      V110       V111        V112
## 1 0.0003533333 0.2383413 0.003331333 0.2431933 0.3644930 0.05328633 0.005250000
## 2 0.0003400000 0.2321840 0.003039333 0.2355610 0.3583200 0.05014967 0.004755333
## 3 0.0003273333 0.2261857 0.002773000 0.2281683 0.3522187 0.04719767 0.004307333
## 4 0.0003150000 0.2203423 0.002530000 0.2210077 0.3463190 0.04441933 0.003901333
## 5 0.0003033333 0.2146500 0.002308333 0.2140717 0.3404873 0.04180433 0.003533667
## 6 0.0002920000 0.2091047 0.002106333 0.2073533 0.3347960 0.03934333 0.003200667
##        V113        V114      V115       V116       V117      V118      V119
## 1 0.5948570 0.012760333 0.2362903 0.01888033 0.12594367 0.4374097 0.2176843
## 2 0.5860857 0.011643667 0.2341553 0.01800533 0.11671333 0.4312180 0.2053780
## 3 0.5774453 0.010624667 0.2320393 0.01717100 0.10815933 0.4251140 0.1937673
```

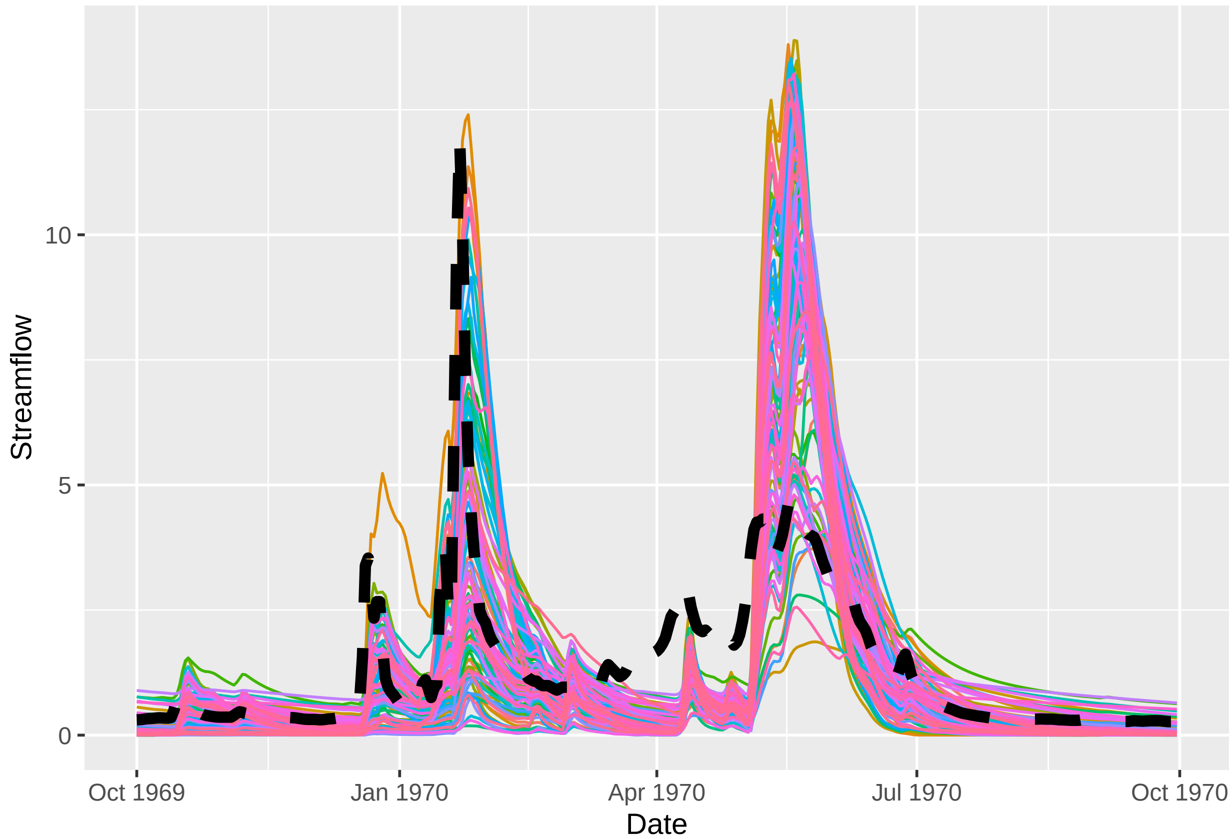
```
## 4 0.5689357 0.009695000 0.2299423 0.01637500 0.10023233 0.4190963 0.1828130
## 5 0.5605520 0.008846667 0.2278643 0.01561600 0.09288633 0.4131640 0.1724780
## 6 0.5522937 0.008072333 0.2258053 0.01489200 0.08607867 0.4073157 0.1627270
##         V120       V121      V122       V123       V124       V125       V126
## 1 0.03378267 0.06285833 0.1675450 0.01840800 0.07664567 0.08750367 0.06550033
## 2 0.03198167 0.05886167 0.1607863 0.01818167 0.07178267 0.07925833 0.06094633
## 3 0.03027667 0.05511900 0.1543007 0.01795833 0.06722800 0.07178967 0.05670900
## 4 0.02866267 0.05161433 0.1480763 0.01773767 0.06296233 0.06502500 0.05276633
## 5 0.02713500 0.04833233 0.1421033 0.01752000 0.05896733 0.05889767 0.04909767
## 6 0.02568833 0.04525933 0.1363710 0.01730467 0.05522600 0.05334767 0.04568400
##        V127      V128      V129      V130      V131      V132       V133
## 1 0.4238063 0.1451923 0.2529733 0.5392687 0.2826070 0.3202217 0.09478400
## 2 0.4133587 0.1420453 0.2425717 0.5297423 0.2725720 0.3132013 0.08795600
## 3 0.4032640 0.1389667 0.2325977 0.5207750 0.2628933 0.3063350 0.08161967
## 4 0.3935287 0.1359547 0.2230337 0.5123903 0.2535583 0.2996190 0.07573967
## 5 0.3841480 0.1330080 0.2138630 0.5044643 0.2445547 0.2930503 0.07028333
## 6 0.3751000 0.1301250 0.2050693 0.4969153 0.2358707 0.2866257 0.06522000
##         V134       V135       V136       V137     V138      V139        V140
## 1 0.06635500 0.11842967 0.06669433 0.04664267 0.300477 0.2028417 0.012289333
## 2 0.06367833 0.11037967 0.06533933 0.04223633 0.294672 0.1982920 0.011173667
## 3 0.06110933 0.10287700 0.06401167 0.03824633 0.289076 0.1938443 0.010159667
## 4 0.05864433 0.09588400 0.06271100 0.03463333 0.283719 0.1894963 0.009237667
## 5 0.05627867 0.08936667 0.06143667 0.03136167 0.278557 0.1852460 0.008399333
## 6 0.05400833 0.08329200 0.06018833 0.02839900 0.273602 0.1810907 0.007637000
##         V141       V142      V143      V144      V145      V146      V147
## 1 0.06128400 0.02764267 0.1804390 0.2829493 0.1520090 0.2241143 0.7156417
## 2 0.06053600 0.02508200 0.1691530 0.2743833 0.1437337 0.2130743 0.7082513
## 3 0.05979700 0.02275867 0.1585730 0.2660767 0.1359090 0.2025780 0.7009373
## 4 0.05906700 0.02065067 0.1486547 0.2580213 0.1285100 0.1925987 0.6936990
## 5 0.05834567 0.01873767 0.1393567 0.2502097 0.1215140 0.1831110 0.6865357
## 6 0.05763333 0.01700167 0.1306403 0.2426347 0.1148987 0.1740907 0.6794463
##        V148      V149       V150       V151       V152       V153        V154
## 1 0.2459190 0.2593303 0.04046233 0.10185033 0.06195833 0.10997067 0.009269667
## 2 0.2405390 0.2468773 0.03690200 0.09695700 0.05648833 0.10079000 0.008794000
## 3 0.2352767 0.2350223 0.03365500 0.09229867 0.05150133 0.09237700 0.008343000
## 4 0.2301293 0.2237367 0.03069367 0.08786433 0.04695433 0.08466767 0.007915000
## 5 0.2250950 0.2129927 0.02799267 0.08364300 0.04280867 0.07760267 0.007509000
## 6 0.2201707 0.2027647 0.02552933 0.07962467 0.03902933 0.07112800 0.007123667
##         V155       V156      V157       V158      V159       V160       V161
## 1 0.08622433 0.10054867 0.2285157 0.08376633 0.5664663 0.10368200 0.06505233
## 2 0.07895133 0.09925867 0.2167053 0.07812267 0.5552560 0.09547367 0.06421500
## 3 0.07229167 0.09798533 0.2055057 0.07285900 0.5442673 0.08791533 0.06338833
## 4 0.06619400 0.09672833 0.1948847 0.06795000 0.5334960 0.08095500 0.06257267
## 5 0.06061067 0.09548733 0.1848123 0.06337167 0.5229380 0.07454600 0.06176733
## 6 0.05549833 0.09426233 0.1752607 0.05910200 0.5125890 0.06864433 0.06097233
##         V162      V163       V164      V165      V166       V167       V168
## 1 0.03208967 0.1484727 0.02082133 0.1788070 0.2103860 0.05299600 0.08575100
## 2 0.02934900 0.1428527 0.01943867 0.1768543 0.2058670 0.05246267 0.08295733
## 3 0.02684233 0.1374453 0.01814767 0.1749230 0.2015403 0.05194533 0.08025467
## 4 0.02454967 0.1322427 0.01694233 0.1730127 0.1974167 0.05144067 0.07764000
## 5 0.02245300 0.1272373 0.01581733 0.1711233 0.1934500 0.05095233 0.07511067
## 6 0.02053500 0.1224213 0.01476667 0.1692543 0.1896473 0.05047133 0.07266367
##         V169         V170      V171       V172      V173      V174      V175
## 1 0.08208500 0.0007126667 0.3321513 0.08189933 0.3378253 0.1432480 0.7430853
```

```
## 2 0.07795867 0.0006753333 0.3250353 0.07565067 0.3255447 0.1332823 0.7382633
## 3 0.07404000 0.0006400000 0.3180720 0.06987867 0.3137103 0.1240100 0.7334727
## 4 0.07031800 0.0006063333 0.3112577 0.06454733 0.3023063 0.1153827 0.7287130
## 5 0.06678333 0.0005746667 0.3045897 0.05962267 0.2913170 0.1073557 0.7239843
## 6 0.06342633 0.0005446667 0.2980643 0.05507367 0.2807270 0.0998870 0.7192863
##        V176      V177       V178      V179      V180       V181      V182
## 1 0.1609307 0.1326143 0.08507667 0.5321190 0.6998950 0.06295467 0.4064717
## 2 0.1496117 0.1302127 0.07844300 0.5224367 0.6909930 0.05740367 0.4009937
## 3 0.1390887 0.1278543 0.07232633 0.5129303 0.6822040 0.05234233 0.3955893
## 4 0.1293057 0.1255390 0.06668667 0.5035970 0.6735270 0.04772733 0.3902580
## 5 0.1202110 0.1232657 0.06148667 0.4944333 0.6649603 0.04351900 0.3849987
## 6 0.1117560 0.1210333 0.05669233 0.4854367 0.6565027 0.03968167 0.3798100
##        V183        V184      V185       V186      V187      V188      V189
## 1 0.1612057 0.011333000 0.5693913 0.10873833 0.3803070 0.5337300 0.1945403
## 2 0.1501753 0.010880000 0.5595980 0.10389400 0.3671423 0.5310793 0.1823263
## 3 0.1398997 0.010444667 0.5499730 0.09926567 0.3544333 0.5284417 0.1708793
## 4 0.1303273 0.010027000 0.5405137 0.09484367 0.3421643 0.5258170 0.1601510
## 5 0.1214097 0.009626000 0.5312170 0.09061867 0.3303200 0.5232057 0.1500963
## 6 0.1131023 0.009241333 0.5220803 0.08658167 0.3188857 0.5206070 0.1406727
##         V190      V191      V192      V193       V194      V195      V196
## 1 0.02710667 0.1718877 0.2836493 0.1334437 0.07881167 0.2935460 0.2200570
## 2 0.02649667 0.1624967 0.2761773 0.1266033 0.07252633 0.2823550 0.2093427
## 3 0.02590033 0.1536187 0.2689023 0.1201153 0.06674233 0.2715907 0.1991500
## 4 0.02531767 0.1452257 0.2618187 0.1139563 0.06141933 0.2612367 0.1894533
## 5 0.02474800 0.1372913 0.2549220 0.1081093 0.05652100 0.2512777 0.1802290
## 6 0.02419133 0.1297903 0.2482067 0.1025590 0.05201333 0.2416983 0.1714537
##          V197       V198       V199       date
## 1 0.011247667 0.07537933 0.04625600 1965-10-01
## 2 0.010750333 0.07278433 0.04515367 1965-10-02
## 3 0.010282667 0.07027900 0.04407767 1965-10-03
## 4 0.009823000 0.06785967 0.04302733 1965-10-04
## 5 0.009406333 0.06552400 0.04200200 1965-10-05
## 6 0.008985333 0.06326867 0.04100100 1965-10-06
```

```r
msage$month = sager$month
msage$year = sager$year
msage$day = sager$day
msage$wy = sager$wy

# and we still have observed data from above
# useful to combine by date to make sure that streamflow and observe match

msage$obs = sager$obs


# how can we plot all results
# to turn all the columns of different outputs into a single column identified by "run"
msagel = msage %>% gather(key="run",value="streamflow", -date, -month, -day, -year, -wy, -obs)

#lets plot water year 1970 otherwise its hard to see
p1=ggplot(subset(msagel, wy == 1970), aes(as.Date(date), streamflow, col=run))+geom_line()+theme(legend
p1
```




```
# lets add observed streamflow
p1+geom_line(aes(as.Date(date), obs), size=2, col="black", linetype=2)+labs(y="Streamflow", x="Date")
```




```
# compute performance measures for all output
res = msage %>% select(-date, -month, -day, -year, -wy ) %>% map_dbl(~nse(m=.x, o=msage$obs))
summary(res)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -1.40236 -0.16063  0.12515  0.04981  0.33553  1.00000
```

```
# one of them has a "perfect score" why?
# redo
res = msage %>% select(-date, -month, -day, -year, -wy, -obs) %>% map_dbl(~nse(m=.x, o=msage$obs))
summary(res)
```

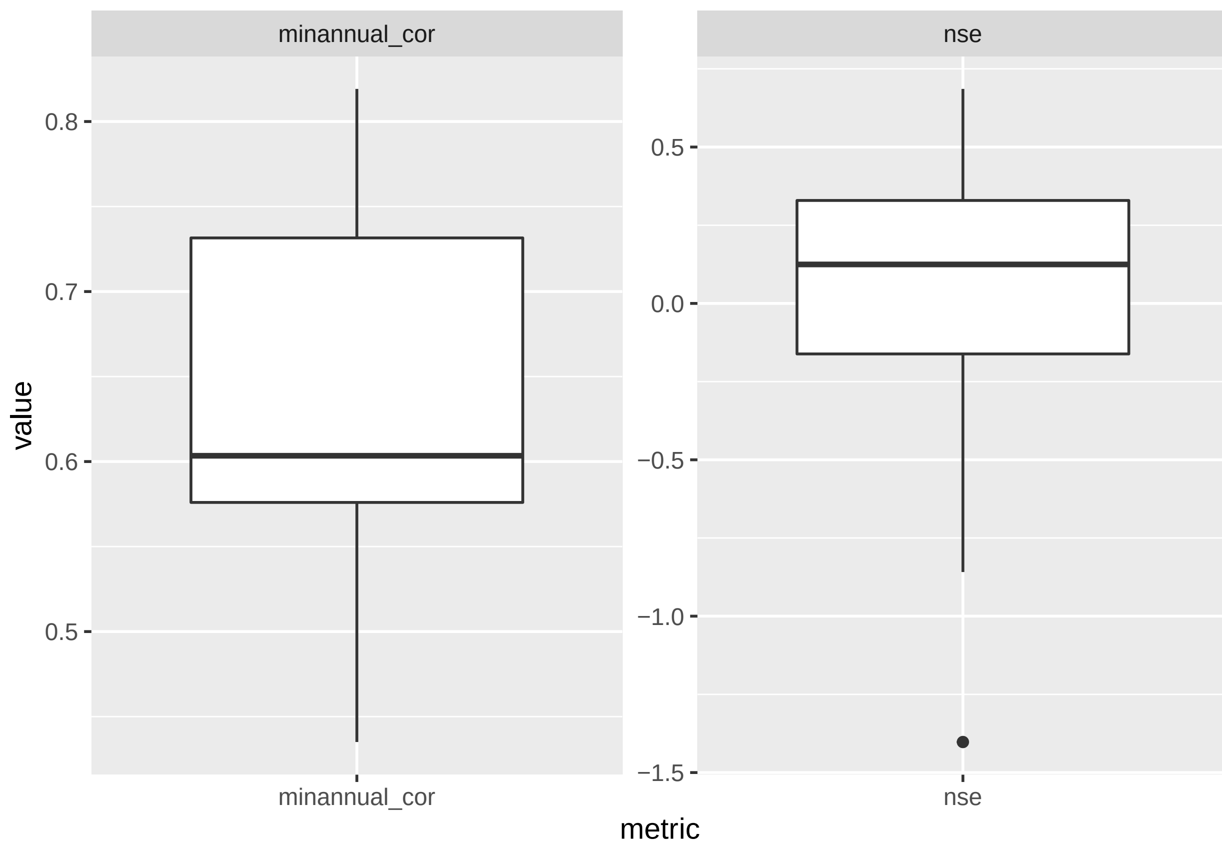
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.4024 -0.1614  0.1247  0.0404  0.3293  0.6859
```

```
# if we want to keep track of which statistics is associated with each run, we need a unique identifies
# a ID that tracks each model output - lets use the column names
simnames = names(msage %>% select(-date, -month, -day,-year,-wy, -obs))
results = cbind.data.frame(simnames=simnames, nse=res)

# another example using our low flow statistics
# use apply to compute for all the data
res = msage %>% select(-date, -month, -day, -year, -wy, -obs ) %>% map_dbl(~check_minannual( o=msage$obs


# add to our results
results$minannual_cor = res
```

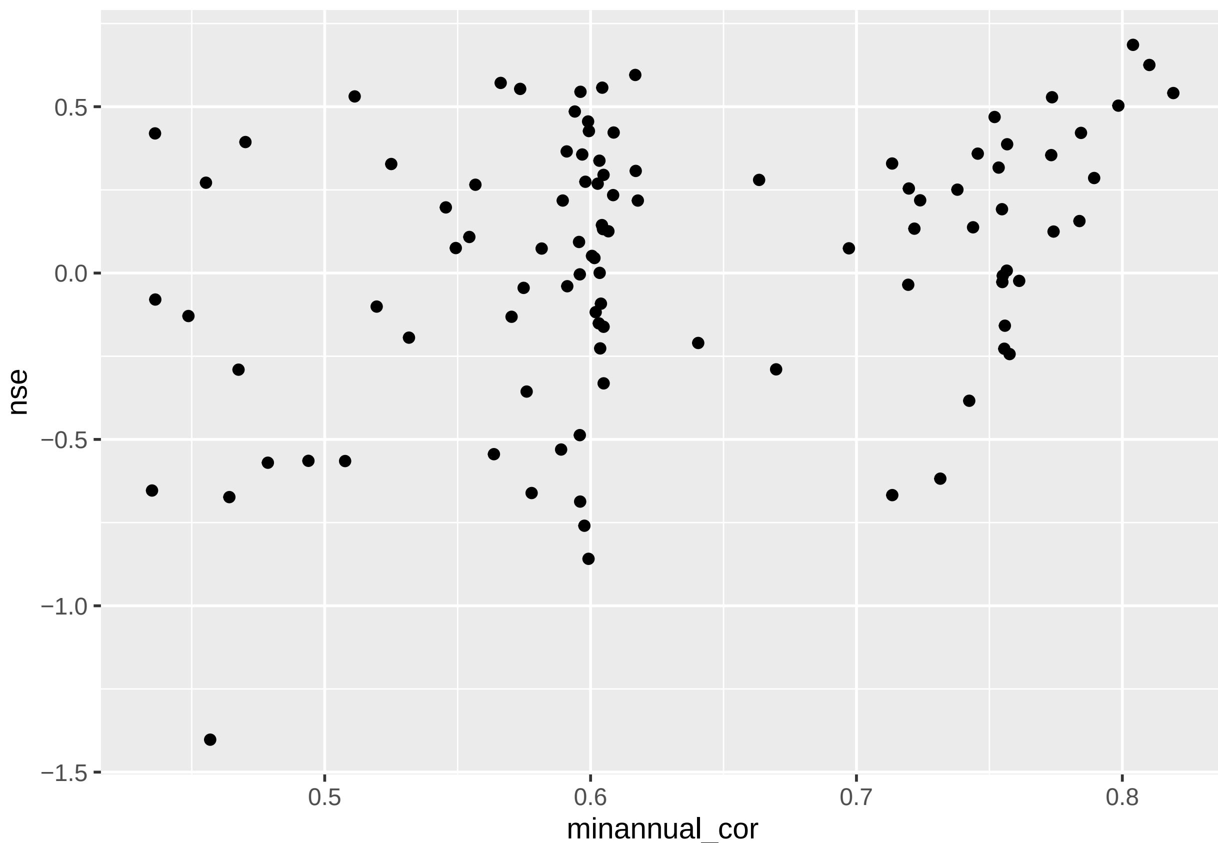
```r
# interesting to look at range of metrics - could use this to decide on
# acceptable values
summary(results)
```

```
##    simnames              nse           minannual_cor
##  Length:101         Min.   :-1.4024   Min.   :0.4350
##  Class :character   1st Qu.:-0.1614   1st Qu.:0.5760
##  Mode  :character   Median : 0.1247   Median :0.6034
##                     Mean   : 0.0404   Mean   :0.6296
##                     3rd Qu.: 0.3293   3rd Qu.:0.7315
##                     Max.   : 0.6859   Max.   :0.8192
```

```r
# graph range of performance measures
resultsl = results %>% gather(key="metric",value="value", -simnames)
ggplot(resultsl, aes(metric, value))+geom_boxplot()+facet_wrap(~metric, scales="free")
```




```r
# are metrics related to each other
# useful for assessing whether there are tradeoffs
ggplot(results, aes(minannual_cor, nse))+geom_point()
```


0.5
0.0
−0.5
−1.0
−1.5
nse
0.5
0.6
0.7
0.8
minannual_cor