

Presentation of new R functions for forecast verification

Stefan Siegert, (s.siegert@exeter.ac.uk)

please go to <https://github.com/sieste/ic3-workshop> and download all files into a new directory

This workshop

- ▶ **<https://github.com/sieste/ic3-workshop>**
- ▶ hands-on session
- ▶ presentation of some new R functions
- ▶ interpretation of the output

Overview of available verification packages in R

- ▶ `verification`
 - ▶ developed at NCAR
 - ▶ methods: Brier, CRPS, ROC, reliability diagram, rank histogram
- ▶ `s2dverification`
 - ▶ currently developed at IC3
 - ▶ methods: ACC, RMSSS, plotting!

New contributions

- ▶ ensemble verification
- ▶ uncertainty estimates
- ▶ comparative verification
- ▶ ... **work in progress**

```
source("R/toydata.r")

## Loading required package: boot

source("R/rankhist.r")
source("R/rel-diag.r")
source("R/ensemble-scores.r")
```

Gaussian toy data

- ▶ Gaussian ensemble data with mean `mu.ens` and stdev `sd.ens`
- ▶ Gaussian verification with mean `mu.ver` and stdev `sd.ver`
- ▶ number of samples `N`
- ▶ number of ensemble members `K`

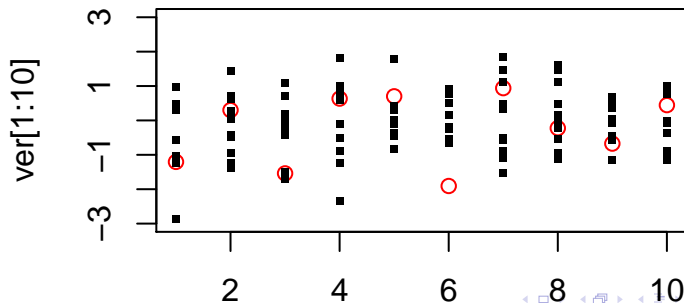
```
K <- 10
toydata <- GenerateToyData(mu.ens=0, sd.ens=1,
                           mu.ver=0, sd.ver=1,
                           K=K, N=100)

objects(toydata)

## [1] "ens" "ver"
```

Gaussian toy data

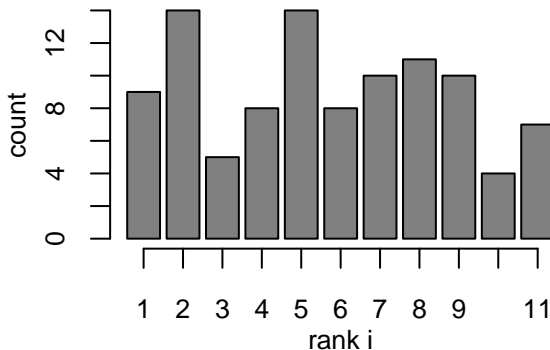
```
with(toydata, {  
  plot(1:10, ver[1:10], col="red", ylim=c(-3,3))  
  matplot(1:10, ens[1:10, ], pch=15, cex=.5,  
          col="black", add=TRUE)  
})
```



Rank histogram

- ▶ r_i : rank of the verification in the ordered ensemble
- ▶ e.g. ver=1.5, ens={1,2,3}: rank = 2
- ▶ histogram over the observed ranks

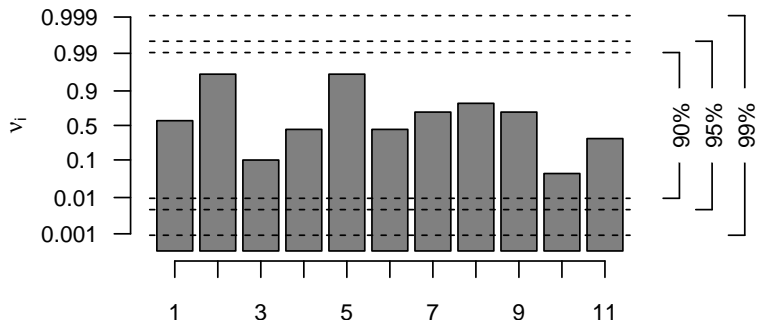
```
source("R/rankhist.r")  
rh <- with(toydata, rankhist(ens, ver))  
PlotRankhist(rh, mode="raw")
```



Rank histogram on probability paper

- ▶ H_0 : the individual rank counts are $\sim \text{Binomial}(N, \frac{1}{K+1})$
- ▶ plot the cumulative likelihood of the observed rank counts under H_0

```
PlotRankhist(rh, mode="prob.paper")
```



Rank histogram significance tests

- ▶ Pearson χ^2 -test
- ▶ Jolliffe-Primo χ^2 -decomposition

```
rh.tests <- rankhist.tests(rh)
print(rh.tests)
```

```
##                pearson.chi2  jp.slope  jp.convex
## test.statistic      11.3200    1.0890    0.7601
## p.value              0.3331    0.2967    0.3833
```

Fair Brier Score for binary ensemble forecasts

- ▶ j ... verification, 1 = yes, 0 = no
- ▶ i ... number of ensemble members that predict the event
- ▶ $Br(i,j) = (j - \frac{i}{K})^2 - \frac{i(K-i)}{K^2(K-1)}$

```
source("R/ensemble-scores.r")
tau <- 1 # exceedance threshold
with(toydata, mean(fairbrier(ens, ver, tau)))

## [1] 0.1024
```

Fair continuously ranked probability score for ensemble forecasts

- ▶ fair Brier Score integrated over all possible thresholds
- ▶ $crps(e, y) = \langle |y - e_i| \rangle - \frac{1}{2K(K-1)} \langle |e_i - e_j| \rangle$

```
source("R/ensemble-scores.r")  
fcrps <- with(toydata, faircrps(ens, ver))  
mean(fcrps)
```

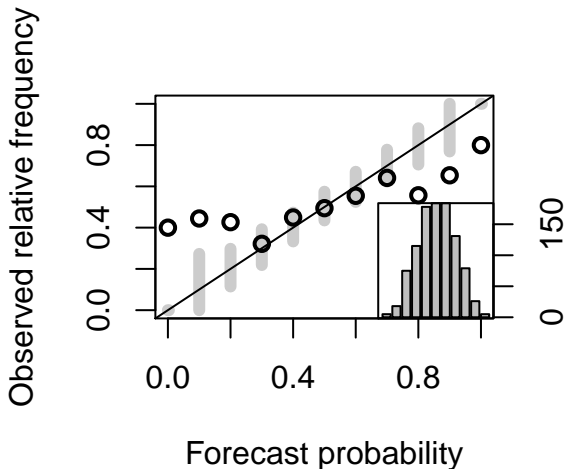
```
## [1] 0.5043
```

Reliability diagram

```
N <- 1000
mu <- runif(N)
toydata <- GenerateToyData(N=N, mu.ens=mu, mu.ver=mu)
tau <- .5
i <- with(toydata, rowSums(ens > tau))
j <- with(toydata, 1 * (ver > tau))
```

Reliability diagram

```
source("R/rel-diag.r")  
rd <- rel.diag(probs=i/K, ver=j, nbins=11, plot=TRUE)
```



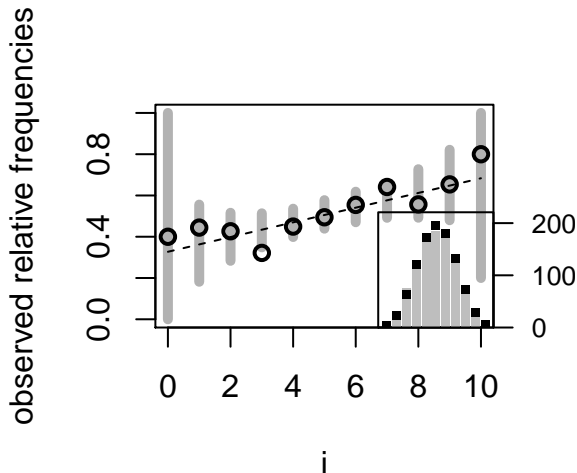
Reliability diagram

```
print(rd)
```

##	p.avg	cond.probs	cbar.lo	cbar.hi
## 1	0.0	0.4000	0.0000	0.0000
## 2	0.1	0.4444	0.0000	0.2727
## 3	0.2	0.4267	0.1147	0.2973
## 4	0.3	0.3217	0.2188	0.3922
## 5	0.4	0.4494	0.3332	0.4727
## 6	0.5	0.4946	0.4350	0.5741
## 7	0.6	0.5543	0.5247	0.6706
## 8	0.7	0.6412	0.6229	0.7778
## 9	0.8	0.5570	0.7059	0.8816
## 10	0.9	0.6538	0.7692	1.0000
## 11	1.0	0.8000	1.0000	1.0000

Fair reliability diagram

```
frd <- fair.rel.diag(i=i, j=j, K=K, plot=TRUE, plot.refin=TRUE)
```



Fair reliability diagram

```
print(frd)
```

##	i	cond.probs	H0.line	cbar.lo	cbar.hi
## 1	0	0.4000	0.3271	0.0000	1.0000
## 2	1	0.4444	0.3627	0.1818	0.5557
## 3	2	0.4267	0.3984	0.2833	0.5161
## 4	3	0.3217	0.4340	0.3486	0.5136
## 5	4	0.4494	0.4696	0.3986	0.5360
## 6	5	0.4946	0.5053	0.4391	0.5773
## 7	6	0.5543	0.5409	0.4702	0.6183
## 8	7	0.6412	0.5765	0.4917	0.6593
## 9	8	0.5570	0.6122	0.4921	0.7273
## 10	9	0.6538	0.6478	0.4800	0.8214
## 11	10	0.8000	0.6834	0.2000	1.0000

Comparative ensemble verification

- ▶ We want to address the question: Is the forecast `ens` better than a reference forecast `ens.ref` at predicting the same verification `ver`?
- ▶ Our hindcast dataset now has 3 members

Comparison of two imperfect ensemble forecasts

- ▶ ens and ver as before
- ▶ additionally: ens.ref, a benchmark ensemble, to which the performance of ens is compared

```
K <- 10
toydata2 <- GenerateToyData(mu.ver=0, sd.ver=1,
                             mu.ref=0.4, sd.ref=1,
                             mu.ens=0.15, sd.ens=1,
                             K=K, K.ref=K, N=100)

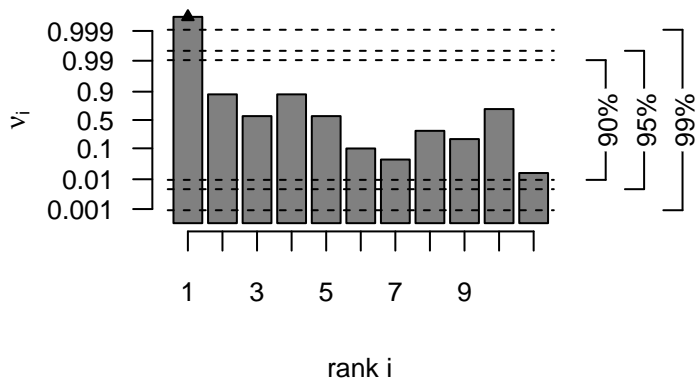
objects(toydata2)

## [1] "ens"      "ens.ref" "ver"
```

Good statistical tests should find that both ensembles are unreliable (biased), and that ens is more reliable than ens.ref.

Rank histogram analysis of ens.ref

```
rh.ref <- with(toydata2, rankhist(ens.ref, ver))  
PlotRankhist(rh.ref, mode="prob.paper")
```



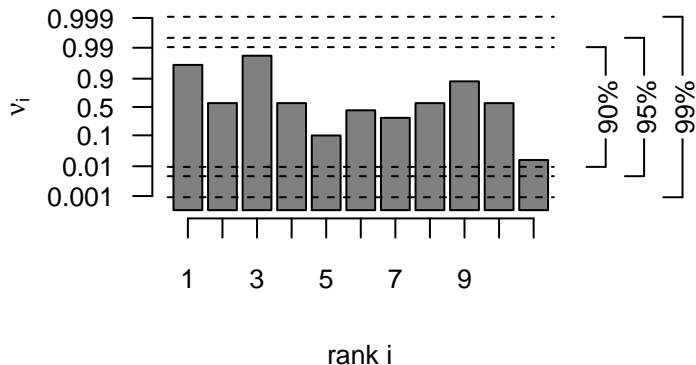
Rank histogram analysis of ens.ref

```
rh.tests.ref <- rankhist.tests(rh.ref)
print(rh.tests.ref)
```

```
##                pearson.chi2  jp.slope  jp.convex
## test.statistic    3.354e+01  1.742e+01    6.5482
## p.value           2.209e-04  2.990e-05    0.0105
```

Rank histogram analysis of ens

```
rh <- with(toydata2, rankhist(ens, ver))  
PlotRankhist(rh, mode="prob.paper")
```



Rank histogram analysis of ens

```
rh.tests <- rankhist.tests(rh)
print(rh.tests)
```

```
##                pearson.chi2  jp.slope  jp.convex
## test.statistic      13.9600   3.84400   0.2051
## p.value             0.1748   0.04992   0.6506
```

Comparison of rank histograms: AnalyzeRankhistDifference

```
rh.comp <- with(toydata2,  
               AnalyzeRankhistDifference(ens, ens.ref,  
                                         ver, n.boot=100))
```

rh.comp is a matrix that summarizes the rank histogram comparison:

```
print(t(as.matrix(rh.comp)))
```

##	pearson.chi2	jp.slope	jp.convex
## score.diff	19.580	13.580	6.343
## p.value	0.060	0.010	0.150
## Q0.01	-38.150	-1.894	-15.417
## Q0.05	-15.191	-1.055	-5.049
## Q0.1	-3.388	3.681	-1.023
## Q0.9	59.906	22.357	31.172
## Q0.95	81.686	24.675	39.391
## Q0.99	116.853	31.923	52.397

Similar for the specific scores for slope and convexity:

- ▶ p-values:

```
print(t(rh.comp[2:3, 1:2]))
```

```
##           jp.slope jp.convex
## score.diff    13.58    6.343
## p.value       0.01    0.150
```

- ▶ bootstrap quantiles:

```
print(rh.comp[2:3, 3:8])
```

```
##           Q0.01  Q0.05  Q0.1  Q0.9  Q0.95  Q0.99
## jp.slope  -1.894 -1.055  3.681 22.36 24.67 31.92
## jp.convex -15.417 -5.049 -1.023 31.17 39.39 52.40
```

Comparison of fair Brier Scores

```
K <- 10  
toydata3 <- GenerateToyData(mu.ver=0, sd.ver=1,  
                             mu.ens=0, sd.ens=1,  
                             mu.ref=0, sd.ref=2,  
                             K=K, K.ref=K, N=100)
```

Analysis of fair Brier score difference

```
fbr.comp <- with(toydata3,  
  AnalyzeFairBrierDifference(ens, ens.ref, ver,  
                             tau=.5, n.boot=100))  
print(as.matrix(fbr.comp))
```

```
##                [,1]  
## fair.brier.diff 0.01956  
## p.value         0.20000  
## Q0.01          -0.03849  
## Q0.05          -0.02274  
## Q0.1           -0.01193  
## Q0.9           0.05360  
## Q0.95          0.05712  
## Q0.99          0.06434
```

Comparison of fair crps

```
fcrps.comp <- with(toydata3,  
  AnalyzeFairCrpsDifference(ens=ens, ens.ref=ens.ref,  
                             ver=ver, n.boot=100))  
print(as.matrix(fcrps.comp))
```

```
##                [,1]  
## fair.crps.diff 0.12023  
## p.value        0.00000  
## Q0.01          0.04691  
## Q0.05          0.05940  
## Q0.1           0.06810  
## Q0.9           0.18077  
## Q0.95          0.19096  
## Q0.99          0.20236
```

Some actual data

- ▶ tropical sea surface temperature data
- ▶ 51 years
- ▶ 10 lead times (10 years)
- ▶ 8 different ensembles
- ▶ up to 10 members each

```
load("R/SST-raw.Rdata")  
print(dim(SST.trop))
```

```
## [1] 51 10 8 10
```

```
print(dim(SST.trop.obs))
```

```
## [1] 51 10
```

Some actual data

- ▶ the available models

```
matrix(dimnames(SST.trop)[[3]], ncol=2)
```

```
##      [,1]      [,2]
## [1,] "gfdl"    "hadcm3_ff"
## [2,] "cancm4"  "ec_earth_ff"
## [3,] "miroc5"  "ec_earth_an"
## [4,] "hadcm3_an" "bcc"
```

A specific analysis

- ▶ set parameters

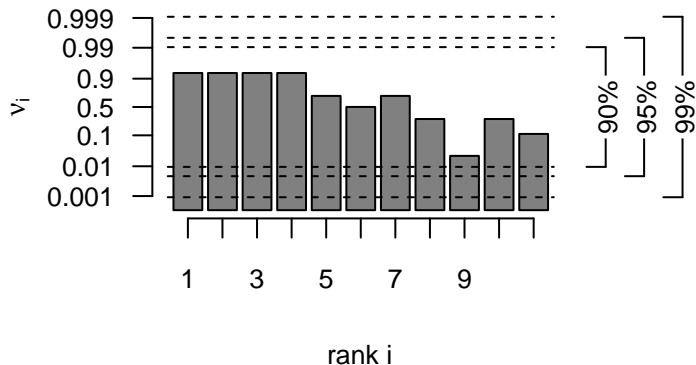
```
lead <- 7
model <- "hadcm3_ff"
model.ref <- "hadcm3_an"
dates <- 1:40
members <- 1:10
members.ref <- 1:10
```

- ▶ get data:

```
ens <- SST.trop[dates, lead, model, members]
ens.ref <- SST.trop[dates, lead, model.ref, members.ref]
ver <- SST.trop.obs[dates, lead]
```

Rank histogram of ens.ref

```
rh.ref <- rankhist(ens.ref,ver)  
PlotRankhist(rh.ref, mode="prob.paper")
```



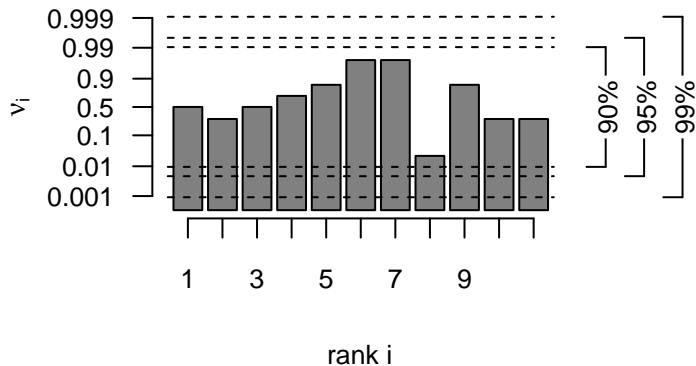
Rank histogram of ens.ref

```
rankhist.tests(rh.ref)
```

```
##                pearson.chi2  jp.slope  jp.convex  
## test.statistic      13.3500  1.122e+01  0.002885  
## p.value             0.2048  8.081e-04  0.957167
```

Rank histogram of ens

```
rh <- rankhist(ens,ver)
PlotRankhist(rh, mode="prob.paper")
```



Rank histogram of ens

```
rankhist.tests(rh)
```

```
##                pearson.chi2  jp.slope  jp.convex  
## test.statistic      13.3500    0.0625    3.9490  
## p.value             0.2048     0.8026    0.0469
```

Rank histogram comparison

```
rh.comp <- AnalyzeRankhistDifference(ens, ens.ref,  
                                     ver, n.boot=100)  
print(t(as.matrix(rh.comp)))
```

##	pearson.chi2	jp.slope	jp.convex
## score.diff	0.000	11.1600	-3.946
## p.value	0.490	0.0000	0.840
## Q0.01	-26.554	-0.6922	-13.215
## Q0.05	-18.727	0.8324	-6.823
## Q0.1	-6.655	3.6180	-4.015
## Q0.9	28.655	18.4220	4.903
## Q0.95	32.532	20.4600	7.525
## Q0.99	50.072	26.1248	14.706

Brier Score comparison

```
br.comp <- AnalyzeFairBrierDifference(ens, ens.ref, ver,  
                                     tau=mean(ver),  
                                     n.boot=100)  
  
print(as.matrix(br.comp))
```

```
##                [,1]  
## fair.brier.diff -3.383e-18  
## p.value         5.100e-01  
## Q0.01          -8.678e-02  
## Q0.05          -6.133e-02  
## Q0.1           -4.478e-02  
## Q0.9           4.006e-02  
## Q0.95          5.353e-02  
## Q0.99          6.488e-02
```

Crps comparison

```
crps.comp <- AnalyzeFairCrpsDifference(ens, ens.ref,  
                                       ver, n.boot=100)  
print(as.matrix(crps.comp))
```

```
##                [,1]  
## fair.crps.diff 0.013999  
## p.value        0.010000  
## Q0.01          0.002906  
## Q0.05          0.005512  
## Q0.1           0.006312  
## Q0.9           0.020968  
## Q0.95          0.023224  
## Q0.99          0.026772
```