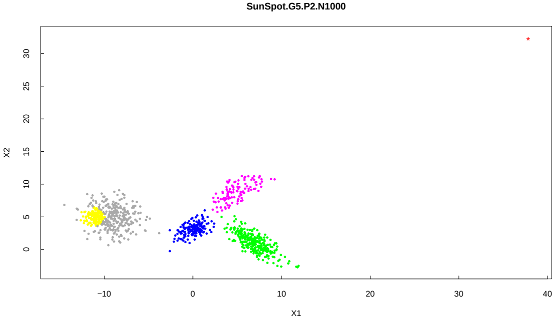
Comparating methods for robust elliptical clustering, including the robust improper ML estimator

Christian Hennig and Pietro Coretto

December 17, 2011

1 The challenge



Gaussian clusters and extreme uniform on P=1,2; standard Gaussian on P>2

Standard Gaussian model-based clustering

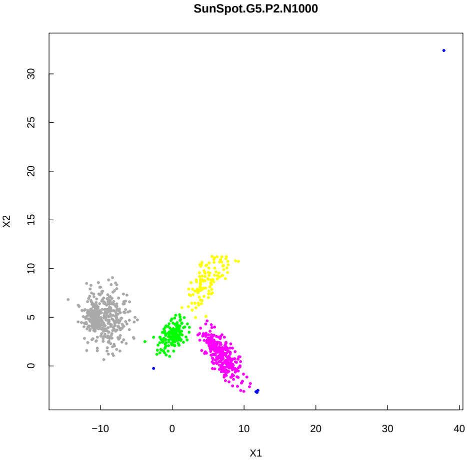
$$f(x) = \sum_{j=1}^{s} \pi_j \varphi_{\mathbf{a}_j,\Sigma_j}(x)$$

Compute $\hat{\pi}_j, \hat{\mathbf{a}}_j, \hat{\Sigma}$ by ML/EM-algorithm,
classify points by

$$\hat{\gamma}(i) = \arg\max_k \frac{\hat{\pi}_k \varphi_{\hat{\mathbf{a}}_k,\hat{\Sigma}_k}(x_i)}{\sum_{j=1}^{s} \hat{\pi}_j \varphi_{\hat{\mathbf{a}}_j,\hat{\Sigma}_j}(x)}.$$

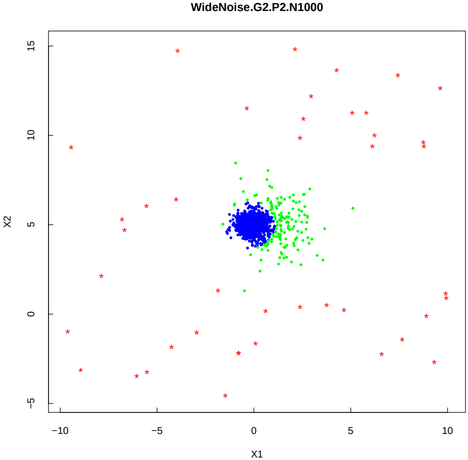
(Bayes rule, used for all mixture-based methods.)

Fixing $k = 5$ (or estimating k by BIC) gives this:

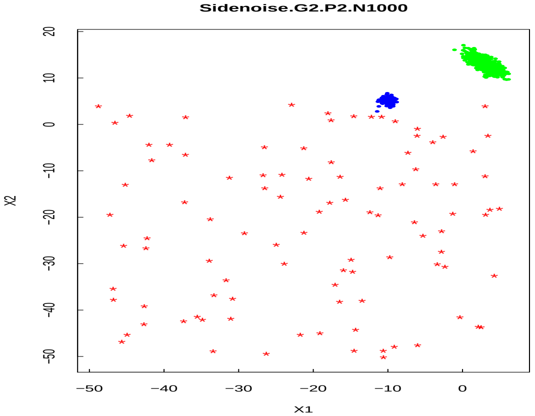


Gaussian clusters and extreme uniform on P=1,2; standard Gaussian on P>2

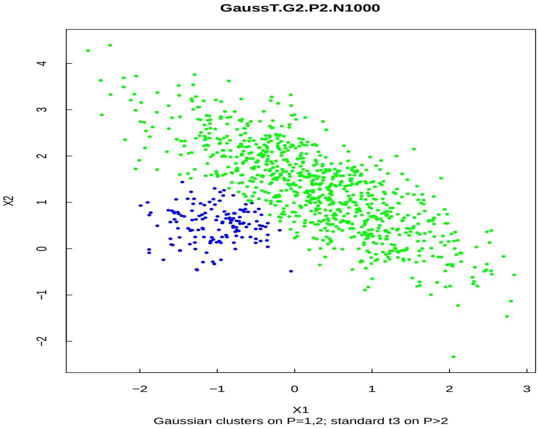
More challenges (2-d and 20-d):

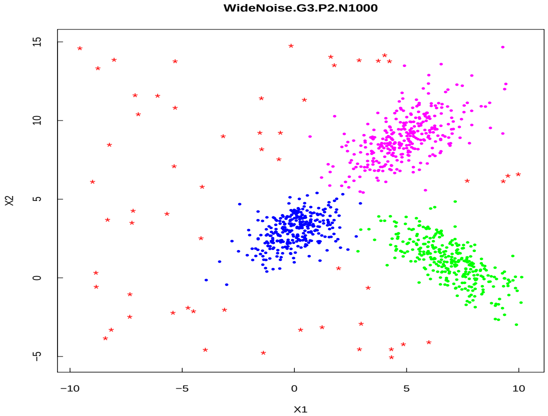


Gaussian clusters and wide uniform on P=1,2; standard Gaussian on P>2

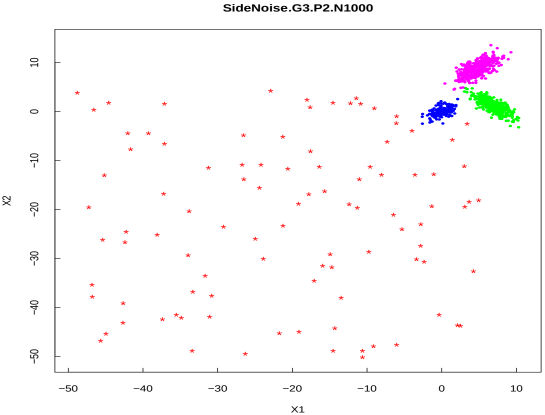


Gaussian clusters and side uniform on P=1,2; standard Gaussian on P>2

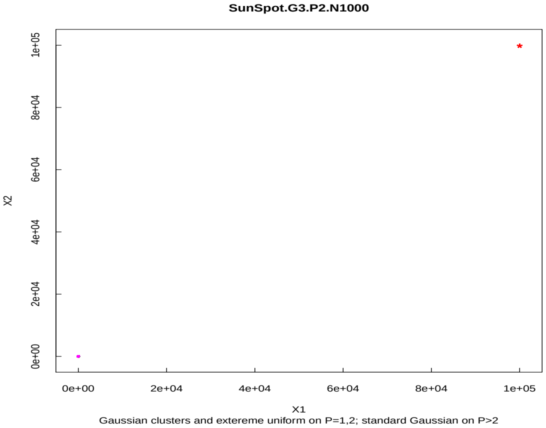

GaussT.G2.P2.N1000
X2
4
3
2
1
0
−1
−2
−2
−1
0
1
2
3
X1
Gaussian clusters on P=1,2; standard t3 on P>2

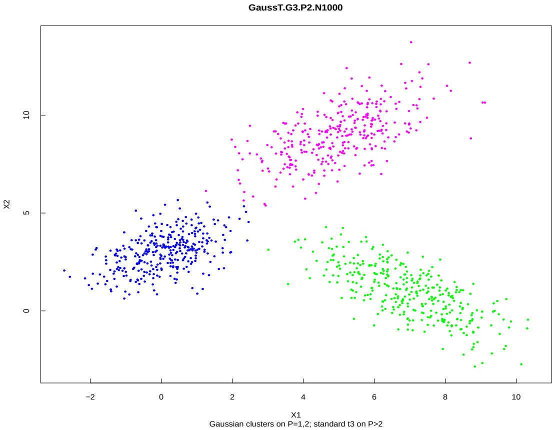


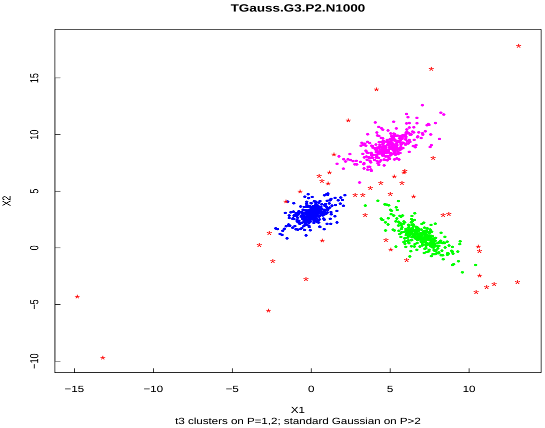
Gaussian clusters and wide uniform on P=1,2; standard Gaussian on P>2

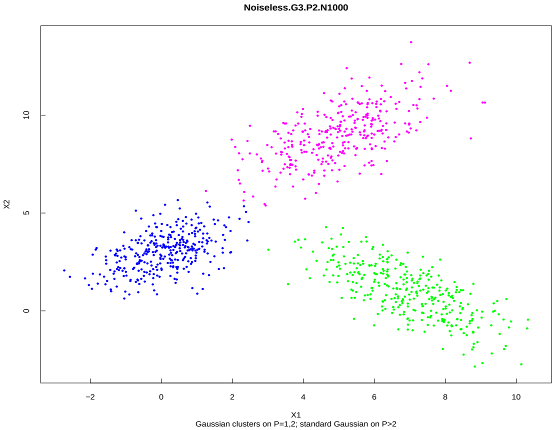


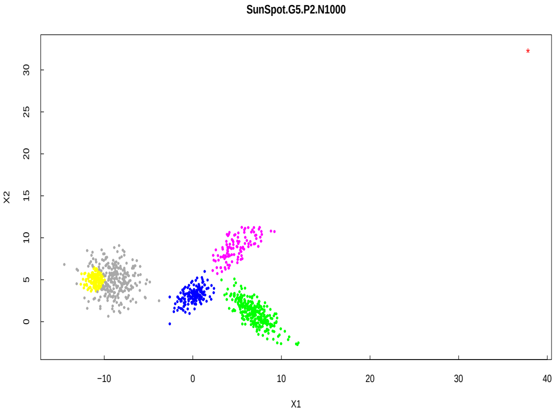
Gaussian clusters and side uniform on P=1,2; standard Gaussian on P>2


SunSpot.G3.P2.N1000
X2
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05
X1
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05
Gaussian clusters and extereme uniform on P=1,2; standard Gaussian on P>2

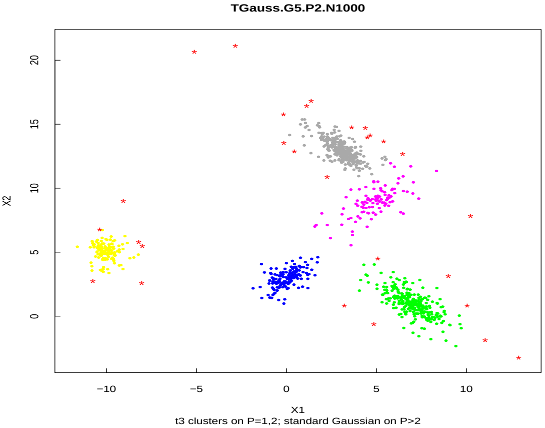

GaussT.G3.P2.N1000
X2
10
5
0
−2
0
2
4
6
8
10
X1
Gaussian clusters on P=1,2; standard t3 on P>2

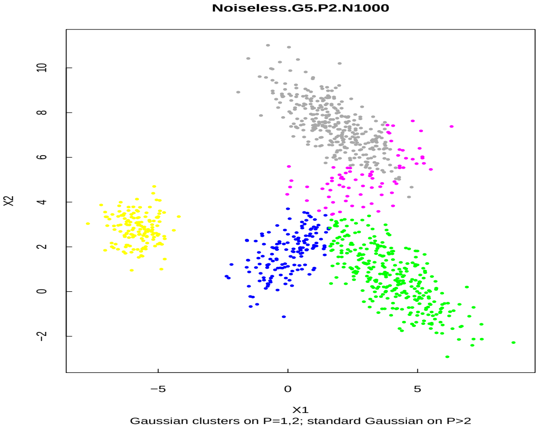

TGauss.G3.P2.N1000
X1
X2
−15
−10
−5
0
5
10
15
t3 clusters on P=1,2; standard Gaussian on P>2


Noiseless.G3.P2.N1000
X2
0
5
10
−2
0
2
4
6
8
10
X1
Gaussian clusters on P=1,2; standard Gaussian on P>2



Gaussian clusters and extreme uniform on P=1,2; standard Gaussian on P>2


TGauss.G5.P2.N1000
X2
X1
20
15
10
5
0
−10
−5
0
5
10
t3 clusters on P=1,2; standard Gaussian on P>2


Noiseless.G5.P2.N1000
X2
10
8
6
4
2
0
-2
−5
0
5
X1
Gaussian clusters on P=1,2; standard Gaussian on P>2

Comparative simulation study with these setups;
k fixed, $n = 1000$ ($p = 2$), $n = 2000$ ($p = 20$).

Vary "nature" of outliers/noise, number of clusters, cluster separation, cluster shape (although more could be tried).

Model-based methods for elliptical clustering with outliers.

4. Measurement of quality

Clustering is about classifying points,
and parameters are not the same for all methods,
so use *misclassification rates*.

Need to define “truth”.

Naive approach:

“true clusters” (and “true outliers”)
defined by mixture component generating the points.

Need to define “truth”.

Naive approach:

“true clusters” (and “true outliers”)
defined by mixture component generating the points.

Problem:

t-distributions generate outliers! (Or not?)
Uniform “noise” may be in the middle of a cluster
and in reality
“true generating mixture component” doesn’t exist.

Approach to define true clusters and outliers
inspired by Davies/Gather/Becker’s (1999) *outlier region*.

$$\mathrm{OR}_\alpha(j) := \{y: \ (y - \mathbf{a}_j)'\Sigma_j^{-1}(y - \mathbf{a}_j) \geq \chi^2_{p,1-\alpha} \ \forall j = 1, 2, \ldots, s\},$$

kth cluster: (tuned $\alpha = 0.0001$)

$$\{y: \ y \notin \mathrm{OR}_\alpha(k) \text{ and } k = \arg\max_{j=1,\ldots,s} q_j(y)\},$$

where $q_j(y)$ true Bayes posterior probability (QDA).

For t-distribution: replace Σ_j
by (Gauss-adjusted) MCD-functional.

2.1 Robust Improper Maximum Likelihood (RIMLE) (Hennig & Coretto)

Fit “pseudo-density” by “pseudo-ML/EM”

$$f(x) = \pi_0 c + \sum_{j=1}^{s} \pi_j \varphi_{\mathbf{a}_j, \Sigma_j}(x),$$

with tuning constant c.

Tuning: choice of c, later.

Motivated by...

2.2 mclust with noise (Banfield & Raftery, 1993; BR)

Fit by ML/EM:

$$f(x) = \pi_0 \frac{1}{V} + \sum_{j=1}^{s} \pi_j \varphi_{\mathbf{a}_j,\Sigma_j}(x),$$

V volume of smallest hyperrectangle covering data.
Classifies points to "noise component" 0.

Hennig (2004):
With well separated clusters and extreme outliers,
BR breaks down and RIMLE doesn't.

2.3.1 Tuning of c:

Automatic tuning for **RIMLE** (**ORIMLE**):
Minimising, for $c \in [0, C]$,

$$\mathrm{K}_n(c) := \max_{i=1,2,\dots,n} \sum_{j=1}^{s} \hat{\pi}_j \left|\mathbb{M}_j(x_i; \hat{\eta}_n(c)) - \chi_p^2(x_i)\right|,$$

where, with $\hat{\delta}_{ij}(c)$ Mahalanobis-distance of x_i to comp. j,

$$\mathbb{M}_j(t; c) = \frac{1}{W} \sum_{i=1}^{n} \hat{\tau}_{ij}(c) \mathbf{1}(\hat{\delta}_{ij}(c) \leq t).$$

Idea: try to find c so that the non-outliers look like Gaussian mixture.

Version (**ORIMLEP**):
Minimise, for $c \in [0, C]$, with $\lambda = 0,\ 0.5,\ 1$,

$$\mathrm{K}_n(c) + \lambda \hat{\pi}_0,$$

to allow some non-normality if this helps to integrate more points into clusters.

2.3.2 Stop the likelihood from degenerating

$$f(x) = \pi_0 c + \sum_{j=1}^{s} \pi_j \varphi_{\mathbf{a}_j,\Sigma_j}(x)$$

Likelihood will degenerate if EV for a $\Sigma_j \rightarrow 0$.

Use Garcia-Escudero et al., 2008 constraints: $\frac{\lambda_{min}(\Sigma_j)}{\lambda_{max}(\Sigma_k)} \geq q$.

2.3.2 Stop the likelihood from degenerating

$$f(x) = \pi_0 c + \sum_{j=1}^{s} \pi_j \varphi_{\mathbf{a}_j, \Sigma_j}(x)$$

Likelihood will degenerate if EV for a $\Sigma_j \to 0$.
Use Garcia-Escudero et al., 2008 constraints: $\frac{\lambda_{min}(\Sigma_j)}{\lambda_{max}(\Sigma_k)} \geq q$.

Choose $q = 18$ in simulation study (*all methods*),
but could choose differently in practice.

2.4 Computation of the RIMLE

2.4.1 Algorithm

Standard EM-algorithm can be used.
Need some tricky decisions about degenerating cases, i.e., $\pi_j \to 0$.

2.4 Computation of the RIMLE

2.4.1 Algorithm

Standard EM-algorithm can be used.
Need some tricky decisions about degenerating cases,
i.e., $\pi_j \longrightarrow 0$.

If EV-ratio constraint violated at end of algorithm,
discard solution for ORIMLE, unless this happens for all c.
(Enforce non-boundary solution if at all possible.)

2.4.2 Initialisation

Difficult and influential issue!

Now use mclust-inspired scheme:

- Find initial outliers by `NNclean` (Byers and Raftery 1998).
- Use hierarchical clustering based on plain Gaussian mixture likelihood on “non-outliers” (unconstrained cov-matrices)
- Reclassify points using “true cluster” definition.

3. Other methods

3.1 Plain Gaussian mixture ML (GM)
3.2 mclust with noise (BR)
3.3 Mixture of t_3-distributions (tmix)
(McLachlan & Peel 2000)
3.4 tclust 10%-trimmed clustering
(Garcia-Escudero et al., 2008)

Unification/comparability issues

- Use EV ratio constraint $q = 18$ for all methods.

Unification/comparability issues

- Use EV ratio constraint $q = 18$ for all methods.
- Use same initialisation for all methods except tclust (use software default there.)

Unification/comparability issues

- Use EV ratio constraint $q = 18$ for all methods.
- Use same initialisation for all methods except tclust (use software default there.)

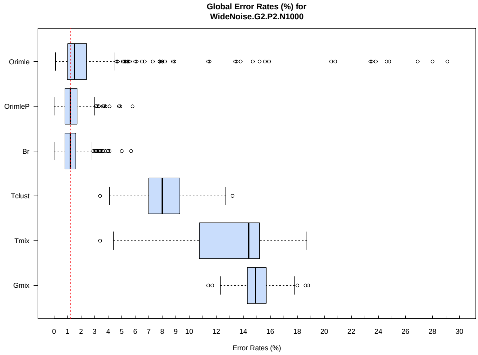
Tried tclust-type initialisation before
but found it slightly worse than hierarchical.

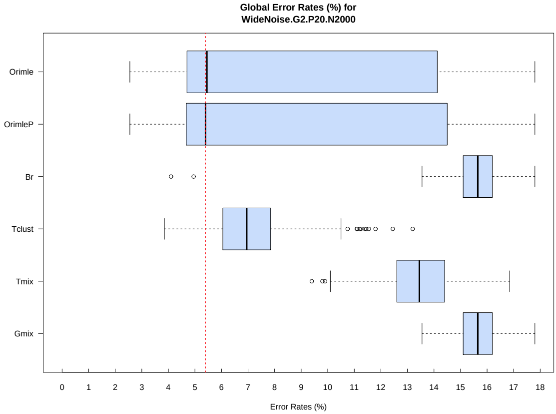
On the other hand, tclust initialisation always worked,
whereas mclust's hc( ) didn't in some situations.

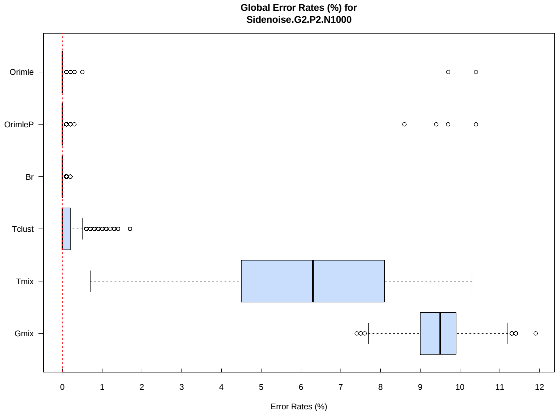
Unification/comparability issues

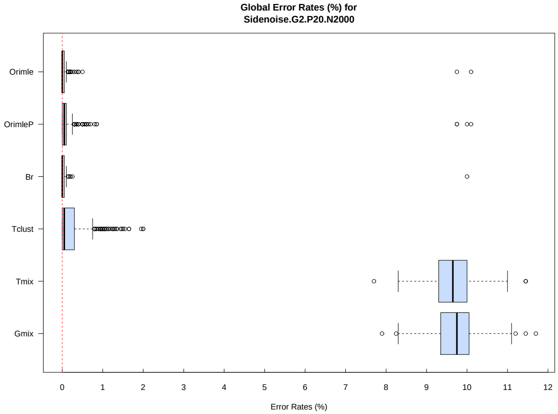
- Use EV ratio constraint $q = 18$ for all methods.
- Use same initialisation for all methods except tclust (use software default there.)
- Use "true clusters" definition to reclassify outliers and clusters for all methods.
(10% fixed trimming rate in tclust doesn't imply 10% "classified outliers".)

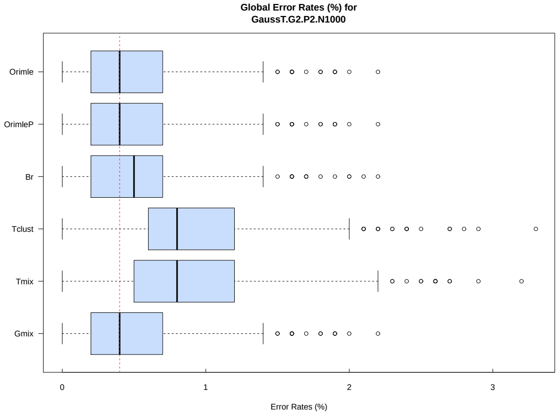
4. The results

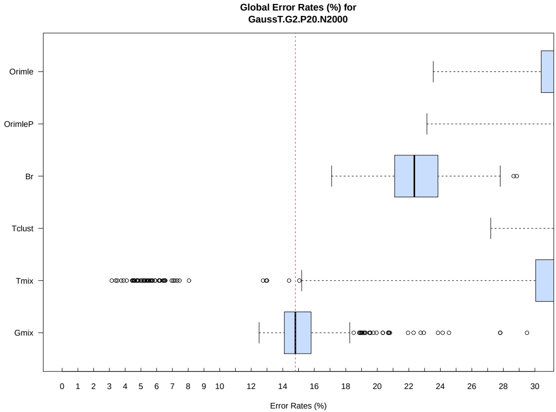


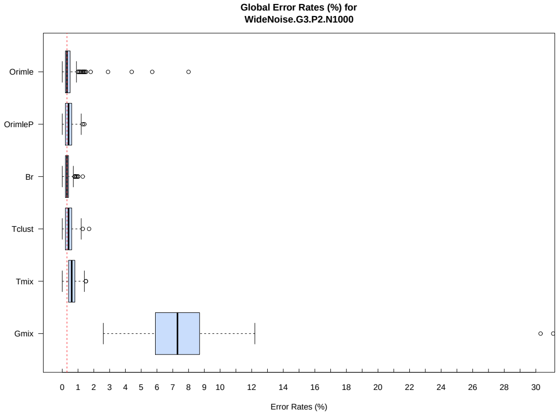

Global Error Rates (%) for
WideNoise.G2.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
Error Rates (%)

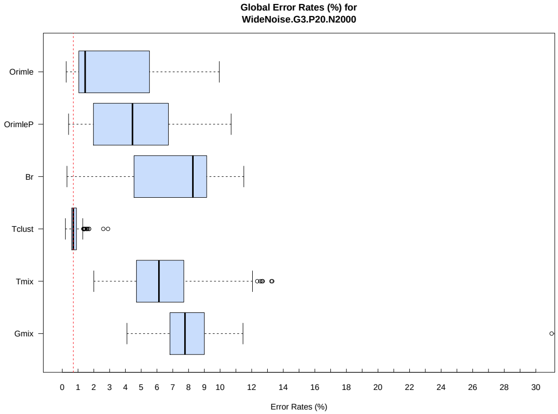

Global Error Rates (%) for
Sidenoise.G2.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
11
12
Error Rates (%)

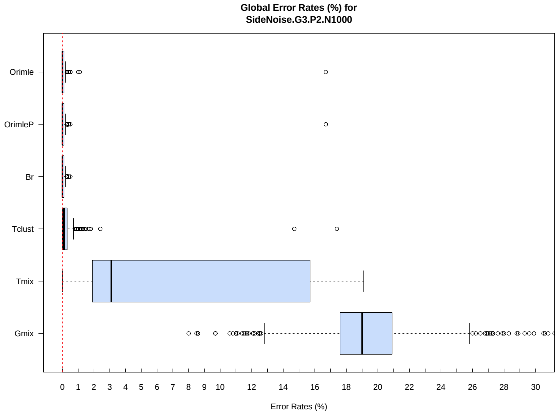

Global Error Rates (%) for
Sidenoise.G2.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
11
12
Error Rates (%)

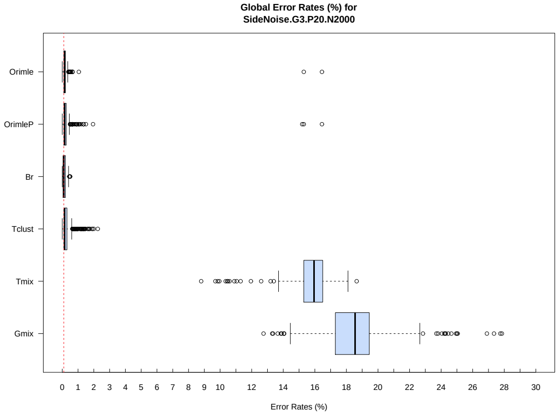

Global Error Rates (%) for
GaussT.G2.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
Error Rates (%)

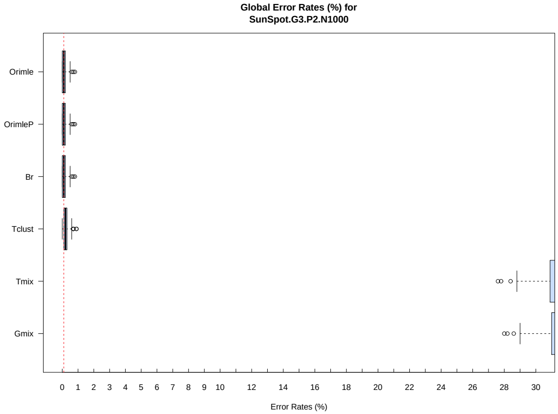

Global Error Rates (%) for
GaussT.G2.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
12
14
16
18
20
22
24
26
28
30
Error Rates (%)

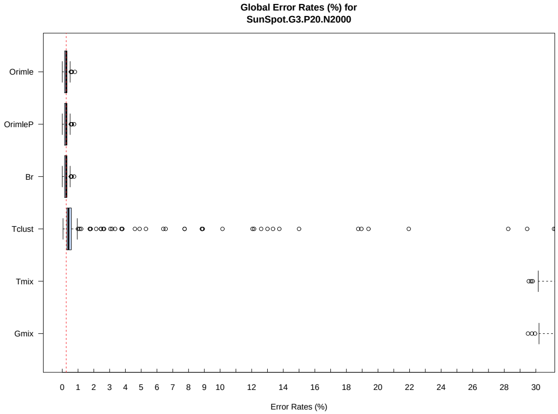

Global Error Rates (%) for
WideNoise.G3.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
12
14
16
18
20
22
24
26
28
30
Error Rates (%)

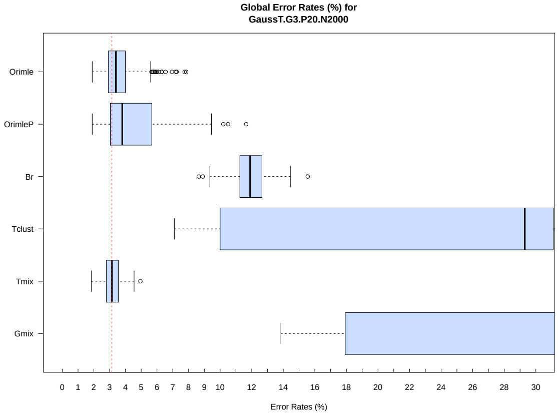

Global Error Rates (%) for
WideNoise.G3.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
12
14
16
18
20
22
24
26
28
30
Error Rates (%)

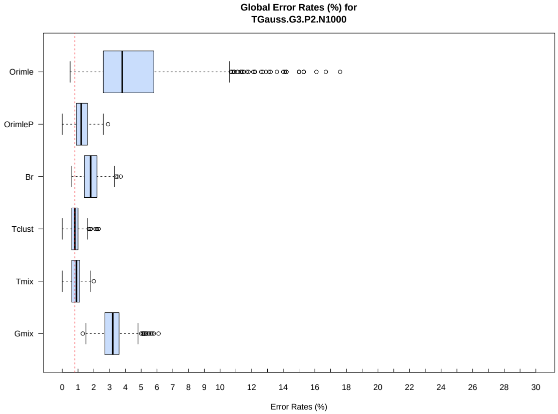

Global Error Rates (%) for
SideNoise.G3.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30
Error Rates (%)

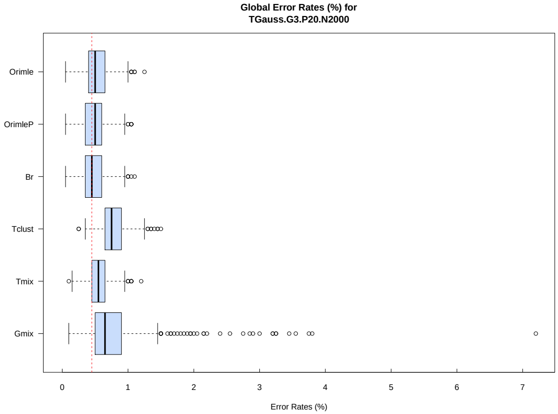

Global Error Rates (%) for
SideNoise.G3.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30
Error Rates (%)

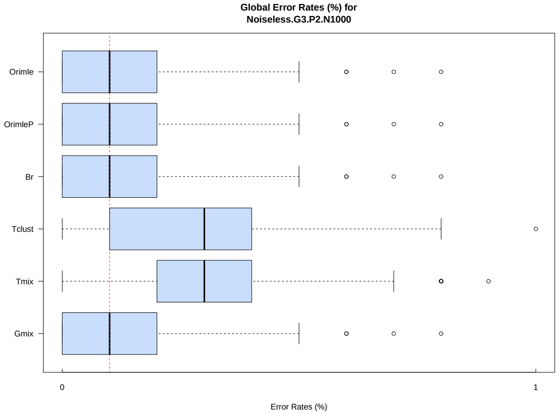

Global Error Rates (%) for
SunSpot.G3.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30
Error Rates (%)

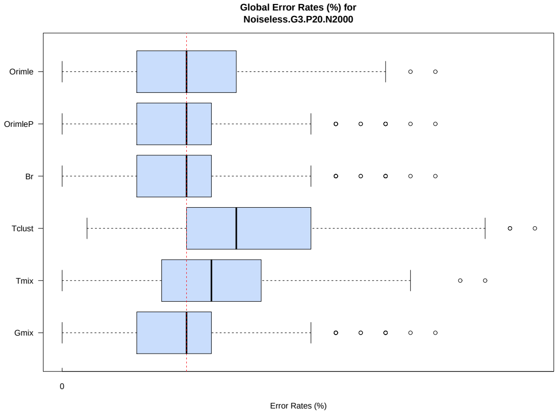

Global Error Rates (%) for
SunSpot.G3.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
12
14
16
18
20
22
24
26
28
30
Error Rates (%)

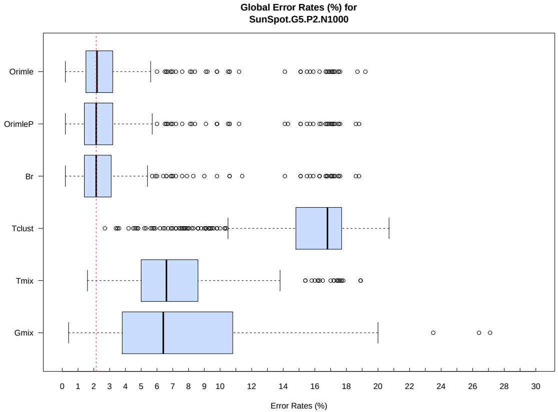

Global Error Rates (%) for
GaussT.G3.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30
Error Rates (%)

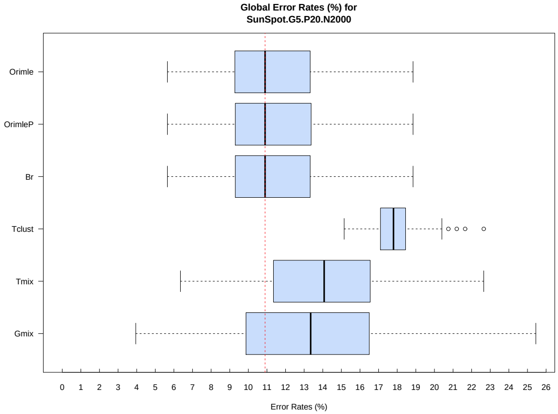

Global Error Rates (%) for
TGauss.G3.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30
Error Rates (%)

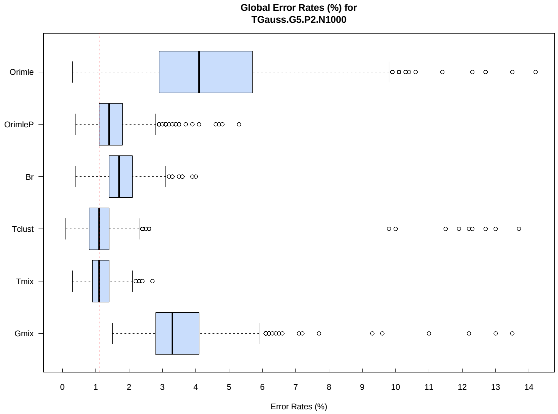

Global Error Rates (%) for
TGauss.G3.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
Error Rates (%)

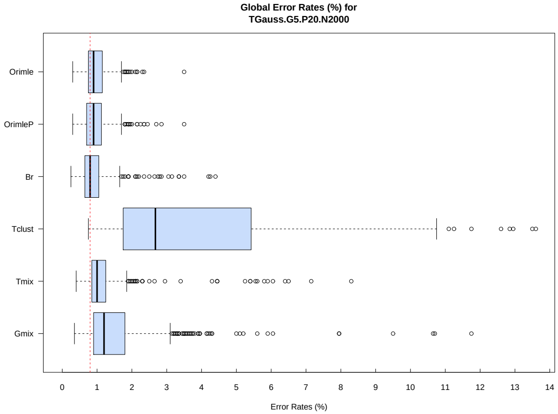

Global Error Rates (%) for
Noiseless.G3.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
Error Rates (%)

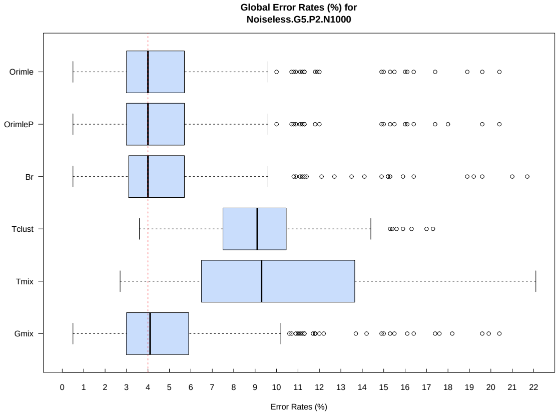

Global Error Rates (%) for
Noiseless.G3.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
Error Rates (%)

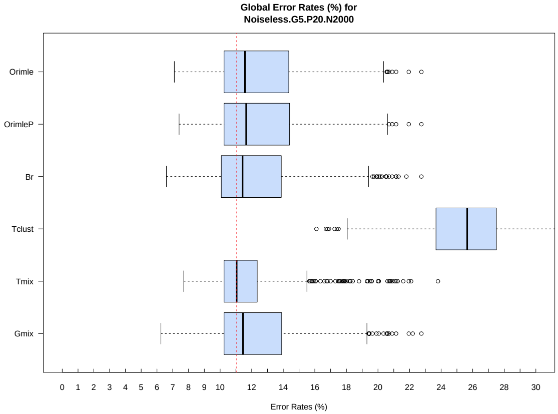

Global Error Rates (%) for
SunSpot.G5.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
12
14
16
18
20
22
24
26
28
30
Error Rates (%)


Global Error Rates (%) for
SunSpot.G5.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
Error Rates (%)


Global Error Rates (%) for
TGauss.G5.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Error Rates (%)


Global Error Rates (%) for
TGauss.G5.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Error Rates (%)


Global Error Rates (%) for
Noiseless.G5.P2.N1000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
Error Rates (%)


Global Error Rates (%) for
Noiseless.G5.P20.N2000
Orimle
OrimleP
Br
Tclust
Tmix
Gmix
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30
Error Rates (%)

The lessons to learn

- Results depend strongly on setup (also on p).

The lessons to learn

- Results depend strongly on setup (also on p).
- Different results for “same” model, $p = 2, 20$.

The lessons to learn

- Results depend strongly on setup (also on p).
- Different results for “same” model, $p = 2, 20$.
- GM generally bad with outliers (as expected).

The lessons to learn

- Results depend strongly on setup (also on p).
- Different results for "same" model, $p = 2, 20$.
- GM generally bad with outliers (as expected).
- BR and ORIMLE similar,
  but BR suffers if there is much t.

The lessons to learn

- Results depend strongly on setup (also on p).
- Different results for "same" model, $p = 2, 20$.
- GM generally bad with outliers (as expected).
- BR and ORIMLE similar,
  but BR suffers if there is much t.
- tmix not very competitive except t-setups.

The lessons to learn

- Results depend strongly on setup (also on p).
- Different results for "same" model, $p = 2, 20$.
- GM generally bad with outliers (as expected).
- BR and ORIMLE similar,
  but BR suffers if there is much t.
- tmix not very competitive except t-setups.
- tclust has highs and lows.
  May suffer from fixed trimming rate and
  (to some extent) from initialisation.

The lessons to learn

- Results depend strongly on setup (also on p).
- Different results for "same" model, $p = 2, 20$.
- GM generally bad with outliers (as expected).
- BR and ORIMLE similar,
  but BR suffers if there is much t.
- tmix not very competitive except t-setups.
- tclust has highs and lows.
  May suffer from fixed trimming rate and
  (to some extent) from initialisation.
- RIMLE automatic tuning not always optimal for same λ.

The lessons to learn

- Results depend strongly on setup (also on p).
- Different results for “same” model, $p = 2, 20$.
- GM generally bad with outliers (as expected).
- BR and ORIMLE similar,
  but BR suffers if there is much t.
- tmix not very competitive except t-setups.
- tclust has highs and lows.
  May suffer from fixed trimming rate and
  (to some extent) from initialisation.
- RIMLE automatic tuning not always optimal for same λ.
- GaussT.G2.P20 makes everything break down.

6. Tuning revisited, and conclusion

6.1 Tuning revisited

Against automatic tuning:

- *The problem definition* requires user tuning, so methods *should* be tuned as well.

6. Tuning revisited, and conclusion

6.1 Tuning revisited

Against automatic tuning:

- *The problem definition* requires user tuning, so methods *should* be tuned as well.
- Automatic tuning doesn't always work well and is difficult.
  Garcia-Escudero et al. (2010) recommend manual tuning with graphical diagnostics.

  Forward search (Atkinson and Riani 2007) use graphical diagnostics, too

In favour of automatic tuning:

- Good for simulation; no interaction between simulated setup and manual tuning.

In favour of automatic tuning:

- Good for simulation; no interaction between simulated setup and manual tuning.
- Users want automatic tuning,
  and many will mess up manual tuning.

6.2 Conclusion

- Comparative simulation studies in "robustness spirit" need to measure how methods do what they are not exactly supposed to do.

6.2 Conclusion

- Comparative simulation studies in “robustness spirit” need to measure how methods do what they are not exactly supposed to do.
- Relative results on robust clustering depend on a plethora of factors.

6.2 Conclusion

- Comparative simulation studies in “robustness spirit” need to measure how methods do what they are not exactly supposed to do.
- Relative results on robust clustering depend on a plethora of factors.
- Robust clustering problem definition requires tuning. Shouldn’t methods be manually tuned, too?

6.2 Conclusion

- Comparative simulation studies in "robustness spirit" need to measure how methods do what they are not exactly supposed to do.
- Relative results on robust clustering depend on a plethora of factors.
- Robust clustering problem definition requires tuning. Shouldn't methods be manually tuned, too?
- ORIMLE looks best but tried fairly hard to achieve this.

6.3 Postscriptum on research ethic

- All researchers want their own method to look good.
- Endless possibilities to make the own method “win”.
- It’s legitimate to improve own method to deal better with some models.
- It’s legitimate to choose models that demonstrate specific pros and cons of methods. But try to cause trouble for own method, too.
- Should try hard to tune competing methods to high quality (using original idea and not knowledge of simulated setups).
- Unfortunately, in published studies it cannot be checked how hard the researchers tried.