

# The S-Plus Robust Library for R

[Kjell Konis](#)

January 26, 2006

# Outline

- Features of the S-Plus Robust Library
  - Which models are available
  - Focus on usability
  - Examples for linear models and covariance estimation
- Porting the Robust Library to an R package
  - Maintain compatibility with S-Plus
- Where to go from there

# The S-Plus Robust Library

- The Insightful Robust Library Team
  - Doug Martin, Kjell Konis,  
Matias Salibian-Barrera, Jeff Wang
- Primary Consultants
  - Victor Yohai, Alfio Marazzi
- Additional Consultants
  - Ricardo Maronna, David Rocke,  
Ruben Zamar

# Important Code Contributions

- Alfio Marazzi
  - Extensive use of the Robeth Library: Marazzi (1993) *Algorithms, Routines and S Functions for Robust Statistics*
- Peter Rousseeuw
  - *A Fast MCD Algorithm for the Minimum Covariance Determinant Estimator* (with K. V. Driessen)
  - Earlier contributions that originally put S-Plus “on the map” with robust methods: LMS, LTS, MVE

# Goals of the Robust Library

- Achieve routine use by data analysts
  - Should be “just like” doing classical model fitting
- Take advantage of the wealth of research results on robustness theory and methods
- Visual comparison of classical versus robust fits
- Classical and robust t-stats, p-values, F-tests, R-squared
- Robust model selection
- Automatically select computational method or estimate type based on problem type and size

# Methods in the Robust Library

- Linear regression
- Analysis of variance (fixed effects)
- Covariance/correlation estimation
- Principal components
- Discriminant analysis
- Generalized linear models (logistic, Poisson)
- Weibull, Gamma, log-normal distributions
- Contributed code: quantile regression, splines, robust Cp

# Conventions in the Robust Library

- GUI ease-of-use (command line for power users)
- Consistent and predictable syntax
  - Follow existing conventions when possible
- Robust inference as well as classical inference
- Automatic computational method selection
- Change the engine under the hood from time to time

# Consistent Syntax

- The goal is to make doing robust data analysis “just like” doing classical data analysis
- The end-user functions have the same arguments as their classical counterpart
- Control arguments (estimator, algorithm, tuning parameters, etc.) are put into a list and passed to the end-user function as a single argument
- Sensible default values

# Example: args of lm and lmRob

```
> args(lm)
function(formula, data, weights, subset, na.action,
          method = "qr", model = F, x = F, y = F,
          contrasts = NULL, ...)
```

```
> args(lmRob)
function(formula, data, weights, subset, na.action,
          model = F, x = F, y = F, contrasts = NULL,
          nrep = NULL,
          robust.control = lmRob.robust.control(...),
          genetic.control = NULL, ...)
```

# Object Oriented Features

- The `lmRob` object inherits from the `lm` object
- Generic functions produce the same output for `lmRob` objects as for `lm` objects
- The `plot` method produces the same diagnostic plots for `lmRob` objects as for `lm` objects
- Other methods (e.g. `anova`, `step`, `tec`.) behave in a similarly predictable way

# Robust Inference as well as Classical Inference

“... just which robust/resistant methods you use is not important – what is important is that you use some. It is perfectly proper to use both classical and robust/resistant methods routinely, and only worry when they differ enough to matter. But when they differ, you should think hard.”

– J. W. Tukey

# Fitting Multiple Models

- The `fit.models` function
- Fit multiple models on the same formula and data
- The generic functions then do sensible things: side-by-side Trellis plots, collated tabular output, etc.
- Example: the Stack Loss Data

```
stack.fit <- fit.models(  
  list(Robust = "lmRob", LS = "lm"),  
  formula = Loss ~ .,  
  data = stack.dat)
```

# The Short Output

Calls:

```
Robust : lmRob(formula = Loss ~ ., data = stack.dat)
      LS : lm(formula = Loss ~ ., data = stack.dat)
```

Coefficients:

	Robust	LS
(Intercept)	-37.6524589	-39.9196744
Air.Flow	0.7976856	0.7156402
Water.Temp	0.5773405	1.2952861
Acid.Conc.	-0.0670602	-0.1521225

Residual standard errors:

```
Robust : 1.837073 on 17 degrees of freedom
      LS : 3.243364 on 17 degrees of freedom
```

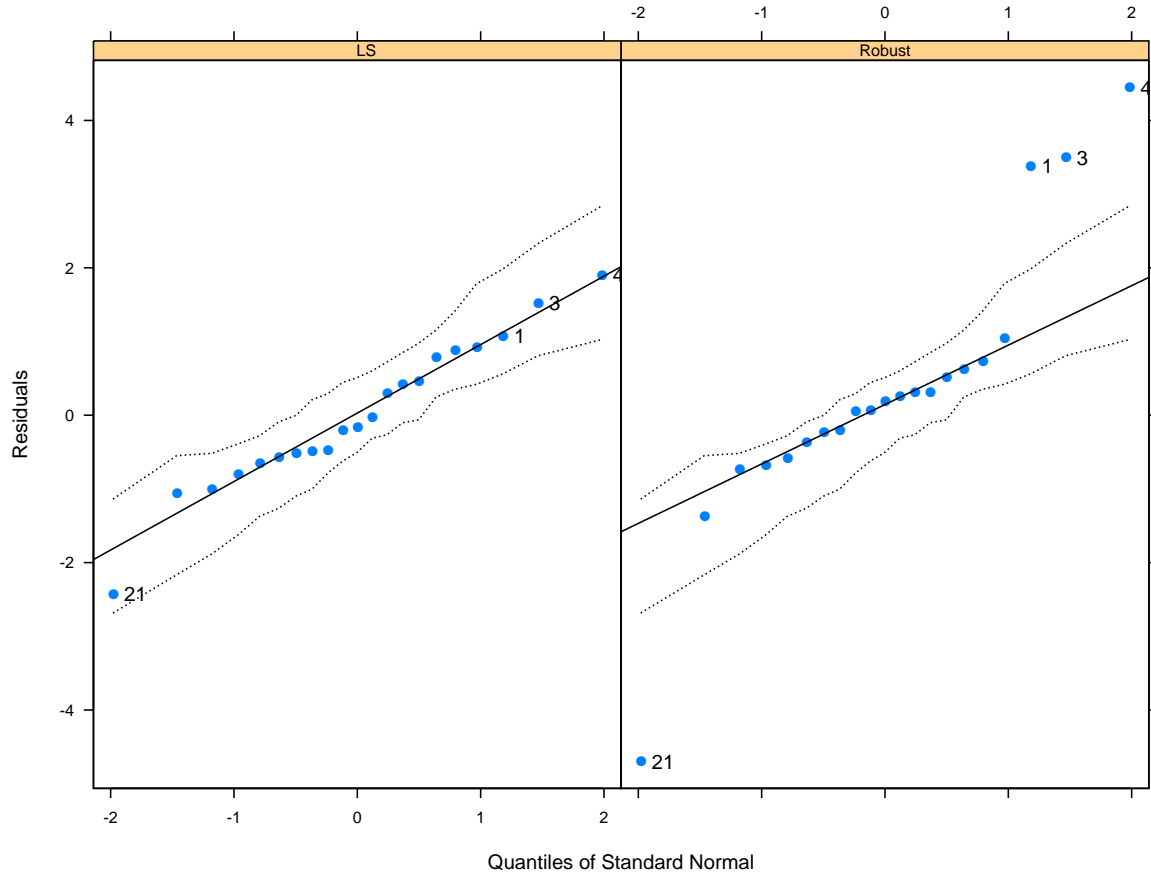
# The Plot Method

Invoking the plot method first offers a menu to choose the plot

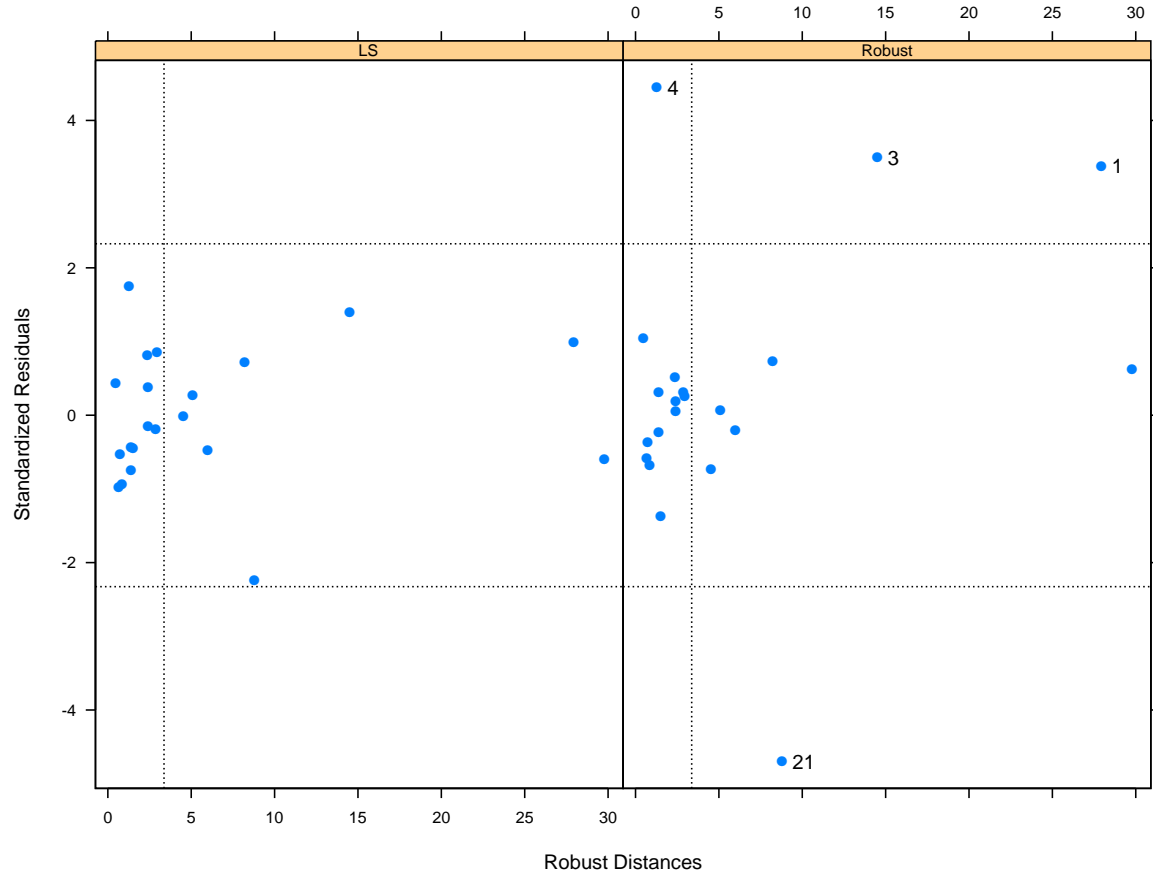
Make plot selections (or 0 to exit):

- 1: plot: All
  - 2: plot: Normal QQ-Plot of Residuals
  - 3: plot: Estimated Kernel Density of Residuals
  - 4: plot: Robust Residuals vs Robust Distances
  - 5: plot: Residuals vs Fitted Values
  - 6: plot: Sqrt of abs(Residuals) vs Fitted Values
  - 7: plot: Response vs Fitted Values
  - 8: plot: Standardized Residuals vs Index (Time)
  - 9: plot: Overlaid Normal QQ-Plot of Residuals
  - 10: plot: Overlaid Estimated Density of Residuals
- Selection(s):

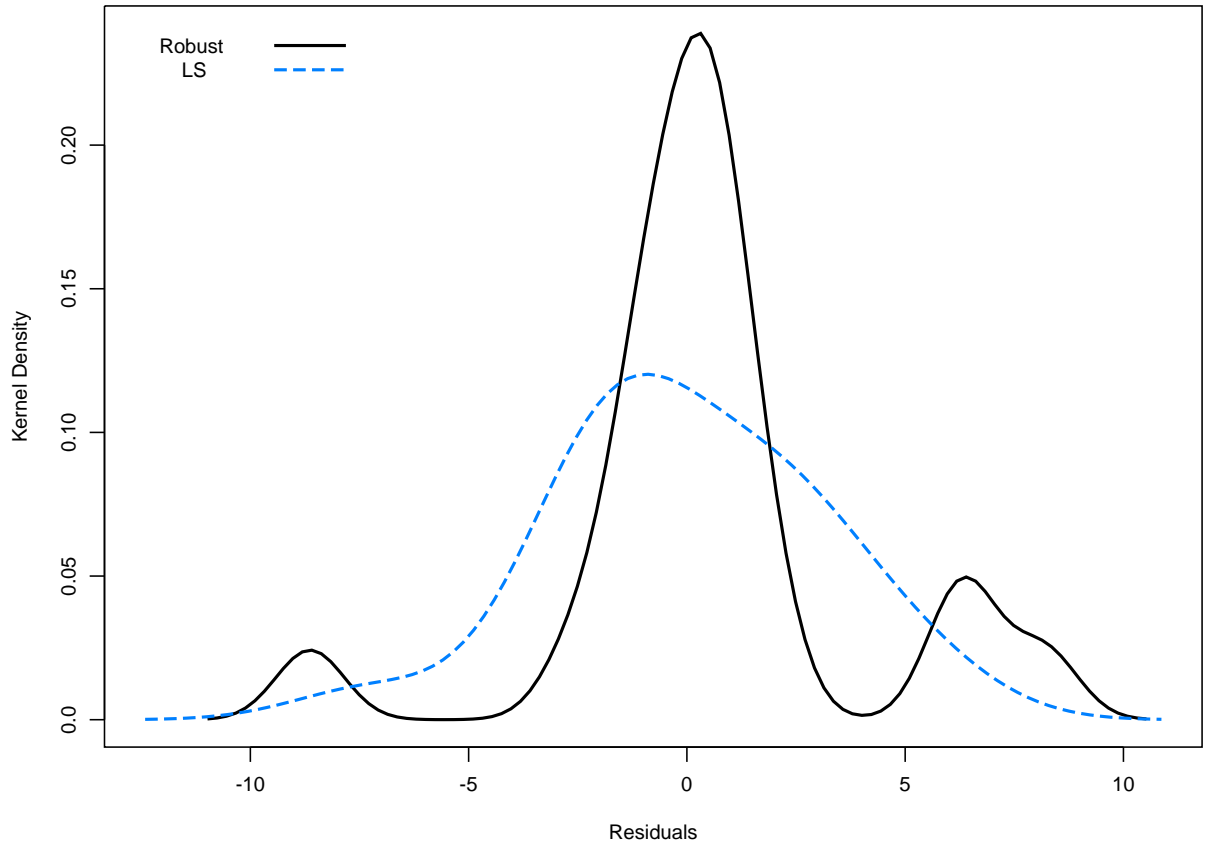
Normal QQ-Plot of Residuals



Standardized Residuals vs Robust Distances



Kernel Density of Residuals



# A Covariance/Correlation Example

- The Woodmod data set
  - A modified version of the wood gravity data set from Rousseeuw and Leroy (1987)

```
woodmod.fit <- fit.models(  
  list(Robust = "covRob",  
        Classical = "cov"),  
  data = woodmod.dat)
```

# The Print Method

Calls:

```
Robust : covRob(data = woodmod.dat)
```

```
Classical : cov(data = woodmod.dat)
```

Comparison of Covariance/Correlation Estimates:  
(unique covariance terms)

	[1,1]	[2,1]	[3,1]	[4,1]	[5,1]	
Robust	0.0102	0.0018	0.0011	-0.0002	-0.0008	...
Classical	0.0083	-0.0003	0.0036	0.0027	-0.0029	...

Comparison of Location Estimates:

	V1	V2	V3	V4	V5
Robust	0.5671	0.1165	0.5050	0.5520	0.9017
Classical	0.5509	0.1330	0.5086	0.5112	0.9070

# The Plot Method

- Again, invoking the plot method offers a menu to choose the plot

Make plot selections (or 0 to exit):

1: plot: All

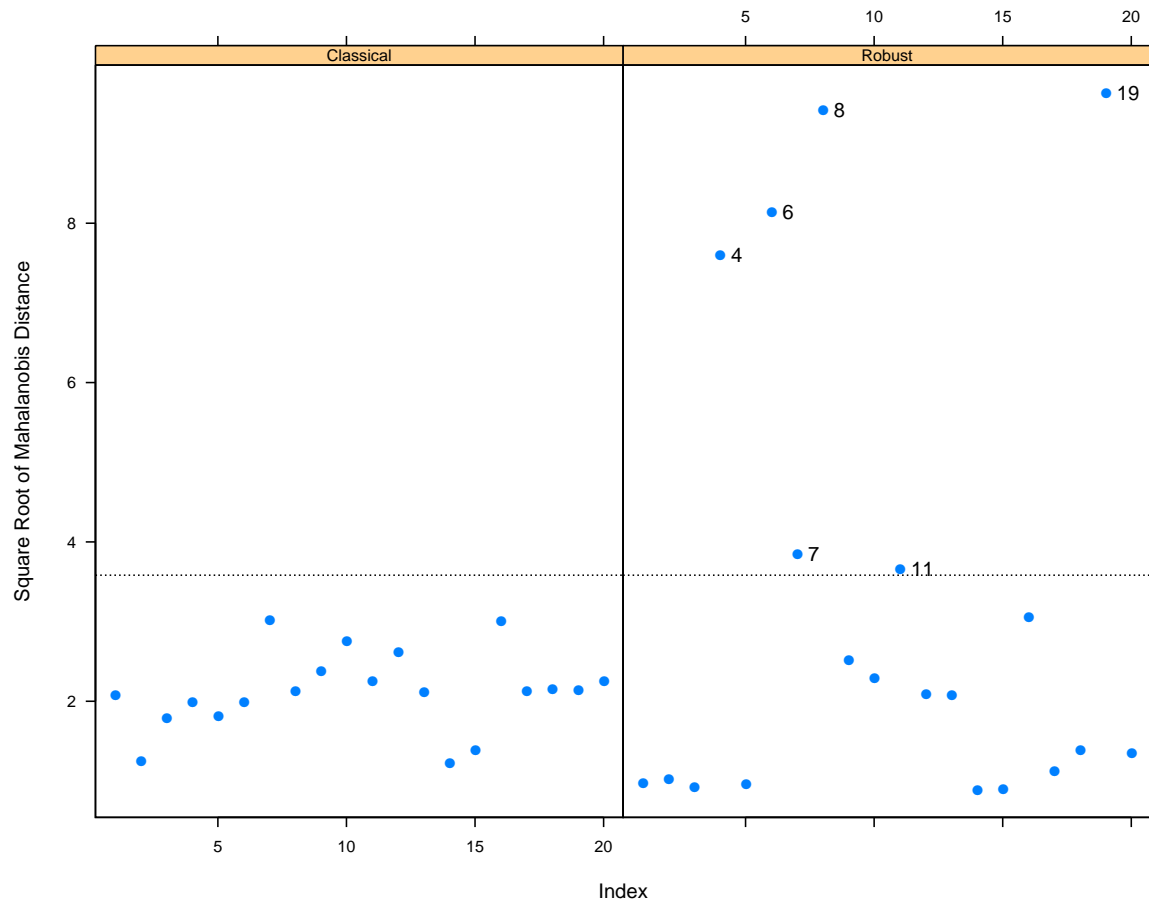
2: plot: Eigenvalues of Covariance Estimate

3: plot: Sqrt of Mahalanobis Distances

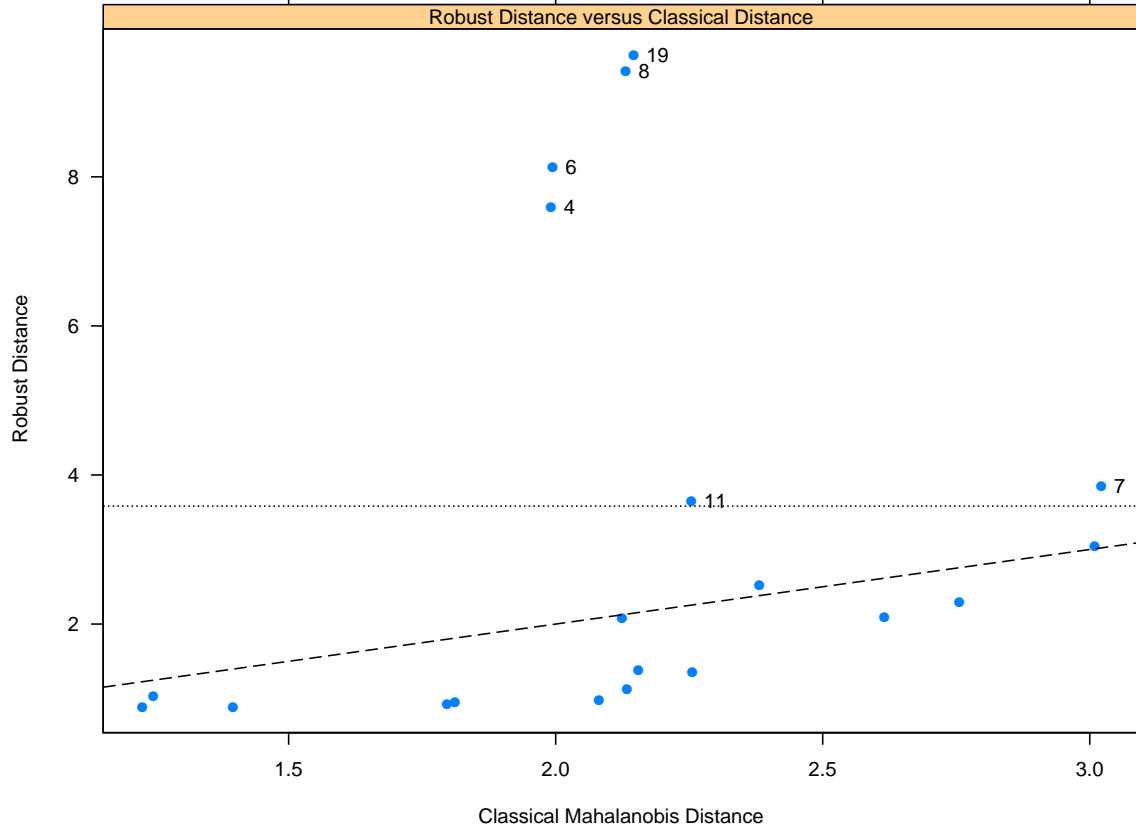
4: plot: Ellipses Matrix

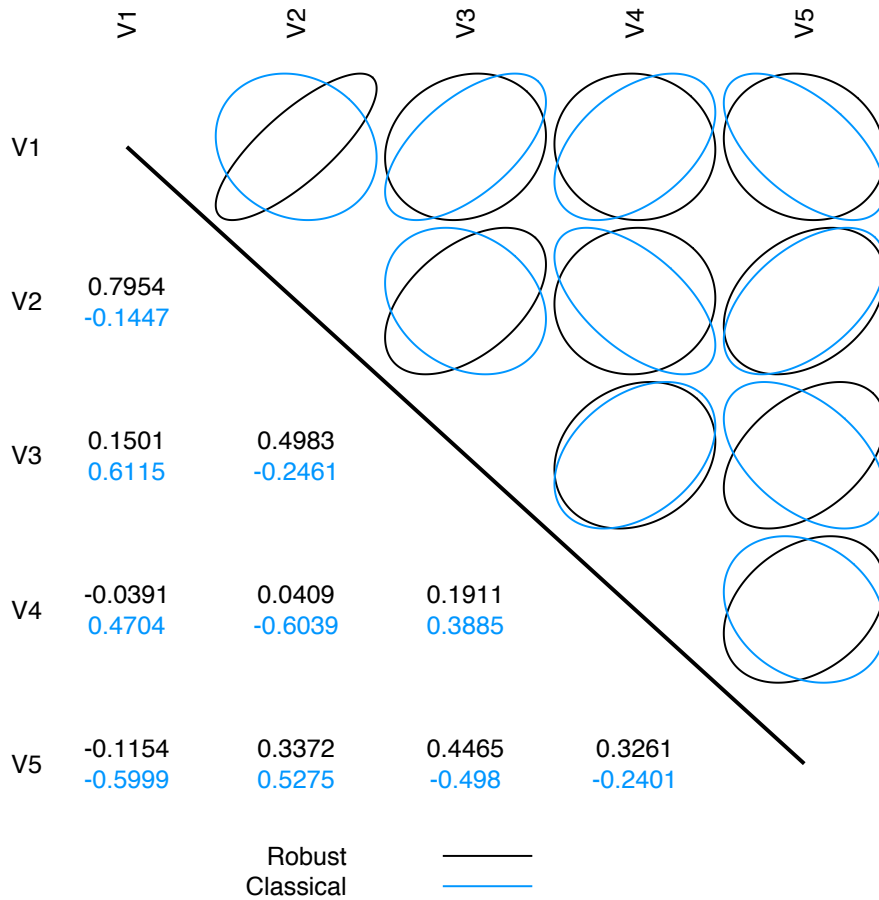
5: plot: Distance - Distance Plot

Selection(s):



Distance-Distance Plot





# Automated Method Selection

- By default the robust estimator is chosen based on the size of the problem.
- The default initial estimator in **lmRob** uses:
  - Exhaustive sampling if there are fewer than 250 observations and 2 variables or fewer than 80 observations and 3 variables
  - The “*fast*” procedure of Pena and Yohai (1999) if there are more than 15 numeric variables
  - Resampling otherwise
- The default final estimator in **lmRob** is an M-estimate

# Automated Method Selection (cont.)

- The default estimator used by `covRob` is:
  - The Donoho-Stahel projection-type estimator if there are fewer than 1000 observations and less than 10 variables or fewer than 5000 observations and less than 5 variables
  - The Fast MCD if there are fewer than 50000 observations and less than 20 variables
  - For larger problems the quadrant-correlation based pairwise estimator of Zamar et al. is used
- Of course power users are free to specify precisely which robust estimator is used

# Changes Under the Hood

- Generic functions operate on the fitted model object regardless of which robust estimator was used to create the object
- New robust methods can be added for fitting the end-user modeling functions without requiring a change in the command syntax
- Similarly, new class methods can be added without requiring changes to the underlying computational code

# Moving Forward

1. Port the Robust Library to R
2. Update the Robust Library/Package to be consistent with the upcoming "Robust Statistics: Theory and Methods" by Ricardo Maronna, Doug Martin and Victor Yohai
  - Maintain Compatibility with both S-Plus and R
3. Initial version available shortly - Final version later this year

The End