An Example Analysis Using LOLOG

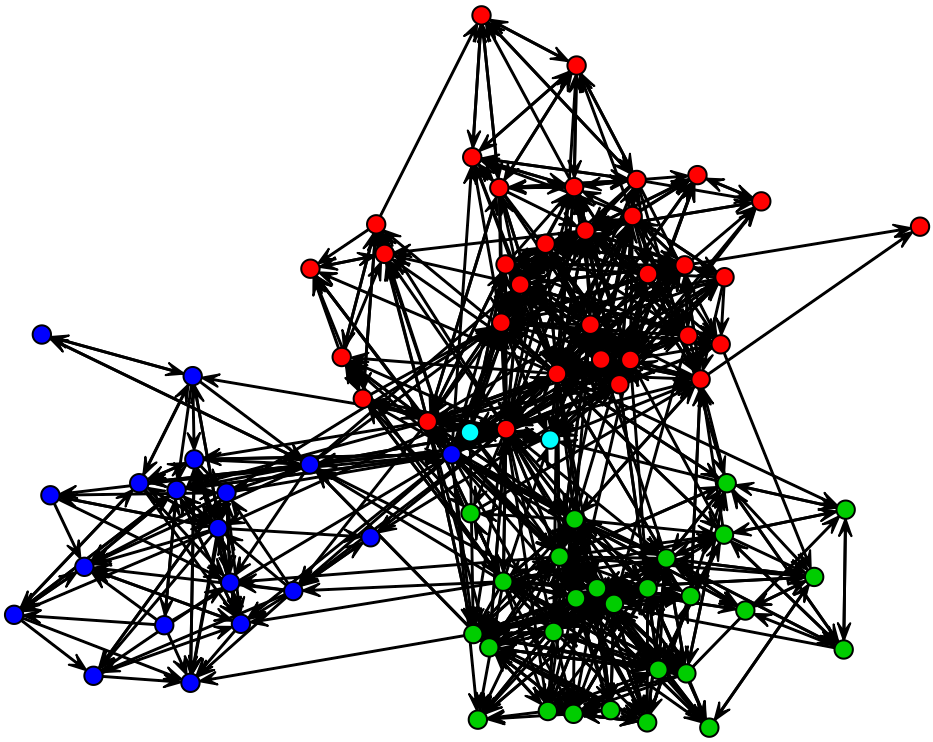
The Statnet Development Team

2018-04-09

The `ukFaculty` dataset

The personal friendship network of a faculty of a UK university, consisting of 81 vertices (individuals) and 817 directed and weighted connections. The school affiliation of each individual is stored as a vertex attribute.

```
suppressPackageStartupMessages(library(network))
library(lolog)
data(ukFaculty)
#?ukFaculty
ukFaculty %v% "Group"  # The school affiliation of the faculty
#>  [1] 3 1 3 3 2 2 2 1 3 2 1 2 2 1 1 2 3 1 1 1 1 2 2 1 1 1 2 2 1 2 1 1 2 1 1
#> [36] 3 1 3 1 2 1 2 1 3 3 1 2 1 2 4 1 1 3 1 1 1 1 1 3 3 3 3 2 1 2 2 2 2 2 4
#> [71] 2 2 3 3 3 2 2 3 1 1 3
ukFaculty %v% "GroupC" # affiliation coded as categorical
#>  [1] "3" "1" "3" "3" "2" "2" "2" "1" "3" "2" "1" "2" "2" "1" "1" "2" "3"
#> [18] "1" "1" "1" "1" "2" "2" "1" "1" "1" "2" "2" "1" "2" "1" "1" "2" "1"
#> [35] "1" "3" "1" "3" "1" "2" "1" "2" "1" "3" "3" "1" "2" "1" "2" "4" "1"
#> [52] "1" "3" "1" "1" "1" "1" "1" "3" "3" "3" "3" "2" "1" "2" "2" "2" "2"
#> [69] "2" "4" "2" "2" "3" "3" "3" "2" "2" "3" "1" "1" "3"
plot(ukFaculty, vertex.col = (ukFaculty %v% "Group" ) + 1)
```



We see a great number of like-to-like ties based on school affiliation, so this will probably be an important thing to model.

A first attempt

Recall from the introductory vignette, a LOLOG represents the probability of a tie, given the network grown up to a time-point as

$$\text{logit}\big(p(y_{s_t} = 1|\eta, y^{t-1}, s_{\leq t})\big) = \theta \cdot c(y_{s_t} = 1|y^{t-1}, s_{\leq t})$$

where $s_{\leq t}$ is the growth order of the network up to time t, y^{t-1} is the state of the graph at time $t-1$. $c(y_{s_t}|y^{t-1}, s_{\leq t})$ is a vector representing the change in graph statistics from time $t-1$ to t if an edge is present, and θ is a vector of parameters.

If the graph statistics are dyad independent (i.e. the change in graph statistics caused by the addition or deletion of an edge depends only on the vertex covariates of the two vertices connected by the edge), then LOLOG reduces to a simple logistic regression of the presence of an edge on the change statistics.

It is usually a good idea to start off model building with a dyad independent model.

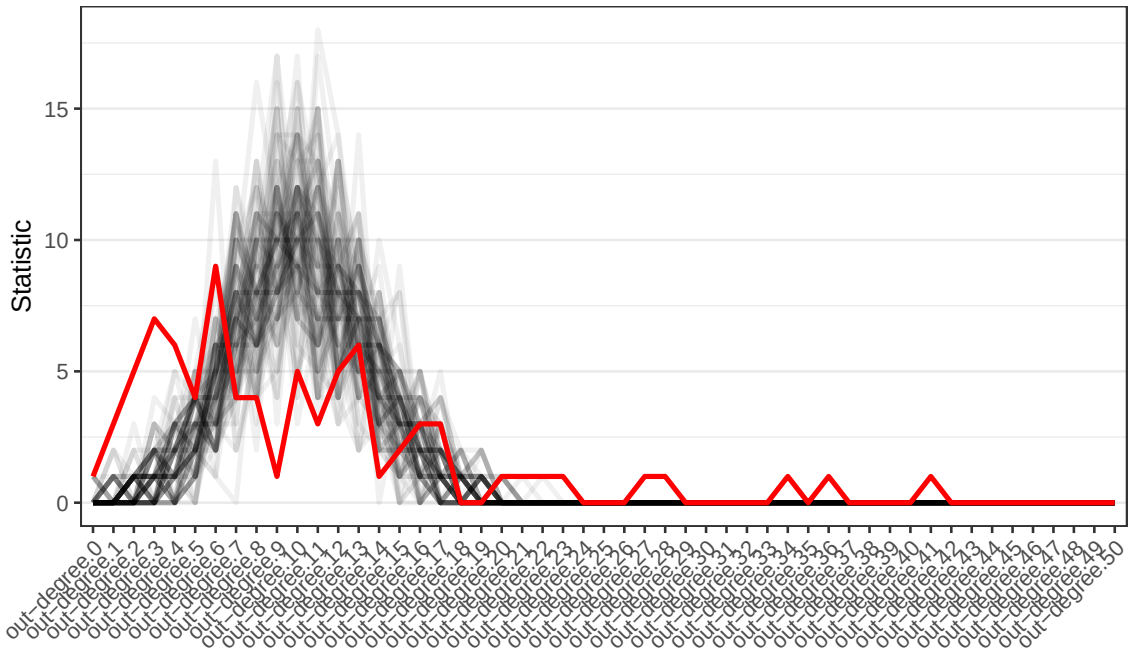
```
fitukInd <- lolog(ukFaculty ~ edges() + nodeMatch("GroupC"))
#> Initializing Using Variational Fit
#>
#>  Model is dyad independent. Replications are redundant. Setting nReplicates <- 1L.
#> Model is dyad independent. Returning maximum likelihood estimate.
summary(fitukInd)
#>                  observed_statistics     theta         se pvalue
#> edges                            817 -3.325131 0.08255648      0
#> nodematch.GroupC                 665  2.554605 0.09494854      0
```

For those familiar with ERGM modeling, dyad independent ergms are identical to dyad independent lolog models.

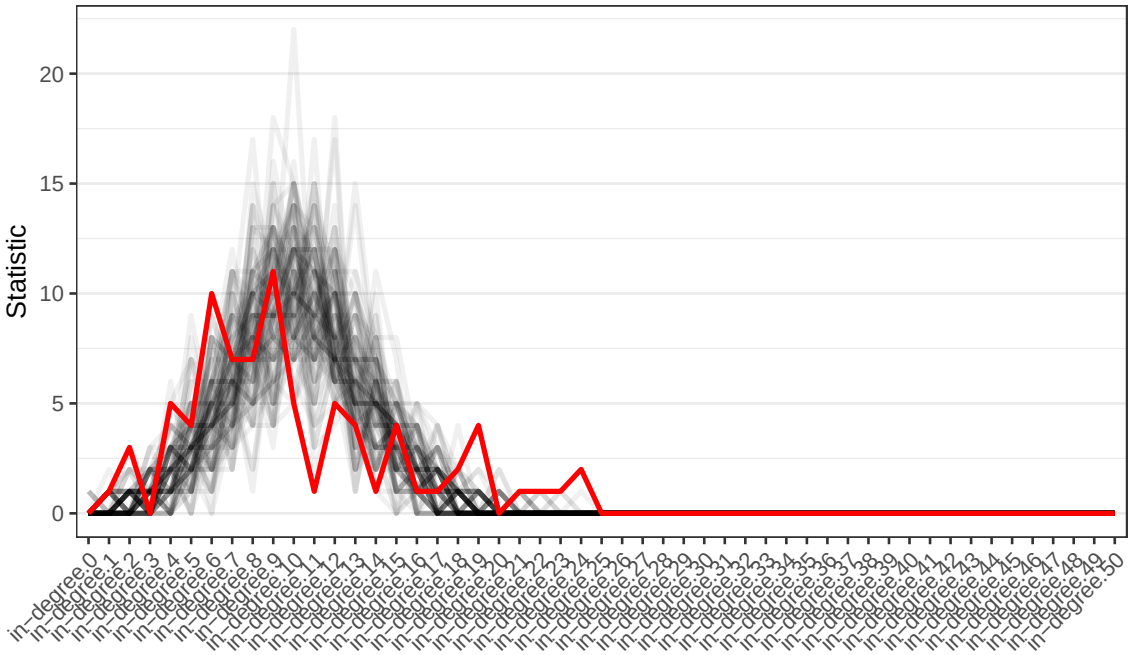
```
suppressPackageStartupMessages(library(ergm))
ergm(ukFaculty ~ edges() + nodematch("GroupC"))
#> Warning in is.na(x): is.na() applied to non-(list or vector) of type 'NULL'
#> Starting maximum pseudolikelihood estimation (MPLE):
#> Evaluating the predictor and response matrix.
#> Maximizing the pseudolikelihood.
#> Finished MPLE.
#> Stopping at the initial estimate.
#> Evaluating log-likelihood at the estimate.
#>
#> MLE Coefficients:
#>            edges  nodematch.GroupC
#>           -3.325             2.555
```

At this point we will evaluate the fit of our first LOLOG model by comparing the in-degree, out-degree and esp distribution of graphs simulated from the model to our observed graph.

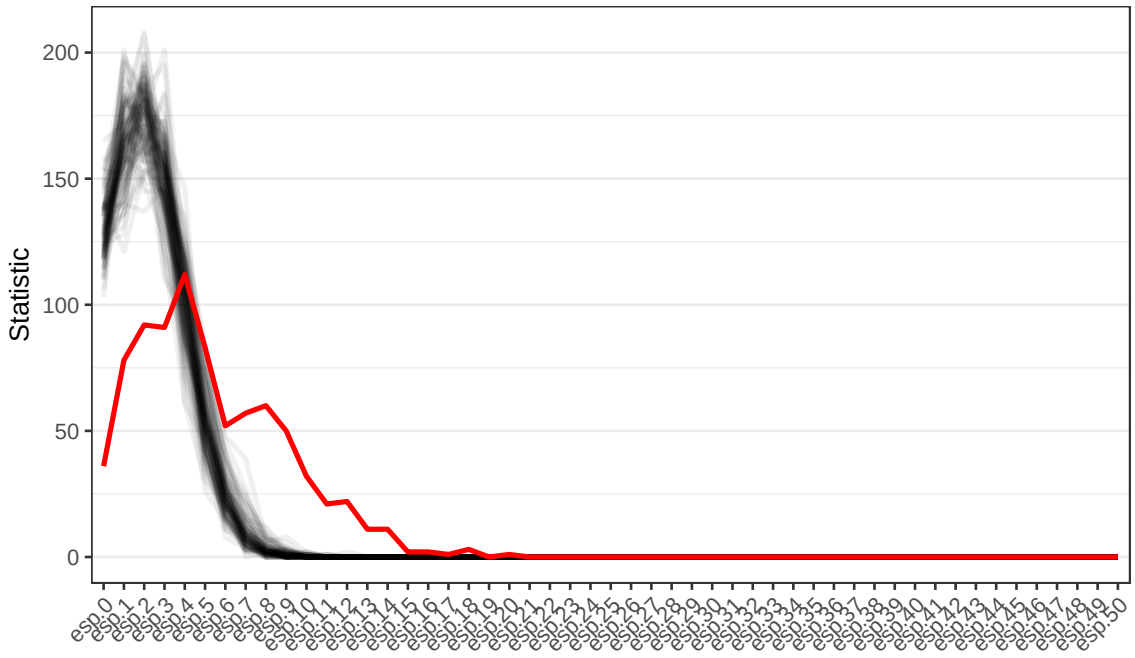
```
g <- gofit(fitukInd, ukFaculty ~ degree(0:50,"out"))
plot(g)
```




```
g <- gofit(fitukInd, ukFaculty ~ degree(0:50,"in"))
plot(g)
```

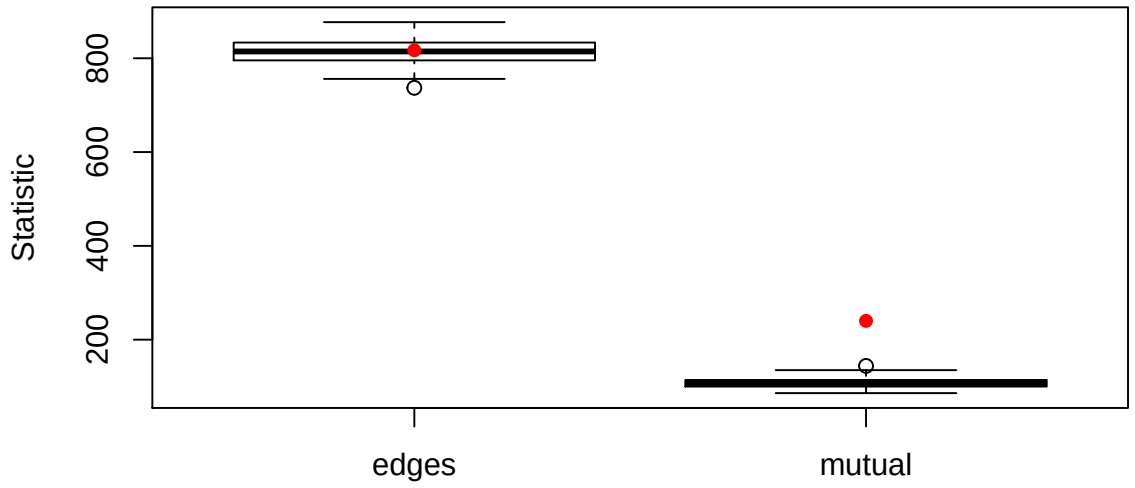



```
g <- gofit(fitukInd, ukFaculty ~ esp(0:50))
plot(g)
```



The statistics of the simulated networks are traced by the black lines, while out observed network is marked in red. The degree distributions are not terribly mismatched, but the ESP distribution indicates simulated networks have far less transitivity than the observed network. Additionally, the number of reciprocated ties (mutual) is far to low in the simulated graphs

```
g <- gofit(fitukInd, ukFaculty ~ edges +mutual)
plot(g,type="box")
```




```
#> NULL
```

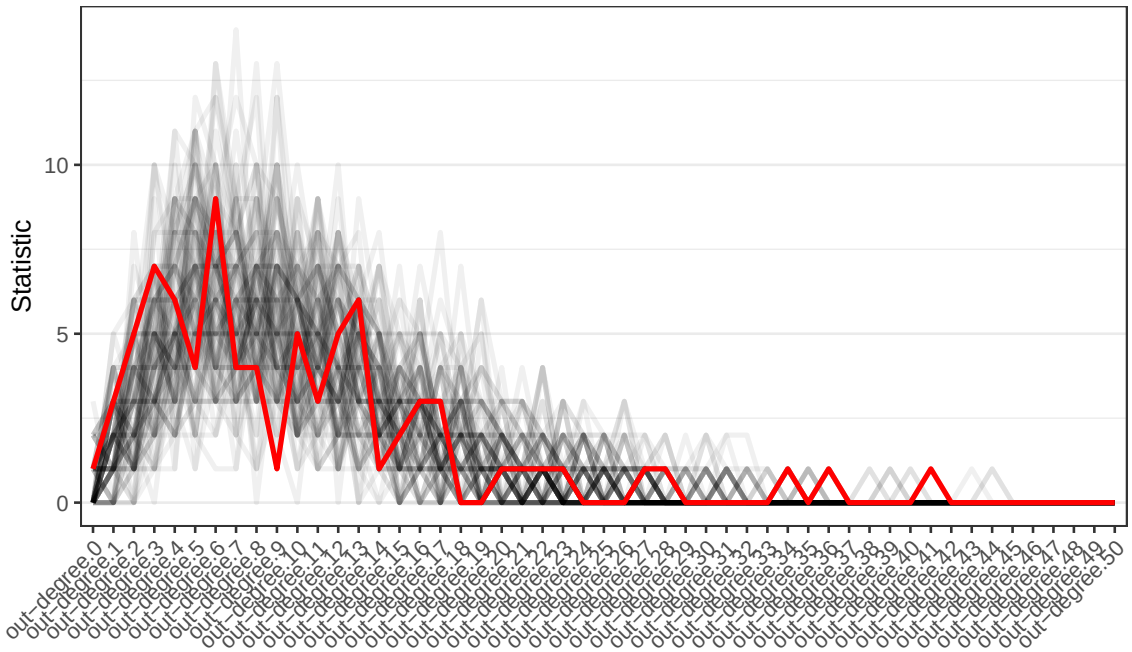
Okay, so let's try adding a `mutual` term to model reciprocity and a `triangles` term for transitivity.

```
fituk <- lolog(ukFaculty ~ edges() + nodeMatch("GroupC") +
            mutual + triangles ,verbose=FALSE)
summary(fituk)
#>                  observed_statistics     theta        se pvalue
#> edges                            817 -3.886303 0.2841351 0.0000
#> nodematch.GroupC                 665  1.943532 0.2530278 0.0000
#> mutual                           240  1.378898 0.5636392 0.0144
```

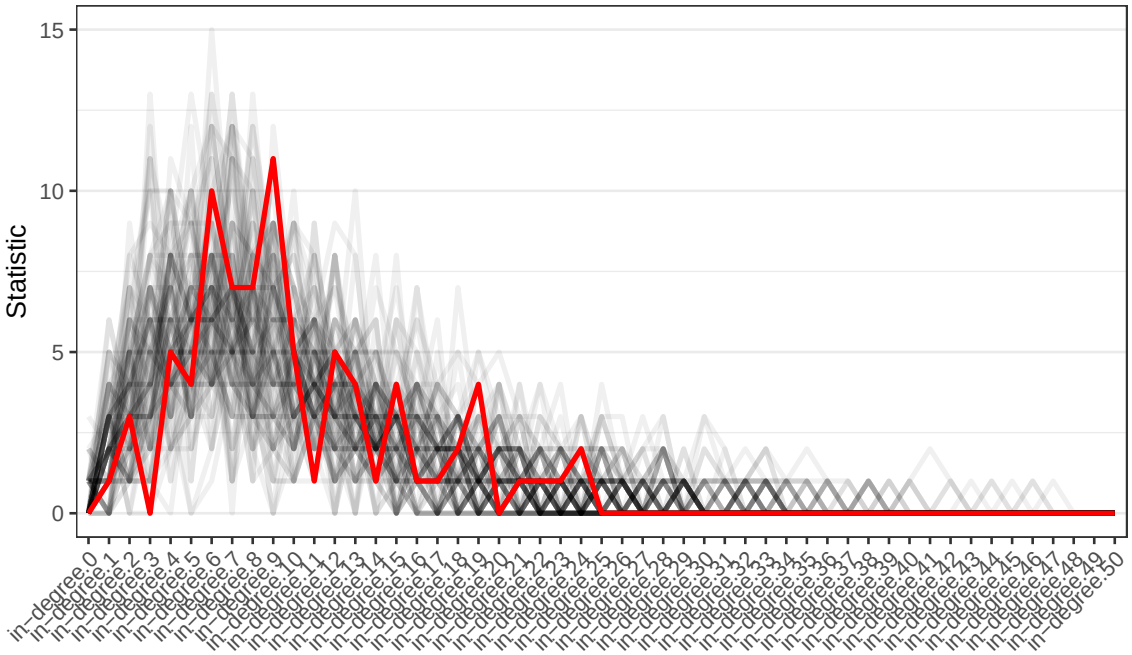
```
#> triangles                    5399  0.420081 0.1344865 0.0018
```

Now let's look at those goodness of fit plots again...

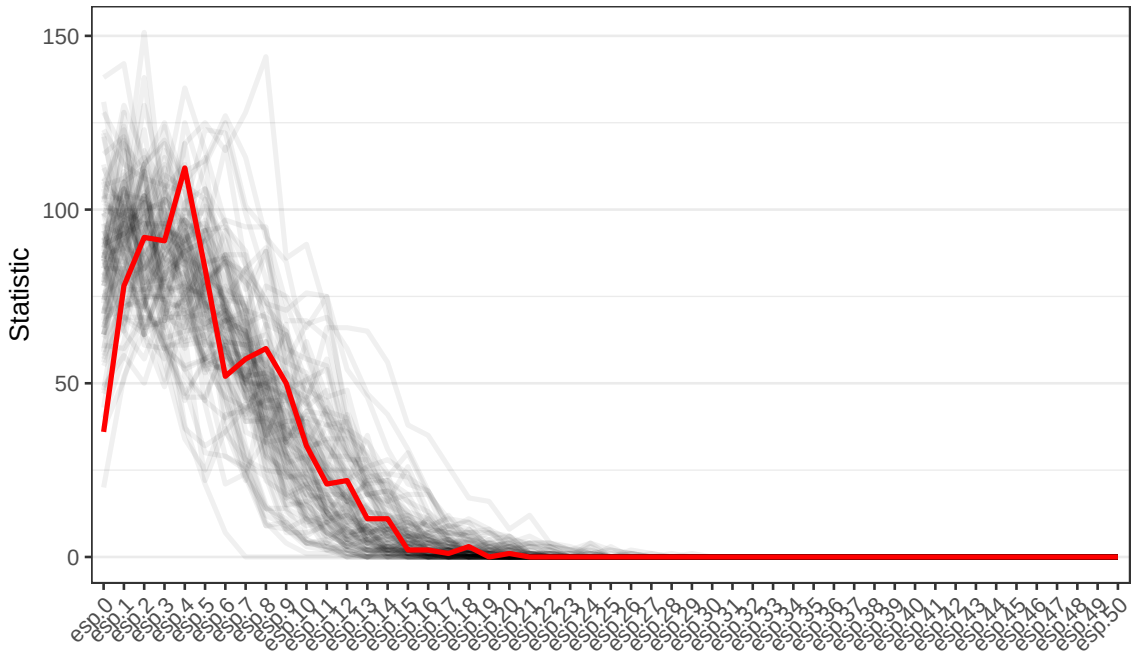
```
g <- gofit(fituk, ukFaculty ~ degree(0:50,"out"))
plot(g)
```




```
g <- gofit(fituk, ukFaculty ~ degree(0:50,"in"))
plot(g)
```

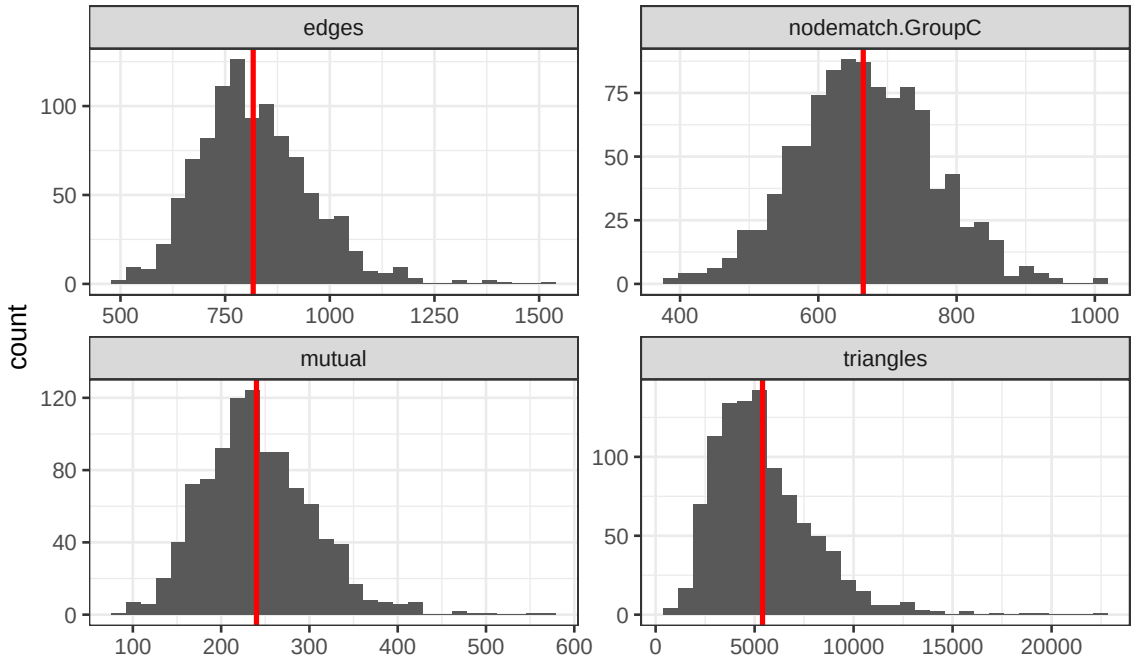



```
g <- gofit(fituk, ukFaculty ~ esp(0:50))
plot(g)
```



These look pretty good, with the observed values falling within the range of values simulated from the model. Additionally, because `lolog` matches the expected model graph statistics with their observed values, we are assured that statistics included in the model will have good goodness of fit. We can see this by plotting the model diagnostics.

```
plot(fituk)
#> `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



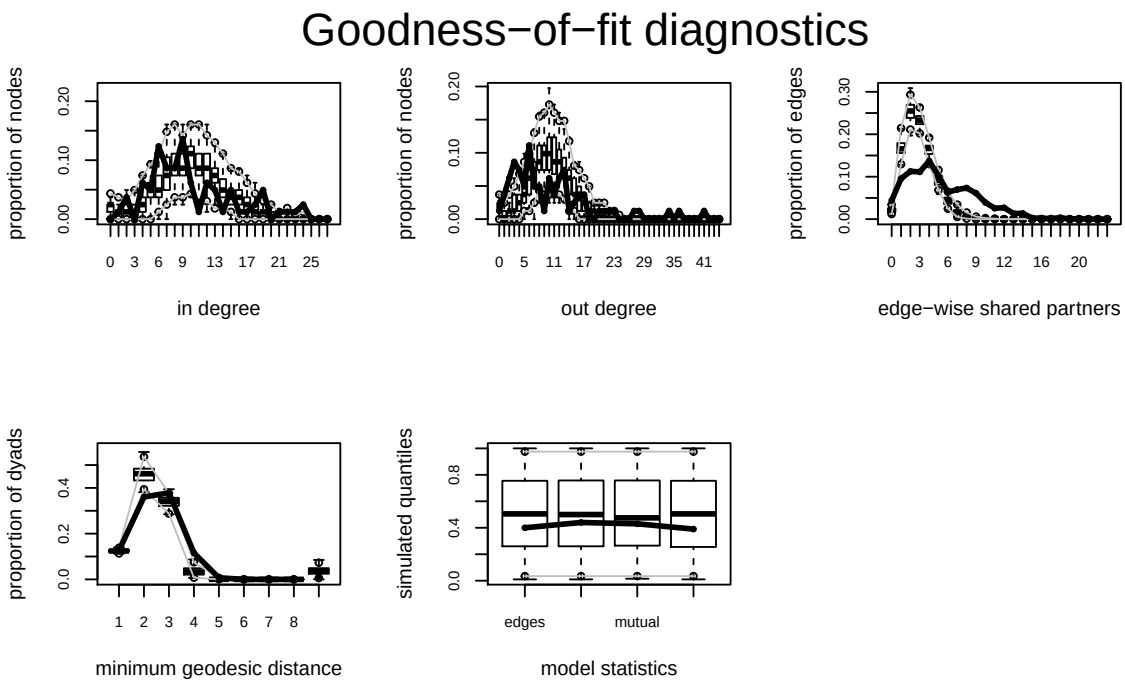
Attempting to fit with ERGM

Like LOLOGs, ERGMs are an incredibly flexible model class. However, they are prone to model degeneracy, so it is recommended that great care is taken in choosing appropriate model statistics to use. Even using best practices in choosing these statistics it is not unusual to be unable to fit a network due to degeneracy related issues.

When modeling transitivity, the best practice for ERGMs is to include a gwesp term, which is "robust" to model degeneracy. We attempted many different models using this term (and others), and the best fitting non-degenerate model that we could find fixed the decay parameter at 0.25. Larger values exhibited degeneracy problems, and allowing the parameter to be fit using curved ERGM estimation failed to converge.

```
fitukErgm <- ergm(ukFaculty ~ edges() + nodematch("GroupC") + mutual +
            gwesp(decay=.25, fixed=TRUE), verbose=FALSE)
#> Warning in is.na(x): is.na() applied to non-(list or vector) of type 'NULL'
#> Starting maximum pseudolikelihood estimation (MPLE):
#> Evaluating the predictor and response matrix.
#> Maximizing the pseudolikelihood.
#> Finished MPLE.
#> Warning in is.na(x): is.na() applied to non-(list or vector) of type 'NULL'
#> Starting Monte Carlo maximum likelihood estimation (MCMLE):
#> Iteration 1 of at most 20:
#> Optimizing with step length 0.259247831836346.
#> The log-likelihood improved by 3.848.
#> Iteration 2 of at most 20:
#> Optimizing with step length 0.302102033559197.
#> The log-likelihood improved by 3.564.
#> Iteration 3 of at most 20:
#> Optimizing with step length 0.526524403995446.
#> The log-likelihood improved by 3.017.
#> Iteration 4 of at most 20:
#> Optimizing with step length 1.
#> The log-likelihood improved by 1.952.
#> Step length converged once. Increasing MCMC sample size.
#> Iteration 5 of at most 20:
#> Optimizing with step length 1.
#> The log-likelihood improved by 0.1366.
#> Step length converged twice. Stopping.
#> Finished MCMLE.
#> Evaluating log-likelihood at the estimate. Using 20 bridges: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
#> This model was fit using MCMC.  To examine model diagnostics and check for degeneracy, use the mcmc.
summary(fitukErgm)
#>
#> ==========================
#> Summary of model fit
#> ==========================
#>
#> Formula:   ukFaculty ~ edges() + nodematch("GroupC") + mutual + gwesp(decay = 0.25,
#>     fixed = TRUE)
#>
#> Iterations:  5 out of 20
#>
#> Monte Carlo MLE Results:
#>                  Estimate Std. Error MCMC % p-value
```

```
#> edges             -5.64266    0.21190       0  <1e-04 ***
#> nodematch.GroupC   1.14803    0.07133       0  <1e-04 ***
#> mutual             2.08049    0.13940       0  <1e-04 ***
#> gwesp.fixed.0.25   1.85190    0.17398       0  <1e-04 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#>      Null Deviance: 8983  on 6480  degrees of freedom
#>  Residual Deviance: 3421  on 6476  degrees of freedom
#>
#> AIC: 3429    BIC: 3457    (Smaller is better.)
g <- gof(fitukErgm)
par(mfrow=c(2,3))
plot(g)
```



The added the gwesp term is highly significant, indicating increased levels of transitivity; however, the goodness of fit plot shows that the ERGM is not capturing the full amount of transitivity in the network. Simulated networks have much lower esp values than the one observed. Unfortunately, using the recommended practices for fitting ERGMs, we were unable to find a non-degenerate ERGM that appropriately captures the degree and transitivity patterns of this dataset.