Possible `ergm` bug involving `ergm.etagrad`

December 17, 2017

The function `ergm.etagrad` of `R` package `ergm` evaluates the gradient of the natural parameter vector. For geometrically weighted model terms, the gradient may be incorrect. The following is based on `R` package `ergm` version 3.8.0 from 2017-08-18, but `R` package `ergm` version 2.0-1 from 2008-01-19 uses the same gradient as `R` package `ergm` version 3.8.0, as downloaded from `https://cran.r-project.org/src/contrib/Archive/ergm/`.

1 The gradient of the natural parameter vector

The natural parameters are defined as

$$\eta_i(\boldsymbol{\theta}) \;=\; \theta_1 \exp(\theta_2)\,(1-(1-\exp(-\theta_2))^i), \quad i = 1, \ldots, n,$$

so that the partial derivatives are given by

$$\begin{aligned}\nabla_{\theta_2}\,\eta_i(\boldsymbol{\theta}) &= \theta_1\,[\exp(\theta_2)\,(1-(1-\exp(-\theta_2))^i) - \exp(\theta_2)\,i\,(1-\exp(-\theta_2))^{i-1}\,\exp(-\theta_2)] \\ &= \theta_1\,[\exp(\theta_2)\,(1-(1-\exp(-\theta_2))^i) - i\,(1-\exp(-\theta_2))^{i-1}]\,.\end{aligned}$$

2 Implementation of gradient in `R` package `ergm`

The `R` script

```
library(ergm)
net <- network.initialize(5)
model <- ergm.getmodel(net ~ gwesp(fixed = FALSE), net)
model$etamap$curved[[1]]$map
model$etamap$curved[[1]]$gradient
```

shows

```
> model$etamap$curved[[1]]$map
function (x, n, ...) {
    i <- 1:n
    x[1] * exp(x[2]) * (1 - (1 - exp(-x[2]))^i)
}
```

and

```
> model$etamap$curved[[1]]$gradient
function (x, n, ...) {
    i <- 1:n
    a <- 1 - exp(-x[2])
    exp(x[2]) * rbind(1 - a^i, x[1] * (1 - a^i - i * a^(i - 1)))
}
```

The implementation of the gradient is equivalent to

$$\nabla_{\theta_2} \eta_i(\boldsymbol{\theta}) \;=\; \theta_1 \left[\exp(\theta_2)\,(1-(1-\exp(-\theta_2))^i) - \mathbf{exp\,(\boldsymbol{\theta_2})}\, i\,(1-\exp(-\theta_2))^{i-1}\right].$$

The boldfaced $\mathbf{exp\,(\boldsymbol{\theta_2})}$ should not be here.

3 The correct gradient

The correct gradient is given by

```
gradient <- function(x, n) {
  i <- 1:n
  a <- 1 - exp(-x[2])
  rbind(exp(x[2]) * (1 - a^i), x[1] * (exp(x[2]) * (1 - a^i) - i * a^(i - 1)))
}
```

4 Example

A simple example is given by

$$\eta_1(\boldsymbol{\theta}) \;=\; \theta_1 \, \exp(\theta_2) \, (1 - (1 - \exp(-\theta_2))) \;=\; \theta_1,$$

which implies that

$$\nabla_{\theta_2} \, \eta_1(\boldsymbol{\theta}) \;=\; 0.$$

`R` package `ergm` reports

```
> model$etamap$curved[[1]]$gradient(c(1,0.5), 3)
           [,1]      [,2]      [,3]
[1,]  1.0000000 1.3934693 1.5482875
[2,] -0.6487213 0.0960268 0.7825317
```

which shows that the $\nabla_{\theta_2} \, \eta_1(\boldsymbol{\theta})$ is evaluated as -0.6487213 rather than 0.

The correct gradient

```
gradient <- function(x, n) {
  i <- 1:n
  a <- 1 - exp(-x[2])
  rbind(exp(x[2]) * (1 - a^i), x[1] * (exp(x[2]) * (1 - a^i) - i * a^(i - 1)))
}
```

reports

```
> gradient(c(1,0.5), 3)
     [,1]      [,2]     [,3]
[1,]    1 1.3934693 1.548287
[2,]    0 0.6065307 1.083833
```

which shows that $\nabla_{\theta_2} \, \eta_1(\boldsymbol{\theta})$ is evaluated as 0, which is indeed correct.

5 Possible implications

If $\theta_2 > 0$, then $\exp(\theta_2) > 1$. Therefore, `R` package `ergm` subtracts from all partial derivatives a term that is too large, because it is multiplied by $\exp(\theta_2) > 1$. An example is $\nabla_{\theta_2} \, \eta_1(\boldsymbol{\theta})$, which

is 0 whereas `R` package `ergm` reports -0.6487213. Hence small, positive partial derivatives can turn into negative partial derivatives, so that the signs of the partial derivatives are wrong and the Monte Carlo maximum likelihood algorithm implemented in `ergm` may venture into the wrong direction. It is possible that this can lead to convergence issues.