ALAAM effects

estimate.alaam{MultivarALAAMalt.R}

Estimate Auto-Logistic Actor Attribute Model (ALAAM)

Description

`estimate.alaam` is used for Bayesian estimation of the Auto-Logistic Actor Attribute (ALAAM). The outcome variable is a binary vector $\boldsymbol{y}$ for vertices $V = \{1, \ldots, n\}$, a matrix for exogenous covariates $\mathbf{W}$, and a binary adjacency matrix $\mathbf{X}$, defined on the network $G(V, E)$, where $E \subset\in V^{(2)}$ or $E \subset \binom{V}{2}$, depending on whether the ties are directed or not, respectively. `estimate.alaam` is a wrapper for the function `BayesALAAM.formula`.

```
estimate.alaam(formula, data, adjacency=NULL,Iterations=1000,
                prevBayes=NULL, recalibrate =FALSE,
                silent=FALSE, PropSigma=NULL, scaling = 1,
                do.scaling = TRUE, initcontagion = NULL,
                burnin = NULL, missingCovs = NULL,
                missingPhi =NULL, priorSigma=NULL, priorMu=NULL,
                scalePrior=NULL, canchange=NULL, MPLE=FALSE,
                saveFreq=NULL, missFreq=100, initTheta=NULL,
                thinning=1, par.burnin=1)
```

Arguments

`formula`	An object of the class `formula` (see `alaam.terms` for details of admissable arguments)
`data`	An object of the class `data.frame` with dimensions $n \times q$.
`adjacency`	An object of the class `matrix`. This matrix needs to be binary, of dimensions $n \times n$, and have a zero diagonal
`Iterations`	The total number of iterations in the MCMC estimation
`prevBayes`	The estimation object from a previous call to `estimate.alaam`. With other arguments as specified by default, the estimation will continue the previous estimation
`recalibrate`	If `prevBayes` is provided, `recalibrate` set to `TRUE` will use the parameter draws in `prevBayes$Thetas` to estimate $\mathbf{\Sigma} = cov(\boldsymbol{\theta} \mid Data)$, and set the proposal variance covariance to $\mathbf{\Sigma} \times (\texttt{scaling}/\sqrt{p})$, if `do.scaling` is `TRUE`
`silent`	If `TRUE`, some printing to console will be supressed

`PropSigma`	If supplied, the proposal variance covariance matrix $\tilde{\boldsymbol{\Sigma}}$ will be $\boldsymbol{\Sigma} \times (\texttt{scaling}/\sqrt{p})$, if `do.scaling` is `TRUE`, and $\boldsymbol{\Sigma}$ if `do.scaling` is `FALSE`. In iteration t, given the current draw $\boldsymbol{\theta}_{t-1}$, a move to $\boldsymbol{\theta}^*$ is proposed from $\boldsymbol{\theta}^* \sim \mathcal{N}_p(\boldsymbol{\theta}_{t-1}, \tilde{\boldsymbol{\Sigma}})$
`scaling`	What scaling constant to use to inflate/deflate the variance-covariance matrix in the proposal of parameters in the MCMC
`do.scaling`	Scaling `PropSigma` or not
`initcontagion`	Initial value of any contagion effects to accelerate convergence (will be depreciated)
`burnin`	To update $\boldsymbol{\theta}_{t-1}$ to $\boldsymbol{\theta}_t$, a draw $\boldsymbol{y}^*$ is made from the likelihood defined by the proposed $\boldsymbol{\theta}^*$. The draw $\boldsymbol{y}^*$ is made using the Metropolis-Hastings (see 4.1 Simulating from the model, Koskinen and Daraganova, 2022) and the number of iterations discarded as burnin is given by `burnin`.
`missingCovs`	*Depreciated*
`missingPhi`	User-defined parameter to determine the amount of missing not at random (MNAR) bias for missing values of $\boldsymbol{y}$ (see 4.3 Missing data, Koskinen and Daraganova, 2022)
`priorSigma`	For use of a prior distribution $\pi(\boldsymbol{\theta})$. Prior distributions are limited to the form $\mathcal{N}_p(\boldsymbol{\mu}, \mathbf{S})$, where `priorSigma` is the $p \times p$ matrix $\mathbf{S}$.
`priorMu`	For use of a prior distribution $\pi(\boldsymbol{\theta})$. Prior distributions are limited to the form $\mathcal{N}_p(\boldsymbol{\mu}, \mathbf{S})$, where `priorMu` is the $p \times 1$ vector $\boldsymbol{\mu}$. If neither `priorSigma` nor `priorMu` provided, an improper prior will be used.
`scalePrior`	A special $\mathcal{N}_p(\boldsymbol{\mu}, \mathbf{S})$ prior, where $\mathbf{S} = \texttt{scalePrior}\mathbf{J}$, where $\mathbf{J}$ is an information matrix under $\boldsymbol{\mu}$ set to the MLE assuming independent observations (see 4.5.1 Prior distributions, Koskinen and Daraganova, 2022).
`canchange`	A Boolean vector of the same length as $\boldsymbol{y}$, indicating whether y_i is fixed, `canchange[i]` set to `FALSE`, or variable `canchange[i]` set to `TRUE`
`MPLE`	Logical, indicating whether MCMC will be forced to evaluate the *pseudo likelihood*, $\tilde{p}(\boldsymbol{y} \mid \boldsymbol{\theta}) = \prod p(y_i \mid \boldsymbol{y}_{-i}\boldsymbol{\theta})$. If no contagion effect is specified, `MPLE` will be set to `TRUE` automatically
`saveFreq`	Every `saveFreq` iteration, the current state of the algorithm will be dumped into `BayesALAAMdump.RData` and if `silent` is `FALSE`, an update will be printed to the console. If not value on `saveFreq` is provided, it will be set to `ceiling(Iterations/10)`

`missFreq`	How often should imputed missing values be stored
`initTheta`	Initial value of $\boldsymbol{\theta}$. This is ignored if `prevBayes` is provided
`thinning`	For iterations $i = 1, \dots, I$, where I is given by `Iterations`, only values $\boldsymbol{\theta}_t$ are stored for $t = q+k, q+2k, q+2k, \dots, q+2M$, where q is `par.burnin` and k is `thinning`. Increasing k will reduce the autocorrelation of the MCMC chains
`par.burnin`	See `thinning`

Details

The algorithm implements the algorithms of Koskinen and Daraganova (2022). A typical model has the form `response` $\sim$ `terms` where `response` is the numeric, binary response vector and terms is a series of admissible terms which specifies a linear predictor for response. A terms specification of the form `first + second` indicates all the terms in first together with all the terms in second with duplicates removed. The specification `first*second` indicates the cross (an interaction) of first and second.
The formula has an implied intercept term and will add a vector of constants (1). The intercept cannot be removed.
The function `estimate.alaam` calls the lower level function `BayesALAAM.formula` which, itself, supplants the earlier function `BayesALAAM`.
Formula terms, on the right-hand side of $\sim$, can be either (column) names of `data`, admissible network metrics based on `adjacency`, or contagion terms. For details see `alaam.terms`.

Value

`estimate.alaam` returns an 'object' of class `estimate.alaam.obj`. The functions `write.res.table`, `plotPost`, `get.gof.distribution`, `post.deviance.alaam`, and more, are defined on 'objects' of class `estimate.alaam.obj`.
An 'object' of class `estimate.alaam.obj` returns the following

`Thetas`	A $M \times p$ matrix of posterior draws from the MCMC. Note that even if `contagion` is specified as `'none'`, the independent model will have a column of zeros in `Thetas` for the `'simple'` contagion
`ALAAMobj`	An 'object' of class `alaam.obj`. This is a version of data formatted for the specific model specified.
`ResTab`	A table summarising the posteriors.
`priorSigma`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`priorMu`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`imputed.obs`	A matrix with imputed $\boldsymbol{y}$ if the original data had any missing values.
`silent`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`do.scaling`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`PropSigma`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`MPLE`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`saveFreq`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`missFreq`	As defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`initTheta`	The last update in the MCMC to be used as a defined in call to `estimate.alaam`, for use in further calls to `estimate.alaam` with `prevBayes`
`acc.ratio`	The proportion of proposals $\boldsymbol{\theta}^*$ that were accepted.

Examples

```
# mock data set
n <- 10
my.dat <- data.frame(agree=runif(n)>.5,
                     mood=rnorm(n),
                     sex=runif(n)>.5)
require('sna')
adj <- rgraph(n,1)

# Independent model:
res.ind <- estimate.alaam(agree ~odegree+mood+sex+mood*sex,
                                  my.dat,
                                  adjacency=adj,
                                  Iterations=1000)
```

```
# Posterior summary:
plotPost(res.ind,showplot=TRUE)

# Simple contagion:
res.cont <- estimate.alaam(agree ~odegree+mood+sex+mood*sex+simple,
                                  my.dat,
                                  adjacency=adj,
                                  Iterations=1000)
# Posterior summary:
plotPost(res.cont,showplot=TRUE)

# Restart estimation with previous results
# and some thinnning and burnin (takes a little longer)
 res.cont.2 <- estimate.alaam(agree ~odegree+mood+sex+mood*sex+simple,
                                  my.dat,
                                  adjacency=adj,
                                  Iterations=5000,
                                  prevBayes = res.cont,
                                  recalibrate = TRUE,
                                  par.burnin=50,
                                  thinning = 5)
```

alaam.terms{MultivarALAAMalt.R}

Effects for ALAAM

Description

This entry describes and lists the formula objects that are permissible in a call to `estimate.alaam`.

Specifying models

The social influence model developed by Robins et al. (2001) and later elaborated by Daraganova (2009) and Daraganova and Robins (2013) and now referred to as the autologistic actor-attribute model (ALAAM), is a model for binary nodal attributes $\boldsymbol{y} = \{Y_i : 1 \leq i \leq n\}$, conditional on a network adjacency matrix $\mathbf{X} = \{X_{ij} : (i,j) \in V \times V\}$, and a matrix of covariates $\mathbf{W}$

$$p_{\boldsymbol{\theta}}(\boldsymbol{y}|\mathbf{X},\mathbf{W}) = \exp\left\{\boldsymbol{\theta}^{\top} z(\boldsymbol{y},\mathbf{X},\mathbf{W}) - \psi(\boldsymbol{\theta};\mathbf{X},\mathbf{W})\right\},$$

for $\boldsymbol{y} \in \mathcal{Y} = \{0,1\}^V$. Here $z(\boldsymbol{y},\mathbf{X},\mathbf{W})$ is a $p \times 1$ vector of statistics calculated for the the dependent variable $\boldsymbol{y}$, the network $\mathbf{X}$, and covariates $\mathbf{W}$. The vector of statistics is defined in three main ways

Covariate effects

Covariate effects are specified in standard `lm` syntax. For example, the term

$$\sum_{i=1}^{n} y_i w_{ij}$$

where the name of column j, for example `mood`, is used in standard fashion `+mood` on the right-hand side. For a model with variables `agree`, `mood`, and `sex`, where `agree` is the dependent variable,

```
agree ~ mood+sex
```

would specify the model defined by

$$z(\boldsymbol{y}, \mathbf{X}, \mathbf{W}) = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} y_i w_{i,mood} \\ \sum_{i=1}^{n} y_i w_{i,sex} \end{bmatrix}$$

To define and add an interaction effect between `mood` and `sex`,

```
agree ~ mood+sex+mood*sex
```

which would specify the model defined by

$$z(\boldsymbol{y}, \mathbf{X}, \mathbf{W}) = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} y_i w_{i,mood} \\ \sum_{i=1}^{n} y_i w_{i,sex} \\ \sum_{i=1}^{n} y_i w_{i,mood} w_{i,sex} \end{bmatrix}$$

Markov effects

If we allow outcome y_i to depend on tie-variables x_{uv}, for which $\{i\} \cap \{u, v\} \neq \emptyset$, the vector of statistics admit as statistics a number of network metrics. In the sequel, we refer to outcomes $y_i = 1$ as a success, and $y_i = 0$ as a failure. These effects are all of the dyadic form

$$z_u(\boldsymbol{y}, \mathbf{X}, \mathbf{W}) = \sum_{i=1}^{n} y_i u_i$$

In previous versions of the program, these statistics u_i were pre-calculated and added to $\mathbf{W}$. For `estimate.alaam`, these are calculated internally if requested.

Effect name	Formula u_i	Interpreation
`degree`	$y_i x_{i\cdot} = y_i \sum_j x_{ij}$	For undirected networks, this measures the association of degree centrality and the probability of success
`idegree`	$y_i x_{\cdot i} = y_i \sum_j x_{ji}$	For directed networks, this measures the association of in-degree centrality and the probability of success
`odegree`	$y_i x_{i\cdot} = y_i \sum_j x_{ij}$	For directed networks, this measures the association of out-degree centrality and the probability of success
`recipties`	$y_i \sum_j x_{ij} x_{ji}$	For directed networks, this measures the association of out-degree centrality and the probability of success
`twostar`	$y_i \binom{x_{i\cdot}}{2}$	For undirected networks, the effect of centrality over and above degree
`intwostar`	$y_i \binom{x_{\cdot i}}{2}$	For directed networks, the effect of indegree centrality over and above indegree
`outtwostar`	$y_i \binom{x_{i\cdot}}{2}$	For directed networks, the effect of outdegree centrality over and above outdegree
`threestars`	$y_i \binom{x_{i\cdot}}{3}$	For undirected networks, the effect of degree centrality over and above twostars
`twopath`	$y_i (x_{\cdot i} x_{i\cdot} - \sum_j x_{ij} x_{ji})$	For directed networks, the association of brokerage on the probability of success.
`inthreestar`	$y_i \binom{x_{\cdot i}}{3}$	For directed networks, the effect of indegree centrality over and above intwostars
`outthreestar`	$y_i \binom{x_{i\cdot}}{3}$	For directed networks, the effect of outdegree centrality over and above outtwostars
`transties`	$y_i \sum_{j,k \neq i} x_{ij} x_{ik} x_{jk}$	For (directed) undirected networks, the effect on probability of success of being embedded in (transitive) triads
`indirties`	$y_i \sum_j x_{ij} \sum_k (1 - x_{ik}) x_{jk}$	For (directed) undirected networks, the effect on probability of success of having ties to people that have ties to many people you are not directly tied to (see 3.1.3 Indirect network and contagion dependencies, Koskinen and Daraganova, for details)

Contagion effects

Daraganova (2009) proposed a number of dependence assumptions for $\mathcal{Y} \times \mathcal{X}$ and derived the associated statistics (see Koskinen and Daraganova, 2022, for details and proofs). If $z(\boldsymbol{y}, \mathbf{X}, \mathbf{W})$ contain these terms, the responses of y_i and y_j may be conditionally dependent, conditional on $\mathbf{X}$ and $\boldsymbol{y}_{-ij} = \{y_k : k \neq i, j\}$, and consequently

$$p_{\boldsymbol{\theta}}(\boldsymbol{y}|\mathbf{X},\mathbf{W}) \neq \prod_{i=1}^{n} \Pr(Y_i = y_i \mid \boldsymbol{\theta},\mathbf{X},\mathbf{W}).$$

The most basic form of dependence, *direct contagion* (see 3.1.2 Network contagion dependence, Koskinen and Daraganova, 2022), yields a statistic z_{DC}, such that

$$\Pr(Y_i = \mid \boldsymbol{y}_{-i},\mathbf{X},\boldsymbol{\theta}) = \frac{\exp\{\theta_{DC} z_{DC}(\Delta^{+}\boldsymbol{y},\mathbf{X})\}}{e^{\theta_{DC} z_{DC}(\Delta^{-}\boldsymbol{y},\mathbf{X})} + e^{\theta_{DC} z_{DC}(\Delta^{+}\boldsymbol{y},\mathbf{X})}} e^{\boldsymbol{\theta}^{\top}_{-DC}(z_{-DC}(\Delta^{-}\boldsymbol{y},\mathbf{X},\mathbf{W}) - z_{-DC}(\Delta^{+}\boldsymbol{y},\mathbf{X},\mathbf{W}))}$$

where $\Delta^{+}\boldsymbol{y}$ is the vector $\boldsymbol{y}$ with element i set to 1, and $\Delta^{-}\boldsymbol{y}$ is the vector $\boldsymbol{y}$ with element i set to 0, and

$$z_{DC}(\boldsymbol{y},\mathbf{X}) = \sum_{i,j} y_i y_j x_{ij}.$$

Currently, the contagion effects are

Effect name	Formula u_i	Interpreation
`simple`	$\sum_{i,j} y_i y_j x_{ij}$	is the probability of success increased by being connected to actors whose outcome is a success
`recip`	$\sum_{i,j} y_i y_j x_{ij} x_{ji}$	is the probability of success increased by being *mutually* tied to actors whose outcome is a success (directed networks only)
`indirect`	$\sum_i y_i \sum_j x_{ij} \sum_{k \neq i,j} y_k x_{jk}$	is the probability of success increased by being indirectly connected to actors whose outcome is a success (see 3.1.3 Indirect network and contagion dependencies, Koskinen and Daraganova, for details))
`closedind`	$\sum_i y_i \sum_j x_{ij} \sum_{k \neq i,j} y_k x_{ik} x_{jk}$	is the probability of success increased by being both indirectly and directly connected to actors whose outcome is a success (see 3.1.3 Indirect network and contagion dependencies as well as supplementary material, Koskinen and Daraganova, for details))
`transitive`	$\sum_i y_i \sum_j x_{ij} y_j \sum_{k \neq i,j} y_k x_{ik} x_{jk}$	is the probability of success increased by being embedded in triads where the two other members have success on the outcome(see 3.1.3 Indirect network and contagion dependencies as well as supplementary material, Koskinen and Daraganova, for details))

Interaction of contagion effects and attributes

If you assume that the dependence, represented by a contagion statistic and parameter, is stronger (weaker), depending on a particular monadic covariate w_{ik} for the focal node i, the contagion effect can be interacted with w_{ik}. Interacting, for example, the (binary) variable `sex` with the direct contagion, the contributions will be

$$\theta_{DC} \sum_{i,j} y_i y_j x_{ij} + \theta_{DC,sex} \sum_{i,j} y_i w_{i,sex} y_j x_{ij}.$$

For $w_{i,sex} = 0$, the contagion parameter is thus

$$\theta_{DC},$$

and for $w_{i,sex} = 1$, the contagion parameter is

$$\theta_{DC} + \theta_{DC,sex}.$$

This can be used to test, for example, if males are more susceptible to social influence than females. An interaction of a covariate with a contagion effect is done using the standard interaction syntax, e.g.

```
agree ~ mood+sex+simple+sex*simple
```

Restrictions

- There is no restriction on the number of covariates that can be interacted with a contagion statistic
- Covariate interactions can only be defined for one contagion statistic at a time
- If a covariate interaction with a contagion statistic is defined, the two main effects must be included

---

`post.deviance.alaam`{`MultivarALAAMalt.R`}

Calculate posterior deviance from Bayesian ALAAM

Description

`post.deviance.alaam` calculates the posterior distribution of the deviance

$$D(\boldsymbol{\theta}) = -2\ell(\boldsymbol{\theta}; Data)$$

for posterior draws $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_M$, obtained from `estimate.alaam`. This function calls the lower order function `eval.like.path.alaam` that estimates log ratio $\psi(\boldsymbol{\theta}^*) - \psi(\boldsymbol{\theta})$ of normalising constants, for each value $\boldsymbol{\theta}$ in the posterior `ALAAMresult$Thetas`, thinned so that that the number of draws is `num.outs`.

```
post.deviance.alaam(ALAAMresult, numBridges=20, thinning.like = 5000,
                    sample.size = 200, cov.sample.burnin = NULL,
                    printFreq=10, mult.fact = 30,
                    num.outs=100)
```

Arguments

`ALAAMresult`	The estimation object from a previous call to `estimate.alaam`
`numBridges`	The number K of bridges used in the call to `eval.like.path.alaam`, which uses the 'path sampler' to calculate the log ratio $\psi(\boldsymbol{\theta}^*) - \psi(\boldsymbol{\theta}$ of normalising constants
`thinning.like`	Passed to `eval.like.path.alaam`, where a sample $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M$ is drawn from $\boldsymbol{y} \sim p(\boldsymbol{y} \mid \boldsymbol{\theta}^{(j)})$, for $j = 0, \ldots, K$, and the thinning k determines how many iterations are discarded between each sample point. Too small values mean that the sample points will be highly correlated.
`sample.size`	Passed to `eval.like.path.alaam`, where a sample $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M$ is drawn from $\boldsymbol{y} \sim p(\boldsymbol{y} \mid \boldsymbol{\theta}^{(j)})$, for $j = 0, \ldots, K$, and the thinning k determines how many iterations are discarded between each sample point. Too small values mean that the sample points will be highly correlated.
`cov.sample.burnin`	Passed to `eval.like.path.alaam`. The number of sample points that are discarded as burn in when drawing from the likelihood $\boldsymbol{y}$
`printFreqn`	The print frequency
`mult.fact`	A constant to set the thinning according to the formula of Snijders (2002)(depreciated)
`num.outs`	The number of posterior draws to evaluate $D(\boldsymbol{\theta})$ in

Details

Value

The function `post.deviance.alaam` returns the approximate value of the deviance $D(\boldsymbol{\theta})$, for each $\boldsymbol{\theta}^{(b)}, \boldsymbol{\theta}^{(b+k)}, \boldsymbol{\theta}^{(b+2k)}, \ldots, \boldsymbol{\theta}^{(M)}$, in the thinned sample with burnin iterations discarded `ALAAMresult$Thetas`. The output, `post.devs`, can be used to compare models graphically using `plot.deviance.alaam` (Aitkin et al., 2017), or by calculating the DIC with `alaam.dic`.

`post.devs`	An $M \times 1$, where M is given by `num.outs`, matrix of posterior draws from the distribution $D(\boldsymbol{\theta})$.

`alaam.dic`{`MultivarALAAMalt.R`}

Calculate Deviance Information Criterion from Bayesian ALAAM

Description

`alaam.dic` calculates the posterior distribution of the deviance

$$DIC = E[D(\boldsymbol{\theta})] + V(D(\boldsymbol{\theta}))/2$$

for posterior draws $D(\boldsymbol{\theta}_1), D(\boldsymbol{\theta}_2), \ldots, D(\boldsymbol{\theta}_M)$ obtained from `post.deviance.alaam`. Models with *smaller* DIC should be preferred to models with *larger* DIC.

```
alaam.dic(Post.dev)
```

Arguments

`Post.dev`	The object returned by a call to `post.deviance.alaam`

References

Aitkin, M. , Vu, D. & Francis, B. (2017) Statistical modelling of a terrorist network. Journal of the Royal Statistical Association (Series A), 180, 751–768.

Daraganova, Galina. 2009. Statistical Models for Social Networks and Network-Mediated Social Influence Processes: Theory and Applications. University of Melbourne.

Daraganova, Galina, and Garry Robins. 2013. 'Autologistic Actor Attribute Models.' In Exponential Random Graph Models for Social Networks: Theory, Methods and Applications, edited by Lusher D., Koskinen J., and G. Robins, 102–14. NY: Cambridge University Press New York.

Koskinen, Johan, and Galina Daraganova. 2022. 'Bayesian Analysis of Social Influence.' Journal of the Royal Statistical Society Series A: Statistics in Society 185.4, pp. 1855–1881.

Robins, Garry, Philippa Pattison, and Peter Elliott. 2001. 'Network Models for Social Influence Processes.' Psychometrika 66 (2). Springer: 161–89.

Snijders, T.A.B. (2002) Markov chain Monte Carlo estimation of exponential random graph models. Journal of Social Structure, 3(2), 1–40.

Version: Tuesday 29$^{\text{th}}$ October, 2024 23:02