

# Package: sclr (via r-universe)

October 31, 2024

**Title** Scaled Logistic Regression

**Version** 0.3.1

**Description** Maximum likelihood estimation of the scaled logit model parameters proposed in Dunning (2006) <doi:10.1002/sim.2282>.

**License** MIT + file LICENSE

**URL** <https://khvorov45.github.io/sclr/>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.0.2

**Depends** R (>= 3.6.0)

**Imports** broom, tibble, dplyr, rlang, stats, purrr

**Suggests** knitr, rmarkdown, testthat (>= 2.1.0)

**VignetteBuilder** knitr

**Repository** <https://khvorov45.r-universe.dev>

**RemoteUrl** <https://github.com/khvorov45/sclr>

**RemoteRef** HEAD

**RemoteSha** 6eafc54f0acc4a4db9036ffa86638f672da071d7

## Contents

check_baseline . . . . .	2
coef.sclr . . . . .	3
find_prot_titre_val . . . . .	3
get_protection_level . . . . .	4
new_sclr . . . . .	5
one_titre_data . . . . .	6
predict.sclr . . . . .	6
print.sclr . . . . .	7
sclr . . . . .	8
sclr_fit . . . . .	10

sclr_ideal_data . . . . .	11
sclr_log_likelihood . . . . .	12
tidy.sclr . . . . .	12
two_titre_data . . . . .	13

<b>Index</b>	<b>15</b>
--------------	-----------

---

check_baseline	<i>Check for baseline boundary</i>
----------------	------------------------------------

---

## Description

Fits the scaled logit model as well as logistic regression. Does a likelihood ratio test.

## Usage

```
check_baseline(
  formula = NULL,
  data = NULL,
  fit_sclr = NULL,
  fit_lr = NULL,
  conf_lvl = 0.95,
  verbose = TRUE
)
```

## Arguments

formula	Formula to use for model fitting.
data	Optional dataframe.
fit_sclr	Fit object returned by <code>sclr</code> .
fit_lr	Fit object returned by <code>glm</code> .
conf_lvl	Confidence level for the test
verbose	Whether to print message based on test result.

## Value

A `tibble` with a summary.

## Examples

```
library(sclr)
l1 <- sclr_ideal_data(n = 50, theta = 1e6, seed = 20191104)
check_baseline(status ~ logHI, l1)
```

---

coef.sclr	<i>ML estimate components</i>
-----------	-------------------------------

---

### Description

coef returns MLE's. vcov returns the estimated variance-covariance matrix at MLE's. confint returns the confidence interval. model.matrix returns the model matrix (x). model.frame returns the model frame (x and y in one matrix).

### Usage

```
## S3 method for class 'sclr'
coef(object, ...)

## S3 method for class 'sclr'
vcov(object, ...)

## S3 method for class 'sclr'
confint(object, parm, level = 0.95, ...)

## S3 method for class 'sclr'
model.matrix(object, ...)

## S3 method for class 'sclr'
model.frame(formula, ...)

## S3 method for class 'sclr'
logLik(object, ...)
```

### Arguments

object, formula	An object returned by <a href="#">sclr</a> .
...	Not used. Needed to match generic signature.
parm	Parameter name, if missing, all parameters are considered.
level	Confidence level.

---

find_prot_titre_val	<i>Search function for scaled logit protection covariate levels</i>
---------------------	---

---

### Description

The search engine behind [get\\_protection\\_level](#). Should not usually be necessary to call this directly.

**Usage**

```
find_prot_titre_val(
  fit,
  var_name,
  newdata = NULL,
  prot_var_name = "prot_point",
  lvl = 0.5,
  ci_level = 0.95,
  tol = 10^(-7)
)
```

**Arguments**

fit	Object returned by <code>sclr</code> .
var_name	Name of the covariate for which the protection values should be calculated. This name should appear in the formula of the call to <code>sclr</code> which was used to generate fit.
newdata	A dataframe with all covariates except the one for which protection values should be calculated.
prot_var_name	A variable name among those returned by <code>predict.sclr</code> which needs to equal lvl at the value of var_name that is supposed to be found.
lvl	Protection level to find titre values for. Default is 0.5 (50%).
ci_level	Confidence level for the calculated interval. Default is 0.95.
tol	Tolerance. The values will be found numerically, once the algorithm converges within tol of lvl it stops looking. Default is $10^{-7}$ .

**Value**

A dataframe. Will have the same variables as newdata with the addition of the var\_name variable.

---

get\_protection\_level    *Protection level calculations*

---

**Description**

Calculates covariate values corresponding to a particular protection level. Only accepts one covariate at a time, fixed values of all the others should be provided. The search engine is `find_prot_titre_val`.

**Usage**

```
get_protection_level(
  fit,
  var_name,
  newdata = NULL,
  lvl = 0.5,
```

```

  ci_level = 0.95,
  tol = 10^(-7)
)
```

### Arguments

fit	Object returned by <code>sclr</code> .
var_name	Name of the covariate for which to find values corresponding to a protection level. This name should appear in the formula in the call to <code>sclr</code> which was used to generate fit.
newdata	A dataframe with all covariates except the one for which protection values should be calculated. If there is only one covariate, can be left as NULL (the default)
lvl	Protection level to find covariate values for. Default is 0.5 (50%)
ci_level	Confidence level for the calculated interval. Default is 0.95.
tol	Tolerance. The values will be found numerically, once the algorithm converges within tol of lvl it stops looking. Default is $10^{-7}$ .

### Value

A [tibble](#). Will have the same variables as newdata with the addition of the var\_name variable.

---

new_sclr	<i>Create a new sclr object</i>
----------	---------------------------------

---

### Description

new\_sclr creates the object `sclr` returns. is\_sclr checks if the object is of class sclr.

### Usage

```
new_sclr(fit, x, y, cl, mf, mt)
```

```
is_sclr(fit)
```

### Arguments

fit	A list returned by <code>sclr_fit</code> .
x	Model matrix.
y	Model response.
cl	Call.
mf	Model frame.
mt	Model terms.

### Value

sclr object

---

one\_titre\_data                      *Simulated one-titre antibody data*

---

### Description

A simulated dataset containing 5000 independent observations on antibody titres and the corresponding infection status. The data was simulated to resemble real influenza infection and haemagglutinin titre data.

### Usage

```
one_titre_data
```

### Format

A data frame with 5000 observations and 2 variables:

**logHI** haemagglutinin-inhibiting (HI) titre. True simulated titre on a log scale.

**status** influenza infection status. 1 - infected. 0 - not infected

### Model

The model behind the simulation was

$$\lambda * (1 - f(\beta_0 + \beta_1 * HI))$$

Where

- $f$  - Inverse logit function
- $\lambda = 0.5$
- $\beta_0 = -5$
- $\beta_1 = 2$

---

predict.sclr                      *Predict method for scaled logit model x.*

---

### Description

Returns only the protection estimates. The only supported interval is a confidence interval (i.e. the interval for the estimated expected value).

### Usage

```
## S3 method for class 'sclr'
predict(object, newdata, ci_lvl = 0.95, ...)
```

**Arguments**

object	Object returned by <code>sclr</code> .
newdata	A dataframe with all covariates. Names should be as they appear in the formula in the call to <code>sclr</code> .
ci_lvl	Confidence level for the calculated interval.
...	Not used. Needed to match generic signature.

**Details**

The model is

$$P(Y = 1) = \lambda(1 - \text{logit}^{-1}(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k))$$

Where  $Y$  is the binary outcome indicator, (e.g. 1 - infected, 0 - not infected).  $X$  - covariate.  $k$  - number of covariates. This function calculates

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k$$

transformations at the covariate values found in `newdata` as well as the variance-covariance matrices of those transformations. This is used to calculate the confidence intervals at the given parameter values. The inverse logit transformation is then applied to point estimates and interval bounds.

**Value**

A `tibble` obtained by adding the following columns to `newdata`:

prot_point_lin	prot_l_lin	prot_u_lin	
			Point estimate, low and high bounds of the linear transformation.
prot_sd_lin			Estimated standard deviation of the linear transformation.
prot_point	prot_l	prot_u	
			Inverse logit-transformed point estimate, low and high bounds of the linear transformation.

---

print.sclr	<i>Print a sclr object.</i>
------------	-----------------------------

---

**Description**

Summarises a `sclr` object for printing. For a dataframe summary, see `tidy`.

**Usage**

```
## S3 method for class 'sclr'
print(x, level = 0.95, ...)

## S3 method for class 'sclr'
summary(object, level = 0.95, ...)
```

**Arguments**

x, object	An object returned by <code>sclr</code> .
level	Confidence level for the intervals.
...	Not used. Needed to match generic signature.

---

sclr	<i>Fits the scaled logit model</i>
------	------------------------------------

---

**Description**

Used to fit the scaled logit model from Dunning (2006).

**Usage**

```
sclr(
  formula,
  data = NULL,
  ci_lvl = 0.95,
  tol = 10(-7),
  algorithm = c("newton-raphson", "gradient-ascent"),
  nr_iter = 2000,
  ga_iter = 2000,
  n_conv = 3,
  conventional_names = FALSE,
  seed = NULL
)
```

**Arguments**

formula	an object of class "formula": a symbolic description of the model to be fitted.
data	a data frame.
ci_lvl	Confidence interval level for the parameter estimates.
tol	Tolerance.
algorithm	Algorithms to run. "newton-raphson" or "gradient-ascent". If a character vector, the algorithms will be applied in the order they are present in the vector.
nr_iter	Maximum allowed iterations for Newton-Raphson.
ga_iter	Maximum allowed iterations for gradient ascent.
n_conv	Number of times the algorithm has to converge (to work around local maxima).
conventional_names	If TRUE, estimated parameter names will be (Baseline), (Intercept) and the column names in the model matrix. Otherwise - lambda, beta_0 and beta_ prefix in front of column names in the model matrix.
seed	Seed for the algorithms.



## Details

The model is logistic regression with an added parameter for the top asymptote. That parameter is reported as `theta` (or `(Baseline)` if `conventional_names = TRUE`). Note that it is reported on the logit scale. See `vignette("sclr-math")` for model specification, log-likelihood, scores and second derivatives. The main default optimisation algorithm is Newton-Raphson. Gradient ascent is used as a fallback by default. Computing engine behind the fitting is `sclr_fit`.

## Value

An object of class `sclr`. This is a list with the following elements:

<code>parameters</code>	Maximum likelihood estimates of the parameter values.
<code>covariance_mat</code>	The variance-covariance matrix of the parameter estimates.
<code>algorithm</code>	Algorithm used.
<code>algorithm_return</code>	Everything the algorithm returned.
<code>n_converge</code>	The number of Newton-Raphson iterations (including resets) that were required for convergence.
<code>x</code>	Model matrix derived from formula and data.
<code>y</code>	Response matrix derived from formula and data.
<code>call</code>	The original call to <code>sclr</code> .
<code>model</code>	Model frame object derived from formula and data.
<code>terms</code>	Terms object derived from model frame.
<code>ci</code>	Confidence intervals of the parameter estimates.
<code>log_likelihood</code>	Value of log-likelihood calculated at the ML estimates of parameters.
<code>formula</code>	Passed formula.
<code>data</code>	Passed data.

Methods supported: `print`, `vcov`, `coef`, `model.frame`, `model.matrix`, `summary`, `predict`, `tidy` (`broom` package), `logLik`.

## References

Dunning AJ (2006). "A model for immunological correlates of protection." *Statistics in Medicine*, 25(9), 1485-1497. <https://doi.org/10.1002/sim.2282>.

## Examples

```
library(sclr)
fit1 <- sclr(status ~ logHI, one_titre_data)
summary(fit1)
```

---

sclr\_fit

*Fitter function for the scaled logit model*


---

### Description

Computing engine behind [sclr](#).

### Usage

```
sclr_fit(
  y,
  x,
  tol = 10-7,
  algorithm = c("newton-raphson", "gradient-ascent"),
  nr_iter = 2000,
  ga_iter = 2000,
  n_conv = 3,
  conventional_names = FALSE,
  seed = NULL
)
```

### Arguments

y	A vector of observations.
x	A design matrix.
tol	Tolerance.
algorithm	Algorithms to run. "newton-raphson" or "gradient-ascent". If a character vector, the algorithms will be applied in the order they are present in the vector.
nr_iter	Maximum allowed iterations for Newton-Raphson.
ga_iter	Maximum allowed iterations for gradient ascent.
n_conv	Number of times the algorithm has to converge (to work around local maxima).
conventional_names	If TRUE, estimated parameter names will be (Baseline), (Intercept) and the column names in the model matrix. Otherwise - lambda, beta_0 and beta_ prefix in front of column names in the model matrix.
seed	Seed for the algorithms.

### Details

The likelihood maximisation can use the Newton-Raphson or the gradient ascent algorithms.

---

sclr\_ideal\_data      *Generate ideal data for the scaled logit model*

---

## Description

Allows variation of all parameters and the creation of an arbitrary number of covariates.

## Usage

```
sclr_ideal_data(  
  n = 1000,  
  theta = 0,  
  beta_0 = -5,  
  covariate_list = list(logHI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par =  
    2)),  
  outcome_name = "status",  
  seed = NULL,  
  attach_true_vals = FALSE,  
  attach_seed = FALSE  
)
```

## Arguments

n	Number of observations.
theta	Baseline risk parameter on the logit scale.
beta_0	Intercept of the linear part.
covariate_list	A list in the form of name = list(gen_fun, true_par) where gen_fun is a function that takes n as an argument and returns a vector of observations, true_par is the true parameter value of that covariate. See examples.
outcome_name	Name to give to the outcome
seed	Seed to set. If NULL, no seed will be set.
attach_true_vals, attach_seed	Whether to attach additional attributes.

## Value

A [tibble](#).

## Examples

```
# One titre  
one_titre <- sclr_ideal_data(  
  covariate_list = list(  
    logHI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par = 2)  
  )  
)
```

```
sclr(status ~ logHI, one_titre) # Verify

# Two titres
two_titre <- sclr_ideal_data(
  covariate_list = list(
    logHI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par = 2),
    logNI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par = 1)
  )
)
sclr(status ~ logHI + logNI, two_titre) # Verify
```

---

sclr\_log\_likelihood    *Log-likelihood*

---

### Description

Computes the log-likelihood of the scaled logit model at a given set of parameter estimates (or the MLE if pars is not supplied). Either fit or x, y and pars need to be supplied.

### Usage

```
sclr_log_likelihood(fit = NULL, x = NULL, y = NULL, pars = NULL)
```

### Arguments

fit	An object returned by <code>sclr</code> . Or a list with parameters, x and y entries corresponding to the parameter matrix, model matrix and model response.
x	Model matrix. Will be taken from fit if fit is provided.
y	Model response. Will be taken from fit if fit is provided.
pars	A named vector of parameter values. Will be taken from fit if fit is provided.

---

tidy.sclr                    *Tidy a sclr object.*

---

### Description

Summarises the objects returned by `sclr` into a `tibble`.

### Usage

```
## S3 method for class 'sclr'
tidy(x, ci_level = 0.95, ...)
```

**Arguments**

x	An object returned by <code>sclr</code> .
ci_level	Confidence level for the intervals.
...	Not used. Needed to match generic signature.

**Value**

A `tibble` with one row per model parameter. Columns:

term	Name of model parameter.
estimate	Point estimate.
std_error	Standard error.
conf_low	Lower bound of the confidence interval.
conf_high	Upper bound of the confidence interval.

---

two_titre_data	<i>Simulated two-titre antibody data</i>
----------------	--

---

**Description**

A simulated dataset containing 5000 independent observations on antibody titres and the corresponding infection status. The data was simulated to resemble real influenza infection and haemagglutinin + neuraminidase titre data.

**Usage**

```
two_titre_data
```

**Format**

A data frame with 5000 observations and 3 variables:

**logHI** haemagglutinin-inhibiting (HI) titre. True simulated titre on a log scale.

**logNI** neuraminidase-inhibiting titre. True simulated titre on a log scale.

**status** influenza infection status. 1 - infected. 0 - not infected

**Model**

The model behind the simulation was

$$\lambda * (1 - f(\beta_0 + \beta_1 * HI + \beta_2 * NI))$$

Where

- $f$  - Inverse logit function

- $\lambda = 0.5$
- $\beta_0 = -7.5$
- $\beta_1 = 2$
- $\beta_2 = 2$

# Index

- \* **datasets**
  - one\_titre\_data, 6
  - two\_titre\_data, 13
- broom, 9
- check\_baseline, 2
- coef, 9
- coef.sclr, 3
- confint.sclr (coef.sclr), 3
- find\_prot\_titre\_val, 3, 4
- get\_protection\_level, 3, 4
- glm, 2
- is\_sclr (new\_sclr), 5
- logLik, 9
- logLik.sclr (coef.sclr), 3
- model.frame, 9
- model.frame.sclr (coef.sclr), 3
- model.matrix, 9
- model.matrix.sclr (coef.sclr), 3
- new\_sclr, 5
- one\_titre\_data, 6
- predict, 9
- predict.sclr, 4, 6
- print, 9
- print.sclr, 7
- sclr, 2–5, 7, 8, 8, 10, 12, 13
- sclr\_fit, 5, 9, 10
- sclr\_ideal\_data, 11
- sclr\_log\_likelihood, 12
- summary, 9
- summary.sclr (print.sclr), 7
- tibble, 2, 5, 7, 11–13
- tidy, 7, 9
- tidy.sclr, 12
- two\_titre\_data, 13
- vcov, 9
- vcov.sclr (coef.sclr), 3