

# Package ‘serosv’

April 7, 2026

**Type** Package

**Title** Model Infectious Disease Parameters from Serosurveys

**Version** 1.3.0

**Description** An easy-to-use and efficient tool to estimate infectious diseases parameters using serological data. Implemented models include SIR models (`basic_sir_model()`, `static_sir_model()`, `mseir_model()`, `sir_subpops_model()`), parametric models (`polynomial_model()`, `fp_model()`), nonparametric models (`lp_model()`), semiparametric models (`penalized_splines_model()`), hierarchical models (`hierarchical_bayesian_model()`). The package is based on the book “Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective” (Hens, Niel & Shkedy, Ziv & Aerts, Marc & Faes, Christel & Damme, Pierre & Beutels, Philippe., 2013) <[doi:10.1007/978-1-4614-4072-7](https://doi.org/10.1007/978-1-4614-4072-7)>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 4.1.0)

**RoxygenNote** 7.3.3

**Imports** dplyr, tidyr, janitor, ggplot2, locfit, purrr, stringr, magrittr, methods, mgcv, mixdist, scam, mvtnorm, patchwork, assertthat, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), rstantools (>= 2.4.0), boot, pROC, stats4, rlang (>= 1.1.0)

**Suggests** covr, knitr, rmarkdown, bookdown, testthat (>= 3.0.0)

**Collate** 'data.R' 'fractional\_polynomial\_models.R'  
'polynomial\_models.R' 'utils.R' 'compare\_models.R'  
'correct\_prevalence.R' 'weibull\_model.R' 'farrington\_model.R'  
'nonparametric.R' 'semiparametric\_models.R' 'mixture\_model.R'  
'hierarchical\_bayesian\_model.R' 'to\_titer.R' 'serosv.R'  
'stanmodels.R' 'plots.R' 'compute\_ci.R' 'age\_time\_model.R'  
'predict.R' 'print.R'

**Config/testthat/edition** 3

**URL** <https://oucru-modelling.github.io/serosv/>,  
<https://github.com/OUCRU-Modelling/serosv>

**VignetteBuilder** knitr

**Biarch** true

**LinkingTo** BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0),  
 RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

**SystemRequirements** GNU make

**BugReports** <https://github.com/OUCRU-Modelling/serosv/issues>

**NeedsCompilation** yes

**Author** Anh Phan Truong Quynh [aut, cre] (ORCID:  
<https://orcid.org/0009-0000-2129-435X>),  
 Nguyen Pham Nguyen The [aut] (ORCID:  
<https://orcid.org/0000-0002-0356-2776>),  
 Long Bui Thanh [aut],  
 Tuyen Huynh [aut],  
 Thinh Ong [aut] (ORCID: <https://orcid.org/0000-0001-6772-9291>),  
 Marc Choisy [aut] (ORCID: <https://orcid.org/0000-0002-5187-6390>)

**Maintainer** Anh Phan Truong Quynh <anhptq@oucru.org>

**Repository** CRAN

**Date/Publication** 2026-04-07 13:20:02 UTC

## Contents

|  |    |
|--|----|
| serosv-package . . . . .                         | 4  |
| add_thresholds . . . . .                         | 5  |
| age_time_model . . . . .                         | 5  |
| compare_models . . . . .                         | 6  |
| compute_ci.age_time_model . . . . .              | 7  |
| compute_ci.default . . . . .                     | 8  |
| compute_ci.fp_model . . . . .                    | 9  |
| compute_ci.hierarchical_bayesian_model . . . . . | 9  |
| compute_ci.lp_model . . . . .                    | 10 |
| compute_ci.mixture_model . . . . .               | 10 |
| compute_ci.penalized_spline_model . . . . .      | 11 |
| compute_ci.weibull_model . . . . .               | 11 |
| correct_prevalence . . . . .                     | 12 |
| estimate_from_mixture . . . . .                  | 13 |
| est_foi . . . . .                                | 15 |
| farrington_model . . . . .                       | 15 |
| find_best_fp_powers . . . . .                    | 17 |
| fp_model . . . . .                               | 18 |
| hav_be_1993_1994 . . . . .                       | 19 |
| hav_be_2002 . . . . .                            | 20 |

|   |    |
|---|----|
| hav_bg_1964 . . . . .                         | 21 |
| hbv_ru_1999 . . . . .                         | 22 |
| hcv_be_2006 . . . . .                         | 23 |
| hierarchical_bayesian_model . . . . .         | 24 |
| lp_model . . . . .                            | 26 |
| mixture_model . . . . .                       | 28 |
| mumps_uk_1986_1987 . . . . .                  | 30 |
| parvob19_be_2001_2003 . . . . .               | 30 |
| parvob19_ew_1996 . . . . .                    | 31 |
| parvob19_fi_1997_1998 . . . . .               | 32 |
| parvob19_it_2003_2004 . . . . .               | 33 |
| parvob19_pl_1995_2004 . . . . .               | 34 |
| pava . . . . .                                | 35 |
| penalized_spline_model . . . . .              | 36 |
| plot.age_time_model . . . . .                 | 38 |
| plot.estimate_from_mixture . . . . .          | 39 |
| plot.farrington_model . . . . .               | 40 |
| plot.fp_model . . . . .                       | 40 |
| plot.hierarchical_bayesian_model . . . . .    | 41 |
| plot.lp_model . . . . .                       | 41 |
| plot.mixture_model . . . . .                  | 42 |
| plot.penalized_spline_model . . . . .         | 42 |
| plot.polynomial_model . . . . .               | 43 |
| plot.weibull_model . . . . .                  | 43 |
| plot_corrected_prev . . . . .                 | 44 |
| plot_gcv . . . . .                            | 44 |
| plot_standard_curve . . . . .                 | 45 |
| plot_titer_qc . . . . .                       | 46 |
| polynomial_model . . . . .                    | 46 |
| predict.age_time_model . . . . .              | 48 |
| predict.farrington_model . . . . .            | 49 |
| predict.fp_model . . . . .                    | 49 |
| predict.hierarchical_bayesian_model . . . . . | 50 |
| predict.lp_model . . . . .                    | 50 |
| predict.penalized_spline_model . . . . .      | 51 |
| predict.polynomial_model . . . . .            | 51 |
| predict.weibull_model . . . . .               | 52 |
| rubella_mumps_uk . . . . .                    | 53 |
| rubella_uk_1986_1987 . . . . .                | 53 |
| set_plot_style . . . . .                      | 54 |
| standardize_data . . . . .                    | 55 |
| tb_nl_1966_1973 . . . . .                     | 56 |
| to_titer . . . . .                            | 57 |
| transform_data . . . . .                      | 58 |
| vzv_be_1999_2000 . . . . .                    | 58 |
| vzv_be_2001_2003 . . . . .                    | 59 |
| vzv_parvo_be . . . . .                        | 60 |
| weibull_model . . . . .                       | 61 |

---

|                |  |
|----------------|--|
| serosv-package | <i>serosv: model infectious disease parameters</i> |
|----------------|--|

---

## Description

An easy-to-use and efficient tool to estimate infectious diseases parameters using serological data. Implemented models include SIR models (`basic_sir_model()`, `static_sir_model()`, `mseir_model()`, `sir_subpops_model()`), parametric models (`polynomial_model()`, `fp_model()`), nonparametric models (`lp_model()`), semiparametric models (`penalized_splines_model()`), hierarchical models (`hierarchical_bayesian_model()`). The package is based on the book "Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective" (Hens, Niel & Shkedy, Ziv & Aerts, Marc & Faes, Christel & Damme, Pierre & Beutels, Philippe., 2013) [doi:10.1007/9781461440727](https://doi.org/10.1007/9781461440727).

## Author(s)

**Maintainer:** Anh Phan Truong Quynh <anhptq@oucru.org> ([ORCID](#))

Authors:

- Nguyen Pham Nguyen The <nguyenpnt@oucru.org> ([ORCID](#))
- Long Bui Thanh
- Tuyen Huynh <tuyenhn@oucru.org>
- Thinh Ong <thinhop@oucru.org> ([ORCID](#))
- Marc Choisy <mchoisy@oucru.org> ([ORCID](#))

## See Also

Useful links:

- <https://oucru-modelling.github.io/serosv/>
- <https://github.com/OUCRU-Modelling/serosv>
- Report bugs at <https://github.com/OUCRU-Modelling/serosv/issues>

---

|                |   |
|----------------|---|
| add_thresholds | <i>Visualize positive threshold at different dilution factors</i> |
|----------------|---|

---

**Description**

Visualize positive threshold at different dilution factors

**Usage**

```
add_thresholds(dilution_factors, positive_threshold = 0.1, shift_text = 0.15)
```

**Arguments**

|                    |   |
|--------------------|---|
| dilution_factors   | dilution factors to be visualized   |
| positive_threshold | titer threshold for sample to be considered positive                                  |
| shift_text         | adjust how much the text is shifted along the x-axis (relative to the threshold line) |

---

|                |  |
|----------------|--|
| age_time_model | <i>Age-time varying seroprevalence</i> |
|----------------|--|

---

**Description**

Fit age-stratified seroprevalence across multiple time points. Also try to monotonize age (or birth cohort) - specific seroprevalence.

**Usage**

```
age_time_model(  
  data,  
  age_col = "age",  
  status_col = "status",  
  pos_col = "pos",  
  tot_col = "tot",  
  time_col = "date",  
  grouping_col = "group",  
  age_correct = F,  
  le = 512,  
  ci = 0.95,  
  monotonize_method = "pava"  
)
```

**Arguments**

|                   |  |
|-------------------|--|
| data              | input data, must have age, status, time, group columns, where group column determines how data is aggregated         |
| age_col           | name of the 'age' column (default age_col="age").  |
| status_col        | name of the 'status' column (default status_col="status").   |
| pos_col           | name of the 'pos' column (default pos_col="pos").  |
| tot_col           | name of the 'tot' column (default tot_col="tot").  |
| time_col          | name of the column for time (default to "date")  |
| grouping_col      | name of the column for time (default to "group")   |
| age_correct       | a boolean, if 'TRUE', monotonize age-specific prevalence. Monotonize birth cohort-specific seroprevalence otherwise. |
| le                | number of bins to generate age grid, used when monotonizing data   |
| ci                | confidence interval for smoothing  |
| monotonize_method | either "pava" or "scam"  |

**Value**

|              |  |
|--------------|--|
|              | a list of class time_age_model with 4 items  |
| out          | a data.frame with dimension n_group x 9, where columns 'info', 'sp', 'foi' store output for non-monotonized data and 'monotonized_info', 'monotonized_sp', 'monotonized_foi', 'monotonized_ci_mod' store output for monotonized data |
| grouping_col | name of the column for grouping  |
| age_correct  | a boolean indicating whether the data is monotonized across age or cohort  |
| datatype     | whether the input data is aggregated or line-listing data  |

---

|                |   |
|----------------|---|
| compare_models | <i>Generate table of metrics for model comparison</i> |
|----------------|---|

---

**Description**

Generate table of metrics for model comparison

**Usage**

```
compare_models(data, method = "AIC/BIC", ...)
```

**Arguments**

|        |   |
|--------|---|
| data   | input data to fit into the models   |
| method | method to compare models. Can be one of the built-in methods or a function to compute the returned metrics (see Details).   |
| ...    | models to be compared. Must be models created by serosv. If models' names are not provided, indices will be used instead for the 'model' column in the returned data.frame. |

**Details**

Built-in comparison methods include:

- computing AIC and BIC, which returns AIC, BIC values of the model if available
- cross validation (perform k-fold validation), which returns MSE and logloss (negative log Binomial likelihood) for aggregated data, or AUC and logloss (negative log Bernoulli likelihood) for linelisting data

**Value**

a data.frame with the following columns

|                 |  |
|-----------------|--|
| label           | name or index of the model   |
| type            | model type of the given model (a serosv model name)  |
| metrics columns | the columns for metrics of comparison, the number of which depends on the function that generate these metrics |

**Examples**

```
comparison_table <- suppressWarnings(
  compare_models(
    data = hav_bg_1964,
    method = "CV",
    polynomial_mod = ~polynomial_model(.x, k=1),
    penalized_spline = penalized_spline_model,
    farrington = ~farrington_model(.x, start=list(alpha=0.3,beta=0.1,gamma=0.03))
  )
)
# view table of metrics
comparison_table
# view the model fitted with the whole dataset
comparison_table$plots
```

---

```
compute_ci.age_time_model
```

*Compute confidence interval for time age model*

---

**Description**

Compute confidence interval for time age model

**Usage**

```
## S3 method for class 'age_time_model'
compute_ci(x, ci = 0.95, le = 100, ...)
```

**Arguments**

|     |  |
|-----|--|
| x   | serosv models                                    |
| ci  | confidence interval                              |
| le  | number of data for computing confidence interval |
| ... | arbitrary argument                               |

**Value**

confidence interval dataframe with n\_group x 3 cols, the columns are 'group', 'sp\_df', 'foi\_df'

---

compute\_ci.default      *Compute confidence interval for a model of serosv*

---

**Description**

Compute confidence interval for a model of serosv

**Usage**

```
## Default S3 method:
compute_ci(x, ci = 0.95, le = 100, ...)
```

**Arguments**

|     |  |
|-----|--|
| x   | serosv models                                    |
| ci  | confidence interval                              |
| le  | number of data for computing confidence interval |
| ... | arbitrary argument                               |

**Value**

confidence interval dataframe with 4 variables, x and y for the fitted values and ymin and ymax for the confidence interval

---

```
compute_ci.fp_model
```

*Compute confidence interval for fractional polynomial model*

---

**Description**

Compute confidence interval for fractional polynomial model

**Usage**

```
## S3 method for class 'fp_model'
compute_ci(x, ci = 0.95, le = 100, ...)
```

**Arguments**

|     |  |
|-----|--|
| x   | serosv models                                    |
| ci  | confidence interval                              |
| le  | number of data for computing confidence interval |
| ... | arbitrary argument                               |

**Value**

confidence interval dataframe with 4 variables, x and y for the fitted values and ymin and ymax for the confidence interval

---

```
compute_ci.hierarchical_bayesian_model
```

*Compute 95% credible interval for hierarchical Bayesian model*

---

**Description**

Compute 95% credible interval for hierarchical Bayesian model

**Usage**

```
## S3 method for class 'hierarchical_bayesian_model'
compute_ci(x, ...)
```

**Arguments**

|     |                     |
|-----|---------------------|
| x   | serosv models       |
| ... | arbitrary arguments |

**Value**

list of confidence interval for seroprevalence and foi. Each confidence interval dataframe with 4 variables, x and y for the fitted values and ymin and ymax for the confidence interval

---

```
compute_ci.lp_model Compute confidence interval for local polynomial model
```

---

**Description**

Compute confidence interval for local polynomial model

**Usage**

```
## S3 method for class 'lp_model'
compute_ci(x, ci = 0.95, ...)
```

**Arguments**

|     |                     |
|-----|---------------------|
| x   | serosv models       |
| ci  | confidence interval |
| ... | arbitrary arguments |

**Value**

confidence interval dataframe with 4 variables, x and y for the fitted values and ymin and ymax for the confidence interval

---

```
compute_ci.mixture_model
Compute confidence interval for mixture model
```

---

**Description**

Compute confidence interval for mixture model

**Usage**

```
## S3 method for class 'mixture_model'
compute_ci(x, ci = 0.95, ...)
```

**Arguments**

|     |                             |
|-----|-----------------------------|
| x   | serosv mixture_model object |
| ci  | confidence interval         |
| ... | arbitrary arguments         |

**Value**

list of confidence interval for susceptible and infected. Each confidence interval is a list with 2 items for lower and upper bound of the interval.

---

```
compute_ci.penalized_spline_model
```

*Compute confidence interval for penalized\_spline\_model*

---

### Description

Compute confidence interval for penalized\_spline\_model

### Usage

```
## S3 method for class 'penalized_spline_model'
compute_ci(x, ci = 0.95, ...)
```

### Arguments

|     |                     |
|-----|---------------------|
| x   | serosv models       |
| ci  | confidence interval |
| ... | arbitrary arguments |

### Value

list of confidence interval for seroprevalence and foi Each confidence interval dataframe with 4 variables, x and y for the fitted values and ymin and ymax for the confidence interval

---

```
compute_ci.weibull_model
```

*Compute confidence interval for Weibull model*

---

### Description

Compute confidence interval for Weibull model

### Usage

```
## S3 method for class 'weibull_model'
compute_ci(x, ci = 0.95, ...)
```

### Arguments

|     |                     |
|-----|---------------------|
| x   | serosv models       |
| ci  | confidence interval |
| ... | arbitrary argument  |

### Value

confidence interval dataframe with 4 variables, x and y for the fitted values and ymin and ymax for the confidence interval

---

|                    |  |
|--------------------|--|
| correct_prevalence | <i>Estimate the true sero prevalence using Frequentist/Bayesian estimation</i> |
|--------------------|--|

---

### Description

Estimate the true sero prevalence using Frequentist/Bayesian estimation

### Usage

```
correct_prevalence(
  data,
  bayesian = TRUE,
  age_col = "age",
  pos_col = "pos",
  tot_col = "tot",
  status_col = "status",
  init_se = 0.95,
  init_sp = 0.8,
  study_size_se = 1000,
  study_size_sp = 1000,
  chains = 1,
  warmup = 1000,
  iter = 2000
)
```

### Arguments

|               |   |
|---------------|---|
| data          | the input data frame, must either have columns for ‘age’, ‘pos’, ‘tot’ (for aggregated data) OR ‘age’, ‘status’ (for linelisting data)        |
| bayesian      | whether to adjust sero-prevalence using the Bayesian or frequentist approach. If set to ‘TRUE’, true sero-prevalence is estimated using MCMC. |
| age_col       | name of the ‘age’ column (default age_col="age").   |
| pos_col       | name of the ‘pos’ column (default pos_col="pos").   |
| tot_col       | name of the ‘tot’ column (default tot_col="tot").   |
| status_col    | name of the ‘status’ column (default status_col="status").  |
| init_se       | sensitivity of the serological test   |
| init_sp       | specificity of the serological test   |
| study_size_se | (applicable when ‘bayesian=TRUE’) study size for sensitivity validation study (i.e., number of confirmed infected patients in the study)      |
| study_size_sp | (applicable when ‘bayesian=TRUE’) study size for specificity validation study (i.e., number of confirmed non-infected patients in the study)  |
| chains        | (applicable when ‘bayesian=TRUE’) number of Markov chains   |
| warmup        | (applicable when ‘bayesian=TRUE’) number of warm up runs  |
| iter          | (applicable when ‘bayesian=TRUE’) number of iterations  |

**Value**

|                |   |
|----------------|---|
|                | a list of 3 items   |
| info           | estimated parameters (when 'bayesian = TRUE') or formula to compute corrected prevalence (when 'bayesian = FALSE')                    |
| df             | data.frame of input data (in aggregated form) with the 95% confidence interval for apparent (i.e. observed) seroprevalence            |
| corrected_sero | data.frame containing age, the corresponding estimated seroprevalence with 95% confidence/credible interval, and adjusted tot and pos |

**Examples**

```
data <- rubella_uk_1986_1987
correct_prevalence(data)
```

---

estimate\_from\_mixture *Estimate seroprevalence and FOI from a fixed mixture model*

---

**Description**

Estimate age-specific seroprevalence and FOI given a fitted mixture model (generated by [serosv::mixture\_model()])

**Usage**

```
estimate_from_mixture(
  age,
  antibody_level,
  threshold_status = NULL,
  mixture_model,
  s = "ps",
  sp = 83,
  monotonize = TRUE
)
```

**Arguments**

|                  |  |
|------------------|--|
| age              | vector of age  |
| antibody_level   | vector of the corresponding raw antibody level   |
| threshold_status | sero status using threshold approach in line listing (optional, for visualization and comparison only) |
| mixture_model    | mixture_model object generated by serosv::mixture_model()  |
| s                | smoothing basis used to fit antibody level   |
| sp               | smoothing parameter  |
| monotonize       | whether to monotinize seroprevalence (default to TRUE)   |

## Details

Antibody level (denoted  $Z$ ) is modeled using a 2-component Gaussian mixture model. Each component  $Z_j$  ( $j \in \{I, S\}$ ) represents the antibody level of the latent Infected and Susceptible sub-populations, following density  $f_j(z_j|\theta_j)$

Let  $\pi_{\text{TRUE}}(a)$  denotes the age-dependent mixing probability (i.e., the true prevalence), the density of the mixture is formulated as

$$f(z|z_I, z_S, a) = (1 - \pi_{\text{TRUE}}(a))f_S(z_S|\theta_S) + \pi_{\text{TRUE}}(a)f_I(z_I|\theta_I)$$

The mean  $E(Z|a)$  thus equals

$$\mu(a) = (1 - \pi_{\text{TRUE}}(a))\mu_S + \pi_{\text{TRUE}}(a)\mu_I$$

From which true prevalence can be computed as

$$\pi_{\text{TRUE}}(a) = \frac{\mu(a) - \mu_S}{\mu_I - \mu_S}$$

And FOI can then be inferred as

$$\lambda_{\text{TRUE}} = \frac{\mu'(a)}{\mu_I - \mu(a)}$$

Function `[serosv::mixture_model()]` fits antibody level data to  $f_S(z_S|\theta_S)$  and  $f_I(z_I|\theta_I)$

Function `[serosv::estimate_mixture()]` will then estimate age-specific antibody level  $\mu(a)$  and infer the estimation for  $\pi_{\text{TRUE}}(a)$  and  $\lambda_{\text{TRUE}}$

Refer to section 11.3. of the the book by Hens et al. (2012) for further details.

## Value

a list of class `estimated_from_mixture` with the following items

|                               |  |
|-------------------------------|--|
| <code>df</code>               | the dataframe used for fitting the model           |
| <code>info</code>             | a fitted "gam" model for $\mu(a)$                  |
| <code>sp</code>               | seroprevalence                                     |
| <code>foi</code>              | force of infection                                 |
| <code>threshold_status</code> | serostatus using threshold method only if provided |

## References

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:10.1007/9781-461440727.

## See Also

`[mgcv::gam()]` for more information about the fitted gam object

---

|         |                                    |
|---------|------------------------------------|
| est_foi | <i>Estimate force of infection</i> |
|---------|------------------------------------|

---

**Description**

Estimate force of infection

**Usage**

```
est_foi(t, sp)
```

**Arguments**

|    |                                |
|----|--------------------------------|
| t  | time (in this case age) vector |
| sp | seroprevalence vector          |

**Value**

computed foi vector

---

|                  |                                     |
|------------------|-------------------------------------|
| farrington_model | <i>The Farrington (1990) model.</i> |
|------------------|-------------------------------------|

---

**Description**

Fit age-stratified seroprevalence data using the Farrington (1990) model, which assumes the force of infection increases linearly with age and subsequently decreases exponentially.

**Usage**

```
farrington_model(  
  data,  
  start,  
  fixed = list(),  
  age_col = "age",  
  pos_col = "pos",  
  tot_col = "tot",  
  status_col = "status"  
)
```

**Arguments**

|            |  |
|------------|--|
| data       | the input data frame, must either have columns for ‘age’, ‘pos’, ‘tot’ (for aggregated data) OR ‘age’, ‘status’ (for linelisting data) |
| start      | Named list of vectors or single vector. Initial values for optimizer.  |
| fixed      | Named list of vectors or single vector. Parameter values to keep fixed during optimization.  |
| age_col    | name of the ‘age’ column (default age_col="age").  |
| pos_col    | name of the ‘pos’ column (default pos_col="pos").  |
| tot_col    | name of the ‘tot’ column (default tot_col="tot").  |
| status_col | name of the ‘status’ column (default status_col="status").   |

**Details**

The force of infection is defined as followed

$$\lambda(a) = (\alpha a - \gamma)e^{-\beta a} + \gamma$$

Where  $\gamma$  is called the long term residual for FOI, as  $a \rightarrow \infty$ ,  $\lambda(a) \rightarrow \gamma$

The seroprevalence can thus be estimated using the non-linear model

$$\pi(a) = 1 - \exp\left\{\frac{\alpha}{\beta}ae^{-\beta a} + \frac{1}{\beta}\left(\frac{\alpha}{\beta} - \gamma\right)(e^{-\beta a} - 1) - \gamma a\right\}$$

Refer to section 6.1.2. of the the book by Hens et al. (2012) for further details.

**Value**

a list of class `farrington_model` with 5 items

|          |   |
|----------|---|
| datatype | type of datatype used for model fitting (aggregated or linelisting) |
| df       | the dataframe used for fitting the model                            |
| info     | fitted "mle" object   |
| sp       | seroprevalence  |
| foi      | force of infection  |

**References**

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:10.1007/9781-461440727.

**See Also**

[stats4::mle()] for more information on the fitted mle object

**Examples**

```
df <- rubella_uk_1986_1987
model <- farrington_model(
  df,
  start=list(alpha=0.07,beta=0.1,gamma=0.03)
)
plot(model)
```

---

find\_best\_fp\_powers     *Returns the powers of the fractional polynomial model which has the lowest deviance score.*

---

**Description**

Return the best powers for a given degree

**Usage**

```
find_best_fp_powers(data, p, mc, degree, link = "logit")
```

**Arguments**

|        |  |
|--------|--|
| data   | the input data frame, must either have columns for 'age', 'pos', 'tot' (for aggregated data) OR 'age', 'status' (for linelisting data) |
| p      | a powers sequence to be tested.  |
| mc     | indicates if the returned model should be monotonic.   |
| degree | the maximum degree (i.e. number of power terms) to search for the best model. Recommended to be $\leq 2$ .                             |
| link   | the link function. Defaulted to "logit".   |

**Value**

list of 3 elements:

|          |                                    |
|----------|------------------------------------|
| p        | The best power for fp model.       |
| deviance | Deviance of the best fitted model. |
| model    | The best model fitted              |

---

fp\_model *A fractional polynomial model.*

---

### Description

Fractional polynomial model is a generalization of polynomial models where the power of the terms can be fractions, allowing more flexibility and better fit for data where asymptotic behavior is expected.

### Usage

```
fp_model(
  data,
  p,
  monotonic = FALSE,
  link = "logit",
  age_col = "age",
  pos_col = "pos",
  tot_col = "tot",
  status_col = "status"
)
```

### Arguments

|            |  |
|------------|--|
| data       | the input data frame, must either have 'age', 'pos', 'tot' columns (for aggregated data) OR 'age', 'status' for (linelisting data)   |
| p          | is either: <ul style="list-style-type: none"> <li>• a numeric vector specifying the powers to apply to the predictors</li> <li>• a named list with two elements, "p_range" and "degree". "p_range" is a sequence of powers and "degree" is the maximum degree. In which case the package will search for the best degree and power combinations</li> </ul> |
| monotonic  | whether the returned model should be monotonic (if a search is specified)  |
| link       | the link function for model. Defaulted to "logit".   |
| age_col    | name of the 'age' column (default age_col="age").  |
| pos_col    | name of the 'pos' column (default pos_col="pos").  |
| tot_col    | name of the 'tot' column (default tot_col="tot").  |
| status_col | name of the 'status' column (default status_col="status").   |

### Details

Instead of a polynomial, the linear predictor is now defined as

$$\eta_m(a, \beta, p_1, p_2, \dots, p_m) = \sum_{i=0}^m \beta_i H_i(a)$$

Where  $m$  is an integer,  $p_1 \leq p_2 \leq \dots \leq p_m$  is a sequence of powers, and  $H_i(a)$  is a transformation given by

$$H_i = \begin{cases} a^{p_i} & \text{if } p_i \neq p_{i-1}, \\ H_{i-1}(a) \times \log(a) & \text{if } p_i = p_{i-1}, \end{cases}$$

Refers to section 6.2. of the the book by Hens et al. (2012) for further details.

### Value

a list of class `fp_model` with 5 items

|                       |   |
|-----------------------|---|
| <code>datatype</code> | type of data used for fitting model (aggregated or linelisting) |
| <code>df</code>       | the dataframe used for fitting the model                        |
| <code>info</code>     | a fitted glm model  |
| <code>sp</code>       | seroprevalence  |
| <code>foi</code>      | force of infection  |

### References

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:10.1007/9781-461440727.

### See Also

[`stats::glm()`] for more information on glm object  
 [`polynomial_models()`]

### Examples

```
df <- hav_be_1993_1994
model <- fp_model(
  df,
  p=c(1.5, 1.6), link="cloglog")
plot(model)
```

---

|                               |  |
|-------------------------------|--|
| <code>hav_be_1993_1994</code> | <i>Hepatitis A serological data from Belgium in 1993 and 1994 (aggregated)</i> |
|-------------------------------|--|

---

### Description

A study of the prevalence of HAV antibodies conducted in the Flemish Community of Belgium in 1993 and early 1994

**Usage**

```
hav_be_1993_1994
```

**Format**

A data frame with 3 variables:

**age** Age group

**pos** Number of seropositive individuals

**tot** Total number of individuals surveyed

**Source**

Beutels, M., Van Damme, P., Aelvoet, W. et al. Prevalence of hepatitis A, B and C in the Flemish population. *Eur J Epidemiol* 13, 275-280 (1997). doi:10.1023/A:1007393405966

**Examples**

```
with(hav_be_1993_1994,  
      plot(  
        age, pos / tot,  
        pty = "s", cex = 0.34 * sqrt(tot), pch = 16, xlab = "age",  
        ylab = "seroprevalence", xlim = c(0, 86), ylim = c(0, 1)  
      )  
    )
```

---

hav\_be\_2002

*Hepatitis A serological data from Belgium in 2002 (line listing)*

---

**Description**

A subset of the serological dataset of Varicella-Zoster Virus (VZV) and Parvovirus B19 in Belgium where only individuals living in Flanders were selected

**Usage**

```
hav_be_2002
```

**Format**

A data frame with 2 variables:

**age** Age of individual

**seropositive** If the individual is seropositive or not

**Source**

Thiry, N., Beutels, P., Shkedy, Z. et al. The seroepidemiology of primary varicella-zoster virus infection in Flanders (Belgium). *Eur J Pediatr* 161, 588-593 (2002). doi:10.1007/s0043100210532

**Examples**

```
library(dplyr)
df <- hav_be_2002 %>%
  group_by(age) %>%
  summarise(pos = sum(seropositive), tot = n())

with(
  df,
  plot(
    age, pos / tot,
    pty = "s", cex = 0.34 * sqrt(tot), pch = 16, xlab = "age",
    ylab = "seroprevalence", xlim = c(0, 86), ylim = c(0, 1)
  )
)
```

---

hav\_bg\_1964

*Hepatitis A serological data from Bulgaria in 1964 (aggregated)*

---

**Description**

A cross-sectional survey conducted in 1964 in Bulgaria. Samples were collected from schoolchildren and blood donors.

**Usage**

hav\_bg\_1964

**Format**

A data frame with 3 variables:

**age** Age group

**pos** Number of seropositive individuals

**tot** Total number of individuals surveyed

**Source**

Keiding, Niels. "Age-Specific Incidence and Prevalence: A Statistical Perspective." *Journal of the Royal Statistical Society. Series A (Statistics in Society)* 154, no. 3 (1991): 371-412. doi:10.2307/2983150

**Examples**

```
with(
  hav_bg_1964,
  plot(
    age, pos / tot,
    pty = "s", cex = 0.6 * sqrt(tot), pch = 16, xlab = "age",
    ylab = "seroprevalence", xlim = c(0, 86), ylim = c(0, 1)
  )
)
```

---

 hbv\_ru\_1999

*Hepatitis B serological data from Russia in 1999 (aggregated)*


---

**Description**

A seroprevalence study conducted in St. Petersburg (more information in the book)

**Usage**

```
hbv_ru_1999
```

**Format**

A data frame with 4 variables:

**age** Age group

**pos** Number of seropositive individuals

**tot** Total number of individuals surveyed

**gender** Gender of cohort (unsure what 1 and 2 means)

**Source**

Mukomolov, S., L. Shliakhtenko, I. Levakova, and E. Shargorodskaya. Viral hepatitis in Russian federation. An analytical overview. Technical Report 213 (3), 3rd edn. St Petersburg Pasteur Institute, St Petersburg, 2000.

**Examples**

```
library(dplyr)
hbv_ru_1999$age <- trunc(hbv_ru_1999$age / 1) * 1
hbv_ru_1999$age[hbv_ru_1999$age > 40] <- trunc(
  hbv_ru_1999$age[hbv_ru_1999$age > 40] / 5
) * 5
df <- hbv_ru_1999 %>%
  group_by(age) %>%
  summarise(pos = sum(pos), tot = sum(tot))
```

```
plot(
  df$age, df$pos / df$tot,
  cex = 0.32 * sqrt(df$tot), pch = 16, xlab = "age",
  ylab = "seroprevalence", xlim = c(0, 72)
)
```

---

hcv\_be\_2006

*Hepatitis C serological data from Belgium in 2006 (line listing)*


---

### Description

A study of HCV infection among injecting drug users. All injecting drug users were interviewed by means of a standardized face-to-face interview and information on their socio-demographic status, drug use history, drug use, and related risk behavior was recorded

### Usage

```
hcv_be_2006
```

### Format

A data frame with 3 variables:

**dur** Duration of injection/Exposure time (years)

**seropositive** If the individual is seropositive or not

### Source

Mathei, C., Shkedy, Z., Denis, B., Kabali, C., Aerts, M., Molenberghs, G., Van Damme, P. and Buntinx, F. (2006), Evidence for a substantial role of sharing of injecting paraphernalia other than syringes/needles to the spread of hepatitis C among injecting drug users. *Journal of Viral Hepatitis*, 13: 560-570. [doi:10.1111/j.13652893.2006.00725.x](https://doi.org/10.1111/j.13652893.2006.00725.x)

### Examples

```
library(dplyr)
# snapping age to aggregated age group
# (credit: https://stackoverflow.com/a/12861810)
groups <- c(0.5:24.5)
range <- 0.5
low <- findInterval(hcv_be_2006$dur, groups)
high <- low + 1
low_diff <- hcv_be_2006$dur - groups[ifelse(low == 0, NA, low)]
high_diff <- groups[ifelse(high == 0, NA, high)] - hcv_be_2006$dur
mins <- pmin(low_diff, high_diff, na.rm = TRUE)
pick <- ifelse(!is.na(low_diff) & mins == low_diff, low, high)
hcv_be_2006$dur <- ifelse(
  mins <= range + .Machine$double.eps, groups[pick], hcv_be_2006$dur
```

```

)
hcv_be_2006 <- hcv_be_2006 %>%
  group_by(dur) %>%
  summarise(tot = n(), pos = sum(seropositive))

with(hcv_be_2006,
  plot(
    dur, pos / tot,
    cex = 0.42 * sqrt(tot), pch = 16,
    xlab = "duration of injection (years)",
    ylab = "seroprevalence", xlim = c(0, 25), ylim = c(0, 1)
  )
)

```

---

hierarchical\_bayesian\_model

*Hierarchical Bayesian Model*


---

## Description

Fit age-stratified seroprevalence to parametric hierarchical Bayesian models. Supported models including Farrington model (2 and 3 parameters variants) and Log Logistic model

## Usage

```

hierarchical_bayesian_model(
  data,
  age_col = "age",
  pos_col = "pos",
  tot_col = "tot",
  status_col = "status",
  type = "far3",
  chains = 1,
  warmup = 1500,
  iter = 5000
)

```

## Arguments

|            |  |
|------------|--|
| data       | the input data frame, must either have columns for ‘age’, ‘pos’, ‘tot’ (for aggregated data) OR ‘age’, ‘status’ (for linelisting data) |
| age_col    | name of the ‘age’ column (default age_col="age").  |
| pos_col    | name of the ‘pos’ column (default pos_col="pos").  |
| tot_col    | name of the ‘tot’ column (default tot_col="tot").  |
| status_col | name of the ‘status’ column (default status_col="status").   |
| type       | type of model ("far2", "far3" or "log_logistic")   |

|        |                         |
|--------|-------------------------|
| chains | number of Markov chains |
| warmup | number of warmup runs   |
| iter   | number of iterations    |

## Details

Consider a model for prevalence that has a parametric form  $\pi(a_i, \alpha)$  where  $\alpha$  is a parameter vector. Under a Bayesian framework, we can constraint the parameter space of the prior distribution  $P(\alpha)$  to achieve monotonicity of the posterior distribution  $P(\pi_1, \pi_2, \dots, \pi_m | y, n)$

Where:

-  $n = (n_1, n_2, \dots, n_m)$  and  $n_i$  is the sample size at age  $a_i$

-  $y = (y_1, y_2, \dots, y_m)$  and  $y_i$  is the number of infected individual from the  $n_i$  sampled subjects

For **Farrington** model with 3 parameters, prevalence is formulated as follow

$$\pi(a) = 1 - \exp\left\{\frac{\alpha_1}{\alpha_2} a e^{-\alpha_2 a} + \frac{1}{\alpha_2} \left(\frac{\alpha_1}{\alpha_2} - \alpha_3\right) (e^{-\alpha_2 a} - 1) - \alpha_3 a\right\}$$

The likelihood model is defined as  $y_i \sim \text{Bin}(n_i, \pi_i)$ , for  $i = 1, 2, 3, \dots, m$

The constraint on the parameter space can be incorporated by assuming truncated normal distribution for the components of  $\alpha$ ,  $\alpha = (\alpha_1, \alpha_2, \alpha_3)$  in  $\pi_i = \pi(a_i, \alpha)$

The flat hyperpriors are defined as  $\mu_j \sim \mathcal{N}(0, 10000)$  and  $\tau_j^{-2} \sim \Gamma(100, 100)$

For **Farrington** model with 2 parameters, it is equivalent to the previous model with  $\alpha_3 = 0$

For **Log logistic model**, seroprevalence is instead defined as

$$\pi(a) = \frac{\beta a^\alpha}{1 + \beta a^\alpha}, \quad \alpha, \beta > 0$$

The likelihood is similarly defined as  $y_i \sim \text{Bin}(n_i, \pi_i)$

The prior model of  $\alpha_1$  is specified as  $\alpha_1 \sim \text{truncated } \mathcal{N}(\mu_1, \tau_1)$  with flat hyperpriors as in Farrington model

$\beta$  is constrained to be positive by specifying  $\alpha_2 \sim \mathcal{N}(\mu_2, \tau_2)$

Refer to section Chapter 10.3 of the the book by Hens et al. (2012) for further details.

## Value

a list of class hierarchical\_bayesian\_model with 6 items

|          |   |
|----------|---|
| datatype | type of datatype used for model fitting (aggregated or linelisting)   |
| df       | the dataframe used for fitting the model                              |
| type     | type of bayesian model far2, far3 or log_logistic                     |
| info     | parameters for the fitted model                                       |
| sp       | seroprevalence  |
| foi      | force of infection  |
| sp_func  | function to compute seroprevalence given age and model parameters     |
| foi      | function to compute force of infection given age and model parameters |

## References

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:10.1007/9781-461440727.

## Examples

```
df <- mumps_uk_1986_1987
model <- hierarchical_bayesian_model(df, type="far3")
model$info
plot(model)
```

---

lp\_model

*A local polynomial model.*

---

## Description

Fit the age-specific seroprevalence to a local polynomial model, where the linear predictor is approximated locally at one particular age.

## Usage

```
lp_model(
  data,
  kern = "tcub",
  nn = 0,
  h = 0,
  deg = 2,
  age_col = "age",
  pos_col = "pos",
  tot_col = "tot",
  status_col = "status"
)
```

## Arguments

|      |   |
|------|---|
| data | the input data frame, must either have columns for ‘age’, ‘pos’, ‘tot’ (for aggregated data) OR ‘age’, ‘status’ (for linelisting data)  |
| kern | Weight function, default = "tcub". Other choices are "rect", "trwt", "tria", "epan", "bisq" and "gauss". Choices may be restricted when derivatives are required; e.g. for confidence bands and some bandwidth selectors. |
| nn   | Nearest neighbor component of the smoothing parameter. Default value is 0.7, unless either h is provided, in which case the default is 0.   |
| h    | The constant bandwidth of the smoothing parameter. Default: 0.  |

|            |  |
|------------|--|
| deg        | Degree of polynomial to use. Default: 2.                   |
| age_col    | name of the ‘age‘ column (default age_col="age").          |
| pos_col    | name of the ‘pos‘ column (default pos_col="pos").          |
| tot_col    | name of the ‘tot‘ column (default tot_col="tot").          |
| status_col | name of the ‘status‘ column (default status_col="status"). |

### Details

Consider a linear predictor  $\eta(a)$  approximated locally at one particular value  $a_0$ .

For a general degree  $p$ , the linear predictor for a neighbor of  $a_0$ , labeled  $a_i$  is equivalent to the Taylor approximation

$$\eta(a_i) = \eta(a_0) + \eta^{(1)}(a_0)(a_i - a_0) + \frac{\eta^{(2)}(a_0)}{2}(a_i - a_0)^2 + \dots + \frac{\eta^{(p)}(a_0)}{p!}(a_i - a_0)^p$$

$\eta(a_i)$  can be estimated by maximizing

$$\sum_{i=1}^N \ell_i \{Y_i, g^{-1}(\beta_0 + \beta_1(a_i - a_0) + \beta_2(a_i - a_0)^2 \dots + \beta_p(a_i - a_0)^p)\} K_h(a_i - a_0)$$

The estimator for the  $k$  – *th* derivative of  $\eta(a_0)$ , for  $k = 0, 1, \dots, p$  (degree of local polynomial) is thus:

$$\hat{\eta}^{(k)}(a_0) = k! \hat{\beta}_k(a_0)$$

The estimator for the prevalence at age  $a_0$  is then given by

$$\hat{\pi}(a_0) = g^{-1}\{\hat{\beta}_0(a_0)\}$$

Where  $g$  is the link function

The estimator for the force of infection at age  $a_0$  by assuming  $p \geq 1$  is as followed

$$\hat{\lambda}(a_0) = \hat{\beta}_1(a_0) \delta\{\hat{\beta}_0(a_0)\}$$

Where  $\delta\{\hat{\beta}_0(a_0)\} = \frac{dg^{-1}\{\hat{\beta}_0(a_0)\}}{d\hat{\beta}_0(a_0)}$

Refer to section 7.1 and 7.2. of the the book by Hens et al. (2012) for further details.

### Value

a list of class lp\_model with 6 items

|          |   |
|----------|---|
| datatype | type of datatype used for model fitting (aggregated or linelisting) |
| df       | the dataframe used for fitting the model                            |
| pi       | fitted locfit object for pi   |
| eta      | fitted locfit object for eta  |
| sp       | seroprevalence  |
| foi      | force of infection  |

## References

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:10.1007/9781-461440727.

## See Also

[`locfit::locfit()`] for more information on the fitted `locfit` object

## Examples

```
df <- mumps_uk_1986_1987
model <- lp_model(
  df,
  nn=0.7, kern="tcub"
)
plot(model)
```

---

mixture\_model

*Fit a mixture model to classify serostatus*

---

## Description

Fit the antibody level data to a 2-component Gaussian mixture model

## Usage

```
mixture_model(
  antibody_level,
  breaks = 40,
  pi = c(0.2, 0.8),
  mu = c(2, 6),
  sigma = c(0.5, 1)
)
```

## Arguments

|                             |   |
|-----------------------------|---|
| <code>antibody_level</code> | vector of the corresponding raw antibody level  |
| <code>breaks</code>         | number of intervals which the <code>antibody_level</code> are grouped into            |
| <code>pi</code>             | proportion of susceptible, infected   |
| <code>mu</code>             | a vector of means of component distributions (vector of 2 numbers in ascending order) |
| <code>sigma</code>          | a vector of standard deviations of component distributions (vector of 2 number)       |

## Details

Antibody level (denoted  $Z$ ) is modeled using a 2-component Gaussian mixture model. Each component  $Z_j$  ( $j \in \{I, S\}$ ) represents the antibody level of the latent Infected and Susceptible sub-populations, following density  $f_j(z_j|\theta_j)$

Let  $\pi_{\text{TRUE}}(a)$  denotes the age-dependent mixing probability (i.e., the true prevalence), the density of the mixture is formulated as

$$f(z|z_I, z_S, a) = (1 - \pi_{\text{TRUE}}(a))f_S(z_S|\theta_S) + \pi_{\text{TRUE}}(a)f_I(z_I|\theta_I)$$

The mean  $E(Z|a)$  thus equals

$$\mu(a) = (1 - \pi_{\text{TRUE}}(a))\mu_S + \pi_{\text{TRUE}}(a)\mu_I$$

From which true prevalence can be computed as

$$\pi_{\text{TRUE}}(a) = \frac{\mu(a) - \mu_S}{\mu_I - \mu_S}$$

And FOI can then be inferred as

$$\lambda_{\text{TRUE}} = \frac{\mu'(a)}{\mu_I - \mu(a)}$$

Function `[serosv::mixture_model()]` fits antibody level data to  $f_S(z_S|\theta_S)$  and  $f_I(z_I|\theta_I)$

Function `[serosv::estimate_mixture()]` will then estimate age-specific antibody level  $\mu(a)$  and infer the estimation for  $\pi_{\text{TRUE}}(a)$  and  $\lambda_{\text{TRUE}}$

Refer to section 11.3. of the the book by Hens et al. (2012) for further details.

## Value

a list of class `mixture_model` with the following items

|                          |   |
|--------------------------|---|
| <code>df</code>          | the dataframe used for fitting the model                                      |
| <code>info</code>        | list of 3 items parameters, distribution and constraints for the fitted model |
| <code>susceptible</code> | fitted distribution for susceptible   |
| <code>infected</code>    | fitted distribution for infected  |

## References

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:10.1007/9781-461440727.

## Examples

```
df <- vzv_be_2001_2003[vzv_be_2001_2003$age < 40.5,]
data <- df$VZVmIUml[order(df$age)]
model <- mixture_model(antibody_level = data)
model$info
plot(model)
```

---

mumps\_uk\_1986\_1987 *Mumps serological data from the UK in 1986 and 1987 (aggregated)*

---

### Description

a large survey of prevalence of antibodies to mumps and rubella viruses in the UK. The survey, covering subjects from 1 to over 65 years of age, provides information on the prevalence of antibody by age

### Usage

```
mumps_uk_1986_1987
```

### Format

A data frame with 3 variables:

**age** Midpoint of the age group (e.g. 1.5 = 1-2 years old, 2.5 = 2-3 years old)

**pos** Number of seropositive individuals

**tot** Total number of individuals surveyed

### Source

Morgan-Capner P, Wright J, Miller C L, Miller E. Surveillance of antibody to measles, mumps, and rubella by age. *British Medical Journal* 1988; 297 :770 [doi:10.1136/bmj.297.6651.770](https://doi.org/10.1136/bmj.297.6651.770)

### Examples

```
with(mumps_uk_1986_1987,  
  plot(age, pos / tot,  
        cex = 0.1 * sqrt(tot), pch = 16, xlab = "age", ylab = "seroprevalence",  
        xlim = c(0, 45), ylim = c(0, 1)  
  )  
)
```

---

parvob19\_be\_2001\_2003 *Parvo B19 serological data from Belgium from 2001-2003 (line listing)*

---

### Description

A seroprevalence survey testing for parvovirus B19 IgG antibody, performed on large representative national serum banks in Belgium, England and Wales, Finland, Italy, and Poland. The sera were collected between 1995 and 2004 and were obtained from residual sera submitted for routine laboratory testing.

**Usage**

```
parvob19_be_2001_2003
```

**Format**

A data frame with 5 variables:

**age** Age of individual

**seropositive** If the individual is seropositive or not

**year** Year surveyed

**gender** Gender of individual

**parvouml** Parvo B19 antibody units per ml

**Source**

MOSSONG, J., N. HENS, V. FRIEDERICHS, I. DAVIDKIN, M. BROMAN, B. LITWINSKA, J. SIENNICKA, et al. "Parvovirus B19 Infection in Five European Countries: Seroepidemiology, Force of Infection and Maternal Risk of Infection." *Epidemiology and Infection* 136, no. 8 (2008): 1059-68. doi:[10.1017/S0950268807009661](https://doi.org/10.1017/S0950268807009661)

**Examples**

```
library(dplyr)
df <- parvob19_be_2001_2003 %>%
  group_by(age) %>%
  summarise(pos = sum(seropositive), tot = n())
plot(df$age, df$pos / df$tot,
     cex = 0.3 * sqrt(df$tot), pch = 16, xlab = "age", ylab = "seroprevalence",
     xlim = c(0, 82), ylim = c(0, 1)
  )
```

---

|                  |   |
|------------------|---|
| parvob19_ew_1996 | <i>Parvo B19 serological data from England and Wales in 1996 (line listing)</i> |
|------------------|---|

---

**Description**

A seroprevalence survey testing for parvovirus B19 IgG antibody, performed on large representative national serum banks in Belgium, England and Wales, Finland, Italy, and Poland. The sera were collected between 1995 and 2004 and were obtained from residual sera submitted for routine laboratory testing.

**Usage**

```
parvob19_ew_1996
```

**Format**

A data frame with 5 variables:

**age** Age of individual

**seropositive** If the individual is seropositive or not

**year** Year surveyed

**gender** Gender of individual

**parvouml** Parvo B19 antibody units per ml

**Source**

MOSSONG, J., N. HENS, V. FRIEDERICHS, I. DAVIDKIN, M. BROMAN, B. LITWINSKA, J. SIENNICKA, et al. "Parvovirus B19 Infection in Five European Countries: Seroepidemiology, Force of Infection and Maternal Risk of Infection." *Epidemiology and Infection* 136, no. 8 (2008): 1059-68. doi:[10.1017/S0950268807009661](https://doi.org/10.1017/S0950268807009661)

**Examples**

```
# Note: This figure will look different to that of in the book, since we
# believe that the original authors has made some errors in specifying
# the sample size of the dots.

library(dplyr)
df <- parvob19_ew_1996 %>%
  mutate(age = round(age)) %>%
  group_by(age) %>%
  summarise(pos = sum(seropositive), tot = n())
plot(df$age, df$pos / df$tot,
     cex = 0.3 * sqrt(df$tot), pch = 16, xlab = "age", ylab = "seroprevalence",
     xlim = c(0, 82), ylim = c(0, 1)
  )
```

---

parvob19\_fi\_1997\_1998 *Parvo B19 serological data from Finland from 1997-1998 (line listing)*

---

**Description**

A seroprevalence survey testing for parvovirus B19 IgG antibody, performed on large representative national serum banks in Belgium, England and Wales, Finland, Italy, and Poland. The sera were collected between 1995 and 2004 and were obtained from residual sera submitted for routine laboratory testing.

**Usage**

```
parvob19_fi_1997_1998
```

**Format**

A data frame with 5 variables:

**age** Age of individual

**seropositive** If the individual is seropositive or not

**year** Year surveyed

**gender** Gender of individual

**parvovuml** Parvo B19 antibody units per ml

**Source**

MOSSONG, J., N. HENS, V. FRIEDERICHS, I. DAVIDKIN, M. BROMAN, B. LITWINSKA, J. SIENNICKA, et al. "Parvovirus B19 Infection in Five European Countries: Seroepidemiology, Force of Infection and Maternal Risk of Infection." *Epidemiology and Infection* 136, no. 8 (2008): 1059-68. doi:[10.1017/S0950268807009661](https://doi.org/10.1017/S0950268807009661)

**Examples**

```
# Note: This figure will look different to that of in the book, since we
# believe that the original authors has made some errors in specifying
# the sample size of the dots.

library(dplyr)
df <- parvob19_fi_1997_1998 %>%
  mutate(age = round(age)) %>%
  group_by(age) %>%
  summarise(pos = sum(seropositive), tot = n())
plot(df$age, df$pos / df$tot,
     cex = 0.4 * sqrt(df$tot), pch = 16, xlab = "age", ylab = "seroprevalence",
     xlim = c(0, 82), ylim = c(0, 1)
  )
```

---

parvob19\_it\_2003\_2004 *Parvo B19 serological data from Italy from 2003-2004 (line listing)*

---

**Description**

A seroprevalence survey testing for parvovirus B19 IgG antibody, performed on large representative national serum banks in Belgium, England and Wales, Finland, Italy, and Poland. The sera were collected between 1995 and 2004 and were obtained from residual sera submitted for routine laboratory testing.

**Usage**

```
parvob19_it_2003_2004
```

**Format**

A data frame with 5 variables:

**age** Age of individual

**seropositive** If the individual is seropositive or not

**year** Year surveyed

**gender** Gender of individual

**parvouml** Parvo B19 antibody units per ml

**Source**

MOSSONG, J., N. HENS, V. FRIEDERICHS, I. DAVIDKIN, M. BROMAN, B. LITWINSKA, J. SIENNICKA, et al. "Parvovirus B19 Infection in Five European Countries: Seroepidemiology, Force of Infection and Maternal Risk of Infection." *Epidemiology and Infection* 136, no. 8 (2008): 1059-68. doi:[10.1017/S0950268807009661](https://doi.org/10.1017/S0950268807009661)

**Examples**

```
library(dplyr)
df <- parvob19_it_2003_2004 %>%
  group_by(age) %>%
  summarise(pos = sum(seropositive), tot = n())
plot(df$age, df$pos / df$tot,
     cex = 0.32 * sqrt(df$tot), pch = 16, xlab = "age", ylab = "seroprevalence",
     xlim = c(0, 82), ylim = c(0, 1)
  )
```

---

parvob19\_pl\_1995\_2004 *Parvo B19 serological data from Poland from 1995-2004 (line listing)*

---

**Description**

A seroprevalence survey testing for parvovirus B19 IgG antibody, performed on large representative national serum banks in Belgium, England and Wales, Finland, Italy, and Poland. The sera were collected between 1995 and 2004 and were obtained from residual sera submitted for routine laboratory testing.

**Usage**

```
parvob19_pl_1995_2004
```

**Format**

A data frame with 5 variables:

**age** Age of individual

**seropositive** If the individual is seropositive or not

**year** Year surveyed

**gender** Gender of individual

**parvovuml** Parvo B19 antibody units per ml

**Source**

MOSSONG, J., N. HENS, V. FRIEDERICHS, I. DAVIDKIN, M. BROMAN, B. LITWINSKA, J. SIENNICKA, et al. "Parvovirus B19 Infection in Five European Countries: Seroepidemiology, Force of Infection and Maternal Risk of Infection." *Epidemiology and Infection* 136, no. 8 (2008): 1059-68. doi:[10.1017/S0950268807009661](https://doi.org/10.1017/S0950268807009661)

**Examples**

```
# Note: This figure will look different to that of in the book, since we
# believe that the original authors has made some errors in specifying
# the sample size of the dots.

library(dplyr)
df <- parvob19_pl_1995_2004 %>%
  mutate(age = round(age)) %>%
  group_by(age) %>%
  summarise(pos = sum(seropositive), tot = n())
plot(df$age, df$pos / df$tot,
     cex = 0.32 * sqrt(df$tot), pch = 16, xlab = "age", ylab = "seroprevalence",
     xlim = c(0, 82), ylim = c(0, 1)
  )
```

---

pava

*Monotonize seroprevalence*

---

**Description**

Monotonize seroprevalence

**Usage**

```
pava(pos = pos, tot = rep(1, length(pos)))
```

**Arguments**

**pos** the positive count vector.

**tot** the total count vector.

**Value**

computed list of 2 items pai1 for original values and pai2 for monotonized value

---

penalized\_spline\_model

*Penalized Spline model*

---

**Description**

Fit age-specific seroprevalence to a semi-parametric model where predictor is modeled with penalized splines. The penalized splines can be estimated by either (1) penalized likelihood framework or (2) mixed model framework

**Usage**

```
penalized_spline_model(
  data,
  age_col = "age",
  pos_col = "pos",
  tot_col = "tot",
  status_col = "status",
  s = "bs",
  link = "logit",
  framework = "pl",
  sp = NULL
)
```

**Arguments**

|            |  |
|------------|--|
| data       | the input data frame, must either have columns for ‘age’, ‘pos’, ‘tot’ (for aggregated data) OR columns for ‘age’, ‘status’ (for linelisting data) |
| age_col    | name of the ‘age’ column (default age_col="age").  |
| pos_col    | name of the ‘pos’ column (default pos_col="pos").  |
| tot_col    | name of the ‘tot’ column (default tot_col="tot").  |
| status_col | name of the ‘status’ column (default status_col="status").   |
| s          | smoothing basis to use   |
| link       | link function to use   |
| framework  | which approach to fit the model ("pl" for penalized likelihood framework, "glmm" for generalized linear mixed model framework)                     |
| sp         | smoothing parameter  |

### Details

In the semi-parametric model, the predictor is formulated as a penalized spline with truncated power basis functions of degree  $p$  and fixed knots  $\kappa_1, \dots, \kappa_k$  as followed

$$\eta(a_i) = \beta_0 + \beta_1 a_i + \dots + \beta_p a_i^p + \sum_{k=1}^k u_k (a_i - \kappa_k)_+^p$$

Where:

$$(a_i - \kappa_k)_+^p = \begin{cases} 0, & a_i \leq \kappa_k \\ (a_i - \kappa_k)^p, & a_i > \kappa_k \end{cases}$$

FOI can then be derived by

$$\hat{\lambda}(a_i) = [\hat{\beta}_1, 2\hat{\beta}_2 a_i, \dots, p\hat{\beta}_p a_i^{p-1} + \sum_{k=1}^k p\hat{u}_k (a_i - \kappa_k)_+^{p-1}] \delta(\hat{\eta}(a_i))$$

Where  $\delta(\cdot)$  is determined by the link function used in the model

In matrix annotation, the mean structure model for  $\eta(a_i)$  becomes

$$\eta = \mathbf{X}\beta + \mathbf{Z}\mathbf{u}$$

Where  $\eta = [\eta(a_1) \dots \eta(a_N)]^T$ ,  $\beta = [\beta_0 \beta_1 \dots \beta_p]^T$ , and  $\mathbf{u} = [u_1 u_2 \dots u_k]^T$  are the regression with corresponding design matrices

$$\mathbf{X} = \begin{bmatrix} 1 & a_1 & a_1^2 & \dots & a_1^p \\ 1 & a_2 & a_2^2 & \dots & a_2^p \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & a_N & a_N^2 & \dots & a_N^p \end{bmatrix}, \mathbf{Z} = \begin{bmatrix} (a_1 - \kappa_1)_+^p & (a_1 - \kappa_2)_+^p & \dots & (a_1 - \kappa_k)_+^p \\ (a_2 - \kappa_1)_+^p & (a_2 - \kappa_2)_+^p & \dots & (a_2 - \kappa_k)_+^p \\ \vdots & \vdots & \dots & \vdots \\ (a_N - \kappa_1)_+^p & (a_N - \kappa_2)_+^p & \dots & (a_N - \kappa_k)_+^p \end{bmatrix}$$

Under **penalized likelihood framework**, the model is fitted by maximizing the following likelihood

$$\phi^{-1}[y^T(\mathbf{X}\beta + \mathbf{Z}\mathbf{u}) - \mathbf{1}^T c(\mathbf{X}\beta + \mathbf{Z}\mathbf{u})] - \frac{1}{2} \lambda^2 \begin{bmatrix} \beta \\ \mathbf{u} \end{bmatrix}^T D \begin{bmatrix} \beta \\ \mathbf{u} \end{bmatrix}$$

Where:

- $X\beta + Zu$  is the predictor
- $D$  is a known semi-definite penalty matrix
- $y$  is the response vector
- $\mathbf{1}$  the unit vector,  $c(\cdot)$  is determined by the link function used
- $\lambda$  is the smoothing parameter (larger values -> smoother curves)
- $\phi$  is the overdispersion parameter and equals 1 if there is no overdispersion

Under the **mixed model framework**, the model instead treats the coefficients  $\mathbf{u}$  in the likelihood formulation as random effects with  $\mathbf{u} \sim N(\mathbf{0}, \sigma_u^2 \mathbf{I})$

Refer to section 8.1 and 8.2 of the the book by Hens et al. (2012) for further details.

**Value**

a list of class `penalized_spline_model` with 6 attributes

|                        |  |
|------------------------|--|
| <code>datatype</code>  | type of datatype used for model fitting (aggregated or linelisting)                                      |
| <code>df</code>        | the dataframe used for fitting the model   |
| <code>framework</code> | either <code>pl</code> or <code>glmm</code>  |
| <code>info</code>      | fitted "gam" model when framework is <code>pl</code> or "gamm" model when framework is <code>glmm</code> |
| <code>sp</code>        | seroprevalence   |
| <code>foi</code>       | force of infection   |

**References**

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:[10.1007/9781-461440727](https://doi.org/10.1007/9781-461440727).

**See Also**

[`mgcv::gam()`], [`mgcv::gamm()`] for more information the fitted gam and gamm model

**Examples**

```
data <- parvob19_be_2001_2003
data$status <- data$seropositive
model <- penalized_spline_model(data, framework="glmm")
model$info$gam
plot(model)
```

---

`plot.age_time_model`    *Plot output for age\_time\_model*

---

**Description**

Plot output for `age_time_model`

**Usage**

```
## S3 method for class 'age_time_model'
plot(x, ...)
```

**Arguments**

- x - a 'age\_time\_model' object
- ... arbitrary params. Supported options include:
- facet: Whether to facet the plot by group.
  - modtype: Which model to plot, either "monotonized" or "non-monotonized".
  - le: Number of bins used to generate the x-axis; higher values produce smoother curves.
  - cex: Adjusts the size of data points (only when facet = TRUE).

**Value**

ggplot object

---

plot.estimate\_from\_mixture

*plot() overloading for result of estimate\_from\_mixture*

---

**Description**

plot() overloading for result of estimate\_from\_mixture

**Usage**

```
## S3 method for class 'estimate_from_mixture'  
plot(x, ...)
```

**Arguments**

- x the mixture\_model
- ... arbitrary params.

**Value**

ggplot object

---

plot.farrington\_model *plot() overloading for Farrington model*

---

**Description**

plot() overloading for Farrington model

**Usage**

```
## S3 method for class 'farrington_model'  
plot(x, ...)
```

**Arguments**

x                    the Farrington model object.  
...                  arbitrary params.

**Value**

ggplot object

---

plot.fp\_model            *plot() overloading for fractional polynomial model*

---

**Description**

plot() overloading for fractional polynomial model

**Usage**

```
## S3 method for class 'fp_model'  
plot(x, ...)
```

**Arguments**

x                    the fractional polynomial model object.  
...                  arbitrary params.

**Value**

ggplot object

---

`plot.hierarchical_bayesian_model`  
*plot() overloading for hierarchical\_bayesian\_model*

---

**Description**

plot() overloading for hierarchical\_bayesian\_model

**Usage**

```
## S3 method for class 'hierarchical_bayesian_model'  
plot(x, ...)
```

**Arguments**

x hierarchical\_bayesian\_model object created by serosv.  
... arbitrary params.

**Value**

ggplot object

---

`plot.lp_model` *plot() overloading for local polynomial model*

---

**Description**

plot() overloading for local polynomial model

**Usage**

```
## S3 method for class 'lp_model'  
plot(x, ...)
```

**Arguments**

x the local polynomial model object.  
... arbitrary params.

**Value**

ggplot object

---

plot.mixture\_model     *plot() overloading for mixture model*

---

**Description**

plot() overloading for mixture model

**Usage**

```
## S3 method for class 'mixture_model'  
plot(x, ...)
```

**Arguments**

x                    the mixture\_model  
...                   arbitrary params.

**Value**

ggplot object

---

plot.penalized\_spline\_model  
                          *plot() overloading for penalized spline*

---

**Description**

plot() overloading for penalized spline

**Usage**

```
## S3 method for class 'penalized_spline_model'  
plot(x, ...)
```

**Arguments**

x                    the penalized\_spline\_model object  
...                   arbitrary params.

**Value**

ggplot object

---

`plot.polynomial_model` *plot() overloading for polynomial model*

---

**Description**

plot() overloading for polynomial model

**Usage**

```
## S3 method for class 'polynomial_model'  
plot(x, ...)
```

**Arguments**

x                    the polynomial model object  
...                   arbitrary params.

**Value**

ggplot object

---

`plot.weibull_model` *plot() overloading for Weibull model*

---

**Description**

plot() overloading for Weibull model

**Usage**

```
## S3 method for class 'weibull_model'  
plot(x, ...)
```

**Arguments**

x                    the Weibull model object.  
...                   arbitrary params.

**Value**

ggplot object

---

plot\_corrected\_prev    *Plot output for corrected\_prevalence*

---

**Description**

Plot output for corrected\_prevalence

**Usage**

```
plot_corrected_prev(x, y = NULL, facet = FALSE)
```

**Arguments**

x                    - the output of 'correct\_prevalence()' function  
y                    - another output of 'correct\_prevalence()' function (optional, for comparison only)  
facet                - whether to plot as facets or on the same plot (only when y is provided)

**Value**

ggplot object

---

plot\_gcv                    *Plotting GCV values with respect to different nn-s and h-s parameters.*

---

**Description**

Refers to section 7.2.

**Usage**

```
plot_gcv(age, pos, tot, nn_seq, h_seq, kern = "tcub", deg = 2)
```

**Arguments**

age                    the age vector.  
pos                    the pos vector.  
tot                    the tot vector.#'  
nn\_seq                Nearest neighbor sequence.  
h\_seq                Smoothing parameter sequence.  
kern                   Weight function, default = "tcub". Other choices are "rect", "trwt", "tria", "epan", "bisq" and "gauss". Choices may be restricted when derivatives are required; e.g. for confidence bands and some bandwidth selectors.  
deg                    Degree of polynomial to use. Default: 2.

**Value**

plot of gcv value

**Examples**

```
df <- mumps_uk_1986_1987
plot_gcv(
  df$age, df$pos, df$tot,
  nn_seq = seq(0.2, 0.8, by=0.1),
  h_seq = seq(5, 25, by=1)
)
```

---

plot\_standard\_curve    *Visualize standard curves for each plate*

---

**Description**

Visualize standard curves for each plate

**Usage**

```
plot_standard_curve(
  x,
  facet = TRUE,
  xlab = "log10(concentration)",
  ylab = "Optical density",
  datapoint_size = 2
)
```

**Arguments**

|                |   |
|----------------|---|
| x              | output of 'to_titer()'  |
| facet          | whether to faceted by plates or plot all standard curves on a single plot |
| xlab           | label of the x axis   |
| ylab           | label of the y axis   |
| datapoint_size | size of the data point (only applicable when 'facet=TRUE')                |

---

plot\_titer\_qc                    *Quality control plot*

---

### Description

Visualize for each sample, whether titer estimates can be computed at the tested dilutions.

### Usage

```
plot_titer_qc(x, n_plates = 18, n_samples = 22, n_dilutions = 3)
```

### Arguments

|             |   |
|-------------|---|
| x           | output of 'to_titer()'                      |
| n_plates    | maximum number of plates to plot            |
| n_samples   | maximum number of samples per plate to plot |
| n_dilutions | number of dilutions used for testing        |

### Details

Each sample is represented by a 'n\_estimates x n\_dilutions' grid where cell color indicate estimate availability (green = estimate available, orange = result too low, red = result too high)

These sample grids are arranged in columns where each column represent samples from a plate

---

polynomial\_model                *Polynomial models*

---

### Description

Fit age-stratified seroprevalence data to serocatalytic models formulated as polynomials.

### Usage

```
polynomial_model(
  data,
  k,
  link = "log",
  age_col = "age",
  pos_col = "pos",
  tot_col = "tot",
  status_col = "status"
)
```

**Arguments**

|            |  |
|------------|--|
| data       | the input data frame, must either have columns for 'age', 'pos', 'tot' (for aggregated data) OR 'age', 'status' (for linelisting data) |
| k          | degree of the polynomial. (k=1 for Muench model, k=2 for Griffith model, k=3 for Grenfell model).                                      |
| link       | link function (default link="log").  |
| age_col    | name of the 'age' column (default age_col="age").  |
| pos_col    | name of the 'pos' column (default pos_col="pos").  |
| tot_col    | name of the 'tot' column (default tot_col="tot").  |
| status_col | name of the 'status' column (default status_col="status").   |

**Details**

The seroprevalence is assumed to follow the general format

$$\pi(a) = 1 - e^{-\sum_{i=1}^k \beta_i a^i}$$

Which implies the force of infection to be  $\lambda(a) = \sum_{i=1}^k \beta_i i a^{i-1}$

Where:

- $\pi$  is the seroprevalence at age  $a$
- $a$  is the variable age
- $k$  is the degree of the polynomial

The seroprevalence  $\pi(a)$  is fitted using a GLM with log link with the linear predictor  $\eta(a) = \sum_{i=1}^k \beta_i a^i$

Muench (1934) model is equivalent to a degree 1 ( $k = 1$ ) linear predictor

Griffith model is equivalent to a degree 2 ( $k = 2$ ) linear predictor

Grenfell & Anderson (1985) suggested a higher order polynomials ( $k \geq 3$ )

Refer to section 6.1.1. of the the book by Hens et al. (2012) for further details.

**Value**

a list of class polynomial\_model with 5 items

|          |   |
|----------|---|
| datatype | type of datatype used for model fitting (aggregated or linelisting) |
| df       | the dataframe used for fitting the model                            |
| info     | fitted "glm" object   |
| sp       | seroprevalence  |
| foi      | force of infection  |

## References

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:10.1007/9781-461440727.

Grenfell, B. T., and R. M. Anderson. 1985. "The Estimation of Age-Related Rates of Infection from Case Notifications and Serological Data." *The Journal of Hygiene* 95 (2): 419–36. doi:10.1017/s0022172400062859.

Muench, Hugo. 1934. "Derivation of Rates from Summation Data by the Catalytic Curve." *Journal of the American Statistical Association* 29 (185): 25–38. doi:10.1080/01621459.1934.10502684.

## Examples

```
data <- parvob19_fi_1997_1998[order(parvob19_fi_1997_1998$age), ]
aggregated <- transform_data(data, stratum_col = "age", status_col="seropositive")

# fit with aggregated data
model <- polynomial_model(aggregated, k = 1)
# fit with linelisting data
model <- polynomial_model(data,
  status_col = "seropositive",
  k = 1)
plot(model)
```

---

predict.age\_time\_model

*Predict from the age\_time\_mdoel*

---

## Description

Predict from the age\_time\_mdoel

## Usage

```
## S3 method for class 'age_time_model'
predict(object, newdata, modtype = "monotonized", ...)
```

## Arguments

|         |  |
|---------|--|
| object  | serosv models  |
| newdata | data.frame with age column to generate prediction                            |
| modtype | either "monotonized" (to predict using monotized model) or "non-monotonized" |
| ...     | arbitrary argument   |

## Value

confidence interval dataframe with n\_group x 3 cols, the columns are 'group', 'sp\_df', 'foi\_df'

---

```
predict.farrington_model
```

*Prediction for serosv Farrington model*

---

**Description**

Prediction for serosv Farrington model

**Usage**

```
## S3 method for class 'farrington_model'
predict(object, newdata = NULL, ...)
```

**Arguments**

|         |   |
|---------|---|
| object  | serosv models                                     |
| newdata | data.frame with age column to generate prediction |
| ...     | arbitrary argument                                |

**Value**

prediction output

---

```
predict.fp_model
```

*Prediction for serosv fractional polynomial model*

---

**Description**

Prediction for serosv fractional polynomial model

**Usage**

```
## S3 method for class 'fp_model'
predict(object, newdata = NULL, ...)
```

**Arguments**

|         |   |
|---------|---|
| object  | serosv models                                     |
| newdata | data.frame with age column to generate prediction |
| ...     | arbitrary argument                                |

**Value**

prediction output

**See Also**

[stats::predict.glm()] for more information on the predict function

---

predict.hierarchical\_bayesian\_model

*Predict from an hierarchical bayesian model*

---

**Description**

Predict from an hierarchical bayesian model

**Usage**

```
## S3 method for class 'hierarchical_bayesian_model'
predict(object, newdata = NULL, ...)
```

**Arguments**

|         |   |
|---------|---|
| object  | serosv models                                     |
| newdata | data.frame with age column to generate prediction |
| ...     | arbitrary arguments                               |

**Value**

list of confidence interval for seroprevalence and foi. Each confidence interval dataframe with 4 variables, x and y for the fitted values and ymin and ymax for the confidence interval

---

predict.lp\_model

*Prediction for serosv local polynomial model*

---

**Description**

Prediction for serosv local polynomial model

**Usage**

```
## S3 method for class 'lp_model'
predict(object, newdata = NULL, ...)
```

**Arguments**

|         |   |
|---------|---|
| object  | serosv models                                     |
| newdata | data.frame with age column to generate prediction |
| ...     | arbitrary argument                                |

**Value**

prediction output

---

`predict.penalized_spline_model`  
*Prediction for serosv penalized spline model*

---

**Description**

Prediction for serosv penalized spline model

**Usage**

```
## S3 method for class 'penalized_spline_model'  
predict(object, newdata = NULL, ...)
```

**Arguments**

|                      |   |
|----------------------|---|
| <code>object</code>  | serosv models                                     |
| <code>newdata</code> | data.frame with age column to generate prediction |
| <code>...</code>     | arbitrary argument                                |

**Value**

prediction output

**See Also**

[`mgcv::predict.gam()`] for more information on the predict function

---

`predict.polynomial_model`  
*Prediction for serosv polynomial model*

---

**Description**

A wrapper of `predict.glm` for direct prediction from `polynomial_model` object

**Usage**

```
## S3 method for class 'polynomial_model'  
predict(object, newdata = NULL, ...)
```

**Arguments**

|         |   |
|---------|---|
| object  | serosv models                                     |
| newdata | data.frame with age column to generate prediction |
| ...     | arbitrary argument                                |

**Value**

prediction output

**See Also**

[stats::predict.glm()] for more information on the predict function

---

`predict.weibull_model` *Prediction for serosv Weibull model*

---

**Description**

Prediction for serosv Weibull model

**Usage**

```
## S3 method for class 'weibull_model'  
predict(object, newdata = NULL, ...)
```

**Arguments**

|         |   |
|---------|---|
| object  | serosv models                                     |
| newdata | data.frame with age column to generate prediction |
| ...     | arbitrary argument                                |

**Value**

prediction output

**See Also**

[stats::predict.glm()] for more information on the predict function

---

rubella\_mumps\_uk      *Rubella - Mumps data from the UK (aggregated)*

---

**Description**

Rubella - Mumps data from the UK (aggregated)

**Usage**

rubella\_mumps\_uk

**Format**

A data frame with 5 variables:

**age** Age group

**NN** Number of individuals negative to rubella and mumps

**NP** Number of individuals negative to rubella and positive to mumps

**PN** Number of individuals positive to rubella and negative to mumps

**PP** Number of individuals positive to rubella and mumps

**Source**

Morgan-Capner P, Wright J, Miller C L, Miller E. Surveillance of antibody to measles, mumps, and rubella by age. *British Medical Journal* 1988; 297 :770 [doi:10.1136/bmj.297.6651.770](https://doi.org/10.1136/bmj.297.6651.770)

---

rubella\_uk\_1986\_1987      *Rubella serological data from the UK in 1986 and 1987 (aggregated)*

---

**Description**

Prevalence of rubella in the UK, obtained from a large survey of prevalence of antibodies to both mumps and rubella viruses.

**Usage**

rubella\_uk\_1986\_1987

**Format**

A data frame with 3 variables:

**age** Midpoint of the age group (e.g. 1.5 = 1-2 years old, 2.5 = 2-3 years old)

**pos** Number of seropositive individuals

**tot** Total number of individuals surveyed

**Source**

Morgan-Capner P, Wright J, Miller C L, Miller E. Surveillance of antibody to measles, mumps, and rubella by age. *British Medical Journal* 1988; 297 :770 [doi:10.1136/bmj.297.6651.770](https://doi.org/10.1136/bmj.297.6651.770)

**Examples**

```
with(rubella_uk_1986_1987,
  plot(age, pos / tot,
    cex = 0.2 * sqrt(tot), pch = 16, xlab = "age", ylab = "seroprevalence",
    xlim = c(0, 45), ylim = c(0, 1)
  )
)
```

---

 set\_plot\_style

*Helper to adjust styling of a plot*


---

**Description**

Helper to adjust styling of a plot

**Usage**

```
set_plot_style(
  sero = "blueviolet",
  ci = "royalblue1",
  foi = "#fc0328",
  sero_line = "solid",
  foi_line = "dashed",
  xlabel = "Age"
)
```

**Arguments**

|           |                                      |
|-----------|--------------------------------------|
| sero      | color for seroprevalence line        |
| ci        | color for confidence interval        |
| foi       | color for force of infection line    |
| sero_line | linetype for seroprevalence line     |
| foi_line  | linetype for force of infection line |
| xlabel    | x label                              |

**Value**

list of updated aesthetic values

---

|                  |   |
|------------------|---|
| standardize_data | <i>Standardize raw serological test data for titer conversion</i> |
|------------------|---|

---

## Description

Validate and prepare raw serological test results for use with ‘to\_titer()’

## Usage

```
standardize_data(  
  df,  
  plate_id_col = "PLATE_ID",  
  id_col = "SAMPLE_ID",  
  result_col = "RESULT",  
  dilution_fct_col = "DILUTION_FACTORS",  
  antitoxin_label = "Anti_toxin",  
  negative_col = "^NEGATIVE_*" )
```

## Arguments

|                  |   |
|------------------|---|
| df               | data.frame with columns for plate id, sample id, result, dilution factor, and (optionally) negative controls                    |
| plate_id_col     | name of the column storing plates id  |
| id_col           | name of the column storing sample id  |
| result_col       | name of the column storing result   |
| dilution_fct_col | name of the column storing dilution factors   |
| antitoxin_label  | how antitoxin is label in the sample id column  |
| negative_col     | regex for columns for negative controls, assumed to be a label followed by the dilution factor (e.g. NEGATIVE_50, NEGATIVE_100) |

## Value

a standardized data.frame that can be passed to ‘to\_titer()’

---

|                 |  |
|-----------------|--|
| tb_nl_1966_1973 | <i>Tuberculosis serological data from the Netherlands 1966-1973 (aggregated)</i> |
|-----------------|--|

---

### Description

A study of tuberculosis conducted in the Netherlands. Schoolchildren, aged between 6 and 18 years, were tested using the tuberculin skin test.

### Usage

```
tb_nl_1966_1973
```

### Format

A data frame with 5 variables:

**age** Age group

**pos** Number of seropositive individuals

**tot** Total number of individuals surveyed

**gender** Gender of cohort (unsure what 0 and 1 means)

**birthyr** Birth year of cohort

### Source

Nagelkerke, N., Heisterkamp, S., Borgdorff, M., Broekmans, J. and Van Houwelingen, H. (1999), Semi-parametric estimation of age-time specific infection incidence from serial prevalence data. *Statist. Med.*, 18: 307-320. doi:[10.1002/\(SICI\)10970258\(19990215\)18:3<307::AID-SIM15>3.0.CO;2-Z](https://doi.org/10.1002/(SICI)10970258(19990215)18:3<307::AID-SIM15>3.0.CO;2-Z)

### Examples

```
with(tb_nl_1966_1973,  
  plot(age, pos / tot,  
        pch = 16, cex = 0.01 * sqrt(tot), xlab = "age",  
        ylab = "prevalence", xlim = c(6, 18)  
  )  
)
```

```
with(tb_nl_1966_1973,  
  plot(birthyr, pos / tot,  
        pch = 16, cex = 0.01 * sqrt(tot), xlab = "year", ylab = "prevalence"  
  )  
)
```

---

|          |   |
|----------|---|
| to_titer | <i>Convert assay readings to titers</i> |
|----------|---|

---

### Description

to\_titer() converts raw assay readings (e.g., OD, fluorescence intensity) to titer by fitting a calibrating model

### Usage

```
to_titer(
  df,
  model = "4PL",
  positive_threshold = NULL,
  ci = 0.95,
  negative_control = TRUE
)
```

### Arguments

|                    |   |
|--------------------|---|
| df                 | a standardized data.frame returned by 'standardize_data()'  |
| model              | either: <ol style="list-style-type: none"> <li>1. A string naming a built-in model (currently supported: "4PL"), or</li> <li>2. A named list with two functions: \$mod for curve fitting and \$quantify_ci for titer estimation with confidence intervals.</li> </ol> |
| positive_threshold | if not NULL, processed_data will have the serostatus labeled  |
| ci                 | confidence interval for the titer estimates (default is .95 i.e., 95% CI)   |
| negative_control   | if TRUE, output tibble will include the result for negative controls  |

### Value

a data.frame with 8 columns

|                     |  |
|---------------------|--|
| plate_id            | id of the plate  |
| data                | list of 'data.frame's containing the raw sample results from each plate                  |
| antitoxin_df        | list of 'data.frame's containing the raw results for antitoxins from each plate          |
| standard_curve_func | list of functions mapping from assay reading to titer for each plate                     |
| std_crv_midpoint    | midpoint of the standard curve, for qualitative analysis                                 |
| processed_data      | list of 'tibble's containing samples with titer estimates (lower, median, upper)         |
| negative_control    | list of 'tibble's containing negative control check results (if 'negative_control=TRUE') |

---

|                |                       |
|----------------|-----------------------|
| transform_data | <i>Aggregate data</i> |
|----------------|-----------------------|

---

**Description**

Generate a dataframe with 't', 'pos' and 'tot' columns from 't' and 'seropositive' vectors.

**Usage**

```
transform_data(data, stratum_col = "age", status_col = "status")
```

**Arguments**

|             |  |
|-------------|--|
| data        | a data frame with columns for age and serostatus     |
| stratum_col | name of the column to stratify by (default to "age") |
| status_col  | name of the column for serostatus                    |

**Value**

dataframe in aggregated format

**Examples**

```
df <- hcv_be_2006
hcv_df <- transform_data(df, stratum_col="dur", status_col="seropositive")
hcv_df
```

---

|                  |   |
|------------------|---|
| vzv_be_1999_2000 | <i>VZV serological data from Belgium (Flanders) from 1999-2000 (aggregated)</i> |
|------------------|---|

---

**Description**

Age-specific seroprevalence of VZV antibodies, assessed in Flanders (Belgium) between October 1999 and April 2000. This population was stratified by age in order to obtain approximately 100 observations per age group.

**Usage**

```
vzv_be_1999_2000
```

**Format**

A data frame with 3 variables:

**age** Age group

**pos** Number of seropositive individuals

**tot** Total number of individuals surveyed

**Source**

Thiry, N., Beutels, P., Shkedy, Z. et al. The seroepidemiology of primary varicella-zoster virus infection in Flanders (Belgium). *Eur J Pediatr* 161, 588-593 (2002). doi:[10.1007/s0043100210532](https://doi.org/10.1007/s0043100210532)

**Examples**

```
with(vzv_be_1999_2000,
  plot(age, pos / tot,
    cex = 0.1 * sqrt(tot), pch = 19, xlab = "age", ylab = "seroprevalence",
    xlim = c(0, 45), ylim = c(0, 1)
  )
)
```

---

vzv\_be\_2001\_2003

*VZV serological data from Belgium from 2001-2003 (line listing)*


---

**Description**

The survey is the same as the one used to study the seroprevalence of parvovirus B19 in Belgium, as described above.

**Usage**

```
vzv_be_2001_2003
```

**Format**

A data frame with 4 variables:

**age** Age of individual

**seropositive** If the individual is seropositive or not

**gender** Gender of individual

**VZVmIUml** VZV milli international units per ml

**Source**

MOSSONG, J., N. HENS, V. FRIEDERICHS, I. DAVIDKIN, M. BROMAN, B. LITWINSKA, J. SIENNICKA, et al. "Parvovirus B19 Infection in Five European Countries: Seroepidemiology, Force of Infection and Maternal Risk of Infection." *Epidemiology and Infection* 136, no. 8 (2008): 1059-68. doi:[10.1017/S0950268807009661](https://doi.org/10.1017/S0950268807009661)

**Examples**

```

library(dplyr)
df <- vzv_be_2001_2003 %>%
  mutate(age = round(age)) %>%
  group_by(age) %>%
  summarise(pos = sum(seropositive), tot = n())
plot(df$age, df$pos / df$tot,
     cex = 0.1 * sqrt(df$tot), pch = 19, xlab = "age", ylab = "seroprevalence",
     xlim = c(0, 45), ylim = c(0, 1)
)

```

---

vzv\_parvo\_be

*VZV and Parvovirus B19 serological data in Belgium (line listing)*


---

**Description**

VZV and Parvovirus B19 serological data in Belgium (line listing)

**Usage**

```
vzv_parvo_be
```

**Format**

A data frame with 7 variables:

**id** ID of individual

**age** Age of individual

**gender** Gender of individual

**parvouml** Parvo B19 antibody units per ml

**parvo\_res** If an individual is positive for parvovirus B19

**VZVmUIml** VZV milli international units per ml

**vzv\_res** If an individual is positive for VZV

**Source**

MOSSONG, J., N. HENS, V. FRIEDERICHS, I. DAVIDKIN, M. BROMAN, B. LITWINSKA, J. SIENNICKA, et al. "Parvovirus B19 Infection in Five European Countries: Seroepidemiology, Force of Infection and Maternal Risk of Infection." *Epidemiology and Infection* 136, no. 8 (2008): 1059-68. doi:[10.1017/S0950268807009661](https://doi.org/10.1017/S0950268807009661)

---

|               |                           |
|---------------|---------------------------|
| weibull_model | <i>The Weibull model.</i> |
|---------------|---------------------------|

---

### Description

Model seroprevalence as a function of duration since vaccination using the Weibull model, where the force of infection is assumed to vary monotonically with duration.

### Usage

```
weibull_model(
  data,
  t_lab = "t",
  pos_col = "pos",
  tot_col = "tot",
  status_col = "status"
)
```

### Arguments

|            |  |
|------------|--|
| data       | the input data frame, must either have columns for ‘t’, ‘pos’, ‘tot’ (for aggregated data) OR ‘t’, ‘status’ (for linelisting data) |
| t_lab      | name of the ‘t’ column (default t_lab="t").  |
| pos_col    | name of the ‘pos’ column (default pos_col="pos").  |
| tot_col    | name of the ‘tot’ column (default tot_col="tot").  |
| status_col | name of the ‘status’ column (default status_col="status").   |

### Details

For a Weibull model, the prevalence is given by

$$\pi(d) = 1 - e^{-\beta_0 d^{\beta_1}}$$

Where  $d$  is exposure time (difference between age of vaccination and age at test)

Which implies the force of infection to be the monotonic function

$$\lambda(d) = \beta_0 \beta_1 d^{\beta_1 - 1}$$

Refer to section 6.1.2. of the the book by Hens et al. (2012) for further details.

### Value

list of class weibull\_model with the following items

|          |   |
|----------|---|
| datatype | type of datatype used for model fitting (aggregated or linelisting) |
| df       | the dataframe used for fitting the model                            |
| info     | fitted "glm" object   |
| sp       | seroprevalence  |
| foi      | force of infection  |

**References**

Hens, Niel, Ziv Shkedy, Marc Aerts, Christel Faes, Pierre Van Damme, and Philippe Beutels. 2012. Modeling Infectious Disease Parameters Based on Serological and Social Contact Data: A Modern Statistical Perspective. *statistics for Biology and Health*. Springer New York. doi:[10.1007/9781-461440727](https://doi.org/10.1007/9781-461440727).

**See Also**

[stats::glm()] for more information on the fitted "glm" object

**Examples**

```
df <- hcv_be_2006[order(hcv_be_2006$dur), ]
df$t <- df$dur
df$status <- df$seropositive
model <- weibull_model(df, t_lab="dur", status_col="seropositive")
plot(model)
```

# Index

## \* datasets

- hav\_be\_1993\_1994, 19
- hav\_be\_2002, 20
- hav\_bg\_1964, 21
- hbv\_ru\_1999, 22
- hcv\_be\_2006, 23
- hierarchical\_bayesian\_model, 24
  
- lp\_model, 26
  
- mixture\_model, 28
- mumps\_uk\_1986\_1987, 30
  
- parvob19\_be\_2001\_2003, 30
- parvob19\_ew\_1996, 31
- parvob19\_fi\_1997\_1998, 32
- parvob19\_it\_2003\_2004, 33
- parvob19\_pl\_1995\_2004, 34
- rubella\_mumps\_uk, 53
- rubella\_uk\_1986\_1987, 53
- tb\_nl\_1966\_1973, 56
- vzv\_be\_1999\_2000, 58
- vzv\_be\_2001\_2003, 59
- vzv\_parvo\_be, 60
  
- hav\_be\_1993\_1994, 19
- hav\_be\_2002, 20
- hav\_bg\_1964, 21
- hbv\_ru\_1999, 22
- hcv\_be\_2006, 23
- hierarchical\_bayesian\_model, 24
  
- lp\_model, 26
  
- mixture\_model, 28
- mumps\_uk\_1986\_1987, 30
  
- parvob19\_be\_2001\_2003, 30
- parvob19\_ew\_1996, 31
- parvob19\_fi\_1997\_1998, 32
- parvob19\_it\_2003\_2004, 33
- parvob19\_pl\_1995\_2004, 34
- pava, 35
- penalized\_spline\_model, 36
- plot.age\_time\_model, 38
- plot.estimate\_from\_mixture, 39
- plot.farrington\_model, 40
- plot.fp\_model, 40
- plot.hierarchical\_bayesian\_model, 41
- plot.lp\_model, 41
- plot.mixture\_model, 42
- plot.penalized\_spline\_model, 42
- plot.polynomial\_model, 43
- plot.weibull\_model, 43
- plot\_corrected\_prev, 44
- plot\_gcv, 44
- plot\_standard\_curve, 45
- plot\_titer\_qc, 46
- polynomial\_model, 46
- predict.age\_time\_model, 48
- predict.farrington\_model, 49
- predict.fp\_model, 49
- predict.hierarchical\_bayesian\_model, 50
- predict.lp\_model, 50

add\_thresholds, 5

age\_time\_model, 5

compare\_models, 6

compute\_ci.age\_time\_model, 7

compute\_ci.default, 8

compute\_ci.fp\_model, 9

compute\_ci.hierarchical\_bayesian\_model, 9

compute\_ci.lp\_model, 10

compute\_ci.mixture\_model, 10

compute\_ci.penalized\_spline\_model, 11

compute\_ci.weibull\_model, 11

correct\_prevalence, 12

est\_foi, 15

estimate\_from\_mixture, 13

farrington\_model, 15

find\_best\_fp\_powers, 17

fp\_model, 18

`predict.penalized_spline_model`, [51](#)  
`predict.polynomial_model`, [51](#)  
`predict.weibull_model`, [52](#)

`rubella_mumps_uk`, [53](#)  
`rubella_uk_1986_1987`, [53](#)

`serosv` (`serosv`-package), [4](#)  
`serosv`-package, [4](#)  
`set_plot_style`, [54](#)  
`standardize_data`, [55](#)

`tb_nl_1966_1973`, [56](#)  
`to_titer`, [57](#)  
`transform_data`, [58](#)

`vzv_be_1999_2000`, [58](#)  
`vzv_be_2001_2003`, [59](#)  
`vzv_parvo_be`, [60](#)

`weibull_model`, [61](#)