Package: EpiPvr (via r-universe)

May 29, 2026

Title Estimating Plant Pathogen Epidemiology Parameters from Laboratory Assays

Version 0.0.1

Description Provides functions for estimating plant pathogen parameters from access period (AP) experiments. Separate functions are implemented for semi-persistently transmitted (SPT) and persistently transmitted (PT) pathogens. The common AP experiment exposes insect cohorts to infected source plants, healthy test plants, and intermediate plants (for PT pathogens). The package allows estimation of acquisition and inoculation rates during feeding, recovery rates, and latent progression rates (for PT pathogens). Additional functions support inference of epidemic risk from pathogen and local parameters, and also simulate AP experiment data. The functions implement probability models for epidemiological analysis, as derived in Donnelly et al. (2025), <doi:10.32942/X29K9P>. These models were originally implemented in the 'EpiPv' 'GitHub' package.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

Biarch true

Depends R (>= 3.5)

Imports methods, stats, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), rstantools (>= 2.5.0), posterior

Suggests bayesplot, knitr, rmarkdown, testthat (>= 3.0.0), ggplot2

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

VignetteBuilder knitr

URL https://CRAN.R-project.org/package=EpiPvr

LazyData true

NeedsCompilation yes

Author Ruairi Donnelly [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0001-7642-0317>)

Maintainer Ruairi Donnelly <rd501@cam.ac.uk>

Config/pak/sysreqs make

Repository https://cran.r-universe.dev

Date/Publication 2025-10-01 07:30:14 UTC

RemoteUrl https://github.com/cran/EpiPvr

RemoteRef HEAD

RemoteSha 169bac7683165c85621d35d7d51608a2a2a8ecce

Contents

`AP_assay_simulator` *Main function for AP assay simulation*

Description

This function simulates the assay data of an AP experiment (calling the helper function AP_insect_simulator.r to simulate on a per insect basis).

Usage

```
AP_assay_simulator(
  assayinput_in,
  fixedComponent_in,
  WF_in,
  smarkpams_in,
  isVerbose = 0,
  virusTypeIn
)
```

Arguments

assayinput_in	Numeric array with 3 rows (required): [1,] assay variable durations, [2,] reps per duration, [3,] zero entries for infected test plants (to be populated with simulated values).
fixedComponent_in	Numeric vector of 3 fixed durations in the order AAP, LAP, IAP; nb. one of these 3 will not be relevant for the sub-assay as it is varied in the assay (code -1) e.g. if AAP sub-assay [1] is -1, if LAP sub-assay [2] is -1, if IAP sub-assay [3] is -1 (required).
WF_in	Number of insect vectors in cohort (required).
smarkpams_in	Numeric vector: 4 elements representing virus rate parameters (hr^-1) (required): [1], acquisition rate; [2], inoculation rate; [3], latent progression rate WITH code -1 for SPT (no significant latency for SPT virus); [4], insect recovery rate.
isVerbose	Binary (optional, default = 0). If 1, prints information from nested functions.
virusTypeIn	Character. Plant virus type, either "SPT" or "PT" (required).

Value

A numeric array with 3 rows corresponding to assay structure and outcome: The first row specifies the assay variable durations, the second the reps per duration, the third the simulated number of infected test plants.

Examples

```
assay1=AP_assay_simulator(
  assayinput_in=rbind(c(6,7,8),rep(10,3),rep(0,3)),
  fixedComponent_in=c(10,-1,10),WF_in=5,
  smarkpams_in=rep(0.1,4),isVerbose=0,virusTypeIn='PT'
)
```

---

ap_data_sim_PT *Simulated access period experiment Data for a hypothetical PT virus*

Description

This dataset contains simulated AP assay results for a PT virus (AAP LAP and IAP sub-assays)

Usage

```
ap_data_sim_PT
```

Format

A List that is a collection of 6 data objects and 4 attributes

d_AAP Numeric 3 x n array for acquistion access sub-assay: variable duration; reps per duration, no. infected test plants per duration

d_LAP Numeric 3 x n array for latent access sub-assay: variable duration; reps per duration, no. infected test plants per duration

d_IAP Numeric 3 x n array for inoculation access sub-assay: variable duration; reps per duration, no. infected test plants per duration

d_durations Numeric 3 x 3 assay of fixed durations (FD): row 1, FDs for acquistion access sub-assay; row 2, FDs latent access sub-assay; row 3, FDs inoculation access sub-assay. nb diagonal entries must be '-1' indicating redundancy since these elements correspond to the variable durations

d_vectorspp Numeric number of insects per plant

d_pathogen Character representing underlying virus

#'

"alpha" Numeric: acquisition rate (h^-1); used to generate the data or '-1' if unknown

"beta" Numeric: inoculation rate (h^-1); used to generate the data or '-1' if unknown

"gamma" Numeric: latent progression rate (h^-1); used to generate the data or '-1' if unknown

"mu" Numeric: insect recovery rate (h^-1); used to generate the data or '-1' if unknown

Source

Simulated using `AP_assay_simulator()`

`ap_data_sim_SPT` *Simulated access period experiment Data for a hypothetical SPT virus*

Description

This dataset contains simulated AP assay results for an SPT virus (AAP and IAP sub-assays)

Usage

```
ap_data_sim_SPT
```

Format

A List that is a collection of 5 data objects and 3 attributes

d_AAP Numeric 3 x n array for acquistion access sub-assay: variable duration; reps per duration, no. infected test plants per duration

d_IAP Numeric 3 x n array for inoculation access sub-assay: variable duration; reps per duration, no. infected test plants per duration

d_durations Numeric 2 x 2 assay of fixed durations (FD): row 1, FDs for acquistion access sub-assay; row 2, FDs acquistion access sub-assay. nb diagonal entries must be '-1' indicating redundancy since these elements correspond to the variable durations

d_vectorspp Numeric number of insects per plant

d_pathogen Character representing underlying virus

#'

"alpha" Numeric: acquisition rate (h^-1); used to generate the data or '-1' if unknown

"beta" Numeric: inoculation rate (h^-1); used to generate the data or '-1' if unknown

"mu" Numeric: insect recovery rate (h^-1); used to generate the data or '-1' if unknown

Source

Simulated using `AP_assay_simulator()`

`calculate_epidemic_probability` *Main function for calculating epidemic probability*

Description

This function calculates epidemic probability from viral and local parameters according to inoculum state (calling the helper function solveInoculumStatesBP.r to calculate transition and fertility matrices)

Usage

```
calculate_epidemic_probability(
  numberInsects,
  start_intervl = (1/10),
  localParameters,
  virusParameters,
  thresh = 10^(-7)
)
```

Arguments

`numberInsects`	Numeric: number of insect vectors per plant (required).
`start_intervl`	Numeric: initial guess for expected time in hours until next event (optional, default = 0.1). This is fine-tuned from within the function. Needed to calculate efficient time step i.e. largest value such that sum of probabilities of individual events < 1.
`localParameters`	A numeric vector of length 5 corresponding to event rates (day^-1) (required): The first element specifies the Vector dispersal rate, the second the Roguing rate, the third the Vector mortality rate, the forth the Harvesting rate, the fifth the Plant latent progression rate
`virusParameters`	Numeric vector of length 3 specifying virus parameters (day^-1) (required): `[1]` Acquisition rate. `[2]` Inoculation rate. `[3]` Insect recovery rate.
`thresh`	Numeric: equilibrium condition for extinction probability (optional, default = 10^(-7)). If no inoculum state changes by more than thresh in a timestep then function assumes extinction probability has stabilised.

Value

Numeric vector of length n: epidemic probability for each of n inoculum states (where n=(numberInsects+1)*3-1).

Preprint reference

The models implemented in this function follow Donnelly et al. (2025, preprint), originally implemented in the EpiPv GitHub package.

References

Donnelly, R., Tankam, I. & Gilligan, C. (2025). "Plant pathogen profiling with the EpiPv package." EcoEvoRxiv, doi:10.32942/X29K9P.

When available, please cite the published version.

Examples

```
qm_out=calculate_epidemic_probability(
    numberInsects=3,start_intervl=(1/10),
    localParameters=rep(0.1,5),
    virusParameters=rep(0.1,3),thresh=10^(-7)
)
```

`estimate_virus_parameters_PT` *Estimate PT virus parameters using Bayesian inference with rstan*

Description

Runs MCMC sampling using a precompiled Stan model for PT plant virus. Analyses PT access period data estimating 5 parameters: (`alpha[1]`, `beta[1]`, `gamma[1]`,`mu[1]`,`bd[1]` - acquisition, inoculation, latent progression, insect recovery and insect lab mortality rates, respectively).

Usage

```
estimate_virus_parameters_PT(
  data,
  D_LSin = 50,
  D_numPtsPdin = 1,
  mcmcOptions = c(500, 1000),
  numChainsIn = 4,
  mc.parallel = 0
)
```

Arguments

`data`	A list containing the AP experiment data for Stan (required).
`D_LSin`	Upper bound on insect vector lifespan in days sets the vector survival discretization (optional, default = 50).
`D_numPtsPdin`	Number of points per day of insect vector sets the vector survival discretization (optional, default = 1).
`mcmcOptions`	A numeric vector of length 2: The first element specifies the number of warm-up iterations (optional, default = 500), and the second element specifies the total number of iterations (optional, default = 1000).
`numChainsIn`	Numeric: number of Markov chains (optional, default = 4).
`mc.parallel`	Binary: whether to use parallelisation for sampling (optional, default = 0, i.e. 1 core only).

Details

Interpreting model output - check stability first

Warnings will be printed to the screen if there are issues with model fitting. **Do not suppress warnings** (e.g., via `suppressWarnings()`), as they contain essential information about potential convergence problems.

Before interpreting any model results, always check the following **core diagnostics**:

1. **R-hat** - Should be close to 1 for all parameters; larger values indicate non-convergence (Stan R-hat documentation).

2. **Effective Sample Size (n_eff)** - Very low values suggest high autocorrelation or insufficient sampling (Stan ESS documentation).
3. **Divergent transitions** - Count should be 0; any non-zero count requires investigation (Stan divergences documentation).
4. **Forward Simulation from Fitted Model (simulated_data)** - Presented conveniently for plotting to assess posterior predictive fit.
   - `array3`: AAP forward simulation 95% credible intervals and original data
   - `array4`: LAP forward simulation 95% credible intervals and original data
   - `array5`: IAP forward simulation 95% credible intervals and original data (Stan posterior predictive checks).

If any of these core diagnostics fail, the model fit may not be trustworthy.

Additional diagnostics for troubleshooting

These are useful when core checks indicate problems, or for deeper exploration of model behaviour:

- **Bayesian R^2** (`r2_bayesA`, `r2_bayesL`, `r2_bayesI`) - Measures explanatory power; low values suggest poor fit to the data (i.e., 3 values for AAP assay, LAP assay,IAP assay).
- **Max treedepth exceeded** - Number of times the sampler hit the maximum tree depth; should be 0 (Stan max treedepth documentation).
- **EBFMI** (Energy Bayesian Fraction of Missing Information) - Low values indicate poor exploration of the posterior.

See the package vignette for worked examples of checking and interpreting these diagnostics.

Preprint reference

The models implemented in this function follow Donnelly et al. (2025, preprint), originally implemented in the EpiPv GitHub package.

Value

A list with seven elements:

array0 MCMC chains for estimated parameters (`alpha[1]`, `beta[1]`, `gamma[1]/mu[1]`,`bd[1]` - and additionally `lp__` for reference only).

array1 MCMC chains for estimated parameters (`alpha[1]`, `beta[1]`, `gamma[1]`,`mu[1]`).

array2 summary_stats (rhat, ess_bulk, ess_tail). `[1]`: R-hat statistic for convergence (should be close to 1); `[2-3]`: statistics for n-eff.

array3 Validation dataset: AAP sub-assay input test plant data, forward simulated 2.5th percentile, forward simulated 97.5th percentile.

array4 Validation dataset: LAP sub-assay input test plant data, forward simulated 2.5th percentile, forward simulated 97.5th percentile.

array5 Validation dataset: IAP sub-assay input test plant data, forward simulated 2.5th percentile, forward simulated 97.5th percentile.

array6 Mean and sd Bayesian R-squared values for model fit assessment, for AAP, LAP and IAP assays.

converge_results A list containing 2 sampler diagnostics, the number of overall divergent transitions and if maximum tree depth has been exceeded. See also screen print for acceptability of energy Bayesian fraction of missing information (E-BFMI).

References

Donnelly, R., Tankam, I. & Gilligan, C. (2025). "Plant pathogen profiling with the EpiPv package." EcoEvoRxiv, doi:10.32942/X29K9P.

When available, please cite the published version.

Examples

```
data("ap_data_sim_PT", package = "EpiPvr")
# run low warm-up and iterations (mcmcOptions) for quick example only
EVPT_pub=estimate_virus_parameters_PT(
  data=ap_data_sim_PT,D_LSin=5,D_numPtsPdin=1,
  mcmcOptions=c(25,50),numChainsIn=1,mc.parallel=0
)
```

`estimate_virus_parameters_SPT`

Estimate SPT virus parameters using Bayesian inference with rstan

---

Description

Runs MCMC sampling using a precompiled Stan model for SPT plant virus. Analyses SPT access period data directly estimating 4 parameters: (`c1[1]`, `c3[1]`, `c2[1]`,`bD[1]`, the first 3 are composite and the 4th corresponds to insect lab mortality rate). The composite parameters can be unpacked into `al[1]`, `be[1]`, `mu[1]` - acquisition, inoculation, and insect recovery rates.

Usage

```
estimate_virus_parameters_SPT(
  data,
  D_LSin = 50,
  D_numPtsPdin = 1,
  mcmcOptions = c(500, 1000),
  numChainsIn = 4,
  mc.parallel = 0
)
```

Arguments

`data`	A list containing the AP experiment data for Stan (required).
`D_LSin`	Upper bound on insect vector lifespan in days, sets the vector survival discretisation (optional, default = 50).
`D_numPtsPdin`	Number of points per day of insect vector lifespan, sets the vector survival discretisation (optional, default = 1).
`mcmcOptions`	A numeric vector of length 2: The first element specifies the number of warm-up iterations (optional, default = 500), and the second element specifies the number of post-warm-up iterations (optional, default = 1000).

`numChainsIn`	Numeric: number of Markov chains (optional, default = 4).
`mc.parallel`	Binary: whether to use parallelisation for sampling (optional, default = 0, i.e. 1 core only).

Details

Interpreting model output - check stability first

Warnings will be printed to the screen if there are issues with model fitting. **Do not suppress warnings** (e.g., via `suppressWarnings()`), as they contain essential information about potential convergence problems.

Before interpreting any model results, always check the following **core diagnostics**:

1. **R-hat** - Should be close to 1 for all parameters; larger values indicate non-convergence (Stan R-hat documentation).
2. **Effective Sample Size (n_eff)** - Very low values suggest high autocorrelation or insufficient sampling (Stan ESS documentation).
3. **Divergent transitions** - Count should be 0; any non-zero count requires investigation (Stan divergences documentation).
4. **Posterior predictive fit (simulated_data)** - Compare model-simulated data with the observed data:
   - `array3`: AAP forward simulation 95% credible intervals and original data
   - `array4`: IAP forward simulation 95% credible intervals and original data (Stan posterior predictive checks).

Additional diagnostics for troubleshooting

These are useful when core checks indicate problems, or for deeper exploration of model behaviour:

- **Bayesian R^2** (`r2_bayesA`, `r2_bayesI`) - Measures explanatory power; low values suggest poor fit to the data (i.e., 2 values for AAP assay, IAP assay).
- **Max treedepth exceeded** - Number of times the sampler hit the maximum tree depth; should be 0 (Stan max treedepth documentation).
- **EBFMI** (Energy Bayesian Fraction of Missing Information) - Low values indicate poor exploration of the posterior.

See the package vignette for worked examples of checking and interpreting these diagnostics.

Preprint reference

The models implemented in this function follow Donnelly et al. (2025, preprint), originally implemented in the EpiPv GitHub package.

Value

A list with seven elements:

array0 First set of MCMC chains for estimated parameters (`c1[1]`, `c3[1]`, `c2[1]`,`bD[1]` - and additionally `lp__` for reference only).

array1 Second set of MCMC chains for virus parameters (`al[1]`, `be[1]`, `mu[1]`).

array2 summary_stats (rhat, ess_bulk, ess_tail). `[1]`: R-hat statistic for convergence (should be close to 1); `[2-3]`: statistics for n-eff.

array3 Validation list, AAP input test plant infection data, forward simulated 2.5th percentile, forward simulated 97.5th percentile.

array4 Validation list, IAP input test plant infection data, forward simulated 2.5th percentile, forward simulated 97.5th percentile.

array5 Mean and sd Bayesian R-squared values for model fit assessment, for AAP and IAP assays.

converge_results A list containing key sampler diagnostics: divergent transitions, maximum tree depth exceedances, See also screen print for acceptability of energy Bayesian fraction of missing information (E-BFMI).).

References

Donnelly, R., Tankam, I. & Gilligan, C. (2025). "Plant pathogen profiling with the EpiPv package." EcoEvoRxiv, doi:10.32942/X29K9P.

When available, please cite the published version.

Examples

```
data("ap_data_sim_SPT", package = "EpiPvr")
# run low warm-up and iterations (mcmcOptions) for quick example only
EVSPT_pub=estimate_virus_parameters_SPT(
  data=ap_data_sim_SPT,D_LSin=5,D_numPtsPdin=1,
  mcmcOptions=c(25,50),numChainsIn=1,mc.parallel=0
)
```

Index