Package ‘vmsae’

May 9, 2025

Title Variational Multivariate Spatial Small Area Estimation

Version 0.1.0

Maintainer Zhenhua Wang <zhenhua.wang@missouri.edu>

Description Variational Autoencoded Multivariate Spatial Fay-Herriot models are designed to efficiently estimate population parameters in small area estimation. This package implements the variational generalized multivariate spatial Fay-Herriot model (VGMSFH) using 'NumPyro' and 'PyTorch' backends, as demonstrated by Wang, Parker, and Holan (2025) <doi:10.48550/arXiv.2503.14710>. The 'vmsae' package provides utility functions to load weights of the pretrained variational autoencoders (VAEs) as well as tools to train custom VAEs tailored to users specific applications.

Depends R (>= 3.5.0)

Imports dplyr, ggplot2, gridExtra, sf, tidyr, reticulate, methods, rlang

URL https://github.com/zhenhua-wang/vmsae

BugReports https://github.com/zhenhua-wang/vmsae/issues

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Zhenhua Wang [aut, cre],
Paul A. Parker [aut, res],
Scott H. Holan [aut, res]

Repository CRAN

Date/Publication 2025-05-09 10:30:02 UTC

Contents

`coef,VGMSFH-method` *Extract Coefficients from a VGMSFH Object*

Description

This method extracts posterior mean estimates of model coefficients from a `VGMSFH` object. It can return either fixed effect coefficients or spatial random effects.

Usage

```
## S4 method for signature 'VGMSFH'
coef(object, var_idx = 1, type = "fixed")
```

Arguments

`object` An object of class `VGMSFH`.

`var_idx` Integer. The index of the variable of interest (for multivariate models). Default is 1.

`type` Character. The type of coefficient to extract. Options are:

- `"fixed"` – extract the posterior mean of fixed effect coefficients (default).
- `"spatial"` – extract the posterior mean of spatial random effects.

Value

A numeric vector of posterior means for the selected coefficient type.

Examples

```
library(vmsae)
example_model <- readRDS(system.file("example", "example_model.Rds", package = "vmsae"))
coef(example_model)  # Get fixed effect coefficients
coef(example_model, type = "spatial")  # Get spatial random effects
```

confint,VGMSFH-method *Compute Credible Intervals for VGMSFH Parameters*

Description

This method computes 95\

Usage

```
## S4 method for signature 'VGMSFH'
confint(object, var_idx = 1, field = "yhat_samples")
```

Arguments

`object`	An object of class `VGMSFH`.
`var_idx`	Integer. The index of the variable of interest (for multivariate models). Default is `1`.
`field`	Character. The name of the slot to summarize (e.g., `"yhat_samples"`, `"beta_samples"`, `"spatial_samples"`). Default is `"yhat_samples"`.

Details

The function extracts posterior samples for the specified variable and then computes quantiles to form 95\

Value

A data frame with two columns:

- `lower`: the 2.5\
- `upper`: the 97.5\

Examples

```
library(vmsae)
example_model <- readRDS(system.file("example", "example_model.Rds", package = "vmsae"))
confint(example_model)  # Get credible intervals for predicted values
confint(example_model, field = "beta_samples")  # For fixed effects
```

Decoder-class *Decoder S4 Class*

Description

An S4 class to represent a neural network decoder, used for emulating spatial priors. The class includes parameters for input and output weight matrices and biases, as well as region identifiers.

Slots

`GEOID` A character vector of region or area identifiers.

`W_in` An array representing the input weight matrix of the decoder.

`B_in` An array representing the input bias vector of the decoder.

`W_out` An array representing the output weight matrix of the decoder.

`B_out` An array representing the output bias vector of the decoder.

download_pretrained_vae *Download and Extract a Pretrained VAE Model*

Description

This function downloads a pretrained VAE model archive from Zenodo, extracts its contents into a specified directory, and removes the downloaded ZIP file after extraction.

Usage

```
download_pretrained_vae(model_name, save_dir, verbose = TRUE)
```

Arguments

`model_name`	Character. The name of the model file (without extension) to download. This should correspond to a `*model_name*.zip` file hosted on Zenodo (e.g., `"ca_county"`).
`save_dir`	Character. The local directory where the model should be saved and extracted.
`verbose`	Logical; if `TRUE` (default), prints progress and error messages.

Value

No return value, called for side effects

Examples

```
## Not run:
library(vmsae)
# this function is time consuming for the first run
install_environment()
load_environment()
download_pretrained_vae("mo_county", tempdir())

## End(Not run)
```

---

install_environment *Install python environment.*

Description

This function creates the vmsae python environment and installs required packages.

Usage

```
install_environment(envname = "vmsae")
```

Arguments

envname	Character. The name of the Python environment to create or update. Default is "vmsae".

Value

No return value, called for side effects

Examples

```
## Not run:
library(vmsae)
# this function is time consuming for the first run
install_environment()          # Install into default "vmsae" environment


# this step is time consuming for the first run
install_environment("custom")  # Install into a custom-named environment

## End(Not run)
```

load_environment *Load Python Environment and Source Model Modules*

Description

This function activates a specified Python virtual environment and sources Python modules used by the **vmsae** package, including models and python scripts.

Usage

```
load_environment(envname = "vmsae")
```

Arguments

envname	Character. The name of the Python environment to create or update. Default is "vmsae".

Details

The function loads four Python scripts located in the package's py/ directory:

- vgmcar.py
- vae.py
- train_vae.py
- car_dataset.py

The environment must be created beforehand (e.g., using install_environment()), and must include all Python dependencies required by these modules.

Value

No return value, called for side effects

Examples

```
## Not run:
library(vmsae)

# this function is time consuming for the first run
install_environment()
load_environment()          # Load default "vmsae" environment

# this function is time consuming for the first run
install_environment("custom")
load_environment("custom") # Load custom virtual environment

## End(Not run)
```

load_vae *Load Pretrained VAE Decoder*

Description

Load a pretrained Variational Autoencoder (VAE) decoder from disk. This function reads the saved PyTorch model weights and corresponding GEOID list, and constructs a Decoder S4 object with the loaded parameters.

Usage

```
load_vae(model_name, save_dir = NULL)
```

Arguments

model_name	Character. The name of the trained VAE model (without .zip extensions).
save_dir	Character. The directory where the trained VAE model is saved. Defaults to the current directory if NULL.

Details

This function assumes the model was trained and saved using train_vae(), and that the decoder weights are stored in a file compatible with torch::load() (via reticulate). It extracts the decoder input/output weights and biases, along with region GEOIDs, and returns them as an S4 object of class Decoder.

Value

An object of class Decoder, containing the decoder weights and region identifiers.

Examples

```
## Not run:
library(vmsae)
# this function is time consuming for the first run
install_environment()
load_environment()
decoder <- load_vae(model_name = "mo_county")

## End(Not run)
```

plot,VGMSFH-method *Plot VGMSFH Result*

Description

This method plots spatial summaries of results from a VGMSFH object, including model estimates and comparisons with direct estimates.

Usage

```
## S4 method for signature 'VGMSFH'
plot(x, shp = NULL, var_idx = 1, type = "compare", verbose = TRUE)
```

Arguments

x	An object of class VGMSFH, containing posterior samples and direct estimates from the model.
shp	An sf object representing the spatial shapefile. If NULL, the function will automatically download a shapefile associated with the pretrained model.
var_idx	Integer. The index of the variable of interest (for multivariate models).
type	Character. The type of plot to generate. Options are: • "compare" – compare direct estimates and model-based estimates. • "estimate" – show the posterior mean and standard deviation of the model estimate.
verbose	Logical; if TRUE (default), prints error messages.

Details

The function provides spatial visualization of model results. It supports both univariate and multivariate response settings. When type = "compare", it generates side-by-side choropleth maps for the direct and model-based estimates. When type = "estimate", it plots the posterior mean and standard deviation of the VGMSFH model output.

If no shapefile is provided, a default geometry is loaded from the pretrained repository.

Value

A ggplot object. The plot is rendered to the active device.

Examples

```
library(vmsae)
library(sf)
example_model <- readRDS(system.file("example", "example_model.Rds", package = "vmsae"))
example_shp <- read_sf(system.file("example", "mo_county.shp", package = "vmsae"))
plot(example_model, shp = example_shp, type = "compare")
plot(example_model, shp = example_shp, type = "estimate", var_idx = 2)
```

summary,VGMSFH-method *Summarize VGMSFH Result*

Description

This method provides a summary of posterior samples from a VGMSFH object, including posterior means and credible intervals for a specified parameter field.

Usage

```
## S4 method for signature 'VGMSFH'
summary(object, var_idx = 1, field = "beta_samples")
```

Arguments

object	An object of class VGMSFH, containing posterior samples from the model.
var_idx	Integer. The index of the variable of interest (for multivariate models). Default is 1.
field	Character. The name of the slot in the VGMSFH object to summarize (e.g., "beta_samples", "spatial_samples", "yhat_samples"). Default is "beta_samples".

Details

This function extracts the posterior samples for the specified variable index, and combines it with confint() to compute credible intervals. The result is a compact summary table of central tendency and uncertainty.

Value

A data frame with columns:

- mean: Posterior mean,
- lower: Lower bound of the credible interval,
- upper: Upper bound of the credible interval.

Examples

```
library(vmsae)
example_model <- readRDS(system.file("example", "example_model.Rds", package = "vmsae"))
summary(example_model)  # Summary of beta_samples for variable 1
summary(example_model, var_idx = 2, field = "yhat_samples")
```

train_vae *Train VAE for CAR Prior*

Description

Trains a Variational Autoencoder (VAE) to learn the spatial structure implied by the Conditional Autoregressive (CAR) prior. The trained VAE parameters are saved and can later be used as a generator within Hamiltonian Monte Carlo (HMC) sampling.

Usage

```
train_vae(
  W,
  GEOID,
  model_name,
  save_dir,
  n_samples = 10000,
  batch_size = 256,
  epoch = 10000,
  lr_init = 0.001,
  lr_min = 1e-07,
  verbose = TRUE
)
```

Arguments

`W` Matrix. A proximity or adjacency matrix representing spatial relationships.

`GEOID` Character vector. Identifiers for spatial units (e.g., region or area codes).

`model_name` Character. The name of the trained VAE model.

`save_dir` Character. Directory to save the trained VAE model and associated metadata. Defaults to the current working directory.

`n_samples` Integer. Number of samples to draw from the prior for training. Default is `10000`.

`batch_size` Integer. Batch size for VAE training. Default is `256`.

`epoch` Integer. Number of training epochs. Default is `10000`.

`lr_init` Numeric. Initial learning rate. Default is `0.001`.

`lr_min` Numeric. Minimum learning rate at the final epoch. Default is `1e-7`.

`verbose` Logical; if `TRUE` (default), prints progress.

Details

The function requires a configured Python environment via the **reticulate** interface, with VAE training implemented in Python. It uses `py$train_vae()` defined in the sourced Python modules (see `load_environment`).

Value

A named list containing:

loss	Total training loss
RCL	Reconstruction error
KLD	Kullback–Leibler divergence

Examples

```
## Not run:
library(vmsae)
library(sf)
# this function is time consuming for the first run
install_environment()
load_environment()

acs_data <- read_sf(system.file("example", "mo_county.shp", package = "vmsae"))
W <- readRDS(system.file("example", "W.Rds", package = "vmsae"))

loss <- train_vae(W = W,
  GEOID = acs_data$GEOID,
  model_name = "test",
  save_dir = tempdir(),
  n_samples = 1000, # set to larger values in practice, e.g. 10000.
  batch_size = 256,
  epoch = 1000)     # set to larger values in practice, e.g. 10000.

## End(Not run)
```

VGMSFH-class *VGMSFH S4 Class*

Description

An S4 class to store results from the Variational Gaussian Markov Small Area Estimation with Fay-Herriot model (VGMSFH). This class holds the posterior samples for various model components as well as the original direct estimates.

Slots

model_name Character. The name of the trained VAE model.

direct_estimate Array. Direct estimates of parameters.

yhat_samples Array. Posterior samples of the estimated parameters.

spatial_samples Array. Posterior samples of the estimated spatial random effects.

beta_samples Array. Posterior samples of the fixed effect coefficients.

all_samples List. Posterior samples of all parameters in the VGMSFH model.

vgmsfh_numpyro *Run VGMSFH Using NumPyro*

Description

This function runs the Variational Generalized Multivariate Spatil Fay-Herriot model (VGMSFH) using NumPyro as the inference backend. It loads pretrained VAE decoder weights, prepares the data, and performs posterior sampling.

Usage

```
vgmsfh_numpyro(
  y,
  y_sigma,
  X,
  W,
  GEOID,
  model_name,
  save_dir = NULL,
  num_warmup = 1000,
  num_samples = 1000
)
```

Arguments

y	Matrix. Response variables (direct estimates).
y_sigma	Matrix. Reported standard deviations of the responses.
X	Matrix. Covariate matrix.
W	Matrix. Proximity or adjacency matrix defining spatial structure.
GEOID	Character vector. FIPS codes or other region identifiers used to match with the pretrained VAE model.
model_name	Character. The name of the pretrained VAE model.
save_dir	Character. The directory where the VAE model is stored. If NULL, a default pretrained model directory is used.
num_warmup	Integer. Number of warmup (burn-in) iterations. Default is 1000.
num_samples	Integer. Number of posterior samples to draw. Default is 1000.

Details

This function uses a pretrained VAE decoder to parameterize the CAR prior and enables scalable inference through NumPyro. It is suitable for both univariate and multivariate response modeling in spatial domains.

Value

An object of class VGMSFH, which contains:

- direct_estimate: the observed response data,
- yhat_samples: posterior samples of the latent population process,
- spatial_samples: posterior samples of spatial random effects (CAR),
- beta_samples: posterior samples of fixed effect coefficients,
- all_samples: a list containing all sampled parameters, including mu, delta, and other intermediate quantities.

References

Wang, Z., Parker, P. A., & Holan, S. H. (2025). Variational Autoencoded Multivariate Spatial Fay-Herriot Models. arXiv:2503.14710. https://arxiv.org/abs/2503.14710

Examples

```
## Not run:
library(sf)
library(vmsae)
# this function is time consuming for the first run
install_environment()
load_environment()

acs_data <- read_sf(system.file("example", "mo_county.shp", package = "vmsae"))
y <- readRDS(system.file("example", "y.Rds", package = "vmsae"))
y_sigma <- readRDS(system.file("example", "y_sigma.Rds", package = "vmsae"))
X <- readRDS(system.file("example", "X.Rds", package = "vmsae"))
W <- readRDS(system.file("example", "W.Rds", package = "vmsae"))

num_samples <- 1000 # set to larger values in practice, e.g. 10000.
model <- vgmsfh_numpyro(y, y_sigma, X, W,
  GEOID = acs_data$GEOID,
  model_name = "mo_county", save_dir = NULL,
  num_samples = num_samples, num_warmup = num_samples)
y_hat_np <- model@yhat_samples
y_hat_mean_np <- apply(y_hat_np, c(2, 3), mean)
y_hat_lower_np <- apply(y_hat_np, c(2, 3), quantile, 0.025)
y_hat_upper_np <- apply(y_hat_np, c(2, 3), quantile, 0.975)

plot(model, shp = acs_data, type = "compare", var_idx = 2)

## End(Not run)
```

Index